

MANUAL

DO PROTOCOLO MODBUS

PLC1, PLC2 E POS2

Idioma: Português 0899.5230 P/1_____

1.1 MODBUS-RTU	4
1.1.1 Modos de Transmissão	4
1.1.2 Estrutura das Mensagens no Modo RTU	5
1.2 Operação do Cartão na Rede Modbus-RTU	7
1.2.1 RS-232	8
1.2.2 RS-485	8
1.3 Configurações do Cartão na Rede Modbus-RTU	8
1.3.1 Endereço do Cartão na Rede	8
1.3.2 Taxa de Transmissão	8
1.4 Acesso aos Dados do Cartão e do Drive	8
1.4.1 Funções Disponíveis e Tempos de Resposta	8
1.4.2 Endereçamento dos Dados	10
1.4.3 Faixa de endereços	14
1.4.4 Funções dos Marcadores de Sistema	15
1.5 Descrição Detalhada das Funções	15
1.5.1 Função 01 - Read Coils	15
1.5.2 Função 02 - Read Inputs Status	16
1.5.3 Função 03 - Read Holding Register	17
1.5.4 Função 04 - Read Input Register	18
1.5.6 Função 06 - Write Single Register	19
1.5.5 Função 05 - Write Single Coil	19
1.5.7 Função 15 - Write Multiple Coils	20
1.5.8 Função 16 - Write Multiple Registers	21
1.5.9 Função 43 - Read Device Identification	22
1.6 Erro de Comunicação	23
1.6.1 Mensagens de Erro	24

PROTOCOLO MODBUS-RTU NO CARTÃO PLC1, PLC2 e POS2

A seguir está descrita a explicação sobre o funcionamento dos cartões PLC1, PLC2 e POS2 na rede Modbus-RTU.

O baud rate é definido através de um parâmetro específico sendo possível os seguintes valores:

- 1 – 1200bps
- 2 – 2400bps
- 3 – 4800bps
- 4 – 9600bps (ajuste de fábrica)
- 5 – 19200bps

A comunicação é RS-232C, sem paridade, 8 bits e 2 stop bits.

Para implementarmos à uma rede, devemos utilizar os conversores MIW-02, que convertam a RS-232C (ponto a ponto) em RS-485 (multiponto).

O endereço do cartão na rede é definido através de um parâmetro específico, podendo variar de 1 a 247 (0 é o endereço para broadcast), tendo como ajuste de fábrica o valor 1.

Obs.: após alterar os parâmetros de endereço ou baudrate o sistema deve ser reinicializado.

O que é possível fazer no cartão utilizando o protocolo Modbus-RTU:

1 – Escrita / leitura em parâmetros e marcadores (comandos 3, 6 e 16):

Através do protocolo Modbus-RTU do cartão, podemos ler ou escrever em parâmetros da placa (P750...P899), parâmetros do drive (P000...P490), parâmetros da macro (PM0...PM31) além de marcadores word e marcadores float. Essa operação pode ser em um único parâmetro ou em um grupo de parâmetros.

2 – Leitura de entradas analógicas (comando 4):

Podemos ler o valor das entradas analógicas 1 e 2 do drive (endereços 101 e 102) e entrada analógica do cartão (endereço 1 quando disponível). A informação do valor é dada em 15 bits, ou seja, para o valor das entradas variando de 0 a 100%, temos um valor lido que vai de 0 a 32767.

3 – Escrita / leitura das entradas e saídas digitais e marcadores de bit (comandos 1, 2, 5 e 15):

Podemos ler e escrever nas saídas digitais, bem como ler as entradas digitais do cartão ou do drive, além da leitura e escrita nos marcadores do tipo bit, retentivos ou não. Essa operação pode ser em um único bit ou em um grupo.

Obs.: As saídas digitais ou marcadores de qualquer tipo, utilizados no programa do usuário, terão prioridade sobre a escrita em relação ao Modbus, ou seja, o programa do usuário sobrescreve o estado imposto pelo protocolo Modbus.

4 – Leitura da identificação da placa (comando 43):

Através do comando 43 pode-se ler dados de identificação da placa tais como, fabricante (WEG), modelo (PLC2, por exemplo) e a versão do firmware (V1.00, por exemplo).

Descrição detalhada do protocolo:

1.1 MODBUS-RTU

O protocolo Modbus foi desenvolvido em 1979. Atualmente, é um protocolo aberto amplamente difundido, utilizado por vários fabricantes em diversos equipamentos. A comunicação Modbus-RTU do cartão foi desenvolvida baseada em dois documentos:

- 1) MODBUS Protocol Reference Guide Rev. J, MODICON, June 1996.
- 2) MODBUS Application Protocol Specification, MODBUS.ORG, may 8th 2002.

Nestes documentos estão definidos o formato das mensagens utilizado pelos elementos que fazem parte da rede Modbus, os serviços (ou funções) que podem ser disponibilizados via rede e também, como estes elementos trocam dados na rede.

1.1.1 Modos de Transmissão

Na especificação do protocolo estão definidos dois modos de transmissão: ASCII e RTU. Os modos definem a forma como são transmitidos os bytes da mensagem.

Não é possível utilizar os dois modos de transmissão na mesma rede.

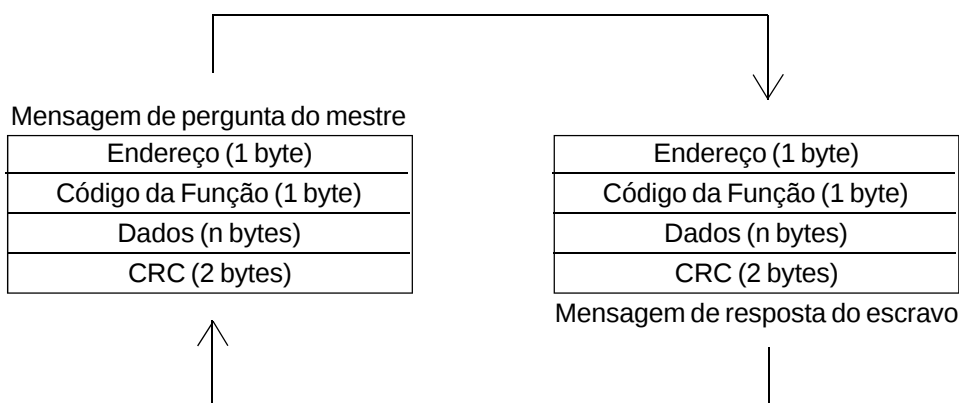
No modo RTU, cada palavra transmitida possui 1 start bit, oito bits de dados, 1 bit de paridade (opcional) e 1 stop bit (2 stop bits caso não se use bit de paridade). Desta forma, a seqüência de bits para transmissão de um byte é a seguinte:

Start	B0	B1	B2	B3	B4	B5	B6	B7	Stop	Stop
-------	----	----	----	----	----	----	----	----	------	------

1.1.2 Estrutura das Mensagens no Modo RTU

No modo RTU, cada byte de dados é transmitido como sendo uma única palavra, com seu valor diretamente em hexadecimal. O cartão utiliza somente este modo de transmissão para comunicação, não possuindo portanto, comunicação no modo ASCII.

A rede Modbus-RTU opera no sistema Mestre-Escravo, onde pode haver até 247 escravos, mas, somente um mestre. Toda comunicação inicia com o mestre fazendo uma solicitação a um escravo e este, responde ao mestre o que foi solicitado. Em ambos os telegramas (pergunta e resposta), a estrutura utilizada é a mesma: Endereço, Código da Função, Dados e CRC. Apenas o campo de dados poderá ter tamanho variável, dependendo do que está sendo solicitado.



Endereço:

O mestre inicia a comunicação enviando um byte, com o endereço do escravo para o qual se destina a mensagem. Ao enviar a resposta, o escravo também inicia o telegrama com o seu próprio endereço. O mestre também pode enviar uma mensagem destinada ao endereço 0 (zero), o que significa que a mensagem é destinada a todos os escravos da rede (broadcast). Neste caso, nenhum escravo irá responder ao mestre.

Código da Função:

Este campo também contém um único byte, onde o mestre especifica o tipo de serviço ou função solicitada ao escravo (leitura, escrita, etc.). De acordo com o protocolo, cada função é utilizada para acessar um tipo específico de dado.

Campo de Dados:

Campo com tamanho variável. O formato e conteúdo deste campo dependem da função utilizada e dos valores transmitidos. Este campo está descrito juntamente com a descrição das funções.

CRC:

A última parte do telegrama é o campo para checagem de erros de transmissão. O método utilizado é o CRC-16 (Cycling Redundancy Check). Este campo é formado por dois bytes, onde primeiro é transmitido o byte menos significativo (CRC-), e depois o mais significativo (CRC+).

O cálculo do CRC é iniciado primeiramente carregando uma variável de 16 bits (referenciado a partir de agora como variável CRC) com o valor FFFFh. Depois, são executados os passos de acordo com a seguinte rotina:

1. Submete-se o primeiro byte da mensagem (somente os bits de dados - start bit, paridade e stop bit não são utilizados) a uma lógica XOR (OU exclusivo) com os 8 bits menos significativos da variável CRC, retornando o resultado na própria variável CRC.
2. Então, a variável CRC é deslocada uma posição à direita, em direção ao bit menos significativo e a posição do bit mais significativo é preenchida com 0 (zero).
3. Após este deslocamento, o bit de *flag* (bit que foi deslocado para fora da variável CRC) é analisado, ocorrendo o seguinte:
 - ☒ Se o valor do bit for 0 (zero), nada é feito
 - ☒ Se o valor do bit for 1, o conteúdo da variável CRC é submetido a uma lógica XOR com um valor constante de A001h e o resultado é retornado à variável CRC.
4. Repete-se os passos 2 e 3 até que oito deslocamentos tenham sido feitos.
5. Repete-se os passos de 1 a 4, utilizando o próximo byte da mensagem, até que toda a mensagem tenha sido processada.

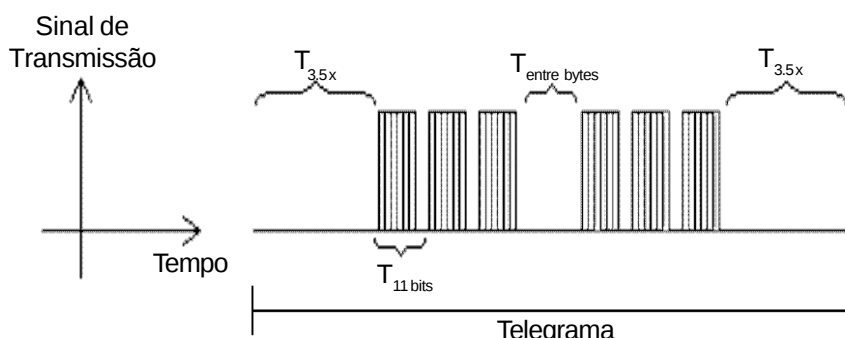
O conteúdo final da variável CRC é o valor do campo CRC que é transmitido no final do telegrama. A parte menos significativa é transmitida primeiro (CRC-) e em seguida a parte mais significativa (CRC+).

Tempo entre Mensagens:

No modo RTU não existe um caracter específico que indique o início ou fim de um telegrama. Desta forma, o que indica quando uma nova mensagem começa ou quando termina, é a ausência de transmissão de dados na rede, por um tempo mínimo de 3,5 vezes o tempo de transmissão de uma palavra de dados (11 bits). Sendo assim, caso um telegrama tenha iniciado após a decorrência deste tempo mínimo sem transmissão, os elementos da rede irão assumir que o caracter recebido representa o início de um novo telegrama. E da mesma forma, os elementos da rede irão assumir que o telegrama chegou ao fim após decorrer este tempo novamente.

Se durante a transmissão de um telegrama, o tempo entre os bytes for maior que este tempo mínimo, o telegrama será considerado inválido, pois o cartão irá descartar os bytes já recebidos e montará um novo telegrama com os bytes que estiverem sendo transmitidos.

A tabela a seguir nos mostra os tempos para três taxas de comunicação diferentes.



Taxa de Comunicação	$T_{11 \text{ bits}}$	$T_{3.5x}$
9600 bits/seg	1.146 ms	4.010 ms
19200 bits/seg	573 μ s	2.005 ms

$T_{11 \text{ bits}}$ = Tempo para transmitir uma palavra do telegrama.
 $T_{\text{entre bytes}}$ = Tempo entre bytes (não pode ser maior que $T_{3.5x}$).
 $T_{3.5x}$ = Intervalo mínimo para indicar começo e fim do telegrama ($3.5 \times T_{11 \text{ bits}}$).

1.2 Operação do Cartão na Rede Modbus-RTU

Os cartões operam como escravos da rede Modbus-RTU, sendo que toda a comunicação inicia com o mestre da rede solicitando algum serviço para um endereço na rede. Se o cartão estiver configurado para o endereço correspondente, então trata o pedido e responde ao mestre o que foi solicitado. Os cartões utilizam uma interface serial para se comunicar com a rede Modbus-RTU. Existem duas possibilidades para a conexão física entre o mestre da rede e um cartão ; interfaces RS232 e RS485:

MANUAL - PROTOCOLO MODBUS-RTU NO CARTÃO PLC1, PLC2 e POS2

- 1.2.1 RS-232
- ☑ Utilizada para conexão ponto a ponto (entre um único escravo e mestre).
 - ☑ Distância máxima: 10 metros.
 - ☑ Níveis de sinal seguem a EIA STANDARD RS-232C.
 - ☑ Três fios: transmissão (TX), recepção (RX) e retorno (0V).
 - ☑ Deve-se utilizar o módulo RS-232 Serial Interface.
- 1.2.2 RS-485
- ☑ Disponível através do conversor MIW-02 conectado à RS-232 do cartão.
 - ☑ Utilizada para conexão multiponto (vários escravos e um mestre).
 - ☑ Distância máxima: 1000 metros (utiliza cabo com blindagem).
 - ☑ Níveis de sinal seguem a EIA STANDARD RS-485.
- 1.3 Configurações do Cartão na Rede Modbus-RTU
- Para que o cartão possa se comunicar corretamente na rede, além da conexão física, é necessário configurar o endereço da mesma na rede, bem como a taxa de transmissão.
- 1.3.1 Endereço do Cartão na Rede
- ☑ Definido através do parâmetro 764 ou 788.
 - ☑ Cada escravo na rede deve possuir um endereço diferente dos demais.
 - ☑ O mestre da rede não possui endereço.
 - ☑ É necessário conhecer o endereço do escravo mesmo que a conexão seja ponto a ponto.
- 1.3.2 Taxa de Transmissão
- ☑ Definida através do parâmetro 765 ou 789.
 - ☑ Taxa de transmissão: 1200, 2400, 4800, 9600 ou 19200 kbits/seg.
 - ☑ Paridade: Nenhuma.
 - ☑ Todos os escravos, e também o mestre da rede, devem estar utilizando a mesma taxa de comunicação e à mesma paridade.
- 1.4 Acesso aos Dados do Cartão e do Drive
- Através da rede, é possível acessar os parâmetros e as entradas e saídas digitais do cartão e do drive, bem como marcadores do cartão.
- 1.4.1 Funções Disponíveis e Tempos de Resposta
- O cartão, os parâmetros e marcadores foram definidos como sendo registradores do tipo *holding*. Além destes registradores, também é possível acessar diretamente entradas e saídas digitais da placa ou do drive, e também os marcadores de bit, que são acessados utilizando as funções do tipo bit, do Modbus. Para acessar estes bits e registradores, foram disponibilizados os seguintes serviços (ou funções):

- ☑ Read Coils
Descrição: Leitura de bloco de bits internos ou bobinas.
Função: lê marcadores bit e saídas digitais do cartão e do drive.
Código da função: 01.
Broadcast: não suportado.
Tempo de resposta: 5 a 10 ms.
- ☑ Read Input Status
Descrição: Leitura de entradas digitais físicas.
Função: lê entradas digitais do cartão e do drive.
Código da função: 02.
Broadcast: não suportado.
Tempo de resposta: 5 a 10 ms.
- ☑ Read Holding Registers
Descrição: Leitura de bloco de registradores do tipo *holding*.
Função: lê marcadores word e float, parâmetros do cartão, do drive e da macro.
Código da função: 03.
Broadcast: não suportado.
Tempo de resposta: 5 a 10 ms.
- ☑ Read Input Registers
Descrição: Leitura de bloco de registradores do tipo *input*.
Função: lê entradas analógicas do cartão e do drive.
Código da função: 04.
Broadcast: não suportado.
Tempo de resposta: 5 a 10 ms.
- ☑ Write Single Coil
Descrição: Escrita em um único bit interno ou bobina.
Função: escreve em marcadores bit e saídas digitais do cartão ou do drive.
Código da função: 05.
Broadcast: suportado.
Tempo de resposta: 5 a 10 ms.
- ☑ Write Single Register
Descrição: Escrita em um único registrador do tipo *holding*.
Função: escreve em um marcador word, parâmetro do cartão, do drive e da macro, saídas analógicas do cartão e do drive.
Código da função: 06.
Broadcast: suportado.

- ☑ Tempo de resposta: 5 a 10 ms.
Write Multiple Coils
Descrição: Escrita em bloco de bits internos ou bobinas.
Função: escreve em múltiplos marcadores bit ou saídas digitais do cartão e do drive.
Código da função: 15.
Broadcast: suportado.
Tempo de resposta: 5 a 10 ms.
- ☑
Write Multiple Registers
Descrição: Escrita em bloco de registradores do tipo *holding*.
Função: escreve em múltiplos marcadores word ou float e parâmetros do cartão, do drive e da macro, saídas analógicas do cartão e do drive.
Código da função: 16.
Broadcast: suportado.
Tempo de resposta: 10 a 20 ms para cada registrador escrito.
- ☑ Read Device Identification
Descrição: Identificação do dispositivo.
Função: lê fabricante, modelo e versão do firmware do cartão.
Código da função: 43.
Broadcast: não suportado.
Tempo de resposta: 5 a 10 ms.

Obs.: Os escravos da rede Modbus-RTU são endereçados de 1 a 247. O endereço 0 (zero) é utilizado pelo mestre para enviar uma mensagem comum para todos os escravos (broadcast).

1.4.2 Endereçamento dos Dados

O endereçamento dos dados do cartão é feito com offset igual a zero, o que significa que o número do endereço equivale ao número dado. Os parâmetros, marcadores, bem como as entradas e saídas digitais, são disponibilizados a partir do endereço 0 (zero).

MANUAL - PROTOCOLO MODBUS-RTU NO CARTÃO PLC1, PLC2 e POS2

PARÂMETROS DO DRIVE		
Número do Parâmetro	Endereço Modbus	
	Decimal	Hexadecimal
P000	0	0000h
P100	100	0064h
⋮	⋮	⋮
P490	490	01EAh

PARÂMETROS DO CARTÃO		
Número do Parâmetro	Endereço Modbus	
	Decimal	Hexadecimal
P750	750	02Eeh
P800	800	0320h
⋮	⋮	⋮
P899	899	0383h

PARÂMETROS DA MACRO		
Número do Parâmetro	Endereço Modbus	
	Decimal	Hexadecimal
PM0	5000	1380h
⋮	⋮	⋮
PM31	5031	13A7h

MARCADORES WORD RETENTIVOS		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
MW6000	6000	1770h
⋮	⋮	⋮
MW6149	6149	1805h

MARCADORES WORD VOLÁTEIS		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
MW7000	7000	1B58h
⋮	⋮	⋮
MW7649	7649	1D1E

MARCADORES FLOAT RETENTIVOS		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
MF9500	9500	251Ch
⋮	⋮	⋮
MF9524	9524	2534h

MARCADORES FLOAT VOLÁTEIS		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
MF9000	9000	2328h
⋮	⋮	⋮
MF9174	9174	2306h

MARCADORES BIT RETENTIVOS		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
MX1000	1000	03E8h
⋮	⋮	⋮
MX1671	1671	0687h

MARCADORES BIT VOLÁTEIS		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
MX2000	2000	07D0h
⋮	⋮	⋮
MX3407	3407	0D4Fh

MARCADORES BIT DE SISTEMA		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
SX0	200	00C8h
SX1	201	0069h

MARCADORES WORD DE SISTEMA		
Número do Marcador	Endereço Modbus	
	Decimal	Hexadecimal
SW0	8000	1F40h
⋮	⋮	⋮
SW5	8005	1F45h

ENTRADAS ANALÓGICAS DO CATÃO		
Número da Entrada Analógica	Endereço Modbus	
	Decimal	Hexadecimal
IW1	1	1h

ENTRADAS ANALÓGICAS DO DRIVE		
Número da Entrada Analógica	Endereço Modbus	
	Decimal	Hexadecimal
IW101	101	0065h
IW102	102	0066h

SAÍDAS ANALÓGICAS DO CATÃO		
Número da Saída Analógica	Endereço Modbus	
	Decimal	Hexadecimal
QW1	8201	2009h
QW2	8202	200Ah

SAÍDAS ANALÓGICAS DO DRIVE		
Número da Saída Analógica	Endereço Modbus	
	Decimal	Hexadecimal
QW101	8301	2060h
QW102	8302	206Eh

MANUAL - PROTOCOLO MODBUS-RTU NO CARTÃO PLC1, PLC2 e POS2

ENTRADAS DIGITAIS DO CARTÃO		
Número da Entrada Digital	Endereço Modbus	
	Decimal	Hexadecimal
IX1	1	1h
⋮	⋮	⋮
IX9	9	9h

ENTRADAS DIGITAIS DO DRIVE		
Número da Entrada Digital	Endereço Modbus	
	Decimal	Hexadecimal
IX101	101	0065h
⋮	⋮	⋮
IX106	106	006Ah

SAÍDAS DIGITAIS DO CARTÃO		
Número da Saída Digital	Endereço Modbus	
	Decimal	Hexadecimal
QX1	1	1h
⋮	⋮	⋮
QX6	6	6h

SAÍDAS DIGITAIS DO DRIVE		
Número da Saída Digital	Endereço Modbus	
	Decimal	Hexadecimal
QX101	101	0065h
QX102	102	0066h
QX103	103	0067h

MANUAL - PROTOCOLO MODBUS-RTU NO CARTÃO PLC1, PLC2 e POS2

1.4.3 Faixa de Endereços

A tabela a seguir traz os tipos de elementos que podem ser utilizados na comunicação, suas faixas de endereços e a quantidade disponível dos mesmos em cada modelo de cartão.

Tipo	Cartões					
	PLC1 V1.7X CFW-09		PLC2 V1.1X CFW-09		POS2 V1.2X SCA05	
	[Início] [Fim]	Quantidade	[Início] [Fim]	Quantidade	[Início] [Fim]	Quantidade
Marcador de Bit Retentivos	%MX1000 %MX1671	672	%MX1000 %MX1671	672	%MX1000 %MX1671	672
Marcador de Bit Voláteis	%MX2000 %MX3407	1308	%MX2000 %MX3407	1308	%MX2000 %MX3407	1308
Marcador de Word Retentivos	%MW6000 %MW6099	100	%MW6000 %MW6099	100	%MW6000 %MW6099	100
Marcador de Word Voláteis	%MW7000 %MW7649	650	%MW7000 %MW7649	650	%MW7000 %MW7649	650
Marcador de Bit de Sistema	%SX0	1	%SX0 %SX1	2	%SX0	1
Marcador de Word de Sistema	%SW0 %SW5	6	%SW0 %SW5	6	%SW0 %SW5	6
Marcador de Float Retentivos	%MF9500 %MF9524	25	%MF9500 %MF9524	25	%MF9500 %MF9524	25
Marcador de Float Voláteis	%MF9000 %MF9174	175	%MF9000 %MF9174	175	%MF9000 %MF9174	175
Parâmetros do Usuário	%UW800 %UW899	100	%UW800 %UW899	100	%UW800 %UW899	100
Entradas Digitais Próprias	%IX1 %IX9	9	%IX1 %IX9	9	%IX1 %IX9	9
Entradas Digitais do Drive	%IX101 %IX106	6	%IX101 %IX106	6	%IX101 %IX106	6
Saídas Digitais Próprias	%QX1 %QX6	6	%QX1 %QX6	6	%QX1 %QX6	6
Saídas Digitais do Drive	%QX101 %QX103	3	%QX101 %QX103	3	%QX101 %QX103	3
Entradas Analógicas Próprias	-	-	%IW1	1	%IW1	1
Entradas Analógicas do Drive	%IW101 %IW102	2	%IW101 %IW102	2	%IW101 %IW102	2
Saídas Analógicas Próprias	-	-	%QW1 %QW2	2	-	-
Parâmetros de Macro	%PM0 %PM31	32	%PM0 %PM31	32	%PM0 %PM31	32

Tabela 1.1 - Faixas de endereços no WLP

MANUAL - PROTOCOLO MODBUS-RTU NO CARTÃO PLC1, PLC2 e POS2

1.4.4 Funções dos Marcadores de Sistema

Os marcadores a seguir têm suas funções pré-definidas pelo sistema e devem ser empregados para tais.

Marcador	Função de Leitura	Função de Escrita
%SX0	Retorno Habilitação do Drive	Habilita Drive
%SX1	Entrada Sensor PTC do Motor	-
%SW0	Retorno Velocidade do Drive (rpm)	-
%SW1	Retorno Velocidade do Drive (13/15 bits)	-
%SW2	-	Gera Erro do Usuário
%SW3	Retorno de Erro do Cartão	-
%SW4	-	Comando Lógico do Drive.
%SW5	Retorno do Estado Lógico do Drive	-

Tabela 1.2 - Funções dos marcadores de sistema

1.5 Descrição Detalhada das Funções

Neste item é feita uma descrição detalhada das funções disponíveis nos cartões para comunicação Modbus-RTU. Para a elaboração dos telegramas, é importante observar o seguinte:

- ☑ Os valores são sempre transmitidos em hexadecimal.
- ☑ O endereço de um dado, o número de dados e o valor dos registradores são sempre representados em 16 bits. Por isso, é necessário transmitir estes campos utilizando dois bytes (high e low). Para acessar bits, a forma para representar um bit depende da função utilizada.
- ☑ Os telegramas, tanto para pergunta quanto para resposta, não podem ultrapassar 128 bytes.

1.5.1 Função 01 - Read Coils

Lê o conteúdo de um grupo de bits (marcadores de bit, marcador de sistema ou saídas digitais do cartão ou do drive) que necessariamente devem estar em sequência numérica. Esta função possui a seguinte estrutura para os telegramas de leitura e resposta (os valores são sempre hexadecimal, e cada campo representa um byte):

Pergunta (Mestre)	Resposta (Escravo)
Endereço do escravo	Endereço do escravo
Função	Função
Endereço do bit inicial (byte high)	Campo Byte Count (no. De bytes de dados)
Endereço do bit inicial (byte low)	Byte 1
Número de bits (byte high)	Byte 2
Número de bits (byte low)	Byte 3
CRC-	etc...
CRC+	CRC-
	CRC+

Cada bit da resposta é colocado em uma posição dos bytes de dados enviados pelo escravo. O primeiro byte, nos bits de 0 a 7, recebe os 8 primeiros bits a partir do endereço inicial indicado pelo mestre. Os demais bytes (caso o número de bits de leitura seja maior que 8), continuam a sequência. Caso o número de bits lidos não seja múltiplo de 8, os bits restantes do último byte devem ser preenchidos com 0 (zero).

Exemplo: leitura das saídas digitais, DO1 a DO6 no endereço 1:

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	01h	Endereço do escravo	01h
Função	01h	Função	01h
Bit inicial (high)	00h	Byte Count	01h
Bit inicial (low)	01h	Estado das saídas 1 a 6	35h
No. De bits (high)	00h	CRC-	91h
No. De bits (low)	06h	CRC+	9Fh
CRC-	Edh		
CRC+	C8h		

Observação: não esquecer que as saídas digitais do drive (RL1, RL2 e RL3) são representadas do cartão como DO101, DO102 e DO103, respectivamente.

1.5.2 Função 02 - Read Inputs Status

Lê o conteúdo de um grupo de entradas digitais do cartão e do drive, que necessariamente devem estar em sequência numérica. Esta função possui a seguinte estrutura para os telegramas de leitura e resposta (os valores são sempre hexadecimal, e cada campo representa um byte):

Pergunta (Mestre)	Resposta (Escravo)
Endereço do escravo	Endereço do escravo
Função	Função
Endereço do bit inicial (byte high)	Campo Byte Count (no. De bytes de dados)
Endereço do bit inicial (byte low)	Byte 1
Número de bits (byte high)	Byte 2
Número de bits (byte low)	Byte 3
CRC-	etc...
CRC+	CRC-
	CRC+

Cada bit da resposta é colocado em uma posição dos bytes de dados enviados pelo escravo. O primeiro byte, nos bits de 0 a 7, recebe os 8 primeiros bits a partir do endereço inicial indicado pelo mestre. Os demais bytes (caso o número de bits de leitura for maior que 8), continuam a sequência. Caso o número de bits lidos não seja múltiplo de 8, os bits restantes do último byte devem ser preenchidos com 0 (zero).

Exemplo: leitura das entradas digitais, DI2 a DI7 no endereço 1:

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	01h	Endereço do escravo	01h
Função	02h	Função	02h
Bit inicial (high)	00h	Byte Count	01h
Bit inicial (low)	02h	Estado das entradas 2 a 7	21h
No. De bits (high)	00h	CRC-	61h
No. De bits (low)	06h	CRC+	90h
CRC-	59h		
CRC+	C8h		

No exemplo, como o número de bits lidos é menor que 8, o escravo precisou de apenas 1 byte para a resposta. O valor do byte foi 21h, que em binário tem a forma 0010 0001. Como o número de bits lidos é igual a 6, somente nos interessa os seis bits menos significativos, que possuem os valores das entradas digitais de 2 a 7. Os demais bits, como não foram solicitados, são preenchidos com 0 (zero).

Observação: não esquecer que as entradas digitais do drive (DI1...DI6) são representadas do cartão como DI101...DI106, respectivamente.

1.5.3 Função 03 - Read Holding Register

Lê o conteúdo de um grupo de marcadores word e float ou parâmetros do cartão ou do drive, que necessariamente devem estar em sequência numérica. Esta função possui a seguinte estrutura para os telegramas de leitura e resposta (os valores são sempre hexadecimal, e cada campo representa um byte):

Pergunta (Mestre)		Resposta (Escravo)	
Endereço do escravo		Endereço do escravo	
Função		Função	
Endereço do registrador inicial (byte high)		Campo Byte Count	
Endereço do registrador inicial (byte low)		Dado 1 (high)	
Número de registradores (byte high)		Dado 1 (low)	
Número de registradores (byte low)		Dado 2 (high)	
CRC-		Dado 2 (low)	
CRC+		etc...	
		CRC-	
		CRC+	

Exemplo: leitura dos valores de velocidade (P002) e corrente do motor (P003) do drive no endereço 1:

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	01h	Endereço do escravo	01h
Função	03h	Função	03h
Registrador inicial (high)	00h	Byte Count	04h
Registrador inicial (low)	02h	P002 (high)	03h
No. De registradores (high)	00h	P002 (low)	84h
No. De registradores (low)	02h	P003 (high)	00h
CRC-	65h	P003 (low)	35h
CRC+	CBh	CRC-	7Ah
		CRC+	49h

Observação importante sobre marcadores tipo FLOAT e parâmetros da macro:

Como um marcador float parâmetros da macro tem 4 bytes, o mestre deverá requisitar dois registros para ler um float, por exemplo:

Para ler o MF9000, o endereço é 9000 e a quantidade deve ser 2, ou seja, será retornado 4 bytes (2 words), que representam o marcador float no formato IEEE 754 (IEEE Standard Floating Point Format).

Se for pedida uma quantidade ímpar de registros, será retornado erro 2.

1.5.4 Função 04 - Read Input Register

Lê o conteúdo das entradas analógicas do cartão e do drive. Esta função possui a seguinte estrutura para os telegramas de leitura e resposta (os valores são sempre hexadecimal, e cada campo representa um byte):

Pergunta (Mestre)	Resposta (Escravo)
Endereço do escravo	Endereço do escravo
Função	Função
Endereço do registrador inicial (byte high)	Campo Byte Count
Endereço do registrador inicial (byte low)	Dado 1 (high)
Número de registradores (byte high)	Dado 1 (low)
Número de registradores (byte low)	Dado 2 (high)
CRC-	Dado 2 (low)
CRC+	Etc...
	CRC-
	CRC+

Exemplo: leitura das entradas analógicas 101 e 102 do drive no endereço 1:

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	01h	Endereço do escravo	01h
Função	04h	Função	04h
Registrador inicial (high)	00h	Byte Count	04h
Registrador inicial (low)	65h	AI101 (high)	1Fh
No. De registradores (high)	00h	AI101 (low)	A0h
No. De registradores (low)	02h	AI102 (high)	0Dh
CRC-	61h	AI102 (low)	20h
CRC+	D4h	CRC-	F9h
		CRC+	3 ^A h

Cada registrador sempre é formado por dois bytes (high e low). Para o exemplo, temos que AI101 = 1FA0h, que em decimal é igual a 8096, e AI102 = 0D20h = 3360. Como as entradas analógicas variam entre 0 e 32767, essa leitura representa respectivamente, 24,7% e 10,25% do valor de fundo de escala das entradas.

Observação: não esquecer que as entradas analógicas do drive (AI1 e AI2) são representadas do cartão como AI101 e AI102, respectivamente.

**1.5.5 Função 05 - Write
Single Coil**

Esta função é utilizada para escrever um valor em um marcador de bit, marcador de sistema ou saída digital. O valor do bit é representado utilizando dois bytes, onde o valor FF00h representa o bit igual a 1, e o valor 0000h representa o bit igual a 0 (zero). Possui a seguinte estrutura (os valores são sempre hexadecimal, e cada campo representa um byte):

Pergunta (Mestre)	Resposta (Escravo)
Endereço do escravo	Endereço do escravo
Função	Função
Endereço do bit (byte high)	Endereço do bit (byte high)
Endereço do bit (byte low)	Endereço do bit (byte low)
Valor para o bit (byte high)	Valor para o bit (byte high)
Valor para o bit (byte low)	Valor para o bit (byte low)
CRC-	CRC-
CRC+	CRC+

Exemplo: acionar a saída digital 2 do cartão no endereço 1:

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	01h	Endereço do escravo	01h
Função	05h	Função	05h
No. Do bit (high)	00h	No. Do bit (high)	00h
No. Do bit (low)	02h	No. Do bit (low)	02h
Valor para o bit (high)	FFh	Valor para o bit (high)	FFh
Valor para o bit (low)	00h	Valor para o bit (low)	00h
CRC-	2Dh	CRC-	2Dh
CRC+	Fah	CRC+	Fah

Para esta função a resposta do escravo é uma cópia idêntica da solicitação feita pelo mestre.

**1.5.6 Função 06 - Write
Single Register**

Esta função é utilizada para escrever um valor em um marcador word, parâmetro do cartão ou do drive, saídas analógicas do cartão e do drive. Não pode ser utilizada para escrita em marcador float. Possui a seguinte estrutura (os valores são sempre hexadecimal, e cada campo representa um byte):

Pergunta (Mestre)	Resposta (Escravo)
Endereço do escravo	Endereço do escravo
Função	Função
Endereço do parâmetro (byte high)	Endereço do parâmetro (byte high)
Endereço do parâmetro (byte low)	Endereço do parâmetro (byte low)
Valor para o parâmetro (byte high)	Valor para o parâmetro (byte high)
Valor para o parâmetro (byte low)	Valor para o parâmetro (byte low)
CRC-	CRC-
CRC+	CRC+

MANUAL - PROTOCOLO MODBUS-RTU NO CARTÃO PLC1, PLC2 e POS2

Exemplo: escrita da referência de velocidade igual a 900 rpm, em um parâmetro do usuário (P800) no endereço 1.

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	01h	Endereço do escravo	01h
Função	06h	Função	06h
Parâmetro (high)	03h	Parâmetro (high)	03h
Parâmetro (low)	20h	Parâmetro (low)	20h
Valor (high)	03h	Valor (high)	03h
Valor (low)	84h	Valor (low)	84h
CRC-	88h	CRC-	88h
CRC+	D7h	CRC+	D7h

Para esta função a resposta do escravo é uma cópia idêntica da solicitação feita pelo mestre. Os marcadores word ou parâmetros são endereçados diretamente pelo seu número, no exemplo acima P800 = 0320h.

1.5.7 Função 15 - Write Multiple Coils

Esta função permite escrever valores para um grupo de marcadores bit ou saídas digitais do cartão ou do drive, que devem estar em sequência numérica. Também pode ser usada para escrever em um único bit (os valores são sempre hexadecimal, e cada campo representa um byte).

Pergunta (Mestre)	Resposta (Escravo)
Endereço do escravo	Endereço do escravo
Função	Função
Endereço do bit inicial (byte high)	Endereço do bit inicial (byte high)
Endereço do bit inicial (byte low)	Endereço do bit inicial (byte low)
Número de bits (byte high)	Número de bits (byte high)
Número de bits (byte low)	Número de bits (byte low)
Campo Byte Count (no. de bytes de dados)	CRC-
Byte 1	CRC+
Byte 2	
Byte 3	
etc...	
CRC-	
CRC+	

O valor de cada bit que está sendo escrito é colocado em uma posição dos bytes de dados enviados pelo mestre. O primeiro byte, nos bits de 0 a 7, recebe os 8 primeiros bits a partir do endereço inicial indicado pelo mestre. Os demais bytes (se o número de bits escritos for maior que 8), continuam em sequência.

Caso o número de bits escritos não seja múltiplo de 8, os bits restantes do último byte devem ser preenchidos com 0 (zero).

Exemplo: ligar as saídas digitais 4 e 5 do cartão , no endereço 1:

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	01h	Endereço do escravo	01h
Função	0Fh	Função	0Fh
Bit inicial (byte high)	00h	Bit inicial (byte high)	00h
Bit inicial (byte low)	04h	Bit inicial (byte low)	04h
No. de bits (byte high)	00h	No. de bits (byte high)	00h
No. de bits (byte low)	02h	No. de bits (byte low)	02h
Byte Count	01h	CRC-	95h
Valor para os bits	03h	CRC+	CBh
CRC-	6Fh		
CRC+	56h		

Como estão sendo escritos apenas dois bits, o mestre precisou de apenas 1 byte para transmitir os dados. Os valores transmitidos estão nos dois bits menos significativos do byte que contém o valor para os bits. Os demais bits deste byte foram deixados com o valor 0 (zero).

1.5.8 Função 16 - Write Multiple Registers

Esta função permite escrever valores para um grupo de marcadores word, marcadores float, parâmetros do cartão ou do drive, saídas analógicas do cartão ou do drive, que devem estar em sequência numérica. Também pode ser usado para escrever um único parâmetro (os valores são sempre hexadecimal, e cada campo representa um byte).

Pergunta (Mestre)		Resposta (Escravo)	
Endereço do escravo		Endereço do escravo	
Função		Função	
Endereço do parâmetro inicial (byte high)		Endereço do parâmetro inicial (byte high)	
Endereço do parâmetro inicial (byte low)		Endereço do parâmetro inicial (byte low)	
Número de parâmetros (byte high)		Número de parâmetros (byte high)	
Número de parâmetros (byte low)		Número de parâmetros (byte low)	
Campo Byte Count (nº de bytes de dados)		CRC-	
Dado 1 (high)		CRC+	
Dado 1 (low)			
Dado 2 (high)			
Dado 2 (low)			
etc...			
CRC-			
CRC+			

Exemplo: escrita do tempo de aceleração (P100) = 1,0 s e tempo de desaceleração (P101) = 2,0 s, do drive no endereço 20:

Pergunta (Mestre)		Resposta (Escravo)	
<i>Campo</i>	<i>Valor</i>	<i>Campo</i>	<i>Valor</i>
Endereço do escravo	14h	Endereço do escravo	14h
Função	10h	Função	10h
Registrador inicial (high)	00h	Registrador inicial (high)	00h
Registrador inicial (low)	64h	Registrador inicial (low)	64h
No. De registradores (high)	00h	No. De registradores (high)	00h
No. De registradores (low)	02h	No. De registradores (low)	02h
Byte Count	04h	CRC-	02h
P100 (high)	00h	CRC+	D2h
P100 (low)	0Ah		
P101 (high)	00h		
P101 (low)	14h		
CRC-	91h		
CRC+	75h		

Observação importante sobre marcadores tipo FLOAT e parâmetros da macro:

Como um marcador float e parâmetros da macro tem 4 bytes, o mestre deverá enviar dois registros para escrever em um float, por exemplo:

Para escrever no MF9000, o endereço é 9000 e a quantidade deve ser 2, ou seja, serão enviados 4 bytes (2 words), que representam o marcador float no formato IEEE 754 (IEEE Standard Floating Point Format).

Se for enviada uma quantidade ímpar de registros, será retornado erro 2.

Como ambos os parâmetro possuem resolução de uma casa decimal, para escrita de 1,0 e 2,0 segundos, devem ser transmitidos respectivamente os valores 10 (000Ah) e 20 (0014h). Função auxiliar, que permite a leitura do fabricante, modelo e versão de firmware do produto. Possui a seguinte estrutura:

**1.5.9 Função 43 - Read
Device Identification**

Pergunta (Mestre)		Resposta (Escravo)	
Endereço do escravo		Endereço do escravo	
Função		Função	
MEI Type		MEI Type	
Código de leitura		Conformity Level	
Número do Objeto		More Follows	
CRC-		Próximo Objeto	
CRC+		Número de objetos	
		Código do Objeto*	
		Tamanho do Objeto*	
		Valor do Objeto*	
		CRC-	
		CRC+	

* Campos são repetidos de acordo com o número de objetos. Esta função permite a leitura de três categorias de informações: Básicas, Regular e Estendida, e cada categoria é formada por um grupo de objetos. Cada objeto é formado por uma sequência de caracteres ASCII. Para o cartão, apenas informações básicas estão disponíveis, formadas por três objetos:

Objeto 00 - VendorName: Sempre 'WEG'.

Objeto 01 - ProductCode: Formado pelo código do produto (PLC1.01, PLC2.00 ou POS2.00) onde .XX indica a versão de hardware.

Objeto 02 - MajorMinorRevision: indica a versão de firmware do cartão, no formato 'VX.XX'. O código de leitura indica quais as categorias de informações estão sendo lidas, e se os objetos estão sendo acessados em sequência ou individualmente. No caso, do cartão suportam os códigos 01 (informações básicas em sequência), e 04 (acesso individual aos objetos).

Exemplo: leitura das informações básicas em sequência, a partir do objeto 00, do cartão no endereço 1:

1.6 Erro de Comunicação

Os erros podem ocorrer na transmissão dos telegramas na rede, ou então no conteúdo dos telegramas recebido. De acordo com o tipo de erro, o cartão poderá ou não enviar resposta para o mestre:

Quando o mestre envia uma mensagem para a placa configurada em um determinado endereço da rede, a mesma não irá responder ao mestre caso ocorra:

- ☒ Erro no CRC.
- ☒ Time out entre os bytes transmitidos (3,5 vezes o tempo de transmissão de uma palavra de 11 bits).

No caso de uma recepção com sucesso, durante o tratamento do telegrama, a POS2 pode detectar problemas e enviar uma mensagem de erro, indicando o tipo de problema encontrado:

- ☒ Função inválida (código do erro = 1): a função solicitada não está implementada para a POS2.
- ☒ Endereço de dado inválido (código do erro = 2): o endereço do dado (parâmetro ou E/S digital) não existe.
- ☒ Valor de dado inválido (código do erro = 3): ocorre nas seguintes situações:
 - ☒ Valor está fora da faixa permitida.
 - ☒ Escrita em dado que não pode ser alterado (registrador somente leitura, registrador que não permite alteração com o drive habilitado ou bits do estado lógico).
 - ☒ Escrita em função do comando lógico que não está habilitada via serial.

1.6.1 Mensagens de Erro

Quando ocorre algum erro no conteúdo da mensagem (não na transmissão de dados), o escravo deve retornar uma mensagem que indica o tipo de erro ocorrido. Os erros que podem ocorrer no tratamento de mensagens para o cartão são os erros de função inválida (código 01), endereço de dado inválido (código 02) e valor de dado inválido (código 03).

As mensagens de erro enviadas pelo escravo possuem a seguinte estrutura:

Resposta (Escravo)
Endereço do escravo
Código da função (com o bit mais significativo em 1)
Código do erro
CRC-
CRC+