CANopen

PLC300

Manual do Usuário

Manual do Usuário CANopen

Série: PLC300

Idioma: Português

N ° do Documento: 10000849433 / 01

Data da Publicação: 11/2010

SUMÁRIO

SOBRE O MANUAL

Este manual fornece a descrição necessária para a operação do controlador programável PLC300 utilizando o protocolo CANopen. Este manual deve ser utilizado em conjunto com manual do usuário do PLC300.

ABREVIAÇÕES E DEFINIÇÕES

CAN	Controller Area Network
CiA	CAN in Automation
COB	Communication Object
COB-ID	Communication Object Identifier
SDO	Service Data Object
PDO	Process Data Object
RPDO	Receive PDO
TPDO	Transmit PDO
NMT	Network Management Object
ro	Read only (somente leitura)
rw	Read/write (leitura e escrita)

REPRESENTAÇÃO NUMÉRICA

Números decimais são representados através de dígitos sem sufixo. Números hexadecimais são representados com a letra 'h' depois do número.

DOCUMENTOS

O protocolo CANopen foi desenvolvido baseado nas seguintes especificações e documentos:

Documento	Versão	Fonte
CAN Specification	2.0	CiA
CiA DS 301 CANopen Application Layer and Communication Profile	4.02	CiA
CiA DRP 303-1 Cabling and Connector Pin Assignment	1.1.1	CiA
CiA DSP 306 Electronic Data Sheet Specification for CANopen	1.1	CiA
CiA DSP 402 Device Profile Drives and Motion Control	2.0	CiA

Para obter esta documentação, deve-se consultar a CiA, que atualmente é a organização que mantém, divulga e atualiza as informações relativas à rede CANopen.

1 INTRODUÇÃO À COMUNICAÇÃO CANOPEN

Para a operação de um equipamento em rede CANopen, é necessário conhecer a forma como a comunicação é feita. Este item traz uma descrição geral do funcionamento do protocolo CANopen, contendo as funções utilizadas pelo PLC300. Para uma descrição mais detalhada pode-se consultar a especificação do protocolo.

1.1 CAN

A rede CANopen é uma rede baseada em CAN, o que significa dizer que ela utiliza telegramas CAN para troca de dados na rede.

O protocolo CAN é um protocolo de comunicação serial que descreve os serviços da camada 2 do modelo ISO/OSI (camada de enlace de dados)[1]. Nesta camada, são definidos os diferentes tipos de telegramas (frames), a forma de detecção de erros, validação e arbitração de mensagens.

1.1.1 Frame de Dados

Os dados em uma rede CAN são transmitidos através de um frame de dados. Este tipo de frame é composto principalmente por um campo identificador de 11 bits[2] (arbitration field), e um campo de dados (data field), que pode conter até 8 bytes de dados.

Identificador	8 bytes de dados							
11 bits	byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7

1.1.2 Frame Remoto

Além do frame de dados, existe também o frame remoto (RTR frame). Este tipo de frame não possui campo de dados, apenas o identificador. Ele funciona como uma requisição para que outro dispositivo da rede transmita o frame de dados desejado.

1.1.3 Acesso à Rede

Em uma rede CAN, qualquer elemento da rede pode tentar transmitir um frame para a rede em um determinado instante. Caso dois elementos tentem acessar a rede ao mesmo tempo, conseguirá transmitir aquele que enviar a mensagem mais prioritária. A prioridade da mensagem é definida pelo identificador do frame CAN, quanto menor o valor deste identificador, maior a prioridade da mensagem. O telegrama com o identificador 0 (zero) corresponde ao telegrama mais prioritário.

1.1.4 Controle de Erros

A especificação CAN define diversos mecanismos para controle de erros, o que a torna uma rede muito confiável e com um índice muito baixo de erros de transmissão que não são detectados. Cada dispositivo da rede deve ser capaz de identificar a ocorrência destes erros, e informar aos demais elementos que um erro foi detectado.

Um dispositivo da rede CAN possui contadores internos que são incrementados toda vez que um erro de transmissão ou recepção é detectado, e decrementado quando um telegrama é enviado ou recebido com sucesso. Cada dispositivo na rede CAN pode ser levado para os seguintes estados, de acordo com a quantidade de erros de transmissão ou recepção detectados:

- *Error Active:* os contadores internos de erro estão em um nível baixo e o dispositivo opera normalmente na rede CAN. Pode enviar e receber telegramas e atuar na rede CAN caso detecte algum erro na transmissão de telegramas.
- *Warning:* quando algum destes contadores passa de um determinado limite, o dispositivo entra no estado de *warning*, significando a ocorrência de uma elevada taxa de erros de comunicação.
- *Error Passive:* quando este valor ultrapassa um limite maior, ele entra no estado de *error passive*, onde ele para de atuar na rede ao detectar que outro dispositivo enviou um telegrama com erro.

[1] Na especificação do protocolo CAN, é referenciada a norma ISO 11898 como definição da camada 1 deste modelo (camada física).
[2] A especificação CAN 2.0 define dois tipos de frames de dados: *standard* (11 bits) e *extended* (29 bits). Para esta implementação, somente frames *standard* são aceitos.

- *Bus Off:* por último, temos o estado de *bus off*, no qual o dispositivo não irá mais enviar ou receber telegramas. O dispositivo opera como se estivesse desconectado da rede.

1.1.5 CAN e CANopen

Somente a definição de como detectar erros, criar e transmitir um frame não são suficientes para definir um significado para os dados que são enviados via rede. É necessário que haja uma especificação que indique como o identificador e os dados devem ser montados e como as informações devem ser trocadas. Desta forma os elementos da rede podem interpretar corretamente os dados que são transmitidos. Neste sentido, a especificação CANopen define justamente como trocar dados entre os equipamentos e como cada dispositivo deve interpretar estes dados.

Existem diversos protocolos baseados em CAN, como DeviceNet, CANopen, J1939, etc., que utilizam frames CAN para a comunicação. Porém estes protocolos não podem operar em conjunto na mesma rede.

1.2 CARACTERÍSTICAS DA REDE CANOPEN

Por utilizar um barramento CAN como forma de transmissão de telegramas, todos os dispositivos da rede CANopen têm os mesmos direitos de acesso à rede, onde a prioridade do identificador é responsável por resolver problemas de conflito quando acessos simultâneos ocorrem. Isto traz o benefício de possibilitar a comunicação direta entre escravos da rede, além do fato de que os dados podem ser disponibilizados de maneira mais otimizada, sem a necessidade de um mestre que controle toda a comunicação fazendo acesso cíclico a todos os dispositivos da rede para atualização dos dados.

Outra característica importante é a utilização do modelo produtor / consumidor para a transmissão de dados. Isto significa dizer que uma mensagem que trafega na rede não possui um endereço fixo na rede como destino. Esta mensagem possui um identificador que indica qual o dado que ela está transportando. Qualquer elemento da rede que necessite utilizar desta informação para a sua lógica de operação, poderá consumi-la e, portanto, uma mesma mensagem pode ser utilizada por vários elementos da rede ao mesmo tempo.

1.3 MEIO FÍSICO

O meio físico para a transmissão de sinais em uma rede CANopen é especificado pela norma ISO 11898. Ela define como barramento de transmissão um par trançado com sinal elétrico diferencial.

1.4 ENDEREÇO NA REDE CANOPEN

Toda a rede CANopen deve possuir um mestre, responsável por serviços de gerenciamento da rede, e também pode possuir um conjunto de até 127 escravos. Cada dispositivo da rede também pode ser chamado de nó. Todo escravo em uma rede CANopen é identificado na rede através de seu endereço, ou Node-ID, que deve ser único para cada escravo da rede, e pode variar de 1 até 127.

Para o controlador programável PLC300, o endereço do escravo é programado através do menu Setup.

1.5 ACESSO AOS DADOS

Cada escravo da rede CANopen possui uma lista, denominada dicionário de objetos, que contém todos os dados que são acessíveis via rede. Cada objeto desta lista é identificado através de um índice, e durante a configuração do equipamento e troca de mensagens, este índice é utilizado para identificar o que está sendo transmitido.

1.6 TRANSMISSÃO DE DADOS

A transmissão de dados numéricos através de telegramas CANopen é feita utilizando a representação hexadecimal do número, e enviando o byte menos significativo do dado primeiro.

Exemplo: transmissão de um inteiro com sinal de 32 bits (12345678h = 305419896 decimal), mais um inteiro com sinal de 16 bits (FF00h = -256 decimal), em um frame CAN.

Identificador	6 bytes de dados					
11 bits	Inteiro 32 bits				Inteiro 16 bits	
	byte 0	byte 1	byte 2	byte 3	byte 4	byte 5
	78h	56h	34h	12h	00h	FFh

1.7 OBJETOS RESPONSÁVEIS PELA COMUNICAÇÃO – COBS

Existe um determinado conjunto de objetos que são responsáveis pela comunicação entre os dispositivos da rede. Estes objetos estão divididos de acordo com os tipos de dados e a forma como são enviados ou recebidos por um dispositivo. Os seguintes objetos de comunicação (COBs) são descritos pela especificação:

Tabela 1.1: Tipos de Objetos de Comunicação (COBs)

Tipo de Objeto	Descrição
Service Data Object (SDO)	Os SDOs são objetos responsáveis pelo acesso direto ao dicionário de objetos de um dispositivo. Através de mensagens utilizando os SDOs, é possível indicar explicitamente (através do índice do objeto), qual o dado que está sendo manipulado. Existem dois tipos de SDOs: Cliente SDO, responsável por fazer uma requisição leitura ou escrita para um dispositivo da rede, e o Servidor SDO, responsável por atender esta requisição. Como os SDOs são utilizados geralmente para configuração de um nó da rede, são menos prioritários que outros tipos de mensagens.
Process Data Object (PDO)	Os PDOs são utilizados para acessar dados do equipamento sem a necessidade de indicar explicitamente qual o objeto do dicionário está sendo acessado. Para isso, é necessário configurar previamente quais os dados que o PDO estará transmitindo (mapeamento dos dados). Também existem dois tipos de PDOs: PDO de recepção e PDO de transmissão. PDOs usualmente são utilizados para transmissão e recepção de dados utilizados durante a operação do dispositivo, e por isso são mais prioritários que os SDOs.
Emergency Object (EMCY)	Este objeto é responsável pelo envio de mensagens para indicar a ocorrência de erros no dispositivo. Quando um erro ocorre em um determinado dispositivo (Produtor EMCY), este pode enviar uma mensagem para a rede. Caso algum dispositivo da rede esteja monitorando esta mensagem (Consumidor EMCY), é possível programar para que uma ação seja tomada (desabilitar demais dispositivos da rede, reset de erros, etc.).
Synchronization Object (SYNC)	Na rede CANopen é possível programar um dispositivo (Produtor SYNC) para enviar, periodicamente, uma mensagem de sincronização para todos os dispositivos da rede. Estes dispositivos (Consumidores SYNC) podem então, por exemplo, enviar um determinado dado que necessita ser disponibilizado periodicamente.
Network Management (NMT)	Toda a rede CANopen precisa ter um mestre que controle os demais dispositivos da rede (escravos). Este mestre será responsável por um conjunto de serviços que controlam a comunicação dos escravos e seu estado na rede CANopen. Os escravos são responsáveis por receber os comandos enviados pelo mestre e executar as ações solicitadas. Dentre os serviços descritos pelo protocolo estão: serviços de controle do dispositivo, onde o mestre controla o estado de cada escravo na rede, e serviços de controle de erros (Node Guarding e Heartbeat), onde o dispositivo envia mensagens periódicas para informar que a conexão está ativa.

Toda a comunicação do escravo com a rede é feita utilizando estes objetos, e os dados que podem ser acessados são os existentes no dicionário de objetos do dispositivo.

1.8 COB-ID

Um telegrama da rede CANopen sempre é transmitido por um objeto de comunicação (COB). Todo COB possui um identificador que indica o tipo de dado que está sendo transportado. Este identificador, chamado de COB-ID, possui um tamanho de 11 bits, e é transmitido no campo identificador de um telegrama CAN. Ele pode ser subdividido em duas partes:

Código da Função				Endereço do nó						
bit 10	bit 9	bit 8	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0

- Código da função: indica o tipo de objeto que está sendo transmitido.
- Endereço do nó: indica com qual dispositivo da rede o telegrama está vinculado.

A seguir é apresentada uma tabela com os valores padrão para os diferentes objetos de comunicação descritos no protocolo. É necessário observar que o valor padrão do objeto depende do endereço do escravo, com exceção dos COB-IDs para NMT e SYNC, que são comuns para todos os elementos da rede. Estes valores também podem ser alterados durante a etapa de configuração do dispositivo.

Tabela 1.2: COB-ID para os diferentes objetos

COB	Código da Função (bits 10 – 7)	COB-ID Resultante (função + endereço)
NMT	0000	0
SYNC	0001	128 (80h)
EMCY	0001	129 – 255 (81h – FFh)
PDO1 (tx)	0011	385 – 511 (181h – 1FFh)
PDO1 (rx)	0100	513 – 639 (201h – 27Fh)
PDO2 (tx)	0101	641 – 767 (281h – 2FFh)
PDO2 (rx)	0110	769 – 895 (301h – 37Fh)
PDO3 (tx)	0111	897 – 1023 (381h – 3FFh)
PDO3 (rx)	1000	1025 – 1151 (401h – 47Fh)
PDO4 (tx)	1001	1153 – 1279 (481h – 4FFh)
PDO4 (rx)	1010	1281 – 1407 (501h – 57Fh)
SDO (tx)	1011	1409 – 1535 (581h – 5FFh)
SDO (rx)	1100	1537 – 1663 (601h – 67Fh)
Node Guarding/ Heartbeat	1110	1793 – 1919 (701h – 77Fh)

1.9 ARQUIVO EDS

Cada dispositivo em uma rede CANopen possui um arquivo de configuração EDS, que contém informações sobre o funcionamento do dispositivo na rede CANopen, bem como a descrição de todos os objetos existentes para comunicação. Em geral este arquivo é utilizado por um mestre ou software de configuração, para programação dos dispositivos presentes na rede CANopen.

O arquivo de configuração EDS é fornecido em um CD juntamente com o produto, e também pode ser obtido através do site http://www.weg.net. É necessário observar a versão de software do equipamento, para utilizar um arquivo EDS que seja compatível com esta versão.

2 INTERFACE DE COMUNICAÇÃO CANOPEN

O controlador programável PLC300 possui por padrão no produto uma interface CAN. Ela pode ser utilizada para comunicação no protocolo CANopen como mestre ou escravo da rede. Características desta interface são descritas a seguir.

2.1 CARACTERÍSTICAS DA INTERFACE CAN

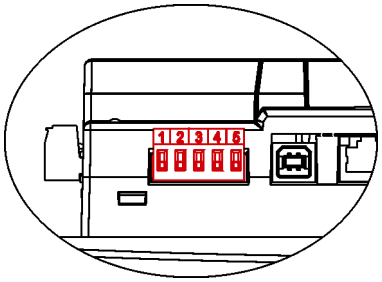


Figura 2.1: Detalhe do conector CAN na parte inferior do produto

- Interface isolada galvanicamente e com sinal diferencial, conferindo maior robustez contra interferência eletromagnética.
- Alimentação externa de 24V.
- Permite a conexão de até 64 dispositivos no mesmo segmento. Uma quantidade maior de dispositivos pode ser conectada e com o uso de repetidores[3].
- Comprimento máximo do barramento de 1000 metros.

2.2 PINAGEM DO CONECTOR

A interface CAN possui um conector *plug-in* de 5 vias (XC6) com a seguinte pinagem:

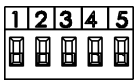


Tabela 2.1: Pinagem do conector XC5 para interface CAN

Pino	Nome	Função
1	V-	Pólo negativo da fonte de alimentação
2	CAN_L	Sinal de comunicação CAN_L
3	Shield	Blindagem do cabo
4	CAN_H	Sinal de comunicação CAN_H
5	V+	Pólo positivo da fonte de alimentação

2.3 FONTE DE ALIMENTAÇÃO

As interfaces CAN necessitam de uma tensão de alimentação externa entre os pinos 1 e 5 do conector da rede. Os dados para consumo individual e tensão de entrada são apresentados na tabela a seguir.

Tabela 2.2: Características da alimentação para interface CAN

Tensão de alimentação (VCC)		
Mínimo	Máximo	Recomendado
11	30	24
Corrente (mA)		
Típico	Máximo	
30	50	

[3] O número limite de equipamentos que podem ser conectados na rede também depende do protocolo utilizado.

2.4 INDICAÇÕES

Além dos marcadores de sistema, que fornecem diversas informações sobre a interface, o controlador programável PLC300 possui um LED bicolor – verde e vermelho – na parte frontal do produto utilizado para indicação da interface CAN.

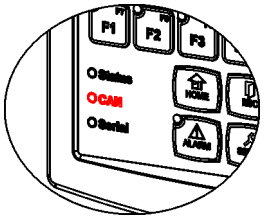


Figura 2.2: LED de indicação da interface CAN

Durante a inicialização do equipamento, ambos os LEDs são acesos para teste por um período de aproximadamente 500 ms alternadamente. Após este período, para o protocolo CANopen, eles farão indicações conforme mostrado a seguir.

2.4.1 Tipos de indicação

Além dos estados aceso e apagado, os seguintes comportamentos também podem ser visualizados:

- Intermitente: o LED fica um período de 200 ms aceso, seguido de um período de 200 ms apagado.
- Uma piscada: o LED fica um período de 200 ms aceso, seguido de um período de 1 segundo apagado.
- Duas piscadas: o LED acende duas vezes por um período de 200 ms (com um período de 200 ms apagado entre estas indicações), seguido de um período de 1 segundo apagado.

2.4.2 Error LED (vermelho)

O LED vermelho indica erros da camada física do barramento CAN, além de erros da comunicação CANopen.

Tabela 2.3: Indicações do LED de erro (vermelho)

Indicação	Estado	Descrição
Apagado	Sem erro	O equipamento está operando normalmente, desligado ou sem alimentação na interface CAN.
Uma piscada	Atingido o estado de warning	Os contadores internos de erro do controlador CAN atingiram o estado de warning, devido a erros da comunicação CAN. Esta indicação também é válida caso o equipamento esteja no estado error passive.
Duas piscadas	Falha no serviço de controle de erros – Node Guarding ou Heartbeat	Após o serviço de Node Guarding ou Heartbeat ter sido inicializa-do pelo mestre, ocorreu timeout na troca de telegra-mas entre o mestre e o escravo, ocasionando este erro.
Aceso	Bus off	O controlador CAN atingiu o estado de bus off.

2.4.3 Run LED (verde)

O LED verde indica o estado do escravo na comunicação CANopen.

Tabela 2.4: Indicações do LED run (verde)

Indicação	Estado	Descrição
Uma piscada	Estado parado (STOPPED)	O equipamento encontra-se no estado parado.
Intermitente	Pré-operacional (PRE-OPERATIONAL)	O equipamento encontra-se no estado pré-operacional.
Aceso	Operacional (OPERATIONAL)	O equipamento encontra-se no estado operacional.

NOTA!
Caso o estado da interface CAN e comunicação CANopen seja tal que ambos os LEDs devam fazer indicações simultâneas, o LED vermelho terá precedência sobre o LED verde, este último permanecendo apagado.

3 INSTALAÇÃO DA REDE CANOPEN

A rede CANopen, como várias redes de comunicação industriais, pelo fato de ser aplicada muitas vezes em ambientes agressivos e com alta exposição a interferência eletromagnética, exige certos cuidados que devem ser tomados para garantir uma baixa taxa de erros de comunicação durante a sua operação. A seguir são apresentadas recomendações para realizar a instalação do produto na rede.

3.1 TAXA DE COMUNICAÇÃO

Equipamentos com interface CANopen em geral permitem configurar a taxa de comunicação desejada, podendo variar de 10Kbit/s até 1Mbit/s. A taxa de comunicação (*baud rate*) que pode ser utilizada por um equipamento também depende do comprimento do cabo utilizado na instalação. A tabela a seguir apresenta a relação entre as taxas de comunicação e o comprimento máximo de cabo que pode ser utilizado na instalação, de acordo com o recomendado pela CiA[4].

Tabela 3.1: Taxas de comunicação suportadas e comprimento do cabo

Taxa de comunicação	Comprimento do cabo
1 Mbit/s	25 m
800 Kbit/s	50 m
500 Kbit/s	100 m
250 Kbit/s	250 m
125 Kbit/s	500 m
100 Kbit/s	600 m
50 Kbit/s	1000 m
20 Kbit/s	1000 m
10 Kbit/s	1000 m

Todos os equipamentos da rede devem ser programados para utilizar a mesma taxa de comunicação. Para o controlador programável PLC300, a taxa de comunicação CAN é programada através do menu Setup.

3.2 ENDEREÇO NA REDE CANOPEN

Todo dispositivo na rede CANopen deve possuir um endereço, ou Node ID, entre 1 e 127. Este endereço precisa ser diferente para cada equipamento. Para o controlador programável PLC300, o endereço do equipamento é programado através do menu Setup.

3.3 RESISTORES DE TERMINAÇÃO

A utilização de resistores de terminação nas extremidades do barramento CAN é fundamental para evitar reflexão de linha, que pode prejudicar o sinal transmitido e ocasionar erros na comunicação. Resistores de terminação no valor de 120Ω / 0.25W devem ser conectados entre os sinais CAN_H e CAN_L nas extremidades do barramento principal.

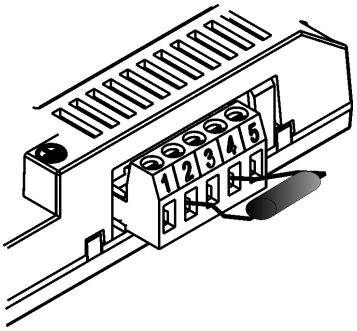


Figura 3.1: Exemplo de instalação do resistor de terminação

[4] Diferentes produtos podem apresentar variações no comprimento máximo do cabo suportado para a instalação.

3.4 CABO

Para a ligação dos sinais CAN_L e CAN_H deve-se utilizar par trançado com blindagem. A tabela a seguir apresenta as características recomendadas para o cabo.

Tabela 3.2: Características do cabo para rede CANopen

Comprimento do cabo (m)	Resistência por metro (mOhm/m)	Área do condutor (mm2)
0 ... 40	70	0.25 ... 0.34
40 ... 300	<60	0.34 ... 0.60
300 ... 600	<40	0.50 ... 0.60
600 ... 1000	<26	0.75 ... 0.80

A utilização de um par trançado adicional para levar a alimentação de 24Vcc para os equipamentos que necessitam deste sinal também é recomendada.

3.5 LIGAÇÃO NA REDE

Para interligar os diversos nós da rede, recomenda-se a conexão do equipamento diretamente a partir da linha principal, sem a utilização de derivações. Durante a instalação dos cabos, deve-se evitar sua passagem próxima a cabos de potência, pois isto facilita a ocorrência de erros durante a transmissão devido à interferência eletromagnética. Para evitar problemas de circulação de corrente por diferença de potencial entre diferentes aterramentos, é necessário que todos os dispositivos estejam conectados no mesmo ponto de terra.

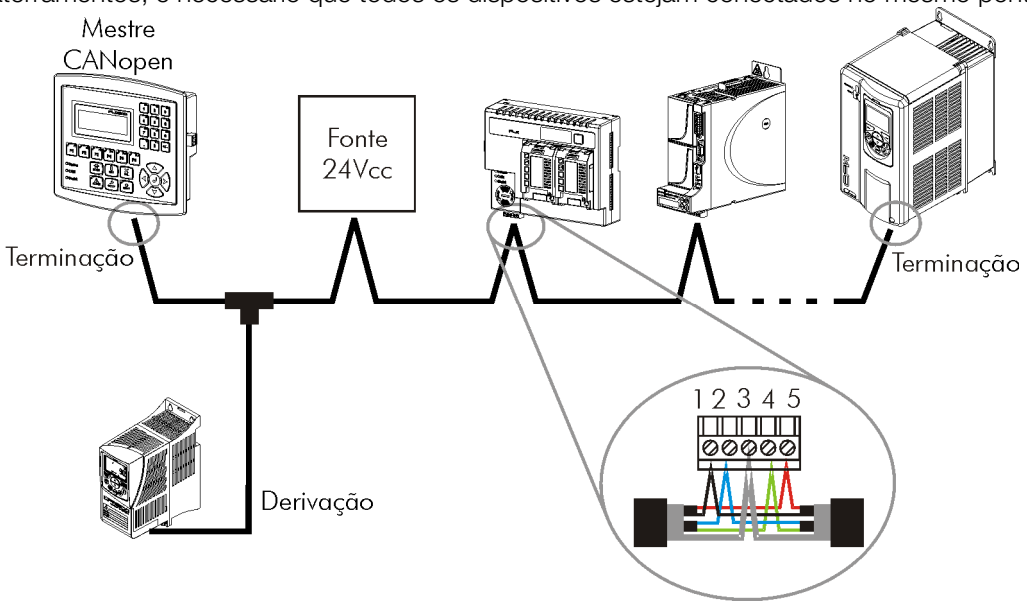


Figura 3.2: Exemplo de instalação em rede CANopen

Para evitar problemas de diferença de tensão na alimentação entre os dispositivos da rede, é recomendado que a rede seja alimentada em apenas um ponto, e o sinal de alimentação seja levado a todos os dispositivos através do cabo. Caso seja necessária mais de uma fonte de alimentação, estas devem estar referenciadas ao mesmo ponto.

O número máximo de dispositivos conectados em um único segmento da rede é limitado em 64. Repetidores podem ser utilizados para conectar um número maior de dispositivos.

4 CONFIGURAÇÃO DA INTERFACE CAN

Para realizar configuração da interface CAN, através do Setup do controlador programável PLC300 são disponibilizados seguintes menus:

BAUD RATE

Faixa de Valores:	0 = 1 Mbit/s 1 = Reservado 2 = 500 Kbit/s 3 = 250 Kbit/s 4 = 125 Kbit/s 5 = 100 Kbit/s 6 = 50 Kbit/s 7 = 20 Kbit/s	**Padrão:** 0

Descrição:
Permite programar o valor desejado para a taxa de comunicação da interface CAN, em bits por segundo. Esta taxa deve ser a mesma para todos os equipamentos conectados na rede.

Caso este parâmetro seja alterado, a alteração terá efeito somente se a interface CAN estiver sem alimentação ou após o equipamento ser desligado e ligado novamente.

ENDEREÇO

Faixa de Valores:	1 a 127	**Padrão:** 1

Descrição:
Permite programar o endereço utilizado para comunicação CAN do dispositivo. É necessário que cada equipamento da rede possua um endereço diferente dos demais.

Caso este parâmetro seja alterado, a alteração terá efeito somente se a interface CAN estiver sem alimentação ou após o equipamento ser desligado e ligado novamente.

5 DICIONÁRIO DE OBJETOS

O dicionário de objetos é uma lista com os diversos dados do equipamento que são acessíveis através da rede CANopen. Um objeto desta lista é identificado através de um índice de 16 bits, e é baseado nesta lista que toda a troca de dados entre os dispositivos é efetuada.

O documento CiA DS 301 define um conjunto mínimo de objetos que todo o escravo da rede CANopen deve possuir. Os objetos disponíveis nesta lista são agrupados de acordo com o tipo de função que ele executa. Os objetos são dispostos no dicionário da seguinte maneira:

Tabela 5.1: Agrupamentos do dicionário de objetos

Índice	Objetos	Descrição
0001h – 025Fh	Definição dos tipos de dados	Utilizado como referência para os tipos de dados suportados pelo sistema.
1000h – 1FFFh	Objetos de comunicação	São objetos comuns a todos os dispositivos CANopen. Contém informações gerais sobre o equipamento e também dados para a configuração da comunicação.
2000h – 5FFFh	Objetos específicos do fabricante	Nesta faixa, cada fabricante de equipamentos CANopen é livre para definir quais dados estes objetos representarão.
6000h – 9FFFh	Objetos padronizados para dispositivos	Esta faixa é reservada para objetos que descrevem o comportamento de equipamentos similares, independente do fabricante.

Demais índices não referenciados nesta lista são reservados.

5.1 ESTRUTURA DO DICIONÁRIO

A estrutura geral do dicionário de objetos possui o seguinte formato:

Índice	Objeto	Nome	Tipo	Acesso

- *Índice:* indica diretamente o índice do objeto no dicionário.
- *Objeto:* descreve que informação o índice armazena (variável simples, array, record, etc.)
- *Nome:* contém o nome do objeto para facilitar sua identificação.
- *Tipo:* indica diretamente o tipo de dado armazenado. Para variáveis simples, este tipo pode ser um inteiro, um float, etc. Para arrays, ele indica o tipo do dado contido no array. Para records, ele indica o formato do record, de acordo com os tipos descritos na primeira parte do dicionário de objetos (índices 0001h – 025Fh).
- *Acesso:* informa se o objeto em questão está acessível somente para leitura (ro), para leitura e escrita (rw), ou é uma constante (const).

Para objetos do tipo array ou records, ainda é necessário um sub-índice, que não é descrito na estrutura do dicionário.

5.2 TIPOS DE DADOS

A primeira parte do dicionário de objetos (índices 0001h – 025Fh) descreve os tipos de dados que podem ser acessados em um dispositivo na rede CANopen. Estes podem ser tipos básicos, como inteiros e floats, ou tipos compostos, formados por um conjunto de entradas, como records e arrays.

5.3 COMMUNICATION PROFILE – OBJETOS PARA COMUNICAÇÃO

Os índices de 1000h até 1FFFh correspondem, no dicionário de objetos, à parte responsável pelas configurações da comunicação na rede CANopen. Estes objetos são comuns a todos os dispositivos, mas somente alguns são obrigatórios. A seguir é apresentada uma lista com alguns dos objetos desta faixa suportados pelo controlador programável PLC300, operando no modo escravo.

Tabela 5.2: Lista de objetos – Communication Profile

Índice	Objeto	Nome	Tipo	Acesso
1000h	VAR	device type	UNSIGNED32	ro
1001h	VAR	error register	UNSIGNED8	ro
1005h	VAR	COB-ID SYNC	UNSIGNED32	rw
100Ch	VAR	guard time	UNSIGNED16	rw
100Dh	VAR	life time factor	UNSIGNED8	rw
1016h	ARRAY	Consumer heartbeat time	UNSIGNED32	rw
1017h	VAR	Producer heartbeat time	UNSIGNED16	rw
1018h	RECORD	Identity Object	Identity	ro
Server SDO Parameter				
1200h	RECORD	1st Server SDO parameter	SDO Parameter	ro
Receive PDO Communication Parameter				
1400h	RECORD	1st receive PDO Parameter	PDO CommPar	rw
1401h	RECORD	2nd receive PDO Parameter	PDO CommPar	rw
...				
1407h	RECORD	8th receive PDO Parameter	PDO CommPar	rw
Receive PDO Mapping Parameter				
1600h	RECORD	1st receive PDO mapping	PDO Mapping	rw
1601h	RECORD	2nd receive PDO mapping	PDO Mapping	rw
...				
1607h	RECORD	8th receive PDO mapping	PDO Mapping	rw
Transmit PDO Communication Parameter				
1800h	RECORD	1st transmit PDO Parameter	PDO CommPar	rw
1801h	RECORD	2nd transmit PDO Parameter	PDO CommPar	rw
...				
1807h	RECORD	8th transmit PDO Parameter	PDO CommPar	rw
Transmit PDO Mapping Parameter				
1A00h	RECORD	1st transmit PDO mapping	PDO Mapping	rw
1A01h	RECORD	2nd transmit PDO mapping	PDO Mapping	rw
...				
1A07h	RECORD	8th transmit PDO mapping	PDO Mapping	rw

Estes objetos somente podem ser lidos e escritos através da rede CANopen, não estão disponíveis via HMI ou outra interface de rede. O mestre da rede, em geral, é o equipamento responsável pela configuração do equipamento antes de iniciar a operação. O arquivo de configuração EDS traz a lista de todos os objetos de comunicação suportados.

Para uma descrição detalhada de quais objetos estão disponíveis nesta faixa do dicionário de objetos, consulte o item 6.

5.4 MANUFACTURER SPECIFIC – OBJETOS ESPECÍFICOS DO FABRICANTE

Nos índices de 2000h até 5FFFh, cada fabricante é livre para definir quais objetos estarão presentes, o tipo e a função de cada objeto. Para o controlador programável PLC300, nesta faixa de objetos foram disponibilizados os marcadores de rede. Estes marcadores comunicados através da interface CANopen podem ser utilizados no software de programação do controlador para elaboração da lógica de operação do equipamento. A tabela a seguir ilustra como estão distribuídos os marcadores no dicionário de objetos.

Tabela 5.3: Lista de objetos – Manufacturer Specific

Índice	Objeto	Nome	Tipo	Acesso
Network Input Data – Byte Access				
3000h	VAR	Network Input Byte 2000 – %IB2000	UNSIGNED8	rw
3001h	VAR	Network Input Byte 2001 – %IB2001	UNSIGNED8	rw
3002h	VAR	Network Input Byte 2002 – %IB2002	UNSIGNED8	rw
...	...	...	...	...
31FFh	VAR	Network Input Byte 2511 – %IB2511	UNSIGNED8	rw
Network Input Data – Word Access				
3400h	VAR	Network Input Word 2000 – %IW2000	UNSIGNED16	rw
3402h	VAR	Network Input Word 2002 – %IW2002	UNSIGNED16	rw
3404h	VAR	Network Input Word 2004 – %IW2004	UNSIGNED16	rw
...	...	...	...	...
35FEh	VAR	Network Input Word 2510 – %IW2510	UNSIGNED16	rw
Network Input Data – Double Word Access				
3800h	VAR	Network Input Double Word 2000 – %ID2000	UNSIGNED32	rw
3804h	VAR	Network Input Double Word 2004 – %ID2004	UNSIGNED32	rw
3808h	VAR	Network Input Double Word 2008 – %ID2008	UNSIGNED32	rw
...	...	...	...	...
39FCh	VAR	Network Input Double Word 2508 – %ID2508	UNSIGNED32	rw
Network Output Data – Byte Access				
4000h	VAR	Network Output Byte 2000 – %QB2000	UNSIGNED8	rw
4001h	VAR	Network Output Byte 2001 – %QB2001	UNSIGNED8	rw
4002h	VAR	Network Output Byte 2002 – %QB2002	UNSIGNED8	rw
...	...	...	...	...
41FFh	VAR	Network Output Byte 2511 – %QB2511	UNSIGNED8	rw
Network Output Data – Word Access				
4400h	VAR	Network Output Word 2000 – %QW2000	UNSIGNED16	rw
4402h	VAR	Network Output Word 2002 – %QW2002	UNSIGNED16	rw
4404h	VAR	Network Output Word 2004 – %QW2004	UNSIGNED16	rw
...	...	...	...	...
45FEh	VAR	Network Output Word 2510 – % QW2510	UNSIGNED16	rw
Network Output Data – Double Word Access				
4800h	VAR	Network Output Double Word 2000 – %QD2000	UNSIGNED32	rw
4804h	VAR	Network Output Double Word 2004 – %QD2004	UNSIGNED32	rw
4808h	VAR	Network Output Double Word 2008 – %QD2008	UNSIGNED32	rw
...	...	...	...	...
49FCh	VAR	Network Output Double Word 2508 – %QD2508	UNSIGNED32	rw

NOTA!

- Os marcadores de entrada (input) de Byte, Word e Double Word compartilham a mesma área de memória interna no produto. Desta forma, por exemplo, os marcadores %IB2000 e %IB2001 ocupam a mesma área de memória que o marcador %IW2000. Diferentes objetos foram criados apenas para disponibilizar para mapeamento dos dados via CANopen objetos de diferentes tamanhos. O mesmo vale para a área de saída (output).
- Os tipos de dados utilizados nestes objetos são definidos como inteiro de 8, 16 ou 32 bits sem sinal. Este tipo é utilizado para definir o tamanho dos dados utilizados na comunicação CANopen apenas. O tipo real que o marcador representa, no entanto, depende do tipo declarado no software de programação do controlador. O marcador %QD2000, por exemplo, pode representar um dado do tipo float, dependendo do que foi declarado no software de programação do produto.
- Os marcadores de rede de entrada (input) podem ser mapeados nos PDOs de recepção, enquanto que os marcadores de rede de saída (output) podem ser mapeados nos PDOs de transmissão.

6 DESCRIÇÃO DOS OBJETOS DE COMUNICAÇÃO

Neste item são descritos detalhadamente cada um dos objetos de comunicação disponíveis para o controlador programável PLC300 operando no modo escravo. É necessário conhecer como estes objetos são operados para utilizar as funções disponíveis para a comunicação do drive.

NOTA!
O controlador programável PLC300 pode operar como mestre ou escravo da rede CANopen. Os objetos descritos a seguir descrevem a operação do equipamento como escravo da rede CANopen. Para a descrição das características do produto operando como mestre da rede CANopen, consulte o item 7 juntamente com o software de configuração da rede CANopen WSCAN.

6.1 OBJETOS DE IDENTIFICAÇÃO

Existe um conjunto de objetos no dicionário utilizados para identificação do equipamento, porém não possuem influência no seu comportamento na rede CANopen.

6.1.1 Objeto 1000h – Device Type

Este objeto fornece um código em 32 bits que descreve o tipo de objeto e sua funcionalidade.

Índice	1000h
Nome	Device type
Objeto	VAR
Tipo	UNSIGNED32

Acesso	ro
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	0000.0000h

Este código pode ser dividido em duas partes: 16 bits inferiores, descrevendo o tipo de perfil (*profile)* que o dispositivo utiliza, e 16 bits superiores, indicando uma função específica, de acordo com o perfil especificado.

6.1.2 Objeto 1001h – Error Register

Este objeto indica a ocorrência ou não de erro no dispositivo. O tipo de erro registrado para o equipamento segue o descrito pela tabela a seguir.

Índice	1001h
Nome	Error register
Objeto	VAR
Tipo	UNSIGNED8

Acesso	ro
Mapeável	Sim
Faixa	UNSIGNED8
Valor Padrão	0

Tabela 6.1: Estrutura do objeto Error Register

Bit	Significado
0	Erro genérico
1	Corrente
2	Tensão
3	Temperatura
4	Comunicação
5	Reservado (sempre 0)
6	Reservado (sempre 0)
7	Específico do fabricante

Caso o dispositivo apresente algum erro, o bit equivalente deve ser ativado. O primeiro bit (erro genérico) deverá ser ativado em qualquer situação de erro.

6.1.3 Objeto 1018h – Identity Object

Traz informações gerais sobre o dispositivo.

Índice	1018h
Nome	Identity objetct
Objeto	Record
Tipo	Identity

Sub-índice	0
Descrição	Número do último sub-índice
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	4

Sub-índice	1
Descrição	Vendor ID
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	0000.0123h

Sub-índice	2
Descrição	Código do produto
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	0000.0220h

Sub-índice	3
Descrição	Número da revisão
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	De acordo com a versão de firmware do equipamento

Sub-índice	4
Descrição	Número serial
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	Diferente para cada PLC300

O Vendor ID é um número que identifica o fabricante junto à CiA. O código do produto é definido pelo fabricante de acordo com o tipo de produto. O número da revisão representa a versão de firmware do equipamento. O sub-índice 4 é um número serial único para cada controlador programável PLC300 em rede CANopen.

6.2 SERVICE DATA OBJECTS – SDOS

Os SDOs são responsáveis pelo acesso direto ao dicionário de objetos de um determinado dispositivo na rede. Eles são utilizados para a configuração e, portanto, possuem baixa prioridade, já que não devem ser utilizados para comunicar dados necessários para a operação do dispositivo.

Existem dois tipos de SDOs: cliente e servidor. Basicamente, a comunicação inicia com o cliente (usualmente o mestre da rede) fazendo uma requisição de leitura (*upload*) ou escrita (*download*) para um servidor, e este responde ao que foi requisitado.

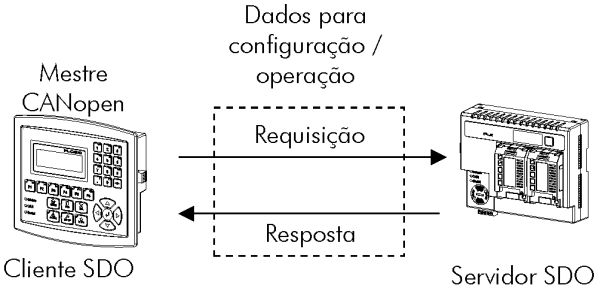


Figura 6.1: Comunicação entre cliente e servidor SDO

6.2.1 Objeto 1200h – Servidor SDO

O controlador programável PLC300 operando no modo escravo possui um único SDO do tipo servidor, que possibilita o acesso a todo o seu dicionário de objetos. Através dele, um cliente SDO pode configurar a comunicação, parâmetros e modos de operação do drive. Todo o servidor SDO possui um objeto, do tipo SDO_PARAMETER, para a sua configuração, possuindo a seguinte estrutura:

Índice	1200h
Nome	Server SDO Parameter
Objeto	Record
Tipo	SDO Parameter

Sub-índice	0
Descrição	Número do último sub-índice
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	2

Sub-índice	1
Descrição	COB-ID Cliente - Servidor (rx)
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	600h + Node-ID

Sub-índice	2
Descrição	COB-ID Servidor - Cliente (tx)
Acesso	RO
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	580h + Node-ID

6.2.2 Funcionamento dos SDOs

Um telegrama enviado por um SDO possui 8 bytes de tamanho, com a seguinte estrutura:

Identificador	8 bytes de dados							
11 bits	Comando	Índice		Sub-índice	Dados do objeto			
	byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7

O identificador depende do sentido da transmissão (rx ou tx) e do endereço (ou Node-ID) do servidor destino. Por exemplo, um cliente que faz uma requisição para um servidor cujo Node-ID é 1, deve enviar uma mensagem com o identificador igual a 601h. O servidor irá receber esta mensagem e responder com um telegrama cujo COB-ID é igual a 581h.

O código do comando depende do tipo de função utilizada. Para as transmissões de um cliente para um servidor, podem ser utilizados os seguintes comandos:

Tabela 6.2: Código dos comandos para cliente SDO

Comando	Função	Descrição	Dados do objeto
22h	*Download*	Escrita em objeto	Indefinido
23h	*Download*	Escrita em objeto	4 bytes
2Bh	*Download*	Escrita em objeto	2 bytes
2Fh	*Download*	Escrita em objeto	1 byte
40h	*Upload*	Leitura de objeto	Não utilizado
60h ou 70h	*Upload segment*	Leitura segmentada	Não utilizado

Ao fazer a requisição, o cliente indicará através de seu COB-ID, qual o endereço do escravo para o qual esta requisição se destina. Somente um escravo (usando seu respectivo servidor SDO) poderá responder para o cliente o telegrama recebido. O telegrama de resposta possuirá também a mesma estrutura do telegrama de requisição, mas os comandos serão diferentes:

Tabela 6.3: Código dos comandos para servidor SDO

Comando	Função	Descrição	Dados do objeto
60h	*Download*	Resposta para escrita em objeto	Não utilizado
43h	*Upload*	Resposta para leitura de objeto	4 bytes
4Bh	*Upload*	Resposta para leitura de objeto	2 bytes
4Fh	*Upload*	Resposta para leitura de objeto	1 byte
41h	*Upload segment*	Inicia resposta segmentada para leitura	4 bytes
01h ... 0Dh	*Upload segment*	Último segmento de dados para leitura	8 ... 2 bytes

Para leituras que envolvem até quatro bytes de dados, uma única mensagem pode ser transmitida pelo servidor; para leitura de uma quantidade maior de bytes, é necessário que cliente e servidor troquem múltiplos telegramas.

Um telegrama somente é completo após a confirmação do servidor para a requisição feita pelo cliente. Caso algum erro seja detectado durante a troca de telegramas (por exemplo, não há resposta do servidor), o cliente poderá abortar o processo com uma mensagem de aviso com o código do comando igual a 80h.

NOTA!
Os valores recebidos através destes objetos não são salvos na memória não volátil. Desta forma, após um desligamento ou reset do equipamento, os objetos modificados pelo SDO voltam para o seu valor padrão.

Exemplo: um cliente SDO solicita para um escravo no endereço 1, a leitura do objeto identificado pelo índice 3000h, sub-índice 0 (zero), que representa um inteiro de 16 bits. O telegrama do mestre possui a seguinte forma:

Identificador	Comando	Índice		Sub-índice	Dados			
601h	40h	00h	30h	00h	00h	00h	00h	00h

O escravo responde à requisição, indicando que o valor para o referido objeto é igual a 999[5]:

Identificador	Comando	Índice		Sub-índice	Dados			
581h	4Bh	00h	30h	00h	E7	03h	00h	00h

6.3 PROCESS DATA OBJECTS – PDOS

Os PDOs são utilizados para enviar e receber dados utilizados durante a operação do dispositivo, que muitas vezes precisam ser transmitidos de forma rápida e eficiente. Por isso, eles possuem uma prioridade maior do que os SDOs.

Nos PDOs, apenas os dados são transmitidos no telegrama (índices e sub-índices são omitidos), e desta forma é possível fazer uma transmissão mais eficiente, com maior volume de dados em um único telegrama. É necessário, porém, configurar previamente o que está sendo transmitido pelo PDO, de forma que, mesmo sem a indicação do índice e sub-índice, seja possível saber o conteúdo do telegrama.

[5] Não esquecer que qualquer dado do tipo inteiro, a ordem de transferência dos bytes vai do menos significativo até o mais significativo.

Existem dois tipos de PDOs, os PDOs de recepção e os PDOs de transmissão. Os PDOs de transmissão são responsáveis por enviar dados para a rede, enquanto que os PDOs de recepção ficam responsáveis por receber e tratar estes dados. Desta forma é possível que haja comunicação entre escravos da rede CANopen, basta configurar um escravo para transmitir uma informação, e um ou mais escravos para receber esta informação.

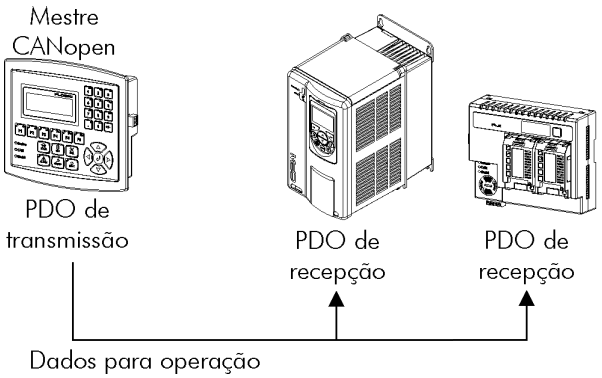


Figura 6.2: Comunicação utilizando PDOs

NOTA!
PDOs somente podem ser transmitidos ou recebidos quando o dispositivo está no estado operacional.

6.3.1 Objetos Mapeáveis para os PDOs

Para um objeto poder ser transmitido através de um PDO, é necessário que ele seja mapeável para o conteúdo do PDO. Na descrição dos objetos de comunicação (1000h – 1FFFh), o campo "Mapeável" informa esta condição. Usualmente, apenas informações necessárias para a operação do dispositivo são mapeáveis, como comandos para habilitação, status do dispositivo, referências, etc. Informações para configuração do dispositivo não são acessíveis através de PDOs, e caso seja necessário acessá-las via rede deve-se utilizar os SDOs.

Para os objetos específicos do fabricante (2000h – 5FFFh), os marcadores de rede de entrada (input) podem ser mapeados nos PDOs de recepção, enquanto que os marcadores de rede de saída (output) podem ser mapeados nos PDOs de transmissão. O item 5.4 descreve os objetos mapeáveis para os PDOs.

O arquivo EDS do equipamento traz a lista de todos os objetos disponíveis, informando se o objeto é mapeável ou não.

6.3.2 PDOs de Recepção

Os PDOs de recepção, ou RPDOs, são responsáveis por receber dados que outros dispositivos enviam para a rede CANopen. O controlador programável PLC300 operando no modo escravo possui 8 PDOs de recepção, cada um podendo receber até 8 bytes de dados. Cada RPDO possui dois parâmetros para sua configuração, um PDO_COMM_PARAMETER e um PDO_MAPPING, conforme descrito a seguir.

PDO_COMM_PARAMETER

Índice	1400h até 1407h
Nome	Receive PDO communication parameter
Objeto	Record
Tipo	PDO COMM PARAMETER

Sub-índice	0
Descrição	Número do último sub-índice
Acesso	ro
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	2

Sub-índice	1
Descrição	COB-ID usado pelo PDO
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	1400h: 200h + Node-ID 1401h: 300h + Node-ID 1402h: 400h + Node-ID 1403h: 500h + Node-ID 1404h – 1407h: 0

Sub-índice	2
Descrição	Tipo de transmissão
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	254

O sub-índice 1 contém o COB-ID do PDO de recepção. Sempre que uma mensagem for enviada para a rede, este objeto irá ler qual o COB-ID desta mensagem, e caso ele seja igual ao valor deste campo, a mensagem será recebida pelo dispositivo. Este campo é formado por um UNSIGNED32 com a seguinte estrutura:

Tabela 6.4: Descrição do COB-ID

Bit	Valor	Descrição
31 (MSB)	0	PDO está habilitado
	1	PDO está desabilitado
30	0	RTR permitido
29	0	Tamanho do identificador = 11 bits
28 – 11	0	Não utilizado, sempre 0
10 – 0 (LSB)	X	COB-ID de 11 bits

O bit 31 permite habilitar ou desabilitar o PDO. Os bits 30 e 29, que devem ser mantidos em 0 (zero), indicam respectivamente que o PDO aceita frames remotos (RTR frames) e que utiliza identificador de 11 bits. Como o drive não utiliza identificadores de 29 bits, os bits de 28 até 11 devem ser mantidos em 0 (zero), enquanto que os bits de 10 até 0 (zero) são usados para configurar o COB-ID para o PDO.

O sub-índice 2 indica o tipo de transmissão deste objeto, de acordo com a tabela a seguir.

Tabela 6.5: Descrição do tipo de transmissão

Tipo de transmissão	Transmissão de PDOs				
	Cíclico	*Acíclico*	*Síncrono*	*Assíncrono*	*RTR*
0		•	•		
1 – 240	•		•		
241 – 251	Reservado				
252			•		•
253				•	•
254				•	
255				•	

- ***Valores 0 – 240:*** qualquer RPDOs programado nesta faixa possui o mesmo funcionamento. Ao detectar uma mensagem, ele irá receber os dados, porém não atualizará os valores recebidos até detectar o próximo telegrama SYNC.
- ***Valores 252 e 253:*** não permitido para PDOs de recepção.
- ***Valores 254 e 255:*** indica que não possui relação com o objeto de sincronização. Ao receber uma mensagem, seus valores serão atualizados imediatamente.

PDO_MAPPING

Índice	1600h até 1607h
Nome	Receive PDO mapping
Objeto	Record
Tipo	PDO MAPPING

Sub-índice	0
Descrição	Número de objetos mapeados
Acesso	RO
Mapeável	Não
Faixa	0 = desabilitado 1 ... 8= número de objetos mapeados
Valor Padrão	0

Sub-índice	1 até 8
Descrição	1º até 8º objeto mapeado no PDO
Acesso	Rw
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	Indicado no arquivo EDS

Este parâmetro indica os objetos mapeados nos PDOs de recepção do controlador programável PLC300. O valor padrão destes objetos é indicado no arquivo EDS do produto. Para cada RPDO, é possível mapear até 8 objetos diferentes, desde que o tamanho total não ultrapasse oito bytes. O mapeamento de um objeto é feito indicando o seu índice, sub-índice[6] e tamanho (em bits) em um campo UNSIGNED32, com o seguinte formato:

UNSIGNED32		
Índice (16 bits)	Sub-índice (8 bits)	Tamanho do objeto (8 bits)

Por exemplo, supondo um mapeamento para um PDO de recepção com os seguintes valores configurados, temos:

- *Sub-índice 0 = 2:* o RPDO possui dois objetos mapeados.
- *Sub-índice 1 = 3400.0010h:* o primeiro objeto mapeado possui índice igual a 3400h, sub-índice 0 (zero), e tamanho igual a 16 bits. Este objeto corresponde ao marcador de entrada %IW2000.
- *Sub-índice 2 = 3804.0020h:* o segundo objeto mapeado possui índice igual a 3804h, sub-índice 0 (zero), e tamanho igual a 32 bits. Este objeto corresponde ao marcador de entrada %ID2004.

É possível modificar este mapeamento, alterando a quantidade ou o número dos objetos mapeados. Lembrar que no máximo podem ser mapeados 8 objetos ou 8 bytes.

NOTA!

- Para poder alterar os objetos mapeados em um PDO, primeiro é necessário escrever o valor 0 (zero) no sub-índice 0 (zero). Desta forma, os valores dos sub-índices 1 até 8podem ser alterados. Depois de feito o mapeamento desejado, deve-se escrever novamente no sub-índice 0 (zero) o número de objetos que foram mapeados, habilitando novamente o PDO.
- Os valores recebidos através destes objetos não são salvos na memória não volátil. Desta forma, após um desligamento ou reset do equipamento, os objetos modificados por um RPDO voltam para o seu valor padrão.
- Não esquecer que os PDOs somente podem ser recebidos caso o dispositivo esteja no estado operacional.

6.3.3 PDOs de Transmissão

Os PDOs de transmissão, ou TPDOs, como o nome diz, são responsáveis por transmitir dados para a rede CANopen. O controlador programável PLC300 possui 8 PDOs de transmissão, cada um podendo transmitir até 8 bytes de dados. De forma semelhante aos RPDOs, cada TPDO possui dois parâmetros para sua configuração, um PDO_COMM_PARAMETER e um PDO_MAPPING, conforme descrito a seguir.

PDO_COMM_PARAMETER

Índice	1800h até 1807h
Nome	Transmit PDO Parameter
Objeto	Record
Tipo	PDO COMM PARAMETER

[6] Caso o objeto seja do tipo VAR e não possua sub-índice, deve ser indicado o valor 0 (zero) para o sub-índice.

Sub-índice	0
Descrição	Número do último sub-índice
Acesso	ro
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	5

Sub-índice	1
Descrição	COB-ID usado pelo PDO
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	1800h: 180h + Node-ID 1801h: 280h + Node-ID 1802h: 380h + Node-ID 1803h: 480h + Node-ID 1804h – 1807h: 0

Sub-índice	2
Descrição	Tipo de transmissão
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	254

Sub-índice	3
Descrição	Tempo entre transmissões
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED16
Valor Padrão	-

Sub-índice	4
Descrição	Reservado
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	-

Sub-índice	5
Descrição	Temporizador de eventos
Acesso	rw
Mapeável	Não
Faixa	0 = desabilitado UNSIGNED16
Valor Padrão	0

O sub-índice 1 contém o COB-ID do PDO de transmissão. Sempre que este PDO enviar uma mensagem para a rede, o identificador desta mensagem será este COB-ID. A estrutura deste campo é descrita na tabela 6.4.

O sub-índice 2 indica o tipo de transmissão deste objeto, que segue o descrito pela tabela 6.5. Porém seu funcionamento é diferente para PDOs de transmissão:

- ***Valor 0:*** indica que a transmissão deve ocorrer imediatamente após a recepção de um telegrama SYNC, mas não periodicamente.
- ***Valores 1 – 240:*** o PDO deve ser transmitido a cada telegrama SYNC detectado (ou ocorrências múltiplas de SYNC, de acordo com o número escolhido entre 1 e 240).
- ***Valor 252:*** indica que o conteúdo da mensagem deve ser atualizado (mas não enviado), após a recepção de um telegrama SYNC. O envio da mensagem deve ser feito após a recepção de um frame remoto (RTR frame).
- ***Valor 253:*** o PDO deve atualizar e enviar uma mensagem assim que receber um frame remoto.
- ***Valores 254:*** o objeto deve ser transmitido de acordo com o timer programado no sub-índice 5.
- ***Valores 255:*** o objeto é transmitido automaticamente quando o valor de algum dos objetos mapeados neste PDO for alterado. Funciona por alteração de estado (Change Of State). Este tipo também permite que o PDO seja transmitido de acordo com o timer programado no sub-índice 5.

No sub-índice 3 é possível programar um tempo mínimo (em múltiplos de 100us) que deve transcorrer para que, depois de transmitido um telegrama, um novo telegrama possa ser enviado por este PDO. O valor 0 (zero) desabilita esta função.

O sub-índice 5 contém um valor para habilitar um temporizador para o envio automático de um PDO. Desta forma, sempre que um PDO for configurado para o tipo assíncrono, é possível programar o valor deste temporizador (em múltiplos de 1ms), para que o PDO seja transmitido periodicamente no tempo programado.

NOTA!

- Deve-se observar o tempo programado neste temporizador, de acordo com a taxa de transmissão utilizada. Tempos muito pequenos (próximos ao tempo de transmissão do telegrama) podem monopolizar o barramento, causando a retransmissão indefinida do PDO e impedindo que outros objetos menos prioritários possam transmitir seus dados.
- O tempo mínimo permitido para esta função no controlador programável PLC300 é 1ms.
- É importante observar o tempo entre transmissões programado no sub-índice 3 principalmente quando o PDO for programado com o valor 255 no sub-índice 2 (Change Of State).
- Não esquecer que os PDOs somente podem ser transmitidos caso o escravo esteja no estado operacional.

PDO_MAPPING

Índice	1A00h até 1A07h
Nome	Transmit PDO mapping
Objeto	Record
Tipo	PDO MAPPING

Sub-índice	0
Descrição	Número do último sub-índice
Acesso	ro
Mapeável	Não
Faixa	0 = desabilitado 1 ... 8= número de objetos mapeados
Valor Padrão	0

Sub-índice	1 até 8
Descrição	1º até 8º objeto mapeado no PDO
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	0

O PDO MAPPING para a transmissão funciona de forma semelhante ao de recepção, porém neste caso são definidos os dados a serem transmitidos pelo PDO. Cada objeto mapeado deve ser colocado na lista de acordo com o descrito a seguir:

UNSIGNED32		
Índice (16 bits)	Sub-índice (8 bits)	Tamanho do objeto (8 bits)

Por exemplo, supondo um mapeamento para um PDO de recepção com os seguintes valores configurados, temos:

- *Sub-índice 0 = 2:* o RPDO possui dois objetos mapeados.
- *Sub-índice 1 = 3000.0008h:* o primeiro objeto mapeado possui índice igual a 3000h, sub-índice 0 (zero), e tamanho igual a 8 bits. Este objeto corresponde ao marcador de entrada %QB2000.
- *Sub-índice 2 = 4804.0020h:* o segundo objeto mapeado possui índice igual a 4804h, sub-índice 0 (zero), e tamanho igual a 32 bits. Este objeto corresponde ao marcador de entrada %QD2004.

É possível modificar este mapeamento, alterando a quantidade ou o número dos parâmetros mapeados. Lembrar que no máximo podem ser mapeados 8 objetos ou 8 bytes.

NOTA!
Para poder alterar os objetos mapeados em um PDO, primeiro é necessário escrever o valor 0 (zero) no sub-índice 0 (zero). Desta forma, os valores dos sub-índices 1 até 8 podem ser alterados. Depois de feito o mapeamento desejado, deve-se escrever novamente no sub-índice 0 (zero) o número de objetos que foram mapeados, habilitando novamente o PDO.

6.4 SYNCHRONIZATION OBJECT – SYNC

Este objeto é transmitido com o objetivo de permitir a sincronização de eventos entre os dispositivos da rede CANopen. Ele é transmitido por um produtor SYNC, e os dispositivos que detectam a sua transmissão são denominados consumidores SYNC.

O controlador programável PLC300 operando no modo escravo possui a função de consumidor SYNC e, portanto, pode programar seus PDOs para serem síncronos. PDOs síncronos são aqueles relacionados com o objeto de sincronização e, portanto, podem ser programados para serem transmitidos ou atualizados com base neste objeto.

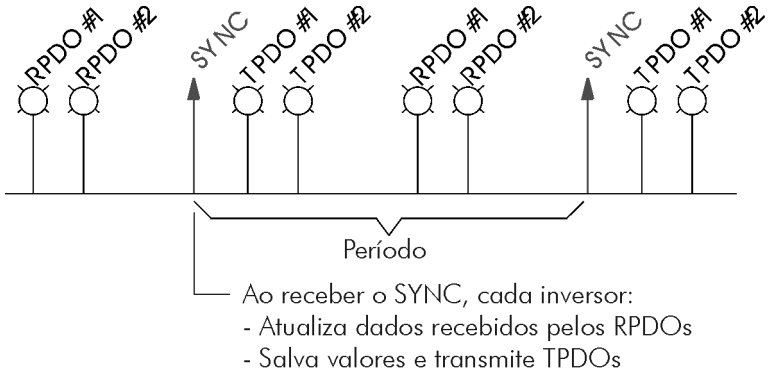


Figura 6.3: SYNC

A mensagem SYNC transmitida pelo produtor não possui dado algum em seu campo de dados, pois seu objetivo é fornecer um evento sincronizado entre os dispositivos da rede. O seguinte objeto está disponível para configuração do consumidor SYNC:

Índice	1015h
Nome	COB-ID SYNC
Objeto	VAR
Tipo	UNSIGNED32

Acesso	rw
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	80h

NOTA!
Deve-se observar o tempo programado no produtor para o período dos telegramas SYNC, de acordo com a taxa de transmissão utilizada e o número de PDOs síncronos a serem transmitidos. É necessário que haja tempo suficiente para a transmissão destes objetos, e também é recomendado que haja folga para possibilitar o envio de mensagens assíncronas, como EMCY, PDOs assíncronos e SDOs.

6.5 NETWORK MANAGEMENT – NMT

O objeto de gerenciamento da rede é responsável por um conjunto de serviços que controlam a comunicação do dispositivo na rede CANopen. Para este objeto estão disponíveis os serviços de controle do nó e de controle de erros (utilizando *Node Guarding* ou *Heartbeat*).

6.5.1 Controle dos Estados do Escravo

Com relação à comunicação, um dispositivo da rede CANopen pode ser descrito pela seguinte máquina de estados:

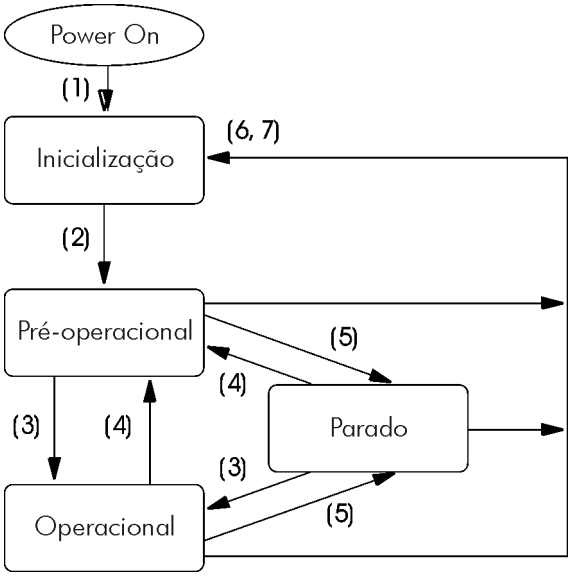


Figura 6.4: Diagrama de estados do nó CANopen

Tabela 6.6: Descrição das transições

Transição	Descrição
1	Dispositivo é ligado e começa a inicialização (automático)
2	Inicialização concluída, vai para o estado pré-operacional (automático)
3	Recebe comando Start Node para entrar no estado operacional
4	Recebe comando Enter Pre-Operational, e vai para o estado pré-operacional
5	Recebe comando Stop Node para entrar no estado parado
6	Recebe comando Reset Node, onde executa o reset completo do dispositivo
7	Recebo comando Reset Communication, onde reinicializa o valor dos objetos e a comunicação CANopen do dispositivo

Durante a inicialização, é definido o Node-ID, criados os objetos e configurada a interface com a rede CAN. Não é possível comunicar-se com o dispositivo nesta etapa, que é concluída automaticamente. No final desta etapa, o escravo envia para rede um telegrama do objeto Boot-up, utilizado apenas para indicar que a inicialização foi concluída e que o escravo entrou no estado pré-operacional. Este telegrama possui identificador 700h + Node-ID, e apenas um byte de dados com valor igual a 0 (zero).

No estado pré-operacional, já é possível comunicar-se com o escravo. Os PDOs, porém, ainda não estão disponíveis para operação. No estado operacional, todos os objetos estão disponíveis, enquanto que no estado parado, apenas o objeto NMT pode receber ou transmitir telegramas para a rede. A tabela a seguir mostra os objetos disponíveis para cada estado.

Tabela 6.7: Objetos acessíveis em cada estado

	Inicialização	Pré-operacional	Operacional	Parado
PDO			•	
SDO		•	•	
SYNC		•	•	
EMCY		•	•	
Boot-up	•			
NMT		•	•	•

Esta máquina de estados é controlada pelo mestre da rede, que envia, para cada escravo, comandos para que seja executada a transição de estados desejada. Estes telegramas não possuem confirmação, o que significa que o escravo apenas recebe o telegrama sem retornar resposta para o mestre. Os telegramas recebidos possuem a seguinte estrutura:

Identificador	byte 1	byte 2
00h	Código do comando	Node-ID destino

Tabela 6.8: Comandos para a transição de estados

Código do comando	Node-ID destino
1 = *START node* (transição 3) 2 = *STOP node* (transição 4) 128 = *Enter pre-operational* (transição 5) 129 = *Reset node* (transição 6) 130 = *Reset comunication* (transição 7)	0 = Todos os escravos 1 ... 127 = Escravo específico

As transições indicadas no código do comando equivalem às transições de estado executadas pelo nó após receber o comando (conforme figura 6.4). O comando *Reset node* faz com que o escravo execute um reset completo do dispositivo, enquanto que o comando *Reset communication* faz com que o escravo reinicialize apenas os objetos relativos à comunicação CANopen.

6.5.2 Controle de Erros – Node Guarding

Este serviço é utilizado para possibilitar a monitoração da comunicação com a rede CANopen, tanto pelo mestre quanto pelo escravo. Neste tipo de serviço, o mestre envia telegramas periódicos para o escravo, que responde o telegrama recebido. Caso ocorra algum erro que interrompa a comunicação, será possível identificar este erro, pois tanto o mestre quanto o escravo serão notificados pelo *timeout* na execução deste serviço. Os eventos de erro são chamados de *Node Guarding* para o mestre, e de *Life Guarding* para o escravo.

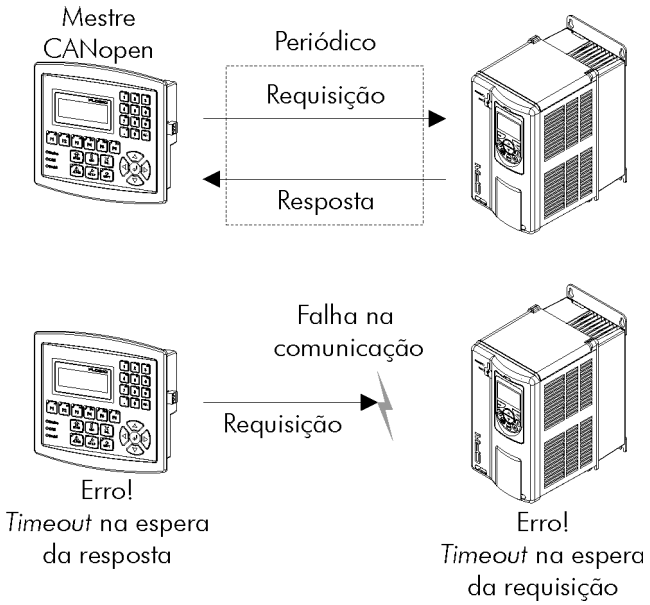


Figura 6.5: Serviço de controle de erros – Node Guarding

Para o serviço de *Node Guarging*, existem dois objetos do dicionário para configuração dos tempos para detecção de erros de comunicação:

Índice	100Ch
Nome	Guard Time
Objeto	VAR
Tipo	UNSIGNED16

Acesso	rw
Mapeável	Não
Faixa	UNSIGNED16
Valor Padrão	0

Índice	100Dh
Nome	Life Time Factor
Objeto	VAR
Tipo	UNSIGNED8

Acesso	rw
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	0

O objeto 100Ch permite programar o tempo necessário (em milisegundos) para que uma ocorrência de falha seja detectada, caso o escravo não receba nenhum telegrama do mestre. O objeto 100Dh indica quantas falhas em sequência são necessárias até que se considere que houve realmente perda da comunicação. Portanto, a multiplicação destes dois valores fornecerá o tempo total necessário para detecção de erros de comunicação utilizando este objeto. O valor 0 (zero) desabilita esta função.

Uma vez configurado, o escravo começa a contar estes tempos a partir do primeiro telegrama *Node Guarding* recebido do mestre da rede. O telegrama do mestre é do tipo remoto, não possuindo bytes de dados. O identificador é igual a 700h + Node-ID do escravo destino. Já o telegrama de resposta do escravo possui 1 byte de dados com a seguinte estrutura:

Identificador	byte 1	
	bit 7	bit 6 ... bit 0
700h + Node-ID	Toggle	Estado do escravo

Este telegrama possui um único byte dados. Este byte contém, nos sete bits menos significativos, um valor para indicar o estado do escravo (4 = Parado, 5 = Operacional e 127 = Pré-operacional), e no oitavo bit, um valor que deve ser alterado a cada telegrama de enviado pelo escravo (*toggle bit*).

Caso o controlador programável PLC300 operando no modo escravo detecte um erro utilizando este mecanismo, ele irá automaticamente para o estado pré-operacional e indicará esta condição no LED CAN.

NOTA!
- Este objeto está ativo mesmo no estado parado (consulte a tabela 6.7).
- O valor 0 (zero) em um dos dois objetos desabilita esta função.
- Depois de detectado o erro, caso o serviço seja habilitado mais uma vez, a indicação do erro é retirada da HMI.
- O valor mínimo aceito para o controlador programável PLC300 é de 1ms. Mas levando-se em conta a taxa de transmissão e o número de pontos na rede, os tempos programados para essa função devem ser coerentes, de maneira que haja tempo suficiente para transmissão dos telegramas e também para que o resto da comunicação possa ser processada.
- Para cada escravo, somente um dos serviços – Heartbeat ou Node Guarding – pode ser habilitado.

6.5.3 Controle de Erros – Heartbeat

A detecção de erros através do mecanismo de *heartbeat* é feita utilizando dois tipos de objetos: o produtor *heartbeat* e o consumidor *heartbeat*. O produtor é responsável por enviar telegramas periódicos para a rede, simulando uma batida do coração, indicando que a comunicação está ativa e sem erros. Um ou mais consumidores podem monitorar estes telegramas periódicos e, caso estes telegramas deixem de ocorrer, significa que algum problema de comunicação ocorreu.

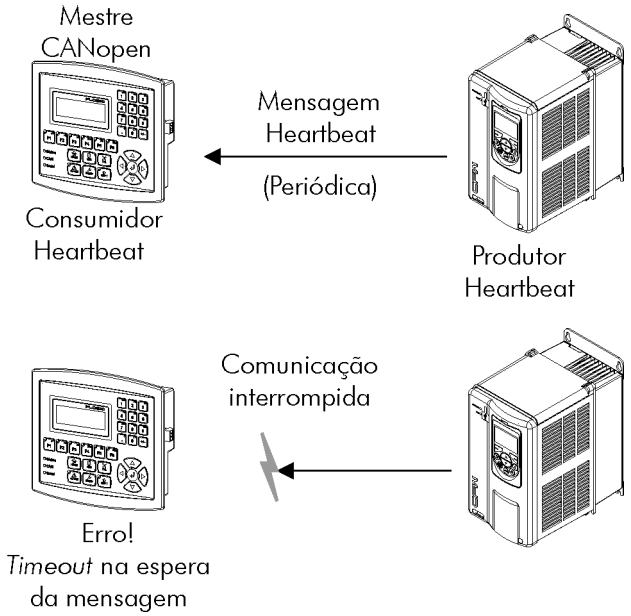


Figura 6.6: Serviço de controle de erros – Heartbeat

Um mesmo dispositivo da rede pode ser produtor e consumidor de mensagens *heartbeat*. Por exemplo, o mestre da rede pode consumir mensagens enviadas por um escravo, permitindo detectar problemas de comunicação com o escravo, e ao mesmo tempo o escravo pode consumir mensagens *heartbeat* enviadas pelo mestre, também possibilitando ao escravo detectar falhas na comunicação com o mestre.

O controlador programável PLC300 possui os serviços de produtor e consumidor *heartbeat*. Como consumidor, é possível programar diferentes produtores para serem monitorados pelo equipamento:

Índice	1016h
Nome	Consumer Heartbeat Time
Objeto	ARRAY
Tipo	UNSIGNED32

Sub-índice	0
Descrição	Número do último sub-índice
Acesso	ro
Mapeável	Não
Faixa	-
Valor Padrão	63

Sub-índices	1 – 63
Descrição	Consumer Heartbeat Time 1 – 63
Acesso	rw
Mapeável	Não
Faixa	UNSIGNED32
Valor Padrão	0

Nos sub-índices de 1 até 63, é possível programar o consumidor escrevendo um valor no seguinte formato:

UNSIGNED32		
Reservado (8 bits)	Node-ID (8 bits)	Heartbeat time (16 bits)

- ***Node-ID:*** permite programar o Node-ID do produtor heartbeat o qual se deseja monitorar.
- ***Heartbeat time:*** permite programar o tempo, em múltiplos de 1 milisegundo, até a detecção de erro, caso nenhuma mensagem do produtor seja recebida. O valor 0 (zero) neste campo desabilita o consumidor.

Depois de configurado, o consumidor *heartbeat* inicia a monitoração após o primeiro telegrama enviado pelo produtor. Caso seja detectado erro pelo fato do consumidor deixar de receber mensagens do produtor *heartbeat*, este irá automaticamente para o estado pré-operacional e indicará esta condição no LED CAN.

Como produtor, o controlador programável PLC300 possui um objeto para configuração deste serviço:

Índice	1017h
Nome	Producer Heartbeat Time
Objeto	VAR
Tipo	UNSIGNED16

Acesso	rw
Mapeável	Não
Faixa	UNSIGNED8
Valor Padrão	0

O objeto 1017h permite programar o tempo em milissegundos no qual o produtor envie um telegrama *heartbeat* para a rede. Uma vez programado, o dispositivo inicia a transmissão de mensagens com o seguinte formato:

Identificador	byte 1	
	bit 7	bit 6 ... bit 0
700h + Node-ID	Sempre 0	Estado do escravo

NOTA!
- Este objeto está ativo mesmo no estado parado (consulte a tabela 6.7).
- O valor 0 (zero) em um dos objetos desabilita esta função.
- Depois de detectado o erro, caso o serviço seja habilitado mais uma vez, a indicação do erro é retirada da HMI.
- O valor de tempo programado para o consumidor deve ser maior do que o programado para o respectivo produtor. Recomenda-se programar o consumidor com valores múltiplos do utilizado para o produtor.
- Para cada escravo, somente um dos serviços – Heartbeat ou Node Guarding – pode ser habilitado.

6.6 PROCEDIMENTO DE INICIALIZAÇÃO

Uma vez conhecido o funcionamento dos objetos disponíveis para o controlador programável PLC300 operando no modo escravo, é necessário agora programar os diferentes objetos para operarem em conjunto na rede. De forma geral, o procedimento para inicialização dos objetos em uma rede CANopen segue o descrito pelo fluxograma a seguir:

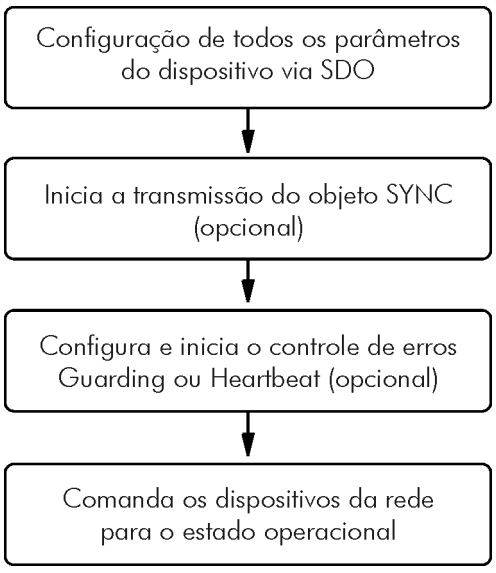


Figura 6.7: Fluxograma do processo de inicialização

É necessário observar que os objetos do controlador programável PLC300 não são armazenados na memória não volátil. Desta forma, sempre que for feito o reset ou desligado o equipamento, é necessário refazer a parametrização dos objetos de comunicação.

7 OPERAÇÃO NA REDE CANOPEN – MODO MESTRE

Além da operação como escravo, o controlador programável PLC300 também permite a operação como mestre da rede CANopen. A seguir serão descritas características e funções do PLC300 como mestre da rede CANopen.

7.1 HABILITAÇÃO DA FUNÇÃO CANOPEN MESTRE

Como padrão, o controlador programável PLC300 está programado para operar como escravo da rede CANopen. A programação do equipamento como mestre da rede deve ser feita utilizando o software WSCAN, que permite também a configuração de toda a rede CANopen. A descrição detalhada das janelas e funções do software WSCAN deve ser obtida no menu "Ajuda" do próprio software.

Depois de elaborada a configuração do mestre, é necessário fazer o download[7] das configurações, utilizando uma das interfaces de programação do produto – consulte o manual do usuário para maiores informações. Uma vez programado como mestre da rede, caso seja necessário apagar estas configurações, a função para apagar o programa do usuário – disponível no menu Setup – também apaga as configurações do mestre CANopen.

NOTA!
A rede CANopen é uma rede flexível e que permite várias formas de configuração e operação. No entanto, esta flexibilidade exige que o usuário tenha bom conhecimento das funções e objetos de comunicação utilizados para configuração da rede, bem como o conhecimento do software de programação WSCAN.

7.2 CARACTERÍSTICAS DO MESTRE CANOPEN

O controlador programável PLC300 permite controlar um conjunto de até 63 escravos, utilizando os seguintes serviços e recursos de comunicação:

- Serviço de gerenciador da rede (NMT)
- 63 PDOs de transmissão
- 63 PDOs de recepção
- 63 Consumidores Heartbeat
- Produtor Heartbeat
- Cliente SDO
- Produtor/consumidor SYNC
- 512 bytes de marcadores de rede de entrada (input)
- 512 bytes de marcadores de rede de saída (output)

As características físicas – instalação, conector, cabo, etc. – são as mesmas, tanto para o PLC300 operando como mestre quanto como escravo. As configurações de endereço e taxa de comunicação também são necessárias para a operação como mestre, mas estas configurações são programadas pelo software WSCAN de acordo com as propriedades definidas para o mestre no próprio software.

NOTA!
Os marcadores de rede de entrada são utilizados para mapear dados nos RPDOs, enquanto que os marcadores de rede de saída são utilizados para mapear dados nos TPDOs. Eles podem ser acessados em Byte (%IB ou %QB), Word (%IW ou %QW) ou Double Word (%ID ou %QD). Sua função, no entanto, não é pré-definida, e depende do aplicativo em ladder desenvolvido para o controlador PLC300.

7.3 OPERAÇÃO DO MESTRE

Uma vez programado para operar como mestre, o controlador programável PLC300 executará as seguintes etapas para realizar a inicialização, em seqüência, para cada um dos escravos:

7 Durante o download das configurações, a comunicação CANopen será desabilitada, sendo reiniciada ao término da operação.

- 1ª: enviado o comando de reset da comunicação para toda a rede, para que os escravos iniciem com valores conhecidos para os objetos de comunicação.
- 2ª: Identificação do equipamento na rede, através da leitura via SDO do objeto 1000h/00h – Object Identification.
- 3ª: Escrita via SDO de todos os objetos programados para o escravo, que usualmente inclui a configuração e mapeamento dos TPDOs e RPDOs, node guarding, heartbeat, além dos objetos específicos do fabricante, caso sejam programados.
- 4ª: Iniciado serviço de controle de erros – node guarding ou heartbeat – caso sejam programados.
- 5ª: envio do escravo para modo operacional.

Se uma destas etapas falhar, será indicado erro de comunicação com o escravo. Dependendo das configurações, a inicialização dos escravos será abortada, e o mestre fará a inicialização do escravo seguinte, retornando para o escravo com erro após tentar inicializar todos os demais escravos da rede.

De forma semelhante, se, durante a operação de um escravo, for identificado erro no serviço de controle de erros, dependendo das configurações feitas para o mestre, o escravo será automaticamente resetado e o procedimento de inicialização será executado novamente.

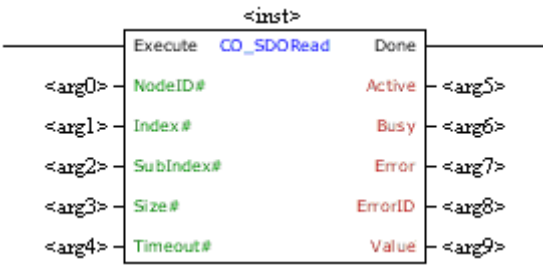
NOTA!
O estado da comunicação e o estado de cada escravo podem ser observados em marcadores de sistema de entrada.

7.4 BLOCOS PARA O MESTRE CANOPEN

Além dos objetos de comunicação e das configurações feitas no software WSCAN, também estão disponíveis blocos para monitoração e envio de comandos, que podem ser utilizados durante a elaboração do aplicativo em ladder para o controlador programável PLC300. Não é necessário utilizar estes blocos durante a operação do equipamento, mas seu uso confere maior flexibilidade e facilita o diagnóstico de problemas de comunicação durante a operação do controlador programável PLC300.

7.4.1 CANopen SDO Read – Leitura de Dados via SDO

Bloco para leitura de dados via SDO de um escravo remoto. Permite realizar a leitura de objetos na rede com tamanho de até 4 bytes.



Possui uma entrada de habilitação do bloco “Execute” e uma saída “Done” que é ativada após o término da execução com sucesso da função. Na transição positiva de “Execute”, quando o cliente SDO do mestre estiver livre, uma nova requisição é enviada para o servidor SDO do escravo. Ao término com sucesso da operação – resposta recebida do escravo – a saída “Done” é ativada, permanecendo ativa enquanto a entrada estiver ativa. Em caso de erro na execução da requisição, a saída “Error” é ativada, e o código do erro é colocado em “ErrorID”.

Entradas:

<arg0>: “NodeID#” – VAR_IN: inserir uma constante.
Tipos de dados: BYTE
Descrição: Endereço do escravo destino – 1 a 127.

<arg1>: “Index#” – VAR_IN: inserir uma constante.
Tipos de dados: WORD
Descrição: Índice do objeto acessado, dentre os objetos disponíveis no dicionário de objetos do escravo – 0 a 65535.

<arg2>: “SubIndex#” – VAR_IN: inserir uma constante.
Tipos de dados: BYTE
Descrição: Sub-índice do objeto acessado – 0 a 255.

<arg3>: “Size#” – VAR_IN: inserir uma constante.
Tipos de dados: BYTE
Descrição: Tamanho do dado acessado, em bytes – 1 a 4.

<arg4>: “Timeout#” – VAR_IN: inserir uma constante.
Tipos de dados: WORD
Descrição: Tempo de espera para chegada da resposta do escravo, a partir do início do envio pelo mestre – 5 a 5000 ms.

Saídas:

<arg5>: “Active” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Bloco ativo, requisição de leitura enviada para o escravo e aguardando resposta.
Nota: A variável tem que ter permissão de escrita.

<arg6>: “Busy” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Bloco habilitado, mas recurso não está disponível (cliente SDO enviando outra requisição), aguardando liberação para que a solicitação seja enviada pelo bloco. Se a entrada de habilitação for retirada enquanto o bloco faz esta indicação, a requisição é descartada.
Nota: A variável tem que ter permissão de escrita.

<arg7>: “Error” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Erro na execução da requisição.
Nota: A variável tem que ter permissão de escrita.

<arg8>: “ErrorID” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BYTE ou USINT
Descrição: Em caso de erro na requisição, indica o tipo de erro ocorrido. Resultados possíveis: 0= "Executado com sucesso"; 1= "Não pode executar a função" (exemplo: mestre não habilitado); 2= "Timeout na resposta do escravo"; 3= "Escravo retornou erro".
Nota: A variável tem que ter permissão de escrita.

<arg9>: “Value” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BYTE[1 ... 4] ou USINT[1 ... 4]
Descrição: Variável ou array onde serão salvos os dados lidos do escravo.
Nota: A variável tem que ter permissão de escrita.

7.4.2 CANopen SDO Write – Escrita de Dados via SDO

Bloco para escrita de dados via SDO para um escravo remoto. Permite realizar a escrita de objetos na rede com tamanho de até 4 bytes.

Possui uma entrada de habilitação do bloco “Execute” e uma saída “Done” que é ativada após o término da execução com sucesso da função. Na transição positiva de “Execute”, quando o cliente SDO do mestre estiver livre, uma nova requisição é enviada para o servidor SDO do escravo. Ao término com sucesso da operação – resposta recebida do escravo – a saída “Done” é ativada, permanecendo ativa enquanto a entrada estiver ativa. Em caso de erro na execução da requisição, a saída “Error” é ativada, e o código do erro é colocado em “ErrorID”.

Entradas:

<arg0>: “NodeID#” – VAR_IN: inserir uma constante.
Tipos de dados: BYTE
Descrição: Endereço do escravo destino – 1 a 127.

<arg1>: “Index#” – VAR_IN: inserir uma constante.
Tipos de dados: WORD
Descrição: Índice do objeto acessado, dentre os objetos disponíveis no dicionário de objetos do escravo – 0 a 65535.

<arg2>: “SubIndex#” – VAR_IN: inserir uma constante.
Tipos de dados: BYTE
Descrição: Sub-índice do objeto acessado – 0 a 255.

<arg3>: “Size#” – VAR_IN: inserir uma constante.
Tipos de dados: BYTE
Descrição: Tamanho do dado acessado, em bytes – 1 a 4.

<arg4>: “Timeout#” – VAR_IN: inserir uma constante.
Tipos de dados: WORD
Descrição: Tempo de espera para chegada da resposta do escravo, a partir do início do envio pelo mestre – 5 a 5000 ms.

<arg5>: “Value” – VAR_IN: inserir uma variável (tag).
Tipos de dados: BYTE[1 ... 4] ou USINT[1 ... 4]
Descrição: Variável ou array com dados para enviar para o escravo.

Saídas:

<arg6>: “Active” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Bloco ativo, requisição de escrita enviada para o escravo e aguardando resposta.
Nota: A variável tem que ter permissão de escrita.

<arg7>: “Busy” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Bloco habilitado, mas recurso não está disponível (cliente SDO enviando outra requisição), aguardando liberação para que a solicitação seja enviada pelo bloco. Se a entrada de habilitação for retirada enquanto o bloco faz esta indicação, a requisição é descartada.
Nota: A variável tem que ter permissão de escrita.

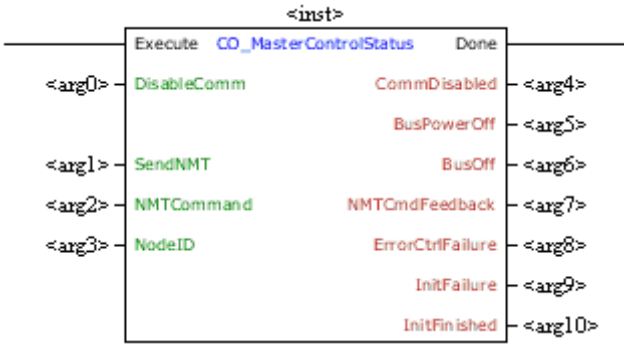
<arg8>: “Error” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Erro na execução da requisição.
Nota: A variável tem que ter permissão de escrita.

<arg9>: “ErrorID” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BYTE ou USINT
Descrição: Em caso de erro na requisição, indica o tipo de erro ocorrido. Resultados possíveis: 0= "Executado com sucesso"; 1= "Não pode executar a função" (exemplo: mestre não habilitado); 2= "Timeout na resposta do escravo"; 3= "Escravo retornou erro".
Nota: A variável tem que ter permissão de escrita.

NOTA!
- É importante que a quantidade de dados lidos ou escritos programado nos blocos seja compatível com o tamanho da variável ou do array com o valor.
- Em caso de erro retornado pelo escravo, é possível obter o código do último erro recebido através dos marcadores de sistema de leitura. Consulte o item 8 para a lista de marcadores disponíveis.

7.4.3 CANopen Master Control/Status – Controle e Estado do Mestre CANopen

Bloco para controle e monitoração do mestre da rede CANopen. Mostra o estado do mestre da rede para diagnóstico e identificação de problemas de comunicação, bem como permite o envio de comandos para os serviços de gerenciamento da rede – NMT.



Possui uma entrada de habilitação do bloco “Execute” e uma saída “Done” que é ativada após o término da execução da função. Se a entrada “Execute” está ativa, ele atualiza os valores das entradas e saídas e ativa a saída “Done”. Se a entrada “Execute” não estiver ativa, os demais valores das entradas são ignorados e todas as saídas são zeradas.

Entradas:

<arg0>: “DisableComm” – VAR_IN: inserir uma constante ou uma variável (tag).
Tipos de dados: BOOL
Descrição: Desabilita a comunicação CANopen. Ao desabilitar o mestre, os contadores e marcadores de status do mestre CANopem também são zerados – 0 ou 1.

<arg1>: “SendNMT” – VAR_IN: inserir uma constante ou uma variável (tag).
Tipos de dados: BOOL
Descrição: Na transição de deste sinal o mestre CANopen dispara o envio de um comando de gerenciamento – NMT – de acordo com o comando e o endereço programado neste bloco – 0 ou 1.

<arg2>: “NMTCommand” – VAR_IN: inserir uma constante ou uma variável (tag).
Tipos de dados: BYTE
Descrição: Indica qual comando deve ser enviado para o escravo: 1= "Start node"; 2= "Stop node"; 128= "Enter pre-operational"; 129= "Reset node"; 130= "Reset communication".

<arg3>: “NodeID” – VAR_IN: inserir uma constante ou uma variável (tag).
Tipos de dados: BYTE ou USINT
Descrição: Endereço do escravo para envio do comando NMT – 0= Broadcast (mensagem para todos os escravos); 1 a 127= Endereço específico do escravo.

Saídas:

<arg4>: “CommDisabled” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre e a comunicação na interface CAN foi desabilitada. É indicado sempre que o comando do usuário para desabilitar a interface for recebido, mas também é indicado nas situações de falta de alimentação na interface CAN ou bus off: 0= "Comunicação habilitada"; 1= "Comunicação desabilitada".
Nota: A variável tem que ter permissão de escrita.

<arg5>: “BusPowerOff” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que foi detectado falha na alimentação da interface CAN: 0= "Interface CAN alimentada"; 1= "Interface CAN sem alimentação".
Nota: A variável tem que ter permissão de escrita.

<arg6>: “BusOff” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que foi detectado erro de bus off na interface CAN: 0= "Sem erro de bus off"; 1= "Com erro de bus off".
Nota: A variável tem que ter permissão de escrita.

<arg7>: “NMTCmdFeedback” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o comando de gerenciamento foi enviado pelo mestre: 0= "Sem comando ou comando não enviado"; 1= "Comando NMT enviado".
Nota: A variável tem que ter permissão de escrita.

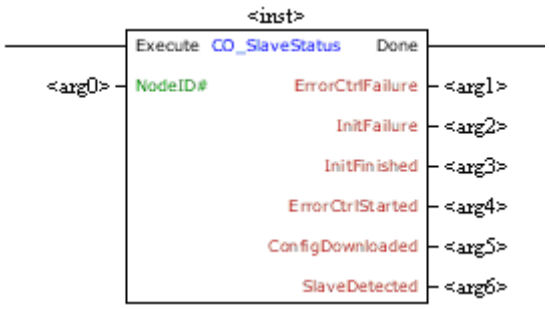
<arg8>: “ErrorCtrlFailure” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre detectou erro no serviço de controle de erros (node guarding ou heartbeat) em pelo menos um escravo da rede: 0= "Sem erro detectado"; 1= "Mestre detectou erro de node guarding ou heartbeat em pelo menos um escravo da rede".
Nota: A variável tem que ter permissão de escrita.

<arg9>: “InitFailure” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre detectou erro durante a inicialização de pelo menos um escravo da rede: 0= "Sem erro detectado"; 1= "Mestre detectou erro na inicialização em pelo menos um escravo da rede".
Nota: A variável tem que ter permissão de escrita.

<arg10>: “InitFinished” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre tentou fazer a inicialização de todos os escravos da rede. Não necessariamente a inicialização foi executada com sucesso, podem ter ocorrido erros durante a inicialização: 0= "Mestre ainda não executou o procedimento de inicialização de todos os escravos"; 1= "Mestre fez a inicialização (com ou sem sucesso) de todos os escravos".
Nota: A variável tem que ter permissão de escrita.

7.4.4 CANopen Slave Status –Estado do Escravo CANopen

Bloco para monitoração do escravo da rede CANopen. Mostra o estado do escravo da rede para diagnóstico e identificação de problemas de comunicação.



Possui uma entrada de habilitação do bloco “Execute” e uma saída “Done” que é ativada após o término da execução da função. Se a entrada “Execute” está ativa, ele atualiza os valores das entradas e saídas e ativa a saída “Done”. Se a entrada “Execute” não estiver ativa, os demais valores das entradas são ignorados e todas as saídas são zeradas.

Entradas:

<arg0>: “NodeID” – VAR_IN: inserir uma constante ou uma variável (tag).
Tipos de dados: BYTE ou USINT
Descrição: Endereço do escravo para identificação do estado da comunicação com o mestre – 1 a 127.

Saídas:

<arg1>: “ErrorCtrlFailure” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre detectou erro no serviço de controle de erros (node guarding ou heartbeat) no escravo indicado: 0= "Sem erro detectado"; 1= "Mestre detectou erro de node guarding ou heartbeat no escravo".
Nota: A variável tem que ter permissão de escrita.

<arg2>: “InitFailure” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre detectou erro durante a inicialização do escravo indicado: 0= "Sem erro detectado"; 1= "Mestre detectou erro na inicialização no escravo".
Nota: A variável tem que ter permissão de escrita.

<arg3>: “InitFinished” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre fez a inicialização completa e com sucesso do escravo indicado: 0= "Mestre não concluiu o procedimento de inicialização do escravo"; 1= "Mestre fez a inicialização do escravo com sucesso".
Nota: A variável tem que ter permissão de escrita.

<arg4>: “ErrprCtrlStarted” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre iniciou o serviço de controle de erros (node guarding ou heartbeat) com o escravo indicado. Se este serviço não for habilitado para o escravo, este bit será ativado após realizar a configuração: 0= "Controle de erros com o escravo não iniciado"; 1= "Controle de erros com o escravo iniciado".
Nota: A variável tem que ter permissão de escrita.

<arg5>: “ConfigDownloaded” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre finalizou com sucesso o download das configurações via SDO para o escravo indicado: 0= "Mestre não finalizou o download das configurações para o escravo"; 1= "Download das configurações para o escravo finalizado com sucesso".
Nota: A variável tem que ter permissão de escrita.

<arg6>: “SlaveDetected” – VAR_OUT: inserir uma variável (tag).
Tipos de dados: BOOL
Descrição: Indica que o mestre conseguiu fazer a leitura da identificação via SDO do escravo indicado: 0= "Escravo não foi contactado"; 1= "Escravo contactado com sucesso".
Nota: A variável tem que ter permissão de escrita.

NOTA!
Os dados acessados utilizando este bloco também estão disponíveis através de marcadores de sistema de leitura e escrita, conforme descrito no item 8.

8 MARCADORES DE SISTEMA PARA CAN/CANOPEN

Para interface CAN e comunicação CANopen, foram disponibilizados os seguintes marcadores de sistema de leitura (%S) e marcadores de sistema de escrita (%C), para controle e monitoração desta interface:

8.1 MARCADORES DE SISTEMA DE LEITURA

Estado da Interface CAN: conjunto de marcadores de leitura para indicar informações sobre o estado da interface CAN.	
Marcador	**Descrição**
%SB3150	Estado da interface CAN: 0 = Inicializando 1 = Reservado 2 = Interface Habilitada 3 = Warning 4 = Error Passive 5 = Bus Off 6 = Sem alimentação
%SB3151	Alimentação da interface CAN. 0 = Sem alimentação 1 = Com alimentação
%SW3152	Número de telegramas recebidos. É zerado sempre que atingir o limite máximo ou quando a interface for desabilitada.
%SW3154	Número de telegramas transmitidos. É zerado sempre que atingir o limite máximo ou quando a interface for desabilitada.
%SW3156	Contador de erros de Bus Off detectados.
%SW3158	Contador de mensagens CAN perdidas (overrun).

Estado da Comunicação CANopen: marcadores de leitura para indicar informações sobre o estado da comunicação CANopen.	
Marcador	**Descrição**
%SB3180	Estado do protocolo CANopen. Indica se a interface está operando corretamente e, no caso da operação como escravo da rede, se foi detectado algum erro nos mecanismos de detecção de erros do protocolo CANopen: 0 = Desabilitado 1 = Reservado 2 = CANopen habilitado 3 = Controle de erros habilitado 4 = Com erro de node guarding 5 = Com erro de heartbeat
%SB3181	Estado do nó CANopen, de acordo com a máquina de estados para escravos da rede CANopen: 0 = Inicializando 1 = Parado 2 = Operacional 3 = Pré-operacional

Estado do Mestre e Escravos CANopen: conjunto de marcadores de leitura para indicar informações sobre o estado geral do mestre CANopen e o estado da comunicação entre o mestre e cada um dos escravos.	
Marcador	**Descrição**
%SW3200	Estado do mestre CANopen: Bit 0: todos os escravos forma contatados. Bit 1: download das configurações dos escravos realizada. Bit 2: controle de erros dos escravos iniciado. Bit 3: fim da inicialização dos escravos. Bit 4: detectado erro na inicialização de pelo menos um escravo. Bit 5: detectado erro no serviço de controle de erros de pelo menos um escravo. Bits 6 e 7: reservado Bit 8: assume o valor do toogle bit (ver %CD3120) após o mestre enviar comando NMT. Bits 9 ... 12: reservado Bit 13: interface CAN no estado de bus off. Bit 14: sem alimentação na interface CAN. Bit 15: comunicação desabilitada.
%SW3202 ... %SW3454	Estado dos escravos CANopen. São 127 marcadores de Word, onde cada marcador está associado a um endereço na rede CANopen, e indica o estado do escravo no endereço: Bit 0: mestre contatou escravo com sucesso. Bit 1: download das configurações do escravo realizada com sucesso. Bit 2: controle de erros do escravo iniciado. Bit 3: fim da inicialização do escravo. Bit 4: detectado erro na inicialização do escravo. Bit 5: detectado erro no serviço de controle de erros do escravo. Bits 6 ... 15: reservado

Último Erro no Cliente SDO: conjunto de marcadores de leitura para informar dados sobre erros no cliente SDO. Caso alguma requisição seja feita ao cliente SDO e o escravo não responder, ou responder com erro, os dados relativos ao último erro detectado pelo cliente SDO são salvos nestes marcadores.	
Marcador	**Descrição**
%SW3460	Endereço do escravo destino, para o qual a requisição SDO foi enviada.
%SW3462	Índice do objeto acesso via SDO.
%SW3464	Sub-índice do objeto acessado.
%SW3466	Tipo de acesso realizado: 1 = leitura, 2 = escrita.
%SD3468	Para acessos de escrita, indica o valor escrito.
%SD3472	Indica o código do erro recebido, de acordo com os erros de comunicação via SDO da especificação do protocolo CANopen.

Último EMCY detectado: conjunto de marcadores de leitura para informar dados sobre erros reportados por produtores EMCY. O mestre CANopen do controlador PLC300 não possui consumidor EMCY. Telegramas EMCY enviados por escravos da rede, no entanto, são capturados pelo mestre, e as informações do último EMCY detectado são salvas nestes marcadores.	
Marcador	**Descrição**
%SB3480	Endereço do escravo que reportou o EMCY.
%SB3481	Reservado.
%SB3482 ... %SB3489	Oito bytes de dados do telegrama EMCY, com informações sobre o código do erro reportado pelo escravo.

8.2 MARCADORES DE SISTEMA DE ESCRITA

Configuração da Interface CAN: conjunto de marcadores de escrita para programar as configurações da interface CAN. Também são acessíveis através do menu Setup.	
Marcador	**Descrição**
%CB3052	Endereço CANopen (Node ID). Valores válidos 1 a 127.
%CB3053	Reservado.
%CB3054	Reservado.
%CB3055	Taxa de comunicação CAN: 0 = 1 Mbit/s 1 = reservado 2 = 500 Kbit/s 3 = 250 Kbit/s 4 = 125 Kbit/s 5 = 100 Kbit/s 6 = 50 Kbit/s 7 = 20 Kbit/s

Controle do Mestre CANopen: conjunto de marcadores de escrita para controlar o mestre CANopen.	
Marcador	**Descrição**
%CD3120	Comando para controle do mestre CANopen e envio de telegrama NMT. Bits 0 ... 7: código do comando NMT: 1 = START 2 = STOP 128 = ENTER PRE-OPERATIONAL 129 = RESET NODE 130 = RESET COMMUNICATION Bit 8: toggle bit, sempre que o valor deste bit for alterado envia o comando programado. Bits 9 ... 14: reservado Bit 15: desabilita comunicação CANopen Bits 16 ... 23: endereço do escravo destino para envio do comando NMT. Bits 24 ... 31: reservado

9 FALHAS E ALARMES RELACIONADOS COM A COMUNICAÇÃO CANOPEN

SEM ALIMENTAÇÃO NA INTERFACE CAN

Descrição:
Indica que a interface CAN não possui alimentação entre os pinos 1 e 5 do conector.

Atuação:
Para que seja possível enviar e receber telegramas através da interface CAN, é necessário fornecer alimentação externa para o circuito de interface.

Se a interface CAN estiver alimentada e for detectada a falta de alimentação na interface CAN, será sinalizada através do LED CAN e nos marcadores da rede CANopen. Se a alimentação do circuito for restabelecida, a comunicação CAN será reiniciada.

Possíveis Causas/Correção:
- Medir se existe tensão dentro da faixa permitida entre os pinos 1 e 5 do conector da interface CAN.
- Verificar se os cabos de alimentação não estão trocados ou invertidos.
- Verificar problemas de contato no cabo ou no conector da interface CAN.

BUS OFF

Descrição:
Detectado erro de *bus off* na interface CAN.

Atuação:
Caso o número de erros de recepção ou transmissão detectados pela interface CAN seja muito elevado[8], o controlador CAN pode ser levado ao estado de *bus off*, onde ele interrompe a comunicação e desabilita a interface CAN.

Neste caso será sinalizada através do LED CAN e nos marcadores da rede CANopen. Para que a comunicação seja restabelecida, é necessário desligar e ligar novamente o produto, ou retirar e ligar novamente a alimentação da interface CAN, para que a comunicação seja reiniciada.

Possíveis Causas/Correção:
- Verificar curto-circuito nos cabos de transmissão do circuito CAN.
- Verificar se os cabos não estão trocados ou invertidos.
- Verificar se todos os dispositivos da rede utilizam a mesma taxa de comunicação.
- Verificar se resistores de terminação com valores corretos foram colocados somente nos extremos do barramento principal.
- Verificar se a instalação da rede CAN foi feita de maneira adequada.

NODE GUARDING/HEARTBEAT

Descrição:
Controle de erros da comunicação CANopen detectou erro de comunicação utilizando o mecanismo de *guarding*.

Atuação:
Utilizando os mecanismos de controle de erro – *Node Guarding* ou *Heratbeat* – o mestre e o escravo podem trocar telegramas periódicos, em um período pré-determinado. Caso a comunicação seja interrompida por algum motivo, tanto mestre quanto escravo poderão detectar erro na comunicação pelo *timeout* na troca destas mensagens.

Neste caso será sinalizada através do LED CAN e nos marcadores da rede CANopen.

[8] Para mais informações sobre detecção de erros, consultar especificação CAN.

Possíveis Causas/Correção:

- Verificar os tempos programados no mestre e no escravo para troca de mensagens. Para evitar problemas devido a atrasos na transmissão e diferenças na contagem dos tempos, recomenda-se que os valores programados para detecção de erros pelo escravo sejam múltiplos dos tempos programados para a troca de mensagens no mestre.
- Verificar se o mestre está enviando os telegramas de *guarding* no tempo programado.
- Verificar problemas na comunicação que possam ocasionar perda de telegramas ou atrasos na transmissão.

WEG Equipamentos Elétricos S.A.
Jaraguá do Sul – SC – Brasil
Fone 55 (47) 3276-4000 – Fax 55 (47) 3276-4020
São Paulo – SP – Brasil
Fone 55 (11) 5053-2300 – Fax 55 (11) 5052-4212
automacao@weg.net
www.weg.net