ControlStar 编程软件

用户手册

资料版本 V1.0

归档时间 2005 年 3 月 24 日

BOM编码 31011119

艾默生网络能源有限公司为客户提供全方位的技术支持，用户可与就近的艾默生网络能源有限公司办事处或客户服务中心联系，也可直接与公司总部联系。

艾默生网络能源有限公司

版权所有，保留一切权利。内容如有改动，恕不另行通知。

艾默生网络能源有限公司

地址：深圳市南山区科技工业园科发路一号

邮编：518057

公司网址：www.emersonnetworkpower.com.cn

客户服务热线：0755-86010800

E-mail：info@emersonnetwork.com.cn

前 言

本书详细地描述了 ControlStar 编程软件的操作方法和界面说明，同时介绍了各功能的使用方法并穿插了应用实例。

如何阅读本书

本书介绍了艾默生网络能源有限公司 EC 系列可编程控制器的 ControlStar 编程环境的操作方法。

本书的章节编排由整体到细节，各章节都具有独立的内容，因此可以从头通读逐步掌握 ControlStar 编程环境的全面内容，也可以随时查阅其中的章节，作为一本技术参考资料。

详细的 PLC 程序语法、指令使用和示例等内容，请参阅《编程参考手册》。

相关文档和参考书目

阅读本手册的同时也可以参考阅读其他手册：用户操作手册、设备选型手册、编程参考手册。

目 录

第一章 概述

1.1 可编程控制器简介

可编程控制器 PLC(Programmable Logic Controller)是一种数字运算操作的电子系统，专为工业环境下应用而设计。它主要将外部的输入信号如：按键、感应器、开关及脉波等的状态读取后，依据这些输入信号的状态或数值并根据內部储存预先编写的程序，以微处理机执行逻辑、顺序、计时、计数及算术运算，产生相对应的输出信号如：继电器的开关、控制机械设备的操作。通过电脑或程序书写器可轻易的编辑/修改程序及监控装置状态，进行现场程序的维护与试机调整。

1.2 基本控制原理

1.2.1 可编程控制器工作原理

可编程控制器采用循环扫描工作方式，包含输入点扫描、用户程序执行、输出点刷新、内部处理和通讯处理几个部分。

在运行可编程控制器前，可以使用编程软件编写输入点和输出点间的控制逻辑并下载到可编程控制器中，在可编程控制器运行过程中，首先会扫描输入点信号，将之读取到可编程控制器中，然后根据控制程序完成运算和逻辑处理，运算和逻辑处理结果将输出点的值改变，最后将输出点中的值转换为电信号输出并控制各种机械设备的运行。

在可编程控制器运行过程中采用循环扫描的工作方式，通过反复的执行输入点扫描、用户程序执行、输出点刷新工作达到接收控制并操作设备的目的。

1.2.2 用户程序控制原理

在用户程序中，可编程控制器的输入点被称为触点，它的功能和工业设备中的开关触点是一样的，代表能流的导通或者关断。在可编程控制器中，输入点存储为一个软元件，当输入点为高电平时，对应的软元件为导通状态，在用户程序中参与逻辑

运算并影响输出点的值；输出点称为线圈，代表输出能流的导通或者关断，输出点对应软元件的值由输入点和控制逻辑计算结果决定。在输出刷新时，软元件的值被转换为电信号在输出点的晶体管或者继电器输出，从而完成对设备的控制。

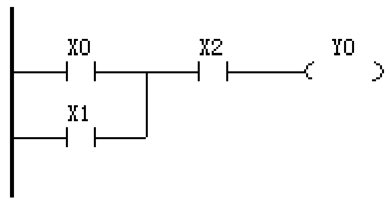
1.3 编程语言

EC 系列可编程控制器的编程工具支持三种常用的语言：梯形图（LD），指令列表（IL）和顺序功能图（SFC）。主程序可以使用上述三种编程语言中的任意一种来编写，但是子程序和中断子程序只能使用梯形图或者指令列表编写。另外，顺序功能图中的内置程序只能使用梯形图编写。

1.3.1 梯形图编程

梯形图是一种和电气原理图非常类似的图形编程语言，在逻辑上也是和电气原理图非常的相似。在梯形图中，各种软元件被表示为触点和线圈，图形上各个触点和线圈之间的电路连接关系就是它们的控制逻辑关系。

示例：



在上面的梯形图中，首先是装载输入点 X0 的值作为当前值，然后装载输入点 X1 的值，将 X1 的值和当前值进行“或”运算后，运算结果成为当前值，然后装载 X2 的值并和当前值进行“与”运算，运算结果将最终控制输出点 Y0 是否有能流导通。

1.3.2 指令列表编程

指令列表程序编辑器是一个文本编辑器，所有的逻辑和运算都使用指令和操作数的方式输入，根据指令所完成的功能和涉及到的操作数中的软元件，完成软元件的值读取、逻辑处理和软元件值写入。

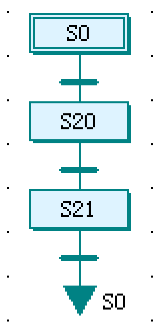
示例：

指令	软元件	指令说明
LD	X0	//装载输入点 X0 的值作为当前值
OR	X1	//装载输入点 X1 的值，和当前值进行“或”运算后结果成为当前值
AND	X2	//装载输入点 X2 的值，和当前值进行“与”运算后结果成为当前值
OUT	Y0	//根据当前值确定输出点 Y0 的值

1.3.3 顺序功能图编程

顺序功能图是根据机械设备的流程或者工序，将控制分成了多个步和步到步之间转换的一种语言。一个标准的顺序功能图由初始步、一般步、步间的转换条件、跳转和重置组成、每一步就是机械设备的一个处理工序，一个步中可以有内置梯形图，也就是这一步需要完成的处理工序。转换条件就是一个工序的完成和下一个工序的启动条件，它也需要内置梯形图来表示转换的条件。

示例：



如上面的顺序功能图所示，初始状态是 S0，当 S0 至 S20 的转换条件不满足时，S0 工序会一直被执行，一旦转换条件满足，就停止 S0 的执行，开始 S20 的执行，以此类推，在最后完成 S21 并满足 S21 到跳转 S0 的转移条件后，停止 S21 工序的执行，重新启动初始步 S0 的执行，通过跳转，可以让整个顺序工序循环不停的执行。

1.3.4 程序转换

上述的三种编程语言可以互相转换，根据使用习惯或者实际应用的控制环境，可以选择合适的编程语言。由于顺序功能图语言比较特殊，因此梯形图或者指令列表必须遵守顺序功能图的语法才能正确转换到顺序功能图。

第二章 安装与卸载

2.1 安装

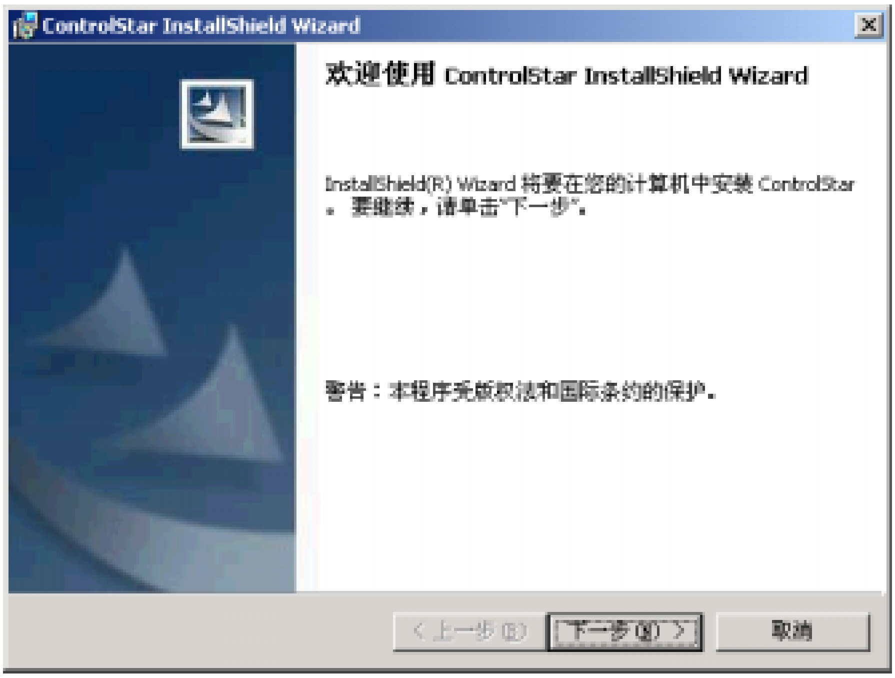
2.1.1 计算机配置要求

ControlStar 编程软件要求在 IBM PC486 以上的微型机或兼容机上运行，支持 Microsoft 的 Windows 98、Me、NT 4.0、Windows 2000 及 XP 等操作系统。需要安装 ControlStar 的计算机的最低配置和推荐配置如下：

	最低配置	推荐配置
CPU	相当于 Intel 公司的 Pentium 233 或以上级别的 CPU	相当于 Intel 公司的 Pentium 1G 或以上级别的 CPU；
内存	64M	128M
显卡	可工作于 640*480 分辨率，256 色模式下	可工作于 800*600 分辨率，65535 色模式下

2.1.2 安装过程

ControlStar 发布的安装包为单独的可执行程序，双击即可启动安装过程。显示如下界面：

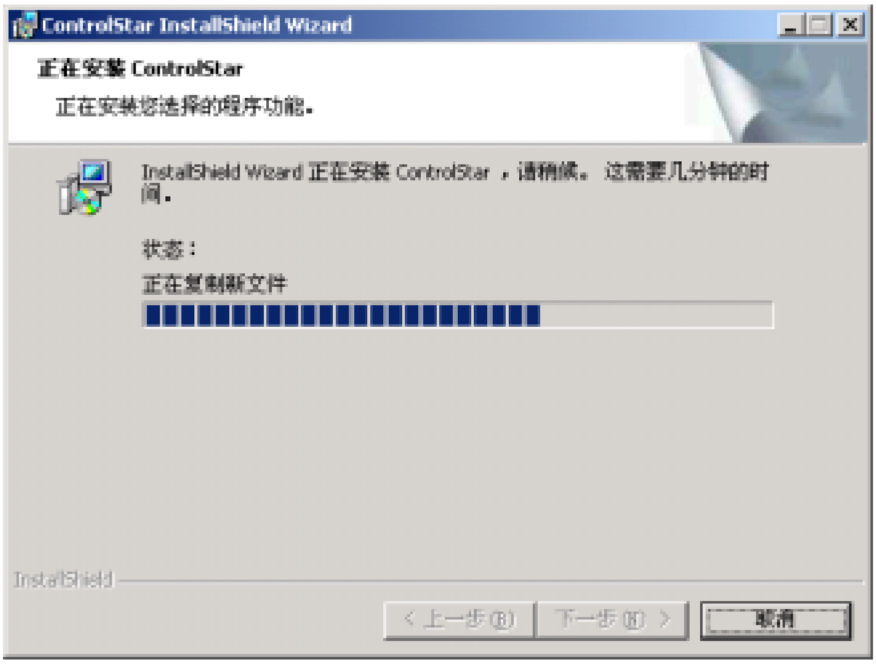


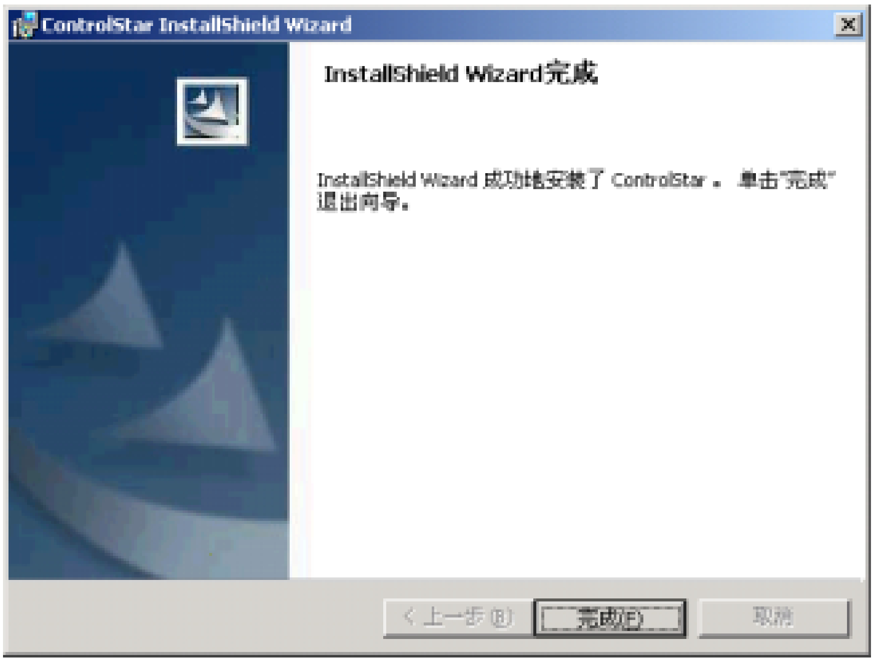
点击[下一步]按钮进入如下信息填制界面：

要求输入用户信息（注：在没有多用户的操作系统上不需要输入用户信息）。完成后点击[下一步]按钮进入如下界面：

在上面的界面中可以点击[更改]按钮改变程序的安装目录。点击[下一步]进行如下界面：


ControlStar InstallShield Wizard
已做好安装程序的准备
向导准备开始安装。
要查看或更改任何安装设置，请单击"上一步"。单击"取消"退出向导。
当前设置：
安装类型：
典型
目的地文件夹：
D:\Program Files\Emerson Network Power\ControlStar\
用户信息：
姓名： Administrator
公司： EMPC
InstallShield
< 上一步(B)
安装(I)
取消


ControlStar InstallShield Wizard
正在安装 ControlStar
正在安装您选择的程序功能。
InstallShield Wizard 正在安装 ControlStar，请稍候。这需要几分钟的时间。
状态：
正在复制新文件
InstallShield
< 上一步(B)
下一步(N) >
取消



安装完毕后，在开始菜单中会出现“ ControlStar ”程序组，同时会把 ControlStar 快捷方式放置在桌面上。

2.2 卸载

通过选择“ ControlStar ”程序组中的“ 卸载 ControlStar ”菜单或者在控制面板中使用“ 添加/删除程序 ”功能可以卸载本软件。

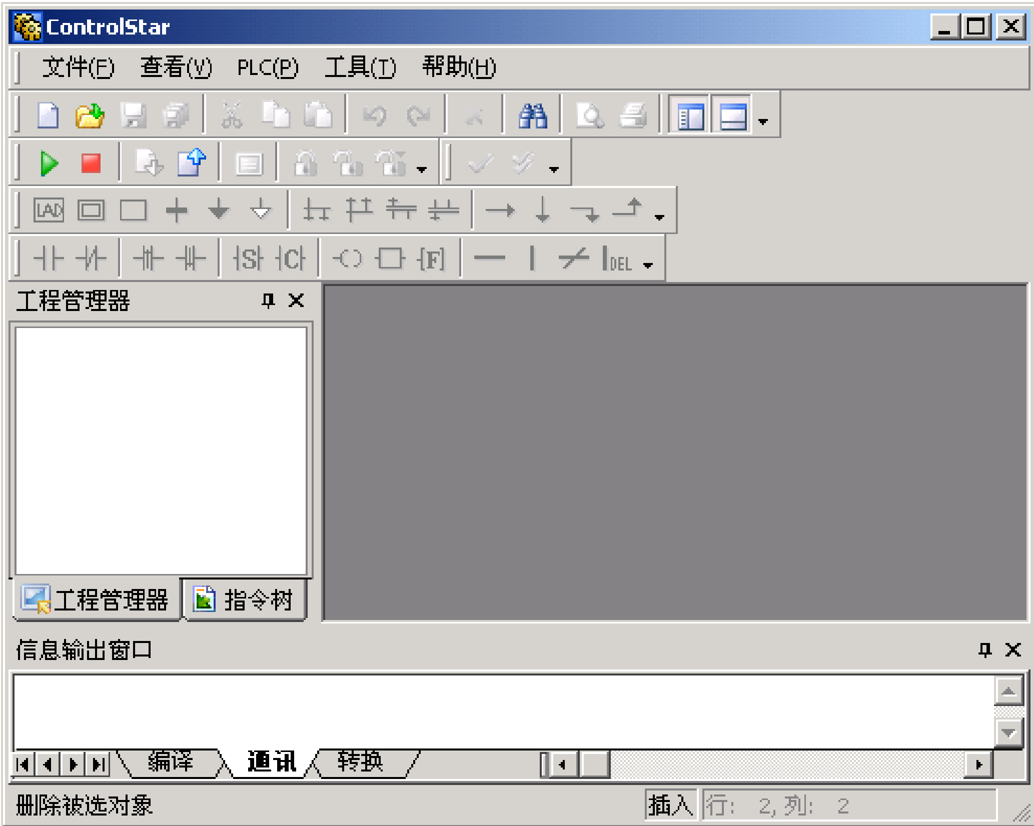
需要重新安装或升级 ControlStar 时，必须首先卸载旧版本的软件。为了避免版本升级可能带来的混乱，在安装和卸载的过程中，软件会自动清理注册表中的相关数据，因此，在重新安装或者升级之后，先前定制过的菜单、工具栏、系统选项等信息需要重新设置。

第三章 快速入门

在本章，我们将通过一个实际的控制例子，说明从如何启动编程环境，到最后将程序下载到可编程控制器中并运行。

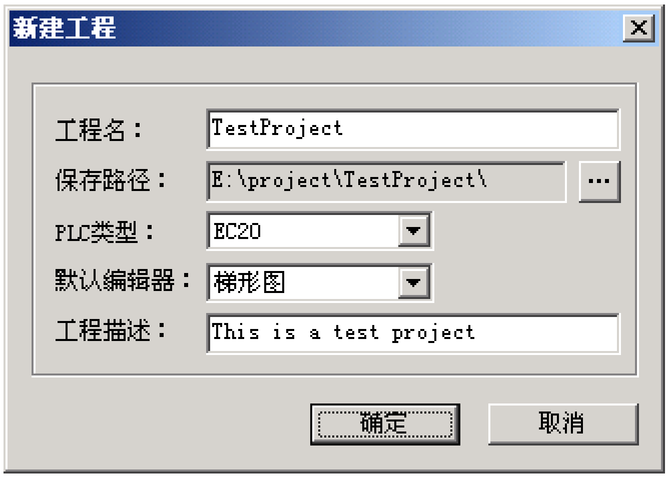
3.1 编程环境启动

ControlStar 正确安装后，从开始菜单中单击 ControlStar 图标启动软件，主界面如下图所示：

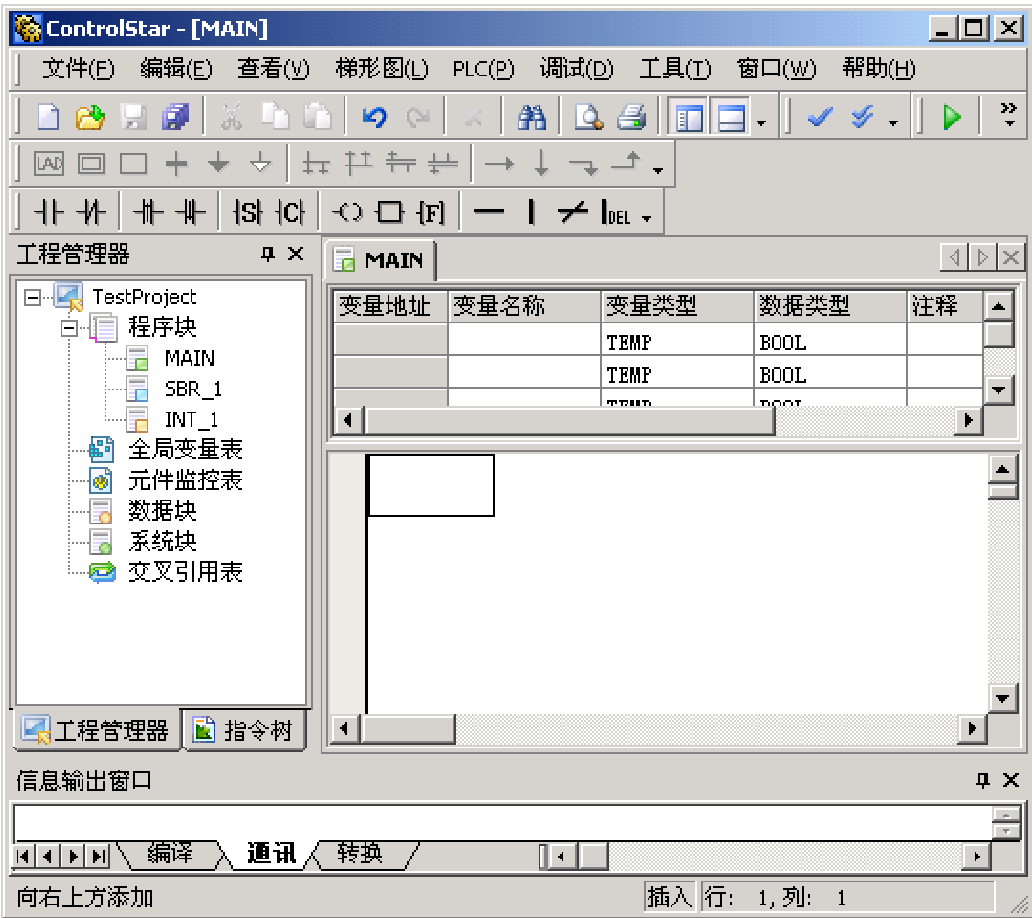


3.2 建立工程

启动编程环境后，首先需要为编写的程序创建一个工程。点击 “ 文件 ” 菜单下的 “ 新建工程 ” 菜单项，软件弹出下面的对话框：



在本示例中，选择 PLC 类型为 EC20，默认编辑器为梯形图，选择完毕点击对话框中的"确定"按钮后，一个新的工程被创建，并且默认打开了主程序进入程序编辑状态，如下图所示：

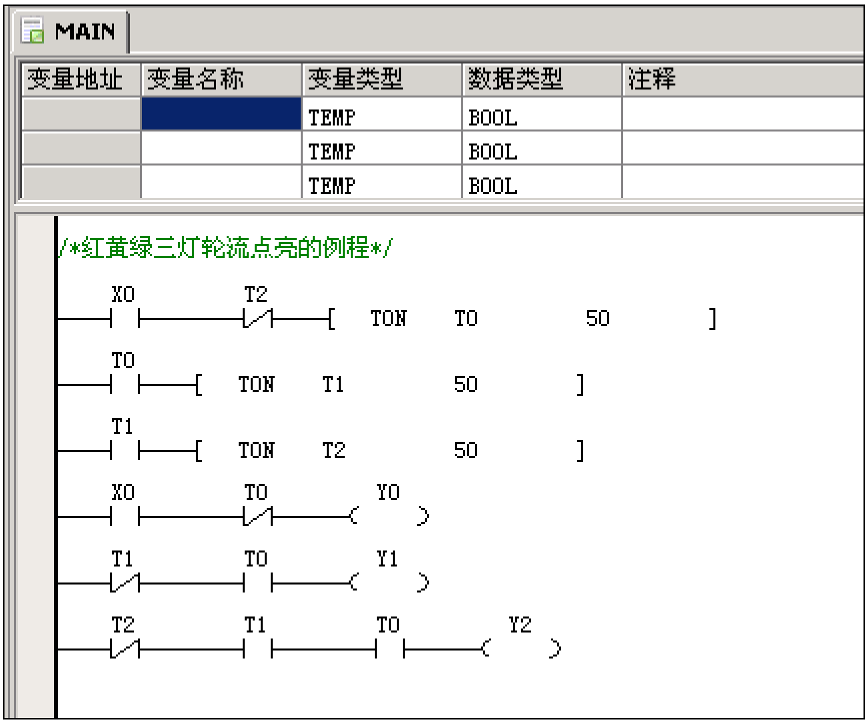


详细的工程管理功能操作，请参阅相关的章节。

3.3 编写控制程序

在本示例中，我们编写一个简单的跑马灯程序，假设输出端 Y0、Y1、Y2 分别控制红、黄、绿三种颜色的灯，输入端 X0 作为启动开关，该程序作如下控制：当开关合上时，红黄绿三盏灯交替点亮，每盏灯的点亮时间为 5 秒钟（使用精度为 100ms 的计时器 T0、T1、T2）；当开关打开时，三盏灯全灭。

1．在编程环境中输入如下的梯形图程序



2．程序编写完成后，可以先保存当前的工程到计算机中，点击“ 文件 ”菜单中的“ 保存工程 ”菜单项即可将当前编写的工程保存。

3．保存完成后，需要检查当前编写的程序是否有错误并且编译为可以下载到可编程控制器中的目标文件。点击“ PLC ”菜单中的“ 全部编译 ”菜单项，软件将当前程序进行全部编译，编译结果会显示在输出窗口中，如果没有错误的话，会显示如下内容：

3.4 建立通讯连接

在开始下载程序到可编程控制器硬件前，需要建立和可编程控制器的串口通讯连接，首先将串口线的两端连接到计算机和可编程控制器上，启动 ControlStar 应用程序，如果不能和 PLC 通讯或者需要使用不同的通讯速率，需要重新设置串口通讯参数。点击“工具”菜单中的“通讯设置”菜单项，会弹出如下所示的通讯设置窗口：

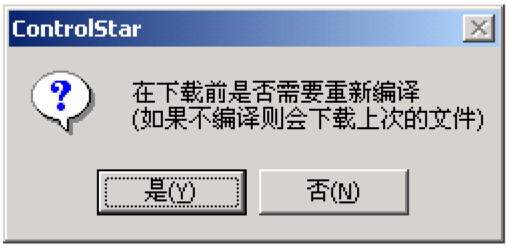
在串口下拉框中选择计算机实际连接的串口，选择通讯波特率后，点击“确定”按钮保存设置结果。

串口线连接和通讯设置完成后，可以测试计算机是否可以和可编程控制器正常通讯，点击“PLC”菜单的“PLC 信息”菜单项，如果通讯正常，会弹出一个窗口显示当前连接的可编程控制器的各种信息，如果多次使用本功能都提示“命令超时”，则说明串口连接或者设置可能不正确，请重新检查硬件连接和通讯设置内容。

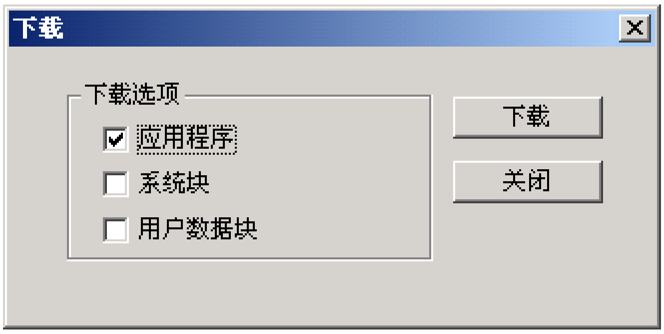
3.5 下载程序

点击“PLC”菜单中的“下载”菜单项，下载程序时，必须要求 PLC 处于停止状态，如果 PLC 正在运行，软件会提示如下的对话框：

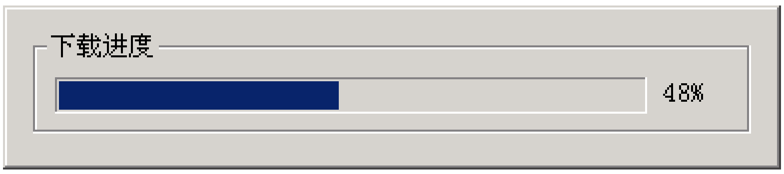
点击“确定”按钮停止 PLC 的运行，软件会弹出如下的编译提示对话框：



如果不重新编译下载上次的编译结果，可以选择“否”，否则选择“是”重新编译程序，编译完毕后，会出现下面的下载窗口：



根据需要，在“下载选项”中选择需要下载的内容，本示例只需要下载编写的应用程序，选中后点击“下载”按钮即可开始下载，下载过程中会显示进度条，完成后将会提示下载成功。



3.6 启动可编程控制器

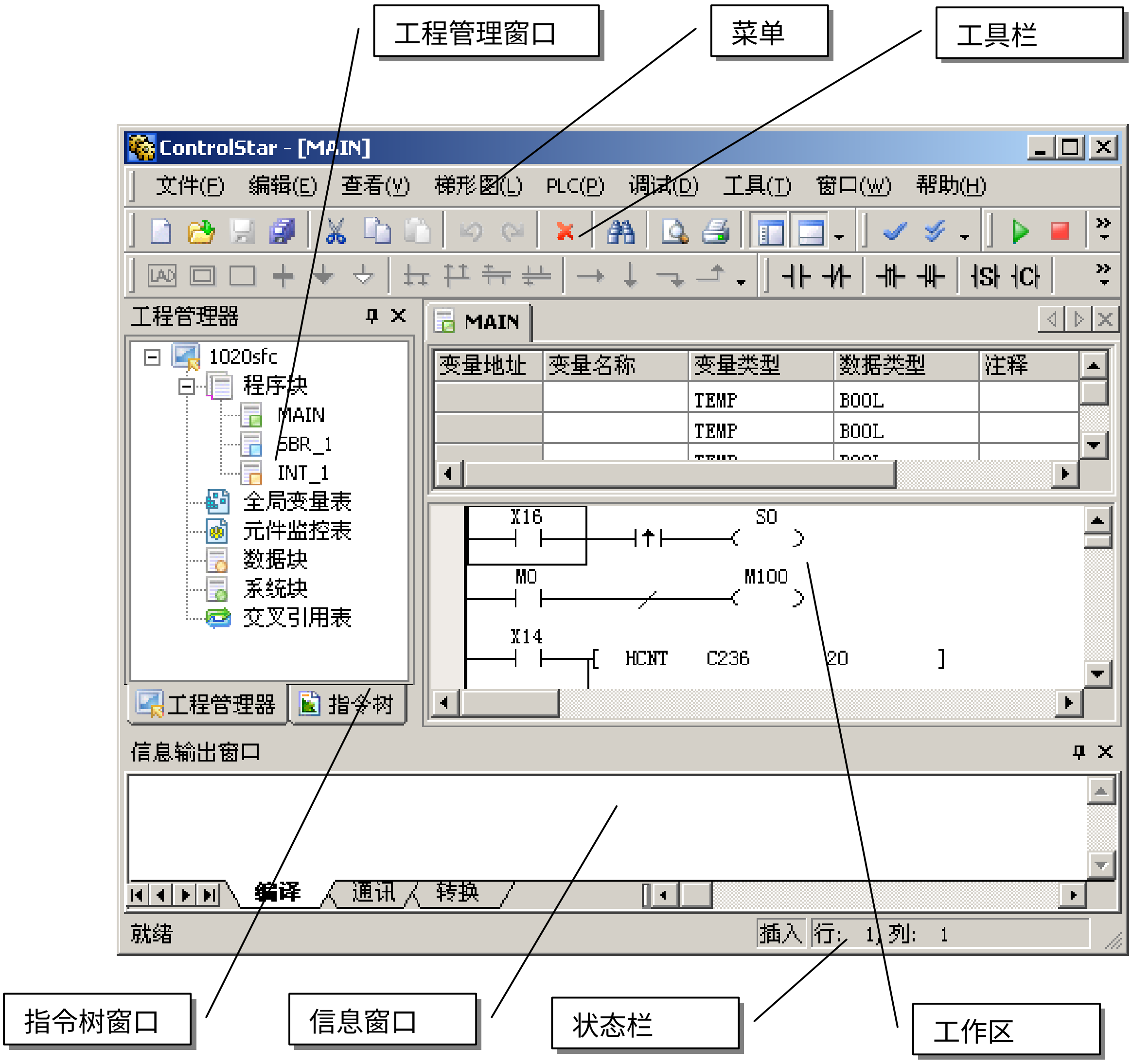
程序下载完成后，需要启动可编程控制器运行，在“PLC”菜单中选择“启动”菜单项，或者拨下可编程序控制器的开关到 RUN 处，在可编程控制器正常运行后，将接在输入点 X0 的开关闭合，就可以看到接在输出点 Y0、Y1、Y2 的三盏灯被轮流点亮。

第四章 编程说明

4.1 编程环境

4.1.1 概述

本程序的主界面基本包括七个部分：菜单、工具栏、工程管理器窗口、指令树窗口、消息窗口、状态栏和工作区。



4.1.2 菜单

菜单包括了一组含有全部命令的子菜单。

当鼠标停留在菜单项上时，菜单项功能的简短描述被显示在状态栏中。每个子菜单在后续章节中有更详细的应用介绍。

子菜单包括下列菜单项（注：这些子菜单的菜单项根据正在使用的程序功能会有一定的差别）。

文件	可用于“新建工程”、“打开工程”、“关闭工程”、“关闭文件”、“保存文件”、“保存工程”、“工程另存为”。它也包含用于打印、打印设置和打印预览的命令
编辑	包含了编辑所必需的全部命令，如编辑的撤销/恢复，插入、剪切和粘贴。另外，它提供了文本和图形程序的搜索和替换文本字符串功能
查看	提供了显示、隐藏不同窗口、工具条的功能。也提供了以不同语言查看当前程序的功能
PLC	提供了与 PLC 硬件互操作的相关功能
工具	提供了设置 ControlStar 相关属性和复杂指令的向导功能
窗口	提供了访问当前打开的窗口，以及重新排布当前窗口的功能
帮助	包含了用于调用帮助和显示关于对话框的所有命令

除前面所说的基本菜单之外，ControlStar 还会随着当前打开的窗口的不同而显示不同的菜单。

4.1.3 工具栏

本程序提供了几个工具栏，其中包含了用于更快地访问要经常使用的操作的不同命令按钮。这些操作也可通过使用菜单项或预定义的快捷键来完成。

工具栏位于菜单栏的下方。默认情况下，所有的工具栏都是可见的。要想隐藏或显示某一工具栏，请在任意一个工具栏上点击鼠标右键，在弹出的快捷菜单中选择/取消选择某个工具栏。

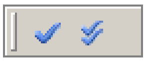
将鼠标光标放在任一图标上（而不点击它）停留片刻，会出现一个简短的描述文本，被称为工具提示。这些工具提示包括了当前图标的名称。如果工具提示没有出现，请在“选项”对话框中使能它们。

标准工具栏 

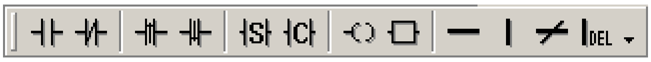
标准工具栏包含编辑 PLC 程序最常用的基本功能。

例如：

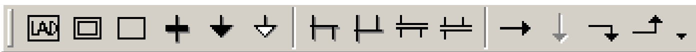
新建工程、打开工程、保存文件、保存全部，剪切、拷贝、粘贴，撤销/恢复，删除，查找，打印预览、打印，显示/隐藏工程管理器、显示/隐藏信息输出窗口。

编译工具栏 

编译工具栏包含编译当前程序和全部编译两个编译最常用的功能：编译当前窗口中的程序/编译所有程序。

梯形图工具栏 

梯形图工具栏包含编辑梯形图程序最常用的功能。

顺序功能图工具栏 

顺序功能图工具栏包含编辑顺序功能图程序最常用的功能。

标签工具栏 

标签工具栏主要应用在指令列表、数据块两个文本编辑器中，功能是在文本窗口中使用可供快速定位的标签。

PLC 工具栏 

PLC 工具栏提供了操作、访问 PLC 硬件最常用的功能。

每个工具栏的功能都可以在界面菜单中找到对应的菜单项。可以根据需要自定义工具栏，也可以添加新的工具栏，加入常用的图标。

缩放工具栏 

在梯形图程序和顺序功能图程序中，可以通过缩放工具栏的调整显示比例。

4.1.4 快捷键

通用快捷键

快捷键	功能
CTRL + N	新建工程
CTRL + O	打开工程
CTRL + A	全选

快捷键	功能
CTRL + C	拷贝
CTRL + X	剪切
CTRL + V	粘贴
CTRL + F	查找
CTRL + H	替换
CTRL + G	定位
CTRL + Z	撤销
CTRL + Y	恢复
CTRL + S	保存
CTRL + P	打印
Delete	删除所选内容
F1	打开联机帮助
F5	PLC 运行
F6	PLC 停止
F7	编译所有文档
CTRL + F7	编译当前文档
F8	下载
F9	监控

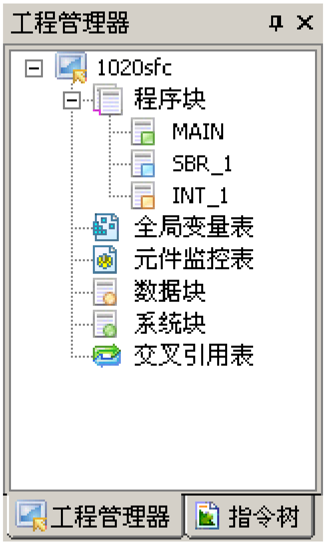
梯形图编辑快捷键

快捷键	功能
CTRL + 1	常开触点
CTRL + 2	常闭触点
CTRL + 3	正缘触点
CTRL + 4	负缘触点
CTRL + 5	步进点
CTRL + 6	比较触点
CTRL + 7	线圈
CTRL + 8	应用指令
CTRL + 9	子程序调用
CTRL + B	插入块注释
CTRL + ↓	竖线
CTRL + SHIFT + ↓	删除竖线
CTRL + →	横线
CTRL + I	插入行
CTRL + L	删除行

顺序功能图编辑快捷键

快捷键	功能
CTRL + 1	梯形图块
CTRL + 2	初始步进符
CTRL + 3	普通步进符
CTRL + 4	转移符
CTRL + 5	跳转符
CTRL + 6	重置符
CTRL + 7	选择分支
CTRL + 8	选择汇合
CTRL + 9	并行分支
CTRL + 0	并行汇合
O	打开内置梯形图
CTRL + ↓	竖线
CTRL + →	横线
CTRL + I	插入行
CTRL + L	删除行

4.1.5 工程管理器窗口



工程管理器窗口中使用一个工程树显示当前工程的名称及工程中所包含的所有文档，其中程序块下的程序节点可以根据实际需求添加或删除（主程序不能删除），而“全局变量表”、“元件监控表”、“数据块”、“系统块”、“交叉引用表”等节点是固定的。

为了能够明显地区分主程序、子程序和中断子程序，在工程树中每一类程序节点对应不同颜色的图标，主程序显示为，子程序显示为，中断子程序显示为。

通过工程树可以实现以下功能：

1．右键单击工程名节点，在弹出菜单中可以选择保存工程、另存工程、关闭工程或者修改工程属性等；

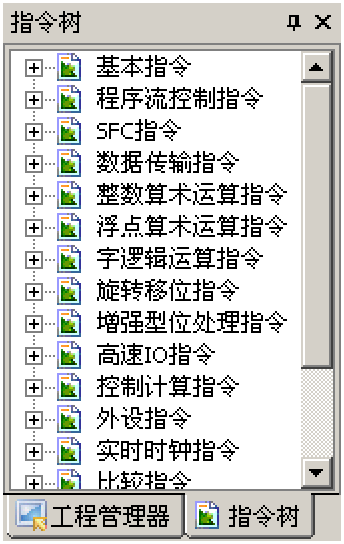
2．右键单击程序块节点，在弹出菜单中可以选择插入子程序或中断子程序；

3．右键单击程序块下的某个程序节点，在弹出菜单中可以选择打开该程序、删除该程序（主程序不能删除）或者修改其属性；

4．右键单击其他节点，在弹出菜单中只能进行打开操作；

4.1.6 指令树窗口

指令树窗口如下所示：



指令树的内容会随着当前工作区窗口的编辑器的改变而变化，如果当前工作区为梯形图编辑器，指令树将显示梯形图语言支持的所有指令，同样，如果当前工作区为指令表编辑器或者顺序功能图编辑器，指令树也显示为指令表语言或者顺序功能图语言支持的所有指令，因此在编写程序过程中，可以通过指令树查找与某个功能匹配的指令。在编程过程中，指令树不仅可以用于查找指令，还可以用于输入指令。

如何通过指令树输入指令

选中要输入的指令，并按下鼠标左键。拖动鼠标。选中的指令将跟随鼠标移动而移动，鼠标在拖动过程中，其形状可能会发生改变，当鼠标的形状变为“箭头”时表示在该位置可以输入该指令。

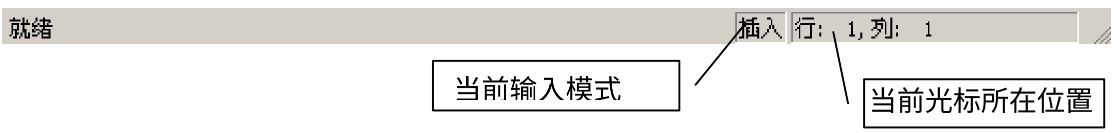
弹起鼠标左键，如果当前编辑器是梯形图编辑器，将启动输入指令的向导，系统会自动检测该指令所需的参数数量及格式，然后自动调整界面和输入格式，详情请参见梯形图编辑的向导式输入。

如果当前编辑器是指令列表或者顺序功能图，编辑器将在鼠标弹起的位置插入该指令。

4.1.7 信息输出窗口

信息输出窗口用于输出软件执行操作之后的结果。包括：编译、通讯和转换三种操作的执行结果信息。

4.1.8 状态栏



状态栏用于提供常用的属性信息。

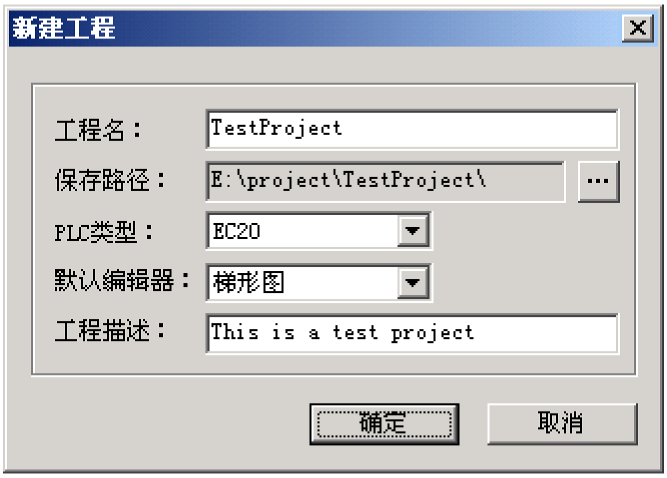
4.1.9 工作区

工作区包括：程序编辑窗口、全局变量表编辑窗口、元件状态监控表窗口、数据块窗口和交叉引用表窗口。

4.2 工程使用说明

在使用 Controlstar 开发程序的过程中，需要编写或者用到程序块、全局变量表、元件监控表、数据块、系统块、元件交叉引用表等各种类型的文件，ControlStar 使用工程对各种类型的文件进行管理。

工程文件的扩展名为 proj，工程有如下属性，可以在新建工程对话框中对工程属性进行选择或者输入：



工程名：工程标识，同时也是工程保存时对应的磁盘文件名。

保存路径：指定工程文件保存的全路径，所指定的路径加上工程名称子目录（自动创建）就是最终的工程文件存放目录。

PLC 类型：表示工程所针对的 PLC 的类型，选择不同的 PLC 类型/版本对下面的内容有影响：CPU 指令集，系统块配置所支持的程序容量其中程序容量会影响到编译和下载过程，编译程序时要做容量检查，如果程序步数超过设定值，给出警告信息，在下载时给出错误信息。

默认编辑器：编辑器类型可选择为：梯形图、指令列表、顺序功能图三种类型。

工程描述：工程的简单说明，工程描述可为空，最长 128 个字符。

新建工程时的默认 PLC 类型和默认编辑器类型可以在系统选项对话框中指定，如下图：

对应每个工程一般有 CompMonTbl.CMT，CrossRefTbl.CRS，SysCfgInfo.SSC，DatBlock.DB 四个文件，这四个文件分别保存状态监控表，交叉引用表，系统配置表，以及数据块信息，他们都保存在 DATA 目录下面。

在 POU 目录下保存着程序单元，包括主程序、子程序、中断子程序等等，选择的编辑器类型不同，生成的程序单元的类型也不同，指令列表程序、梯形图程序、顺序功能图程序的扩展名分别为*.IL、*.LD、*.SFC。

全局变量表信息信息一般也保存在该目录下，全局变量表的文件名为 GlobalVarTbl.GVT。

ControlStar 使用如下的工程管理器管理一个工程：

📖 注意

请不要在 Windows 中直接删除或者修改上面描述的文件，否则可能导致工程无法打开或者程序数据被破坏。

4.3 子程序和调用方式

4.3.1 程序块的分类和作用

软件提供主程序、子程序和中断子程序三种不同的程序单元。

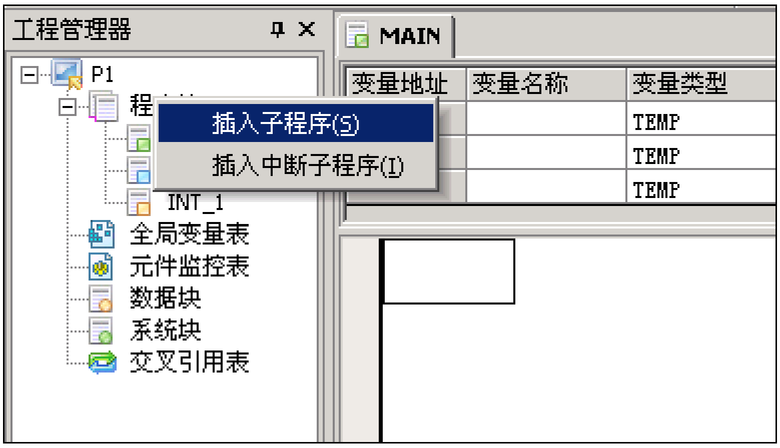
主程序：主程序只能有一个，由软件默认提供，主程序是 PLC 应用程序启动执行的程序。主程序可以由梯形图、顺序功能图或者指令列表语言中的任何一种编写。

子程序：一个工程内可以有多个子程序，不能超过 64 个，子程序由主程序或者其他子程序调用，主要完成一些常用或者需要重复使用的功能，子程序只能由梯形图或者指令列表编写，不能用顺序功能图编写。

中断子程序：可以有多个，通过和某个中断事件相关联，在中断事件发生时调用。中断子程序不能超过 40 个。中断子程序只能由梯形图或者指令列表编写，不能用顺序功能图编写。

使用子程序的目的是为了对程序进行分段和分块，把一些常用的功能模块写成子程序后可以进行重复调用，通过使用较小的程序块，只在需要时才调用程序块，可以更有效地使用 PLC，因为所有的程序块可能无须执行每次扫描，当主程序调用子程序并执行时，子程序执行全部指令直至结束。然后，系统将控制权返回至调用子程序网络中的主程序。

4.3.2 如何建立子程序或中断子程序



右键单击“工程管理器”窗口的程序块，选择插入子程序或者插入中断子程序，新的子程序名的默认名称是 SBR_*，新的中断子程序名的默认名称是 INT_*（其中的*

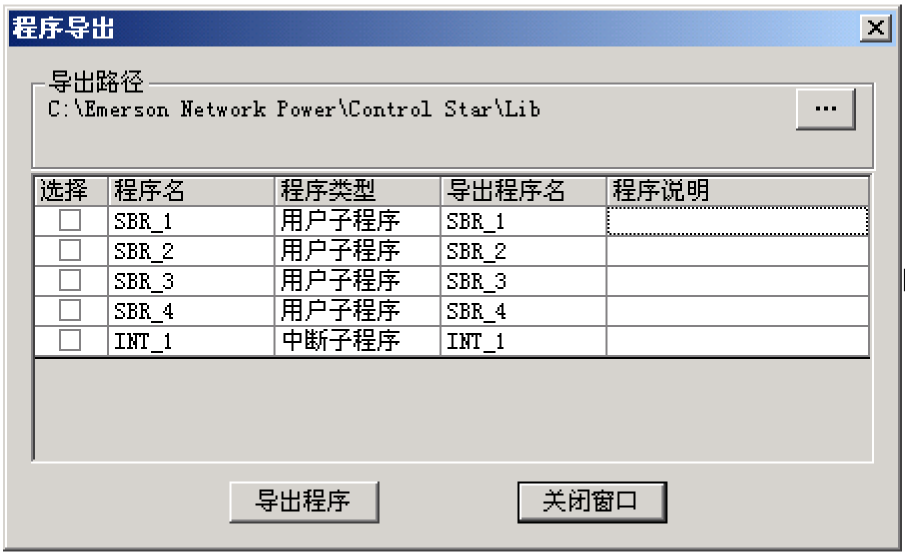
是由软件自动计算的一个数字），建立完成后可以通过子程序属性对话框将默认程序名改为一个更有意义的名称。

在工程树上插入新程序节点的同时，该程序也会被打开，可以马上对它进行编辑。

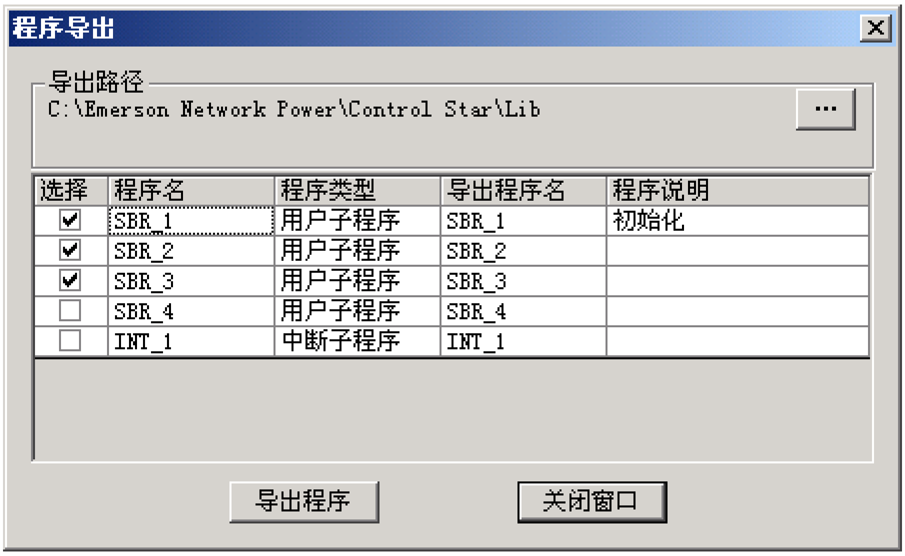
4.3.2 导出子程序

软件提供了子程序库的导入和导出功能，对于某个通用功能，可以在子程序中编写完成后，通过导入/导出功能，在不同的工程中再次使用，避免相同功能的重复代码编写。ControlStar 使用程序库文件的方式来管理这些程序，通用的子程序首先可以导出到某个目录形成可以重复使用的程序库文件，在需要使用这些子程序的工程中，使用导入功能可以将程序库文件导入到当前工程中作为标准的子程序使用。

选择菜单“文件”下的“程序文件导出…”菜单项，打开程序导出对话框：



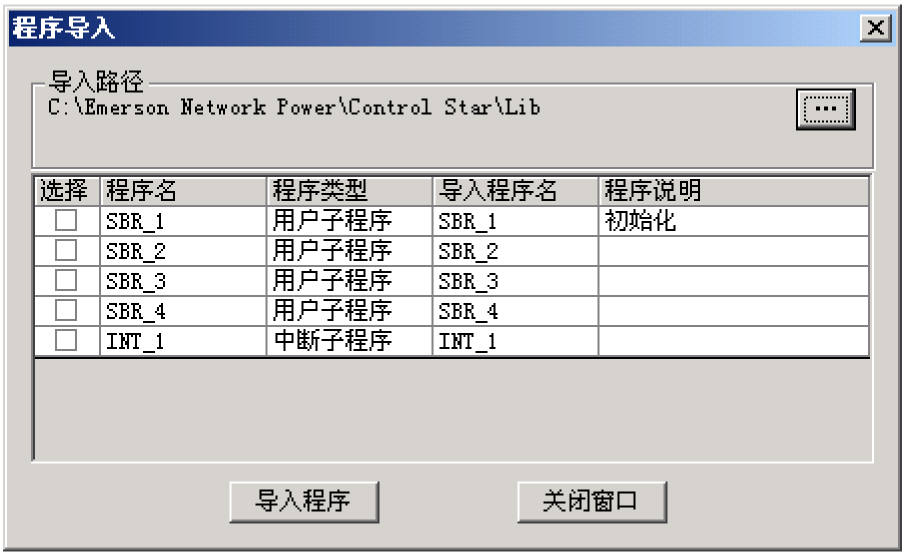
对话框中自动把该工程中可以导出的程序列出来，程序导出的路径第一次使用时默认为软件安装路径下的\Lib\目录下，可以点击“…”按钮来更改，更改后路径自动成为下次默认的导入/导出路径。



可以编辑程序的导出名以及程序说明。最后在需要导出的程序前打勾，然后点击“导出程序”按钮。程序正确导出后该程序前的勾自动消失，否则仍然处于导出前的状态。

4.3.3 导入子程序

选择菜单“文件”下的“程序文件导出…”菜单项，打开程序导出对话框：



对话框中自动列出了在默认导入导出路径下可以导入的程序信息，可以点击“ ... ”按钮来更改路径。导入前可以编辑程序导入工程后的名称，如果所指定的程序名不合法或者与工程中已有的程序重名时，该程序导入不能成功，并且当导入程序名不合法时，在列表中用红颜色加下划线来提示。选择好程序后，点击“ 导入程序 ”按钮，程序导入到工程。

4.3.4 编辑子程序

从工程树中双击某个子程序节点，可以打开该程序。程序编辑区域由两个部分组成，上面为局部变量表，下面为程序编辑窗口。如果需要为子程序指定参数，可以使用该子程序的局部变量表定义参数，每个子程序调用的最大输入／输出参数限制为 16。

4.3.5 调用子程序

在子程序局部变量表中定义参数（如果有），参数必须有一个符号名、一个变量类型和一个数据类型。

局部变量表中的变量类型域定义参数的类型：输入参数（IN）、输入输出参数（IN_OUT）或输出参数（OUT）。需要增加参数数量时，从右键菜单中选择“ 插入行 ”，在当前行的上方会插入一个新行，新插入行的变量类型和当前行相同。

调用参数类型

IN：如果调用子程序时输入参数是元件，元件中包含的值被传入子程序的局部变量表中对应的变量中；如果参数是数据常量（例如 16#1234），常量数值被输入子程序。

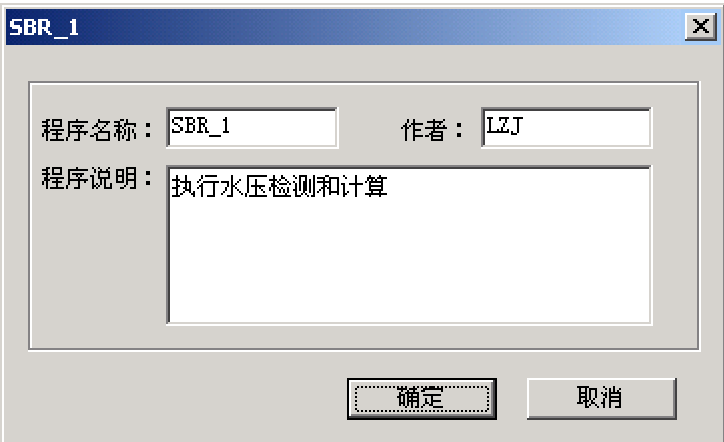
IN_OUT：调用该子程序时，位于指定参数位置的数值被输入子程序中对应的局部变量，来自子程序的结果数值最后通过相同的局部变量返回至调用时作为输入参数的元件中。

OUT：来自子程序的结果数值被返回至指定的参数位置。

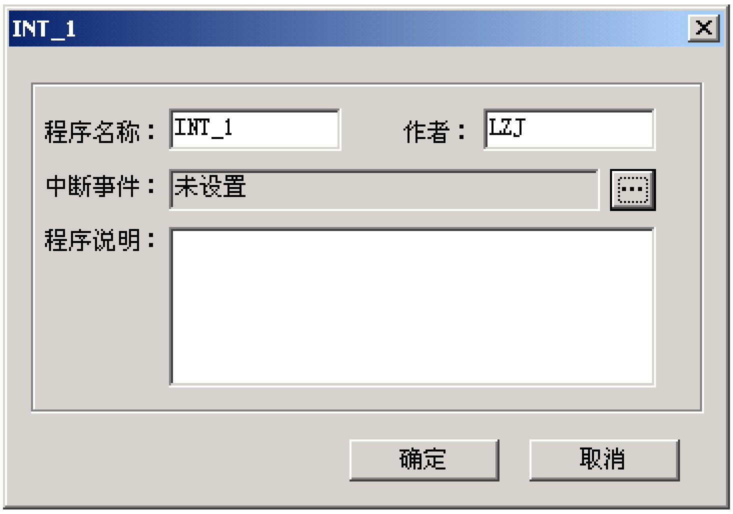
TEMP：未用作交接参数的任何本地内存，每个循环周期重复进入子程序时，TEMP 型的变量都会被重新初始化，上次的计算结果不会保留。

4.3.6 修改子程序（中断子程序）属性

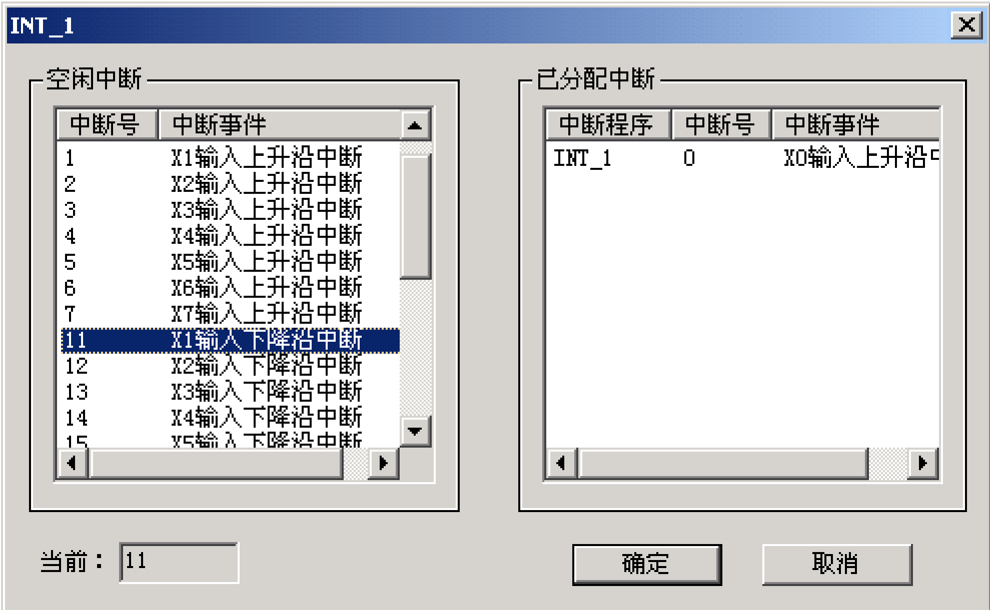
在工程树上选中要修改的子程序节点，从右键菜单中选择[属性(P)]，可以打开文件属性对话框，在子程序属性对话框中可以修改该子程序的程序名称、作者及程序说明，如下图：



对于中断子程序，可以为其指定中断号（新建的中断子程序默认中断号为-1，表示未设置），中断子程序属性对话框如下：



点击“…”按钮，会弹出中断分配对话框，在这里显示当前可用的空闲中断以及已经被其他中断子程序使用的中断，可以在空闲中断中选择一个中断号指定给当前中断子程序，如下图：



由于系统中的某些中断事件是相互冲突的，比如 X0 上升沿和 X0 下降沿是不能同时被系统响应的，因此如果 X0 上升沿中断被分配后，X0 下降沿中断就不能再被使用了，在这种情况下，X0 下降沿中断会在空闲中断中被屏蔽掉。相互冲突的中断事件如下：

X0 输入上升沿中断（中断号：0）<——> X0 输入下降沿中断（中断号：10）

X1 输入上升沿中断（中断号：1）<——> X1 输入下降沿中断（中断号：11）

X2 输入上升沿中断（中断号：2）<——> X2 输入下降沿中断（中断号：12）

X3 输入上升沿中断（中断号：3）<——> X3 输入下降沿中断（中断号：13）

X4 输入上升沿中断（中断号：4）<——> X4 输入下降沿中断（中断号：14）

X5 输入上升沿中断（中断号：5）<——> X5 输入下降沿中断（中断号：15）

X6 输入上升沿中断（中断号：6）<——> X6 输入下降沿中断（中断号：16）

X7 输入上升沿中断（中断号：7）<——> X7 输入下降沿中断（中断号：17）

使用子程序的几个规则：

1．子程序可以嵌套使用（在某个子程序中调用另外一个子程序），最大嵌套深度为 6。不能从中断程序调用子程序。

2．子程序之间禁止显式循环调用，如子程序 A 和 B 互相调用对方。

3．子程序禁止递归调用，例如程序 A 调用程序 B，程序 B 调用程序 C，程序 C 再调用程序 A，形成一个环。此外子程序不允许调用本身。

4.4 数据类型说明

系统支持 BOOL、WORD、DWORD、INT、DINT、REAL 六种数据类型，每种数据类型说明如下。

数据类型	描述	位	范围
BOOL	布尔	1	ON、OFF
WORD	字	16	0 至 65535
DWORD	双字	32	0 至 4294967295
INT	整数	16	-32768 至+32767
DINT	双整数	32	-2147483648 至+2147483647
REAL	实数	32	±1.175494351 E - 38 ～ ±3.402823466 E + 38

4.5 使用联机帮助

ControlStar 软件提供了丰富的联机帮助功能，在软件使用过程中遇到的问题，应该首先使用联机帮助寻求解答。可以在开始菜单的“ ControlStar ”程序组或者在软件的任何窗口中选择“ 帮助→帮助主题 ”菜单，打开联机帮助。

为了能够快速准确的定位帮助信息，软件还提供了上下文相关的帮助功能：

在工作区中按“ F1 ”键，软件会根据当前窗口的类型（比如梯形图编辑窗口、顺序工程图编辑窗口或者全局变量表编辑窗口）打开与之相对应的文档类型的帮助；

在使用某个对话框时，按“ F1 ”键可以打开帮助文档并自动定位到针对此对话框的帮助部分；

在编译、转换或者通讯过程中产生的任何错误信息，可以在信息输出窗口点击选中该信息，然后按“ F1 ”键可以打开针对该错误的原因描述和解决方案。

第五章 程序编辑

ControlStar 系统提供了梯形图、指令表、顺序功能图三种程序编辑器；为了定义和使用全局变量和局部变量，ControlStar 系统还提供了全局变量表和局部变量表编辑器；此外系统还提供交叉引用表以用于记录元件在各个程序中的使用情况。

为了使编写程序更加方便和快速，ControlStar 系统提供了一些通用的编辑功能，合理使用这些编辑功能将会更加高效的使用 ConstrolStar 系统编写程序。

5.1 通用编辑功能

5.1.1 如何进行查找、替换和定位

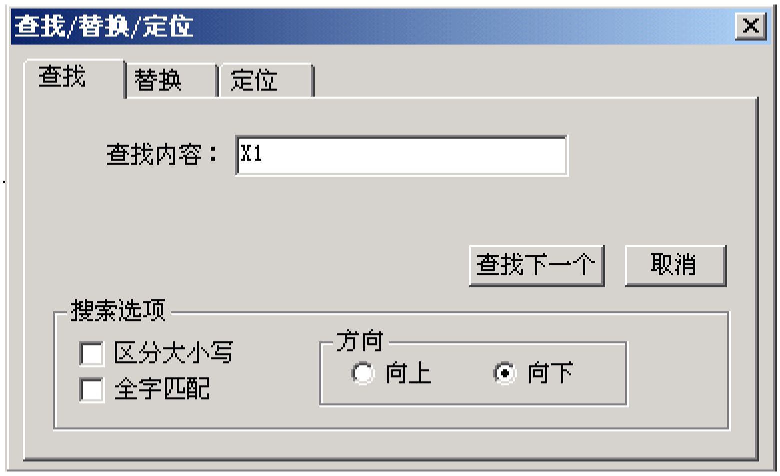
查找

选择 [编辑/查找] 菜单或按 CTRL+F 键，可以激活查找对话框，然后可以在当前编辑器中进行查找操作。查找功能可以查找指定的字符串，包括元件地址、变量符号等，前提是目的字符串必须在当前编辑器中有显示。对于梯形图编辑器，块注释中的文字不在搜索范围内。

查找功能支持全字匹配、大小写匹配、查找方向等选项。无论选择向上或向下查找，在查找到一端结束后，都会从另一端重新开始。

对于指令表编辑器，顺序功能图编辑器、以及各个表格将查找到的目标的背景色显示为深色，对于梯形图编辑器则将查找到的目标会用虚线方框标识出来。

查找对话框如下所示：

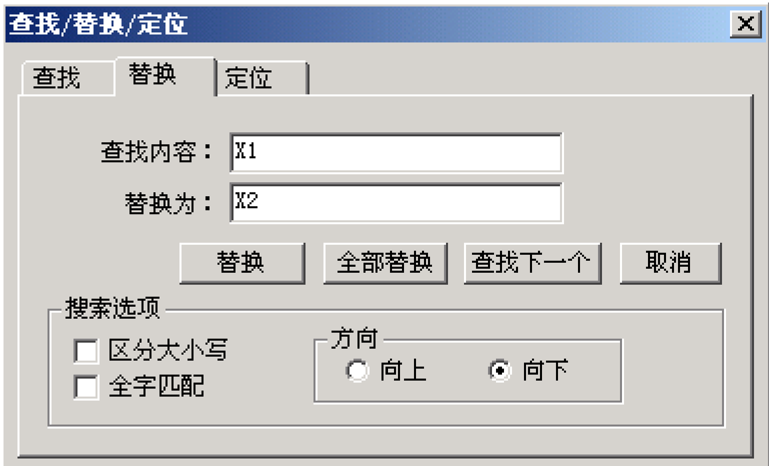


替换

选择 [编辑/替换] 菜单或按 CTRL+H 键，可以激活替换对话框，然后可以在当前编辑器中进行替换操作。

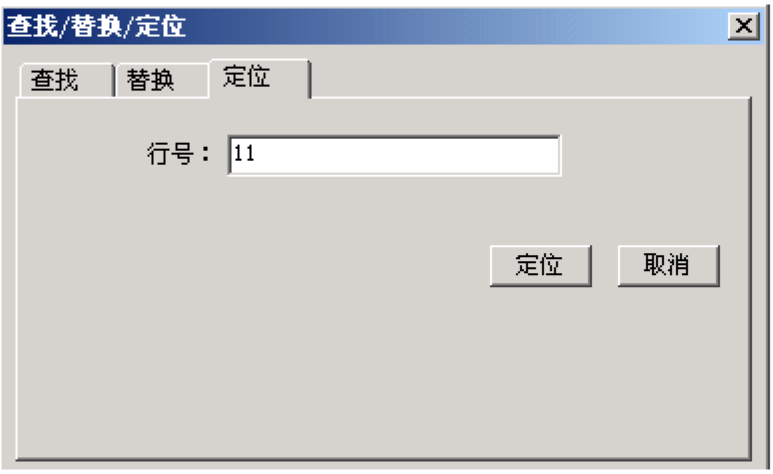
执行替换功能时，要求查找内容和替换内容都不能为空，而且二者不能完全相同。替换后的元件同样要进行正确性校验，如果检验失败将不会执行替换。在替换过程中，可以选择逐个替换或者全部替换，如果选择逐个替换，替换失败时将会提示详细的失败原因，但是在全部替换时则不会。

替换功能支持与查找功能相同的选项。替换对话框如下图：



定位

选择 [编辑/定位] 菜单或按 CTRL+G 键，可以激活定位对话框，使用定位功能可以快速定位到当前编辑器中的某一行。定位对话框如下图所示：



5.1.2 如何进行复制、剪切和粘贴

复制

选中某个单元格或者拖动鼠标选中某个区域后，使用 [编辑/复制] 菜单或者按 CTRL+C 键，可以将所选的元件拷贝到系统剪贴板。

剪切

选中某个单元格或者拖动鼠标选中某个区域后，使用 [编辑/剪切] 菜单或者按 CTRL+X 键，可以将所选的元件拷贝到系统剪贴板，同时清除选中的元件。

粘贴

使用 [编辑/粘贴] 菜单或者按 CTRL+V 键可以将剪切板上内容粘贴到当前编辑器的当前位置中。

提示

1．对于梯形图编辑器，只有当某元件所占用的单元格的最左边的一格在选择区域内时，该元件才会被拷贝；

2．对于梯形图编辑器，当复制的区域内有竖线时，只有当该竖线所跨过的上下单元格都在选择范围内时，该竖线才会被拷贝到剪贴板中；

3．无论当前使用哪种编辑模式，执行粘贴功能时一律使用覆盖模式处理，即先将要粘贴的范围内的元件全部清除，然后再把剪贴板中的元件逐个粘贴上来；

4．不同编辑器之间不能被粘贴。

5.1.3 如何进行插入行、删除行或者添加行

插入行

使用行插入功能可以在编辑器中插入一空行，需要使用此功能时，先在要插入的行上选中任一单元格，然后从右键菜单中选择 [行插入]，系统会将当前行及下面所有行的元件向下移动一行，从而在当前行腾出一个空行出来。

请注意，对于梯形图程序如果当前行的元件和上面一行的元件有竖线相连的话，竖线也会被自动拉伸以保持原有连接关系不变。如果当前梯形图中已经达到 4000 行，则不能再插入行。

删除行

使用行删除功能可以删除当前编辑器中的某一行，需要使用此功能时，先在要删除的行上选中任一单元格，然后从右键菜单中选择 [行删除]，系统会将当前行清空，并将下面所有行的元件向上移动一行。

添加行

添加行指在表格的末尾添加一个新行，在全局变量表、局部变量表、元件状态监控表中提供了该功能，可以通过从右键菜单中选择[添加行]，也可以直接按键盘的[↓]键为当前表格添加一个新行。

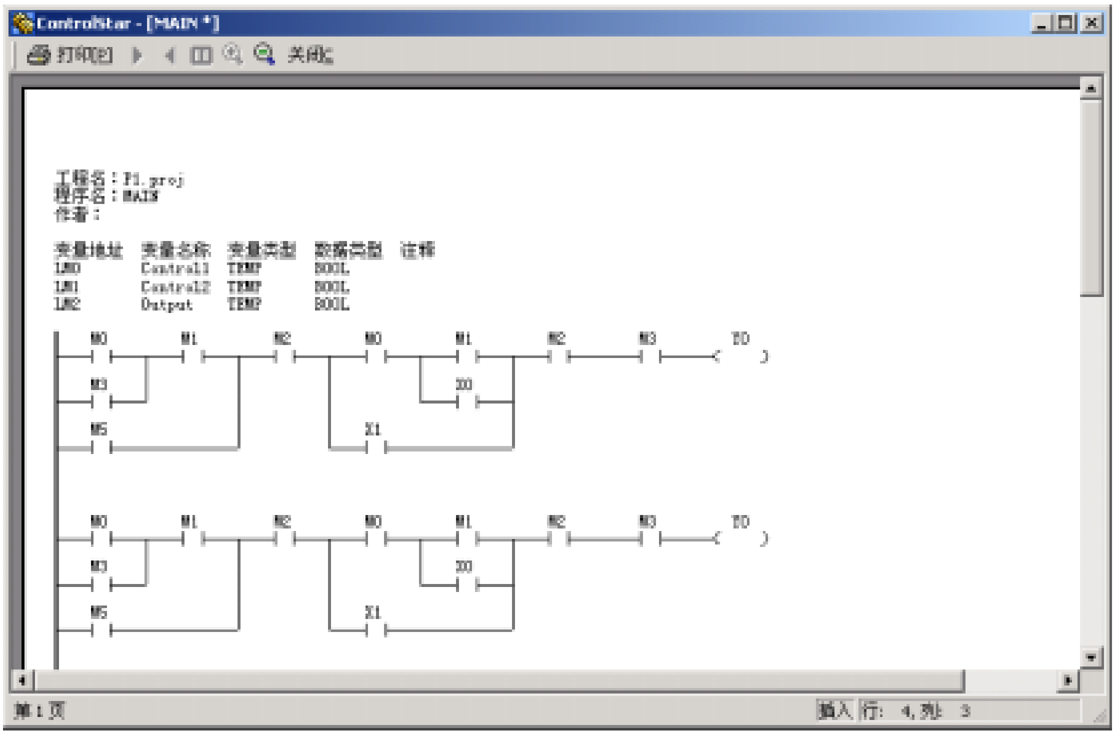
5.1.4 如何进行撤销恢复

在程序编辑过程中可以单击常用工具栏的“撤销”按钮或者编辑菜单下的“撤销”菜单撤销误操作，最多可撤销最近的 1000 次操作；也可以单击“恢复”按钮或者选择编辑菜单下的“恢复”菜单恢复最近一次撤销的操作，最多可以恢复最近 1000 次撤销操作。

5.1.5 如何进行打印预览、打印

打印预览

打印之前，可以在屏幕上进行打印预览，点击 [文件/打印预览] 菜单或者工具栏上的按钮，可以打开打印预览窗口，如下：



在打印预览窗口中的预览工具栏的功能如下：

打印[P]：打印当前程序到打印机上

▶：查看下一页

◀：查看上一页

：每屏双页显示

：放大显示比例

：缩小显示比例

关闭C：关闭预览窗口

打印

需要打印时，请点击工具栏上的 按钮，或在打印预览窗口中点击 打印[P] 按钮，首先弹出标准打印对话框，可以在此设置打印选项，确定后即可将当前程序输出到打印机。

📖 提示

1．梯形图和顺序功能图打印时，如果某一行因太长而一页打印不下的话，系统会在后面的纸上继续打印该行的剩余部分，直到该行完全被打印出来；

2．梯形图程序打印时，如果某一页打印的最后一行有竖线的话，则其后面一页的顶行会将这些竖线重复打印一遍以示连接关系；

5.1.6 如何检查程序错误

当完成程序的编辑后，点击工具栏上的“编译”按钮（），可以检查当前程序中是否有错误，如果没有错误，则输出窗口中显示编译成功信息；输出窗口如下图所示：

如果有错误，则提示错误信息和错误位置，双击该错误信息将定位到错误处；输出窗口如下图所示：

5.2 梯形图编辑器

5.2.1 梯形图的工作原理及逻辑组成

梯形图是一种与电气中继图相似的图形语言。一个梯形图程序是由若干个逻辑网络组成的，网络又是由若干个相互连接的图形组件组成的，这些图形组件是构成梯形图程序的最基本元素，大致分为以下几种类型：

1．触点（）：代表电气图中的开关。电流仅在触点关闭时通过常开触点，或在触点打开时通过常闭触点。

2．线圈（ ）：代表由功率流充电的中继或输出。

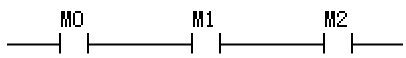
3．功能块（ ）：又称应用指令，代表当功率流到达时执行的一项功能（例如，数据传输、数据运算、计时器、计数器等）。

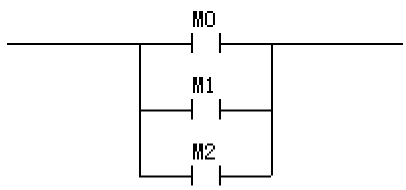
4．连接线（ ）：代表电气图中的导线，用于导通彼此相连的其他元件。

网络由以上元素组成并代表一个完整的线路。电源从左边的电源杆流过（在梯形图编辑器中由窗口左边的一条垂直线代表）闭合触点，为线圈或功能指令充电。

5.2.2 连接关系

与电气图类似，在梯形图中，各种元件或由元件组成的块之间有两种关系：串联和并联，分别如下图所示：





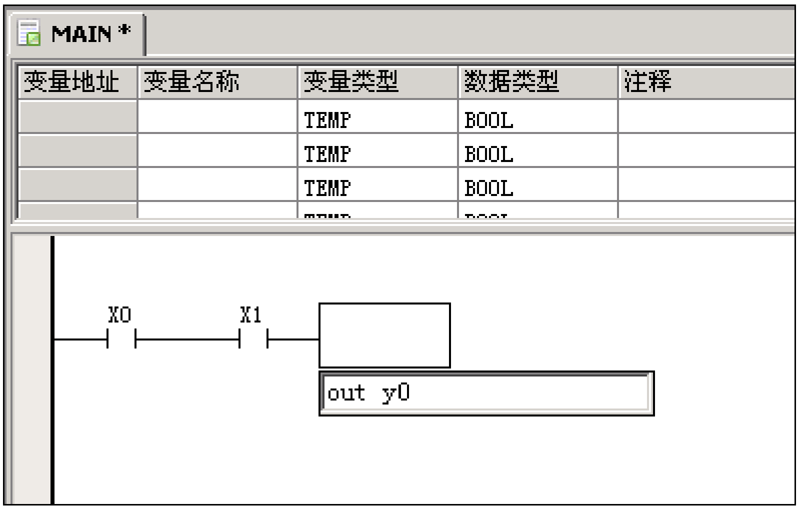
5.2.3 如何输入梯形图元素

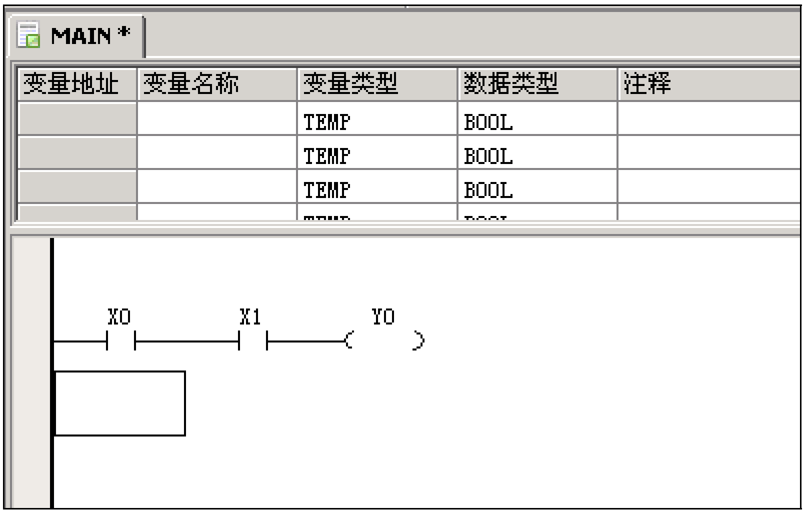
在梯形图中，整个输入区域被自动划分为许多个隐式的单元格，可以通过鼠标点击或键盘的方向键来选择单元格，选中的单元格会被黑框包围以提示当前位置。在选定单元格后，才可以在相应位置输入元件，元件根据自身大小不同，可能占用一个或多个单元格。

梯形图中有两种基本的元件输入方法，以下分别介绍。

直接输入

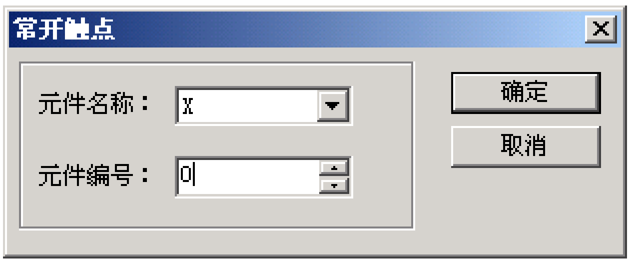
在选中单元格处，直接用键盘输入字母，指令输入对话框即自动弹出，在此对话框中输入完整的指令和参数后，点击确认按钮或回车，软件会对所输入的指令进行正确性校验，校验无误后即生成该指令所代表的图形元件。

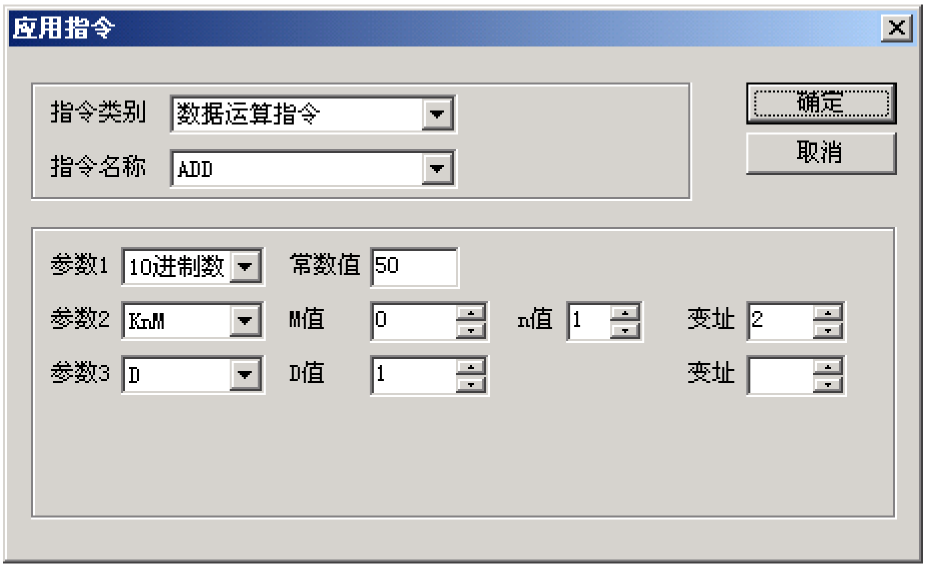




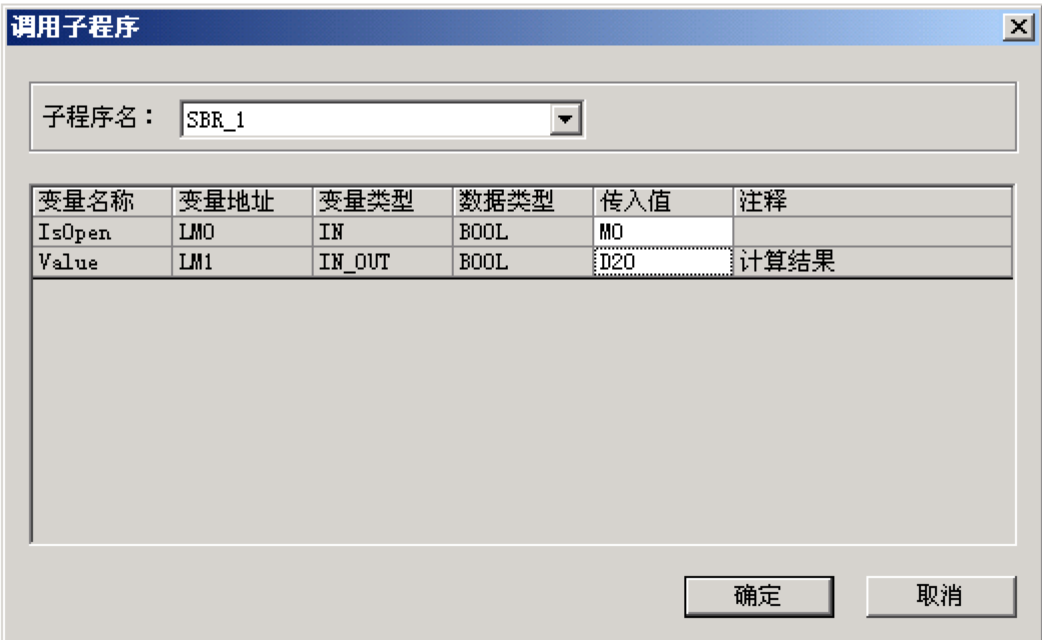
向导式输入

在选中单元格后，还可以通过选择菜单或工具栏图标来输入相应的元件。所谓向导式，就是在选择了元件类型或指令名称后，软件会自动检测所需的参数数量及格式，然后自动调整界面输入格式，如下图：





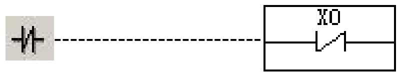
由于 CALL 指令的比较特殊，它的参数数量是可变的，而且只有选择了调用的子程序名称（CALL 指令的第一个参数）后才能确定其余的参数，因此针对 CALL 指令单独设计了另外一种对话框，如下图：



在此对话框中，选择子程序名后，下面的表格中会显示该子程序所有的输入/输入输出/输出型参数的名称、地址、数据类型等信息，可以在“传入值”域输入想要传入的参数。确定后，如果输入无误，就会生成一个 CALL 指令元件，此元件固定占用两个单元格，如果有参数显示不下的，会以省略号表示，当把鼠标移动到该元件上时，会有浮动框显示完整指令。

各工具栏图标所对应的元件类型列举如下：

常开触点



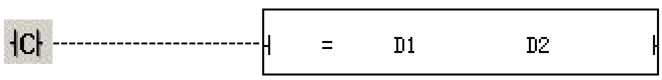
常闭触点

正缘触点

负缘触点

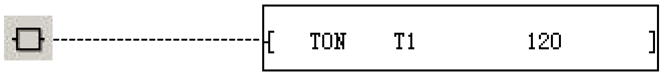
反转触点

步进触点

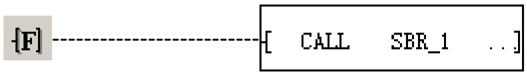


比较触点

线圈



应用指令



CALL 指令

横线

竖线

限制条件

1．某些指令只能在指令列表编辑器中输入，而不能在梯形图中输入，比如 MPS、MPP 等。当在梯形图中直接输入这些指令时，将不会生成任何元件；

2．由于线圈和应用指令代表了梯形图一行的结束，所以在这些元件后面，不能够再输入其他任何元件；

3．由于竖线会占用两行的空间，因此在输入竖线时，如果在下一行的相应位置刚好是其他元件的中间部分，则此竖线不能输入；

4．梯形图中最大输入列数为 32 列，最大输入行数为 4000 行；

🕮 提示

1．在梯形图中，按“ Ctrl + 右 ”键可直接输入横线，按“ Ctrl + 下 ”键可直接输入竖线，按“ Ctrl + Shift + 下 ”键可以删除竖线。

2．元件输入有两种模式，分别为覆盖和插入模式。在覆盖模式下，新输入的元件会覆盖它所占用的单元格内的所有其他元件；而在插入模式下，原位置的元件及其相连的其他元件会先被向右移动若干个单元格，直至给新输入的元件腾出足够的空间。可以使用键盘上的 [Insert] 键在这两种模式之间切换。

5.2.4 如何修改梯形图元件

如何修改

需要修改梯形图中已经存在的元件时，用鼠标双击要修改的元件或者在该元件处按回车键，就会弹出直接键盘输入对话框，并将所选元件对应的指令显示在输入框中，然后可以对该指令的各个参数进行修改，但指令名称是不能改变的。

限制条件

1．由于指令名称不允许修改，因此一些没有参数的元件，比如正缘触点、负缘触点、反转触点等，是不能被修改的。横线、竖线也是不能被修改的。

2．修改后的指令，同样必须能够正确通过指令校验，才能够被软件接受。

5.2.5 如何删除梯形图元件

竖线的删除

需要删除梯形图中的某条竖线时，可以用鼠标点选竖线所属单元格，然后点击工具栏上的竖线删除按钮（）或使用“ Ctrl + Shift + 下 ”组合键，可删除该竖线。

非竖线元件的删除

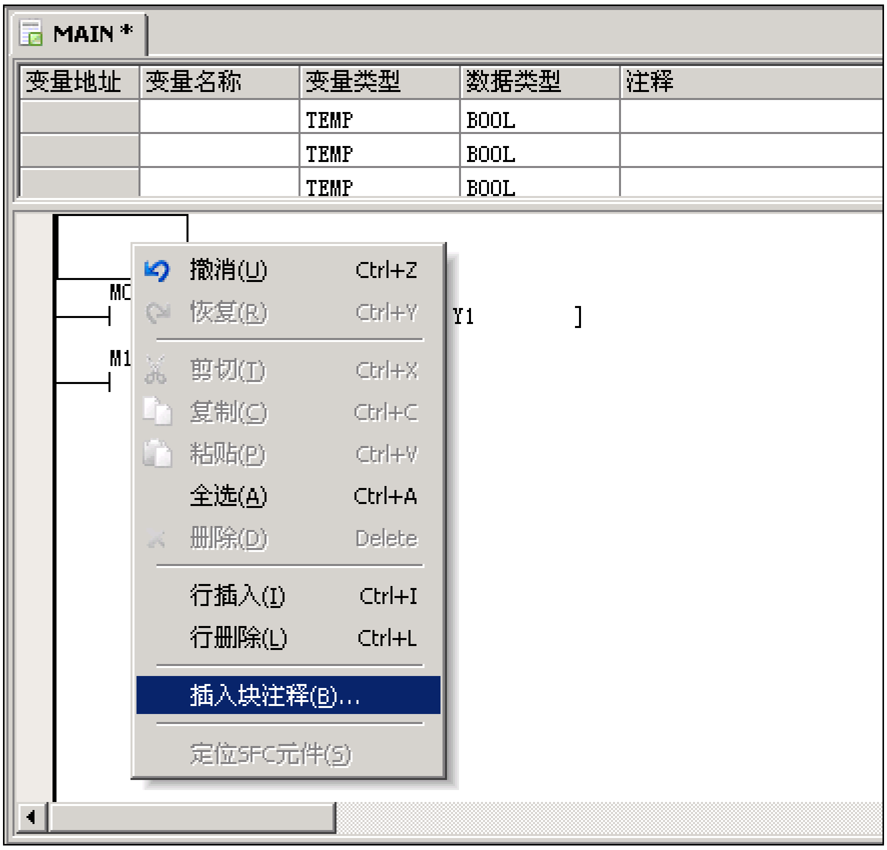
对于竖线之外的其他元件，删除时可以用鼠标点选元件所属单元格，然后点击工具栏上的删除按钮（），或者按[Delete]键，即可删除该元件。如果需要清除某一范围内的多个元件，可以通过拖动鼠标，选中要清除的区域，然后用与删除单个元件相同的操作，清除所选范围内的所有元件。请注意，当所选范围内包含竖线元件时，仅当此竖线所跨过的上下单元格都在选择范围内时，此竖线才会被删除。

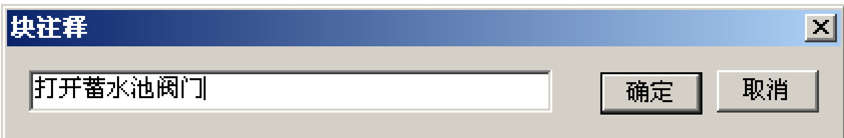
5.2.6 如何输入注释

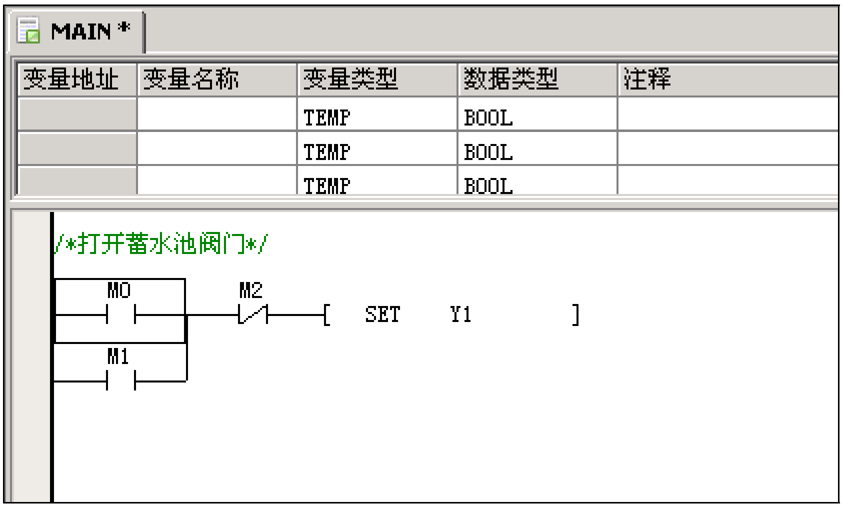
如何操作

为了使梯形图具有更好的可读性，可以在其中输入一些注释，这些注释称为块注释，每个块注释都会占用一整行的空间。

需要输入块注释时，首先选择一个空行，然后从右键菜单中选择[插入块注释]，在弹出的块注释对话框中输入注释文字并确认，软件会自动在所输入的文字两侧分别加上“ /* ”和“ */ ”，并以绿色显示。以上操作示意图如下：


MAIN *
变量地址
变量名称
变量类型
数据类型
注释
TEMP
BOOL
TEMP
BOOL
TEMP
BOOL
撤消(U) Ctrl+Z
恢复(R) Ctrl+Y
剪切(T) Ctrl+X
复制(C) Ctrl+C
粘贴(P) Ctrl+V
全选(A) Ctrl+A
删除(D) Delete
行插入(I) Ctrl+I
行删除(L) Ctrl+L
插入块注释(B)...
定位SFC元件(S)
M1
Y1]


块注释
打开蓄水池阀门
确定
取消

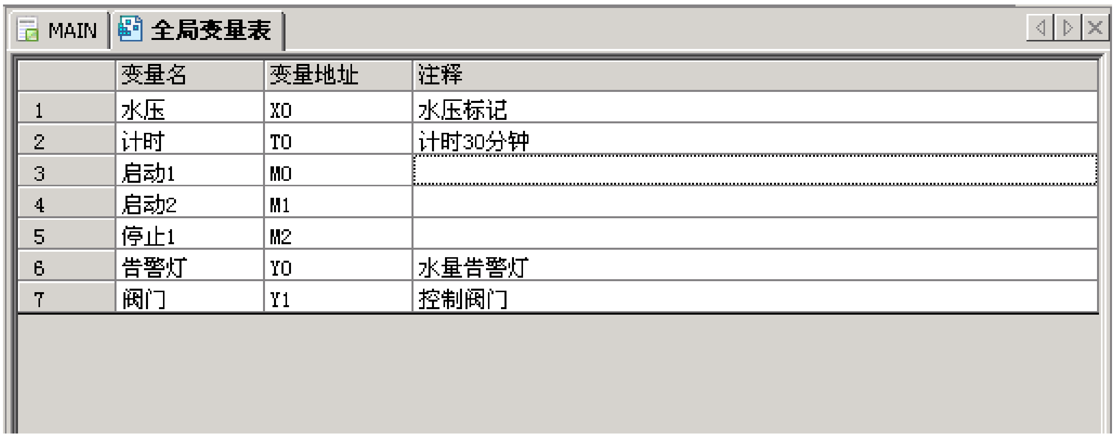

MAIN *
变量地址
变量名称
变量类型
数据类型
注释
TEMP
BOOL
TEMP
BOOL
TEMP
BOOL
/*打开蓄水池阀门*/
M0
M2
SET Y1]
M1

限制条件

由于块注释会占用整行空间，因此如果某行已经存在其他元件，则此行不能再输入块注释；同理，已经被块注释占用的行，也不能再输入其他任何元件。

5.2.7 如何在梯形图中使用已定义的变量

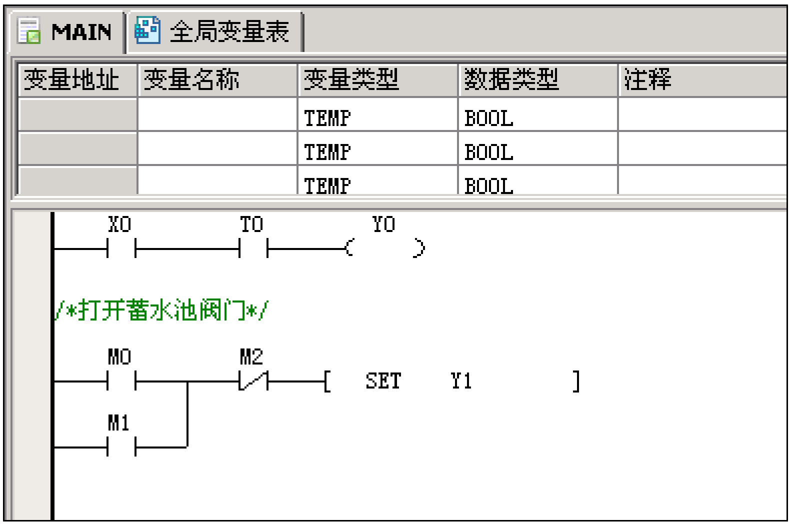
变量可以在全局变量表和局部变量表中定义（具体定义方法请参见相关部分说明），正确定义的变量可以在梯形图中使用。需要使用某个地址时，可以用代表此地址的变量名来代替，以增强程序的可读性。下图是全局变量表中定义的变量：

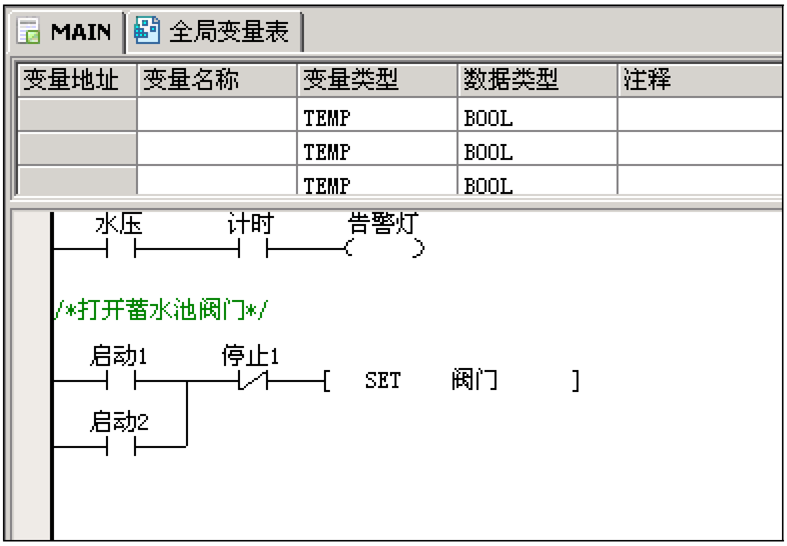


	变量名	变量地址	注释
1	水压	X0	水压标记
2	计时	T0	计时30分钟
3	启动1	M0	
4	启动2	M1	
5	停止1	M2	
6	告警灯	Y0	水量告警灯
7	阀门	Y1	控制阀门

符号编址

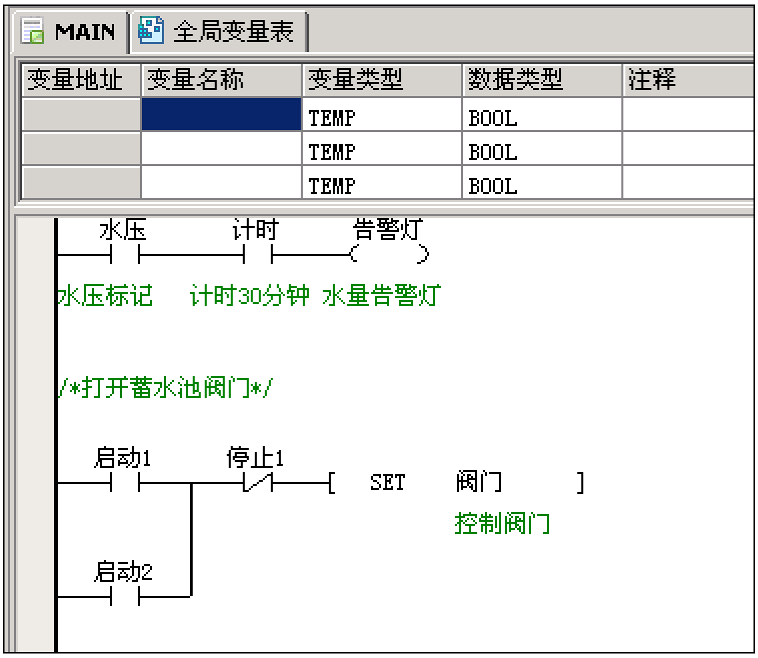
当使用了定义变量后，可以通过选择 [符号编址] 菜单在变量名和元件地址之间切换，下图分别展示了两种显示模式下的同一梯形图程序：





元件注释

可以通过选择 [元件注释] 菜单控制是否显示元件注释。以下是在显示元件注释时的梯形图程序：



5.2.8 如何优化梯形图程序的显示

为使梯形图程序更加整齐和美观，软件提供了梯形图程序“优化显示”功能，此功能对当前打开的梯形图程序进行整理，即将所有输入元件尽量向左对齐并将输出元件和应用指令对齐到某一指定列。默认情况下，该指定列就是当前窗口的最右端的一列，如果需要对齐到其他列，可以通过系统选项对话框指定，如下图：

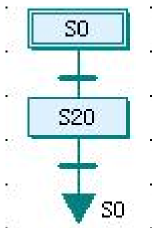
需要使用优化显示功能时，请打开某个梯形图程序，然后点击 [查看/优化显示] 菜单。

5.3 顺序功能图编辑器

5.3.1 预备知识

顺序功能图是根据机械设备的流程或者工序，将控制分成了多个步和步到步之间转换的一种语言。一个标准的顺序功能图由初始步、一般步、步间的转换条件、跳转和重置组成、每一步就是机械设备的一个处理工序，一个步中可以有内置梯形图，也就是这一步需要完成的处理工序。转换条件就是一个工序的完成和下一个工序的启动条件，它也需要内置梯形图来表示转换的条件。

示例：



如上面的顺序功能图所示，初始状态是 S0，完成 S0 工序中的内置梯形图后，如果满足转换条件，就停止执行 S0，开始执行 S20 工序，如果不满足转换条件，则继续执行 S0 中的内置梯形图；在最后完成 S20 并满足 S20 到跳转 S0 的转移条件后，停止 S20 工序的执行，重新启动初始步 S0 的执行，通过跳转，可以让整个顺序工序循环不停的执行。

5.3.2 顺序功能图元件说明

步进符

步进符分为初始步进符和普通步进符。它们利用内部软元件状态（S），在顺控程序上面进行工序步进控制。

1）初始步进符：

一个顺序功能图网络的执行必须由初始步进符开始。

初始步进符对应的图元符号是：S0*，可以单击按钮在选中的位置添加初始步进符元件。

初始步进符对应的 S 软元件地址范围是：S0～S19。

2）普通步进符：

普通步进符对应的图元符号是：S23*，可以单击按钮在选中的位置添加普通步进符元件。

普通步进符对应的 S 软元件地址范围是地址大于 19 的所有 S 软元件。

梯形图块

梯形图块对应的图元符号是：LAD1*，可以单击按钮在选中的位置添加梯形图块元件。

梯形图块包含一个内置梯形图，它可以用来定义在顺序功能图网络之外执行的代码。

转移符

转移符对应的图元符号是：，可以单击按钮在选中的位置添加转移符元件。

转移符中包含一个内置梯形图，可以在其中编写代码决定是否将与之连接的 S 软元件状态置位。

重置符

重置符对应的图元符号是：

，可以单击按钮在选中的位置添加重置符元件。

重置符的作用是将指定的 S 软元件复位。

跳转符

跳转符对应的图元符号是：

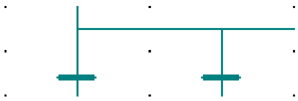
，可以单击按钮在选中的位置添加跳转符元件。

跳转符的作用是将指定的 S 软元件状态置位，使其包含的内置梯形图代码得以执行。

连接线

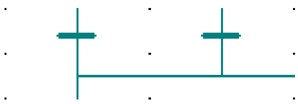
连接线将上面所述的顺序功能图元件按自上而下的顺序连接在一起。连接线除起连接作用之外，还包含选择分支、选择汇合、并行分支、并行汇合四种条件连接关系。

1）选择分支：

选择分支对应的图元符号是：，可以单击按钮在选中的位置添加选择分支连接线和两个转移符元件。

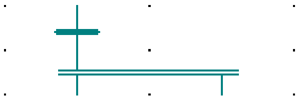
选择分支必须从一个顺序功能图步进符开始，到转移符结束。它代表的意义是：如果顺序功能图步进符执行成功，则根据转移符中规定的条件，选择一条支路继续向下执行。

2）选择汇合：

选择汇合对应的图元符号是：，可以单击按钮在选中的位置添加两个转移符和选择汇合连接线。

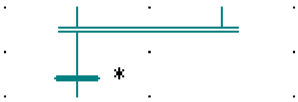
选择汇合必须从多个转移符开始，到一个普通步进符、跳转符、重置符结束。它代表的意义是：只要有一个转移符的条件满足，就运行下面连接的元件。

3）并行分支：

并行分支对应的图元符号是：，可以单击按钮在选中的位置添加一个转移符和并行分支连接线。

并行分支必须从一个转移符开始，到普通步进符、跳转符、重置符结束。并行分支代表的意义是：如果转移符中的条件被满足，则与并行分支连接的元件将被一同执行。

4）并行汇合：

并行汇合对应的图元符号是：，可以单击按钮在选中的位置添加并行汇合连接线和一个转移符。

并行汇合必须从普通步进符开始，到一个转移符结束。它代表的意义是：只要有一个普通步进符执行通过，就去执行转移符中的代码。

5.3.3 编辑顺序功能图程序

输入顺序功能图元件

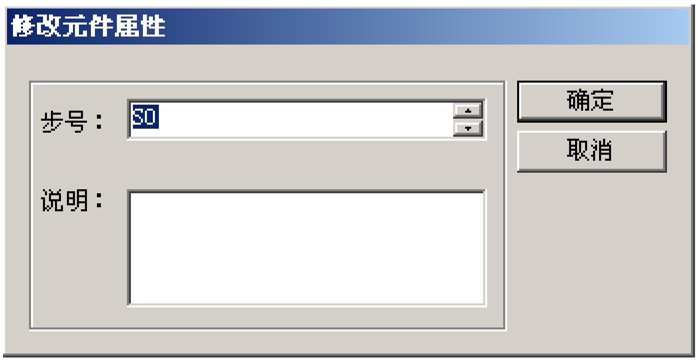
所有的顺序功能图元件都可以在顺序功能图工具栏和顺序功能图菜单中找到对应的图标或菜单项。可以通过单击特定的图标输入所需元件，并在顺序功能图元件输入对话框中输入指定顺序功能图元件的属性。同时，在顺序功能图工具栏和顺序功能图菜单中还提供了添加连接线的快捷方式，可以根据程序的需要在顺序功能图元件中间建立连接关系。

修改顺序功能图元件

针对不同的顺序功能图元件有不同的属性修改范围。初始步进符、不同步进符、跳转符、重置符元件，可以修改的属性包括步号、注释；梯形图块元件可以修改的属性有元件名；转移符元件可以修改的属性有元件注释。

具体操作：双击指定的元件，界面显示顺序功能图元件修改对话框，在顺序功能图元件修改对话框中修改选中的元件的属性。

顺序功能图元件属性修改对话框：



修改顺序功能图元件内置梯形图

在梯形图块、步进符、转移符这三种元件中可以添加自定义的内置梯形图，可以通过选中元件->点击鼠标右键->点击“ 内置梯形图 ”菜单的办法，打开内置梯形图编辑器。

使用复制/粘贴功能

选中某个单元格或者拖动鼠标选中某个区域后，使用 [编辑->复制] 菜单或者按 CTRL+C 组合键，可以将所选的元件拷贝到系统剪贴板，然后使用 [编辑->粘贴] 菜单或者按 CTRL+V 组合键可以将这些元件粘贴到其他位置或其他顺序功能图中。

添加/删除行

1．添加行：

行插入功能指在顺序功能图中插入一行空白行。

具体操作：先在要插入的行上选中任一单元格，然后从右键菜单中选择 [行插入]，软件会将当前行及下面所有行的元件向下移动一行，从而在当前行腾出一个空行出来。

2．删除行：

行删除功能指删除顺序功能图中的特定行。

具体操作：先在要删除的行上选中任一单元格，然后从右键菜单中选择 [行删除]，软件会将当前行清空，并将下面所有行的元件向上移动一行。

5.3.4 变量的使用

在顺序功能图程序中，只能使用全局变量表中定义的 S 软元件对应的变量。

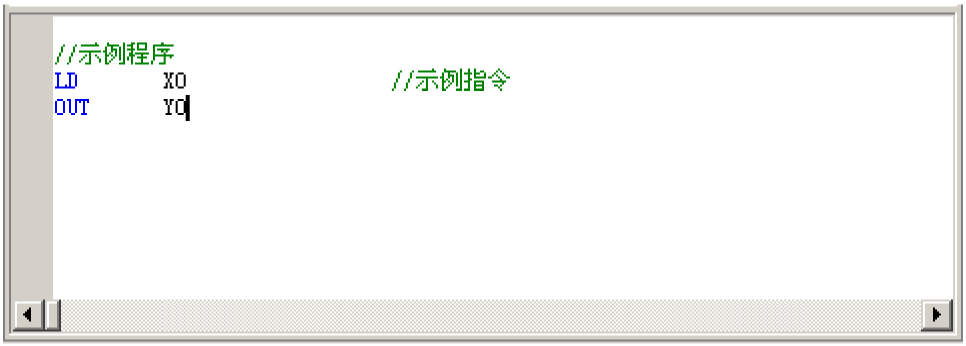
顺序功能图提供“符号”和“元件地址”两种编址方式，用鼠标点击“查看”菜单中的“符号编址”菜单项可以在两种编址方式之间切换，“符号编址”菜单项前有选中标记时，表示当前所有定义了变量的元件地址都会使用变量名来表示；“符号编址”菜单项前没有选中标记时，则无论是否定义了变量，都显示为元件地址，输入变量名也会自动转换为元件地址。

5.4 指令列表编辑器

5.4.1 指令编辑

指令输入

指令列表编辑器是一个文本编辑器，用于输入指令、操作数和注释。在编辑器中输入需要的指令后，软件会自动对齐指令、操作数和注释并将指令关键字用蓝色显示，操作数用黑色显示，注释用绿色显示，如下图所示：



在编辑器的任意位置输入双斜线“ // ”表示当前行后面的内容是注释。

拷贝/剪切/粘贴

使用鼠标在文本编辑器中拖动选择要拷贝的文字内容，当文字显示高亮状态时表示该部分内容已经被选中，使用快捷键 Ctrl+C 或者用鼠标点击“ 编辑 ”菜单中的“ 拷贝 ”菜单项，可以将选中的内容拷贝到 Windows 剪贴板中。被拷贝的内容可以在任何的标准文本编辑器中粘贴，包括 Windows 写字板、记事本或者 Word 等。

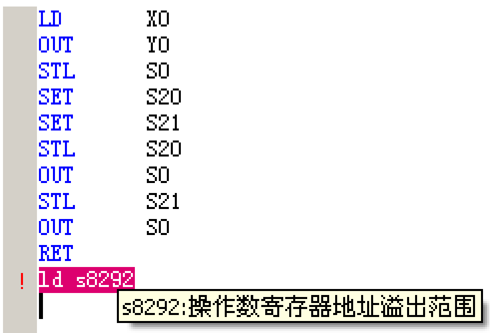
剪切的操作和拷贝非常类似，不同的是被剪切的内容会从编辑器中被删除，使用快捷键 Ctrl+X 或者用鼠标点击“ 编辑 ”菜单中的“ 剪切 ”菜单项，可以将选中的内容剪切到 Windows 剪贴板中。

拷贝或者箭切完毕后，用鼠标或者键盘将当前光标定位到想要粘贴的位置，使用快捷键 Ctrl+V 或者用鼠标点击“ 编辑 ”菜单中的“ 粘贴 ”菜单项，即可将被拷贝或者剪切的内容粘贴到光标所在的位置。

正确性校验

输入每一行指令后，如果光标切换到其他行，则会对离开的行进行正确性校验，如果有语法错误，软件会将有错误的文本行显示为红色，当鼠标悬浮在有错误的文本行时，软件会用工具提示的方式提示错误信息。只有校验正确的语句才会格式化显示，否则会在行头显示一个“ ! ”并且用红色突出有语法错误的行。在一次粘贴多行文本内容到编辑器中时，编辑器会对每一行作语法检查。

有语法错误时，请将鼠标放在有错误的位置，软件显示如下图所示：



在图中，LD 指令的操作数 S8292 超过了 S 元件的最大范围，因此认为是非法的。

5.4.2 变量的使用

在软件中，可以定义全局和局部变量，具体定义方法请参阅相关章节，定义好的变量可以在指令列表中使用，在输入元件地址时，如果该元件地址定义了变量，则可以用变量名代替元件地址作为操作数输入，如果使用的是局部变量，软件自动在变量名前加上“#”号和全局变量区别。

指令列表提供“符号”和“元件地址”两种编址方式，用鼠标点击“查看”菜单中的“符号编址”菜单项可以在两种编址方式之间切换，“符号编址”菜单项前有选中标记时，表示当前所有定义了变量的元件地址都会使用变量名来表示；“符号编址”菜单项前没有选中标记时，则无论是否定义了变量，都显示为元件地址，输入变量名也会自动转换为元件地址。

5.4.3 标签的使用

标签是类似书签功能的标记，它可以在指令列表程序中的任意行加入，使用工具栏上的按钮，或者点击“编辑”菜单中的“添加/删除标签”可以在光标所在行添加标签或者将标签删除；点击或者按钮，可以迅速定位到光标位置的下一个书签或者上一个书签所在的行；点击按钮将清除所有已经存在的标签。

5.5 全局变量表

5.5.1 全局变量的定义

全局变量是为 PLC 的某个地址定义的有意义的符号名称，该符号名称在整个工程范围内都可被访问到，相当于使用该变量对应元件，全局变量在全局变量表中定义。

全局变量声明包含变量名称、变量地址、变量注释三个属性。

全局变量的定义规则为：由 A～Z、a～z、0～9、下划线、汉字混合组合而成，变量名称不能以数字开始，也不能是单独的数字。名称不区分大小写，长度不能超过 8 个字节，不能使用元件类型字母+数字作为程序和变量名称。名称中不能包含空格，不能使用和关键字相同的名称，保留的关键字包括：基本数据类型名称、指令名称、指令表语言中的运算符。

全局变量个数不能超过 500 个。

5.5.2 编辑全局变量表

双击工程管理窗口的全局变量表可以打开全局变量表：

可以在全局变量表中查看和编辑变量名、元件地址和元件注释。在编辑过程中可以对其进行全局变量表的合法性检查：

- 如果元件地址错误，软件使用不同颜色显示该元件地址；
- 符号表中输入的变量名称必须唯一，对于重复的变量名，软件用不同的颜色显示变量名。
- 元件注释最大长度为 20 个字节。

使用鼠标单击符号表表格的列头，可以对变量名、元件地址和元件注释中的任何一列升序或者降序排列。

5.5.3 全局变量的使用

定义的全局变量可以在各种程序中使用，如果定义了相同名称的局部变量或者全局变量，变量名前加上“#”表示该变量是使用一个局部变量，否则是使用全局变量。

5.6 局部变量表

5.6.1 局部变量的定义

对于每个程序，根据类别不同可以定义该子程序使用的输入参数，输出参数和局部变量，对参数和变量的说明在程序的局部变量表中定义。

局部变量是为 PLC 的某个地址定义的某个有意义的符号名称，该变量具有局部范围，只在某个程序单元内有效，局部变量在局部变量表中定义。

局部变量声明包含变量地址、变量名称、变量类型、数据类型、变量注释五个属性，变量类型有三种类型（IN、IN_OUT、OUT、TEMP）其值是不可以修改的。

局部变量的命名规则与全局变量完全相同。

对于主程序、中断子程序，可以在每个程序的局部变量表中定义局部变量。局部变量只在定义它的程序中有效，程序运行完毕后从内存中释放。局部变量在局部变量表中的类型关键字为“ TEMP ”，一个程序中定义的局部变量个数不能超过 64 个。

对于子程序，可以在局部变量表中除了可以定义局部变量外，还可以定义该子程序的调用参数和返回参数。

输入参数在局部变量表中的类型关键字为“ IN ”。

输出参数在局部变量表中的类型关键字为“ OUT ”。

输入输出参数在局部变量表中的类型关键字为“ IN_OUT ”。

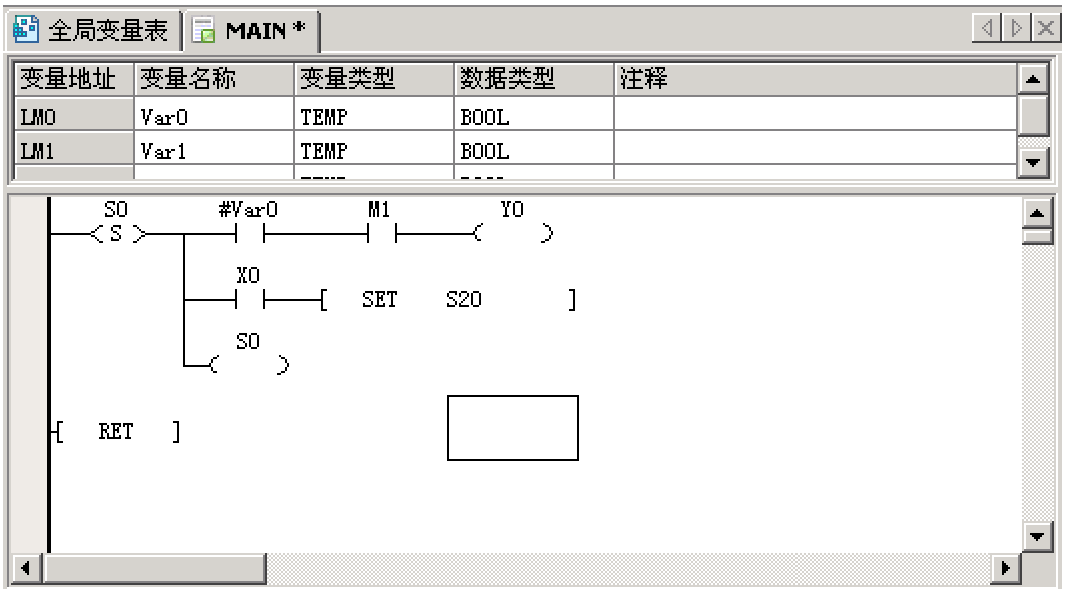
三种参数定义同时不能超过 16 个。

可以在局部变量表中查看和修改变量名称、数据类型和变量注释（注：变量类型不可修改），数据类型支持 BOOL、WORD、DWORD、INT、DINT 和 REAL。

变量注释最大长度为 20 个字节。

5.6.2 编辑局部变量表

每个程序都有自己的局部变量表，局部变量表是程序的一个组成部分，创建了程序（主程序、子程序、中断子程序）后也就创建了相应的局部变量表。编辑窗口的上半部分显示局部变量表，下半部分显示程序。



可以在局部变量表中查看和编辑变量名称、变量类型、数据类型和注释。变量地址是自动分配的，只能查看不可编辑；

在编辑局部变量表的过程中可以对其进行合法性检查：

- 对输入不合法的变量名，软件用不同的颜色给出提示。
- 局部变量表中所有输入的变量名称和参数名必须唯一，输入确认后，对于重复的变量名和参数名，软件给出提示并用不同的颜色显示变量名。

变量地址的自动分配：在定义输入/输出参数和局部变量时，一旦确定变量的存储类型，立刻为该变量或者参数在 LM 或者 V 寄存器中分配具体的地址。

对位变量的分配使用 LM 寄存器，对字和双字变量的分配使用 V 寄存器，其中双字变量的寄存器地址显示为第一个字的地址。当修改某个变量的数据类型或者删除某个变量导致占用的字节数改变时，该变量的后续变量地址也会重新分配。

5.6.3 局部变量的使用

定义的局部变量和参数可以在程序中使用，在变量名前加上“#”表示该变量是使用一个局部变量，否则是在使用全局变量，对于一个局部变量且没有相同的全局变量，对指令格式化后应自动在变量名前加上#。

📖 注意

如果当前处于元件地址编程模式，不显示变量名，则地址重新分配时不调整代码中的地址。

5.7 交叉引用表

5.7.1 交叉引用表的定义和使用

交叉引用表用于记录元件在各个程序中的使用情况，可以通过双击交叉引用表中的记录定位该交叉引用信息所在的程序和行列。

交叉引用表不可以编辑，只可以进行查找和定位，要使用查找和定位功能，可以在编辑菜单下选择的查找和转至子菜单，也可以点击按钮。

程序在编译或者转换完成后产生交叉引用表，在下次编译成功前保持不变，退出工程后交叉引用表内容保存到文件中。交叉引用表包含的信息如下：

元素：如果元件在全局变量表中定义为变量，且当前显示模式设置为显示变量模式，则显示元件变量名，否则显示元件地址；

块：元件所在的程序；

位置：元件所在程序的行号和列号；

上下文：对元件的使用方式。

5.7.2 切换编址方式

如果在交叉引用表中存在一个或者多个全局变量表或局部变量表中定义的变量，则既可以将该行数据显示为符号也可以显示为地址，你可以通过选择[符号编址]菜单在变量名和元件地址之间切换，地址编址模式下所有元件显示为元件地址，符号编址模式下在全局变量表或者局部变量表中定义的地址显示为符号名称，没有定义的显示为元件地址。

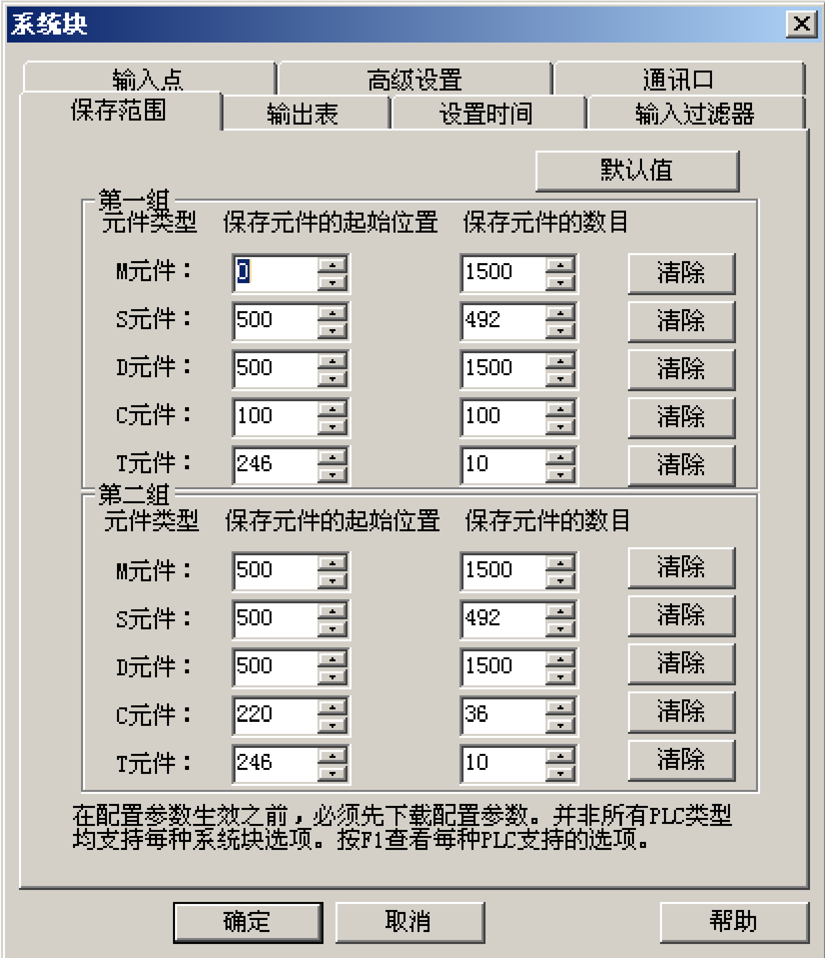
第六章 PLC 设置

6.1 系统块

通过系统块可以对掉电保存的元件、时间、通讯、输入点等进行设置。设置完以后就可以将系统块下载至 PLC 中，这些设置将会生效。

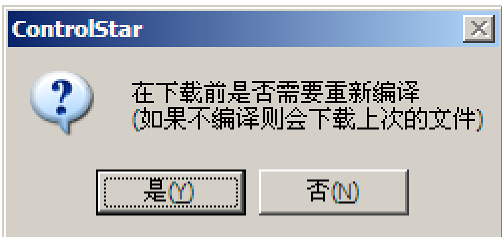
从的工程管理器窗口中双击选择系统块，将会弹出系统块对话框，如下图所示：

系统块设置的对话框如下图所示：



配置好系统块以后，选择 [PLC/全部编译] 菜单，项目的系统块文件被编译。

配置好系统块以后，选择 [PLC/下载] 菜单，会弹出消息对话框提示是否需要重新编译，选择“是”，则系统块信息会被编译并下载。



6.1.1 系统块保存范围

双击打开对话框以后，在对话框的第一页可以看到“保存范围”，配置保存元件地址范围。请注意：可保存的元件地址范围会因 PLC 型号的不同而异。

在默认情况下，D、M、S、T 和 C 元件均被设为保存一定范围，可以在此修改需要保存的元件地址范围。

PLC 电源中断后的数据保存

1．功能说明：配置元件保存地址范围。

2．元件类型和范围

元件类型	保存范围	默认设置
M	M000～M1999	M500～M1999
S	S000～S991	S500～S991
C（16 位）	C0～C199	C100～C199
C（32 位）	C200～C255	C220～C255
T	T246-255	T246-255
D	D0～D7999	D500～D1999

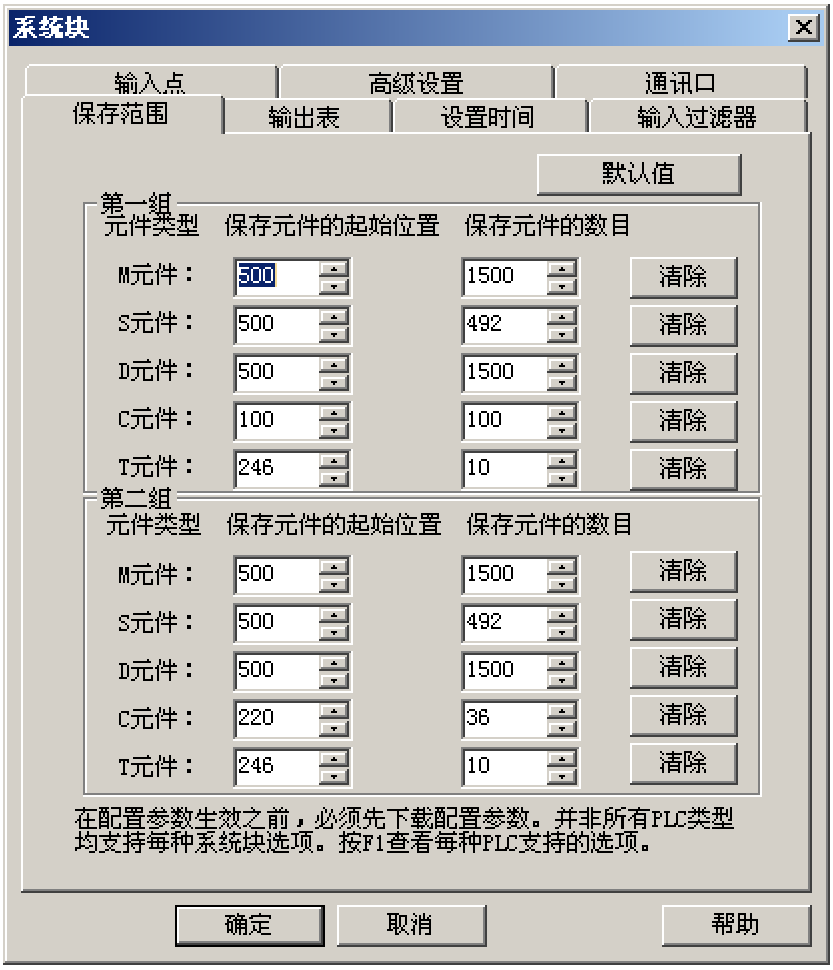
最多可以定义两组保留范围，最终的保存范围是定义的两组的保存范围的并集。

掉电时

PLC 将按照上图中定义的范围，将元件保存到 SRAM 中，由电池保存。

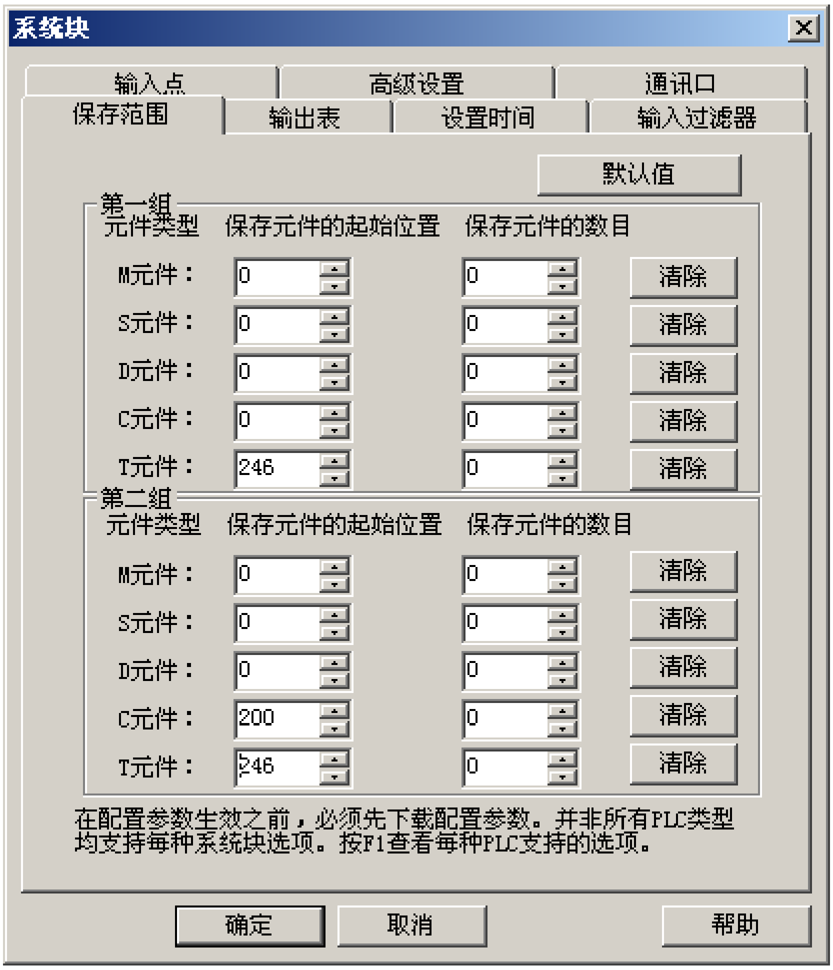
上电时

PLC 检查 SRAM 中保存的数据是否正确，如果 SRAM 数据被成功保存，SRAM 内存的保留区不变。如果保存 SRAM 的内容错误， 则 PLC 将把 SRAM（包括保留和非保留范围）中的元件清零。



3．清除

当点击清除后，相应元件的保存个数被设为零。如下图所示：



6.1.2 系统块输出表

单击 " 输出表 " 标签，可以设置 PLC 停机时的 I/O 点输出状态。

1．功能说明

设置停止状态时的输出点组态，当 CPU 处于停止状态时，输出点有三种选择：

1）保持停止前的输出点状态

2）禁止所有的输出点

3）按输出点组态表（Y00～Y177 可组态）设置输出状态

2．输出点组态表

	0	1	2	3	4	5	6	7
Y000	√							
Y010								
Y020								
Y030								
Y040			√					
Y050								
Y060								
Y070								
Y100					√			
Y110								
Y120								
Y130								
Y140								
Y150			√					
Y160								
Y170								

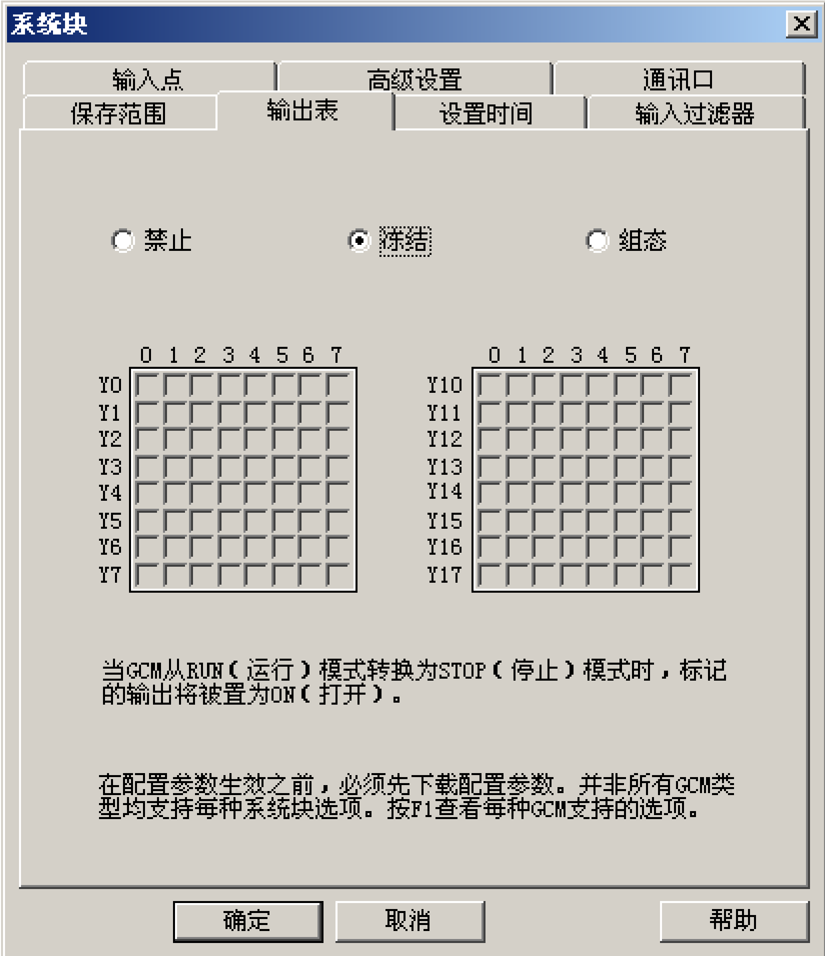
注：√表示输出 ON

3．禁止

选择“禁止”选项，PLC 将在停机时禁止所有的输出点，在运行到停止的时候生效。如下图所示：

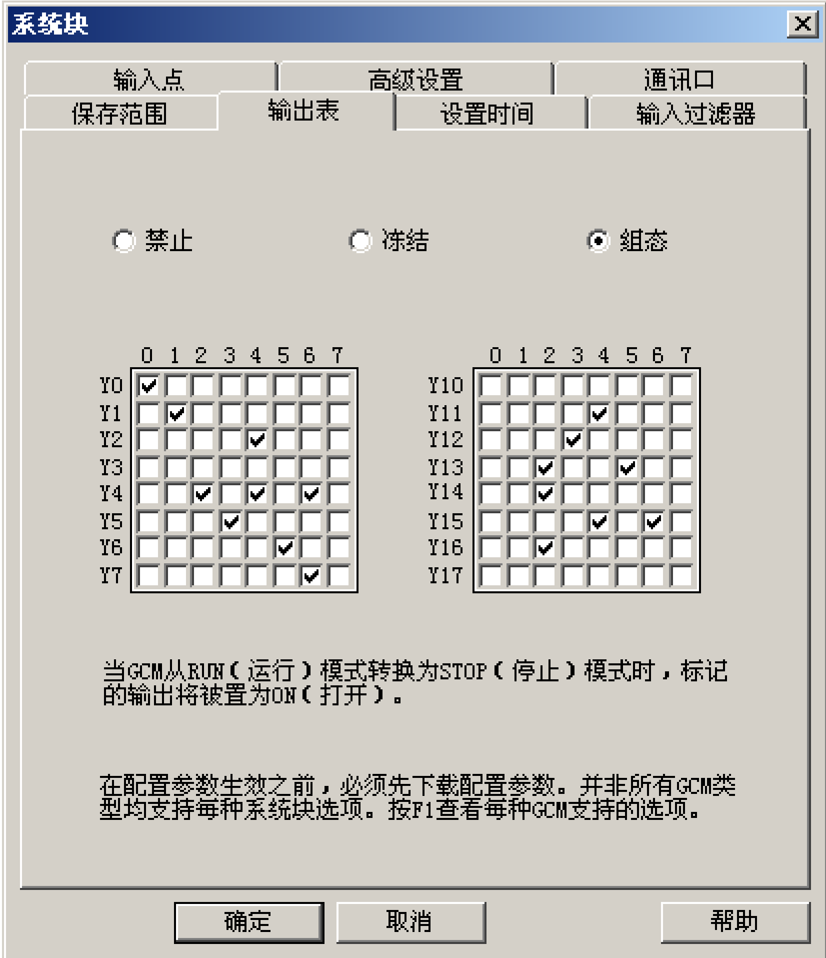
4．冻结（保持原状态输出）

选择“冻结”选项，PLC 将在停机时将所有数字输出冻结在最后的状态中。如下图所示：



5．组态（按输出组态输出）

选择“组态”选项，PLC 将在停机时设置输出点为已知状态。所有输出点默认状态为“关闭”（0）的状态。如下图所示：



6.1.3 系统块设置时间

看门狗定时器

功能：设置用户程序运行看门狗时间。看门狗时间是指允许用户程序运行的最大时间，当用户程序的实际执行时间超过看门狗时间时，PLC 会停止用户程序，点亮程序告警灯（红色），输出按系统配置进行输出。看门狗定时器时间可设定范围为：0ms～1000ms，默认值为 200ms。

恒定扫描时间

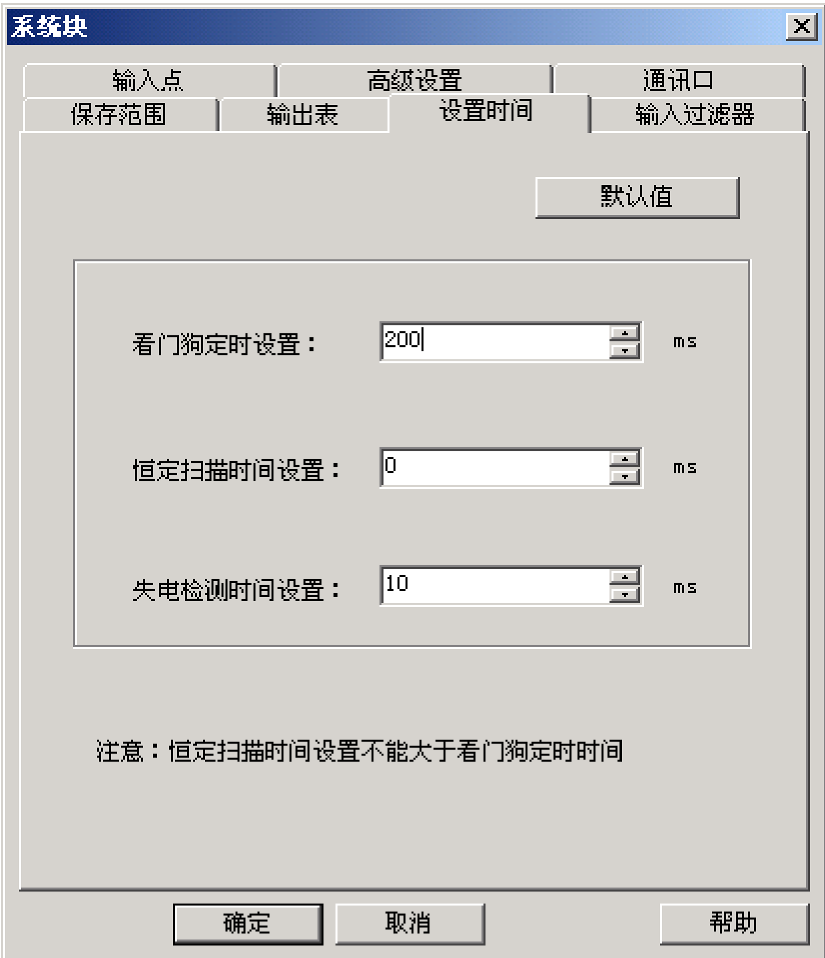
功能：恒定扫描时间是指系统在恒定时间扫描寄存器的时间。读取系统的恒定时间扫描设定寄存器，在恒定时间内对用户程序只作一次扫描。恒定时间可设定范围为：0ms～1000ms。默认值为 0ms。

失电检测时间

功能：当瞬时失电时间大于设定的失电检测时间时，PLC 会进入停止状态。系统按照配置的停电保持元件范围进行元件值的保存。失电检测时间可设定范围为：0ms～100ms。默认值为 0ms。

📖 注意

恒定扫描时间设置不能大于看门狗定时时间

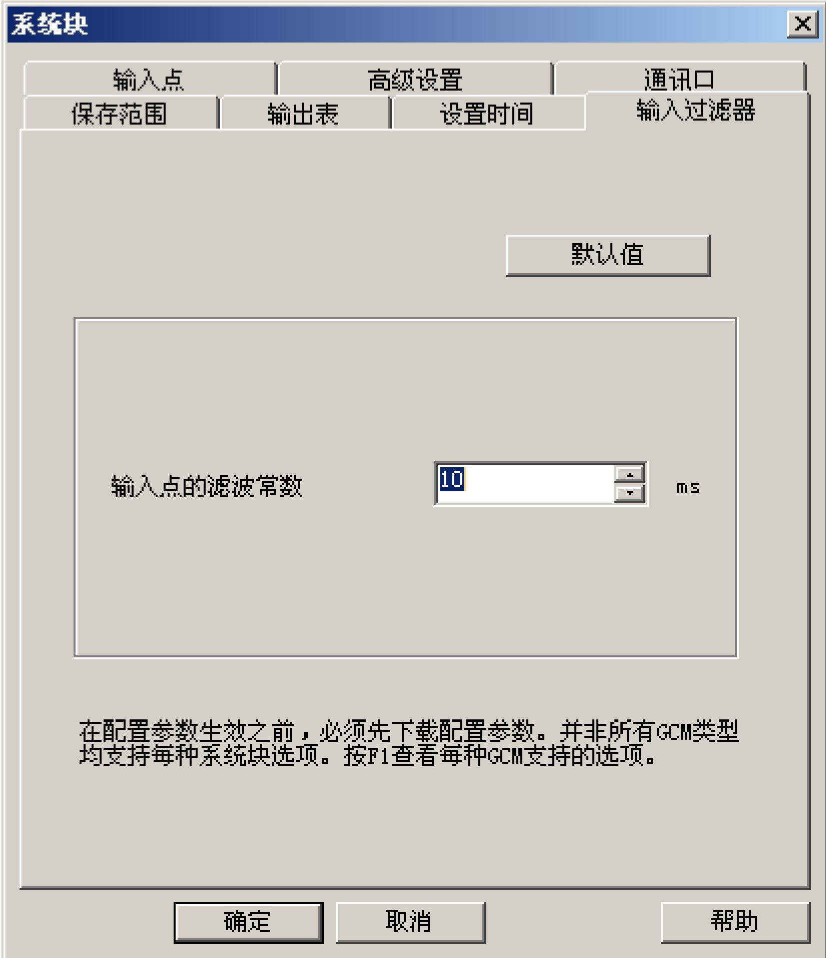


6.1.4 系统块输入过滤器

单击“输入过滤器”标签，可以为 PLC 输入点设置一个延迟时间，过滤掉外部的干扰。通过设置输入延迟时间，可以过滤数字输入信号。

功能

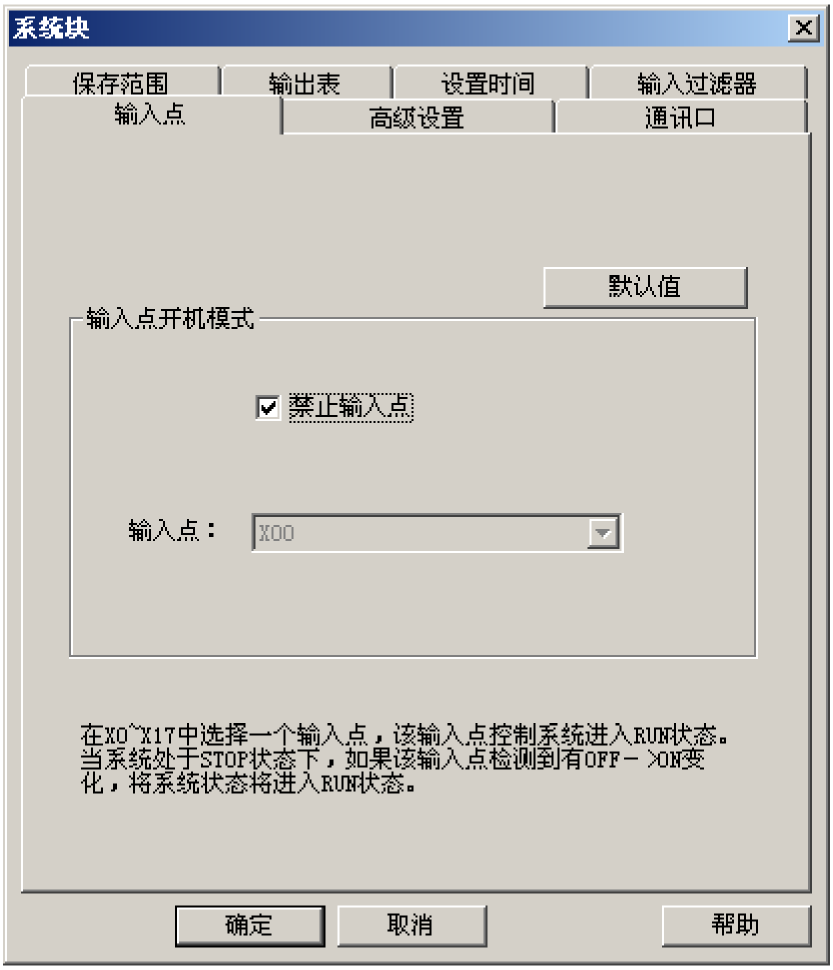
可以利用编程软件设置前十六个(X0～X17)输入点数字滤波常数。数字滤波常数的范围是 0-60ms，默认值为 10ms。



6.1.5 系统块输入点

输入点启动

指定在 X0～X17 中某个输入点为强制 RUN 输入点，当系统处于 STOP 状态下，检测该点为 ON 时将系统状态由停止切换到运行状态。



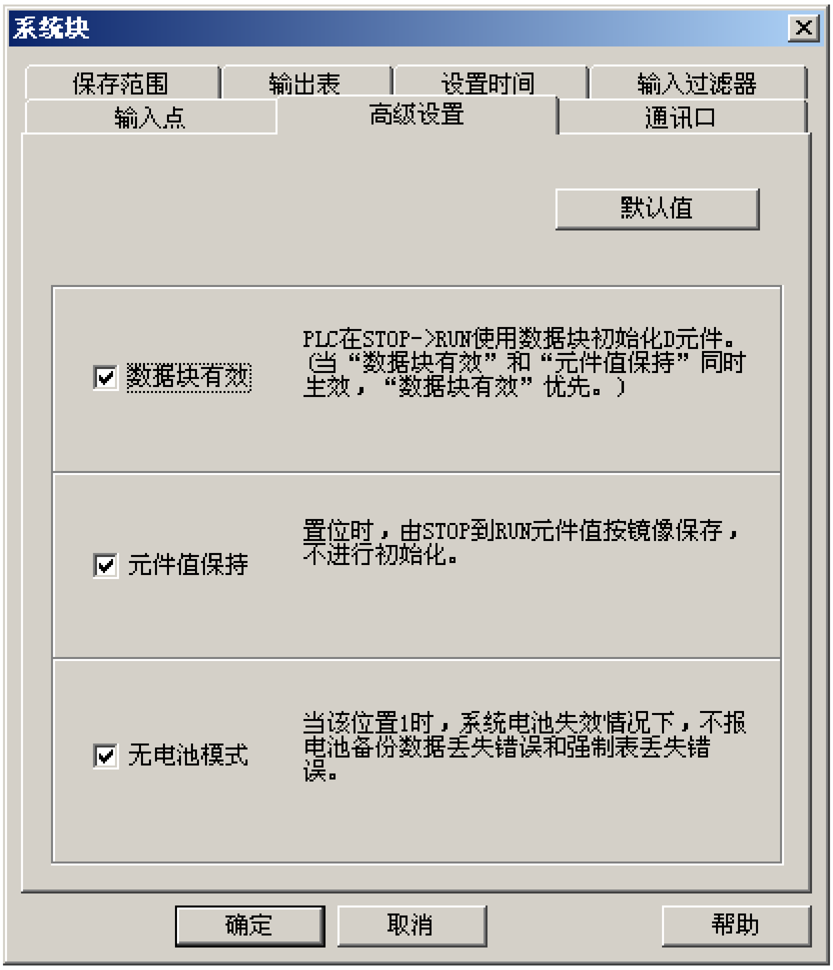
禁止输入点

选择单选框“禁止输入点”，禁止输入点启动。

6.1.6 系统块高级设置

功能

配置一些诸如数据块有效、元件值保持、无电池模式等高级设置。



数据块有效：PLC 在 STOP->RUN 使用数据块初始化 D 元件。

元件值保持：置位时，由 STOP 到 RUN 元件值按镜像保存，不进行初始化。

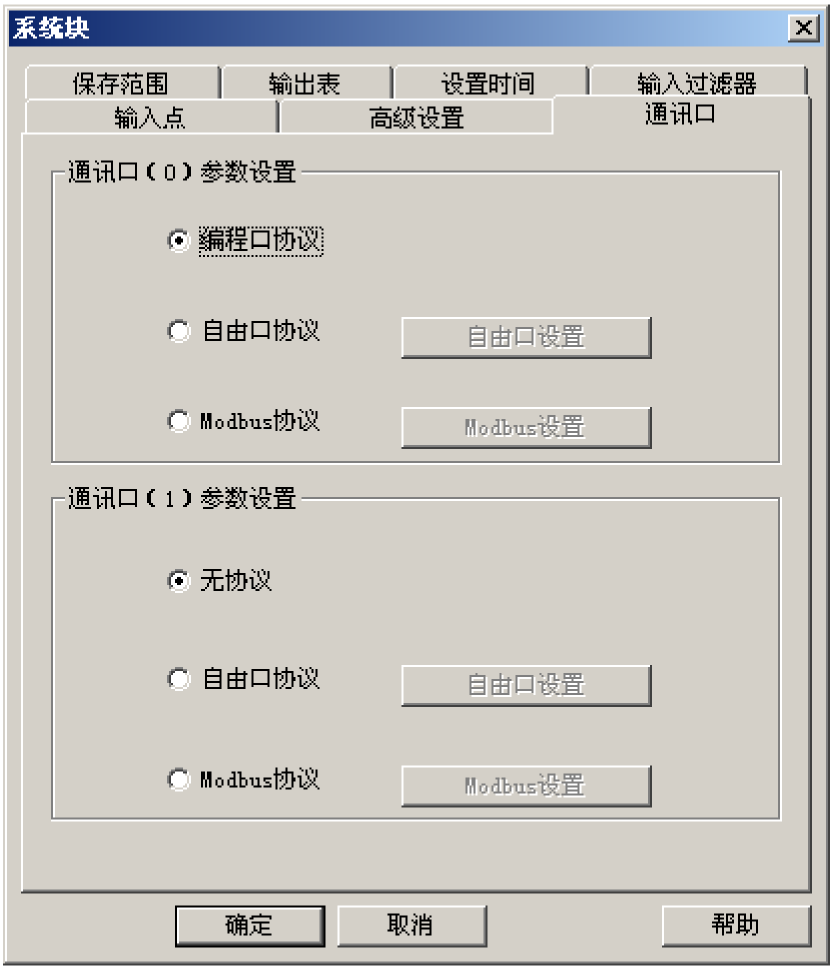
无电池模式：置位时，系统电池失效情况下，不报电池备份数据丢失错误和强制表丢失错误。

注意

当“数据块有效”和“元件值保持”同时生效，“数据块有效”优先。

6.1.7 通讯口设置

通过此页面可以对软件的通讯口进行设置，包括使用什么样的通讯协议、如何支持 Modbus 协议、自由口协议、编程口协议等。

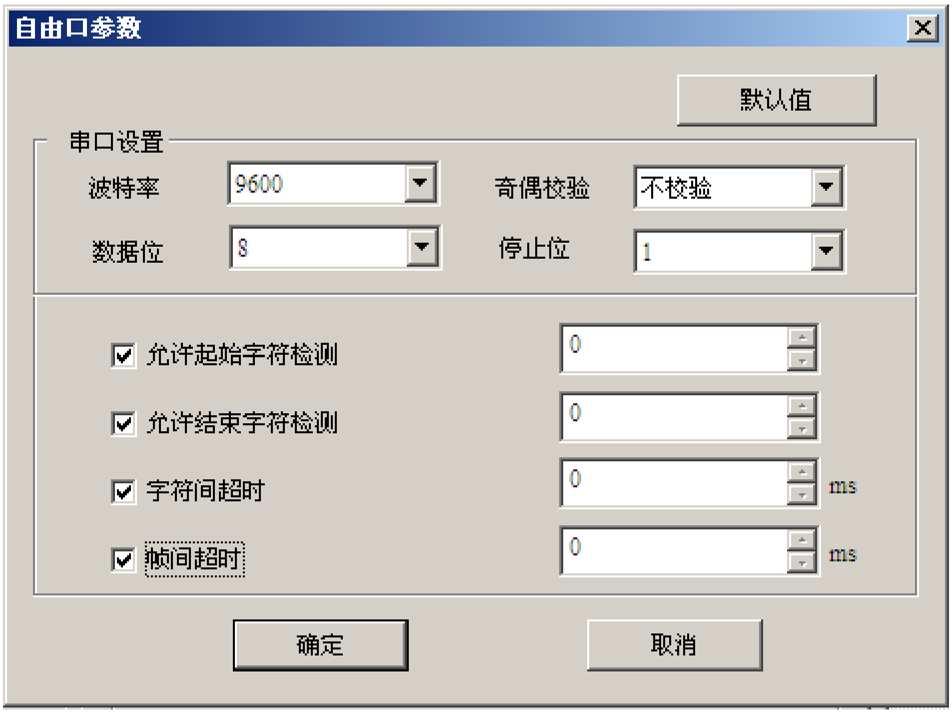


系统对话框中的通讯口 0 默认是编程口协议。通讯口 1 的默认是无协议的。可以分别对通讯口 0 和通讯口 1 进行设置。

自由口协议

自由口协议是自定义数据文件格式的通讯方式。自由端口通讯模式支持 ASCII 和二进制两种数据格式。只有在 PLC 位于 RUN（运行）模式时才能使用自由端口通讯，在自由端口模式下，无法与编程设备通讯；STOP 状态下，串口 0 只能用于编程口通讯。

可以设定传输的信息中信息字符的起始位，字符间超时时间、接收字符的最大值。接收指令允许接收一个或多个字符的缓冲器，最多可达 255 个字符。串口按照接收开始条件，开始自由口接收。



Modbus 协议

通过 MODBUS 协议 MODBUS 的主站可以与从站（包括变频器）进行通讯，通过 MODBUS 协议提供的 MODBUS 功能码，发送对从站的控制帧。

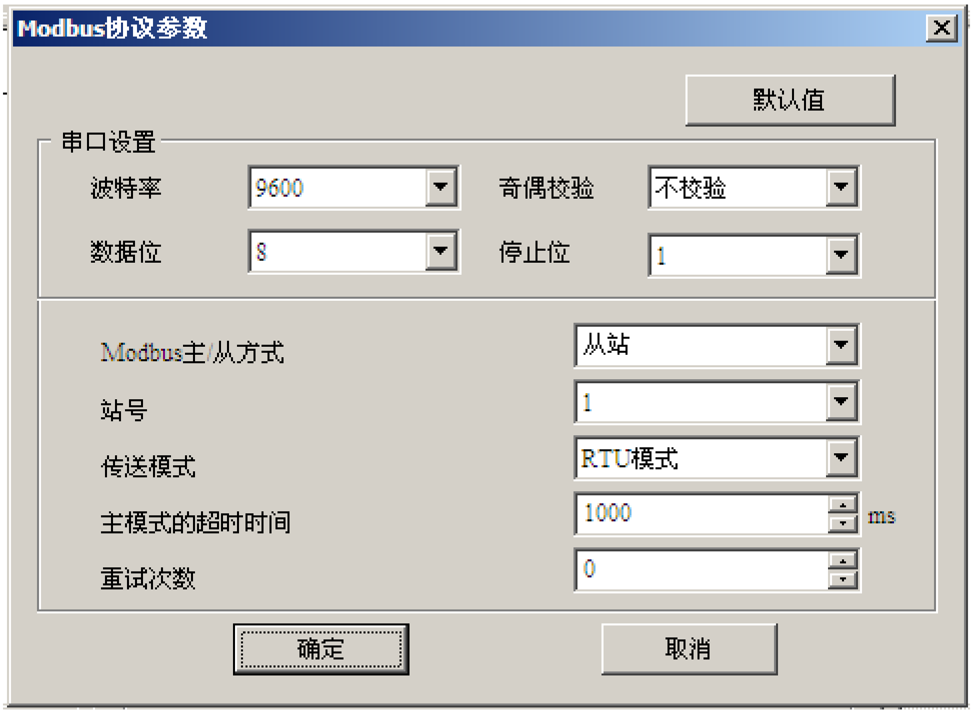
（1）通讯口【0】：

可以设置为 MODBUS 从站。通过 MODBUS 协议提供的功能码进行通讯，对主站的请求进行回应，判断是本地的报文，进行处理，同时统计接收的报文信息，供 MODBUS 主站查询。

可设置内容包括：波特率、数据位、奇偶校验、停止位、起始字符检测、结束字符检测、字符间超时、帧间超时。

（2）通讯口【1】：

串口 1 可以设为从站和主站，最多支持 31 个从站，1～31 可设，主站与从站站号不能相同。等待接收，在超时时间（可以通过系统块和 SD 设置）内接收到数据，进行地址、CRC（RTU 模式）或 LRC（ASCII 模式）校验、字符间超时（RTU 模式）的判断，从而判断通讯的正确与否。



可设置内容包括：波特率、数据位、奇偶校验、停止位、主/从方式、站号、传送模式、站号、最大超时时间，重试次数。

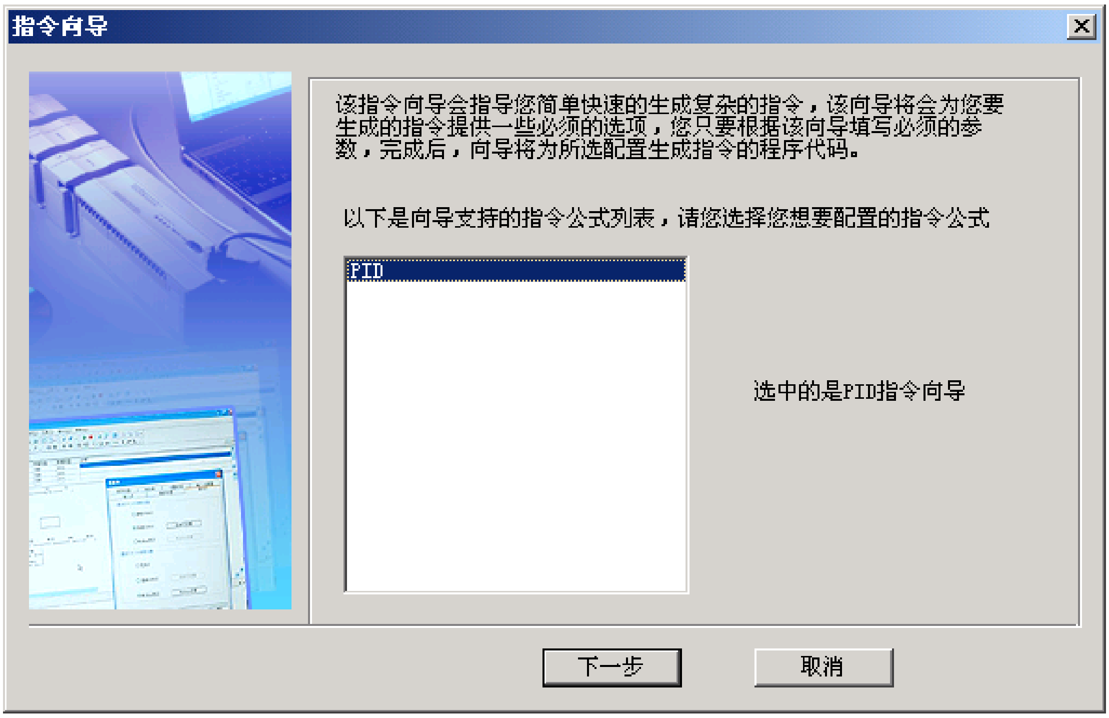
6.2 PID 指令向导

6.2.1 PID 指令向导的功能

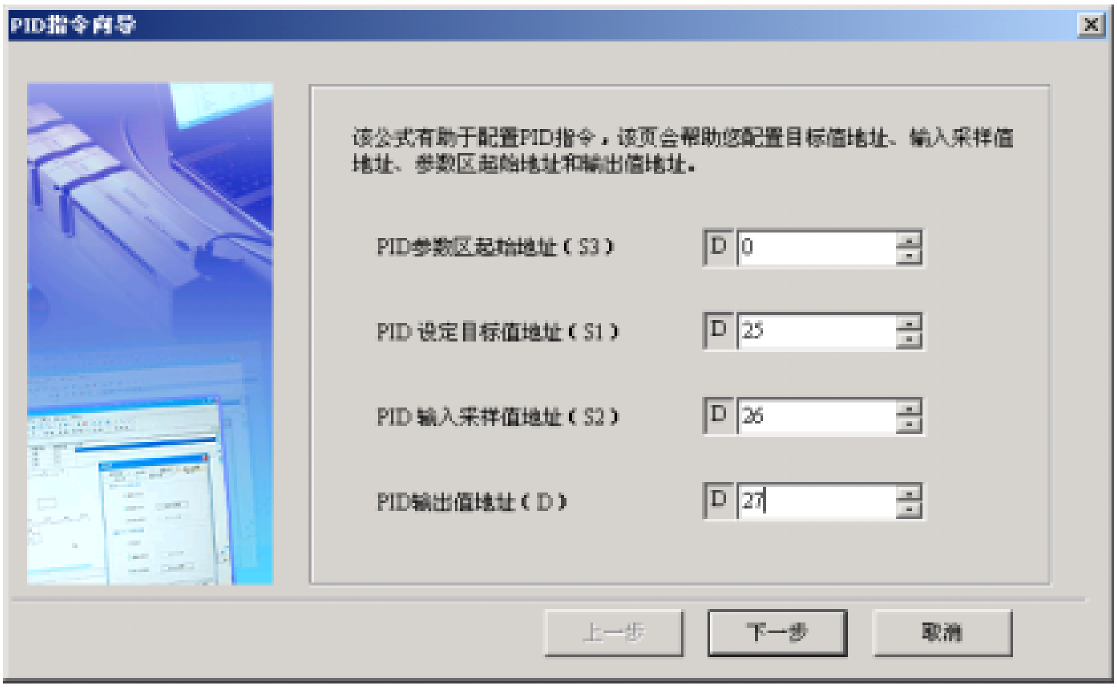
对于许多参数复杂的指令，生成这些指令不是很方便，所以软件提供指令向导功能以方便的生成这些指令。

PID 指令向导通过对话框列出 PID 计算的各项参数的含义和输入框，输入完成后，PID 指令向导自动生成 PID 参数设置子程序和 PID 执行两个子程序。PID 参数设置子程序是一次设置 PID 指令的参数，当能流有效时，每个运行周期每次执行 PID 指令的子程序.

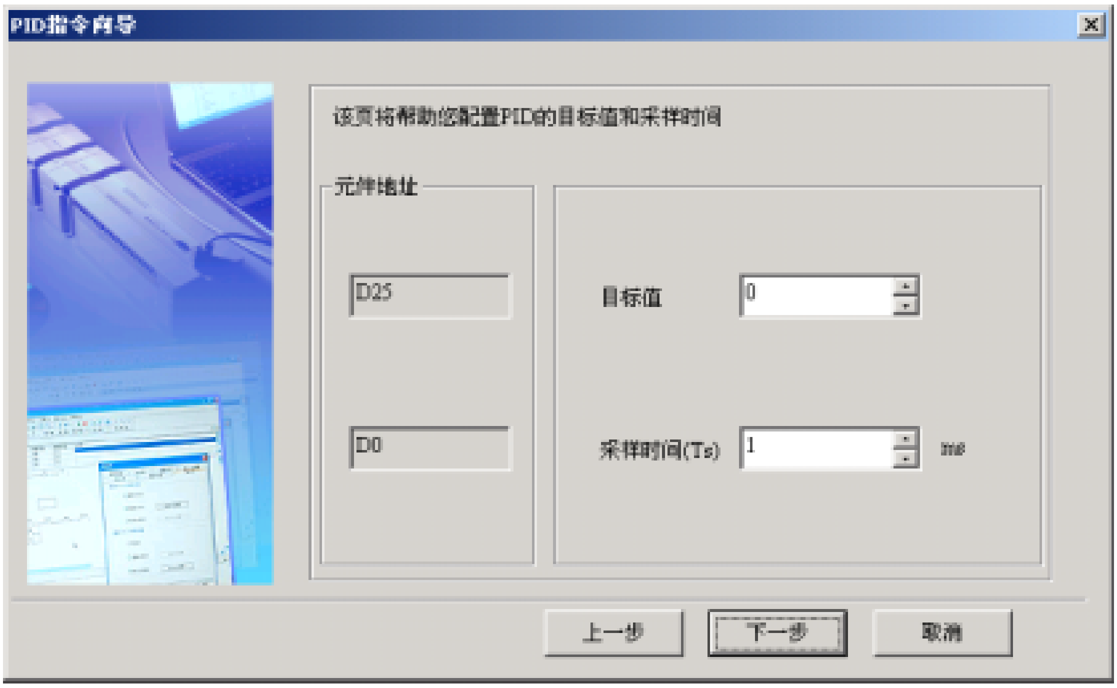
1、选择指令向导的类型



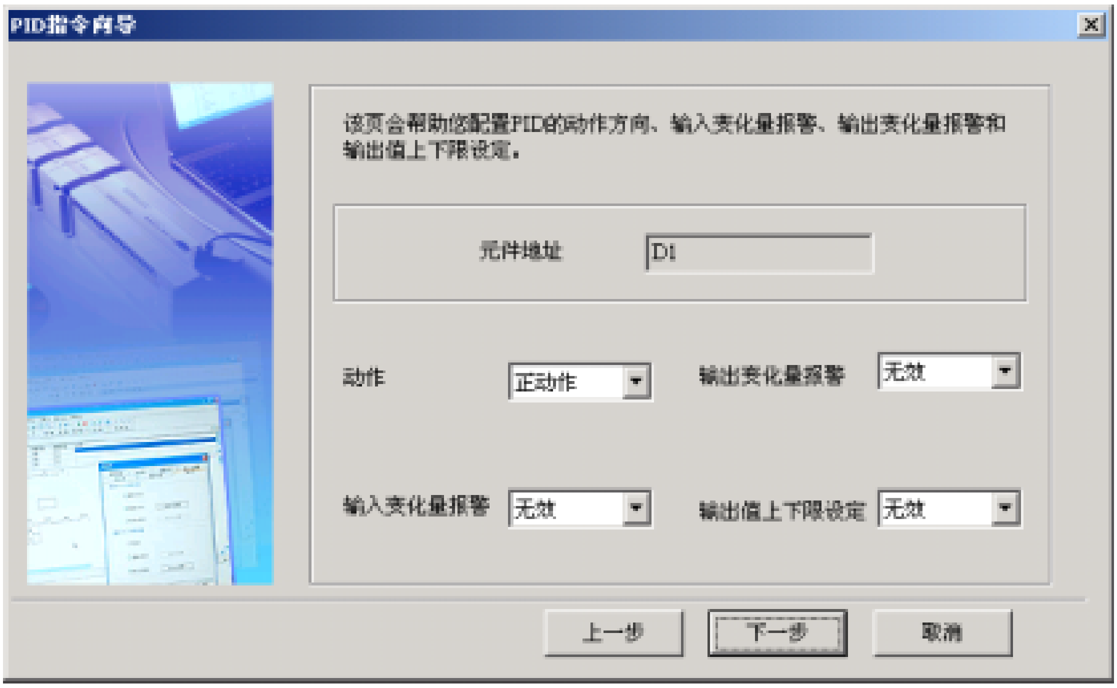
2. 输入 PID 的参数区的起始地址、目标地址、输入采样地址、输出地址。



3. 输入目标值和采样时间

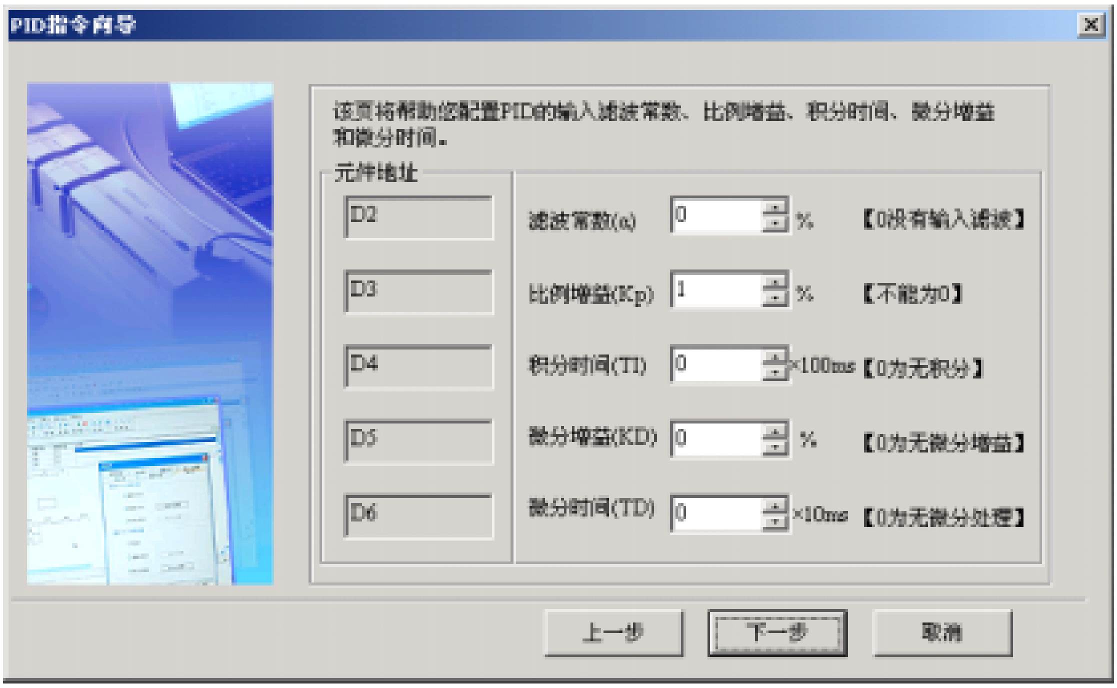


4．输入 S3+1 单元——动作方向和报警是否有效

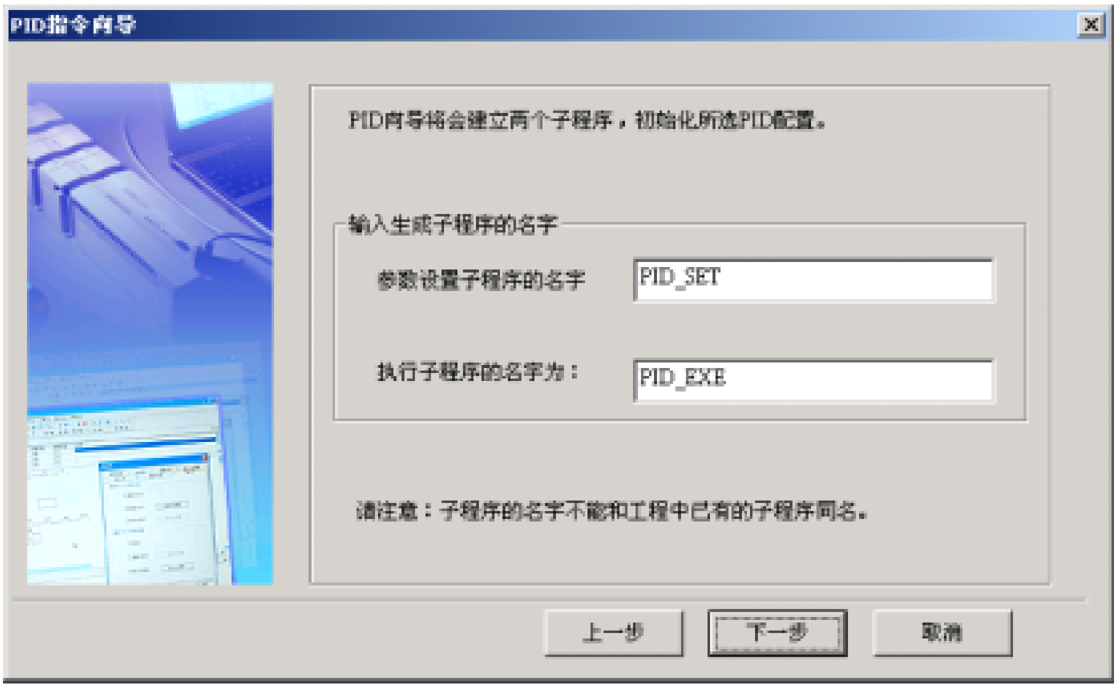


5．输入变化量报警值和输出变化量报警值

6．输出输入滤波常数、比例增益、积分时间、微分增益、微分时间。



7．输入初始化子程序和执行子程序的名字



6.2.2 PID 参数设置子程序

需要确定的参数一共是 25 个，各个具体参数如下所示：

对 S3 参数的说明：

S3 单元——采样时间(Ts) 范围为 1～32767(ms)但比运算周期短的时间数值无法执行；

S3+1 单元——动作方向(ACT)；

BIT0 为 0 正动作，为 1 逆动作；

BIT1 为 0 输入变化量报警无效，为 1 输入变化量报警有效；

BIT2 为 0 输出变化量报警无效，为 1 输出变化量报警有效；

BIT3 没使用；

BIT4 没使用；

BIT5 为 0 输出值上下限设定无效，为 1 输出值上下限设定有效；

BIT6～BIT15 没使用；

S3+2 单元——输入滤波常数(α)范围 0～99[%]，为 0 没有输入滤波；

S3+3 单元——比例增益(Kp)范围 1～32767[%]；

S3+4 单元——积分时间(TI)范围 0～32767(×100ms)，为 0 时作为∞处理(无积分)；

S3+5 单元——微分增益(KD)范围 0～100[%]，为 0 时无微分增益；

S3+6 单元——微分时间(TD)范围 0～32767(×10ms)，为 0 时无微分处理；

。。。。。。

S3+15 单元——输入变化量(增侧)报警设定值 0～32767(S3+1 的 BIT1=1 时)；

S3+16 单元——输入变化量(减侧)报警设定值 0～32767(S3+1 的 BIT1=1 时)；

S3+17 单元——输出变化量(增侧)报警设定值 0～32767(S3+1 的 BIT2=1 和 BIT5=0 时)；

输出上限设定值-32768～32767(S3+1 的 BIT2=0 和 BIT5=1 时)；

S3+18 单元——输出变化量(减侧)报警设定值 0～32767(S3+1 的 BIT2=1 和 BIT5=0 时)；

输出下限设定值-32768～32767(S3+1 的 BIT2=0 和 BIT5=1 时)；

S3+19 单元——报警输出；

BIT0 输入变化量(增侧)溢出；

BIT1 输入变化量(减侧)溢出；

BIT2 输出变化量(增侧)溢出；

BIT3 输出变化量(减侧)溢出；

指令向导最终会生成一个 PID 参数设置子程序（IL 类型），在这个子程序中设置以上这些信息。

6.2.3 PID 执行子程序

指令向导还会生成一个包含 PID 指令调用的子程序，该子程序会针对上面的参数设置，将合适的元件地址传入到 PID 指令的各个参数中。需要调用 PID 指令时，可以很方便的直接调用这个 PID 执行子程序，而无需自己去写各个参数。

指令格式：

PID	**S1**	S2	S3	S4

S1 为目标值用 SV 表示；

S2 为实际的测量值用 PV 表示；　执行程序时，运算结果 MV 被存入 D 中；

S3—S3+6 为设定控制参数；

6.3 数据块设置

数据块用于设置 D 寄存器中的缺省值，设置完成并通过编译后可以下载到可编程控制器中，在可编程控制器中的程序运行使用到 D 寄存器时，将使用所设置的值。

在数据块编辑器中，可以对 D 寄存器（数据内存）进行初始数据赋值，可以对 D 内存的字或双字赋值，但是不能对字节赋值。也可以在数据块编辑器中编写注释，在字符串前增加双斜线可以将其后的内容设为注释。

数据块赋值可以通过“ D 地址　值 1[,值 2][,值 N...] ”的方式为一个或者多个地址赋值，每个数值顺序占用下一个存储区域，系统自动根据值需要占用的字节数分配存储空间和下一个值的起始地址；在行的上方已经指定过起始地址的情况下，也可以通过直接输入数字“ 值 1[,值 2][,值 N...] ”来完成对存储空间的赋值。

对于值只需要占用一个字但是需要按照双字使用时，可以用 DD 寄存器，例如“DD100 10”表示将 10 作为一个双字存放到 D100 和 D101 寄存器表示的一个双字中。

数据块的第一行如果没有一个明确地址赋值，则默认从 D0 开始赋值，其后的行可包含明确或隐含地址赋值，已经显示或者隐含赋值的存储区域不能重复赋值。在单地址赋值后键入多个数据值或键入仅包含数据值的行时，由编辑器指定隐含地址赋值。编辑器根据先前的地址分配及数据值大小（字或双字）指定适当的 D 内存数量。

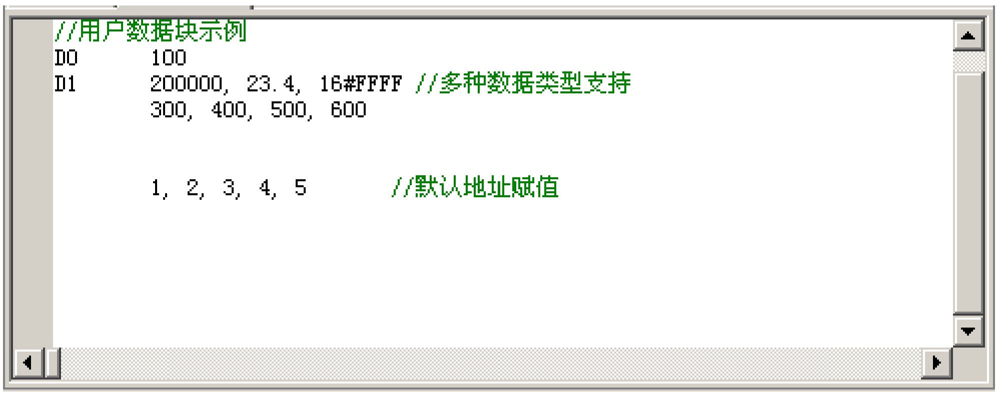
数据块编辑器是一种自由格式文本编辑器，支持文本编辑器中的常用编辑操作包括拷贝、剪切、粘贴和拖动。键入一行按 ENTER 键、粘贴完毕或者拖动后，数据块编辑器格式化行（对齐地址列、数据、注解；捕获 D 内存地址）并重新显示行。数据块编辑器接受大小写字母并允许使用逗号、制表符或空格，作为地址和数据值之间的分隔符。

数据块经过编译后，可以下载到 PLC 中使用。

6.3.1 数据块编辑

寄存器值设置

数据块编辑器是一个文本编辑器，用于输入寄存器地址、寄存器值和注释，它和指令列表编辑器是非常类似的，支持文本编辑器中的常用编辑操作包括拷贝、剪切、粘贴和拖动。在编辑器中输入需要的指令后，软件会自动对齐指令、操作数和注释并将指令关键字用蓝色显示，操作数用黑色显示，注释用绿色显示，如下图所示：

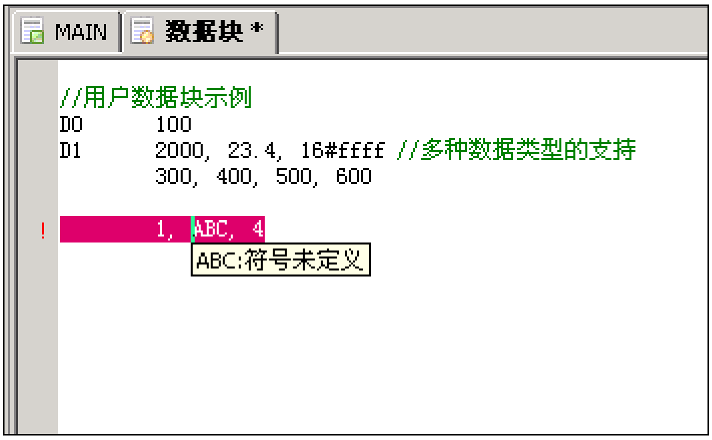


在编辑器的任意位置输入双斜线“//”表示当前行后面的内容是注释。

正确性校验

输入每一行指令后，如果光标切换到其他行，则会对离开的行进行正确性校验，如果有语法错误，软件会将有错误的文本行显示为红色，当鼠标悬浮在有错误的文本行时，软件会用工具提示的方式提示错误信息。只有校验正确的语句才会格式化显示，否则会在行头显示一个"!"并且用红色突出有语法错误的行。在一次粘贴多行文本内容到编辑器中时，编辑器会对每一行作语法检查。

有语法错误时，请将鼠标放在有错误的位置，软件显示如下图所示：



在图中，由于不能使用数值外的字符赋值，因此软件不能识别“ ABC ”。

6.3.2 数据块编译

编辑完成数据块后，点击工具栏上的按钮，可以检查当前数据块中是否有错误，如果无错误通过编译，则输出窗口中显示如下内容：

6.3.3 其他编辑功能

数据块编辑器和指令列表编辑器一样提供拷贝/剪切/粘贴，查找/替换/定位，标签和预览和打印功能，这些功能的使用请参阅指令列表编辑器操作的相关说明。

第七章 PLC 控制

PLC 控制功能是通过 ControlStar 编程软件通过串口对 PLC 完成各种控制操作，包括控制 PLC 的运行状态、改变 PLC 中存储的数据内容、从 PLC 中读取各种信息等等。

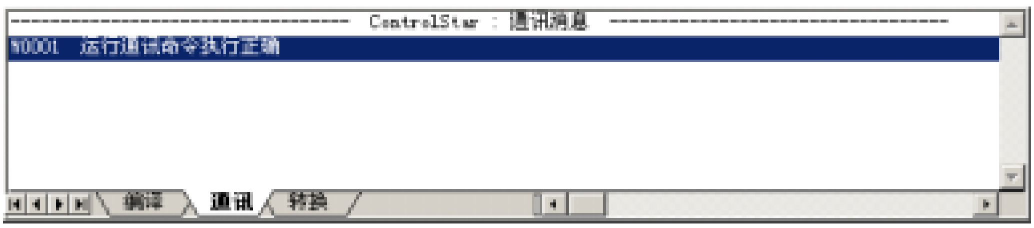
7.1 PLC 控制方法

在使用 PLC 控制功能前，需要将计算机和 PLC 硬件通过串口正确连接，在 ControlStar 编程软件中设置连接信息，设置方法请参阅 “ 通讯配置 ” 的相关章节。

在 ControlStar 编程软件中，和 PLC 控制相关的功能在主菜单的 “ PLC(P) ” 菜单中，常用的控制按钮也可以在 “ PLC 工具栏 ” 中找到。

控制 PLC 时，使用 “ PLC(P) ” 菜单中的某个子菜单，控制结果会在输出窗口的 “ 通讯 ” 页面中显示。

下图所示为控制 PLC 运行成功后的提示：



下图所示为控制 PLC 运行失败后的提示：

某些控制操作需要 PLC 处于停止状态，如果使用时 PLC 正处于运行状态，系统会提示是否自动停止 PLC 的运行，如下图所示：

点击“确定”按钮让软件停止 PLC 运行继续操作，点击取消按钮退出当前操作。

某些控制操作可能会使 PLC 运行数据丢失或者无法继续正常工作，包括 PLC 元件值清除、PLC 程序清除、PLC 数据块清除、PLC 格式化和恢复默认配置。在控制生效前，软件会显示确认对话框，请输入和对话框中显示的数字相同的内容后继续，确保这些控制操作不是误操作引起的。

注意

在使用上述控制前，请确保操作结果不会影响 PLC 的正常工作甚至引发其他意外情况。

7.2 PLC 启动

通过 ControlStar 软件启动 PLC 运行，无论 PLC 的开关处于哪个位置，都会使 PLC 启动运行，如果 PLC 已经运行，控制不会引起任何的变化。

7.3 PLC 停止

通过 ControlStar 软件停止 PLC 运行，无论 PLC 的开关处于哪个位置，都会使 PLC 停止运行，如果 PLC 已经停止，控制不会引起任何的变化。

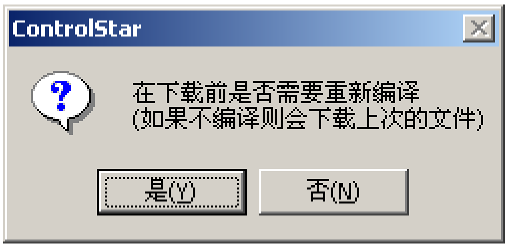
7.4 PLC 复位

可以通过 ControlStar 软件复位 PLC，PLC 复位后，停止执行用户程序，PLC 内部状态恢复到初始状态，然后重新启动运行。

7.5 下载

下载功能用于把 ControlStar 软件生成的系统块、数据块、用户程序通过串口下载到 PLC 中，下载时要求 PLC 处于停止运行状态。

下载时，如果上次全部编译后程序发生了改变，会提示是否需要重新编译程序，如下图所示：



选择“是”按钮重新编译程序，如果程序有错误，软件退出下载过程，需要先修改程序错误。选载“否”按钮不编译当前程序继续下载操作。

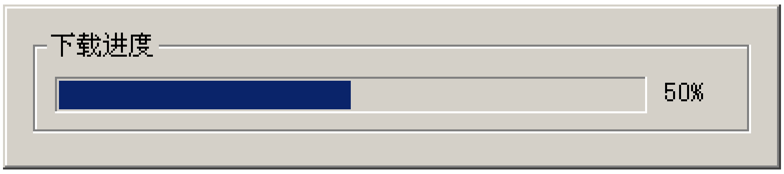
注意：选择“否”不重新编译，软件会使用上次编辑的结果，下载到 PLC 中运行的程序和软件界面显示的程序会不一样。

编译正确后，软件弹出下载窗口，选择要下载的内容。

选择好需要下载的数据项后，“下载”按钮变为可用，点击“下载”按钮下载数据到 PLC，也可以选择“关闭”按钮终止下载操作。

选择“下载”按钮后，如果设有下载密码，并且在启动软件后没有输入过下载密码，软件会弹出密码窗口要求输入写密码，如下图所示：

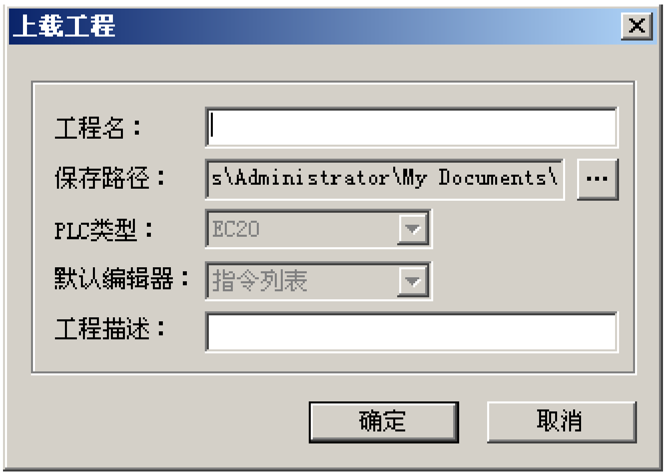
密码输入校验正确后开始下载，密码错误会提示重新输入密码。点击“取消”按钮退出下载。开始下载后软件显示下载进度，下载完毕后进度框自动消失。



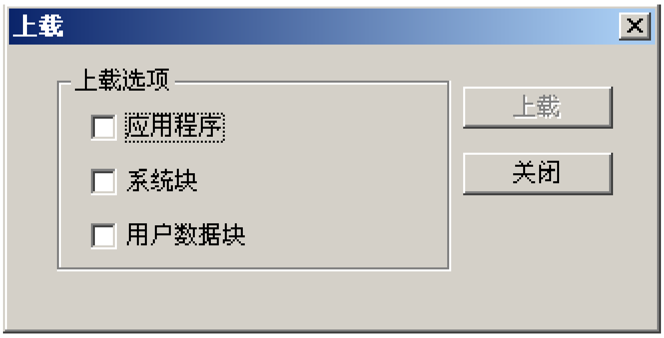
7.6 上载

上载功能用于把 PLC 中的系统块、数据块、用户程序等内容通过串口上载到计算机中，并且为其新建工程保存。

上载时软件会保存关闭已经打开的工程，自动新建工程保存上载内容，此时需要为上载内容指定工程名，点击“确定”继续上载过程，点击“取消”终止上载过程。

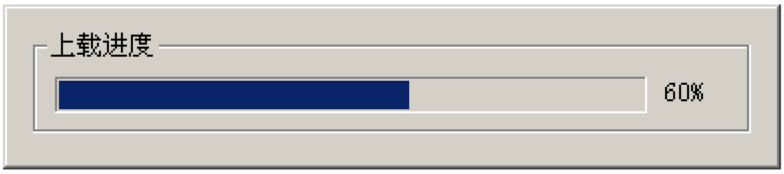


在上载对话框中指定要上载的内容，选择“关闭”按钮可以退出上载。



选择“上载”按钮后，如果没有上载密码，并且在启动软件后没有输过上载密码，软件会弹出密码窗口要求输入写密码，否则直接开始上载。

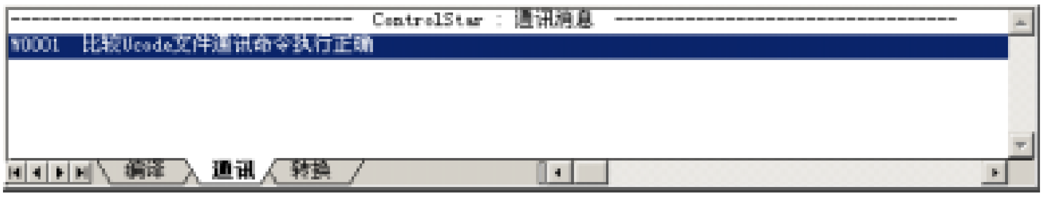
如果密码输入正确则开始上载，密码错误则软件提示并返回上载对话框界面。开始上载后，软件弹出上载进度框，显示上载进度，上载完毕后进度框自动消失。



上载成功后，软件会自动在工程中打开上载的主程序。

7.7 程序比较

程序比较功能用于比较计算机和 PLC 中用户程序是否相同，如果程序相同，信息输出窗口中显示执行正确的信息：



如果程序不相同，提示计算机和 PLC 中的程序不同：

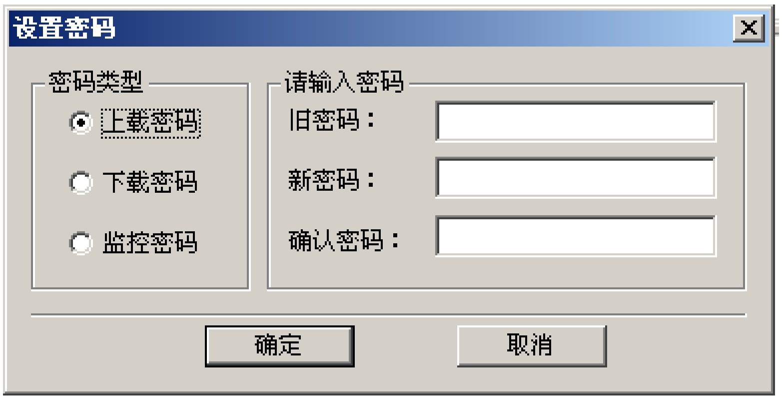
注意

在计算机和 PLC 中的用户程序一致时，如果使用转换功能，将当前的程序使用不同的编程语言显示，程序执行逻辑不会发生改变，但是可能会导致生成的用户程序和 PLC 中保存的用户程序不完全一致，此时使用比较功能也会提示不相同。

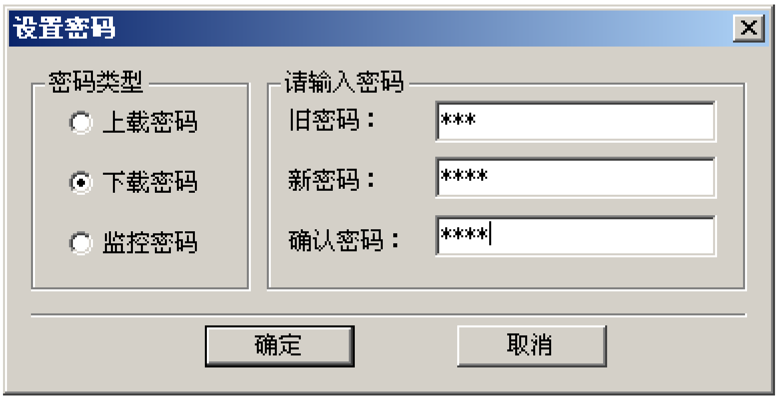
7.8 密码设置

密码设置功能用来设置 PLC 上载、下载、监控密码，设置好的密码将保存在 PLC 控制器中。

打开设置密码的窗口。



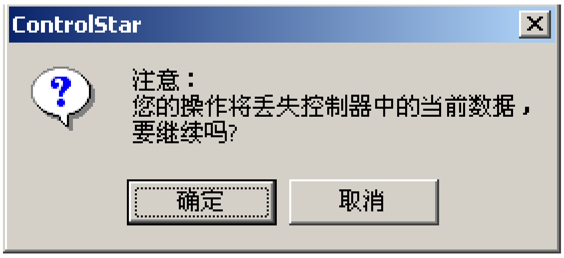
在密码设置对话框中，通过左边的选项选择密码类型，输入设置的新密码并且在“确认密码”中将新密码再输入一次。输入完毕点击“确定”按钮，如果 PLC 中已经设置过密码，需要在“旧密码”中输入原有的密码，只有密码正确，才允许修改。



7.9 PLC 元件值清除

PLC 元件值清除功能清除 PLC 中所有元件值，清除元件值需要 PLC 处于停止运行状态。

清除 PLC 中的元件值可能导致 PLC 运行不正确或者中间工作数据丢失，请谨慎使用此功能，操作时软件会显示确认窗口，继续操作选择“确定”按钮，退出操作选择“取消”按钮。



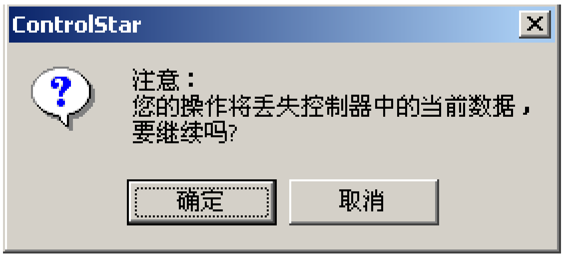
📖 注意

在系统块中设置为掉电保持的元件，清除元件值时也全部被清除。

7.10 PLC 程序清除

PLC 程序清除功能清除 PLC 中的用户程序，清除用户程序需要 PLC 处于停止运行状态。

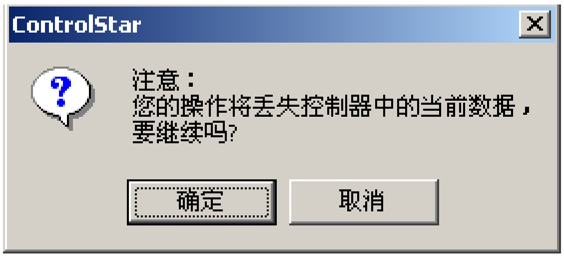
清除 PLC 中的用户程序会导致 PLC 运行后不执行任何用户程序，仅完成输入输出、通讯、内部处理功能，请谨慎使用此功能，操作时软件会显示确认窗口，继续操作选择 “ 确定 ” 按钮，退出操作选择 “ 取消 ” 按钮。



7.11 PLC 数据块清除

PLC 数据块清除功能清除 PLC 中所有数据块设置，清除数据块需要 PLC 处于停止运行状态。

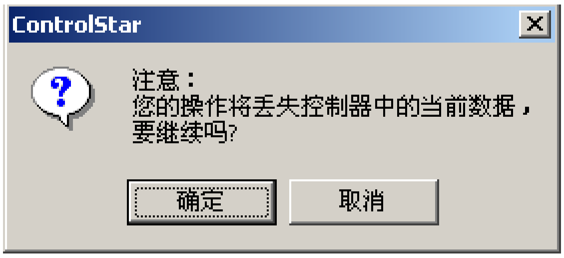
清除 PLC 中的数据块会导致 PLC 运行后不再使用数据块的预设值初始化 D 元件，请谨慎使用此功能，操作时软件会显示确认窗口，继续操作选择 “ 确定 ” 按钮，退出操作选择 “ 取消 ” 按钮。



7.12 PLC 格式化

PLC 格式化功能格式化 PLC 中所有数据，包括清除用户程序、恢复默认配置、清除清除数据块，清除数据块需要 PLC 处于停止运行状态。

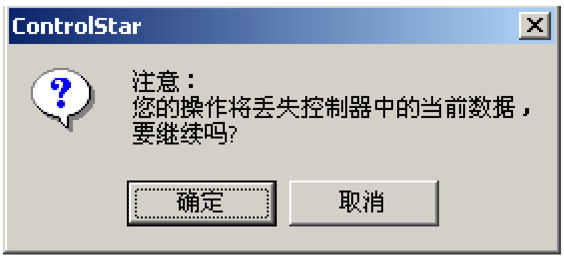
本操作会丢失 PLC 中已经下载和设置的全部数据，请谨慎使用此功能，操作时软件会显示确认窗口，继续操作选择“确定”按钮，退出操作选择“取消”按钮。



7.13 恢复默认设置

恢复默认值设置功能恢复 PLC 中所有设置为出厂设置，操作需要 PLC 处于停止运行状态。

本操作会将 PLC 中的设置恢复为出厂的默认配置，已经下载的系统块内容不再起作用，出厂默认配置和实际的使用情况可能不相符合，有可能导致 PLC 无法正确工作，请谨慎使用此功能，操作时软件会显示确认窗口，继续操作选择“确定”按钮，退出操作选择“取消”按钮。



7.14 PLC 信息

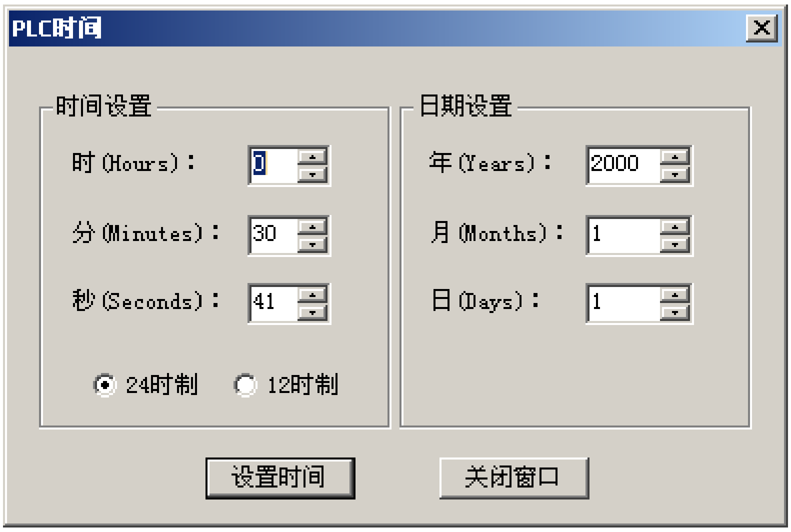
PLC 信息功能从 PLC 中获取并显示各种运行数据和 PLC 重要信息。

在信息显示窗口上可以看到 PLC 当前运行的重要信息，如下图所示：

点击“确定”按钮退出。

7.15 PLC 时钟

PLC 时钟功能用于显示和设置 PLC 当前时间，时钟窗口如下图所示：



窗口中显示当前从 PLC 中读取的日期和时间，可以输入新的日期和时间，点击“设置时间”按钮设置新的时间到 PLC 中。点击“关闭窗口”按钮退出 PLC 时钟窗口。

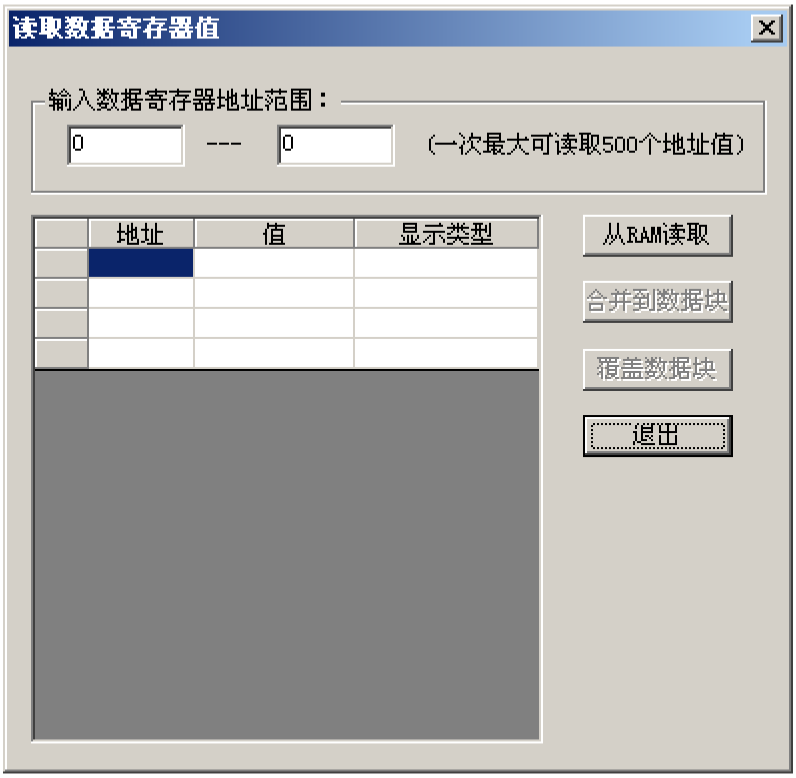
7.16 PLC 类型

PLC 类型功能显示当前 PLC 类型和版本号，如下图所示：

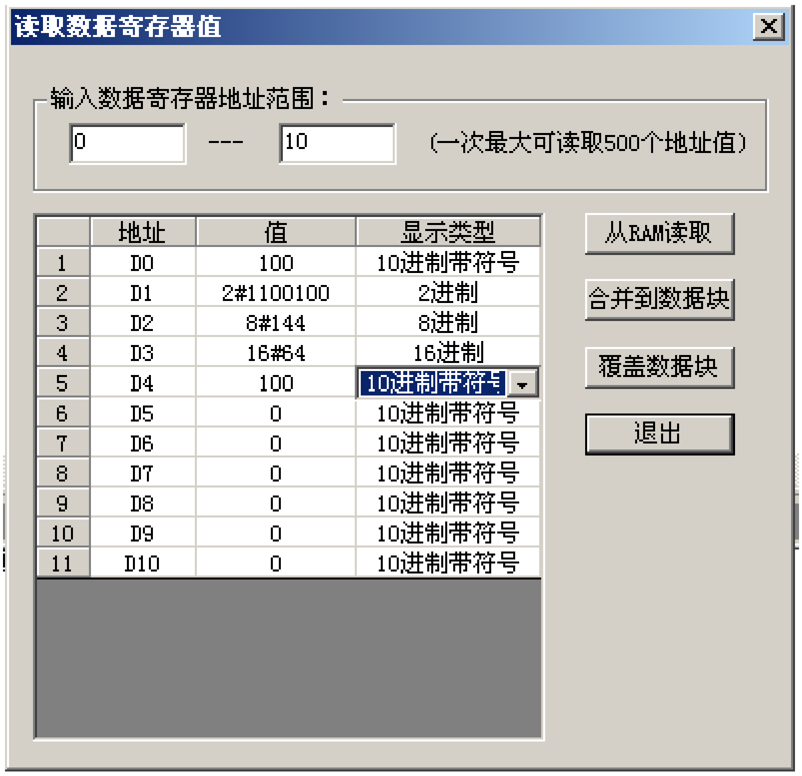
7.17 从 RAM 生成数据块

从 PLC 中连续读取最多 500 个 D 寄存器的数据值显示出来，结果可以合并到数据块中或者覆盖原来的数据块。

打开从 RAM 生成数块窗口，如下图所示：

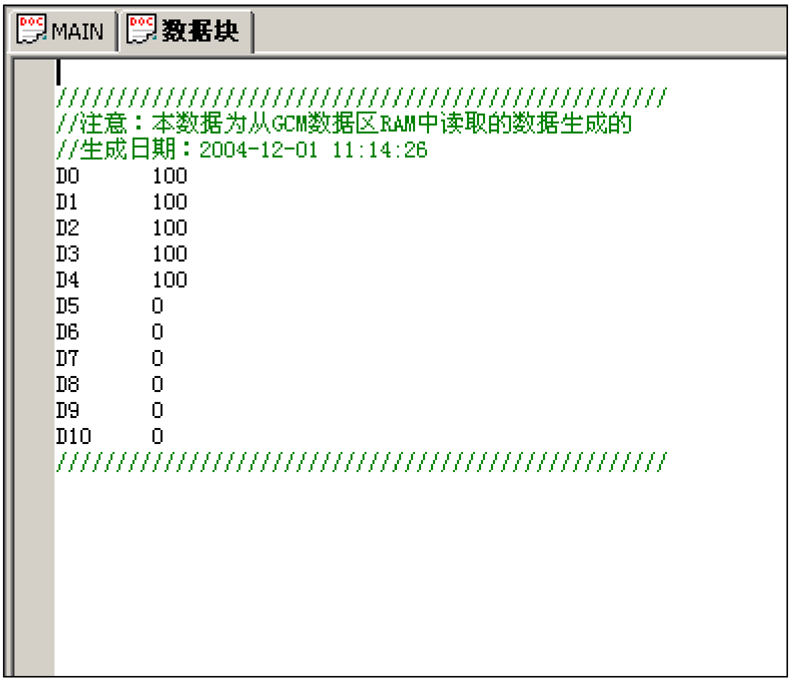


输入要读取的数据块的范围，点击“ 从 RAM 读取 ”按钮，执行正确后数据被读取到列表中。



可以在“ 显示类型 ”中选择以 16、10、8、2 进制来显示数据。

读取成功后“合并到数据块”和“覆盖数据块”按钮变为可用，点击“合并到数据块”按钮将生成结果追加到当前数据块的内容后面；点击“覆盖数据块”按钮将生成结果替代数据块中已经存在的内容。退出寄存器值读取窗口后，软件提示数据块已经发生变化并自动打开数据块窗口。



第八章 程序调试

程序下载到硬件设备中后，为了检测程序逻辑的正确性，可以对程序进行调试。需要进行程序调试，请确认编程设备与 PLC 硬件之间通讯正常，且 PLC 中的程序和当前打开的程序是一致的。

注意

程序下载或者上载后，如果将当前的程序使用不同的编程语言显示，程序执行逻辑不会发生改变，但是可能会导致生成的程序和 PLC 中的保存的程序不完全一致。

8.1 如何进入调试状态

选择 [调试/监控] 菜单或点击工具栏上的监控按钮（），软件会首先校验计算机和 PLC 中程序的一致性，不一致则会给出提示；然后会检查硬件设备中的程序是否需要监控密码，如果需要，必须正确输入监控密码后才能开始监控。如果上述检查都能通过的话，即可进入监控状态。

8.2 如何退出调试状态

需要停止调试时，同样选择[调试/监控]菜单或点击工具栏上的监控按钮（），程序会退出调试状态，恢复为编辑状态。

提示

1．在使用调试功能时，软件并不要求 PLC 硬件必须处于运行状态，但是如果需要调试硬件中的程序的逻辑正确性的话，建议将硬件设置为运行状态，否则调试结果可能会不准确。

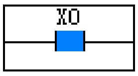
2．在调试状态下，软件定时到 PLC 硬件中读取软元件值，并显示在程序窗口中，即调试结果并不是实时的。同时，PLC 硬件的 CPU 循环扫描周期很短，远远小于软件定时读取的周期，因此，在调试时观察到的元件值和实际值比起来，必然有一定的滞后，在 9600 的串口通讯速率并且没有干扰的情况下，界面的元件值更新比 PLC 中有 0.2 秒的滞后，通讯不畅通的情况下，可能会有数秒的滞后。了解这一点，对正确把握程序逻辑及判断执行结果，都是非常重要的。

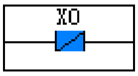
3．为了保证调试的效率，软件只在进入调试状态时检测计算机和 PLC 中程序的一致性，所以如果在调试过程中切换所连接的 PLC 的话，软件并不能识别这个操作，这样就会导致调试结果与实际运行的程序不一致。因此，建议不要进行这样的操作。

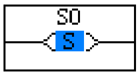
8.3 梯形图程序调试

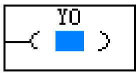
8.3.1 可调试的元件

在梯形图中，以下元件可以调试（即在调试时会根据值的变化有不同的显示）

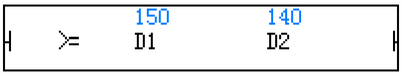
1．常开触点：当软元件值为 1 时，显示为连通状态，如图：

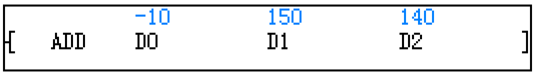
2．常闭触点：当软元件值为 0 时，显示为连通状态，如图：

3．步进触点：当状态位为 1 时，显示为连通状态，如图：

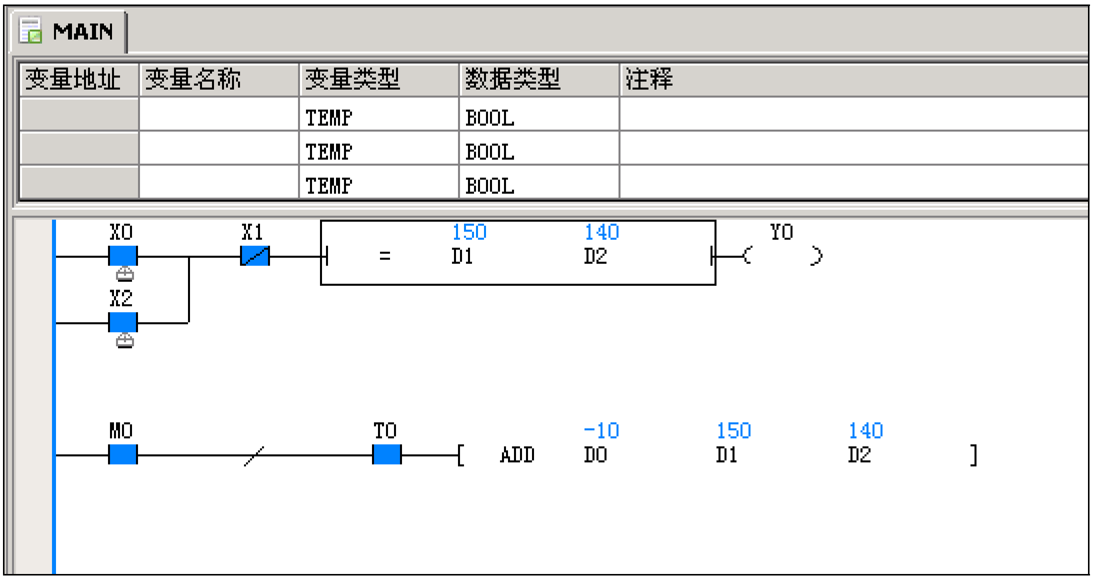
4．线圈：当软元件值为 1 时，显示为连通状态，如图：

5．比较触点及应用指令：此类元件引用多个地址，且每个地址不能简单的以逻辑值 1 和 0 表示，必须显示具体的数值，如图：





一个正在调试中的完整的梯形图程序如下图所示：

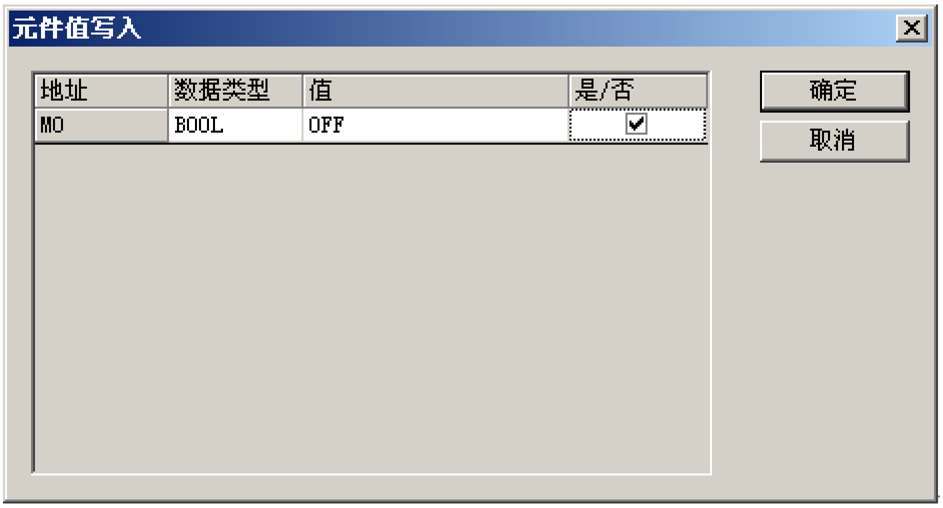


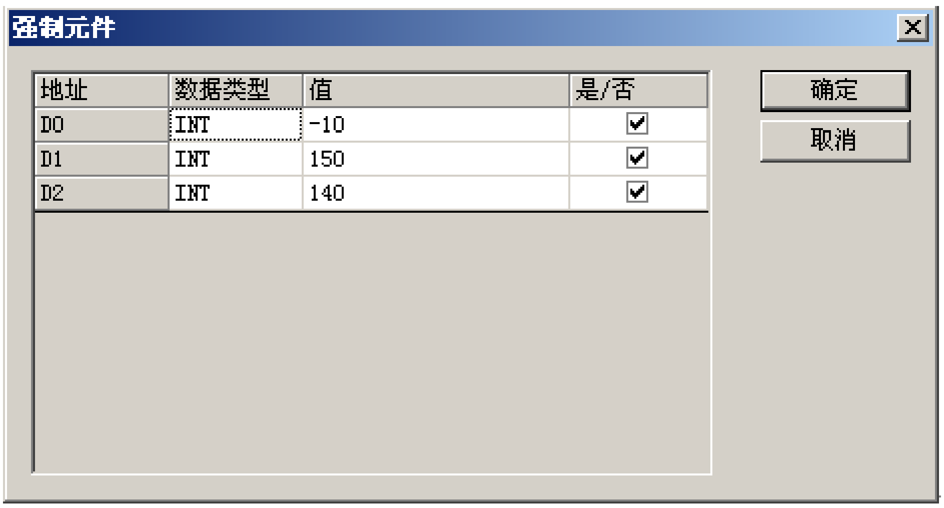
8.3.2 元件值写入和强制

在调试过程中，可能为了达成某些条件，而需要手工改变某些软元件的值，元件值写入和强制提供了这个功能。写入与强制的不同之处在于：写入元件值仅一次有效，写入后的值可能会随着程序的运行被改变，但是强制的元件值会一直记录在PLC 硬件中，直到取消强制为止。

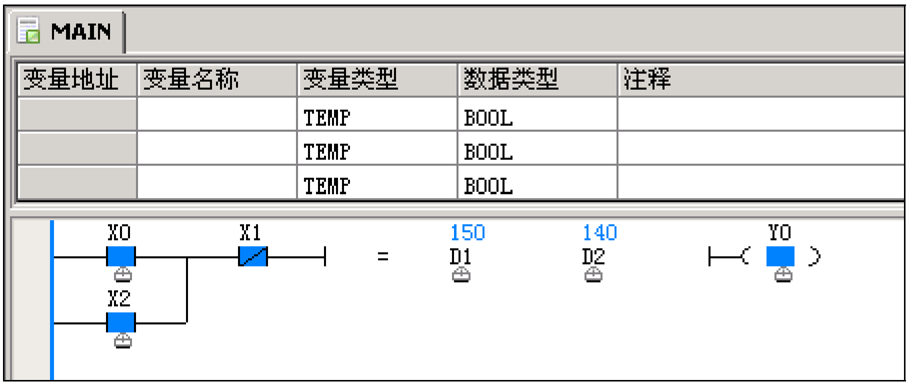
需要执行写入或强制功能时，首先选中要写入或强制的元件，从右键菜单中选择[写入]或者[强制]，此时会弹出相应对话框，列出所选元件引用到的所有软元件地址，可以有选择的写入或强制某些软元件值，确定后这些值就会下发到 PLC 硬件，当这些值在硬件中生效后，在后续的调试过程中就可以看到变化结果。

写入、强制对话框示意图如下：



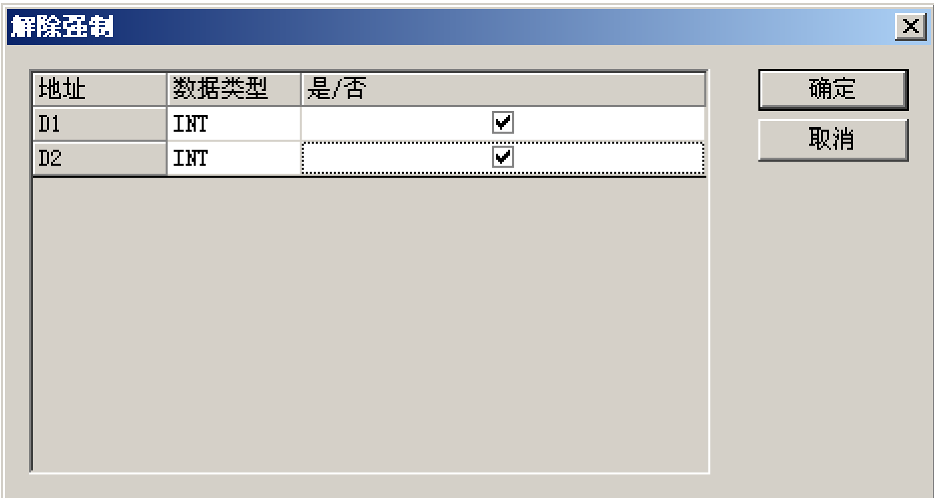


强制的软元件，在梯形图中会有一个锁标记，如图：



8.3.3 解除强制

对于不再需要强制的元件，可以将它解除强制。需要使用解除强制功能时，先选中要解除的元件，从右键菜单选择 [取消强制]，此时会弹出相应对话框，列出所选元件中已经被强制过的软元件，可以有选择的解除某些软元件的强制，点击确定后，这些强制值会从 PLC 硬件中删除，该软元件对应的锁标记也会消失。



8.3.4 强制结果及限制规则

强制结果的显示及其含义对照表：

强制结果	说明	界面显示
没有强制	该地址没有任何强制信息	无
部分强制（左）	某个双字地址的左半个单字被强制，而右半个单字没有被强制	
部分强制（右）	某个双字地址的右半个单字被强制，而左半个单字没有被强制	
间接强制	某个双字地址被强制，则它所包含的两个单字地址都被间接强制	
完全强制	对于位地址或单字地址，当它被直接强制时； 对于双字地址，当它包含的两个单字地址都被强制时	

写入和强制的限制

1. 只读的元件和常数不能够被写入或强制；

2. LM 和 V 元件不能被写入或强制；

3. 带变址的元件不能被写入或强制；

4. 所有 Kn 寻址的元件都不能被强制，Kn 寻址的 SM 元件不能被写入；

5. SPD 指令在执行时，对它的输入端口（操作数 1）使用的软元件强制无效，PLSY、PWM 指令在执行时，对它的输出端口（操作数 3）使用的软元件强制无效，PLSR 指令在执行时，对它的输出端口（操作数 4）使用的软元件强制无效；

6. 对于单字元件，如果它所属的双字元件被强制，则它不能再被强制；

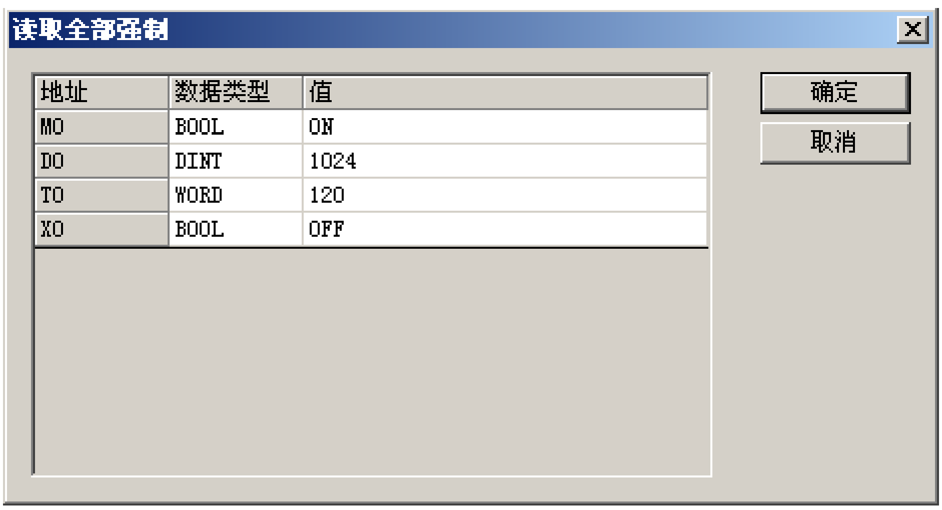
7. 对于双字元件，如果所包含的两个单字软元件任一个被其他元件强制过，则它不能再被强制；

解除强制的限制

对应数据长度（位、单字、双字）的软元件被直接强制时，才能够解除强制。

8.3.5 读取所有强制元件值

由于被强制的所有软元件都是保存在 PLC 硬件中的，因此在调试时可以将这些被强制的软元件读取上来查看。从右键菜单选择 [读取所有强制]，即弹出读取全部强制对话框，显示当前已被强制的所有软元件及强制值。如图：



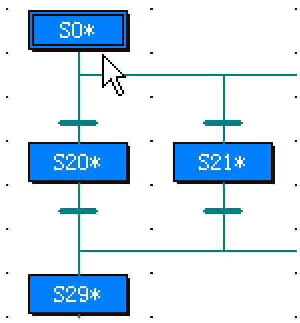
8.4 指令列表程序调试

指令列表程序的调试功能比较简单，在监控状态下，在有可监控的软元件的代码行点击鼠标右键，会看到“ 添加到监控表 ”菜单项，点击该菜单项就可完成将当前行的可监控软元件添加到监控表的操作。可以在监控表中查看被添加的软元件的监控值。

8.5 顺序功能图程序调试

8.5.1 可调试的元件

只有初始步进符和普通步进符元件可以被调试。当他们对应的 S 软元件状态为“ ON ”的时候，他们显示为通路：

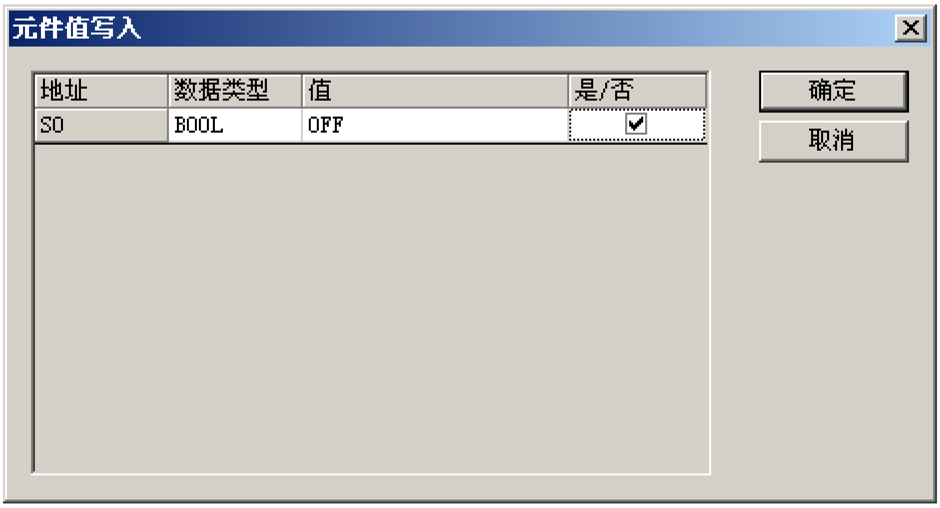


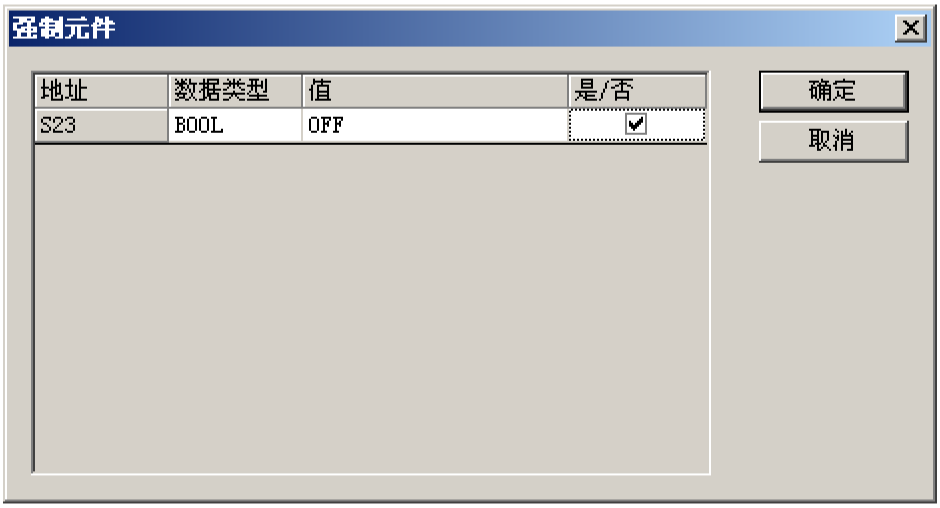
8.5.2 元件值写入和强制

在调试过程中，可能为了达成某些条件，而需要手工改变某些软元件的值，元件值写入和强制提供了这个功能。写入与强制的不同之处在于：写入元件值仅一次有效，写入后的值可能会随着程序的运行被改变，但是强制的元件值会一直记录在 PLC 硬件中，直到取消强制为止。

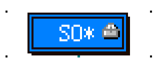
需要执行写入或强制功能时，首先选中要写入或强制的元件，从右键菜单中选择 [写入] 或者 [强制]，此时会弹出相应对话框，列出所选元件引用到的所有软元件地址，可以有选择的写入或强制某些地址值，确定后这些值就会下发到 PLC 硬件，当这些值在硬件中生效后，在后续的调试过程中就可以看到变化结果。

写入、强制对话框示意图如下：





强制的软元件地址，在顺序功能图中会有一个锁标记，如图：

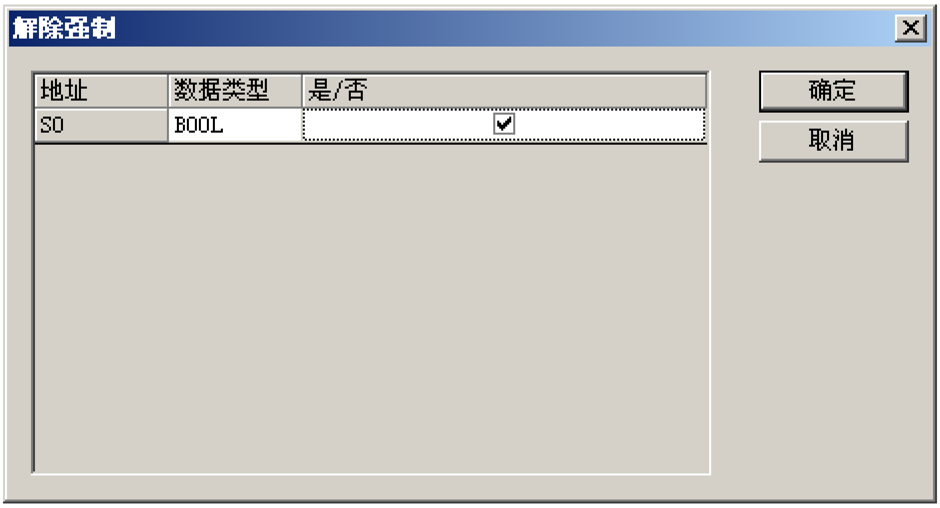


8.5.3 解除强制

对于不再需要强制的元件，可以将它解除强制。需要使用解除强制功能时，先选中要解除的元件，从右键菜单选择 [取消强制]，此时会弹出相应对话框，列出所选元件

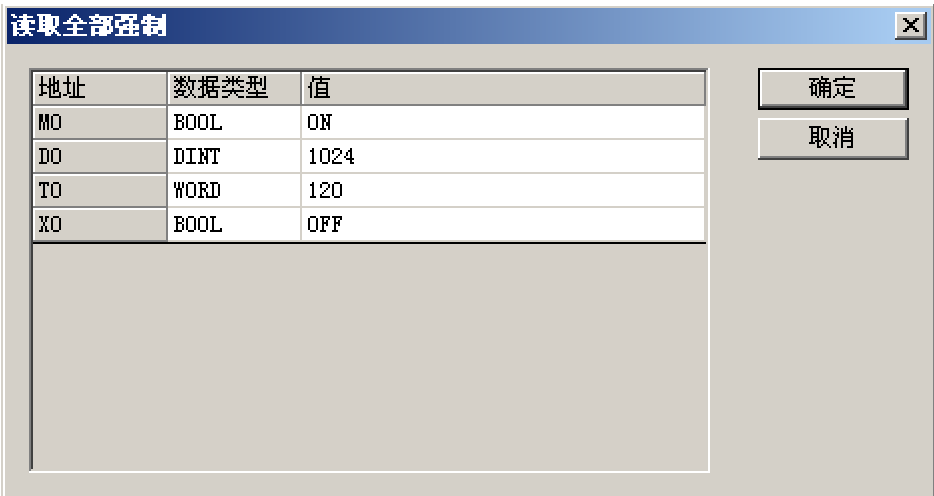
中已经被强制过的软元件地址，可以有选择的解除对某些地址的强制，点击确定后，这些强制值会从 PLC 硬件中删除，该地址对应的锁标记也会消失。

取消强制对话框如图：



8.5.4 读取所有强制元件值

由于被强制的所有元件地址都是保存在 PLC 硬件中的，因此在调试时可以将这些被强制的地址读取上来查看。从右键菜单选择 [读取所有强制]，即弹出读取全部强制对话框，显示当前已被强制的所有软元件地址及强制值。如图：



8.6 元件状态监控表

元件状态监控表提供了在调试过程中监控元件值的功能，调试程序时允许将程序输入元件、输出元件置入元件监控表中以便在将程序下载至 PLC 之后追踪其状态。

元件监控表有两种模式：编辑辑模式和监控模式。编辑模式下可以执行所有的编辑功能，不能执行监控的功能。在监控模式下，可以同时执行监控功能和编辑功能。

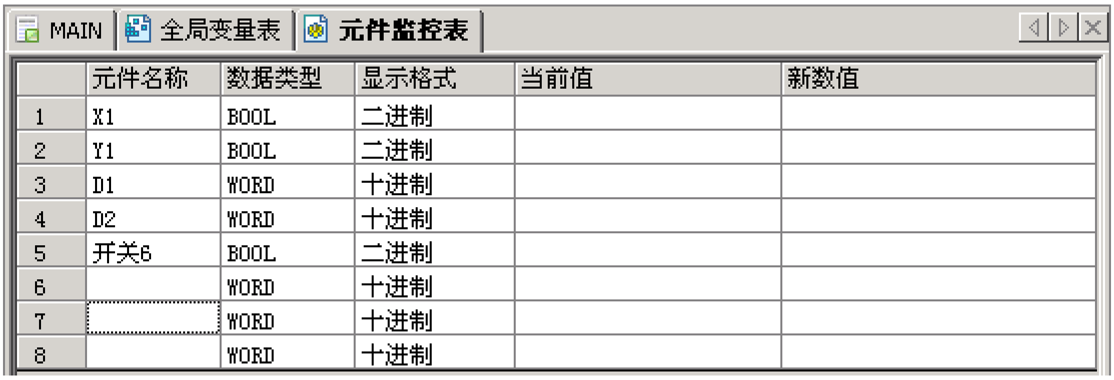
在线梯形图功能中，修改元件值和强制元件后，元件自动被加入元件状态监控表。元件状态监控表在监控模式下自动刷新元件值。

元件状态监控表可提供编辑功能、排序、查找、自动刷新显示指定元件的当前值、写入元件值、强行指定元件/变量的值、解除强制等功能。

元件状态监控表的字段包括：

- 元件地址
- 数据类型
- 显示格式
- 当前值
- 新数值（写入值）

下图是元件状态监控表的示意图：



	元件名称	数据类型	显示格式	当前值	新数值
1	X1	BOOL	二进制		
2	Y1	BOOL	二进制		
3	D1	WORD	十进制		
4	D2	WORD	十进制		
5	开关6	BOOL	二进制		
6		WORD	十进制		
7		WORD	十进制		
8		WORD	十进制		

8.6.1 编辑模式与监控模式

元件监控表的监控模式和编辑模式可以相互切换，从编辑状态进入监控状态，读取 PLC 数据，开始刷新元件值，基本编辑功能不能使用，可以开始监控；从监控状态退出到编辑状态，停止读取 PLC 数据和刷新元件状态，监控功能不能执行，可以开始编译。

操作：在编辑模式下从主菜单调试—>监控，或者点击图标，将进入监控模式；在监控模式下执行相同的操作将进入编辑模式。

在监控模式下，除了具有编辑功能外还提供以下功能。

元件值的写入功能

功能：输入要写入的元件值，系统将对输入内容作合法性检查（写入数据类型，写入值等），合法的数据在执行写入功能后被写入 PLC，并立即刷新该元件当前值。不合法的输入不能执行写入功能，并提示输入错误。

操作：选定元件后，点击右键，在弹出菜单中选择“ 写入元件值 ”。

注意：

有些元件不可以被写入，具体的元件参见梯形图程序的调试。

元件强制功能

功能：将选定的元件状态设置为强制状态，如果这时在值写入区有合法值，则元件进入强制状态，同时值写入区的值被写入 PLC。强制后能在元件状态监控表中明显看到元件被强制，同时在线梯形图中也要能被识别出来。

操作：选定元件后，点击右键，在弹出菜单中选择“ 强制 ”

注意：

有些元件不可以被强制，具体元件参见梯形图程序调试。

元件解除强制功能

功能：将选择某个被强制的元件解除强制，也可以选择解除全部被强制元件强制功能，强制解除后，在元件状态监控表和在线梯形图上可以明显看到元件强制被解除。

操作：选定元件后，点击右键，在弹出菜单中选择“ 解除强制 ”

注意：

某些元件不可以被强制，具体元件参见梯形图程序调试

元件显示功能

功能：元件状态监控表中的元件可以显示为不同的显示格式和数据类型，显示格式包括二进制、十进制、十六进制三种格式，因为不同的元件支持不同的数据类型，在输入元件地址编辑数据类型时，数据类型列以下拉列表的方式显示该元件支持的所有数据类型。

📖 注意：

元件值可能被显示为半个锁、灰锁等不同的形式，具体请参见梯形图程序的调试。

8.6.2 切换编址方式

在元件监控表中输入了元件地址后，如果该元件地址在全局变量表中有定义，可以通过选择 [符号编址] 菜单在变量名和元件地址之间切换， 地址编址模式下所有元件显示为元件地址。

符号编址模式下在全局变量表中定义的地址显示为全局变量名称，没有定义的显示为元件地址。注意：局部变量不能切换为符号编址。

8.7 在线修改

需要在 PLC 运行状态下修改其中的程序时，可以使用 “ 在线修改 ” 功能。

如何操作

点击 [调试/在线修改] 菜单可以切换 “ 在线修改 ” 状态，使用此功能之前，请确保软件已经与 PLC 硬件成功建立了通讯关系，且 PLC 处于运行状态。

在 “ 在线修改 ” 状态下，可以与正常编辑时一样修改主程序、子程序及中断子程序的内容，修改完之后，点击[PLC/下载] 菜单，软件将会编译当前工程的所有程序并自动下载至 PLC 硬件中。下载完成后，PLC 就会按照新下载的程序运行。

限制条件

1．“ 在线修改 ” 状态下，不能修改全局变量表及任何程序的局部变量表，也不能添加或删除任何子程序/中断子程序；

2．当程序处于 “ 在线修改 ” 状态下时，如果 PLC 被停止，软件会自动退出 “ 在线修改 ” 状态；

附录一 软元件说明

软元件定义

元件标号	元件名	项目	范围	功能描述
X	外部输入 继电器		X000～X377(8 进制编码)： 256 点	按 X00-X007，X010-X017…… ；Y000-Y007，Y010-Y017……8 进制计数方式分配输入继电器、输出继电器的地址号，扩展单元、扩展模块的地址号。
Y	外部输出 继电器		Y000～Y377(8 进制编码)： 256 点	
M	辅助继电器	M0～M1999：2000 点		辅助继电器是 PLC 的内部继电器。 电池保持元件范围可设置
LM	局部辅助继电器		LM0～LM63 64 点	局部有效的辅助继电器
SM	特殊辅助继电器		SM0～SM255 256 点	系统功能的辅助继电器
S	步进状态 继电器		S0～S127： 128 点	用于 SFC 中表示工序号使用的继电器，它不作为工序号使用时，可作为普通辅助继电器使用。也可以作为信号报警器，用于外部故障诊断。合计 992 点，电池保持元件范围可设置
			S128～S991： 864 点	
T	计时器	100ms	T0～T209， 共 210 点	合计 256 点。电池保持元件范围可设置
		10ms	T210～T251，共 42 点	
		1ms	T252-T255，共 4 点	
C	计数器	普通增计数器（16 位）	一般用：C00～C199， 共 200 点	16 位计数器：供增计数 计数范围：1-32，767 合计 200 点 电池保持元件范围可设置
		增/减型计数器（32 位）	C200～C235， 共 36 点	32 位计数器：供增减数计数 计数范围：-2，147，483，648～2，147，483，647 合计 35 点
		高速计数器	C236～255	
D	数据寄存器		D000～D7899， 共 7900 点	合计 8000 点 电池保持元件范围可设置
			D7900～D7999 共 100 点	
V	局部数据寄存器		V0～V63	在各程序中局部有效的数据寄存器

元件标号	元件名	项目	范围	功能描述
Z	变址寻址寄存器		Z0～Z15 共 16 点	用于变址寻址方式
SD	特殊数据寄存器		SD0～SD255	系统功能的数据寄存器
10 进制常数 (格式例：7890)		16 位	－32,768～32,767	常数用来作为定时器与计数器的设定值和当前值，还可以作为应用指令的操作数
		32 位	－2,147,483,648～2,147,483,647	
16 进制常数 （格式例：16#3AB）		16 位	0～FFFFH	
		32 位	0～FFFFFFFFH	
浮点常数数 (格式例：3.14)		32 位	负数范围： -3.402823E38～-1.401298E-45， 正数范围： +1.401298E-45～ +3.402823E38。	IEEE-754 格式

元件的间接寻址方式

提供变址寻址方式，局部数据寄存器 Z 为变址寄存器使用。

位元件的变址寻址例举

LD M1

MOV 6 Z1

SFTR X0Z1 M0 8 2

以上命令实际等价于：

LD M1

SFTR X6 M0 8 2

其寻址过程如下：

Z1=6

X0Z1 = X(0+ Z1) = X6

位串的变址寻址例举

LD M1

MOV 3 Z10

MOV K1X0Z10 D0

以上命令等价于：

LD M1

MOV K1X3 D0

其寻址过程如下：

Z10=3

K1X0Z10＝K1X（0+Z10）＝K1X3

字元件的变址寻址例举

LD M1

MOV 30 Z20

MOV D100Z20 D0

以上命令等价于：

LD M1

MOV D130 D0

其寻址过程如下：

Z20=30

D100Z20＝D（100+ Z20）＝D130

附录二 指令一览表

本处列出指令一览表，指令详细描述请参考指令手册。

指令类别	指令	指令功能说明	备注
基本指令	LD	常开触点能流装载	
	LDI	常闭触点能流装载	
	AND	常开触点能流与	
	ANI	常闭触点能流与	
	OR	常开触点能流或	
	ORI	常闭触点能流或	
	OUT	能流输出	
	SET	置位	
	RST	清零	
	ANB	能流块与	
	ORB	能流块或	
	INV	能流取反	
	NOP	空操作	
	MPS	输出能流入栈	
	MRD	读输出能流栈顶值	
	MPP	输出能流栈出栈	
	EU	能流上升沿检测	
	ED	能流下降沿检测	
	TON	接通延时计时指令	
	TOF	断开延时计时指令	
	TMON	单稳计时指令	
	TONR	接通延时记忆计时指令	
	CTU	16 位计数器增计数指令	
	CTR	16 位计数器循环计数指令	
	DCNT	32 位计数指令	

指令类别	指令	指令功能说明	备注
程序流控制指令	LBL	跳转标号定义	
	CJ	条件跳转	
	CALL	用户子程序调用	
	CSRET	用户子程序条件返回	
	CFEND	用户主程序条件结束	
	CIRET	用户中断子程序条件返回	
	FOR	循环指令	
	NEXT	循环返回	
	MC	主控	
	MCR	主控清除	
	WDT	用户程序看门狗清零	
	STOP	用户程序停止	
	EI	中断使能指令	
	DI	中断禁止指令	
顺序功能图指令	STL	顺序功能图状态装载指令	
	SET Sxx	顺序功能图状态转移	
	OUT Sxx	顺序功能图状态跳转	
	RST Sxx	顺序功能图状态清除	
	RET	顺序功能图程序结束	
数据传输指令	MOV	字数据传输	
	DMOV	长字数据传输指令	
	RMOV	浮点数数据传输指令	
	BMOV	块数据传输指令	
	SWAP	上下字节交换指令	
	XCH	字交换指令	
	DXCH	长字换指令	
	FMOV	块数据填充	
	DFMOV	数据块双字填充指令	
	WSFR	字串右移指令	
	WSFL	字串左移指令	
	PUSH	加入队列	
	FIFO	先入先出指令	
	LIFO	后入先出指令	

指令类别	指令	指令功能说明	备注
整数/长整数算术运算指令	ADD	整数加法指令	
	DADD	长整数加法指令	
	SUB	整数减法指令	
	DSUB	长整数减法指令	
	INC	整数增一指令	
	DINC	长整数增一指令	
	DEC	整数减一指令	
	DDEC	长整数减一指令	
	MUL	整数乘法指令	
	DMUL	长整数乘法指令	
	DIV	整数除法指令	
	DDIV	长整数除法指令	
	VABS	整数绝对值指令	
	DVABS	长整数绝对值指令	
	NEG	整数取负指令	
	DNEG	长整数取负指令	
	SQT	整数平方根指令	
	DSQT	长整数平方根指令	
浮点算术运算指令	RADD	浮点数加法指令	
	RSUB	浮点数减法指令	
	RMUL	浮点数乘法指令	
	RDIV	浮点数除法指令	
	RVABS	浮点数绝对值指令	
	RNEG	浮点数取负指令	
	RSQT	浮点数平方根指令	
	SIN	浮点数 SIN 指令	
	COS	浮点数 COS 指令	
	TAN	浮点数 TAN 指令	
	LN	浮点数自然对数指令 LN	
	EXP	浮点数自然数幂指令 EXP	
	POWER	浮点数求幂指令	
累加器指令	SUM	16 位累加器	
	DSUM	32 位累加器	
	RSUM	浮点累加器	

指令类别	指令	指令功能说明	备注
字/长字逻辑运算	WAND	字与指令	
	DWAND	长字与指令	
	WOR	字或指令	
	DWOR	长字或指令	
	WXOR	字异或指令	
	DWXOR	长字异或指令	
	WINV	字取非指令	
	DWINV	长字取非指令	
移位转指令	ROR	字位循环右移指令	
	DROR	长字位循环右移指令	
	ROL	字位循环左移指令	
	DROL	长字位循环左移指令	
	RCR	字位带进位循环右移指令	
	DRCR	长字位带进位循环右移指令	
	RCL	字位带进位循环左移指令	
	DRCL	长字带进位循环左移指令	
	SHR	字位右移指令	
	DSHR	长字位右移指令	
	SHL	字左移指令	
	DSHL	长字位左移指令	
	SFTL	位串左移指令	
	SFTR	位串右移指令	
增强型位处理指令	DECO	解码指令	
	ENCO	编码指令	
	BITS	字中 ON 位统计指令	
	DBITS	长字中 ON 位统计指令	
	ZRST	批量位清零指令	
	ZSET	批量位置位指令	

指令类别	指令	指令功能说明	备注
高速 IO	HCNT	高速计数器驱动指令	
	DHSCS	高速计数比较置位指令	
	DHSCR	高速计数比较复位指令	
	DHSCI	高速计数比较中断触发指令	
	DHSZ	高速计数区间比较指令	
	DHST	高速计数表格比较指令	
	DHSP	高速计数表格比较脉冲输出指令	
	SPD	测频指令	
	PLSY	计数脉冲输出指令	
	PLSR	带加减速的计数脉冲输出指令	
	PWM	PWM 脉冲输出指令	
控制计算指令	PID	PID 功能指令	
	RAMP	斜坡信号输出指令	
	HACKLE	锯齿波发生器指令	
	TRIANLE	三角波函数发生器指令	
外设指令	FROM	字读特殊模块缓冲寄存器指令	
	DFROM	长字读特殊模块缓冲寄存器指令	
	TO	字写特殊模块缓冲寄存器指令	
	DTO	长字写特殊模块缓冲寄存器指令	
	VRRD	读模拟电位器值指令	
	REFF	设置输入滤波常数指令	
	REF	刷新 I/O 镜像指令	
实时时钟指令	TRD	实时时钟读指令	
	TWR	实时时钟写指令	
	TADD	时间加指令	
	TSUB	时间减指令	
	HOUR	小时计指令	

指令类别	指令	指令功能说明	备注
日期时间比较指令	DCMP=	日期等于比较	
	DCMP<	日期小于比较	
	DCMP>	日期大于比较	
	DCMP<=	日期小于等于比较	
	DCMP>=	日期大于等于比较	
	DCMP<>	日期不等于比较	
	TCMP=	时间等于比较	
	TCMP<	时间小于比较	
	TCMP>	时间大于比较	
	TCMP<=	时间小于等于比较	
	TCMP>=	时间大于等于比较	
	TCMP<>	时间不等于比较	
比较触点指令	LD=	整数比较 LD=指令	
	LDD=	长整数比较 LD=指令	
	LDR=	浮点数比较 LD=指令	
	LD=	整数比较 LD=指令	
	LDD>	长整数比较 LD>指令	
	LDR>	浮点数比较 LD>指令	
	LD>=	整数比较 LD>=指令	
	LDD>=	长整数比较 LD>=指令	
	LDR>=	浮点数比较 LD>=指令	
	LD<	整数比较 LD<指令	
	LDD<	长整数比较 LD<指令	
	LDR<	浮点数比较 LD<指令	
	LD<=	整数比较 LD<=指令	
	LDD<=	长整数比较 LD<=指令	
	LDR<=	浮点数比较 LD<=指令	
	LD<>	整数比较 LD<>指令	
	LDD<>	长整数比较 LD<>指令	
	LDR<>	浮点数比较 LD<>指令	
	AND=	整数比较 AND=指令	
	ANDD=	长整数比较 AND=指令	
	ANDR=	浮点数比较 AND=指令	
	AND>	整数比较 AND>指令	
	ANDD>	长整数比较 AND>指令	

指令类别	指令	指令功能说明	备注
	ANDR>	浮点数比较 AND>指令	
	AND>=	整数比较 AND>=指令	
	ANDD>=	长整数比较 AND>=指令	
	ANDR>=	浮点数比较 AND>=指令	
	AND<	整数比较 AND<指令	
	ANDD<	长整数比较 AND<指令	
	ANDR<	浮点数比较 AND>指令	
	AND<=	整数比较 AND<=指令	
	ANDD<=	长整数比较 AND<=指令	
	ANDR<=	浮点数比较 AND<=指令	
	AND<>	整数比较 AND<>指令	
	ANDD<>	长整数比较 AND<>指令	
	ANDR<>	浮点数比较 AND<>指令	
	OR=	整数比较 OR=指令	
	ORD=	长整数比较 OR=指令	
	ORR=	浮点数比较 OR=指令	
	OR>	整数比较 OR>指令	
	ORD>	长整数比较 OR>指令	
	ORR>	浮点数比较 OR>指令	
	OR>=	整数比较 OR>=指令	
	ORD>=	长整数比较 OR>=指令	
	ORR>=	浮点数比较 OR>=指令	
	OR<	整数比较 OR<指令	
	ORD<	长整数比较 OR<指令	
	ORR<	浮点数比较 OR>指令	
	OR<=	整数比较 OR<=指令	
	ORD<=	长整数比较 OR<=指令	
	ORR<=	浮点数比较 OR<=指令	
	OR<>	整数比较 OR<>指令	
	ORD<>	长整数比较 OR<>指令	
	ORR<>	浮点数比较 OR<>指令	

指令类别	指令	指令功能说明	备注
数值转换指令	ITD	整数转换长整数指令	
	DTI	长整数转换整数指令	
	FLT	整数转换浮点数指令	
	DFLT	长整数转换浮点数指令	
	INT	浮点数转换整数指令	
	DINT	浮点数转换长整数指令	
	BCD	整数转换为 BCD 整数指令	
	DBCD	长整数转换为 BCD 整数指令 DBCD	
	BIN	BCD 整数转换整数指令	
	DBIN	BCD 整数转换整数指令	
	GRY	整数转换为格雷码指令	
	DGRY	长整数转换为格雷码指令	
	GBIN	格雷码转换为整数指令	
	DGBIN	32 位格雷码转换为长整数指令	
	SEG	BCD 码转换位 7 段码	
字触点指令	BLD	字位触点 LD 指令	
	BLDI	字位触点 LDI 指令	
	BAND	字位触点 AND 指令	
	BANI	字位触点 ANI 指令	
	BOR	字位触点 OR 指令	
	BORI	字位触点 ORI 指令	
	BSET	字位触点置位指令	
	BRST	字位触点清除指令	
	BOUT	字位触点输出指令	
通信指令	XMT	自由口发送（XMT）指令	
	RCV	自由口发送（RCV）指令	
	MODBUS	modbus 协议指令	
数据校验指令	CCITT	CCITT 校验指令	
	CRC16	CRC16 校验指令	
	LRC	LRC 校验指令	

附录三 编译/转换错误代码处理表

错误代码	W0000
错误简述	中断子程序没有指定中断号
错误原因	工程中存在一个没有分配中断号的中断子程序，这个中断子程序将不会被编译。
解决措施	从工程树上右键点击该子程序，选择[属性]菜单，在弹出对话框中为它指定中断号。

错误代码	W2001
错误简述	通讯命令执行正确
错误原因	该信息表明软件和 PLC 进行了正常通讯，没有错误发生
解决措施	

错误代码	W2002
错误简述	通讯命令执行错误
错误原因	软件和 PLC 进行通讯时发生未知错误。
解决措施	先重试，若仍然出现此问题，应检查计算机串口或 PLC 硬件是否存在问题。

错误代码	W2003
错误简述	通讯命令超时
错误原因	软件和 PLC 进行通讯时超时。
解决措施	1．重试。 2．检查软件和 PLC 是否正确连接。 3．检查软件当前串口设置是否正确。

错误代码	W2004
错误简述	串口不能打开
错误原因	软件当前设置的串口不能打开。
解决措施	1．检查当前串口是否正被其他应用程序占用，如果是请关闭该应用程序。 2．检查当前配置的串口是否有效。 3．重新启动计算机。

错误代码	W2005
错误简述	密码错误
错误原因	软件执行某个需要密码的通讯命令时（比如上载、下载、监控），所输入的密码与PLC 中记录的密码不符。
解决措施	1．重新输入正确的密码。 2．如果忘记密码，可以格式化 PLC，密码会被清除；但是用户程序、数据块、系统块等内容都会丢失。请慎重！

错误代码	W2006
错误简述	计算机和 PLC 中文件不一致
错误原因	软件执行文件比较命令或试图进入监控状态时，发现计算机和 PLC 中的程序不一致。
解决措施	重新编译并下载当前工程中的所有程序。

错误代码	E0001
错误简述	输入了错误的指令
错误原因	输入了软件不支持的指令。 如输入“ LDD M0 ”，LDD 是不合法的指令，正确指令是 LD。
解决措施	在《编程参考手册》查找正确的指令。

错误代码	E0002
错误简述	指令输入的操作数数目不符
错误原因	输入的某指令的操作数个数与指令定义的个数不一致。 如输入“ LD M0 X1 ”，LD 指令只有一个操作数，而这里输入了两个操作数。
解决措施	在《编程参考手册》查找所需指令的操作数定义。

错误代码	E0003
错误简述	指令信息查询失败
错误原因	ControlStar 软件出现故障
解决措施	关闭然后重新启动 ControlStar 软件，如仍不能解决，请重新启动计算机

错误代码	E0004
错误简述	操作数类型错误
错误原因	输入的某指令操作数的数据类型与该指令中定义的数据类型不一致。 如输入“ ADD D0 1.50 D4 ”，ADD 指令操作数 2 的数据类型是整型，而这里输入了一个浮点型。
解决措施	在《编程参考手册》查找所需指令的操作数定义。

错误代码	E0005
错误简述	操作数读写属性错误
错误原因	输入的某指令操作数的读写属性与该指令中定义的读写不一致。 如输入“ MOV D0 SD0 ”，MOV 指令的第二个操作数必须是可写的，但是 SD0 是只读的。
解决措施	在《编程参考手册》查找所需指令的操作数定义。

错误代码	E0006
错误简述	操作数寄存器地址溢出
错误原因	输入的某寄存器地址超出了软件定义的范围。 如 MOV 10 D9000，D 寄存器的地址范围是 0～7999，9000 超出了此范围。
解决措施	在《编程参考手册》查找各种寄存器的地址范围。

错误代码	E0007
错误简述	寄存器类型查询失败
错误原因	ControlStar 软件出现故障
解决措施	关闭然后重新启动 ControlStar 软件，如仍不能解决，请重新启动计算机

错误代码	E0008
错误简述	十六进制数溢出
错误原因	输入的十六进制操作数超过了软件支持的十六进制数的最大范围 16#0～16#FFFFFFFF
解决措施	将操作数改至上述范围内

错误代码	E0009
错误简述	整数溢出
错误原因	输入的十进制操作数超过了软件支持的十进制整数的最大范围–2,147,483,648 ～ 4,294,967,295
解决措施	将操作数改至上述范围内

错误代码	E0010
错误简述	八进制数溢出
错误原因	输入的八进制操作数超过了软件支持的八进制数的最大范围 8#0～8#177777
解决措施	将操作数改至上述范围内

错误代码	E0011
错误简述	二进制数溢出
错误原因	输入的二进制操作数超过了软件支持的二进制数的最大范围 2#0～2#1111111111111111111111111111111
解决措施	将操作数改至上述范围内

错误代码	E0012
错误简述	操作数变址寄存器地址溢出
错误原因	输入的某指令操作数使用了变址寻址，但变址寄存器 Z 的地址超出了软件定义范围。 如 MOV 10 D0Z16，Z 寄存器的地址范围是 0～15，16 超出了此范围
解决措施	将变址寄存器改至 0～15 范围内

错误代码	E0013
错误简述	操作数不支持变址
错误原因	输入的某指令操作数使用了变址寻址，但在指令定义中该操作数不支持变址。 如“ LD M0Z1 ”，LD 指令操作数不支持变址
解决措施	

错误代码	E0014
错误简述	位串寻址方式下 n 值溢出
错误原因	输入的某指令操作数使用了位串寻址方式，但是 Kn 中的 n 值超过了 1～8 的范围。 如 K9M0。
解决措施	在《编程参考手册》查找所需指令的操作数定义以确定正确的 n 值

错误代码	E0015
错误简述	符号未定义
错误原因	输入的某指令操作数使用了软件不能识别的标识符。 如 ADD abc 100 D0，其中操作数 abc 既不是有效的寄存器地址，也没有在全局变量表或局部变量表中定义，所以不能识别
解决措施	在全局变量表或局部变量表中定义该符号

错误代码	E0016
错误简述	指令操作数错误
错误原因	对于 POWER 指令，当操作数 1 和操作数 2 都是常数，并且满足下列两种情况之一时出现该错误： 1．操作数 1 等于 0 且操作数 2 小于等于 0； 2．操作数 1 小于 0 且操作数 2 不是整数；
解决措施	

错误代码	E0017
错误简述	子程序调用参数数目不符
错误原因	使用 CALL 指令调用子程序时，输入的子程序参数数目与子程序定义的不相等
解决措施	打开被调用的子程序，根据其局部变量表中定义的参数修改 CALL 指令

错误代码	E0018
错误简述	子程序调用参数数据类型错误
错误原因	使用 CALL 指令调用子程序时，输入的子程序参数数据类型与被调子程序的相应参数定义的类型不一致
解决措施	打开被调用的子程序，根据其局部变量表中定义的参数修改 CALL 指令

错误代码	E0019
错误简述	子程序未定义
错误原因	使用 CALL 指令调用了工程中不存在的子程序
解决措施	检查是否误输入了子程序名或者增加以该名称命名的子程序

错误代码	E0023
错误简述	子程序循环递归调用
错误原因	程序中出现了子程序被循环调用的情况，如子程序 SBR_1 调用 SBR_2，SBR_2 又再调用 SBR_1
解决措施	修改程序逻辑避免这种情况

错误代码	E0024
错误简述	程序嵌套调用不能超过 6 层
错误原因	软件允许的程序嵌套调用的最大深度是 6 层，如果超过 6 层则出现该错误
解决措施	修改程序逻辑避免这种情况

错误代码	E0025
错误简述	指令操作数 3 使用计数器时必须和操作数 2 相同
错误原因	对于 DHSCR 指令，当操作数 3 使用计数器（C）地址时，必须和操作数 2 相同。 如 DHSCR 1000 C242 C243，操作数 3 使用了 C 寄存器，但是和操作数 2 不同
解决措施	

错误代码	E0026
错误简述	跳转标号不存在
错误原因	指令 CJ 所指示的跳转标号在本程序中不存在。 如子程序 SBR_1 中使用了指令 CJ 5，而 SBR_1 中没有使用 LBL 5 指令定义标号 5
解决措施	使用 LBL 指令定义该标号

错误代码	E0027
错误简述	跳转标号允许的范围是 0～127
错误原因	使用 LBL 指令定义跳转标号时，标号必须在 0～127 范围内，否则就会出现该错误
解决措施	

错误代码	E0028
错误简述	子程序数量超出范围
错误原因	工程中定义的子程序数量最多为 64 个，超过 64 个就会出现该错误
解决措施	删除或合并子程序，将子程序数量控制在 64 个以内

错误代码	E0030
错误简述	指令必须有操作数
错误原因	使用 CALL 指令时，至少要有一个操作数标识被调子程序，如果没有则出现该错误
解决措施	

错误代码	E0031
错误简述	寄存器地址只能使用 8 进制数表示
错误原因	X、Y 寄存器的地址只能使用 8 进制数表示
解决措施	将地址索引转换为 8 进制，比如 X8 改为 X10，X9 改为 X11

错误代码	E0032
错误简述	局部变量未定义
错误原因	指令中显式使用了局部变量（以“#”打头的字符串），但在该程序的局部变量表中并没有定义这个变量。 如 MOV #abc D0，而局部变量表中并没有定义 abc 变量
解决措施	

错误代码	E0035
错误简述	ED、ED 指令的使用超过了最大数量限制
错误原因	在一个工程内，EU、ED 指令累计最多可以使用 1000 次
解决措施	

错误代码	E0036
错误简述	RAMP、HACKLE、TRIANGLE 指令的使用超过了最大数量限制
错误原因	在一个工程内，RAMP、HACKLE、TRIANGLE 指令累计最多可以使用 100 次
解决措施	

错误代码	E0037
错误简述	数据块中 D 寄存器地址重复使用
错误原因	在数据块定义中，出现重复使用某个地址的情况。 如： D0 100 2000 3000 D2 15 其中 D2 地址被重复使用。
解决措施	

错误代码	E0039
错误简述	文件打开失败
错误原因	试图编译某个程序时，发现不能打开该程序对应的磁盘文件。文件可能被误删除或者被损坏
解决措施	尝试从工程树上打开此程序，如果能够正常打开，请重新将此程序保存一次；如果不能打开，请删掉此程序然后重新添加

错误代码	E0040
错误简述	操作数的取值范围为 0～7
错误原因	某些指令要求操作数的取值必须在 0～7 之间，比如 MC 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0041
错误简述	操作数（高速计数器）的取值范围为 C236～C255
错误原因	某些高速 IO 指令要求高速计数器地址必须在 C236～C255 之间，比如 DHSCI 指令的第二个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0042
错误简述	操作数（常数）的取值范围为 20～25
错误原因	某些指令要求操作数（常数）的取值必须在 20～25 之间，比如 DHSCI 指令的第三个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0044
错误简述	操作数（计数器）的取值范围为 C0～C199
错误原因	某些指令要求计数器地址必须在 C0～C199 之间，比如 CTU、CTR 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0045
错误简述	操作数 2（常数）的取值范围为 1～128
错误原因	某些指令要求操作数 2（常数）的取值必须在 1～128 之间，比如 DHSP 指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0046
错误简述	操作数的取值范围为 1～100000
错误原因	某些指令要求操作数的取值必须在 1～100000 之间，比如 PLSY 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0047
错误简述	操作数必须大于 0
错误原因	某些指令要求操作数必须大于 0，比如 PLSY 指令的第二个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0048
错误简述	操作数元件地址索引只能为 0 或 1
错误原因	某些指令要求操作数寄存器地址索引必须为 0 或 1，比如 PLSY 指令的第三个操作数，只能是 Y0 或者 Y1。具体请参考《编程参考手册》
解决措施	

错误代码	E0049
错误简述	操作数的取值范围为 10～20000
错误原因	某些指令要求操作数（常数）的取值必须在 10-20000 之间，比如 PLSR 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0050
错误简述	操作数 2 的值必须大于 110
错误原因	某些指令要求操作数 2（常数）必须大于 110，比如 PLSR 指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0052
错误简述	操作数 1（计数器地址）的正常范围为 C200～C235
错误原因	某些指令要求操作数 1（计数器地址）的取值必须在 C200～C235 之间，比如 DCNT 指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0054
错误简述	操作数的值必须大于 1
错误原因	某些指令要求操作数的值必须大于 1，比如 PWM 指令的第二个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0055
错误简述	操作数的取值范围为 0～15
错误原因	某些指令要求操作数的取值必须在 0～15 之间，比如指令 SEG 的第一个操作数及 BOUT 指令的第二个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0056
错误简述	操作数的取值范围为 X0～X5
错误原因	某些指令要求操作数的取值必须在 X0～X5 之间，比如 SPD 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0060
错误简述	操作数的取值范围为 0～60
错误原因	某些指令要求操作数的取值必须在 0～60 之间，比如指令 REFF 的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0061
错误简述	操作数必须在 0～255 范围内并且是 8 的倍数
错误原因	某些指令要求操作数的取值必须在 0～255 范围内并且是 8 的倍数。比如 REF 指令的第二个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0062
错误简述	操作数的取值范围为 1～32767
错误原因	某些指令要求操作数的取值必须在 1～32767 之间，比如 FROM 指令的第四个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0063
错误简述	操作数（常数）只能为 0 或 1
错误原因	某些指令要求操作数（常数）的值必须是 0 或 1，比如 RCV、XMT 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0064
错误简述	操作数不支持此寄存器
错误原因	某些指令的操作数不支持某些寄存器。如 MOV 指令第二个操作数不支持 KnX 类型。具体请参考《编程参考手册》
解决措施	

错误代码	E0065
错误简述	反编译 UCODE 文件失败
错误原因	上载程序时，软件需要对上载的 UCODE 文件进行反编译，如果反编译失败则会报此错误。原因可能是通讯过程中产生错误或者软件版本不配套
解决措施	先重试，若仍然不能成功，请检查软件版本和 PLC 版本是否配套，若配套没有问题，说明可能 PLC 中的程序已经被破坏，不能再被反编译

错误代码	E0066
错误简述	主程序不能为空
错误原因	当工程中的主程序为空时出现这个错误，因为这样相当于整个应用程序都为空
解决措施	

错误代码	E0067
错误简述	跳转标号重复定义
错误原因	在同一个程序中，每个跳转标号只能定义一次，如果被定义了多次（相当于多个 LBL 指令的操作数相同）就会出现这个错误。 如程序中 … LBL 10 … CJ 10 … LBL 10 *//标号 10 被重复定义* …
解决措施	

错误代码	E0068
错误简述	操作数值的取值范围为 0～255
错误原因	某些指令要求操作数的取值必须在 0～255 之间，比如 SUM 指令的第二个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0069
错误简述	操作数的取值不能小于 0
错误原因	某些指令要求操作数的的取值不能小于 0，比如 PWM 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0070
错误简述	操作数使用位串寻址时 n 值必须为 4
错误原因	某些指令要求当操作数使用位串寻址时，Kn 中的 n 值必须为 4，比如 ROR 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0071
错误简述	操作数 1、2 的 Kn 值必须相同
错误原因	某些指令要求当操作数使用位串寻址时，操作数 1 和操作数 2 的 Kn 中的 n 值必须相同。比如 XCH、DXCH 等指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0072
错误简述	两个操作数必须同时使用位串寻址或者都不使用
错误原因	某些指令要求，如果两个操作数中的一个使用了位串寻址方式，则另外一个也必须使用，否则将提示该错误。比如 XCH、DXCH 等指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0074
错误简述	双字地址超过 D 寄存器地址范围
错误原因	当使用 D7999 来存放一个双字的数值时会提示该该错误。 **示例：** 如果数据块中有如下定义： D7999 1.25 由于 1.25（浮点数）是双字的，D7999 已经达到了 D 寄存器的最大边界，存放不下这个数，导致错误。
解决措施	

错误代码	E0075
错误简述	HOUR 指令的使用次数不能超过 256 次
错误原因	在同一个工程中，HOUR 指令的使用次数不能超过 256 次，否则提示该错误
解决措施	

错误代码	E0076
错误简述	操作数地址索引必须是 8 的倍数
错误原因	某些指令要求操作数的寄存器地址索引必须是 8 的倍数，比如指令指令 REF 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0077
错误简述	使用了冲突的 HCNT、SPD 指令或者硬件中断
错误原因	高速 I/O 指令 HCNT、SPD 和中断的使用是有限制的，如果违反限制规则就会出现这个错误。 比如 X000 在不同的高速 I/O 指令中不可重复使用，以下情况为错误： 1. 使用 HCNT 指令同时驱动了（C236 C242 C244 C246 C247 C249 C251 C252 C254）中的任意两个。 2. 使用 HCNT 指令驱动了（C236 C242 C244 C246 C247 C249 C251 C252 C254）中任一个，同时使用了 SPD X0 指令。 3. 使用 HCNT 指令驱动了（C236 C242 C244 C246 C247 C249 C251 C252 C254）中任一个，或使用了 SPD X0 指令，同时又编写了中断编号为 0 或 10 的中断子程序。 具体请参考《编程参考手册》。
解决措施	

错误代码	E0078
错误简述	操作数的取值范围为 0～127
错误原因	某些指令要求操作数的取值必须在 0～127 之间，比如 CJ 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0079
错误简述	元件不能作 CALL 指令的输出参数
错误原因	在 CALL 指令中，软元件 SM、SD、KnSM 不能使用在子程序的输出参数或者输入输出参数中，否则提示该错误
解决措施	

错误代码	E0080
错误简述	元件读写属性与子程序参数定义不符
错误原因	在 CALL 指令中，传递给子程序的输出型和输入输出型参数的元件地址必须是可写的，否则将提示该错误
解决措施	

错误代码	E0081
错误简述	数据块定义不能超过 5000 个元件
错误原因	数据块最多可以定义 5000 个 D 元件，否则将提示该错误
解决措施	

错误代码	E0082
错误简述	高速计数指令的使用最多不能超过 256 个
错误原因	某些高速计数指令的使用最多不能超过 256 个，比如 DHSCI、DHSCS 、DHSCR、DHSZ、DHST、DHSP 等指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0084
错误简述	操作数使用位串寻址时 n 值必须为 8
错误原因	某些指令要求当操作数使用位串寻址时，Kn 中的 n 值必须为 8，比如 DROR 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0085
错误简述	操作数 3 不能小于操作数 4
错误原因	某些指令要求操作数 3 不能小于操作数 4，比如 WSFR、WSFL 等指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0086
错误简述	操作数 1 不能小于操作数 2
错误原因	某些指令要求操作数 1 不能小于操作数 2，比如 PWM 指令。具体请参考《编程参考手册》
解决措施	

错误代码	E0087
错误简述	MODBUS 指令操作数 1 只能为 1
错误原因	对于 MODBUS 指令，第一个操作数必须为 1，否则提示该错误
解决措施	

错误代码	E0088
错误简述	操作数的取值范围为-32767～32767
错误原因	某些指令要求操作数的取值必须在-32767～32767 之间，比如 VABS 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0089
错误简述	操作数值的取值范围为-2147483647～2147483647
错误原因	某些指令要求操作数的取值必须在-2147483647～2147483647 之间，比如 DVABS 指令的第一个操作数。具体请参考《编程参考手册》
解决措施	

错误代码	E0090
错误简述	反编译数据块文件失败
错误原因	上载数据块时，软件需要对上载的数据块文件进行反编译，如果反编译失败则会报此错误。原因可能是通讯过程中产生错误或者软件版本不配套
解决措施	先重试，若仍然不能成功，请检查软件版本和 PLC 版本是否配套，若配套没有问题，说明可能 PLC 中的数据块已经被破坏，不能再被反编译

错误代码	E0113
错误简述	程序编译后的步数超过 PLC 的容量限制
错误原因	用户程序编译后生成的目标文件太大，其步数超过了 PLC 的容量限制
解决措施	简化程序以节省空间

错误代码	E0114
错误简述	全局变量表不能超过 24K 字节
错误原因	用户程序中定义的全局变量表太大，超过了 PLC 的容量限制（24K 字节），不能下载到 PLC 中
解决措施	删除部分全局变量以节省空间

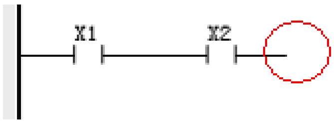
错误代码	E0115
错误简述	POU 表不能超过 8K 字节
错误原因	用户程序中定义的 POU（程序单元）信息太大，超过了 PLC 的的容量限制（8K 字节），不能下载到 PLC 中。POU 信息包括子程序、中断子程序以及其参数的信息
解决措施	删除/合并 POU 或者减少程序参数以节省空间

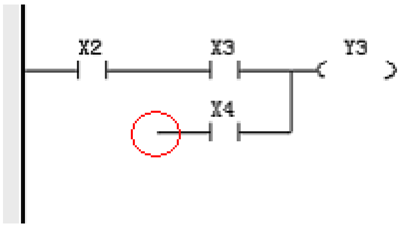
错误代码	E0116
错误简述	数据块源程序不能超过 24K 字节
错误原因	用户程序中定义的数据块源程序太大，超过了 PLC 的容量限制（24K 字节），不能下载到 PLC 中
解决措施	修改/减少数据块源程序以节省空间

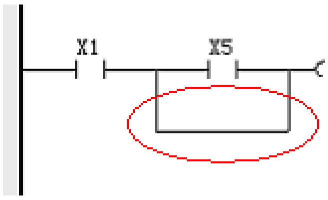
错误代码	E1000
错误简述	FOR、NEXT 指令不匹配，或 FOR～NEXT 指令中间嵌套了不匹配的 MC、MCR 指令
错误原因	1、FOR 指令后面缺少相应的 NEXT 指令，请检查程序中每一个 FOR 指令后面是否有对应的 NEXT 指令，必须保证 FOR 和 NEXT 指令成对使用。 2、NEXT 指令前缺少相应的 FOR 指令，请检查每个 NEXT 指令前是否有 FOR 指令。 3、FOR～NEXT 指令块与 MC～MCR 指令块交叉。 如下面的语句所示： FOR …… MC　　*// MC 指令位于 FOR～NEXT 指令之间，错误* …… NEXT …… MCR
解决措施	

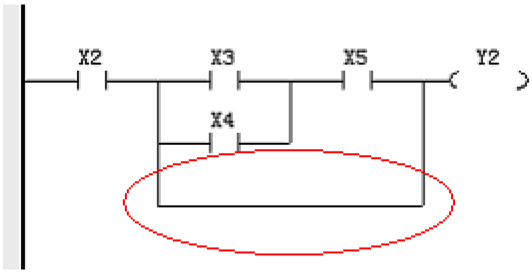
错误代码	E1001
错误简述	MC、MCR 指令不匹配，或 MC、MCR 指令中间嵌套了不匹配的 FOR、NEXT 指令
错误原因	1、MC 指令后面缺少相应的 MCR 指令，请检查程序中每一个 MC 指令后面是否有对应的 MCR 指令，必须保证 MC 和 MCR 指令是成对使用。 2、MCR 指令前缺少相应的 MC 指令，请检查每个 MCR 指令前是否有 MC 指令。 3、MC～MCR 指令块与 FOR～NEXT 指令块交叉使用。 如下面的语句所示： MC …… FOR　　*// FOR 指令位于 MC～MCR 指令之间，错误* …… MCR …… NEXT
解决措施	

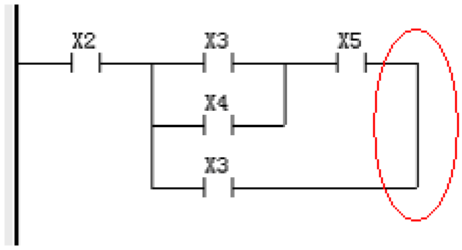
错误代码	E1002
错误简述	元件不能直接连接左母线
错误原因	1．除非在 SFC 的内置梯形图中，否则输出元件以及应用指令元件不能直接连接左母线。 2．反向、上升沿、下降沿元件不能直接连接左母线。
解决措施	

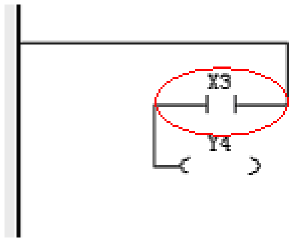
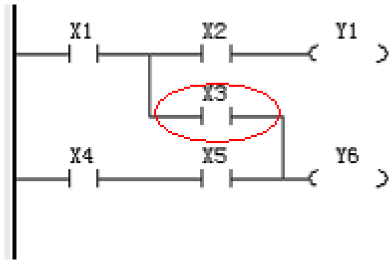
<table>
<tr><td>错误代码</td><td>E1003</td></tr>
<tr><td>错误简述</td><td>开路</td></tr>
<tr><td>错误原因</td><td>某个触点型元件的前端或后端没有连接任何其他元件，
如下两图分别在 X2 处和 X4 处开路：


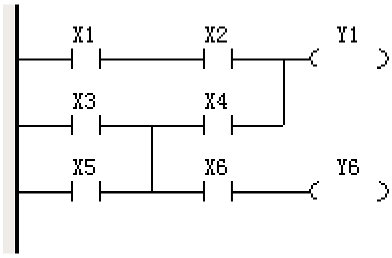

</td></tr>
<tr><td>解决措施</td><td></td></tr>
</table>

<table>
<tr><td>错误代码</td><td>E1004</td></tr>
<tr><td>错误简述</td><td>短路</td></tr>
<tr><td>错误原因</td><td>某一个或者多个元件，被连接线直接将前端和后端连接在一起。
如下两图，分别是 X5 和 X3、X4、X5 组成的元件块被短路。



</td></tr>
<tr><td>解决措施</td><td>删除导致短路的连接线或者在连接线中插入至少一个元件</td></tr>
</table>

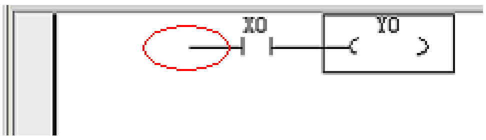
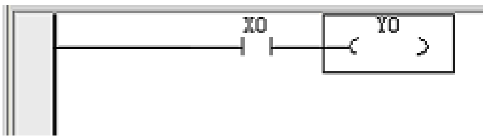
错误代码	E1005
错误简述	网络缺少输出
错误原因	一个梯形图网络中没有短路或开路，并且没有输出元件或应用指令。 如下图所示： 
解决措施	在网络右侧加上输出元件或应用指令

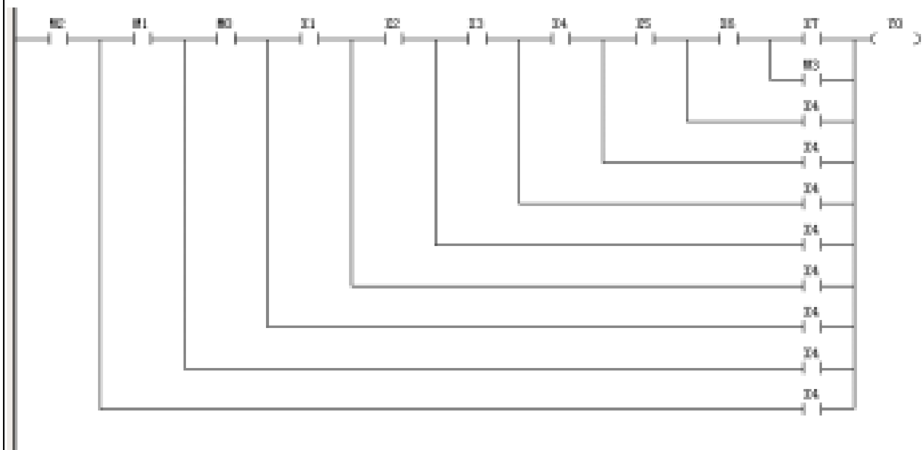
错误代码	E1006
错误简述	存在回流
错误原因	在梯形图中，电流流过元件时必须从左边流入右边流出，否则便认为是回流。 如下两图中均在 X3 元件处出现回流：  
解决措施	

错误代码	E1007
错误简述	网络太复杂
错误原因	对于一个正确的网路，将其进行有限次递归或反复运算，最终都能够化简为一个单输入对应单输出的网络，如果不能这样化简并且没有发现其他已定义的错误，则认为网络过于复杂。 下面的梯形图程序将提示该错误： 
解决措施	

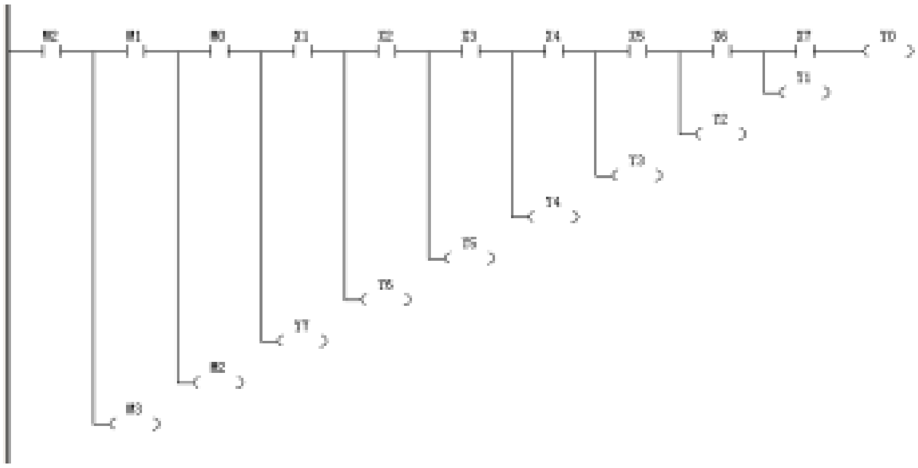
错误代码	E1008
错误简述	每个顺序功能图网络中只能有一个初始步进符
错误原因	对于顺序功能图程序，每一个网络中只能有一个初始步进符，如果一个顺序功能图网络中出现多个初始步进符（使用了多个初始步进元件），则认为存在该错误。 对于梯形图和指令列表程序，如果转换后的顺序功能图中存在上述错误，也认为存在该错误。
解决措施	

错误代码	E1009
错误简述	转移符前缺少步进符
错误原因	在顺序功能图程序中，转移符元件前面连接的元件必须是步进符，否则即为错误的连接关系。 在指令列表程序中，如果转换后的顺序功能图中存在上述错误，也认为存在该错误
解决措施	

错误代码	E1010
错误简述	网络没有连接左母线
错误原因	梯形图中的每个网络都应该至少有一个元件直接连接到左母线，否则将提示该错误。 如下图所示： 
解决措施	将整个网络向左移动直到连接上左母线，或使用横线将网络中的某个元件连接到左母线。 如下图所示： 

错误代码	E1012
错误简述	输入栈中元素个数不能超过 8 个
错误原因	下图将会出现输入栈中元素个数大于 8 的错误： 
解决措施	简化网络结构

错误代码	E1014
错误简述	MPS 指令前缺少 LD 指令
错误原因	在使用 MPS 指令时必须保证当前有能流点，否则将提示该错误。 例如： MPS *// MPS 指令前缺少 LD 指令* MPP OUT Y1
解决措施	在 MPS 指令前面增加 LD 指令

错误代码	E1015
错误简述	输出栈中元素个数不能超过 8 个
错误原因	下图将会出现输出栈中元素个数大于 8 的错误： 
解决措施	简化网络结构

错误代码	E1016
错误简述	MRD 指令前缺少 MPS 指令
错误原因	在使用 MRD 指令前需要使用 MPS 指令将能流点压栈，否则将提示该错误。 例如： MRD *// MRD 指令前缺少 MPS 指令* OUT Y1
解决措施	在 MRD 指令前面增加 MPS 指令

错误代码	E1017
错误简述	MPP 指令前缺少 MPS 指令
错误原因	在使用 MPP 前需要使用 MPS 将能流点压栈，否则将提示该错误。 例如： MPP *// MPP 指令前缺少 MPS 指令* OUT Y1
解决措施	在 MPP 指令前面增加 MPS 指令

错误代码	E1018
错误简述	块操作指令前缺少输入指令
错误原因	在使用 ANB、ORB 等块操作指令前，必须保证输入栈中至少有两个值，否则将提示该错误。 例如： LD X1 ORB *// ORB 指令前的输入栈只有一个值*
解决措施	在输入栈中增加指令，例如增加一个 LD X0；或者删除 ORB 指令

错误代码	E1020
错误简述	指令前缺少能流点
错误原因	AND、OR、ANB、ORB、INV、EU、ED 等指令使用时前面必须有能流点，否则将提示该错误
解决措施	在前面使用 LD 指令载入能流点

错误代码	E1021
错误简述	输出能流点后不能跟 OR、ORB
错误原因	1．如果 ORB 前面的两个能流点至少有一个是输出能流点，则提示该错误。 2．如果 OR 的前面的能流点是输出能流点，也提示该错误。 例如： LD X0 MPS OUT Y0 MPP ORB *// 此处使用非法，如果使用 OR 也是非法*
解决措施	

错误代码	E1022
错误简述	跳转、转移和重置符后面不能跟内置梯形图
错误原因	在顺序功能图中，跳转和重置符都不能有内置梯形图，转移的条件也不应该有输出元件或应用指令，如果某个指令表程序对应的顺序功能图程序违反了以上规则，则提示该错误
解决措施	将复杂的指令放到步进中

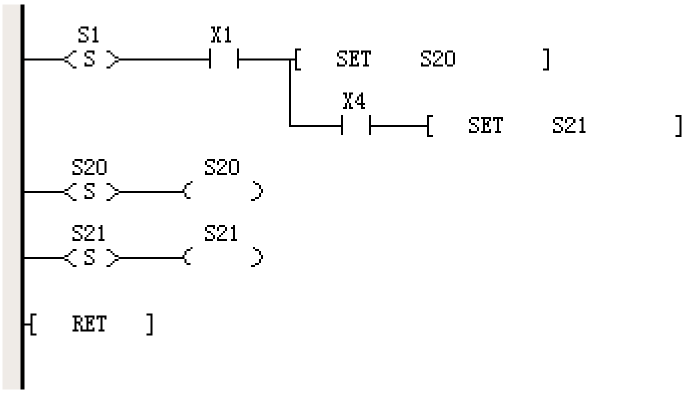
错误代码	E1023
错误简述	MC 指令使用了重复的标号
错误原因	MC 与 MCR 嵌套时，不能使用相同的标号，包括不同进制表示但是值相同的情况。 如下程序将认为是 MC 与 MCR 使用了相同的标号。 …… LD X1 MC 1 …… LD X2 MC 2#1 *//与上面的 MC 1 重复* …… MCR 2#1 …… MCR 1 ……
解决措施	

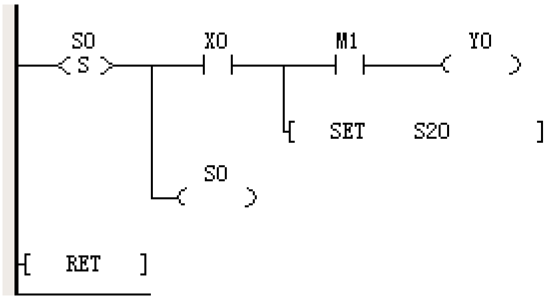
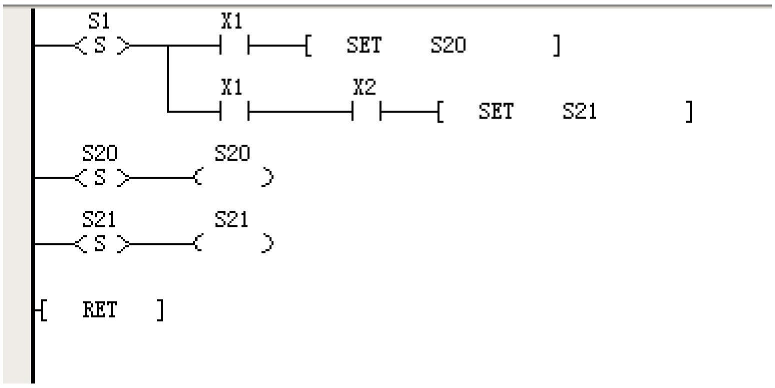
错误代码	E1024
错误简述	FOR 循环嵌套超过 8 层
错误原因	在梯形图和指令列表程序中，FOR..NEXT 循环可以嵌套使用，但嵌套层数不能超过 8 层，否则将出现该错误
解决措施	

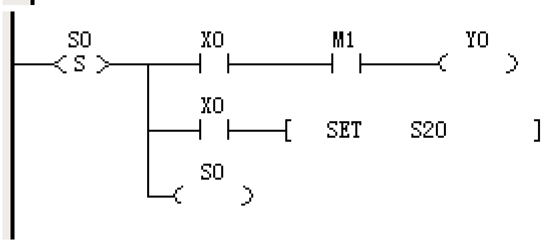
错误代码	E1025
错误简述	OUT 或者应用指令前缺少 LD 指令
错误原因	在梯形图中（非内置梯形图），OUT 指令和应用指令不能连接左母线，否则将提示该错误。 如果指令列表程序转换后的梯形图程序后出现以上错误，也将提示该错误。
解决措施	

错误代码	E1026
错误简述	MCR 指令没有对应的 MC 指令
错误原因	MCR 指令和 MC 指令必须一一对应，如果缺少 MC 指令，将提示该错误
解决措施	

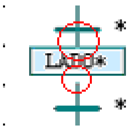
错误代码	E1027
错误简述	MCR 指令和 MC 指令标号不匹配
错误原因	使用 MC 与 MCR 时，后面的标号必须相同，否则将提示该错误
解决措施	

<table>
<tr><td>错误代码</td><td>E1031</td></tr>
<tr><td>错误简述</td><td>转移的条件太复杂，无法转换</td></tr>
<tr><td>错误原因</td><td>转移的内置梯形图，经过有限次的运算后最终未能化简为一个单输入对应 TRANS 的网络。
对于如下的两个梯形图图形，转换成顺序功能图网络时，不能将 S20 和 S21 作为并行分支（转移的条件不同），也不能将其作为选择分支。



</td></tr>
<tr><td>解决措施</td><td>将上图两图形修改为下面的两个图形：



</td></tr>
</table>

错误代码	E1032
错误简述	转移的条件不能为空
错误原因	转移的内置梯形图中应该至少含有一个 TRANS 指令（无条件转移），在编辑顺序功能图时添加一个转移元件将自动增加一个 TRANS 指令，如果该 TRANS 指令被删除，将提示该错误。 解决措施： 增加转移条件及 TRANS 指令。
解决措施	

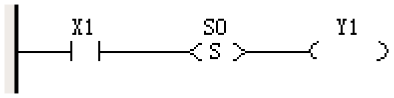
错误代码	E1033
错误简述	普通梯形图块不能和其他顺序功能图元件连接
错误原因	普通梯形图块元件不可以与其他顺序功能图元件连接。 如图： 
解决措施	

错误代码	E1038
错误简述	并行汇合支路个数不能超过 16 个
错误原因	顺序功能图的并行汇合的分支个数不能超过 16 个，否则将提示该错误。 如果梯形图或指令列表转换后的顺序功能图中，并行汇合的分支个数超过 16 个，也将提示该错误。
解决措施	

错误代码	E1048
错误简述	INV、EU 和 ED 元件不能与其他元件并联
错误原因	INV（反向）、EU（上升沿）、ED（下降沿）三个元件不能和其他元件并联，否则将提示该错误
解决措施	

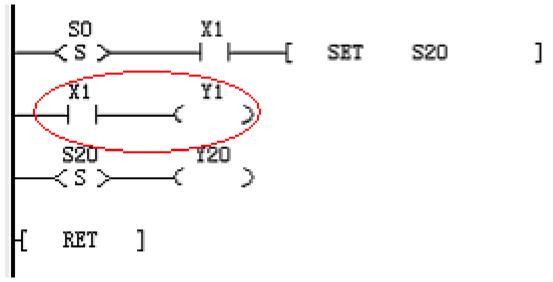
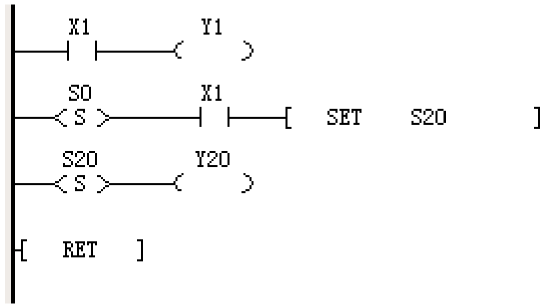
错误代码	E1049
错误简述	网络缺少输出指令
错误原因	每一个网络都应该包含至少一个输出节点，如果指令表程序转换为梯形图或者顺序功能图后出现没有输出指令的网络，将提示该错误
解决措施	

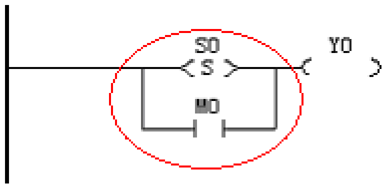
错误代码	E1050
错误简述	跳转符和重置符不能包含内置梯形图
错误原因	在顺序功能图程序中，跳转符和重置符是不能包含内置梯形图的。如果指令列表程序转换为顺序功能图后出现跳转符和重置符包含内置梯形图的情况，将提示该错误
解决措施	

错误代码	E1055
错误简述	元件位置不对
错误原因	梯形图网络中的 STL 元件的左边必须连接左母线或者其他 STL 元件，否则即为该错误。 如下图，将提示 S0 元件位置不对： 
解决措施	

错误代码	E1056
错误简述	RET 指令前缺少顺序功能图网络
错误原因	在梯形图和指令列表程序中，每一个顺序功能图网络都应该以 RET 指令结束，同样每一个 RET 指令前都应该有一个顺序功能图网络，如果 RET 前面没有顺序功能图网络，将提示该错误。
解决措施	删除多余的 RET 指令

错误代码	E1057
错误简述	元件必须直连左母线
错误原因	NEXT、MCR、LBL、RET 等指令必须直接连接左母线或者 STL 元件，否则将提示该错误
解决措施	

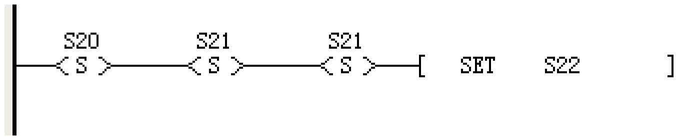
错误代码	E1061
错误简述	普通梯形图块不能出现在顺序功能图网络的中间
错误原因	下图中，红圈内的元件转换后将生成一个插在 S0 和 S20 之间的普通梯形图块： 
解决措施	改为下面的程序： 

错误代码	E1065
错误简述	STL 元件不能和其他元件并联使用
错误原因	STL（步进点）元件不能和其他元件并联，否则将出现该错误。 如下图： 
解决措施	

错误代码	E1068
错误简述	顺序功能图转移指令不能出现在步进符的内置梯形图中
错误原因	顺序功能图的转移指令不能出现在步进符的内置梯形图中。梯形图和指令列表程序转换为顺序功能图后，如果出现这个情况将提示该错误。 例如下面的代码是错误的： STL S0 LD M0 OUT Y0 SET S21 *// 此时能流点在 M0 后，转移到 S21 接入了 S0 的内置梯形图中* RET
解决措施	STL S0 LD M0 OUT Y0 LD SM0 *// 加入一个能流节点，使转移条件形成一个完整的梯形图网络* SET S21 RET

错误代码	E1071
错误简述	转换后梯形图网络将超过*列的最大限制
错误原因	普通梯形图最多可容纳为 32 列元件，内置梯形图最多可容纳为 16 列元件。如果转换后的梯形图或者顺序功能图的内置梯形图超过了这个限制，将提示该错误
解决措施	简化程序逻辑或使用中间变量以减少列数

错误代码	E1073
错误简述	网络超过*列的最大限制
错误原因	在将顺序功能图转换为其他语言时，如果发现某个内置梯形图中的元件超过了 16 列，就会提示该错误
解决措施	简化程序逻辑或使用中间变量以减少列数

错误代码	E1074
错误简述	并行汇合的分支不允许使用相同的步进符
错误原因	对于并行汇合，其分支不能使用相同的步进符号。 如下的梯形图程序和指令列表程序都重复使用了 S21： 梯形图：  指令列表： STL S20 STL S21 STL S21 SET S22
解决措施	

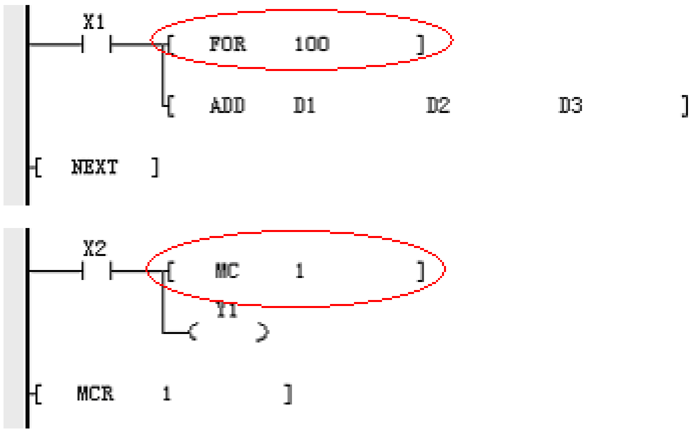
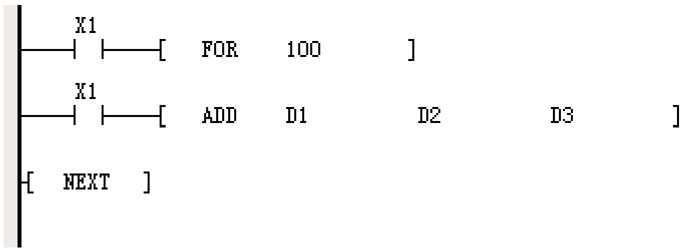
错误代码	E1075
错误简述	元件不能出现在顺序功能图的步进内
错误原因	SET S**，OUT S**，RST S**，STL ***等指令不能出现在步进符的内置梯形图中，否则将提示该错误。 如果梯形图或者指令列表程序转换后的顺序功能图程序出现以上情况，也提示该错误
解决措施	

错误代码	E1076
错误简述	元件不能出现在顺序功能图的转移内
错误原因	SET S**，OUT ***，RST S**，STL **，应用指令（TRANS 除外）不能出现在顺序功能图的转移中，否则将提示该错误。 如果梯形图或指令列表程序转换后的顺序功能图程序出现以上情况，也提示该错误。
解决措施	

错误代码	E1077
错误简述	元件不能出现在顺序功能图的普通梯形图块内
错误原因	TRANS，STL** 不能出现在顺序功能图网络的普通梯形图块中，否则将提示该错误。 如果梯形图或指令列表程序转换后的顺序功能图程序出现以上情况，也提示该错误。
解决措施	

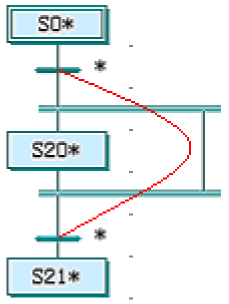
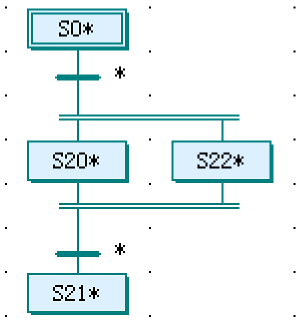
错误代码	E1078
错误简述	顺序功能图网络缺少 RET 指令
错误原因	在梯形图和指令列表程序中，如果该程序表示了一个顺序功能图，则每个网络之间都应该用 RET 指令隔离开，否则提示该错误。
解决措施	

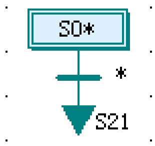
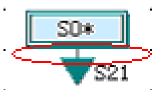
错误代码	E1079
错误简述	转换后梯形图网络将超过*行的最大限制
错误原因	梯形图程序的最大行数为 4000 行，从其他程序向梯形图程序转换过程中，如果生成的梯形图程序超过 4000 行将提示该错误
解决措施	简化程序逻辑以减少行数

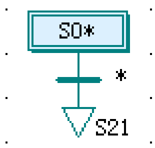
错误代码	E1083
错误简述	FOR、MC 指令所在的梯形图网络不能存在其他的输出元件或应用指令
错误原因	FOR 与 MC 指令作为某个网络的输出后，那么这个网络就不能再使用其他的输出元件或应用指令了。 如下梯形图程序所示： 
解决措施	可以按照下面的程序修改： 

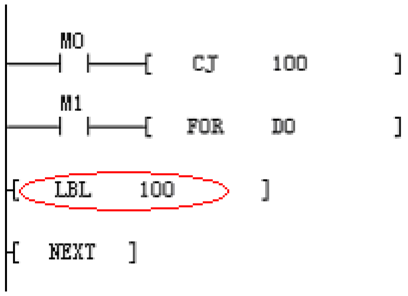
错误代码	E1084
错误简述	步进符不能重复使用
错误原因	在顺序功能图程序中，步进符是不能重复使用的。如果指令列表程序转换后的顺序功能图中出现一个步进符被多次使用的情况，将提示该错误
解决措施	

错误代码	E1085
错误简述	普通步进符不能作为顺序功能图网络的起始使用
错误原因	顺序功能图网络必须以初始步进符为起始，否则提示该错误。 对于指令列表或梯形图程序，如果转换后的顺序功能图网络不是以初始步进符开头，也提示该错误。
解决措施	将提示错误的元件改为初始步进符

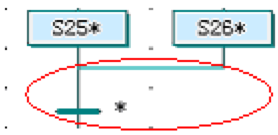
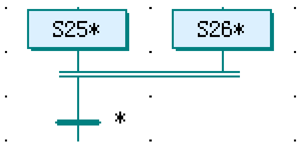
错误代码	E1086
错误简述	转移符前必须接步进符
错误原因	示例： 错误的图形：（S21 前的转移符上方连接了 S20 和 S0 下方的转移符）  正确的图形： 
解决措施	

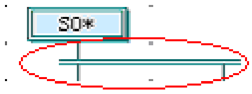
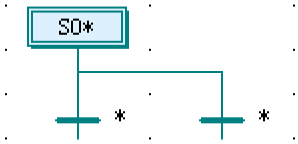
错误代码	E1087
错误简述	跳转符前必须接转移符
错误原因	跳转符和步进符间必须有转移符连接。 示例： 正确的图形：  错误的图形： 
解决措施	

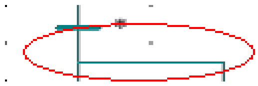
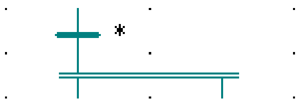
错误代码	E1088
错误简述	重置符前必须接转移符
错误原因	步进符和重置符间必须有转移符连接。 示例： 正确的图形：  错误的图形： 
解决措施	

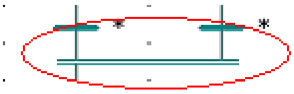
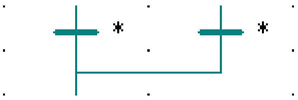
错误代码	E1089
错误简述	CJ 不能跳入到 FOR 与 NEXT 循环内
错误原因	FOR…NEXT 循环外的 CJ 指令跳入了 FOR…NEXT 循环内的 LBL 标号上。 示例： 错误的图形（标号 100 被定义在 FOR 循环内部，而指令 CJ 跳转到标号 100）： 
解决措施	

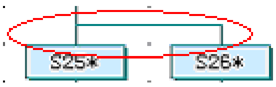
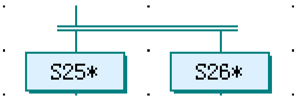
错误代码	E1090
错误简述	CJ 不能跳入到 MC 与 MCR 之间
错误原因	MC…MCR 外的 CJ 指令跳入了 MC…MCR 内的 LBL 标号上
解决措施	

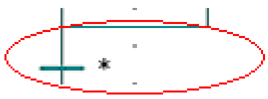
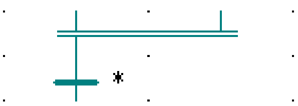
错误代码	E1092
错误简述	步进符后不能接选择汇合
错误原因	在顺序功能图中，步进符元件下面不能直接与选择汇合连接
解决措施	将选择汇合改为并行汇合。 示例： 错误的图形：  正确的图形： 

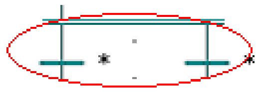
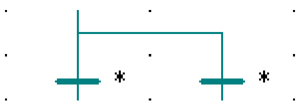
错误代码	E1093
错误简述	步进符后不能接并行分支
错误原因	在顺序功能图中，步进符元件下面不能直接与并行分支连接
解决措施	将并行分支改为选择分支。 示例： 错误的图形：  正确的图形： 

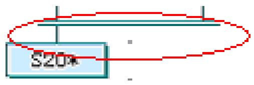
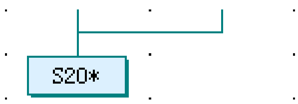
错误代码	E1094
错误简述	转移符后不能接选择分支
错误原因	在顺序功能图中，转移符下面不能直接与选择分支连接
解决措施	将选择分支改为并行分支。 示例： 错误的图形：  正确的图形： 

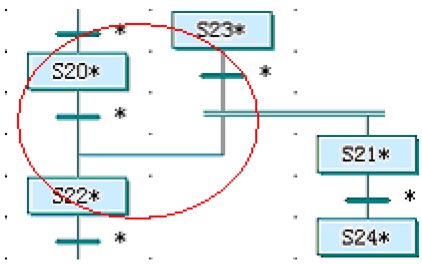
错误代码	E1095
错误简述	转移符后不能接并行汇合
错误原因	在顺序功能图中，转移符下面不能直接与并行汇合连接
解决措施	将并行汇合改为选择汇合。 示例： 错误的图形：  正确的图形： 

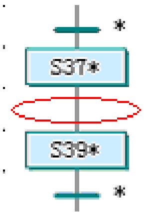
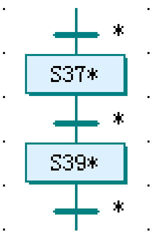
错误代码	E1096
错误简述	步进符前不能接选择分支
错误原因	在顺序功能图中，步进符上面不能直接与选择分支连接
解决措施	将选择分支改为并行分支。 示例： 错误的图形：  正确的图形： 

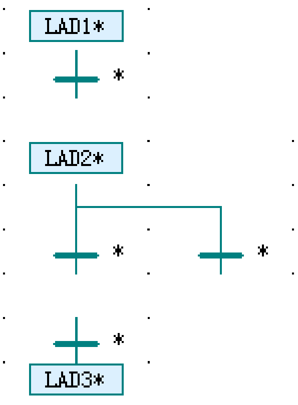
错误代码	E1097
错误简述	转移符前不能接选择汇合
错误原因	在顺序功能图中，转移符上面不能直接与选择汇合连接
解决措施	将选择汇合改为并行汇合。 示例： 错误的图形：  正确的图形： 

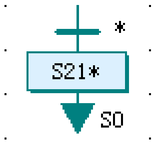
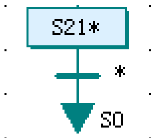
错误代码	E1098
错误简述	转移符前不能接并行分支
错误原因	在顺序功能图中，转移符上面不能直接与并行分支连接
解决措施	将并行分支改为选择分支。 示例： 错误的图形：  正确的图形： 

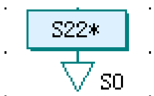
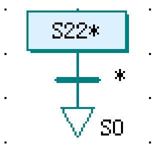
错误代码	E1099
错误简述	步进符前不能接并行汇合
错误原因	在顺序功能图中，步进符上面不能直接与并行汇合连接
解决措施	将并行汇合改为选择汇合。 示例： 错误的图形：  正确的图形： 

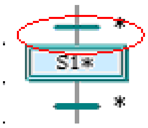
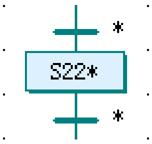
错误代码	E1100
错误简述	SFC 连接错误
错误原因	在顺序功能图程序中，汇合与分支不能连接到同一个元件中。 错误的图形如下（步进符 S22 的与 S20 的关系是汇合，但是与 S23 的关系是分支）： 
解决措施	

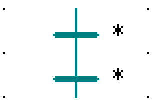
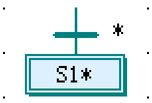
错误代码	E1101
错误简述	步进符不能互连
错误原因	根据顺序功能图的语法，两个步进符元件不能直接连接
解决措施	在两个需要连接的步进符元件之间加入一个转移符元件。 示例： 错误的图形：  正确的图形： 

错误代码	E1102
错误简述	梯形图块不能与其他元件连接
错误原因及解决措施	普通梯形图块元件在顺序功能图程序中是不能与其他元件连接的， 示例： 以下的图都是错误的： 

错误代码	E1103
错误简述	步进符不能连接跳转符
错误原因	在顺序功能图中，步进符元件不能直接与跳转符元件连接
解决措施	在步进符与跳转符之间增加一个转移符。 示例： 错误的图形：  正确的图形： 

错误代码	E1104
错误简述	重置符前必须接转移符
错误原因	在顺序功能图中，重置符元件前面只能与转移符直接连接
解决措施	在步进符和重置符之间加入一个转移符。 示例： 错误的图形：  正确的图形 

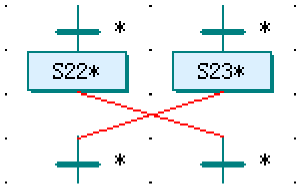
错误代码	E1106
错误简述	初始步进符只能做为 SFC 网络的起始
错误原因	在顺序功能图中，初始步进符只能作为顺序功能图网络的起始使用
解决措施	将错误的初始步进符改为普通步进符。 示例： 错误的图形：  正确的图形： 

错误代码	E1107
错误简述	转移符后必须接步进符、跳转符、重置符
错误原因及解决措施	在顺序功能图中，转移符不能互相连接，且转移符的后面只能是普通步进符、跳转符或重置符。 示例： 下图的图都是错误的：  

错误代码	E1108
错误简述	普通梯形图块不能使用其他软元件地址或变量名称
错误原因	在顺序功能图中，普通梯形图块不能使用有效的元件地址或变量名称等作标识符，比如 S0、X0 等，否则会提示该错误
解决措施	

错误代码	E1109
错误简述	超出元件可用地址范围
错误原因	在顺序功能图中，使用了当前所选的硬件型号不支持的 S 元件地址。 比如在 EC20 产品中，可用的 S 软元件地址是 S0～S991。对于初始步进符元件，可用的 S 软元件地址只有 S0～S19，对于普通步进符元件，可用的 S 软元件地址是 S20～S991。
解决措施	

错误代码	E1110
错误简述	无法处理的连接线
错误原因	在指令列表和梯形图程序中，如果它们的代码转换为顺序功能图后，出现类似于 E1100 错误中的连接关系，将提示该错误
解决措施	

错误代码	E1111
错误简述	出现交叉连接关系，无法转换
错误原因	在顺序功能图的网络结构中，是不允许出现类似于下图所示的交叉连接关系的：  目前在顺序功能图中是无法写出上面的程序的，但是在指令列表和梯形图程序中可以写出与上图中类似的连接关系，此时如果编译程序，将提示该错误。
解决措施	

错误代码	E1112
错误简述	该指令只能在主程序（不含内置）中使用
错误原因	在可用的指令中，有些指令被限制为只能在主程序中使用，在其他程序中调用就会提示该错误，比如 STL 指令。
解决措施	

错误代码	E1113
错误简述	该指令只能在子程序中使用
错误原因	在可用的指令中，有些指令被限制为只能在子程序中使用，在其他程序调用该指令将导致该错误，比如 CSRET 指令
解决措施	

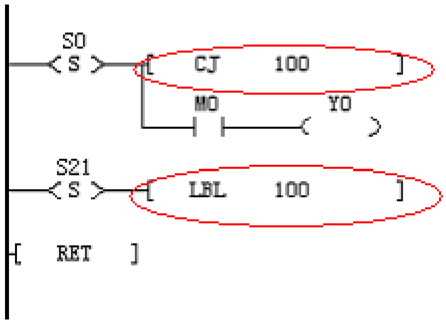
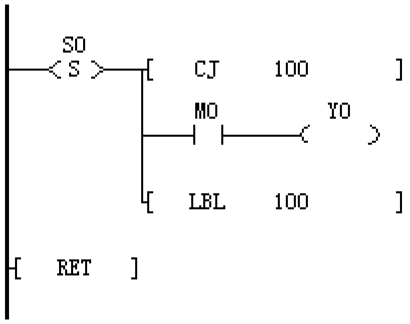
错误代码	E1114
错误简述	该指令只能在中断子程序中使用
错误原因	在可用的指令中，有些指令被限制为只能在中断子程序中使用，在其他程序中调用会导致该错误。比如 CIRET 指令
解决措施	

错误代码	E1115
错误简述	生成的 SFC 网络过于复杂，无法显示
错误原因	在从指令表或梯形图向顺序功能图转换时，可能存在逻辑太复杂而无法转换到顺序功能图的情况，此时将提示该错误
解决措施	

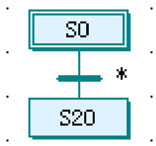
错误代码	E1117
错误简述	该指令不能在转移符的内置梯形图中使用
错误原因	由于转移符的特殊性，有些指令在转移符内置梯形图程序中是无法使用的，在内置梯形图程序中使用这些指令将导致该错误。比如 OUT S**
解决措施	

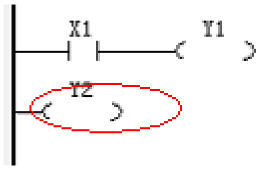
错误代码	E1118
错误简述	CALL 指令不能在中断子程序中使用
错误原因	中断子程序中禁止调用其他子程序，因此 CALL 指令不能在中断子程序中使用
解决措施	

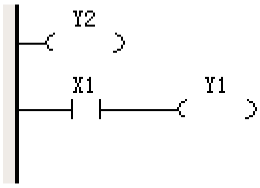
错误代码	E1119
错误简述	该指令只能在转移符的内置梯形图中使用
错误原因	TRANS 指令只能在转移符的内置梯形图中使用，如果在其他程序中使用 TRANS 指令，将提示该错误。
解决措施	

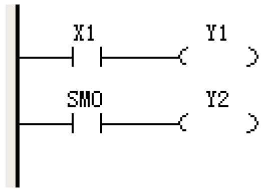
错误代码	E1120
错误简述	步进符内的 CJ 不能跳转到步进符外
错误原因及解决措施	如果在步进内使用 CJ，则相应标号的定义必须在该步进内，否则将提示该错误。 示例： 错误的图形：  正确的图形： 

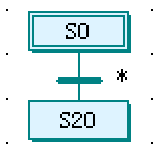
错误代码	E1121
错误简述	CJ 的标号不存在或者标号在步进内
错误原因	1）CJ 的标号不存在。 2）如果 CJ 不在步进内（如在 LAD 内），则不允许跳到步进内
解决措施	

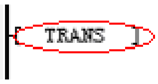
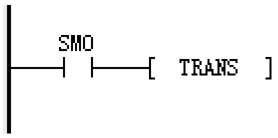
<table>
<tr><td>错误代码</td><td>E1122</td></tr>
<tr><td>错误简述</td><td>该指令不能连接内置梯形图的左母线</td></tr>
<tr><td>错误原因及解决措施</td><td>1．对于顺序功能图中步进的内置梯形图，输出指令或者应用指令如果直连左母线，并且该指令前面还存在一个多于两条指令的网络，则提示该错误。
示例：
错误的图形：


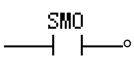
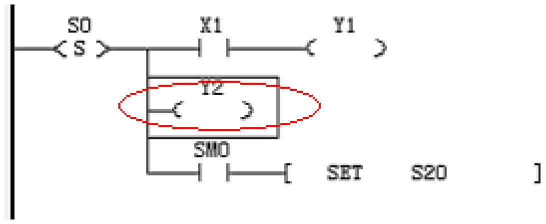
S0 的内置梯形图为：

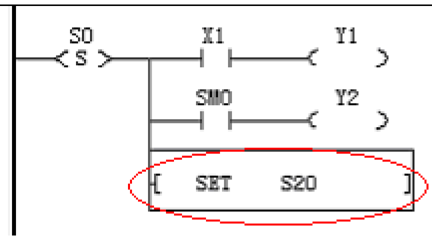
正确的图形：


或者

2．对于顺序功能图中转移的内置梯形图，如果其中只有一个 TRANS 指令，则提示该错误。
示例：


如果上图中转移的内置梯形图为则提示该错误。
正确的图形：
将转移的内置梯形图修改为：


3．存在上述情况 1 和情况 2 的顺序功能图，其对应的梯形图程序也会提示该错误。
解决措施是在提示错误的元件前增加一个元件。
示例：
错误的图形：

</td></tr>
</table>

	 4．存在上述情况 2 的顺序功能图，其对应的 IL 程序将提示该错误，解决措施是在提示错误的行的前面增加一行 LD SM0。 示例： 错误的程序： STL S0 OUT Y1 OUT Y2 SET S21 *//转换后的转移符的内置梯形图只有一个 TRANS* RET