



Servoconversor SCA-05

Manual da Comunicação

CANopen Slave

05/2005

Série: SCA-05
Versão: P/2

Sumário

Lista de Tabelas	5
Lista de Figuras	5
Sobre o manual	6
Abreviações e Definições	6
Documentos	6
1 Introdução ao protocolo CANopen	7
1.1 CAN	7
1.1.1 Frame de dados	7
1.1.2 Frame remoto	7
1.1.3 Acesso à rede	7
1.1.4 Controle de erros	8
1.1.5 CAN e CANopen	8
1.2 Características da rede CANopen	8
1.3 Meio físico	9
1.4 Endereço na rede CANopen	9
1.5 Acesso aos dados	9
1.6 Transmissão de dados	9
1.7 Objetos responsáveis pela comunicação - COBs	10
1.8 COB-ID	11
1.9 Arquivo EDS	12
2 Instalação	13
2.1 Conexão com a rede	13
2.2 Fonte de alimentação	14
2.3 Cabos e terminações	14
2.4 Taxa de comunicação	15
3 Parâmetros da comunicação CANopen	17
3.1 P070 - Estado do controlador CAN	17
3.2 P071 - Número de telegramas CAN recebidos	17
3.3 P072 - Número de telegramas CAN transmitidos	18
3.4 P073 - Número de erros de bus-off registrados	18
3.5 P075 - Estado da rede CANopen	18
3.6 P076 - Estado do nó CANopen	18
3.7 P313 - Ação para erro de comunicação	19
3.8 P700 - Protocolo CAN	19
3.9 P701 - Endereço na rede CAN	20
3.10 P702 - Taxa de comunicação	20
3.11 P703 - Reset de <i>bus off</i>	21
4 Dicionário de objetos	22
4.1 Estrutura do dicionário	22
4.2 Tipos de dados	23
4.2.1 Tipos básicos	23
4.2.2 Tipos compostos	23
4.2.3 Tipos estendidos	24

4.3	Communication Profile - Objetos para comunicação	24
4.4	Manufacturer Specific - Objetos específicos do SCA-05	25
4.5	Device Profile - Objetos comuns para drives	26
5	Descrição dos objetos de comunicação	27
5.1	Objetos de identificação	27
5.1.1	Objeto 1000h - Device Type	27
5.1.2	Objeto 1001h - Error Register	27
5.1.3	Objeto 1003h - Pre-defined error field	28
5.1.4	Objeto 1018h - Identity object	29
5.2	Service Data Objects - SDOs	30
5.2.1	Objeto 1200h - Servidor SDO	31
5.2.2	Funcionamento dos SDOs	31
5.3	Process Data Objects - PDOs	33
5.3.1	Objetos mapeáveis para os PDOs	34
5.3.2	PDOs de recepção	35
5.3.3	PDOs de transmissão	39
5.4	Emergency Object - EMCY	43
5.5	Synchronization Object - SYNC	43
5.6	Network Management - NMT	44
5.6.1	Controle dos estados do escravo	45
5.6.2	Controle de Erros - Guarding	46
5.7	Procedimento de inicialização	48
6	Descrição dos objetos para drives	50
6.1	Device Control - Objetos para controle do drive	52
6.1.1	Objeto 6040h - Controlword	54
6.1.2	Objeto 6041h - Statusword	55
6.1.3	Objeto 6060h - Modes of Operation	56
6.1.4	Objeto 6061h - Modes of Operation Display	57
6.2	Factor Group - Objetos para conversão de unidades	58
6.2.1	Objeto 6089h - Position Notation Index	58
6.2.2	Objeto 608Ah - Position Dimension Index	58
6.2.3	Objeto 608Bh - Velocity Notation Index	59
6.2.4	Objeto 608Ch - Velocity Dimension Index	59
6.2.5	Objeto 608Dh - Acceleration Notation Index	60
6.2.6	Objeto 608Eh - Acceleration Dimension Index	60
6.3	Position Control Function - Controlador de Posição	61
6.3.1	Objeto 6063h - Position Actual Value	61
6.3.2	Objeto 6064h - Position Actual Value in user units	61
6.4	Profile Position Mode - Modo Posicionamento	62
6.4.1	Objeto 607Ah - Target Position	62
6.4.2	Objeto 6083h - Profile Acceleration	62
6.4.3	Objeto 6084h - Profile Deceleration	62
6.4.4	Objeto 6086h - Motion Profile Type	63
6.5	Profile Velocity Mode - Modo Velocidade	64
6.5.1	Bits de controle e estado	64
6.5.2	Objeto 6069h - Velocity Sensor Actual Value	64
6.5.3	Objeto 606Bh - Velocity Demand Value	65
6.5.4	Objeto 606Ch - Velocity Actual Value	65

6.5.5	Objeto 60FFh - Target Velocity	65
6.6	Profile Torque Mode - Modo Torque	66
6.6.1	Bits de controle e estado	66
6.6.2	Objeto 6071h - Target Torque	66
6.6.3	Objeto 6077h - Torque actual value	66
6.6.4	Objeto 6087h - Torque Slope	67
6.6.5	Objeto 6088h - Torque Profile Type	67
7	Erros da comunicação CANopen	68
7.1	E33 - Sem alimentação	68
7.2	E34 - <i>Bus off</i>	68
7.3	E35 - Erro de guarda do nó	68
8	Exemplos de utilização da interface CANopen	69
8.1	Exemplo 1 - Controlando o estado de um dispositivo	70
8.2	Exemplo 2 - Acessando objetos utilizando SDOs	71
8.3	Exemplo 3 - Habilitando o controle de erros - <i>Guarding</i>	71
8.4	Exemplo 4 - Configurando um PDO para transmissão	73
8.5	Exemplo 5 - Configurando um PDO para recepção	75
8.6	Exemplo 6 - Utilizando o objeto SYNC para envio de PDOs	76
8.7	Exemplo 7 - Detecção de erros utilizando EMCY	78

Lista de Tabelas

1	Documentação técnica sobre CANopen	6
2	Tipos de objetos de comunicação - COBs	11
3	COB-ID para os diferentes objetos	12
4	Pinagem do conector utilizado pelo SCA-05	14
5	Dados para alimentação via rede	14
6	Características do cabo	15
7	Taxas de comunicação suportadas e tamanho da instalação	16
8	Agrupamentos do dicionário de objetos	22
9	<i>Record</i> para configuração dos PDOs	23
10	<i>Record</i> para mapeamento dos dados de um PDO	24
11	<i>Record</i> para configuração dos SDOs	24
12	<i>Record</i> para identificação do dispositivo	24
13	Lista de objetos do SCA-05 - Communication Profile	25
14	Lista de objetos do SCA-05 - Manufacturer Specific	26
15	Estrutura do objeto Error Register	28
16	Tabela de códigos de erro	29
17	Código dos comandos para cliente SDO	32
18	Código dos comandos para servidor SDO	32
19	Lista de parâmetros mapeáveis para PDOs	35
20	Descrição do COB-ID	36
21	Descrição do tipo de transmissão	36
22	Descrição das transições	45
23	Objetos acessíveis em cada estado	46
24	Comandos recebidos pelo escravo	46
25	Lista de objetos do SCA-05 - Drive Profile	51
26	Comandos da palavra de controle	54
27	Bits da palavra de controle específicos do modo de operação	55
28	Estados do servo indicados através da palavra de estado	56
29	Bits da palavra de estado específicos do modo de operação	56

Lista de Figuras

1	Modelo do funcionamento do SCA-05 na rede CANopen	11
2	Posição do conector CAN no SCA-05	13
3	Exemplo de conectores	13
4	Rede CANopen	15
5	Comunicação entre Cliente e Servidor SDO	31
6	Comunicação utilizando PDOs	34
7	EMCY	43
8	SYNC	44
9	Diagrama de estados do nó CANopen	45
10	Serviço de controle de erros - <i>Guarding</i>	47
11	Fluxograma do processo de inicialização	49
12	Arquitetura de comunicação para um drive na rede CANopen	50
13	Máquina de estados do SCA-05	52
14	Rede utilizada nos exemplos de aplicação	69

Sobre o manual

Este documento descreve o funcionamento do protocolo CANopen para o servoconversor SCA-05. Ele deve ser utilizado em conjunto com o manual do servoconversor SCA-05.

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
EMCY	Emergency Object
SYNC	Synchronization Object
NMT	Network Management Object
ASCII	American Standard Code for Information Interchange
ro	Read only (somente leitura)
rw	Read/write (leitura e escrita)
const	Constante



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 para SCA-05 foi desenvolvido baseado nas seguintes especificações e documentos:

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

Tabela 1: Documentação técnica sobre CANopen

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

1 Introdução ao protocolo CANopen

Para a operação do servoconversor SCA-05 em rede CANopen, é necessário conhecer a forma como a comunicação é feita. Para isto, este item traz uma descrição geral do funcionamento do protocolo CANopen, contendo as funções utilizadas pelo SCA-05. Para uma descrição detalhada do protocolo, consulte a documentação CANopen indicada na tabela 1.

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 arbitragem de mensagens.

1.1.1 Frame de dados

Os dados em uma rede CAN são transmitidos através de um frame (telegrama) de dados. Este tipo de frame é composto principalmente por um campo identificador de 11 bits² (*arbitration field*), e um campo de dados (*data field*), que pode conter até 8 bytes de dados.

<i>Identificador</i>	<i>8 bytes de dados</i>							
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 da rede 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.

¹Na especificação do protocolo CAN, é referenciada a norma ISO 11898 como definição da camada 1 deste modelo (camada física).

²A especificação CAN 2.0 define dois tipos de frames de dados: *standard* (11bits) e *extended* (29 bits). Para o protocolo CANopen do SCA-05, somente frames *standard* são aceitos.

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 os 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. Caso ocorra uma quantidade considerável de erros, o dispositivo pode ser levado para os seguintes estados:

- *Warning*: quando esse contador passa de um determinado limite, o dispositivo entra no estado de *warning*, significando a ocorrência de uma elevada taxa de erros.
- *Error Passive*: quando este valor ultrapassa um limite maior, ele entra no estado de *error passive*, onde ele pára de atuar na rede ao detectar que um outro dispositivo enviou um telegrama com erro.
- *Bus Off*: por último, temos o estado de *bus off*, no qual o dispositivo não irá mais enviar ou receber telegramas.

1.1.5 CAN e CANopen

Somente a especificação de como detectar erros, criar e transmitir um frame não é suficiente 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, e 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 outros protocolos baseados em CAN, como DeviceNet, J1939, etc., que também 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 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.

O servoconversor SCA-05 utiliza ainda um circuito de interface com a rede isolado com alimentação externa. O componente responsável pela transmissão e recepção de sinais é denominado transceiver, que obedece o especificado pela ISO 11898.

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.

O SCA-05 não possui funções que implementam os serviços de gerenciamento de rede, e portanto ele deve ser utilizado em conjunto com algum equipamento que possua tais serviços.

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.

Uma descrição mais detalhada de como o dicionário de objetos está estruturado é fornecida no item 4.

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.

<i>Identificador</i>	<i>6 bytes de dados</i>					
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. O SCA-05 suporta os seguintes objetos de comunicação (COBs):

<i>Tipo de objeto</i>	<i>Descrição</i>
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: <i>Cliente SDO</i> , responsável por fazer uma requisição leitura ou escrita para um dispositivo da rede, e o <i>Servidor SDO</i> , 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. Somente um SDO do tipo servidor está disponível para o SCA-05.
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: <i>PDO de recepção</i> e <i>PDO de transmissão</i> . 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 (<i>Produtor EMCY</i>), este pode enviar uma mensagem para a rede. Caso algum dispositivo da rede esteja monitorando esta mensagem (<i>Consumidor EMCY</i>), é possível programar para que uma ação seja tomada (desabilitar demais dispositivos da rede, <i>reset</i> de erros, etc.). O SCA-05 possui apenas a funcionalidade de produtor EMCY.
Synchronisation Object (SYNC)	Na rede CANopen é possível programar um dispositivo (<i>Produtor SYNC</i>) para enviar, periodicamente, uma mensagem de sincronização para todos os dispositivos da rede. Estes dispositivos (<i>Consumidores SYNC</i>) podem então, por exemplo, enviar um determinado dado que necessita ser disponibilizado periodicamente. O SCA-05 possui a função de consumidor SYNC.

<i>Tipo de objeto</i>	<i>Descrição</i>
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 então são responsáveis por receber os comandos enviados pelo mestre e executar as ações solicitadas. O SCA-05 opera como um escravo da rede CANopen, e disponibiliza dois tipos de serviços que o mestre pode utilizar: serviços de controle do dispositivo, onde o mestre controla o estado de cada escravo na rede, e serviços de controle de erros (guarda do nó - <i>Guarding</i>), onde tanto o mestre quanto o escravo trocam telegramas periódicos para checarem se não há erros na comunicação.

Tabela 2: Tipos de objetos de comunicação - COBs

Toda a comunicação do servo com a rede é feita utilizando-se estes objetos, e os dados que podem ser acessados são os existentes no dicionário de objetos do dispositivo. Uma descrição mais detalhada do funcionamento de cada COB é feita na seção 5. O modelo do funcionamento do servoconversor SCA-05, do ponto de vista da interface com a rede CANopen, pode ser descrito pela seguinte figura:

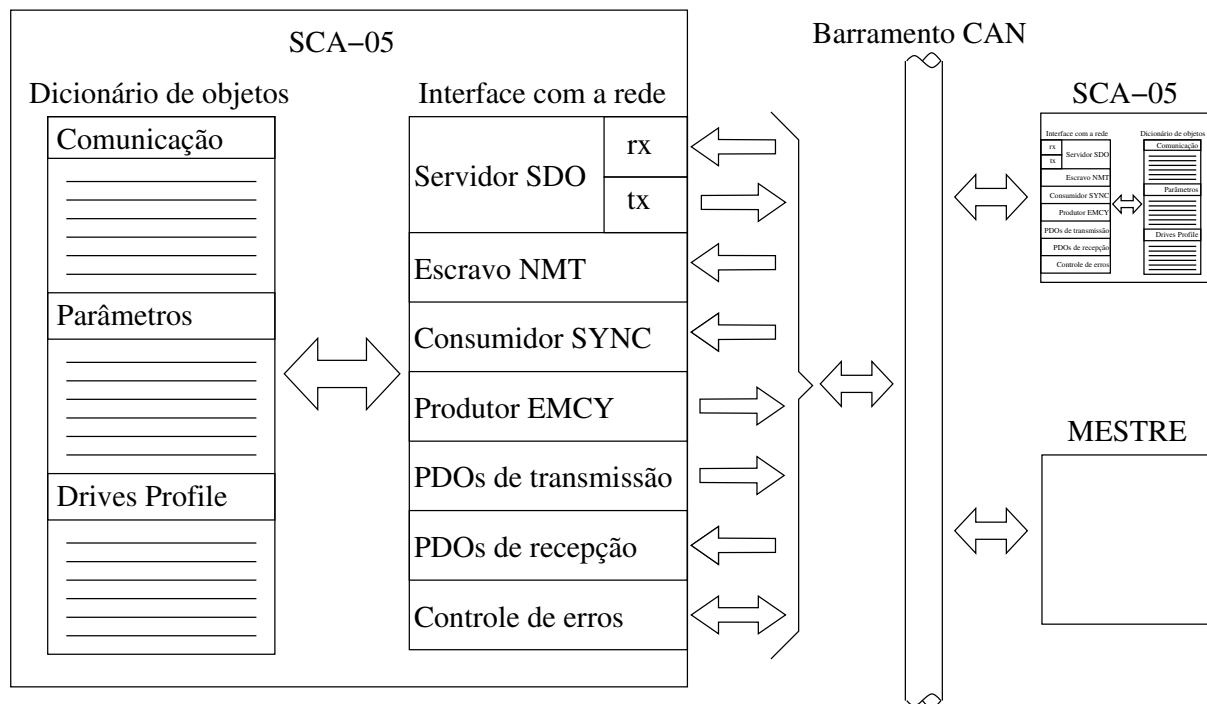


Figura 1: Modelo do funcionamento do SCA-05 na rede CANopen

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:

<i>Código da Função</i>				<i>Endereço do nó</i>						
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 disponíveis no servo. 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.

COB	<i>Código da Função</i> (bits 10 - 7)	<i>COB-ID Resultante</i> (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	1110	1793 - 1919 (701h - 77Fh)

Tabela 3: COB-ID para os diferentes objetos

1.9 Arquivo EDS

Cada dispositivo em uma rede CANopen possui um arquivo de configuração EDS, que contém diversas 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 para o SCA-05 é fornecido juntamente com o produto, e também pode ser obtido através do *site* <http://www.weg.com.br>. É necessário observar a versão de software do servo, para utilizar um arquivo EDS que seja compatível com esta versão.

2 Instalação

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 instalação deve seguir o descrito pela norma ISO 11898, porém a seguir são apresentadas recomendações para realizar a instalação do servoconversor SCA-05.

2.1 Conexão com a rede

A interface para conexão com o barramento CAN está disponível no conector X5. A seguir é apresentado um modelo utilizando um SCA-05.

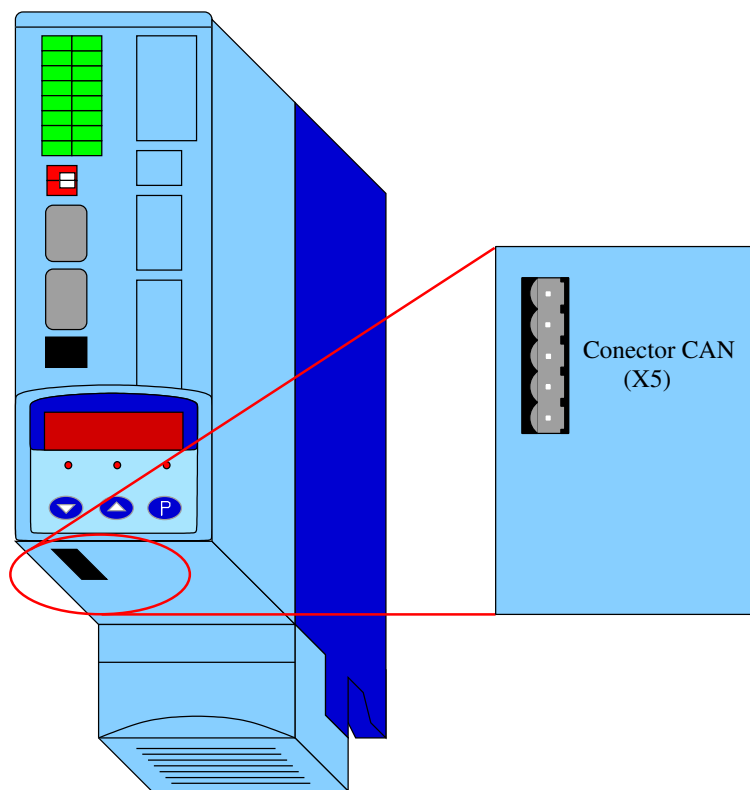


Figura 2: Posição do conector CAN no SCA-05

A tabela 4 descreve a função de cada pino do conector.

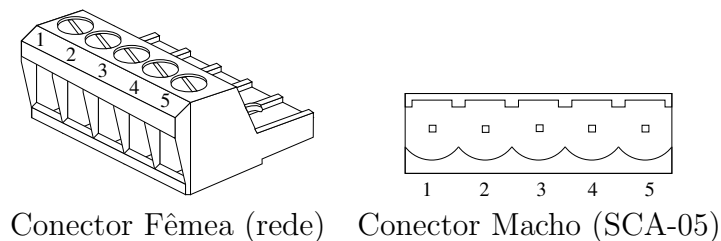


Figura 3: Exemplo de conectores

Conector X5	
1	V-
2	CAN_L
3	Shield
4	CAN_H
5	V+

Tabela 4: Pinagem do conector utilizado pelo SCA-05

A ligação deve ser feita conectando cada um dos sinais (V- conectado ao V-, CAN_L conectado ao CAN_L, etc.) nos diversos equipamentos ligados em rede. Caso algum outro equipamento não necessite de alimentação via rede, basta não conectar os pontos V- e V+. A blindagem do cabo deve ser ligada no pino 3, que internamente é conectado ao terra do dispositivo através de um circuito RC.

2.2 Fonte de alimentação

Para alimentar o circuito responsável pela comunicação no SCA-05, é necessário fornecer uma tensão de alimentação entre os pinos 1 e 5 do conector da rede. Para evitar problemas de diferença de tensão 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 necessário mais de uma fonte de alimentação, estas devem estar referenciadas ao mesmo ponto. Os dados para consumo individual e tensão de entrada são apresentados na tabela a seguir.

Tensão de alimentação (V_{CC})		
Mínimo	Máximo	Recomendado
11	30	24
Corrente (mA)		
Mínimo	Máximo	Médio
20	50	30

Tabela 5: Dados para alimentação via rede

O próprio SCA-05 possui uma fonte de alimentação de 24V que pode ser utilizada para alimentação da interface CAN. Porém, é necessário observar a capacidade de corrente fornecida por esta fonte (consulte o manual do SCA-05), para definir quantos equipamentos podem ser alimentados.



NOTA!

Esta alimentação é utilizada apenas pelos os componentes isolados eletricamente do restante do circuito de controle do SCA-05. Para que o servo possa se comunicar com o barramento, é necessário que a alimentação principal também esteja ligada.

2.3 Cabos e terminações

É recomendado a utilização de um cabo blindado com dois pares trançados - um par para os pinos 2 e 4 (CAN_L e CAN_H) e outro para os pinos 1 e 5 (V- e V+).

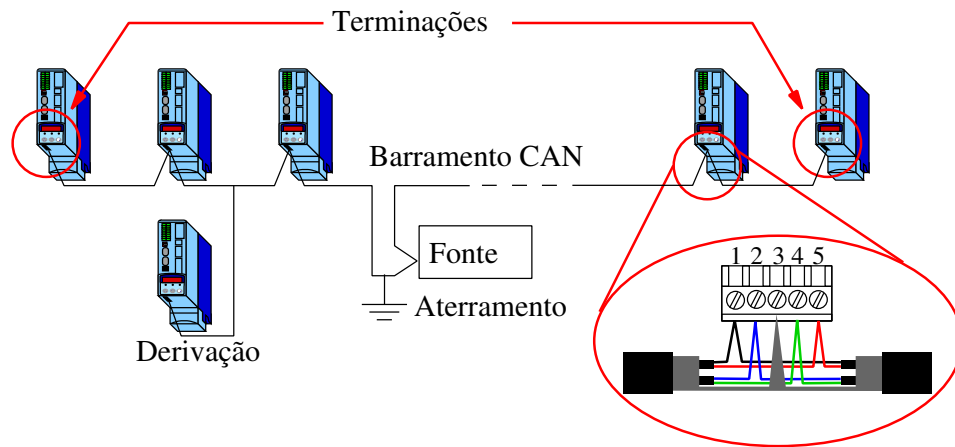


Figura 4: Rede CANopen

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 próximo a cabos de potência, pois devido a interferência eletromagnética, isto facilita a ocorrência de erros durante a transmissão. 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.

O cabo para a ligação dos sinais CAN_L e CAN_H deve ter impedância característica de aproximadamente 120Ω, e um atraso máximo de propagação do sinal de 5ns/m. Outras características dependem do comprimento do cabo, que deve estar de acordo com a tabela a seguir.

<i>Comprimento do cabo</i> (m)	<i>Resistência por metro</i> (mΩ/m)	<i>Área do condutor</i> (mm ²)
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

Tabela 6: Características do cabo

As extremidades do barramento CAN devem possuir um resistor de terminação no valor de 120Ω / 0.25W, conectando os sinais CAN_H e CAN_L. Caso o SCA-05 seja o elemento de início ou fim do barramento, este resistor pode ser ligado diretamente conector da rede, entre os pinos 2 e 4.

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.

2.4 Taxa de comunicação

A taxa de comunicação que pode ser utilizada por um equipamento na rede CANopen depende do comprimento do cabo utilizado na instalação. A tabela a seguir mostra as taxas de comunicação disponíveis para o SCA-05, e o comprimento máximo de cabo que pode ser utilizado na instalação de acordo com o recomendado pela CiA.

<i>Taxa de comunicação</i>	<i>Comprimento do cabo</i>
1 Mbit/s	40 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

Tabela 7: Taxas de comunicação suportadas e tamanho da instalação

3 Parâmetros da comunicação CANopen

O SCA-05 possui um conjunto de parâmetros, descritos a seguir, para a configuração do dispositivo na rede, e também para diagnóstico e monitoração dos estados do servoconversor.

Os demais parâmetros não citados aqui não possuem relação direta com comunicação CANopen, porém são importantes para a operação do servoconversor. Desta forma, deve-se saber como utilizar o servo via parâmetros, pois os mesmos também poderão ser utilizados durante sua operação via rede CANopen. Deve-se consultar o manual do SCA-05 para a lista completa de parâmetros e sua descrição.

3.1 P070 - Estado do controlador CAN

Fornece a informação do estado do dispositivo com relação ao barramento CAN. Ele irá indicar se o controlador está operando corretamente, ou então informará o tipo de erro que o servo apresenta com relação à comunicação.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 = Comunicação não está habilitada 1 = Reservado 2 = Habilitado e sem erro 3 = <i>Warning</i> 4 = <i>Error Passive</i> 5 = <i>Bus Off</i> 6 = Sem alimentação	-	Somente leitura

Estes erros são função do número de telegramas inválidos recebidos ou transmitidos para a rede, de acordo com o descrito no item 1.1.4. O estado *error passive*, por exemplo, ocorre quando somente um equipamento está conectado à rede, enviando telegramas sem que outro equipamento reconheça esses telegramas. O estado *bus off* pode ocorrer, por exemplo, quando dispositivos com diferentes taxas de comunicação são conectados na mesma rede, ou devido a problemas na instalação, como a falta de resistores de terminação. É necessário também alimentar a interface com $24V_{CC}$ através do conector da rede, e a falta desta alimentação também é indicada através deste parâmetro.

3.2 P071 - Número de telegramas CAN recebidos

Este parâmetro funciona como um contador cíclico, que é incrementado toda vez que um telegrama CAN é recebido. Ele apenas fornece um retorno para o operador se o SCA-05 está conseguindo se comunicar com a rede.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 ... 32767	-	Somente leitura

3.3 P072 - Número de telegramas CAN transmitidos

Da mesma forma que o P071, este parâmetro funciona como um contador cíclico, que é incrementado toda vez que um telegrama CAN é transmitido, fornecendo um retorno para o operador se o SCA-05 está conseguindo se comunicar com a rede.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 ... 32767	-	Somente leitura

3.4 P073 - Número de erros de bus-off registrados

Também é um contador cíclico, indicando o número vezes que o SCA-05 entrou em estado de erro de bus-off na rede CAN.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 ... 32767	-	Somente leitura

Sempre que o SCA-05 é desligado ou efetuado um *reset* de erros, estes contadores (P071, P072 e P073) voltam para o valor 0 (zero), e iniciam novamente a contagem. Ao passar do valor 32767, os contadores também voltam para 0 (zero), e continuam a contagem.

3.5 P075 - Estado da rede CANopen

Indica o estado do SCA-05 com relação a rede CANopen, informando se o protocolo foi habilitado e se o serviço de controle de erros está ativo (guarda do nó - *Guarding*).

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 = Protocolo não está habilitado 1 = Reservado 2 = CANopen habilitado 3 = Guarda do nó habilitada 4 = Erro de guarda do nó	-	Somente leitura

A habilitação é feita através do parâmetro P700. O protocolo também não estará habilitado caso não seja fornecida a alimentação do barramento. Uma vez habilitado, o dispositivo está pronto para se comunicar através da rede. Uma das funções importantes para o dispositivo poder detectar erros, seja nos escravos ou no mestre da rede, é o serviço de guarda do dispositivo. Para saber como habilitar esta função, consulte o item [5.6.2](#).

3.6 P076 - Estado do nó CANopen

O SCA-05 opera como um escravo da rede CANopen, e como tal possui uma máquina de estados que controla o seu comportamento com relação à comunicação. Este parâmetro indica em qual estado encontra-se o servoconversor.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 = Inicialização 4 = Pré-operacional 5 = Operacional 127 = Parado	-	Somente leitura

Caso não seja fornecida a alimentação do barramento, este parâmetro irá indicar o valor 0 (zero).

3.7 P313 - Ação para erro de comunicação

Quando o SCA-05 está sendo operado via rede, caso o servo esteja habilitado e um erro de comunicação ocorra (cabo rompido, queda na tensão de alimentação da rede, etc.), não será possível enviar comandos para desabilitá-lo. Dependendo da aplicação, isto pode ser um problema, e para evitá-lo é possível programar no P313 uma ação que o SCA-05 executará automaticamente no caso de falha de rede.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 = Sem ação 1 = Causa falha 2 = Executa função STOP 3 = Desabilita servo	0	Leitura/ escrita

A execução da função STOP é feita através da escrita automática do valor 1 no parâmetro P432, enquanto que a desabilitação é feita pela escrita automática do valor 0 (zero) no parâmetro P099. São considerados erros na comunicação os eventos de falha na alimentação (E33), *bus off* (E34) e erro na guarda do nó (E35). No caso de erro de *bus off*, este somente será indicado se o reset deste erro estiver programado para o modo manual (P703).

Caso seja programado neste parâmetro os valores 0 (zero), 2 ou 3, sempre que um erro de comunicação ocorrer, caso este seja resolvido, será removida a indicação deste erro na IHM. Porém com a opção 1, será gerado em erro fatal no drive, que somente será removido em caso de reset de erros.

3.8 P700 - Protocolo CAN

O P700 permite a seleção do protocolo da camada de aplicação desejado para o barramento CAN no SCA-05. É necessário selecionar a opção '1' para habilitar a comunicação CANopen para o servo.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 = Desabilitado 1 = CANopen 2 = DeviceNet 3 = CANMS	0 = Desabilitado	Leitura/ escrita

**NOTA!**

- A alteração deste parâmetro somente será válida após um *reset* do SCA-05.
- Diferentes protocolo CAN não podem operar em conjunto no mesmo barramento.

3.9 P701 - Endereço na rede CAN

Este parâmetro possibilita a seleção do endereço (Node-ID) do SCA-05 na rede CANopen.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
1 ... 127	63	Leitura/ escrita

Cada dispositivo da rede precisa ter um Node-ID distinto, e portanto podem haver até 127 dispositivos em uma única rede. Este Node-ID também é utilizado para definir o valor inicial para alguns objetos de comunicação no SCA-05.

**NOTA!**

A alteração do Node-ID somente será válida após um *reset* do SCA-05.

3.10 P702 - Taxa de comunicação

Este parâmetro possibilita a seleção da taxa de comunicação (*baudrate*) utilizada pelo dispositivo.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
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 8 = 10 kbit/s	0 = 1 Mbit/s	Leitura/ escrita

Para que os dispositivos da rede possam se comunicar, é necessário que todos possuam o mesmo *baudrate* configurado. Não esquecer também que existe uma limitação da taxa de comunicação de acordo com o comprimento do cabo usado na instalação (ver tabela 7).

**NOTA!**

A alteração do *baudrate* somente será válida após um *reset* do SCA-05.

3.11 P703 - Reset de *bus off*

Quando o número de erros ocorridos na rede CAN é muito grande, o dispositivo pode entrar no estado de *bus off* (ver item 1.1.4), onde ele deixa de acessar a rede. Caso este erro ocorra, o parâmetro P703 permite programar se o SCA-05 deve permanecer no estado de *bus off*, ou fazer o *reset* do erro automaticamente e reiniciar a comunicação.

<i>Faixa de valores</i>	<i>Padrão</i>	<i>Acesso</i>
0 = Permanecer em erro	0	Leitura/
1 = Reset automático		escrita

4 Dicionário de objetos

O dicionário de objetos é uma lista com os diversos dados do equipamento que são acessíveis via 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:

<i>Índice</i>	<i>Objetos</i>	<i>Descrição</i>
0001h - 0360h	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 irão representar.
6000h - 9FFFh	Objetos padronizados para dispositivos	Esta faixa é reservada para objetos que descrevem o comportamento de equipamentos similares, independente do fabricante.

Tabela 8: Agrupamentos do dicionário de objetos

Demais índices não referenciados nesta lista são reservados para uso futuro.

4.1 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 - 0360h).
- *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.

4.2 Tipos de dados

A primeira parte do dicionário de objetos (índices 0001h - 0360h) 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*. A seguir são apresentados os tipos de objetos utilizados pelo SCA-05.

4.2.1 Tipos básicos

Os tipos básicos de dados suportados são os seguintes:

- *Inteiros com sinal*: existem três tipos de inteiros com sinal suportados pelo SCA-05, INTEGER8, INTEGER16 e INTEGER32, que representam, respectivamente, inteiros com 8, 16 e 32 bits de dados. Inteiros com sinal são calculados utilizando complemento de dois, e durante a transmissão, sempre o byte menos significativo é transmitido primeiro em um telegrama CAN.
- *Inteiros sem sinal*: existem três tipos de inteiros sem sinal suportados pelo SCA-05, UNSIGNED8, UNSIGNED16 e UNSIGNED32, que representam, respectivamente, inteiros com 8, 16 e 32 bits de dados. Também durante a transmissão, sempre o byte menos significativo é transmitido primeiro.

4.2.2 Tipos compostos

É possível formar novos tipos de dados através do agrupamento de tipos básicos em listas (*arrays* - formados por um único tipo de dado) e estruturas (*records* - formado por diversos tipos de dados). Neste caso, cada item deste tipo é identificado através de um sub-índice. Os tipos compostos utilizados pelo servo são listados abaixo.

- *PDO_COMM_PARAMETER*: este *record* define as informações necessárias para configurar um PDO para a comunicação CANopen. O conteúdo e forma como cada campo é utilizado são detalhados no item 5.3.

Sub-índice	Descrição da entrada	Tipo
00h	Número de entradas suportadas neste <i>record</i>	UNSIGNED8
01h	COB-ID	UNSIGNED32
02h	Transmission type	UNSIGNED8
03h	Inhibit time	UNSIGNED16
04h	Reservado	UNSIGNED8
05h	Event timer	UNSIGNED16

Tabela 9: *Record* para configuração dos PDOs

- *PDO_MAPPING*: este *record* define como mapear os dados que serão transmitidos por um PDO durante a comunicação CANopen. O conteúdo e forma como cada campo é utilizado são detalhados no item 5.3.

Sub-índice	Descrição da entrada	Tipo
00h	Número de objetos mapeados no PDO	UNSIGNED8
01h	1º objeto mapeado	UNSIGNED32
02h	2º objeto mapeado	UNSIGNED32
⋮	⋮	⋮
40h	64º objeto mapeado	UNSIGNED32

Tabela 10: *Record* para mapeamento dos dados de um PDO

- *SDO_PARAMETER*: este *record* define as informações necessárias para configurar um SDO para a comunicação CANopen. O conteúdo e forma como cada campo é utilizado são detalhados no item 5.2.

Sub-índice	Descrição da entrada	Tipo
00h	Número de entradas suportadas neste <i>record</i>	UNSIGNED8
01h	COB-ID cliente → servidor	UNSIGNED32
02h	COB-ID servidor → cliente	UNSIGNED32
03h	Node-ID do cliente/servidor	UNSIGNED8

Tabela 11: *Record* para configuração dos SDOs

- *IDENTITY*: este *record* é utilizado para descrever o tipo de dispositivo presente na rede.

Sub-índice	Descrição da entrada	Tipo
00h	Número de entradas suportadas neste <i>record</i>	UNSIGNED8
01h	Vendor-ID	UNSIGNED32
02h	Product Code	UNSIGNED32
03h	Revision Number	UNSIGNED32
04h	Serial Number	UNSIGNED32

Tabela 12: *Record* para identificação do dispositivo

4.2.3 Tipos estendidos

O SCA-05 não possui tipos estendidos.

4.3 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, porém somente alguns são obrigatórios. A seguir é apresentada uma lista com os objetos desta faixa suportados pelo SCA-05.

<i>Índice</i>	<i>Objeto</i>	<i>Nome</i>	<i>Tipo</i>	<i>Acesso</i>
1000h	VAR	device type	UNSIGNED32	ro
1001h	VAR	error register	UNSIGNED8	ro

<i>Índice</i>	<i>Objeto</i>	<i>Nome</i>	<i>Tipo</i>	<i>Acesso</i>
1003h	ARRAY	pre-defined error field	UNSIGNED32	ro
1005h	VAR	COB-ID SYNC	UNSIGNED32	rw
100Ch	VAR	guard time	UNSIGNED16	rw
100Dh	VAR	life time factor	UNSIGNED8	rw
1014h	VAR	COB-ID EMCY	UNSIGNED32	ro
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
1402h	RECORD	3rd receive PDO Parameter	PDO CommPar	rw
1403h	RECORD	4th 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
1602h	RECORD	3rd receive PDO mapping	PDO Mapping	rw
1603h	RECORD	4th 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
1802h	RECORD	3rd transmit PDO Parameter	PDO CommPar	rw
1803h	RECORD	4th 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
1A02h	RECORD	3rd transmit PDO mapping	PDO Mapping	rw
1A03h	RECORD	4th transmit PDO mapping	PDO Mapping	rw

Tabela 13: Lista de objetos do SCA-05 - Communication Profile

Demais objetos não mostrados nesta lista não são utilizados pelo SCA-05, ou então estão em faixas reservadas do dicionário.

4.4 Manufacturer Specific - Objetos específicos do SCA-05

Nos índices de 2000h até 5FFFh, cada fabricante é livre para definir quais objetos estarão presentes, o tipo e a função de cada objeto. No caso do SCA-05, nesta faixa de objetos foi disponibilizada toda a lista de parâmetros. Através destes parâmetros é possível operar o SCA-05, executando qualquer função que o servo possa realizar.

Os parâmetros foram disponibilizados a partir do índice 2000h, e com o número do parâmetro somado a este índice para obter sua posição no dicionário. A tabela a seguir ilustra como estão distribuídos os parâmetros no dicionário de objetos.

<i>Índice</i>	<i>Objeto</i>	<i>Nome</i>	<i>Tipo</i>	<i>Acesso</i>
2000h	VAR	P000 - Parâmetro de acesso	INTEGER16	rw
2002h	VAR	P002 - Velocidade do motor	INTEGER16	ro
2003h	VAR	P003 - Corrente do motor	INTEGER16	ro
2004h	VAR	P004 - Tensão CC	INTEGER16	ro
2006h	VAR	P006 - Estado do Servoconversor	INTEGER16	ro
200Ch	VAR	P012 - Estado DI1 ... DI6	INTEGER16	ro
200Eh	VAR	P014 - Último erro	INTEGER16	ro
⋮	⋮	⋮	⋮	⋮
2063h	VAR	P099 - Habilitação	INTEGER16	rw
2064h	VAR	P100 - Tempo de aceleração	INTEGER16	rw
2065h	VAR	P101 - Tempo de desaceleração	INTEGER16	rw
⋮	⋮	⋮	⋮	⋮

Tabela 14: Lista de objetos do SCA-05 - Manufacturer Specific

Para a lista completa e uma descrição detalhada dos parâmetros, consulte o manual do SCA-05. É necessário saber operar o servo através dos parâmetros para poder programar corretamente sua operação via rede CANopen.

4.5 Device Profile - Objetos comuns para drives

A documentação CANopen também inclui propostas para padronização de determinados tipos de dispositivos. No caso do SCA-05, ele segue o descrito pela *CiA DPS 402 - Device Profile Drives and Motion Control*. Este documento descreve um conjunto de objetos que devem ser comuns para drives, independente do fabricante. Isto facilita a interoperabilidade entre dispositivos com a mesma função (como servoconversores), pois tanto os dados quanto o comportamento do dispositivo são disponibilizados de uma forma padronizada.

Para estes objetos foram reservados os índices de 6000h até 9FFFh. É possível operar o servoconversor via rede CANopen, tanto através dos parâmetros (localizados a partir do índice 2000h) quanto através destes objetos padronizados. Para uma descrição detalhada de quais objetos estão disponíveis nesta faixa do dicionário de objetos, consulte a seção 6.

5 Descrição dos objetos de comunicação

Nesta seção são descritos detalhadamente cada um dos objetos citados na tabela 13, além de descrever também o funcionamento dos objetos de comunicação (COBs) referenciados no item 1.7. É necessário conhecer como estes objetos são operados para utilizar as funções disponíveis para a comunicação do SCA-05.

5.1 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.

5.1.1 Objeto 1000h - Device Type

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

<i>Índice</i>	1000h
<i>Nome</i>	Device type
<i>Objeto</i>	VAR
<i>Tipo</i>	UNSIGNED32
<i>Acesso</i>	ro
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	0002.0192h

Este código pode ser dividido em duas partes: 16 bits inferiores, descrevendo o tipo de *Profile* que o dispositivo utiliza, e 16 bits superiores, indicando uma função específica, de acordo com o *Profile* especificado. Para o SCA-05, estes valores são respectivamente 0192h (segue o especificado pelo documento *Device Profile Drives and Motion Control*), e 0002h (possui funcionalidades de *Servo Drive*).

5.1.2 Objeto 1001h - Error Register

Este objeto indica a ocorrência ou não de erro no dispositivo. O tipo de erro registrado para o SCA-05 segue o descrito pela tabela 15.

<i>Índice</i>	1001h
<i>Nome</i>	Error register
<i>Objeto</i>	VAR
<i>Tipo</i>	UNSIGNED8
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Faixa</i>	UNSIGNED8
<i>Valor Padrão</i>	0

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

Tabela 15: Estrutura do objeto Error Register

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.

5.1.3 Objeto 1003h - Pre-defined error field

Este objeto armazena a lista de erros existentes no dispositivo, e que foram reportados via EMCY. O sub-índice 0 (zero) indica a quantidade de erros ocorridos, e os demais sub-índices informam os códigos de erro ocorridos. Uma vez que o erro tenha sido corrigido, o código do erro é removido da lista. Pode-se limpar esta lista escrevendo o valor 0 (zero) no sub-índice 0 (zero). O SCA-05 suporta apenas um código de erro, logo a lista possui apenas uma posição.

<i>Índice</i>	1003h
<i>Nome</i>	Pre-defined error field
<i>Objeto</i>	Array
<i>Tipo</i>	UNSIGNED32

<i>Sub-índice</i>	0
<i>Descrição</i>	Número de erros
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED8
<i>Valor Padrão</i>	0

<i>Sub-índice</i>	1
<i>Descrição</i>	Código do erro
<i>Acesso</i>	ro
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	0

O código de erro de 32 bits é montado a partir de duas informações básicas: o número do erro de acordo com a tabela de erros especificada pela CiA e o número do erro ocorrido no SCA-05, que são agrupados da seguinte forma:

<i>Código de erro SCA-05</i>	<i>Código de erro CiA</i>
UNSIGNED16	UNSIGNED16

Os campos de código de erro seguem o descrito pela tabela abaixo.

<i>Código de erro SCA-05</i>	<i>Código de erro CiA</i>	<i>Descrição</i>
00	2340h	Curto-circuito entre fases na saída
01	3210h	Sobretensão no link CC
02	3220h	Subtensão no link CC
04	4310h	Sobretensão no dissipador de potência
05	2310h	Sobrecarga na saída
06	9000h	Erro externo
08	6010h	Erro na CPU (<i>watchdog</i>)
11	2330h	Curto-circuito fase-terra na saída
12	7112h	Sobrecarga no resistor de frenagem
28	8100h	<i>Timeout</i> na comunicação serial
29	8100h	Cartão fieldbus opcional está offline
30	8100h	Erro de acesso no cartão fieldbus opcional
32	4210h	Sobretensão no motor
33	FF00h	Falta na alimentação da interface de rede
34	FF00h	Detectado erro de bus off
35	8130h	<i>Timeout</i> na guarda do nó

Tabela 16: Tabela de códigos de erro

5.1.4 Objeto 1018h - Identity object

Traz informações gerais sobre o dispositivo.

<i>Índice</i>	1018h
<i>Nome</i>	Identity object
<i>Objeto</i>	Record
<i>Tipo</i>	Identity
<i>Sub-índice</i>	0
<i>Descrição</i>	Número do último sub-índice
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED8
<i>Valor Padrão</i>	4
<i>Sub-índice</i>	1
<i>Descrição</i>	Vendor ID
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	0000.0123h

<i>Sub-índice</i>	2
<i>Descrição</i>	Código do produto
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	Modelo 220V 4/8A: 0000.0100h Modelo 220V 8/16A: 0000.0101h Modelo 220V 24/48A: 0000.0102h Modelo 380V 30/60A: 0000.0103h
<i>Sub-índice</i>	3
<i>Descrição</i>	Número da revisão
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	De acordo com a versão de firmware do equipamento
<i>Sub-índice</i>	4
<i>Descrição</i>	Número serial
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	Diferente para cada SCA-05

O *Vendor ID* é um número que identifica o fabricante junto à CiA. Neste caso, WEG Indústrias S.A. - Divisão Automação é representada pelo número 0000.00123h. O código do produto é definido pelo fabricante, e varia de acordo com o modelo do servoconversor. O número da revisão representa a versão de *firmware* do equipamento. O sub-índice 4 é um número serial único para cada servoconversor SCA-05 em rede CANopen.

5.2 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.

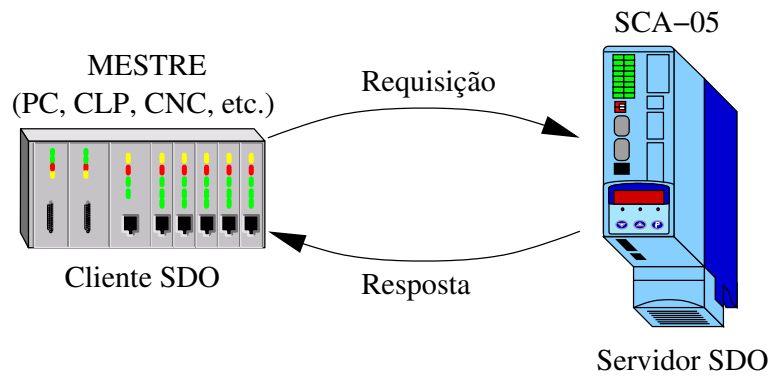


Figura 5: Comunicação entre Cliente e Servidor SDO

5.2.1 Objeto 1200h - Servidor SDO

O SCA-05 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 servoconversor. Todo o servidor SDO possui um objeto, do tipo SDO_-PARAMETER (ver item 4.2.2), para a sua configuração, possuindo a seguinte estrutura:

<i>Índice</i>	1200h
<i>Nome</i>	Server SDO Parameter
<i>Objeto</i>	Record
<i>Tipo</i>	SDO Parameter
<i>Sub-índice</i>	0
<i>Descrição</i>	Número do último sub-índice
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED8
<i>Valor Padrão</i>	2
<i>Sub-índice</i>	1
<i>Descrição</i>	COB-ID Cliente - Servidor (rx)
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	600h + Node-ID
<i>Sub-índice</i>	2
<i>Descrição</i>	COB-ID Servidor - Cliente (tx)
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	580h + Node-ID

5.2.2 Funcionamento dos SDOs

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

<i>Identificador</i>	<i>8 bytes de dados</i>							
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:

<i>Comando</i>	<i>Função</i>	<i>Descrição</i>	<i>Dados do objeto</i>
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

Tabela 17: Código dos comandos para cliente SDO

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, porém os comandos serão diferentes:

<i>Comando</i>	<i>Função</i>	<i>Descrição</i>	<i>Dados do objeto</i>
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

Tabela 18: Código dos comandos para servidor SDO

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!**

Quando o SDO é utilizado para escrita nos objetos que representam os parâmetros do SCA-05 (objetos a partir do índice 2000h), este valor é salvo na memória não volátil do servoconversor. Desta forma, após desligado ou feito o *reset* do equipamento, os valores configurados não são perdidos. Para os demais objetos, estes valores não são salvos automaticamente, de maneira que é necessário reescrever os valores desejados.

Exemplo: um cliente SDO solicita para um SCA-05 no endereço 1, a leitura do objeto identificado pelo índice 2000h, sub-índice 0 (zero), que representa um inteiro de 16 bits. O telegrama do mestre possui a seguinte forma:

<i>Identificador</i>	<i>Comando</i>	<i>Índice</i>		<i>Sub-índice</i>	<i>Dados</i>			
601h	40h	00h	20h	00h	00h	00h	00h	00h

O SCA-05 responde à requisição, indicando que o valor para o referido objeto é igual a 1234h³:

<i>Identificador</i>	<i>Comando</i>	<i>Índice</i>		<i>Sub-índice</i>	<i>Dados</i>			
581h	4Bh	00h	20h	00h	34h	12h	00h	00h

5.3 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. Porém é necessário 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.

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.

³Não esquecer que qualquer dado do tipo inteiro, a ordem de transferência dos bytes vai do menos significativo até o mais significativo

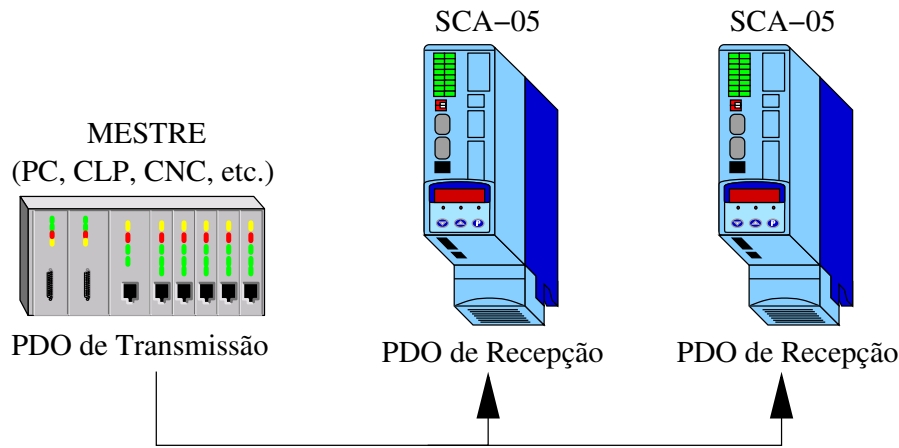


Figura 6: Comunicação utilizando PDOs

**NOTA!**

PDOs somente podem ser transmitidos ou recebidos quando o dispositivo está no estado operacional. A figura 9 ilustra os estados disponíveis para um nó da rede CANopen.

5.3.1 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 se ele é ou não mapeável. 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 SCA-05 (2000h - 5FFFh), a tabela a seguir apresenta quais são os objetos mapeáveis para os PDOs. Parâmetros com acesso apenas para leitura (ro) podem ser utilizados apenas por PDOs de transmissão, enquanto que os demais parâmetros podem ser utilizados apenas por PDOs de recepção.

<i>Índice</i>	<i>Objeto</i>	<i>Nome</i>	<i>Tipo</i>	<i>Acesso</i>
2002h	VAR	P002 - Velocidade do motor	INTEGER16	ro
2003h	VAR	P003 - Corrente do motor	INTEGER16	ro
2004h	VAR	P004 - Tensão CC	INTEGER16	ro
2006h	VAR	P006 - Estado do Servoconversor	INTEGER16	ro
200Ch	VAR	P012 - Estado DI1 ... DI6	INTEGER16	ro
200Eh	VAR	P014 - Último erro	INTEGER16	ro
2012h	VAR	P018 - Valor da AI1	INTEGER16	ro
2013h	VAR	P019 - Valor da AI2	INTEGER16	ro
2032h	VAR	P050 - Posição absoluta do eixo	INTEGER16	ro
2034h	VAR	P052 - Posição do eixo - pulsos	INTEGER16	ro
2035h	VAR	P053 - Posição do eixo - voltas	INTEGER16	ro
2063h	VAR	P099 - Habilitação	INTEGER16	rw
2064h	VAR	P100 - Tempo de aceleração	INTEGER16	rw
2065h	VAR	P101 - Tempo de desaceleração	INTEGER16	rw
206Fh	VAR	P111 - Sentido de giro	INTEGER16	rw
2075h	VAR	P117 - Referência de posição	INTEGER16	rw
2077h	VAR	P119 - Referência de torque	INTEGER16	rw
2079h	VAR	P121 - Referência de velocidade	INTEGER16	rw
20CAh	VAR	P202 - Modo de operação	INTEGER16	rw
21B0h	VAR	P432 - Aciona função STOP	INTEGER16	rw
21B3h	VAR	P435 - Aciona função MOVE	INTEGER16	rw

Tabela 19: Lista de parâmetros mapeáveis para PDOs

5.3.2 PDOs de recepção

Os PDOs de recepção, ou RPDOs, são responsáveis por receber dados que outros dispositivos enviam para a rede CANopen. O SCA-05 possui 4 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

<i>Índice</i>	1400h - 1403h
<i>Nome</i>	Receive PDO communication parameter
<i>Objeto</i>	Record
<i>Tipo</i>	PDO_COMM_PARAMETER
<i>Sub-índice</i>	0
<i>Descrição</i>	Número do último sub-índice
<i>Acesso</i>	ro
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED8
<i>Valor Padrão</i>	2

<i>Sub-índice</i>	1
<i>Descrição</i>	COB-ID usado pelo PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1400h: 200h + Node-ID 1401h: 300h + Node-ID 1402h: 400h + Node-ID 1403h: 500h + Node-ID
<i>Sub-índice</i>	2
<i>Descrição</i>	Tipo de transmissão
<i>Acesso</i>	rw
<i>Mapeável</i>	Não possível
<i>Faixa</i>	UNSIGNED8
<i>Valor Padrão</i>	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:

<i>Bit</i>	<i>Valor</i>	<i>Descrição</i>
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 pelo SCA-05, sempre 0
10 - 0 (LSB)	X	COB-ID de 11 bits

Tabela 20: Descrição do COB-ID

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 servoconversor não utiliza identificadores de 29 bits, os bits de 28 - 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.

<i>Tipo de transmissão</i>	<i>Transmissão de PDOs</i>				
	<i>Cíclico</i>	<i>Acíclico</i>	<i>Síncrono</i>	<i>Assíncrono</i>	<i>RTR</i>
0		•	•		
1 - 240	•		•		
241 - 251	Reservado				
252			•		•
253				•	•
254				•	
255				•	

Tabela 21: Descrição do tipo de transmissão

- *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

<i>Índice</i>	1600h - 1603h
<i>Nome</i>	Receive PDO mapping
<i>Objeto</i>	Record
<i>Tipo</i>	PDO_MAPPING

<i>Sub-índice</i>	0
<i>Descrição</i>	Número de objetos mapeados
<i>Acesso</i>	ro
<i>Mapeável</i>	Não
<i>Faixa</i>	0 = desabilitado 1 ... 4 = número de objetos mapeados
<i>Valor Padrão</i>	1600h: 1 1601h: 2 1602h: 2 1603h: 2

<i>Sub-índice</i>	1
<i>Descrição</i>	1º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1600h: 6040.0010h 1601h: 6040.0010h 1602h: 6040.0010h 1603h: 6040.0010h

<i>Sub-índice</i>	2
<i>Descrição</i>	2º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1600h: 0 1601h: 6060.0008h 1602h: 607A.0020h 1603h: 60FF.0020h

<i>Sub-índice</i>	3
<i>Descrição</i>	3º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1600h: 0 1601h: 0 1602h: 0 1603h: 0
<i>Sub-índice</i>	4
<i>Descrição</i>	4º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1600h: 0 1601h: 0 1602h: 0 1603h: 0

Este parâmetro indica os objetos mapeados para cada um dos quatro PDOs de recepção disponíveis. Para um único PDO, é possível mapear até quatro objetos diferentes, desde que o tamanho total não ultrapasse oito bytes. O mapeamento de um objeto é feito indicando o seu índice, sub-índice⁴ e tamanho (em bits) em um campo UNSIGNED32, com o seguinte formato:

<i>UNSIGNED32</i>		
Índice (16 bits)	Sub-índice (8 bits)	Tamanho do objeto (8 bits)

Como exemplo, podemos analisar o primeiro PDO de recepção, onde temos:

- *Sub-índice 0 = 1*: o RPDO possui apenas um objeto mapeado.
- *Sub-índice 1 = 6040.0010h*: o primeiro objeto mapeado possui índice igual a 6040h, sub-índice igual a 0 (zero) e tamanho de 16 bits. Este objeto corresponde à palavra de controle do servo (ver item 6.1.1).

Desta forma, sempre que este PDO receber um telegrama, ele vai saber que o telegrama deverá conter dois bytes de dados, com o conteúdo para a palavra de controle do servo. Estes valores são valores padrão para o SCA-05. É possível modificar este mapeamento, alterando a quantidade ou o número dos objetos mapeados. Lembrar que no máximo podem ser mapeados 4 objetos ou 8 bytes.

⁴caso o objeto seja do tipo VAR e não possua sub-índice, deve ser indicado o valor 0 (zero) para o sub-índice

**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 4 podem ser alterados. Após feito o mapeamento desejado, deve-se escrever novamente no sub-índice 0 (zero) o número de objetos que foram mapeados, habilitando novamente o PDO.
- Para agilizar a atualização dos dados via PDO, os valores recebidos através destes objetos não são salvos na memória não volátil do servoconversor. Desta forma, após um desligamento ou *reset* do equipamento, os objetos modificados por um RPDO voltam para o seu valor padrão.

5.3.3 PDOs de transmissão

Os PDOs de transmissão, ou TPDOs, como o nome diz, são responsáveis por transmitir dados para a rede CANopen. O SCA-05 possui 4 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.

<i>Índice</i>	1800h - 1803h
<i>Nome</i>	Transmit PDO Parameter
<i>Objeto</i>	Record
<i>Tipo</i>	PDO_COMM_PARAMETER

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

<i>Sub-índice</i>	1
<i>Descrição</i>	COB-ID usado pelo PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1800h: 180h + Node-ID 1801h: 280h + Node-ID 1802h: 380h + Node-ID 1803h: 480h + Node-ID

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

<i>Sub-índice</i>	3
<i>Descrição</i>	Tempo entre transmissões
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED16
<i>Valor Padrão</i>	-
<i>Sub-índice</i>	4
<i>Descrição</i>	Reservado
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED8
<i>Valor Padrão</i>	-
<i>Sub-índice</i>	5
<i>Descrição</i>	Temporizador de eventos
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	0 = desabilitado UNSIGNED16
<i>Valor Padrão</i>	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 20.

O sub-índice 2 indica o tipo de transmissão deste objeto, que segue o descrito pela tabela 21. 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 e 255*: o objeto deve ser transmitido de acordo com o timer programado no sub-índice 5.

No sub-índice 3 é possível programar um tempo mínimo (em múltiplos de $100\mu s$) que deve transcorrer para que, após 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 a 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. O valor máximo permitido para esta programação é de 3276ms.



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.
- Não esquecer que os PDOs somente podem ser transmitidos ou recebidos no estado operacional.

PDO_MAPPING

<i>Índice</i>	1A00h - 1A03h
<i>Nome</i>	Transmit PDO mapping
<i>Objeto</i>	Record
<i>Tipo</i>	PDO_MAPPING

<i>Sub-índice</i>	0
<i>Descrição</i>	Número do último sub-índice
<i>Acesso</i>	ro
<i>Mapeável</i>	Não
<i>Faixa</i>	0 = desabilitado 1 ... 4 = número de objetos mapeados
<i>Valor Padrão</i>	1A00h: 1 1A01h: 2 1A02h: 2 1A03h: 2

<i>Sub-índice</i>	1
<i>Descrição</i>	1º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1A00h: 6041.0010h 1A01h: 6041.0010h 1A02h: 6041.0010h 1A03h: 6041.0010h

<i>Sub-índice</i>	2
<i>Descrição</i>	2º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1A00h: 0 1A01h: 6061.0008h 1A02h: 6064.0020h 1A03h: 606C.0020h

<i>Sub-índice</i>	3
<i>Descrição</i>	3º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1A00h: 0 1A01h: 0 1A02h: 0 1A03h: 0
<i>Sub-índice</i>	4
<i>Descrição</i>	4º objeto mapeado no PDO
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	1A00h: 0 1A01h: 0 1A02h: 0 1A03h: 0

O PDO_MAPPING para a transmissão funciona de forma semelhante que para a 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:

<i>UNSIGNED32</i>		
Índice (16 bits)	Sub-índice (8 bits)	Tamanho do objeto (8 bits)

Como exemplo, podemos analisar o primeiro PDO de transmissão, onde temos:

- *Sub-índice 0 = 1*: este PDO possui apenas um objeto mapeado.
- *Sub-índice 1 = 6041.0010h*: o primeiro objeto mapeado possui índice igual a 6041h, sub-índice 0 (zero), e tamanho igual a 16 bits. Este objeto corresponde à palavra de estado do servoconversor (ver item [6.1.2](#)).

Desta forma, sempre que este PDO for transmitir os seus dados, ele vai elaborar o seu telegrama contendo quatro bytes de dados, com os valores dos parâmetros P006 e P014. Estes valores são valores padrão para o SCA-05. É possível modificar este mapeamento, alterando a quantidade ou o número dos parâmetros mapeados. Lembrar que no máximo podem ser mapeados 4 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 4 podem ser alterados. Após feito o mapeamento desejado, deve-se escrever novamente no sub-índice 0 (zero) o número de objetos que foram mapeados, habilitando novamente o PDO.

5.4 Emergency Object - EMCY

O objeto de emergência (EMCY) é utilizado para sinalizar a ocorrência de um erro no dispositivo. Sempre que um erro ocorrer no SCA-05 (curto-circuito, sobretensão, falha na comunicação, etc.), este objeto irá enviar uma mensagem de emergência para a rede. Esta mensagem poderá ser interpretada por um consumidor EMCY (usualmente o mestre da rede), que poderá tomar uma ação de acordo com o programado para a aplicação, como fazer o *reset* de erro ou desabilitar os demais dispositivos da rede.

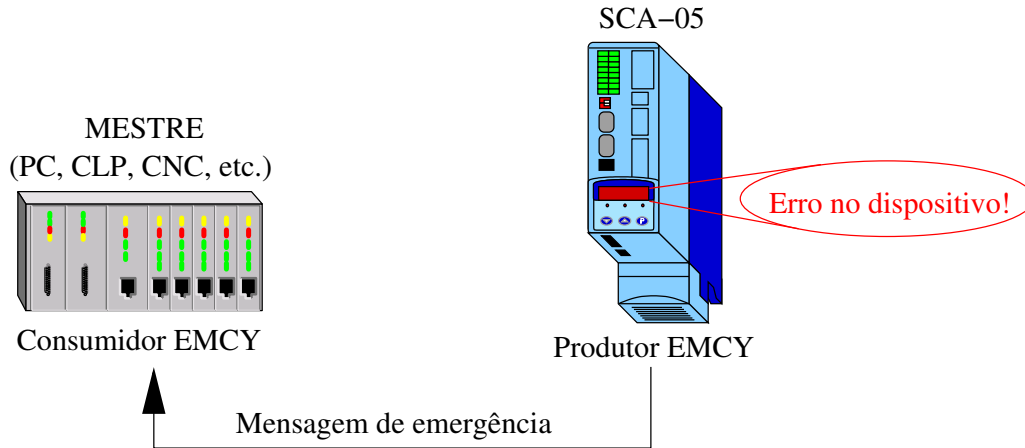


Figura 7: EMCY

Ao transmitir uma mensagem, três informações são enviadas nos oito bytes de dados do telegrama: o código de erro CiA, o objeto 1001h (*error register*) e o código do erro do SCA-05. Estes códigos estão descritos na tabela 16. O telegrama possui então a seguinte estrutura:

byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7
Código do erro CiA		Objeto 1001h Error Register	Código do erro SCA-05		Reservado (00h)		

No SCA-05, este objeto está sempre ativo, e portanto sempre reportará para a rede a ocorrência de erros no servo. Existe um único parâmetro que possibilita ler qual o COB-ID do objeto, ou seja, qual o identificador do telegrama para as mensagens de erro enviadas.

<i>Índice</i>	1014h
<i>Nome</i>	COB-ID emergency message
<i>Objeto</i>	VAR
<i>Tipo</i>	UNSIGNED32
<i>Acesso</i>	ro
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	80h + Node-ID

5.5 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 SCA-05 possui a função de consumidor SYNC, e portanto pode programar seus PDOs para serem síncronos. Como descrito na tabela 21, PDOs síncronos são aqueles relacionados com o objeto de sincronização, e portanto podem ser programados para serem transmitidos ou atualizados com base neste objeto.

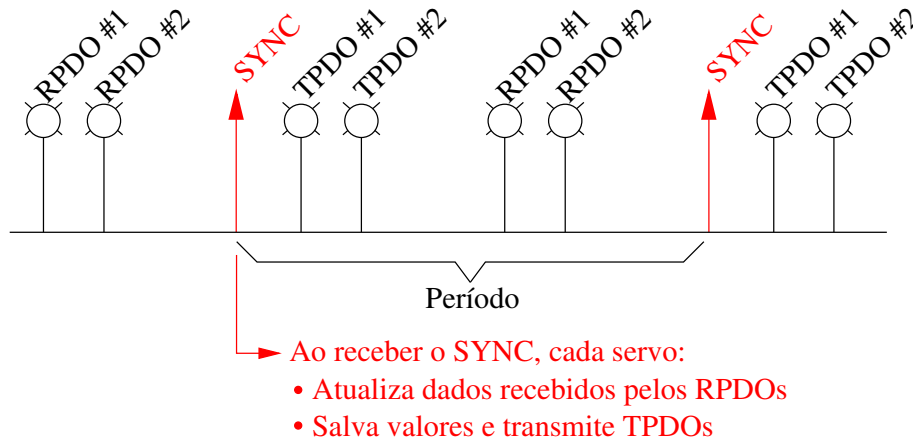


Figura 8: SYNC



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.

A mensagem SYNC transmitida pelo produtor não possui dado algum em seu campo de dados, pois seu objetivo é fornecer uma base de tempo para os demais objetos. No SCA-05, existe um objeto para a configuração do COB-ID do consumidor SYNC.

<i>Índice</i>	1005h
<i>Nome</i>	COB-ID SYNC
<i>Objeto</i>	VAR
<i>Tipo</i>	UNSIGNED32
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED32
<i>Valor Padrão</i>	80h

5.6 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 o SCA-05 estão disponíveis os serviços de controle do nó e de controle de erros (utilizando *Guarding*).

5.6.1 Controle dos estados do escravo

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

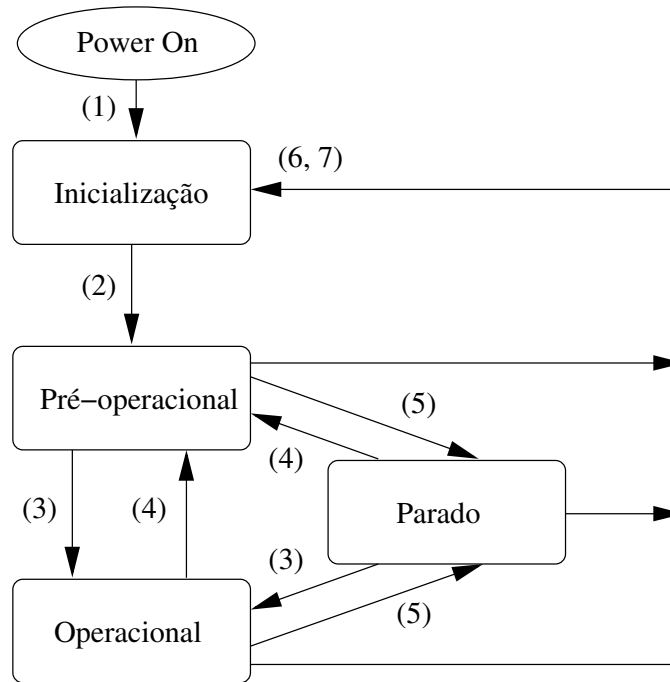


Figura 9: Diagrama de estados do nó CANopen

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

Tabela 22: Descrição das transições

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 $700h + \text{Node-ID}$, e apenas um byte de dados com valor igual a 0 (zero).

No estado pré-operacional, já é possível comunicar-se com o escravo, porém os PDOs 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.

	<i>Inicialização</i>	<i>Pré-operacional</i>	<i>Operacional</i>	<i>Parado</i>
PDO			•	
SDO		•	•	
SYNC		•	•	
EMCY		•	•	
Boot-up	•			
NMT		•	•	•

Tabela 23: Objetos acessíveis em cada estado

O parâmetro P076 permite observar em qual estado o SCA-05 encontra-se no momento, de acordo com o descrito no item 3.6.

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:

<i>Identificador</i>	<i>byte 1</i>	<i>byte 2</i>
00h	Código do comando	Node-ID destino

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

Tabela 24: Comandos recebidos pelo escravo

As transições indicadas no código do comando equivalem às transições de estado executadas pelo nó após receber o comando (conforme figura 9). O comando *Reset node* faz com que o SCA-05 execute um *reset* completo do dispositivo, enquanto que o comando *Reset communication* faz com que o dispositivo reinicialize apenas os objetos relativos à comunicação CANopen.

5.6.2 Controle de Erros - Guarding

Existem dois serviços para controle de erros do dispositivo: *heartbeat* e *guarding*. Destes, apenas o serviço de guarda do nó (*guarding*) está implementado no SCA-05.

A guarda do dispositivo é utilizada 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.

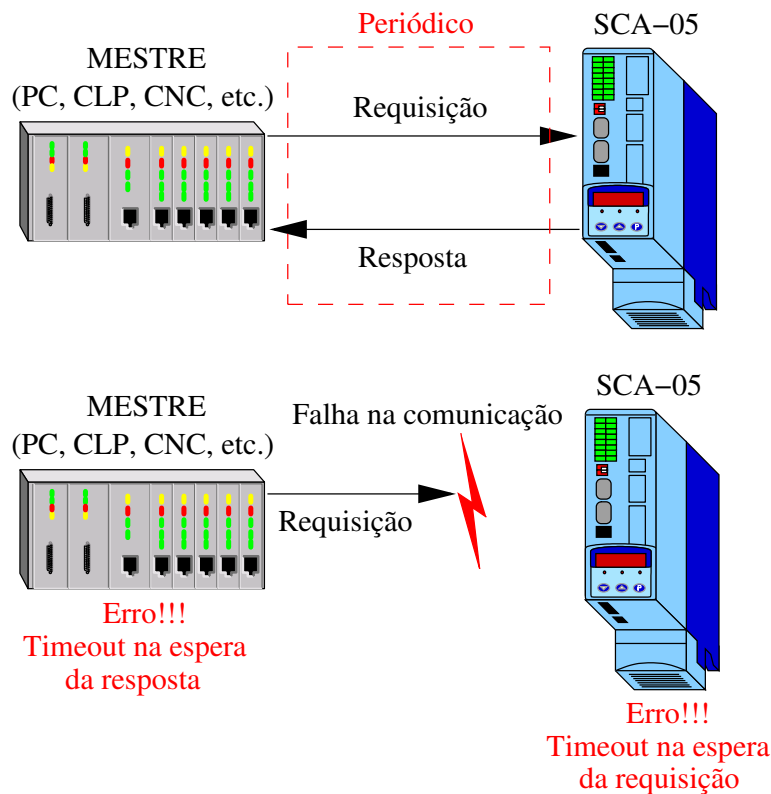


Figura 10: Serviço de controle de erros - *Guarding*

Para o serviço da guarda do dispositivo CANopen, existem dois objetos do dicionário para configuração dos tempos para indicação de erro.

<i>Índice</i>	100Ch
<i>Nome</i>	Guard time
<i>Objeto</i>	VAR
<i>Tipo</i>	UNSIGNED16

<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED16
<i>Valor Padrão</i>	0

<i>Índice</i>	100Dh
<i>Nome</i>	Life time factor
<i>Objeto</i>	VAR
<i>Tipo</i>	UNSIGNED8

<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Faixa</i>	UNSIGNED8 0 = Desabilitado
<i>Valor Padrão</i>	0

O objeto 100Ch permite programar o tempo necessário (em milissegundos - valor máximo permitido igual a 3276) para que uma ocorrência de falha seja detectada, caso o SCA-05 não receba nenhum telegrama de guarda do mestre. O objeto 100Dh indica quantas falhas em sequência são necessárias até que se considere que houve realmente erro de 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 SCA-05 começa a contar estes tempos a partir do primeiro telegrama de guarda recebido do mestre da rede.

O telegrama do mestre é do tipo remoto, e não possui 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 byte de dados 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 guarda enviado pelo escravo (*toggle bit*).

Caso o SCA-05 detecte um erro utilizando este mecanismo, ele automaticamente irá para o estado pré-operacional. O estado do SCA-05 pode ser conferido no parâmetro P076, enquanto que a ocorrência de erro de guarda pode ser observada no parâmetro P075. É possível também programar o SCA-05 para tomar uma ação quando este erro ocorrer, através do parâmetro P313. Consulte o item 3 para a descrição detalhada dos parâmetros.



NOTA!

Com relação à utilização deste serviço, deve-se observar os seguintes pontos:

- Este objeto está ativo mesmo no estado parado (tabela 23).
- O valor máximo permitido para o objeto 100Ch é 3276ms.
- Os tempos para habilitação desta função devem ser programados tanto no mestre da rede quanto no escravo, para que possam operar em conjunto.
- 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 de guarda, e também para que o resto da comunicação possa ser processada.

5.7 Procedimento de inicialização

Uma vez conhecido o funcionamento dos objetos disponíveis para o SCA-05, 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:

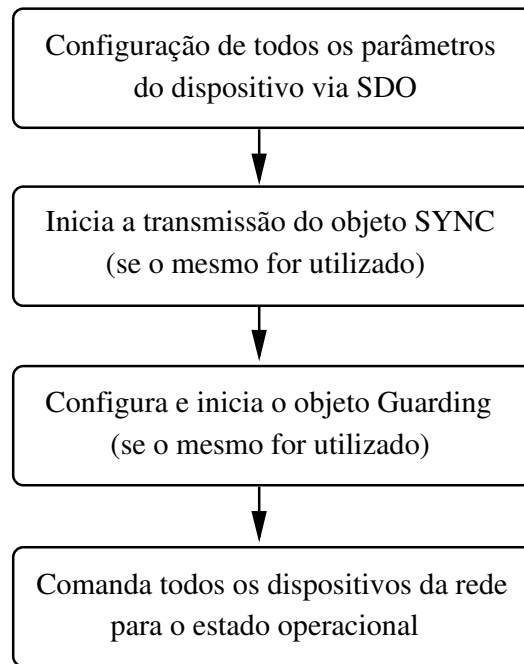


Figura 11: Fluxograma do processo de inicialização

É necessário observar que os objetos de comunicação do SCA-05 (1000h até 1FFFh) não são armazenados na memória não volátil do servoconversor. Desta forma, sempre que for feito o *reset* ou desligado o equipamento, é necessário refazer a parametrização dos objetos de comunicação. Para os objetos específicos do fabricante (2000h até 2FFFh, que representam os parâmetros), estes são armazenados na memória não volátil, e portanto podem ser parametrizados uma única vez.

6 Descrição dos objetos para drives

Neste item serão descritos os objetos comuns para drives, definidos pela especificação CANopen, no documento *CiA DSP 402*. Os objetos citados aqui possuem descrição e operação semelhantes, independente do fabricante do drive. Isto facilita a interoperabilidade e intercambiabilidade entre diferentes dispositivos.

A figura 12 mostra um diagrama com a arquitetura lógica de funcionamento de um drive através da rede CANopen, com os diferentes modos de operação definidos nesta especificação. Cada modo de operação possui um conjunto de objetos que permite a configuração e operação do drive na rede.

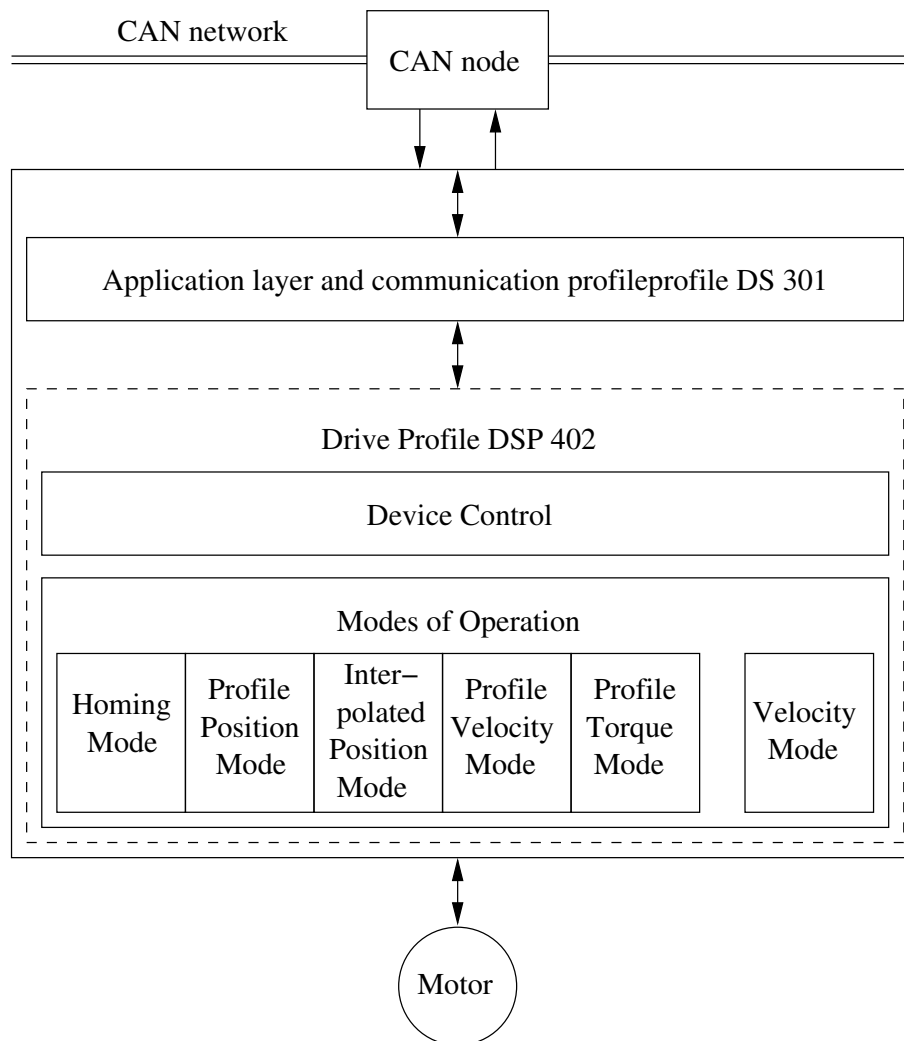


Figura 12: Arquitetura de comunicação para um drive na rede CANopen

Para o SCA-05, somente *Profile velocity mode* e *Profile torque mode* são suportados para a rede CANopen. A tabela 25 mostra a lista dos objetos disponíveis para o SCA-05, divididos de acordo com os diferentes modos de operação do drive. Os grupos *Device control*, *Factor group* e *Position control function* descrevem objetos comuns aos modos de operação.

Índice	Objeto	Nome	Tipo	Acesso	Mapeável
<i>Device Control</i>					

<i>Índice</i>	<i>Objeto</i>	<i>Nome</i>	<i>Tipo</i>	<i>Acesso</i>	<i>Mapeável</i>
6040h	VAR	Controlword	UNSIGNED16	rw	Sim
6041h	VAR	Statusword	UNSIGNED16	ro	Sim
6060h	VAR	Modes of Operation	INTEGER8	rw	Sim
6061h	VAR	Modes of Operation Display	INTEGER8	ro	Sim
<i>Factor Group</i>					
6089h	VAR	Position Notation Index	INTEGER8	rw	Não
608Ah	VAR	Position Dimension Index	UNSIGNED8	rw	Não
608Bh	VAR	Velocity Notation Index	INTEGER8	rw	Não
608Ch	VAR	Velocity Dimension Index	UNSIGNED8	rw	Não
608Dh	VAR	Acceleration Notation Index	INTEGER8	rw	Não
608Eh	VAR	Acceleration Dimension Index	UNSIGNED8	rw	Não
<i>Position Control Function</i>					
6063h	VAR	Position Actual Value	INTEGER32	ro	Sim
6064h	VAR	Position Actual Value in user units	INTEGER32	ro	Sim
<i>Profile Position Mode</i>					
607Ah	VAR	Target Position	INTEGER32	rw	Sim
6083h	VAR	Profile Acceleration	UNSIGNED32	rw	Sim
6084h	VAR	Profile Deceleration	UNSIGNED32	rw	Sim
6086h	VAR	Motion Profile Type	INTEGER16	rw	Sim
<i>Profile Velocity Mode</i>					
6069h	VAR	Velocity Sensor Actual Value	INTEGER32	ro	Sim
606Bh	VAR	Velocity Demand Value	INTEGER32	ro	Sim
606Ch	VAR	Velocity Actual Value	INTEGER32	ro	Sim
60FFh	VAR	Target Velocity	INTEGER32	rw	Sim
<i>Profile Torque Mode</i>					
6071h	VAR	Target Torque	INTEGER16	rw	Sim
6077h	VAR	Torque Actual Value	INTEGER16	ro	Sim
6087h	VAR	Torque Slope	UNSIGNED32	rw	Sim
6088h	VAR	Torque Profile Type	INTEGER16	rw	Sim

Tabela 25: Lista de objetos do SCA-05 - Drive Profile

Sempre que um objeto desta lista for lido ou escrito, o SCA-05 irá mapear suas funções nos parâmetros do drive. Desta forma, ao operar o sistema através destes objetos, o valor dos parâmetros podem ser alterados de acordo com a função utilizada. Nos itens seguintes, é feita uma descrição detalhada de cada um destes objetos, onde são indicados os parâmetros do servo que são utilizados para executar as funções destes objetos.

6.1 Device Control - Objetos para controle do drive

Todo o drive que opera em uma rede CANopen seguindo a especificação *DSP 402* deve estar de acordo com o descrito pela seguinte máquina de estados:

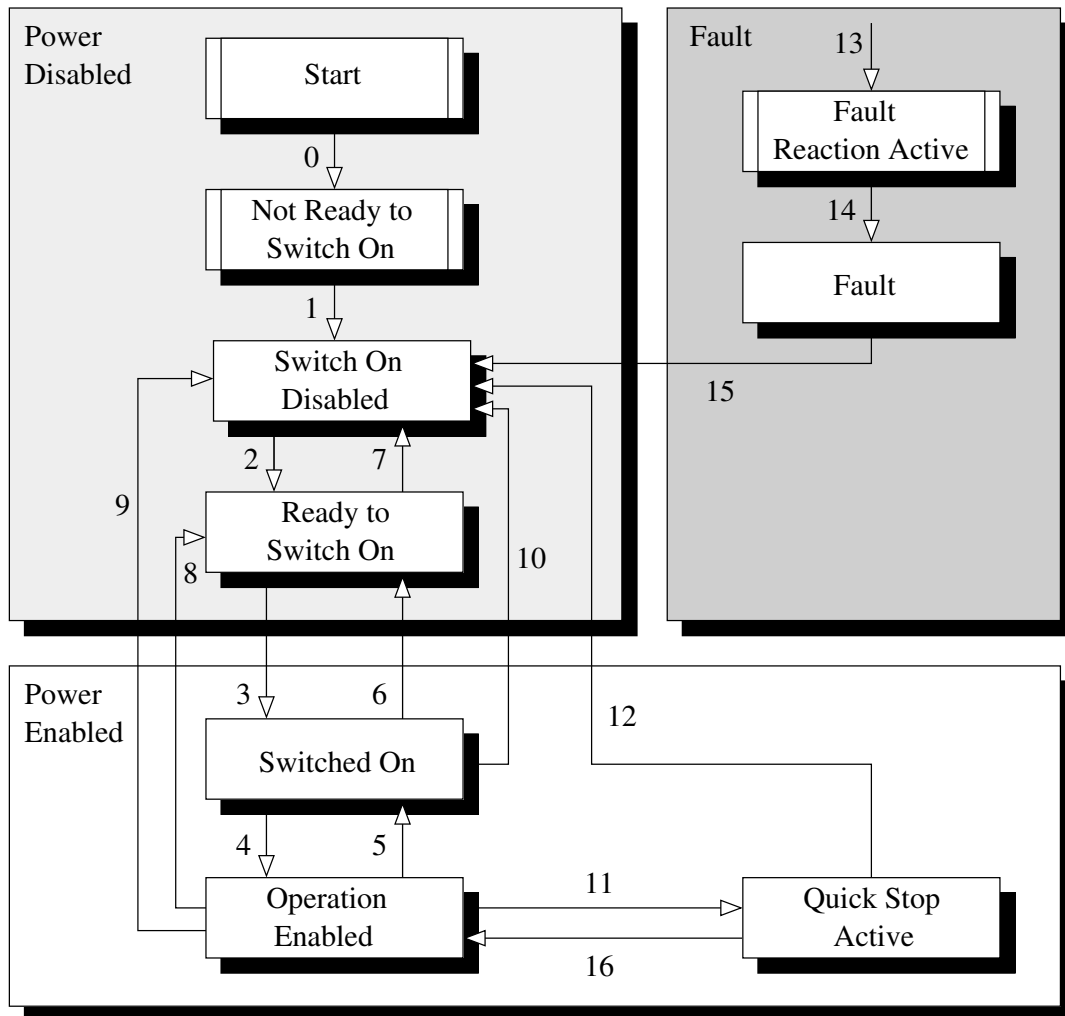


Figura 13: Máquina de estados do SCA-05

Descrição dos estados:

- *Not ready to switch on*: O drive está sendo inicializado, não pode ser comandado.
- *Switch on disabled*: Inicialização completa, drive pode receber comandos.
- *Ready to switch on*: Comando para permitir a alimentação do drive foi recebido.
- *Switched on*: Comando para energizar a potência do servo foi recebido.
- *Operation enabled*: O drive está habilitado, sendo controlado de acordo com o modo de operação programado. Potência está sendo aplicada ao motor pelo servo.
- *Quick stop active*: Durante a operação, o comando de *quick stop* foi recebido. Potência está sendo aplicada ao motor pelo servo.
- *Fault reaction active*: Uma falha ocorreu e o drive está executando a ação relativa ao tipo de erro.

- *Fault*: Servoconversor com erro. Função desabilitada, sem potência sendo aplicada no motor.



NOTA!

O servoconversor SCA-05 não possui chave para bloqueio / habilitação da alimentação de potência do equipamento. Desta forma, os estados descritos no grupo *Power disabled* foram implementados por questões de compatibilidade com a máquina de estados descrita, porém a alimentação de potência do dispositivo permanece ativa mesmo nestes estados.

Descrição das transições:

- *Transição 0*: O drive é ligado e inicia o procedimento de inicialização. A alimentação de potência do drive está ativa.
- *Transição 1*: Inicialização completa (automático).
- *Transição 2*: Comando *Shutdown* recebido. É feita a transição de estados, mas nenhuma ação é tomada pelo SCA-05.
- *Transição 3*: Comando *Switch on* recebido. É feita a transição de estados, mas nenhuma ação é tomada pelo SCA-05.
- *Transição 4*: Comando *Enable operation* recebido. O drive é habilitado. Equivale à escrita de $P099 = 1$.
- *Transição 5*: Comando *Disable operation* recebido. O drive é desabilitado. Equivale à escrita de $P099 = 0$.
- *Transição 6*: Comando *Shutdown* recebido. É feita a transição de estados, mas nenhuma ação é tomada pelo SCA-05.
- *Transição 7*: Comandos *Quick stop* e *Disable voltage* recebidos. É feita a transição de estados, mas nenhuma ação é tomada pelo SCA-05.
- *Transição 8*: Comando *Shutdown* recebido. Durante a operação do drive este é desabilitado, bloqueando a alimentação para o motor. Equivale à escrita de $P099 = 0$.
- *Transição 9*: Comando *Shutdown* recebido. Durante a operação do drive este é desabilitado, bloqueando a alimentação para o motor. Equivale à escrita de $P099 = 0$.
- *Transição 10*: Comando *Quick stop* ou *Disable voltage* recebido. É feita a transição de estados, mas nenhuma ação é tomada pelo SCA-05.
- *Transição 11*: Comando *Quick stop* recebido. Servo executa a função STOP, que equivale a $P432 = 1$.
- *Transição 12*: Comando *Disable voltage* recebido. O drive é desabilitado. Equivale à escrita de $P099 = 0$.
- *Transição 13*: Erro é detectado e o drive é desabilitado.
- *Transição 14*: Depois de desabilitar o servo, ele vai para o estado de erro (automático).

- *Transição 15*: Comando *Fault reset* recebido. O servo executa o reset do erro e retorna para o estado desabilitado e sem falha.
- *Transição 16*: Comando *Enable operation* recebido. A função STOP é desativada, e o servo retorna para a operação. Equivale a $P432 = 0$.

Esta máquina de estado é controlada pelo objeto 6040h, e os estados podem ser monitorados pelo objeto 6041h. Ambos os objetos são descritos a seguir.

6.1.1 Objeto 6040h - Controlword

Controla o estado do servoconversor.

<i>Valor padrão</i>	0000h
<i>Mínimo</i>	0000h
<i>Máximo</i>	FFFFh
<i>Tipo</i>	UNSIGNED16
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P099, P432

Os bits desta palavra possuem as seguintes funções:

15 - 9	8	7	6 - 4	3	2	1	0
<i>Reservado</i>	<i>Halt</i>	<i>Fault reset</i>	<i>Operation mode specific</i>	<i>Enable operation</i>	<i>Quick stop</i>	<i>Enable voltage</i>	<i>Switch on</i>

Os bits 0, 1, 2, 3 e 7 permitem controlar a máquina de estados do drive. Os comandos para transição de estados são dados através das combinações de bits indicadas na tabela 26. Os bits marcados com 'x' são irrelevantes para a execução do comando.

Comando	Bits da palavra de controle					Transições
	Fault reset	Enable operation	Quick stop	Enable voltage	Switch on	
Shutdown	0	×	1	1	0	2, 6, 8
Switch on	0	0	1	1	1	3
Disable voltage	0	×	×	0	×	7, 9, 10, 12
Quick stop	0	×	0	1	×	7, 10, 11
Disable operation	0	0	1	1	1	5
Enable operation	0	1	1	1	1	4, 16
Fault reset	0 → 1	×	×	×	×	15

Tabela 26: Comandos da palavra de controle

Os bits 4, 5, 6 e 8 possuem diferentes funções de acordo com o modo de operação utilizado. A descrição detalhada da função destes bits para os modos velocidade (*Profile velocity mode*) e torque (*Profile torque mode*) está descrita respectivamente nos itens 6.5.1 e 6.6.1.

Bit	Modo de operação	
	Profile velocity mode	Profile torque mode
4	reservado	reservado
5	reservado	reservado
6	reservado	reservado
8	Halt	Halt

Tabela 27: Bits da palavra de controle específicos do modo de operação

6.1.2 Objeto 6041h - Statusword

Indica o estado atual do SCA-05.

<i>Valor padrão</i>	-
<i>Mínimo</i>	0000h
<i>Máximo</i>	FFFFh
<i>Tipo</i>	UNSIGNED16
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P002, P006, P099

Os bits desta palavra possuem as seguintes funções:

<i>Bit</i>	<i>Descrição</i>
0	Ready to switch on
1	Switched on
2	Operation enabled
3	Fault
4	Voltage enabled
5	Quick stop
6	Switch on disabled
7	Warning
8	Reservado
9	Remote
10	Target reached
11	Internal limit active
12- 13	Operation mode specific
14 - 15	Reservado

Nesta palavra, os bits 0, 1, 2, 3, 5 e 6 indicam o estado do dispositivo de acordo com a máquina de estados descrita na figura 13. A tabela 28 descreve as combinações destes bits para indicação dos estados. Os bits marcados com 'x' são irrelevantes para a indicação do estado.

Valor (binário)	Estado
xxxx xxxx x0xx 0000	Not ready to switch on
xxxx xxxx x1xx 0000	Switch on disabled
xxxx xxxx x01x 0001	Ready to switch on
xxxx xxxx x01x 0011	Switched on
xxxx xxxx x01x 0111	Operation enabled
xxxx xxxx x00x 0111	Quick stop active
xxxx xxxx x0xx 1111	Fault reaction active
xxxx xxxx x0xx 1000	Fault

Tabela 28: Estados do servo indicados através da palavra de estado

Demais bits indicam uma condição específica para o drive.

- *Bit 4 - Voltage enabled*: Indica que a potência do drive está sendo alimentada. Está sempre ativo para o SCA-05.
- *Bit 7 - Warning*: Não utilizado para o SCA-05.
- *Bit 9 - Remote*: Indica quando o drive aceita comandos via rede CANopen. Está sempre habilitado para o SCA-05.
- *Bit 10 - Target reached*: Indica quando o drive está operando no valor da referência, que depende do modo de operação utilizado. Também é colocado em 1 quando as funções *quick stop* ou *halt* são acionadas.
- *Bit 11 - Internal limit active*: Indica quando o drive está operando em um valor limite interno, como limite de corrente.
- *Bits 12 e 13 - Operation mode specific*: Dependem do modo de operação do drive.

Bit	Modo de operação	
	Profile velocity mode	Profile torque mode
12	Speed	reservado
13	Max slippage error	reservado

Tabela 29: Bits da palavra de estado específicos do modo de operação

6.1.3 Objeto 6060h - Modes of Operation

Permite selecionar o modo de operação do servo.

<i>Valor padrão</i>	3
<i>Mínimo</i>	80h
<i>Máximo</i>	07h
<i>Tipo</i>	INTEGER8
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P202

Os valores aceitos por este objeto são descritos na tabela a seguir. Demais valores são reservados.

<i>Valor</i>	<i>Modo de operação</i>
3	Profile Velocity Mode
4	Profile Torque Mode

6.1.4 Objeto 6061h - Modes of Operation Display

O modo de operação utilizado pelo SCA-05 pode ser monitorado através deste objeto. Os valores que descrevem os modos de operação são os mesmos são os mesmos utilizados pelo objeto 6060h.

<i>Valor padrão</i>	-
<i>Mínimo</i>	80h
<i>Máximo</i>	7Fh
<i>Tipo</i>	INTEGER8
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P202

6.2 Factor Group - Objetos para conversão de unidades

Este grupo de objetos permite fazer a conversão de unidades para os objetos que representam valores de posição, velocidade ou aceleração. Estes valores, que são utilizados pelo servoconversor nos diferentes modos de operação, terão sua escala e dimensão definidas de acordo com os valores de notação e dimensão programados, conforme descritos a seguir.

6.2.1 Objeto 6089h - Position Notation Index

Este objeto determina a resolução decimal para a representação de posição.

<i>Valor padrão</i>	00h
<i>Mínimo</i>	00h
<i>Máximo</i>	00h
<i>Tipo</i>	INTEGER8
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Parâmetros utilizados</i>	-

Para representação de posição, somente o valor 0 (zero) é aceito para este objeto.

6.2.2 Objeto 608Ah - Position Dimension Index

Este objeto especifica a unidade de posição utilizada.

<i>Valor padrão</i>	FFh
<i>Mínimo</i>	00h
<i>Máximo</i>	FFh
<i>Tipo</i>	UNSIGNED8
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Parâmetros utilizados</i>	-

Valores possíveis para este objeto são:

<i>Valor</i>	<i>Descrição</i>
41h	graus
42h	minutos
43h	segundos
FFh	unidades internas (pulsos, onde 1 rev. = 65536 pulsos)



NOTA!

Escalas de posição programadas pelo usuário podem ter resolução diferente do que as escalas utilizadas internamente pelo servoconversor. Neste caso, durante a conversão dos valores de posição (de unidades internas para unidades do usuário e vice-versa), estes valores poderão ser truncados, ocasionando pequenas diferenças nos valores calculados.

6.2.3 Objeto 608Bh - Velocity Notation Index

Este objeto determina a resolução decimal para a representação de velocidade.

<i>Valor padrão</i>	00h
<i>Mínimo</i>	00h
<i>Máximo</i>	00h
<i>Tipo</i>	INTEGER8
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Parâmetros utilizados</i>	-

Para representação de velocidade, somente o valor 0 (zero) é aceito para este objeto.

6.2.4 Objeto 608Ch - Velocity Dimension Index

Este objeto especifica a unidade de velocidade utilizada.

<i>Valor padrão</i>	FFh
<i>Mínimo</i>	00h
<i>Máximo</i>	FFh
<i>Tipo</i>	UNSIGNED8
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Parâmetros utilizados</i>	-

Valores possíveis para este objeto são:

<i>Valor</i>	<i>Descrição</i>
A3h	rev/s
A4h	rev/min
A5h	rev/h
FFh	unidades internas (valor prop. à velocidade, onde 7FFF.FFFFh = 10000 rpm)



NOTA!

Escalas de velocidade programadas pelo usuário podem ter resolução diferente do que as escalas utilizadas internamente pelo servoconversor. Neste caso, durante a conversão dos valores de velocidade (de unidades internas para unidades do usuário e vice-versa), estes valores poderão ser truncados, ocasionando pequenas diferenças nos valores calculados.

6.2.5 Objeto 608Dh - Acceleration Notation Index

Este objeto determina a resolução decimal para a representação de aceleração.

<i>Valor padrão</i>	00h
<i>Mínimo</i>	00h
<i>Máximo</i>	00h
<i>Tipo</i>	INTEGER8
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Parâmetros utilizados</i>	-

Para representação de aceleração, somente o valor 0 (zero) é aceito para este objeto.

6.2.6 Objeto 608Eh - Acceleration Dimension Index

Este objeto especifica a unidade de aceleração utilizada.

<i>Valor padrão</i>	FFh
<i>Mínimo</i>	00h
<i>Máximo</i>	FFh
<i>Tipo</i>	UNSIGNED8
<i>Acesso</i>	rw
<i>Mapeável</i>	Não
<i>Parâmetros utilizados</i>	-

Valores possíveis para este objeto são:

<i>Valor</i>	<i>Descrição</i>
AFh	rev/s ²
B0h	rev/min ²
FFh	unidades internas (ms/krpm, mesma escala que P100 e P101)



NOTA!

Escala de aceleração programadas pelo usuário podem ter resolução diferente do que as escalas utilizadas internamente pelo servoconversor. Neste caso, durante a conversão dos valores de aceleração (de unidades internas para unidades do usuário e vice-versa), estes valores poderão ser truncados, ocasionando pequenas diferenças nos valores calculados.

6.3 Position Control Function - Controlador de Posição

Este grupo de objetos é utilizado para descrever o funcionamento do controlador de posição em malha fechada.

6.3.1 Objeto 6063h - Position Actual Value

Representa a posição atual do eixo do motor, em incrementos. Um volta completa representa 65536 incrementos.

<i>Valor padrão</i>	0000.0000h
<i>Mínimo</i>	8000.0000h
<i>Máximo</i>	0000.FFFFh
<i>Tipo</i>	INTEGER32
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	-

O valor deste objeto representa sempre a posição do eixo em uma volta apenas. O número de voltas não é contado por este objeto.

6.3.2 Objeto 6064h - Position Actual Value in user units

Representa a posição atual do eixo do motor. O valor deste objeto pode ser transformado de unidades internas para valores definidos pelo usuário, de acordo com o programado nos objetos 6089h e 608Ah.

<i>Valor padrão</i>	0000.0000h
<i>Mínimo</i>	8000.0000h
<i>Máximo</i>	7FFF.FFFFh
<i>Tipo</i>	INTEGER32
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	-

O valor deste objeto representa sempre a posição do eixo em uma volta apenas. O número de voltas não é contado por este objeto.

6.4 Profile Position Mode - Modo Posicionamento

O SCA-05 não suporta completamente este modo de acordo com o descrito pela especificação DSP 402. Porém alguns objetos definidos neste módulo estão disponíveis pois são utilizados também por outros modos de operação.

6.4.1 Objeto 607Ah - Target Position

Permite programar a referência de posição para o servo no modo posicionamento. A escala utilizada para escrita neste objeto depende dos valores programados nos objetos 6089h e 608Ah.

<i>Valor padrão</i>	0000.0000h
<i>Mínimo</i>	8000.0001h
<i>Máximo</i>	7FFF.FFFFh
<i>Tipo</i>	UNSIGNED32
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	-

6.4.2 Objeto 6083h - Profile Acceleration

Permite programar a rampa de aceleração até que o eixo do motor atinja a velocidade programada. A escala pode ser programada nos objetos 608Dh e 608Eh.

<i>Valor padrão</i>	0000.0000h
<i>Mínimo</i>	8000.0001h
<i>Máximo</i>	7FFF.FFFFh
<i>Tipo</i>	UNSIGNED32
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P100

6.4.3 Objeto 6084h - Profile Deceleration

Permite programar a rampa de desaceleração até que o eixo do motor atinja a velocidade zero. A escala pode ser programada nos objetos 608Dh e 608Eh.

<i>Valor padrão</i>	0000.0000h
<i>Mínimo</i>	8000.0001h
<i>Máximo</i>	7FFF.FFFFh
<i>Tipo</i>	UNSIGNED32
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P101

O valor da desaceleração é utilizado para parar o motor quando acionadas as funções *Halt* e *Quick stop*.

6.4.4 Objeto 6086h - Motion Profile Type

Permite programar o perfil da rampa de aceleração e desaceleração para o drive.

<i>Valor padrão</i>	FFFFh
<i>Mínimo</i>	FFFFh
<i>Máximo</i>	0000h
<i>Tipo</i>	INTEGER16
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P229

Valores possíveis para este objeto são:

<i>Valor</i>	<i>Descrição</i>
FFFFh	Sem rampa
0000h	Rampa linear

6.5 Profile Velocity Mode - Modo Velocidade

Este modo de operação permite o controle de velocidade do motor através de diversas sub-funções, tais como:

- Valor de referência via gerador de perfil de velocidade.
- Captura e monitoração da velocidade.
- Sistema de controle de velocidade.
- Limitação de torque do motor.

Estas funções são executadas com base em um conjunto de objetos para configuração deste modo de operação.

6.5.1 Bits de controle e estado

Os bits 4, 5, 6 e 8 da palavra de controle (objeto 6040h - *Controlword*) possuem as seguintes funções no modo velocidade:

Bit	Nome	Valor	Descrição
4	reservado		
5	reservado		
6	reservado		
8	Halt	0	Executa movimento
		1	Pára eixo (P432 = 1)

Para a palavra de estado (objeto 6041h - *Statusword*), o comportamento para os bits 10, 12 e 13 é descrito a seguir:

Bit	Nome	Valor	Descrição
10	Target reached	0	Halt = 0, referência de velocidade não alcançada Halt = 1, velocidade diferente de 0 (zero)
		1	Halt = 0, referência de velocidade alcançada Halt = 1, velocidade igual a 0 (zero)
12	Speed	0	Velocidade diferente de 0 (zero)
		1	Velocidade igual a 0 (zero)
13	Max slippage error	Não implementado	

6.5.2 Objeto 6069h - Velocity Sensor Actual Value

Permite a leitura do valor do sensor utilizado para medir a velocidade do motor. O SCA-05 utiliza um resolver como sensor de posição (a velocidade angular é obtida derivando este valor no tempo), logo o este sensor fornece um valor proporcional à posição angular. O sensor possui resolução de 14 bits, e uma volta completa fornece 16384 valores diferentes de posição.

<i>Valor padrão</i>	-
<i>Mínimo</i>	0000.0000h
<i>Máximo</i>	0000.3FFFh
<i>Tipo</i>	INTEGER32
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P050

6.5.3 Objeto 606Bh - Velocity Demand Value

Indica a velocidade fornecida pelo gerador de trajetória do servoconversor, utilizada pelo regulador de velocidade para controle do motor. O valor fornecido por este objeto depende da escala programada nos objetos 608Bh e 608Ch.

<i>Valor padrão</i>	-
<i>Mínimo</i>	8000.0001h
<i>Máximo</i>	7FFF.FFFFh
<i>Tipo</i>	INTEGER32
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	-

6.5.4 Objeto 606Ch - Velocity Actual Value

Indica a velocidade do motor. O valor fornecido por este objeto depende da escala programada nos objetos 608Bh e 608Ch.

<i>Valor padrão</i>	-
<i>Mínimo</i>	8000.0001h
<i>Máximo</i>	7FFF.FFFFh
<i>Tipo</i>	INTEGER32
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P002

6.5.5 Objeto 60FFh - Target Velocity

Permite programar a referência de velocidade para o servoconversor no modo velocidade. A escala utilizada para escrita neste objeto depende dos valores programados nos objetos 608Bh e 608Ch.

<i>Valor padrão</i>	0000.0000h
<i>Mínimo</i>	8000.0001h
<i>Máximo</i>	7FFF.FFFFh
<i>Tipo</i>	INTEGER32
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P121

6.6 Profile Torque Mode - Modo Torque

Este modo possibilita o controle de torque do drive via uma referência de torque recebida pela rede CANopen.

6.6.1 Bits de controle e estado

Os bits 4, 5, 6 e 8 da palavra de controle (objeto 6040h - *Controlword*) possuem as seguintes funções no modo torque:

<i>Bit</i>	<i>Nome</i>	<i>Valor</i>	<i>Descrição</i>
4	reservado		
5	reservado		
6	reservado		
8	Halt	0	Executa movimento
		1	Pára eixo (P432 = 1)

Para a palavra de estado (objeto 6041h - *Statusword*), o comportamento para os bits 10, 12 e 13 é descrito a seguir:

<i>Bit</i>	<i>Nome</i>	<i>Valor</i>	<i>Descrição</i>
10	Target reached	0	Referência de torque não alcançada
		1	Atigiu referência de torque
12	reservado		
13	reservado		

6.6.2 Objeto 6071h - Target Torque

Permite programar a referência de torque para o servoconversor no modo torque. A escala utilizada para escrita neste objetos é fornecida em partes por mil do torque nominal do motor.

<i>Valor padrão</i>	0000h
<i>Mínimo</i>	8001h
<i>Máximo</i>	7FFFh
<i>Tipo</i>	INTEGER16
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P119

6.6.3 Objeto 6077h - Torque actual value

Indica o torque atual do motor. O valor é fornecido em parte por mil do torque nominal do motor.

<i>Valor padrão</i>	-
<i>Mínimo</i>	8001h
<i>Máximo</i>	7FFFh
<i>Tipo</i>	INTEGER16
<i>Acesso</i>	ro
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P003

6.6.4 Objeto 6087h - Torque Slope

Permite programar a taxa de variação do torque no tempo (rampa de torque) para o servoconversor. A escala utilizada é de partes por mil do torque nominal do motor por segundo.

<i>Valor padrão</i>	0000.0000h
<i>Mínimo</i>	0000.0000h
<i>Máximo</i>	FFFF.FFFFh
<i>Tipo</i>	UNSIGNED32
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	-

6.6.5 Objeto 6088h - Torque Profile Type

Utilizado para indicar o perfil da rampa de torque do servoconversor.

<i>Valor padrão</i>	FFFFh
<i>Mínimo</i>	FFFFh
<i>Máximo</i>	0000h
<i>Tipo</i>	INTEGER16
<i>Acesso</i>	rw
<i>Mapeável</i>	Sim
<i>Parâmetros utilizados</i>	P229

Valores possíveis para este objeto são:

<i>Valor</i>	<i>Descrição</i>
FFFFh	Sem rampa
0000h	Rampa linear de torque

7 Erros da comunicação CANopen

A seguir serão descritos os erros do SCA-05 específicos da interface CAN e do protocolo CANopen.

7.1 E33 - Sem alimentação

Caso o protocolo CANopen esteja habilitado, é necessário fornecer uma tensão de alimentação para a interface CAN. No caso de falta desta tensão de alimentação, será indicado E33 na IHM do produto, e a comunicação CANopen será desabilitada. O protocolo somente volta a operar caso esta alimentação seja fornecida novamente.

7.2 E34 - *Bus off*

Quando uma quantidade muito grande de erros de comunicação é detectada por um dispositivo da rede CAN, este dispositivo pode ir para o estado de *bus off*, onde ele para de acessar o barramento. Para o SCA-05, caso ocorra esta condição, será mostrado E34 na IHM do produto. É importante observar que, caso o parâmetro P703 esteja programado para fazer o *reset* de *bus off* automaticamente, este erro será desconsiderado e não será indicado.

Diversos problemas podem ocasionar este erro. Alguns itens que podem ser observados para resolver este problema são:

- Programação incorreta da taxa de comunicação. É necessário que todos os dispositivos da rede CANopen estejam programados com a mesma taxa de comunicação. Caso algum dispositivo esteja com a programação incorreta, este pode causar erro em si próprio ou nos demais.
- Falta do resistor de terminação. É necessário habilitar resistores de terminação nos extremos do barramento, para que a comunicação possa ocorrer sem problemas.
- Instalação incorreta. Deve-se checar os cabos para saber se não existem ligações invertidas, se a passagem dos cabos está correta, e se o cabo e os dispositivos estão devidamente aterrados.

7.3 E35 - Erro de guarda do nó

Um dos serviços disponíveis para a rede CANopen é o de monitoramento do dispositivo pelo mestre da rede, onde telegramas periódicos são trocados para que possa-se assegurar que a comunicação esteja ocorrendo sem problemas (*Node Guarding*). Após iniciado a troca destes telegramas, caso a comunicação seja interrompida por um tempo maior do que o programado, o servo irá mostrar E35 na IHM do produto, indicando erro de guarda do nó.

Esta função é dependente das configurações feitas pelo mestre da rede. Eventuais erros que possam ocorrer podem ser decorrentes de problemas na comunicação, ou de programação inadequada do mestre ou dos objetos de comunicação através dos quais esta função é habilitada. Para a descrição detalhada desta função deve-se consultar o item [5.6.2](#).

8 Exemplos de utilização da interface CANopen

Nesta seção serão mostrados exemplos de parametrização e operação do SCA-05, utilizando objetos existentes no servo para operá-lo via rede. Para os exemplos, será considerada a seguinte configuração para a rede:

- *1 Mestre*: será utilizado um equipamento genérico, programável, com uma interface para a rede CANopen, que possua as funcionalidades de:
 - Mestre da rede (gerencia estados do dispositivos e *node guarding*)
 - Produtor SYNC
 - Consumidor EMCY
 - Cliente SDO
- *2 Escravos SCA-05*: serão utilizados 2 servoconvertidores como escravos da rede CANopen.

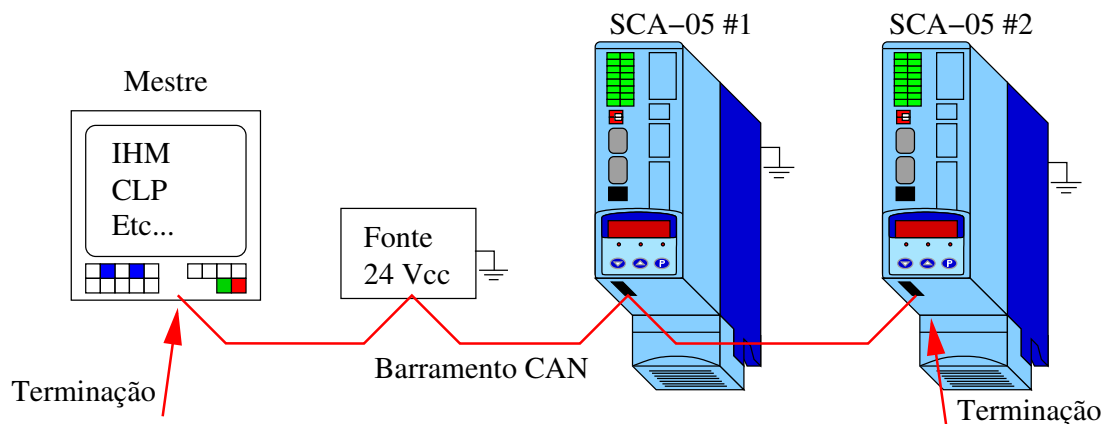


Figura 14: Rede utilizada nos exemplos de aplicação

Com relação à instalação, deve-se conectar resistores de terminação, no valor de 120Ω / 0.25W, nos dois extremos do barramento. Isto pode ser feito ligando o resistor diretamente no conector da rede, entre os pinos CAN.L e CAN.H. É necessário aterrar todos os pontos da rede, preferencialmente no mesmo ponto de terra, e utilizar uma fonte para alimentar a interface CAN via barramento.

Para a parametrização do SCA-05, os seguintes parâmetros são relevantes para a comunicação CANopen:

- P700: deve-se habilitar o protocolo CANopen para ambos os servoconvertidores. Para tanto, é necessário programar P700 igual a 1.
- P701: o Node-ID é programado no P701, e deve ser diferente para cada escravo da rede. Neste exemplo, um SCA-05 será programado para o endereço #1 e outro para o endereço #2.
- P702: considerando que o comprimento do cabo de rede seja menor que 40 metros, vamos utilizar a taxa de transmissão máxima, de 1 Mbit/s. Isto é feito programando P702 em 0 (zero).

- P703: o P703 pode ser programado de acordo com o desejado para a aplicação. Neste caso, vamos fazer com que o *reset* do erro *Bus off* seja feito de forma manual (P703 igual a 0 (zero)) .

Uma vez feito a parametrização, é importante desligar ou fazer o *reset* do SCA-05, para que as novas configurações sejam aceitas pelo servoconversor. Na sequência, pode-se observar, a título de diagnóstico, os seguintes parâmetros:

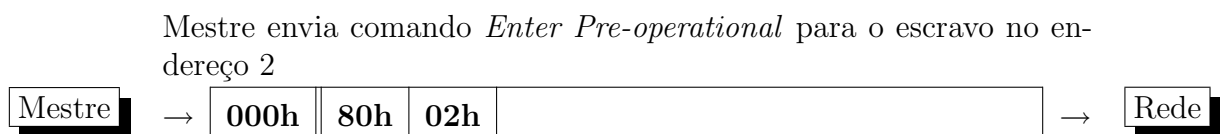
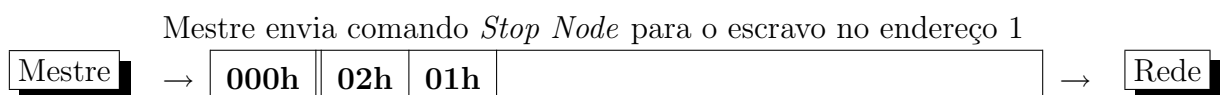
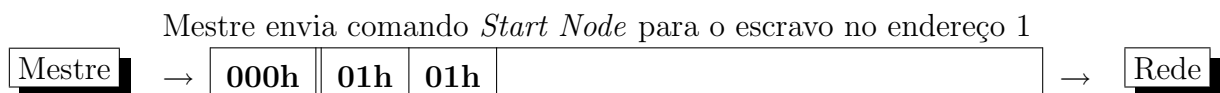
- P070: indica o estado da rede CAN. Se o protocolo CANopen estiver programado, o *baudrate* estiver correto e os equipamentos estiverem conectados na rede, o valor deste parâmetro deve estar em 2 (habilitado e sem erro).
- P075: o valor deste parâmetro deverá estar em 2, indicando que o protocolo CANopen foi inicializado sem problemas.
- P076: enquanto o SCA-05 não começar a ser operado pelo mestre da rede, este parâmetro deverá apresentar o valor 127, indicando que o servo está no estado pré-operacional.

Depois de concluída a instalação e parametrização, o SCA-05 está pronto para ser operado via rede CANopen.

8.1 Exemplo 1 - Controlando o estado de um dispositivo

No item 5.6.1 é apresentado um diagrama de estados que descreve o comportamento de um escravo da rede. Estes estados são controlados pelo mestre, através dos serviços de gerenciamento.

Após a inicialização, todos os escravos encontram-se no estado pré-operacional. O mestre pode então solicitar o *reset* ou enviar os escravos para cada um dos estados existentes, de acordo com o telegrama enviado. Exemplos destes telegramas são apresentados a seguir. Lembrar que os telegramas não possuem confirmação por parte do escravo, ou seja, somente o mestre envia as mensagens para a rede.



Mestre envia comando *Reset Node* para todos os escravos da rede



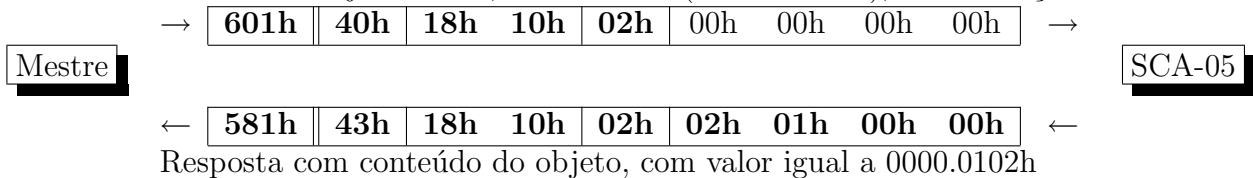
Observar que o COB-ID do telegrama é sempre 0 (zero). Todos os escravos receberão o telegrama, porém somente executará o comando (que é indicado no byte 1) o escravo com o Node-ID de destino (indicado no byte 2). Caso o byte 2 tenha valor 0 (zero), significa que é uma mensagem do tipo *broadcast*, e todos os escravos devem executar o comando.

8.2 Exemplo 2 - Acessando objetos utilizando SDOs

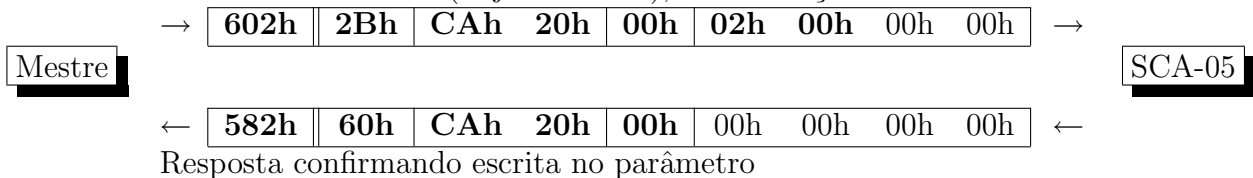
Neste exemplo serão mostrados telegramas do tipo SDO onde o mestre lê e escreve parâmetros e outros objetos dos escravos da rede. Lembrando, um telegrama SDO possui a seguinte estrutura:

COB-ID	Comando	Índice		Sub-índice	Dados do objeto			
11 bits	byte 0	byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7

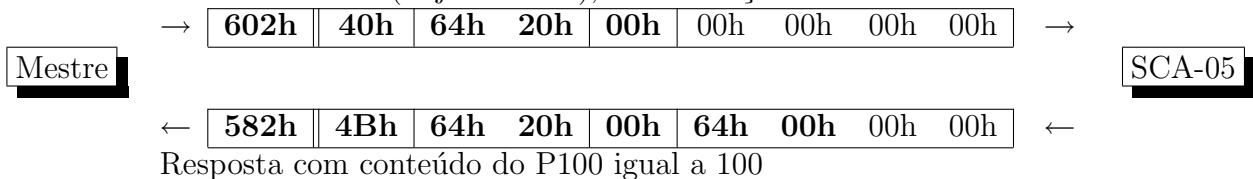
Leitura do objeto 1018h, sub-índice 2 (*Product code*), do endereço 1



Escrita de P202 = 2 (objeto 20CAh), no endereço 2



Leitura de P100 (objeto 2064h), do endereço 2



8.3 Exemplo 3 - Habilitando o controle de erros - *Guarding*

O serviço de controle de erros permite aos integrantes da rede detectar erros ocorridos em dispositivos remotos. O SCA-05 usa o serviço de guarda do nó para possibilitar esta detecção. Para poder configurar este serviço, é necessário primeiro definir os seguintes pontos:

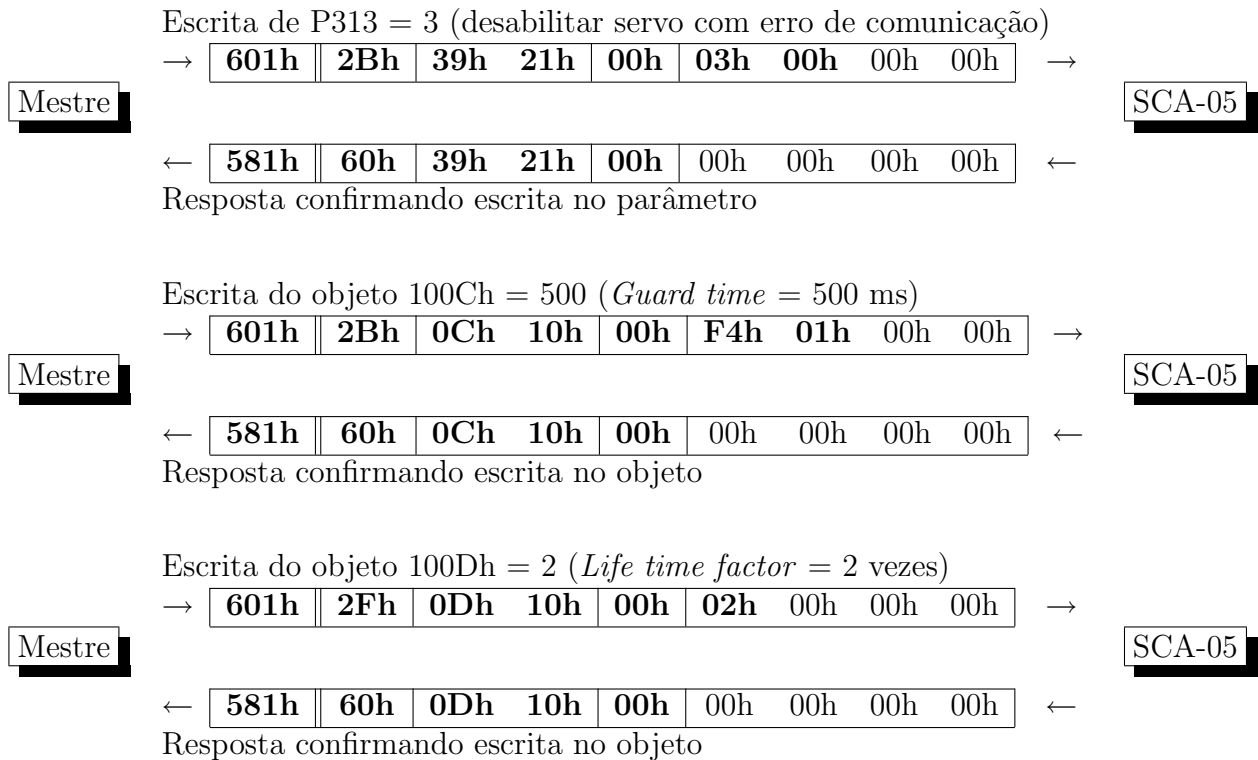
- Caso ocorra uma falha na rede, qual deve ser a ação tomada pelo SCA-05?
Para este exemplo, caso ocorra um erro de comunicação, o servo deverá ser desabilitado. Para isto, será programado o parâmetro P313 igual a 3, conforme descrito no item 3.7.

Isto pode ser programado diretamente pela IHM do equipamento, ou através da rede CANopen, utilizando um SDO para escrever no objeto 2139h (que equívale ao P313).

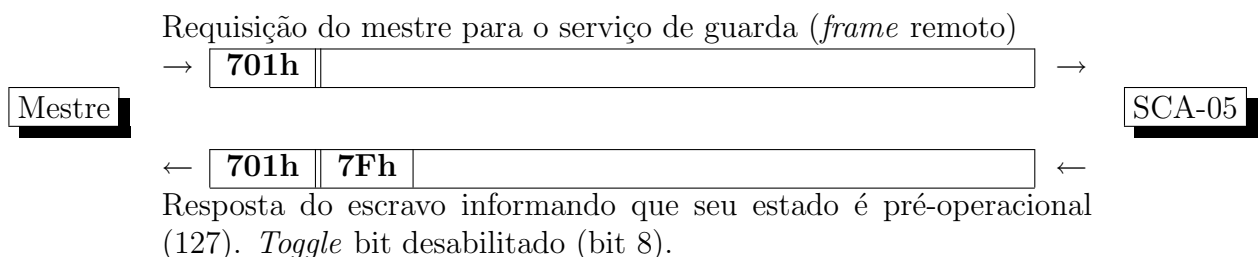
- Caso ocorra uma falha na rede, qual o tempo máximo que o SCA-05 pode levar para detectar este erro?

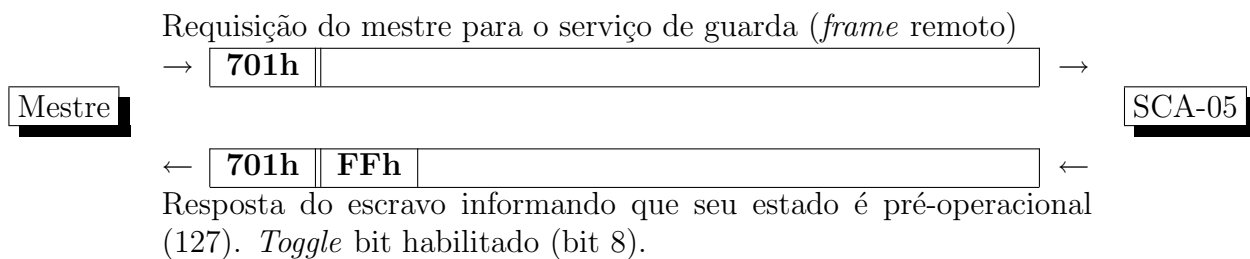
Para esta aplicação, será considerado que o tempo limite para detecção de erros de comunicação seja 1 segundo. Baseado neste tempo, pode-se definir valores para os objetos 100Ch - *Guard time*, e 100Dh - *Life time factor*, de maneira que o servo possa atuar no tempo programando. Neste caso, pode-se programar o objeto 100Ch igual a 500 (significando que o servo deve receber uma mensagem do mestre a cada 500 ms, caso contrário ele considerará como uma falha neste serviço), e o objeto 100Dh igual a 2 (significando que, caso ocorram duas falhas na sequência, será considerado erro na comunicação). Estes valores precisam necessariamente ser escritos utilizando o SDO do servo.

Estas configurações serão feitas para o escravo no endereço 1, porém a configuração do escravo no endereço 2 pode ser feita de maneira similar.



Uma vez configurados estes objetos, o SCA-05 está pronto para iniciar o serviço de guarda do nó. O mesmo serviço deve ser programado no mestre, para que os dispositivos possam operar em conjunto. Uma vez iniciada a comunicação pelo mestre, o servo começará a contar os tempos estipulados para o controle de erros.





Estes telegramas serão enviados ciclicamente. Uma vez habilitado este serviço, é possível verificar seu estado através do parâmetro P075 do SCA-05, que deverá conter o valor 3 (guarda do nó habilitada).

Se a comunicação for interrompida, caso o servo esteja habilitado, ele executará o comando de desabilita geral (P099 igual a 0 (zero)), e irá para o estado pré-operacional (caso não encontre-se neste estado). O parâmetro P075 assumirá o valor 4, indicando que a troca de mensagens de guarda foi interrompida por um tempo maior do que o programado no objetos 100Ch e 100Dh.

8.4 Exemplo 4 - Configurando um PDO para transmissão

Os PDOs de transmissão são utilizados para enviar dados de um determinado equipamento para a rede. Para o SCA-05, estas informações geralmente indicam o estado do equipamento (parado ou girando o motor, com erro ou sem erro, etc.), ou então informam alguma variável que deseja-se monitorar (velocidade, corrente, posição, etc.).

As informações que se deseja transmitir devem ser mapeadas em um dos quatro PDOs de transmissão disponíveis para o servo. Já existe um mapeamento padrão, porém outras informações podem ser acrescentadas ou retiradas, dependendo da aplicação. Primeiro deve-se ficar atento para as seguintes questões:

- Quais são as informações que o servo deve transmitir para a rede?

É necessário definir quais são as informações que serão enviadas pelo PDO. Para este exemplo, será transmitida as informações de estado do servo e de posição do eixo do motor, do SCA-05 no endereço 1.

- Em quais objetos estas informações estão disponíveis?

Estas informações podem ser obtidas tanto através dos parâmetros do servo, quanto através dos objetos específicos para drives, disponíveis a partir do índice 6000h. Como já existe um mapeamento padrão para o SCA-05 que utiliza os objetos específicos para drives, serão utilizados estes objetos. A informação de estado do servo pode ser obtida através do objeto 6041h (*Status Word*), e a informação da posição do eixo pode ser obtida através do objeto 6064h (*Position actual value*). Ambos os objetos são mapeáveis nos PDOs, conforme tabela 25.

- Qual será o COB-ID para a mensagem?

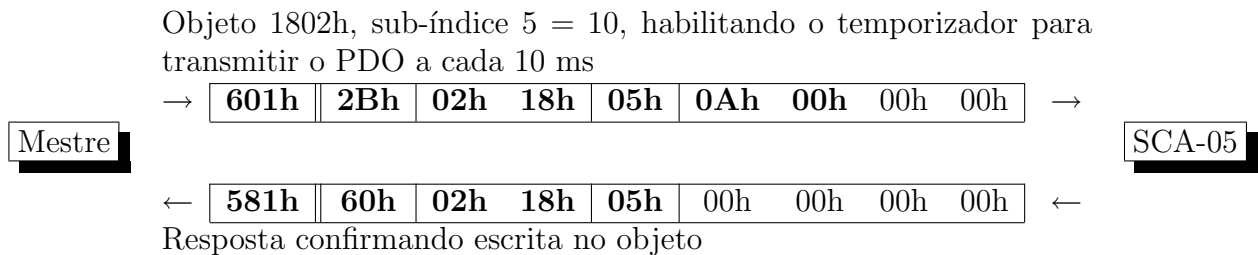
Se o servo vai transmitir este PDO, é porque um ou mais equipamentos vão receber esta informação. Desta forma, é importante saber o COB-ID da mensagem transmitida, para poder programar os PDOs de recepção que irão consumir a mensagem. Neste exemplo, será utilizado o terceiro PDO de transmissão, para o servo no endereço 1. O COB-ID para o PDO não será alterado, sendo mantido o seu valor padrão, que é igual a 381h (objeto 1802h, sub-índice 1).

- Como deve ser a transmissão deste PDO, e qual o período para essa transmissão?

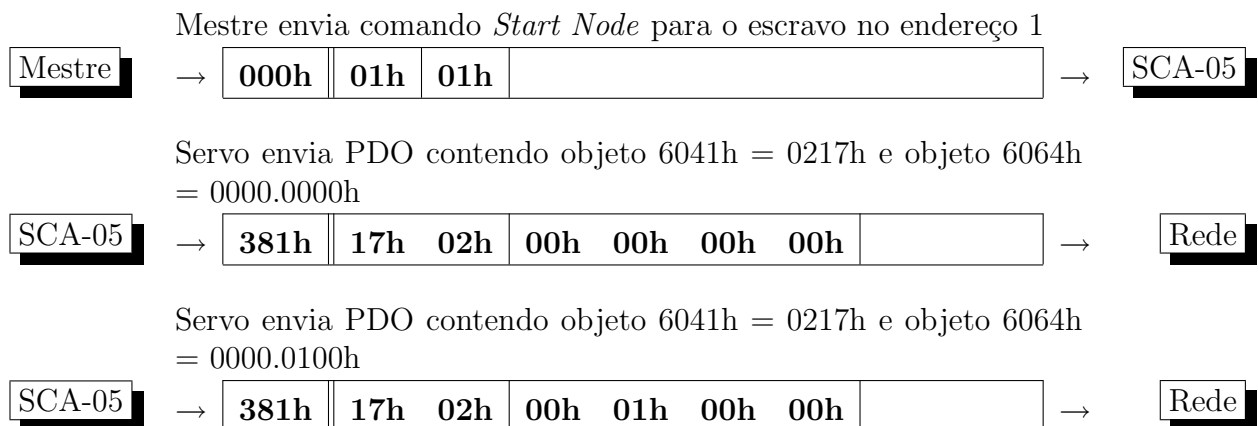
Existem diferentes formas de fazer com que o servo transmita um PDO: vincular este PDO a um telegrama SYNC, programar o temporizador do PDO para que ele seja transmitido periodicamente, ou então enviar telegramas remotos solicitando sua transmissão, conforme descrito na tabela 21. Neste caso, usaremos o temporizador para transmissão automática do PDO a cada 10 ms. O tipo de transmissão é programado no objeto PDO_COMM_PARAMETER do respectivo PDO, no sub-índice 3.

De acordo com o descrito, inicialmente deve feito o mapeamento dos objetos a serem transmitidos pelo PDO. Como o terceiro PDO de transmissão foi escolhido, este já possui no seu mapeamento padrão as informações de estado do servo e de posição do eixo do motor, que são as informações desejadas para transmissão. Desta forma não será necessário a alteração do mapeamento deste TPDO.

Definido o mapeamento, é necessário configurar o temporizador do PDO para que ele seja transmitido a cada 10 ms. Isto é configurado no objeto 1800h, sub-índice 5. Os sub-índices 2 e 3 serão mantidos com seus valores padrão.



Feito isto, o SCA-05 está pronto para iniciar a transmissão do seu primeiro PDO, contendo as informações de estado e posição. Porém somente será iniciada a transmissão após o mestre enviar o comando para o escravo entrar no estado operacional, pois este é o único estado em que o escravo pode transmitir ou receber PDOs.



Desta forma, estas informações serão enviadas para a rede a cada 10 ms. Caso um ou mais dispositivos da rede queiram receber estes dados, basta programar um PDO de recepção com o COB-ID desta mensagem, e fazer o mapeamento de acordo com o desejado. O conteúdo dos objetos 6041h e 6064h deve ser interpretado de acordo com sua descrição no item 6.

8.5 Exemplo 5 - Configurando um PDO para recepção

Os PDOs de recepção do SCA-05 são utilizados principalmente para recepção de dados de controle e referências para a operação do servo. A configuração de um PDO de recepção também depende da aplicação, e para isto deve-se ter as seguintes informações:

- Quais são as informações que o servo deve receber da rede?

É necessário definir quais são as informações que serão recebidas pelo PDO. Neste caso, será utilizado o servo no endereço 2 da rede, programado para receber os comandos de habilitação e referência de velocidade (assumindo que o SCA-05 está programando para modo velocidade, onde $P202 = 2$).

- Em quais objetos estas informações estão disponíveis?

Neste caso serão utilizados os objetos 6040h e 60FFh, que representam respectivamente a palavra de controle do servo e a referência de velocidade para o motor. Ambos os objetos são mapeáveis nos PDOs, conforme tabela 25.

- Qual será o COB-ID para a mensagem?

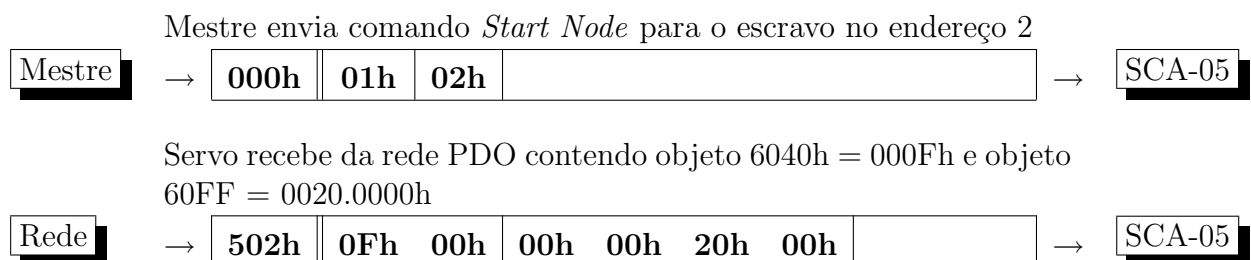
Será utilizado o quarto PDO de recepção do SCA-05, e para este RPDO também será mantido seu COB-ID padrão, igual a 502h (objeto 1403h, sub-índice 1). Este PDO somente irá receber da rede mensagens cujo COB-ID seja igual ao seu, e para que isso aconteça é necessário que um outro elemento da rede produza mensagens com este COB-ID.

- Como deve ser a recepção deste PDO?

Conforme a tabela 21, um PDO de recepção pode estar vinculado ao telegrama SYNC ou não. Neste exemplo vamos manter o tipo de transmissão para o RPDO com seu valor padrão, de 254, indicando que não possui relação com telegramas SYNC. Desta forma, sempre que um telegrama for recebido, os valores dos objetos mapeados serão automaticamente atualizados.

Baseado nestas informações, o primeiro passo é fazer o mapeamento dos dados escolhidos. Como será utilizado o quarto PDO de recepção, o seu mapeamento já possui as informações de controle e referência de velocidade no seu mapeamento padrão e desta forma também não será necessário alterar este mapeamento.

Definido o mapeamento, ainda é necessário enviar o SCA-05 para o estado operacional. Uma vez neste estado, todo o telegrama cujo COB-ID for igual a 502h será recebido pelo servo e os valores dos objetos mapeados serão atualizados.



O conteúdo para os objetos 6040h e 60FFh será interpretado pelo servo de acordo com a sua descrição no item 6.

8.6 Exemplo 6 - Utilizando o objeto SYNC para envio de PDOs

O objeto SYNC pode ser utilizado como forma de sincronizar a transmissão ou recepção de dados pelos dispositivos da rede. Para utilizar o SYNC nos servos, é necessário que algum dispositivo da rede seja o produtor deste objeto. Primeiramente serão definidos os seguintes pontos:

- Qual o período que deve ser utilizado pelo produtor SYNC para a transmissão do telegrama?

Neste exemplo, o período para a transmissão de um telegrama SYNC será de 100 ms. Esta programação é feita apenas no produtor SYNC, que neste caso é o mestre da rede. O COB-ID para este telegrama será 080h, que é o valor padrão.

- Existe algum PDO de transmissão que deva estar vinculado a este objeto? Estes PDOs devem estar programados para serem transmitidos a cada SYNC, ou em valores múltiplos de SYNC?

Serão programados dois servos, e cada um terá um PDO de transmissão vinculado ao telegrama SYNC. O SCA-05 no endereço 1 deve enviar seu PDO a cada telegrama SYNC detectado, enquanto que o SCA-05 no endereço 2 deve enviar seu PDO a cada 2 telegramas SYNC.

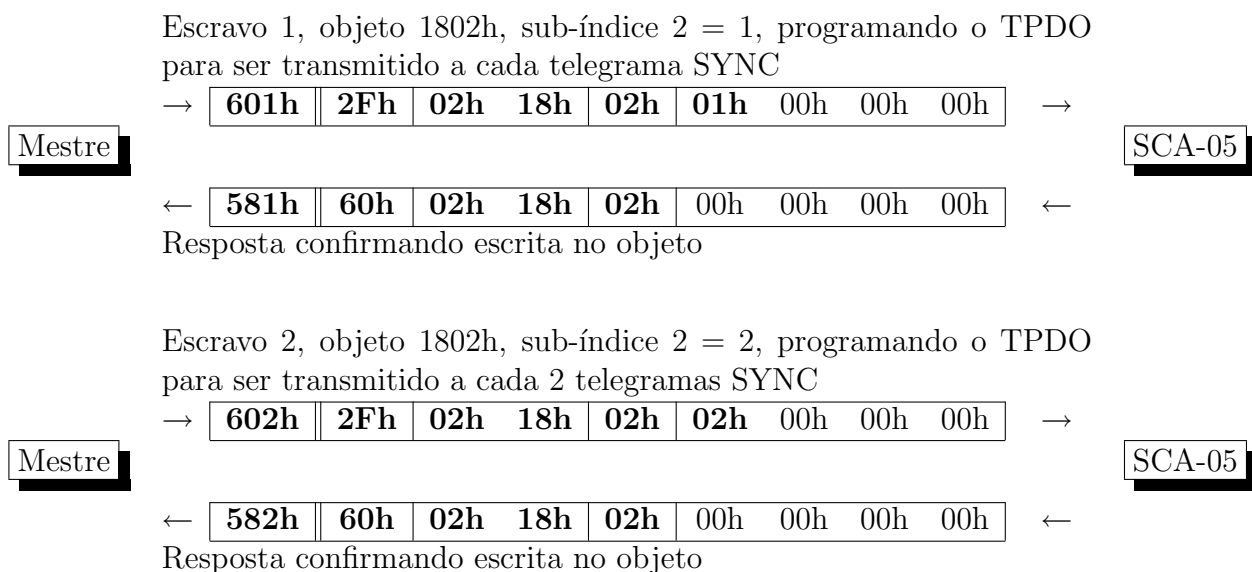
- Quais os dados devem ser transmitidos por cada PDO?

Para que não haja necessidade de mapeamento dos PDOs, será utilizado o terceiro TPDO de cada servo, que possui as informações de estado e posição do servo. Estes serão os objetos transmitidos pelos PDOs.

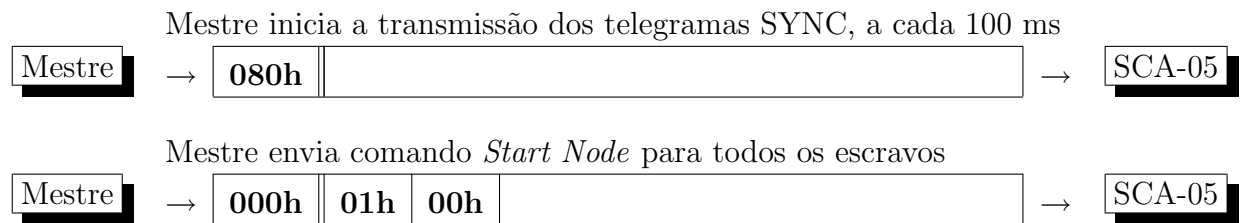
- Existe algum PDO de recepção que deva estar vinculado ao SYNC?

Neste exemplo não serão utilizados PDOs de recepção vinculados ao objeto de sincronização.

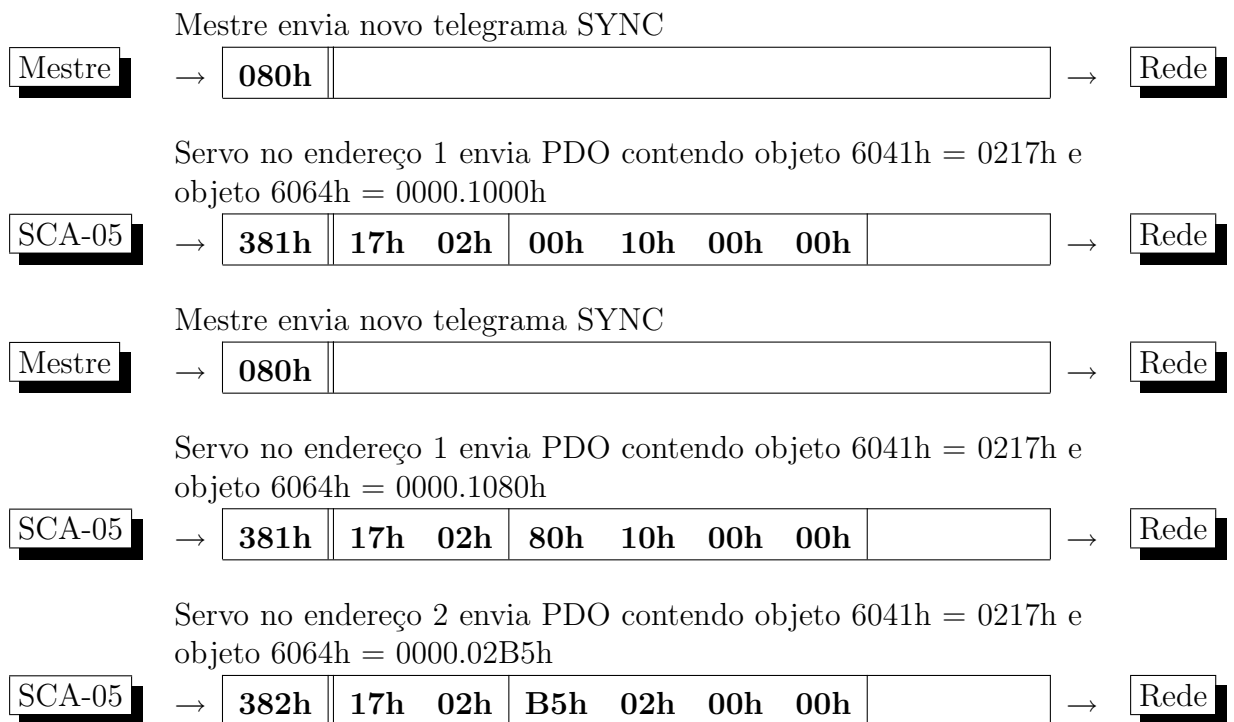
Deve-se então fazer a configuração dos PDOs utilizando o SDO de cada escravo:



Uma vez configurado, é necessário habilitar no mestre a produção de telegramas SYNC, que pode ser transmitidos periodicamente. Porém os escravos não transmitirão seus PDOs até serem colocados no modo operacional, comando este também enviado pelo mestre.



Ao colocar os escravos no modo operacional, o escravo no endereço 1 enviará um PDO a cada telegrama SYNC, enquanto que o escravo número 2 enviará um PDO a cada 2 telegramas SYNC recebidos. Estes telegramas serão repetidos indefinidamente, enquanto o sistema estiver em operação. Os dados transmitidos pelos PDOs poderão ser consumidos por outros dispositivos da rede, de acordo com a lógica de operação desejada.



No instante em que o telegrama SYNC for detectado, cada TPDO irá salvar os valores de estado e posição, que serão transmitidos em sequência. O conteúdo destes objetos deve ser interpretado de acordo com o descrito no item 6.

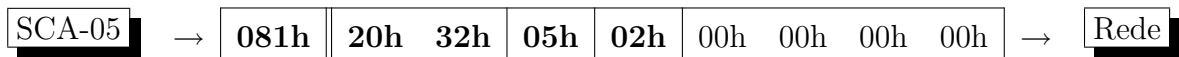
Da mesma forma como o telegrama SYNC é utilizado por PDOs de transmissão, estes também poderiam ser utilizados por PDOs de recepção para sincronizar o recebimento de dados pelos dispositivos da rede. Basta programar um RPDO para o tipo síncrono, que os dados recebidos por este PDO somente serão atualizados no dicionário de objetos quando o próximo telegrama SYNC for detectado.

8.7 Exemplo 7 - Detecção de erros utilizando EMCY

Os telegramas EMCY podem ser utilizados para sinalizar a ocorrência de um erro em um dispositivo da rede. Os servos possuem um produtor EMCY que, sempre que ocorrer uma falha no dispositivo, ele envia uma mensagem para a rede contendo o tipo de erro ocorrido. Caso o mestre da rede possua um consumidor EMCY que esteja monitorando este evento, poderá detectar o erro, e fazer a sinalização ou tomar a ação de acordo com o programado para a aplicação.

Os servos possuem um único objeto para a leitura do COB-ID deste objeto (objeto 1014h), que por padrão assume o valor 080h + Node-ID do escravo. A seguir é mostrado exemplos de telegramas enviados por este objeto para a sinalização de erros. Os códigos de erro utilizados são descritos na tabela 16.

Escravo no endereço 1 envia telegrama EMCY indicando erro de subtensão



Escravo no endereço 2 envia telegrama EMCY, indicando erro na guarda do nó

