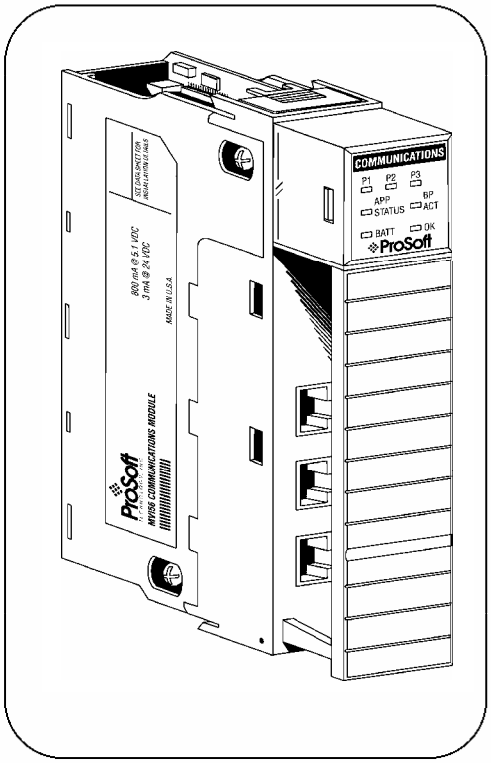
MVI56




# MVI56-MCM

ControlLogix 平台
Modbus 通讯模块

"This is not the latest revision. Please refer to the English version of this manual."

用户手册
August 2003

ProSoft
TECHNOLOGY

## 请阅读以下注意事项

成功的应用这个模块需要对 Allen-Bradley PLC/SLC 硬件知识和现场应用方式有充分的了解。因此，对于负责完成应用的工作人员，了解应用需求并确保人员和设备不处于不安全或不适当的工作环境是非常重要的。

这本手册是用作帮助用户。我们力求提供的每个信息都是准确的，而且如实的反映产品的安装要求。为确保对本产品操作的完全理解，用户必须阅读有关 A-B 硬件操作的所有 Allen-Bradley 应用文档。

在任何条件下，ProSoft Technology, Inc.都不负责间接的或由用户使用或应用本产品而造成的损害。

在没有得到 ProSoft Technology 许可的情况下，禁止任何对本手册内容整体或部分性的复制。

本手册内容如有更改，恕不通知。ProSoft Technology, Inc. 并不承担这个义务。并会随时改进和/或更改此文档或产品。这些更改会阶段性的进行，以更改技术的不准确和印刷排版错误。

**ProSoft Technology, Inc.**
1675 Chester Avenue
Fourth Floor
Bakersfield, CA 93301
(661) 716-5100
(661) 716-5101 Fax
www.prosoft-technology.com


06.MV56.MCM.OO.02.CN
August 2003

# 目录

目录 ..... iii
1 介绍 ..... 1
1.1 基本概念 ..... 1
1.2 设置模块 ..... 1
2 理解结构框架 ..... 3
2.1 主逻辑循环 ..... 4
2.2 ControlLogix处理器未处于运行状态 ..... 4
2.3 背板数据传输 ..... 4
2.4 常规数据传输 ..... 5
2.4.1 读数据块 ..... 6
2.4.2 写数据块 ..... 7
2.5 设置数据传送 ..... 7
2.5.1 模块设置数据 ..... 8
2.6 主站命令数据列表 ..... 9
2.7 从站状态数据块 ..... 10
2.8 命令控制数据块 ..... 12
2.8.1 事件命令 ..... 12
2.8.2 命令控制 ..... 13
2.8.3 写设置 ..... 14
2.8.4 热启动 ..... 15
2.8.5 冷启动 ..... 15
2.9 Pass-Through控制数据块 ..... 15
2.9.1 无格式Pass-Through控制数据块 ..... 15
2.9.2 格式Pass-Through控制数据块 ..... 16
2.9.2.1 功能 5 ..... 16
2.9.2.2 功能 6 和 16 ..... 16
2.9.2.3 功能 15 ..... 17
MVI56-MCM模块和ControlLogix处理器之间的数据流 ..... 18
2.9.3 从站驱动 ..... 18
2.9.4 主站驱动模式 ..... 20
2.9.4.1 主站命令列表 ..... 21
3 修改模块设置 ..... 23
3.1 上电 ..... 23
3.2 运行中更改参数 ..... 23
3.3 装配模块 ..... 23
3.4 模块数据对象 (MCMModuleDef) ..... 29
3.4.1 设置对象 ..... 30
3.4.1.1 数据传输参数 (MCMModule) ..... 31

3.4.1.2 Modbus端口参数 (MCMPort) ..... 32
3.4.1.3 Modbus 主站命令 (MCMCmd) ..... 33
3.4.2 状态对象 (MCMInStat) ..... 34
3.5 用户数据对象 ..... 34
3.6 从站轮询控制和状态 ..... 35
3.7 Modbus 讯息数据 ..... 35
4 修改样例梯形逻辑程序 ..... 37
4.1 上电程序（Power Up） ..... 37
4.2 主程序（MainRoutine） ..... 38
4.3 读数据程序（ReadData） ..... 38
4.4 写数据程序（WriteData） ..... 44
5 诊断和纠错 ..... 51
5.1 从模块读取状态数据 ..... 51
5.1.1 硬件要求 ..... 51
5.1.2 软件要求 ..... 52
5.1.3 端口使用 ..... 52
5.1.4 菜单选项 ..... 53
5.1.4.1 A=数据分析 ..... 53
5.1.4.1.1 1=选择端口 1 ..... 53
5.1.4.1.2 2=选择端口 2 ..... 54
5.1.4.1.3 5=1 mSec 标记 ..... 54
5.1.4.1.4 6=5 mSec标记 ..... 54
5.1.4.1.5 7=10 mSec标记 ..... 54
5.1.4.1.6 8=50 mSec标记 ..... 54
5.1.4.1.7 9=100 mSec标记 ..... 54
5.1.4.1.8 0=No mSec标记 ..... 54
5.1.4.1.9 H=Hex 格式 ..... 54
5.1.4.1.10 A=ASCII 格式 ..... 54
5.1.4.1.11 B=开始 ..... 54
5.1.4.1.12 S=停止 ..... 55
5.1.4.1.13 M = 主菜单 ..... 55
5.1.4.2 B=块传输统计 ..... 55
5.1.4.3 C=模块设置 ..... 55
5.1.4.4 D=查看Modbus数据库 ..... 56
5.1.4.5 0-9 寄存器页码代表 0-9000 ..... 56
5.1.4.6 S=再次显示 ..... 56
5.1.4.6.1 - = 回退 5 页 ..... 57
5.1.4.6.2 P = 前页 ..... 57
5.1.4.6.3 + = 跳过 5 页 ..... 57
5.1.4.6.4 N = 下页 ..... 57
5.1.4.6.5 D = 十进制显示 ..... 57

5.1.4.6.6 H = 十六进制显示 ..... 57
5.1.4.6.7 F = 浮点数显示 ..... 58
5.1.4.6.8 A = ASCII 显示 ..... 58
5.1.4.6.9 M = 主菜单 ..... 58
5.1.4.7 E 和 F=主站命令错误（端口 1 和 2） ..... 58
5.1.4.7.1 S = 再次显示 ..... 58
5.1.4.7.2 - = 回退 2 页 ..... 58
5.1.4.7.3 P = 前页 ..... 58
5.1.4.7.4 + = 跳过 2 页 ..... 59
5.1.4.7.5 N = 下页 ..... 59
5.1.4.7.6 D = 十进制显示 ..... 59
5.1.4.7.7 H = 十六进制显示 ..... 59
5.1.4.7.8 M = 主菜单 ..... 59
5.1.4.8 I 和 J=主站命令列表（端口 1 和 2） ..... 59
5.1.4.8.1 S = 再次显示 ..... 59
5.1.4.8.2 - = 回退 5 页 ..... 59
5.1.4.8.3 P = 前页 ..... 59
5.1.4.8.4 + = 跳过 5 页 ..... 61
5.1.4.8.5 N = 下页 ..... 61
5.1.4.8.6 M = 主菜单 ..... 61
5.1.4.9 O and P=从站状态列表（端口 1 和 2） ..... 61
5.1.4.10 V=版本信息 ..... 61
5.1.4.11 W=热启动模块 ..... 62
5.1.4.12 Y=传送模块配置到处理器 ..... 62
5.1.4.13 1 and 2=通讯状态（端口 1 和 2） ..... 63
5.1.4.14 6 and 7=端口设置（端口 1 和 2） ..... 63
5.1.4.15 Esc=退出程序 ..... 63
5.2 LED 状态指示 ..... 63
5.2.1 清除故障状态 ..... 66
5.2.2 纠错 ..... 67
6 电缆连接 ..... 69
6.1 Modbus通讯端口 ..... 69
6.1.1 连接电缆到连接器 ..... 69
6.1.1.1 RS-232 ..... 70
6.1.1.2 RS-485 ..... 70
6.1.1.3 RS-422 ..... 70
6.2 RS-232 设置/调试端口 ..... 71
附录 A - MVI56-MCM 数据库定义 ..... 73
附录B – MVI56-MCM 状态数据定义 ..... 75
附录C – MVI56-MCM 设置数据定义 ..... 77
背板设置 ..... 77

端口 1 设置 ..... 78
端口 2 设置 ..... 80
端口 1 命令 ..... 82
端口 2 命令 ..... 82
各种状态 ..... 82
命令控制 ..... 84
附录 D – MVI56-MCM命令控制 ..... 85
附录 E – 产品规格 ..... 87
总体规格 ..... 87
从站功能规格 ..... 88
Modbus主站功能规格 ..... 88
外形 ..... 88
ControlLogix 接口 ..... 88
硬件规格 ..... 88
支持，服务和保证 ..... 91

# 1 介绍

MVI56-MCM (“Modbus 通讯模块”) 产品可以让 Allen-Bradley ControlLogix I/O 兼容处理器轻松的和其它 Modbus 协议兼容设备取得通讯。兼容的设备不仅包括 Modicon PLC (都支持 Modbus 协议)，还包括类别广阔的终端设备。

MVI56-MCM 模块是 Modbus 网络和 Allen-Bradley backplane 之间的一个网关。来自于 ControlLogix 处理器的数据传送异步于 Modbus 网络上的数据活动。模块内部具有 5000 个字的寄存器，用于处理器和 Modbus 网络之间的数据交换。

## 1.1 基本概念

以下讨论包括一些概念，这些概念对于理解 MVI56-MCM 模块的运作是十分重要的。

上电时，模块开始执行以下逻辑功能：

1. 初始化硬件组件
    a. 初始化 ControlLogix 背板驱动
    b. 测试并复位所有 RAM
    c. 初始化串行通讯端口
2. 等待来自于 ControlLogix 处理器的模块配置
3. 初始化模块寄存器空间
4. 在所选端口上，启动从站驱动
5. 在所选端口上，启动主站驱动

当模块已经接收来自处理器的模块配置后，模块会根据配置和网络上的其它节点开始通讯。

## 1.2 设置模块

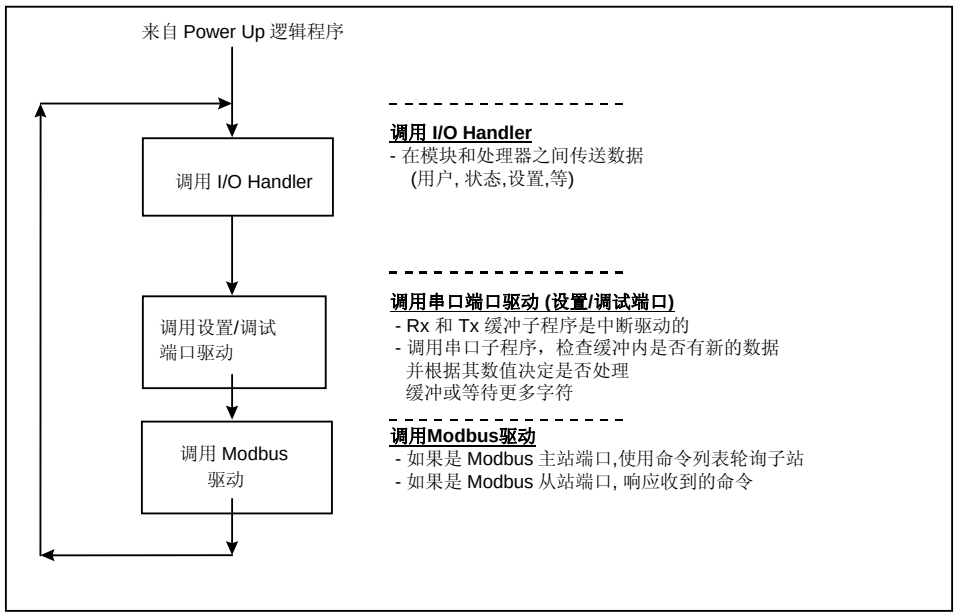
在模块安装完毕后，您可以开始修改模块配置和梯形逻辑程序的工作。在开始这项工作前，您应该了解在下章节提到的框架结构。余下的章节会解释如何修改现存的.cfg 文件和样例梯形逻辑程序。

# 2 理解框架结构

本章节给读者关于 MVI56-MCM 模块一个功能概观。内容并不覆盖有关梯形逻辑程序和内存映射的详细情况(请参阅**模块设置** 部分)。成功的在用户应用中使用这个模块，就需要彻底理解本文档所包含的信息。如果您已经了解本部分的内容，就请参阅**模块设置**部分设置并运行模块。如果您不熟悉数据传输和 Modbus 协议操作，就请在设置模块前阅读本章节内容。

## 2.1 主逻辑循环

在完成上电设置过程后，模块就进入一个无限循环，并执行以下功能：




## 2.2 ControlLogix 处理器未处于运行状态

在任何时候，如果模块检测到处理器不再处于运行模式 (比如故障或编程状态)，会根据用户事先的定义来关闭 Modbus 端口。当处理器回到运行状态，模块会自动恢复网络上的通讯。

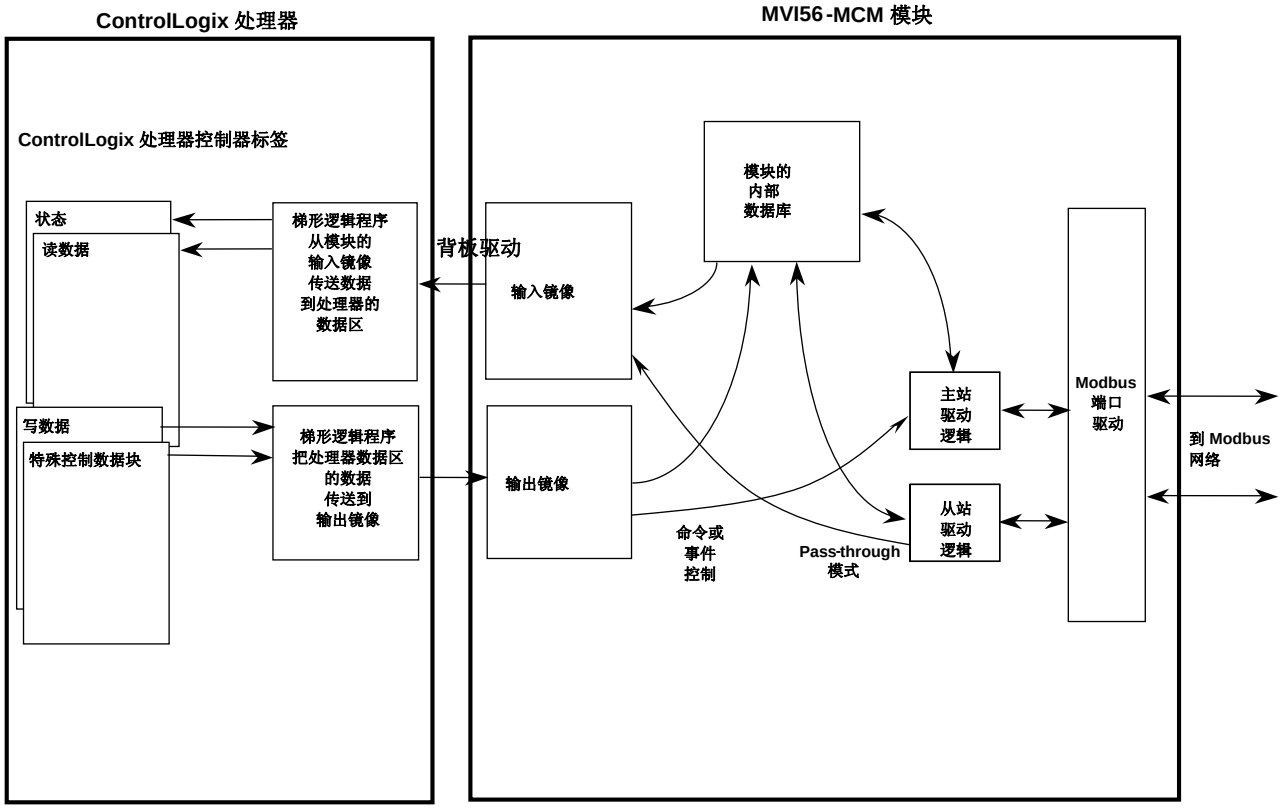
## 2.3 背板数据传输

MVI56-MCM 模块对 ControlLogix 背板使用方式是非常独特的。通过背板，模块使用自身的输入和输出镜像以分页的方式，接收数据和发送数据到处理器。镜像刷新的频率取决于用户为模块定义的计划扫描频率（scheduled scan rate）和模块的通讯负荷。典型的刷新频率在 2 到 10 毫秒之间。

双向的数据传送是这样实现的，模块在其输入镜像中填充数据，再发送到处理器。梯形逻辑程序把输入镜像中的数据放置在处理器的控制器标签（Controller Tags）中。模块的输入镜像是 250 个字。这个大型数据区让数据在模块和处理器之间快速的交换。

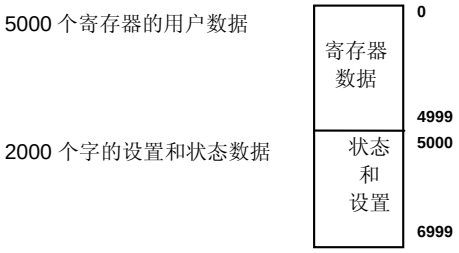
需要传送到模块的数据被处理器插放到模块的输出镜像。模块程序会把这些数据提取出来，放置在模块的内部数据库。模块的输出镜像共有 248 个字。这个大型数据区让数据在模块和处理器之间快速的交换。

下面的示意图描述了在 ControlLogix 处理器，MVI56-MCM 模块和 Modbus 网络之间数据移动的方式。




正如示意图中显示的那样，模块和处理器之间通过背板的数据交换是通过输入和输出镜像来实现的。在 ControlLogix 处理器中的梯形逻辑程序需要把控制器标签（Controller Tags）数据和输入输出镜像数据连接起来。模块使用的所有数据都存储在其内部数据库中。这个数据库也是一个虚拟 Modbus 数据表，地址从 0（40001 Modbus）到 6999（47000 Modbus）。以下是数据库布局示意图：

**模块内部数据库结构**




ControlLogix 梯形逻辑程序和模块程序共同协调工作，把数据库中的数据以分页的方式和输入和输出镜像做交换。从模块到处理器单次可传送 248 个字。从处理器到模块单次可传送 247 个字。每个镜像都具有定义的结构，这取决于数据内容和数据传送的功能。下面做详细说明。

## 2.4 常规数据传输

常规数据传输包括在模块内部数据库，0 到 4999 内发现的用户数据及状态数据传输。这些数据传输通过读（输入镜像）和写（输出镜像）数据块来实现。数据块中使用的数据对象

的定义和需要的梯形逻辑程序，请参阅**模块设置**部分。下面将介绍每个数据块的结构和功能。

### 2.4.1 读数据块

这些数据块把信息从模块传送到 ControlLogix 处理器。用于传输这些数据的输入镜像结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 读数据块 ID | 1 |
| 2 – 201 | 读数据 | 200 |
| 202 | 程序扫描计数器 | 1 |
| 203 – 204 | 产品编码 | 2 |
| 205 – 206 | 产品版本 | 2 |
| 207 – 208 | 操作系统 | 2 |
| 209 – 210 | 运行数 | 2 |
| 211 – 217 | 端口 1 错误状态 | 7 |
| 218 – 224 | 端口 2 错误状态 | 7 |
| 225 – 230 | 数据传输状态 | 6 |
| 231 | 端口 1 当前错误/索引 | 1 |
| 232 | 端口 1 最后错误/索引 | 1 |
| 233 | 端口 2 当前错误/索引 | 1 |
| 234 | 端口 2 最后错误/索引 | 1 |
| 235 – 248 | 空 | 14 |
| 249 | 读数据块 ID | 1 |

读数据块 ID 作为索引，来决定存放在 ControlLogix 处理器控制器标签中，模块读数据矩阵的位置。每次传输能够移动 200 个字（数据块偏移 2 到 201）的数据。除了移动用户数据外，数据块还包含模块的状态数据。最后部分数据通过每个新的数据块传送，这用于高速数据移动。

数据块关联的写数据块 ID 用来从 ControlLogix 处理器请求数据传送。在正常程序运行下，模块会序列地发送读数据块并请求写数据块。例如，如果应用中使用了三个读数据块和两个写数据块，那么顺序就如下所示：

R1W1-->R2W2-->R3W1-->R1W2-->R2W1-->R3W2-->R1W1-->

这个顺序会一直运行直到被处理器发送的其它写数据块号码打断，或是来自于 Modbus 网络某个节点的命令请求，或是经由模块设置/调试端口的操作员控制。

### 2.4.2 写数据块

这些数据块把信息从 ControlLogix 处理器传送到模块。用于传输这些数据的输出镜像结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 写数据块 ID | 1 |
| 1 – 200 | 写数据 | 200 |
| 201 – 247 | 空 | 47 |

写数据块 ID 作为索引来决定存放到模块数据库的位置。每次传送能移动 200 个字（数据块偏移 1 到 200）的数据。

## 2.5 设置数据传送

当模块执行一次重启的操作时，它会从 ControlLogix 处理器请求设置信息。这些数据以特殊格式写数据块（输出镜像）的方式传送。 模块会在读数据块（输入镜像）中设置需要的写数据块号码，来询问每个数据块。数据块中使用的数据对象的定义和需要的梯形逻辑程序，请参阅**模块设置**部分。设置数据块的格式在下面的内容中介绍。

### 2.5.1 模块设置数据

这个数据块被用于将设置信息从处理器传送到模块。数据包含在以 9000 为 ID 号的数据块。块结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 9000 | 1 |
| 1 – 6 | 背板设置 | 6 |
| 7 – 31 | 端口 1 设置 | 25 |
| 32 – 56 | 端口 2 设置 | 25 |
| 57 – 59 | 端口 1 辅助设置 | 3 |
| 60 – 62 | 端口 2 辅助设置 | 3 |
| 63 – 247 | 空 | 185 |

用做请求设置的读数据块结构是这样的结构：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 9000 | 1 |
| 2 | 模块设置错误 | 1 |
| 3 | 端口 1 设置错误 | 1 |
| 4 | 端口 2 设置错误 | 1 |
| 5 – 248 | 空 | 244 |
| 249 | -2 或 –3 | 1 |

如果设置中存在任何错误，那么在 3 个设置错误字中相关的那个位就会被置位。在模块开始正常工作前，必须纠正错误。

## 2.6 主站命令数据列表

模块的每个端口都可以设置成 Modbus 主站设备，每个 Modbus 主站都有自己的 100 条指令。命令从处理器中读取，使用如下写数据块 ID：Modbus 端口 1 – 6000 到 6003，Modbus 端口 2 – 6100 到 6103。模块会按顺序的向处理器轮询每个数据块。梯形逻辑程序要能够处理每次数据传送。数据块的结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 6000 到 6003 ， 6100 到 6103 | 1 |
| 1 – 8 | 命令定义 | 8 |
| 9 – 16 | 命令定义 | 8 |
| 17 – 24 | 命令定义 | 8 |
| 25 – 32 | 命令定义 | 8 |
| 33 – 40 | 命令定义 | 8 |
| 41 – 48 | 命令定义 | 8 |
| 49 – 56 | 命令定义 | 8 |
| 57 – 64 | 命令定义 | 8 |
| 65 – 72 | 命令定义 | 8 |
| 73 – 80 | 命令定义 | 8 |
| 81 – 88 | 命令定义 | 8 |
| 89 – 96 | 命令定义 | 8 |
| 97 – 104 | 命令定义 | 8 |
| 105 – 112 | 命令定义 | 8 |
| 113 – 120 | 命令定义 | 8 |
| 121 – 128 | 命令定义 | 8 |
| 129 – 136 | 命令定义 | 8 |
| 137 – 144 | 命令定义 | 8 |
| 145 – 152 | 命令定义 | 8 |
| 153 – 160 | 命令定义 | 8 |
| 161 – 168 | 命令定义 | 8 |
| 169 – 176 | 命令定义 | 8 |
| 177 – 184 | 命令定义 | 8 |
| 185 – 192 | 命令定义 | 8 |
| 193 – 200 | 命令定义 | 8 |

## 2.7 从站状态数据块

从站状态数据块负责传送每个主站端口上的从站状态信息。主站端口上的从站可以有以下几种状态：

| | |
|---|---|
| 0 | 从站未处于活动状态，并且没有定义在主站命令列表中。 |
| 1 | 从站处于活动状态，被主站轮询或由主站端口控制。通讯成功。 |
| 2 | 主站端口和从站设备通讯失败。按照用户事先定义， 根据命令列表的扫描，与该从站的通讯被挂起。 |
| 3 | 与从站的通讯被梯形逻辑程序禁止。通讯不会开始直到梯形逻辑程序清除禁止状态。 |

当模块初始化主站命令列表时，系统定义所有从站。在这个初始步骤中，每个从站定义成状态 1。如果主站端口不能和某个从站建立通讯 （命令重试达到重试次数），主站会在状态表中把这个从站的状态设为 2。和这个从站的通讯也会挂起，并按用户定义的扫描计数开始计数（每个端口 **MCMPort** 对象中的 **ErrorDelayCntr** 值）。每次扫描到和通讯挂起的从站相关联的命令时，延迟计数器计数会减少。当这个数值达到 0 时，这个从站的状态会设置成 1。这就能够再次轮询这个从站。

| 数据块 ID | 描述 |
|---|---|
| 3002 | 请求端口 1 上的前 128 个从站状态数值 |
| 3003 | 请求端口 1 上的后 128 个从站状态数值 |
| 3102 | 请求端口 2 上的前 128 个从站状态数值 |
| 3103 | 请求端口 2 上的后 128 个从站状态数值 |

这些数据块的格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 3002 – 3003 或 3102 – 3103 | 1 |
| 1 – 247 | 空 | 246 |

接受到特殊写数据块编码时，模块会识别这个请求，并以一个读数据块来返回。这个读数据块的格式如下所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 写数据块 ID | 1 |
| 2 – 129 | 从站轮询状态数据 | 128 |
| 130 – 248 | 空 | 119 |
| 249 | 3002 – 3003 或 3102 – 3103 | 1 |

可以编写梯形逻辑程序来覆盖从站状态表中的数据。可以从处理器向从站发送特殊数据块来实现禁止命令（状态数值 3）。使用数据块 3000 来禁止端口 1 的从站，数据块 3100 来禁止端口 2 的从站。每个数据块包含禁止用的从站节点地址。数据块的结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 3000 或 3100 | 1 |
| 1 | 数据块中从站的数目 | 1 |
| 2 – 201 | 从站索引 | 200 |
| 202 – 247 | 空 | 46 |

模块会返回一个和接收到的数据块识别号一致的数据块，并指明根据数据块动作的从站数目。返回的数据块格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 写数据块 ID | 1 |
| 2 | 处理从站的数目 | 1 |
| 3 – 248 | 空 | 246 |
| 249 | 3000 或 3100 | 1 |

可以编写梯形逻辑程序覆盖从站状态表中的数据，发送特殊数据块来启动从站（状态数值 1）。端口 1 的从站可以使用数据块 3001 来使能，端口 2 的从站使用数据块 3101 来使能。每个数据块包含用来启动的从站节点地址。该数据块的格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 3001 或 3101 | 1 |
| 1 | 数据块中的从站数目 | 1 |
| 2 – 201 | 从站索引 | 200 |
| 202 – 247 | 空 | 46 |

模块会返回一个和接收到的数据块识别号一致的数据块，并指明根据数据块动作的从站数目。返回的数据块格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 写数据块 ID | 1 |
| 2 | 处理从站的数目 | 1 |
| 3 – 248 | 空 | 246 |
| 249 | 3001 或 3101 | 1 |

## 2.8 命令控制数据块

命令控制数据块是一种特殊的数据块，它可以控制模块或向模块请求特殊的数据。目前版本的软件支持 5 个命令控制数据块：事件命令控制，命令控制，写设置，热启动和冷启动。

### 2.8.1 事件命令

事件命令控制数据块可以直接从梯形逻辑程序发送 Modbus 命令到任何一个主站端口。这些数据块的格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 1000 – 1255 或 2000 – 2255 | 1 |
| 1 | 内部数据库地址（Internal DB Address） | 1 |
| 2 | 点数（Point Count） | 1 |
| 3 | 交换代码（Swap Code） | 1 |
| 4 | Modbus 功能代码（Modbus Function Code） | 1 |
| 5 | 设备数据库地址（Device Database Address） | 1 |
| 6 – 247 | 空 | 242 |

数据块号码定义了使用的 Modbus 端口和涉及到的从站节点。在地址 1000 范围内的数据块指向 Modbus 端口 1，地址 2000 范围内的数据块指向 Modbus 端口 2。数据块号码，0 到 255，代表从站地址。这两个数值的和决定了数据块号码。随模块传递的其它参数用作组建命令。**Internal DB Address** 参数指与命令关联的模块数据库地址。**Point Count** 参数定义了命令处理的点数或寄存器个数。**Swap Code** 是使用 Modbus 功能 3 时更改字或字节次序。**Modbus Function Code** 可以是以下数值 1, 2, 3, 4, 5, 6, 15 或 16。**Device Database Address** 是命令关联的远程从站设备中的 Modbus 寄存器或点。当命令接收到这个数据块时，模块会处理并把它放到命令队列中。模块对应每个事件命令数据块，都会返回一个具有下列格式的数据块：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 写数据块 ID | 1 |
| 2 | 0 = 失败, 1 = 成功 | 1 |
| 3 – 248 | 空 | 246 |
| 249 | 1000 – 1255 或 2000 - 2255 | 1 |

数据块的第 2 个字可以被梯形逻辑程序利用来判断命令是否已经放置在模块的命令队列中。命令只有在端口的命令队列排满时（每个队列 100 条命令）才会失败。

### 2.8.2 命令控制

命令控制数据块负责把命令列表中的命令放到命令队列中。每个端口有一个多达 100 条指令的命令队列。模块对队列中命令的服务要优先于主站命令列表。这就是说队列中的命令有更高的优先级。通过这种机制放置在队列中的命令，需要事先在主站命令列表中定义。在命令列表正常执行的情况下，模块会只执行那些使能参数是 1 或 2 的命令。如果是 0，就会跳过这个命令。在命令列表中可以放置使能参数为 0 的命令。而这些命令就可以通过命令控制数据块来之行。

单个请求可以放置 1 到 6 个命令到命令队列中。数据块的格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 5001 – 5006 或 5101 – 5106 | 1 |
| 1 | 命令索引 (MCM.P1.CMD [*命令索引/数值*]) | 1 |
| 2 | 命令索引 (MCM.P1.CMD [*命令索引/数值*]) | 1 |
| 3 | 命令索引 (MCM.P1.CMD [*命令索引/数值*]) | 1 |
| 4 | 命令索引 (MCM.P1.CMD [*命令索引/数值*]) | 1 |
| 5 | 命令索引 (MCM.P1.CMD [*命令索引/数值*]) | 1 |
| 6 | 命令索引 (MCM.P1.CMD [*命令索引/数值*]) | 1 |
| 7 – 247 | 空 | 241 |

地址 5001 到 5006 的数据块供 Modbus 端口 1 使用，地址 5101 到 5106 的数据块供 Modbus 端口 2 使用。数据块编码的最后一个数字表明数据块处理命令的数目。比如，编码为 5003 的数据块包含 3 个命令索引用于 Modbus 端口 1。数据块中的命令索引参数的范围是 0 到 99，对应于主站命令列表中的输入。

对应于一个命令控制数据块，模块会返回一个数据块，其中包含加入到端口命令队列中的命令数目。数据块格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 写数据块 ID | 1 |
| 2 | 加到命令队列中的命令数目 | 1 |
| 3 – 248 | 空 | 246 |
| 249 | 5000 – 5006 或 5100 - 5106 | 1 |

### 2.8.3 写设置

ControlLogix 处理器发送这个数据块到模块，强制模块把其当前的设置返回写到处理器。这个功能应用在当模块的设置通过数据库写操作被远程更改时，同步设置数据。这个数据块的第一个字的数值是-9000。模块会返回一个包含模块设置数据的数据块。编写的梯形逻辑程序要能处理接收这些数据块。这个数据块有如下格式：

**数据块 -9000，常规设置数据：**

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | -9000 | 1 |
| 2 – 7 | 背板设置 | 6 |
| 8 – 32 | 端口 1 设置 | 25 |
| 33 – 57 | 端口 2 设置 | 25 |
| 58 – 60 | 端口 1 辅助设置 | 3 |
| 61 – 63 | 端口 2 辅助设置 | 3 |
| 64 – 248 | 空 | 185 |
| 249 | -9000 | 1 |

相对应，数据块 -6000 到 -6003 和 -6100 到 6103 就是端口 1 和 2 的主站命令列表数据：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | -6000 到 –6003 和 –6100 到 –6103 | 1 |
| 2 – 9 | 命令定义 | 8 |
| 10 –17 | 命令定义 | 8 |
| 18 – 25 | 命令定义 | 8 |
| 26 – 33 | 命令定义 | 8 |
| 34 – 41 | 命令定义 | 8 |
| 42 – 49 | 命令定义 | 8 |
| 50 – 57 | 命令定义 | 8 |
| 58 – 65 | 命令定义 | 8 |
| 66 – 73 | 命令定义 | 8 |
| 74 – 81 | 命令定义 | 8 |
| 82 – 89 | 命令定义 | 8 |
| 90 – 97 | 命令定义 | 8 |
| 98 – 105 | 命令定义 | 8 |
| 106 – 113 | 命令定义 | 8 |
| 114 – 121 | 命令定义 | 8 |
| 122 – 129 | 命令定义 | 8 |
| 130 – 137 | 命令定义 | 8 |
| 138 – 145 | 命令定义 | 8 |
| 146 – 153 | 命令定义 | 8 |
| 154 – 161 | 命令定义 | 8 |
| 162 – 169 | 命令定义 | 8 |
| 170 – 177 | 命令定义 | 8 |
| 178 – 185 | 命令定义 | 8 |

| | | |
|---|---|---|
| 186 – 193 | 命令定义 | 8 |
| 194 – 201 | 命令定义 | 8 |
| 202 – 248 | 空 | 47 |
| 249 | -6000 到 –6003 和 –6100 到 –6103 | 1 |

为保证模块正常运作，梯形逻辑程序要能处理这些数据块。

### 2.8.4 热启动

当模块需要执行一个热启动（软件重置）的操作时，ControlLogix 处理器可以发送这个数据块到模块（输出镜像）。这个数据块经常用于当控制器标签数据中的设置数据更改的时候。这会强制模块读取新的设置信息并重启动。控制数据块的结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 9998 | 1 |
| 1 – 247 | 空 | 247 |

### 2.8.5 冷启动

当模块需要执行一个冷启动（硬件重置）的操作时，ControlLogix 处理器可以发送这个数据块到模块（输出镜像）。这个数据块用于当梯形逻辑程序检测到硬件问题，并需要硬件重置时。控制数据块的结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 9999 | 1 |
| 1 – 247 | 空 | 247 |

## 2.9 Pass-Through 控制数据块

### 2.9.1 无格式 Pass-Through 控制数据块

如果模块的一个或多个从站端口设置成无格式 pass-through 模式运行，每次收到写命令时，模块会传送 ID 号为 9996 的数据块到处理器。使用这个 ID 号，任何 Modbus 功能 5，6，15 和 16 的命令都会从端口发送到处理器。梯形逻辑程序要能够处理接收所有到达处理器的 Modbus 写功能，并按照远程 Modbus 主站设备需要的内容响应数据。无格式 pass-through 控制数据块的格式如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 9996 | 1 |
| 2 | Modbus 讯息中的字节数 | 1 |
| 3 – 248 | 接收到的 Modbus 讯息 | 246 |
| 249 | 9996 | 1 |

梯形逻辑程序必须复制解析接收到的讯息，并按照主站设备的需要控制处理器。处理器必须响应这些 pass-through 控制数据块，格式如下：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 9996 | 1 |
| 1 – 247 | 空 | 247 |

这会通知模块命令已经被执行了，并可以从 pass-through 列表中清除。

## 2.9.2 格式 Pass-Through 控制数据块

如果模块的一个或多个从站端口设置成格式 pass-through 模式运行，每次收到写命令时，模块会传送 ID 号为 9996 的数据块到处理器。使用这个 ID 号，任何 Modbus 功能 5，6，15 和 16 都会从端口发送到处理器。梯形逻辑程序要能够处理接收所有到达处理器的 Modbus 写功能，并按照远程 Modbus 主站设备需要的内容响应数据。无格式 pass-through 控制数据块的格式如下表所示：

### 2.9.2.1 功能 5

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 9958 | 1 |
| 2 | 1 | 1 |
| 3 | 位地址 | 1 |
| 4 | 数据 | 1 |
| 5 – 248 | 接收到的 Modbus 讯息 | 244 |
| 249 | 9958 | 1 |

梯形逻辑程序必须复制解析接收到的讯息，并按照主站设备的需要控制处理器。处理器必须响应这些 pass-through 控制数据块，格式如下：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 9958 | 1 |
| 1 – 247 | Spare | 247 |

这会通知模块命令已经被执行了，并可以从 pass-through 列表中清除。

### 2.9.2.2 功能 6 和 16

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 9956/9957 （浮点数） | 1 |
| 2 | 数据字的数目 | 1 |
| 3 | 数据地址 | 1 |
| 4 – 248 | 数据 | 244 |
| 249 | 9956/9957 | 1 |

梯形逻辑程序必须复制解析接收到的讯息，并按照主站设备的需要控制处理器。处理器必须响应这些 pass-through 控制数据块，格式如下：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 9956/9957 | 1 |
| 1 – 247 | 空 | 247 |

这会通知模块命令已经被执行了，并可以从 pass-through 列表中清除。

### 2.9.2.3 功能 15

在 pass-through 模式中，当模块收到功能代码 15 时，模块会使用数据块 ID 9959 的多位数据来写数据。首先，使用位屏蔽来清除需要更新的位。这可以通过用现存数据和反相位的屏蔽数据求“与”来实现。然后，用结果数和现存数求“或”。这就避免了 INT 寄存器中其他的位不受影响。.

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 9959 | 1 |
| 2 | 字的数目 | 1 |
| 3 | 字地址 | 1 |
| 4 – 53 | 数据 | 50 |
| 54 – 103 | 屏蔽数 | 50 |
| 104 – 248 | 空 | 145 |
| 249 | 9959 | 1 |

梯形逻辑程序必须复制解析接收到的讯息，并按照主站设备的需要控制处理器。处理器必须响应这些 pass-through 控制数据块，格式如下：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 9959 | 1 |
| 1 – 247 | 空 | 247 |

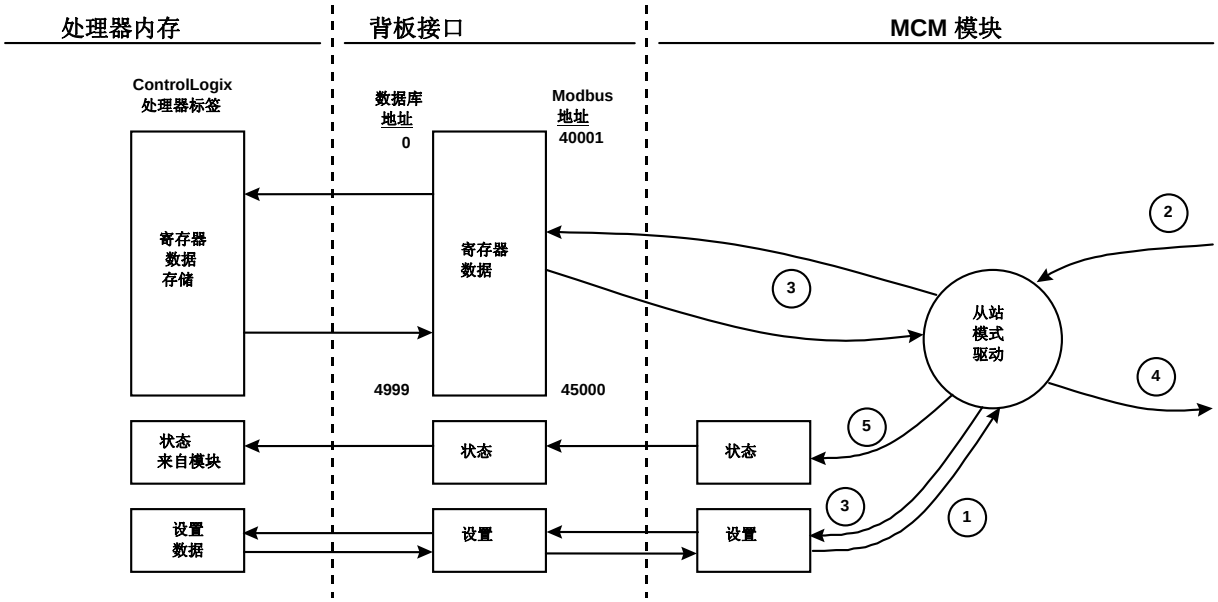
这会通知模块命令已经被执行了，并可以从 pass-through 列表中清除。

# MVI56-MCM 模块和 ControlLogix 处理器之间的数据流

下面讨论在模块不同运作模式下，两块硬件（ControlLogix 处理器和 MVI56-MCM 模块）和 Modbus 网络其他节点之间的数据流动。模块上的每个端口都可以设置模拟成一个 Modbus 主站设备或 Modbus 从站设备。每个端口的运作取决于端口的设置。下面讨论每种模式的运行。

## 2.9.3 从站驱动

当 MVI56-MCM 模块处于从站驱动模式时，模块可以响应来自于 Modbus 网络上主站设备发出的数据读和写命令。下面通过流向图并结合表格来详细说明进出模块的数据流动。

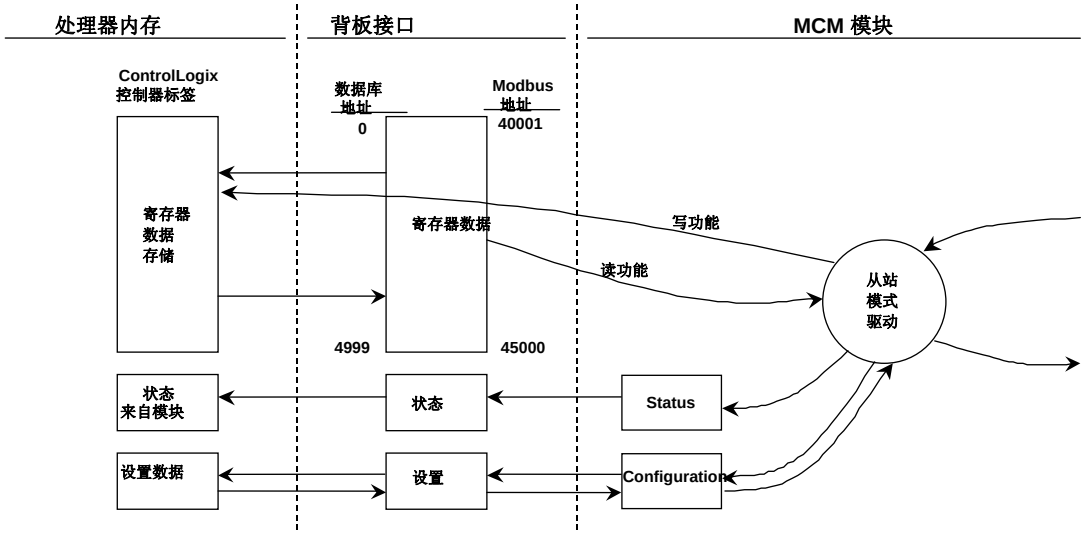



| 步骤 | 描述 |
|---|---|
| 1 | Modbus 从站端口驱动从 ControlLogix 处理器接收设置信息。用这些信息设置串口，并且定义从站节点特征。另外，设置信息还包含在数据库中做数据地址偏移的数据，这些地址是从主站接收到的通讯信息需要的。 |
| 2 | 某个主站设备，比如 Modicon PLC 或 MMI 软件包，向模块节点地址发送读或写命令。端口驱动在接受它之前会先验证。 |
| 3 | 一旦命令被模块接受，数据会立即传输到或从模块的内部数据库。如果命令是读命令，数据就从数据库读出并且创建响应讯息。如果命令是写命令，数据会直接写到数据库里并且创建响应讯息。 |
| 4 | 当步骤 2 中的数据处理过程结束，模块会响应讯息到主站节点。 |
| 5 | 在状态数据块中包含的计数器使梯形逻辑程序能够判断从站驱动的活动程度。 |

在**模块设置** 章节可以查看到所有定义从站端口时所需要的完整参数列表。

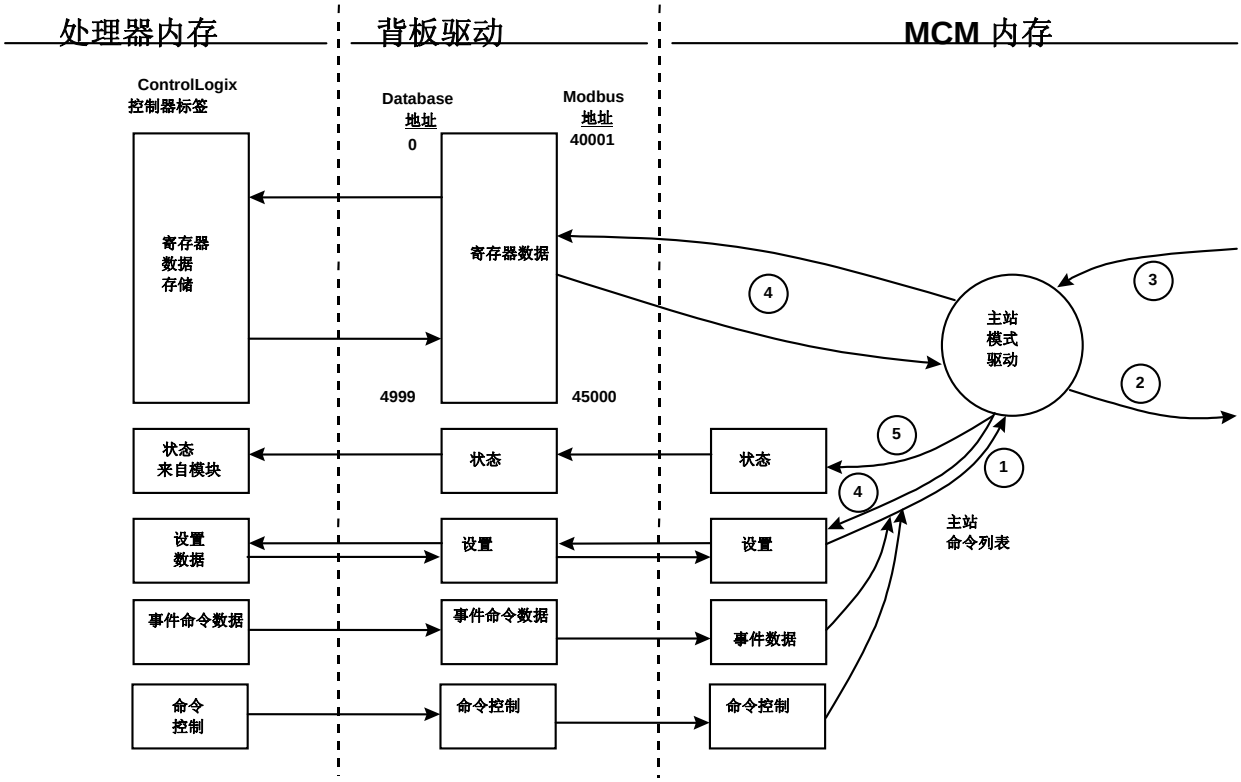
在这种常规运行模式下有一种例外情况，就是需要执行 pass-through 模式。 在这种模式下，所有写请求会直接传递到处理器，而不是放到数据库里。这就能在没有中间数据库的条件下直接的，远程控制处理器。这种模式对于不需要发送两种状态控制的主站设备非常有用处。比如，一个 SCADA 系统也许只会对数字控制点发送启动命令，而不会发送清除

状态的命令。SCADA 系统需要本地逻辑程序来清除控制位。这种模式必须通过 Pass-through 方式来模拟。下面的图表阐明了 pass-through 功能开启下从站端口的数据流：



### 2.9.4 主站驱动模式

在主站运行模式下，MVI56-MCM 模块负责发送读或写命令到 Modbus 网络上的从站节点。用户在模块的主站命令列表种设置这些命令，这些命令从 ControlLogix 处理接收或直接由 ControlLogix 处理器发出（事件命令控制）。在命令列表状态数据块中，每个命令的状态被返回到处理器。用户自己定义这个状态数据块在模块内部数据库中的位置。下面通过流向图并结合表格来详细说明进出模块的数据流动。




| 步骤 | 描述 |
|---|---|
| 1 | 主站驱动从 ControlLogix 处理器获得设置数据。设置数据包括命令的个数和主站命令列表。主站驱动使用这些数值来决定发送到 Modbus 网络上其它节点的命令类型（参阅 **MVI56-MCM 模块设置指导**）。 |
| 2 | 设置完之后，主站驱动开始向网络上的其它节点传送读和/或写命令。如果向另一个节点写数据，写命令中的数据就从模块内部数据库获得，并创建命令。 |
| 3 | 假设命令对象的节点成功处理了数据，主站驱动会接收到一个响应讯息供处理。 |
| 4 | 假设是读命令，来自于网络节点的数据传送到模块的内部数据库。 |
| 5 | 主站命令列表中的每个命令的状态返回到 ControlLogix 处理器。 |

参阅 **模块设置** 章节，可以找到定义虚拟 Modbus 主站端口所需要的参数的完整定义。可以参考 **MCM 驱动** 文档来获取有关每个命令的结构和内容的信息。为预先准备模块运作，在创建命令列表时要有足够的注意。如果有两个命令写到模块的同一个内部数据库地址，那结果肯定不是事先想要得到的。所有包含无效数据的命令也会被模块所忽略。

### 2.9.4.1 主站命令列表

主站模式的功能实现依赖于模块主站命令列表的定义。列表包含 100 个独立的输入条目，每个条目都包含创建有效命令需要的信息。这些信息包含以下内容：

- 命令使能模式（（0）禁止，（1）连续或（2）条件执行）
- 从站节点地址
- 命令类型 – 每条命令读或写可达 125 个字（16000 个位）
- 数据库源和目标寄存器地址 – 决定数据存放和/或获取的地址
- 个数 – 选择传送字的个数 – 1 到 100 当 FC 3, 4, 或 16。选择位的个数当 FC 1, 2, 15。

当列表从处理器读入和命令在处理执行时，模块内保存了每个命令的错误数值。这个错误列表可以发送回处理器。模块产生的错误信息如下表所示。

**标准 Modbus 协议错误**

| 编码 | 描述 |
|---|---|
| 1 | 非法的功能 |
| 2 | 非法的数据地址 |
| 3 | 非法的数值 |
| 4 | 相关设备失败 |
| 5 | 应答 |
| 6 | 忙，讯息退回 |

**模块通讯错误编码**

| 编码 | 描述 |
|---|---|
| -1 | 在传输前没有把 CTS modem 控制线置位 |
| -2 | 讯息传送超时 |
| -11 | 请求发送后，接收响应超时 |
| 253 | 响应错误的从站地址 |
| 254 | 响应错误的功能代码 |
| 255 | 响应无效的 CRC/LRC 数值 |

**命令列表输入错误**

| 编码 | 描述 |
|---|---|
| -41 | 无效的使能代码 |
| -42 | 内部数据库 > 最大地址 |
| -43 | 无效节点地址 (< 0 或 > 255) |
| -44 | 个数参数设置成 0 |
| -45 | 无效的功能代码 |
| -46 | 无效的交换代码 |

# 3 修改模块设置

为使 MVI56-MCM 模块运作，至少要设置一些参数发送到模块。下面的表格说明了各种设置数据，这些数据取决于要支持的运作模式。

| 模块寄存器地址 | 相关功能模式 | 名称 | 描述 |
|---|---|---|---|
| 5000-5009 | 数据传输 | 模块总体设置 | 这部分设置数据包含了定义模块和 ControlLogix 处理器之间数据交换的模块设置。 |
| 5010-5039 和 5040-5069 | 主站和从站 | 端口设置 | 这个部分定义了模块的每个 Modbus 串行通讯端口的特征。为使模块正常工作，这些参数必须设置正确。 |
| 5070-5869 和 5870-6669 | 主站 | 主站命令列表 | 如要实现模块主站功能，必须设置主站命令列表。 |
| 6750–6770 | 主站和从站 | 辅助端口设置 | 这个部分定义了模块的每个 Modbus 串行通讯端口的特征。为使模块正常工作，这些参数必须设置正确。 |

参阅**模块设置**章节，可以查询到模块设置的描述。MVI56-MCM 模块在上电时至少要设置一次才能工作。之后如果参数必须更改，可以在任何时候再重新设置。

## 3.1 上电

上电后，模块就进入一个逻辑循环，等待处理器的设置数据。接收设置数据完毕后，如果存在命令列表，模块就开始执行命令列表 。

## 3.2 运行中更改参数

如上面的表格所示，模块的设置数据在模块的数据库里有一个复制的映射。当模块从 ControlLogix 处理器接收到设置数据时，这些数值都被初始化。网络上的任何节点都可以改变这些数值。模块的主站端口可以查询从站来获取这些数据，或从站端口可以从远程主站那里得到这些数据。模块在接收到命令之前是不会使用这些数据的。梯形逻辑程序可以发送**写设置**命令数据块(9997)到模块。远程设备可以在地址 6800 写数据 9997，来向处理器下载设置。还可以从模块的设置/调试端口直接发送指令来下载设置。所有三种方式都可以强制模块下载设置到 ControlLogix 处理器。处理器中的梯形逻辑要能接收模块发送的数据块。如果设置都正确，模块可以从远程设备接收设置数据。

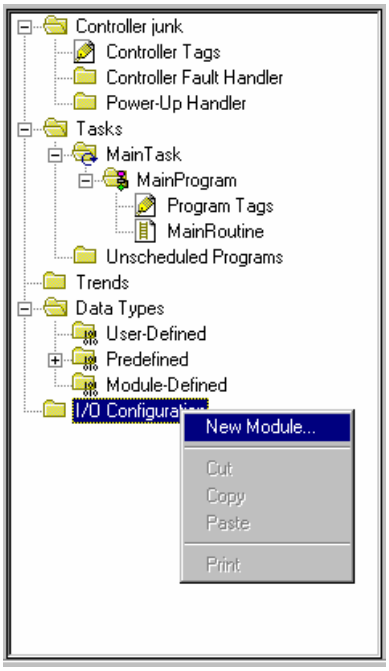
## 3.3 装配模块

装配 MVI56-MCM 模块仅需要 RSLogix 5000 软件。完成模块的应用，最简单的方法是从模块附带的样例程序(MVI56_MCM_EX1.ACD)开始。

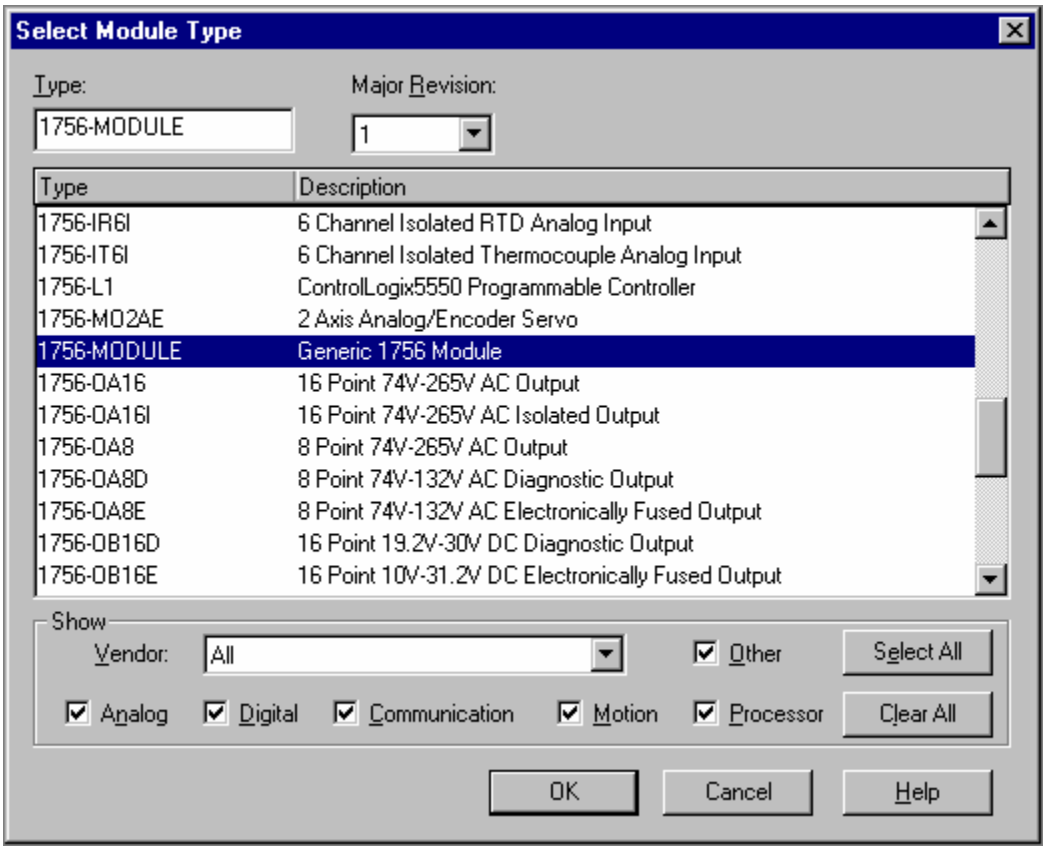
如果在现存的应用中添加模块，你可以把需要的样例程序的要素复制到你的应用中。

**注意：** 模块只有在软件离线的状态下才能添加到程序中去。

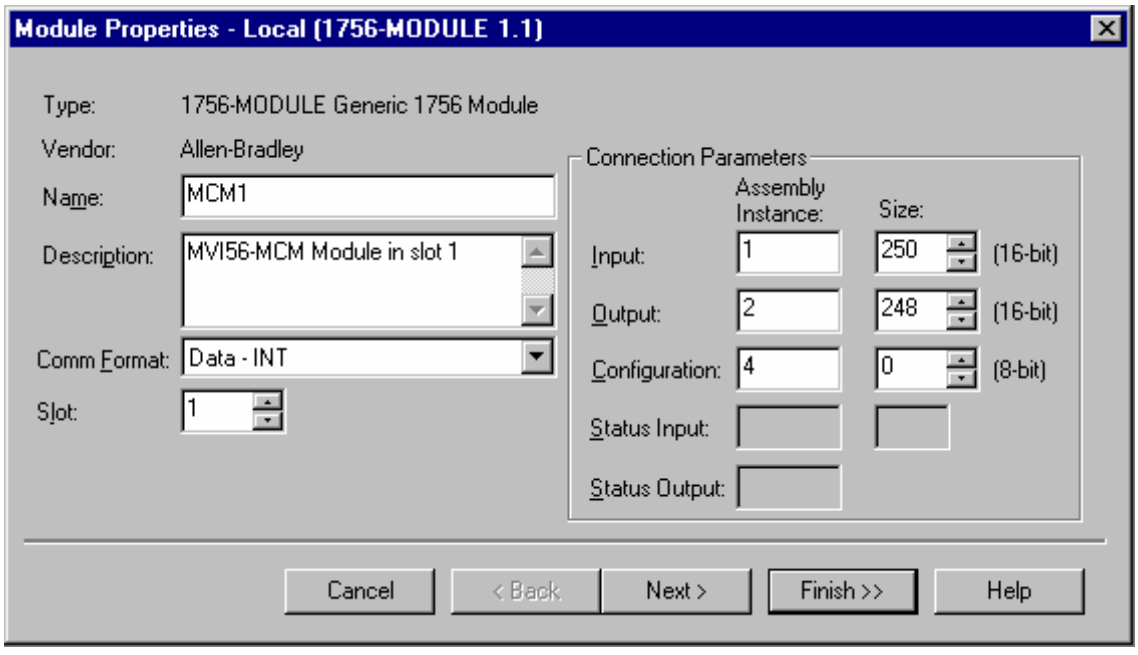
设置模块的第一步是在系统里定义模块。在控制器组织窗口（Controller Organization window ）的 I/O 设置选项（I/O Configuration ）上点右键，会弹出一个菜单。从 I/O Configuration 菜单选择 New Module：




程序会弹出下面这个窗口：

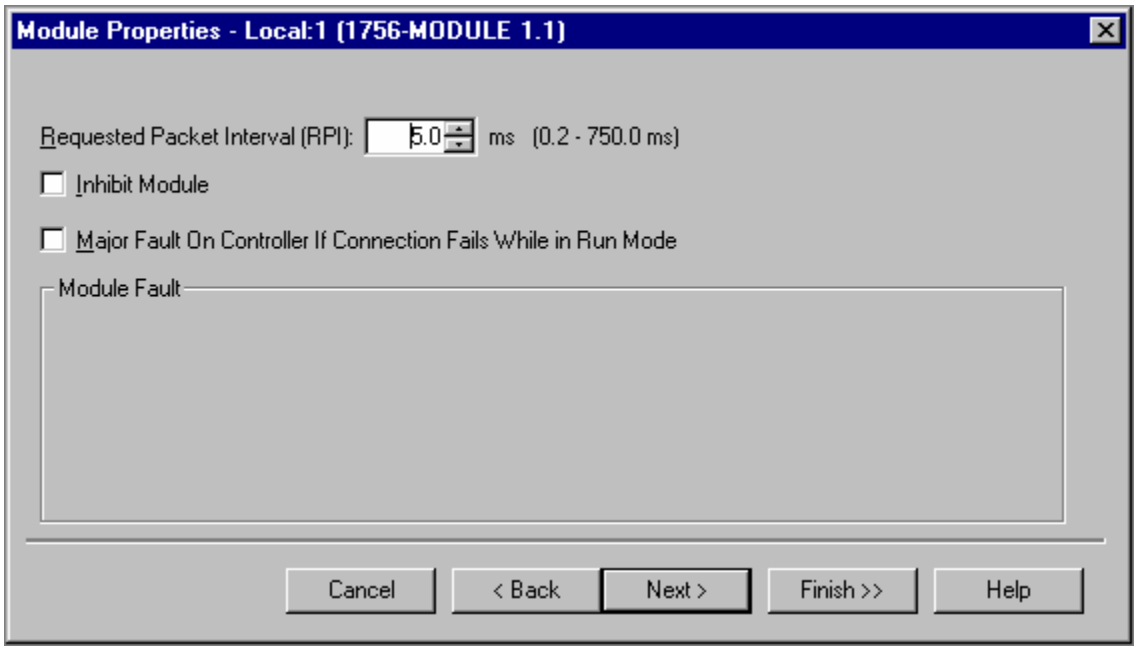



在菜单里选择 1756-Module (Generic 1756 Module)，按 OK 键。然后会显示下面的窗口：



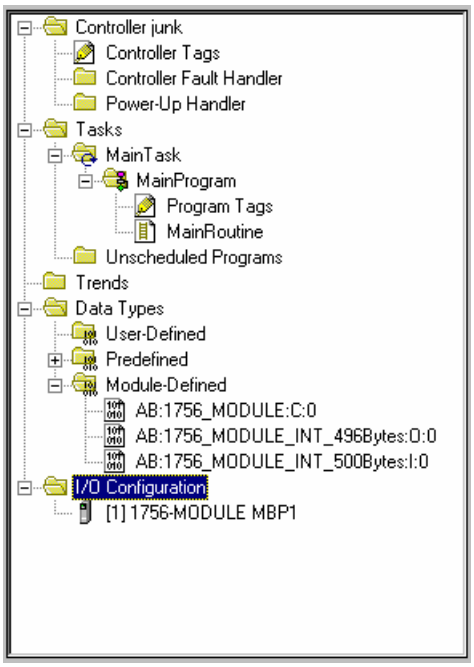

如图显示，为您的应用填写以下内容：名称（Name），描述（Description）和槽（Slot）选项。请确定选择**Comm Format** 里的**Data – INT** 选项。如果 **Assembly Instance** 和

**Size** 数值设置不正确，模块和ControlLogix槽架背板的通讯就不会成功。在下面显示的对话框中选择下一步。

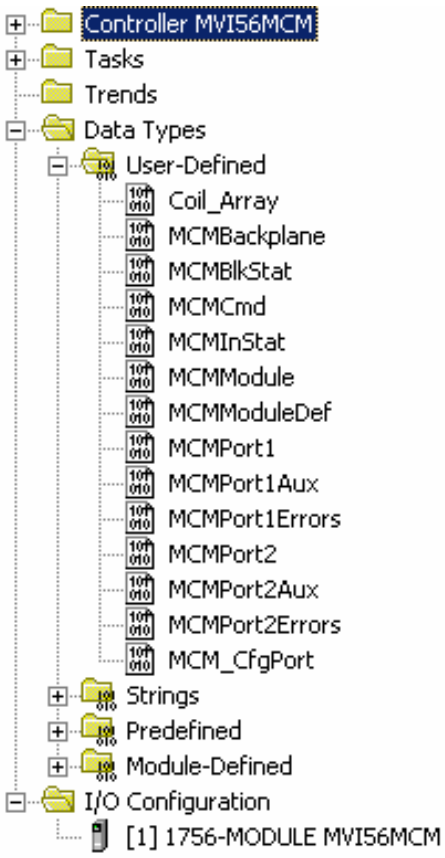



填写 Request Packet Interval 数值来确定对模块 I/O 的扫描。这个数值代表了模块处理计划事件的最小频率。数值不能低于 1 毫秒。对于绝大多数的应用，可以选择 1 到 10 毫秒的数值。

在完成模块的设置后，在控制器组织窗口（Controller Organization window）里会显示出模块。然后定义模块需要的数据，在控制器标签区数据域（Controller Tags data area）里分配这些数据对象的空间。下面是一个控制器组织窗口（Controller Organization window）的示例：

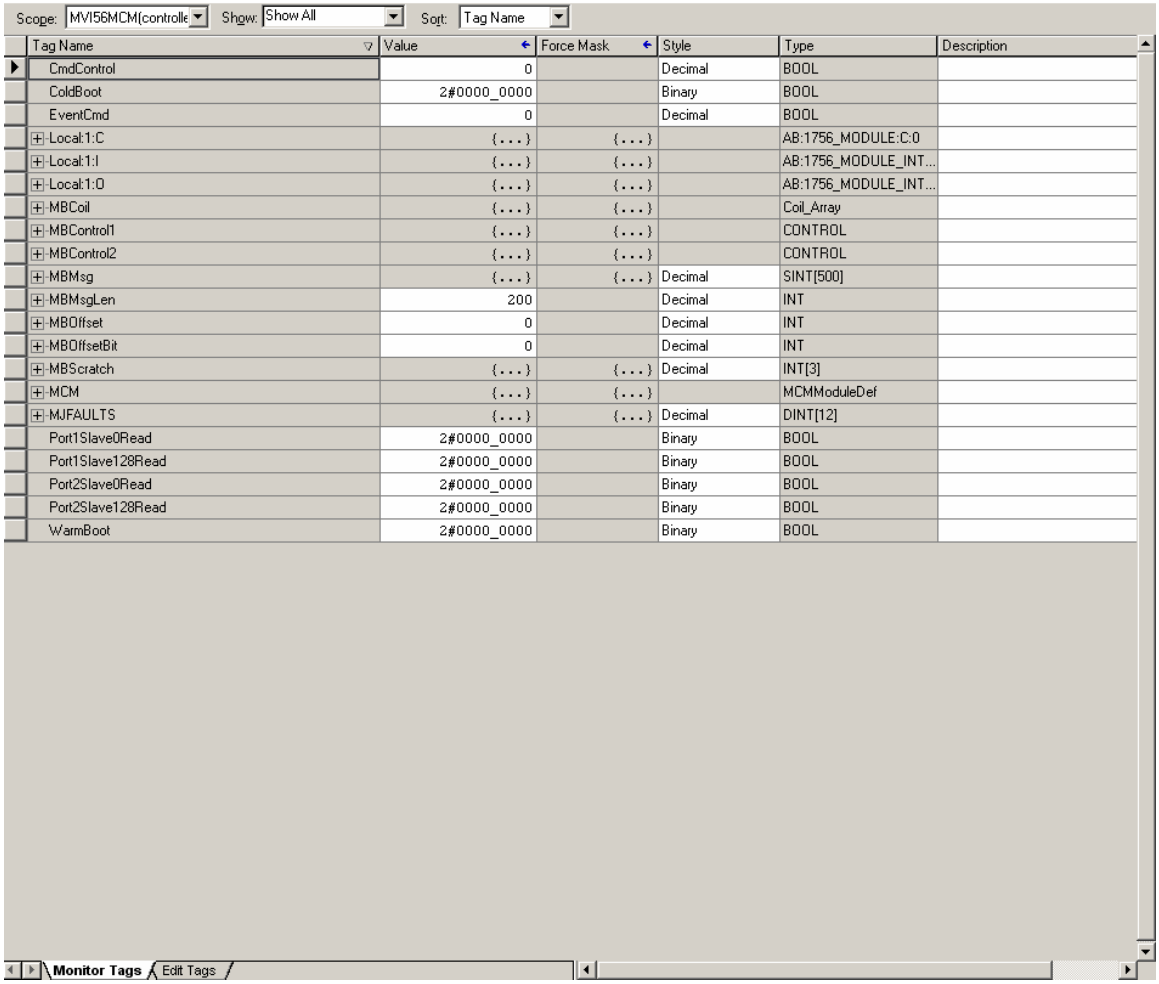



接下来的步骤是定义模块的用户定义数据类型（User Defined Data Types）。如果您不使用提供的样例程序，那可以从样例程序中复制这些数据类型。如果您使用的是样例程序，那么这些数据类型都已经定义好。如下图所示，控制器窗口（Controller Organization window）会显示出定义过的用户数据类型（User Defined Data Types）：



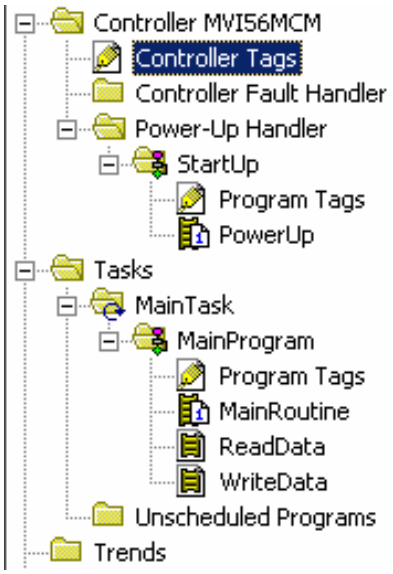
下一步定义和模块及梯形逻辑程序交接的数据。

打开控制器标签编辑窗口（Controller Tags Edit Tags dialog box），按照下面的图示来填写数据。MVI56-MCM 模块在样例中定义为 MCM1。您可以按您的需要来定义任何合法的名称。如果您使用的是样例程序，那么这些步骤都已经实现。




这里，需要用一些时间填写 MCM1 数据表中的设置数值和调节矩阵的尺寸。有关模块设置的内容，请参阅模块数据对象章节（**Module Data Object** ）。

模块设置最后的步骤是添加梯形逻辑程序。如果您使用我们提供的样例程序，则可以把它做适当的调整以适合您的应用。如果您不使用样例程序，那可以复制下图中的梯形逻辑程序到您的应用中。

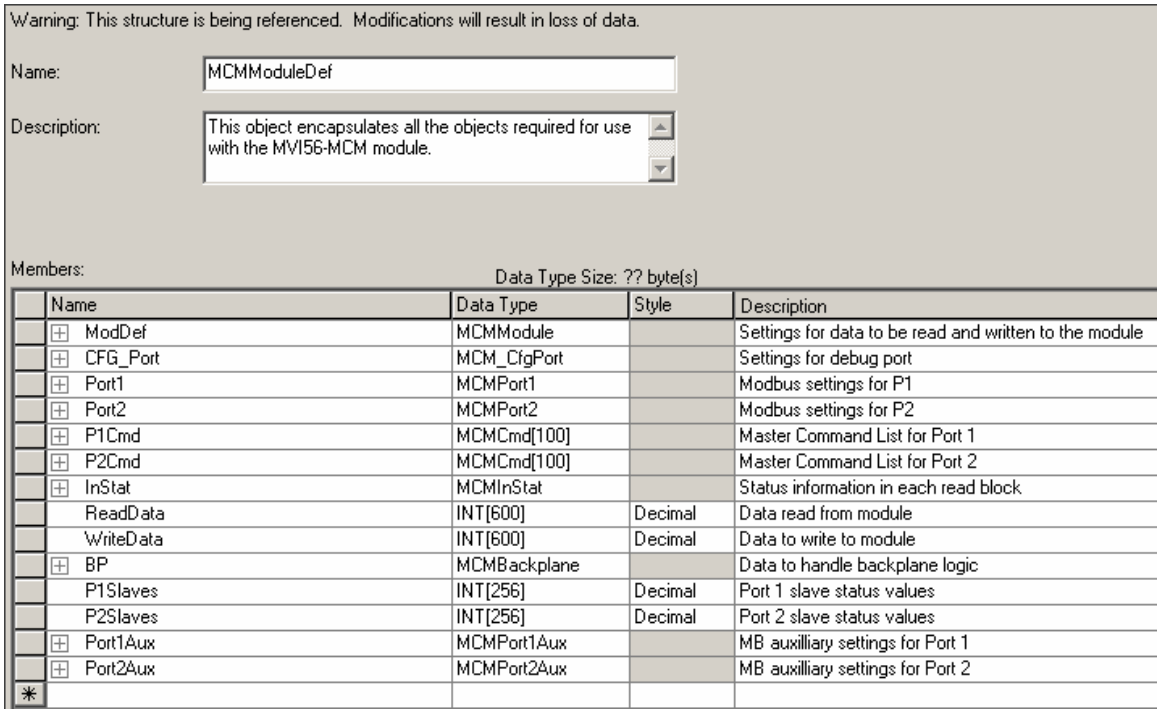



现在，模块已经设置好，可以实现您的应用。

把新的应用程序下载到处理器中，并把处理器切换到运行模式。如果所有的设置数据都正确，而且模块已经连接到 Modbus 网络，模块面板上的 应用指示灯(APP LED)应该熄灭，背板活动指示灯(BP ACT)应该快速闪烁。如果您的模块出错，请参阅 **诊断和纠错** 章节。可以连接一台计算机或终端设备到模块的调试/设置端口，使用模块内置的调试器来检查模块的状态。

## 3.4 模块数据对象 (MCMModuleDef)

和 MVI56-MCM 模块相关的所有数据都存储在用户定义的数据类型里。在使用模块前，必须先定义这些数据类型。这个步骤非常简单，只需要在控制器标签编辑窗口中声明一个变量就可以了。数据对象的结构如下表所示。



Warning: This structure is being referenced. Modifications will result in loss of data.

Name: MCMModuleDef

Description: This object encapsulates all the objects required for use with the MVI56-MCM module.

Members: Data Type Size: ?? byte(s)

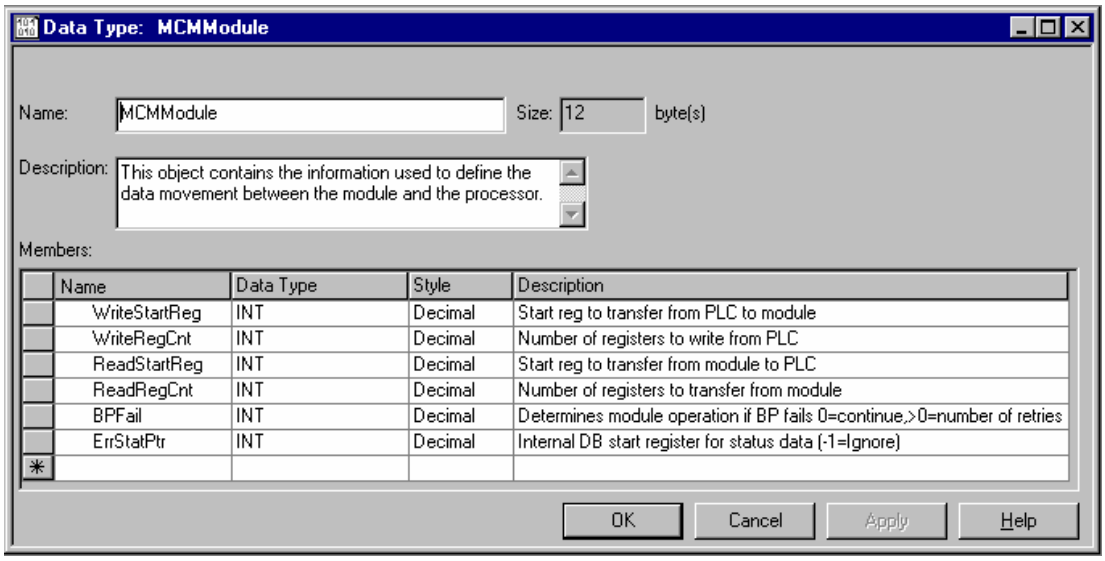
| Name | Data Type | Style | Description |
|---|---|---|---|
| ModDef | MCMModule | | Settings for data to be read and written to the module |
| CFG_Port | MCM_CfgPort | | Settings for debug port |
| Port1 | MCMPort1 | | Modbus settings for P1 |
| Port2 | MCMPort2 | | Modbus settings for P2 |
| P1Cmd | MCMCmd[100] | | Master Command List for Port 1 |
| P2Cmd | MCMCmd[100] | | Master Command List for Port 2 |
| InStat | MCMInStat | | Status information in each read block |
| ReadData | INT[600] | Decimal | Data read from module |
| WriteData | INT[600] | Decimal | Data to write to module |
| BP | MCMBackplane | | Data to handle backplane logic |
| P1Slaves | INT[256] | Decimal | Port 1 slave status values |
| P2Slaves | INT[256] | Decimal | Port 2 slave status values |
| Port1Aux | MCMPort1Aux | | MB auxilliary settings for Port 1 |
| Port2Aux | MCMPort2Aux | | MB auxilliary settings for Port 2 |

这个对象结构包含了定义设置，用户数据，状态和命令控制数据，这些和模块相关数据的对象结构。在下面的章节里讨论这些对象结构。

### 3.4.1 设置对象

对模块的设置只需要简单的在控制器标签编辑窗口中定义的模块对象中添写数据。模块需要的每个参数在这个对象中都有确定的位置。下面的表格和内容会解释在这个窗口中设置的这些数据。您可以通过打开在控制器组织窗口（Controller Organization window）的用户定义数据类型选项（User Defined Data Type）来查看这些表格。

### 3.4.1.1 数据传输参数 (MCMModule)




这个对象用来定义模块和处理器之间数据移动的参数。这些数值决定了梯形逻辑程序和应用程序需要的数据尺寸。ReadData 和 WriteData 这两个矩阵的长度必须等于或大于输入的个数数值（count values）。梯形逻辑程序要负责处理需要传送的数据块的个数。数据块的个数可以通过下面的公式来计算：

BlockCnt = INT(RegCnt/200) + if(MOD(RegCnt,200), 1,0)

如果寄存器的个数恰好可以被 200 整除，那么数据块的个数就很容易计算，而且梯形逻辑程序也非常容易编写。如果个数不能被 200 整除，那么需要编写一些特别的程序来处理最后的那个数据块，因为这个需要传送的数据块的长度小于 200。建议用户在应用中最好把数据设置成能被 200 整除的数据。

参数 BPFail 定义当背板传输失败时，模块是否继续在 Modbus 网络上的通讯。数值 0 表明当背板不再工作时，继续进行 Modbus 的通讯。如果设置的数值大于 0，背板会按照输入数值相等的次数来重试运作，如果还是失败，模块会报告一个错误，端口上的通讯也会停止。当背板通讯恢复时，模块会开始和 Modbus 网络进行通讯。比如，如果您输入 10，那么模块连续 10 次尝试背板通讯都失败时，会停止所有的 Modbus 通讯。如果模块确定传输再次成功，则会继续在 Modbus 网络上的通讯。

参数 ErrStatPtr 定义在模块数据库中存放错误/状态数据的位置。如果数值设置成 –1，这些数据则不会存放在用户数据区域。如把参数设置成 0 到 4939，那么模块程序会把数据存放在指定的位置。

### 3.4.1.2 Modbus 端口参数 (MCMPort)

Name: MCM_CfgPort

Description: Settings for the CFG port of the module. These values are to be used on the Hyperterminal connection to the debug port.

Members: Data Type Size: 12 byte(s)

| Name | Data Type | Style | Description |
|---|---|---|---|
| Baudrate | DINT | Decimal | Baudrate for P1 debug port. Fixed value and cannot be changed with this setting. |
| Parity | INT | Decimal | 0=None. Fixed value |
| DataBits | INT | Decimal | 8 data bits. Fixed value |
| StopBits | INT | Decimal | 1 stop bits. Fixed value |

Name: MCMPort1

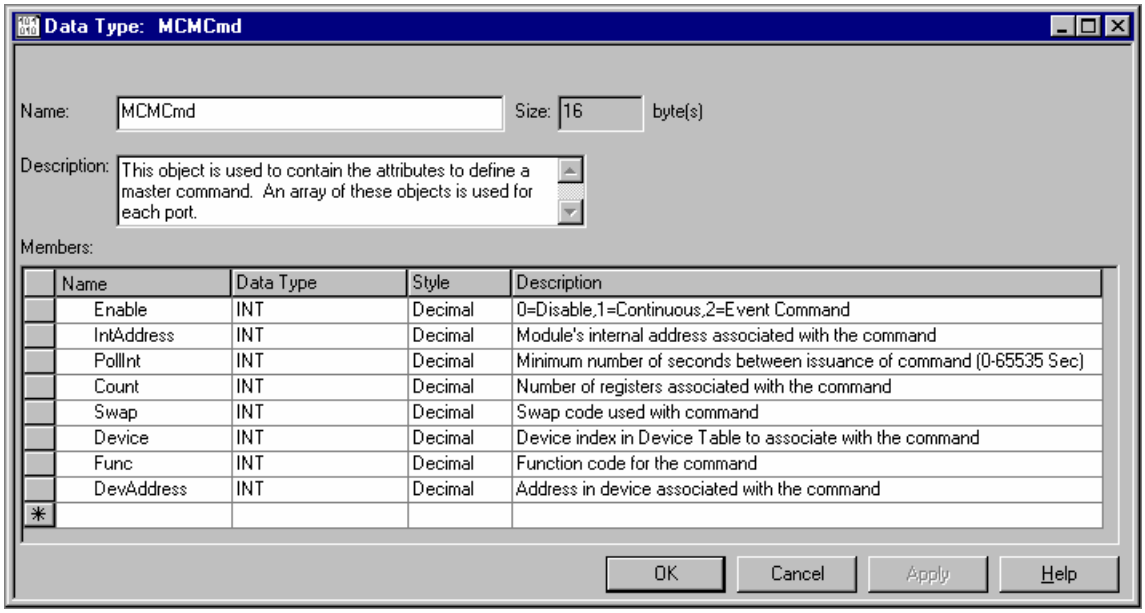
Description: The serial port configuration for the MVI56MCM module.

Members: Data Type Size: 52 byte(s)

| Name | Data Type | Style | Description |
|---|---|---|---|
| Enabled | INT | Decimal | 0=Port Disabled,1=Port Enabled |
| Type | INT | Decimal | 0=Master, 1=Slave, 2=Slave: pass-through, 3=Slave: formatted pass-through/data swapped, 4=Slave: form. pass-through |
| FloatFlag | INT | Decimal | 0=No floating-point data, 1=Use floating-point data |
| FloatStart | INT | Decimal | Register offset in message for floating-point data |
| FloatOffset | INT | Decimal | Internal DB offset to start of floating-point data |
| Protocol | INT | Decimal | 0=Modbus RTU, 1=Modbus ASCII |
| Baudrate | INT | Decimal | Baudrate for port (110 to 115.2K) |
| Parity | INT | Decimal | 0=None, 1=Odd, 2=Even, 3=Mark, 4=Space |
| DataBits | INT | Decimal | 5 to 8 data bits |
| StopBits | INT | Decimal | 1 or 2 stop bits |
| RTSOn | INT | Decimal | 0-65535 mSec delay before data |
| RTSOff | INT | Decimal | 0-65535 mSec delay after data |
| MinResp | INT | Decimal | 0-65535 mSec minimum time before response to request |
| UseCTS | INT | Decimal | 0=No, 1=Yes to use CTS modem line |
| SlaveID | INT | Decimal | 1-255 Modbus Node Address (Slave) |
| BitInOffset | INT | Decimal | Internal DB offset to bit input data (Slave) |
| WordInOffset | INT | Decimal | Internal DB offset to word input data (Slave) |
| OutOffset | INT | Decimal | Internal DB offset to bit output data (Slave) |
| HoldOffset | INT | Decimal | Internal DB offset to holding register data (Slave) |
| CmdCount | INT | Decimal | Command list count (Master) |
| MinCmdDelay | INT | Decimal | 0-65535 mSec minimum time between each command (Master) |
| CmdErrPtr | INT | Decimal | Internal DB location to place command error list (Master) |
| RespTO | INT | Decimal | 0-65535 mSec response timeout for command (Master) |
| Retry_Count | INT | Decimal | Retry count for failed request (Master) |
| ErrorDelayCntr | INT | Decimal | 0-65535 Command cycle count if error (Master) |

这个对象定义了模块上的每个 Modbus 端口的运作参数。参阅附录 C 查看每个参数的定义。

### 3.4.1.3 Modbus 主站命令(MCMCmd)




这个对象用来定义主站命令列表中每个命令的参数。 **MCMModuleDef** 对象里包含了为每个端口定义命令列表的这些对象的矩阵。每个命令都需要这些参数的定义，内容如下表所示：

| 参数 | 描述 |
|---|---|
| Enable | 此参数定义命令被执行或被忽略。下面的数值是合法的：0=禁止命令，命令不被执行。1=命令在每次命令列表扫描的时候执行，并受控于 **PollInt** 参数。2=表明命令仅会在上次命令执行后命令相关数据发生变化时才发送。这个选项仅对写命令有效。 |
| IntAddress | 参数指定与命令相关的内部寄存器的起始地址。合法的输入数值是寄存器 0 到 4999。如果是位寻址的命令就是位 0 到 65535 。 |
| PollInt | 参数定义了连续命令的执行间隔，以秒计算(Enable=1)。这个轮询命令间隔可以减轻放慢网络的负荷。对这个参数的有效输入是 0 到 65535。 |
| Count | 此参数定义命令执行时处理的寄存器个数。参数的有效输入是 1 到 125 个字或 16000 位。 |
| Swap | 当使用 Modbus 功能代码 3 来读取 Modbus 网络节点的数据时，这个参数用于确定是否要对数据进行更改。数值可以按以下内容设置：0=不进行数据交换，1=交换字数值， 2=交换字和字节的数值和 3=交换字节数值。当模块和其它设备进行 ASCII 和浮点数通讯时使用这个选项。 |
| Device | 这个参数指派了发送到 Modbus 网络上的命令要到达的从站地址。合法输入是 0 到 255。大多数 Modbus 网络的上限值是 247。 |
| Func | 此参数确定了执行了 Modbus 功能。有效输入是 1, 2, 3, 4, 5, 6, 15 和 16。 |
| DevAddress | 此参数定义命令相关的设备数据首地址。这里输入的数值取决于 Modbus 节点的数据库的定义。请参考设备生产厂商的数据库定义， |

| 参数 | 描述 |
|---|---|
| | 来确定命令执行所涉及的数据地址。 |

### 3.4.2 状态对象 (MCMInStat)

这个对象用于查看模块的状态。下图显示的对象 **MCMInStat**，在处理器每次收到读数据块时刷新一次。这些数据可以用作对模块的状态进行“实时数率”监测。

Name: MCMInStat

Description: This status data is returned on each read block and can be used to detect proper module operation.

Members: Data Type Size: 72 byte(s)

| Name | Data Type | Style | Description |
|---|---|---|---|
| PassCnt | INT | Decimal | Program cycle counter |
| Product | INT[2] | Hex | Product Name |
| Rev | INT[2] | Hex | Revision Level Number |
| OP | INT[2] | Hex | Operating Level Number |
| Run | INT[2] | Hex | Run Number |
| Prt1Errs | MCMPort1Errors | | Port error statistics |
| CmdReq | INT | Decimal | Total number of command list requests sent |
| CmdResp | INT | Decimal | Total number of command list responses received |
| CmdErr | INT | Decimal | Total number of command list errors |
| Requests | INT | Decimal | Total number of requests for port |
| Responses | INT | Decimal | Total number of responses for port |
| ErrSent | INT | Decimal | Total number of errors sent |
| ErrRec | INT | Decimal | Total number of errors received |
| Prt2Errs | MCMPort2Errors | | |
| CmdReq | INT | Decimal | Total number of command list requests sent |
| CmdResp | INT | Decimal | Total number of command list responses received |
| CmdErr | INT | Decimal | Total number of command list errors |
| Requests | INT | Decimal | Total number of requests for port |
| Responses | INT | Decimal | Total number of responses for port |
| ErrSent | INT | Decimal | Total number of errors sent |
| ErrRec | INT | Decimal | Total number of errors received |
| BlkErrs | MCMBlkStat | | Block transfer statistics |
| Read | INT | Decimal | Total number of read block transfers |
| Write | INT | Decimal | Total number of write block transfers |
| Parse | INT | Decimal | Total number of blocks parsed |
| Event | INT | Decimal | Total number of event blocks received |
| Cmd | INT | Decimal | Total number of command blocks received |
| Err | INT | Decimal | Total number of block transfer errors |
| Port1CurErr | INT | Decimal | Current error/index for Port 1 |
| Port1LErr | INT | Decimal | Last error/index for Port 1 |
| Port2CurErr | INT | Decimal | Current error/index for Port 2 |
| Port2LErr | INT | Decimal | Last error/index for Port 2 |

存储在这个对象中的所有数据列表，请参考附录 B。

## 3.5 用户数据对象

这些对象用于控制处理器和 MVI56-MCM 模块之间的传送的数据。用户数据就是在处理器和模块之间传送的读和写数据，这些数据按“页”方式传送。每“页”是 200 个字长。

| | | | | |
|---|---|---|---|---|
| | ReadData | INT[600] | Decimal | Data read from module |
| | WriteData | INT[600] | Decimal | Data to write to module |

读数据(**ReadData**)是一个长度和 **MCMModule** 对象中 **ReadRegCnt** 参数一致的数组。为便于使用，这个数组应该按 200 个字的偶数级增长来定义。 当数据从模块传送到处理器时，每次只能传送 200 个字。ReadData 负责把接收到的数据放置到读数据数组的争取位置中去。 这个数据可用于处理器中梯形逻辑程序的状态和控制。

写数据(**WriteData**) 是一个长度和 **MCMModule** 对象中 **WriteRegCnt** 参数一致的数组。为便于使用，这个数组应该按 200 个字的偶数级增长来定义。 当数据从模块传送到处理器时，每次只能传送 200 个字。WriteData 负责把写的数据放置正确的输出镜像中，以传输到模块。这些数据从处理器传送到模块，是用于网络上其它节点的状态和控制信息。如果数组长度> 600 个寄存器，那就需要更改在梯形逻辑程序里 ReadData 第二行以及 Writedata 第 10 行中 LIM 的高数值。

## 3.6 从站轮询控制和状态

在模块的主对象中，分配了两个数组来保存主站端口上对每个从站轮询的状态。这个状态数据可用于判断端口上哪个从站处于活动状态，通讯错误或挂起，禁止了该从站的轮询。处理器的梯形逻辑程序可以检测和控制主站端口上每个从站的状态。对象的使用见下图：

| | | | |
|---|---|---|---|
| P1Slaves | INT[256] | Decimal | Port 1 slave status values |
| P2Slaves | INT[256] | Decimal | Port 2 slave status values |

使用特殊的数据块，处理器能够向从站请求当前的数据。使用其它一些数据块，处理器可以使能或禁止对选定从站的轮询。

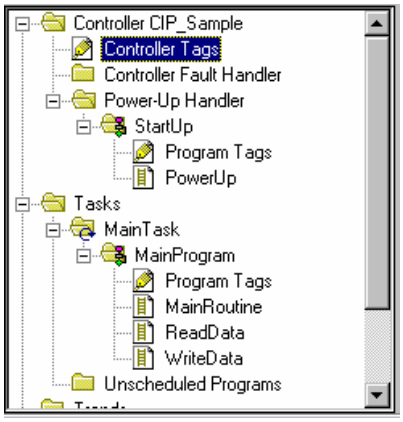
## 3.7 Modbus 讯息数据

这个新版本的模块程序包括 pass-through 模式的操作。在这种模式下，发送到从站端口的写讯息被直接发送到处理器。这个特性的使用要求梯形逻辑程序负责处理接收到的讯息。模式运作需要两个数据对象：一个变量用于保存讯息的长度，一个缓存来保存讯息。模块传送这个信息到处理器使用的是数据块 ID 号 9996。 数据块的第二个字包含了讯息的长度，讯息内容从第三个字开始。另外，还需要其它一些控制器标签来存储包含在这些讯息里的数据。Modbus 协议支持控制二进制输出(coils – 功能 5 和 15)和寄存器(功能 6 和 16)。

# 4 修改样例梯形逻辑程序

MVI56-MCM 模块的应用需要梯形逻辑程序。梯形逻辑程序需要处理的任务包括设置，数据传输，特殊数据块处理和状态数据接收。这部分内容讨论模块需要的这些程序处理面。此外，还要编写上电处理程序，来处理模块数据的初始化以及清除处理器的故障状态。

MVI56-MCM 模块的样例程序中，控制器组织窗口如下图所示。




## 4.1 上电程序（Power Up）

上电程序用来初始化 MVI56-MCM 模块使用的数据对象以及让处理器上电时从故障状态中恢复到正常。执行这些任务的梯形逻辑程序如下图所示：

如果失电或重新启动处于运行模式的处理器时出现故障，这行程序可以让处理器从故障中恢复。您也许也需要处理其它故障情况。此外，使用故障句柄能让处理器处理其它故障。在用于梯形逻辑程序前，对象 MJFAULTS 必须在控制器标签（Controller Tags）中定义：


此行程序的功能是初始化最后读和写的数值以及 MVI56-MCM 模块的输出镜像和写数据区清零。 最后读 (**MCM.BP.LastRead**) 和写 (**MCM.BP.LastWrite**)的数值用于数据传输的梯

形逻辑程序。MVI56-MCM 模块的输出镜像 (**Local:1:O.Data[]**) 用于从处理器传送数据到模块。写数据区 (**MCM.WriteData[]**) 用于存储需要使用输出镜像写到模块的处理器数据。

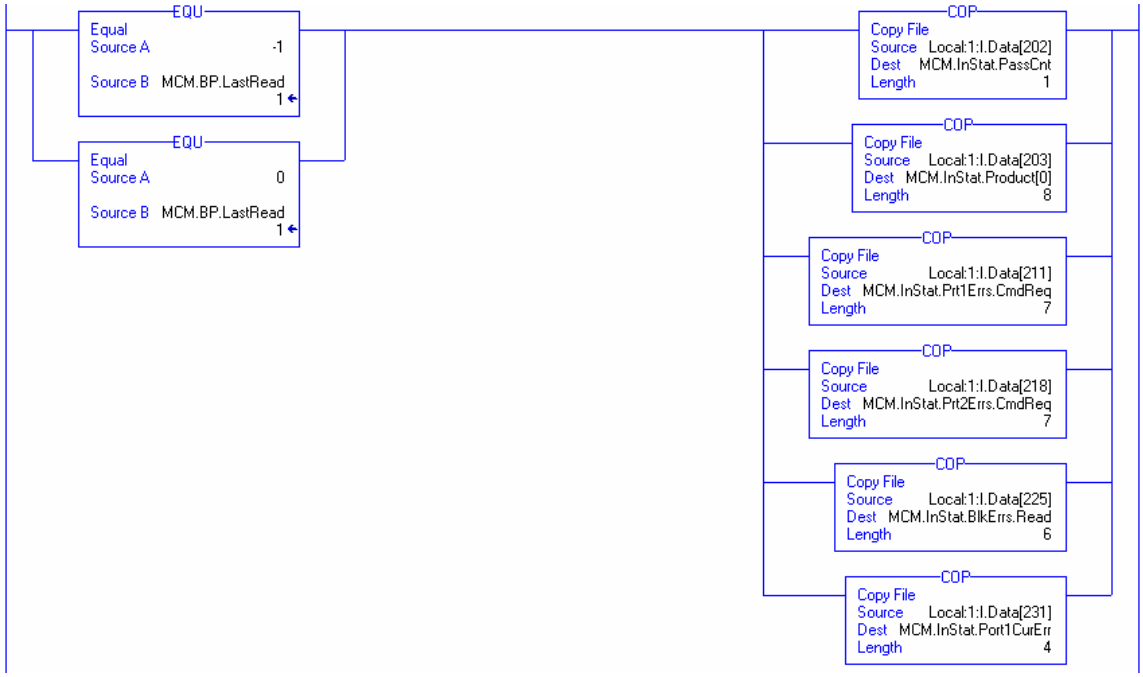

## 4.2 主程序（MainRoutine）

主程序（MainRoutine）判断是否有新的读数据从模块传送到处理器。模块会在读数据块的队列循环中把数据传送到处理器。每次新的数据到达时，模块会在输入镜像中（**Local:1:I:Data[249]**）写入数据块的数值。梯形逻辑程序连续循环扫描 这个输入字，以获取新的数值。如果有新的数值出现，梯形逻辑程序会按顺序的执行读数据（ ReadData）和写数据（WriteData）任务。

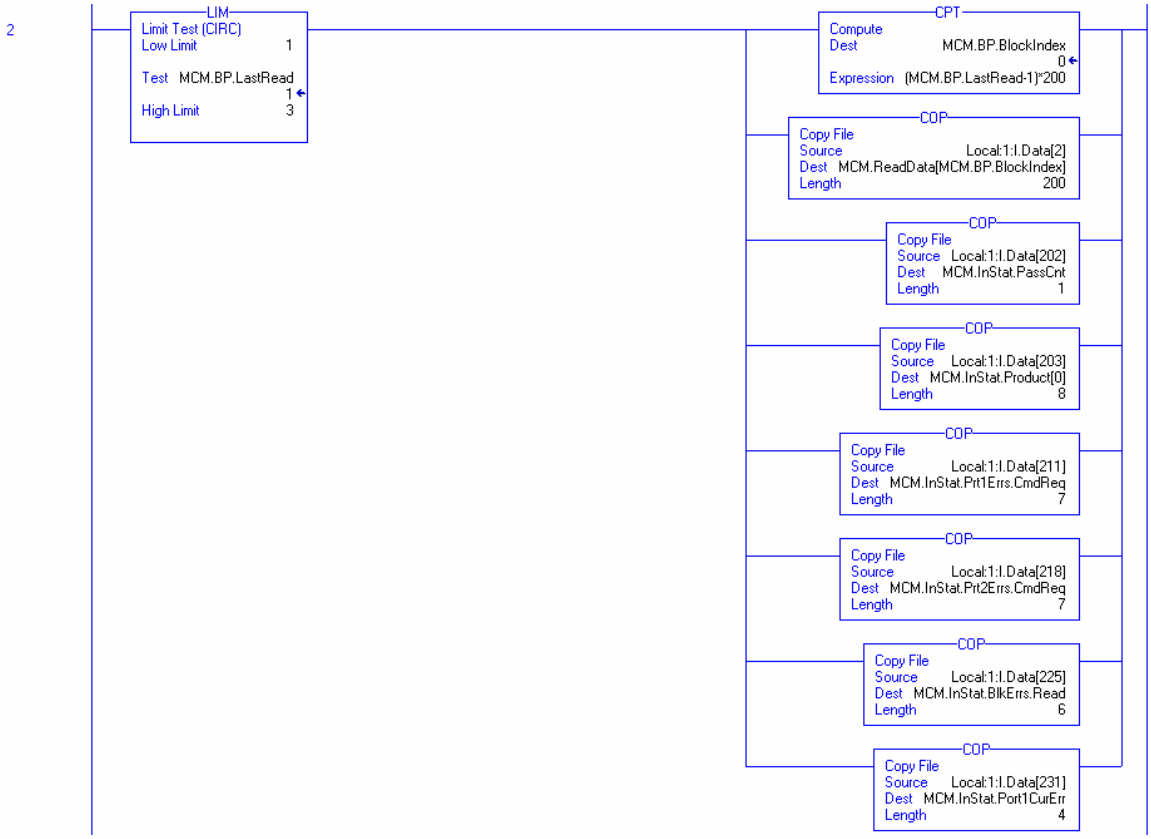

## 4.3 读数据程序（ReadData）

读数据（ReadData）任务负责处理所有从模块接收到的新数据，并把它放置在处理器正确的位置里。数据从模块传送到处理器使用的是模块的输入镜像（**Local:1:I:Data[]**）.任务程序的第一行是把最后的读数据块号（**MCM1.BP.LastRead**）改成当前模块发送过来的数据块号码（**Local:1:I:Data[249]**）。

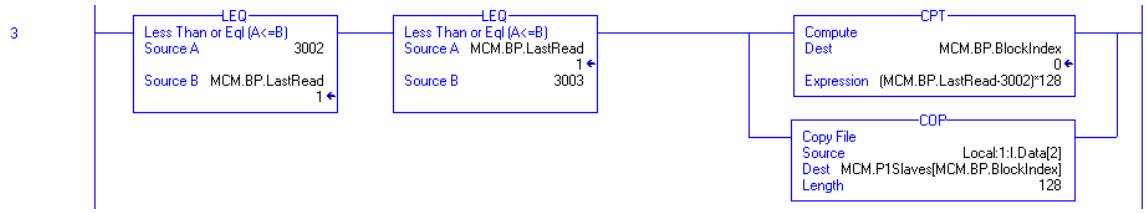

如果模块设置成传递一个或零个数据块，那么它会传送以零或-1 为标识的数据块。这些数据块中不会包含用户数据，而仅仅包含状态数据。下面显示的程序行是用来处理这些数据块的。



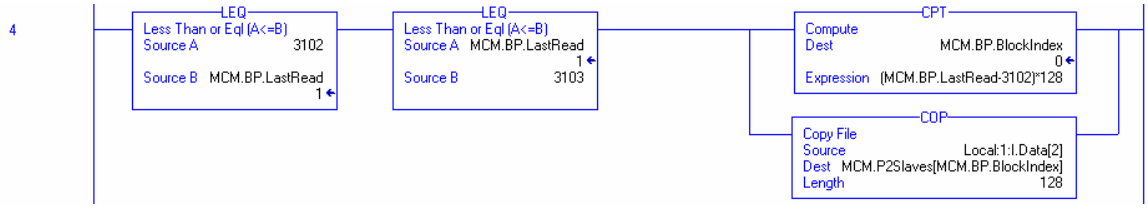

下一行程序判断在输入镜像中接收到的新数据是否是用户数据。如果是用户数据，梯形逻辑程序会把数据正确的放到处理器的读数据区内（**MCM.ReadData[]**）。每次块传输可以传送 200 个数据字。除用户数据以外，数据中还包含重要的状态数据。这些数据也要复制到模块的响应数据区域中去（**MCM.InStat**）。通过这些状态字，可以判断 MVI56-MCM 模块的“健康”程度。




接下来两行程序处理从站节点状态数据的接收。这些数据块是处理器通过写数据（WriteData）任务发送请求后，从模块传送到处理器的。下面这两行显示了如何处理这些数据块。

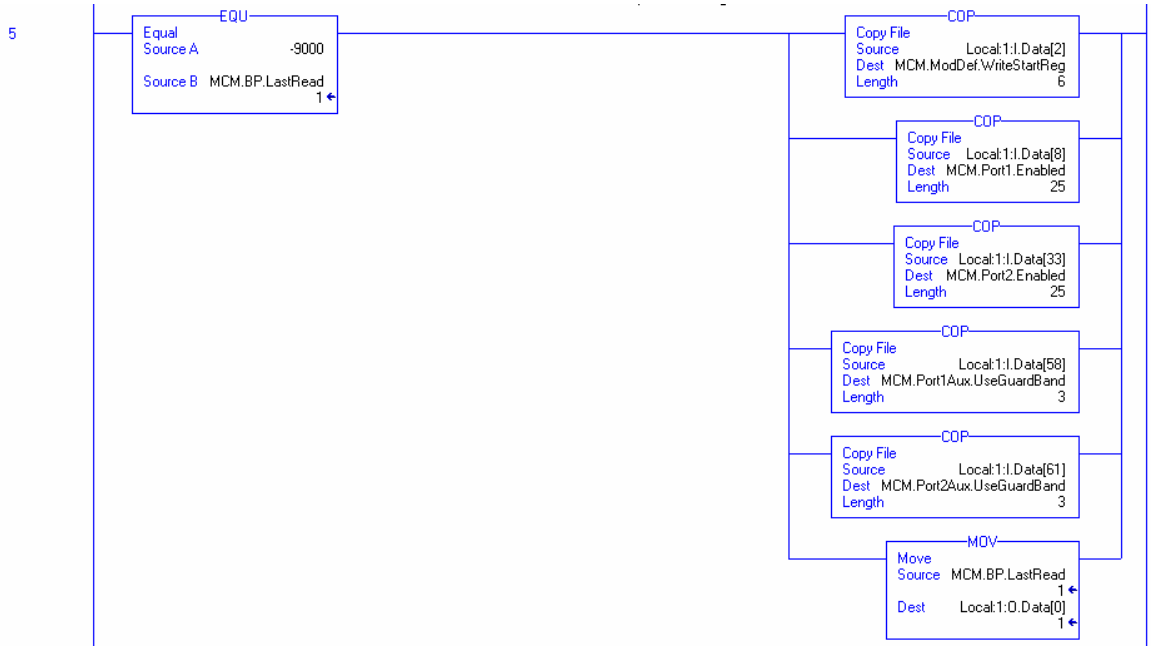



这行程序处理连接在 Modbus 端口 1 上的从站。两个数据块共 128 个从站，每个数据都由这行程序处理并放置在正确的数组位置中。

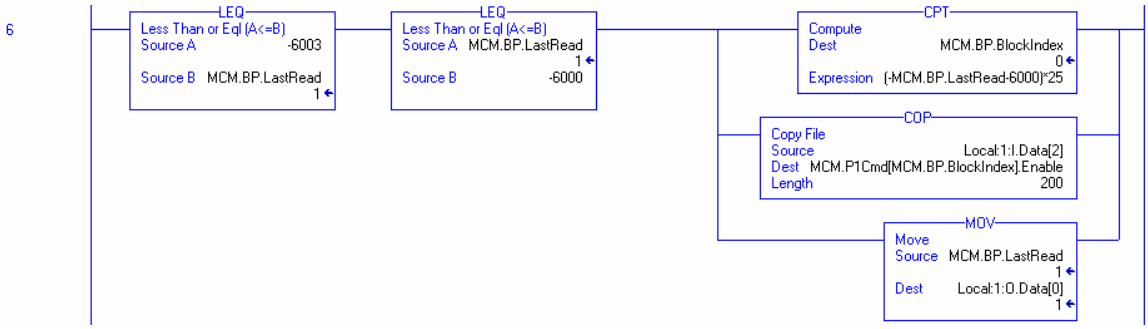



这行程序处理连接在 Modbus 端口 2 上的从站。

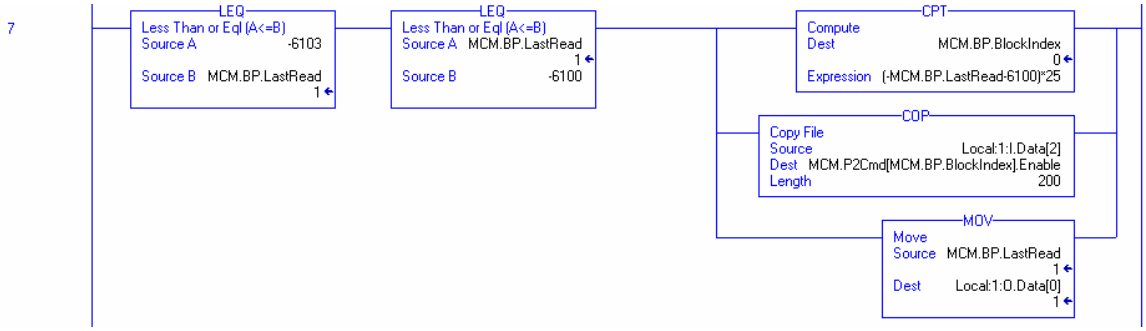
如果处理器准备接收来自于远程，通过模块数据库对模块的设置，那么必须要有专门的程序来处理这些特殊的数据块。从模块发送到处理器的这些设置信息，通过数据块-9000，-6000 到-6003 和-6100 到-6103 来传送。下面是完成此功能的样例梯形逻辑程序。




此行程序处理模块常规设置数据的接收。




此行程序处理 Modbus 端口 2 上主站命令列表数据的接收。

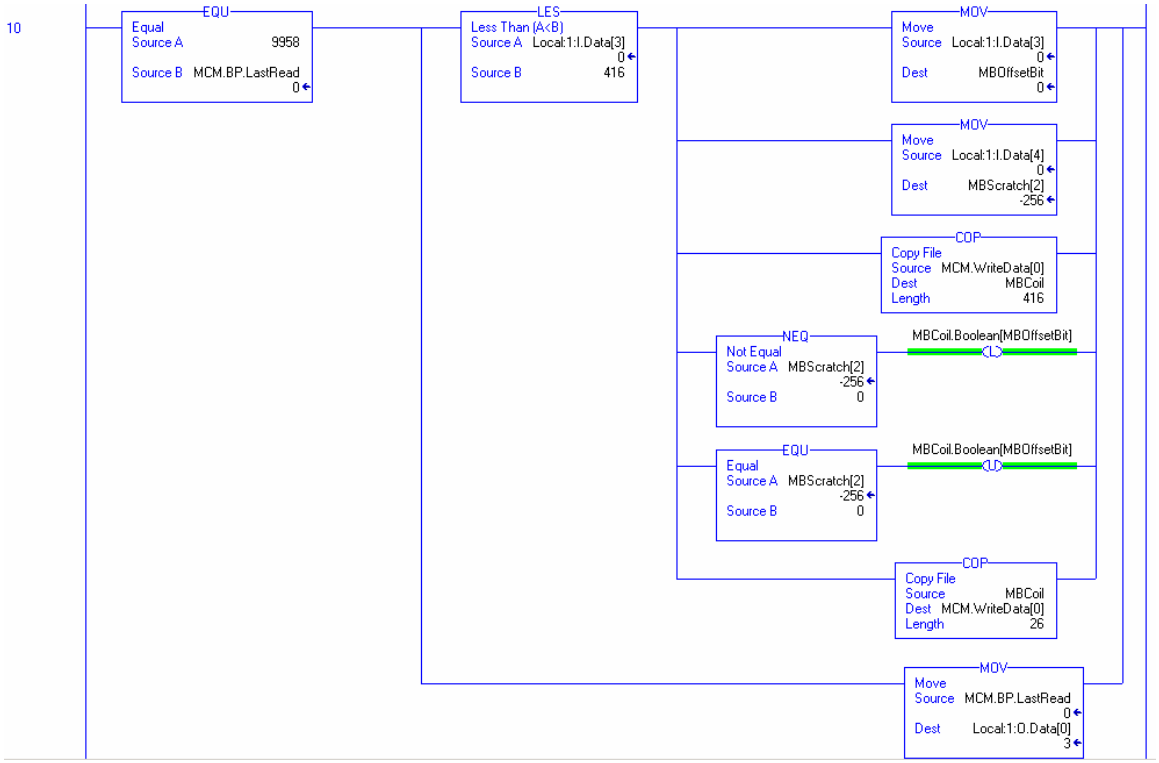



此行程序处理 Modbus 端口 3 上主站命令列表数据的接收。在未来对产品进行升级时，还会有其它的一些数据块需要处理。

下面的程序复制来自于远程主站的讯息。这些讯息通过模块（pass-through 模式）直接传送到处理器。


当模块运作在 Pass-Thru 模式下，这行程序负责处理功能代码 6 和 16 的请求。


当模块作为从站在 Pass-Thru 模式下工作，这行程序负责处理功能代码 5 的请求：

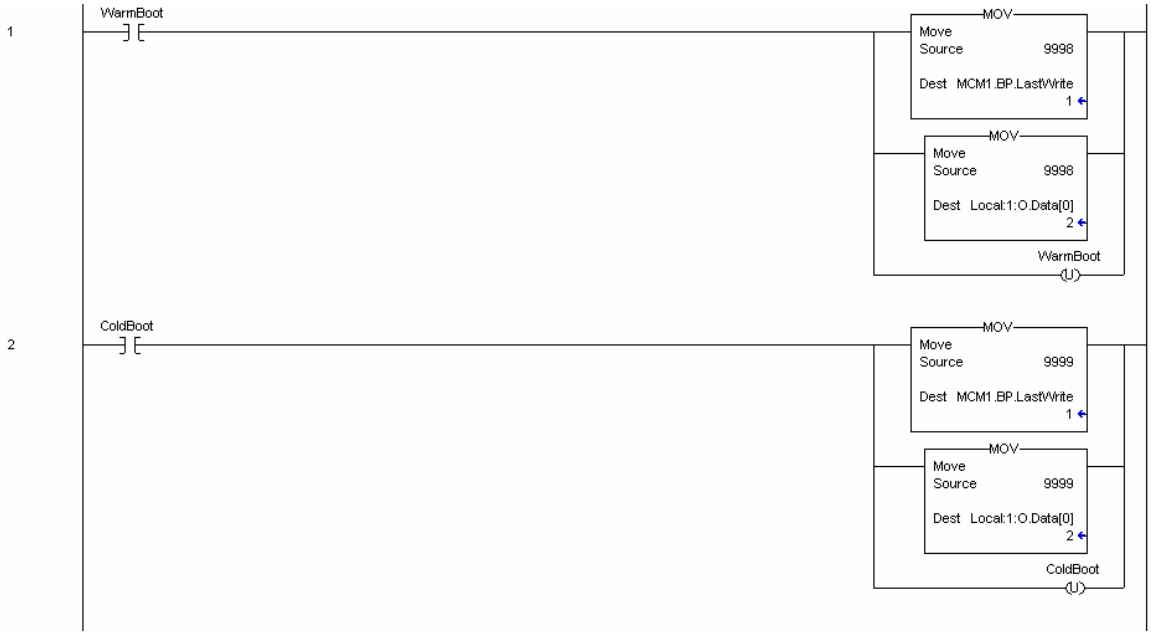



**注意：**第 11 行程序（此处未显示）负责处理 pass-thru 逻辑的功能代码 15。此行程序**不可**修改。

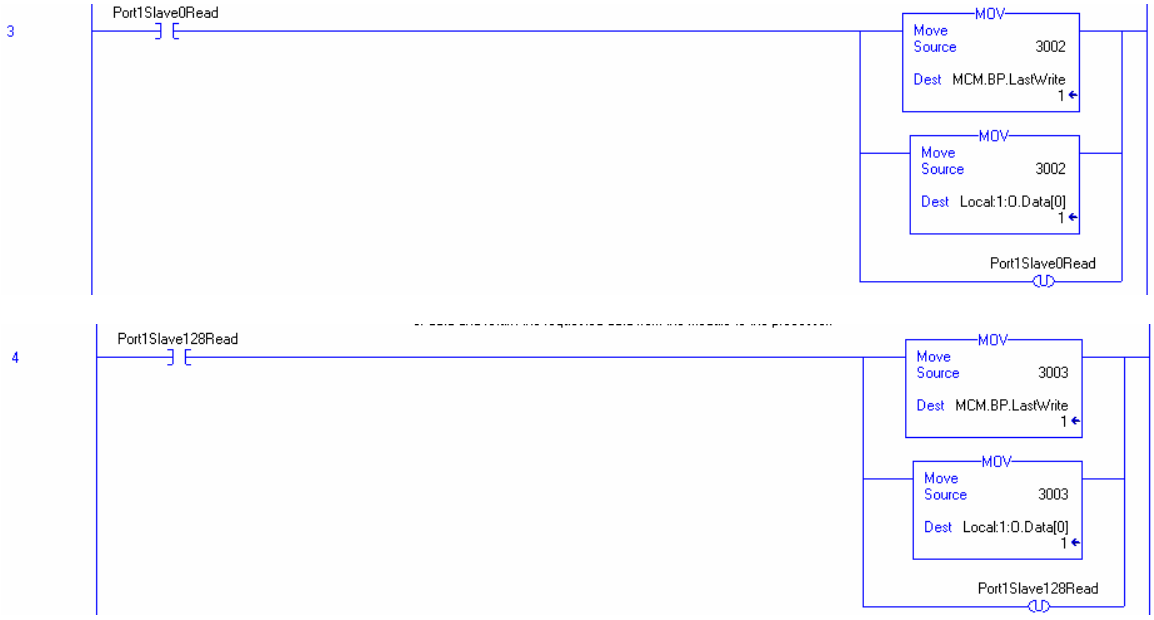
## 4.4 写数据程序（WriteData）

写数据（WriteData）任务负责从处理器发送数据到 MVI56-MCM 模块 。数据使用模块的输出镜像（**Local:1:O:Data[]**）从处理器传递到模块。第一行程序把当前请求的数据集存储到模块的 **MCM.BP.LastWrite** 数据对象中。如果输入字（**Local:1:I:Data[1]**）在处理过程中发生变化，这个对象则仍然能在后续的梯形逻辑程序中使用。

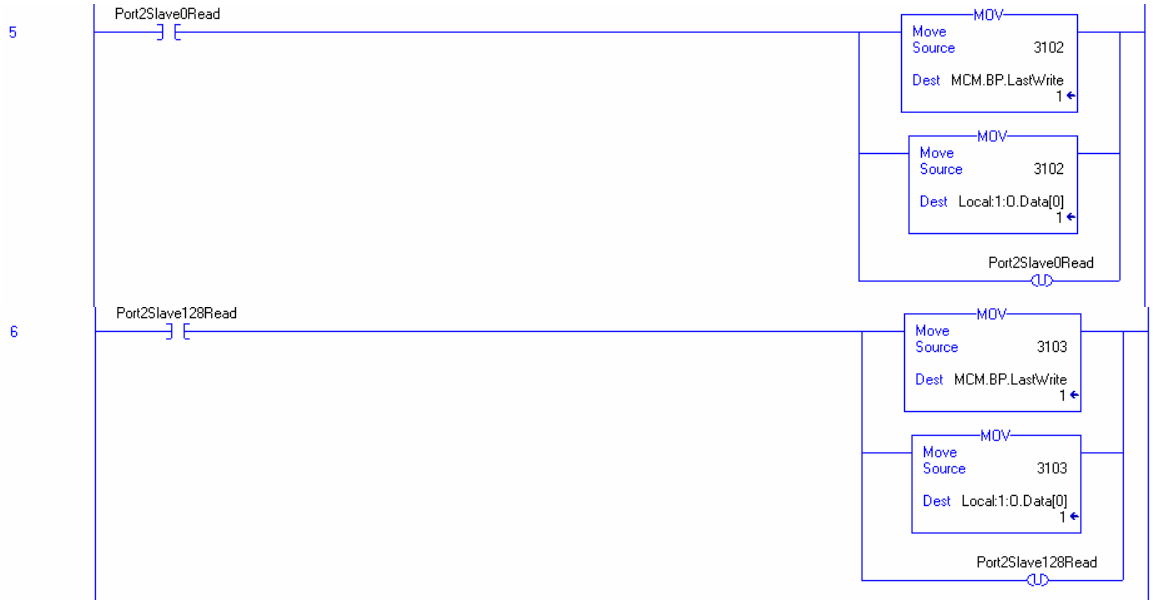

下面两行程序用来处理处理器对模块的控制。这种控制通过使用冷启动和热启动来实现。当处理器要求模块进行这当中的某个操作时，只需要简单的复制数据块号码到模块的输出镜像，模块就会执行这个操作。 注意一定要在 last write 对象中写入需要的数据块号码，这样就避免了写数据（WriteData）任务的深度处理。每个控制数据块的样例如下。




接下来的四行程序用来请求每个主站端口上从站节点的状态数据。每个端口要有两个请求，这是为了获得在一个端口上可能存在的 256 个从站地址的数据。 下面的梯形图演示了如何从 Modbus 端口 1 上请求这些数据。



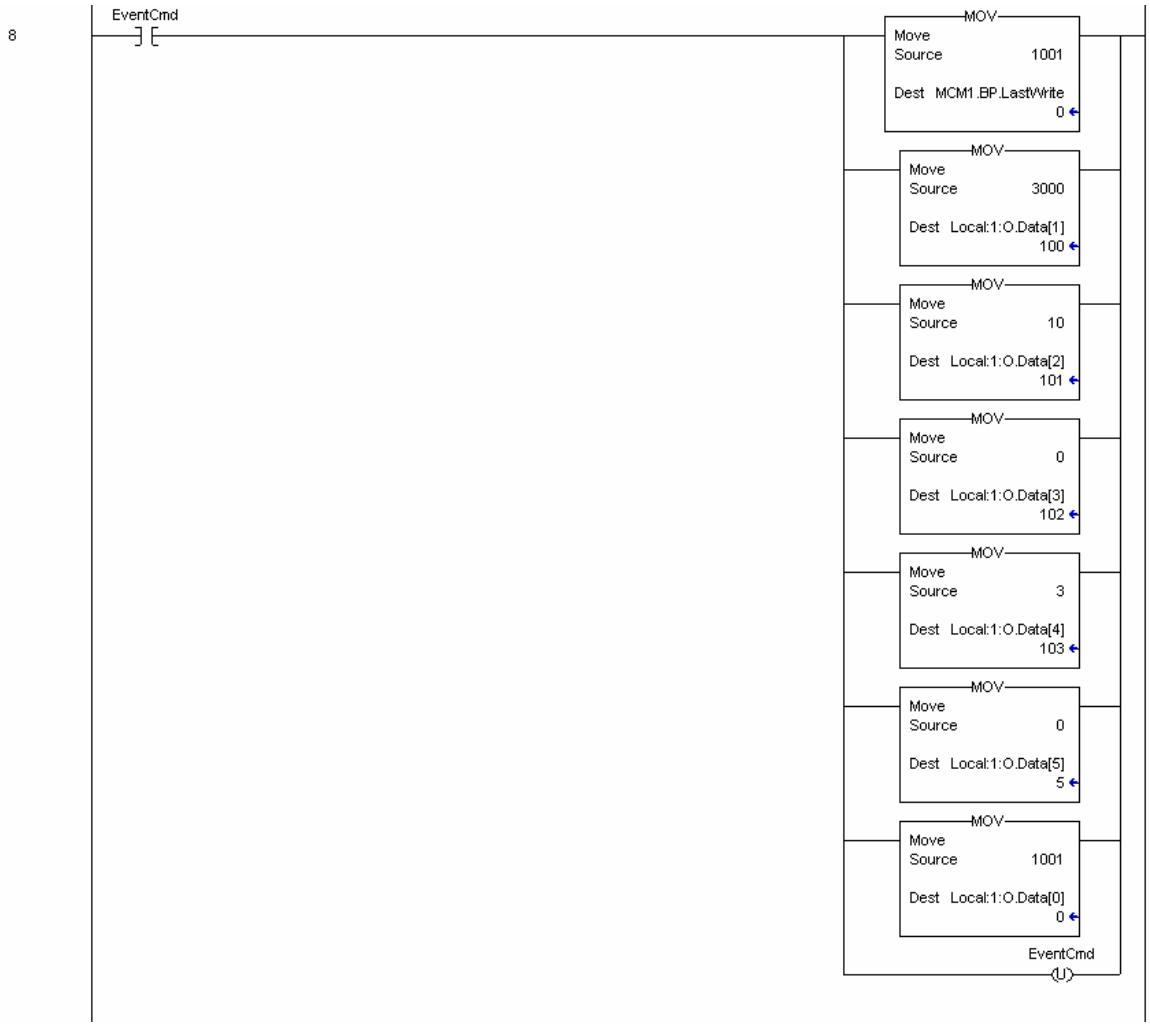

下面两行程序处理 Modbus 端口 2 上从站的状态/控制数据。



接下来的这行程序是个命令控制的样例。这个数据块从处理器发送到模块，以执行主站端口命令列表里的某个命令。


当 CmdControl 位被置位时，端口 1 主站命令（索引 0）被放到命令队列中执行。每个请求能够把最多 6 条命令从命令列表中放到命令队列中执行。

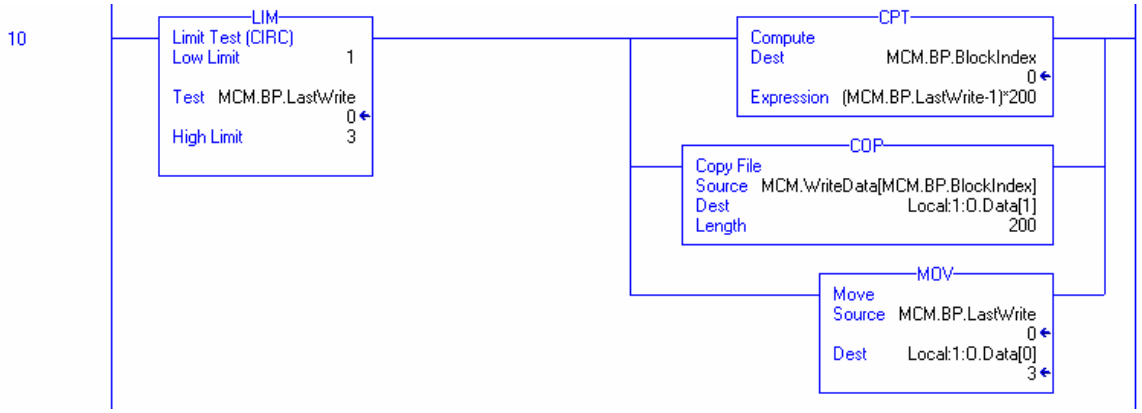
下面这行程序在端口 1 上发布了一条事件讯息（用户创建结构的讯息）。




当 EventCmd 位被置位，这行程序就开始执行。它把程序中包含的命令放到命令队列中以待执行。使用这种技术可以在没有创建主站命令列表的情况下，在端口上发布命令。或是要执行一些需要在特殊条件才发布的命令（例如每天或每星期执行一次重启动命令）。

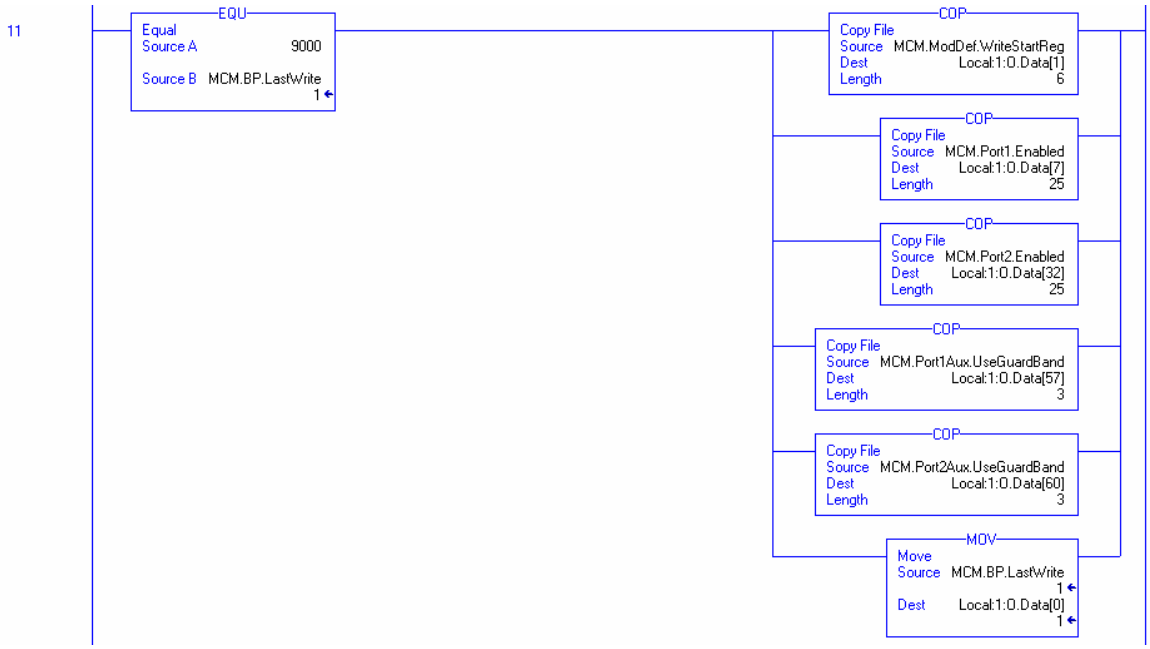
如果模块设置成无数据块传输，或只有一个数据块传输，那就需要一些特殊的处理过程。模块必须要监视模块输出镜像的变化，以获取新的数据。如果数值一直不变化，模块就不会处理这些数据。这就意味着，当模块传输少于 2 个数据块到处理器时可能会引发问题。为了解决这个问题，模块会在输入字里写-1 和 0。当把模块设置成零个写数据块，就会出现下面这样的数据块请求顺序：-1，0，-1，0。当把模块设置成 1 个写数据块，就会出现下面这样的的数据块请求顺序：1，0，1，0，1，0。下面的程序就是来处理这些情况的。

下面这行程序在梯形逻辑程序中是最重要的。它负责传送处理器数据到模块。每次能传送 200 个在处理器（**MCM.WriteData[]**）中的用户数据字到模块。

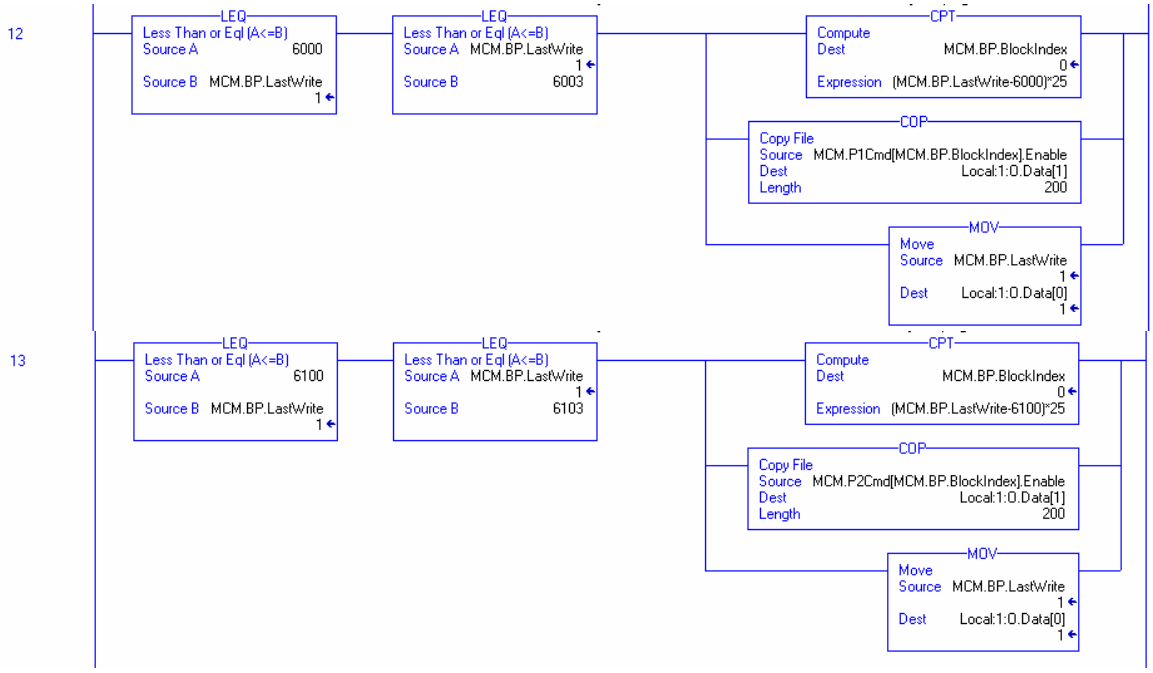



为了对模块进行设置，必须要从处理器数据表中把设置信息发送到模块。传送模块需要的所有信息需要几个数据块。模块的运行要求每个数据块都要经过编程处理。

第一个设置数据块的代码值是 9000。这个数据传送数据块尺寸信息（**MCM.ModDef**）和 Modbus 端口设置信息（**MCM.Port[]**）。 当模块启动时，这是模块请求的第一个数据集。




而模块请求的最后一组设置信息是每个端口的主站命令列表。这个列表以每次 25 个命令的速度传送到模块。传送命令列表到模块的梯形逻辑程序如下面显示：


12
LEQ
Less Than or Eql (A<=B)
Source A 6000
Source B MCM.BP.LastWrite
1
LEQ
Less Than or Eql (A<=B)
Source A MCM.BP.LastWrite
1
Source B 6003
CPT
Compute
Dest MCM.BP.BlockIndex
0
Expression (MCM.BP.LastWrite-6000)*25
COP
Copy File
Source MCM.P1Cmd[MCM.BP.BlockIndex].Enable
Dest Local:1:O.Data[1]
Length 200
MOV
Move
Source MCM.BP.LastWrite
1
Dest Local:1:O.Data[0]
1
13
LEQ
Less Than or Eql (A<=B)
Source A 6100
Source B MCM.BP.LastWrite
1
LEQ
Less Than or Eql (A<=B)
Source A MCM.BP.LastWrite
1
Source B 6103
CPT
Compute
Dest MCM.BP.BlockIndex
0
Expression (MCM.BP.LastWrite-6100)*25
COP
Copy File
Source MCM.P2Cmd[MCM.BP.BlockIndex].Enable
Dest Local:1:O.Data[1]
Length 200
MOV
Move
Source MCM.BP.LastWrite
1
Dest Local:1:O.Data[0]
1

# 5 诊断和纠错

模块为用户提供了 3 种方式获取诊断信息。1) 模块传送状态数值到 ControlLogix 处理器里定义好的数据文件内。2)连接终端模拟器，模块包含的所有数据都可以从设置/调试端口来查看。3) 模块的前端面板还设有 LED 状态指示灯，可以直接反映模块的状态。

下面的章节来解释如何从模块获取状态数值，以及模块的每个 LED 的含义。

## 5.1 从模块读取状态数据

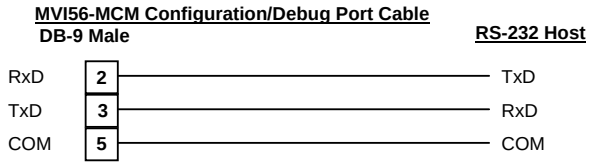
MVI56-MCM 模块会返回一个 29 个字的状态数据块，通过这个数据块用户可以判断模块的运行状态。这些数据位于模块的内部数据库 6670 到 6698 寄存器内，并同时存放在设置位置。而且，这些数据通过每个读数据块，连续不断的传送到 ControlLogix 处理器。状态数据对象的结构内容可以参考**模块设置** 章节。

### 5.1.1 硬件要求

能够连接设置/调试端口的硬件要求并不苛刻。一台具有标准串行口的个人电脑就可以满足要求。为达到理想的性能，至少要求达到下列条件：

80486 处理器（推荐 Pentium）
1 M 内存
至少一个可用的串行通讯口

此外，随模块还附带了一根 null-modem 电缆，用来连接您的 PC 和端口。模块附带的 RJ-45 到 DB-9 接头电缆的末端是一个 DB-9 针型接头。用电缆的 RJ-45 端连接 MVI56-MCM 模块端口 1（顶端端口）。连接电缆要求有如下接线方式：



### 5.1.2 软件要求

在您的个人电脑里，连接设置/调试端口的软件要求与操作系统有关。下列软件是经过测试并通过的：

| | |
|---|---|
| DOS | ProComm, PS-Term 和其它终端模拟程序 |
| Windows 3.1 | Terminal |
| Windows 95/98 | HyperTerminal 和 PS-Term |
| Windows NT | HyperTerminal |
| Linux | Minicom |

您的操作系统提供的任何 ASCII 终端模拟软件包，只要按照下面的内容进行设置都是可以使用的：

| | |
|---|---|
| 波特率 | 57,600 |
| 奇偶校验 y | 无 |
| 数据位 | 8 |
| 停止位 | 1 |
| 文件传输协议 | Zmodem |

### 5.1.3 端口使用

通过以下步骤来连接设置/调试端口：

1. 使用 null-modem 电缆连接您的电脑和模块的端口。
2. 运行您电脑中的终端模拟程序，按照**软件要求**中的要求来对通讯参数进行设置（57600K, N, 8, 1）。
3. 在电脑中输入 '?'。如果设置都正确，端口的菜单就会显示出来。

如果模块没有任何反应，请检查一下通讯设置和电缆。此外，还要确保您连接的是正确的计算机和模块端口。

### 5.1.4 菜单选项

在你的电脑上，只需要单击键盘按键就可以进入到设置/调试端口的每个选项里。端口上有一个主菜单和一些子菜单。如想查看当前的选项，就按 '?' 键。如果您在主菜单里，那么就会显示下面的菜单：

```
MODBUS MASTER/SLAVE COMMUNICATION MODULE (MVI56-MCM) MENU
  ?=Display Menu
  A=Data Analyzer
  B=Block Transfer Statistics
  C=Module Configuration
  D=Modbus Database View
  Master Command Errors : E=Port 1   F=Port 2
  Master Command List   : I=Port 1   J=Port 2
  Slave Status List     : O=Port 1   P=Port 2
  V=Version Information
  W=Warm Boot Module
  Y=Transfer Module Cfg to Processor
  Communication Status : 1=Port 1   2=Port 2
  Port Configuration   : 6=Port 1   7=Port 2

  Esc=Exit Program
```

如果不是这个菜单，就可以按 'M' 进入主菜单。所有用于设置/调试工作的所以工具都显示在主菜单内。下面的章节叙述每个选项的内容。

#### 5.1.4.1 A=数据分析器 Data Analyzer

选择这个选项就让程序进入分析菜单模式。这个模式可以显示模块生成和接收到的 Modbus 讯息帧。想要查看此模式下有用的模式，就按 '?' 键，就会有如下的菜单显示：

```
Data Analyzer Mode Selected

MODBUS DATA ANALYZER VIEW MENU
 ?=Display Menu
 1=Select Port 1
 2=Select Port 2
 5=1 mSec Ticks
 6=5 mSec Ticks
 7=10 mSec Ticks
 8=50 mSec Ticks
 9=100 mSec Ticks
 0=No mSec Ticks
 H=Hex Format
 A=ASCII Format
 B=Start
 S=Stop
 M=Main Menu

 Port = 1, Format=HEX, Tick=10
```

这个工具在判断模块的运行情况和每个端口上网络节点的状况方面有非常大的用处。最下面显示的参数是当前分析器的设置。下面解释每个菜单的内容。

##### 5.1.4.1.1 1=选择端口 1 Select Port 1

这个选项选择分析 Modbus 端口 1。这时分析器显示的数据就是和这个端口相关的数据。

#### 5.1.4.1.2 2=选择端口 2 Select Port 2

这个选项选择分析 Modbus 端口 2。这时分析器显示的数据就是和这个端口相关的数据。

#### 5.1.4.1.3 5=1 mSec 标记 1 mSec Ticks

这个选项将生成以 1 mSec 为时标的显示。这对判断通讯时标特性有所帮助。

#### 5.1.4.1.4 6=5 mSec 标记 5 mSec Ticks

这个选项将生成以 5 mSec 为时标的显示。这对判断通讯时标特性有所帮助。

#### 5.1.4.1.5 7=10 mSec 标记 10 mSec Ticks

这个选项将生成以 10 mSec 为时标的显示。这对判断通讯时标特性有所帮助。

#### 5.1.4.1.6 8=50 mSec 标记 50 mSec Ticks

这个选项将生成以 50 mSec 为时标的显示。这对判断通讯时标特性有所帮助。

#### 5.1.4.1.7 9=100 mSec 标记 100 mSec Ticks

这个选项将生成以 100 mSec 为时标的显示。这对判断通讯时标特性有所帮助。

#### 5.1.4.1.8 0=No mSec Ticks

这个选项将把时间标签从显示中去除。

#### 5.1.4.1.9 H= Hex 格式 Hex Format

这个选项让数据以十六进制格式显示。查看 Modbus RTU 协议讯息时，这个选项特别有用。

#### 5.1.4.1.10 A= ASCII 格式 ASCII Format

这个选项让数据以 ASCII 格式显示。查看 Modbus ASCII 协议讯息时，这个选项特别有用。

#### 5.1.4.1.11 B=开始 Start

这个选项启动数据分析器。按了这个按键后，所选择的端口上所有的发送和接收数据都会显示出来。下面是个例子：

```
<R+><01><03><00><00><00><0A><C5><CD><R->_TT_[01][03][14][00][00][00][00][00][00]
_TT_[00][00][00][00][00][00][00][00][00][00][00][00][00][A3][67]_TT_<R+><01>
<03><00><00><00><0A><C5><CD><R->_TT_[01][03][14][00][00][00][00][00][00][00][00]
[00][00][00][00][00][00][00]_TT_[00][00][00][00][00][A3][67]_TT_<R+><01><03><00>
<00><00><0A><C5><CD><R->_TT_[01][03][14][00][00][00][00][00][00][00][00][00][00]
[00][00][00][00][00]_TT_[00][00][00][00][00][A3][67]_TT_<R+><01><03><00><00><00>
<0A><C5><CD><R->_TT_[01][03][14][00][00][00][00][00][00]_TT_[00][00][00][00][00][00]
[00][00][00][00][00][00][00][00][00][A3][67]_TT_<R+><01><03><00><00><00><0A><C5>
<CD><R->_TT_[01][03][14][00][00][00][00][00][00]_TT_[00][00][00][00][00][00][00]
[00][00][00][00][00][00][00][A3][67]_TT_<R+><01><03><00><00><00><0A><C5><CD><R->
_TT_[01][03][14][00][00][00][00][00]_TT_[00][00][00][00][00][00][00][00][00][00]
[00][00][00][00][00][A3][67]_TT_<R+><01><03><00><00><00><0A><C5><CD><R->_TT_[01]
[03][14][00][00][00][00][00][00][00][00][00][00][00][00][00]_TT_[00][00]
[00][00][00][A3][67]_TT_<R+><01><03><00><00><00><0A><C5><CD><R->_TT_[01][03][14]
[00][00][00][00][00][00][00][00][00][00][00][00][00]_TT_[00][00][00][00]
[00][A3][67]_TT_<R+><01><03><00><00><00><0A><C5><CD><R->_TT_[01][03][14][00][00]
[00][00][00]_TT_[00][00][00][00][00][00][00][00][00][00][00][00][00][00][A3]
[67]_TT_<R+><01><03><00><00><00><0A><C5><CD><R->_TT_[01][03][14][00][00][00][00]
[00][00]_TT_[00][00][00][00][00][00][00][00][00][00][00][00][00][A3][67]_TT_
```

下面的表格解释显示用到的特殊字符的含义：

| | |
|---|---|
| [ ] | 此符号内的内容是模块接收到的数据。 |
| < > | 此符号内的内容是模块发送出去的数据。 |
| <R+> | 当端口上的 RTS 信号由低到高电平时，显示中会插入这个字符。 |
| <R-> | 当端口上的 RTS 信号由高到低电平时，显示中会插入这个字符。 |
| <CS> | 当识别到 CTS 信号到高电平时，显示中会插入这个字符。 |
| _TT_ | 当时标到达时，显示中会插入这些字符。这个参数是用户定义的。 |

#### 5.1.4.1.12 S=停止 Stop

这个选项停止数据分析器。使用这个选项来暂停数据显示，方便数据分析。重新启动分析器，可以按 'B' 键。

**警告 –** 在分析器模式下，程序的执行速度会减缓。请仅当需要做纠错工作时，才使用这个工具。当从模块中推出时，请禁止分析器，好让模块进入正常模式。

#### 5.1.4.1.13 M =主菜单 Main Menu

这个菜单选项用来返回主菜单模式。

### 5.1.4.2 B=块传输统计 Block Transfer Statistics

在这个菜单选项里，可以查看到背板数据传输的设置和统计数字。进入这个选项后，可以看到类似下面的显示。以 1 秒钟的间隔选择这个选项，则可以来判断每秒传输的数据块数量。

```
BACKPLANE STATISTICS:

DATA TRANSFER CONFIGURATION:
  WRITE DATA TRANSFER:
        Start : 0       Count : 400     Max Blocks : 2       Last : 2
  READ DATA TRANSFER:
        Start : 0       Count : 600     Max Blocks : 3       Last : 2
  BLOCK COUNTS:
        Retry : 20      Failed: 0       Fail Cnt: 0
        Read  : 43009   Write : 43021   Parsing : 43008
        Error : 59833   Event : 0       Command : 0
```

### 5.1.4.3 C=模块设置 Module Configuration

这个选项显示了 MVI56-MCM 模块的常规模块设置信息。在选择这个选项后，会显示如下的画面。

```
MODULE CONFIGURATION:

MVI56-MCM, ProSoft Technology, Inc.

  DATABASE:
        Err/Stat Blk Pointer : 600
  BLOCK TRANSFER:
        READ  -- Start : 0        Count : 600      Max : 3
        WRITE -- Start : 0        Count : 400      Max : 2
        FAIL COUNT     : 20
```

### 5.1.4.4 D=查看 Modbus 数据库 Modbus Database View

选择这个选项后可以进入数据库查看菜单模式。此模式操作显示内部数据库数值。按“?”键查看本模式下的菜单，如下面的画面所示。

```
MODBUS DATABASE VIEW MENU
 ?=Display Menu
 0-6=Display 0-6000
 S=Show Again
 -=Back 5 Pages
 P=Previous Page
 +=Skip 5 Pages
 N=Next Page
 D=Decimal Display
 H=Hexadecimal Display
 F=Float Display
 A=ASCII Display
 M=Main Menu
```

使用菜单选项能够查看模块数据库的所有数据。下面讨论菜单内的每个选项的内容。

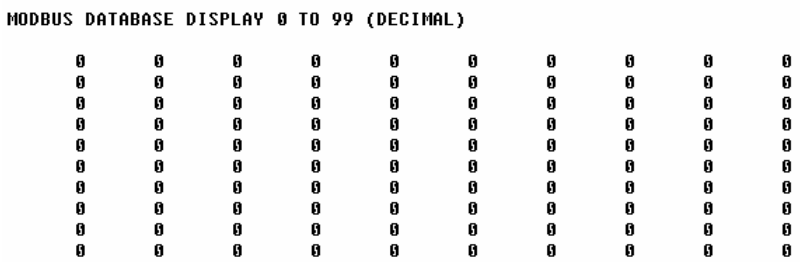
### 5.1.4.5 0-9 寄存器页码代表 0-9000 0-9 Register Pages 0-9000

这个菜单选项能跳转到数据库特定的寄存器组并显示这些数据。按键功能如下表所示。

| 按键 | 功能 |
|---|---|
| 0 | 显示寄存器 0 到 99 |
| 1 | 显示寄存器 1000 到 1099 |
| 2 | 显示寄存器 2000 到 2099 |
| 3 | 显示寄存器 3000 到 3099 |
| 4 | 显示寄存器 4000 到 4099 |
| 5 | 显示寄存器 5000 到 5099 |
| 6 | 显示寄存器 6000 到 6099 |
| 7 | 显示寄存器 7000 到 7099 |
| 8 | 显示寄存器 8000 到 8099 |
| 9 | 显示寄存器 9000 到 9099 |

### 5.1.4.6 S=再次显示 Show Again

这个菜单选项显示数据库当前页的 100 个寄存器。下面的例子就是数据库显示的形式：




#### 5.1.4.6.1 - =回退 5 页 Back 5 Pages

这个菜单选项用来回退跳过 500 个寄存器，显示之前的数据

#### 5.1.4.6.2 P =前页 Previous Page

这个菜单选项选择并显示前 100 个寄存器的数据。

#### 5.1.4.6.3 + =跳过 5 页 Skip 5 Pages

这个菜单选项跳过 500 个寄存器，显示当前新页面的数据。

#### 5.1.4.6.4 N =下页 Next Page

这个菜单选项选择并显示下 100 个寄存器的数据。

#### 5.1.4.6.5 D =十进制显示 Decimal Display

这个菜单选项使当前页面的数据以十进制格式显示。

#### 5.1.4.6.6 H = Hexadecimal Display

这个菜单选项使当前页面的数据以十六进制格式显示。

#### 5.1.4.6.7 F =浮点数显示 Float Display

这个菜单选项使当前页面的数据以浮点数格式显示。模块程序会假设认为浮点数数据都是以偶数寄存器为数据边界排列的。如果浮点数数值并不是这样排列的，那这些数据就会显示的不正确。

#### 5.1.4.6.8 A = ASCII 显示 ASCII Display

这个菜单选项使当前页面的数据以 ASCII 格式显示。这个选项对于包含 ASCII 数据的数据库区域很有用处。

#### 5.1.4.6.9 M =主菜单 Main Menu

使用这个菜单选项可以回到主菜单模式。

### 5.1.4.7 E 和 F=主站命令错误（端口 1 和 2）
### E and F=Master Command Errors (Ports 1 and 2)

选择这些菜单则使程序进入指定端口的主站命令列表错误菜单模式。这个模式显示多页面的主站命令列表错误/状态数据。按“?”键可以查看本模式下的菜单选项。如下的菜单会显示出来。

```
COMMAND ERROR LIST MENU (MASTER Port 1)
 ?=Display Menu
 S=Show Again
 -=Back 2 Pages
 P=Previous Page
 +=Skip 2 Pages
 N=Next Page
 D=Decimal Display
 H=Hexadecimal Display
 M=Main Menu
```

下面的章节叙述每个菜单选项的内容。

#### 5.1.4.7.1 S =再次显示 Show Again

这个选项显示了当前页面的主站命令错误/状态数据。选择这个选项后，会有如下的显示出现。

```
COMMAND ERROR LIST FOR PORT 1, COMMANDS 0 TO 19 (DECIMAL)

    0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0
```

屏幕上显示的每个数值对应于主站命令列表响应命令的错误/状态代码。显示的代码含义以及完整列表参见 **模块设置**章节。

#### 5.1.4.7.2 - =回退 2 页 Back 2 Pages

这个选项回退 20 个命令并显示数据。

#### 5.1.4.7.3 P =前页 Previous Page

这个选项显示前页面的数据。

#### 5.1.4.7.4 + =跳过 2 页 Skip 2 Pages

这个选项跳过下 20 条命令，并显示数据。

#### 5.1.4.7.5 N =下页 Next Page

这个选项显示下页的主站命令列表错误/状态数据。

#### 5.1.4.7.6 D =十进制显示 Decimal Display

这个选项使当前数据以十进制格式显示。

#### 5.1.4.7.7 H = 十六进制显示 Hexadecimal Display

这个选项使错误/状态数据以十六进制格式显示。

#### 5.1.4.7.8 M = 主菜单 Main Menu

这个选项使程序回到主菜单模式

### 5.1.4.8 I 和 J=主站命令列表（端口 1 和 2） I and J=Master Command List (Ports 1 and 2)

选择这些菜单则使程序进入指定端口的主站命令列表菜单模式。这个模式显示多页面的主站命令列表数据。按“?”键可以查看本模式下的菜单选项。如下的菜单会显示出来。

```
MASTER COMMAND LIST MENU (Port 1)
 ?=Display Menu
 S=Show Again
 -=Back 5 Pages
 P=Previous Page
 +=Skip 5 Pages
 N=Next Page
 M=Main Menu
```

下面的章节叙述每个菜单选项的内容：

#### 5.1.4.8.1 S =再次显示 Show Again

这个选项显示当前页面的主站命令。每个页面显示 10 条指令，如下图所示：

```
COMMAND LIST FOR PORT 1-- COMMANDS 0 TO 9

EN MBREG POLLINT COUNT SWAP NODE FUNC ADDRS LASTERR
 1   400       0    10    0    1    3     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
 0     0       0     0    0    0    0     0  0X0000
```

#### 5.1.4.8.2 - =回退 5 页 Back 5 Pages

这个菜单选项将会显示回退 50 条命令的主站命令列表。

#### 5.1.4.8.3 P =前页 Previous Page

这个菜单选项显示前页的主站命令列表数据。

#### 5.1.4.8.4 + =跳过 5 页 Skip 5 Pages

这个菜单选项在跳过下 50 条命令后，显示主站命令列表数据。

#### 5.1.4.8.5 N =下页 Next Page

这个菜单选项显示下页的主站命令列表数据。

#### 5.1.4.8.6 M =主菜单 Main Menu

这个选项返回主菜单运行模式。

### 5.1.4.9 O and P=从站状态列表（端口 1 和 2） Slave Status List (Port 1 and 2)

选择这些菜单选项可以显示指定端口的 256 个从站状态数值。显示数据的定义如下：0 = 从站未使用， 1 = 从站正在被轮询， 2 = 从站被挂起，3 = 从站禁止。

```
Main Menu Selected
SLAVE STATUS LIST FOR PORT 1
0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

### 5.1.4.10 V=版本信息 Version Information

这个选项显示模块当前使用的软件版本和其它重要数值。在选择这个选项后，会显示如下的画面。

```
VERSION INFORMATION:

  MODBUS MASTER/SLAVE COMMUNICATION MODULE (MVI56-MCM)
  (c) 1999, ProSoft Technology, Inc.

  PRODUCT NAME CODE          : MCM
  SOFTWARE REVISION LEVEL    : 0.00
  OPERATING SYSTEM REVISION : 0100
  RUN NUMBER                 : 2602
  PROGRAM SCAN COUNTER       : 26961
```

在对产品的技术支持时，需要提供这些信息。显示画面底部的数据对于判断模块运行是非常重要的。模块程序每次扫描结束，**程序扫描计数（Program Scan Counter）** 数值都会增加。每秒按一次“V”键，用户就可以通过这个数值判断程序执行的频率。

### 5.1.4.11 W=热启动模块 Warm Boot Module

当模块需要热启动操作时就可以使用这个选项。在 ControlLogix 处理器控制器标签数据区内对模块设置进行修改后，为应用修改后的设置，通常需要对模块进行一次热启动。选择这个选项，会显示如下画面。

```
Press 'Y' key to confirm warm boot!

Warm booting module....

Reloading Program Values....

Read Configuration...
Read Port 1 Command Configuration...
MODBUS MASTER/SLAVE COMMUNICATION MODULE (MVI56-MCM)
(c) 1999, ProSoft Technology, Inc.

     PRODUCT NAME CODE         : MCM
     SOFTWARE REVISION LEVEL   : 0.00
     OPERATING SYSTEM REVISION : 0100
     RUN NUMBER                : 2602

Press ? for menu help.
```

### 5.1.4.12 Y=传送模块配置到处理器 Transfer Module Cfg to Processor

这个选项把当前设置数据传送到 ControlLogix 处理器。处理器里要有相应的程序才能成功的实现这个操作。选择后，如下画面会显示表明操作成功：

```
Press 'Y' key to confirm transfer option!

Sending configuration to processor....

Send complete....Error code = 0.

Warm booting module....

Reloading Program Values....

Read Configuration...
Read Port 1 Command Configuration...
MODBUS MASTER/SLAVE COMMUNICATION MODULE (MVI56-MCM)
(c) 1999, ProSoft Technology, Inc.

     PRODUCT NAME CODE         : MCM
     SOFTWARE REVISION LEVEL   : 0.00
     OPERATING SYSTEM REVISION : 0100
     RUN NUMBER                : 2602

Press ? for menu help.
```

如果操作不成功，会返回一个错误代码。错误代码解释如下：

| 代码 | 描述 |
|---|---|
| 0 | 传送成功 |
| -1 | 传送模块设置数据出错 ( 数据块 – 9000) |
| -2 | 传送端口 1 主站命令列表出错 ( 数据块-6000 to -6003) |
| -3 | 传送端口 2 主站命令列表出错(数据块-6100 to -6103) |

在数据成功传送后，模块会执行一次热启动操作来读取新的数值。

### 5.1.4.13 1 and 2=通讯状态（端口 1 和 2）Communication Status (Ports 1 and 2)

这些选项显示指定 Modbus 端口的通讯状态和统计数字。这些信息可用于诊断网络问题。选择后，会显示如下信息。

```
PORT 1 MODBUS STATUS:
  Enabled : Y
  Retries : 0       Cur Cmd : 0       State   : 100
  ComState: 0

  Number of Command Requests: 5431
  Number of Cmd Responses   : 5430
  Number of Command Errors  : 0
  Number of Requests        : 5430
  Number of Responses       : 5430
  Number of Errors Received : 0
  Number of Errors Sent     : 0
```

### 5.1.4.14 6 and 7=端口设置（端口 1 和 2） Port Configuration (Ports 1 and 2)

这些选项显示选择的 Modbus 端口的设置信息。选择后，会显示如下信息。

```
PORT 1 CONFIGURATION:
  Enabled    : Y     Port Type : (0) - MASTER
  SLAVE SETUP:
      Modbus Slave ID: 0      Pass-Through = DISABLED
      Offsets:
        BitIn: 0        WordIn: 0        Output: 0        Holding: 0
      Floating-point Data:
        Flag: N   Start: 0        Offset: 0
      Route Count : 0
      Function 99 Offset : 0
      Use Packet Gap : N   Packet Gap Delay : 0
  MASTER SETUP:
      Command Count   : 3        Cmd Delay: 0        Cmd Offset : 350
      Response Timeout: 1000     Retries  : 0        Delay Count: 0
  COMMUNICATION PARAMETERS:
      Protocol: 1 (Modbus ASCII)
      Baud: 38400    Parity: NONE    Databits: 8   Stopbits: 1
      RTS On: 0       RTS Off: 0       Use CTS Line: N
```

### 5.1.4.15 Esc=退出程序 Exit Program

这个选项将退出程序并显示操作系统提示符。此操作仅当 ProSoft 技术支持组要求使用时。如果您选择这个操作，模块会停止运作。Modbus 端口和模块之间以及模块和 ControlLogix 处理器之间的数据传送都会停止。这会扰乱当前正运行的处理。

## 5.2 LED 状态指示

LED 会显示模块的运行状态：

| ProSoft 模块 | 颜色 | 状态 | 指示 |
|---|---|---|---|
| P1 | 绿色 | On | 使用设置/调试端口时，数据正在模块和远程终端之间传输。 |
| | | Off | 设置/调试端口上没有数据传输。 |
| P2 | 绿色 | On | 模块的 Modbus 端口 2 和 Modbus 网络之间正在数据传送。 |
| | | Off | 端口上没有数据传输。 |
| P3 | 绿色 | On | 模块的 Modbus 端口 3 和 Modbus 网络之间正在数据传送。. |
| | | Off | 端口上没有数据传输。 |
| APP | 黄色 | On | MVI56-MCM 正常运行。 |
| | | Off | MVI56-MCM 模块程序发现其 Modbus 端口上存在通讯错误。 |
| BP ACT | 黄色 | On | 当模块对背板执行写操作时，LED 会亮。 |
| | | Off | 当模块对背板执行读操作时，LED 熄灭。正常运行时，LED 会迅速闪烁。 |
| | | | |
| OK | 红色/绿色 | Off | 模块没有得电，并没有牢靠的插在槽架里。 |
| | | Green | 模块工作正常。. |
| | | Red | 程序检测到错误或正在被设置。如果 LED 保持红色 10 秒钟以上，程序很可能已经停止。从槽架里拔出模块再重新插一次，这可以重启动模块程序。 |
| BAT | 红色 | Off | 电池电压正常并发挥作用。 |
| | | On | 电池电压低或没有电池。请更换模块的电池。 |

在模块设置进行中，OK 灯是红色，APP 和 BP ACT 灯会点亮。如果这些指示灯长时间锁死在这个模式下，请检查设置错误字和设置请求数据块。数据块结构如下表所示：

| 偏移 | 描述 | 长度 |
|---|---|---|
| 0 | 保留 | 1 |
| 1 | 9000 | 1 |
| 2 | 模块设置错误 | 1 |
| 3 | 端口 1 设置错误 | 1 |
| 4 | 端口 2 设置错误 | 1 |
| 5 – 248 | 空 | 244 |
| 249 | -2 或 –3 | 1 |

每个设置字的位描述如下表。模块的设置错误字有如下定义：

| 位 | 描述 | 数值 |
|---|---|---|
| 0 | 读数据块开始数值大于数据库尺寸。 | 0x0001 |
| 1 | 读数据块开始数值小于 0。 | 0x0002 |
| 2 | 读数据块个数小于 0。 | 0x0004 |
| 3 | 读数据块个数 + 开始数值大于数据库尺寸。 | 0x0008 |
| 4 | 写数据块开始数值大于数据库尺寸。 | 0x0010 |
| 5 | 写数据块开始数值小于 0。 | 0x0020 |
| 6 | 写数据块个数小于 0。 | 0x0040 |
| 7 | 写数据块个数 + 开始数值大于数据库尺寸。 | 0x0080 |
| 8 | | 0x0100 |
| 9 | | 0x0200 |
| 10 | | 0x0400 |
| 11 | | 0x0800 |
| 12 | | 0x1000 |
| 13 | | 0x2000 |
| 14 | | 0x4000 |
| 15 | | 0x8000 |

端口设置错误字有如下定义：

| 位 | 描述 | 数值 |
|---|---|---|
| 0 | 类型编码不是合法数值。输入数值 0（主站）或 1（从站）。 | 0x0001 |
| 1 | 浮点数标记（float flag）参数数值非法。 | 0x0002 |
| 2 | 浮点数开始（float start）参数数值非法。 | 0x0004 |
| 3 | 浮点数偏移（float offset）参数数值非法。 | 0x0008 |
| 4 | 协议（Protocol）参数数值非法。 | 0x0010 |
| 5 | 波特率（Baud rate）参数数值非法。 | 0x0020 |
| 6 | 奇偶校验（Parity）参数数值非法。 | 0x0040 |
| 7 | 数据位（Data bits）参数数值非法。 | 0x0080 |
| 8 | 停止位（Stop bits）参数数值非法。 | 0x0100 |
| 9 | 从站 ID（Slave ID）参数数值非法。 | 0x0200 |
| 10 | Input bit 或 word, output word 和/或 holding register 偏移参数数值非法。 | 0x0400 |
| 11 | 命令个数参数数值非法。 | 0x0800 |
| 12 | 空 | 0x1000 |
| 13 | 空 | 0x2000 |
| 14 | 空 | 0x4000 |
| 15 | 空 | 0x8000 |

为使模块正常工作，请纠正设置中任何非法的数值。如果设置包含非法的参数集，设置字中所有的位都会被清除。这并不意味用户应用的设置有效。请确保每个参数都有正确的设置。

如果 APP, BP ACT 和 OK LED 以一秒钟的频率闪烁，请致电 ProSoft Technology, Inc. 支持。此时模块发生严重的错误，需要送回 ProSoft。