WEINVIEW

EasyBuilder8000
使用手册

目 录

第一章 关于EasyBuilder8000的安装

1.1 安装EasyBuilder8000 (或简称EB8000)

1. 软件来源

可由光盘获取，有特殊需求也可进入威纶通公司网站：http://www.weinview.cn 下载所有可用软件(包括简体中文、繁体中文及英文版本)及最新软件更新信息。

2. 计算机硬件要求(建议配置):

CPU：INTEL Pentium II以上等级

内存：64MB以上

硬盘：2.5GB以上，最少有10MB以上的磁盘空间

光驱：4倍速以上光驱一个

显示器：支持分辨率800 x 600以上的彩色显示器

鼠标键盘：各一个

以太网端口：[工程下载]/[工程上传]/[网络监测]时使用

打印机：一台

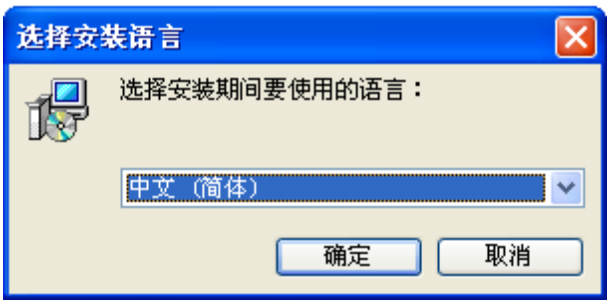
3. 操作系统

Windows XP/ Windows VISTA/WIN 7 均可。

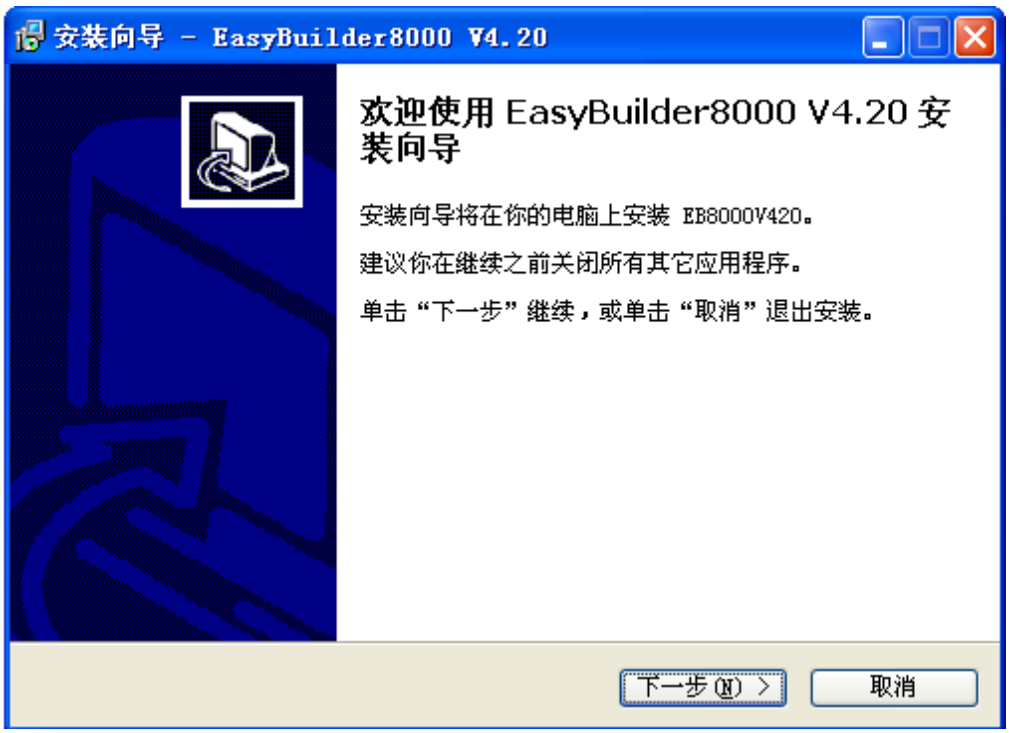
4. 安装步骤(以EB8000 V4.2.0简体中文版为例)

（1）将光盘放入光驱，计算机将会自动执行安装程序，或者您手动执行光盘根目录下的[Autorun.exe]文件，屏幕将显示安装程序如下：

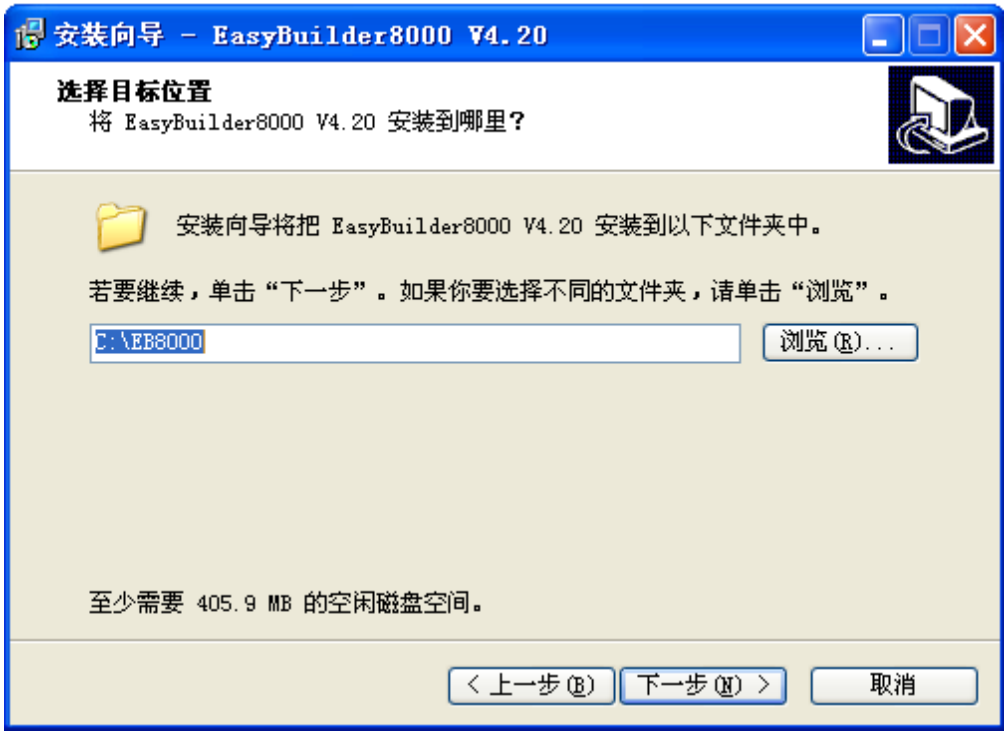
（2）点击“EB8000V4.20”按钮，屏幕显示如下，选择安装期间要使用的语言：



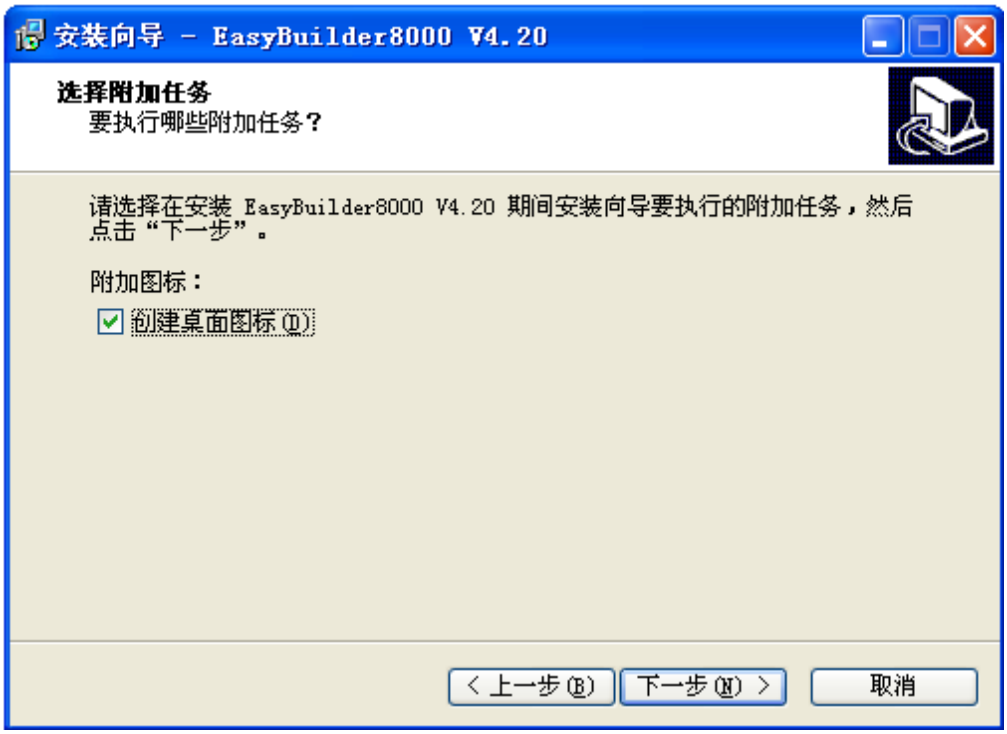
（3）点击“确定”，屏幕显示如下，此时根据向导提示，点击“下一步”：

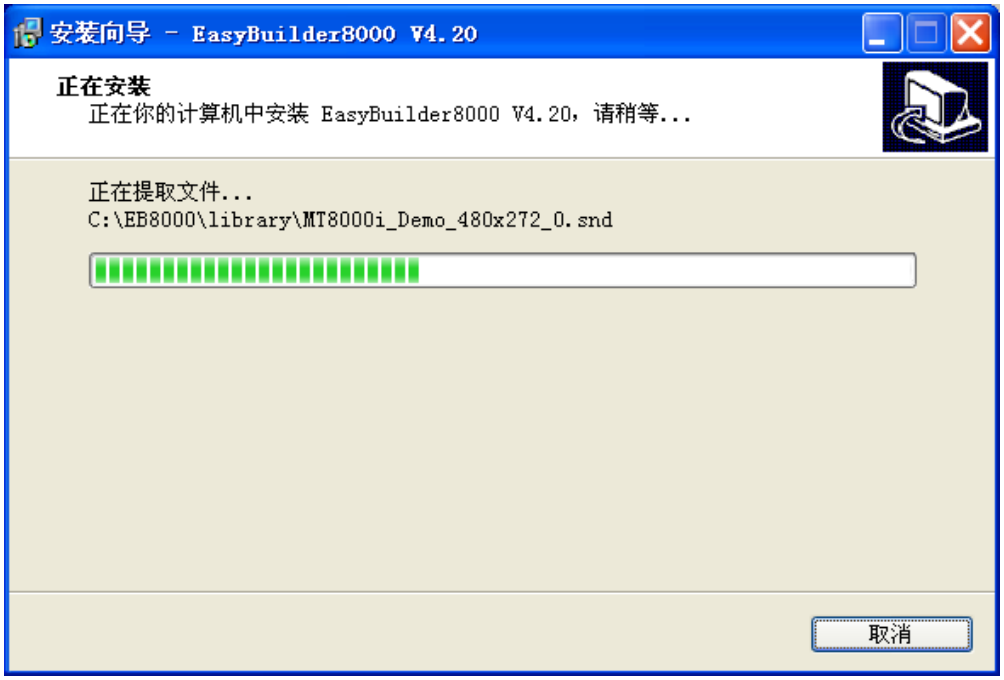


（4）选择软件安装的文件夹或选择默认路径，并点击“下一步”：



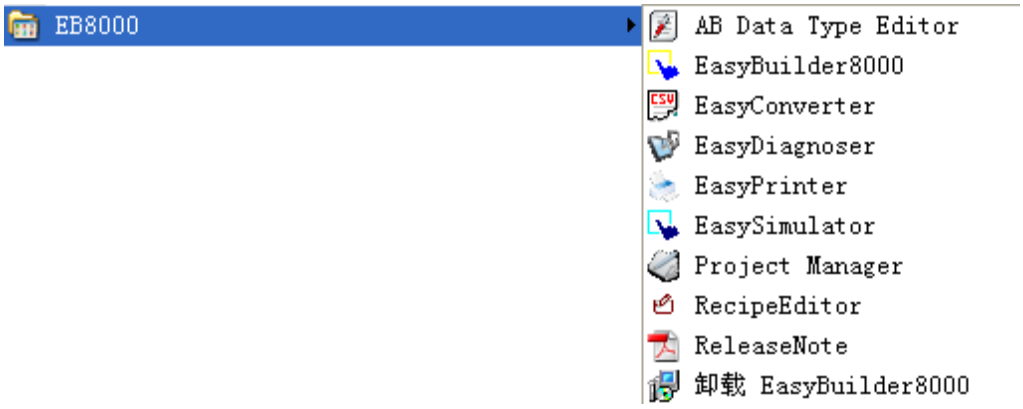
(5) 根据向导提示，点击“下一步”确认安装，在安装完成后点击“关闭”即完成安装程序。




安装向导 - EasyBuilder8000 V4.20
正在安装
正在你的计算机中安装 EasyBuilder8000 V4.20，请稍等...
正在提取文件...
C:\EB8000\library\MT8000i_Demo_480x272_0.snd
取消


安装向导 - EasyBuilder8000 V4.20
完成 EasyBuilder8000 V4.20 安装
安装向导已完成 EasyBuilder8000 V4.20 的安装。可以通过选择已安装的图标来运行应用程序。
点击“完成”退出安装。
完成(F)

（6）要执行程序时，可以从菜单[开始]/[程序]/[EB8000]下找到相对应的执行程序即可。

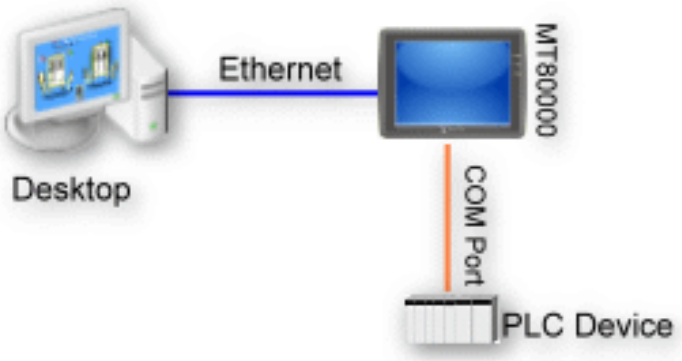


软件目录下各选项的含义如下：

EasyBuilder8000.exe	EB8000触控屏编辑软件
EasyConverter.exe	取样数据与事件记录文件转文件工具
EasyDiagnoser.exe	通讯诊断工具
EasyPrinter.exe	远程打印服务器软件
EasySimulator.exe	离线、在线模拟工具
Project Manager.exe	MT8000综合管理软件
RecipeEditor.exe	配方/扩展内存编辑器工具
ReleaseNote.pdf	版本及相关最新信息说明

1.2 系统连接图

MT8000系列触控屏之应用一般可按照如下连接方式



MT8000系列触控屏系统连接界面具备的有：

USB Host

支持各种USB接口的设备，如鼠标、键盘、U盘、打印机等。

USB Client

连接PC，提供项目上传及下载，包括工程文档、配方数据传送、事件记录，备份等。

以太网口

连接具网络通讯功能之设备，如PLC、笔记本电脑等等，透过网络做信息交流。

CF卡/SD卡接口

提供项目上传及下载，包括工程文档、配方数据传送、事件记录，备份等。

串口

所支持的COM端口可连接到PLC或其它设备使用，界面规格具有：RS-232、RS485 2w/4w。在这里，我们把COM端口的RS-422方式等同为 RS485 4W方式。请根据“与PLC连接手册”来连接PLC和触摸屏，以保证正确的连接。

1.3 MT8000系列触摸屏系统设定

当您第一次操作MT8000前，必须先在触摸屏上完成各项系统设定，设定完成后即可使用EB8000编辑软件开发个人专属的操作画面。

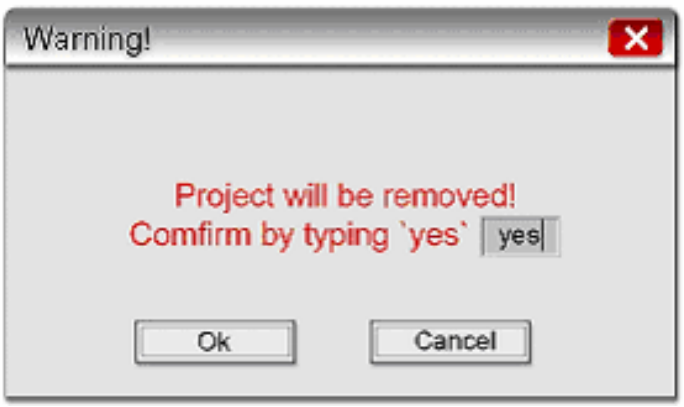
以下说明MT8000触摸屏启用时各项设定画面。

(1)重置系统为出厂设定

每台触摸屏背后都有一组重置按钮及拨码开关，当使用拨码开关SW1做不同模式切换时，将可激活相对应之功能(相关内容请参考安装手册)。

当使用者遗失或忘记MT8000的系统设定密码时，可以将拨码开关SW1切至ON，然后将MT8000重新上电。此时MT8000会先切换至屏幕触控校正模式，在用户完成校正动作后，会弹出一个对话窗口如下图。此对话窗口将询问使用者是否将MT8000的系统设定密码恢复为出厂设定，若为是则选定YES即可，反之则选择NO。

当选择为YES后，会弹出另一个对话窗口，如下图。此对话窗口将再次确认用户是否要将MT8000的系统设定密码恢复为出厂设定，并且要求用户输入YES做为确认，在完成输入后触控OK即可。(MT8000系列出厂时的系统预设密码为111111，但必须重设其它密码，包括download与upload所使用的密码皆需重设。)



注意：当进行MT8000的重置动作时，人机内的工程档案以及所储存的资料将全部被清除

● 系统工具条

启动触摸屏后可利用在屏幕右下方的[工具条]做系统设定；您只需将鼠标接上，用鼠标点击或触控屏幕右下角箭头  ，即会弹出工具条。在工具条中出现可以使用的图标，如下图：

大键盘：

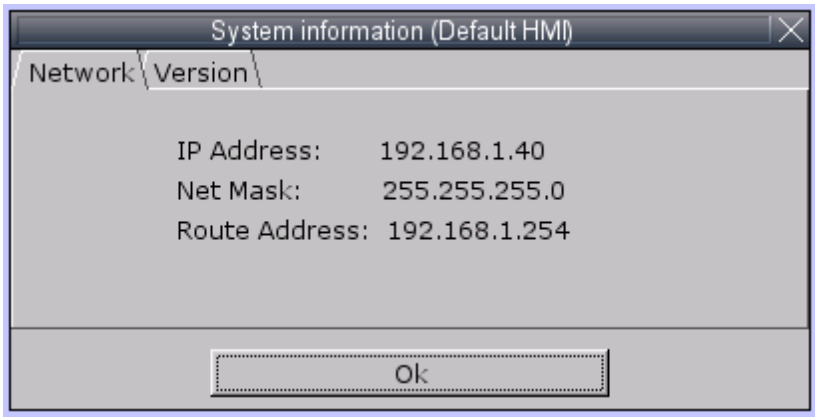
可以利用此键盘进行文字输入等信息。

小键盘

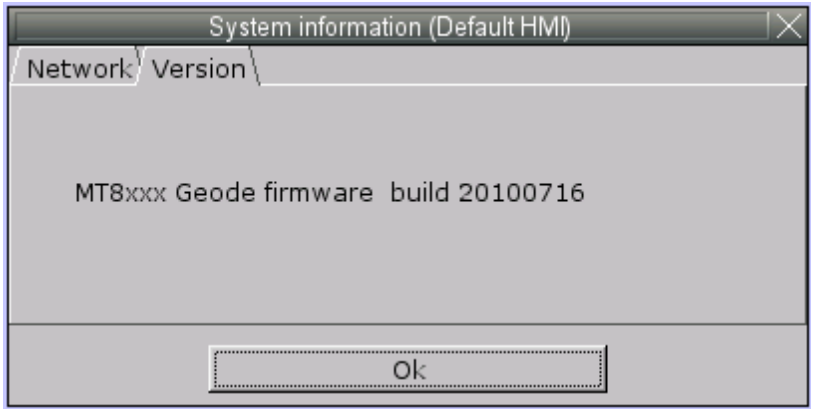
可以利用此键盘进行数字输入等信息。

系统信息：

Network：显示网络信息，包括触摸屏的IP地址和网关等相关信息。



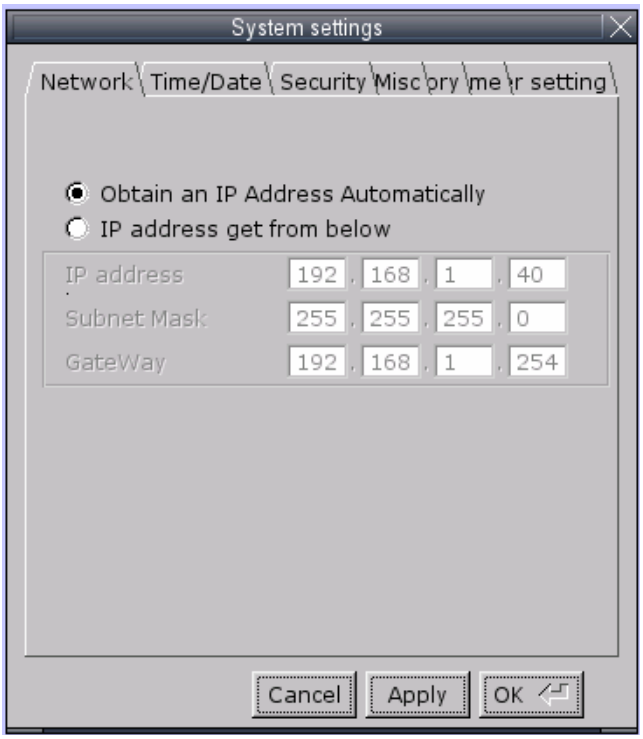
Version：显示触摸屏系统版本信息。



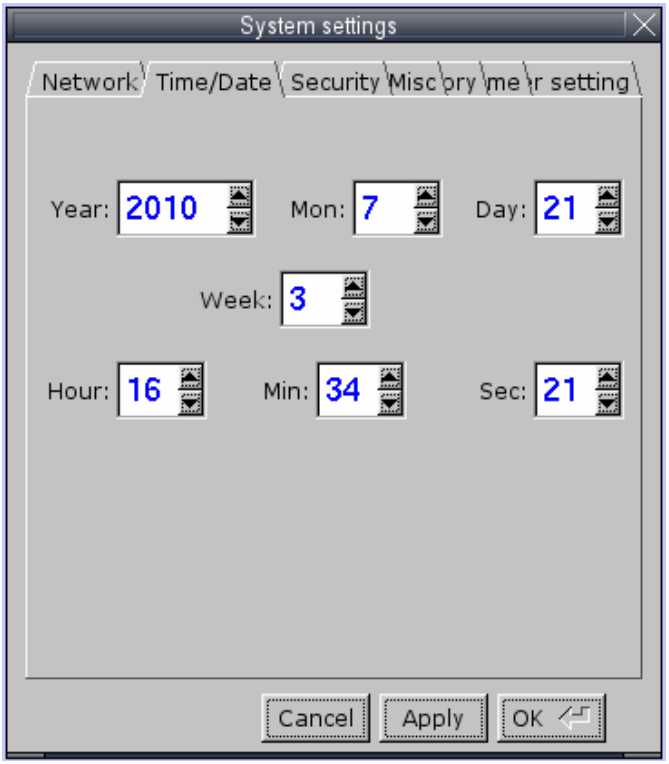
系统设定：

设定或更改触摸屏的各项系统参数，基于安全考虑必须进行密码确认（出厂密码为：111111）：

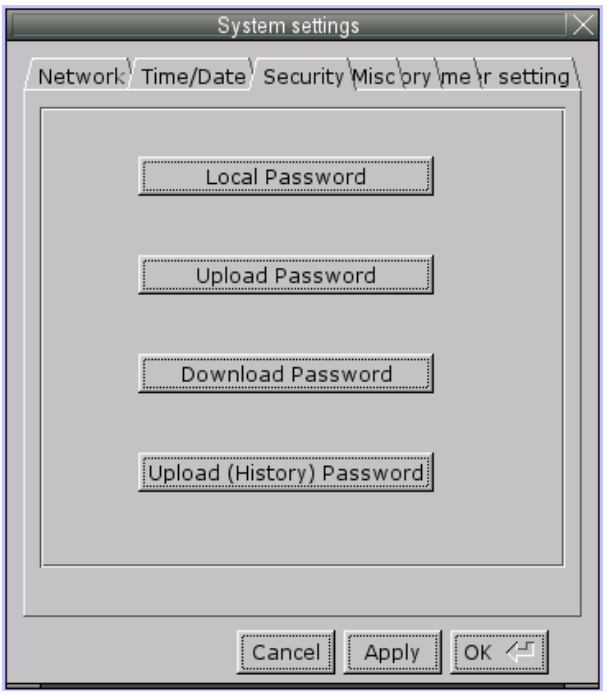
a. Network：用户可使用以太网下载画面程序到触摸屏上，但需正确设定操作对象（即触摸屏）的IP地址。选择[Obtain an IP Address Automatically]时，触摸屏的IP地址由所处的网域DHCP自动分配IP；若选用[IP address get from below]时，则必须手动输入IP地址及其他网络参数。



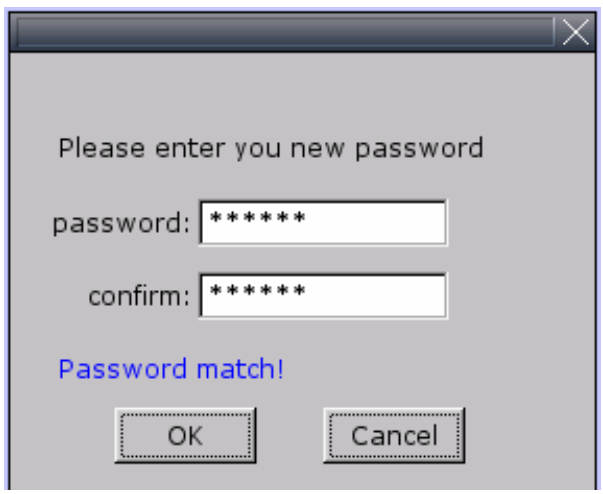
b. Time/Date：此设定页可设定触摸屏的本地时间和日期。



c. Security 触摸屏操作提供严谨的密码防护，预设密码为111111

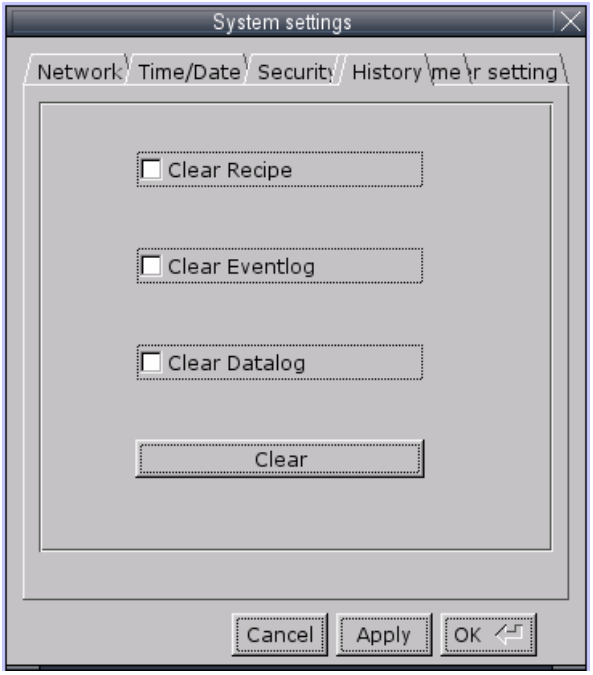


[Local Password]：进入系统设定时所需的密码。
[Upload Password]：上传工程文件时所需的密码。
[Download Password]：下载工程文件时所需的密码。
[Upload (History) Password]：上传资料取样数据与事件记录文件时所需的密码。
密码确认窗口如下：



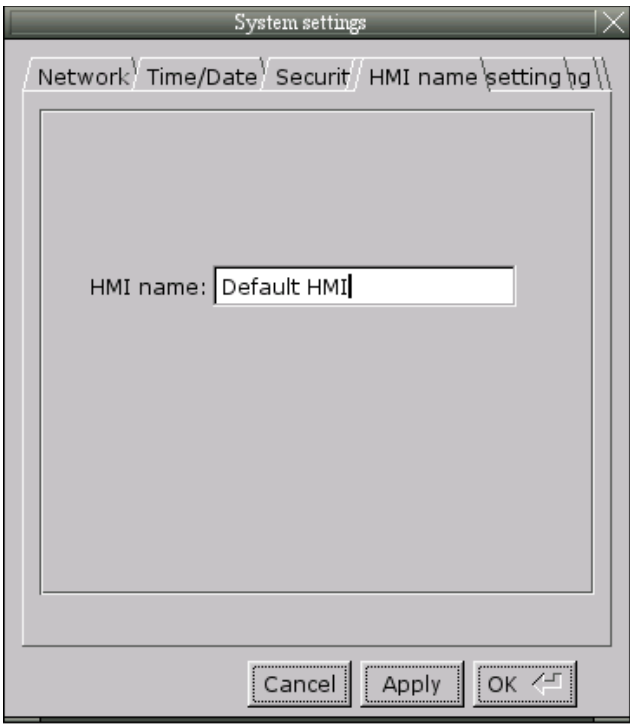
d. History：此设定页可清除存在触摸屏内的历史记录数据文件：

[配方数据]、[事件记录]、[数据取样记录]

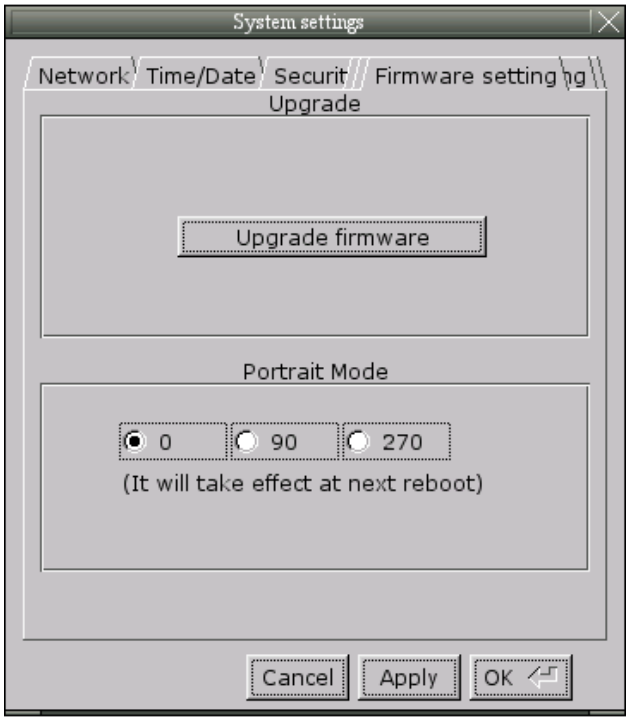


e. HMI name

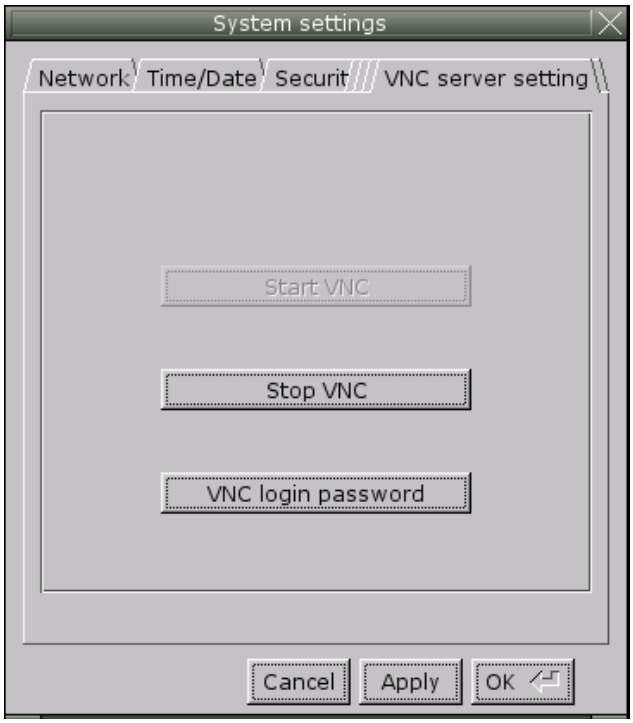
设定HMI名称以方便下载或上传工程文件。



f. Firmware setting: 用户可在此更新系统韧体及启用直立模式(只有i系列支持直立模式)。



g. VNC server: 启用此功能后, 可透过以太网络, 监控该触摸屏。

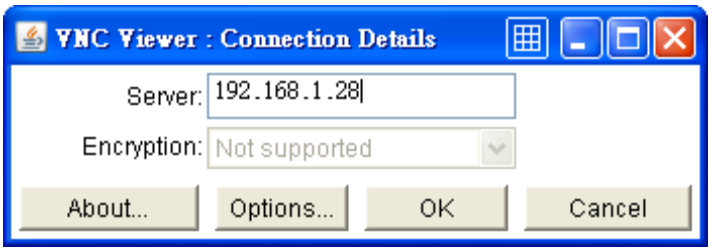


步骤1：打开触摸屏的VNC server设置登录密码

步骤2：安装Java IE或VNC viewer到PC

安装Java IE后可通过IE输入远端触摸屏的IP地址，例如：http://192.168.1.28

或者通过VNC viewer输入远端触摸屏的IP地址和密码

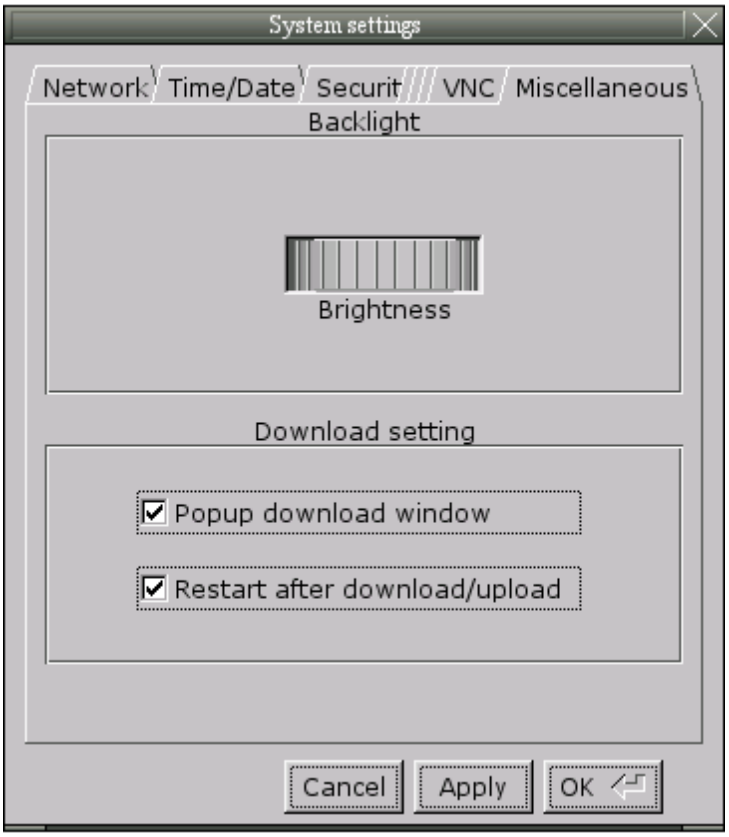


注意:

1) 一台触摸屏同时只能允许一个用户登入VNC server

2) 如果持续一个小时没有操作VNC server, 系统将会自动登出。

h. Miscellaneous: 利用屏幕上的旋钮可调整LCD画面亮度。



触摸屏触控校准模式

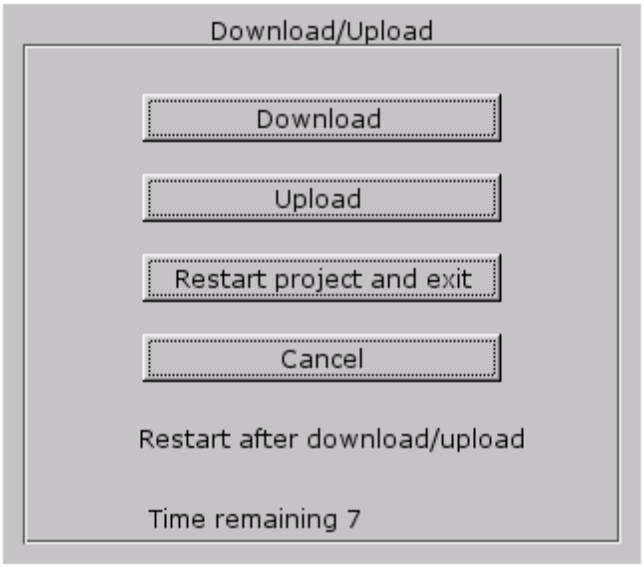
在这个模式下，当MT8000系列触摸屏电源开启时，就会在屏幕的左上角出现一个“+”，使用触控笔或手指触控这个“+”的中间，它就会移动。它会出现在左上角、右上角、右下角、左下角以及屏幕中央位置，当这五个点都被准确触控之后“+”会自动消失。同时校准参数会保留在触摸屏的系统里面。

注意：i系列触摸屏支持指拨开关1启动校准模式;X系列则在工具条中有启用校准模式的快捷按钮。

1.4 下载设定

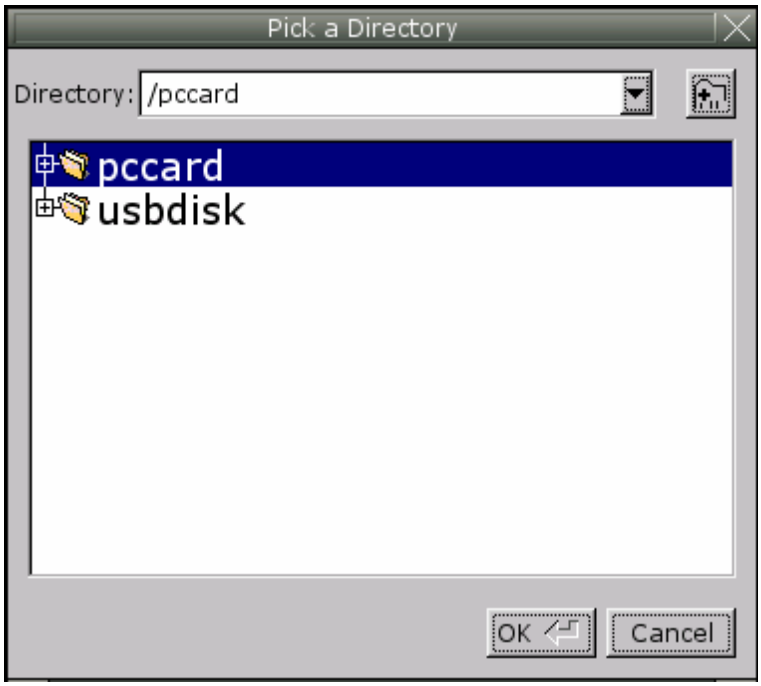
MT8000提供使用外部装置下载工程文件到触摸屏的方式，将外部装置插入时，并指定要下载的工程文件目录名称后，MT8000会将此目录下的所有数据下载到触摸屏上。

当触摸屏侦测有外部装置插入时(例如插入CF卡/SD卡或U盘)，会出现以下画面：



此时可选择需要的功能，如选定下载功能时必须先确认防护密码：

在完成确认密码后会显示外部设备底下目录的名称。(pccard: SD卡(CF卡); usbdisk: U盘)：



此时可选择工程文件的存放路径并触控[OK]键，即会开始下载。

注意：此处要下载的数据文件，用户必须先使用[建立储存在CF/SD卡与U盘中的下载资料…]来产生。

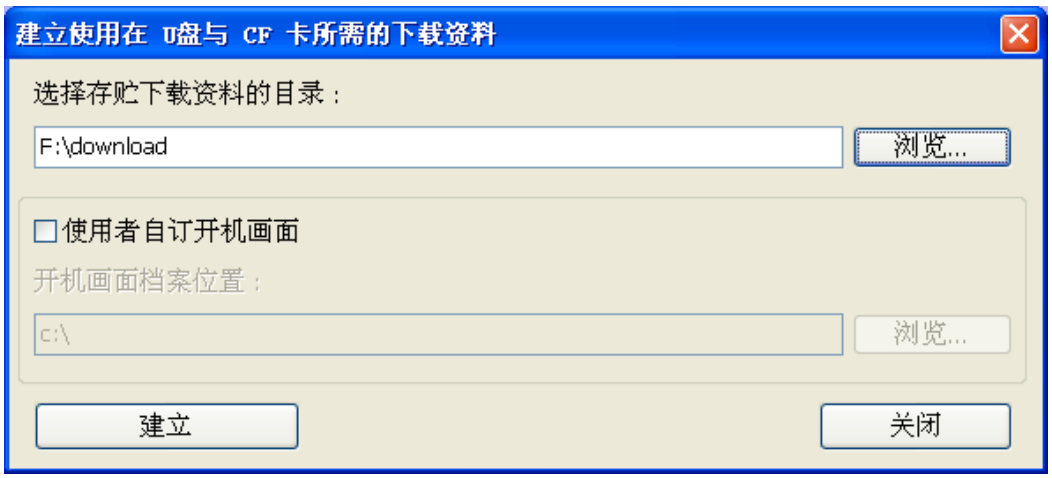
一般来说，所建立的下载文件分为两个目录：

[MT8000]

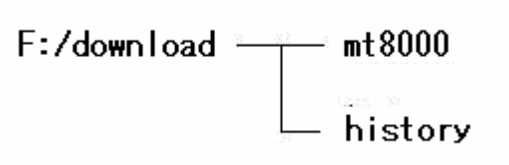
存放工程文件

[History]

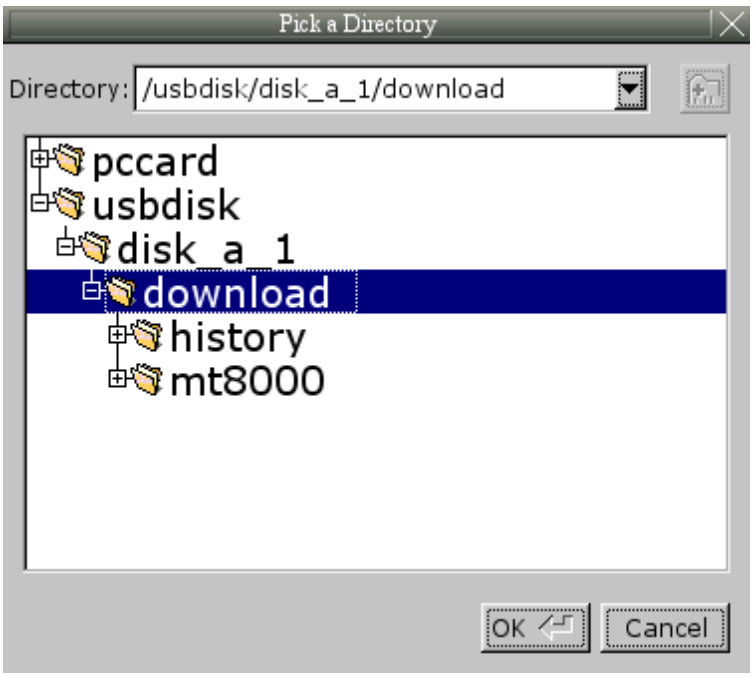
当需要下载历史数据时，会产生此目录，也就是说如果下载文件的存放位置设定如下：



则下载数据的存放结构为：



此时下载文件时必须选择存放下载数据的最上层路径，也就是说，若以上图为例，必须选择download，不可选择mt8000或history。



下载过程中，触摸屏画面会触控图所示次序改变：

停止当前工程文件：

开始下载新的工程文件：

成功下载新的工程文件后，画面如下图所示：

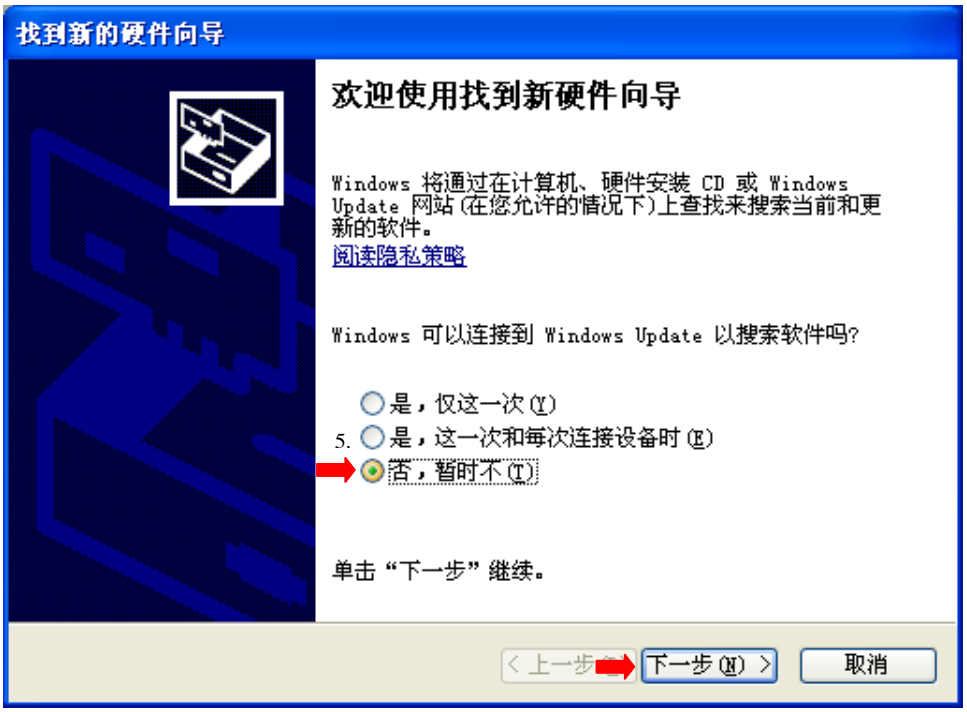
1.5 i系列HMI USB driver 安装说明

MT6000/8000 i系列HMI产品，可以使用USB to Mini USB cable下载工程文件至HMI，当利用此USB cable下载工程文件时，HMI相对于PC的角色为从属设备(PC为Master，HMI为Client)，因此在首次使用前必须安装EB8000V2.00版（或更新）的软件，并在PC端安装HMI接口的USB驱动程序。

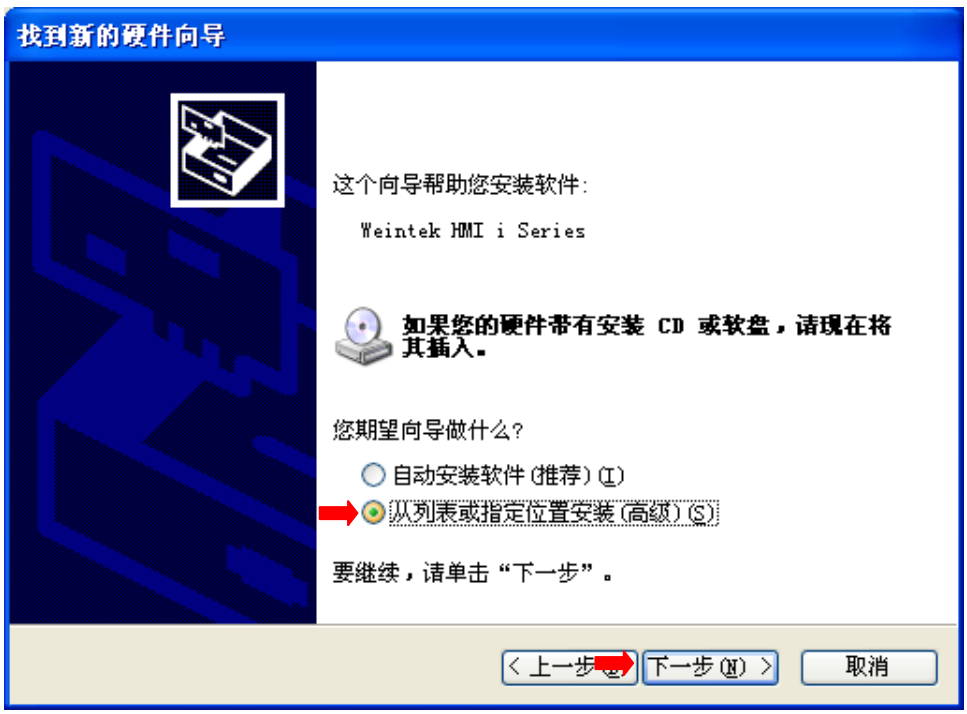
以下为USB驱动程序安装说明

注意：EB8000 V4.20版本或更新既可自动安装，也可手动安装，以下为手动安装操作步骤：

1. 将USB线连接HMI Mini USB Port后，另一端再连接至计算机的USB port，此时PC会弹出“找到新的硬件向导”的窗口，请选择第三选项“否,暂时不”，并触控“下一步”。



2. 选择第二选项“从列表或指定位置安装(高级)”，并触控“下一步”。



3.“在这些位置上搜索最佳驱动程序”选项下勾选“在搜索中包括这个位置”，浏览驱动程序安装路径指向C:\EB8000\usbdriver，触控“确定”及“下一步”。

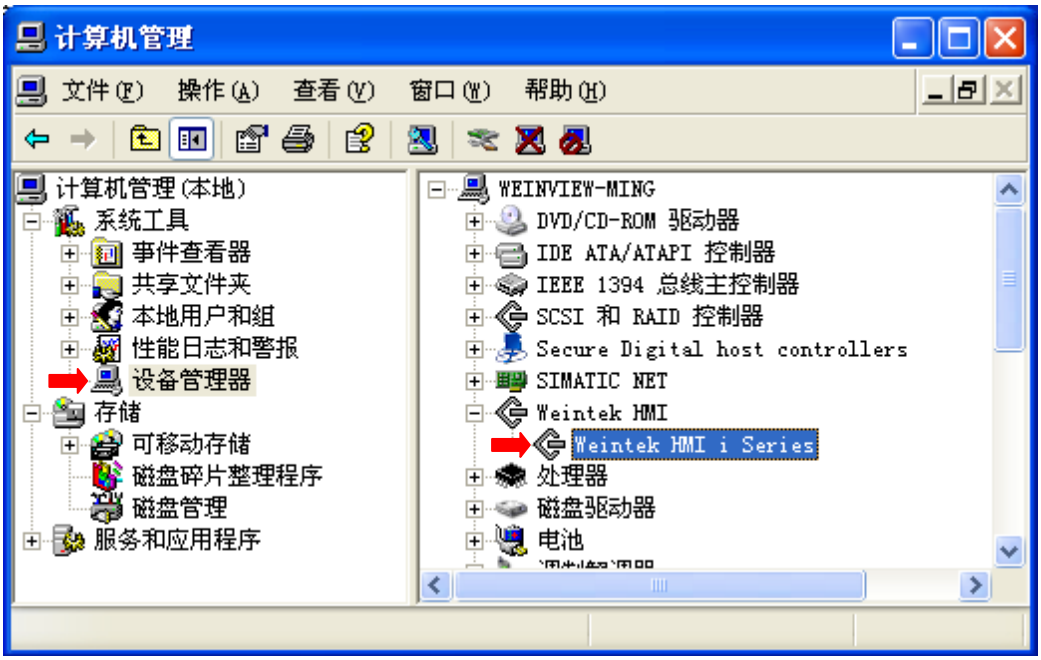
4. 此时，计算机会从C:\EB8000\usbdriver安装Weintek HMI i Series驱动程序。

(安装过程1)

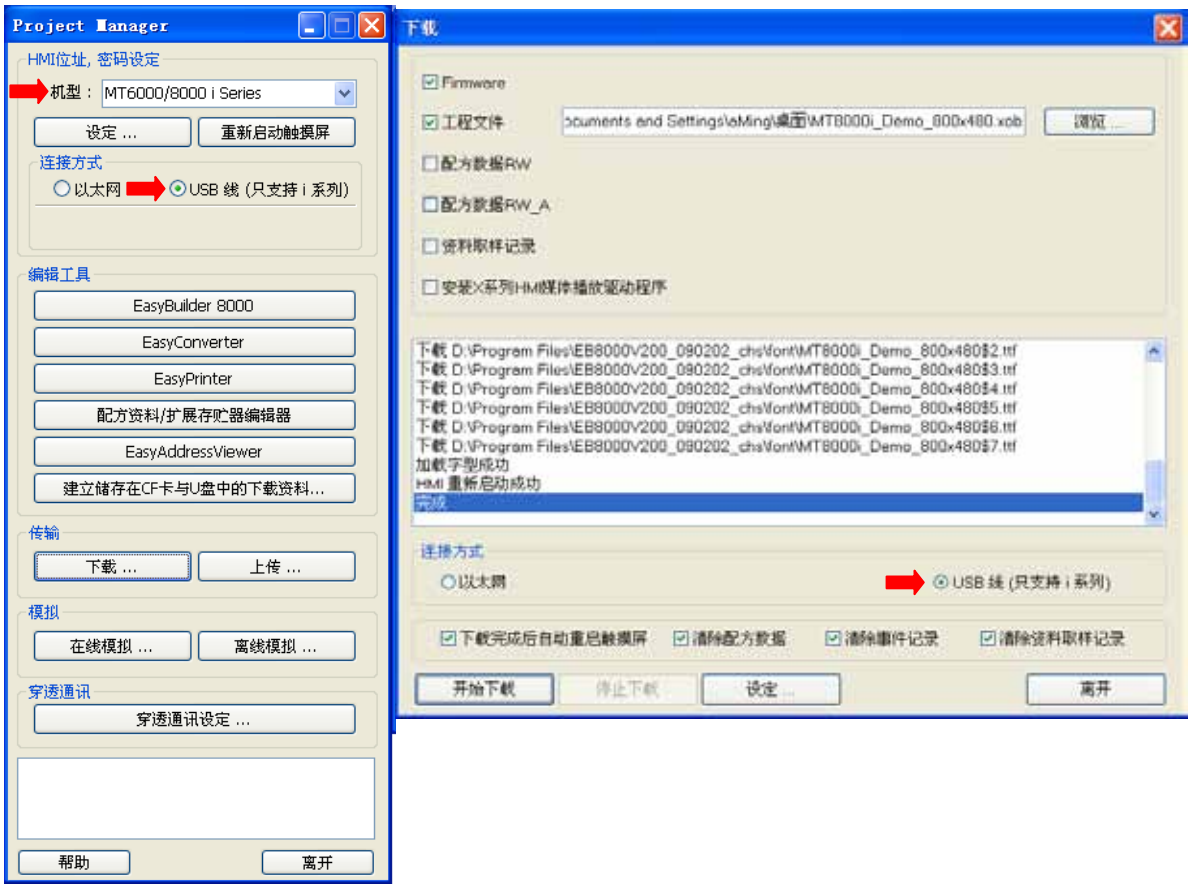
(安装过程2)

5. 当出现 “完成找到新硬件向导” 后, 触控 “完成” , 即完成了驱动程序的安装。

6. 使用者可打开计算机管理 / 设备管理器, 确认Weintek HMI i Seires是否安装成功。



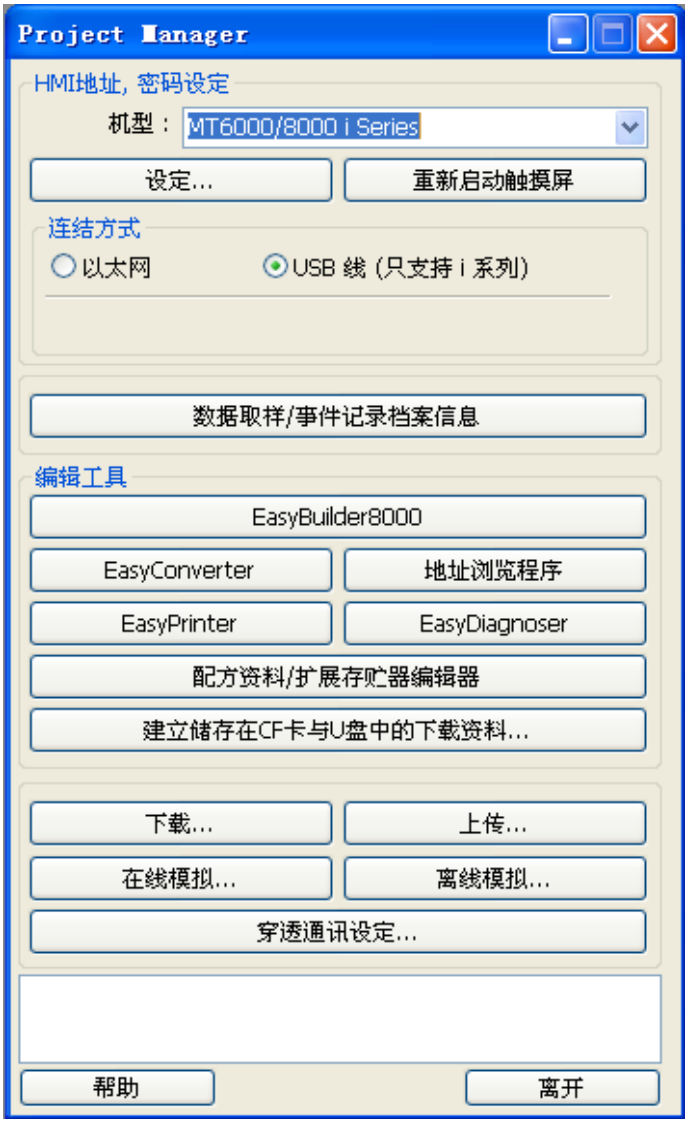
7. 驱动程序安装完成后, 打开Project Manager或EB8000, 选用USB联机方式即可进行工程文件上传、下载动作。



第二章　关于Project Manager

前言

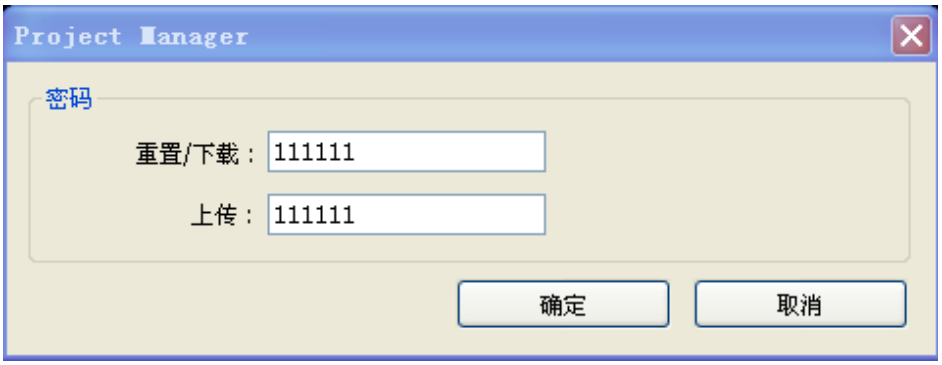
在EB8000安装完成之后，用户在电脑桌面上会看到“Project Manager”的快捷方式，点击这个快捷方式就可以打开Project Manager，如下图所示：Project Manager是EB8000软件的综合管理器，整合了EasyBuilder 8000所提供的各项功能：



下面我们详细来介绍Project Manager的各项功能：

2.1　HMI密码设定

设定



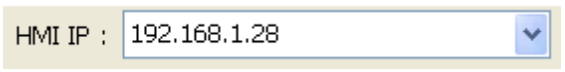
使用以太网或usb线操作MT8000/MT6000触摸屏时，需正确设定操作对象所需的密码，“下载”与“重新启动触摸屏”功能使用同一组密码，“上传”功能则使用另外一组密码。如何更改与检视HMI的IP地址与密码请参考相关章节。

重新启动触摸屏

在某些情况下需重置系统，例如更新触摸屏内部的文件。使用此功能可在不需重新开机的情况下，重新激活系统，并恢复到开机时的状态。

连接方式选择

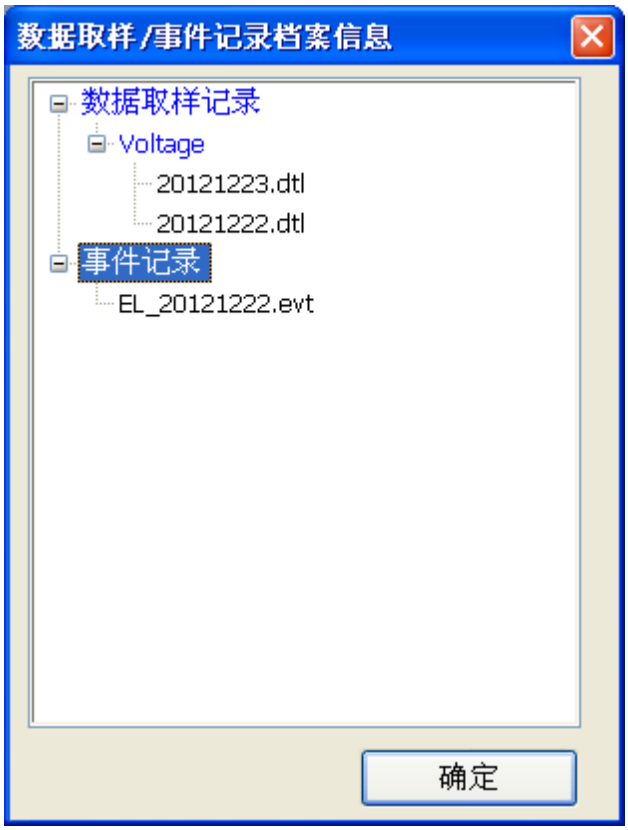
1. 设定要操作HMI的IP：使用Ethernet操作MT8000时，需正确设定操作对象的IP地址。



2. 使用i系列触摸屏时，可以选择USB线对触摸屏进行操作。

数据取样/事件记录档案信息

在使用i系列的触摸屏时，可以通过usb直连线直接查询触摸屏内历史记录。



2.2 编辑工具

EasyBuilder 8000

激活EB8000图形编辑器。

EasyConverter

事件记录与资料取样记录转文件工具

地址浏览程序

检视各个PLC支持的设备类型及地址范围

EasyDiagnoser

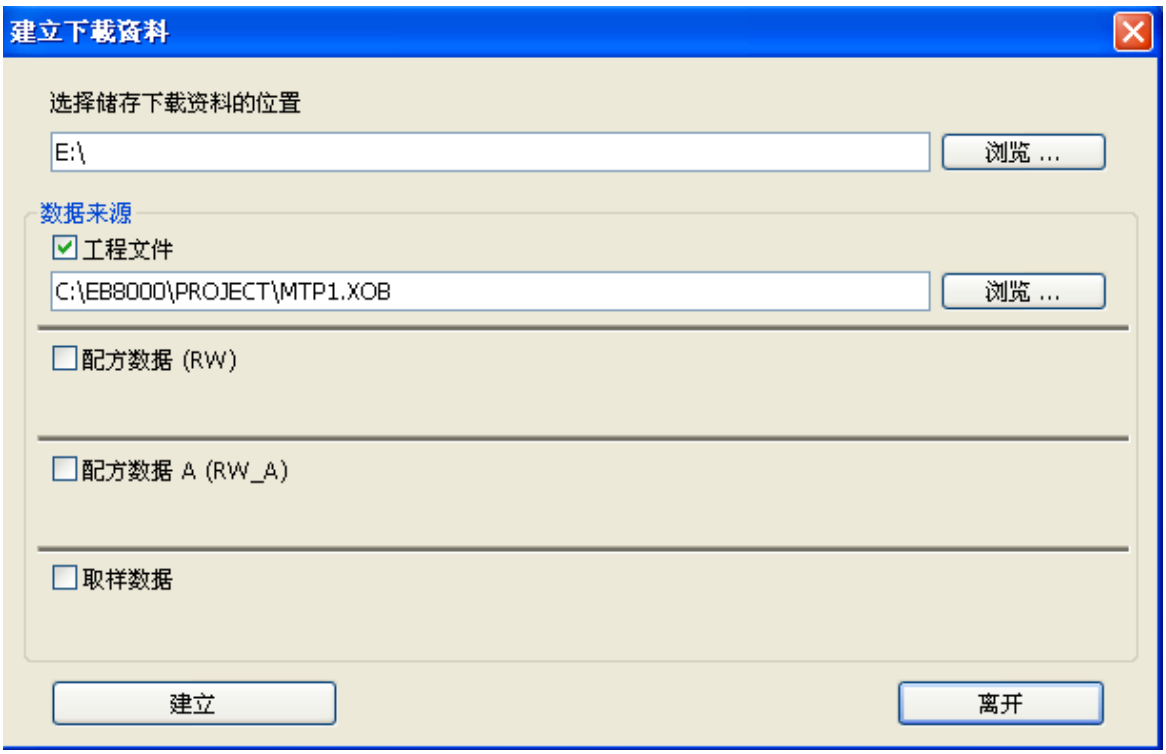
在线监控程序的出错工具

配方资料/扩展存储器编辑器

配方数据编辑/扩展储存器编辑工具

建立储存在CF/SD卡与U盘中的下载资料

除了使用以太网和USB线之外，EB8000也提供使用CF卡/SD卡与U盘下载资料到HMI的方式，此项功能即是用来建立这些被下载的资料，下图为此项功能的设定画面。



[选择储存下载数据的位置]

在指定下载资料的存放位置(或目录名称)，触控“建立”后在此目录下建立下载资料的各项内容。可以直接指定存放位置在CF卡/SD卡或U盘上，或在资料建立完成后将整个目录拷贝至CF卡/SD卡或U盘上即可。

将CF卡/SD卡或U盘插上HMI，并指定要下载的目录名称后，EB8000会将此目录下的所有内容下载到HMI上。

注意：存储资料夹的路径有限制，存放位置必需包含子目录名称，避免只是指定根目录，例如“c:\”或“f:\\”皆为不合法的名称。

[工程文件]

利用EB8000可以将画面组态内容(MTP文件), 编译获得MT8000上所使用的XOB文件, 此选项可选择CF卡/SD卡或U盘所需的XOB文件。

[配方数据 (RW)]

此选项可选择CF卡/SD卡或U盘所需的RW配方数据文件, 文件有效的最大容量为512K, 更详细的内容请参考“配方资料传送”章节的说明。

[配方数据 A (RW_A)]

此选项可选择CF卡/SD卡或U盘所需的RW_A配方数据文件, 文件有效的最大容量为64K, 更详细的内容请参考“配方资料传送”章节的说明。

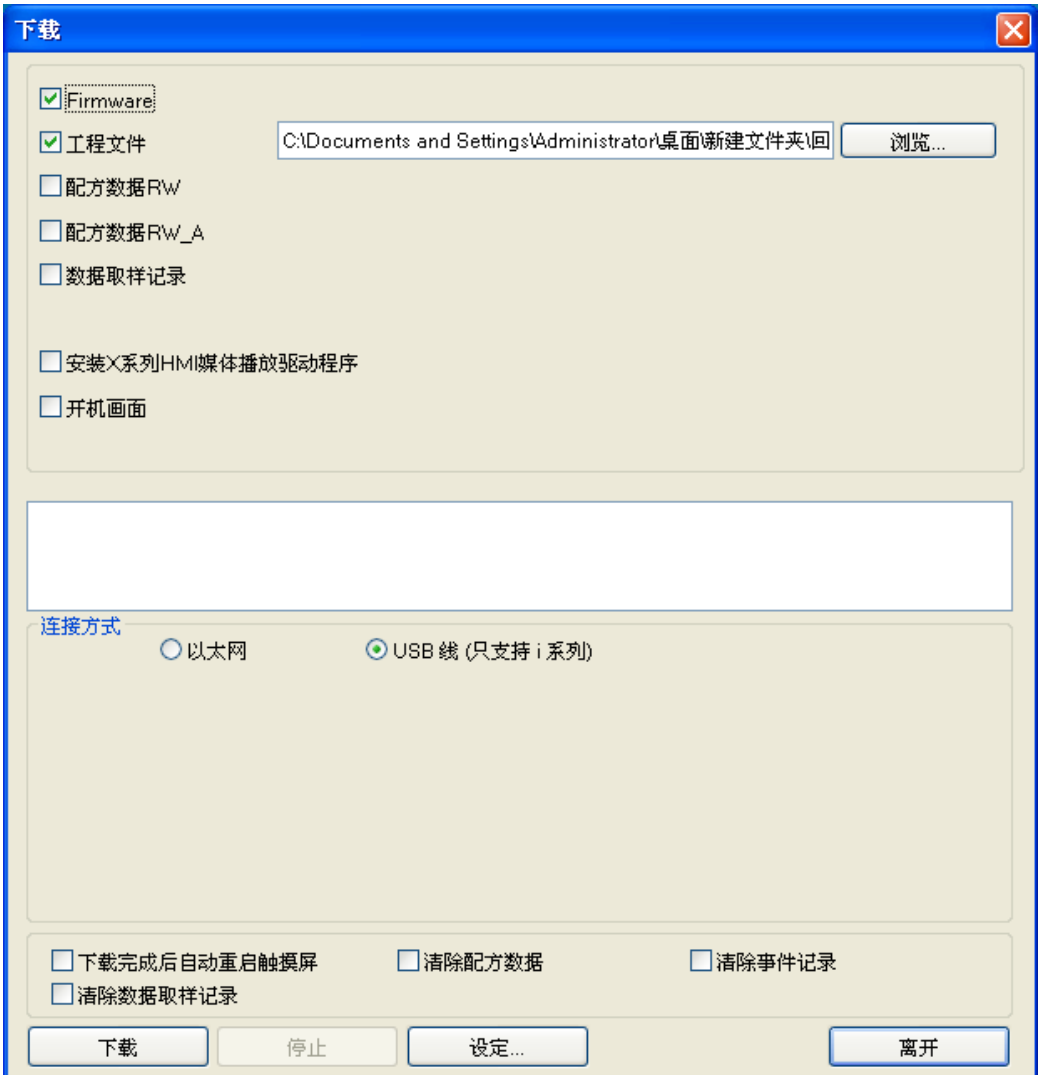
[资料取样记录]

此选项可选择CF卡/SD卡或U盘所需的资料取样文件, 更详细的内容请参考“资料取样元件”的说明。

2.3 传输

下载

使用以太网或USB线下载工程文件到MT8000/MT6000的HMI上。触控此功能按键后将出现下图的画面。



[Firmware]

勾选此选项表示要更新HMI上的所有核心程序。第一次下载工程到触摸屏时，一定要勾选此选项。

[工程文件]

MT8000上所使用的是XOB文件，工程文件经过EB8000编译后即可产生XOB文件。此项内容用来选择要下载到HMI的XOB文件的名称。

[配方数据RW]

选择要下载至MT8000上的RW配方资料，文件有效的最大容量为512K，更详细的内容请参考“配方传送”的说明。

[配方数据RW_A]

选择要下载至MT8000上的RW_A配方资料，文件有效的最大容量为64K，可参考“配方传送”的说明。

[资料取样记录]

选择要下载至MT8000上的资料取样文件，更详细的内容请参考“资料取样元件”的说明。

[安装X系列HMI媒体播放驱动程序]

第一次使用EB8000下载程序到X系列触摸屏前，请先下载此驱动程序

[开机画面（i系列）]

勾选此选项，就可以将指定的BMP格式图片下载到 HMI 中。下载完成后，重新启动触摸屏，就会先显示这个图片，然后再载入下载的画面程序，即实现开机LOGO的功能。

[下载完成后自动重启触摸屏]

勾选此选项，触摸屏会在下载程序成功后自动会重启触摸屏。

[清除配方数据]

此选项如被勾选，下载程序进行前会先将所有配方的数据设定为0。

[清除事件记录]

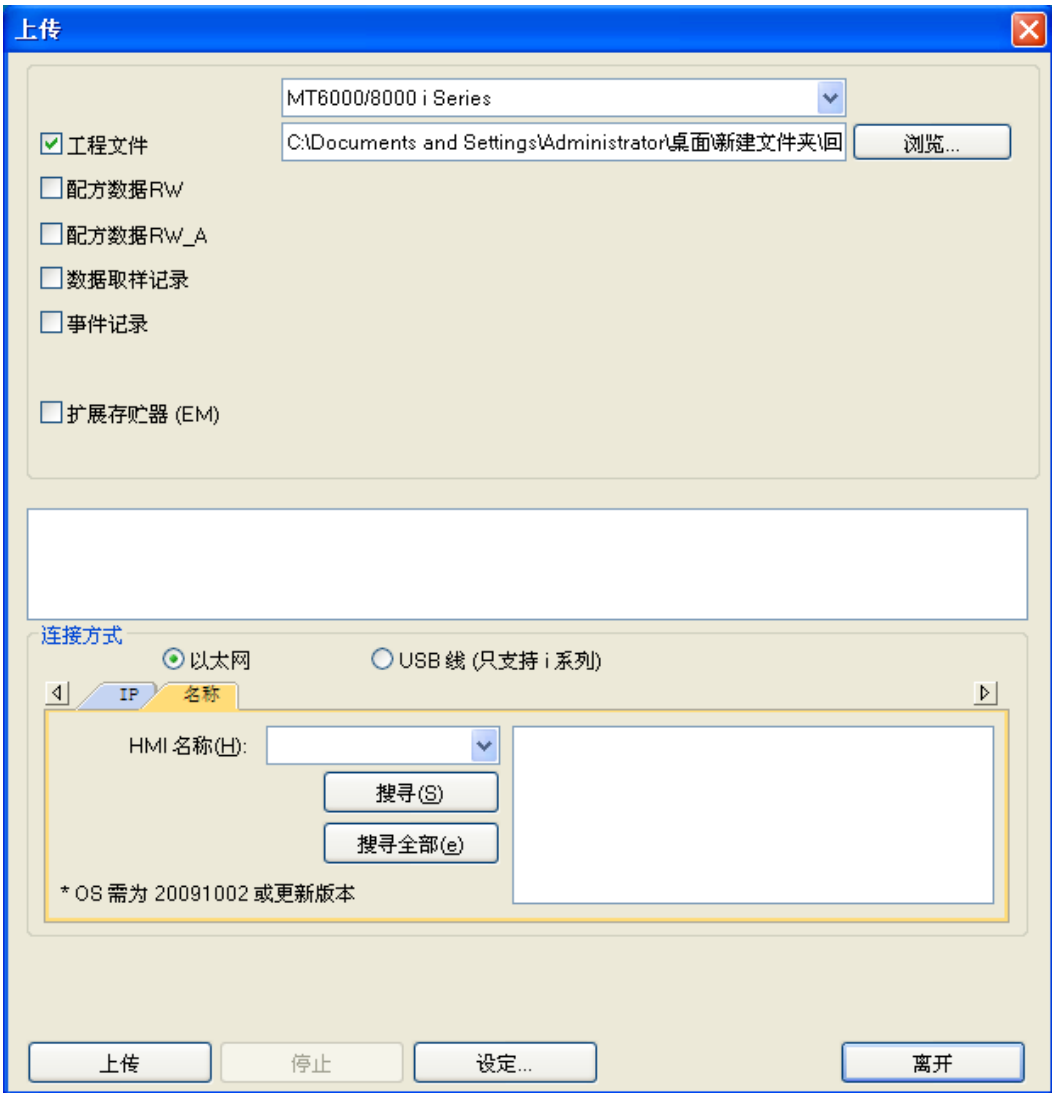
此选项如被勾选，下载程序进行前会先清除HMI上存在的所有事件记录文件。

[清除资料取样记录]

此选项如被勾选，下载程序进行前会先清除HMI上存在的所有资料取样文件。

上传

使用以太网或USB在线传MT8000/MT6000触摸屏上的工程文件到PC，触控此功能后将出现下图的画面，上传前必须先选择存放工程文件的路径。



[工程文件]

选择XOB文件上传后的存放位置。

[配方数据RW]

选择配方数据RW上传后的存放位置。

[配方数据RW_A]

选择配方数据RW_A上传后的存放位置。

[资料取样记录]

选择资料取样记录文件上传后的存放位置。

[事件记录]

选择事件记录文件上传后的存放位置

2.4 模拟

在线模拟/离线模拟

EB8000提供两种模拟功能：在线模拟和离线模拟。

使用在线模拟功能，可以在没有触摸屏的情况下，实现电脑和PLC或者控制器之间的通讯，在手边没有触摸屏的情况下，可以先行调试程序，大大节约了调试时间。

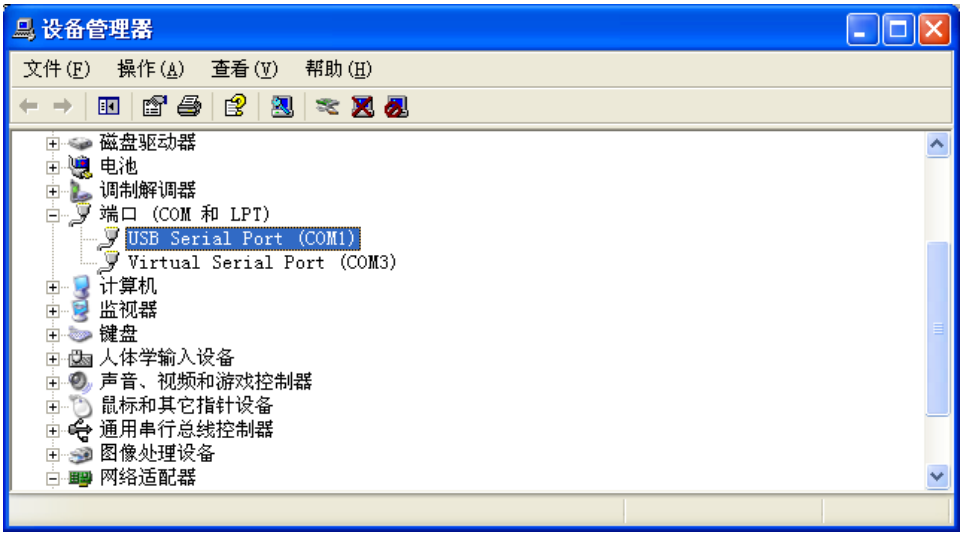
使用离线模拟功能，可以不需要将编写好的画面程序下载到触摸屏，只是在电脑上模拟就可以查看画面的布局效果等。离线模拟功能不需正确连接PLC即可模拟。反之，在线模拟功能则必须正确设定参数及连接PLC才可执行。

注意:

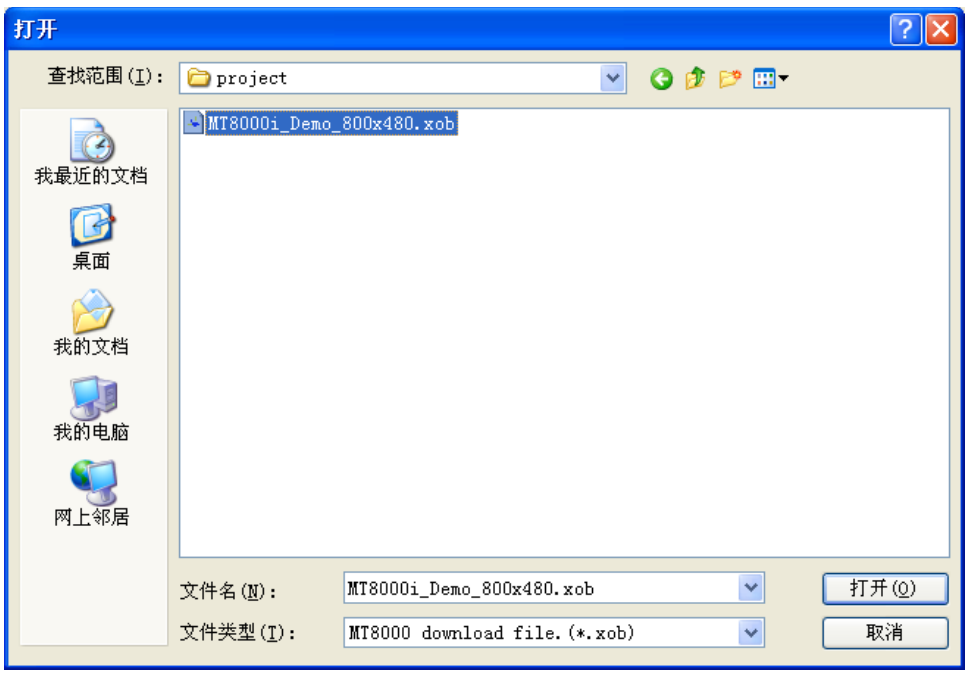
1.在线模拟有10分钟的限制。

2.在线模拟为PC与PLC通过PLC的编程电缆（串口型）进行直连；

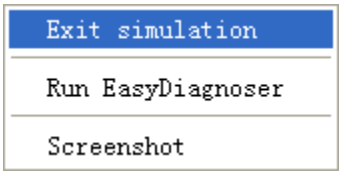
3. 做在线模拟时, 在设备列表中设定与PLC通讯的端口号, 必须与PC和PLC连接的端口号一致, 如下图所示, 与FX 1N PLC做在线模拟, 那么在EB8000中设定MT8000的COM1端口与PLC相连, 则在PC上的设备管理中, 也必须设定为COM1端口;



执行离线与在线模拟功能, 需先选择XOB文件的来源, 如下图:



在执行在线模拟/离线模拟时，点击鼠标右键可以执行以下三项功能：



a. “Exit simulation”
退出模拟状态。

b. “Run EasyDiagnoser”
执行 Run EasyDiagnoser 监看目前的通讯状态。

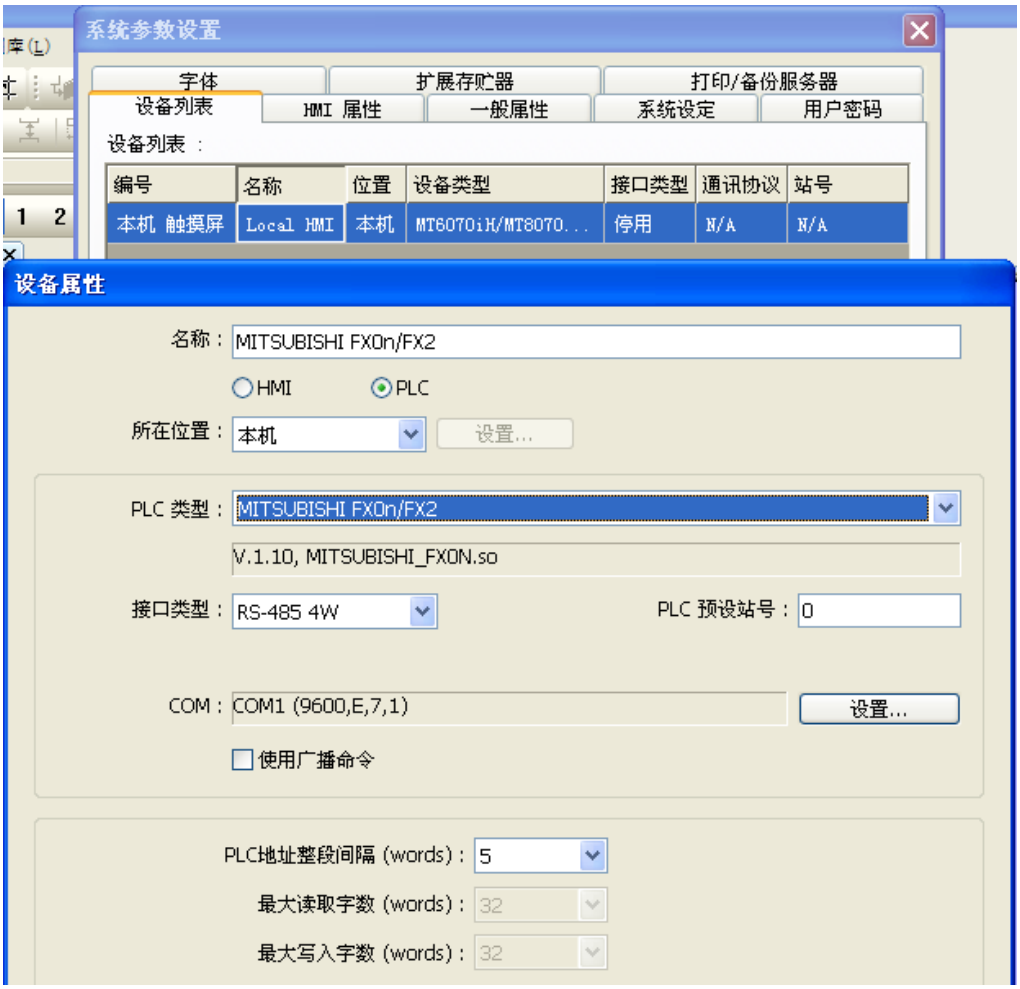
c. “Screenshot”
将目前显示的画面使用图片的方式储存到安装目录下的 screenshot 文件夹下。

第三章　制作一个简单的工程

下文以连接三菱的PLC为例，说明如何制作一个简单的工程。首先触控工具条上开启新文件的工作按钮，如下图：

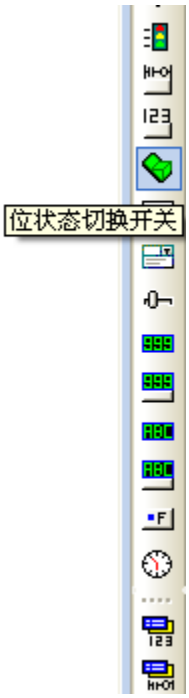
随后选择正确的机型与显示模式：

在触控确定键后，下一步除了要正确设定系统参数属性外，需在[设备列表]中使用[新增 …]功能增加一个新的设备，设定内容如下图。

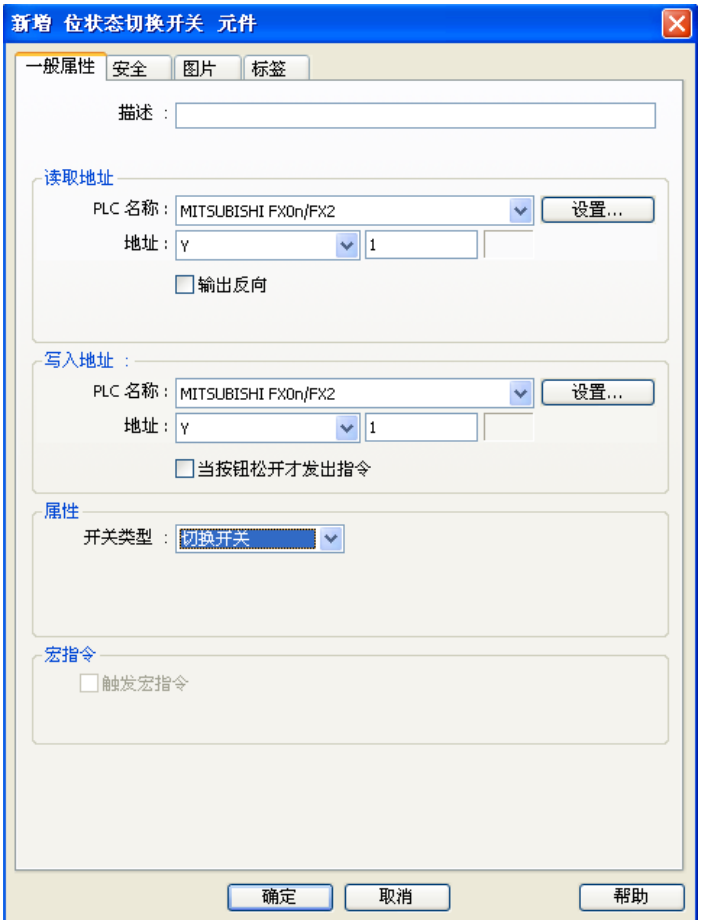


触控确定键后可以发现[设备列表]增加了一个新的设备“MISUBISHI FX0n/FX2”

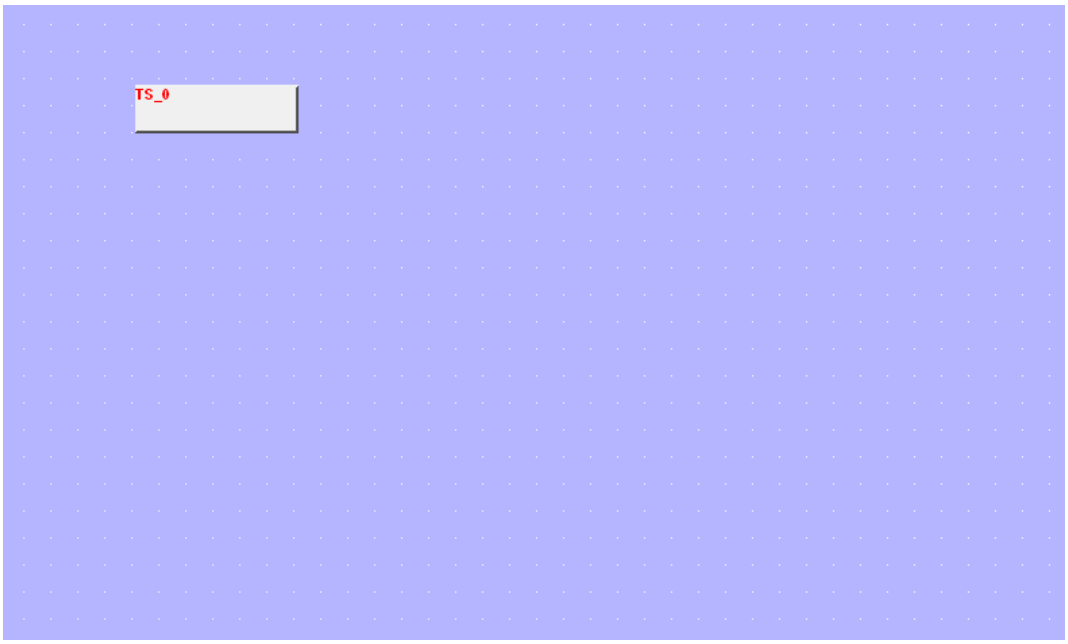
假设现在要增加一个[位状态切换开关]元件, 可触控如下图所示的元件按钮。



随后将会出现下图所示的对话窗，在正确设定各项属性后，触控确定键并将元件置放在适当位置。

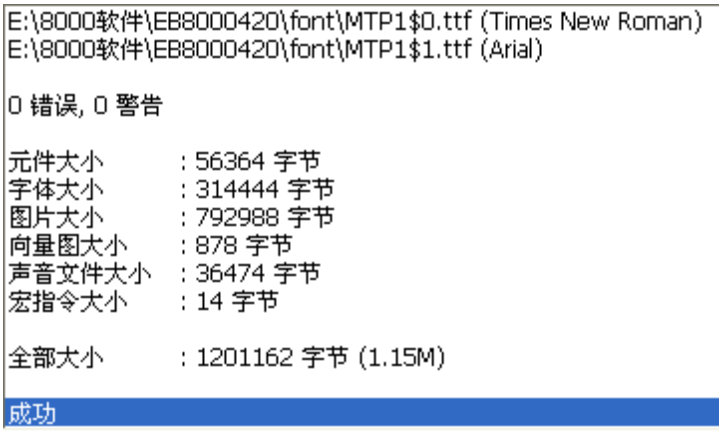


最后window 10的画面将如下图所示，如此即完成一个简单的工程文件。

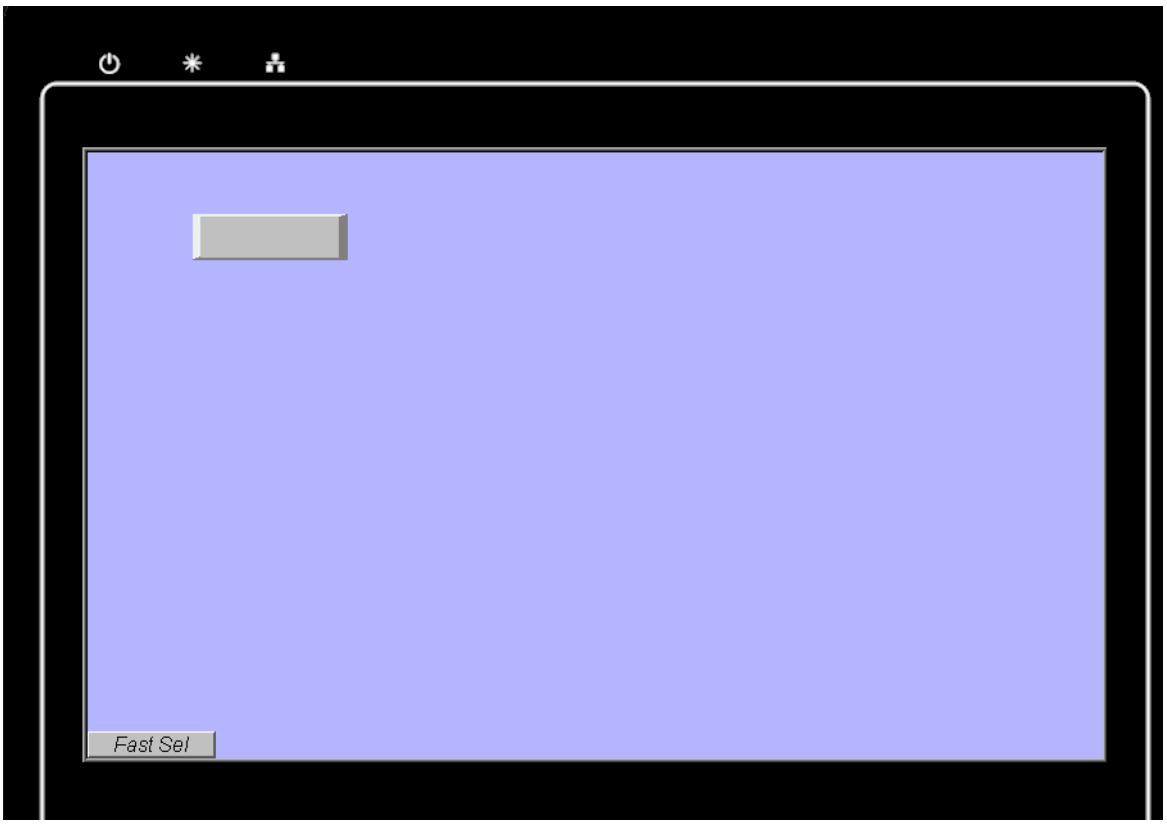


保存完成后用户可以使用编译功能，检查画面规划是否正确，编译功能的执行按钮如下图所示。

假使编译结果如下图所示，并不存在任何错误，即可执行离线模拟功能。



上图为离线模拟的执行按钮，执行后部分画面如下：



如需进行联机模拟，在接上设备后使用下图的工作按钮即可进行。

第四章　编译、模拟与下载

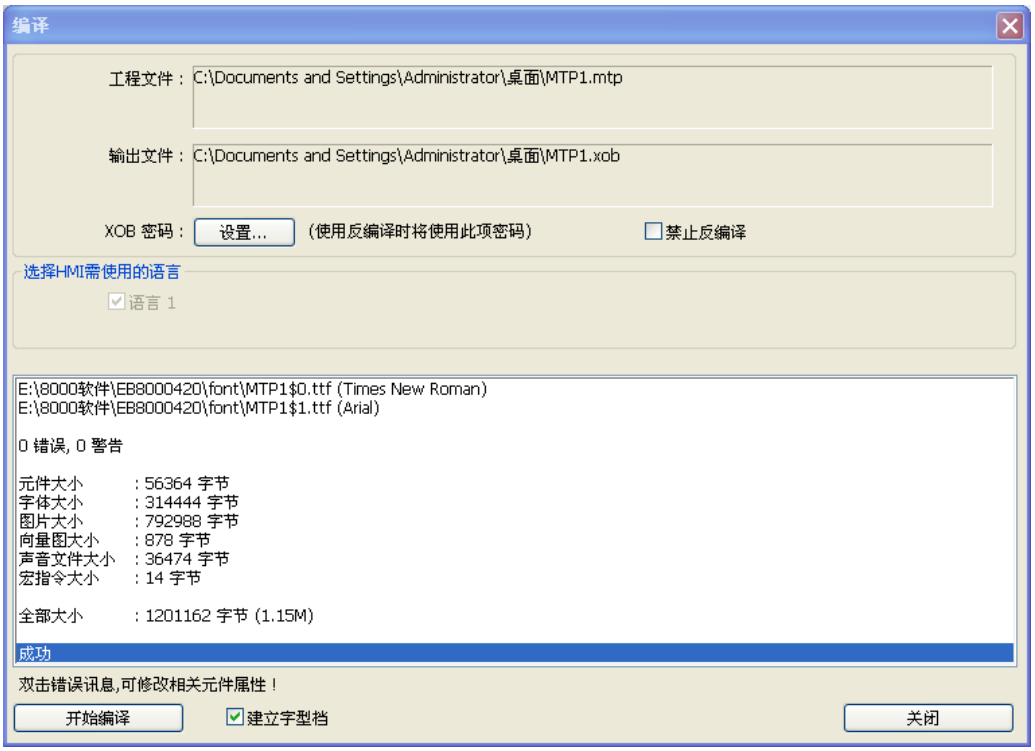
一个完整的设计步骤包括：画面编辑、编译、模拟与下载，下文说明各项步骤的内容。

4.1 画面编辑

使用EB8000规划所需的各种画面，编辑完成后可将画面规划内容保存为MTP文件。

4.2 编译

在使用EB8000完成工程文件(MTP)后，下一步需使用EB8000 提供的编译功能，将MTP文件编译为下载至HMI所需的XOB文件。在触控工具条上的编译按钮可获得“编译”对话窗，如下图所示：



“编译对话窗”中的[工程文件]显示目前工程文件的名称，[输出文件]则显示编译成功后将得到的XOB文件名称。在触控[开始编译]按钮后，编译对话窗中的编译信息窗口将显示下列信息：

字型文件

文字显示所需要的字型文件，这些文件将被下载至HMI中。

元件大小

元件编译后的大小。

字体大小

所需字型文件的全部大小。

图片大小

所使用图片文件的大小。

向量图大小

所使用向量图文件的大小。

声音文件大小

所使用声音文件的大小。

宏命令大小

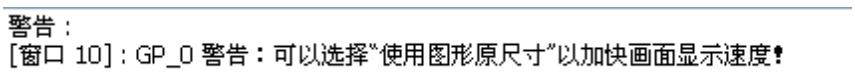
所使用宏命令文件的大小。

全部大小

所使用全部文件的大小。

信息窗口出现“0 错误”的信息表示编译成功，此时可继续执行各项模拟功能。若编译结果存在错误，则需依照信息的指示修正这些错误。

有时信息窗口会出现下列信息：



此种警告信息是在提醒用户，所使用的图片因为没有使用原尺寸，将影响到画面的更新速度，因此建议用户尽量勾选“使用图形原尺寸”这个选项，来提升画面的更新速度。参考下图。

注意: 警告信息并不会妨碍各项模拟功能的执行。

4.3 模拟

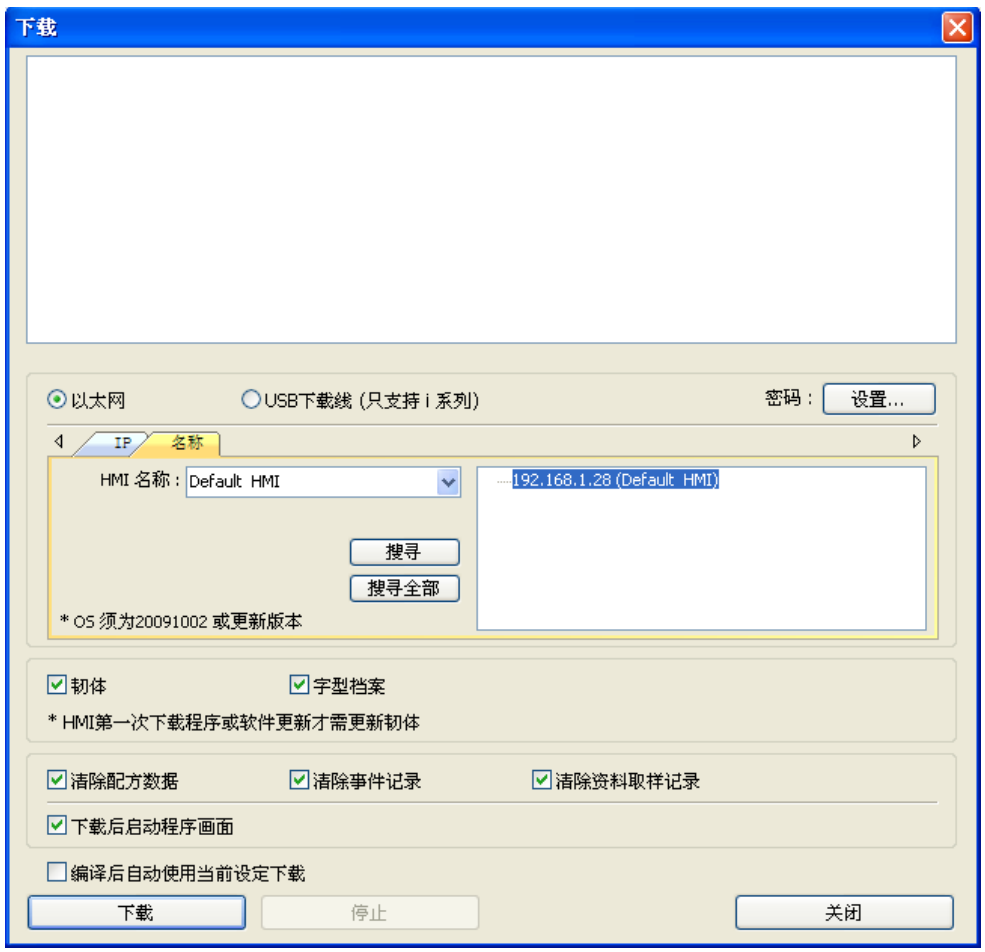
模拟可分为“离线模拟”与“在线模拟”两种，“离线模拟”不需接上PLC，PC会使用虚拟设备模拟PLC的行为；“在线模拟”则需接上PLC，并需正确设定与这些PLC的通讯参数。

注意: 在PC上进行“在线模拟”时，如监控对象为本地的PLC(也就是接在本地PC上的PLC)，则监控时间会有10分钟的限制。

要进行“离线模拟”与“在线模拟”，可以使用Project Manager的“离线模拟”与“在线模拟”的功能；另一种方式是使用EB8000所提供的功能，触控工具条上的“离线模拟”与“在线模拟”按钮后即可进行这些功能。

4.4 下载

在完成模拟动作确认画面规划结果无误，下一步需将XOB文件下载到触摸屏中。下载XOB文件的一种方式是利用Project Manager的“下载”功能，此部分请参考有关Project Manager的说明。另一种下载方式是使用EB8000的下载功能，触控工具条上的下载按钮将出现 “下载对话窗”，如下图所示：



“下载对话窗”中的各设定项目说明如下：

[HMI IP]

用来设定下载触摸屏的IP地址。

[密码]

下载密码，输入下载所需的密码，具体可参考“硬件设定说明”。

[韧体]

勾选此选项表示要更新触摸屏上的所有核心程序。第一次下载工程到触摸屏时，一定要选择此选项。

[清除配方数据]

此选项如被勾选，下载程序进行前会先将所有配方的数据设定为0。

[清除事件记录]

此选项如被勾选，下载程序进行前会先清除触摸屏上存在的所有事件记录文件。

[清除资料取样记录]

此选项如被勾选，下载程序进行前会先清除触摸屏上存在的所有资料取样文件。

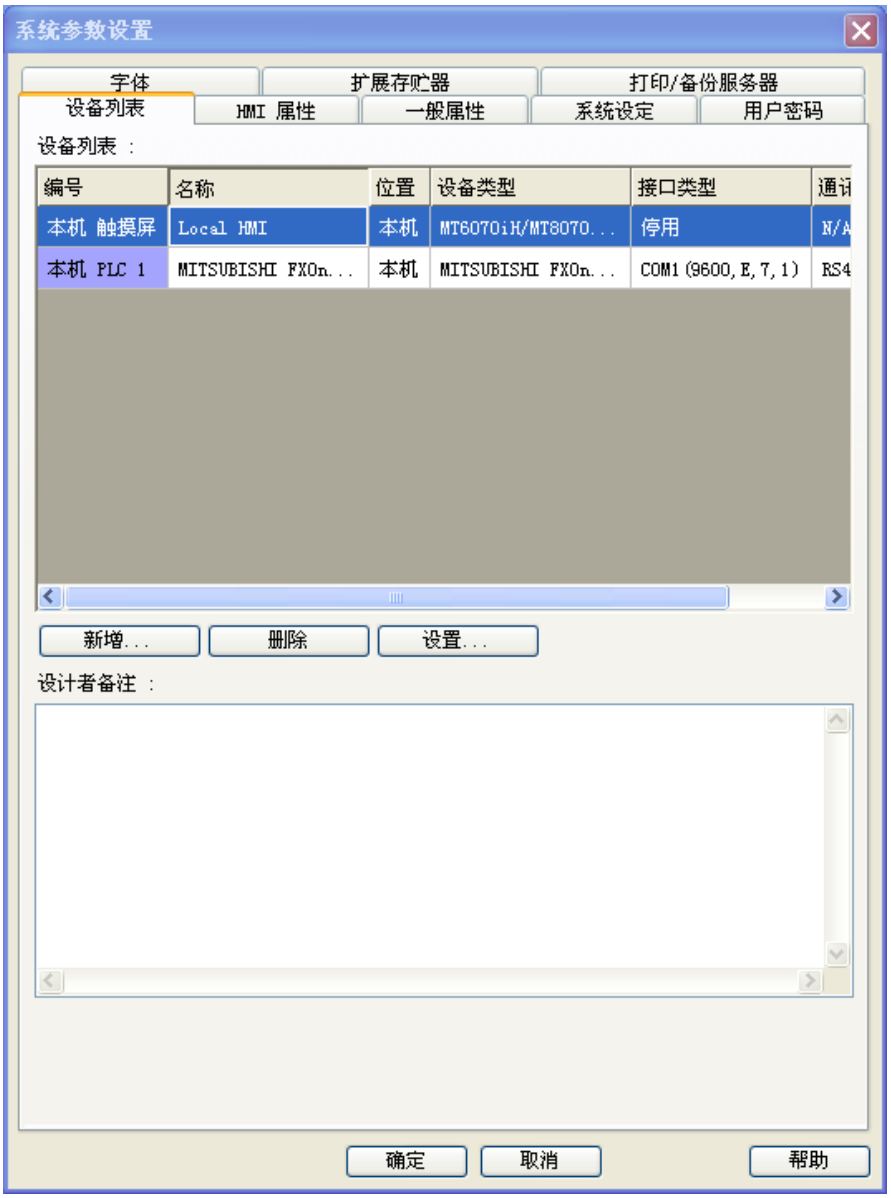
[下载后启动程序画面]

此选项如被勾选，下载程序完成后会重新启动触摸屏。

触控[下载]按钮后将执行下载动作，信息窗口中会显示目前已下载完成的文件。

第五章　系统参数

在选择EB8000窗口[编辑]下的[系统参数设置 …]后，可以得到系统参数设定对话窗口，如下图。

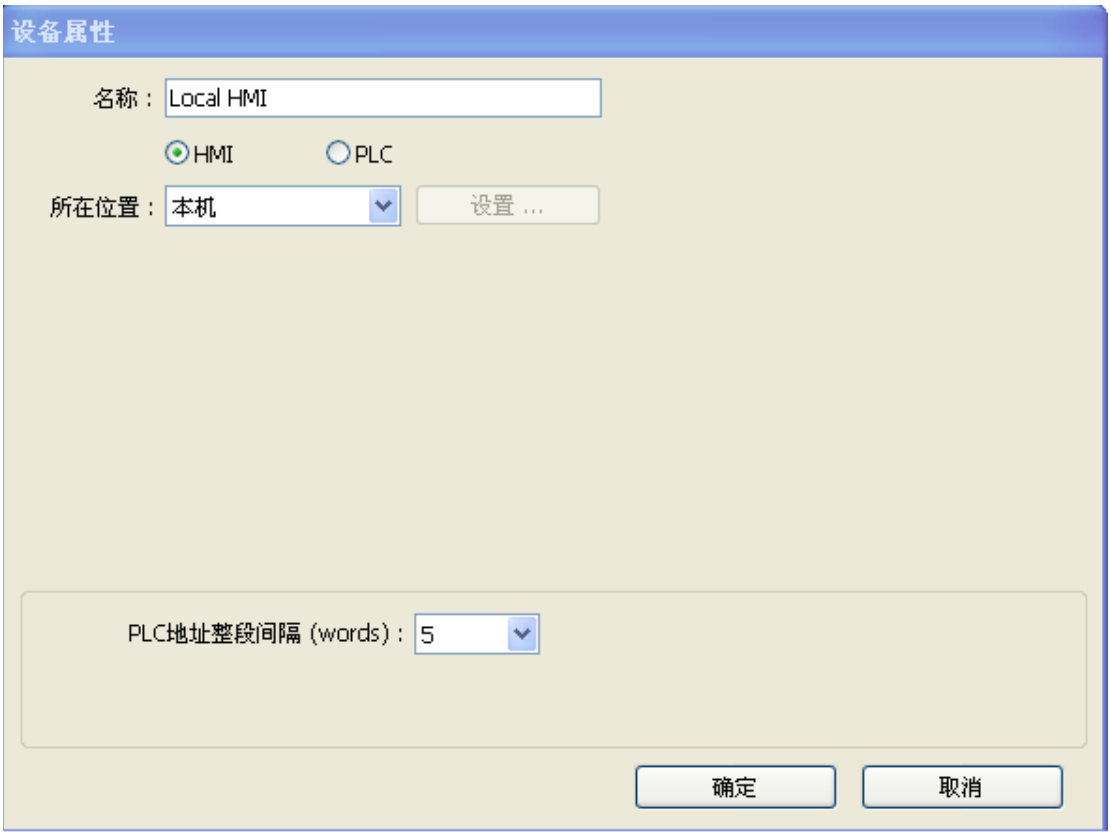


系统参数共分为[设备列表]、[HMI属性]、[一般属性] 、[系统设定]、[用户密码]、[字体]、[扩展存储器] 、[打印/备份服务器]等八个部分，下面将说明各部分的内容。

5.1 设备列表

[设备列表]用来设定被触摸屏所操作的各项设备的属性，这些设备包括各种PLC，远程的触摸屏与PC。

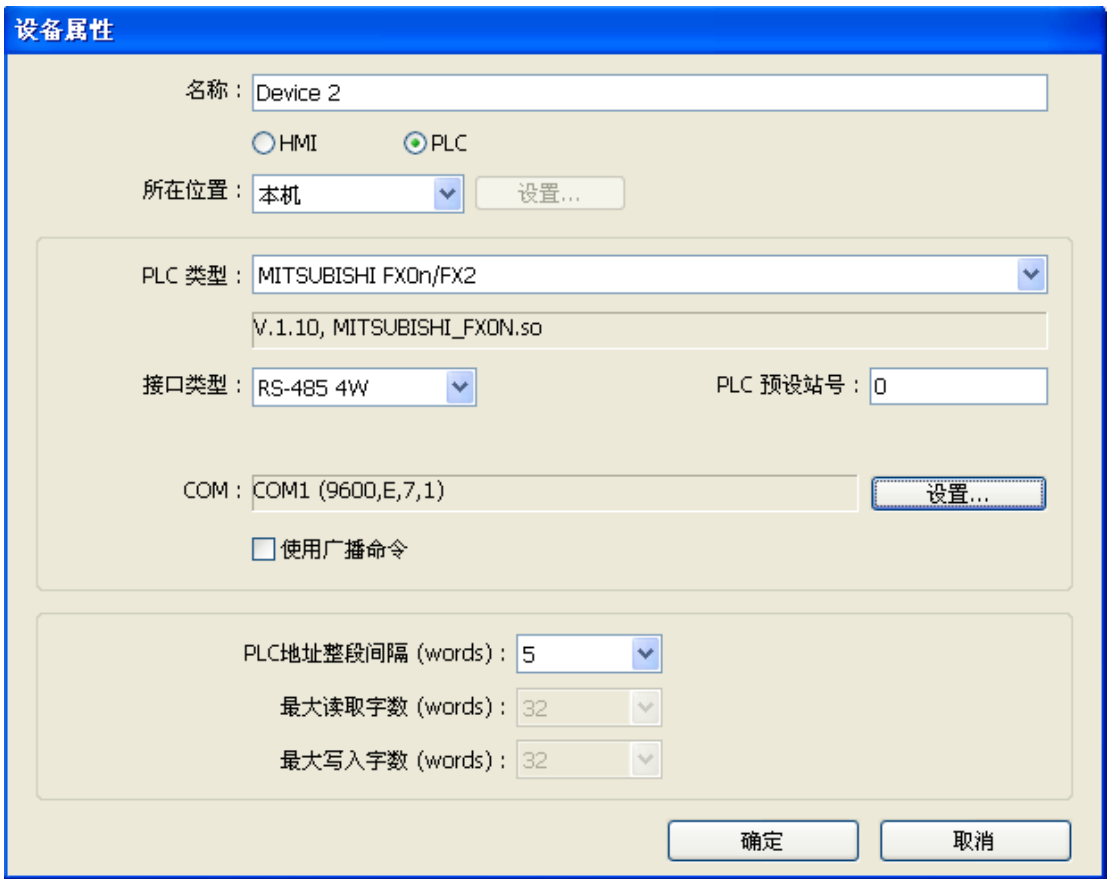
当使用 EB8000开启一个新的MTP文件后，可以发现在“设备列表”中，存在一个预设设备：“Local HMI”，“Local HMI”被用来识别本机，也可称为本地触摸屏，每一个MTP文件至少需包含一个“Local HMI”设备。当使用[设定…]可开启“Local HMI”的设定对话窗口，由图中可以发现“Local HMI”的属性为“HMI”，位置为“本机”。



下面说明新增一台设备的步骤：

(1) 如何控制一台本机PLC

所谓“本机PLC”是指直接连接在本地触摸屏上的PLC，此时要控制本机PLC需先增加此种类型的设备。在触控[新增…]按键后，只需在设定对话窗口中正确设定各项属性即可。



各项设定的说明如下：

[名称]

设备名称。

[HMI]或[PLC]

连接设备为PLC，所以此时选择“PLC”。

[所在位置]

可以选择“本机”或“远端”，因PLC连接在本机触摸屏上，所以此时选择“本机”。

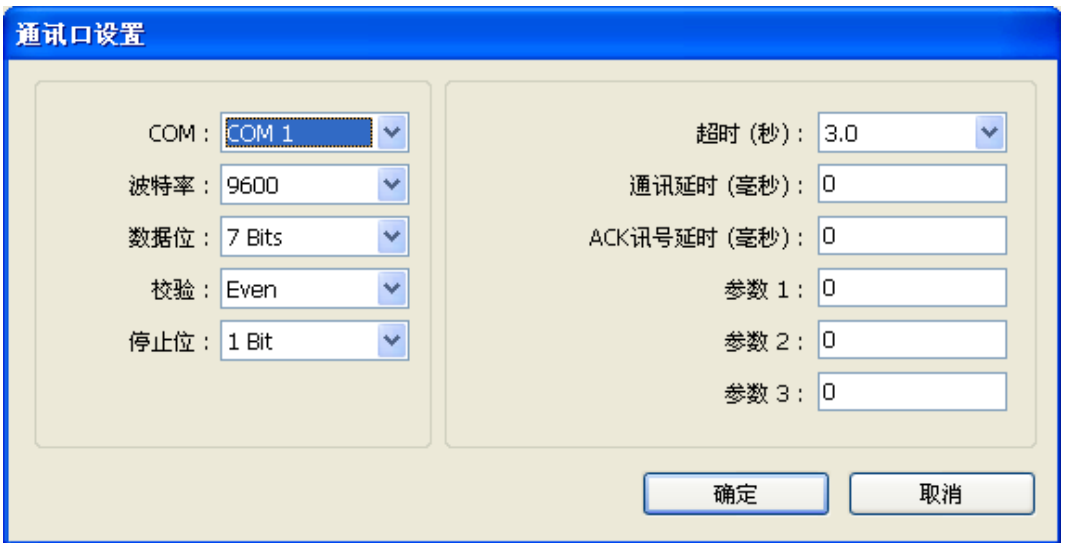
[PLC 种类]

选择PLC的型号。

[接口类型]

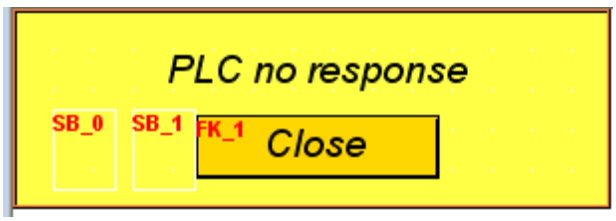
PLC所使用的接口类型，可以选择“RS-232”、“RS-485 2W”、“RS232-485 4W”、“以太网”。

接口类型如果为“RS-232”或“RS-485 2W”或“RS232-485 4W”，触控[设定 …]后可以得到通讯参数设定对话窗口，用户必须设定正确的通讯参数。



[超时]

通讯中断超过此项设定值(单位为秒)时，触摸屏会使用5号窗口作为“PLC No Response”的提示。

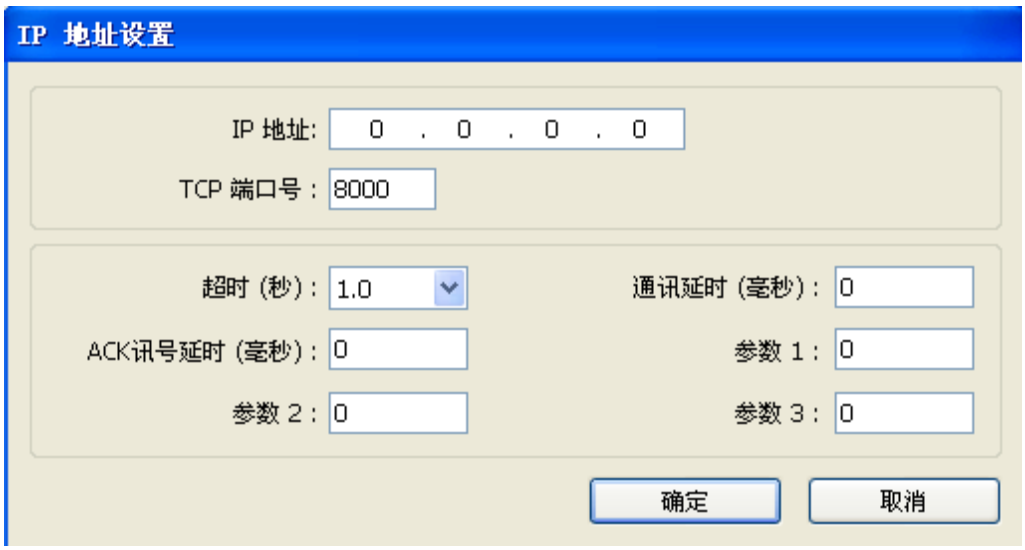


[通讯延时]

触摸屏在送出下一个命令给PLC前，会刻意先延迟此项设定值(单位为毫秒)，再送出命令。此项设定值会降低HMI与PLC间的通讯效率，因此如无特殊需求，此项设定值一般设定为0即可。

注意：如果使用的PLC为SIEMENS S7-200系列，则不能忽略此项设定值，使用此种PLC一般建议将[通讯延时]设定为5，[ACK讯号延时]则设定为30。

接口类型如果为“以太网”，触控[设定…]后可以得到通讯参数设定对话窗口，用户必须正确设定PLC的IP地址与TCP端口号。



接口类型如果为“USB”，则不需做进一步设定，请检查[设备列表]上的各项设定值是否正确。

[PLC 预设站号]

PLC所使用的预设地址站号。当地址内容不包括站号信息时，EB8000将使用此项设定值做为PLC的站号。

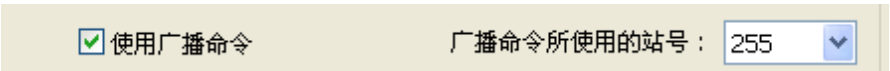
另外，也可将PLC站号信息直接置于地址内容中，此时所使用的地址格式为

ABC#DEFGH

其中ABC表示PLC所使用的站号，此时ABC必须大于等于0，且小于等于255。DEFGH用来指定PLC的地址，两种数据间使用“#”区隔。以下图为例，显示此时将读取站号为3的PLC的X12地址之内容。

[使用广播站号]

勾选此选项后，可以设定广播命令所使用的站号，用户此种站号的命令被视为广播命令。以下图为例，广播命令所使用的站号被设定为255，也就是当地址内容为255#200时，触摸屏将只负责发送此种命令给PLC，并忽略PLC接收到此种命令后对触摸屏所做出的任何回复 (只在Modbus有作用)。



[PLC地址整段间隔 (words)]

不同读取命令的读取地址的间距若小于此项设定值，这些命令可以合并为同一笔命令。此项设定值如果设定为0，将取消命令合并功能。

举例来说，假使此项设定值为5，当分别需从LW3读取1个word与从LW6读取2个word的数据(即读取LW6与LW7的内容)时，因LW3与LW6的地址差距小于5，此时可以将此两个命令合并为1个命令，合并后的命令内容为从LW3开始连续读取5个word的数据(读取LW3~LW7)。需注意，可以被合并的命令的读取数据大小将不会大于[最大读取字数 (words)]

[最大读取字数 (words)]

一次可以从设备读取数据的最大量，单位为word。

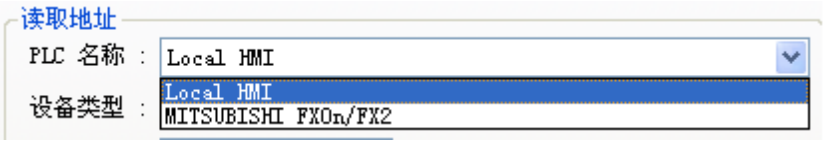
[最大写入字数 (words)]

一次可以写入到设备的数据的最大量，单位为word。

可以利用更改安装目录下的devicetype.def的内容，来改变[最大读取字数 (words)]与[最大写入字数 (words)]的设定值，但更改此两项设定值需配合设备的特性，不适当的更改将引起通讯失效。

完成上述的各项设定后，在设备列表中可以发现新增了一个名称为“本机PLC 1”的设备。

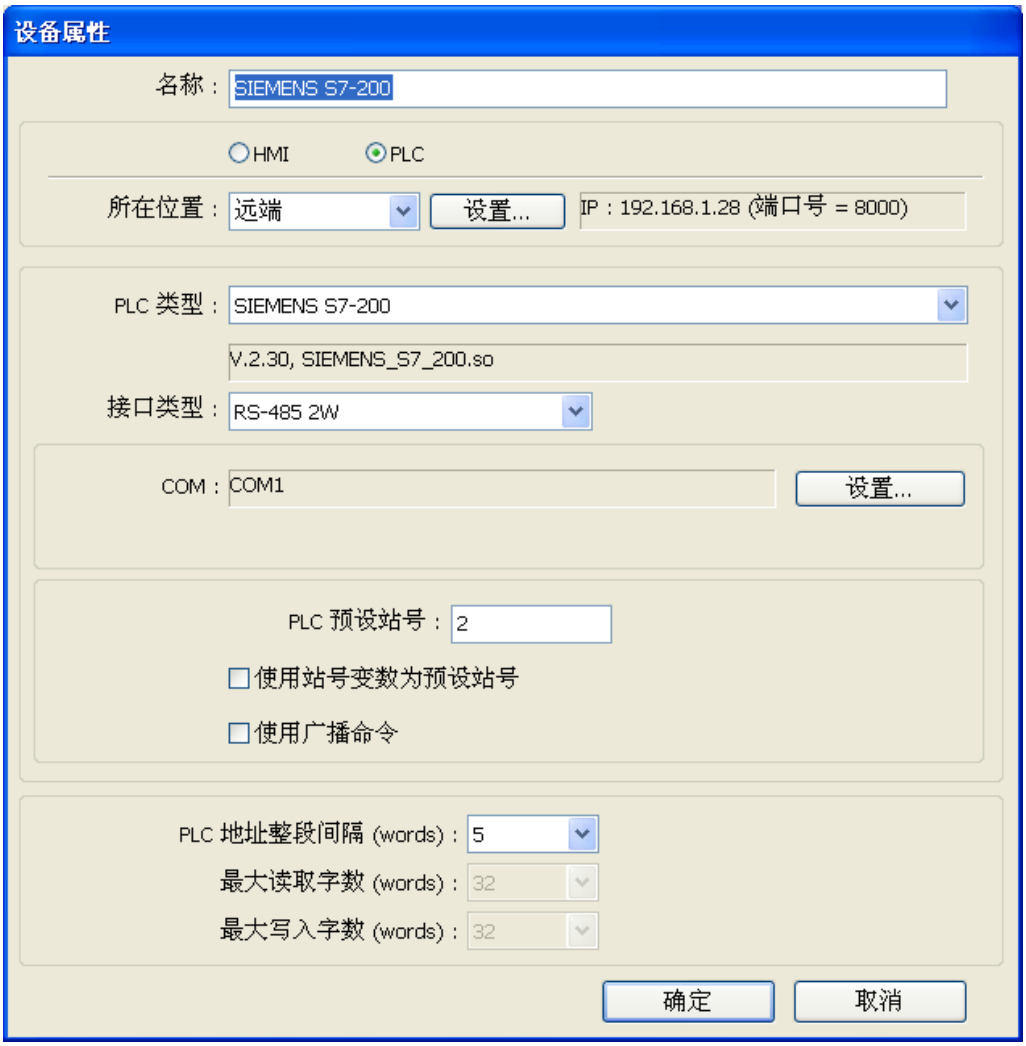
此时在所有元件的[PLC 名称]选项中，也可发现增加了“MITSUBISHI Fx0n/FX2”的选项，选择这个项目即可操作、修改此台PLC上的数据。



(2) 如何控制一台远端PLC(remote PLC)

所谓“远端PLC”是指直接连接在远端HMI的PLC，此时要控制远端PLC需先增加此种类型的设备。在触控[新增…]按键后，只需在设定对话窗口中正确设定各项属性即可。

以下以远端PLC：siemens S7-200为例：



各项设定的说明如下：

[名称]

设备名称。

[HMI]或[PLC]

连接设备为PLC，所以此时选择“PLC”。

[所在位置]

可以选择“本机”或“远端”，因PLC连接在远端触摸屏上，所以此时选择“远端”，并且必须设定远端触摸屏的IP地址。在触控[所在位置]的[设置…]按钮后，可以设定远端触摸屏的IP地址和端口号。

[PLC 种类]

选择远端PLC的型号。

[接口类型]

远端PLC所使用的接口类型，当远端PLC使用COM端口时，接口需选择“RS-232”，“RS-485 2W”，“RS485 4W”任一种。

[PLC 预设站号]

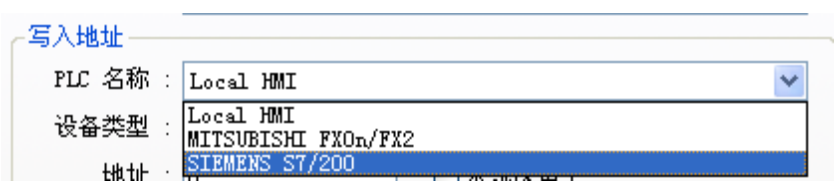
远端PLC所使用的站号。

[COM]

远端PLC在远端HMI上所使用的COM端口，此项内容必须正确设定。

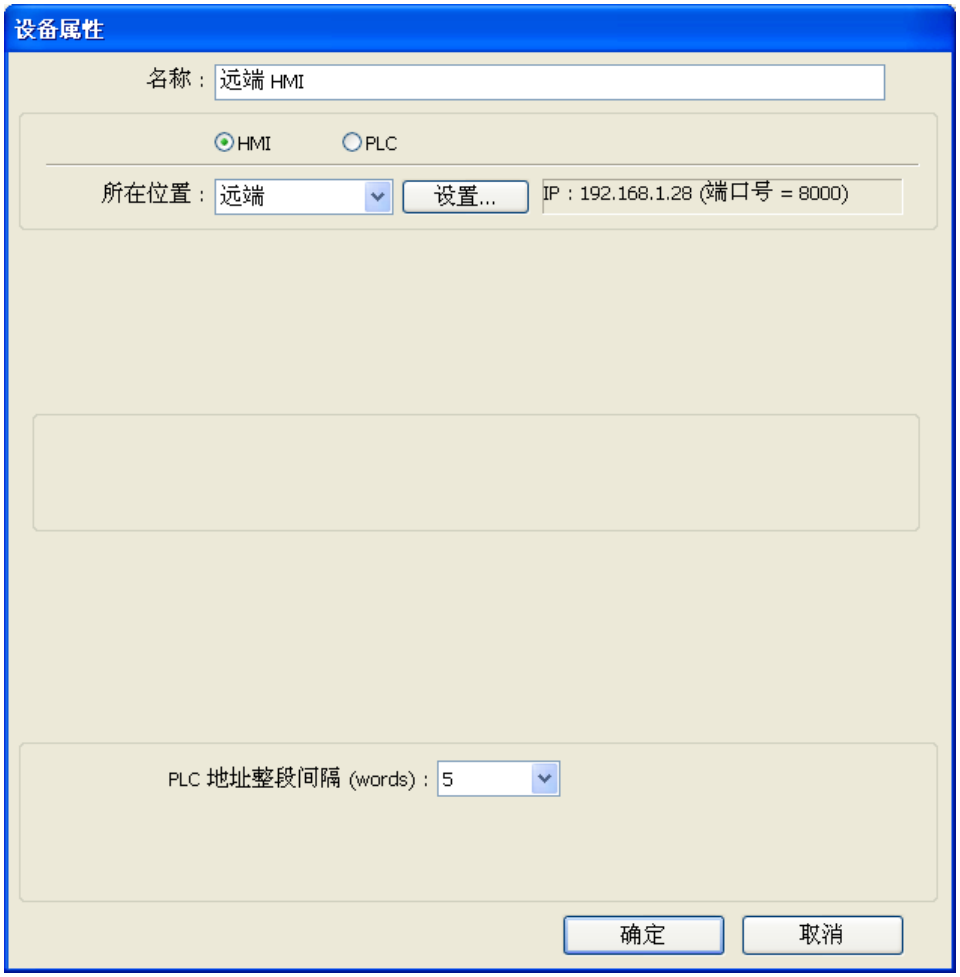
完成上述的各项设定后，在设备列表中可以发现新增了一个名称为“远端 PLC 1”的设备。

此时在所有元件的[PLC 名称]选项中，将可发现这些已完成设定的设备，选择指定的设备即可操作这些PLC。



(3) 如何控制一台远端HMI (remote HMI)

所谓“远端HMI”是指非本地HMI的所有其它HMI，PC也被视为远端HMI的一种，此时要控制远端HMI需先增加此种类型的设备。在触控[新增…]按键后，只需在设定对话窗口中正确设定各项属性即可。



各项设定的说明如下：

[名称]

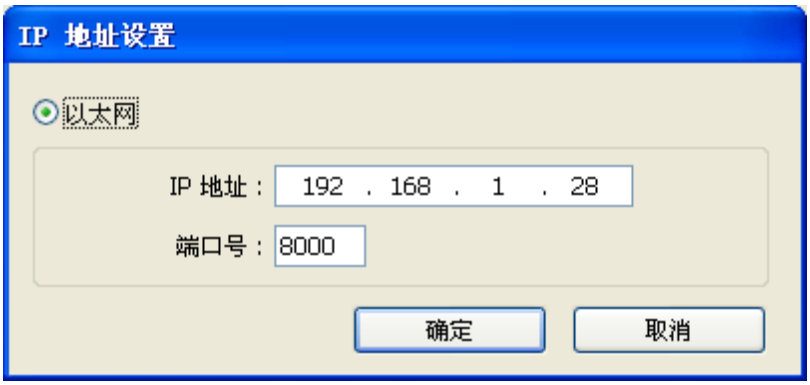
设备名称。

[HMI]或[PLC]

连接设备为HMI，所以此时选择“HMI”。

[所在位置]

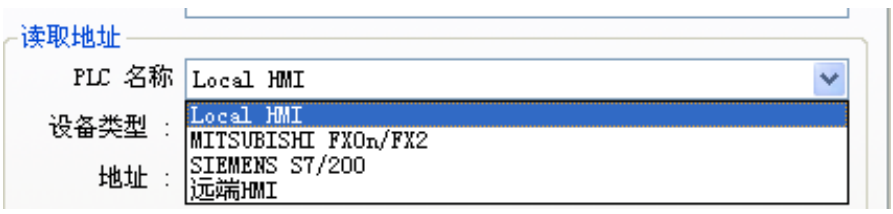
可以选择“本机”或“远端”，此时选择“远端”，并且必须设定远端HMI的地址。在触控[所在位置]的[设定…]后，可以设定远端HMI的IP地址，如下图。此时也必须正确设定连接端口，远端HMI的连接端口在开启远端HMI所使用的MTP文件后，检视系统参数中的[HMI属性]即可发现。



完成上述的各项设定后，在设备列表可以发现新增了一个名称为“远端 触摸屏 1”的设备

编号	名称	位置	设备类型
本机 触摸屏	Local HMI	本机	MT6070iH/I
本机 PLC 1	MITSUBISHI FX0n...	本机	MITSUBISH
远端 PLC 1	SIEMENS S7/200	远端(IP:192.168.1.28, 端口号=8000)	SIEMENS S
远端 触摸屏 1	远端 HMI	远端(IP:192.168.1.28, 端口号=8000)	MT8xxx

此时在所有元件的[PLC 名称]选项中，也可发现增加了“远端 HMI”的选项，选择这个项目即可读写“远端 HMI”上的数据。



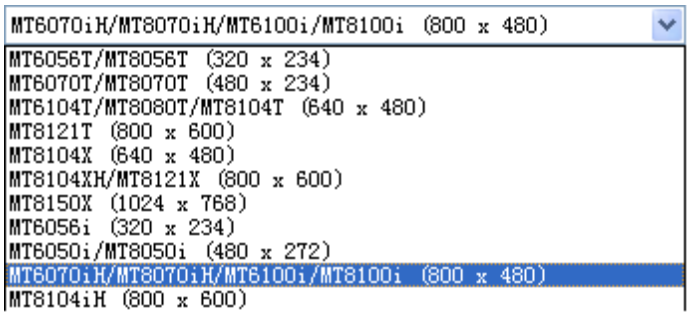
5.2 HMI属性

[HMI属性]设定页用来设定HMI的机型，“时钟”的来源以及打印机有关的设定。

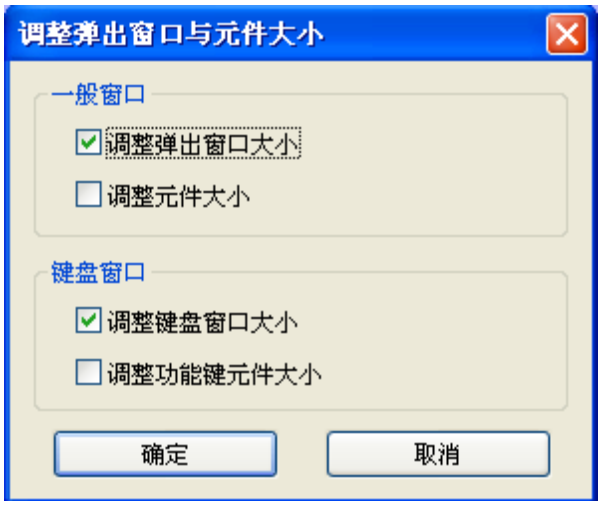
各项设定的说明如下：

[HMI 型号]

选择目前所使用的机型，可选择内容如下图。



当用户更改别的触摸屏型号并按确认键时，会弹出“调整弹出窗口与元件大小”的窗口，如下图：



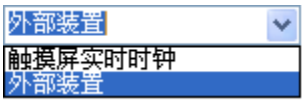
[HMI 站号]

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。

[时钟来源]



设定时间信号的来源, 被用在[资料取样], [事件登录]等需时间记录的元件上。

当选择“触摸屏实时时钟”时, 表示时间信号来自触摸屏上内含的时钟。当选择“外部装置”时, 表示时间信号来自外部的设备, 此时需正确设定时间信号的来源地址, 以下图的设定为例, 表示时间来自“本机PLC”的TV, 此时地址TV从地址0开始的连续6个word内容分别存放下列的信息:

TV 0 –> 秒

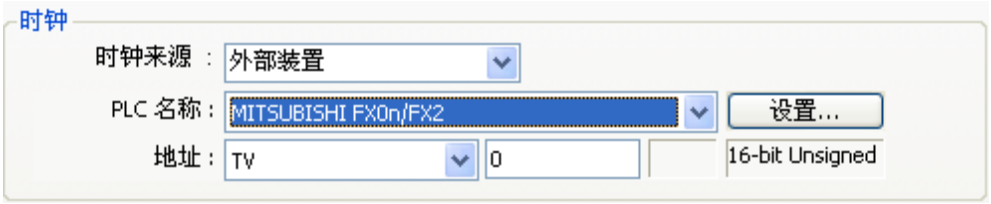
TV 1 –> 分

TV 2 –> 时

TV 3 –> 日

TV 4 –> 月

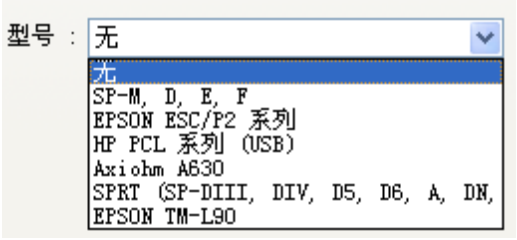
TV 5 –> 年



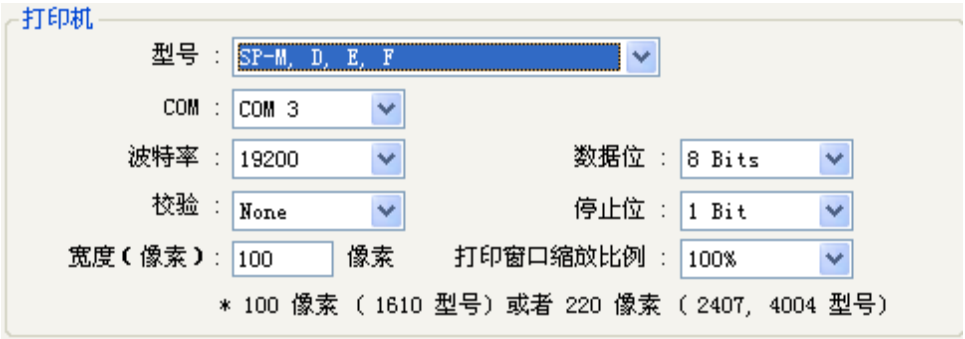
[打印机]

[型号]

显示目前支持的打印机类型，其中HP PCL Series需使用USB端口连接，其它类型打印机需使用COM端口连接。更详细的机型描述请参考“第二十三章 MT8000支持的打印机型号”



使用COM端口连接的打印机需正确设定COM端口的通讯参数。当打印机的型号为SP-M, D, E,F时，需小心设定 [宽度（像素）]此项设定值，此设定值不可以超过打印机每行可以打印的点数，否则将造成错误的打印结果。



[储存空间分配(以i系列HMI为例全部储存空间:128MB)]

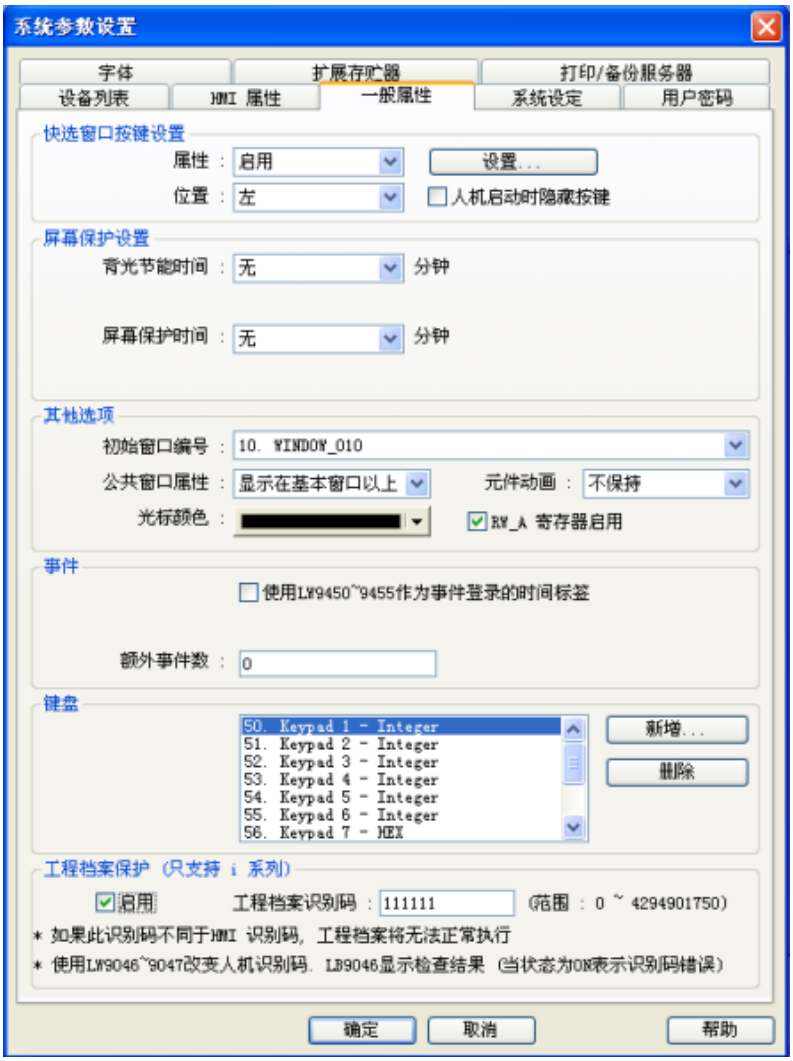
1) i系列运行的操作系统（OS）占用空间：64MB；

2) 工程文件（Project）占用空间为：16MB；

3) 历史数据文件：48MB，包括事件记录，资料取样记录等；

5.3 一般属性

[一般属性]设定页用来设定与画面操作有关的各项属性。

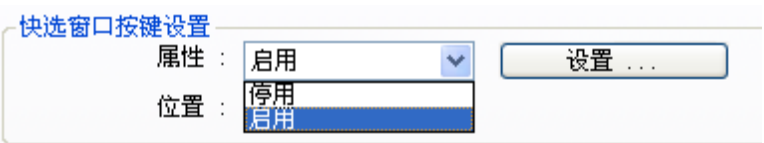


各项设定的说明如下：

[快选窗口按键设置]

设定快选窗口(3号窗口)的各项属性，要使用快选窗口前需先建立3号窗口。

a. [属性]



选择是否使用快选窗口，在选择“启用”后可以使用[设置…]功能，设定快选窗口按键的各项属性，包括颜色与标示其上的文字。

b. [位置]

选择快选窗口按钮的出现位置，选择“左”则快选窗口按钮出现在画面的左下角； 选择“右”则快选窗口按钮出现在画面的右下角。

[屏幕保护设定]

a. [背光节能时间]

当未触控屏幕的持续时间等于此设定值时，将关闭背光灯，设定的时间单位为分钟。关闭背光灯后只需触控到屏幕，即可重开背光灯。当设定值选择“无”时，HMI将不使用背光节能的功能。

b. [屏幕保护时间]

当未触控屏幕的持续时间等于此设定值时，将自动切换到[屏幕保护使用窗口]所指定的窗口，设定的时间单位为分钟。当设定值选择“无”时，HMI将不使用屏幕保护的功能。

c. [屏幕保护使用窗口]

指定屏幕保护功能执行时要切换的页面。

[其它选项]

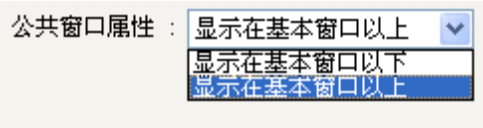
a. [初始窗口编号]

选择HMI开机后的起始页面。

b. [额外事件数]

系统预设的事件记录总数为1000笔，如要增加记录的笔数，可以更改此项设定值，目前设定的上限为10000。

c. [公共窗口属性]

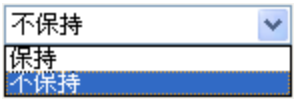


公共窗口(4号窗口)内的元件会出现在每个基本窗口中, 此选项用来选择公共窗口内的元件是出现在基本窗口原来元件的上层或下层。

d. [光标颜色]

设定在使用[数字输入]或[文字输入]元件时, 在输入项中出现的光标颜色。

e. [元件动画]

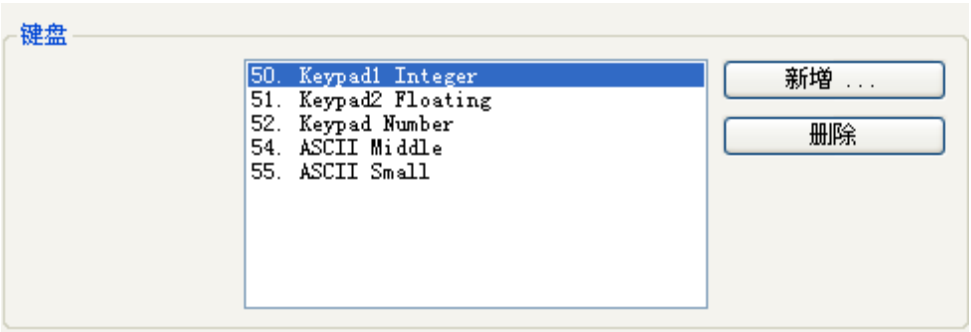


如果选择 “保持” 模式,则HMI在运作时, [动画]与[移动图形]元件将显示在其它类型元件的上方, 与元件的建立顺序无关。如果选择 “不保持” 模式, 则元件的显示顺序依照元件的建立先后, 先建立者先出现。

f. [RW_A 寄存器启用]

是否启用配方资料RW_A。需在启用RW_A后, 元件才可以操作RW_A的内容, RW_A的大小为64K。

[键盘]



显示键盘存在的页面, 这些页面代表在使用[数值输入]与[文字输入]元件时, 可以选择的键盘类型, 最多可以新增到32个键盘。

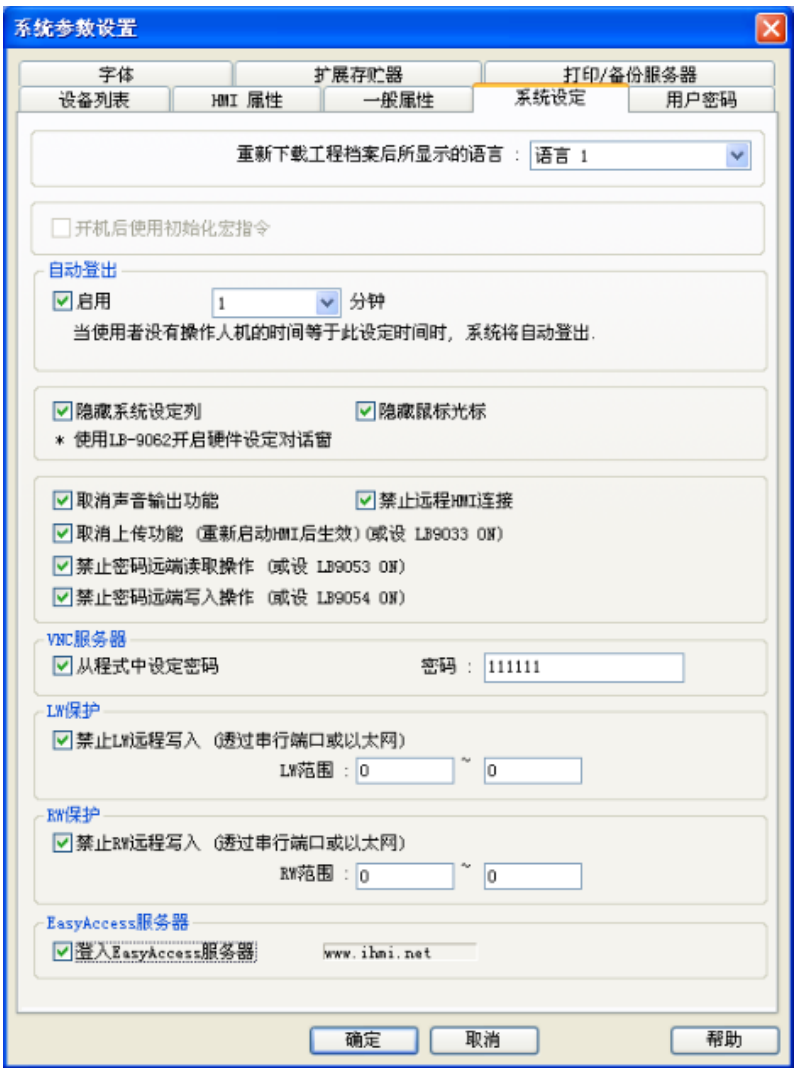
用户要建立自订的键盘时, 需先在已存在的页面规划好要使用的键盘, 并使用[新增 …]功能选择这些页面并加入到列表中即可。更详细的功能请参考 “键盘的设计与使用” 这个章节。若用户想要将键盘固定在画面上,不需要将键盘加入,用法请参考第十二章。

[工程档案保护(只支持i系列)]

用户的工程文件可被限定在特定的触摸屏上执行(只限i系列触摸屏),相关设定详见《第三十章 工程档案保护功能》

5.4 系统设定

此项用来设定EB8000中各式各样的功能。



“系统设定”中有些功能是从系统寄存器中复制而来,例如:隐藏鼠标光标(LB9018)、取消声音输入功能(LB9019)、隐藏系统设定列(LB9020)、取消上传功能(LB9033)以及禁止远端HMI连接(LB9044)。也就是说,用户可以选择系统寄存器来实现这些功能。要使用系统寄存器,用户可以在新增元件时,在设定页勾选“系统寄存器”并选择相应的设备类型。若要检视所有的系统寄存器,可以在EB8000“图库”下拉菜单中“地址标签库”的系统寄存器中查看。

[重新下载工程文件后所显示的语言]

选择在重新下载工程文件并启动触摸屏时所显示的语言。

[开机后使用初始化宏指令]

指定当触摸屏开机后所执行的宏指令。

[自动登出]

当用户没有操作触摸屏的时间等于此设定时间时，密码防护等级将自动登出。

[VNC服务器]

设定登入VNC服务器所使用的密码。

[LW保护]和[RW保护]

如果用户勾选“禁止LW/RW远程写入”，并在“LW/RW范围”内设定了所要保护的范围，此范围内的数据将不能通过远端的触摸屏进行调整。

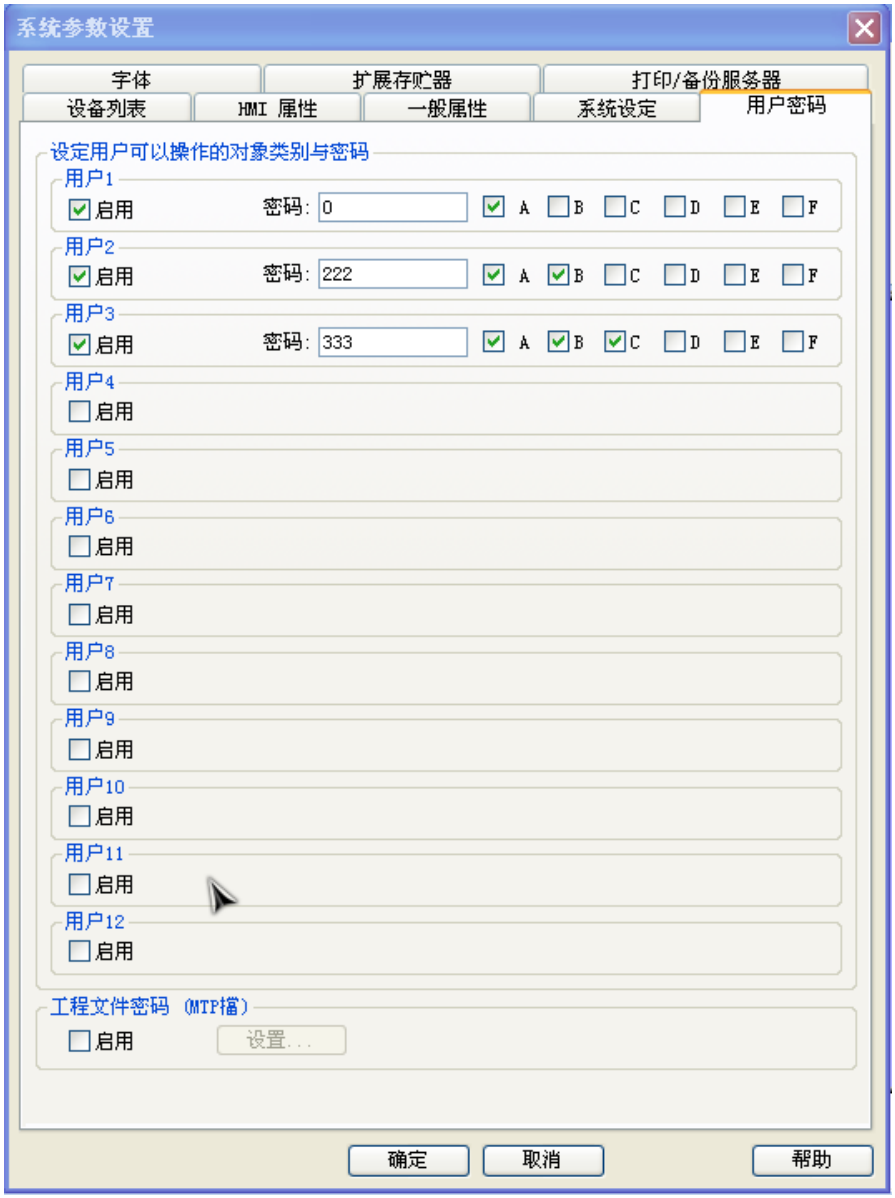
[Easy Access服务器]

通过这项技术，用户可以轻易的透过网络检测网络连接的MT8000i/X系列触摸屏，并直接在电脑上操作，就像直接将触摸屏拿在手上一样。

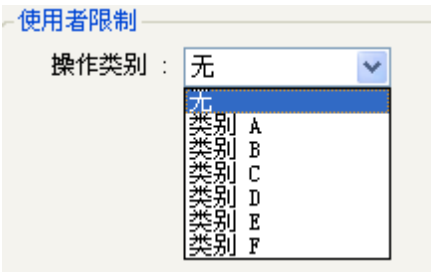
特别的是，Easy Access不需要传输更新的图片，只需传输即时资料，这使得传输更快、更有效率。更详细的资料请参阅《Easy Access》说明文档。

5.5 用户密码

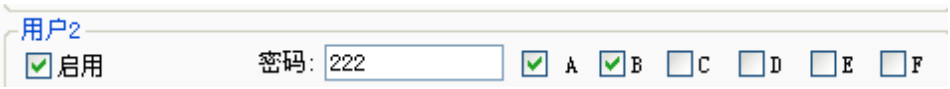
系统参数中的[用户密码]设定页用来设定用户的密码，并规划每个用户可操作的元件类别，在EB8000中，元件被划分为“无”与“A~F”等共7个类别。用户的密码必须是由0~9的数字所组成，B8000最多可规划12个用户。



HMI运作时，用户在成功输入密码后，EB8000会依照用户的设定内容决定用户可以操作的元件类别。在工程文件中，元件的类别被区分为“无”与“类别A”至“类别F”共7种，元件可以设定所属的类别，参考下图。类别属于“无”的元件，开放给所有用户使用。



当“用户2”的设定内容如下时，则该用户只被允许使用类别属于“无”与A、B的元件。详细说明请参考相关章节[元件安全防护]。



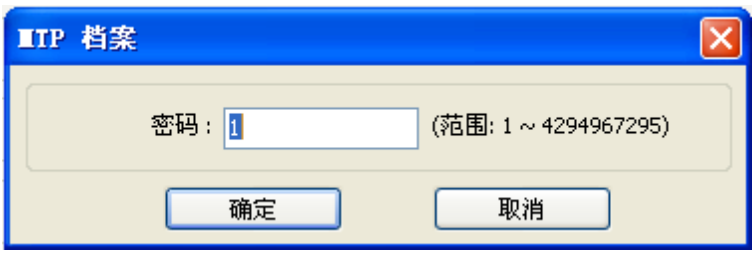
[工程文件密码]

用户可以设定保护MTP文件的密码。

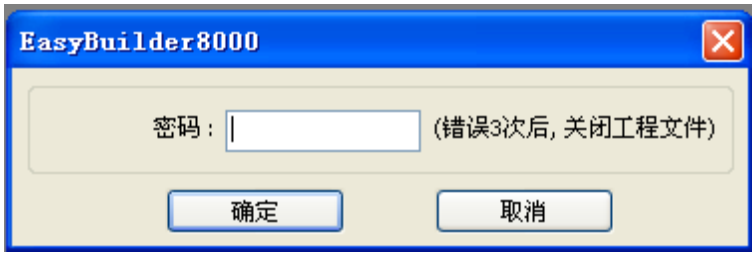
若有设定此密码，当用户想要编辑MTP文件时，必须输入这个密码才可以打开编辑。

密码的设定范围是1～4294967295。

勾选“启用”，再按“设置”按钮即可弹出如下对话框：

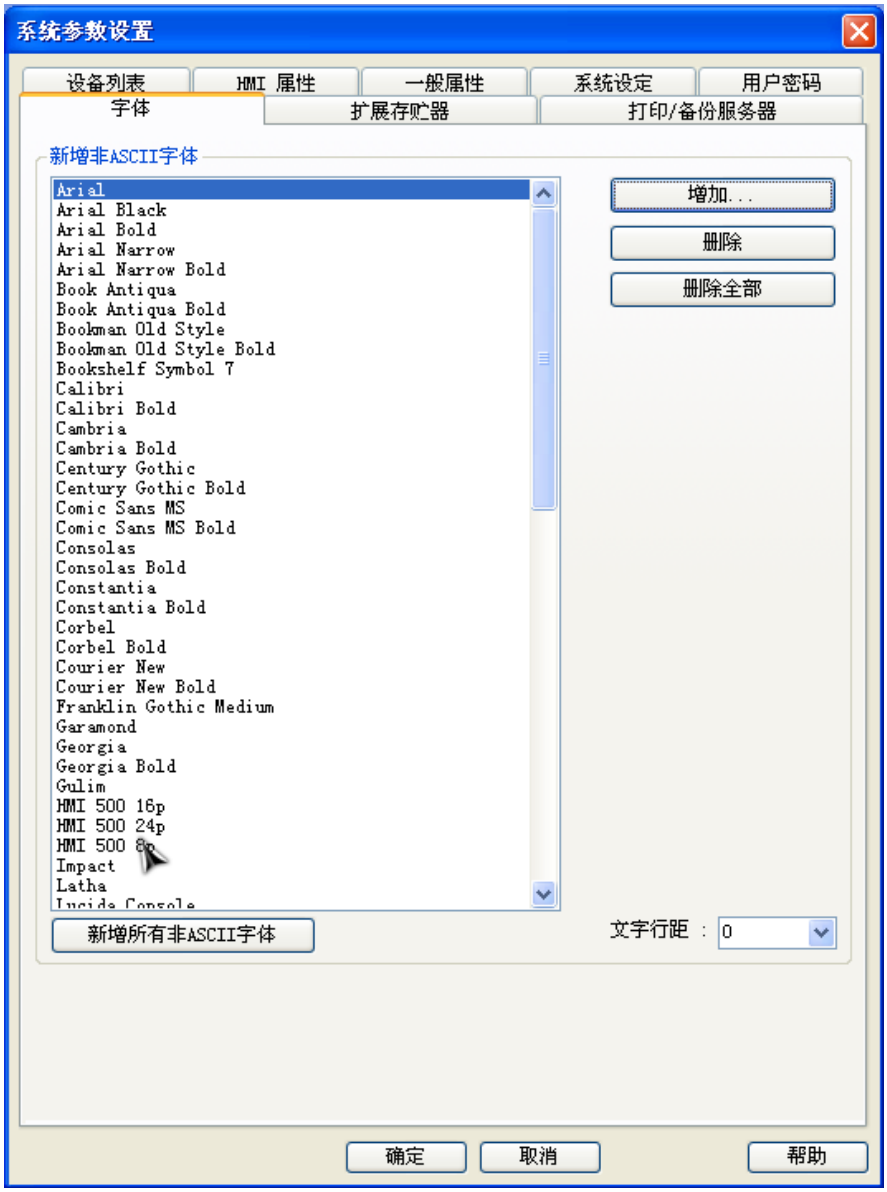


完成设定后，再打开该工程文件时，会弹出下面的对话框，要求用户输入密码才能进入工程文件。



5.6 字体

此设定页用来设定非ASCII文字所使用的字型。



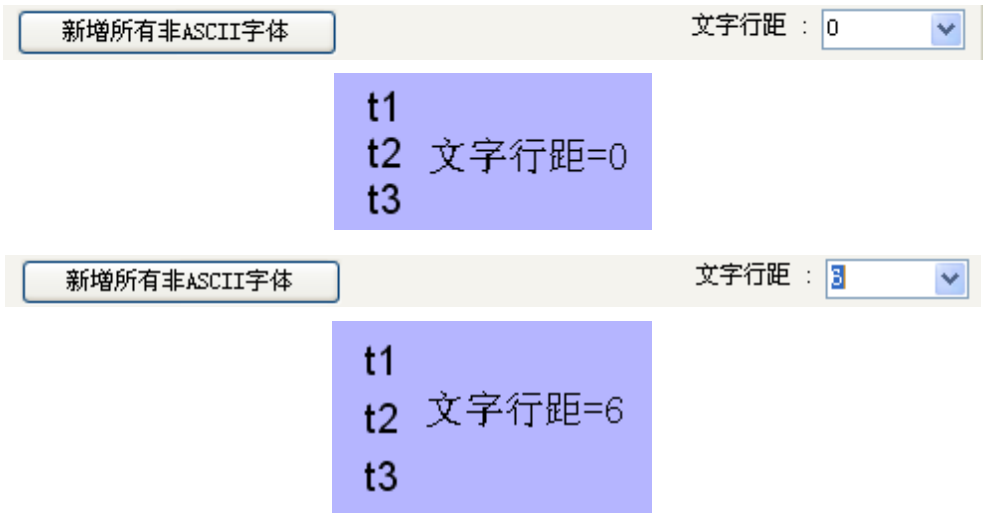
[新增非ASCII字体]

在此项目表列出的字型为非ASCII的文字所使用的字型。当用户使用了非ASCII的文字或双字符文字(例如简体中文字、繁体中文字、日文、韩文等), 并使用了非表列中的字型时, EB8000会自动为这种文字选用表列中的字型。

用户可以测试WINDOWS有哪些包含非ASCII文字的字型可以使用在EB8000中, 并将其加入到[非ASCII字体列表]中。

[文字行距]

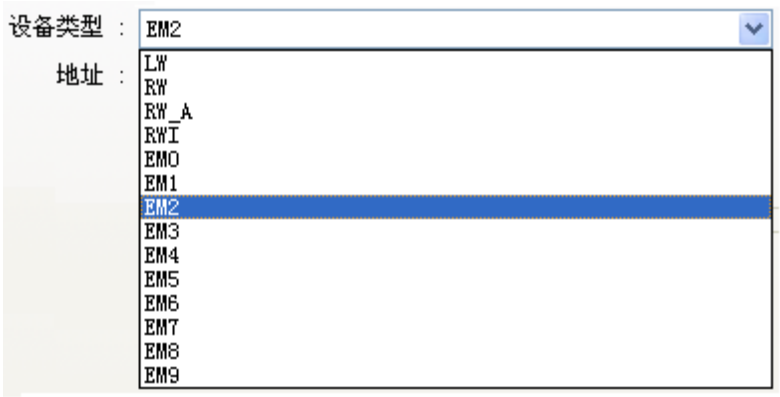
决定行与行之间的间距。



5.7 扩展存储器

此页用来设定扩展内存的位置

扩展内存包含EM0~EM9,使用方式与其它HMI上的设备类型相似(类似使用LW或RW地址类型),使用者只需在设备类型中指定使用EM0~EM9即可,每个扩展内存最多可以存放2G word的资料。



扩展内存中的数据使用文件的形式存放在SD卡、U盘1或U盘2上，EM0~EM9所使用的文件名称分别为em0.emi~em9.emi，用户可以使用RecipeEditor.exe开启这些文件并编辑扩展内存中的数据。

扩展内存中的数据不会因HMI断电而消失，也就是下次开机后，扩展内存中的数据会恢复为关机前的状态，与配方数据(RW、RW_A)类似。较特别的是用户可以指定扩展内存的位置，可以选择SD卡、U盘1或U盘2。

当作为扩展内存的设备不存在时，此时若读取扩展内存中的数据，数据内容将一律为0；当作为扩展内存的设备不存在时，此时若将数据写入到扩展内存中，HMI将作出“PLC no response”的反应。

MT8000支持“热插拔”功能，在HMI不需断电的情况下，可以随时插上或移除SD卡、U盘1或U盘2。用户可以利用这项特性，更新或撷取扩展内存中的数据。

5.8 打印/备份服务器

此页用来设定MT远端打印机的相关数据

输出设定：

[方向]

可设定文字或图片要以水平或垂直方式输出

[打印大小]

可设定输出时原始大小或是配合打印机边界

[边界]

可设定纸张的上、下、左、右的边界

通讯设定：

[IP 地址]

用户需设定正确

[端口]

用户需设定正确

[使用者名称]

用户自行设定

[密码]

用户自行设定

※ 详情请参阅《第二十六章 EasyPrinter》

第六章　窗口

窗口是构成MT8000触摸屏画面的基本元素，也是一个重要的元素。有了窗口后，画面上的各个元件、图形、文字等信息才可以显示在触摸屏上。一般的工程文件中，不会只有一个窗口，所以一个功能需要建立多个窗口。EB8000提供了3～1999、总共1997个窗口。而每一个工程文件具体能够使用的窗口数量，由所有建立的窗口数量的容量不大于MT8000触摸屏所能存储画面容量大小所决定。例如：MT8000 i系列触摸屏的画面存储容量为16MB，则所编写的画面工程容量只要不超过16MB，则可以建立尽可能多的画面窗口。

6.1 窗口类型

依照功能与使用方式的不同，EB8000将窗口分为下列四种类型：

(1) 基本窗口

(2) 公用窗口

(3) 快选窗口

(4) 系统信息窗口

依次说明如下：

(1) 基本窗口 (base window)

这是最常见的窗口，一般被当作主画面的用途之外，也被用在：

a. 底层窗口，可提供其它窗口作为背景窗口。

b. 键盘窗口。

c. [功能键]元件所使用的弹出窗口。

d. [间接窗口]与[直接窗口]元件所使用的弹出窗口。

e. 屏幕保护画面。

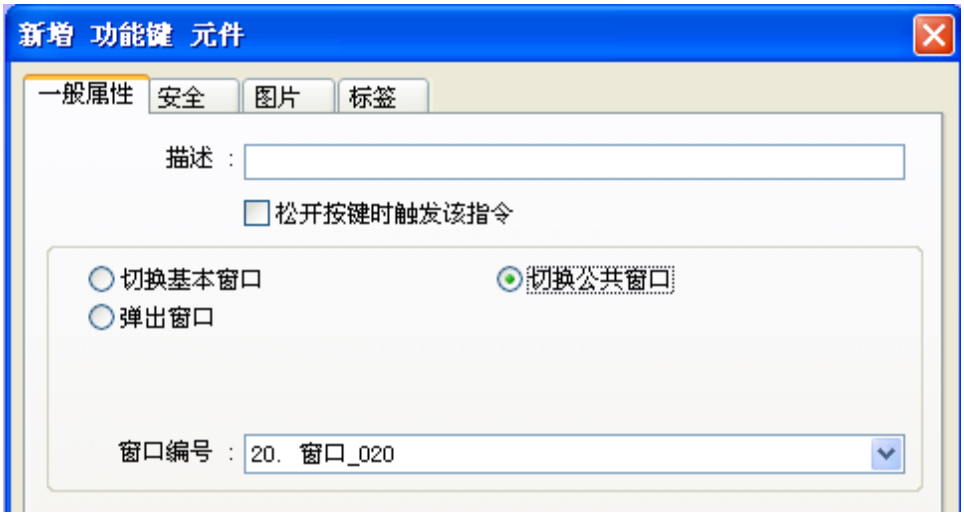
基本窗口必须与触摸屏的屏幕大小一样，也就是说，基本窗口的分辨率需要与触摸屏的分辨率一致。

下图显示开机时的画面，此时的窗口即是基本窗口

(2) 公用窗口 (common window)

4号窗口为预设的公用窗口，此窗口中的元件也会出现在其它窗口中，因此通常会将各窗口共享的元件放置在公用窗口中。例如产品的logo图片，或者某些公用的按键等。

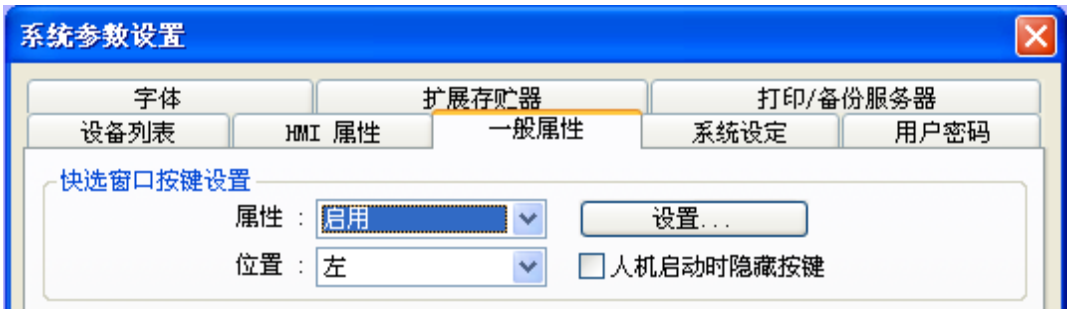
系统运作时可以使用[功能键]的[切换公用窗口]([Change common window])模式，更改公用窗口的来源，例如可将公用窗口由4号窗口更改为20号窗口。



(3) 快选窗口 (fast selection window)

3号窗口为快选窗口，此种窗口可以与基本窗口同时存在，因此一般被用在放置常用的工作按钮，如下图。

要使用快选窗口除了需先建立3号窗口外，需再设定“快选窗口按钮”(fast Selection button)的各项功能，上图的[Shortcut]按钮即是快选窗口按钮，用来切换快选窗口的出现与消失，快选窗口按钮的各项设定在系统参数中，参考下图。



除了可以使用快选窗口按钮来切换快选窗口的出现与消失之外，系统保留寄存器也提供下列地址，让用户可以利用对地址中数据的操作，控制快选窗口与快选窗口按钮。相关的系统保留寄存器如下，更详细的内容请参考《第二十二章 HMI 状态监控(系统保留寄存器地址)》。

[LB9013]　　隐藏/显示快选窗口

[LB9014]　　隐藏/显示快选按键

[LB9015]　　隐藏/显示快选窗口/按键

(4) 系统信息窗口(system message window)

5号窗口、6号窗口、7号窗口与8号窗口为系统预设的信息窗口。

5号窗口为“PLC NO Response”信息窗口，当无法接收到来自PLC的信号时，系统将自动弹出此窗口。

6号窗口为“HMI Connection”信息窗口，当无法连接到远程的HMI时，系统将自动跳出此窗口。

7号窗口为“Password Restriction”信息窗口，当用户的操作权限不足以操作正要操作的元件时，会依元件的设定内容，决定是否弹出此窗口作为警示用途。

8号窗口为“Storage Space Insufficient”信息窗口，当HMI内存、U盘或CF卡/SD卡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。用户也可以使用下列的系统保留寄存器检视HMI内存、U盘或CF卡/SD卡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目前的可用空间(单位 K bytes)

[LW 9076]　　USB 1目前的可用空间(单位 K bytes)

[LW 9078]　　USB 2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保留寄存器检视是哪一个类型的储存设备空间不足。

[LB 9035]　　HMI 可用空间不足警示(当状态为ON)

[LB 9036]　　SD可用空间不足警示(当状态为ON)

[LB 9037]　　USB 1 可用空间不足警示(当状态为ON)

[LB 9038]　　USB 2 可用空间不足警示(当状态为ON)

5号窗口到8号窗口里面的内容提示，可以根据自己的实际需要来修改。例如：5号窗口的显示内容

“PLC No Responds”, 我们可以改为“触摸屏与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, 方便操作人员识别故障信息。

注意:

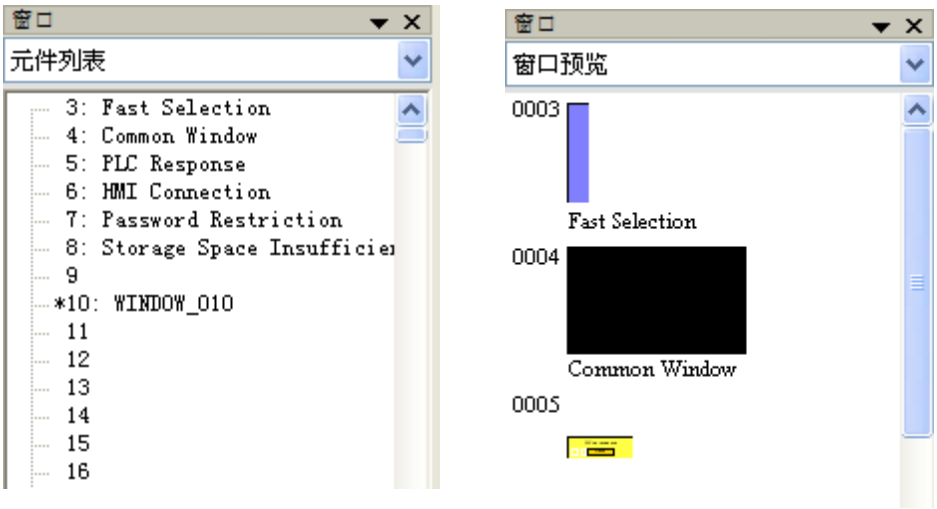
1. 最多可同时打开16个弹出窗口, 包含系统讯息窗口,直接窗口及间接窗口。

2. 同一个窗口只能同时打开一次, 因此不能在同一个基本窗口上使用2个直接(或间接)窗口打开同一个窗口。

3. 窗口3–9为系统内部使用, 窗口10–1999为使用者可任意操作窗口。

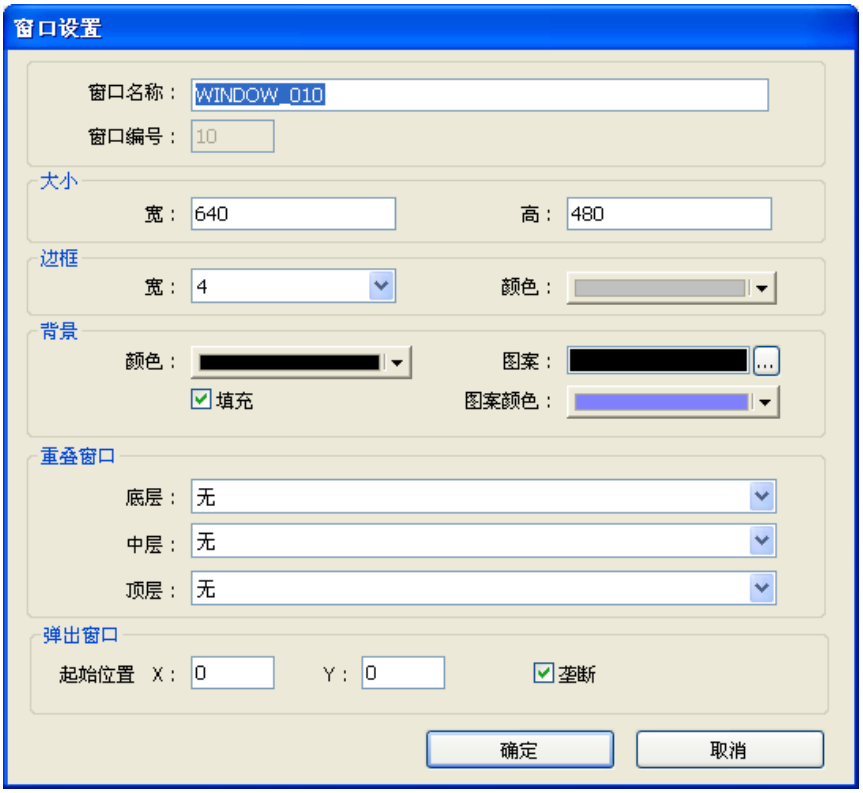
6.2 窗口的建立、删除与设定

EB8000的软件中窗口的建立, 可以通过编辑画面左边的视窗列表来查看, 如下图所示: 窗口列表有两种查看方式, 如果将下图的“元件列表”显示方式改为“窗口预览”方式, 则每个窗口会以图片的方式来显示窗口里面的内容, 下面详细说明如何建立和设定这些的窗口:



(1) 窗口的建立

建立窗口有两种方式, 第一种方式是在窗口树状图上选择要建立的窗口, 并触控鼠标的右键, 在窗口出现后选择[新增]将出现设定对话窗, 在完成各项设定并触控确认键后, 即可建立新的窗口, 参考下图。



[窗口名称]

窗口编好后所显示的名称，参考下图。一般以容易读懂和记住为原则，例如：手动操作画面，自动操作画面等。

[窗口编号]

窗口编号, 从3~1999。

尺寸大小设定项目

[宽]、[高]窗口大小。一般基本窗口的分辨率与所选用的触摸屏的分辨率一样。例如: 使用的触摸屏是MT6100i, 他的分别率是800*480, 那么这个新建的窗口的宽就设定为800, 高就设定为480。

边框设定项目

[宽] 外框宽度。此项设定是对窗口设定一个外框的宽度。范围为0~16, 预设值为4。

[颜色] 外框颜色。设定外框的颜色, 可以在颜色列表中选择一个自己需要的颜色, 也可以单击“custom”自己定义一个颜色, 如果外框宽度设定为“0”, 则忽略此项设定。

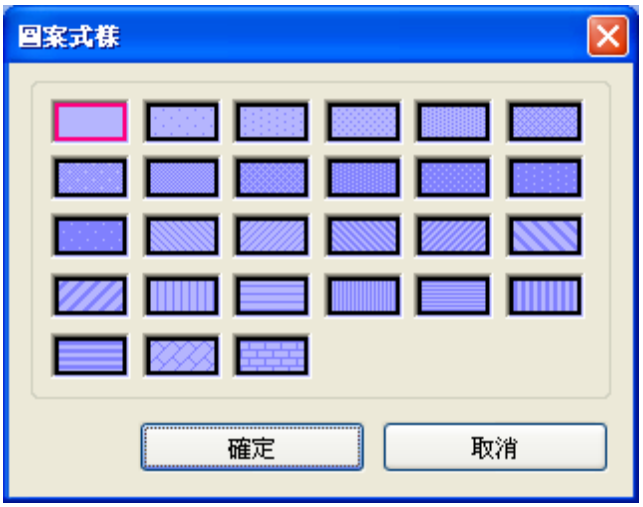
背景设定项目

[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。

[图案]

图案样式, 如果需要, 可以在背景图案列表中选择需要的图案样式作为背景图案。



[图案颜色]

选择背景图案样式的颜色。

[填充]

选择窗口是否填充背景颜色与样式。

重叠窗口设定项目

[底层]、[中层]、[顶层]

每一个基本窗口最多可以再选择其它三个基本窗口作为背景，从[底层]开始到[顶层]结束，这些背景窗口内的元件将在基本窗口中依序出现。系统预设值均为无底层窗口。

弹出窗口设定项目

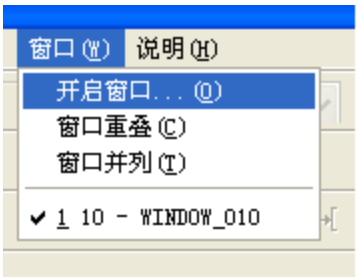
[X]，[Y]

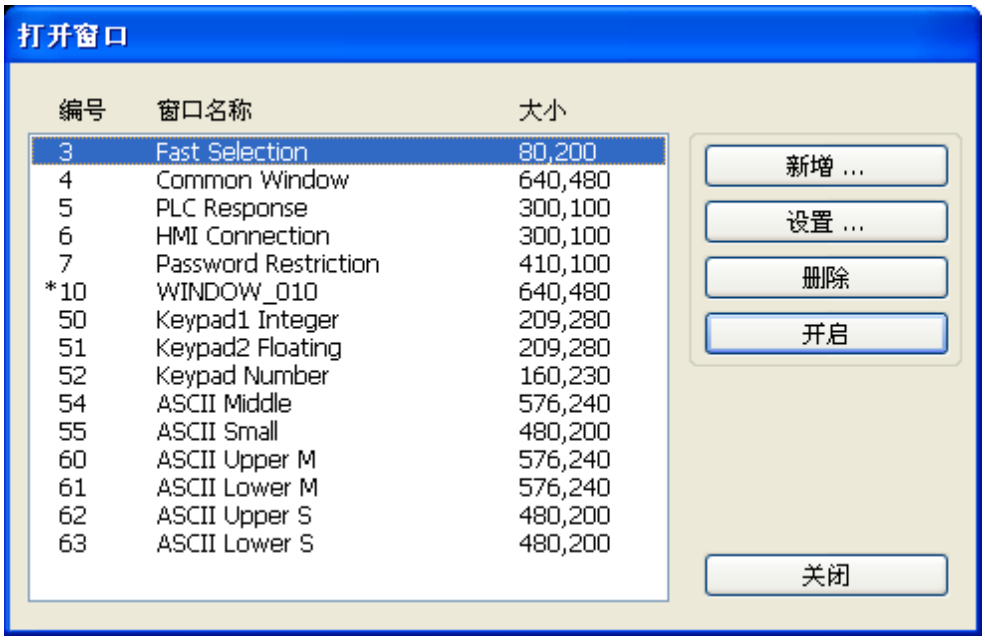
基本窗口也可当作弹出窗口，[X]与[Y]用来设定基本窗口在屏幕弹出的坐标位置。坐标原点在屏幕的左上角。

[垄断]

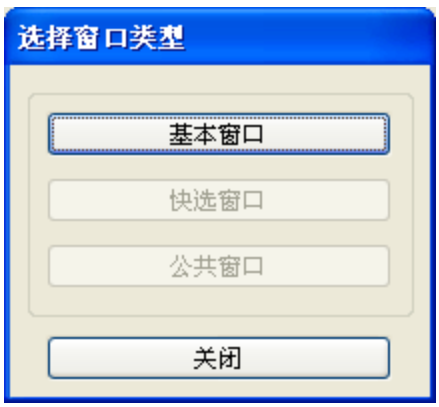
如勾选此选项，当基本窗口作为弹出窗口并出现时，在此基本窗口未关闭前，将无法操作其它窗口。当基本窗口被作为键盘窗口时，自动具有此项属性。

另一种建立的窗口方式是使用“菜单”上的[窗口]，选择[开启窗口]后可以得到“打开窗口”对话窗，参考下图。





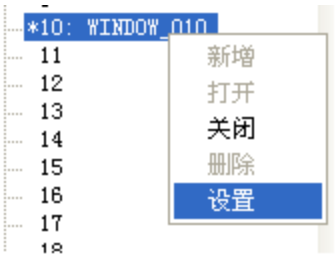
“打开窗口”对话窗的信息会显示窗口的编号、窗口名称。此时触控[新增 ..]按键，利用“选择窗口类型对话框”，选择要建立的窗口类型，在完成各项设定并触控确认键后，即可建立新的窗口。



当基本窗口建立好之后,可以修改窗口编号,也可以修改其尺寸大小、颜色、窗口名称等内容。

(2) 窗口的设定

EB8000提供三种方式更改窗口的属性，第一种方式是在窗口树状图上选择要设定的窗口，并触控鼠标的右键，在窗口出现后选择[设定]即可出现设定对话窗，此时即可更改窗口的属性。



第二种方式是在各窗口中，在未选择到任何元件时触控鼠标的右键，选择[属性]即可出现设定对话窗，此时也可更改窗口的属性。

另一种更改窗口属性的方式是使用“菜单”上的[窗口]，选择[开启窗口]后可以得到“打开窗口”对话窗，在触控[设定…]按键后也会出现设定对话窗，此时即可更改窗口的属性。

(3) 窗口的开启、关闭与删除

要开启已存在的窗口除了可以使用鼠标双击窗口树状图上的页码外，另一种方式是在窗口树状图上选择要开启的窗口，并触控鼠标的右键，在窗口出现后选择[打开]即可开启特定的窗口。

要关闭与删除存在的窗口也是使用相同的操作方式，唯一的限制是要删除的窗口必须为关闭状态。

第七章 事件登录 (event log)

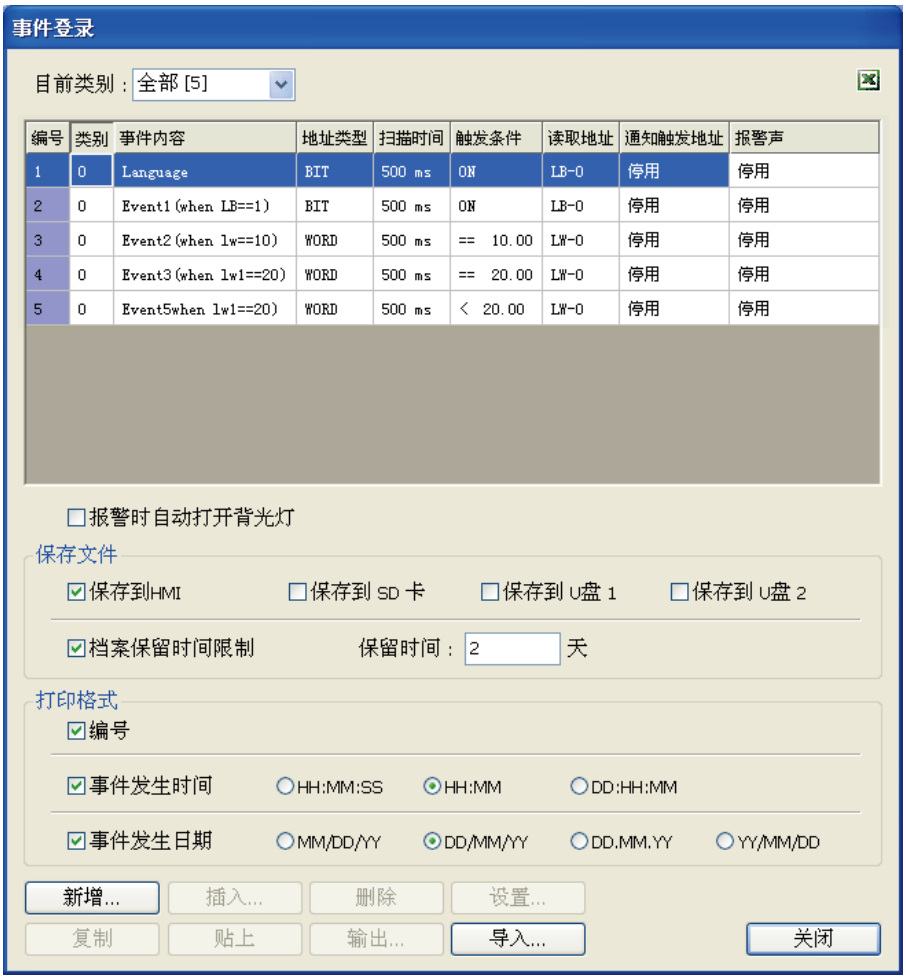
“事件登录” 用来定义事件的内容与触发这些事件的条件, EB8000可以将已被触发的事件(这时事件又被称为报警)与这些事件的处理过程储存到指定的位置, 如触摸屏内部或外接存储设备。所储存文件名称一律使用EL_yyyymmdd.evt格式, 其中yyyymmdd为文件建立的时间, 由系统自行加入。例如事件记录文件名称为EL_20101215.evt, 即表示此文件记录2010年12月15日所发生的事件。

EB8000提供下列系统保留寄存器来管理这些事件记录文件:

[LB 9021]	重置目前的事件记录
[LB 9022]	删除日期最早的事件记录文件
[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

7.1 事件登录管理

搭配[报警条]、[报警显示]与[事件显示]等元件, 用户可以清楚了解事件从发生、等待处理到警报消失的整个周期。要使用这些元件, 首先必须先定义事件的内容, 触控工具条上的 “事件登录按钮” 后会出现 “事件定义管理对话窗” , 参考下图



[目前类别]

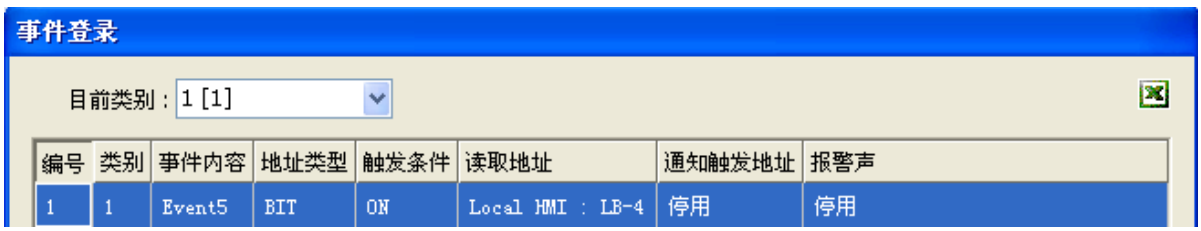
EB8000提供事件分类功能，将事件分成0~255个类别。报警条、报警显示与事件显示元件可以限制所显示的事件类别。

此选项用来选择目前所显示的事件类别。

下图 “0[4]” 的[4]表示类别0目前存在4个事件定义。

编号	类别	事件内容	地址类型	触发条件	读取地址	通知触发地址	报警声
1	0	Event1	BIT	ON	Local HMI : LB-0	停用	停用
2	0	Event2	BIT	ON	Local HMI : LB-1	停用	停用
3	0	Event3	BIT	ON	Local HMI : LB-2	停用	停用
4	0	Event4	BIT	ON	Local HMI : LB-3	停用	停用

下图 “1[1]” 的[1]表示类别1目前只存在1个事件定义。



编号	类别	事件内容	地址类型	触发条件	读取地址	通知触发地址	报警声
1	1	Event5	BIT	ON	Local HMI : LB-4	停用	停用

保存文件项目

用来指定事件记录文件的储存位置，但在PC使用模拟功能时，文件一律储存在与EasyBuilder 8000.exe相同目录下的eventlog子目录中。

[保存到HMI]

将事件记录文件储存在MT8000触摸屏上。

[保存到SD卡]

将事件记录文件储存到SD卡中。

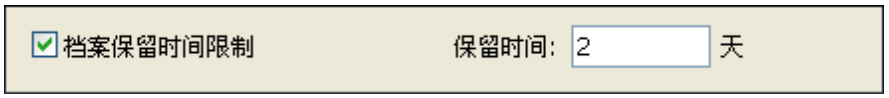
[保存到U盘1]

将事件记录文件储存到U盘1中，U盘的编号规则是：先插入USB插槽的U盘编号为1，依次类推；这与插槽的位置无关。

[保存到U盘2]

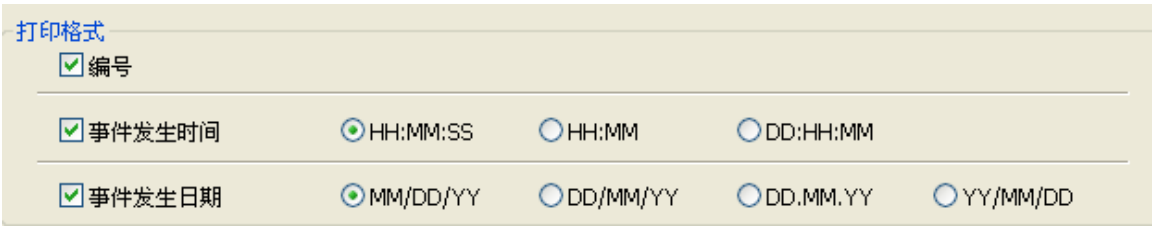
将事件记录文件储存到U盘2中。

当选择保存文件时，可以发现[档案保留时间[限制]此项设定，此项设定值用来决定事件记录文件被保留的时间。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事件记录，超过这个时间范围的文件将自动被删除，以避免储存空间被耗尽。



打印格式项目

使用者需先在系统参数选择打印机类型，才允许设定打印格式。



[新增...]

建立新的事件。

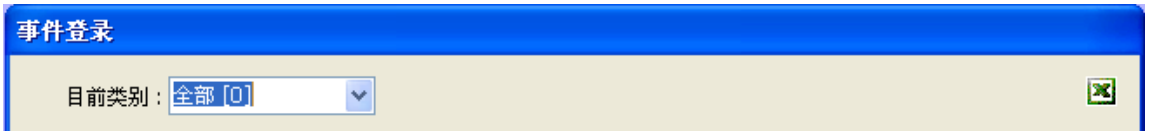
[删除]

删除特定的事件。

[设置...]

修改特定事件的定义。

7.2 Excel编辑



在右上角有一个Excel的图示，可以使用此功能来编辑“事件登录”。

一个完整的设计步骤包括：Excel编辑、导入到Event log、输出到Excel，下面介绍各项步骤的具体内容：

1) Excel 编辑：

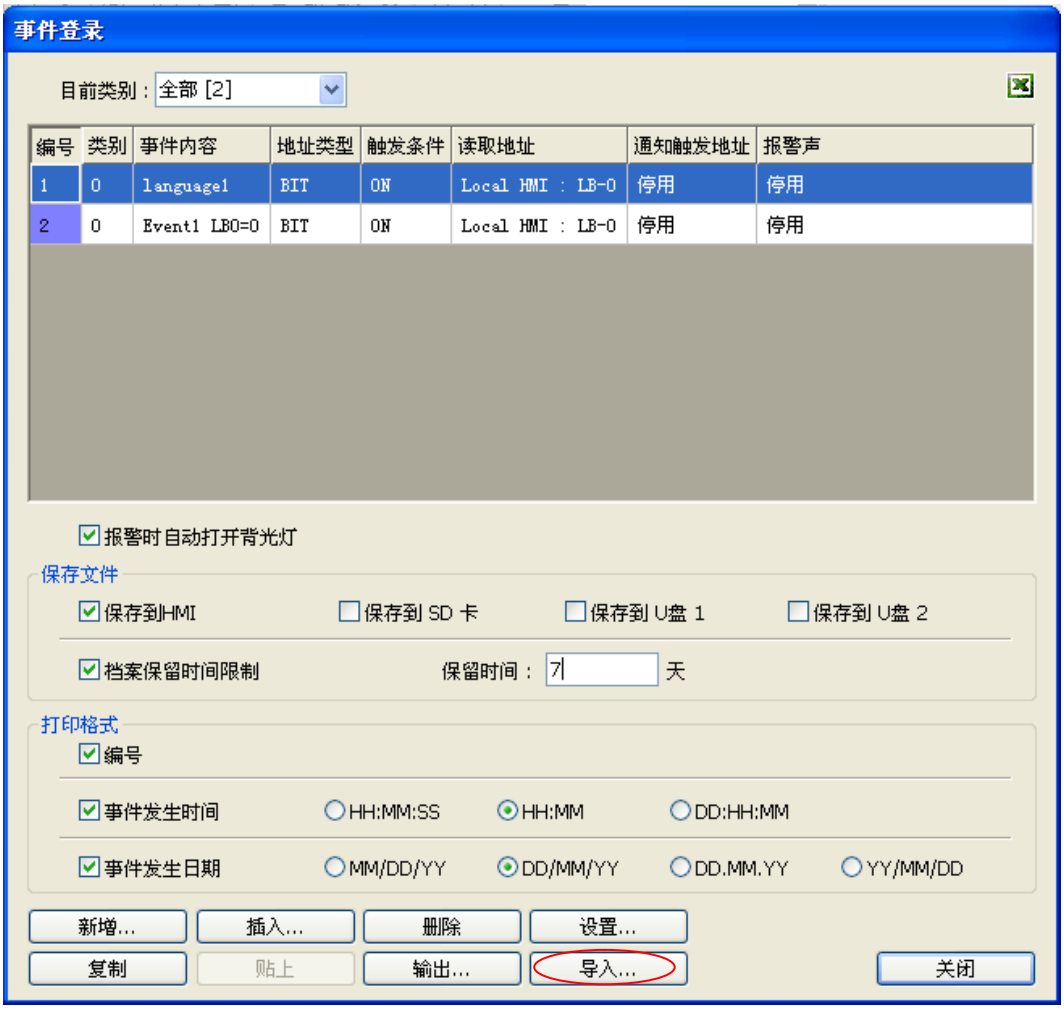
建议用户使用由我们所提供在C:\EB8000\EventLogExample.xls的Excel范例文件做编辑，范例文件中有设计好的验证机制和下拉式选单方便用户使用。

	A	B	C	D	E	F	G	H	I	J	K
1	Category	Priority level	Address type	PLC name	Device type	System tag	User-defined tag	Address	Index	Data Format	Enab
2	0	Middle	Word	Local HMI	EM0	False	False	22	null	32-bit Signed	True
3	1	Low	Bit	Local HMI	LB-9009 : initialized as ON	True	True	122	IDX 1	16-bit BCD	False
4	2	High	Word	Local HMI	RW1	False	False	2222	IDX 4	32-bit BCD	
5										16-bit BCD	
6										32-bit BCD	
7										16-bit Unsigned	
										16-bit Singed	
										32-bit Unsigned	
										32-bit Signed	

以下几点请用户注意：

- [System tag]与[User-defined tag]请勿同时设定为true，若同时设定为true，系统会将其视为System tag，也就是将User-defined tag视为false。若使用在Device type栏所输入的资料是User-defined tag，请将System tag设为false。
- Color所接受的格式为R:G:B，其中R、G、B为介于0~255之间的整数。
- 点击Event log右上角Excel小图标即可打开Excel范例文件。

2) 导入到Event log：点击[导入]按钮导入excel文件：

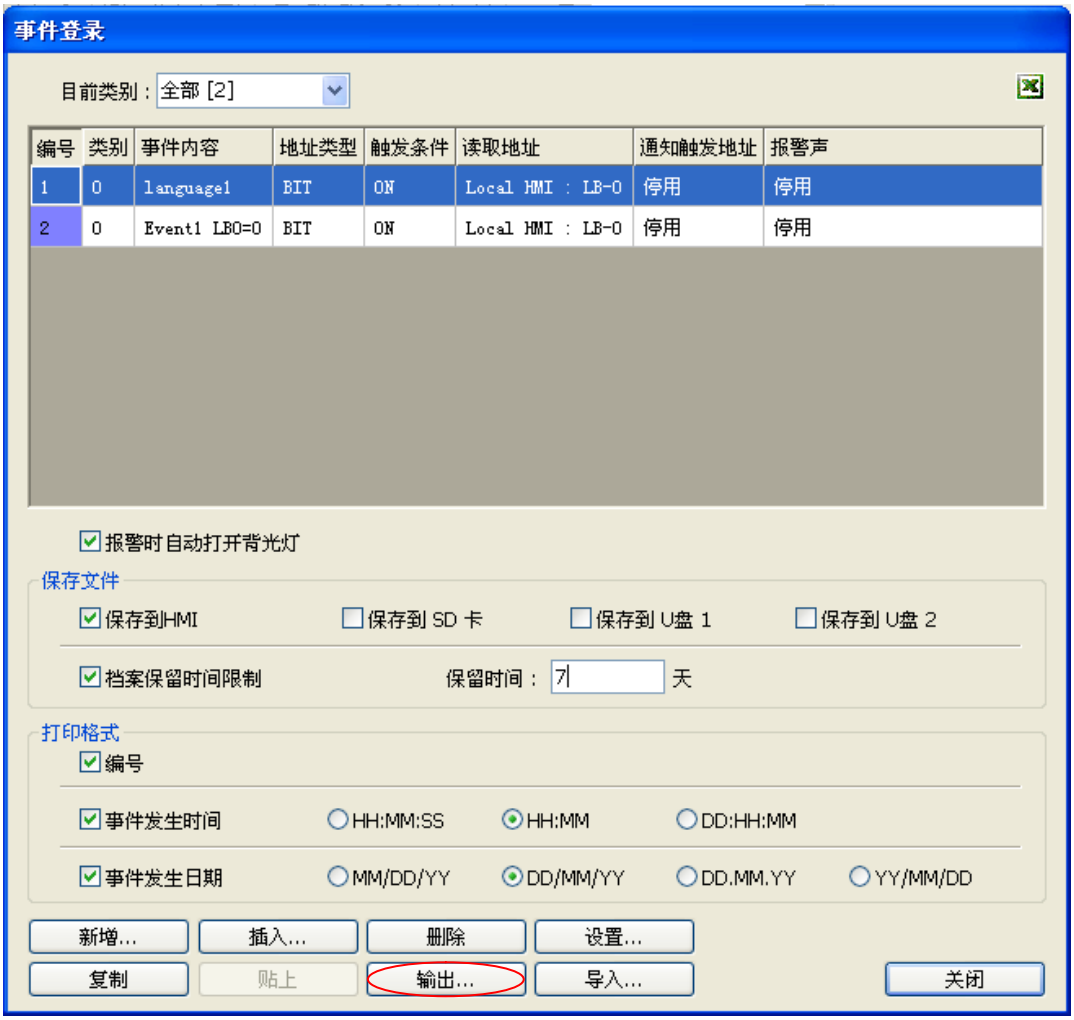


以下几点请用户注意：

- 若用户在Excel表中将user-defined tag设定为true，但若系统将device type与系统中用户自定的tag做比对，却找不到合适的tag，则系统会自动将event log中的user-defined tag设定为false。
- 在导入library（label library和sound library）之前，请先确认系统中已存在对应名称，否则在系统中找不到对应名称的情况下，系统将会使用excel文件上的名称。

3) 输出Event log

点击[输出]按钮将event log资料输出到excel文件：



7.3 建立一个新的事件记录

在触控[新增…]后，将出现“报警（事件）登录对话窗”，包含两个分页，下图为[一般属性]设定页。

[类别]

事件的类别。

[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笔，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]

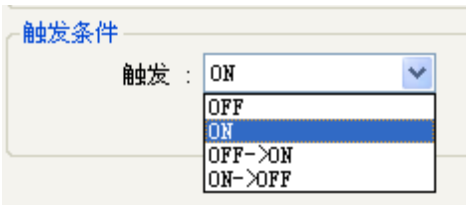
系统利用读取此地址所获得的数据，来检查事件是否满足触发条件。其余设定请参考“元件一般属性设定”。

[通知触发地址]

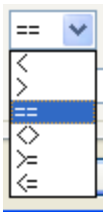
当事件被触发时，将利用此项地址送出特定的信号。选择[开]将对此特定地址送出ON信号；选择[关]则对此特定地址送出OFF信号。其余设定请参考“元件一般属性设定” 。

[触发条件]

当事件的[地址类型]选择“Bit”时，触发条件可选择“开”(ON)或“关”(OFF)，参考下图。 此时选择的触发条件为“开”，也就是当[读取地址]的状态由OFF变为 ON时，事件将被触发，并产生一个事件记录(或称报警记录)。

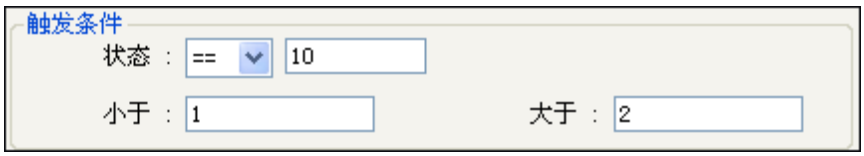


当事件的[地址类型]触发条件选择“Word”时，触发条件可选择项目如下。



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：

设定范例1:



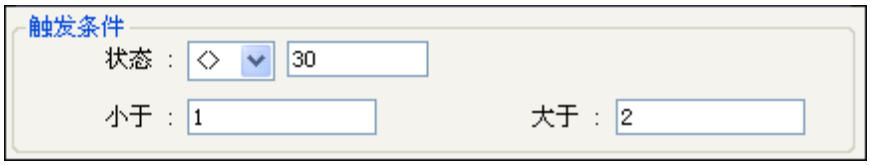
上面的设定内容表示当[读取地址]中的数据大于等于9(= 10 –1)且小于等于11(=10 + 1)时, 事件将被触发。也就是事件被触发的的条件为

9 <= [读取地址]中的数据 <= 11

事件被触发后, 当 [读取地址]中的数据大于12(=10 +2)或小于8(= 10 –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 < 8或[读取地址]中的数据 > 12

设定范例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。

文字项目

[内容]

事件记录在[报警条]、[报警显示]与[事件显示]元件中显示的内容。其余设定请参考“元件一般属性设定”。

可以在显示内容中包含事件被触发当时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、d作为结束符号。

例如显示内容被设定为“High Temperature = %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13，则在“事件显示”元件中的显示内容将为“High Temperature = 13”。

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

录的[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、d作为结束符号。

例如显示内容被设定为“High Temperature = $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”元件中的显示内容将为“High Temperature = 42”。

[字体][颜色]

每一个事件可以单独设定字体与颜色，在[报警显示]与[事件显示]元件中所显示事件的字体与颜色来自这些设定值。以下图为例，可以看出两个事件使用不同的字体与颜色。

[事件确认时写入]

事件在事件显示元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“元件”章节中有关“事件显示元件”的说明。

[报警声]

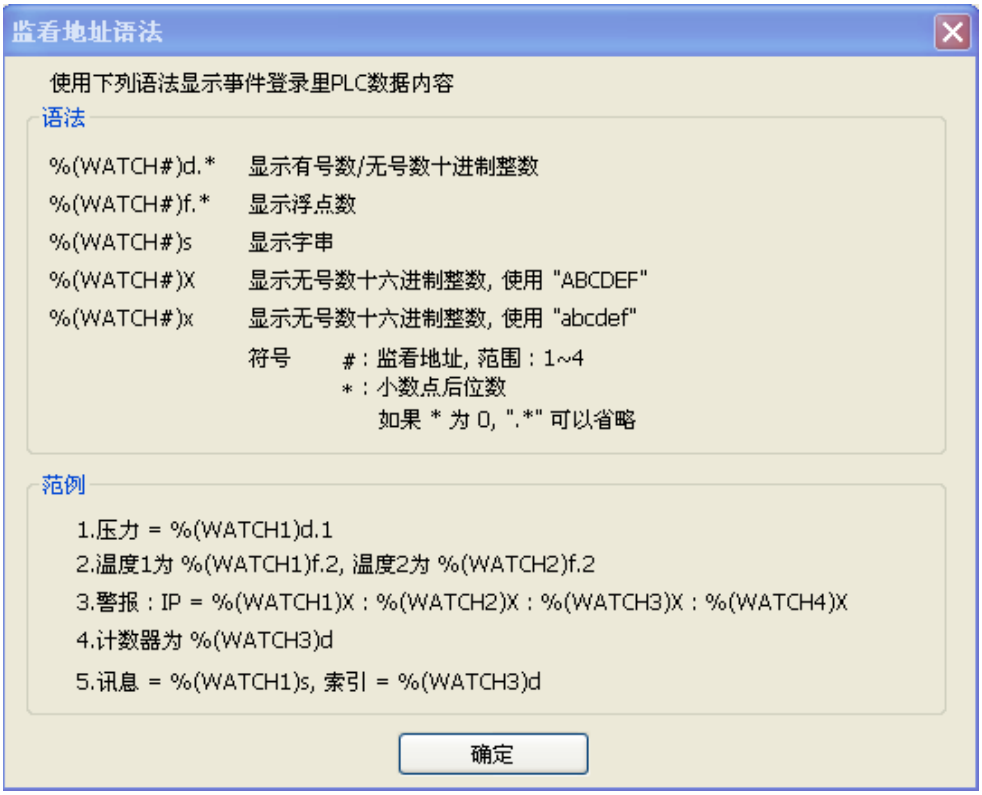
事件被触发时，可选择使用音效警示。可在声音库选择警示的声音，并使用[播放]按钮确认选择的声音。

完成上述的各项设定后，即可增加一笔新的事件定义，参考下图。

编号	类别	事件内容	地址类型	触发条件	读取地址	通知触发地址	报警声
1	0	Event1	BIT	ON	Local HMI : LB-0	停用	停用
2	0	Event2	BIT	ON	Local HMI : LB-1	停用	停用
3	0	Event3	BIT	ON	Local HMI : LB-2	停用	停用
4	0	Event4	BIT	ON	Local HMI : LB-3	停用	停用
5	1	Event5	BIT	ON	Local HMI : LB-4	停用	停用

[监看地址]

用户可以按照[语法]显示事件登录里PLC数据内容，关于语法的使用，请参考下面的对话框：


监看地址语法
使用下列语法显示事件登录里PLC数据内容
语法
%(WATCH#)d.* 显示有号数/无号数十进制整数
%(WATCH#)f.* 显示浮点数
%(WATCH#)s 显示字串
%(WATCH#)X 显示无号数十六进制整数, 使用 "ABCDEF"
%(WATCH#)x 显示无号数十六进制整数, 使用 "abcdef"
符号 #：监看地址, 范围：1~4
*：小数点后位数
如果 * 为 0, ".*" 可以省略
范例
1.压力 = %(WATCH1)d.1
2.温度1为 %(WATCH1)f.2, 温度2为 %(WATCH2)f.2
3.警报：IP = %(WATCH1)X：%(WATCH2)X：%(WATCH3)X：%(WATCH4)X
4.计数器为 %(WATCH3)d
5.讯息 = %(WATCH1)s, 索引 = %(WATCH3)d
确定

第八章 资料取样 (Data sampling)

“资料取样”用来定义资料被取样的方式，包括取样周期与取样位置，EB8000可以将已获得的取样资料储存到指定的位置。

储存的方式为：[保存位置]\[资料取样名称]\yyyymmdd.dtl

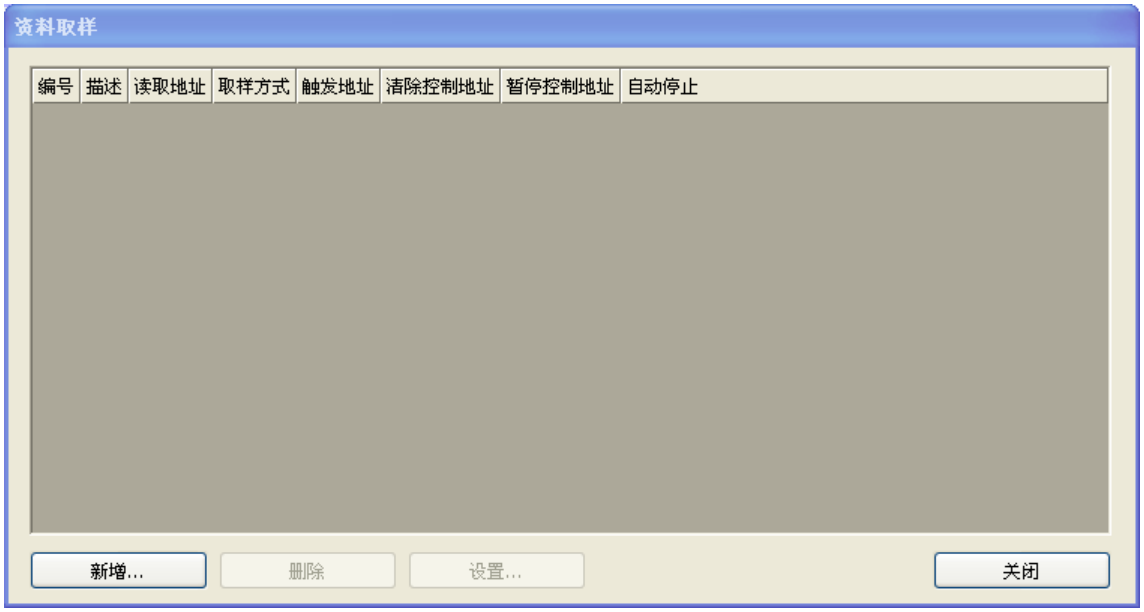
其中保存位置可能为HMI、CF（SD）卡、U盘1或U盘2，用户可自由指定；资料取样名称通常是用户为此资料取样所取的一个有意义的名字，并且不能重复；yyyymmdd为文件建立的日期，由系统自动加上。

EB8000并提供下列系统保留寄存器来管理这些事件记录文件：

[LB 9025]	删除日期最早的资料取样文件
[LB 9026]	删除全部的资料取样文件
[LB 9027]	更新资料取样统计信息
[LW 9063]	目前存在的资料取样文件的数目
[LW 9064]	全部资料取样文件的大小

8.1 资料取样记录管理

使用“趋势图”元件及“历史数据显示”元件可以检视资料取样记录的内容，但首先必须定义资料取样的方式，触控工具条上的工作按钮后会出现“资料取样管理对话框”，参考下图。



[新增...]

增加一个新的“资料取样”定义。

[删除]

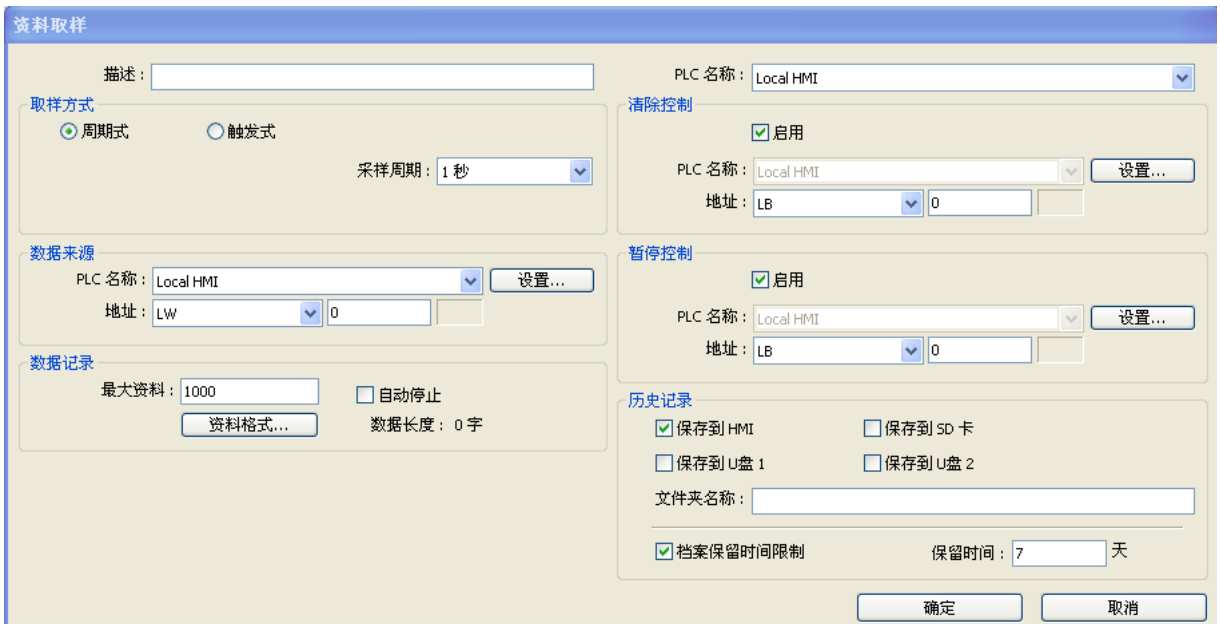
删除特定的“资料取样”。

[设置...]

修改特定的“资料取样”定义。

8.2 建立一个新的资料取样

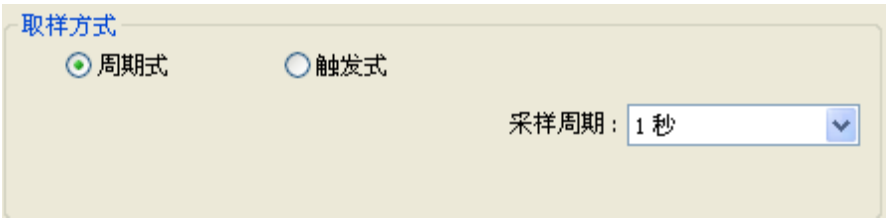
触控[新增…]后会出现“资料取样”设定页，参考下图。



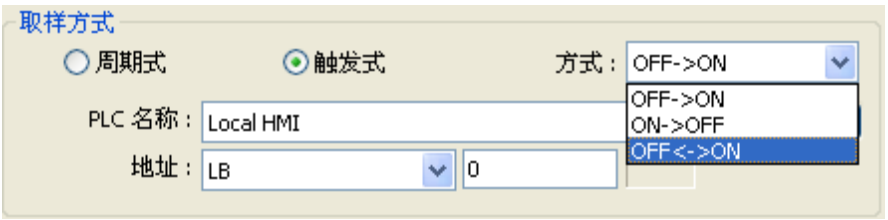
[取样方式]

EB8000提供两种资料取样方式： [周期式]和[触发式]。

选择[周期式]模式时，EB8000将使用固定的时间频率进行资料取样动作，用户必须设定[采样周期]的时间间隔。



选择[触发式]模式时，用户则可以利用一个特定的位地址的状态，来触发取样动作。



设定触发资料取样的条件：

[OFF → ON]若指定位地址的状态从OFF变为ON，资料取样的动作会被触发。

[ON → OFF]若指定的位地址的状态从ON变为OFF，资料取样的动作会被触发。

[ON←→ OFF] 只要指定位地址的状态发生转变，资料取样的动作就会被触发。

[数据来源]

选择一个设备地址作为取样数据的来源。

[数据记录]

[最大资料]

一个资料取样项目一天最多可以记录的取样资料笔数，最多为86400笔。

1、如果EB8000在[趋势图]资料来源为[即时模式]的话，将删除较旧的取样资料，并加入刚刚获得的资料并显示在[趋势图]上。

2、如果是[历史模式]，则资料取样还是继续被取样。

[自动停止]

当已经获得的取样资料数据等于[最大资料]设定值是，若勾选了此项设定，将会停止取样动作。

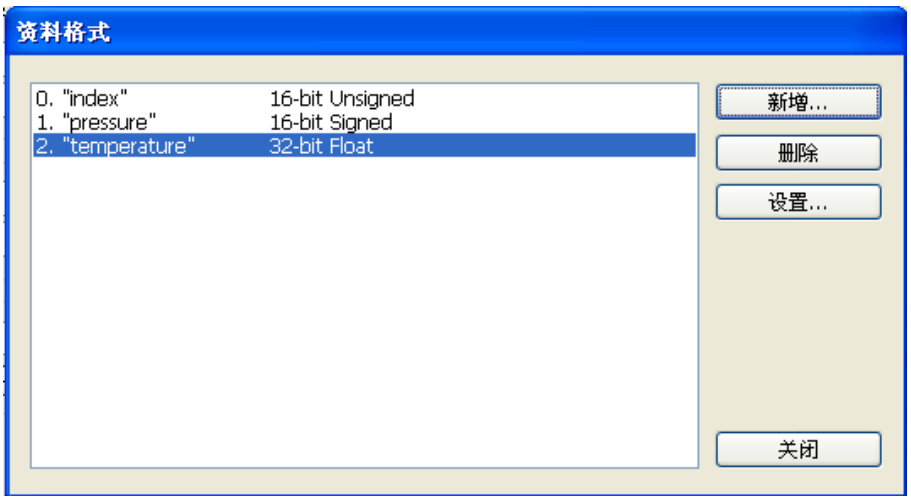
例如：

[最大资料]设定值为10，未勾选[自动停止]选项：[趋势图]的即时模式会显示最新的10笔资料；资料取样则会删除旧资料并采集新资料。

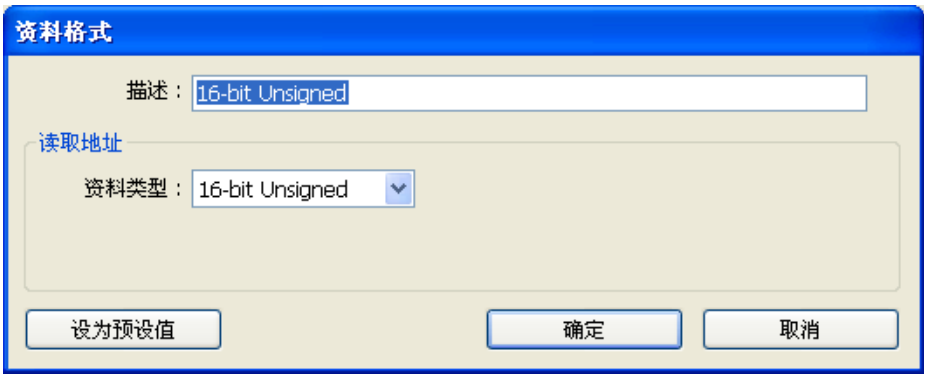
[最大资料]设定值为10，勾选[自动停止]选项：[趋势图]的即时模式会显示至第10笔资料后停止显示；资料取样到第10笔资料后停止记录。

[资料格式...]

一笔取样资料所包含的数据型态。一笔取样资料可能包含超过一项以上的数据，EB8000提供的资料取样动作可以同时撷取不同型态的数据。触控[资料格式]后，用户可以利用“数据型态对话窗”自行定义一笔取样资料的内容。以下图的例子来说，使用者共定义了三个数据，分别为“index”(16-bit Unsigned)、“pressure”(16-bit Signed)与“temperature”(32-bit Float)，长度为总共为4 words。 也就是说每次的取样动作，EB8000会自指定的地址每次撷取长度为4 words的数据，作为一笔取样资料的内容。



读取地址项目中的其它设定项目请参考“元件一般属性的设定”。



注意：在执行离线模拟后,不可再更改资料取样,如果需要改变,请将C:\EB8000\HMI_memory\datalog内记录历史数据显示的文件删除,再进行离线模拟的动作。

[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，但不影响已存入文件中的取样资料，此项功能只适用于[趋势图]的“即时模式”。其它设定项目请参考“元件一般属性的设定”。

[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“元件一般属性的设定”。

[历史记录]

用来指定取样资料的储存位置，但在PC使用模拟功能时，文件一律储存在与EasyBuilder 8000.exe相同目录下的datalog子目录。

[保存到HMI]：将取样资料储存在MT8000上。

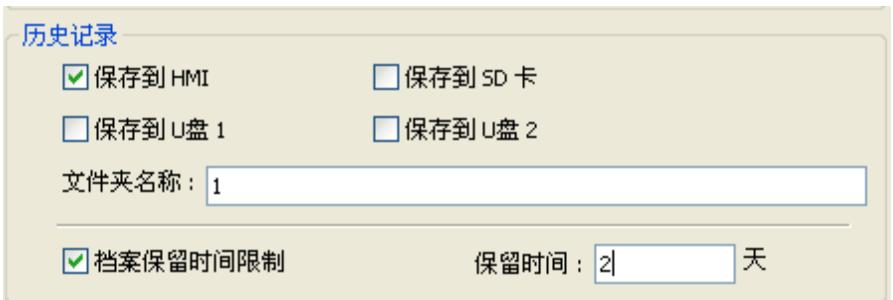
[保存到SD卡]：将取样资料储存到SD卡中。

[保存到U盘1]：将取样资料储存到U盘1中，U盘的编号规则是：先插入USB插槽的U盘编号为1，后续为2，最后一个为3，与插槽的位置无关。

[保存到U盘2]：将取样资料储存到U盘2中。

[文件夹名称]：设定资料取样名称。

当选择保存文件时，可以发现[档案保留时间限制]此项设定，此项设定值用来决定资料取样记录文件被保留的时间。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文件，超过这个时间范围的文件将自动被删除，以避免储存空间被耗尽。例如：今天是7月1号，HMI只会保存6月30日和6月29日2天的资料，6月28日的资料则会被自动删除。



注意:资料必须要到达4kb,才能被储存,如果少于4kb,可以使用LB-9034来强迫储存资料。

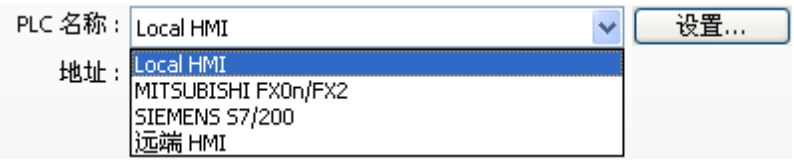
第九章　元件一般属性

元件(object)“一般属性设定”的内容包含下面项目：

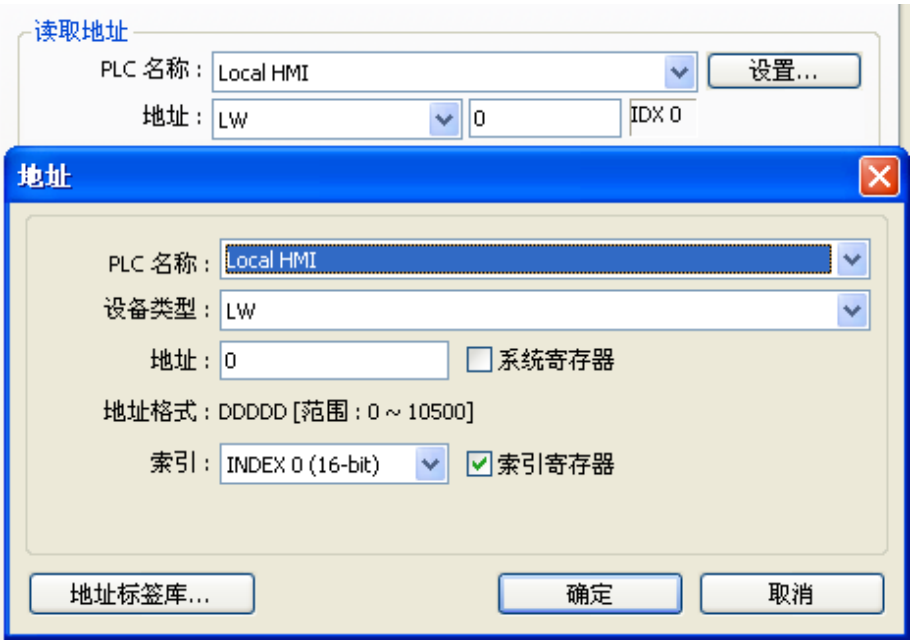
1) 选择PLC设备
2) 读写地址(reading and writing address)设定
3) 向量图库(shape library)与图形库(picture library)的使用
4) 标签内容设定(text)
5) 轮廓调整(profile)

9.1 选择PLC

某些元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“MITSUBISHI FX0n/FX2”，这些PLC名称来自“系统参数”(system parameters)中“设备列表”(device list)的内容。



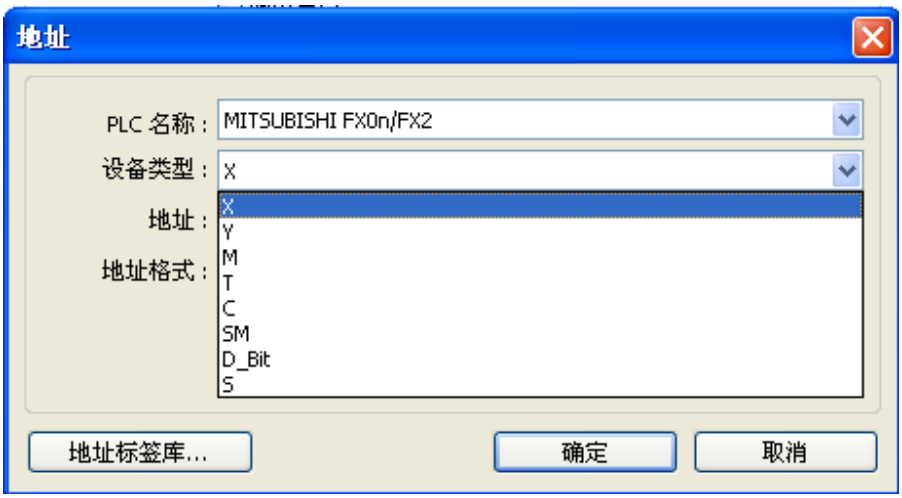
9.2 读写地址设定



上图可以看出一般地址的设定包含下列项目：

[设备类型]

选择地址类型，当PLC不同时，将出现不同的地址类型。

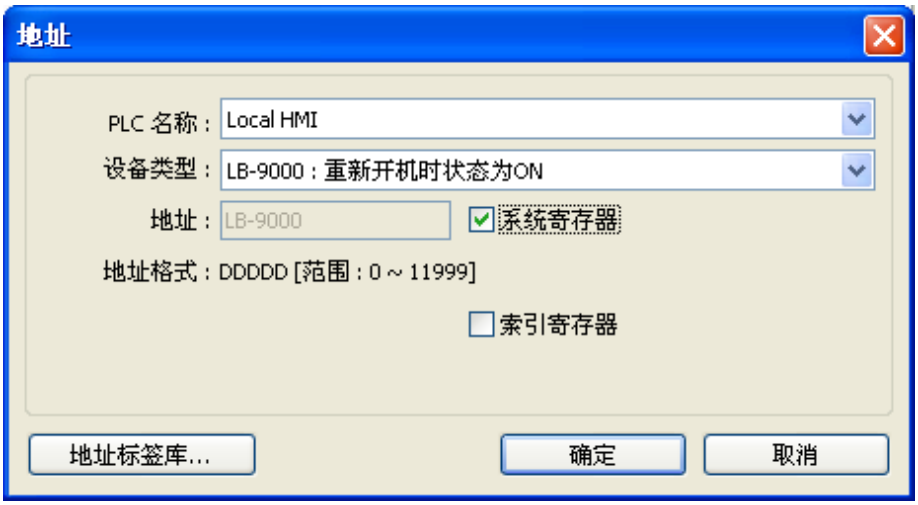


[地址]

设定读写的地址。

[系统寄存器]

地址标签库包含“系统寄存器”(system tag)与“使用者定义”(user-defined tag)。此项目用来选择是否使用“地址标签库”。系统寄存器为系统保留作为特殊用途的地址，分为bit地址系统寄存器与word地址系统寄存器，在选择使用“系统寄存器”后，除了[设备类型]将显示系统寄存器的内容之外，[地址]将显示目前所选用的系统系统寄存器，如下图所示。



下图为bit地址系统寄存器与word地址系统寄存器的部分内容，其它内容可参考“地址标签库”(tag library)的使用说明。

bit地址系统保留寄存器

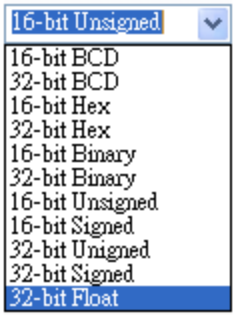
word地址系统保留寄存器

[索引寄存器]

选择是否使用 “索引寄存器” (index register), 可参考《第十一章 索引寄存器》的说明。

资料格式选择

EB8000支持下列的资料格式, 需正确选择资料格式, 尤其是在使用地址标签(address tag)时。



9.3 向量图库(shape library)与图片库(picture library)的使用

某些元件可以使用向量图库与图形库的图形, 增加元件的视觉效果。向量图库与图形库的使用在元件属性页中的[图片]分页中设定, 见下图。

[图片]设定页各项设定的说明如下:

向量图库设定项

[图库…]: 选择样式, 此项目请参考后面的说明。

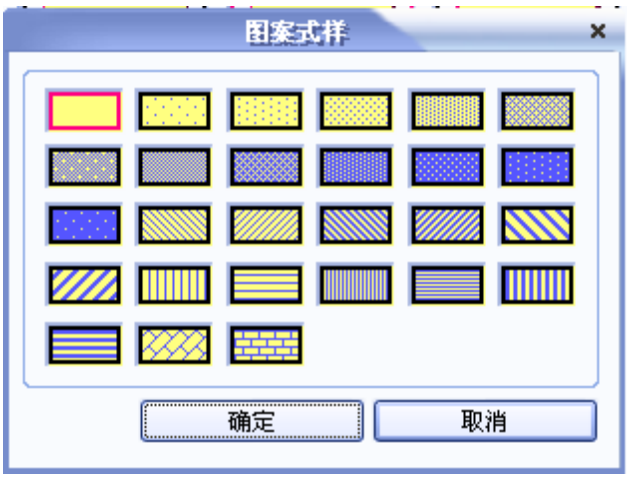
[使用向量图]: 选择图案是否使用向量图库的图形。

[内部]: 选择是否使用图案的内底, 触控颜色设定钮后所出现的 “色彩对话窗”, 可用来设定内底的颜色, 如下图所示。用户也可以设定[自定义色彩], EB8000会记住用户设定的[自定义色彩]。

[边框]: 选择是否使用图案的外框，触控颜色设定钮后所出现的“色彩对话窗”，可用来设定外框的颜色。

[内部图案]: 用来设定内底填充图案的颜色。

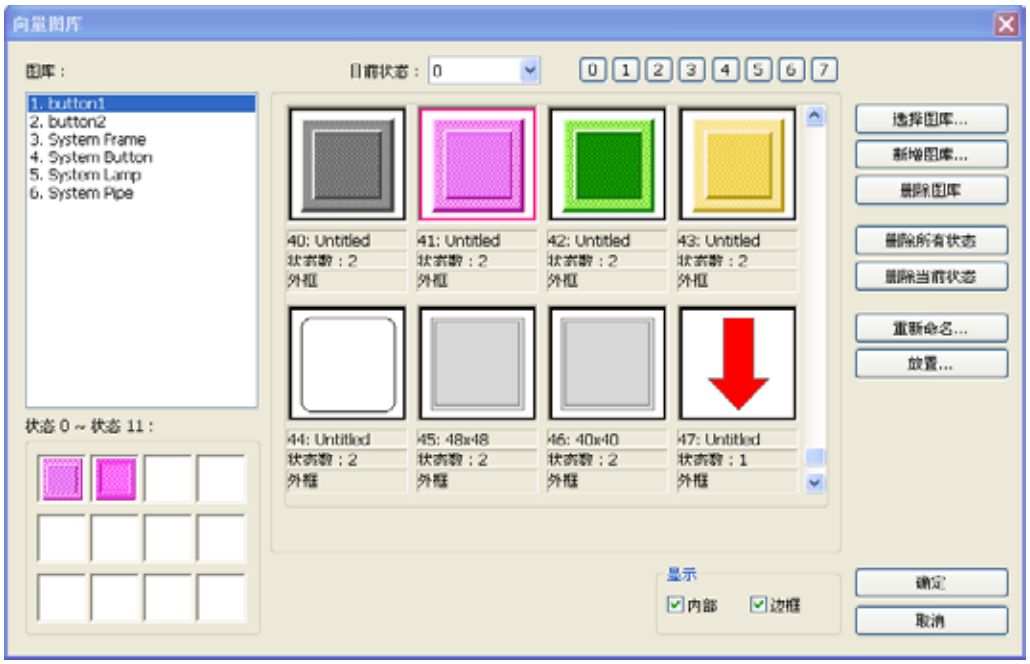
[图案式样]: 触控设定钮后将出现下图所示的对话窗，可用来选择填充样式。



[把设定复制到所有状态]: 将目前状态各项属性设置到其它状态。

如何使用向量图库:

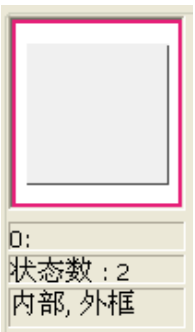
在触控[图库…]按钮后可以得到下面的“向量图库对话窗”，由对话窗中可看出目前选择的样式会使用红色的外框加以标示。



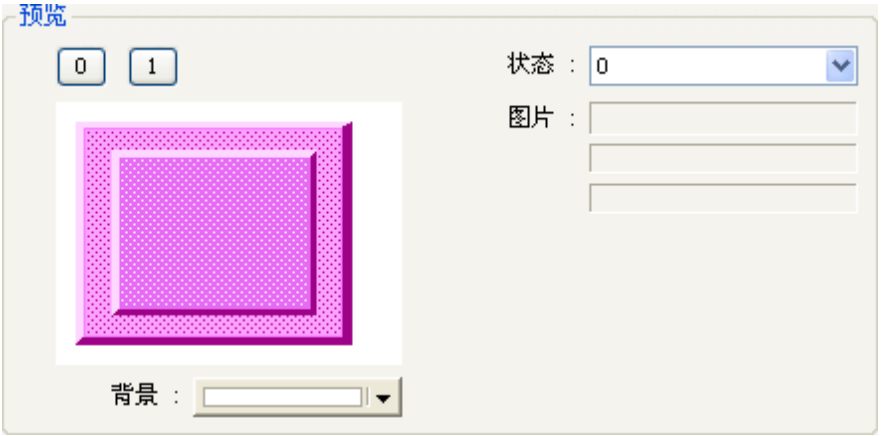
上图显示向量图库中某一样式的信息，这些信息的意义如下：

41: Untitled	表示此向量图的名称与向量图库编号
状态数 :2	此向量图的状态个数
外框	表示此向量图只具备外框

下图则显示此向量图具备外框与内底。



“向量图管理对话窗”各项目的说明可参考《第十四章 向量图库、图形库的建立与使用》。在完成各项设定并触控确认键后，元件将使用目前所选择的样式，如下图。



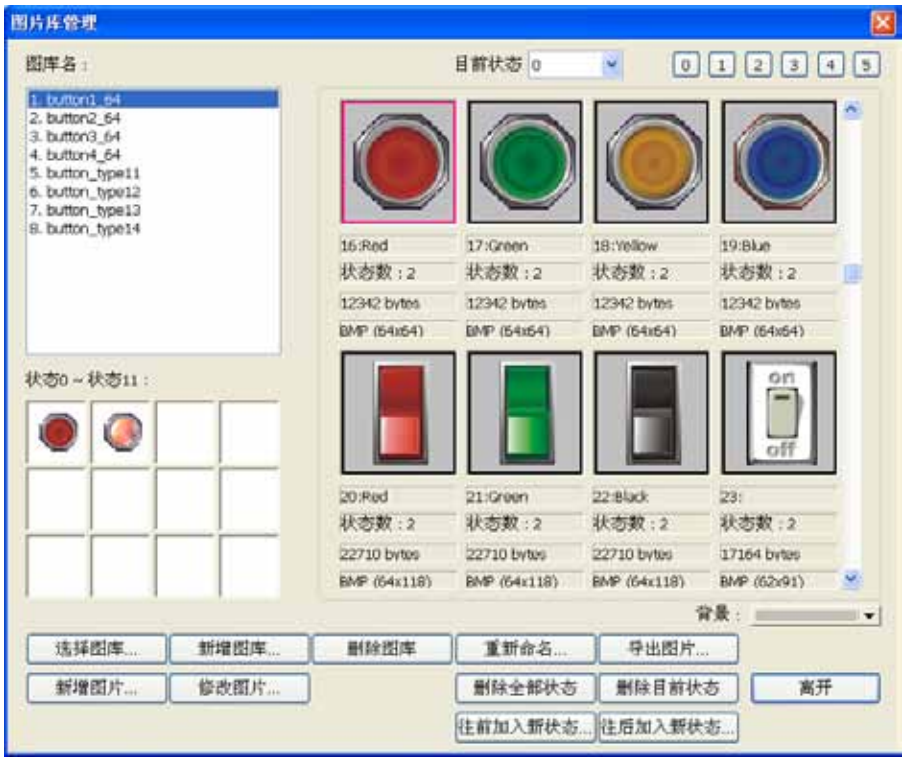
图片库设定项

[图库]：选择图片，此项目请参考后面的说明。

[使用图片]：选择是否使用图片库的图形。

如何使用图形库：

在触控[图库…]按钮后可以得到下面的“图片库对话窗”，由对话窗中可看出目前选择的图形会使用红色的外框加以标示



上图显示图形库中某一图形的信息，这些信息的意义如下：

16: Red	图形编号与的名称
状态数 : 2	图形的状态个数
12342 bytes	图形的大小
BMP (64×64)	图形的格式与原尺寸，BMP表示图形使用bitmap格式，图形格式也可以为JPG、GIF、PNG。64×64表示图形的原尺寸长为64 pixels，高为64 pixels。

“图形库对话窗”各项目的说明参考“向量图、图形库的建立与使用”。在完成各项设定并触控确认键后，元件将使用目前所选择的图形，如下图所示。

9.4 文字内容设定

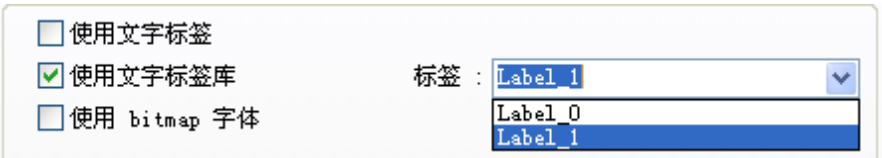
元件内文字的使用在元件属性页中的[标签]分页中设定，如下图。

[使用文字标签]

勾选此选项元件才允许使用文字标签

[使用文字标签库]

勾选此选项表示文字内容将来自文字标签库，如下图。

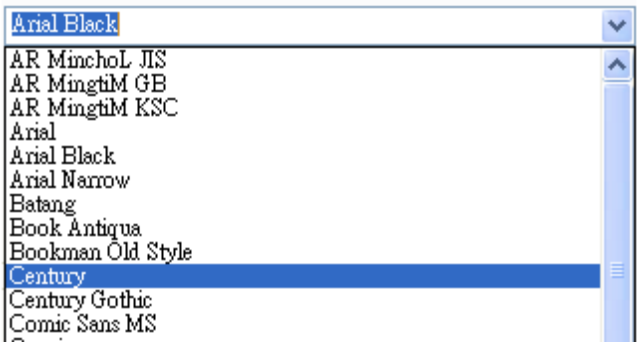


[文字标签库...]

检视标签库的内容, 可参考《第十五章 文字标签库与多国语言使用》。

[字体]

选择文字所使用的字型。EB8000支持WINDOWS的true-font字型, 如下图。

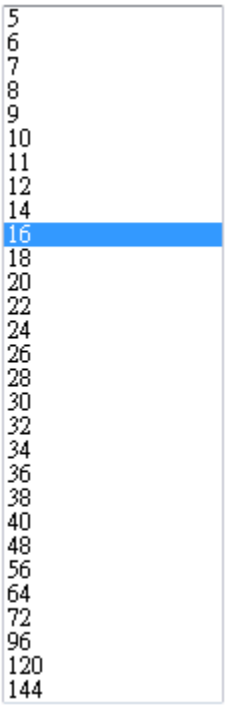


[颜色]

选择文字所使用的颜色。

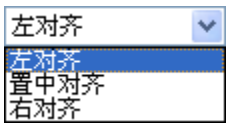
[尺寸]

选择文字所使用的大小。EB8000支持下图显示的字号。



[对齐]

选择多行文字的对齐方式，可选择的方式如下：



下图为选择“左对齐”的对齐方式。

111
222222
333333333

下图为选择“置中对齐”的对齐方式。

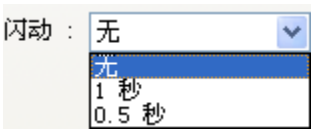
111
222222
333333333

下图为选择“右对齐”的对齐方式。

111
222222
333333333

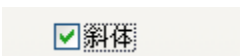
[闪动]

选择文字闪烁方式，可选择不闪烁“无”，或闪烁时间间隔为“1 秒”或“0.5秒”的闪烁方式。



[斜体]

使用斜体字体。



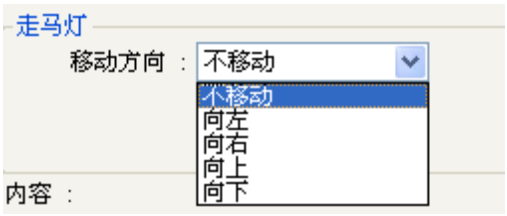
[底线]

文字加上底线。

动作设定项

[走马灯]

设定走马灯的效果并选择文字的移动方向，有下列的选择：



[持续移动]

当文字选择走马灯的效果时，下图的文字具有两种显示方式：

未勾选此选项，则文字需在全部消失后才出现后续的文字，如下图。

有勾选此选项，则文字会连续出现，如下图。

[速度]

选择文字的移动速度。

[内容]

文字内容。如使用文字标签库，此项内容将来自文字标签库。

[编辑时文字位置连动]

勾选此选项时，移动某个状态的文字将连带移动其它状态的文字。

[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。

9.5 轮廓调整

如下图，元件的外型大小可以利用[轮廓]设定页加以调整。

位置设定项目

[图钉]：锁定设定，勾选此选项后将无法改变元件的位置与大小。

[X]、[Y]元件左上角的坐标。

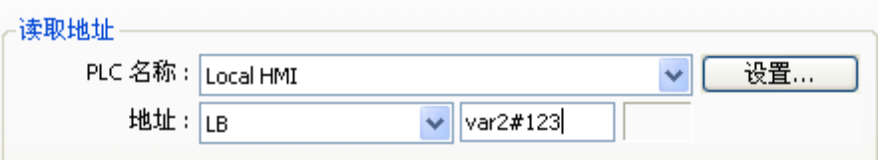
尺寸设定项目

[宽度]：元件的宽度。

[高度]：元件的高度。

9.6 站号变量的使用

EB8000 V1.31版或更新的版本，在PLC的地址设定中允许使用站号变量，请参考下图，其中“var2”为16个站号变量中的一个



站号变量的使用语法如下：

varN#address

其中N的范围为0～15的整数，address为PLC的地址

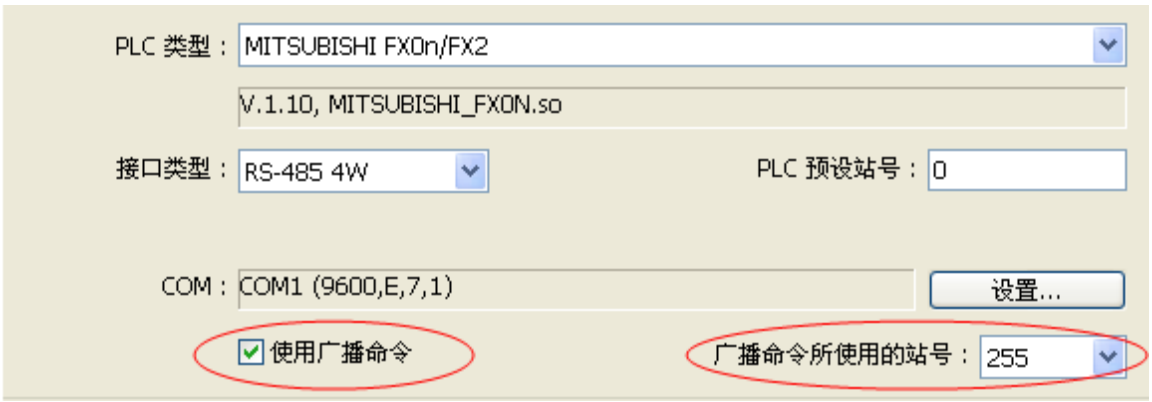
MT6000/8000目前提供16个站号变量：var0～var15，这些站号变量的实际数据读取自LW10000～LW10015。下面为站号变量所对应的系统寄存器地址：

var0	LW10000
var1	LW10001
var2	LW10002
var3	LW10003
var4	LW10004
var5	LW10005
var6	LW10006
var7	LW10007
var8	LW10008
var9	LW10009
var10	LW10010
var11	LW10011
var12	LW10012
var13	LW10013
var14	LW10014
var15	LW10015

例如：“var0”的数据读取自LW10000，所以当LW10000中的数值等于32时，表示使用var0#234等同于使用32#234，也就是此时的站号为32；同样的，“var13”的数据读取自LW10013，所以当LW10013中的数值等于5时，表示使用var13#234等同于使用5#234。

9.7 广播站号的使用

MT6000/8000提供两种方式让客户开启广播站号的使用，第一种方式是直接在系统参数中直接设定PLC的属性，如下图所示：



第二种方式是使用系统寄存器开启/关闭广播站号功能，并更改广播站号。相关的系统寄存器地址内容如下：

LB9065	停用/开启COM 1广播站号
LB9066	停用/开启COM 2广播站号
LB9067	停用/开启COM 3广播站号

LW9565	COM 1广播站号
LW9566	COM 2广播站号
LW9567	COM 3广播站号

第十章 元件的安全防护

EB8000的元件安全防护分为：

1. 用户密码与可操作元件类别设定
2. 元件操作安全防护

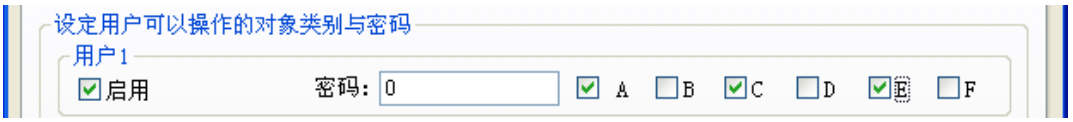
10.1 用户密码与可操作元件类别设定

系统参数中的[用户密码]设定页用来设定用户的密码，并规划每个用户可操作的元件类别。

在EB8000中，元件被划分为“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，EB8000最多可规划12个用户。

触摸屏运行时，用户在成功输入密码后，EB8000会依照用户的设定内容决定用户可以操作的元件类别。例如，当“用户1”的规划内容如下时，此位用户只被允许操作元件类别属于“无”与A、C、E的元件。



正确的密码输入过程：

1、使用[LW9219] (共1个words, 16-bit)指定目前的用户；

2、将密码输入到密码输入地址[LW9220](共2个words, 32-bit)。

注意：[LW9219]中的数据必须为1 ~ 12，分别用来表示“用户1”至“用户12”。

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保留寄存器[LW9500]至[LW 9522], 共24 words的内容获得。

EB8000也提供用户在线更改密码

当系统保留寄存器[LB9061]的状态由OFF转变为ON时, EB8000会使用系统保留寄存器[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: 此时用户可使用的元件类别并不会因密码的变更而改变。

当[LB9050] (用户登出 – user logout)的状态由ON变为OFF时, 可强迫目前的用户注销系统, 此时系统将只允许类别属于 “无” 的元件被操作。

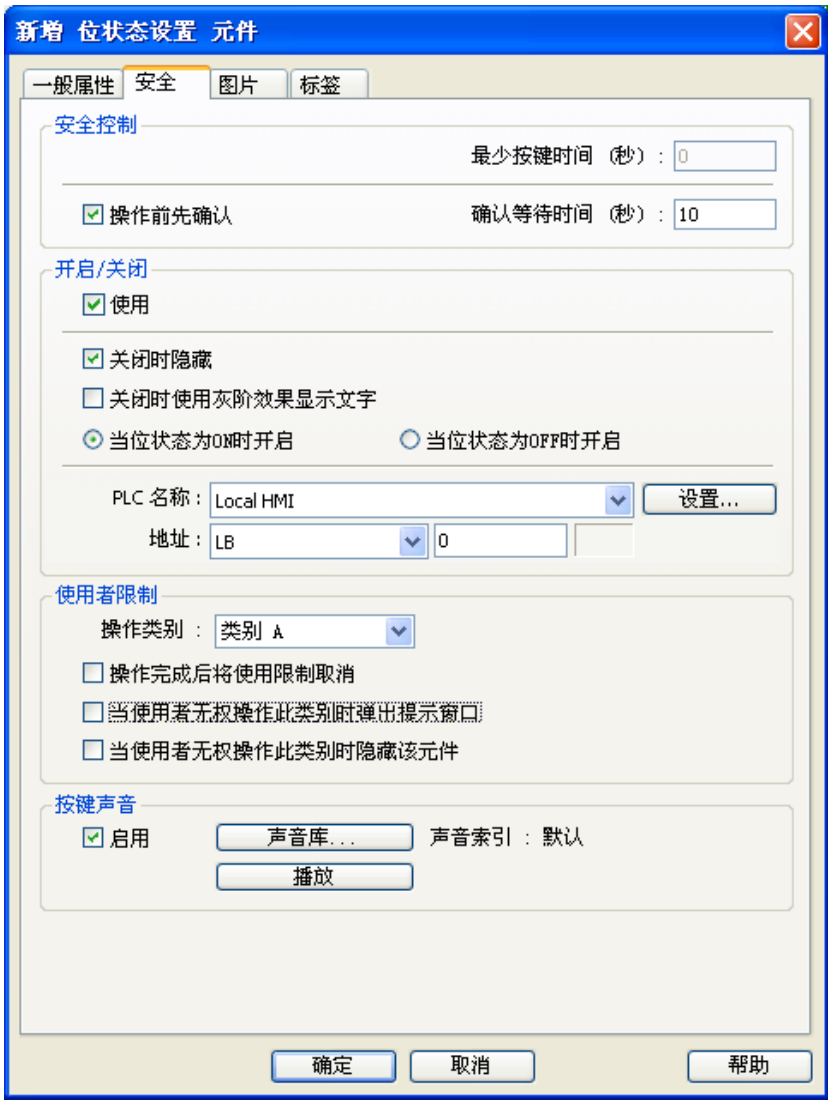
[LW9222]记录目前的用户可以操作的元件类别

bit 0为1表示目前的用户可操作类别属于 “A” 的元件;

bit 1为1表示目前的用户可操作类别属于 “B” 的元件;

其余bit所表示的意义依此类推。

10.2 元件操作安全防护



上图为有关“元件操作安全防护”的内容，可分为几个部分：

a. 安全控制

b. 开启/关闭

c. 使用者限制

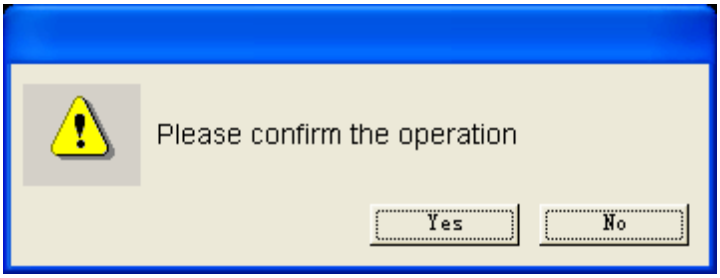
d. 按键声音

a. 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操作元件，目前提供两种保护方式

[最少按键时间]：连续触控元件的时间不小于此项设定值才能成功操作元件。

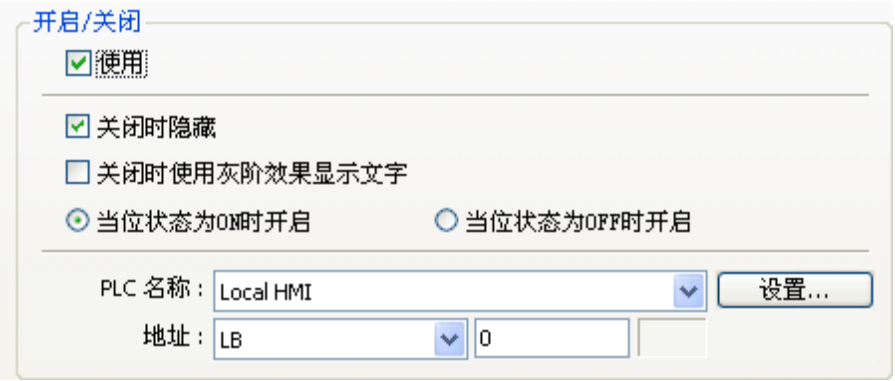
[操作前先确认]: 在触控元件后将出现下图的对话窗, 用户可以依照实际需要, 确认是否执行此项动作。超过[确认等待时间(sec)]所设定的时间后仍未决定是否执行此项动作, 对话框会自动消失并且取消此项执行动作。



对话框中的提示的文字(上图为“操作确认”)被定义在[系统信息]中, 用户可以利用 [系统信息]对话框更改提示文字的内容。触控工具条上的[系统信息]按钮后, 会出现[系统信息]对话框, 其中第一项文字的内容被作为操作确认提示用途。

b. 开启/关闭

当元件使用此项功能时，此元件是否允许被操作，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。

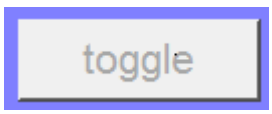


举例来说，假使某一个“位状态设定”元件使用“开启/关闭”，并且它的“开启/关闭”地址为[LB0]，则必须在[LB0]状态为ON时，才允许操作此元件。“开启/关闭”提供下列的设定：

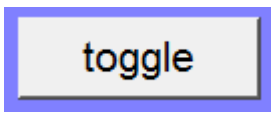
[开启/关闭]：勾选此选项则此元件将使用“开启/关闭”功能。

[关闭时隐藏]：当元件使用“开启/关闭”功能且“开启/关闭”位地址的状态为OFF时，隐藏此元件。

[关闭时使用灰阶效果显示文字]：当元件使用“开启/关闭”功能且“开启/关闭”位地址的状态为OFF时，此物件使用灰阶效果显示文字。



使用灰阶效果



正常显示

c. 用户限制

此项功能可以设定元件的操作类别，元件将只被允许操作此种类别的用户所操作。当“元件操作类别”选择“无”时，表示任意用户皆可操作此元件。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元件的操作条件后，将永远停止对此元件的操作类别限制检查；也就是说即使目前的用户身份改变了，也不会影响对此元件的操作。

[当用户无权操作此类别时弹出提示窗口]：当用户目前的操作身份无法符合此元件的操作条件时，触控此元件将出现警示对话窗，如下图。

注意：EB8000使用7号窗口作为操作身份权限不足时出现的警示对话窗，用户可以自行设计警示对话窗的内容。

[当用户无权操作此类别时隐藏此元件]：当用户目前的操作身份无法符合此元件的操作条件时，隐藏此元件。

d. 按键声音

每个元件可以分别设定是否使用蜂鸣器发出特定声音。EB8000也提供系统保留寄存器[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，EB8000将使用前一次对蜂鸣器的设定状态。

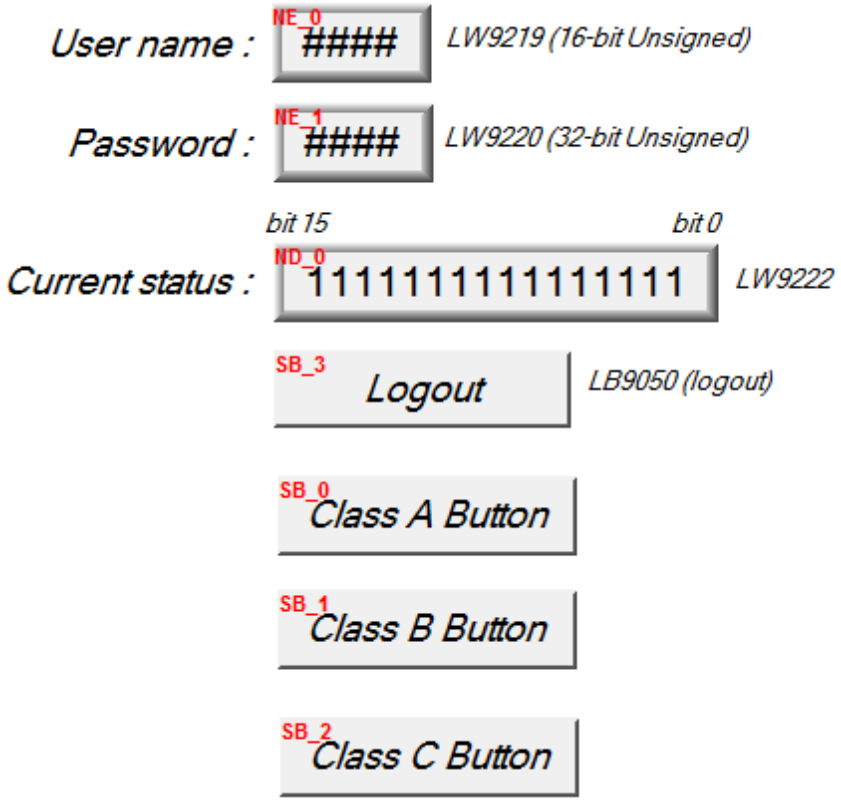
e. 范例说明

下面举例说明元件安全防护的使用。

1、新建立一个工程文件，并在[系统参数]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元件类别。如下图所示：

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

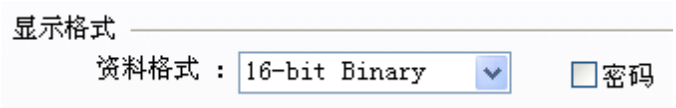
2、可在10号窗口设计如下图所示的元件



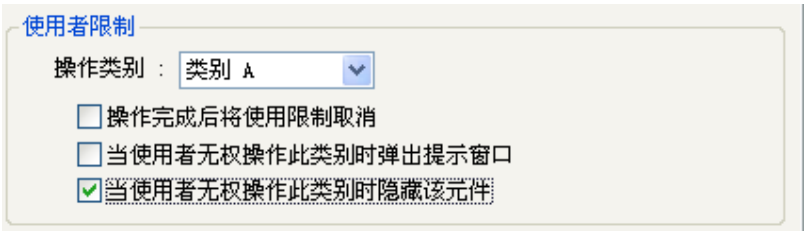
上图的[NE_0]与[NE_1]皆为数值输入元件，地址为[LW9219]与[LW9220]，分别用来输入用户名与用户密码。其中系统保留寄存器[LW9219]被用来输入用户名(1~12)，长度为1个word，因此此元件必须选择16-bit Unsigned的资料格式，如下图：

系统保留寄存器[LW9220]被用来输入用户密码，长度为2个words，因此此元件必须选择32-bit Unsigned的资料格式，如下图：

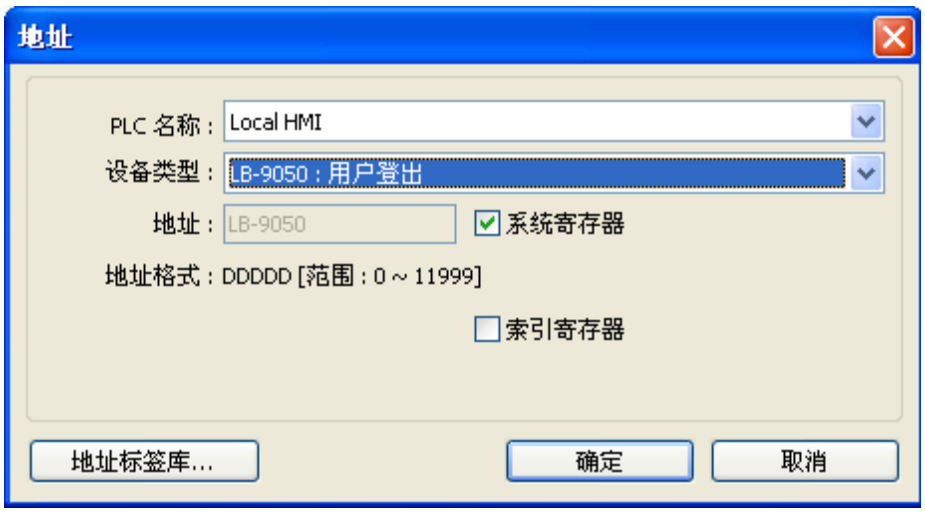
[ND_0]为“数值显示”元件，地址为[LW9222]。用来显示目前用户的状态。此元件必须选择16-bit Binary的资料格式。



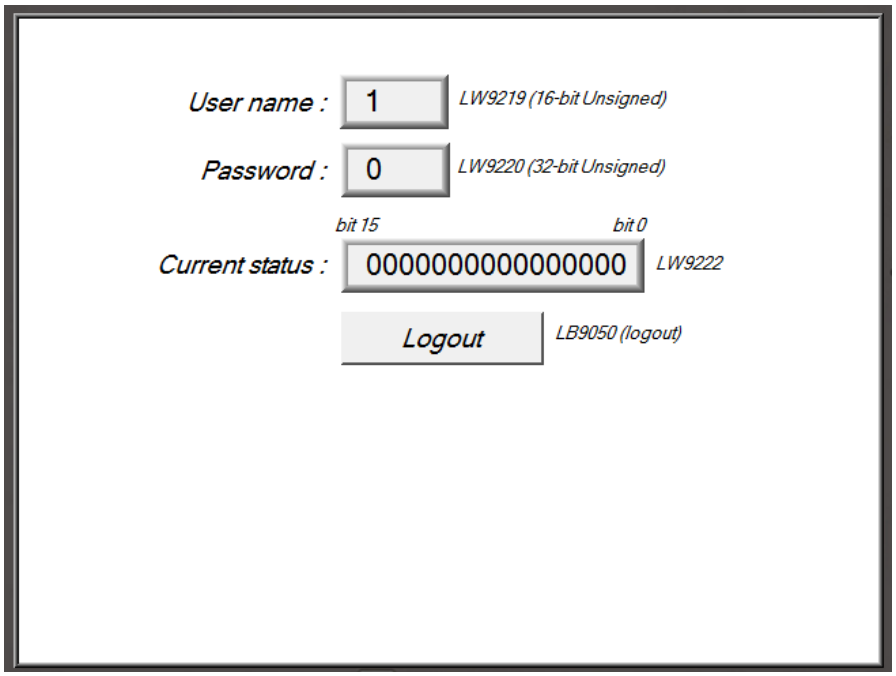
[SB_0]~[SB_2]为“位状态设定”元件，这三个元件刻意选择不同的对象类别，但皆选择[当用户无权操作此类别时隐藏此元件]。其中[SB_0]的对象类别为“A”，[SB_1]的对象类别为“B”，[SB_2]的对象类别为“ C”。下图为[SB_0]的对象类别设定内容。



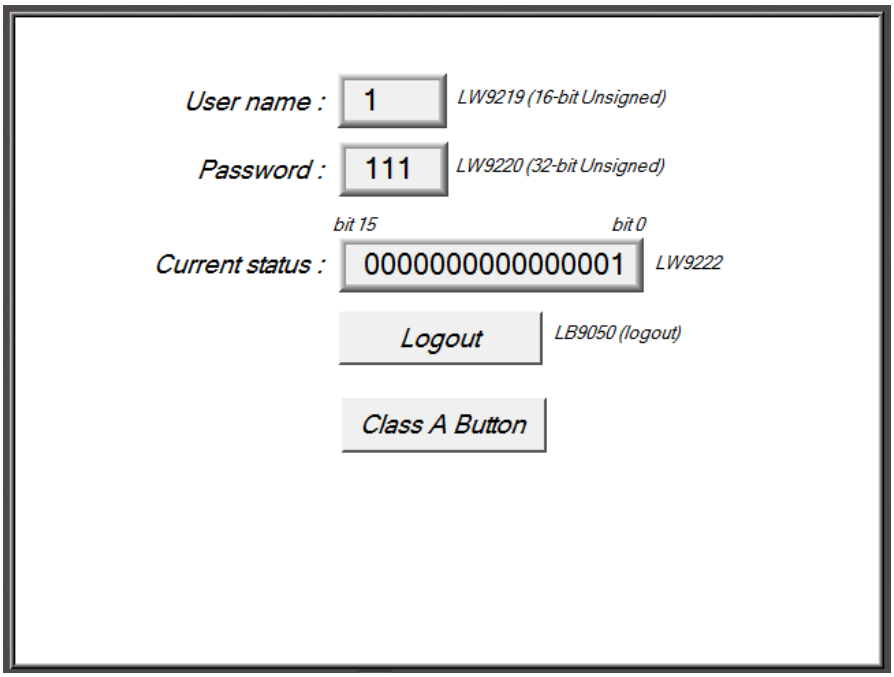
另外画面也设计一个“位状态设定”按键(SB_3, LB9050)，作为用户注销(logout)的用途，参考下图。



完成上述的各项设计并在存盘与编译后即可执行离线模拟功能，下图为离线模拟功能的起始画面，此时因尚未输入任何密码，所以[ND_0]元件显示“00000000000000”，表示目前的使用者仅只能使用类别为“无”的对象，且因[SB_0]~[SB_2]元件分别属于类别“A”~“C”并选择[当用户无权操作此类别时隐藏此元件]，所以[SB_0]~[SB_2]皆被系统所隐藏。

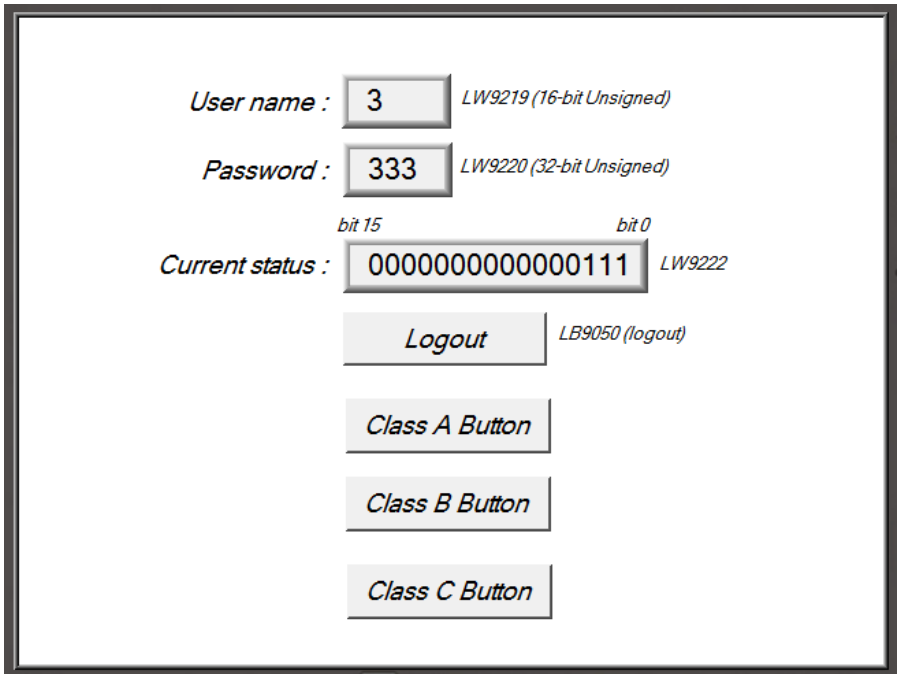


接着使用者可输入用户1的密码(111), 输入完成后画面如下:



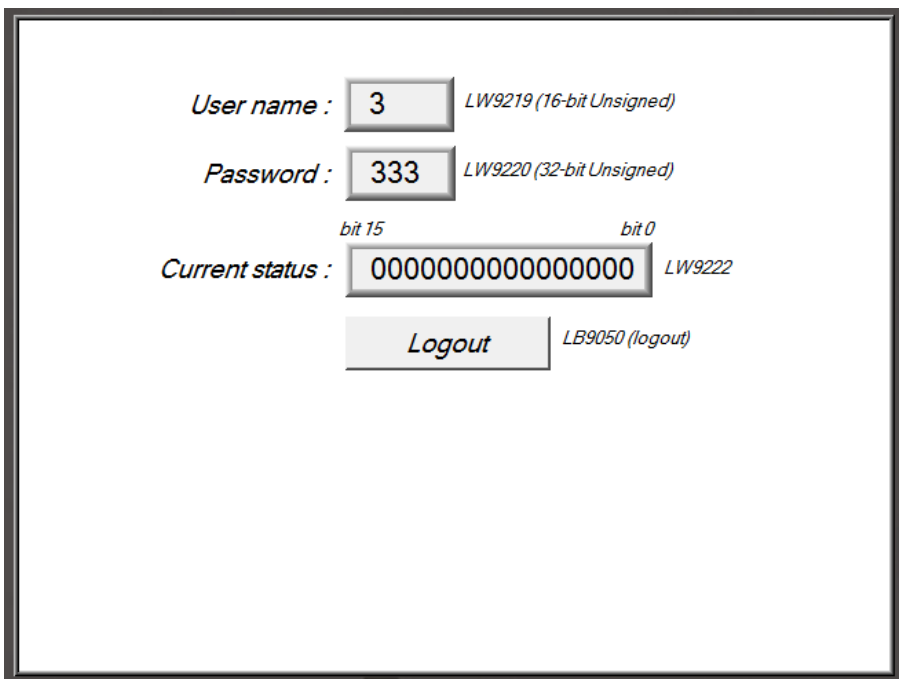
因原来规划“用户1”允许使用类别属于A的元件, 所以此时[SB_0]元件将出现并允许使用者操作。此时也可发现[LW9222]的bit 0已变为1, 表示此时的用户允许使用类别属于A的元件。

接着使用者可输入用户3的密码(333), 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的元件。此时[LW9222]的bit 0~bit 2皆变为1, 表示此时的用户的确被允许使用类别A~C的元件。

此时若触控[SB_3]强迫用户注销, 可以发现系统将恢复到起始状态, 此时将不允许用户操作类别不属于 “无” 的对象。



第十一章　索引寄存器

概要

索引寄存器是EB8000软件提供的用于变换地址的寄存器，有了索引寄存器后，用户可以在不改变元件地址内容的情况下，画面程序运行时，在触摸屏上就可以直接修改元件的读取和写入地址。EB8000提供32个索引寄存器，让用户对地址的操作更具弹性。

32个索引寄存器的位置分别为：

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)

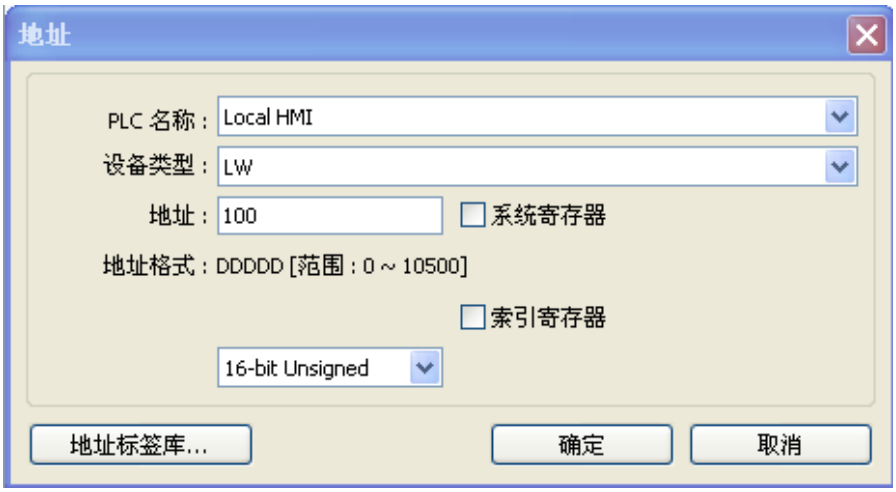
其中INDEX 0 ~ INDEX 15为16–bit索引寄存器，INDEX 16 ~ INDEX 31为32–bit索引寄存器。因此INDEX 0 ~ INDEX 15可以寻址的范围为65536 words，INDEX 16 ~ INDEX 31可以寻址的范围为 4G words。

使用“索引寄存器”后，所使用“设备类型”的地址则为“设定的常数地址+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”中所有“设备列表”均有效，对bit格式和word格式的地址都有效。

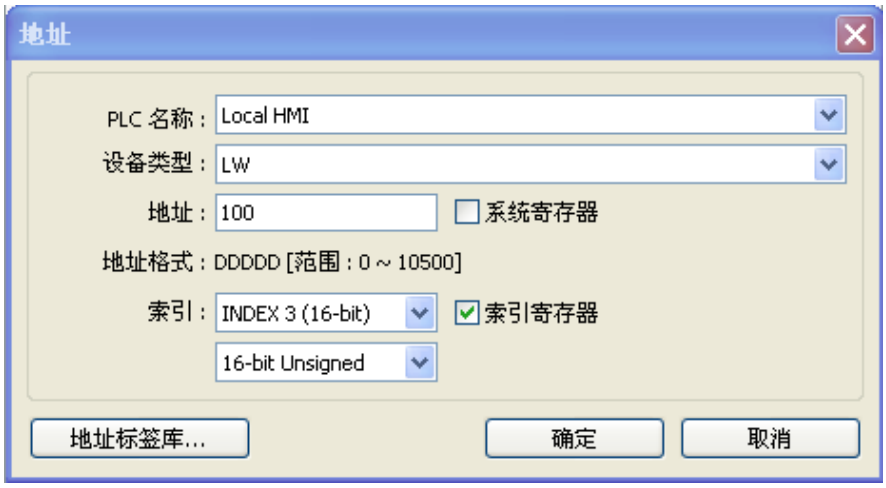
范例

以一个实际例子说明索引寄存器的使用方式；以下图为例，因为未勾选[索引寄存器]选项，此时的读取地址为[LW100]。

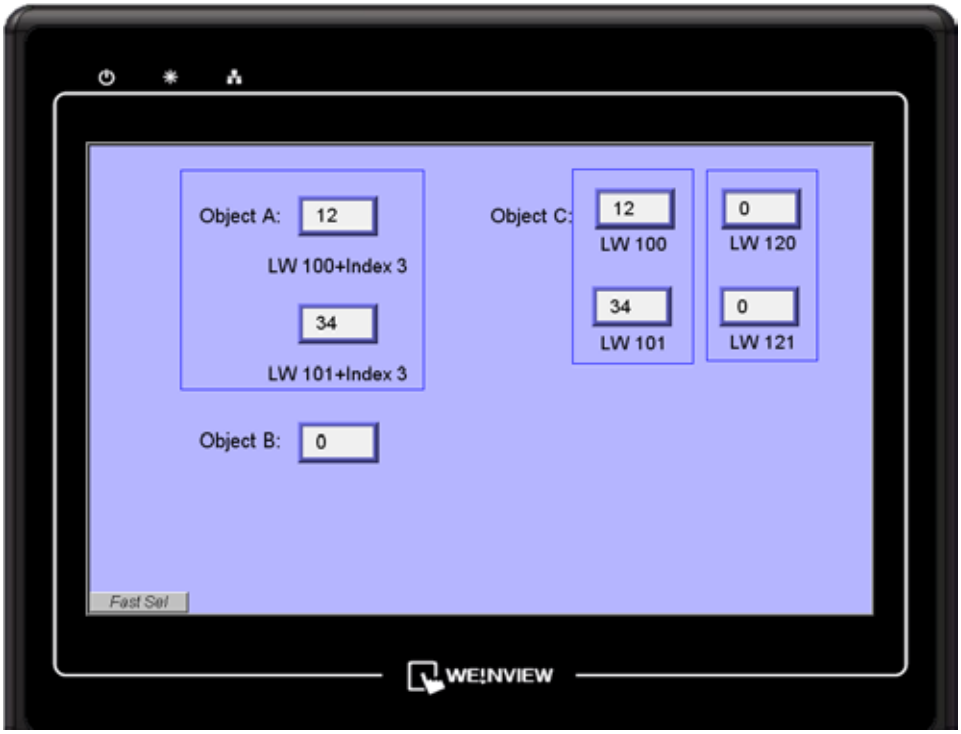
触控“设置”按钮



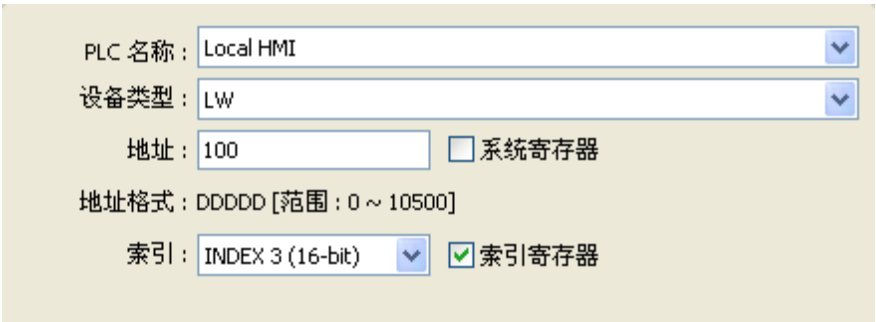
但下图的[索引寄存器]选项被勾选，且选择的索引寄存器为INDEX3，此时的读取位置变为[LW(100 + INDEX 3)]，其中的INDEX 3表示索引寄存器3或 [LW9203]地址中的数据。也就是说如果此时[LW9203]地址中的数据为5，则下图的读取位置变为[LW（100+5）]，即LW[105].



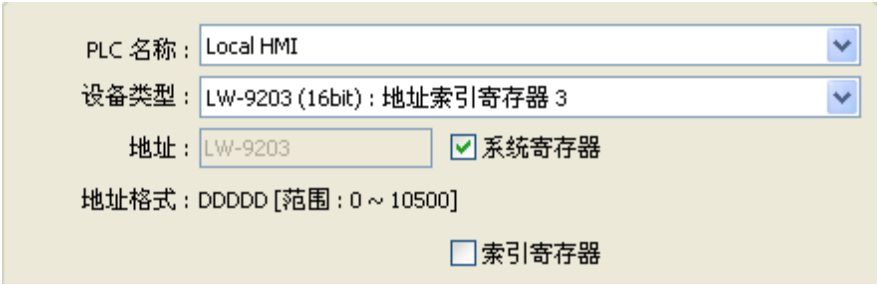
下图显示，此时INDEX3为0，也就是[LW9203]地址中的数据为0，则读取[LW100+INDEX3]将等同读取[LW100]的内容。



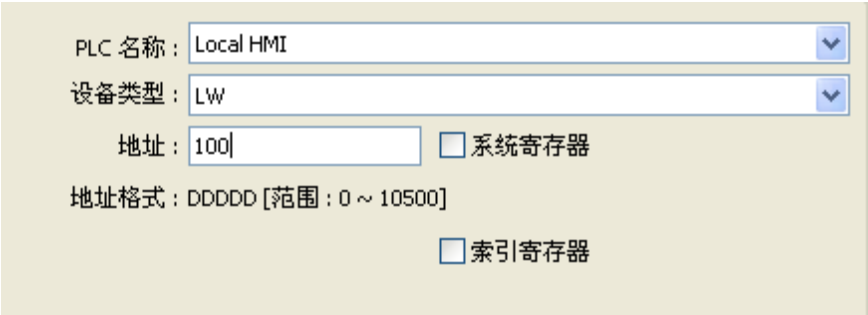
此时Object A的读取地址设定如下图。



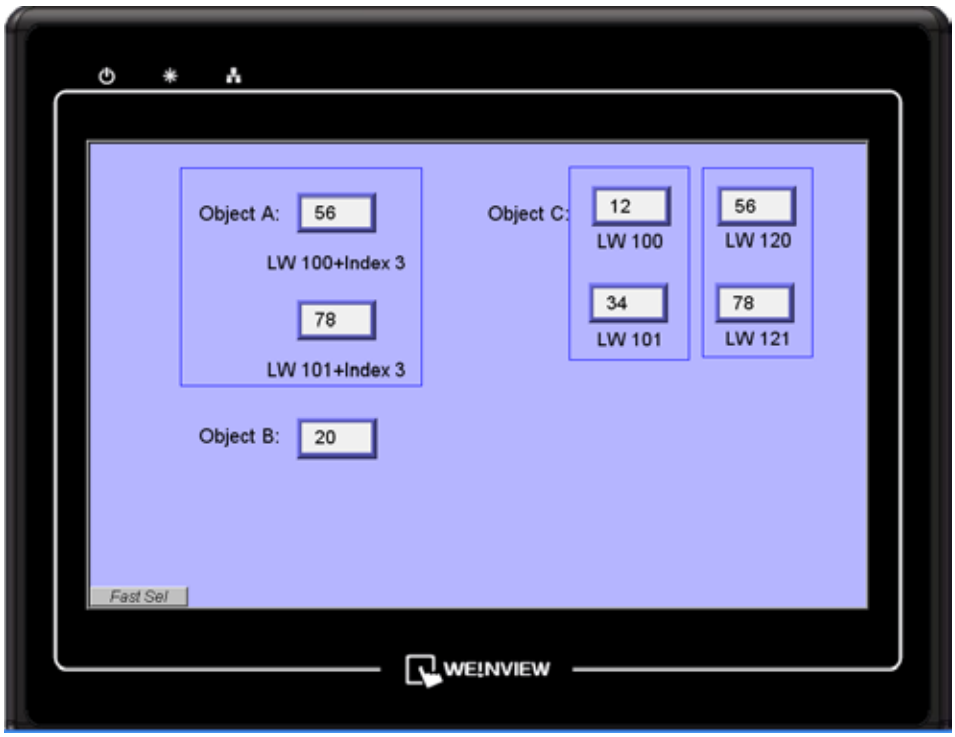
Object B的读取地址设定如下图。



Object C的读取地址设定如下图。



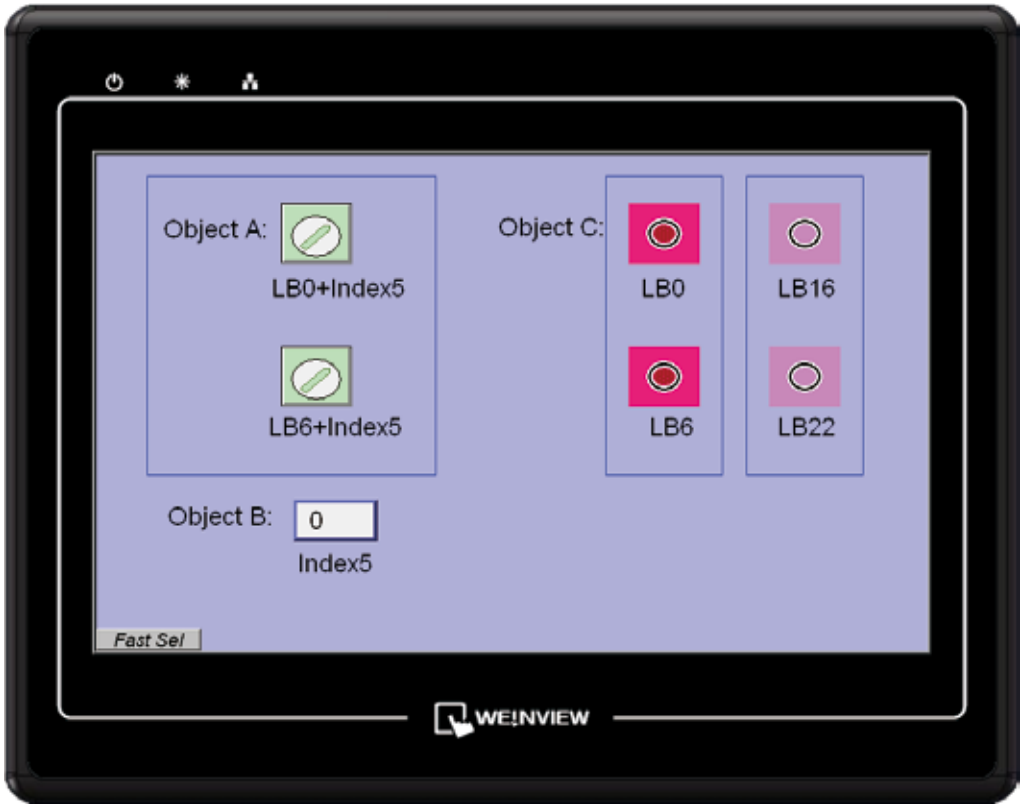
此时若将INDEX3的内容设定为20，则读取LW(100+INDEX3)及LW（101+INDEX3）将等同读取LW120及LW121的内容，如下图所示。



同样的，索引寄存器也可以使用于位元件地址。

1个字元=16个位元，所以索引寄存器数值改变1相当于16个位元。

如下图所示，当index 5设定为“0”，此时位状态指示灯LB0及LB6的状态是与位状态切换开关[LB(0+index 5)]及[LB(6+index 5)]相同，显示为ON。



现在如果我们将index 5数值设定为“1”，此时位状态指示灯LB16和LB22的状态是和位状态切换开关的[LB(0+index 5)]和[LB(6+ index 5)]相同，显示为ON。

总结：经过上面的说明，我们了解到索引寄存器其实就是一个变址寄存器，通过索引寄存器，我们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的值，即可改变同一个元件读取或者写入不同的地址的资料。这样，我们就可以实现不同区域地址间资料的传送和交换。

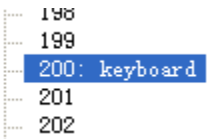
第十二章 键盘的设计与使用

“数值输入”与“字元输入”皆需使用键盘做为输入工具，除了可以使用呼叫键盘的方式,另外还能呼叫没有移动窗口控制条的直接窗口，或者直接将键盘设计在屏幕上的固定位置。同时也可以制作Unicode文字输入的键盘。数字键盘以及字元键盘均是使用“功能键”来制作的，下文说明键盘的设计过程及用法。

设计自制的弹出键盘

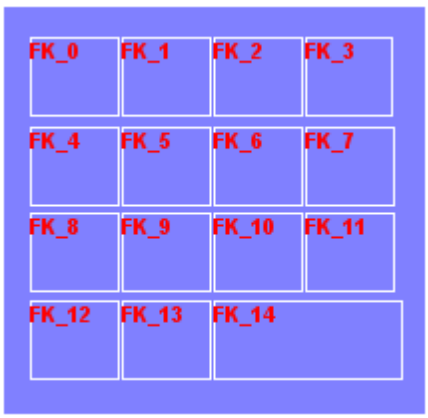
步骤一

先建立并开启要作为键盘的窗口，假设现在将“窗口200”作为键盘所在页。

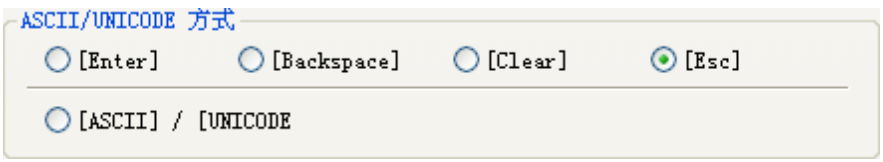


步骤二

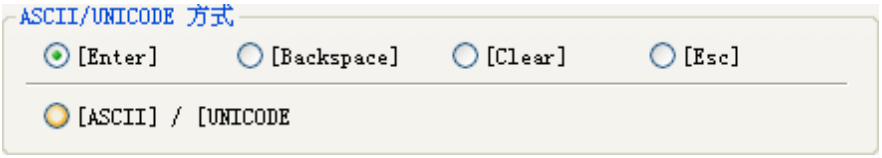
调整“窗口200”的长度与宽度，并在上面安排各式“功能键”元件，当触控“功能键”元件时将触发各种输入信号。



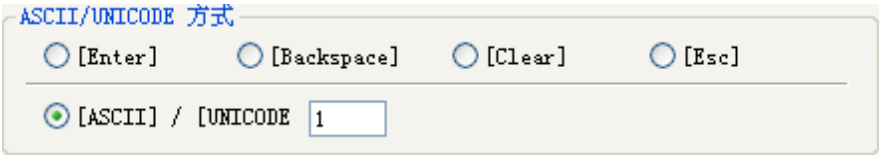
“窗口200”上的“功能键”元件安排如上图，此时的“功能键”元件皆须选择[ASCII/UNICODE模式]，其中FK_11用来触发取消(ESC)信号，FK_11的部分设定内容如下图。



FK_14用来触发“输入”(Enter)信号，FK_14的部分设定内容如下图。



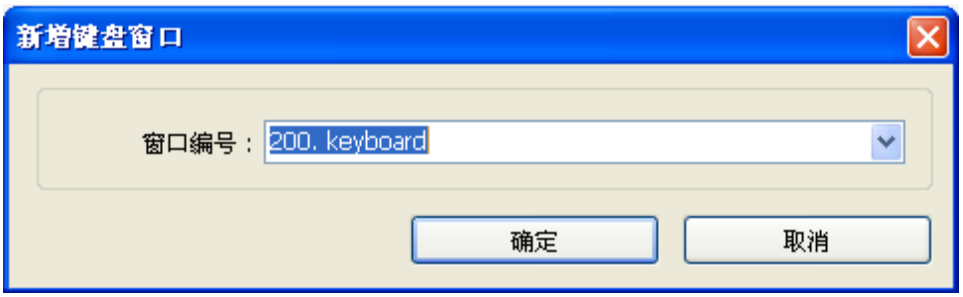
其它大部分“功能键”元件是用来触发数值或文字输入信号，例如FK_0是用来触发数值“1”的输入信号，FK_0的部分设定内容如下图。



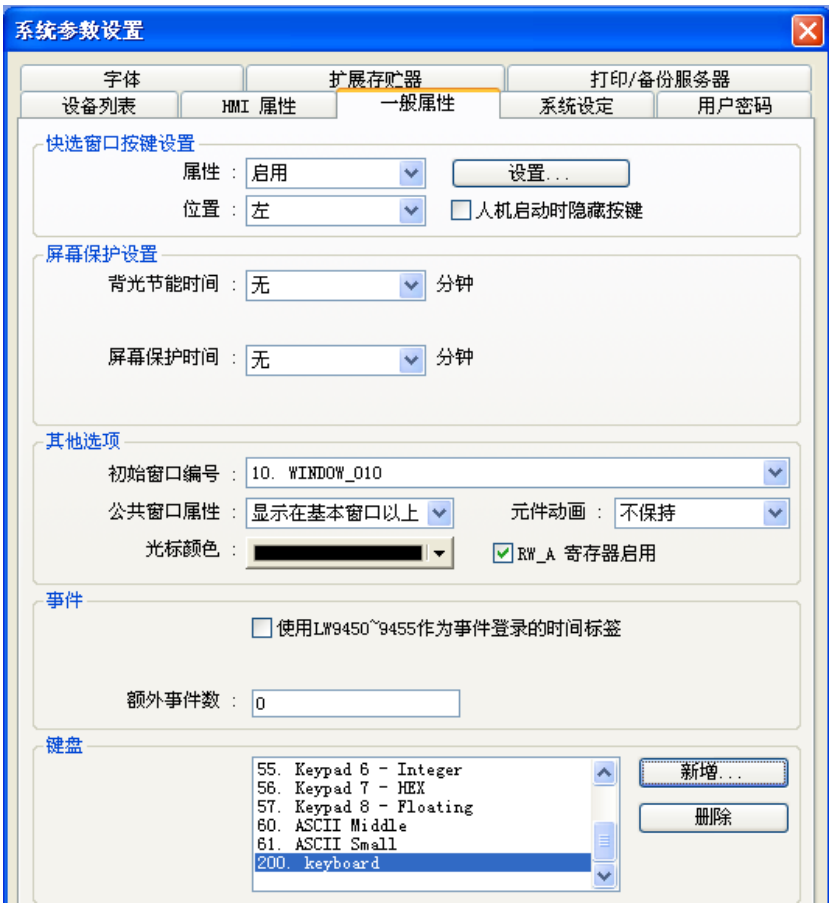
最后为“功能键”元件选择合适的图形，GP_0为“图片”元件，被置于所有元件的最下层，作为背景图案，如下图。

步骤三

在系统参数的[一般属性]中的[键盘]设定项目中，触控[新增…]后可得到下图所示的对话窗，此时选择加入“窗口200”并触控确认键。



最后会发现在系统参数的[一般属性]中的[键盘]设定项目中增加了一个项目：“200. Keyboard”，如下图所示。



在完成上述的所有步骤后，当用户打开“数值输入”或“字元输入”元件的设定页时，即可发现在[键盘]设定项目的[窗口编号]中，增加了“200. 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。EB8000将屏幕分为9个区域，键盘的左上角将出现在所选择区域的左上角位置。

在选择“200. Keyboard”后，当用户触控“数值输入”或“字元输入”元件时，EB8000画面中将自动弹出“窗口200”，触控“窗口200”上的“功能键”元件将等同键盘输入信号，如下图所示。

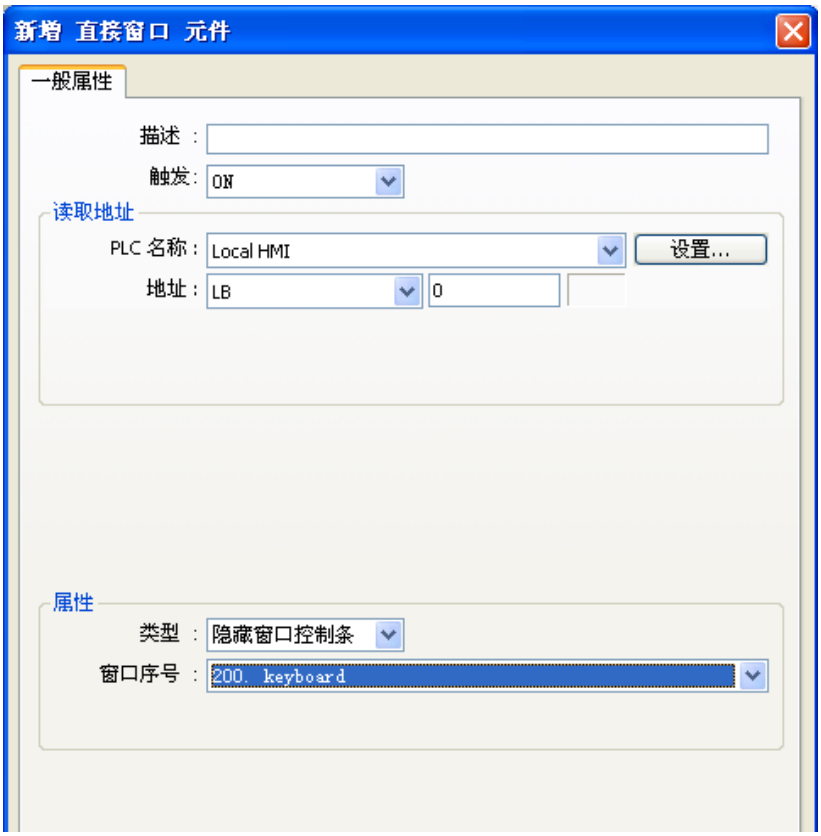
使用直接窗口的方式来设计键盘

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。

步骤一

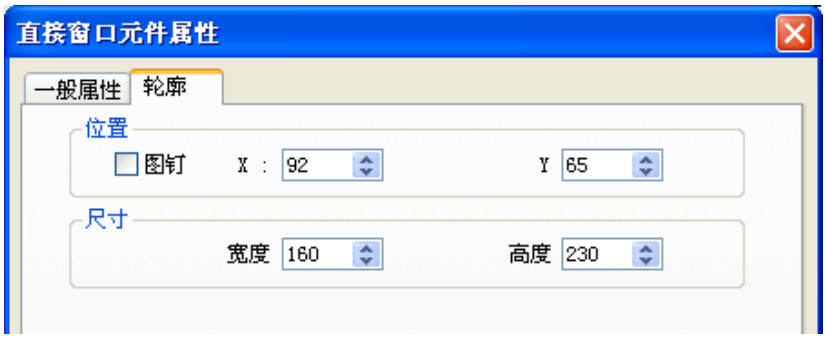
新增一个直接窗口，设定读取地址来激活直接窗口。

在属性内选择隐藏窗口控制条及键盘所在窗口序号。



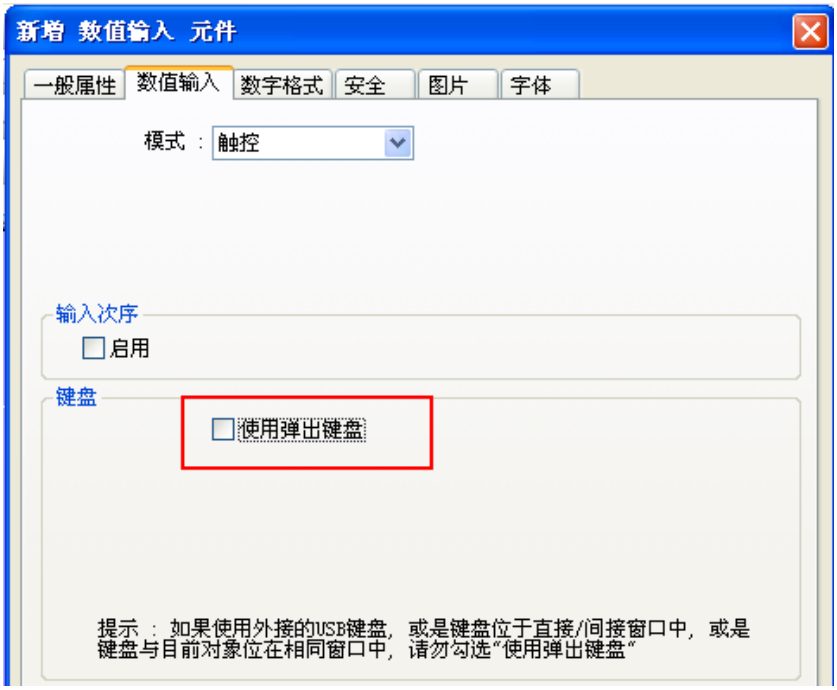
步骤二

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



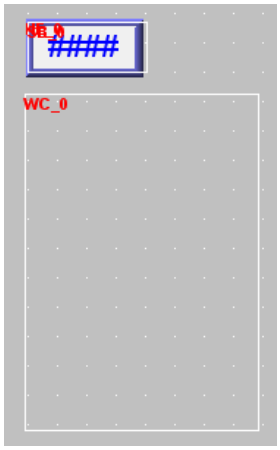
步骤三

新增数值输入元件，在数值输入属性内不要勾选“使用弹出键盘”



步骤四

设定一个位状态设定元件LB0，设为ON，并重叠在数值输入元件上。然后在该需要弹出的“键盘”的ENTER键和ESC功能键上，分别放置一个位状态设定元件LB0，设为OFF，这样就会在触控这两个键的任何一个时，将“键盘”即直接窗口关闭。



将键盘固定在属于输入的窗口上

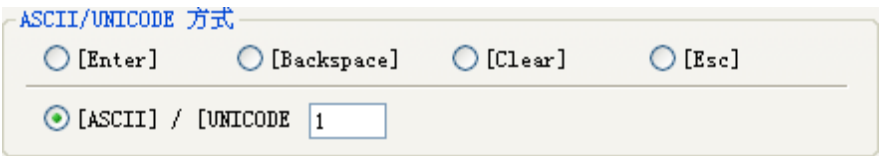
另外可以设定将功能键固定在屏幕上而不是采用调用方式或是使用直接窗口来固定键盘所在位置，采用此方式则无法移动或取消键盘。

步骤一

新增数值输入元件，在键盘属性中那个不勾选“使用弹出键盘”

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：



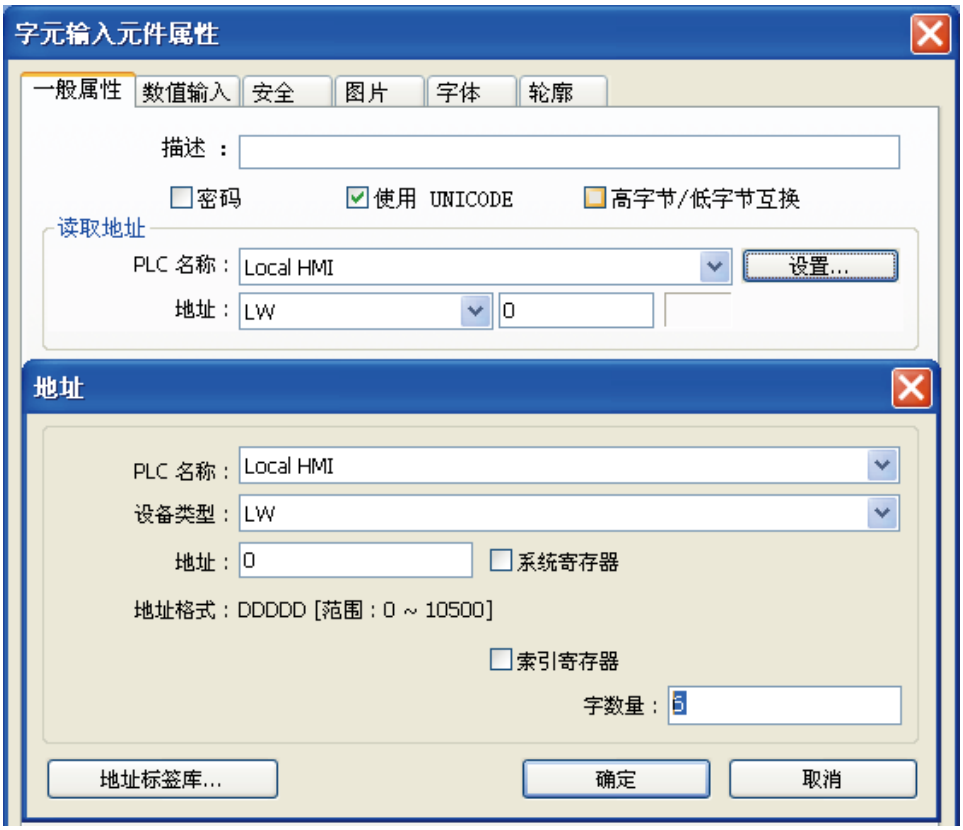
步骤三

当触控“数值输入”元件时，用户可以用键盘上的功能键来输入数据：

制作UNICODE键盘

制作UNICODE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“威纶通”这三个文字输入功能键，再制作一个ENTER输入功能键，即做好了一个简单的文字键盘。再放置一个“字元输入”元件在窗口画面上，字数选择为6（一个文字为2个字元），并勾选“使用UNICODE”，最后做好的画面如下所示：



小结：数字键盘和字元键盘均是使用“功能键”元件制作，并结合在一起形成的。用户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，也可以将自制的键盘，添加到：系统参数–一般属性– 键盘中，作为新增的系统键盘。

第十三章 元件

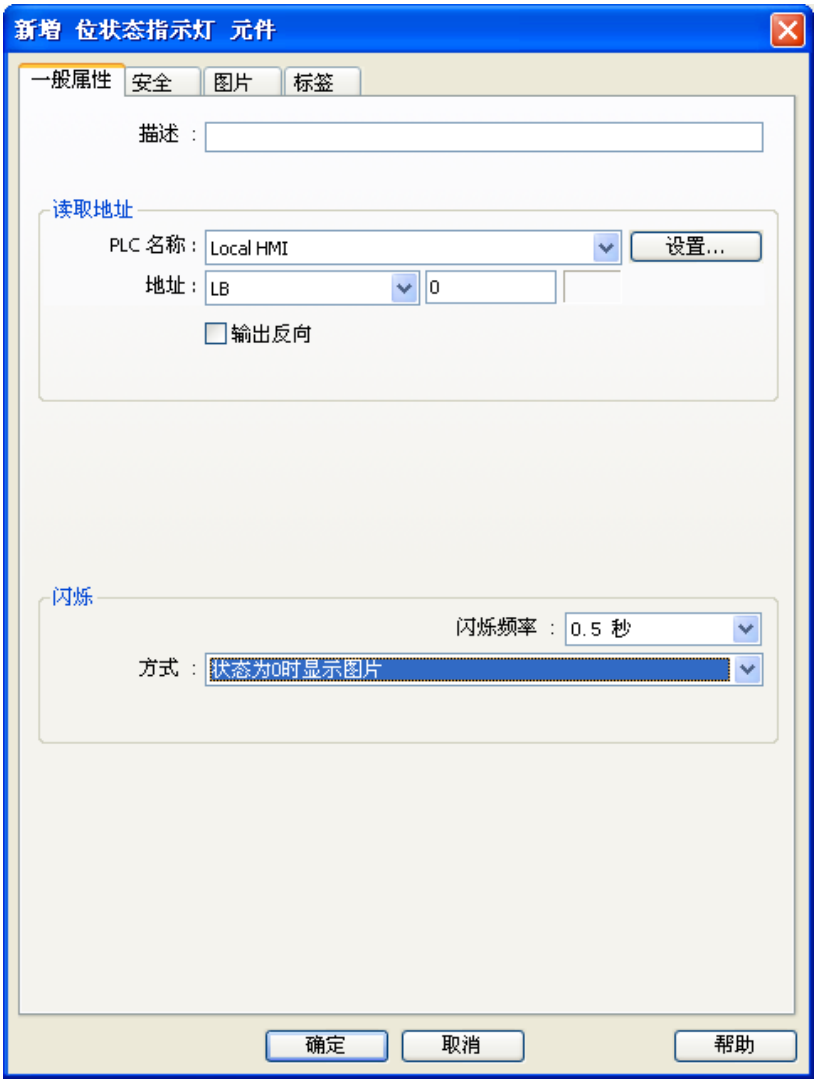
本章说明各种元件的使用方式与相关的设定，未说明的设定请参考《第九章 元件一般属性》中的说明。

13.1 位状态指示灯元件 (bit lamp)

“位状态指示灯”元件用来显示位寄存器的状态。状态为OFF，则显示所使用图形的状态0；状态为ON，则显示所使用图形的状态1。

[设定]：

触控工具条上的“位状态指示灯”按钮后即会出现“位状态指示灯元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“位状态指示灯”元件，参考下图。

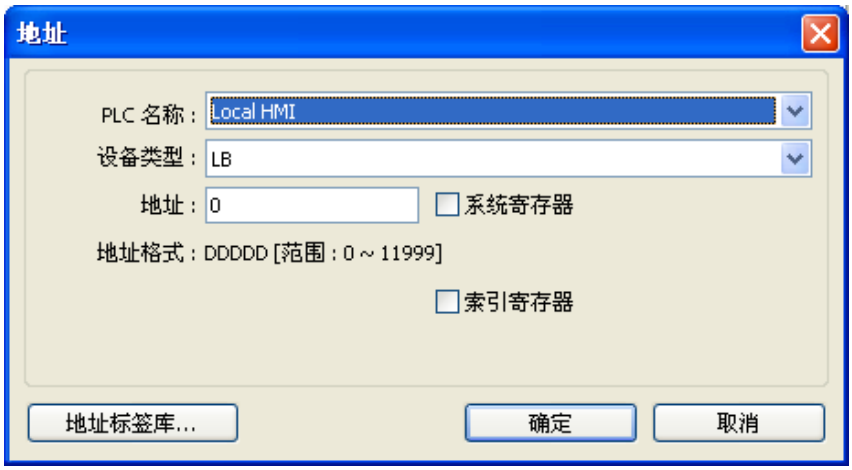


描述

用户可以为此元件描述相关信息。

读取地址

点击“设置”后选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制位状态指示灯元件，用户也可在“一般属性”页中设定位地址。



输出反相

可以将读取的状态作反相显示，例如获得的状态为OFF，但元件会显示ON的图形。

闪烁项目

设定元件的闪烁方式。

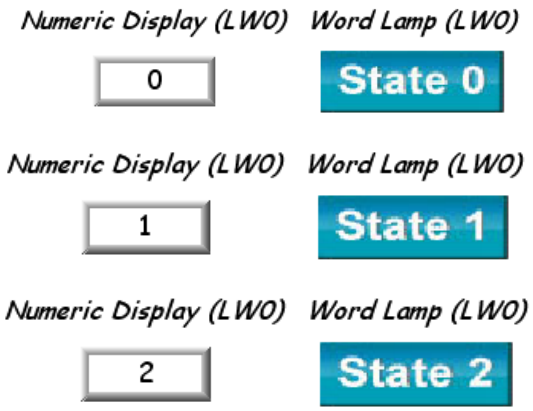
[模式]

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。

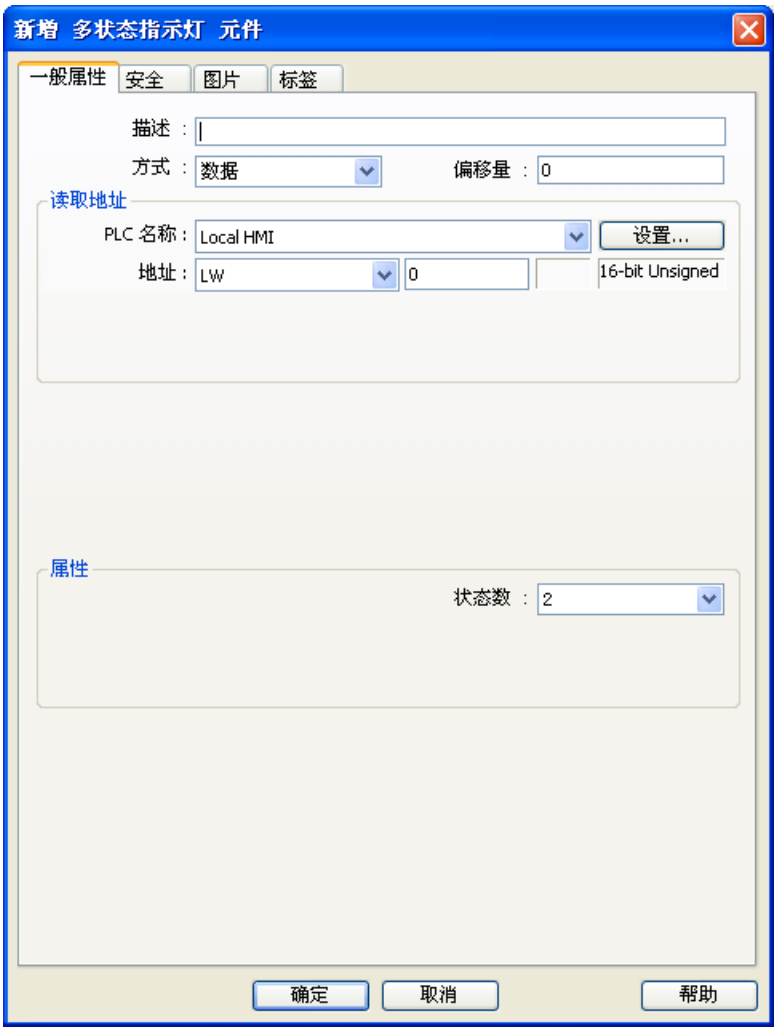
13.2 多状态指示灯元件 (word lamp)

“多状态指示灯”元件利用寄存器内的数据，显示相对的状态与图形(EB8000最多支持256种状态的显示)。



设定

触控工具条上的“多状态指示灯”按钮后即会出现“多状态指示灯元件属性对话窗”，正确设定各项属性后触控确认键，即可新增加一个“多状态指示灯”元件，参考下图。



[方式]/ [偏移量]

“多状态指示灯”元件提供下列三种显示方式：

a. “数据”显示方式

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

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

LW200　　LW200, 偏移量=3

b. “LSB”显示方式

此方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的最低位决定元件目前的状态。以地址[LW200]地址内的数据为例：

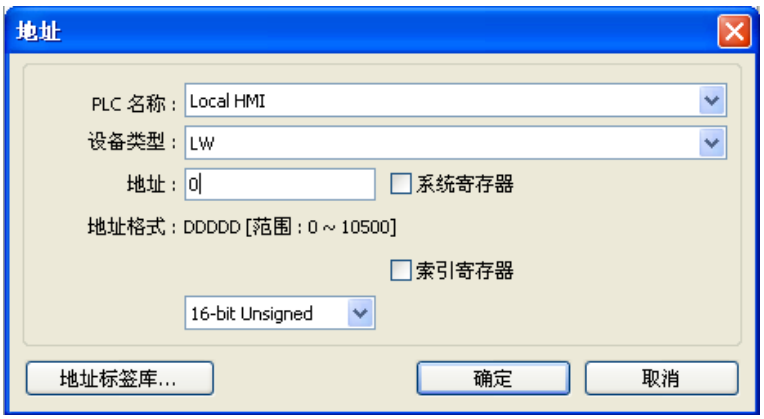
十进制	二进制	显示的状态
0	0000	全部 bit 皆为 0，则显示状态 0
1	0001	不为 0 的最低位为 bit 0，此时显示状态 1
2	0010	不为 0 的最低位为 bit 1，此时显示状态 2
3	0011	不为 0 的最低位为 bit 0，此时显示状态 1
4	0100	不为 0 的最低位为 bit 2，此时显示状态 3
7	0111	不为 0 的最低位为 bit 0，此时显示状态 1
8	1000	不为 0 的最低位为 bit 3，此时显示状态 4

c. “周期转换状态” 显示方式

元件的状态与寄存器无关，元件会使用固定的频率依序变换状态。用户可以利用[频率]设定状态改变频率。

读取地址项目

点击 “设置” 后选择字寄存器设备类型的 “PLC名称” 、“地址” 、“设备类型” 、“系统寄存器” 、“索引寄存器” 来控制多状态指示灯元件。用户可以在 “一般属性” 页中设定地址：

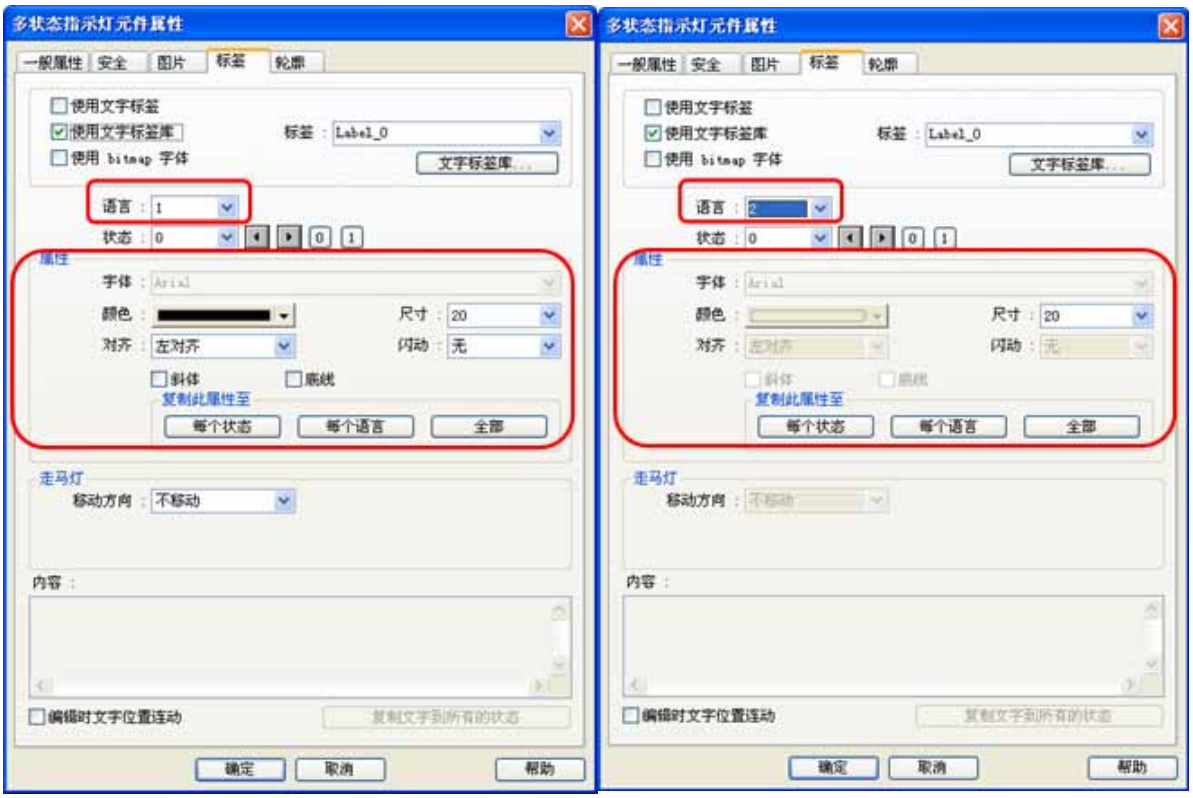


属性项目

[状态号]

元件的状态数目，状态从0开始编号，因此能显示的最大状态为[状态号] – 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，EB8000会显示最后一个状态。

限制：在标签页中，语言1能够改变自行相关性设定，但是语言2–8只能改变字的尺寸，其他属性皆与语言1相同。参考下图：


多状态指示灯元件属性
一般属性
安全
图片
标签
轮廓
使用文字标签
使用文字标签库
使用 bitmap 字体
标签 : Label_0
文字标签库...
语言 : 1
状态 : 0
属性
字体 : Arial
颜色 :
尺寸 : 20
对齐 : 左对齐
闪动 : 无
斜体
底线
复制此属性至
每个状态
每个语言
全部
走马灯
移动方向 : 不移动
内容 :
编辑时文字位置连动
复制文字到所有的状态
确定
取消
帮助
多状态指示灯元件属性
一般属性
安全
图片
标签
轮廓
使用文字标签
使用文字标签库
使用 bitmap 字体
标签 : Label_0
文字标签库...
语言 : 2
状态 : 0
属性
字体 : Arial
颜色 :
尺寸 : 20
对齐 :
闪动 :
斜体
底线
复制此属性至
每个状态
每个语言
全部
走马灯
移动方向 :
内容 :
编辑时文字位置连动
复制文字到所有的状态
确定
取消
帮助

13.3 位状态设置元件 (set bit)

“位状态设置”元件提供“手动操作”与“自动执行”两种操作方式。使用“手动操作”方式，用户可以利用“位状态设置”元件在窗口上定义一个触控区域，触控此区域可以将寄存器的状态设定为ON或OFF。

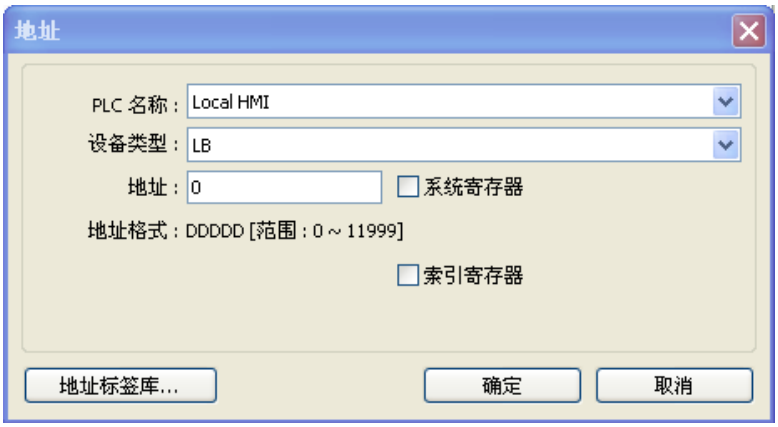
若使用“自动执行”方式，则在某些特定条件下会自动执行元件定义的动作，使用此种操作方式，在触控元件定义的触控区域时，元件将不作任何反应。

设定

触控工具条上的“位状态设置”按钮后即会出现“位状态设置元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“位状态设置”元件，参考下图。

写入地址项目

点击“设置”后，选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制位状态设置元件，用户可在一般属性页中设定地址



[当按钮松开才发出指令]

使用此项设定表示在触控元件后，必须完全松开触控动作，元件定义的操作方式才会被执行。如未使用此项设定，只要一触控此区域，将立刻执行元件的动作。但操作方式如果选择复归型(momentary)开关，将不支持此项功能。

属性

[开关类型]

选择元件的操作方式，可选择项目如下：

设为ON	在触控元件定义的区域后，所指定寄存器的状态将被设定为ON。
设为OFF	在触控元件定义的区域后，所指定寄存器的状态将被设定为OFF。
切换开关	切换型开关。每次触控元件定义的区域后，所指定寄存器的状态将被反相。也就是状态由OFF变为ON或由ON变为OFF。
复归型	复归型开关。每次触控元件定义的区域时，所指定寄存器的状态将先被设定为ON，但离开触控区域后，状态将被设定为OFF。
周期切换开关	周期性复归型开关。所指定寄存器的状态将在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窗口(combo box)选择切换周期。 时间间隔： 1.0 秒
窗口打开时设ON	元件所在位置的窗口被打开时，所指定寄存器的状态将自动被设定为ON。

窗口打开时设OFF	元件所在位置的窗口被打开时，所指定寄存器的状态将自动被设定为OFF。
窗口关闭时设ON	元件所在位置的窗口被关闭时，所指定寄存器的状态将自动被设定为ON。
窗口关闭时设OFF	元件所在位置的窗口被关闭时，所指定寄存器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，所指定寄存器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，所指定寄存器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，所指定寄存器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，所指定寄存器的状态将自动被设定为OFF。

宏指令项目

操作“位状态设置”元件时，可以搭配执行宏指令(macro)，但要选择此项功能前需先建立宏指令，如何建立宏命令请参考相关的章节。

[触发模式]

当元件的操作方式，选择“切换开关”时，可以设定执行宏指令的条件，可以选择状态由OFF变为ON或由ON变为OFF时，才执行宏指令，也可选择状态改变时 (ON<->OFF)，即执行宏指令。除此之外，其它操作方式皆在状态改变时，即执行宏命令。

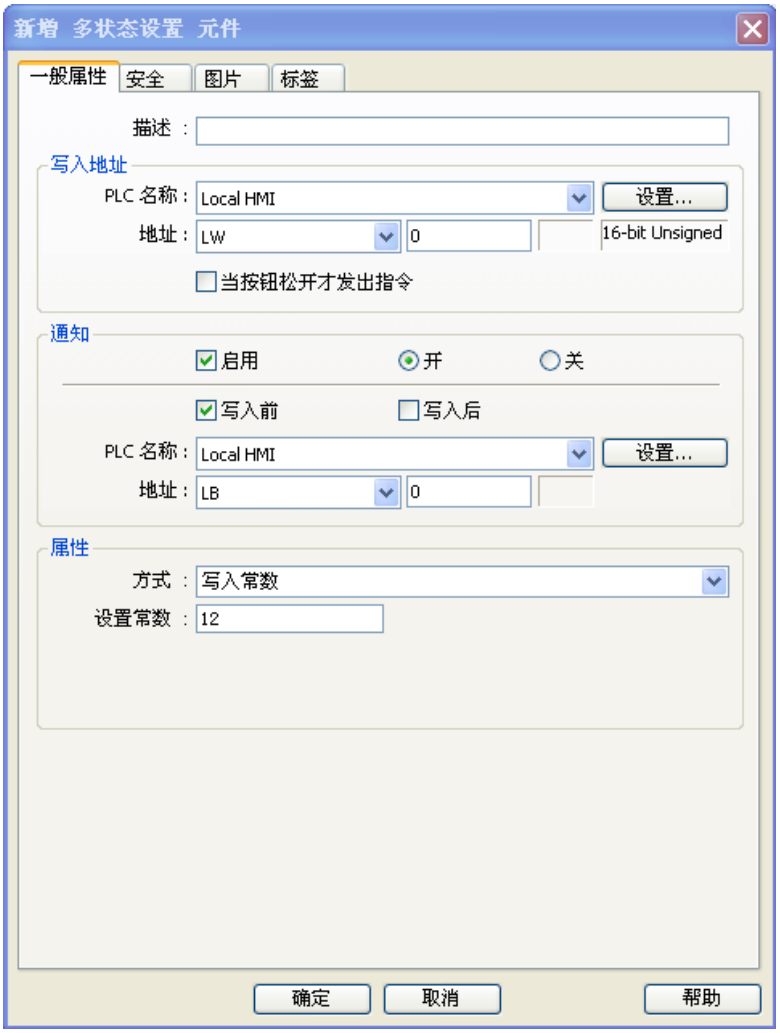
13.4 多状态设置元件 (set word)

“多状态设置”元件提供“手动操作”与“自动执行”两种操作方式。使用“手动操作”方式，用户可以利用“多状态设置”元件在窗口上定义一个触控区域，触控此区域可以设定所指定寄存器内的数据。

若使用“自动执行”方式，则在某些特定条件下会自动执行元件定义的动作，使用此种操作方式，在触控元件定义的触控区域时，元件将不作任何反应。

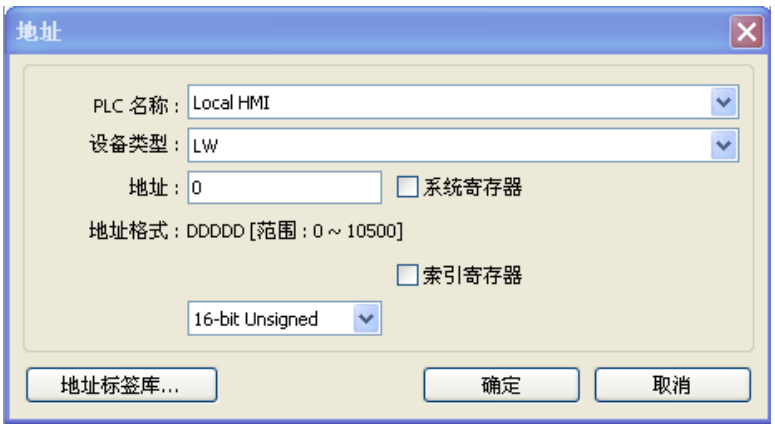
设定

触控工具条上的“多状态设定”按钮后即会出现“多状态设置元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“多状态设置”元件，参考下图。



写入地址项目

点击“设置”按钮后选择字寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制多状态设置元件，用户也可以在“一般属性”页中设定地址。



[当按钮松开才发出该指令]

使用此项设定表示在触控元件定义的触控区域后，必须完全离开此区域才会执行元件定义的动作。如未使用此项设定，则只要一触控到此区域，将立刻执行元件定义的动作。

通知项目

使用此项设定，则在使用“手动操作”方式时，在完成动作后可以连带设定此项目所指定寄存器的状态，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。

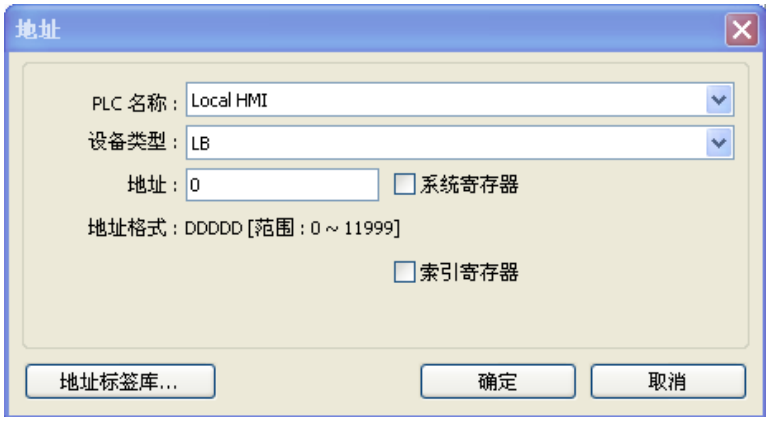
[写入前]

在写入动作进行前先设定所指定寄存器的状态。

[写入后]

在写入动作完成后才设定所指定寄存器的状态。

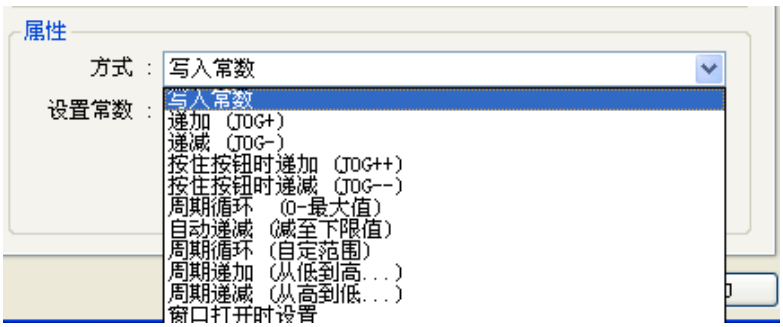
点击“设置”后选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制通知位地址项目，用户也可以在“一般属性”页中设定位地址。



属性项目

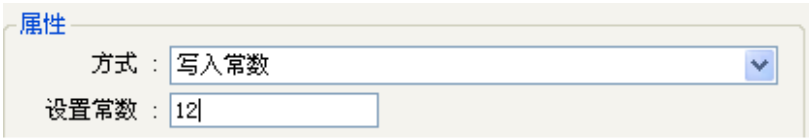
[方式]

选择元件的动作方式，可以选择的方式如下：



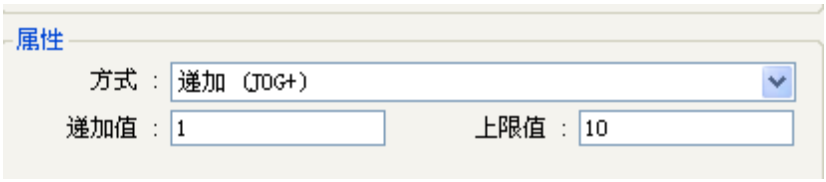
a. “写入常数”

设置常数功能。每触控一次元件，[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，在 “写入地址” 项目中决定。



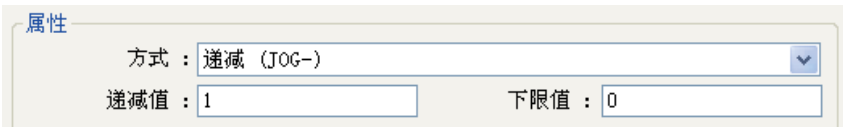
b. “递加 (JOG+)”

加值功能。每触控一次元件，所指定寄存器内的数据将加上[递加值]的设定值，但递加的结果将不超过[上限值]中的设定值。



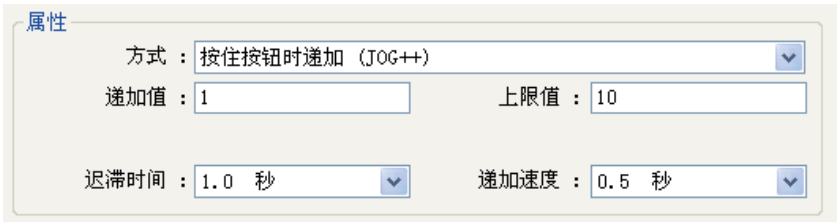
c. “递减 (JOG-)”

减值功能。每触控一次元件，所指定寄存器内的数据将减去[递减值]中的设定值，但是递减的结果不会低于[下限值]中的设定值。



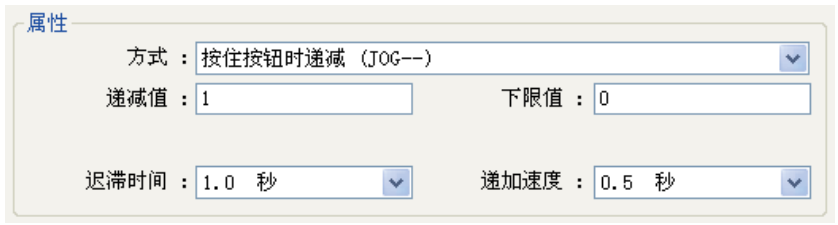
d. “按住按钮时递加 (JOG++)”

按住按钮时递加功能。若触控元件超过[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]中设定的值，但递加的结果将不超过[上限值]中的设定值。



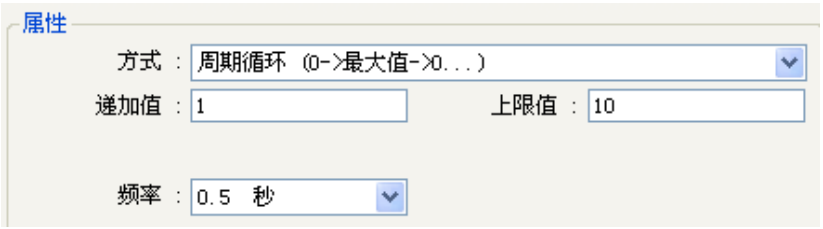
e. “按住按钮时递减 (JOG--)”

按住按钮时递减功能。若触控元件超过[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，每次减少[递减值]中的设定值，但递减的结果不会低于[下限值]中的设定值。



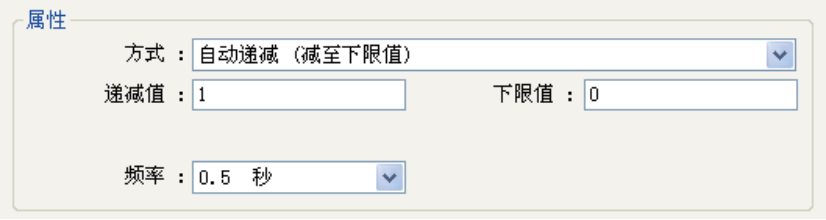
f. “周期循环 (0->最大值->0)”

周期性循环功能。“多状态设置” 元件会使用[频率]设定的周期与[递加值]中的设定值，自动增加指定寄存器内的数据，但增加的结果不超过[上限值]中的设定值。



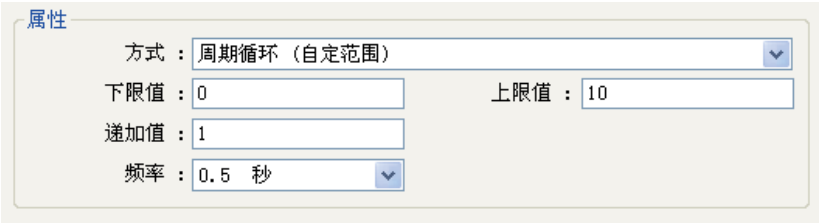
g. “自动递减 (减至下限值)”

周期性递减功能。“多状态设置” 元件会使用[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]中的设定值，但递减的结果将不低于[下限值]中的设定值。



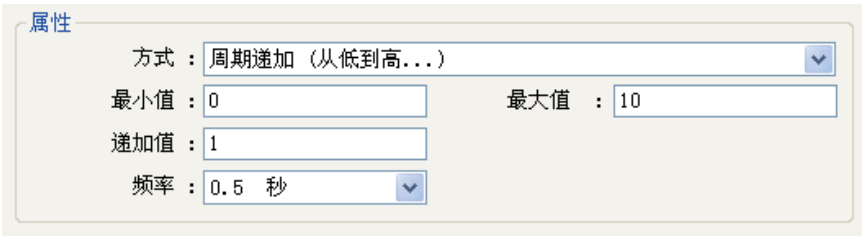
h. “周期循环 (自定范围)”

周期性循环功能。“多状态设置” 元件会使用[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。



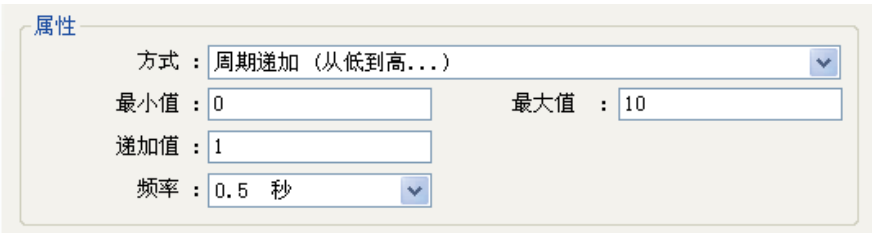
i. “周期递加(从低到高)”

步进功能。“多状态设置”元件会使用[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。



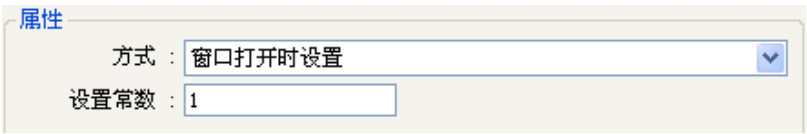
j. “周期递减(从高到低)”

步退功能。“多状态设置”元件会使用[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. “窗口打开时设置”

开启元件所在位置的窗口时，会将[设定常数]中的设定值自动写至指定的寄存器中。



l. “窗口关闭时设置”

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

m. “背光灯打开时设定值”

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

n. “背光灯关闭时设定值”

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

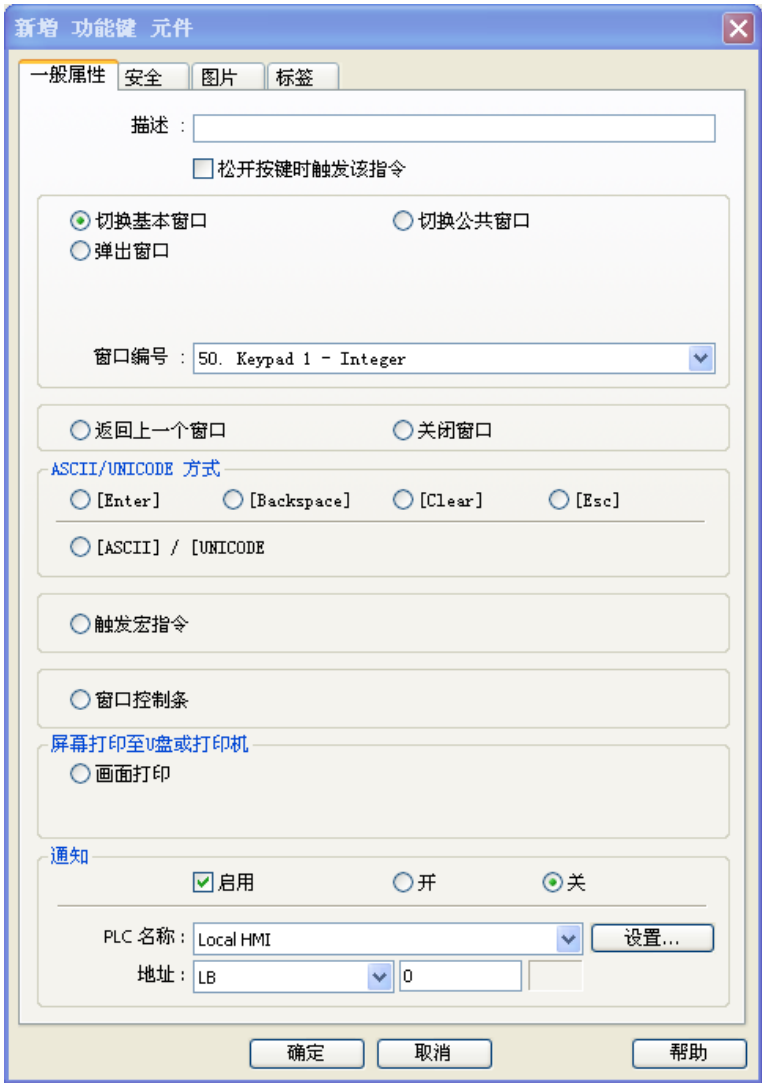
13.5 功能键元件 (function key)

概要

“功能键”元件提供窗口切换、调用其它窗口、关闭窗口等功能，也可用来设计键盘的按键。

设定

触控工具条上的“功能键”按钮后即会出现“功能键元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“功能键”元件，参考下图。



“功能键”元件提供下列几种操作方式：

[松开按键时触发该指令]

使用此选项表示必须在释放触控元件的动作后，选择的动作才会被执行。未选择此选项，则在触控元件后，将立刻执行选择的动作。

[切换基本窗口]

切换基本窗口。

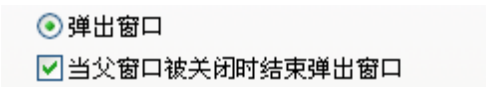
注意：在触摸屏的屏幕上请勿使用直接/间接窗口在开启其它窗口后，再以功能键进入同一窗口。

[切换公共窗口]

切换公用窗口，请参考“窗口”此章节的说明。

[弹出窗口]

调用其它窗口。此时调用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[当父窗口被关闭时结束弹出窗口]，参考下图。选择此属性则调用的窗口会在发生换页动作时自动消失，否则用户必须自行在被调用的窗口上设计[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”、“切换公用窗口”、“弹出窗口”时所使用的窗口。

[返回上一个窗口]

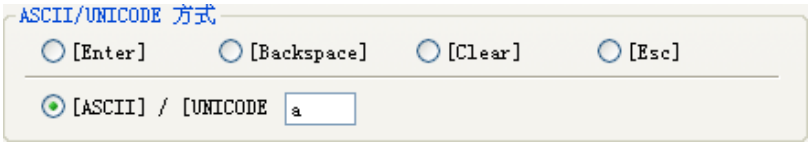
返回前一页基本窗口。例如当由“窗口10”切换到“窗口20”时，使用此功能可以再返回“窗口10”。此功能只对基本窗口有效。

[关闭窗口]

关闭在基本窗口上被调用的窗口，包括信息窗口。

ASCII/UNICODE方式项目

[ASCII模式]：被用来作为键盘的输入信号，主要用在“数值输入”与“字元输入”元件需要使用键盘来输入数字或文字的场合。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。

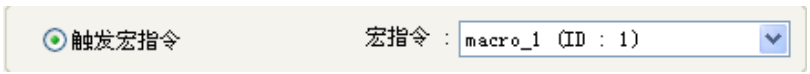
[Backspace]: 与键盘的后退删除(backspace)动作相同。

[Clear]: 清除目前对 “数值输入” 与 “文字输入” 元件已输入的资料。

[Esc]: 与使用[关闭窗口]功能相同, 皆可用来关闭弹出的键盘窗口。

[ASCII/UNICODE]: 设定对 “数值输入” 与 “文字输入” 元件的输入字符, 可选择0, 1, 2, …数字键或a, b, c, …等其它ASCII码。

[触发宏指令]

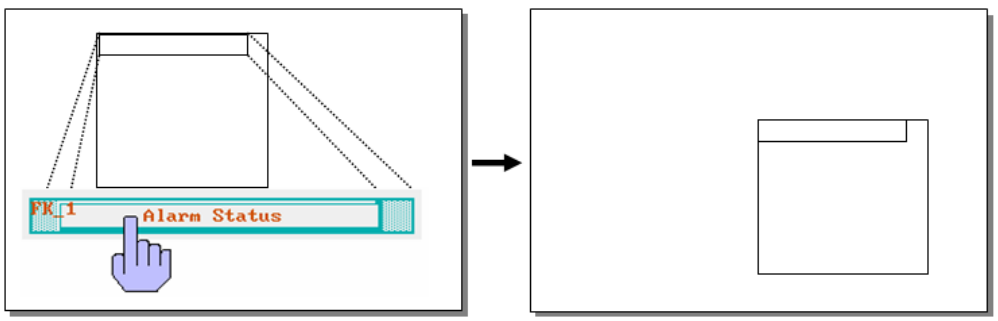


选择此项功能, 将执行指定的宏指令, 选择此项功能前需先建立宏指令。如何建立宏指令请参考相关的章节。

[窗口控制条]

定义为 “窗口控制条” 的功能键可以用来改变 “弹出窗口” 在屏幕上的位置。一个拥有这种功能键的弹出窗口, 可以在屏幕上移动弹出窗口, 方法是触控这个功能键然后再按第二个位置, 然后这个窗口就移动到了第二个位置。

注意: 此功能只在直接/间接窗口元件没有使用显示窗口控制条时才有用。

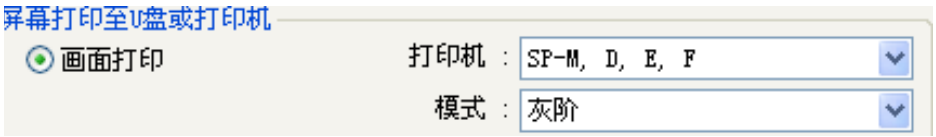


首先点击窗口控制条

接着在要移动的位置触控一下, 该窗口就移动到该位置了

[画面打印]

此项功能用来打印当前的画面。要选择此项功能前需先在“系统参数”中选择所使用的打印机型号。否则画面只能以图片格式打印到U盘或SD卡上。使用单色打印机时，勾选[灰阶效果]可以提升画面的辨识度，但也会影响文字的显示效果，因此如果是强调文字的打印效果，并不需使用灰阶功能。



通知项目

[启用]：使用此项设定，则在完成动作后可以连带设定此项目所指定的寄存器的状态，使用[开](ON)与[关](OFF)选择要设定的状态。

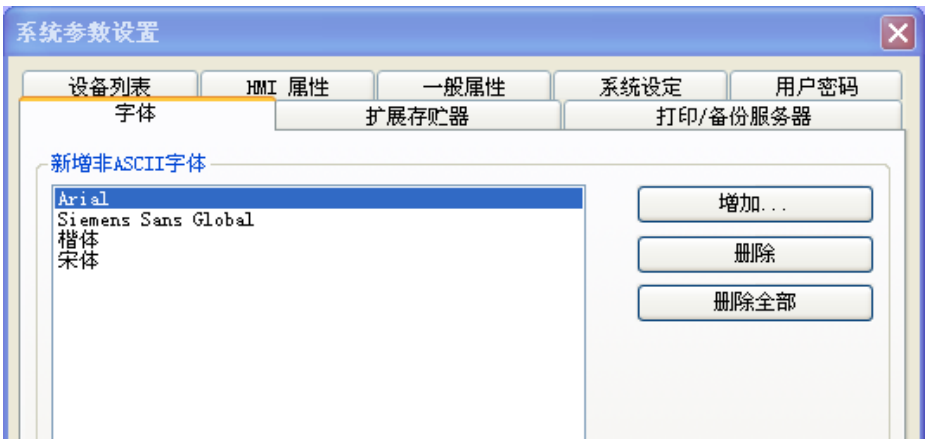
点击“设置”后选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制通知位项目，用户可以在“一般属性”页中设定地址。

非ASCII字元输入

下文说明如何在MT6000/8000系列触摸屏上输入与显示非ASCII文字（例如繁体中文、简体中文、日文、希腊文等），详细说明各个步骤。

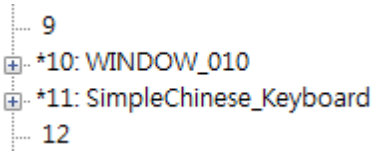
步骤1：设定非ASCII文字所使用的字体

首先规划非ASCII文字所使用的字体，并将这些字体加入系统参数设定中的“字体”设定页中。例如：规划日文使用“宋体”字体，简体中文使用“楷体”字体，韩文使用“Siemens Sans Global”字体，希腊字使用“Arial”字体，繁体中文使用“细明体”字体，参考下图：



步骤2：设计非ASCII文字输入键盘

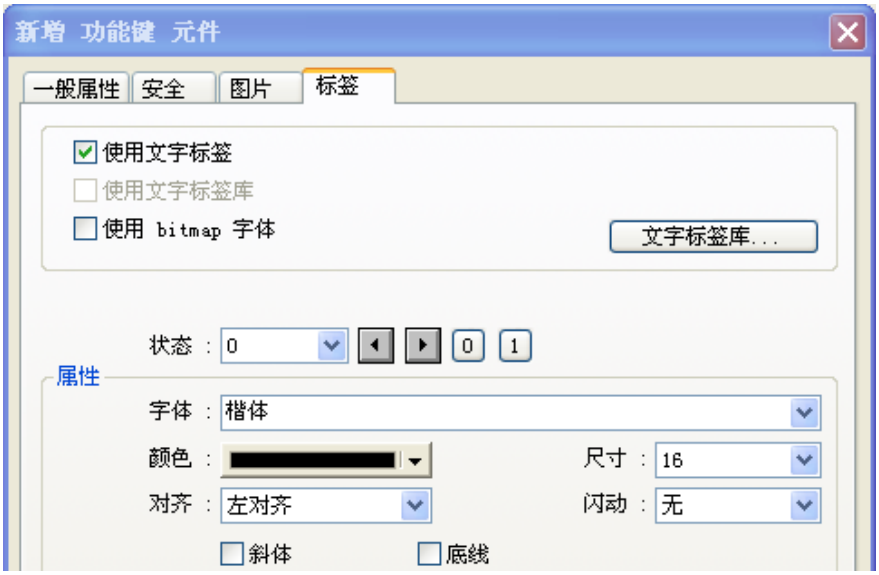
此时新增“窗口11”作为非ASCII文字的输入键盘，键盘的页面规划内容参考下图：



此时窗口上的元件皆为功能键元件，并依实际用途选择适当的属性。以输入“简”这个文字的功能键为例，需选择[ASCII]/[UNICODE]方式，并将输入内容设定为“简”，参考下图：

接着在“标签”页中，选择“使用文字标签”，并将标签内容设定为“简”，最终要选择的字体为“楷体”字体，必须与步骤一的设定内容相搭配，参考下图：

另外，在同一个键盘中作为非ASCII文字输入用途的功能键，所使用的字体必须相同，以简体中文键盘为例，此时作为非ASCII文字输入用途的功能键都选择使用“楷体”字体。



在完成所有按键的设定后，将窗口加入到系统参数设定的键盘列表中。

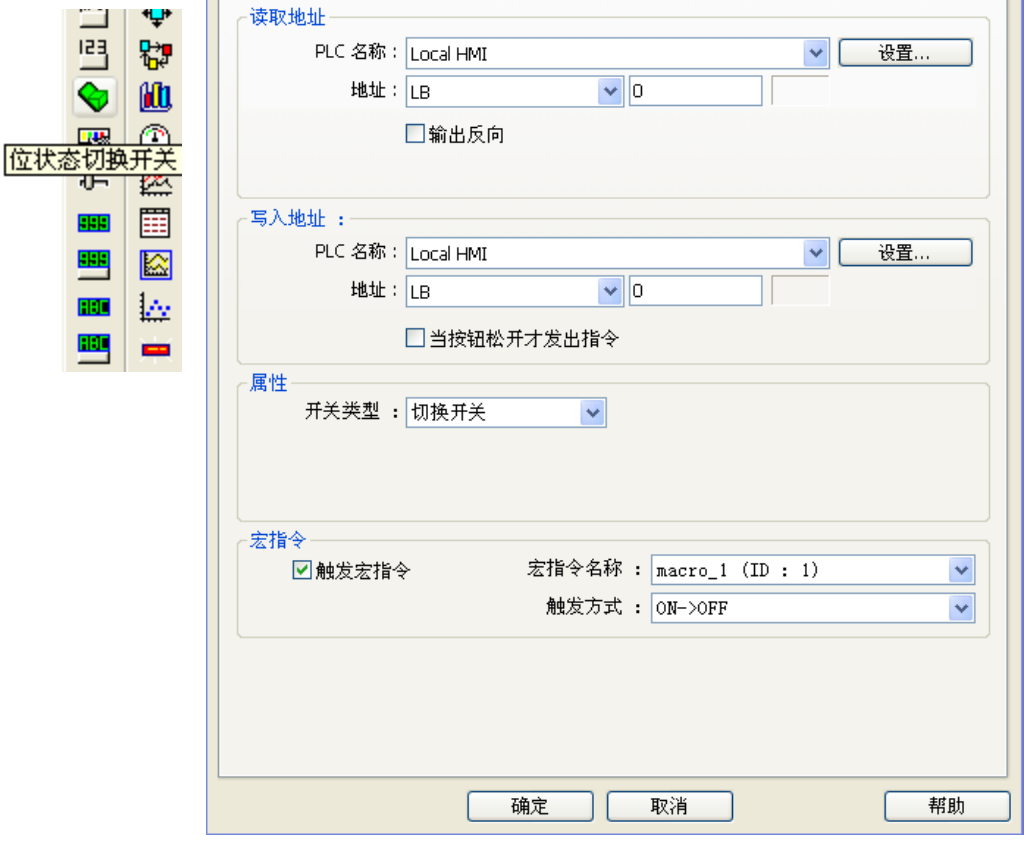
13.6 位状态切换开关元件 (toggle switch)

概要

“位状态切换开关”为“位状态指示灯”元件与“位状态设置”元件的组合。此元件除了可以用来显示寄存器的状态外，也可以利用这个元件在窗口上定义一个触控区域，触控此区域设定所指定寄存器的状态为ON或OFF。

设定

触控工具条上的“位状态切换开关”按钮后即会出现“新增位状态切换开关元件”属性对话窗，正确设定各项属性后触控确认键，即可新增一个位状态切换开关元件，参考下图。



读取地址项目

点击“设置”后选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制位状态切换开关的状态，用户也可在“一般属性”页中设定位地址。

输出反向

可以将读取的状态作反向显示，例如获得的状态为OFF，但是元件会显示ON的图形。

写入地址项目

点击“设置”后选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制位状态切换开关元件，用户可以在“一般属性”页中设定地址，该寄存器可以与“读取地址”所指定的寄存器相同或不同。

[当按钮松开才发出指令]

使用此项表示必须在释放触控元件的动作后，选择的动作才会被执行。若未选择此选项，则在触控元件后，将立刻执行选择的动作。

属性项目

选择元件的操作方式，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”元件的说明。

宏指令项目

操作位状态切换开关元件时，可以搭配执行宏指令(macro)，使用方式与“位态设置”元件相同，可参考“位状态设置”元件对此项功能的说明。

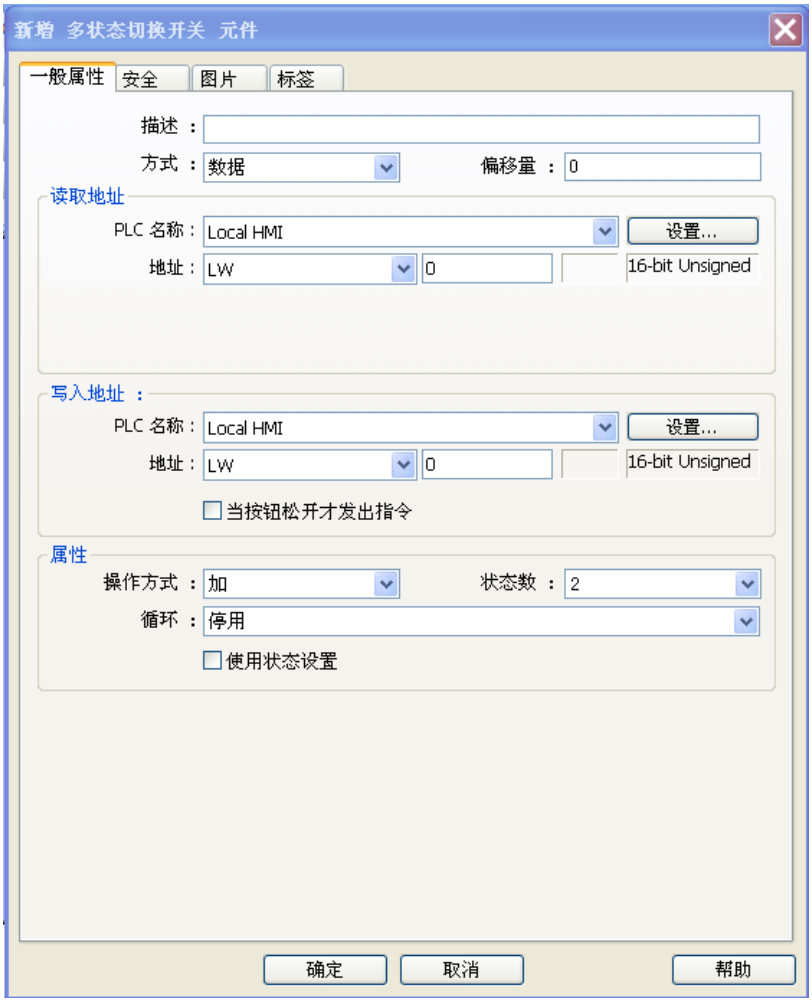
13.7 多状态切换开关元件 (multi-state switch)

概要

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

设定

触控工具条上的“多状态切换开关”按钮后即会出现“多状态切换开关元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“多状态切换开关”元件，参考下图。



方式

提供“数据”与“LSB”显示方式，可参考“多状态指示灯”元件的说明。

读取地址项目

点击“设置”后选择字寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来显示多状态切换开关的状态，用户也可在“一般属性”页中设定字地址。

写入地址项目

点击“设置”后选择字寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制多状态切换开关元件，用户也可在“一般属性”页中设定字地址，该寄存器可以与“读取地址”所指定的寄存器相同或不同。

[当按钮松开才发出指令]

使用此选项表示必须在释放触控元件的动作后，选择的动作才会被执行。若未勾选，则在触控元件后，将立即执行选择的动作。

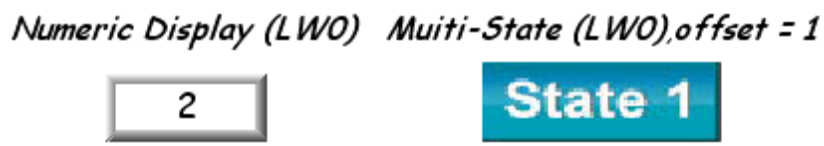
属性项目

选择元件的操作方式。

操作方式

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。



a. “加(JOG+)”

每触控一次元件，将对“写入地址项目”所指定的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，且[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)。

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-)”

每触控一次元件, 将对 “写入地址项目” 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。

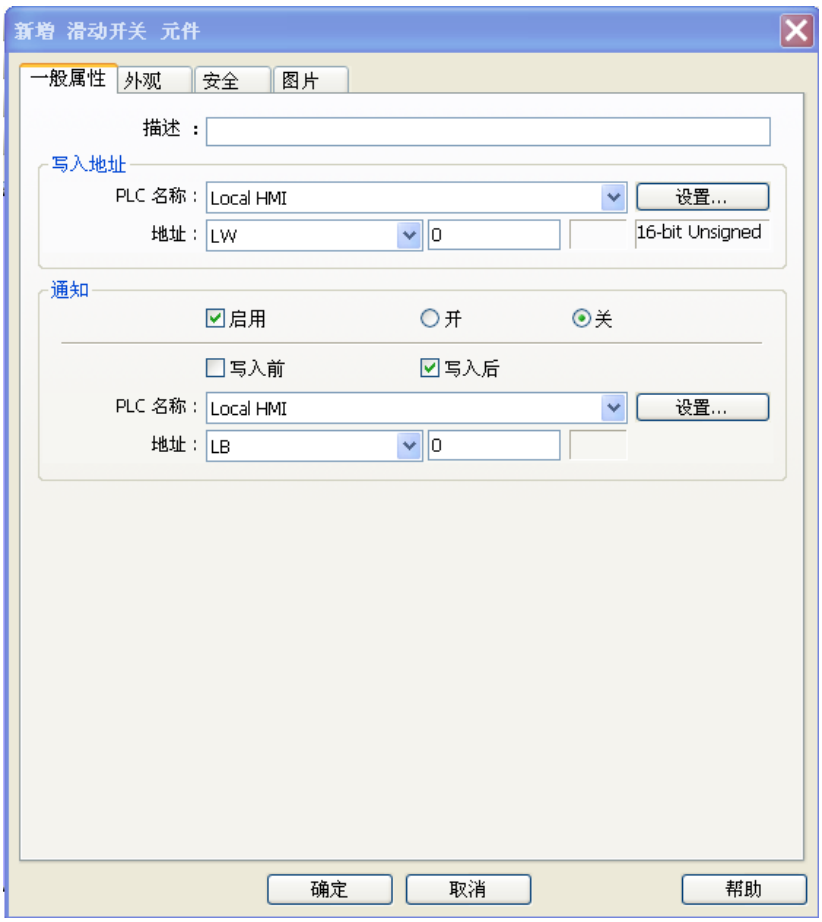
13.8 滑动开关元件 (slide object)

概要

滑动开关元件是用来建立一个滑动块区域或滑动滑轨来改变指定寄存器内的数值。

设定

使用工具条上的工作按钮后即会出现“滑动开关”元件属性对话窗，正确设定各项属性后触控确认键，即可新增一个“滑动开关”元件，参考下图。



[写入地址]

点击“设置”后选择字寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制滑动开关元件，用户也可在“一般属性”页中设定字地址。

[通知]

使用此项设定，则在完成动作之前/之后可以连带设定此项目所指定寄存器的状态，使用“开”与“关”选择要设定的状态。

点击“设置”后选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制通知位地址，用户也可在“一般属性”页中设定位地址。

[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。

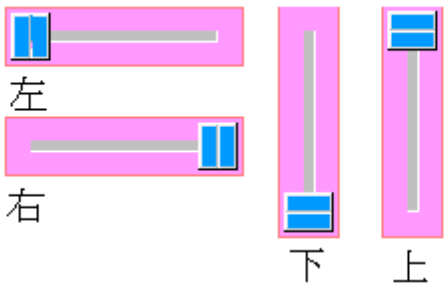
[写入后]

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]

滑动开关元件可以有四个方向来显示（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，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

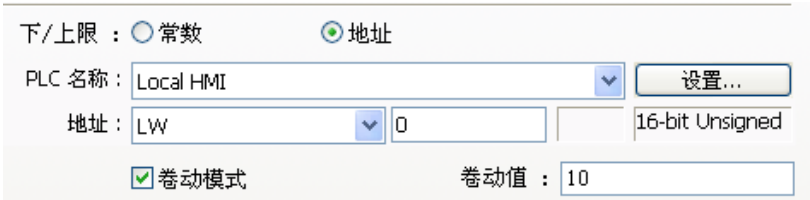
N=1，数值显示为每一次的显示都是依据1的刻度来变化，而且是任意的1的倍数

[下限]&[上限]

a) 常数：可直接设定字寄存器的上下限

b) 地址：由指定的地址来控制字寄存器的上下限

点击“设置”后选择字寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制上下限数值。



控制地址	下限	上限
16-bit 格式	地址+0	地址+1
32-bit 格式	地址+0	地址+2

[卷动模式]

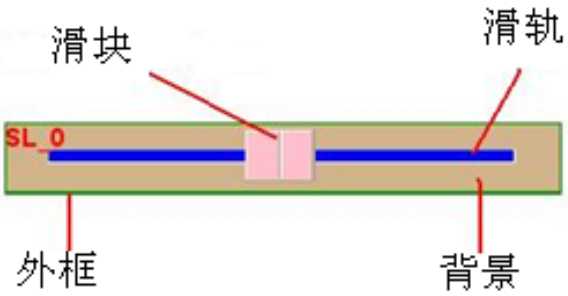
使滑动开关元件依此卷动值来递增或递减。

[滑块]

共有四种滑块样式可供选择，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。



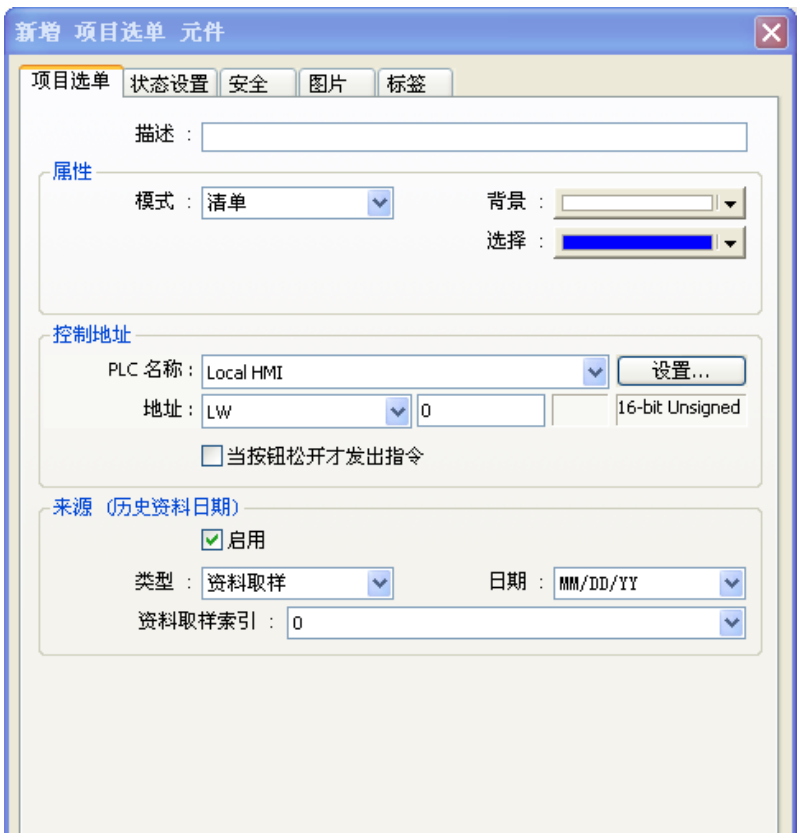
13.9 项目选单(Option List)

概要

项目选单元件可以显示多样项目成一个列表,用户可以借此去检视并选择,一旦用户选择了某一项目,相对应的项目数值将被写入到字寄存器。

设定

按下工具条上的“项目选单”按钮后即会出现“项目选单元件属性对话框”,正确设定各项属性后按下确认键,即可新增一个“项目选单”元件,参考下图:



[项目选单设定页]

属性

■ 模式：选择此元件的显示模式，清单或下拉式选单

■ 状态数：设定此元件的状态数，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到“控制地址”

■ 背景：选择元件的背景颜色

■ 选择：设定项目被选择时所标示的背景颜色

控制地址

选择字寄存器的“PLC名称”，“设备类型”，“地址”去控制元件的显示和系统状态数值写入。

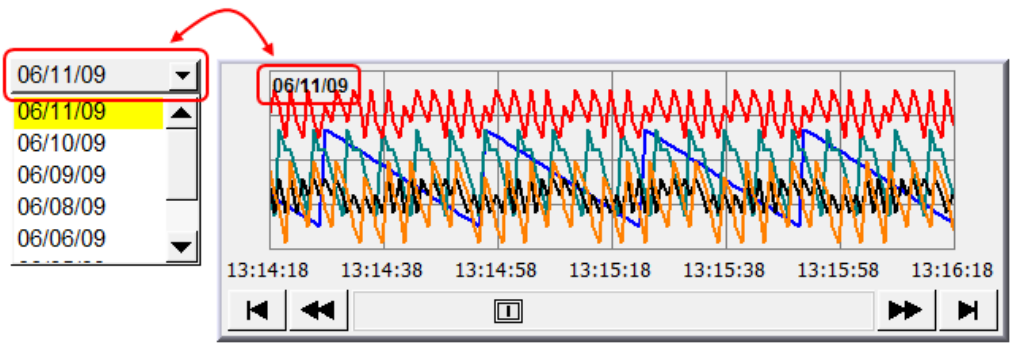
当按钮松开才发出指令

- 未勾选：当用户触控某项目时，系统将数值写入到“控制地址”
- 勾选：当用户触控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。

来源（历史资料日期）

项目选单元件能够搭配使用在历史模式下的显示元件，例如趋势图及历史数据显示元件，用来显示上述元件的历史文件并显示在屏幕上，显示方式如下图：



[类型]：若搭配“历史事件显示”，请选择“报警（事件）登录”，若搭配“历史趋势图”或“历史数据显示”，则选择“资料取样”。

[日期]：日期显示格式，有以下四种可供选择

a.MM/DD/YY

b.DD/MM/YY

c.DD.MM.YY

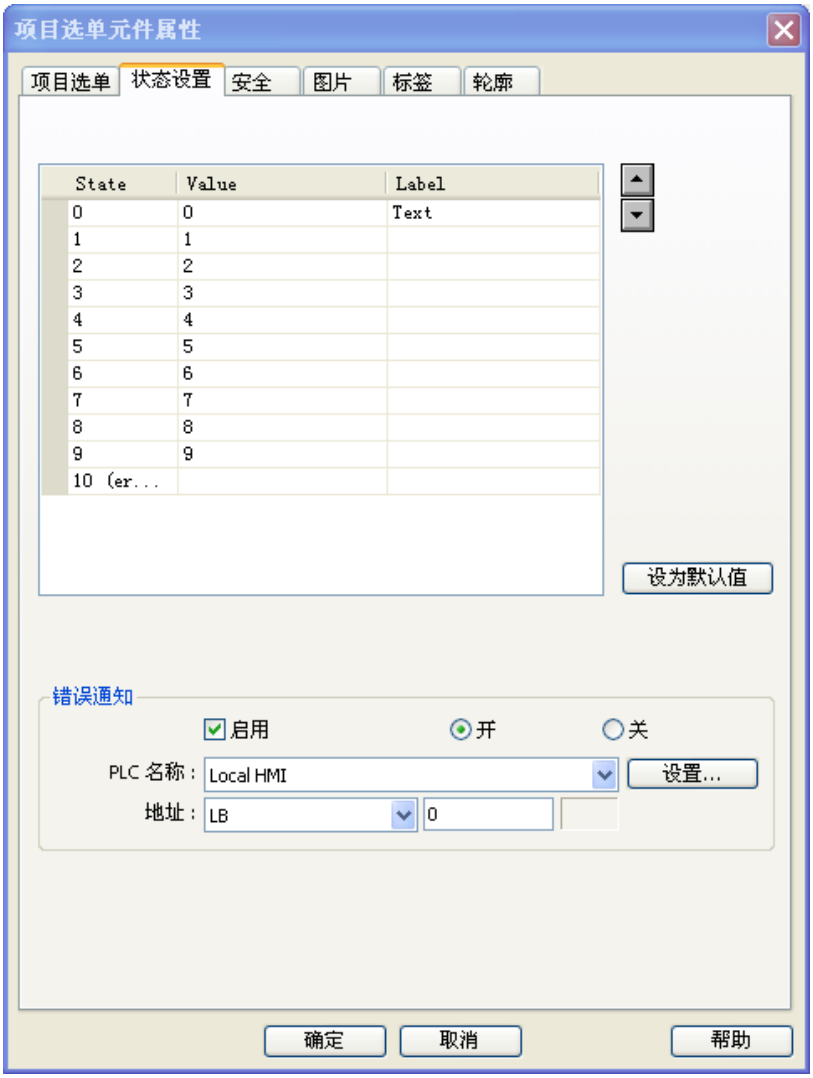
d.YY/MM/DD

[资料取样索引]: 当“类型”中选择“资料取样”时, 必须同时选择资料取样元件的编号, 一般来说选择与搭配趋势图的历史模式或历史数据元件相同即可。

备注:

1. 当启用来源(历史资料日期), 由于系统在执行期间将自动产生映射表(Mapping table), 故此处用户不需再填写。
2. 当项目选单进入错误状态时, 项目选单将显示“?”

[状态设置设定页]



状态设定

此设定页显示所有状态/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”

- 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上，此栏为只读栏。
- 数值：用户可为每个项目设定数值，但须遵守以下两个规范

  a.“读取”如果系统检测到“控制地址”的内容有任何改变，元件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错误通知位（如果有设定）

  b.“写入”当用户选择某项目，系统将数值写入至“控制地址”
- 文字：用户可为每个项目设定文字，项目选单元件将显示所有项目的文字在列表上供用户检视和选择。
- 错误状态：

  a. 如上图所示，当“状态数”设定为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

  b.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被显示错误状态的文字。

  c. 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。
- 设为预设值：将所有状态数值设为预设值。例如：设状态0为0，状态1为1…等等。
- 错误通知：

  启用：当错误发生时，系统将某个特定位设定为ON/OFF，此寄存器的通知位可以使用于触发某个动作来修正错误。

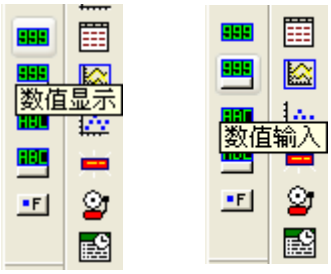
13.10 数值输入与数值显示元件 (numeric input and numeric display)

概要

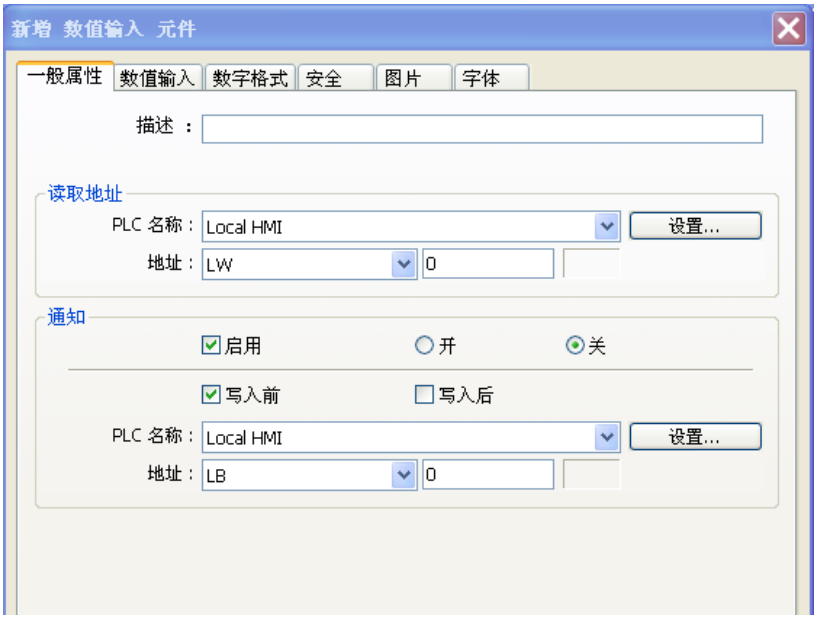
“数值输入”与“数值显示”元件皆可以用来显示所指定寄存器内的数值，其中“数值输入”元件还可以使用键盘输入数据，来改寄存器内的数据。

设定

触控工具条上的工作按钮后即会出现“数值输入”或“数值显示”元件属性对话窗，正确设定各项属性后触控确认键，即可新增一个“数值输入”或“数值显示”元件，参考下图。



“数值输入元件属性对话窗”与“数值显示元件属性对话窗”的差别在于“数值输入”元件增加了“通知”与键盘输入的设定项目。下图为“数值输入”元件的[一般属性]设定页。



读取地址项目

点击“设置”后选择字寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来显示数值，用户也可在“一般属性”页中设定字地址。

通知项目

在“数值输入”元件中使用此项设定，则在成功更改寄存器内的数值时(输入值必须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寄存器的状态，使用“开”(ON)与“关”(OFF)选择要设定的状态。

点击“设置”后选择位寄存器设备类型的“PLC名称”、“地址”、“设备类型”、“系统寄存器”、“索引寄存器”来控制通知位地址，用户也可在“一般属性”页中设定位地址。

[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之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。

[模式]

触控：用户通过触控元件来启动输入程序

位控制：用户通过控制指定位寄存器来启动及结束输入程序，当位地址被开启时启动输入，位地址被关闭时结束输入。但是若原先已有一个输入元件正在输入，此时就算位地址被开启，还是要等前一个元件输入结束后才能进入输入程序。

[允许输入位地址]

指定允许输入位地址的“PLC名称”，“设备地址”，“位地址”。允许输入位地址被用在控制启动和结束输入程序。

[输入次序]

用户可通过设定输入次序及输入次序群组达到多个输入元件连续输入的需求，当完成目前输入动作后（触控“ENT”），系统将自动跳至下一个输入元件继续输入。

启用：用户勾选“启用”并设定输入次序启动此项功能，另外可勾选“群组”来设定输入次序群组。

a) 输入次序：范围 1~511

b) 输入次序群组：范围 1~15

c) 若没有勾选“群组”时输入次序群组为1

寻找下一个输入元件的准则：

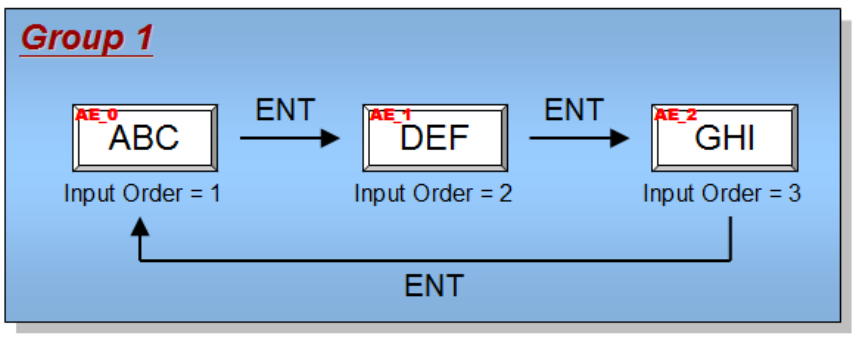
a) 系统只寻找一个输入次序群组中的输入元件

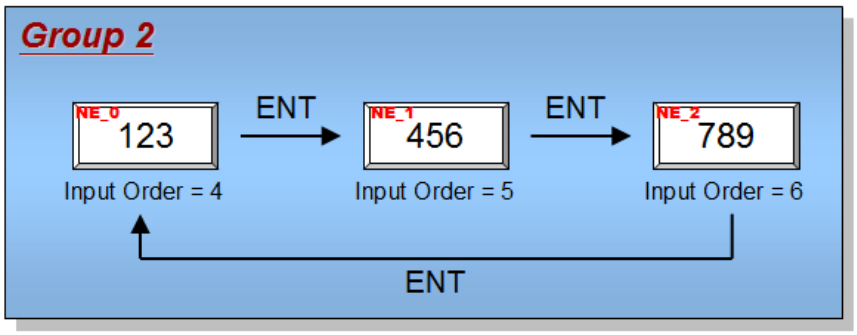
b) 越小的输入次序数值代表输入顺序排在越前面，反之则越后面。

c) 若两个输入元件有相同的输入次序群组及输入次序，则较下层的输入元件将先输入。

当[模式]设定为“触控”时：

以下图为例，当用户完成输入“AE_2”后，下一个输入元件为“AE_0”而非“NE_0”，因为“AE_2”与“NE_0”分属不同的输入群组。





当[模式]设定为“位控制”时:

a) 系统将自动启用输入次序功能

b) 不须设定输入次序群组,所有此类的输入元件将编在同一类别群组中

[键盘项目]

勾选“使用弹出键盘”:指定键盘窗口以及键盘弹出的位置,当启动输入时,系统将在指定位置弹出键盘窗口,并于结束输入时关闭。

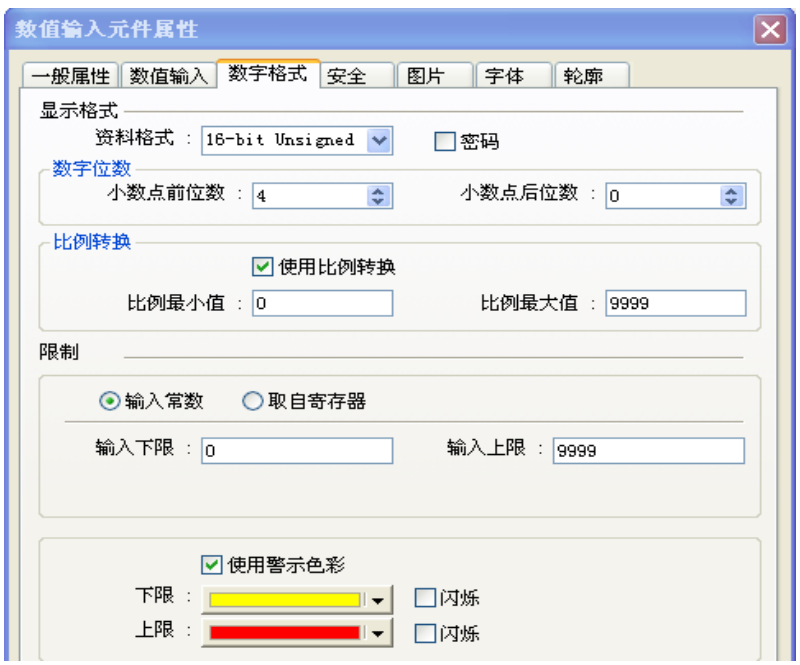
不勾选“使用弹出键盘”:启动输入时系统将不会弹出键盘窗口,用户必须以下列方式进行输入动作

a) 自行在窗口中设定键盘

b) 使用外接键盘

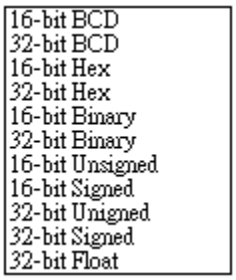
注意:当[模式]设定为“位控制”时,系统将自动禁用“键盘”中的“使用弹出键盘”。

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



[资料格式]

选择寄存器内数据的格式,可选择的项目如下图。



[密码]

数值显示时将使用“*”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 = [比例最小值] + (A – [输入最小值]) * ratio

其中ratio = ([比例最大值] – [比例最小值]) / ([输入上限] – [输入下限])

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 10 + (15 – 0) * (50 – 10) / (20 – 0) = 40,元件上将显示40。

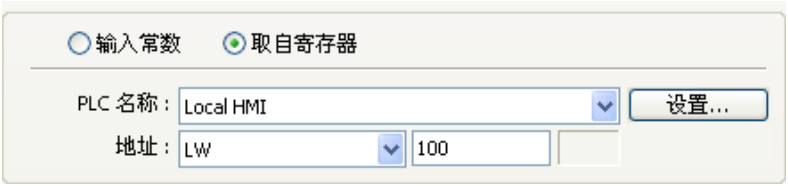
限制项目

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。若输入值不在输入上、下限定义的范围内，将无法更改寄存器内的数值。

[取自寄存器]



选择输入数值的上、下限来自指定的寄存器。此时寄存器的资料长度与元件所显示的数据类型有关。举例来说，上图的输入上、下限来自[LW100]，此时输入上下限的存放地址如下：

a. 若显示的数据型态为“16–bit”，如16–bit unsigned，则

[LW100] 下限存放地址(16–bit)

[LW100 + 1] 上限存放地址(16–bit)

b. 若显示的数据型态为“32–bit”，如32–bit unsigned，则

[LW100] 下限存放地址(32–bit)

[LW100 + 2] 上限存放地址(32–bit)

c. 若显示的数据的型态为“32–bit float”，则

[LW100] 下限存放地址(32–bit float)

[LW100 + 2] 上限存放地址(32–bit float)

[下限]

当寄存器内的数值小于下限值时，元件会使用此项颜色显示数值。

[上限]

当寄存器内的数值大于上限值时,元件会使用此项颜色显示数值。

[闪烁]

当寄存器内的数值小于下限值或大于上限值时,元件会使用闪烁的效果加以警示。

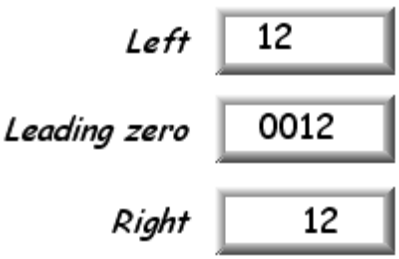
下图为“数值输入”与“数值显示”元件的[字体]设定页,用来设定数值显示时所使用的字体、尺寸与颜色,另外也包括数字的对齐方式。

[颜色]

当数值在上下限的范围内时,使用此项颜色显示。

[对齐]

提供三种数字对齐方式:“左对齐”(left)、“前导零”(leading zero)、“右对齐”(right),使用不同对齐方式的表现行为可参考下图。



[尺寸]

设定字型大小

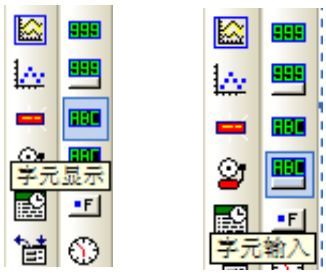
13.11 字元输入与字元显示元件 (ASCII input and ASCII display)

概要

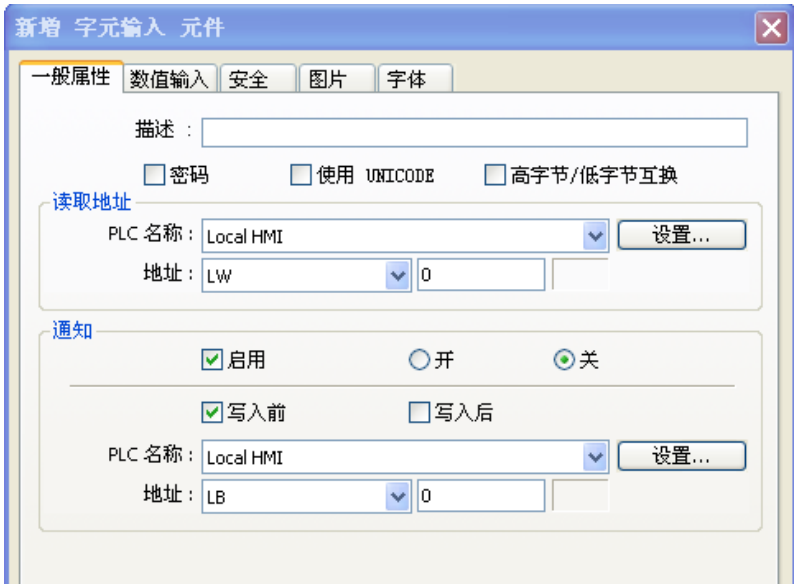
“字元输入”与“字元显示”元件使用ASCII的编码方式显示所指定寄存器中的数据，“字元输入”元件可以使用键盘来输入数值，来更改寄存器内的数据。

设定

触控工具条上的工作按钮后即可使用“字元输入”或“字元显示”元件属性对话窗，正确设定各项属性后触控确认键，即可新增一个“字元输入”或“字元显示”元件，参考下图。



“字元输入”与“字元显示”元件属性对话窗的差别，在于“字元显示”元件增加“通知”与键盘输入功能的设定项目。下图为“字元输入”元件的[一般属性]设定页。



密码项目

字元显示时将使用“*”代替所有字元。

使用UNICODE项目

勾选“使用UNICODE”选项,可显示UNICODE格式的资料,否则系统会显示ASCII格式,此功能可代替功能键的“ASCII/UNICODE”。

高字节/低字节互换项目

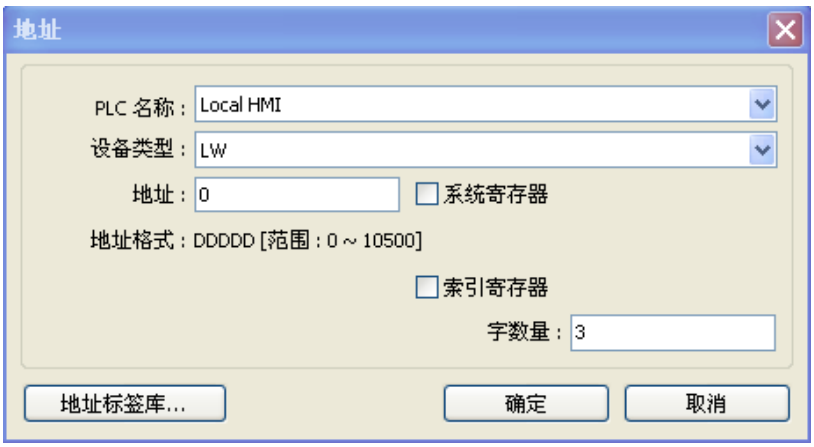
正常情况下,ASCII code的显示顺序为“低字节”,“高字节”,但是在某些情况下需互换顺序时,可勾选此功能。

读取地址项目

点击“设置”后选择字寄存器设备类型的“PLC名称”,“地址”,“设备类型”,“系统寄存器”,“索引寄存器”来显示字元,用户也可以在“一般属性”页中设定地址。

字数量:点击“设置”后选择字元最多可显示的资料长度,单位为word,可选择的最小值为1,因为每个ASCII字元程度为一个byte,所以每次最少会显示两个字元。

如下范例,元件可显示3*2=6个字元



通知项目

使用此项设定,则在完成动作之前/之后可以连带设定此项目所指定寄存器的状态,使用“开”和“关”来选择要设定的状态。

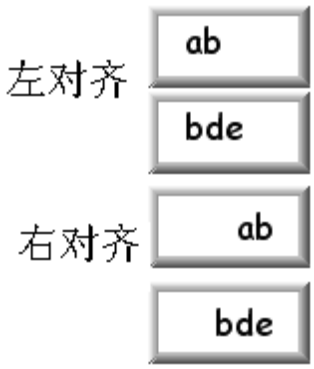
点击“设置”后选择位寄存器设备类型的“PLC名称”,“地址”,“设备类型”,“系统寄存器”,“索引寄存器”来控制通知位项目,用户也可在“一般属性”页中设定位地址。

写入前/写入后:在写入动作前/后来设定所指定寄存器的状态。

下图为“字元输入”与“字元显示”元件的“字型”设定页,用来设定字元显示时所使用的字体,大小与颜色,另外也包括字元对齐的方式。

[对齐]

提供两种文字对齐方式:“左对齐”(left)、“右对齐”(right),使用不同对齐方式的表现行为可参考下图。



[尺寸]

设定字体大小

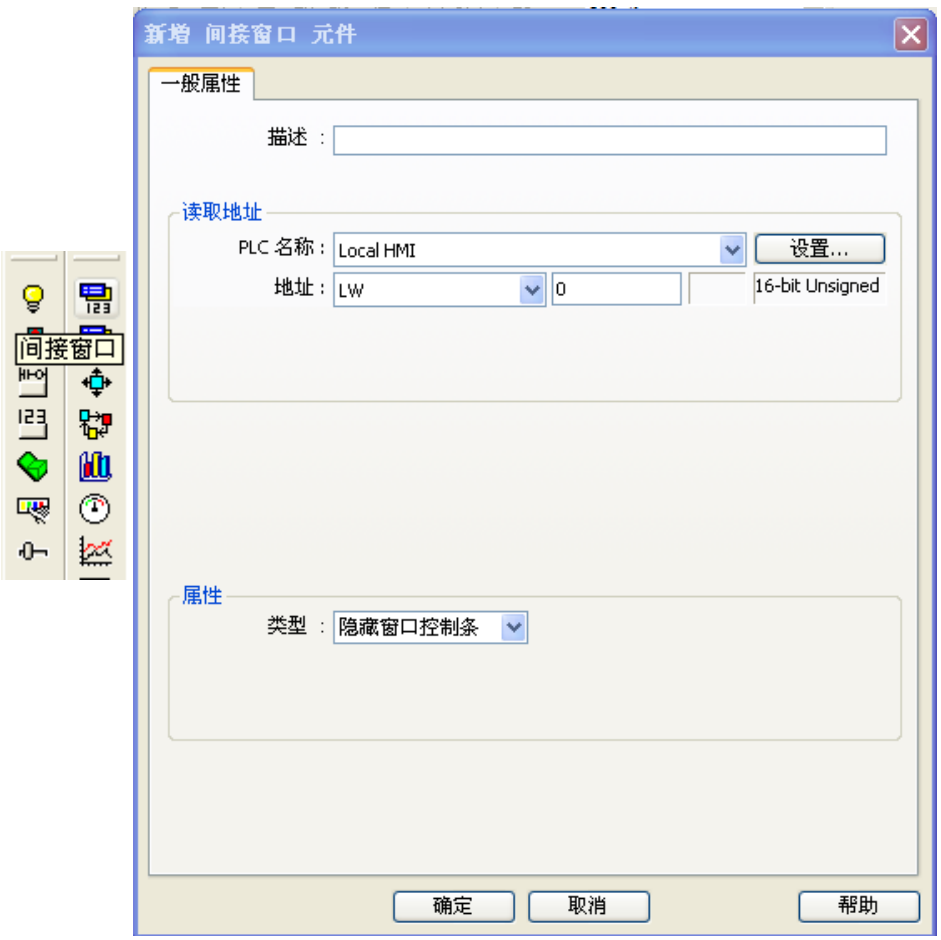
13.12 间接窗口元件 (indirect window)

概要

“间接窗口”元件可以在窗口上定义一个显示区域，并在完成相关寄存器的设定后，当此寄存器内的数据与已存在的窗口号码相同时，将在此显示区域内显示此窗口的内容。所显示窗口的长度与高度不会大于此显示区域。要关闭此窗口也可以使用此寄存器，只需将寄存器的值设定为0即可。

设定

触控工具条上的间接窗口按钮后即会出现“间接窗口元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个间接窗口元件，参考下图。



读取地址项目

点击“设置”后选择字寄存器设备类型的“PLC名称”，“地址”，“设备类型”，“系统寄存器”，“索引寄存器”来控制窗口弹出，用户也可以在“一般属性”页中设定字地址。

属性项目

类型: 设定弹出窗口的样式, 支持两种样式, “隐藏窗口控制条” 和 “显示窗口控制条”。

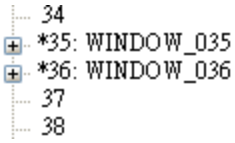
a. “隐藏窗口控制条”

弹出的子窗口不包含窗口控制条, 但是它的窗口位置被固定在预设位置无法拖动。

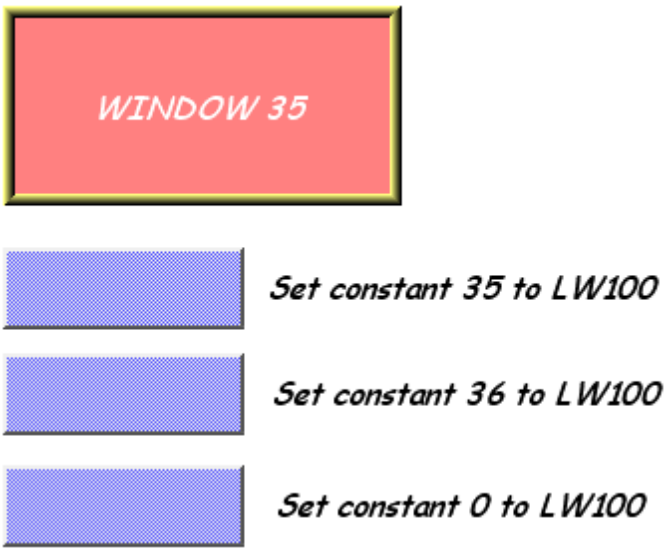
b. “显示窗口控制条”

弹出的子窗口包含窗口控制条, 他的窗口位置可通过控制条任意拖动。

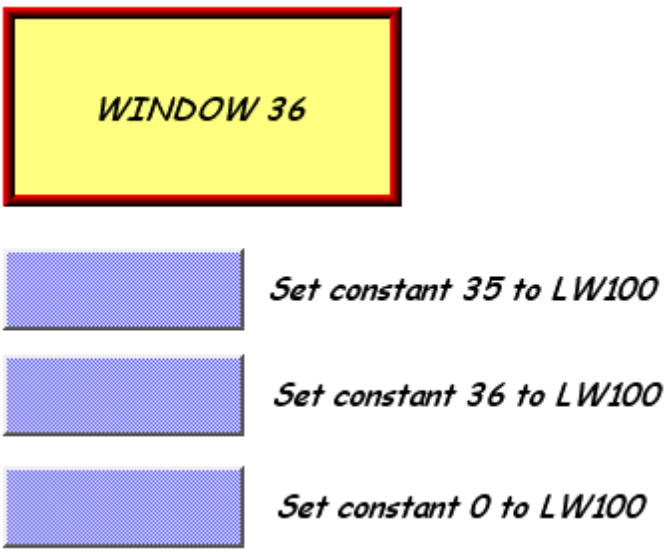
现在使用一个简单的例子说明间接窗口的使用方式,下图为间接窗口元件的设定内容, 此时使用[LW100]用来指定要出现的窗口号码, 并预先建立 “窗口35” 与 “窗口36”。



可以使用“多状态设定”元件SW_0, 将[LW100]设定为35, 此时窗口显示的画面如下。



如果继续使用“多状态设定”元件SW_1, 将[LW100]设定为36, 将可以关闭“窗口35”, 并且弹跳出“窗口36”, 参考下图。



要关闭“窗口35”或“窗口36”除了可以使用“多状态设定”元件SW_2将[LW100]设定为0之外, 另一种方式是在“窗口35”与“窗口36”上设计一个“功能键”元件, 并使用[关闭窗口]模式, 在触控此元件后即可关闭这些窗口。

注意: 在触摸屏的屏幕上请勿使用功能键/直接窗口在开启其它窗口后,再以间接窗口进入同一窗口。

13.13 直接窗口元件 (direct window)

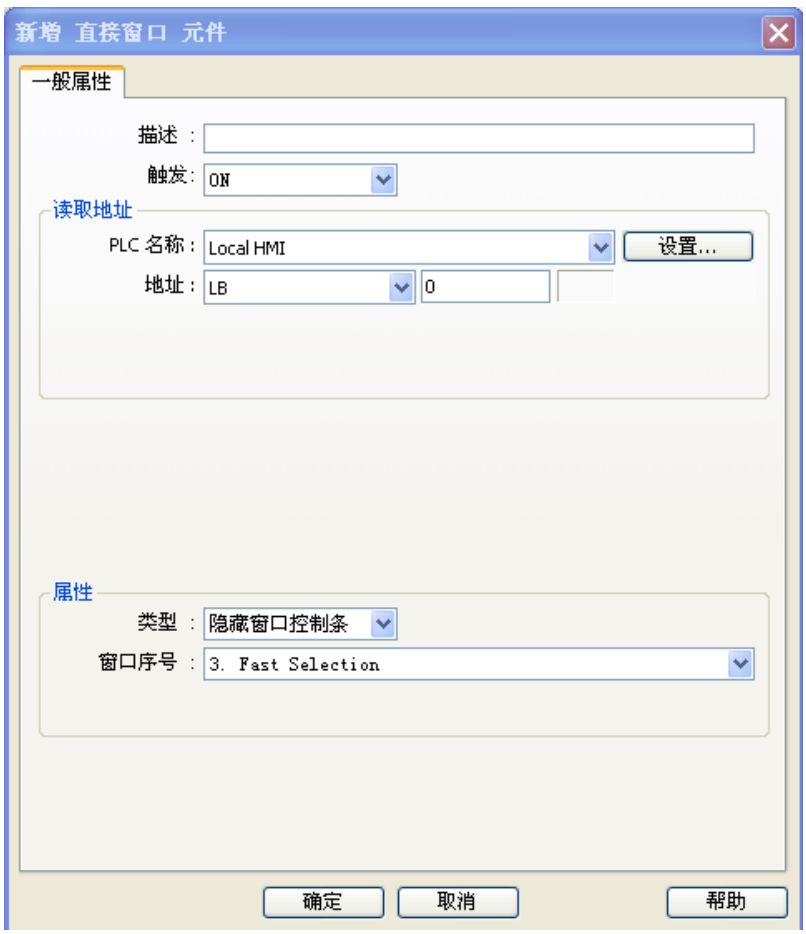
概要

“直接窗口”元件可以在窗口上定义一个显示区域，当所指定寄存器的状态由OFF变为ON时，将在此显示区域内显示指定窗口的内容。所显示窗口的长度与高度不会大于此显示区域。要关闭此时所显示的窗口，只需将用来触发窗口出现的寄存器的状态由ON变为OFF即可。

“直接窗口”与“间接窗口”元件的差别在于，“直接窗口”已经事先设定好要显示的窗口，系统运作时，将利用所指定寄存器的状态决定显示或关闭此窗口。

设定

触控工具条上的“直接窗口”按钮后即会出现“直接窗口元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“直接窗口”元件，参考下图。



读取地址项目

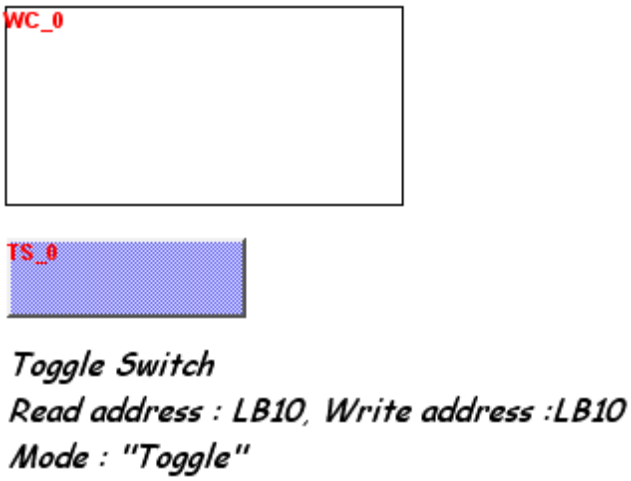
点击“设置”后选择位寄存器设备类型的“PLC名称”，“地址”，“设备类型”，“系统寄存器”，“索引寄存器”来控制窗口弹出，用户也可在“一般属性”页中设定位地址。

属性项目

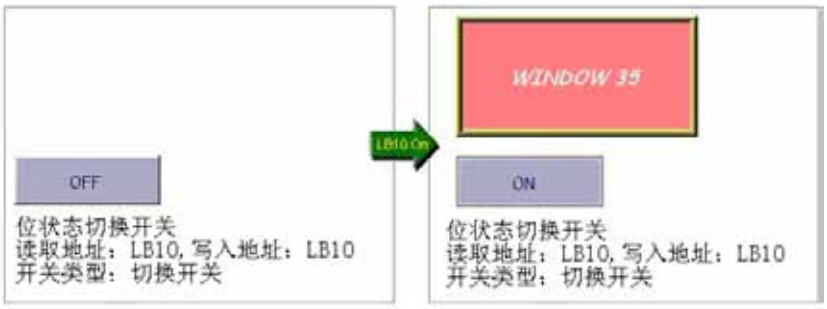
类型：设定弹出窗口样式，支持两种样式，“隐藏窗口控制条”和“显示窗口控制条”

窗口序号：设定要弹出的窗口序号。

现在使用一个简单的例子说明“直接窗口”的使用方式，下图为“直接窗口”元件的设定内容，此时使用LB10来决定是否显示“窗口35”。



当LB10状态为ON时，“窗口35”将出现；当LB10状态为OFF时，“窗口35”将消失。参考下图。



注意：在触摸屏的屏幕上请勿使用功能键/间接窗口在开启其它窗口后，再以直接窗口进入同一窗口。

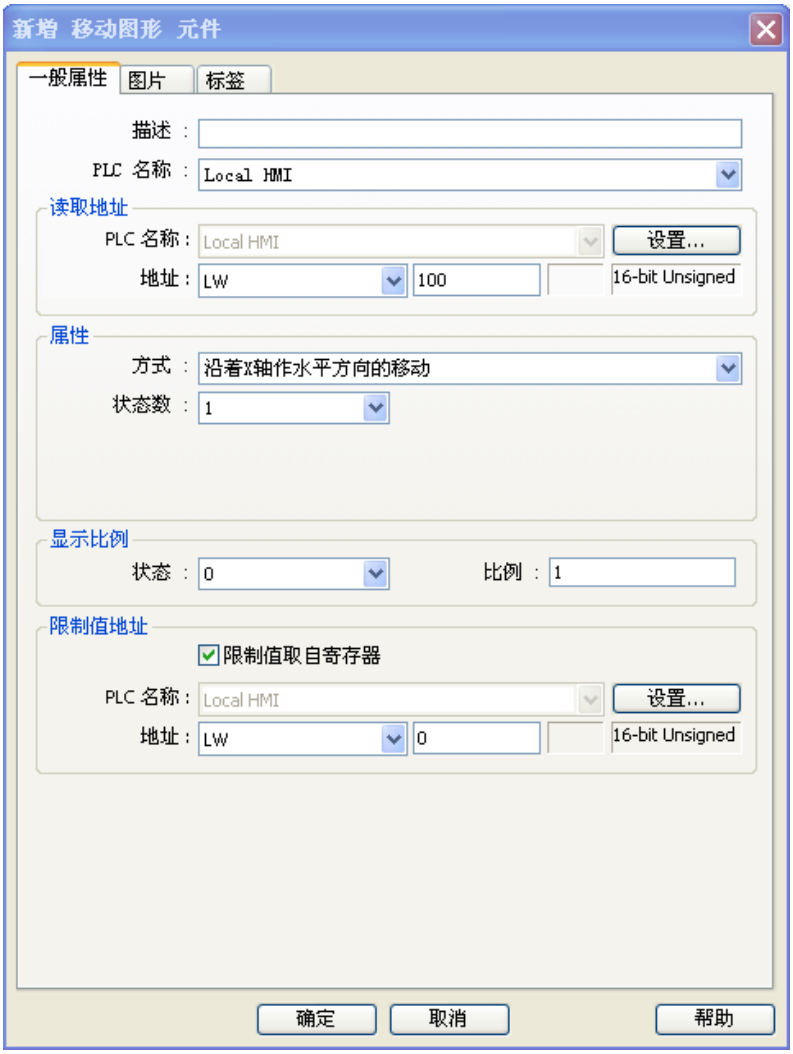
13.14 移动图形元件 (moving shape)

概要

“移动图形”元件可定义元件的状态和移动距离，元件会利用三个连续的寄存器内的数据，来决定元件的状态与元件的移动距离。第一个寄存器为控制元件的状态，第二个寄存器为控制元件的水平位置移动距离(X)，第三个寄存器为控制元件的垂直位置移动距离(Y)。

设定

触控工具条上的“移动图形”按钮后即会出现“移动图形元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“移动图形”元件，参考下图



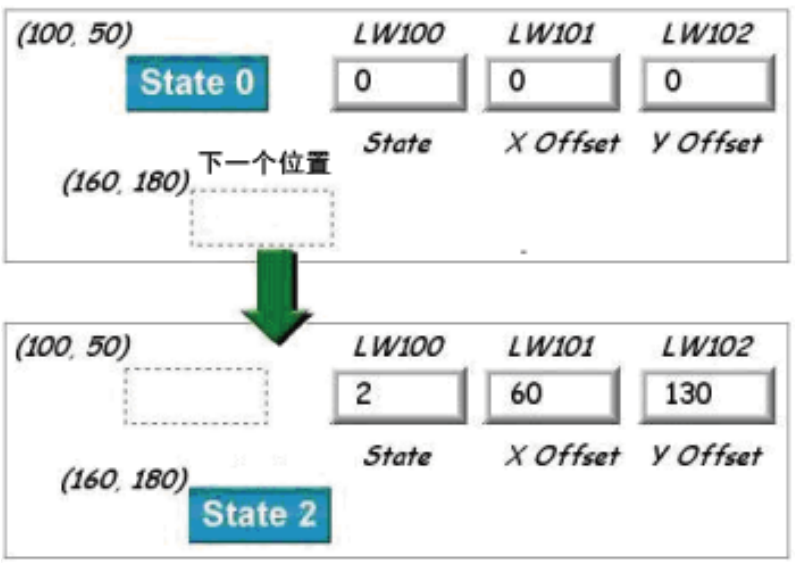
读取地址项目

点击“设置”后选择字寄存器设备类型的“PLC名称”，“地址”，“设备类型”，“系统寄存器”，“索引寄存器”来控制移动图形状态和移动位置，用户也可在“一般属性”页中设定地址，下列表显示在不同格式时，需使用的控制地址：

变量型态	元件状态读取地址	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，若寄存器为[LW100]，且资料格式使用“16–bit Unsigned”，则[LW100]存放元件的状态，[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，元件的地址为[LW100]且起始地址为(100, 50)，假使现在要移动元件至(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。

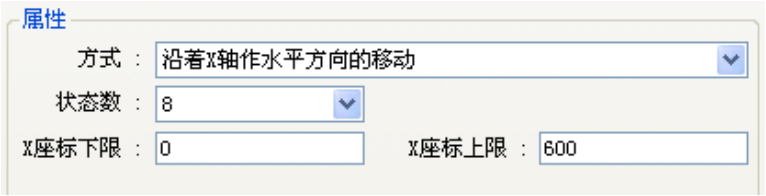


属性项目

选择元件的移动方式、移动的范围。

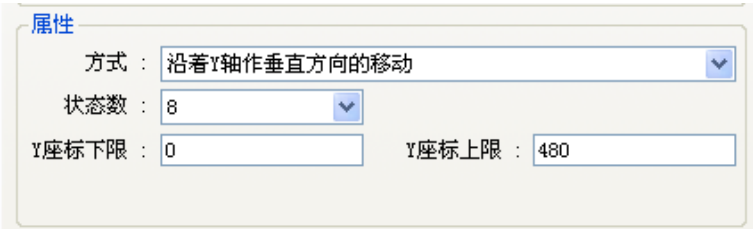
a. 沿着X轴作水平方向的移动

只允许元件沿着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。



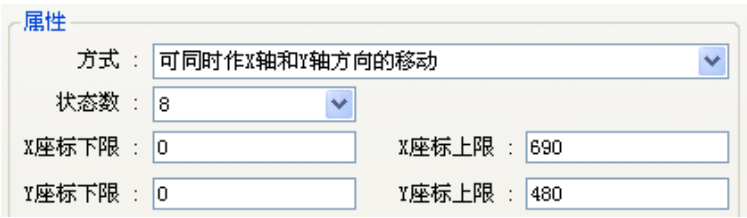
b. 沿着Y轴作垂直方向的移动

只允许元件沿着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

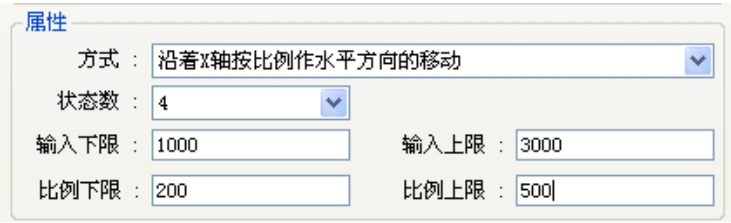
允许元件沿着X轴与Y轴移动。移动范围由[X轴坐标下限]、[X轴坐标上限]与[Y轴坐标下限]、[Y轴坐标上限]来决定。



d. 沿着X轴、按比例作水平方向的移动

只允许元件沿着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])



例如元件只允许作200~500大小的位移，但寄存器数据的大小范围为1000~3000，此时可以将[输入下限]设定为1000，[输入上限]设定为3000，[比例下限]设定为200，[比例上限]设定为500，元件即会在要求的范围内移动。

e. 沿着Y轴、按比例作垂直方向的移动

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

f. 沿着X轴、按反比例作水平方向的移动

此项功能与“沿着X轴、按比例作水平方向的移动”相同，但移动方向相反。

g. 沿着Y轴、按反比例作垂直方向的移动

此项功能与“沿着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

元件各个状态的图形在显示时，可以分开设定缩放比例，参考下图。

限制值地址项目

元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

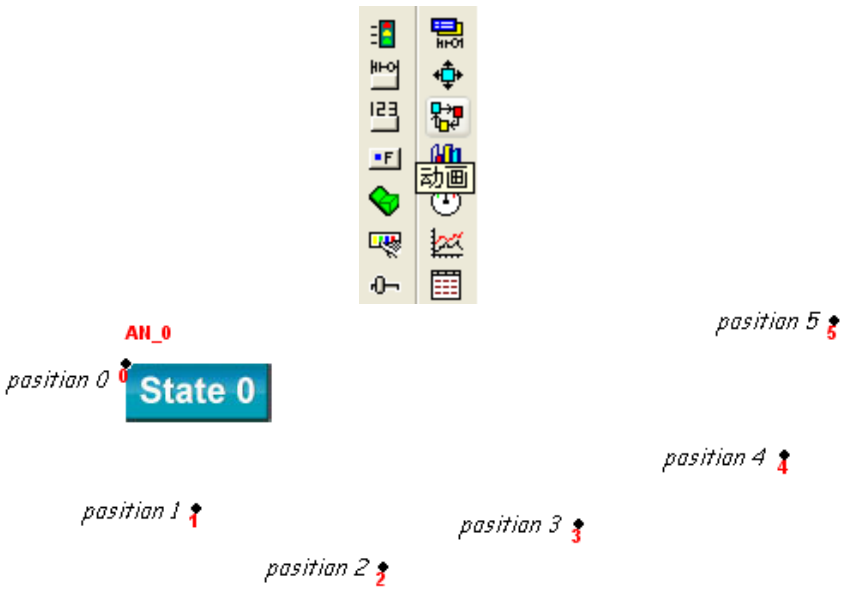
13.15 动画元件 (animation)

概要

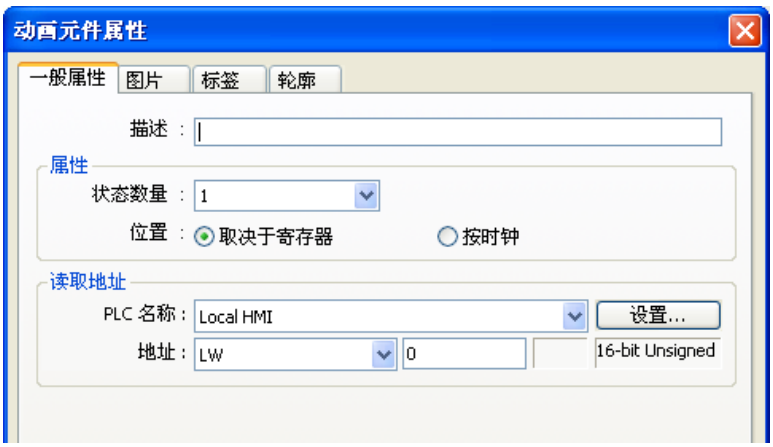
用户可以预先定义“动画”元件的移动轨迹，并利用更改寄存器内的数据，控制元件的状态与元件在移动轨迹上的位置。系统将使用两个连续寄存器内的数据来控制动画元件，第一个寄存器用来控制元件的状态，第二个用来控制元件的位置。

设定

触控工具条上的“动画”按钮后，在适当位置触控鼠标的左键，即可定义一个新的移动位置，定义完成全部的移动位置后，触控鼠标的右键即可完成移动轨迹的规划，新增一个新的“动画”元件，参考下图。



要更改元件的属性，可以使用鼠标左键双击(double click)元件所在位置，利用出现的“动画元件属性对话窗”，即可更改元件的各项属性，下图为“动画”元件一般属性设定页。



属性项目

状态数量：设定元件的状态数目。

位置：可以选择“取自寄存器”或者“按时钟”两种方式

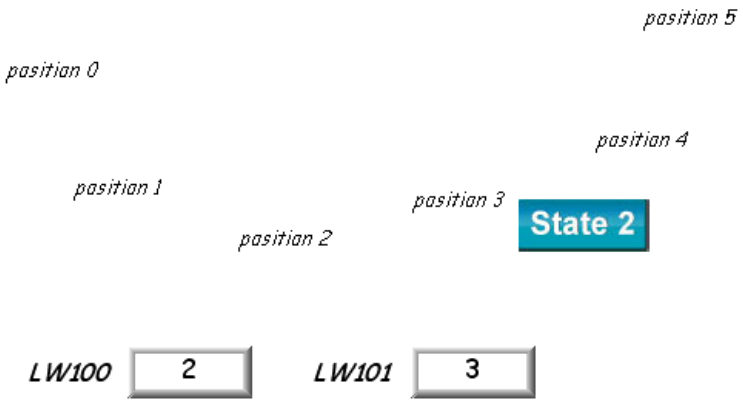
a) 选择“取决于寄存器器”，则元件的状态与位置由寄存器中的数据决定。

读取地址：

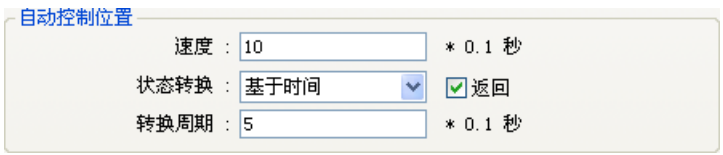
如果元件的状态与位置由寄存器中的数据决定，必须正确设定元件状态与位置的读取地址。读取地址整理如下表。表中的address表示寄存器的地址值，例如寄存器为[LW100]时，address等于100。

变量型态	元件状态读取地址	元件位置读取地址
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

举例来说，若寄存器为[LW100]，且变量类型使用“16–bit Unsigned”，则[LW100]存放元件的状态，[LW101]存放元件的显示位置。以下图为例，[LW100] = 2，[LW101] = 3，所以元件显示状态2，并出现在位置3。



b) 若元件不选择“取决于寄存器”而选择“按时钟”的变化，则元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：位置改变的速度，单位为0.1秒。例如设定为10，则元件每隔1秒钟变换一个位置。

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

position 0–> position 1–>position 2–>position 3–> position 0–> position 1–> position 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position 0，再重复原来位置改变方式，移动位置整理顺序如下：

position 0–> position 1–>position 2–>position 3–> position 2–> position 1–> position 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。

下图的对话窗用来设定“动画”元件的外型大小，也可利用鼠标双击“动画”元件，即可出现。

向量图尺寸项目

用来设定元件所显示图形的大小。

轨迹项目

用来设定移动轨迹上各点的位置。

13.16 棒图元件 (bar graph)

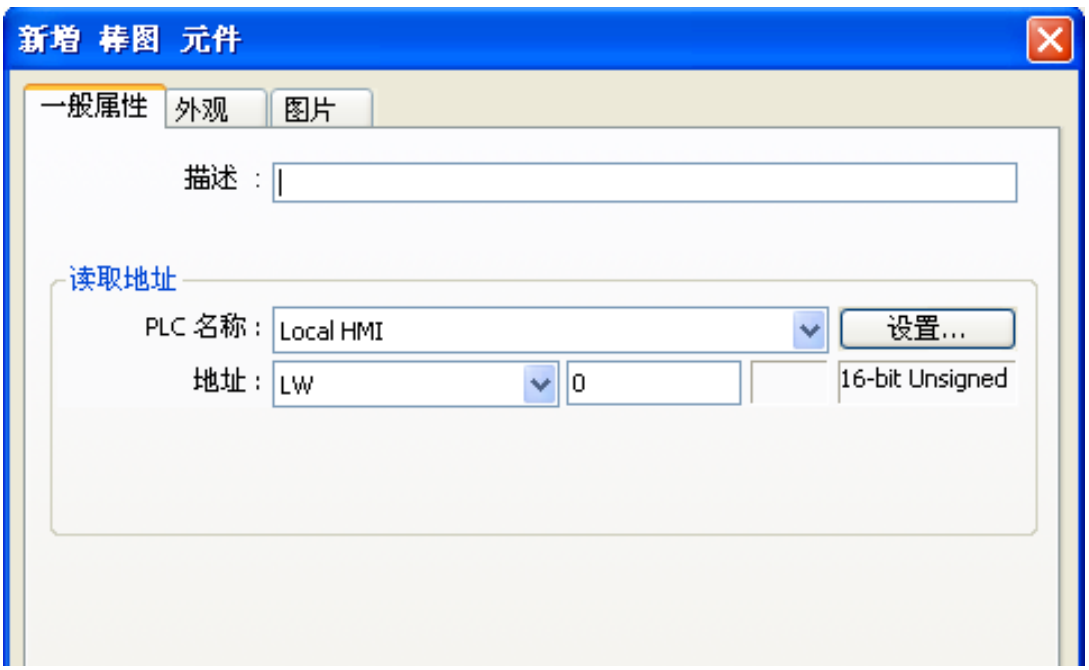
概要

“棒图”元件使用百分比例与棒图的方式，显示寄存器中的数据。

设定

触控工具条上的“棒图”按钮后即会出现“棒图元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“棒图”元件，参考下图。

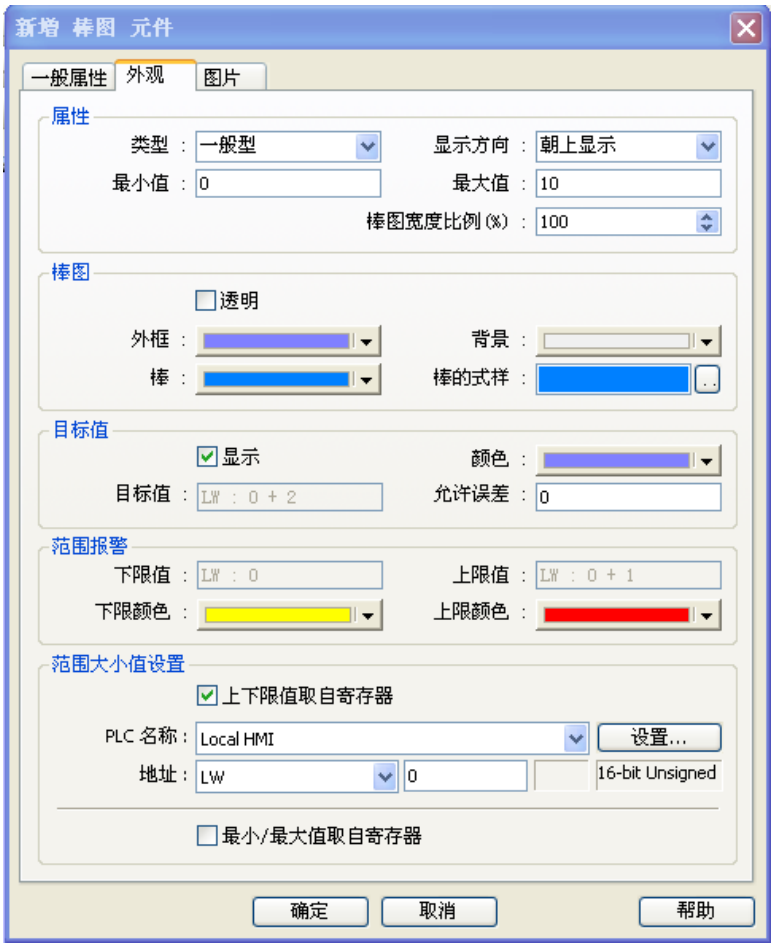
下图为“棒图”元件的一般属性设定页。



读取地址项目

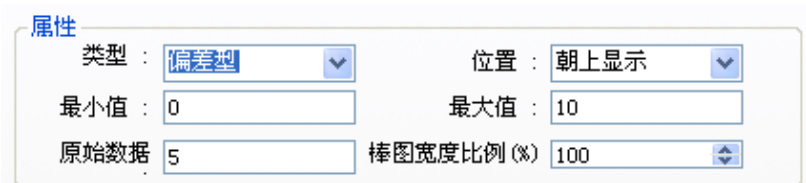
点击“设置”后选择字寄存器设备类型的“PLC名称”，“地址”，“设备类型”，“系统寄存器”，“索引寄存器”来作为棒图显示的数据依据，用户也可在“一般属性”页中设定字地址。

下图为“棒图”元件的外型设定页。



属性项目

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。



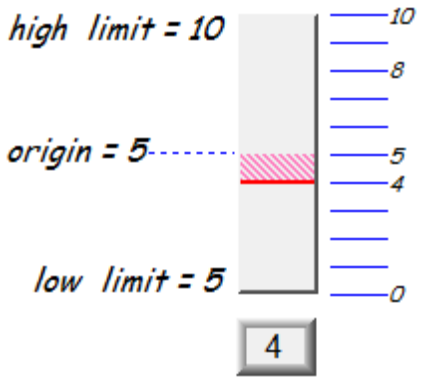
显示方向：用来选择棒图的显示方向，可以选择“朝上显示”、“朝下显示”、“朝右显示”、“朝左显示”。

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：

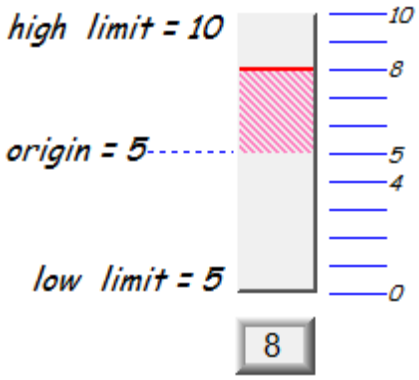
棒图填充区域百分比 = (寄存器数据 – [最小值]) / ([最大值] – [最小值]) *100%

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。

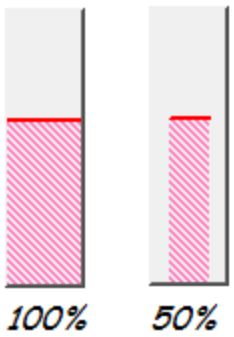
当读取值为4：



当读取值为8：

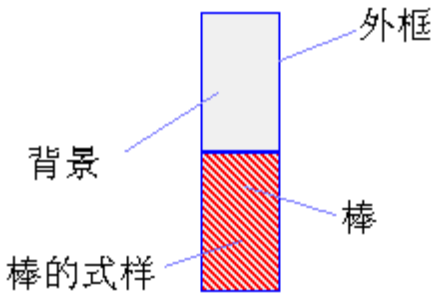


棒图宽度比例(%)：设定棒图的显示宽度与元件宽度间的百分比率，下图为使用两种不同设定值的显示情形。100%和50%。



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样式与颜色，参考下图。

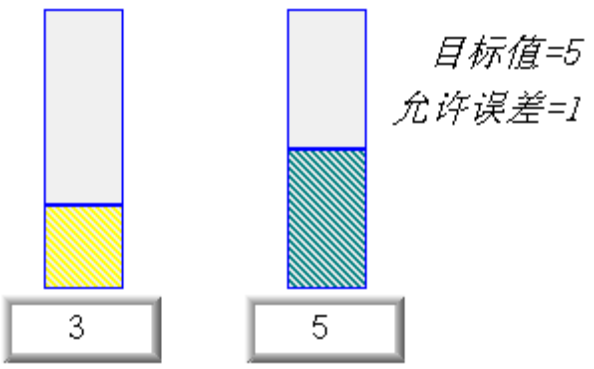


目标值项目

当寄存器内的数据符合下列条件时，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”。



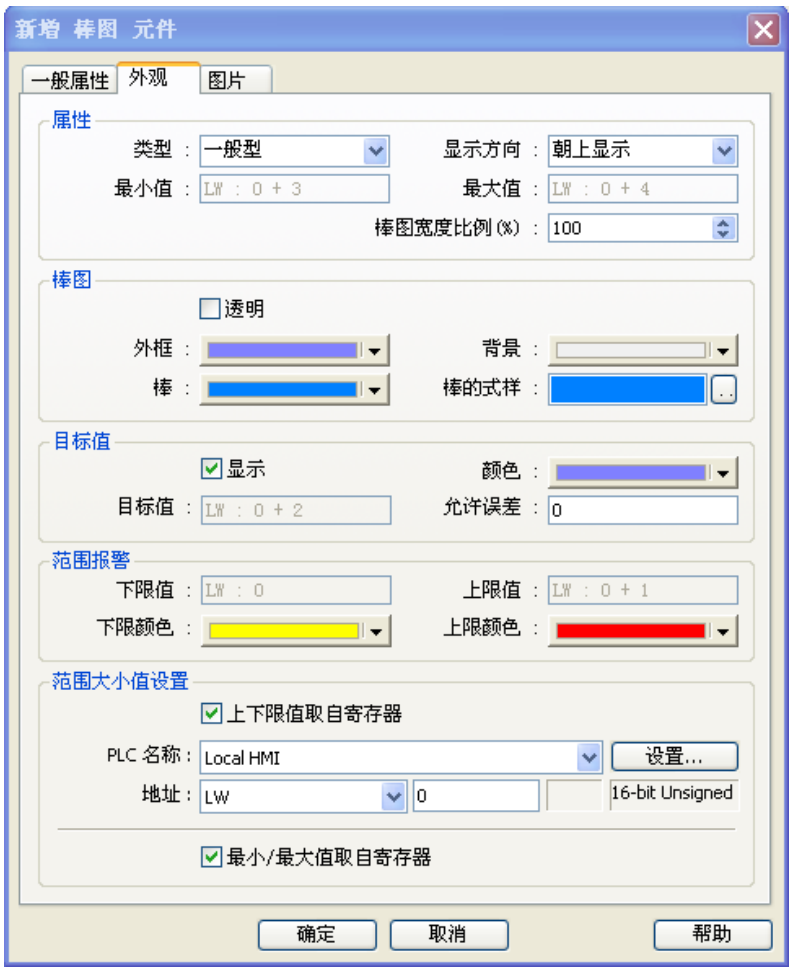
范围报警项目

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。

范围大小值设定项目

当选择[上下限值取自寄存器]，“范围报警项目”中所使用的[下限值]、[上限值]与“目标值项目”中的[目标值]皆读取自指定的寄存器，参考下图。



下表整理了当使用范围取自寄存器，上下限与目标值的读取地址，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	下限值读取地址	上限值读取地址	目标值读取地址
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

13.17 表针元件 (meter display)

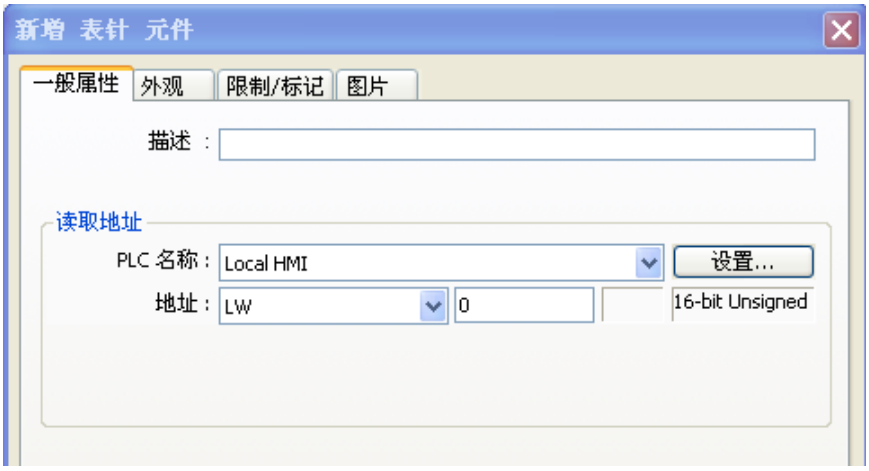
概要

“表针”元件使用仪表的方式，指示目前寄存器中的数据。

设定

触控工具条上的“表针”按钮后即会出现“表针元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“表针”元件，参考下图。

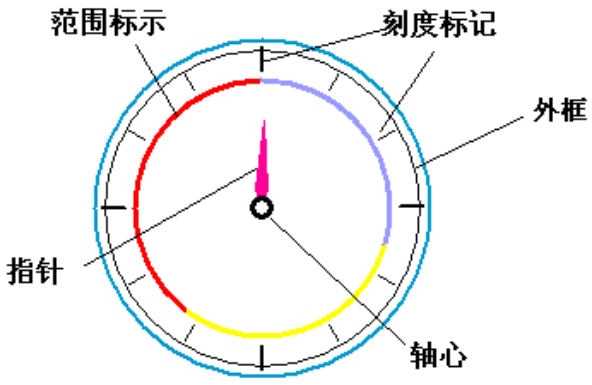
下面说明“表针元件属性对话窗”中各设定页的内容。



读取地址项目

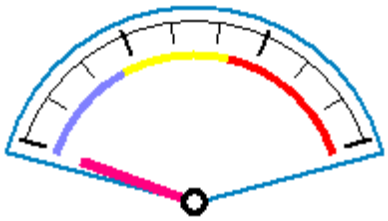
点击“设置”后选择字寄存器设备类型的“PLC名称”，“地址”，“设备类型”，“系统寄存器”，“索引寄存器”来作为表针显示的数据依据，用户也可在“一般属性”页中设定字地址。

上图的设定对话窗用来设定“表针”元件的外观，各部分的名称可以参考下图。

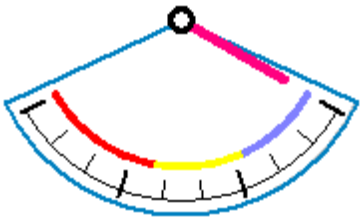


角度项目

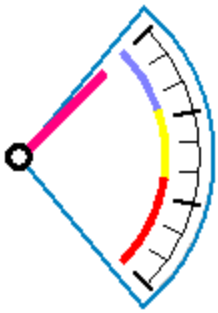
用来设定元件的起始角度与结束角度，角度可设定范围皆为0~360度。不同设定值对元件外观的影响，可参考下面的几种不同的设定。



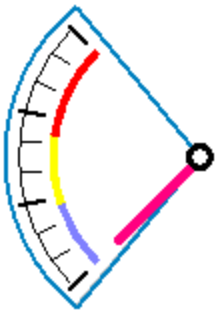
[起始角度] = 290, [结束角度] = 70



[起始角度] = 120, [结束角度] = 240



[起始角度] = 40, [结束角度] = 140

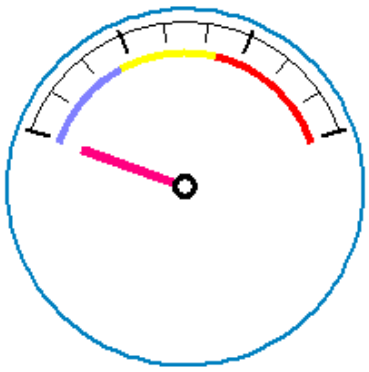


[起始角度] = 225, [结束角度] = 315

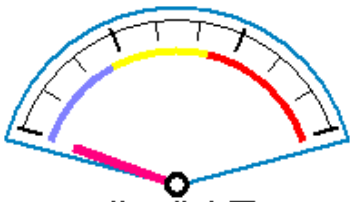
背景项目

设定元件的背景与圆周的颜色。

全圆：选择“全圆”，元件将显示全圆，反之则只显示角度定义的范围，参考下图。



使用全圆



使用非全圆

透明：选择“透明”，元件将不显示背景与圆周的颜色。

刻度标记项目

设定元件的刻度数目与颜色。

指针项目

设定元件指针的样式、长度、宽度与颜色。

轴心项目

设定元件指针轴心的样式、半径与颜色。

下图为限制/标记页面设定对话窗

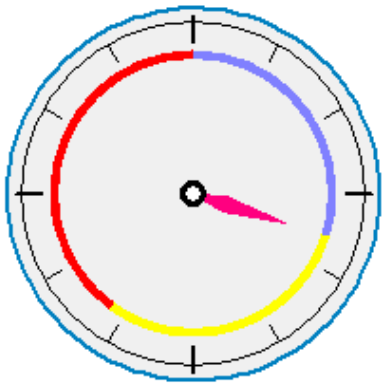
数值项目

设定元件所要显示的数值范围。“表针”元件会利用[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

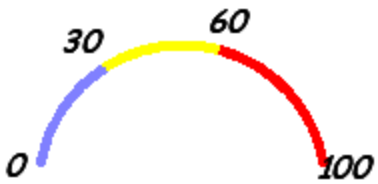
指针将指示在108度的位置，参考下图。



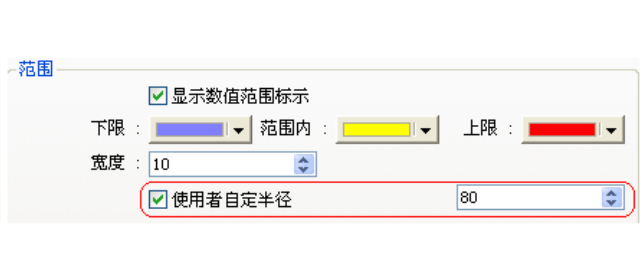
范围项目

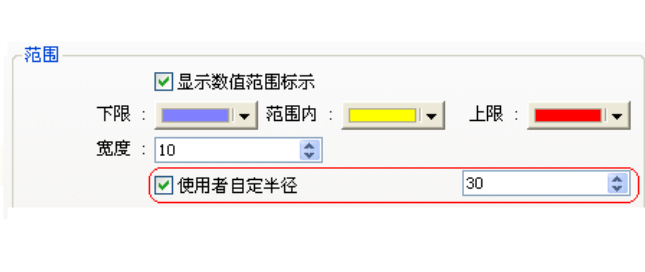
设定高、低限值，高、低限标志的显示颜色与宽度。

显示不同数值范围内的颜色：选择是否显示高低限标志，下图则为利用上面的设定值所显示的高低限标志。

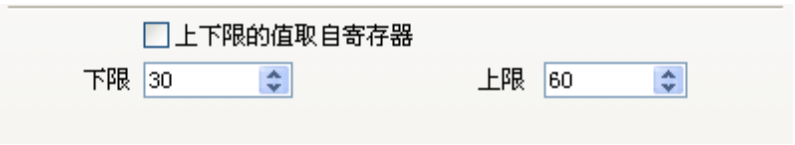


使用者自定半径：

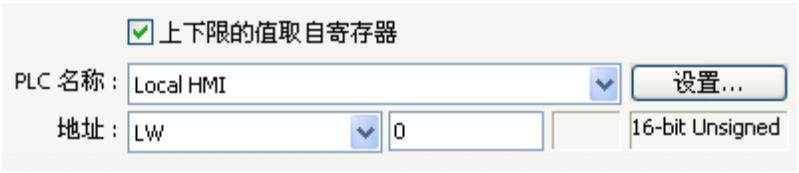




上下限的值不取自寄存器：未选择“上下限的值取自寄存器”，则上、下限值为固定值，来自直接设定的内容，参考下图。此时上限值为60，下限值为30。



上下限的值取自寄存器：若选择“上下限的值取自寄存器”，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	下限读取地址	上限读取地址
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

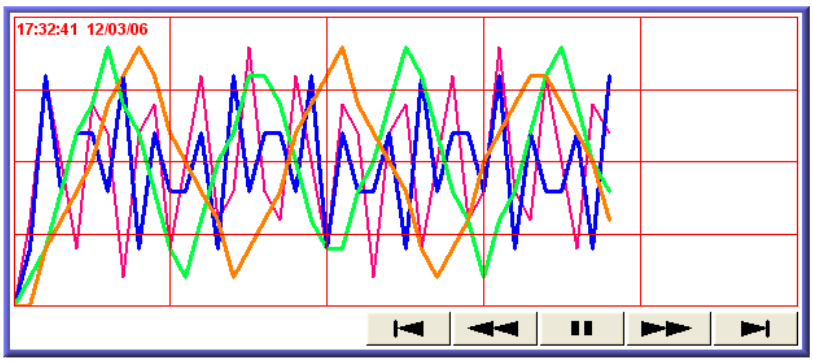
刻度符号项目

设定是否使用刻度符号于表针上

13.18 趋势图元件 (trend display)

概要

“趋势图”元件使用连续的线段描绘资料取样元件所记录的资料，如此可清楚显示数据变化的趋势，下图为一个“趋势图”元件的使用情形。



其中各按钮的功能描述如下：

 触控后画面将显示最初的取样资料，并关闭画面自动卷动功能。

 触控后画面将显示1个垂直间隔前的取样资料，并关闭画面自动卷动功能。

 显示此图形表示目前已关闭画面自动卷动功能，触控后将重新开启此项功能。

 触控后画面将显示1个垂直间隔后的取样资料。

 触控后画面将显示最新的取样资料。

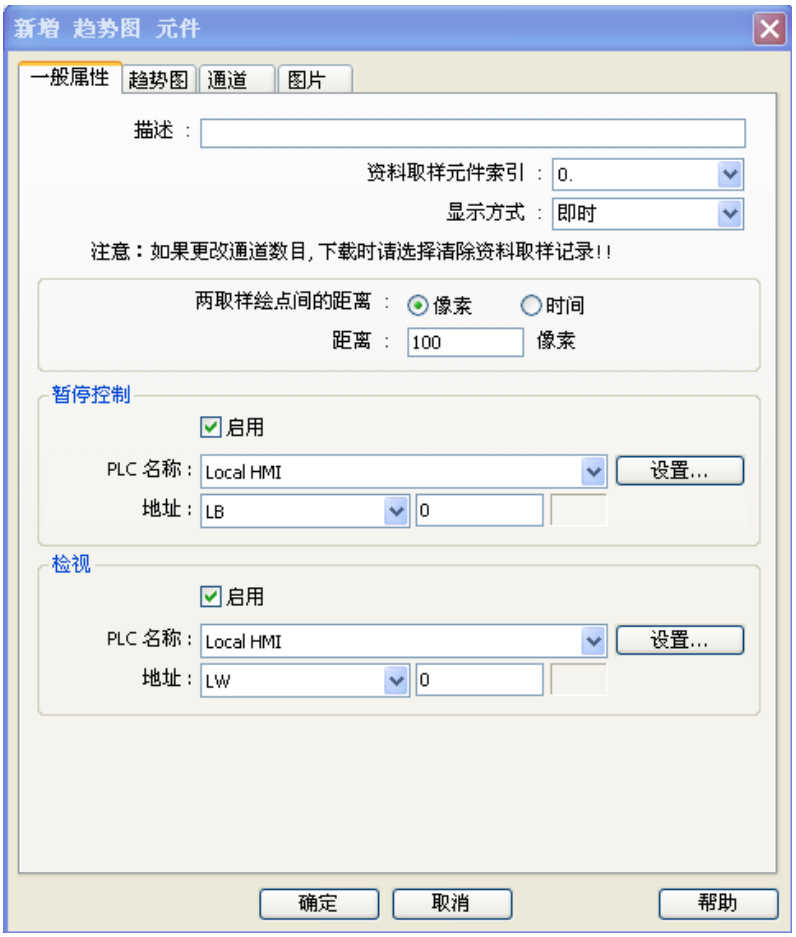
 显示此图形表示目前画面自动卷动功能已被开启，触控后将关闭此项功能。

设定

触控工具条上的“趋势图”按钮后即会出现“趋势图元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“趋势图”元件，参考下图。

注意: 必须先建立资料取样, 才可创建趋势图元件。

下图为“趋势图”元件的一般属性设定页。



[资料取样元件索引]

选择“资料取样”元件作为绘图所需的数据来源, 可参考“资料取样”元件的说明。

[显示模式]

选择数据来源的形式, 可以选择“即时”或“历史”。

a. 即时

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

可以利用“暂停控制”功能暂停元件画面更新的动作，但仅只暂停画面刷新，并不会暂停“资料取样”元件的取样动作。

b. 历史

历史记录来自“资料取样”元件使用日期来分类并储存的取样资料。使用“历史”模式可以利用 [资料取样元件索引](data sampling object index)选定要显示的历史记录，并利用“历史数据控制”选择不同日期的历史记录。下图为“历史数据控制”的设定画面。

EB8000会将取样资料的历史记录文件依时间先后排序，日期最新的文件为记录0(一般是今日已存盘的取样资料)，日期次新的文件为记录1，其余记录依此类推。

在“历史控制”中所指定寄存器中的数据如果为0，“趋势图”元件将显示记录为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。

举一个简单的例子说明“历史数据控制”的使用方式，上图的寄存器为[LW200]，假使目前的“资料取样”元件已储存的取样数据文件依时间先后分别为pressure_20101120.dtl、pressure_20101123.dtl、pressure_20101127.dtl、pressure_20101203.dtl，共4笔文件，并且今日时间为2010/12/3，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_20101203.dtl
1	pressure_20101127.dtl
2	pressure_20101123.dtl
3	pressure_20101120.dtl

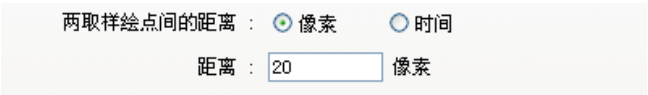
也就是说[LW200]中的数据愈小，所观察到的为与今日时间愈接近的历史记录；另一种情形是，当[LW200]中的数据并无相对应的取样数据文件时，EB8000将显示最后一个历史记录，例如[LW200]的值为4时，EB8000仍显示pressure_20101120.dtl此笔文件。

[通道数目]

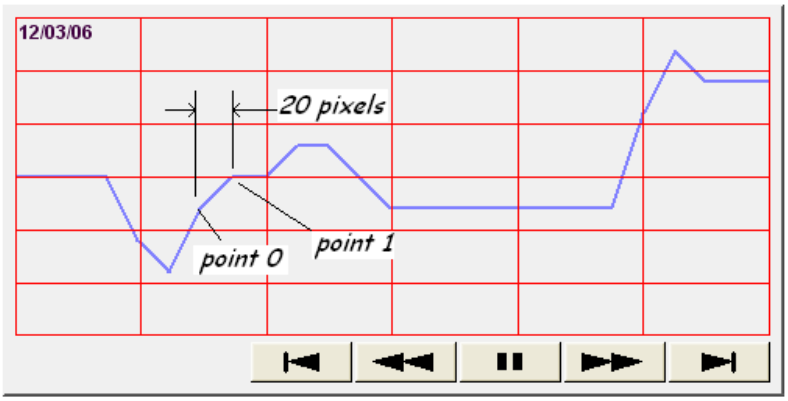
元件可显示的线条数目，一个线条代表“资料取样”元件中对某一地址连续取样获得的数据，最多可显示20个通道数目。

[两取样绘点之间的距离]

像素：

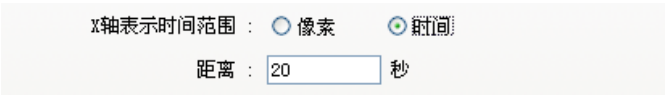


选择[像素]，则[距离]用来设定各取样点的描绘距离，参考下图。

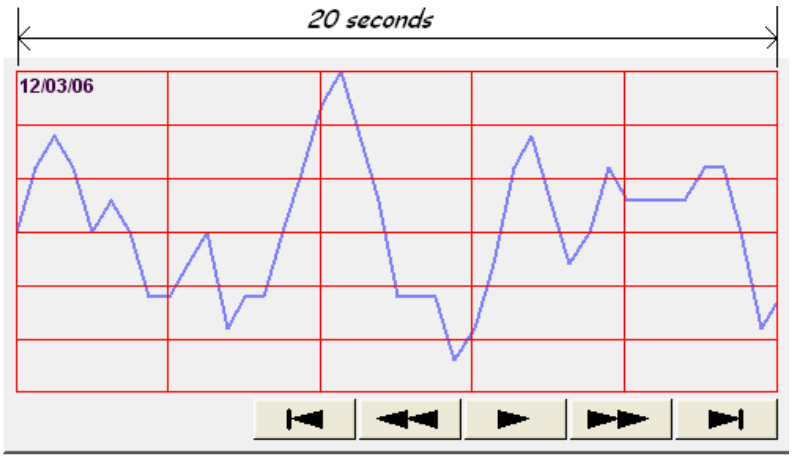


[X轴表示时间范围]

时间：



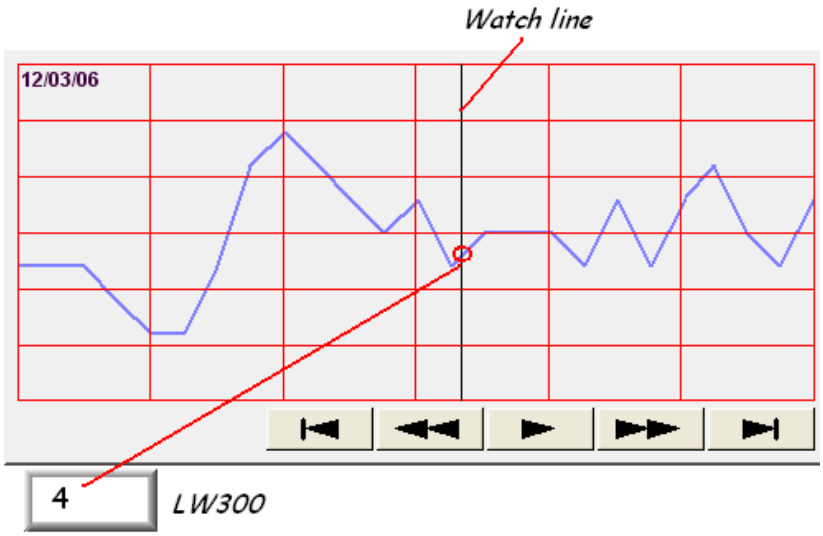
选择[时间]，则[距离]用来设定元件宽度所显示资料的时间范围，参考下图。



此外，选择时间范围后，可在趋势图页面的网格项目启用时间刻度功能。

检视项目

使用“检视”的功能可以让用户触控“趋势图”元件时产生标记符号(watch line)，并可以将标记符号所在位置的取样数据输出到指定的寄存器，以下图为例，将标记所在位置的取样数据写至[LW300]中。



“检视”功能也可以输出多条取样曲线的取样数据，EB8000会依照“资料取样”元件中所定义的取样资料资料格式，依序将标记所在位置的取样数据，从“检视”功能所定义的起始位置依序写入。例如“资料取样”元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“检视”功能所定义的寄存器，则检视线(watch line)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)

下图为“趋势图”元件的“趋势曲线”设定页。

[外框]

元件的外框颜色。

[背景]

元件的背景颜色。

[使用画面卷动控制按钮]

启用/取消画面卷动控制按钮

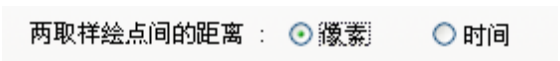
网格项目

设定网格线的数目与颜色。

显示: 选择是否使用网格线。

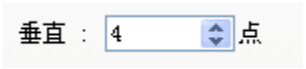
水平: 设定网格线水平线的数目。

垂直:



当选择[像素]来设定取样点的描绘距离时(参考上图与一般属性设定页), 则[垂直]用来选择每两

个垂直网格线间将包含几个取样点，参考下图。



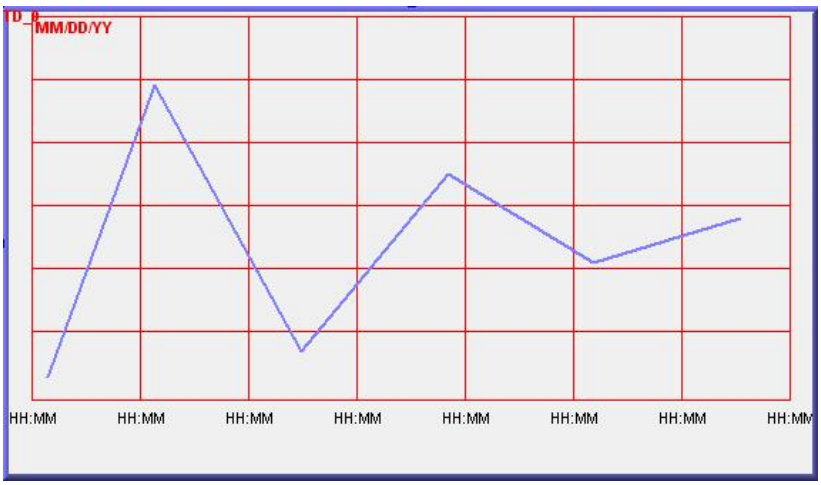
若选择[时间]来设定元件宽度所显示资料的时间范围，则[垂直]用来选择每两个垂直网格线间所显示资料的时间范围，参考下图。

EB8000会利用这些设定，自动计算垂直网格线的数目。

时间刻度：格式（选择时间刻度格式为HH:MM或HH:MM:SS）。

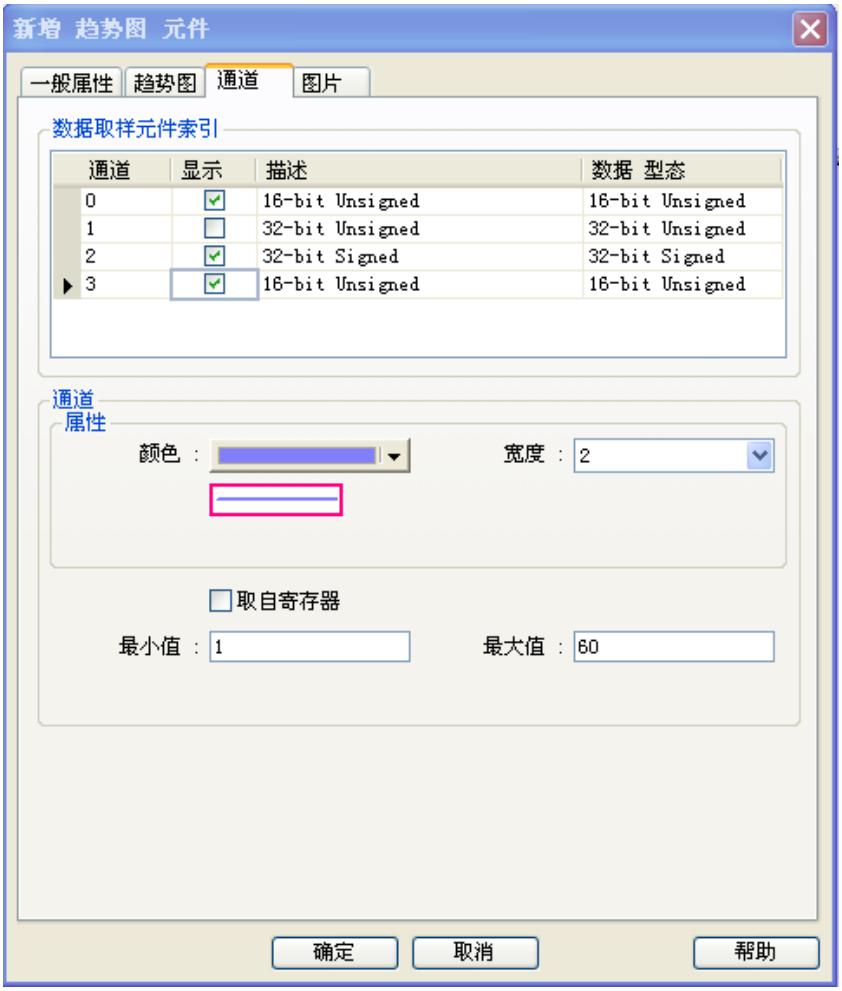
字型（选择字的字型）。

尺寸（选择字的尺寸大小，预设值为8）。



时间/日期：最新的取样资料所获得的时间会被表示在元件的左上角，此项目用来设定时间的显示格式与颜色。

下图为“趋势图”元件的“通道”设定页。



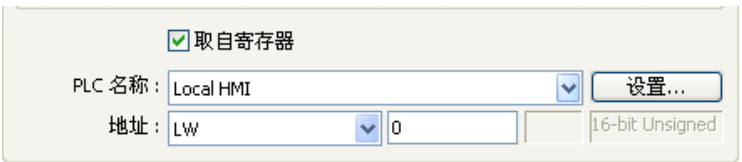
通道项目

设定各个曲线的样式与颜色，与曲线所能描绘数据的上下限值。最多可同时支持20个通道。

[最小值]、[最大值]

不勾选“取自寄存器”：[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1，最大值为60，则[最小值]与[最大值]需设定为[1]与[60]，如此所有的取样数据才会完全被描绘在元件中。

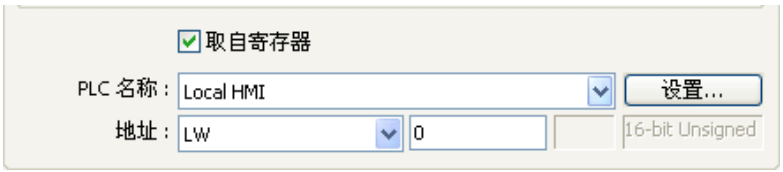
取自寄存器：选择输入数值的上下限来自所指定的寄存器。此时寄存器必须存在的资料长度与元件所显示的数据类型有关。举例来说，下图的上下限来自[LW0]:



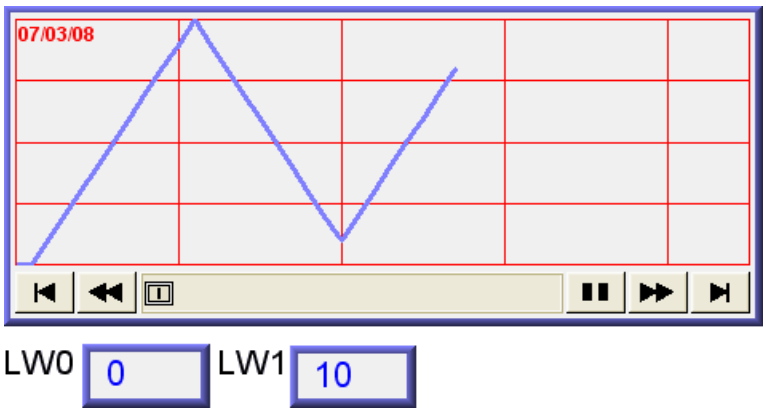
此时上下限的存放地址如下：

资料格式	最小值	最大值
16-bit 格式	位址	位址 + 1
32-bit 格式	位址	位址 + 2

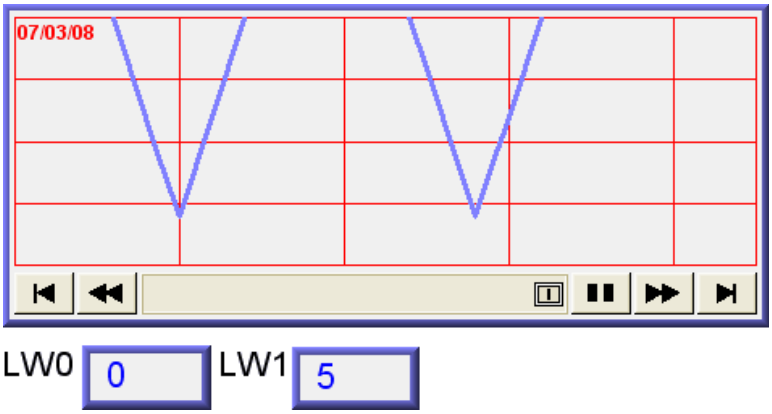
缩放趋势图的功能范例：



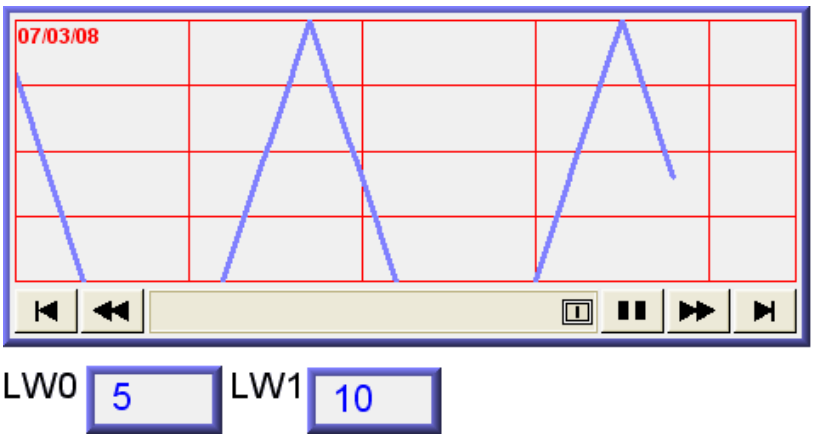
可以透过改变指定寄存器的值来放大或缩小趋势图，由上图可看出数值是经由LW0及LW1来做放大及缩小的参考。



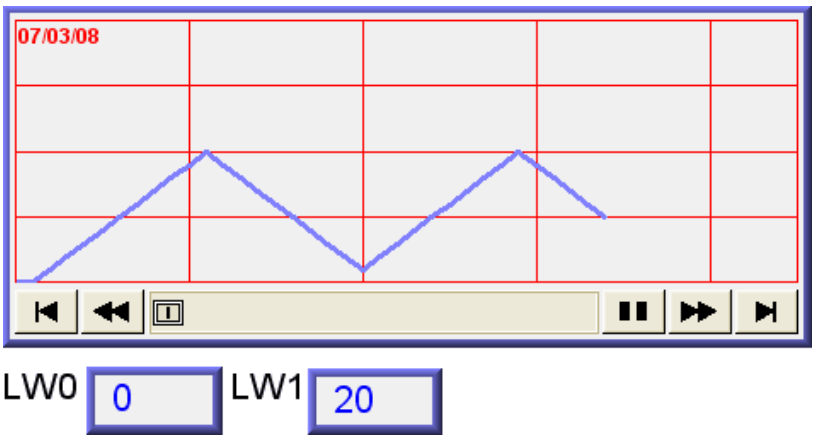
上图是正常尺寸的显示图，范围在0~10之间可以表示出整个资料的状态。



此张图为经由改变LW1的值来放大范围在0~5之间的资料状态供使用者可以观察细微变化。



此张图为经由改变LW0的值来放大范围在5~10之间的资料状态供使用者可以观察细微变化。



此张图为经由改变LW1的值来缩小整个趋势图。

13.19 历史数据显示元件 (history data display)

概要

“历史数据显示”元件用来显示已经储存的取样资料数据，跟趋势图不同的是“历史数据显示”元件使用列表的方式直接显示这些数据的内容，由于是历史数据，所以要显示最新的资料需经由切换画面来显现，如下图所示。

编号	时间	日期	频率	电流	输入电压	输出电压
9	09:39	25/07/08	0250	0350	0450	400
8	09:39	25/07/08	0000	0000	0000	0
7	09:38	25/07/08	2750	3850	4950	4400
6	09:38	25/07/08	2250	3150	4050	3600
5	09:38	25/07/08	1750	2450	3150	2800
4	09:38	25/07/08	1250	1750	2250	2000
3	09:38	25/07/08	0750	1050	1350	1200
2	09:38	25/07/08	0250	0350	0450	400
1	09:38	25/07/08	0000	0000	0000	0

设定

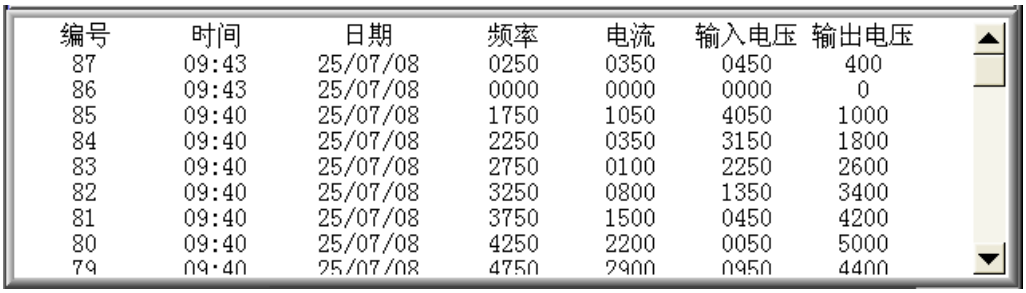
触控工具条上的“历史数据显示”按钮后即会出现“历史数据显示元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“历史数据显示”元件，参考下图。

[资料取样元件索引]

选择“资料取样”元件作为所需的数据来源，可参考“资料取样”元件的说明。

网格项目

选择元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。



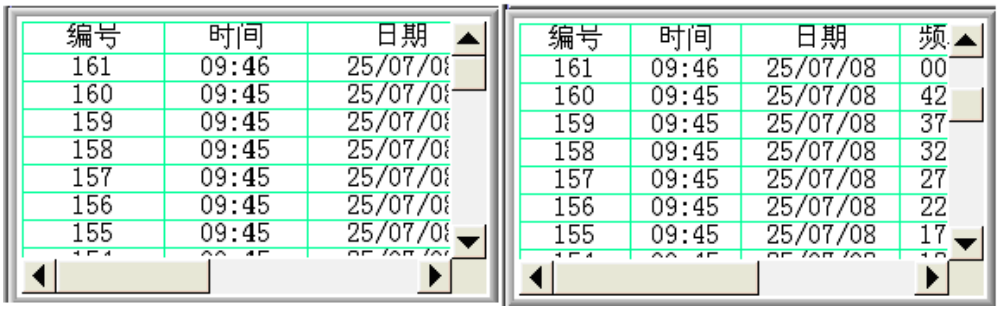
编号	时间	日期	频率	电流	输入电压	输出电压
87	09:43	25/07/08	0250	0350	0450	400
86	09:43	25/07/08	0000	0000	0000	0
85	09:40	25/07/08	1750	1050	4050	1000
84	09:40	25/07/08	2250	0350	3150	1800
83	09:40	25/07/08	2750	0100	2250	2600
82	09:40	25/07/08	3250	0800	1350	3400
81	09:40	25/07/08	3750	1500	0450	4200
80	09:40	25/07/08	4250	2200	0050	5000
79	09:40	25/07/08	4750	2900	0950	4400

[颜色]

网格线所使用的颜色

[栏宽]

此项设定值用来调整各字段间的距离，下图为使用不同[字段距离]设定时的显示情形。



编号	时间	日期
161	09:46	25/07/08
160	09:45	25/07/08
159	09:45	25/07/08
158	09:45	25/07/08
157	09:45	25/07/08
156	09:45	25/07/08
155	09:45	25/07/08

编号	时间	日期	频
161	09:46	25/07/08	00
160	09:45	25/07/08	42
159	09:45	25/07/08	37
158	09:45	25/07/08	32
157	09:45	25/07/08	27
156	09:45	25/07/08	22
155	09:45	25/07/08	17

外观项目

设定元件的外框与背景颜色，若勾选[透明]表示不使用外框与背景颜色，此时元件的外观如下图所示(此时元件也未使用图形与向量图)：

编号	时间	日期	频
654	09:57	25/07/08	00
653	09:57	25/07/08	12
652	09:57	25/07/08	07
651	09:57	25/07/08	02
650	09:57	25/07/08	02
649	09:57	25/07/08	07
648	09:57	25/07/08	12

时间、日期与序号项目

用来选择是否显示资料取样的时间与日期，并决定时间与日期的显示格式。

按时间顺序：将先显示取样时间较早的资料，参考下图。

编号	时间	日期	频
1	09:38	25/07/08	00
2	09:38	25/07/08	02
3	09:38	25/07/08	07
4	09:38	25/07/08	12
5	09:38	25/07/08	17
6	09:38	25/07/08	22
7	09:38	25/07/08	27

按时间逆序：将先显示取样时间较晚的资料，参考下图。

编号	时间	日期	频
772	10:00	25/07/08	00
771	09:59	25/07/08	37
770	09:59	25/07/08	42
769	09:59	25/07/08	47
768	09:59	25/07/08	47
767	09:59	25/07/08	42
766	09:59	25/07/08	37

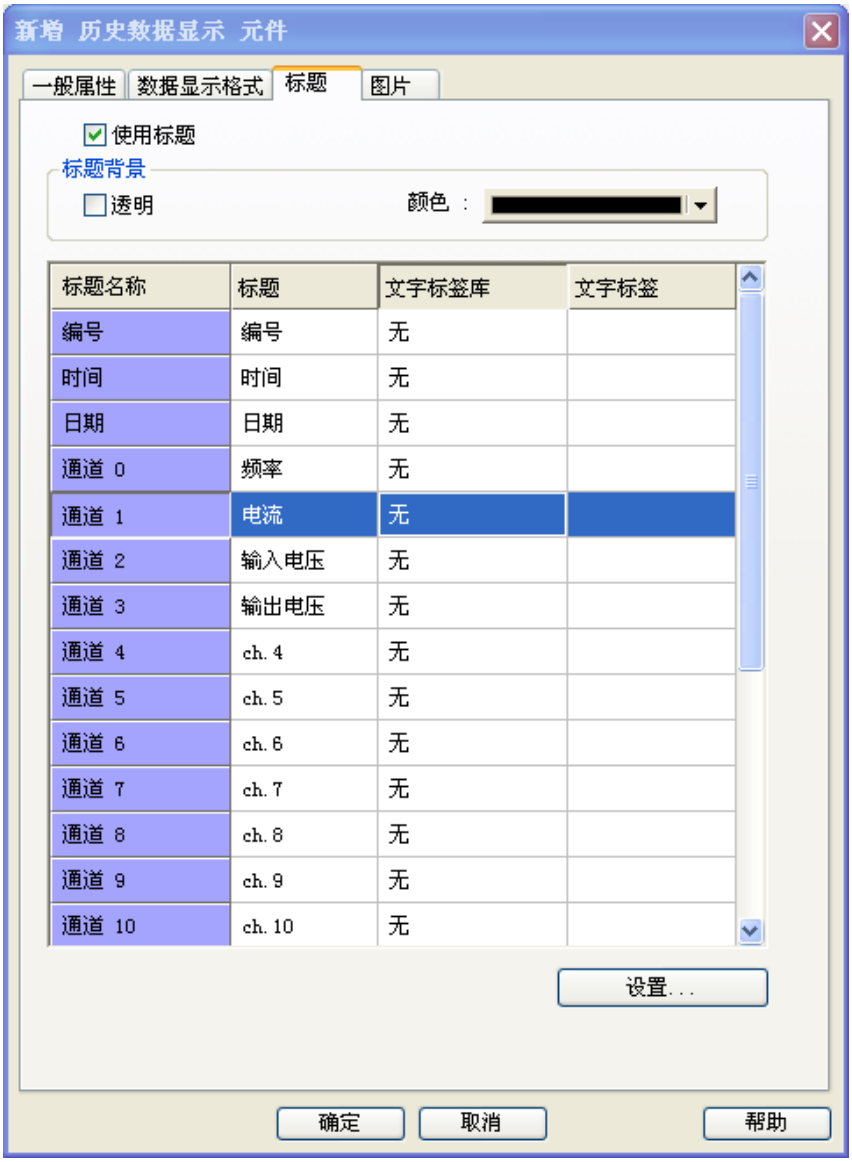
历史控制项目

EB8000会将资料取样的历史记录文件依时间先后排序，日期最新的文件为记录0(一般是今日已存盘的取样资料)，日期次新的文件为记录1，其余记录依此类推。“历史控制”项目则用来指定要显示哪一个记录，可以参考“趋势图”元件对此项目的说明。

上图的对话窗用来设定资料取样的显示格式，由上图可以发现目前使用的“资料取样”元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资料格式(例如通道0为16-bit Unsigned)，这些皆定义在“资料取样”元件中。由上图可以看出目前设定只显示通道0与通道3的数据，参考下图。

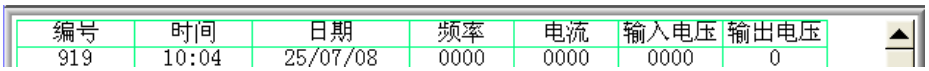
编号	时间	日期	频率	输出电压
875	10:02	25/07/08	00000	0
874	10:01	25/07/08	01750	3000
873	10:01	25/07/08	02250	3800
872	10:01	25/07/08	02750	4600
871	10:01	25/07/08	02750	5000
870	10:01	25/07/08	03000	4800
869	10:01	25/07/08	02750	3600
868	10:01	25/07/08	03000	3200
867	10:01	25/07/08	03000	2400

下图是历史数据显示元件的“标题”设定页



使用标题项目

选择是否使用标题，上图的对话窗用来设定元件所使用的标题，对照下图即可了解标题的用法，标题显示在元件的第一行。



[使用标题]

选择是否使用标题

[透明]

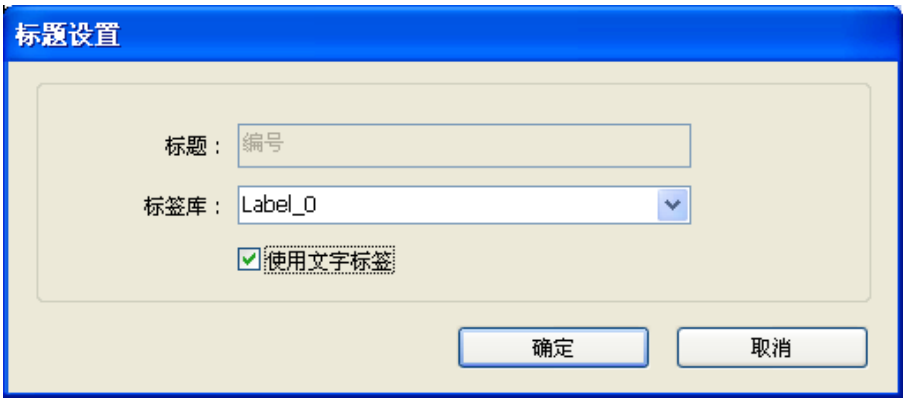
勾选[透明]表示不使用标题文字的背景色

[背景颜色]

设定标题文字的背景色

设定项目

设定标题的文字：标题也可以使用文字标签库，显示多国语言，只需使用[设置 …]功能，在出现设定对话窗后，选择使用文字标签即可。

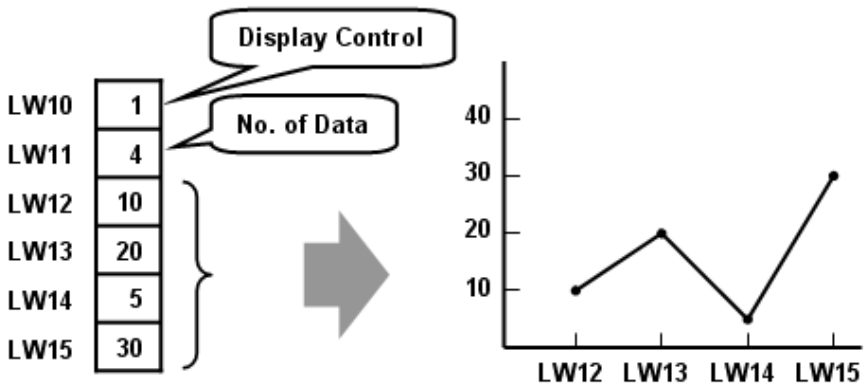


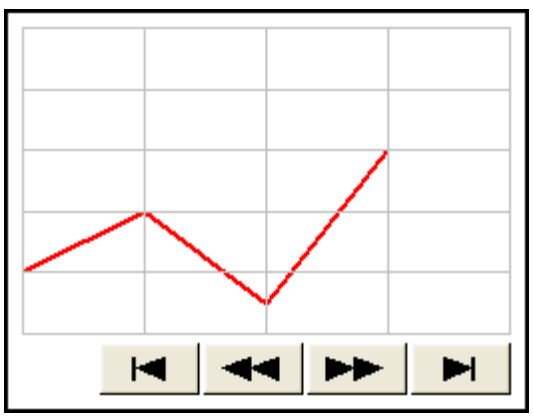
注意:当改变过“资料取样“的资料格式后，若要再执行离线模拟，需先将C:\EB8000\HMI_memory\datalog的历史数据记录删除,方可再进行离线模拟的动作。

13.20 数据群组显示元件(Data Block Display)

概要

一个数据群组(或区块)是指一组连续地址中的数据，例如LW12、LW13、LW14、LW15等。数据群组显示元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，用户可通过此方式来观察及比较各个寄存器中的数据。下图为使用数据群组显示元件显示单一数据群组LW12~LW15中的数据。



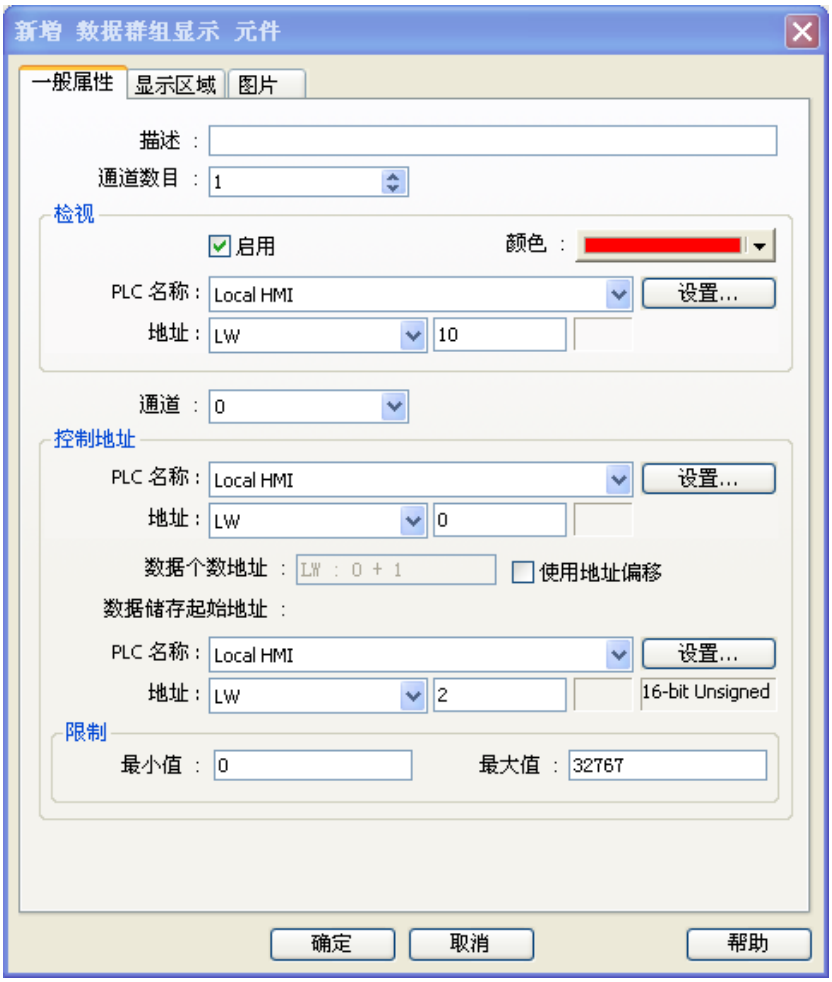


实际执行结果

设定

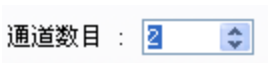
[新增元件项目]

1. 触控工具条上的“数据群组显示”按钮，随即出现元件属性对话窗。



2. [一般属性项目]

a. 通道数目: 通道数目用来设定用户欲观察数据群组的组数。每个通道表示一组群组数据, 最多可同时支持12组。



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

b. 检视: 当使用“检视”功能, 用户若触控此元件, 将可显示检视在线的数据到指定的寄存器。

通道: 用来指定要设定的数据群组

c. 控制地址:

PLC名称: 选择数据群组的数据来源

设备类型: 选择目前所指定数据群组的地址类型

控制地址: “控制地址”用来控制图形的显示及清除; 当完成“控制地址”的设定时, EB8000将自动计算产生“数据个数地址”, 地址差距为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”为LW(10+1)也就是LW11。

0 = 无动作 (预设值)

1 = 绘图

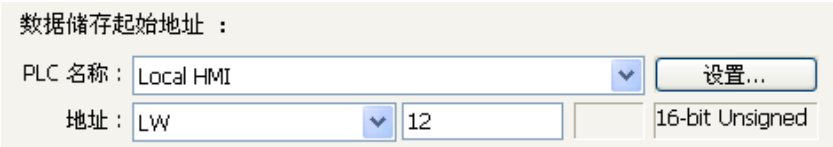
2 = 清除

3 = 清除旧图形, 显示新图形

当执行完上述动作后, 系统会自动将控制地址重新设定为0。.

数据个数地址: 设定所指定数据群组要显示的数据个数。

数据储存起始地址: 实际的数据读取地址。



数据类型: 设定资料格式。

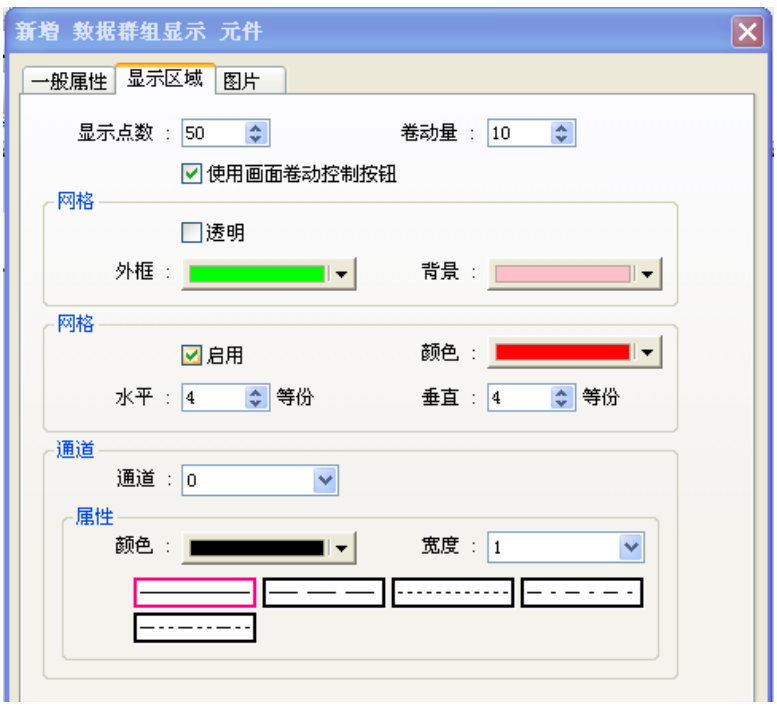
如果数据类型设定为16-bit unsigned, 则[数据储存起始地址]各加一个地址; 如果为32-bit unsigned或32-bit float, 则[数据储存起始地址]各加二个地址。

例如LW12为[数据储存起始地址], 当资料格式设定为16 bit Unsigned时, LW12为Data 1、LW13为Data 2, 依此类推。

但当资料格式设定为32 bit Unsigned或32-bit float时, LW12为Data 1、LW14为Data 2, 依此类推。

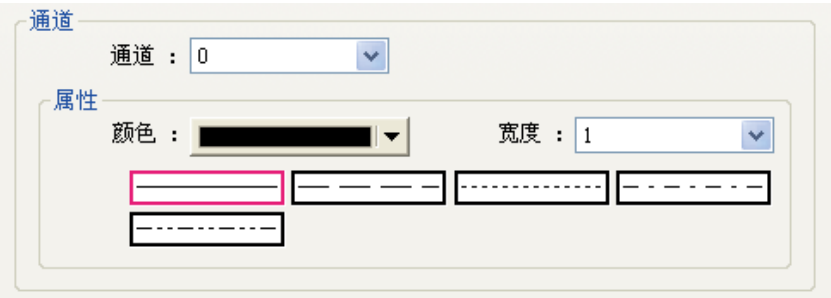
d.限制: 用来设定所显示图形之上、下限。

3. [区域显示项目]



a. 显示属性：设定图形一页所能显示最大资料笔数、左右卷动笔数及框线、背景颜色。

b. 网格属性：设定各数据群组图形之线条颜色、粗细及样式。



操作方式

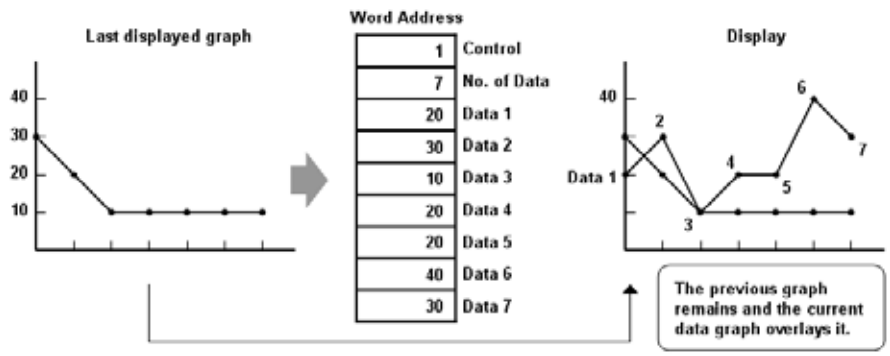
1. 如何显示数据群组的内容

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MT8000将以折线图画出目前寄存器的内容（并保留先前图形）。

d.MT8000在完成前项动作后将对[控制地址]写入“0”



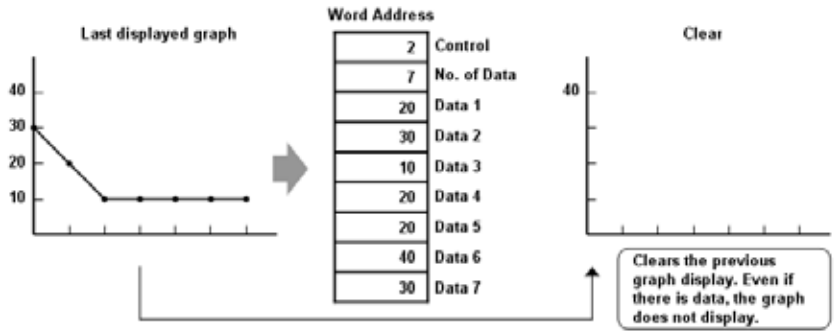
NOTE

- 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”（将此地址的bit 1设定为ON）；将清除先前所画之线图。

b.MT8000在完成前项动作后将于[控制地址]写入“0”。



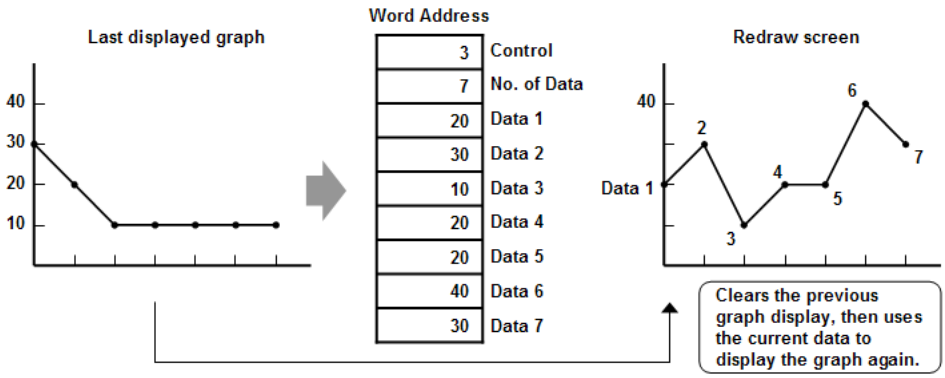
3. 清除已显示的图形并显示新数据的图形

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MT8000会先将先前的线图清除，再画出目前地址内的内容。

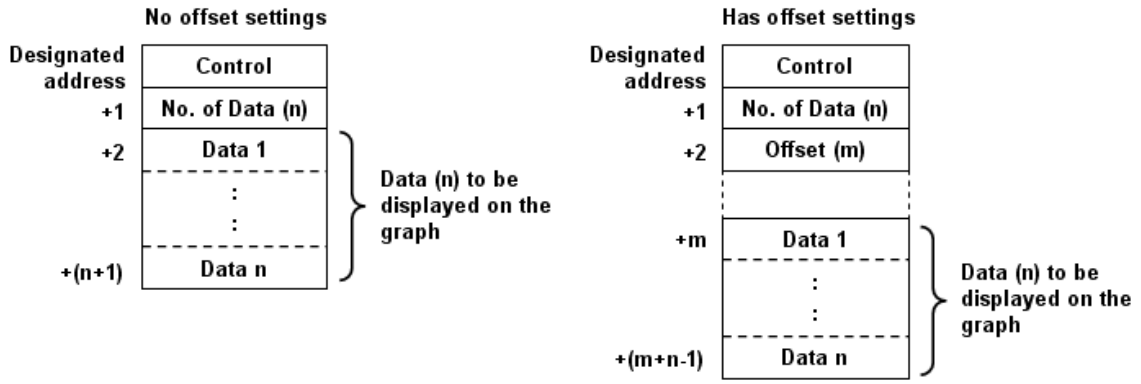
d.MT8000在完成前项动作后将于[控制地址]写入“0”。



4. 地址偏移模式

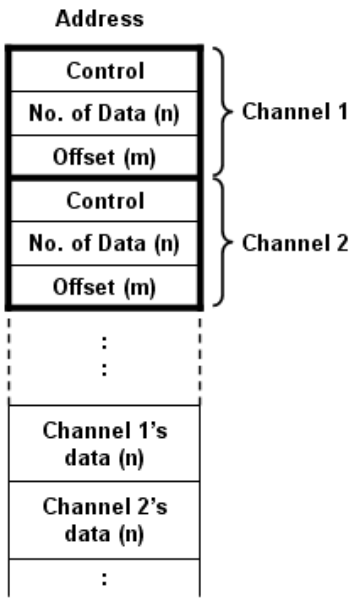
若勾选[地址偏移]模式，则原本[数据储存起始地址]将变成[数据储存偏移地址]，请参考下图。左图为未勾选[地址偏移]模式，在此种模式下[数据储存起始地址]为实际设定地址。

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。



NOTE

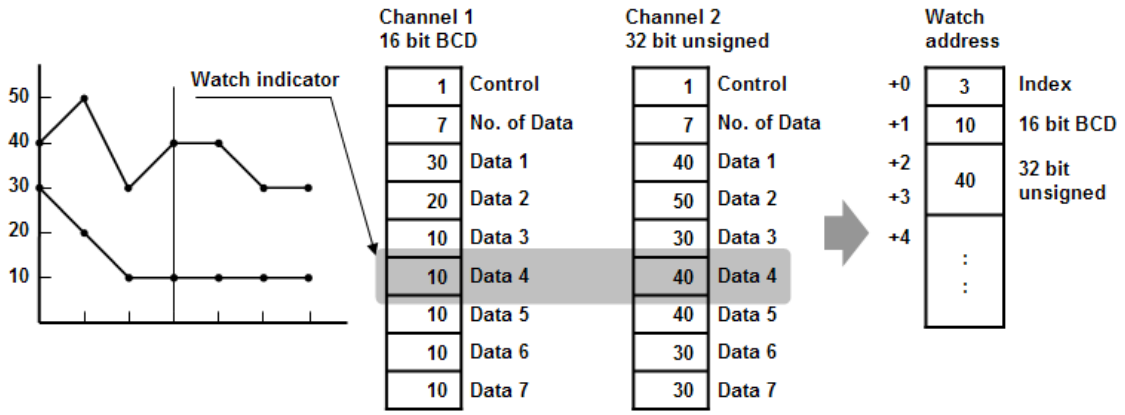
- [控制地址]、[数据个数地址]及[数据储存偏移地址]皆固定为16位无符号类型, 在元件属性对话窗所选择之数据类型是针对[Data]。
- 当指定地址后, [数据个数地址]为指定地址加1及[数据储存偏移地址为指定地址加2.
- 元件在建立后将持续地读取[控制地址]、[数据个数地址]及[数据储存偏移地址]内容, 但只有在[控制地址]的bit 0为ON时才去读取[Data]内容。
- 当指定了两个以上的通道(channel)、且各通道使用同类型寄存器时, 建议使用地址偏移模式。请参考下图: 将两个通道的[控制地址]、[数据个数地址]及[数据储存偏移地址]设定在连续的地址, 系统即可在一个通讯周期中将其全部读回, 可有效提升系统效率。



5. 数据检视功能(Watch)

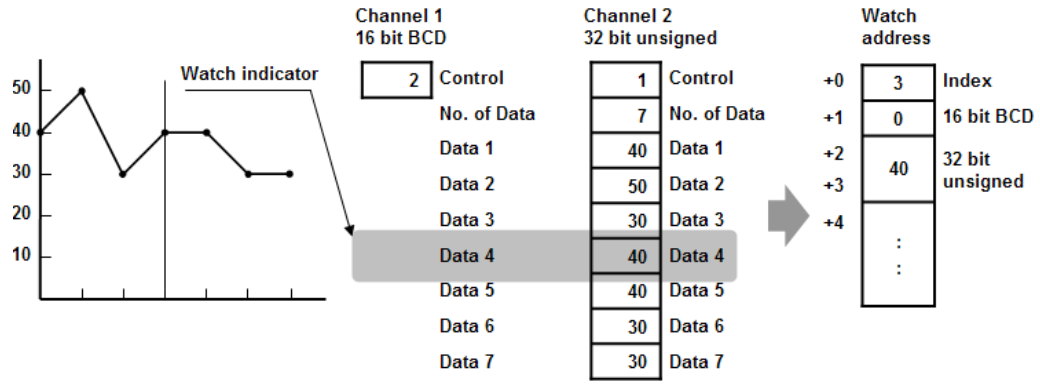
用户除了可以利用图形比较各数据群组外，亦可使用[数据检视]功能，检视各描绘点的数据。开启此功能时，使用者只需触控画面上的输出图形，EB8000将依序将目前所检视的数据编号(Data Index)与各通道的数据依序写入指定的地址中，再通过数值显示(Numeric Display)等元件读出实际内容。所写入各通道数据的资料格式则依照原来各通道定义的资料格式。

下图中显示了两组数据群组，通道1(数据群组1)为16 bit BCD的资料格式，通道2(数据群组2)为32 bit Unsigned；当检视Data 4时，元件会依序将Data Index(zero-based，也就是检视Data 4时，Index的值将为3)及两组数据群组的Data 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。

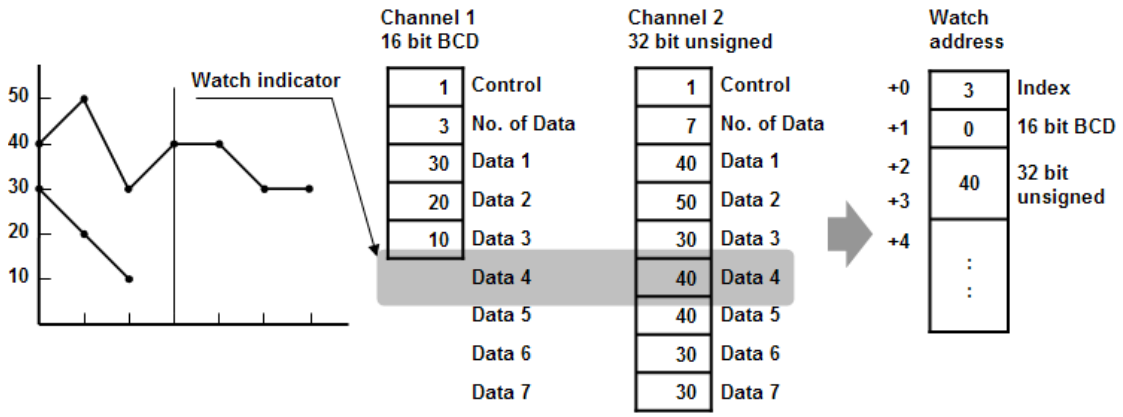


NOTE

- 数据编号[Data Index]为一个从零开始的16 bit Unsigned格式的数据；当指定的寄存器为32位时，只有较低的16位产生作用。
- 通道1可通过将[控制地址]设定为1来显示出不同时间点的数据内容(见“**操作方式–如何显示数据群组的内容**”)，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。
- 如图，若通道1在检视前被清除(或尚未显示)，则其数据将以0代替。



- 如图,若通道1仅有3笔资料,当检视Data 4时(数据笔数不足),其数据将以0代替。



[限制]

1. 可支持的通道数目上限为12。
2. 线图上限个数为32; 当到达上限后, 则不再接受显示命令。
3. 每个通道最多可显示1024个点。

13.21 XY 曲线图 (XY Plot)

概要

XY 曲线图可以同时显示最多16组的曲线，可让用户通过此方式来观察及比较各寄存器中的数据，负数亦可使用。

设定

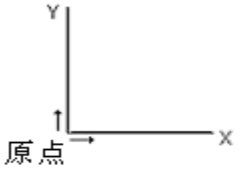
[新增元件项目]

触控工具条上的“XY 曲线图” 按钮， 随即出现元件属性对话窗。

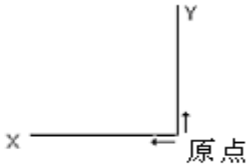
一般属性项目

方向: 选择朝右显示,朝左显示,朝上显示或朝下显示图标。

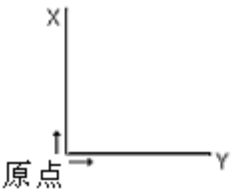
朝右显示:



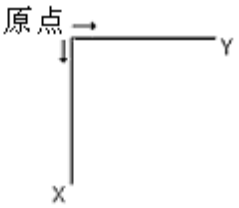
朝左显示:



朝上显示:



朝下显示:



通道数目项目

设定[通道数目]。通道数目用来设定用户欲观察数据群组的组数。

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

控制地址项目

PLC 名称：选择曲线图的控制及数据个数地址来源

设备类型：选择目前所指定曲线的地址类型

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LW0，则数据个数地址为LW(0+1)

1= 显示图形

在[控制地址]写入"1" (将此地址的bit 0设定为ON)；此时MT8000将以折线图画出目前寄存器的内容 (并保留先前图形)。MT8000在完成前项动作后将对[控制地址]写入"0"。

2= 清除图形

在[控制地址]写入"2" (将此地址的bit 1设定为ON)；将清除先前所画之线图。MT8000在完成前项动作后将对[控制地址]写入"0"。

3= 清除已显示图形并显示新图形

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

MT8000会先将先前的线图清除，再画出目前地址内的内容。MT8000在完成前项动作后将于[控制地址]写入“0”。

在设定完控制地址后，EB8000会自动设定数据个数地址。例如，如果LW0=控制地址；此地址是用来控制曲线的显示及数据清除；LW1会自动被设成数据个数地址；此地址是用来储存资料显示的数量。

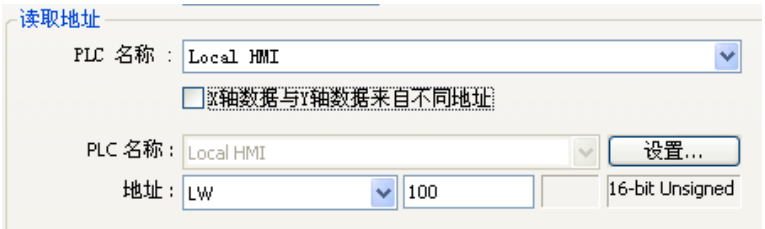
数据个数地址：设定所指定XY曲线图要显示的数据个数;每个通道的数据须小于1024点(0~1023)。

通道：用来指定要设定的曲线图。

[读取地址]

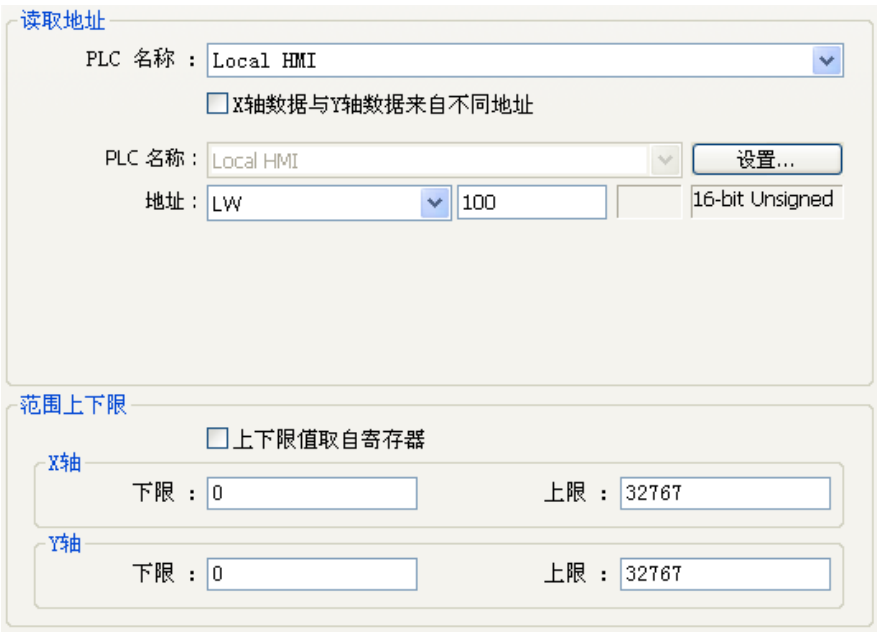
PLC 名称：选择曲线图的数据来源。点击“设置”去选择读取字地址的“PLC名称”，“设备类型”，“地址”，“系统寄存器”，“索引寄存器”。

PLC地址：



每种地址的使用方法如下

1、“上下限取自寄存器”未勾选，“X轴数据与Y轴数据来自不同地址”未勾选：



范例1：读取地址为LW100

X数据0从LW100读取数据

X数据1从LW101读取数据

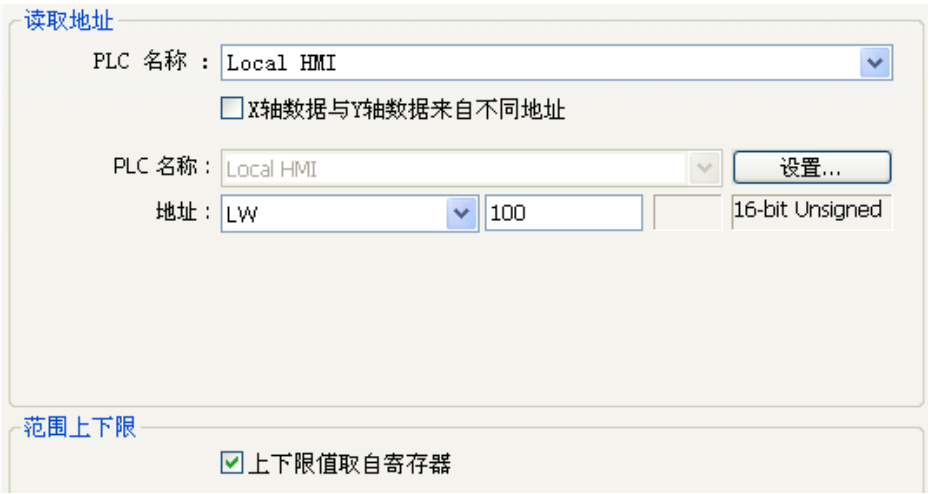
X数据2从LW102读取数据

X数据3从LW103读取数据

X数据4从LW104读取数据

X数据5从LW105读取数据……等等

2、“上下限取自寄存器”勾选，“X轴数据与Y轴数据来自不同地址”未勾选：



范例2：读取地址为LW100

X下限从LW100读取数据

X上限从LW101读取数据

Y下限从LW102读取数据

Y上限从LW103读取数据

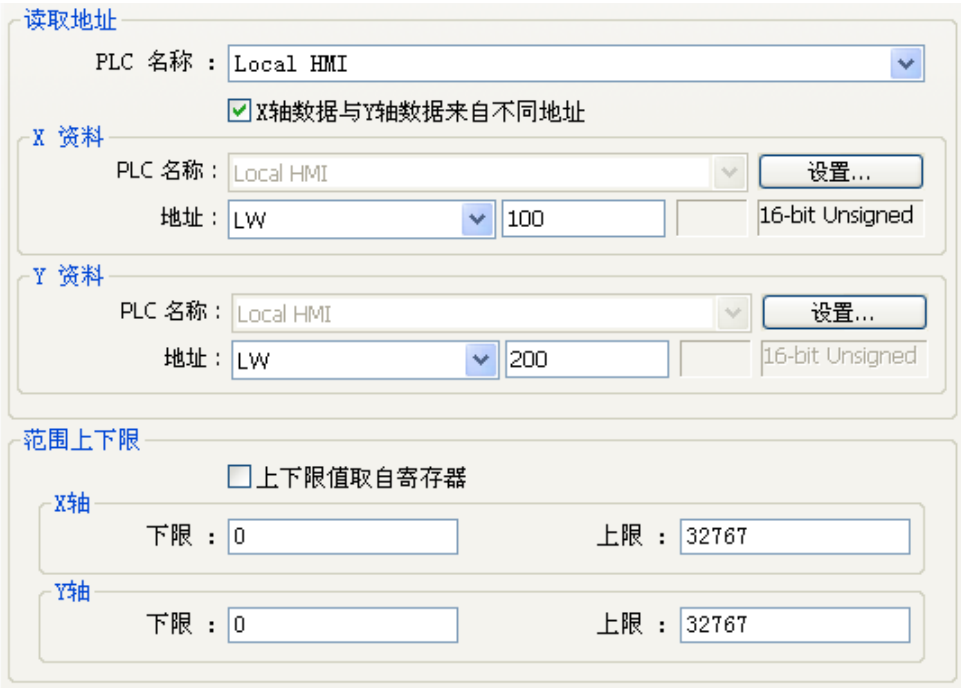
X数据0从LW104读取数据

Y数据0从LW105读取数据

X数据1从LW106读取数据

Y数据2从LW107读取数据

3、“X轴数据与Y轴数据来自不同地址”勾选，“上下限取自寄存器”未勾选



范例3：读取地址为LW100和LW200

X数据

X数据0从LW100读取数据

X数据1从LW101读取数据

X数据2从LW102读取数据

X数据3从LW103读取数据……等等

Y数据

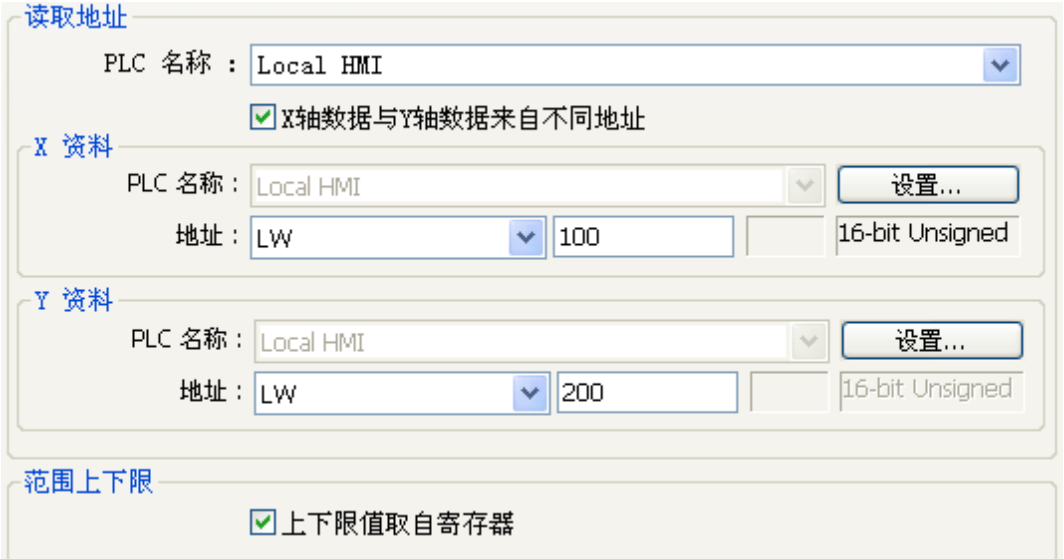
Y数据0从LW200读取数据

Y数据1从LW201读取数据

Y数据2从LW202读取数据

Y数据3从LW203读取数据……等等

4、“上下限取自寄存器”勾选，“X轴数据与Y轴数据来自不同地址”勾选



范例4: 读取地址为LW100和LW200

X数据

X下限从LW100读取数据

X上限从LW101读取数据

X数据0从LW102读取数据

X数据1从LW103读取数据

X数据2从LW104读取数据

X数据3从LW105读取数据……等等

Y数据

Y下限从LW200读取数据

Y上限从LW201读取数据

Y数据0从LW202读取数据

Y数据1从LW203读取数据

Y数据2从LW204读取数据

Y数据3从LW205读取数据……等等

[范围上下限]

上述的设定是指“上下限取自寄存器时”，如果没有勾选的话，则可自行设定上下限

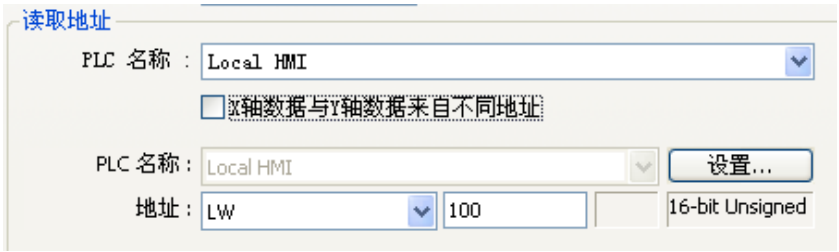
上下限用于计算X、Y轴的刻度百分比

例如：X或Y%=（X或Y读取数值–下限）/（上限–下限）

根据设定，内存分配是依据X坐标资料和资料格式来分派。

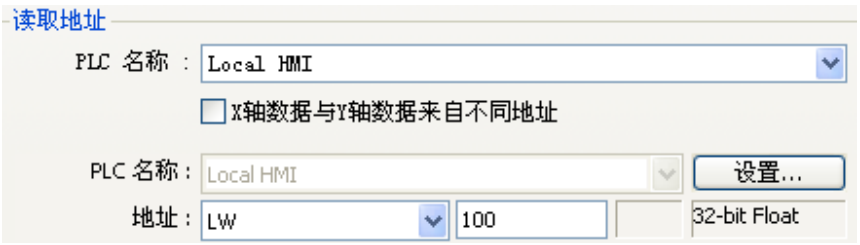
下面说明当格式为16–bit时并使用“上下限取自寄存器”：

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



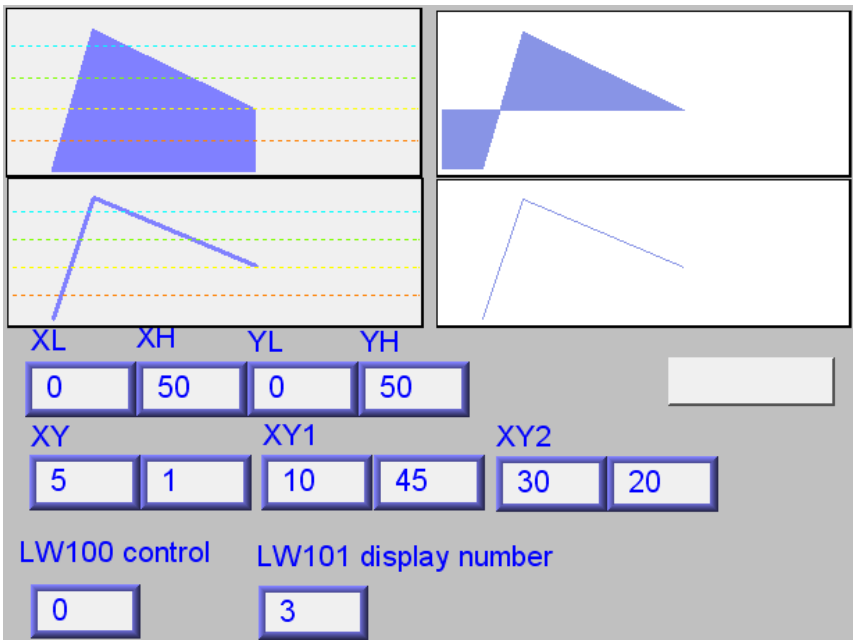
下面说明当格式为32–bit时并使用“上下限取自寄存器”。

2 word (32–bit Float)：

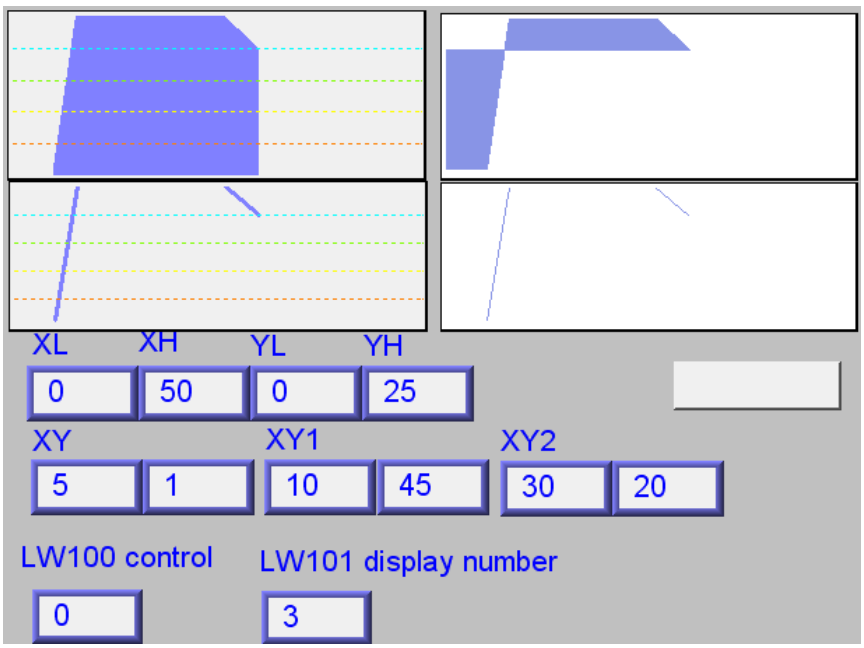


比例范围是依照X的上下限数值来显示。

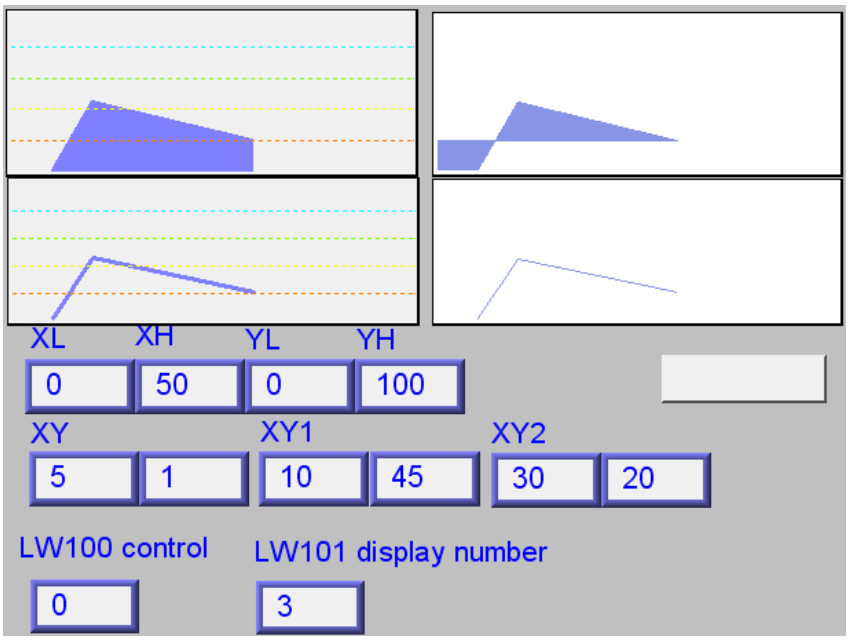
当使用“上下限取自寄存器”时，可以依照改变X轴及Y轴的上下限来达到放大及缩小的功能。(请参考“趋势图”元件的用法)



在X轴及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轴的上限可让使用者观察Y轴0~25范围的资料,可达成放大的效果。



改变Y轴的上限，可达到缩小的效果。(也可修改X轴的值来达到放大及缩小的目的)

[显示区域] 项目

外观

勾选“透明”选项时背景为透明,无勾选则依照所选择的色彩来表现外框及背景。

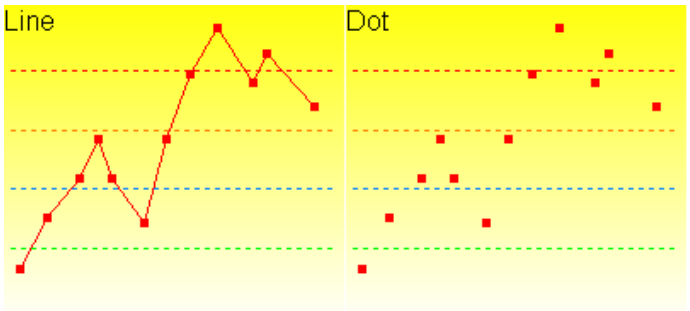
曲线

可在此设定通道所要显示的属性。

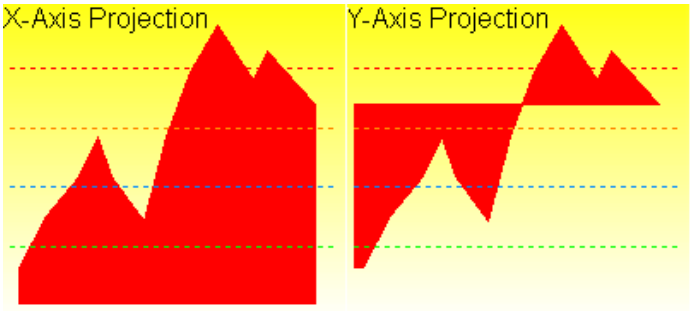
样式

设定屏幕以线,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



备注：

下图中的曲线有7个点构成，从P0到P6，系统画出X轴投影方式如以下步骤：

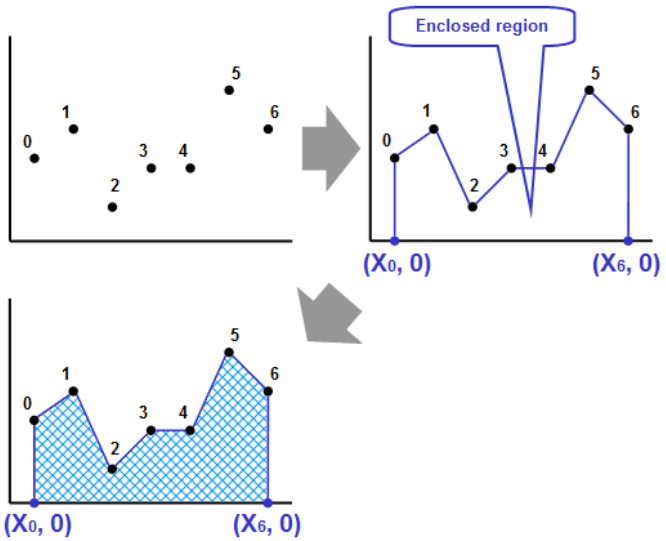
a) 自动计算出两个投影的点：X轴 – (X0, 0) and (X6, 0).

b) 依照出现的顺序，连接所有的点(X0, 0), P0, P1… P6, (X6, 0)并且最后连接第一个点

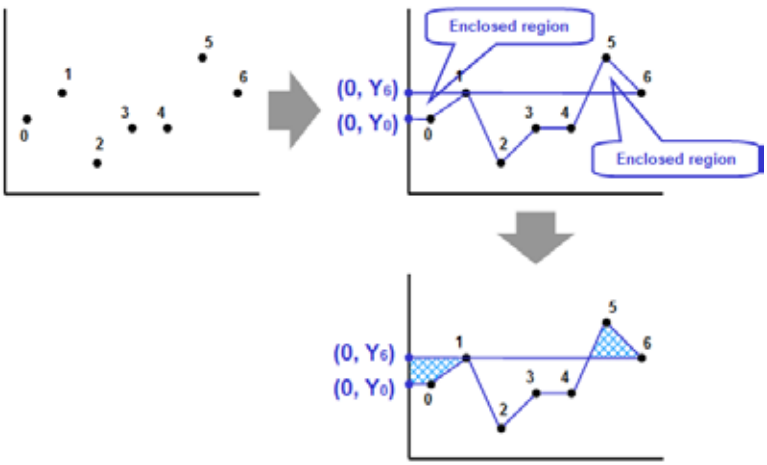
c) 填满封闭区域

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

X轴投影



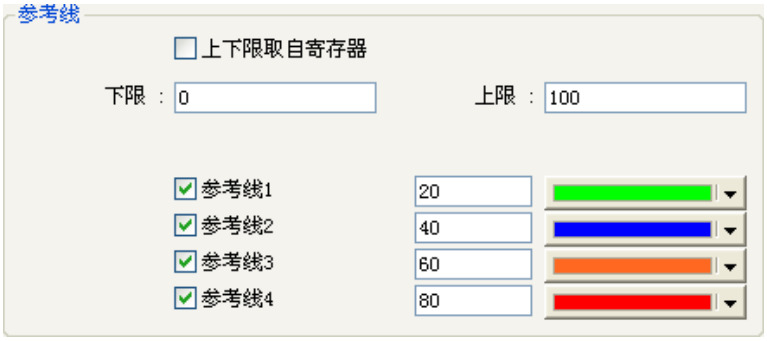
Y轴投影



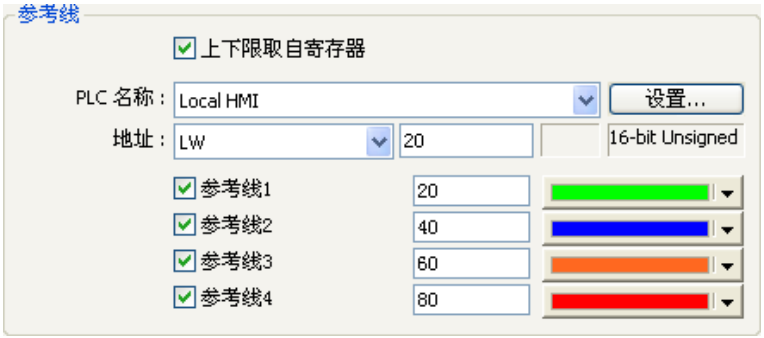
参考线

最多可画四条参考线在曲线图上,用户可以自行选择线条的颜色及参考的数值,并且依据所

设定数值来显示在屏幕上。



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。



注意: XY曲线图最多可以重复画32次.计算方法如下:

1. 1个通道可以重复画32次,

2. 若是有两个通道, 则只能重复画16次,

3. 这是依照有多少通道去除以32才能得到最多重复画的次数。

13.22 报警条与报警显示元件 (alarm bar and alarm display object)

概要

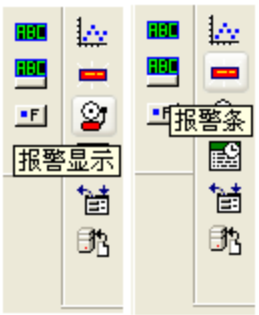
“报警条”与“报警显示”元件用来显示已被定义在“事件登录”(event log)中，且系统目前状态满足触发(trigger)条件的事件信息，此时这些事件也被称为报警(alarm)。“报警条”与“报警显示”元件将利用事件被触发的时间先后，依序显示这些报警，其中“报警条”元件将使用单行游动文字（走马灯）显示所有报警的内容；“报警显示”元件则使用多行文字列表，各行文字显示单一报警的内容。下图显示不同元件对报警的表示方式。有关“事件登录”的说明可以参考相关章节。

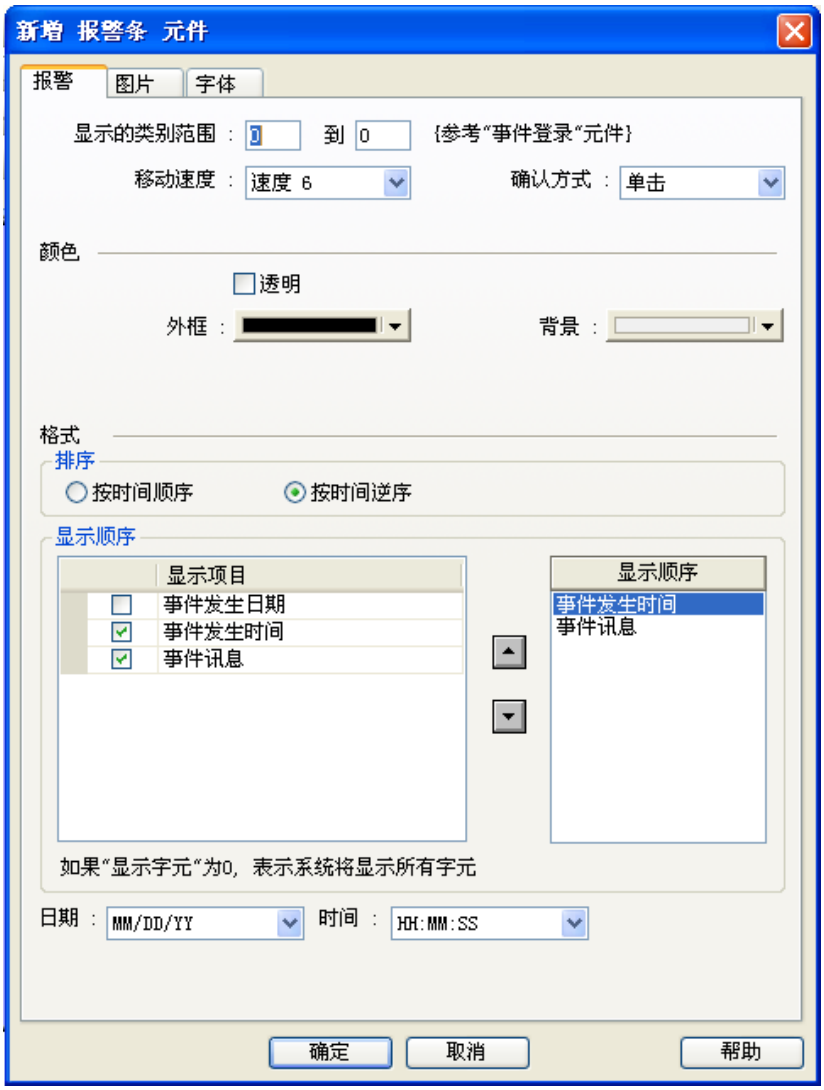
“报警条”元件

“报警显示”元件

设定

触控工具条上的“报警条”按钮后，即会出现“报警条元件属性对话窗”；相同方式，触控工具条上的“报警显示”按钮后，即会出现“报警显示元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个元件，参考下图。





[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”元件的“类别”此时被设定为2到4, 则仅有“类别”等于2或3或4的事件, 才会被显示在该“报警条”元件中。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”元件中所显示文字的移动速度。

[颜色]

设定元件的外框及背景颜色。

格式项目

a. 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：较晚发生的报警被排列在后（或在下）。

按时间逆序：较晚发生的报警被排列在前（或在上）。

b. 显示顺序

设计者可自行定义所要显示项目及排列方式。

c. 日期（事件发生日期）模式

选择显示事件发生的日期格式，共有以下4种模式：

1. MM/DD/YY　2. DD/MM/YY　3. DD.MM.YY　4. YY/MM/DD

d. 时间（事件发生时间）模式

选择显示事件发生的时间格式，共有以下3种模式：

1、HH:MM:SS；2、HH:MM；3、DD:HH:MM

可以使用“字体设定对话窗”设定元件所使用文字的尺寸与是否使用斜体效果，参考下图。各个报警所使用的字体与颜色在“事件登录”中设定。

注意：报警显示对象可支持显示多行警报内容但报警条没有支持此项功能。

13.23 事件显示元件 (event display)

概要

“事件显示”元件可以用来显示已被定义在“事件登录”(event log)中，且曾经满足触发(trigger)条件的事件，“事件显示”元件将利用事件被触发的时间先后，依序显示这些事件，参考下图。

注意:

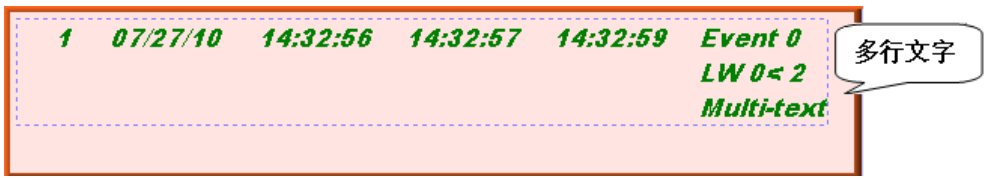
事件显示元件与报警显示元件的区别:

1. 报警显示元件只显示报警被触发时的信息，报警解除后，显示信息消失。

2.事件显示元件可显示事件被触发、确认与恢复正常状态（也就是系统状态不再满足触发条件）的时间信息，可以不同颜色表示不同的状态。

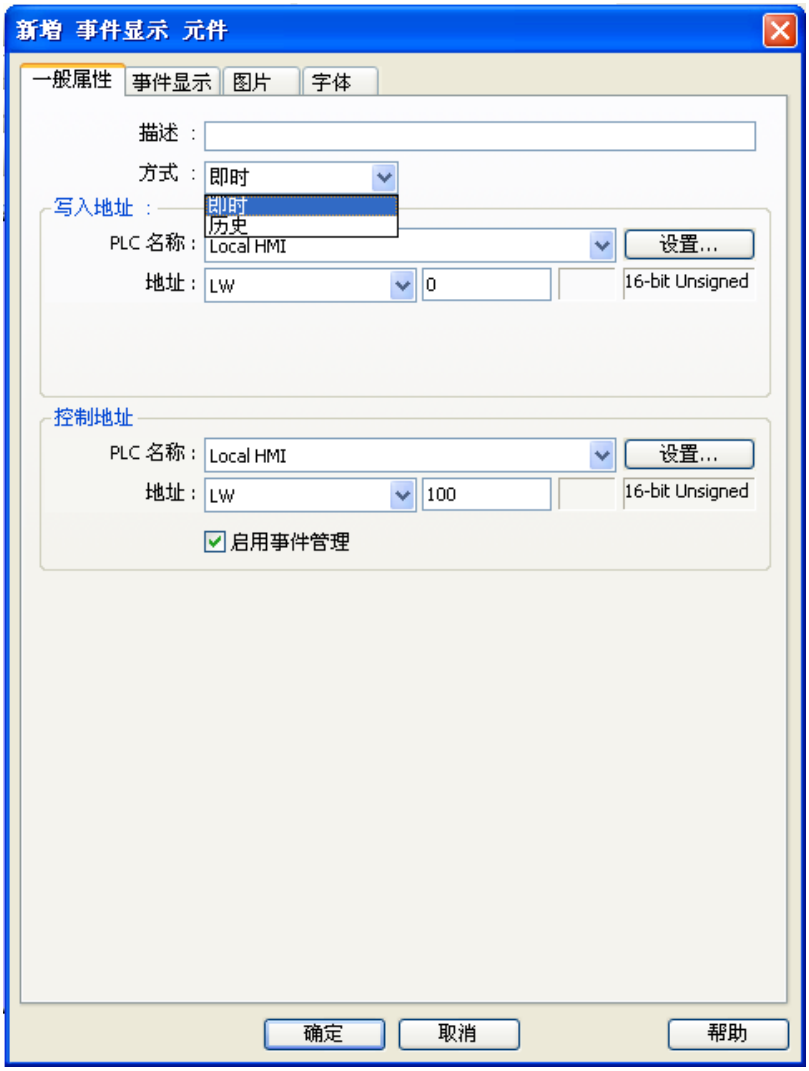
3.报警解除后“事件显示”元件仍可显示保存在HMI内存中（即时方式，断电消失）或Flash上（历史方式，断电保持）的信息。

4. 另外事件显示元件支持多行文字显示。



设定

触控工具条上的“事件显示”按钮后，即会出现“事件显示元件属性对话窗”，正确设定各项属性后触控确认键，即可新增一个“事件显示”元件，参考下图。



[模式]

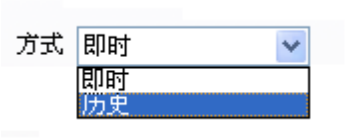
选择事件来源的形式, 可以选择 “即时” 或 “历史”。

a. 即时

写入地址

事件显示元件可显示从开机到目前被触发的事件, 即便事件触发条件已不满足。当事件被确认时, 在 “事件登录” – “信息” – “事件确认时写入” 中的数据会被输出到 “事件显示元件” 的写入地址, 请参考 “事件登录” 章节。

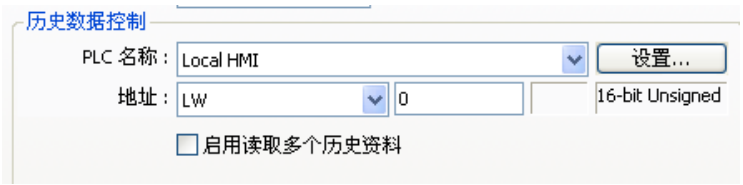
b. 历史



- “启用读取多个历史资料”

  在历史模式下可显示事件的历史记录，每一天的事件记录被储存在不同的文件内。

  下图为设定历史数据控制的控制地址



系统通过索引来选择历史记录。

在连续每天都记录历史事件的情况下，

输入0则显示当天事件信息；

输入1则显示昨天事件信息；

输入2则显示前天事件信息；

以此类推。

如果不是连续每天都开机记录历史事件时，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[LW0]，假使目前已储存的事件历史记录文件依时间先后分别为EL_20101120.evt、EL_20101123.evt、EL_20101127.evt、EL_20101203.evt，并且今日时间为2010/12/3,则依[LW0]中的数据，“事件显示”所显示的事件历史记录文件整理如下：

[LW100]中的数据	所显示的事件历史记录档案
0	EL_20101203.evt
1	EL_20101127. evt
2	EL_20101123. evt
3	EL_20101120. evt

也就是说[LW0]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。

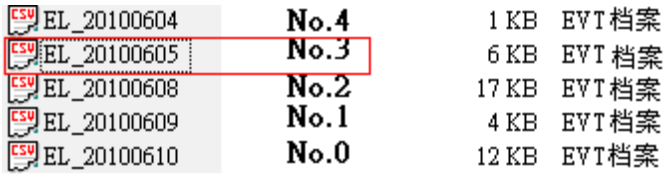
另一种情形是，当[LW0]中的数据并无相对应的取样数据文件时，EB8000将显示最后一个历史记录，例如[LW0]的值为4时，EB8000仍显示EL_20101120. evt此笔文件。

- “启用读取多个历史资料”

  定义：同时显示多天历史资料。

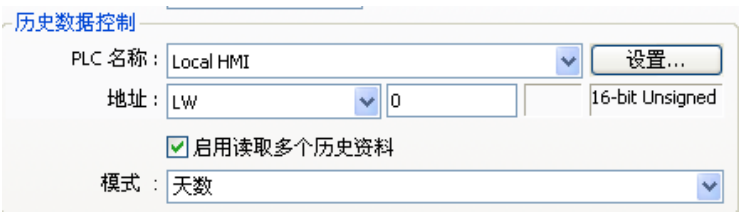
  说明：假设“历史控制”的控制地址为LW0，则LW0与LW1构成欲显示的历史资料的范围，LW0的值代表开启的第一笔历史资料。

  如下图，为了说明清楚，首先将历史资料库按日期优先作标记（No.0、No.1、No.2…），LW0输入数值为“3”，表示下图历史资料库中标记No. 3的资料。



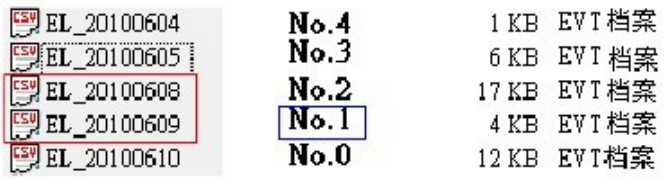
而LW1有以下两种模式：

(1) 天数

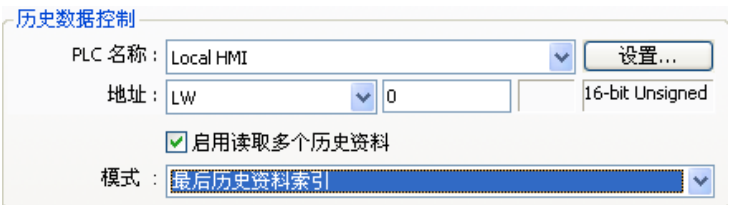


历史数据显示范围有LW0标记开始算起，LW1的值表示往前推算几天。

例如：如下图，假设LW0输入数值为“1”，LW1输入数值为“3”，则表示显示的历史数据范围有20100609开始，往前推算三天（包括20100609），历史数据库中20100607资料不存在，所以显示的历史数据只有20100609和20100608等资料。

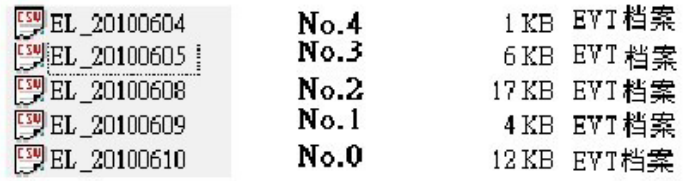


(2) 最后历史资料索引



历史数据显示范围由LW0标记开始算起，LW1的值表示资料标记结束。

例如：假设LW0输入数值为“1”，LW1输入数值为“3”，显示的历史数据为下图中的No.1、No.2、No.3的历史数据。



系统最多可显示4M历史数据，超出部分系统将略过。

以下显示资料过大的例子：

5个历史数据，每个0.5MB →最多可显示：5 x0.5MB

5个历史数据，每个1MB→最多可显示：4 x1MB

5个历史数据，每个1.5MB→最多可显示：2 x1.5MB+1 x1MB(局部)

控制地址项目

1、选择需要显示或隐藏“已确认、已恢复”事件。

2、在事件显示“即时方式”中，选择需要删除的事件。

说明：假设“控制地址”为LW100：

a. 设定[LW100+0]地址值为0 → 显示所有事件

b. 设定[LW100+0]地址值为1 → 隐藏“已确认”事件

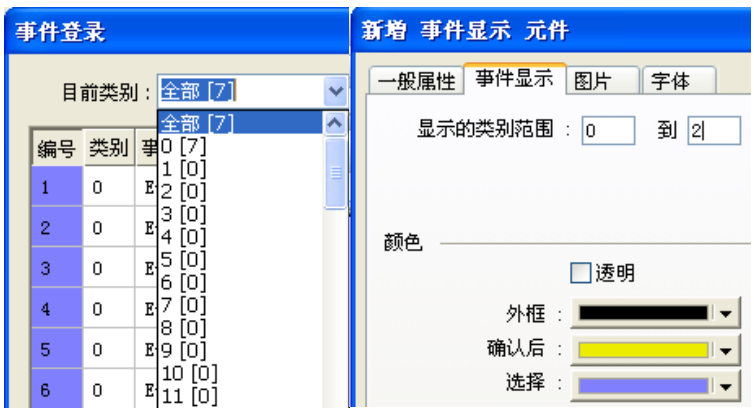
c. 设定[LW100+0]地址值为2 → 隐藏“已恢复”事件

d. 设定[LW100+0]地址值为3 → 隐藏“已恢复”和“已确认”事件

e. 设定[LW100+1]地址值为1 →表示用户可以在即时模式下选择需要删除的事件

[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”元件的“类别”被设定为2到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”元件中。可以参考“事件登录”说明中有关“类别”的解释。

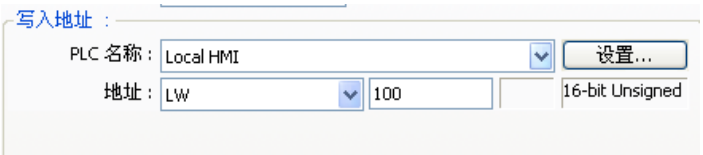


[确认方式]

可选择单击或双击的确认方式，当事件发生时，用户可依设定方式来做确认的动作。

此处“确认”动作是指：用户点击已发生并显示在“事件显示”元件上的事件，此时HMI会转变该事件的显示颜色外，也会将此事件预先设定的输出值，赋值到[写入地址]中。

以下图为例，当输出地址为[LW100]，且事件确认时的写入值为31，则当用户使用“确认”的动作时，[LW100]中的数据将被设定为31。



注意：利用此项功能搭配“间接窗口”元件，可以让不同的事件弹跳出不同的提示窗口，这些窗口通常被用来详细说明事件的内容和解决方法。

[最大事件数]

元件所能显示事件的最大数目。

当元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已发生最早一笔事件记录。按照先进先出原则，删除旧资料增添新资料。

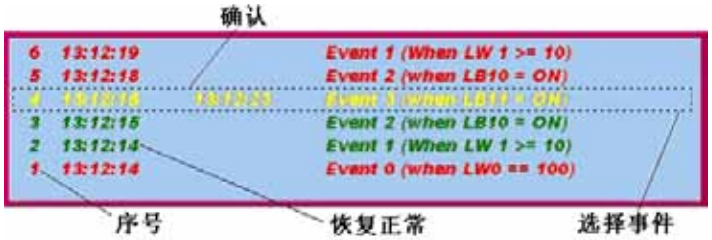
颜色项目

设定事件在各种状态下的显示颜色。

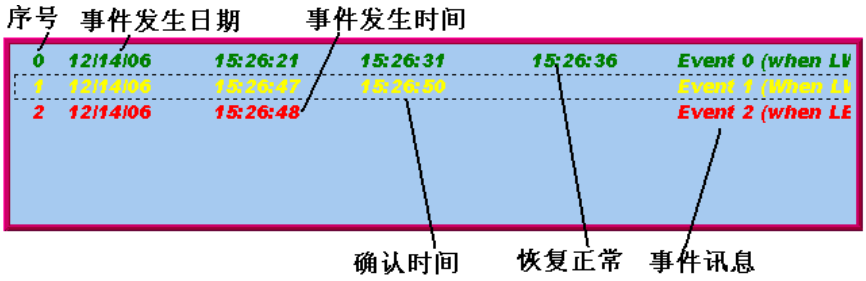
a. 确认后：事件被确认后，所使用的显示颜色。

b. 恢复正常后：系统状态无法满足事件的触发条件时，事件的显示颜色。

c. 选择：事件被选择时，作为高亮矩形的显示颜色。



格式项目



a. 排序

设定事件显示的顺序。

按时间顺序：较晚发生的事件被排列在后（或在下）。

按时间逆序：较晚发生的事件被排列在前（或在上）。

b. 显示顺序

用户可自定义所要显示的信息及排列方式

c. 日期（事件发生日期）格式

选择显示事件发生的日期格式，共有以下4种格式：

1. MM/DD/YY　2. DD/MM/YY　3. DD.MM.YY　4. YY/MM/DD

d. 时间（事件发生时间）格式

选择显示事件发生的事件格式，共有以下3种格式：

1、HH:MM:SS　2、HH:MM　3、DD:HH:MM

可以使用“字体设定对话框”设定事件所使用文字的尺寸与是否使用斜体效果，参考下图，各个事件所使用的字体在“事件登录”中设定

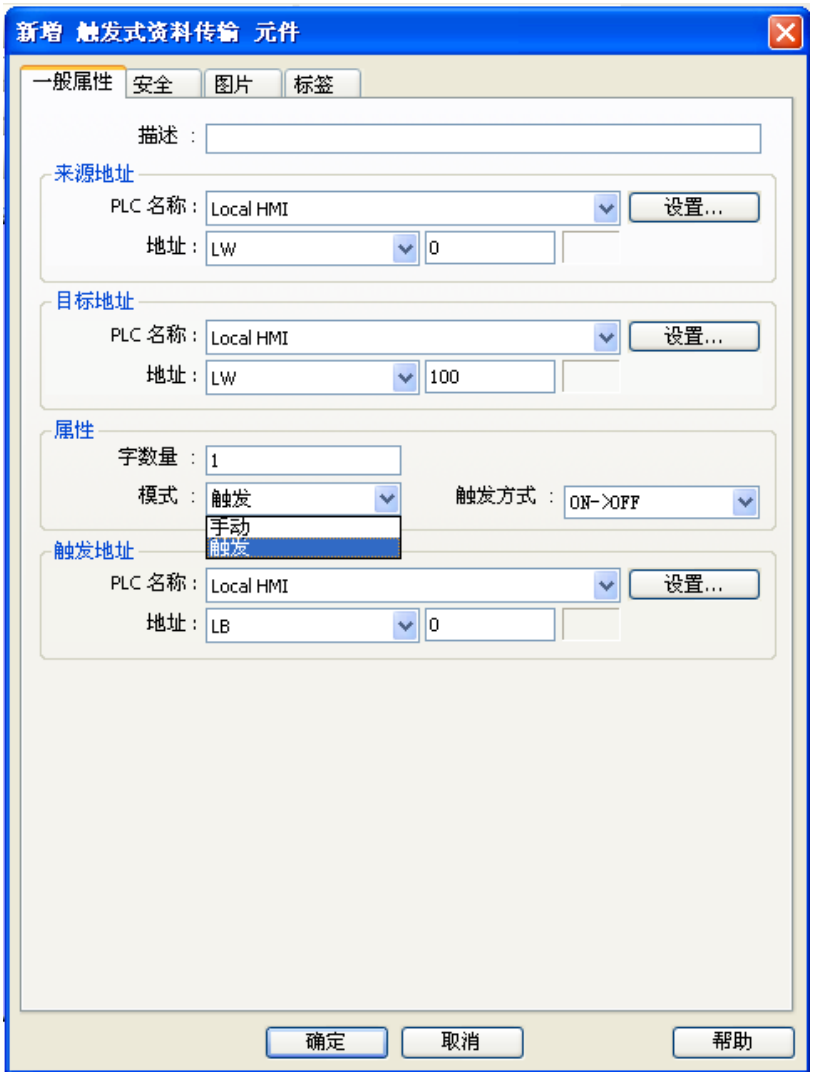
13.24 触发式资料传输元件 (trigger-based data transfer)

概要

“触发式资料传输”元件可以将指定寄存器地址区域中的数据传送到其它寄存器地址中去，可以使用手动按钮的方式传送数据，也可以使用特定地址的状态改变，来触发动作。

设定

触控工具条上的“触发式资料传输”按钮后，即会出现“触发式资料传输元件属性对话框”，正确设定各项属性后触控确认键，即可新增一个“触发式资料传输”元件，参考下图



来源地址项目

设定被传送数据的来源地址。

目标地址项目

设定数据传送的目的地址。

属性项目

设定资料传送的激活模式。

[字数]

数据的传送数量，单位为字（word）。

[模式]

可以选择“手动”或“触发”模式。

a. 手动模式

需使用手动的方式触控元件，才会进行资料传输的动作。

b. 触发模式

利用所指定寄存器状态的改变来触发资料传输的动作，利用[触发模式]选择需要的触发方式，这些触发方式包括：

[ON–>OFF]

当指定寄存器的状态由ON变为OFF，将触发资料传输动作。

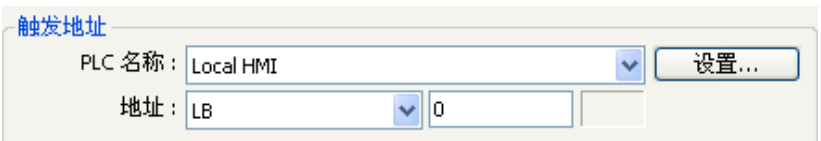
[OFF–>ON]

当指定寄存器的状态由OFF变为ON，将触发资料传输动作。

[ON<–>OFF]

当指定寄存器的状态改变，将触发资料传输动作。

触发模式所使用的寄存器地址在[触发地址]中设定，参考下图。



注意:

1.“触发式资料传输”元件常用于配方数据的传输，结合“索引寄存器”元件，使用者在HMI上选择要下载的配方组别，将一组配方数据从HMI或U盘传输到PLC上去，以适应现场生产需求。

2. 为避免触发式资料传输元件受窗口切换后中断影响，可在设计时将此“触发式资料传输”元件放置于EB8000工程画面的4号公共窗口，且不选择图片隐形显示，而在选择操作画面上由一个位开关元件触发即可。

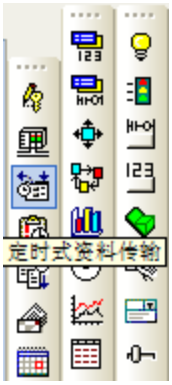
13.25 定时式资料传输元件 (time-based data transfer)

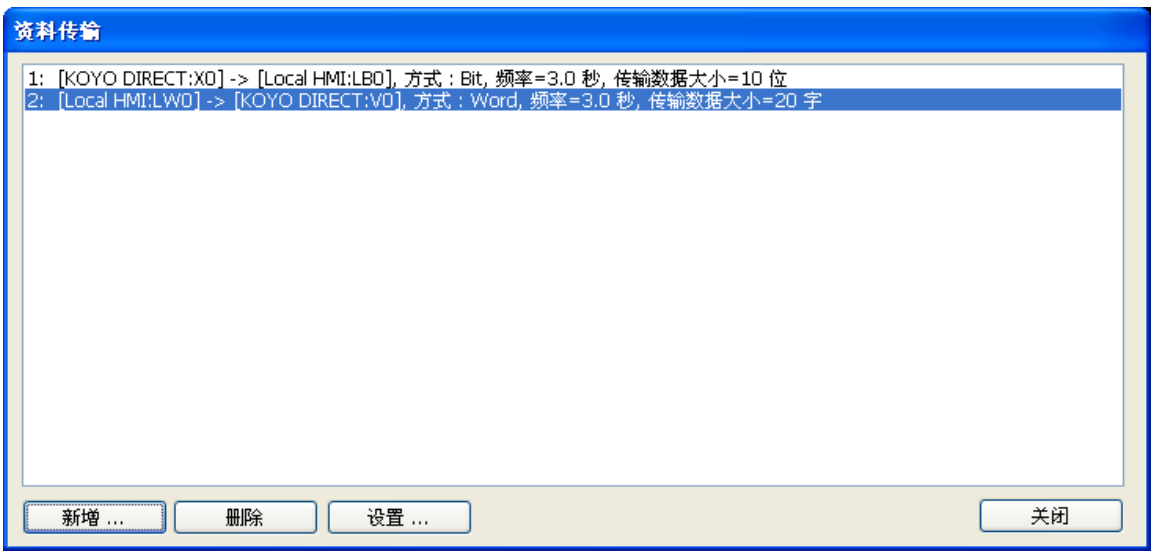
概要

“定时式资料传输”元件与“触发式资料传输”元件相同，皆用来将指定寄存器地址中的数据传送到其它地址中。与“触发式资料传输”元件不同的是“定时式资料传输”元件使用固定的频率、自动执行资料传送的工作，并且可以传送使用位（bit）为单位的数据。

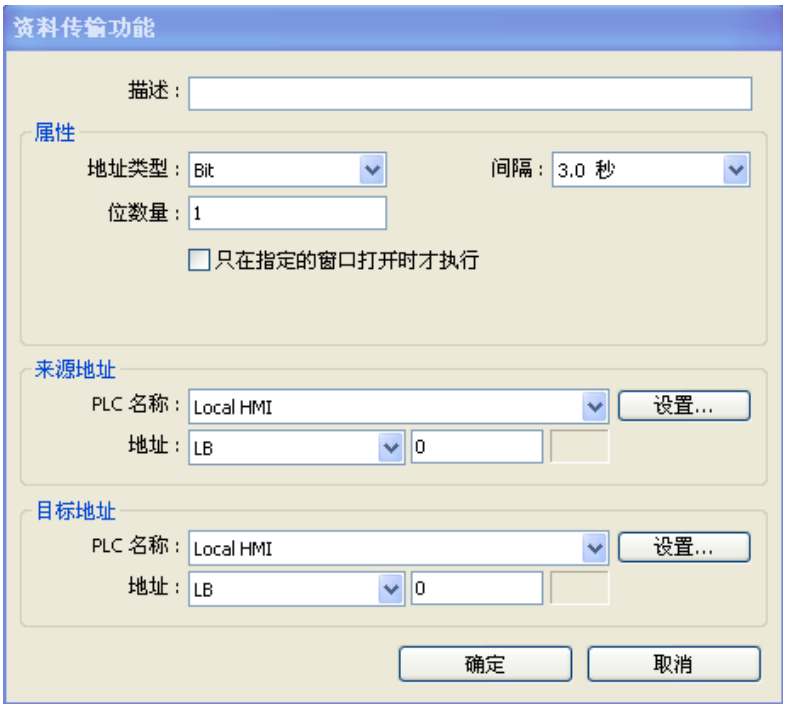
设定

触控工具条上的“定时式数据传输”按钮，即可打开“定时式资料传输列表”，参考下图：





触控“新增”按钮，即可打开“定时式资料传输功能设定对话窗”，在正确设定各项属性后，即可新增一个“定时式资料传输”元件，参考下图。



属性项目

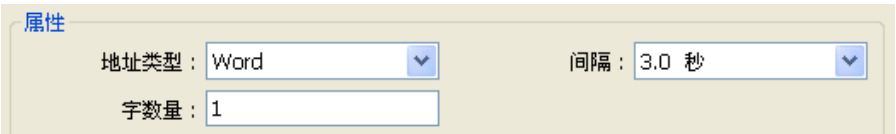
[地址类型]

选择被传送数据的类型，可以选择字（word）或位（bit）类型的数据。

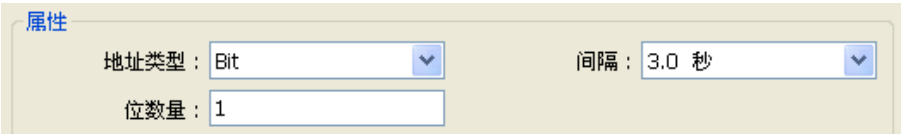
在目标地址中,如果PLC名称不是本机触摸屏（Local HMI）时,则只能选择word类型的地址。

[位数量]或[字数量]

当[地址类型]选择“Word”类型时,数据传送单位为字(word),使用[字数量]设定传送数量,参考下图。



当[地址类型]选择“Bit”类型时,数据传送单位为位(bit),使用[位数量]设定传送数量。



[间隔]

选择数据传送频率,例如选择3秒,则EB8000将每隔3秒,传送数据到指定的地址中。

注意:

1. 较小的间隔或是大量的资料传输可能会导致系统执行速度变慢,建议用户拉长传送的间隔或是一次传送少量的资料,以避免影响系统执行速度。

2. 当需要设定短时间的传输时,请注意设定间隔时间要大于资料传输时间。例如:如果数据传输操作需要2秒,您必须设置间隔时间超过2秒。

来源地址项目

设定数据的传送来源地址。

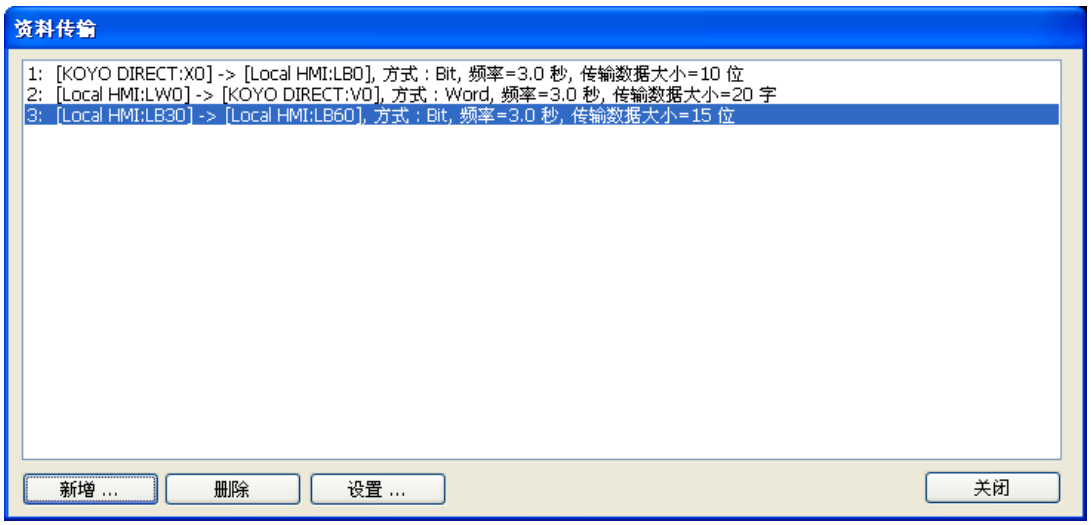
点击“设置”,设定来源地址的PLC名称,地址。

目标地址项目

设定数据传送的目标地址

点击“设置”,设定目标地址的PLC名称,地址。

在完成各项设定并触控“确定”键后,即可新增一个“定时式资料传输”元件,由“定时式资料传输”管理对话窗中可看出此新元件的内容,此元件将LB30开始的连续15个bit的状态传送到LB60的寄存器中,每隔3秒传送一次。



13.26 备份元件 (backup)

概要

利用备份元件可以将配方资料（RW, RW_A）、事件记录或指定的资料取样记录复制到指定的设备 (U盘1、U盘 2或SD卡), 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 1, 此时可以在不关机的情形下插上U盘 2, 并利用备份元件复制一份相同的资料到U盘2, 并在不关机的情形下, 直接拔取U盘 2, 这些数据即可转移到PC并做进一步的分析。当备份动作进行中, 系统寄存器LB9039状态将保持为ON。

设定

触控工具条上的 “备份” 按钮后即会出现 “备份元件属性对话窗”, 正确设定各项属性后触控确认键, 即可新增一个 “备份” 元件, 参考下图。

来源项目

[RW]、[RW_A]、[事件记录]、[资料取样记录]

这些选项用来选择要复制的文件来源，当选择备份来源为[资料取样记录]时，需使用[资料取样元件索引]选择要复制哪一个“资料取样”元件所产生的取样记录。

备份位置项目

设定来源文件的备份目标位置

a. U盘1或U盘2：U盘需事先插在触摸屏上。

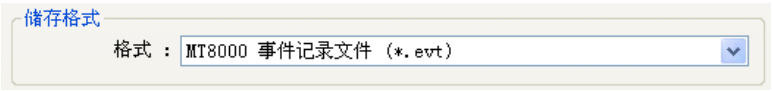
b. SD卡：SD卡需事先插在触摸屏上。

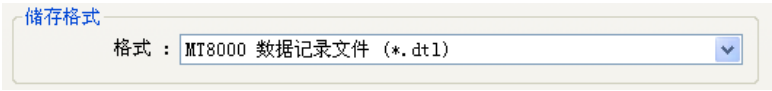
c. 远程打印/备份服务器：若使用此选项，设计者需事先在EB8000软件“编辑”菜单–“系统参数设置”的“打印/备份服务器”中做设定。

储存格式项目

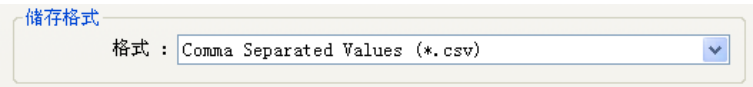
用户可选择想要存储的格式

a. MT8000事件记录文件（*.evt）/MT8000资料取样记录文件（*.dtl）





b. Comma Separated Value（*.csv）



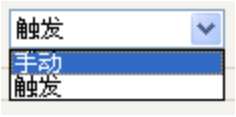
范围项目

[起始时间]可以选择[今天]或[昨天]，选择[昨天]将不复制[今天]产生的文件。

几天内：选择哪些时间范围内的文件需要被复制，例如[起始时间]选择[昨天]，并选择复制“2天”内的文件，则表示只需复制昨天与前天的文件即可。若选择“全部”则表示复制全部的文件，其它选项请参考下图。

属性项目

选择元件的执行方式，可以选择“手动”与“触发”两种模式，参考下图



a. 手动：使用者只需触控元件，即可执行文件复制动作。

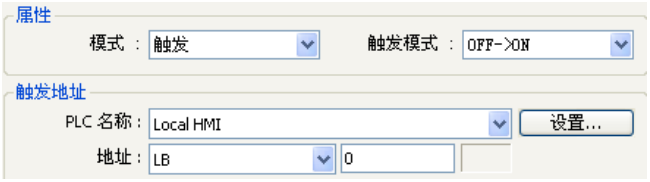
b. 触发：当指定的寄存器状态改变符合触发条件时，元件将执行文件复制动作。触发条件包含下列几种方式：

[ON–>OFF]　当指定寄存器的状态由ON变为OFF（下降沿），将执行文件复制动作。

[OFF–>ON]　当指定寄存器的状态由OFF变为ON（上升沿），将执行文件复制动作。

[ON<–>OFF]　当指定寄存器的状态改变，将执行文件复制动作。

触发地址：　当使用“触发”模式时，触发地址被用来指定元件将使用哪一个寄存器来触发文件复制动作。



13.27 排程(Scheduler)

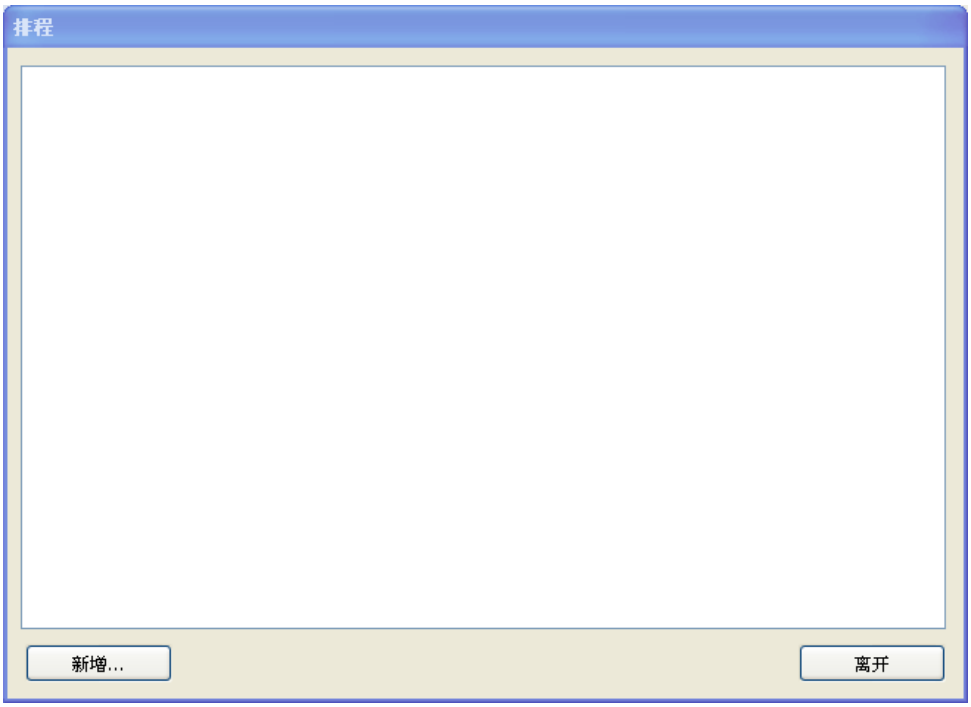
概要

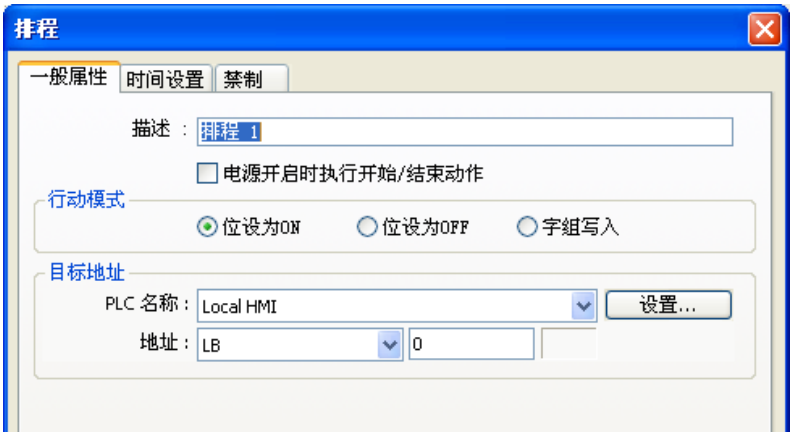
排程用来设定时刻表,让指定的位(bit)置ON或置OFF,也可以对某一个字(word)地址,写入特定数值,适合用来规划一周内的例行程序。

设定

触控工具条上的“排程”按钮后即会出现“排程对话框”,触控新增按钮,即可进入排程的设定页,参考下图:

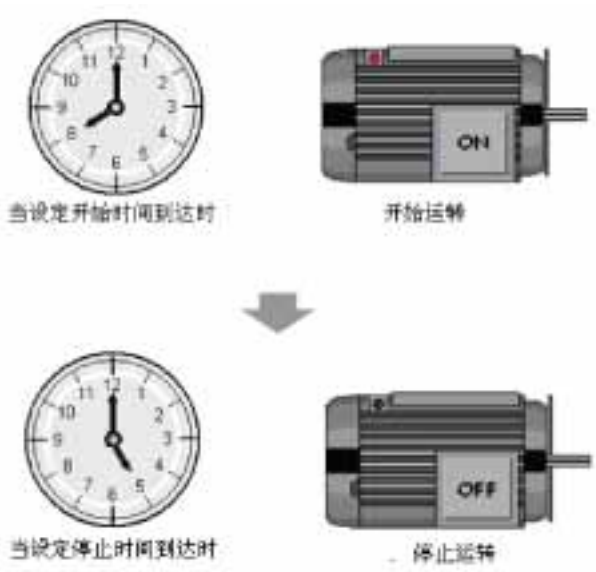
排程的对话框,触控“新增”按钮进入设定页:





[简介1-设定程序]

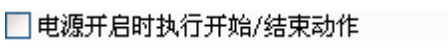
马达(地址:LB100)从星期一一直运转到星期五,时间从每天上午8点到下午5点。在此介绍的设定程序为在起始时间(早上8点)将地址LB100设为ON,在结束时间(下午5点)将地址LB100设定为OFF。



从元件菜单中选取“排程”选项或触控工具条上的“排程”按钮 ,随即出现排程新增对话框。

[设定一般页]

决定是否勾选“电源开启时执行开始/结束动作”,详细内容请参阅“一般时间设定排程导览”

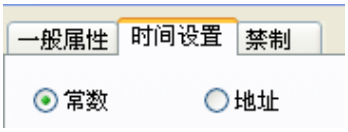


1、选定“行动模式”为“位设为ON”。

2、设定“动作地址”(例如:LB100)

[设定时间设定页]

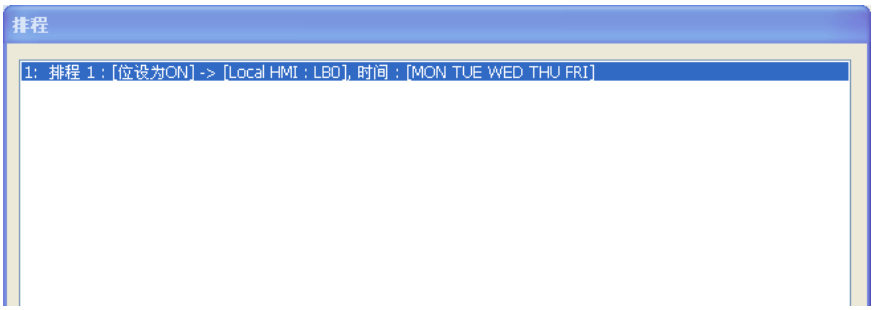
3、选择“时间设定”选项,接着选择“常数”



4、设定“开始时间及日期”,将时间设定为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”勾选项。

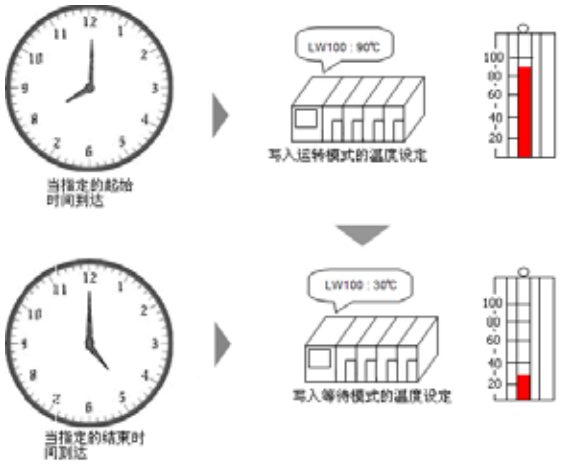
5、设定“结束时间及日期”,勾选“启用结束动作”勾选项,将结束时间设定为17点0分0秒。

6、触控“确定”键后,即可看到排程的日程表,如下图:

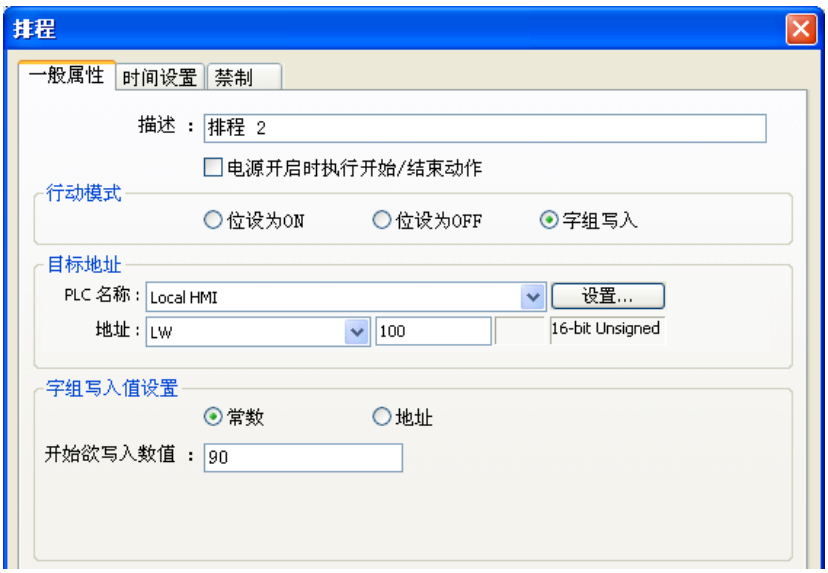


[简介2-设定程序]

从星期一到星期二,在起始时间8点把温度值90度写入字地址LW100,此时系统进入运转模式。在结束时间17点把温度设定值30度写入字地址LW100,此时系统进入等待模式。

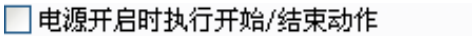


从元件菜单中选取“排程”选项或触控工具条上的“排程”按钮，随即出现排程新增对话框，触控“新增”按钮，新增“排程”元件。

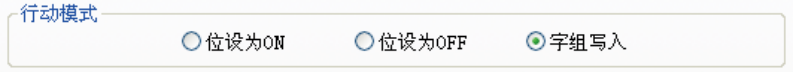


[设定一般页：]

a. 决定是否勾选“电源开启时执行开始/结束动作”，详细内容请参阅“一般时间设定排程导览”



b. 选定“行动模式”为“字组写入”。

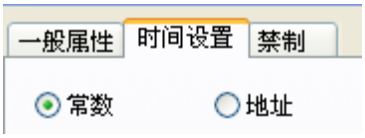


c. 设定“目标地址”（例如:LW100）

d. 选择“常数“,设定“开始欲写入的数据”为90。

[设定时间设定页]

e. 选择“时间设定”选项,接着选择“常数”



f. 设定“开始时间及日期”,将时间设定为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”勾选项。

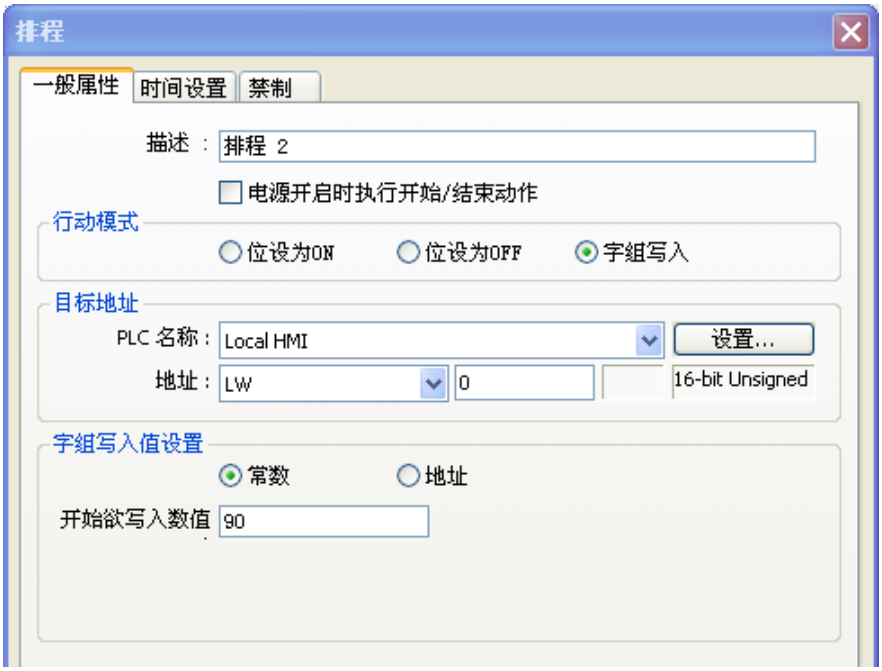
g. 设定“结束时间及日期”,勾选“启用结束动作”勾选项,将结束时间设定为17点0分0秒。

h. 选择“一般”属性页,设定“结束欲写入数值”为30

i. 触控“确定”键

一般时间设定排程导览

■ 动作



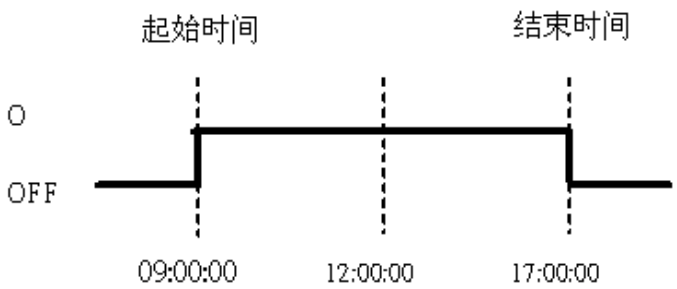
行动模式

选择在设定的时间要操作的类型

[位设为ON]:在起始时间时,把指定的位设定为ON。在结束时,设为OFF。

例如:起始时间:09:00:00

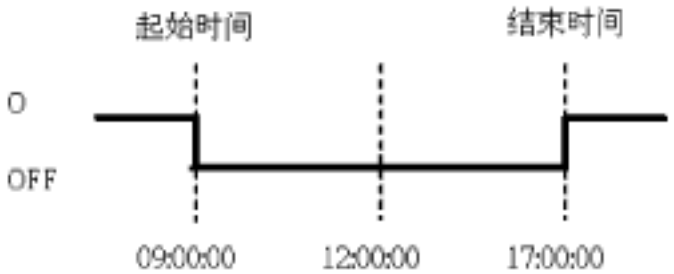
结束时间:17:00:00



[位设为OFF]:在起始时间时,把指定的位设定为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]:在起始时间时,把指定值"开始欲写入数值"写入字地址,在结束时,写入"结束欲写入数值"。

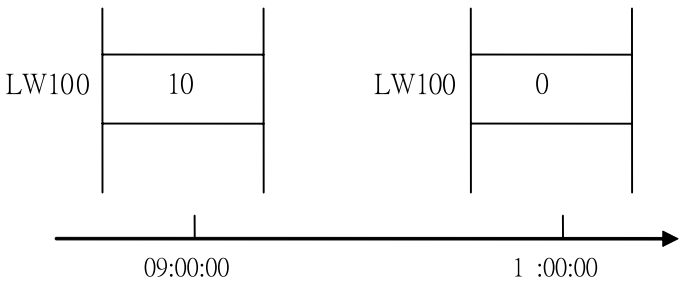
例如: 设备地址:LW100

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



[排程动作地址]:用来控制排程的指定地址。

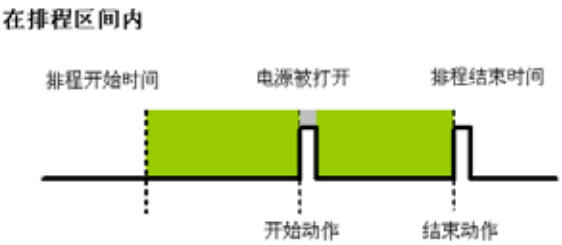
[电源开启时执行开始/结束动作]:当电源打开时,执行已设定的动作。

- 启用时

假如HMI的电源在排程区间内被打开,则开始动作会被执行

假如HMI的电源在排程区间外被打开,则结束动作被被执行

在排程区间内



在排程区间外

- 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,然后结束动作会自动执行

  当然,假如结束动作未被设定,将无法正确的判定排程区间,因此动作将不会被执行。

[排程开始欲写入数值]:当到达指定排程的开始时间,将数值写入排程动作地址。

- 选“常数”时

  排程起始时,欲写入的数值;

- 选“地址”时

  排程起始时,存放写入数值的地址。

[排程结束欲写入数值]:当到达指定排程的结束时间,将数值写入排程动作地址。

- 选“常数”时

  排程结束时,欲写入的数值

- 选“地址”时

  排程结束时,存放写入数值的地址。

  你必须在[时间设定]对话框中勾选[启用结束动作]才能使用这个选项。

NOTE

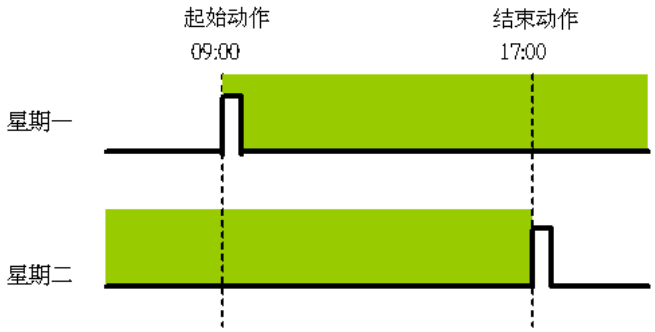
■ 时间设定/当用户选择“常数”

[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:特定地址存储时间和日期的信息

[设定为单一日期]:

- 启用时:假如用户欲设定一个范围为2天以上的排程,可以勾选这个选项,单只能设定单一的起始时间和单一的结束时间。



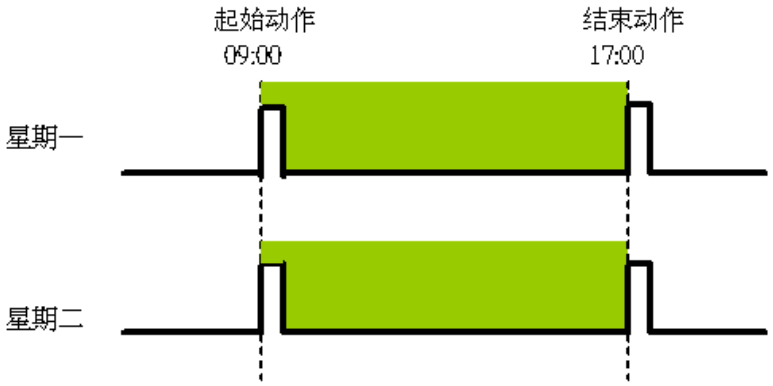
1、你必须要输入开始时间和结束时间

2、你不能在开始时间和结束时间栏里输入一模一样的时间和日期。

- 停用时:

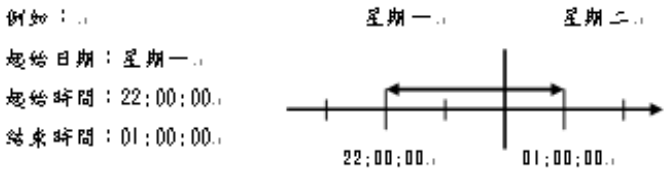
排程时间必须被设定在一天之内(起始时间和结束时间必须在24小时内),可于排程内选择多个起始和结束日期,也可在每天的相同时间执行特定动作

当用户想指定结束时间,请勾选“启用结束动作”



1、你不能在起始时间和结束时间栏里输入一模一样的时间和日期

2、此种时间排程只适用于一天之内的排程,因此如果所写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执行。

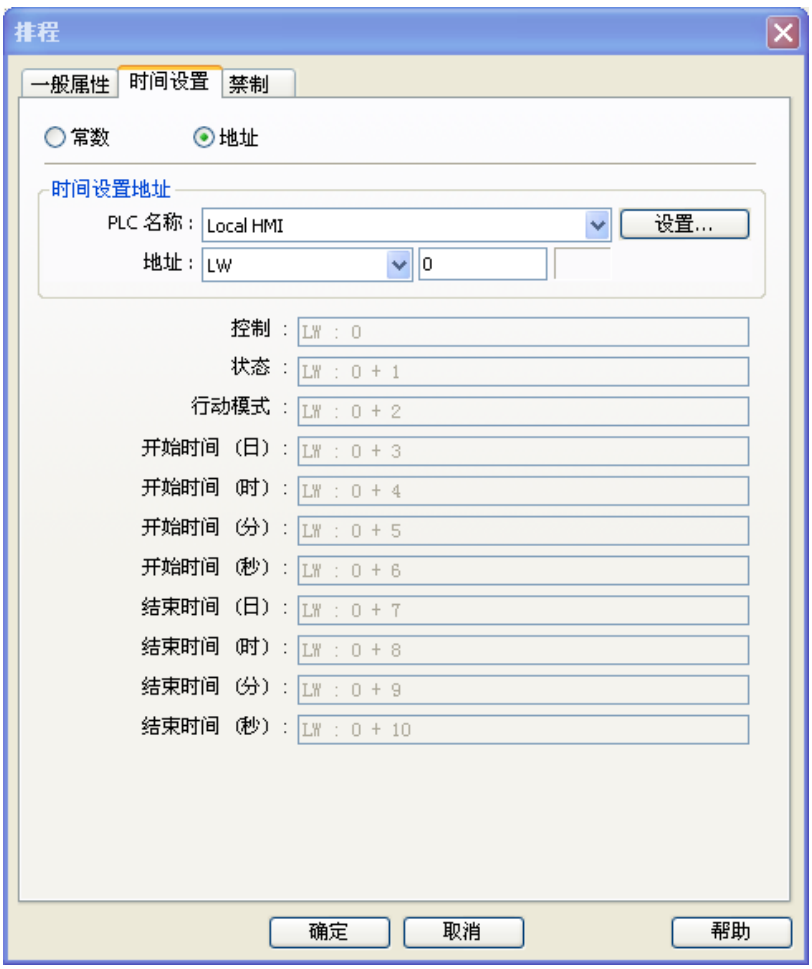


[开始]：选择开始的时间和日期，当“设定为单一日期”停用时，你可以指定一天以上的日期。

[结束]：当“启用结束动作”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。

■ 时间设定/当用户选择“地址”

当地址被选择后，系统会显示控制地址为字地址型态，如下图，此时用户可经由控制这些字地址来手动设定开始时间及结束时间。



用户只需定义时间设定地址，其余的11个控制地址会自动产生并列示出来。以上资料长度皆以16位为例。

a. 控制(时间设定地址+0)

当“更新时间地址”位检测为ON(0→1)时,则读出“行动模式”,“开始时间”和“结束时间”

15	0	位元
保留 (0 固定)	0	

位元 00： 更新时间位（0：无动作 1：读取排程时间资料

NOTE

HMI并不会定期地读取时间设定地址的“行动模式”(地址+2)到“结束时间(秒)”(地址+10)里的资料。所以,当排程时间资料改变时,请务必把“控制”中的“更新时间地址”设为ON(0→1)

b. 控制(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“时间读取完成地址”设为ON(0→1),同样的,若输入的时间资料不正确,“错误通知地址”将会同时被设为ON(0→1)。

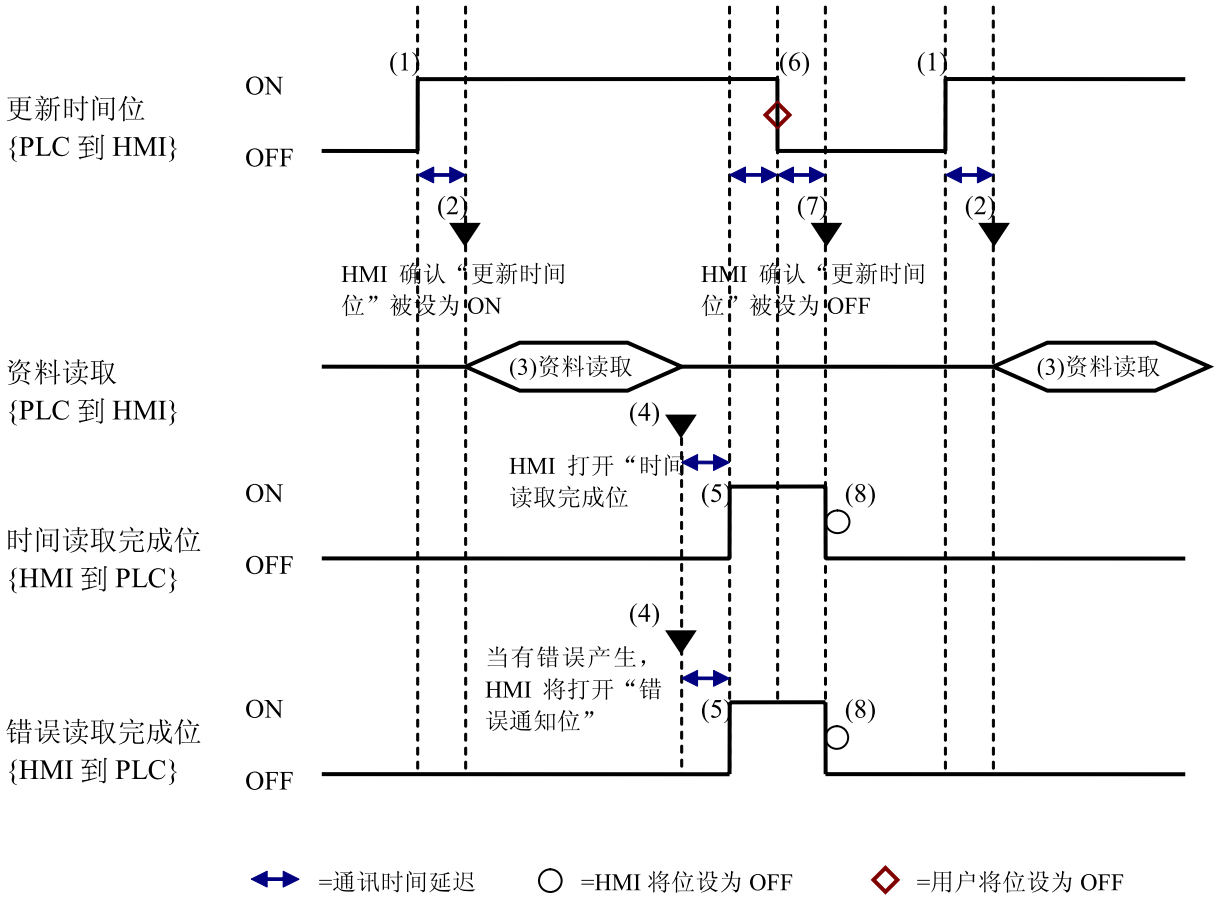
15	02	01	00	位元
保留 (0 固定)		0	0	

位元 00： 时间读取完成位（0：还没开始或是正在读取时间资料；1：时间料读取完成）

位元 01： 错误通知位（0：时间资料被正确更新；1：时间资料中包含错误）

NOTE

一旦发现“时间读取完成位”被触发,请务必把“控制”中的“更新时间地址”设为OFF(1→0)。一旦这个位设为OFF(1→0),则“状态”中的“时间读取完成位”及“错误通知位”将同时被设为OFF(1→0)。



c. 行动模式（时间设定地址+2）

启用或停用“结束时间动作设定”和“单一日期指定模式”，不管“结束时间动作设定”的状态是如何，所有的时间资料（“时间设定位”中的11个字组地址）会被读取。

15 … 02	01	00	位元
保留 (0 固定)	0	0	

位元 00： 结束时间动作设定（0：停用 1：启用）
位元 01： 单一日期指定模式（0：停用 1：启用）

NOTE

若“结束时间动作设定”输入0(停用)，仍会读取结束时间资料但忽略其内容若“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)

指定触发开始/结束动作的日期。

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

位元 00：星期日(0：无；1：指定)
位元 01：星期一(0：无；1：指定)
位元 02：星期二(0：无；1：指定)
位元 03：星期三(0：无；1：指定)
位元 04：星期四(0：无；1：指定)
位元 05：星期五(0：无；1：指定)
位元 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过上面的范围，将会产生错误。

NOTE

用户所输入的时间资料为16bit unsigned格式，系统不接受BCD格式的时间资料。结束时间取决于“行动模式”(地址+2)设定。同样的，“结束时间动作设定”(位元00)有效与否取决于“单一日期指定模式”(位元01)的使用

单一日期指定模式	使用	不使用	
结束时间动作设定	使用	使用	不使用

■ 禁制

[使用禁制功能]

启用时: 在执行开始动作前HMI将读取该位状态, 若为ON, 则略过此次开始及结束动作(若存在), 反之则正常执行设定动作。

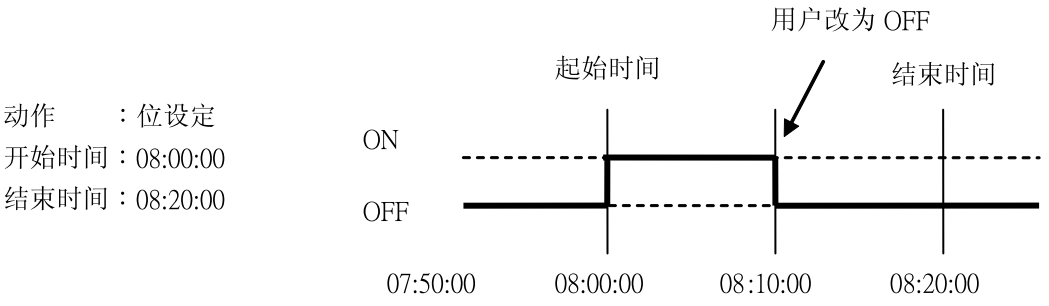
[声音]

启用时: 在执行开始及结束动作(若存在)同时播放指定音效。

使用排程的限制:

■ 最多可注册32个 "排程" 元件

■ 时间排程的特性为一次动作。当开始时间到达时, 特定的设备地址会被写入一次, 这个写入的动作将不会重复。



■ “开始/结束写入数值” 和 “禁止操作位” 只会在执行开始动作前读取一次。所以当开始动作执行后，就算再去改变 “禁止操作位” 状态或 “结束写入数值” 都无法改变结束动作的执行与否及写入数值。另外，为了读取 “开始/结束写入数值” 和 “禁止操作位” 资料，起始动作可能因数据通讯而有少许延迟。

■ 当用户改变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。假如编辑的元件位于新范围中，则开始动作会被执行。假如结束动作未被设定，系统无法确认新范围，则这个动作将不会被执行。

■ 当相同的起始和结束时间出现在多个排程元件中，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“时间设定”指定为“地址”且指定的开始时间和结束时间超过时间的合法的范围，则设定的时间可能不会正确的运作。而且，不能使用BCD编码数据当成输入值。

■ 当“时间设定”指定为“地址”，排程元件直到第一次成功更新时间资料，才开始运作。

13.28 计时器(Timer)

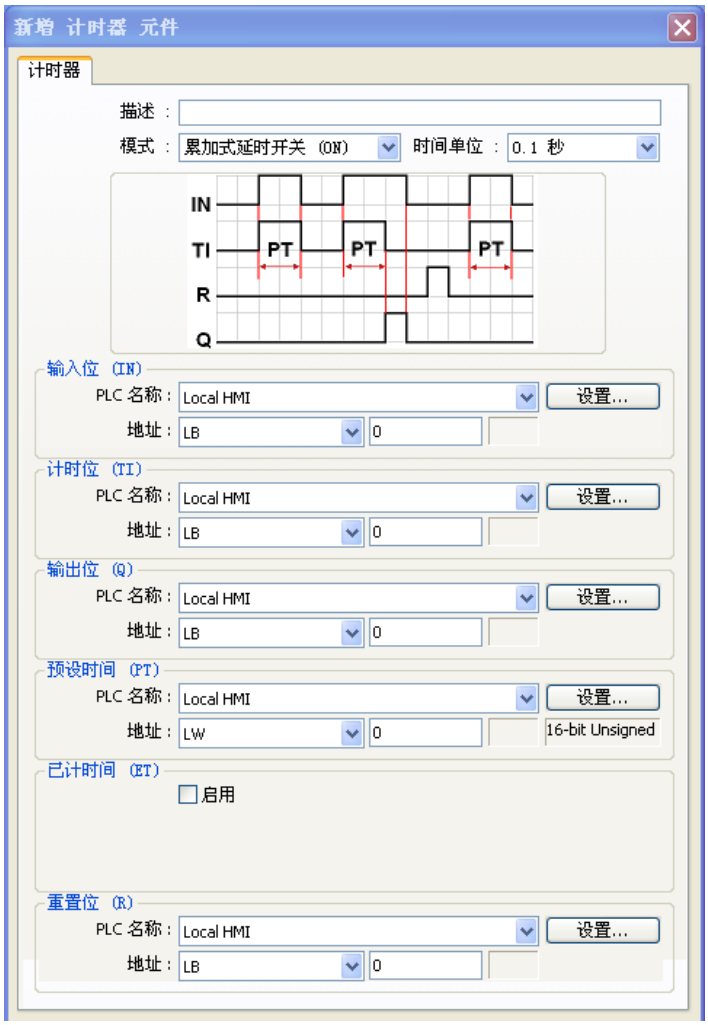
概要

使用计时器的变量来启用计时器功能,计时器的变量组合包含下列六项变量:

计时器变数	变量类型	叙述
输入位(IN)	bit位变量	计时器总开关
计时位(TI)	bit位变量	计时开始时设ON
输出位(Q)	bit位变量	计时结束时设ON
预设时间(PT)	word字变量	设定计时器时间数值
已计时间(ET)	word字变量	显示计时器目前已计时间
重置位(R)	bit位变量	将目前计时器已计时间（ET）归零

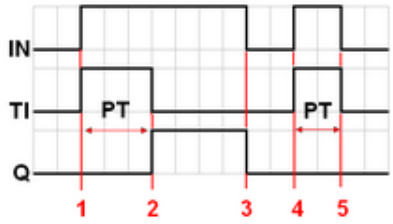
设定

按下工具条上的“计时器”按钮后即会出现“计时器元件属性对话框”,正确设定各项属性后按下确认键,即可新增一个“计时器”元件,参考下图:



模式

[延时开关 (ON delay)]



输入位(IN):计时器的总开关。

计时位(TI):计时开始时设ON。

输出位(Q):计时结束后设ON。

预设时间(PT):设定计时器时间数值。

已计时间(ET):显示计时器目前已计时间。

如上图例子:

Point 1: 当输入位IN设ON时, 计时位TI被开启, 已计时间ET开始计数, 输出位Q保持OFF。

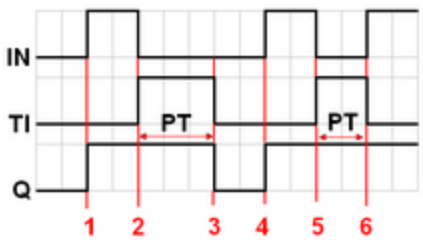
Point 2: 当已计时间ET等于预设时间PT时, 计时位TI被关闭, 同时输出位Q被开启。

Point 3: 当输入位IN设OFF时, 输出位Q被关闭, 同时已计时间ET归零。

Point 4: 当输入位IN设ON时, 计时位TI被开启, 已计时间ET开始计数, 输出位Q保持OFF。

Point 5: 当在已计时间ET到达预设时间PT 的数值前设输入位IN为OFF, 计时位TI将被关闭, 同时已计时间ET归零。

[延时开关(OFF delay)]



输入位(IN):计时器的总开关。

计时位(TI):计时开始时设ON。

输出位(Q):计时结束后设ON。

预设时间(PT): 设定计时器时间数值。

已计时间(ET): 显示计时器目前已计时间。

如上图例子,

Point 1: 当输入位IN设ON时, 计时位TI保持OFF, 输出位Q被开启, 已计时间ET归零。

Point 2: 当输入位IN设OFF时, 计时位TI被开启, 输出位Q保持ON, 已计时间ET开始计数。

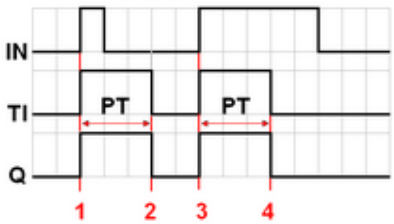
Point 3: 当已计时间ET等于预设时间PT时, 输出位Q和计时位TI被关闭。

Point 4: 当输入位IN设ON时, 计时位TI保持OFF, 输出位Q被开启, 已计时间归零。

Point 5: 当输入位IN设OFF时, 计时位TI被开启, 输出位Q保持ON, 已计时间ET开始计数。

Point 6: 当在已计时间ET到达预设时间PT的数值前设输入位IN 为ON, 计时位TI 被关闭,同时输出位Q保持ON, 已计时间ET归零。

[脉冲启动开关(Pulse)]



输入位(IN): 计时器的总开关。

计时位(TI): 计时开始时设ON。

输出位(Q): 计时结束后设ON。

预设时间(PT): 设定计时器时间数值。

已计时间(ET): 显示计时器目前已计时间。

如上图例子,

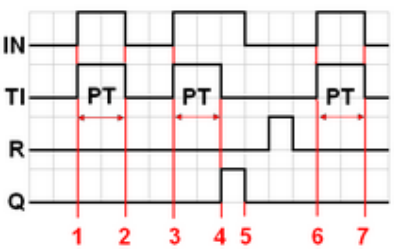
Point 1: 当输入位IN设ON时, 计时位TI和输出位Q同时被开启, 已计时间ET开始计数。

Point 2: 当已计时间ET等于预设时间PT时, 输出位Q和计时位TI同时被关闭(因为在计数同时已先将输入位IN设OFF, 所以已计时间ET将被自动归零)。

Point 3: 当输入位IN设ON时, 计时位TI和输出位Q同时被开启, 已计时间ET开始计数。

Point 4: 当已计时间ET等于预设时间PT时, 输出位Q和计时位TI同时被关闭。

[累加式延时开关(Accumulated ON delay)]



输入位(IN): 计时器的总开关。

计时位(TI): 计时开始时设ON。

输出位(Q): 计时结束后设ON。

预设时间(PT): 设定计时器时间数值。

已计时间(ET): 显示计时器目前已计时间。

重置位(R): 将目前计时器已计时间(ET) 归零。

如上图例子,

Point 1: 当输入位IN设ON时, 计时位TI被开启, 已计时间ET开始计数, 输出位Q保持OFF。

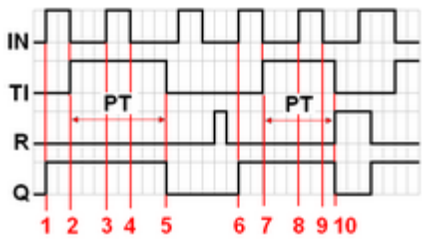
Point 2: 当输入位IN设OFF时, 如果已计时间ET未到达预设时间PT, 计时位TI被关闭, 同时输出位Q保持OFF。已计时间ET保持现在的状态数值。

Point 3: 当输入位IN再度设ON时, 计时位TI被开启, 同时已计时间ET再次由刚刚保持的状态数值开始计数。

Point 4: 当已计时间ET等于预设时间PT时, 计时位TI被关闭, 同时输出位Q被开启。

Point 5: 设输入位IN为OFF, 同时输出位Q也被关闭。(此时设重置位ON 使已计时间ET 归零后再设为OFF。)

[累加式延时开关(Accumulated OFF delay)]



输入位(IN): 计时器的总开关。

计时位(TI): 计时开始时设ON。

输出位(Q): 计时结束后设ON。

预设时间(PT): 设定计时器时间数值。

已计时间(ET): 显示计时器目前已计时间。

重置位(R): 将目前计时器已计时间(ET) 归零。

如上图例子,

Point 1: 当输入位IN设ON时, 计时位TI保持OFF, 同时输出位Q被开启。

Point 2: 当输入位IN设OFF时, 计时位TI被开启, 同时输出位Q保持ON。已计时间ET开始计数。

Point 3: 当输入位IN再度设ON时, 计时位TI和输出位Q保持ON, 同时已计时间ET暂停计数。

Point 4: 当输入位IN再度设OFF 时, 已计时间ET再次由刚刚保持的状态数值开始计数。

Point 5: 当已计时间ET等于预设时间PT时, 计时位TI和输出位Q同时被关闭。(此时设重置位ON使已计时间ET归零后再设为OFF。)

13.29 媒体播放器 (media player)

注意: 第一次使用媒体播放器时, 必须要使用以太网下载工程文件到触摸屏, EB8000将会自动安装媒体播放器的驱动。

概要

媒体播放器功能, 可以播放动态视频文件, 展示企业形象, 介绍操作和维修保养技能, 让现场作业人员更容易了解和接受。

媒体播放器元件不只是播放视频影片, 也提供额外操作功能, 例如: 搜索、放大缩小、音量调整等。此元件只限于MT8000X系列HMI。

设定

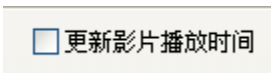
目的: 播放U盘1中example目录下的instruction_video.avi影片文件 (完整路径为: [USB1]/example/instruction_video.avi) 。

从元件菜单中选取 “媒体播放器” 选项或触控工具条上的 “媒体播放器” 按钮 , 即会出现 “新增媒体播放器元件对话框” 。

一般属性页

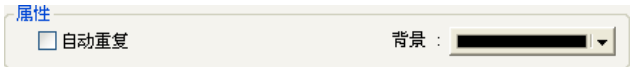
a. 在“控制地址”中勾选“启用”，设定“控制地址”（例如：LW0）

b. 不勾选“更新影片播放时间”



c. 选定“外部装置”U盘1，设定“目录名称”为example

d. 不要勾选“自动重复”，并设定“背景颜色”为黑色



使用预览页：

用户可利用预览功能来检查MT8000是否支持要播放的影片格式。

a. 触控 载入... ，选择事先储存在U盘中的**instruction_video.avi**文件（用户需将影片放在U盘的example文件夹内）。

b. 若影片开始播放则表示此影片被MT8000所支持，此时可以使用 << 及 >> 来往前或往后快进1分钟。

c. 若想开启另一个影片文件请先触控 停止 ，并关闭目前的文件，再回到步骤a。

影片来源：

a. 除PLC外，将连接在MT8000的所有外围设备移除

b. 将事先准备好（内部存有影片文件）的U盘插到MT8000

NOTE 动作a的目的在于确保之后插入（包含影片文件）的U盘为U盘1

开始及停止播放影片

i. 开始播放影片

a. 在 “参数1” 写入0

b. 在 “命令” 写入1, 将开启影片文件并开始播放

c. MT8000在收到命令后将于 “命令” 写入0

NOTE

● 在上述动作b和c之间, 请勿更改 “命令” 、“参数1” 及 “参数2” 内容, 否则可能产生非预期效果。

ii. 停止播放影片

d. 在 “命令” 写入5; 将停止影片播放并关闭文件。

e. MT8000在收到命令后将于 “命令” 写入0。

NOTE

● 在上述动作d和e之间, 请勿更改 “命令” 、“参数1” 及 “参数2” 内容, 否则可能产生非预期效果。

媒体播放器设定

一般设定:

寄存器控制项目

[启用控制地址]

◎ 启用时

a.用户可针对 “媒体播放器” 进行控制并且得到播放信息

b.必须指定一地址用来控制元件行为

◎ 取消使用寄存器控制

此设定无法手动控制影片的播放状态, 在窗口开启时, 系统自动播放影片。

[命令]

控制 “媒体播放器” 的运作模式

◎ 命令（控制地址+0）

[参数1]

相对于特定命令所输入的参数1

◎ 参数1（控制地址+1）

[参数2]

相对于特定命令所输入的参数2

◎ 参数2（控制地址+2）

[状态]

记录文件状态、播放情况及错误代码。

◎ 状态（控制地址+3）

[档案索引]

播放的文件位于制定目录下的索引（以文件名排序, 建议以数字为起始文件名）

◎ 档案索引（控制地址+4）

[开始时间]

影片开始时间（秒）。（通常为0）

◎ 开始时间（控制地址+5）

[结束时间]

影片结束时间（秒）。（影片的时间长度）

◎ 结束时间（控制地址+6）

[播放时间]

更新影片播放时间: 启用时, MT8000每隔 “更新周期”（秒）会将影片已播放的时间写入“播放时间” 寄存器中。

更新周期: “播放时间” 的更新周期, 范围有1到60秒

播放时间: 影片已播放时间（秒）（介于 “开始时间” 与 “结束时间” 之间）

◎ 已播放时间(控制地址+7)

外部装置项目

[SD卡]

选择播放SD卡里的文件

[USB1]

选择播放U盘1里的文件

[USB2]

选择播放U盘2里的文件

[资料夹名称]

影片文件放置的资料夹名称,文件必须被放置于资料夹中且资料夹只能为一层,多层资料夹将不会被接受(例如指定资料夹名称为 “example\ex” 将会出现错误

NOTE

- [资料夹名称]不能为空
- [资料夹名称]不能包含 /\:*?” <>|中的任一字元。

属性项目

[自动重复]

当影片播放结束之后,再次从第一个影片自动重复播放。

[背景颜色]

指定元件背景颜色。

* 预设的寄存器格式为16位无符号数据;当指定寄存器为32位数是,只有较低的16位产生作用,并请将较高的16位固定为0

控制命令

a. 播放索引文件

[命令]=1

[参数1]=文件索引

[参数2]=忽略(应设为0)

NOTE

- 文件以文件名排序。
- 假如找不到文件,则将 “状态” 的第8位设为ON
- 若勾选 “自动重复”,文件播放完毕后自动从头播放同一个文件。

b. 播放上一个文件

[命令]=2

[参数1]=忽略（应设为0）

[参数2]=忽略（应设为0）

NOTE

- 若“档案索引”原本为0, 则从头播放原文件。
- 假如找不到文件, 则将“状态”的第8位设为ON
- 若勾选“自动重复”, 文件播放完毕后自动从头播放同一个文件。

c. 播放下一个文件

[命令]=3

[参数1]=忽略（应设为0）

[参数2]=忽略（应设为0）

NOTE

- 如果找不到文件, 则播放索引值0的文件
- 假如找不到文件, 则将“状态”的第8位设为ON
- 若勾选“自动重复”, 文件播放完毕后自动从头播放同一个文件。

d. 暂停/播放 切换

[命令]=4

[参数1]=忽略（应设为0）

[参数2]=忽略（应设为0）

e. 停止播放并关闭文件

[命令]=5

[参数1]=忽略（应设为0）

[参数2]=忽略（应设为0）

f. 从指定位置开始播放

[命令]=6

[参数1]=目标位置（以秒为单位）

[参数2]=忽略（应设为0）

NOTE

- 参数1(目标位置)应小于结束时间, 若超出结束时间则有结束时间前1秒开始播放。

g. 往后跳跃

[命令]=7

[参数1]=目标位置(以秒为单位)

[参数2]=忽略(应设为0)

NOTE

- 从目前时间往后跳跃指定秒数开始播放, 若超出结束时间则由结束时间前1秒开始播放。

h. 往前跳跃

[命令]=8

[参数1]=目标位置(以秒为单位)

[参数2]=忽略(应设为0)

NOTE

- 从目前时间往前跳跃指定秒数开始播放, 若超出开始时间则由开始时间播放。

i. 音量设定

[命令]=9

[参数1]=音量(0~128)

[参数2]=忽略(应设为0)

NOTE

- 预设最大音量(128)

j. 设定影像放大倍率

[命令]=10

[参数1]=影像大小(0~16)

[参数2]=忽略(应设为0)

NOTE

- [0]: 适合元件大小
- [1~16]: 设定值除以4即为放大倍率; 例如设定1, 则放大倍率为1/4（25%）。

k. 状态（控制地址+3）

15	09	08		02	01	00	位元
保留 （0 固定）	0	0			0	0	

位元 00： 档案开启位（0：未开启档案；1：已开启档案）
位元 01： 档案播放位（0：未播放档案；1：档案播放中）
位元 08： 错误命令位（0：正确命令格式；1：错误命令或参数）
位元 09： 档案错误位（0：档案格式及读取正确；1：档案格式或读取错误）

当MT8000正在播放影片, 则“档案开启位”的位0及“档案播放位”的位1将被同时设定为ON（0→1）。

NOTE

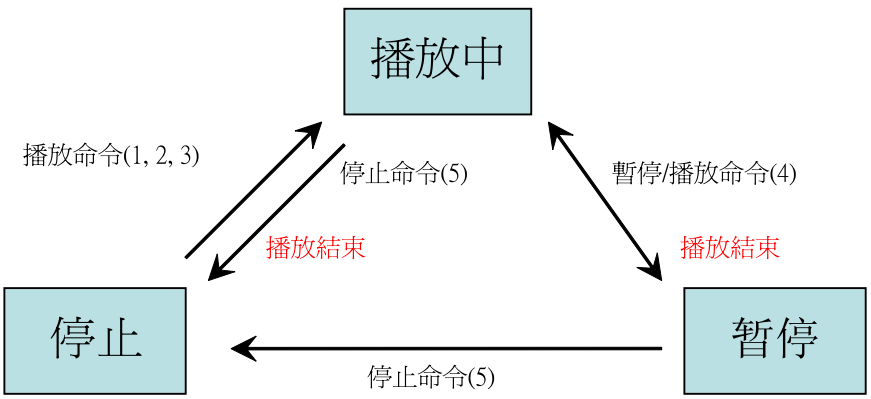
1.若在播放过程中发现文件格式错误或任何U盘错误（例如: U盘被拔出）, 则“档案错误位”的位9将被设定为ON（0→1）。

2.参考一下“媒体播放器”之状态转换图可知:

“停止”时状态=0

“暂停”时状态=1

“播放”时状态=3



◎ 用户仅应透过设定 “命令”，“参数1” & “参数2” 来操作元件，并将其他位视为只读。

限制

■ 用户需注意MT8000上同一时间只能有一个影片文件被开启；

■ 假如用户没有启用 “控制地址” 且没有设定 “自动重复”，则系统会自行将影片关闭；

■ 当没有启用“控制地址”时,元件生成后自动到指定目录下找寻第一个文件(以文件名排序)开始播放;

■ 当影片可以使用媒体播放器的预览功能，表示触摸屏支持此影片格式并且可以播放，若是在触摸屏上有播放品质不佳的状况请调整影片的解析度、编码方式及分辨率；

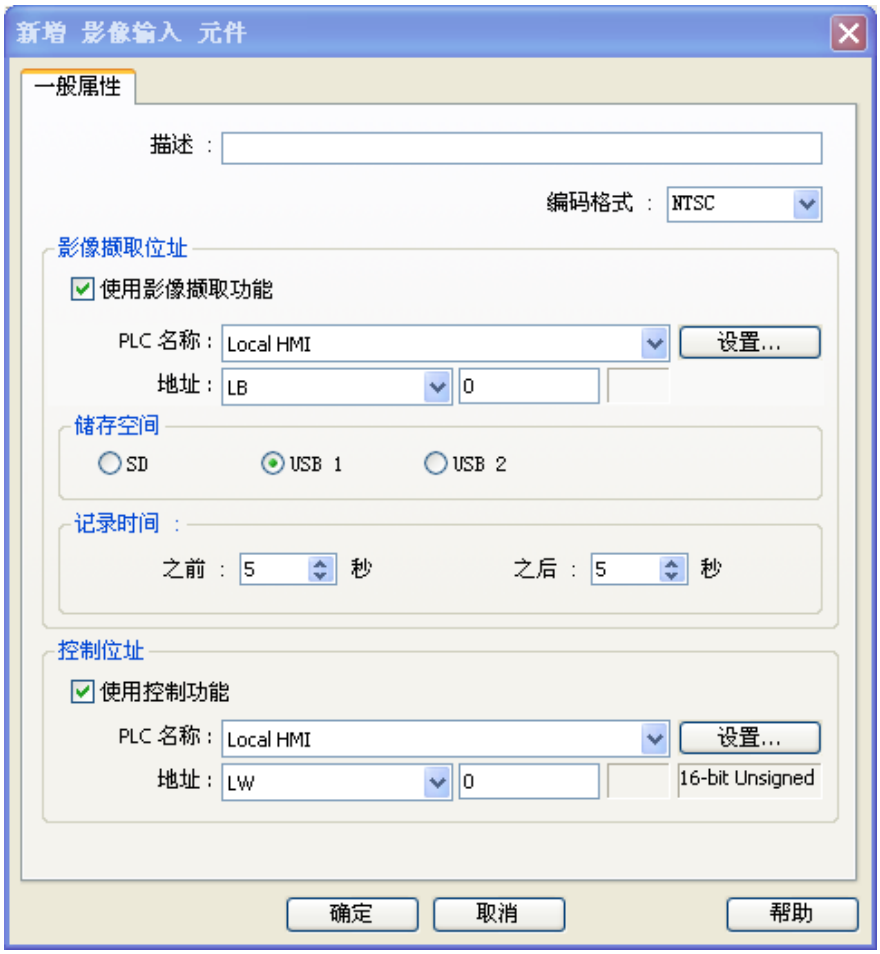
■ 支持的文件格式有：mpeg4，xvid，flv…等等。

13.30 影像输入（Viedo Input Port）

影像输入功能主要适用于MT8000X系列带有Viedo视频接口的触摸屏，可以将现场影像通过摄像头传输到HMI上显示。并可将指定某一刻前后的画面截屏存储。

设定

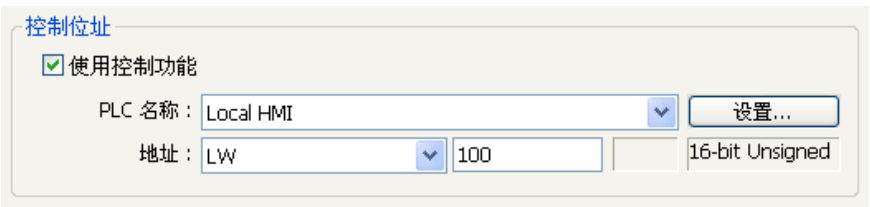
触控工具条上的“影像输入”按钮后即会出现“影像输入元件属性对话框”，正确设定各项属性后按下确认键，即可新增一个“影像输入”元件，参考下图：



[控制地址]

定义：勾选“使用控制功能”启用外部影像输入控制；

说明：



例如：指定控制地址为LW100

A. 用户可借由设定“控制地址+0”来启动/停止影像输入：

[LW100] = 0 →停止播放

[LW100] = 1 →开启影像输入信道 1 影像输入并显示于屏幕上

[LW100] = 2 →开启影像输入信道 2 影像输入并显示于屏幕上

[LW100] = 3→开启影像输入信道 1 影像输入但不显示于屏幕上（仍可执行影像撷取功能）

[LW100] = 4→开启影像输入信道 2 影像输入但不显示于屏幕上（仍可执行影像撷取功能）

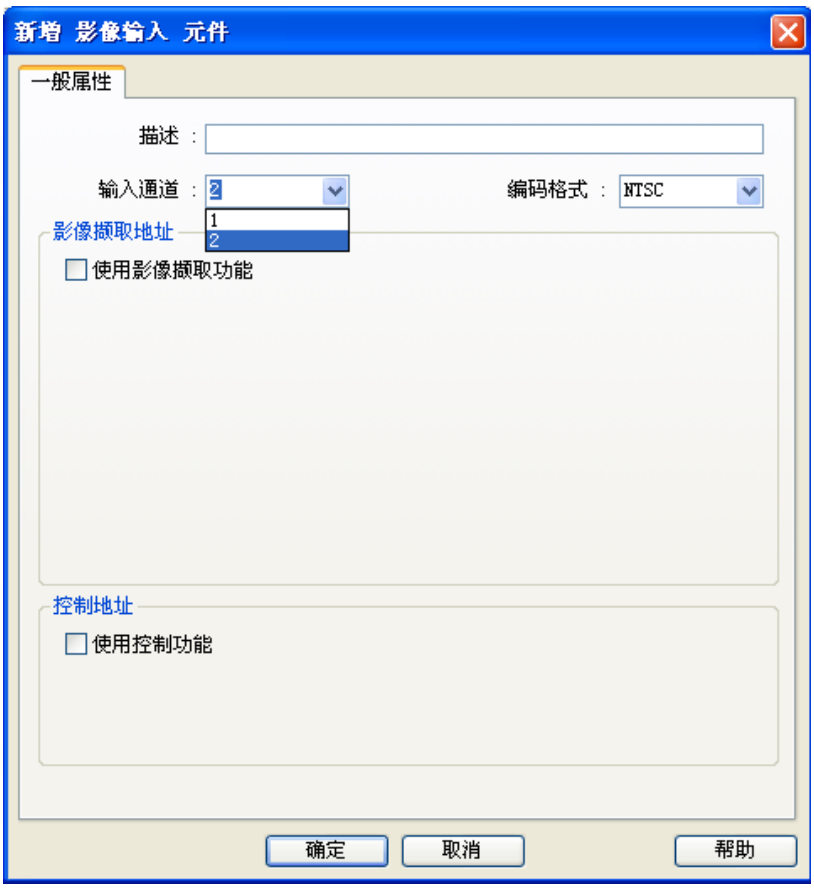
B. 用户可借由设定“控制地址+1”对影像显示做额外控制

[LW101] = 1 →暂停/继续播放

C. 用户在变更“控制地址+0”的值后，系统将保留变更后的值

D. 用户在变更“控制地址+1”的值后，系统将在执行对应命令结束后将其清除为0

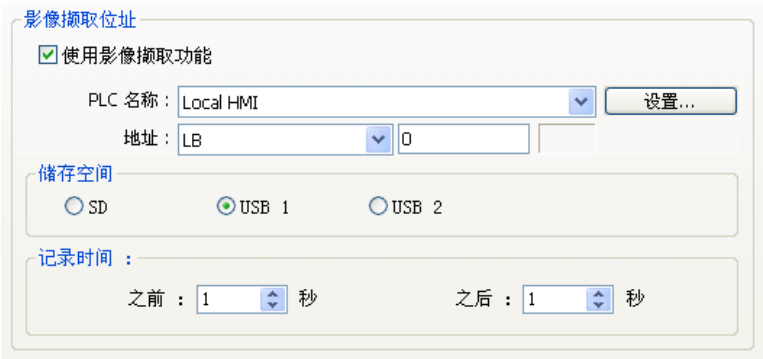
E. 若不启用“使用控制功能”，系统将自动播放“输入通道”指定的影像输入。



[影像撷取地址]

定义：勾选“使用影像撷取功能”，则撷取输入影像画面为图片并进行保存；

说明:



A. 影像撷取地址: 触发系统截取图片的控制地址

B. 储存空间: 选择图片存储设备: SD卡, U盘1或U盘2

–影像输入通道1撷取图片将储存在存储设备的VIP1文件夹目录中, 影像输入通道2撷取图片则存储在VIP2文件夹目录中。

C. 记录时间: 设定撷取画面之时间范围

–最大撷取范围为 “影像撷取地址” 触发时间的前/后10秒

–系统每秒截图一次

–图片文件命名规则:

“影像撷取地址” 触发前后: YYYYMMDDhhmmss.jpg

“影像撷取地址” 触发当时: YYYYMMDDhhmmss@.jpg

以上图为例, 设定记录时间为前后5秒, 当 “影像撷取地址” 由OFF→ON时, 系统将从触发时间点起算, 每秒1张, 截取前后5秒之输入画面, 共有11张图片。

注意:

1. 影像输入元件仅可在有搭配影像输入功能的MT8000X系列上使用;

2. 系统中任何时间点只能有一组影像输入通道开启影像输入;

3. 影像撷取功能不受暂停播放控制之影响, 所截取的图片仍是外部影像输入的即时画面;

4. 推荐的格式类型和解析度;

	1:1	50%
NTSC	720 x 480	360 x 240
PAL	720 x 576	360 x 288

此功能只支持NTSC和PAL格式类型。

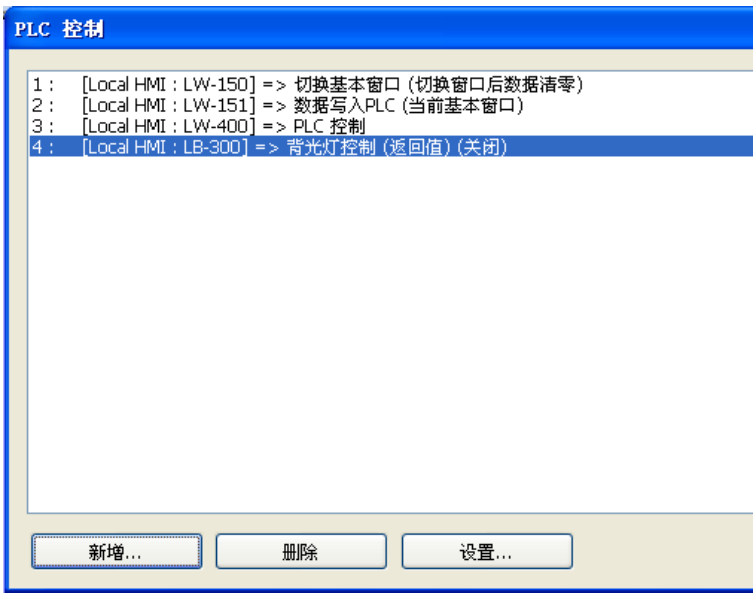
13.31 PLC控制元件 (PLC control)

概要

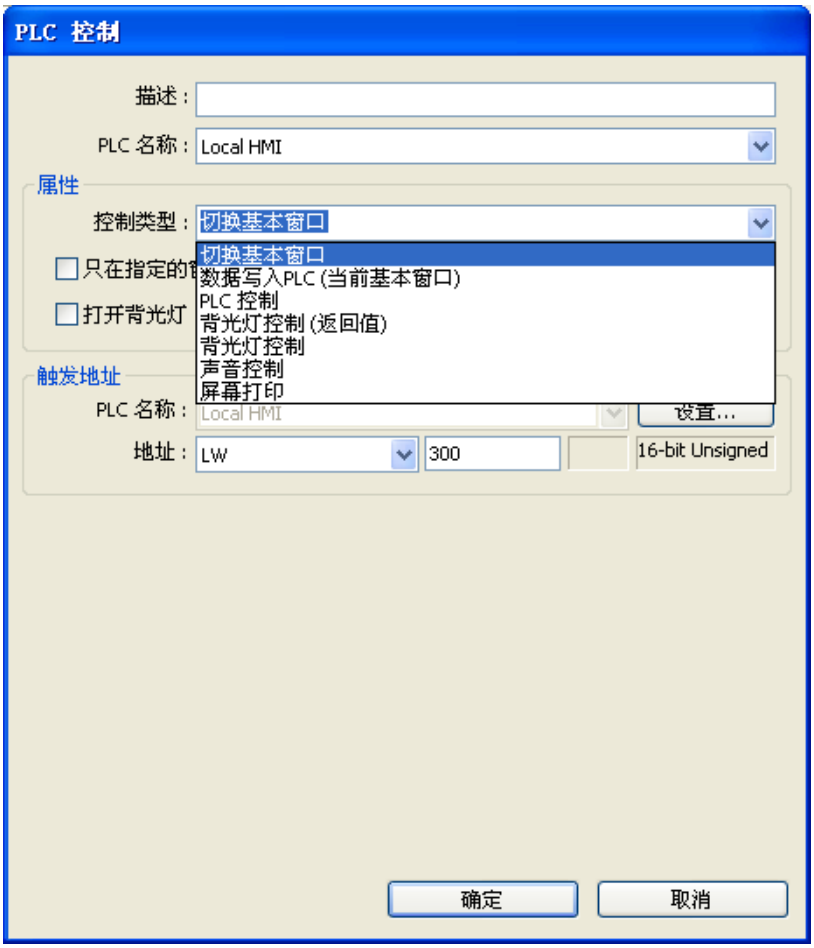
当相应的控制条件被触发时,“PLC控制”元件能激活一个特定的动作。

设定

触控工具条上的“PLC控制”按钮后即会出现“PLC控制元件管理对话窗”,接着可触控“新增 …”按键,并利用出现的“PLC控制元件设定对话窗”正确设定元件的各项属性,最后触控确认键即可新增一个“PLC控制”元件,参考下图。



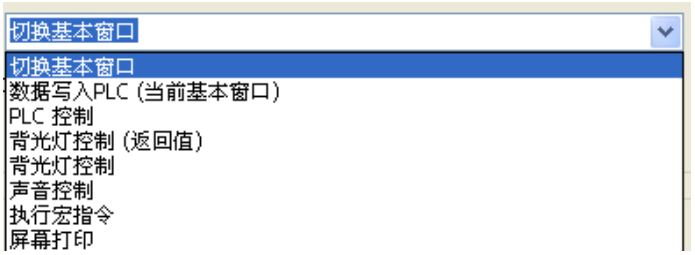
下图为触控“新增…”按键后所出现的“PLC控制元件设定对话窗”。



属性项目

[控制类型]

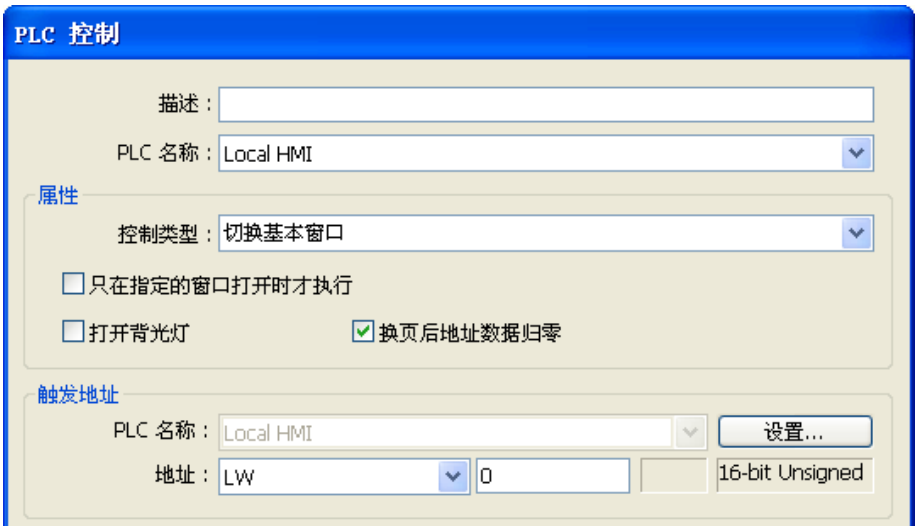
选择PLC控制元件的控制类型,可选择项目如下图。



a.“切换基本窗口”

切换基本窗口功能。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,且元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,EB8000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。

例如:当使用32-bit Unsigned地址做为触发地址时,系统将会写入值到“触发地址+2”的地址。

变量资料类型	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[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[换页后打开背光灯],则在切换窗口成功后会自动开启背光灯。

b.“数据写入PLC (当前基本窗口)”

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,EB8000将利用[触发地址]中所设定的地址开始,连续四个寄存器中的数据,决定数据传输类型、数据传送数量、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] = DM100,即表示使用DM100~DM103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,共有四种类型类型 当寄存器被写入新的数据时,MT8000 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	存放欲传输资料大小,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 OMRON PLC 为例,如果此时设定的[触发地址]为 DM100,而在寄存器[触发地址] + 2 也就是 DM102 中的资料为“5”,则传输的数据来源的起始地址为 DM109,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)或者本地资料寄存器(LW)的起始地址。	以 OMRON PLC 为例,如果此时设定的[触发地址]为 DM100,而在寄存器[触发地址] + 3 也就是 DM103 中的资料为“100”,则传输过程中操作的 RW 或 LW 的起始地址为 RW100 或 LW100。

举例说明如下:

假如现在需要使用“PLC控制”的功能,将OMRON PLC中从DM100起始的16 words的资料,传输到HMI配方内存RW200开始的地址中,实现的方法如下:

i. 首先,假设我们用DM10起始的四个资料寄存器来控制传输,则应该先建立一个PLC控

制元件,选择类型为“PLC控制”,读取地址设定为DM10。

ii. 接下来,应该确定操作数据的大小和地址的偏移量,将DM11设定为16,表示传输资料的大小为16 words;将DM12设定为86,表示数据的来源地址为DM100 (100 = 10+4+86);将DM13设定值为200,表示目标地址为RW200。

iii. 最后,依照数据传输的方向,设定传输类型。应该将DM10设定为1,表示将传输PLC寄存器中的数据到HMI上的RW寄存器中。如果设定DM10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HMI的数据寄存器变成了本地资料寄存器LW。

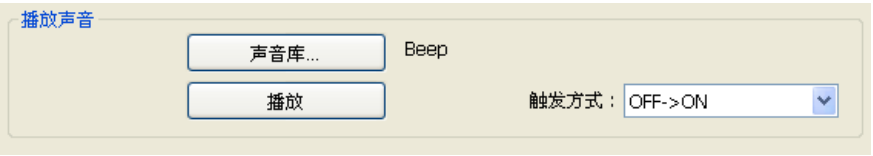
d.“背光灯控制 (返回值)”

当[触发地址]的状态由OFF变为ON时,MT8000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,用户只需触控屏幕,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,MT8000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”



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

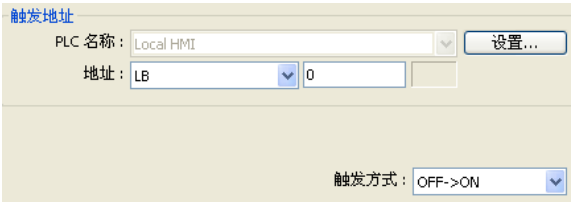
(1) 状态由OFF变为ON (OFF–>ON)

(2) 状态由ON变为OFF (ON–>OFF)

(3) 只需状态改变 (OFF<–>ON)

注意: MT6000i/TK系列产品无音频输出接口, 只支持发出蜂鸣声(Beep), 而具有音频输出接口的MT8000i(部分)/MT8000X和WT3010支持外接音箱, 可播放任意设定之音频。

g.“执行宏指令”



当[触发地址]的状态改变符合触发条件时,“ PLC控制 ”元件将执行指定的宏指令。[触发方式]可以选择:

(1) 状态由OFF变为ON (OFF–>ON)。

(2) 状态由ON变为OFF (ON–>OFF)。

(3) 状态改变时 (OFF<–>ON)。

(4) 只需状态维持在ON(当状态为ON是即执行)。

h.“屏幕打印”

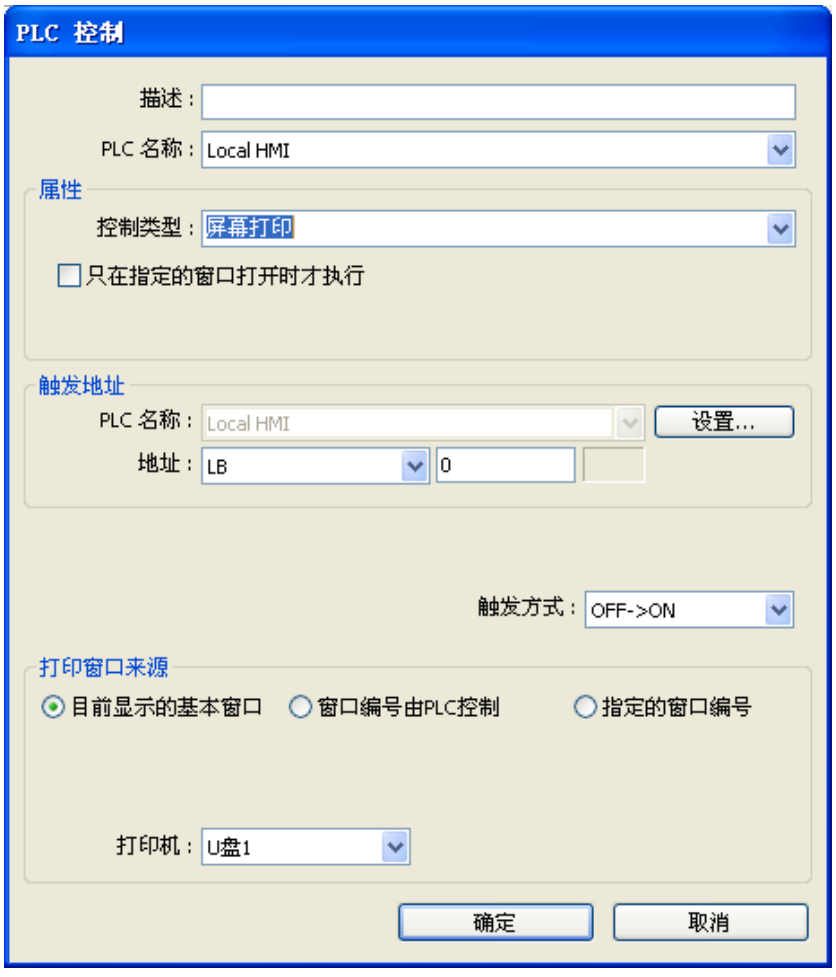
当[触发地址]的状态改变符合触发条件时,“ PLC控制 ”元件将打印指定的画面。触发方式可以选择:

(1) 状态由OFF变为ON (OFF–>ON)

(2) 状态由ON变为OFF (ON–>OFF)

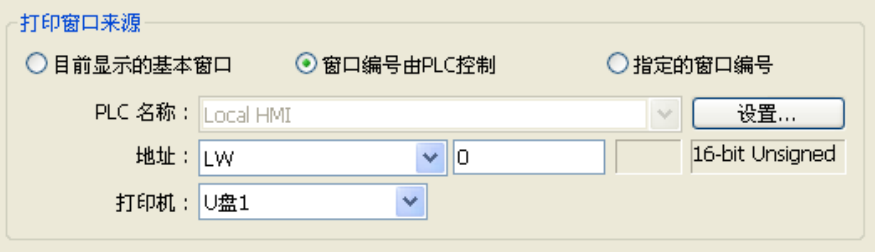
(3) 只需状态改变 (OFF<–>ON)

可以选择要打印的画面,共有三种指定方式,参考下图。

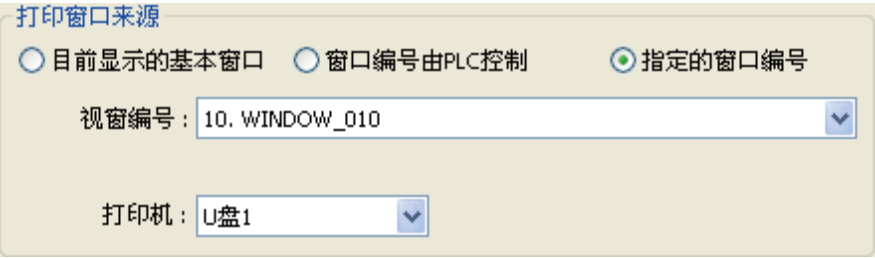


目前显示的基本窗口：“PLC控制”元件将打印目前显示的画面

窗口编号由PLC控制：“PLC控制”元件将利用下面的地址读取数据，此数据即为窗口的编号，如果此窗口存在，则打印这个窗口的内容。



指定的窗口编号：直接指定要打印的窗口，参考下图。



备注：

1. 当指定被打印的窗口不是当前窗口时，系统提供背景打印
2. 当被指定为背景列的窗口，用户请不要将直接窗口或间接窗口指定为此窗口

13.32 系统信息（System Message）

当信息窗口弹出时，所显示的文字内容。可以自定义编辑。

设定	叙述
操作确认提示	要操作受保护的元件时，首先弹出窗口向用户显示此信息，以确认操作
禁止写入命令	当系统寄存器 LB9196((本地 HMI 只支持监视功能)设为 ON 时，显示此信息
允许写入命令	当系统寄存器 LB9196((本地 HMI 只支持监视功能)设为 OFF 时，显示此信息

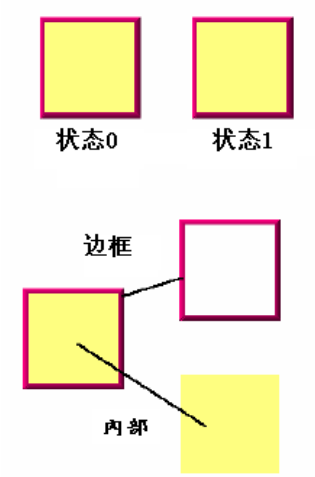
第十四章 向量图库、图片库的建立与使用

EB8000提供向量图库与图片库的使用，增加元件的视觉效果，每个向量图与图片最多可包含256个状态。下文将说明如何建立向量图与图形库。

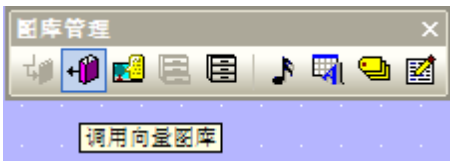
向量图、图片库的使用请参考《第九章 元件一般属性》。

14.1 向量图库的建立

向量图是由直线、矩形、圆等绘图元件所构成的图形；一个完整的向量图可能具有一个以上的状态，每个状态都可包含两个部分：边框(frame)与内部(inner)，参考下图：



元件可以单独选择使用向量图的边框或内部，或两者同时使用。在触控工具条的向量图库按钮，即可进入向量图库管理对话窗，见下图。



[图库]

显示已加入此工程文件的向量图库，要选择使用哪一个向量图库只需点击图库名称即可。

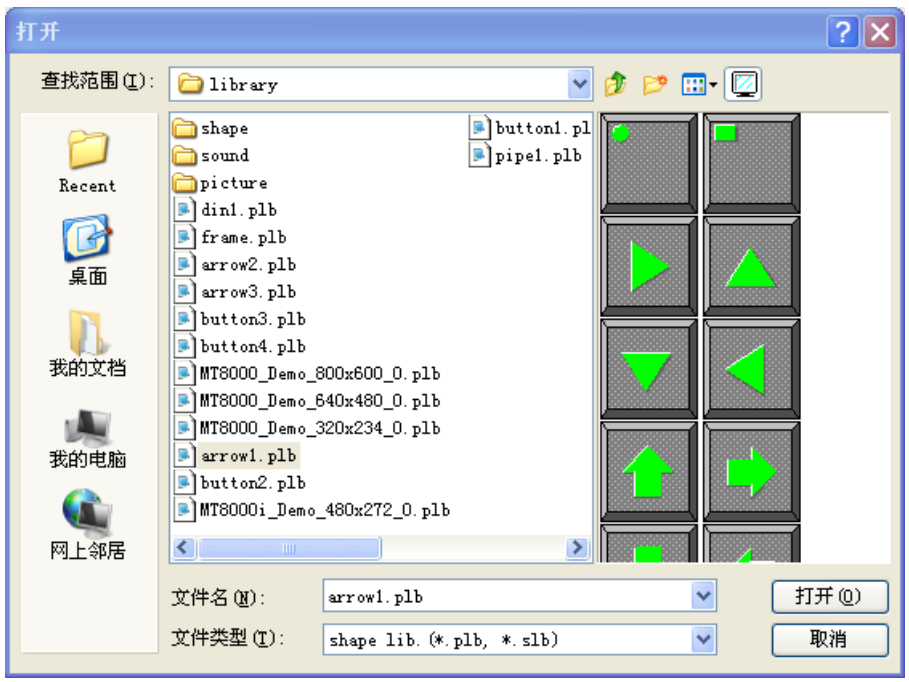
[目前状态]

选择向量图目前要显示的状态，当窗口中未显示向量图时，表示该向量图不存在，或此向量图在目前的状态并未被定义。

[选择图库...]

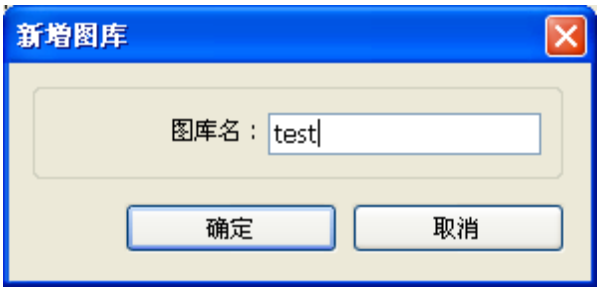
触控按钮后可出现下图的画面，可选择要加入此工程的向量图库。

在窗口的右半部则可先预览图库的内容，再将合适的图库加入。



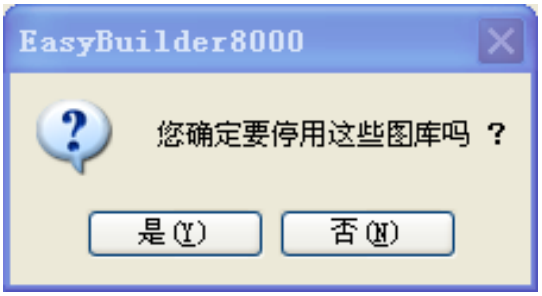
[新增图库 ...]

触控按钮后可出现下图的画面, 可用来增加一个空的向量图库。



[删除图库 ...]

触控按钮后可出现下图的画面, 可将[图库]中显示的向量图库从此工程文件中移除。



[删除所有状态]

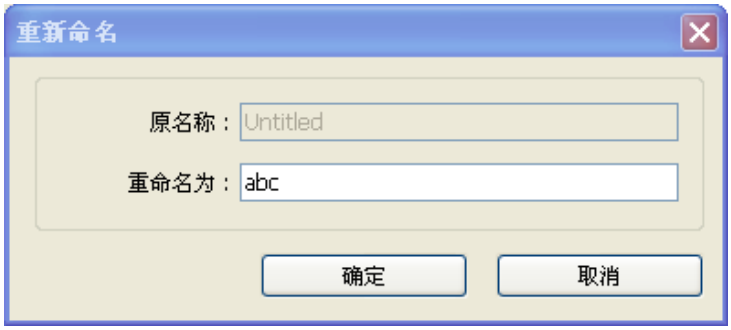
用来删除目前所选择向量图的全部状态。

[删除当前状态]

用来删除目前所选择向量图所显示的状态。

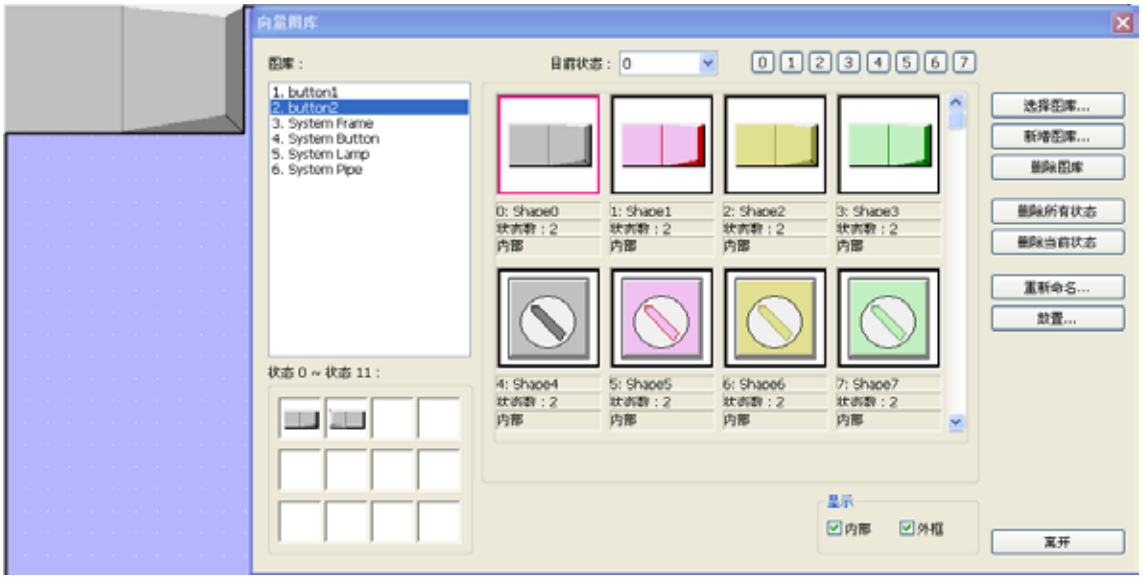
[重新命名...]

触控按钮后将出现下图的画面, 可重新命名目前所选择的向量图。



[放置 ...]

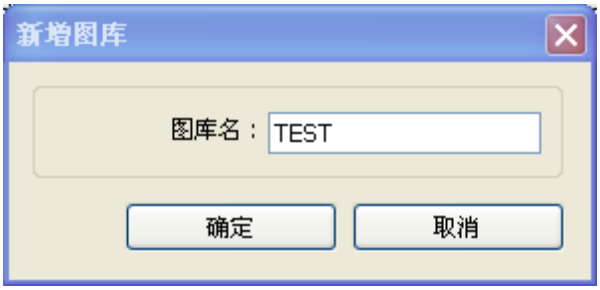
可将目前选择的向量图输出到使用中的窗口上，如下图所示。



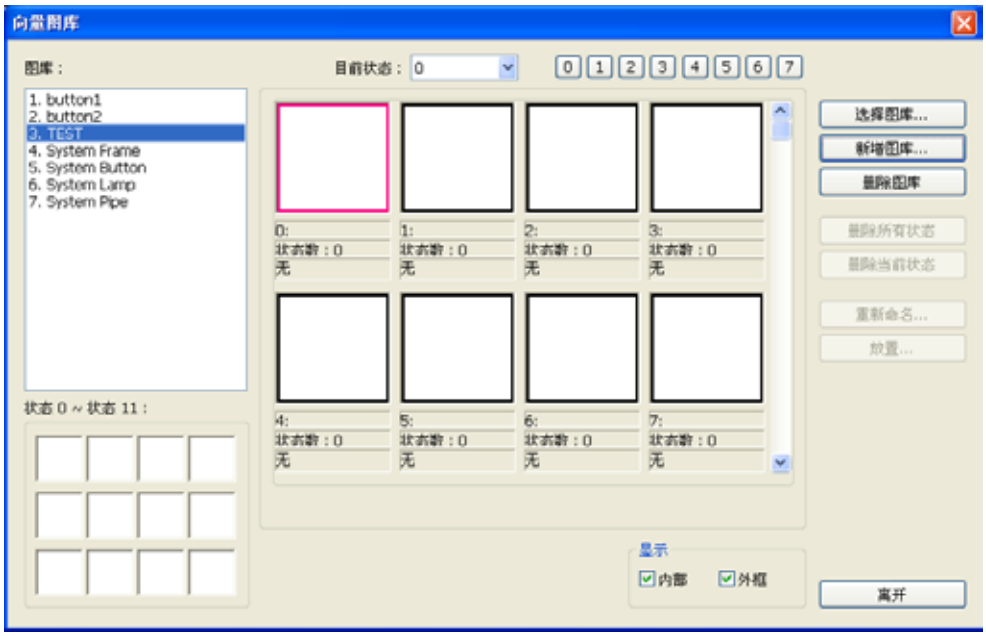
下面说明如何建立一个新的向量图库，并在此图库中加入一个具有两个状态的向量图。

步骤一

触控[新增图库 …]后，在对话窗中输入新的向量图库名称。

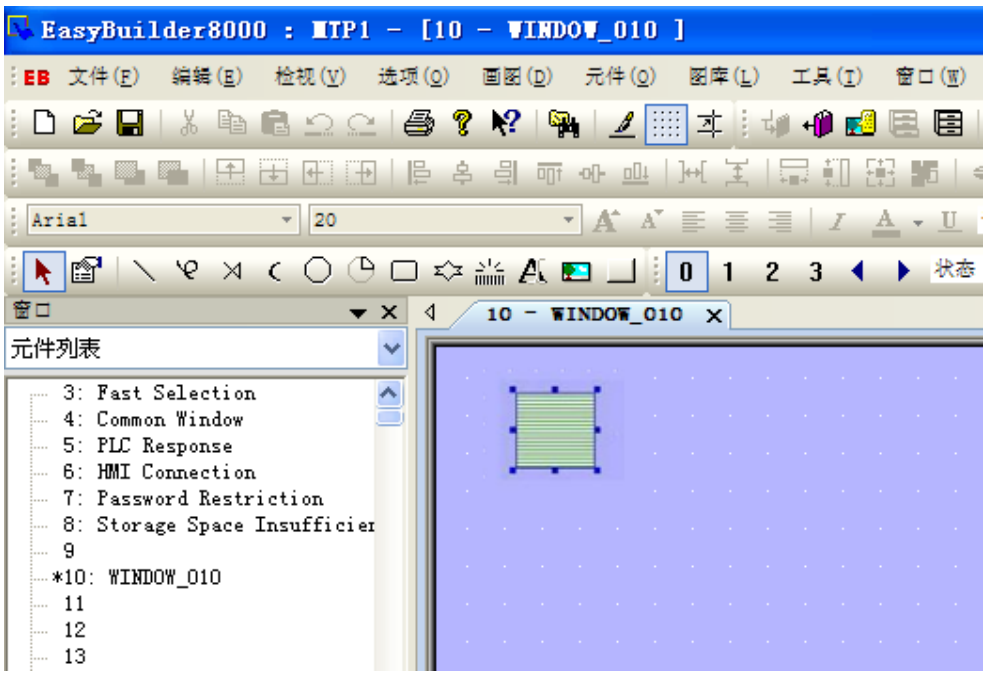


此时可以发现向量图库管理对话窗中增加一个新的向量图库“TEST”，且此新的向量图库中并未包含任何向量图，参考下图。



步骤二

对特定的向量图，加入一个状态。首先使用绘图工具在窗口上绘出需要的图形，并框选要加到向量图库的图形。



接着在框选图形组件后，触控工具条上的“储存至向量图库”按钮，可以得到下面的对话窗：

[图库]

选择目前的图形是要加到哪一个向量图库，目前选择“TEST”。

[描述]

向量图的名称

[图型编号]

选择目前的图形要加到“TEST”向量图库中的哪一个向量图中。

[状态]

选择目前的图形将作为向量图的哪一个状态，此范例是选择状态0。EB8000可支持256个状态。

[边框]

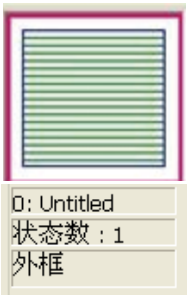
若选中此项目，目前的图形将作为向量图的边框。

[内部]

若选中此项目，目前的图形将作为向量图的内部。

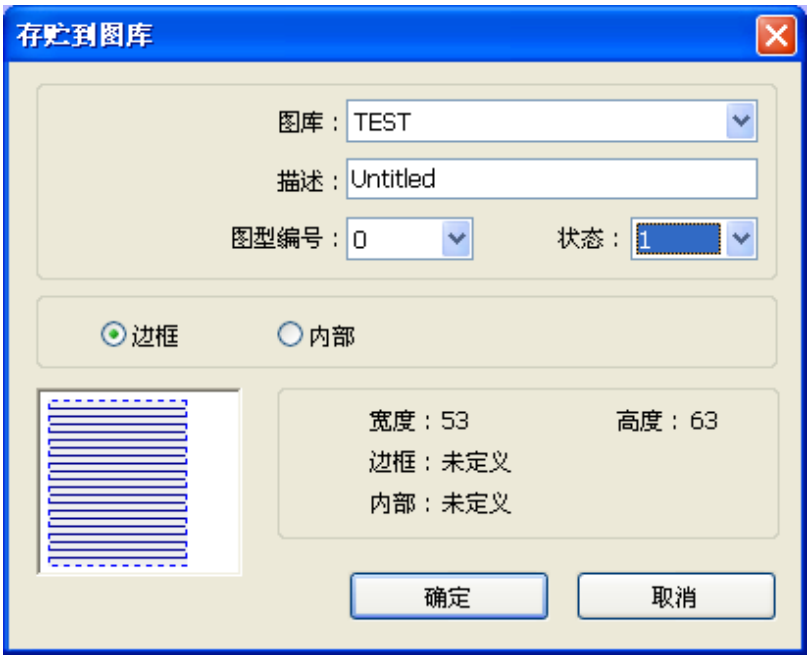
上图的信息也显示“TEST”向量图库中编号0的向量图，目前的状态(state 0)并未定义任何边框与内部。

在触控确认键后可以发现图形已经加入到向量图库中，见下图。由下图也可看出编号0的向量图只具有一个状态，且边框已被定义。

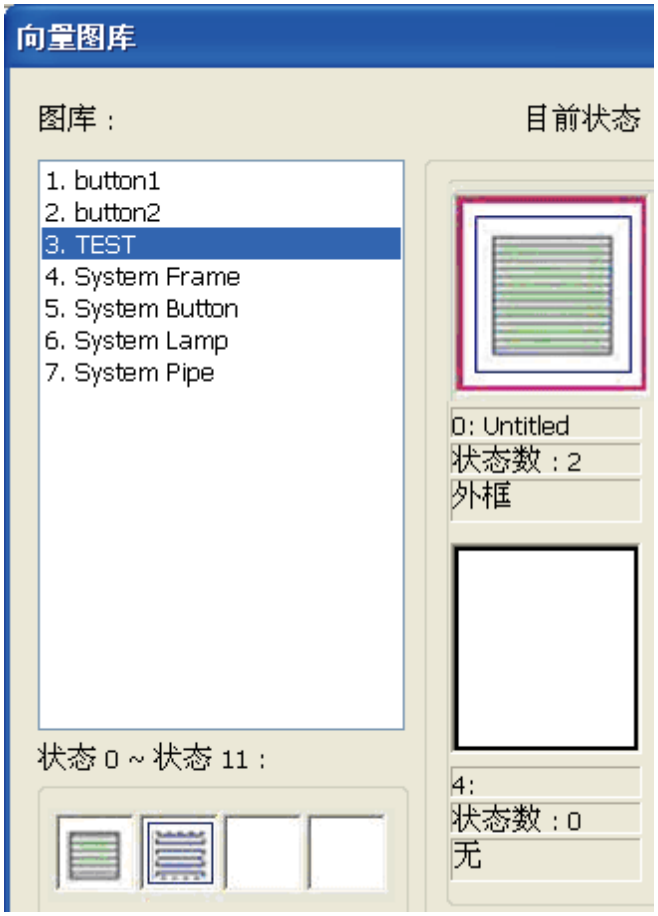


步骤三

使用步骤二相同的方式，但新加入的图形需选择作为状态1，参考下图。

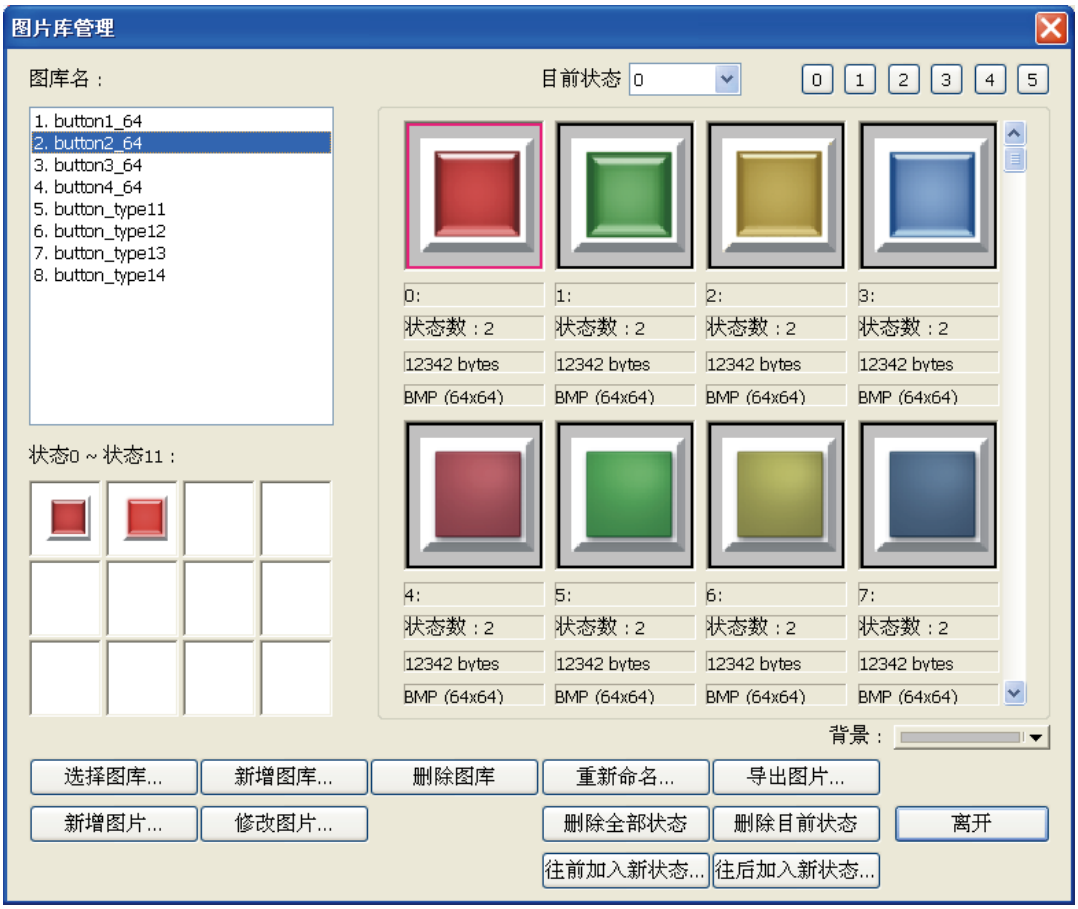


在完成上述的各项动作后，即建立一个完整的向量图，可参考下图：



14.2 图片库的建立

触控工具条上的工作按钮后将出现“图片库管理对话窗”，参考下图。



[图库名]

显示已加入此工程文件的图片库，要选择使用哪一个图片库只需点击图库名称即可。

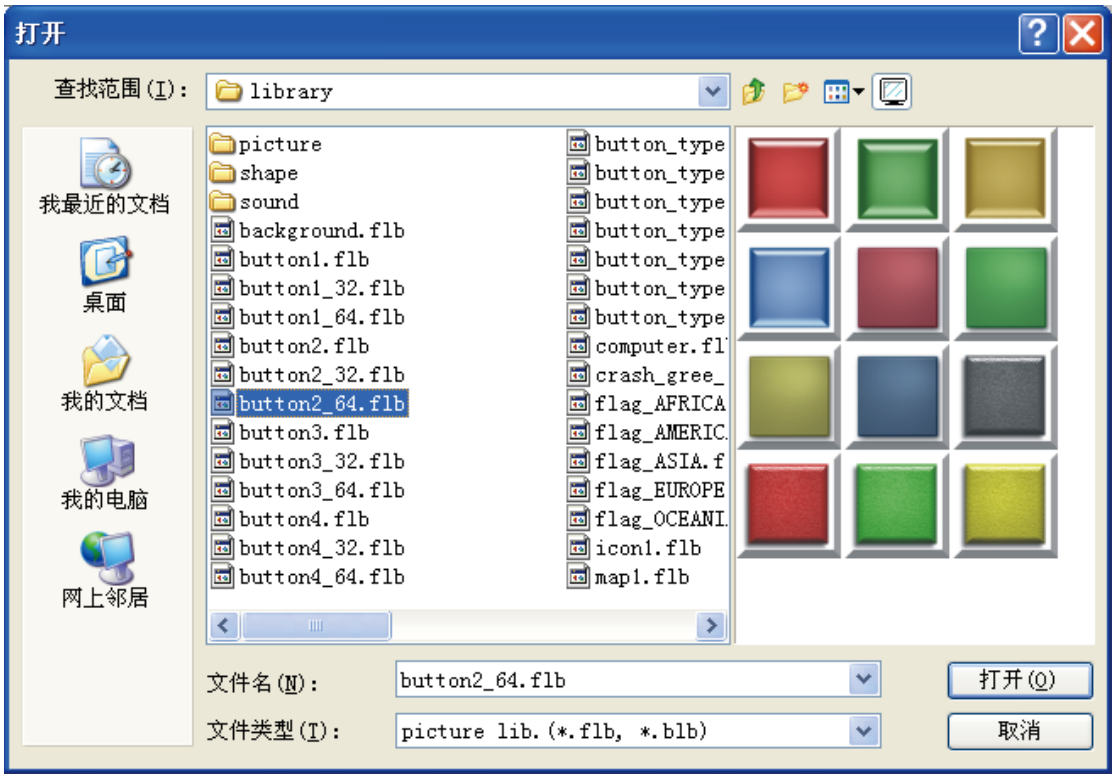
[目前状态]

选择图片目前要显示的状态，当窗口中未显示图形时，表示该图形不存在，或此图形在此状态并未被定义。

[选择图库 ...]

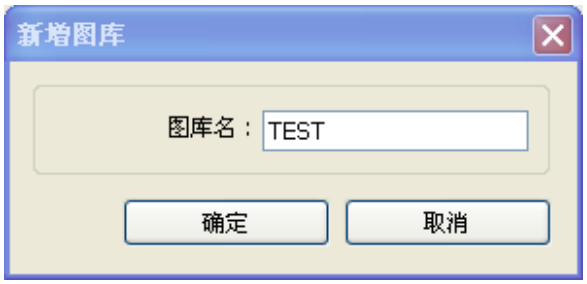
触控按钮后可出现下图的画面，可选择要加入此工程文件的图片库。

在窗口的右半部则可先预览图库的内容，再将合适的图库加入。



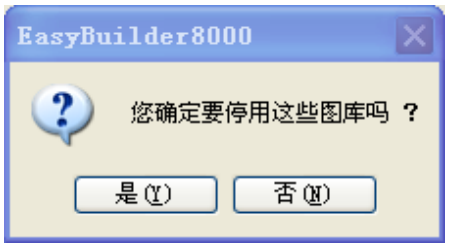
[新增图库 ...]

触控按钮后可出现下图的画面，可用来增加一个空的图片库。



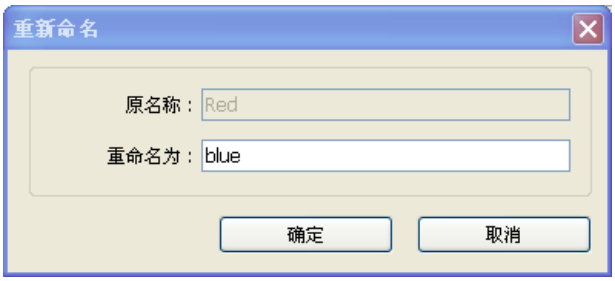
[删除图库 ...]

触控按钮后可出现下图的画面，可将[图库]中显示的图片库从此工程文件中移除。



[重命名 ...]

触控按钮后将出现下图的对话窗，可重新命名目前选择的图形。



[往前加入新状态]

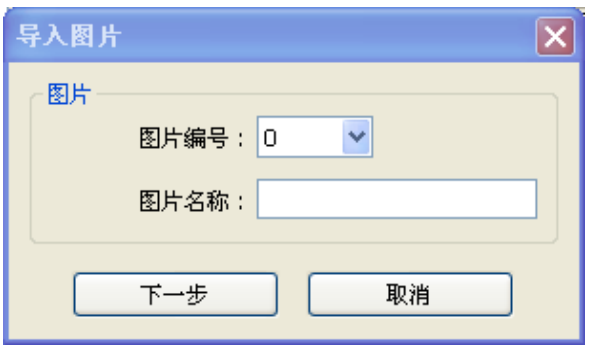
在目前所显示的状态前加入一个新状态。

[往后加入新状态]

在目前所显示的状态后加入一个新状态。

[新增图形]

在图片库中增加一个新的图形。

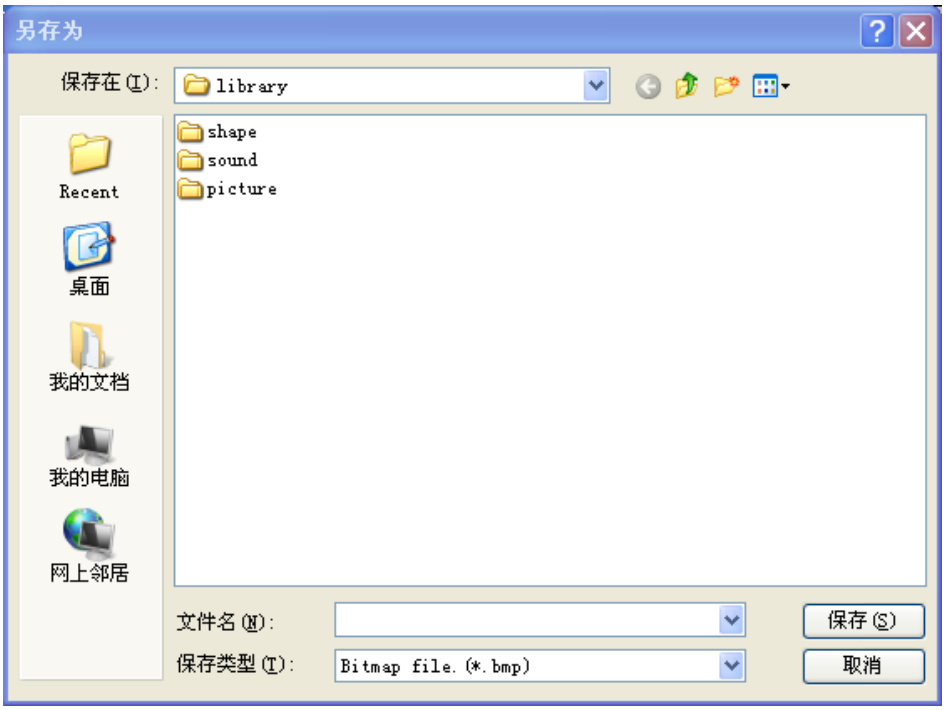


[修改图片]

利用此项功能可以修改图片目前的状态。

[导出图片 ...]

可将目前选择的图形输出到指定的位置，如下图所示，让用户可以获得原始图形。



[删除全部状态]

用来删除目前所选择图片的全部状态。

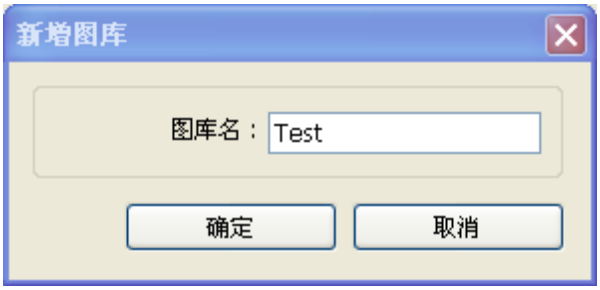
[删除目前状态]

用来删除目前所选择图片所显示的状态。

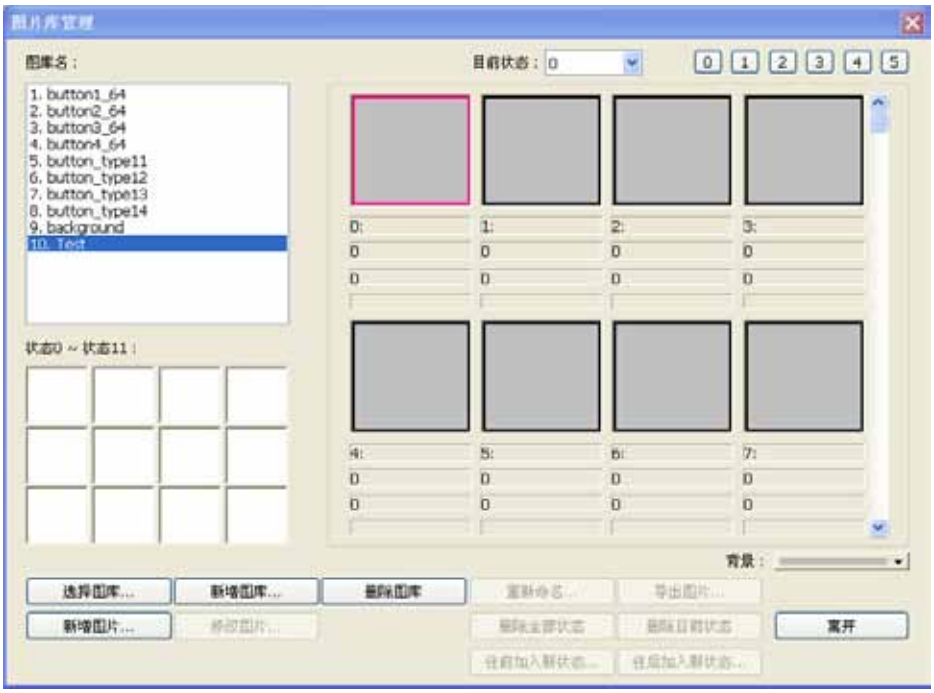
下面说明如何建立一个新的图片库，并在此图库中加入一个具有两个状态的图片。

步骤一

触控[新增图库…]后，在对话窗中输入新的图片库名称。

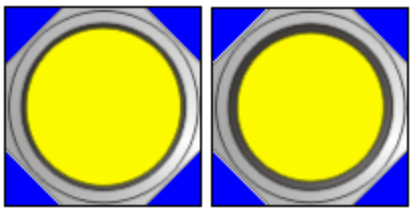


此时可以发现图片库管理对话窗中增加一个新的图形库“Test”，且此新的图形库中并未包含任何图形，见下图。

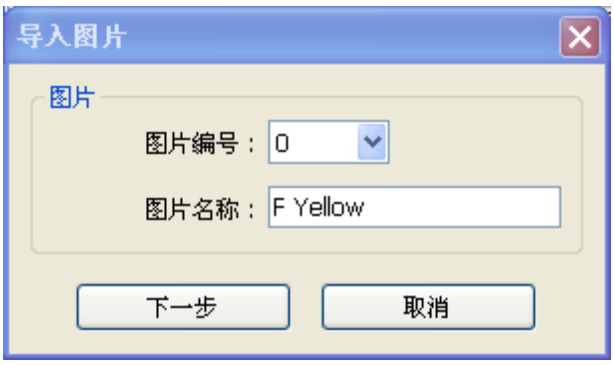


步骤二

先准备好要加入的图片（客户自有图片照片资源，支持BMP、JPG、GIF、PNG等图片格式）；假设现在要将下面的两个图片分别用来表示状态0与状态1。



首先按下[新增图片…]按键，可出现下图的对话窗，选择图片编号，并输入图片名称，如：F Yellow，然后按下[下一步]。



步骤三

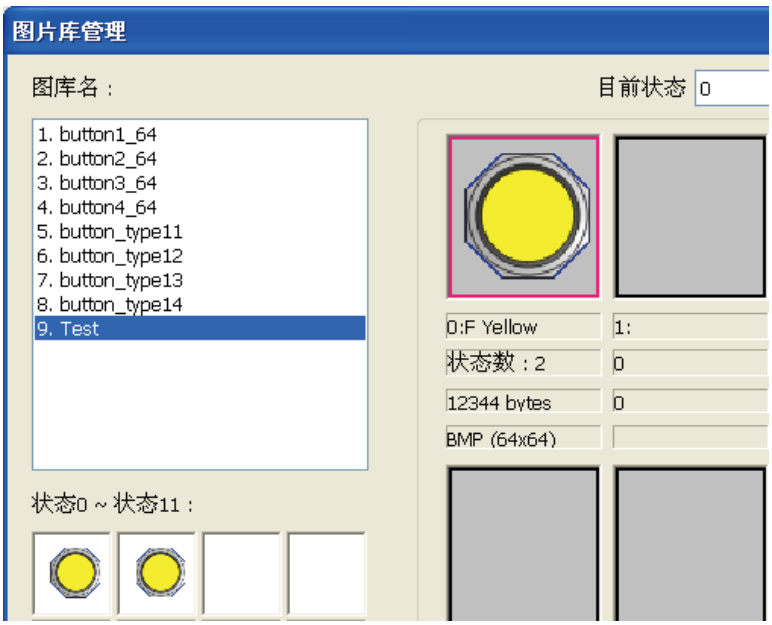
在出现下图的对话窗时，此时需选择状态0的图片来源，并选择正确的透明色，此时选择蓝色RGB(0, 0, 255)为透明色。完成状态0的设定后，因还包含另一个状态，继续执行[下一步]。

要选择透明色首先需先勾选[使用透明色]，接着使用鼠标点击欲作为透明区域的位置（如上图，鼠标点击蓝色区域），此时EB8000会自动显示作为透明色的RGB值，以上图为例，实际显示的图形如下。

步骤四

与上一步骤相同，选择状态1的图形来源并选择正确的透明色。最后触控[完成]即完成包含两个状态图形的建立。

在完成上述的各项动作后，即建立一个完整的图片，可参考下图。这时在图形管理对话窗中可以发现新加入的图形“F Yellow”，由图片信息中也可看出此图片为bitmap形式，且包含两个状态。



注意：对JPG格式图片进行透明处理时，可能会造成在触摸屏上显示产生锯齿，可使用其它格式图片（如BMP）或先在图像处理软件（如Photoshop等）中做透明处理之后，再以GIF或PNG的格式导入（GIF和PNG格式图片本身支持透明效果）；数码照片拍下的图片，一般像素较大，也需要先在图像处理软件处理之后再导入，并以原始尺寸在HMI上显示，以避免造成HMI处理能力负荷加重，影响换页显示速度。

第十五章 文字标签库与多国语言使用

文字标签库一般使用在需要多国语言的环境中，用户可以依照实际需要预先建立文字标签库的内容。在需要使用文字的场合，只需由文字标签库中选择需要的标签即可。

介绍

系统在运作时也会依照设定，显示当时使用语言相对应的文字。EB8000同时支持8种不同语言的文字显示。触控工具条上的工作按钮将出现“文字标签库管理对话窗”，参考下图。

[目前状态]

目前显示的状态，每一个文字标签最多可拥有256个状态（0~255）。状态数量受“语言数量”影响，若用户用8个语言，则256/8=32（状态数），若用户用4个语言，则256/4=64（状态数）

[新增...]

新增加一笔文字标签。

[设置...]

修改文字标签的内容。

[删除]
删除指定的文件标签。

[删除全部]
删除现存所有标签

[复制]
复制文字标签的内容

[贴上]
贴上复制的文字标签内容

[储存文字库档案]
储存所有文字标签为lbl格式文件

[汇入文字库档案]
将现存文字标签lbl文件汇入文字标签库

[汇出为Excel档案]
使用CSV或xls格式将此文字标签库的所有内容输出到特定位置。此功能不支持Unicode。

[汇入Excel档案]
将已存在的且为CSV或xls格式的文字标签库加到目前的画面工程（MTP）。此功能不支持Unicode。

15.1 文字标签库字体设定

在“文字标签库管理对话窗”中可以看出目前已经存在的标签，标签所包含的8种语言分列如下：

不同语言可选择不同的字型，参考下图。

[字体]

[字体]设定页用来设定在使用多国文字时, 各种语言所使用的字型。

[备注]

每种字型的注释。

15.2 文字标签的建立

下面说明如何建立这些项目。

首先, 打开“文字标签库管理对话窗”后触控[新增…], 即可出现下图的对话窗, 正确设定后触控确认键。

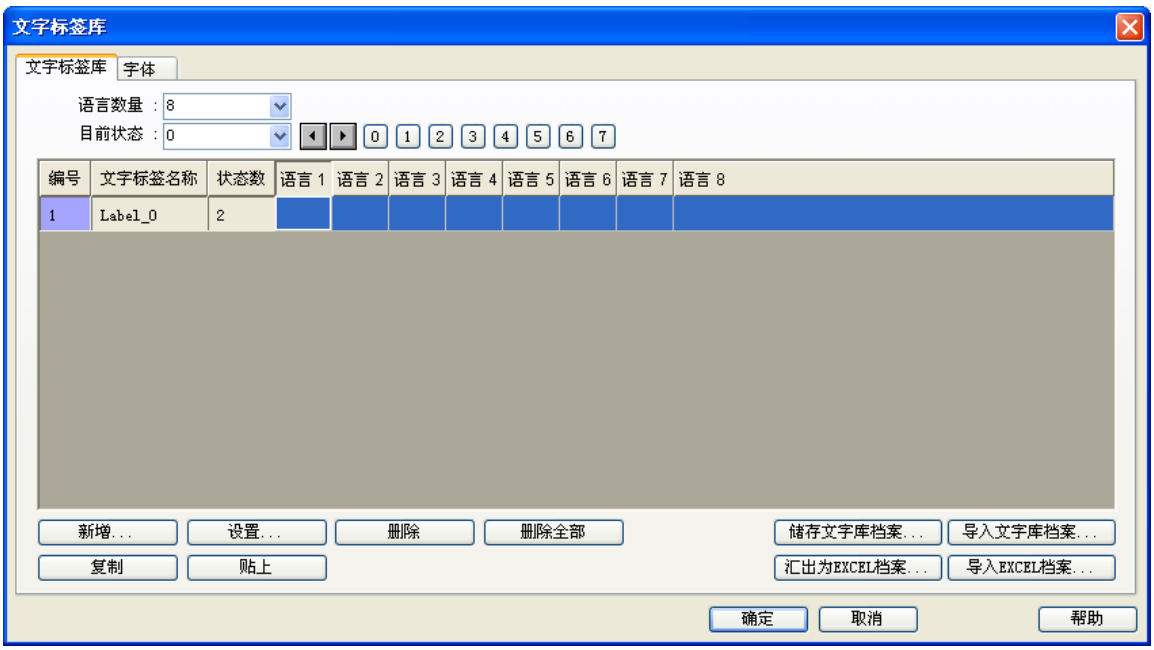
[标签名]

文字标签的名称, 此范例将标签命名为“Label_0”。

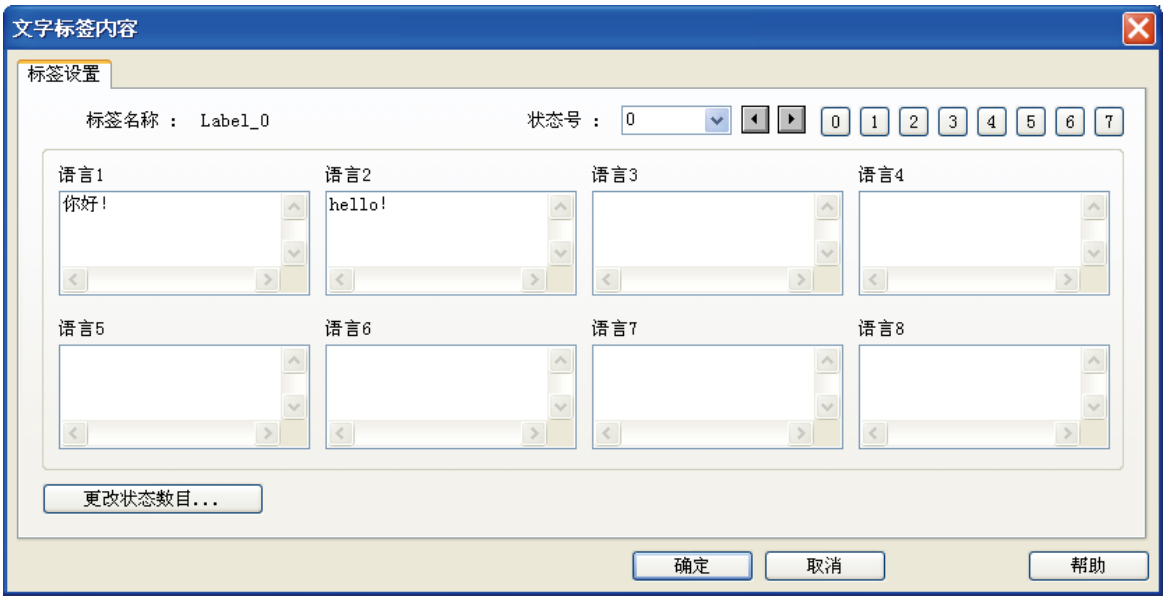
[状态数]

文字标签所拥有的状态。

完成此步骤后可以发现已存在一个新的标签“Label_0”, 并拥有两个状态, 参考下图。



最后, 选择“Label_0”并触控[设置…], 即可使用下图的对话窗设定相关语言的内容。



15.3 文字标签库的使用

当文字标签库存在已定义完成的标签,在元件的[标签]设定页中可以勾选[使用文字标签库],并且在[标签]的项目中可以发现目前这些标签。

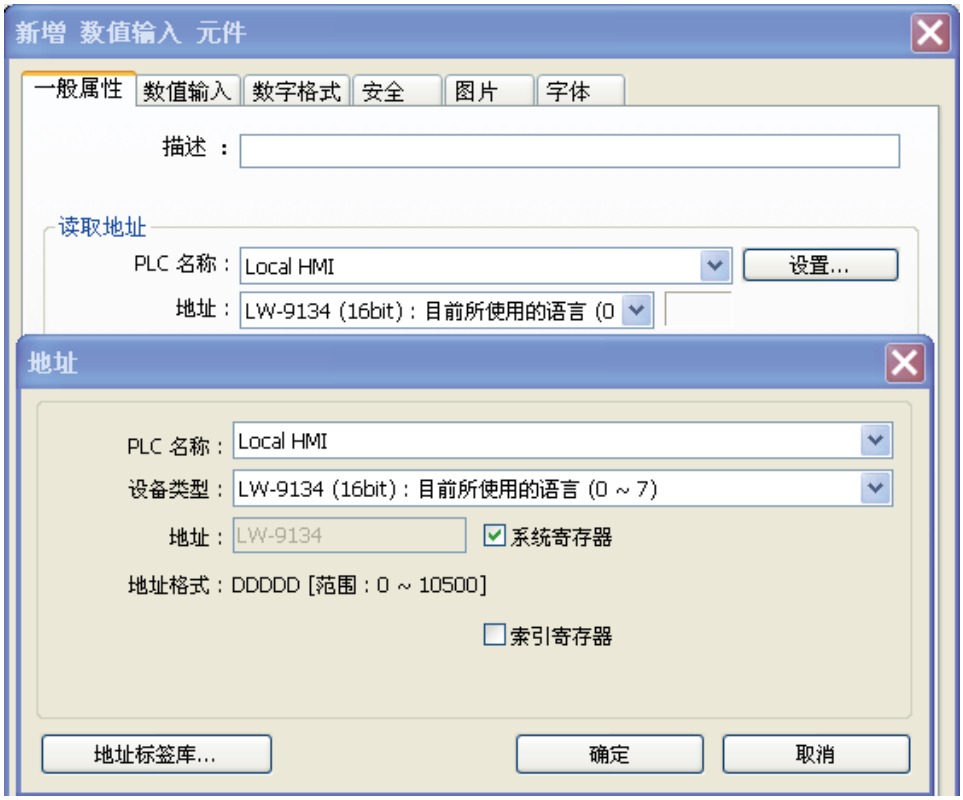
选择这些标签后可以发现[内容]中所显示为文字标签的内容,并且所使用的字型也为文字标签库的内容。

15.4 多国语言的使用

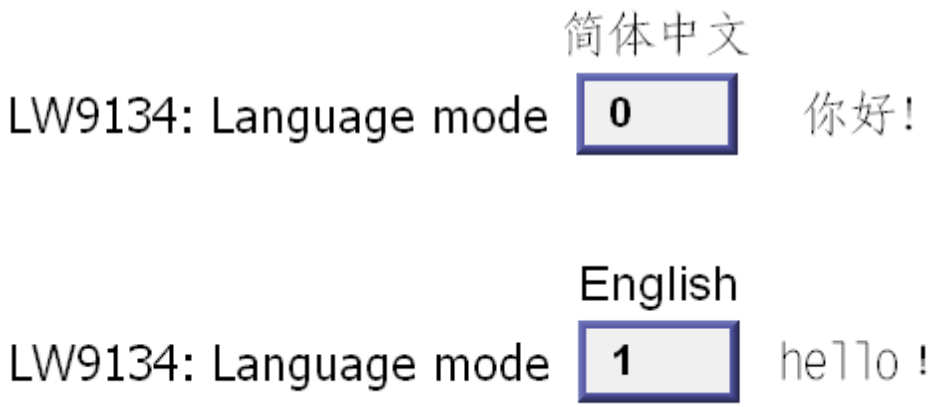
当元件的文字内容要求表现出多国语言的效果时，除了需使用文字标签外，也需搭配系统保留寄存器[LW9134]的使用。[LW9134]的有效可设定值范围为0~7，不同的数据对应到需显示的语言种类，下文一个简单的范例，说明如何使用多国语言。

首先建立一个文字元件，它的文字设定内容如下，可清楚看出此时使用文字标签。

下一步再建立一个数值输入元件，地址设定内容如下。可清楚看出此时所使用的地址为系统保留寄存器[LW9134]。



此时画面的模拟效果如下图，当我们更改[LW9134]的内容时，即可更换文字元件所显示的内容。



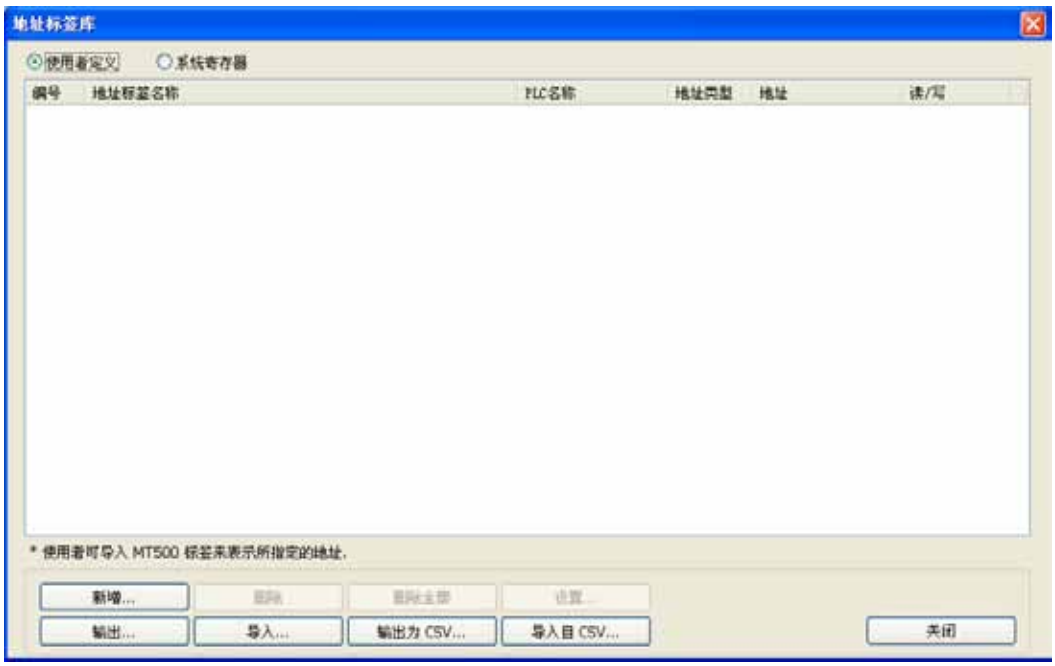
第十六章 地址标签库的建立与使用

一般会在画面设计开始时将常用的地址定义在地址标签库中，以文字变量的方式表示。

使用地址标签库，可以省去复杂的地址输入，也增加元件信息的可读性。在元件地址较多的情况下，修改一个地址标签的实际地址，所有在工程中使用过此标签的地址将全部更新，避免重复劳动；

16.1 地址标签库的建立

触控工具条上的按钮将出现“地址标签库管理对话窗”，参考下图。



[使用者定义]

显示用户自定义的地址标签。

[系统寄存器]

显示系统保留地址的地址标签。

[新增...]

增加一个新的地址标签。

[删除]

删除指定的地址标签。

[删除全部]

删除所有现存的地址标签。

[设置...]

更改指定的地址标签。

[输出...]

以.tgl格式储存地址标签文件。

[导入...]

将现存的.tgl文件载入地址标签库

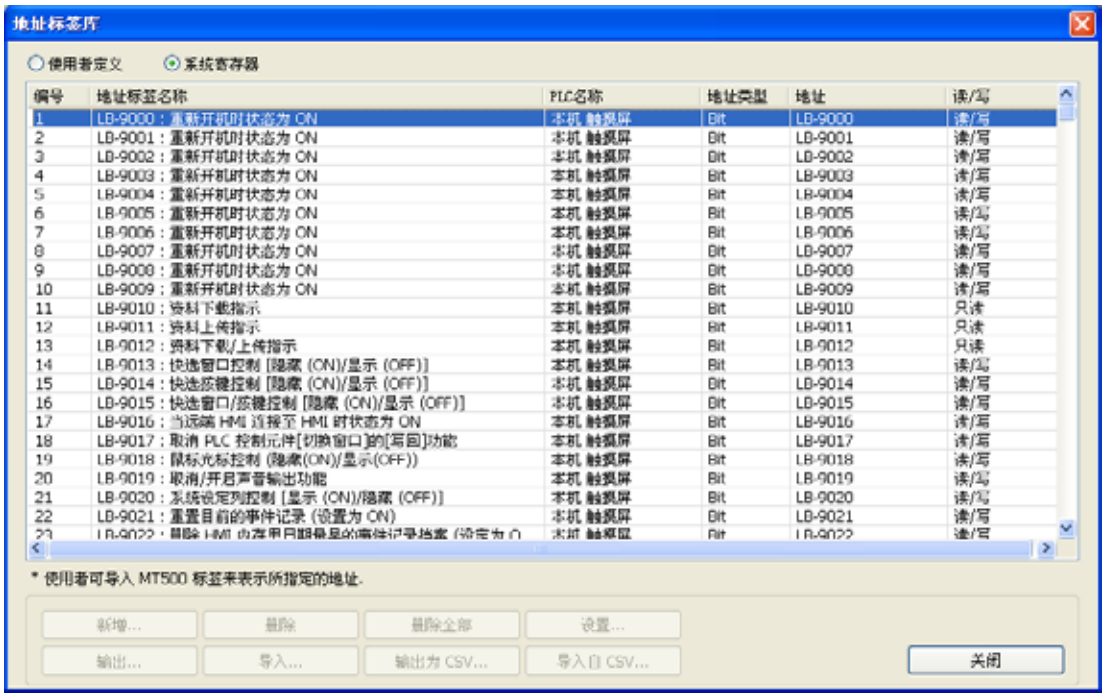
[输出为CSV]

将目前的地址标签库使用CSV文件格式输出到指定位置

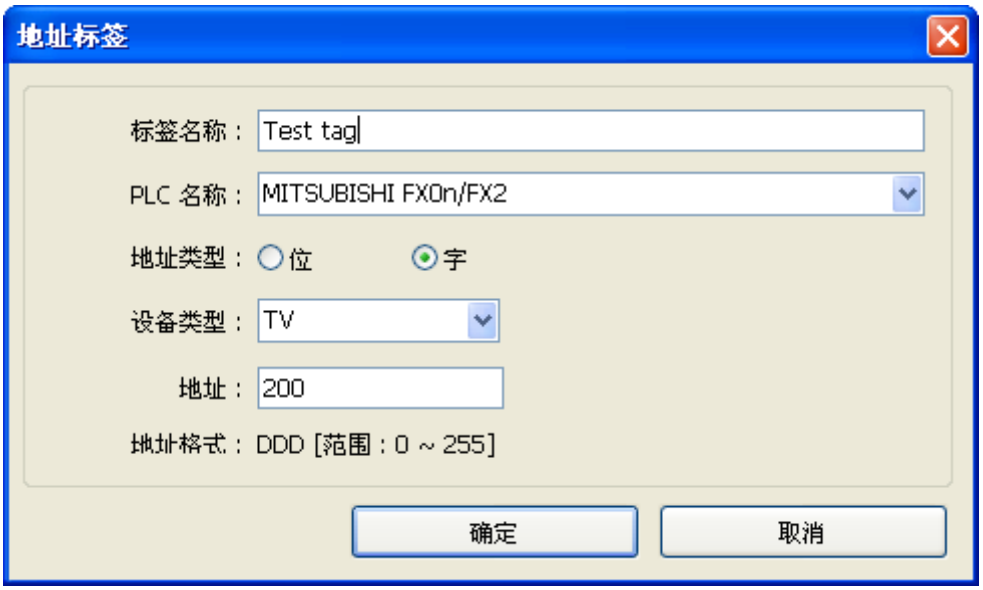
[导入自CSV]

将使用CSV文件储存的地址标签文件导入到目前的工程文件中

另一种地址标签为HMI系统的保留寄存器，可参考下图。



要使用地址标签首先需增加地址标签库的内容，在触控[新增 …]后可以得到下图显示的窗口。



[标签名称]

地址标签名称。

[PLC名称]

PLC 名称，可选择的名称来自设备清单的内容。

[地址类型]

地址类型，可选择Bit或Word类型。

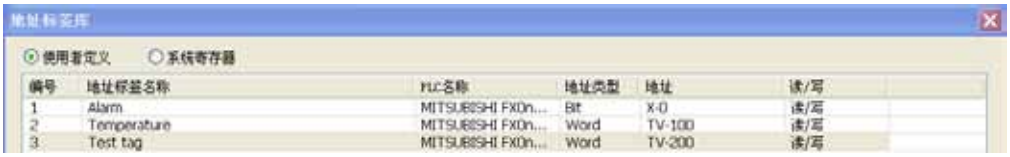
[设备类型]

设备类型，可选择类型与[PLC名称]、[地址类型]有关。

[地址]

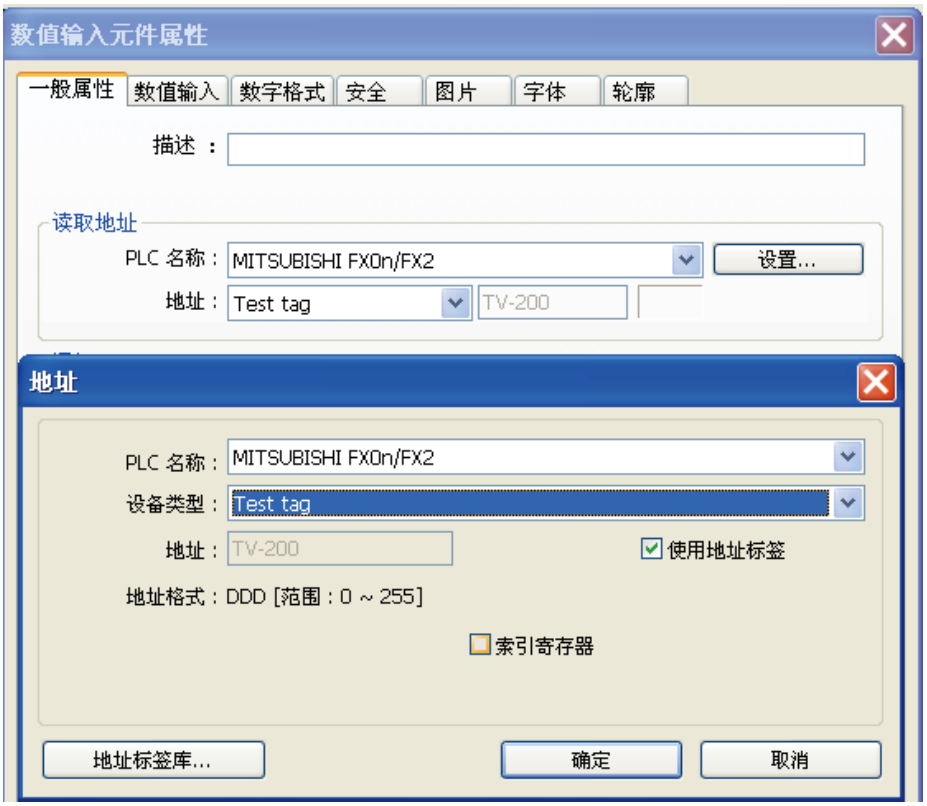
地址内容。

完成各项设定后触控确认键，即可在使用者自定义地址标签中发现一笔新的标签，参考下图。

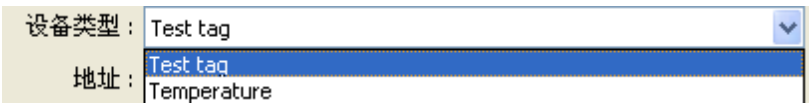


16.2 地址标签库的使用

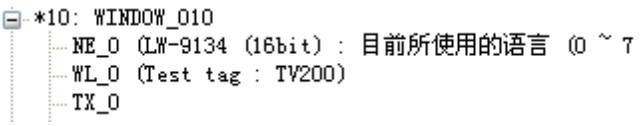
完成地址标签库的设计，并在元件的[一般属性]页中选择与这些标签有关的PLC后触控[设置]，即可发现存在[使用地址标签]的选项，勾选此选项后即可使用这些地址标签，参考下图。



此时[设备类型]具有下图所显示的选项：



在设定完成后，在目录树的元件列表中即可发现元件所使用的地址标签的名称，参考下图。



第十七章 配方资料传送

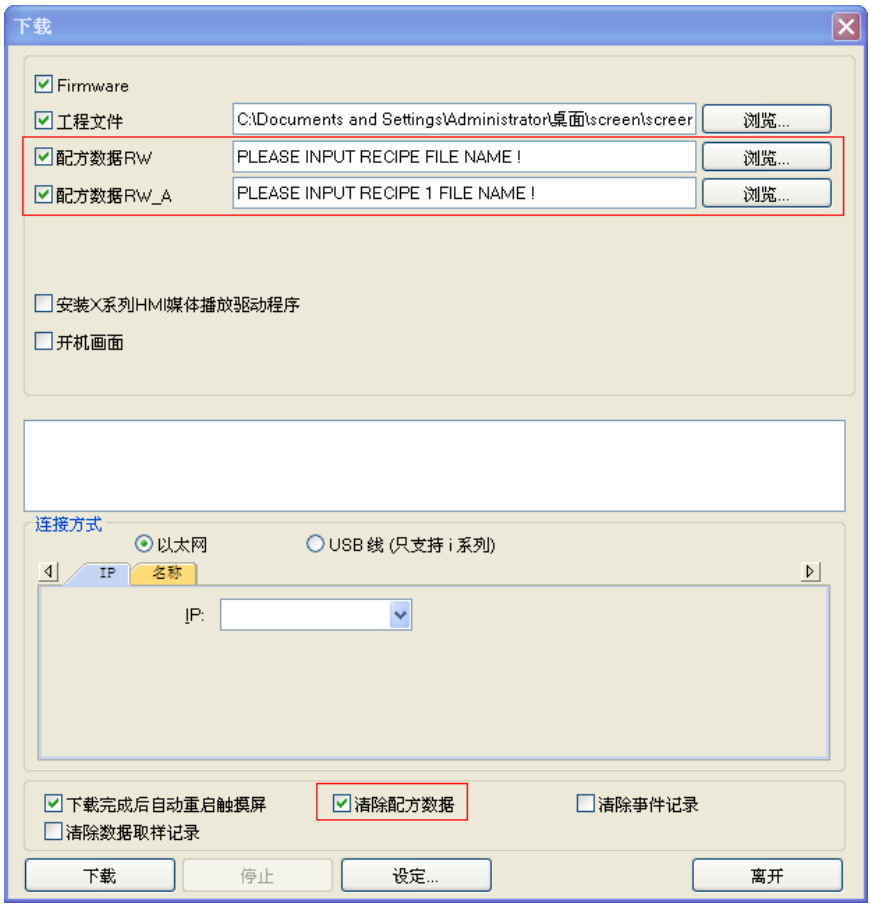
所谓配方资料一般是指存储在HMI内部RW与RW_A寄存器地址上的数据，读写这些地址的方式与读写一般word地址的方式并无不同，配方资料的特殊在于断电保持，关机后这些数据将保存在HMI上，重新开机后RW与RW_A地址上的数据将维持前一次记录的内容。

RW与RW_A地址上的配方资料大小分别为512K words和64K words，用户可以利用SD卡、U盘、USB数据线或以太网线更新配方资料，并利用这些资料更新PLC上的数据。也可以利用USB 数据线或以太网在线传配方数据到指定位置；此外，用户也可以将PLC上的数据保存在配方资料中。下文将说明针对配方资料的各种操作。

1. 使用以太网线或USB数据线更新配方资料

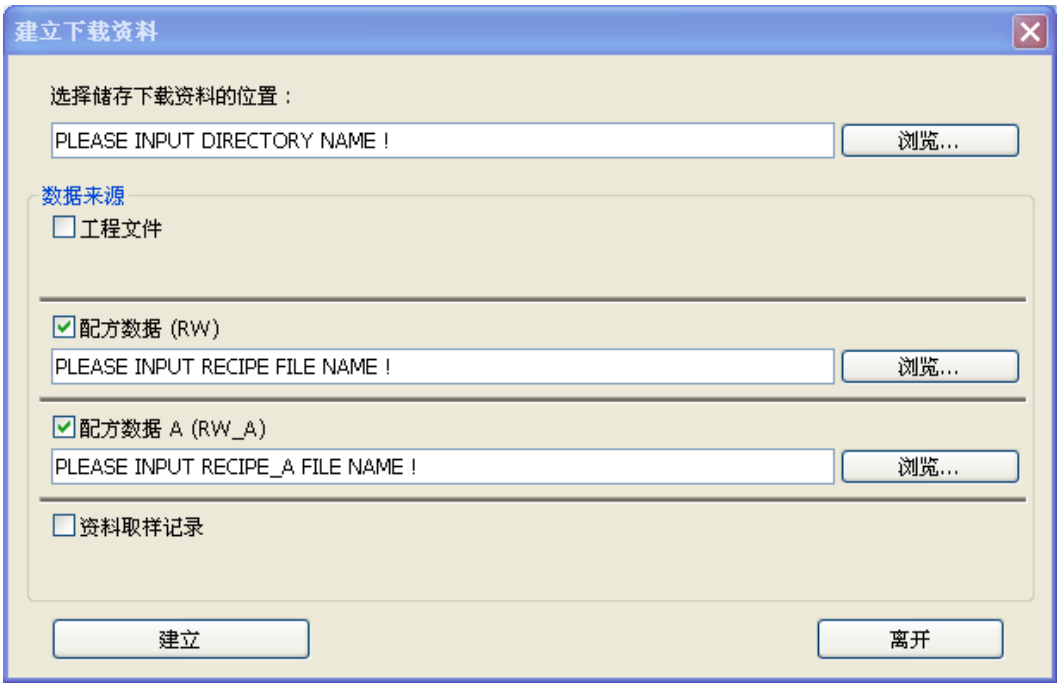
通过Project Manager的[下载…]功能，在勾选[RW]与[RW_A]后选择要下载的文件来源。下载成功后重新启动触摸屏，即可更新RW与RW_A的内容。

如果[清除配方数据]选项被勾选，在进行任何下载动作前，EB8000会先将[RW]与[RW_A]上的数据内容全部设定为0。



2. 使用SD卡或U盘更新配方数据

在Project Manager点击[建立储存在CF卡/SD卡与U盘中的下载资料]



将SD卡或U盘插入电脑后，点击“浏览”指定文件资料所要存放的路径名称，之后点击“建立”，EB8000会自动把所有要建立的来源资料文件写入到SD卡或U盘里。

注意：储存资料文件的路径有限制，请勿建立于电脑根目录，例如“c:\” 。“f:\\”亦是无效路径，应写成：“f:\”。

3. 传送配方资料

可以使用“触发式资料传输”元件，将配方资料传送到指定地址；也可以将指定地址的数据保存在[RW]与[RW_A]中。此部分请参考有关“触发式资料传输”元件的说明。

4. 配方资料强迫储存

为了增加HMI上flash的使用寿命，EB8000以每隔1分钟的时间间隔将配方资料保存在HMI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。EB8000提供[LB9029：强迫储存配方资料到触摸屏]，让用户可以自行进行配方资料的储存动作，只需对[LB9029]送出ON的信号，EB8000即会执行一次配方资料储存动作。另外如对[LB9028：重置配方资料]送出ON的信号，EB8000会将所有的配方数据复归为0。

第十八章 宏指令(macro)使用说明

宏指令提供了应用程式之外所需的附加功能。在MT8000触摸屏运行时，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、字符串处理和用户与工程之间的交流等功能。本章主要介绍宏指令的语法、如何使用和编辑方法等。希望通过本章的说明，能够使各位快速的掌握EB8000软件提供的强大的宏指令功能。

18.1 宏指令的结构

宏指令是由各种语句组成的。这些语句包含常量、变量和各种运算符号。这些语句放置在特定的顺序位置以便执行后达到一个希望的执行结果。

宏指令的结构一般为以下格式：

```
总体变量声明                          ----------可选

Sub Function Block Declarations(子函数声明) ----------可选
    局部变量声明
End Sub(结束子函数)

macro_command main()    [主函数]          ----------必须
    局部变量声明
    [各式语句]
end macro_command       [结束主函数]      ----------必须
```

一个宏指令必须有一个且只有一个主函数，用来开始宏指令的执行。格式为：

macro_command 函数名称()

end macro_command

变量声明必须放在宏指令语句的前面，否则如果语句放置在变量声明的前面，将会造成宏指令无法编译通过。

局部变量一般用在宏指令主函数或者自定义的子函数中。他的合法性只在指定的函数中有效。

总体变量一般是定义在所有宏指令函数的前面，且它在整个宏指令中均具有效性。在同一个函数中，当局部变量和总体变量被定义为相同的名称时，只有局部变量有效。

下面就是一个简单的宏指令，其中就包含了变量声明和函数调用。

macro_command main()

```
short pressure = 10          // 局部变量声明
SetData(pressure, "Allen-Bradley DF1", N7, 0, 1)     // 函数调用
end macro_command
```

此章节将说明宏指令的语法与编写的方式，包含以下几个部分：

18.2 宏指令的语法

1.常量与变量

a. 常量

常量是一个可以被各式语句直接使用的固定资料。有如下格式：

常量类型	使用说明	举例
十进制常量		345, -234, 0, 23456
十六进常量	必须以 0x 开头	0x3b, 0xffff, 0x237
字符型	字符必须使用单引号	'a', 'data', '设备名称'
布尔型		true, false

下面即为一个简单的常量使用的范例：

```
macro_command main()
   short A, B              // 声明A和B为短整型变量
   A = 1234
   B = 0x12                // 1234 和 0x12 即为常量
end macro_command
```

b. 变量

变量是程序执行时保存数据的量，在宏指令中，这些数据可以随着宏指令语句执行的结果而改变。

(1) 变量的命名规则

- 必须以英文字母为开头
- 变量名称长度不得超过32个字符
- 系统保留字名称不能作为变量名称

下面为8种不同的变量类型，前5种为有符号数据，后3种为无符号数据。

变量类型	描述	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 (两个字)	-2147483648~2147483647
Float 浮点型	32 bits (两个字)	
Unsigned Char 字符型	8 bits (一个字节)	0~255
Unsigned Short 短整型	16 bits (一个字)	0~65535
Unsigned Int 双整型	32 bits (两个)	0~4,294,967,295

(2) 变量声明

变量必须在使用前声明, 所以, 在宏指令中, 所有的变量都必须在语句使用前进行声明。声明变量时, 先定义变量的类型, 后面再跟上变量名称。

如下范例:

```
int     a
short b, switch
float  pressure
unsigned short  c
```

(3) 数组声明

宏指令支持一维数组(下标从0开始)。声明数组变量时 , 先定义数组变量的类型, 变量名称, 接着就是该数组变量元素的个数, 元素个数必须放置在 "[]" 符号中, 数组变量的长度为1~4096., 一个宏指令中只支持4096个变量。

如下范例:

```
int      a[10]
short   b[20], switch[30]
float   pressure[15]
```

数组的下标最小为0, 最大下标为(数组的长度–1)

如下范例:

```
char data [100]                  // 数组变量的长度为100
```

所以: 起始的数组元素为 "data[0]" , 最大下标的数组元素为 "data[99]"

(4) 变量和数组的初始化

有两种方法可以让变量初始化

1、使用语句中的赋值语句(=)

如下范例:

int a

float b[3]

a = 10

b[0] = 1

2、声明变量时直接赋值

char a = '5' , b = 9

数组变量的声明是一个特殊的情况, 一个完整的数组被初始化时, 可以在数组变量声明时将资料放置在 "{}" 中, 各资料使用逗号分开。

如下所示:

float data[4] = {11, 22, 33, 44} //这样data[0] = 11, data[1] = 22….

c. 运算符号

运算符通常被用来指定资料是如何被操作的, 在任何一个语句中, 运算符左边的变量结果均依据运算符右边的条件而获得。

运算符号	描述	举例
=	赋值运算符号	pressure = 10

数学运算符号	描述	举例舉例
+	加	A = B + C
-	减	A = B – C
*	乘	A = B * C
/	除	A = B / C
%	求余（返回剩余数）	A = B % 5

比较运算符号	描述	举例
<	小于	if A < 10 then B = 5
<=	小于或者等于	if A <= 10 then B = 5
>	大于	if A > 10 then B = 5
>=	大于或者等于	if A >= 10 then B = 5
==	等于	if A == 10 then B = 5
<>	不等于	if A <> 10 then B = 5

逻辑运算符号	描述	举例
And	与	if A < 10 and B > 5 then C = 10
Or	或	if A >= 10 or B > 5 then C = 10
Xor	异或	if A xor 256 then B = 5
Not	非	if not A then B = 5

移位运算符和位运算符通常被用来操作字符型变量、短整型变量和双整型变量的位。在一个语句中, 这些运算符号的优先权是在从该语句的左边到右边依次执行的, 即在语句中左边位置的优先执行, 依次从左到右执行;

移位运算符号	描述	举例
<<	往左移动指定位数	A = B << 8
>>	往右移动指定位数	A = B >> 8

位运算符号	描述	举例
&	位与运算	A = B & 0xf
|	位或运算	A = B | C
^	位异或运算	A = B ^ C
~	位取反运算	A = ~B

1) 所有运算符号的优先权:

上述所有运算符号的优先权从高到低详细如下所述:

位于圆括号里的运算符号最优先

数学运算符号

移位和位运算符号

比较运算符号

逻辑运算符号

赋值运算符号

2) 关键字

下面的关键字为宏指令保留使用。这些均不能用来作为变量名称、数组名或者函数名称等:

+, –, *, /, %, >=, >, <=, <, <>, ==, and, or, xor, not, <<, >>,=, &, |, ^, ~

exit, macro_command, for, to, down, step, next, return, bool, short, int, char, float, void, if, then, else, break, continue, set, sub, end, while, wend, true, false

ACOS、ADDSUM、ASCII2DEC、ASCII2FLOAT、ASCII2HEX、ASIN、ASYNC_TRIG_MACRO、ATAN、BCD2BIN、Beep、BIN2BCD、COS、COT、CRC、CSC、CUBERT、DEC2ASCII、DELAY、FILL、FindDataSamplingDate、FindDataSamplingIndex、FindEventLogDate、FindEventLogIndex、FLOAT2ASCII、GETBIT、GetCTS、GetData、GetDataEx、GetError、HEX2ASCII、HIBYTE、HIWORD、

INPORT、INVBIT、LOBYTE、LOG、LOG10、LOWORD、OUTPORT、POW、PURGE、RAND、SEC、SETBITOFF、SETBITON、SetData、SetDataEx、SetRTS、SIN、SQRT、StringBin2DecAsc、StringBin2HexAsc、StringCat、StringCompare、StringCompareNoCase、StringCopy、StringDecAsc2Bin、StringDecAsc2Float、StringExcluding、StringFind、StringFindOneOf、StringFloat2DecAsc、StringGet、StringGetEx、StringHexAsc2Bin、StringIncluding、StringInsert、StringLength、StringMid、StringReverseFind、StringSet、StringSetEx、StringToLower、StringToReverse、StringToUpper、StringTrimLeft、StringTrimRight、SWAPB、SWAPW、SYNC_TRIG_MACRO、TAN、TRACE、XORSUM。

18.3 语句

1. 定义语句

这个定义语句包含了变量和数组的声明，正式的格式如下：

类型　名称　当定义一个名称的类型时

举例：

int A　　//定义了变量A为双整型数据类型

类型　数组名称[数组长度]　当定义数组名称的类型

举例：

int B[10]　　//定义了一维数组变量B的长度为10，类型为双整型

2. 赋值语句

赋值语句使用赋值运算符号将赋值运算符号右边表达式运算的结果放置到运算符左边的变量中，一个表达式是由变量、常量和各种运算符号组成，执行后产生一个新的结果：

变量 = 表达式

举例：

A = 2　　//这样变量A就被赋值为2

3. 逻辑运算

逻辑运算语句是根据逻辑（布尔）表达式的结果来执行相应的动作，它的语句如下所示：

(1) 单行

```
if < Condition > then
        [Statements]
    else
        [Statements]
end if
```

举例：

```
if a = = 2 then
  b = 1
    else
      b = 2
end if
```

(2) 区块形式

```
If < Condition > then
        [Statements]
else if < Condition- n> then
        [Statements]
else
        [Statements]
end if
```

```
if a = = 2 then
  b = 1
    else if  a = = 3   then
      b = 2
        else
          b = 3
end if
```

语法说明

if	必须用在该语句的开始部分
Condition	必要，为一条件表达式，条件表达式值为零，则为假(false)，条件表达式值为非零，则为真(true)。
Then	必须放置在需要执行的语句之前，当 Condition 为（true）时执行。
Statements	在区块形式中是可选择的参数，在单行形式中，且没有 else 子句时，为必要参数，该语句在 Condition 为真时执行。
Else If	可选，一条或多条语句，在相对应的 Condition – n 为 true 时执行。
Condition-n	可选，解释同 condition
Else	可选，在上述 Condition 和 Condition – n 都不为 true 时执行。
End if	必要。在一个 if-then 语句中使用这个来结束 if-then 语句

(3) 循环控制

循环可控制依据循环条件来反复执行一个任务，循环控制有两种表达方式。

1) for–next语句

For–next语句通常用于循环次数已确定的情况，一个变量用作任务执行次数的计数器和结束循环任务执行的条件，这个变量为固定执行的次数。语法结构如下：

```
for [Counter] = <StartValue> to <EndValue> [step <StepValue>]
        [Statements]
    next [Counter]
```

或者

```
    for [Counter] = <StartValue> down <EndValue> [step <StepValue>]
        [Statements]
    next [Counter]
```

举例：

```
for   a = 0   to 10  step 2
      b = a
next a
```

语法说明

For	必须用在该语句的开始部分
Counter	必要，循环计数器的数值变量，该变量的结果用来计数循环的次数
StartValue	必要，Counter 的初值。
to/down	必要，用来决定是递增还是递减 “to”　以<StepValue>为步长递增<Counter> “down”以<StepValue>为步长递减<Counter>
EndVlaue	必要，Counter 的终值，测试点，当<Connter>大于该值时，宏指令将结束这个循环任务。
Step	可选，指定<Step Value>的步长，指定为 1 以外的数值。
StepValue	可选，Counter 的步长，只能是数值，如果没有指定，则预设为1。
Statements	可选，for 和 next 之间的语句区块，该语句区块将执行所指定的次数。
Next	必要
Counter	可选

2) while–wend语句

while–wend语句是用来执行不确定次数的循环任务，设置一个变量用来判断结束循环的条件，当条件为TRUE时，该语句将一直循环执行直到条件变为FALSE，语法结构如下：

```
while <Condition>
    [Statements]
wend
```

举例：

```
while a < 10
        a = a + 10
wend
```

语法说明

While	必须用在该语句的开始部分
Condition	必要，这是一个控制语句，当为TRUE时，开始执行循环命令，当为false时，结束执行循环命令
Return[value]	判断为 true 时，继续执行循环命令
Wend	While-wend 语句的结束标志

3) select–case语句

select–case可用来处理多重判断的语句，其功能类似if–else语句，根据所指定变量的值，分别对应到符合该值的case，并执行case下面的语句，直到遇到break语句时，才跳到结束符号end select处，语法结构如下：

没有预设case的形式：

```
Select Case [variable]
    Case [value]
    [Statements]
    break
end Select
```

举例：

```
        Select Case A
            Case 1
```

```
    b=1
  break
end Select
```

有预设case的形式

```
Select Case [variable]
    Case [value]
    [Statements]
    break
    Case else
    [Statements]
    break

end Select
```

举例：

```
        Select Case A
          Case 1
                b=1
          break
          Case else
                b=0
           break
        end Select
```

多个不同case对应到相同区域：

```
Select Case [variable]
    Case [value1]
    [Statements]
    Case [value2]
    [Statements]
    break

end Select
```

举例：

```
Select Case A
  Case 1
  Case 2
    b=2
   break
  Case 3
    b=3
 break
end Select
```

语法描述：

Select Case	必须用在该语句的开始部分
Variable	必要，此变量将会与每一个case做比较
Case else	可选，代表预设 case，当 variable 的值不符合任何一个 case 时，将会执行此叙述下面的区域，在没有预设 case 的情况，当 variable 的值不符合任何一个 case 时，将不会做任何动作而直接跳出 select 控制结构
break	可选，跳到某一个 case 下面执行时，将一句一句执行 case 语句下面的叙述直到遇到 break 命令才结束，并跳到 end select 叙述，当 case 叙述下面没有任何 break 命令时，流程将不断往下执行，直到遇到 end select 叙述，才结束并跳出 select 控制结构
end Select	select-case 语句的结束标志

4) 其他控制命令

break	使用在 for-next 和 while-wend 语句中，执行到此语句时，退出循环或条件语句。
continue	使用在 for-next 和 while-wend 语句中，执行到 continue 时，立即中断这一次循环的执行，直接执行下一次循环。
return	可用在自定 function 的回传值叙述，当在主函数里面时，用来强制跳出主函数。

18.4 子函数

使用子函数可以有效的减少循环命令的代码，子函数必须在使用前被定义，且可以使用任何变量和语句类型。在主函数中，将子函数的参数放置在子函数名称后面的圆括号中，即可调用子函数，子函数被执行后，将执行后的结果返回到主函数需要的赋值语句或条件中。定义子函数时，不一定要有返回值，且参数部分可以为空。在主函数中调用子函数时，调用方式符合其定义。语法结构如下：

有返回值的子函数语法：

```
sub type <函数名称> [(parameters)]
    Local variable declarations
    [Statements]
    [return [value]]
end sub
```

举例：

```
sub int Add(int x, int y)
    int result
    result = x +y
    return result
end sub

macro_command main()
    int  a = 10, b = 20, sum
    sum = Add(a, b)
end macro_command
```

或：

```
sub int Add()
    int result, x=10, y=20
    result = x +y
    return result
end  sub

macro_command main()
    int  sum
```

```
    sum = Add()
end macro_command
```

没有返回值的子函数语法：

```
sub <函数名称> [(parameters)]
    Local variable declarations
    [Statements]
end sub
```

举例：

```
sub Add(int x, int y)
    int result
    result = x +y
end sub

macro_command main()
    int  a = 10, b = 20
    Add(a, b)
end macro_command
```

或：

```
sub Add()
    int result, x=10, y=20
     result = x +y
end sub

macro_command main()
     Add()
end macro_command
```

语法描述：

sub	必须用在该子函数的开始部分
type	可选，用来定义子函数执行后返回的资料类型，子函数也可以不回传任何值
parameters	可选，这些参数保留了从主函数传入的数值，这些被传入的参数必须使用与在参数变量声明的类型一致： 例如：sub int MyFunction(int x, int y). x 和 y 必须为从主函数中传过来的双整型资料格式的资料，调用此函数的语句格式大致为这样：ret = MyFunction(456, pressure)，其中 pressure 需为双整型资料格式方符合子函数参数变量的声明。 当执行这个子函数后，一个双整型资料将会返回给变量 ret
Local variable declaration	除了被传递的参数之外，子函数中使用的变量必须事先声明，在上面的“举例”中，x 和 y 就是子函数可以使用的变量，总体变量也可以用在子函数中
Statements	需要执行的语句
return [value]	可选，用来将执行的结果返回给调用语句，这个结果可以是一个常量或是变量，返回值同时也结束了子函数的执行，子函数也可以不回传任何值，但是当 type 部分有定义时，则必须加上次 return 叙述
end sub	必要，用来结束子函数

18.5 内置函数功能

Easybuilder8000软件宏指令中本身提供了一些内建的函数用来从PLC获取资料和传输资料到PLC，资料处理和数学运算等。

(1)数值运算

函数名称	SQRT
语法	SQRT(source, result)
描述	SQRT 将 source 的开根号结果储存在 result 中，source 可以是常数或变量，result 必须为变量。
举例	macro_command main() float source, result SQRT(15, result) source = 9.0 SQRT(source, result)//　执行后result = 3.0 end macro_command

函数名称	CUBERT
语法	CUBERT (source, result)
描述	开三次方根，资料来源“source”可以是常量或变量，但是存放结果的“result”必须为变量，资料来源必须为一个正数

举例	macro_command main()
	float source, result
	
	CUBERT (27, result) //　执行后result = 3.0
	
	source = 27.0
	CUBERT (source, result)//　执行后result = 3.0
	
	end macro_command

函数名称	POW
语法	POW (source1, source2, result)
描述	计算 source1 的某次方（source2），资料来源“source1”和“source2”可以是常量或是变量，但是存放结果的“result”必须为变量，资料来源必须为一个正数
举例	macro_command main() float y, result y = 0.5 POW (25, y, result) //　执行后result = 5 end macro_command

函数名称	SIN
语法	SIN(source, result)
描述	三角函数的正弦计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	SIN(90, result)//　result is 1
	
	source = 30　　//单位为度
	SIN(source, result)//　result is 0.5
	
	end macro_command

函数名称	COS
语法	COS(source, result)
描述	三角函数的余弦计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	COS(90, result)// result is 0
	
	source = 60 //单位为度
	
	COS(source, result)// result is 0.5
	
	end macro_command

函数名称	TAN
语法	TAN(source, result)
描述	三角函数的正切计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	TAN(45, result)// result is 1
	
	source = 60 //单位为度
	TAN(source, result)// result is 1.732
	
	end macro_command

函数名称	COT
语法	COT(source, result)
描述	三角函数的余切计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	COT(45, result)// result is 1
	
	source = 60 //单位为度
	COT(source, result)// result is 0.5774
	
	end macro_command

函数名称	SEC
语法	SEC(source, result)
描述	反三角函数中反正割计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	SEC(45, result)// result is 1.414
	
	source = 60 //单位为度
	SEC(source, result)// if source is 60, result is 2
	
	end macro_command

函数名称	CSC
语法	CSC(source, result)
描述	反三角函数中反余割计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	CSC(45, result)// result is 1.414
	
	source = 30 //单位为度
	CSC(source, result)// result is 2
	
	end macro_command

函数名称	ASIN
语法	ASIN(source, result)
描述	反三角函数中反正弦计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	ASIN(0.8660, result)// result is 60
	
	source = 0.5
	ASIN(source, result)// result is 30
	
	end macro_command

函数名称	ACOS
语法	ACOS(source, result)
描述	反三角函数中反余弦计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	ACOS(0.8660, result)//　result is 30
	
	source = 0.5
	ACOS(source, result)//　result is 60
	
	end macro_command

函数名称	ATAN
语法	ATAN(source, result)
描述	反三角函数中反正切计算，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	float source, result
	
	ATAN(1, result)//　result is 45
	
	source = 1.732
	ATAN(source, result)//　result is 60
	
	end macro_command

函数名称	LOG
语法	LOG (source, result)
描述	从取得 source 的自然对数，存入 result 变量 Source 可为变量或常量 存放结果的“result”必须为变量
举例	macro_command main() float source=100, result LOG (source, result)//　result约等于4.6052 end macro_command

函数名称	LOG10
语法	LOG10 (source, result)
描述	从取得 source 的以 10 为基数的对数，存入 result 变量 Source 可以是变量或常量 存放结果的“result”必须为变量
举例	macro_command main() float source=100, result LOG10 (source, result)//　result等于2 end macro_command

函数名称	RAND
语法	RAND(result)
描述	产生一个随机数 存放结果的“result”必须为变量
举例	macro_command main() short result RAND (result)//　result is not a fixed value when executes macro every time end macro_command

(2) 数值转换

函数名称	BIN2BCD
语法	BIN2BCD(source, result)
描述	将 BIN 格式的资料（source）转换为 BCD 格式的资料(result)，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	
	unsigned　short source, result
	
	BIN2BCD(1234, result)//　result is 0x1234
	
	source = 5678
	BIN2BCD(source, result)//　result is 0x5678
	
	end macro_command

函数名称	BCD2BIN
语法	BCD2BIN(source, result)
描述	将 BCD 格式的资料（source）转换为 BIN 格式的资料(result)，资料来源“source”可以是常量或者变量，但是存放结果的“result”必须为变量

举例	macro_command main()
	
	unsigned short source, result
	
	BCD2BIN(0x1234, result)// result is 1234
	
	source = 0x5678
	BCD2BIN(source, result)// result is 5678
	
	end macro_command

函数名称	DEC2ASCII
语法	DEC2ASCII(source, result[start], len)
描述	将十进制的资料（source）转换为 ASCII 格式的资料，并存放在一个一维数组（result）中，len 表示这个转换后的字串的长度，同时这个长度也取决于存放结果的一维数组的资料格式。例如：如果 result 一维数组的格式为“char”,（字符型，长度为一个字节），则长度为“字节长度”；如果 result 一维数组的格式为“short”（短整型资料，2 个字节），则长度为“字长”；依次类推。转换后的第一个字放置 result[start]中，第二个放在 result[start+1]中，最后一个放置 result[start+（len-1）]中。 Source 和 len 可以是常量或者是变量，但是 result 必须为变量，start 必须为常量

举例	macro _ commandmain( )
	unsigned short source
	unsigned char result1[4]
	unsigned short result2[4]
	
	source = 5678
	DEC2ASCII(source, result1[0], 4)
	// result1[0] is '5', result1[1] is '6', result1[2] is '7', result1[3] is '8'
	// the length of the string (result1) is 4 bytes(= 1 * 4)
	
	DEC2ASCII(source, result2[0], 4)
	// result2[0] is '5', result2[1] is '6', result2[2] is '7', result2[3] is '8'
	// the length of the string (result2) is 8 bytes(= 2 * 4)
	
	end macro_command

函数名称	HEX2ASCII
语法	HEX2ASCII(source, result[start], len)
描述	将十六进制的资料（source）转换为 ASCII 格式的资料，并将结果存放在一个一维数组（result）中，len 表示这个转换后的字串的长度，同时这个长度也取决于存放结果的一维数组的资料格式。例如：如果 result 一维数组的格式为“char”，（字符型，长度为一个字节），则长度为“字节长度”；如果 result 一维数组的格式为“short”（短整型资料，2 个字节），则长度为“字长”；依次类推。 Source 和 len 可以是常量或者是变量，但是 result 必须为变量，start 必须为常量
举例	macro_command main() unsigned short source unsigned char result[4] source = 0x5678 HEX2ASCII(source, result[0], 4) // result[0] is '5', result[1] is '6', result[2] is '7', result[3] is '7' end macro_command

函数名称	FLOAT2ASCII
语法	FLOAT2ASCII (source, result[start], len)
描述	将浮点格式的资料（source）转换为 ASCII 格式的资料，并将结果存放在一个一维数组（result）中，len 表示这个转换后的字串的长度，同时这个长度也取决于存放结果的一维数组的资料格式。例如：如果 result 一维数组的格式为“char”，（字符型，长度为一个字节），则长度为“字节长度”；如果 result 一维数组的格式为“short”（短整型资料，2 个字节），则长度为“字长”；依次类推。 Source 和 len 可以是常量或者是变量，但是 result 必须为变量，start 必须为常量
举例	macro_command main() float source unsigned char result[4] source = 56.8 FLOAT2ASCII (source, result[0], 4) // result[0] is '5', result[1] is '6', result[2] is '.', result[3] is '8' end macro_command

函数名称	ASCII2DEC
语法	ASCII2DEC(source[start], result, len)
描述	将字符型 ASCII 资料（source）转换为十进制格式的资料，并存放在 result 变量中，ASCII 的长度即为 len，第一个字符的位置即为 source[start]的资料。 Source 和 len 可以是常量或者是变量，但是 result 必须为变量，start 必须为常量
举例	macro_command main() unsigned char source[4] unsigned short result source[0] = '5' source[1] = '6' source[2] = '7' source[3] = '8' ASCII2DEC(source[0], result, 4) // result is 5678 end macro_command

函数名称	ASCII2HEX
语法	ASCII2HEX (source[start], result, len)
描述	将字符型 ASCII 资料（source）转换为十六进制格式的资料，并存放在 result 变量中，第一个字符的位置即存放在 source[start]中。 Source 和 len 可以是常量或者是变量，但是 result 必须为变量，start 必须为常量
举例	macro_command main() unsigned char source[4] unsigned short result source[0] = '5' source[1] = '6' source[2] = '7' source[3] = '8' ASCII2HEX(source[0], result, 4) // result is 0x5678 end macro_command

函数名称	ASCII2FLOAT
语法	ASCII2FLOAT (source[start], result, len)
描述	将字符型 ASCII 资料（source）转换为浮点数格式的资料，并存放在 result 变量中，ASCII 的长度即为 len，第一个字符的位置即为 source[start]的资料。 Source 和 len 可以是常量或者是变量，但是 result 必须为变量，start 必须为常量
举例	macro_command main() unsigned char source[4] float result source[0] = '5' source[1] = '6' source[2] = '.' source[3] = '8' ASCII2FLOAT(source[0], result, 4) // result is 56.8 end macro_command

(3) 数值操作

函数名称	FILL
语法	FILL(source[start], preset, count)
描述	依序将预设值（preset）放置到一维数组 source[start]开始的数组中，放置的资料个数由 count 决定 Source 和 start 必须为变量，preset 可以为一个常量或者变量

举例	macro_command main()
	char result[4]
	char preset
	
	FILL(result[0], 0x30, 4)
	// result[0] is 0x30, result[1] is 0x30, , result[2] is 0x30, , result[3] is 0x30
	
	preset = 0x31
	FILL(result[0], preset, 2) // result[0] is 0x31, result[1] is 0x31
	
	end macro_command

函数名称	SWAPB
语法	SWAPB(source, result)
描述	将一个 16 位字(source)的高低字节互换，并将结果存放在 result 中，source 可以是常量或者变量，但是 result 必须为变量。

举例	macro_command main()
	unsigned short source, result
	
	SWAPB(0x5678, result)// result is 0x7856
	
	source = 0x123
	SWAPB(source, result)// result is 0x2301
	
	end macro_command

函数名称	SWAPW
语法	SWAPW(source, result)
描述	将一个 32 位双整型资料的高位字和低位字颠倒，并将结果存放在 result 变量中 source 可以为一个常量或者变量，但是 result 必须为变量

举例	macro_command main()
	unsigned int source, result
	
	SWAPW(0x12345678, result)// result is 0x56781234
	
	source = 0x12345
	SWAPW(source, result)// result is 0x23450001
	
	end macro_command

函数名称	LOBYTE
语法	LOBYTE(source, result)
描述	获取一个 16 位字的低字节，并且放置在 result 变量中 source 可以为一个常量或者变量，但是 result 必须为变量

举例	macro_command main()
	unsigned short source, result
	
	LOBYTE(0x1234, result)// result is 0x34
	
	source = 0x123
	LOBYTE(source, result)// result is 0x23
	
	end macro_command

函数名称	HIBYTE
语法	HIBYTE(source, result)
描述	获取一个 16 位字的高字节，并且放置在 result 变量中 source 可以为一个常量或者变量，但是 result 必须为变量

举例	macro_command main()
	unsigned short source, result
	
	HIBYTE(0x1234, result)// result is 0x12
	
	source = 0x123
	HIBYTE(source, result)// result is 0x01
	
	end macro_command

函数名称	LOWORD
语法	LOWORD(source, result)
描述	获取一个 32 位字中的低字，并且放置在 result 变量中 source 可以为一个常量或者变量，但是 result 必须为变量

举例	macro_command main()
	unsigned int source, result
	
	LOWORD(0x12345678, result)// result is 0x5678
	
	source = 0x12345
	LOWORD(source, result)// result is 0x2345
	
	end macro_command

函数名称	HIWORD
语法	HIWORD(source, result)
描述	获取一个 32 位字中的高字，并且放置在 result 变量中 source 可以为一个常量或者变量，但是 result 必须为变量

举例	macro_command main()
	unsigned int source, result
	
	HIWORD(0x12345678, result)// result is 0x1234
	
	source = 0x12345
	HIWORD(source, result)// result is 0x0001
	
	end macro_command

(4) 位操作

函数名称	GETBIT
语法	GETBIT(source, result, bit_pos)
描述	获取资料或者变量（source）指定位的状态，并将结果放置在result变量中，result的资料将为1或者0 Source和bit-pos可以是常量或者变量，但是result必须为变量

举例	macro_command main()
	unsigned int source, result
	unsigned short bit_pos
	
	GETBIT(9, result, 3)// result is 1
	
	source = 4
	bit_pos = 2
	GETBIT(source, result, bit_pos)// result is 1
	
	end macro_command

函数名称	SETBITON
语法	SETBITON(source, result, bit_pos)
描述	将资料或者变量（source）的指定位设置为1，并将改变后的资料放置在result变量中 Source和bit-pos可以是常量或者变量，但是result必须为变量

举例	macro_command main()
	unsigned int source, result
	unsigned short bit_pos
	
	SETBITON(1, result, 3)// result is 9
	
	source = 0
	bit_pos = 2
	SETBITON (source, result, bit_pos)// result is 4
	
	end macro_command

函数名称	SETBITOFF
语法	SETBITOFF(source, result, bit_pos)
描述	将资料或者变量（source）的指定位设置为0，并将改变后的资料放置在result变量中 Source和bit-pos可以是常量或者变量，但是result必须为变量

举例	macro_command main()
	unsigned int source, result
	unsigned short bit_pos
	
	SETBITOFF(9, result, 3)// result is 1
	
	source = 4
	bit_pos = 2
	SETBITOFF(source, result, bit_pos)// result is 0
	
	end macro_command

函数名称	INVBIT
语法	INVBIT(source, result, bit_pos)
描述	将资料或者变量（source）的指定位的状态取反，并将改变后的资料放置在result变量中 Source和bit-pos可以是常量或者变量，但是result必须为变量

举例	macro_command main()
	unsigned int source, result
	unsigned short bit_pos
	
	INVBIT(4, result, 1)// result = 6
	
	source = 6
	bit_pos = 1
	INVBIT(source, result, bit_pos)// result = 4
	
	end macro_command

(5) 通讯

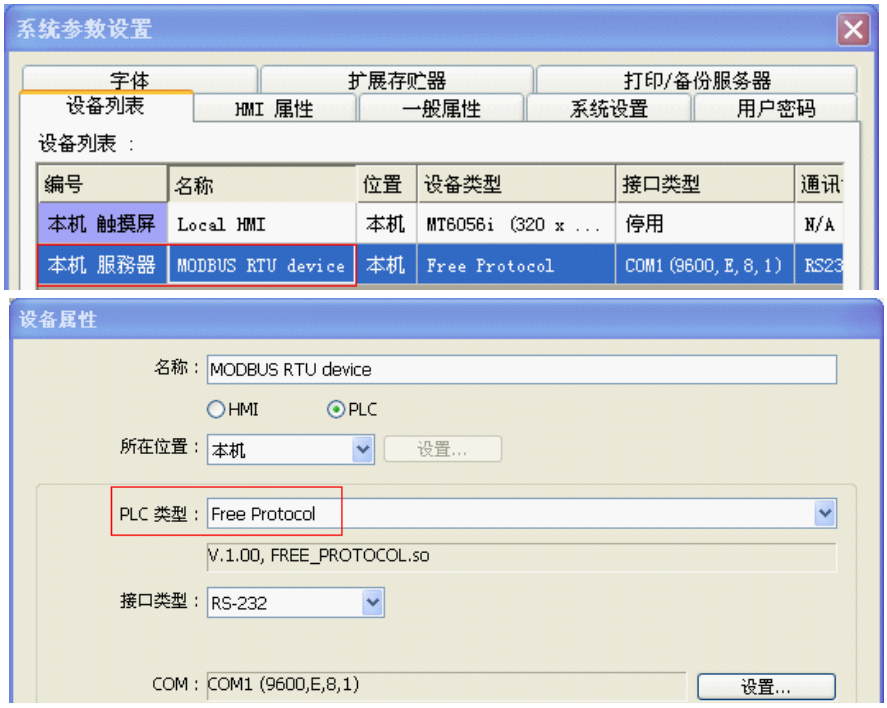
函数名称	DELAY
语法	DELAY(time)
描述	让宏指令暂停执行，持续的时间至少是指定的这个时间 time 可以是常量或者变量

举例	macro_command main()
	unsigned int time = 500
	
	DELAY(100)// delay 100 ms
	DELAY(time)// delay 500 ms
	
	end macro_command

函数名称	ADDSUM
语法	ADDSUM(source[start], result, data_count)
描述	将 source[start]到 source[start+data-count-1]的所有一维数组的资料累加起来，以获得 checksum（校验和），并将结果存放在 result 变量中 Result 必须为变量，data-count 是进行累加的资料的个数，可以是常量或者是变量
举例	macro_command main() unsigned char data[5] unsigned short checksum data[0] = 0x1 data[1] = 0x2 data[2] = 0x3 data[3] = 0x4 data[4] = 0x5 ADDSUM(data[0], checksum, 5)// checksum is 0xf end macro_command

函数名称	XORSUM
语法	XORSUM(source[start], result, data_count)
描述	将 source[start]到 source[start+data-count-1]的所有一维数组的资料进行异或运算，以获得 checksum（校验和），并将结果存放在 result 变量中 result 必须为变量，data-count 是进行异或计算的资料的个数，可以是常量或者是变量
举例	macro_command main() unsigned char data[5] = {0x1, 0x2, 0x3, 0x4, 0x5} unsigned short checksum XORSUM(data[0], checksum, 5)// checksum is 0x1 end macro_command

函数名称	CRC
语法	CRC(source[start], result, data_count)
描述	将 source[start]到 source[start+data-count-1]的所有一维数组的资料进行 16-bit CRC 计算，以获得 checksum（校验和），并将结果存放在 result 变量中 result 必须为变量，data-count 是进行计算的资料的个数，可以是常量或者是变量
举例	macro_command main() unsigned char data[5] = {0x1, 0x2, 0x3, 0x4, 0x5} unsigned short 16bit_CRC CRC(data[0], 16bit_CRC, 5)// 16bit_CRC is 0xbb2a end macro_command

函数名称	OUTPORT
语法	OUTPORT(source[start], device_设备名称, data_count)
描述	将放置在从 source[start]到 source[start+count-1]的所有资料通过串口或者以太网口传送给 PLC 或者控制器中 device_设备名称是在“设备列表”中定义的“PLC 名称”，而这个 device 必须选择为“Free Protocol”这个 PLC 类型 data-count 是发送资料的个数，可以是常量
举例	要使用 OUTPORT 函数，必须要在 PLC 类型中选择“Free Protocol”，如下图所示： 

举例	这里的 device-设备名称即为“MODBUS RTU Device”，串口属性也是依据在这个“系统参数”中的设定，例如，在此设定为（9600，E，8，1…） 下面时一个范例程式，使用自由协议，以 MODBUS RTU 的协议格式，将单个寄存器设定为 ON macro_command main() char command[32] short address, checksum FILL(command[0], 0, 32)//　初始化变量 command[0] = 0x01// 站号 command[1] = 0x05// 功能码：写单个位 address = 0 HIBYTE(address, command[2]) LOBYTE(address, command[3]) command[4] = 0xff//　使该bit设置为ON command[5] = 0 CRC(command[0], checksum, 6) LOBYTE(checksum, command[6]) HIBYTE(checksum, command[7]) //　将命令通过串口送出去 OUTPORT(command[0], **"MODBUS RTU Device"**, 8) end macro_command

函数名称	INPORT
语法	INPORT(read_data[start], device_设备名称, read_count, return_value)
描述	从串口或者以太网口读取数据到触摸屏上，这些资料保存在 read-data[start]~read-data[start+read-count-1]这个一维数组中，同样的 device_h 设备名称见上面的说明，在此不再详述 read-count 是设定的需要读取的命令的长度，它可以是一个常量，如果这个函数能够成功的从 PLC 或者控制器中读取到资料，则 return_value 的值为 1，否则就为 0
举例	下面就是一个使用 INPORT 函数读取一个 MODBUS 设备保持寄存器资料的范例： //　读取保持寄存器资料 macro_command main() char command[32], response[32] short address, checksum short read_no, return_value, read_data[2] FILL(command[0], 0, 32)//　初始化变量 FILL(response[0], 0, 32) command[0] = 0x 01// 站号 command[1] = 0x03// 功能码 ：03H address = 0 HIBYTE(address, command[2]) LOBYTE(address, command[3]) read_no = 2//　read 2 words (4x_1 and 4x_2) HIBYTE(read_no, command[4]) LOBYTE(read_no, command[5]) CRC(command[0], checksum, 6) LOBYTE(checksum, command[6]) HIBYTE(checksum, command[7]) //　使用OUTPORT函数将命令送出去 OUTPORT(command[0], "MODBUS RTU Device", 8) //　使用INPORT函数读取返回的命令 INPORT(response[0], "MODBUS RTU Device", 9, return_value)

	if return_value > 0 then
	read_data[0] = response[4] + (response[3] << 8)// data in 4x_1
	read_data[1] = response[6] + (response[5] << 8)// data in 4x_2
	
	SetData(read_data[0], "Local HMI", LW, 100, 2)
	end if
	
	end macro_command

函数名称	GetData
语法	GetData(read_data[start], device_设备名称, device_type, address_offset, data_count) or GetData(read_data, device_设备名称, device_type, address_offset, 1)
描述	获取 PLC 的资料，资料是存储在 read_data[start]~read_data [start+data_count-1]这个一维数组变量中 data_count 是设定的读取数据的个数，一般来说 read_data 是一个一维数组，但是如果 data_count 是 1，read_data 可以是一个一维数组，也可以是一个普通的变量，下面是两种从 PLC 中读取一个字的方法： macro_command main() short read_data_1[2],read_data_2 GetData(read_data_1[0],“FATEKFB Series”, RT,5, 1) GetData(read_data_2,“FATEKFBSeries”,RT,5,1) end macro_command 此处的 device-设备名称，即为在“系统参数”中建立 PLC 类型时，设定的“PLC 名称”，在此 PLC 名称被设定为“FATEK FB Series”，如下图所示：  device_type 是设备类型和 PLC 中资料的编码方式，例如：如果 device_type 是 LW_BIN，那么读取的设备类型为 LW，资料编码方式为 BIN，如果使用 BIN 编码方式，“_BIN”可以忽略

如果 device_type 是 LW_BCD，表示设备类型 LW，资料的编码方式为 BCD 格式
address_offset 是 PLC 中的地址偏移量
例如： GetData(read_data_1[0], “FATEK FB Series”, RT, 5, 1) 代表读取的设备地址偏移量为 5
如果 address_offset 使用格式为“N#AAAAA”，N 表示 PLC 的站号，AAAAA 表示地址偏移量，此情况一般使用在同一个串口上连接有多台 PLC 或者控制器的情况下。例如：GetData(read_data_1[0], “FATEK FB Series”, RT, 2#5, 1)表示读取站号为 2 的 PLC 的资料。如果 GetData（）使用“系统参数/设备列表”中设定的默认站号，在此可以不填这个站号。

从 PLC 中读取的资料个数，根据 read_data 变量的类型和 data_count 的值来决定，如下表所示：

read_data 的类型	data_count 的值	读取 16bit 数据的个数
char (8-bit)	1	1
char (8-bit)	2	1
bool (8-bit)	1	1
bool (8-bit)	2	1
short (16-bit)	1	1
short (16-bit)	2	2
int (32-b t)	1	2
int (32-bit)	2	4
float (32-bit)	1	2
float (32-bit)	2	4

当 Getdata()函数读取 32bit 的资料类型（int 或者 float 型）时，此函数会自动转换这个资料，例如：

```
macro_command main()
float f
GetData(f, "MODBUS", 6x, 2, 1)   // f 中将会是浮点型的数据
end macro_command
```

举例	macro_command main()
	bool a
	bool b[30]
	short c
	short d[50]
	int e
	int f[10]
	float g
	
	// 读取 LB2 的状态到变量 a 中
	GetData(a, “Local HMI”, LB, 2, 1)
	
	// 读取 LB0～LB29 共 30 个状态，到变量 b[0]～b[29]中
	GetData(b[0], “Local HMI”, LB, 0, 30)
	
	// 读取 LW2 的资料到变量 c 中
	GetData(c, “Local HMI”, LW, 2, 1)
	
	// 读取 LW0~LW49 共 50 个字到 d[0]到 d[49]中
	GetData(d[0], “Local HMI”, LW, 0, 50)
	
	// 读取两个字 LW6~LW7 到变量 e 中
	// 注意此时变量 e 的类型为 int
	GetData(e, “Local HMI”, LW, 6, 1)
	
	// 读取 LW0~LW19 共 20 个字到 f[0]~f[9]中，数组 f[10]的变量类型定义为 int。
	GetData(f[0], “Local HMI”, LW, 0, 10)
	
	// 读取 LW2~LW3 共 2 个字到变量 g 中
	GetData(g, “Local HMI”, LW, 2, 1)
	
	end macro_command

函数名称	GetDataEx
语法	GetDataEx (read_data[start], device_设备名称, device_type, address_offset, data_count) or GetDataEx (read_data, device_设备名称, device_type, address_offset, 1)
描述	获取 PLC 的资料，不等待 PLC 回应，径自往下执行 read_data、device_设备名称、device_type、address_offset 和 data_count 的说明和 GetData 相同

举例	macro_command main()
	bool a
	bool b[30]
	short c
	short d[50]
	int e
	int f[10]
	float g
	
	// 读取 LB2 的状态到变量 a 中
	GetDataEx (a, “Local HMI”, LB, 2, 1)
	
	// 读取 LB0~LB29 共 30 个状态，到变量 b[0]~b[29]中
	GetDataEx (b[0], “Local HMI”, LB, 0, 30)
	
	// 读取 LW2 的资料到变量 c 中
	GetDataEx (c, “Local HMI”, LW, 2, 1)
	
	// 读取 LW0~LW49 共 50 个字到 d[0]到 d[49]中
	GetDataEx (d[0], “Local HMI”, LW, 0, 50)
	
	// 读取两个字 LW6~LW7 到变量 e 中
	// 注意此时变量 e 的类型为 int
	GetDataEx (e, “Local HMI”, LW, 6, 1)
	
	// 读取 LW0~LW19 共 20 个字到变量 f[0]~f[9]中，数组 f[10] 的变量类型定义为 int
	GetDataEx (f[0], “Local HMI”, LW, 0, 10)
	
	// 读取 LW2~LW3 共 2 个字到变量 g 中
	GetDataEx (g, “Local HMI”, LW, 2, 1)
	
	end macro_command

函数名称	SetData
语法	SetData(send_data[start], device _ 设备名称, device _ type, address_offset, data_count) or SetData(send_data, device_设备名称, device_type, address_offset, 1)
描述	将数据写到 PLC 中，资料保存在 send_data[start]~send_data[start+data_count-1]中 data_count 时写入到 PLC 中资料的个数，一般来说 send_data 是一个数组，但是如果 data_count 是 1，send_data 可以是一个数组也可以是一个普通的变量，下面是写一个资料到 PLC 中的方法： macro_command main() short send_data_1[2] = { 5, 6}, send_data_2 = 5 SetData(send_data_1[0], “FATEK FB Series”, RT, 5, 1) SetData(send_data_2, “FATEK FB Series”, RT, 5, 1) end macro_command device_函数名称详见上面的说明，在此不再说明。 device_type 是设备类型和 PLC 中资料的编码方式，例如：如果 device_type 是 LW_BIN，那么写入的设备类型为 LW，资料编码方式为 BIN，如果使用 BIN 编码方式，“_BIN”可以忽略 如果 device_type 是 LW_BCD，表示设备类型 LW，资料的编码方式为 BCD 格式 address_offset 是 PLC 中的地址偏移量 例如： SetData(read_data_1[0], “FATEK FB Series”, RT, 5, 1) 代表写入的设备地址偏移量为 5 如果 address_offset 使用格式为“N#AAAAA”，N 表示 PLC 的站号，AAAAA 表示地址偏移量，此情况一般使用在同一个串口上连接有多台 PLC 或者控制器的情况下。例如：SetData(read_data_1[0], “FATEK FB Series”, RT, 2#5, 1)表示设定站号为 2 的 PLC 的资料。如果 SetData（）使用“系统参数/设备列表”中设定的默认站号，在此可以不填这个站号。 设定到 PLC 的资料个数，根据 send_data 变量的类型和 data_count 的值来决定，如下表所示：

send_data 的类型	data_count 的值	设定 16bit 数据的个数
char (8-bit)	1	1
char (8-bit)	2	1
bool (8-bit)	1	1
bool (8-bit)	2	1
short (16-bit)	1	1
short (16-bit)	2	2
int (32-bit)	1	2
int (32-bit)	2	4
float (32-bit)	1	2
float (32-bit)	2	4

当 Setdata()函数写入 32bit 的资料类型(int 或者 float 型)到 PLC 时，此时函数会自动转换这个资料，例如：

```
macro_command main()
float f = 2.6
SetData(f, "MODBUS", 6x, 2, 1)   //在此将会设定一个浮点数到 PLC 中
end macro_command
```

举例

```
macro_command main()
int i
bool a = true
bool b[30]
short c = false
short d[50]
int e = 5
int f[10]

for i = 0 to 29
  b[i] = true
next i

for i = 0 to 49
  d[i] = i * 2
next i

for i = 0 to 9
  f [i] = i * 3
next i
```

	// 将变量 a 的数值设定到 LB2 中
	SetData(a, “Local HMI”, LB, 2, 1)
	
	// 设定 LB0~LB29 共 30 个位的状态
	SetData(b[0], “Local HMI”, LB, 0, 30)
	
	// 将变量 c 的值设定到 LW2 中
	SetData(c, “Local HMI”, LW, 2, 1)
	
	// 设定 LW0~LW49 共 50 个数据
	SetData(d[0], “Local HMI”, LW, 0, 50)
	
	// 将变量 e 的值写入到 LW6 ~LW7 两个寄存器中，注意变量 e 的类型为 int
	SetData(e, “Local HMI”, LW, 6, 1)
	
	// 设定 LW0~LW19 共 20 个字的数据
	// 10 个双整型数据等于 20 个 16bit 整型数
	SetData(f[0], “Local HMI”, LW, 0, 10)
	
	end macro_command

函数名称	SetDataEx
语法	SetDataEx（send_data[start], device_设备名称, device_type, address_offset, data_count) or SetDataEx (send_data, device_设备名称, device_type, address_offset, 1)
描述	将数据写到 PLC 中，不等待 PLC 回应，径自往下执行。 send_data、device_函数名称、device_type、address_offset 和 data_count 的说明和 SetData 相同。
举例	macro_command main() int i bool a = true bool b[30] short c = false short d[50] int e = 5 int f[10]

```
for i = 0 to 29
  b[i] = true
next i

for i = 0 to 49
  d[i] = i * 2
next i

for i = 0 to 9
  f [i] = i * 3
next i

//    将变量 a 的数值设定到 LB2 中
SetDataEx(a, “Local HMI”, LB, 2, 1)


//   设定 LB0~LB29 共 30 个位的状态
SetDataEx (b[0], “Local HMI”, LB, 0, 30)

// 将变量 c 的值设定到 LW2 中
SetDataEx (c, “Local HMI”, LW, 2, 1)

//   设定 LW0~LW49 共 50 个数据
SetDataEx (d[0], “Local HMI”, LW, 0, 50)

// 将变量 e 的值写入到 LW6 ~LW7 两个寄存器中，注意变量 e 的类型为 int。
SetDataEx (e, “Local HMI”, LW, 6, 1)

//   设定 LW0~LW19 共 20 个字的数据
//   10 个双整型数据等于 20 个 16bit 整型数
SetDataEx (f[0], “Local HMI”, LW, 0, 10)

end macro_command
```

函数名称	GetError
语法	GetError(err)
描述	取得错误码

举例	macro_command main()
	unsigned short err
	char byData[10]
	
	GetDataEx(byData[0], “MODBUS RTU”, 4x, 1, 10)// 读取10 byte资料
	
	//当错误代码(err)为0，表示GetDataEx成功执行
	GetError(err)//读取错误代码，存入变量err
	
	end macro_command

函数名称	PURGE
语法	PURGE (com_port)
描述	com_port 表示串口号，支持 COM1~COM3，此参数可以为变量或常量 可利用这个指令来清空 COM port 的缓存区
举例	macro_command main() unsigned int com_port=3 PURGE (com_port) PURGE (1) end macro_command

函数名称	SetRTS
语法	SetRTS(com_port, source)
描述	拉高或拉低 RS-232 的 RTS 讯号 com_port 表示串口编号，支持 COM1，此参数可以为变量或常量，source 参数也可以为变量或常量 当传入的 source 参数大于 0 时，将拉高 RTS 电位，当 source 参数值等于 0 时，将拉低 RTS 电位

举例	macro_command main()
	char com_port=1
	char value=1
	
	SetRTS(com_port, value) // value>0，拉高COM1的RTS电位
	
	SetRTS(1, 0) //拉低COM1的RTS电位
	
	end macro_command

函数名称	GetCTS
语法	GetCTS(com_port, result)
描述	侦测 RS-232 的 CTS 讯号 com_port 表示串口编号，支持 COM1，此参数可以为变量或常量，result 参数必须为变量 此函数可侦测 CTS 电位值，当 CTS 在高电位时，result 将写入 1，否则写入 0

举例	macro_command main()
	char com_port=1
	char result
	
	GetCTS(com_port, result) // 侦测COM1的CTS电位值
	
	GetCTS (1, result) //侦测COM1的CTS电位值
	
	end macro_command

函数名称	Beep
语法	Beep ()
描述	发出系统警示音 此函数可以发出 800 赫兹，30 毫秒的系统警示音
举例	macro_command main() Beep() end macro_command

(6) 其他函数

函数名称	SYNC_TRIG_MACRO
语法	SYNC_TRIG_MACRO(macro_id)
描述	一个执行中的macro可以使用此函数利用同步的方式触发执行其他macro，使用macro_id指定被触发的macro 使用此函数的macro将暂停执行，直到被触发的macro执行完成才会继续执行后续的命令 macro_id 可以是常量或使用变量来表示

举例	macro_command main() char ON = 1, OFF = 0 SetData(ON, “Local HMI”, LB, 0, 1) SYNC_TRIG_MACRO(5)// call a macro (its ID is 5) SetData(OFF, “Local HMI”, LB, 0, 1) end macro_command

函数名称	ASYNC_TRIG_MACRO
语法	ASYNC_TRIG_MACRO(macro_id)
描述	一个执行中的macro可以使用此函数，利用非同步的方式触发执行其他macro，使用macro_id指定被触发的macro Macro在使用此函数后将立即执行后续的命令，不需等待被触发的macro执行完成 macro_id 可是时常量或使用变量来表示
举例	macro_command main() char ON = 1, OFF = 0 SetData(ON, “Local HMI”, LB, 0, 1) ASYNC_TRIG_MACRO(5)// call a macro (its ID is 5) SetData(OFF, “Local HMI”, LB, 0, 1) end macro_command

函数名称	TRACE
语法	TRACE(format, argument)
描述	一个执行中的macro可以使用此函数，检视变量内容的变化，并打印字符串，以协助除错，用户可以开启EasyDiagnoser观看此函数的输出结果 当TRACE函数抓取一个%开头的特殊字，将同时从*argument*抓取一个参数做格式化后输出 *Format* 代表打印格式，支持%开头的特殊字符，特殊字符格式如下，其中方括号内的栏尾可选，粗体字栏位为必须： **%**[flags] [width] [.precision] **type** 每个栏位的意义如下所示： *flags* (可选)： - +

	width (可选)： 十进制正整数，指定应预留的字长，不足部分以空格补齐 *precision* (可选)： 十进制正整数，指定精确度，以及输出字数 *type*： C 或 c ： 以字节方式输出 d ： 以 signed 十进制整数输出 i ： 以 signed 十进制整数输出 o ： 以 unsigned 八进制整数输出 u ： 以 unsigned 十进制整数输出 X 或 x ： 以 unsigned 十六进制整数输出 E 或 e ： 以科学表示法输出 f ： 以单精度浮点数输出 *format* 字串最长支持 256 个字节，多出的字节将被忽略 *argument* 部分可写可不写，单一个特殊字符应搭配一个变量

举例	macro_command main()
	char c1 = 'a'
	short s1 = 32767
	float f1 = 1.234567
	
	TRACE("The results are") // 输出： The results are
	TRACE("c1 = %c, s1 = %d, f1 = %f", c1, s1, f1)
	// 输出： c1 = a, s1 = 32767, f1 = 1.234567
	
	end macro_command

18.6 如何建立和执行宏指令

(1) 如何建立一个宏指令

按照以下步骤可以建立一个宏指令。

步骤1

单击EB8000软件工具栏的宏指令图标 ，打开宏指令管理框，如下图所示：

在宏指令管理框中，已经编译成功的宏指令会出现在“已编译成功”列表上，未完成编译的会出现在“未完成编译”列表中，下面时宏指令管理框中个按键的功能描述：

[新增]

新增一个宏指令，并打开新建宏指令的编辑器

[删除]

删除选择的宏指令

[编辑]

打开宏指令的编辑器，并开启选择的宏指令

[复制]

复制选择的宏指令

[粘贴]

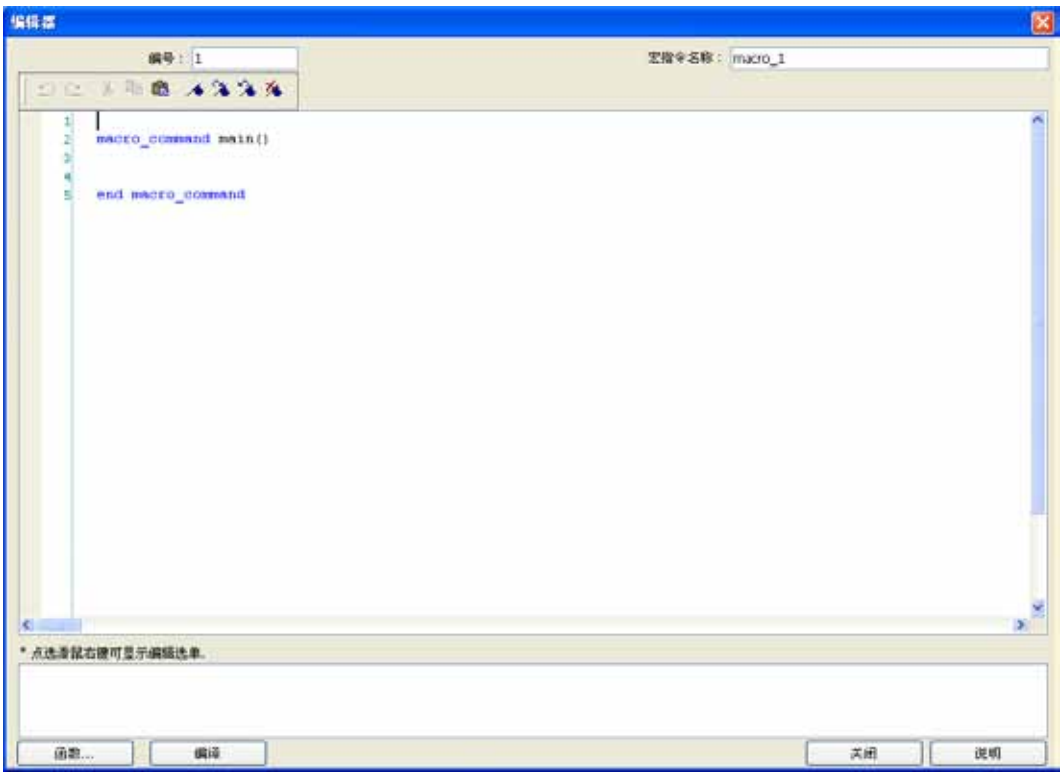
将刚刚选择需要复制的宏指令，粘贴到“已编译完成”列表区，并产生一个新的宏指令名称

[密码保护]

宏指令提供密码保护功能。

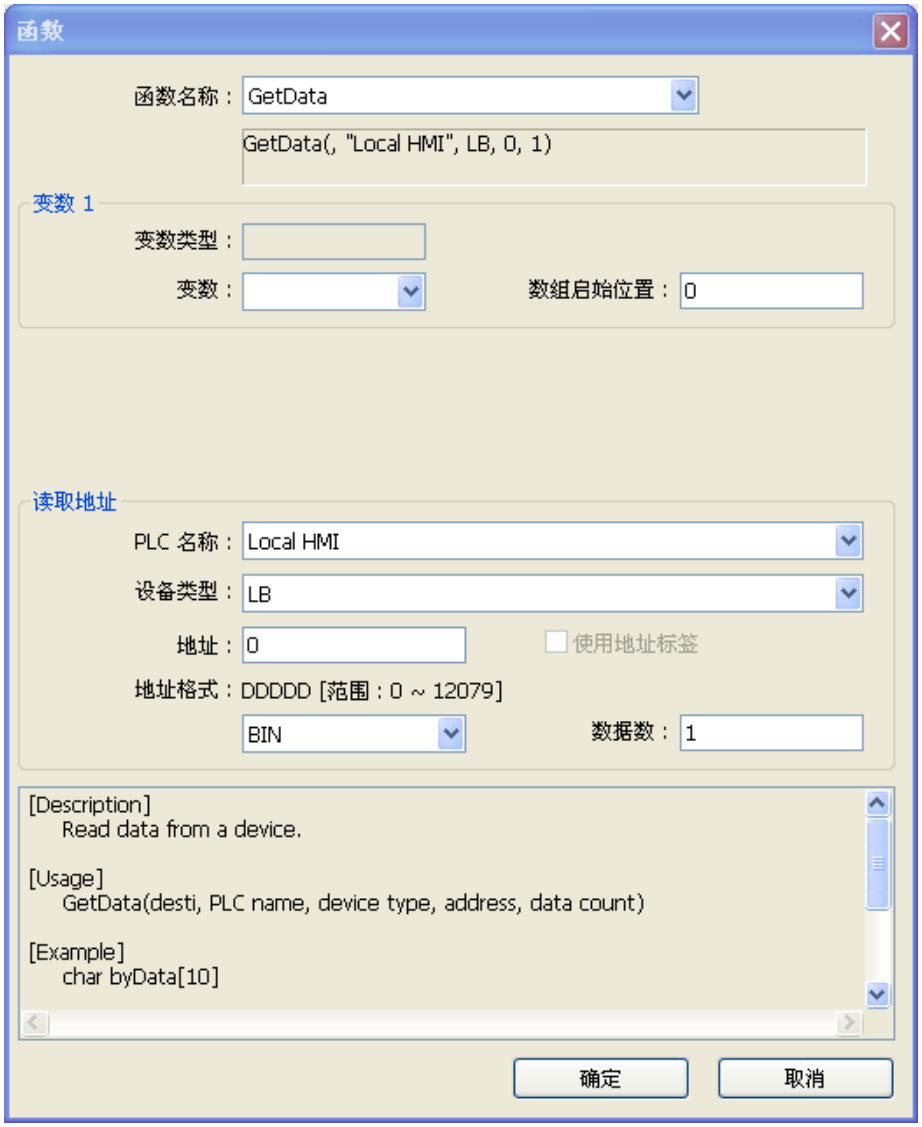
步骤2

触控“新增”按钮，打开一个新增的宏指令编辑器，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”这个栏目中也必须输入宏指令的名称，否则编译将无法通过。



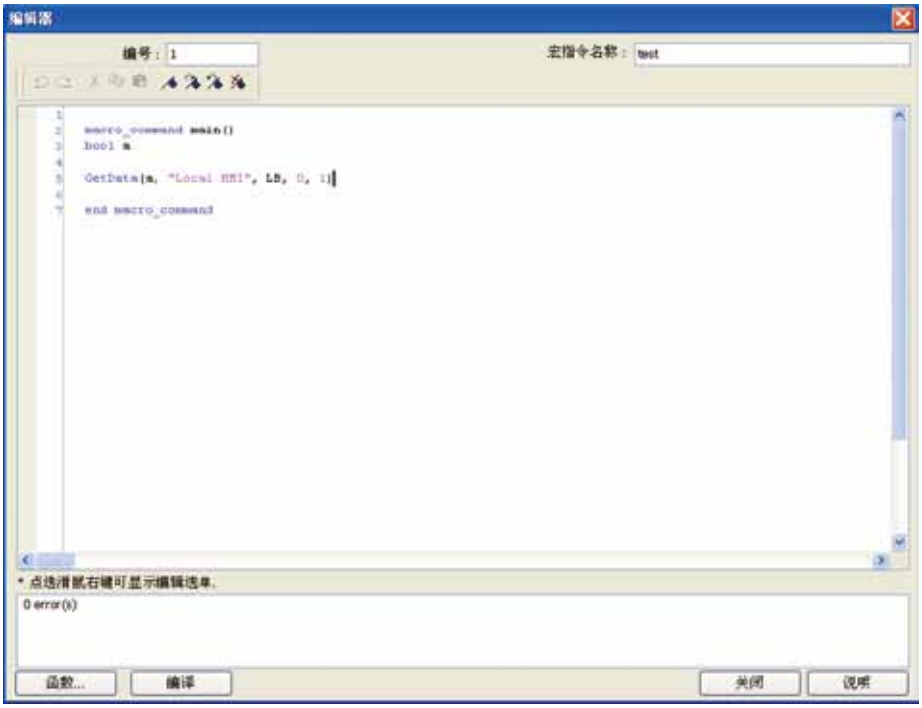
步骤3

设计属于您的宏指令程序，如果有必要的话，使用内建的函数，例如Setdata或者Getdata等函数。单击“函数…”按键打开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4

编辑完成一个新建的宏指令程序后，单击“编译”按钮对该宏指令进行编译工作。



如果没有错误，单击“关闭”按键，这样在“已编译成功”区会发现新增了一个“test”这个名称的宏指令。

(2)如何执行宏指令

执行宏指令有多种不同的方法,下面分别说明:

a. 使用 “PLC控制” 元件

1.打开PLC控制元件, 并设定属性为 “执行宏指令”

2.选择需要执行的宏指令名称, 设定一个位作为宏指令的触发地址并设置触发条件, 在条件满足时, 该宏指令将会被重复执行, 为了每次只让宏指令执行一次, 设计时需在宏指令最后将该触发位复位。

3.使用一个 “位状态设定” 元件或者 “位状态切换开关” 元件作为这个位的控制开关

b. 使用 “位状态设定” 元件或者 “位状态切换开关” 元件

1.在 “位状态设定” 元件或者 “位状态切换开关” 元件的一般属性页中, 勾选 “使用宏指令”

2.选择执行宏指令的编号, 当这个元件被执行时, 选择的宏指令就会被执行一次

c. 使用功能键

1.在功能键的一般属性页对话框中, 勾选 “触发宏指令”

2.选择需要执行的宏指令的编号, 每按一下这个功能键时, 选择的宏指令就会被执行一次

18.7 使用宏指令时的注意事项

1. 存储局部变量的空间为4KB, 所以各种不同变量类型的最大数组大小如下:

   char a[4096]

   bool b[4096]

   short c[2048]

   int d[1024]

   float e[1024]

2. 一个EasyBuilder8000工程中最多包含256条宏指令

3. 宏指令有可能造成HMI当机, 可能的原因为:

   – 宏指令执行了一个闭环命令

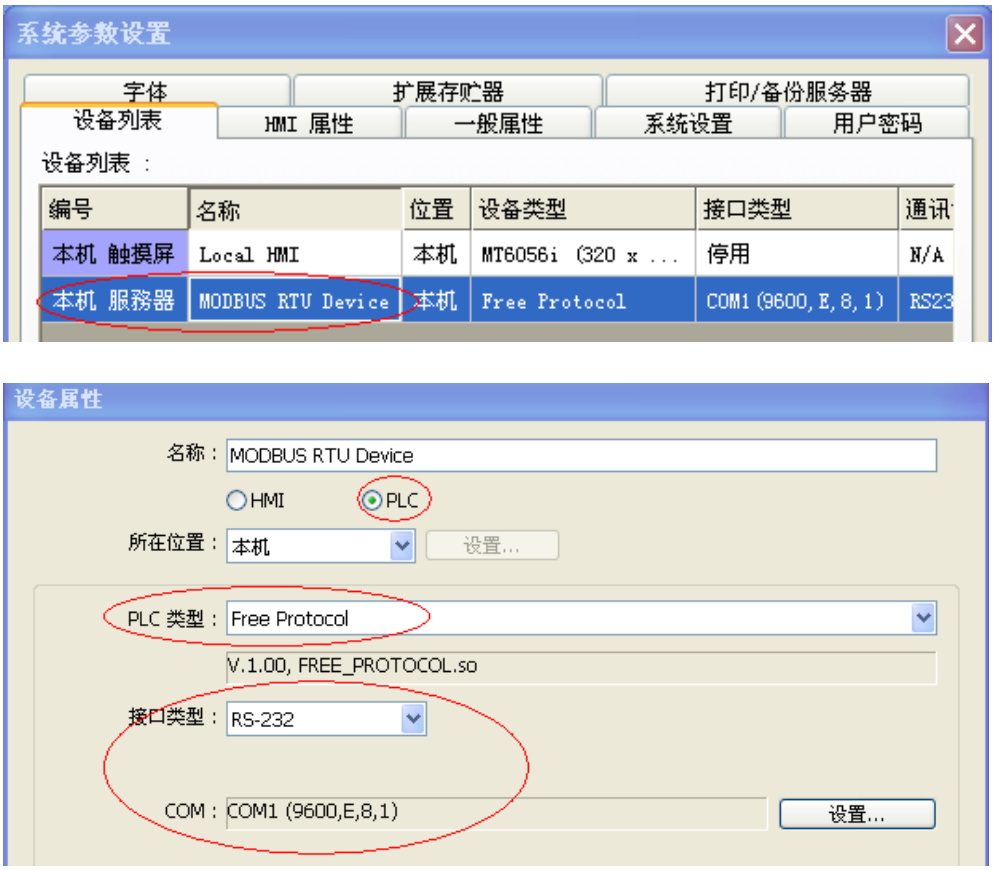
   – 数组的大小超过了宏指令的变量容量

4. 使用过多的宏指令, 可能会造成与PLC的通讯速度慢

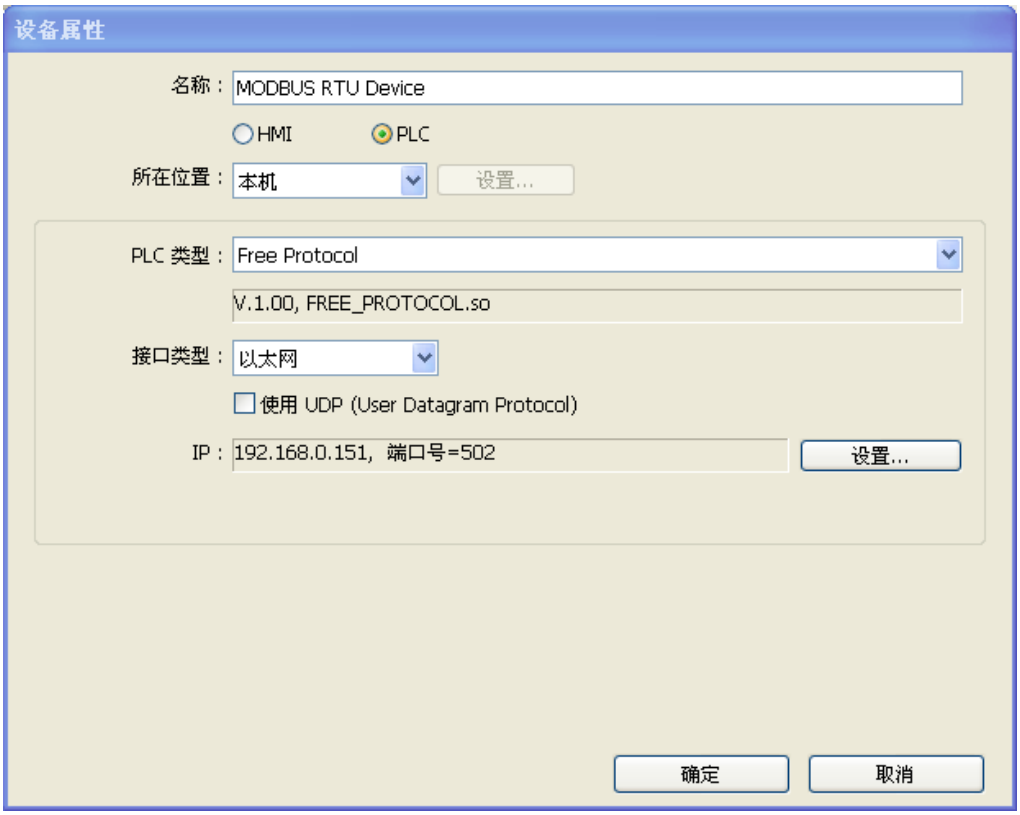
18.8 使用自由协议去控制一个设备

当EasyBuilder8000还没有一个建立好的驱动程序与某一个设备通讯时，用户也可以使用宏指令中的OUTPORT和INPORT函数来实现与该设备的通讯。使用OUTPORT和INPORT函数发送和接收资料，必须遵行该设备的通讯协议。下面的范例程序说明了如何使用两个函数来控制一个MODBUS RTU设备。

首先，在系统参数/设备列表中建立一个新的设备，这个新建的“PLC 类型”设置为“Free Protocol”，“PLC名称”设置为“MODBUS RTU Device”，如下图所示：



这里的设备连接使用RS232，如果连接一个MODBUS TCP/IP设备，这个界面接口类型必须设定为“以太网口”，同时必须设定正确的IP地址和端口号，如下图所示：



假如MT8000触摸屏将要读取设备中的4X_1和4X_2两个数据寄存器。首先，使用OUTPORT函数发送读取命令给这个设备，OUTPORT函数的写法为：

OUTPORT(command[start], device_函数名称, cmd_count)

因为“MODBUS RTU Device”是一个MODBUS RTU设备，读数据的命令必须遵行MODBUS RTU协议的命令规则，所以必须使用0x03这个命令去读取4X_1和4X_2这两个寄存器的数据。下图说明了读取命令的格式（省略了设备的站号和最后的两个CRC校验）。

Request

Function code	1 Byte	0x03
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

Response

Function code	1 Byte	0x03
Byte count	1 Byte	2 x N*
Register value	N* x 2 Bytes	

*N = Quantity of Registers

Error

Error code	1 Byte	0x83
Exception code	1 Byte	01 or 02 or 03 or 04

根据这个MODBUS RTU协议，发送命令的内容如下所示：（总共8个字节）

command[0] : station number (BYTE 0)站号
command[1] : function code (BYTE 1)功能码
command[2] : high byte of starting address (BYTE 2)起始位高字节
command[3] : low byte of starting address (BYTE 3)起始位低字节
command[4] : high byte of quantity of registers (BYTE 4)数据高字节
command[5] : low byte of quantity of registers (BYTE 5)数据低字节
command[6] : low byte of 16-bit CRC (BYTE 6)CRC低字节
command[7] : high byte of 16-bit CRC (BYTE 7)CRC高字节

所以读数据的命令宏指令程序设计如下：

```
char command[32]
short address, checksum

FILL(command[0], 0, 32) // 初始化command[0]~command[31] to 0

command[0] = 0x01 // 站号
command[1] = 0x03 // 读寄存器 (功能码是 0x03)

address = 0// 起始地址 (4x_1) 是0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

read_no = 2// 总的寄存器数目为2
HIBYTE(read_no, command[4])
LOBYTE(read_no, command[5])

CRC(command[0], checksum, 6)// CRC校验计算

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])
```

最后，使用OUTPORT函数将这个读命令发送给这个MODBUS RTU设备。

OUTPORT(command[0], “MODBUS RTU Device”, 8)// 发送读命令

发送完这个命令后，使用INPORT函数读取该MODBUS RTU设备返回的命令，根据MODBUS RTU协议，这个回复的命令内容如下所示（总共9个字节）：

command[0] : station number	(BYTE 0)站号
command[1] : function code	(BYTE 1)功能码
command[2] : byte count	(BYTE 2)字节长度
command[3] : high byte of 4x_1	(BYTE 3)第一个数据的高字节
command[4] : low byte of 4x_1	(BYTE 4)第一个数据的低字节
command[5] : high byte of 4x_2	(BYTE 5)第二个数据的高字节
command[6] : high byte of 4x_2	(BYTE 6)第二个数据的低字节
command[7] : low byte of 16–bit CRC	(BYTE 7)CRC低字节
command[8] : high byte of 16–bit CRC	(BYTE 8)CRC高字节

此时，INPROT函数的语句如下：

INPORT(response[0], “MODBUS RTU Device”, 9, return_value)// 读取回复的命令

在这里，读取的真正返回的字节长度存放在变量return_value（变量类型为字节）中，如果return_value的数据为0，则表示使用INPORT读取命令失败。

根据MODBUS RTU协议，如果命令回复的正确，则response[1]必须为0x03，当读取到正确的命令后，计算出4x_1和4x_2这两个寄存器的值，并将这两个资料送到触摸屏的LW100和LW101寄存器中。

```
if (return_value >0 and response[1] == 0x3) then
  read_data[0] = response[4] + (response[3] << 8)// 计算4x_1的数据
  read_data[1] = response[6] + (response[5] << 8)// 计算4x_2的数据
  SetData(read_data[0], “Local HMI”, LW, 100, 2)//计算后的数据送到触摸屏上来显示
  end if
```

完整的宏指令程序如下：

```
//   Read Holding Registers
macro_command main()

char command[32], response[32]
short address, checksum
short read_no, return_value, read_data[2], i

FILL(command[0], 0, 32)//   initialize command[0]~command[31] to 0
FILL(response[0], 0, 32)

command[0] = 0x1//   station number
command[1] = 0x3//   read holding registers (function code is 0x3)

address = 0
address =   0//   starting address (4x_1) is 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

read_no = 2/   the total words of reading is 2 words
HIBYTE(read_no, command[4])
LOBYTE(read_no, command[5])

CRC(command[0], checksum, 6)//   calculate 16-bit CRC

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])

OUTPORT(command[0], "MODBUS RTU Device", 8 )//   send request
INPORT(response[0], "MODBUS RTU Device", 9, return_value)//   read response

if (return_value > 0 and response[1] == 0x3) then
   read_data[0] = response[4] + (response[3] << 8)//   4x_1
   read_data[1] = response[6] + (response[5] << 8)//   4x_2

   SetData(read_data[0], "Local HMI", LW, 100, 2)
end if

end macro_command
```

下面举例说明如何使用自由协议设定MODBUS RTU设备中0x_1的状态，这个是使用MODBUS RTU协议中的“写单个寄存器”的功能码“0x05”来实现。

Request

Function code	1 Byte	**0x05**
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

Response

Function code	1 Byte	**0x05**
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

Error

Error code	1 Byte	**0x85**
Exception code	1 Byte	01 or 02 or 03 or 04

完整的宏指令程序如下：

```
//   Write Single Coil (ON)
macro_command main()

char command[32], response[32]
short address, checksum
short i, return_value

FILL(command[0], 0, 32)//    initialize command[0]~ command[31] to 0
FILL(response[0], 0, 32)

command[0] = 0x1//    station number
command[1] = 0x5//    function code : write single coil
address = 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

command[4] = 0xff//    force 0x_1 on
command[5] = 0

CRC(command[0], checksum, 6)

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])
OUTPORT(command[0], "MODBUS RTU Device", 8)//    send request
INPORT(response[0], "MODBUS RTU Device", 8, return_value)//    read response

end macro_command
```

18.9 编译错误提示信息

1.错误提示格式

error C# : error描述

（#为编译错误编号）

例如：error C37 : undeclared identifier : i

当编译没有通过时，这个错误的描述内容可以参考错误资讯编号。

2. 错误描述

错误C1：出现此项错误提示时，一般是多了或者少了一个符号；

出现此项错误提示时, 有很多种可能;

例如:

```
macro_command main()
  char i, 程      //这是一个不支持的记号

end macro_command
```

错误C2: 宏指令程序只支持静态数组, 在定义数组变量时, 必须指定变量大小;

宏指令必须定义声明的数组变量的大小;

例如:

```
macro_command main()
   char i
   int g[i]   // i必须为一个数值常量
end macro_command
```

错误C3: 函数、变量名称在其有效区域内必须是唯一的;

函数名和变量名称在其有效区域内必须是唯一的;

例如:

```
macro_command main()
   int g[10], g //   'g' 重复使用
end macro_command
```

错误C4: 关键字、常数等不能作为函数名称;

系统保留的关键词和常数不能定义为函数名称;

例如:

```
sub int if()  // if为关键词, 函数名称定义错误
```

错误C5: 左括弧、右括弧没有成对出现;

语句中缺少了 '(' 或者 ')';

例如:

```
macro_command main  )  // 缺少左括号
```

错误C6: if语句中没有合法的表达式;

也就是if语句中缺少表达式

错误C7: if语句没有配对的then;

也就是if和then没有成对出现。

错误C8: 没有end if语句;

缺少了 'end if'

错误C9: end if语句前面没有配对的if;

'end if' 前面没有配对的 'if'

错误C10: 不合法的else语句;

If语句的语法结构为:

```
if 条件表达式then
        …….
[else [if 条件表达式 then ]]
        …….
end if
```

任何与以上格式不符合的语句, 在编译时就会出现错误。

错误C17: for循环不配对, next前面应有for关键字;

'for' 语句错误, 在 'next' 前面缺少 'for' 语句。

错误C18: 不合法的变量类型;

变量类型定义错误, 此处应为整数型或字符型变量。

错误C19: 此处应为赋值符号(可能缺少符号=);

此处应为赋值符号(可能缺少符号 '=')。

错误C20: 此处应为关键字to或者down(缺少关键字 'to' 或者 'down');

缺少关键词 'to' 或 'down'

错误C21: 缺少next关键字(非法的表达式, 缺少 'next');

for循环语法结构为:

for 变量 = 初值 to 终值 [step 步长]

next [变量]

不符合这种表达形式的for循环语句都是不合法的, 编译时将出现此项错误信息。

错误C22: while循环不配对, wend前面应该有关键字while;

'while' 循环不配对, 'wend' 前面应有 'while' 关键词。

错误C23: 缺少wend关键字;

缺少 'wend' 关键词;

while循环语法结构为:

while 条件表达式

…….

wend

不符合这种表达形式的while循环语句都是不合法的, 编译时将出现此项错误错误信息。

错误C24: 不合法的break语句;

不合法的 'break' 语句。break语句只能在for循环、while循环结构中使用, 且break单独成一行。

错误C25: 不合法的continue语句;

不合法的 'continue' 语句。continue语句只能在for循环、while循环中使用, 且continue单独成一行。

错误C26: 语法错误;

表达式不正确

错误C27: 不合法的运算;

表达式中出现与运算符号不匹配的操作数时, 编译时将出现此项错误信息。

例如：

```
macro_command main( )
    int a, b
    for a = 0 to 2
    b = 4 + 程// 不合法之处："程"变量没有被定义
    next a
end macro_command
```

错误C28：此处应为macro_command;

此处应为‘macro_command’。

错误C29：必须有关键字‘sub’;

此处应为‘sub’子函数的定义形式为：

```
sub 数据类型 函数名(…)
……..
end sub
```

例如：

```
sub int pow(int exp)
……..
end sub
```

不符合这种形式的函数定义，编译时将出现此项错误信息。

错误C30：参数个数不正确;

参数个数不对

错误C31：参数类型不正确;

参数数据类型不匹配。调用函数时，参数必须在数据类型、个数上一一对应才能通过编译，否则编译时将出现此项错误信息。

错误C32：变量类型不正确;

变量类型不正确

错误C33: 没有定义的函数名称;

没有定义的函数

错误C34: 不合法的数组下标表达式;

不合法的数组下标表达形式

错误C35: 不合法的数组定义;

不合法的数组定义

错误C36: 不合法的数组下标;

不合法的数组下标

错误C37: 使用没有定义或宣告的变量;

使用没有定义或宣告的变量。只能使用已经定义或宣告的变量和函数, 否则编译时将出现此项错误信息。

错误C38: PLC不支持此种地址类型;

通讯函数GetData(…)、SetData(…)的参数中有包含PLC地址类型信息, 当PLC地址类型不是此种PLC支持的地址类型时, 编译时将出现此项错误信息。

错误C39:下标必须是整数, 字符或常量;

数组的格式为:

宣告时: 数组名[常数] (常数描述数组的大小)

使用时: 数组名[整形, 字符形变量或常数]

不符合这种表达形式, 编译时将出现此项错误信息。

错误C40: 变量定义或宣告语句的前面不能有执行语句。

变量定义或宣告语句的前面不能有执行语句

例如:

```
macro_command main( )
    int    a, b
```

```
for    a = 0  to  2
b = 4 + a

int h, k // 宣告语句在此处是错误的, 在一个函数内宣告语句的前面不能有执行语句,
         // 例如   b = 4 + a

 next a
end macro_command
```

错误C41: 移位运算中, 运算元不能为浮点数;

移位运算中, 操作数不能为浮点数。

错误C42: 函数应有返回值;

函数应有返回值

错误C43: 函数不应有返回值;

函数不应有返回值

错误C44: 运算中不能有float型资料;

运算中不能有float型资料

错误C45: PLC地址错误;

PLC地址错误

错误C46: 数组的大小超过4K;

一维数组的大小超过4k

错误C47: 宏指令程序入口只能有一个;

宏指令程序入口只能有一个

错误C48: 指令入口函数不唯一;

宏指令入口函数不唯一。宏指令的入口函数只能有一个, 形式为:

```
macro_command 函数名( )
    ……
end macro_command
```

错误C49: 扩展地址内的站号大小只能从0到255;

在宏指令中, 扩展地址内的站号大小只能从0到255

例如:

```
SetData(bits[0] , ‘PLC 1’ , LB , 300#123, 100)
```

300#123中的300表示站号为300, 站号最大值只能设定为255。

错误C50: PLC名称并没有定义在系统参数的设备清单中;

在宏指令中, PLC的名称并未定义在系统参数的设备列表中

例如:

```
SetData(bits[0] , ‘PLC 1’ , LB , 200#123, 100)
```

此时并未在设备列表中发现 ‘PLC 1’

错误C51: 宏指令只能控制本机的设备;

宏指令只能控制本机的设备

例如:

```
SetData(bits[0] , ‘PLC 1’ , LB ,300#123, 100)
```

此时 ‘PLC 1’ 为连接在其它HMI上的远程设备, 所以不能被执行。

18.10 宏指令的范例程序

1) for循环,各种表达式(算术,移位元,逻辑,关系表达式)

```
macro_command main()
    int a[10], b[10], i
    b[0]= (400 + 400 << 2) / 401
    b[1]= 22 *2 – 30 % 7
    b[2]= 111 >> 2
    b[3]= 403 > 9 + 3 >= 9 + 3 < 4 + 3 <= 8 + 8 == 8
    b[4]= not 8 + 1 and 2 + 1 or 0 + 1 xor 2
    b[5]= 405 and 3 and not 0
    b[6]= 8 & 4 + 4 & 4 + 8 | 4 + 8 ^ 4
    b[7]= 6 – (〃4)
    b[8]= 0x11
    b[9]= 409
      for i = 0 to 4 step 1
        if (a[0] == 400) then
      GetData(a[0], “Device 1”, 3x, 0,9)
       GetData(b[0], “Device 1”, 3x, 11,10)
  end If
  next i
end macro_command
```

2) while, if, break语句

```
macro_command main()
    int b[10], i
    i = 5
      while i == 5 – 20 % 3
        GetData(b[1], “Device 1”, 3x, 11, 1)
          if b[1] == 100 then
          break
```

```
            end if
        wend

end macro_command
```

3) 全域变量,子函数调用

```
char g
sub int fun(int j, int k)
     int y

      SetData(j, “Local HMI”, LB, 14, 1)
      GetData(y, “Local HMI”, LB, 15, 1)
      g = y

      return y
end Sub

macro_command main()
     int a, b, i

     a = 2
     b = 3
     i = fun(a, b)
     SetData(i, “Local HMI”, LB, 16, 1)
end macro_command
```

4) if结构语句

```
macro_command main()
   int k[10], j

   for j = 0 to 10
   k[j] = j
```

```
    next j

    if k[0] == 0 then
    SetData(k[1], “Device 1”, 3x, 0, 1)
end if

if k[0] == 0 then
    SetData(k[1], “Device 1”, 3x, 0, 1)
    else
    SetData(k[2], “Device 1”, 3x, 0, 1)
end if

if k[0] == 0 then
    SetData(k[1], “Device 1”, 3x, 1, 1)
else if k[2] == 1 then
    SetData(k[3], “Device 1”, 3x, 2, 1)
end If

if k[0] == 0 then
    SetData(k[1], “Device 1”, 3x, 3, 1)
else if k[2] == 2 then
    SetData(k[3], “Device 1”, 3x, 4, 1)
    else
    SetData(k[1], “Device 1”, 3x, 5, 1)
end If
end macro_command
```

5) while和wend结构语句

```
macro_command main()
    char i = 0
    int a[13], b[14], c = 4848
    b[0] = 13
    while b[0]
```

```
        a[i] = 20 + i * 10
          if a[i] == 120  then
            c =200
              break
            end if
          i = i + 1
        wend
    SetData(c, “Device 1”, 3x, 2, 1)
end macro_command
```

6) break、continue结构

```
macro_command main()
      char i = 0
      int a[13], b[14], c = 4848
      b[0] = 13
     while b[0]
      a[i] = 20 + i * 10
      if a[i] == 120 then
         c =200
        i = i + 1
        continue
end if
        i = i + 1
         if c == 200 then
        SetData(c, “Device 1”, 3x, 2, 1)
        break
end if
wend
end macro_command
```

7)数组结构

```
macro_command main()
    int a[25], b[25], i
    b[0] = 13
        for i = 0 to b[0] step 1
            a[i] = 20 + i * 10
        next i
    SetData(a[0], “Device 1”, 3x, 0, 13)
end macro_command
```

18.11 TRACE函数

1) 宏指令新增TRACE函数，搭配使用EasyDiagnoser，可以用来检视所使用变量目前的内容。

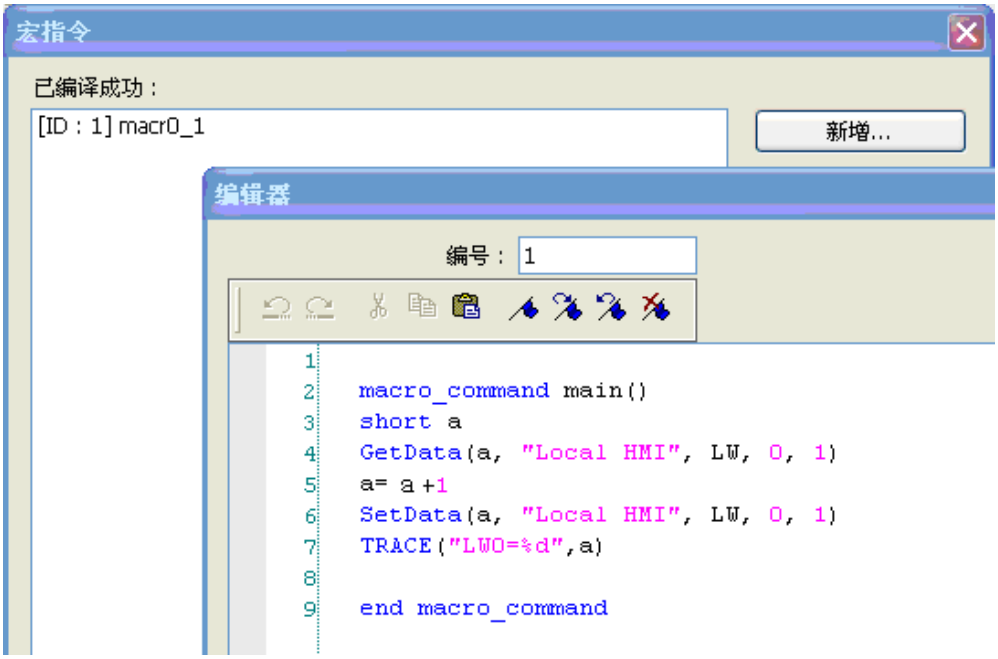
下面介绍如何在宏指令中利用TRACE命令：

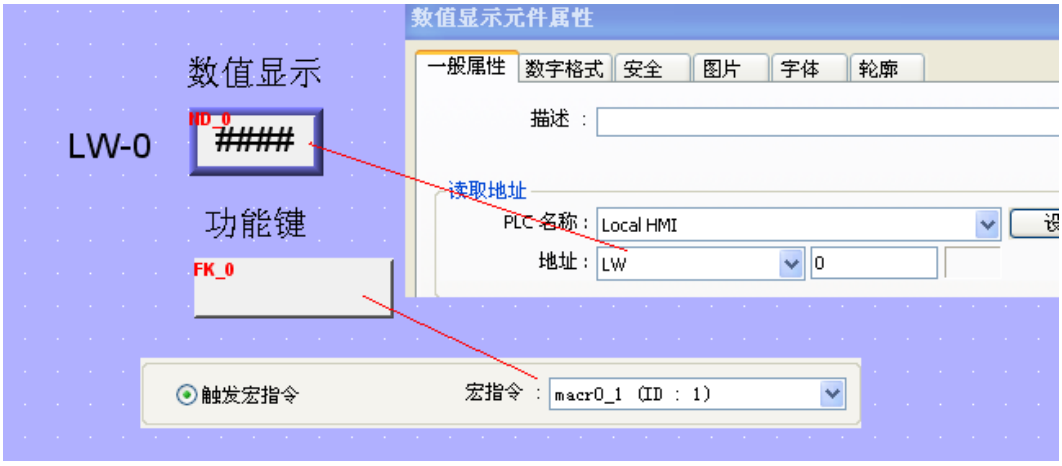
首先，请在程序中新增macro_1，并在macro_1的内容中加入TRACE(“LW = %d”, a)，“%d”表示使用10进制显示LW目前的数值。Macro_1的内容如下：

```
macro_command main()
    short a

    GetData(a, “Local HMI”, LW, 0, 1)

    a= a + 1

    SetData(a, “Local HMI”, LW, 0, 1)

    TRACE(“LW0 = %d”, a)

end macro_command
```

(TRACE命令更详细的用法请参考下面的说明）

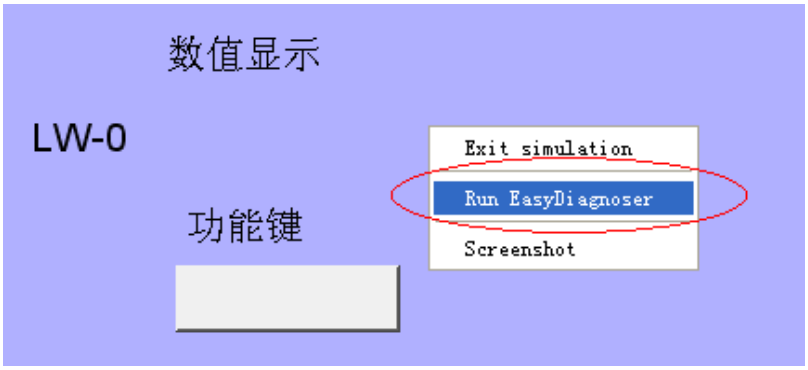
接着在程序的第10页上分别加上“数值显示”和“功能键”元件，元件的设定内容请参考下图，功能键用来执行macro_1。



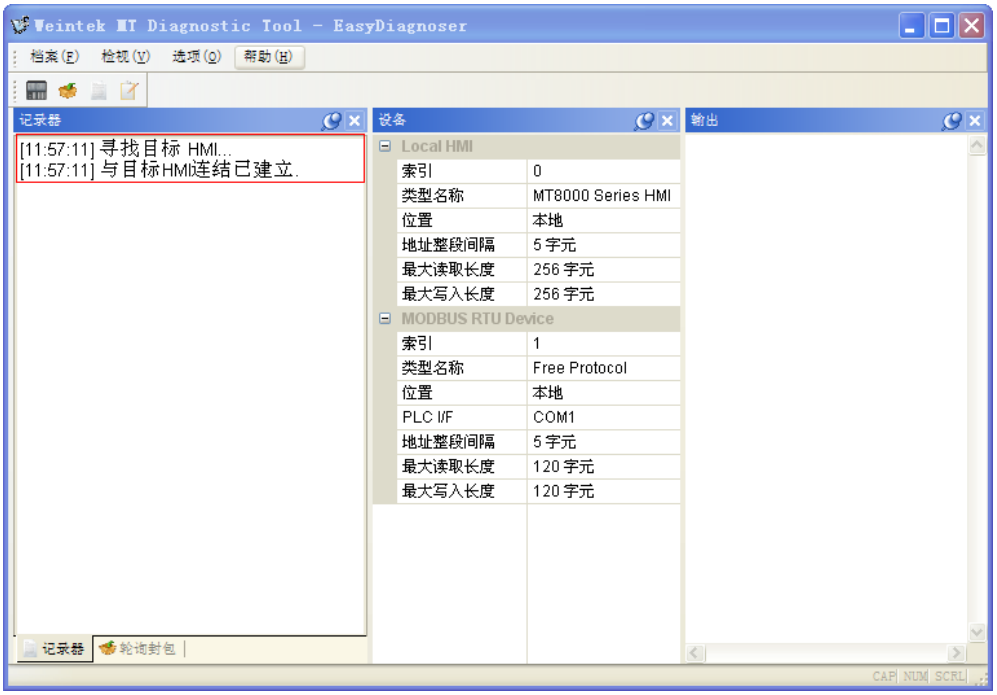


最后编译已完成的工程文件并执行离线模拟或在线模拟。

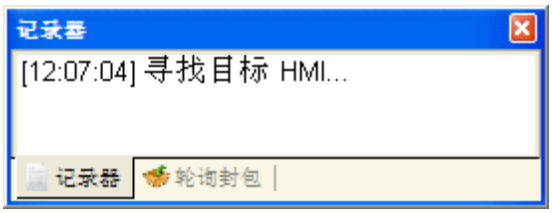
在电脑上进行模拟功能时，点击鼠标右键并选择“Run EasyDiagnoser”



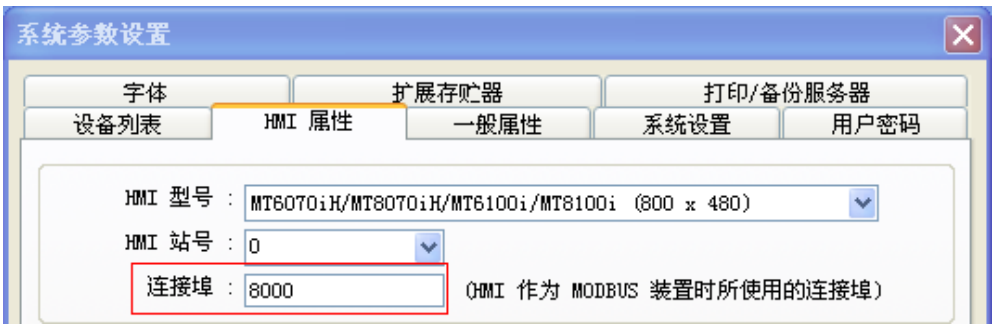
此时即会出现EasyDiagnoser的画面，“记录器”窗口用来显示Easydiagnoser是否可以连接上需要监视的触摸屏，“输出”窗口用来显示TRACE的执行结果，下图表示EasyDiagnoser已成功连接上触摸屏。



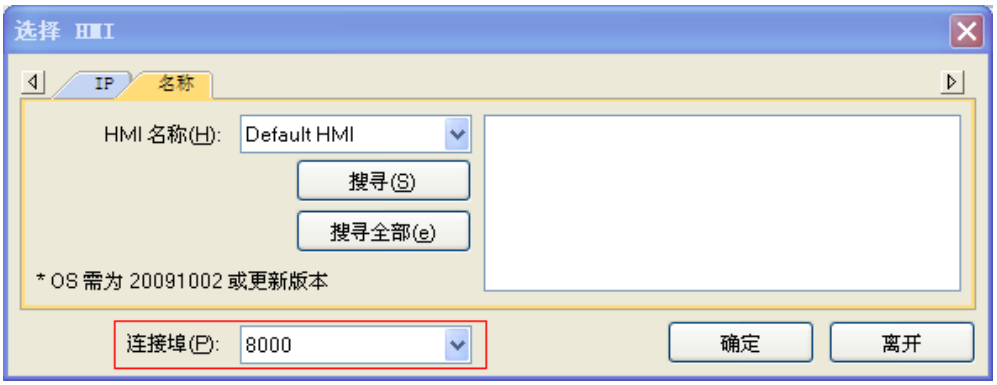
若未成功连接上触摸屏，“记录器”窗口会显示下面的内容



未连接成功可能的原因是：未成功执行模拟功能；另一个原因是在执行模拟功能的工程所使用的端口号不正确（可能已被系统占用），此时请更改工程的端口号（请参考下图），再次编译重新执行模拟功能即可。



打开Easydiagnoser时，应设定与工程相同的端口号，才能进行通讯。



从所设定的端口号算起，连续三个端口将保留给触摸屏做通讯使用。例如上图中设定的端口号为8000，则8000、8001、8002三个端口将被保留，因此在电脑上模拟时，应确定这些需要保留的端口未被其他程序所占用。

2) TRACE命令语法

函数名称	TRACE
语法	TRACE(format, argument)
描述	一个执行中的宏指令可以使用此函数，监视变量内容的变化，并打印字符，以协助除错。用户应开启EasyDiagnoser观看此函数的输出结果。 当TRACE函数抓取一个%开头的特殊字符，将同时从argument抓取一个参数做格式化后输出。 fromat代表打印格式，支持%开头的特殊字符，特殊字符的格式如下，其中方括号内的为可选，粗体字栏为必须： **%**[flags] [width] [.precision] **type** 每栏的含义如下： *flags* (可选)： - + *width* (可选)： 十进制正整数，指定应预留的字节长度，不足部分以空格补齐。 *precision* (可选)： 十进制正整数，指定精度以及输出字节数。 *type*： C 或 c ： 以字符方式输出 d ： 以 signed 十进制整数输出 i ： 以 signed 十进制整数输出 o ： 以 unsigned 八进制整数输出 u ： 以 unsigned 十进制整数输出

	X 或 x ： 以 unsigned 十六进制整数输出 E 或 e ： 以科学表示法输出 f ： 以单精度浮点数输出 *format* 字串最长支持 256 个字节，多出的字节将被忽略。 *Argument* 部分可写可不写，但一个特殊字符应搭配一个变量。

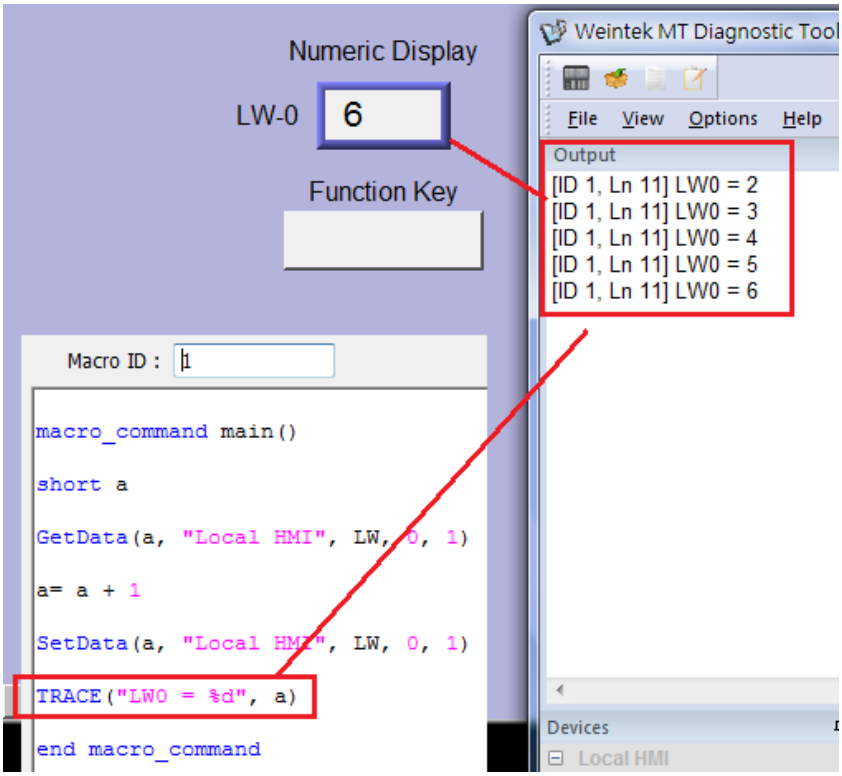
舉例	macro_command main()
	char c1 = 'a'
	short s1 = 32767
	float f1 = 1.234567
	
	TRACE("The results are") // 输出： The results are
	TRACE("c1 = %c, s1 = %d, f1 = %f", c1, s1, f1)
	// 输出： c1 = a, s1 = 32767, f1 = 1.234567
	
	end macro_command

3) 新增LB9059–disable MACRO TRACE function（when ON）
当设定为ON时，TRACE将不会把数据输出到EasyDiagnoser。

4) 也可以直接利用Project Manager执行EasyDiagnoser.exe，Project Manager将会显示网络上目前存在的触摸屏，此时只要选择要监控通讯状态的触摸屏即可。请注意Project Port的部分应设定与工程文件所使用的端口号相同

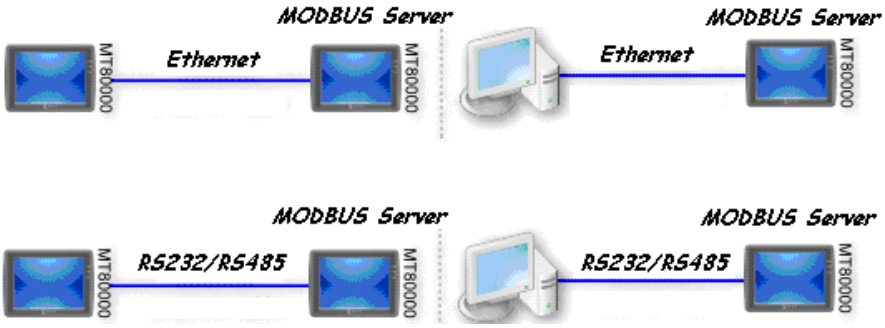
5) 将工程文件下载到触摸屏实际操作，当EasyDiagnoser无法连接上需要监视的触摸屏式，一般可能的原因是触摸屏未上电，或是端口号不正确，可能发生EasyDiagnoser不断连上触摸屏又断线的情况。EasyDiagnoser应设定与工程文件相同的端口号，更改方式如前面所述。

6) 当EasyDiagnoser成功与触摸屏连接后，只要执行macro_1，即可发现“输出”端口显示目前TRACE的执行结果。


Numeric Display
LW-0
6
Function Key
Macro ID :
1
macro_command main()
short a
GetData(a, "Local HMI", LW, 0, 1)
a= a + 1
SetData(a, "Local HMI", LW, 0, 1)
TRACE("LW0 = %d", a)
end macro_command
Weintek MT Diagnostic Tool
File View Options Help
Output
[ID 1, Ln 11] LW0 = 2
[ID 1, Ln 11] LW0 = 3
[ID 1, Ln 11] LW0 = 4
[ID 1, Ln 11] LW0 = 5
[ID 1, Ln 11] LW0 = 6
Devices
Local HMI

第十九章 如何将 HMI 设定成MODBUS设备

将 HMI 设定成MODBUS设备后，透过MODBUS协议即可读写HMI上的数据。

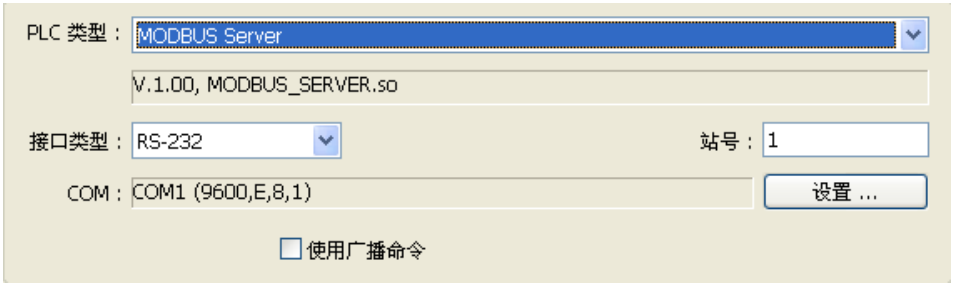


上图显示MT8000被设定成MODBUS设备(又称为MODBUS Server)，HMI、PC或其它设备只需使用MODBUS协议，透过以太网或RS232/485接口，即可读写MT8000上的数据。下文将说明如何将HMI设定成MODBUS设备，并说明读写HMI数据的方式。

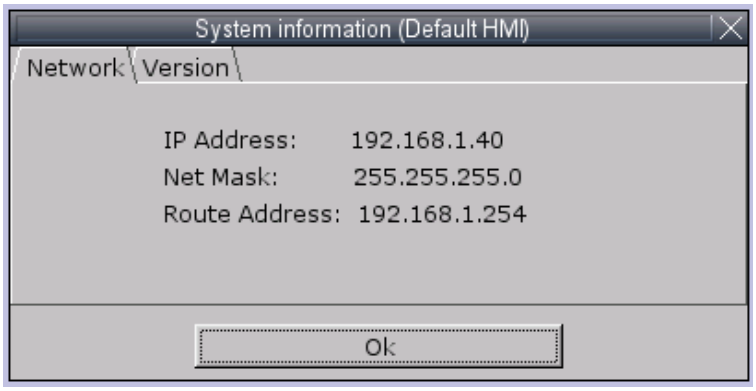
19.1 建立一个MODBUS Server设备

要将 HMI 设定为MODBUS设备，首先需在HMI使用的MTP程序的设备列表(device table)中增加一个新的设备，此时PLC种类需选择“MODBUS Server”，PLC接口可以选择RS232、RS485 2W、RS485 4W或以太网。

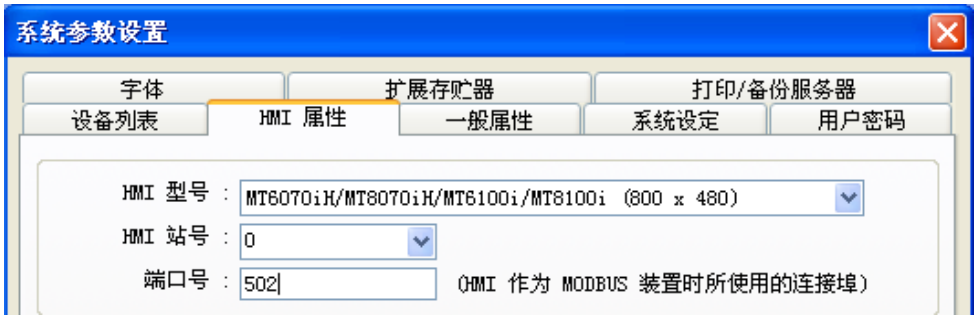
当接口选择使用RS232或RS485时，需选择使用的连接端口(COM 1 ~COM 3)，并设定正确的通讯参数，参考下图，此时MODBUS Server的站号设定为1。



当接口选择使用以太网时，IP地址等同于触摸屏的IP地址，如下图：



因MODBUS Server与HMI使用相同的连接端口，要更改MODBUS Server的连接端口需在[HMI属性]设定页中更改。

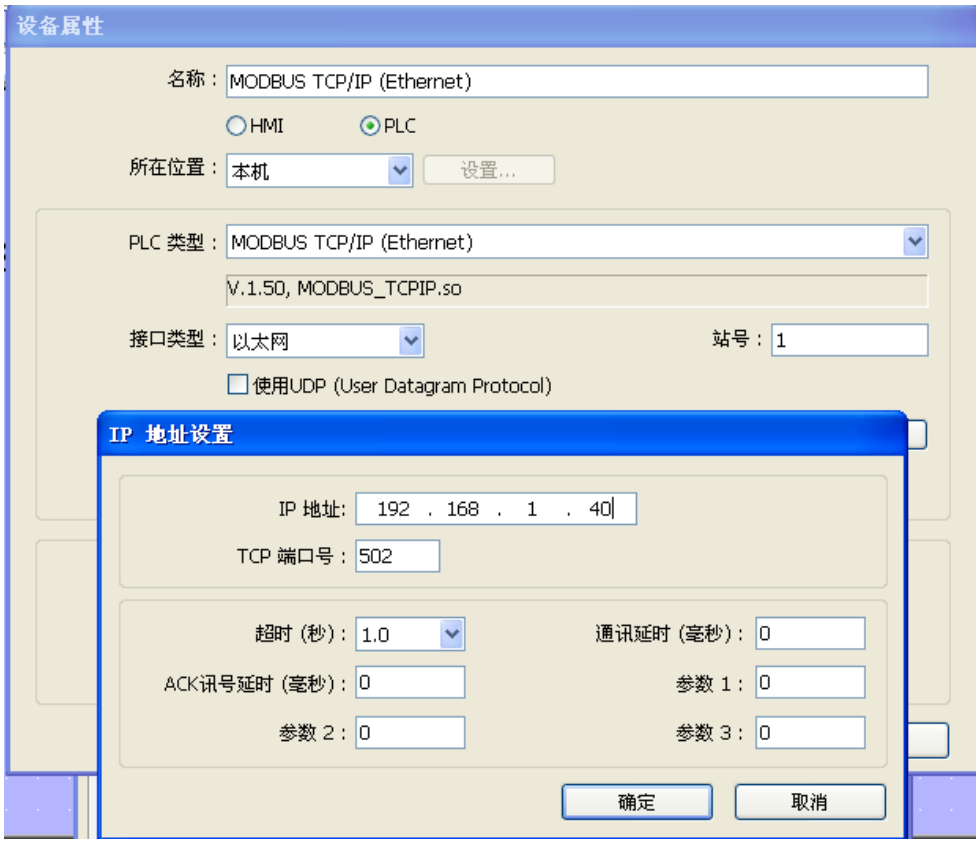


在触控“确定”键后可以在设备列表中发现一个新的设备：MODBUS Server，此时即完成了MODBUS设备的设定，在完成MTP文件的编译并将获得的XOB文件下载到HMI后，即可透过MODBUS协议读写HMI上的数据。

19.2 如何读写一个MODBUS Server设备

一台MT8000(又称为Client端)透过MODBUS协议可以读写另一台已设定为MODBUS设备的MT8000(又称为Server端)。

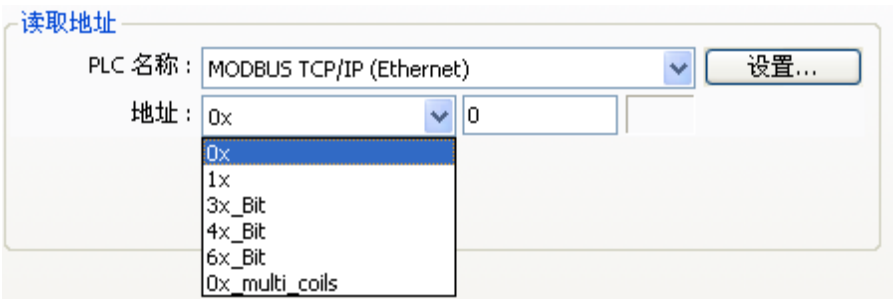
首先在Client端所使用的MTP文件的设备列表中，需增加一个新的设备，此时Client端若使用以太网接口。则PLC种类需选择“MODBUS TCP/IP (Ethernet)”，并正确设定IP(即Server端所在位置的IP)、连接端口与站号。



若client端要使用RS232/485接口。则PLC种类需选择 MODBUS RTU,并正确设定各项通讯参数。

完成各项设定并触控“确定”键后,即可在设备列表中发现一个新的设备:“MODBUS TCP/IP (Ethernet)”。

打开各个对象的设定页,在PLC名称中选择“MODBUS TCP/IP (Ethernet)”后,即可发现可以设定MODBUS设备的各项读写地址。



此时因被读写的设备(server端)为MT8000，所以实际读写的位置的对应关系如下：

读写0x/1x(1～9999)	对应到 读写LB(0～9998)
读写3x/4x/5x(1～9999)	对应到 读写LW(0～9998)
读写3x/4x/5x(10000～75535)	对应到 读写RW(0～65535)

19.3 如何在线更改MODBUS Server的站号

EB8000提供下列系统保留寄存器，让用户可以在在线更改 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)

19.4 关于MODBUS各地址的说明

EB8000软件中MODBUS协议的设备类型为0x、1x、3x、4x、5x、6x，还有3x_bit，4x_bit，6x_bit，0x_multi_coils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。该设备类型读取位状态的时候，发出的功能码是01H，写位状态的时候发出的功能码是05H。写多个寄存器时发出的功能码是0fH。

1x：是一个只读的设备类型，相当于读取PLC的输入点。读取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，相当于读取PLC的模拟量。读数据的时候，发出的功能码是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。当读取数据的时候，发出的功能码是03H，当写数据的时候发出的功能码时10H，可写多个寄存器的数据。

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0x12345678，那么使用5x设备类型读到的数据是0x56781234。

6x：是一个可读可写的设备类型，读取数据的时候，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时06H，即写单个寄存器的数据。

3x_bit：该设备类型发出的功能码与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位的状态。

4x_bit：该设备类型发出的功能码与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位的状态。

6x_bit：该设备类型发出的功能码与6x设备类型完全一致，不同之处是，6x是读数据，而6x_bit是读数据中的某一个位的状态。

0x_multi_coils：该设备类型发出的功能码与0x设备类型完全一致，不同之处是，0x是读16整数倍位数的bit，而0x_multi_coils可读任意位数的bit。

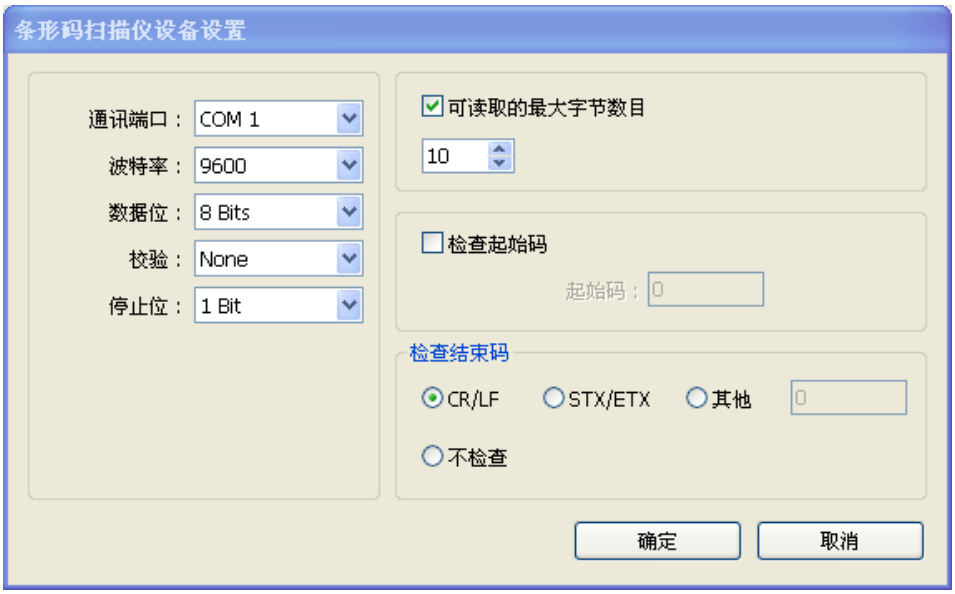
第二十章 如何使用Barcode设备

Barcode设备，指的是能够读取条形码的设备，如条码扫描枪等。

1. 如何使用Barcode设备

要使MT8000可以读取barcode设备的信号，首先须在“设备列表”中增加一个barcode设备，参考下图：

触控上图的“设置”键后，可以设定barcode设备的各项属性，参考下图：



[通讯端口]、[波特率]、[数据位]、[校验]、 [停止位]

设定barcode设备的通讯参数，barcode设备需连接至COM1～COM3

[可读取的最大字节数目]

选用此项设定会限制可以读取的字节数目，以避免barcode设备读取过多的数据，此设定范围为0～512。

例如：目前设定“可读取的最大字节数”为10，假如barcode设备原来应该读取到的数据为：

0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37 0x30 0x38 0x33 0x38

但因为限制读取数据为10个字节，所以真正读取到的数据为：

0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37 0x30 0x38

[检查起始码]

选用此项设定时，barcode设备所读取的第一个数据必须与起始码相同，MT8000才会将读取的数据视为合法的输入，否则会忽略目前读取到的数据。

起始码不会被存放barcode设备对应的地址中，例如：起始码为255（0xff），且读取到的数据为：

0xff 0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37

则实际存放在barcode设备对应地址中的数据为：

0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37

[检查结束码]

结束码用来标示数据的结尾，当读取到结束码时，表示读取到一笔完整的数据。

[CR/LF] 0x0a或0x0d皆为结束码

[STX/ETX] 0x02或0x03皆为结束码

[其他] 由用户设定数据的结束码

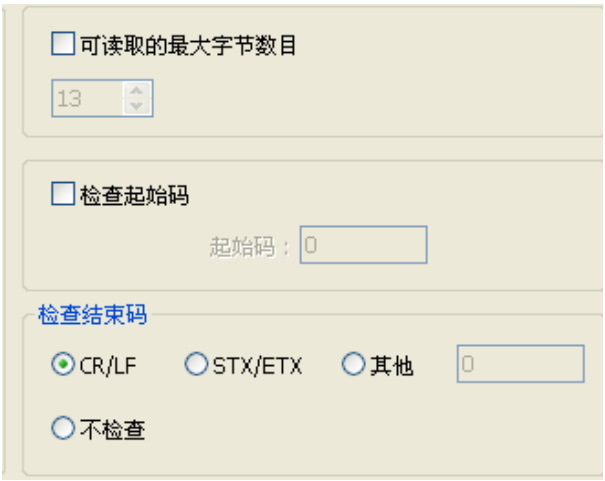
[不检查] 选择此项设定后，MT8000会将全部读取到的数据存放至barcode设备所对应的地址中，完成各项设定后，会在“设备列表”中增加一个新的设备barcode：

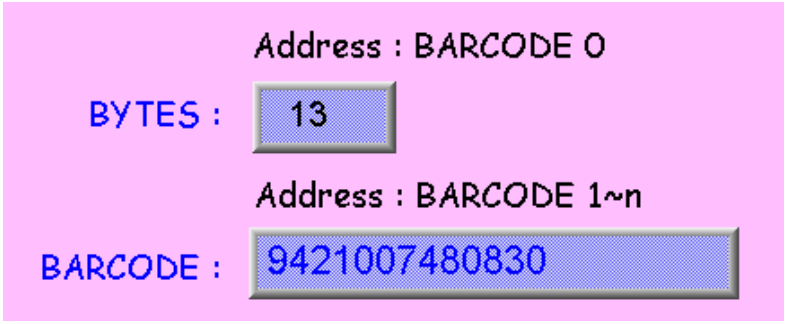
编号	名称	位置	设备类型	接口类型	通讯
本机 触摸屏	Local HMI	本机	MT6070iH/MT8070...	停用	N/A
本机 服務器	Barcode (USB/COM)	本机	Barcode (USB/COM)	COM1 (9600, N, 8, 1)	RS23

此时在元件的设定页中的“PLC名称”中可选择barcode设备，并可发现它提供两种地址类型：

地址名称	地址类型	用途
FLAG	位地址	FLAG0 用来指示数据是否完成读取，当读取到一笔完整的数据时，FLAG0 状态将由 OFF 变为 ON
BARCODE	字地址	BARCODE 0　记录目前读取到的字节数 BARCODE 1~n　存放 barcode 装置读取的数据

假如目前barcode设备的设定如下，且读取到的barcode为9421007480830，其中“数据显示”元件（字节数）的地址为BARCODE 0，“字元显示”元件（barcode数据）的地址为BARCODE 1~n。





此时barcode设备对应的地址存放的数据如下：

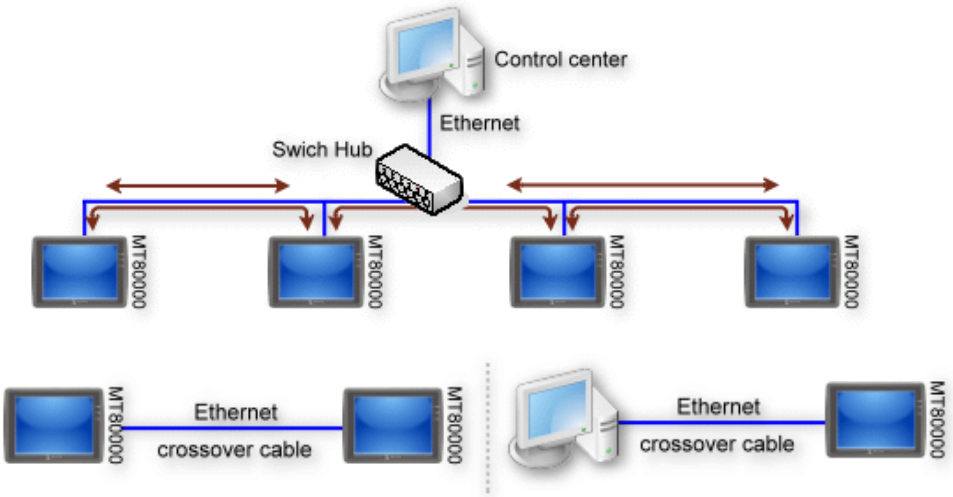
Barcode 装置對應地址	數據
BARCODE 0	13 bytes(decimal) 但实际存入地址中的数据为 14 bytes = 7 words 也就是当读取字节数目为奇数时，会加上一个字节的数据(0x00)
BARCODE 1	3439HEX
BARCODE 2	3132HEX
BARCODE 3	3030HEX
BARCODE 4	3437HEX
BARCODE 5	3038HEX
BARCODE 6	3338HEX
BARCODE 7	0030HEX
BARCODE 8	不使用

第二十一章 以太网络通讯与多台 HMI 联机

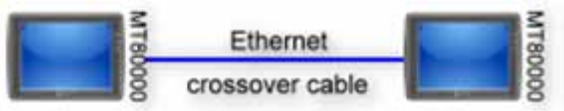
利用以太网络联机，EB8000提供了下列几种数据交换的方式：

1. HMI与HMI间的通讯
2. PC与HMI间的通讯
3. 控制连接在其它HMI上的PLC

以太网络连接的方式有两种；可以使用RJ45平行网络线(straight through cable)，搭配集线器(hub)使用；另一种是使用RJ45交叉网络线(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PC对HMI)。下面说明各项联机方式的设定与操作。



21.1 HMI与HMI间的通讯



不同的HMI间透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)，HMI可以控制另一台HMI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 “位状态设置” 元件控制HMI B的[LB123]地址的内容，则使用在HMI A上的工程文件(MTP)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“192.168.1.11”与“192.168.1.12”。

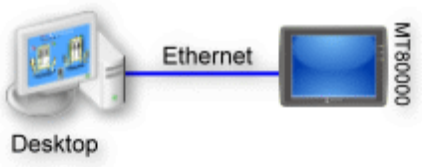
步骤二

设定HMI A上的工程档案, 在EB8000的[系统参数设定]中的[设备列表], 增加一台远程HMI (HMI B)。下图为HMI A所使用MTP档案的设备列表设定内容。

步骤三

在“位状态设置”元件设定页的[PLC名称]中, 选择“HMI B”, 此时HMI A即可操作HMI B的LB123地址之内容

21.2 PC与HMI间的通讯



利用EB8000的在线模拟功能，PC可以透过以太网络撷取HMI上的数据，并可以将这些数据保存在PC上。

PC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PC的行为表现，例如要求PC储存HMI或PLC上的数据。

一台PC可以控制不限数目的HMI。

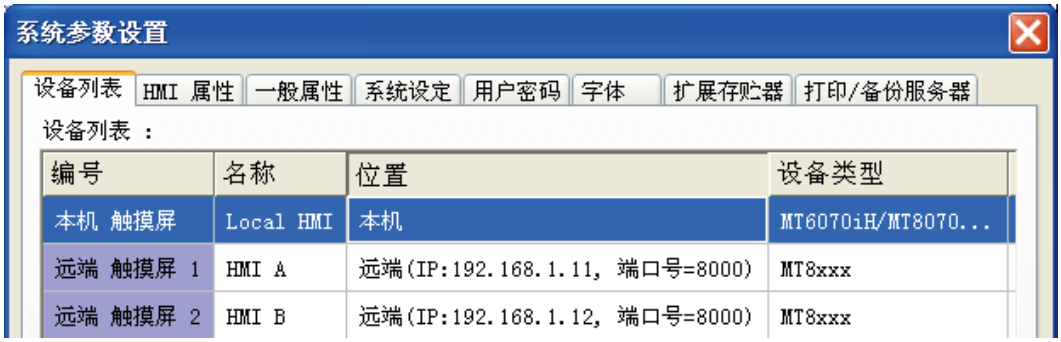
假使PC欲通讯的对象为与两台HMI (HMI A与 HMI B)，则PC端所使用MTP文件的设定步骤如下：

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“192.168.1.11”与“192.168.1.12”。

步骤二

设定HMI A上的工程档案，在 EB8000的[系统参数设定]中的[设备列表]，增加一台远程HMI (HMI B)。下图为HMI A所使用MTP档案的设备列表设定内容。



步骤三

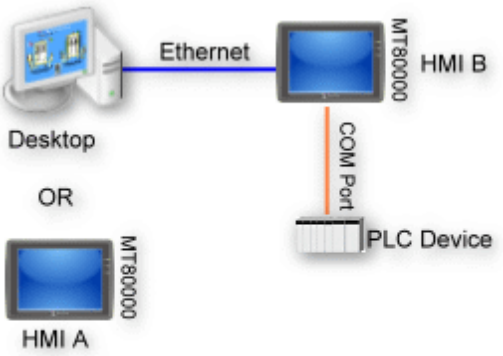
在“位状态设置”元件设定页的[PLC名称]中，选择“HMI B”，此时HMI A即可操作HMI B的LB123地址之内容。

步骤四

在PC端利用这个MTP文件，使用EB8000执行模拟功能(离线或在线模式皆可)，即可在PC端操作HMI A与HMI B上的所有数据。

HMI也允许操作PC上的数据，此时只需将PC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远端HMI，并将此远端HMI的IP指向PC即可。

21.3 控制连接在其它HMI上的PLC



透过以太网络，PC或HMI可以操作连接在其它HMI上的PLC；举例来说，假设现有一台三菱PLC连接到HMI B的COM1，当PC或HMI A欲读取此台PLC上的数据，则PC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“192.168.1.12”。

步骤二

设定PC或HMI A的工程文件，在EB8000[系统参数设定]里的[设备列表]中，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”，因为该台远端PLC是连接在HMI B上，所以设定的IP为HMI B的IP (192.168.1.12)。

步骤三

假设要使用“位状态设置”组件控制HMI B上的三菱PLC，则只需将组件设定页中的[PLC名称]选择“PLC on HMI B”，即可在PC使用模拟的方式，控制连接在远程HMI B上的PLC了。

第二十二章 HMI 状态监控(系统保留寄存器地址)

EB8000软件保留了一些位和字的寄存器给系统使用，这些系统保留寄存器分别有不同的功用，我们将系统保留寄存器地址分类如下：

22.1 本机HMI内存地址范围

- 位地址

内存地址	设备类型	地址范围	地址格式
本机位地址寄存器	**LB**	**0~12095**	**DDDDD**
本机字地址寄存器取位寄存器	**LW_BIT**	**0~1079915**	**DDDDDdd** **DDDDD:**字地址 **dd:** 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

保存寄存器位地址索引寄存器	**RBI**	**0~65535f**	**DDDDDh** **DDDDD:**字地址 **h:**位地址 **(0~f)** 举例 **567a** **RW_Bit 地址 = 567 + [LW9000]** 位地址 **= a**，即bit10
保存寄存器（RW）的位地址	**RW_Bit**	**0~524287f**	**DDDDDh** **DDDDD:**字地址 **h:**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**	**0~65535f**	**DDDDDh** **DDDDD:**字地址 **h:**位地址**(0~f)** 举例： **567a** 字地址 **= 567** 位地址**= a**

- 字地址

内存地址	设备类型	地址范围	地址格式
本机字地址	**LW**	**0~10799**	**DDDDD** **DDDDD:**字地址
本机保持寄存器	**RW**	**0~524287**	**DDDDD** **DDDDD:**字地址
保持寄存器索引寄存器	**RWI**	**0~65535**	**DDDDD** **DDDDD:**字地址 举例： **567** **RW address = 567 + [LW9000]**
本机保持寄存器 RW_A	**RW_A**	**0~65535**	**DDDDD** **DDDDD:**字地址
本机扩展寄存器	**EM0~EM9**	**DDDDDDDDDD** 最小容量由外部设备决定，最大容量为 2 GB	

22.2 一般状态与控制

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9000~ LB9009	重新开机时状态为 ON ◆HMI 开机时，这些位的状态将被初始化为 ON	读/写	读/写	读/写
LB-9010	资料下载指示	读	读	读
LB-9011	资料上传指示	读	读	读
LB-9012	资料下载/上传指示	读	读	读
LB-9017	取消 PLC 控制元件[切换窗口]的[写回]功能	读/写	读/写	读/写
LB-9039	档案备份动作状态(备份中状态为 ON)	读	读	读
LW-9050	目前机器上所显示的基本窗口之编号	读	读	读
LW-9100	（16bit）：画面规划文件的名称（16 words）	读	读	读
LW-9116	（32bit）：画面规划文件的大小(单位：byte)	读	读	读
LW-9118	（32bit）：画面规划文件的大小(单位：K byte)	读	读	读
LW-9120	（32bit）：画面规划文件所使用的编译器版本	读	读	读
LW-9122	（16bit）：画面规划文件编译时间（年）	读	读	读
LW-9123	（16bit）：画面规划文件编译时间（月）	读	读	读
LW-9124	（16bit）：画面规划文件编译时间（日）	读	读	读
LW-9125	（16bit）：HMI 以太网所使用的闸道 0（只在机器上有效）	读/写	读/写	读/写
LW-9126	（16bit）：HMI 以太网所使用的闸道 1（只在机器上有效）	读/写	读/写	读/写
LW-9127	（16bit）：HMI 以太网所使用的闸道 2（只在机器上有效）	读/写	读/写	读/写
LW-9128	（16bit）：HMI 以太网所使用的闸道 3（只在机器上有效）	读/写	读/写	读/写
LW-9129	（16bit）：HMI 以太网所使用的 IP0（只在机器上有效）	读/写	读/写	读/写
LW-9130	（16bit）：HMI 以太网所使用的 IP1（只在机器上有效）	读/写	读/写	读/写
LW-9131	（16bit）：HMI 以太网所使用的 IP2（只在机器上有效）	读/写	读/写	读/写
LW-9132	（16bit）：HMI 以太网所使用的 IP3（只在机器上有效）	读/写	读/写	读/写
LW-9133	（16bit）：以太网所使用的连接埠（只在机器上有效）	读/写	读/写	读/写
LW-9134	（16bit）：目前所使用的语言(0~7)	读/写	读/写	读/写
LW-9135	（16bit）：实体地址（MAC）0	读	读	读
LW-9136	（16bit）：实体地址（MAC）1	读	读	读
LW-9137	（16bit）：实体地址（MAC）2	读	读	读

LW-9138	（16bit）：实体地址（MAC）3	读	读	读
LW-9139	（16bit）：实体地址（MAC）4	读	读	读
LW-9140	（16bit）：实体地址（MAC）5	读	读	读
LW-10750	（16bit）：HMI 以太网所使用的掩码 0（只在机器上有效）	读/写	读/写	读/写
LW-10751	（16bit）：HMI 以太网所使用的掩码 1（只在机	读/写	读/写	读/写
LW-10752	（16bit）：HMI 以太网所使用的掩码 2（只在机器上有效）	读/写	读/写	读/写
LW-10753	（16bit）：HMI 以太网所使用的掩码 3（只在机器上有效）		/	读/写

22.3 数值输入状态

地址	说明	读写控制	Marco	远程 HMI 控制
LW-9002	（32bit-float）：数值输入元件允许输入的上限值	读	读	读
LW-9004	（32bit-float）：数值输入元件允许输入的下限值	读	读	读
LW-9052	（32bit-float）：数值输入元件的前一次输入值	读	读	读
LW-9150	（32 words）：显示目前键盘所输入的资料（ASCII）	读	读	读
LW-9540	（16bit）：键盘大小写切换	读	读	读

22.4 配方及扩展存储器

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9028	重置配方资料（设置为 ON）	写	写	写
LB-9029	强迫存贮配方资料到触摸屏（设置为 ON）	写	写	写
LB-9460	EM0 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9461	EM1 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9462	EM2 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9463	EM3 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9464	EM4 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9465	EM5 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9466	EM6 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9467	EM7 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9468	EM8 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9469	EM9 的存储装置（SD 卡）不存在（当状态为 ON）	读	读	读
LB-9470	EM0 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9471	EM1 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9472	EM2 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9473	EM3 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9474	EM4 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读

LB-9475	EM5 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9476	EM6 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9477	EM7 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9478	EM8 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9479	EM9 的存储装置（U 盘 1）不存在（当状态为 ON）	读	读	读
LB-9480	EM0 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9481	EM1 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9482	EM2 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9483	EM3 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9484	EM4 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9485	EM5 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9486	EM6 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9487	EM7 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9488	EM8 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读
LB-9489	EM9 的存储装置（U 盘 2）不存在（当状态为 ON）	读	读	读

22.5 快选窗口控制

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9013	快选窗口控制[隐藏（ON）/显示（OFF）]	读/写	读/写	读/写
LB-9014	快选按键控制[隐藏（ON）/显示（OFF）]	读/写	读/写	读/写
LB-9015	快选窗口/按键控制[隐藏（ON）/显示（OFF）]	读/写	读/写	读/写

22.6 事件记录

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9021	重置目前的事件记录（设置为 ON）	写	写	写
LB-9022	删除 HMI 内存里日期最早的事件记录档案（设定为 ON）	写	写	写
LB-9023	删除 HMI 内存里全部事件记录档案（设定为 ON）	写	写	写
LB-9024	更新 HMI 内存里事件记录统计信息（设定为 ON）	写	写	写
LB-9042	确认全部事件记录（设置为 ON）	写	写	写
LB-9043	存在未经确认的事件记录(当状态为 ON)	读	读	读
LB-11940	删除 SD 卡里日期最早的事件记录档案（设定为 ON）	写	写	写
LB-11941	删除 SD 卡里全部事件记录档案（设定为 ON）	写	写	写
LB-11942	更新 SD 卡里事件记录统计信息（设定为 ON）	写	写	写
LB-11943	删除 U 盘 1 里日期最早的事件记录档案（设定为 ON）	写	写	写
LB-11944	删除 U 盘 1 里全部事件记录档案（设定为 ON）	写	写	写
LB-11945	更新 U 盘 1 里事件记录统计信息（设定为 ON）	写	写	写
LB-11946	删除 U 盘 2 里日期最早的事件记录档案（设定为 ON）	写	写	写

LB-11947	删除 U 盘 2 里全部事件记录档案（设定为 ON）	写	写	写
LB-11948	更新 U 盘 2 里事件记录统计信息（设定为 ON）	写	写	写
LW-9060	（16bit）：目前 HMI 内存里存在的事件记录档案的数目	读	读	读
LW-9061	（32bit）：HMI 内存里全部事件记录档案的大小	读	读	读
LW-9450	（16bit）：事件登录的时间标签-秒	读/写	读/写	读/写
LW-9451	（16bit）：事件登录的时间标签-分	读/写	读/写	读/写
LW-9452	（16bit）：事件登录的时间标签-时	读/写	读/写	读/写
LW-9453	（16bit）：事件登录的时间标签-日	读/写	读/写	读/写
LW-9454	（16bit）：事件登录的时间标签-月	读/写	读/写	读/写
LW-9455	（16bit）：事件登录的时间标签-年	读/写	读/写	读/写
LW-10480	（16bit）：目前 SD 卡里存在的事件记录档案的数目	读	读	读
LW-10481	（32bit）： SD 卡里全部事件记录档案的大小	读	读	读
LW-10483	（16bit）：目前 U 盘 1 里存在的事件记录档案的数目	读	读	读
LW-10484	（32bit）： U 盘 1 里全部事件记录档案的大小	读	读	读
LW-10486	（16bit）：目前 U 盘 2 里存在的事件记录档案的数目	读	读	读
LW-10487	（32bit）： U 盘 2 里全部事件记录档案的大小	读	读	读

22.7 资料取样

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9025	删除 HMI 内存里日期最早的数据取样档案（设定为 ON）	写	写	写
LB-9026	删除 HMI 内存里全部数据取样档案（设定为 ON）	写	写	写
LB-9027	更新 HMI 内存里数据取样统计信息（设定为 ON）	写	写	写
LB-9034	强迫储存事件记录与取样数据至 HMI，U 盘，SD 卡（设定为 ON）	写	写	写
LB-11949	删除 SD 卡里日期最早的资料取样档案（设定为 ON）	写	写	写
LB-11950	删除 SD 卡里全部资料取样档案（设定为 ON）	写	写	写
LB-11951	更新 SD 卡里资料取样统计信息（设定为 ON）	写	写	写
LB-11952	删除 U 盘 1 里日期最早的资料取样档案（设定为 ON）	写	写	写
LB-11953	删除 U 盘 1 里全部资料取样档案（设定为 ON）	写	写	写
LB-11954	更新 U 盘 1 里资料取样统计信息（设定为 ON）	写	写	写
LB-11955	删除 U 盘 2 里日期最早的资料取样档案（设定为 ON）	写	写	写
LB-11956	删除 U 盘 2 里全部资料取样档案（设定为 ON）	写	写	写
LB-11957	更新 U 盘 2 里资料取样统计信息（设定为 ON）	写	写	写
LW-9063	（16bit）：目前 HMI 内存里存在的数据取样档案的数目	读	读	读
LW-9064	（32bit）：HMI 内存里全部数据取样档案的大小	读	读	读
LW-10489	（16bit）：目前 SD 卡里存在的资料取样档案的数目	读	读	读
LW-10490	（32bit）：SD 卡里全部资料取样档案的大小	读	读	读
LW-10492	（16bit）：目前 U 盘 1 里存在的资料取样档案的数目	读	读	读
LW-10493	（32bit）：U 盘 1 里全部资料取样档案的大小	读	读	读
LW-10495	（16bit）：目前 U 盘 2 里存在的资料取样档案的数目	读	读	读
LW-10496	（32bit）：U 盘 2 里全部资料取样档案的大小	读	读	读

22.8 密码与操作等级

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9050	用户登出	写	写	写
LB-9060	密码输入错误提示	读	读	读
LB-9061	更新密码（设置为 ON）	写	写	写
LW-9219	(16bit)：用户编号（1~12）	读/写	读/写	读/写
LW-9220	(32bit)：用户密码	读/写	读/写	读/写
LW-9222	(16bit)：目前用户可操作类别 (bit 0:A, bit 1:B,bit 2:C, ...)	读	读	读
LW-9500	(32bit)：用户 1 密码	读/写	读/写	读/写
LW-9502	(32bit)：用户 2 密码	读/写	读/写	读/写
LW-9504	(32bit)：用户 3 密码	读/写	读/写	读/写
LW-9506	(32bit)：用户 4 密码	读/写	读/写	读/写
LW-9508	(32bit)：用户 5 密码	读/写	读/写	读/写
LW-9510	(32bit)：用户 6 密码	读/写	读/写	读/写
LW-9512	(32bit)：用户 7 密码	读/写	读/写	读/写
LW-9514	(32bit)：用户 8 密码	读/写	读/写	读/写
LW-9516	(32bit)：用户 9 密码	读/写	读/写	读/写
LW-9518	(32bit)：用户 10 密码	读/写	读/写	读/写
LW-9520	(32bit)：用户 11 密码	读/写	读/写	读/写
LW-9522	(32bit)：用户 12 密码	读/写	读/写	读/写

22.9 HMI时间

地址	说明	读写控制	Marco	远程 HMI 控制
LW-9010	(16bit-BCD)：本地时间(秒)	读/写	读/写	读/写
LW-9011	(16bit-BCD)：本地时间(分)	读/写	读/写	读/写
LW-9012	(16bit-BCD)：本地时间(时)	读/写	读/写	读/写
LW-9013	(16bit-BCD)：本地时间(日)	读/写	读/写	读/写
LW-9014	(16bit-BCD)：本地时间(月)	读/写	读/写	读/写
LW-9015	(16bit-BCD)：本地时间(年)	读/写	读/写	读/写
LW-9016	(16bit-BCD)：本地时间(星期)	读	读	读
LW-9017	(16bit)：本地时间(秒)	读/写	读/写	读/写
LW-9018	(16bit)：本地时间(分)	读/写	读/写	读/写
LW-9019	(16bit)：本地时间(时)	读/写	读/写	读/写
LW-9020	(16bit)：本地时间(日)	读/写	读/写	读/写
LW-9021	(16bit)：本地时间(月)	读/写	读/写	读/写
LW-9022	(16bit)：本地时间(年)	读/写	读/写	读/写

LW-9023	(16bit)：本地时间(星期)	读	读	读
LW-9030	(32bit)：系统时间(单位：0.1 秒)	读	读	读
LW-9048	(16bit)：时间（0：AM，1：PM）	读/写	读/写	读/写
LW-9049	(16bit)：本地时间（12 小时制）	读/写	读/写	读/写

22.10 HMI硬件操作

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9018	鼠标光标控制[隐藏（ON）/显示（OFF）]	读/写	读/写	读/写
LB-9019	取消/开启声音输出功能[取消（ON）/开启（OFF）]	读/写	读/写	读/写
LB-9020	系统设定列控制[显示（ON）/隐藏（OFF）]	读/写	读/写	读/写
LB-9033	取消（当状态为 ON）/启用（当状态为 OFF）HMI 上传功能(只支援 I 系列)	读/写	读/写	读/写
LB-9040	背光灯调亮（设置为 ON）	写	写	写
LB-9041	背光灯调暗（设置为 ON）	写	写	写
LB-9047	重新启动人机（设定为 ON，并当 LB9048 状态为 ON 时）	写	写	写
LB-9048	重启机制保护	读/写	读/写	读/写
LB-9062	开启硬件设定对话窗（设定为 ON）	写	写	写
LB-9063	当插上 U 盘时，弹出下载窗口控制[隐藏(ON)/显示(OFF)(只支持 I 系列)	读/写	读/写	读/写
LW-9008	（32bit-float）：电池电压(只支持 I 系列)	读	读	读
LW-9025	（16bit）：CPU 使用率	读	读	读
LW-9026	（16bit）：OS 版本（年）	读	读	读
LW-9027	（16bit）：OS 版本（月）	读	读	读
LW-9028	（16bit）：OS 版本（日）	读	读	读
LW-9040	（16bit）：背光灯目前亮度值	读	读	读
LW-9080	（16bit）：背光节能时间（单位：分钟）	读/写	读/写	读/写
LW-9081	（16bit）：屏幕保护时间（单位：分钟）	读/写	读/写	读/写

22.11 与远端HMI联机状态

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9068	自动连接远端 HMI 1(当状态为 ON)	读/写	读/写	读/写
LB-9069	自动连接远端 HMI 2(当状态为 ON)	读/写	读/写	读/写
LB-9070	自动连接远端 HMI 3(当状态为 ON)	读/写	读/写	读/写
LB-9071	自动连接远端 HMI 4(当状态为 ON)	读/写	读/写	读/写
LB-9072	自动连接远端 HMI 5(当状态为 ON)	读/写	读/写	读/写
LB-9073	自动连接远端 HMI 6(当状态为 ON)	读/写	读/写	读/写

LB-9074	自动连接远端 HMI 7(当状态为 ON)	读/写	读/写	读/写
LB-9075	自动连接远端 HMI 8(当状态为 ON)	读/写	读/写	读/写
LB-9100	与远端触摸屏 1 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9101	与远端触摸屏 2 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9102	与远端触摸屏 3 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9103	与远端触摸屏 4 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9104	与远端触摸屏 5 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9105	与远端触摸屏 6 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9106	与远端触摸屏 7 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9107	与远端触摸屏 8 的通讯状态(设 ON 重连一次)	读/写	读/写	读/写
LB-9149	当线上更改远端 HMI 的 IP 时，设 ON 重新连接远端 HMI	读/写	读/写	读/写

22.12 与PLC(COM)通讯状态

	说明	读写控制	Marco	远程HMI控制
LB-9150	自动连接 PLC1（COM1）（当状态为 ON）	读/写	读/写	读/写
LB-9151	自动连接 PLC2（COM2）（当状态为 ON）	读/写	读/写	读/写
LB-9152	自动连接 PLC3（COM3）（当状态为 ON）	读/写	读/写	读/写
LB-9200	与 PLC1 的通讯状态（站号 0，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9201	与 PLC1 的通讯状态（站号 1，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9202	与 PLC1 的通讯状态（站号 2，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9203	与 PLC1 的通讯状态（站号 3，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9204	与 PLC1 的通讯状态（站号 4，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9205	与 PLC1 的通讯状态（站号 5，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9206	与 PLC1 的通讯状态（站号 6，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9207	与 PLC1 的通讯状态（站号 7，COM1），设 ON 重连一次	读/写	读/写	读/写
LB-9500	与 PLC2 的通讯状态（站号 0，COM2），设 ON 重连一次	读/写	读/写	读/写
LB-9501	与 PLC2 的通讯状态（站号 1，COM2），设 ON 重连一次	读/写	读/写	读/写
LB-9502	与 PLC2 的通讯状态（站号 2，COM2），设 ON 重连一次	读/写	读/写	读/写

LB-9503	与 PLC2 的通讯状态（站号 3，COM2），设 ON 重连一次	读/写	读/写	读/写
LB-9504	与 PLC2 的通讯状态（站号 4，COM2），设 ON 重连一次	读/写	读/写	读/写
LB-9505	与 PLC2 的通讯状态（站号 5，COM2），设 ON 重连一次	读/写	读/写	读/写
LB-9506	与 PLC2 的通讯状态（站号 6，COM2），设 ON 重连一次	读/写	读/写	读/写
LB-9507	与 PLC2 的通讯状态（站号 7，COM2），设 ON 重连一次	读/写	读/写	读/写
LB-9800	与 PLC3 的通讯状态（站号 0，COM3），设 ON 重连一次	读/写	读/写	读/写
LB-9801	与 PLC3 的通讯状态（站号 1，COM3），设 ON 重连一次	读/写	读/写	读/写
LB-9802	与 PLC3 的通讯状态（站号 2，COM3），设 ON 重连一次	读/写	读/写	读/写
LB-9803	与 PLC3 的通讯状态（站号 3，COM3），设 ON 重连一次	读/写	读/写	读/写
LB-9804	与 PLC3 的通讯状态（站号 4，COM3），设 ON 重连一次	读/写	读/写	读/写
LB-9805	与 PLC3 的通讯状态（站号 5，COM3），设 ON 重连一次	读/写	读/写	读/写
LB-9806	与 PLC3 的通讯状态（站号 6，COM3），设 ON 重连一次	读/写	读/写	读/写
LB-9807	与 PLC3 的通讯状态（站号 7，COM3），设 ON 重连一次	读/写	读/写	读/写

22.13 与PLC(以太网)通讯状态

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9153	自动连接 PLC4（以太网）（当状态为 ON）	读/写	读/写	读/写
LB-9154	自动连接 PLC5（以太网）（当状态为 ON）	读/写	读/写	读/写
LB-9155	自动连接 PLC6（以太网）（当状态为 ON）	读/写	读/写	读/写
LB-9156	自动连接 PLC7（以太网）（当状态为 ON）	读/写	读/写	读/写
LB-9157	自动连接 PLC8（以太网）（当状态为 ON）	读/写	读/写	读/写
LB-9158	自动连接 PLC9（以太网）（当状态为 ON）	读/写	读/写	读/写
LB-10070	当线上更改 PLC4(以太网)的 IP 或系统参数时，设 ON 重新连接 PLC	读/写	读/写	读/写
LB-10071	当线上更改 PLC5(以太网)的 IP 或系统参数时，设 ON 重新连接 PLC	读/写	读/写	读/写

LB-10072	当线上更改 PLC6(以太网)的 IP 或系统参数时，设 ON 重新连接 PLC	读/写	读/写	读/写
LB-10073	当线上更改 PLC7(以太网)的 IP 或系统参数时，设 ON 重新连接 PLC	读/写	读/写	读/写
LB-10074	当线上更改 PLC8(以太网)的 IP 或系统参数时，设 ON 重新连接 PLC	读/写	读/写	读/写
LB-10075	当线上更改 PLC9(以太网)的 IP 或系统参数时，设 ON 重新连接 PLC	读/写	读/写	读/写
LB-10100	与 PLC4 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-10400	与 PLC5 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-10700	与 PLC6 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11000	与 PLC7 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11300	与 PLC8 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11600	与 PLC9 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11900	与 PLC10 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11901	与 PLC11 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11902	与 PLC12 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11903	与 PLC13 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11904	与 PLC14 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11905	与 PLC15 的通讯状态（以太网），设 ON 时重连 PLC 一次	读/写	读/写	读/写
LB-11906	与 PLC16 的通讯状态（以太网），设 ON 时重连	读/写	读/写	读/写

22.14 与本机连接的远程HMI

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9016	当远端 HMI 连接至 HMI 时状态为 ON	读	读	读
LW-9006	（16bit）：目前连接到本机的远端 HMI 数目	读	读	读

22.15 MODBUS Server通讯

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9055	MODBUS server（COM1）接收到合法的命令（当状态为 ON）	读	读	读
LB-9056	MODBUS server（COM2）接收到合法的命令（当状态为 ON）	读	读	读
LB-9057	MODBUS server（COM3）接收到合法的命令（当状态为 ON）	读	读	读
LB-9058	MODBUS server（以太网）接收到合法的命令（当状态为 ON）	读	读	读
LW-9270	(16bit)：请求的功能码-MODBUS server（COM1）	读	读	读
LW-9271	(16bit)：请求的开始地址-MODBUS server（COM1）	读	读	读
LW-9272	(16bit)：请求的地址数目-MODBUS server（COM1）	读	读	读
LW-9275	(16bit)：请求的功能码-MODBUS server（COM2）	读	读	读
LW-9276	(16bit)：请求的开始地址-MODBUS server（COM2）	读	读	读
LW-9277	(16bit)：请求的地址数目-MODBUS server（COM2）	读	读	读
LW-9280	(16bit)：请求的功能码-MODBUS server（COM3）	读	读	读
LW-9281	(16bit)：请求的开始地址-MODBUS server（COM3）	读	读	读
LW-9282	(16bit)：请求的地址数目-MODBUS server（COM3）	读	读	读
LW-9285	(16bit)：请求的功能码-MODBUS server（以太网）	读	读	读
LW-9286	(16bit)：请求的开始地址-MODBUS server（以太网）	读	读	读
LW-9287	(16bit)：请求的地址数目-MODBUS server（以太网）	读	读	读
LW-9541	(16bit)：MODBUS/ASCII server 站号 (COM 1)	读/写	读/写	读/写
LW-9542	(16bit)：MODBUS/ASCII server 站号 (COM 2)	读/写	读/写	读/写
LW-9543	(16bit)：MODBUS/ASCII server 站号 (COM3)	读/写	读/写	读/写
LW-9544	(16bit)：MODBUS/ASCII server 站号 (以太网)	读/写	读/写	读/写
LW-9570	(32bit)：已接收的数据（bytes）（COM1 MODBUS server）	读	读	读
LW-9572	(32bit)：已接收的数据（bytes）（COM2 MODBUS server）	读	读	读

LW-9574	(32bit)：已接收的数据（bytes）（COM3 MODBUS server）	读	读	读
LW-9576	(32bit)：已接收的数据（bytes）（Ethernet MODBUS server）	读	读	读

22.16 通讯参数设定

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9030	更新 COM1 通讯参数（设定为 ON）	读/写	读/写	读/写
LB-9031	更新 COM2 通讯参数（设定为 ON）	读/写	读/写	读/写
LB-9032	更新 COM3 通讯参数（设定为 ON）	读/写	读/写	读/写
LB-9065	停用/启用 COM1 广播站号	读/写	读/写	读/写
LB-9066	停用/启用 COM2 广播站号	读/写	读/写	读/写
LB-9067	停用/启用 COM3 广播站号	读/写	读/写	读/写
LW-9550	（16bit）：COM 1 类型（0：RS232，1：RS485 2W，2：RS485 4W）	读/写	读/写	读/写
LW-9551	（16bit）：COM 1 波特率(7:1200,8:2400,0:4800,1：9600，2：19200 3：38400，4：57600，5：115200，6:187.5K)	读/写	读/写	读/写
LW-9552	（16bit）：COM 1 数据位（7：7 bits，8：8 bits）	读/写	读/写	读/写
LW-9553	（16bit）：COM 1 校验（0：none，1：even，2：odd，3：mark，4：space）	读/写	读/写	读/写
LW-9554	（16bit）：COM 1 停止位（1：1 bit，2：2 bits）	读/写	读/写	读/写
LW-9555	（16bit）：COM 2 类型（0：RS232，1：RS485 2W，2：RS485 4W）			
LW-9556	（16bit）：COM 2 波特率(7:1200,8:2400,0:4800,1：9600，2：19200 3：38400，4：57600，5：115200，6:187.5K)	读/写	读/写	读/写
LW-9557	（16bit）：COM 2 数据位（7：7 bits，8：8 bits）	读/写	读/写	读/写
LW-9558	（16bit）：COM 2 校验（0：none，1：even，2：odd，3：mark，4：space）	读/写	读/写	读/写
LW-9559	（16bit）：COM 2 停止位（1：1 bit，2：2 bits）	读/写	读/写	读/写
LW-9560	（16bit）：COM3 类型（0：RS232，1：RS485 2W）	读/写	读/写	读/写
LW-9561	（16bit）：COM3 波特率(7:1200,8:2400,0:4800,1：9600，2：19200 3：38400，4：57600，5：115200，6:187.5K)	读/写	读/写	读/写
LW-9562	（16bit）：COM 3 数据位（7：7 bits，8：8 bits）	读/写	读/写	读/写
LW-9563	（16bit）：COM 3 校验（0：none，1：even，2：odd，3：mark，4：space）	读/写	读/写	读/写
LW-9564	（16bit）：COM 3 停止位（1：1 bit，2：2 bits）	读/写	读/写	读/写
LW-9565	（16bit）：COM 1 广播站号	读/写	读/写	读/写
LW-9566	（16bit）：COM 2 广播站号	读/写	读/写	读/写
LW-9567	（16bit）：COM 3 广播站号	读/写	读/写	读/写
LW-10500	（16bit）：PLC 1 超时（单位：100ms）	读/写	读/写	读/写
LW-10501	（16bit）：PLC 1 通讯延时（单位：ms）	读/写	读/写	读/写

LW-10502	（16bit）：PLC 1 ACK 讯号延时（单位：ms）	读/写	读/写	读/写
LW-10503	（16bit）：PLC1 参数 1	读/写	读/写	读/写
LW-10504	（16bit）：PLC 1 参数 2	读/写	读/写	读/写
LW-10505	（16bit）：PLC2 超时（单位：100ms）	读/写	读/写	读/写
LW-10506	（16bit）：PLC 2 通讯延时（单位：ms）	读/写	读/写	读/写
LW-10507	（16bit）：PLC 2 ACK 讯号延时（单位：ms）	读/写	读/写	读/写
LW-10508	（16bit）：PLC 2 参数 1	读/写	读/写	读/写
LW-10509	（16bit）：PLC 2 参数 2	读/写	读/写	读/写
LW-10510	（16bit）：PLC 3 超时（单位：100ms）	读/写	读/写	读/写
LW-10511	（16bit）：PLC 3 通讯延时（单位：ms）	读/写	读/写	读/写
LW-10512	（16bit）：PLC 3 ACK 讯号延时（单位：ms）	读/写	读/写	读/写
LW-10513	（16bit）：PLC 3 参数 1	读/写	读/写	读/写
LW-10514	（16bit）：PLC 3 参数 2	读/写	读/写	读/写
LW-10515	（16bit）：PLC 4 超时（单位：100ms）	读/写	读/写	读/写
LW-10516	（16bit）：PLC 4 通讯延时（单位：ms）	读/写	读/写	读/写
LW-10517	（16bit）：PLC 4 ACK 讯号延时（单位：ms）（SIEMENS S7/400 连接类型）	读/写	读/写	读/写
LW-10518	（16bit）：PLC 4 参数 1（SIEMENS S7/400 机座）	读/写	读/写	读/写
LW-10519	（16bit）：PLC 4 参数 2（SIEMENS S7/400 CPU 插槽）	读/写	读/写	读/写
LW-10520	（16bit）：PLC 5 超时（单位：100ms）	读/写	读/写	读/写
LW-10521	（16bit）：PLC 5 通讯延时（单位：ms）	读/写	读/写	读/写
LW-10522	（16bit）：PLC 5 ACK 讯号延时（单位：ms）（SIEMENS S7/400 连接类型）	读/写	读/写	读/写
LW-10523	（16bit）：PLC5 参数 1（SIEMENS S7/400 机座）	读/写	读/写	读/写
LW-10524	（16bit）：PLC 5 参数 2（SIEMENS S7/400 CPU 插槽）	读/写	读/写	读/写
LW-10525	（16bit）：PLC 6 超时（单位：100ms）	读/写	读/写	读/写
LW-10526	（16bit）：PLC6 通讯延时（单位：ms）	读/写	读/写	读/写
LW-10527	（16bit）：PLC 6 ACK 讯号延时（单位：ms）（SIEMENS S7/400 连接类型）	读/写	读/写	读/写
LW-10528	（16bit）：PLC6 参数 1（SIEMENS S7/400 机座）	读/写	读/写	读/写
LW-10529	（16bit）：PLC 6 参数 2（SIEMENS S7/400 CPU 插槽）	读/写	读/写	读/写
LW-10530	（16bit）：PLC 7 超时（单位：100ms）	读/写	读/写	读/写
LW-10531	（16bit）：PLC 7 通讯延时（单位：ms）	读/写	读/写	读/写
LW-10532	（16bit）：PLC 7 ACK 讯号延时（单位：ms）（SIEMENS S7/400 连接类型）	读/写	读/写	读/写
LW-10533	（16bit）：PLC 7 参数 1（SIEMENS S7/400 机座）	读/写	读/写	读/写
LW-10534	（16bit）：PLC 7 参数 2（SIEMENS S7/400 CPU 插槽）	读/写	读/写	读/写
LW-10535	（16bit）：PLC 8 超时（单位：100ms）	读/写	读/写	读/写
LW-10536	（16bit）：PLC 8 通讯延时（单位：ms）	读/写	读/写	读/写
LW-10537	（16bit）：PLC 8 ACK 讯号延时（单位：ms）（SIEMENS S7/400 连接类型）	读/写	读/写	读/写
LW-10538	（16bit）：PLC 8 参数 1（SIEMENS S7/400 机座）	读/写	读/写	读/写
LW-10539	（16bit）：PLC 8 参数 2（SIEMENS S7/400 CPU 插槽）	读/写	读/写	读/写

22.17 存储空间管理

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9035	HMI可用空间不足警示 (当状态为 ON)	读	读	读
LB-9036	SD卡可用空间不足警示 (当状态为 ON)	读	读	读
LB-9037	U 盘 1可用空间不足警示 (当状态为 ON)	读	读	读
LB-9038	U 盘 2可用空间不足警示 (当状态为 ON)	读	读	读
LW-9070	（16bit）：可用空间警示下限（Mega bytes）	读	读	读
LW-9071	（16bit）：系统保留的可用空间（K bytes）	读	读	读
LW-9072	（32bit）：HMI 目前的可用空间（K bytes）	读	读	读
LW-9074	（32bit）：SD卡目前的可用空间(K bytes）	读	读	读
LW-9076	（32bit）：U 盘 1 目前的可用空间（K bytes）	读	读	读
LW-9078	（32bit）：U 盘 2 目前的可用空间（K bytes）	读	读	读

22.18 PLC & 远程 HMI的IP地址设定

地址	说明	读写控制	Marco	远程 HMI 控制
LW-9600	PLC4 的 IP0（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9601	PLC4 的 IP1（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9602	PLC4 的 IP2（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9603	PLC4 的 IP3（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9604	PLC4 的连接埠	读/写	读/写	读/写
LW-9605	PLC5 的 IP0（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9606	PLC5 的 IP1（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9607	PLC5 的 IP2（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9608	PLC5 的 IP3（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9609	PLC5 的连接埠	读/写	读/写	读/写
LW-9610	PLC6 的 IP0（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9611	PLC6 的 IP1（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9612	PLC6 的 IP2（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9613	PLC6 的 IP3（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9614	PLC6 的连接埠	读/写	读/写	读/写
LW-9615	PLC7 的 IP0（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9616	PLC7 的 IP1（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9617	PLC7 的 IP2（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9618	PLC7 的 IP3（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9619	PLC7 的连接埠	读/写	读/写	读/写
LW-9620	PLC8 的 IP0（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9621	PLC8 的 IP1（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写

LW-9622	PLC8 的 IP2（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9623	PLC8 的 IP3（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9624	PLC8 的连接埠	读/写	读/写	读/写
LW-9625	PLC9 的 IP0（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9626	PLC9 的 IP1（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9627	PLC9 的 IP2（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9628	PLC9 的 IP3（IP 地址=IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9629	PLC9 的连接埠	读/写	读/写	读/写
LW-9800	（16bit）：远端 HMI1 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9801	（16bit）：远端 HMI1 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9802	（16bit）：远端 HMI1 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9803	（16bit）：远端 HMI1 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9804	（16bit）：远端 HMI1 连接埠	读/写	读/写	读/写
LW-9805	（16bit）：远端 HMI2 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9806	（16bit）：远端 HMI2 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9807	（16bit）：远端 HMI2 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9808	（16bit）：远端 HMI2 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9809	（16bit）：远端 HMI2 连接埠	读/写	读/写	读/写
LW-9810	（16bit）：远端 HMI3 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9811	（16bit）：远端 HMI3 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9812	（16bit）：远端 HMI3 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9813	（16bit）：远端 HMI3 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9814	（16bit）：远端 HMI3 连接埠	读/写	读/写	读/写
LW-9815	（16bit）：远端 HMI4 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9816	（16bit）：远端 HMI4 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9817	（16bit）：远端 HMI4 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写

LW-9818	（16bit）：远端 HMI4 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9819	（16bit）：远端 HMI4 连接埠	读/写	读/写	读/写
LW-9820	（16bit）：远端 HMI5 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9821	（16bit）：远端 HMI5 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9822	（16bit）：远端 HMI5 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9823	（16bit）：远端 HMI5 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9824	（16bit）：远端 HMI5 连接埠	读/写	读/写	读/写
LW-9825	（16bit）：远端 HMI6 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9826	（16bit）：远端 HMI6 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9827	（16bit）：远端 HMI6 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9828	（16bit）：远端 HMI6 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9829	（16bit）：远端 HMI6 连接埠	读/写	读/写	读/写
LW-9830	（16bit）：远端 HMI7 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9831	（16bit）：远端 HMI7 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9832	（16bit）：远端 HMI7 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9833	（16bit）：远端 HMI7 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9834	（16bit）：远端 HMI7 连接埠	读/写	读/写	读/写
LW-9835	（16bit）：远端 HMI8 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9836	（16bit）：远端 HMI8 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9837	（16bit）：远端 HMI8 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9838	（16bit）：远端 HMI8 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-9839	（16bit）：远端 HMI8 连接埠	读/写	读/写	读/写

22.19 远程打印/备份服务器设定

地址	说明	读写控制	Marco	远程 HMI 控制
LB-10069	当线上更改远端打印/备份伺服器的 IP 时，设 ON 时重新连接远端打印/备份伺服器。	读/写	读/写	读/写
LW-9770	(16bit):远端打印/备份服务器的 IP0(IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9771	(16bit):远端打印/备份服务器的 IP1(IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9772	(16bit):远端打印/备份服务器的 IP2(IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9773	(16bit):远端打印/备份服务器的 IP3(IP0:IP1:IP2:IP3)	读/写	读/写	读/写
LW-9774	（6words）：登入远端打印/备份服务器所需的使用者名称	读/写	读/写	读/写
LW-9780	（6words）：登入远端打印/备份服务所需的密码	读/写	读/写	读/写

22.20 地址索引功能

地址	说明	读写控制	Marco	远程 HMI 控制
LW-9200	（16bit）：地址索引寄存器 0	读/写	读/写	读/写
LW-9201	（16bit）：地址索引寄存器 1	读/写	读/写	读/写
LW-9202	（16bit）：地址索引寄存器 2	读/写	读/写	读/写
LW-9203	（16bit）：地址索引寄存器 3	读/写	读/写	读/写
LW-9204	（16bit）：地址索引寄存器 4	读/写	读/写	读/写
LW-9205	（16bit）：地址索引寄存器 5	读/写	读/写	读/写
LW-9206	（16bit）：地址索引寄存器 6	读/写	读/写	读/写
LW-9207	（16bit）：地址索引寄存器 7	读/写	读/写	读/写
LW-9208	（16bit）：地址索引寄存器 8	读/写	读/写	读/写
LW-9209	（16bit）：地址索引寄存器 9	读/写	读/写	读/写
LW-9210	（16bit）：地址索引寄存器 10	读/写	读/写	读/写
LW-9211	（16bit）：地址索引寄存器 11	读/写	读/写	读/写
LW-9212	（16bit）：地址索引寄存器 12	读/写	读/写	读/写
LW-9213	（16bit）：地址索引寄存器 13	读/写	读/写	读/写
LW-9214	（16bit）：地址索引寄存器 14	读/写	读/写	读/写
LW-9215	（16bit）：地址索引寄存器 15	读/写	读/写	读/写
LW-9230	（32bit）：地址索引寄存器 16	读/写	读/写	读/写
LW-9232	（32bit）：地址索引寄存器 17	读/写	读/写	读/写
LW-9234	（32bit）：地址索引寄存器 18	读/写	读/写	读/写

LW-9236	（32bit）：地址索引寄存器 19	读/写	读/写	读/写
LW-9238	（32bit）：地址索引寄存器 20	读/写	读/写	读/写
LW-9240	（32bit）：地址索引寄存器 21	读/写	读/写	读/写
LW-9242	（32bit）：地址索引寄存器 22	读/写	读/写	读/写
LW-9244	（32bit）：地址索引寄存器 23	读/写	读/写	读/写
LW-9246	（32bit）：地址索引寄存器 24	读/写	读/写	读/写
LW-9248	（32bit）：地址索引寄存器 25	读/写	读/写	读/写
LW-9250	（32bit）：地址索引寄存器 26	读/写	读/写	读/写
LW-9252	（32bit）：地址索引寄存器 27	读/写	读/写	读/写
LW-9254	（32bit）：地址索引寄存器 28	读/写	读/写	读/写
LW-9256	（32bit）：地址索引寄存器 29	读/写	读/写	读/写
LW-9258	（32bit）：地址索引寄存器 30	读/写	读/写	读/写
LW-9260	（32bit）：地址索引寄存器 31	读/写	读/写	读/写

22.21 触控位置

地址	说明	读写控制	Marco	远程 HMI 控制
LW-9041	（16bit）：触控状态字组（位 0 on=使用者正在触控屏幕）	读	读	读
LW-9042	（16bit）：触控时，X 的位置	读	读	读
LW-9043	（16bit）：触控时，Y 的位置	读	读	读
LW-9044	（16bit）：离开时，X 的位置	读	读	读
LW-9045	（16bit）：离开时，Y 的位置	读	读	读

22.22 变量站号

地址	说明	读写控制	Marco	远程 HMI 控制
LW-10000	（16bit）：Var0--站号变量(语法：var0#地址)	读/写	读/写	读/写
LW-10001	（16bit）：Var1--站号变量(语法：var1#地址)	读/写	读/写	读/写
LW-10002	（16bit）：Var2--站号变量(语法：var2#地址)	读/写	读/写	读/写
LW-10003	（16bit）：Var3--站号变量(语法：var3#地址)	读/写	读/写	读/写
LW-10004	（16bit）：Var4--站号变量(语法：var4#地址)	读/写	读/写	读/写
LW-10005	（16bit）：Var5--站号变量(语法：var5#地址)	读/写	读/写	读/写
LW-10006	（16bit）：Var6--站号变量(语法：var6#地址)	读/写	读/写	读/写
LW-10007	（16bit）：Var7--站号变量(语法：var7#地址)	读/写	读/写	读/写
LW-10008	（16bit）：Var8--站号变量(语法：var8#地址)	读/写	读/写	读/写
LW-10009	（16bit）：Var9--站号变量(语法：var9#地址)	读/写	读/写	读/写
LW-10010	（16bit）：Var10--站号变量(语法：var10#地址)	读/写	读/写	读/写

LW-10011	（16bit）：Var11--站号变量(语法：var11#地址)	读/写	读/写	读/写
LW-10012	（16bit）：Var12--站号变量(语法：var12#地址)	读/写	读/写	读/写
LW-10013	（16bit）：Var13--站号变量(语法：var13#地址)	读/写	读/写	读/写
LW-10014	（16bit）：Var14--站号变量(语法：var14#地址)	读/写	读/写	读/写
LW-10015	（16bit）：Var15--站号变量(语法：var15#地址)	读/写	读/写	读/写

22.23 本地/远端操作限制

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9044	禁止远端控制（当状态为 ON）	读/写	读/写	读/写
LB-9053	禁止密码远端读取操作（当状态为 ON）	读/写	读/写	读/写
LB-9054	禁止密码远端写入操作（当状态为 ON）	读/写	读/写	读/写
LB-9196	本地 HMI 只支援检视功能（当状态为 ON）	读/写	读/写	读/写
LB-9197	只允许远端 HMI 使用检视功能（当状态为 ON）	读/写	读/写	读/写
LB-9198	禁止本地 HMI 触发宏指令（当状态为 ON）	读/写	读/写	读/写
LB-9199	禁止远端 HMI 触发宏指令（当状态为 ON）	读/写	读/写	读/写

22.24 人机与工程档案识别码

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9046	工程档案识别码与人机识别码不同（当状态为 ON）	读	读	读
LW-9046	（32bit）：人机识别码(只支持 i 系列）	读/写	读/写	读/ 写

22.25 EasyAccess

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9051	与 EasyAccess 伺服器断线（设 OFF）/连线（设 ON）	读/写	读/写	读/写
LB-9052	与 EasyAccess 伺服器连线状态(当连线中状态为 ON）	读	读	读

22.26 与PLC(USB)通讯状态

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9190	自动连接 PLC(USB1)(当状态为 0N)	读/写	读/写	读/写
LB-9191	与 PLC 的通讯状态（USB1），设 ON 重连一次	读/写	读/写	读/写
LB-9193	自动连接 PLC（USB 2）(当状态为 ON)	读/写	读/写	读/写
LB-9194	与 PLC 的通讯状态（USB2），设 ON 重连一次	读/写	读/写	读/写

22.27 禁止弹出 PLC NO Response视窗

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9192	禁止弹出 USB1 PLC 的“**PLC NO Response**”窗口（当状态为 ON）	读/写	读/写	读/写
LB-9195	禁止弹出 USB2 PLC 的“**PLC NO Response**”窗口（当状态为 ON）	读/写	读/写	读/写
LB-11960	禁止弹出 PLC1 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写
LB-11961	禁止弹出 PLC2 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写
LB-11962	禁止弹出 PLC3 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写
LB-11963	禁止弹出 PLC4 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写
LB-11964	禁止弹出 PLC5 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写
LB-11965	禁止弹出 PLC6 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写
LB-11966	禁止弹出 PLC7 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写
LB-11967	禁止弹出 PLC8 的“**PLC NO Response**”窗口 （当状态为 ON）	读/写	读/写	读/写

22.28 通讯错误信息及未处理命令数

地址	说明	读写控制	Marco	远程 HMI 控制
LW-9350	（16bit）：本机尚未处理的命令数目	读	读	读
LW-9351	（16bit）：PLC1（COM1）尚未处理的命令数目	读	读	读
LW-9352	（16bit）：PLC2（COM2）尚未处理的命令数目	读	读	读
LW-9353	（16bit）：PLC3（COM3）尚未处理的命令数目	读	读	读

LW-9354	（16bit）：PLC4（以太网）尚未处理的命令数目	读	读	读
LW-9355	（16bit）：PLC5（以太网）尚未处理的命令数目	读	读	读
LW-9356	（16bit）：PLC6（以太网）尚未处理的命令数目	读	读	读
LW-9357	（16bit）：PLC7（以太网）尚未处理的命令数目	读	读	读
LW-9390	（16bit）：PLC（USB）尚未处理的命令数目	读	读	读
LW-9400	（16bit）：与 PLC1 通讯错误时产生的错误信息	读	读	读
LW-9401	（16bit）：与 PLC2 通讯错误时产生的错误信息	读	读	读
LW-9402	（16bit）：与 PLC3 通讯错误时产生的错误信息	读	读	读
LW-9403	（16bit）：与 PLC4 通讯错误时产生的错误信息	读	读	读
LW-9404	（16bit）：与 PLC5 通讯错误时产生的错误信息	读	读	读
LW-9405	（16bit）：与 PLC6 通讯错误时产生的错误信息	读	读	读
LW-9406	（16bit）：与 PLC7 通讯错误时产生的错误信息	读	读	读
LW-9407	（16bit）：与 PLC8 通讯错误时产生的错误信息	读	读	读
LW-9490	（16bit）：与 USB PLC 通讯错误时产生的错误信息	读	读	读

22.29 穿透通讯设定

地址	说明	读写控制	Marco	远程 HMI 控制
LW-9900	（16bit）：HMI 工作模式（0：正常模式，1~3：测试模式 (COM 1~COM 3)	读/写	读/写	读/写
LW-9901	（16bit）：穿透通讯数据来源串列埠口（1~3：COM1~COM3）	读/写	读/写	读/写
LW-9902	（16bit）：穿透通讯数据目标串列埠口（1~3：COM1~COM3）	读/写	读/写	读/写

22.30 与远端PLC通讯状态

地址	说明	读写控制	Marco	远程 HMI 控制
LW-10050	（16bit）：连接远端 PLC1 的 HMI 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10051	（16bit）：连接远端 PLC1 的 HMI 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10052	（16bit）：连接远端 PLC1 的 HMI 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10053	（16bit）：连接远端 PLC1 的 HMI 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10054	（16bit）：连接远端 PLC1 的 HMI 的连接埠	读/写	读/写	读/写
LW-10055	（16bit）：连接远端 PLC2 的 HMI 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10056	（16bit）：连接远端 PLC2 的 HMI 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写

LW-10057	（16bit）：连接远端 PLC2 的 HMI 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10058	（16bit）：连接远端 PLC2 的 HMI 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10059	（16bit）：连接远端 PLC2 的 HMI 的连接埠	读/写	读/写	读/写
LW-10060	（16bit）：连接远端 PLC3 的 HMI 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10061	（16bit）：连接远端 PLC3 的 HMI 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10062	（16bit）：连接远端 PLC3 的 HMI 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10063	（16bit）：连接远端 PLC3 的 HMI 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10064	（16bit）：连接远端 PLC3 的 HMI 的连接埠	读/写	读/写	读/写
LW-10065	（16bit）：连接远端 PLC4 的 HMI 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10066	（16bit）：连接远端 PLC4 的 HMI 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10067	（16bit）：连接远端 PLC4 的 HMI 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10068	（16bit）：连接远端 PLC4 的 HMI 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10069	（16bit）：连接远端 PLC4 的 HMI 的连接埠	读/写	读/写	读/写
LW-10300	（16bit）：远端 PLC1 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10301	（16bit）：远端 PLC1 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10302	（16bit）：远端 PLC1 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10303	（16bit）：远端 PLC1 的 IP3（IP 地址=IP0：IP1：	读/写	读/写	读/写
LW-10304	（16bit）：远端 PLC1 的连接埠	读/写	读/写	读/写
LW-10305	（16bit）：远端 PLC2 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10306	（16bit）：远端 PLC2 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10307	（16bit）：远端 PLC2 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10308	（16bit）：远端 PLC2 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10309	（16bit）：远端 PLC2 的连接埠	读/写	读/写	读/写
LW-10310	（16bit）：远端 PLC3 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10311	（16bit）：远端 PLC3 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写

LW-10312	（16bit）：远端 PLC3 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10313	（16bit）：远端 PLC3 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10314	（16bit）：远端 PLC3 的连接埠	读/写	读/写	读/写
LW-10315	（16bit）：远端 PLC4 的 IP0（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10316	（16bit）：远端 PLC4 的 IP1（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10317	（16bit）：远端 PLC4 的 IP2（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10318	（16bit）：远端 PLC4 的 IP3（IP 地址=IP0：IP1：IP2：IP3）	读/写	读/写	读/写
LW-10319	（16bit）：远端 PLC4 的连接埠	读/写	读/写	读/写

22.31 DLT_645设置

地址	说明	读写控制	Marco	远程 HMI 控制
LW-10700	（4words）：DLT_645 使用者（COM1）	读/写	读/写	读/写
LW-10704	（4words）：DLT_645 密码（COM1）	读/写	读/写	读/写
LW-10708	（6words）：DLT_645 地址（COM1）	读/写	读/写	读/写
LW-10715	（4words）：DLT_645 使用者（COM2）	读/写	读/写	读/写
LW-10719	（4words）：DLT_645 密码（COM2）	读/写	读/写	读/写
LW-10723	（6words）：DLT_645 地址（COM2）	读/写	读/写	读/写
LW-10730	（4words）：DLT_645 使用者（COM3）	读/写	读/写	读/写
LW-10734	（4words）：DLT_645 密码（COM3）	读/写	读/写	读/写
LW-10738	（6words）：DLT_645 地址（COM3）	读/写	读/写	读/写

22.32 其他

地址	说明	读写控制	Marco	远程 HMI 控制
LB-9045	Memory-map 通讯失败（当状态为 ON）	读	读	读
LB-9049	启用（设 ON）/取消（设（OFF）看门狗（只支援 i 系列）	读/写	读/写	读/写
LB-9059	关闭宏指令 TRACE 功能（当状态为 ON）	读/写	读/写	读/写
LB-9064	启用 USB 条码扫描器装置（键盘功能关闭）（当状态为 ON）	读/写	读/写	读/写
LB-12080	自动连接 PLC(CAN bus) （当状态为 ON）	**写**	**写**	**写**
LB-12081	与 PLC 的通讯状态(CAN bus)设 ON 重连一次	**写**	**写**	**写**
LB-12082	通讯控制符			
LW-9024	（16bit）：Memory link 系统寄存器	读/写	读/写	读/写

LW-9032	（8 words）：备份历史记录到 SD 卡，U 盘 的文件夹名称	读/写	读/写	读/写
LW-9050	（16bit）：目前显示的基本窗口编号	读	读	读
LW-9134	（16bit）：目前所使用的语言（0~7）	读/写	读/写	读/写
LW-9300	（16bit）：连接在本机的 PLC1 所使用的驱动程序编号	读	读	读
LW-9301	（16bit）：连接在本机的 PLC2 所使用的驱动程序编号	读	读	读
LW-9302	（16bit）：连接在本机的 PLC3 所使用的驱动程序编号	读	读	读
LW-9303	（16bit）：连接在本机的 PLC4 所使用的驱动程序编号	读	读	读
LW-9530	（8 words）：VNC 服务器密码	读/写	读/写	读/写

第二十三章 MT8000支持的打印机型号

1. EPSON ESC/P2

点阵式打印机:

LQ–300, LQ–300+, LQ–300K+ (RS232)

LQ–300+II (RS232, USB)

喷墨式打印机:

Stylus Photo 750 (USB)

激光式打印机:

EPL–5800 (USB)

2. HP PCL 系列

USB接口, HP PCL level 3 协议

激光打印机

HP LaserJet P1505n、P1606d: HP PCL 5e

- PCL 5在1990年3月发布于HP Laser Jet III。新增功能: Intellifont 字体缩放(由Compugraphic, 现隶属Agfa研发), outline 字型与HP–GL/29(向量)图形。
- PCL 5e(PCL 5进阶版)在1992年10月发布于HP Laser Jet4。新增功能: 打印机与PC间双向通讯及Windows字型。

注意: 我们**不支持**如下所列的HP打印机;

1.HP LaserJet P1005 (打印机语言非PCL 5)

2.HP LaserJet P1006

3.HP LaserJet 1000 (打印机语言为HostBase, 所以MT8000不支持)

4.HP LaserJet 1010 (打印机语言为HostBase, 所以MT8000不支持)

5.HP Color LaserJet 1500 (打印机语言为HostBase, 所以MT8000不支持)

6.HP Color LaserJet 3500 (打印机语言为HostBase, 所以MT8000不支持)

在连接打印机及MT8000系列前请确认HP打印机是否支持PCL 5，否则可能会造成MT8000无法正常工作。

喷墨打印机:

HP DeskJet 920C，930C，D2368，D2560，D1568，D1668,D2568，D1000

3. SP-M, D, E, F

EPSON ESC协议串口微型打印机

RS232 端口

A、SIUPO

SP–M, D, E, F 系列

SP–E1610SK (paper width: 45mm)

SP–E400–4S (paper width: 57.5mm)

SP–MDEF

在使用打印机之前请先阅读打印机的说明书。

B、炜煌打印机： A72R90–31E72A––16点阵

4. Axiohm A630

法国爱克胜微型打印机，使用串口连接。

5. SPRT (SP-DIII, DIV, D5, D6, A, DN, T)

6. EPSON TM-L90

7. 炜煌打印机：BRIGHTEK WH-E19

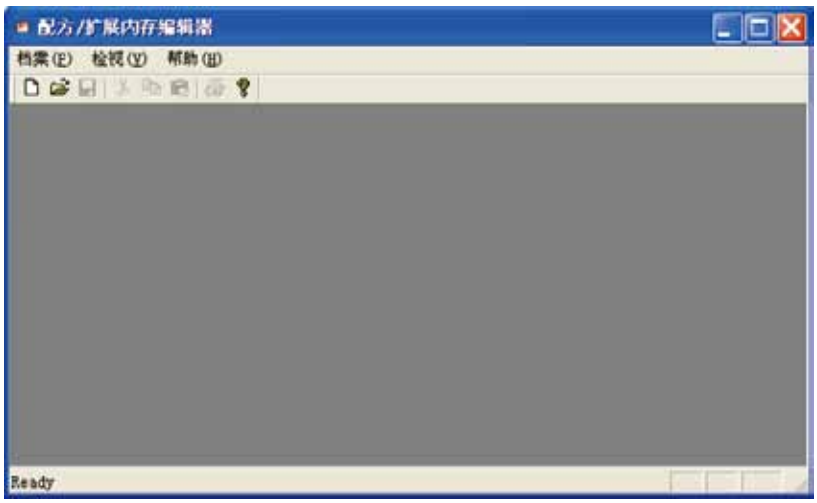
8. 炜煌打印机：BRIGHTEK WH-C1/C2--16点阵

第二十四章 配方编辑器（Recipe Editor）

Recipe Editor是属于Win32的应用程序并且只能在Microsoft Windows XP、Vista和WIN7下执行。此资料编辑软件可以用来新增、浏览、修改rcp、emi资料文件，以及导入和导出CSV文档。

1、介绍

Recipe Editor编辑器界面：



在Recipe Editor编辑器下点击“档案”→“开新档案”，出现设定资料格式的对话框，如下图所示：

设定	叙述
[存取范围]	填入起始地址和结束地址，以字为单位
[新增…]	用户可以在数据类型区域中编辑新的资料格式
[删除]	删除所选择的类型
[删除全部]	删除所有的数据
[修改…]	修改所选择的资料格式设定
[储存格式]	将目前的资料格式储存为范本，之后当用户需要此相同格式时，只需要载入即可。范本将储存为“data.fmt”档案并存放在 EB8000 安装目录资料夹下。
[删除格式]	删除资料格式范本
[选择资料格式]	选择已储存的资料格式范本检视配方或 emi 资料 

点击“新增”，“数据类型”对话框如下图所示：

用户可以在“描述”处输入资料格式的名称，并选择所需要的数据类型，如果选择“string”，需输入字串的长度。

2、Recipe Editor设定

如何增加Recipe/EMI文件

(1) 在Recipe Editor编辑器下点击“档案”→“开新档案”，将出现以下画面：

(2) 设定存储范围并点击“增加”，选择“16 bit Unsigned”做为资料格式类型。

(3) 触控确定后，新的资料格式将显示如下：

(4) 用户可以在此修改或检视数据

(5) 在“另存新档”选择所要存储的文件格式。

输出“CSV档案”

在打开recipe或EMI文件后，选择“另存新档”并选择储存的格式为CSV。

导入“CSV档案”

在Recipe Editor编辑器下点击“档案”→“导入CSV档案”，选择所要打开的CSV格式文档，在编辑完成后，可将它储存成recipe 或EMI文件并下载到触摸屏。

第二十五章 EasyConverter

此工具用来将MT8000所存储的资料取样(dtl)或事件登录(evt)的历史记录转换为可阅读的文件(csv)；若用户电脑已经安装Microsoft Excel，也可直接将其转换结果输出到Excel程序中。

1、介绍

在Project Manager中，点击“EasyConverter”，会弹出相应的应用程序。

下面将介绍四个功能

1、输出至Excel

2、Scaling功能

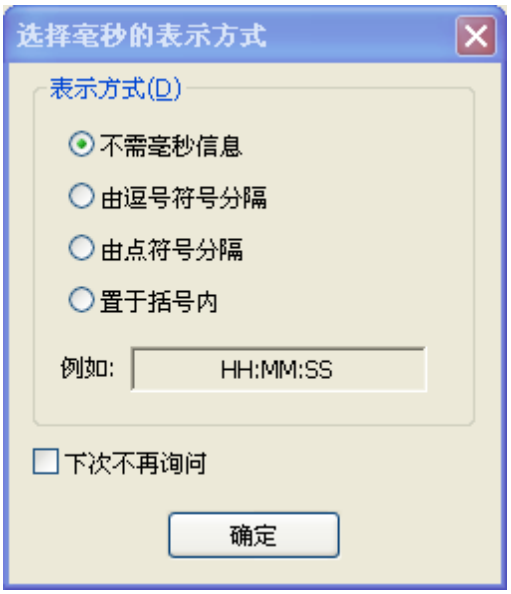
3、多档案转换

4、命令模式

2、EasyConverter 设定

● 如何输出至Excel

当开启文件，会弹出设定窗口如下：



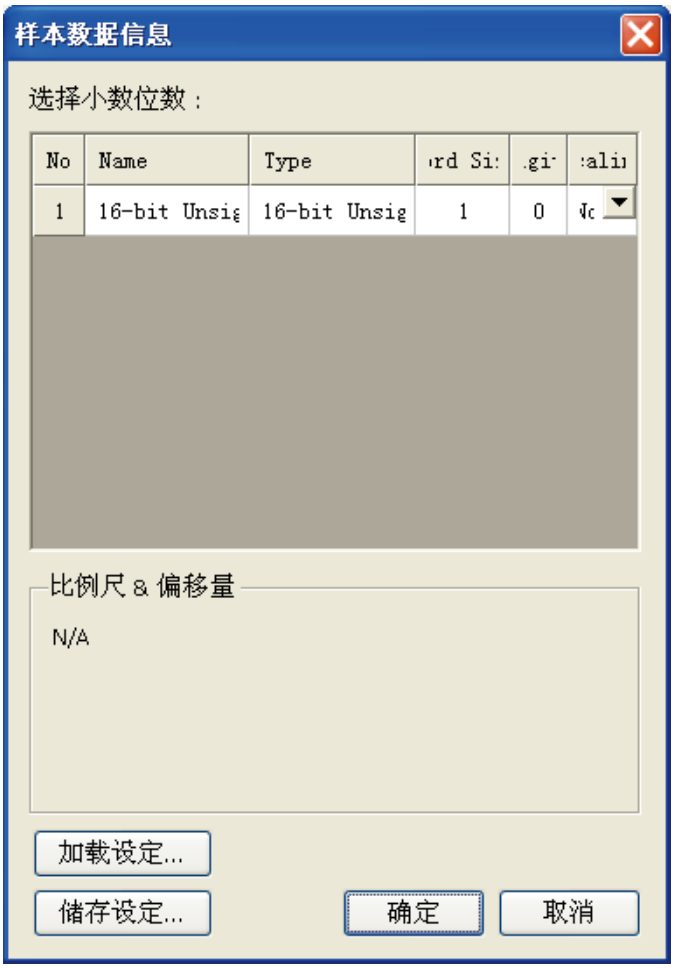
有四个毫秒的表示方法可供用户选择

不需毫秒信息	例如: HH:MM:SS
由逗号符号分隔	例如: HH:MM:SS,###
由点符号分隔	例如: HH:MM:SS.###
置于括号内	例如: HH:MM:SS(###)

如果勾选“下次不再询问”，则下次开启“EasyConverter”不会弹出此设定窗口，而会沿上次的设定。

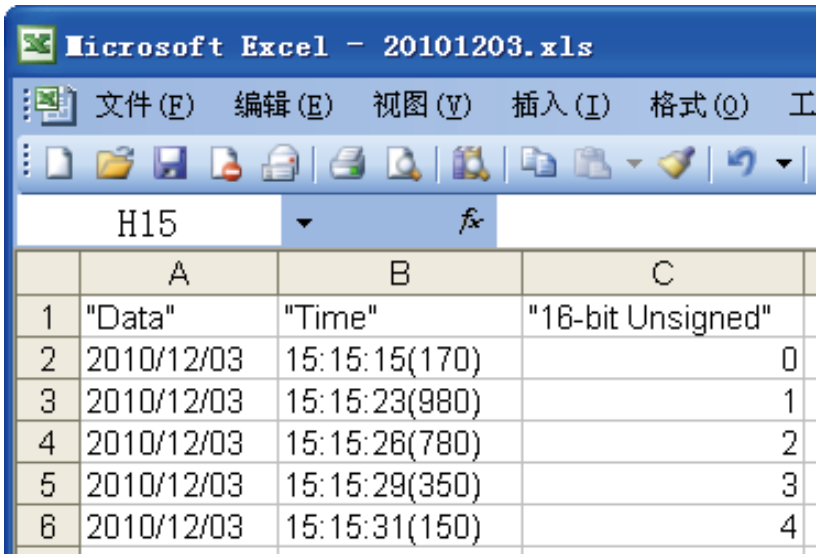
如果需要修改“选择毫秒的表示方式”，可由“选项/时间格式”开启此设定窗口。

设定好后触控“确定”按钮，会弹出下一设定窗口如下：



触控“确定”后

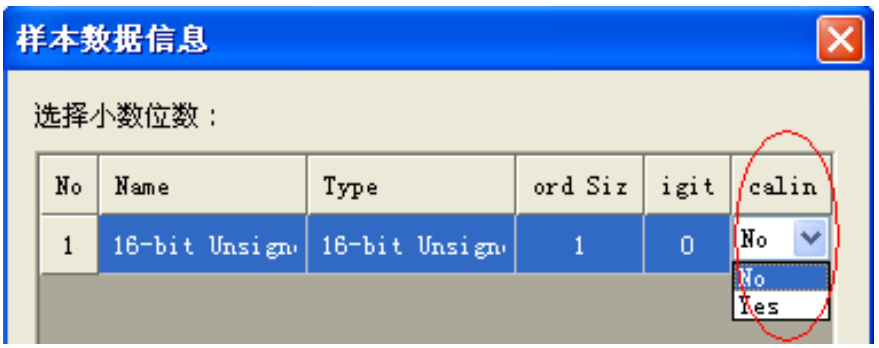
再触控“导出至Excel”即可。



	A	B	C
1	"Data"	"Time"	"16-bit Unsigned"
2	2010/12/03	15:15:15(170)	0
3	2010/12/03	15:15:23(980)	1
4	2010/12/03	15:15:26(780)	2
5	2010/12/03	15:15:29(350)	3
6	2010/12/03	15:15:31(150)	4

- 如何使用Scaling功能

Scaling功能使用方式如下:

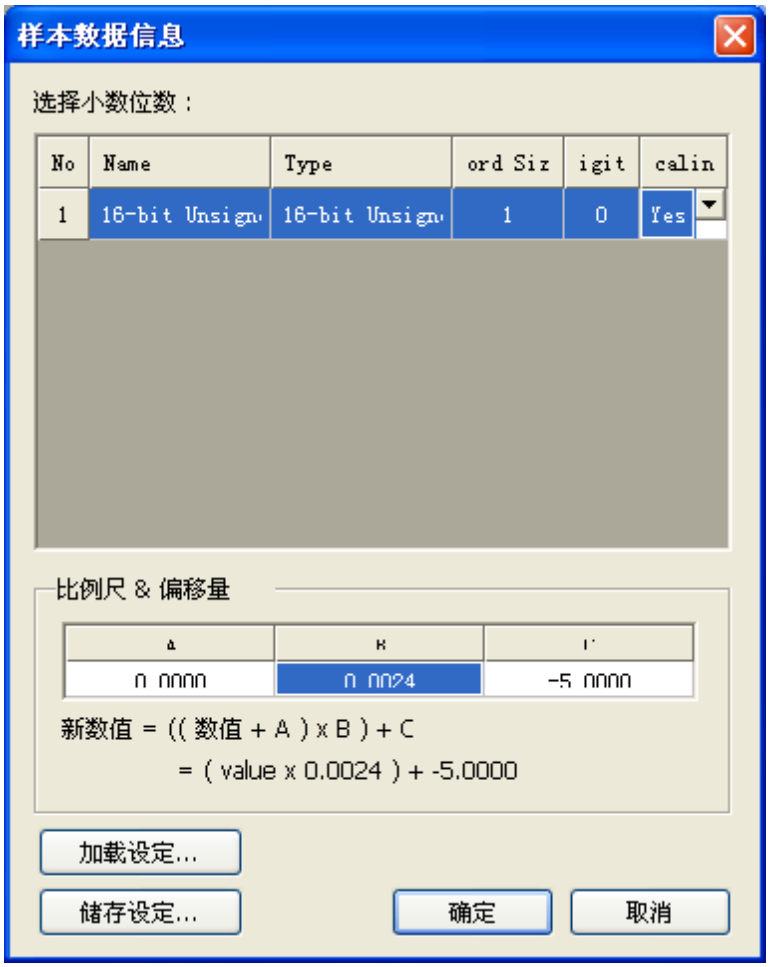


新数值={{数值+A}+C, 用户可以在A、B、和C设定数值。

为什么需要Scaling呢?

以下举个例子做说明, 有一个电压资料, 其格式时16-bit unsigned, 电压值是介于0至4096, 用户要将其电压值转换成伏特, 介于-5V至+5V之间。

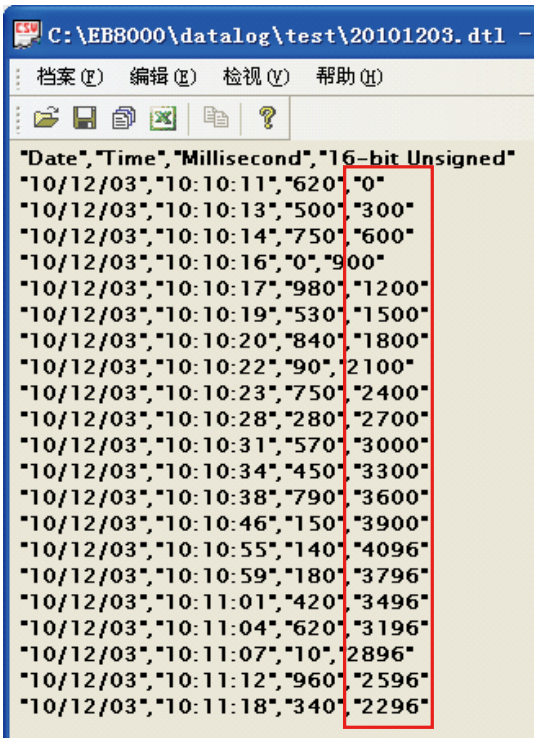
例如: 新数值={{数值+0}x0.0024}+(-5):



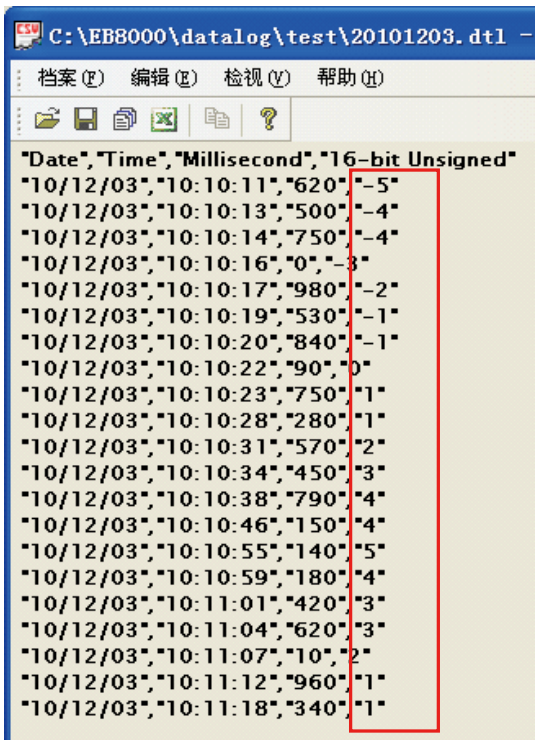
以上的资料设定，可以储存成样本，并于下次使用时可以直接载入设定。

设定完成之后

原始文件



使用scaling之后的文件



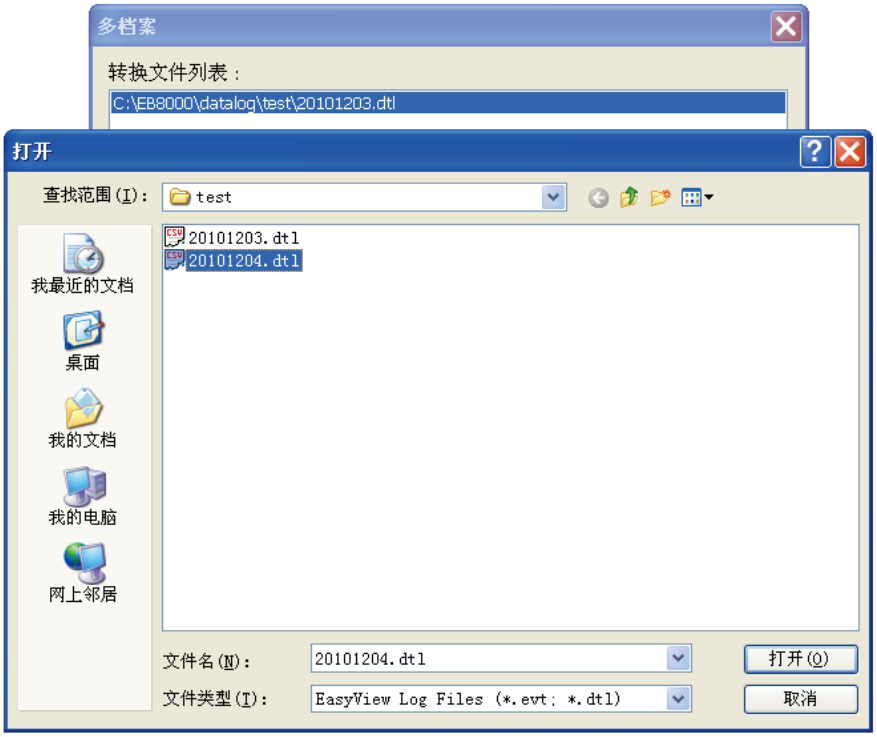
- 如何使用多档案转换

步骤1

点击“档案/多档案”，会弹出设定窗口

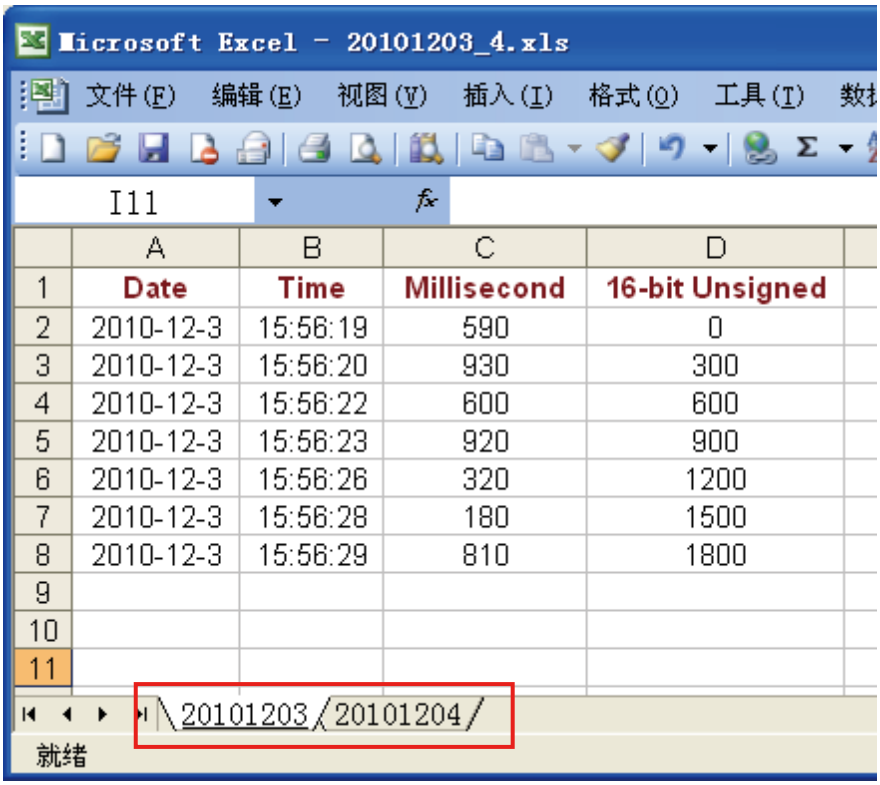
步骤2

用户可以点击“增加档案…”加入多个文件合并为一个Excel。

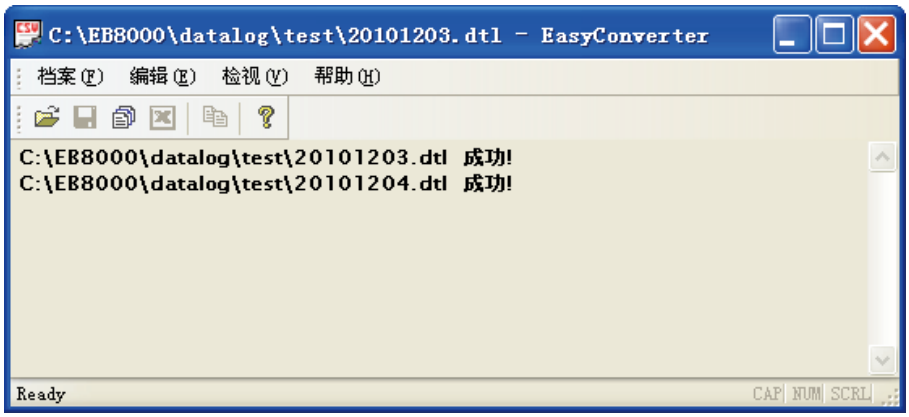


步骤3

增加完文件之后，点击“将转换结果组合为单一档案”，文件将输出为Excel（xls）。



注意: 如不点击此选项而触控确定, 文件将会分别被输出至Excel

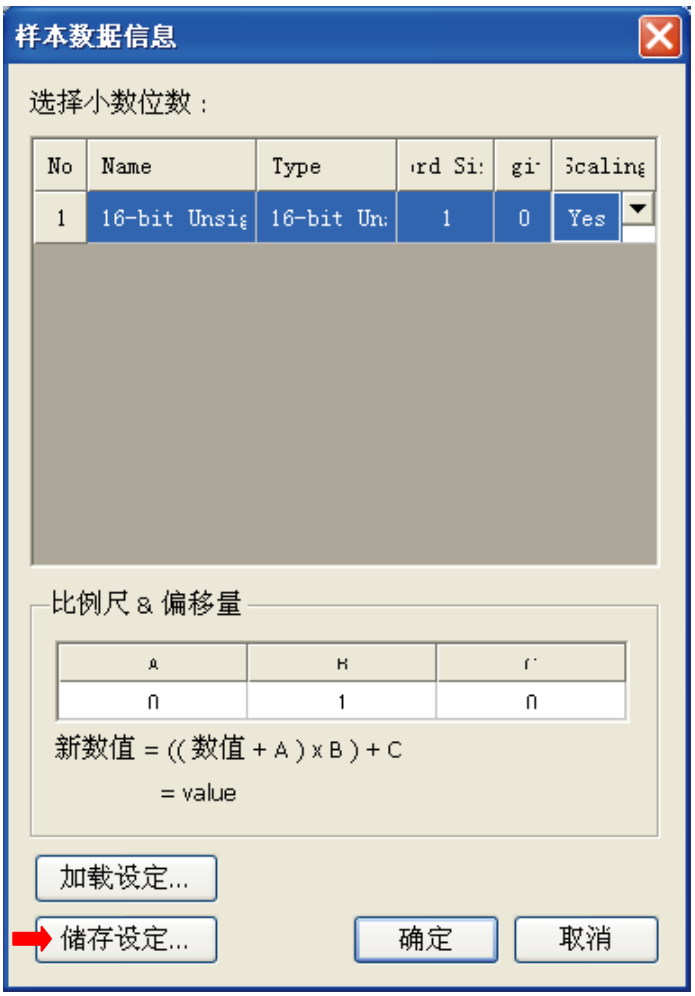


3、启用设定档案

用户也可以载入已储存的设定文件至合并档。

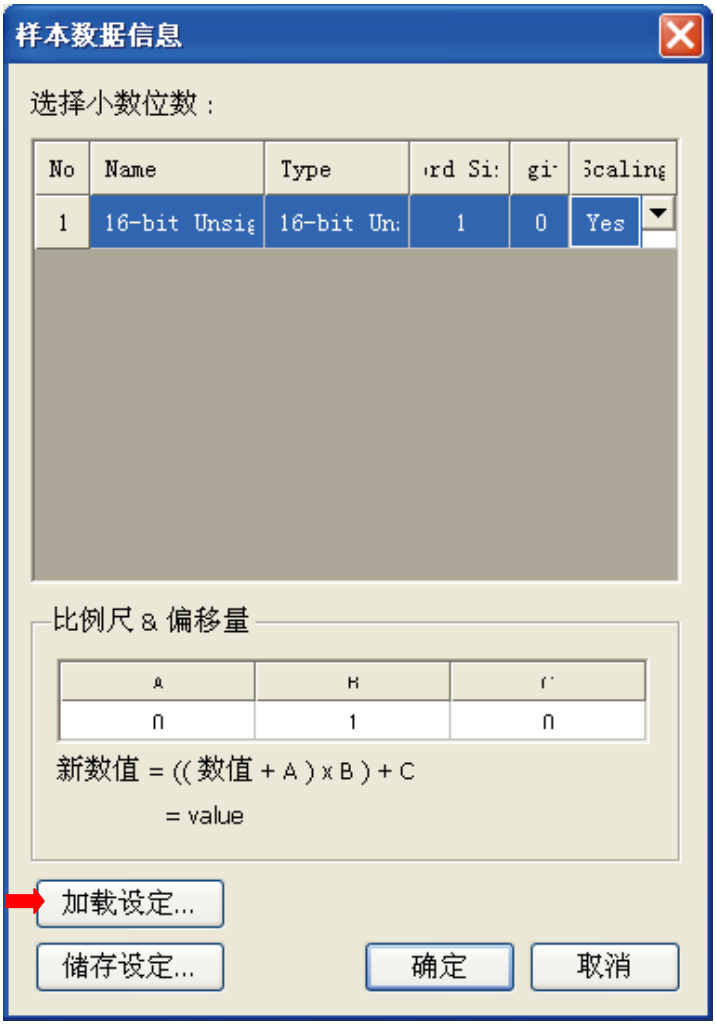
步骤1:

在设定完Scaling各项数据之后, 储存设定档。



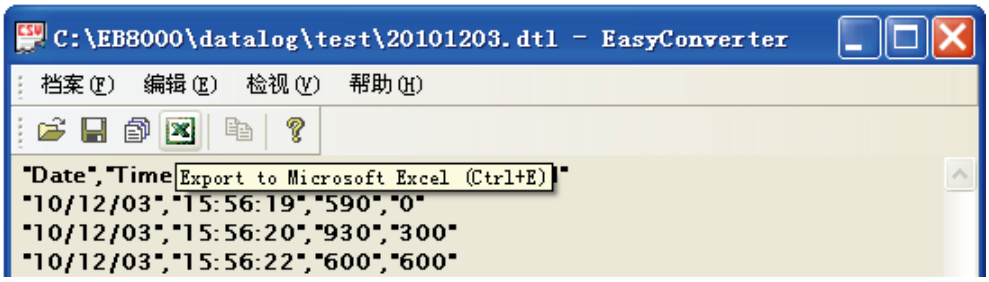
步骤2

在新的样本数据信息对话框中，点击“加载设定”



步骤3

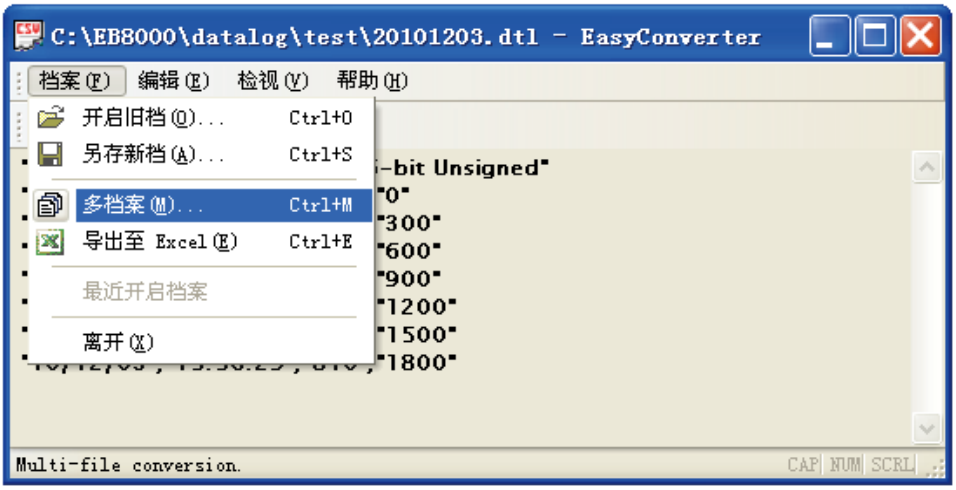
点击：**[Export to Microsoft Excel]**之后就可以查看资料：



- 合并并使用设定档案

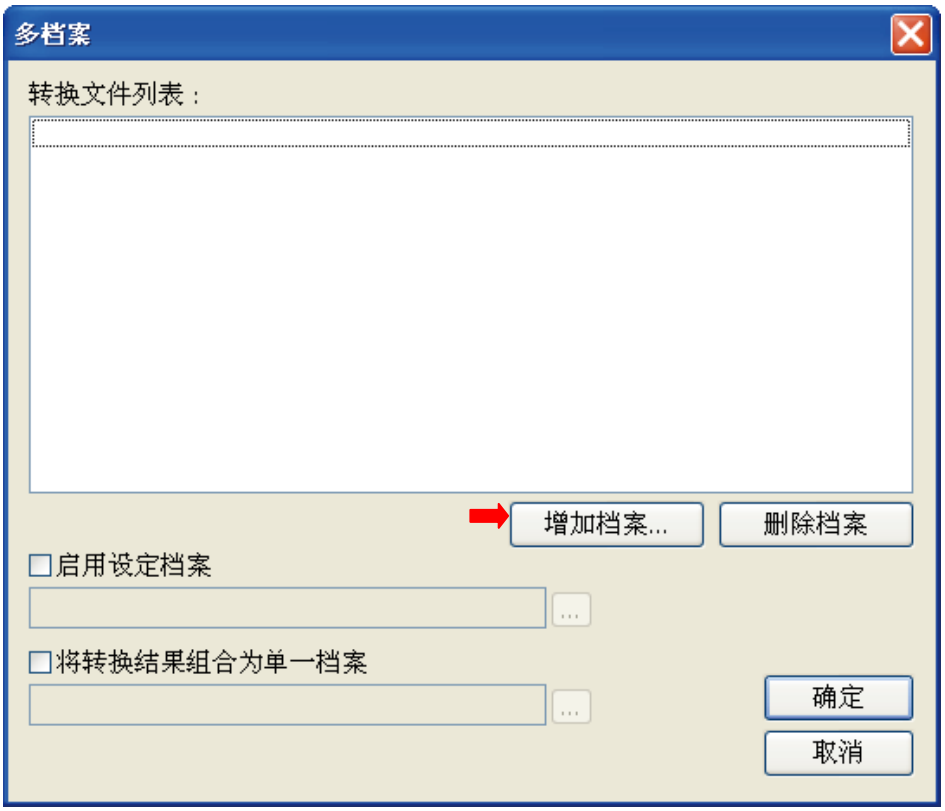
步骤1

点击“多档案”



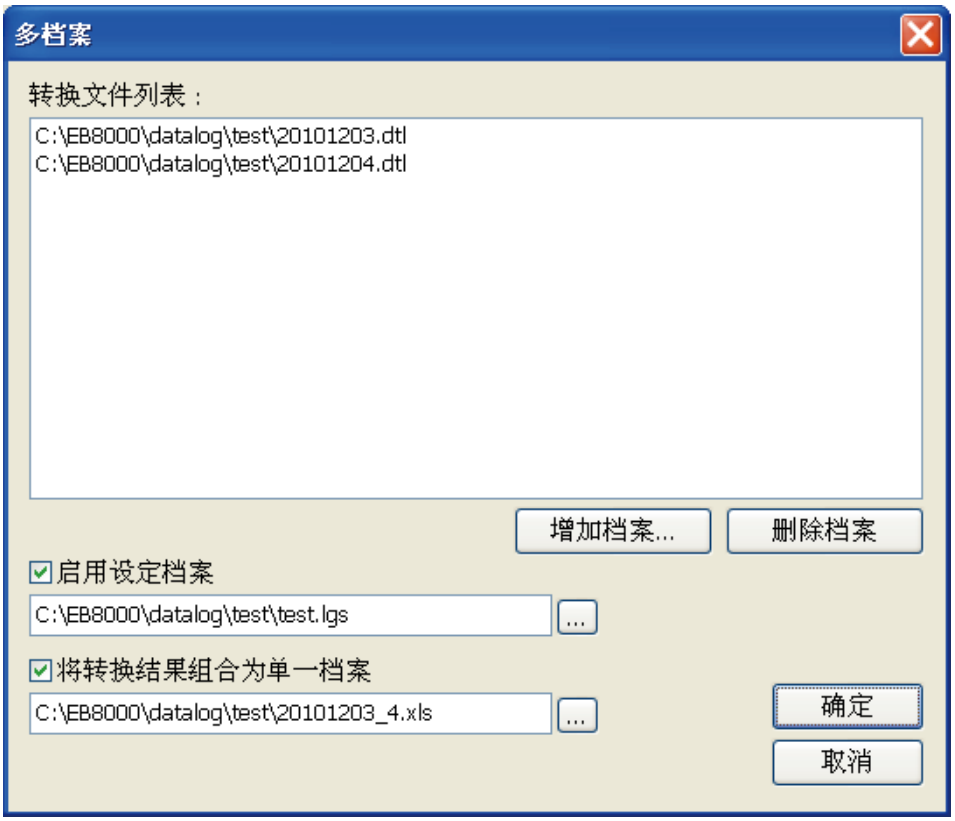
步骤2

点击“增加档案”



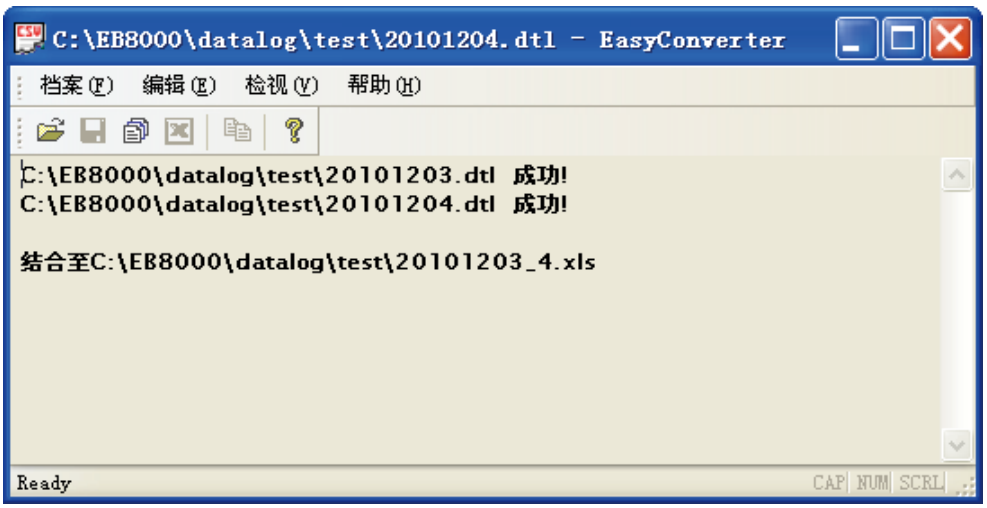
步骤3

选择所要合并的文件，点击“启用设定档案”和“将转换结果组合为单一档案”并选择所要合并的文件名称。



步骤4

点击“确定”，信息将会显示在对话框中：



步骤5

此时可以去打开刚刚所合并的Excel查看资料。

4、命令模式

用户可以在命令模式下执行EasyConverter。

EasyConverter [/c] [/s] [/t[num]] 设定来源目标

设定	指定设定档 (*.lgs)
来源	指定来源档 (*.dtl 或 *.evt)
目标	指定目标档 (*.csv 或 *.xls)
/c	输出档案类型，输入此符号，档案将被输出为 CSV，反之为 EXCEL 格式档案
/s	输入此符号，系统将使用指定的设定档
/t[num]	时间格式 例如：t2 指定为“有点符号分隔”

例如：EasyConverter.exe /c /s /t3 “C:\EB8000\datalog\test\test.lgs”

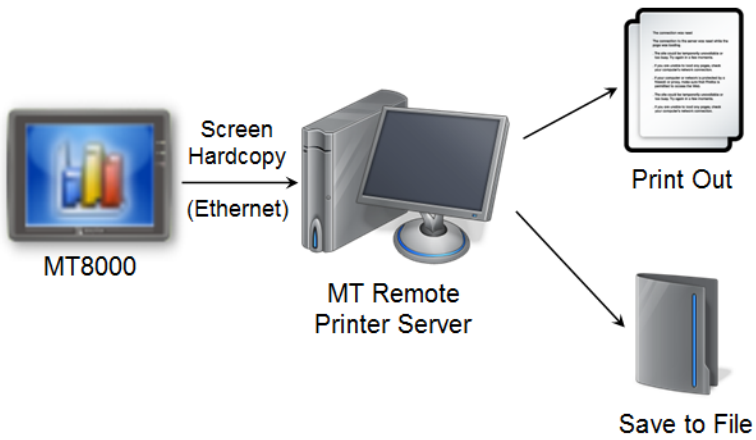
“C:\EB8000\datalog\test\20101203.dtl”“ C:\EB8000\datalog\test\”

第二十六章 EasyPrinter

EasyPrinter是属于Win32的应用程序，因此只能在Microsoft Windows XP、Vista和WIN7 下执行。此功能是MT8000系列触摸屏，可以通过以太网，在运行了EasyPrinter客户端软件的远端电脑上，进行屏幕输出打印或数据备份，详见以下说明。

EasyPrinter功能，是一个打印服务器或备份服务器。

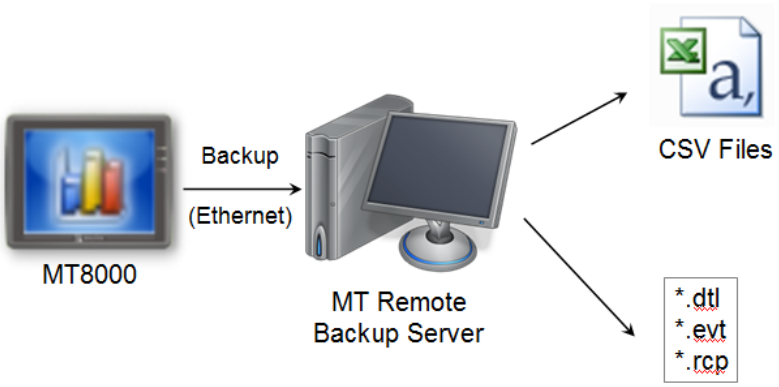
它可以运行在Microsoft Windows XP、Vista和WIN7下执行等32位操作系统下。



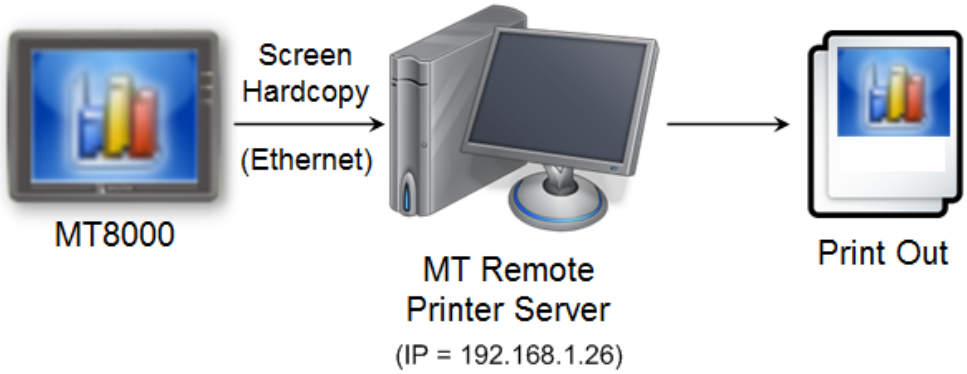
以下为使用EasyPrinter的优点：

- EasyPrinter提供两种屏幕打印输出模式：输出至远端打印机、保存至文件夹。用户可以使用其中一种或两种；
- 由于EasyPrinter在Windows系统下运行，因此支持市面上大部分的打印机；
- 多台MT8000触摸屏可以共享一台打印机，用户不需为每台触摸屏准备一台打印机。

另外，EasyPrinter可以是一台备份服务器，用户可使用触摸屏上的备份文件，通过以太网，将资料取样与事件记录等历史备份至远端PC，详见下面说明：



26.1 使用EasyPrinter为打印服务器



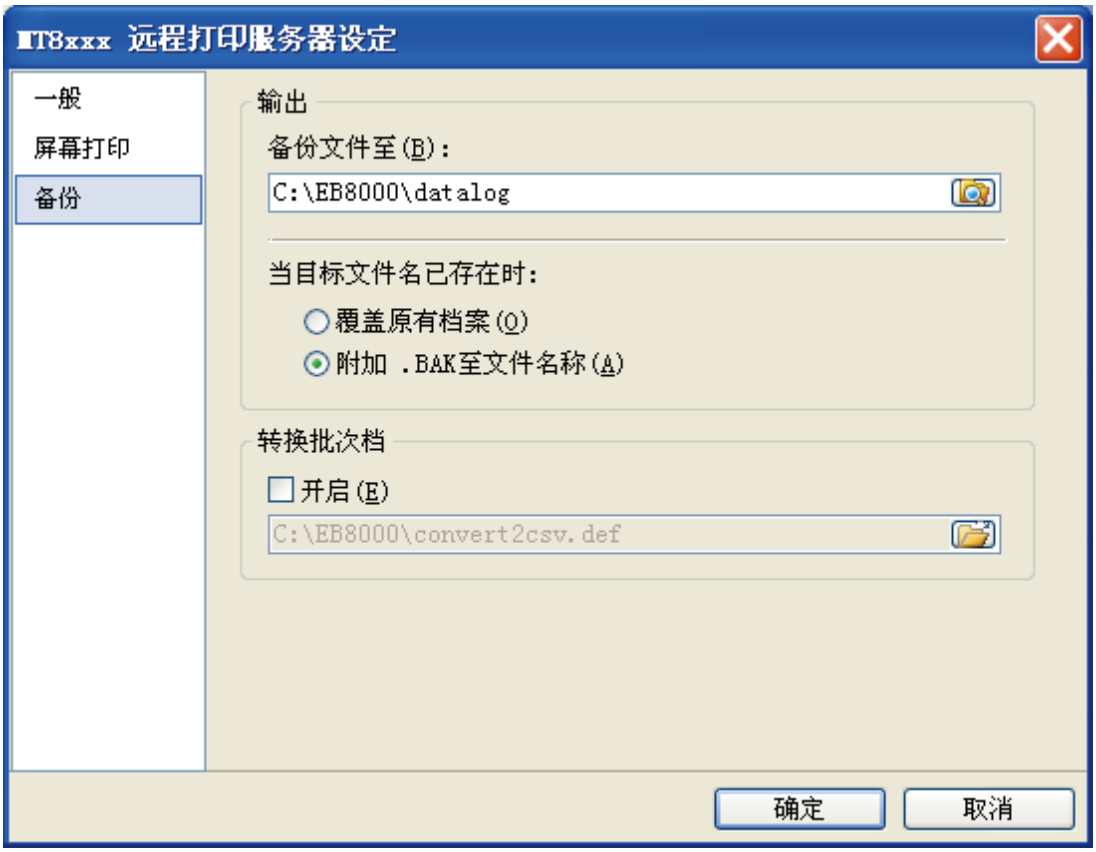
用户可以使用“功能键”元件来操作屏幕打印，这些屏幕打印会透过以太网被传送至MT远端打印服务器，然后被打印出来。

- EasyPrinter设定程序

在EasyPrinter设定页下点击“选项”/“设定”即会出现下面的对话框：

1. 在[伺服], 指定[连接端口]为 “8005”, [使用者名称]为 “admin”, [密码]为 “111111”（以上皆为预设值）。
2. 在[目录名称命名方式], 点选[使用IP地址]并指定 “IP_” 为[文件名称]
3. 在[属性], 选择[缩小至工具条]

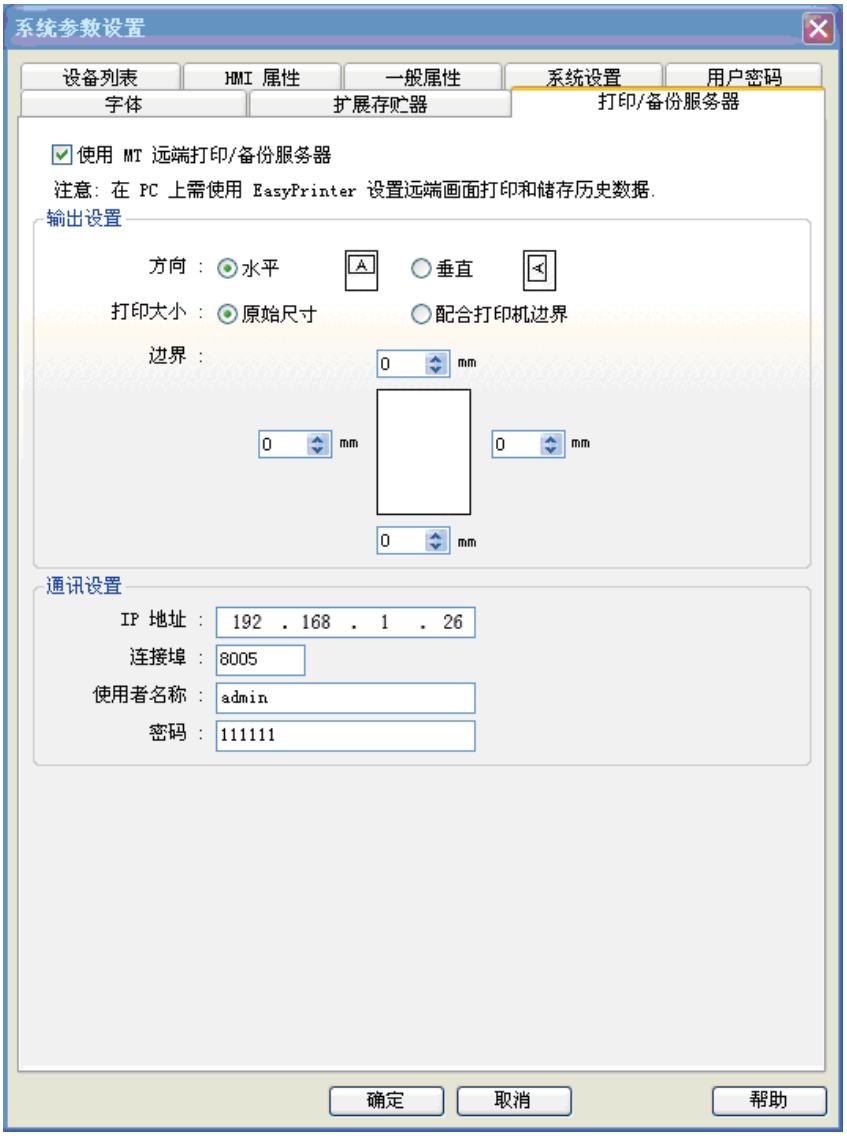
选择左列 “屏幕打印” 如下:



4. 在[输出], 点选 “输出至” 并选择一台打印机作为屏幕打印的输出设备（注意: 用户只能选择自己的系统中存在的打印机, 因此上列打印机仅为参考）
5. 触控确认键使用以上设定
6. 在EasyPrinter设定页下点击 “档案” / “允许输出”, EasyPrinter会将这些打印指令输出, 也就是打印屏幕。

- **EasyBuilder8000设定程序**

  在 “EB8000” / “编辑” / “系统参数设定”, 点击 “打印/备份服务器”, 并勾选 “使用MT远端打印/备份服务器”, 会出现以下窗口:



7. 在[输出设定], 指定适当的边界, (在此例中上下左右边界皆设定为0)

8. 在[通讯设置], 输入打印服务器[IP地址], 如192.168.1.26, 指定[连接端口]号为“8005”, [使用者名称]为“admin”, [密码]为“111111”

在画面上放置一个“功能键”元件, 并在其设定页中选择“画面打印”, 打印机: 选择“MT远程打印/备份服务器”。

9. 将[功能键]元件放置于公共窗口, 用户便可以随时开始屏幕打印。

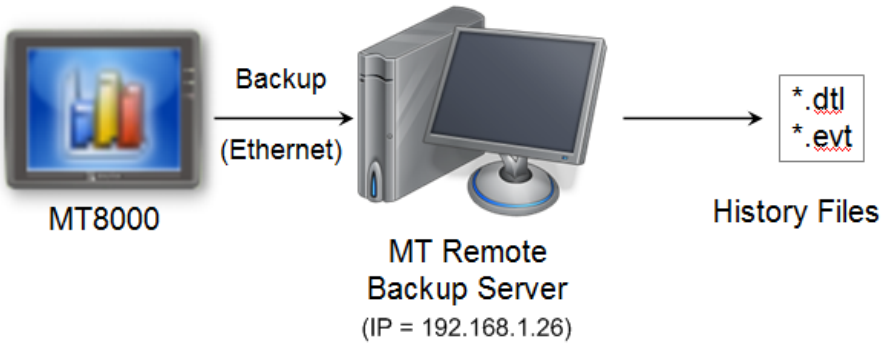
10.[编译]及[下载]工程文件至MT8000触摸屏, 触控前面设定的“功能键”, 开始打印。

NOTE

- 用户亦可以通过[PLC 控制]元件来达成屏幕打印
- 警报资料无法通过 EasyPrinter 打印

- EasyPrinter 只能通过以太网与触摸屏通讯，因此这个功能不适用于 MT6000 系列触摸屏

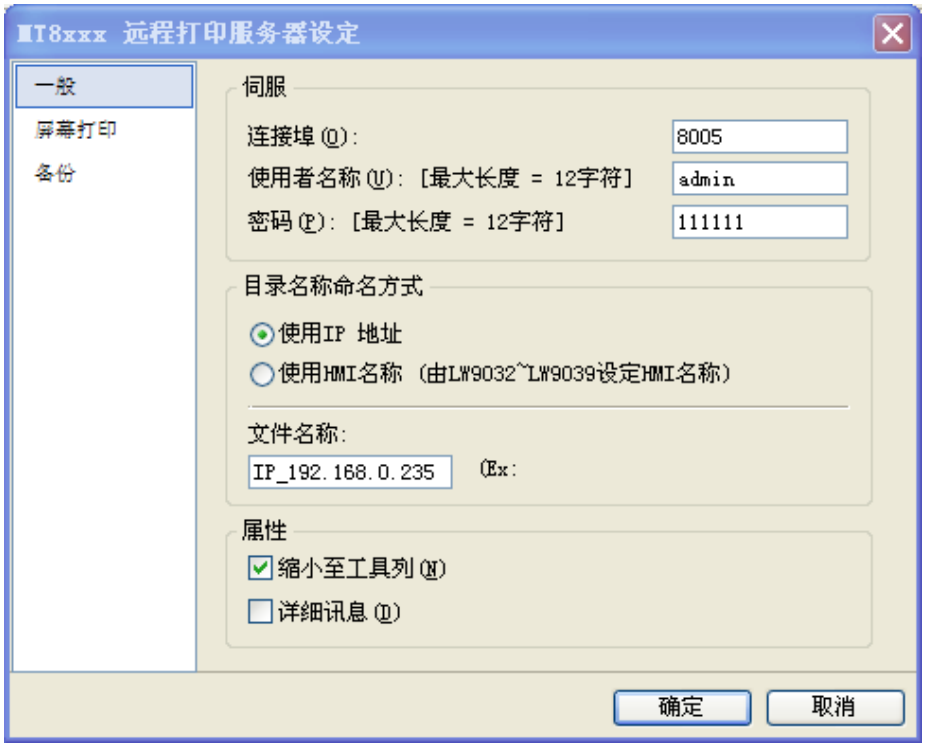
26.2 使用EasyPrinter为备份服务器



用户在使用“备份”元件，将历史资料，例如资料取样与事件记录等上传至MT远端备份服务器。

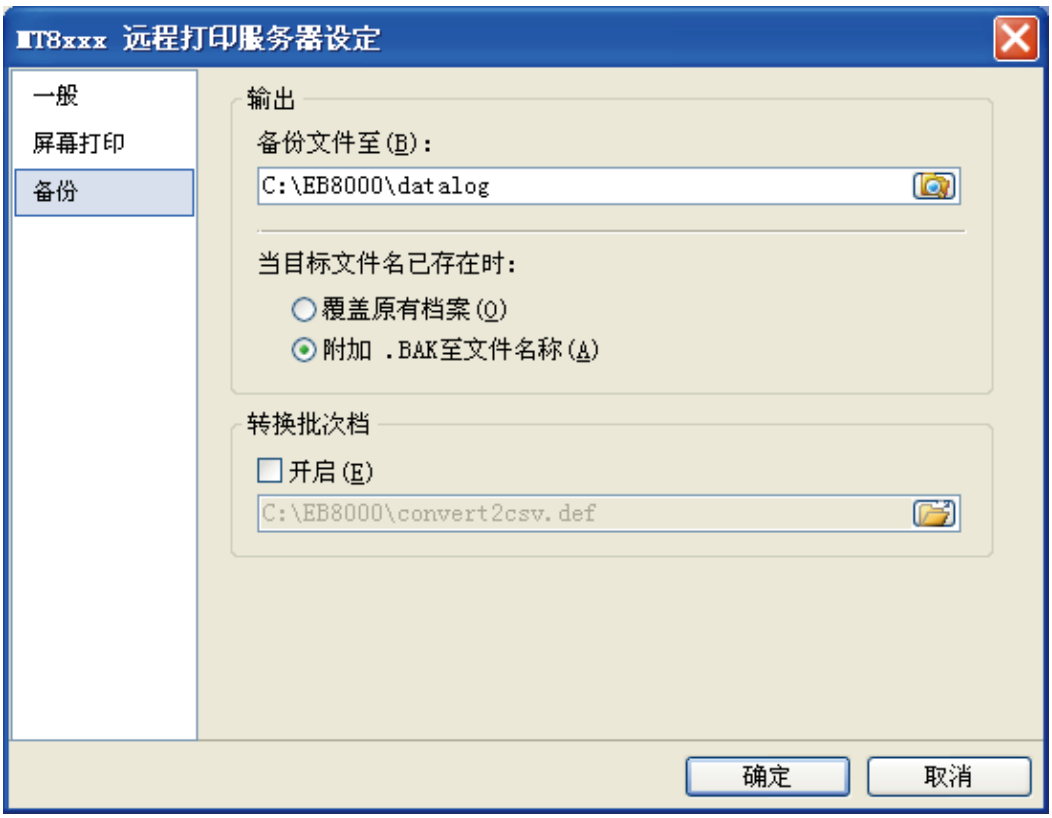
- EasyPrinter备份设定程序

在EasyPrinter设定页下点击“选项”/“设定”，即会出现下面的对话框：



1. 在[伺服], 指定[连接端口]为 “8005” , [使用者名称]为 “admin” , [密码]为 “111111” 。(以上皆为预设值)
2. 在[目录名称命名方式], 选择[使用IP地址]并指定 “IP_” 为[文件名称]。
3. 在[属性], 选择[缩小至工具条]

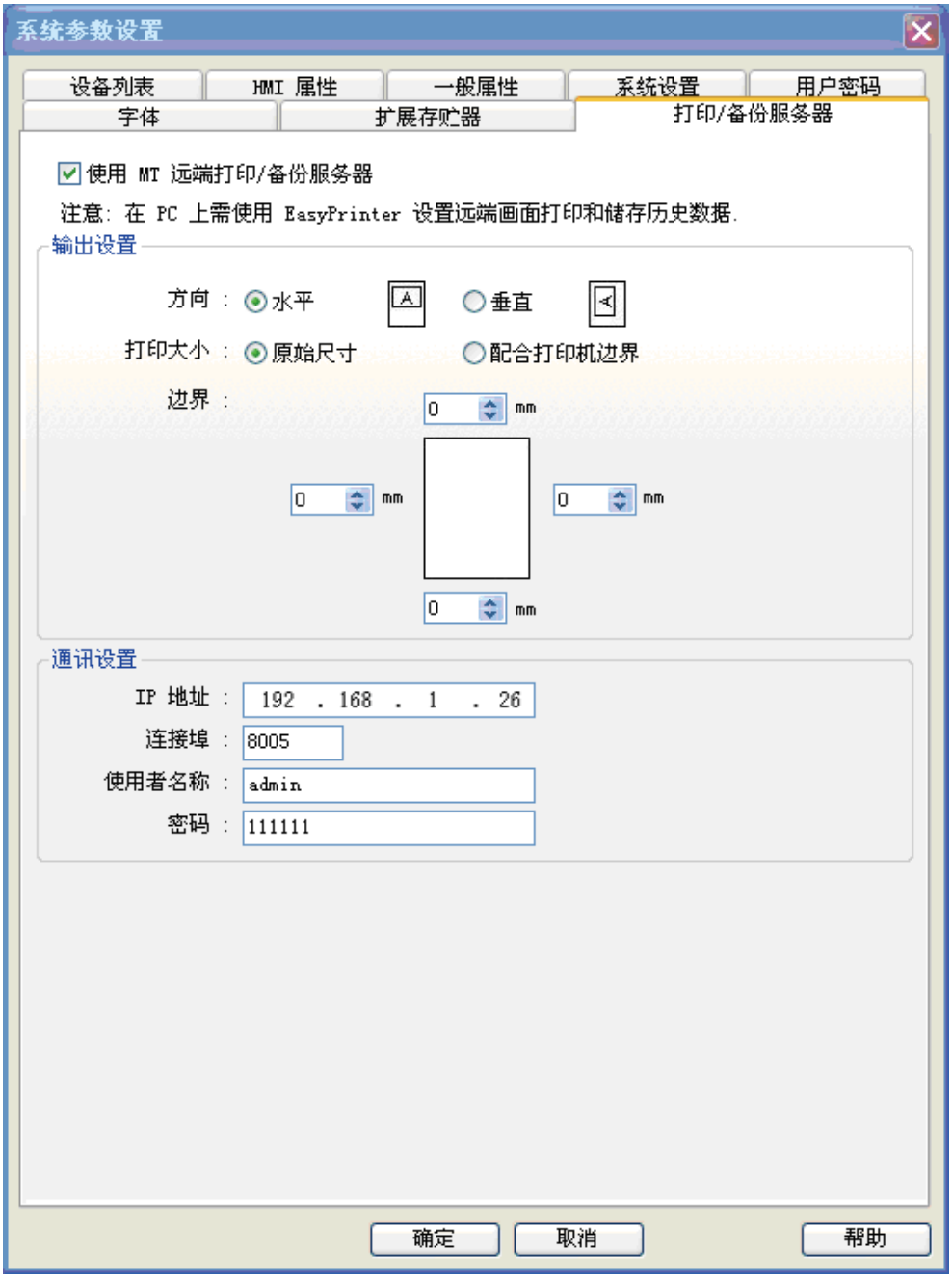
点击左列[备份]如下图:



4. 在[输出]点击 来浏览及选择历史资料的存储路径。
5. 触控确认按钮使用以上设定
6. 在EasyPrinter设定页下点击[档案]/[允许输出],EasyPrinter会将备份资料储存在方才所选的路径。

● EasyBuilder8000备份设定程序

在[EB8000]/[编辑]/[系统参数设定], 点击[打印/备份服务器], 并勾选[使用MT远端打印/备份服务器], 会出现以下窗口:



7. 在[通讯设置], 输入打印服务器[IP地址], 如192.168.1.26, 指定[连接端口]号为“8005”, [使用者名称]为“admin”, [密码]为“111111”

点击[EB8000]/[元件]/[备份]会出现以下对话框:

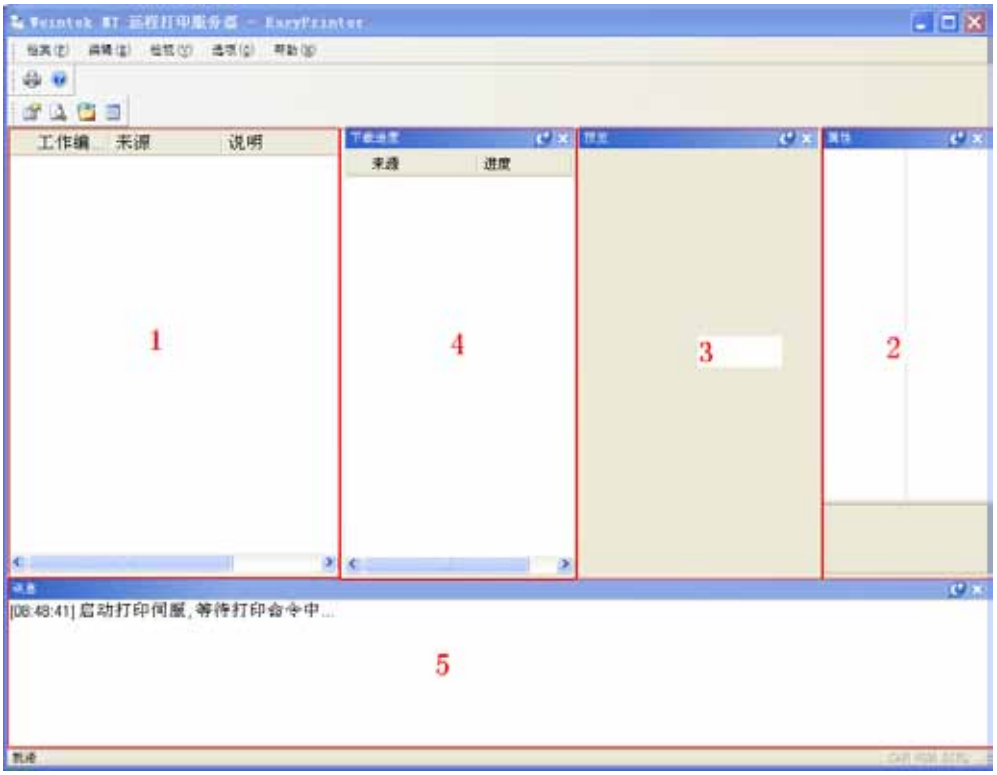
8. 在[来源], 选择[事件记录]
9. 在[备份位置], 选择[远端打印/备份服务器]
10. 在[范围], 选择[今天]和[全部]
11. 在[属性], 选择[手动]
12. 将[备份]元件放置到公共窗口, 用户便可以随时开始资料备份
13. [编译]并[下载]工程文件至MT8000触摸屏, 触控前面设定的[备份]元件, 开始备份历史资料

NOTE

- [备份]可用位地址来触发。
- 用户可以放置一个[排程]元件，在一周后的最后一天转为 ON，用以触发[备份]元件自动备份所有历史资料。

26.3 EasyPrinter操作说明

- 窗口界面



区域	名称	说明
1	工具列表	此窗口显示所有工作，包括屏幕打印与备份工作
2	属性窗口	此窗口显示工具列表中所选工作的信息
3	预览窗口	此窗口显示工具列表中所选屏幕打印的预览图片
4	下载进度窗口	此窗口显示新近工作的下载进度
5	信息窗口	此窗口显示工作执行中的时间与信息，例如密码错误等

- 操作说明

以下表格说明EasyPrinter选单下各选项使用方式所代表的意义：

[档案]	说明
[允许输出]	• 选择：**EasyPrinter** 执行工作 • 不选择：**EasyPrinter** 将工作保留在内存中

NOTE

- EasyPrinter 只能将最多 128M 的工作资料保存在内存中，若内存已满，所有的新近工作都会被拒绝，用户可以选择**[允许输出]**直接执行，或是删除部分工作来清除内存空间给新近工作。

[编辑]	说明
[编辑]	编辑一个[屏幕打印]工作。  用户可以在此自由设定**[方向][打印大小][边界]**。
[删除]	永久删除所选工作
[选择全部]	选择[工具列表]上的所有工作。

NOTE

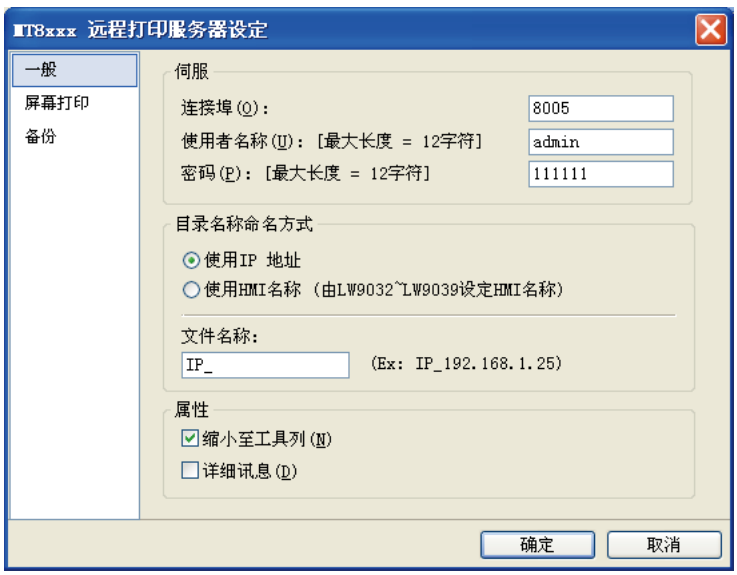
- 备份工作不可编辑
- 只有在选择工作后才能**[编辑]**。
- 选择至少一样工作方可**[删除]**。

[检视]	说明
属性视窗	显示或隐藏属性窗口
预览窗口	显示或隐藏预览窗口
下载窗口	显示或隐藏下载进度窗口
讯息窗口	显示或隐藏讯息窗口

NOTE

- 在[下载进度]窗口中，用户可以选择下载进度的显示方式，点击[进度]如下：

- EasyPrinter 可以保留 10,000 笔[讯息]窗口中的讯息，当新讯息产生时，最旧的讯息就会被删除。

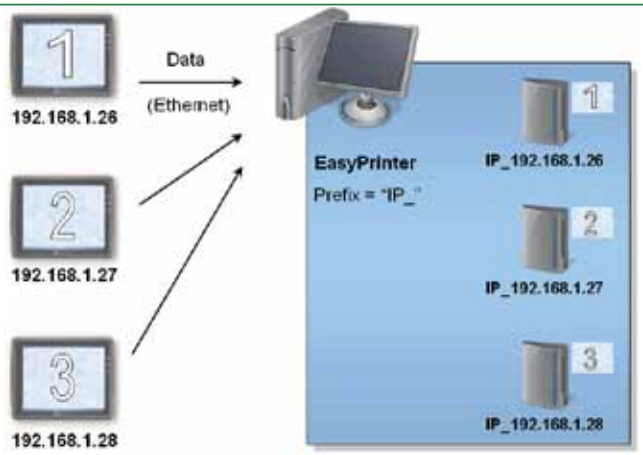
[选项]	说明
[工具列]	显示或隐藏工具列
[状态列]	显示或隐藏状态列
[设定]	EasyPrinter操作设定，请见下图： [一般]  ● [伺服]→[连接端口] 设定以太网接口让触摸屏连接，范围1~65535,8005是预设值。 ● [伺服]→[使用者名称]及[密码] 设定使用者名称和密码，让有授权的触摸屏能使用EasyPrinter功能。

●[目录名称命名方式]

EasyPrinter使用不同的资料夹来储存来自不同触摸屏的各种档案(屏幕打印点阵图档, 备份档等等)。有两种方式可以命名这些资料夹:

a. [使用IP地址]

在使用所设定IP地址的触摸屏发出指令后, EasyPrinter会为资料夹命名: [文件名称]+[IP地址]。详见下图:



b. [使用HMI名称]

EasyPrinter 使用送出指令的 HMI的名称来为资料夹命名:

[文件名称]+[HMI名称]。

● [属性] → [缩小至工具列]

勾选此项, EasyPrinter会被缩小并置于状态栏, 用户只要双击状态栏的图像, 即可打开EasyPrinter。

● [属性] → [详细讯息]

勾选此项, 讯息窗口中会显示更详细的事件讯息。

[屏幕打印]

● [输出]

EasyPrinter 提供两种屏幕打印结果输出模式：

a. 输出至（打印机）

选择此项告知EasyPrinter将屏幕打印结果通过指定打印机打印出来。

b. 储存至（档案）

选择此项告知EasyPrinter将屏幕的打印结果转换成图片，并储存于指定路径，用户可以在以下路径找到图片。

[用户指定路径] →

[HMI资料夹] →

yymmdd_hhmm.bmp

举例来说，当一个屏幕打印指令发生于17:35:00 2009年1月12日，图片档案将被命名为“090112_1735.bmp”。如果同一分钟有另一个图片档产生，新图片将被命名为“090112_1735_01.bmp”，以此类推。

[备份]

● [输出]

EasyPrinter将备份档案储存于特定的路径下。

事件记录历史资料路径：

[使用者指定路径] →

[HMI资料夹] →

[事件记录] →

EL_yyyymmdd.evt

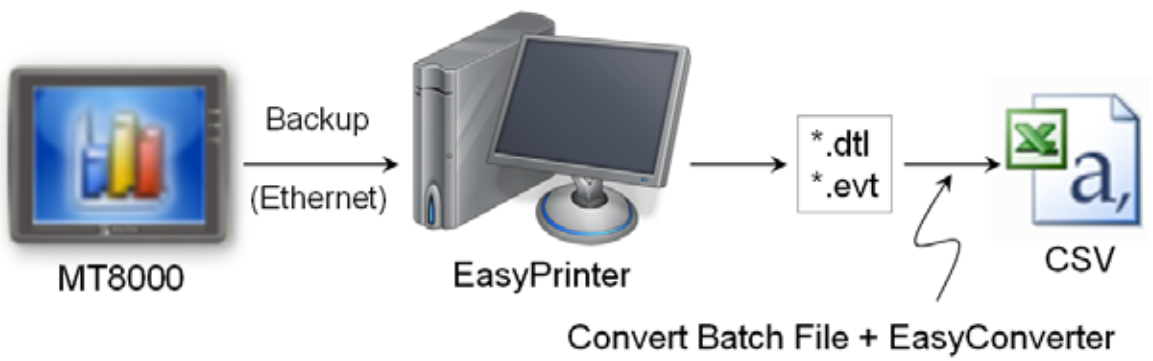
	资料取样历史资料路径： [使用者指定路径] → [HMI资料夹] → [资料取样记录] → [资料取样元件的档案名称] → yyyymmdd.dtl 配方数据备份路径： [使用者指定路径] → [HMI资料夹] → [配方数据] → recipe.rcp 或recipe_a.rcp ● **[转换批次档]** 勾选[开启]指定自动将上传的历史资料档案转换成CSV或EXCEL档的转换批次档案。详情请见下面说明。

NOTE

- 用户可以使用系统寄存器 LW9032 to LW9039 来指定 HMI 名称。
- 若尚未设定 HMI 名称，EasyPrinter 会使用 IP 地址来命名档案文件夹。

26.4 转换批次档

EasyPrinter提供一个转换档功能，可将上传的资料取样与事件记录等两种历史文件自动保存为CSV文档，用户若需要此功能，需先转变一个转换批次档，告知EasyPrinter如何转换历史文件。



如上图所示，此转换功能实际时由EasyConverter执行，EasyPrinter至少遵照转换批次档的标准用正确的参数启动EasyConverter达成转档指示。

NOTE

- EasyConverter 是另一个 Win32 应用程式，可将历史资料转换成 CSV 或 MS Excel(*.xls) 等档案，用户可在 EasyBuilder8000 下载路径中找到这个程式。
- 用户若使用此功能，须先确定 EasyPrinter 及 EasyConverter 皆被放置于相同路径下。

● 转换批次档预设值

以下为EasyBuilder8000软件所包含的预设转换批次档：

预设转换批次档（convert2csv.def）

1: “dtl”, “EasyConverter /c $(路径名称)”

2: “evt”, “EasyConverter /c $(路径名称)”

文件文字会以两行呈现，每行含有两个参数，用逗号隔开，形成对应特定类型文件（资料取样和事件记录）的处理标准。第一个参数显示该文件类型的扩展名，第二个参数显示操作模式所需执行的命令。“$(路径名称)”是关键字，告诉EasyPrinter用需转档的备份文件名称来取代它。例如：资料取样文件名称为20090112.dtl，已被上传并储存，EasyPrinter会输出以下指令到命令窗口：

```
EasyConverter /c 20090112.dtl
```

如此一个名称为20090112.csv的文件即被建立。

因此，转换批次档的预设标准如下：

1. 转换所有资料取样历史记录（*.dtl）为CSV文件。
2. 转换所有事件记录历史记录（*.evt）为CSV文件。

NOTE

- 事实上，第二个参数中的“$(（路径名称）”代表档案的完整路径名称，在前面的例子中，EasyPrinter 以下面名称取代：

[用户指定路径] \ [HMI 资料夹] \ [事件记录] \
[资料取样元件工程名称] \ 20090112.dtl

- EasyPrinter 以一行档案文字为单位来解读转换批次档，也就是说，一行文字形成一个标准。
- 任两个参数皆需以逗号隔开。
- 任一参数皆需以双引号标示。
- 同一参数中切勿放逗号。
- 详细介绍请见《第二十五章 EasyConverter》。

● 特定标准

有时用户可能需要针对特定的触摸屏上传的资料使用特别的操作，如下例：

针对HMI IP = 192.168.1.26 的特别定义标准

3:“dtl”,“EasyConverter /c $(路径名称)”,“192.168.1.26”

用户也可以用触摸屏的名称来辨别该触摸屏。

针对HMI 名称= Weinview_01的特别定义标准

4:“dtl”,“EasyConverter /c $(路径名称)”,“Weinview_01”

或是针对不同的资料取样历史记录需要不同的操作方式。

针对资料取样元件文件名称= Voltage的特别定义标准

5:“dtl”,“EasyConverter /s Voltage.lgs $(路径名称)”,“*”,“Voltage”

上面的标准5只能用于自[资料取样]元件上传的历史资料文件名称为“voltage”时。

第三个参数（“*”）表示接受来自任何触摸屏的资料取样中符合标准者。

用户可以转换第三个参数为“192.168.1.26”，“192.168.1.*”，触摸屏名称等等，用以减少目标触摸屏的范围。

● 转换批次文件格式

以下列出一个标准中的各参数：

编号	参数	说明
1	档案例行	这个参数特定出此标准所针对的上传档案类型的副档名(e.g. “dtl”为资料取样历史记录，“evt”为事件记录历史档案)。
2	指令（行）	当上传档案符合标准时，EasyPrinter送至命令串口的确切指令。
3	a. HMI IP b. HMI名称	此参数将这个标准所针对的触摸屏指定出来
4	条件 1	● 如果档案类型是“dtl” 此参数将这个标准所针对的[资料取样]元件的资料夹名称指定出来。 ● 对其他资料类型无效。
5	条件 2	未使用（保留为以后使用）

● 标准验证顺序

EasyPrinter于每个文件上传后由下往上验证各标准，一旦符合标准，就会停止验证，并开始处理下一个文件。因此，用户可将较广泛的标准放在下方，而将较明确的标准置于上方。以上面提及的5个标准为例，正确的顺序为：

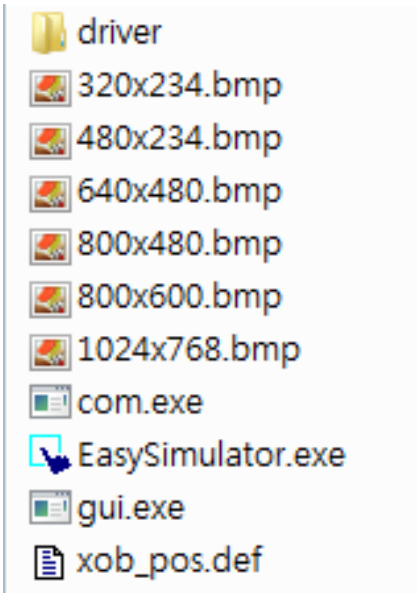
“dtl”，“EasyConverter /s Voltage.lgs $(路径名称)”，“*”，“Voltage”
“dtl”，“EasyConverter /c $(路径名称)”，“WeinVeiw”
“dtl”，“EasyConverter /c $(路径名称)”，“192.168.1.26”
“dtl”，“EasyConverter /c $(路径名称)”
“evt”，“EasyConverter /c $(路径名称)”

第二十七章 EasySimulator

EasySimulator功能允许用户在不安装整个EB8000软件即可执行在线/离线模拟，但是用户必须先准备好以下所需的相关资料。

1、准备相关资料

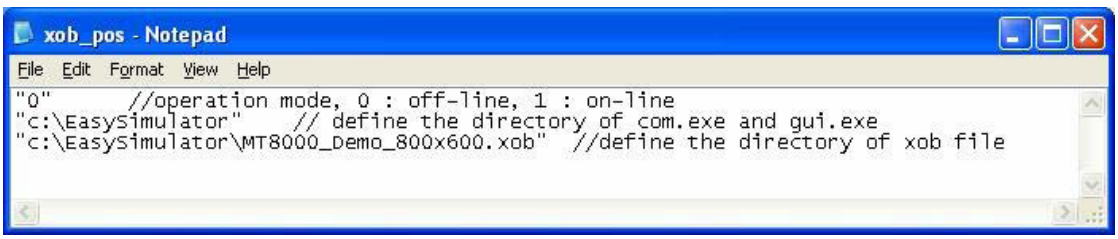
1. [driver] → [win32]
2. 320x234.bmp
3. 480x234.bmp
4. 640x480.bmp
5. 800x480.bmp
6. 800x600.bmp
7. 1024x768.bmp
8. com.exe
9. gui.exe
10. xob_pos.def
11. EasySimulator.exe



注意：以上相关资料皆可在EB8000安装目录文件夹里找到，也就时说，用户仍需在某个PC上先安装EasyBuilder8000软件，再将这些文件复制到目标PC上。

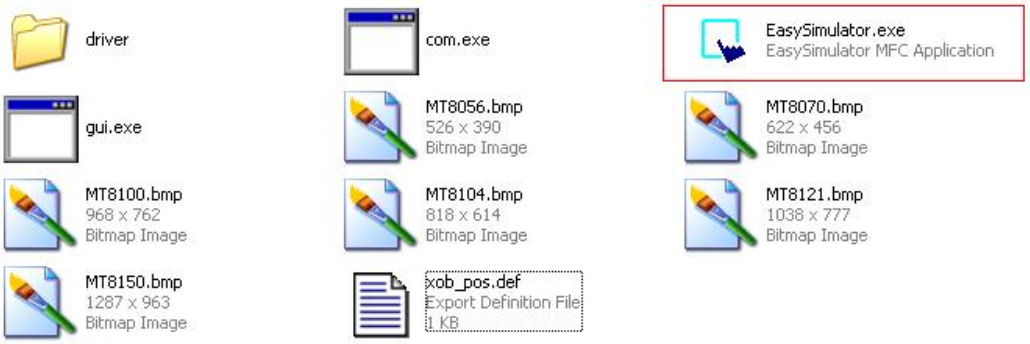
2、设定xob_pos.def内容

步骤1：使用文字编辑工具（记事本）打开xob_pos.def，并正确设定相关内容：



行数	叙述
1	**["0"]**：执行离线模拟 **["1"]**：执行在线模拟
2	指出相关文件的完整存放路径 (e.g. com.exe, gui.exe, EasySimulator.exe…等等)
3	指出 *.xob 文件完整的存放路径

步骤2：双击EasySimulator.exe，即可开始执行模拟：



步骤3：在线/离线模拟的结果将显示在电脑屏幕上。

第二十八章 使用串口实现一机多屏功能(主从模式)

使用串口实现一机多屏是指: 触摸屏透过串口(COM port)连接远端的触摸屏, 并读取连接在远端触摸屏上PLC的数据, 参考下图:

上图显示PLC连接在HMI 1上, HMI 1与HMI 2使用串口直接连接, HMI 2可以透过HMI 1读取PLC上的数据。

下面将以上图为例, 说明如何使用EB8000规划HMI 1与HMI 2所使用的工程文件, 实现一机多屏的功能。

28.1 如何设定HMI 1(主机)所使用工程文件的内容

下图为HMI 1所使用工程文件[系统参数]中[设备列表]的内容。

a、因为HMI 1的COM1连接PLC, 所以设备列表中需存在[本机PLC 1], 并设定正确的PLC通讯参数, 假设此时所连接的PLC为FATEK FB Series。

b、因为HMI 1的COM2用来接收来自HMI 2的命令, 所以必须建立[Master-Slave Server]类型的设备, 用来设定COM2的属性。

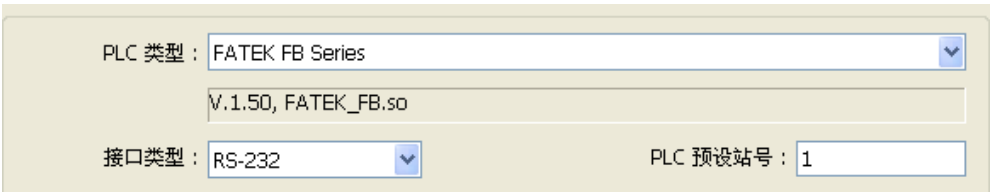
由上图可以发现COM 2的通讯参数为[115200、E、8、1], 并使用RS232通讯。此项参数并不限定需与PLC的通讯参数相同, 但限制数据位(data bits)必须为8。另外, 尽可能设定为较快的通讯速度, 这样HMI 2可以比较有效率读取到PLC的数据。

28.2 如何设定HMI 2（从机）所使用工程文件的内容

编号	名称	位置	设备类型	接口类型
本机 触摸屏	Local HMI	本机	MT6070iH/MT8070...	停用
*远端 PLC 1	FATEK FB Series	COM 1 (主-从模式)	FATEK FB Series	COM1 (115200

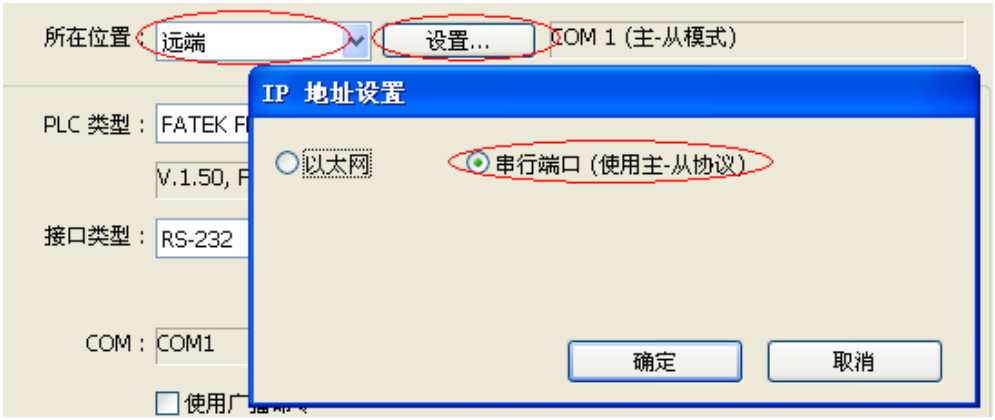
上图为HMI 2工程文件[系统参数]/[设备列表]的设定内容，因为HMI 2所读取的PLC连接在HMI 1上，所以HMI 2将PLC视为远端PLC，因此在设备列表中需存在[*远端PLC 1]，此时所连接的PLC为FATEK FB Series。下文说明如何建立[*远端PLC 1]。

步骤1：在设备列表中建立一个新的设备，[PLC类型]请选择“FATEK FB Series”，[PLC预设站号]需与PLC所使用的站号相同。



步骤2：设定正确的通讯参数，此时HMI 2的COM 1是与HMI 1的COM 2相互连接，并不是与PLC直接连接，因此必须忽略PLC的通讯参数，而应让HMI 2的COM1与HMI 1的COM 2所使用的通讯参数相同，因此HMI 1的COM 2使用RS232通讯，通讯参数为[115200、E、8、1]，所以HMI 2的COM1也需依此参数设定，参考下图：

步骤3：因为HMI 2视PLC为远端PLC，所以需选择[所在位置]为[远端]，并选择使用 [串行端口(使用主–从协议)]的连接方式连接远端HMI (即HMI 1)。



编号	名称	位置	设备类型	接口类型
本机 触摸屏	Local HMI	本机	MT6070iH/MT8070...	停用
*远端 PLC 1	FATEK FB Series	COM 1 (主-从模式)	FATEK FB Series	COM1 (115200

完成上述的各项步骤，在[设备列表]中可以发现新增一项设备：[*远端PLC 1]，此设备包含“*”符号，用来表示名称中包含[远端]，但实际上仍由本机的串行端口发送命令与接收回复，所以与PLC的连接状态只需检视本机的系统保留地址即可；也就是[*远端PLC 1]，[*远端PLC2]，[*远端PLC 3]与[本机PLC 1]，[本机PLC2]，[本机PLC 3]使用相同的系统保留地址，这些系统保留地址包含：

地址	叙述
LB-9150	状态为 ON 时，若与连接在 COM1 的 PLC 断线，系统将自动连线 状态为 OFF 时，忽略与此 PLC 的断线状态
LB-9151	状态为 ON 时，若与连接在 COM2 的 PLC 断线，系统将自动连线 状态为 OFF 时，忽略与此 PLC 的断线状态
LB-9152	状态为 ON 时，若与连接在 COM3 的 PLC 断线，系统将自动连线 状态为 OFF 时，忽略与此 PLC 的断线状态

LB-9200~ LB-9455	这些寄存器用来指示与连接在 COM1 的 PLC 间的连线状态 LB9200 指示与站号为 0 的 PLC 的连线状态，LB9201 指示与站号为 1 的 PLC 的连接状态，以此类推 状态为 ON 表示目前连线正常 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 再连线一次
LB-9500~ LB-9755	这些寄存器用来指示与连接在 COM2 的 PLC 间的连线状态 LB9500 指示与站号为 0 的 PLC 的连线状态，LB9501 指示与站号为 1 的 PLC 的连接状态，以此类推 状态为 ON 表示目前连线正常 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 再连线一次
LB-9800~ LB-10055	这些寄存器用来指示与连接在 COM3 的 PLC 间的连线状态 LB9800 指示与站号为 0 的 PLC 的连线状态，LB9801 指示与站号为 1 的 PLC 的连接状态，以此类推 状态为 ON 表示目前连线正常 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 再连线一次

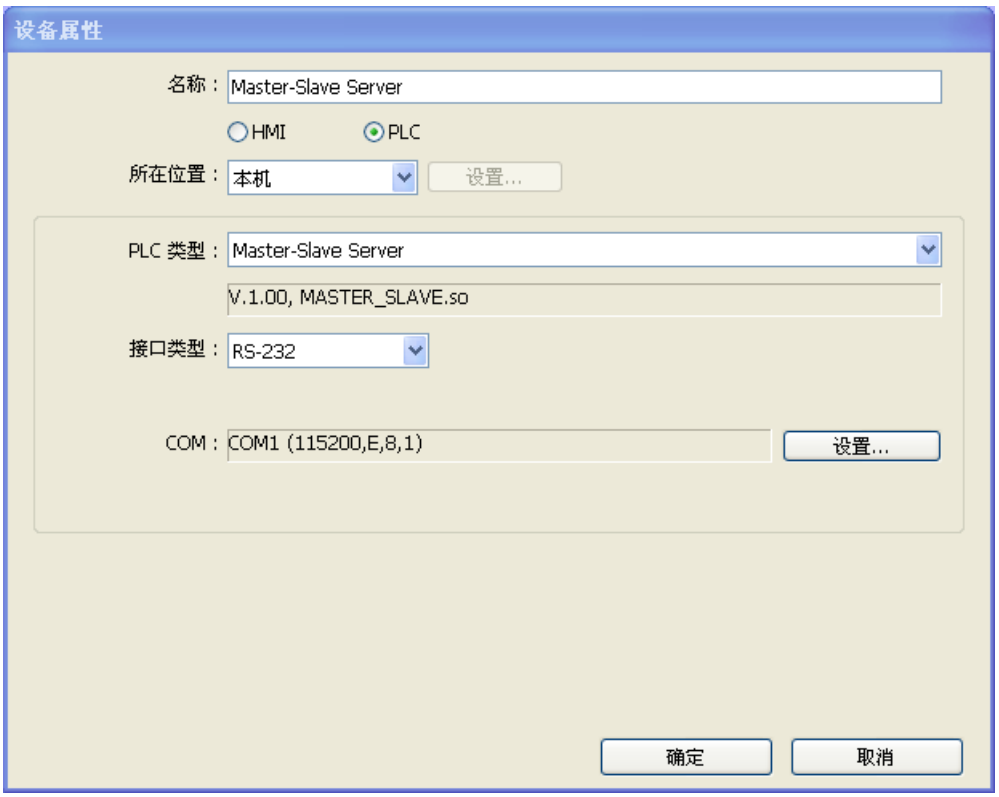
28.3 如何连接MT500

让MT500可以使用MT500的Master-Slave protocol来读写MT6000/8000的Local data和连接的PLC的数据。

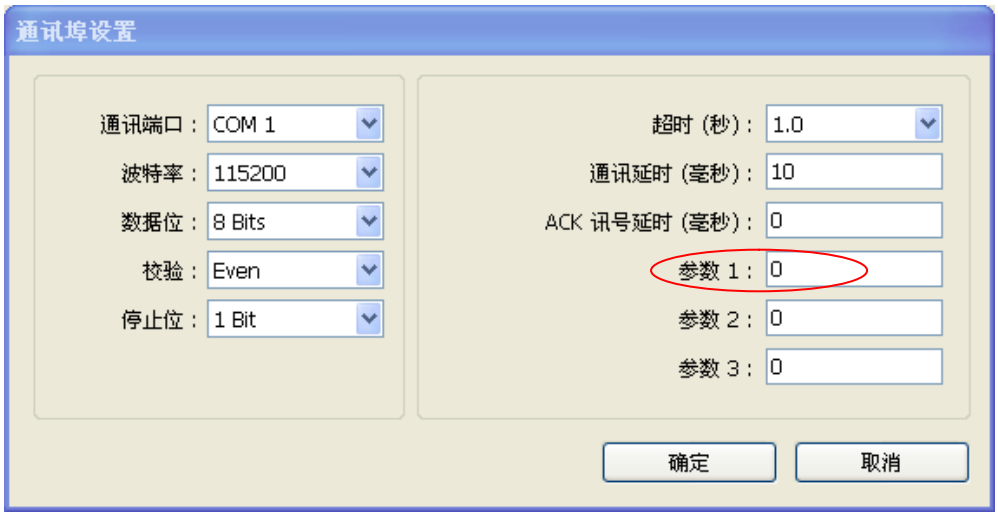
◎ MT8000设定方式

步骤1：选取“Master-Slave Server”驱动并点击[设定]。

步骤2: 选择RS232接口类型, 并点击[设置]



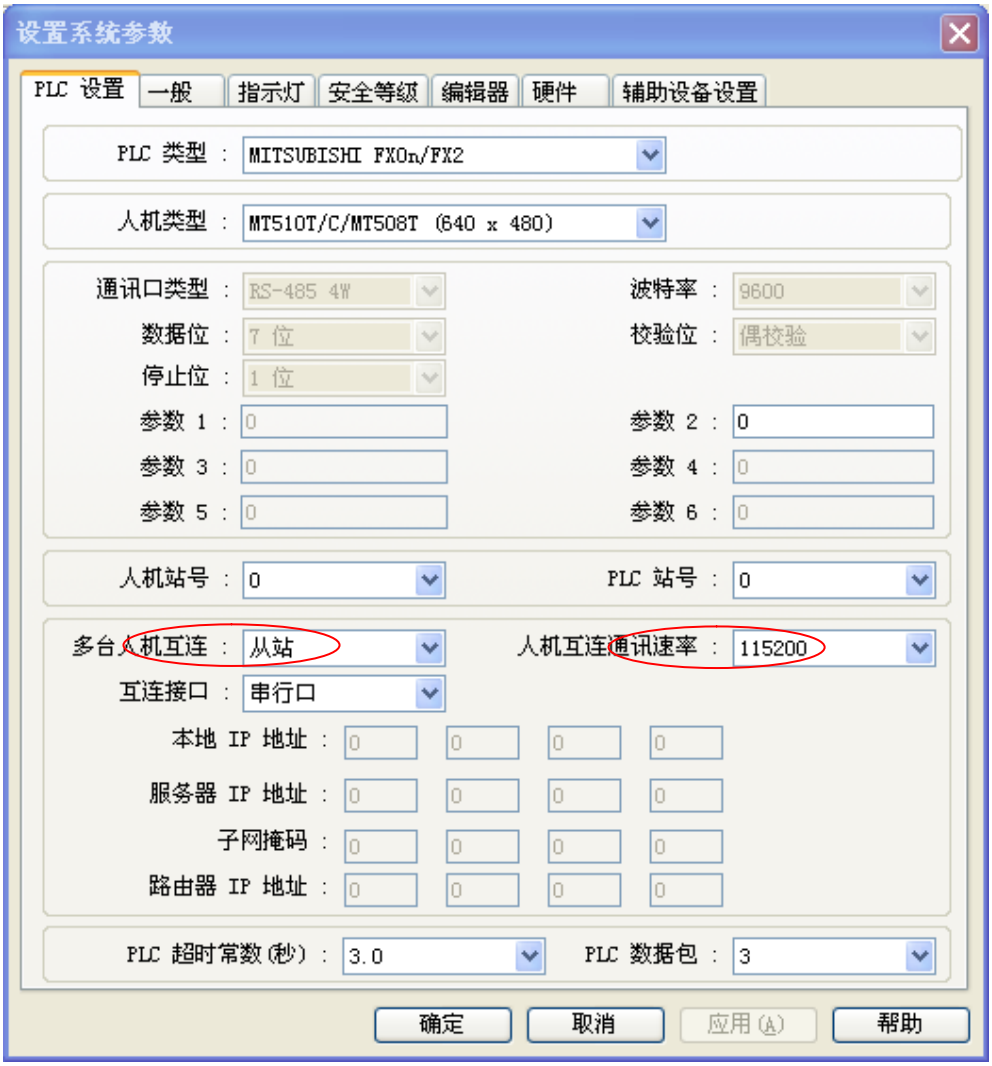
步骤3: 参数1要填入MT500 PLC ID No.(请参考MT500设定)



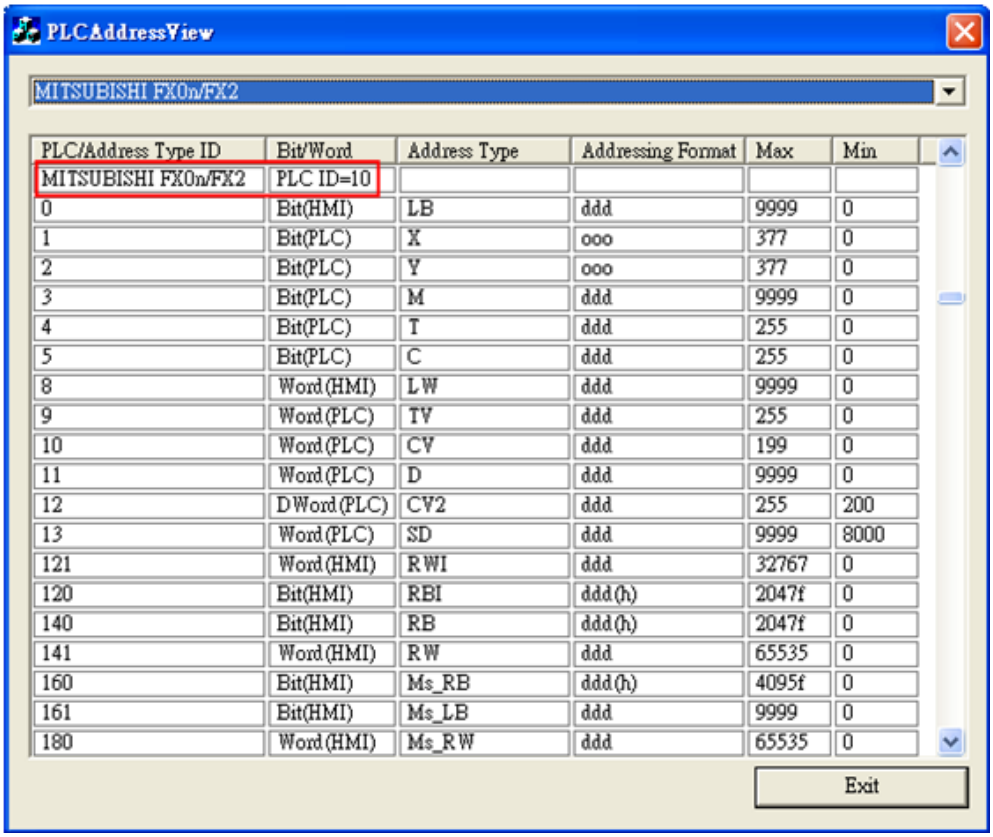
◎ MT500设定方法

步骤1: 在EB500的系统参数设定内设定多台HMI互连: 从机; HMI互连通讯速度: 115200。

注意: HMI互连通讯速率的设定MT500与MT8000要相同。



步骤2：双击PLC Address View.exe来查询PLC的ID No.并填入MT8000的参数1内。



步骤3：COM Port要使用RS232和MT8000的RS232连接后即可通讯。

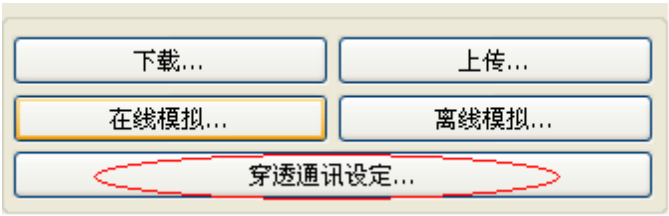
Device address:

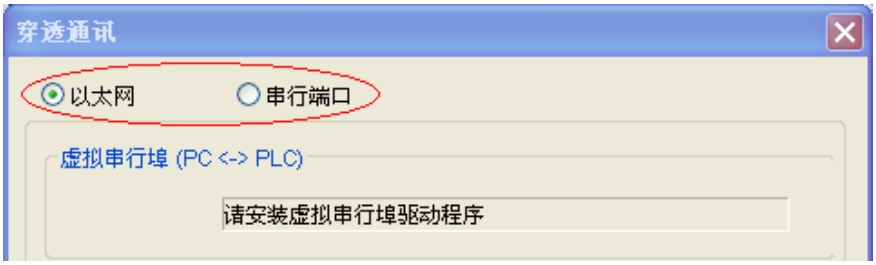
Bit/Word	MT500	MT8000	Range	Memo
B	Ms_RB	RW_Bit	dddd:0~4095 (h): 0~f	
B	Ms_LB	LB	dddd:0~9999	
W	Ms_RW	RW	ddddd:0~65535	
W	Ms_LW	LW	dddd:0~9999	

第二十九章 穿透通讯功能

MT8000所提供的穿透通讯功能允许PC上的应用程序透过HMI直接控制PLC，此时HMI所扮演的角色类似转换器（converter）的功能。

穿透通讯功能包含[串行端口]（COM）与[以太网]（Ethernet）两种模式，点击[Project Manager]的[穿透通讯设定…]按钮，即可检视这两种模式的设定内容。

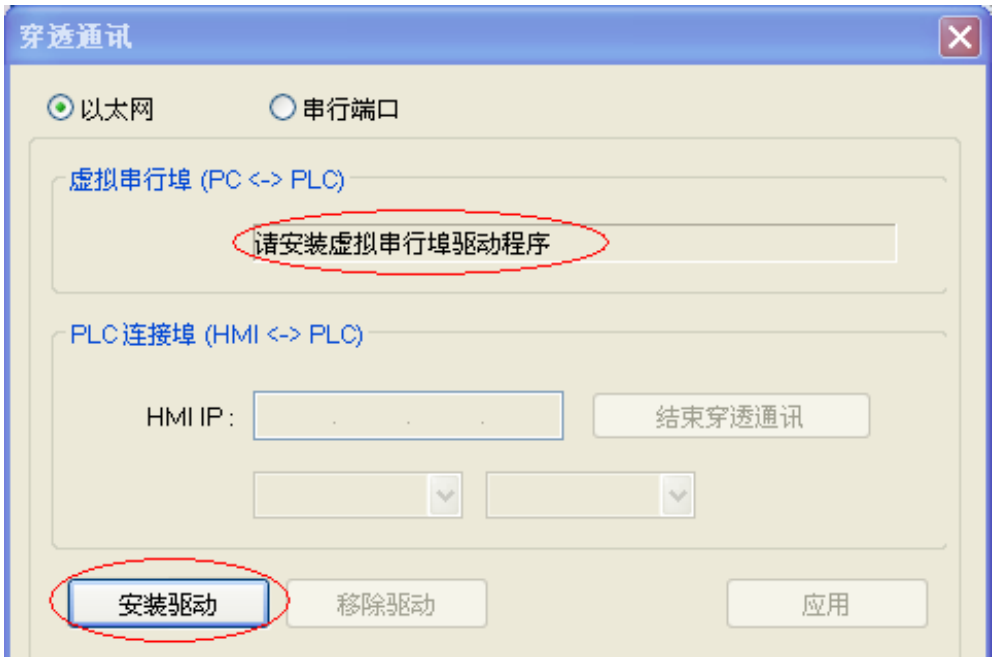




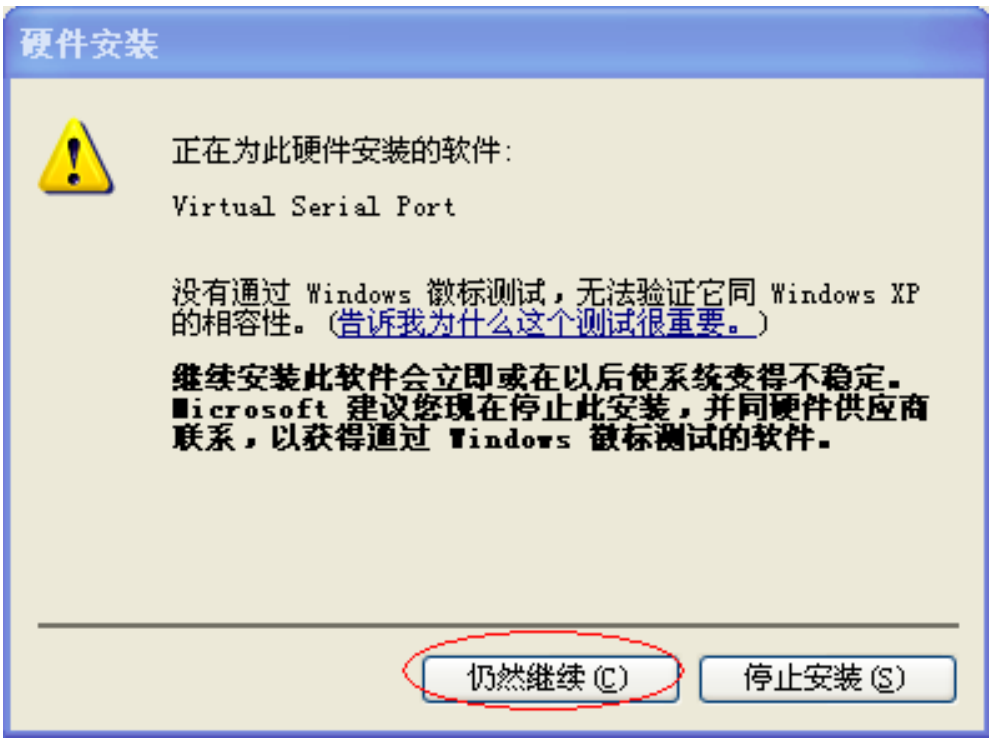
29.1 以太网模式

如何安装虚拟串行端口驱动程序：在使用[以太网]穿透通讯功能前，要先安装Weinview虚拟串行端口驱动程序，安装步骤如下：

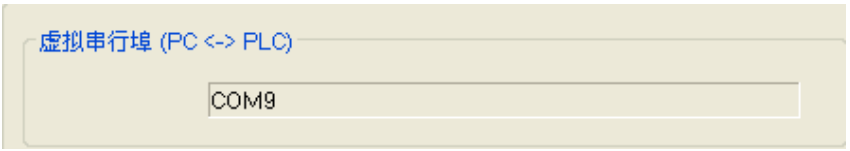
（1）首先打开Project Manager检视目前驱动程序的安装状态，若画面显示[请安装虚拟串行端口驱动程序]，请点击[安装驱动]按钮，参考下图：



（2）在安装驱动程序的过程中若出现下列画面，请选择[仍然继续]

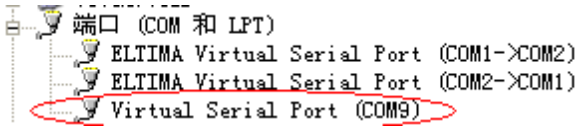


（3）在完成驱动程序安装后，原来显示[请安装虚拟串行端口驱动程序]的位置将显示目前所使用的虚拟串行端口，下图显示目前所使用的虚拟串行端口为COM 9。

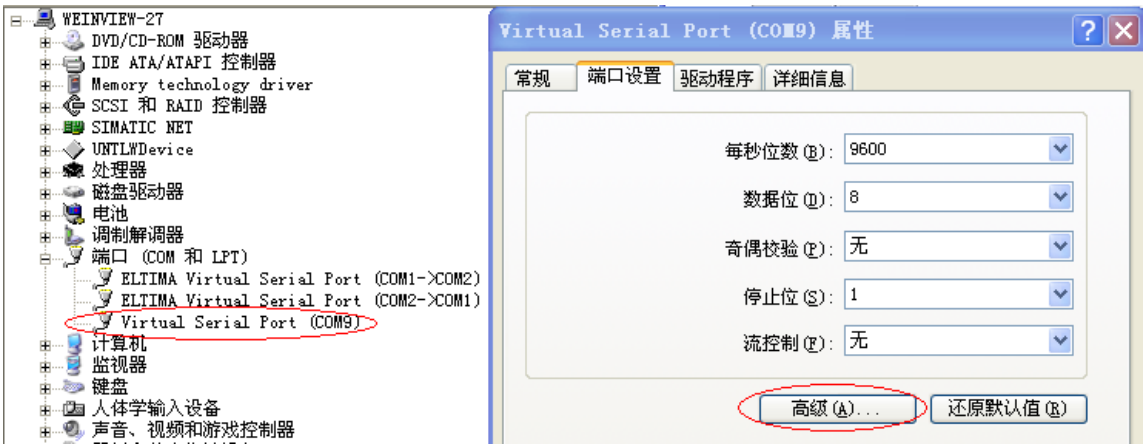


- 如何更改虚拟串行端口

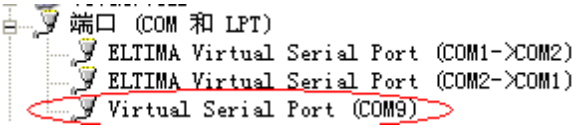
在[计算机管理]→[设备管理器]的内容中也可发现已安装好 [Virtual Serial Port]。

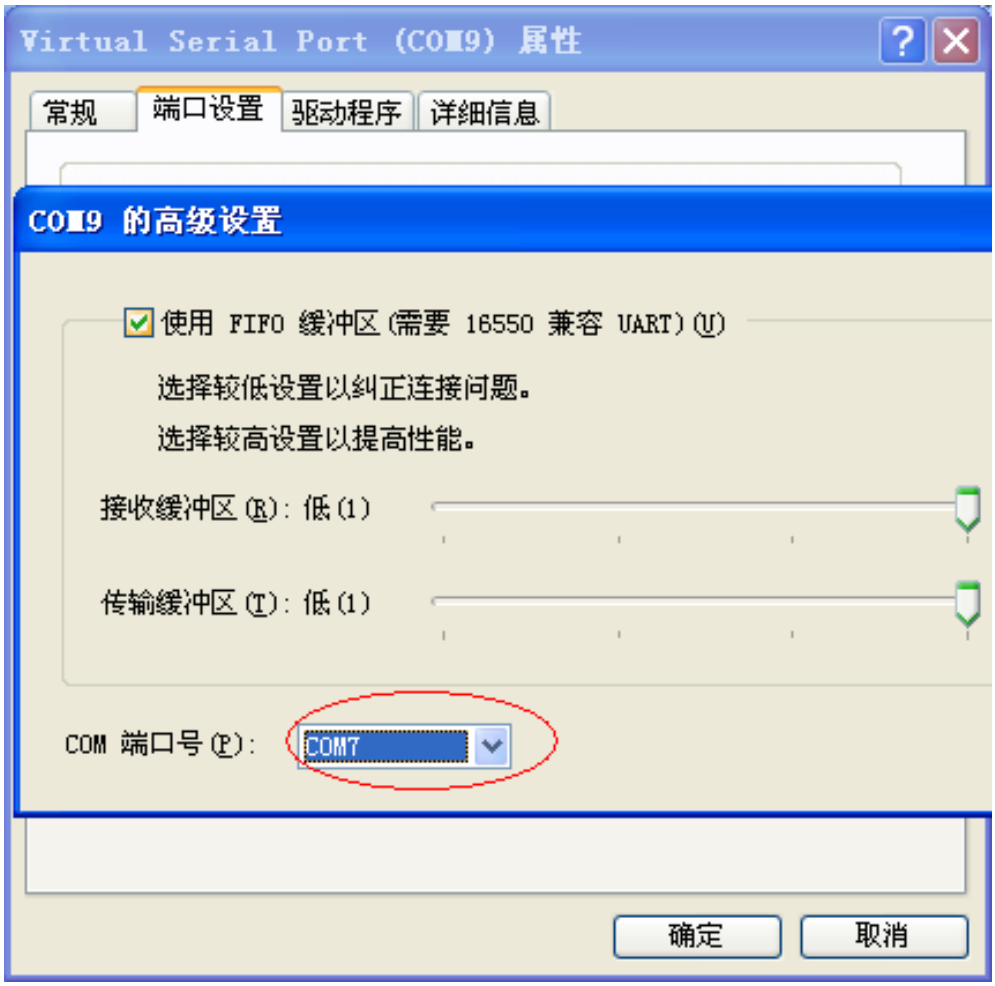


若要更改虚拟串行端口的号码，只需点击[Virtual Serial Port]并选择[端口设置]下的[高级]按钮，即可更改虚拟串行端口的号码。

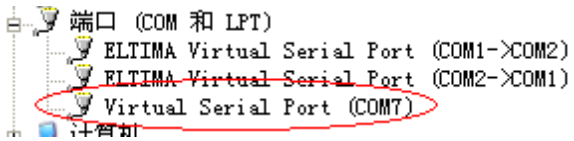


下图显示虚拟串行端口将更改为COM 7

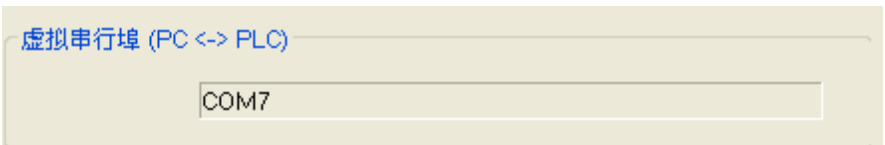




触控[确定]按钮后，虚拟串行端口已被更改为COM 7



打开[Project Manager]同样可以发现虚拟串行端口已被更改为COM 7



- **如何使用以太网实现穿透通讯**

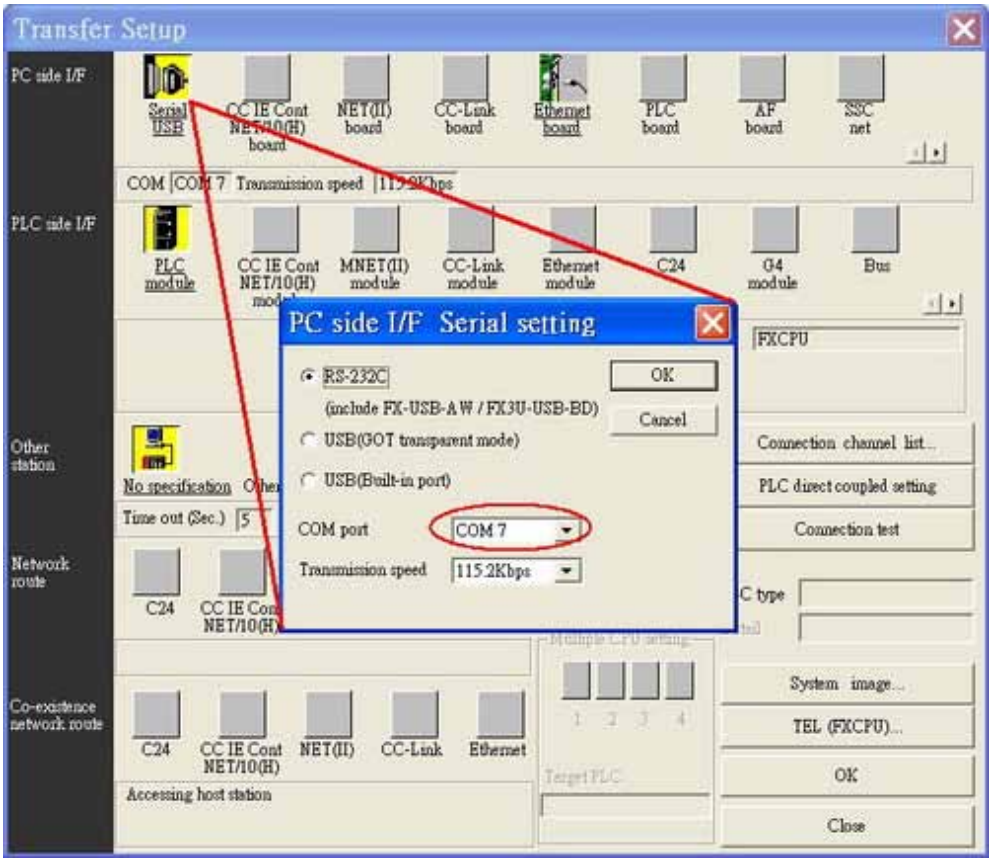
在安装完成虚拟串行端口驱动程序后，只需触控面四个步骤即可使用以太网实现穿透通讯。

步骤1: 首先设定连接PLC的HMI的IP地址, 下图显示目前的IP地址为192.168.1.28.

步骤2: 指定触摸屏连接PLC的串行端口与串行端口属性, 下图显示此时使用COM2, RS232通讯方式连接PLC

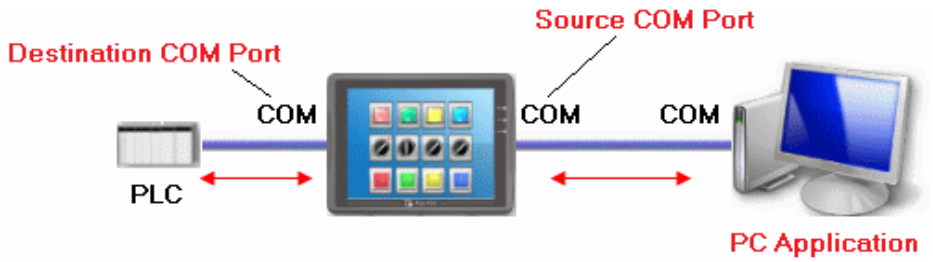
步骤3: 完成所有设定后, 需触控[应用]按钮, 所有属性才会生效。

步骤4: 在执行电脑上的应用程序时, 所使用的串行端口需指定使用虚拟串行端口, 以Mitsubishi的应用程序为例, 假如此时的虚拟串行端口为COM 7, 则在[PC side I/F Serial setting]对话框中的[COM port]需选择COM 7, 请参考下图:



完成上面各项设定后，用户在执行电脑上的PLC应用程序时，触摸屏会自动切换为穿透通讯模式（**此时将暂停触摸屏与PLC之间的通讯**），此时可以将应用程序视为直接使用虚拟串行端口（virtual serial port）控制PLC，请参考下图；在关闭应用程序时，触摸屏也会自动关闭穿透通讯模式。

29.2 串行端口模式



[数据来源串口] (source COM port)是指MT8000与电脑连接的串口

[数据目标串口] (destination COM port)是指MT8000与PLC连接的串口

在使用[串行端口]实现穿透通讯时，需正确设定这两个串口的属性。

● 串行端口设定

MT8000提供两种方式让用户启动[串行端口]穿透通讯功能

[方法一]：使用Project Manager开启串行端口穿透通讯功能

[方法二]：使用触摸屏的系统保留地址

LW9901设定数据来源串口(1~3: COM1~COM3)

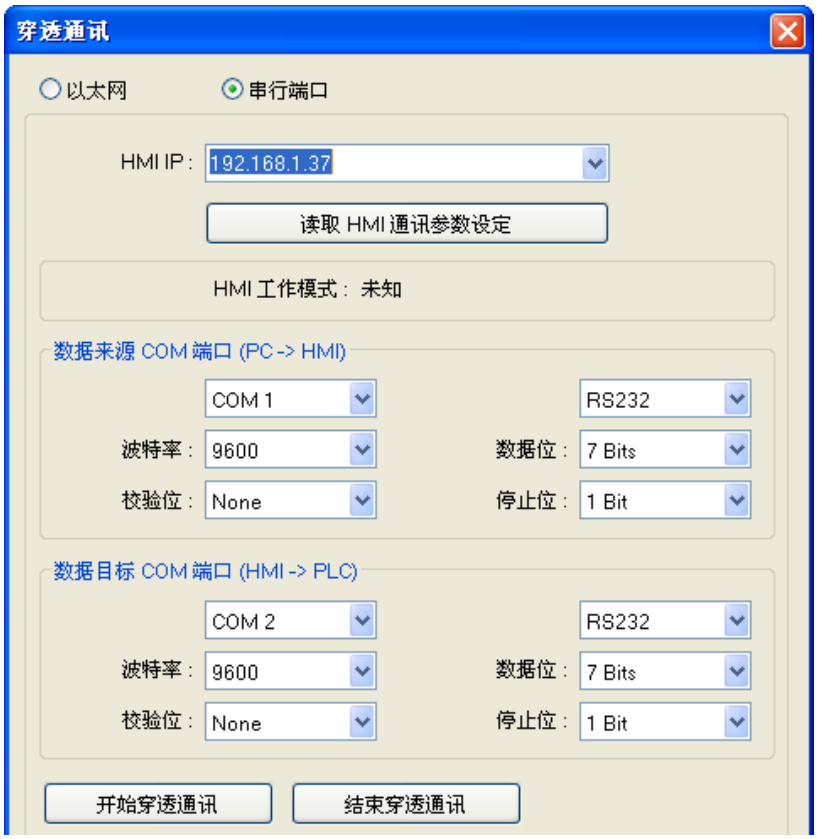
LW9902设定数据目标串口(1~3: COM1~COM3)，即可启动穿透通讯功能

注意：当使用完穿透通讯功能，需要点击[结束穿透通讯]来关闭穿透通讯功能，此时HMI才会重新开启和PLC的通讯。

[使用Project Manager开启串行端口穿透通讯功能]

可以使用Project Manager开启串行端口穿透通讯设定页，参考下图：

下面说明串行端口穿透通讯功能设定页的各项功能。



[HMI IP]: 当使用Project Manager开启串行端口穿透通讯功能时, 需指定触摸屏的IP地址。

[读取HMI通讯参数设定]: 此项功能用来读取触摸屏上数据来源串口与目标串口的各项设定值, 这些设定值来自不同的系统保留字, 下面列出这些系统保留字的详细内容。

数据来源与目标串口

地址	叙述
LW9901 (数据来源串口)	1 : COM 1　2 : COM 2　3 : COM 3
LW9902 (数据目标串口)	1 : COM 1　2 : COM 2　3 : COM 3

COM 1设定值

地址	叙述
LW9550 (通讯模式)	0 : RS232　1 : RS485/2W　2 : RS485/4W
LW9551 (传输速率)	0 : 4800　1 : 9600　2 : 19200　3 : 38400 4 : 57600　5 : 115200
LW9552 (数据位)	7 : 7 bits　8 : 8 bits
LW9553 (校验位)	0 : none　1 : even　2 : odd
LW9554 (停止位)	1 : 1 bit　2 : 2 bits

COM 2设定值

位址	敘述
LW9556 (传输速率)	0 : 4800　1 : 9600　2 : 19200　3 : 38400 4 : 57600　5 : 115200
LW9557 (数据位)	7 : 7 bits　8 : 8 bits
LW9558 (校验位)	0 : none　1 : even　2 : odd
LW9559 (停止位)	1 : 1 bit　2 : 2 bits

COM 3设定值

地址	叙述
LW9560 (通讯模式)	0 : RS232　1 : RS485/2W
LW9561 (传输速率)	0 : 4800　1 : 9600　2 : 19200　3 : 38400 4 : 57600　5 : 115200
LW9562 (数据位)	7 : 7 bits　8 : 8 bits
LW9563 (校验位)	0 : none　1 : even　2 : odd
LW9564 (停止位)	1 : 1 bit　2 : 2 bits

在触控[读取HMI通讯参数设定]按钮后, 所有通讯参数将被更新。

- HMI工作模式

共有三种显示目前触摸屏的工作模式

设定	叙　述
未知	在未读取触摸屏的设定值前，所显示的触摸屏的工作模式为“未知”
正常模式	在读取触摸屏的设定值后，工作模式如为“正常模式”，表示 HMI 处在正常通讯状态，不接受来自数据来源串口的任何数据。
穿透模式	若工作模式为“穿透模式”，表示触摸屏目前处在穿透模式状态，此时 PC 上的应用程序可以透过数据来源串口直接控制连接在数据目标串口上的 PLC。

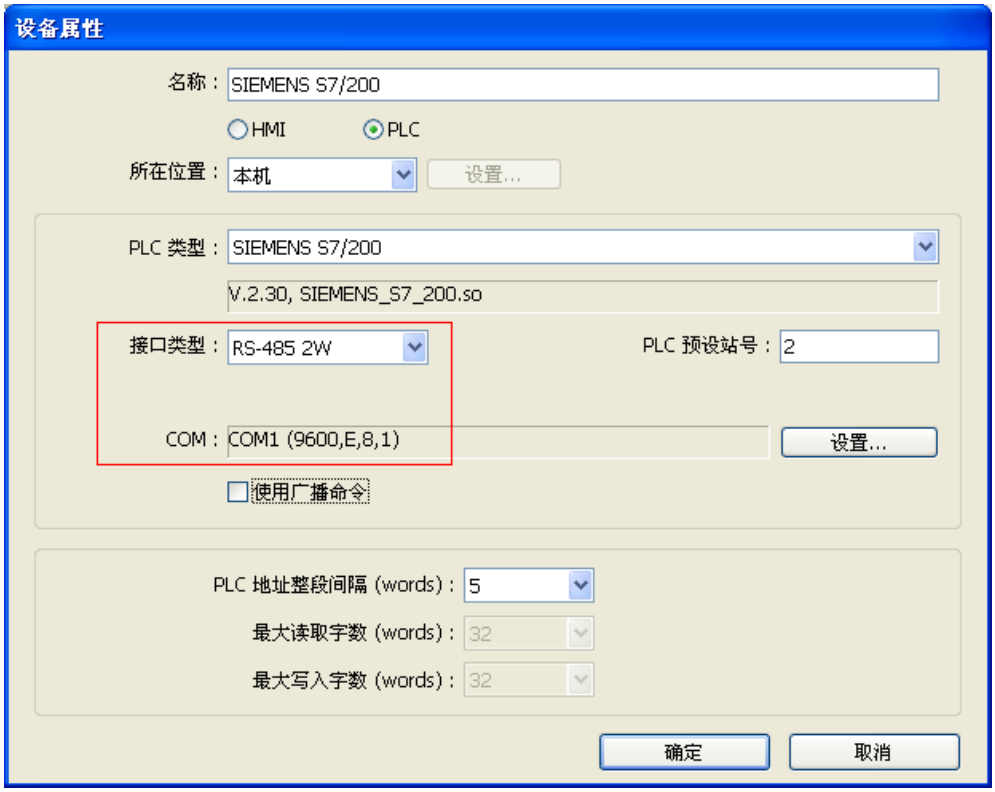
[数据来源串行端口]、[数据目标串行端口]

用来显示与设定数据来源串口和数据目标串口的各项通讯参数，当用户使用[开始穿透通讯]，启动串行端口穿透通讯功能时，将使用[数据来源串行端口]、[数据目标串行端口]中所设定的内容执行穿透通讯功能。

通常[数据来源串行端口]与[数据目标串行端口]中的**“波特率”**、**“数据位”**、**“校验位”**、**“停止位”**需设定相同。[数据来源串行端口]因为是连接到PC，通讯模式通常选择为“RS232”；[数据目标串行端口]则是因为连接到PLC，所以通讯模式需依照PLC的要求来决定，可以选择“RS232”、“RS485 2W”、“RS485 4W”。

下图为连接SIEMENS S7–200时的设定内容，COM2连接到PC，COM1连接到PLC，此时PLC所使用的通讯参数为“9600 E 8 1”，PLC使用RS485 2W通讯模式。

在使用穿透通讯前，请先将工程文件设定正确的通讯参数并下载到触摸屏。



下载程序到触摸屏之后，打开PLC的工程文件，将PLC界面及串行端口切换至COM2 RS232（因为PC使用COM2与触摸屏相连）

点击[穿透通讯]设定触摸屏IP，例如：192.168.1.37。

最后，点击[读取HMI通讯参数设定]，如下设定：

点击[开始穿透通讯], 触摸屏将切换至穿透通讯模式, 用户可以执行在线监控, 这时PC应用程序可以通过触摸屏控制PLC, 而触摸屏此时可以被视为转换器。

注意:触摸屏与PLC之间的通讯在执行此功能时会被暂停,如果用户想恢复两者之间的通讯,请点击[结束穿透通讯]停止穿透功能。

29.3 使用HMI的系统保留字启动穿透通讯功能

另一种启动触摸屏穿透通讯功能的方式为直接更改触摸屏的系统保留字LW9901（数据来源串口）与LW9902（数据目标串口）中的数据内容，当LW9901与LW9902中的数据符合下列条件时，触摸屏将自动启动穿透通讯功能：

a. LW9901与LW9902中的数据需为1或2或3 (1、2、3分别表示为COM 1、COM 2、COM 3) 。

b. LW9901与LW9902中的数据不可相同。

如有需要更改各串口的通讯参数，只需更改各参数相对应的系统保留字中的数据，并对系统保留地址LB9030、LB9031、LB9032送出ON的信号，强迫HMI使用这些新的设定即可。这些系统保留字请参考前面的说明，其中：

LB9030：设定为ON，HMI将使用相关系统保留字中的数据更新COM 1的通讯设定

LB9031：设定为ON，HMI将使用相关系统保留字中的数据更新COM 2的通讯设定

LB9032：设定为ON，HMI将使用相关系统保留字中的数据更新COM 3的通讯设定

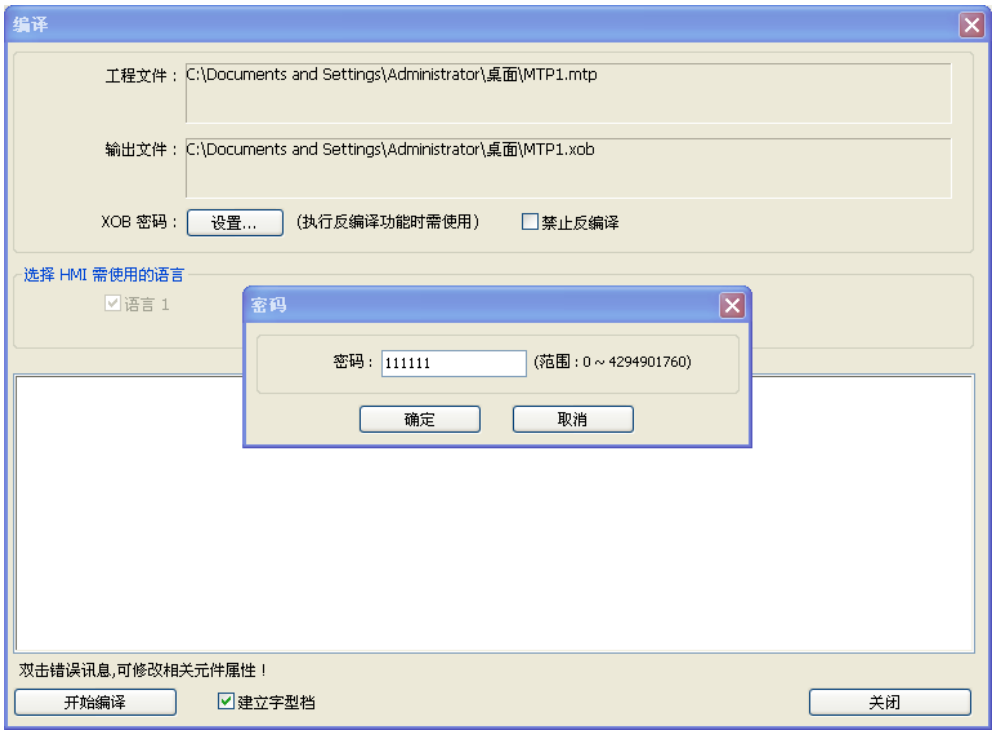
注意：如果关闭触摸屏的穿透通讯功能，只需将LW9901与LW9902中的数据更改为非1、2、3的数据即可（例如更改为0）。

第三十章 工程档案保护功能

EB8000提供工程档案的保护功能，用来保护作者的设计成果。

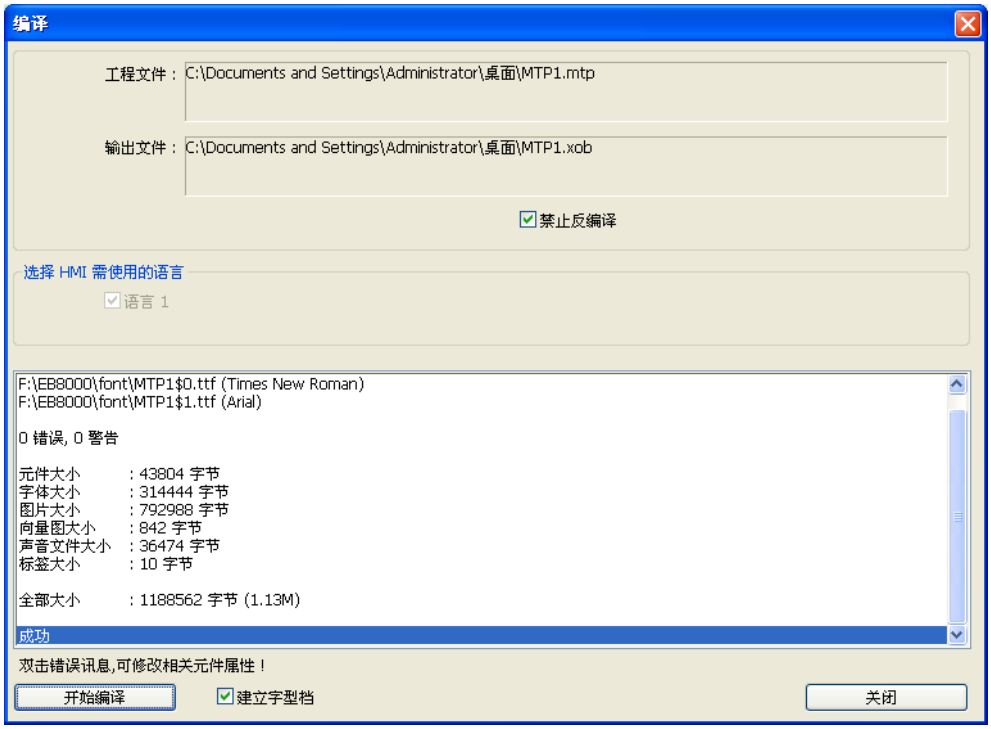
1 XOB文件加密功能

在使用EB8000完成工程文件（MTP）时，使用EB8000提供的编译功能，将MTP文档编译为下载至触摸屏所需的XOB文件，用户可设定[XOB密码]，若要反编译XOB文件，则必须输入密码才能完成动作。XOB密码设定（XOB密码范围：0~4294901760）。



2 禁止反编译

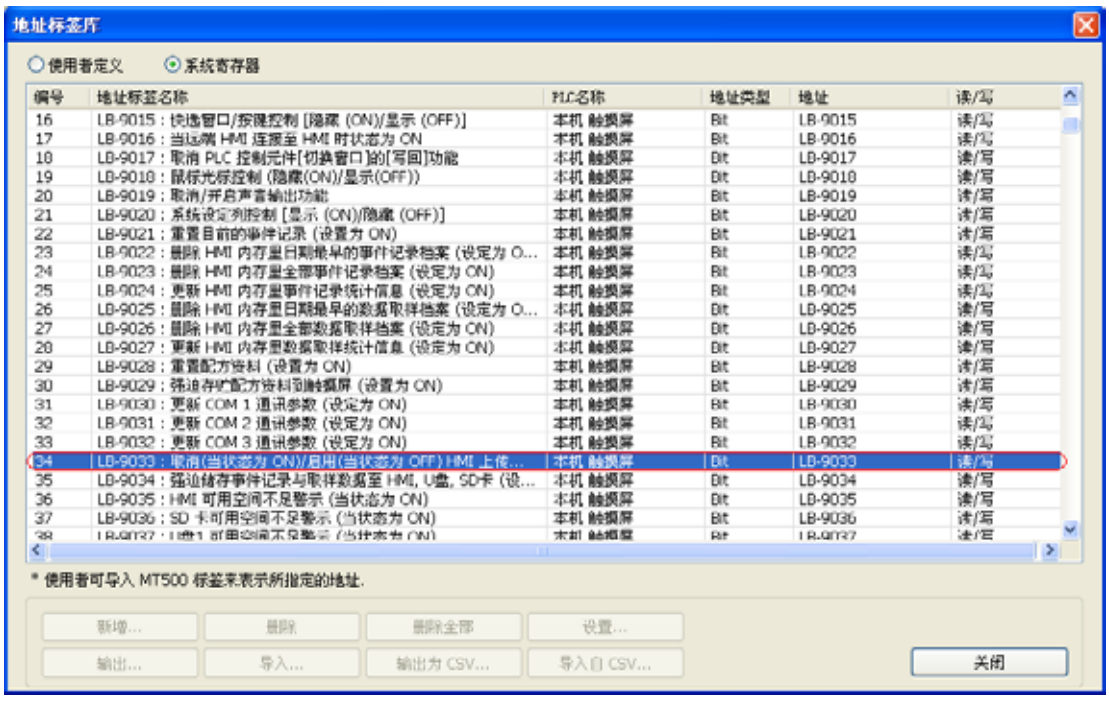
使用EB8000提供的编译，若勾选[禁止反编译]，则用户无法设定[XOB密码]，而且无法将XOB文件反编译为MTP文件。



3 禁止XOB上传功能

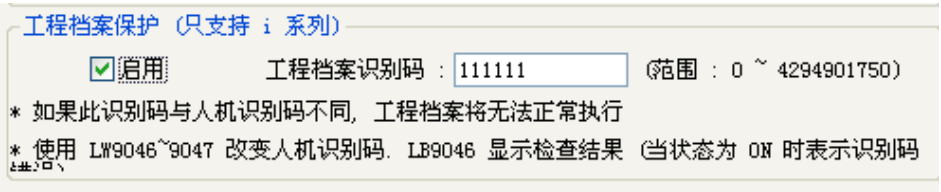
EB8000提供两种方式，禁止XOB文件从触摸屏上传：

- 使用系统寄存器LB9033：当此位地址被设定为ON时，触摸屏将关闭XOB的上传功能；使用此项功能，必须重新启动触摸屏，才会应用设定后的内容；
- 在“系统参数设定–系统设置”中勾选“取消上传功能”。

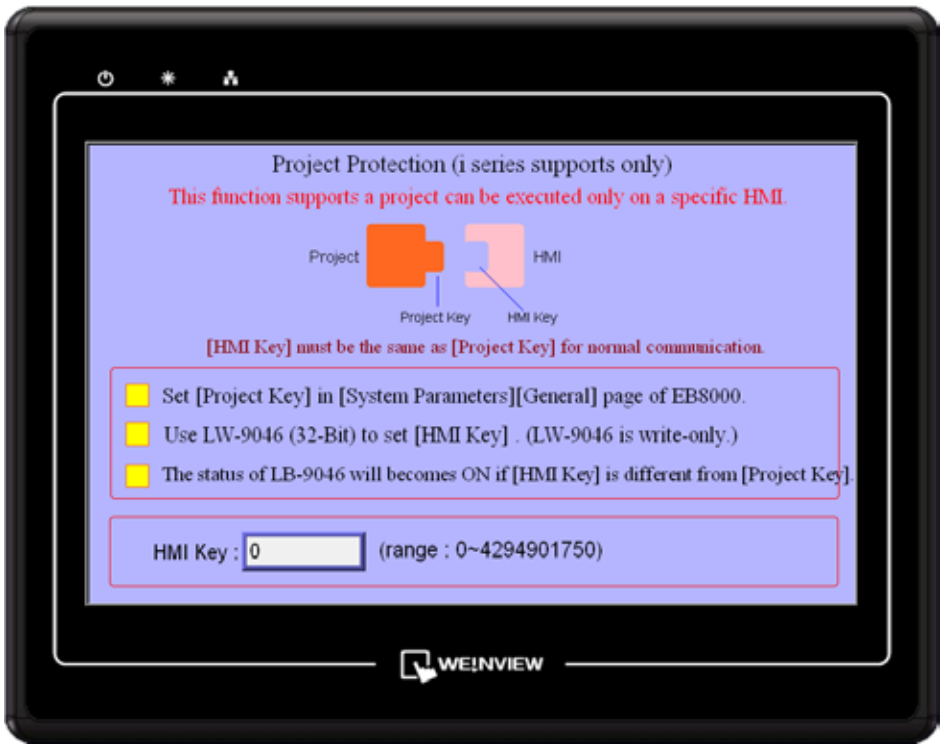


4 工程档案识别码

用户的工程文件可以被限制在指定的触摸屏上使用（该功能只支持i系列触摸屏），下图为[系统参数]/[一般属性]设定页中[工程档案保护]的设定画面。

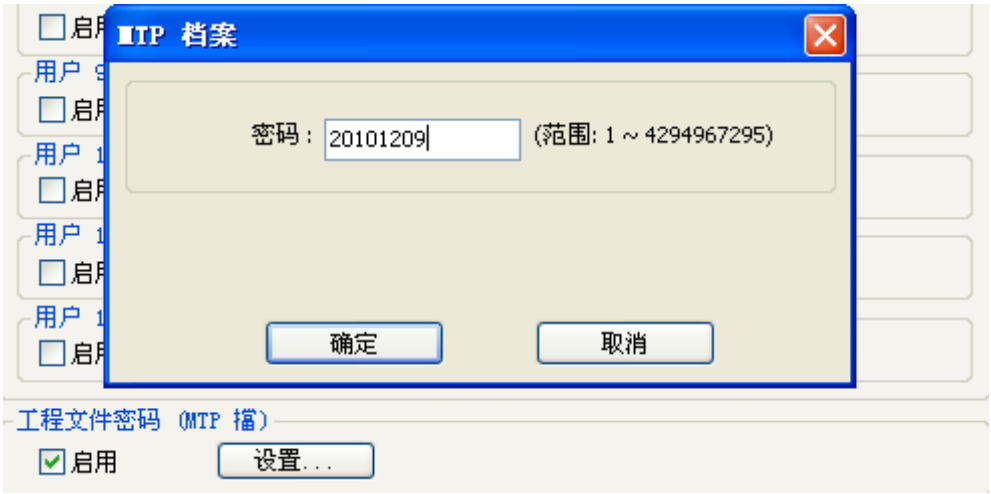


用户可以使用LW9046～LW9047(共32-bit)设定触摸屏的[HMI识别码]，其数据无法被取代或远端输入，当启用工程档案保护功能时，用户可设定[工程档案识别码]（范围0~4294901750），且编译后所得到的XOB文件，只能在[HMI识别码]与[工程档案识别码]相同的触摸屏上执行。若[HMI识别码]与[工程档案识别码]不相同时，LB9046的状态将被设定为ON；修改[HMI识别码]，必须重新启动触摸屏，该参数才有效。



5 工程文件密码(MTP档)

用户可以设定保护MTP文件的密码,下图为[系统参数]/[用户密码]设定页。



启用此功能后,当用户想要编辑MTP文件时,必须输入设定密码,范围为1~4294967295

完成此项设定后,每次要打开这个工程文件时,都会弹出一个窗口请用户输入密码才能进入工程文件。

第三十一章 Memory Map

MemoryMap通信协议类似于IBM 3764R通信协议，它的应用的场合一般是所对应寄存器数据的变化量比较少。(太频繁的变化会导致MemoryMap通信不堪重负)它是两台设备之间的通信协议。MemoryMap通信协议特征是两台设备必须一方为主方，另一方为从方。在通常情况下，主方和从方并没有建立通信，只有当某一方所指定的寄存器数据变化时，通信才建立，双方数据一致后，通信断开。所以通信的目的是保持两台设备(主方和从方)之间相对应的一块相同大小寄存器数据的一致性。

其中主方和从方中对应的寄存器具备MT8000中地址类型为MW(MB)寄存器相同的性质(这块大小为1000字的MW(MB)正是MT8000保留给MemoryMap通信协议所用)寄存器特征是：MB和MW是对相同寄存器区域的映射，根据下表格式，即MB0~MBf映射到MW0，MB10~MB1f映射到MW1…。它们都是指向相同的寄存器内容。

设备名称	格式	范围
MB	ddd(h)	ddd：0~999　h：0~F(hex)
MW	ddd	ddd：0~999

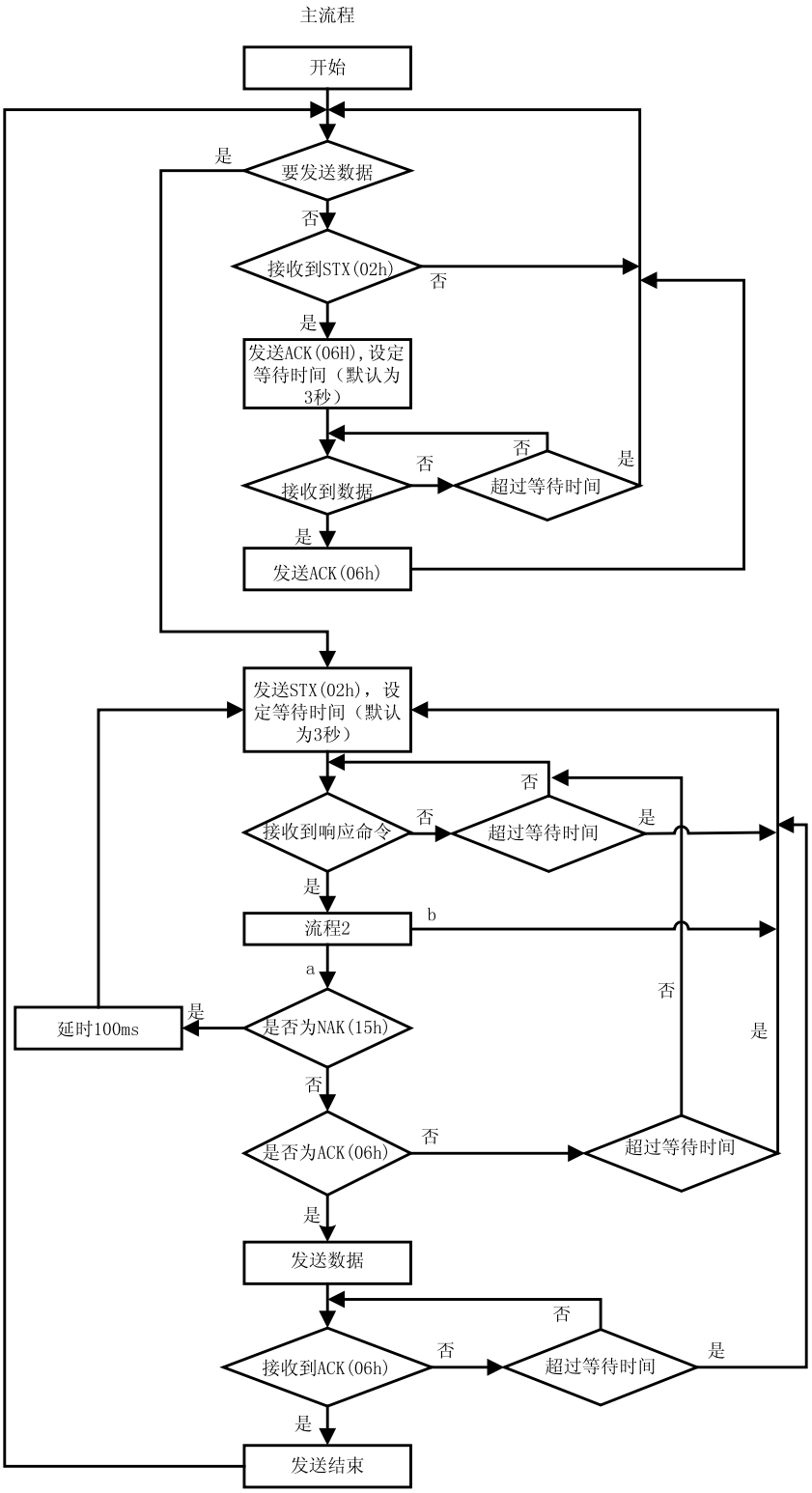
使用MemoryMap通信协议时，主方和从方必须使用相同的通信参数。其接线方式如下：

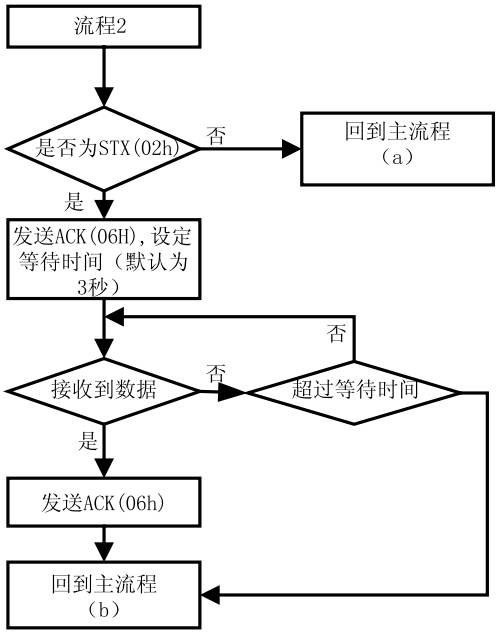
RS232	
主方设备	从方设备
TX(#)	RX(#)
RX(#)	TX(#)
GND(#)	GND(#)

RS485(4W)	
Master 设备	Slaver 设备
TX+(#)	RX+(#)
TX-(#)	RX-(#)
RX+(#)	TX+(#)
RX-(#)	TX-(#)
GND(#)	GND(#)

注：#表示由具体PLC或控制器决定。

通信过程的流程图如下所示：





注意：其中流程2对从方有效，对主方无效，STX为通信请求信号，ACK为响应请求信号，NAK为忙信号。

数据的格式可分为两种，一种是对MB的操作格式，一种是对MW的操作格式：

对于 MB 的命令		
偏移量(字节)	格式	描述
0	0x02	对 MB 操作的标志
1 2	0x## 0x##	地址(低字节) 地址(高字节) 如果是 MB12，则 1*16+2=18，为 0x12，0x00
3	0x00(或 0x01)	表示所指定 MB 地址的数据内容(因为是 Bit 类型，只能是 0 或 1)
4，5	0x10，0x03	结束标志
6	0x##	校验和 xor 从第 0 个字节到第 5 个字节

对于 MW 的命令		
偏移量(字节)	格式	描述
0	0x01	对 MW 操作的标志
1 2	0x## 0x##	地址(低字节) 地址(高字节) 如果地址数据中包含一个 0x10，则在 0x10 后再插入一个 0x10，地址表示多出一个字节，命令格式相应的向后推移一个字节，例如地址为 0x10，0x04，则变为 0x10，0x10，0x04
3	0x##	传送的字节数(由于对字操作，字节数一定为偶数)，如果字节数为 0x10， 则在 0x10 后再插入一个 0x10 命令格式相应的向后推移一个字节
4~4+n-1	0x##(L) 0x##(H) 0x##(L)…	为 1，2 字节所对应地址为起始地址的数据，其中 n 为数据的字节数，如果数据中有 0x10，则在 0x10 后再插入一个 0x10，而”传送字节数”不变，n 则为 n+1，依次类推。
4+n，4+n+1	0x10，0x03	结束标志
4+n+2	0x##	校验和，xor 校验和前面所有字节

下面我们来举一个例子来观察通信过程以增加理解。我们假设主方把MW3的内容置为0x0a，根据这个协议，主方立刻会和从方建立通信，从而使得从方接收到数据后把它对应的MW3的内容置为0x0a。过程为：

1. 主方发送STX(0x02h)。
2. 从方接收到主方发送的STX(0x02h)后，发送返回命令ACK(0x06h)。
3. 主方接收到从方的返回命令ACK(0x06h)。
4. 主方发送数据0x01，0x03，0x00，0x02，0x0a，0x00，0x10，0x03，0x19，如下表所示：

偏移量(字节)	格式	描述
0	0x01	对 MW 操作的标志
1 2	0x03 0x00	地址(低字节) 地址(高字节)
3	0x02	传送的字节数(MW3 为两个字节)
4，5	0x0a，0x00	MW3 的内容为 0x0a，0x00
6，7	0x10，0x03	结束标志
8	0x19	校 验 和， 0x01^0x03^0x00^0x02^0x0a^0x00^0x10^0x03=0x19

5. 从方收到主方发送的数据后，发送返回命令ACK(0x06h)。

6. 主方接收到从方的返回命令ACK(0x06h)。

通信完成，主方把更改的MW的地址和内容传送给了从方，从方再更改MW的数据，使得主方和从方对应节点地址内容保持一致。

我们再举一个例子，其中地址和数据中包括0x10，请注意观察资料格式的变化。我们假设从方把MW16的内容置为0x10，根据这个协议，从方立刻会和主方建立通信，从而使得主方接收到数据后把它对应的MW16的内容置为0x10。过程为：

1. 从方发送STX(0x02h)。

2. 主方接收到从方发送的STX(0x02h)后，发送返回命令ACK(0x06h)。

3. 从方接收到主方的返回命令ACK(0x06h)。

4. 从方发送数据0x01，0x10，0x10，0x00，0x02，0x10，0x10，0x00，0x10，0x03，0x10 如下表所示：

偏移量(字节)	格式	描述
0	0x01	对 MW 操作的标志
1 2 3	0x10 0x10 0x00	地址(低字节) 插入一个 0x10 字节 地址(高字节)
4	0x02	传送的字节数(MW10 为两个字节)
5 6 7	0x10 0x10 0x00	MW10 的低字节内容为 0x10 插入一个 0x10 字节 高字节内容为 0x00
8，9	0x10，0x03	结束标志
10	0x10	校 验 和， 0x01^0x10^0x10^0x00^0x02^0x10^0x10^0x00^0x10^0x03=0x10

5. 主方收到从方发送的数据后，发送返回命令ACK(0x06h)。

6. 从方接收到主方的返回命令ACK(0x06h)。

通信完成，从方把更改的MW的地址和内容传送给了主方，主方再更改MW的数据，使得从方和主方对应节点地址内容保持一致。

下面来做一个两台触摸屏之间用MemoryMap方式进行通信的例子。

首先在EasyBuilder8000中创建一个新的工程。

设置[编辑]/[系统参数]/[PLC设置]如下所示：

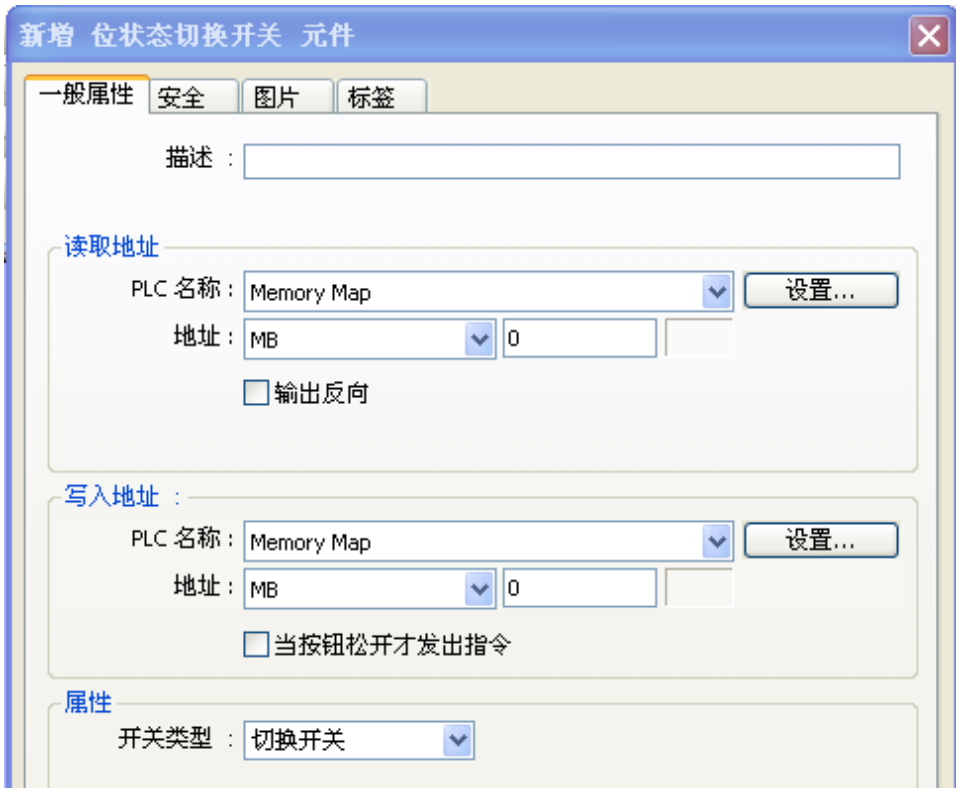
注意:

1.MT8000不像MT500，会区分MemoryMap_Master，MemoryMap_Slaver，而EB8000都选择 Memory Map即可。

2. [数据位]必须为8位。

3. 两台触摸屏的所有其他设置都必须一致。

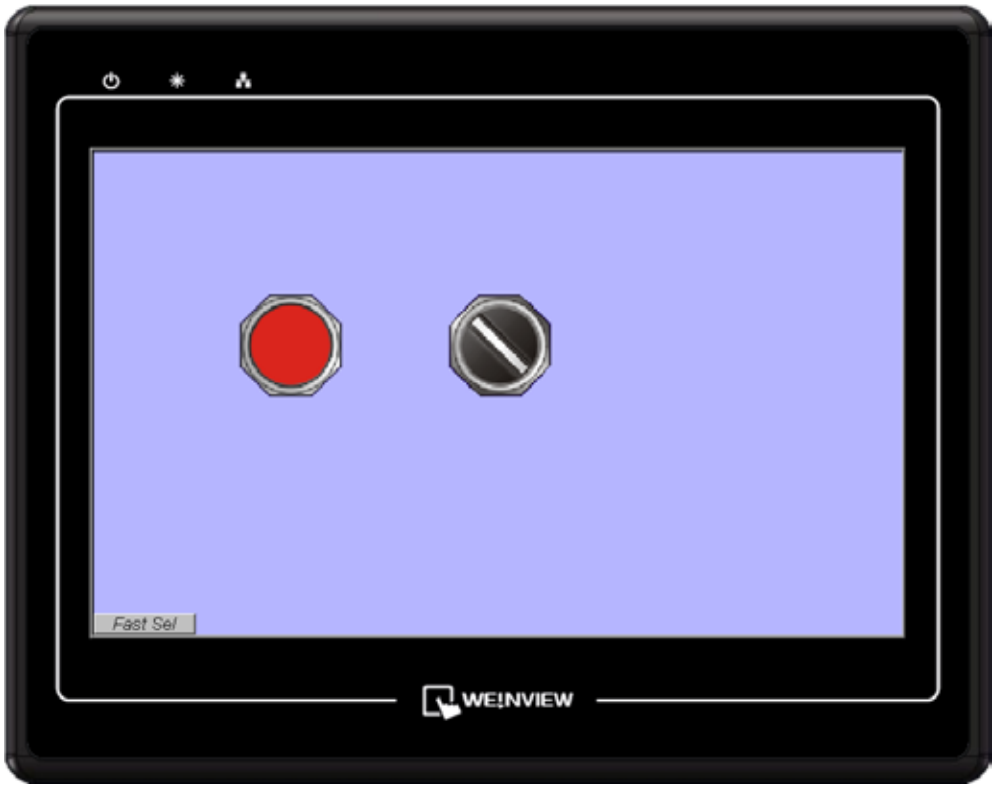
下面来往窗口10上添加2个元件，一个位状态切换开关其设置如下：



一个多状态切换开关设置如下：

[保存], [编译], [下载]。改变[系统参数]/[PLC设置]的参数，然后把该工程下载到另一台触摸屏。

最后触摸屏画面显示如下：



您可以试着触控一下任意一个按钮，对应另一台触摸屏的该按钮也将跟着动作，它们的状态始终保持一致。

一台触摸屏和任一台控制器之间的通信其方式与上述类似，其根本原理是2台设备的相同寄存器的数据要保持一致。

第三十二章 MT8000 ASCII通讯协议

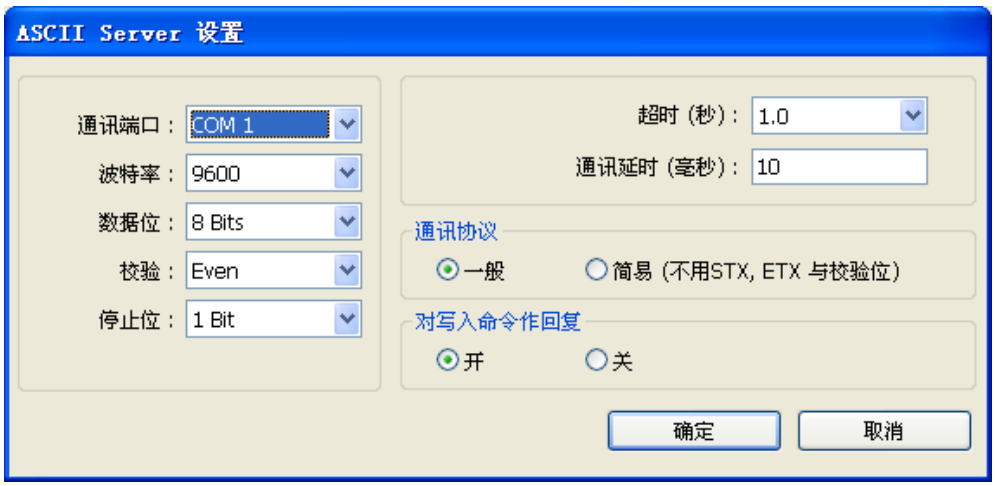
32.1 指令列表

ASCII主机使用下列指令来和MT8000进行通讯

指令	指令名称	描述
RD	Batch Read	读取连续的数据块
WD	Batch Write	写入连续的数据块
RR	Random Read	读取不连续的数据
RW	Random Write	写入不连续的数据
RC	Read Coil	读取位状态
WC	Write Coil	写入位状态

32.2 参数选项

参数设置如下图所示：



通讯协议：

一般：这个选项是使用不可打印的字节STX(02H), ETX(03H), ACK(06H)和NAK(15H)；还包含了2个位的校验和。

简易：有些设备（如运动控制器）是不能够产生非打印字节，或计算校验和。在这种模式下，形成的数据包定义如下，但不包括STX, ACK, ETX, NAK或校验和。在每个封包结尾都会加上0x0D，在MT8000的回应封包中，在最后也有一个0x0D。

对写入命令作回复：设定MT8000是否要对写入的命令回复。

开: 回复命令

关: 不回复命令

注意: 如果设定为关, 通讯延时参数将会失效。

32.3 网络功能

- 连线

  MT8000 ASCII通讯协议可以使用RS 485 2W, RS485 4W或RS232.

- 站号

  每一台使用ASCII通讯协议的MT8000都需要设定站号, 站号的范围是1到255

- 广播命令

  MT8000 ASCII不支持广播命令。

32.4 指令的使用

- RD批次读取

  主机发出命令

  这一个指令一次可以从触摸屏读取99个连续的LW寄存器。指令的长度固定为14个字节。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
1 Byte	2 Bytes	2 Bytes	4 Bytes	2 Bytes	1 Byte	2 Bytes
STX	站号	RD	地址编号	读取数量	ETX	校验和

Byte 1:开始符号, 固定为STX (0x02)

Bytes 2, 3: 要读取的HMI站号 (2 Bytes的 Hex digits)

Bytes 4, 5: 要执行的指令

Bytes 6–9: 读取的开始地址

Bytes 10, 11: 要读取的数量, 最大为99个字节

Byte 12: 结束符号 ETX (0x03)

Bytes 13, 14: 校验和是第2到12个字节的总和取低8位并转换成ASCII.

例如: 从站号为10(0AH)的触摸屏读取LW100开始的3个字, 即读取LW100~LW102。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
STX	0A	RD	0100	03	ETX	2E
02	30, 41	52, 44	30, 31, 30, 30	30, 33	03	32, 45

校验和(bytes 13,14)时第2到12个字节的总和取低8位并转换成ASCII。

30 + 41+ 52 + 44 + 30 + 31 + 30 + 30 + 30 + 33 + 03 = 22E.

取最低8位并转换成ASCII得到2E.

触摸屏回复命令

回复命令的长度是: L = (N * 4) + 8 其中N是读取的长度

如果该命令成功, HMI回复的长度至少为12个字节, 最长可达到404个字节。它是以STX开始, 再来是每4个字节为一组的资料, 然后是结束符ETX和校验和。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10-13	Bytes 14-17	Bytes 18 - (L-7)	Bytes (L-6) - (L-3)
STX	Station	CMD	Data 1	Data 2	Data 3	Data 4 - Data (N-1)	Data N

Byte L-2	Byte L-1, L
ETX	Checksum

上面的例子回复以下内容

地址	数据
100	75 (4BH)
101	8047 (1F6FH)
102	16,321 (3FC1H)

下面是触摸屏回复的命令:

STX	'0'	'A'	'R'	'D'	'0'	'0'	'4'	'B'	'1'	'F'	'6'	'F'	'3'	'F'	'C'	'1'
02H	30H	41H	52H	44H	30H	30H	34H	42H	31H	46H	36H	46H	33H	46H	43H	31H

ETX	'C'	'2'
03H	43H	32H

每个回复的数据是以16进制的格式, 校验和的计算是从bytes 2到(L-2).

如果是异常的情况则回复

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	'R', 'D'	Err Code

- WD批次写入

主机发出命令

这个指令可以最多往触摸屏写入99个连续的16位的LW寄存器, 封包的长度是:

L = (N * 4) + 14　　N是写入的长度

这个指令的封包长度最少是18个字节, 最长是410个字节

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Bytes 12-15	Bytes 16-19	Bytes 20 - (L-7)	Bytes (L-6) - (L-3)	Byte L-2	Byte L-1, L
STX	Station	WD	Addr.	No. of Items	Data 1	Data 2	Data 3 - Data (N-1)	Data N	ETX	Check-sum

Byte 1: 开始符, 固定为STX (0x02)

Bytes 2, 3: 要读取的触摸屏的站号 (2 Bytes的 Hex digits)

Bytes 4, 5: 要执行的命令

Bytes 6–9: 写入的起始地址的编号, 必须要4个Bytes

Bytes 10, 11:写入的数据数量, 长度为2个Bytes.

Bytes 12 – (L–3): 写入的数据, 最多为99个数据, 每4个字节为一组数据

Byte (L–2): 结束符 ETX (0x03).

Bytes L–1, L: 校验和

例如：触摸屏写入3个字到站号为17（11H）触摸屏的LW201，这个将写入数据到LW201～LW203。

LW201 = 101 (0x65)

LW202 = 575 (0x23F)

LW203 = 1049 (0x419)

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Bytes 12-15	Bytes 16-19	Bytes 20-23	Byte 24	Bytes 25, 26
STX	11	WD	0201	03	0065	023F	0419	ETX	9A
02	31, 31	57, 44	30, 32, 30, 31	30, 33	30, 30, 36, 35	30, 32, 33, 46	30, 34, 31, 39	03	39, 41

校验和(bytes 25, 26)是第2到24字节的总和取低8位并转换成ASCII.

31+ 31 + 57 + 44 + 30 + 32 + 30 + 31 + 30 + 33 + 30 + 30 + 36 + 35 + 30 + 32 + 33+ 46 + 30 + 34+ 31+ 39 + 03 = 49A.

取低8位并转换成ASCII得到9A。

触摸屏回复命令

如果该命令成功, 触摸屏回复:

Byte 1	Byte 2, 3	Byte 4, 5
ACK	Station	'W', 'D'

如果是异常情况, 触摸屏则回复:

Byte 1	Byte 2, 3	Byte 4, 5	Byte 6
NAK	Station	'W', 'D'	Err Code

● RR随机读取

主机发出命令

这个指令一次可以从触摸屏读取99个不连续的LW寄存器, 指令长度为:

L = (N * 4) + 8　　N是读取的数量

命令的长度最少是12个字节, 最长是402个字节。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10-13	Bytes 14 - (L-7)	Bytes (L-6) - (L-3)	Byte L-2	Byte L-1, L
STX	Station	RR	Addr 1	Addr 2	Addr 3 - Addr (N-1)	Addr N	ETX	Check-sum

Byte 1: 开始符, 固定为STX (0x02)

Bytes 2, 3: 要读取的触摸屏的站号 (2 Bytes的 Hex digits)

Bytes 4, 5: 要执行的命令

Bytes 6–9: 第一个要读取的数据的地址., 必须要4个Bytes,

Bytes 10–13: 第二个要读取的数据的地址, 必须要4个Bytes,

Bytes 14 – (L–7): 其他要读取的数据的地址, 必须要4个Bytes,

Byte (L–2): 结束符 ETX (0x03).

Bytes L–1, L:校验和, 是第2到L–2个字节的总和取低8位并转换成ASCII。

触摸屏回复的命令

回复的长度是: L= (N * 4) + 8N是读取的长度。

如果该命令成功, 触摸屏回复的长度至少为12个字节, 最长可达406个字节, 它是以STX开始, 再来是每4个字节为一组数据, 然后是结束符ETX和校验和。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6–9	Bytes 10–13	Bytes 14–17	Bytes 15 – (L–7)	Bytes (L–6) – (L–3)
STX	Station	Cmd	Data 1	Data 2	Data 3	Data 4 – Data (N–1)	Data N

Byte L–2	Byte L–1, L
ETX	Checksum

回复的数据是16进制的格式,其中校验和是第2到L–2个字节的总和取其低8位并转换成ASCII。

如果时异常的状况则回复命令:

Byte 1	Byte 2, 3	Byte 4, 5	Byte 6
NAK	Station	'R', 'R'	Err Code

● RW随机写入

主机发出命令

这个指令最多可以向触摸屏写入99个不连续的16bit的LW寄存器。封包长度是:

L = (N * 8) + 8 其中N是写入的长度。

这个指令的封包长度最少是16个字节, 最长是800个字节。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6–9	Bytes 10–13	Bytes 14–17	Bytes 18–21	…
STX	Station	RW	Addr 1	Data 1	Addr 2	Data 2	…

Bytes (L–10) – (L–7)	Bytes (L–6) – (L–3)	Byte L–2	Byte L–1, L
Addr N	Data N	ETX	Check–sum

Byte 1: 开始符, 固定为STX (0x02)

Bytes 2, 3: 要读取的触摸屏的站号 (2 Bytes的Hex digits)

Bytes 4, 5: 要执行的命令

Bytes 6–9: 写入的地址编号, 必须要4个Bytes

Bytes 10–13 : 写入的数据, 4个字节为一组数据.

Bytes 14 – (L–3): 写入的地址编号和资料

Byte (L–2): 结束符, ETX (0x03).

Bytes L–1, L: 校验和

触摸屏回复命令

如果该命令成功, 触摸屏回复如下命令:

Byte 1	Byte 2,3	Byte 4,5
ACK	Station	'R', 'W'

如果是异常的状况, 则回复以下命令

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	'R', 'W'	Err Code

● RC读取触点

主机发出命令

这个指令一次可以从触摸屏读取99个连续的LB寄存器, 指令的长度固定为14Bytes。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
1 Byte	2 Bytes	2 Bytes	4 Bytes	2 Bytes	1 Byte	2 Bytes
STX	Station	RC	Addr.	No. of Items	ETX	Checksum

Byte 1: 开始符, 固定为STX (0x02)

Bytes 2, 3: 要读取的触摸屏站号 (2 Bytes的 Hex digits)

Bytes 4, 5: 要执行的命令

Bytes 6–9: 读取的起始位编号

Bytes 10, 11: 要读取的数量, 最多为99位.

Byte 12: 结束符, ETX (0x03)

Bytes 13, 14: 校验和是第2到12个低字节的总和取低8位并转换成ASCII

例如: 从7(07H)站号的触摸屏读取LB100开始的12个位, 即读取LB100~LB111。

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
STX	07	RC	0100	02	ETX	22
02	30, 37	52, 43	30, 31, 30, 30	30, 32	03	32, 32

校验和(bytes 13、14)是第2到12个字节的总和取低8位并转换成ASCII。

30 + 37 + 52 + 43 + 30 + 31 + 30 + 30 + 30 + 32 + 03 = 222.

取低8位并转换成ASCII得到22。

触摸屏回复命令

回复命令的长度是: L = N + 8 其中N时读取的长度。

如果该命令成功, 触摸屏回复的长度至少是9个字节, 最长可达107个字节, 它是以STX开始, 再来是4个字节为一组数据, 然后是结束符ETX和校验和。

Byte 1	Bytes 2, 3	Bytes 4, 5	Byte 2	Byte 3	Byte 4	Bytes 5 - (L-4)
STX	Station	RC	Data 1	Data 2	Data 3	Data 4 - Data (N-1)

Byte (L-3)	Byte L-2	Byte L-1, L
Data N	ETX	Checksum

在触摸屏里的数据是:

LB100	101	102	103	104	105	106	107	108	109	110	111
0	0	1	0	1	0	1	1	0	0	0	1

以下是触摸屏发送的数据:

STX	'0'	'7'	'R'	'C'	'0'	'0'	'1'	'0'	'1'	'0'	'1'	'1'	'0'	'0'	'0'
02H	30H	37H	52H	43H	31H	30H	31H	31H	31H	30H	31H	31H	30H	30H	30H

‘1’	ETX	‘4’	‘6’

如果是异常的状况，则回复以下命令：

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	‘R’, ‘C’	Err Code

● WC写入触点

主机发出命令

这个命令最多可以向触摸屏写入99个连续的LB寄存器，封包的长度是：

L = N + 14　其中N是写入的长度。

这个指令的封包长度最少是15个字节，最长是113个字节。

Byte 1	Bytes 2,3	Bytes 4, 5	Bytes 6-9	Bytes 10-11	Byte 12	Byte 13	Bytes 14 - (L-4)
STX	Station	WC	Addr.	No. of Items	Data 1	Data 2	Data 3 - Data (N-1)

Byte (L-3)	Byte L-2	Byte L-1, L
Data N	ETX	Check-sum

Byte 1: 开始符，固定为STX (0x02)

Bytes 2, 3: 要读取的触摸屏的站号 (2 Bytes的 Hex digits)

Bytes 4, 5: 要执行的命令

Bytes 6–9: 写入的起始地址的编号，必须要4个Bytes

Bytes 10, 11: 写入的数量，最长为2个Bytes.

Bytes 12 – (L–3): 写入的数据，最多为99个，每一个字节为一个位的数据。

Byte (L–2): 结束符，ETX (0x03).

Bytes L–1, L: 校验和

例如：触摸屏写入5个位到12（0CH）号站的LB214，这将写入数据到LB214～LB218。

写入的数据为：

LB214	215	216	217	218
1	1	0	0	1

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16	Byte 17	Bytes 18, 19
STX	0C	WC	0214	05	1	1	0	0	1	ETX	2F
02	30, 43	57, 43	30, 32, 31, 34	30, 35	31	31	30	30	31	03	32, 46

校验和(bytes 18.、19)是第2个到17个字节的总和取低8位并转换成ASCII。

30 + 43+ 57 + 43 + 30 + 32 + 31 + 34 + 30 + 35 + 31 + 31 + 30 + 30 + 31 + 03 = 32F.

取低8为并转换成ASCII得到2F。

触摸屏回复命令

如果该命令成功，触摸屏则回复：

Byte 1	Byte 2, 3	Byte 4, 5
ACK	Station	'W', 'C'

如果是异常的状况，触摸屏则回复：

Byte 1	Byte 2, 3	Byte 4, 5	Byte 6
NAK	Station	'W', 'C'	Err Code

● 错误代码

下表列出了错误状况以及相对应的错误代码：

代码	叙述
06H	校验和错误
10H	未定义的命令
11H	数据长度错误－数据长度超过接受的寄存器
12H	通讯内容异常－缺少 ETX
7AH	无效的地址
7BH	读取超过 99 个数据

第三十三章 EasyDiagnoser

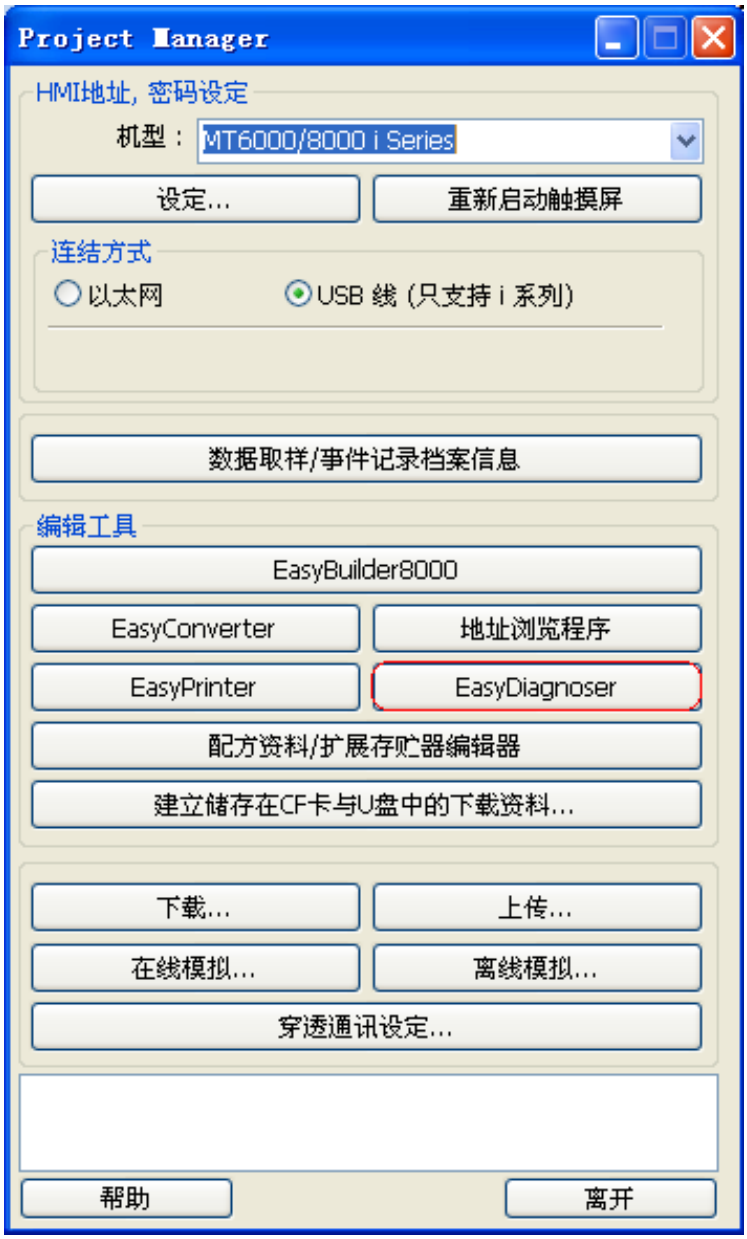
33.1 简介与设定方式

简介：

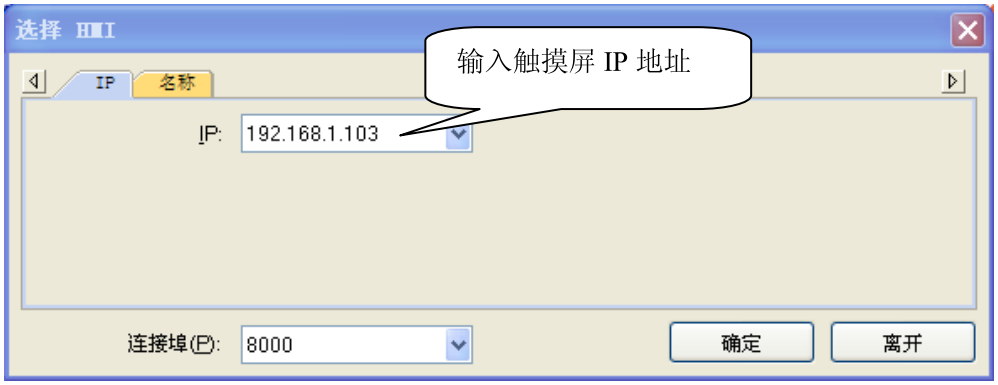
Easydiagnoser是一种可以用来找出触摸屏与PLC之间通讯出错的工具。

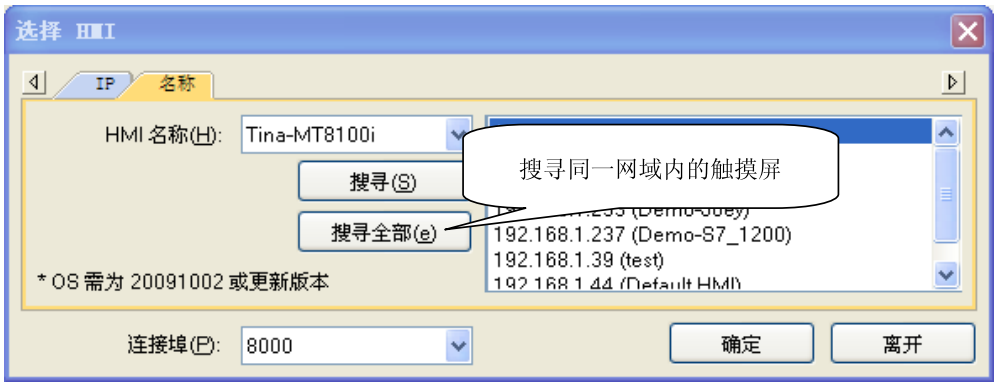
设定方式：

步骤一：开启Project Manager并点击EasyDiagnoser

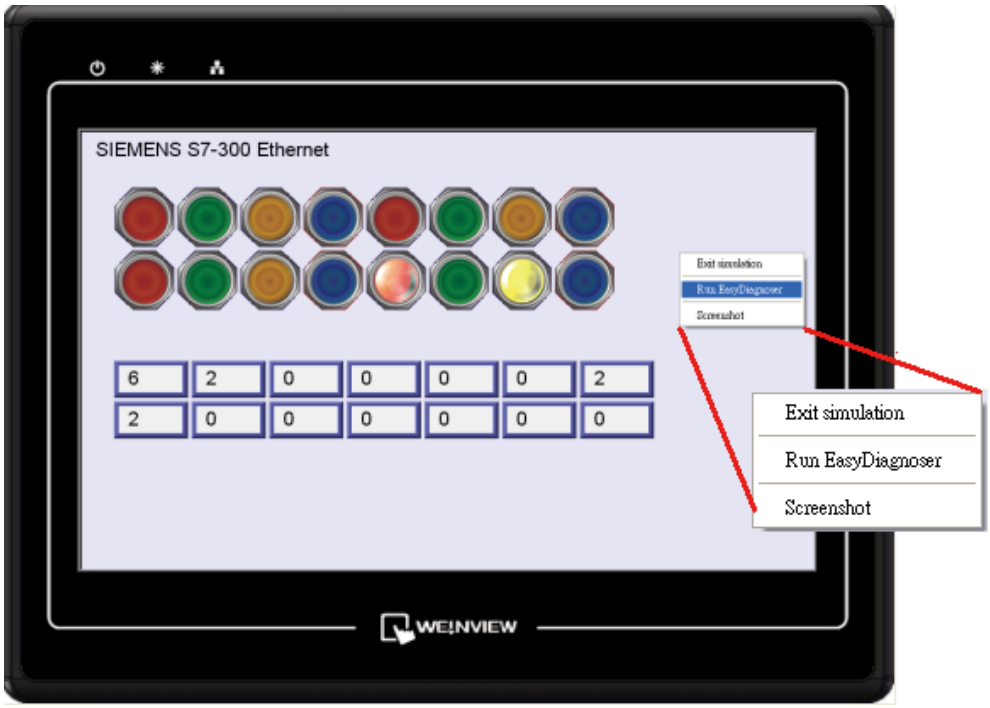


步骤二：设定欲进行通讯的触摸屏的IP地址，可选择自行输入IP地址或使用“搜索全部”功能，并输入“连接端口”

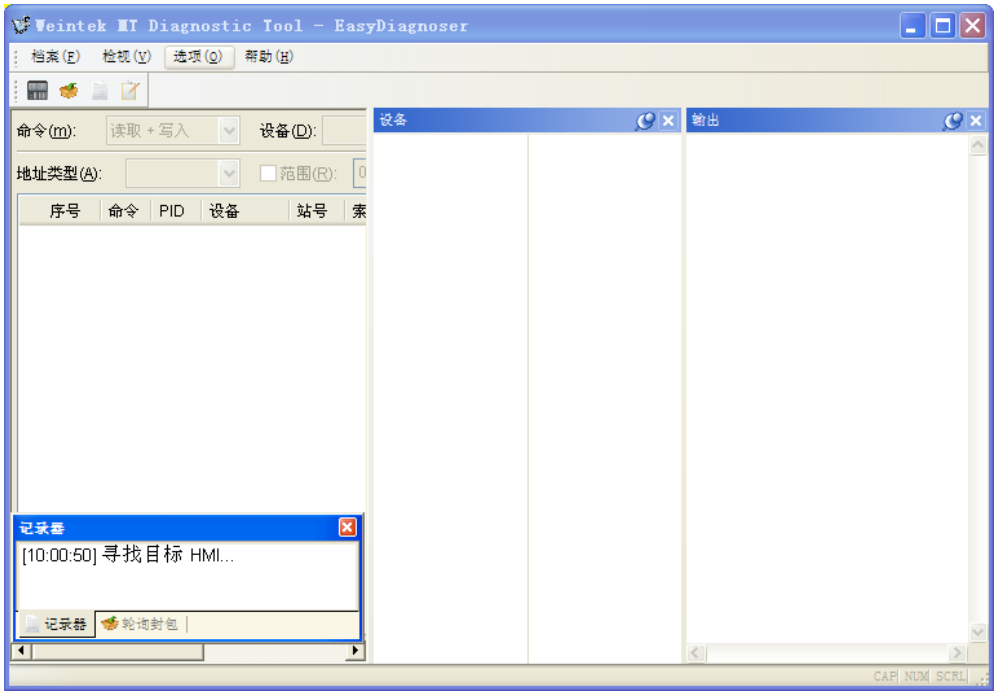




另外还提供在EB8000执行在线模拟时，触控右键可直接选择“Run EasyDiagnoser”，进入EasyDiagnoser窗口。



完成以上设定之后，触控OK，EasyDiagnoser操作画面如下图：

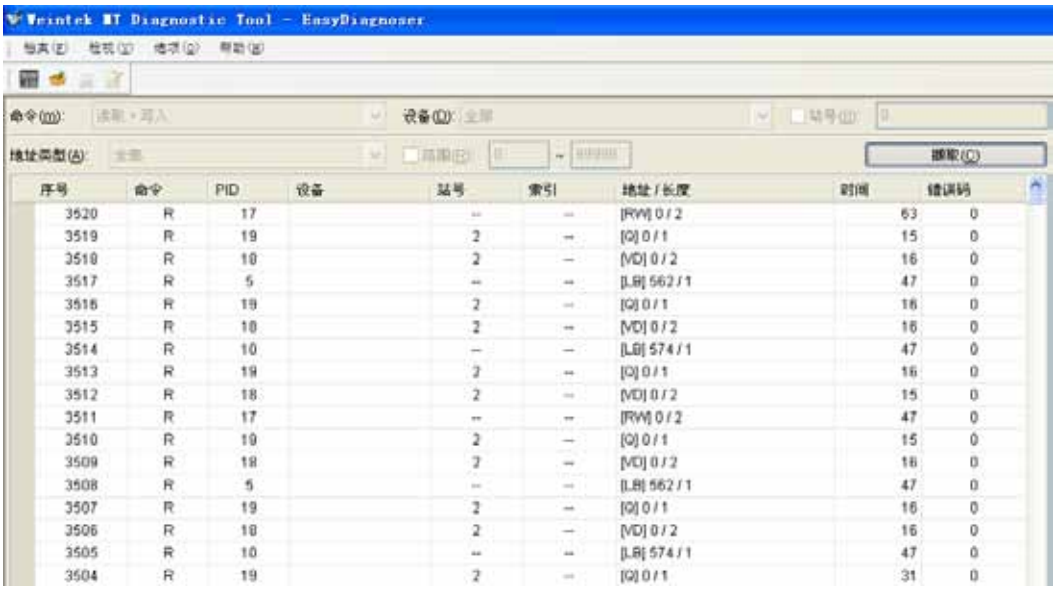


33.2 EasyDiagnoser设定

工具栏	叙述
档案	**另存新档** 可将截取下来的通讯资料，储存成 xls，并使用 EXCEL 打开 
	离开 离开当前档案
检视 	点击[设备列表]可显示设备列表窗口 点击[封包列表]可显示封包列表窗口 点击[讯息视窗]可显示讯息窗口 点击[输出视窗]可显示输出窗口
选项 	**工具列** 显示[设备列表][封包列表][讯息视窗][输出视窗]的工具列 
	状态列 在 EasyDiagnoser 视窗的最底部，显示 CAP， NUM 或 SCRL 的信息 
	更新封包列表 显示目前触摸屏窗口的封包
	显示元件 ID 显示触摸屏上元件的 ID 号 
	清除通讯记录 清除所有通讯中所记录的讯息
帮助	显示 EasyDiagnoser 版本讯息

● 通讯记录区

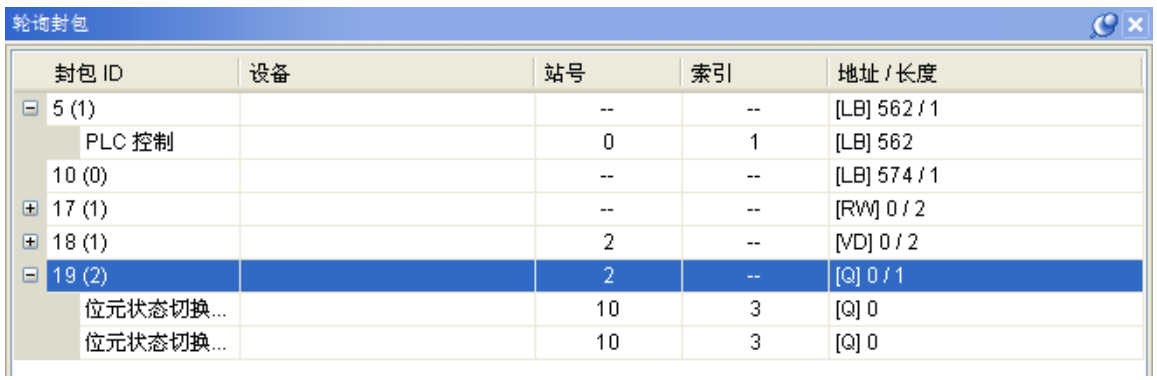
在通讯记录区，用户可以观察触摸屏和PLC之间的通讯。



项目	叙述
命令	a. 读取+写入 显示读和写的命令在通讯记录区
	b. 读取 只显示读的命令在通讯记录区
	c. 写入 只显示写的命令在通讯记录区
设备	a. 全部 显示本地触摸屏和 PLC 的讯息 如果设定[命令：读取+写入]，在通讯记录区会显示本地触摸屏和 PLC 的读和写的讯息。 如果设定[命令：读取]，在通讯记录区会显示本地触摸屏和 PLC 的读的讯息。 如果设定[命令：写入]，在通讯记录区会显示本地触摸屏和 PLC 的写的讯息。
	b. Local HMI 显示本地触摸屏的讯息 如果设定[命令：读取+写入]，在通讯记录区会显示本地触摸屏的读和写的讯息。 如果设定[命令：读取]，在通讯记录区会显示本地触摸屏的读的讯息。 如果设定[命令：写入]，在通讯记录区会显示本地触摸屏的写的讯息。
	c. PLC 显示 PLC 的讯息 如果设定[命令：读取+写入]，在通讯记录区会显示 PLC 的读和写的讯息。 如果设定[命令：读取]，在通讯记录区会显示 PLC 的读的讯息。 如果设定[命令：写入]，在通讯记录区会显示 PLC 的写的讯息。
站号	选择想显示的 PLC 的站号(当选择“全部”时无法使用此功能)
地址类型	用户可以选择全部或是其中的设备地址类型显示在屏幕上(当选择“全部”时无法使用此功能)
范围	设定要截取的地址范围(当选择“全部”时无法使用此功能)
擷取	点击“擷取按钮”开始或停止擷取通讯信息
错误码	请参考本附件的最后一节

● 轮询封包

项目	叙述
封包 ID	封包的 ID 号 可由通讯记录区看出哪一个封包 ID 的元件有问题
设备	显示触摸屏和 PLC 型号
站号	显示 PLC 站号
索引	显示元件所使用的索引寄存器编号
地址/长度	显示设备类型地址及封包内的字长度



项目	**叙述**
元件	封包 ID 内的元件
窗口	元件在程序中所在的窗口
ID 号	元件的 ID 号码
地址	元件地址

注意:

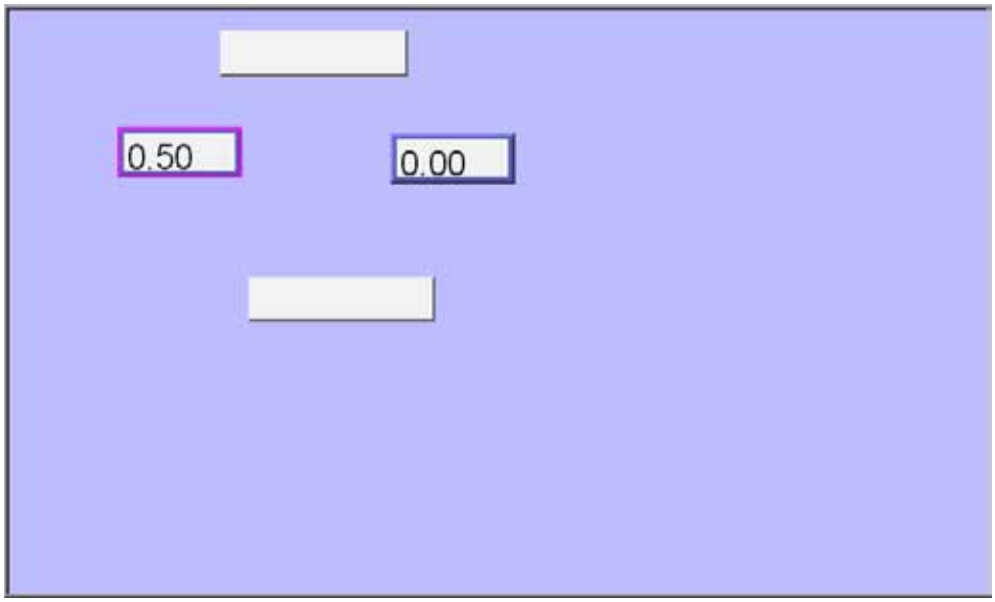
a. 点击封包ID后, 第三栏会显示设备的站号

封包 ID	设备	站号	索引	地址 / 长度
5 (1)		--	--	[LB] 562 / 1
PLC 控制		0	1	[LB] 562
10 (0)		--	--	[LB] 574 / 1
17 (1)		--	--	[RW] 0 / 2
18 (1)		2	--	[VD] 0 / 2
19 (2)		2	--	[Q] 0 / 1
位元状态切换...		10	3	[Q] 0
位元状态切换...		10	3	[Q] 0

b. 双击封包ID之后选择元件，可显示元件所在位置

例如：选择数值输入，同时窗口显示10，表示此元件在程序的第10个窗口，同时此元件在触摸屏上被粉红色的框框标示出来，如下图：

元件		视窗	ID	地址
5 (1)		--	--	[LB] 562 / 1
10 (0)		--	--	[LB] 574 / 1
17 (1)		--	--	[RW] 0 / 2
数值输入		10	0	[RW] 0
18 (1)		2	--	[VD] 0 / 2
19 (2)		2	--	[Q] 0 / 1



● 设备

设备窗口显示触摸屏及PLC的相关信息

- 输出

  搭配使用Macro所提供的Trace函数，能够侦测到Macro执行的状态，请参考使用手册《第十八章 宏指令(macro)使用说明》，有详细的说明。

33.3 错误代码

在通讯记录区可从错误代码找出错误原因，请参考下列错误代码：

0 : Normal

1 : Time out

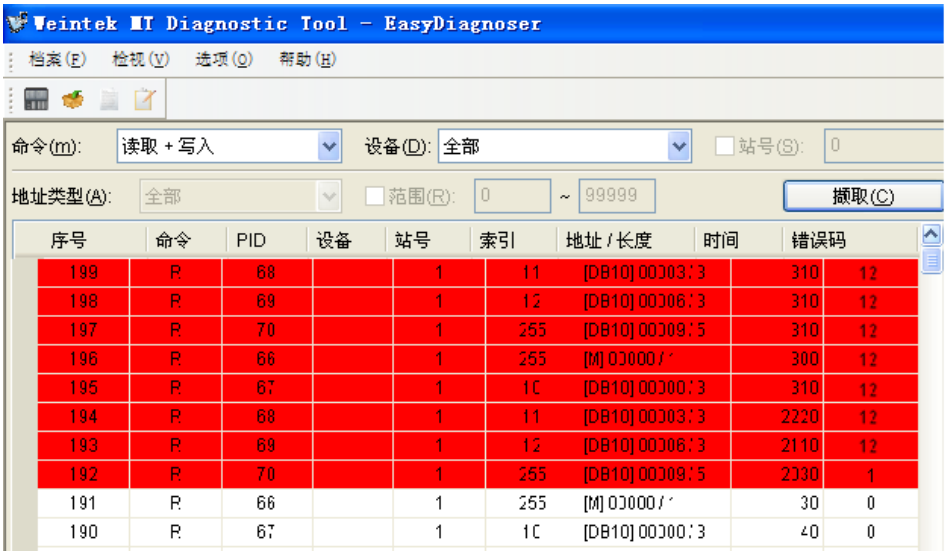
2 : Fail Error

12 : Ignore

当错误发生时，错误的信息会变成红色，如下图所示：

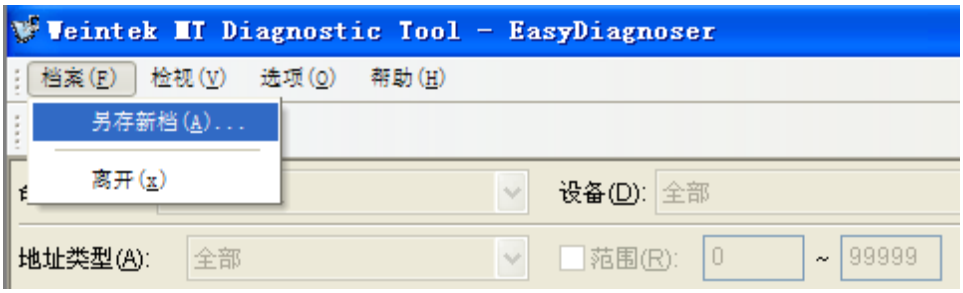
错误代码1是由于PLC与触摸屏之间的通讯中断

错误代码12，表示出现 “PLC No response” 信息窗口



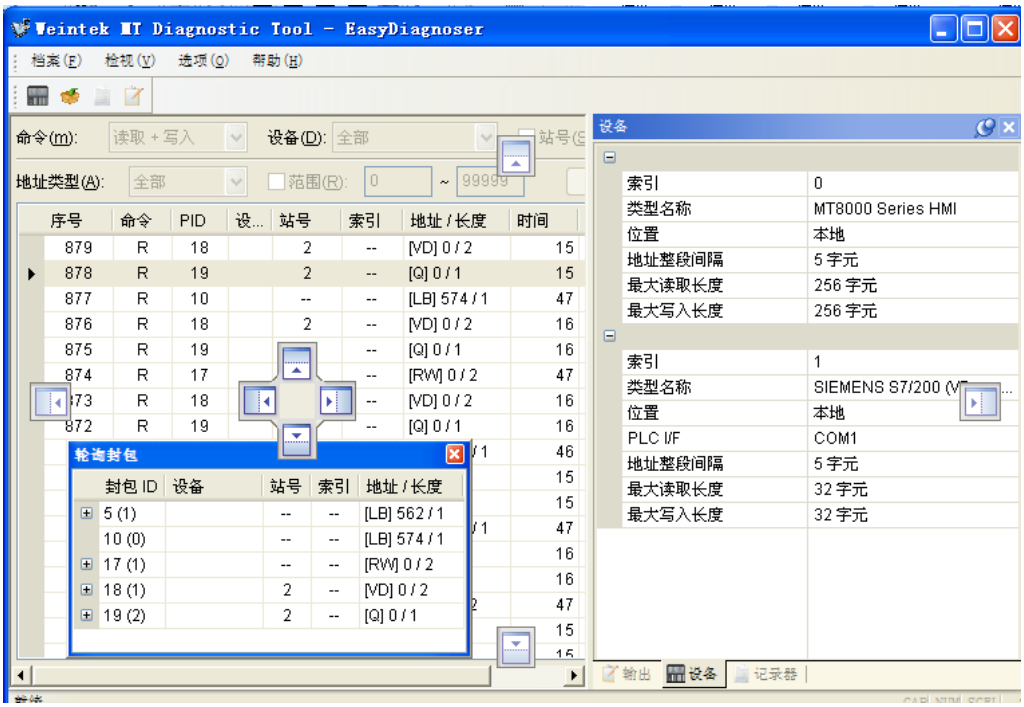
33.4 另存新档

可将所攫取的文件另存为*.xls格式，并且可以使用Excel打开文件浏览。



33.5 窗口调整

用户可以使用拖动功能，通过显示在编辑画面上的多个定点图标，来放置窗口到合适的位置。



注意：

EasyDiagnoser 不支持使用Siemens S7/1200 (Ethernet) and Allen-Bradley Ethernet/IP (CompactLogix/ControlLogix) – Free Tag Names此两款使用tag的PLC。

第三十四章 AB EtherNet/IP Free Tag Names

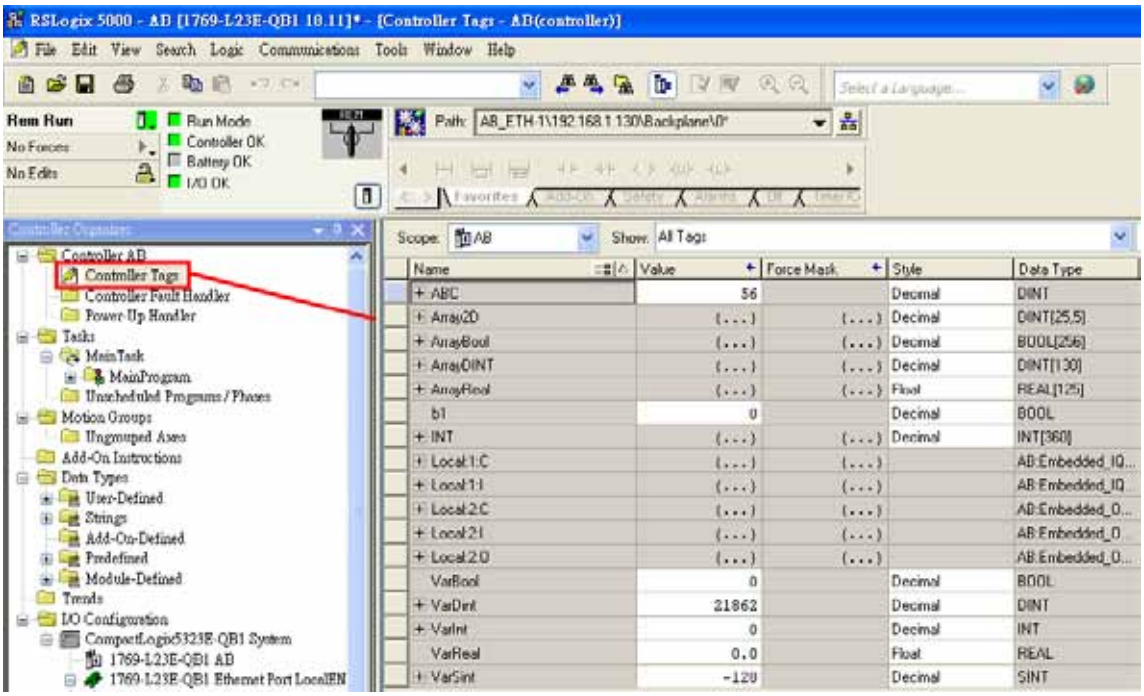
EB8000在使用Allen-Bradley EtherNet/IP-Tag (CompactLogix/ControlLogix)的驱动时可以从RSLogix5000的CSV文件导入Tag, 可是User-Defined, Predefined和Module-Define的结构并不会被导入。

	A	B	C	D	E	F	
7	TYPE	SCOPE	NAME	DESCRIPT	DATATYPE	SPECIFIER	ATTRIBUTES
8	TAG		Local:1:C		AB:Embedded_IQ16F:C:0		
9	TAG		Local:1:I		AB:Embedded_IQ16F:I:0		
10	TAG		Local:2:C		AB:Embedded_OB16:C:0		
11	TAG		Local:2:I		AB:Embedded_OB16:I:0		
12	TAG		Local:2:O		AB:Embedded_OB16:O:0		
13	TAG		Array2D		DINT[25,5]		(RADIX := Decimal, Cons
14	TAG		ArrayBool		BOOL[256]		(RADIX := Decimal, Cons
15	TAG		ArrayDINT		DINT[130]		(RADIX := Decimal, Cons
16	TAG		ArrayReal		REAL[125]		(RADIX := Float, Constant
17	TAG		B001		INT[15]		(RADIX := Decimal, PLCI
18	TAG		b003		INT[255]		(RADIX := Decimal, PLCI
19	TAG		b1		BOOL		(RADIX := Decimal, Cons

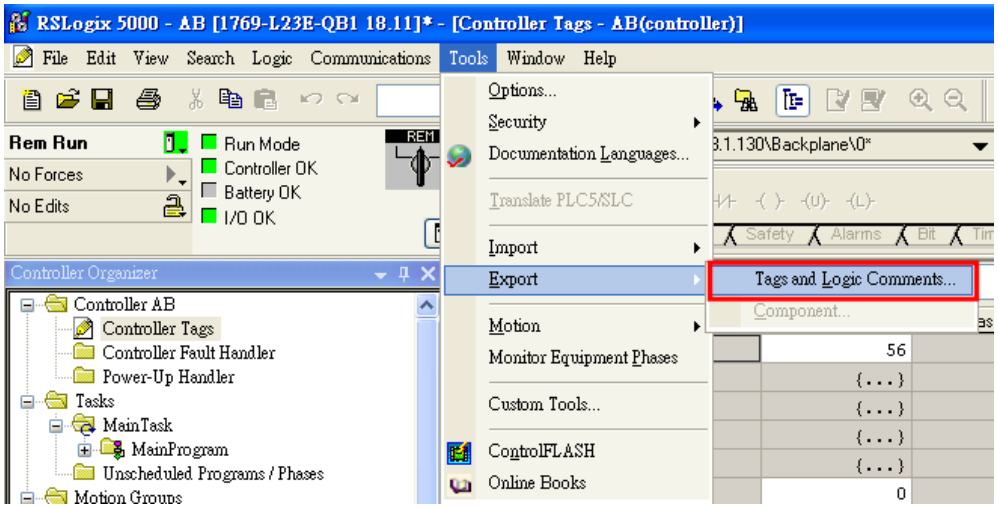
所以要用EB8000的AB Data Type Editor来输入和编辑User-Defined, Predefined和Module-Defined。

34.1 导入用户自定义的AB Tag CSV文件至EB8000

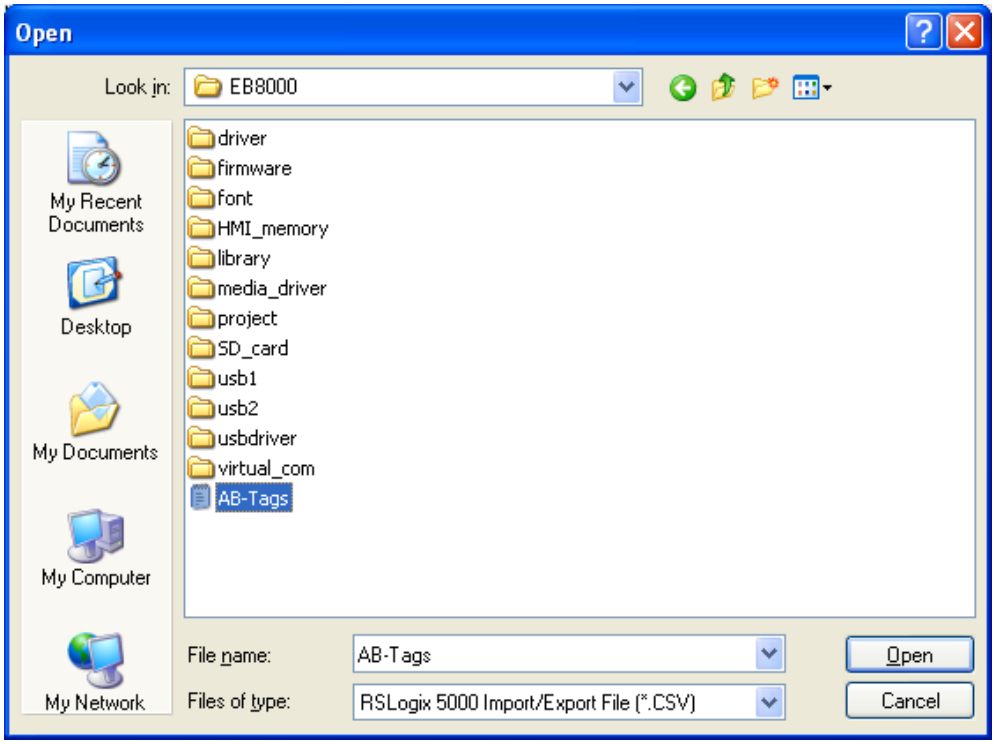
步骤1: 在RSLogix5000建立Tags。



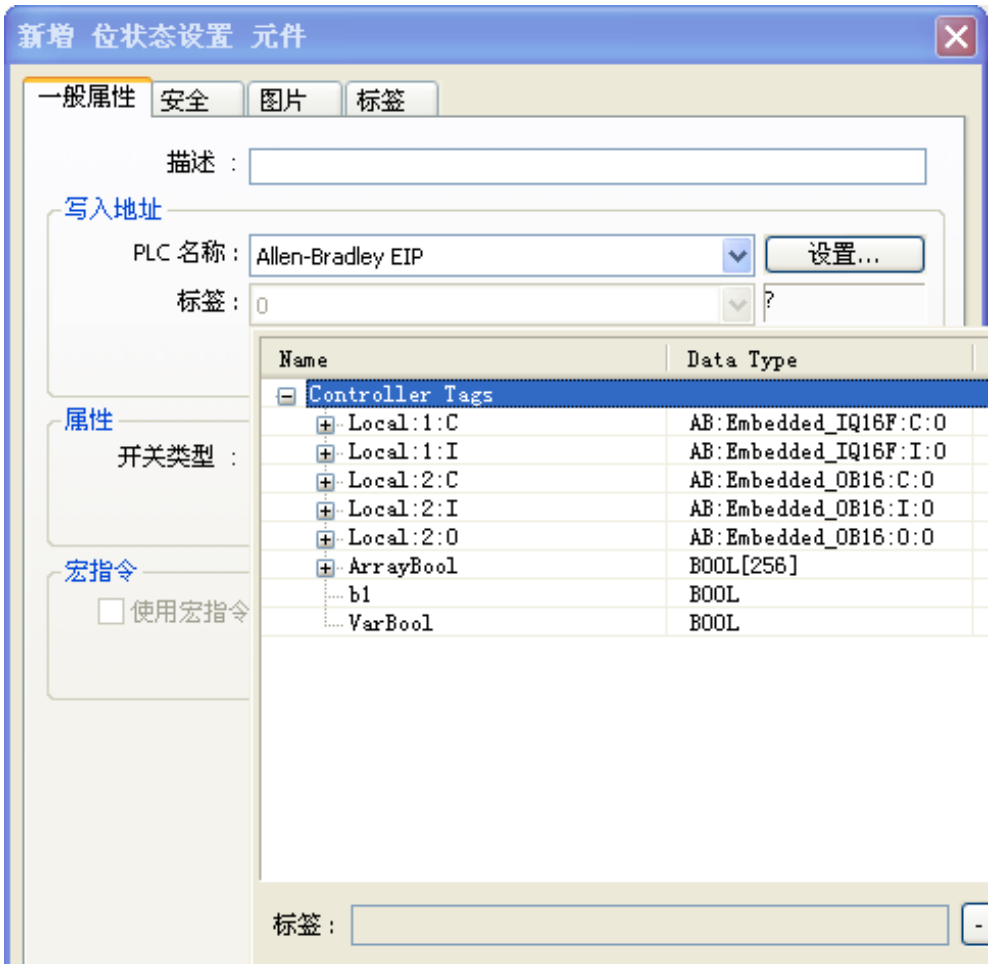
步骤2：导出Tags成CSV文档：



步骤3：在EB8000建立Allen-Bradley EtherNet/IP-Tag (CompactLogix/ControlLogix)驱动，输入PLC的IP地址并点击“导入标签”

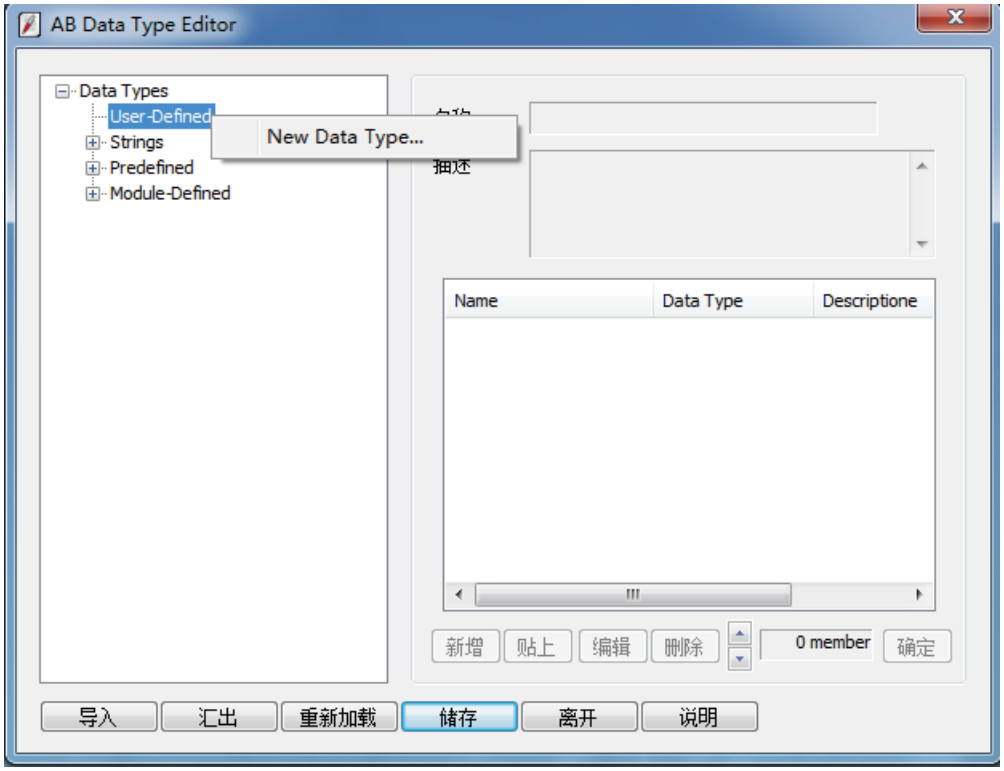


步骤4：在元件对话框中选择PLC，并选择controller tag即可：



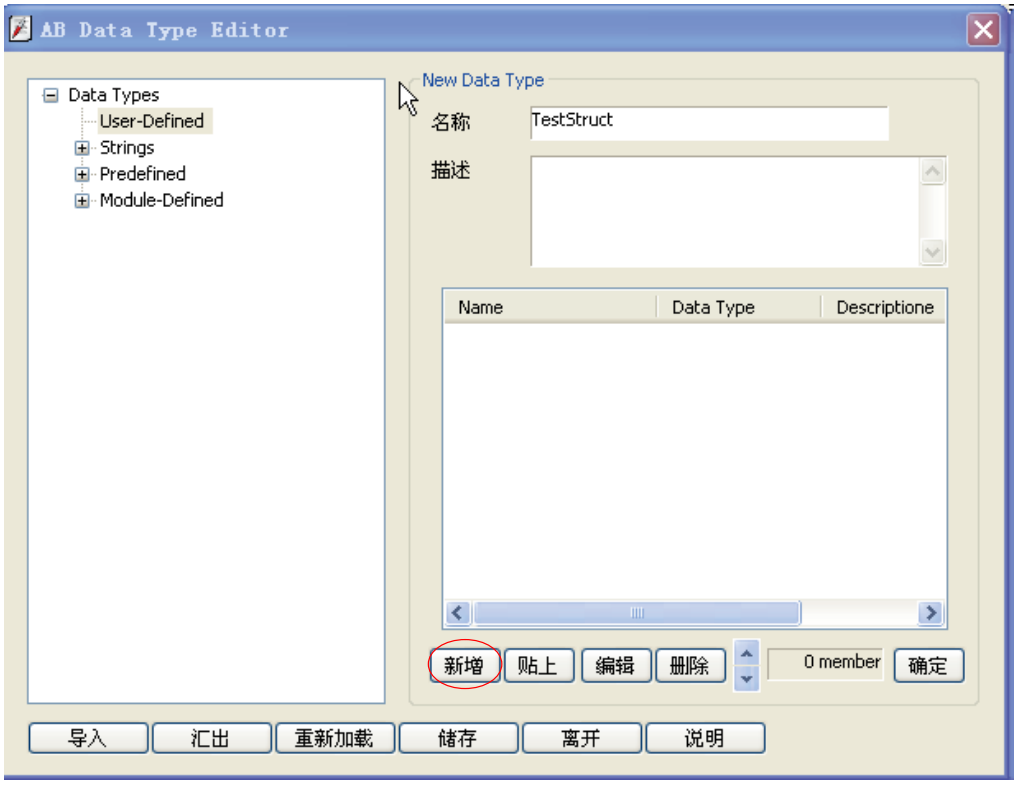
34.2 新增资料格式

步骤1: 在所属的类型上点击鼠标右键[通常为User-Defined], 点击选单中的New Data Type即可开始编辑。

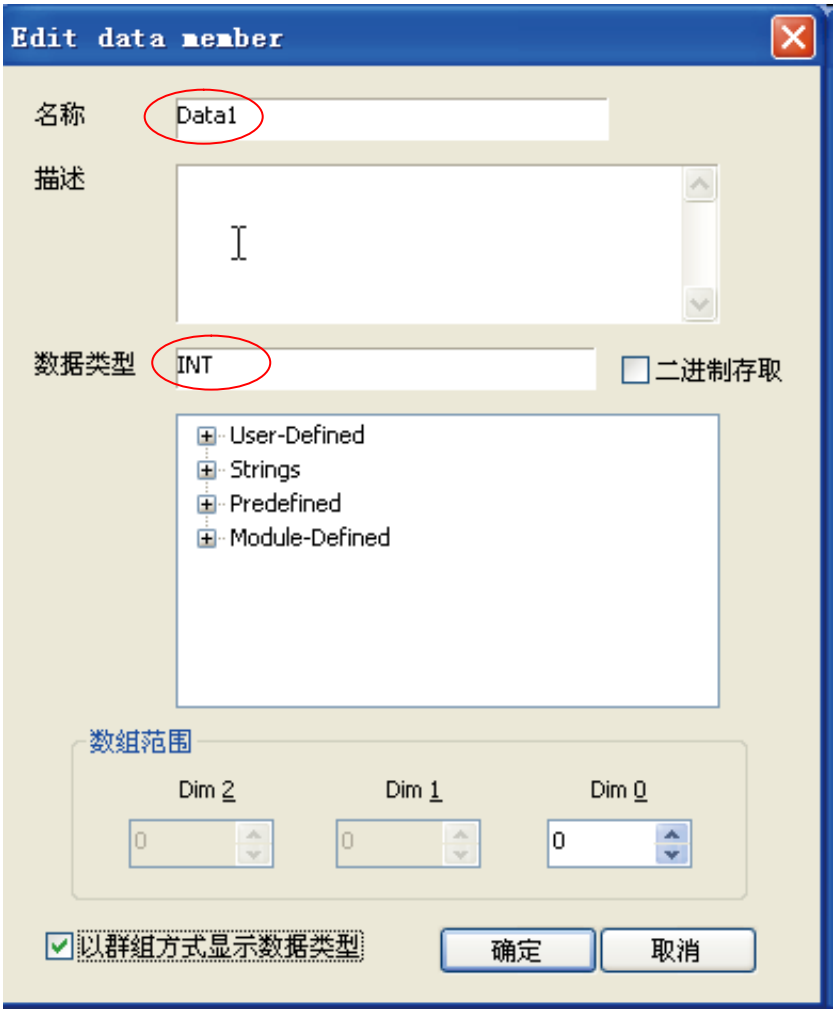


步骤2: 输入格式名称后, Description可略过。

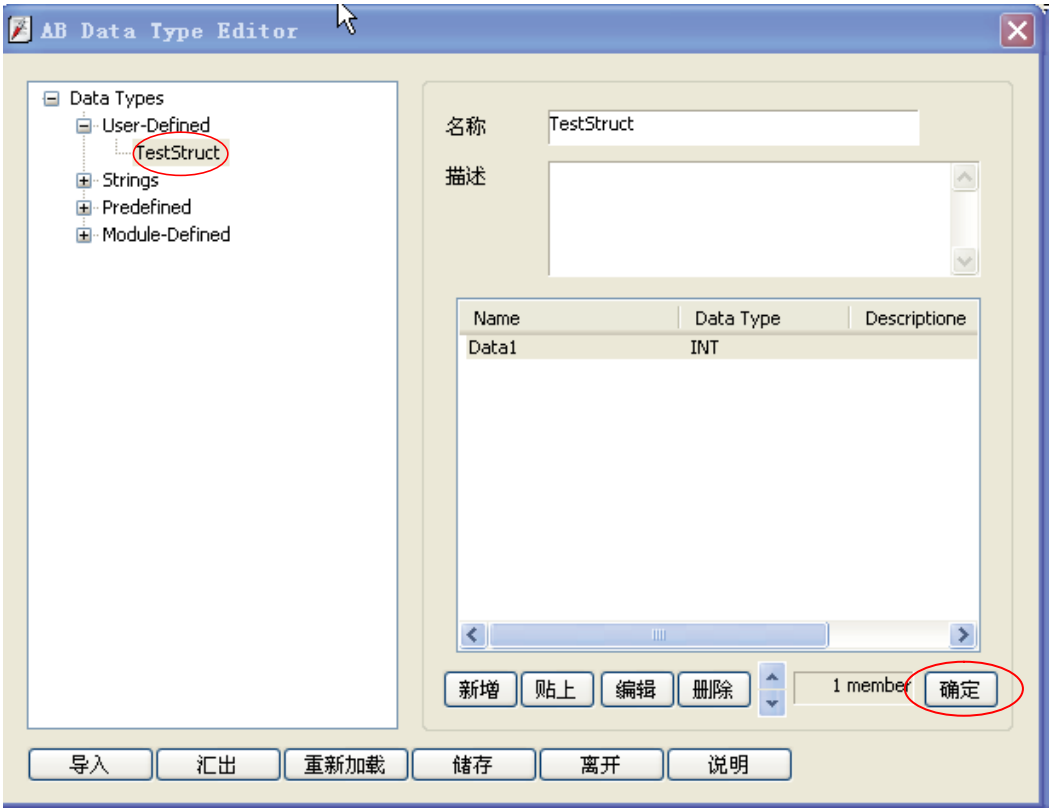
新增数据, 点击Add按钮。



步骤3：输入数据的名称与类型后触控OK离开。



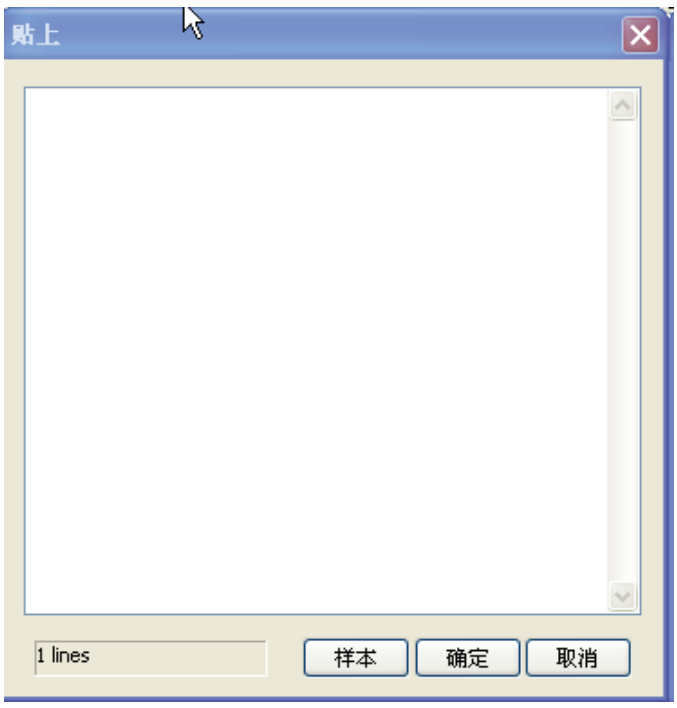
步骤4：增加完所有的数据后，触控OK键，此时左边的类别列表即可出现刚建立的类别。



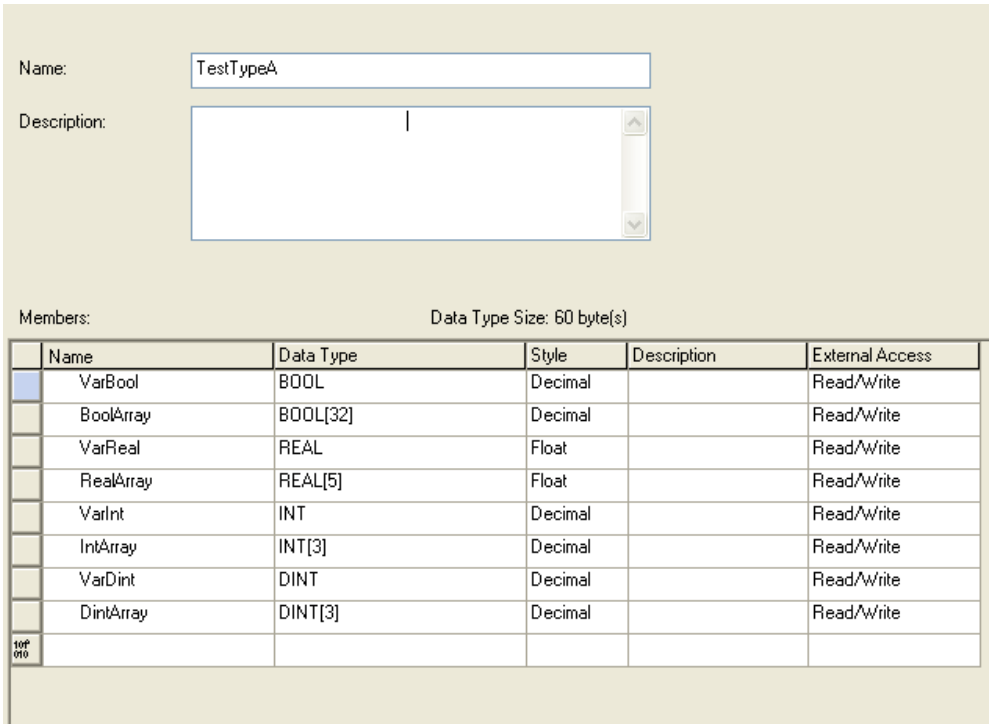
修改资料类型名称与描述后需触控OK才进行更改。

34.3 Paste功能

步骤1: 在新增资料时, 使用此功能可一次新增多笔资料; 方法为在主画面触控Paste按钮:

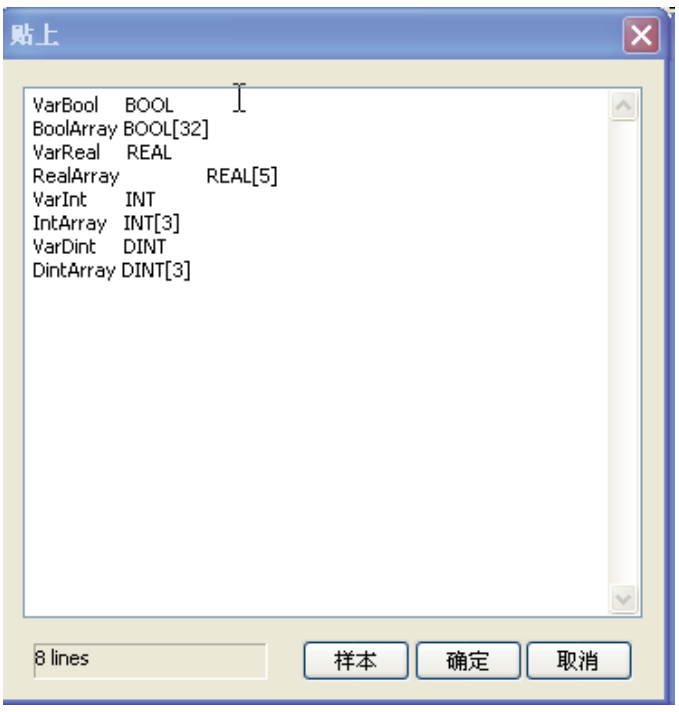


步骤2：编辑方法每行先输入资料名称后接一个空格或tab，可按sample按钮参考；建立从RSLogix 5000中直接复制粘贴以免输入错误。

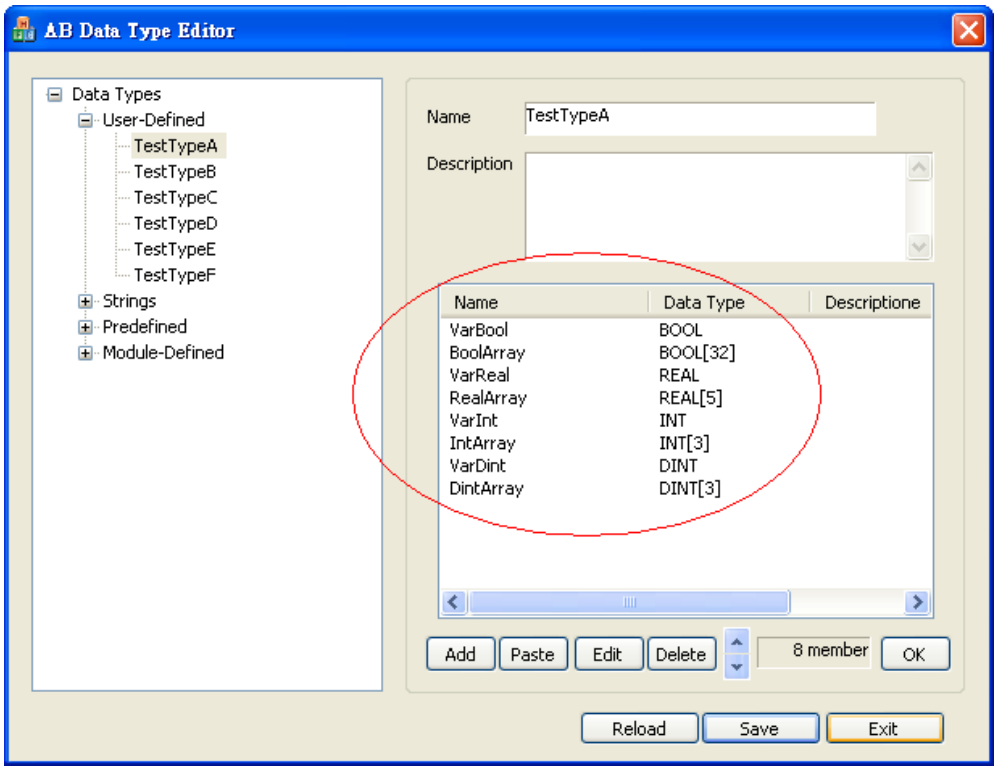


Name	Data Type	Style	Description	External Access
VarBool	BOOL	Decimal		Read/Write
BoolArray	BOOL[32]	Decimal		Read/Write
VarReal	REAL	Float		Read/Write
RealArray	REAL[5]	Float		Read/Write
VarInt	INT	Decimal		Read/Write
IntArray	INT[3]	Decimal		Read/Write
VarDint	DINT	Decimal		Read/Write
DintArray	DINT[3]	Decimal		Read/Write

步骤3：上图为一个在RSLogix中自定义的类型，使用鼠标将Name与DataType选取起来，可从第一项选择并压住鼠标移至底部直到滚动条滑到底部后放开，如此便可全部选取；接下来触控ctrl+c进行复制，然后粘贴到Paste编辑画面上。



步骤4：此时触控ok完成操作回到主画面便可看到已成功新增的多笔资料。



34.4 其他功能

- 修改资料：

  直接双击主画面上要修改的资料，或点击该资料后触控Edit按钮。

- 删除资料

  选取要删除的资料，触控Delete按钮；若要删除所有资料则压住键盘上的Delete键并点击住画面上的Delete按钮。

- 调整资料排列顺序

  选取单笔资料数据后，可利用主画面上的上下按钮调整顺序，以增加在EB8000软件中选取数据的方便性。

- 删除资料格式：

  在主画面左边的格式列表选取要删除的格式触控键盘上的Delete键，此时弹出确认画面并

触控Yes后删除该类型。

- 储存修改结果：

  修改完成后必须触控主画面上的Save按钮，此时重启EB8000即可看到变更的结果。

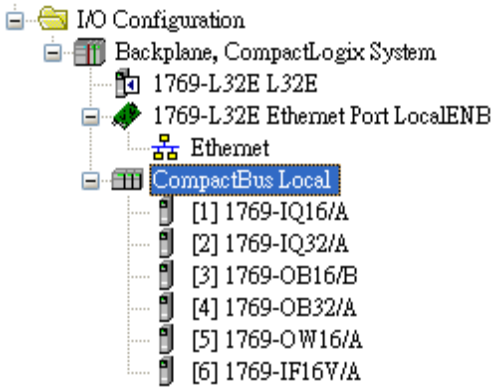
- 重新编辑：

  若要放弃所有修改重新编辑，触控主画面上的Reload按钮。

34.5 模组预设结构

模组预设结构Module-Defined：这里示范如何建立一个模组的预设结构。

在RSLogix5000的I/O Configuration里设定了I/O模组：

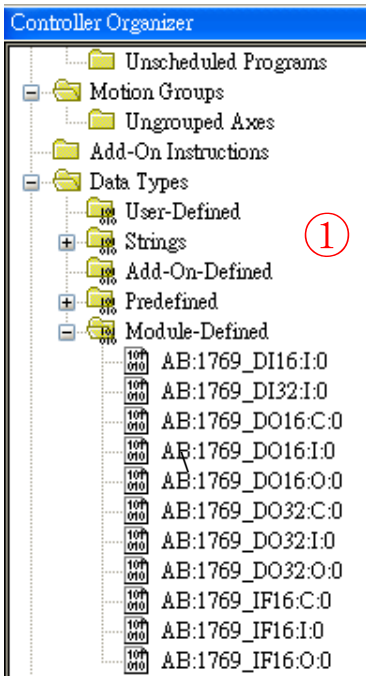


这些模组的Tag在输出到CSV文档时并不会把结构列出来，所以我们必须帮它建立：

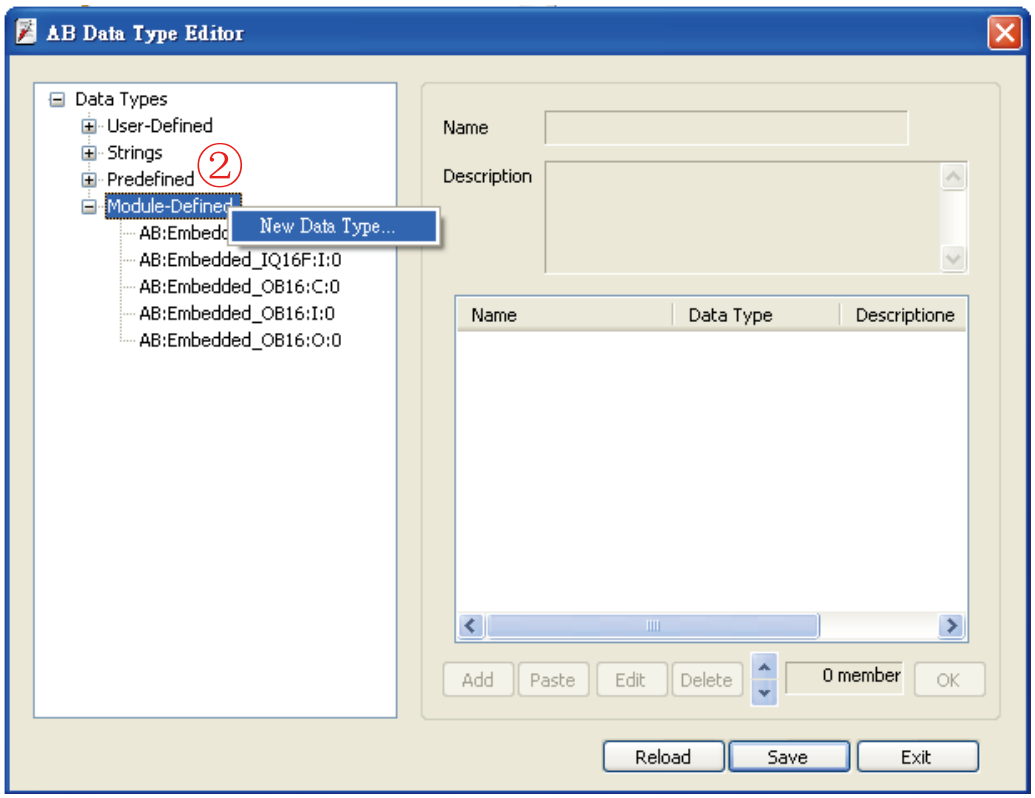
	A	B	C	D	E	F	G	H
7	TYPE	SCOPE	NAME	DESCRIPT	DATATYPE	SPECIFIER	ATTRIBUTES	
8	TAG		Local:1:I		AB:1769_DI16:I:0			
9	TAG		Local:2:I		AB:1769_DI32:I:0			
10	TAG		Local:3:C		AB:1769_DO16:C:0			
11	TAG		Local:3:I		AB:1769_DO16:I:0			
12	TAG		Local:3:O		AB:1769_DO16:O:0			
13	TAG		Local:4:C		AB:1769_DO32:C:0			
14	TAG		Local:4:I		AB:1769_DO32:I:0			
15	TAG		Local:4:O		AB:1769_DO32:O:0			
16	TAG		Local:5:C		AB:1769_DO16:C:0			
17	TAG		Local:5:I		AB:1769_DO16:I:0			
18	TAG		Local:5:O		AB:1769_DO16:O:0			
19	TAG		Local:6:C		AB:1769_IF16:C:0			
20	TAG		Local:6:I		AB:1769_IF16:I:0			
21	TAG		Local:6:O		AB:1769_IF16:O:0			

① 在RSLogix5000的Controller Organizer/Data Types/Module-Defined，双击DataType， 就会弹出对话框显示模组的Data Type的Members.

Copy Members里的Name和Data Type.

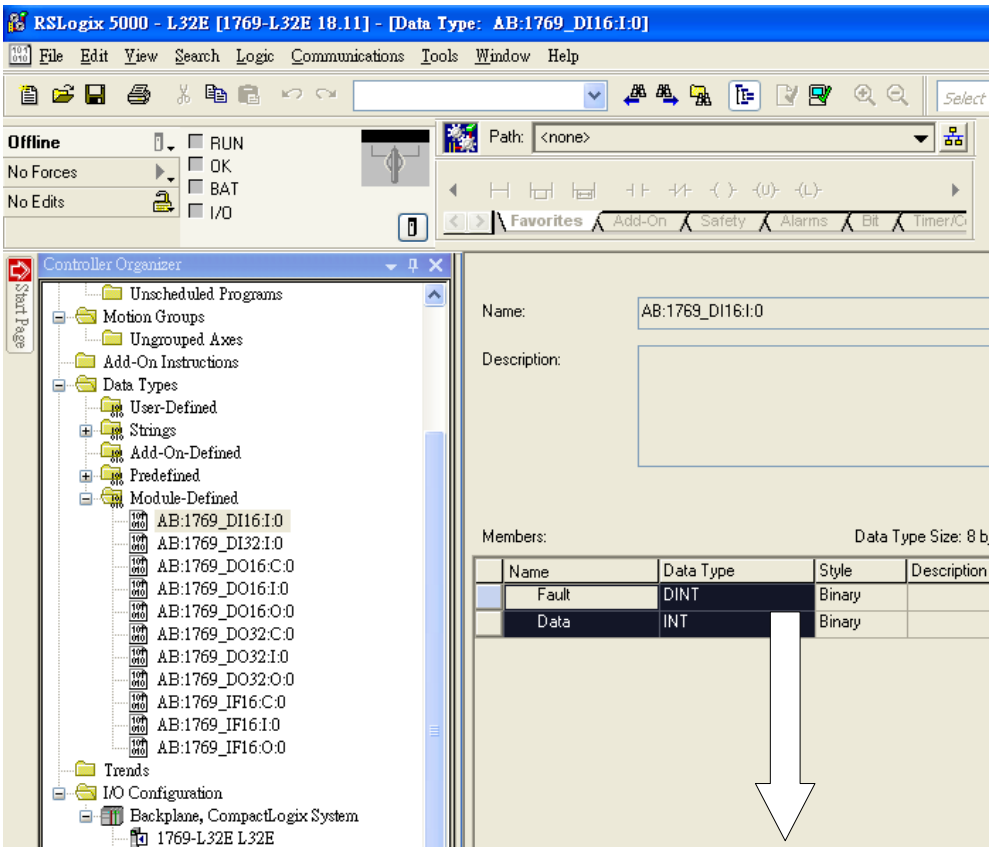


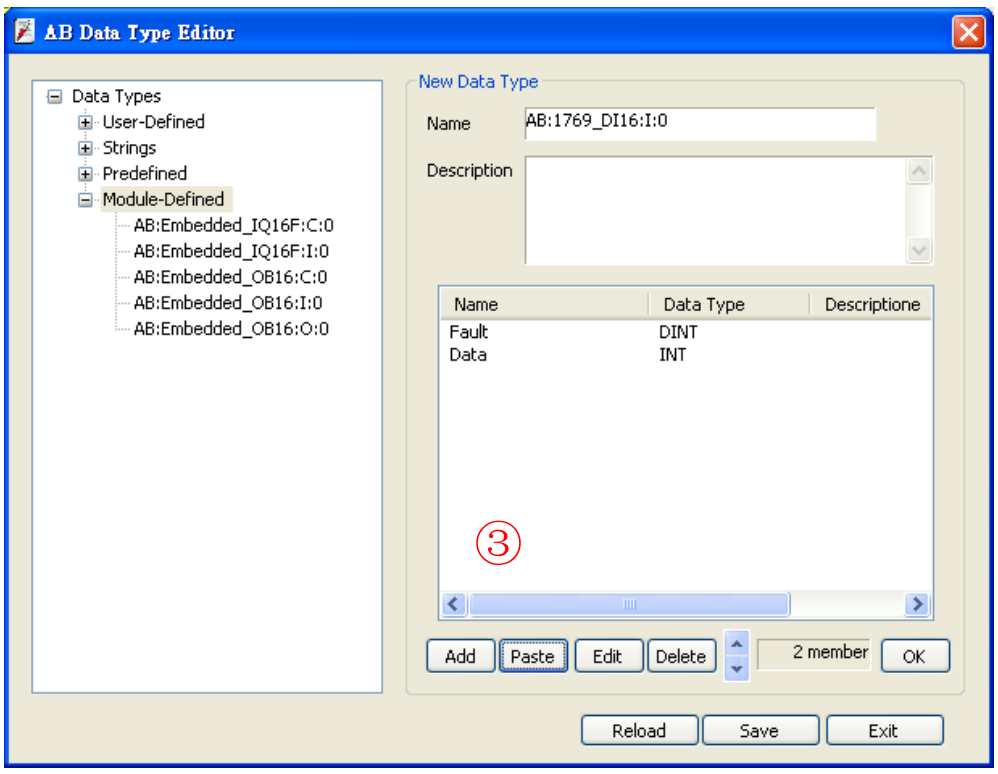
② 在EB8000的AB Data Type Editor.exe，右击Module-Defined，选择New Data Type…



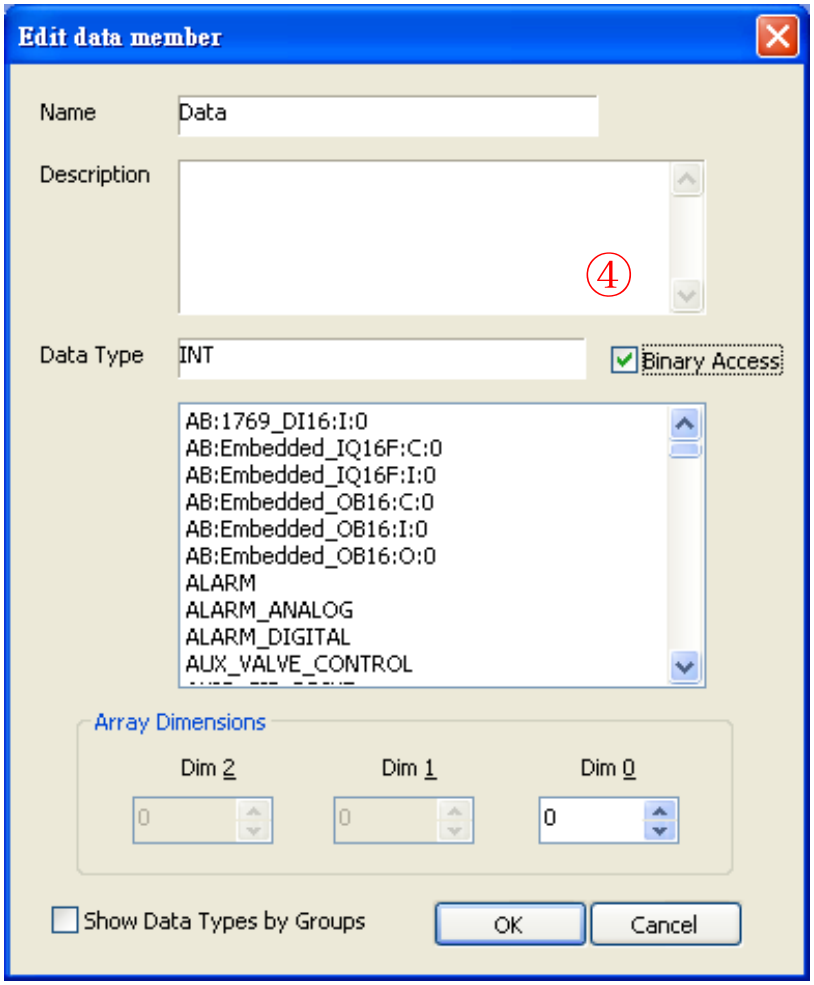
在New Data Type的Name栏写入Module-Defined Name,

③ 触控Paste按钮, 在对话框中触控Ctrl+V把Name和Data Type粘贴上。





④ 点击Data再触控Edit按钮，因为模组的资料可以用bit来操作，所以这里要勾选 Binary Access，触控OK回到Data Type Editor。



触控OK完成设定。

第三十五章 FTP Server运用

除了使用U盘或EasyPrinter将历史数据由HMI备份出来，现在也能利用FTP Server实现这个目标；当工程文件下载到HMI后可透过FTP Server进行备份历史数据,配方数据及备份或更新配方数据的动作，但是无法删除在FTP Server内的文件。

35.1 登入FTP Server

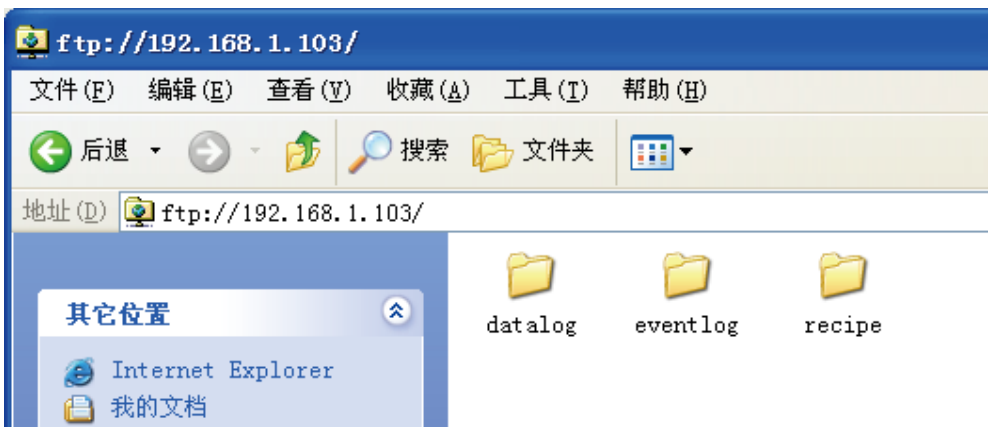
步骤1:在登入FTP Server之前请先确认是否使用以下的OS Image版本

MT8000i系列：OS Image 20100818或之后版本

MT8000 X系列：OS Image 20100906或之后版本

步骤2:在资源管理器的网址列输入HMI的IP地址ftp://192.168.1.103,然后登入账号uploadhis,密码为HMI的history upload password.(若没有更改过密码，皆为111111)。或是直接输入ftp://uploadhis:111111@192.168.1.103/

步骤3：输入IP地址后，于地址栏会显示为ftp://192.168.1.103/，并且可看到datalog，evenlog及recipe的文件夹。



35.2 备份历史数据及更新配方数据

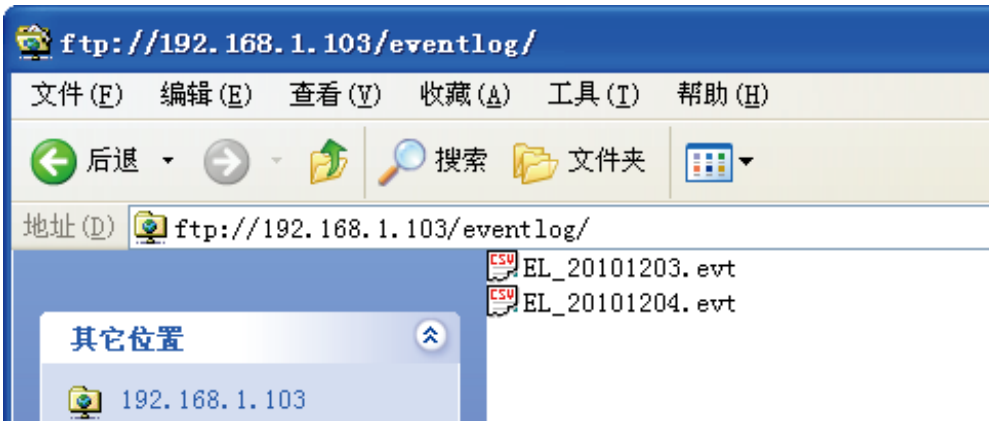
步骤1: 备份“数据取样”记录

点选datalog文件夹后, 可看到datalog在EB8000设定的文件夹名称, 再次点选文件夹名称后即可看到datalog的文件。

可使用复制并粘贴的功能将数据取样的记录保存在PC上。

步骤2: 备份“报警及事件”记录

点选eventlog文件夹后, 即可看到事件的记录文件。同样可使用复制及粘贴的功能将事件记录保存在PC上。



步骤3: Recipe配方数据

点选recipe文件夹后, 即可看到事件的记录文件。同样可使用复制及粘贴的功能将事件记录保存到PC上。

要将新的recipe档案更新到HMI上, 可将新的recipe.rcp复制到此目录, 并覆盖原有文件, 且于1分钟内将HMI重开机即可。

注意事项:若是更新recipe.rcp后，务必于1分钟内重开机或是使用LB-9048 LB-9047将HMI重开机。需先将LB-9048设ON后再将LB-9047设ON以顺利完成重开机的动作。

LB-9048 重启机制保护;

LB-9047 重新启动人机（设定为ON，并当LB9048状态为ON时）。

第三十六章 Easy Watch

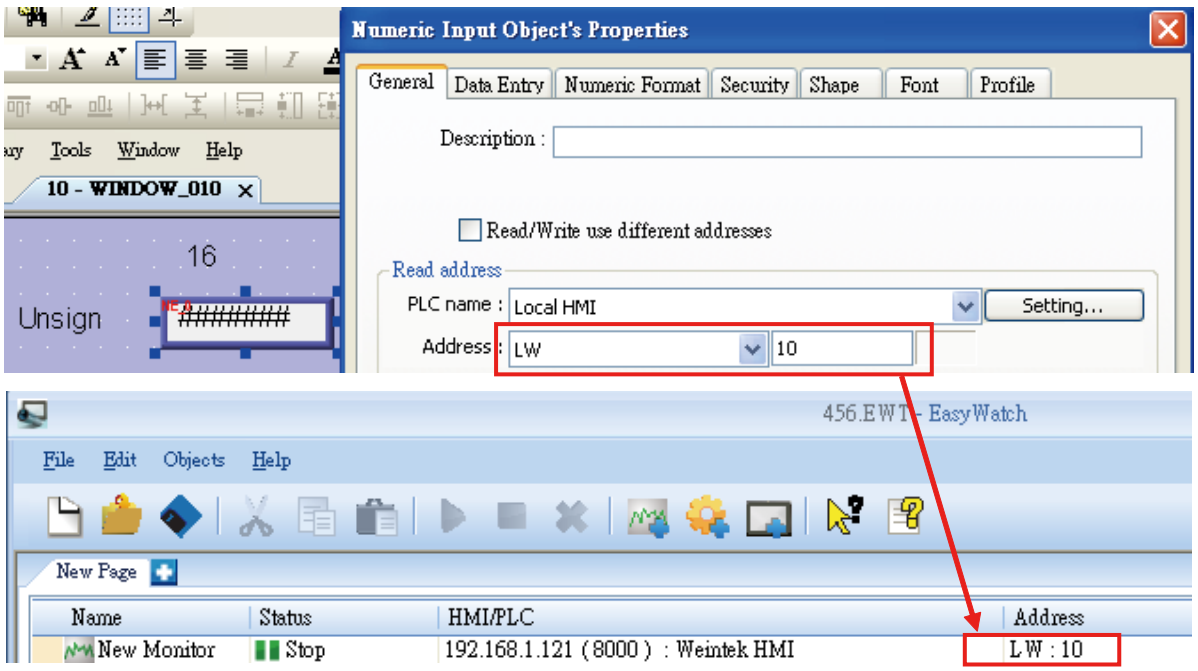
36.1 概论

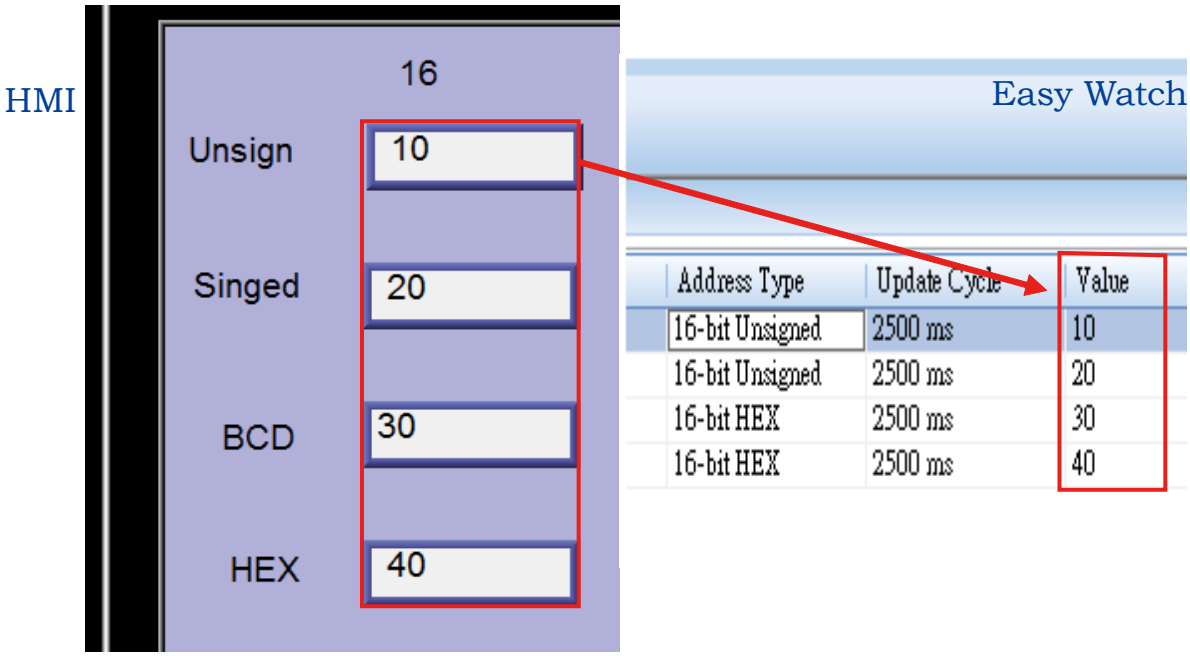
36.1.1 什么是Easy Watch

Easy Watch是可以透过HMI监看或设定HMI和PLC内的地址数值，以达到监控的目的，同时也可以进行Macro的呼叫，更方便使用者进行除错及远程监控使用。此手册主要介绍基本功能操作、Monitor设定、Macro设定、HMI管理等功能。希望通过手册的说明，能让使用者能够快速的掌握Easy Watch的监控功能。

36.1.2 为什么需要设计Easy Watch

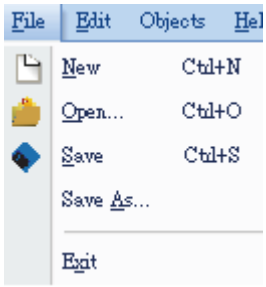
使用者使用EasyBuilder8000新增project时，可透过Easy Watch查看设定值及数据的正确性，以下是在EasyBuilder8000新增数值对象，地址设定成LW10，再到Easy Watch新增相同地址，当执行Monitor时，Status为Connected，且Value显示正确数值，表示已联机，即可开始监看。设定正确无误时，Easy Watch会显示出与 HMI 相同的数值。

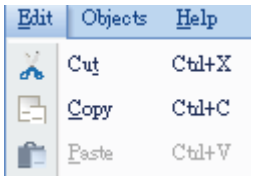
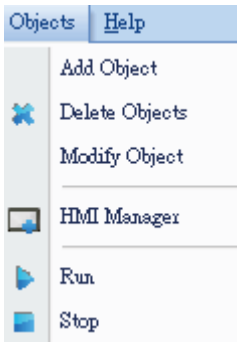
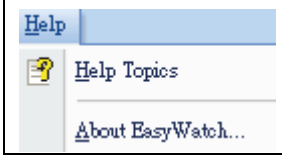




36.2 基本功能介绍

36.2.1 基本功能接口

项目	叙述
文件 	**新建(New)** 开启新增的 Easy Watch 文件
	打开(Open) 开启已编辑的 Easy Watch 文件
	保存(Save) 储存 Easy Watch 文件设定
	另存为新文件(Save As) 可将 Easy Watch 文件设定，储存成 EWT 格式
	退出(Exit) 关闭 Easy Watch

编辑 	**剪切(Cut)** 剪下选取的对象至剪贴簿中
	复制(Copy) 复制选取的对象至剪贴簿中
	粘贴(Paste) 贴上剪贴簿中的对象
对象 	**新增对象(Add Object)** 可新增 Monitor 或 Macro 对象
	删除对象(Delete Objects) 选择欲删除的对象，会跳出信息，确认删除选择「是」即可删除
	修改对象(Modify Object) 选择欲修改的对象，即可修改内容
	HMI 管理 (HMI Manager) 对 HMI 进行新增、修改、移除的管理
	执行(Run) 选择欲执行的对象，即可执行此对象
	停止(Stop) 选择执行中的对象，即可停止此对象
帮助 	**功能说明(Help Topics)** 提供基本功能的操作方法，供使用者参考
	About Easy Watch 显示此版本信息

36.2.2 快速工具条

 新建：新建Easy Watch文件。

 打开：打开已编辑的Easy Watch文件。

 保存：保存Easy Watch文件设定。

剪切: 剪下选取的对象至剪贴簿中。

复制: 复制选取的对象至剪贴簿中。

粘贴: 粘贴剪贴簿中的对象。

开始执行对象: 选择欲执行的对象, 即可执行对象。

停止执行对象: 选择欲执行中的对象, 即可停止对象。

删除对象: 选择欲删除的对象, 即可删除对象。

Monitor: 新增监视对象。

Macro: 新增宏对象。

HMI管理: 对HMI进行新增、修改、移除的管理。

Help: 点选欲查询的功能, 即可出现说明。

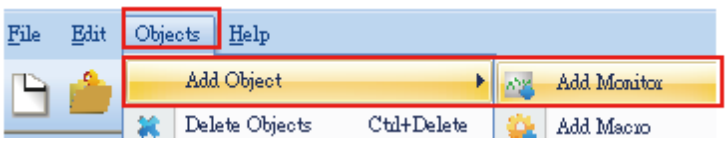
Help Topics: 说明基本功能

36.3 Monitor设定

36.3.1 新增Monitor对象

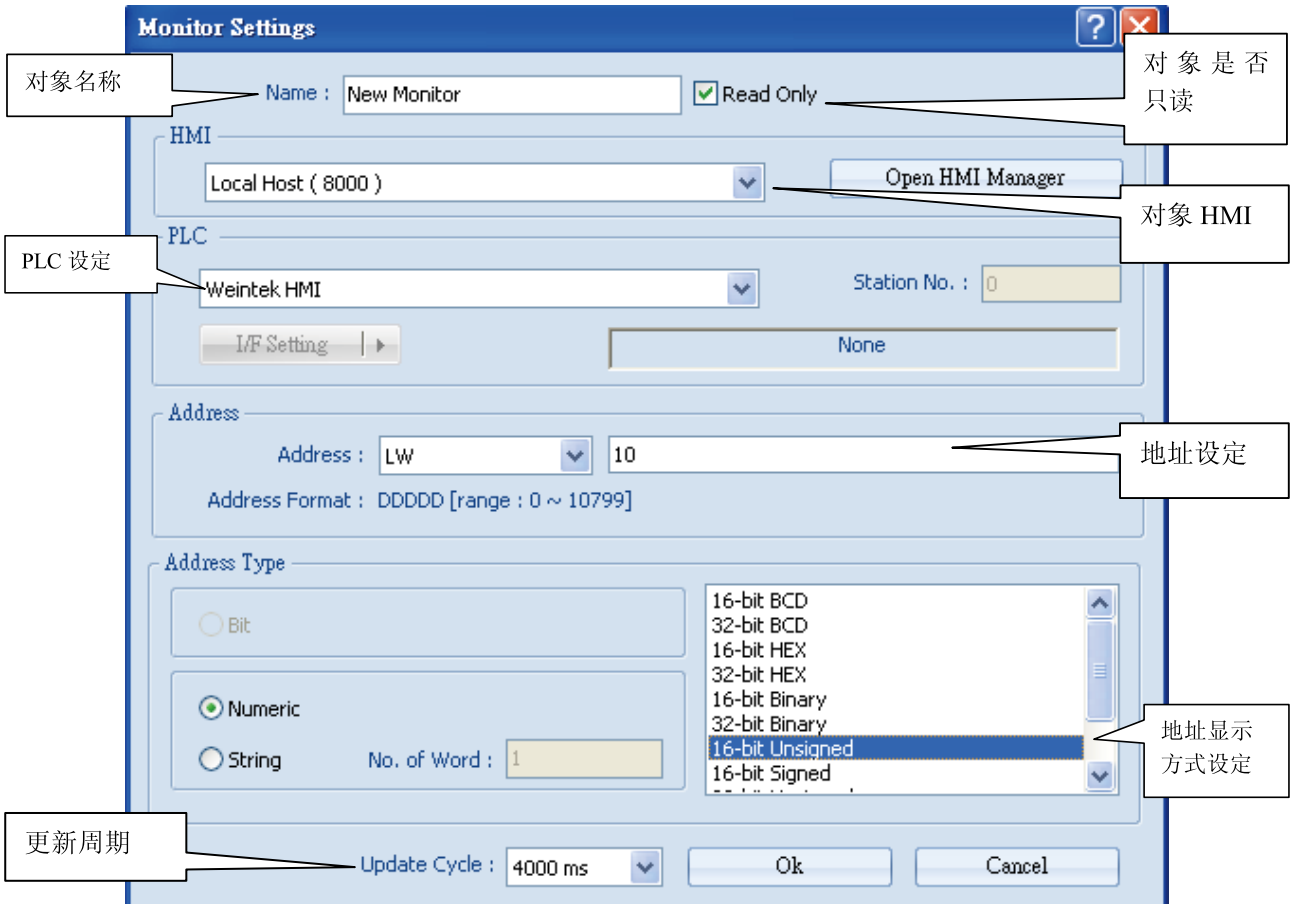
有两种方法可以新增:

a、在基本工具栏选择Objects->Add Object->Add Monitor



b、在快速工具栏选择新增Monitor对象

36.3.2 Monitor设定接口



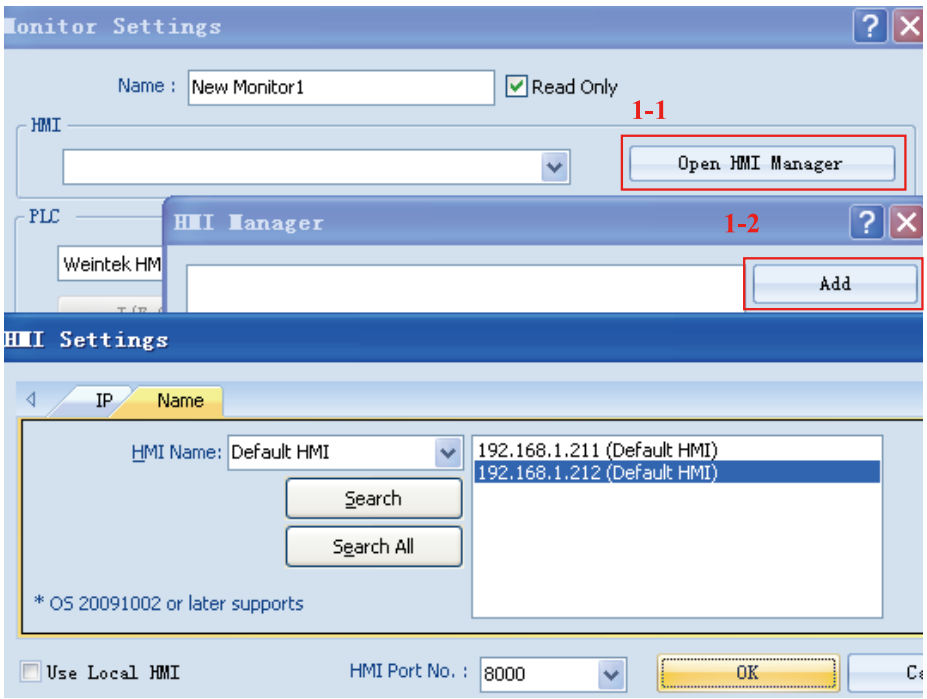
1. 对象名称: 对对象命名, 名称不可重复。
2. 对象只读: 若将对象设为只读时, 将不能设定该地址的数值。
3. 对象HMI: 欲监视的HMI。
4. PLC设定: 设定该地址所属PLC的类型和站号以及联机方式。
5. 地址设定: 设定该地址的型别以及地址。
6. 地址显示方式设定: 会依照地址类别列出可以选择的显示方式, 执行时会依照显示方式来解析并显示该地址。
7. 更新周期: 地址的更新周期, 若同时运行过多的对象将导致误差与延迟。

36.3.3 新增Monitor设定

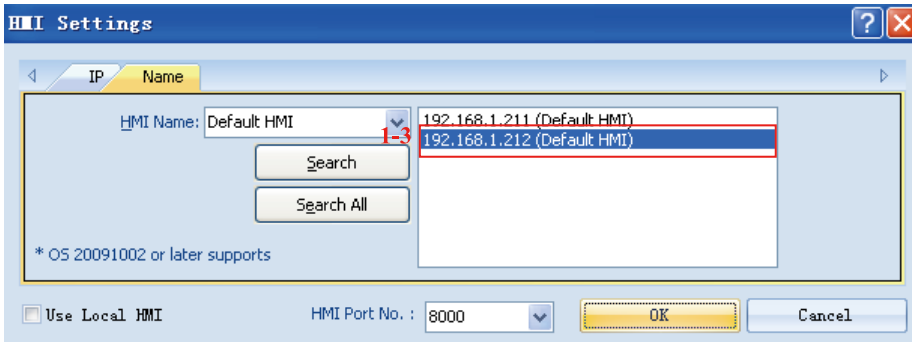
1. 开启Monitor Settings，选择欲操作的HMI，若HMI不存在可透过下列步骤来新增：

1–1 点选Open HMI Manager钮，会开启HMI选单。

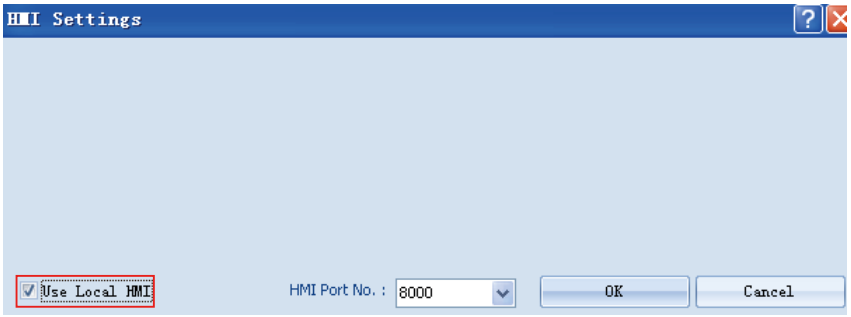
1–2 点选Add,会跳出HMI选取对话框，可透过搜寻功能列出局域网络中的HMI。



1–3 选取欲操作的HMI，并点击OK钮即可完成新增动作。

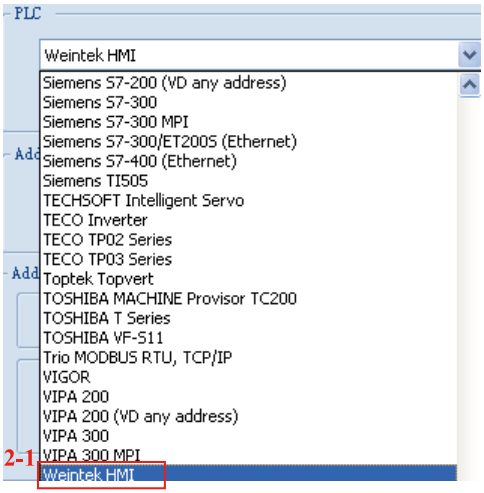


1–4 另外也可透过勾选Use Local HMI，来新增脱机模拟中的HMI。

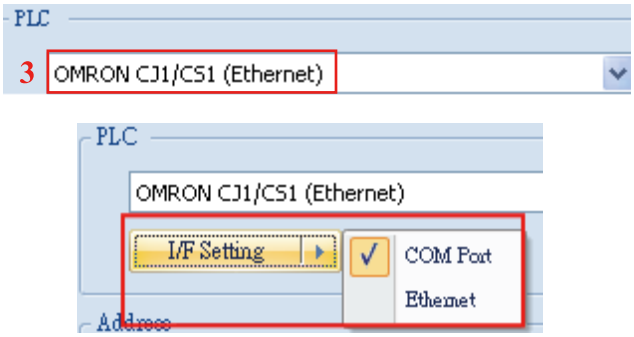


2. 选择PLC，可选择直接操作Weintek HMI或PLC。

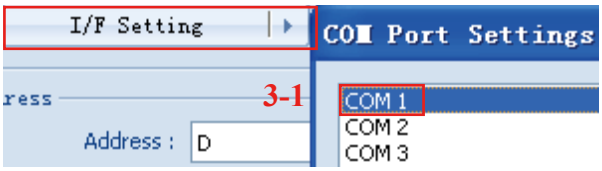
2-1 选择Weintek HMI，即可直接对本机HMI操作。



3. 选择监视PLC时，PLC连结方式(I/F Setting)可使用COM Port，或是Ethernet。

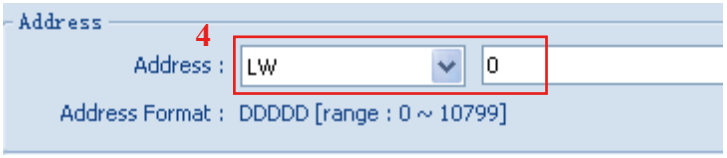


3-1 选择COM Port时，点击I/F Setting会跳出COM Port选项可供选择。



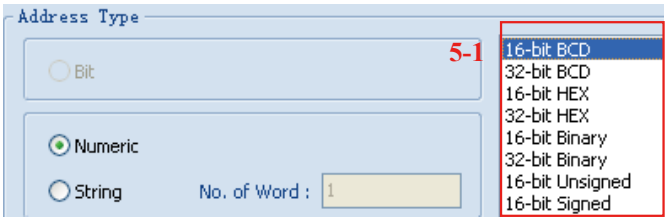
3-2 选择Ethernet时，点击I/F Setting会跳出IP Address设定字段。

4. 设定PLC地址。

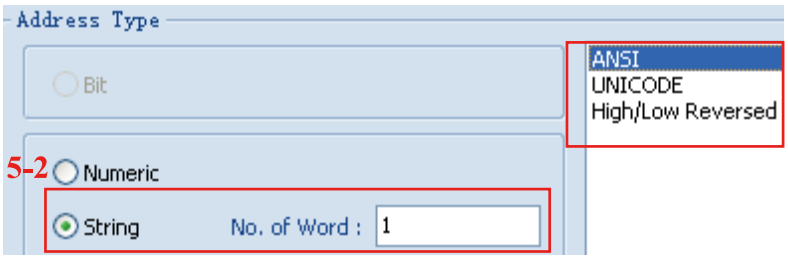


5. 地址格式(Address Type)可设定该地址为数值或字符串。

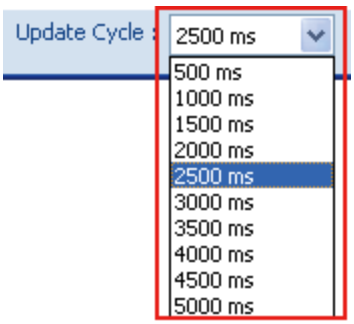
5–1选择数值：自下拉选单选择读取地址缓存器内的数据格式。



5–2选择字符串：可设定ANSI、 UNICODE、High/Reversed三种格式的数据。并可由No. of Word 设定欲读取的字符数。



6. 设定执行地址的时间间隔。

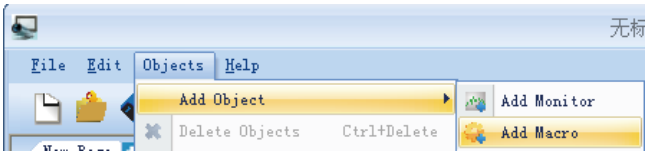


36.4 Macro设定

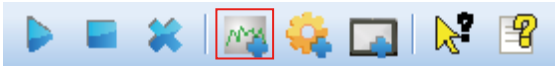
36.4.1 新增Macro

有两种方法可以新增:

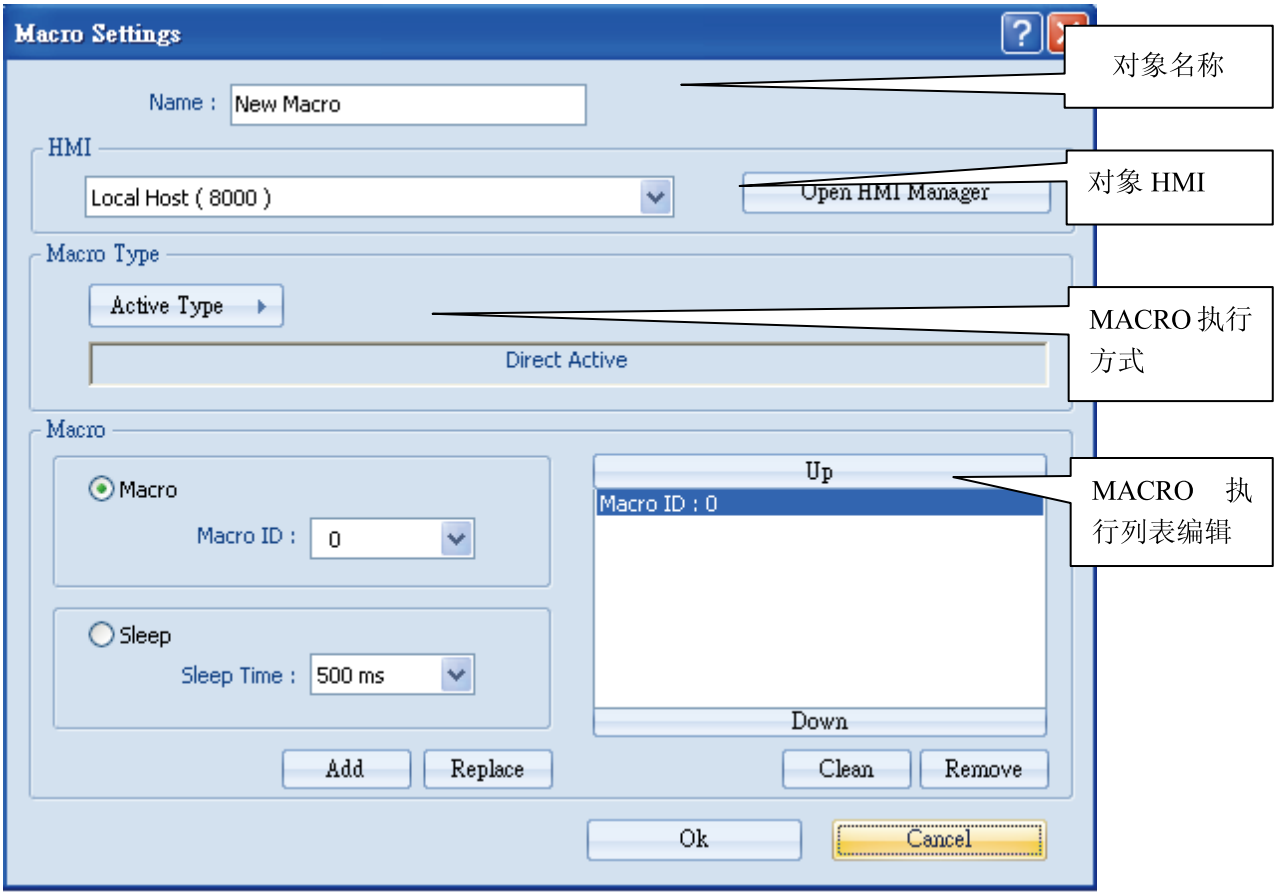
a、在基本工具栏选择Objects–>Add Object–>Add Macro



b、在快速工具栏选择新增Macro



36.4.2 Macro设定窗



1. 对象名称: 对对象命名, 名称不可重复。

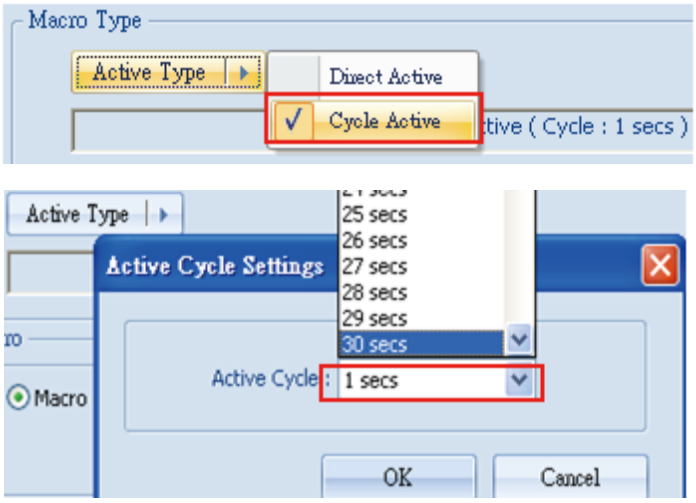
2. 对象HMI：MACRO位于的HMI。
3. MACRO执行方式：分为直接呼叫和周期性呼叫:
4. MACRO执行列表编辑：每个MACRO对象可执行数个MACRO，MACRO与MACRO执行之间可设定间隔时间。

36.4.3 如何新增Macro设定

1. 欲新增HMI 可参照 “3.3 新增Monitor 设定”。
2. 选择Macro Type，可选择Direct Active和Cycle Active。

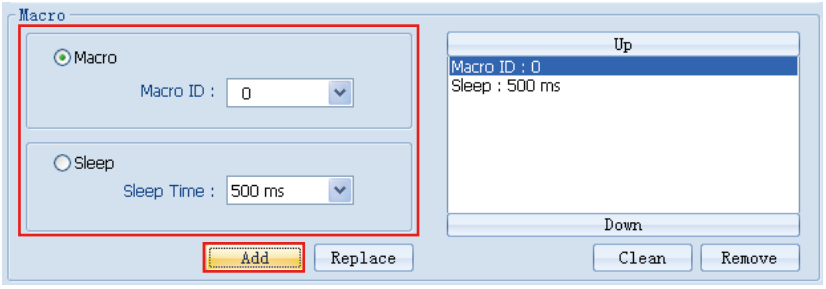
2-1选择Direct Active：MACRO 会直接执行，并执行一次

2-2选择Cycle Active：点击Active Type，选择Cycle Active，可设定MACRO执行周期。假设在Active Cycle设定为5秒，当执行完所有Macro，下次重新执行Macro的时间为5秒之后。

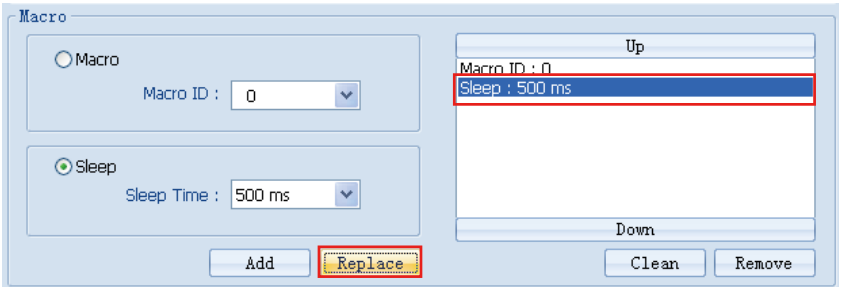


3. Macro列表中包含执行Macro与Sleep间隔时间。Macro可以设定要执行的Macro ID；Sleep可设定间隔时间。透过Add或Replace可新增或取代Macro列表中所选取的部份。

3-1选择要执行的Macro ID，点击Add即可新增到Macro列表中。



3–2选择Sleep Time的时间，点击Replace即可取代选择的项目。

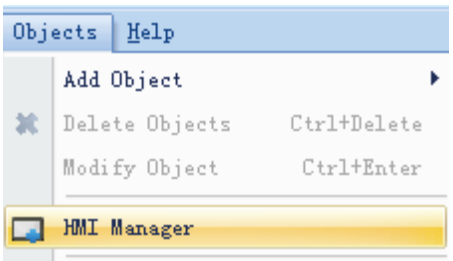


36.5 HMI设定

36.5.1 HMI设定

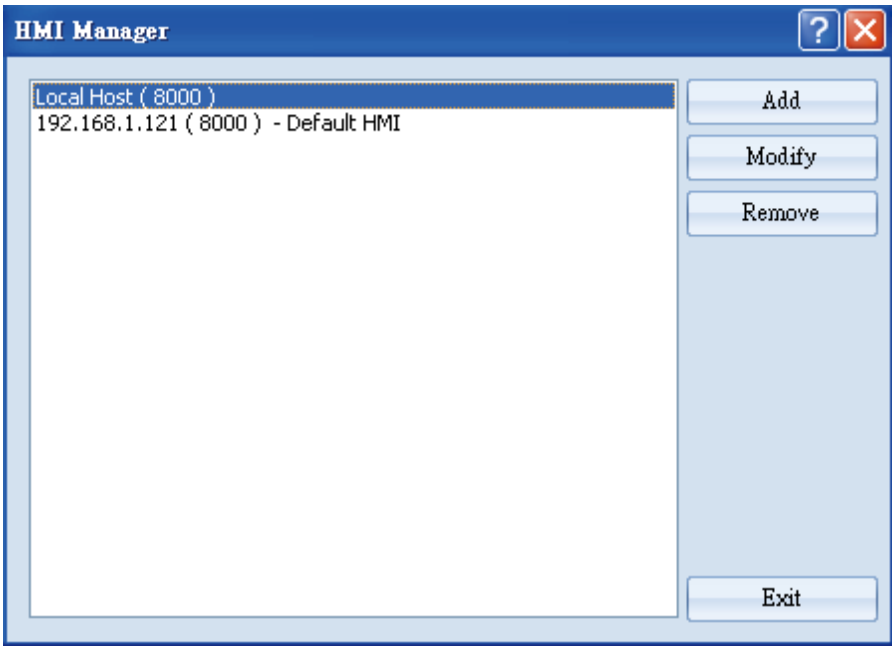
有两种方法可以开启HMI设定：

a、在基本工具栏选择Objects–>HMI Manager



b、在快速工具栏选择HMI Manager

36.5.2 HMI设定接口

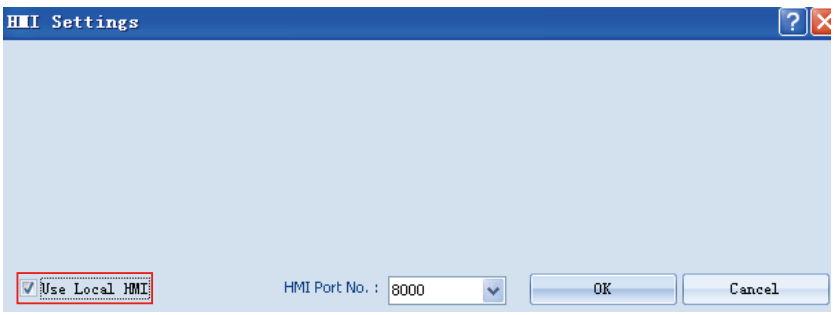


EasyWatch可支持同时监控不同HMI上的地址，方便用户管理数个HMI。

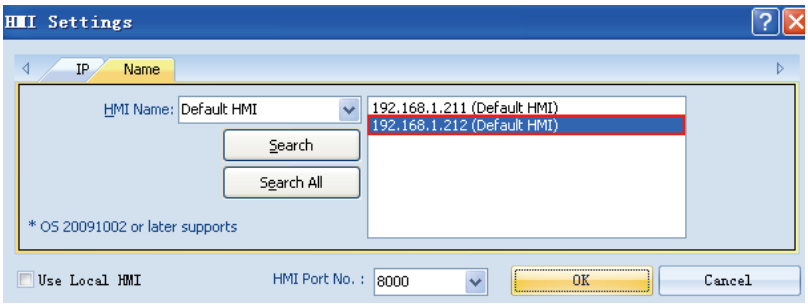
36.5.3如何新增HMI设定

1. HMI设定可新增、修改、删除。

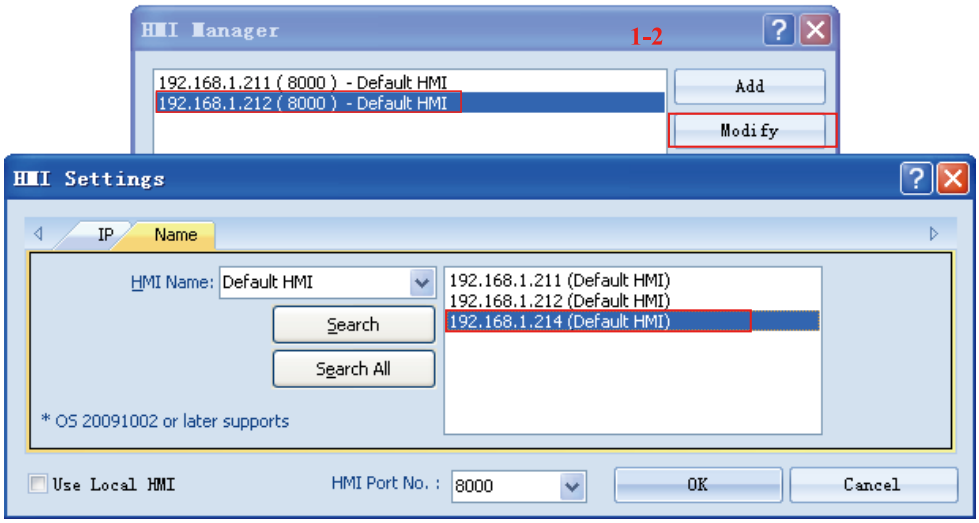
1-1新增：勾选Use Local HMI时，可新增脱机模拟中的HMI。



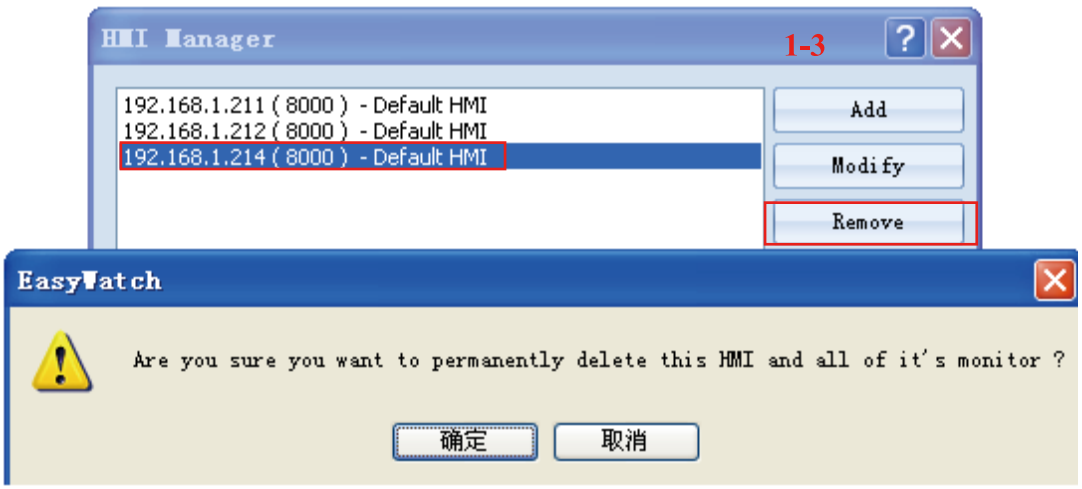
未勾选Use Local HMI时，可透过搜寻功能列出局域网络中存在的HMI。相同IP的HMI会被视为同一HMI，即使HMI Port No.不同也无法新增。



1–2修改：选择要修改HMI项目即可做修改。



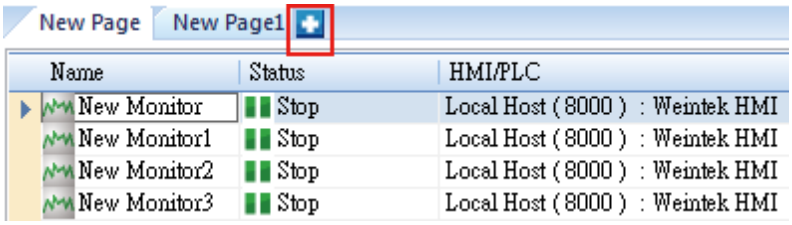
1–3移除：选择要移除HMI项目，确认后即可移除。



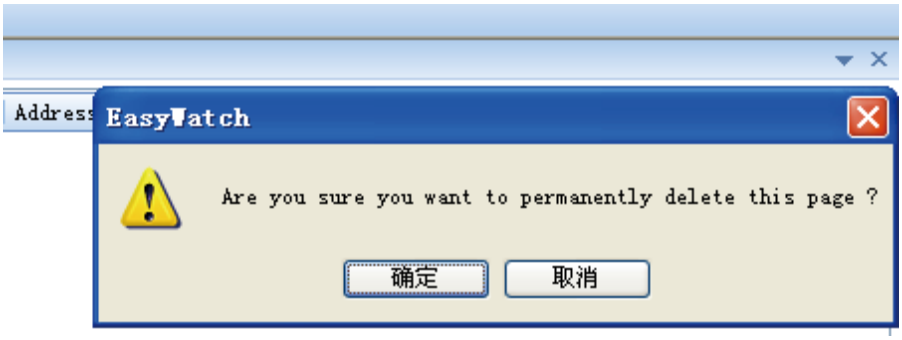
36.6 对象显示列表

36.6.1 页面设定

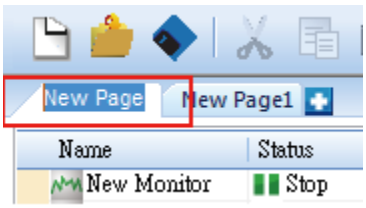
1–1新增页面：可以点选+来新增页面。



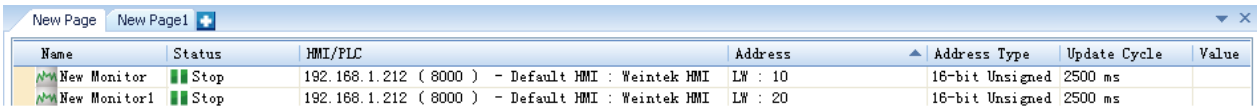
1–2删除页面：点击×并经确认后可删除该页面。



1–3更名页面：在页面的名称上双击后，通过输入新的名称即可更名。



36.6.2 对象显示字段



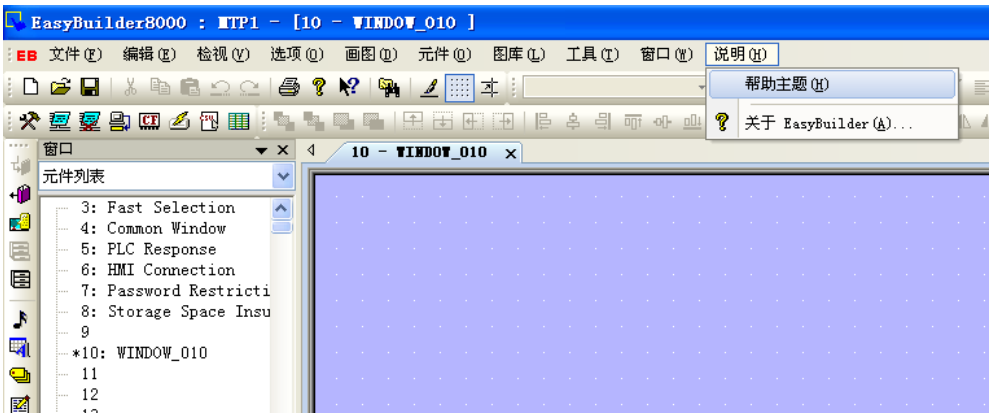
1. Name：显示对象名称，并且透过显示图形，可以方便用户识别对象的种类。
2. Status：会显示目前对象的执行状态，分别有Connecting、Connected或是Stop，同时也可显示错误信息，若该HMI不在线或是Port No.输入错误会显示「HMI Not Found」；若为Monitor对象且地址设定有误，会显示「Address Error」的信息。

3. HMI/PLC：会显示目前对象所操作HMI/PLC相关信息。
4. Address、Address Type：若为Monitor对象会显示其地址相关设定数据。
5. Update Cycle：显示更新周期。
6. Value：若为Monitor对象，并且状态为Connected时，会显示目前HMI上该地址的数值，当Monitor对象不具有Read Only 属性时也可透过修改该字段来设定监视地址的内容。若为Macro对象，并且型态为Direct Active，在Value字段上会显示按钮，点击后可直接执行该Macro。
7. 对象显示字段顺序：可依使用者喜好自行排列选择。

第三十七章 MT8000系列HMI与常见厂牌PLC的连接方法资料获取

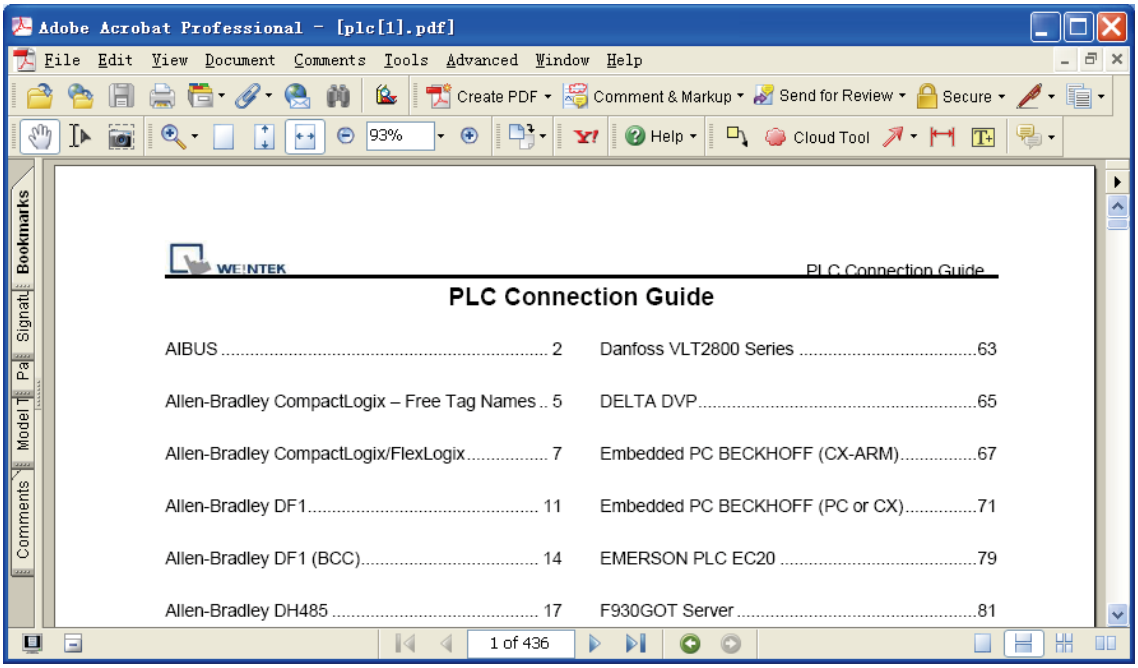
1、通过EB8000帮助获取

（1）点击“说明”菜单–帮助主题



（2）在“帮助”窗口中，展开“参考信息”目录，点击“PLC连接手册”；

（3）弹出一份PLC 连接手册的pdf文档；



2、通过WEINVIEW官方网站获取

在浏览器地址栏中输入网址：http://www.weinview.cn ，于“下载中心”栏目中下载“EasyBuilder8000使用手册.pdf”

3、向WEINVIEW供应商索取“与PLC连接手册”书籍。

威纶通科技有限公司

www.weinview.cn

[深圳]
地址：深圳市南山区登良路与南海大道交汇处恒裕中心Ｂ座十层
电话：0755-26456333
传真：0755-86036588

[苏州]
地址：苏州市工业园区星汉街5号新苏工业坊B幢02－10
电话：0512-69001690
传真：0512-62990145

[北京]
地址：北京市海淀区长春桥路11号万柳亿城中心C2-602
电话：010-58819930
传真：010-58819932

[武汉]
地址：武汉市武昌区中南路7号中商广场写字楼A座A1309-10
电话：027-87269732
传真：027-87269733

[重庆]
地址：重庆市九龙坡区渝州路4号一城精英国际20-16
电话：023-89128322
传真：023-89128313

[济南]
地址：山东省济南市经一路88号明珠商务港2511室
电话：0531-82635489
传真：0531-82631895