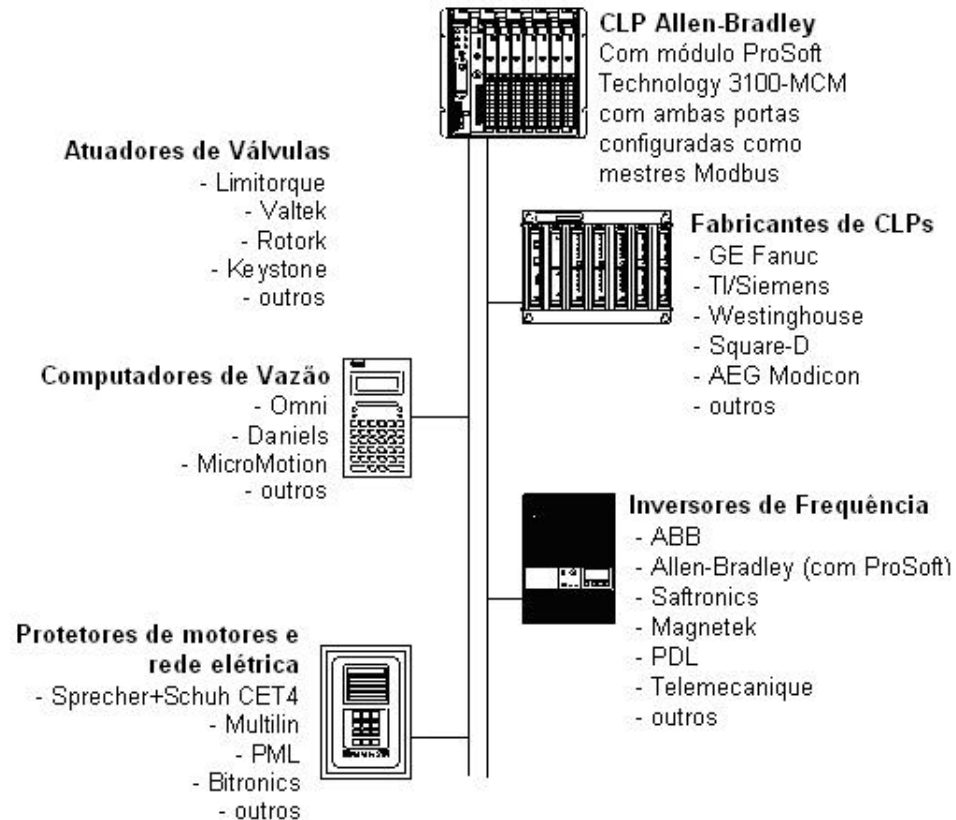


**3100/3150-MCM**  
**Módulo de Comunicação Modbus**

Revisão 2  
 Abril de 1997



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

**MANUAL DE USUÁRIO**

ProSoft Technology  
 9801 Camino Media, Suite 105  
 Bakersfield CA 93311  
 (661)664-7208  
 (661)664-7233 (fax)  
 prosoft@prosoft-technology.com  
 www.prosoft-technology.com

**AVISO IMPORTANTE**

Uma aplicação bem sucedida da placa MCM requer um razoável domínio de operação de funcionamento do hardware Allen\_Bradley PLC/ SLC, e da aplicação na qual a combinação será usada. Por isto, é importante que os responsáveis pela implementação do MCM garantam que a combinação atenda os requisitos da aplicação sem expor pessoal ou equipamentos a condições de operação inseguras ou não apropriadas.

Este manual pretende ajudar o usuário. Todo esforço foi feito de modo a garantir que a informação provida seja precisa e proporcione pleno entendimento dos requisitos de instalação do produto. Para garantir um pleno entendimento da operação do produto, o usuário deve ler com atenção toda documentação Allen\_Bradley aplicável à operação do hardware A-B.

Em nenhuma condição, a ProSoft\_Technology, Inc será passível de ser responsabilizada por danos diretos ou indiretos resultantes de uso ou aplicação de produto MCM.

A reprodução do conteúdo deste manual, integral ou parcialmente, é expressamente proibida sem autorização por escrito da própria ProSoft\_Technology, Inc.

As informações contidas neste manual poderão ser modificadas ou alteradas sem prévio aviso e não representam um compromisso da parte da ProSoft\_Technology, Inc e quaisquer melhoramentos ou mudanças neste manual ou no produto poderão ser feitos em qualquer tempo. Estas mudanças serão feitas periodicamente para corrigir imprecisões técnicas ou erros tipográficos.

## **Novos itens na revisão 2**

A revisão 2 do produto MCM representa o primeiro grande upgrade introduzido no produto. Foram incorporados muitos itens novos, alguns dos quais sugeridos por nossos atuais clientes e outros que resultaram de nosso aprendizado em como os clientes usam nossos produtos. A seguir apresentamos uma descrição de alguns destes novos itens.

### **Driver Modbus Mestre**

#### **Bits de Status de Comando**

Adicionados à informação que retornou para lógica Ladder foram providos os bits de comando "Done" e "Error". Estes bits permitem à lógica Ladder rastrear cada execução de comando Modbus Mestre e seu sucesso ou falha. Em conexão com o novo Modo de Controle de Comando (ver abaixo), o programa Ladder agora é capaz de controlar execução de cada comando.

#### **Tempo de Varredura de Comando**

Cada comando pode agora ser configurado com um "Tempo de Varredura" para efetivamente controlar a frequência de execução. O timer tem resolução de um (1) segundo, e cada comando pode ser configurado até 65535 segundos.

#### **Modo de Controle de Comando**

O modo de controle de comando é provavelmente o item mais requisitado. Neste modo, o driver Modbus Mestre somente executa comandos na Lista de Comando quando recebe o bit "Habilitado" da lógica Ladder. O módulo retorna os bits "Realizado" e "Erro" como parte da informação de status de comando regular que retorna para o Ladder (ver abaixo) permitindo que os comandos sejam enviados somente uma vez quando desejados.

#### **Comandos de Escrita Iniciados por Evento**

O driver Modbus Mestre no módulo MCM suporta execução de **Comandos de Escrita Iniciados por Evento**. Este tipo de suporte provê o programa lógico Ladder com outro método além dos comandos de Escrita Condicional (documentados no manual) e Modo de Controle de Comando para enviar comandos Modbus de escrita a um Escravo (FC 5, 6, 15, 16).

### **Driver Modbus Escravo**

#### **Modo Pass\_Through**

O modo Pass\_Through permite a uma porta Modbus Escravo transferir comandos de escrita recebidos de um host diretamente através do barramento para manuseio pela lógica Ladder. Embora este item requeira mais lógica Ladder para implementar uma solução, há certas situações onde esta funcionalidade pode ser útil. Algumas destas situações incluem:

- 1- Quando Escravo precisa saber quando tem que ser gravado;
- 2- Quando a aceitação de dados pode requerer algum condicionamento;
- 3- Quando os registros de dados do host devem se sobrepor ao espaço de registro de leitura.

#### **Modo Routing**

O modo Routing permite à porta Modbus Escravo rotear comandos recebidos a partir de um host para a porta Mestre. Uma lista de até seis (6) endereços de Escravo roteados pode se alimentada na tabela de configuração. Sempre que um comando de host (seja por leitura ou escrita) for recebido na porta Escravo que corresponde a um de seis endereços, o comando é roteado fora da porta Mestre e a resposta é roteada de volta para a porta Escravo.

### Guia de Implementação de Início Rápido

A integração do módulo MCM em uma aplicação PLC ou SLS será fácil se uma série de etapas forem seguidas. Para ajudar usuários iniciantes a rapidamente se tornarem operacionais, preparamos um guia de implementação etapa por etapa.



#### Usuários Iniciantes

Embora as etapas a seguir pretendam ajudá-lo a implementar o módulo, recomendamos que você tente experimentar o exemplo lógico provido no disco com o módulo ou disponível em nosso site FTP antes de executar sua aplicação. Esta etapa proporciona uma visão de como o módulo trabalha antes de tomar decisões que afetem o sucesso a longo prazo da instalação.

Começando com um programa de lógica Ladder providos no disco com MCM complete as seguintes etapas:

Se entrar manualmente a lógica Ladder no SLS, lembre-se de:

configure slot com o módulo 1746-BAS no modo 5/02;

certifique-se da entrada de bits de Transferência "Habilitado" e "Realizado", como mostrado no exemplo lógico:

**a-** Edite lógica Ladder provida no disco conforme necessário para aplicação (ver Seção 3.0)

Verifique localização rack e slot no programa;

Modifique endereços de instrução Ladder conforme necessário

**b-** Ajuste os parâmetros de configuração de comunicação (ver Seção 4.2)

determine cada requisito de configuração de comunicação da porta:

Mestre/Escravo; Paridade; Bits de Parada; Taxa de Bits; Requisito de atraso RTS;

identifique requisitos de mapeamento de memória;

set Dados de Leitura, Dados de Escrita; e Parâmetros de Quantidade de Blocos de Comando;

set ponteiros de Tabela de Erro Escravo/Mestre para aplicação;

**c-** Ajuste a Lista de Comando se estiver configurando Mestre (ver Seção 4.4)

certifique-se de rever mapa de dispositivo Escravo para prover um mapa de memória mais efetivo.

**d-** Identifique os requisitos de jumper de módulo (ver apêndice D);

**e-** Prepare os cabos de comunicação (ver Seção 8). Certifique-se que, não importando qual tipo de conexão usada, o jumper esteja no lugar para atender a um sinal GTS. Normalmente, este sinal será jumpeado com RTS;

**f-** Coloque processador no modo run;

**g-** Monitore a tabela de dados para valores de Status de erro Escravo/Mestre.

#### **Documentos de Teste "ProSoft Tested"**

Dentro dos esforços de manter atualizado nosso programa "ProSoft Tested", mantemos uma lista crescente de dispositivos sabidamente interfaceados com nosso módulo. Adicionalmente, também temos documentado diversos dispositivos testados. Para acessar esta informação, por favor, visite nosso site, e proceda da seguinte forma:

<http://www.prosoft-technology.com>

selecione "Web Site Index"

selecione "MCM Connectivity Listing"

selecione "Test Document" para o produto desejado

**ÍNDICE**

1	<b>Visão Geral Funcional</b>
1.1	Geral
1.2	Visão Geral do Hardware
1.3	Conceitos Gerais
1.3.1	Ligação do Módulo e Reiniciação
1.3.2	Lógica Principal de Loop
1.3.3	Espaço de dados no módulo
1.3.4	Processo de transferência de Dados de Barramento
1.3.5	Intertravamento das transferências de Bloco
1.3.6	Configuração de Processador SLC
1.4	Fluxo de Dados
1.4.1	Conceitos Gerais
1.4.2	Leitura de dados a partir do módulo
1.4.3	Escrita de dados no módulo
1.4.4	Driver de Porta Mestre
1.4.5	Driver de Porta Escravo
1.4.6	Porta Escravo - Modo Normal
1.4.7	Porta Escravo - Modo Pass_Through
1.4.8	Porta Escravo - Modo Routing
1.5	Endereçamento de ModBus
1.5.1	Conceitos de Endereçamento de Modbus
1.5.2	Suporte MCM de Funcionalidade Modbus
1.5.3	Mapeamento de endereços ModBus para Endereços de Dados Ladder
2	<b>Avançando - Etapa por Etapa</b>
3	<b>Visão Geral da Lógica Ladder</b>
3.1	Visão Geral Operacional
3.2	Lógica Ladder
4	<b>Escrita no Módulo</b>
4.1	Transferência de bloco para módulo
4.2	Configuração de comunicação (ID DO BLOCO BTW 255)
4.3	Escrita na memória de dados de módulo (ID DO BLOCO BTW 0-79)
4.3.1	Lógica Ladder para gravar dados no módulo
4.3.2	Estrutura de Dados de Transferência de Blocos
4.4	Configuração da Lista de Comando- Modo Mestre (ID DO BLOCO BTW 80-99)
4.4.1	Lógica Ladder da Lista de Comando
4.4.2	Estrutura da Lista de Comando
4.4.3	Editando a Lista de Comando
4.5	Modo de Controle de Comando - Modo Mestre
4.5.1	Estrutura de Bloco BTW
4.5.2	Controlando os Comandos
4.5.3	Lista de Comando Exemplar
4.6	Comandos Iniciados por Evento- Modo Mestre (ID DO BLOCO BTW 10-119)
4.6.1	Lógica Ladder
4.6.2	Estrutura de Bloco BTW
5	<b>Leitura do Módulo</b>
5.1	Transferindo dados a partir do módulo (ID DO BLOCO BTR 0-79)
5.1.1	Estrutura de Bloco de Dados de Leitura
5.1.2	Movendo os dados do módulo para o processador
5.1.3	Lógica Ladder para ler Dados de Módulo
5.1.4	Tabela de Código de Extremidade Escravo
5.1.5	Tabela de código de Erro Mestre
5.1.6	Códigos de Status de Erro
5.2	Modo de Pass_Through - Modo Escravo (ID DO BLOCO BTR 256-259)
5.2.1	Estrutura de Bloco
5.2.2	Recebendo escritas de registros (ID DO BLOCO BTR 256 e 257)
5.2.3	Recebendo Escritas de Bit único (ID DO BLOCO BTR 258)
5.2.4	Recebendo Escritas de múltiplos bits (ID DO BLOCO BTR 259)
5.3	Decodificando Bits de Comando "Done" e Error"- Modo Mestre
5.3.1	Estrutura de Bloco
5.3.2	Lógica Ladder
6	<b>Configuração de Comando ModBus</b>
6.1	Comandos Modbus
6.2	Suporte de Ponto Flutuante
6.2.1	Suporte de Ponto Flutuante ENRON
7	<b>Diagnósticos e Solução de Problemas</b>
7.1	Ponteiros LED Plataforma PLC 3100
7.2	Ponteiros LED Plataforma SLC 3100
7.3	Solução de Problemas - Geral
8	<b>Diagramas de Conexão de Cabos</b>

A	Suporte, Manutenção, e Garantia
B	Especificações do Produto
C	Especificação de Protocolo Modbus
D	Configurações de Jumpers
E	Histórico de Revisão de Produto
F	Uso da Quantidade de Blocos de Leitura, Escrita, e Comando
G	Lógica Ladder de Exemplo

*(Programas Ladder de exemplo de referência separados no manual de exemplo de aplicação)*

## 1 Visão Geral Funcional

Esta seção pretende dar ao leitor uma visão geral dos conceitos operacionais do módulo MCM. Os detalhes associados à lógica Ladder e a transferência de dados no barramento serão cobertos em seções posteriores e no apêndice.

### 1.1 Geral

Os produtos MCM são módulos residentes em racks de slot único projetados de modo a prover uma interface de comunicação Modbus integrada de modo compacto às plataformas Allen\_Bradley 1771 e 1746I/O. O produto deverá prover suporte para os seguintes processadores:

3100-MCM para Plataforma 1771  
 família PLC 5  
 família PLC 2  
 família PLC 3

3150 MCM para plataforma 1746  
 SLC 5/02, 5/03, 5/04

O módulo deverá operar no rack local com processador ou pode ser instalado em Rack remoto usando Comunicação I/O Remota para se conectar aos racks, no caso do PLC, ou pode ser colocado em uma extensão de rack no caso do SLC.

As formas na qual o produto pode ser fornecido podem ser vistas abaixo:



3100 Module  
1771 Platform

Módulo 3100  
Plataforma 1771



3150 Module  
1746 Platform

Módulo 3150  
Plataforma 1746

### 1.2 Visão Geral do Hardware

Os projetos dos módulos MCM para duas plataformas de hardware são muito parecidos. A discussão a seguir, a menos que especificado de forma diferente, se aplica a ambas plataformas, 3100 e 3150. A figura abaixo mostra os componentes funcionais dos módulos.

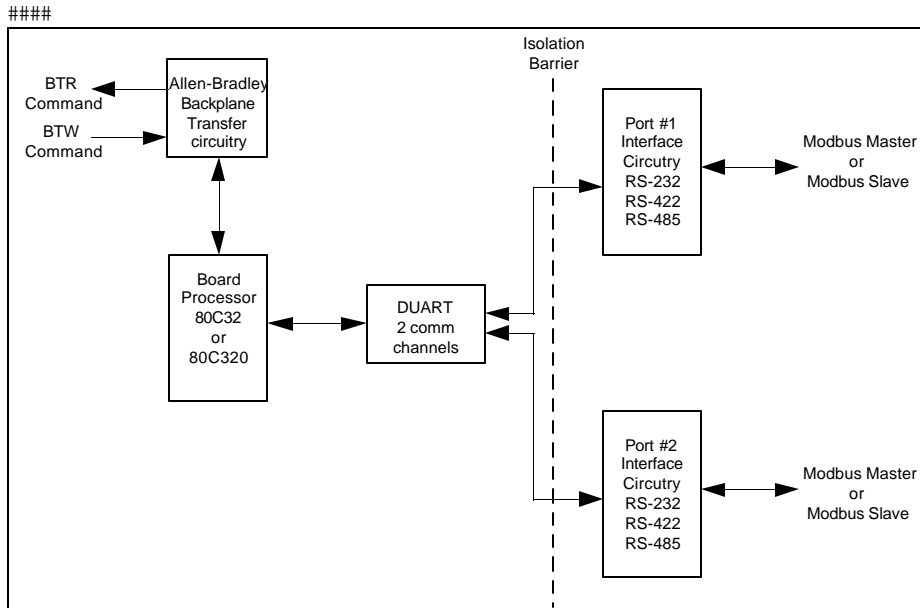


Diagrama de Lay-Out dos módulos 3100/3150

Os componentes funcionais primários das placas são:

um microcontrolador responsável pela operação global da placa, incluindo:

- Comunicação de barramento com processador Allen\_Bradley;
- Transferência de dados do módulo para o PLC;
- Aceitação de dados do PLC no módulo
- Prover comunicações DUART
- LEDs para Indicação de Status

Um "chipset" de barramento Allen\_Bradley responsável pelas comunicações entre módulo e processador A-B. O "chipset" contém tecnologia proprietária licenciada de AB projetada para:

- no caso do PLC, o "chipset" foi projetado para comunicar com o barramento usando os comandos de Transferência de Bloco, transferindo 64 palavras de uma vez
- no caso do SLC, o "chipset" foi projetado para se comunicar com o barramento usando arquivos MO/ML. Como não há funcionalidade "Transferência de Bloco" real no SLC, foi implementada uma forma de transferência de bloco usando a tabela I/O para controlar handshaking entre o módulo e o processador. Até 64 palavras podem ser transferidas de uma vez. Mostrado abaixo, presumindo o módulo no slot 1, há estes bits:

E 1/0 - Transferência Habilitada

Este bit é provido pelo módulo e usado a lógica Ladder para habilitar o movimento de dados através de barramento

S 1/0 - Transferência Realizada

Este bit é provido pela lógica Ladder para se comunicar com o módulo que o Ladder completou a transferência de dados.

Os circuitos de interface de porta provêm a interface física para o mundo real. As portas e os circuitos de interface são opticamente isolados do resto da placa, e por conseguinte do barramento, provendo um elevado nível de proteção para o processador A-B. Ambas portas suportam:

- RS-232
- RS-422, também chamada de conexão de 4 fios
- RS-485, também chamada de conexão de 2 fios

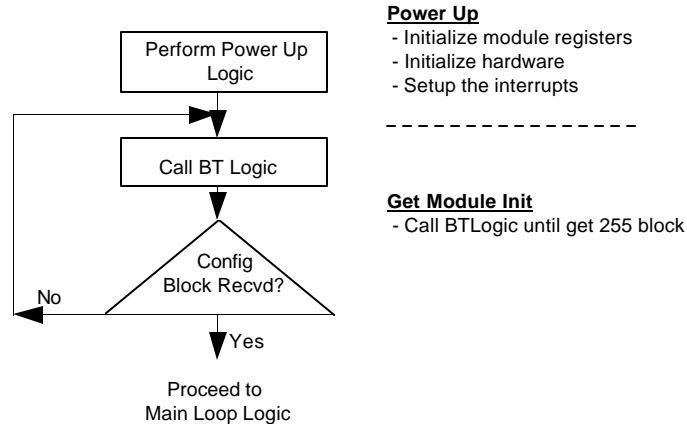
### 1.3 Conceitos Gerais

A discussão que se segue cobre diversos conceitos chaves para o entendimento da operação do módulo ProSoft.

#### 1.3.1 Energização e Reiniciação do Módulo

Na energização, ou quando se pressiona a tecla "Reiniciar" (somente 3100), o módulo começa a desempenhar suas funções. Estas funções são mostradas no fluxograma.

####



Incluído aqui

- 1- Inicializar hardware
  - Inicializar barramento
  - inicializar DUART
- 2- Inicializar registradores de módulo
  - limpar bloco de dados de módulo
  - limpar Lista de Comando
  - limpar Tabelas de Status de Erro
  - instalar constantes

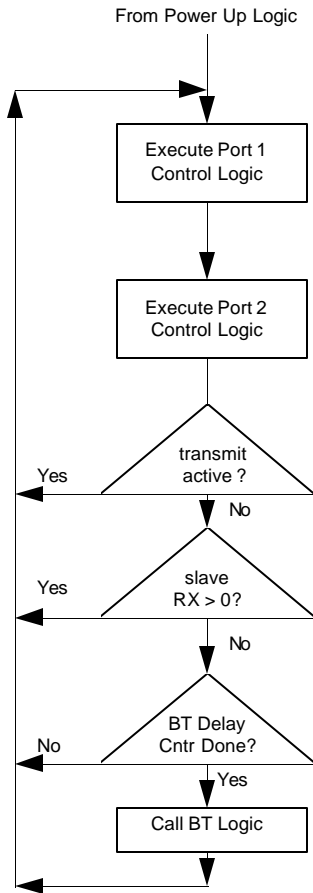
uma vez o espaço de registrador tendo sido inicializado, o módulo iniciará transferência de bloco com lógica Ladder para mover o bloco de configuração 255 para o módulo. Uma vez o módulo configurado, será iniciado a Repetição de Lógica Principal

#### 1.3.2 Lógica de Loop Principal

Completando o processo de configuração de energização, o módulo entra em loop infinito que inclui as seguintes funções:

1. Manoplas de Porta 1 e Porta 2
  - detecta fim de condição de mensagem
  - pede manoplas de mensagem
  - inicia comando se for Mestre
2. Transferência de Bloco
  - Testa pino CTS para garantir que o módulo não está no modo de transmissão
  - Testa buffer de porta escravo para garantir que não está recebendo
  - Testa Contador de Atraso de Transferência de Bloco
  - se tudo OK, então transfere o bloco

####



**Port 1 Control Logic**

- if port in RX mode, then test for message received
- if port in TX mode, then test if message transmit has completed
- If port is a master and ready for new command then create new command

**Port 2 Control Logic**

- if port in RX mode, then test for message received
- if port in TX mode, then test if message transmit has completed
- If port is a master and ready for new command then create new command

**Test Transmit Status**

- If either port is in process of transmitting then do not execute BT logic. The module uses the CTS pin status to detect the transmit status

**Test Slave port for characters**

- If either slave port is in the process of receiving characters then skip BT logic. This is done to assure timely response from the slave

**Test the Block Transfer Delay Counter**

- If the Block Transfer Delay counter has incremented beyond the counter preset (set in config) then go ahead and perform block transfer

**Execute Block Transfer Logic**

- Calls BT Logic which executes the BTR and BTW logic

**1.3.3 Espaço de Dados no módulo**

Um dos conceitos que é importante para desenvolver o entendimento do relacionamento entre o espaço de dados no módulo e como estes dados podem se movimentar entre o módulo e o processador PLC/SLC

A seguir se explica a estrutura de dados no módulo e como estes dados podem se movimentar entre o módulo e o programa Ladder. Alguns pontos a serem entendidos:

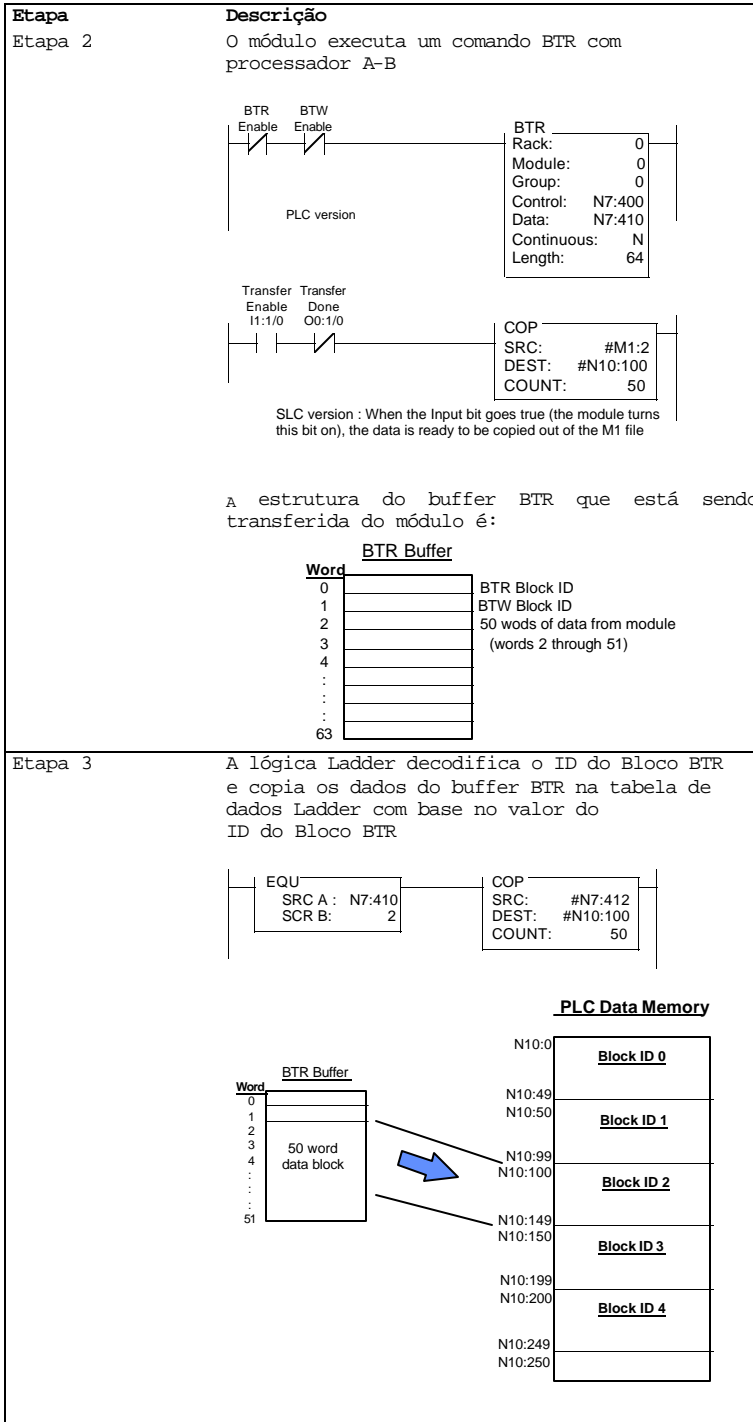
Ponto Chave	Descrição
Tamanho do espaço de registrador no módulo	<p>O módulo mantém um espaço de dados de 4000 palavras que podem ser usadas conforme aplicação para armazenar dados</p> <p><b>Module Memory</b> 4000 word block of 16 bit registers Addresses : 0 to 3999</p>

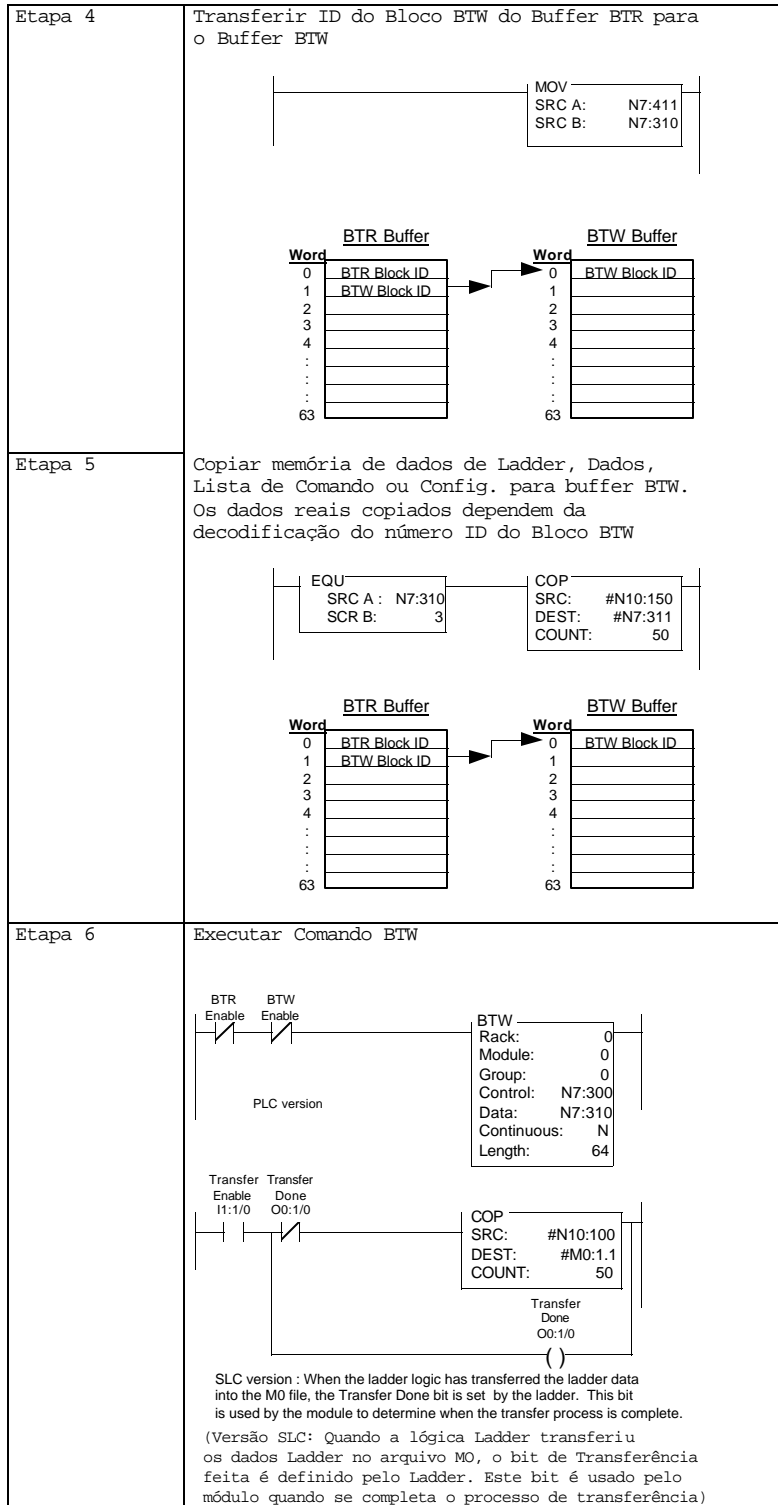
<p>Como o espaço de dados de módulo de 4000 palavras é dividido em blocos</p>	<p>Este bloco de 4000 palavras de dados é dividido logicamente em 80 blocos de 50 palavras 80 bls x 50 pals= 4000 pals</p> <p style="text-align: center;"><u>Memória de Módulo</u> Bl. de 4000 pals de registradores de 16 bits ID BL 0 a 79 Endereços 0 a 3999</p> <p style="text-align: center;">ID do Bloco 0 ID do Bloco 1 ID do Bloco 2</p> <p style="text-align: center;">ID do Bloco 77 ID do Bloco 78 ID do Bloco 79</p>								
<p>Como os dados são paginados entre módulo e processador</p>	<p>Via seqüência de transferência de dados definida na próxima seção, blocos de 50 palavras de dados (ou páginas) são transferidos bidirecionalmente entre o módulo e o processador PLC/SLC</p> <table style="width: 100%; border: none;"> <tr> <td style="text-align: center; width: 50%;"><u>Buffer BTR</u></td> <td style="text-align: center; width: 50%;"><u>Buffer BTW</u></td> </tr> <tr> <td style="text-align: center;">ID do Bloco BTR</td> <td style="text-align: center;">ID do Bloco</td> </tr> <tr> <td style="text-align: center;">ID do Bloco BTW</td> <td style="text-align: center;">BTW</td> </tr> <tr> <td style="text-align: center;">50 pals de dados</td> <td style="text-align: center;">50 pals de dados</td> </tr> </table>	<u>Buffer BTR</u>	<u>Buffer BTW</u>	ID do Bloco BTR	ID do Bloco	ID do Bloco BTW	BTW	50 pals de dados	50 pals de dados
<u>Buffer BTR</u>	<u>Buffer BTW</u>								
ID do Bloco BTR	ID do Bloco								
ID do Bloco BTW	BTW								
50 pals de dados	50 pals de dados								
<p>Como página de dados é colocado na tabela de dados do processador</p>	<p>A colocação de dados no processador PLC/SLC é controlada pelo usuário e lógica ladder de aplicação. Qualquer arquivo de dado disponível pode ser usado como fonte de dados para o módulo e como destino de dados a partir do módulo.</p> <div style="border: 1px solid black; padding: 5px; margin: 10px auto; width: fit-content;"> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">EQU</td> <td style="border: 1px solid black; padding: 2px;">COP</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">SRC A: N7:410</td> <td style="border: 1px solid black; padding: 2px;">SRC: #N7:412</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;">SCR B: 4</td> <td style="border: 1px solid black; padding: 2px;">DEST: #N10:200</td> </tr> <tr> <td style="border: 1px solid black; padding: 2px;"></td> <td style="border: 1px solid black; padding: 2px;">COUNT: 50</td> </tr> </table> </div>	EQU	COP	SRC A: N7:410	SRC: #N7:412	SCR B: 4	DEST: #N10:200		COUNT: 50
EQU	COP								
SRC A: N7:410	SRC: #N7:412								
SCR B: 4	DEST: #N10:200								
	COUNT: 50								

#### 1.3.4 Processo de Transferência de Dados por Barramento

A tabela a seguir provê uma visão geral do processo de transferência de dados entre o processador A-B e o módulo. Este processo é efetivamente controlado pela lógica Ladder no processador. A seguir há uma reflexão com respeito às etapas que ocorrem no módulo e no Ladder para realizar uma transferência de dados bem sucedida. Pode ser feita referência à lógica de exemplo no Apêndice para ver uma implementação real.

Etapa	Descrição
Etapa 1	<p>O módulo gera números ID do Bloco BTR e ID do Bloco BTW com base na seguinte lógica:</p> <p><u>ID do Bloco BTW</u> Se ID do Bl. BTW = Cont de Bl de escrita ID do Bloco BTW = 80 Se ID do Bl. BTW = 80 + Cont de Bl de cmd ID BL.BTW = Início de BL.de escrita Ainda ID BL BTR= ID BL.BTW + 1</p> <p><u>ID do Bloco BTR</u> Se ID do Bl BTR = Cont de BL. de Leitura ID do Bl BTR = Início de Bl.de Leitura Ainda ID do Bl BTR = ID do Bl. BTR + 1</p>





continuação

Etapa 7	O módulo recebe o dado BTW. Depois de decodificar o número ID do Bloco BTW, o módulo transfere o dado de buffer BTW para o local correto na memória de módulo.
---------	--

### 1.3.5 Intertravando Transferências de Bloco

Uma das considerações fundamentais que o módulo faz é fato de haver um comando BTR per BTW. NO módulo ao completar a instrução BTR, módulo passa imediatamente para a instrução BTW. Para o programador que segue nossa lógica de exemplo isto tem implicações mínimas.

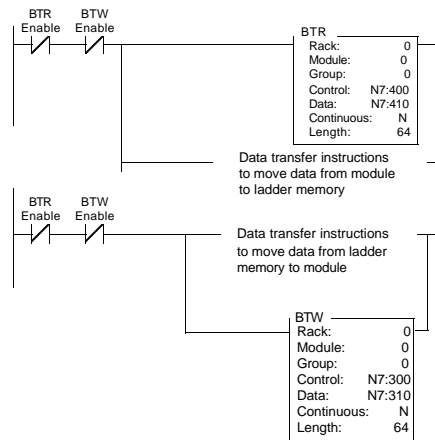
O problema surge no entanto quando a se tenta implementação de lógica Ladder que não atende às expectativas de transferência de bloco do módulo. Especificamente, o seguinte deve ser adicionado quando se programar a lógica Ladder para o módulo.

#### Programa PLC usando Instruções BTR/BTW

Nos 1771 tipos de processadores (PLC2, PLC3 e PLC5), os bits Habilitar BTR e BTW devem ser usados para habilitar as instruções de transferência de bloco. Com este tipo de programação, garante-se que o PLC não executa duas transferências de bloco ao mesmo tempo, e garante-se que as instruções BTR e BTW são garantidas se alternem.

Exemplos amplos deste tipo de programação de transferência de bloco são disponíveis na documentação A\_B seção no programa de lógica Ladder de exemplo no Apêndice.

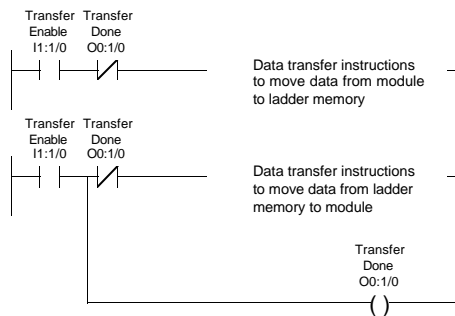
#####



#### Programa SLC usando Instruções M0/M1

Em processadores SLC, não há nenhum mecanismo para garantir a integridade de transferências de bloco de dado, como há na plataforma PLC. Por esta razão desenvolvemos um mecanismo "handshaking" projetado para garantir que todas as palavras em arquivos M1 e M2 sejam transferidos em conjunto. Seguir este mecanismo é o único modo que temos para garantir que os dados em um bloco correspondam ao bloco ID que estiver sendo transferido. A programação Ladder básica que deve ser implementada em uma aplicação SLS é a seguinte:

#####



### 1.3.6 Configuração de Processador SLC

Quando inicialmente se estabelece o arquivo SLC ou quando se move o módulo de um slot para outro, o usuário pode comunicar o slot para aceitar o módulo MCM

É importante que o slot contendo o módulo ProSoft seja configurado da seguinte forma:

- módulo 1746-BAS com 5/02 ou uma configuração maior;
- ou entrar 13106 para o módulo de código ID
- Configurar arquivos M0/M1 para 64 palavras
- Configurar E/S para 8 palavras

Segue-se uma etapa de como configurar estes arquivos usando software Allen\_Bradley APS. Outros usuários de pacotes de software devem seguir etapas similares.

A partir do menu principal :

- 1- selecione o programa de processador correto e use F3 para programação fora de linha (OffLine).
- 2- use F1 para funções de processador
- 3- use F1 para mudar processador
  - Aqui é necessário modificar o processador (notar que MCM somente trabalha com processadores 5/02 ou maiores)
- 4- use F5 para configurar E/S
  - Selecionar módulo 1746-BAS para SLC 5/02 ou maior, ou entrar 1316 para código de módulo
- 5- use F9 para Config SPIO quando o slot correto for iluminado
- 6- use F5 para Setup avançado
- 7- use F5 para comprimento de arquivo Mo - teclle 64 e Enter
- 8- use F6 para mostrar comprimento de arquivo - teclle 64 e Enter

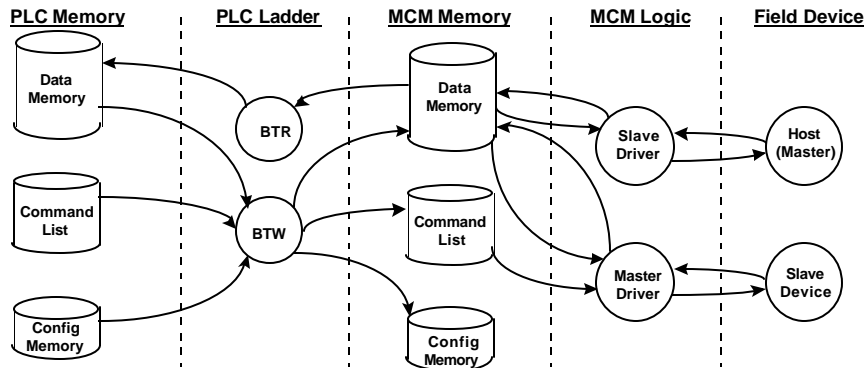
Sair e Salvar a configuração

## 1.4 Fluxo de Dados

### 1.4.1 Conceitos Gerais

No desenvolvimento de um sólido entendimento da operação do módulo, é importante entender o movimento de dados entre lógica Ladder, módulo, e drivers Mestre/ Escravo.

####



A discussão a seguir cobre o fluxo de dados nos diferentes estágios. Uma discussão posterior se encontra em seções posteriores com respeito ao fluxo de dados em diferentes modos de operação das portas

### 1.4.2 Lendo dados a partir do módulo

O módulo mantém 4000 blocos de memória de dados. Esta memória contém

- 1- O resultado das transações de porta Mestre
- 2- Dados movimentados para o módulo através da porta Escravo
- 3- Dados de Status de porta Escravo
- 5- Informação de Revisão de módulo

Durante transferência de dados do módulo para o PLC, a lógica Ladder é capaz de acessar esta informação.

### 1.4.3 Escrevendo dados no módulo

O módulo dependendo da configuração das portas, requer três tipos básicos de dados para operar corretamente. Os três tipos de memória que podem ser transferidos para o módulo são os seguintes:

- 1- Dados de Configuração. Estes dados contém todos o parâmetros necessários para o módulo configurar as portas seriais e estabelecer as transferências de dados entre o módulo e a lógica Ladder.
- 2- Lista de Comando. Este jogo de dados contém todos parâmetros que o módulo requer para codificar comandos válidos que serão transferidos da porta Mestre para outros dispositivos escravos Modbus
- 3- Memória de dados. Este tipo de memória é levado para o módulo para prover valores de dados necessários para a porta Escravo atender os pedidos de leitura (dados que um host recebe em consequência de um comando lido) ou para a porta Mestre atender pedidos gravados (isto é, os dados gravados para os Escravos).

### 1.4.4 Driver de Porta Mestre

Em aplicações normais, a porta Mestre é usada primariamente para emitir comandos de leitura a dispositivos escravos, daí atuando como coletor de dados e então transferindo os dados lidos para a lógica Ladder.

O módulo usa as entradas da Lista de Comando para codificar comandos Modbus válidos. À medida que cada comando é executado, o módulo escaneia a próxima entrada na Lista de Comando. Se a porta Mestre estiver emitindo um comando de leitura, os resultados da leitura são incluídos na Memória de Dados. Se a porta Mestre estiver emitindo um comando de escrita, os dados da Memória de Dados são gravados no dispositivo escravo.

Para cada comando que o módulo executa, o status do comando pode ser encontrado na Tabela de Erro Mestre. Esta Tabela que pode ser localizada em qualquer lugar no bloco de Memória de Dados é lida novamente na lógica Ladder como parte do processo de transferência de dados normal.

**1.4.5 Driver de Porta Escravo**

A operação da porta Mestre na verdade pode função de como a porta Escravo está configurada. A porta pode operar em 3 modos.

- 1- Modo Normal. Neste modo, Os comandos lidos pelos host são atendidos diretamente fora do bloco de memória e os dados de comando de escrita do host são enviados diretamente para a memória de dados.
- 2- Modo Pass\_Through. Neste modo, os pedidos de leitura pelos hosts são atendidos como no Modo Normal diretamente fora do bloco de Memória de Dados. A principal diferença é fato de os comandos de escrita do host serem transferidos para o buffer BTR para processamento na lógica Ladder.
- 3- Modo Route. Este modo é disponível quando uma porta é configurada como porta Mestre e a outra como porta Escravo. Neste modo, o driver Escravo verifica cada endereço escravo de comando contra até seis entradas na tabela Routing (set\_up de usuário). Se houver coincidência, o comando que for recebido na porta escravo é roteada para fora da porta mestre. A resposta do escravo é recebida pelo módulo e transferido para a porta escravo para ser transmitido para o host.

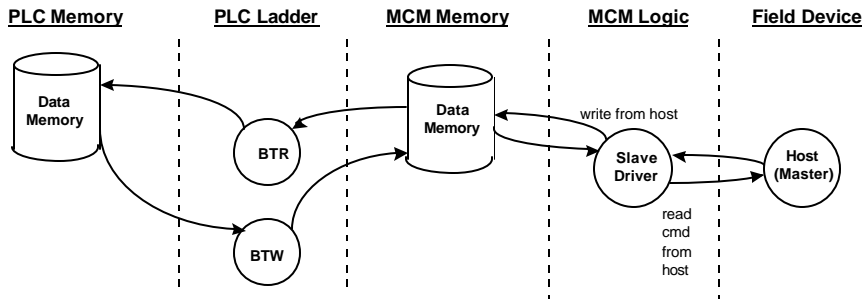
As seções a seguir provêem alguns diagramas de fluxo de dados para ajudar a entender os diferentes modos.

**1.4.6 Porta Escravo- Modo Normal**

Em operação normal (isto é nenhum item especial habilitado) a porta Escravo atende todos os pedidos de leitura e escrita usando a Memória de Dados. Isto proporciona a vantagem de ser muito simples de prover o set\_up e requer muito pouca lógica Ladder para ser implementada.

O requisito primário é que o espaço de dados em que o host está gravando não se sobreponha ao espaço de dados sobre o qual a lógica Ladder está gravando. Em muitas condições isto não traz nenhum problema, sendo que para estas condições se trata do melhor modo de configurar o módulo.

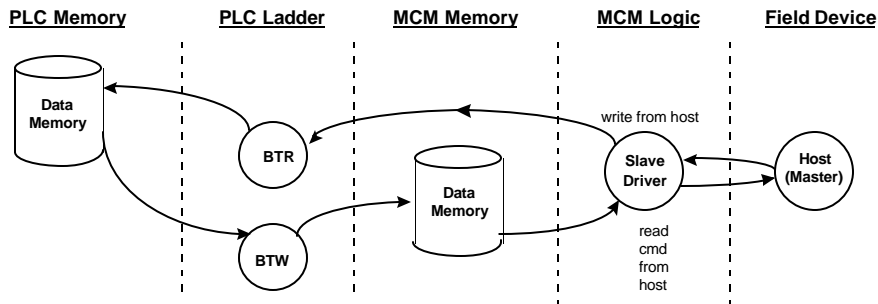
####



**1.4.7 Porta Escravo- Modo Pass\_Through**

O modo Pass\_Through proporciona a vantagem de superar aquela limitação imposta pelo Modo normal de Operação. Como mencionado acima, se os comandos de escrita de host recebidos pela porta Escravo devem se sobrepor ao espaço de endereço sobre o qual a lógica Ladder estiver gravando, então o modo Normal não poderá ser usado.

####



No modo Pass\_Through, todos os comandos de escrita recebidos pelo módulo (endereçados para endereço de escravo local ou broadcast) são transferidos para o buffer BTR para ser trabalhado pela lógica Ladder. Este modo apresenta diversas vantagens:

- 1- A tabela de dados Ladder pode ser acessada diretamente por comandos de host;
- 2- Todos os comandos de escrita podem ser aceitos implementando lógica Ladder que limita coisas como gama de endereços, limites de valor de dados, etc..
- 3- O programa Ladder reconhecerá definitivamente quando a porta Escravo tiver recebido um comando de escrita a partir do host. No modo Normal, a lógica Ladder não pode diferenciar entre os tipos de comando trabalhados.

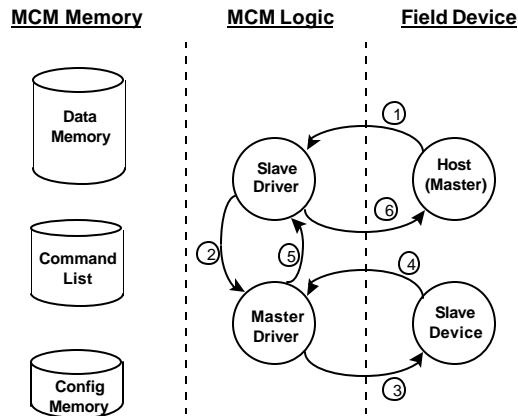
A única consideração à implementação do Modo Pass\_Through é o fato de os requisitos de lógica Ladder serem um pouco mais substanciais que aqueles do Modo normal.

#### 1.4.8 Porta Escravo- Modo Route

O Modo Route pode ser uma ferramenta poderosa se um host tiver acesso periférico a dados a partir de um escravo conectado à porta mestre do módulo. Isto pode ser um requisito comum em tipos SCADA de aplicações onde pode haver alguns outros dispositivos Modbus conectados à porta Mestre (tal como Computadores de Fluxo) tendo grandes volumes de dados úteis para um host mas não requeridos para o PLC/SLC local.

Quando o driver Escravo detecta que um comando foi recebido por um dispositivo na Lista de Route do módulo (configurado no Setup), as etapas que ocorrem são as seguintes:

####



- 1- O host emite um comando de escrita ou leitura para um endereço de escravo coincidindo com uma das entradas na Lista de Route do módulo MCM.
- 2- O driver escravo MCM para coincidir com a entrada na Lista de Route e envia o comando para o driver Mestre.
- 3- O driver Mestre executa o comando no menor tempo possível (i.e. , assim que completar a transação de comando em execução)
- 4- O escravo endereçado responde ao comando
- 5- A resposta do escravo passa do driver Mestre para o driver Escravo.
- 6- O driver Escravo retorna a resposta para o Host.

#### 1.5 Endereçamento de Modbus

Quando se aplica o módulo MCM, quer como Mestre ou Escravo, é importante entender pelo menos minimamente os itens associados ao endereçamento Modbus. Esta seção provê uma primeira consideração com respeito ao endereçamento para familiarizar os novos usuários de Modbus. Excertos da Especificação de Protocolo de Modbus se encontram disponíveis no Apêndice.

##### 1.5.1 Conceitos de Endereçamento Modbus

O esquema de endereçamento Modbus foi desenvolvido no início em torno da tabela de dados e estrutura E/S em PLCs Modicon. Em consequência, o protocolo Modbus suporta acesso a vários espaços de dados no PLC Modicon.

De longe o espaço de dados mais comum é o espaço 4xxxx que usa os códigos de função 3, 6, 16. Este espaço é usado para transferir valores de registro de 16 bit, valores de ponto flutuante e mesmo dados mapeados de bit. Usando terminologia de endereçamento Modbus formal, este espaço de dados começa de fato no endereço 4001 (Modbus baseado em um (1), enquanto o endereçamento de tabela de dados MCM é baseado em zero (0)).

O acesso a diferentes espaços de dados é determinado pela função de código usado. O quadro abaixo mostra os quatro diferentes tipos de espaços de dados, as gamas numéricas destes espaços, os códigos de função que são usados para executar instruções de leitura e escrita dentro destes espaços de dados.

####

Register Space		Read Command	Write Command
0XXXX	Coils (Discrete Out)	1	5,15
1XXXX	Inputs (Discrete In)	2	N/A
3XXXX	Input Registers (Analog In)	4	N/A
4XXXX	Holding Registers (Memory Regs)	3	6,16

Relacionamento entre os códigos de função Modbus e esquema de endereçamento Modbus

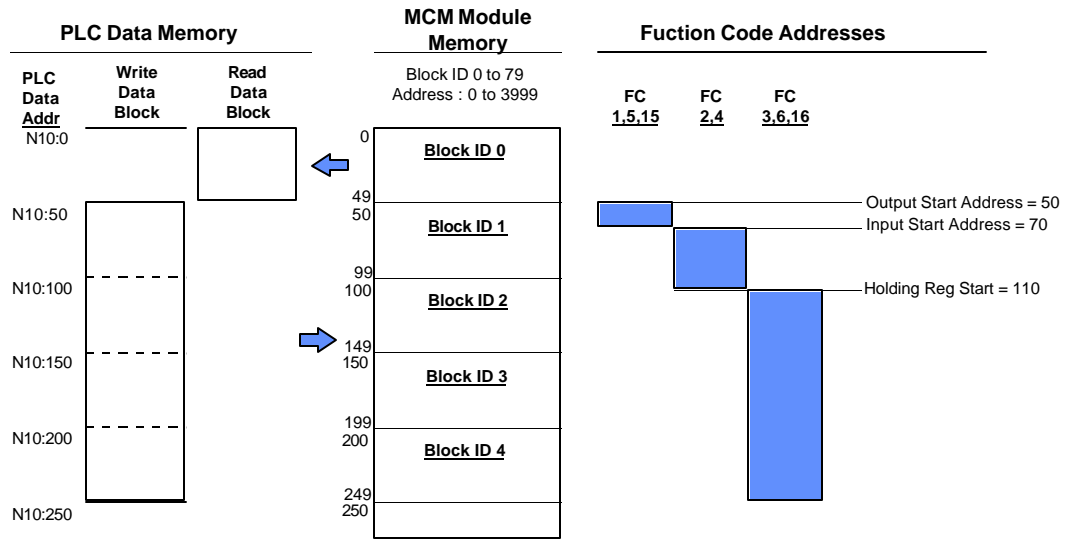
#### 1.5.2 Suporte MCM de funcionalidade Modbus

O módulo MCM suporta todos os códigos de função Modbus usados para transferência de dados. A tabela a seguir documenta os códigos de função e as gamas de endereçamento suportadas (ver seção 6 para mais detalhes).

Código de Função	Descrição	Gama de Endereço
1	Read Output Status	0001 - 9999
2	Read Input Status	10001 - 29999
3	Read Holding Regs	40001 - 49999
4	Read Input Regs	30001 to 39999
5	Force Single Coil	00001 - 9999
6	Preset Single Reg	40001 - 49999
8	Loopback Test	
15	Force Multiple Coils	0001 - 9999
16	Preset Multiple Regs	40001 - 49999

#### 1.5.3 Mapeando Endereços Modbus para Endereços de Dados Ladder

O relacionamento entre a lógica Ladder e esquema de endereçamento Modbus é controlado pelo Programador de Aplicação. O diagrama abaixo é um exemplo do mapa de memória que poderia ser provido em uma aplicação envolvendo o módulo



glossário do diagrama acima  
**Memória de Dado PLC**

**Memória de Dado MCM**

**End.s de Código de Função**  
 End de Início de Saída= 50  
 End de Início de Entrada= 70  
 Início de Reg de Mem= 110

Neste exemplo, o MCM está estabelecido em um escravo Modbus respondendo às consultas de um host. O host deverá ler os dados do módulo começando no registrador do módulo 50. A seguinte tabela de memória corresponde a este layout.

Código de função	Endereço de módulo	Endereço de Modbus
1,5,15	50 a 69	1 a 320
2,4	70 a 109	10001 a 10640 3001 a 30040
3,6,16	110 a 249	40001 a 40140

## 2 Avançando- Etapa por Etapa

A instalação do módulo 3100-3150 MCM pode ser facilmente realizada. A instalação em um sistema requer somente poucas etapas. A seguir apresentamos um procedimento etapa por etapa para implementar uma aplicação operacional.

Etapa	Exemplo	Aplicação do Usuário
1. Identificar posição do Rack	Rack 0 Grupo 2 Slot 0	Rack Grupo Slot
2. Identificar Utilização de arquivos de dados PLC	Buffers BT N7 Controle BT N7 Arquivo Config N7 Arquivo de Dados N10	Buffers BT Controle BT Arquivo Config. Arquivo de Dados
3. Lógica Ladder	Exemplo no disco e no Apêndice (diversos exemplos a escolher)	Selecionar o exemplo mais próximo a sua aplicação e modificar de acordo com necessário
Modificar Lógica para posição do Rack	PLC BTR Rng 2:0 BTW Linha 2:1 SLC Endereços E:x0 Endereços S:x 0 Endereços M0 x Endereços M1 x	Modificar estas instruções de acordo com necessário com base na posição de Rack requerida. Esteja certo de configurar o slot no SLC
5. Modificar Lógica para utilização de arquivo de dados	N7 e N10 são usados como espaços de dados para o módulo	Criar arquivos e mudar referências de N7 e N10.
6. Instalar placa no Rack	Rack de módulo desligamento e instalação	Módulo de desligamento e instalação
7. Conectar um cabo de com à frente do módulo	Decidir o tipo de cabo necessário para aplicação	
8. Ligar o sistema e colocar PLC em execução (RUN)	Monitorar o arquivo de status e os LEDs na frente do módulo	

Com o hardware instalado e os programas necessários para o processador descarregados, o sistema está pronto (os demais componentes do sistema prontos em segurança).

### 3 Visão Geral da Lógica Ladder

A transferência de dados entre processador e módulo da ProSoft\_Technology ocorre usando comandos de transferência de bloco no caso do PLC e comando de transferência de dados MO/M1 no caso do SLC. Estes comandos transferem até 64 registros físicos per transferência. O comprimento de dados lógico se altera dependendo da função de transferência de dados.

A discussão e seções a seguir detalham as estruturas de dados usadas para transferir os diferentes tipos de dados entre módulo ProSoft\_Technology e processador. O termo "Transferência de Dados" é usado genericamente a seguir para representar a transferência de blocos de dados entre processador e módulo da ProSoft\_Technology. Embora uma função de Transferência de Bloco verdadeira não exista no SLC, implementamos um pseudo-comando de transferência de bloco para garantir a integridade de dados no nível de bloco. Exemplos de Lógica Ladder PLC e SLC estão incluídos no Apêndice.



Para funcionar o módulo da ProSoft\_Technology, PLC/SLC devem estar em modo RUN, ou REM RUN. Se estiver em qualquer outro modo (Fault/PGM), o módulo interrompe todas comunicações até reiniciarem as transferências de blocos.

#### 3.1 Visão Geral Operacional

Quando se energiza o módulo move um 255 para a palavra 1 do arquivo de dados BTR. Este é um sinal de que o módulo precisa receber dados de comunicação antes de avançar. Uma vez a configuração recebida, o módulo deve iniciar transferência de dados para e a partir do processador dependendo se quantos blocos de leitura e escrita terem sido configurados. Uma vez completado, o módulo deve transferir os blocos de comando se algum tiver sido configurado.

#### 3.2 Lógica Ladder

O fluxo da Lógica Ladder em uma certa extensão é pré-definida da maneira que o módulo foi programado. O fluxo esperado da lógica Ladder deve ser o seguinte:

##### Linha de leitura

- 1- Ler Dados a partir do Módulo. No caso de PLC, os dados de módulo serão transferidos para o Buffer BTR. No caso do SLC, os dados de módulo serão acessados diretamente fora do arquivo M1.
- 2- Decodificar o número ID do Bloco BTR. Dependendo do valor do ID do Bloco BTR, copiar os dados de módulo na localização correta na tabela de dados de lógica Ladder.
- 3- Mover o número ID do Bloco BTR da palavra 1 do Buffer BTR para a palavra 0 do Buffer BTW. No caso do SLC, a transferência será na verdade da palavra 1 do arquivo M1 para a palavra 0 do arquivo M0. O número ID do Bloco BTW deve ser manipulado se necessário de modo a garantir que os dados não sejam sobre postos no módulo. (a ramificação de teste LIM faz isto na lógica de exemplo)
- 4- Testar o Comando Iniciado por Evento e a configuração do módulo.

##### Linha de escrita

- 1- Decodificar o número ID do Bloco BTW e dependendo do valor mover quer os valores de dados, valores de Lista de Comando ou valores de Configuração para o buffer BTW (arquivo M0 no SLC).
- 2- Se a transferência de configuração estiver habilitada, então cancelar o bit de habilitar configuração
- 3- Se o Comando Iniciado por Evento estiver habilitado, então cancelar o bit habilitar
- 4- Executar transferência de BTW. No PLC, Isto será feito habilitando a instrução BTW. No SLC, isto será feito instalando o bit de Transferência Realizada (um bit de saída foi designado para esta função no projeto do módulo).

## 4 Escrevendo no Módulo

Esta seção provê detalhes de nível de referência com respeito à transferência de dados a partir do processador PLC/SLC para o módulo MCM. Este tipo de transferência permite à lógica Ladder enviar configuração, lista de comando, e dados para o módulo.

### 4.1 Transferência de Bloco para o Módulo

A transferência de dados para o módulo a partir do processador é executada com a função de Escrever Transferência de Bloco. Os diferentes tipos de dados transferidos requerem estruturas de bloco de dados diferentes, mas a estrutura básica é a seguinte:

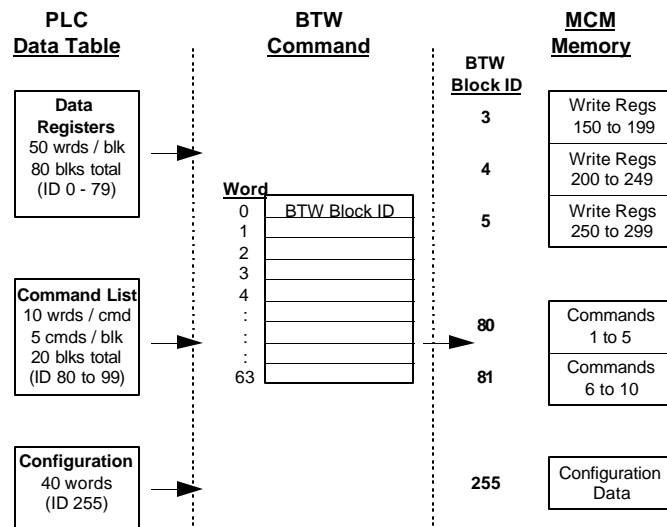
Palavra	Nome	Descrição
0	ID do Bloco BTW	Um código identificador de página de bloco. Este bloco é usado pelo módulo ProSoft para determinar o que fazer com o bloco de dados. Códigos válidos são: Cod BTW Descrição 0-79 80-99 100-120 255 Memória de dados de módulo Lista de Comando Escritas por Evento Configuração de Comunicação de Módulo



Embora nem todas as 64 palavras do Buffer de dados possam ser usadas, os comprimentos BTW e M0 devem ser configuradas para 64 palavras, caso contrário a operação do módulo será imprevisível

1 a 63	Dados	Os dados a serem escritos no módulo. A estrutura de dados depende do código do ID do Bloco. As seções a seguir provêem detalhes às diferentes estruturas
--------	-------	--

####



**Transferência de dados do PLC para o MCM:** Valores de dados e entrada da Lista de Comando são paginados no módulo MCM. O tipo de dado e a localização sendo gravados em correspondência no número do ID do Bloco BTW. O número de Bloco BTW é controlado pelo módulo MCM, como será discutido mais adiante nesta seção.

#### 4.2 Configuração de Comunicação (ID do Bloco BTW 255)

Os parâmetros de comunicação de firmware da ProSoft\_Technology devem ser configurados uma vez quando a placa for energizada pela primeira vez, e sempre que os parâmetros forem alterados.

##### Energização

Na energização, o módulo entra em loop lógico aguardando dados de configuração a partir do processador. Enquanto aguarda, o módulo determina a segunda palavra no buffer BTR (ID do Bloco BTW) em 255, dizendo ao processador que o módulo deve ser configurado imediatamente. O módulo permanece continuamente realizando transferências até receber o bloco de parâmetro de configuração de comunicação. Na recepção, o módulo deve começar a execução da Lista de Comando se houver, ou procurando pela Lista de Comando a partir do processador.

##### Mudando parâmetros durante operação

A mudança de valores na tabela de configuração poderá ser feita a qualquer momento. O módulo não aceita quaisquer das mudanças até o processo de reconfiguração ser iniciado. Isto pode ser realizado de diversas maneiras, incluindo:

- 1- Reenergizando o rack
- 2- Pressionando a botão reset no módulo (somente 3100)
- 3- Movendo 255 para posição de ID Bloco BTW (ver lógica de exemplo com B3/0 aplicado).

No processo, o LED "CFG" acende, indicando que o módulo recebeu o bloco de config..



Transferir os parâmetros de Configuração de Comunicação para o módulo provoca um reset da porta de comunicação, assim como leva DTR para 200 ms pulsos para reset qualquer hardware anexo.

A estrutura de bloco de dados de configuração que deve ser transferida do processador para o módulo é mostrada na tabela abaixo:

Buffer BTW	End Dado	Nome	Valor de Exemplo
0		BTW Block ID	255
		<b>Port 1 Config</b>	
1	N[ ]:0	Port Configuration Word	0 - Master 1 - Slave
2	N[ ]:1	Port Slave Addr	1
3	N[ ]:2	Baud Rate	5
4	N[ ]:3	RTS to TxD Delay	0
5	N[ ]:4	RTS Off Delay	0
6	N[ ]:5	Response Timeout	0
7	N[ ]:6	Intercharacter Delay	0
8	N[ ]:7	Setup Parm #1	0
9	N[ ]:8	Setup Parm #2	0
10	N[ ]:9	Setup Parm #3	0
		<b>Port 2 Config</b>	
11	N[ ]:10	Port Configuration Word	0 - Master 1 - Slave
12	N[ ]:11	Port Slave Addr	1
13	N[ ]:12	Baud Rate	5
14	N[ ]:13	RTS to TxD Delay	0
15	N[ ]:14	RTS Off Delay	0
16	N[ ]:15	Response Timeout	0
17	N[ ]:16	Intercharacter Delay	0
18	N[ ]:17	Setup Parm #1	0
19	N[ ]:18	Setup Parm #2	0
20	N[ ]:19	Setup Parm #3	0
21	N[ ]:20	Read Block Cnt	3
22	N[ ]:21	Write Block Cnt	1
23	N[ ]:22	Cmd Block Cnt	2
24	N[ ]:23	Slave Err Ptr	100
25	N[ ]:24	Master Error Ptr	120

26	N[ ]:25	BT Delay Cntr	0
27	N[ ]:26	Floating Point Offset	0
28	N[ ]:27	Read Block ID Start	0
29	N[ ]:28	Write Block ID Start	0
30	N[ ]:29	Spare	0
31 a -36	N[ ]:30 a N[ ]35	Route Mode Slaves 1 to 6	0

## Configuração de Portas 1 e 2

Endereço de Dado	Nome	Descrição																											
N[ ]:0 N[ ]:10	Palavra de configuração de porta	<p>Este registro contém diversos parâmetros de configuração de comunicação codificados na palavra, que são as seguintes:</p> <p><u>Modo de protocolo</u>: O protocolo da porta é selecionado por estes bits</p> <p><b>Bits 210</b></p> <p>000_ Modbus Mestre Modo RTU  001_ Modbus Escravo Modo RTU  010_ Modbus Escravo Modo ASCII 7 bit  011_ Modbus Mestre Modo ASCII 7 bit  100_ Modbus Escravo Modo ASCII 8 bit  101_ Modbus Escravo Modo ASCII 8 bit</p> <p><u>Modo Pass_Through</u>: O modo de operação da Porta Escravo é selecionado por este bit</p> <p><b>Bit 3</b></p> <p>0_ Pass_Trough Desabilitado  1_ Pass_Trough Habilitado</p> <p><u>Modo Routing</u>: Habilitar Escravo para o modo Routing Mestre</p> <p><b>Bit 4</b></p> <p>0_ Modo Routing Desabilitado  1_ Modo Routing Habilitado</p> <p><u>Stop Bits</u>: o número de stop bits a ser usado é definido da seguinte forma:</p> <p><b>Bits 13 12</b></p> <p>0 0_ um stop bit  0 1_ dois stop bits  1 x_ Configuração de Porta inválida</p> <p><u>Paridade</u>: O modo paridade a ser usado pelo módulo é definido por esta palavra da seguinte forma:</p> <p><b>Bits 15 14</b></p> <p>0 0_ Sem paridade  0 1_ Paridade ímpar  1 0_ Paridade par  1 1_ Configuração de Porta inválida</p>																											
N[ ]:1 N[ ]:11	Endereço de Escravo	Quando a Porta é configurada no modo Escravo, o valor digitado neste registro é usado como endereço Escravo de Modbus. Os valores válidos variam de 1 a 247																											
N[ ]:2 N[ ]:12	Baud_Rate	<p>Baud_rate na qual a porta deve operar.</p> <table border="1"> <thead> <tr> <th></th> <th>Valor</th> <th>Baud Rate</th> </tr> </thead> <tbody> <tr> <td></td> <td>0</td> <td>300 Baud</td> </tr> <tr> <td></td> <td>1</td> <td>600 Baud</td> </tr> <tr> <td></td> <td>2</td> <td>1200 Baud</td> </tr> <tr> <td></td> <td>3</td> <td>2400 Baud</td> </tr> <tr> <td></td> <td>4</td> <td>4800 Baud</td> </tr> <tr> <td></td> <td>5</td> <td>9600 Baud</td> </tr> <tr> <td></td> <td>6</td> <td>19200 Baud</td> </tr> <tr> <td></td> <td>7</td> <td>38400 Baud</td> </tr> </tbody> </table>		Valor	Baud Rate		0	300 Baud		1	600 Baud		2	1200 Baud		3	2400 Baud		4	4800 Baud		5	9600 Baud		6	19200 Baud		7	38400 Baud
	Valor	Baud Rate																											
	0	300 Baud																											
	1	600 Baud																											
	2	1200 Baud																											
	3	2400 Baud																											
	4	4800 Baud																											
	5	9600 Baud																											
	6	19200 Baud																											
	7	38400 Baud																											

		<div style="border: 1px solid black; padding: 5px;"> <p>As duas portas do módulo são limitadas a um baud rate maior que 19200 ou 38400 baud. O módulo não pode ser configurado com uma porta a 19200 e outra a 38400. Se for configurado desta forma, um Erro de Configuração de Porta será mostrado.</p> </div>
N[]:3 N[]:13	Atraso RTS a TXD	<p>Este valor representa o tempo em incrementos de 1 ms a ser inserido entre RTS declarado e a efetiva transmissão de dados. O atraso, se maior em duração que o atraso de tempo do hardware associado ao CTS, ultrapassa a linha CTS até timeout completo.</p> <p>Este parâmetro configurável é útil para interfacear dispositivo de modem, sempre que o ruído de linha subsistir antes de os dados forem transmitidos ou se a velocidade da transmissão de dados for reduzida.</p> <p>Valores válidos variam de 0 a 65535 (0xffff)</p>
N[]:4 N[]:14  N[]:5 N[]:15	Atraso RTS off  Timeout resposta de Mensagem	<p>O valor nesta palavra representa o número de incrementos de atraso de tempo de 1 ms inserido depois de o último caractere ser transmitido e antes de RTS cair. O módulo automaticamente insere uma largura de um caracter Off_Delay, garantindo que RTS não caia até o último caractere ter sido completamente enviado. A menos que em condições não usuais, este valor será normalmente configurado com valor zero.</p> <p>Valores válidos variam de 0 a 65535 (0xffff)</p> <p>Este registro representa o período de timeout da resposta de mensagem em incrementos de 1 ms. Este é o tempo em que a porta configurada como Mestre aguarda antes de retransmitir um comando se nenhuma resposta for recebida do escravo endereçado. O volume é determinado dependendo dos tempos de resposta do Escravo.</p> <p>A faixa admissível de valores é 0 a 65535 (0xffff). Se for fornecido o valor de zero, módulo estabelece um valor default de valor de timeout de um segundo (1000 ms)</p>
N[]:6 N[]:16	Timeout Inter-Caractere	<p>Este registro é usado em situações onde o fim do atraso de timeout de caractere de mensagem deve ser estendido além da largura normal de 3,5 caracteres. O valor fornecido representa o número de intervalos de 1 ms de "nenhuma transmissão" que será contado antes de aceitar a mensagem. Este parâmetro será útil em satélites ou instalação de rádio de pacote onde uma transmissão de dados pode ser dividida em dois pacotes.</p> <p>Aumentar este valor além do tempo de manipulação de pacote elimina erros de timeout</p> <p>Valores válidos variam de 0 a 65535 (0xffff)</p>
N[]:7 N[]:17	Parâmetro de Setup #1 Modo Mestre: Não Usado Modo Escravo: Endereço de Início de Memória de Entrada	<p><u>Modo Escravo Modbus</u></p> <p>Este valor define o endereço de offset no espaço de dados de 4000 palavras que o Porta Escravo MCM usa quando responde aos comandos 2 e 4 do código de função. Por exemplo, para iniciar o espaço de endereço na palavra 150, dê 150. Um comando 2 e 4 de função com endereço zero (10001 ou 30001) inicia leitura na palavra 150.</p> <p>Valores válidos variam de 0 a 3999.</p>
N[]:8 N[]:18	Parâmetro de Set_Up #2 Modo Mestre: Não Usado Modo Escravo: Endereço de Início de Memória de Entrada	<p><u>Modo Escravo Modbus</u></p> <p>Este valor define o endereço de offset no espaço de dados de 4000 palavras que a Porta Escravo MCM usa quando responde aos comandos 1, 5, 15 do código de função. Por exemplo, para localizar a imagem de saída na palavra 100, dê 100.</p> <p>Um comando 1 de código função com endereço zero então inicia leitura na palavra 100.</p>

N[]:9 N[]:19	Parâmetro de Setup #2 Modo Mestre: Não Usado Modo Escravo: Endereço de Início de Memória de Entrada	<u>Modo Escravo Modbus</u> Este valor define o endereço de offset no espaço de dados de 4000 palavras que a Porta Escravo MCM usa quando responde aos comandos 3, 6, 16 do código de função. Por exemplo, para localizar endereço 4001 na palavra 100 no módulo, dê 100. Um comando de função 3 com endereço zero (40001) inicia leitura na palavra 100. Valores válidos variam de 0 a 3999.
-----------------	--	---

#### Configuração de Sistema

End	Nome	Descrição
N[]:20	Quantidade de Blocos de Dados de Leitura	<p>Este valor define o número de 50 blocos de dados de palavras que deve ser transferido do módulo MCM para o processador. Os voltam do início do módulo no bloco 0 e incrementam a partir deste. A quantidade máxima de bloco é 80.</p> <p>Como exemplo, um valor de 5 retorna blocos de dados de ID do Bloco BTR 0, 1, 2, 3, 4 ou registros de módulo 0 a 249.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Se for dado um valor maior que 80, será ativado Erro de Configuração de Sistema.</p> </div>
N[]:21	Quantidade de Blocos de Dados de Escrita	<p>Este valor representa o número de 50 blocos de dados de palavra que deve ser transferido do processador para o módulo MCM. O módulo usa este valor para retornar um número de ID do Bloco BTW para o processador. A lógica Ladder pode usar este valor para determinar qual dado mover para MCM através de escrita de transferência de bloco. A quantidade máxima de bloco é 80.</p> <p>Como exemplo, se um valor de 5 for dado, MCM retorna números de ID do Bloco BTW 0, 1, 2, 3, 4 para lógica Ladder (ver seção 4.2).</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Se for dado um valor maior que 80, será ativado Erro de Configuração de Sistema.</p> </div>
N[]:22	Quantidade de Blocos de Comando	<p>Esse valor representa o número de 50 blocos de comando de palavra que deve ser transferido do processador para o módulo MCM. Este valor será zero se o módulo não for configurado com porta Mestre. Ver a discussão na Seção 4.1.3 para detalhes com respeito ao número de blocos de comando necessários. A máxima quantidade de bloco é 20.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Se for dado um valor maior que 20, será ativado Erro de Configuração de Sistema.</p> </div>

<p>N[]:23</p>	<p>Ponteiro de Bloco de Erro de Escravo</p>	<p>Este valor representa a posição de início relativa na tabela de dados de módulo dentro do qual o bloco de dados de erro de Escravo de Modbus é colocado. A Tabela de Erro de Escravo é um bloco de 20 palavras contendo status de Porta Escravo e diversos contadores de comunicação. O dado de erro pode ser colocado em qualquer lugar no espaço de dados do módulo (0 a 3999). O conteúdo da Tabela de Erro pode então ser obtido como parte dos dados de Registros regulares.</p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p>Se for dado um valor maior que 3980, será ativado Erro de Configuração de Sistema.</p> </div> <div style="text-align: center;"> <p><b>MCM Module Memory</b></p> <p>Block ID 0 to 79 Address : 0 to 3999</p> <p>The diagram shows a vertical stack of memory blocks. Block ID 0 is at address 0, Block ID 1 at 50, Block ID 2 at 100, Block ID 3 at 150, and Block ID 4 at 200. A horizontal line labeled 'Slave Error Table Pointer= 100' points to the start of Block ID 2. Below the diagram, it states: 'Slave Error Table. The data registers 100 to 119 will contain the Slave Error Table.'</p> </div> <p><u>glossário</u>                      Ponteiro de Tabela de Erro=100</p> <p><u>Tabela de Erro Escravo</u> Os registros de dados 100 a 119 contêm a Tabela de Erro de Escravo</p>
<p>N[]:24</p>	<p>Ponteiro de Bloco de Erro de Mestre</p>	<p>Este valor representa a posição de início relativa na Tabela de registro de dados do módulo no qual o Bloco de Dados de Erro Mestre é colocado. O bloco de erro (120 palavras) pode ser colocado em qualquer lugar no espaço de dados do módulo (0 a 3999). O conteúdo da Tabela de Erro então pode ser obtido como parte dos Dados de Registro Regulares.</p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p>Se for dado um valor maior que 3880, será ativado Erro de Configuração de Sistema.</p> </div> <div style="text-align: center;"> <p><b>MCM Module Memory</b></p> <p>Block ID 0 to 79 Address : 0 to 3999</p> <p>The diagram shows a vertical stack of memory blocks. Block ID 0 is at address 0, Block ID 1 at 50, Block ID 2 at 100, Block ID 3 at 150, and Block ID 4 at 200. A horizontal line labeled 'Master Error Table Pointer= 120' points to the start of Block ID 2. Below the diagram, it states: 'Master Error Table. Data registers 120 to 240 will contain the Master Error Table.'</p> </div> <p><u>glossário</u>                      Ponteiro de Tabela de Erro=100</p> <p><u>Tabela de Erro Escravo</u> Os registros de dados 120 a 240 contêm a Tabela de Erro de Escravo</p>

N[]:25	Contador de Atraso de Transferência de Bloco	<p>Este é um valor empírico usado pelo módulo para balancear o tempo gasto pelo módulo na transferência de bloco e o tempo gasto manipulando comunicação pela porta. O valor fornecido é usado como contador de loop no módulo, onde a quantidade é incrementada a cada loop. Quando a quantidade iguala o contador de Atraso de Transferência de bloco uma seqüência de Transferência de Bloco é iniciada. A gama neste valor varia de 0 a 255.</p> <p><u>Exemplo</u> Em aplicações no Modo mestre com o módulo em rack remoto, a frequência de execução de comando pode ser melhorada fornecendo um valor de 75-150. O valor deve ser determinado empiricamente.</p>
N[]:26	Offset de Ponto Flutuante	<p>Este valor é usado pelo driver de porta Escravo do módulo para suportar endereçamento de leitura e escrita de Registros de Ponto Flutuante quando registros de endereçamento &gt; 700, comumente chamado de versão ENRON do protocolo Modbus. O valor de offset é usado da seguinte forma pelo módulo:</p> <p>End. de Reg MCM = Offset de Ponto Flutuante + (End Reg - 700)*2</p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p>Offset de Ponto Flutuante não é usado no Modo Pass_Through, o endereço fornecido para lógica Ladder é calculado como sendo:</p> <p>End = End Reg - 7000</p> </div>
N[]:27	Início de ID do Bloco de Leitura	<p>Este valor determina o início do número de ID do Bloco BTR que retorna do módulo. Como exemplo, se a lógica Ladder precisa receber Blocos 2 a 5 do módulo, o parâmetro deve ser configurado com "2" e o contador de Bloco de Leitura deve ser configurado com "4". Os valores válidos variam de 0 a 79.</p>
N[]:28	Início de ID do Bloco De Escrita	<p>Este valor determina o início do número de ID do Bloco BTW que retorna para a lógica Ladder. Como exemplo, se a lógica Ladder precisa gravar nos Blocos de 4 a 5 no módulo, este valor deveria ser configurado com 4 e o Contador de Bloco de Escrita com 2. Valores válidos variam de 0 a 79</p>
N[]:30 a N[]:35	End. de Escravo de Modo Route #1 #6	<p>Estes seis endereços são providos para quando o módulo MCM for configurado com Modo Routing Habilitado. Neste modo, qualquer comando que chegue na porta Escravo e que coincida com um dos endereços do Modo Route será retransmitido a partir da porta Mestre. A resposta deste Escravo será roteada de volta para o host através da porta Escravo.</p>

#### 4.3 Escrevendo na Memória de Dados de Módulo (Códigos de ID do Bloco BTW 0-79)

A escrita no espaço de registro de MCM é realizada usando uma Escrita de Transferência de Bloco com Código de ID do Bloco BTW de 0 a 79 seguido de 50 palavras de dados.

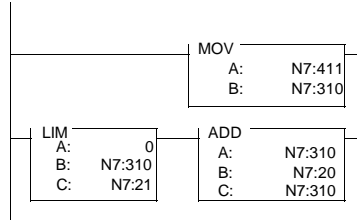


É preciso tomar cuidado com layout da memória para assegurar que os comandos de leitura e escrita não se sobreponham a dados que estão sendo movidos a partir da lógica Ladder de processador. Os dados Modbus não podem ser movidos para os blocos de 50 palavras que também é atualizado pelo processador. Os exemplos de lógica Ladder no Apêndice abordam assunto.

#### 4.3.1 Lógica Ladder para Escrever Dados no módulo

A Lógica Ladder requerida para mover dados para o módulo é uma simples série de ramificações EQU-COP ou pode ser implementada usando endereçamento indireto. O modo no qual implementamos a transferência para o módulo em todos os nossos exemplos (ver Apêndice e Notas de Aplicação) usa um processo de duas etapas, onde:

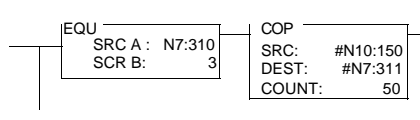
Etapa 1: Durante o processo BTR, o módulo alimenta para a Lógica Ladder um número de ID do Bloco BTW na segunda palavra da Buffer de Dados BTR. A Lógica Ladder é implementada de modo a aceitar este valor, acondicioná-lo se necessário, e então movê-lo para a localização ID do Bloco BTW. A Lógica Ladder para isto é mostrada abaixo:



##### Estabelecer o Número de ID do Bloco BTW

Localizado na base da linha BTR (Linha 0), esta lógica move o ID do Bloco BTW recebido do módulo e o desloca com o contador de Bloco de Leitura (N7:20) para garantir que o dado PLC não se sobreponha ao dado que volta do módulo para o PLC. Veja lógica no Apêndice quanto a implementação de detalhes.

Etapa 2: Durante o processamento da linha BTW, a Lógica Ladder testa o valor no registro do ID do Bloco BTW e com base neste valor, copia dados da tabela de dados no buffer de transferência de dados BTW. Este processo requer que cada ID do Bloco BTW a ser processado seja provido com uma ramificação de lógica.



##### Testar ID Bloco BTW e mover dados para Buffer BTW

Esta ramificação localizada na linha BTW (linha 1) é um exemplo da lógica que deve ser implementada por cada bloco de dados a ser movido para o módulo. Ver lógica no Apêndice para exemplo de implementação.

#### 4.3.2 Estrutura de Dados de Transferência de Blocos

A estrutura do buffer de transferência de bloco quando escreve dados no módulo é a seguinte:

Palavra	Nome	Descrição										
0	ID do Bloco BTW	<p>O número identificador de bloco permite ao módulo MCM decodificar em que página de 50 palavras no espaço de dados de 4000 palavras o dado será gravado. O espaço de dados a ser escrito pode ser determinado multiplicando o ID do Bloco BTW por 50. O resultado será a primeira palavra da página. Ver exemplo abaixo:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>ID do Bloco BTW</td> <td>Espaço de dados</td> </tr> <tr> <td>0</td> <td>0 a 49</td> </tr> <tr> <td>1</td> <td>50 a 99</td> </tr> <tr> <td>10</td> <td>500 a 549</td> </tr> <tr> <td>20</td> <td>1000 a 1049</td> </tr> </table> <p>Paginando os diferentes blocos de dados no módulo, o processador pode controlar o conteúdo de memória de dados do módulo</p>	ID do Bloco BTW	Espaço de dados	0	0 a 49	1	50 a 99	10	500 a 549	20	1000 a 1049
ID do Bloco BTW	Espaço de dados											
0	0 a 49											
1	50 a 99											
10	500 a 549											
20	1000 a 1049											
1 a 50	Dado	O dado a ser escrito no módulo										

**4.4 Configuração de Lista de Comando- Modo Mestre (Códigos de ID Bloco BTW 80 a 99)**

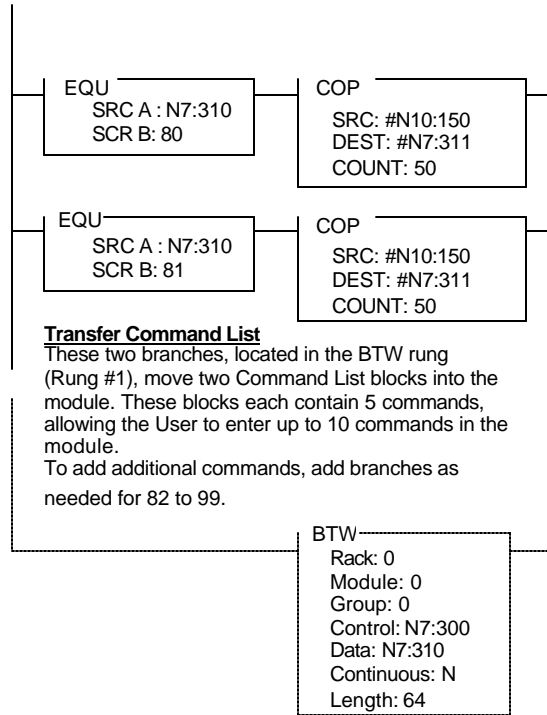
Uma porta Mestre Modbus MCM estabelece comunicação e realiza várias funções de comunicação com base nos dados que o usuário colocou na Lista de Comando. A Lista de Comando consiste de até 100 blocos de dados de comando individualmente configurados (10 palavras reservadas por comando) compartilhadas entre as duas portas disponíveis (no caso de o módulo ser configurado com duas portas Mestres).

**4.4.1 Lógica Ladder de Lista de Comando**

Esta lista, fornecida para a Tabela de Dados do processador, é transferida para a memória do módulo usando códigos 80 a 99 do ID do Bloco BTW, sendo que cada código representa um bloco de 50 palavras ou 5 comandos.

Exemplo de Lógica Ladder para mover os comandos para o módulo:

####



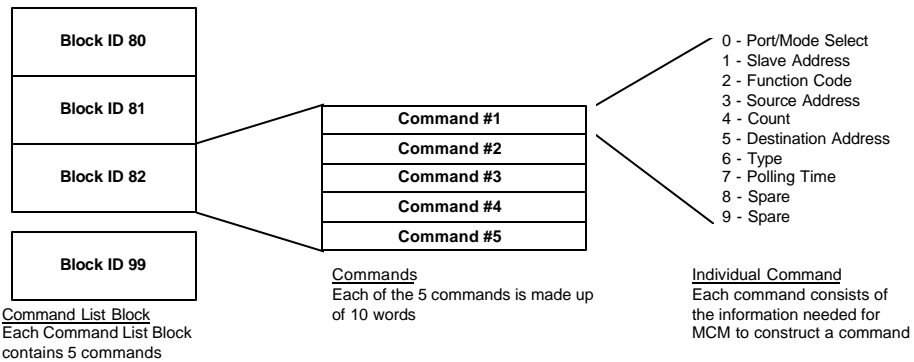
**Transfer Command List**

These two branches, located in the BTW rung (Rung #1), move two Command List blocks into the module. These blocks each contain 5 commands, allowing the User to enter up to 10 commands in the module.

To add additional commands, add branches as needed for 82 to 99.

**4.4.2 Estrutura da Lista de Comando**

A estrutura do bloco contendo a Lista de Comando está mostrada no diagrama abaixo



Veja seção 6 com respeito a detalhes da configuração de Comandos Modbus

Nome	Descrição																		
Seleção de Porta/Modo	<p>O parâmetro de seleção permite à aplicação selecionar qual porta o Módulo MCM usa para executar o comando, e se o comando deve ser realizado continuamente ou somente quando uma mudança de dados for detectada (Condicional) ou sob controle de Lógica Ladder direta (Controle). Os valores válidos são:</p> <table border="1"> <thead> <tr> <th>Porta/Modo</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Comando Desabilitar</td> </tr> <tr> <td>1</td> <td>Comando Contínuo Porta 1</td> </tr> <tr> <td>2</td> <td>Comando Contínuo Porta 2</td> </tr> <tr> <td>5</td> <td>Comando Condicional Porta 1</td> </tr> <tr> <td>6</td> <td>Comando Condicional Porta 2</td> </tr> <tr> <td>9</td> <td>Comando de Controle Porta 1</td> </tr> <tr> <td>10</td> <td>Comando de Controle Porta 2</td> </tr> </tbody> </table> <p>Contínuo vs Condicional: Códigos de função 5, 6, 15, 16. Quando se configura comandos de escrita na Lista de Comando, o driver Mestre MCM suporta dois tipos de comandos de escrita de dados: Contínuo e Condicional. As diferenças entre eles são:</p> <p><u>Contínuo</u>: Comandos fornecidos desta maneira serão executados a cada vez que a Lista de Comando for escaneada.</p> <p><u>Condicional</u>: Comandos Condicionais serão executados somente quando for detectada mudança no bloco de dados a ser gravado. Cada vez que a Lista de Comando for escaneada o módulo compara o dados a ser escrito contra o último dado escrito. Se for detectada mudança em qualquer valor, bit, ou palavra, será escrito o bloco de dados inteiro controlado pelo comando</p> <p><u>Modo de Lista de Comando</u></p> <p>No Modo de Comando de Controle, o comando somente será executado quando o Bit de Habilitação de Comando (ver Seção 4.5) passar de 0 a 1. O comando será executado uma vez por transição (i.e., o módulo realiza uma lógica de um-pulso para garantir que o comando somente execute um). Para cancelar esta um-pulso no módulo, e o Bit de Comando Habilitado deve mudar o estado de um para zero.</p>	Porta/Modo	Descrição	0	Comando Desabilitar	1	Comando Contínuo Porta 1	2	Comando Contínuo Porta 2	5	Comando Condicional Porta 1	6	Comando Condicional Porta 2	9	Comando de Controle Porta 1	10	Comando de Controle Porta 2		
Porta/Modo	Descrição																		
0	Comando Desabilitar																		
1	Comando Contínuo Porta 1																		
2	Comando Contínuo Porta 2																		
5	Comando Condicional Porta 1																		
6	Comando Condicional Porta 2																		
9	Comando de Controle Porta 1																		
10	Comando de Controle Porta 2																		
Endereço de escravo	O endereço de escravo representa o endereço Modbus de escravo da estação escravo ao qual o comando é direcionado. Endereços devem ser dados em forma decimal																		
Código de Função	<p>O código de função dado para a tabela diz ao módulo MDCM qual comando executar. As diferentes escolhas estão detalhadas na seção 6, mas de modo geral são:</p> <table border="1"> <thead> <tr> <th>Cod de Função.</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Read Output Status</td> </tr> <tr> <td>2</td> <td>Read Input Status</td> </tr> <tr> <td>3</td> <td>Read Multiple Data Registers</td> </tr> <tr> <td>4</td> <td>Read Input Registers</td> </tr> <tr> <td>5</td> <td>Force Single Coil (Latch/ Unlatch)</td> </tr> <tr> <td>6</td> <td>Preset (Write) Single Data Register</td> </tr> <tr> <td>15</td> <td>Multiple Coil Latch/ Unlatch</td> </tr> <tr> <td>16</td> <td>Preset (Write) Multiple Data Register</td> </tr> </tbody> </table>	Cod de Função.	Descrição	1	Read Output Status	2	Read Input Status	3	Read Multiple Data Registers	4	Read Input Registers	5	Force Single Coil (Latch/ Unlatch)	6	Preset (Write) Single Data Register	15	Multiple Coil Latch/ Unlatch	16	Preset (Write) Multiple Data Register
Cod de Função.	Descrição																		
1	Read Output Status																		
2	Read Input Status																		
3	Read Multiple Data Registers																		
4	Read Input Registers																		
5	Force Single Coil (Latch/ Unlatch)																		
6	Preset (Write) Single Data Register																		
15	Multiple Coil Latch/ Unlatch																		
16	Preset (Write) Multiple Data Register																		
Endereço de Fonte	<p>O valor representa o registro ou endereço de bit para ambos comandos de leitura e escrita, a partir dos quais os dados podem ser obtidos. A distinção entre os dois é a seguinte:</p> <ul style="list-style-type: none"> <li>- Quando se emite um comando de leitura, o Registro de Fonte é a localização de registro do escravo onde o comando começa a tomar dados.</li> <li>- Quando se emite um comando de escrita, o endereço de registro de Fonte é o registro no módulo onde o comando começa a tomar os dados a serem escritos no escravo.</li> </ul>																		

Quantidade	O número de palavras ou bits que o comando Modbus deve ler ou escrever. Veja seção 6 para uma discussão detalhada quanto à palavra e bits obtendo os dados a serem especificados para os diferentes comandos										
Endereço de Destino	O valor representa o endereço de registro ou bits, para ambos comandos de leitura e escrita, aos quais os dados devem ser gravados. A distinção entre os dois é: - Quando se emite um comando de leitura, o Endereço de Destino é a localização de registro no módulo quando o comando começa a colocar os dados a partir do escravo. - Quando se emite um comando de escrita, o Endereço de Destino é o registro no escravo onde o comando começa a colocar os dados a serem escritos no escravo.										
Tipo	O campo de Tipo é relevante somente durante um comando de Código de Função 3 (Múltipla Leitura de Registro). O Campo Tipo diz para o módulo para executar troca de palavra nos dados que estão sendo recebidos por aquele particular comando. Isto é muito útil e importante quando se lê dados de ponto flutuante (duas palavras por valor) a partir de alguns instrumentos (alguns instrumentos armazenam as palavras em seus dados de ponto flutuante na direção oposta do processador. Nestes casos, a troca de palavras permite a um comando COP de Lógica Ladder copiar os dados diretamente de um arquivo inteiro para um arquivo de Ponto Flutuante). As opções disponíveis são:  <table border="1"> <thead> <tr> <th><u>Tipo</u></th> <th><u>Descrição</u></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Valor Default. Não realizar nenhuma troca</td> </tr> <tr> <td>1</td> <td>Trocar palavras em cada par de palavras recebido do escravo durante este comando.</td> </tr> <tr> <td>2</td> <td>Trocar palavras em cada par recebido e então trocar os bytes dentro das palavras</td> </tr> <tr> <td>3</td> <td>Trocar bytes dentro de cada palavra (não trocar palavras)</td> </tr> </tbody> </table> Se for usada função 6 ou 16 e o registro de destino for maior que 47000, usar 1 neste campo para desabilitar Extensão ENROM	<u>Tipo</u>	<u>Descrição</u>	0	Valor Default. Não realizar nenhuma troca	1	Trocar palavras em cada par de palavras recebido do escravo durante este comando.	2	Trocar palavras em cada par recebido e então trocar os bytes dentro das palavras	3	Trocar bytes dentro de cada palavra (não trocar palavras)
<u>Tipo</u>	<u>Descrição</u>										
0	Valor Default. Não realizar nenhuma troca										
1	Trocar palavras em cada par de palavras recebido do escravo durante este comando.										
2	Trocar palavras em cada par recebido e então trocar os bytes dentro das palavras										
3	Trocar bytes dentro de cada palavra (não trocar palavras)										
Preset Tempo de varredura	O valor preset de Tempo de Varredura permite a cada comando ter uma frequência de execução configurável. No módulo, um timer é mantido para cada comando. O timer é decrementado uma vez por segundo, até chegar a zero. Quando o timer chega a zero, o comando é habilitado para execução e o timer reset para o valor Preset de Tempo de Varredura. Valores válidos são de 0 65535 (0xffff).										

#### 4.4.3 Editando a Lista de Comando

Entrar na Lista e Comando é uma questão de fornecer valores corretos para a tabela de dados PLC. Usando o software de programação de lógica Ladder, entrar com os valores necessários para ajustar um ou mais comandos válidos.

##### **Dicas para deixar a vida mais fácil**

Quando se ajusta pela primeira vez a Lista de Comando recomendamos começar com um comando. Este comando permitirá que o módulo comece a transmitir se tudo o mais estiver OK (i.e., Lógica Ladder, cabos conectados, etc.). Uma vez o módulo transmitindo, então tente se comunicar com escravo, então entre um outro qualquer.

Um exemplo de uma Lista de Comando está mostrado abaixo. Note que os comandos podem ser fornecidos em fileiras e uma vez a definição de coluna entendida, revisar a Lista de Comando fica muito fácil.

	0	1	2	3	4	5	6	7
	Num de Porta	End Escr	Cod de Função	End SRC	CNT	End Dest	Tipo	Tempo Varr
N7:50	1	3	1	0	10	100	0	0
N7:60	1	3	4	20	20	100	0	0
N7:70	2	2	16	200	10	137	0	0
N7:80	6	2	16	210	10	150	0	0

**Lista de Comando de Exemplo**

Um exemplo de múltiplos blocos de dados de configuração de mensagem está mostrado na tabela abaixo (as colunas 8-9 não estão mostradas para maior clareza).

**4.5 Modo Controle de Comando - Modo Mestre**

Em algumas condições de operação especiais, pode ser necessário que a Lógica Ladder seja capaz de coordenar e controlar de perto a execução de comandos na Lista de Comando. Para atender este requisito, o módulo MCM suporta o chamado Modo de Controle de Comando.

Quando configurado no Modo de Controle de Comando, a Lógica Ladder é capaz de prover um controle de habilitação de Comando com base nas entradas da Lista de Comando. Adicionalmente, quando usada em conexão com Bits de Comando "Realizado" (Ver seção 5.3) a Lógica Ladder é capaz de uma-tentativa para cada comando, se desejado.

**4.5.1 Estrutura de Bloco BTW**

A estrutura de bits de comando Habilitado, quando eles são escritos dentro do módulo no buffer de Transferência de Bloco BTW, é a seguinte:

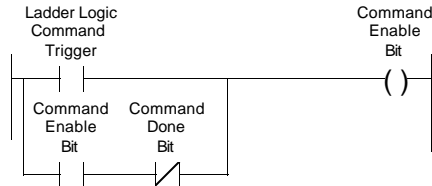
	Nome	Descrição														
0	ID do Bloco BTW	Os bits de Comando Habilitado são movidos para o módulo durante cada transação BTW. Daí, todos números ID do Bloco BTW poderão ser usado aqui														
150	Dados	Dados de módulo e Lista de Comando, como já mostrado														
51-56	Bits de Comando Habilitado	Estes comandos contêm Bits de Comando Habilitado para cada comando na Lista de Comando, até os primeiros 96 comandos. Os Bits "Habilitado" são bits mapeados nas palavras que dependem da posição relativa na Lista de Comando. O mapeamento nas palavras é feito da seguinte forma: <table border="1" style="margin-left: 40px;"> <thead> <tr> <th>Palavra</th> <th>Comandos</th> </tr> </thead> <tbody> <tr> <td>51</td> <td>1 a 16</td> </tr> <tr> <td>52</td> <td>17 a 32</td> </tr> <tr> <td>53</td> <td>33 a 48</td> </tr> <tr> <td>54</td> <td>49 a 64</td> </tr> <tr> <td>55</td> <td>65 80</td> </tr> <tr> <td>56</td> <td>81 a 96</td> </tr> </tbody> </table> Exemplo: Palavras 51 bit 0 é Comando #1 Habilitado	Palavra	Comandos	51	1 a 16	52	17 a 32	53	33 a 48	54	49 a 64	55	65 80	56	81 a 96
Palavra	Comandos															
51	1 a 16															
52	17 a 32															
53	33 a 48															
54	49 a 64															
55	65 80															
56	81 a 96															

**4.5.2 Controlando os Comandos**

Quando um comando é configurado no Modo de Controle de Comando e quando o módulo detecta o estado de mudança de bit de Comando Habilitado de 0 a 1, o módulo tenta executar o comando (três tentativas são feitas para executar o comando. Se o comando for enviado com sucesso, o bit de Comando Realizado será estabelecido. Se ocorrer erro no processo de envio, então o bit de Comando Erro será estabelecido).

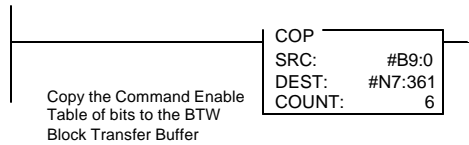
Um exemplo da Lógica Ladder que pode ser implementada para controlar um comando aparece na estrutura como segue:

####



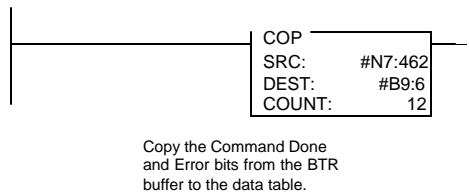
A implementação mais simples seria manter uma tabela binária de Bits de Comando Habilitado que é copiada para o buffer BTW a cada transação. A seguinte ramificação de lógica pode ser adicionada à Linha BTW (transfere dados para o módulo).

#####



Os bits Realizado e Erro de Comando podem ser copiados para o mesmo arquivo binário e com referência na Lógica Ladder logo depois de transferidos. As seguintes instruções podem ser adicionadas à linha BTR (ler dados no módulo) realizando o que segue:

####



**4.5.3 Exemplo de Lista de Comando**

Comandos podem ser controlados através da configuração do Comando Habilitado

	0	1	2	3	4	5	6	7
	<u>N PORTA</u>	<u>END</u> <u>ESCR</u>	<u>COD</u> <u>FUNÇÃO</u>	<u>END</u> <u>SRC</u>	<u>CNT</u>	<u>END</u> <u>DEST</u>	<u>TIPO</u>	<u>TEMPO</u> <u>VARR</u>
N7:50	9	3	1	0	10	100	0	0
N7:60	10	3	4	50	20	100	0	0

**Exemplo de Lista de Comando**

Um exemplo onde o comando em N7:50 é configurado como Modo de Comando de Controle para a Porta 1 enquanto o Comando N7:60 está configurado para a Porta 2

**4.6 Comandos Iniciados por Evento- Modo Mestre (Cod ID Bloco BTW 100- 119)**

Adicionalmente a comandos continuamente habilitados que podem ser configurados na lista de Comando, o módulo MCM também suporta comandos Iniciados por Evento. Estes Comandos por Evento podem ser usados para leitura/escrita de dados condicionalmente com um escravo. Aplicações de exemplo incluem estabelecer o tempo em um escravo, zerar contador de batch, etc..



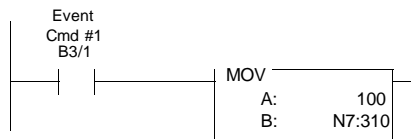
**Dicas para deixar a vida mais fácil**

Um dos benefícios proporcionados por um Comando de Escrita Iniciado por Evento (FC 5. 6. 15. 16) é o fato de garantir que o conteúdo dos dados gravados no escravo esteja coordenado com a execução do comando. Note que isto não é necessariamente o caso de quando se executa os comandos fora da Lista de Comando.

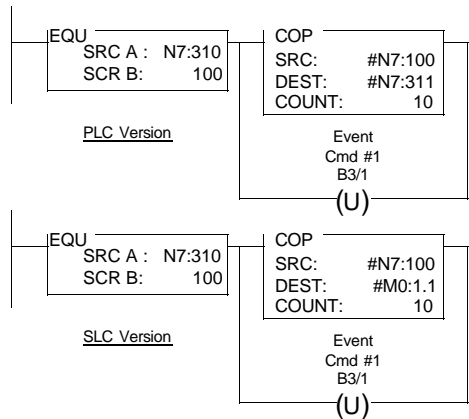
**4.6.1 Lógica Ladder**

A execução de um Comando Iniciado por Evento é realizada transferindo dados para o buffer BTW, enquanto o número de ID do Bloco fica entre 100 e 119 inclusive. O bloco de dados que é transferido, que será detalhado na próxima seção, contém os dados necessários para o módulo codificar um comando válido. As adições à Lógica Ladder, que devem ser feitas para suportar esta funcionalidade, são as seguintes:

####



This branch is added to the Read rung, just above the MOV 255 to N7:310 branch. The B3/1 bit, selected here for example purposes only, is one-shot set in the ladder logic



This branch is added to the BTW rung, and serves to copy the Event Initiated Command block structure to the module and then Unlatches the command enable bit which was set in the ladder program.

#### 4.6.2 Estrutura de Bloco BTW

A estrutura do bloco tendo Comando Iniciado por Evento está mostrada na tabela a seguir:



Veja seção 6 com respeito a detalhes na configuração de comandos

Palavra BTW	Offset de Dado	Nome	Descrição																		
0		ID do Bloco BTW	O ID do Bloco BTW é usado pelo módulo para determinar que o buffer de transferência de dados contém um Comando Iniciado por Evento. Valores válidos estão entre 100 e 119. O valor determina a posição relativa na Tabela de Erro Mestre.																		
1	N[:]:0	Número de porta	O parâmetro de n° de porta permite à aplicação selecionar qual porta o módulo deve usar para executar a comando. Os valores esperados são: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Porta/Modo</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Porta 1</td> </tr> <tr> <td>2</td> <td>Porta 2</td> </tr> </tbody> </table>	Porta/Modo	Descrição	1	Porta 1	2	Porta 2												
Porta/Modo	Descrição																				
1	Porta 1																				
2	Porta 2																				
2	N[:]:1	Endereço de escravo	O endereço de escravo representa o endereço de escravo de Modbus da estação escravo ao qual o comando é dirigido. Endereços devem ser dados em forma decimal.																		
3	N[:]:2	Código de função	O código de função dado para a tabela diz para o módulo MCM qual comando executar. As diferentes escolhas estão detalhadas na Seção 6, em geral são: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Cód de Função</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Read Output Status</td> </tr> <tr> <td>2</td> <td>Read Input Status</td> </tr> <tr> <td>3</td> <td>Read Multiple Data Registers</td> </tr> <tr> <td>4</td> <td>Read Input Registers</td> </tr> <tr> <td>5</td> <td>Force Single Coil (Latch/ Unlatch)</td> </tr> <tr> <td>6</td> <td>Preset (Write) Single Data Register</td> </tr> <tr> <td>15</td> <td>Multiple Coil Latch/ Unlatch</td> </tr> <tr> <td>16</td> <td>Preset (Write) Multiple Data Register</td> </tr> </tbody> </table>	Cód de Função	Descrição	1	Read Output Status	2	Read Input Status	3	Read Multiple Data Registers	4	Read Input Registers	5	Force Single Coil (Latch/ Unlatch)	6	Preset (Write) Single Data Register	15	Multiple Coil Latch/ Unlatch	16	Preset (Write) Multiple Data Register
Cód de Função	Descrição																				
1	Read Output Status																				
2	Read Input Status																				
3	Read Multiple Data Registers																				
4	Read Input Registers																				
5	Force Single Coil (Latch/ Unlatch)																				
6	Preset (Write) Single Data Register																				
15	Multiple Coil Latch/ Unlatch																				
16	Preset (Write) Multiple Data Register																				

4	N[]:3	Endereço de fonte	O valor representa o registro ou endereço de bit, para comandos de leitura, a partir dos quais os dados serão obtidos: - Quando emite um comando de leitura, o Endereço de Registro de Fonte é a localização de registro no escravo onde o comando começa a tomar dados Quando executa um comando Iniciado por Evento, este valor não tem significado
5	N[]:4	Quantidade	Nº de palavras ou bits que o comando Modbus deve ler ou escrever. Ver seção V para discussão detalhada dos comprimentos de bit e palavra a serem especificados para diferentes comandos
6	N[]:5	Endereço de destino	O valor representa o registro ou endereço de bit, para ambos comandos de leitura e escrita, ao qual os dados serão escritos. A distinção entre os dois é a seguinte - Quando emite um comando de leitura, o Endereço de Destino é a localização do registro no módulo onde o comando começa a colocar os dados a partir do escravo. - Quando emite um comando de escrita, o Endereço de Destino é o registro no escravo onde o comando começa a colocar aos dados a serem escritos no escravo.
7	N[]:6	Tipo	O campo Tipo é relevante somente durante comando 3 Código de Função (Leitura de múltiplos registros). O campo Tipo diz para o módulo executar troca de palavra nos dados que estão sendo recebidos por aquele particular comando. Ver seção 4.4.2 para uma completa discussão deste parâmetro
8-51	N[]:7	Dados de escrita	Este registro contém os valores de dados gravados que serão enviados para o escravo pela seleção de Código de Função

	0	1	2	3	4	5	6	7	8	9
	N DE PORTA	END DE ESCR	COD DE FUNC	END SRC	CNT	END DE DEST	TIPO	DADO	DADO	DADO
N7:100	1	3	1	0	10	100	0	0	0	0
N7:110	1	3	6	0	1	1	0	1267		

**Exemplos de Comandos de Escrita Iniciados por Evento**

O 1º comando emite FCI para o Escravo 3, lendo 10 bits do bit 0 no registro 100 no módulo.  
O 2º comando escreve o valor 1267 no registro 1 no escravo.

## 5. Lendo a partir do Módulo

Esta seção provê detalhes do nível de referência na transferência de dados do processador PLC/SLC para o módulo MCM. Este tipo de transferência permite à Lógica Ladder enviar configuração, lista de comando e dados para o módulo.

### 5.1 Transferindo dados do módulo (ID do Bloco BTR 0 a 79)

Quando o driver de porta Mestre lê os dados a partir do escravo ou quando um host grava no driver de porta escravo, o dado resultante é colocado no espaço de dados do módulo ProSoft (endereços 0 a 3999). Este espaço de dado de módulo é o mesmo bloco de memória no qual PLC/SLC pode escrever.

A transferência de dados do módulo ProSoft\_Technology para o processador é executada através da função Transferência de Bloco de Leitura. A seção a seguir detalha o manuseio dos dados lidos.



Embora as 64 palavras físicas de buffer de dados não precisem ser usadas, os comprimentos BTR e M1 devem ser configurados para um comprimento de 64 palavras, ou de outro modo a operação será imprevisível.

#### 5.1.1 A Estrutura de Bloco de Dados de Leitura

A definição de buffer BTR é a seguinte:

Palavra	Nome	Descrição										
0	ID do Bloco BTR	<p>A Lógica Ladder usa este valor para determinar o conteúdo da porção de dados do buffer BTR. Com algum teste condicional na Lógica Ladder, os dados a partir do módulo podem ser colocados na tabela de dados PLC/SLC</p> <div style="text-align: center;"> </div> <p>A relação entre número de ID do Bloco BTR e a tabela de registro pode ser colocada na equação:</p> <p style="text-align: center;">Endereço de Registro de Início = N° ID do Bloco *50</p> <p>Códigos válidos se encontram entre 0 e 79</p>										
1	ID do Bloco BTW	<p>O módulo retorna este valor para o processador para ser usado para habilitar o movimento do dado de registro e blocos de lista de comando para o módulo. O número ID do Bloco BTW é desenvolvido com base nos parâmetros fornecidos nos parâmetros 21 e 22 do Bloco 255. Este valor é somente ser uma sugestão para facilitar os requisitos de programação de Lógica Ladder. Se for desejado desenvolver uma série diferente de dados, isto será facilmente realizado na Lógica Ladder.</p> <p>Códigos válidos são:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Código BTW</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>0-79</td> <td>Dado de Módulo</td> </tr> <tr> <td>80-99</td> <td>Lista de Comando</td> </tr> <tr> <td>100-119</td> <td>Comando Iniciado por Evento</td> </tr> <tr> <td>255</td> <td>Configuração de Módulo</td> </tr> </tbody> </table>	Código BTW	Descrição	0-79	Dado de Módulo	80-99	Lista de Comando	100-119	Comando Iniciado por Evento	255	Configuração de Módulo
Código BTW	Descrição											
0-79	Dado de Módulo											
80-99	Lista de Comando											
100-119	Comando Iniciado por Evento											
255	Configuração de Módulo											

Continuação

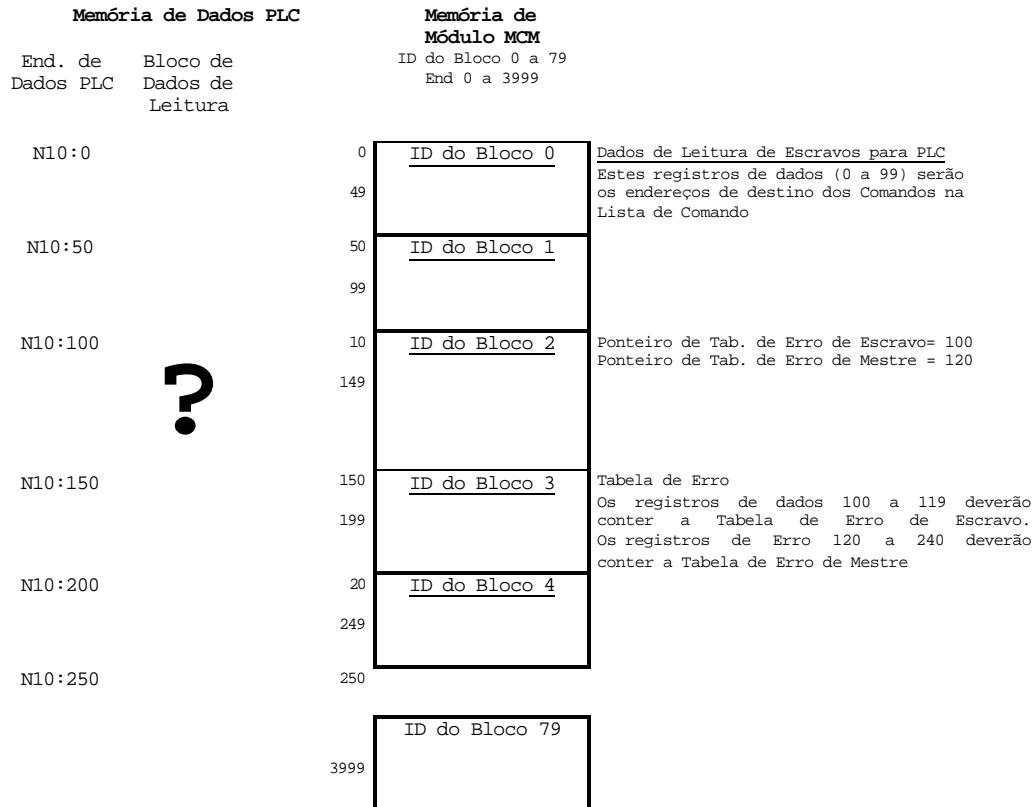
2 a 51	Dados	O conteúdo do espaço de Dados de Registro do módulo (0 a 3999). Estes dados contêm dados recebidos dos escravos, dados movidos do processador, e Tabelas de Erro de Escravo e Mestre. Os valores serão de 16 bits e deverão ser colocados em arquivos inteiros. Note que a Lógica Ladder de aplicação de usuário controla a colocação e uso dos registros de dados.
52 a 63	Bits de Comando Erro e Realizado	Ver Seção 5.3

**5.1.2 Movendo os dados do módulo para o processador**

Dado lido a partir de dispositivos escravo (driver escravo) ou escrito a partir de um host (driver Escravo) é colocado em uma tabela de registro de 4000 palavras. Esta tabela começa de zero e vai até 3999.

A tabela de registro de dados é transferida do módulo para a Lógica através de um mecanismo de paginação projetado para superar o limite físico de 64 palavras da instrução BTR. O de paginação foi mostrado na discussão acima, mas o importante é entender o relacionamento entre os números de página (números de ID do Bloco BTR) e os endereços de registros no módulo.

O diagrama também mostra o Layout para aplicação de exemplo. Note que o número de blocos que retornaram do módulo para a Lógica Ladder é determinado pelo valor fornecido para o registro Quantidade de Blocos de Leitura para a configuração do Sistema. No exemplo foi assumido um valor de Quantidade de Blocos de Leitura igual a 5.

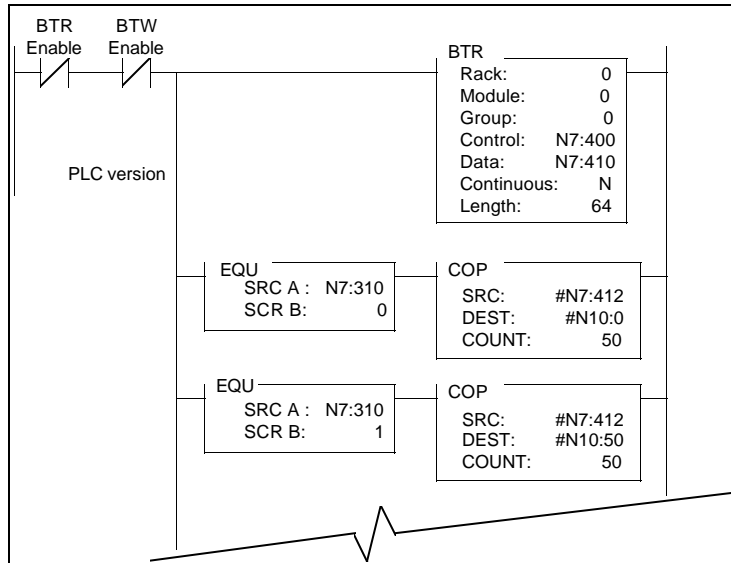


Blocos de dados de leitura a partir do Módulo MCM

Note que este diagrama assume um valor de Quantidade de Bloco de Leitura igual a 5, daí retornando Registros de 0 a 249 a partir do módulo. Este valor pode ser alterado dependendo da aplicação.

**5.1.3 Lógica Ladder para Ler Dados de Módulo**

A Lógica Ladder deve ser programada para olhar o buffer BTR, decodificar diversas palavras, e então agir. A seguir é mostrado um exemplo de tal Lógica Ladder:



**Ladder de exemplo para transferir dados do módulo**

Esta lógica mostra um método para mover dados do módulo para a tabela de dados PLC

**5.1.4 Tabela de Código de Erro de Escravo**

O módulo MCM monitora status de todos os comandos de porta escravo. Este status é transmitido para os processadores de uma Tabela de Código de Erro de Escravo.



A tabela de Código de Erro de Escravo é inicializada em zero na energização e a cada vez que o módulo recebe 255 blocos de dados de configuração.

A Tabela de Erro de Escravo é um bloco de 20 palavras. A localização da Tabela de Erro é determinada pelo parâmetro de Ponteiro de Tabela de Erro de Escravo no bloco de Configuração. A estrutura do bloco de dados é a seguinte.

**Códigos de Status da Porta 1**

Palavra	End de Exemplo	Nome	Descrição
0	N10:100	Status de Porta Corrente	Este valor representa o valor corrente do código de erro para a porta. Este valor será somente válido se a porta for configurada como Escravo. Os valores possíveis são detalhados na seção seguinte.
1	N10:101	Último Erro Transmitido	Este valor é o último código de erro transmitido para o mestre para esta porta escravo. Códigos de erro que podem ser esperados neste campo são 0, 1, 2, 3, 6. O campo somente será limpo reconfigurando o módulo (ID do Bloco 255)
2	N10:102	Total de Mensagens para este Escravo	Este valor representa o número total de mensagens que corresponderam a este endereço de escravo nesta porta, Quer se o escravo efetivamente determinou como válido (requer resposta) ou não.

**Códigos de Status da Porta 1**

3	N10:103	Total de respostas a partir deste escravo	Este valor representa o número de respostas boas (sem erro) que o escravo enviou para o mestre por esta porta. Presumindo-se que se o escravo responde, então a mensagem foi boa.
4	N10:104	Total de mensagens vistas por este escravo	Este valor representa o número total de comandos vistos pelo escravo nesta porta, a despeito do endereço do escravo.

**Códigos de Status da Porta 2**

Palavra	End de Exemplo	Nome	Descrição
5	N10:105	Status de Porta Corrente	Ver referência acima
6	N10:106	Último Erro Transmitido	Ver referência acima
7	N10:107	Total de Mensagens para este Escravo	Ver referência acima
8	N10:108	Total de Respostas a partir deste Escravo	Ver referência acima
9	N10:109	Total de Mensagens visto por este Escravo	Ver referência acima

**Informação de Sistema**

Palavra	End de Exemplo	Nome	Descrição
10-11	N10:110 N10:111	Nome de Produto (ASCII)	Estas duas palavras representam o nome de produto do módulo em ASCII. No caso de produto MCM, as letras MCM devem ser mostradas quando se coloca o software de programação no modo de representação de dados ASCII.
12-13	N10:112 N10:113	Revisão (ASCII)	Estas duas palavras representam o nível de revisão de produto do firmware em uma representação ASCII. UM exemplo dos dados apresentados seria 1.45 quando se coloca o software no modo de representação de dados ASCII
14-15	N10:114 N10:115	Revisão de Sistema Operacional (ASCII)	Estas duas palavras representam o nível de revisão do sistema operacional interno do módulo em uma representação ASCII.
16-17	N10:116 N10:117	Número de Produção (ASCII)	Este número representa o número de lote ao qual pertence seu particular chip em uma representação ASCII
18-19	N10:118 N10:119	Reserva	

Todos contadores na Tabela de Erro de Escravo retornam para zero depois de chegar a 65535

**5.1.5 Tabela de Código de Erro Mestre**

O módulo MCM monitora o status de todos os comandos de porta Mestre. Este status é transmitido para os processadores na forma de uma Tabela de Código de Erro de Mestre, a posição da qual é controlada pelo Ponteiro de Tabela de Erro de Mestre no Setup de Configuração de Comunicação. Cada comando Mestre gera um Código de Erro para o usuário.

A Tabela de Código de Erro de Mestre é inicializada em zero na energização e a cada vez que o módulo recebe 255 blocos de dados de configuração.

A Tabela de Código de Erro é um bloco de 120 palavras. O relacionamento entre a colocação dos códigos de erro dentro da Tabela de Erro e os comandos estão de acordo com a posição relativa do comando na Lista de Comando.

O método mais simples de obter a Tabela de Status de Erro de Mestre é localizá-la na extremidade do mapa de dados de aplicação e então trazê-la de volta para tabela de dados PLC/SLC como parte de dado regular. A estrutura da Tabela de Erro de Mestre é a seguinte:

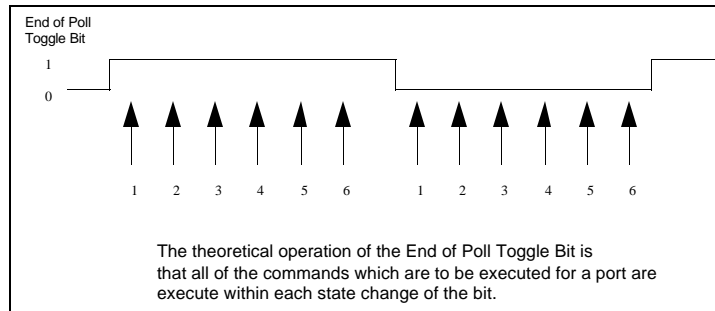
Palavra	Descrição
0	Fim da Lista de Comando de Status de Varredura
1	Status de Erro #1 de Comando
2	Status de Erro #2 de Comando
98	Status de Erro #98 de Comando
99	Status de Erro #99 de Comando
100-120	Futuro

Onde

**Fim da Lista de Comando de Status de Varredura:** Este registro provê uma indicação de quando o Mestre completou um ciclo através da Lista de Comando. Um bit na palavra será revertido cada vez que a Lista de Comando tiver sido completada. O status é indicado para cada porta mestre como segue:

Bit	
0	Porta Mestre 1
1	Porta Mestre 2

####



Status de Erro de Comando: Os Códigos de Status de Erro, recebidos dos escravos ou gerados pelo módulo, são colocados na tabela. Ver a próxima seção com respeito aos significados dos códigos de erro. Os valores são valores de 16 bits que devem ser colocados em um arquivo inteiro. Note que a Lógica Ladder de aplicação de usuário controla colocação e uso destes registros.

Exemplo de Tabela de Status de Erro									
Ponteiro de Tabela de Erro de Mestre									
	Pal 0	Pal 1	Pal 2	Pal 3	Pal 4	Pal 5	Pal 6	Pal 7	Pal 8
N10:120	0	0	8	0	0	0	0	0	0
N10:130	0	0	0	0	0	0	0	0	0
N10:140	0	0	0	0	0	0	0	0	0
N10:150	0	0	0	0	0	0	0	0	0
N10:160	0	0	0	0	0	0	0	0	0
N10:170	0	0	0	0	0	0	0	0	0
N10:180	0	0	0	0	0	0	0	0	0
N10:200	0	0	0	0	0	0	0	0	0
N10:210	0	0	0	0	0	0	0	0	0
N10:220	0	0	0	0	0	0	0	0	0
N10:230	0	0	0	0	0	0	0	0	0

Estes registros correspondem aos registros usados no programa de amostra para PL5 atrás deste manual. Sua aplicação pode requerer seu próprio programa específico. Neste caso um código de erro 8 foi gerado para o comando 2 - todos os demais comandos foram executados sem erro. A coluna zero é usada para identificar que a porta mestre alcançou o fim da Lista de Comando e está começando no topo da Lista de Comando.

### 5.1.6 Códigos de Status de Erro

Os Códigos de Erro que retornam para a Tabela de Códigos de Erro Mestre e Escravo refletem o efeito dos comandos e respostas executadas pelo módulo. Note que em todos os casos em que retornou um zero não houve erro. Os códigos de Status de Erro válidos são os seguintes:

Código	Nome	Descrição
0	Tudo OK	O módulo está operando como desejado
1	Função Ilegal	Um pedido de código de função ilegal está sendo tentado
2	Endereço de Dados Ruim	O endereço ou uma gama de endereços cobertos por um pedido do mestre não se encontra dentro dos limites permitidos
3	Valor de Dado Ruim	O valor no campo de dados do comando não é permitido
4	Resposta Incompleta Detectada	O erro indica que uma resposta incompleta foi recebida a uma consulta de mestre. Frequentemente isto indica que o dispositivo escravo pode estar respondendo demasiadamente rápido ou que há excessivo ruído na linha.
6	Módulo Ocupado	O código de status ocupado do módulo retorna quando um comando de escrita a partir do mestre ainda não tiver sido completado quando um segundo comando de escrita for recebido
8	Erro de Timeout	As comunicações com o escravo endereçado não tiveram sucesso devido à falta de resposta a partir do escravo. A porta Mestre tenta um comando três vezes antes de se mover para o próximo comando.
10	Sobrecarga de Buffer	O buffer de recepção foi sobrecarregado, então coloque a quantidade de caracter em zero. Se esta condição ocorrer tente ler menos parâmetros de uma vez.
16	Erro de Configuração de Porta	Se este valor retornou do módulo, uma ou ambas portas seriais perdem a configuração. Para determinar a fonte exata do problema, verifique: <ul style="list-style-type: none"> <li>- Configuração de paridade</li> <li>- Configuração de stop bit</li> <li>- Configuração de Baud Rate</li> <li>- Endereço de Registro de Entrada de Início</li> <li>- Endereço de Registro de Saída de Início</li> </ul>
18	Erro de Configuração de Sistema	Se este erro retornar do módulo, um dos parâmetros de configuração de sistema foi detectado fora da gama, Para determinar a fonte, verifique: <ul style="list-style-type: none"> <li>- Quantidade de Bloco de Leitura = 80</li> <li>- Quantidade de Bloco de Escrita = 80</li> <li>- Quantidade de Bloco de Comando = 20</li> <li>- Ponteiro de Erro de Escravo = 3850</li> <li>- Ponteiro de Erro de Mestre = 3880</li> </ul>
254	Erro de Soma	O escravo determinou que o acumulado de mensagem estava errado, portanto descarte a mensagem
255	Timeout de Hardware TX	Ocorreu timeout de transmissão, indicando que módulo não foi capaz de transmitir o comando. Verifique se o jumper RTS-CTS na porta ainda está conectado.

### 5.2 Modo Pass\_Through- Modo Escravo (ID do Bloco, BTR 256 a 259)

Quando uma porta escravo é configurada para suportar o modo Pass\_Through, quaisquer comandos de escrita Modbus endereçados para os endereços de escravo local passam pelo barramento para processamento pela Lógica Ladder. A lógica Ladder no Apêndice provê um exemplo de como os comandos Pass\_Through podem ser decodificados.

### 5.2.1 Estrutura de Bloco

A definição de estrutura de buffer BTR para o Modo Pass\_Through é a seguinte:

Palavra	Nome	Descrição										
0	ID do Bloco BTR	O valor do registro ID do Bloco BTR representa o tipo de comando de escrita recebido a partir do host. Códigos válidos são:  <table border="1"> <thead> <tr> <th>ID BTR</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>256</td> <td>Escrita de Registro</td> </tr> <tr> <td>257</td> <td>Escrita de Registro ENRON FLOAT</td> </tr> <tr> <td>258</td> <td>Escrita de Bit Único</td> </tr> <tr> <td>259</td> <td>Escrita de Múltiplos Bits</td> </tr> </tbody> </table>	ID BTR	Descrição	256	Escrita de Registro	257	Escrita de Registro ENRON FLOAT	258	Escrita de Bit Único	259	Escrita de Múltiplos Bits
ID BTR	Descrição											
256	Escrita de Registro											
257	Escrita de Registro ENRON FLOAT											
258	Escrita de Bit Único											
259	Escrita de Múltiplos Bits											
1	ID do Bloco BTW	Igual à descrição acima										
2- 62	Dados	O conteúdo destes registros depende do número de ID do Bloco BTR (i.e., o comando recebido a partir do host)										

### 5.2.2 Recebendo Escritas de Registros (ID do Bloco BTW 256 e 257)

A definição de buffer BTR é a seguinte:

Palavra	Nome	Descrição										
0	ID do Bloco BTR	O valor do registro ID do Bloco BTR representa o tipo de comando de escrita recebido a partir do host. Códigos válidos são:  <table border="1"> <thead> <tr> <th>ID BTR</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>256</td> <td>Escrita de Registro</td> </tr> <tr> <td>257</td> <td>Escrita de Registro ENRON FLOAT</td> </tr> <tr> <td>258</td> <td>Escrita de Bit Único</td> </tr> <tr> <td>259</td> <td>Escrita de Múltiplos Bits</td> </tr> </tbody> </table>	ID BTR	Descrição	256	Escrita de Registro	257	Escrita de Registro ENRON FLOAT	258	Escrita de Bit Único	259	Escrita de Múltiplos Bits
ID BTR	Descrição											
256	Escrita de Registro											
257	Escrita de Registro ENRON FLOAT											
258	Escrita de Bit Único											
259	Escrita de Múltiplos Bits											
1	ID do Bloco BTW	Igual à descrição acima										
2	Quantidade	O número de registros é escrito pelo Mestre. Os números válidos que serão recebidos variam de 1 a 60.										
3	Endereço de Destino	Este valor é usado pela Lógica Ladder para determinar o endereço na tabela de dados dos processadores para iniciar escrita de dados.										
4-62	Dados	Os valores de dados escritos a partir do mestre. O valor será um valor de registro de 16 bit.										

### 5.2.3 Recebendo Escritas de Bit Único (ID do Bloco 258)

A definição de buffer BTR é a seguinte:

Palavra	Nome	Descrição				
0	ID do Bloco BTR	O valor do registro ID do Bloco BTR representa o tipo de comando de escrita que foi recebido do host. Códigos válidos são:  <table border="1"> <thead> <tr> <th>ID BTR</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>258</td> <td>Escrita de Registro Único</td> </tr> </tbody> </table>	ID BTR	Descrição	258	Escrita de Registro Único
ID BTR	Descrição					
258	Escrita de Registro Único					
1	ID do Bloco BTW	Igual à descrição acima				
2	Endereço de Bit	Representa o bit atuado				
3	Ação de Controle	A ação é comandada pelo mestre. Quando o valor for zero, reset o bit endereçado, e quando o valor for 1, set o bit endereçado				

#### 5.2.4 Recebendo Escritas de Múltiplos Bit (ID do Bloco BTR 259)

Palavra	Nome	Descrição				
0	ID do Bloco BTR	O valor do registro ID do Bloco BTR representa o tipo de comando de escrita recebido do host. Códigos válidos são:  <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>ID BTR</th> <th>Descrição</th> </tr> </thead> <tbody> <tr> <td>259</td> <td>Escrita de Múltiplos Bit</td> </tr> </tbody> </table>	ID BTR	Descrição	259	Escrita de Múltiplos Bit
ID BTR	Descrição					
259	Escrita de Múltiplos Bit					
1	ID do Bloco BTW	Igual ao descrito acima				
2	Quantidade de Palavras	Representa o número de palavras no bloco de dados que contém dados de bit válidos. Os números válidos variam de 1 a 60 (isto limita o número de bits escrito em um comando 30*16)				
3	Endereço de Início de Palavras	Representa o endereço de palavra off_set no qual o bloco de dados começa a ser escrito. Quando o mestre endereça uma escrita de bit, ele envia o endereço de bit de início. O endereço de bit de início é usado pelo módulo para gerar este endereço de início de palavra (endereço de Bit/ 16)				
4-33	Dados	Estes registros contêm os dados de escrita de bit recebidos do mestre. Note que escritas de <u>bit de comprimento de palavra parcial são aceitáveis</u> . Os bits de mascara e alguma Lógica Ladder protege bits não endereçados dentro de uma palavra comum.  Estas palavras mascaram bits não endereçados. Isto permite iniciar endereços que não estão em um limite de palavra e comprimentos que não acabam no limite de palavra. Ver a Lógica Ladder de exemplo no Apêndice				
34-63	Mascara					

#### 5.3 Decodificando Bits de Comando Done e Error - Modo Mestre

Os bits de comando Realizado e Erro retornam para uso no programa de Lógica Ladder durante cada transferência de bloco de dados (ID do Bloco BTR 0 a 79). Estes bits podem ser usados para acompanhar execução do comando ou desabilitar comandos quando um comando estiver configurado no Modo de Controle de Comando (Ver Seção 4.5).

##### 5.3.1 Estrutura de Bloco

A estrutura de bits Realizado e Erro, quando retornam no buffer de Transferência de Bloco BTR, é a seguinte:

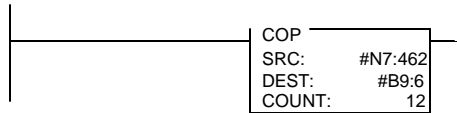
Pal	Nome	Descrição														
0	ID do Bloco BTR	Quando o valor do ID do Bloco BTR se encontra entre 0 e 79, o Buffer BT contém os bits de Comando Realizado e Erro, como mostrado abaixo														
1	ID do Bloco BTW	Igual ao descrito acima														
2-51	Dados	Dados de Módulo, como mostrado acima														
52-57	Bits de Comando Realizado	Estes registros contêm flags de bit Realizado para cada comando na Lista de Comando, até os primeiros 96 comandos. Os bits Realizado são mapeados por bit nas palavras dependendo de sua posição relativa na Lista de Comando. O mapeamento com Bits Realizado é o seguinte:  <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Palavra</th> <th>Comandos</th> </tr> </thead> <tbody> <tr> <td>52</td> <td>1 a 16</td> </tr> <tr> <td>53</td> <td>17 a 32</td> </tr> <tr> <td>54</td> <td>33 a 48</td> </tr> <tr> <td>55</td> <td>49 a 64</td> </tr> <tr> <td>56</td> <td>65 a 80</td> </tr> <tr> <td>57</td> <td>81 a 96</td> </tr> </tbody> </table> Exemplo: Palavra 52 bit 0 é o comando #1:	Palavra	Comandos	52	1 a 16	53	17 a 32	54	33 a 48	55	49 a 64	56	65 a 80	57	81 a 96
Palavra	Comandos															
52	1 a 16															
53	17 a 32															
54	33 a 48															
55	49 a 64															
56	65 a 80															
57	81 a 96															

Palavra	Nome	Descrição														
58-63	Bits de Comando de Erro	Estes registros contêm flags de Bit Erro para cada comando na Lista de Comando até os primeiros 96 comandos. Os Bits de Erro são mapeados nas palavras dependendo de sua posição relativa na Lista de Comando. O mapeamento dentro dos Bits Realizado é o seguinte:														
		<table border="1"> <thead> <tr> <th>Palavra</th> <th>Comandos</th> </tr> </thead> <tbody> <tr> <td>58</td> <td>1 a 16</td> </tr> <tr> <td>59</td> <td>17 a 32</td> </tr> <tr> <td>60</td> <td>33 a 48</td> </tr> <tr> <td>61</td> <td>49 a 64</td> </tr> <tr> <td>62</td> <td>65 a 80</td> </tr> <tr> <td>63</td> <td>81 a 96</td> </tr> </tbody> </table>	Palavra	Comandos	58	1 a 16	59	17 a 32	60	33 a 48	61	49 a 64	62	65 a 80	63	81 a 96
Palavra	Comandos															
58	1 a 16															
59	17 a 32															
60	33 a 48															
61	49 a 64															
62	65 a 80															
63	81 a 96															
		Exemplo: Palavra 52 bit 0 é o comando #1														

### 5.3.2 Lógica Ladder

Uma Linha única pode ser dar entrada para mover os bits "Realizado" e "Erro" do buffer BTR para a tabela de dados PLC/SLC. Ver exemplo abaixo:

####



Copy the Command Done and Error bits from the BTR buffer to the data table.

## 6 Configuração de Comando Modbus

Os drivers de comunicação Mestre e Escravo de Modbus MCM da ProSoft\_Technology suportam comandos de leitura e escrita. Quando se configura uma porta Mestre, a decisão com respeito a qual comando usar é feita dependendo do tipo de dados endereçados, e do nível de suporte Modbus no equipamento escravo. Quando se configura como escravo, é importante entender como os comandos Modbus funcionam para determinar como estruturar os dados de aplicação.

Incluimos um excerto a partir da Especificação de Protocolo Modbus no Apêndice para ajudar a entender plenamente a funcionalidade do protocolo Modbus.

### 6.1 Comandos Modbus

O módulo MCM suporta um subconjunto de comandos da especificação Modbus que consiste primariamente dos Códigos de Função requeridos para ler e escrever dados. As seções abaixo detalham os diferentes comandos suportados pelo módulo. A perspectiva é dada principalmente a partir daquela de uma porta Mestre, mas a discussão vai ajudar a implementar as portas escravo.

Cod. Função	Comando	Gama de endereços	Comentários Driver Escravo	Comentários Driver Mestre
1	Read Output Status	Coil 0001 a 9999	Módulo retorna dados binários a partir do espaço do registro de "Status de Saída". O módulo suporta até 125 palavras de dados em um comando.	<u>Endereço Fonte:</u> Iniciando endereço de bit no escravo a partir do qual o dado deve ser lido. Entrar com 0 no coil de endereço 0001.  <u>Quantidade:</u> Número de bits a serem lidos (até 125 palavras de comprimento).  <u>Endereço de Destino:</u> Iniciando endereço de palavra na Memória de Registro do Módulo no qual o dado deve ser colocado, iniciando com bit 0.
2	Read Input Status	Bit 10001 a 29999	Módulo retorna dados binários a partir do espaço do registro de "Status de Saída". O módulo suporta até 125 palavras de dados em um comando.	<u>Endereço Fonte:</u> Iniciando endereço de bit no escravo a partir do qual o dado deve ser lido. Entrar com um 0 no coil de endereço 0001.  <u>Quantidade:</u> Número de bits a serem lidos (até 125 palavras de comprimento).  <u>Endereço de Destino:</u> Iniciando endereço de palavra na Memória de Registro do Módulo no qual o dado deve ser colocado, iniciando com bit 0.

Cod.de Função	Comando	Gama de endereços	Comentários Driver Escravo	Comentários Driver Mestre
3	Read Multiple Registers	Registros 40001 a 47999	Módulo retorna dados de palavra a partir do espaço do registro. Todas 4000 palavras no módulo compõem o espaço de registro endereçável por um host. A palavra 0 no módulo corresponde ao Endereço de Modbus 40001. O módulo suporta até 125 palavras de dados em um comando.	<u>Endereço Fonte:</u> Iniciar endereço de bit no escravo a partir do qual dado deve ser lido. Entrar 0 para endereçar 40001 no escravo <u>Quantidade:</u> Número de palavras ou valores a serem lidos (até 125 palavras de comprimento). <u>Endereço de Destino:</u> Iniciar endereço de palavra na Memória de Registro do Módulo no qual o dado deve ser colocado. <u>Tipo:</u> Controlar troca de byte e palavra para leitura de ponto flutuante (ver seção 4 para detalhes)
4	Read Input Registers	Registros 30001 a 39999	Módulo retorna dados a partir do espaço de Registro de entrada no módulo. O módulo suporta até 125 palavras de dados em um comando.	<u>Endereço Fonte:</u> Iniciar endereço de bit no escravo a partir do qual dado deve ser lido. Entrar 0 para endereçar 30001 no escravo <u>Quantidade:</u> Número de palavras ou valores a serem lidos (até 125 palavras de comprimento). <u>Endereço de Destino:</u> Iniciar endereço de palavra na Memória de Registro do Módulo no qual o dado deve ser colocado.
5	Single Bit/Coil Write	Coil 0001 a...	<u>Modo Normal:</u> O bit escrito será colocado no espaço de dados de módulo <u>Modo Pass Through</u> O bit escrito passa para PLC/SLC para manuseio na Lógica Ladder	<u>Endereço Fonte</u> Iniciar no MCM que deve ser usado para determinar ação set/ reset de bit do comando. <u>Quantidade:</u> Não é usada, default de um <u>Endereço de destino</u> O endereço de bit no escravo que deve ser set/ reset. Entrar endereço 0 para coil de endereço 0001 no escravo.
6	Single Register Write	Registros 40001 a 47999	<u>Modo Normal:</u> O bit escrito será colocado no espaço de dados de módulo <u>Modo Pass Through</u> O bit escrito passa para PLC/SLC para manuseio na Lógica Ladder	<u>Endereço Fonte</u> Iniciar endereço de registro no MCM que deve ser usado para determinar a fonte do dado a ser escrito. <u>Quantidade:</u> Não é usada, default de um <u>Endereço de destino</u> O endereço de bit no escravo que deve ser set/ reset. Entrar endereço 0 para registro de endereço 4001 no escravo

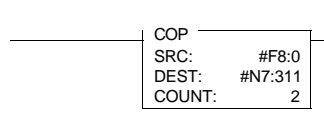
15	Multiple Bit/Coil Write		<p><u>Modo Normal:</u> O bit escrito será colocado no espaço de dados de módulo. Até 125 dados de bit máximo</p> <p><u>Modo Pass Through</u> O bit escrito passa para PLC/SLC para manuseio na Lógica Ladder. Até 30 palavras de dados de bit máximo</p>	<p><u>Endereço Fonte</u> Iniciar endereço de registro no MCM que deve ser usado para determinar a fonte do dado a ser escrito.</p> <p><u>Quantidade:</u> O número de bits a serem escritos (até 125 palavras no total)</p> <p><u>Endereço de destino</u> O endereço de bit no escravo que deve ser set/reset. Entrar endereço 0 para coil de endereço 0001 no escravo</p>
16	Multiple Register Write		<p><u>Modo Normal:</u> O bit escrito será colocado no espaço de dados de módulo. Até 125 dados de bit máximo</p> <p><u>Modo Pass Through</u> O bit escrito passa para PLC/SLC para manuseio na Lógica Ladder. Até 30 palavras de dados de bit máximo</p>	<p><u>Endereço Fonte</u> Iniciar endereço de registro no MCM que deve ser usado para determinar a fonte do dado a ser escrito.</p> <p><u>Quantidade:</u> O número de bits a serem escritos (até 125 palavras no total)</p> <p><u>Endereço de destino</u> O endereço de bit no escravo que deve ser set/reset. Entrar endereço 0 para coil de endereço 0001 no escravo</p>

## 6.2 Suporte de Ponto Flutuante

O movimento do dado de ponto flutuante entre módulo MCM e outros dispositivos é facilmente realizado desde que o dispositivo suporte o formato de Ponto Flutuante IEEE 754. Este formato é um formato de ponto flutuante com precisão de 32 bit único.

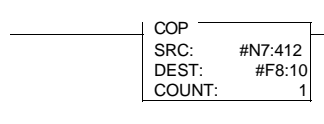
A programação necessária para mover o dado de ponto flutuante é aproveitar o comando COP no PLC/SLC. O comando COP é único nos comandos de movimento de dado PLC/SLC pelo fato de ser uma função que independe do tipo, ou seja nenhuma conversão de dado é feita quando se move dado entre tipos de arquivo, (i.e., é cópia de imagem e não cópia de valor)

A estrutura do comando COP para mover dado de um arquivo de ponto flutuante para um arquivo inteiro (algumas vezes poderia mover os valores de ponto flutuante para o módulo) é a seguinte:



Este comando move um valor de ponto flutuante em duas imagens de 16 bit inteiras para o arquivo inteiro. Para valores de ponto flutuante múltiplos simplesmente aumentar campo de quantidade de um fator de 2 por valor de ponto flutuante.

A estrutura do comando COP para mover dados de um arquivo inteiro para um arquivo de ponto flutuante (algumas vezes é preciso receber valores de ponto flutuante do módulo) é a seguinte:



Este comando move dois registros inteiros de 16 bit contendo uma imagem de valor de ponto flutuante para o arquivo de ponto flutuante. Para múltiplos valores simplesmente aumente o campo de quantidade

#### 6.2.1 Suporte de Ponto Flutuante ENRON

Muitos fabricantes implementaram um suporte especial em seus drivers para suportar o que normalmente é chamado versão ENRON do protocolo Modbus.. Nesta implementação, endereços de registro >7000 são presumidos valores de ponto campo de flutuante. Isto significa que o "campo de quantidade agora se torna "número de valores". No formato de ponto flutuante, cada valor representa duas palavras. Isto foi implementado no MCM da seguinte maneira.

Cod de Função	Driver Mestre	Driver Escravo
3- Register Read	Se o endereço de destino for = 7000, então espere pedido de 2 palavras por Quantidade.	Se o registro endereçado pelo host for = 700 então a seguinte equação será usada para determinar o endereço de registro MCM a ser lido a partir do endereço: MCM= float_offset + (fim de reg - 7000)*2
4- Register Write	Se o endereço de destino for = 7000, então envie valor de duas palavras por Quantidade (i.e., Valor de Quantidade se torna número de valores e não número de palavras)	Se o registro endereçado pelo host é =700, então a seguinte equação será usada para determinar endereço de registro MCM a ser escrito em: End. MCM= float_offset + (fim de reg - 7000)*2

## 7 Diagnóstico e Solução de Problemas

Diversas capacidades de diagnóstico de hardware foram implementadas usando luzes LED na frente do módulo. As seções a seguir explicam o significado de LEDs para ambas plataformas PLC e SLC.

### 7.1 Indicadores LED de Plataforma PLC 3100

O produto MCM de plataforma PLC se baseia na plataforma e hardware CIM ProSoft. A tabela a seguir documenta os LEDs no hardware 3100-MCM e explica operação dos LEDs.

Placa ProSoft CIM			
ATIVA	0	0	FLT
CFG	0	0	BPLN
ERR1	0	0	ERR2
TXD1	0	0	TX2
RXD1	0	0	RXD2

ProSoft CIM	Cor	Status	Indicação
ACT		Piscante (rápida)	Estado normal O módulo está operando normalmente e Transferindo Blocos com sucesso
		On	O módulo recebe energia pelo barramento, mas deve haver um outro problema
		Off	O módulo está tentando transferir dados com PLC e não consegue. O PLC pode estar no modo PGM ou com problema
FLT	Vermelho	Off	Estado normal Nenhum problema de sistema foi detectado durante diagnóstico geral
		On	Um problema de sistema foi detectado durante diagnóstico geral. Favor contatar fábrica para suporte técnico
CFG	Verde	Off	Estado normal Não está ocorrendo nenhuma atividade relacionada à configuração neste momento
		Piscante	Esta luz pisca cada vez que um Bloco de Configuração de Módulo (ID=255) é recebido da Lógica Ladder de processador.
		On	A luz permanece acesa sempre que for detectado um erro de configuração. O erro pode estar na Configuração de Porta ou nos dados de Configuração de Sistema. Ver Seção 4 para detalhes.
BPLN	Vermelho	Off	Estado normal Quando esta luz está desligada e a luz ACT pisca rapidamente, o módulo está transferindo dados por Transferência de Bloco com PLC
		On	Indica que a transferência de Bloco entre PLC e módulo falhou (Não foi ativado no release inicial do produto)
ERR1 ERR2	Ambar	Off	Estado normal Quando o LED de erro está desligado e a correspondente porta está transferindo dados, não há nenhum erro de comunicação.
		Piscante	Erros de comunicação periódica estão ocorrendo durante comunicação de dados. Ver seção 4 para determinar condição de erro.
		On	Este LED permanece ligado em diversas condições <ul style="list-style-type: none"> <li>- Entrada CTS não está sendo atendida</li> <li>- Erro de Configuração de Porta</li> <li>- Erro de Configuração de Sistema</li> <li>- Comunicação mal sucedida no escravo MCM</li> <li>- Condição de erro recorrente no Mestre MCM</li> </ul>
TX1 TX2	Verde	Piscante	A porta está transmitindo dados
Rx1 Rx2	Verde	Piscante	A porta está recebendo dados

### 7.2 Indicadores LED de Plataforma 3150 SLC

A tabela a seguir documenta os LEDs no hardware 3150-MCM e explica a operação dos LED

#### COMUNICAÇÕES

?	ACT	?	FALHA
?	CFG	?	BPLN
?	PRT1	?	ERR1
?	PRT2	?	ERR2

Nome do LED	Cor	Status	Indicação
ACT	Verde	Piscante (Rápido)	<u>Estado normal</u> O módulo está operando normalmente, e transferindo Blocos com sucesso com SLC
		On	O módulo está recebendo energia do barramento, mas deve haver um outro problema.
		Off	O módulo está tentando Transferir Bloco com SLC sem sucesso. O SLC pode estar no modo PLC ou pode haver um outro problema.
FLT	Vermelho	Off	<u>Estado Normal</u> Nenhum problema de sistema foi detectado durante diagnóstico geral
		On	Um problema de sistema foi detectado durante diagnóstico geral. Favor contatar fábrica para suporte técnico.
CFG	Verde	Off	<u>Estado Normal</u> Não está ocorrendo nenhuma atividade relacionada à configuração neste momento.
		Piscante	A luz pisca cada vez que é recebido um bloco de Configuração de módulo (ID=255) a partir da Lógica Ladder do processador.
		On	A luz permanece acesa sempre que for detectado um erro de configuração. O erro pode estar nos dados de Configuração de Porta ou nos dados de Configuração de Sistema. Ver Seção 4 para detalhes.
BPLN	Vermelho	Off	<u>Estado Normal</u> Quando esta luz está desligada e a luz ACT piscando rapidamente, o módulo está ativamente transferindo dados por Transferência de Bloco com SLC
		On	Indica que falhou a Transferências de Bloco entre SLC e módulo.
Err1 Err2	Ambar	Off	<u>Estado Normal</u> Quando o LED de erro está desligado e a correspondente porta está transferindo dados ativamente, não há nenhum erro de comunicação
		Piscante	Erros periódicos de comunicação ocorrem durante comunicação de dados. Ver seção 4 para determinar a condição de erro.
		On	Este LED permanece ligado em diversas condições <ul style="list-style-type: none"> <li>- Entrada CTS não está sendo atendida</li> <li>- Erro de Configuração de Porta</li> <li>- Erro de Configuração de Sistema</li> <li>- Comunicação mal sucedida no escravo MCM</li> <li>- Condição de erro recorrente no Mestre MCM</li> </ul>
PRT1 PRT2	Verde	Piscante	A porta se encontra em comunicação, quer transmitindo ou recebendo dados

### 7.3 Solução de Problemas- Geral

Para ajudá-lo na solução de problemas do módulo, foram providas tabelas a seguir. Mas se tiver outras questões ou problemas, não hesite em contatar-nos.

As entradas nesta seção foram colocadas na ordem em que os problemas mais provavelmente ocorrem depois de energizar o módulo.

Descrição do Problema	Etapas a serem seguidas
Luz BLPN acesa (SLC)	<p>A luz BLPN acende quando o módulo não pensa que SLC está no modo Run (i.e., SLC em PGM ou falhou). Se SLC está executando, então verifique:</p> <ul style="list-style-type: none"> <li>- Arquivo de status SLC para garantir que o slot se encontra habilitado</li> <li>- O Bits de Transferência Habilitado/Realizado (Bits E/S 0 para o slot com o módulo) deve ser controlado pela Lógica Ladder. Ver Seção 2.x para detalhes ou exemplo de Lógica Ladder no Apêndice</li> <li>- Se a Lógica Ladder para o módulo estiver em um arquivo de sub-rotina, verifique se há um comando JSR chamando SBR</li> </ul>
Luz CFG não acende depois de energizada (sem LED ERR)	<p>O número ID do Bloco BTW 255 não está sendo detectado pelo módulo. Isto pode ser devido a uma falha de Transferência de Bloco (PLC) ou a um erro na Lógica Ladder, impedindo de o valor 255 ser movido para o buffer BTW.</p>
Luz CFG não acende depois de energizada (com LED ERR)	<p>Se a luz BLPN acendeu, então muitos dos valores de Porta e Configuração de Sistema são valores verificados pelo módulo garantindo que entradas legais entraram na tabela de dados. Verifique na Tabela de Status de Erro indicação de erro de configuração.</p>
Luz CFG muda	<p>Em condições normais, o LED CFG deve apagar imediatamente depois da recepção. Se a luz CFG muda, isto usualmente indica que a condição lógica que coloca o valor de ID do Bloco 255 no buffer BTW não está sendo liberada. Verifique a Lógica Ladder para garantir se a condição de mover o valor 255 não está correta.</p>
Módulo não está transmitindo	<p>Presumindo que o processadores se encontra em Run, verifique:</p> <ul style="list-style-type: none"> <li>- Entrada CTS não está sendo atendida (verifique jumper RTS/CTS)</li> <li>- Verifique nos códigos de Status de Erro cod. 255.</li> <li>- No modo Escravo, verifique o endereço de escravo que está sendo pedido a partir do Host.</li> <li>- No modo Mestre, verifique a configuração da Lista de Comando e se a Lista de Comando está sendo movida para o módulo (i.e. verifique a Quantidade de Bloco de Comando e a Lógica Ladder associada).</li> </ul>
Código de Erro 255 na Tabela de Status	<p>Isto é causado somente por uma entrada CTS perdida na porta. Se um cabo está conectado na porta, então verifique se há um jumper instalado entre pinos RTS e CTS. Se houver, então pode ser um problema de hardware</p>
Sobrescrevendo blocos de dados	<p>Esta condição normalmente ocorre quando se esquece que o valor de ID do Bloco BTW está sendo manipulado pelo módulo, e que sempre começa em 0. Favor verificar se a configuração do módulo - Quantidade de Blocos de Leitura e Escrita - não está fazendo os dados do PLC/SLC sobrescreverem dados que retornam do módulo. Um método simples de verificação é realizar um histograma no registro de ID do Bloco BTW.</p>
Ocorrendo Troca de Dados	<p>Sob diversas circunstâncias ocorre troca de dados no módulo. Esta troca sempre é associada ao jumper de porta 8/16 na traseira da placa. Favor verificar, se o jumper se encontra na 8ª posição.</p>
Novos valores de configuração não são aceitos pelo módulo.	<p>Para novos valores serem movidos para o módulo, uma Escrita de Transferência de Bloco com ID do Bloco de 255 deve ser transmitido para o módulo. O "Bit Config de Usuário" na lógica de exemplo causa isto. Na lógica de exemplo, o bit deve ser aplicado à tabela de dados manualmente, ou o módulo deve ser desligado e reset.</p> <p>Para download da configuração, quando se passa de PGM para RUN, simplesmente adicione Run para estabelecer o "Bit Config de Usuário" com base no primeiro Bit de Status de Escaneamento (S1:1/15).</p>

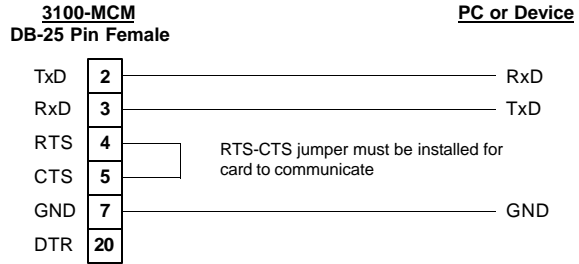
Descrição do Problema	Etapas a serem seguidas
Códigos de Erro retornam para localizações sem comandos (Configuração Mestre)	<p>Esteja certo que o valor de Quantidade de Bloco de Comando está ajustado corretamente. Deve haver uma ramificação ou lógica na Linha de escrita que corresponda a cada Bloco de Comando a ser escrito (i.e., Quantidade de Bloco de Comando de 2 deve ter 2 ramificações de lógica para trabalhar) IDs do Bloco BTW 80 e 81.</p> <p>Se o valor de configuração de Quantidade de Bloco de Configuração exceder o número de ramificações na lógica, a Lista de Comando será inadvertidamente duplicada. Para resolver este problema, adicione mais ramificações de lógica ou reduza o valor de Quantidade de Bloco de Comando para corresponder ao número de ramificações de lógica BTW.</p>
RX1 ou RX2 estão contínuos (somente 3100)	Os LEDs TX e RX no módulo são ligados ao estado de hardware das portas (i.e., não são controlados diretamente por firmware). Quando o LED RX está contínuo isto normalmente indica que a polaridade da conexão de cabo à porta está trocada.

## 8 Diagramas de Conexão de Cabo

Os diagramas a seguir mostram os requisitos de conexão para as portas nos módulos 3100/3150.

Módulo 3150-MCM

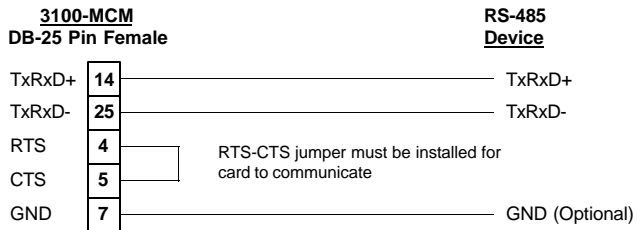
**R\_S232 sem Handshaking de Hardware**  
 Conexão de Porta com outra porta de comunicação



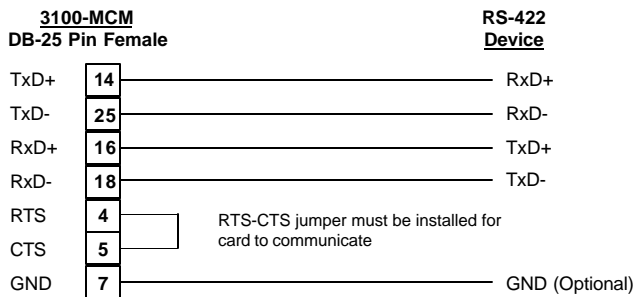
**R\_S232 com Handshaking de Hardware**  
 Conexão de Porta com modem ou outro dispositivo similar



**Conexão de Fio RS\_485/2**  
 O jumper no módulo deve ser ajustado na posição RS\_485 para todas 2 aplicações de fio



**Conexão de Fio RS\_422/4**  
 O jumper no módulo deve estar ajustado na posição RS\_422 para todas aplicações de 4 fios



Módulo 3150-MCM

<p><b>RS232 sem Handshaking de Hardware</b> Conexão de Porta com outra porta de comunicação</p>	<p><b>3150-MCM DB-9 Pin Male</b> <span style="float: right;"><b>PC or Device</b></span></p>
<p><b>RS232 com Handshaking de Hardware</b> Conexão de Porta com modem ou outro dispositivo similar</p>	<p><b>3150-MCM DB-9 Pin Male</b> <span style="float: right;"><b>Modem or other Comm Device</b></span></p>
<p><b>Conexão de Fio RS_485/2</b> O jumper no módulo deve ser ajustado na posição RS_485 para todas 2 aplicações de fio</p>	<p><b>3150-MCM DB-9 Pin Male</b> <span style="float: right;"><b>RS-485 Device</b></span></p>
<p><b>Conexão de Fio RS_422/4</b> O jumper no módulo deve estar ajustado na posição RS_422 para todas aplicações de 4 fios</p>	<p><b>3150-MCM DB-9 Pin Male</b> <span style="float: right;"><b>RS-422 Device</b></span></p>

RS\_485 e RS\_422  
Se a comunicação no modo RS\_422/RS\_485 não funcionar, a despeito de todas tentativas, tente trocar polaridade de terminação. Alguns fabricantes interpretam polaridades A/B e +/- diferentemente.

## **A Suporte, Manutenção, e Garantia**

### **Suporte técnico**

ProSoft\_Technology se distingue por sua capacidade de prover suporte integral a seus clientes. Qualquer questão ou problema que surgir, por favor nos contate:

### **Fábrica/ Suporte Técnico**

ProSoft Technology  
 9801 Camino Media, Suite 105  
 Bakersfield CA 93311  
 (661)664-7208  
 (800)326-7066  
 (661)664-7233 (fax)  
 prosoft@prosoft-technology.com  
 www.prosoft-technology.com

Antes de ligar para o suporte, se prepare. Para prover o melhor e mais rápido suporte possível, pediremos as seguintes informações (que poderão ser enviados por fax antes da consulta por telefone propriamente dita).

- 1 Número da Versão do Produto
- 2 Informação de Configuração
  - Configuração de Comunicação
  - Lista de Comando Mestre
  - Posições de Jumper
- 3 Hierarquia do Sistema
- 4 Informação de conexão física
  - RS\_232, RS\_422 ou RS\_485
  - Configuração de cabo
- 5 Operação do Módulo
  - Operação de Transferência de Bloco
  - Padrões de LED

Um sistema automático depois do horário (no número de Bakersfield) dá acesso por pager a um de nossos técnicos e/ou engenheiros de suporte a qualquer tempo para responder questões urgentes.

### **Manutenção e Reparo de Módulo**

A placa MCM é um produto eletrônico, projetada e fabricada para funcionar em condições adversas. Mas como qualquer produto, por envelhecimento, má aplicação, ou outro motivo qualquer, pode precisar de reparo em algum momento.

Quando comprado da ProSoft\_Technology, o módulo tem garantia de um ano para peças e mão de obra de acordo com os limites especificados na garantia. Substituição e/ou retorno devem ser dirigidos para o distribuidor de quem o produto foi comprado. Se a placa precisar retornar para reparo, será preciso antes obter o número RMA da ProSoft\_Technology. Por favor, contate a fábrica para obtê-lo, e uma vez disponível o coloque de modo bem visível na embalagem.

### **Política de Garantia Geral**

A ProSoft\_Technology Inc (doravante simplesmente ProSoft) garante que o Produto se encontra em conformidade e apresenta o desempenho de acordo com especificações técnicas incluídas no material anexo, e não deve apresentar defeitos de material e/ou mão de obra, no período indicado, sendo que este período de garantia começa a contar na data de recebimento do produto.

Esta garantia é limitada a reparos e substituição, a critério da ProSoft, de produtos com defeitos ou não-conformes, e a ProSoft não será responsável por falhas do produto em realizar funções especificadas ou outras não-conformidades causadas ou atribuída a: (a) qualquer mau uso do Produto; (b) falha do Cliente em cumprir qualquer especificação ou instrução da ProSoft; (c) negligência, abuso, ou acidente infligido ao Produto; ou (d) qualquer equipamento ou software associado ou complementar não fornecido pela ProSoft.

Serviço de garantia limitado pode ser obtido entregando o Produto à ProSoft com respectiva prova de compra ou recibo. O cliente concorda em garantir o produto ou assumir riscos de perda ou dano em trânsito e pagar adiantado taxas de embarque e transporte à ProSoft e usar embalagem original ou equivalente. Contate o Serviço ao Cliente da ProSoft para maiores informações.

#### **Limite de Responsabilidade**

EXCETO COMO EXPRESSAMENTE ESPECIFICADO AQUI, A **PROSOFT** NÃO DÁ GARANTIA DE QUALQUER TIPO, EXPRESSO OU SUBTENDIDO, COM RESPEITO A QUALQUER EQUIPAMENTO, PARTE, OU SERVIÇO PROVIDO DE ACORDO COM ESTE TERMO, INCLUINDO MAS NÃO SE LIMITANDO À GARANTIA DE ADEQUAÇÃO E COMERCIAL PARA UM PARTICULAR PROPÓSITO, NEM A **PROSOFT** E SEU REPRESENTANTE SERÃO RESPONSÁVEIS POR QUALQUER DANO, INCLUINDO, MAS NÃO SE LIMITANDO A, DANOS DIRETOS, INDIRETOS, ACIDENTAIS, ESPECIAIS, OU PROVOCADOS, QUER POR CONTRATO OU POR CULPA INCLUINDO NEGLIGÊNCIA E RESPONSABILIDADE ESTRITA TAL COMO, MAS NÃO SE LIMITANDO A, PERDA, LUCROS CESSANTES OU BENEFÍCIOS RESULTANTES, PROPICIADAS POR, OU EM CONEXÃO COM, USO OU FORNECIMENTO DO EQUIPAMENTO, PARTE, OU SERVIÇO DESCRITO AQUI OU DESEMPENHO, USO, INABILIDADE NO USO DO MESMO, MESMO SE A RESPONSABILIDADE TOTAL DA **PROSOFT** OU SEU REPRESENTANTE EXCEDER O PREÇO PAGO PELO PRODUTO.

Onde estabelecido por Lei Estadual, algumas das exclusões ou limitações acima poderão não ser aplicáveis em alguns Estados. Esta garantia provê direitos legais específicos; outros direitos que variem de Estado para Estado ainda poderão existir. Esta garantia não será aplicada na extensão em que qualquer provisão desta garantia seja proibida por qualquer Lei Federal, Estadual, ou Municipal que não possa ser pressuposta.

#### **Detalhes de Garantia para o Hardware**

##### Período de Garantia:

A ProSoft garante hardware pelo período de 1 (um) ano.

##### Procedimento de Garantia

No retorno do Produto de Hardware a ProSoft deve, a seu critério, reparar ou substituir o Produto sem nenhum custo adicional, exceto como estabelecido acima. O reparo de Partes ou substituição de todo o Produto será provido com base de troca por Partes ou Produtos novos ou reconicionados. Produtos e Partes substituídos se tornam propriedade da ProSoft. Se a ProSoft determinar que o Produto não tem garantia, a ProSoft, a critério do Cliente, pode reparar o Produto usando custos correntes para Partes e Mão de Obra, retornando o produto para coleta do transportador.

## B Especificações de Produto

A família de produto (Módulo de Comunicação Modbus) 3100/3150 MCM permite a Allen\_Bradley 1771 e processadores compatíveis 1746 E/S facilmente interfacearem dispositivos compatíveis de protocolo Modbus tal como Mestre Modbus e Escravo Modbus.

O produto MCM inclui as seguintes características padrão:

### Especificação Geral

Duas portas seriais totalmente configuráveis, cada uma delas capaz de suportar Mestre Modbus e Escravo Modbus.

As Configurações disponíveis incluem:

<u>Configurações Disponíveis</u>	<u>Porta 1</u>	<u>Porta 2</u>
Mestre-Mestre	Mestre	Mestre
Mestre-Escravo	Mestre	Escravo
Escravo-Escravo	Escravo	Escravo

- Suporte para armazenamento e transferência de até 4000 registros para tabelas de dados PLC/SLC.
- Suporte de movimento para tipos de dado binário, inteiro, ASCII e ponto flutuante.
- Mapeamento de Memória é completamente definido pelo usuário através de configuração da tabela de dados.
- Handshaking RS\_232C para aplicações radio/modem SCADA.
- Compatível com RS\_232/ RS\_485 para aplicações multidrop com até 32 escravos por porta.
- Suporte de Satélite e Pacote de Rádio com Timeout Inter\_caracter configurável disponível por porta
- Configuração de software (a partir da Lógica Ladder do processador

Endereço de Escravo	:	1 a 247 (o é broadcast)
Paridade	:	Nenhum, ímpar, ou par
Stop Bit	:	1 ou 2
Baud Rate	:	300 a 38400
RTS a TxD	:	0 a 65535 ms, 1 ms de resolução
RTS Off	:	0 a 65535 ms, 1 ms de resolução
Timeout	:	0 a 65535 ms, 1 ms de resolução

- Tempo de resposta

Os drivers de protocolo Mestre e Escravo Modbus são escritos em Assembly e em linguagem compilada de alto nível. Assim, as capacidades de interrupção do hardware são totalmente utilizadas para minimizar atrasos, e otimizar o desempenho do produto.

### Especificações de Escravo Modbus

Modos de protocolo

- modo RTU com verificação de erro CRC-16
- modo ASCII com verificação de erro LRC (formato de 7 e 8 bits)

Códigos de Função Modbus suportados

1	Read Output Status
2	Read Input Status
3	Read Multiple Data Registers
4	Read Input Registers
5	Force Single Coil (Latch/Unlatch)
6	Preset (Write) Single Data Register
8	Loopback Test (Test 0 only)
15	Multiple Coil Latch/Unlatch
16	Preset (Write) Multiple Data Register

- Suporta comando de broadcast a partir do host
- Status de Erro e Estatística de Comunicação retornam para o processador Ladder
- Modo Pass\_Through, seleção configurável
  - Seleção para transferir comandos de escrita a partir do host diretamente para o Ladder para processamento. Permite aceitação condicional de dados de escrita por Lógica Ladder.
- Modo Routing de Comando
  - Suporta routing de Escravo para Mestre até seis endereços de escravo. Permite a um supervisor na porta escravo acessar dados dos escravos roteados.

**Especificações de Mestre Modbus**

- Modos de protocolo
  - Modo RTU com verificação de erro CRC-16
  - Modo ASCII com verificação de erro LRC (formato de 7 e 8 bit)
- Códigos de Função Modbus suportados

1	Read Output Status
2	Read Input Status
3	Read Multiple Data Registers
4	Read Input Registers
5	Force Single Coil (Latch/Unlatch)
6	Preset (Write) Single Data Register
8	Loopback Test (Test 0 only)
15	Multiple Coil Latch/Unlatch
16	Preset (Write) Multiple Data Register

- Suporta até 100 entradas da Lista de Comando, cada uma delas individualmente configurável com os seguintes parâmetros.

Port/Mode Section  
 Slave Address  
 Function Code  
 Source/Destination data address  
 Number of Values to transfer  
 Polling Time

- Modo de Controle de Comando
  - Permite controle de execução individual feito em Lógica Ladder habilitando que uma Lista de Comando seja executada com base em eventos no PLC/SLC.
- Bits individuais de comando "Realizado" e "Erro" disponíveis
- Suporta Escritos dirigidos por Evento iniciados diretamente da Lógica Ladder
- Códigos de Status de Erro de Comando Individual retornam para o processador Ladder
- Suporta comandos broadcast para escravos.

**Especificações de Hardware**

Corrente do Barramento

3100 : 0.65 A  
 3150 : 0.15 A/ 5 V  
       : 0.04 A/ 24 V

- Temperatura de operação: 0 a 60°C
- Temperatura de Armazenamento: -40° a 85°C
- Conexões

3100 : 2 -Conectores Fêmea DB25  
 3150 : 2 -Conectores Macho DB9

## C Especificação de Protocolo Modbus

### Leia Status de Saída (Código de Função 01)

#### Consulta

Esta função permite ao usuário obter status On/Off de coils lógicos usados para controlar saídas discretas somente a partir do escravo endereçado. O modo broadcast não é suportado com esta função. Em adição ao endereço de escravo e aos campos de função, a mensagem requer que o campo de informação contenha o endereço de coil inicial a ser lido (Endereço de Início) e número de localizações que será interrogado para o obter o status de dados.

O endereçamento permite que até 2000 coils sejam obtidos em cada pedido, no entanto, o dispositivo escravo específico pode ter restrições que baixem a quantidade máxima. Os coils são numerados a partir de zero (Nº de coil 1= zero, Nº de coil 2= um, Nº de coil 3= dois, etc.)

A figura C1 é um pedido de read output status de amostra para coils de leitura 0020 a 0056 a partir do número de dispositivo de escravo 11.

ADR	FUNC	DATA START PT HO	DATA START PT LO	DATA # OF PTS HO	DATA # OF PTS LO	ERROR CHECK FIELD
11	01	00	13	00	25	CRC

Figura C1, Read Output Status Query Message

#### Resposta

Uma resposta de exemplo ao read output status está mostrada na figura C2. O dado é agrupado um bit para cada coil. A resposta inclui o endereço de escravo, código de função, quantidade de caracteres de dado e verificação de erro. Dado será agrupado com um bit para cada coil (1= ON e 0 = OFF). O bit de ordem baixa do primeiro caracter contém o coil endereçado, seguindo o restante. Para quantidades de coil que agora não são pares de oito, o último caracter será preenchido de zeros na extremidade de ordem alta. A quantidade de caracteres de dados será sempre especificada como quantidade de caracteres RTU, i.e. o número será o mesmo quer para RTU ou ASCII.

Uma vez que o dispositivo escravo de interface é atendido ao fim de um escaneamento do controlador, o dado refletirá o status de coil ao fim do escaneamento. Alguns escravos limitam a quantidade de coils provida a cada escaneamento, assim para grandes quantidades de coil, múltiplas transações PC deverão ser feitas usando status de coil a partir de um escaneamento seqüencial.

ADR	FUNC	BYTE COUNT	DATA COIL STATUS 20-27	DATA COIL STATUS 28-35	DATA COIL STATUS 36-43	DATA COIL STATUS 44-51	DATA COIL STATUS 52-56	ERROR CHECK FIELD
11	01	05	CD	6B	B2	0E	1B	CRC

Figura C2, Read Output Status Response Message

O status dos coils 20-27 está mostrado como CD (HEX)=1100 1101 (Binário). Lendo da esquerda para a direita, isto mostra que coils 27, 26, 23, 22 e 20 estão todos ON. Os outros bytes de dados de coils são decodificados do mesmo modo. Devido à quantidade de status de coils pedida, o último campo de dados - 1B(HEX)=0001 1011 (Binário) - contém o status de apenas 5 coils (52 a 56) ao invés de 8 coils. Os 3 bits mais a esquerda são providos como zeros para encher o formato de 8-bit.

### Read Input Status (Código de Função 02)

#### Consulta

Esta função permite ao usuário obter status ON/OFF de entradas discretas no escravo endereçado. O modo Broadcast de PC escravo não é suportado com este código de função. Em adição ao endereço de escravo e campos de função, a mensagem requer que o campo de informação contenha o endereço de entrada a ser lido (Endereço de Início) e o número de localizações que será interrogado para obter o dado de status inicial.

O endereçamento permite que até 2000 entradas sejam obtidas por pedido; no entanto, o dispositivo escravo específico pode ter restrições que baixem a quantidade máxima. As entradas são numeradas a partir de zero (entrada 10001= zero, entrada 10002= um, entrada 10003= dois etc., para 584).

A figura C3 é um pedido de read input status de amostra para ler entradas 10197 - 10218 a partir do número de escravo 11.

ADR	FUNC	DATA START PT HO	DATA START PT LO	DATA #OF PTS HO	DATA #OF PTS LO	ERROR CHECK FIELD
11	02	00	C4	00	16	CRC

Figura C3, Read Input Status Query Message

#### Resposta

Uma resposta de exemplo a read input status está mostrada na figura C4. O dado é agrupado um bit para cada entrada. A resposta inclui endereço de escravo, quantidade de caracteres de dados, verificação de erro. O dado será agrupado com um bit para cada entrada (1= ON, 0= OFF). O bit de ordem mais baixa do primeiro caractere contém entrada endereçada, seguindo o resto. Para quantidades de entrada que não são múltiplos de oito, o último caractere será preenchido com zeros na extremidade de ordem alta. A quantidade de caracteres de dados é sempre especificada como quantidade de caracteres RTU, i.e., o número é o mesmo para RTU ou ASCII.

Uma vez que o dispositivo de interface escravo é atendido no fim do escaneamento de controle, o dado refletirá o status de entrada no fim do escaneamento. Alguns escravos limitam a quantidade de entradas de cada escaneamento, assim, para grandes quantidades de coils, múltiplas transações PC deverão ser feitas usando status de coil para escaneamentos seqüenciais.

ADR	FUNC	BYTE COUNT	DATA DISCRETE INPUT 10197-10204	DATA DISCRETE INPUT 10205-10212	DATA DISCRETE INPUT 10213-10218	ERROR CHECK FIELD
11	02	03	AC	DB	35	CRC

Figura C4, Read Input Status Response Message

O status de entradas 10197-10204 está mostrado como AC(HEX)= 10101 1100 (binário). Lendo da esquerda para a direita, isto mostra que as entradas 10204, 10202, 10199 estão ON. Os outros bytes de dados de entrada são decodificados do mesmo modo.

Devido à quantidade de status de entrada pedida, o último campo de dado mostrado como 35 HEX = 0011 0101 (binário) contém o status de somente 6 entradas (10213-10218) ao invés de 8 entradas. Os dois bits mais a esquerda são providos como zero para preencher o formato de 8-bit.

#### Read Holding Registers (Código de Função 03)

##### Consulta

Leitura dos Registros de Guarda de (3) permite que o usuário obtenha o conteúdo binário de registros de guarda 4xxx no escravo endereçado. Os registros podem armazenar os valores numéricos de timers e contadores associados que podem ser dirigidos para dispositivos externos. O endereçamento permite até 125 registros em cada pedido; no entanto, o dispositivo escravo específico pode ter restrições que baixem esta quantidade máxima. Os registros são numerados de zero (40001= zero, 40002= um, etc.). O modo broadcast não sendo permitido.

No exemplo abaixo, leia registros 40108 a 401110, a partir do escravo 584 número 11.

ADR	FUNC	DATA START REG HO	DATA START REG LO	DATA #OF REGS HO	DATA #OF REGS LO	ERROR CHECK FIELD
11	03	00	6B	00	03	CRC

Figura C5, Read Holding Register Query Message

##### Resposta

O escravo endereçado responde com endereço e código de função, seguido do campo de informação. O campo de informação contém 1 byte que descreve a quantidade de bytes de dados que devem retornar. O conteúdo dos registros pedido (DADO) são dois bytes cada, sendo que o conteúdo binário fica alinhado a direita dentro de cada par de caracteres. O primeiro byte inclui os bits de ordem alta e o segundo byte os bits de ordem baixa.

Uma vez que o dispositivo de interface de escravo é normalmente atendido no fim do escaneamento do controlador, o dado refletirá o conteúdo de registro no fim do escaneamento. Alguns escravos limitarão a quantidade de conteúdo do registro provido em cada escaneamento, assim para uma quantidade maior de registros serão feitas várias transmissões usando conteúdo de registro a partir de escaneamentos seqüenciais.

No exemplo, os registros 40108-40110 têm respectivamente os conteúdos 555, 0, e 100.

ADR	FUNC	BYTE CNT	H.O. DATA	L.O. DATA	H.O. DATA	L.O. DATA	H.O. DATA	L.O. DATA	ERROR CHECK FIELD
11	03	06	02	2B	00	00	00	64	CRC

Figura C6, Read Holding Registers Response Message

#### Read Input Registers (Código de Função 04)

##### Consulta

O código de função 04 obtém o registro de entrada do controlador nos endereços 3xxxx. Estas localizações recebem seus valores de dispositivos conectados à estrutura E/S e somente poderão ser referenciadas, não podendo ser alteradas de dentro do controlador. O endereçamento permite que até 125 registros sejam obtidos em cada pedido; no entanto, o dispositivo escravo específico pode ter restrições que baixem esta quantidade máxima. Os registros são numerados de zero (30001= zero, 30002= um, etc.). O modo Broadcast não é permitido.

O exemplo pede o conteúdo do registro 3009 no escravo número 11.

ADR	FUNC	DATA START REG HO	DATA START REG LO	DATA #OF REGS HO	DATA #OF REGS LO	ERROR CHECK FIELD
11	04	00	08	00	01	CRC

Figura C7, Read Input Register Query Message

##### Resposta

O escravo endereçado responde com endereço e código de função seguido do campo de informação. O campo de informação contém 1 byte que descreve a quantidade de bytes de dado que deve retornar. O conteúdo dos registros pedidos (DATA) é 2 bytes cada, sendo que o conteúdo binário fica alinhado a direita dentro de cada par de caracteres. O primeiro byte inclui os bits de ordem alta e o segundo byte os bits de ordem baixa.

Uma vez que a interface de escravo é normalmente atendida no fim do escaneamento do controlador, os dados refletirão o conteúdo do registro no fim do escaneamento. Cada PC limita a quantidade de conteúdo de registro provida a cada escaneamento; assim para grandes quantidades de registros, serão requeridos múltiplos escaneamentos PC, e os dados providos serão a partir de escaneamentos seqüenciais.

No exemplo abaixo, o registro 3009 contém o valor decimal 0.

ADR	FUNC	BYTE COUNT	DATA INPUT REG HO	DATA INPUT REG LO	ERROR CHECK FIELD
11	04	02	00	00	E9

Figura C8, Read Input Register Response Message

#### Force Single Coil (Código de Função 5)

##### Consulta

Esta mensagem força um único coil quer ON ou OFF. Qualquer coil dentro do controlador pode ser forçado para um destes estados. (ON/OFF). No entanto, uma vez que o controlador esteja escaneando ativamente, a menos que o coil esteja desabilitado, o controlador pode também alterar o estado do coil. Os coils são numerados de zero (coil 0001= zero, coil 0002= um, etc.). O valor de dado 65,280 (FF00HEX) set coil ON e o valor zero OFF, todos os demais valores são ilegais e não afetam o coil.

O uso de endereço escravo 00 (Modo broadcast) força todos escravos anexos a modificar o coil desejado.

##### NOTA

As funções 5, 6, 15, 16 são somente mensagens que serão reconhecidas como válida para broadcast.

O exemplo abaixo é um pedido para o número de escravo 11 ativar o coil 0173

ADR	FUNC	DATA COIL # HO	DATA COIL # LO	DATA ON/OFF IND	DATA	ERROR CHECK FIELD
11	05	00	AC	FF	00	CRC

Figura C9, Force Single Coil Query Message

#### Resposta

A resposta normal para o pedido de Comando é retransmitir a mensagem depois de o estado de coil tiver sido alterado.

ADR	FUNC	DATA COIL # HO	DATA COIL # LO	DATA ON/ OFF	DATA	ERROR CHECK FIELD
11	05	00	AC	FF	00	CRC

Figura C10, Force Single Coil Response Message

Um coil será forçado através da função Modbus 5 a despeito de se o coil endereçado estiver desabilitado ou não. (Nos produtos ProSoft, o coil somente será afetado se a Lógica Ladder estiver implementada).

#### NOTA

O protocolo Modbus não inclui funções padrão para testar ou mudar estado DESABILITADO de entradas ou saídas discretas. Onde aplicável, isto poderá ser realizado através de comandos de programa específicos (Nos produtos ProSoft, isto somente será realizado através de programação de Lógica Ladder).

Coils reprogramados no programa de lógica do controlador não são automaticamente cancelados na energização. Assim, se este coil for ativado pelo Código de Função 5, e (mesmo meses depois) for conectada uma saída àquele coil, a saída será "quente"

#### Preset Single Register (Código de Função 06)

##### Consulta

A Função 06 permite que o usuário modifique o conteúdo do holding\_register. Qualquer holding\_register no controlador pode ter seu conteúdo alterado por esta mensagem. No entanto, uma vez que o controlador esteja escaneando ativamente, ele também pode alterar o conteúdo de qualquer holding\_register. Os valores são providos em binário até a máxima capacidade do controlador. Os bits de ordem alta não usados devem ser colocados em zero (modo Broadcast) e todos os controladores carregarão o registro especificado com os conteúdos especificados.

#### NOTA

As funções 5, 6, 15 são as únicas mensagens reconhecidas como válidas para broadcast.

ADR	FUNC	DATA START REG HO	DATA START REG LO	DATA #OF REGS HO	DATA #OF REGS LO	ERROR CHECK FIELD
11	06	00	01	00	03	CRC

Figura C11, Preset Single Register Query Message

A resposta a um pedido de preset single register deve retransmitir a mensagem de consulta depois de o registro tiver sido alterado.

ADR	FUNC	DATA REG H	DATA REG LO	DATA INPUT REG HO	DATA INPUT REG LO	ERROR CHECK FIELD
11	06	00	01	00	03	CRC

Figura C12, Preset Single Register Response Message

**Force Multiple Coils****Consulta**

Esta mensagem força cada coil em um bloco consecutivo de coils para o estado ON/OFF desejado. Qualquer coil no controlador pode ser forçado para um dos estados (ON/OFF). No entanto, uma vez que o controlador esteja ativamente escaneando, amenos que os coils estejam desabilitados, o controlador pode também alterar estado do coil. Os coils são numerados a partir de zero (coil 00001= zero, coil 00002= um, etc.). O estado desejado de cada coil é agrupado no campo de dado, um bit para cada coil (1= ON, 0= OFF). O uso de endereço de escravo 0 (Modo broadcast) força todos escravos anexados a modificar os coils desejados.

**NOTA**

As funções 5, 6, 15, 16 são as únicas mensagens (diferentes de Teste de Diagnóstico de Loopback) válidas para broadcast.

O exemplo força 10 coils a iniciarem no endereço 20 (13HEX). Os dois campos de dados, CD= 1100 e 00= 0000 000, indicam que coils 27, 26, 23, 22 e 20 devem ser forçados ON.

ADR	FUNC	H.O. ADD	L.O. ADD	QUANTITY	BYTE CNT	DATA COIL STATUS 20-27	DATA COIL STATUS 28-29	ERROR CHECK FIELD
11	0F	00	13	00 0A	02	CD	00	CRC

Figura C15, Force Multiple Coils Query Message

**Resposta**

A resposta normal será eco do endereço de escravo, código de função, e quantidade de coils forçados.

ADR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	ERROR CHECK FIELD
11	0F	00	13	00 0A	CRC

Figura C16, Force Multiple Coils Response Message

A escrita de coils através da função Modbus 15 será realizada quer os coils endereçados estejam desabilitados ou não.

Coils que forem reprogramados no programa lógico não serão automaticamente cancelados na energização. Assim, se o coil for ativado pelo código de função 15 e se (mesmo meses depois) uma saída for conectada àquele coil, a saída será quente.

**Preset Multiple Registers****Consulta**

Holding Registers no controlador podem ter seu conteúdo modificado por esta mensagem (no máximo 60 registros). No entanto, desde que o controlador esteja ativamente escaneando, ele também poderá alterar o conteúdo de qualquer holding register a qualquer tempo. Os valores são providos em binário até a capacidade máxima do controlador (16 bits para 184/384 e 584); bits de ordem alta que não foram utilizados devem ser colocados em zero. Quando registros os registros forem especificados com conteúdo. especificado

NOTA Códigos de função 5, 6, 15, 16 são as únicas mensagens reconhecidas como válidas

ADR	FUNC	H.O. ADD	L.O. ADD	QUANTITY	BYTE CNT	H.O. DATA	L.O. DATA	H.O. DATA	L.O. DATA	ERROR CHECK FIELD
11	10	00	87	00 02	04	00	0A	01	02	CRC

Figura C13, Preset Multiple Coils Query Coils Message

**Resposta**

A recepção normal a uma consulta de função 16 é eco do endereço, código de função, endereço de início, e número de registros a serem carregados.

ADR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	ERROR CHECK FIELD
11	10	00	87	00 02	56

Figura C14, Preset Multiple Registers Response Message

## D Configurações de Jumper

### Visão Geral do Hardware

Ao comprar o produto MCM, há duas configurações disponíveis. As escolhas são:

<u>PLC</u>	Descrição do Num de CAT da ProSoft	<u>SLC</u>
Módulo provido pela ProSoft	3100	3150

Ao comprar o módulo da ProSoft\_Technology, a configuração de jumper será instalada de fábrica nas posições default para teste antes da entrega.

### Configurações de Jumper do Módulo

A seção a seguir detalha as configurações de jumper para as soluções de plataforma 1771 e 1746. Conforme necessário, são salientadas as diferenças entre as soluções baseadas em módulo e firmware.

#### 3100 para plataforma 1771

A seguir são mostradas as posições de jumper para o módulo 3100-MCM da ProSoft.

<u>Jumper</u>	<u>3100</u>
JW1	N/A
JW2	N/A
JW3	N/A
JW4	Modo PGM/Run Flash
JW5	8 Pt
JW6	Não usado
JW7	Habilitado
JW8	Config RS232/422/485 de Porta 2
JW9	Config RS232/422/485 de Porta 1

<b>JW4</b>	<b>Seleção de Modo Pgm/Run Flash</b>	<b>Posição Run</b>
	A posição deste jumper deve ser somente alterada se for necessário reprogramar a memória FLASH MCM. Isto somente precisará ser feito, se o módulo sofrer upgrade no campo para uma versão mais nova ou firmware.	
<b>JW5</b>	<b>Ponto 8/16 no barramento</b>	<b>Ponto 8</b>
	O módulo deve ser operado na configuração de ponto 8 a menos que especificamente indicado de outra forma pela fábrica.	
<b>JW7</b>	<b>Bateria Habilitada/ Desabilitada</b>	<b>Habilitada</b>
	Este jumper deve ser colocado na posição Habilitado quando o módulo estiver energizado. Embora não crítico para operação do módulo, provê back_up de alguns registros de dados no módulo durante falta de energia ou reset.	
<b>JW8/9</b>	<b>Configuração RS de Portas 1 e 2</b>	<b>R_S232, RS_422, RS_485</b>
	RS232 é default de fábrica, mas todas opções são suportadas pelo firmware MCM	

#### 3150 para plataforma 1746

A seguir as posições de jumper para o módulo 3150-MCM da ProSoft\_Technology

<u>Jumper</u>	<u>3100</u>
JW1	Conforme necessário
JW2	Conforme necessário
JW3	N/A
JW4	N/A

<b>JW1/2</b>	<b>Configuração RS de Portas 1 e 2</b>	<b>Posição RS_232</b>
	RS232 é default de fábrica, mas RS422 e RS485 são suportadas pelo firmware e hardware. Ver o diagrama a seguir:	

## **E Histórico de Revisões do Produto.**

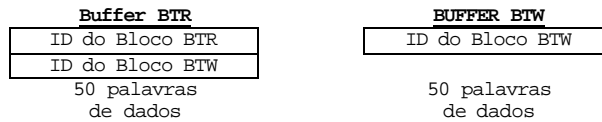
24/05/95	<b>Revisão 1.40- 2</b> primeiro release do produto.
04/07/95	<b>Revisão 1.41- 3</b> Conserto de FC15 quando a escrita tiver menos de 8 bits Adição de suporte adicional para operação em loop nas portas.
23/07/95	<b>Revisão 1.42- 4</b> Conserto de problema que causa fechamento de ambas portas quando no modo multi-Mestre.
01/09/96	<b>Revisão 1.44</b> - Mudança de atrasos RTS para suportar valores de palavra ao invés de apenas valores de bit. Agora capaz de atrasos RTS até 65534. - Re-Link de 1.42 para suportar upgrade de firmware 1771-db. - Adição de capacidade de escrita acionada por evento para porta mestre. - Adição de modo Pass_Through para a porta escravo - Suporte a ponto flutuante tipo variável (FC 3, 6, 16 maior que registro 47001). - Incorporação de melhor lógica de imunidade a ruído para iniciar mensagens. - Adição de suporte para mais que 255 caracteres para buffer para suportar todas necessidades do modo ASCII. - Adição de capacidade routing Mestre-Escravo até 6 endereços de escravo - Adição de valores de configuração para Início de Bloco de Leitura e Início de Bloco de Escrita. - No Modo Mestre, somente retorna Erro 8 depois de tentar 3 vezes
3/12/96	<b>Revisão 2.0 liberada</b>

## F Uso de Valores de Quantidade de Blocos de Leitura, Escrita, e Comando

Como parte do processo de configuração, o Usuário é capaz de configurar diversos parâmetros no Bloco de Dados de Configuração de Comunicação que exerce um grande impacto em como o módulo transfere dados com processador PLC/SLC.

### Visão Geral

Como mostrado nas seções 4 e 5 do manual, o buffer BTR contém números de ID dos Blocos BTR e BTW. O ID do Bloco BTR é usado para identificar o conteúdo dos dados, enquanto o ID do Bloco BTW é usado pela Lógica Ladder para determinar qual dado mover para o módulo. Em diagrama o relacionamento é o seguinte:



### Parâmetros de Configuração

Três parâmetros que são importantes para transferir dados são:

**Read Block Count:** Este valor representa o número de 50 blocos de dados de palavras que devem ser transferidos do Módulo MCM para o processador. Os blocos que voltam do módulo iniciam no valor fornecido no registro de "Início de Bloco de Leitura" e se incrementam a partir daí.

**WriteBlock Count:** Este valor representa o número de 50 blocos de dados de palavra que deve ser transferido do processador para o Módulo MCM

**CmdBlock Count:** Este valor representa o número de 50 blocos de comando de palavra que deve ser transferido do processador para o Módulo MCM.

Estes valores são usados pelo módulo para determinar como os Códigos ID de Blocos BTW e BTR devem ser manipulados. Parte da funcionalidade que o módulo provê se destina a controlar incremento e reset dos códigos de ID de Blocos BTR e BTW. Isto foi feito com propósito de limitar a quantidade de Lógica Ladder requerida para suportar o módulo.

### Operação do Módulo

Como resulta dos parâmetros de configuração fornecidos, o módulo cicla através da gama de Blocos BTW e BTR. O ciclo se baseia nas seguintes equações:

#### ID do Bloco BTW

Se (ID do Bloco BTW = Cont. Bloco Escrita),  
então ID do Bloco BTW = 80  
ainda  
se (ID do Bloco BTW = 80 + Cont. Bloco Com),  
então ID do Bloco BTW= Início do Bloco de Escrita  
ainda  
ID do Bloco BTW= ID do Bloco BTW + 1

#### ID do Bloco BTR

Se (ID do Bloco BTR = Cont. Bloco de Escrita),  
então ID do Bloco BTR= Início de Bloco de Leitura  
ainda  
ID do Bloco BTR= ID do Bloco BTR + 1

Como exemplo, assumindo que estamos configurados com os seguintes valores:

Cont. de ReadBlock	4
Cont. de WriteBlock	1
Cont. de CmdBlock	2
Início do ReadBlock	1
Início do WriteBlock	0

Estes valores de configuração levam ao seguinte ciclo de códigos de ID de Bloco:

ID do Bloco BTW	ID do Bloco BTR
0	1
80	2
81	3
0	4
80	1
81	2
0	3

Note que não há uma relação implícita entre o valor absoluto nos IDs de Bloco BTW e BTR

## **G**                    **Lógica Ladder de Exemplo**

A seguinte lógica de exemplo foi provida para ajudá-lo a desenvolver aplicações mais efetivamente. Estes exemplos são providos em manuais separados intitulado Lógica Ladder de Exemplo (dois destes manuais são disponíveis, um para PLC e outro para SLC).

### Exemplos de Modo Escravo

Exemplo #1: Modo Escravo com Pass\_Through - Configuração Mínima

MCM5EX1S	PLC5
MCM3EX1S	SLC5/03

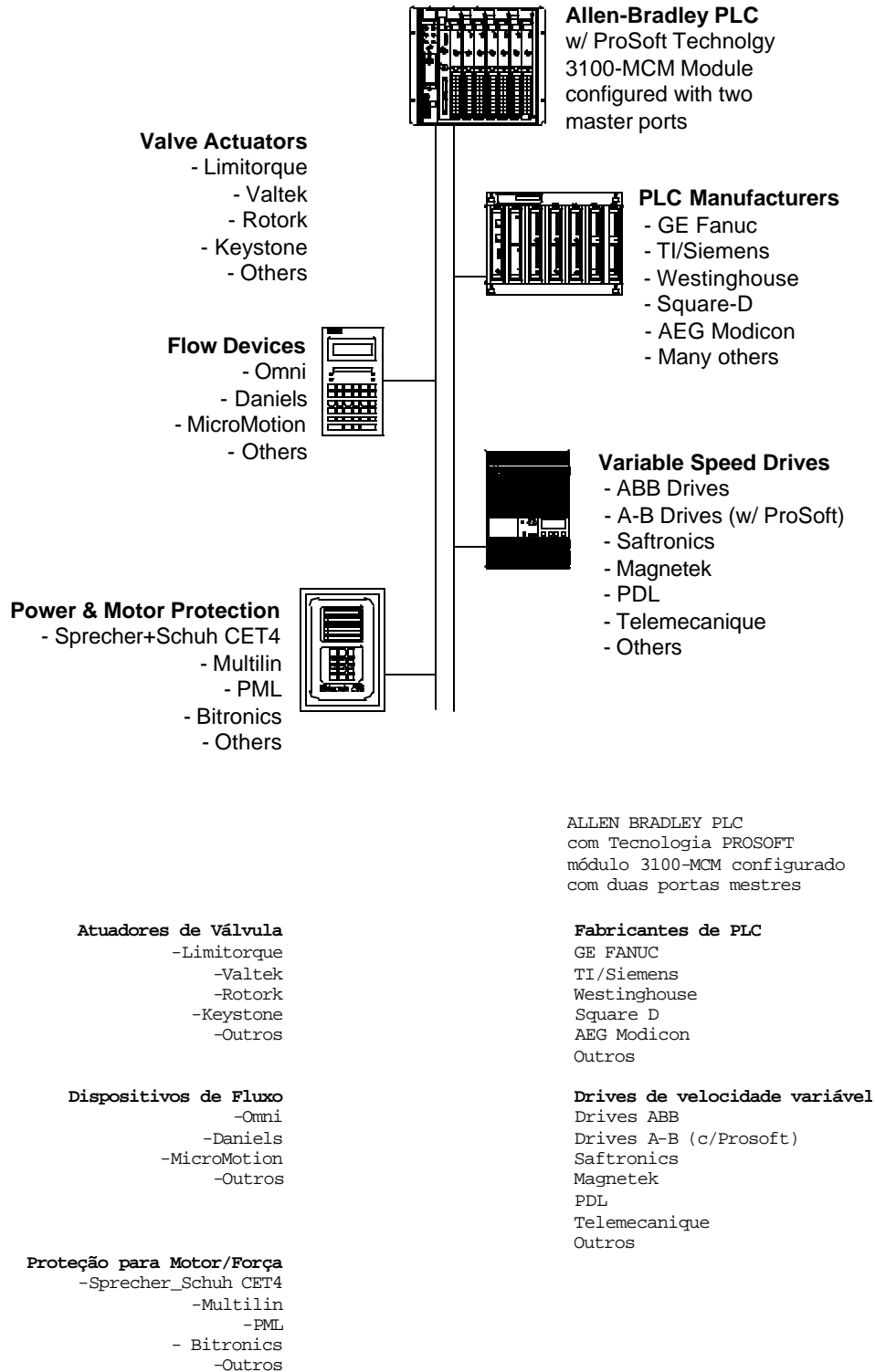
### Exemplo de Modo Mestre

Exemplo #2: Modo Mestre com Controle de Comando

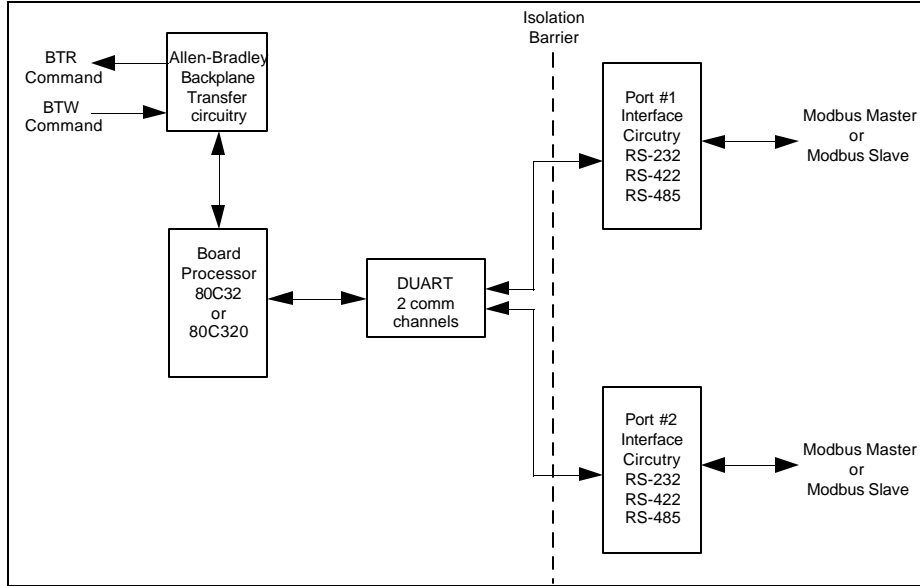
MCM5EX2M	PLC5
MCM3EX2M	SLC5/03

**ANEXO**  
**TRADUÇÃO DE FIGURAS E DIAGRAMAS**

**CAPA**

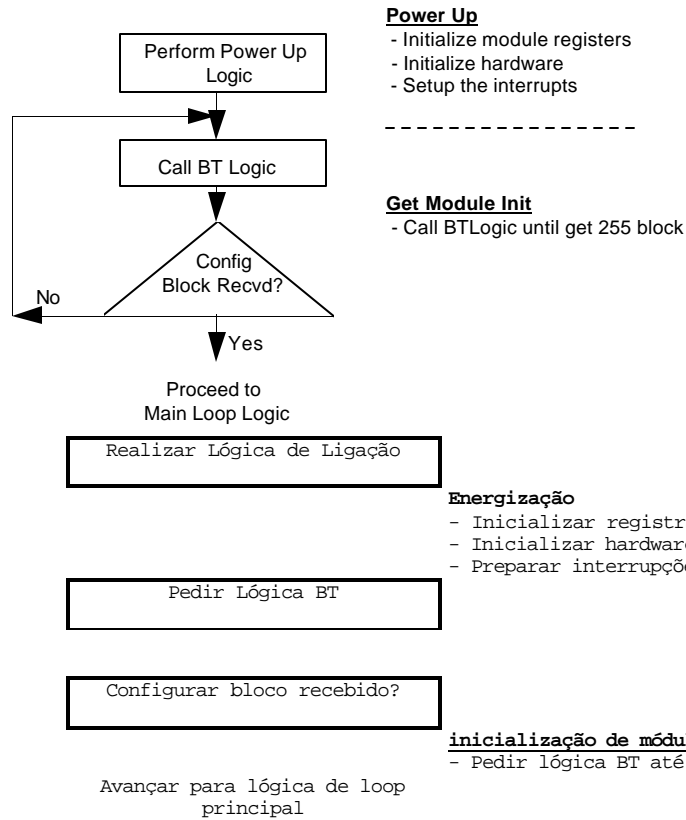


1.2 - página 2

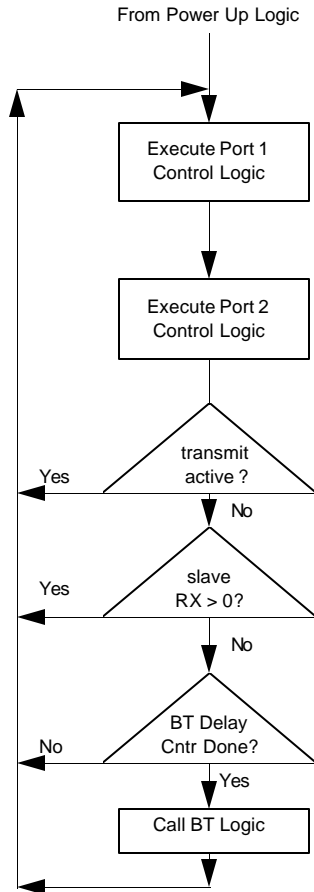


Comando <i>BTR</i>		Circuitos de Transferência de Barramento Allen_Bradley				
Comando <i>BTW</i>					Circuitos de Interface da Porta 1 RS_232 RS_422 RS_485	Mestre Modbus ou Escravo Modbus
		Processador 80C32 ou 80C320	2 Canais de Com Duart			
					Circuitos de Interface da Porta 2 RS_232 RS_422 RS_485	Mestre Modbus ou Escravo Modbus

1.3.1 página 3



### 1.3.2- página 4



#### Port 1 Control Logic

- if port in RX mode, then test for message received
- if port in TX mode, then test if message transmit has completed
- If port is a master and ready for new command then create new command

#### Port 2 Control Logic

- if port in RX mode, then test for message received
- if port in TX mode, then test if message transmit has completed
- If port is a master and ready for new command then create new command

#### Test Transmit Status

- If either port is in process of transmitting then do not execute BT logic. The module uses the CTS pin status to detect the transmit status

#### Test Slave port for characters

- If either slave port is in the process of receiving characters then skip BT logic. This is done to assure timely response from the slave

#### Test the Block Transfer Delay Counter

- If the Block Transfer Delay counter has incremented beyond the counter preset (set in config) then go ahead and perform block transfer

#### Execute Block Transfer Logic

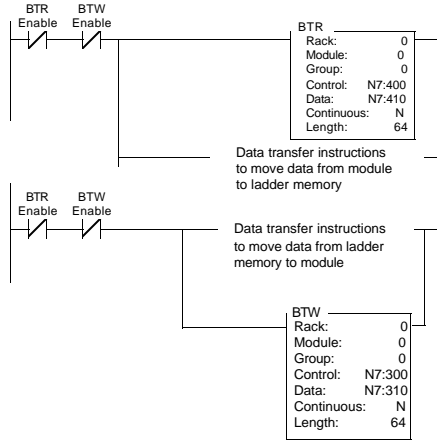
- Calls BT Logic which executes the BTR and BTW logic

(Cont)

a partir da lógica de energização

	Executar Lógica de Controle de Porta 1	<p><b><u>Lógica de Controle de Porta 1</u></b></p> <ul style="list-style-type: none"> <li>- se a porta está no modo RX, testa mensagem recebida</li> <li>- se a está porta no modo TX, testa se a transmissão foi completada</li> <li>- se a porta é mestre e estiver pronta para novo comando, então cria novo comando</li> </ul>
	Executar Lógica de Controle de Porta 2	<p><b><u>Lógica de Controle de Porta 2</u></b></p> <ul style="list-style-type: none"> <li>- se a porta está no modo RX, testa mensagem recebida</li> <li>- se porta está no modo TX, testar se a transmissão foi completada</li> <li>- se a porta é mestre e estiver pronta para novo comando, então criar um novo comando</li> </ul>
Sim	Transmissão ativa?	<p><b><u>Testa Status de Transmissão</u></b></p> <p>Se ambas portas estão em processo de transmissão então não execute a lógica BT. O módulo usa status de pino CTS para detectar status de transmissão</p>
	Não	
Sim	Esravo RX>0?	<p><b><u>Testa Porta Escravo por caracteres</u></b></p> <p>Se ambas portas estão recebendo caracteres então desconsidere lógica BT. Isto garante uma resposta temporizada do escravo siga e proceda a transferência de bloco</p>
	Não	
Não	Controle de Atraso BT realizado?	<p><b><u>Testa Contador de Atraso de Transferência de Bloco</u></b></p> <p>Se o contador de Atraso de Transferência de Bloco for incrementado além de um valor pré-ajustado, então</p>
	Sim	
	Pedir Lógica BT	<p><b><u>Executa Lógica de Transferência de Bloco</u></b></p> <p>Pedir Lógica BT que execute lógica BTR e BTW</p>

1.3.5 - página 8

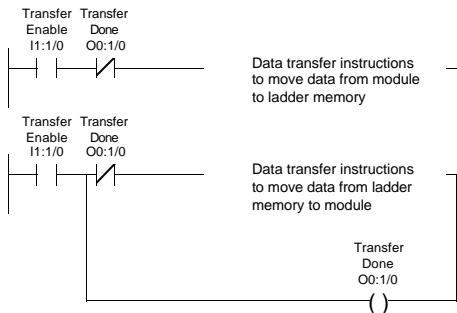


Transf            Transf  
Habilitada       Realizada

(Instruções de transferência de dados para mover dados do módulo para a memória Ladder)

(Instruções de transferência de dados para mover os dados da memória Ladder para o módulo)

1.3.5 página 9

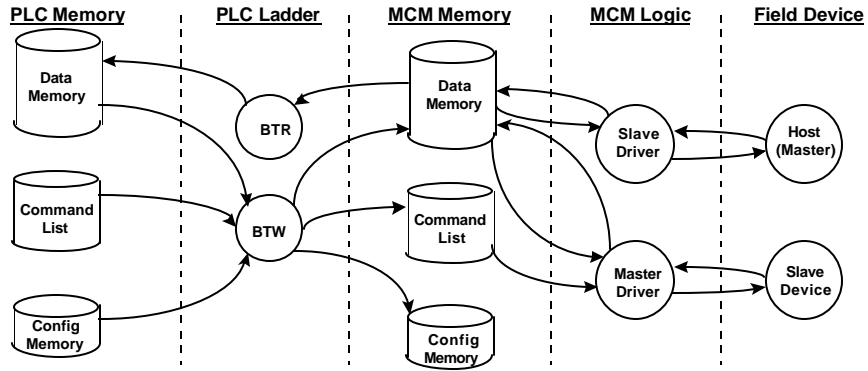


Transf            Transf  
Habilitada       Realizada

(Instruções de transferência de dados para mover dados do módulo para a memória Ladder)

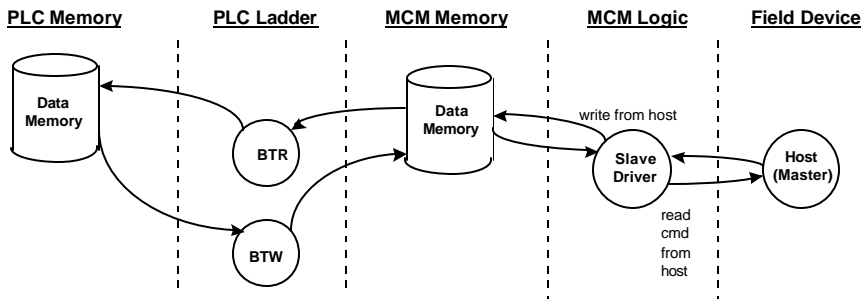
(Instruções de transferência de dados para mover os dados da memória Ladder para o módulo)

1.4.1- página 10



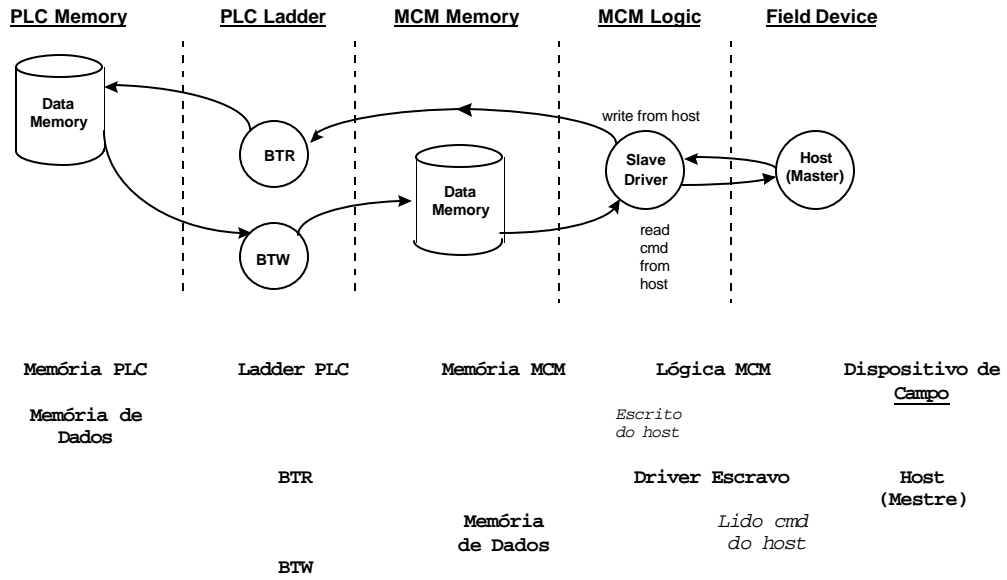
<u>Memória PLC</u>	<u>Ladder PLC</u>	<u>Memória MCM</u>	<u>Lógica MCM</u>	<u>Dispositivo de Campo</u>
Memória de Dados		Memória de Dados	Driver Escravo	Host (Mestre)
	BTR			
Lista de Comando		Lista de Comando	Driver Mestre	Dispositivo Escravo
	BTW			
Memória de Configuração		Memória de Configuração		

1.4.6 página 11

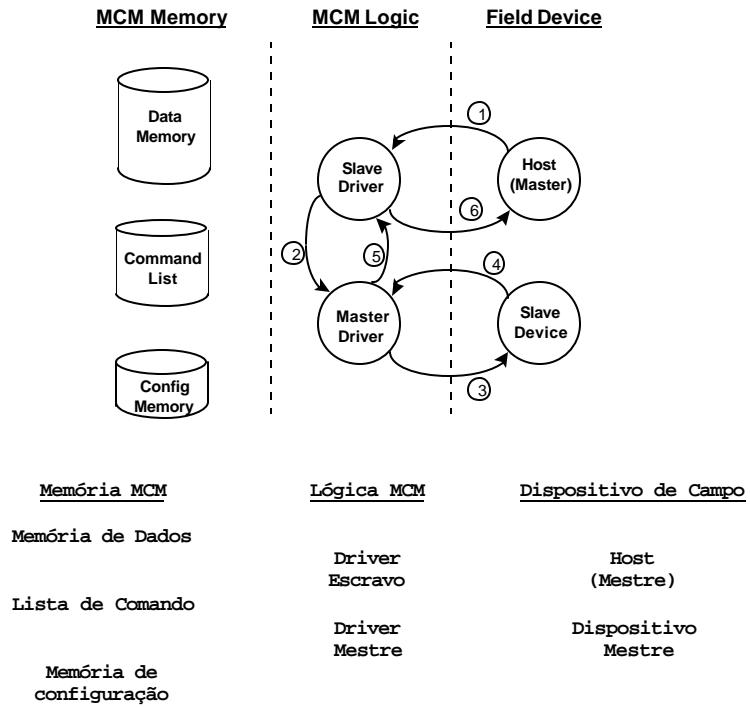


<u>Memória PLC</u>	<u>Ladder PLC</u>	<u>Memória MCM</u>	<u>Lógica MCM</u>	<u>Dispositivo de Campo</u>
Memória de Dados		Memória de Dados	Escrito do host Driver Escravo Lido cmd do host	Host (Mestre)
	BTR			
	BTW			

1.4.7 página 12



1.4.8 página 12



1.5.1 página 13

<b>Register Space</b>			<b>Read Command</b>	<b>Write Command</b>
0XXXX	Coils (Discrete Out)	1	1	5,15
		9999		
1XXXX	Inputs (Discrete In)	1	2	N/A
		9999		
3XXXX	Input Registers (Analog In)	1	4	N/A
		9999		
4XXXX	Holding Registers (Memory Regs)	1	3	6,16

<b>Espaço de Registro</b>			<b>Cmd de Leitura</b>	<b>Cmd de Escrita</b>
	Coils (Discrete_out)			
	Inputs Discrete In RGs deEntradas analógicos			
	Rgs de Entrada analógicos			
	Holding Rgs RGs de Mem			

4.1- página 17

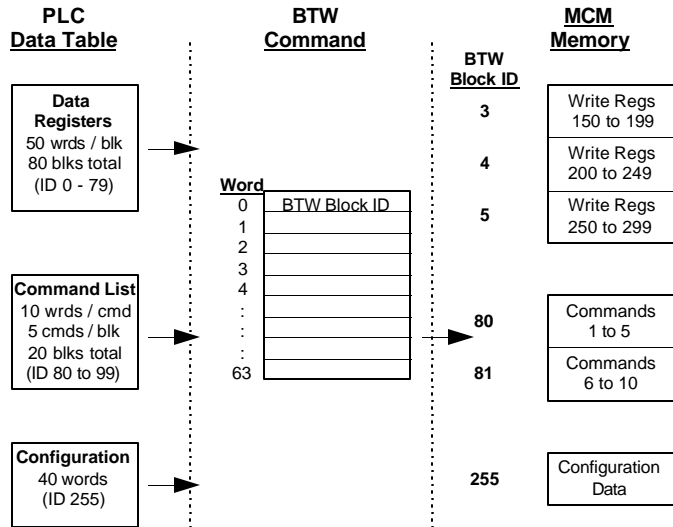
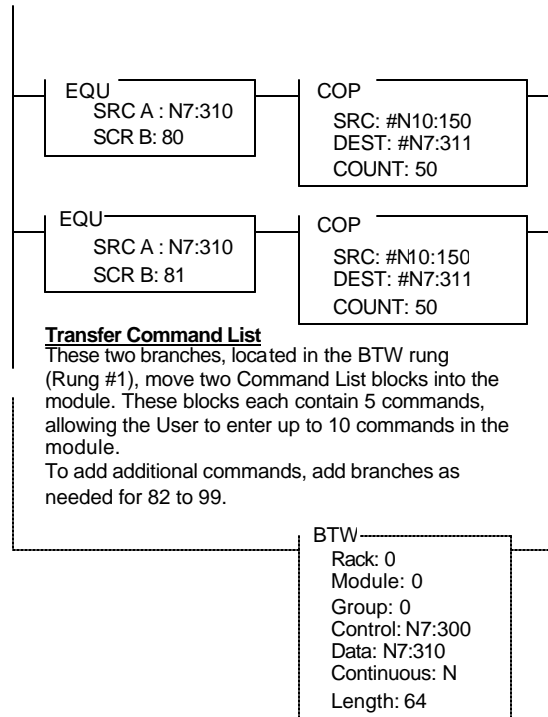


Tabela de dados PLC	Comando BTW	ID do Bloco BTW	Memória MCM
Registros de dados 50 pals/bl 80 blocos (ID 0-79)		3 4	Rgs de Escrita 159 a 199 Rgs de Escrita 200a 249 Rgs de Escrita 250 a 299
	pal 0 ID Bloco BTW 1 2 3 63	5 80	
		81	Comandos 1 a 5
			Comandos 6 a 10
Lista de Comando 10 pals/bl 20 blocos (ID 80 a 99)			
Configuração 40 pals (ID 255)		255	Dado de Configuração

#### 4.4.1 página 25



#### **Transfer Command List**

These two branches, located in the BTW rung (Rung #1), move two Command List blocks into the module. These blocks each contain 5 commands, allowing the User to enter up to 10 commands in the module.

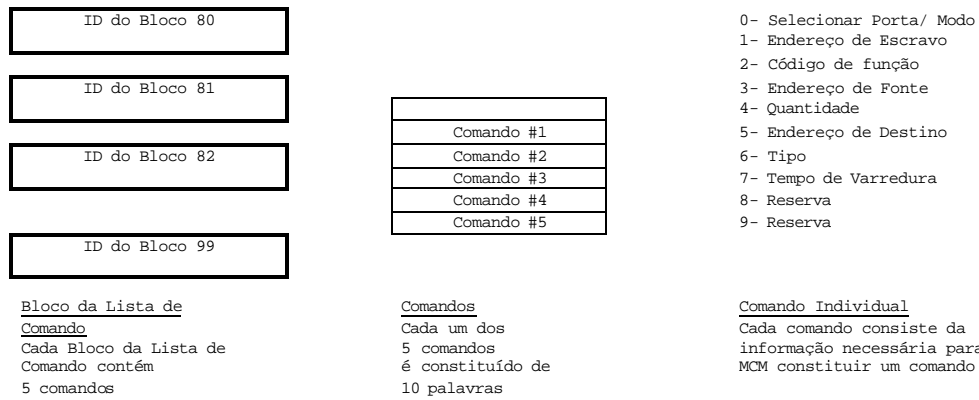
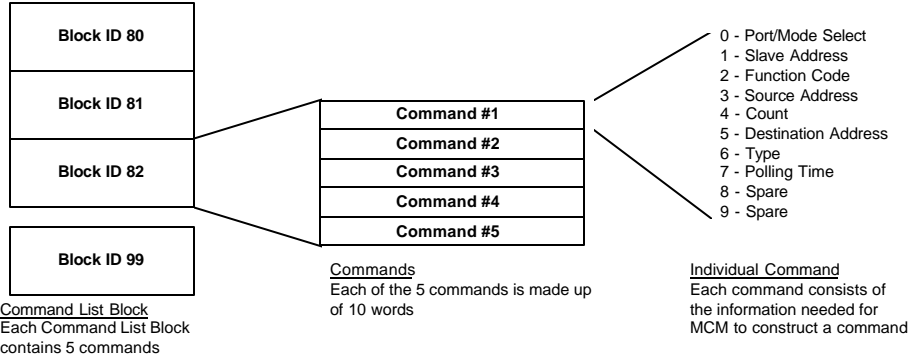
To add additional commands, add branches as needed for 82 to 99.

#### **Lista de Comando de Transferência**

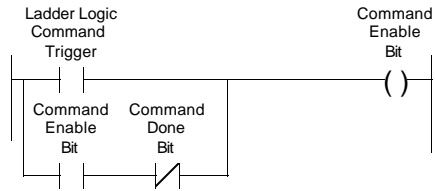
Estas duas ramificações localizadas na Linha BTW (Linha #1) movem dois blocos da Lista de Comando para o módulo. Cada um destes blocos contém 5 comandos permitindo que o Usuário forneça até 10 comandos no módulo

Para mais comandos, adicionar ramificações, conforme necessário, de 82 a 99.

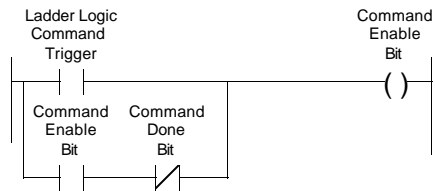
**4.4.2 Página 26**



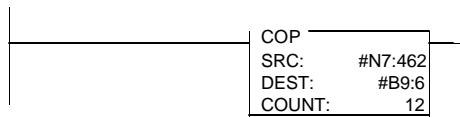
**4.5.2 página 29**



Gatilho de Cmd de Lógica Ladder			Bit Habilitado de Cmd
Bit Habilitado de Cmd	Bit Realizado de Cmd		



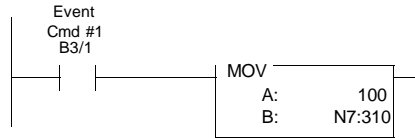
Gatilho de Cmd de Lógica Ladder			Bit Habilitado de Cmd
Bit Habilitado de Cmd	Bit Realizado de Cmd		



Copy the Command Done and Error bits from the BTR buffer to the data table.

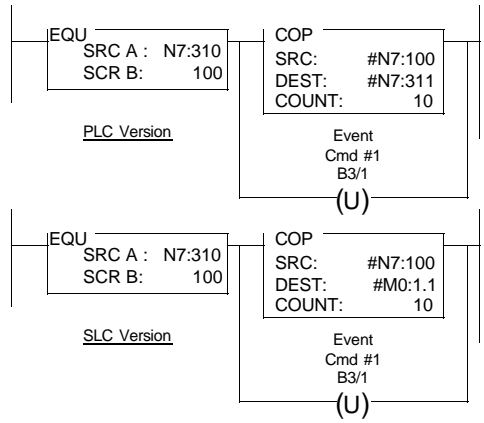
	Copie bits de erro e realizado de Cmd do Buffer BTR para a Tabela de dados
--	--

**4.6.1 página 30**



This branch is added to the Read rung, just above the MOV 255 to N7:310 branch. The B3/1 bit, selected here for example purposes only, is one-shot set in the ladder logic

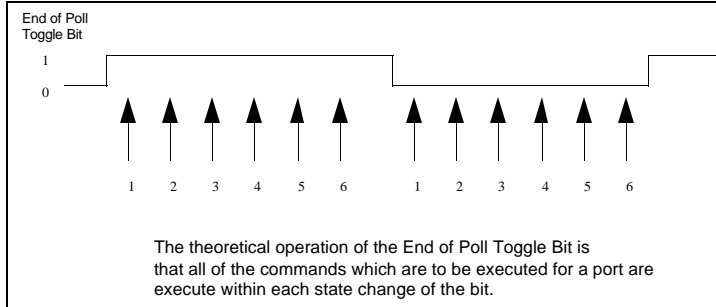
Esta ramificação é adicionada à linha de Leitura logo acima da ram MOV 255 para N7:310. O bit 3/1 como exemplo é colocado em um pulso na Lógica Ladder



This branch is added to the BTW rung, and serves to copy the Event Initiated Command block structure to the module and then Unlatches the command enable bit which was set in the ladder program.

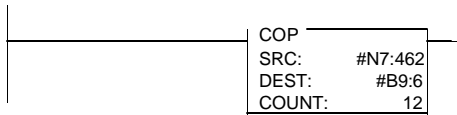
Esta ramificação é adicionada à linha BTW e serve para copiar a estrutura de Bloco de Comando iniciado por Evento e então libera o bit Habilitado de Comando colocado no programa Ladder

**5.1.5 página 36**



Fim do Bit Reversível de Varredura	
	A operação teórica do Fim do Bit Reversível de Varredura prevê que todos os comandos que devam ser executados para uma porta sejam executados em cada mudança de estado do bit

**5.3.2 página 40**



Copy the Command Done and Error bits from the BTR buffer to the data table.

Copiar bits de Comando Realizado e Erro do buffer BTR para a tabela de dados