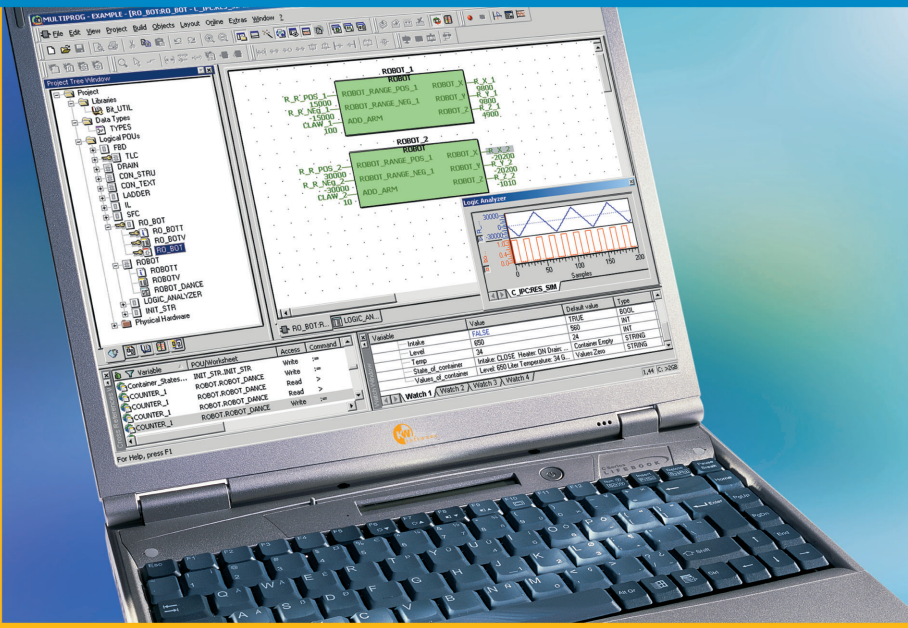
KW software



MULTIPROG®

Quick Start Guide

MULTIPROG
快速入门指南

手册发行日期：2002 年 4 月

KW-Software GmbH – Far East Office
〒107-0062 東京都港区南青山 1-15-18

电话： +86 10 62015642
传真： +86 10 82015862
网址： http://www.kw-software.com
电子邮件：china-info@kw-software.com

Software

内容

内容 **2**

介绍 **4**

关于本手册 5

PLC 编程系统要求 **6**

硬件要求 6

软件要求 6

安装软件 **7**

开发一个示例工程 **11**

用工程向导创建一个新的工程 11

启动工程向导 11

使用工程向导 12

开发 LD 代码 17

插入 LD 网络 17

声明属性 18

用编辑向导插入一个计数器 22

插入计数器的'RESET'触点 24

声明计数器的'RESET'触点的属性 25

插入第二个 LD 网络，并编辑网络描述注释 30

编译示例工程 37

'制作'工程 37

处理错误和消息 39

将工程下装到 PLC 或 IO 仿真程序 40

调试工程 42

调试模式 42

联机编辑 43

交叉参考窗口 46

变量监视窗口 47

强制和覆盖 48

断点 49

打印工程文档 52

选择打印机 52

设置页面布局 52

打印工程 53

打印预览 53

打印单个工作单 54

其它特征 **55**

使用 I/O 配置 55
创建一个用户自定义的函数 58
修改任务循环时间 66
附录 68
编程系统中的 IEC 工程组件 68
程序组织单元 (POU) 68
POU 和功能块的实例化 70
变量和数据类型 70
变量类型 70
变量地址 70
数据类型 72

介绍

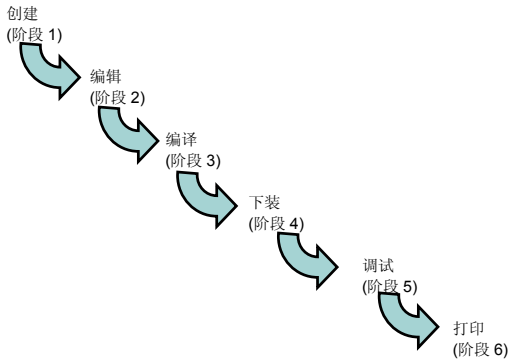
MULTIPROG 是一个标准的编程系统，可用于根据 IEC 标准设计的 PLC 以及传统的 PLC。它基于 IEC 61131-3 标准，并且包括了 IEC 特征的全部范围。

这个编程系统基于现代的 32 位 Windows 技术，提供了便捷的操作，包括缩放、拖拽和可对接窗口。系统允许处理 IEC 配置元素，允许包含库，并提供了一个功能强大的调试系统。对于 MULTIPROG，所有的功能都可以容易地通过菜单访问，并且，只需要用几个对话框就可以完成一个工程的产生。完成之后，您可以立即开始开发您的程序。

编程系统包括一个独立于 PLC 的内核，用于运用各种 IEC 编程语言编程。为此，提供了 ST 和 IL 等文本语言以及 LD、FBD 和 SFC 等图形语言。每个编辑器提供了一个编辑向导，使得插入关键字、语句、操作符、函数和功能块时尽可能地快而且容易。编辑向导也可以用于声明数据类型。适应于不同 PLC 的专用部分完善了编程系统的独立内核。

关于本手册

本手册提供了对于使用 MULTIPROG 开发、编辑和运行一个梯形图(LD)示例程序的循序渐进的指导。示例工程的开发被分为几个阶段，如下图所示：



每一阶段 – 从工程产生到工程文档，都将被详细阐述而无任何缝隙。

本手册中所解释的示例应用程序是一个基本的马达控制电路。该逻辑需要操作员按三次启动按钮，以便启动马达。马达在 20 秒之后停止运行。

PLC 编程系统要求

硬件要求

设备	至少	推荐
带有奔腾处理器的 IBM 兼容 PC 机	200 MHz	350 MHz
系统内存	64 MB	128 MB
硬盘	60MB 空闲 存储空间	
CD ROM 驱动器		
VGA 监视器 颜色设置 分辨率	 256 色 800 x 600	 真彩色 1024 x 768
RS232 接口	可选	
鼠标	推荐	

软件要求

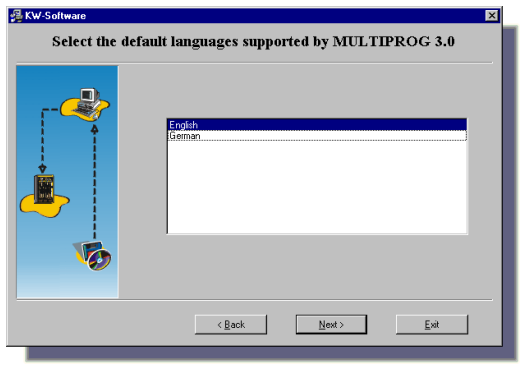
- Microsoft Windows 95, Microsoft Windows 98, Microsoft Windows NT 4.0 SP5 或者 Windows 2000
- Microsoft Internet Explorer 4.02

安装软件

要想安装此软件，请将其 CD 插入您的 CD ROM 驱动器。

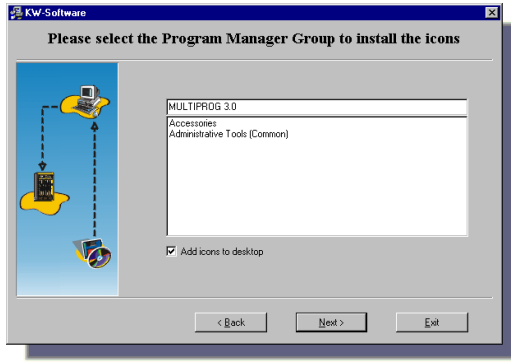
执行 setup 文件来启动安装向导，它将指导您完成安装过程。

点击'Next'继续。



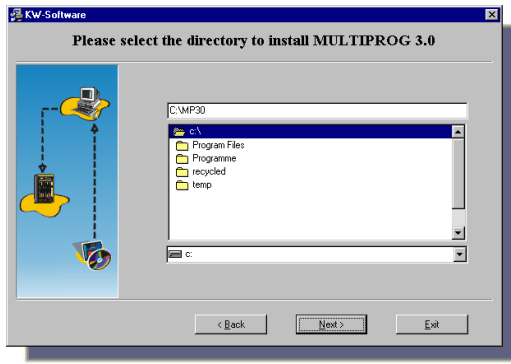
请选择安装向导安装 MULTIPROG 3.0 时，您所要使用的默认语言。.

点击'Next'继续。

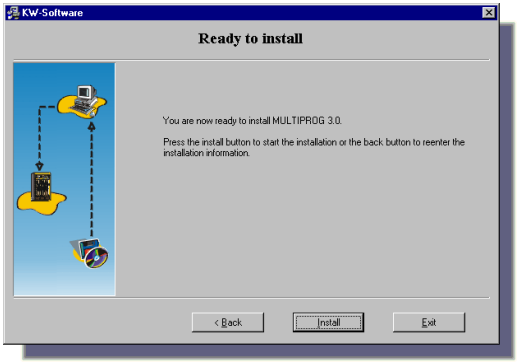


选择您希望的程序管理器组，安装向导将会把它添加到您的系统上。'Multiprog 3.0'作为默认组给出。如果您不希望安装向导在您的桌面上添加一个图标，请取消对相应复选框的选择。

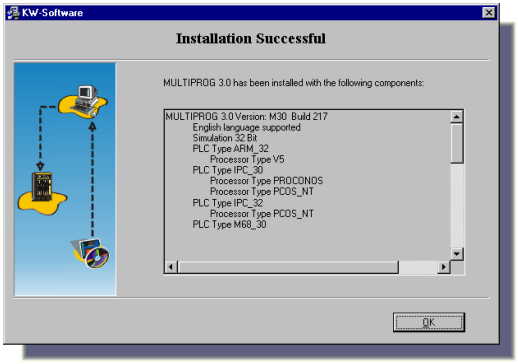
点击'Next'继续。



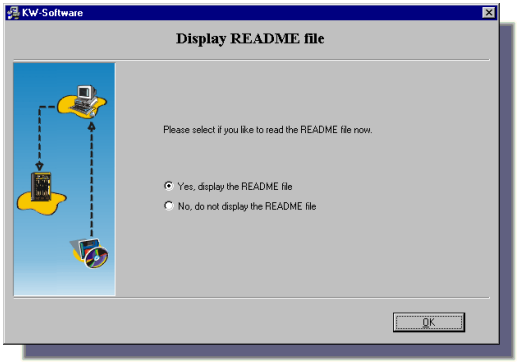
选择将要安装 MULTIPROG 的目录。默认路径已被给出。点击'Next'继续。



要开始安装，请点击'Install'。如果您想中止安装，请按‘Exit’按钮。如果您想修改任何选项，请点击'Back'返回到前一步。



点击'OK'继续。



选择您是否想要阅读 README 文件，其中含有关于 MULTIPROG 的最新信息。如果您不想阅读此文件，请选择 'No'。然后点击‘OK’。

在完成安装后，您必须重新启动您的计算机。点击'OK'。

开发一个示例工程

启动编程系统。

我们现在将使用梯形图(LD)编程语言开发一个示例工程。在第一个阶段，我们将创建一个新的空工程。

为了获得可能达到的最好结果，我们推荐使用我们在此手册中所使用的标志符和名称。

阶段 1 用工程向导创建一个新的工程

工程向导指导您在 6 步之内经历创建新工程的整个过程。在这里，您必须定义工程的名称和路径、编程语言、以及所使用的 PLC 类型。

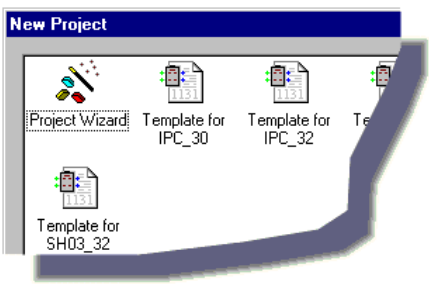
第 1 步

启动工程向导

点击'新建工程'图标：

出现'新建工程'对话框。

双击'工程向导'图标：

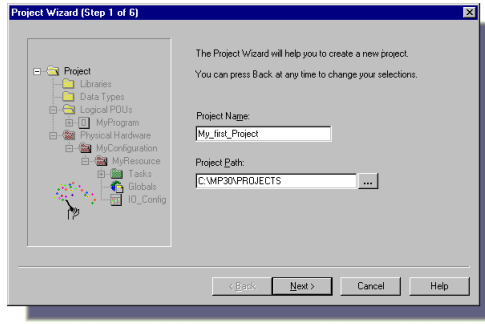


出现‘工程向导(第 1 步，共 6 步)’对话框。

第 2 步

使用工程向导

图 1:
用于指定工程名称和工程路径的对话框‘工程向导(第 1 步，共 6 步)’



填充对话框域，如下所示：

a. 将想要使用的工程名称('My_first_Project')输入到第一个输入域中，如上图所示。工程向导将会把工程保存到相应的文件'My_first_Project.mwt'中，并创建一个同名的子文件夹，其中将保存代码本体、变量的文件等等。

根据工程的命名规则，工程的名称和路径一定不能含有任何空格或特殊字符。工程名称最多能有 24 个字符。

工程的默认路径被自动输入。
如果您想要将您的工程保存到另一个路径，请在第二个输入域中指定这项内容，如下所述：

b. 点击浏览按钮：

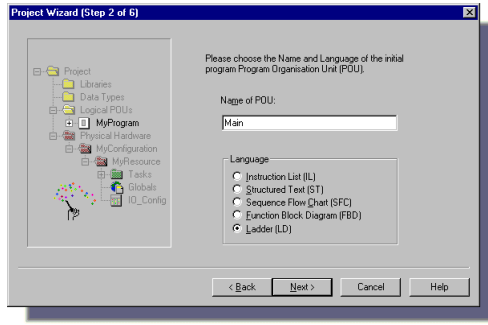
出现‘选择目录’对话框。

c. 为新工程选择一个文件夹。

d. 点击‘下一步’按钮继续。

出现‘工程向导(第 2 步，共 6 步)’对话框。

图 2:
用于输入第一个 POU 和选择编程语言的对话框'工程向导(第 2 步，共 6 步)'

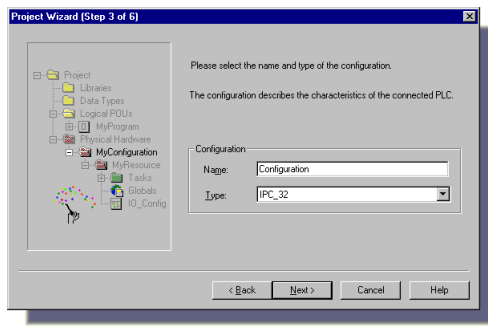


e. 输入第一个 POU 的名称，工程向导在创建工程时，会将它自动插入到工程树中。对于我们的示例 POU，输入'Main'。
通过激活相应的单选按钮，来选择用于第一个程序的语言。
因为我们想要用梯形图这种图形语言来对我们的示例工程编程，请选择'梯形图(LD)'。

f. 点击'下一步'继续。

出现‘工程向导(第 3 步，共 6 步)’对话框。

图 3:
用于选择配置名称和类型的对话框‘工程向导(第 3 步，共 6 步)’



g. 将所要使用的配置名称输入到输入域。一个配置可以被比作一个可编程控制器系统，例如一个导轨。在我们的例子中，我们输入了'Configuration'。

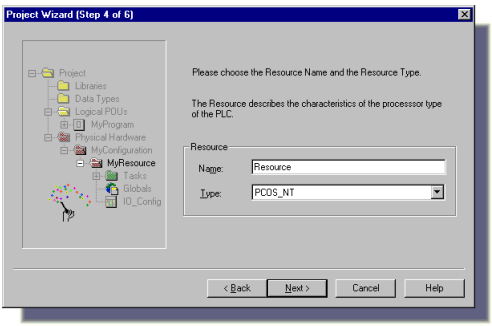
h. 为工程选择一个配置类型。这很有必要，因为系统在编译工程时，会产生专用于具体 PLC 的代码。
请在列表框中选择 PLC 类型。在我们的例子中，我们选择了'IPC_32'，所以编译器将产生用于 ProConOS 3.2 的 Intel®代码。

关于 PLC 类型的详细信息，请参考相应的 PLC 手册。

i. 点击'下一步'继续。

出现‘工程向导(第 4 步，共 6 步)’对话框。

图 4:
用于选择资源名称和类型的对话框‘工程向导(第 4 步，共 6 步)’

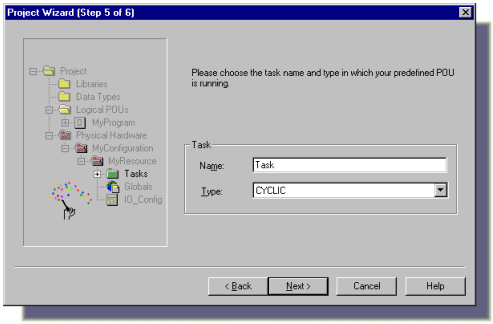


j. 为资源输入一个名称(在我们的例子中是'Resource')。资源可以被比作一个可以插入在导轨内(即：插入在配置内)的 CPU。
在列表框中选择资源类型。列表框只提供那些属于您所定义的配置的 CPU 类型，定义配置是在'工程向导(第 3 步，共 6 步)'对话框(请参考 图 3)中进行的。

k. 点击'下一步'继续。

出现‘工程向导(第 5 步，共 6 步)’对话框。

图 5:
用于选择任务名称和类型的对话框‘工程向导(第 5 步，共 6 步)’



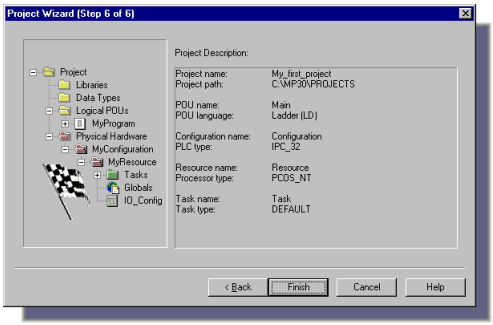
1. 将第一个任务的名称输入到输入域(在我们的例子中是'Task')。
在列表框中选择任务类型'CYCLIC'。

关于不同任务类型的详细信息可以在联机帮助系统中找到。

m. 点击'下一步'继续。

出现‘工程向导(第 6 步，共 6 步)’对话框。

图 6:
显示工程设置总览的对话框‘工程向导(第 6 步，共 6 步)’



这个对话框显示了对工程的描述，这些描述是对您在第 1 步到第 5 步中所做的设定的一个总览。如果输入了一个无效的名称，则会出现'无效的名称'等错误信息，并且'完成'按钮被

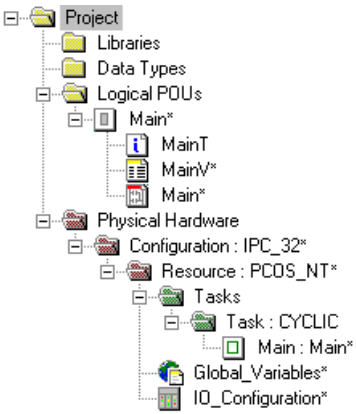
显示为灰色。如果出现这种情况，请检查受怀疑的标志符的拼写。

要想更正错误，请使用‘后退’按钮浏览相应步骤。务必要注意遵循定义标识符的规则。

n. 如果没有出现错误消息，请点击'结束'：

新的工程将被创建，并将被插入到工程树窗口中。
在工程树窗口中，您可以看一下这个新产生的工程：其 POU ‘Main’处于‘逻辑 POU’子树中，而配置、资源和任务处于‘物理硬件’子树中。

图 7:
工程树中自动产生的工程



程序'Main'的空的代码本体工作单被自动打开。

阶段 2 开发 LD 代码

既然我们已经创建了新的工程，我们将开始第 2 个阶段，开发工程代码。

为此，我们将使用梯形图(LD)编程语言。在完成编辑工程代码之后，可以编译、下装和调试工程。

在下面的步骤中，我们将解释如何

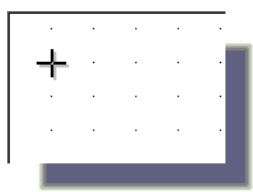
- 插入第一个 LD 网络
- 声明 LD 对象的属性，这些对象是随着第一个 LD 网络被自动插入的
- 在 LD 代码本体工作单中，用编辑向导插入并连接一个功能块
- 在 LD 代码工作单中，用连接模式插入和连接一个触点
- 声明触点和线圈的属性
- 插入第二个 LD 网络，并编辑网络描述注释

第 1 步

插入 LD 网络

我们想插入第一个 LD 网络'001'：

a. 用鼠标左键点击工作单内部，以在如下所示的位置设置一个插入标记。在此，将插入网络。



b. 点击'触点网络'图标，以插入 LD 网络：

c. 网络从一个触点和一个线圈开始，并且其宽度也被自动设定。

图 8:
工作单中新的 LD 网络

第 2 步

声明属性

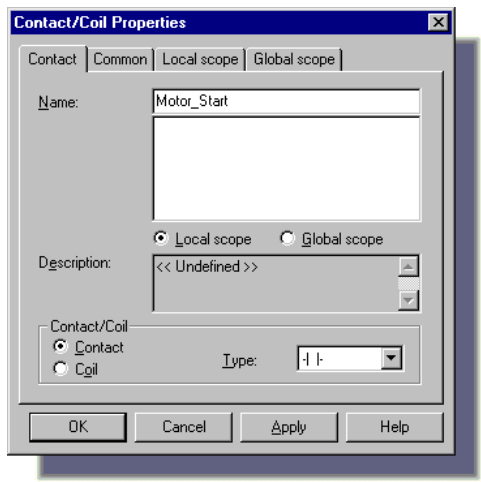
我们现在想声明 LD 对象的属性，这些属性已被自动插入到第一个 LD 网络

a. 双击触点'C000'。
作为一个选项，您可以通过先点击触点然后按<Enter>键的方法，来标记该触点。

出现'触点/线圈属性'对话框。

b. 将变量名从其默认名称'C000'改为'Motor_Start'。

图 9:
用于设置触点属性的'触点/线圈属性'对话框

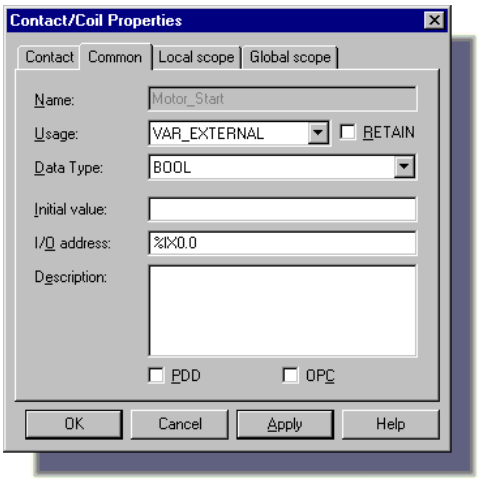


c. 点击'应用'或者按<Enter>键。对话框的'公共'页面被自动打开。

d. 在'用途'列表框中（见 图 10），选择'VAR_EXTERNAL'，以便将变量声明为全局变量，这意味着它可用于工程中的每个 POU。

e. 我们现在想将变量'Motor_Start'指派给我们的 I/O 仿真程序，这样，我们可以在屏幕上测试逻辑。为此，我们必须在输入域'I/O 地址'中分配变量的物理 PLC 地址。因为这个触点是打算用于启动马达的，我们需要声明一个输入变量。
输入'%IX0.0'，作为对定位变量'Motor_Start'的声明。这是仿真程序的位置，其中，'0.0'选定'第一个模块.第一个点'。

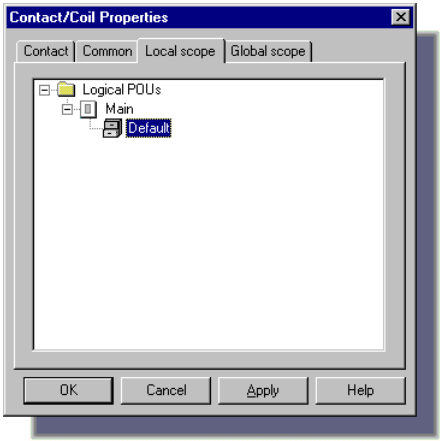
图 10:
用于设置触点属性的'触点/线圈属性'对话框



f. 打开对话框页面'局部范围'，并从树中选择'默认'。
通过选择这个项，我们定义：新变量的 VAR_EXTERNAL 声明将被插入到局部变量表格工作单的'默认'变量组中。

因为我们正在声明一个全局变量（使用 VAR_EXTERNAL），我们必须两个都选：一个局部范围和一个全局范围。

图 11:
用于设置触点属性的'触点/线圈属性'对话框



g. 打开对话框页面'全局范围'，并从树中选择'默认'。
通过选择这个项，我们定义：新变量的 VAR_GLOBAL 声明将被插入到资源的全局变量表格工作单的'默认'变量组中。

因为我们正在声明一个全局变量（使用 VAR_EXTERNAL），我们必须两个都选：一个局部范围和一个全局范围。

图 12:
用于设置触点属性的'触点/线圈属性'对话框

关于变量的声明以及局部和全局变量的功能等方面的详细信息，请参考附录中的'变量和数据类型'一节。

h. 点击'确定'，确认'触点/线圈属性'对话框。变量及其声明将被插入。

现在，您的屏幕应该显示如下：

图 13:
带有全局输入变量 'Motor_Start'的 LD 代码工作单

第 3 步

用编辑向导插入一个计数器

现在，我们想用编辑向导将一个计数器插入并连接到 LD 代码工作单中。

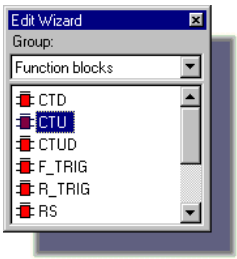
a. 因为我们想在'Motor_Start'和'C001'之间插入计数器，请点击其间的连线，以标记它：

图 14:
所标记的线，用于指定插入计数器的位置

b. 如果编辑向导还未被打开，请点击工具栏中的'编辑向导'图标：

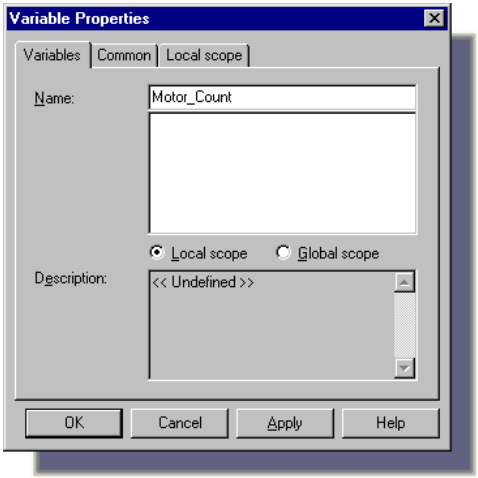
出现编辑向导。

c. 打开'功能块'组 (如果还没有打开)。在功能块列表中，浏览查找'CTU'列表项，并双击它。



出现'变量属性'对话框。

图 15:
用于设置计数器属性的对话框'变量属性'

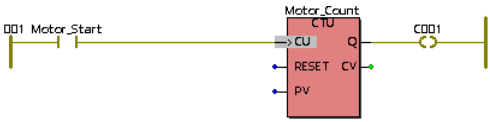


d. 在'名称'输入域中，将实例名称从其默认名称改为'Motor_Count'，并点击'确定'。对话框的'公共'页面被自动打开。通过'确定'来确认此对话框。

e. 通过再次点击'编辑向导'来隐藏编辑向导：

您的工作单现在看起来应该是如下的样子：

图 16:
插入了'CTU'计数器的 LD 代码工作单



第 4 步

插入计数器的'RESET'触点

现在，我们想要定义 CTU 功能块的'Reset'输入端，并将它连接到左侧电源轨线。

a. 点击'Reset'以标记它。

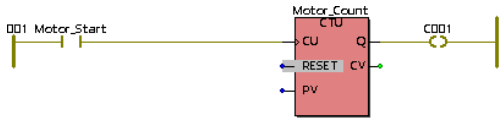


图 17:
计数器中所标记的输入，用于连接一个触点

b. 点击工具栏中的'添加左触点'图标，以在'Reset'输入端的左侧插入一个触点。

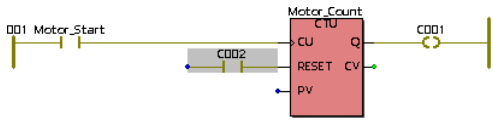


图 18:
在 LD 网络中插入一个触点

c. 点击工具栏中的‘连接对象’图标：

d. 点击'C002'触点，使其成为连线的起始点。

e. 将鼠标移动到左侧电源轨线，并再次点击鼠标，以终止连线。

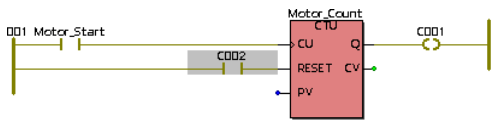


图 19:
使用连接模式连接触点

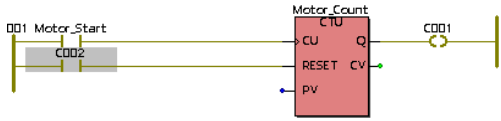
移动触点：

f. 通过按<Esc>键或者点击'标记'图标，将连接模式切换到标记模式：

g. 点击触点'C002'。
向左侧电源轨线方向拖拽此触点，直到它定位于'Motor_Start'的下方。

h. 释放鼠标按钮，以放置触点'C002'：

图 20:
移动触点



第 5 步

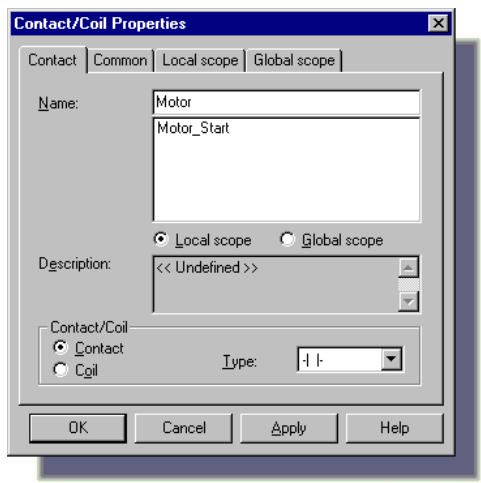
声明计数器的'RESET'触点的属性

接下来，我们声明复位触点'C002'的属性。

a. 双击'C002'触点，以打开'触点/线圈属性'对话框。

b. 将其名称从其默认名称'C002'改为'Motor'。

图 21:
用于设置触点属性的'触点/线圈属性'对话框



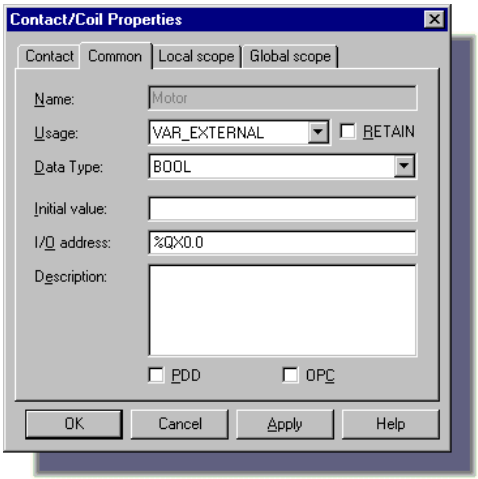
c. 点击'应用'或者按<Enter>键。对话框的'公共'页面被自动打开。

d. 在'用途'列表框中 (见 图 22)，选择'VAR_EXTERNAL'，以便将变量声明为全局变量，这意味着它可用于工程中的每个 POU。

e. 我们现在想将变量'Motor'指派给我们的 I/O 仿真程序，这样，我们可以在屏幕上测试逻辑。

输入域'I/O 地址：'指定变量的物理 PLC 地址。输入'%QX0.0'作为变量'Motor'的声明，其中'Q'表示'输出'，而'0.0'表示'第一个输出模块.第一个点'。

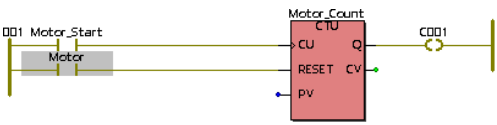
图 22:
用于设置触点属性的'触点/线圈属性'对话框



因为我们已经为第一个变量'Motor_Count'指定了局部或全局范围(即：变量组)，所以，有必要为相同组中的所有进一步的变量选择一个组。
关于位置和尺寸前缀的进一步的信息，请参考附录中的'变量和数据类型'一节。

f. 点击'确定'或按<Enter>键，确认'触点/线圈属性'对话框。现在，您的 LD 工作单应该显示如下：

图 23:
LD 代码工作单中，新近声明的变量'Motor'



定义计数器参数

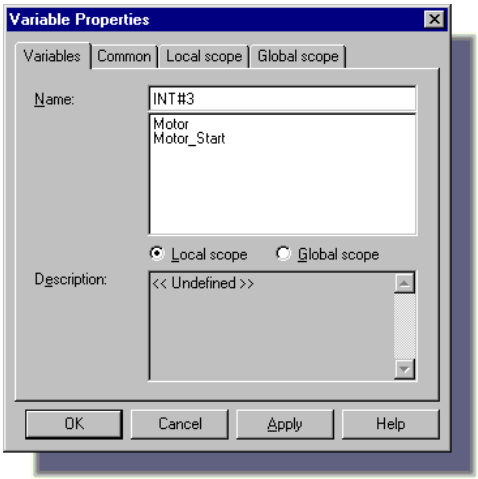
定义计数器的预置值：

g. 双击预置值'PV'的蓝色连接点。

出现'变量属性'对话框 (见 图 24)。

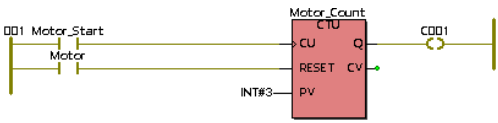
h. 因为我们想让马达在按了 3 次启动按钮之后启动，所以，请在'名称'输入域中输入'INT#3'，作为预置值。'INT'意味着'Integer'，'#'指示一个常量，而'3'是实际值。

图 24:
用于定义变量的类型和名称的对话框'变量属性'



i. 通过'确定'来确认此对话框。整数值被直接插入到代码本体中。

图 25:
LD 代码工作单中，新近声明的常量



配置计数器的当前值'CV'：

j. 双击当前值输出'CV'的绿色连接点。

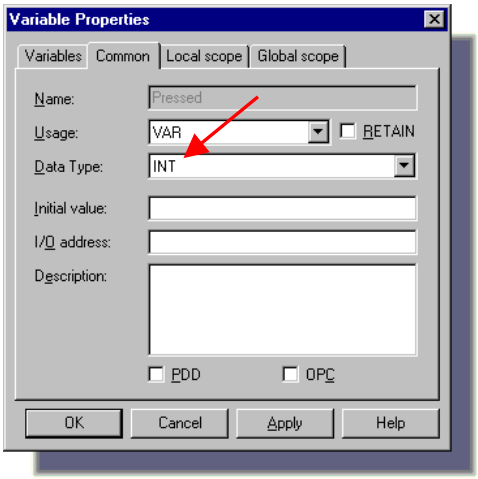
出现'变量属性'对话框。

k. 输入'Pressed'，作为变量名。计数器的当前值现在将被保存到变量'Pressed'中。

l. 点击'OK'。对话框的'公共'页面被自动打开。

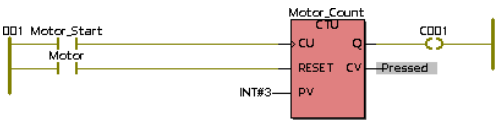
因为当前值是一个整数，所以，请从'数据类型'列表框中选择'INT'数据类型。

图 26:
用于设置变量属性的'变量属性'对话框



m. 点击'OK'。您的工作单看起来应该是如下的样子：

图 27:
LD 代码工作单中，新近声明的变量

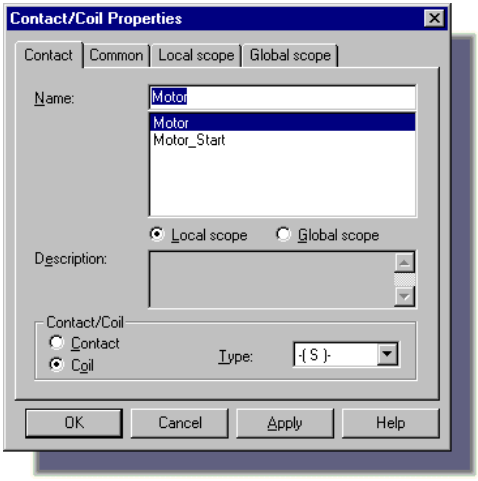


配置线圈'C001'：

n. 双击'C001'。出现‘触点/线圈属性’对话框。

o. 对于这个线圈，我们选择现有变量'Motor'。在'名称'输入域下面的列表含有已经声明的所有变量(局部的或全局的)，取决于所激活的范围单选按钮。
从列表中选择局部变量'Motor'。

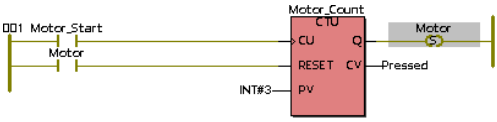
图 28:
用于为线圈选择变量的'触点/线圈属性'对话框



p. 因为我们想让马达在启动之后就连续运转，所以，我们使用了一个 SET 线圈。
从'类型'列表框中选择 '-(S)-' (见 图 28)。

q. 点击'OK'。您的工作单看起来应该是如下的样子：

图 29:
所插入的'Motor'变量



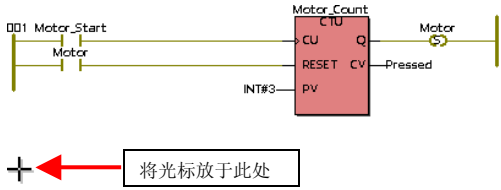
第 6 步

插入第二个 LD 网络，并编辑网络描述注释

现在，我们想要插入一个停止马达的逻辑，并要编辑网络描述注释。

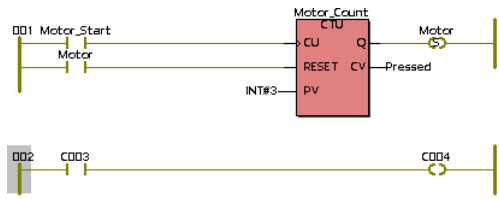
a. 在现有 LD 网络下方的适当距离处，点击鼠标左键，以便在如下所示的位置设置一个插入标记。

图 30:
用于插入第二个 LD 网络的插入标记



b. 点击'触点网络'图标，以插入一个新的 LD 网络：

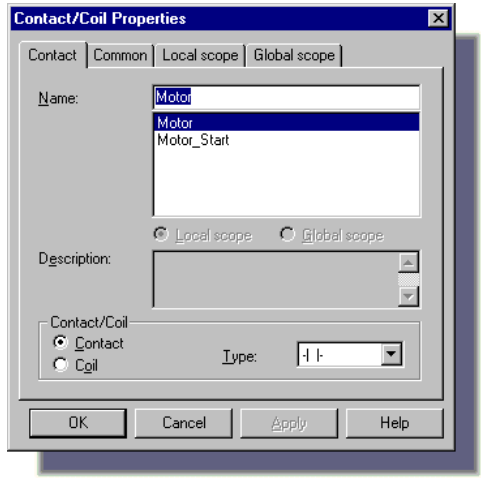
图 31:
第二个 LD 网络被插入到 LD 代码本体工作单中



c. 双击'C003'触点，以声明该触点的属性。

d. 在所出现的'触点/线圈属性'对话框内，从列表中选择局部变量'Motor'：

图 32:
用于为触点设置变量类型的'触点/线圈属性'对话框



e. 点击'OK'。变量'Motor'被插入在触点'C003'处。

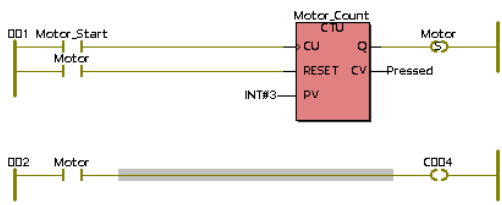
插入一个定时器：

现在，我们**将插入一个定时器**，它控制着马达将运行多长时间。

f. 通过点击工具栏中的'编辑向导'图标，打开编辑向导。

g. 在第二个 LD 网络中，标记'Motor'与'C004'之间的连线，以便在此位置插入并连接一个功能块。

图 33:
所标记的连线

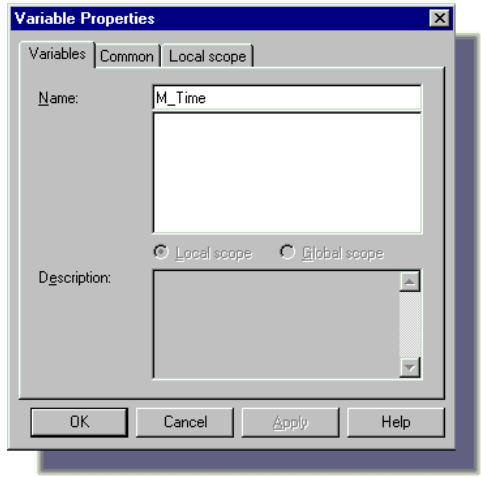


h. 从编辑向导的'组'列表框中，选择'功能块'组。

i. 从功能块列表中选择定时器'TON' ('Timer On Delay，接通延迟定时器')，并通过双击之而插入它。

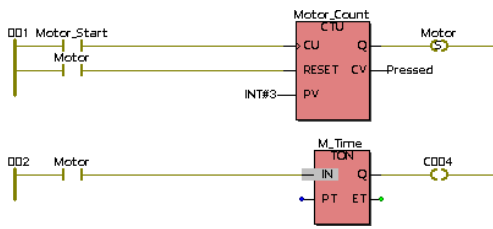
j. 出现'变量属性'对话框。在'名称'输入域中输入'M_Time'作为实例名称：

图 34:
用于声明实例名称的'变量属性'对话框



k. 点击'OK'。对话框的'公共'页面被自动打开。通过'确定'来确认此对话框。
因为您在插入对象之前已经标记了连线，所以，功能块将被直接插入并连接到指定的位置。
您的屏幕看起来应该是如下的样子：

图 35:
带有第二个 LD 网络和功能块'TON'的 LD 代码工作单



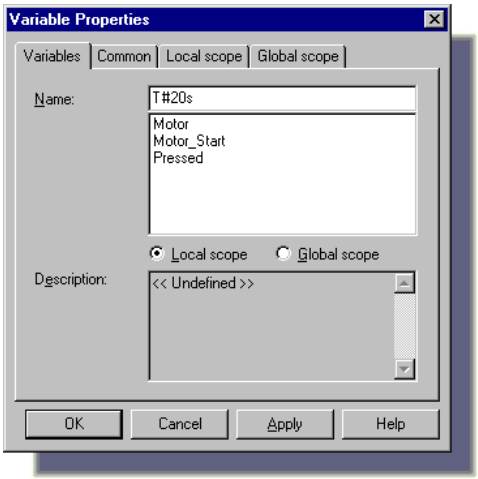
l. 通过再次点击'编辑向导'来隐藏编辑向导：

现在，我们想要**确定定时器的预置时间'PT'**，它将控制马达运行的时间长度：

m. 双击'PT'输入端的蓝色连接点。.

出现‘变量属性’对话框。

图 36:
用于声明一个局部变量的'变量属性'对话框



n. 在'名称'输入域中输入'T#20s'，作为时间常量。这里'T'表示一个时间值，'#'表示'常量'，而'20s'是 20 秒的实际时间值(因为我们想让马达运转 20 秒)。

o. 点击'OK'。常量'T#20s'被直接插入在 PT 输入端 (见 图 39)。

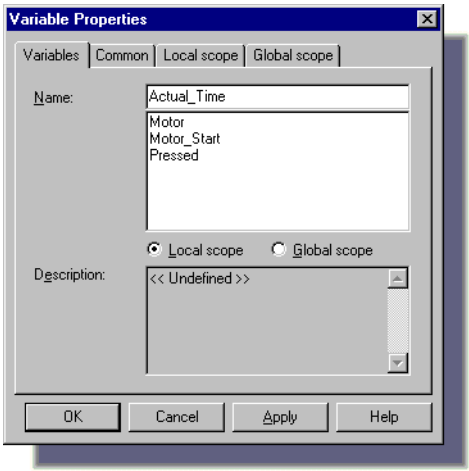
现在，我们将要**定义一个变量来保持已流逝的时间'ET'**：

p. 双击'ET'输出端的绿色连接点。

出现‘变量属性’对话框。

q. 输入'Actual_Time'作为这个局部变量的名称。

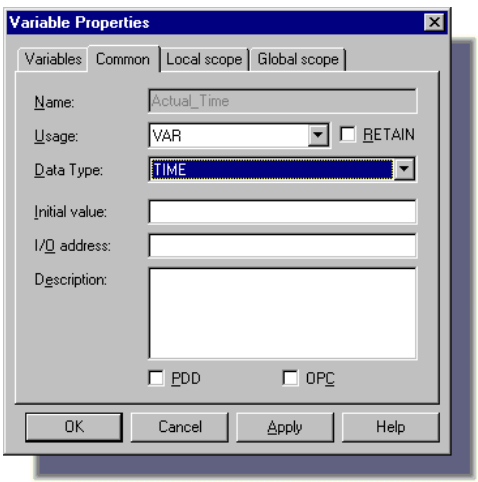
图 37:
用于声明一个局部变量的'变量属性'对话框



r. 点击'OK'。对话框的'公共'页面被自动打开。

s. 定时器的'ET'输出端需要一个数据类型为'TIME'的变量。因此，请从'数据类型'列表框中选择'TIME'数据类型。

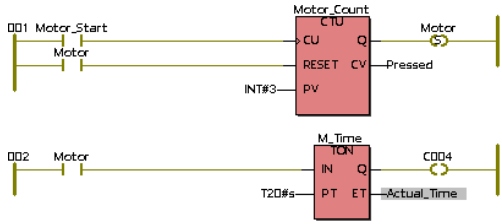
图 38:
用于声明一个局部变量的'变量属性'对话框



t. 点击'确定'来插入新近声明的变量。

您的工作单看起来应该是如下的样子：

图 39:
带有第二个 LD 网络和功能块'TON'的 LD 代码工作单

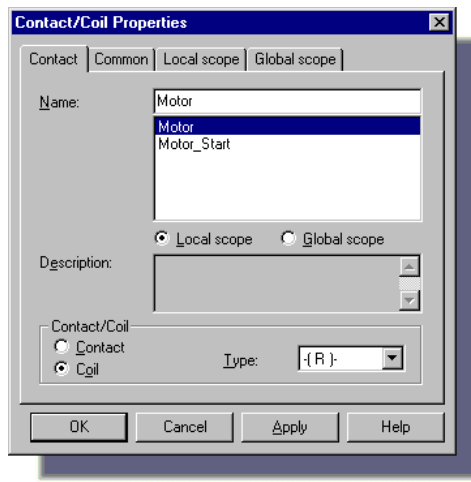


我们必须声明的最后一个变量是用于'C004'线圈的那一个。

u. 双击'C004'线圈，以打开'触点/线圈属性'对话框。从变量列表中选择'Motor'变量。

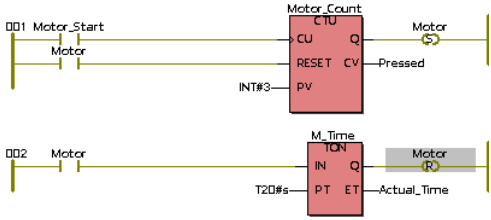
v. 这个线圈在接通时将会停掉马达。因为我们使用了一个**置位**线圈来启动马达，我们需要使用一个**复位**线圈来停掉它。因此，要通过选择相应的列表框中的项，将线圈类型设置为'RESET'，如下图所示。

图 40:
用于设置线圈和变量类型的'触点/线圈属性'对话框



w. 点击'OK'。您的工作单看起来应该是如下的样子：

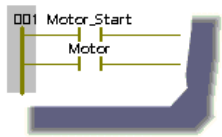
图 41:
带有所插入的第二个 LD 网络和'TON'功能块的 LD 代码工作单



最后，我们将插入网络描述注释。

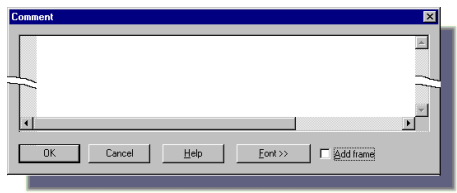
x. 双击 LD 代码工作单中的左侧电源轨线：

图 42:
所标记的电源轨线



出现'注释'对话框：

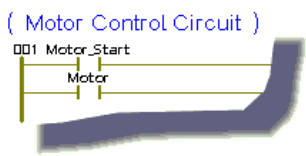
图 43:
用于在 LD 代码工作单中输入注释的'注释'对话框



y. 在'注释'对话框中，键入'Motor Control Circuit（马达控制电路）'。

通过点击'字体>>'按钮，您可以修改字体属性。选择'**蓝色**'作为颜色，并选择字体宽度'**20**'。

z. 点击'OK'。



阶段 3 编译示例工程

既然完成了编辑过程，我们必须编译此工程。在编译过程中，工作单的内容被翻译和转换为可以由 PLC 执行的特殊代码。

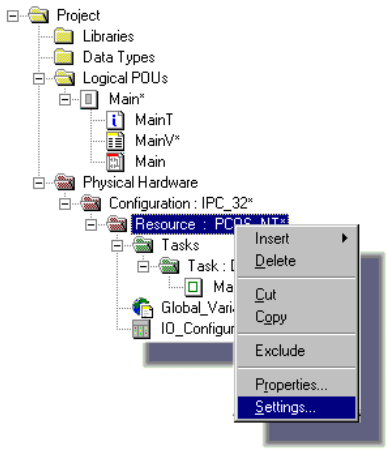
编程系统提供了几种编译的可能性。详细信息请参考联机帮助。

第 1 步

'制作'工程

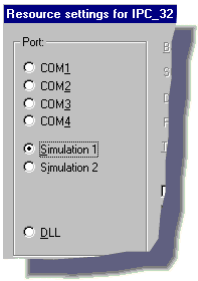
a. 在我们的例子中，我们使用仿真程序。要确保仿真程序已被激活。为此，要在工程树中用鼠标右键点击'资源'文件夹，并从所出现的上下文菜单中选择'设置...'：

图 44:
带有用于调用资源设置的上下文菜单的工程树



出现'资源设置'对话框：

图 45:
用于设置输出设备的'资源设置'对话框

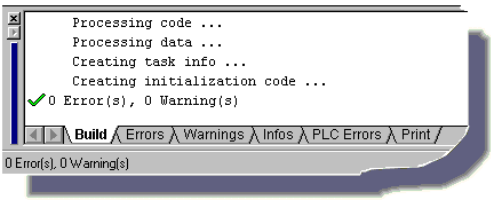


b. 激活'仿真程序 1'（如果有必要），并用'确定'来关闭对话框。

c. 点击工具栏中的'制作'图标：

编译过程被显示在消息窗口的‘建立’标签内。编译过程中所检测出的错误和警告信息被记录在消息窗口的相应页面内。你可以利用消息窗口通过双击错误消息来访问某个特定的代码本体工作单。

图 46:
用'制作'命令编译完工程后的消息窗口



第 2 步

处理错误和消息

在'制作'过程中，您将可能检测到错误和警告。

错误将阻止编译过程的完成，并将这类结果算入语法错误或结构问题。
警告指出潜在问题，如：某个变量未被使用。警告并不阻止编译过程的完成。

您可以忽略警告，但是您必须定位错误，以继续下去这个练习。

- 要显示所检查到的错误，请在消息窗口中点击'错误'标签。于是，一个错误列表就显示于消息窗口中了。
- 要显示警告列表，请点击'警告'标签。
- 在大部分情况下，双击某个错误/警告，将直接打开发生编程错误/警告原因所在的工作单。相应的行或对象被标记。您也可以标记错误并按<SHIFT> + <F1>，来得到相应的帮助主题，这些帮助主题含有关于出错的原因以及解决这些问题所需的更进一步的步骤等方面的信息。
- 解决所有的错误(如果出现过)，并用'制作'图标重新编译工程。
- 只有在那时，您才可以将程序下装到 PLC。

阶段 4 将工程下装到 PLC 或 IO 仿真程序

现在，所编译的工程必须被下装(即：发送)到 PLC 或 I/O 仿真程序。

与 PLC 或仿真程序之间的通讯是通过 PLC 控制对话框（被命名为'资源'）来完成的。

当使用多个资源时，会使用不同的对话框来下装工程并控制目标系统。
请参考联机帮助系统，来获取详细信息。

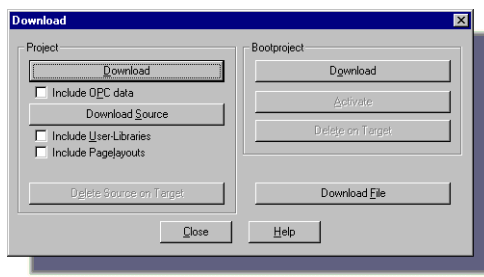
a. 点击'工程控制对话框'图标，以打开资源控制对话框。

出现标题栏上显示资源名称的控制对话框。

图 47:
用于控制 PLC 或仿真程序的'资源' 对话框

b. 按'下装'按钮。
出现'下装'对话框：

图 48:
用于开始工程下装的'下装'对话框



此对话框用于启动下装过程。您既可以向 PLC 或仿真程序发送一个"正常的"工程，也可以发送一个压缩了的工程源代码(可用作备份)。

c. 按下'工程'区域中的'下装'按钮，以下装工程：

成功的下装过程由一个位于屏幕底部的蓝色状态条指示。

d. 按下控制对话框里的‘冷启’按钮，来执行一个冷启动：

资源的状态由'停止'变为'运行'：

阶段 5 调试工程

下面，将讲解编程系统的调试工具。系统支持多中调试工具，提供了一种使您的应用程序联机运行的快速而容易的方法。

第 1 步

调试模式

可以将工作单从编辑模式(脱机模式)切换到调试模式(联机模式)，反之亦然。联机模式用于检测编程错误，以及确保 PLC 程序正确地运行。在联机模式下，显示变量的当前值或状态。

a. 要确保 PLC/仿真程序在运行。PLC 状态显示于'资源'控制对话框的顶部。如果程序没有运行，则要通过按下控制对话框中的'冷启'按钮，来执行一个冷启动。

b. 要想激活调试模式，务必要使我们的'Main' POU 的代码本体处于打开状态，并点击工具栏中的'调试开/关'图标：

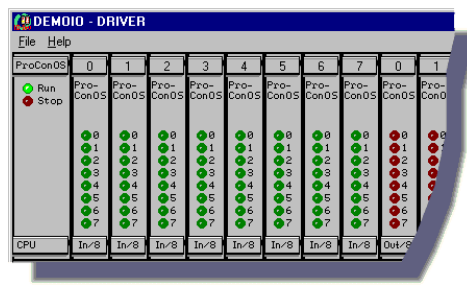
注意：变量的状态和当前值显示于多种颜色，指示不同状态：

- 兰色 = false
- 红色 = true

您可以通过点击'调试开/关'图标，在联机模式和脱机模式之间切换。

c. 点击 Windows 任务栏中的'DEMOIO - DRIVER'按钮，以打开 I/O 仿真程序：

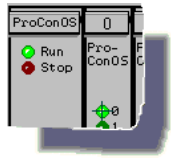
图 49:
I/O 仿真程序



d. 如果有必要，请将 I/O 仿真程序放置(通过拖放)到屏幕的一个角上，以免遮挡工作单。

e. 通过点击第一个输入模块的第一个绿色的"虚拟 LED"，来接通和断开模块 0 的第 0 位三次：

图 50:
切换位，以便启动示例程序



观察工作单中的反应

- 在标记'Motor_Start'三次之后，马达开始运转，因为当前值'CV'达到了预置值'PV' (注意屏幕上的更新)。
- 当'Motor'置位线圈被接通时，它也在第 002 号梯形图网络启动'M_Time'定时器。
- 'M_Time'(实际时间)运行 20 秒，直到'ET'(流逝了的时间)达到了预置的时间值'PT'。'Motor'复位线圈在第 002 号梯形图网络内被接通，因此，在第 001 号梯形图网络内解锁'Motor'置位线圈，并关掉马达。

第 2 步

联机编辑

可以在不停止 PLC/仿真程序上的程序执行的情况下，联机编辑。这种操作叫做'修补 POU'。当您使用'修补 POU'时，会编译您对代码所做的修改，产生相关的代码，并随后将其自动下装到 PLC。在整个修补过程中，PLC 上的代码执行没有被中断。

作为一个**修补 POU** 的例子，我们想为马达插入一个**急停**：激活'Emergency_Stop'输入将会立即停止马达。

a. 通过点击'调试开/关'图标，切换到脱机模式：

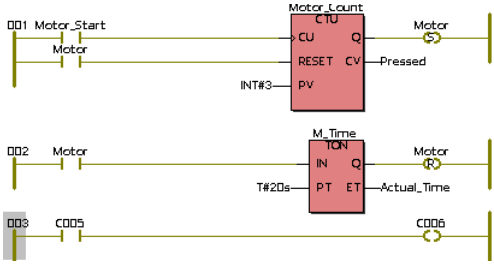
我们的代码本体工作单再次出现在编辑模式下。而资源，正如一个真正的控制器，仍在运行：

b. 在 LD 代码工作单中第 002 号梯形图网络的下方，设置插入标记。

c. 点击'网络'图标，插入一个新的 LD 网络：

您的工作单看起来应该是如下的样子：

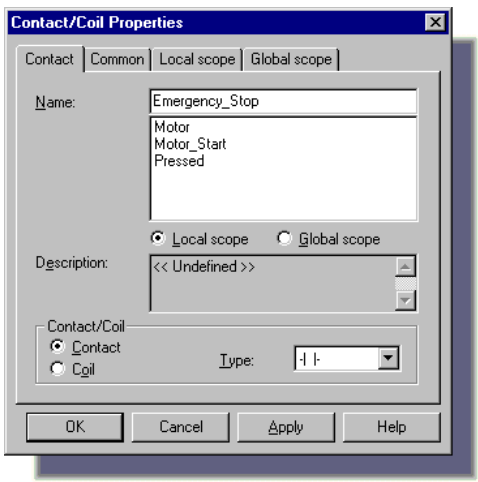
图 51:
插入一个 LD 网络



d. 双击'C005'触点，以打开'触点/线圈属性'对话框。

e. 将名称从默认名称'C005'改为'Emergency_Stop'。

图 52:
用于设置触点属性的'触点/线圈属性'对话框

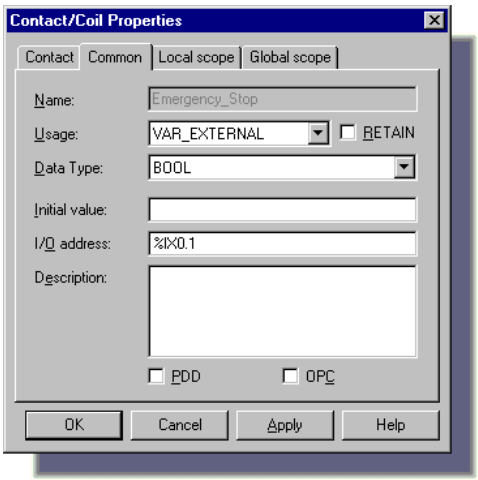


f. 点击'OK'。对话框的'公共'页面被自动打开。

g. 从'用途'列表框中，选择'VAR_EXTERNAL'，以便将'Emergency_Stop'声明为一个全局变量。

h. 我们将使用 I/O 仿真程序中的第二个输入点作为急停。在'I/O 地址'输入域中输入'%IX0.1'作为这个变量的地址，并点击'确定'。

图 53:
用于设置触点属性的'触点/线圈属性'对话框

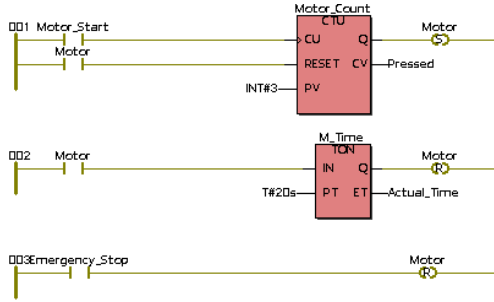


关于位置和尺寸前缀的进一步的信息，请参考附录中的'变量和数据类型'一节。

i. 双击'C006'触点。出现'触点/线圈属性'对话框。从'类型'列表框中选择'-(R)-'，并从全局'变量'列表框中选择'Motor'。然后点击'OK'。

您的工作单看起来应该是如下的样子：

图 54:
联机编辑，所修改的变量属性



既然我们已经修改了代码，我们就来进行一次修补 POU。这个过程将会在不停止 PLC 运行的情况下，编译所修改的部分，并将它们下装到 PLC 中。

j. 点击工具栏中的'修补 POU'图标，来编译所修改的代码，并将其下装到 I/O 仿真器：

在修补过程被成功地完成后，工作单将被自动设置为联机模式。

k. 点击 Windows 的任务栏中的'DEMOIO - DRIVER'按钮，以打开 I/O 仿真程序。

l. 通过点击相应的输入点(LED)，来接通和断开模块 0 的第 0 位三次 (请参考 图 50 ，它在第43页)。

m. 使用这个新的 Emergency_Stop 触点，可以通过点击 I/O 仿真程序中的输入模块 0 的第 1 位，而立即停止马达。

第 3 步

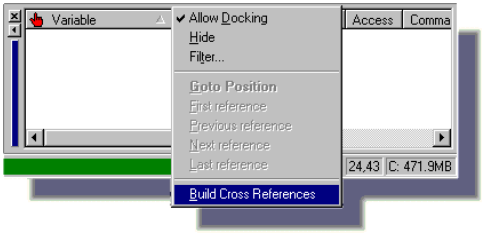
交叉参考窗口

交叉参考列表包含了当前工程中使用的所有变量、功能块、跳转、标号和连接符。这个工具对于调试和错误隔离特别有帮助。

a. 点击工具栏中的'交叉参考窗口'图标，以打开交叉参考窗口(如果还没有打开)：

b. 将光标放于交叉参考窗口中，并用鼠标右键点击窗口背景，以打开其上下文菜单：

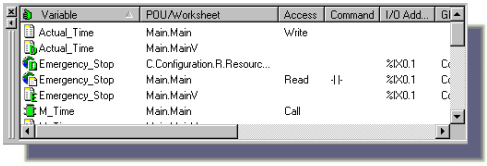
图 55:
带有用于建立交叉参考的上下文菜单的交叉参考窗口



c. 选择‘建立交叉参考’菜单项。

将创建交叉参考列表。

图 56:
示例工程中的交叉参考列表



d. 双击交叉参考窗口中的某个变量，将会打开使用了这个变量的工作单，并加亮此变量。
另外，如果您在工作单中标记一个变量，交叉参考窗口中的相应变量也将被标记。

e. 通过点击相应的工具栏图标，来关闭交叉参考窗口和消息窗口。


和

第 4 步

变量监视窗口

变量监视窗口是一个功能强大的工具，允许你很容易地将不同变量插入到列表中，并观察其运行期行为。一旦某个变量被添加到监视窗口，则不必打开相应工作单，就可以监视其当前值。结果，您可以专注于您想更容易地看到的那些变量。

a. 如果还不是这种情况，请将工作单切换到联机模式，这要通过按下'调试开/关'图标来进行。

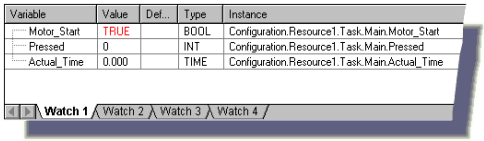
b. 用鼠标右键点击工作单内部，并从上下文菜单中选择'打开监视窗口...'，或者点击'监视窗口'按钮。

出现监视窗口。

c. 在联机工作单中，用鼠标右键点击'Motor_Start'，以打开其上下文菜单，并选择'添加到监视窗口'，以便将此变量插入列表中。

d. 对变量'Pressed'和'Actual_Time'重复这个过程。这三个变量将出现于列表中，如下图所示。

图 57:
变量监视窗口



您现在可以使用 I/O 仿真程序来操作各触点，并可以在逻辑中和监视窗口中同时观察变量值的变化。

第 5 步

强制和覆盖

在联机模式下，可以强制或覆盖变量。两种情况下，都有一个新的值被赋给相应的变量。

强制：将一个值赋给一个变量(通常是一个触点或线圈)。该值将一直保持到复位强制时。

覆盖：由用户将一个值临时赋给一个变量。该值将一直保持到程序在下一个程序循环中用原值再次覆盖掉这个值的时候。

强制和覆盖一个变量的必要步骤几乎是相同的。

当 PLC 运行时，要特别小心地进行强制或覆盖变量。强制和覆盖变量意味着用强制的或覆盖的变量值执行 PLC 程序。

在我们的例子中，我们想要强制**'Motor_Start'**变量：

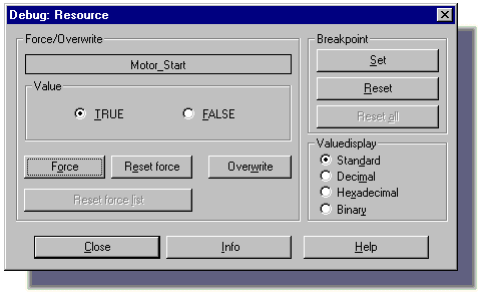
a. 要确保工作单处于联机模式。否则，按工具栏中的'调试开/关'图标：

b. 点击 Windows 任务栏中的'DEMOIO - DRIVER'按钮，以打开 I/O 仿真程序。

c. 要确保所有的输入都被设置为'FALSE'，这要通过点击每个点亮的 LED 来完成 (不应该点亮任何 LED)。

d. 双击'Motor_Start'。出现'调试：资源'对话框：

图 58:
'调试：资源'对话框，用于强制和覆盖变量



e. 选择单选按钮'TRUE'，然后点击'强制'。结果，'Motor_Start'将被强制为'接通'，并在联机工作单中被加亮为红色。

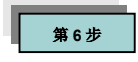
f. 再次双击'Motor_Start'，并选择'复位强制'来取消强制。

如果您重复 e 和 f 步骤，逻辑将开始执行。

g. 在'调试'对话框中，点击'复位强制列表'。

现在，我们想要覆盖'Motor'变量。

h. 双击第 001 号梯形图网络中的'Motor'线圈，然后点击'覆盖'。这会启动'Motor_Time'。在 20 秒之后，在第 002 号梯形图网络中的'Motor'复位线圈将关掉'Motor'。



断点

可以在所有工作单中联机地设置断点，这要使用'调试:资源'对话框的右边区域中的控件，如 图 58.所示：

当设置了断点时，程序的执行会暂停在断点处，直到开发者令其继续。编程系统提供了将程序一直执行到下一个断点(单步)或者一直执行到再次到达同一断点(单循环)的可能性。

如果到达了断点，PLC 状态就变为**暂停[调试]**，并且控制对话框显示出‘运行’、‘单步’和‘跟踪’等按钮，便于继续。

运行：点击'运行'会使程序一直运行到遇到下一个断点。

单步：点击'单步'会使程序执行下一条指令。

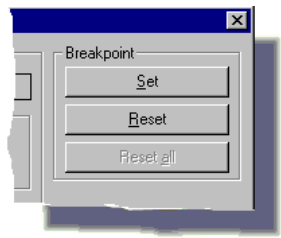
跟踪：如果到达一个用户自定义函数或功能块调用，则会打开函数或功能块代码本体，并且一步一步地调试。

当 PLC 在运行时，使用断点要非常小心，因为断点会实际上暂停程序的执行。到达断点时，I/O 的行为取决于 PLC 类型。

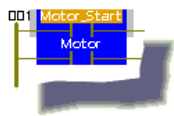
a. 要想观察断点操作，相应的工作单必须处于联机模式('调试开/关'图标被按下)：

b. 点击'PLC 控制'图标，以打开资源控制对话框：

c. 双击'Motor_Start'，并在'调试:资源'对话框中选择'设置'，以便在这个变量处设置一个断点。



在联机工作单中，'Motor_Start'被加亮为橙色，指示程序的执行将要停止的位置。



d. 按下'资源'对话框中的'运行'按钮，激活程序的执行，直到遇到下一个断点。

因为我们只设置了一个断点，所以，程序会再次停在'Motor_Start'。这叫做单循环。

e. 点击几次'单步'，可以看到每点击一次，橙色的加亮区都移动到下一条指令，指示出程序的执行已经停止了。这是一个单步执行。还可以看到'Motor_Start'有一个红色的加亮区，指示出断点设置的位置。

f. 双击'Motor_Start'，并按调试对话框中的'复位'，以复位断点。然后，点击'资源'对话框中的'运行'，以恢复程序的执行。

g. 再次点击'调试开/关'图标，以切换到脱机模式：

h. 在控制对话框中，点击'停止'按钮停止 PLC，并使用'关闭'按钮关闭'资源'控制对话框。

阶段 6 打印工程文档

打印整个工程对于文档管理来说是很有用的。编程系统提供了几种打印工程文档的可能性。'文件'菜单含有各种命令，用于预览当前页，用于定义打印机设置，用于打印整个工程或个别工作单。

第 1 步

选择打印机

打印机选项位于'文件'菜单中的'打印机设置...'。选择这一项将打开标准的 Windows 对话框'打印机设置'。

第 2 步

设置页面布局

一个页面布局定义了所要打印的工作单的外观，如：页面尺寸、页面边距(边框)、源代码区域、页脚和含有诸如公司徽标、日期、工程名称或页号等信息的页眉。

编程系统允许使用不同页面布局。而且，可以修改页面布局。当打印工程或其中的某些部分时，会自动使用默认页面布局。

要想改变默认页面布局：

a. 选择'附加 | 选项...'菜单。

b. 打开'页面布局'标签。

c. 选择想要使用的页面布局。在我们的例子中，我们使用了默认的页面布局。

页面布局是用页面布局编辑器创建和编辑的。关于进一步的信息，请参考您的联机帮助系统。

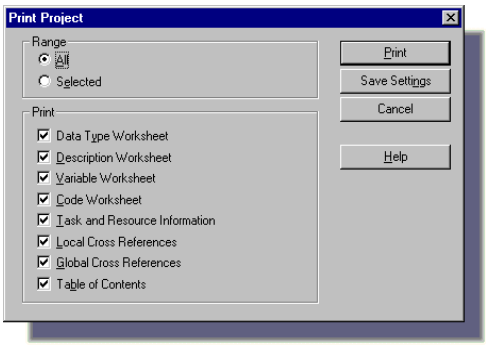
第 3 步

打印工程

a. 从'文件'菜单中选择'打印工程...'菜单项。出现'打印工程'对话框。

b. 在'打印工程'对话框中，取消对工程中您不想打印的那些部分的选择，这要通过撤消对相应复选框的选择来完成。

图 59:
'打印工程'对话框



c. 点击'打印'按钮。

第 4 步

打印预览

打印预览允许您看一下工作单在被打印出来时将会是什么样子，并在必要时修改它。这有助于用清晰而结构化的方法组织页面上的元素。

交叉参考不会显示于预览中。

如何调用预览：

a. 要确保您要看的工作单是活动窗口。

b. 从'文件'菜单中选择'打印预览...'菜单项。
将显示活动工作单的打印预览。

c. 要想打印所显示的那个工作单，请点击'打印'按钮。

第 5 步

打印单个工作单

您可以打印图形编辑器或文本编辑器中所打开的个别工作单。

使用‘打印’菜单项，不会打印交叉参考。

如何进行：

a. 要确保您想要打印的工作单是活动的。

b. 从'文件'菜单中选择'打印'菜单项。
该工作单将被打印。

其它特征

使用 I/O 配置

'I/O 配置'对话框用于编辑 I/O 配置工作单。I/O 配置一般包含了对 I/O 模块的声明，例如：模块的逻辑地址(开始和结束地址)，对设备的声明(驱动程序名称或存储器地址)，等等。

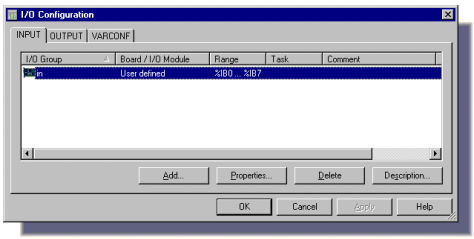
下列步骤解释了如何使用 I/O 配置。在我们的例子中，我们将现有 I/O 组中的输入模块的数目改为 10。

a. 要想修改 I/O 配置，请双击'物理硬件'子树中的'IO_Configuration'：

图 60:
'物理硬件'子树中的'IO_Configuration'图标

出现'I/O 配置'对话框：

图 61:
'I/O 配置'对话框



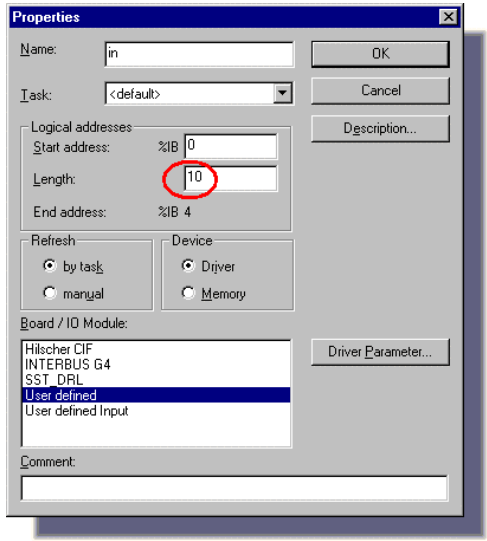
在这里，您需要从列表中选择您将要使用的驱动程序，并按照相应驱动程序手册中所叙述的来配置它。
当前 I/O 配置显示于对话框中。我们想要修改这个配置，即：我们想在现有组中定义 10 个输入模块。

b. 在'I/O 配置'对话框中，点击'属性'按钮：

出现'属性'对话框。

c. 在'长度'域中输入 10，并按<TAB>键，以更新'结束地址'域中的项：

图 62:
用于配置 I/O 仿真程序的'属性'对话框



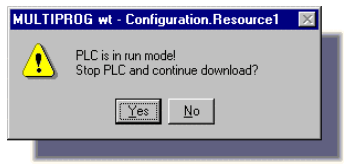
d. 用'确定'来确认'属性'和 I/O 配置对话框，以返回编程。

e. 通过点击工具栏中的'制作'图标，编译工程。

关于编译的详细信息，请参考本快速入门手册的第 37 页上的阶段 3。

f. 将工程下装到目标系统，如第 40 页上的阶段 4 中所述。

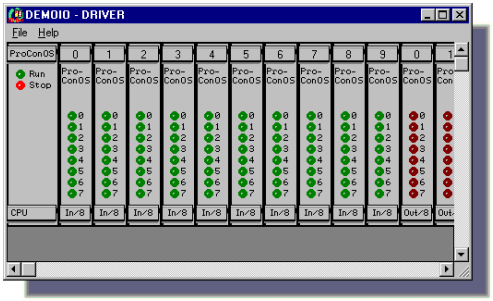
如果此时 PLC/仿真程序仍在运行，则在下装前会出现如下消息对话框：



如果是这种情况，请点击'是'继续下装。

g. 点击 Windows 任务栏中的'DEMOIO - DRIVER'按钮，以打开 I/O 仿真程序。现在应该有 10 个可用的输入模块：

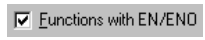
图 63:
在 I/O 配置被改为 10 个输入模块之后的 I/O 仿真程序



创建一个用户自定义的函数

在这一章中，我们想要在我们的工程中插入一个用户自定义的函数。该函数应该用文本语言 ST 来产生，它计算马达处于活动状态的时间。

在您开始之前：
选择'附加 | 选项'。在'选项'对话框，选择'图形编辑器'标签，并确保使能了'带有 EN/ENO 的函数'：



EN/ENO 为那些使用 LD 和 FBD 编程语言的 IEC 61131 函数指定了一个额外的布尔输入‘EN’(=使能)和输出‘ENO’(=使能输出)。

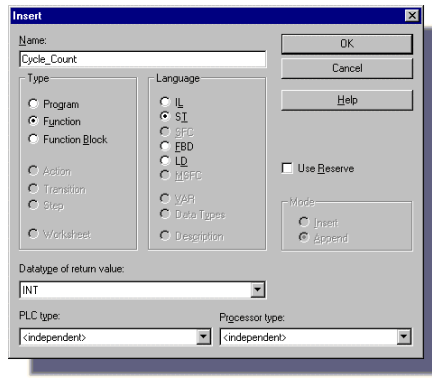
并非所有的目标系统都支持 EN/ENO。

a. 要想插入一个用户自定义的函数，请点击工具栏中的'添加函数'图标：

出现‘插入’对话框。

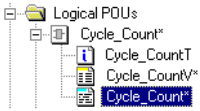
b. 输入'Cycle_Count'作为名称，并选择语言'ST'。在'返回值的数据类型'域中，您要指定应用于函数输出的数据类型。连接到函数输出的变量，必须符合其返回值的数据类型。在我们的例子中，我们使用'INT'。

图 64:
用于插入一个用户自定义 POU 的'插入'对话框



c. 按下'确定'。函数被添加到工程树中。函数名称末尾的星号，指示新的 POU 还没有编译。

d. 通过双击'Cycle_Count'代码工作单来打开它，以便编辑 ST 代码。



e. 在工作单中，输入下列代码：

```
1    Cycle_Count:=Count+1;
```

这行代码将产生一个值，每次启动马达，该值都连续地增加。

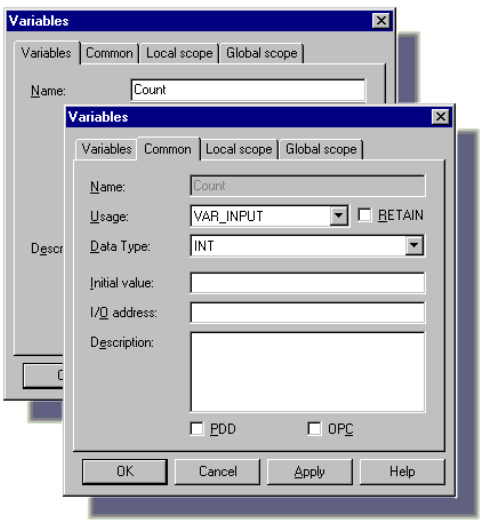
f. 将光标放在'Count'变量上：

```
1    Cycle_Count:=Count+1;
```

g. 点击'变量'图标

并将此局部变量声明为'INT'和'VAR_INPUT'，如 图 65 所示：

图 65:
‘变量’对话框



h. 通过'确定'来确认此对话框。声明将被自动插入到新的 ST POU 的变量工作单中。

i. 关闭 ST 工作单，并保存所做的修改。

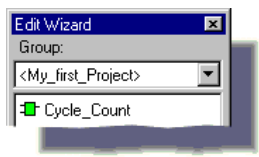
在关闭 ST 工作单之后，这个新的用户自定义函数就可用于编辑向导中，并可被插入本工程的其它工作单内。

我们现在想要在我们的程序 POU 'Main'中调用这个用户自定义的函数'Cycle_Count'。

j. 在程序'Main'的代码本体中，在 LD 网络'003'下方设置插入标记，并用'网络'图标插入一个新的网络。

k. 标记 LD 网络'004'中的 C007 和 C008 之间的连线。

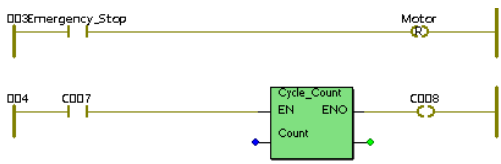
l. 打开编辑向导，并选择'My_First_Project'组：



这个组含有当前工程中的所有用户自定义函数和功能块(在我们的例子中，只有'Cycle_Count')。

m. 双击'Cycle_Count'函数，以便将这个用户自定义的函数插入在前面所指定的位置。

图 66:
所插入的用户自定义函数 'Cycle_Count'



完成这个之后，再次关闭向导。

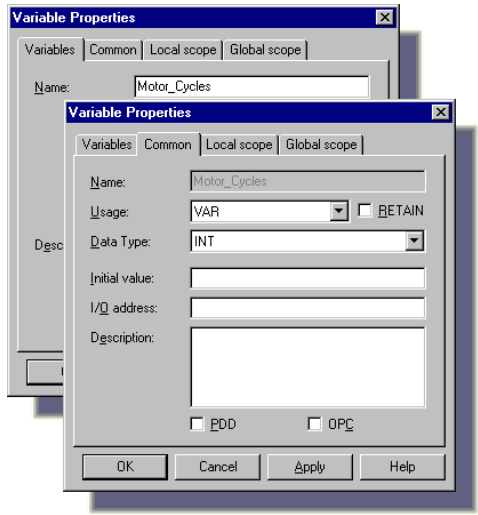
现在，我们必须将一个新的变量连接到'Count'输入端，这样我们可以看到内部值。

n. 双击'Cycle_Count'的'Count'输入端的蓝色连接点。

o. 出现'变量属性'对话框。

p. 按如下方法声明局部变量'Motor_Cycles'：

图 67:
'变量属性"对话框

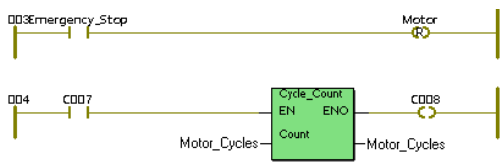


q. 用'确定'确认对话框，以便将变量插入到代码本体，并将其声明插入到局部变量工作单。

现在，必须将同一个变量连接到函数的输出上。

r. 双击'Cycle_Count'输出端的绿色连接点，并且从'变量属性'对话框的变量选择列表中，选择'Motor_Cycles'变量。

s. 通过'确定'来确认，以插入变量。



t. 使用'触点/线圈属性'对话框，将'C007'触点的名称改为'Motor'。要想打开此对话框，请双击触点。

要确保标记了'Motor'触点。

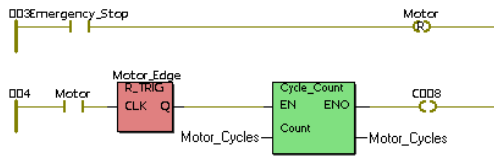
u. 激活编辑向导。选择'功能块'组，并通过双击 R_TRIG 功能块而插入之。

v. 在'变量属性'对话框的'名称'域中，输入'Motor_Edge'。

w. 按下'确定'。出现'公共'对话框。如果有必要，输入一个描述，并再次按下'确定'，来插入功能块及其声明。

x. 通过再次点击'编辑向导'来隐藏编辑向导：

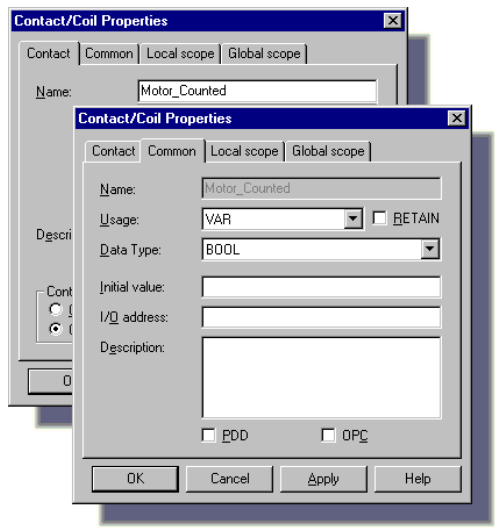
因为在编辑向导中选择功能块时，已经标记了'Motor'触点，所以'Motor_Edge'被直接连接到'Motor'。



y. 双击'C008'触点。

z. 按如下所述声明线圈：

图 68:
'触点/线圈属性'对话框

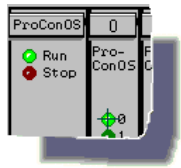


用'制作'图标编译工程，然后，下装它。

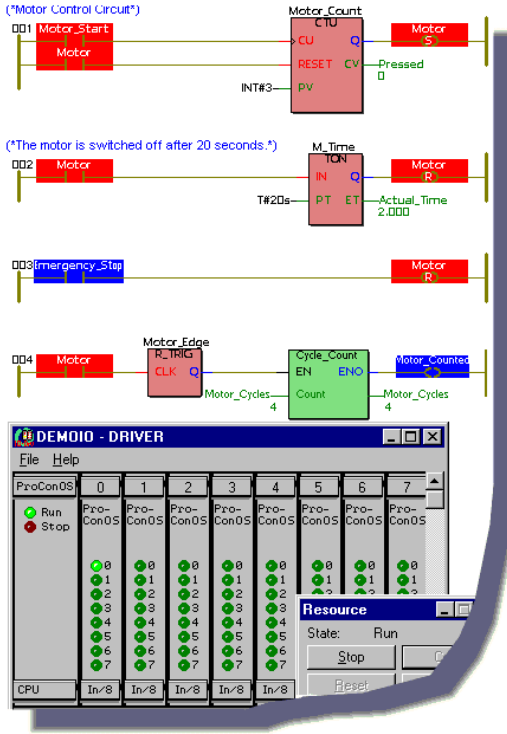
现在我们的示例工程就完成了。您可以用**联机**模式下的工作单和编程系统的 I/O **仿真程序**，检查程序的行为 (见本手册的第 42 页上的**阶段** 5)。

a. 将工作单切换到联机模式，并点击 Windows 任务栏中的 'DEMOIO - DRIVER'按钮，以打开 I/O 仿真程序。

b. 接通和断开模块 0 的位 0 三次，这要通过点击其输入点来完成。



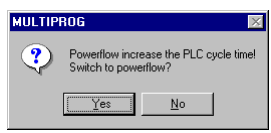
程序执行情况如下：



注意：每当主程序中的逻辑执行一次(马达启动，运转 20 秒，然后停止)，'Motor_Cycles'的值就增加 1。

它可以跳转到用户自定义函数'Cycle_Count'的内部 (即：调用相关的代码本体工作单)，而不离开当前工作单。

a. 双击 LD 工作单中的函数'Cycle_Count'。
出现下列对话框：



b. 用'是'来确认对话框，以便从变量状态切换到能量流。
将打开'Cycle_Count'函数的代码本体工作单。在当前激活了的能量流中，累加器的当前值由符号显示于工作单中。

关于能量流和所用符号的详细信息可以在联机帮助系统中找到。

c. 关闭代码本体工作单，以返回 LD 工作单。

修改任务循环时间

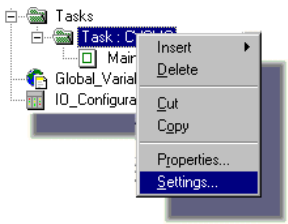
编程系统允许您修改**任务循环时间**，即：循环扫描型任务执行的时间间隔。因此，减少任务循环时间将加速过程的执行。很重要的一点是得到与扫描时间尽可能接近的执行周期。

在我们的例子中，我们将任务循环时间从 100ms 改为 90ms。

可能得到的最短的任务循环时间取决于所使用的 PLC。

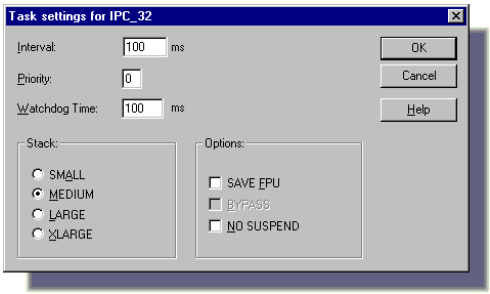
a. 要确保系统处于脱机模式，即：没有按下'调试开/关' 图标。

b. 要想修改任务配置，请用鼠标右键点击'物理硬件'子树中的'TASK : CYCLIC'图标。在出现的上下文菜单中，选择'设置...'。



出现'用于 IPC_32 的任务设置'对话框：

图 69:
用于修改任务循环时间的'IPC_32 任务设置'对话框



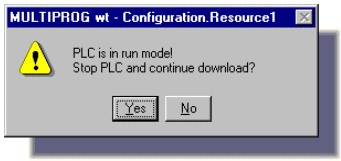
当前任务配置显示于对话框中。我们想要将任务循环时间从 100ms 改为 90ms。

c. 在'时间间隔'域中输入 90，并点击'确定'来确认该对话框。

d. 通过点击'制作'图标来编译工程。关于编译的详细信息，请参考本快速入门手册的第 37 页上的阶段 3。

e. 将工程下装到目标系统，如第 40 页上的阶段 4 中所述。

如果 PLC/仿真程序仍在运行，则在下装前会出现如下消息对话框：



如果是这种情况，请点击'是'继续下装。

f. 如果您喜欢，可以按照本手册的第 42 页上所叙述的**阶段 5** '调试工程'，来调试此工程。

附录

编程系统中的 IEC 工程组件

符合 IEC 61131-3 的编程系统含有下列组件元素：

- 配置
- 资源
- 任务

如果您选择了工程树的'硬件'标签，则将显示这些。

配置可被比作一个可编程控制器系统，例如：一个机架导轨。

资源可被比作一个可以插入到机架导轨中的 CPU。在一个资源中，可以声明仅在该资源内有效的全局变量。在一个资源中，可以执行一个或多个任务。

一般情况下，**任务**决定所关联的程序的时间调度。这意味着程序必须被关联到任务上。任务的设置决定了时间的调度。系统提供了一个将被分配给您的程序的循环扫描任务。

程序组织单元 (POU)

程序组织单元 (POU) 一个 IEC 61131-3 控制系统的语言构造块。它们是包含了程序代码的小的、独立的软件单位。POU 的名称在工程内必须是唯一的。

在 IEC 61131-3 中，支持三种类型的 POU：

- 函数
- 功能块
- 程序

函数是带有多个输入参数和恰好一个输出参数的 POU。调用带有相同值的函数总是返回相同的结果。返回值可以是简单数据类型。在一个函数内，可以调用另外的函数，但不能调用功能块或程序。不允许递归调用。

IEC 61131-3 列出了不同类型的标准函数：

- 类型转换函数，如：ANY_INT_TO_REAL
- 数值函数，如：ABS 和 LOG

- 标准算术运算函数，如：ADD 和 MUL
- 位串函数，如：AND 和 SHL
- 选择和比较函数，如：SEL 和 GE
- 字符串函数，如：RIGHT 和 INSERT
- 时间数据类型函数，如：带有 TIME 数据类型的 SUB ('SUB_T_T')

功能块是带有多个输入/输出参数和内部存储单元的 POU。由一个功能块返回的值取决于其内部存储器的值。在一个功能块内，可以调用另外的功能块或函数。不允许递归调用。

IEC 61131-3 列出了不同类型的标准功能块：

- 边沿检测功能块，如：R_TRIG 和 F_TRIG
- 计数器，如：CTU 和 CTD
- 定时器功能块，如：TON 和 TOF
- 双稳态功能块 SR 和 RS

程序是这样的 POU：它们根据控制器处理的需要，包含了函数和功能块的一个逻辑组合。程序的行为和用途类似于功能块。程序具有内部存储器。程序一定要被关联到任务上。在程序内部，可以调用函数和功能块。不允许递归调用。

POU 和功能块的实例化

根据 IEC 61131-3，一个 FB POU (功能块) 可以在一个工程中被重复使用，这是通过在另一个 POU 中，用一个唯一的名称来调用这个 FB 来进行的。这称为"实例化"。通过调用 FB 实例，FB 代码一定只被定义一次。如果调用了 FB 实例，则 FB 的内部存储器被分配给所调用的实例这允许对不同存储区域的使用。

每个实例具有一个相关的标志符"实例名称"，并含有输入和输出参数以及用于 POU 或 FB 的内部存储器。一个 FB 可以在另一个 FB 或者在某个程序中被实例化。一个 FB 的实例名称必须在将要使用它的程序或 FB 的 VAR 声明中声明。

变量和数据类型

IEC 61131 的另一个强大的特征是对变量的使用，而不用传统 PLC 系统的直接寻址方案。这增加了灵活性，并拓宽了可以在程序中执行的功能的范围。

变量类型

变量必须被首先声明，以便用于逻辑。

当将一个变量插入一个工作单时，您可以声明两种变量类型：

1. 局部变量

2. 全局变量

一个**局部变量**只用于一个 POU，而一个**全局变量**可以用于相应工程的每个 POU。

局部变量是在使用它的那个 POU 的局部变量工作单中声明的。

全局变量必须在一个资源的全局变量声明中，被声明为 'VAR_GLOBAL'，而在每个使用它的 POU 中，被声明为 'VAR_EXTERNAL'。

编程系统提供了程序产生过程中变量及其属性的自动声明，如分配 I/O 地址/逻辑名称。也可以在变量工作单中手工声明变量。

变量地址

您可以用'I/O 地址'输入域直接给您的变量编址。

根据 IEC 61131，定位声明由关键字 AT、百分号"%"、一个位置前缀、一个尺寸前缀和逻辑地址名称组成。
在编程系统，不必输入关键字 AT 和百分号'%'。
一个可能的变量地址例子：'QX0.0'。

下表显示了定位变量的位置和尺寸前缀：

位置前缀	描述
I	物理输入
Q	物理输出
M	PLC 存储器内的物理地址
尺寸前缀	**描述**
X	单个二进制位尺寸(仅用于 BOOL 数据类型)
无	单个二进制位尺寸
B	字节尺寸(8 位)
W	字尺寸(16 位)
D	双字尺寸(32 位)

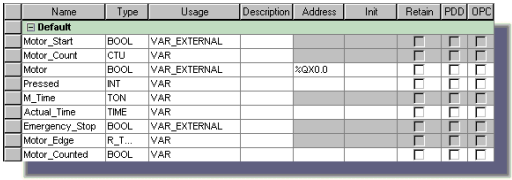
当在系统中声明一个变量时，'变量属性'对话框被自动打开。使用这个对话框，当前变量的声明被自动插入或修改到相应的变量工作单。
局部变量被插入工程树中相应 POU 的变量工作单，全局变量被插入'物理硬件'子树中的全局变量工作单。

也可以通过点击'属性'来调用'变量属性'对话框。

如果您想看一下声明，请点击工具栏中的'变量工作单'图标

来打开 POU 的变量表格工作单 (**局部变量表格工作单**)：

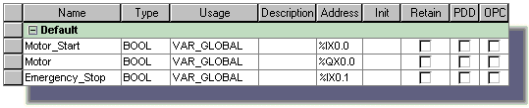
图 70:
局部变量表格工作单



	Name	Type	Usage	Description	Address	Init	Retain	PDD	OPC
	Default								
	Motor_Start	BOOL	VAR_EXTERNAL						
	Motor_Count	CTU	VAR						
	Motor	BOOL	VAR_EXTERNAL		%QX0.0				
	Pressed	INT	VAR						
	M_Time	TON	VAR						
	Actual_Time	TIME	VAR						
	Emergency_Stop	BOOL	VAR_EXTERNAL						
	Motor_Edge	R_T...	VAR						
	Motor_Counted	BOOL	VAR						

或者双击工程树中的'全局变量'，以打开**全局变量表格工作单**：

图 71:
全局变量表格工作单



	Name	Type	Usage	Description	Address	Init	Retain	PDD	OPC
	Default								
	Motor_Start	BOOL	VAR_GLOBAL		%IX0.0				
	Motor	BOOL	VAR_GLOBAL		%QX0.0				
	Emergency_Stop	BOOL	VAR_GLOBAL		%IX0.1				

数据类型

数据类型决定了变量可以具有的值的种类。数据类型定义初始值、可能值的范围和位数。

IEC 61131-3 区分三种数据类型：

- **基本数据类型：**
  IEC 61131-3 中所叙述的基本数据类型的取值范围和尺寸显示于下表：

数据类型	描述	大小	范围
BOOL	布尔	1	0...1
SINT	短整数	8	-128...127
INT	整数	16	-32768 ... 0 ... 32767
DINT	双整数	32	-2,147,483,648 至 2.147.483.647
USINT	无符号短整数	8	0 至 255
UINT	无符号整数	16	0 至 65535
UDINT	无符号双整数	32	0 至 4,294,967,295
REAL	实数	32	+/-1.18 x 10^-38 至 +/-3.40x10^38
TIME	持续时间	32	+# 4.294.976.295 ms 至 +# 4.294.976.295 s
BYTE	长度为 8 的位串	8	0x00...0xFF
STRING	字符序列	80	
WORD	长度为 8 的位串	16	0x0000 ...0xFFFF
DWORD	长度为 8 的位串	32	0x00000000 ... 0xFFFFFFFF

- **类属数据类型：**
  类属数据类型包括多组基本数据类型。它们被称为诸如 ANY_BIT 或者 ANY_INT 等等。
- **用户自定义数据类型：**
  用户自定义数据类型是不同数据类型的组，为了一个特定的目的而被组合在一起，定义为 ARRAY(数组)和 STRUCT(结构)。

Deutschland
KW-Software GmbH
Lagesche Straße 32
32657 Lemgo
Germany
Phone +49 52 61 93 73-0
Fax +49 52 61 93 73-26
www.kw-software.com
info@kw-software.com

Far East
KW-Software
c/o Euro-Far East
Lilas Nogizaka Bldg. 9F
1-15-18 Minami Aoyama
Minato-ku, Tokyo
107-0062 - Japan
Phone +81 3 34 70-87 68
Fax +81 3 34 78-86 48
www.kw-software.com
rch@euro-fareast.co.jp

USA
KW-Software
3536 Edwards Rd.
Cincinnati, OH 45208
USA
Phone +1 51 33 21-93 85
Fax +1 51 33 21-69 92
www.kw-softwareusa.com
info@kw-softwareusa.com