DRIVER MODICON MODBUS PARA ELIPSE

*elipse*software

Nome do arquivo:	ModiconModbus.dll
Fabricante:	Modicon
Protocolo:	Modbus ASC/RTU/TCP
Última versão:	

Introdução

Esse driver implementa o protocolo Modbus Master/Slave, permitindo ao Elipse atuar como Mestre em uma rede Modbus, comunicando-se com qualquer equipamento escravo que implemente os protocolos Modbus modos ASCII ou RTU, ou Modbus TCP.

O protocolo MODBUS é um protocolo baseado em mensagens, posicionado no nível 7 do modelo OSI, que possibilita comunicação cliente/servidor entre equipamentos conectados a diferentes tipos de redes. É um protocolo baseado em mensagens de comando e resposta, oferecendo serviços com funções definidas por um código de 8 bits.

Existem três categorias de códigos de funções:

- **Códigos de Funções Públicas**: Funções bem definidas pelo protocolo, com garantia de unicidade, validadas pela comunidade Modbus.org e publicamente documentadas em MB IETF RFC.

- **Códigos de Funções Definidas pelo Usuário**: Funções definidos pelo usuário não são padronizados e não precisam de aprovação pela Modbus.org, não tendo portando qualquer garantia de unicidade, podendo ser livremente implementadas.

- **Códigos de Funções reservadas**: Códigos atualmente usados por alguns fabricantes em produtos antigos, não estando disponíveis para uso público.

Esse driver implementa todas as funções públicas com exceção das funções 22, 23 e 43, bem como algumas funções específicas de fabricantes.

O protocolo Modbus foi desenvolvido inicialmente pela Modicon (atual Schneider Electric), em 1979, sendo hoje um padrão aberto, implementado por centenas de fabricantes em milhares de equipamentos.

Maiores informações referentes ao protocolo Modbus podem ser obtidas no site **www.modbus.org.**

Configurando o equipamento

Parâmetros [P] de configuração do driver

Esse driver não utiliza os parâmetros P´s. Todas as configurações de comunicação devem ser realizadas nas janelas de configurações extras do IOKit.

As configurações específicas do protocolo devem ser realizadas na aba 'Modbus', dentro da janela de configurações extras (**Parâmetros de Configurações Extras do Driver**).

Observações para Comunicação Ethernet: A porta padrão para comunicação com MODBUS/TCP é 502.

Parâmetros [N] de endereçamento de tags PLC

N1 Endereço da CPU (0= Modo Broadcast)

N2 Número da função criada em "Extras" para leitura e escrita

N3 Não utilizado

N4 Endereço da variável

Se forem utilizadas as funções 20 ou 21, N3 será utilizado para informar o número do arquivo.

Parâmetros [B] de endereçamento de tags bloco

B1 Endereço da CPU (0= Modo Broadcast)

B2 Número da função criada em "Extras" para leitura e escrita

B3 Não utilizado

B4 Endereço da variável

Se forem utilizadas as funções 20 ou 21, B3 será utilizado para informar o número do arquivo.

Tags especiais para leitura do código da última exceção

B1 Endereço da CPU

B2 99999 (ver Tabela1)

Tabela1 - Parâmetros B2

Elemento	Descrição
Elemento[0]	Código da exceção
Elemento[1]	Parâmetro N2/B2
Elemento[2]	Parâmetro N3/B3
Elemento[4]	Parâmetro N4/B4

Esse tag lê ou escreve em um registrador interno do driver, o qual registra o último código de exceção enviado por um determinado equipamento escravo. A cada comunicação bem sucedida com esse equipamento que não retorne exceção, o driver automaticamente atribuirá zero a esse registrador. Como esse tag aceita escrita, o usuário tem a possibilidade de zerá-lo manualmente, por script.

Os elementos 1, 2 e 3 especificam os parâmetros do tag do driver que gerou a exceção registrada. Caso tais informações não sejam necessárias, pode-se alternativamente acessar o código da última exceção com um tag PLC análogo, com N1=endereço e N2=99999.

O timestamp retornado por esse tag representa o momento em que a exceção foi recebida do equipamento.

O item "**Uso dos tags especiais para leitura do código da última exceção**" deste manual fornece algumas dicas de como utilizar adequadamente este recurso.

Parâmetros de Configurações Extras do Driver

Através da aba Modbus da janela de configurações extras, é possível configurar parâmetros adicionais do driver.

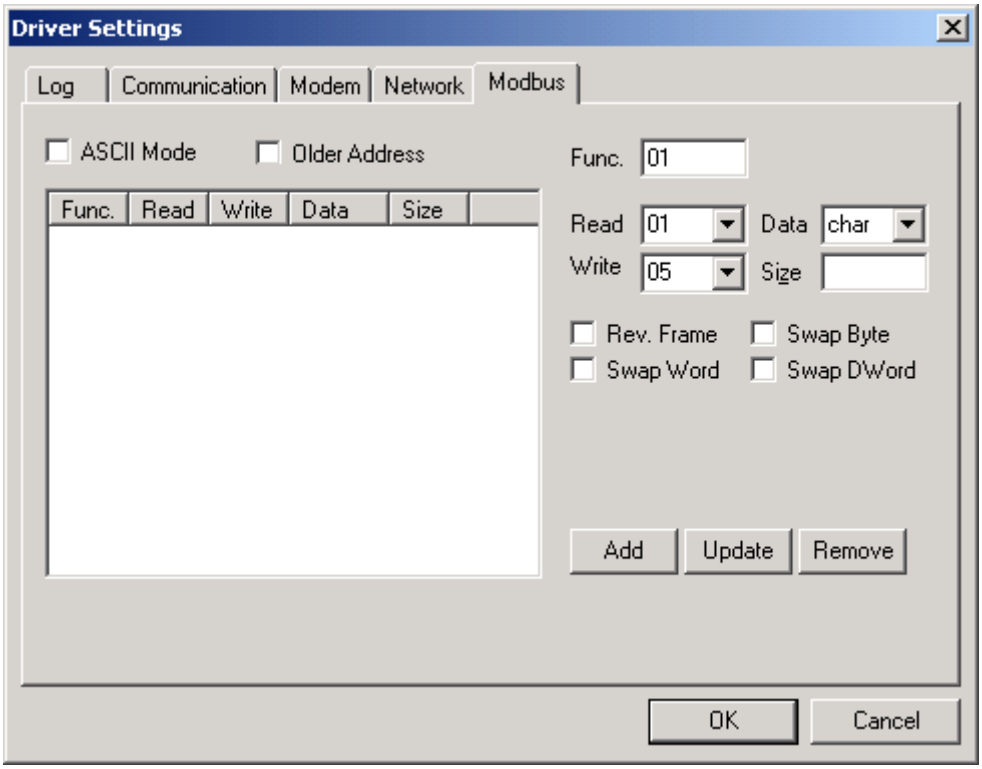

Figura 1: Janela das configurações extras do driver

Modbus functions

Aqui são especificadas as funções Modbus que serão utilizadas nos parâmetros N2/B2 dos tags. A cada código de função devem ser especificados uma função Modbus para Leitura e outra para Escrita, além do tipo de dado que será manipulado.

As funções Modbus implementas são as seguintes:

Leitura

Leitura

Opção	Descrição
01	Leitura de Bit (Read Coil Status - 0x)
02	Leitura de Bit (Read Input Status - 1x)
03	Leitura de Words (Read Holding Registers - 4x)
04	Leitura de Words (Read Input Registers - 3x)
07	Leitura de Status (Read Exception Status)
20	Leitura da Memória Extendida (Read General Reference - 6x)

Funções Especiais

Opção	Descrição
6503	Leitura da Memória de Massa (somente para o equipamento ABB MGE 144)

Escrita

Escrita

Opção	Descrição
05	Escrita de Bit (Force Single Coil - 0x)
06	Escrita de Word Simples (Preset Single Register - 4x)
15	Escrita de Bits (Force Multiple Coils - 0x)
16	Escrita de Words (Preset Multiple Registers - 4x)
21	Escrita na Memória Extendida (Write General Reference - 6x)

Funções Especiais

Opção	Descrição
65 01	65 01 - Reseta medidor de energia (somente para o equipamento ABB MGE 144)
65 02	65 02 - Reseta memória de máximo e mínimo (somente para o equipamento ABB MGE 144)

Tipos de Dados

Opção	Descrição
Char	Palavra de 8 bits, caracter.
Byte	Palavra de 8 bits sem sinal.
Int8	Palavra de 8 bits com sinal.
Int16	Palavra de 16 bits com sinal.
Int32	Palavra de 32 bits com sinal.
Word	Palavra de 16 bits sem sinal.
Dword	Palavra de 32 bits sem sinal.
Float	Ponto Flutuante de 32 bits (IEEE 754) (4 bytes na ordem: EXP F2 F1 0).
Double	Real de 64 bits.
String	Palavra de N chars (texto). O tipo string nesse driver não possui um limite máximo de tamanho definido, sendo esse limite o limite máximo permitido pelo protocolo para a área de dados do frame de uma determinada função.
BCD	Valor numérico BCD (decimal codificado em binário). Quando utilizando esse tipo, a aplicação deve fornecer um valor decimal positivo e inteiro, a ser enviado no formato BCD, respeitando o tamanho especificado. O campo 'size', no caso do tipo BCD, refere-se ao número de bytes a serem enviados para representar o valor. Uma vez que na codificação BCD cada algarismo será convertido em um nibble, temos que os valores permitidos devem possuir um número máximo de algarismos igual ao dobro do valor especificado no campo 'size'. Ou seja, se for selecionado 2 para o campo size, o máximo valor que poderá ser enviado será 99999. Já se size=4, o valor máximo será 9999999999. Os valores permitidos para o campo size no caso de tipos BCD são 2 (WORD) e 4 (DOUBLE WORD). Para maiores detalhes sobre a codificação BCD, consulte o item 6.1 deste manual.

As demais opções disponíveis na janela das configurações extras do driver são as seguintes:

Size: Deve ser informado o tamanho em bytes de cada elemento do tipo de dado selecionado. Esse campo é preenchido automaticamente para tipos de dados com tamanho fixo, como os tipos BYTE, WORD e int16, devendo ser preenchido para dados tipo string e BCD.

Rev Frame: Indica que o sentido dos bytes no frame está invertido.

Swap Byte: Indica que o driver deverá inverter a ordem dos bytes um a um para obter o valor.

Swap Word: Indica que o driver deverá inverter a ordem dos bytes dois a dois (em words) para obter o valor.

Swap Dword: Indica que o driver deverá inverter a ordem dos bytes quatro a quatro (em dwords) para obter o valor.

Import Configuration: Esta opção permite importar configurações de funções de versões anteriores a 2.0 do driver Modbus Master/Slave, que armazenavam essas configurações em um arquivo 'modbus.ini'. Esse driver não utiliza mais arquivos .ini para armazenar tais configurações, as quais são agora armazenadas no próprio arquivo da aplicação.

Export Configuration: Esta opção faz a operação inversa da anterior, gerando um arquivo .ini contendo as configurações de funções, no mesmo formato das versões anteriores desse driver. Dessa forma é possível guardar-se em um arquivo as configurações de funções, que podem vir a serem usadas em outras aplicações.

Os botões Add/Update/Remove definem a lista de funções Modbus que serão utilizadas no sistema.

- **Botão Add**: Adiciona um novo item a lista.
- **Botão Update**: Atualiza um item selecionado na lista.
- **Botão Remove**: Remove um item selecionado na lista.

Protocol options

Protocol Options

Opção	Descrição
Older Address	Habilita informar, de modo direto, o valor da posição de memória (automaticamente o driver fará a subtração por 1 do valor informado, antes de enviar o frame de comunicação).
Modbus mode	Nessa caixa de combinação é possível selecionar o modo a ser utilizado. São três as opções disponíveis: modo RTU (default), modo ASCII, ou ModbusTCP.

Notas

Codificação BCD

A codificação BCD (Binary Coded Decimal ou Decimal Codificado em Binário) foi originalmente concebida para contornar limitações quanto ao número máximo de dígitos passíveis de serem representados nos formatos mais tradicionais de armazenamento de valores. Formatos como a representação de números reais em ponto flutuante mostram-se normalmente aceitáveis para cálculos matemáticos e científicos. Porém, erros de aproximação causados pela existência de algarismos que não possam ser representados por problemas de overflow ou underflow, podem não ser admissíveis em certas aplicações, como em procedimentos financeiros. Para superar esse tipo de limitação foi desenvolvida a codificação BCD, a qual permite a representação de números até o último algarismo.

Nessa representação, cada algarismo decimal é representado em binário de per si, sem limitações no que se refere ao número de algarismos.

A tabela abaixo mostra os algarismos decimais e seus valores correspondentes em BCD:

Algorítmo decimais e seus valores correspondentes em BCD

DECIMAL	BCD	DECIMAL	BCD
0	0000b	5	0101b
1	0001b	6	0110b
2	0010b	7	0111b
3	0011b	8	1000b
4	0100b	9	1001b

Tabela1: Códigos BCD dos algarismos decimais de 0 a 9

A fim de melhorar a eficiência dessa codificação, é comum representar-se dois algarismos por byte, já que cada algarismo decimal requer apenas 4 bits para sua codificação. Tal representação é chamada de BCD comprimido (packed BCD), e é a representação utilizada por este driver. Ou seja, os pacotes enviados por esse driver com valores BCD utilizam um byte de dado para cada dois algarismos do valor decimal representado. Por isso o campo size, no caso de tipos de dado BCD, deve ser definido como a metade do número máximo de algarismos a serem representados nos valores a serem lidos ou escritos.

Uso dos tags especiais para leitura do código da última exceção

Conforme já mencionado neste manual, os tags especiais para leitura do código da última exceção são utilizados para ler o último código de exceção enviado por um determinado equipamento escravo.

Como já mencionado, tais códigos são armazenados automaticamente pelo driver em registradores internos, que podem ser acessados por meio deste tag. Além disso, a cada comunicação bem sucedida com determinado equipamento, em que nenhuma exceção for retornada, o driver zera automaticamente o registrador associado ao mesmo.

Os códigos de exceção são usados pelo escravo para informar uma falha ao executar uma determinada função. Os equipamentos escravos não retorna exceções no caso de falhas de comunicação, situação em que os mesmos simplesmente não respondem. Os códigos de exceção são retornados pelos escravos em situações em que a solicitação do mestre foi recebida com sucesso, porém não pôde ser executada por algum motivo, como por exemplo a tentativa de ler ou escrever em um registrador inexistente. Neste caso o código de exceção retornado indica o tipo de erro ocorrido.

A especificação do protocolo Modbus define 9 códigos de exceção, os quais são apresentados na tabela 2 deste manual. Além desses códigos, alguns fabricantes definem códigos adicionais, específicos de seus equipamentos.

A forma mais usual de utilizar este tag, durante o scan normal dos tags de funções, é através de um evento OnRead do tag de exceção. Nesse caso o script deve antes de tudo rejeitar valores nulos, pois esses indicam o não recebimento de exceções. Em seguida pode-se tratar a exceção, executando os procedimentos adequados, conforme o código recebido. Constitui-se em uma boa prática zerar o registrador de exceção ao sair do script, de forma a indicar que a exceção já foi tratada. Veja o exemplo abaixo, em Elipse Basic (SCADA):

```
// EVENTO OnRead DA TAGEXC
// Obs: Para esse exemplo, considere TAGEXC com leitura e escrita automatica
habilitadas
IF TAGEXC==0
  EXIT
ENDIF
IF TAGEXC==1
  ...    // TRATA EXCECAO 1
ELSEIF TAGEXC==2
 ...     // TRATA EXCECAO 2
ELSE
 ...     // TRATA DEMAIS EXCECOES
ENDIF
TAGEXC=0 // ZERA REGISTRADOR DE EXCECOES
```

Já nas operações de escrita por script, em que se precise testar o retorno de exceções logo em seguida ao envio do comando, deve-se primeiramente zerar o registrador de exceções. Isto evita que uma eventual exceção provocada pelo comando de escrita seja confundida com uma preexistente, que possua o mesmo código. Executa-se então a operação de escrita e testa-se o valor do tag especial, que deve retornar zero caso nenhuma exceção tenha sido recebida. Caso o mesmo retorne um valor diferente de zero, pode-se então tratar apropriadamente a exceção recebida. Veja o exemplo abaixo, em Elipse Basic (SCADA):

```
// Obs: Para esse exemplo, considere TAGEXC com leitura e escrita automatica
habilitadas,
//          e TAG com escrita automatica desabilitada.
TAGEXC=0 // ZERA REGISTRADOR DE EXCECOES
TAG = 10
TAG.WRITE()
IF TAGEXC<>0
  ...    // TRATA EXCECAO
ENDIF
```

OBS: O tag especial B2=99999 retorna, além do código da exceção, retornado no elemento zero, também os parâmetros do tag cuja comunicação teria provocado a exceção. Caso essas informações não sejam necessárias, pode-se perfeitamente ler o mesmo registro através de um tag PLC com N2=9999. Nesse caso os procedimentos recomendados permanecem os mesmos.

Códigos de exceção

CÓDIGO	NOME	SIGNIFICADO
1	ILLEGAL FUNCTION	O código de função recebido não é válido. Isso pode indicar que a função não está implementa, ou que o escravo encontra-se em um estado inadequado para processá-la.
2	ILLEGAL DATA ADDRESS	Endereço de dados recebido não é um endereço válido. Mais especificamente, a combinação do endereço de referência e a quantidade de dados a serem transferidos é inválida.
3	ILLEGAL DATA VALUE	Valor presente na requisição do Mestre não é válido. Isto indica uma falha na estrutura de dados remanescente de uma requisição complexa, como quando o tamanho informado para o bloco de dados não está correto. Esta exceção não indica que os valores submetidos para escrita estejam fora do escopo esperado pela aplicação, uma vez que tal informação não é acessível ao protocolo.
4	SLAVE DEVICE FAILURE	Ocorreu um erro irrecuperável durante o processamento da função solicitada.
5	ACKNOWLEDGE	Usado com comandos de programação. O escravo aceitou a mensagem e a está processando. Porém esse processamento leverá um longo tempo. Essa exceção previne um time-out no mestre. O fim da requisição deve ser testado por um processo de *polling*.
6	SLAVE DEVICE BUSY	Usado com comandos de programação. Indica que o escravo está processando um outro comando de longa duração, e que a solicitação deve ser retransmitida mais tarde, quando o escravo estiver novamente disponível.
8	MEMORY PARITY ERROR	Usado em conjunto com as funções 20 e 21, reference type 6, para indicar que a área extendida de arquivos falhou em um teste de consistência. O equipamento escravo pode estar precisando de manutenção.
0A	GATEWAY PATH UNAVAILABLE	Usado em conjunto com gateways, para indicar que o gateway não foi capaz de alocar um caminho interno para o processamento da solicitação. Geralmente indica que o gateway está desconfigurado ou sobrecarregado.
0B	GATEWAY TARGET DEVICE FAILED TO RESPOND	Usado em conjunto com gateways, para indicar que não foi recebida nenhuma resposta do equipamento destino. Geralmente indica que tal equipamento não está presente na rede.

Tabela 2: Códigos de exceção padronizados pelo protocolo Modbus.

Referências

- **MODBUS Application Protocol Specification**, Modbus.org, 2002.05.02
- **IOKit** - User´s Manual - Elipse Software

Histórico das revisões do driver

Versão	Data	Autor	Comentários