



# Comunicação

Para sistemas de comando que  
são programados com SILworX



SAFETY  
NONSTOP



Todos os produtos HIMA mencionados neste manual estão protegidos pela marca registrada da HIMA. A não ser que seja mencionado de outra forma, isso também se aplica aos outros fabricantes e seus produtos mencionados.

Todos os dados e avisos técnicos neste manual foram elaborados com o máximo de cuidado, considerando medidas de controle de garantia de qualidade efetiva. Em caso de dúvidas, dirija-se diretamente à HIMA. A HIMA ficaria grata por quaisquer sugestões, p. ex., informações que ainda devem ser incluídas no manual.

Os dados técnicos estão sujeitos a alterações sem notificação prévia. A HIMA ainda se reserva o direito de modificar o material escrito sem aviso prévio.

Informações mais detalhadas encontram-se na documentação do DVD HIMA e na nossa homepage em <http://www.hima.com>.

© Copyright 2011, HIMA Paul Hildebrandt GmbH

Todos os direitos reservados.

## Contato

O endereço da HIMA é:

HIMA Paul Hildebrandt GmbH

Postfach 1261

D-68777 Brühl

Tel.: +49 6202 709-0

Fax: +49 6202 709-107

E-Mail: [info@hima.com](mailto:info@hima.com)

Índice de revisão	Alterações	Tipo de alteração	
		técnica	redacional
4.00	Adaptado ao SILworX V4 Edição em português (traduzida)		

## Índice

<b>1</b>	<b>Introdução .....</b>	<b>10</b>
1.1	Estrutura e utilização do manual.....	10
1.2	Grupo alvo .....	11
1.3	Convenções de representação.....	11
1.3.1	Avisos de segurança.....	11
1.3.2	Avisos de utilização .....	12
<b>2</b>	<b>Segurança .....</b>	<b>13</b>
2.1.1	Condições de utilização .....	13
2.2	Perigos residuais .....	16
2.3	Medidas de precaução de segurança .....	16
2.4	Informações para emergências .....	16
<b>3</b>	<b>Descrição do produto .....</b>	<b>17</b>
3.1	Protocolo direcionado à segurança (safeethernet) .....	17
3.2	Protocolos padrão .....	18
3.3	Redundância .....	20
3.4	Registrar e ativar os protocolos.....	21
3.5	Interfaces Ethernet .....	22
3.5.1	Características das interfaces Ethernet.....	22
3.5.2	Configuração das interfaces Ethernet.....	24
3.5.3	Portas de rede utilizadas para a comunicação Ethernet .....	29
3.6	Interfaces de barramento de campo .....	30
3.6.1	Instalação.....	30
3.6.2	Registrar e ativar.....	32
3.6.3	Pinagens das conexões D-Sub.....	32
<b>4</b>	<b>safeethernet .....</b>	<b>35</b>
4.1	O que é safeethernet .....	36
4.2	Configuração de uma conexão safeethernet redundante.....	39
4.2.1	Conectar variáveis de processo.....	40
4.2.2	Verificação da comunicação safeethernet.....	41
4.3	safeethernet-Editor .....	42
4.4	Visualização de detalhes do Editor safeethernet .....	44
4.4.1	Registro: Variáveis de sistema .....	44
4.4.2	Registro: definições de fragmento .....	47
4.5	Possíveis conexões safeethernet .....	47
4.5.1	Conexão safeethernet mono (canal 1).....	47
4.5.2	Conexão safeethernet redundante (canal 1 e canal 2).....	47
4.5.3	Combinações permitidas .....	47
4.6	Parâmetros safeethernet.....	49
4.6.1	Tempo de ciclo máximo (tempo de Watchdog mínimo) do sistema de comando HIMax .....	49
4.6.2	Tempo de ciclo máximo do sistema de comando HIMatrix .....	51
4.6.3	Receive Timeout .....	51
4.6.4	ResponseTime .....	51
4.6.5	Sync/Async .....	52

4.6.6	ResendTMO .....	52
4.6.7	Acknowledge Timeout .....	52
4.6.8	Production Rate .....	53
4.6.9	Queue .....	53
<b>4.7</b>	<b>Tempo máximo de reação para safeethernet .....</b>	<b>53</b>
4.7.1	Cálculo do tempo máximo de reação de dois sistemas de comando HIMax .....	55
4.7.2	Cálculo do tempo máximo de reação em conexão com um sistema de comando HIMatrix .....	55
4.7.3	Cálculo do tempo máximo de reação com dois sistemas de comando HIMatrix ou Remote I/Os .....	56
4.7.4	Cálculo do tempo máximo de reação com dois sistemas de comando HIMax e um sistema de comando HIMatrix .....	56
4.7.5	Cálculo do tempo máximo de reação de dois sistemas de comando HIMatrix .....	57
4.7.6	Cálculo do tempo máximo de reação com dois Remote I/Os .....	58
4.7.7	Cálculo do tempo máximo de reação, dois sistemas de comando HIMatrix, um sistema de comando HIMax .....	58
4.7.8	Perfis safeethernet .....	59
4.7.9	Perfil I (Fast & Cleanroom) .....	60
4.7.10	Perfil II (Fast & Noisy) .....	60
4.7.11	Perfil III (Medium & Cleanroom) .....	61
4.7.12	Perfil IV (Medium & Noisy) .....	61
4.7.13	Perfil V (Slow & Cleanroom) .....	62
4.7.14	Perfil VI (Slow & Noisy) .....	62
<b>4.8</b>	<b>Comunicação entre projetos .....</b>	<b>63</b>
4.8.1	Variantes para a comunicação entre projetos .....	64
<b>4.9</b>	<b>Comunicação entre projetos com SILworX e ELOP II Factory .....</b>	<b>66</b>
4.9.1	Configuração do HIMax no projeto SILworX .....	66
4.9.2	Configuração do HIMatrix no ELOP II Factory .....	70
<b>4.10</b>	<b>Control Panel (safeethernet) .....</b>	<b>72</b>
4.10.1	Campo de exibição (conexão safeethernet) .....	73
<b>5</b>	<b>PROFINET-IO .....</b>	<b>74</b>
<b>5.1</b>	<b>Blocos funcionais PROFINET-IO .....</b>	<b>74</b>
<b>5.2</b>	<b>Controlar o Consumer/Provider Status (IOxS) .....</b>	<b>75</b>
5.2.1	Variáveis de comando no controlador HIMA .....	75
5.2.2	Variáveis de comando no dispositivo DO HIMA .....	75
5.2.3	Variáveis de comando no dispositivo DI HIMA .....	75
<b>5.3</b>	<b>PROFIsafe .....</b>	<b>76</b>
5.3.1	Control-Byte e Status-Byte no PROFIsafe .....	77
5.3.2	F_WD_Time (tempo de Watchdog PROFIsafe) .....	77
5.3.3	SFRT (Safety Function Response Time) .....	79
<b>5.4</b>	<b>Restrições para a operação segura de PROFIsafe .....</b>	<b>81</b>
5.4.1	Endereçamento .....	81
5.4.2	Aspectos relacionados à rede .....	82
<b>5.5</b>	<b>HIMA PROFINET-IO Controller e PROFIsafe F-Host .....</b>	<b>83</b>
<b>5.6</b>	<b>Exemplo PROFINET-IO/PROFIsafe .....</b>	<b>84</b>
5.6.1	Criar o controlador HIMA PROFINET-IO no SILworX .....	84
<b>5.7</b>	<b>Funções de menu do controlador PROFINET-IO .....</b>	<b>89</b>
5.7.1	Exemplo para a árvore de estrutura do controlador PROFINET-IO .....	89
5.7.2	PROFINET-IO Controller .....	89

5.7.3	Dispositivo PROFINET-IO (no controlador) .....	91
5.7.4	DAP Module (Device Access Point Module) .....	91
5.7.5	Input/Output PROFINET-IO Module .....	93
5.7.6	Submódulo Input .....	94
5.7.7	Submódulo Output .....	99
5.7.8	Submódulo entradas e saídas .....	101
5.7.9	Application Relation (características) .....	103
5.7.10	Alarm CR (características) .....	104
5.7.11	Input CR (características) .....	105
<b>5.8</b>	<b>HIMA PROFINET-IO Device .....</b>	<b>108</b>
<b>5.9</b>	<b>Requisitos de sistema .....</b>	<b>108</b>
<b>5.10</b>	<b>Exemplo PROFINET-IO/PROFIsafe .....</b>	<b>109</b>
5.10.1	Configuração do dispositivo PROFINET-IO no SILworX .....	109
<b>5.11</b>	<b>Funções de menu do dispositivo PROFINET-IO .....</b>	<b>112</b>
5.11.1	Função de menu características .....	112
5.11.2	Módulos HIMA PROFINET-IO .....	113
5.11.3	Módulos HIMA PROFIsafe .....	114
5.11.4	Módulo PROFINet IO e PROFIsafe .....	115
<b>6</b>	<b>PROFIBUS DP .....</b>	<b>118</b>
<b>6.1</b>	<b>Master HIMA PROFIBUS DP .....</b>	<b>119</b>
6.1.1	Criar um master HIMA PROFIBUS DP .....	119
<b>6.2</b>	<b>Exemplo PROFIBUS DP .....</b>	<b>120</b>
6.2.1	Configurar o slave PROFIBUS DP .....	120
6.2.2	Configurar master PROFIBUS DP .....	122
<b>6.3</b>	<b>Funções de menu do master PROFIBUS DP .....</b>	<b>129</b>
6.3.1	Edit .....	129
6.3.2	Função de menu características .....	130
<b>6.4</b>	<b>Os métodos de acesso ao barramento do PROFIBUS-DP .....</b>	<b>134</b>
6.4.1	Protocolo Master/Slave .....	134
6.4.2	Protocolo Token .....	134
6.4.3	Tempo de rotação do token (Ttr) .....	134
6.4.4	Calcular o tempo de rotação do token Ttr .....	135
<b>6.5</b>	<b>Ciclo isocrônico PROFIBUS DP (a partir de DP-V2) .....</b>	<b>137</b>
6.5.1	Isochronous Mode (a partir de DP-V2) .....	138
6.5.2	Isochronous Sync Mode (a partir de DP-V2) .....	138
6.5.3	Isochronous Freeze Mode (a partir de DP-V2) .....	138
<b>6.6</b>	<b>Funções de menu do slave PROFIBUS DP (no master) .....</b>	<b>139</b>
6.6.1	Criar um slave PROFIBUS DP (no master) .....	139
6.6.2	Edit .....	139
6.6.3	Características .....	140
<b>6.7</b>	<b>Ler o arquivo GSD .....</b>	<b>145</b>
<b>6.8</b>	<b>Editar os parâmetros de usuários .....</b>	<b>146</b>
<b>6.9</b>	<b>Blocos funcionais PROFIBUS .....</b>	<b>148</b>
6.9.1	Bloco funcional MSTAT .....	149
6.9.2	Bloco funcional RALRM .....	152
6.9.3	Bloco funcional RDIAG .....	156
6.9.4	Bloco funcional RDREC .....	160
6.9.5	Bloco funcional SLACT .....	163
6.9.6	Bloco funcional WRREC .....	166

<b>6.10</b>	<b>Blocos funcionais auxiliares PROFIBUS .....</b>	<b>169</b>
6.10.1	Bloco funcional auxiliar ACTIVE .....	169
6.10.2	Bloco funcional auxiliar ALARM .....	170
6.10.3	Bloco funcional auxiliar DEID .....	171
6.10.4	Bloco funcional auxiliar ID .....	172
6.10.5	Bloco funcional auxiliar NSLOT .....	173
6.10.6	Bloco funcional auxiliar SLOT .....	173
6.10.7	Bloco funcional auxiliar STDDIAG .....	174
<b>6.11</b>	<b>Códigos de erro dos blocos funcionais .....</b>	<b>176</b>
<b>6.12</b>	<b>Control Panel (PROFIBUS DP Master) .....</b>	<b>177</b>
6.12.1	Menu de contexto (PROFIBUS Master) .....	177
6.12.2	Menu de contexto (Slave PROFIBUS) .....	177
6.12.3	Campo de exibição (PROFIBUS Master) .....	178
6.12.4	Estado do master PROFIBUS DP .....	179
6.12.5	Comportamento do master PROFIBUS DP .....	179
6.12.6	Função do LED FBx no master PROFIBUS DP .....	180
6.12.7	Função do LED FAULT no master PROFIBUS DP (Apenas HIMax) .....	180
<b>6.13</b>	<b>Slave HIMA PROFIBUS DP .....</b>	<b>181</b>
6.13.1	Criar um slave HIMA PROFIBUS DP .....	181
<b>6.14</b>	<b>Funções de menu slave PROFIBUS DP .....</b>	<b>182</b>
6.14.1	Edit .....	182
6.14.2	Características .....	184
<b>6.15</b>	<b>Control Panel (Slave PROFIBUS DP) .....</b>	<b>186</b>
6.15.1	Campo de exibição (Slave PROFIBUS DP) .....	186
<b>6.16</b>	<b>Função do LED FBx no slave PROFIBUS DP .....</b>	<b>187</b>
<b>6.17</b>	<b>Função do LED FAULT no slave PROFIBUS DP (Apenas HIMax) .....</b>	<b>187</b>
<b>7</b>	<b>Modbus .....</b>	<b>188</b>
<b>7.1</b>	<b>Topologia de barramento RS485 .....</b>	<b>188</b>
7.1.1	Pinagem H 7506 .....	189
7.1.2	Cabo de barramento .....	189
7.1.3	Características da transmissão RS485 .....	190
<b>7.2</b>	<b>Master HIMA Modbus .....</b>	<b>191</b>
7.2.1	Exemplo Modbus .....	192
7.2.2	Exemplo para o endereçamento alternativo de Register/Bit .....	196
7.2.3	Funções de menu do master Modbus HIMA .....	197
7.2.4	Códigos de função Modbus do master .....	200
7.2.5	Ethernet Slaves (TCP/UDP-Slaves) .....	207
7.2.6	Modbus Gateway (TCP/UDP Gateway) .....	210
7.2.7	Modbus serial .....	213
7.2.8	Control Panel (Modbus Master) .....	215
7.2.9	Control Panel (Modbus Master -> Slave) .....	216
7.2.10	Função do LED FBx no master Modbus .....	217
7.2.11	Função do LED FAULT no master Modbus (só HIMax) .....	217
<b>7.3</b>	<b>Slave HIMA Modbus .....</b>	<b>218</b>
7.3.1	Configuração do slave Modbus TCP .....	219
7.3.2	Configuração do slave Modbus TCP redundante .....	219
7.3.3	Regras para o slave Modbus TCP redundante .....	220
7.3.4	Funções de menu do Slave Set Modbus HIMA .....	221
7.3.5	Atribuir variáveis de envio/recepção .....	223
7.3.6	Variáveis de sistema Slave Set Modbus .....	223

7.3.7	Modbus Slave e Modbus Slave Redundant.....	223
7.3.8	Códigos de função Modbus do slave Modbus HIMA .....	226
7.3.9	Códigos de função específicos HIMA .....	228
7.3.10	Endereçamento Modbus via Bit e Register.....	230
7.3.11	Offsets para endereçamento Modbus alternativo .....	232
7.3.12	Control Panel (Modbus Slave).....	235
7.3.13	Função do LED FBx no slave Modbus .....	237
7.3.14	Função do LED FAULT no slave Modbus (só HIMax).....	237
7.3.15	Códigos de erro da conexão Modbus TCP/IP .....	237
<b>8</b>	<b>Send &amp; Receive TCP .....</b>	<b>238</b>
<b>8.1</b>	<b>Requisitos de sistema .....</b>	<b>238</b>
8.1.1	Criar um protocolo S&R TCP .....	238
<b>8.2</b>	<b>Exemplo: Configuração S&amp;R TCP .....</b>	<b>239</b>
8.2.1	Configuração S&R TCP do sistema de comando Siemens SIMATIC 300 .....	241
8.2.2	Configuração S&R TCP do sistema de comando HIMax .....	245
<b>8.3</b>	<b>Funções de menu protocolo S&amp;R TCP .....</b>	<b>247</b>
8.3.1	Edit.....	247
8.3.2	Características .....	247
<b>8.4</b>	<b>Funções de menu conexão TCP .....</b>	<b>249</b>
8.4.1	Edit.....	249
8.4.2	Variáveis de sistema .....	249
8.4.3	Características .....	250
<b>8.5</b>	<b>Troca de dados .....</b>	<b>252</b>
8.5.1	Conexões TCP .....	252
8.5.2	Troca de dados cíclica .....	253
8.5.3	Troca de dados acíclica com blocos funcionais.....	253
8.5.4	Troca de dados cíclica e acíclica simultaneamente.....	253
8.5.5	Controle de fluxo.....	254
<b>8.6</b>	<b>Sistemas de outros fabricantes com Pad Bytes .....</b>	<b>254</b>
<b>8.7</b>	<b>Blocos funcionais S&amp;R TCP .....</b>	<b>255</b>
8.7.1	TCP_Reset .....	256
8.7.2	TCP_Send .....	259
8.7.3	TCP_Receive .....	262
8.7.4	TCP_ReceiveLine .....	266
8.7.5	TCP_ReceiveVar .....	270
<b>8.8</b>	<b>Control Panel (Send/Receive over TCP) .....</b>	<b>275</b>
8.8.1	Campo de exibição parâmetros gerais .....	275
8.8.2	Campo de exibição conexões TCP.....	275
8.8.3	Código de erro da conexão TCP .....	276
8.8.4	Códigos de erro adicionais dos blocos funcionais .....	277
8.8.5	Estado da conexão .....	277
8.8.6	Estado da conexão parceiro .....	277
<b>9</b>	<b>SNTP Protocol .....</b>	<b>278</b>
<b>9.1</b>	<b>SNTP Client .....</b>	<b>278</b>
<b>9.2</b>	<b>SNTP Client (Server Info) .....</b>	<b>280</b>
<b>9.3</b>	<b>SNTP Server .....</b>	<b>280</b>
<b>10</b>	<b>X-OPC Server .....</b>	<b>282</b>
<b>10.1</b>	<b>Equipamentos necessários e requisitos de sistema.....</b>	<b>283</b>

<b>10.2</b>	<b>Properties X-OPC Server .....</b>	<b>284</b>
<b>10.3</b>	<b>Características do sistema de comando HIMA para a conexão X-OPC .....</b>	<b>285</b>
<b>10.4</b>	<b>Ações necessárias no caso de alterações.....</b>	<b>286</b>
<b>10.5</b>	<b>Forcing de variáveis globais em módulos de E/S .....</b>	<b>286</b>
<b>10.6</b>	<b>Configuração de uma conexão OPC Server .....</b>	<b>287</b>
10.6.1	Software necessário: .....	287
10.6.2	Requisitos para a operação do servidor X-OPC: .....	287
10.6.3	Instalação no Host PC.....	288
10.6.4	Configurar o servidor OPC no SILworX.....	291
10.6.5	Ajustes do OPC Server no Editor safeethernet .....	292
10.6.6	Configurar o servidor OPC Data Access no SILworX .....	293
10.6.7	Configurar o X-OPC Alarm & Event Server no SILworX .....	295
10.6.8	Parametrização de fragmentos e prioridades no SILworX .....	298
<b>10.7</b>	<b>Alarm &amp; Event Editor.....</b>	<b>301</b>
<b>10.7.1</b>	<b>Eventos booleanos .....</b>	<b>301</b>
<b>10.7.2</b>	<b>Eventos escalares .....</b>	<b>302</b>
<b>10.8</b>	<b>Parâmetros das características do servidor X-OPC .....</b>	<b>305</b>
10.8.1	OPC Server-Set .....	305
10.8.2	OPC Server .....	308
<b>10.9</b>	<b>Desinstalação do servidor X-OPC .....</b>	<b>308</b>
<b>11</b>	<b>Interface serial síncrona.....</b>	<b>309</b>
<b>11.1</b>	<b>Requisitos de sistema .....</b>	<b>309</b>
<b>11.2</b>	<b>Diagrama de blocos .....</b>	<b>309</b>
<b>11.3</b>	<b>Tomadas D-Sub FB1 e FB2.....</b>	<b>310</b>
<b>11.4</b>	<b>Configuração entre COM e submódulo SSI .....</b>	<b>310</b>
<b>11.5</b>	<b>Configuração da interface SSI .....</b>	<b>310</b>
11.5.1	Comprimento de condutores e frequências de relógio recomendadas .....	311
<b>11.6</b>	<b>Avisos de aplicação .....</b>	<b>312</b>
<b>12</b>	<b>ComUserTask .....</b>	<b>313</b>
<b>12.1</b>	<b>Requisitos de sistema .....</b>	<b>313</b>
12.1.1	Criar uma ComUserTask.....	313
<b>12.2</b>	<b>Requisitos .....</b>	<b>314</b>
<b>12.3</b>	<b>Abreviações .....</b>	<b>314</b>
<b>12.4</b>	<b>Interface CUT no SILworX .....</b>	<b>315</b>
12.4.1	Schedule Interval [ms].....	315
12.4.2	Pré-processamento do Scheduling .....	315
12.4.3	Pós-processamento do Scheduling.....	315
12.4.4	STOP_INVALID_CONFIG.....	316
12.4.5	Variáveis da interface CUT (CPU<->CUT).....	316
12.4.6	Função de menu características .....	317
12.4.7	Função de menu Edit .....	318
<b>12.5</b>	<b>Funções CUT .....</b>	<b>320</b>
12.5.1	Funções COM User Callback.....	320
12.5.2	Funções COM User Library.....	320
12.5.3	Header Files .....	320
12.5.4	Área de código e de dados e Stack para a CUT .....	321
12.5.5	Função inicial CUCB_TaskLoop.....	321



12.5.6	Interfaces seriais RS485/RS232 IF .....	322
12.5.7	UDP/TCP-Socket-IF .....	330
12.5.8	Timer IF .....	344
12.5.9	Diagnóstico .....	345
<b>12.6</b>	<b>Funções para SEW .....</b>	<b>346</b>
12.6.1	COM User IRQ Task.....	346
12.6.2	NVRam-IF .....	348
12.6.3	Semaphore-IF .....	349
12.6.4	COM-IO-IF (Só HM 30).....	352
<b>12.7</b>	<b>Instalação do ambiente de desenvolvimento.....</b>	<b>354</b>
12.7.1	Instalação do ambiente Cygwin .....	354
12.7.2	Instalação do compilador GNU .....	356
<b>12.8</b>	<b>Criar novo projeto CUT .....</b>	<b>358</b>
12.8.1	CUT Makefiles .....	359
12.8.2	Editar código fonte em C .....	361
12.8.3	Integrar ComUserTask ao projeto.....	365
12.8.4	Erro ao carregar uma configuração com CUT .....	368
<b>13</b>	<b>Informações gerais.....</b>	<b>369</b>
<b>13.1</b>	<b>Configuração dos blocos funcionais.....</b>	<b>369</b>
13.1.1	Aquisição das bibliotecas de blocos funcionais.....	369
13.1.2	Configuração dos blocos funcionais no programa de aplicação.....	369
13.1.3	Configuração dos blocos funcionais na árvore de estrutura do SILworX .....	370
<b>13.2</b>	<b>Fatia de tempo de comunicação máxima .....</b>	<b>372</b>
13.2.1	Para sistemas de comando HIMax .....	372
<b>13.3</b>	<b>BLimite de carga .....</b>	<b>372</b>
	<b>Anexo .....</b>	<b>373</b>
	<b>Glossário .....</b>	<b>373</b>
	<b>Lista de figuras .....</b>	<b>374</b>
	<b>Lista de tabelas .....</b>	<b>377</b>
	<b>Índice remissivo .....</b>	<b>386</b>

# 1 Introdução

O Manual de comunicação descreve as características e a configuração dos protocolos de comunicação dos sistemas de comando direcionados à segurança HIMax e HIMatrix com ajuda da ferramenta de programação SILworX.

Com os protocolos disponibilizados, os sistemas de comando HIMax/HIMatrix podem ser conectados entre si e com os sistemas de comando de outros fabricantes.

O conhecimento das normas e a implementação técnica impecável dos avisos contidos neste manual por parte do pessoal qualificado são pré-requisitos para o planejamento, o projeto, a programação, instalação, colocação em funcionamento, operação e manutenção dos sistemas de comando HIMax/HIMatrix.

A HIMA não assume nenhuma garantia por graves ferimentos, danos materiais e no meio ambiente que podem ser causados pelo seguinte: trabalhos realizados nos equipamentos por pessoal não qualificado, desligar ou contornar ("bypass") funções de segurança ou a inobservância dos avisos deste manual (e pelas avarias resultantes disso ou pela restrição das funções de segurança).

Equipamentos de automação HIMax são desenvolvidos, fabricados e testados de acordo com os padrões e regulamentos de segurança aplicáveis. Eles só podem ser utilizados para as aplicações previstas nas descrições e sob as condições ambientais especificadas.

## 1.1 Estrutura e utilização do manual

O conteúdo deste manual é parte da descrição do hardware do sistema eletrônico programável HIMax.

O manual é dividido nos seguintes capítulos principais:

- Introdução
- Segurança
- Descrição do produto
- Colocação em funcionamento
- Operação
- Manutenção preventiva
- Colocação fora de serviço
- Transporte
- Eliminação

Adicionalmente devem ser observados os seguintes documentos:

Nome	Conteúdo	Nº do documento
Manual de sistema HIMax	Descrição do Hardware do sistema HIMax	HI 801 242 P
Manual de segurança HIMax	Funções de segurança do sistema HIMax	HI 801 241 P
Manual de segurança HIMatrix	Funções de segurança do sistema HIMatrix	HI 800 526 P
Manual de elaboração do projeto HIMatrix	Considerações gerais sobre a elaboração do projeto de sistemas HIMatrix	HI 800 529 P
Manuais do hardware HIMatrix	Descrição do Hardware do respectivo sistema de comando HIMatrix	HI 800 xxx P
Primeiros passos	Introdução ao SILworX	HI 801 239 P

Tabela 1: Manuais adicionalmente em vigor

Os manuais atuais sempre podem ser encontrados na homepage da HIMA em [www.hima.com](http://www.hima.com). Com ajuda do índice de revisão na linha de rodapé, a atualidade de manuais eventualmente disponíveis pode ser comparada à versão na internet.

## 1.2 Grupo alvo

Este documento dirige-se a planejadores, projetistas e programadores de sistemas de automação, bem como pessoas autorizadas para colocação em funcionamento, operação e manutenção dos equipamentos e do sistema. Pressupõem-se conhecimentos especializados na área de sistemas de automatização direcionados à segurança.

## 1.3 Convenções de representação

Para a melhor legibilidade e para clarificação, neste documento valem as seguintes convenções:

<b>Negrito</b>	Ênfase de partes importantes do texto. Denominações de botões, itens de menu e registros no SILworX que podem ser clicados.
<i>Itálico</i>	Parâmetros de sistema e variáveis
Courier	Introdução de dados tal qual pelo usuário
RUN	Denominações de estados operacionais em letras maiúsculas
Cap. 1.2.3	Notas remissivas são hiperlinks, mesmo quando não são especialmente destacadas. Ao posicionar o cursor nelas, o mesmo muda sua aparência. Ao clicar, o documento salta para o respectivo ponto.

Avisos de segurança e utilização são destacados de forma especial.

### 1.3.1 Avisos de segurança

Os avisos de segurança no documento são representados como descrito a seguir. Para garantir o menor risco possível devem ser observados sem exceção. A estrutura lógica é

- Palavra sinalizadora: Perigo, Atenção, Cuidado, Nota
- Tipo e fonte do perigo
- Consequências do perigo
- Como evitar o perigo

#### PALAVRA SINALIZADORA



**Tipo e fonte do perigo!**

**Consequências do perigo**

**Como evitar o perigo**

O significado das palavras sinalizadoras é

- Perigo: No caso de não-observância resultam lesões corporais graves até a morte
- Atenção: No caso de não-observância há risco de lesões corporais graves até a morte
- Cuidado: No caso de não-observância há risco de lesões corporais leves
- Nota: No caso de não-observância há risco de danos materiais

**NOTA**

**Tipo e fonte dos danos!**  
**Como evitar os danos**

---

## 1.3.2

**Avisos de utilização**

Informações adicionais são estruturadas de acordo com o seguinte exemplo:

---

**i**

Neste ponto está o texto das informações adicionais.

---

Dicas úteis e macetes aparecem no formato:

---

**DICA**

Neste ponto está o texto da dica.

---

## 2 Segurança

É imprescindível ler informações de segurança, avisos e instruções neste documento. Apenas utilizar o produto observando todos os regulamentos e normas de segurança.

Este produto é operado com SELV ou PELV. Do módulo em si não emana nenhum perigo. Utilização na área Ex é permitida apenas com medidas adicionais.

### 2.1.1 Condições de utilização

Os equipamentos foram desenvolvidos para satisfazerem os requisitos das seguintes normas para CEM e requisitos climáticas e de meio-ambiente:

Norma	Conteúdo
IEC/EN 61131-2	Sistemas de controlador lógico programável, Parte 2 Requisitos e verificações de meios operacionais
IEC/EN 61000-6-2	CEM Norma técnica básica, Parte 6-2 Resistência a interferência, ambiente industrial
IEC/EN 61000-6-4	Compatibilidade eletromagnética (CEM) Norma técnica básica emissão de interferências, ambiente industrial

Tabela 2: Normas para requisitos de CEM, climáticas e do meio-ambiente

Para a utilização dos sistemas de comando direcionados à segurança HIMax devem ser respeitados os seguintes requisitos gerais:

Tipo de requisito	Conteúdo do requisito
Classe de proteção	Classe de proteção II conforme IEC/EN 61131-2
Contaminação	Grau de contaminação II conforme IEC/EN 61131-2
Altura de instalação	< 2000 m
Caixa	Padrão: IP 20/IP 00 Se as normas aplicáveis (p. ex., EN 60204) o exigirem, o equipamento deve ser montado numa caixa do grau de proteção exigido (p. ex., IP 54).

Tabela 3: Requisitos gerais

### Condições climáticas

Os mais importantes testes e valores limite para os requisitos climáticos são listados na tabela a seguir:

IEC/EN 61131-2	Testes climáticos
	Temperatura de operação: 0...+60 °C (Limites de teste: -10...+70 °C)
	Temperatura de armazenamento: -40...+85 °C
	Calor e frio secos; testes de resistência: +70 °C/-25 °C, 96 h, alimentação de corrente não ligada
	Mudança de temperatura; teste de resistência e insensibilidade: -25 °C/+70 °C e 0 °C/+55 °C, alimentação de corrente não ligada
	Ciclos com calor úmido; testes de resistência: +25 °C/+55 °C, 95% umidade relativa, alimentação de corrente não ligada

Tabela 4: Requisitos climáticos

### Requisitos mecânicos

Os mais importantes testes e valores limite para os requisitos mecânicos são listados na tabela a seguir:

IEC/EN 61131-2	Testes mecânicos
	Teste de insensibilidade a oscilações: 5...9 Hz/3,5 mm amplitude 9...150 Hz, 1 g, objeto de teste em operação, 10 ciclos por eixo
	Teste de insensibilidade a choques: 15 g, 11 ms, objeto de teste em operação, 3 choques por eixo e direção (18 choques)

Tabela 5: Testes mecânicos

### Requisitos CEM

Para sistemas direcionados à segurança são exigidos níveis mais elevados na resistência contra interferências. Os sistemas HIMax satisfazem estes requisitos conforme IEC 62061 e IEC 61326-3-1. Veja a coluna “Critério SF” (Segurança funcional).

Normas de teste	Testes de resistência contra interferência	Critério SF
IEC/EN 61000-4-2	Teste ESD: 6 kV contato-, 8 kV descarga pelo ar	6 kV, 8 kV
IEC/EN 61000-4-3	Teste de RFI (10 V/m): 80 MHz...2 GHz, 80% AM Teste de RFI (3 V/m): 2 GHz...3 GHz, 80% AM Teste de RFI (20 V/m): 80 MHz...1 GHz, 80% AM	- - 20 V/m
IEC/EN 61000-4-4	Teste Burst: Tensão de alimentação: 2 kV e 4 kV Condutores de sinal: 2 kV	4 kV 2 kV
IEC/EN 61000-4-12	Teste com oscilações atenuadas: 2,5 kV L-, L+/PE 1 kV L+/L -	- -
IEC/EN 61000-4-6	Alta frequência, assimétrica: 10 V, 150 kHz...80 MHz, 80% AM 20 V, frequências ISM, 80% AM	10 V -
IEC/EN 61000-4-3	Pulsos de 900 MHz	-
IEC/EN 61000-4-5	Tensão de choque: Tensão de alimentação: 2 kV CM, 1 kV DM Condutores de sinal: 2 kV CM, 1 kV DM com E/S AC	2 kV/1 kV 2 kV

Tabela 6: Testes de resistência contra interferência

IEC/EN 61000-6-4	Testes de emissão de interferência
EN 55011 Classe A	Emissão de interferências: por irradiação, via conexão de cabo

Tabela 7: Testes de emissão de interferência

### Alimentação com tensão

Os mais importantes testes e valores limite para a alimentação com tensão dos equipamentos são listados na tabela a seguir:

IEC/EN 61131-2	Verificação das características da alimentação com corrente contínua
	Alternativamente, a alimentação com tensão deve satisfazer as seguintes normas: IEC/EN 61131-2 ou SELV (Safety Extra Low Voltage) ou PELV (Protective Extra Low Voltage)
	A proteção dos equipamentos HIMax deve ocorrer de acordo com as indicações deste manual
	Verificação da faixa de tensão: 24 VDC, -20%...+25% (19,2 V...30,0 V)
	Teste de insensibilidade a interrupções por breve tempo da alimentação com corrente externa: DC, PS 2: 10 ms
	Inversão da polaridade da tensão de alimentação: Nota no respectivo capítulo do manual de sistema ou na folha de dados da alimentação com corrente.
	Duração do amortecedor, teste de resistência: Teste B, 1000 h

Tabela 8: Verificação das características da alimentação com corrente contínua

### Medidas de proteção contra ESD

Apenas pessoal com conhecimentos sobre medidas de proteção contra ESD pode efetuar alterações ou ampliações do sistema ou a substituição de um módulo.

#### NOTA



**Descargas eletrostáticas podem danificar componentes eletrônicos montados nos sistemas de comando!**

- Usar para os trabalhos um posto de trabalho protegido contra descarga eletrostática e usar uma fita de aterramento.
- Guardar módulos protegidos contra descarga eletrostática, p. ex., na embalagem.

**Alterações ou ampliações na fiação do sistema apenas podem ser efetuadas por pessoal que tiver conhecimento de medidas de proteção contra ESD.**

## 2.2 Perigos residuais

Do módulo HIMax em si não emana nenhum perigo.

Perigos residuais podem ser causados por:

- Erros do projeto
- Erros no programa de aplicação
- Erros na fiação

## 2.3 Medidas de precaução de segurança

Observar as normas de segurança em vigor no local de utilização e usar o equipamento de proteção prescrito.

## 2.4 Informações para emergências

Um sistema de comando HIMax é parte da tecnologia de segurança de uma instalação. A falha do sistema de comando coloca a instalação no estado seguro.

Em casos de emergência é proibida qualquer intervenção que impeça a função de segurança dos sistemas HIMax.



### 3 Descrição do produto

Os sistemas HIMax/HIMatrix podem trocar dados de processo mediante uma conexão de dados entre si e com sistemas de outros fabricantes. A configuração dos protocolos descritos neste manual é efetuada na ferramenta de programação SILworX.

Para a programação dos sistemas HIMatrix no SILworX, os sistemas HIMatrix devem estar equipados com os sistemas operacionais a partir de CPU OS V7 e COM OS V12. Uma conexão **safeethernet** aos sistema sHIMatrix com sistema operacional mais antigo pode ocorrer mediante a comunicação no nível superior ao do projeto, veja Capítulo 4.8.

No contexto da melhora de produtos, a HIMA continua desenvolvendo os sistemas operacionais. Estas versões melhoradas podem ser carregadas aos módulos com ajuda do SILworX, veja manuais de sistema HIMax HI 801 240 P, HIMatrix HI 800 528 P e HI 800 527 P.

Nas seguintes duas seções (3.1 e 3.2) damos uma visão geral sobre o protocolo direcionado à segurança **safeethernet** e os protocolos padrão disponíveis.

#### 3.1 Protocolo direcionado à segurança (safeethernet)

Todos os sistemas HIMax/HIMatrix podem comunicar-se via Ethernet de forma direcionada à segurança de acordo com SIL 3. A comunicação direcionada à segurança ocorre através do protocolo **safeethernet**.

O protocolo direcionado à segurança **safeethernet** será usado para a troca de dados de processo direcionada à segurança entre os sistemas de comando HIMax e HIMatrix numa rede Ethernet.

Descrição do protocolo direcionado à segurança **safeethernet** a partir do Capítulo 4.

#### **⚠ ATENÇÃO**



**Manipulação da transmissão de dados direcionada à segurança!**

**Danos pessoais**

A empresa operadora deve garantir que a rede Ethernet utilizada para **safeethernet** seja protegida de forma suficiente contra manipulações (p. ex., por hackers). O tipo e a abrangência das medidas devem ser autorizados pela respectiva instituição de certificação.

### 3.2 Protocolos padrão

Para uma ótima integração do sistema HIMax/HIMatrix com equipamentos de campo e sistemas de gestão, há adicionalmente protocolos padrão comprovados à disposição. Usam-se para este fim tanto Ethernet quanto protocolos de barramento de campo.

Uma série de protocolos de comunicação permite apenas uma transmissão de dados não direcionada à segurança. Estes protocolos podem ser utilizados para aspectos não direcionados à segurança de uma tarefa de automação.

Elemento	HIMax	HIMatrix L3	HIMatrix L2	Descrição
Módulo/sistema de comando necessário	No máximo 20 módulos de comunicação por sistema de comando HIMax	Módulo de comunicação integrado do sistema de comando	Módulo de comunicação integrado do sistema de comando	Protocolos padrão são executados no módulo de comunicação
Interfaces	Interfaces Ethernet e interfaces de barramento de campo dos módulos COM.	Interfaces Ethernet e interfaces de barramento de campo do sistema de comando.	Interfaces Ethernet e interfaces de barramento de campo do sistema de comando.	Para informações mais detalhadas, veja Tabela 14.
Quantidade máx. de protocolos padrão	<ul style="list-style-type: none"> <li>20 por sistema de comando HIMax.</li> <li>6<sup>1)</sup> por módulo COM</li> </ul>	6 <sup>1)</sup>	4 <sup>1)</sup>	Protocolos padrão disponíveis, veja Tabela 10.
Volume de dados de processo de todos os protocolos padrão de um sistema de comando	128 kB enviar 128 kB receber	64 kB enviar 64 kB receber	16 kB enviar 16 kB receber	O volume máximo de dados de processo do sistema de comando não pode ser ultrapassado. Neste caso, a parametrização do sistema de comando é recusada ao carregar.

<sup>1)</sup> X-OPC Server, SNTP Client e SNTP Server não entram neste cálculo

Tabela 9: Protocolos padrão

#### **⚠ ATENÇÃO**



**Utilização de dados importados não seguros.**

**Dados inseguros não podem ser utilizados para as funções de segurança do programa de aplicação.**

Os seguintes protocolos padrão estão disponíveis:

Protocolo	por módulo HIMax	por HIMatrix	Descrição
PROFINET IO Controller	1	---	Capítulo 5.5
PROFINET IO Device	1	---	Capítulo 5.8
PROFIBUS DP Master	2	1 para F20, 2 para F30, F35, F60	Capítulo 6.1
PROFIBUS DP Slave	1	1	Capítulo 6.13
Modbus Master	1	1	Capítulo 7.2
Modbus Slave	1	1	Capítulo 7.3
S&R TCP	1	1	Capítulo 8
HIMA X-OPC Server <sup>1)</sup>	---	---	Capítulo 10
ComUserTask	1	1	Capítulo 12
<sup>1)</sup> O HIMA X-OPC Server é instalado num PC host e serve como interface de transmissão entre até 255 sistemas de comando HIMax/HIMatrix e sistemas de outros fabricantes que disponham de uma interface OPC.			

Tabela 10: Protocolos padrão disponíveis

## i

No máximo 1280 conexões TCP por sistema de comando HIMax com 20 módulos COM.  
No máximo 64 conexões TCP por sistema de comando HIMatrix L3 ou módulo COM HIMax.

No máximo 32 conexões TCP por sistema de comando HIMatrix L2.

### A quantidade máxima de protocolos ativos num sistema HIMatrix ou módulo COM HIMax

No máximo 64 soquetes TCP estão disponíveis por HIMatrix L3 ou módulo COM HIMax.

Exemplo 1:

Protocolo	Conexões
1 Modbus Master	TCP 44 conexões slave
1 Modbus Slave	TCP 20 conexões master

Tabela 11: Protocolos num módulo de comunicação

Exemplo 2:

Protocolo	Conexões
1 PROFIBUS DP Master	122 conexões slave
1 PROFIBUS DP Slave	1 conexão master

Tabela 12: Protocolos num módulo de comunicação

### 3.3 Redundância

O sistema HIMax foi concebido como sistema altamente disponível e também disponibiliza redundância para a comunicação. Uma conexão de comunicação é redundante se dois caminhos de transmissão físicos existem e 2 módulos do sistema HIMax são utilizados para este fim.

Para HIMatrix, uma comunicação redundante apenas está prevista via **safeethernet** e com uma interface Ethernet.

#### Redundância via safeethernet

A redundância é configurada no Editor **safeethernet** selecionando as interfaces Ethernet para o segundo caminho de transporte (veja Capítulo 4.2).

#### Redundância via protocolos padrão

PROFIBUS DP Master PROFIBUS DP Slave PROFINET IO TCP S&R Modbus Master	A redundância do respectivo protocolo padrão deve ser configurada no programa de aplicação de maneira que este monitore os caminhos de transporte redundantes e associe os dados de processo transmitidos de forma redundante ao respectivo caminho de transporte.
Modbus Slave	A redundância pode ser ajustada no SILworX para o HIMax.

### 3.4 Registrar e ativar os protocolos

Os seguintes protocolos estão disponíveis para os sistemas HIMax/HIMatrix e podem ser ativados como segue:

**i**

Num X-COM é necessário para cada protocolo um código separado de liberação do software. Exceção: Basta um código de liberação do software para operar 2 Modbus Slaves num X-COM.

Protocolo	Interfaces	Ativação
HIMA safe <b>ethernet</b>	Ethernet	[1]
SNTP Server/Client	Ethernet	[1]
HIMA X-OPC Server (roda no Host PC)	Ethernet	[4]
PROFINET IO Controller <sup>1)</sup>	Ethernet	[4]
PROFINET IO Device <sup>1)</sup>	Ethernet	[4]
Modbus TCP Master	Ethernet	[4]
Modbus TCP Slave	Ethernet	[4]
TCP Send/Receive	Ethernet	[4]
PROFIBUS DP Master	FB1 e FB2	[2]
PROFIBUS DP Slave	FB1 ou FB2	[2]
Modbus Master RS485	FB1 ou FB2	[3]
Modbus Slave RS485	FB1 ou FB2	[3]
ComUserTask RS232, RS485, RS422, SSI	FB1 ou FB2	[3]
<sup>1)</sup> Não disponível para HIMatrix L2		

Tabela 13: Protocolos disponíveis do sistema HIMax/HIMatrix

[1]. A função está liberada por padrão em todos os sistemas HIMax/HIMatrix.

[2]. Para PROFIBUS Master e PROFIBUS Slave, a liberação ocorre mediante instalação de uma submódulo de barramento de campo.

[3]. Para submódulo de barramento de campo RS485 (Modbus RS485) e submódulo de barramento de campo RS232, RS422, SSI (ComUserTask), é necessário adquirir adicionalmente um código de liberação do software para o protocolo de barramento de campo selecionado.

[4]. O código de liberação do software pode ser gerado na homepage com ajuda do número ID de sistema (p. ex., 60000) do seu sistema de comando; seguir as instruções na homepage da HIMA [www.hima.com](http://www.hima.com) -> *Products* -> *Registration* -> *Communication Options SILworX*

**i**

O código de liberação do software está associado de forma inseparável a este número ID de sistema. Uma licença só pode ser usada uma vez para um determinado ID de sistema. Por isso, a liberação apenas deve ser efetuada depois de ter a ID de sistema definitiva.

Todos os protocolos Ethernet podem ser testados por 5000 horas de operação, mesmo sem código de liberação do software.

**i**

Encomendar o código de liberação do software com antecedência!

Depois de esgotar as 5000 horas de operação, a comunicação continua até parar o sistema de comando. Depois disso, o programa de aplicação não pode ser mais iniciado sem código válido para a liberação do software para os protocolos projetados (configuração inválida).

**Assim introduz-se o código de liberação do software no SILworX:**

1. Selecionar na árvore de estrutura **Configuration, Resource, License Management**.
2. Clicar com o botão direito em **License Management** e selecionar no menu de contexto **New, License Key**.  
☒ A chave de licença é adicionada.
3. Clicar com o botão direito em **License Key** e selecionar no menu de contexto **Properties**.
4. Introduzir no campo **Activation Code** o código de liberação do software gerado.

**3.5 Interfaces Ethernet**

A comunicação com sistemas externos ocorre mediante a interface Ethernet dos módulos CPU e COM do HIMax e dos sistemas de comando HIMatrix. Cada interface Ethernet pode processar simultaneamente vários protocolos.

**i**

A comunicação de dados de processo não é possível pelas interfaces Ethernet do grupo de barramento de sistema HIMax X-SB 01. As interfaces Ethernet UP e DOWN são previstas exclusivamente para a conexão dos suportes básicos HIMax entre si. Se o barramento de sistema for operado em estrutura de rede, também a interface Ethernet DIAG pode ser usada para conexão dos suportes básicos HIMax. Veja Manual de sistema HI 801 242 P.

**3.5.1 Características das interfaces Ethernet**

Característica	Módulo CPU HIMax	Módulo COM HIMax	HIMatrix
Portas	4	4	4, 2 com F20 e Remote I/Os
Padrão de transmissão	10/100/1000 Base-T, Semi- e Fullduplex	10/100 Base-T, Semi- e Fullduplex	10/100 Base-T 1000 Base-T (só L3) Semi- e Fullduplex
Auto Negotiation	Sim	Sim	Sim
Auto-Crossover	Sim	Sim	Sim
Tomada de conexão	RJ 45	RJ 45	RJ 45
Endereço IP	Livremente configurável <sup>1)</sup>	Livremente configurável <sup>1)</sup>	Livremente configurável <sup>1)</sup>
Subnet Mask	Livremente configurável <sup>1)</sup>	Livremente configurável <sup>1)</sup>	Livremente configurável <sup>1)</sup>
Protocolos suportados	<b>safeethernet</b> Aparelho de programação (PADT), SNTP	<b>safeethernet</b> Protocolos padrão	<b>safeethernet</b> Protocolos padrão Aparelho de programação (PADT), SNTP
<sup>1)</sup> Regras geralmente válidas para a atribuição de endereços IP e máscara de subrede devem ser observadas.			

Tabela 14: Características interfaces Ethernet

Cada módulo CPU e módulo COM HIMax e cada sistema de comando HIMatrix possui um Ethernet Switch com um endereço IP livremente configurável.

O Ethernet Switch estabelece uma conexão direcionada entre dois parceiros de comunicação para a transmissão de dados. Isso impede colisões e alivia a carga da rede.

Para encaminhamento direcionado de dados, uma tabela de atribuição de endereços MAC-/IP (cache ARP) é criada e endereços MAC são atribuídos a determinando endereços IP. Agora, pacotes de dados apenas são encaminhados aos endereços IP listados no cache ARP.

---

**i**

Substituição de um módulo CPU/COM do HIMax ou de um sistema de comando HIMatrix com o mesmo endereço IP.

Se um dispositivo é substituído para o qual foi ajustado um *ARP Aging Time* = 5 Minutes e um valor *MAC-Learning* = Conservative, então, o parceiro de comunicação passa a usar este novo endereço MAC somente após no mínimo 5 e no máximo 10 minutos. Durante este tempo, não existe possibilidade de comunicação através do dispositivo substituído.

Além do ARP Aging Time ajustável, deve ser aguardado no mínimo o MAC Aging Time não alterável do Switch (aprox. 10 segundos) para uma comunicação através do dispositivo substituído ser novamente possível.

---

### 3.5.2 Configuração das interfaces Ethernet

A configuração das interfaces Ethernet ocorre no SILworX pela visualização de detalhe do módulo CPU ou COM.

Para sistemas de comando HIMax/HIMatrix, os valores padrão dos parâmetros *Speed Mode* e *Flow Control Mode* são ajustados para AutoNeg.

---

**i**

Perda da comunicação!

No caso de um ajuste inadequado dos parâmetros Ethernet, o dispositivo não pode ser acessado mais. Efetuar um reset do dispositivo!

---

**Assim pode ser aberta a visualização de detalhes do módulo de comunicação:**

1. Selecionar na árvore de estrutura **Configuration, Resource, Hardware**.
2. Clicar com o botão direito em **Hardware** e selecionar no menu de contexto **Edit** para abrir o Hardware Editor.
3. Clicar com o botão direito em **Communication Module** e selecionar no menu de contexto **Detail View** para abrir a visualização de detalhes.

---

**i**

Os valores nas características dos módulos COM e CPU devem ser novamente compilados com o programa de aplicação e transferidos ao sistema de comando para que se tornem efetivos para a comunicação do sistema HIMax/HIMatrix.

---



## Módulos

Elemento	Descrição
Name	Nome do módulo de comunicação.
Use Max CPU Load for HH Protocol	<ul style="list-style-type: none"> <li>Ativado: Transferir o limite da carga da CPU do campo <i>Max. CPU Load [%]</i>.</li> <li>Desativado: Não usar limite da carga da CPU para <b>safeethernet</b>.</li> </ul>
Max. CPU Load for HH Protocol [%]	<p>Carga máxima da CPU do módulo que pode ser produzida ao processar o protocolo <b>safeethernet</b>.</p> <hr/> <p><b>i</b> A carga máxima deve ser dividida entre todos os protocolos que usam este módulo de comunicação.</p>
IP Address	Endereço IP da interface Ethernet
Subnet Mask	Máscara de endereço 32 Bit para subdividir um endereço de IP em endereço de rede e host.
Speed Mode	<hr/> <p><b>i</b> Apenas ajuste Autoneg permitido! Com outros ajustes, o módulo entra em STOP.</p> <hr/>
Flow Control Mode	
Standard interface	Ativado: A interface é usada como interface padrão para o login de sistema.
Default Gateway	Endereço IP do Default Gateway
Activate Extended Settings	Utilizar os parâmetros <i>ARP Aging Time [s]</i> , <i>MAC Learning</i> e <i>IP Forwarding</i> .
ARP Aging Time [s]	<p>Um módulo CPU ou COM grava os endereços MAC de seus parceiros de comunicação em uma tabela de correspondência (ARP Cache).</p> <p>Se durante um período de 1 a 2 vezes o <i>ARP Aging Time</i></p> <ul style="list-style-type: none"> <li>chegarem mensagens dos parceiros de comunicação, o endereço MAC é mantido no cache ARP.</li> <li>não chegarem mensagens dos parceiros de comunicação, o endereço MAC é excluído do cache ARP.</li> </ul> <p>O valor típico para o <i>ARP Aging Time</i> em uma rede local é de 5 s...300 s.</p> <p>O conteúdo do cache ARP não pode ser lido pelo usuário.</p> <p>Ao utilizar roteadores ou gateways, adaptar (aumentar) o <i>ARP Aging Time</i> ao retardo adicional para o caminho de ida e volta. Com o <i>ARP Aging Time</i> insuficiente, o módulo CPU/COM exclui o endereço MAC do parceiro de comunicação do cache ARP e a comunicação é efetuada apenas com atraso ou é interrompida. Para a utilização eficaz, o ARP Aging Time deve ser &gt; Receive Timeouts dos protocolos usados.</p> <p>Faixa de valores: 1 s...3600 s Valor padrão: 60 s</p>

Elemento	Descrição
MAC Learning	<p>Com MAC Learning e <i>ARP Aging Time</i>, o usuário ajusta quão rápido um endereço MAC deve ser apreendido.</p> <p>Os seguintes ajustes são possíveis:</p> <ul style="list-style-type: none"> <li>▪ <b>Conservative (Recomendado):</b> Se no cache ARP já se encontram endereços MAC de parceiros de comunicação, estas entradas estão travadas pelo período de no mínimo 1 vez o <i>ARP Aging Time</i> até o máximo de 2 vezes o <i>ARP Aging Time</i> e não podem ser substituídos por outros endereços MAC. Desta forma é garantido que os pacotes de dados são possam ser desviados intencionalmente ou acidentalmente para participantes estranhos da rede (ARP spoofing).</li> <li>▪ <b>Tolerant:</b> Ao receber uma mensagem, o endereço IP na mensagem é comparado com os dados no cache ARP e o endereço MAC armazenado no cache ARP é imediatamente sobrescrito com o endereço MAC da mensagem. Utilize o ajuste "Tolerant" quando a disponibilidade da comunicação for mais importante que o acesso seguro (authorized access) ao sistema de comando.</li> </ul>
IP Forwarding	<p>Permite a um módulo CPU ou COM trabalhar como roteador e encaminhar pacotes de dados de outros nós da rede.</p> <ul style="list-style-type: none"> <li>▪ <b>Ativado:</b> Encaminhamento está ligado.</li> <li>▪ <b>Desativado:</b> Encaminhamento está desligado.</li> </ul>
ICMP Mode	<p>O Internet Control Message Protocol (ICMP) permite às camadas mais altas do protocolo detectar estados de falha no nível de intermediação e, assim, permite otimizar a transmissão dos pacotes de dados.</p> <p>Tipos de mensagens do Internet Control Message Protocol (ICMP) que são apoiados pelo módulo CPU:</p> <ul style="list-style-type: none"> <li>▪ <b>No ICMP Responses</b> Todos os comandos ICMP estão desligados. Assim, é alcançada uma alta segurança contra sabotagem que poderia ocorrer pela rede.</li> <li>▪ <b>Echo Response</b> Quando "Echo Response" estiver ligado, o nó responde a um comando de Ping. Assim, pode ser determinado se um nó pode ser alcançado. A segurança ainda continua elevada.</li> <li>▪ <b>Host Unreachable</b> Sem importância para o usuário. Apenas para testes no fabricante.</li> <li>▪ <b>All Implemented ICMP Responses</b> Todas as respostas ICMP implementadas estão ligadas. Assim, é obtido um diagnóstico de falhas mais detalhado no caso de avarias na rede.</li> </ul>

Tabela 15: Parâmetros de configuração

## Routings

Elemento	Descrição
Name	Denominação do ajuste de Routing
IP Address	Endereço IP de destino do parceiro de comunicação (no caso de Host-Routing direto) ou endereço de rede (no caso de Subnet-Routing) Faixa de valores: 0.0.0.0...255.255.255.255 Valor padrão: 0.0.0.0
Subnet Mask	Define a faixa de endereços de destino para uma entrada de Routing (roteamento). 255.255.255.255 (para Host-Routing direto) ou Subnet Mask da subrede endereçada. Faixa de valores: 0.0.0.0...255.255.255.255 Valor padrão: 255.255.255.255
Gateway	Endereço IP do gateway para a rede endereçada. Faixa de valores: 0.0.0.0...255.255.255.255 Valor padrão: 0.0.0.1

Tabela 16: Parâmetros de roteamento

## Switch Ethernet

Elemento	Descrição
Port	Número da porta e identificação na caixa; para cada porta apenas pode haver uma configuração. Faixa de valores: 1...4
Speed [Mbit/s]	10 MBit/s: Taxa de dados 10 MBit/s 100 MBit/s: Taxa de dados 100 MBit/s 1000 MBit/s: Taxa de dados 1000 MBit/s (módulo CPU) Autoneg (10/100/1000): ajuste automático da Baudrate Valor padrão: Autoneg
Flow Control	Full Duplex: Comunicação simultânea em ambas as direções Semi-duplex: Comunicação em uma das direções Autoneg: Controle automático da comunicação Valor padrão: Autoneg
Autoneg also with Fixed Values	O "Advertising" (transmissão das características de <i>Speed</i> e <i>Flow Control</i> ) também é efetuado no caso de valores fixos ajustados para <i>Speed</i> e <i>Flow Control</i> . Assim, outros dispositivos cujas portas estão ajustadas para <i>Autoneg</i> reconhecem o ajuste das portas HiMax.
Limit	Limitar pacotes de entrada Multicast e/ou Broadcast. Desliga: sem limitação Broadcast: limitar Broadcast (128 kbit/s) Multicast e Broadcast: limitar Multicast e Broadcast (1024 kbit/s) Valor padrão: Broadcast

Tabela 17: Parâmetros do switch Ethernet

### VLAN (Port based VLAN)

(a função não é apoiada pelos sistemas de comando HIMatrix)

Configura a utilização de port-based VLAN.

**i**

Se VLAN deve ser apoiado, "Port based VLAN" deve estar desligado, para que cada porta possa comunicar-se com qualquer outra porta do Switch.

É possível ajustar para cada porta de um switch para qual outra porta do switch podem ser enviados os frames Ethernet recebidos.

A tabela no registro VLAN contém entradas pelas quais a conexão entre duas portas pode ser comutada para ativa ou inativa.

	Eth1	Eth2	Eth3	Eth4
Eth1				
Eth2	ativa			
Eth3	ativa	ativa		
Eth4	ativa	ativa	ativa	
COM	ativa	ativa	ativa	ativa

Tabela 18: Registro VLAN

Ajuste padrão: todas as conexões entre as portas estão ativas

### LLDP

(a função não é apoiada pelos sistemas de comando HIMatrix)

LLDP (Link Layer Discovery Protocol) transmite em intervalos periódicos via Multicast informações sobre o próprio dispositivo (p. ex., endereço MAC, nome do dispositivo, número da porta) e recebe as mesmas informações de dispositivos vizinhos.

Dependendo do fato de PROFINET estar configurado no módulo de comunicação, os seguintes valores de LLDP são usados:

PROFINET no módulo COM	ChassisID	TTL (Time to Live)
usado	Nome da estação	20 s
não usado	Endereço MAC	120 s

Tabela 19: Valores para LLDP

Os módulos de processador e comunicação apóiam LLDP nas portas Eth1, Eth2, Eth3 e Eth4.

Os seguintes parâmetros definem como a respectiva porta trabalha:

Off	LLDP desativado nesta porta
Send	LLDP envia frames Ethernet LLDP, frames Ethernet recebidos são excluídos sem processar os mesmos
Receive	LLDP não envia frames Ethernet LLDP, mas frames Ethernet recebidos são processados
Send/Receive	LLDP envia e processa frames Ethernet LLDP recebidos

Ajuste padrão: Send/Receive

### Mirroring

(a função não é apoiada pelos sistemas de comando HIMatrix)

Configura se o módulo Ethernet duplica pacotes em uma porta, assim que eles possam ser lidos também por um dispositivo ligado no mesmo, p. ex., para fins de testes.

Os seguintes parâmetros definem como a respectiva porta trabalha:

Off	Esta porta não participa do Mirroring (espelhamento).
Egress:	Dados de saída desta porta são duplicados.
Ingress:	Dados de entrada desta porta são duplicados.
Ingress/Egress:	Dados de entrada e saída desta porta são duplicados.
Dest Port:	Os dados duplicados são enviados para esta porta.

Ajuste padrão: Off

### 3.5.3 Portas de rede utilizadas para a comunicação Ethernet

#### Portas UDP/Utilização

123	SNTP (sincronização de tempo entre PES e Remote I/O, bem como dispositivos externos)
502	Modbus Slave (pode ser alterado pelo usuário)
6010	safeethernet e OPC
8001	Configuração das Remote I/O pelo PES
8000	Programação e operação com SILworX
34964	PROFINET Endpointmapper (necessário para estabelecer a conexão)
49152	PROFINET RPC-Server
49153	PROFINET RPC-Client

#### Portas TCP/Utilização

502	Modbus Slave (pode ser alterado pelo usuário)
Xxx	TCP-SR atribuído pelo usuário

### i

Todas as portas acima listadas são Destination Ports. As Source Ports dos componentes de comunicação são “variable” e não podem ser influenciadas.

A tarefa ComUserTask pode usar qualquer porta se a mesma ainda não está ocupada por um outro protocolo.

### 3.6 Interfaces de barramento de campo

Os submódulos do barramento de campo permitem a comunicação pelas interfaces de barramento de campo do HIMax X-COM 01, bem como dos sistemas de comando HIMatrix F20, F30, F35 e F60 CPU 01. Os submódulos do barramento de campo são uma opção e são instalados em fábrica.

A interface de barramento de campo FB3 dos sistemas de comando HIMatrix está atribuída por padrão de fábrica com RS485 para Modbus (Master ou Slave) ou ComUserTask.

Comunicação direcionada à segurança não é possível através das interfaces de barramento de campo.

A seguinte visão geral mostra os submódulos de barramento de campo disponíveis:

Denominação	Descrição
PROFIBUS Master	Submódulo de barramento de campo PROFIBUS DP Master
PROFIBUS Slave	Submódulo de barramento de campo PROFIBUS DP Slave
Módulo RS485	Submódulo de barramento de campo RS485 para a utilização com Modbus (Master ou Slave) ou ComUserTask
Módulo RS232	Submódulo de barramento de campo RS232 para a utilização com ComUserTask
Módulo RS422	Submódulo de barramento de campo RS422 para a utilização com ComUserTask
INTERBUS Master <sup>1)</sup>	Barramento de campo INTERBUS Master
Módulo SSI	Submódulo de barramento de campo SSI para a utilização com ComUserTask
<sup>1)</sup> Apenas para HIMatrix com sistema operacional anterior a CPU OS V7 e COM OS V12 Apenas suportado na ferramenta de programação ELOP II Factory.	

Tabela 20: Submódulos de barramento de campo disponíveis

#### 3.6.1 Instalação

Os sub-módulos do barramento de campo são uma opção e são instalados em fábrica. A definição ocorre no momento da encomenda pelo número de peça. Adicionalmente, devem ser ativados os protocolos utilizados.

#### CUIDADO



**Abertura incorreta do módulo COM ou do HIMatrix**

**Danos no módulo COM ou no HIMatrix**

**O reequipamento de sub-módulos de barramento de campo apenas pode ser efetuado pela HIMA.**

##### 3.6.1.1 Estrutura do número de peça

As seções a seguir descrevem como número de peça é modificado com a atribuição das interfaces de barramento de campo no caso do HIMax X-COM 01 ou de um sistema de comando HIMatrix.

Para o número de peça são atribuídos números aos sub-módulos de barramento de campo, veja Tabela 21.

Opções para FB1 e FB2	Descrição
0	sem sub-módulo de barramento de campo montado
1	RS485 para Modbus (Master ou Slave) ou ComUserTask
2	PROFIBUS DP Master
3	PROFIBUS DP Slave
4	INTERBUS Master (apenas para HIMatrix com ELOP II Factory)
5	RS232 para ComUserTask
6	RS422 para ComUserTask
7	SSI para ComUserTask

Tabela 21: Opções para interfaces de barramento de campo FB1 e FB2

### 3.6.1.2 Número de peça do módulo COM HIMax

O módulo COM forma uma unidade funcional com a Connector Board X-CB 001 02. A Connector Board deve ser encomendada em separado.

Ao equipar o X-COM 01 com um ou mais sub-módulos de barramento de campo, além do número de peça, também muda a denominação do módulo de X-COM 01 para X-COM 010 XY.

A seguinte tabela contém os componentes disponíveis:

Denominação	Descrição
X-COM 01	Módulo de comunicação sem sub-módulos de barramento de campo
X-COM 010 <b>XY</b> <sup>1)</sup>	Módulo de comunicação com sub-módulo de barramento de campo
X-CB 001 02	Connector Board
<sup>1)</sup> <b>X</b> : Opção para interfaces de barramento de campo FB1 conforme Tabela 21 <b>Y</b> : Opção para interfaces de barramento de campo FB2 conforme Tabela 21	

Tabela 22: Componentes HIMax disponíveis

A seguinte tabela mostra exemplos para números de peça e denominações:

Número de peça	Denominação	Sub-módulo 1 barramento de campo (FB1)	Sub-módulo 2 barramento de campo (FB2)
98 52600 <b>21</b>	X-COM 010 <b>21</b>	PROFIBUS Master (máx. 12 MBit)	RS485
98 52600 <b>23</b>	X-COM 010 <b>23</b>	PROFIBUS Master (máx. 12 MBit)	PROFIBUS Slave (máx. 1,5 MBit)
98 52600 <b>11</b>	X-COM 010 <b>11</b>	RS485	RS485
98 5260000	X-COM 01	---	---

Tabela 23: Exemplos para números de peça e denominações de módulos COM

## i

A HIMA recomenda operar PROFIBUS DP pela interface de barramento de campo FB1 (taxa de transmissão máxima 12 MBit). Pela interface de barramento de campo FB2 é permitida uma taxa de transmissão máxima 1,5 MBit.

Denominação e número de peça (Part-Nr.) são impressas na placa de identificação do módulo.

## 3.6.1.3 Números de peça dos sistemas de comando HIMatrix

Os sistemas de comando HIMatrix podem ser equipados com sub-módulos de barramento de campo de acordo com a seguinte tabela:

Sistema de comando	FB1	FB2	FB3
F20	Pode ser equipado livremente	RS485 instalado	---
F30	Pode ser equipado livremente	Pode ser equipado livremente	RS485 instalado
F35	Pode ser equipado livremente	Pode ser equipado livremente	RS485 instalado
F60	Pode ser equipado livremente	Pode ser equipado livremente	---

Tabela 24: Equipamento de sistemas de comando HIMatrix com sub-módulos de barramento de campo

Mediante seleção do respectivo sub-módulo de barramento de campo, o número de peça muda:

98 22**XY**...

**X**: Opção para interfaces de barramento de campo FB1 conforme Tabela 21

**Y**: Opção para interfaces de barramento de campo FB2 conforme Tabela 21

A seguinte tabela mostra exemplos para números de peça:

Número de peça	Sistema de comando	FB1	FB2
98 221 <b>0</b> 417	F20 01	RS485	---
98 223 <b>2</b> 415	F30 01	PROFIBUS Slave	PROFIBUS Master
98 223 <b>2</b> 472	F30 01 SILworX	PROFIBUS Slave	PROFIBUS Master
98 224 <b>2</b> 416	F35 01	INTERBUS Master	PROFIBUS Master
98 223 <b>2</b> 497	F35 03 SILworX	PROFIBUS Slave	PROFIBUS Master
98 221 <b>2</b> 126	F60 CPU 01	RS485	PROFIBUS Master
98 221 <b>2</b> 137	F60 CPU 01 SILworX	RS485	PROFIBUS Master
98 001 <b>2</b> 139	F60 CPU 03 SILworX	RS485	PROFIBUS Master

Tabela 25: Exemplos para números de peça dos sistemas de comando HIMatrix

## 3.6.2 Registrar e ativar

Dependendo dos submódulos de barramento de campo, as opções de comunicação são ativadas, veja Capítulo 3.4.

## 3.6.3 Pinagens das conexões D-Sub

As seguintes tabelas descrevem as pinagens das interfaces de barramento de campo dependendo do submódulo de barramento de campo montado.



### 3.6.3.1 Submódulo de barramento de campo RS485 para Modbus Master, Slave ou ComUserTask

Conexão	Sinal	Função
1	---	---
2	RP	5 V, desacoplado por diodos
3	RxD/TxD-A	Dados de recepção/envio A
4	CNTR-A	Sinal de comando A
5	DGND	Potencial de referência dos dados
6	VP	5 V, pólo positivo tensão de alimentação
7	---	---
8	RxD/TxD-B	Dados de recepção/envio B
9	CNTR-B	Sinal de comando B

Tabela 26: Pinagem das conexões D-Sub para RS485

### 3.6.3.2 Submódulo de barramento de campo PROFIBUS DP Master ou Slave

Conexão	Sinal	Função
1	---	
2	---	
3	RxD/TxD-A	Dados de recepção/envio A
4	RTS	Sinal de comando
5	DGND	Potencial de referência dos dados
6	VP	5 V, pólo positivo tensão de alimentação
7	---	---
8	RxD/TxD-B	Dados de recepção/envio B
9	---	---

Tabela 27: Pinagem das conexões D-Sub para PROFIBUS DP

### 3.6.3.3 Barramento de campo para INTERBUS

INTERBUS apenas está disponível para sistemas de comando HIMatrix com sistema operacional anterior a CPU OS V7 e COM OS V12. Apenas suportado na ferramenta de programação ELOP II Factory.

Conexão	Sinal	Função
1	DO	Saída de dados positiva
2	DI	Entrada de dados positiva
3	COM	Condutor 0 V conjunto
4	---	---
5	---	---
6	DO-	Entrada de dados negativa
7	DI-	Saída de dados negativa
8	---	---
9	---	---

Tabela 28: Pinagem das conexões D-Sub para INTERBUS

## 3.6.3.4 Submódulo de barramento de campo RS232 para ComUserTask

Conexão	Sinal	Função
1	---	---
2	TxD	Dados de envio
3	RxD	Dados de recepção
4	---	---
5	DGND	Potencial de referência dos dados
6	---	---
7	RTS	Solicitação para enviar (Request to Send)
8	---	---
9	---	---

Tabela 29: Pinagem das conexões D-Sub para RS232

## 3.6.3.5 Submódulo de barramento de campo RS422 para ComUserTask

Conexão	Sinal	Função
1	---	---
2	RP	+5 V, desacoplado por diodos
3	RxA	Dados de recepção A
4	TxA	Dados de envio A
5	DGND	Potencial de referência dos dados
6	VP	+5 V tensão de alimentação
7	---	---
8	RxB	Dados de recepção B
9	TxB	Dados de envio B

Tabela 30: Pinagem das conexões D-Sub para RS422

## 3.6.3.6 Submódulo de barramento de campo para SSI

Conexão	Sinal	Função
1	D2+	Entrada de dados canal 2+
2	D1-	Entrada de dados canal 1-
3	CL2+/D3+	Saída de impulso de deslocamento canal 2+ ou entrada de dados canal 3+
4	CL1+	Saída de impulso de deslocamento canal 1+
5	GND	Potencial de referência
6	D1+	Entrada de dados canal 1+
7	D2-	Entrada de dados canal 2-
8	CL2-/D3-	Saída de impulso de deslocamento canal 2- ou entrada de dados canal 3-
9	CL1-	Saída de impulso de deslocamento canal 1-

Tabela 31: Pinagem das conexões D-Sub para SSI

## 4 safeethernet

Todos os sistemas HIMax/HIMatrix têm capacidade para **safeethernet**. Eles podem comunicar-se via Ethernet de forma direcionada à segurança de acordo com SIL 3 (HIMax 1 Gbit/s, HIMatrix 100 Mbit/s).

As respectivas interfaces Ethernet dos módulos HIMax CPU/COM e dos sistemas de comando HIMatrix podem ser usados simultaneamente para outros protocolos.

A comunicação **safeethernet** entre os sistemas de comando pode ocorrer mediante diversar topologias de rede Ethernet. Adaptar os parâmetros do protocolo **safeethernet** à rede Ethernet usada para aumentar a velocidade e eficácia da transferência de dados.

Estes parâmetros podem ser ajustados mediante os chamados perfis de rede. O ajuste em fábrica dos parâmetros garante a comunicação sem que o usuário precise se familiarizar com os detalhes da configuração de rede por enquanto.

### i

O protocolo **safeethernet** é direcionado à segurança e certificado pela TÜV até SIL 3 conforme IEC 61508.

#### Equipamentos necessários e requisitos de sistema:

Sistema de comando HIMA	HIMax com módulo CPU HIMatrix L2/L3
Ativação	Esta função está liberada por padrão em todos os sistemas HIMax/HIMatrix.

#### safeethernet (Properties):

Elemento	HIMax	HIMatrix L3	HIMatrix L2	Descrição
Módulo/sistema de comando necessário	No máximo 4 módulos processadores por HIMax	Módulo processador integrado do sistema de comando	Módulo processador integrado do sistema de comando	<b>safeethernet</b> é executado no módulo processador direcionado à segurança.
Interfaces Ethernet:	Módulo CPU 1 Gbit/s Módulo COM 100 Mbit/s	100 Mbit/s	100 Mbit/s	As interfaces Ethernet podem ser usadas simultaneamente para outros protocolos.
Conexões:	por HIMax 255	128	64	Conexões <b>safeethernet</b>
Conexões redundantes:	por HIMax 255	128	32 Remotel/O: n. a	Operação de 2 canais Conexões <b>safeethernet</b> redundantes entre sistemas de comando HIMax/HIMatrix podem ser ajustadas no Editor do <b>safeethernet</b> .
Caminhos de transporte redundantes	Via 2 módulos separados, respectivamente	Restrição, pois há só um dispositivo	Restrição, pois há só um dispositivo	Caminhos de transporte <b>safeethernet</b> redundantes
Volume de dados de processo por conexão	1100 Bytes	1100 Bytes	900 Bytes	Por conexão <b>safeethernet</b> .
n. a: não aplicável				

Tabela 32: Protocolo direcionado à segurança (safeethernet)

- 
- i
- Comunicação no nível superior ao do projeto!  
Conexões **safeethernet** a um recurso num outro projeto ou num sistema de comando HIMatrix com sistema operacional anterior a CPU OS V7 e COM OS V12 podem ser configuradas no SILworX, veja Capítulo 4.8.
- 

#### 4.1 O que é safeethernet

Na área da técnica de processos e automação, exigências como determinismo, confiabilidade, intercambiabilidade, modularidade e, principalmente, segurança, são assuntos centrais.

**safeethernet** é um protocolo de transmissão para a transmissão de dados direcionados à segurança até SIL 3 com base na tecnologia Ethernet.

**safeethernet** contém mecanismos que detectam os seguintes erros e reagem a eles de forma direcionada à segurança:

- Adulteração de dados transmitidos (Bits duplicados, perdidos, alterados)
- Endereçamento incorreto de mensagens (emissor, destinatário)
- Sequência incorreta de dados (repetição, perda, inversão)
- Comportamento de tempo incorreto (retardamento, eco)

**safeethernet** é baseado no padrão IEEE 802.3.

A transmissão de dados relevantes para a segurança usa o quadro do protocolo Ethernet padrão.

**safeethernet** utiliza “canais inseguros de transmissão de dados” (Ethernet) segundo o princípio “Black Channel” e os monitora no emissor e receptor mediante mecanismos de protocolo direcionados à segurança. Assim, componentes de rede Ethernet, tais como hubs, switches, roteadores, podem ser utilizados dentro de uma rede direcionada à segurança.

**safeethernet** utiliza as capacidades da Ethernet padrão de uma forma que possibilita a segurança e capacidade de operação em tempo real. Um mecanismo de protocolo especial garante um comportamento determinístico mesmo no caso de falha ou entrada de participantes de comunicação. O sistema automaticamente integra novos componentes no sistema em execução. Todos os componentes numa rede são substituíveis durante a operação em andamento. Com a utilização de switches, os tempos de transmissão podem ser claramente definidos. Assim, a Ethernet ganha capacidade de operação em tempo real.

Também a velocidade de transmissão de dados de até 1 Gbit/s para dados direcionados à segurança está acima dos padrões normalmente usados. Como meios de transmissão podem ser usados, p. ex., condutores de cobre e condutores de FO.

Conexões à intranet da empresa e também conexões à internet são possíveis com **safeethernet**. Assim, apenas uma rede é necessária para a transmissão de dados seguros e não seguros.

- 
- i
- A rede pode ser utilizada por outros participantes se houver o suficiente de capacidade de transmissão.
-

**⚠ PERIGO**

**Manipulação da transmissão de dados direcionada à segurança!**

**Possibilidade de danos pessoais**

A empresa operadora deve garantir que a rede Ethernet utilizada para safeethernet seja protegida de forma suficiente contra manipulações (p. ex., por hackers). O tipo e a abrangência das medidas devem ser autorizados pela respectiva instituição de certificação.

A **safeethernet** permite estruturas de sistema flexíveis para a automação descentralizada com tempos definidos de reação. Conforme a necessidade, a inteligência pode ser distribuída opcionalmente de forma central ou descentralizada entre os participantes dentro da rede.

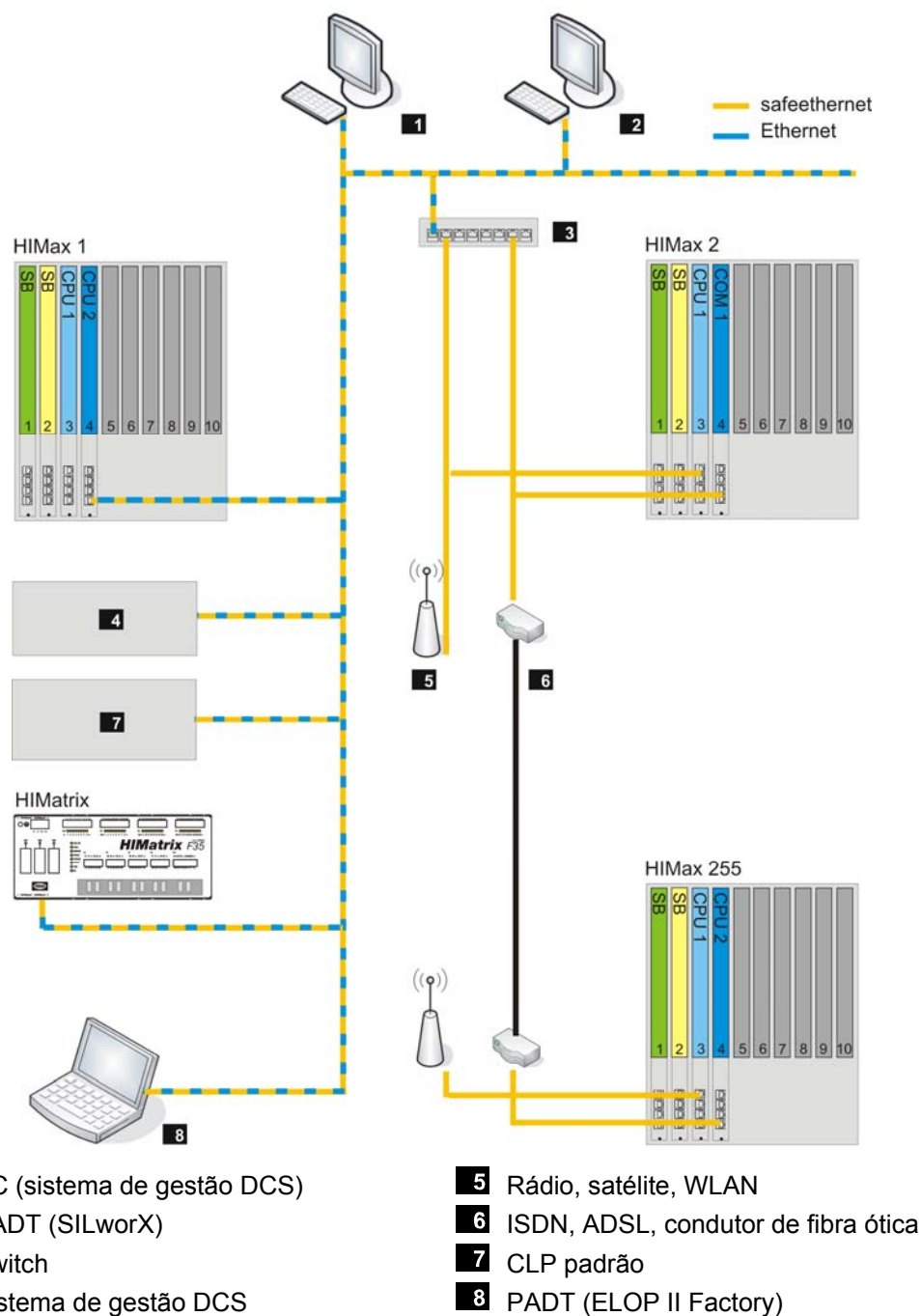


Figura 1: Estruturas de sistema

---

**i**

É possível uma passagem involuntária para o estado seguro!

Na ligação deve ser observado que não sejam criados laços de rede. Pacotes de dados apenas podem chegar por um caminho ao sistema de comando.

---

## 4.2 Configuração de uma conexão safeethernet redundante

Neste exemplo, é configurada uma conexão **safeethernet** HIMax/HIMax redundante.

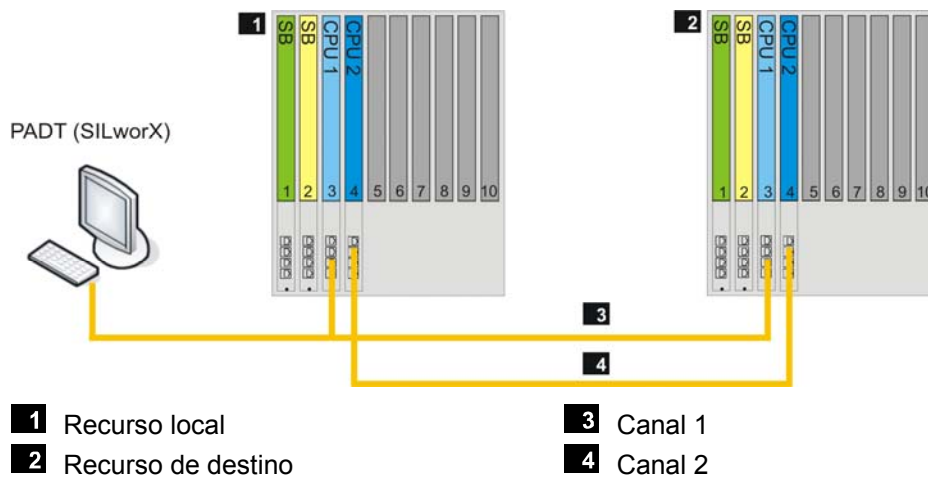


Figura 2: Estrutura para a configuração de uma conexão redundante

i

A HIMA recomenda no caso de uma conexão **safeethernet** redundante conduzir os dois caminhos de transporte (canal 1 e canal 2) por módulos de comunicação separados. Neste caso, a largura de banda e o retardo nos dois caminhos de transporte devem ser praticamente idênticos.

## Estabelecer uma conexão safeethernet

Estabelecer no Editor **safeethernet** uma conexão **safeethernet** entre o recurso local e o recurso de destino.

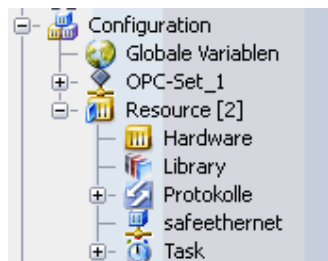


Figura 3: Árvore de estrutura do recurso

**Assim abre-se o Editor de safeethernet do recurso local:**

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **saferethernet** e selecionar no menu de contexto **Edit**.  
☒ Na seleção de objetos estão os recursos de destino.

**Assim se estabelece a conexão safeethernet ao recurso de destino:**

1. Na seleção de objetos, clicar no **Target Resource** e puxar mediante Drag&Drop para um local livre na área de trabalho do safe**ethernet** Editor.

**i**

O caminho de comunicação inverso é adicionado automaticamente ao recurso de destino no Editor **safeethernet**.

**Assim se configura a conexão safeethernet:**

1. Selecionar **Ethernet Interfaces Channel 1** do recurso local e do recurso de destino.
2. Selecionar **Ethernet Interfaces Channel 2** do recurso local e do recurso de destino.
3. Selecionar **Network Profile** (p. ex., Fast&Noisy) da conexão safeethernet.
4. Calcular **Receive Timeout** e **Response Time** e introduzir (veja Capítulo 4.6).

F	Partner	IF CH 1 (local)	IF CH 2 (local)	IF CH 1 (target)	IF CH 2 (target)	Profile	Response Time [ms]	Receive Timeout [ms]
1	Resource_01	22.0.3 (192.168.0.21)	22.0.4 (192.168.0.22)	33.0.3 (192.168.0.11)	33.0.4 (192.168.0.12)	Fast & ...	100	500

F	Resources	Configuration	System ID
1	Resource_01 [33]	Configuration	

Figura 4: Valores dos parâmetros de uma conexão safeethernet redundante

## 4.2.1 Conectar variáveis de processo

**Assim se abre a visualização de detalhes duma conexão safeethernet:**

Requisito: Editor safeethernet do recurso local está aberto.

1. Abrir o menu de contexto do **Target Resource** mediante clique com o botão direito.
2. Selecionar **Detail View**.
3. Selecionar o registro **Resource (Target) <-> Resource (local)**.

F	Data type	Global Variable
1	WORD	Resource2_Resource1_01
2	WORD	Resource2_Resource1_02

F	Data type	Global Variable
1	BYTE	Resource1_Resource2_01
2	BYTE	Resource1_Resource2_02
3	WORD	Resource1_Resource2_03

F	Name	Data type	Init Value	Description	Technical unit	Retain	Constant	Pathname
1	Resource1_Resource2_01	BYTE				<input type="checkbox"/>	<input type="checkbox"/>	/Configuration/Var Globals
2	Resource1_Resource2_02	BYTE				<input type="checkbox"/>	<input type="checkbox"/>	/Configuration/Var Globals
3	Resource1_Resource2_03	WORD				<input type="checkbox"/>	<input type="checkbox"/>	/Configuration/Var Globals

Figura 5: Visualização de detalhes Editor safeethernet

i

Apenas podem ser usadas variáveis globais do contexto da configuração, não do contexto do recurso!

**Assim se adicionam as variáveis de envio safeethernet:**

As variáveis de envio são enviadas do recurso local ao recurso de destino

1. Selecionar a área **Resource (Target) -> Resource (local)**.
2. Na seleção de objetos, selecionar uma **Global Variable** e puxar via Drag&Drop para a coluna **Resource (Target) -> Resource (local)**.
3. Repetir este passo para outras variáveis de envio safeethernet.



**Assim se adicionam as variáveis de recepção safeethernet:**

As variáveis de recepção são recebidas do recurso local.

1. Selecionar a área **Resource (Target) -> Resource (local)**.
2. Na seleção de objetos, selecionar uma **Global Variable** e puxar via Drag&Drop para a coluna **Resource (Target) <- Resource (local)**.
3. Repetir este passo para outras variáveis de envio safeethernet.

**Verificar uma conexão safeethernet:**

1. Abrir na árvore de estrutura **Configuration, Resource, safeethernet**.
2. Clicar no botão **Verification** no Action Bar e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir erros onde for necessário.



A configuração da conexão safeethernet deve ser novamente compilada com o programa de aplicação do recurso local e do recurso de destino e transmitida para os sistemas de comando antes de se tornar efetiva para a comunicação do sistema HIMax.

---

#### 4.2.2 Verificação da comunicação safeethernet

Resetar no Control Panel os indicadores *Wrong Messages* e *Resends* para zero.

1. Operar o sistema HIMax sob carga total:
  - Todos os protocolos de comunicação estão em operação (safeethernet e protocolos padrão).
  - Puxar os módulos processadores para desconectar e inserir novamente como descrito no Capítulo 4.6.1.
  - Carregar o programa de aplicação via Reload (alteração da configuração safeethernet via Reload não é possível).



Para verificar a configuração correta duma conexão safeethernet redundante deve-se separar primeiro uma conexão redundante e adicionar novamente, depois a outra. Neste caso não podem ocorrer falhas na comunicação safeethernet.

---

2. Verificar nos Control Panels dos dois sistemas de comando os indicadores *Wrong Messages* e *Resends*.  
Verificar se os contadores de *Bad Messages* e *Resends*  
**= 0**, os ajustes da safeethernet estão OK.  
**≥ 0**, os ajustes safeethernet devem ser verificados de novo.
  - Calcular o *Receive Timeout* novamente usando o tempo máximo de ciclo, veja Capítulos 4.6.1 e 4.6.3.
  - Variar o *Response Time* conforme descrito no Capítulo 4.6.4.



Outras causas para *bad messages* e *resends*!

Verificar a estrutura correta da rede (p. ex., condutores, Switches, PCs).  
Se a rede Ethernet não for utilizada exclusivamente para safeethernet, deve ser verificada além disso a carga de rede (probabilidade de colisões de dados).

---

### 4.3 safeethernet-Editor

No Editor **safeethernet** são criadas e configuradas as conexões de **safeethernet** aos parceiros de comunicação (recursos).

**Assim abre-se o Editor de safeethernet do recurso local:**

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **safeethernet** e selecionar no menu de contexto **Edit**.

☒ O Editor **safeethernet** contém a área de trabalho e a seleção de objetos.

No Editor **safeethernet** são criadas e configuradas as conexões de **safeethernet** aos parceiros de comunicação (recursos). Para isso, puxam-se os recursos da seleção de objetos para a área de trabalho.

Para a configuração da conexão **safeethernet**, os seguintes parâmetros do protocolo **safeethernet** devem ser ajustados:

Parâmetro	Descrição
Partner	Nome do recurso do parceiro no vínculo
IF CH...	Interfaces Ethernet disponíveis no recurso (local) e no recurso (destino), veja também Capítulo 3.5.
Profile	Combinação de parâmetros compatíveis de <b>safeethernet</b> , veja também Capítulo 4.7.8.
Response Time [ms]	Tempo até a confirmação de recepção de uma mensagem no remetente, veja também Capítulo 4.6.4.
Receive Timeout [ms]	Tempo de supervisão em PES 1, dentro do qual uma resposta correta deve ser recebida do PES 2, veja também Capítulo 4.6.3.
Resend Timeout [ms]	Tempo de supervisão no PES1 dentro do qual PES2 deve ter confirmado a recepção de um pacote de dados, outrossim, o pacote de dados é repetido, veja também Capítulo 4.6.6.
Acknowledge Timeout [ms]	Tempo depois do qual um pacote de dados recebido deve ser confirmado pela CPU o mais tardar, veja também Capítulo 4.6.7.
Prod.-Rate	Production Rate: Menor intervalo de tempo entre dois pacotes de dados, veja também Capítulo 4.6.8.
Queue	Quantidade de pacotes de dados que podem ser enviados sem confirmação de recepção, veja também Capítulo 4.6.9.

Parâmetro	Descrição
Freeze Values on Lost Connection [ms]	<p>Comportamento das variáveis de entrada desta conexão <b>safeethernet</b> no caso da interrupção da conexão.</p> <p>Use initial values      Os dados iniciais são usados para as variáveis de entrada.</p> <p>Not limited              As variáveis de entrada são congeladas para o valor atual e são usadas até a conexão ser restabelecida.</p> <p>Limited                  Introdução: clique duplo no campo de seleção e introduzir o tempo. As variáveis de entrada são congeladas para o valor atual e são usadas até depois de um Timeout parametrizado ser ultrapassado. Depois disso, os dados iniciais são usados. O Timeout pode se prolongar por até mais um ciclo de CPU.</p> <div style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p><b>⚠ CUIDADO</b></p> <p><b>Para funções direcionadas à segurança que são realizadas via safeethernet, apenas pode ser usado o ajuste <i>Use Initial Data</i>.</b></p> </div>
Fragments per Cycle	<p>Ajuste fixo: um fragmento por ciclo do sistema de comando é transmitido ao parceiro de comunicação.</p> <p>Um fragmento de um sistema HIMax é um fragmento <math>\leq 1100</math> Byte</p> <p>Um fragmento de um sistema HIMax é um fragmento <math>\leq 900</math> Byte</p>
Priority of Events	<p>A função apenas está ativada para a conexão ao X-OPC Server. Assim, define-se com que prioridade o X-OPC Server solicita eventos do sistema de comando.</p> <p>Fragmentos com a prioridade <b>n</b> e fragmentos com a prioridade <b>m</b> são enviados na proporção <b>n</b> a <b>m</b> vezes.</p>
Priority of State Values	<p>A função apenas está ativada para a conexão ao X-OPC Server. Assim, define-se com que prioridade o X-OPC Server solicita valores de estado do sistema de comando.</p> <p>Fragmentos com a prioridade <b>n</b> e fragmentos com a prioridade <b>m</b> são enviados na proporção <b>n</b> a <b>m</b> vezes.</p>
Number of ignored warnings	É a quantidade de alertas que precisam ocorrer sequencialmente dentro do período de <i>Warning Period [ms]</i> até serem incluídos no diagnóstico ou na estatística de erros de comunicação.
Warning Period [ms]	Atualmente, 0 ms é o único valor admissível.
Enable SOE	Valor padrão: Desativado

Tabela 33: Parâmetros protocolo **safeethernet**

### Seleção de objetos

A seleção de objetos disponibiliza todos os recursos dentro deste projeto com os quais este recurso pode ser conectado via **safeethernet**.

## i

Para conexões **safeethernet** para recursos fora de um projeto ou para um sistema de comando HIMatrix (projetado em ELOP II Factory), há uma função de exportação à disposição (veja Capítulo 4.8).

## 4.4 Visualização de detalhes do Editor safeethernet

O **Detail View** sempre se refere ao recurso local para o qual o safeethernet Editor foi iniciado.

**Assim se abre a visualização de detalhes duma conexão safeethernet:**

1. Clicar com o botão direito na conexão safeethernet, abrir o menu de contexto.
2. Clicar em **Detail View**.
  - ☒ A **Detail View** contém o registro **System Variables, Fragment Definitions e Resource (local) <-> Resource (Target)**.

### 4.4.1 Registro: Variáveis de sistema

É possível controlar a conexão safeethernet no programa de aplicação com ajuda de variáveis de sistema e avaliar o seu status.

Variável de sistema	Descrição										
Ack. Frame No.	Contador de recepção (circular).										
Number of bad messages	Quantidade de todas as mensagens defeituosas por canal (CRC incorreto, header incorreto, outros erros)										
Number of bad messages for the redundant channel											
Number of successful connections	Quantidade de sucessos de conexão depois de resetar a estatística.										
Number of lost messages	Quantidade de mensagens falhadas em um dos dois caminhos de transporte depois de resetar a estatística. O contador apenas é atualizado até a falha completa de um canal.										
Number of lost messages for the redundant channel											
Early Queue Usage	Quantidade de mensagens que foram colocadas na Early Queue depois de resetar a estatística, veja também Capítulo 4.6.9.										
Bad messages	Quantidade de mensagens descartadas depois de resetar a estatística.										
Frame No.	Contador de envio (circular).										
Channel state	<p>Estado atual do canal 1. O estado do canal é o estado atual do canal 1 no momento (Seq-No X-1) da recepção de uma mensagem com Seq-No X.</p> <table> <tr> <th>Status</th><th>Descrição</th></tr> <tr> <td>0</td><td>Não há mensagem sobre o estado do canal 1.</td></tr> <tr> <td>1</td><td>Canal 1 OK.</td></tr> <tr> <td>2</td><td>A última mensagem tinha erros, a atual está OK.</td></tr> <tr> <td>3</td><td>Erro no canal 1.</td></tr> </table>	Status	Descrição	0	Não há mensagem sobre o estado do canal 1.	1	Canal 1 OK.	2	A última mensagem tinha erros, a atual está OK.	3	Erro no canal 1.
Status	Descrição										
0	Não há mensagem sobre o estado do canal 1.										
1	Canal 1 OK.										
2	A última mensagem tinha erros, a atual está OK.										
3	Erro no canal 1.										
Layout version	Assinatura do layout de dados usado na comunicação.										
Last channel latency	<p>A latência do canal indica o retardo entre os dois caminhos de transporte redundantes até o momento da recepção de mensagens com o SeqNo idêntico. Para isso é mantida uma estatística com latência média, mínima, máxima e última.</p> <p>Se o valor mín &gt; valor máx, os valores da estatística são inválidos.</p> <p>Neste caso, a Última latência do canal e a Média da latência do canal são 0.</p>										
Last latency of the red. channel											
Max. channel latency											
Max. channel latency of the red. channel											
Min. channel latency											
Min. channel latency of the red. channel											
Average channel latency											

Variável de sistema	Descrição																		
Average latency of the red. channel																			
Monotony	Contador de envio de dados de trabalho (circular).																		
New layout version	Assinatura do novo layout de dados.																		
Quality of channel 1	Estado do caminho de transporte principal. <table><tr><th>Bit N°.</th><th>Bit = 0</th><th>Bit = 1</th></tr><tr><td>0</td><td>Caminho de transporte não liberado</td><td>Caminho de transporte liberado</td></tr><tr><td>1</td><td>Caminho de transporte não utilizado</td><td>Caminho de transporte utilizado ativamente</td></tr><tr><td>2</td><td>Caminho de transporte não conectado</td><td>Caminho de transporte conectado</td></tr><tr><td>3</td><td>-</td><td>Caminho de transporte entrega a mensagem primeiro</td></tr><tr><td>4–7</td><td>Reservado</td><td>Reservado</td></tr></table>	Bit N°.	Bit = 0	Bit = 1	0	Caminho de transporte não liberado	Caminho de transporte liberado	1	Caminho de transporte não utilizado	Caminho de transporte utilizado ativamente	2	Caminho de transporte não conectado	Caminho de transporte conectado	3	-	Caminho de transporte entrega a mensagem primeiro	4–7	Reservado	Reservado
Bit N°.	Bit = 0	Bit = 1																	
0	Caminho de transporte não liberado	Caminho de transporte liberado																	
1	Caminho de transporte não utilizado	Caminho de transporte utilizado ativamente																	
2	Caminho de transporte não conectado	Caminho de transporte conectado																	
3	-	Caminho de transporte entrega a mensagem primeiro																	
4–7	Reservado	Reservado																	
Quality of channel 2	Estado do caminho de transporte redundante, veja estado canal 1 (caminho de transporte principal).																		
Receive Timeout	Tempo em milissegundos (ms) em PES 1, dentro do qual uma resposta válida deve ser recebida do PES 2, veja também Capítulo 4.6.3.																		
Response Time	Tempo em milissegundos (ms) até a confirmação de recepção de uma mensagem no remetente, veja também Capítulo 4.6.4.																		
Reset safeethernet statistics	Resetar os valores de estatística para a conexão de comunicação no programa de aplicação (p. ex., <i>Number of Bad Messages</i> , <i>Channel State</i> , <i>Timestamp for the Last Fault on the Red. Channel [s]</i> , <i>Resends</i> ). <table><tr><th>Valor</th><th>Função</th></tr><tr><td>0</td><td>Sem Reset</td></tr><tr><td>1–255</td><td>Reset safeethernet statistics</td></tr></table>	Valor	Função	0	Sem Reset	1–255	Reset safeethernet statistics												
Valor	Função																		
0	Sem Reset																		
1–255	Reset safeethernet statistics																		
Transmission Control Ch1	Controle do transporte do canal 1 <table><tr><th>Bit 0</th><th>Função</th></tr><tr><td>FALSE</td><td>Caminho de transporte liberado</td></tr><tr><td>TRUE</td><td>Caminho de transporte bloqueado</td></tr></table> <table><tr><th>Bit 1</th><th>Função</th></tr><tr><td>FALSE</td><td>Caminho de transporte liberado para testes</td></tr><tr><td>TRUE</td><td>Caminho de transporte bloqueado</td></tr></table> Bit 2...7 reservado.	Bit 0	Função	FALSE	Caminho de transporte liberado	TRUE	Caminho de transporte bloqueado	Bit 1	Função	FALSE	Caminho de transporte liberado para testes	TRUE	Caminho de transporte bloqueado						
Bit 0	Função																		
FALSE	Caminho de transporte liberado																		
TRUE	Caminho de transporte bloqueado																		
Bit 1	Função																		
FALSE	Caminho de transporte liberado para testes																		
TRUE	Caminho de transporte bloqueado																		
Transmission Control Ch2	Controle do transporte do canal 2, veja Controle do transporte do canal 1.																		

Variável de sistema	Descrição										
Connection Control	<p>Com esta variável de sistema, a conexão <b>safeethernet</b> pode ser controlada pelo programa de aplicação.</p> <table> <tr> <th>Comando</th><th>Descrição</th></tr> <tr> <td>Autoconnect (0x0000)</td><td>Valor padrão: Após perda da comunicação <b>safeethernet</b>, o sistema de comando tenta restabelecer a conexão no próximo ciclo de CPU.</td></tr> <tr> <td>Toggle Mode 0 (0x0100) Toggle Mode 1 (0x0101)</td><td> <p>Após a perda da comunicação, é possível restabelecer a conexão mediante mudança do Toggle Mode controlada pelo programa.</p> <ul style="list-style-type: none"> <li>▪ TOGGLE MODE_0 (0x100) atribuído: Colocar em TOGGLE MODE 1 (0x101) para restabelecer a conexão.</li> <li>▪ TOGGLE MODE 1 (0x101) atribuído: Colocar em TOGGLE MODE 0 (0x100) para restabelecer a conexão.</li> </ul> </td></tr> <tr> <td>Disabled (0x8000)</td><td>A comunicação <b>safeethernet</b> está desligada.</td></tr> </table>	Comando	Descrição	Autoconnect (0x0000)	Valor padrão: Após perda da comunicação <b>safeethernet</b> , o sistema de comando tenta restabelecer a conexão no próximo ciclo de CPU.	Toggle Mode 0 (0x0100) Toggle Mode 1 (0x0101)	<p>Após a perda da comunicação, é possível restabelecer a conexão mediante mudança do Toggle Mode controlada pelo programa.</p> <ul style="list-style-type: none"> <li>▪ TOGGLE MODE_0 (0x100) atribuído: Colocar em TOGGLE MODE 1 (0x101) para restabelecer a conexão.</li> <li>▪ TOGGLE MODE 1 (0x101) atribuído: Colocar em TOGGLE MODE 0 (0x100) para restabelecer a conexão.</li> </ul>	Disabled (0x8000)	A comunicação <b>safeethernet</b> está desligada.		
Comando	Descrição										
Autoconnect (0x0000)	Valor padrão: Após perda da comunicação <b>safeethernet</b> , o sistema de comando tenta restabelecer a conexão no próximo ciclo de CPU.										
Toggle Mode 0 (0x0100) Toggle Mode 1 (0x0101)	<p>Após a perda da comunicação, é possível restabelecer a conexão mediante mudança do Toggle Mode controlada pelo programa.</p> <ul style="list-style-type: none"> <li>▪ TOGGLE MODE_0 (0x100) atribuído: Colocar em TOGGLE MODE 1 (0x101) para restabelecer a conexão.</li> <li>▪ TOGGLE MODE 1 (0x101) atribuído: Colocar em TOGGLE MODE 0 (0x100) para restabelecer a conexão.</li> </ul>										
Disabled (0x8000)	A comunicação <b>safeethernet</b> está desligada.										
Connection State	<p>O estado da conexão avalia o status da comunicação entre dois sistemas de comando no programa de aplicação.</p> <table> <tr> <th>Status/valor</th><th>Descrição</th></tr> <tr> <td>Closed (0)</td><td>A conexão está encerrada e não será tentado abri-la novamente.</td></tr> <tr> <td>Try_open (1)</td><td>Tentativa de abrir a conexão, porém, ainda não está aberta. Este estado vale igualmente para o lado ativo e passivo.</td></tr> <tr> <td>Connected (2)</td><td>A conexão foi estabelecida e está operando (Supervisão de tempo e troca de dados ativas)</td></tr> </table>	Status/valor	Descrição	Closed (0)	A conexão está encerrada e não será tentado abri-la novamente.	Try_open (1)	Tentativa de abrir a conexão, porém, ainda não está aberta. Este estado vale igualmente para o lado ativo e passivo.	Connected (2)	A conexão foi estabelecida e está operando (Supervisão de tempo e troca de dados ativas)		
Status/valor	Descrição										
Closed (0)	A conexão está encerrada e não será tentado abri-la novamente.										
Try_open (1)	Tentativa de abrir a conexão, porém, ainda não está aberta. Este estado vale igualmente para o lado ativo e passivo.										
Connected (2)	A conexão foi estabelecida e está operando (Supervisão de tempo e troca de dados ativas)										
Repeats	Quantidade de repetições depois de resetar a estatística.										
Timestamp for the last fault on the red. channel [ms]	Fração de milissegundos do carimbo de hora (hora de sistema atual).										
Timestamp for the last fault on the red. channel [s]	Fração de segundos do carimbo de hora (hora de sistema atual).										
Timestamp for the last fault [ms]	Fração de milissegundos do carimbo de hora (hora de sistema atual).										
Timestamp for the last fault [s]	Fração de segundos do carimbo de hora (hora de sistema atual).										
State of the red. channel	<p>Estado atual do canal 2.</p> <p>O estado do canal é o estado atual do canal 2 no momento (Seq-No X-1) da recepção de uma mensagem com Seq-No X.</p> <table> <tr> <th>Status</th><th>Descrição</th></tr> <tr> <td>0</td><td>Não há mensagem sobre o estado do canal 2</td></tr> <tr> <td>1</td><td>Canal 2 OK.</td></tr> <tr> <td>2</td><td>A última mensagem tinha erros, a atual está OK.</td></tr> <tr> <td>3</td><td>Erro no canal 2.</td></tr> </table>	Status	Descrição	0	Não há mensagem sobre o estado do canal 2	1	Canal 2 OK.	2	A última mensagem tinha erros, a atual está OK.	3	Erro no canal 2.
Status	Descrição										
0	Não há mensagem sobre o estado do canal 2										
1	Canal 2 OK.										
2	A última mensagem tinha erros, a atual está OK.										
3	Erro no canal 2.										

Tabela 34: Registro variáveis de sistema no **safeethernet** Editor

#### 4.4.2 Registro: definições de fragmento

Veja Capítulo 10.6.8.2.

### 4.5 Possíveis conexões safeethernet

É possível estabelecer uma conexão **safeethernet** entre dois sistemas de comando HIMax de forma mono ou redundante.

As interfaces Ethernet disponíveis para uma conexão **safeethernet** sempre são exibidas em relação ao recurso (local) para o qual o Editor **safeethernet** foi iniciado.

Todas as interfaces Ethernet de um sistema de comando são exibidas nos menus se seleção dos parâmetros **IF CH...**

Elemento	Descrição
IF CH1 (local)	Interface Ethernet (canal 1) do recurso
IF CH2 (local)	Interface Ethernet (canal 2) do recurso
IF CH1 (destino)	Interface Ethernet (canal 1) do parceiro vinculado
IF CH2 (destino)	Interface Ethernet (canal 2) do parceiro vinculado

Tabela 35: Interfaces Ethernet disponíveis

#### 4.5.1 Conexão safeethernet mono (canal 1)

Para uma conexão mono, é necessário atribuir as interfaces Ethernet *IF CH1 (local)* e *IF CH1 (target)* no recurso local.

#### 4.5.2 Conexão safeethernet redundante (canal 1 e canal 2)

Caminhos de transporte **safeethernet** redundantes entre dois sistemas de comando HIMax/HIMax são possíveis.

#### i

Os caminhos de transporte redundantes devem ser iguais ao ponto da sua largura de banda e o seu retardo serem quase idênticos.

No momento em que em um caminho de transporte a defasagem de mensagens recebidas se tornar excessiva ou as mensagens chegarem com atraso maior do que o Response Time, o diagnóstico do caminho de transporte não está trabalhando como previsto e está interpretando estes atrasos como falhas do caminho de transporte.

Para a avaliação do diagnóstico do caminho de transporte, veja variáveis de sistema *State of the Red. Channel* e *Channel State*.

Para uma conexão redundante, as seguintes interfaces Ethernet podem ser usadas:

- As interfaces Ethernet *IF CH1 (local)* e *IF CH1 (target)* para canal 1.
- As interfaces Ethernet *IF CH2 (local)* e *IF CH2 (target)* para canal 2.
- Para uma conexão redundante via canal 1 e canal 2 através de apenas uma interface Ethernet, selecionar no Editor **safeethernet** a mesma interface Ethernet em *IF CH1 (local)* canal 1 e *IF CH2 (local)* canal 2.

#### i

O caminho de comunicação inverso é adicionado automaticamente ao recurso de destino no Editor **safeethernet**.

#### 4.5.3 Combinações permitidas

Na seguinte tabela são mostradas as combinações possíveis para uma conexão **safeethernet** redundante.

Canal 1 IF CH1 (local)/IF CH1 (destino)	Canal 2 IF CH2 (local)/IF CH2 (destino)
CPU1/CPU1	CPU2/CPU2
CPU1/CPU1	CPU1/CPU1
COM1/COM1	COM2/COM2
CPU1/COM1	CPU2/COM2
CPU1/COM2	CPU2/COM1
CPU1/CPU1	COM1/COM1

Tabela 36: Combinações para conexões safeethernet

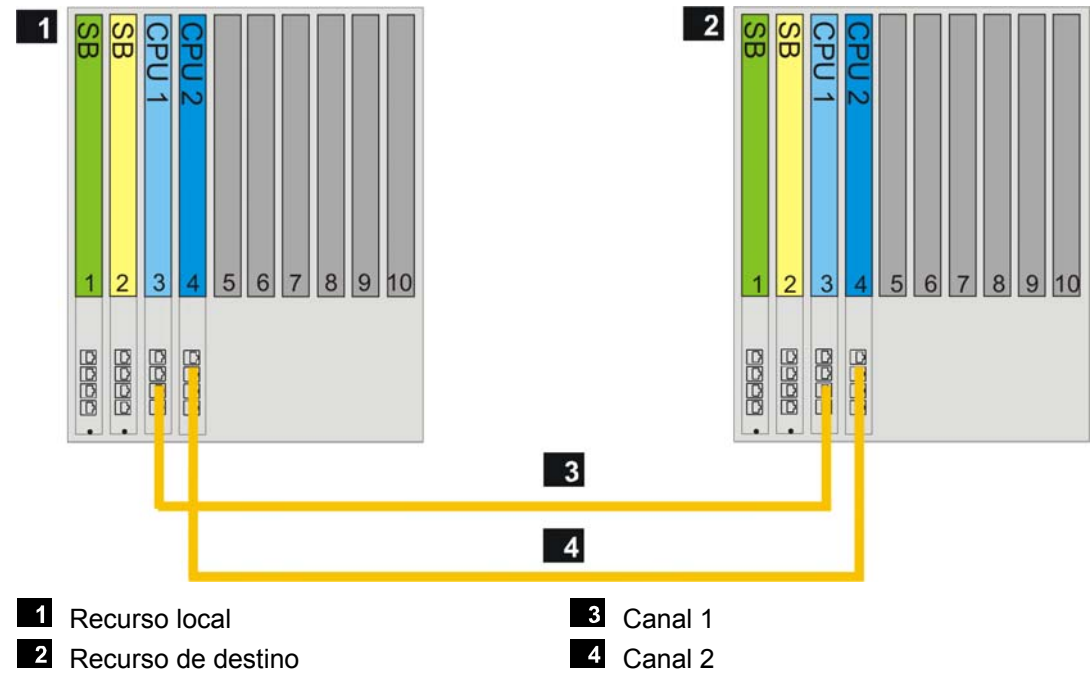


Figura 6: Conexão redundante entre dois sistemas de comando HIMax



A seguinte conexão configurada como redundante (via um condutor) faz sentido para se obter um fluxo de dados constante apesar de pacotes de dados perdidos (p. ex., por causa de interferências CEM).

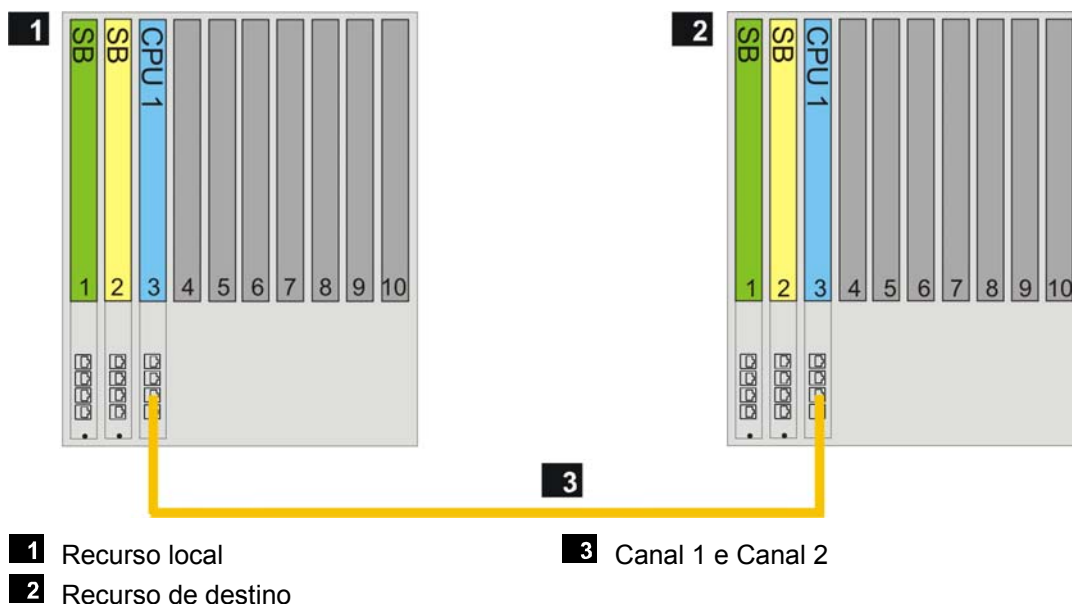


Figura 7: Conexão redundante entre dois sistemas de comando HIMax através de um condutor

## 4.6 Parâmetros safeethernet

A comunicação direcionada à segurança é configurada no Editor **safeethernet**. Para este fim, devem ser parametrizados os parâmetros descritos neste capítulo.

Para o cálculo dos parâmetros **safeethernet** *Receive Timeout* e *Response Time* vale a seguinte condição:

A fatia de tempo deve ser suficientemente grande para processar em um ciclo de CPU todas as conexões **safeethernet**, veja Capítulo 13.2.

### 4.6.1 Tempo de ciclo máximo (tempo de Watchdog mínimo) do sistema de comando HIMax

Para determinar o tempo de ciclo máximo (tempo de Watchdog mínimo) para um sistema de comando HIMax, a HIMA recomenda o seguinte procedimento em um sistema completo onde todos os módulos processadores estão colocados.

1. Ajustar um tempo de Watchdog elevado para o teste.
2. Operar o sistema sob carga integral. Neste momento, todas as conexões de comunicação devem estar em operação, tanto via **safeethernet** quanto através de protocolos padrão. Ler com frequência o tempo de ciclo no Control Panel e anotar as oscilações do tempo de ciclo.
3. Sucessivamente remover cada módulo processador e voltar a inseri-lo no suporte básico. Antes de remover um módulo processador, aguardar até que o módulo processador inserido há pouco tenha se sincronizado.

---

**i**

Ao acrescentar um módulo processador, este se sincroniza automaticamente com a configuração dos módulos processadores presentes. O tempo necessário para a sincronização prolonga o ciclo do sistema de comando para o tempo de ciclo máximo.

O tempo necessário para a sincronização cresce com a quantidade de módulos processadores já sincronizados.

Para mais informações sobre a descrição da montagem e desmontagem de um módulo processador, veja manual X-CPU 01, HI 801 254 P.

---

4. No histórico de diagnóstico do módulo não sincronizado, ler o tempo de sincronização de n para n+1 módulos processadores em cada processo de sincronização.
  5. Repetir estes passos para o parceiro de comunicação (segundo sistema de comando HIMax). A maior parte destes tempos de sincronização é utilizado para a determinação do tempo de Watchdog.
- 

**i**

Anotar também os tempos de sincronização dos dois sistemas de comando HIMax!

---

6. Calcular o mínimo tempo de Watchdog:  
maior tempo de sincronização + 12 ms reserva + reserva para as oscilações anotadas.
- Desta forma está determinada uma definição adequada do tempo de ciclo máximo (tempo de Watchdog mínimo) para os cálculos posteriores.
- 

**DICA**

Executar o cálculo no passo 6 para ambos os sistemas de comando HIMax com os respectivos tempos de sincronização anotados.

Os tempos de ciclo máximos (tempos de Watchdog mínimos) determinados desta maneira podem ser utilizados para o tempo de Watchdog do respectivo recurso, veja Manual de segurança HI 801 241 P.

---

#### 4.6.2 Tempo de ciclo máximo do sistema de comando HIMatrix

Para determinar o tempo de ciclo máximo (tempo de Watchdog mínimo) para um sistema de comando HIMatrix, a HIMA recomenda o seguinte procedimento:

##### Determinar o tempo de ciclo máximo do sistema de comando HIMatrix

1. Operar o sistema sob carga total. Neste momento, todas as conexões de comunicação devem estar em operação, tanto via safeethernet quanto através de protocolos padrão. Ler com frequência o tempo de ciclo no Control Panel e anotar o tempo de ciclo máximo.
2. Repetir passo 1 para o parceiro de comunicação (segundo sistema de comando HIMatrix).
3. O maior dos dois tempos de ciclo máximos determinados é o tempo de ciclo máximo procurado.

O tempo de ciclo máximo está determinado e entra nos cálculos posteriores.

#### 4.6.3 Receive Timeout

*ReceiveTMO* é o tempo de supervisão em milissegundos (ms) dentro do qual uma resposta correta do parceiro de comunicação precisa ser recebida.

Se dentro do *ReceiveTMO* não chegar uma resposta correta do parceiro de comunicação, a comunicação direcionada à segurança é encerrada. As variáveis de Input destas conexão **safeethernet** se comportam de acordo com o parâmetro ajustado *Freeze Data on Lost Connection [ms]*.

Para funções direcionadas à segurança que são realizadas via **safeethernet**, apenas pode ser usado o ajuste **Use Initial Data**.

Como o *ReceiveTMO* é relevante para a segurança e faz parte do Worst Case Reaction Time  $T_R$  (tempo máximo de reação, veja Capítulo 4.7.1 e continuação), o *ReceiveTMO* deve ser calculado como segue e introduzido no **safeethernet** Editor.

##### **ReceiveTMO $\geq 4 * \text{Delay} + 5 * \text{tempo máx. de ciclo}$**

Condição: A fatia de tempo de comunicação deve ser suficientemente grande para processar em um ciclo de CPU todas as conexões **safeethernet**.

Delay: Retardo no trajeto de transmissão, p.ex., causado por (Switch, Satélite, etc.)

Tempo máx. de ciclo: Tempo máximo do ciclo dos dois sistemas de comando



Uma tolerância desejável a falhas da comunicação pode ser alcançada mediante aumento do *ReceiveTMO*, caso seja admissível para o processo de aplicação em termos de tempo.

---

#### 4.6.4 ResponseTime

O *ResponseTime* é o tempo em milissegundos (ms) que passa até o remetente de uma mensagem recebe a confirmação do recebimento do destinatário.

Para a parametrização com uso de um perfil de **safeethernet**, deve ser definido um *ResponseTime* previsto de acordo às características físicas do trajeto de transmissão.

O *ResponseTime* definido influencia a configuração de todos os parâmetros da conexão **safeethernet** que deve ser calculado como segue:

$$\text{ResponseTime} \leq \text{ReceiveTMO} / n$$

$n = 2, 3, 4, 5, 6, 7, 8, \dots$

A relação entre o *ReceiveTMO* e o *ResponseTime* influencia a capacidade de tolerância de falhas, p. ex., no caso de perdas de pacotes (repetição de pacotes de dados perdidos) ou no caso de atrasos no trajeto de transmissão.

Numa rede onde podem ocorrer perdas de pacotes deve ser satisfeito o seguinte requisito:

$$\text{Min. Response Time} \leq \text{ReceiveTMO} / 2 \geq 2 * \text{Delay} + 2,5 * \text{tempo máx. de ciclo}$$

Se este requisito for satisfeito, a perda de ao menos um pacote de dados pode ser amortecida sem interromper a conexão **safeethernet**.

---

i

Se este requisito não for satisfeito, a disponibilidade de uma conexão **safeethernet** apenas pode ser garantida numa rede sem colisões ou interferências. Porém, isso não significa um problema de segurança para o módulo processador!

---



---

i

Garantir que o sistema de comunicação respeite o Response Time parametrizado!

Se isso nem sempre pode ser garantido, há uma variável de sistema da conexão à disposição para a supervisão do Response-Time. Se o Response-Time medido é ultrapassado não apenas em casos raros pela metade do ReceiveTMO, então, o Response-Time parametrizado deve ser aumentado.

O Receive Timeout deve ser adaptado ao novo Response Time parametrizado.

---

#### 4.6.5 Sync/Async

Sync Não suportado no momento.

Async É o ajuste padrão.

Com o ajuste *Async* a instância do protocolo **safeethernet** recebe na fase de entrada da CPU e envia de acordo com as suas regras de envio na fase de saída da CPU.

#### 4.6.6 ResendTMO

*ResendTMO não pode ser configurado manualmente, mas é calculado com base no perfil e no ResponseTime.*

Tempo de supervisão em milissegundos (ms) no PES1 dentro do qual PES2 deve ter confirmado a recepção de um pacote de dados, outrossim, o pacote de dados é repetido.

Regra:  $\text{ResendTMO} \leq \text{Receive-Timeout}$

No caso de configuração diferente do *ResendTMO* nos parceiros de comunicação, o parceiro ativo do protocolo (SRS menor) determina o valor efetivo do *ResendTMO* da conexão de protocolo.

#### 4.6.7 Acknowledge Timeout

*AckTMO não pode ser configurado manualmente, mas é calculado com base no perfil e no ResponseTime.*

*AckTMO* é o período de tempo dentro do qual a CPU deve confirmar a recepção de um pacote de dados.

Para uma rede rápida, o *AckTMO* é zero, ou seja, a recepção de um pacote de dados é confirmada imediatamente. Para uma rede lenta (p. ex., trajeto por modem por linha telefônica), o *AckTMO* é maior do que zero. Neste caso tenta-se transmitir a mensagem de confirmação junto com dados de processo para reduzir a carga de rede, evitando a incidência de blocos de endereços e de segurança.

Regras:

- *AckTMO* deve ser  $\leq$  *ReceiveTMO*
- *AckTMO* deve ser  $\leq$  *ResendTMO out*, se *Production-Rate* > *ResendTMO*.

#### 4.6.8 Production Rate

*ProdRate* não pode ser configurado manualmente, mas é calculado com base no perfil e no *ResponseTime*.

Menor intervalo de tempo em milissegundos (ms) entre dois pacotes de dados.

O objetivo do *ProdRate* é limitar a quantidade de pacotes de dados a uma dimensão que não sobrecarregue um canal de comunicação (lento). Assim, obtém-se uma carga uniforme no meio de transmissão e a recepção de dados caducados do lado do receptor é evitada.

Regras:

- *ProdRate*  $\leq$  *ReceiveTimeout*
- *ProdRate*  $\leq$  *Resend-Timeout*, se *Acknowledge-Timeout* > *Resend-Timeout*

**i**

Uma *Production Rate* de zero significa que a cada ciclo do programa de aplicação é possível transmitir pacotes de dados.

#### 4.6.9 Queue

*Queue* não pode ser configurado manualmente, mas é calculado com base no perfil e no *ResponseTime*.

*Queue* (profundidade *Queue*) é a quantidade de pacotes de dados que podem ser enviados sem precisar aguardar a sua confirmação de recepção.

O valor depende da capacidade de transmissão da rede e dos possíveis atrasos causados pelo tempo do trajeto da rede.

Todas as conexões **safeethernet** dividem entre si a memória de mensagens disponível na CPU.

### 4.7 Tempo máximo de reação para safeethernet

Nos seguintes exemplos as fórmulas para o cálculo do tempo máximo de reação no caso de uma conexão com sistemas de comando HIMatrix apenas valem se nos mesmos estiver ajustado o tempo de segurança = 2 \* tempo de Watchdog. Para sistemas de comando HIMax, estas fórmulas valem sempre.

**i**

O tempo máximo de reação admissível depende do processo e deve ser autorizado pela respectiva instituição de certificação.

Conceitos:

**ReceiveTMO:** Tempo de supervisão no PES 1, dentro do qual deve ser recebido uma resposta válida do PES 2. Caso contrário, a comunicação

	direcionada à segurança é encerrada depois de esgotar este tempo.
Production Rate:	Distância mínima entre duas missivas de dados.
Watchdog Time:	Duração máxima permitida de um ciclo de RUN num sistema de comando. A duração do ciclo de RUN depende da complexidade do programa de aplicação e do número de conexões <b>safeethernet</b> . O tempo de Watchdog (WDT) deve ser introduzido nas características do recurso.
Worst Case Reaction Time:	Tempo máximo de reação para a transmissão da alteração do sinal de uma entrada física (In) de uma PES 1 até a alteração da saída física (Out) de um PES 2.
Delay:	Retardos de um trajeto de transmissão, p. ex., no caso de conexões por modem ou satélite. No caso de conexões diretas pode ser assumido inicialmente um retardo de 2 ms. O retardo real do trajeto de transmissão pode ser medido pelo administrador responsável da rede.

Para os seguintes cálculos dos tempos máximos de reação admissíveis valem as seguintes condições:

- Os sinais que são transmitidos via **safeethernet** devem ser processados nos respectivos sistemas de comando dentro de um ciclo de CPU.
- Os tempos de reação dos sensores e atuadores devem ser somados adicionalmente.

Os cálculos valem também para sinais na direção inversa.

#### 4.7.1 Cálculo do tempo máximo de reação de dois sistemas de comando HIMax

Calcular o tempo máximo de reação  $T_R$  ("Worst Case") da mudança de estado de um transdutor do sistema de comando 1 (In) até a reação da saída (Out) do sistema de comando 2:

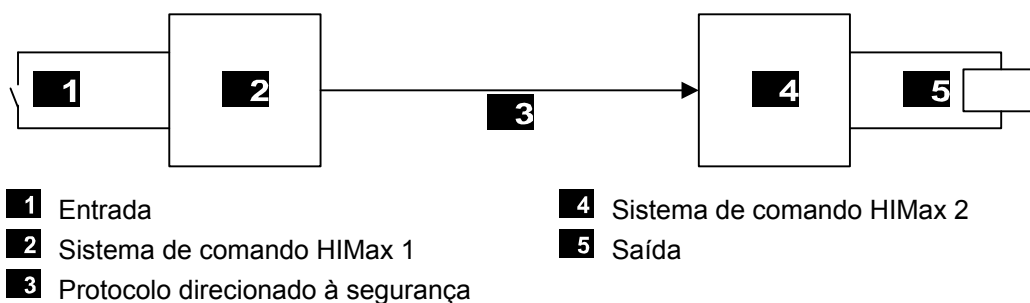


Figura 8: Tempo de reação ao conectar dois sistemas de comando HIMax

$$T_R = t_1 + t_2 + t_3$$

$T_R$  Worst Case Reaction Time

$t_1$  Tempo de segurança do sistema de comando HIMax 1

$t_2$  *ReceiveTMO*

$t_3$  Tempo de segurança do sistema de comando HIMax 2

#### 4.7.2 Cálculo do tempo máximo de reação em conexão com um sistema de comando HIMatrix

Calcular o tempo máximo de reação  $T_R$  ("Worst Case") da mudança de estado de um transdutor (In) do sistema de comando HIMax até a reação da saída (Out) do sistema de comando HIMatrix:

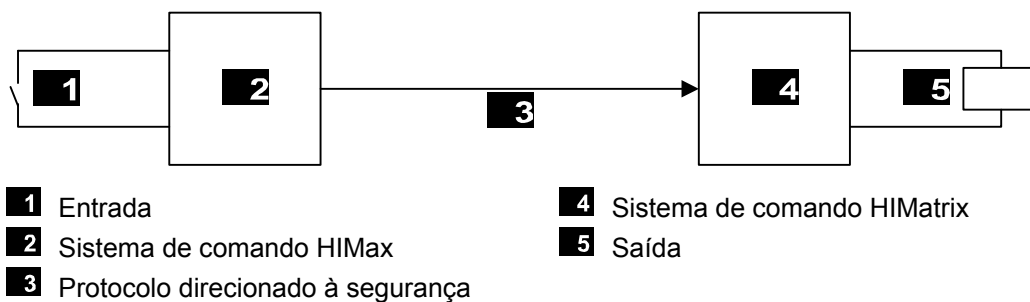


Figura 9: Tempo de reação na conexão de um sistema de comando HIMax com um sistema de comando HIMatrix

$$T_R = t_1 + t_2 + t_3$$

$T_R$  Worst Case Reaction Time

$t_1$  Tempo de segurança do sistema de comando HIMax

$t_2$  *ReceiveTMO*

$t_3$  2 \* tempo de Watchdog do sistema de comando HIMatrix

#### 4.7.3 Cálculo do tempo máximo de reação com dois sistemas de comando HIMatrix ou Remote I/Os

Calcular o tempo máximo de reação  $T_R$  ("Worst Case") da mudança de estado de um transdutor (In) no primeiro sistema de comando HIMatrix ou na Remote I/O (p. ex., F3 DIO 20/8 01) até a reação da saída no segundo sistema de comando HIMatrix ou na Remote I/O (Out):

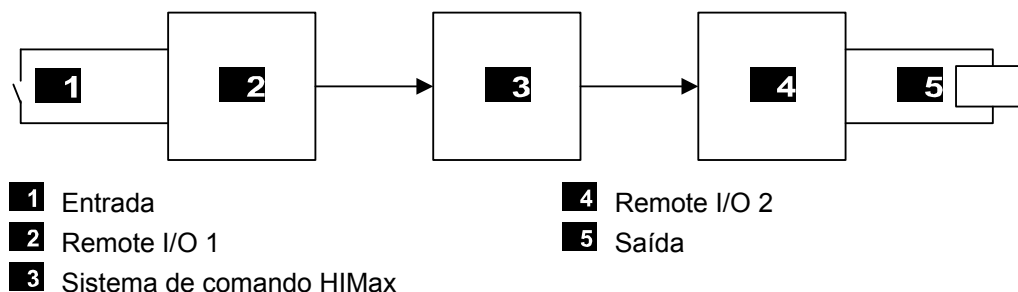


Figura 10: Tempo de reação com dois Remote I/Os e um sistema de comando HIMax

$$T_R = t_1 + t_2 + t_3 + t_4 + t_5$$

$T_R$  Worst Case Reaction Time

$t_1$  2 \* tempo de watchdog do Remote I/O 1

$t_2$  *ReceiveTMO1*

$t_3$  2 \* tempo de Watchdog do sistema de comando HIMax

$t_4$  *ReceiveTMO2*

$t_5$  2 \* tempo de watchdog do Remote I/O 2

**i**

Os dois Remote I/Os 1 e 2 também podem ser idênticos. Os tempos também valem se ao invés de uma Remote I/O um sistema de comando HIMatrix for utilizado.

#### 4.7.4 Cálculo do tempo máximo de reação com dois sistemas de comando HIMax e um sistema de comando HIMatrix

Calcular o tempo máximo de reação  $T_R$  ("Worst Case") da mudança de estado de um transdutor (In) do primeiro sistema de comando HIMax até a reação da saída (Out) do segundo sistema de comando HIMax:

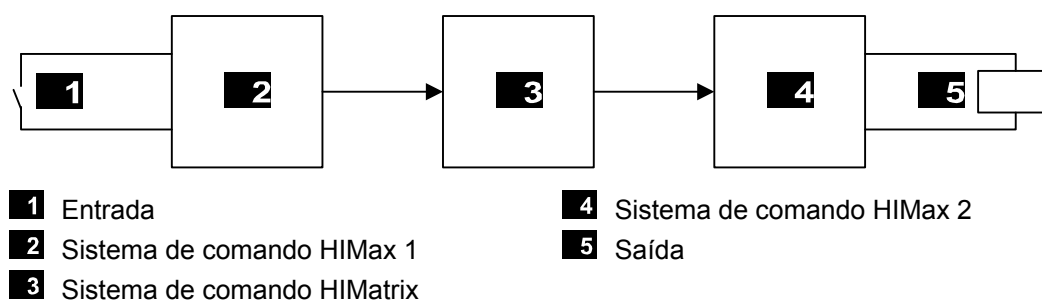


Figura 11: Tempo de reação com dois sistemas de comando HIMax e um sistema de comando HIMatrix



$$T_R = t_1 + t_2 + t_3 + t_4 + t_5$$

$T_R$  Worst Case Reaction Time

$t_1$  Tempo de segurança do sistema de comando HIMax 1

$t_2$  *ReceiveTMO1*

$t_3$  2 \* tempo de Watchdog do sistema de comando HIMatrix

$t_4$  *ReceiveTMO2*

$t_5$  Tempo de segurança do sistema de comando HIMax 2

**i**

Os dois sistema de comando HIMax 1 e 2 também podem ser idênticos.

O sistema de comando HIMatrix também pode ser um sistema de comando HIMax.

#### 4.7.5 Cálculo do tempo máximo de reação de dois sistemas de comando HIMatrix

O tempo máximo de reação  $T_R$  ("Worst Case") da mudança de estado de um transdutor do sistema de comando 1 até a reação da saída do sistema de comando 2 pode ser calculado como segue:

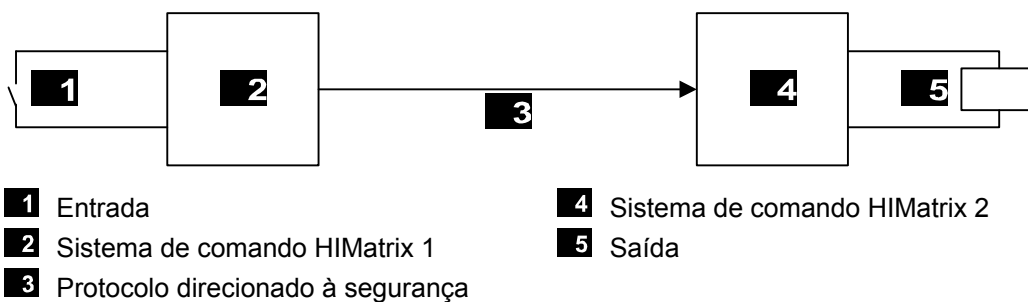


Figura 12: Tempo de reação ao conectar dois sistemas de comando HIMatrix

$$T_R = t_1 + t_2 + t_3$$

$T_R$  Worst Case Reaction Time

$t_1$  2 \* tempo de Watchdog do sistema de comando HIMatrix 1

$t_2$  *ReceiveTMO*

$t_3$  2 \* tempo de Watchdog do sistema de comando HIMatrix 2

#### 4.7.6 Cálculo do tempo máximo de reação com dois Remote I/Os

O tempo máximo de reação  $T_R$  ("Worst Case") da mudança de estado de um transdutor (In) do primeiro sistema de comando HIMatrix ou Remote I/O (p. ex., F3 DIO 20/8 01) até a reação da saída do segundo sistema de comando HIMatrix ou Remote I/O (Out) pode ser calculado como segue:

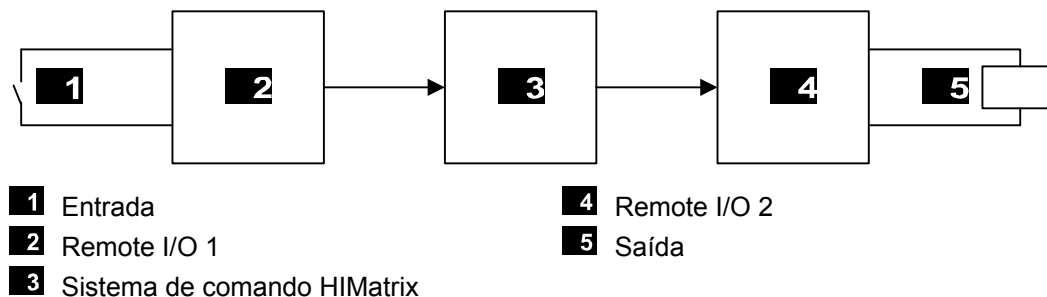


Figura 13: Tempo de reação com Remote I/Os

$$T_R = t_1 + t_2 + t_3 + t_4 + t_5$$

$T_R$  Worst Case Reaction Time

$t_1$  2 \* tempo de watchdog do Remote I/O 1

$t_2$  ReceiveTMO<sub>1</sub>

$t_3$  2 \* tempo de Watchdog do sistema de comando HIMatrix

$t_4$  ReceiveTMO<sub>2</sub>

$t_5$  2 \* tempo de watchdog do Remote I/O 2

Observação: Os dois Remote I/Os 1 e 2 também podem ser idênticos. Os tempos também valem se ao invés de uma Remote I/O um sistema de comando HIMatrix for utilizado.

#### 4.7.7 Cálculo do tempo máximo de reação, dois sistemas de comando HIMatrix, um sistema de comando HIMax

O tempo máximo de reação  $T_R$  ("Worst Case") da mudança de estado de um transdutor (In) do primeiro sistema de comando HIMatrix até a reação da saída do segundo sistema de comando HIMatrix (Out) pode ser calculado como segue:

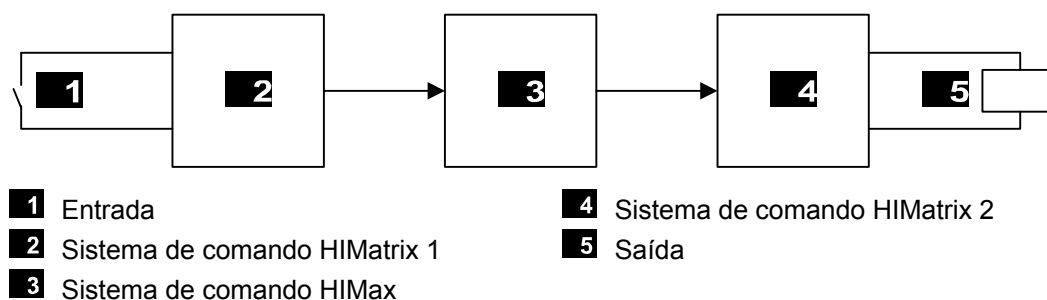


Figura 14: Tempo de reação com dois sistemas de comando HIMatrix e um sistema de comando HIMax

$$T_R = t_1 + t_2 + t_3 + t_4 + t_5$$

$T_R$  Worst Case Reaction Time

$t_1$  2 \* tempo de Watchdog do sistema de comando HIMatrix 1

$t_2$  ReceiveTMO<sub>1</sub>

$t_3$  2 \* tempo de Watchdog do sistema de comando HIMax

$t_4$  ReceiveTMO<sub>2</sub>

$t_5$  2 \* tempo de Watchdog do sistema de comando HIMatrix 2

#### 4.7.8 Perfis safeethernet

Perfis **safeethernet** são combinações de parâmetros correspondentes compatíveis que podem ser ajustados automaticamente ao selecionar um dos perfis **safeethernet**. Para a parametrização neste caso apenas precisa ser configurado individualmente o Receive-Timeout e o Response-Time previsto.

O objetivo do perfil **safeethernet** é otimizar o fluxo de dados na rede considerando as características físicas.

Condição para a eficácia da otimização são os seguintes requisitos:

- A fatia de tempo de comunicação é suficientemente grande para processar em um ciclo de CPU todas as conexões safeethernet.
- Se o tempo médio de ciclo de CPU < Response Time.
- Se o tempo médio de ciclo de CPU < ProdRate o ProdRate = 0.

#### NOTA



**Avárias da comunicação safeethernet até a falha total são possíveis!**

**Combinações não compatíveis de ciclo de CPU, fatia de tempo para a comunicação, Response Time e ProdRate não são recusados ao gerar o código e no Download/Reload, porém podem levar a interferências na comunicação.**

**Verificar nos Control Panels dos dois sistemas de comando a visualização *bad messages e resends*.**

Seis perfis **safeethernet** estão à disposição dos quais pode ser escolhido o perfil **safeethernet** adequado para o trajeto de transmissão.

Para a comunicação de dados de processo direcionada à segurança apenas podem ser usados os perfis *Fast&Noisy*, *Medium&Noisy* e *Slow&Noisy*.

Fast & Cleanroom	Não adequado para a comunicação de dados de processo direcionada à segurança!
Fast & Noisy	
Medium & Cleanroom	Não adequado para a comunicação de dados de processo direcionada à segurança!
Medium & Noisy	
Slow & Cleanroom	Não adequado para a comunicação de dados de processo direcionada à segurança!
Slow & Noisy	

## 4.7.9 Perfil I (Fast &amp; Cleanroom)

**NOTA**

**Não adequado para a comunicação de dados de processo direcionada à segurança!**

- Para a comunicação de dados de processo direcionada à segurança apenas podem ser usados os perfis *Fast&Noisy*, *Medium&Noisy* e *Slow&Noisy*. Oferecido por causa da compatibilidade a versões mais antigas de SILworX e ELOP II Factory.

## Utilização

O perfil *Fast & Cleanroom* é adequado para aplicações em ambientes ideais, p. ex., laboratórios!

- Para a máxima velocidade na transmissão do volume de dados
- Para aplicações que exigem transmissão rápida de dados
- Para aplicações que exigem o Worst Case ReactionTime menor possível

## Requisitos de rede

- Fast: tecnologia 100-Mbit (100 Base TX), tecnologia 1-Gbit
- Clean: rede sem avarias.  
Perda de dados por sobrecarga da rede, interferências externas ou manipulações da rede devem ser evitadas.
- LAN Switches são necessários!

## Características do caminho de comunicação

- Atrasos mínimos
- ResponseTime esperado  $\leq$  ReceiveTMO  
(outrossim, ERRO na parametrização)

## 4.7.10 Perfil II (Fast &amp; Noisy)

## Utilização

O perfil *Fast & Noisy* é o perfil padrão do SILworX para a comunicação via safeethernet.

- Para boa velocidade na transmissão do volume de dados
- Para aplicações que exigem transmissão rápida de dados
- Para aplicações que exigem um Worst Case ReactionTime tão baixo quanto possível

## Requisitos de rede

- Fast: tecnologia 100-Mbit (100 Base TX), tecnologia 1-Gbit
- Noisy: a rede não é livre de avarias.  
Baixa probabilidade para a perda de pacotes de dados  
Tempo para  $\geq 1$  repetição
- LAN Switches são necessários!

## Características do caminho de comunicação

- Atrasos mínimos
- ResponseTime esperado  $\leq$  ReceiveTMO / 2  
(outrossim, ERRO na parametrização)

## 4.7.11 Perfil III (Medium &amp; Cleanroom)

**NOTA**

**Não adequado para a comunicação de dados de processo direcionada à segurança!**

- Para a comunicação de dados de processo direcionada à segurança apenas podem ser usados os perfis *Fast&Noisy*, *Medium&Noisy* e *Slow&Noisy*. Oferecido por causa da compatibilidade a versões mais antigas de SILworX e ELOP II Factory.

**Utilização**

O perfil *Medium & Cleanroom* serve para aplicações que exigem transmissão razoavelmente rápida de dados numa rede livre de avarias.

- Para velocidade média da transmissão do volume de dados
- Adequado para Virtual Private Networks (VPN), onde a troca de dados é lenta devido a dispositivos de segurança intermediários (Firewalls, encriptação), porém livre de erros.
- Adequado para aplicações onde o Worst Case ReactionTime não é um fator crítico.

**Requisitos de rede**

- Medium: tecnologia 10-Mbit (10 Base T), 100-Mbit (100 Base TX), 1-Gbit
- LAN Switches são necessários!
- Clean: rede sem avarias.  
Perda de dados por sobrecarga da rede, interferências externas ou manipulações da rede devem ser evitadas.  
Tempo para  $\geq 0$  repetições.

**Características do caminho de comunicação**

- Atrasos moderados
- ResponseTime esperado  $\leq$  ReceiveTMO (outrossim, ERRO na parametrização)

## 4.7.12 Perfil IV (Medium &amp; Noisy)

**Utilização**

O perfil *Medium & Noisy* serve para aplicações que exigem transmissão razoavelmente rápida de dados.

- Para velocidade média da transmissão do volume de dados
- Para aplicações que exigem apenas transmissão razoavelmente rápida de dados
- Adequado para aplicações onde o Worst Case ReactionTime não é um fator crítico

**Requisitos de rede**

- Medium: tecnologia 10-Mbit (10 Base T), 100-Mbit (100 Base TX), 1-Gbit
- LAN Switches são necessários!
- Noisy: a rede não é livre de avarias.  
Baixa probabilidade para a perda de pacotes de dados  
Tempo para  $\geq 1$  repetição

**Características do caminho de comunicação**

- Atrasos moderados
- ResponseTime esperado  $\leq$  ReceiveTMO / 2 (outrossim, ERRO na parametrização)

## 4.7.13 Perfil V (Slow &amp; Cleanroom)

**NOTA**

**Não adequado para a comunicação de dados de processo direcionada à segurança!**

- Para a comunicação de dados de processo direcionada à segurança apenas podem ser usados os perfis *Fast&Noisy*, *Medium&Noisy* e *Slow&Noisy*. Oferecido por causa da compatibilidade a versões mais antigas de SILworX e ELOP II Factory.

## Utilização

O perfil *Slow & Cleanroom* serve para aplicações que exigem apenas transmissão lenta de dados, numa rede livre de avarias.

- Para velocidade lenta da transmissão do volume de dados.
- Para aplicações que apenas exigem uma transmissão de dados lenta a sistemas de comando (provavelmente muito distantes) e em situações onde as condições do trajeto de comunicação não podem ser previstas.

## Requisitos de rede

- Slow: transferência de dados via ISDN, linha dedicada ou conexão via feixes hertzianos.
- Clean: rede livre de avarias.  
Perda de dados por sobrecarga da rede, interferências externas ou manipulações da rede devem ser evitadas.  
Tempo para  $\geq 0$  repetições.

## Características do caminho de comunicação

- Atrasos moderados
- ResponseTime esperado = ReceiveTMO (outrossim, ERRO na parametrização)

## 4.7.14 Perfil VI (Slow &amp; Noisy)

## Utilização

O perfil *Slow & Noisy* é adequado para aplicações que apenas exigem uma transmissão de dados lenta a sistemas de comando (provavelmente muito distantes).

- Para velocidade lenta da transmissão do volume de dados
- Para aplicações onde predomina a transferência de dados por linhas telefônicas precárias ou trajetos de feixes hertzianos com interferências.

## Requisitos de rede

- Slow: transferência de dados via telefone, satélite, rádio, etc.
- Noisy: a rede não é livre de avarias.  
Baixa probabilidade para a perda de pacotes de dados  
Tempo para  $\geq 1$  repetição

## Características do caminho de comunicação

- Atrasos moderados a longos
- ResponseTime esperado  $\leq$  ReceiveTMO / 2 (outrossim, ERRO na parametrização)

## 4.8 Comunicação entre projetos

A comunicação entre diferentes projetos é usada para

- Conectar recursos de diversos projetos entre si.
- Conectar entre si sistemas de comando com sistema operacional a partir da V7 (para SILworX) e sistemas de comando com sistema operacional anterior a V7 (para ELOP II Factory) via **safeethernet**, veja Capítulo 4.9.

A comunicação entre os dois projetos ocorre via **safeethernet** e é configurada no Editor **safeethernet**.

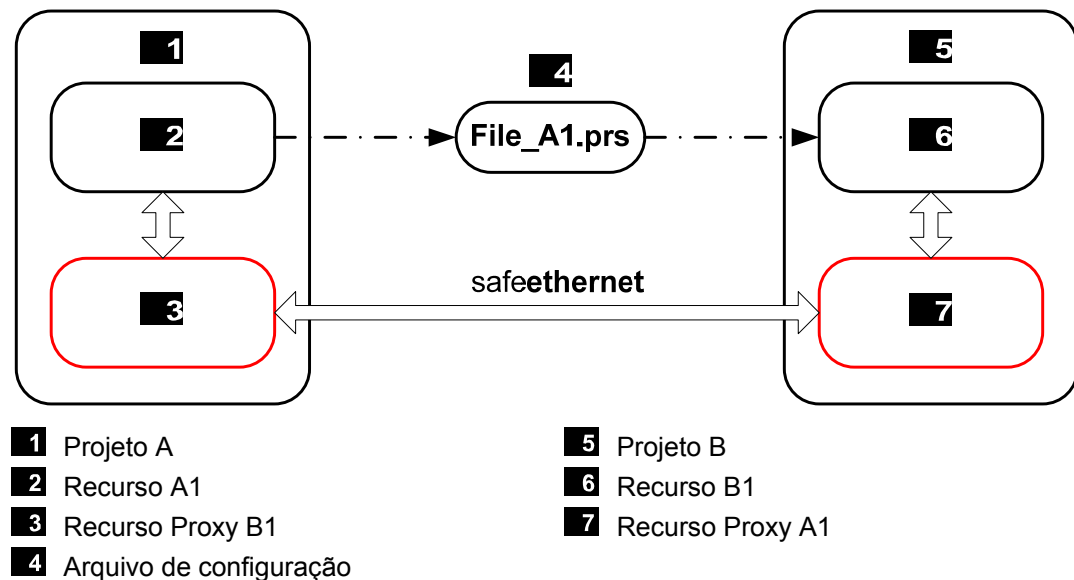


Figura 15: Conexão **safeethernet** entre recurso A1 no projeto A e recurso B1 no projeto B

É denominado projeto local o projeto onde a configuração da conexão **safeethernet** é efetuada e onde o arquivo de configuração é criado.

É denominado projeto de destino o projeto onde o arquivo de configuração é importado.

Na troca de dados, o projeto local e o projeto de destino são parceiros de comunicação com direitos iguais.

O respectivo recurso Proxy serve como curinga para o recurso correspondente do projeto externo e é usado para a importação e exportação das conexões **safeethernet**.

O *Proxy Resource B1* em *Project A* é o curinga do *Resource B1* do *Project B*.

O *Proxy Resource A1* no *Project B* é o curinga do *Resource A1* do *Project A*.

No projeto local (aqui *Project A*), o recurso Proxy (aqui *Proxy Resource B1*) deve ser criado e configurado manualmente. Depois da configuração, importar o arquivo de configuração (aqui *File\_A1.prs*) no projeto de destino (aqui de *Resource B1*).

O arquivo de configuração *File\_A1.prs* contém a descrição completa de *Resource A1* para a conexão **safeethernet** com *Resource B1*. Depois de importar o arquivo de configuração *File\_A1.prs* para o *Resource B1*, o *Proxy Resource A1* é criado automaticamente no *Project B*.

### 4.8.1 Variantes para a comunicação entre projetos

Nas seguintes duas variantes, os projetos A e B se comunicam via **safeethernet**.

Neste caso, na primeira variante, o projeto A é o projeto local e na segunda variante, o projeto B é o projeto local. A princípio, é uma escolha do usuário em qual dos dois projetos prefere criar a configuração.

O esforço necessário para ambos os caminhos de configuração é quase igual e leva à mesma configuração final.

#### Projeto local A

Configurar no projeto local A a comunicação para o projeto de destino B e criar os arquivos de configuração. Isso possui a vantagem que apenas o recurso Proxy B1 no projeto local precisa ser criado manualmente.

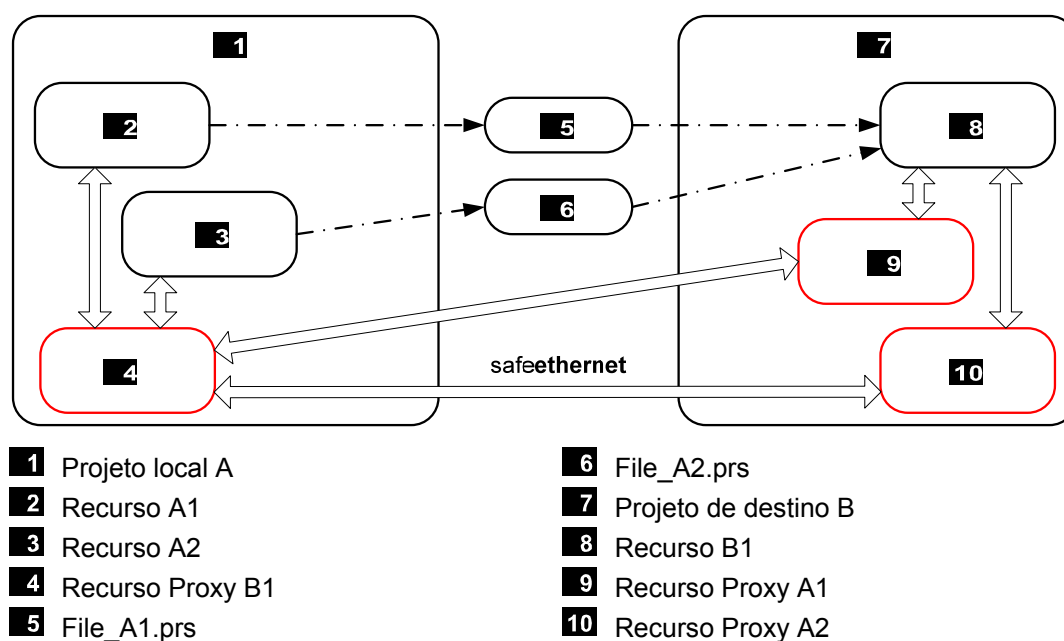


Figura 16: Variante Projeto A como projeto local



### Projeto local B

Configurar no projeto local B a comunicação para o projeto de destino A e criar os arquivos de configuração. Isso possui a desvantagem que dois recursos Proxy (A1 e A2) precisam ser criados manualmente no projeto local B.

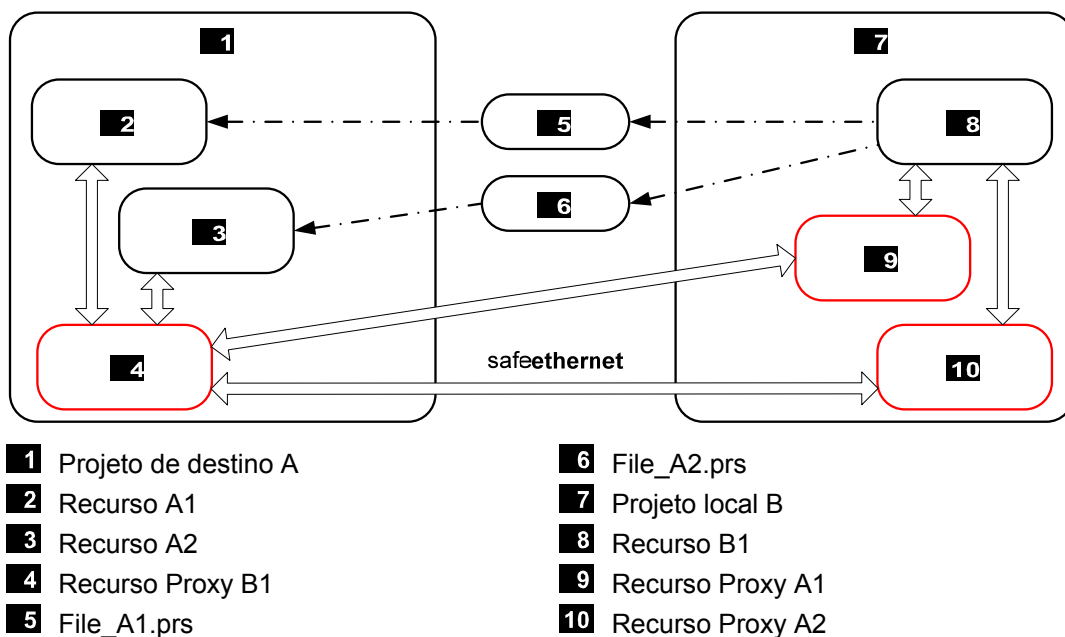


Figura 17: Variante Projeto B como projeto local

## 4.9 Comunicação entre projetos com SILworX e ELOP II Factory

Neste exemplo, é configurada uma conexão **safeethernet** entre HIMax (SILworX) e HIMatrix (ELOP II Factory).

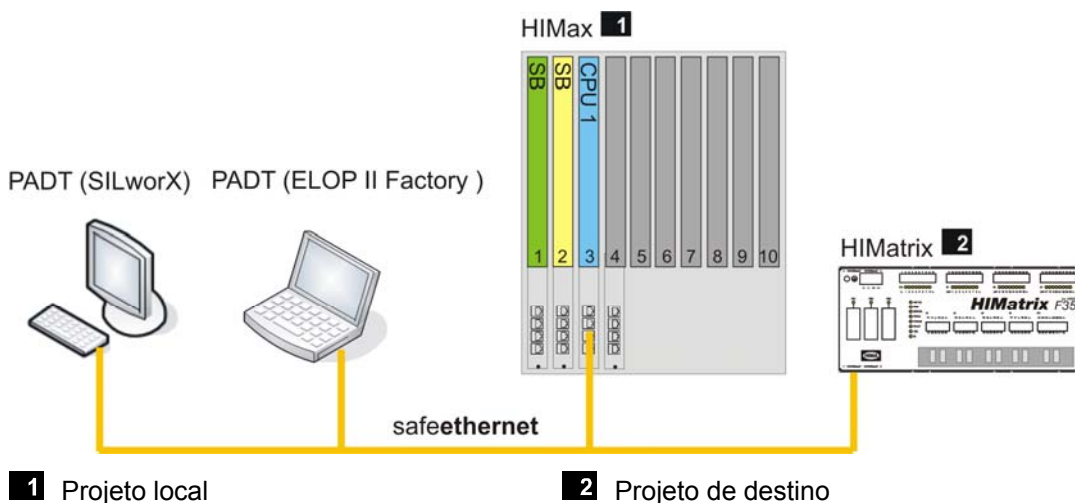


Figura 18: Comunicação entre projetos com SILworX e ELOP II Factory

Abrir o recurso do projeto de destino (HIMatrix) que deve servir como recurso Proxy no projeto local (HIMax).

Determinar os seguintes parâmetros para este recurso de destino:

- ID de sistema
- Safety Time [ms]
- Tempo de watchdog [ms]
- Endereço IP

**i**

Estas características do recurso são relevantes para a segurança e estão sujeitos a restrições. Indicações a este respeito encontram-se nos manuais de segurança dos respectivos sistemas de comando.

### 4.9.1 Configuração do HIMax no projeto SILworX

#### 4.9.1.1 Criar o recurso Proxy

Um recurso Proxy serve como curinga para o recurso correspondente num projeto externo e é usado para a importação e exportação das conexões **safeethernet**.

**Assim é criado um recurso Proxy no projeto local:**

1. Abrir o projeto local onde o recurso Proxy deve ser criado.
2. Abrir na árvore de estrutura **Configuration**.
3. Clicar com o botão direito em **Configuration** e seleccionar **New, Proxy Resource, ELOP II Factory**.
  - ☒ Um novo recurso Proxy é adicionado.

**Assim é configurado um recurso Proxy no projeto local:**

1. Selecionar no menu de contexto do recurso Proxy o item **Properties**.
2. Introduzir no campo **Name** nomes únicos.  
Usar para o recurso Proxy no projeto local o nome do recurso no projeto de destino.
3. Introduzir os valores de **System-ID**, **Safety Time [ms]** e **Watchdog Time [ms]** anteriormente determinados para este recurso Proxy.
4. Clicar em **OK**. Os outros parâmetros podem continuar com os valores padrão.

**Abrir a estrutura do recurso Proxy:**

1. Clicar com o botão direito em **Hardware** e **Edit**, selecionar **HIMatrix Proxy**.
2. Confirmar com **OK** para abrir o Hardware Editor do recurso Proxy.

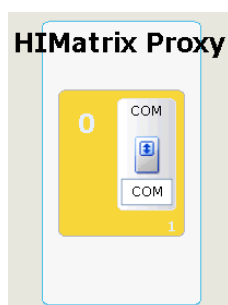


Figura 19: Recurso Proxy HIMatrix

3. Clicar duas vezes em **COM-Module** e introduzir o **IP Address** anteriormente determinado para o recurso Proxy.
4. Clicar em **Save** e depois em **Close**.
5. Repetir estes passos para todos os outros recursos Proxy no projeto local.

**4.9.1.2 Conectar o recurso local com o recurso Proxy**

Criar no Editor **safeethernet** uma conexão **safeethernet** entre o recurso local e o recurso Proxy.

**Assim abre-se o Editor de safeethernet do recurso local:**

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **safeethernet** e selecionar no menu de contexto **Edit**.  
Na seleção de objetos encontra-se o recurso Proxy criado.

**Assim cria-se a conexão safeethernet ao recurso Proxy:**

1. Na seleção de objetos, clicar no **Proxy Resource** e puxar mediante Drag&Drop para um local livre na área de trabalho do **safeethernet** Editor.
2. Selecionar as interfaces **Ethernet** do recurso local e do recurso Proxy.  
Os seguintes parâmetros determinam o fluxo de dados e a tolerância a erros e colisões da conexão **safeethernet**:
4. Selecionar **Network Profile** (p. ex., Fast & Noisy) da conexão **safeethernet**.
5. Calcular o **Receive Timeout** e **Response Time** e introduzir.

Exemplo para os valores dos parâmetros de uma conexão safeethernet ao recurso Proxy:

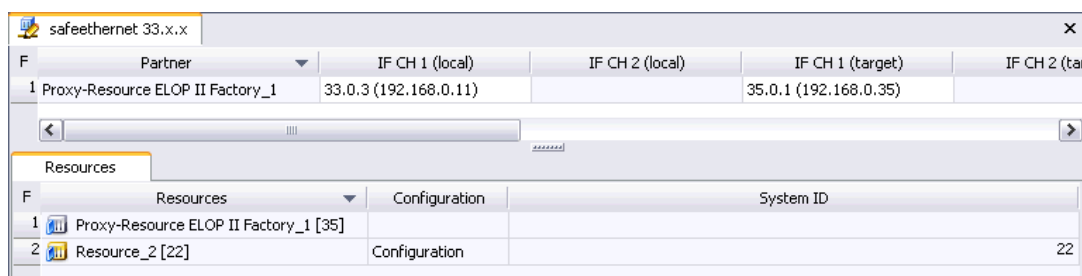


Figura 20: Parâmetros de uma conexão safeethernet ao recurso Proxy

#### 4.9.1.3 Conectar variáveis de processo

Conectar as variáveis de processo na visualização de detalhes da conexão safeethernet.

##### Assim se abre a visualização de detalhes duma conexão safeethernet:

Requisito: O Editor safeethernet do recurso local deve estar aberto.

1. Clicar com o botão direito na linha **Proxy Resource** e abrir o recurso Proxy.
2. Selecionar no menu de contexto **Detail View** para abrir a visualização de detalhes da conexão safeethernet.
3. Selecionar o registro **Resource <-> Proxy-Resource**.

##### Assim se adicionam as variáveis de envio safeethernet:

As variáveis de envio são enviadas do recurso local ao recurso Proxy.

1. Selecionar a área **Resource -> Proxy-Resource**.
2. Na seleção de objetos, selecionar uma **Global Variable** e puxar via Drag&Drop para a coluna **Resource (Target) -> Resource (local)**.
3. Repetir este passo para outras variáveis de envio safeethernet.

##### Assim se adicionam as variáveis de recepção safeethernet:

As variáveis de recepção são recebidas do recurso local.

1. Selecionar a área **Resource <- Proxy-Resource**.
2. Na seleção de objetos, selecionar uma **Global Variable** e puxar via Drag&Drop para a coluna **Resource (Target) <- Resource (local)**.
3. Repetir este passo para outras variáveis de recepção safeethernet.

#### 4.9.1.4 Exportar o arquivo de configuração do SILworX

A conexão **safeethernet** configurada no SILworX deve ser exportada como arquivo de configuração com a extensão **\*.prs**. Este arquivo de configuração agora pode ser importado no ELOP II Factory para transferir a conexão **safeethernet** para o sistema de comando HIMatrix.

##### Assim exporta-se uma conexão **safeethernet**:

1. Clicar no **safeethernet** Editor em **Proxy Resource** e abrir o menu de contexto.
2. Selecionar no menu de contexto **Export Connection with Proxy Resource**:  
Abre-se uma janela de diálogo padrão para salvar um arquivo.
3. Introduzir na janela de diálogo o nome de arquivo para o arquivo de configuração e salvar o arquivo com a extensão **\*.prs**.
4. Fechar o projeto local.

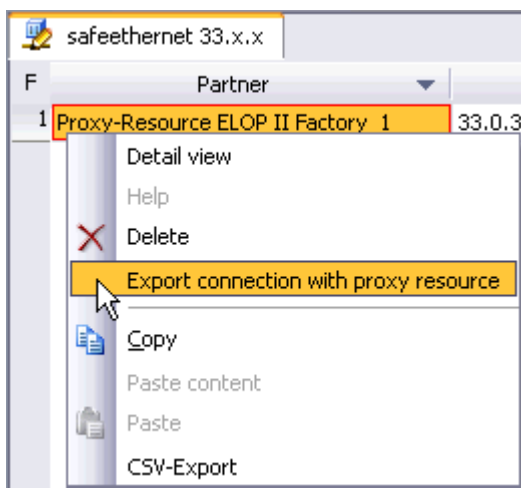


Figura 21: Exportar uma conexão **safeethernet**

##### Verificação da conexão **safeethernet**:

1. Abrir na árvore de estrutura **Configuration, Resource, safeethernet**.
2. Clicar no botão **Verification** no **Action Bar** e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log e corrigir erros onde for necessário.



A configuração da conexão **safeethernet** e o programa de aplicação do recurso HIMax devem ser novamente compilados e carregados ao sistema de comando para que se tornem efetivos para a comunicação do HIMax.

---

## 4.9.2 Configuração do HIMatrix no ELOP II Factory

**Assim importa-se o arquivo de configuração no projeto de destino (HIMatrix):**

1. Iniciar o ELOP II Factory.
2. Abrir o projeto de destino do HIMatrix para o qual o arquivo de configuração deve ser importado.
3. Selecionar na árvore de estrutura o recurso de destino e abrir o menu de contexto.
4. Selecionar no menu de contexto **Import Connections**:  
Abre-se uma janela de diálogo para carregar um arquivo com a extensão \*.prs.
5. Selecionar na janela de diálogo o arquivo de configuração que foi criado no projeto local HIMax e clicar em **OK**.  
☒ Depois de importar o arquivo de configuração, o recurso HIMax é automaticamente criado como recurso Proxy no projeto de destino HIMatrix.

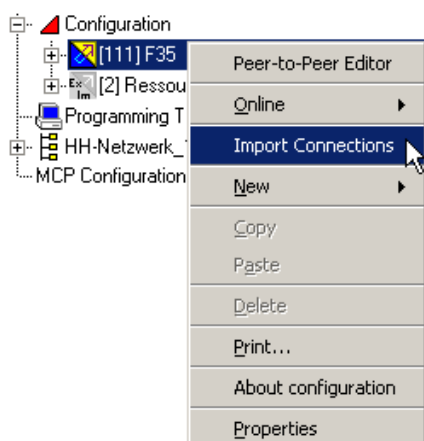


Figura 22: Importar conexões no ELOP II Factory

### 4.9.2.1 Atribuir sinais de processo do ELOP II Factory

Conectar os sinais de processo no recurso de destino (HIMatrix).

Abrir o **Signal Editor** pela barra de menus **Signals, Editor**.

**Assim abre-se o Editor Peer-to-Peer para o recurso de destino no ELOP II Factory:**

1. Abrir na árvore de estrutura **Configuration, Resource, Peer-to-Peer-Editor**.
2. Introduzir no **Peer-to-Peer-Editor** a rede HH para esta conexão.
3. Clicar no **Peer-to-Peer-Editor** em **Connect Process Signals**.

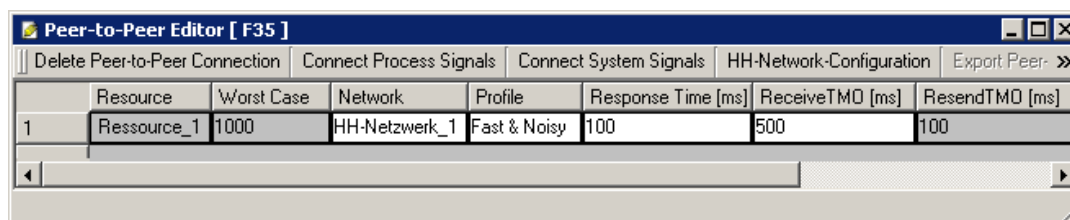


Figura 23: Peer-to-Peer-Editor no ELOP II Factory

**i**

Certificar-se de que está usando para ambos os parceiros de comunicação o mesmo perfil e os mesmos ajustes (transferido automaticamente ao importar o arquivo de configuração).

**Assim atribuem-se os sinais de envio Peer-to-Peer:**

Os sinais de envio Peer-to-Peer são enviados pelo recurso HIMatrix ao recurso HIMax.

1. Selecionar o registro **HIMatrix Resource-> HIMax Proxy-Resource**.  
O registro contém os sinais de envio Peer-to-Peer importados.
2. Clicar no Editor de sinais em um **process signal** e puxar via Drag&Drop para cima do sinal de envio na janela de diálogo sinais de processo P2P que deve ser conectado.
3. Repetir este passo para os demais sinais de envio Peer-to-Peer.

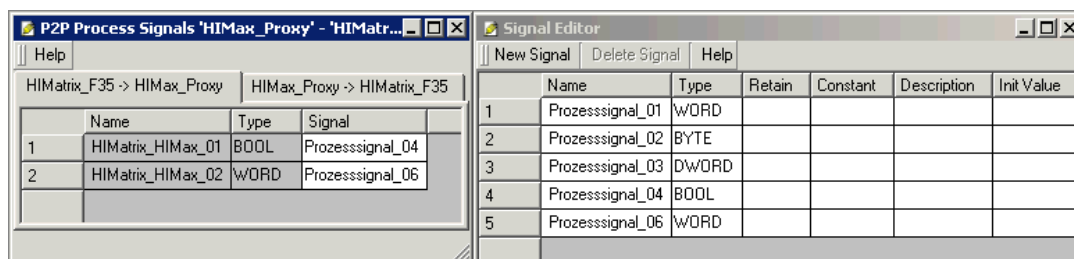


Figura 24: Atribuir sinais de envio no ELOP II Factory

**Assim atribuem-se os sinais de recepção Peer-to-Peer:**

Os sinais de recepção Peer-to-Peer são recebidos pelo recurso HIMatrix.

1. Selecionar o registro **Resource <- Proxy-Resource**.  
O registro contém os sinais de recepção Peer-to-Peer importados.
2. Clicar no Editor de sinais em um **process signal** e puxar via Drag&Drop para cima do sinal de recepção na janela de diálogo sinais de processo P2P que deve ser conectado.
3. Repetir este passo para os demais sinais de recepção Peer-to-Peer.

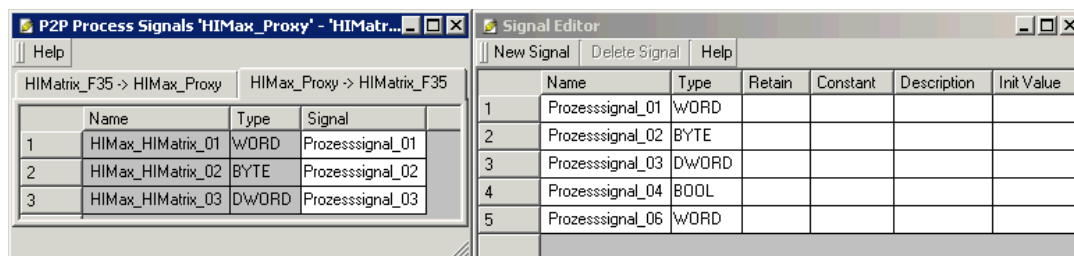


Figura 25: Atribuir sinais de recepção no ELOP II Factory



Informações mais detalhadas a respeito da conexão de sinais de processo no ELOP II Factory encontram-se na ajuda online do ELOP II Factory.



A configuração da conexão Peer-to-Peer e o programa de aplicação do recurso HIMax devem ser novamente compilados e carregados ao sistema de comando. Só então a conexão Peer-to-Peer se torna efetiva para o HIMatrix.

### 4.10 Control Panel (safeethernet)

No Control Panel, o usuário pode verificar e controlar os ajustes da conexão **safeethernet**. Além disso, informações de status atuais (p. ex., tempo de ciclo, estado do barramento, etc.) da conexão **safeethernet** são exibidas.

**Assim abre-se o Control Panel para a supervisão da conexão safeethernet:**

1. Selecionar na árvore de estrutura **Resource**.
2. Selecionar **Online** no menu de contexto do recurso.
3. Introduzir no **System Log-in** os dados de acesso para abrir do Control Panel do recurso.
4. Selecionar na árvore de estrutura do Control Panel **safeethernet**.

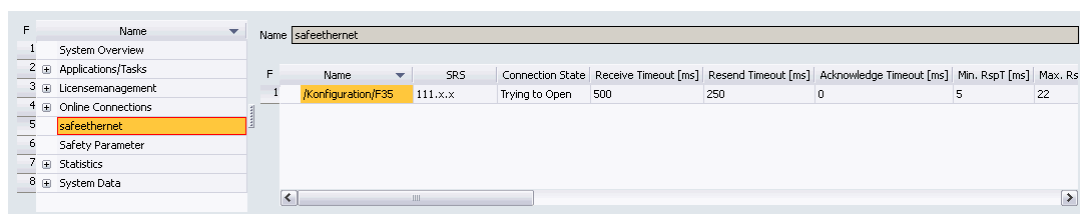


Figura 26: Control Panel para a supervisão da conexão

**Resetar valores de estatística:**

Mediante esta função do menu de contexto, é possível resetar para zero os dados estatísticos (tempo de ciclo mín, máx, etc.).

**Assim resetam-se os dados estatísticos da conexão safeethernet:**

1. Selecionar na árvore de estrutura a conexão **safeethernet**.
2. Selecionar no menu de contexto da conexão **safeethernet** o item **Reset safeethernet Statistics**.



## 4.10.1 Campo de exibição (conexão safeethernet)

No campo de exibição são mostrados os seguintes valores da conexão safeethernet selecionada:

Elemento	Descrição
Name	Nome do recurso do parceiro de comunicação
SRS	System.Rack.Slot
Connection State	Estado da conexão safeethernet (veja também Capítulo 4.4)
Receive-Timeout [ms]	(veja também Capítulo 4.6.3)
Resend-Timeout [ms]	(veja também Capítulo 4.6.6)
Acknowledge Timeout [ms]	(veja também Capítulo 4.6.7)
Min. RspT [ms]	Response-Time atual como valor mínimo, máximo, último e médio. Veja também Capítulo 4.6.4.
Max. RspT [ms]	
Last RspT [ms]	
Av. RspT [ms]	
Bad messages	Quantidade de mensagens descartadas depois de resetar a estatística.
Repeats	Quantidade de repetições depois de resetar a estatística.
Number of successful connections	Quantidade de sucessos de conexão depois de resetar a estatística.
Early Queue Usage	Quantidade de mensagens que foram colocadas na Early Queue depois de resetar a estatística. Veja também Capítulo 4.6.9.
Frame No.	Contador de envio circular
Ack. Frame No.	Contador de recepção circular
Monotony	Contador de envio de dados de trabalho circular
Layout version	Assinatura do ponto final atual da comunicação
New layout version	Assinatura do novo ponto final atual da comunicação
Connection Control	Status do controle da conexão. Veja também Capítulo 4.4.
Transmission Control Ch 1	Liberação do caminho de transporte canal 1. Veja também Capítulo 4.4.
Transmission Control Ch 2	Liberação do caminho de transporte canal 2. Veja também Capítulo 4.4.
Quality of channel 1	Estado do caminho de transporte canal 1. Veja também Capítulo 4.4.
Quality of channel 2	Estado do caminho de transporte canal 2. Veja também Capítulo 4.4.
Late Received Redundant Messages	No caso de caminhos de transporte redundantes. Quantidade de mensagens recebidas tardiamente depois de resetar a estatística.
Lost redundant messages	No caso de caminhos de transporte redundantes. Quantidade de mensagens apenas recebidas em um dos dois caminhos de transporte depois de resetar a estatística.
Protocol Version	0–1 Versão antiga do protocolo para HIMatrix com sistema operacional de CPU < V7 2 Versão nova do protocolo para HIMax e HIMatrix com sistema operacional de CPU a partir de V7

Tabela 37: Campo de exibição da conexão safeethernet

## 5 PROFINET-IO

PROFINET-IO é o protocolo de transmissão baseado na tecnologia Ethernet da organização de usuários de automação PROFIBUS.

Como no caso do PROFIBUS-DP, os equipamentos de campo decentrais no caso do PROFINET-IO são integrados no SILworX via uma descrição do equipamento (arquivo GSDML).

O controlador PROFINET-IO da HIMA corresponde ao nível Conformance Class A e suporta a comunicação Non-Real-Time (NRT) e Real Time (RT) aos dispositivos PROFINET-IO. Nesta caso, a comunicação Real-Time é utilizada para a troca de dados crítica em relação ao tempo e a comunicação Non-Real-Time, para processos não críticos em relação ao tempo (leitura/escrita acíclica).

Uma conexão redundante PROFINET-IO apenas pode ser acessada mediante a configuração de um segundo controlador/dispositivo PROFINET-IO e adaptações no programa de aplicação.

### 5.1 Blocos funcionais PROFINET-IO

Para a troca de dados acíclica, o SILworX disponibiliza os mesmos blocos funcionais, funcionalmente iguais, como no PROFIBUS-DP.

Os seguintes blocos funcionais PROFINET-IO estão à disposição:

Bloco funcional	Descrição da função
MSTAT 6.9.1	Controlar o estado do controlador pelo programa de aplicação
RALRM 6.9.2	Ler mensagens de alarme dos dispositivos
RDREC 6.9.4	Ler os conjuntos de dados dos dispositivos
SLACT 6.9.5	Controlar o estado dos dispositivos pelo programa de aplicação
WRREC 6.9.6	Escrever os conjuntos de dados dos dispositivos

Tabela 38: Visão geral – Blocos funcionais PROFINET-IO

Os blocos funcionais PROFINET-IO são parametrizados como os blocos funcionais PROFIBUS-DP, veja a partir do Capítulo 6.9.

## 5.2 Controlar o Consumer/Provider Status (IOxS)

Com as variáveis de sistema descritas neste capítulo, é possível controlar o Consumer/Provider Status (IOxS) pelo programa. Se o controle pelo programa de aplicação não for desejado, então, as variáveis de saída de um equipamento devem ser atribuídas com uma constante com o valor TRUE. Neste caso, os status são colocados para o valor GOOD logo que o módulo de comunicação recebeu valores de processo válidos no módulo processador.

A seguinte figura mostra a troca das variáveis de sistema entre o controlador HIMA e um dispositivo DO e um dispositivo DI, respectivamente.

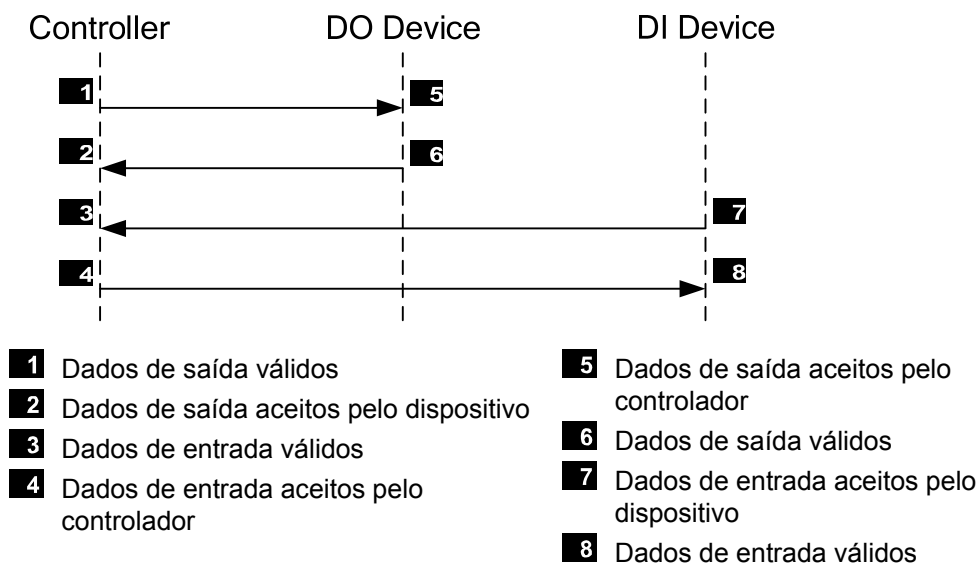


Figura 27: Controlar o Consumer/Provider Status (IOxS)

### 5.2.1 Variáveis de comando no controlador HIMA

Mediante as variáveis de saída *Valid Output Data* **1** e *Input Data Accepted by Controller* **4**, é possível controlar o Consumer/Provider Status (IOxS) pelo programa de aplicação.

Mediante as variáveis de entrada *Output Data Accepted by Device* **2** e *Valid Input Data* **3**, é possível ler o Consumer/Provider Status (IOxS) pelo programa de aplicação.

### 5.2.2 Variáveis de comando no dispositivo DO HIMA

Mediante a variável de saída *Valid Output Data* **6**, é possível controlar o Consumer/Provider Status (IOxS) pelo programa de aplicação.

Mediante a variável de entrada *Output Data Accepted by Controller* **5**, é possível ler o Consumer/Provider Status (IOxS) pelo programa de aplicação.

### 5.2.3 Variáveis de comando no dispositivo DI HIMA

Mediante a variável de saída *Input Data Accepted by Device* **7**, é possível controlar o Consumer/Provider Status (IOxS) pelo programa de aplicação.

Mediante a variável de entrada *Valid Input Data* **8**, é possível ler o Consumer/Provider Status (IOxS) pelo programa de aplicação.

### 5.3 PROFIsafe

Pressupõe-se como conhecida a especificação PROFIsafe da PNO!

PROFIsafe usa o protocolo PROFINET para a transmissão de dados direcionados à segurança até SIL 3 com base na tecnologia Ethernet.

O protocolo PROFIsafe está sobreposto ao protocolo PROFINET e contém dados de trabalho seguros, bem como as informações para o backup de dados. Os dados seguros PROFIsafe são transmitidos em conjunto com os dados PROFINET não relevantes para a segurança pelo protocolo PROFINET que está por baixo.

O PROFIsafe usa “canais de transmissão de dados não seguros” (Ethernet), de forma semelhante ao princípio “Black Channel” para a transmissão de dados seguros. Desta forma, o F-Host e o F-Device trocam os dados seguros PROFIsafe.

#### PROFIsafe com sistemas de comando HIMA

De acordo com a especificação PROFIsafe, o F-Host continua enviando pacotes de mensagens repetidos até o F-Device retornar uma confirmação de recepção ao F-Host. Somente então, o F-Host envia um novo pacote de mensagens ao F-Device.

Através de cada pacote de mensagens PROFIsafe repetido, o valor momentâneo do valor de processo é transmitido. Assim, pode ocorrer que o mesmo sinal de processo tenha valores diferentes nos pacotes de mensagens repetidos.

Nos equipamentos HIMA, o PROFIsafe está configurado de forma a transferir valores de processo apenas uma vez na primeira recepção de um pacote de mensagens. Os valores de processo dos pacotes de mensagens repetidos (com o mesmo número sequencial do pacote de mensagens) são descartados.

No caso de uma perda de conexão, depois de esgotar o  $F\_WD\_Time$ , as variáveis do valor de processo PROFIsafe assumem o valor inicial.

Para que um determinado valor de processo seja recebido do lado oposto (F-Host/F-Device), o valor de processo deve estar ativo de forma não alterada pelo menos pelo seguinte tempo:

$$2 * F\_WD\_Time + F\_WD\_Time2$$

O sistema PROFIsafe deve ser parametrizado pelo usuário de maneira que o  $SFRT$  (Safety Function Response Time) seja adequado para as respectivas funções de segurança.

Fórmulas de cálculo, veja Capítulo 5.3.3.

---

1

Para um comportamento conforme PROFIsafe, os seguintes requisitos devem estar satisfeitos:

- Os valores iniciais das variáveis de valor de processo devem estar configurados com “0”
- O parâmetro *AutoAwaitFParamsOnConnLoss* deve estar desativado, veja Capítulo 5.11.

Os ajustes padrão no SILworX correspondem ao comportamento conforme ao PROFIsafe!

---

### 5.3.1 Control-Byte e Status-Byte no PROFIsafe

As duas variáveis de sistema Control-Byte e Status-Byte estão contidas em cada submódulo PROFIsafe e são trocadas durante a comunicação entre o F-Host e F-Device, veja Capítulo 5.7.6.3 e Capítulo 5.11.4.

O Control Byte PROFIsafe é descrito no F-Host e lido no F-Device.

O Status Byte PROFIsafe é descrito no F-Device e lido no F-Host.

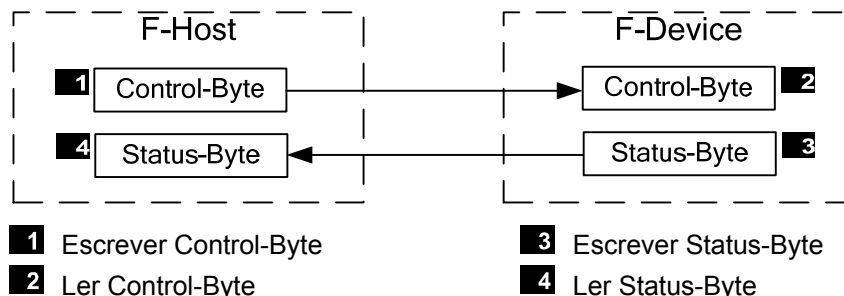


Figura 28: Control-Byte e Status-Byte no PROFIsafe



As variáveis de sistema Control-Byte e Status-Byte possuem funções adicionais na HIMA, que desviam da especificação PROFIsafe, veja Tabela 48 e Tabela 65.

---

### 5.3.2 F\_WD\_Time (tempo de Watchdog PROFIsafe)

Para uma conexão PROFIsafe funcional entre um F-Host HIMA e um F-Device vale o seguinte:

*F\_WD\_Time >*

*3 \* tempo de ciclo CPU \* quantidade de fatias de tempo de comunicação +  
2 \* distância de produção do controlador PROFINet<sup>1)</sup> +  
1 \* DAT (F-Device Acknowledgement Time) +  
2 \* tempo interno de barramento do F-Device +  
2 \* distância de produção do dispositivo PROFINet<sup>1)</sup> +  
2 \* Ethernet Delay*

<sup>1)</sup> *A distância de produção do controlador PROFINET e do dispositivo PROFINET normalmente são iguais e se calculam como segue: Fator de redução  
\* SendClockFaktor \* 31,25 µs*



DAT (F-Device Acknowledgement Time) e o *internal device bus time* devem ser consultados da descrição do equipamento do fabricante do F-Device!

No caso de F-Devices" HIMax e HIMatrix-L3, o *DAT* = *DATout* = *DATin* = 2 \* *WDT-CPU*

---

### 5.3.2.1 Observações sobre F\_WD\_Time (tempo de Watchdog PROFIsafe)

1. DAT (F-Device Acknowledgement Time) é o tempo que um F-Device precisa para reagir com uma resposta a uma mensagem PROFIsafe recebida. F-Device identifica a unidade segura (no caso da HIMA, o módulo CPU) que executa o F-Device-Stack. Em especial no caso de sistemas/equipamentos modulares, os tempos das funções/dos componentes não direcionadas/os à segurança não estão incluídos. Esta definição de DAT diverge da especificação PROFIsafe V2.5c, Capítulo 9.3.3. nos seguintes pontos:
  - DAT não inclui os tempos para o barramento interno do F-Device.
  - DAT não inclui a parte proporcional da distância de produção do dispositivo PROFINET.
  - DAT não inclui os atrasos como, p. ex., filtros dos valores de entrada/saída, retardo da física de saída/entrada, etc.
  - DAT representa DATin (Entrada) ou DATout (Saída), de acordo com a conexão.
  - Para todos os tempos devem ser usados os respectivos valores máximos.
2. Tempo de barramento interno do F-Device para HIMatrix-L3 e para HIMax é o *(max. Number of Communication Time Slices - 1) \* WDT CPU*.
3. Requisito: o F-Device trabalha ciclicamente e o seu DAT é  $DAT = 2 * \text{max. cycle}$ .
  - O F-Device **não trabalha com** communication time slices.  
 Se  
 $\text{HIMA CPU cycle time} * \text{number of communication time slices} < \text{F-Device cycle time}$ ,  
 então para o HIMA CPU cycle ainda deve ser acrescido ao cálculo do *F\_WD\_Time*  
 o  $\Delta = \text{F-Device cycle time} - \text{HIMA CPU cycle time} * \text{number of communication time slices}$ .
  - O F-Device trabalha **com** communication time slices.  
 Se  
 $(\text{HIMA CPU cycle time} * \text{number of communication time slices}) < (\text{F-Device cycle time} * \text{number of F-Device communication time slices})$ ,  
 então para o HIMA CPU cycle time deve ser acrescido ao cálculo do *F\_WD\_Time* ainda  $\Delta = (\text{F-Device cycle time} * \text{number of communication time slices}) - (\text{HIMA CPU cycle time} * \text{number of communication time slices})$ .

### 5.3.3 SFRT (Safety Function Response Time)

#### 5.3.3.1 Cálculo do SFRT entre um F-Device e um F-Host HIMA

O SFRT admissível para uma conexão PROFIsafe entre um F-Device e um HIMA F-Host com saída local é calculado como segue:

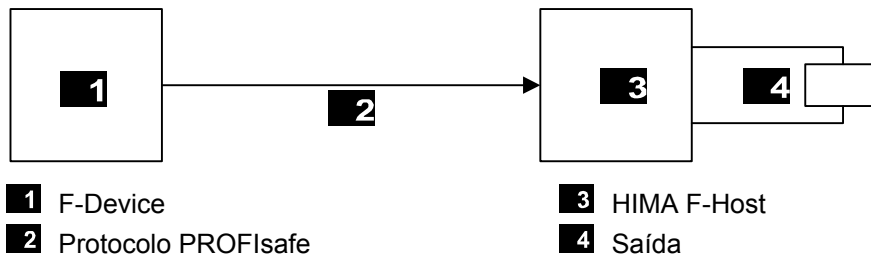


Figura 29: Tempo de reação entre um F-Device e um HIMA F-Host

$$\text{SFRT} \leq \text{MaxDataAgeIn} + 2 * \text{F\_WD\_TIME} + \text{MaxDataAgeOut} + \text{Tu}$$

Observações:

No caso do HIMax/HIMatrix-L3 e utilização local dos dados, ou seja, quando não são emitidos num HIMax IO, o tempo de tolerância a falhas do módulo CPU pode ser substituído por  $2 * \text{WDT-CPU}$ . Veja também *Remark to the SFRT calculations*, no Capítulo 5.3.3.3.

#### 5.3.3.2 Cálculo do SFRT com F-Devices e um HIMA F-Host

O SFRT admissível para uma conexão PROFIsafe entre um HIMA F-Host e um F-Device com saída local é calculado como segue:

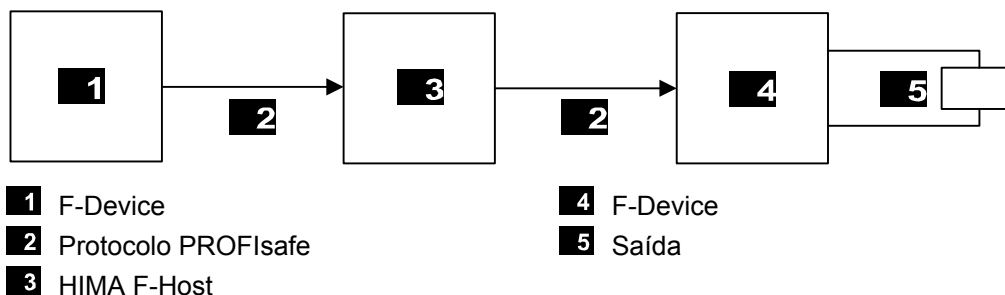


Figura 30: Tempo de reação com um HIMA F-Host e dois F-Devices

$$\text{SFRT} \leq \text{MaxDataAgeIn} + 2 * \text{F\_WD\_TIME}(\text{input}) + 2 * \text{WDT-CPU} + 2 * \text{F\_WD\_TIME}(\text{output}) + \text{MaxDataAgeOut} + \text{Tu}$$

#### 5.3.3.3 Observações sobre cálculos do SFRT (Safety Function Response Time)

1. Definição **SFRT** conforme IEC61784-3 Ed.2
2. Quaisquer atrasos adicionais eventuais no programa de aplicação (p. ex., por blocos funcionais TOF, TON, etc.) ou em módulos/componentes (p. ex., por filtros de saída, filtros de entrada, relés, etc.) devem ser acrescentados.
3. **MaxDataAgeIn** é a idade máxima de um valor de processo lida numa entrada física que um F-Device insere numa mensagem PROFIsafe. Porém, apenas a parte que ainda não está incluída no DATin.

i

No caso de HIMatrix-L3, o *MaxDataAgeIn* deve ser definido para 0 ms.

No caso de um sistema HIMax, *MaxDataAgeIn*

- possui para entradas físicas um tamanho de até  $FTT\ CPU^{1)} - 2 * WDT-CPU - DATin$ .
- deve ser definido com 0 ms para dados formados pelo programa de aplicação.

#### 4. **MaxDataAgeOut**

é o tempo de reação Worst-Case de um F-Output-Device ou F-Host para

- a) Emissão de valores de processo recebidos numa saída física,
- b) Comandar as saídas físicas depois de esgotar o *F\_WD\_Time* e
- c) Desativar as saídas físicas no caso de falha do equipamento.

- Para o HIMatrix-L3,  $MaxDataAgeOut = 3 * WDT-CPU$  (para saídas físicas).
- Para o HIMax, o  $MaxDataAgeOut = WDT-CPU + FTT-CPU^{1)}$  (para saídas físicas).
- Ao esgotar o *F\_WD\_TIME*, HIMax e HIMatrix-L3 reagem sem erro o mais tardar depois de  $2 * WDT-CPU$ .
- Se o F-Host/F-Device (com módulo CPU HIMA) falhar imediatamente antes desta reação, então, com o HIMatrix-L3, as saídas se tornam desenergizadas após o *WDT CPU*. Com o HIMax, isso ocorre o mais tardar após  $FTT\ CPU^{1)} - WDT\ CPU$ , pois o módulo de saída não recebeu nenhuma mensagem durante o período de um *WDT CPU*.
- No caso de assumir apenas um erro, o  $1 * WDT\ CPU$  pode ser subtraído do *MaxDataAgeOut* para HIMatrix-L3 e HIMax.

5. **Tu** é o mínimo do *DATin*, *DATout*, *WDT CPU*. Teoricamente, é possível calcular para *DATin* e *DATout* a metade de cada um, porém, para isso, o fabricante deve ter especificado com que imprecisão o dispositivo monitora o *F\_WD\_Time*. Se o dispositivo trabalhar ciclicamente, o  $DAT = 2 * max.\ device\ cycle$ . Com HIMA HIMatrix-L3 e HIMax F-Devices, o  $DATout = DATin = 2 * WDT\ CPU$ .

6. **F\_WD\_TIME**, veja Capítulo 5.3.2.

<sup>1)</sup> Tempo de tolerância a falhas do módulo CPU



## 5.4 Restrições para a operação segura de PROFIsafe

### 5.4.1 Endereçamento

A rede PROFIsafe na HIMA corresponde à rede Ethernet PROFINET pela qual mensagens PROFIsafe podem ser transmitidas. Rede aqui refere-se a uma rede lógica que pode se estender através de várias redes físicas parciais.

A separação de rede adequada para PROFIsafe está presente quando não é possível que mensagens PROFIsafe possam ultrapassar a separação de rede. Isso é o caso, por exemplo, ao usar um roteador com base no IP e as redes são conectadas a diferentes interfaces Ethernet do roteador.

Uma separação de rede para PROFIsafe não está presente quando, por exemplo, as redes são conectadas via roteadores de uma porta, Switches, Hubs ou Ethernet-Bridge.

Mesmo usando Switches comandáveis e se as redes PROFIsafe são separadas por exemplo por VLANs baseadas em portas, deve ser implementado endereçamento inequívoco, mesmo assim. Isso impede que, por exemplo, durante trabalhos de manutenção ou expansão, sejam criadas conexões entre redes PROFIsafe acidentalmente.

Os seguintes pontos devem ser respeitados para o endereçamento de equipamentos PROFIsafe:

- Deve ser garantido que os endereços-F dos equipamentos/módulos PROFIsafe numa rede PROFIsafe sejam inequívocos.
- Para a segurança de endereçamento ainda se recomenda selecionar os endereços-F de forma unívoca, mesmo com redes PROFIsafe separadas.
- Durante a colocação em funcionamento e alterações das funções de segurança deve ser verificado que a função de segurança use as entradas e saídas corretas dos equipamentos PROFIsafe corretos para além do PROFIsafe.
- Os módulos-F PROFIsafe devem ser configurados (p. ex., atribuindo um endereço-F ou F\_WD\_Time adequado) de maneira que numa rede PROFIsafe os módulos-F com comprimento de dados de entrada e de saída tenham assinaturas CRC1 diferentes. O usuário pode ler o CRC1 no SILworX.

#### **Observação:**

Neste caso, isso é de qualquer forma garantido para os módulos-F afetados se numa rede PROFIsafe apenas se usa um F-Host e se os parâmetros dos módulos-F numa rede PROFIsafe apenas se diferenciam pelo endereço-F.

Para que não seja gerada a mesma assinatura CRC1 acidentalmente, deve ser observado que, por exemplo, os parâmetros *F\_WD\_Time*, *F\_Prm\_Flag1/2* sejam iguais para todos os módulos-F e que os módulos-F não usem um CRC i-Par.

#### **O risco de equipamentos PROFIsafe com o mesmo comprimento de dados de entrada e saída**

Equipamentos PROFIsafe apenas podem ser operados se o comprimento de dados F-INPUT for desigual ao comprimento de dados F-OUTPUT da mesma conexão PROFIsafe.

Caso contrário, erros de endereçamento dos componentes padrão podem permanecer despercebidos no PROFIsafe e causar comportamento direcionado à segurança incorreto.

Com módulos-F HIMA que são configurados como parte do sistema de comando HIMA, os comprimentos de F-Input e F-Output devem ser selecionados diferentes um do outro. Para prevenir o risco de uma falha de uma função direcionada à segurança, apenas usar módulos F-Input ou F-Output. Não usar módulos F-Input/Output!

### 5.4.2 Aspectos relacionados à rede

A rede usada para a transmissão de mensagens PROFIsafe deve possuir disponibilidade e qualidade de transmissão suficiente.

i

Se o PROFIsafe detectar uma qualidade de transmissão reduzida que não é detectada pelos dispositivos de transmissão padrão (Ethernet), a reação de segurança ocorre.

Após uma reação de segurança causada por qualidade de transmissão reduzida, os problemas devem ser eliminados para garantir uma qualidade de transmissão satisfatória novamente. Apenas depois das medidas para isso necessárias é permitido acionar a confirmação para o reinício de PROFIsafe. Esta confirmação ocorre pelo sinal Operator-Acknowledge ou por um Reset.

#### **⚠ ATENÇÃO**



**Operator-Acknowledge e Reset apenas podem ser usados se não existir mais nenhum estado perigoso.**

Uma rede PROFIsafe deve ser protegida contra intervenções inadmissíveis (DoS, Hackers, ...). As medidas devem ser decididas em conjunto com as autoridades de supervisão. Isso possui relevância especial se técnicas Wireless são usadas para a transmissão.

Informações mais detalhadas, veja PROFIsafe Specification V2.5c, Tabela 23 e Tabela 24.

#### **Disponibilidade diante mensagens inseridas**

Pacotes de mensagens podem ser armazenados, p. ex., por componentes de rede como Switches e podem ser inseridos (enviados) num momento posterior<sup>1)</sup>. Estes pacotes de mensagens levam a um desligamento se são mais velhos (veja Consecutive Number Tabela 48 e Tabela 65) do que o pacote de mensagens recebido por último pelo equipamento PROFIsafe.

<sup>1)</sup> Suposição de erro para comunicação direcionada à segurança

## 5.5 HIMA PROFINET-IO Controller e PROFIsafe F-Host

Este capítulo descreve as características do HIMA PROFINET-IO Controller e PROFIsafe F-Host, bem como as funções de menu e diálogos no SILworX que são necessários para a configuração do HIMA PROFINET-IO Controller e PROFIsafe Host.

### Requisitos de sistema PROFINET-IO Controller

Equipamentos necessários e requisitos de sistema:

Elemento	Descrição
Sistema de comando	HIMax com módulo COM HIMatrix L3
Módulo CPU	As interfaces Ethernet do módulo processador não podem ser usadas para PROFINET-IO.
Módulo COM	Ethernet 10/100BaseT.
Ativação	A liberação ocorre mediante código de liberação do software, veja Capítulo 3.4.

Tabela 39: Requisitos de sistema e equipamentos necessários para PROFINET-IO Controller

### PROFINET-IO Controller e PROFIsafe Host Properties

Características	Descrição
Direcionado à segurança	Não
Taxa de transmissão	100 Mbit/s full duplex
Caminho de transporte	Interfaces Ethernet dos módulos COM As interfaces Ethernet em uso podem ser usadas simultaneamente para outros protocolos.
Classe de conformidade	O PROFINET-IO Controller satisfaz as exigências da Conformance Class A.
Real Time Class	RT classe 1
Quantidade máx. de controladores PROFINET-IO	Para cada módulo COM é possível configurar um controlador PROFINET-IO.
Quantidade máx. de dispositivos PROFINET-IO relações de aplicação (ARs)	Um controlador PROFINET-IO pode estabelecer relações de aplicação (AR) com no máx. 64 dispositivos PROFINET-IO.
Quantidade de relações de comunicação (CRs por AR)	Padrão: 1 Input CR, 1 Output CR, 1 Alarm CR
Comprimento máx. de dados de processo de uma relação de comunicação (CR)	Output: máx. 1440 Bytes Input: máx. 1440 Bytes
Ciclo de envio	Possível pelo ajuste do <i>Reduction Rate</i> no nível do dispositivo.
<b>As seguintes propriedades valem para PROFIsafe</b>	
Quantidade máx. F-Hosts (HIMax)	1024
Quantidade máx. F-Hosts (HIMatrix L3)	512
Comprimento máx. de dados de processo de uma relação de comunicação (CR)	Output: máx. 123 Bytes dados de trabalho + 5 Bytes <sup>1)</sup> Input: máx. 123 Bytes dados de trabalho + 5 Bytes <sup>1)</sup>
Tamanho máx. dados de trabalho HIMax: HIMatrix L3:	1024 x 123 Bytes = 125 952 Bytes 512 x 123 Bytes = 62 976 Bytes
<sup>1)</sup> 5 Bytes dados de gestão (Status/Control Bytes e CRC)	

Tabela 40: Características controlador PROFINET-IO

## 5.6 Exemplo PROFINET-IO/PROFIsafe

Neste exemplo é descrito como se configura um controlador PROFINET-IO que tem uma conexão a um dispositivo PROFINET-IO num sistema de comando HIMax.

O dispositivo PROFINET-IO está equipado com um módulo PROFINET-IO e um módulo PROFIsafe. No controlador PROFINET-IO o dispositivo PROFINET-IO deve ser configurado exatamente da forma como é estruturado.

### 5.6.1 Criar o controlador HIMA PROFINET-IO no SILworX

**Assim cria-se um novo controlador PROFINET-IO:**

1. Selecionar na árvore de estrutura **Configuration, Resource, Protocols**.
2. Selecionar no menu de contexto dos protocolos **New, PROFINET-IO Controller** para adicionar um novo controlador PROFINET-IO.
3. Selecionar no menu de contexto do controlador PROFINET-IO **Properties**.
4. Introduzir no campo **Name** o nome de equipamento do controlador.
5. Selecionar **COM-Module**.

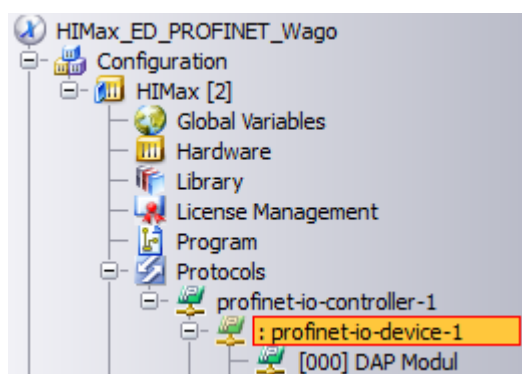


Figura 31: Controlador PROFINET-IO na árvore de estrutura HIMax

#### 5.6.1.1 Configuração do dispositivo no controlador HIMA PROFINET-IO

**Assim cria-se um dispositivo PROFINET-IO no controlador PROFINET-IO:**

1. Selecionar no menu de contexto do controlador PROFINET-IO **New, PROFINET-IO Device**.

**Assim lê-se o arquivo de biblioteca GSDML de uma fonte de dados externa (p. ex., CD, USB-Stick, Internet):**

1. Selecionar na árvore de estrutura **Configuration, Resource, Protocols, PROFINET-IO Controller, GSDML Library**.
2. Selecionar no menu de contexto da biblioteca GSDML **New** e adicionar o arquivo GSDML correspondente ao dispositivo PROFINET-IO.

**Assim carrega-se o arquivo GSDML para um novo dispositivo PROFINET-IO:**

1. Selecionar na árvore de estrutura **Configuration, Resource, Protocols, PROFINET-IO Controller, PROFINET-IO Device**.
2. Selecionar no menu de contexto **Properties** e abrir o registro **Parameter**.
3. Introduzir no campo **Name** o nome de equipamento do dispositivo.
4. Introduzir o endereço IP do PROFINET-IO Device no campo **IP Address**.

5. Selecionar no menu de seleção **GSDML file** o arquivo de biblioteca GSDL correspondente ao dispositivo PROFINET-IO e fechar **Properties**.

**Assim seleciona-se o Device Access Point (DAP) para o dispositivo PROFINET-IO:**

1. Selecionar na árvore de estrutura **Configuration, PROFINET-IO Controller, PROFINET-IO Device, DAP module**.
2. Selecionar no menu de contexto **Edit** e selecionar o conjunto de dados adequado de (*DAP = Device Access Point*) para o dispositivo PROFINET-IO.

**i**

Muitas vezes, o arquivo de biblioteca GSDML contém vários (*DAP = Device Access Point*) de um fabricante.

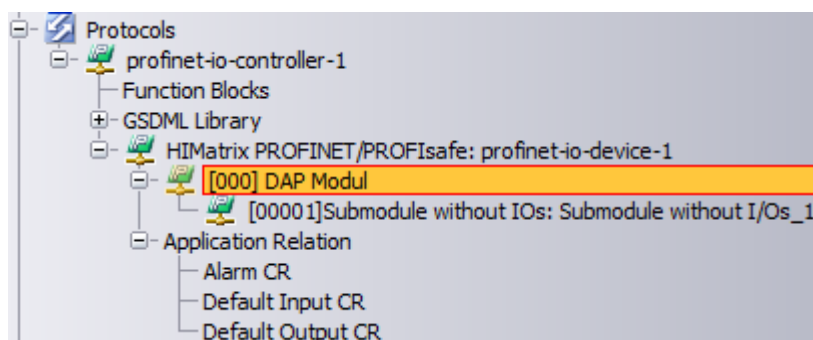


Figura 32: Device Access Point (DAP) para o dispositivo PROFINET-IO

**Assim configuram-se os slots de módulo:**

1. Abrir na árvore de estrutura **Protocols, PROFINET-IO Device**.
2. Selecionar no menu de contexto PROFINET-IO Device **New** para abrir a lista de módulos.
3. Selecionar os módulos adequados para o dispositivo PROFINET-IO da lista de módulos e confirmar com **Add Module(s)**.

**Assim numeram-se os módulos de dispositivo PROFINET-IO:**

O **Device Access Point (DAP) Module** possui o slot 0 fixo. Todos os demais módulos PROFINET-IO Device devem ser numerados.

1. Selecionar no menu de contexto PROFINET-IO Device Modul **Properties**.
2. Introduzir no campo **Slot** os slots dos módulos de dispositivo da forma como o dispositivo PROFINET-IO é estruturado de fato.
3. Repetir este passo para os demais **PROFINET IO device modules**.

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML.

**Assim configura-se a Application Relation:**

1. Abrir na árvore de estrutura **PROFIsafe-IO Device, Application Relation**.
2. Clicar com o botão direito em **Default Input CR** e selecionar no menu de contexto **Properties**.
3. Adaptar o parâmetro fator de redução, p. ex., ajustar para 4.
4. Clicar com o botão direito em **Default Output CR** e selecionar no menu de contexto **Properties**.
5. Adaptar o parâmetro fator de redução, p. ex., ajustar para 4.

### 5.6.1.2 Configuração do módulo Device Access Point (DAP)

#### Assim, configura-se o módulo Device Access Point (DAP)

1. Selecionar **[000] DAP Module, [xxxxx] DAP Submodule**.
2. Clicar com o botão direito em **[xxxxx] DAP Submodule** e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **System Variables** e atribuir uma variável global com o valor inicial TRUE à variável de saída *Input Data Accepted by Controller* se controlar o Consumer/Provider Status com lógica especial do programa de aplicação não for desejado.

---

**i**

Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

---

#### Ajustar os parâmetros de cabeçalho, p. ex., para um DAP com ajustes de alarme.

#### Assim ajustam-se os parâmetros de cabeçalho do Device Access Point (DAP) para o dispositivo PROFINET-IO:

1. Selecionar na árvore de estrutura **Protocols, PROFINET-IO Controller, PROFINET-IO Device, DAP Module, [xxxxx] DAP Submodule, Alarm Settings (Header): Parameters**.
2. Selecionar no menu de contexto **Properties**.
3. Introduzir no campo **Name** o nome de parâmetro dos parâmetros de cabeçalho.
4. Mediante o botão **Edit** abre-se um diálogo onde é possível editar os ajustes para interfaces e diagnóstico/alarmes.

### 5.6.1.3 Configuração dos módulos PROFINET-IO Device

---

**i**

A soma das variáveis em Byte deve corresponder ao tamanho do respectivo módulo em Byte.

---

#### Assim configura-se o módulo PROFINET-IO Device

1. Selecionar **[001] PROFINET-IO Device Module, [xxxxx] PROFINET-IO Device Submodule**.
2. Clicar com o botão direito em **[xxxxx] Submodule** e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Input Signals**.
5. Clicar com o botão direito em um espaço vazio na área **Input Signals** e selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.
6. Selecionar no diálogo **Edit** o registro **System Variables** e atribuir uma variável global com o valor inicial TRUE às variáveis de saída *Input Data Accepted by Controller* e *Valid Output Data* se controlar o Consumer/Provider Status não for desejado.

---

**i**

Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

---

#### 5.6.1.4 Configuração dos módulos PROFIsafe-IO Device



A soma das variáveis em Byte deve corresponder ao tamanho do respectivo módulo em Byte.

##### Assim configura-se o módulo PROFIsafe-IO Device

1. Selecionar **[001] PROFsafe-IO Device Module, [xxxxx] PROFINET-IO Device Submodule**.
2. Clicar com o botão direito em **[xxxxx] Submodule** e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Input Signals**.
5. Clicar com o botão direito em um espaço vazio na área **Input Signals** e selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.
6. Selecionar no diálogo **Edit** o registro **System Variables** e atribuir uma variável global com o valor inicial TRUE às variáveis de saída *Input Data Accepted by Controller* e *Valid Output Data* se controlar o Consumer/Provider Status não for desejado.



Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

##### Assim configuram-se os F-Parameters

1. Selecionar **[001] PROFsafe-IO Device Module, [xxxxx] PROFIsafe-IO Device Submodule, F-Parameters**.
2. Clicar com o botão direito em **F-Parameters** e selecionar no menu de contexto **Properties**.
3. Ajustar os seguintes parâmetros:
  - F\_Dest\_Add: ajustar o endereço de destino do módulo de dispositivo
  - F\_WD\_Time: tempo de Watchdog da conexão a este módulo de dispositivo

##### Verificar PROFINET-IO Configuration

1. Na árvore de estrutura, **Configuration, Resource, Protocols, PROFINET-IO Controller** deve ser aberto.
2. Clicar com o botão direito em **PROFINET-IO Controller** e selecionar no menu de contexto **Verification**.
3. Verificar cuidadosamente as entradas no log, corrigir erros onde for necessário.



Recompilar o recurso e carregar ao sistema de comando para que a configuração para a comunicação PROFINET-IO se torne efetiva.

### 5.6.1.5 Determinar os dispositivos PROFINET-IO na rede

**Assim localiza-se o dispositivo PROFINET-IO na rede Ethernet:**

1. Efetuar o login do módulo para o módulo de comunicação com o **PROFINET-IO Controller**.
2. Selecionar na árvore de estrutura da visualização online **PROFINET-IO-Controller, PROFINET IO Station**.
3. Selecionar no menu de contexto **Get PROFINET IO Network Stations**.
  - ☒ Todos os dispositivos PROFINET na rede deste controlador PROFINET-IO são listados.

**Assim configura-se o PROFINET-IO Device na visualização online:**

1. Na lista de equipamentos, clicar com o botão direito no dispositivo PROFINET-IO a ser configurado para alterar os ajustes.
2. Com a função de menu de contexto **Name the PROFINET IO Device** atribui-se o nome do equipamento.
  - ☒ Certificar-se de que o nome de equipamento PROFINET-IO corresponda ao do projeto. (Apenas letras minúsculas são permitidas!)
3. Com a função de menu de contexto **Network Settings** atribuem-se o endereço IP, subnet mask e gateway.

---

**i**

O nome de equipamento PROFINET-IO e os ajustes de rede do dispositivo comunicação devem estar introduzidos no controlador PROFINET-IO, senão, a comunicação não é possível.

---



## 5.7 Funções de menu do controlador PROFINET-IO

### 5.7.1 Exemplo para a árvore de estrutura do controlador PROFINET-IO

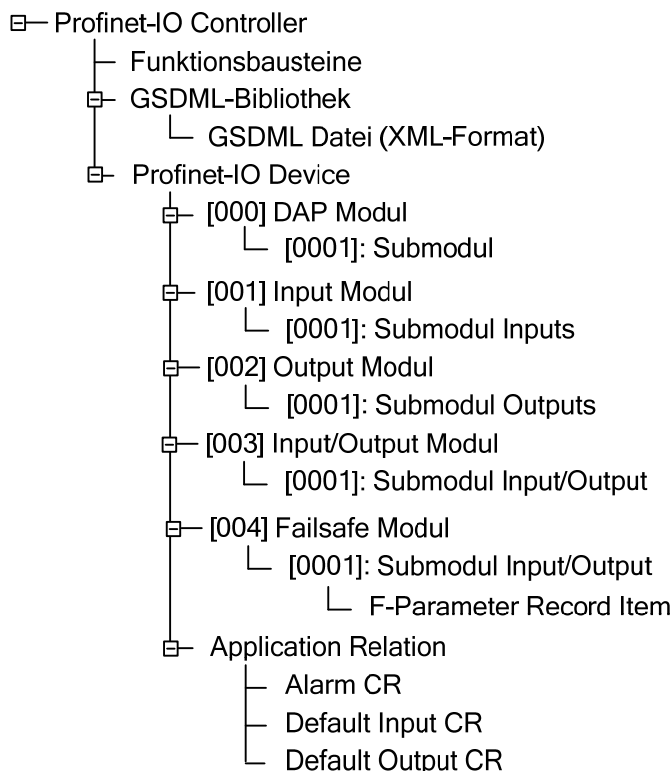


Figura 33: Árvore de estrutura do controlador PROFINET-IO

### 5.7.2 PROFINET-IO Controller

A função de menu **Properties** do menu de contexto do controlador PROFINET-IO abre o diálogo **Properties**. A janela de diálogo contém os seguintes registros:

#### 5.7.2.1 Registro PROFINET-IO (características)

Elemento	Descrição
Type	PROFINET-IO Controller
Name	Nome de equipamento do controlador PROFINET-IO
Process Data Refresh Rate [ms]	Tempo de atualização em milissegundos com o qual os dados do protocolo são trocados entre COM e CPU. Se o <i>Refresh Rate</i> for zero ou menor do que o tempo de ciclo do sistema de comando, a troca de dados ocorre o mais rápido possível. Faixa de valores: 4 ... (2 <sup>31</sup> - 1) Valor padrão: 0
Force Process Data Consistency	Ativado: Transferência dos dados completos do protocolo de CPU para COM dentro de um ciclo de CPU. Desativado: Transferência dos dados completos do protocolo de CPU para COM, distribuídos sobre vários ciclos de CPU cada vez 1100 Byte por direção de dados. Com isso, eventualmente também é possível reduzir o tempo de ciclo do sistema de comando. Valor padrão: Ativado

Elemento	Descrição
Module	Seleção do módulo COM no qual o protocolo é processado.
Activate Max. $\mu$ P Budget	Ativado: Transferir o limite do $\mu$ P-Budget do campo <i>Max. <math>\mu</math>P-Budget in [%]</i> . Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P-Budget in [%]	Valor máximo de $\mu$ P-Budget do módulo que pode ser produzido ao processar o protocolo. Faixa de valores: 1...100% Valor padrão: 30%
RPC Port Server	Remote Procedure Call Port Faixa de valores: 1024...65535 Valor padrão: 49152 RPC Port Server e RPC Port Client não podem ser idênticos!
RPC Port Client	Remote Procedure Call Port Faixa de valores: 1024...65535 Valor padrão: 49153 RPC Port Server e RPC Port Client não podem ser idênticos!
F_Source_Add	Endereço do controlador (F-Host). O usuário deve usar endereços inequívocos de PROFIsafe Controller/Device numa rede PROFIsafe. <i>Veja também IEC61784-3-3 V2.5c. Capítulo 9.7</i>

Tabela 41: Registro PROFINET-IO (características)

### 5.7.3 Dispositivo PROFINET-IO (no controlador)

A função de menu **Properties** do menu de contexto do controlador PROFINET-IO abre o diálogo **Properties** que contém os seguintes registros:

#### 5.7.3.1 Registro Parâmetros (características)

Elemento	Descrição
Name	Nome de equipamento do dispositivo PROFINET-IO
IP Address	Endereço IP do parceiro de comunicação. Faixa de valores: 0.0.0.0...255.255.255.255 Valor padrão: 192.168.0.99 Não usar endereços IP atribuídos, veja Capítulo 3.5.3.
Subnet Mask	Subnet Mask da subrede endereçada na qual o dispositivo se encontra. Faixa de valores: 0.0.0.0...255.255.255.255 Valor padrão: 255.255.255.0
GSDML File	GSDML é a abreviação para “Generic Station Description Markup Language” e é uma linguagem de descrição baseada em XML. O arquivo GSDML contém os dados de origem do dispositivo PROFINET

Tabela 42: Registro Parâmetros (características)

#### Registro Model e Features

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML, como, p. ex., *Manufacturer Name*, *Device Description* ou *Supported Factors*. Estes dados servem como informação de apoio para a parametrização do equipamento e não podem ser alterados.

### 5.7.4 DAP Module (Device Access Point Module)

Dentro de um dispositivo PROFINET sempre é criado um módulo para a ligação do barramento (DAP: Device Access Point Module). O módulo DAP é padrão e não pode ser excluído.

A função de menu **Properties** do menu de contexto do módulo DAP abre o diálogo **Properties** que contém os seguintes registros:

#### Registro Parâmetros (características)

Elemento	Descrição
Name	Nome para o módulo DAP
Slot	Não pode ser alterado. Valor padrão: 0

Tabela 43: Registro Parâmetros (características)

#### Registro Model e Features

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML, como, por exemplo, *Module ID*, *Hardware/Software Version*. Estes dados servem como informação de apoio para a parametrização do equipamento e não podem ser alterados.

## 5.7.4.1 Submódulo DAP (características)

A função de menu **Properties** do menu de contexto do submódulo DAP abre o diálogo **Properties** que contém os seguintes registros:

## Registro Parâmetros

Elemento	Descrição
Name	Nome do submódulo de entrada
Sub-Slot	Valor padrão: 1
IO Data CR, Inputs	Seleção da relação de comunicação (Communication Relation CR) na qual as entradas do submódulo devem ser transferidas. - None: nenhuma - Default Input CR
Input Data Accepted by Controller	Seleção da relação de comunicação (Communication Relation CR) na qual o IO Consumer status (CS) do submódulo deve ser transferido. - None: nenhuma - Default Output CR

Tabela 44: Registro Parâmetros

## Registro Model e Features

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML, por exemplo *Submodule ID*, *Data lengths*. Estes dados servem como informação de apoio para a parametrização do equipamento e não podem ser alterados.

## 5.7.4.2 Submódulo DAP (Edit)

A função de menu **Edit** do menu de contexto dos submódulos de entrada abre o diálogo **Edit** que contém os seguintes registros:

## Registro variáveis de sistema

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar ou controlar o estado do submódulo PROFINET-IO no programa de aplicação.

Elemento	Descrição
Valid input data	True    Dados de entrada válidos GOOD
	False    Dados de entrada inválidos BAD
Input Data Accepted by Controller	True    Valid input data GOOD
	False    Dados de entrada inválidos BAD

Tabela 45: Registro variáveis de sistema



Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

---

#### 5.7.4.3 Parâmetros de cabeçalho

Alguns dispositivos contêm os chamados parâmetros de cabeçalho nos quais é possível, por exemplo, ativar/desativar diagnóstico, alarme e interfaces.

#### 5.7.5 Input/Output PROFINET-IO Module

Um módulo Input/Output PROFINET-IO pode ter vários submódulos. Controladores HMax PROFINET-IO possuem em cada módulo Input/Output PROFINET-IO um submódulo.

Nos módulos de entrada PROFINET-IO, são introduzidas as variáveis de entrada do controlador HMax PROFINET-IO que são enviadas pelo dispositivo PROFINET-IO.

Nos módulos de saída PROFINET-IO, são introduzidas as variáveis de saída do controlador HMax PROFINET-IO que são enviadas pelo dispositivo PROFINET-IO.

##### Assim criam-se os módulos PROFINET-IO necessários:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFINET-IO Device**.
2. Selecionar no menu de contexto PROFINET-IO Device **New**.
3. Selecionar módulos adequados.

A função de menu **Properties** do menu de contexto dos módulos de entrada/saída PROFINET-IO abre o diálogo **Properties** que contém os seguintes registros:

##### Registro Parâmetros

Elemento	Descrição
Name	Nome do módulo Input/Output PROFINET-IO
Slot	0 a 32767 Valor padrão: 1

Tabela 46: Registro Parâmetros do módulo I/O PROFINET-IO

##### Registro Model e Features

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML, como, por exemplo, *Module ID*, *Hardware/Software Version*. Estes dados servem como informação de apoio para a parametrização do equipamento e não podem ser alterados.

### 5.7.6 Submódulo Input

Com os parâmetros dos submódulos são ajustadas as relações de comunicação e o comportamento dos módulos no caso de uma interrupção da comunicação.

#### 5.7.6.1 Submódulo Input (características)

A função de menu **Properties** do menu de contexto dos *Input Submodules* abre o diálogo **Properties** que contém os seguintes registros:

##### Registro Parâmetros

Elemento	Descrição
Name	Nome do submódulo de entrada
Sub-Slot	Não pode ser alterado para controladores HIMax PROFINET-IO. Valor padrão: 1
IO Data CR, Inputs	Seleção da relação de comunicação (Communication Relation CR) na qual as entradas do submódulo devem ser transferidas. - None: nenhuma - Default Input CR
Input Data Accepted by Controller	Seleção da relação de comunicação (Communication Relation CR) na qual o IO Consumer status (CS) do submódulo deve ser transferido. - None - Default Output CR
Shared Input	Ativado Vários controladores PROFINET-IO podem acessar as entradas. Desativado Apenas um controlador PROFINET-IO pode acessar as entradas.
Input Values if IO CR is Disconnected	Comportamento das variáveis de entrada deste submódulo PROFINET-IO no caso da interrupção da conexão. Retain Last Value As variáveis de entrada são congeladas para o valor atual e são usadas até a conexão ser restabelecida. Adopt Initial Values Os dados iniciais são usados para as variáveis de entrada.

Tabela 47: Registro Parâmetros

##### Registro Model e Features

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML, como, por exemplo, *Submodule ID*, *Hardware/Software Version* ou *Data Length*. Estes dados servem como informação de apoio para a parametrização do equipamento e não podem ser alterados.

### 5.7.6.2 Submódulo Input (Edit)

A função de menu **Edit** do menu de contexto dos submódulos de entrada abre o diálogo **Edit**. A janela de diálogo contém os seguintes registros:

#### Registro variáveis de sistema

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar ou controlar o estado do submódulo PROFINET-IO no programa de aplicação.

Elemento	Descrição	
Valid input data	True	Valid Input Data GOOD
	False	Dados de entrada inválidos BAD
Input Data Accepted by Controller	True	Valid input data GOOD
	False	Dados de entrada inválidos BAD
<b>Os seguintes parâmetros apenas estão disponíveis para módulos PROFIsafe</b>		
Valid Output Data	True	Dados de saída válidos GOOD
	False	Dados de saída inválidos BAD
Output Data Accepted by Device	True	Valid Output Data GOOD
	False	Dados de saída inválidos BAD



Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

---

Elemento	Descrição
PROFIsafe Control	<p>PROFIsafe envia do controlador com cada mensagem o PROFIsafe Control Byte ao dispositivo que pode ser atribuído no programa de aplicação. Veja também Capítulo 5.3.1.</p> <p>Bit 0    iPar_EN_C: Para carregar novos iParameter no F-Device, iPar_EN_C deve ser colocado em TRUE para desbloquear o F-Device. Enquanto iPar_EN_C estiver ajustado para TRUE, valores seguros "0" são trocados entre F-Host e F-Device.</p> <p>Bit 1    OA_C: Operator Acknowledge. Após um erro PROFIsafe (p. ex., erro de CRC ou Timeout), o Bit deve ser ajustado em TRUE, durante no mínimo um ciclo PROFIsafe. Se uma conexão PROFIsafe deve ser (novamente) iniciada, a liberação (Operator Acknowledge) somente pode ser dada se não houver estados perigosos.</p> <p>Bit 2    Reserved</p> <p>Bit 3    Reserved</p> <p>Bit 4    Activate_FV_C: FALSE: Valores de processo são trocados entre F-Host e F-Device. TRUE: Valores seguros "0" são trocados entre o F-Host e o F-Device.</p> <p>Bit 5    Reserved</p> <p>Bit 6    Reserved</p> <p>Bit 7    Reset_Comm: Reset da comunicação PROFIsafe, o Stack do protocolo é colocado no estado inicial. <i>O Bit deve ser colocado em TRUE até que do PROFIsafe Status</i> <i>Bit 2 Reset_Comm o valor TRUE seja lido de volta.</i></p>
PROFIsafe RoundTrip Time last	<p>Este é o tempo para o F-Host entre o envio de uma mensagem de dados (com Consecutive Number N) e a recepção do Acknowledgments correspondente (com Consecutive Number N), medido em milissegundos.</p>



Elemento	Descrição
PROFIsafe Status	<p>PROFIsafe recebe no Host em cada mensagem o PROFIsafe Status Byte que pode ser avaliado no programa de aplicação.</p> <p>Bit 0 iPar_OK_S TRUE: novos iParameter recebidos FALSE: sem alteração</p> <p>Bit 1 OA_Req_S Operator Acknowledge Requested – Confirmação do operador solicitada.</p> <p>Bit 2 Reset_Comm É o valor Reset_Comm, lido de volta do Host-Control Byte. Este Bit assinala se o Reset_Comm chegou.</p> <p>Bit 3 FV_activated_S</p> <p>Bit 4 Toggle_h</p> <p>Bit 5 Device_Fault TRUE: O F-Device comunicou uma falha de dispositivo. FALSE: O F-Device não comunicou uma falha de dispositivo.</p> <p>Bit 6 WD_timeout TRUE: Ou o F-Device comunicou um WD-Timeout ou o no F-Host o Host-Timeout esgotou. FALSE: Nem no F-Device nem no F-Host ocorreu um Timeout.</p> <p>Bit 7 CRC TRUE: Ou o F-Device comunicou um erro de CRC ou no F-Host ocorreu um erro de CRC. FALSE: Nem no F-Device nem no F-Host ocorreu um erro de CRC.</p>

Tabela 48: Registro variáveis de sistema

No registro **Process Variables** são introduzidas as variáveis de entrada.

## 5.7.6.3 F-Parameter do submódulo Input (só para módulos PROFIsafe)

Os F-Device PROFIsafe precisam para a troca segura de dados de processo os F-Parameter normalizados. O F-Device apenas inicia a comunicação se F-Parameters válidos foram ajustados. Os parâmetros em cinza não podem ser editados e são parcialmente definidos pelo arquivo GSDML ou calculados automaticamente.

Elemento	Descrição
Name	Nome do módulo
Index	Index do módulo
F_Par_Version	Apenas V2-mode é suportado. V1-mode é rejeitado. Definido pelo arquivo GSDML.
F_Source_Add	O F_Source_Address do F-Host deve ser inequívoco dentro da rede PROFIsafe! Faixa de valores: 1 a 65534
F_Dest_Add	O F_Destination_Address do F-Device deve ser inequívoco dentro da rede PROFIsafe! Faixa de valores: 1 a 65534
F_WD_Time	Watchdogtime Faixa de valores: 1 ms a 65534 ms
F_iPar_CRC	Introduzir o F_iPar_CRC do F-Device neste campo.
F_SIL	Indicador no nível SIL 0 – SIL1 1 – SIL2 2 – SIL3 3 – NoSIL Definido pelo arquivo GSDML.
F_Check_iPar	Indicador do iParameter CRC Definido pelo arquivo GSDML.
F_Block_ID	Estrutura dos F-Parameters Definido pelo arquivo GSDML.
F_CRC_Length	Indica se o CRC de 3 Bytes ou de 4 Bytes é usado. Definido pelo arquivo GSDML.
F_Par_CRC	Indicador do F-Parameter CRC (CRC1) É calculado a partir dos F-Parameters atuais

Tabela 49: F-Parameters do submódulo Input (características)

### 5.7.7 Submódulo Output

Com os parâmetros dos submódulos são ajustadas as relações de comunicação e o comportamento dos módulos no caso de uma interrupção da comunicação.

#### 5.7.7.1 Submódulo Output (características)

A função de menu **Properties** do menu de contexto dos *Output Submodules* abre o diálogo **Properties** que contém os seguintes registros:

##### Registro Parâmetros

Elemento	Descrição
Name	Nome do submódulo de saída
Sub-Slot	Não pode ser alterado para controladores HIMax PROFINET-IO. Valor padrão: 1
IO Data CR, Outputs	Seleção da relação de comunicação (Communication Relation CR) na qual as saídas do submódulo devem ser transferidas. - None: nenhuma - Default Input CR
Output Data Accepted by Device	Seleção da relação de comunicação (Communication Relation CR) na qual o IO Consumer status (CS) do submódulo deve ser transferido. - None: nenhuma - Default Output CR

Tabela 50: Registro Parâmetros

##### Registro Model e Features

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML, por exemplo *Submodule ID*, *Hardware/Software Version*, *Data lengths*. Estes dados servem como informação de apoio para a parametrização do equipamento e não podem ser alterados.

## 5.7.7.2 Submódulo Output (Edit)

A função de menu **Edit** do menu de contexto dos submódulos de saída abre o diálogo **Edit** que contém os seguintes registros:

## Registro variáveis de sistema

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar ou controlar o estado do submódulo PROFINET-IO no programa de aplicação.

Elemento	Descrição	
Valid Output Data	True	Valid Output Data GOOD
	False	Dados de saída inválidos BAD
Output Data Accepted by Device	True	Valid Output Data GOOD
	False	Dados de saída inválidos BAD
<b>Os seguintes parâmetros apenas estão disponíveis para módulos PROFIsafe</b>		
Valid input data	True	Dados de entrada válidos GOOD
	False	Dados de entrada inválidos BAD
Input Data Accepted by Controller	True	Valid input data GOOD
	False	Dados de entrada inválidos BAD
Outros parâmetros para módulos PROFIsafe, veja Tabela 48.		

Tabela 51: Registro variáveis de sistema

---

**i**

Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

---

No registro **Process Variables** são introduzidas as variáveis de saída.

## 5.7.7.3 F-Parameter do submódulo Output (só para módulos PROFIsafe)

Descrição dos F-Parameters, veja Capítulo 5.7.6.3.

### 5.7.8 Submódulo entradas e saídas

Com os parâmetros dos submódulos são ajustadas as relações de comunicação e o comportamento dos módulos no caso de uma interrupção da comunicação.

#### 5.7.8.1 Submódulo entradas e saídas (características)

A função de menu **Properties** do menu de contexto dos *Input and Output Submodules* abre o diálogo **Properties** que contém os seguintes registros:

##### Registro Parâmetros

Elemento	Descrição
Name	Nome do submódulo de entrada/saída
Sub-Slot	Não pode ser alterado para controladores HIMax PROFINET-IO. Valor padrão: 1
IO Data CR, Inputs	Seleção da relação de comunicação (Communication Relation CR) na qual as entradas do submódulo devem ser transferidas. - None: nenhuma - Default Input CR
IO Data CR, Outputs	Seleção da relação de comunicação (Communication Relation CR) na qual as saídas do submódulo devem ser transferidas. - None: nenhuma - Default Output CR
Input Data Accepted by Controller	Seleção da relação de comunicação (Communication Relation CR) na qual o IO Consumer status (CS) do submódulo deve ser transferido. - None - Default Output CR
Output Data Accepted by Device	Seleção da relação de comunicação (Communication Relation CR) na qual o IO Consumer status (CS) do submódulo deve ser transferido. - None - Default Input CR
Input Values if IO CR is Disconnected	- Manter último valor - Assumir valores iniciais

Tabela 52: Registro Parâmetros

##### Registro Model e Features

Os registros **Model** e **Features** mostram mais detalhes do arquivo GSDML, como, por exemplo, *Submodule ID*, *Hardware/Software Version* ou *Data Length*. Estes dados servem como informação de apoio para a parametrização do equipamento e não podem ser alterados.

### 5.7.8.2 Submódulo entradas e saídas (Edit)

A função de menu **Edit** do menu de contexto dos submódulos de entrada/saída abre o diálogo **Edit** que contém os seguintes registros:

#### Registro variáveis de sistema

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar ou controlar o estado do submódulo PROFINET-IO no programa de aplicação.

Elemento	Descrição	
Valid Output Data	True	Valid Output Data GOOD
	False	Dados de saída inválidos BAD
Output Data Accepted by Device	True	Valid Output Data GOOD
	False	Dados de saída inválidos BAD
Valid input data	True	Dados de entrada válidos GOOD
	False	Dados de entrada inválidos BAD
Input Data Accepted by Controller	True	Valid input data GOOD
	False	Dados de entrada inválidos BAD
Parâmetros para módulos PROFIsafe, veja Tabela 48.		

Tabela 53: Registro variáveis de sistema

No registro **Process Variables** são introduzidas as variáveis de entrada e as variáveis de saída na respectiva área.

·  
**i**

Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

### 5.7.8.3 F-Parameter do submódulo Inputs/Outputs (só para módulos PROFIsafe)

Descrição dos F-Parameters, veja Capítulo 5.7.6.3.

### 5.7.9 Application Relation (características)

Uma relação de aplicação (AR: Application Relation) é um objeto lógico para a troca de dados entre controlador e dispositivo. A transmissão de dados dentro da relação de aplicação ocorre neste exemplo (veja Figura 34) pelas relações de comunicação padrão (Alarm CR, Default Input CR e Default Output CR). Estas relações de comunicação já estão configuradas por padrão nos módulos de entrada e saída.

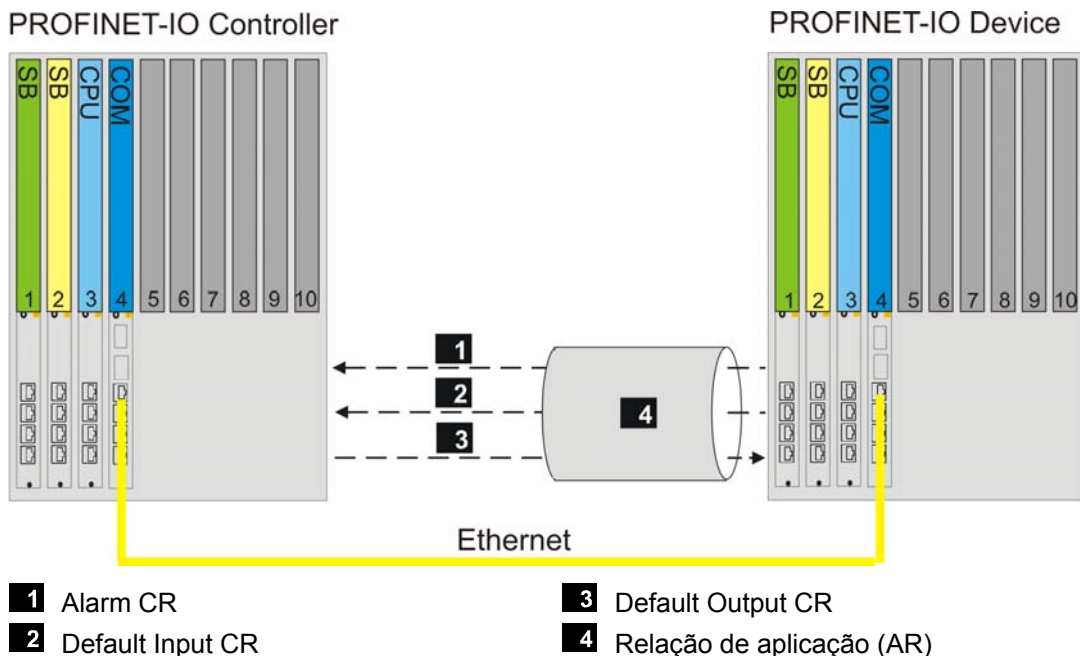


Figura 34: Comunicação via PROFINET/PROFIsafe

A função de menu **Properties** do menu de contexto de *Application Relation* abre o diálogo **Properties**.

Elemento	Descrição
Name	Não pode ser alterado
AR UUID	Número identificador para identificar a relação de aplicação (AR). Não pode ser alterado
Connection Establishment Timeout Factor	Com este parâmetro é calculado o tempo máximo que pode passar do ponto de vista do dispositivo PROFINET-IO ao estabelecer a conexão entre o envio da resposta à requisição de conexão e a recepção de uma nova requisição do controlador PROFINET-IO. Faixa de valores: 1...1000 (x 100 ms) Valor padrão: 600
Supervisor may adopt the AR	Definição se um supervisor PROFINET-IO pode adotar a relação de aplicação (AR). 0 Não permitido 1 Permitido Valor padrão: 0

Tabela 54: Application Relation (características)

## 5.7.10 Alarm CR (características)

Dentro de uma relação de aplicação podem ser estabelecidas várias relações de comunicação (CRs: Communication Relations).

Através da relação de comunicação Alarm CR, um dispositivo PROFINET-IO transmite alarmes ao controlador PROFINET-IO.

A função de menu **Properties** do menu de contexto do Application Relation abre o diálogo **Properties** que contém os seguintes registros:

Elemento	Descrição
Name	Não pode ser alterado
VLAN ID, High Priority	A cada LAN virtual (VLAN) é atribuído um número inequívoco para fins de isolamento. Um equipamento que pertence ao VLAN com o ID = 1, pode comunicar-se com qualquer outro equipamento no mesmo VLAN, porém, não com um equipamento num outro VLAN com o ID = 2, 3, ... Faixa de valores, veja também IEC-61158-6: 0x000 Sem VLAN 0x001 VLAN padrão 0x002 a 0xFFFF Veja IEEE 802.1 Q Valor padrão: 0
VLAN ID, Low Priority	Descrição, veja VLAN ID, High Priority Valor padrão: 0
Alarm Priority	Use user priority A prioridade atribuída pelo usuário é usada. Ignore user priority A prioridade atribuída pelo usuário é ignorada. O alarme gerado sempre é de baixa prioridade.
Alarm Resends	Quantidade máxima de tentativas de envio do dispositivo se o controlador não responder. Faixa de valores 3 a 15 Valor padrão: 10
Alarm Timeout Factor	Com o RTA Timeout Factor calcula-se o tempo de timeout do dispositivo que pode passar no máximo após o envio de um RTA_Data(Alarm) Frame e o recebimento do RTA_ACK Frame. $RTA\ Timeout = RTA\ Timeout\ Factor \times 100\ ms$ Faixa de valores: 1...65535 Valor padrão: 5

Tabela 55: Alarm CR (características)



## 5.7.11 Input CR (características)

Através da relação de comunicação Input CR, um dispositivo PROFINET-IO transmite variáveis ao controlador PROFINET-IO.

A função de menu **Properties** do menu de contexto de Default Input CR abre o diálogo **Properties**. A janela de diálogo contém os seguintes parâmetros:

Elemento	Descrição
Name	Nome livre inequívoco para uma Input CR A Default Input CR não pode ser alterada
Type	1 (não pode ser alterado)
Send Clock Factor	O Send Clock Faktor determina o Send Clock da transmissão de dados cíclica de uma IO-CR. <b>Send Clock = Send Clock Factor x 31,25 µs</b> Faixa de valores: 1...128 Valor padrão: 32
Reduction Factor	O Reduction Factor permite a redução do tempo de ciclo real do envio dos dados de uma IO-CR em relação ao Send Clock. O tempo de ciclo real dos dados calcula-se de: <b>Send Cycle = Reduction Factor x Send Clock</b> Faixa de valores: 1...16384 Valor padrão: 32 (depende do equipamento)
Watchdog Factor	Com o Watchdog factor calcula-se aquele tempo que pode passar no máximo do ponto de vista de um IO-CR-Consumer entre a recepção de dois frames: <b>Watchdog Time = Watchdog Factor x Send Cycle</b> Faixa de valores: 1...7680 Valor padrão: 3
VLAN ID	A cada LAN virtual (VLAN) é atribuído um número inequívoco para fins de isolamento. Um equipamento que pertence ao VLAN com o ID = 1, pode comunicar-se com qualquer outro equipamento no mesmo VLAN, porém, não com um equipamento num outro VLAN com o ID = 2, 3,.. Faixa de valores, veja também IEC-61158-6: 0x000 Sem VLAN 0x001 VLAN padrão 0x002 a 0xFFFF Veja IEEE 802.1 Q Valor padrão: 0

Tabela 56: Input CR (características)

## 5.7.11.1 Input CR (Edit)

A função de menu **Edit** do menu de contexto de Default Input CR abre o diálogo **System Variables** e contém as seguintes variáveis de sistema.

Elemento	Descrição	
Data Status Input CR	Valor	Descrição
	0	State Primary descreve o canal primário em conexões redundantes 1 = Primary 0 = Backup No caso de conexões mono vale: - 1 = Conectado - 0 = Não conectado
	1	Não usado
	2	Data Valid Inválido é atribuído durante a fase de inicializar ou se o aplicativo não consegue comunicar erros pelo IOPS 1 = Valid 0 = Invalid
	3	Não usado
	4	Process State Apenas possui caráter informativo, a validade real dos dados é comunicada via IOPS. 1 = Run 0 = Stop
	5	Problem Indicator No caso de "Problem detected", detalhes sobre os dados de diagnóstico da Alarm CR são disponibilizados. 1 = Regular operation 0 = Problem detected
	6	Não usado
	7	Não usado

Tabela 57: Input CR (Edit)

## 5.7.11.2 Output CR (características)

Dentro de uma relação de aplicação podem ser estabelecidas várias relações de comunicação (CRs: Communication Relations).

Através da relação de comunicação Output CR, um dispositivo PROFINET-IO transmite variáveis ao dispositivo PROFINET-IO.

A função de menu **Properties** do menu de contexto da Output CR abre o diálogo **Properties** que contém os seguintes registros:

Elemento	Descrição
Name	Nome livre inequívoco para uma Output CR A Default Output CR não pode ser alterada
Type	2 (não pode ser alterado)
Send Clock Factor	O Send Clock Factor determina o Send Clock da transmissão de dados cíclica de uma IO-CR. <b>Send Clock = Send Clock Factor x 31,25 µs</b> Faixa de valores: 1...128 Valor padrão: 32
Reduction Factor	Para ajustar a frequência de transmissão. O Reduction Factor permite a redução do tempo de ciclo real do envio dos dados de uma IO-CR. O tempo de ciclo real dos dados calcula-se de: <b>Send Cycle = Reduction Factor x Send Clock</b> Faixa de valores: 1...16384 Valor padrão: 32
Watchdog Factor	Com o Watchdog factor calcula-se aquele tempo que pode passar no máximo do ponto de vista de um IO-CR-Consumer entre a recepção de dois frames: <b>Watchdog Time = Watchdog Factor x Send Cycle</b> Faixa de valores: 1...7680 Valor padrão: 3
VLAN ID	A cada LAN virtual (VLAN) é atribuído um número inequívoco para fins de isolamento. Um equipamento que pertence ao VLAN com o ID = 1, pode comunicar-se com qualquer outro equipamento no mesmo VLAN, porém, não com um equipamento num outro VLAN com o ID = 2, 3,... Faixa de valores, veja também IEC-61158-6: 0x000 Sem VLAN 0x001 VLAN padrão 0x002 a 0xFFFF Veja IEEE 802.1 Q  Valor padrão: 0

Tabela 58: Output CR (características)

## 5.8 HIMA PROFINET-IO Device

Este capítulo descreve as características do HIMA PROFINET-IO Device, bem como as funções de menu e diálogos no SILworX que são necessários para a configuração do HIMA PROFINET-IO Controller.

## 5.9 Requisitos de sistema

Equipamentos necessários e requisitos de sistema:

Elemento	Descrição
Sistema de comando	HIMax com módulo COM HIMatrix L3
Módulo CPU	As interfaces Ethernet do módulo processador não podem ser usadas para PROFINET-IO.
Módulo COM	Ethernet 10/100BaseT
Ativação	A liberação ocorre mediante código de liberação do software, veja Capítulo 3.4.

Tabela 59: Requisitos de sistema e equipamentos necessários para PROFINET-IO Controller

### Propriedades do dispositivo PROFINET-IO:

Elemento	Descrição
Direcionado à segurança	Não
Taxa de transmissão	100 Mbit/s full duplex
Caminho de transporte	Interfaces Ethernet dos módulos COM As interfaces Ethernet em uso podem ser usadas simultaneamente para outros protocolos.
Classe de conformidade	O PROFINET-IO Device satisfaz as exigências da Conformance Class A.
Real Time Class	RT classe 1
Quantidade máx. dispositivos PROFINET-IO	Para cada módulo COM é possível configurar um dispositivo PROFINET-IO.
Quantidade máx. de relações de aplicação (ARs) ao controlador PROFINET IO	Um dispositivo PROFINET-IO pode estabelecer no máx. 5 relações de aplicação (AR) para um controlador PROFINET-IO.
Quantidade relações de comunicação (CRs por AR)	Padrão: 1 Input, 1 Output, 1 Alarm
Comprimento máx. dos dados de processo de todos os módulos PROFINET-IO	Output: máx. 1440 Bytes Input: máx. 1440 Bytes
Priorização dos dados	Possível pelo ajuste do <i>Reduction Rate</i> no nível do dispositivo.
<b>As seguintes propriedades valem para PROFIsafe</b>	
Quantidade máx. F-Devices por módulo COM (HIMax e HIMatrix L3)	63
Comprimento máx. de dados de processo de uma relação de comunicação (CR)	Output: máx. 123 Bytes dados de trabalho + 5 Bytes <sup>1)</sup> Input: máx. 123 Bytes dados de trabalho + 5 Bytes <sup>1)</sup>
Tamanho máx. dados de trabalho HIMax: HIMatrix L3	1024 x 123 Bytes = 125 952 Bytes 512 x 123 Bytes = 62 976 Bytes
<sup>1)</sup> 5 Bytes dados de gestão (Status/Control Bytes e CRC)	

Tabela 60: Características controlador PROFINET-IO

## 5.10 Exemplo PROFINET-IO/PROFIsafe

O seguinte capítulo descreve a configuração do dispositivo HIMA PROFINET-IO/PROFIsafe.

### 5.10.1 Configuração do dispositivo PROFINET-IO no SILworX

**Assim cria-se um novo dispositivo HIMA PROFINET-IO:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Selecionar no menu de contexto de protocolos **New, PROFINET-IO Device**, para adicionar um novo dispositivo PROFINET-IO.
3. Selecionar no menu de contexto do dispositivo PROFINET-IO **Properties**.
4. Introduzir no campo **Name** o nome de equipamento do dispositivo PROFINET-IO.
5. Selecionar **COM-Module**.

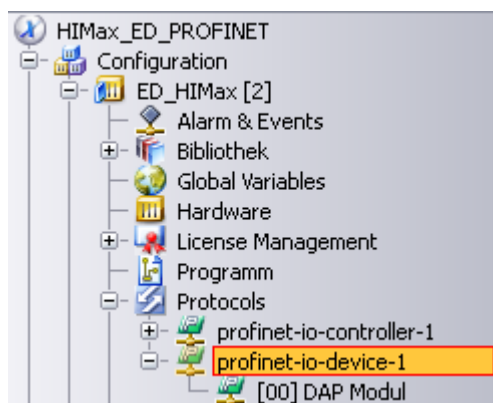


Figura 35: Dispositivo PROFINET-IO na árvore de estrutura HIMax

**Assim criam-se os módulos PROFINET-IO necessários:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFINET-IO Device**.
2. Selecionar no menu de contexto PROFINET-IO Device **New**.
3. Selecionar os seguintes módulos adequados para este exemplo.

PROFINET-IO/PROFIsafe Modul	Slot
In 1 Byte	1
Safe Out 1 Byte	2

**Assim numeram-se os módulos de dispositivo PROFINET-IO:**

1. Clique direito no primeiro **PROFINET-IO Device Module** e selecionar no menu de contexto **Properties**.
2. Introduzir **1** no campo **Slot**.
3. Repetir este passo para os demais **PROFINET-IO device modules** e numerar em sequência.

**i**

**Numerar** os módulos da forma como o dispositivo PROFINET-IO é estruturado de fato.

**O seguinte passo apenas é necessário para os módulos PROFIsafe!**

4. Introduzir no campo *PROFIsafe F\_Destination\_Address* o endereço do módulo PROFIsafe.

## 5.10.1.1 Configuração do módulo de entrada do dispositivo PROFINET-IO

i

A soma das variáveis em Byte deve corresponder ao tamanho do respectivo módulo em Byte.

**Assim configura-se o módulo de entrada [01] In 1 Byte**

1. Selecionar no PROFINET-IO Device o módulo de entrada **[01] In 1 Byte**.
2. Clicar com o botão direito em **[01] In 1 Byte** e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Input Signals**.
5. Clicar com o botão direito em um espaço vazio na área **Input Signals** para abrir o menu de contexto.
6. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.
7. Selecionar no diálogo **Edit** o registro **System Variables** e atribuir uma variável global com o valor inicial TRUE à variável de saída *Input Data Accepted by Device* se controlar o Consumer/Provider Status não for desejado.

i

Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

## 5.10.1.2 Configuração do módulo de saída PROFIsafe-IO Device

**Assim configura-se o módulo de saída [02] Safe Out 1 Byte**

1. Selecionar no PROFIsafe Device o módulo de saída **[02] Safe Out 1 Byte**.
2. Clicar com o botão direito em **[02] Safe Out 1 Byte** e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Output Signals**.
5. Clicar com o botão direito em um espaço vazio na área **Output Signals** para abrir o menu de contexto.
6. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.
7. Selecionar no diálogo **Edit** o registro **System Variables** e atribuir uma variável global com o valor inicial TRUE às variáveis de saída *Valid Output Variable* e *Input Data Accepted by Device* se controlar o Consumer/Provider Status não for desejado.

i

Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

### 5.10.1.3 Verificação da configuração PROFINET-IO Device

**Verificar a configuração PROFINET-IO Device:**

1. Selecionar na árvore de estrutura **Configuration, Resource, Protocols, PROFINET-IO Device**.
2. Clicar no botão **Verification** no Action Bar e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.



A configuração do PROFINET-IO Device deve ser novamente compilada com o programa de aplicação do recurso PROFINET-IO Device e depois carregada nos sistemas de comando para que se torne efetiva para a comunicação PROFINET-IO.

---

## 5.11 Funções de menu do dispositivo PROFINET-IO

### 5.11.1 Função de menu características

A função de menu **Properties** do menu de contexto do dispositivo PROFINET-IO abre o diálogo **Properties**.

Elemento	Descrição
Type	PROFINET-IO Device
Name	Nome livre inequívoco para um dispositivo PROFINET-IO
Process data refresh rate [ms]	Intervalo de atualização em milissegundos com o qual os dados do protocolo são trocados entre COM e CPU. Se o intervalo de atualização for zero ou menor do que o tempo de ciclo do sistema de comando, a troca de dados ocorre o mais rápido possível. Faixa de valores: $4 \dots (2^{31} - 1)$ Valor padrão: 0
Force Process Data Consistency	Ativado: Transferência dos dados completos do protocolo de CPU para COM dentro de um ciclo de CPU. Desativado: Transferência dos dados completos do protocolo de CPU para COM, distribuídos sobre vários ciclos de CPU cada vez 1100 Byte por direção de dados. Com isso, eventualmente também é possível reduzir o tempo de ciclo do sistema de comando. Valor padrão: Ativado
Module	Seleção do módulo COM no qual o protocolo é processado.
Activate Max. $\mu$ P Budget	Ativado: Transferir o limite do $\mu$ P-Budget do campo <i>Max. <math>\mu</math>P-Budget in [%]</i> . Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P-Budget in [%]	Valor máximo de $\mu$ P-Budget do módulo que pode ser produzido ao processar o protocolo. Faixa de valores: 1...100% Valor padrão: 30%
RPC Port Server	Remote Procedure Call Port Faixa de valores: 1024...65535 Valor padrão: 49152 RPC Port Server e RPC Port Client não podem ser idênticos!
RPC Port Client	Remote Procedure Call Port Faixa de valores: 1024...65535 Valor padrão: 49153 RPC Port Server e RPC Port Client não podem ser idênticos!
AutoAwaitF ParamsOn ConnLoss	<b>Este parâmetro apenas é usado para módulos PROFIsafe.</b> Após cada perda de conexão ao F-Host, os F-Parameter devem ser novamente carregados no F-Device. Para facilitar a colocação em funcionamento do PROFIsafe, este parâmetro pode ser ativado. Depois disso, o F-Device adota automaticamente os F-Parameters necessários no caso de reiniciar ou perda de conexão. <b>Depois da colocação em funcionamento, este parâmetro deve ser obrigatoriamente desativado para obter comportamento conforme PROFIsafe.</b> Ativado: O F-Device entra automaticamente no estado <i>Wait for F Parameters</i> Desativado: O F-Device deve ser colocado manualmente no estado <i>Wait for F Parameters</i> pelo usuário, mediante um comando online. Valor padrão: Desativado

Tabela 61: Características gerais do dispositivo PROFINET-IO



## 5.11.2 Módulos HIMA PROFINET-IO

Os seguintes módulos PROFINET-IO estão à disposição no HIMA PROFINET-IO Device.

Módulo PROFINET-IO	Tamanho máx. das variáveis de entrada	Tamanho máx. das variáveis de saída
In 1 Byte	1 Byte	
In 2 Bytes	2 Bytes	
In 4 Bytes	4 Bytes	
In 8 Bytes	8 Bytes	
In 16 Bytes	16 Bytes	
In 32 Bytes	32 Bytes	
In 64 Bytes	64 Bytes	
In 128 Bytes	128 Bytes	
In 256 Bytes	256 Bytes	
In 512 Bytes	512 Bytes	
In 1024 Bytes	1024 Bytes	
In-Out 1 Byte	1 Byte	1 Byte
In-Out 2 Bytes	2 Bytes	2 Bytes
In-Out 4 Bytes	4 Bytes	4 Bytes
In-Out 8 Bytes	8 Bytes	8 Bytes
In-Out 16 Bytes	16 Bytes	16 Bytes
In-Out 32 Bytes	32 Bytes	32 Bytes
In-Out 64 Bytes	64 Bytes	64 Bytes
In-Out 128 Bytes	128 Bytes	128 Bytes
In-Out 256 Bytes	256 Bytes	256 Bytes
In-Out 512 Bytes	512 Bytes	512 Bytes
Out 1 Byte		1 Byte
Out 2 Bytes		2 Bytes
Out 4 Bytes		4 Bytes
Out 8 Bytes		8 Bytes
Out 16 Bytes		16 Bytes
Out 32 Bytes		32 Bytes
Out 64 Bytes		64 Bytes
Out 128 Bytes		128 Bytes
Out 256 Bytes		256 Bytes
Out 512 Bytes		512 Bytes
Out 1024 Bytes		1024 Bytes

Tabela 62: Módulos PROFINET-IO

## 5.11.3 Módulos HIMA PROFIsafe

Os seguintes módulos PROFIsafe estão à disposição no HIMA PROFINET-IO Device.

Módulo PROFIsafe	Tamanho máx. das variáveis de entrada	Tamanho máx. das variáveis de saída
Safe In 1 Byte	1 Byte	
Safe In 2 Bytes	2 Bytes	
Safe In 4 Bytes	4 Bytes	
Safe In 8 Bytes	8 Bytes	
Safe In 16 Bytes	16 Bytes	
Safe In 32 Bytes	32 Bytes	
Safe In 64 Bytes	64 Bytes	
Safe In 123 Bytes	123 Bytes	
Safe In-Out 1 Byte	1 Byte	1 Byte
Safe In-Out 2 Bytes	2 Bytes	2 Bytes
Safe In-Out 4 Bytes	4 Bytes	4 Bytes
Safe In-Out 8 Bytes	8 Bytes	8 Bytes
Safe In-Out 16 Bytes	16 Bytes	16 Bytes
Safe In-Out 32 Bytes	32 Bytes	32 Bytes
Safe In-Out 64 Bytes	64 Bytes	64 Bytes
Safe In-Out 123 Bytes	123 Bytes	123 Bytes
Safe Out 1 Byte		1 Byte
Safe Out 2 Bytes		2 Bytes
Safe Out 4 Bytes		4 Bytes
Safe Out 8 Bytes		8 Bytes
Safe Out 16 Bytes		16 Bytes
Safe Out 32 Bytes		32 Bytes
Safe Out 64 Bytes		64 Bytes
Safe Out 123 Bytes		123 Bytes

Tabela 63: Módulos PROFIsafe

#### Assim cria-se um módulo PROFINet ou PROFIsafe:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFINET-IO Device**.
2. Selecionar no menu de contexto **PROFINET-IO Device Insert Modules**.
3. Selecionar o módulo adequado para transportar os dados de processo exigidos.
4. Clicar com o botão direito no módulo selecionado e selecionar **Edit**.
  - No registro **Process Variables** são introduzidas as variáveis de entrada e/ou saída.
  - Atribuir no registro **System Variables** uma variável global com o valor inicial TRUE às variáveis de saída *Valid Output Variable* e *Input Data Accepted by Device* se controlar o Consumer/Provider Status não for desejado.

·  
1

Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

#### O seguinte ajuste apenas é necessário para módulos PROFIsafe.

- Introduzir no registro **Properties** o *Slot* e o *F\_Destination\_Address Register*.

#### 5.11.4 Módulo PROFINet IO e PROFIsafe

Com os parâmetros dos módulos são ajustadas as relações de comunicação e o comportamento dos módulos no caso de uma interrupção da comunicação.

##### Características

A função de menu **Properties** do menu de contexto dos módulos abre o diálogo **Properties**. A janela de diálogo contém os seguintes registros:

Elemento	Descrição
Name	Nome do módulo de dispositivo
Slot	0 a 32767
Module ID	Número inequívoco
Sub-Slot	Quantidade de subslots
Process data behavior	Valor dos dados de processo após interrupção da conexão - Manter os últimos dados de processo válidos - Adotar dados iniciais
Length of IO Input Data	1...123
Length of IO Output Data	1...123
PROFIsafe F_Destination_Address	O F_Destination_Address do F-Device deve ser inequívoco dentro da rede PROFIsafe! Faixa de valores: 1 a 65534

Tabela 64: Características gerais do módulo Device

##### Edit

A função de menu **Edit** do menu de contexto dos submódulos abre o diálogo **Edit**.

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar o estado do submódulo no programa de aplicação.



Estas variáveis de sistema não podem ser usadas para controlar o Consumer/Provider Status, veja Capítulo 5.2.

Elemento	Descrição
Valid Output Data	True Valid Output Data GOOD False Dados de saída inválidos BAD
Output Data Accepted by Controller	True Valid Output Data GOOD False Dados de saída inválidos BAD
Valid input data	True Dados de entrada válidos GOOD False Dados de entrada inválidos BAD
Input Data Accepted by Device	True Valid input data GOOD False Invalid input data BAD

Elemento	Descrição
<b>As seguintes variáveis apenas estão usadas para módulos PROFIsafe</b>	
PROFIsafe Control	<p>O Control Byte PROFIsafe enviado pelo controlador pode ser lido no dispositivo, veja também Capítulo 5.3.1.</p> <p>Bit 0    iPar_EN_DC Liberação do controlador destrava o dispositivo para carregar novos iParameters para o dispositivo.</p> <p>Bit 1    OA_Req_DC Operator Acknowledge do Host-Control Byte</p> <p>Bit 2    Reset_Comm É o valor Reset_Comm, lido de volta do Host-Control Byte</p> <p>Bit 3    activate_FV_DC FALSE: Valores de processo são trocados entre F-Host e F-Device. TRUE: Failsafe Values "0" são trocados entre o F-Host e o F-Device.</p> <p>Bit 4    Toggle_d Toggle Bit do F-Device</p> <p>Bit 5    Cons_nr_R O Consecutive Number sempre é adotado se o Bit Toggle_d mostrar uma alteração entre dois Control Bytes subsequentes, ou seja, isso independe da ocorrência de um erro.</p> <p>Bit 6    F_ParamValid TRUE: Parametrização F ocorreu FALSE: outro</p> <p>Bit 7    F_Param_ConfiguredTwice TRUE: O F-Device foi parametrizado no mínimo duas vezes com F-Parameters diferentes FALSE: outro</p>
PROFIsafe F_iPar_CRC	<p>iParameters são parâmetros F-Device individuais ou específicos de uma tecnologia. O iPar_CRC resulta da configuração do F-Device.</p> <hr/> <p><b>i</b> Está na responsabilidade do usuário ajustar após a iParametrização o iPar_CRC correto e depois mudar para a operação quente.</p> <hr/>
PROFIsafe F_SIL	<p>0        SIL1</p> <p>1        SIL2</p> <p>2        SIL3</p> <p>3        Sem SIL</p>
PROFIsafe RoundTrip Time last	PROFIsafe precisa determinar o RoundTripTimeLast em um F-Host. Este é o tempo para um F-Host que passa entre o envio de uma mensagem de dados (com consecutive Number N) e a recepção do Acknowledgments correspondente (com Consecutive Number N), medido em milissegundos.

Elemento	Descrição
PROFIsafe Status	<p>PROFIsafe envia do dispositivo com cada mensagem o ROFIsafe Control Byte ao controlador que pode ser atribuído no programa de aplicação.</p> <p>Veja também Capítulo 5.3.1.</p> <p>Bit 0     iPar_OK_DS Novos iParameters recebidos.</p> <p>Bit 1     Device_Fault_DS TRUE: Erro de dispositivo FALSE: Não há erro de dispositivo Apenas é considerado a partir do estado PROFIsafe 21 <i>Await Message</i>.</p> <p>Bit 2     Reserved</p> <p>Bit 3     Reserved</p> <p>Bit 4     FV_activated_DS Fail-safe value ativado</p> <p>Bit 5     Reserved</p> <p>Bit 6     </p> <p>Bit 7     Reset_Comm Protocol Stack é conduzido para o estado inicial.</p>

Tabela 65: Diálogo Edit submódulo

No registro **Process Variables** são introduzidas as variáveis de entrada.

## 6 PROFIBUS DP

PROFIBUS DP é um padrão de barramento de campo internacional aberto e é usado em todos os lugares onde é exigido um tempo de reação rápido com predominantemente volumes reduzidos de dados.

O Master HIMA PROFIBUS DP e o Slave HIMA PROFIBUS DP satisfazem os critérios da norma europeia EN 50170 [7] e da norma mundialmente em vigor para PROFIBUS DP, IEC 61158.

O master HIMA PROFIBUS DP pode trocar dados ciclicamente e aciclicamente com slaves PROFIBUS DP.

Para a troca de dados acíclica, o SILworX disponibiliza diversos blocos funcionais. Com estes blocos funcionais, é possível adaptar o master HIMA PROFIBUS DP e os slaves PROFIBUS DP de forma otimizada às exigências do seu projeto.

Uma conexão redundante PROFIBUS DP apenas pode ser acessada mediante a configuração de um segundo master/slave PROFIBUS DP e adaptações no programa de aplicação.

- Master PROFIBUS DP (Veja Capítulo 6.1)
- Slave PROFIBUS DP (Veja Capítulo 6.13)

## 6.1 Master HIMA PROFIBUS DP

Este capítulo descreve as características do master HIMA PROFIBUS DP, bem como as unções de menu e diálogos no SILworX que são necessários para a configuração do aster HIMA PROFIBUS DP.

### Equipamentos necessários e requisitos de sistema:

Elemento	Descrição
Sistema de comando HIMA	HIMax com módulo COM HIMatrix a partir de CPU OS V7 e COM OS V12
Módulo COM	O módulo COM deve estar equipado com um submódulo master HIMA PROFIBUS DP opcional na interface de barramento de campo serial usada (FB1 ou FB2), veja Capítulo 3.6.
Ativação	Liberação mediante submódulo de barramento de campo, veja Capítulo 3.4.

Tabela 66: Equipamentos necessários e requisitos de sistema

### i

A HIMA recomenda operar HIMax, PROFIBUS DP pela interface de barramento de campo FB1 (taxa de transmissão máxima 12 MBit). Pela interface de barramento de campo FB2 é permitida uma taxa de transmissão máxima 1,5 MBit.

### Propriedades do master PROFIBUS DP:

Elemento	Descrição
Tipo de Master HIMA PROFIBUS DP	DP-V1 Master Classe 1 com funções adicionais DP-V2
Taxa de transmissão	9,6 kbit/s... 12 Mbit/s
Endereço do barramento	0...125
Quantidade máx. masters PROFIBUS DP	Para cada módulo COM HIMax ou HIMatrix F20, F30, F35, F60 podem ser configurados dois master PROFIBUS DP. Por HIMatrix F20, apenas um master PROFIBUS DP.
Quantidade máx. slaves PROFIBUS DP	É possível configurar até 122 slaves para cada recurso (em todas as instâncias de protocolo do master). Aqui incide a restrição que no máximo 31 slaves podem ser conectados a um segmento do barramento sem repetidor.
Comprimento máx. dos dados de processo a um slave	DP-Output =: máx. 244 Bytes DP-Input =: máx. 244 Bytes

Tabela 67: Características do master PROFIBUS DP

Conforme a norma, no total três repetidores são admissíveis, assim, no máximo 122 participantes do barramento por interface serial de um master são possíveis.

### 6.1.1 Criar um master HIMA PROFIBUS DP

#### Assim cria-se um novo master HIMA PROFIBUS DP:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Selecionar no menu de contexto dos protocolos **New, PROFIBUS DP Master** para adicionar um novo master PROFIBUS DP.
3. Selecionar no menu de contexto do master PROFIBUS DP **Properties, General**.
4. Selecionar **Modules** e as **Interfaces**.

6.2 Exemplo PROFIBUS DP

Neste exemplo, um master HIMA PROFIBUS DP troca variáveis com um slave HIMA PROFIBUS DP.

Aqui mostramos como criar e parametrizar o master HIMA PROFIBUS DP e o slave HIMA PROFIBUS DP.

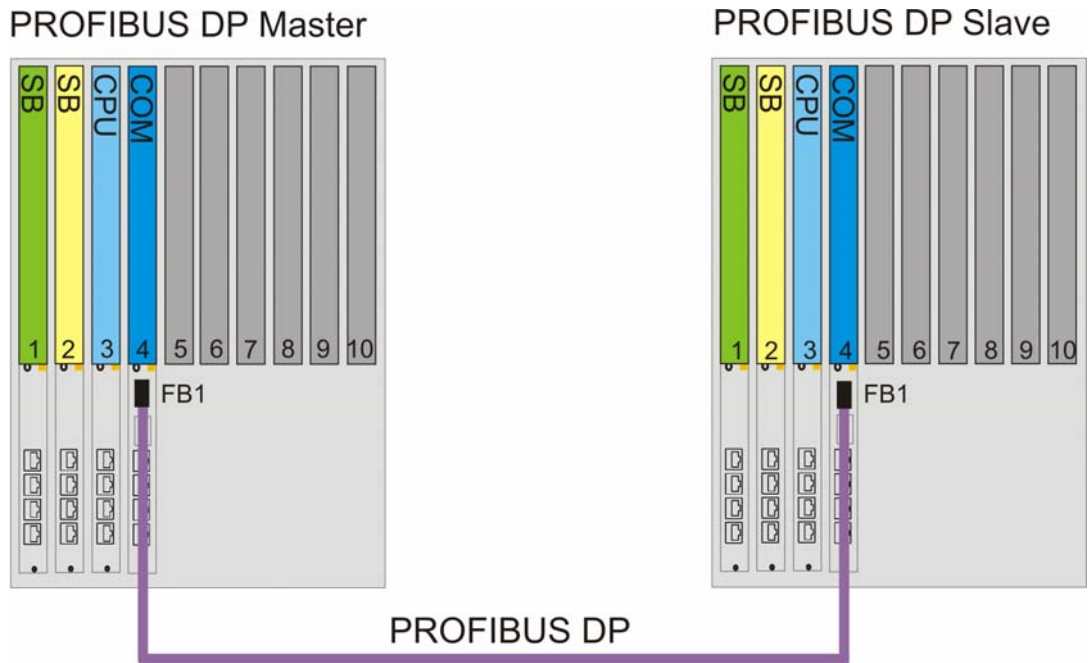


Figura 36: Comunicação via PROFIBUS DP

Os módulos COM dos dois sistemas de comando HIMax devem estar equipados na interface de barramento de campo 1 com o respectivo submódulo PROFIBUS DP, veja Capítulo 3.6.

As seguintes variáveis globais devem ser criadas no SILworX para este exemplo:

Variáveis globais	Tipo
PB_Slave_Master1	UINT
PB_Slave_Master2	DWORD
PB_Slave_Master3	DWORD
PB_Slave_Master4	BYTE
PB_Master_Slave1	DWORD
PB_Master_Slave2	BYTE

6.2.1 Configurar o slave PROFIBUS DP

Configuração do slave PROFIBUS DP.

**Assim cria-se um novo slave HIMA PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Selecionar no menu de contexto dos protocolos **New, PROFIBUS DP Slave** para adicionar um novo slave PROFIBUS DP.
3. Selecionar no menu de contexto do slave PROFIBUS DP **Edit**.
4. Selecionar no registro **Properties** o **COM-Module** e as **Interfaces** (p. ex., FB1).



**Atribuição de variáveis no slave HIMA PROFIBUS DP:**

1. Selecionar no menu de contexto do slave PROFIBUS DP **Edit**.
2. Selecionar no diálogo **Edit** o registro **Process Variables**.



O endereço inicial das variáveis de entrada e saída do slave HIMA PROFIBUS DP sempre começa em 0. Se o master PROFIBUS DP (de um outro fabricante) esperar um endereço inicial mais alto, devem ser inseridas variáveis vazias antes das variáveis de trabalho.

---

**Saídas no slave HIMA PROFIBUS DP**

Nome	Tipo	Offset	Variáveis globais
PB_Slave_Master1	UINT	0	PB_Slave_Master1
PB_Slave_Master2	DWORD	2	PB_Slave_Master2
PB_Slave_Master3	DWORD	6	PB_Slave_Master3
PB_Slave_Master4	BYTE	10	PB_Slave_Master4

Tabela 68: Saídas slave HIMA PROFIBUS DP

1. Na seleção de objetos, puxar a Global Variables para o envio com Drag&Drop para a área **Output Variables**.



As variáveis de saída do slave HIMA PROFIBUS DP neste exemplo consistem em **four variables** com no total **11 Bytes**. A variável de saída com o menor offset possui o endereço inicial **0**.

---

2. Clicar com o botão direito em um espaço vazio na área **Output Variables** para abrir o menu de contexto.
3. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

**Entradas no slave HIMA PROFIBUS DP**

Nome	Tipo	Offset	Variáveis globais
PB_Master_Slave1	DWORD	14	PB_Master_Slave1
PB_Master_Slave2	BYTE	18	PB_Master_Slave2

Tabela 69: Entradas slave HIMA PROFIBUS DP

1. Selecionar na seleção de objetos as variáveis globais para a recepção e puxar mediante Drag&Drop para a área **Input Variables**.



As variáveis de entrada do slave HIMA PROFIBUS DP neste exemplo consistem em **two variables** com no total **5 Bytes**. A variável de entrada com o menor offset possui o endereço inicial **0**.

---

2. Clicar com o botão direito em um espaço vazio na área **Input Variables** para abrir o menu de contexto.
3. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

**Verificar a configuração do slave PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Slave**.
2. Clicar no botão **Verification** no Action Bar e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.



A configuração do slave PROFIBUS DP deve ser novamente compilada com o programa de aplicação do recurso slave PROFIBUS DP e depois carregada nos sistemas de comando para que se torne efetiva para a comunicação PROFIBUS DP.

---

## 6.2.2 Configurar master PROFIBUS DP

**Assim cria-se um novo master HIMA PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Selecionar no menu de contexto dos protocolos **New, PROFIBUS DP Master** para adicionar um novo master PROFIBUS DP.
3. Selecionar no menu de contexto do master PROFIBUS DP **Properties, General**.
4. Selecionar no registro **General** o **COM-Module** e as **Interfaces** (p. ex., FB1).



Os seguintes passos servem para a configuração do slave HIMax PROFIBUS DP no master HIMax PROFIBUS DP.

---

**Assim cria-se um novo slave HIMax PROFIBUS DP no master PROFIBUS DP:**

1. Selecionar no menu de contexto do master PROFIBUS DP **New, PROFIBUS DP Slave**.

**Assim lê-se o arquivo GSD para o novo slave PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, PROFIBUS Slave**.
2. No menu de contexto do master PROFIBUS DP, selecionar **Read GSD File** e selecionar o arquivo GSD correspondente do slave PROFIBUS DP (p. ex., hax100ea.gsd).



Os arquivos GSD para os sistemas de comando HIMax encontram-se na homepage da HIMA [www.hima.com](http://www.hima.com).

---

### 6.2.2.1 Criar os módulos HMax PROFIBUS DP

No master PROFIBUS DP deve ser configurada a quantidade de Bytes a ser transmitida realmente. Isso pode ser alcançado mediante seleção de *Modules*, até alcançar a configuração física do slave.

**i**

Não importa quantos módulos são usados para chegar na quantidade necessária de Bytes enquanto a quantidade máxima de 32 módulos não for ultrapassada.

Para não dificultar a configuração do master PROFIBUS DP desnecessariamente, a quantidade dos módulos selecionados deve ser mantida a menor possível.

#### Assim criam-se os módulos PROFIBUS DP necessários:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, PROFIBUS Slave**.
2. Selecionar na barra de menu **PROFIBUS DP Master, Add modules**.
3. Selecionar os módulos adequados para este exemplo para receber do slave PROFIBUS DP **11 Bytes** e enviar **3 Bytes**.

#### Assim numeram-se os módulos PROFIBUS DP:

1. Clique direito no primeiro **PROFIBUS DP Module** e selecionar no menu de contexto **Properties**.
2. Introduzir **0** no campo **Slot**.
3. Repetir este passo para os demais **PROFIBUS DP Module** e numerar em sequência.

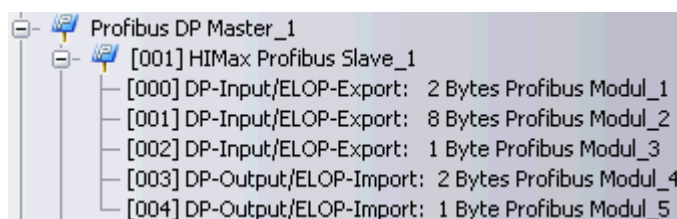


Figura 37: Slave HMax PROFIBUS DP com módulos

**i**

**Numerar** os módulos HMax PROFIBUS DP iniciando com **0** em sequência ascendente e sem lacunas.

A sequência dos módulos PROFIBUS DP não importa para a função, neste caso. Para a visão geral mais clara, porém, os módulos DP-Input e os módulos DP-Output devem ser criados de forma organizada.

## 6.2.2.2 Configuração dos módulos de entrada/saída

i

A soma das variáveis em Byte deve corresponder ao tamanho do respectivo módulo em Byte.

#### Assim configura-se o módulo de entrada [000] DP-Input/ELOP-Export: 2 Bytes:

1. No slave PROFIBUS DP, selecionar o módulo de entrada **[000] DP-Input/ELOP-Export: 2 Bytes**.
2. Clicar com o botão direito em Input Module e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Input Signals** do módulo de entrada **[000] DP-Input/ELOP-Export: 2 Bytes**.

Nome	Tipo	Offset	Variáveis globais
PB_Slave_Master1	UINT	0	PB_Slave_Master1

Tabela 70: Variáveis do módulo de entrada [000] DP-Input/ELOP-Export: 2 Bytes

5. Clicar com o botão direito em um espaço vazio na área **Input Signals** para abrir o menu de contexto.
6. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

#### Assim configura-se o módulo de entrada [001] DP-Input/ELOP-Export: 8 Bytes:

1. No slave PROFIBUS DP, selecionar o módulo de entrada **[001] DP-Input/ELOP-Export: 8 Bytes**.
2. Clicar com o botão direito em Input Module e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Input Signals** do módulo de entrada **[001] DP-Input/ELOP-Export: 8 Bytes**.

Nome	Tipo	Offset	Variáveis globais
PB_Slave_Master2	DWORD	0	PB_Slave_Master2
PB_Slave_Master3	DWORD	4	PB_Slave_Master3

Tabela 71: Variáveis do módulo de entrada [001] DP-Input/ELOP-Export: 8 Bytes

5. Clicar com o botão direito em um espaço vazio na área **Input Signals** para abrir o menu de contexto.
6. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

#### Assim configura-se o módulo de entrada [002] DP-Input/ELOP-Export: 1 Byte:

1. No slave PROFIBUS DP, selecionar o módulo de entrada **[002] DP-Input/ELOP-Export: 1 Byte**.
2. Clicar com o botão direito em Input Module e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Input Signals** do módulo de entrada **[002] DP-Input/ELOP-Export: 1 Byte**.

Nome	Tipo	Offset	Variáveis globais
PB_Slave_Master4	BYTE	0	PB_Slave_Master4

Tabela 72: Variáveis do módulo de entrada [002] DP-Input/ELOP-Export: 1 Byte

5. Clicar com o botão direito em um espaço vazio na área **Input Signals** para abrir o menu de contexto.
6. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

**Assim configura-se o módulo de saída [003] DP-Output/ELOP-Import 2 Bytes:**

1. Selecionar no slave PROFIBUS DP o módulo de saída **[003] DP-Output/ELOP-Import 2 Bytes**.
2. Clicar com o botão direito em Output Module e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Output Signals** do módulo de saída **[003] DP-Output/ELOP-Import 2 Bytes**.

Nome	Tipo	Offset	Variáveis globais
PB_Master_Slave1	UINT	0	PB_Master_Slave1

Tabela 73: Variáveis módulo de saída [003] DP-Output/ELOP-Import: 2 Bytes

5. Clicar com o botão direito em um espaço vazio na área **Output Signals** para abrir o menu de contexto.
6. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

**Assim configura-se o módulo de saída [004] DP-Output/ELOP-Import 1 Bytes:**

1. Selecionar no slave PROFIBUS DP o módulo de saída **[004] DP-Output/ELOP-Import 1 Bytes**.
2. Clicar com o botão direito em Output Module e selecionar no menu de contexto **Edit**.
3. Selecionar no diálogo **Edit** o registro **Process Variables**.
4. Selecionar na seleção de objetos a variável adequada e puxar mediante Drag&Drop para a área **Output Signals** do módulo de saída **[004] DP-Output/ELOP-Import 1 Bytes**.

Nome	Tipo	Offset	Variáveis globais
PB_Master_Slave2	BYTE	0	PB_Master_Slave2

Tabela 74: Variáveis módulo de saída [004] DP-Output/ELOP-Import: 1 Byte

5. Clicar com o botão direito em um espaço vazio na área **Output Signals** para abrir o menu de contexto.
6. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

**6.2.2.3 Criar os dados de usuário no master PROFIBUS DP****Assim criam-se os dados de usuário no master PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, PROFIBUS Slave**.
2. Clicar com o botão direito em **PROFIBUS DP Slave** e selecionar no menu de contexto **Properties**.
3. Selecionar o registro **Data** e clicar o botão **Edit** ao lado dos dados de usuário.

No campo dos dados de usuário com tamanho de 32 Bytes são definidos o *endereço inicial* e a *quantidade de blocos*, veja também Capítulo 6.8.

4. Para este exemplo, devem ser introduzidos os seguintes dados de usuário:
  - 4, para o master PROFIBUS DP receber **four variables**.
  - 2 para o master PROFIBUS DP enviar **two variables**.
  - O endereço inicial do bloco de entrada e saída começa sempre com **0**.

Figura 38: Campo dos dados de usuário

#### Verificar a configuração do slave PROFIBUS DP:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master**.
2. Clicar no botão **Verification** no Action Bar e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.

F	Date/Time	Severity	
2	07/08/2008 14:37:54.828	Error	Please select a interface for the profibus dp master
3	07/08/2008 14:37:54.828	Info	Verification finished. Warnings: 0, Errors 2.

Error: Command failed: 2 error(s).

Figura 39: Janela de diálogo verificação



A configuração do master PROFIBUS DP deve ser novamente compilada com o programa de aplicação do recurso master PROFIBUS DP e depois carregada nos sistemas de comando para que se torne efetiva para a comunicação PROFIBUS DP.

#### 6.2.2.4 Otimizar os parâmetros PROFIBUS DP

Provavelmente, a comunicação PROFIBUS é possível sem problemas com os valores padrão dos parâmetros PROFIBUS. Mediante uma otimização dos ajustes porém, é possível alcançar uma troca de dados mais rápida e uma detecção de erros melhor.

##### Assim determina-se o Target Rotation Time TTR [ms] real

1. Abrir o Control Panel do master HMax PROFIBUS DP.
2. Selecionar na árvore de estrutura do Control Panel **PROFIBUS DP Master** e o **Target Rotation Time TTR [ms]** real. Anotar este valor.

##### Assim determinam-se os parâmetros necessários do slave PROFIBUS DP

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, PROFIBUS DP Slave**.
2. Clicar com o botão direito em **HMax PROFIBUS Slave** e selecionar **Properties**.
3. Selecionar o registro **Features** e ler o parâmetro **Min. Slave Interval MSI [ms]** para este slave PROFIBUS DP. Anotar este valor.
4. Selecionar o registro **Transfer Rate** e ler o parâmetro **Max. Tsdr** para a taxa de transmissão usada. Anotar este valor.

##### Assim são introduzidos os parâmetros determinados

1. Clicar com o botão direito em **PROFIBUS DP Master** e selecionar no menu de contexto **Properties**.
2. Selecionar o registro **Timings**.

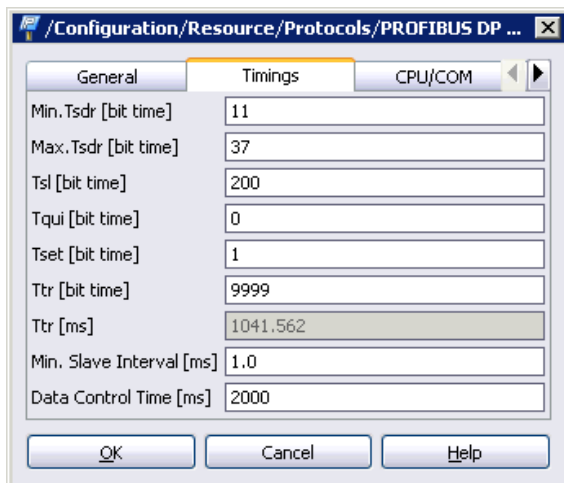


Figura 40: Características do master PROFIBUS DP

3. Converter o valor **Max. Tsdr** em **bit Time** e introduzir.
4. Converter **Target Rotation Time TTR [ms]** em **bit Time** e introduzir com acréscimo de segurança de 1/3 no campo **Target Rotation Time TTR [ms]**.
5. Introduzir **Min. Slave Interval MSI [ms]**.

**i**

Se vários slaves estão configurados, devem ser usados o valor mais alto de MaxTsdr [bit time] e o maior Min. Slave Intervall [ms].

6. O tempo de tempo de supervisão dos dados de trabalho [ms] deve ser ajustado  $\geq 6 \cdot Ttr$  [ms].

**Introduzir o tempo de Watchdog para o slave PROFIBUS-DP**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, PROFIBUS DP Slave**.
2. Clicar com o botão direito em **HIMax PROFIBUS Slave** e selecionar **Properties**.

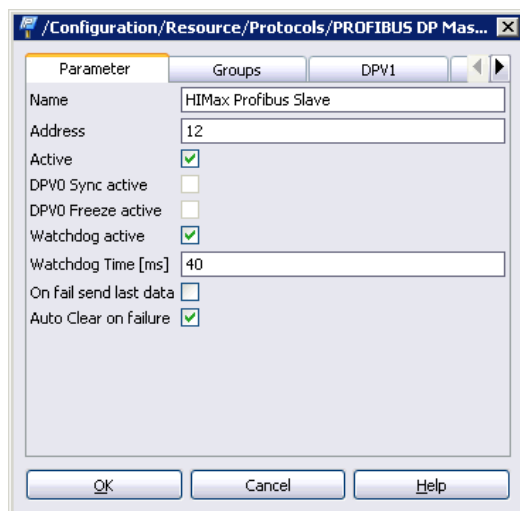


Figura 41: Características do slave PROFIBUS DP

3. Selecionar o registro **Parameter** e colocar o ganchinho na caixa de controle **Watchdog active**.
4. Introduzir tempo de Watchdog [ms]  $\geq 6 \cdot T_{tr}$  [ms] no campo **Watchdog Time [ms]**.

---

**i**

A configuração do master e slave PROFIBUS DP deve ser novamente compilada com o programa de aplicação dos recursos master e slave PROFIBUS DP e depois carregada nos sistemas de comando para que se torne efetiva para a comunicação PROFIBUS DP.

---



### 6.3 Funções de menu do master PROFIBUS DP

#### 6.3.1 Edit

A função de menu **Edit** do menu de contexto do master PROFIBUS DP abre o diálogo **Edit**.

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar o estado do master PROFIBUS DP no programa de aplicação.

Elemento	Descrição																
Number of IO Errors	Quantidade de erros depois de resetar a estatística.																
Baud rate	Baudrate (bit/s) com a qual o barramento é operado.																
Bus Error	<p>Se ocorrer um erro de barramento, na variável de sistema <i>Bus Error</i> é colocado um código de erro. Um código de erro continuá presente até o erro de barramento ser eliminado.</p> <table border="1"> <thead> <tr> <th>Código</th><th>Significado</th></tr> </thead> <tbody> <tr> <td>0</td><td>OK, sem erro de barramento</td></tr> <tr> <td>1</td><td>Erro de endereço: Endereço do master já existe no barramento</td></tr> <tr> <td>2</td><td>Avaria no barramento Uma avaria no barramento foi registrada (p. ex., barramento não corretamente terminado, vários participantes enviam ao mesmo tempo).</td></tr> <tr> <td>3</td><td>Erro de protocolo Pacote com codificação incorreta recebido.</td></tr> <tr> <td>4</td><td>Erro de hardware O hardware comunicou um erro, p. ex., no caso de tempos ajustados muito curtos</td></tr> <tr> <td>5</td><td>Erro desconhecido O master mudou de estado por motivo desconhecido</td></tr> <tr> <td>6</td><td>Controller Reset O chip do controlador é resetado no caso de avarias graves do barramento.</td></tr> </tbody> </table> <p>Para avaliar a variável de status <i>Bus Error</i> no programa de aplicação, a mesma deve ser conectada a uma variável.</p>	Código	Significado	0	OK, sem erro de barramento	1	Erro de endereço: Endereço do master já existe no barramento	2	Avaria no barramento Uma avaria no barramento foi registrada (p. ex., barramento não corretamente terminado, vários participantes enviam ao mesmo tempo).	3	Erro de protocolo Pacote com codificação incorreta recebido.	4	Erro de hardware O hardware comunicou um erro, p. ex., no caso de tempos ajustados muito curtos	5	Erro desconhecido O master mudou de estado por motivo desconhecido	6	Controller Reset O chip do controlador é resetado no caso de avarias graves do barramento.
Código	Significado																
0	OK, sem erro de barramento																
1	Erro de endereço: Endereço do master já existe no barramento																
2	Avaria no barramento Uma avaria no barramento foi registrada (p. ex., barramento não corretamente terminado, vários participantes enviam ao mesmo tempo).																
3	Erro de protocolo Pacote com codificação incorreta recebido.																
4	Erro de hardware O hardware comunicou um erro, p. ex., no caso de tempos ajustados muito curtos																
5	Erro desconhecido O master mudou de estado por motivo desconhecido																
6	Controller Reset O chip do controlador é resetado no caso de avarias graves do barramento.																
Average cycle time	Tempo de ciclo médio medido, em milissegundos.																
Last cycle time	Tempo de ciclo medido, em milissegundos.																
Master State	<p>Mostra o estado atual do protocolo.</p> <p>0: OFFLINE 1: STOP 2: CLEAR 3: OPERATE</p> <p>Para avaliar a variável de status <i>Master State</i> no programa de aplicação, a mesma deve ser conectada a uma variável.</p>																
Maximum Cycle Time	Tempo de ciclo máximo medido, em milissegundos.																
Min Slave Interval	Intervalo slave mínimo medido de um dos slaves atribuídos a este master.																
Minimum Cycle Time	Tempo de ciclo mínimo medido, em milissegundos.																
Target Rotation Time	Tempo projetado para a rotação do token.																

Tabela 75: Variáveis de sistema do master PROFIBUS DP

## 6.3.2 Função de menu características

A função de menu **Properties** do menu de contexto do master PROFIBUS DP abre o diálogo **Properties**.

A janela de diálogo contém os seguintes registros:

## 6.3.2.1 Registro geral

Elemento	Descrição																																												
Type	PROFIBUS DP Master																																												
Name	Nome livre inequívoco para um master PROFIBUS DP																																												
Module	Seleção do módulo COM no qual o protocolo é processado.																																												
Max. $\mu$ P-Budget	Ativado: Transferir o limite do $\mu$ P-Budget do campo <i>Max. <math>\mu</math>P-Budget in [%]</i> . Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.																																												
Max. $\mu$ P-Budget in [%]	Valor máximo de $\mu$ P-Budget do módulo que pode ser produzido ao processar o protocolo.  Faixa de valores: 1...100% Valor padrão: 30%																																												
Behavior on CPU/COM Connection Loss	No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. (P. ex., se o módulo de comunicação é puxado para fora durante a comunicação).  Adopt Initial Data      As variáveis de entrada são resetadas para os valores iniciais.  Retain Last Value      As variáveis de entrada mantém o último valor.																																												
Address	Endereço de estação do master. O endereço de estação do master apenas pode existir uma vez no barramento.  Faixa de valores: 0...125 Valor padrão: 0																																												
Interface	Interface COM que deve ser usada para o master. Faixa de valores: FB1, FB2																																												
Baud rate	Baudrate (Bit/s) com a qual o barramento é operado. Valores possíveis: <table><tr><th>Valor</th><th>Baud Rate</th><th>FB1</th><th>FB2</th></tr><tr><td>9600</td><td>9,6 kbit/s</td><td>X</td><td>X</td></tr><tr><td>19200</td><td>19,2 kbit/s</td><td>X</td><td>X</td></tr><tr><td>45450</td><td>45,45 kbit/s</td><td>X</td><td>X</td></tr><tr><td>93750</td><td>93,75 kbit/s</td><td>X</td><td>X</td></tr><tr><td>187500</td><td>187,5 kbit/s</td><td>X</td><td>X</td></tr><tr><td>500000</td><td>500 kbit/s</td><td>X</td><td>X</td></tr><tr><td>1500000</td><td>1,5 Mbit/s</td><td>X</td><td>X</td></tr><tr><td>3000000</td><td>3 Mbit/s</td><td>X</td><td>-</td></tr><tr><td>6000000</td><td>6 Mbit/s</td><td>X</td><td>-</td></tr><tr><td>12000000</td><td>12 Mbit/s</td><td>X</td><td>-</td></tr></table>	Valor	Baud Rate	FB1	FB2	9600	9,6 kbit/s	X	X	19200	19,2 kbit/s	X	X	45450	45,45 kbit/s	X	X	93750	93,75 kbit/s	X	X	187500	187,5 kbit/s	X	X	500000	500 kbit/s	X	X	1500000	1,5 Mbit/s	X	X	3000000	3 Mbit/s	X	-	6000000	6 Mbit/s	X	-	12000000	12 Mbit/s	X	-
Valor	Baud Rate	FB1	FB2																																										
9600	9,6 kbit/s	X	X																																										
19200	19,2 kbit/s	X	X																																										
45450	45,45 kbit/s	X	X																																										
93750	93,75 kbit/s	X	X																																										
187500	187,5 kbit/s	X	X																																										
500000	500 kbit/s	X	X																																										
1500000	1,5 Mbit/s	X	X																																										
3000000	3 Mbit/s	X	-																																										
6000000	6 Mbit/s	X	-																																										
12000000	12 Mbit/s	X	-																																										

Tabela 76: Características gerais do master PROFIBUS DP

## 6.3.2.2 Registro tempos

Elemento	Descrição
MinTsdr [bit time]	Min. Station Delay Time: Tempo que um slave PROFIBUS DP deve esperar no mínimo antes de poder responder. Faixa de valores: 11...1023 Valor padrão: 11
MaxTsdr [bit time]	Max. Station Delay Time: Tempo que um slave PROFIBUS DP pode demorar no máximo antes de responder. Max Tsdr $\geq$ Tsdr (do Slave conectado com o Tsdr maior) Os valores MaxTsdr dos slaves são lidos dos arquivos GSD e estão no diálogo <b>Properties</b> dos slaves no registro <b>Transfer Rate</b> . Faixa de valores: 37...65535 Valor padrão: 100
Tsl [bit time]	Slot Time Intervalo mínimo de tempo no qual o master aguarda uma resposta do slave. $Tsl > MaxTsdr + 2 \cdot Tset + Tqui + 13$ Faixa de valores: 37...16383 Valor padrão: 200
Tqui [bit time]	Quiet Time for Modulator (Tempo de acalmar o modulador) Tempo que um participante pode demorar para comutar do envio à recepção. Faixa de valores: 0...493 Valor padrão: 0
Tset [bit time]	Setup Time Tempo de reação a um evento Faixa de valores: 1...494 Valor padrão: 1
Ttr [bit time]	Tempo projetado para uma rotação de token Tempo máximo disponibilizado para uma rotação do token. Uma estimativa baixa do Ttr pode ser obtida por um cálculo, veja Capítulo 6.4.4. Faixa de valores: 256...16777215 Valor padrão: 9999
Ttr [ms]	Tempo de rotação do token real em ms
Min. Slave Interval [ms]	Tempo mínimo que deve passar entre duas requisições cíclicas de um slave. Este é respeitado pelo master e nunca pode ser menor. Porém, o ciclo PROFIBUS DP por aumentar se o modo isocrônico não estiver ativo e a proporção de telegramas acíclicos num ciclo aumentar. O valor para <i>Min. Slave Interval</i> do slave é lido do arquivo GSD e está no diálogo <b>Properties</b> do slave, no registro <b>Features</b> . No "Isochronous Mode", <i>Min. Slave Interval</i> define o tempo do ciclo isocrônico. O modo isocrônico é ativado quando Isochronous Sync Mode ou Isochronous Freeze Mode estão ativados. Veja também o tempo de atualização entre CPU e COM (registro CPU/COM). Faixa de valores: 0...65535 (largura do passo 100 $\mu$ s) Valor padrão: 10 (= 1 ms)

Elemento	Descrição
User Data Monitoring Time [ms]	Intervalo de tempo dentro do qual o master deve comunicar o seu estado atual no barramento. Valor de orientação: Tempo de supervisão dos dados de trabalho = WDT do slave Faixa de valores: 0...65535 (largura do passo 10 ms) Valor padrão: 200 (= 2000 ms)

Tabela 77: Registro Tempos no diálogo de características do master PROFIBUS DP

### 6.3.2.3 Registro CPU/COM

Os valores pré-definidos para os parâmetros garantem a troca de dados mais rápida possível dos dados PROFIBUS DP entre o módulo COM (COM) e o módulo CPU (CPU) no sistema de comando HIMax. Estes parâmetros apenas devem ser alterados se uma redução da carga COM e/ou CPU é necessária para uma aplicação e se o processo permitir.

**i** A alteração dos parâmetros apenas se recomenda para programadores experientes. Um aumento do tempo de atualização COM e CPU também significa que o tempo de atualização real dos dados PROFIBUS DP aumenta. Verificar requisições de tempo do sistema.

Observar também o parâmetro *Min. Slave Interval [ms]* que define o tempo de atualização dos dados PROFIBUS DP do/ao slave PROFIBUS DP. Este pode ser aumentado de acordo com o tempo de atualização COM/CPU.

Elemento	Descrição
Process data refresh rate [ms]	Tempo de atualização em milissegundos com o qual os dados do protocolo são trocados entre COM e CPU. Se o <i>Refresh Rate</i> for zero ou menor do que o tempo de ciclo do sistema de comando, a troca de dados ocorre o mais rápido possível. Faixa de valores: 0...( $2^{31} - 1$ ) Valor padrão: 0
Force Process Data Consistency	Ativado: Transferência dos dados completos do protocolo de CPU para COM dentro de um ciclo de CPU.  Desativado: Transferência dos dados completos do protocolo de CPU para COM, distribuídos sobre vários ciclos de CPU cada vez 1100 Byte por direção de dados. Com isso, eventualmente também é possível reduzir o tempo de ciclo do sistema de comando.  Valor padrão: Ativado

Tabela 78: Registro CPU/COM no diálogo de características do master PROFIBUS DP

## 6.3.2.4 Registro Outros

Elemento	Descrição
Max. Number of Resends	Quantidade máxima de repetições de envio de um master quando um slave não responde. Faixa de valores: 0...7 Valor padrão: 1
Highest Active Address	Highest Station Address (HSA) Endereço de estação mais alto a ser esperado de um master. Master com endereços além do HSA não são incluídos no token ring. Faixa de valores: 0...125 Valor padrão: 125
Isochronous Sync Mode	O Isochronous Sync Mode permite uma regulação de ciclo síncrona no master e slave e uma ativação simultânea das saídas físicas de vários slaves. Se o Isochronous Sync Mode estiver ativo, o master envia o comando "Sync" como telegrama broadcast a todos os slaves. Logo que slaves que suportam o modo isocrônico recebem o comando "Sync", comutam os dados do programa de aplicação simultaneamente para as saídas físicas. Os valores das saídas físicas permanecem congelados até o próximo comando Sync. O tempo de ciclo é pré-definido pelo "Min. Slave Interval". Condição: $T_{tr} < \text{Min. Slave Interval}$  Valor padrão: Desativado
Isochronous Freeze Mode	O Isochronous Freeze Mode permite uma transferência simultânea dos dados de entrada de vários slaves. Se o Isochronous Freeze Mode estiver ativo, o master envia o comando "Freeze" como telegrama broadcast a todos os slaves. Logo que os slaves que suportam o modo isocrônico recebem o comando "Freeze", as variáveis das entradas físicas são congeladas no valor momentâneo. Então, os valores podem ser lidos pelo master. Somente após mais um comando "Freeze", os dados de entrada são atualizados. O tempo de ciclo é pré-definido pelo "Min. Slave Interval". Condição: $T_{tr} < \text{Min. Slave Interval}$  Valor padrão: Desativado
Auto Clear on Error	O master entra no estado CLEAR quando um slave falhar no qual Auto Clear no caso de falha estiver ajustado.  Valor padrão: Desativado
Time Master	O master também é master de tempo e envia a hora de sistema periodicamente pelo barramento. Valor padrão: Desativado
Clock Sync Interval [ms]	Intervalo de sincronização do relógio. Intervalo de tempo dentro do qual o master envia o tempo de sistema no barramento. Faixa de valores: 0...65535 (largura do passo 10 ms) Valor padrão: 0 = sem master de tempo

Tabela 79: Outras características do master PROFIBUS DP

## 6.4 Os métodos de acesso ao barramento do PROFIBUS-DP

O método de acesso ao barramento disponibiliza a cada participante do barramento uma janela de tempo definida dentro da qual o participante do barramento deve executar sua tarefa de comunicação.

### 6.4.1 Protocolo Master/Slave

A atribuição do barramento entre um master PROFIBUS-DP e um slave PROFIBUS-DP é garantida pelo método Master/Slave.

Um master PROFIBUS-DP ativo comunica-se com slaves PROFIBUS-DP passivos.

O master PROFIBUS-DP que possui o token está com autorização de envio e pode se comunicar com os slaves PROFIBUS-DP a ele atribuídos. O master atribui o barramento a um slave por determinado tempo dentro do qual o slave precisa responder.

### 6.4.2 Protocolo Token

A atribuição do barramento entre os dispositivos de automação (Master Classe 1) e/ou aparelhos de programação (Master Classe 2) é garantida via Token Passing.

Todos os masters PROFIBUS-DP que estão conectados em conjunto a um barramento formam um Token Ring. O master PROFIBUS-DP ativo que está em posse do token assume neste período a função de master no barramento.

Os master PROFIBUS-DP são ordenados no Token Ring pelos endereços de estação em sequência ascendente e o Token é passado neste ordem até o master PROFIBUS-DP com o endereço de estação mais alto.

Este então passa o Token ao master com o endereço de estação mais baixo para fechar o Token Ring.

O tempo de rotação do Token corresponde a uma rotação do Token por todos os masters PROFIBUS-DP. O Target Rotation Time  $T_{tr}$  é o tempo máximo permitido para uma rotação do token.

### 6.4.3 Tempo de rotação do token ( $T_{tr}$ )

Valores de orientação para diversas taxas de transmissão.

Observar na configuração do master PROFIBUS DP que uma parte dos parâmetros no registro **Timings** depende do Baudrate ajustado no registro **General**. Usar para a primeira configuração (inicial) os valores de orientação indicados na seguinte tabela. Num passo posterior, os valores são otimizados.

	9,6 k	19,2 k	45,45 k	93,75 k	187,5 k	500 k	1,5 M	3 M	6 M	12 M
MinTsdr	11	11	11	11	11	11	11	11	11	11
MaxTsdr	60	60	400	60	60	100	150	250	450	800
Tsl bit time	100	100	640	100	100	200	300	400	600	1000
Tqui bit time	0	0	0	0	0	0	0	3	6	9
Tset bit time	1	1	95	1	1	1	1	4	8	16

Tabela 80: Valores de orientação para o tempo de rotação do token com diferentes taxas de transmissão

Todas as indicações de tempos são dadas em Tbit ( $1\text{Tbit} = 1/[\text{bit/s}]$ ).

MinTsdr possui um comprimento mínimo de 11 Tbit, pois um caracter é composto de 11 Bits, (1 bit inicial, 1 bit de parada, 1 bit de paridade e 8 bits de dados).

Tempo de transmissão para um carácter

Baud Rate	Tbit bit = 1/Baudrate	Tempo
9600 bit/s	$1 / 9600 = 104,166 \mu\text{s}$	$11 * 104,166 \mu\text{s} = 114,583 \mu\text{s}$
6 Mbit/s	$1 / 6 * 1066 = 166,667 \text{ ns}$	$11 * 166,667 \text{ ns} = 1,833 \mu\text{s}$

Tabela 81: Tempo de transmissão para um carácter com diferentes taxas de transmissão

#### 6.4.4 Calcular o tempo de rotação do token $T_{tr}$

Calcular o tempo de rotação do token  $T_{tr}$  mínimo como segue:

$$T_{tr_{\min}} = n * (198 + T_1 + T_2) + b * 11 + 242 + T_1 + T_2 + T_{sl}$$

Elemento	Significado
n	Quantidade de slaves ativos
b	Quantidade de bytes de dados de E/S dos slaves ativos (Input mais Output)
T <sub>0</sub>	$35 + 2 * T_{set} + T_{qui}$
T <sub>1</sub>	Se $T_0 < \text{MinTsdr}$ : $T_1 = \text{MinTsdr}$ Se $T_0 > \text{MinTsdr}$ : $T_1 = T_0$
T <sub>2</sub>	Se $T_0 < \text{MaxTsdr}$ : $T_2 = \text{MaxTsdr}$ Se $T_0 > \text{MaxTsdr}$ : $T_2 = T_0$
T <sub>sl</sub>	Slot Time Intervalo máximo de tempo no qual o master aguarda uma resposta do slave
198	Duas vezes cabeçalho do telegrama com comprimento variável (para requisição e resposta)
242	Global_Control, FDL_Status_Req e passagem de Token

Tabela 82: Elementos para calcular o tempo de rotação do token

**i**

Esta estimativa do tempo de rotação do token  $T_{tr}$  apenas é válida se apenas um master é operado no barramento, se nenhuma transmissão precisa ser repetida e se são transmitidos dados acíclicos.

Nunca ajustar um  $T_{tr}$  menor do que o calculado com a fórmula acima, pois outrossim o funcionamento sem falhas não pode mais ser garantido. A HIMA recomenda ajustar o dobro ou triplo do valor calculado.

#### Exemplo para calcular o tempo de rotação do token $T_{tr}$

A seguinte configuração é assumida:

5 slaves ativos

**(n = 5)**

20 bytes de E/S por Slave

**(b = 100)**

As seguintes constantes de tempo para uma taxa de transmissão de 6 Mbit/s foram retiradas da Tabela 82:

- $\text{MinTsdr} = 11 T_{\text{bit}}$
- $\text{MaxTsdr} = 450 T_{\text{bit}}$
- $T_{\text{sl bit time}} = 600 T_{\text{bit}}$
- $T_{\text{qui bit time}} = 6 T_{\text{bit}}$
- $T_{\text{set bit time}} = 8 T_{\text{bit}}$

$$T0 = 35 + 2 * Tset + Tqui$$

$$T0 = 35 + 2 * 8 + 6$$

$$T0 = 57 T_{bit}$$

**Como  $T0 > MinTsdr$ :  $T1 = T0 = 57 T_{bit}$**

**Como  $T0 < MaxTsdr$ :  $T2 = MaxTsdr = 450 T_{bit}$**

Inserir os valores determinados na fórmula para o tempo de rotação do token mínimo:

$$Ttr_{min} = n * (198 + T1 + T2) + b * 11 + 242 + T1 + T2 + Tsl$$

$$Ttr_{min} = 5 (198 + 57 + 450) + 100 * 11 + 242 + 57 + 450 + 600$$

$$Ttr_{min} [Tbit] = 5974 T_{bit}$$

Resultado:

$$Ttr_{min} [\mu s] = 5974 T_{bit} * 166,67 \text{ ns} = 995,68 \mu s$$

---

## i

*Ttr* é verificado durante a introdução na janela de diálogo.

Se o valor introduzido do *Ttr* for menor do que o calculado pelo SILworX, ocorre uma mensagem de erro no log. Adicionalmente é sugerido um valor mínimo para *Ttr*.

Se *Isochronous Sync Mode* ou *Isochronous Freeze Mode* foram selecionados, o tempo de ciclo é determinado pelo parâmetro *MinSlaveInterval*. Neste caso, o *Ttr* de qualquer forma deve ser menor do que o *Minimum Slave Interval*.

Se não respeitar esta condição no Iso Mode, uma mensagem de erro é causada.

---



### 6.5 Ciclo isocrônico PROFIBUS DP (a partir de DP-V2)

O ciclo PROFIBUS DP aqui consiste numa parte fixa, cíclica do telegrama e uma parte causada pelo evento, acíclica.

A parte acíclica do telegrama num ciclo PROFIBUS DP pode aumentar a duração do mesmo proporcionalmente, o que é indesejável em algumas aplicações, p. ex., na técnica de acionamentos.

Para alcançar um tempo de ciclo constante ( $t_{\text{const}}$ ), é ativado o modo isocrônico no master no qual o parâmetro *Min. Slave Interval [ms]* define o tempo de ciclo constante ( $t_{\text{const}}$ ). O ciclo de PROFIBUS DP isocrônico assim programado possui uma precisão de ciclo com um desvio de < 10 ms.

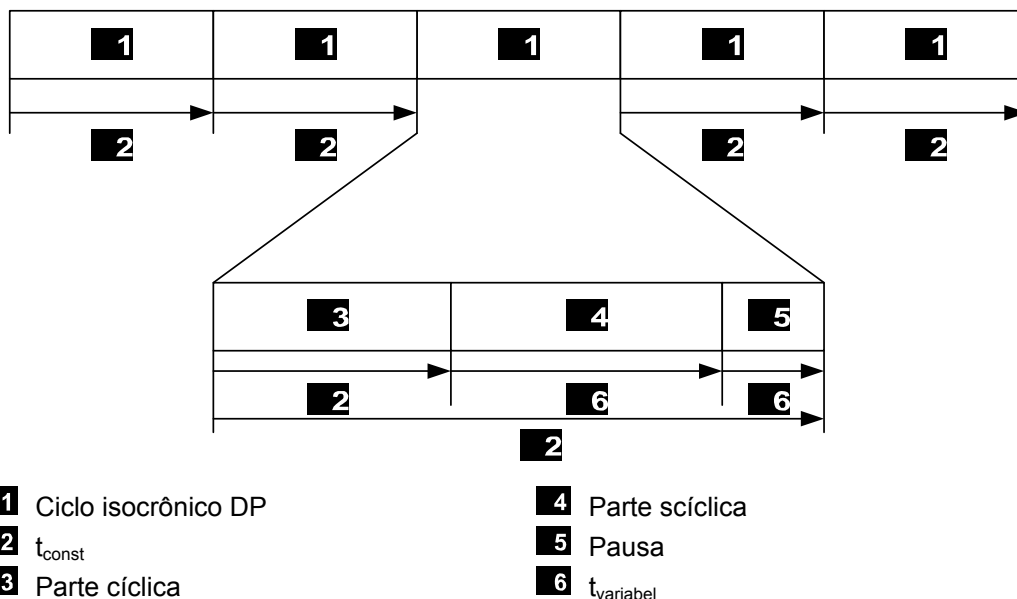


Figura 42: Ciclo isocrônico PROFIBUS DP

Para determinar a parte acíclica, o usuário deve calcular o tempo de rotação do token mínimo.

Adicionalmente, deve ser reservado um intervalo de tempo suficientemente grande para a parte acíclica (tipicamente duas a três vezes o tempo de rotação do token mínimo  $T_{tr}$ ). Se o tempo reservado não for necessário, é inserida uma pausa antes do próximo ciclo para manter o tempo de ciclo constante (veja também Capítulo 6.4.3, tempo de rotação do token  $T_{tr}$ ).

#### i

O master é parametrizado através de *Min. Slave Interval [ms]* com o tempo de ciclo DP determinado pelo usuário.

Para que o modo isocrônico *Isochronous Mode* se torne efetivo, deve ser ativado no master no mínimo um dos dois parâmetros *Isochronous Sync Mode* ou *Isochronous Freeze Mode*.

No barramento apenas pode ser operado um master no modo isocrônico. Outros masters não são permitidos.

### 6.5.1 Isochronous Mode (a partir de DP-V2)

Esta função permite uma regulação sincronizada do ciclo no master e slave, independente da carga do barramento. O ciclo do barramento é sincronizado com um desvio de ciclo de < 10 ms. Assim, processos de posicionamento de alta precisão podem ser realizados.

---

#### i

As vantagens do *Isochronous Mode* também podem ser usadas de forma restrita por slaves (DP-V0-Slaves) que não apoiam o *Isochronous Mode*. Para este fim, ativa-se nos *Slaves Sync* e/ou *Freeze* e os atribui ao grupo 8.

Tipicamente usa-se o modo *Sync* e *Freeze* simultaneamente.

---

### 6.5.2 Isochronous Sync Mode (a partir de DP-V2)

O *Isochronous Mode Sync* permite uma regulação de ciclo sincrônica no master e slave e uma ativação simultânea das saídas físicas de vários slaves.

### 6.5.3 Isochronous Freeze Mode (a partir de DP-V2)

O *Isochronous Freeze Mode* permite uma transferência simultânea dos dados de entrada de vários slaves.

## 6.6 Funções de menu do slave PROFIBUS DP (no master)

### 6.6.1 Criar um slave PROFIBUS DP (no master)

**Assim cria-se um novo slave PROFIBUS DP no master HIMA PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master**.
2. Selecionar no menu de contexto do master PROFIBUS DP **New, PROFIBUS DP Slave** para adicionar um novo slave PROFIBUS DP.

### 6.6.2 Edit

A função de menu **Edit** do menu de contexto do slave PROFIBUS DP abre o diálogo **System Variables**.

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar o estado do slave PROFIBUS DP no programa de aplicação.

Elemento	Descrição	
Activation control	Mudança de 0 para 1 desativa o slave. Mudança de 1 para 0 ativa o slave anteriormente desativado. Ativado = 0 Desativado = 1	
PNO Ident Number	Número de 16 Bits atribuído pela Associação PROFIBUS DP que identifica um produto (equipamento de campo) de forma inequívoca.	
Standard Diagnosis	Pelo diagnóstico padrão o slave comunica ao master o seu estado atual. Esta variável sempre contém o último diagnóstico padrão. Os parâmetros correspondem ao telegrama de diagnóstico conforme IEC 61158.	
Connection Count	É incrementado com cada nova conexão. Conta a partir do reset do contador	
Connection State	Valor	Descrição
	0	Desativado: Os conjuntos de parâmetros para estes slaves são carregados, os slaves, porém, são completamente ignorados. Os dados de entrada são colocados nos seus valores iniciais, no barramento não aparece nenhuma atividade em relação a estes slaves.
	1	Inativo (não conectado): Se um slave não pode (mais) ser ativado, os dados de entradas são colocados nos valores iniciais. Para cada slave, é possível selecionar as seguintes opções: <ul style="list-style-type: none"> <li>▪ O master continua enviando dados de saída ao slave ou</li> <li>▪ O master tenta parametrizar o slave de novo.</li> </ul>
	2	Ativo (conectado): Os slaves trocam dados de E/S com a CPU
Counter Slave Alarm	Quantidade de alarmes produzidos até então. Conta a partir do reset do contador.	
Standard Diagnosis Count	Quantidade de mensagens de diagnóstico produzidas até então. Conta a partir do reset do contador.	

Tabela 83: Variáveis de sistema do slave PROFIBUS DP

## 6.6.3 Características

A função de menu **Properties** do menu de contexto do slave PROFIBUS DP abre o diálogo **Properties**. A janela de diálogo contém os seguintes registros:

## 6.6.3.1 Registro Parâmetros

Elemento	Descrição
Name	Nome do slave
Address	Endereço do slave Faixa de valores: 0...125 Valor padrão: 0
Active	Estado do slave Apenas um slave ativo pode comunicar-se com um master PROFIBUS DP. Valor padrão: Ativado
DP V0 Sync active	O <i>Sync Mode</i> permite uma uma ativação simultânea das saídas de vários DP-V0-Slaves. <b>Atenção:</b> No caso de DP-V2-Slaves que trabalham no <i>Isochronous Sync Mode</i> , este campo deve estar desativado. Valor padrão: Desativado
DP V0 Freeze active	O <i>Freeze Mode</i> permite uma transferência simultânea dos dados de entrada de vários DP-V0-Slaves. <b>Atenção:</b> No caso de DP-V2-Slaves que trabalham no <i>Isochronous Freeze Mode</i> , este campo deve estar desativado. Valor padrão: Desativado
Watchdog active	Quando selecionado, o slave reconhece a falha de um master e entra um estado seguro. Valor padrão: Desativado
Watchdog Time [ms]	A caixinha de controle Watchdog deve estar ativada. Se dentro deste período de tempo não ocorreu a troca de dados entre o master e slave, o slave desliga e reseta todos os dados de saída DP para o seu valor inicial. 0 = Desativado Valor de orientação: Tempo de Watchdog do slave > 6 * T <sub>tr</sub> Faixa de valores: 0...65535 Valor padrão: 0
On failure send last data	FALSE: No caso de erro, a conexão é interrompida e novamente estabelecida. TRUE: No caso de erro continua enviando dados, mesmo sem confirmação do slave. Valor padrão: Desativado
Auto Clear on Failure	Se <i>Auto Clear on Failure</i> estiver ajustado para TRUE no mastes e neste slave, o master comuta o PROFIBUS DP inteiro para o estado seguro no caso de uma falha deste slave. Valor padrão: Ativado

Tabela 84: Registro Parâmetros do slave PROFIBUS DP

### 6.6.3.2 Registro Grupos

Neste registro é possível organizar os slaves em diversos grupos. Os comandos do tipo *GlobalControl Sync* e *Freeze* podem abordar de forma direcionada um ou vários grupos neste caso.

Elemento	Descrição	
Membro no grupo 1	Membro no grupo 1	Valor padrão: Desativado
Membro no grupo 2	Membro no grupo 2	
Membro no grupo 3	Membro no grupo 3	
Membro no grupo 4	Membro no grupo 4	
Membro no grupo 5	Membro no grupo 5	
Membro no grupo 6	Membro no grupo 6	
Membro no grupo 7	Membro no grupo 7	
Membro no grupo 8	Membro no grupo 8	

Tabela 85: Registro Grupos no diálogo de características do slave PROFIBUS DP

### 6.6.3.3 Registro DP-V1

Neste registro há parâmetros que apenas são definidos a partir de DP-V1. No caso de slaves DP-V0, não é possível selecionar nada aqui. Quais parâmetros são suportados pelo slave pode ser verificado na coluna Suportado.

Elemento	Descrição
DP-V1	Se o modo DP-V1 não estiver ativado, não é possível usar as outras funções DP-V1. O slave se comporta como um slave DP-V0, neste caso. Eventualmente, também os dados de parametrização devem ser alterados neste caso (veja manual do slave) Valor padrão: Desativado
Failsafe	Com este modo ativado, o master não envia zeros como dados de saída ao slave no estado CLEAR, mas um pacote de dados vazio, (pacote failsafe). Isso sinaliza ao slave que agora deve colocar os dados de saída seguros nas saídas (que não necessariamente são todos zero). Valor padrão: Desativado
Isochronous Mode	Esta função permite uma regulação sincronizada do ciclo no master e slave, independente da carga do barramento. O ciclo do barramento é sincronizado com um desvio de ciclo de < 1 ms. Assim, processos de posicionamento de alta precisão podem ser realizados. Valor padrão: Desativado
Publisher active	Esta função é necessária para a comunicação transversal de slave. Isso permite a comunicação direta e rápida entre os slaves via Broadcast, sem o desvio pelo master. Valor padrão: Desativado
Prm Block Struct. Supp.	O slave suporta dados de parametrização estruturados (apenas leitura). Valor padrão: Desativado
Check. Cfg.-Mode	Controle de configuração reduzido se "Check Cfg.-Mode" estiver ativado; neste caso o slave pode ser operado sem a configuração completa. Para a colocação em funcionamento, este campo deve ser desativado. Valor padrão: Desativado

Tabela 86: Registro DP-V1 no diálogo de características do slave PROFIBUS DP

## 6.6.3.4 Registro Alarmes

Nesta tela podem ser ativados alarmes. Porém, isso apenas funciona com slaves DP-V1, se DP-V1 estiver ativado e o slave suportar alarmes. Quais alarmes são suportados pode ser verificado pelo ganchinho na coluna **Supp.** Se um alarme estiver prescrito, isso pode ser verificado na coluna **Required.**

Elemento	Descrição	
Update Alarm	Alarme se parâmetros do módulo mudaram.	Valor padrão: Desativado
Status Alarm	Alarme se o estado do módulo mudou.	
Vendor Alarm	Alarme específico do fabricante.	
Diagnostic Alarm	Alarme com determinados eventos, como curto circuito, sobretemperatura etc. num módulo.	
Process Alarm	Alarme no caso de eventos importantes no processo.	
Pull & Plug Alarm	Alarme ao encaixar ou retirar um módulo.	

Tabela 87: Registro Alarmes no diálogo de características do slave PROFIBUS DP

## 6.6.3.5 Registro Dados

Neste registro encontram-se informações sobre os comprimentos de dados suportados, bem como os dados de usuário (dados de parametrização expandidos).

Elemento	Descrição
Max. Input Len	Comprimento máximo dos dados de entrada.
Max. Output Len	Comprimento máximo dos dados de saída.
Max. Data Len	Comprimento máximo total dos dados de dados de entrada e saída.
User Data Len	Tamanho dos dados de usuário.
User Data	Dados de parametrização. Não se recomenda efetuar a alteração aqui. É mais confortável com o diálogo <i>Edit User Parameter</i> , Capítulo 6.8.
Max. Diag. Data Len	Comprimento máximo dos dados de diagnóstico enviados pelo slave.

Tabela 88: Registro Dados no diálogo de características do slave PROFIBUS DP

## 6.6.3.6 O registro Model

Nesta tela estão várias informações que se explicam por si só:

Elemento	Descrição
Model	Denominação do fabricante do modelo de slave PROFIBUS DP
Manufacturer	Fabricante do equipamento de campo
Ident. Number	Identificação do slave da Associação PROFIBUS (PNO)
Revision	Estado da revisão do slave PROFIBUS DP
Hardware-Release	Estado da revisão do hardware do slave PROFIBUS DP
Software-Release	Estado da revisão do software do slave PROFIBUS DP
GSD file name	Nome do arquivo GSD
Infotext	Informação adicional sobre o slave PROFIBUS DP

Tabela 89: Registro Model no diálogo de características do slave PROFIBUS DP

## 6.6.3.7 Registro Features

Elemento	Descrição
Modularstation	TRUE: Estação modular FALSE: Estação compacta
First slot number	A numeração dos módulos (slots) deve iniciar com este número e seguir na sequência.
Max. Modules	Quantidade máxima de módulos que uma estação modular pode receber.
Supported for "Set Slave Add"	O slave suporta a atribuição dinâmica de endereços.
Min. Slave Interval [ms]	Tempo mínimo que deve passar entre duas requisições cíclicas de um slave.
Diag. Update	Quantidade de ciclos de interrogação até o diagnóstico do slave refletir o estado atual.
Support for WDBase1ms	O slave suporta 1 ms como base de tempo para o Watchdog.
Support for DP-V0 Sync	O slave suporta DP-V0 Sync
Support for DP-V0 Freeze	O slave suporta DP-V0 Freeze
DP-V1 Datatypes	O slave suporta os tipos de dados DP-V1.
Extra Alarm SAP	O slave suporta SAP 50 para a confirmação de alarmes.
Alarm Seq. Mode Count	Indica quantos alarmes ativos o slave pode processar simultaneamente. Zero precisa um alarme de cada tipo.

Tabela 90: Registro Features no diálogo de características do slave PROFIBUS DP

## 6.6.3.8 Registro Taxas de transmissão

Neste registro, encontram-se as taxas de transmissão que o slave suporta, bem como o respectivo *MaxTsdr*.

*MaxTsdr* é o intervalo de tempo no qual o slave precisa responder a uma requisição do master, o mais tardar. A faixa de valores depende do slave e da velocidade de transmissão e está entre 15 e 800 Tbit.

Elemento	Descrição
9,6 k	MaxTsdr = 60
19,2 k	MaxTsdr = 60
31,25 k	Não é suportado.
45,45 k	MaxTsdr = 60
93,75 k	MaxTsdr = 60
187,5 k	MaxTsdr = 60
500 k	MaxTsdr = 70
1,5 M	MaxTsdr = 75
3 M	MaxTsdr = 90
6 M	MaxTsdr = 100
12 M	MaxTsdr = 120

Tabela 91: Registro Taxas de transmissão no diálogo de características do slave PROFIBUS DP

## 6.6.3.9 Registro Acíclico

Neste registro encontram-se alguns parâmetros para uma transmissão de dados acíclica:

Elemento	Descrição
Support for C1 Read/Write	Slave suporta transmissão de dados acíclica.
C1 Read/Write-required	Slave exige transmissão de dados acíclica.
C1 Max Data Len [Byte]	Comprimento máximo do pacotes de dados acíclico.
C1 Response Timeout [ms]	Timeout para transmissão de dados acíclica.

Tabela 92: Registro Acíclico no diálogo de características do slave PROFIBUS DP



## 6.7 Ler o arquivo GSD

O arquivo GSD contém os dados para a parametrização do slave PROFIBUS DP.

**Assim lê-se o arquivo GSD para o novo slave PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, PROFIBUS Slave**.
2. No menu de contexto do master PROFIBUS DP, selecionar **Read GSD File** e selecionar o arquivo GSD correspondente do slave PROFIBUS DP (p. ex., hax100ea.gsd).

### i

Os arquivos GSD para os sistemas de comando HIMax/HIMatrix encontram-se na homepage da HIMA [www.hima.com](http://www.hima.com).

O respectivo fabricante do equipamento de campo correspondente é responsável pela correção do arquivo GSD.

Nos arquivos GSD para HIMax (hax100ea.gsd) e HIMatrix (hix100ea.gsd), encontram-se os seguintes módulos:

Módulos de entrada master PROFIBUS DP	Tipo	Quantidade
DP-Input/ELOP-Export	Byte	1
DP-Input/ELOP-Export	Bytes	2
DP-Input/ELOP-Export	Bytes	4
DP-Input/ELOP-Export	Bytes	8
DP-Input/ELOP-Export	Bytes	16
DP-Input/ELOP-Export	Word	1
DP-Input/ELOP-Export	Words	2
DP-Input/ELOP-Export	Words	4
DP-Input/ELOP-Export	Words	8
DP-Input/ELOP-Export	Words	16
Módulos de saída master PROFIBUS DP	Tipo	Quantidade
DP-Output/ELOP-Import	Byte	1
DP-Output/ELOP-Import	Bytes	2
DP-Output/ELOP-Import	Bytes	4
DP-Output/ELOP-Import	Bytes	8
DP-Output/ELOP-Import	Bytes	16
DP-Output/ELOP-Import	Word	1
DP-Output/ELOP-Import	Words	2
DP-Output/ELOP-Import	Words	4
DP-Output/ELOP-Import	Words	8
DP-Output/ELOP-Import	Words	16

Tabela 93: Arquivo GSD do Slave HIMax PROFIBUS DP

## 6.8 Editar os parâmetros de usuários

No campo dos dados de usuário são definidos o **endereço inicial** e a **quantidade de variáveis** dos blocos.

Adicionalmente, no master PROFIBUS DP deve ser configurada a quantidade de Bytes a ser transmitida realmente. Isso ocorre mediante a seleção dos módulos PROFIBUS DP do arquivo GSD do slave PROFIBUS DP (veja também Capítulo 6.2.2).

**Assim, abre-se o diálogo Edit User Parameters:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master**.
2. Clicar com o botão direito em **PROFIBUS DP Slave** e selecionar **Properties**.
3. Selecionar o registro **Data** e clicar o **botão ...** ao lado dos dados de usuário.

---

### i

A estrutura do diálogo **Edit User Parameters** é definida pelo arquivo GSD do slave.

---

#### Princípio de estrutura do campo de dados de usuário de 32 Bytes:

O campo de dados de usuário de 32 Bytes possui a seguinte estrutura:

Os 32 Bytes estão em **eight groups**, com **four bytes per group** respectivamente.

Os blocos 1..4 definem quais e quantas variáveis o master PROFIBUS DP recebe do slave PROFIBUS DP.

Os blocos 5..8 definem quais e quantas variáveis o master PROFIBUS DP envia ao slave PROFIBUS DP.

Os primeiros dois bytes de cada bloco especificam o endereço inicial para a primeira variável a ser lida ou escrita de um bloco.

Os últimos dois bytes de cada bloco especificam a quantidade de variáveis que devem ser recebidas ou enviadas.

#### Configuração dos dados de usuário em diferentes blocos:

Normalmente não é necessário distribuir as variáveis (dados de usuário) em diferentes blocos. É o suficiente apenas definir sempre o primeiro bloco de variável das variáveis de entrada e saída e ler e escrever os dados *em bloco*.

Em aplicações onde é necessário apenas ler ou escrever algumas variáveis selecionadas, é possível definir até quatro blocos de variáveis para as variáveis de saída e entrada.

#### Exemplo

O master PROFIBUS DP envia e recebe as seguintes variáveis do slave PROFIBUS DP:

1º Bloco: **4** variáveis de entrada a partir do endereço inicial **0**.

2º Bloco: **6** variáveis de entrada a partir do endereço inicial **50**.

4º Bloco: **9** variáveis de entrada a partir do endereço inicial **100**.

5º Bloco: **2** variáveis de saída a partir do endereço inicial **10**.

**Configuração dos dados de usuário no master PROFIBUS DP:**

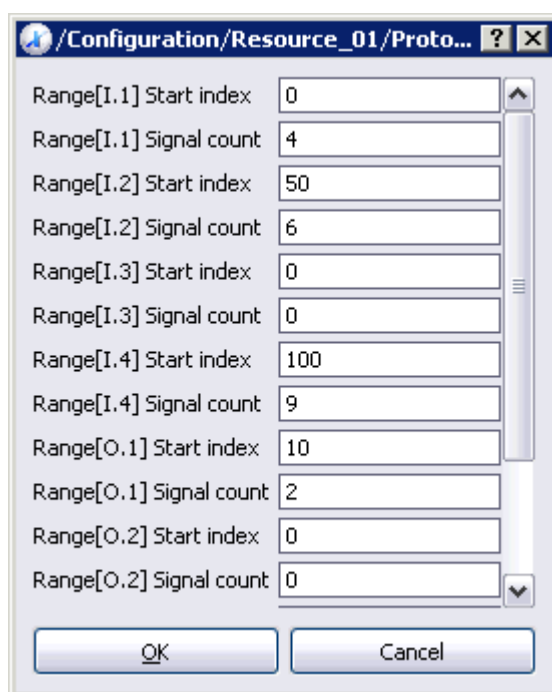
Master Import/Slave Export	Endereço inicial	Qtde. variáveis
1º Bloco (Byte 0..3)	0,0	0,4
2º Bloco (Byte 4..7)	0,50	0,6
3º Bloco (Byte 8..11)	0,0	0,0
4º Bloco (Byte 12..15)	0,100	0,9

Tabela 94: Exemplo: Blocos 1..4 do campo dos dados de usuário

Master Export/Slave Import	Endereço inicial	Qtde. variáveis
5º Bloco (Byte 16..19)	0,10	0,2
6º Bloco (Byte 20..23)	0,0	0,0
7º Bloco (Byte 24..27)	0,0	0,0
8º Bloco (Byte 28..31)	0,0	0,0

Tabela 95: Exemplo: Blocos 5..8 do campo dos dados de usuário

Diálogo *Edit User Parameters* de um slave PROFIBUS DP HIMatrix ou HIMax.

Figura 43: Diálogo *Edit User Parameters*

## 6.9 Blocos funcionais PROFIBUS

Com os blocos funcionais PROFIBUS, é possível adaptar o master HIMA PROFIBUS DP e os slaves PROFIBUS DP a ele atribuídos de forma otimizada às exigências do seu projeto.

Os blocos funcionais parametrizam-se no programa de aplicação, assim que as funções do master e dos slaves (alarmes, dados de diagnóstico, estados) possam ser atribuídos e lidos no programa de aplicação.

- 
- i** Os blocos funcionais apenas são necessários para aplicações especiais. Para a troca normal cíclica entre master e slave, estes blocos funcionais não são necessários!
- Princípios gerais de configuração dos blocos funcionais PROFIBUS DP, veja Capítulo 13.1.
- 

Os seguintes blocos funcionais estão à disposição:

Bloco funcional	Descrição da função	Adequado a partir do nível de desempenho DP
MSTAT 6.9.1	Controlar o estado do master pelo programa de aplicação	DP-V0
RALRM 6.9.2	Ler mensagens de alarme dos slaves	DP-V1
RDIAG 6.9.3	Ler mensagens de diagnóstico dos slaves	DP-V0
RDREC 6.9.4	Ler os conjuntos de dados acíclicos dos slaves	DP-V1
SLACT 6.9.5	Controlar o estado dos slaves pelo programa de aplicação	DP-V0
WRREC 6.9.6	Escrever os conjuntos de dados acíclicos dos slaves	DP-V1

Tabela 96: Visão geral – Blocos funcionais PROFIBUS DP

- 
- i** Os masters HIMA PROFIBUS DP trabalham com o nível de desempenho DP-V1.  
Os slaves HIMA PROFIBUS DP trabalham com o nível de desempenho DP-V0.  
Observar que desta forma, nem todos os blocos funcionais do master HIMA PROFIBUS DP podem controlar um slave HIMA PROFIBUS DP.
-

6.9.1 Bloco funcional MSTAT

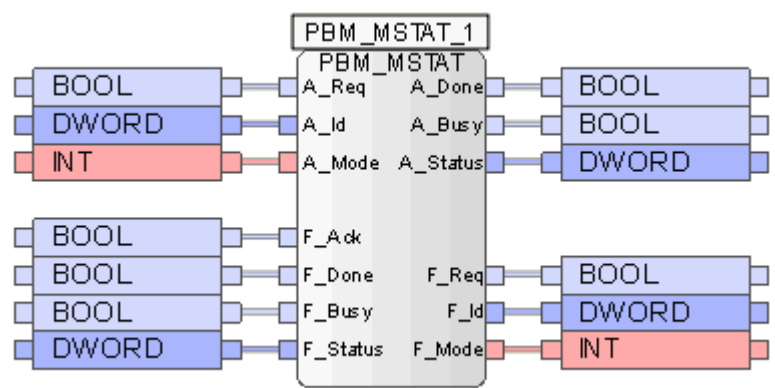


Figura 44: Bloco funcional MSTAT

Com o bloco funcional **MSTAT** (a partir de DP-V0), o master PROFIBUS DP pode ser controlado pelo programa de aplicação. Assim, mediante um interruptor mecânico numa entrada física ou através de um Timer, é possível colocar o master PROFIBUS DP em um dos seguintes estados:

- 0: OFFLINE
- 1: STOP
- 2: CLEAR
- 3: OPERATE

**i** Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Flanco positivo inicia o bloco	BOOL
A_ID	Master ID (não utilizado)	DWORD
A_Mode	O master PROFIBUS DP pode ser colocado nos seguintes estados 0: OFFLINE 1: STOP 2: CLEAR 3: OPERATE	INT

Tabela 97: Entradas A bloco funcional MSTAT

Saídas A	Descrição	Type
A_Done	TRUE: O master PROFIBUS DP foi colocado no estado definido na entrada A_Mode.	BOOL
A_Busy	TRUE: A atribuição do master PROFIBUS DP ainda não terminou.	BOOL
A_Status	Status ou código de erro (veja Capítulo 6.11)	DWORD

Tabela 98: Saídas A bloco funcional MSTAT

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional MSTAT na árvore de estrutura. O prefixo "F" significa "Field" – Campo.

**i**

A conexão do bloco funcional MSTAT na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional MSTAT (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **MSTAT** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **MSTAT** na árvore de estrutura também.

Entradas F	Tipo
F_ACK	BOOL
F_DONE	BOOL
F_BUSY	BOOL
F_STATUS	DWORD

Tabela 99: Entradas F bloco funcional MSTAT

Conectar as *F-Outputs* do bloco funcional **MSTAT** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **MSTAT** na árvore de estrutura também.

Saídas F	Tipo
F_REQ	BOOL
F_ID	DWORD
F_MODE	INT

Tabela 100: Saídas F bloco funcional MSTAT

**Assim, cria-se o bloco funcional MSTAT na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, Function Blocks, New**.
2. Selecionar o bloco funcional **MSTAT**.
3. Clicar com o botão direito no bloco funcional **MSTAT** e selecionar **Edit**.  
☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **MSTAT** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **MSTAT** no programa de aplicação.

Entradas	Tipo
M_ID	DWORD
MODE	INT
REQ	BOOL

Tabela 101: Variáveis de sistema de entrada

Conectar as saídas do bloco funcional do bloco funcional **MSTAT** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **MSTAT** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
DONE	BOOL
STATUS	DWORD

Tabela 102: Variáveis de sistema de saída

**Assim, opera-se o bloco funcional MSTAT:**

1. Colocar no programa de aplicação a entrada *A\_Mode* para o estado desejado.  
Se não for atribuído *A\_Mode*, será emitido depois do passo 2 um código de erro na saída *A\_Status* e o master PROFIBUS DP não é atribuído.
2. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.

---

**i**

O bloco funcional reage a uma mudança de flanco positiva em *A\_Req*.

- 
- ☒ A saída *A\_Busy* está TRUE até o comando MSTAT estiver processado. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Done* para TRUE.

---

**i**

Se não foi possível atribuir o modo definido, um código de erro é emitido na saída *A\_Status*.

O modo atual do master pode ser conferido na variável Master Status (veja Capítulo 6.10).

---

6.9.2 Bloco funcional RALRM

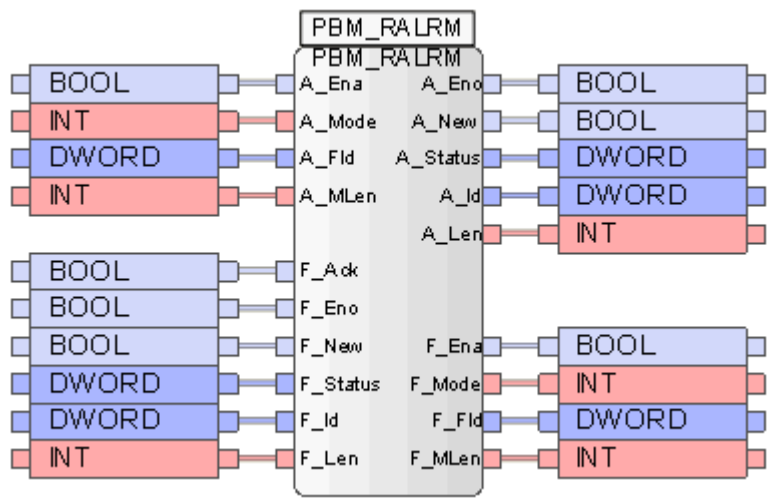


Figura 45: Bloco funcional RALRM

O bloco funcional **RALRM** (a partir de DP-V1) é utilizado para a avaliação de alarmes.

Alarmes são um tipo especial de mensagens de diagnóstico que são tratadas com prioridade. Alarmes comunicam eventos importantes à aplicação que exigem reações da aplicação (p. ex., um WRREC). Porém, isso depende do fabricante e pode ser consultado no manual de operação do slave PROFIBUS DP.

Enquanto o bloco funcional **RALRM** estiver ativo ele aguarda mensagens de alarme dos slaves. Ao receber um alarme, a saída **A\_NEW** é comutada para TRUE por pelo menos um ciclo e os dados do alarme podem ser lidos pelo telegrama de alarme. Antes do próximo alarme, **A\_NEW** entra em FALSE por pelo menos um ciclo. Todos os alarmes são confirmados implicitamente. Nenhum alarme é perdido.

No caso da utilização de vários blocos funcionais **RALRM**, o programa de aplicação deve ser criado de forma que sempre só um bloco funcional **RALRM** esteja ativo.

**i** Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Ena	Com TRUE, o bloco funcional é liberado	BOOL
A_Mode	Não usado	INT
A_FID	Não usado	DWORD
A_MLen	Comprimento máximo esperado em bytes dos dados do alarme a serem recebidos.	INT

Tabela 103: Entradas A bloco funcional RALRM



Saídas A	Descrição	Type
A_Eno	TRUE: Bloco funcional ativo FALSE: Bloco funcional não ativo	BOOL
A_New	TRUE: Um novo alarme foi recebido FALSE: Nenhum alarme novo	BOOL
A_Status	Status ou código de erro (veja Capítulo 6.11)	DWORD
A_ID	Número de identificação do slave que disparou o alarme	DWORD
A_Len	Comprimento em bytes dos dados do alarme recebidos	INT

Tabela 104: Saídas A bloco funcional RALRM

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional RALRM na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **RALRM** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **RALRM** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **RALRM** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **RALRM** na árvore de estrutura também.

Entradas F	Tipo
F_ACK	BOOL
F_ENO	BOOL
F_NEW	BOOL
F_STATUS	DWORD
F_ID	DWORD
F_LEN	INT

Tabela 105: Entradas F bloco funcional RALRM

Conectar as *F-Outputs* do bloco funcional **RALRM** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **RALRM** na árvore de estrutura também.

Saídas F	Tipo
F_Ena	BOOL
F_MODE	INT
F_FID	DWORD
F_MLEN	INT

Tabela 106: Saídas F bloco funcional RALRM

**Assim, cria-se o bloco funcional RALRM na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, Function Blocks, New**.
2. Selecionar o bloco funcional **RALRM**.
3. Clicar com o botão direito no bloco funcional **RALRM** e selecionar **Edit**.
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **RALRM** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **RALRM** no programa de aplicação.

Entradas	Tipo
EN	BOOL
F_ID	DWORD
MLEN	INT
MODE	INT

Tabela 107: Variáveis de sistema de entrada

Conectar as saídas do bloco funcional do bloco funcional **RALRM** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **RALRM** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
ENO	BOOL
ID	DWORD
LEN	INT
NEW	BOOL
STATUS	DWORD

Tabela 108: Variáveis de sistema de saída

No registro “variáveis de processo” do bloco funcional **RALRM** na árvore de estrutura devem ser definidas variáveis cuja estrutura é compatível com os dados do alarme. Se não são definidas variáveis, os dados do alarme podem ser requisitados, mas não lidos.

Uma mensagem de alarme contém no mínimo quatro bytes. Os primeiros quatro bytes da mensagem de alarme contém os dados do alarme padrão.

Para a avaliação facilitada dos alarmes padrão, a HIMA disponibiliza o bloco funcional auxiliar **ALARM** (veja Capítulo 6.10). Se quiser utilizar este bloco, junte os primeiros quatro bytes numa variável do tipo DWORD e colocar esta variável na entrada *IN* do bloco funcional auxiliar **ALARM**.

---

i

Se um telegrama de alarme tiver mais bytes do que definido no registro “Dados”, apenas a quantidade definida de bytes é transferida. O resto é cortado.

---

Dados de alarme	Descrição
Byte 0	Comprimento da mensagem de alarme em bytes (4..126)
Byte 1	Identificação do tipo de alarme 1: Alarme diagnóstico 2: Alarme de processo 3: Alarme de retirar módulo 4: Alarme de encaixar módulo 5: Alarme de status 6: Alarme de Update 31: Falha da extensão de um master ou slave 32..126: Específico do fabricante: O significado deve ser consultado na descrição do fabricante do equipamento.
Byte 2	Número de slot do componente que disparou o alarme
Byte 3	0: Sem outra informação 1: Alarme de chegada, slot avariado 2: Alarme de saída, slot não está mais avariado 3: Alarme de saída, slot continua avariado
Byte 4 a 126	O significado deve ser consultado na descrição do fabricante do equipamento.

Tabela 109: Dados de alarme

---

*i*

A estrutura dos alarmes padrão (bytes 0..3) é normada e idêntica para todos os fabricantes. Para os bytes 4..126 utilizados de forma específica pelos fabricantes, consultar o manual de operação do slave PROFIBUS DP.

Observar que equipamentos de acordo com o padrão DP-V0 não suportam telegramas de alarme.

---

**Assim, opera-se o bloco funcional RALRM:**

1. No programa de aplicação, definir na entrada *AMlen* a quantidade máxima de dados de alarme em bytes a esperar. *A\_Mlen* não pode ser alterado durante a operação.
  2. Colocar no programa de aplicação a entrada *A\_Ena* em TRUE.
- 

*i*

Ao contrário de outros blocos funcionais, o bloco funcional **RALRM** só está ativo enquanto a entrada *A\_Ena* estiver TRUE.

---

Se o bloco foi iniciado com êxito, a saída *A\_Eno* assume o valor TRUE. Se não foi possível iniciar o bloco, um código de erro é emitido na saída *A\_Status*.

Ao receber um alarme novo, a saída *A\_New* é comutada para TRUE por pelo menos um ciclo. Durante este tempo, as saídas contêm os dados do alarme do slave que disparou o alarme que podem ser avaliados.

Depois disso, a saída *A\_New* é comutada para FALSE novamente por pelo menos um ciclo. As saídas *A\_ID* e *A\_Len* são resetadas a zero antes que a próxima mensagem de alarme possa ser recebida e avaliada.

## 6.9.3 Bloco funcional RDIAG

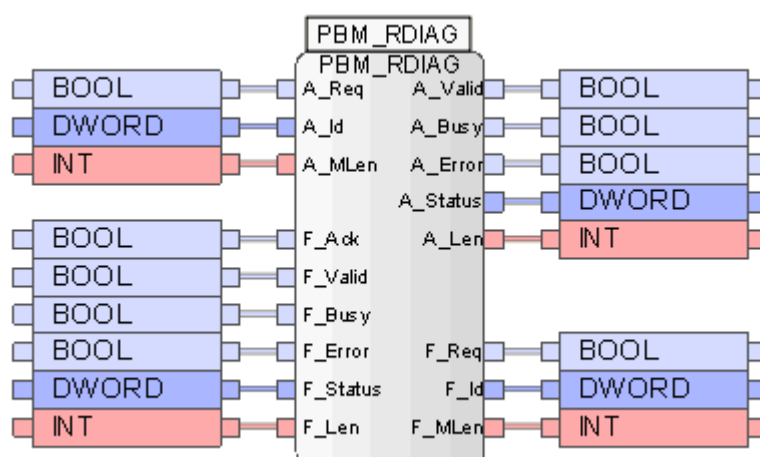


Figura 46: Bloco funcional RDIAG

O bloco funcional **RDIAG** (a partir de DP-V0) é utilizado para ler a mensagem de diagnóstico atual (6 Byte...240 Bytes) de um slave.

No master HIMA PROFIBUS DP, o número de blocos **RDIAG** ativos simultaneamente é livre.

## i

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Um flanco positivo inicia a requisição de uma mensagem de diagnóstico	BOOL
A_ID	Número de identificação do slave (veja Capítulo 6.10)	DWORD
A_MLen	Comprimento máximo esperado em bytes dos dados de alarme a serem lidos.	INT

Tabela 110: Entradas A bloco funcional RALRM

Saídas A	Descrição	Type
A_Valid	Uma nova mensagem de diagnóstico foi recebida e é válida	BOOL
A_Busy	TRUE: A leitura ainda não terminou	BOOL
A_Error	TRUE: Durante a leitura ocorreu um erro	BOOL
A_Status	Status ou código de erro (veja Capítulo 6.11)	DWORD
A_Len	Comprimento dos dados de diagnóstico em bytes	INT

Tabela 111: Saídas A bloco funcional RALRM

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **RDIAG** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **RDIAG** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **RDIAG** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **RDIAG** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **RDIAG** na árvore de estrutura também.

Entradas F	Tipo
F_ACK	BOOL
F_VALID	BOOL
F_BUSY	BOOL
F_ERROR	BOOL
F_Status	DWORD
F_LEN	INT

Tabela 112: Entradas F bloco funcional RDIAG

Conectar as *F-Outputs* do bloco funcional **RDIAG** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **RDIAG** na árvore de estrutura também.

Saídas F	Tipo
F_Req	BOOL
F_Id	DWORD
F_MLen	INT

Tabela 113: Saídas F bloco funcional RDIAG

**Assim, cria-se o bloco funcional RDIAG na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, Function Blocks, New**.
2. Selecionar o bloco funcional **RDIAG**.
3. Clicar com o botão direito no bloco funcional **RDIAG** e selecionar Edit.
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **RDIAG** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **RDIAG** no programa de aplicação.

Entradas	Tipo
ID	DWORD
MLEN	INT
REQ	BOOL

Tabela 114: Variáveis de sistema de entrada

Conectar as saídas do bloco funcional do bloco funcional **RDIAG** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **RDIAG** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
ERROR	BOOL
LEN	INT
Status	DWORD
VALID	BOOL

Tabela 115: Variáveis de sistema de saída

### Dados de diagnóstico

No registro **Data** devem ser definidas variáveis cuja estrutura deve ser compatível com os dados de diagnóstico. Uma mensagem de diagnóstico contém no mínimo seis bytes e no máximo 240 bytes. Os primeiros quatro bytes da mensagem de diagnóstico contêm os dados do diagnóstico padrão.

Para a avaliação facilitada do diagnóstico padrão, a HIMA disponibiliza o bloco funcional auxiliar **STDDIAG** (veja Capítulo 6.10). Se quiser utilizar este bloco, junte os primeiros quatro bytes numa variável do tipo DWORD e colocar esta variável na entrada *IN* do bloco funcional auxiliar **STDDIAG**.

i

Se um telegrama de diagnóstico tiver mais bytes do que definido no registro “Dados”, apenas a quantidade definida de bytes é transferida. O resto é cortado

Dados de diagnóstico	Descrição
Byte 0	Os bytes 0..3 contêm o diagnóstico padrão. O diagnóstico padrão pode ser decodificado como variável do tipo DWORD com ajuda do bloco funcional auxiliar <b>STDDIAG</b> .
Byte 1	
Byte 2	
Byte 3	Endereço de barramento do master ao qual um slave está atribuído.
Byte 4	High-Byte (identificação do fabricante)
Byte 5	Low-Byte (identificação do fabricante)
Byte 6...240	O significado deve ser consultado na descrição do fabricante do equipamento

Tabela 116: Dados de diagnóstico

i

Os slaves HIMA produzem um telegrama de diagnóstico de um comprimento de seis bytes. O significado destes bytes é padronizado.

Para os slaves de outros fabricantes, apenas os primeiros seis bytes são funcionalmente idênticos.

Informações mais detalhadas sobre o telegrama de diagnóstico podem ser consultadas na descrição do fabricante do slave.

### Assim, opera-se o bloco funcional **RDIAG**:

1. Colocar no programa de aplicação o endereço do slave na entrada *A\_ID*.
2. No programa de aplicação, definir na entrada *A\_Mlen* a quantidade máxima de dados de alarme em bytes a esperar.
3. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.

---

**i**

O bloco funcional reage a uma mudança de flanco positiva em *A\_Req*.

---

A saída *A\_Busy* está em TRUE até a requisição diagnóstico terminar de ser processada. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Valid* ou *A\_Error* para TRUE.

Se o telegrama de diagnóstico estiver válido, a saída *A\_Valid* passa para TRUE. Os dados de diagnóstico podem ser avaliados mediante as variáveis definidas no registro Dados. A saída *A\_Len* contém a quantidade de dados de diagnóstico que realmente foram lidos em bytes.

Se não foi possível ler o telegrama de diagnóstico com êxito, a saída *A\_Error* muda para TRUE e um código de erro é emitido na saída *A\_Status*.

## 6.9.4 Bloco funcional RDREC

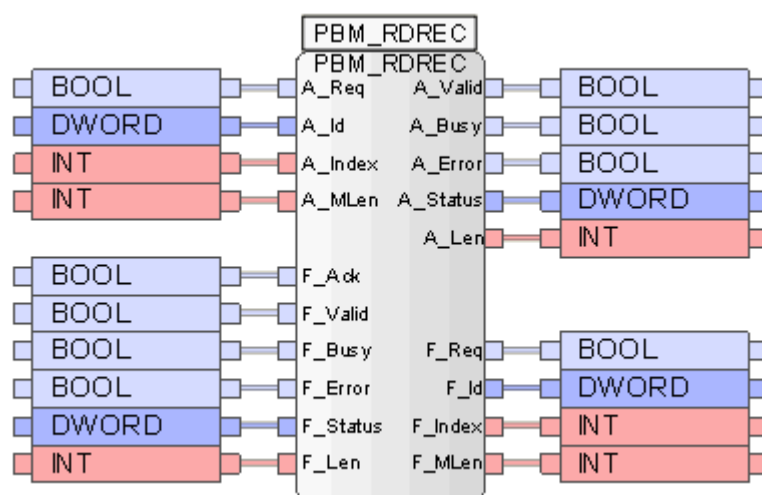


Figura 47: Bloco funcional RDREC

O bloco funcional **RDREC** é utilizado para a leitura acíclica de um conjunto de dados de um slave, endereçado na entrada *A\_Index*. Deve ser consultado do manual de operação do slave quais dados foram lidos.

Esta função está definida apenas a partir de DP-V1 e é opcional.

No master HIMA PROFIBUS DP, até 32 blocos **RDREC** e/ou **WRREC** podem ser ativos simultaneamente.

## i

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Um flanco positivo inicia a requisição de leitura.	BOOL
A_Id	Número de identificação do slave, veja Capítulo 6.10	DWORD
A_Index	Número do conjunto de dados do conjunto de dados a ser lido. O significado deve ser consultado na descrição do fabricante do equipamento.	INT
A_MLen	Comprimento máximo em bytes dos dados a serem lidos.	INT

Tabela 117: Entradas A bloco funcional RDREC

Saídas A	Descrição	Type
A_Valid	Um novo conjunto de dados foi recebido e é válido.	BOOL
A_Busy	TRUE: O processo de leitura ainda não terminou.	BOOL
A_Error	TRUE: Ocorreu um erro FALSE: Sem erro	BOOL
A_Status	Status ou código de erro, veja Capítulo 6.11	DWORD
A_Len	Comprimento da informação do conjunto de dados em bytes.	INT

Tabela 118: Saídas A bloco funcional RDREC



### Entradas e saídas do bloco funcional com o prefixo F

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **RDREC** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **RDREC** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **RDREC** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **RDREC** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **RDREC** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Valid	BOOL
F_Busy	BOOL
F_Error	BOOL
F_Status	DWORD
F_Len	INT

Tabela 119: Entradas F bloco funcional RDREC

Conectar as *F-Outputs* do bloco funcional **RDREC** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **RDREC** na árvore de estrutura também.

Saídas F	Tipo
F_Req	BOOL
F_Id	DWORD
F_Index	INT
F_Mlen	INT

Tabela 120: Saídas F bloco funcional RDREC

### Assim, cria-se o bloco funcional RDREC na árvore de estrutura:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, Function Blocks, New**.
2. Selecionar o bloco funcional **RDREC**.
3. Clicar com o botão direito no bloco funcional **RDREC** e selecionar **Edit**.  
☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **RDREC** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **RDREC** no programa de aplicação.

Entradas	Tipo
ID	DWORD
INDEX	INT
MLEN	INT
REQ	BOOL

Tabela 121: Variáveis de sistema de entrada

Conectar as saídas do bloco funcional do bloco funcional **RDREC** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **RDREC** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
ERROR	BOOL
LEN	INT
STATUS	DWORD
VALID	BOOL

Tabela 122: Variáveis de sistema de saída

Dados	Descrição
Sem variáveis pré-definidas	No registro <i>Process Variables</i> , é possível definir livremente uma estrutura de dados, mas ela deve ser compatível com a estrutura do conjunto de dados. A estrutura do conjunto de dados deve ser consultada no manual de operação do fabricante do slave.

Tabela 123: Dados

#### Assim, opera-se o bloco funcional RDREC:

1. Colocar no programa de aplicação o endereço do slave na entrada *A\_ID*.
2. Colocar no programa de aplicação o index específico do slave para o conjunto de dados (manual do fabricante) na entrada *A\_Index*.
3. Colocar no programa de aplicação o comprimento do conjunto de dados a ser lido na entrada *A\_Len*.
4. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.

### i

O bloco funcional reage a uma mudança de flanco positiva em *A\_Req*.

A saída *A\_Busy* está em TRUE até a requisição do conjunto de dados terminar de ser processada. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Valid* ou *A\_Error* para TRUE.

Se o conjunto de dados estiver válido, a saída *A\_Valid* passa para TRUE. O conjunto de dados pode ser avaliado mediante as variáveis definidas no registro Dados. A saída *A\_Len* contém o comprimento do conjunto de dados realmente lido em bytes.

Se não foi possível ler o conjunto de dados com êxito, a saída *A\_Error* muda para TRUE e um código de erro é emitido na saída *A\_Status*.

## 6.9.5 Bloco funcional SLACT

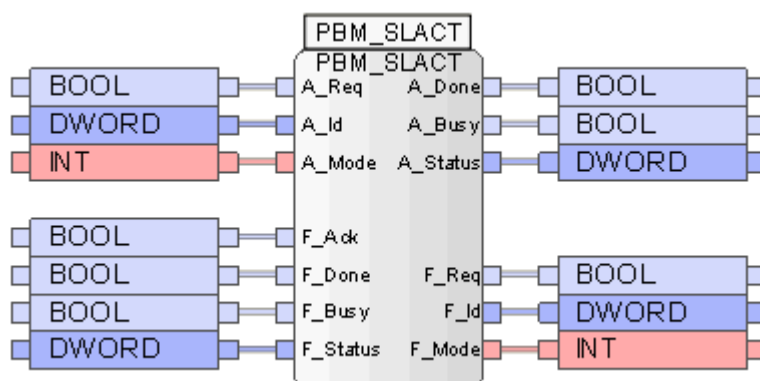


Figura 48: Bloco funcional SLACT

O bloco funcional **SLACT** (a partir de DP-V0) é utilizado para ativar e desativar um slave do programa de aplicação do master PROFIBUS DP. Assim, mediante um interruptor mecânico numa entrada física no master PROFIBUS DP ou através de um Timer, é possível colocar o slave em um dos seguintes estados.

≠ 0: Ativo

= 0: Não ativo

No caso da utilização de vários blocos funcionais **SLACT**, o programa de aplicação deve ser criado de forma que sempre só um bloco funcional **SLACT** esteja ativo.

### i

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

#### Entradas e saídas do bloco funcional com o prefixo A:

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Flanco positivo inicia o bloco	BOOL
A_Id	Número de identificação do slave (veja Capítulo 6.10)	DWORD
A_Mode	Estado no qual o slave PROFIBUS DP deve ser colocado: ≠ 0: Ativo (conectado) = 0: Não ativo (desativado)	INT

Tabela 124: Entradas A bloco funcional SLACT

Saídas A	Descrição	Type
A_Done	TRUE: O slave PROFIBUS DP foi colocado no estado definido na entrada A_Mode.	BOOL
A_Busy	TRUE: A atribuição do slave PROFIBUS DP ainda não terminou.	BOOL
A_Status	Status ou código de erro (veja Capítulo 6.11)	DWORD

Tabela 125: Saídas A bloco funcional SLACT

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **SLACT** na árvore de estrutura. O prefixo "F" significa "Field" – Campo.

**i**

A conexão do bloco funcional **SLACT** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **SLACT** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **SLACT** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **SLACT** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Done	BOOL
F_Busy	BOOL
F_Status	DWORD

Tabela 126: Entradas F bloco funcional **SLACT**

Conectar as *F-Outputs* do bloco funcional **SLACT** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **SLACT** na árvore de estrutura também.

Saídas F	Tipo
F_Req	BOOL
F_Id	DWORD
F_Mode	INT

Tabela 127: Saídas F bloco funcional **SLACT**

**Assim, cria-se o bloco funcional **SLACT** na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, Function Blocks, New**.
2. Selecionar o bloco funcional **SLACT**.
3. Clicar com o botão direito no bloco funcional **SLACT** e selecionar **Edit**.  
☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **SLACT** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **SLACT** no programa de aplicação.

Entradas	Tipo
ID	DWORD
MODE	INT
REQ	BOOL

Tabela 128: Variáveis de sistema de entrada

Conectar as saídas do bloco funcional do bloco funcional **SLACT** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **SLACT** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
DONE	BOOL
STATUS	DWORD

Tabela 129: Variáveis de sistema de saída

**Assim, opera-se o bloco funcional SLACT:**

1. Colocar no programa de aplicação a entrada *A\_Mode* no estado desejado.
2. Colocar no programa de aplicação o identificador com o endereço do slave na entrada *A\_ID*.
3. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.

---

**i**

*O bloco funcional reage a uma mudança de flanco positiva em A\_Req.*

---

A saída *A\_Busy* está TRUE até o comando SLACT estiver processado. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Done* para TRUE.

Na saída *A\_Status* é emitido o Slave Mode, se foi possível ajustar o Slave Mode.

Na saída *A\_Status* é emitido um código de erro, se não foi possível ajustar o Slave Mode.

## 6.9.6 Bloco funcional WRREC

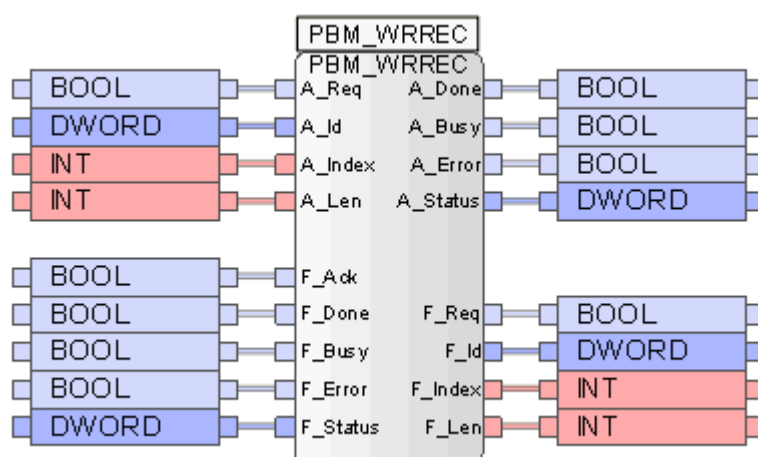


Figura 49: Bloco funcional WRREC

O bloco funcional **WRREC** (a partir de DP-V1) é utilizado para a escrita acíclica de um conjunto de dados endereçado para um slave com *A\_Index*. Deve ser consultado no manual de operação do slave quais dados podem ser escritos.

No master HIMA PROFIBUS DP, até 32 blocos **RDREC** e/ou **WRREC** podem ser ativos simultaneamente.

- i** Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Um flanco positivo inicia a requisição de escrita de um conjunto de dados.	BOOL
A_ID	Número de identificação do slave (veja Capítulo 6.10)	DWORD
A_Index	Número do conjunto de dados do conjunto de dados a ser escrito. O significado deve ser consultado na descrição do fabricante do equipamento.	INT
A_Len	Comprimento do conjunto de dados a ser escrito em bytes	INT

Tabela 130: Entradas A bloco funcional WRREC

Saídas A	Descrição	Type
A_Done	TRUE: O bloco funcional encerrou o processo de escrita.	BOOL
A_Busy	TRUE: O bloco funcional ainda não encerrou o processo de escrita.	BOOL
A_Error	TRUE: Ocorreu um erro durante o processo de escrita.	BOOL
A_STATUS	Status ou código de erro (veja Capítulo 6.11)	DWORD

Tabela 131: Saídas A bloco funcional WRREC

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **WRREC** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **WRREC** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **WRREC** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **WRREC** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **WRREC** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Done	BOOL
F_Busy	BOOL
F_Error	BOOL
F_Status	DWORD

Tabela 132: Entradas F bloco funcional **WRREC**

Conectar as *F-Outputs* do bloco funcional **WRREC** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **WRREC** na árvore de estrutura também.

Saídas F	Tipo
F_Req	BOOL
F_Id	DWORD
F_Index	INT
F_Len	INT

Tabela 133: Saídas F bloco funcional **WRREC****Assim, cria-se o bloco funcional **WRREC** na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master, Function Blocks, New**.
2. Selecionar o bloco funcional **WRREC**.
3. Clicar com o botão direito no bloco funcional **WRREC** e selecionar **Edit**.
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **WRREC** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **WRREC** no programa de aplicação.

Entradas	Tipo
ID	DWORD
INDEX	INT
LEN	INT
REQ	BOOL

Tabela 134: Variáveis de sistema de entrada

Conectar as saídas do bloco funcional **WRREC** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **WRREC** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
DONE	BOOL
ERROR	BOOL
STATUS	DWORD

Tabela 135: Variáveis de sistema de saída

Dados	Descrição
Sem variáveis pré-definidas	No registro <i>Process Variables</i> , é possível definir livremente uma estrutura de dados, mas ela deve ser compatível com a estrutura do conjunto de dados. A estrutura do conjunto de dados deve ser consultada no manual de operação do fabricante do slave.

Tabela 136: Dados

**Para a operação do bloco funcional WRREC são necessários os seguintes passos:**

1. Colocar no programa de aplicação o endereço do slave na entrada *A\_ID*.
2. Colocar no programa de aplicação o index específico do slave para o conjunto de dados (manual do fabricante) na entrada *A\_Index*.
3. Colocar no programa de aplicação o comprimento do conjunto de dados a ser escrito na entrada *A\_Len*.
4. Ajustar no programa de aplicação o conjunto de dados como definido no registro "Dados".
5. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.

---

**i**

O bloco funcional reage a uma mudança de flanco positiva em *A\_Req*.

---

A saída *A\_Busy* está em TRUE até a requisição do conjunto de dados terminar de ser escrito. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Done* para TRUE.

Se não foi possível escrever o conjunto de dados com êxito, a saída *A\_Error* muda para TRUE e um código de erro é emitido na saída *A\_Status*.



## 6.10 Blocos funcionais auxiliares PROFIBUS

Os blocos funcionais auxiliares são utilizados para a parametrização e avaliação das entradas e saídas dos blocos funcionais.

Os seguintes blocos funcionais auxiliares estão à disposição:

Blocos funcionais auxiliares	Descrição da função
ACTIVE (veja Cap. 6.10.1)	O slave está ativo ou inativo
ALARM (veja Cap. 6.10.2)	Decodificar os dados do alarme
DEID (veja Cap. 6.10.3)	Decodificar o número de identificação
ID (veja Cap. 6.10.4)	A função ID gera um identificador a partir de quatro bytes
NSLOT (veja Cap. 6.10.5)	Criar um número sequencial de identificação para os slots
SLOT (veja Cap. 6.10.6)	Criar um número de identificação SLOT com o número de slot
STDDIAG (veja Cap. 6.10.7)	Decodificar o diagnóstico padrão de um slave
LATCH	É usado apenas dentro de outros blocos funcionais
PIG	É usado apenas dentro de outros blocos funcionais
PIGII	É usado apenas dentro de outros blocos funcionais

Tabela 137: Visão geral – Blocos funcionais auxiliares

### 6.10.1 Bloco funcional auxiliar ACTIVE

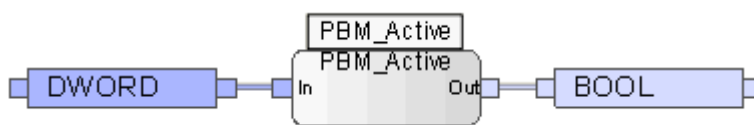


Figura 50: Bloco funcional auxiliar ACTIVE

O bloco funcional auxiliar ACTIVE determina a partir do diagnóstico padrão de um slave PROFIBUS DP se o slave está ativo ou inativo no momento.

**i**

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

Entradas	Descrição	Type
IN	Diagnóstico padrão do slave	DWORD

Tabela 138: Entradas bloco funcional auxiliar ACTIVE

Saídas	Descrição	Type
OUT	TRUE: O slave está ativo FALSE: O slave está inativo	BOOL

Tabela 139: Saídas bloco funcional auxiliar ACTIVE

6.10.2 Bloco funcional auxiliar ALARM  
(Decodificar os dados de alarme)

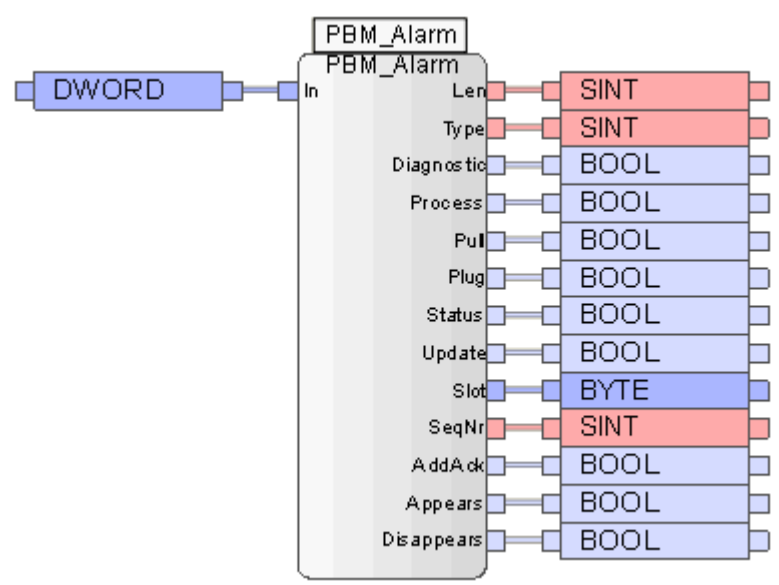


Figura 51: Bloco funcional auxiliar ALARM

O bloco funcional auxiliar **ALARM** decodifica os dados de alarme padrão de um slave PROFIBUS DP.

**i** Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

Entradas	Descrição	Type
IN	Alarme padrão	DWORD

Tabela 140: Entradas bloco funcional auxiliar ALARM

Saída	Descrição	Type
Len	Comprimento da mensagem de alarme completa	SINT
Type	1: Alarme diagnóstico 2: Alarme de processo 3: Alarme de retirar módulo 4: Alarme de encaixar módulo 5: Alarme de status 6: Alarme de Update Outros números estão reservados ou são específicos do respectivo fabricante. O significado deve ser consultado na descrição do fabricante do equipamento.	SINT
Diagnostic	True = Alarme diagnóstico	BOOL
Process	True = Alarme de processo	BOOL
Pull	True = Módulo foi retirado	BOOL
Plug	True = Módulo foi colocado novamente	BOOL
Status	True = Status Alarm	BOOL
Update	True = Update Alarm	BOOL
Slot	Módulo que disparou o alarme	BYTE
SeqNr	Número sequencial do alarme	SINT

Saída	Descrição	Type																											
AddAck	TRUE significa que o slave que disparou o alarme está esperando uma confirmação adicional pela aplicação. Qual é exatamente, deve ser consultado no manual do fabricante do slave.	BOOL																											
Appears Disappears	<table border="1"> <thead> <tr> <th>Saída</th><th>Valor</th><th>Descrição</th></tr> </thead> <tbody> <tr> <td>Appears</td><td>TRUE</td><td>Se ambos são FALSO, não ocorreu nenhum erro até este momento</td></tr> <tr> <td>Disappears</td><td>FALSE</td><td></td></tr> <tr> <td>Appears</td><td>TRUE</td><td>Ocorreu um erro e ainda está ativo.</td></tr> <tr> <td>Disappears</td><td>FALSE</td><td></td></tr> <tr> <td>Appears</td><td>TRUE</td><td>Um erro tinha ocorrido está desaparecendo neste momento.</td></tr> <tr> <td>Disappears</td><td>FALSE</td><td></td></tr> <tr> <td>Appears</td><td>TRUE</td><td>Se ambos são TRUE, o erro desaparece, mas o slave continua avariado.</td></tr> <tr> <td>Disappears</td><td>FALSE</td><td></td></tr> </tbody> </table>	Saída	Valor	Descrição	Appears	TRUE	Se ambos são FALSO, não ocorreu nenhum erro até este momento	Disappears	FALSE		Appears	TRUE	Ocorreu um erro e ainda está ativo.	Disappears	FALSE		Appears	TRUE	Um erro tinha ocorrido está desaparecendo neste momento.	Disappears	FALSE		Appears	TRUE	Se ambos são TRUE, o erro desaparece, mas o slave continua avariado.	Disappears	FALSE		BOOL
Saída	Valor	Descrição																											
Appears	TRUE	Se ambos são FALSO, não ocorreu nenhum erro até este momento																											
Disappears	FALSE																												
Appears	TRUE	Ocorreu um erro e ainda está ativo.																											
Disappears	FALSE																												
Appears	TRUE	Um erro tinha ocorrido está desaparecendo neste momento.																											
Disappears	FALSE																												
Appears	TRUE	Se ambos são TRUE, o erro desaparece, mas o slave continua avariado.																											
Disappears	FALSE																												

Tabela 141: Saídas bloco funcional auxiliar ALARM

### 6.10.3 Bloco funcional auxiliar DEID (Decodificar o número de identificação)

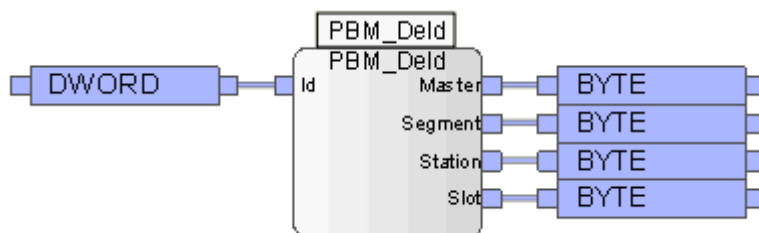


Figura 52: Bloco funcional auxiliar DEID

O bloco funcional auxiliar **DEID** decodifica o número de identificação e o divide nos seus quatro componentes.

**i**

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

Entradas	Descrição	Type
Id	Número de identificação do slave	DWORD

Tabela 142: Entradas bloco funcional auxiliar DEID

Saídas	Descrição	Type
Master	Endereço de barramento do master	BYTE
Segment	Segmento	BYTE
Station	Endereço de barramento do slave	BYTE
Slot	Número do slot ou módulo	BYTE

Tabela 143: Saídas bloco funcional auxiliar DEID

6.10.4      Bloco funcional auxiliar ID  
(Gerar o número de identificação)

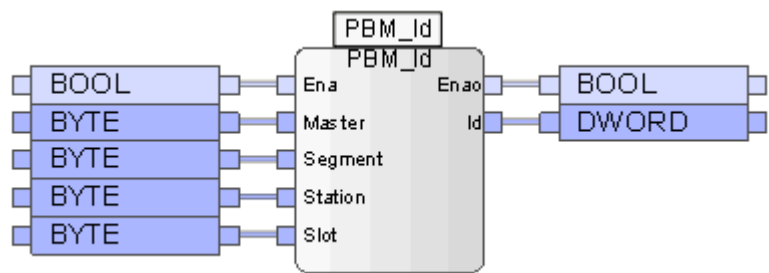


Figura 53: Bloco funcional auxiliar ID

O bloco funcional auxiliar **ID** gera um identificador a partir de quatro bytes (número de identificação) que é usado pelos outros blocos funcionais auxiliares.

**i** Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

Entradas	Descrição	Type
Ena	Não usado	BOOL
Master	Endereço do barramento	BYTE
Segment	Segmento	BYTE
Station	Endereço de barramento do slave	BYTE
Slot	Número do slot ou módulo	BYTE

Tabela 144: Entradas bloco funcional auxiliar ID

Saídas	Descrição	Type
Enao	Não usado	BOOL
Id	Número de identificação do slave	DWORD

Tabela 145: Saídas bloco funcional auxiliar ID

## 6.10.5 Bloco funcional auxiliar NSLOT

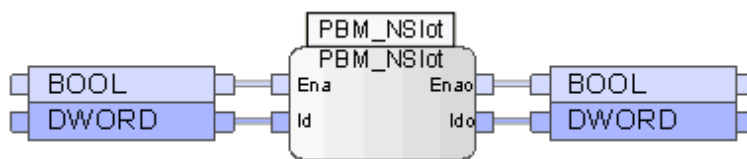


Figura 54: Bloco funcional auxiliar NSLOT

O bloco funcional auxiliar **NSLOT** gera a partir de um identificador um novo identificador que endereça o próximo slot no mesmo slave. Ena deve ser ajustado para TRUE para que o bloco funcional auxiliar seja executado.

Enao é TRUE quando o resultado na saída Ido estiver válido.

**i**

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

Entradas	Descrição	Type
Ena	Enquanto TRUE estiver ativo, o bloco está em operação.	BOOL
Id	Número de identificação do slave	DWORD

Tabela 146: Entradas bloco auxiliar NSLOT

Saídas	Descrição	Type
Enao	TRUE = Resultado válido FALSE = Não há outros números de slot	BOOL
Ido	Número de identificação do slave	DWORD

Tabela 147: Saídas bloco auxiliar NSLOT

## 6.10.6 Bloco funcional auxiliar SLOT

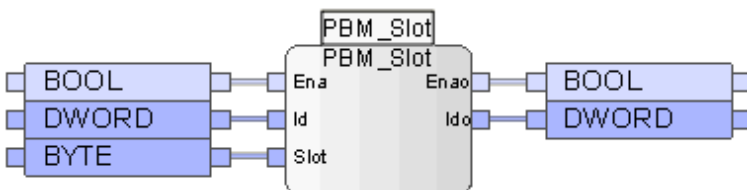


Figura 55: Bloco funcional auxiliar SLOT

O bloco funcional auxiliar **SLOT** gera um novo identificador a partir de um identificador e um número de slot, para endereçar o mesmo slave como o identificador antigo, porém, com o novo número de slot.

**i**

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

Entradas	Descrição	Type
Ena	Não usado	BOOL
Id	Endereço lógico do componente slave (Slave ID e número de slot)	DWORD
Slot	Novo número de slot ou módulo	BYTE

Tabela 148: Entradas bloco auxiliar SLOT

Saídas	Descrição	Type
Enao	Não usado	BOOL
Ido	Número de identificação do slave	DWORD

Tabela 149: Saídas bloco auxiliar SLOT

### 6.10.7 Bloco funcional auxiliar STDDIAG

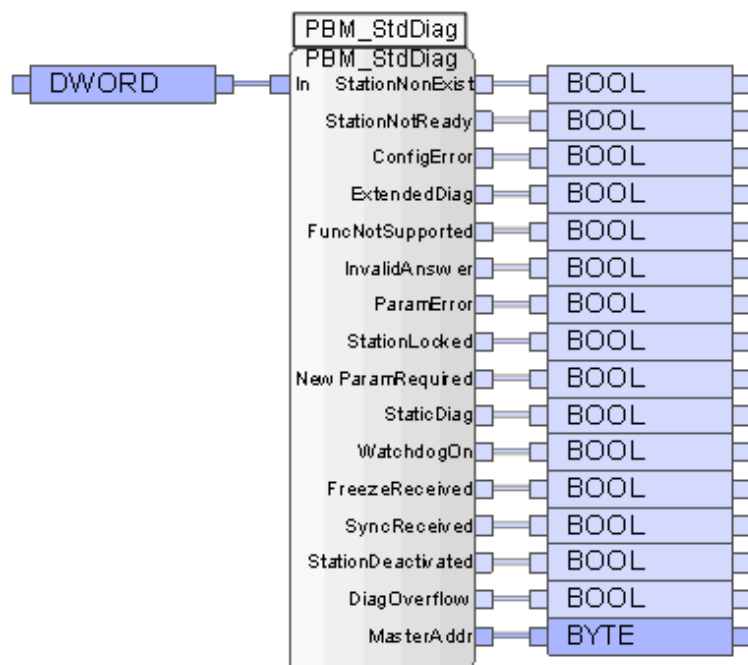


Figura 56: Bloco funcional auxiliar STDDIAG

O bloco funcional auxiliar diagnóstico padrão **STDDIAG** decodifica o diagnóstico padrão de um slave PROFIBUS DP.

As saídas do tipo BOOL do bloco funcional **STDDIAG** são TRUE se o bit correspondente no diagnóstico padrão estiver colocado.

#### i

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

Entradas	Descrição	Type
IN	Diagnóstico padrão do slave	DWORD

Tabela 150: Entradas bloco auxiliar STDDIAG

Saídas	Descrição	Type
StationNonExist	O slave não existe	BOOL
StationNotReady	O slave não está pronto	BOOL
ConfigError	Erro de configuração	BOOL
ExtendedDiag	Diagnóstico extenso segue	BOOL
FuncNotSupported	Função não é suportada	BOOL
InvalidAnswer	Resposta inválida do slave	BOOL
ParamError	Erro de parametrização	BOOL
StationLocked	O slave está bloqueado por um outro master	BOOL
NewParamRequired	Novos dados de parametrização são necessários	BOOL
StaticDiag	Diagnóstico estático	BOOL
WatchdogOn	Watchdog ativado	BOOL
FreezeReceived	Comando Freeze recebido	BOOL
SyncReceived	Comando Sync recebido	BOOL
StationDeactivated	O slave foi desativado	BOOL
DiagOverflow	Transbordamento de diagnóstico	BOOL
MasterAddr	Endereço de barramento do master	BYTE

Tabela 151: Saídas bloco auxiliar STDDIAG

**Assim lê-se o diagnóstico padrão do slave PROFIBUS DP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master**.
2. Clicar com o botão direito em **PROFIBUS Slave** e selecionar **Edit**.
3. Puxar a variável global do tipo DWORD para o campo *Standard Diagnosis*.
4. Conectar esta variável global com a entrada do bloco funcional **STDDIAG**.

### 6.11 Códigos de erro dos blocos funcionais

Se um bloco funcional não conseguiu executar um comando corretamente, é emitido um código de erro na saída **A\_Status**. O significado do código de erro deve ser conferido na seguinte tabela:

Código de erro	Símbolo	Descrição
16#40800800	TEMP_NOT_AVAIL	O serviço temporariamente não está disponível
16#40801000	INVALID_PARA	Parâmetro inválido
16#40801100	WRONG_STATE	O slave não suporta DP-V1
16#40808000	FATAL_ERR	Erro fatal de programa
16#40808100	BAD_CONFIG	Erro de configuração na área de dados
16#40808200	PLC_STOPPED	O sistema de comando foi parado
16#4080A000	READ_ERR	Erros ao ler um Record
16#4080A100	WRITE_ERR	Erros ao escrever um Record
16#4080A200	MODULE_FAILURE	Erro não pode ser especificado
16#4080B000	INVALID_INDEX	Index inválido
16#4080B100	WRITE_LENGTH	Comprimento incorreto ao escrever
16#4080B200	INVALID_SLOT	Número de slot inválido
16#4080B300	TYPE_CONFLICT	Tipo incorreto
16#4080B400	INVALID_AREA	Área de leitura ou escrita incorreta
16#4080B500	STATE_CONFLICT	Master no estado incorreto
16#4080B600	ACCESS_DENIED	O slave não está ativo (ou algo semelhante)
16#4080B700	INVALID_RANGE	Área de leitura ou escrita incorreta
16#4080B800	INVALID_PARAMETER	Valor de parâmetro incorreto
16#4080B900	INVALID_TYPE	Tipo de parâmetro incorreto
16#4080C300	NO_RESOURCE	Slave não existe
16#4080BA00	BAD_VALUE	Valor inválido
16#4080BB00	BUS_ERROR	Erro de barramento
16#4080BC00	INVALID_SLAVE	ID do slave inválido
16#4080BD00	TIMEOUT	Ocorreu um Timeout
16#4080C000	READ_CONSTRAIN	Restrição de leitura
16#4080C100	WRITE_CONSTRAIN	Restrição de escrita
16#4080C200	BUSY	Um bloco deste tipo já está ativo
16#4080C300	NO_RESOURCE	Slave não ativo

Tabela 152: Códigos de erro blocos funcionais



## 6.12 Control Panel (PROFIBUS DP Master)

No Control Panel, o usuário pode verificar e controlar os ajustes do master PROFIBUS DP. Além disso, informações de status atuais (p. ex., tempo de ciclo, estado do barramento, etc.) do master e do slave correspondente são exibidas.

### **Assim abre-se o Control Panel para a supervisão do master PROFIBUS DP:**

1. Selecionar na árvore de estrutura **Hardware** e no menu de contexto **Online**.
2. Introduzir no **System Login** os dados de acesso para abrir a visualização online do hardware.
3. Clique duplo em **COM-Module** e selecionar na árvore de estrutura **PROFIBUS DP Master**.

### 6.12.1 Menu de contexto (PROFIBUS Master)

No menu de contexto do master PROFIBUS DP selecionado, é possível escolher os seguintes comandos:

#### **Offline:**

Desliga o master PROFIBUS DP selecionado. Se o master estiver desligado, não ocorrem atividades.

#### **Stop:**

Pára o master PROFIBUS DP selecionado. O master PROFIBUS DP continua participando no protocolo token, mas não envia dados aos slaves.

#### **Clear:**

Mediante acionamento do botão CLEAR, o master PROFIBUS DP selecionado é colocado no estado seguro e então troca dados seguros com os slaves. Os dados de saída enviados aos slaves apenas contêm zeros. Slaves FailSave recebem telegramas FailSave que não contêm dados. Os dados de entrada dos slaves são ignorados pelo master PROFIBUS DP e ao invés disso, são utilizados valores iniciais no programa de aplicação.

#### **Operate:**

Inicia o master PROFIBUS DP selecionado. O master HIMA PROFIBUS DP troca dados de E/S ciclicamente com os slaves.

#### **Reset Statistics:**

O botão *Reset Statistics* reseta para zero os dados estatísticos (tempo de ciclo mín, máx, etc.).

### 6.12.2 Menu de contexto (Slave PROFIBUS)

No menu de contexto do slave PROFIBUS DP selecionado, é possível escolher os seguintes comandos:

#### **Activate:**

Ativa o slave PROFIBUS DP selecionado que agora pode trocar dados com o master PROFIBUS DP.

#### **Deactivate:**

Desativa o slave selecionado. A comunicação é encerrada.

## 6.12.3 Campo de exibição (PROFIBUS Master)

No campo de exibição são mostrados os seguintes valores do master PROFIBUS DP selecionado.

Elemento	Descrição
Name	Nome do master PROFIBUS DP
Master State	Mostra o estado atual do protocolo (veja Capítulo 6.12.4). 0 = OFFLINE 1 = STOP 2 = CLEAR 3 = OPERATE 100 = UNDEFINED
Bus State	Código de erro: erro de barramento 0...6: 0 = OK 1 = Erro de endereço: O endereço do master já existe no barramento. 2 = Avaria no barramento: Uma avaria no barramento foi registrada, p. ex., barramento não corretamente terminado, vários participantes enviam ao mesmo tempo. 3 = Erro de protocolo: Um pacote com codificação incorreta recebido. 4 = Erro de hardware: O hardware comunicou um erro, p. ex., no caso de tempos ajustados muito curtos. 5 = Erro desconhecido: O master mudou de estado por motivo desconhecido. 6 = Controller Reset: No caso de avarias graves do barramento, acontece que o chip do controlador entra um estado não definido e é resetado. O código de erro continua presente até o erro de barramento ser eliminado.
Fieldbus Interface	FB1, FB2
μP load (planned) [%]	Exibição da carga projetada do módulo COM para este protocolo.
μP Load (actual) [%]	Exibição da carga atual do módulo COM para este protocolo.
Baud Rate [bps]	Baudrate do master. O master consegue comunicar-se com todas as taxas de transmissão especificadas no padrão. Tempos de ciclo até um limite inferior de 2 ms são possíveis.
Fieldbus address	Endereço de barramento do master (0...125)
PNO-IdentNo	Número de 16 Bits atribuído pela Associação PROFIBUS DP que identifica um produto (equipamento de campo) de forma inequívoca.
Bus Error Count	Quantidade de erros de barramento até então
MSI [ms]	Min. Slave Interval em ms, resolução 0.1 ms
TTR [ms]	Target Rotation Time em ms, resolução 0.1 ms
Last Cycle Time [ms]	Último tempo de ciclo [ms] do PROFIBUS DP

Elemento	Descrição
Minimum Cycle Time [ms]	Tempo de ciclo mínimo [ms] do PROFIBUS DP
Average Cycle Time [ms]	Tempo de ciclo médio [ms] do PROFIBUS DP
Maximum Cycle Time [ms]	Tempo de ciclo máximo [ms] do PROFIBUS DP

Tabela 153: Campo de exibição Master PROFIBUS

#### 6.12.4 Estado do master PROFIBUS DP

O estado do master é mostrado no campo de exibição do Control Panel e pode ser avaliado no programa de aplicação com a variável de status Master Connection State.

Master State	Estado do master
OFFLINE	O master está desligado, não ocorrem atividades do barramento.
STOP	O master continua participando no protocolo token, mas não envia dados aos slaves.
CLEAR	O master está no estado seguro e troca dados com os slaves. <ul style="list-style-type: none"> <li>Os dados de saída enviados aos slaves apenas contêm zeros.</li> <li>Os slaves FailSave recebem telegramas FailSave (estes não contêm dados).</li> <li>Os dados de entrada dos slaves são ignorados pelos slaves e ao invés disso, são utilizados valores iniciais.</li> </ul>
OPERATE	O master PROFIBUS DP está no modo de trabalho e ciclicamente trocando dados de E/S com os slaves.
UNDEFINED	A atualização do Firmware do módulo master PROFIBUS DP está em andamento.

Tabela 154: Estado do master PROFIBUS DP

#### 6.12.5 Comportamento do master PROFIBUS DP

Comportamento do master PROFIBUS DP em decorrência do estado operacional do sistema de comando.

Estado do sistema de comando	Comportamento do master HIMA PROFIBUS DP
STOP *)	Sistema de comando em STOP, então, o master está no estado OFFLINE.
RUN	O sistema de comando está em RUN, o master tenta chegar no estado OPERATE.
STOP	Sistema de comando em STOP, o master entra no estado CLEAR. Se o master já estiver em STOP ou OFFLINE, permanece neste estado.

\*) Depois de ligar o sistema de comando ou depois de carregar a configuração

Tabela 155: Comportamento master PROFIBUS DP

## 6.12.6 Função do LED FBx no master PROFIBUS DP

O estado do protocolo PROFIBUS-DP é sinalizado pelo LED FBx da interface correspondente do barramento de campo. Os estados do LED FBx são mostrados na seguinte tabela:

LED FBx	Descrição
DESLIGA	Sem configuração ou configuração inválida do master PROFIBUS DP.
Piscando em ritmo de 2 segundos	Configuração válida. O master PROFIBUS DP está no estado OFFLINE ou STOP.
LIGA	O master PROFIBUS DP está no estado OPERATE ou CLEAR se todos os slaves ativados estão conectados.
Piscando a cada segundo	No mínimo um slave falhou.

Tabela 156: LED FBx

## 6.12.7 Função do LED FAULT no master PROFIBUS DP (Apenas HiMax)

A avaria do protocolo PROFIBUS-DP é sinalizada pelo LED FAULT da interface correspondente do barramento de campo. Os estados do LED FAULT são mostrados na seguinte tabela:

LED FAULT	Cor	Descrição
DESLIGA	Vermelho	Protocolo PROFIBUS-DP sem avaria.
Piscando	Vermelho	Protocolo avariado. <ul style="list-style-type: none"> <li>No mínimo um slave falhou</li> <li>Erro de barramento detectado.</li> <li>Tempo de processamento ultrapassado.</li> </ul> Se durante mais de 5 segundos não ocorrer nenhum evento de erro, a exibição muda para o estado <i>Protocol not disturbed</i> .

Tabela 157: FAULT FBx

### 6.13 Slave HIMA PROFIBUS DP

Este capítulo descreve as características do slave HIMA PROFIBUS DP, bem como as funções de menu e diálogos no SILworX que são necessários para a configuração do slave HIMA PROFIBUS DP.

#### Equipamentos necessários e requisitos de sistema

Elemento	Descrição
Sistema de comando HIMA	HIMax com módulo COM HIMatrix a partir de CPU OS V7 e COM OS V12
Módulo COM	O módulo COM deve estar equipado com um submódulo slave HIMA PROFIBUS DP opcional na interface de barramento de campo serial usada (FB1 ou FB2). Atribuição de interfaces, veja Capítulo 3.6.
Ativação	Liberação mediante módulo de encaixe, veja Capítulo 3.4.

Tabela 158: Equipamentos necessários e requisitos de sistema do slave HIMA PROFIBUS DP

#### i

A HIMA recomenda operar HIMax, PROFIBUS DP pela interface de barramento de campo FB1 (taxa de transmissão máxima 12 Mbit). Pela interface de barramento de campo FB2 é permitida uma taxa de transmissão máxima 1,5 Mbit.

#### PROFIBUS DP Slave Properties

Elemento	Descrição
Tipo de slaves HIMA PROFIBUS DP	DP-V0
Taxa de transmissão	9,6 kbit/s... 12 Mbit/s
Endereço do barramento	0...125
Quantidade máx. de slaves	Para cada módulo COM pode ser configurado um slave HIMA PROFIBUS DP.
Volume de dados de processo de um slave HIMA PROFIBUS DP	DP-Output: máx. 192 Bytes DP-Input: máx. 240 Bytes Porém, no total: máx. 256 Bytes
Protocolo Watchdog	Se o COM está no estado RUN e a conexão ao master PROFIBUS DP se perder, isso é detectado pelo slave DP depois do timeout do Watchdog esgotar (deve ser parametrizado no master). Neste caso, os dados de saída DP (dados de entrada do ponto de vista do recurso) são resetados para o seu valor inicial e o flag <i>Data Valid</i> (variável de status do protocolo do slave DP) é ajustado para FALSE.

Tabela 159: Características do slave HIMA PROFIBUS DP

#### 6.13.1 Criar um slave HIMA PROFIBUS DP

##### Assim cria-se um novo slave HIMA PROFIBUS DP:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Selecionar no menu de contexto dos protocolos **New, PROFIBUS DP Slave** para adicionar um novo slave PROFIBUS DP.
3. Selecionar no menu de contexto do slave PROFIBUS DP **Edit**.
4. Selecionar no registro **Properties** o **Module** e as **Interfaces**.

## 6.14 Funções de menu slave PROFIBUS DP

### 6.14.1 Edit

A janela de diálogo **Edit** do master PROFIBUS DP contém os seguintes registros.

#### Variáveis de processo

No registro **Process Variables** são criadas as variáveis para recepção e envio.

#### Variáveis de entrada

As variáveis que devem ser recebidas por este sistema de comando são introduzidas na área *Input Signals*.

Na área *Input Signals* pode ser criado um número livre de variáveis. Os offsets e tipos de variáveis, porém, devem ser idênticos com os offsets e tipos de variáveis de envio do parceiro de comunicação.

#### Variáveis de saída

As variáveis para a troca de dados cíclica que devem ser enviadas por este sistema de comando são introduzidas na área *Output Signals*.

Na área *Output Signals* pode ser criado um número livre de variáveis. Os offsets e tipos de variáveis, porém, devem ser idênticos com os offsets e tipos de variáveis de recepção do parceiro de comunicação.

#### Variáveis de sistema

No registro **System Variables** é definido quais variáveis devem ser lidos para o sistema de comando.

O registro **System Variables** disponibiliza as seguintes variáveis de sistema que permitem avaliar o estado do slave PROFIBUS DP no programa de aplicação.

Elemento	Descrição
Current baud rate	Baud rate com a qual trabalha o protocolo slave PROFIBUS-DP.
Data valid	<p>Se a variável de status <i>Data Valid</i> foi definida para TRUE, então, o slave recebeu dados de importação válidos do master.</p> <p>A variável de status é definida para FALSE se o tempo de Watchdog estiver esgotado no slave.</p> <p>Valor padrão: FALSE</p> <p>Nota:</p> <p>Se o Watchdog do slave não foi ativado pelo master e a conexão se perder, a variável de status <i>Data Valid</i> mantém o valor TRUE, pois o slave PROFIBUS DP não tem como detectar a perda da conexão.</p> <p>Esta situação deve ser impreterivelmente observada ao usar esta variável!</p>
Error code	<p>Se um erro ocorreu no protocolo slave PROFIBUS-DP, isso é transmitido nesta variável. Sempre é exibido o erro atualmente ocorrido.</p> <p>Possíveis valores (hexadecimais) são:</p> <p>0x00: Sem erro</p> <p>0xE1: Parametrização incorreta pelo master PROFIBUS DP</p> <p>0xD2: Configuração incorreta pelo master PROFIBUS DP Master</p> <p>Valor padrão: 0x00</p>
Master Address	<p>Este é o endereço do master PROFIBUS DP que parametrizou e configurou o seu próprio slave PROFIBUS DP.</p> <p>Possíveis valores (decimais) são:</p> <p>0–125: Endereço do master</p> <p>255: Slave não está atribuído a um Master atualmente</p> <p>Valor padrão: 255</p>
Protocol State	<p>Descreve o estado do protocolo slave PROFIBUS DP.</p> <p>Possíveis valores (hexadecimais) são:</p> <p>0xE1: O sistema de comando está separado do barramento ou inativo.</p> <p>0xD2: O sistema de comando aguarda uma configuração pelo master PROFIBUS DP.</p> <p>0xC3: O sistema de comando está trocando dados ciclicamente com o master PROFIBUS DP.</p> <p>Valor padrão: 0xE1</p>
Slave Address	<p>Este é o endereço do slave PROFIBUS DP do sistema de comando. Este endereço foi anteriormente configurado pelo usuário via PADT.</p> <p>Possíveis valores (decimais) são:</p> <p>0–125: Endereço do slave</p>
Watchdog Time	Tempo de Watchdog em milissegundos parametrizado no master. Veja Capítulo 6.6.3.

Tabela 160: Variáveis de sistema slave PROFIBUS DP

## 6.14.2 Características

O registro **Properties** do slave HIMA PROFIBUS DP contém os seguintes parâmetros para a configuração do slave PROFIBUS DP.

Os valores pré-estabelecidos dos parâmetros *Within one cycle* e *Process Data Refresh Rate [ms]* garantem a troca de dados rápida dos dados PROFIBUS DP entre o módulo COM (COM) e o hardware do slave PROFIBUS DP do sistema de comando HIMax/HIMatrix.

Estes parâmetros apenas devem ser alterados se uma redução da carga COM é necessária para uma aplicação e se o processo permitir.

**i**

A alteração dos parâmetros apenas se recomenda para programadores experientes.

Um aumento do tempo de atualização do hardware COM e PROFIBUS DP também significa que o tempo de atualização real dos dados PROFIBUS DP aumenta. As requisições de tempo do sistema devem ser verificadas.

Observar também o parâmetro **Min. Slave Intervall [ms]** (veja Capítulo 6.3.2 no registro Times) que define o tempo de atualização mínimo dos dados PROFIBUS DP entre o master PROFIBUS DP e o slave PROFIBUS DP.

Elemento	Descrição
Type	PROFIBUS DP Slave
Name	Nome para o slave PROFIBUS DP.
Module	Seleção do módulo COM no qual este protocolo é processado.
Activate Max. $\mu$ P Budget	Ativado: Transferir o limite do $\mu$ P Budget do campo Max. $\mu$ P-Budget in [%].  Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P Budget in [%]	Valor máximo de $\mu$ P-Budget do módulo que pode ser produzido ao processar o protocolo. Faixa de valores: 1...100% Valor padrão: 30%
Behavior on CPU/COM Connection Loss	No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. (P. ex., se o módulo de comunicação é puxado para fora durante a comunicação).  Adopt Initial Data      As variáveis de entrada são resetadas para os valores iniciais.  Retain Last Value      As variáveis de entrada mantêm o último valor.
Station address	Endereço de estação do slave. O endereço de estação do slave apenas pode existir uma vez no barramento. Faixa de valores: 1...125 Valor padrão: 0
Interface	Interface do barramento de campo que deve ser usada para o slave PROFIBUS DP. Faixa de valores: fb1, fb2 Valor padrão: nenhum



Elemento	Descrição																																												
Baud Rate [bps]	<p>Baudrate com a qual o barramento é operado.</p> <p>Valores possíveis:</p> <table><tr><th>Valor</th><th>Baud Rate</th><th>FB1</th><th>FB2</th></tr><tr><td>9600</td><td>9,6 kbit/s</td><td>X</td><td>X</td></tr><tr><td>19200</td><td>16,2 kbit/s</td><td>X</td><td>X</td></tr><tr><td>45450</td><td>45,45 kbit/s</td><td>X</td><td>X</td></tr><tr><td>93750</td><td>93,75 kbit/s</td><td>X</td><td>X</td></tr><tr><td>187500</td><td>187,5 kbit/s</td><td>X</td><td>X</td></tr><tr><td>500000</td><td>500 kbit/s</td><td>X</td><td>X</td></tr><tr><td>1500000</td><td>1,5 Mbit/s</td><td>X</td><td>X</td></tr><tr><td>3000000</td><td>3 Mbit/s</td><td>X</td><td>-</td></tr><tr><td>6000000</td><td>6 Mbit/s</td><td>X</td><td>-</td></tr><tr><td>12000000</td><td>12 Mbit/s</td><td>X</td><td>-</td></tr></table>	Valor	Baud Rate	FB1	FB2	9600	9,6 kbit/s	X	X	19200	16,2 kbit/s	X	X	45450	45,45 kbit/s	X	X	93750	93,75 kbit/s	X	X	187500	187,5 kbit/s	X	X	500000	500 kbit/s	X	X	1500000	1,5 Mbit/s	X	X	3000000	3 Mbit/s	X	-	6000000	6 Mbit/s	X	-	12000000	12 Mbit/s	X	-
Valor	Baud Rate	FB1	FB2																																										
9600	9,6 kbit/s	X	X																																										
19200	16,2 kbit/s	X	X																																										
45450	45,45 kbit/s	X	X																																										
93750	93,75 kbit/s	X	X																																										
187500	187,5 kbit/s	X	X																																										
500000	500 kbit/s	X	X																																										
1500000	1,5 Mbit/s	X	X																																										
3000000	3 Mbit/s	X	-																																										
6000000	6 Mbit/s	X	-																																										
12000000	12 Mbit/s	X	-																																										
Process data refresh rate [ms]	<p>Tempo de atualização em milissegundos com o qual os dados do protocolo são trocados entre COM e hardware do slave PROFIBUS DP.</p> <p>Faixa de valores: 4...1000</p> <p>Valor padrão: 10</p>																																												
Force Process Data Consistency	<p>Ativado:</p> <p>Transferência dos dados completos do protocolo de CPU para COM dentro de um ciclo de CPU.</p> <p>Desativado:</p> <p>Transferência dos dados completos do protocolo de CPU para COM, distribuídos sobre vários ciclos de CPU cada vez 1100 Byte por direção de dados. Com isso, eventualmente também é possível reduzir o tempo de ciclo do sistema de comando.</p> <p>Valor padrão: Ativado</p>																																												

Tabela 161: Características do slave: Registro geral

## 6.15 Control Panel (Slave PROFIBUS DP)

No Control Panel, o usuário pode verificar e controlar os ajustes do slave PROFIBUS DP. Além disso, informações de status atuais (p. ex., tempo de ciclo, estado do barramento, etc.) do slave são exibidas.

### Assim abre-se o Control Panel para a supervisão do slave PROFIBUS DP:

1. Selecionar na árvore de estrutura **Hardware** e no menu de contexto **Online**.
2. Introduzir no **System Login** os dados de acesso para abrir a visualização online do hardware.
3. Clique duplo em **COM-Module** e selecionar na árvore de estrutura **PROFIBUS DP Slave**.

### 6.15.1 Campo de exibição (Slave PROFIBUS DP)

No campo de exibição são mostrados os seguintes valores do slave PROFIBUS DP selecionado.

Elemento	Descrição
Name	Nome do slave PROFIBUS DP
Fieldbus Interface	Interface de barramento de campo atribuída do slave
Protocol State	Estado da conexão 0 = Desativado, 1 = Inativo (tenta estabelecer conexão), 2 = Conectado
Error State	Veja Capítulo 6.14.1
Timeout	Tempo de Watchdog em milissegundos parametrizado no master. Veja Capítulo 6.6.3
Watchdog Time [ms]	É ajustado no master. Veja Capítulo 6.6.3.
Fieldbus address	Veja Capítulo 6.14.2.
Master Address	Endereço do master PROFIBUS DP.
Baud Rate [bps]	Taxa de transmissão atual em Baud. Veja Capítulo 6.14.2.
µP Budget (planned) [%]	Exibição da carga projetada do módulo COM para este protocolo.
µP Budget (actual) [%]	Exibição da carga atual do módulo COM para este protocolo.

Tabela 162: Campo de exibição (Slave PROFIBUS DP)

### 6.16 Função do LED FBx no slave PROFIBUS DP

O estado do protocolo PROFIBUS-DP é sinalizado pelo LED FBx da interface correspondente do barramento de campo. Os estados do LED FBx são mostrados na seguinte tabela:

LED FBx	Cor	Descrição
DESLIGA	Amarelo	O protocolo PROFIBUS-DP não está ativo! Ou seja, o sistema de comando está no estado STOP ou não há nenhum slave PROFIBUS DP configurado.
Piscando em ritmo de 2 segundos	Amarelo	Não há troca de dados! O slave PROFIBUS DP está configurado e pronto.
LIGA	Amarelo	O protocolo PROFIBUS-DP está ativo e está trocando dados com o master PROFIBUS DP.

Tabela 163: LED FBx

### 6.17 Função do LED FAULT no slave PROFIBUS DP (Apenas HIMax)

A avaria do protocolo PROFIBUS-DP é sinalizada pelo LED FAULT da interface correspondente do barramento de campo. Os estados do LED FAULT são mostrados na seguinte tabela:

LED FAULT	Cor	Descrição
DESLIGA	Vermelho	Protocolo PROFIBUS-DP sem avaria.
Piscando	Vermelho	Protocolo avariado. <ul style="list-style-type: none"> <li>Há erros na configuração do master PROFIBUS DP e/ou do slave PROFIBUS DP.</li> <li>Tempo de processamento ultrapassado.</li> </ul> Se durante mais de 5 segundos não ocorrer nenhum evento de erro, a exibição muda para o estado <i>Protocol not disturbed</i> .

Tabela 164: FAULT FBx

## 7 Modbus

O acoplamento dos sistemas HIMax/HIMatrix ao Modbus pode ocorrer quase em todos os sistemas de gestão de processo ou de visualização via interfaces RS485 ou pelas interfaces Ethernet dos sistemas de comando. Os sistemas HIMax/HIMatrix podem ser operados tanto como master quanto como slave.

A funcionalidade Modbus facilita principalmente a vinculação de painéis de comando ou de outros sistemas de comando. Pela intensa distribuição e utilização em projetos no mundo inteiro, Modbus está amplamente comprovado na prática.

Modbus Master (veja Capítulo 7.1)

A redundância do master Modbus deve ser configurada no programa de aplicação de maneira que este monitore os caminhos de transporte redundantes e associe os dados de processo transmitidos de forma redundante ao respectivo caminho de transporte.

Modbus Slave (veja Capítulo 7.3)

O slave Modbus pode ser configurado de forma redundante.

i

Ao utilizar as interfaces Ethernet como canal de transporte, o sistema de comando HIMax/HIMatrix e o parceiro de comunicação devem estar na mesma subrede ou, no caso de usar um roteador, possuir os mesmos registros de roteamento.

### 7.1 Topologia de barramento RS485

A ilustração a seguir mostra a estrutura de uma topologia de barramento RS485 com componentes HIMA. Como bornes de barramento são usados H 7506. O comprimento total do barramento não pode exceder 1200 m. Distâncias maiores exigem a utilização de um repetidor, p. ex., H 7505<sup>1)</sup>. No total, podem ser utilizados 3 repetidores. Ou seja, o barramento pode ter uma extensão de no máximo 4800 m.

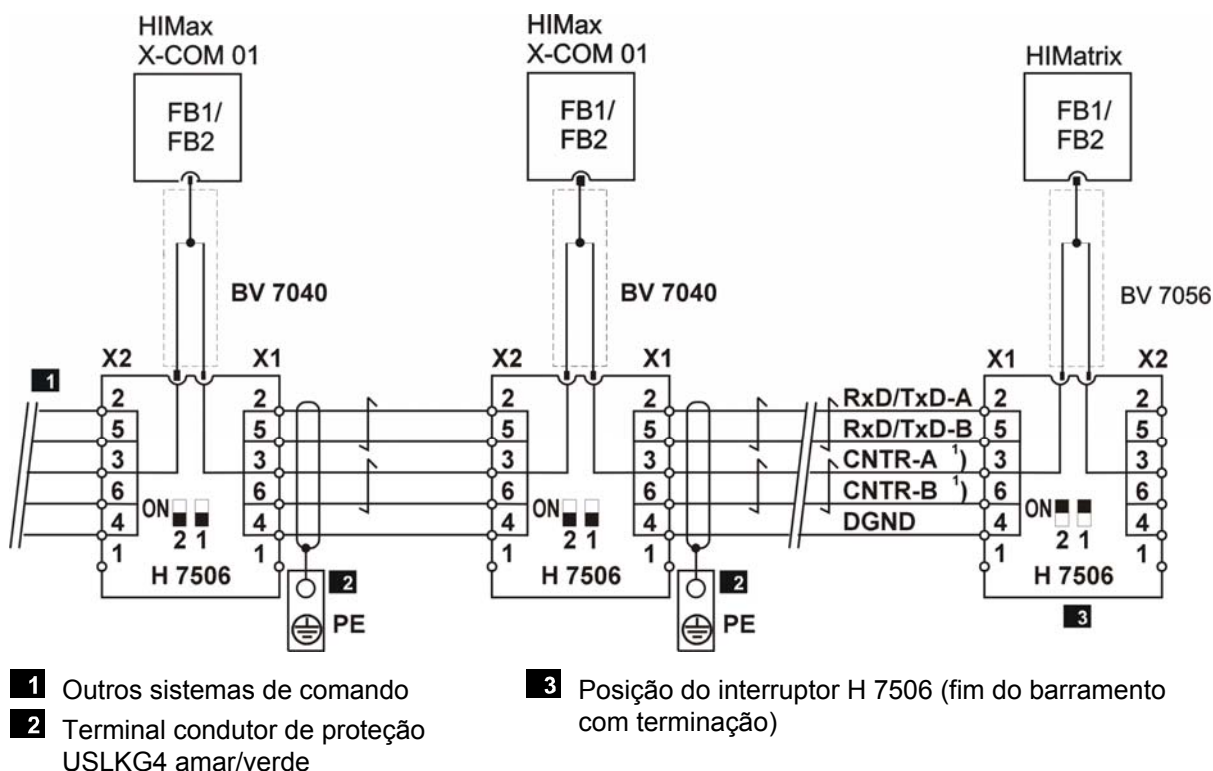


Figura 57: Topologia de barramento RS485

<sup>1)</sup> Se conversores de condutores de FO/RS485 são usados no barramento, o H7505 não pode ser utilizado (não há comutação automática da direção de dados).



Se o barramento for conduzido sobre distâncias maiores, deve ser efetuada uma ligação equipotencial.

### 7.1.1 Pinagem H 7506

A seguinte tabela mostra a pinagem do terminal de barramento HIMA H 7506. O cabo HIMA BV 7040 conecta o H 7506 com a interface de barramento de campo FBx do sistema de comando.

X1/X2	Cor	Descrição
1	-	-
2	branco	RxD/TxD-A, condutor de dados
3	verde	CNTR-A, condutor de comando para repetidor
4	cinza	DGND
5	marrom	RxD/TxD-B, condutor de dados
6	amarelo	CNTR-B, condutor de comando para repetidor

Tabela 165: Pinagem H 7506



Informações sobre este e outros componentes HIMA RS485 encontram-se na homepage da HIMA.

### 7.1.2 Cabo de barramento

Como cabo de barramento recomendamos um condutor de par trançado de dois fios blindado com as seguintes características:

Elemento	Descrição
Tipo de cabo	LiYCY 3 x 2 x 0,25 mm <sup>2</sup>
Seção transversal dos fios	> 0,25 mm <sup>2</sup>
Impedância característica	100...120 Ω

Tabela 166: Cabo de barramento

## 7.1.3 Características da transmissão RS485

Elemento	Descrição
Topologia de rede	Barramento linear, terminação ativa do barramento nas duas pontas
Meio	Condutor de par trançado de dois fios, blindado
Conectores de encaixe	Conectores de encaixe SUB-D de 9 pinos. Pinagem, veja Capítulo 3.6.
Participantes do barramento por segmento	32 participantes do barramento em cada segmento sem repetidor <sup>1)</sup>
Participante do barramento para cada barramento no total	1 Master Modbus, 3 Repetidores <sup>1)</sup> 121 Slaves Modbus
Comprimento máx. de um segmento barramento	1200 m por segmento
Comprimento máx. do barramento	4800 m, 4 Segmentos com 3 Repetidores <sup>1)</sup>
Taxa de Baud máx.	HIMax: 38400 bit/s HIMatrix: 115000 bit/s
<sup>1)</sup> Por repetidor utilizado reduz-se o número máximo de participantes do barramento neste segmento em 1. Isso significa que neste segmento podem ser operados no máximo 31 participantes do barramento. De acordo com a norma, no total são admissíveis três repetidores, assim que no máximo 121 slaves de Modbus por interface serial de um master Modbus podem ser conectados.	

Tabela 167: Características da transmissão RS485

## 7.2 Master HIMA Modbus

A transmissão de dados entre o master Modbus HIMA e os slaves Modbus, pode ocorrer via interface serial (RS485) ou TCP/UDP (Ethernet). Além disso, é possível usar o master Modbus HIMA como gateway (Modbus von TCP/UDP -> RS485).

### Equipamentos necessários e requisitos de sistema

Elemento	Descrição
Sistema de comando HIMA	HIMax com módulo COM HIMatrix a partir de CPU OS V7 e COM OS V12
Módulo CPU	As interfaces Ethernet do módulo processador não podem ser usadas para Modbus TCP.
Módulo COM	Ethernet 10/100BaseT Conexões D-Sub FB1 e FB2 Se for utilizado Modbus RTU, o módulo COM nas interfaces seriais de barramento de campo (FB1 e/ou FB2) deve estar equipado com um submódulo HIMA RS485 opcional. Atribuição de interfaces, veja Capítulo 3.6.
Ativação	Cada uma das duas funções Modbus Master deve ser ligada individualmente, veja Capítulo 3.4. Modbus Master RTU (RS 485) e Modbus Master TCP. O Modbus Gateway exige uma licença Modbus Master RTU.

Tabela 168: Equipamentos necessários e requisitos de sistema para Modbus Master

### Modbus Master Properties

Característica	Descrição
Modbus Master	Para cada módulo COM ou sistema de comando HIMatrix pode ser configurado um master Modbus. O master Modbus consegue simultaneamente <ul style="list-style-type: none"> <li>- trocar dados com TCP/UDP slaves</li> <li>- trocar dados com slaves seriais</li> <li>- e servir como gateway do Modbus TCP para o Modbus RTU.</li> </ul>
Quantidade máx. de slaves Modbus HIMax/HIMatrix	Um master Modbus pode atender até 247 slaves. <ul style="list-style-type: none"> <li>- 121 slaves Modbus por interface serial (FB1, FB2)</li> <li>- 64 slaves TCP via conexão TCP/IP</li> <li>- 247 slaves UDP via conexão UDP/IP</li> </ul> A quantidade máxima de slaves UDP é limitada, pois os slaves devem ser gerenciados no lado do master.
Quantidade máx. de telegramas de requisição	Podem ser configurados até 988 telegramas de requisição por master Modbus.
Comprimento máx. de dados de processo por telegrama de requisição	A comprimento dos dados de processo em telegramas de requisição específicos da HIMA é de 1100 bytes, veja Capítulo 7.2.4.2.
Tamanho máx. dos dados de envio	Veja Tabela 9 Protocolos padrão
Tamanho máx. dos dados de recepção	<div style="display: flex; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin-right: 10px;">1</div> <div>Os bytes de status do master e os Statusbytes de cada slave vinculado devem ser subtraídos do tamanho máx. dos dados de envio.</div> </div>

Característica	Descrição				
Formato de representação dos dados Modbus	Os sistemas de comando HIMax/HIMatrix usam o formato Big Endian. Exemplo de dados 32 Bit (p. ex., DWORD, DINT):				
	Dados 32 Bit (hex)		0x12345678		
	Offset de memória	0	1	2	3
	Big Endian (HIMax/HIMatrix)	12	34	56	78
	Middle Endian (H51q)	56	78	12	34
	Little Endian	78	56	34	12

Tabela 169: Características Modbus Master

Conforme a norma, no total três repetidores são admissíveis, assim, no máximo 121 slaves por interface serial de um master são possíveis.

7.2.1 Exemplo Modbus

Neste exemplo, um master Modbus HIMA troca dados com um slave Modbus HIMA via Modbus TCP. Os dois sistemas de comando são conectados pelas interfaces Ethernet dos módulos de comunicação.

i

Se o master Modbus e o slave Modbus estiverem em subredes diferentes, na tabela de roteamento devem ser introduzidas as respectivas rotas definidas pelo usuário.

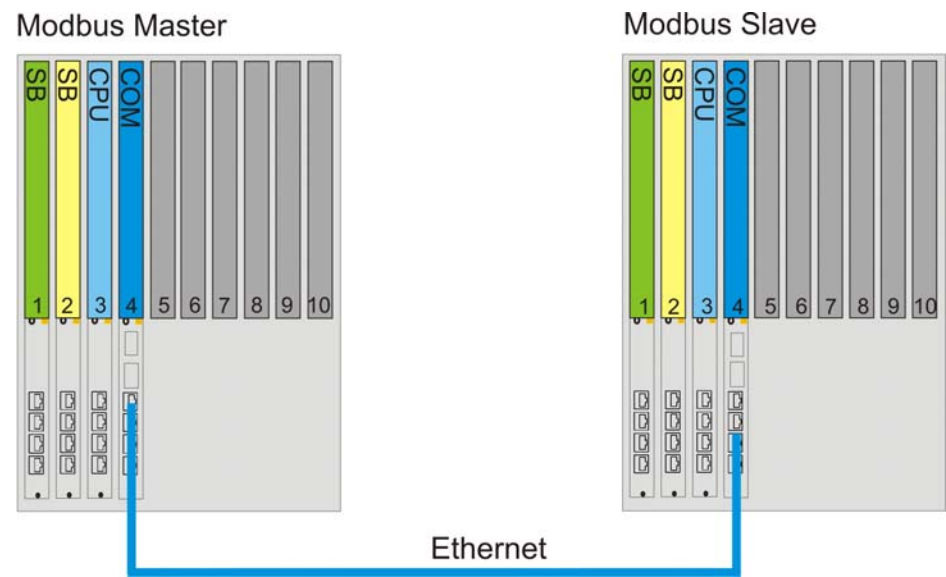


Figura 58: Comunicação via Modbus TCP

As seguintes variáveis globais devem ser criadas no SILworX para este exemplo:

Variáveis globais	Tipo
Master->Slave_BOOL_00	BOOL
Master->Slave_BOOL_01	BOOL
Master->Slave_BOOL_02	BOOL
Master->Slave_WORD_00	WORD
Master->Slave_WORD_01	WORD
Slave->Master_WORD_00	WORD
Slave->Master_WORD_01	WORD



### 7.2.1.1 Configuração do slave Modbus TCP

#### Assim cria-se um novo slave Modbus HIMA:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. No menu de contexto de protocolos, seleccionar **New, Modbus Slave Set** para adicionar um novo Modbus Slave Set.
3. No menu de contexto do Modbus Slave Set, seleccionar **Edit** e abrir **Modbus Slave Set Properties**, manter os valores padrão.
4. Seleccionar o registo **Modbus Slave** e efetuar os seguintes ajustes:
  - **Seleccionar COM-Modules**
  - Ativar Enable TCP
  - Os demais parâmetros mantêm os valores padrão.

#### Assim configuram-se as variáveis de entrada de bit do slave Modbus:



No registo variáveis de bit devem ser introduzidas as variáveis booleanas que o master endereça por bit (código de função 1, 2, 5, 15).

---

1. No menu de contexto do slave Modbus, seleccionar **Edit, Bit Variables**.
2. No **Object Panel**, seleccionar as seguintes variáveis globais e puxar mediante Drag&Drop para a área **Bit Inputs**.

Endereço bit	Variável bit	Tipo
0	Master->Slave_BOOL_00	BOOL
1	Master->Slave_BOOL_01	BOOL
2	Master->Slave_BOOL_02	BOOL

3. Abrir o menu de contexto com clique direito num lugar vazio na área **Register Inputs** e seleccionar **New Offsets** para renumerar os offsets das variáveis.

#### Assim configuram-se as variáveis de entrada de registo do slave Modbus:



No registo variáveis de registo devem ser introduzidas as variáveis que o master endereça por registo (código de função 3, 4, 6, 16, 23).

---

1. No menu de contexto do slave Modbus, seleccionar **Edit, Register Variables**.
2. No **Object Panel**, seleccionar as seguintes variáveis globais e puxar mediante Drag&Drop para a área **Register Inputs**.

Endereço do registo	Variáveis do tipo Register	Tipo
0	Master->Slave_WORD_00	WORD
1	Master->Slave_WORD_01	WORD

3. Abrir o menu de contexto com clique direito num lugar vazio na área **Register Inputs** e seleccionar **New Offsets** para renumerar os offsets das variáveis.

#### Assim configuram-se as variáveis de saída de registo do slave Modbus:

1. No menu de contexto do slave Modbus, seleccionar **Edit, Register Variables**.
2. No Object Panel, seleccionar as seguintes variáveis globais e puxar mediante Drag&Drop para a área **Register Outputs**.

Endereço do registro	Variáveis do tipo Register	Tipo
0	Slave->Master_WORD_00	WORD
1	Slave->Master_WORD_01	WORD

3. Abrir o menu de contexto com clique direito num lugar vazio na área **Register Outputs** e selecionar **New Offsets** para renumerar os offsets das variáveis.

**Assim verifica-se a configuração do slave Modbus TCP:**

1. Abrir o menu de contexto do slave Modbus TCP e selecionar **Verification**.
2. Verificar cuidadosamente as entradas no log, corrigir se for necessário.

### 7.2.1.2 Configuração do master Modbus TCP

**Assim cria-se o master Modbus HIMA:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. No menu de contexto de protocolos, selecionar **New, Modbus Master** para adicionar um novo master Modbus.
3. No menu de contexto do master Modbus, selecionar **Properties, General**.
4. Selecionar **COM-Module**.  
Os demais parâmetros mantêm os valores padrão.

**Assim cria-se no master Modbus a conexão ao slave Modbus TCP:**

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master, Ethernet Slaves**.
2. Clicar com o botão direito em **Ethernet Slaves** e selecionar no menu de contexto **New**.
3. Selecionar da lista **TCP/UDP-Slave** e confirmar com **OK**.
4. Configuração do slave TCP/UDP no master Modbus:
  - Selecionar **Edit** para atribuir as variáveis de sistema, veja Capítulo 7.2.6.2.
  - Selecionar **Properties** para configurar as características, veja Capítulo 7.2.6.3.
 Nas características do slave deve ser introduzido o **IP Address** do slave TCP/UDP.

Os demais parâmetros mantêm os valores padrão.

**Assim configura-se o telegrama de requisição para escrever as variáveis de saída de bit:**

1. Clicar com o botão direito em **TCP/UDP-Slave** e selecionar no menu de contexto **New**.
2. Selecionar da lista o telegrama de requisição **Write Multiple Coils (15)**.
3. Clicar com o botão direito no telegrama de requisição **Write Multiple Coils (15)** e selecionar no menu de contexto **Properties**.
  - Introduzir Start address of the write area, **0**.
4. Clicar com o botão direito no telegrama de requisição **Write Multiple Coils (15)** e selecionar no menu de contexto **Edit**.
5. No **Object Panel**, selecionar as seguintes variáveis e puxar mediante Drag&Drop para o registro **Output Variables**.

Offset	Variáveis de bit	Tipo
0	Master->Slave_BOOL_00	BOOL
1	Master->Slave_BOOL_01	BOOL
2	Master->Slave_BOOL_02	BOOL

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Variables** e selecionar **New Offsets** para renumerar os offsets das variáveis.

**Assim configura-se o telegrama de requisição para escrever as variáveis de saída de registro:**

1. Clicar com o botão direito em **TCP/UDP Slave** e seleccionar no menu de contexto **New**.
2. Seleccionar da lista o telegrama de requisição **Write Multiple Registers (16)**.
3. Clicar com o botão direito no telegrama de requisição **Write Multiple Registers (16)** e seleccionar no menu de contexto **Properties**.
  - Introduzir Start address of the write area, **0**.
4. Clicar com o botão direito no telegrama de requisição **Write Multiple Registers (16)** e seleccionar no menu de contexto **Edit**.
5. No **Object Panel**, seleccionar as seguintes variáveis e puxar mediante Drag&Drop para o registro **Output Variables**.

Offset	Variáveis do tipo Register	Tipo
0	Master->Slave_WORD_00	WORD
1	Master->Slave_WORD_01	WORD

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Variables** e seleccionar **New Offsets** para renumerar os offsets das variáveis.

**Assim define-se no master Modbus o telegrama de requisição para a leitura das variáveis de entrada:**

1. Clicar com o botão direito em **TCP/UDP Slave** e seleccionar no menu de contexto **New**.
2. Seleccionar da lista o telegrama de requisição **Read Holding Registers (03)**.
3. Clicar com o botão direito no telegrama de requisição **Read Holding Registers (03)** e seleccionar no menu de contexto **Properties**.
  - **Introduzir o endereço inicial da área de leitura, 0.**
4. Clicar com o botão direito no telegrama de requisição **Read Holding Registers (03)** e seleccionar no menu de contexto **Edit**.
5. No **Object Panel**, seleccionar as seguintes variáveis e puxar mediante Drag&Drop para o registro **Input Variables**.

Offset	Variáveis do tipo Register	Tipo
0	Slave->Master_WORD_00	WORD
1	Slave->Master_WORD_01	WORD

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Input Variables** e seleccionar **New Offsets** para renumerar os offsets das variáveis.

**Assim verifica-se a configuração do master Modbus TCP:**

1. Abrir o menu de contexto do master Modbus TCP e seleccionar **Verification**.

**Assim verifica-se a configuração do master Modbus TCP:**

1. Abrir o menu de contexto do master Modbus TCP e seleccionar **Verification**.
2. Verificar cuidadosamente as entradas no log, corrigir se for necessário.

**Assim cria-se o código para os sistemas de comando**

1. Iniciar o Code Generator do recurso master e slave.
2. Certificar-se de que os códigos tenham sido gerados sem erros.
3. Carregar os respectivos códigos para os sistemas de comando do master e slave.

### 7.2.2 Exemplo para o endereçamento alternativo de Register/Bit

Neste exemplo, a configuração do Capítulo 7.2.1 é estendido por 16 variáveis booleanas na área de registro. As 16 variáveis booleanas são lidas pelo telegrama de requisição **Write Multiple Coils (15)**, veja também Capítulo 7.3.11.

**Assim configuram-se as variáveis de entrada do slave Modbus:**

1. No menu de contexto do slave Modbus, selecionar **Edit, Register Variables**.
2. No **Object Panel**, selecionar as 16 novas variáveis booleanas e puxar mediante Drag&Drop para a área **Register Inputs**.

Endereço do registro	Variáveis do tipo Register	Tipo
0	Master->Slave_WORD_00	WORD
1	Master->Slave_WORD_01	WORD
2	Master->Slave_BOOL_03 ..._18	BOOL

Adicionar novo

3. Abrir o menu de contexto com clique direito num lugar vazio na área **Register Inputs** e selecionar **New Offsets** para renumerar os offsets das variáveis.

**Assim configura-se no slave Modbus o endereçamento alternativo de Register/Bit:**

1. Selecionar no menu de contexto do slave Modbus **Edit, Offsets** e ativar **Use Alternative Register/Bit Addressing**.
2. Utilizar para este exemplo os seguintes offsets para as áreas alternativas:

Register Area Offset Bits Input	1000
Register Area Offset Bits Output	1000
Bit Area Offset Register Input	8000
Bit Area Offset Register Output	8000

#### i

Para poder acessar com o telegrama de requisição Modbus **Write Multiple Coils (15)** variáveis booleanas na área **Register Variables**, as variáveis devem ser espelhadas para a área **Bit Variables**.

**Assim configura-se no master Modbus o telegrama de requisição para escrever as variáveis de saída (BOOL):**

1. Clicar com o botão direito em **TCP/UDP-Slave** e selecionar no menu de contexto **New**.
2. Selecionar da lista o telegrama de requisição **Write Multiple Coils (15)**.
3. Clicar com o botão direito no telegrama de requisição **Write Multiple Coils (15)** e selecionar no menu de contexto **Properties**.
  - Introduzir Start address of the write area, **8032**.
4. Clicar com o botão direito no telegrama de requisição **Write Multiple Coils (15)** e selecionar no menu de contexto **Edit**.
5. No **Object Panel**, selecionar as seguintes variáveis e puxar mediante Drag&Drop para o registro **Output Variables**.

Offset	Variável Register espelhada	Tipo
0 a 15	Master->Slave_BOOL_03..._18	BOOL

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Variables** e selecionar **New Offsets** para renumerar os offsets das variáveis.

### 7.2.3 Funções de menu do master Modbus HIMA

#### 7.2.3.1 Edit

A janela de diálogo **Edit** do master Modbus contém o seguinte registro.

##### Variáveis de sistema

O registro **System Variables** disponibiliza variáveis de sistema que permitem avaliar o estado do master Modbus no programa de aplicação e controlar o master Modbus.

Elemento	Descrição
Slave Connection Error Count	Quantidade de conexões com erro aos slaves Modbus que estão no estado "Ativado". Slaves Modbus desativados não são considerados aqui.
Modbus Master Activation Control	Com isso, o master Modbus pode ser parado ou iniciado pelo programa de aplicação. 0: Ativar 1: Desativar (Disparado por flanco! O master Modbus pode ser ativado via PADT mesmo quando o controle de ativação do master Modbus = 1.)
Modbus Master Bus Error	Erro de barramento em RS485, p. ex., erro de telegrama (códigos desconhecidos, etc.) erro de comprimento.
Modbus Master State	O estado do master Modbus mostra o estado atual do protocolo: 1: OPERATE 0: OFFLINE
Reset All Slave Errors	Com uma mudança de FALSE->TRUE, todos os erros de slave e erros de barramento são resetados.

Tabela 170: Variáveis de sistema master Modbus

### 7.2.3.2 Características

A função de menu **Properties** do menu de contexto do master Modbus abre o diálogo *Properties*.

O diálogo contém os seguintes registros:

#### Informações gerais

No registro **General** são introduzidos o nome e a descrição para o master Modbus. Além disso, aqui são ajustados os parâmetros se o master Modbus deve adicionalmente trabalhar como gateway TCP e/ou UDP.

Elemento	Descrição
Type	Modbus Master
Name	Nome para o master Modbus
Description	Descrição para o master Modbus
Module	Seleção do módulo COM no qual este protocolo é processado.
Activate Max. $\mu$ P Budget	Ativado: Transferir o limite do $\mu$ P Budget do campo Max. $\mu$ P-Budget in [%].  Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P Budget in [%]	Valor máximo de $\mu$ P-Budget do módulo que pode ser produzido ao processar o protocolo.  Faixa de valores: 1...100% Valor padrão: 30%
Behavior on CPU/COM Connection Loss	No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. (P. ex., se o módulo de comunicação é puxado para fora durante a comunicação). Adopt Initial Data      As variáveis de entrada são resetadas para os valores iniciais. Retain Last Value      As variáveis de entrada mantêm o último valor.
Enable TCP Gateway	Se o gateway Modbus TCP estiver ativado, deve ser configurado no mínimo uma interface RS485 Modbus.
TCP Server Port	Padrão: 502 Também podem ser configurados outras portas TCP. Neste caso, deve ser observada a atribuição de portas da <i>Internet Corporation for Assigned Names and Numbers</i> (ICANN).
Maximum Number of TCP Connections Operating as Server	Quantidade máxima de conexões TCP como servidor simultaneamente abertas.  Faixa de valores: 1...64 Valor padrão: 5
Enable UDP Gateway	Se o gateway Modbus UDP estiver ativado, deve ser configurado no mínimo uma interface RS485 Modbus
UDP Port	Padrão: 502 Também podem ser configurados outras portas UDP. Neste caso, deve ser observada a atribuição de portas da <i>Internet Corporation for Assigned Names and Numbers</i> (ICANN).

Elemento	Descrição
Maximum length of the queue	Comprimento da fila de espera do gateway para telegramas de requisição ainda não respondidos por outros masters. Isso apenas é considerado se um gateway estiver sido ativado.  Faixa de valores: 1...20 Valor padrão: 3

Tabela 171: Características gerais do Modbus Master

### CPU/COM

Os valores padrão para os parâmetros garantem a troca de dados mais rápida possível dos dados Modbus entre o módulo COM e o módulo CPU no sistema de comando HIMax/HIMatrix.

Estes parâmetros apenas devem ser alterados se uma redução da carga COM e/ou CPU é necessária para uma aplicação e se o processo permitir.

### i

A alteração dos parâmetros apenas se recomenda para programadores experientes. Um aumento do tempo de atualização COM e CPU também significa que o tempo de atualização real dos dados Modbus aumenta. As requisições de tempo do sistema devem ser verificadas.

Elemento	Descrição
Process data refresh rate [ms]	Tempo de atualização em milissegundos com o qual os dados do protocolo são trocados entre COM e CPU. Se o <i>Process Data Refresh Rate [ms]</i> for zero ou menor do que o tempo de ciclo do sistema de comando, a troca de dados ocorre o mais rápido possível.  Faixa de valores: 0...(2 <sup>31</sup> - 1) Valor padrão: 0
Force Process Data Consistency	Ativado: Transferência dos dados completos do protocolo de CPU para COM dentro de um ciclo de CPU.  Desativado: Transferência dos dados completos do protocolo de CPU para COM, distribuídos sobre vários ciclos de CPU cada vez 1100 Byte por direção de dados. Com isso, eventualmente também é possível reduzir o tempo de ciclo do sistema de comando.  Valor padrão: Ativado

Tabela 172: Parametros COM/CPU

### 7.2.4 Códigos de função Modbus do master

Com os códigos de função (telegramas de requisição) existe a possibilidade de escrever e ler variáveis nas duas direções. É possível ler ou escrever variáveis individuais ou várias variáveis em sequência.

#### Assim cria-se um novo telegrama de requisição para um slave TCP/UDP:

1. Na árvore de estrutura **Resource, Protocols, Modbus Master, Ethernet Slaves** selecionar um **TCP/UDP Slave**.
2. Clicar com o botão direito em **TCP/UDP Slave** e selecionar no menu de contexto **New**.
3. Selecionar no diálogo **New Object** um **Request Telegram**.

#### Assim cria-se um novo telegrama de requisição para um slave de gateway:

1. Na árvore de estrutura **Resource, Protocols, Modbus Master, Modbus gateway** selecionar um **Gateway Slave**.
2. Clicar com o botão direito em **Gateway Slave** e selecionar no menu de contexto **New**.
3. Selecionar no diálogo **New Object** um **Request Telegram**.

#### Assim cria-se um novo telegrama de requisição para um slave Modbus RS485:

1. Na árvore de estrutura **Resource, Protocols, Modbus Master, Serial Modbus** selecionar um **Modbus Slave**.
2. Clicar com o botão direito em **Modbus Slave** e selecionar no menu de contexto **New**.
3. Selecionar no diálogo **New Object** um **Request Telegram**.

#### 7.2.4.1 Códigos de função Modbus padrão

Os seguintes códigos de função Modbus padrão são suportados pelo master Modbus HIMA.

Elemento	Código	Type	Significado
READ COILS	01	BOOL	Ler várias variáveis (BOOL) do slave.
READ DISCRETE INPUTS	02	BOOL	Ler várias variáveis (BOOL) do slave.
READ HOLDING REGISTERS	03	WORD	Ler várias variáveis de tipo livre do slave.
READ INPUT REGISTERS	04	WORD	Ler várias variáveis de tipo livre do slave.
WRITE SINGLE COIL	05	BOOL	Escrever um sinal individual (BOOL) para o slave.
WRITE SINGLE REGISTER	06	WORD	Escrever um sinal individual (WORD) para o slave.
WRITE MULTIPLE COILS	15	BOOL	Escrever várias variáveis (BOOL) para o slave.
WRITE MULTIPLE REGISTERS	16	WORD	Escrever várias variáveis de tipo livre para o slave.
READ WRITE HOLDING REGISTERS	23	WORD	Ler e escrever várias variáveis de tipo livre do e para o slave.

Tabela 173: Códigos de função Modbus



Informações mais detalhadas sobre o Modbus encontram-se na especificação *Modbus Application Protocol Specification* [www.modbus.org](http://www.modbus.org)



#### 7.2.4.2 Códigos de função específicos HIMA

Os códigos de função específicos HIMA correspondem aos códigos de função Modbus padrão. As duas diferenças são o comprimento dos dados de processo máximo admissível de 1100 bytes e o formato do Request e Response-Header.

Elemento	Código	Tipo	Significado
Read Coils Extended	100 (0x64)	BOOL	Corresponde ao Functioncode 01 Ler várias variáveis (BOOL) da área de importação ou exportação <sup>1)</sup> do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read Discrete Inputs Extended	101 (0x65)	BOOL	Corresponde ao Functioncode 02 Ler várias variáveis (BOOL) da área de exportação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read Holding Registers Extended	102 (0x66)	WORD	Corresponde ao Functioncode 03 Ler várias variáveis de tipo livre da área de importação ou exportação <sup>1)</sup> do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read Input Registers Extended	103 (0x67)	WORD	Corresponde ao Functioncode 04 Ler várias variáveis de tipo livre da área de exportação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Write Multiple Coils Extended	104 (0x68)	BOOL	Corresponde ao Functioncode 15 Escrever várias variáveis (BOOL) para a área de importação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Write Multiple Registers Extended	105 (0x69)	WORD	Corresponde ao Functioncode 16 Escrever várias variáveis de tipo livre para a área de importação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read/Write Multiple Registers Extended	106 (0x6A)	WORD	Corresponde ao Functioncode 23 Escrever e ler várias variáveis de tipo livre da e para a área de importação ou exportação do slave. Comprimento máximo dos dados de processo: 1100 bytes (telegrama de requisição do master Modbus) 1100 bytes (resposta ao master).

### 7.2.4.3 Formato dos cabeçalhos para Request e Response

Os cabeçalhos de Request e Response dos códigos de função específicos HIMA possuem a seguinte estrutura:

Código	Request	Response
100 (0x64)	1 byte código de função 0x64 2 bytes endereço inicial 2 bytes quantidade de Coils 1...8800(0x2260)	1 byte código de função 0x64 2 bytes quantidade de Bytes = N N bytes dados Coil (8 Coils são empacotados em um Byte)
101 (0x65)	1 byte código de função 0x65 2 bytes endereço inicial 2 bytes quantidade de Discrete Inputs 1...8800(0x2260)	1 byte código de função 0x65 2 bytes quantidade de Bytes = N N bytes dados Discrete Inputs (8 Discrete Inputs são empacotados em um Byte)
102 (0x66)	1 byte código de função 0x66 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226)	1 byte código de função 0x66 2 bytes quantidade de Bytes = N N bytes dados de registro
103 (0x67)	1 bytes código de função 0x67 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226)	1 byte código de função 0x67 2 bytes quantidade de Bytes = N N bytes dados de registro
104 (0x68)	1 byte código de função 0x68 2 bytes endereço inicial 2 bytes quantidade de Coils 1...8800(0x2260) 2 bytes quantidade de Bytes = N N bytes dados Coil	1 byte código de função 0x68 2 bytes endereço inicial 2 bytes quantidade de Coils 1...8800(0x2260)
105 (0x69)	1 byte código de função 0x69 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226) 2 bytes quantidade de Bytes = N N bytes dados de registro	1 byte código de função 0x69 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226)
106 (0x6A)	1 byte código de função 0x6a 2 bytes endereço inicial leitura 2 bytes quantidade de registros de leitura 1...550(0x226) 2 bytes endereço inicial escrita 2 bytes quantidade de registros de escrita 1...550(0x226) 2 bytes quantidade de bytes para escrever = N N bytes dados de registro	1 byte código de função 0x6a 2 bytes quantidade de Bytes = N N bytes dados de registro

#### 7.2.4.4 Telegramas de requisição para leitura

Com os códigos de função Read, é possível ler variáveis do slave.

Um telegrama de requisição do master Modbus contém além da função Modbus o endereço inicial da área de leitura/escrita.

Para a leitura de variáveis, o master Modbus envia um *Read Request Telegram* to the Modbus Slave ao slave Modbus.

O slave Modbus, então, envia um telegrama de resposta com as variáveis solicitadas ao master Modbus.

##### Assim configura-se um telegrama de requisição para leitura:

1. Selecionar na árvore de estrutura **Request Telegram** para configurar.
2. Clicar com o botão direito em **Request Telegram** e selecionar no menu de contexto **Edit**.
3. Selecionar no **Object Panel** uma variável global que deve servir como variável de recepção Modbus e puxar a mesma mediante Drag&Drop para um local vazio na área **Inputs Signals**.
4. Repetir este passo para todas as demais variáveis de recepção Modbus.
5. Abrir o menu de contexto com clique direito num lugar vazio na área **Input Signals** e selecionar **New Offsets** para renumerar os offsets das variáveis.

##### Os seguintes *Read Request Telegrams* estão à disposição:

##### Read Coils (01) e Extended (100)

Ler várias variáveis (BOOL) do slave.

Elemento	Significado
Type	Função Modbus Read Coils
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the read area	0...65535

Tabela 174: Telegrama de requisição Read Coils

##### Read Discrete Inputs (02) e Extended (101)

Ler várias variáveis (BOOL) do slave.

Elemento	Significado
Type	Função Modbus Read Discrete Inputs
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the read area	0...65535

Tabela 175: Telegrama de requisição Read Discrete Inputs

##### Read Holding Registers (03) e Extended (102)

Ler várias variáveis de tipo livre do slave.

Elemento	Significado
Type	Função Modbus Read Holding Registers
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the read area	0...65535

Tabela 176: Telegrama de requisição Read Holding Registers

## Read Input Registers (04) e Extended (103)

Ler várias variáveis de tipo livre do slave

Elemento	Significado
Type	Função Modbus Read Input Registers
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the read area	0...65535

Tabela 177: Telegrama de requisição Read Input Registers

### 7.2.4.5 Telegrama de requisição para leitura e escrita

Para a leitura e escrita de variáveis, o master Modbus envia um *Read/Write Request Telegram* to the Modbus Slave ao slave Modbus.

Primeiramente o master Modbus escreve as variáveis de escrita definidas para a área de importação definida do slave Modbus.

A seguir, o master Modbus lê as variáveis de leitura da área de exportação definida do slave Modbus.

i

As funções escrever e ler também no *Read/Write Request Telegram* dependem uma da outra, apenas são enviadas num telegrama de requisição conjunto.

Uma aplicação frequente para o *Read/Write Request Telegram*, porém, é que as variáveis escritas do master Modbus podem ser lidas de volta novamente. Assim é verificado se as variáveis enviadas foram escritas corretamente.

#### Assim configura-se um telegrama de requisição para leitura e escrita:

1. Selecionar na árvore de estrutura **Request Telegram** para configurar.
2. Clicar com o botão direito em **Request Telegram** e selecionar no menu de contexto **Edit**.

#### Assim configura-se a variável para leitura:

1. Selecionar no **Object Panel** uma variável global que queira conectar com a nova variável de recepção Modbus e puxar a mesma mediante Drag&Drop para a coluna **Global Variable** das variáveis de recepção Modbus.
2. Repetir este passo para todas as demais variáveis de recepção Modbus.
3. Abrir o menu de contexto com clique direito num lugar vazio na área Input Signals e selecionar **New Offsets** para renumerar os offsets das variáveis.

#### Assim configura-se as variáveis para escrita:

1. Selecionar no **Object Panel** uma variável global que queira conectar com a nova variável de recepção Modbus e puxar a mesma mediante Drag&Drop para a coluna **Global Variable** das variáveis de envio Modbus.
2. Repetir este passo para todas as demais variáveis de envio Modbus.
3. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Signals** e selecionar **New Offsets** para renumerar os offsets das variáveis.

## Read Write Holding Register (23) e Extended (106)

Escrever e ler várias variáveis de tipo livre da e para a área de importação do slave.

Elemento	Significado
Type	Função Modbus <i>Read Write Holding Registers</i>
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the read area	0...65535
Start address of the write area	0...65535

Tabela 178: Registro Read Write Holding

## 7.2.4.6 Telegrama de requisição para escrita

Com os códigos de função Write as variáveis apenas são escritas para a área de importação de um slave.

Um telegrama de requisição do master Modbus contém além da função Modbus o endereço inicial da área de leitura/escrita.

Para a escrita de variáveis, o master Modbus envia um *Write Request Telegram* ao slave Modbus.

O slave Modbus escreve as variáveis recebidas para a sua área de importação.

No diálogo *Variable Connections* de um *write request telegram* devem ser inseridas as variáveis que o master Modbus escreve ao slave Modbus.

**Assim configura-se um telegrama de requisição para escrita:**

1. Selecionar na árvore de estrutura Request Telegram para configurar.
2. Clicar com o botão direito em **Request Telegram** e selecionar no menu de contexto **Edit**.
3. Selecionar no **Object Panel** uma variável global que deve servir como variável de envio Modbus e puxar a mesma mediante Drag&Drop para um local vazio na área **Send Signals**.
4. Repetir este passo para todas as demais variáveis de envio Modbus.
5. Abrir o menu de contexto com clique direito num lugar vazio na área Send Signals e selecionar **New Offsets** para renumerar os offsets das variáveis.

**Os seguintes *Write Request Telegrams* estão à disposição:**

## Write Multiple Coils (15) e Extended (104)

Escrever várias variáveis (BOOL) para a área de importação do slave.

Elemento	Significado
Type	Função Modbus <i>Write Multiple Coils</i>
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the write area	0...65535

Tabela 179: Telegrama de requisição Write Multiple Coils

### Write Multiple Registers (16) e Extended (105)

Escrever várias variáveis de tipo livre para a área de importação do slave.

Elemento	Significado
Type	Função Modbus <i>Write Multiple Registers</i>
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the write area	0...65535

Tabela 180: Telegrama de requisição Write Multiple Registers

### Write Single Coil (05)

Escrever uma única variável (BOOL) para a área de importação do slave.

Elemento	Significado
Type	Função Modbus <i>Write Single Coil</i>
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the write area	0...65535

Tabela 181: Telegrama de requisição Write Single Coil (05)

### Write Single Register (06)

Escrever uma única variável (WORD) para a área de importação do slave.

Elemento	Significado
Type	Função Modbus <i>Write Single Register</i>
Name	Nome livre inequívoco para a função Modbus
Description	Descrição da função Modbus
Start address of the write area	0...65535

Tabela 182: Telegrama de requisição Write Single Register

### 7.2.5 Ethernet Slaves (TCP/UDP-Slaves)

O master Modbus pode comunicar-se com até 64 slaves TCP/IP e 247 slaves UDP/IP.

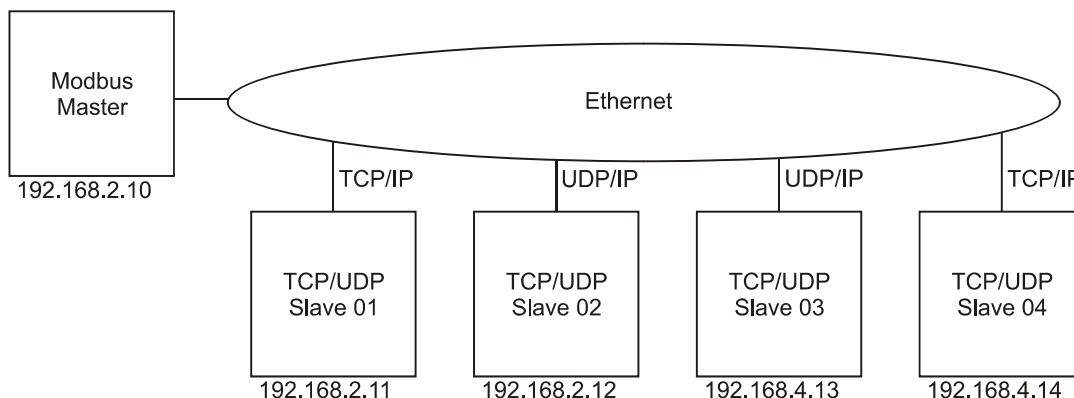


Figura 59: Rede Modbus

**Assim cria-se no master Modbus uma nova conexão a um slave TCP/UDP:**

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master, Ethernet-Slaves**.
2. Clicar com o botão direito em **Ethernet Slaves** e selecionar no menu de contexto **New**.
3. Selecionar da lista **TCP/UDP-Slaves** e confirmar com **OK**.
4. Configuração do slave TCP/UDP no master Modbus:
  - Edit** para atribuir as variáveis de sistema, veja Capítulo 7.2.5.1.
  - Properties** para configurar as características, veja Capítulo 7.2.5.2.

**i**

Se os slaves TCP/UDP e o master Modbus estiverem em subredes diferentes, na tabela de roteamento devem ser introduzidas as respectivas rotas definidas pelo usuário.

O master HIMax/HIMatrix Modbus TCP sempre envia com os seus telegramas ao slave Modbus TCP adicionalmente ao endereço IP um endereço slave Modbus (Unit identifier), este sempre é FF (255).

### 7.2.5.1 Variáveis de sistema do slave TCP/UDP

O registro *System Variables* disponibiliza variáveis de sistema que permitem avaliar e controlar o estado do slave TCP/UDP no programa de aplicação.

O status do slave TCP/UDP pode ser avaliado no programa de aplicação mediante as seguintes variáveis de status:

Elemento	Descrição
Modbus Slave Activation Control	Com isso, o slave TCP/UDP pode ser desativado ou ativado pelo programa de aplicação. 0: Ativar 1: Desativar (Disparado por flanco! Slave Modbus Slave pode ser ativado via PADT mesmo se <i>Modbus Slave Activation Control</i> = 1.)
Modbus Slave Error	Error code Os códigos de erro 0x01..0x0b correspondem aos Exception Codes da especificação do protocolo Modbus. 0x00: Sem erro  Exception Codes: 0x01: Código de função inválido 0x02: Endereçamento inválido 0x03: Dados inválidos 0x04: (Não usado) 0x05: (Não usado) 0x06: Device Busy (só Gateway) 0x08: (Não usado) 0x0a: (Não usado) 0x0b: No Response from Slave (só Gateway)  Códigos específicos HIMA: 0x10: Frame defeituoso recebido 0x11: Frame com Transaction ID incorreto recebido 0x12: Resposta inesperada recebida 0x13: Resposta recebida pela conexão errada 0x14: Resposta incorreta para um pedido de escrita  0xff: Slave Timeout
Modbus Slave State	Status de conexão do slave TCP/UDP: 0: Desativado 1: Não conectado 2: Conectado

Tabela 183: Variáveis de sistema slaves TCP/UDP



## 7.2.5.2 Características slaves TCP/UDP

Para a configuração da conexão ao slave TCP/UDP, devem ser ajustados os seguintes parâmetros no master Modbus.

Elemento	Descrição
Type	Slave TCP/UDP
Name	Nome livre inequívoco para o slave TCP/UDP
Description	Descrição livre inequívoca para o slave TCP/UDP
Master-Slave Data Exchange [ms]	Intervalo para a troca de dados com este slave 1 a ( $2^{31} - 1$ ). Se o slave não pôde ser acessado depois do <i>Maximal Number of Retries</i> , o intervalo <i>Master-Slave Data Exchange</i> é aumentado pelo fator quatro.
TCP connection only on demand	Se o protocolo de transporte for TCP, aqui é ajustado se a conexão a este slave deve ser automaticamente desfeita após cada troca de dados. TRUE: Desfazer a conexão. FALSE: Não desfazer a conexão. Valor padrão: FALSE
Receive Timeout [ms]	Receive Timeout para este slave [ms]. Depois deste tempo é iniciada uma nova tentativa de envio.
IP Address	Endereço IP do slave TCP/UDP
Port	Padrão: 502 Também podem ser configurados outras portas TCP/UDP. Neste caso, deve ser observada a atribuição de portas da Internet Corporation for Assigned Names and Numbers (ICANN).
Type of communication IP Protocol	TCP ou UDP Valor padrão: TCP
Maximum Number of Resends	Quantidade máxima de repetições de envio quando o slave não responde. A quantidade de repetições de envio pode ser ajustada livremente (0...65535). No caso de TCP/IP sempre é zero, não pode ser alterado. Recomenda-se uma quantidade de zero a oito repetições de envio.

Tabela 184: Parâmetros de configuração

### 7.2.6 Modbus Gateway (TCP/UDP Gateway)

Para que o master Modbus possa trabalhar como gateway Modbus, uma licença de master Modbus RTU é necessária. Neste modo, as Master Requests que o gateway recebe via Ethernet são repassadas aos slaves RS485 e/ou Ethernet conectados ao gateway. De forma correspondente, as respostas dos slaves são repassadas pelo gateway ao master Modbus.

Até 121 slaves Modbus seriais podem ser endereçados pela interface serial.

A área dos endereços de slave é 1 a 247. O master Modbus 2 (gateway Modbus) precisa de uma licença master Modbus mesmo se apenas o gateway Modbus for utilizado.

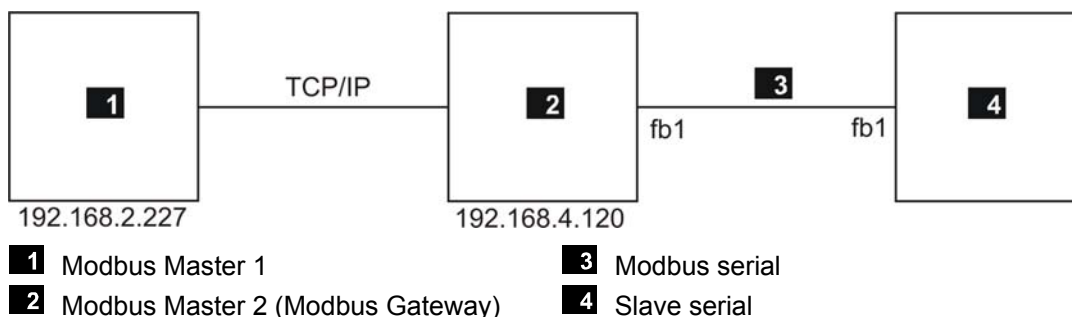


Figura 60: Modbus Gateway

i

Se o gateway Modbus e o master Modbus estiverem em subredes diferentes, na tabela de roteamento devem ser introduzidas as respectivas rotas definidas pelo usuário.

#### Modbus Master 1:

##### Assim cria-se no master Modbus 1 a conexão ao gateway Modbus:

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master**.
2. Clicar com o botão direito em **Modbus Master** e selecionar no menu de contexto **New**.
3. Selecionar da lista **Modbus Gateway** e confirmar com **OK**.
4. Configuração do gateway Modbus no Modbus Master 1:  
Selecionar **Properties** para configurar as características, veja Capítulo 7.2.7.3.  
Nas características, introduzir o **IP Address** do Modbus Master 2 (Modbus Gateway).

##### Assim cria-se no master Modbus 1 a conexão ao slave de gateway:

No Modbus Master 1 deve ser criado o slave serial como slave de gateway.

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master, Modbus Gateway**.
2. Clicar com o botão direito em **Modbus Gateway** e selecionar no menu de contexto **New**.
3. Selecionar da lista **Gateway Slave** e confirmar com **OK**.
4. Configuração do slave de gateway no Modbus Master 1:  
**Edit** para atribuir as variáveis de sistema, veja Capítulo 7.2.6.2.  
**Properties** para configurar as características, veja Capítulo 7.2.6.3.  
Nas características do slave deve ser introduzido o **Serial Address** do slave de gateway.

**Assim definem-se no Modbus Master 1 as variáveis de entrada e saída ao slave serial:**

1. Clicar com o botão direito em **Gateway Slave** e selecionar no menu de contexto **New**.
2. Selecionar da lista os **Request Telegram** necessários.
3. Clicar com o botão direito em **Request Telegram** e selecionar no menu de contexto **Edit**. No registro *Process Variables* são introduzidas as variáveis de entrada e/ou saída.

**Modbus Master 2 (Modbus Gateway):**

Nas características do Modbus Master 02 deve ser ativada a função de gateway. Assim, os slaves de gateway configurados no Master 01 são conectados aos slaves seriais.

**Assim ativa-se a função de gateway no Modbus Master 2:**

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master**.
2. Clicar com o botão direito em **Modbus Master** e selecionar no menu de contexto **Properties**.
3. Ligar o parâmetro **Enable TCP Gateway** para que o master Modbus possa trabalhar adicionalmente como gateway TCP.
4. Ligar o parâmetro **Enable UDP Gateway**, para que o master Modbus trabalhe adicionalmente como gateway UDP.

**Assim configura-se o Modbus serial no Modbus Master 2:**

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master**.
2. Clicar com o botão direito em **Modbus Master** e selecionar no menu de contexto **New**.
3. Selecionar da lista **Serial Modbus** e confirmar com **OK**.
4. Configuração do **Serial Modbus**  
Selecionar **Properties** e introduzir a interface. Baudrate, etc.

**Assim configura-se a conexão ao Modbus serial no Modbus Master 2:**

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master, Serial Modbus**.
2. Clicar com o botão direito em **Serial Modbus** e selecionar no menu de contexto **New**.
3. Selecionar da lista **Modbus Slave** e confirmar com **OK**.
4. Selecionar a configuração do **Modbus Slave**  
**Properties** e introduzir **Slave-Address** do slave serial.

**Slave serial**

**Assim configura-se o slave serial Modbus:**

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Slave**.
2. Clicar com o botão direito em **Modbus Slave** e selecionar no menu de contexto **Edit**.
3. Selecionar a configuração do **Modbus Slave**  
**Properties** e introduzir **Slave-Address** do slave serial.

### 7.2.6.1 Características do gateway

O master Modbus comunica-se com os seus slaves Modbus pelo gateway Modbus.

Para a configuração da conexão ao gateway Modbus, devem ser ajustados os seguintes parâmetros no master Modbus.

Elemento	Descrição
Type	Modbus Gateway
Name	Nome livre inequívoco para o gateway
Description	Descrição livre inequívoca para o slave TCP/UDP
Communication IP Protocol	TCP ou UDP Valor padrão: TCP
IP Address	Endereço IP do gateway pelo qual o master Modbus deve comunicar-se com os seus slaves Modbus. Valor padrão: (0.0.0.0)
Port	Valor padrão: 502

Tabela 185: Parâmetros de conexão gateway Modbus

### 7.2.6.2 Variáveis de sistema slave de gateway

No Editor, há três variáveis de status à disposição:

Elemento	Descrição
Modbus Slave Activation Control	Com isso, o slave de gateway pode ser desativado ou ativado pelo programa de aplicação. 0: Ativar 1: Desativar (Disparado por flanco! Slave Modbus Slave pode ser ativado via PADT mesmo se <i>Modbus Slave Activation Control</i> = 1.)
Modbus Slave Error	Parâmetros como no slave TCP/UDP, veja Capítulo 7.2.5.1.
Modbus Slave State	Status de conexão do slave de gateway: 0: Desativado 1: Não conectado 2: Conectado

Tabela 186: Variáveis de status slave de gateway

### 7.2.6.3 Características do slave de gateway

Para a configuração da conexão ao slave de gateway, devem ser ajustados os seguintes parâmetros no master Modbus.

Elemento	Descrição
Type	Gateway-Slave
Name	Nome livre inequívoco para o slave de gateway
Description	Descrição livre inequívoca para o slave de gateway
Slave Address	1...247
Demais parâmetros como no slave TCP/UDP, veja Capítulo 7.2.5.2	

Tabela 187: Parâmetros de conexão slave de gateway

### 7.2.7 Modbus serial

O master Modbus pode comunicar-se com até 247 slaves seriais. Conforme a norma, no total três repetidores são admissíveis, assim, no máximo 121 participantes do barramento por interface serial de um master são possíveis.

**i**

Para a pinagem das conexões D-Sub (FB1, FB2) do módulo X-COM, veja Capítulo 3.6.

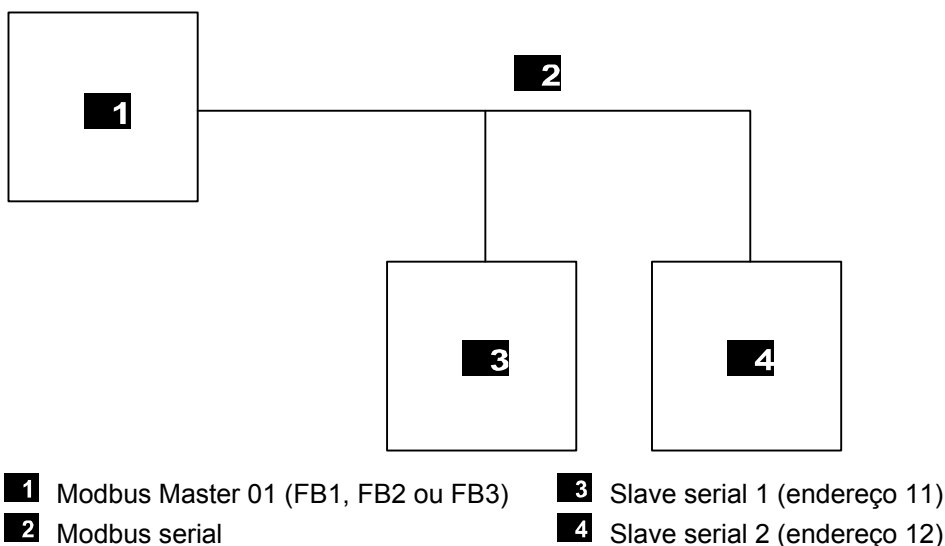


Figura 61: Modbus serial

O master Modbus HIMA suporta a transmissão de dados no formato RTU (Remote Terminal Unit).

O telegrama RTU inicia e termina com o caracter de Idle definido pelo usuário (valor padrão: 5 caracteres de Idle).

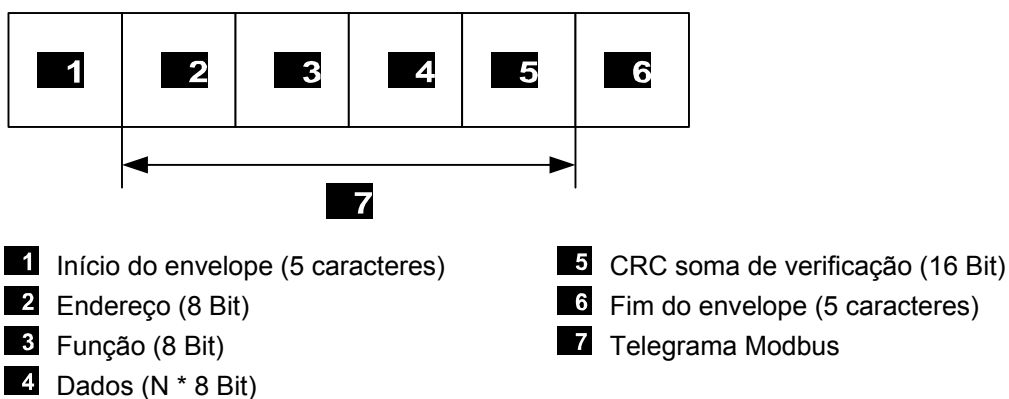


Figura 62: Telegrama Modbus

**Assim cria-se um Modbus serial no master Modbus:**

1. Abrir na árvore de estrutura **Resource, Protocols, Modbus Master, Serial Modbus**.
2. Clicar com o botão direito em **Serial Modbus** e selecionar no menu de contexto **New**.
3. Selecionar da lista **Modbus Slave** e confirmar com OK.
4. Configuração do slaves Modbus no master Modbus:  
 Selecionar **Edit** para atribuir as variáveis de sistema, veja 7.2.7.2.  
 Selecionar **Properties** para configurar as características, veja Capítulo 7.2.7.3.

**7.2.7.1 Características Modbus serial**

Para a configuração do Modbus serial no master Modbus, devem ser ajustados os seguintes parâmetros.

Elemento	Descrição
Type	Modbus serial
Name	O nome do Modbus serial pode ser escolhido pelo usuário
Description	Descrição livre inequívoca para o Modbus serial
Interface	A interface de barramento de campo que deve ser usada para o master Modbus (FB1, FB2).
Baud rate [bps]	Velocidade de transmissão para RS485 valores possíveis: 300 bit/s 600 bit/s 1200 bit/s 2400 bit/s 4800 bit/s 9600 bit/s 19200 bit/s 38400 bit/s 57600 bit/s (baud rate máximo HIMax a partir de V4) 62500 bit/s (só HIMatrix) 76800 bit/s (só HIMatrix) 115000 bit/s (só HIMatrix)
Parity	None: nenhuma Odd: impar Even: par Valor padrão: Even: par
Stop Bits	Padrão (adapta a quantidade de stop bits à paridade: com paridade = 1 stop bit, sem paridade 2 stop bits.) One stop bit: um bit de parada Two stop bits: dois bits de parada Valor padrão: Standard
Number of Idle Chars	A quantidade de caracteres de Idle no início e fim de um envelope de telegrama RTU. Faixa de valores: 0...65535 Valor padrão: 5 caracteres

Tabela 188: Parâmetros master Modbus serial

### 7.2.7.2 Variáveis de sistema slave Modbus

No Editor Edit há três variáveis de status (variáveis de sistema) à disposição.

Elemento	Descrição
Modbus Slave Activation Control	Ativar ou desativar o slave Modbus mp programa de aplicação. 0: Ativar 1: Desativar (Disparado por flanco! Slave Modbus Slave pode ser ativado via PADT mesmo se <i>Modbus Slave Activation Control</i> = 1.)
Modbus Slave Error	Parâmetros como no slave TCP/UDP, veja Capítulo 7.2.5.2.
Modbus Slave State	Status de conexão do slave Modbus: 0: Desativado 1: Não conectado 2: Conectado

Tabela 189: Variáveis de sistema slave Modbus

### 7.2.7.3 Características slave Modbus

Para a configuração da conexão aos slaves seriais, devem ser ajustados os seguintes parâmetros no master Modbus.

Elemento	Descrição
Type	Modbus Slave
Name	Nome do slave Modbus, pode ser escolhido pelo usuário
Description	Descrição livre inequívoca para o slave Modbus
Slave Address	1...247
Demais parâmetros como no slave TCP/UDP, veja Capítulo 7.2.5.2.	

Tabela 190: Parâmetros de conexão master Modbus

## i

No slave Modbus serial, o Receive Timeout depende da velocidade de transmissão ajustada.

Se a taxa de Baud for 19200 [bit/s] ou maior, o valor pré-definido para o Receive Timeout pode ser usado. Com taxas de Baud mais baixas do que 19200 [bit/s], o Receive Timeout deve ser aumentado.

### 7.2.8 Control Panel (Modbus Master)

No Control Panel, o usuário pode verificar e controlar os ajustes do master Modbus. Além disso, informações de status atuais (p. ex., estado do master) do master são exibidas.

#### Assim abre-se o Control Panel para a supervisão do master Modbus:

1. Selecionar na árvore de estrutura **Hardware** e no menu de contexto **Online**.
2. Introduzir no **System Login** os dados de acesso para abrir a visualização online do hardware.
3. Clique duplo em **COM-Module** e selecionar na árvore de estrutura **Master**.

### 7.2.8.1 Menu de contexto (Modbus Master)

No menu de contexto do master Modbus selecionado, é possível escolher os seguintes comandos:

#### **Offline**

Com este comando, o master Modbus é parado.

#### **Operate**

Com este comando, o master Modbus é iniciado.

#### **Resetar valores de estatística**

Reseta os dados estatísticos (p. ex., quantidade de erros de barramento, tempo de ciclo mín, máx, etc.) a zero.

### 7.2.8.2 Campo de exibição (Modbus Master)

No campo de exibição são mostrados os seguintes valores do master Modbus selecionado.

Elemento	Descrição
Name	Nome do master Modbus
Master State	O estado do master Modbus mostra o estado atual do protocolo: OPERATE OFFLINE
Bus Error Count	Contador para a quantidade de erros de barramento
Disturbed Connections	Contador para a quantidade de conexões avariadas
µP Load (planned)	Veja Capítulo 7.2.3.2
µP Load (actual)	

Tabela 191: Campo de exibição Modbus Master

### 7.2.9 Control Panel (Modbus Master -> Slave)

No Control Panel, o usuário pode verificar e controlar os ajustes dos parceiros de comunicação do master Modbus e ativar/desativar. Além disso, informações de status atuais (p. ex., estado do slave) do parceiro de comunicação são exibidas.

#### **Assim abre-se o Control Panel para a supervisão da conexão Modbus:**

1. Selecionar na árvore de estrutura **Hardware** e no menu de contexto **Online**.
2. Introduzir no **System Login** os dados de acesso para abrir a visualização online do hardware.
3. Clique duplo em **COM-Module** e selecionar na árvore de estrutura **Modbus Master, Slave**.



### 7.2.10 Função do LED FBx no master Modbus

O estado do protocolo Modbus é sinalizado pelo LED FBx da interface correspondente do barramento de campo. Os estados do LED FBx são mostrados na seguinte tabela:

LED FBx	Cor	Descrição
DESLIGA	Amarelo	O protocolo master Modbus não está ativo! Ou seja, o sistema de comando está no estado STOP ou não há nenhum master Modbus configurado.
Piscando	Amarelo	O protocolo master Modbus está ativo e está em troca de dados com os slaves Modbus.

Tabela 192: LED FBx

### 7.2.11 Função do LED FAULT no master Modbus (só HIMax)

Uma avaria do protocolo Modbus é sinalizada pelo LED FAULT da interface correspondente do barramento de campo. Os estados do LED FAULT são mostrados na seguinte tabela:

LED FAULT	Cor	Descrição
DESLIGA	Vermelho	O protocolo master Modbus está livre de avarias.
Piscando	Vermelho	Os seguintes eventos causam uma avaria. <ul style="list-style-type: none"><li>▪ Resposta incorreta ou mensagem de erro recebida do slave</li><li>▪ Timeout para um ou mais slaves</li><li>▪ Tempo de processamento ultrapassado</li></ul> Se durante mais de 5 segundos não ocorrer nenhum evento de erro, a exibição muda para o estado "Protocol not disturbed".

Tabela 193: FAULT FBx

### 7.3 Slave HIMA Modbus

Pela interface serial (RS485) e via Ethernet (TCP/UDP), o slave Modbus HIMA pode atender vários master Modbus simultaneamente.

#### Equipamentos necessários e requisitos de sistema

Elemento	Descrição
Sistema de comando HIMA	HIMax com módulo COM HIMatrix a partir de CPU OS V7 e COM OS V12
Módulo CPU	As interfaces Ethernet do módulo processador não podem ser usadas para Modbus TCP.
Módulo COM	Ethernet 10/100BaseT Conexões D-Sub FB1 e FB2 Se for utilizado Modbus RTU, o módulo COM nas interfaces seriais de barramento de campo (FB1 e/ou FB2) deve estar equipado com um submódulo HIMA RS485 opcional. Atribuição de interfaces, veja Capítulo 3.6.
Ativação	Cada uma das duas funções Modbus slave deve ser ligada individualmente, veja Capítulo 3.4. Modbus Slave RTU (RS485) Modbus Slave TCP Para o slave Modbus redundante, 2 licenças são necessárias, uma para cada módulo X-COM.

Tabela 194: Equipamentos necessários e requisitos de sistema para slave Modbus HIMA

#### Slave Modbus (características)

Elemento	Descrição																									
Slave Modbus	Para cada módulo COM pode ser configurado um slave Modbus.																									
Redundância	Só para HIMax! No máximo 10 pares de módulos de comunicação slave Modbus podem ser operados num sistema HIMax. Enquanto um par de módulos de comunicação slave Modbus trabalhar de forma redundante, os mesmos dados de entrada e saída são trocados pelos dois módulos de comunicação com o master Modbus, veja Capítulo 7.3.3.																									
Quantidade de acessos master	RTU: Devido à técnica de transmissão RS485, apenas um master Modbus pode acessar o slave. TCP: No máximo 20 master Modbus podem acessar o slave. UDP: Uma quantidade illimitada de master Modbus podem acessar o slave.																									
Tamanho máx. dos dados de envio	Veja Tabela 9 Protocolos padrão																									
Tamanho máx. dos dados de recepção																										
Formato de representação dos dados Modbus	Os sistemas de comando HIMax/HIMatrix usam o formato Big Endian. Exemplo de dados 32 Bit (p. ex., DWORD, DINT): <table><tr><td>Dados 32 Bit (hex)</td><td colspan="4">0x12345678</td></tr><tr><td>Offset de memória</td><td>0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>Big Endian (HIMax/HIMatrix)</td><td>12</td><td>34</td><td>56</td><td>78</td></tr><tr><td>Middle Endian (H51q)</td><td>56</td><td>78</td><td>12</td><td>34</td></tr><tr><td>Little Endian</td><td>78</td><td>56</td><td>34</td><td>12</td></tr></table>	Dados 32 Bit (hex)	0x12345678				Offset de memória	0	1	2	3	Big Endian (HIMax/HIMatrix)	12	34	56	78	Middle Endian (H51q)	56	78	12	34	Little Endian	78	56	34	12
Dados 32 Bit (hex)	0x12345678																									
Offset de memória	0	1	2	3																						
Big Endian (HIMax/HIMatrix)	12	34	56	78																						
Middle Endian (H51q)	56	78	12	34																						
Little Endian	78	56	34	12																						

Tabela 195: Características slave Modbus

### 7.3.1 Configuração do slave Modbus TCP

**Assim cria-se um novo slave Modbus HIMA:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. No menu de contexto de protocolos, seleccionar **New, Modbus Slave Set** para adicionar um novo Modbus Slave Set.
3. No menu de contexto do Modbus Slave Set, seleccionar **Edit** e abrir **Modbus Slave Set Properties**, manter os valores padrão.
4. Seleccionar o registo **Modbus Slave** e efetuar os seguintes ajustes:
  - Seleccionar **COM-Modules**
  - Ativar Enable TCP
  - Os demais parâmetros mantêm os valores padrão.



Um exemplo de configuração para a conexão de um slave Modbus TCP HIMA com um master Modbus TCP HIMA está descrito no Capítulo 7.2.1.

---

### 7.3.2 Configuração do slave Modbus TCP redundante

**Assim cria-se um novo slave Modbus HIMA redundante:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, Modbus Slave Set**.
2. No menu de contexto do Modbus Slave Set, seleccionar **Edit** e abrir **Modbus Slave Set Properties**, e efetuar o seguinte ajuste:
  - Ativar Set Redundant Operation.
  - ☒ O registo **Modbus Slave Redundant** é acrescentado automaticamente
3. Seleccionar o registo **Modbus Slave Redundante** e efetuar os seguintes ajustes:
  - Seleccionar **COM-Modules**
  - Ativar Enable TCP

Os demais parâmetros mantêm os valores padrão.



As variáveis de envio e de recepção atribuídas no Modbus Slave Set são válidas para ambos os slaves Modbus.

---

### 7.3.3 Regras para o slave Modbus TCP redundante

Para a operação redundante de módulos de comunicação slave Modbus HIMax recomenda-se uma configuração de sistema redundante, veja Manual de sistema HI 801 242 P.

Caso contrário, já na ocorrência do primeiro erro no sistema HIMax não é possível garantir que os pares de módulos de comunicação slave Modbus se comportem de forma consistente em relação ao seu parceiro externo (master Modbus).

#### 7.3.3.1 Slots admissíveis dos módulos COM slave Modbus redundantes

Para minimizar possíveis colisões no barramento de sistema HIMax, os segmentos de barramento de sistema (1 a 3) no suporte básico devem ser respeitados. Por isso, os módulos de comunicação Modbus redundantes sempre só devem ser colocados no mesmo segmento de um suporte básico nos seguintes slots:

Segmento	Slot
1	3...6 (se não estão previstos módulos processadores aqui)
2	7...14
3	15...18

Tabela 196: Slots admissíveis dos módulos COM slave Modbus redundantes

#### 7.3.3.2 Módulos COM slave Modbus redundantes em diferentes suportes básicos

Não podem ser operados mais do que dois pares de módulos de comunicação slave Modbus redundantes cujos módulos de comunicação slave Modbus redundantes estão em suportes básicos diferentes (0 a 15).

Os módulos de comunicação slave Modbus redundantes mesmo neste caso apenas podem estar no suporte básico adjacente.

Adicionalmente, outros 8 pares de módulos de comunicação slave Modbus podem ser operados no mesmo sistema HIMax conforme as regras do Capítulo 7.3.3.1.

### NOTA



**Avarias operacionais são possíveis!**

**Usar slots para módulos de comunicação slave Modbus redundantes apenas de acordo com as regras mencionadas!**

**Entre um módulo X-COM e os módulos X-CPU, é admissível um tempo de funcionamento adicional (devido a comprimento de cabos, Switches) de no máximo 50 µs.**

**Recomendação: Operar módulos X-COM o mais próximo possível aos módulos X-CPU (p. ex., Rack 0, Rack 1)**

### 7.3.4 Funções de menu do Slave Set Modbus HIMA

A função de menu Edit do menu de contexto do Slave Set Modbus abre o diálogo *Modbus Slave Set Properties*. A janela de diálogo contém os seguintes registros:

#### 7.3.4.1 Características Slave Set Modbus

No registro *Modbus Slave Set Properties* são ajustados os seguintes parâmetro para o slave Modbus.

Elemento	Descrição
Name	Nome do Modbus Slave Set
Activate Max. $\mu$ P Budget	Ativado: Transferir o limite do $\mu$ P-Budget do campo <i>Max. <math>\mu</math>P-Budget in [%]</i> .  Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P-Budget in [%]	Carga máxima $\mu$ P do módulo COM que pode ser produzida ao processar o protocolo. Faixa de valores: 1...100% Valor padrão: 30%
Set Redundant Operation	Ativado: Operação de redundância Desativado: Operação Mono Valor padrão: Desativado
Max. Response Time [ms]	Período de tempo após recebimento de uma solicitação dentro do qual o slave Modbus ainda pode responder. <b>No caso de sistemas de comando HIMatrix, este valor deve ser colocado a 0.</b> Faixa de valores: 0 - ( $2^{31} - 1$ ) ms Valor padrão: 5000 ms (0 = sem limitação)
Area for reading function codes 1, 3, 100, 102	O parâmetro determina de qual área de dados os dados para o código de função 1, 3, 100, 102 devem ser lidos. Faixa de valores: <ul style="list-style-type: none"> <li>Import Area: Área de importação</li> <li>Export Area: Área de exportação (compatível a 51q)</li> </ul>
Area for Reading Function Code 23, 106	Aqui o usuário pode determinar a área do slave Modbus da qual o código de função 23 deve ser lido. Import Area – Área de importação: O master acessa a área de importação do slave para leitura e escrita. Export Area – Área de exportação: O master lê da área de exportação do slave e escreve para a área de importação do slave. Nota: A leitura e escrita ocorrem dentro de um ciclo de CPU. Ou seja, os dados lidos foram disponibilizados pelo <u>último</u> ciclo de CPU.
COM: Values at Connection Loss to Master	No caso da perda de conexão do módulo de comunicação ao master Modbus, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. Adopt Initial Data      As variáveis de entrada são resetadas para os valores iniciais. Retain Last Value      As variáveis de entrada mantêm o último valor.

Elemento	Descrição
CPU: Values at Connection Loss to COM	<p>No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador.</p> <p>(P. ex., se o módulo de comunicação é puxado para fora durante a comunicação).</p> <p>Se um projeto deve ser convertido de um SILworX menor V3, este valor deve ser colocado em <b>Retain Last Value</b>, para não alterar o CRC.</p> <p>Para sistemas de comando HIMatrix menores do que CPU OS V8, este valor sempre deve ser ajustado para <b>Retain Last Value</b>.</p> <p>The same behaviour as the COM to the Master – O mesmo comportamento como COM a master</p> <p>Retain Last Value</p> <p>Veja ajustes em parâmetro <i>Values at Connection Loss to Master</i></p> <p>As variáveis de entrada mantêm o último valor.</p> <p>Valor padrão: The same behaviour as the COM to the Master – O mesmo comportamento como COM a master</p>
Alternative register/bit addressing	<p>Ativado Usar endereçamento alternativo</p> <p>Desativado Não usar endereçamento alternativo</p> <p>Valor padrão: Desativado, veja Capítulo 7.3.11</p>
Register Area Offset Bits Input	<p>Faixa de valores: 0...65535</p> <p>Valor padrão: 0</p>
Register Area Offset Bits Output	<p>Faixa de valores: 0...65535</p> <p>Valor padrão: 0</p>
Bit Area Offset Register Input	<p>Faixa de valores: 0...65535</p> <p>Valor padrão: 0</p>
Bit Area Offset Register Output	<p>Faixa de valores: 0...65535</p> <p>Valor padrão: 0</p>
Process data refresh rate [ms]	<p>Tempo de atualização em milissegundos com o qual os dados do protocolo são trocados entre COM e CPU.</p> <p>Se o <i>Refresh Rate</i> for zero ou menor do que o tempo de ciclo do sistema de comando, a troca de dados ocorre o mais rápido possível.</p> <p>Faixa de valores: 0...(2<sup>31</sup> - 1)</p> <p>Valor padrão: 0</p>
Force Process Data Consistency	<p>Ativado: Transferência dos dados completos do protocolo de CPU para COM dentro de um ciclo de CPU.</p> <p>Desativado: Transferência dos dados completos do protocolo de CPU para COM, distribuídos sobre vários ciclos de CPU cada vez 1100 Byte por direção de dados.</p> <p>Com isso, eventualmente também é possível reduzir o tempo de ciclo do sistema de comando.</p> <p>Valor padrão: Ativado</p>

Tabela 197: Registro Características Slave Set Modbus

#### 7.3.4.2 Variável de registro (acesso de registro)

No registro variáveis de registro devem ser introduzidas as variáveis que o master endereça a cada 16 Bit (código de função 3, 4, 6, 16, 23, 102, 103, 105, 106).

### 7.3.4.3 Variáveis de bit

(acesso de Bit ou Coil)

Na aba variáveis de bit devem ser introduzidas as variáveis que o master endereça por 1 bit (código de função 1, 2, 5, 15, 100, 101, 104).

### 7.3.5 Atribuir variáveis de envio/recepção

No registro **Inputs** são atribuídas todas as variáveis que o slave Modbus recebe do master Modbus.

#### Assim configuram-se as variáveis de envio do slave Modbus:

1. Selecionar na árvore de estrutura o **Modbus Slave** que gostaria de configurar.
2. Clicar com o botão direito em **Modbus Slave** e selecionar **Edit**.
3. Selecionar o registro **Register Variables** ou **Bit Variables**.
4. Selecionar no *Object Panel* uma **Variable** e puxar a mesma mediante Drag&Drop para a área *Register Outputs*.
5. Repetir este passo para cada outra variável que gostaria de definir como variável de envio para o slave Modbus.
6. Clicar com o botão direito na área *Register Outputs* e selecionar **New Offsets**.

#### Assim configuram-se as variáveis de recepção do slave Modbus:

1. Selecionar na árvore de estrutura o **Modbus Slave** que gostaria de configurar.
2. Clicar com o botão direito do mouse em **Modbus Slave** e selecionar **Edit**.
3. Selecionar o registro **Register Variables** ou **Bit Variables**.
4. Selecionar no *Object Panel* uma **Variable** e puxar a mesma mediante Drag&Drop para a área *Register Inputs*.
5. Repetir este passo para cada outra variável que gostaria de definir como variável de recepção para o slave Modbus.
6. Clicar com o botão direito na área *Register Inputs* e selecionar **New Offsets**.

### 7.3.6 Variáveis de sistema Slave Set Modbus

O registro **System Variables Modbus Slave Set** disponibiliza a seguinte variável de sistema.

Elemento	Descrição
Redundant State	<p>Este parâmetro descreve o estado de redundância do par de módulos de comunicação slave Modbus redundante.</p> <p>0: Módulos COM slave Modbus redundantes ativos</p> <p>1: Primeiro módulo COM slave Modbus não ativo</p> <p>2: Módulo COM slave Modbus redundante não ativo</p> <p>3: Ambos os módulos COM slave Modbus não ativos</p>

Tabela 198: Registro Slave Set Modbus

### 7.3.7 Modbus Slave e Modbus Slave Redundant

No registro **Modbus Slave** há os dois registros Características e Variáveis de sistema.



A pinagem das conexões D-Sub (FB1, FB2) é descrita nas folhas de dados dos respectivos módulos HIMax.

## Características

Elemento	Descrição
Module	Seleção do módulo COM no qual este protocolo é processado.
Master Monitoring Time [ms]	Período de tempo após recebimento de uma solicitação dentro do qual o slave Modbus deve reagir. No caso da perda de conexão do módulo de comunicação ao master Modbus, em dependência do parâmetro <i>X-COM: Values at Connection Loss to Master</i> as variáveis de entrada são inicializadas ou repassadas ao módulo processador sem alterações. Veja Capítulo 7.3.4.1. Faixa de valores $1 \dots (2^{31} - 1)$ [ms] Valor padrão: 0 ms (sem limitação)
Parâmetros da interface Ethernet	
Enable the TCP/IP connection	Ativado      Conexão TCP/IP ativada Desativado      Conexão TCP/IP desativada Valor padrão: Desativado
TCP Port	Valor padrão: 502
Maximum Number of TCP Connections	Quantidade máxima de conexões TCP como servidor simultaneamente abertas. Faixa de valores: 1...20 Valor padrão: 3
UDP Enable	Ativado      Conexão UDP/IP ativada Desativado      Conexão UDP/IP desativada Valor padrão: Desativado
UDP Port	Valor padrão: 502
Parâmetros da interface serial	
Name	Nome da interface serial
Interface	Seleção das interfaces de barramento de campo disponíveis que podem ser usadas para o slave Modbus (nenhuma, fb1, fb2).
Slave Address	Endereço de barramento do slave Faixa de valores: 1...247
Baud rate [bps]	Velocidade de transmissão para RS485 valores possíveis: 300 bit/s 600 bit/s 1200 bit/s 2400 bit/s 4800 bit/s 9600 bit/s 19200 bit/s 38400 bit/s 57600 bit/s (baud rate máximo HIMax a partir de V4) 62500 bit/s (só HIMatrix) 76800 bit/s (só HIMatrix) 115000 bit/s (só HIMatrix)
Parity	Faixa de valores: ▪ None: nenhuma ▪ Odd: impar ▪ Even: par Valor padrão: Even
Stop Bits	Faixa de valores: Standard (adapta a quantidade de stop bits à paridade: com paridade = 1 stop bit, sem paridade 2 stop bits.) One stop bit: um bit de parada Two stop bits: dois bits de parada Valor padrão: Standard



Elemento	Descrição
Number of Idle Chars	A quantidade de caracteres de Idle no início e fim de um envelope de telegrama RTU. Faixa de valores: 1...65535 Valor padrão: 5 caracteres

Tabela 199: Registro Portas TCP e UDP para slave Modbus HIMA

O registro **System Variables** disponibiliza variáveis de sistema que permitem avaliar o estado do slave Modbus no programa de aplicação e controlar o slave Modbus.

Elemento	Descrição
Average Buffer Fill Level for Requests	Quantidade média de requisições simultâneas do master
Valid Master Requests	Quantidade das requisições válidas do master desde o último reset de todos os contadores ou desde ligar.
Master Requests	A quantidade total de requisições do master desde o último reset de todos os contadores ou desde ligar.
Master Monitoring Time [ms]	Período de tempo após recebimento de uma requisição dentro do qual o slave Modbus deve reagir, veja Capítulo 7.3.7.
Master Connection State	FALSE: Não conectado TRUE: Conectado
Maximum Buffer Fill Level for Requests	Quantidade máxima de requisições simultâneas do master
Reset All Counters	Com esta variável de sistema, é possível resetar todos os contadores pelo programa de aplicação. Uma mudança de 0 para 1 aciona a função Reset Valores > 1 são tratados como 1.
Invalid Master Requests	Quantidade das requisições inválidas do master desde o último reset de todos os contadores ou desde ligar. Requisições inválidas são as que o slave Modbus responde com um código de erro para o master Modbus. Transmissões com erro que já são detectadas e filtradas no nível do controlador (erros de Framing, erros de CRC, erros no comprimento) não são contados aqui, mas são comunicados pelo diagnóstico.
Rejected Requests	Quantidade das requisições rejeitadas do master desde o último reset de todos os contadores ou desde ligar.
Response Timeout	Quantidade de ocorrências de tempo ultrapassado com <i>response timeouts</i> no caso de respostas desde o último reset ou ligamento. O tempo ultrapassado ao responder é o tempo máximo que pode passar até a confirmação de recepção de uma mensagem no remetente.

Tabela 200: Registro Variáveis de sistema slave Modbus HIMA

### 7.3.8 Códigos de função Modbus do slave Modbus HIMA

#### 7.3.8.1 Códigos de função Modbus

Os seguintes códigos de função Modbus são suportados pelo slave Modbus HIMA.

Elemento	Código	Tipo	Significado
READ COILS	01	BOOL	Ler várias variáveis (BOOL) da área de importação ou exportação <sup>1)</sup> do slave. Comprimento máximo dos dados de processo: 251 Bytes
READ DISCRETE INPUT	02	BOOL	Ler várias variáveis (BOOL) da área de exportação do slave. Comprimento máximo dos dados de processo: 251 Bytes
READ HOLDING REGISTER	03	WORD	Ler várias variáveis de tipo livre da área de importação ou exportação <sup>1)</sup> do slave. Comprimento máximo dos dados de processo: 250 Bytes
READ INPUT REGISTER	04	WORD	Ler várias variáveis de tipo livre da área de exportação do slave. Comprimento máximo dos dados de processo: 250 Bytes
WRITE SINGLE COIL	05	BOOL	Escrever um sinal individual (BOOL) para a área de importação do slave. Comprimento máximo dos dados de processo: 1 Byte
WRITE SINGLE REGISTER	06	WORD	Escrever um sinal individual (WORD) para a área de importação do slave. Comprimento máximo dos dados de processo: 2 Bytes
Diagnostics	08	x	Só Subcode 0: função Loopback do slave
WRITE MULTIPLE COILS	15	BOOL	Escrever várias variáveis (BOOL) para a área de importação do slave. Comprimento máximo dos dados de processo: 247 Bytes
WRITE MULTIPLE REGISTER	16	WORD	Escrever várias variáveis de tipo livre para a área de importação do slave. Comprimento máximo dos dados de processo: 246 Bytes
READ WRITE MULTIPLE REGISTER	23	WORD	Escrever e ler várias variáveis de tipo livre da e para a área de importação ou exportação do slave. Comprimento máximo dos dados de processo: 242 bytes (telegrama de requisição do master Modbus) 250 bytes (resposta ao master).
Read Device Identification	43	x	Fornece os dados de identificação do slave para o master.
<sup>1)</sup> A área de exportação só pode ser selecionada nos slaves HIMA			

Tabela 201: Códigos de função Modbus do slave Modbus HIMA

Os códigos de função 03, 04, 16 e 23 suportam além do tipo de dados WORD (2 Bytes) também inúmeros outros tipos de dados.

Para cada Request, deve ser introduzido o endereço inicial da primeira variável a ser transmitida e a quantidade de registros/Bits das variáveis a serem transmitidas.

Códigos de erro:

- Se o master enviar um telegrama com código de função desconhecido, o sistema de comando responde com o código de erro 1 (Invalid Code).
- Se o telegrama do master não corresponder à configuração do slave Modbus (ou seja, p. ex., o telegrama de requisição não termina de forma clara num limite de variáveis), o slave responde com código de erro 2 (Invalid Data).
- Se o master enviar um telegrama com valores incorretos (p. ex., campo de comprimento), o slave responde com código de erro 3 (Invalid Value).

A comunicação apenas ocorre no estado RUN do módulo COM. Requisições do master em todos os outros estados operacionais do módulo COM não são respondidas.

#### **Nota sobre a função Modbus: Read Device Identification (43)**

O slave Modbus HIMax fornece os dados de identificação ao master e suporta os seguintes Object-Ids:

Basic:

0x00 VendorName "HIMA Paul Hildebrandt GmbH"  
0x01 ProductCode "<Module Serial Number>"  
0x02 MajorMinorRevision "<COM Vx.y CRC>"

Regular:

0x03 VendorUrl "http://www.hima.com"  
0x04 ProductName "HIMax"  
0x05 ModelName "HIMax"  
0x06 UserApplicationName "-----[S.R.S]"

Extended:

0x80 vazio "-----"  
0x81 vazio "-----"  
0x82 vazio "-----"  
0x83 vazio "-----"  
0x84 vazio "-----"  
0x85 vazio "-----"  
0x86 CRC do arquivo modbus.config "<0x234adcef>"  
(arquivo de configuração do protocolo de slave Modbus no sistema de arquivos da CPU.  
Conferir com as indicações no SILworX em Online/Comparação de versões).

São suportados os seguintes ReadDevice ID Codes:

- (1) Read Basic device identification (stream access)
- (2) Read regular device identification (stream access)
- (3) Read extended device identification (stream access)
- (4) Read one specific identification object (individual access)

(Informações mais detalhadas sobre o Modbus encontram-se na especificação "Modbus Application Protocol Specification" [www.modbus.org](http://www.modbus.org))

### 7.3.9 Códigos de função específicos HIMA

Os códigos de função específicos HIMA correspondem aos códigos de função Modbus padrão. As únicas diferenças são o comprimento dos dados de processo máximo admissível de 1100 bytes e o formato do Request e Response-Header:

Elemento	Código	Tipo	Significado
Read Coils Extended	100 (0x64)	BOOL	Corresponde ao Functioncode 01 Ler várias variáveis (BOOL) da área de importação ou exportação <sup>1)</sup> do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read Discrete Inputs Extended	101 (0x65)	BOOL	Corresponde ao Functioncode 02 Ler várias variáveis (BOOL) da área de exportação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read Holding Registers Extended	102 (0x66)	WORD	Corresponde ao Functioncode 03 Ler várias variáveis de tipo livre da área de importação ou exportação <sup>1)</sup> do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read Input Registers Extended	103 (0x67)	WORD	Corresponde ao Functioncode 04 Ler várias variáveis de tipo livre da área de exportação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Write Multiple Coils Extended	104 (0x68)	BOOL	Corresponde ao Functioncode 15 Escrever várias variáveis (BOOL) para a área de importação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Write Multiple Registers Extended	105 (0x69)	WORD	Corresponde ao Functioncode 16 Escrever várias variáveis de tipo livre para a área de importação do slave. Comprimento máximo dos dados de processo: 1100 Bytes
Read/Write Multiple Registers Extended	106 (0x6A)	WORD	Corresponde ao Functioncode 23 Escrever e ler várias variáveis de tipo livre da e para a área de importação ou exportação do slave. Comprimento máximo dos dados de processo: 1100 bytes (telegrama de requisição do master Modbus) 1100 bytes (resposta ao master).

## 7.3.9.1 Formato dos cabeçalhos para Request e Response

Os cabeçalhos de Request e Response dos códigos de função específicos HIMA possuem a seguinte estrutura:

Código	Request	Response
100 (0x64)	1 byte código de função 0x64 2 bytes endereço inicial 2 bytes quantidade de Coils 1...8800(0x2260)	1 byte código de função 0x64 2 bytes quantidade de Bytes = N N bytes dados Coil (8 Coils são empacotados em um Byte)
101 (0x65)	1 byte código de função 0x65 2 bytes endereço inicial 2 bytes quantidade de Coils 1...8800(0x226)	1 byte código de função 0x65 2 bytes quantidade de Bytes = N N bytes dados Coil (8 Coils são empacotados em um Byte)
102 (0x66)	1 byte código de função 0x66 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226)	1 byte código de função 0x66 2 bytes quantidade de Bytes = N N bytes dados de registro
103 (0x67)	1 bytes código de função 0x67 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226)	1 byte código de função 0x67 2 bytes quantidade de Bytes = N N bytes dados de registro
104 (0x68)	1 byte código de função 0x68 2 bytes endereço inicial 2 bytes quantidade de Coils 1...8800(0x2260) 2 bytes quantidade de Bytes = N N bytes dados Coil	1 byte código de função 0x66 2 bytes endereço inicial 2 bytes quantidade de Coils 1...8800(0x2260)
105 (0x69)	1 byte código de função 0x69 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226) 2 bytes quantidade de Bytes = N N bytes dados de registro	1 byte código de função 0x69 2 bytes endereço inicial 2 bytes quantidade de registros 1...550(0x226)
106 (0x6A)	1 byte código de função 0x6a 2 bytes endereço inicial leitura 2 bytes quantidade de registros de leitura 1...550(0x226) 2 bytes endereço inicial escrita 2 bytes quantidade de registros de escrita 1...550(0x226) 2 bytes quantidade de bytes para escrever = N N bytes dados de registro	1 byte código de função 0x6a 2 bytes quantidade de Bytes = N N bytes dados de registro

### 7.3.10 Endereçamento Modbus via Bit e Register

Este modo de endereçamento corresponde ao endereçamento padrão Modbus e só conhece os dois comprimentos de dados Bit (1 Bit) e Register (16 Bit) com os quais todos os tipos de dados admissíveis podem ser transmitidos.

No slave Modbus existe uma **Register Area** (entradas e saídas) e uma **Bit Area** (entradas e saídas). Ambas as áreas são separadas entre si e podem receber todos os tipos de dados admissíveis. A diferença entre estas áreas está nos códigos de função Modbus permitidos com os quais é possível acessar estas áreas.

**i**

O endereçamento Modbus via Bit e Register não garante a integridade de variáveis, ou seja, com este acesso podem ser lidas/escritas partes não limitadas de variáveis. Variáveis do tipo BOOL são depositadas em forma de pacote, ou seja, cada variável do tipo BOOL é depositada como Bit dentro de um Byte.

#### 7.3.10.1 Área Register

Na aba **Register Variables** são criadas as variáveis da área Register. Informações mais detalhadas sobre a atribuição das variáveis de envio/recepção, veja Capítulo 7.3.5.

**i**

Para acessar as variáveis na área Register mediante os códigos de função Modbus 1, 2, 5, 15, as variáveis devem ser espelhadas para a área Bit, veja Capítulo 7.3.11.1.

Apenas é possível acessar as variáveis na área Register mediante os códigos de função Modbus 3, 4, 6, 16, 23. Para isso, deve ser introduzido o endereço inicial da primeira variável nas características do código de função.

Exemplo: Acesso às variáveis na área Register do slave Modbus

Variáveis do tipo Register	Register.Bit	Bit
00_Register_Area_WORD	0.0	0
01_Register_Area_SINT	1.8	16
02_Register_Area_SINT	1.0	24
03_Register_Area_REAL	2.0	32
04_Register_Area_BOOL	4.8	64
05_Register_Area_BOOL	4.9	65
06_Register_Area_BOOL	4.10	66
07_Register_Area_BOOL	4.11	67
08_Register_Area_BOOL	4.12	68
09_Register_Area_BOOL	4.13	69
10_Register_Area_BOOL	4.14	70
11_Register_Area_BOOL	4.15	71
12_Register_Area_BOOL	4.0	72
13_Register_Area_BOOL	4.1	73
14_Register_Area_BOOL	4.2	74
15_Register_Area_BOOL	4.3	75
16_Register_Area_BOOL	4.4	76
17_Register_Area_BOOL	4.5	77
18_Register_Area_BOOL	4.6	78
19_Register_Area_BOOL	4.7	79

Tabela 202: Variáveis do tipo Register na área Register do slave Modbus

Configuração master Modbus HIMA do telegrama de requisição

**Assim, lêem-se no master Modbus as variáveis 01\_Register\_Area\_SINT até 03\_Register\_Area\_REAL:**

1. Clicar com o botão direito em **TCP/UDP-Slave** e selecionar no menu de contexto **New**.
2. Selecionar da lista **Read Holding Registers (3)**.
3. Clicar com o botão direito em **Read Holding Registers (3)** e selecionar **Properties**.
  - Introduzir Start address of the read area, **1**.
4. Clicar com o botão direito em **Read Holding Registers (3)** e selecionar **Edit**.
5. Puxar do **Object Panel** as seguintes variáveis mediante Drag&Drop para o registro **Input Variables**.

Variáveis do tipo Register	Offset
01_Register_Area_SINT	0
02_Register_Area_SINT	1
03_Register_Area_REAL	2

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Variables** e selecionar **New Offsets** para renumerar os offsets das variáveis.

### 7.3.10.2 Área Bit

Na aba **Bit Variables** são criadas as variáveis da área Bit. Informações mais detalhadas sobre a atribuição das variáveis de envio/recepção, veja Capítulo 7.3.5.

**i**

Para acessar as variáveis na área Bit mediante os códigos de função Modbus 3, 4, 6, 16, 23, as variáveis devem ser espelhadas para a área Register, veja Capítulo 7.3.11.2.

Apenas é possível acessar as variáveis na área Bit mediante os códigos de função Modbus 1, 2, 5, 15. Para isso, deve ser introduzido o endereço inicial da primeira variável nas características do código de função.

Exemplo: Acesso às variáveis na área Bit do slave Modbus

Variáveis de bit	Bit	Register.Bit
00_BIT_Area_WORD	0	0.0
01_BIT_Area_SINT	16	1.8
02_BIT_Area_SINT	24	1.0
03_BIT_Area_REAL	32	2.0
04_BIT_Area_BOOL	64	4.8
05_BIT_Area_BOOL	65	4.9
06_BIT_Area_BOOL	66	4.10
07_BIT_Area_BOOL	67	4.11
08_BIT_Area_BOOL	68	4.12
09_BIT_Area_BOOL	69	4.13
10_BIT_Area_BOOL	70	4.14
11_BIT_Area_BOOL	71	4.15
12_BIT_Area_BOOL	72	4.0
13_BIT_Area_BOOL	73	4.1
14_BIT_Area_BOOL	74	4.2
15_BIT_Area_BOOL	75	4.3
16_BIT_Area_BOOL	76	4.4
17_BIT_Area_BOOL	77	4.5
18_BIT_Area_BOOL	78	4.6
19_BIT_Area_BOOL	79	4.7

Tabela 203: Variáveis Bit na área Bit do slave Modbus

### Configuração master Modbus HIMA do telegrama de requisição

Assim lêem-se as variáveis 04\_BIT\_Area\_BOOL até 06\_Area\_BOOL no master Modbus:

1. Clicar com o botão direito em **TCP/UDP-Slave** e selecionar no menu de contexto **New**.
2. Selecionar da lista **Read Coils (1)**.
3. Clicar com o botão direito em **Read Coils (1)** e selecionar no menu de contexto **Properties**.
  - Introduzir Start address of the read area, **64**.
4. Clicar com o botão direito em **Read Coils (1)** e selecionar no menu de contexto **Edit**.
5. Puxar do **Object Panel** as seguintes variáveis mediante Drag&Drop para o registro **Input Variables**.

Variáveis de bit	Offset
04_BIT_Area_BOOL	0
05_BIT_Area_BOOL	1
06_BIT_Area_BOOL	2

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Variables** e selecionar **New Offsets** para renumerar os offsets das variáveis.

#### 7.3.11 Offsets para endereçamento Modbus alternativo

Para acessar as variáveis na **Bit Area** com os códigos de função Modbus (tipo Register) e as variáveis na **Register Area** com os códigos de função Modbus (tipo Bit), as variáveis devem ser espelhadas para a respectiva outra área.

Na aba **Properties/Offsets** são introduzidos os Offsets das variáveis espelhadas.

**Assim espelham-se as variáveis para a área Bit e Register:**

1. Selecionar no menu de contexto do slave Modbus **Edit, Offsets** e ativar **Use Alternative Register/Bit Addressing**.
  - ☒ Assim, as variáveis são espelhadas para a respectiva outra área.
2. Introduzir o offset para as variáveis espelhadas na área Bit e Register.

**i**

As variáveis espelhadas na área Bit/Register e as variáveis existentes na área Bit/Register não podem se sobrepor em relação aos endereços Modbus.

Elemento	Descrição/faixa de valores
Alternative register/bit addressing	Ativado Usar endereçamento alternativo Desativado Não usar endereçamento alternativo Valor padrão: Desativado
Register area offset/bit inputs	0...65535
Register area offset/bit outputs	0...65535
Bit area offset/register inputs	0...65535
Bit area offset/register outputs	0...65535

Tabela 204: Aba Offsets para slave Modbus HIMA



### 7.3.11.1 Acesso às variáveis tipo Register na área Bit do slave Modbus

Para acessar as variáveis Register com os códigos de função Modbus 1, 2, 5, 15 (tipo Bit), as variáveis Register devem ser espelhadas para a **Bit Area**.

Na aba **Properties/Offsets** devem ser introduzidos os Offsets das variáveis Register espelhadas.

Exemplo:

Área Bit Offset/Register Entradas 8000

Área Bit Offset/Register Saídas 8000

Aqui estão as variáveis espelhadas da área Register para a área Bit, a partir do endereço Bit 8000.

Variáveis Register espelhadas	Bit
00_Register_Area_WORD	8000
01_Register_Area_SINT	8016
02_Register_Area_SINT	8024
03_Register_Area_REAL	8032
04_Register_Area_BOOL	8064
05_Register_Area_BOOL	8065
06_Register_Area_BOOL	8066
07_Register_Area_BOOL	8067
08_Register_Area_BOOL	8068
09_Register_Area_BOOL	8069
10_Register_Area_BOOL	8070
11_Register_Area_BOOL	8071
12_Register_Area_BOOL	8072
13_Register_Area_BOOL	8073
14_Register_Area_BOOL	8074
15_Register_Area_BOOL	8075
16_Register_Area_BOOL	8076
17_Register_Area_BOOL	8077
18_Register_Area_BOOL	8078
19_Register_Area_BOOL	8079

Tabela 205: Variáveis espelhadas da área Register para a área Bit

#### Configuração master Modbus HIMA do telegrama de requisição

**Assim lêem-se as variáveis 04\_Register\_Area\_BOOL até 06\_Register\_Area\_BOOL no master Modbus:**

1. Clicar com o botão direito em **TCP/UDP-Slave** e selecionar no menu de contexto **New**.
2. Selecionar da lista **Read Coils (1)**.
3. Clicar com o botão direito em **Read Coils (1)** e selecionar no menu de contexto **Properties**.
  - Introduzir Start address of the read area, **8064**.
4. Clicar com o botão direito em **Read Coils (1)** e selecionar no menu de contexto **Edit**.
5. Puxar do **Object Panel** as seguintes variáveis mediante Drag&Drop para o registro **Input Variables**.

Variáveis Register espelhadas	Offset
04_Register_Area_BOOL	0
05_Register_Area_BOOL	1
06_Register_Area_BOOL	2

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Variables** e selecionar **New Offsets** para renumerar os offsets das variáveis.

### 7.3.11.2 Acesso às variáveis Bit na área Register do slave Modbus

Para acessar as variáveis Bit com os códigos de função Modbus 3, 4, 6, 16, 23 (tipo Register), as variáveis Register devem ser espelhadas para a **Register Area**. Na aba **Properties/Offsets** devem ser introduzidos os Offsets das variáveis Bit espelhadas.

Exemplo:

Área Register Offset/Bit Entradas: 1000

Área Register Offset/Bit Saídas: 1000

Aqui estão as variáveis espelhadas da área Bit para a área Register, a partir do endereço Bit 1000.

Variáveis Bit espelhadas	Register.Bit
00_BIT_Area_WORD	1000.0
01_BIT_Area_SINT	1001.8
02_BIT_Area_SINT	1001.0
03_BIT_Area_REAL	1002.0
04_BIT_Area_BOOL	1004.8
05_BIT_Area_BOOL	1004.9
06_BIT_Area_BOOL	1004.10
07_BIT_Area_BOOL	1004.11
08_BIT_Area_BOOL	1004.12
09_BIT_Area_BOOL	1004.13
10_BIT_Area_BOOL	1004.14
11_BIT_Area_BOOL	1004.15
12_BIT_Area_BOOL	1004.0
13_BIT_Area_BOOL	1004.1
14_BIT_Area_BOOL	1004.2
15_BIT_Area_BOOL	1004.3
16_BIT_Area_BOOL	1004.4
17_BIT_Area_BOOL	1004.5
18_BIT_Area_BOOL	1004.6
19_BIT_Area_BOOL	1004.7

Tabela 206: Variáveis espelhadas da área Bit para a área Register

Configuração master Modbus HIMA do telegrama de requisição

**Assim, lêem-se no master Modbus as variáveis 01\_BIT\_Area\_SINT até 03\_BIT\_Area\_REAL:**

1. Clicar com o botão direito em **TCP/UDP-Slave** e selecionar no menu de contexto **New**.
2. Selecionar da lista Read Holding Registers (3).
3. Clicar com o botão direito em **Read Holding Registers (3)** e selecionar **Properties**.
  - Introduzir Start address of the read area, 1001.
4. Clicar com o botão direito em **Read Holding Registers (3)** e selecionar **Edit**.
5. Puxar do **Object Panel** as seguintes variáveis mediante Drag&Drop para o registro **Input Variables**.

Variáveis Bit espelhadas	Offset
01_BIT_Area_SINT	0
02_BIT_Area_SINT	1
03_BIT_Area_REAL	2

6. Abrir o menu de contexto com clique direito num lugar vazio na área **Output Variables** e selecionar **New Offsets** para renumerar os offsets das variáveis.

### 7.3.12 Control Panel (Modbus Slave)

No Control Panel, o usuário pode verificar e controlar os ajustes do slave Modbus. Além disso, informações de status atuais (p. ex., estado do master) do slave são exibidas.

**Assim abre-se o Control Panel para a supervisão do slave Modbus:**

1. Selecionar na árvore de estrutura **Hardware** e no menu de contexto **Online**.
2. Introduzir no **System Login** os dados de acesso para abrir a visualização online do hardware.
3. Clique duplo em **COM-Module** e selecionar na árvore de estrutura **Modbus Slave**.

#### 7.3.12.1 Menu de contexto (Modbus Slave)

No menu de contexto do slave Modbus selecionado, é possível escolher o seguinte comando:

**Resetar valores de estatística**

Reseta os dados estatísticos (tempo de ciclo mín, máx, etc.) a zero.

## 7.3.12.2 Campo de exibição (Modbus Slave)

No campo de exibição são mostrados os seguintes valores do slave Modbus selecionado.

Elemento	Descrição
Name	Nome do slave Modbus
Planned $\mu$ P Budget [%]	Veja Capítulo 7.3.4
Actual $\mu$ P Budget [%]	
SRS of Redundant Module	SRS do módulo COM redundante
Response Time [ms]	Período de tempo após recebimento de uma requisição dentro do qual o slave Modbus responde.

Tabela 207: Campo de exibição Slave Modbus

## 7.3.12.3 Campo de exibição (dados master)

No campo de exibição dados master são mostrados os seguintes valores.

Elemento	Descrição
Name	Nome dos dados master
Requests	A quantidade total de requisições do master desde o último reset do contador.
Valid Requests	A quantidade total de requisições válidas do master desde o último reset do contador.
Invalid Requests	A quantidade total de requisições inválidas do master desde o último reset do contador. Apenas contam como requisições inválidas as requisições que foram confirmadas pelo master. Requisições incorretas recebidas com erro de CRC são automaticamente descartadas.
Master-Timeout [ms]	Tempo de timeout dentro do qual o slave deve ter recebido no mínimo uma Request do seu master. Se o slave não receber nenhuma Request dentro do tempo do timeout, o <i>Master Connection Status</i> é colocado em <i>not connected</i> .
Connection State	0=Não monitorado (Master Request Timeout é zero) 1=Não conectado 2=Conectado
Response Timeout	Quantidade de <i>response timeouts</i> desde o último reset de todos os contadores ou desde ligar. O <i>response timeout</i> é o tempo máximo que pode passar até a confirmação de recepção de uma mensagem no remetente.
Rejected Requests	Quantidade das requisições rejeitadas do master desde o último reset de todos os contadores ou desde ligar.
Maximum Buffer Fill Level for Requests	Quantidade máxima de requisições simultâneas do master
Average Buffer Fill Level for Requests	Quantidade média de requisições simultâneas do master

Tabela 208: Campo de exibição dados master

### 7.3.13 Função do LED FBx no slave Modbus

O estado do protocolo Modbus é sinalizado pelo LED FBx da interface correspondente do barramento de campo. Os estados do LED FBx são mostrados na seguinte tabela:

LED FBx	Cor	Descrição
DESLIGA	Amarelo	O protocolo slave Modbus não está ativo! Ou seja, o sistema de comando está no estado STOP ou não há nenhum master Modbus configurado.
Piscando	Amarelo	O protocolo slave Modbus está ativo e está em troca de dados com o master Modbus.

Tabela 209: LED FBx

### 7.3.14 Função do LED FAULT no slave Modbus (só HIMax)

Uma avaria do protocolo Modbus é sinalizada pelo LED FAULT da interface correspondente do barramento de campo. Os estados do LED FAULT são mostrados na seguinte tabela:

LED FAULT	Cor	Descrição
DESLIGA	Vermelho	O protocolo slave Modbus está livre de avarias.
Piscando	Vermelho	O protocolo slave Modbus está com avarias. Os seguintes eventos causam uma avaria. <ul style="list-style-type: none"> <li>▪ Código de função desconhecido recebido</li> <li>▪ Requisição com endereçamento incorreto recebida</li> <li>▪ Tempo de processamento ultrapassado</li> </ul> Se durante mais de 5 segundos não ocorrer nenhum evento de erro, a exibição muda para o estado <i>Protocol not disturbed</i> .

Tabela 210: FAULT FBx

### 7.3.15 Códigos de erro da conexão Modbus TCP/IP

Os códigos de erro da conexão Modbus TCP/IP são exibidos na janela de diálogo **Diagnóstico**.

Error code	Descrição
35	Operação bloqueada
48	Porta já está em uso
50	Rede não opera
53	Conexão interrompida pelo software
54	Conexão interrompida pelo parceiro
55	Não há memória de buffer disponível
60	Ocorreu um Timeout, conexão encerrada
61	Conexão recusada (pelo parceiro)
65	Não há entrada de roteamento ao parceiro

Tabela 211: Códigos de erro Modbus TCP/IP

## 8 Send & Receive TCP

S&R TCP é um protocolo padrão independente do fabricante para a troca de dados cíclica e acíclica e usa nenhum protocolo especial exceto o TCP/IP.

Com o protocolo S&R TCP, o sistema de comando HIMax/HIMatrix suporta praticamente qualquer sistema de outros fabricantes, bem como PCs com interface Socket instalada (p. ex., Winsock.dll) para TCP/IP.

S&R TCP é compatível com a interface Siemens SEND/RECEIVE e permite a comunicação com os sistemas de comando Siemens via TCP/IP. A troca de dados ocorre pelos blocos funcionais S7 AG\_SEND (FC5) e AG\_RECV (FC6).

### 8.1 Requisitos de sistema

**Equipamentos necessários e requisitos de sistema:**

Elemento	Descrição
Sistema de comando	HIMax com módulo COM HIMatrix a partir de CPU OS V7 e COM OS V12
Módulo CPU	As interfaces Ethernet do módulo processador não podem ser usadas para Send & Receive TCP.
Módulo COM	Ethernet 10/100BaseT Para cada módulo COM pode ser configurado um protocolo S&R TCP.
Ativação	A liberação ocorre mediante código de liberação do software, veja Capítulo 3.4.

Tabela 212: Equipamentos necessários e requisitos de sistema S&R TCP

**Propriedades do protocolo S&R TCP:**

Elemento	Descrição
Direcionado à segurança	Não
Troca de dados	Troca de dados cíclica e acíclica via TCP/IP.
Blocos funcionais	Os blocos funcionais S&R TCP devem ser usados na troca de dados acíclica.
Conexões TCP	Até 32 conexões TCP podem ser configuradas num sistema de comando se o tamanho máximo dos dados de envio ou dados de recepção não for ultrapassado.
Tamanho máx. dos dados de envio	<p>Veja Tabela 9 Protocolos padrão</p> <p><b>i</b> Para determinar a quantidade máxima dos dados de trabalho, todas as variáveis de status das conexões TCP usadas e dos blocos funcionais TCP/SR devem ser subtraídas da quantidade máxima de dados de envio. A divisão para as conexões TCP individuais é livre.</p>
Tamanho máx. dos dados de recepção	

Tabela 213: Características S&R TCP

#### 8.1.1 Criar um protocolo S&R TCP

**Assim cria-se um novo protocolo S&R TCP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. No menu de contexto de protocolos, selecionar **New, Send/Receive over TCP** para adicionar um novo protocolo S&R TCP.
3. No menu de contexto do protocolo Send/Receive-over-TCP **Properties, General** selecionar o **COM-Module**.

## 8.2 Exemplo: Configuração S&R TCP

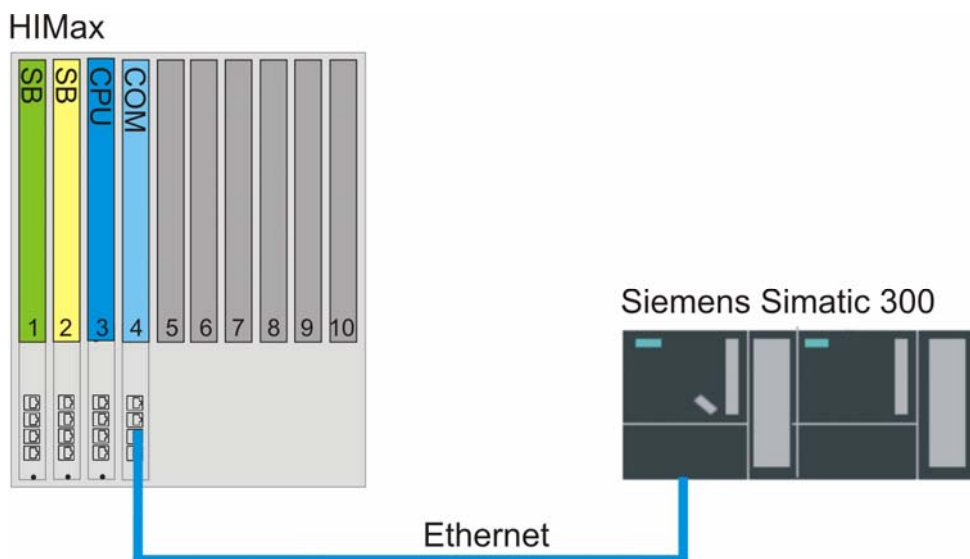


Figura 63: Ligação entre HIMAx e sistema de comando Siemens

Neste exemplo, é criado o protocolo Send/Receive over TCP num sistema de comando HIMAx. O sistema HIMAx deve comunicar-se ciclicamente via S&R TCP com um sistema de comando Siemens (p. ex., SIMATIC 300).

Neste caso o HIMAx (Client) é a estação ativa que estabelece a conexão TCP para o Siemens SIMATIC 300 passivo (Server). Depois de estabelecer a conexão, porém, ambos os sistemas de comando tem direitos iguais e podem enviar e receber a qualquer momento.

Ao ligar o HIMAx e o Siemens SIMATIC 300 deve ser observado o seguinte:

Para o HIMAx valem os requisitos descritos na seção 8.1 (requisitos de sistema).

O HIMAx e o Siemens SIMATIC 300 são conectados pelas suas interfaces Ethernet.

O HIMAx e o Siemens SIMATIC 300 devem estar na mesma subrede e no caso de usar um roteador, devem possuir as respectivas entradas de roteamento.

Neste exemplo, devem ser enviados dois BYTES e um WORD do sistema de comando HIMAx ao Siemens SIMATIC 300. As variáveis são recebidas no SIMATIC 300 pelo bloco AG\_RECV (FC 6) e internamente transferidas ao bloco AG\_SEND (FC 5). Mediante o bloco AG\_SEND (FC 5) o SIMATIC 300 envia as variáveis de volta ao sistema de comando HIMAx sem alterações.

A transmissão das variáveis pode ser verificada pelo usuário após configuração com o HIMA Force-Editor.

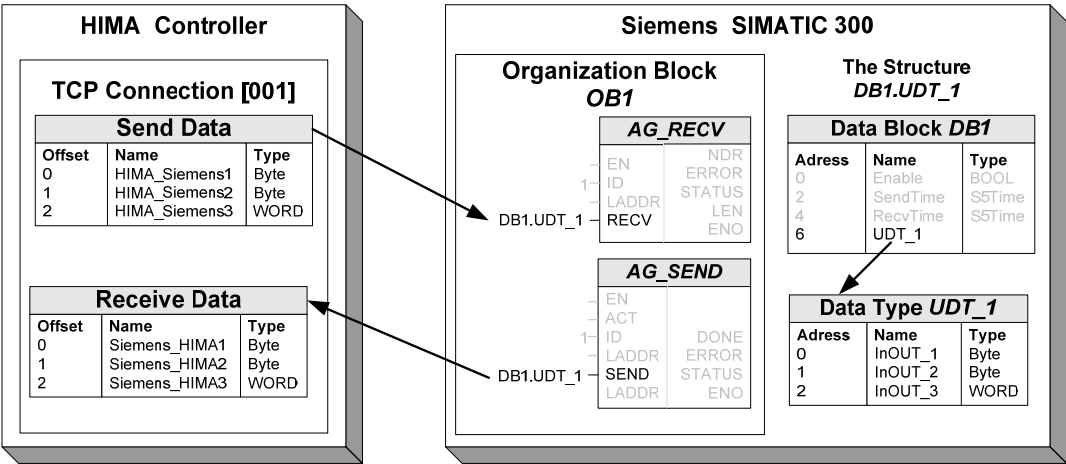


Figura 64: Transmissão de dados entre HIMax e sistema de comando Siemens

Descrição da configuração sistema de comando HIMax:

Elemento	Descrição
TCP Connection [001]	Neste diálogo estão todos os parâmetros necessários para a comunicação com o parceiro de comunicação (Siemens SIMATIC 300).
Send Data	Os offsets e tipos de variáveis no sistema de comando devem corresponder ao endereço e tipo de variáveis no tipo de dados <i>UDT_1</i> do SIMATIC 300.
Receive Data	Os offsets e tipos de variáveis no sistema de comando HIMax devem corresponder ao endereço e tipo de variáveis no tipo de dados <i>UDT_1</i> do SIMATIC 300.

Tabela 214: Configuração sistema de comando HIMax

Descrição da configuração sistema de comando Siemens SIMATIC 300:

Elemento	Descrição
Organization Block <i>OB1</i>	Os blocos funcionais <i>AG_RECV</i> (FC6) e <i>AG_SEND</i> (FC 5) devem ser criados e configurados no bloco de organização <i>OB1</i> .
<i>AG_RECV</i> (FC 6)	O bloco funcional <i>AG_RECV</i> (FC 6) transfere os dados recebidos do parceiro de comunicação ao tipo de dados <i>DB1.UDT_1</i> . As entradas <i>ID</i> e <i>LADDR</i> devem ser configuradas de acordo para a comunicação com o parceiro de comunicação.
<i>AG_SEND</i> (FC 5)	O bloco funcional <i>AG_SEND</i> (FC 5) transmite os dados do tipo de dados <i>DB1.UDT_1</i> ao parceiro de comunicação. As entradas <i>ID</i> e <i>LADDR</i> devem ser configuradas de acordo para a comunicação com o parceiro de comunicação.
Data Block <i>DB1</i>	O tipo de dados <i>UDT_1</i> é definido no bloco de dados <i>DB1</i> .
Data Type <i>UDT_1</i>	Os endereços e tipos de variáveis no sistema de comando SIMATIC 300 devem corresponder aos offsets e tipos do sistema de comando. O tipo de dados <i>UDT_1</i> transfere os dados de trabalho recebidos e memoriza os mesmos até sua transmissão ao parceiro de comunicação.

Tabela 215: Configuração Siemens SIMATIC 300



## 8.2.1 Configuração S&amp;R TCP do sistema de comando Siemens SIMATIC 300

**i**

As seguintes instruções passo-a-passo para a configuração do sistema de comando Siemens não pretendem ser completas.

Todas as indicações são sem garantia, para elaborar o projeto do sistema de comando Siemens, a documentação da Siemens é decisiva.

**Assim cria-se o S&R TCP-Server no projeto do SIMATIC 300:**

1. Iniciar o SIMATIC Manager.
2. No SIMATIC Manager, abrir o projeto do sistema de comando SIMATIC 300.
3. Neste projeto, criar e configurar a *Industrial Ethernet* e as conexões *MPI*.

**Assim cria-se o tipo de dados UDT1 com as seguintes variáveis:**

1. Abrir a pasta *Function Blocks* no Siemens SIMATIC Manager.
2. No menu principal, abrir **Add, S7 Function Block, Data Block** e criar um tipo de dados.
3. Atribuir ao tipo de dados o nome **UDT1**
4. Atribuir ao tipo de dados o nome simbólico **UDT1\_1**.
5. No tipo de dados *UDT\_1*, criar as três variáveis **InOut\_x** como na figura abaixo.

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	InOut_1	BYTE	B#16#0	
+1.0	InOut_2	BYTE	B#16#0	
+2.0	InOut_3	WORD	W#16#0	
=4.0		END_STRUCT		

Figura 65: Lista de variáveis no bloco UDT1 da Siemens

**i**

Na troca de dados cíclica e acíclica deve ser observado que alguns sistemas de comando (p. ex., SIMATIC 300) inserem os chamados *Pad Bytes*. Isso garante que todos os tipos de dados que são maiores do que um Byte sempre iniciam num offset par e que o comprimento total de todas as variáveis definidas também sempre seja um valor par.

Nestes casos, devem ser inseridos bytes vazios nos respectivos lugares no sistema de comando HIMax (veja Capítulo 8.6, Sistemas de outros fabricantes com Pad Bytes).

Assim, cria-se o bloco de dados DB1 para os blocos funcionais FC 5 e FC 6:

1. No menu principal, seleccionar **Add, S7 Function Block, Data Block** e criar um bloco de dados.
2. Atribuir ao bloco de dados o nome **DB1**.
3. Atribuir ao bloco de dados o nome simbólico **DB1**.
4. Atribuir ao bloco de dados *DB1* o tipo de dados *UDT\_1*.
5. No bloco de dados *DB1*, parametrizar os tipos de dados como mostrado na figura abaixo.

Address	Name	Type	Initial val	Comment
0.0		STRUCT		
+0.0	Enable	BOOL	TRUE	
+2.0	SendTime	S5TIME	S5T#100MS	
+4.0	RecvTime	S5TIME	S5T#10MS	
+6.0	UDT_1	"UDT_1"		
=10.0		END_STRUCT		

Figura 66: Lista de variáveis no bloco DB1 da Siemens

Assim criam-se os seguintes símbolos no Symbol Editor

1. Abrir a janela de diálogo *KOP/AWL/FUP* com clique duplo no bloco de organização **OB1**.
2. Abrir no menu principal **Extras, Symbol Table** o editor de símbolos.
3. Completar o *Symbol Editor* com as variáveis **M 1.0...MW 5** como mostrado na figura abaixo.

S7-Programm(1) (Symbole) -- S7_COMTEST\SIMATIC 300-				
	Status	Symbol	Address	Data type
1		DB1	DB 1	DB 1
2		RecDone	M 1.0	BOOL
3		RecError	M 1.1	BOOL
4		SendDone	M 1.2	BOOL
5		SendError	M 1.3	BOOL
6		RecStatus	MW 1	WORD
7		RecLen	MW 3	INT
8		SendStatus	MW 5	WORD
9		Cycle Execution	OB 1	OB 1
10		UDT_1	UDT 1	UDT 1
11				

Figura 67: SIMATIC-Symbol Editor

**Assim cria-se o bloco FC AG\_RECV (FC 6):**

1. Abrir a janela de diálogo *KOP/AWL/FUP*.
2. Selecionar os seguintes blocos FC em sequência da estrutura na parte esquerda do Symatic-Manager:
  - 1 *OR gate*
  - 1 *S\_VIMP*
  - 1 *AG\_RECV (FC 6)*.
3. Puxar estes *function blocks* mediante Drag&Drop para o bloco de organização *OB1*.
4. Conectar e configurar os *blocos funcionais* como mostrado na figura abaixo.
5. Clicar com o botão direito no bloco FC **AG\_RECV (FC 6)** e selecionar **Properties**.
6. Desativar **Active Connection Setup** e configurar as portas.
7. Anotar os parâmetros de bloco *LADDR* e introduzir os mesmos no esquema de função no bloco *AG\_RECV (FC 6)*.

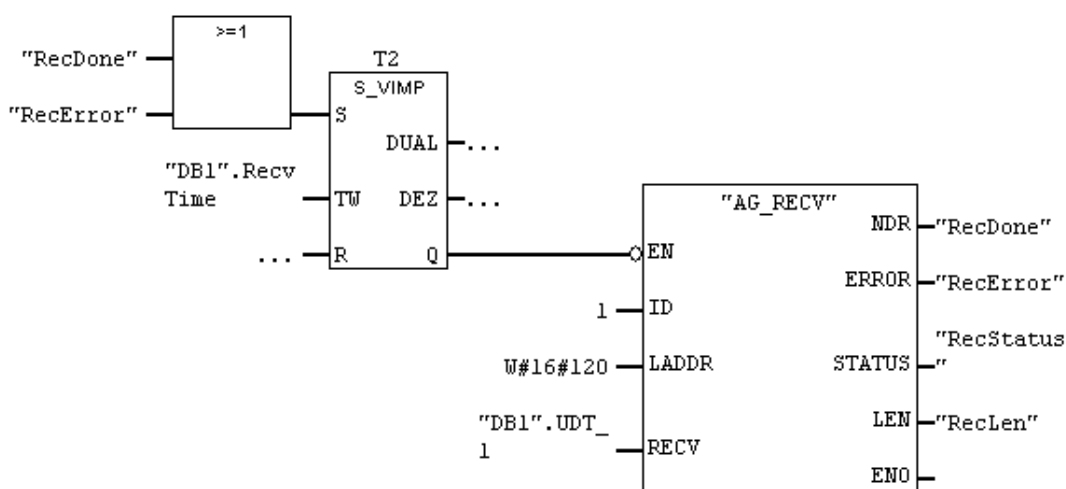


Figura 68: Esquema de função para recepção

**Assim cria-se o bloco FC AG\_SEND (FC 5):**

1. Abrir a janela de diálogo *KOP/AWL/FUP*.
2. Selecionar os seguintes blocos FC em sequência da estrutura na parte esquerda do Symatic-Manager:
  - 1 *OR gate*
  - 1 *S\_VIMP*
  - 1 *AG\_SEND (FC 5)*.
3. Puxar estes *function blocks* mediante Drag&Drop para o bloco de organização *OB1*.
4. Conectar e configurar os *blocos funcionais* como mostrado na figura abaixo.
5. Clicar com o botão direito no bloco FC **AG\_SEND (FC 5)** e selecionar **Properties**.
6. Desativar **Active Connection Setup** e configurar as portas.
7. Anotar os parâmetros de bloco *LADDR* e introduzir os mesmos no esquema de função no bloco *AG\_SEND (FC 5)*.

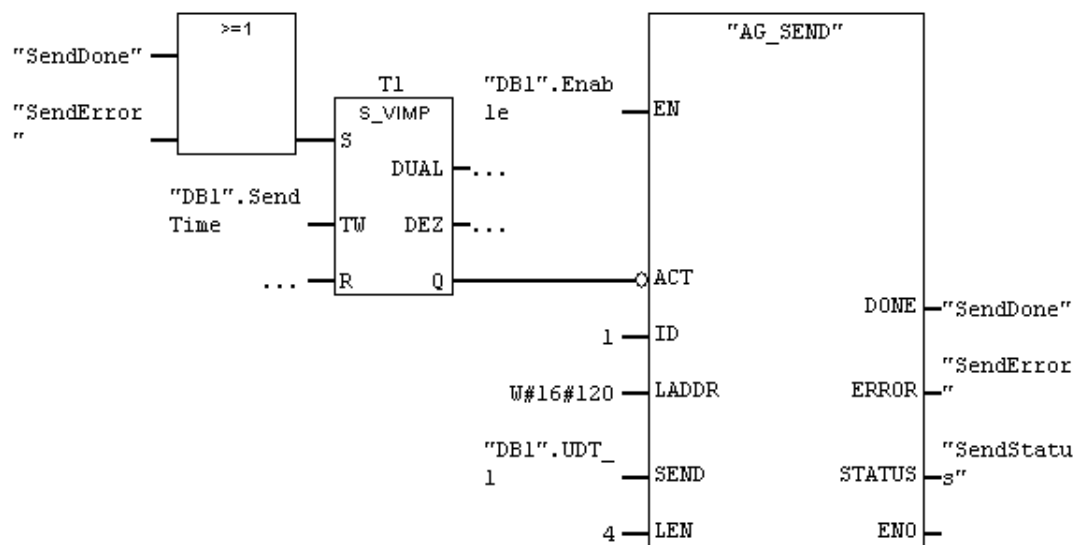


Figura 69: Esquema de função para envio

**Assim carrega-se o código para o sistema de comando SIMATIC 300:**

1. Iniciar o **Code Generator** para o programa.
2. Certificar-se de que os códigos tenham sido gerados sem erros.
3. Carregar o código para o sistema de comando SIMATIC 300.

### 8.2.2 Configuração S&R TCP do sistema de comando HIMax

Para a configuração dos sistemas de comando HIMax e para a utilização da ferramenta de programação SILworX, recomendamos o manual “Primeiros passos SILworX”.

**Assim criam-se as seguintes variáveis globais no editor de variáveis:**

1. Selecionar na árvore de estrutura **Configuration, Global Variables**.
2. Clicar com o botão direito em **Global Variables** e selecionar **Edit**.
3. Criar as variáveis globais como na Tabela 216.

Name	Tipo
Siemens_HIMA1	Byte
Siemens_HIMA2	Byte
Siemens_HIMA3	WORD
HIMA_Siemens1	Byte
HIMA_Siemens2	Byte
HIMA_Siemens3	WORD

Tabela 216: Variáveis globais

**Assim cria-se o protocolo S&R TCP no recurso:**

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **Protocols** e selecionar no menu de contexto **New**.
3. Selecionar **Send/Receive over TCP** e introduzir um nome para o protocolo.
4. Confirmar com OK para criar um novo protocolo.
5. Clicar com o botão direito em **Send/Receive over TCP** e selecionar no menu de contexto **Properties**.
6. Selecionar **COM-Module**. Os demais parâmetros mantêm os valores padrão.

**Assim cria-se a conexão TCP:**

1. Clicar com o botão direito em **Send/Receive over TCP** e selecionar no menu de contexto **New**.
2. Clicar com o botão direito em **TCP Connection** e selecionar no menu de contexto **Properties**.
3. Configurar as características como na figura abaixo.

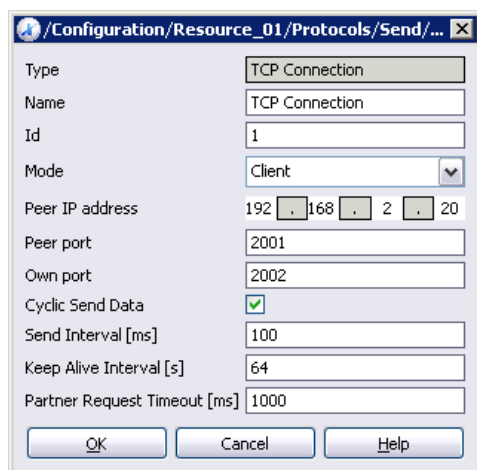


Figura 70: Características da conexão TCP no SILworX



*Se uma troca de dados cíclica entre dois sistemas de comando deve ser parametrizada, deve estar ativada a opção Cyclic data transfer no diálogo Properties da conexão TCP.*

**Assim configuram-se os dados de recepção do sistema de comando H1Max.**

1. Clicar com o botão direito em **TCP Connection** e selecionar no menu de contexto **Edit**.
2. Selecionar o registro **Process Variables**.
3. No **Object Panel**, selecionar as seguintes variáveis globais e puxar mediante Drag&Drop para a área **Inputs Signals**.

Variáveis globais	Tipo
Siemens_H1MA1	Byte
Siemens_H1MA2	Byte
Siemens_H1MA3	WORD

Tabela 217: Variáveis para dados de recepção

4. Abrir o menu de contexto com clique direito num lugar vazio na área **Register Inputs** e selecionar **New Offsets** para renumerar os offsets das variáveis.

---

*i*

Observe que os offsets das variáveis no sistema de comando H1Max devem corresponder ao endereço das variáveis no tipo de dados *UDT\_1* do SIMATIC 300.

---

**Assim configuram-se os dados de envio do sistema de comando H1Max.**

1. Clicar com o botão direito em **TCP Connection** e selecionar no menu de contexto **Edit**.
2. Selecionar o registro **Process Variables**.
3. No **Object Panel**, selecionar as seguintes variáveis globais e puxar mediante Drag&Drop para a área **Inputs Signals**.

Variáveis globais	Tipo
H1MA_Siemens1	Byte
H1MA_Siemens2	Byte
H1MA_Siemens3	WORD

Tabela 218: Variáveis para dados de envio

4. Abrir o menu de contexto com clique direito num lugar vazio na área **Register Inputs** e selecionar **New Offsets** para renumerar os offsets das variáveis.

---

*i*

Observe que os offsets das variáveis no sistema de comando H1Max devem corresponder ao endereço das variáveis no tipo de dados *UDT\_1* do SIMATIC 300.

---

**Verificar a configuração S&R TCP:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, Send/Receive over TCP**.
2. Clicar no botão **Verification** no Action Bar e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.

### 8.3 Funções de menu protocolo S&R TCP

#### 8.3.1 Edit

A janela de diálogo **Edit** do protocolo S&R TCP contém o seguinte registro.

##### Variáveis de sistema

Com as *System variables*, é possível avaliar o estado do protocolo TCP Send Receive no programa de aplicação.

Elemento	Descrição
Active Connection Count	Variável de sistema que fornece a quantidade de conexões ativas (não avariadas).
Disturbed Connection Count	Variável de sistema que fornece a quantidade de conexões avariadas. Avariado significa que a conexão TCP foi interrompida por um timeout ou erro.
Status	Sem função

Tabela 219: Variáveis de sistema S&R TCP

#### 8.3.2 Características

A troca de dados via conexão TCP ocorre de forma cíclica ou acíclica. Para a troca de dados acíclica, são necessários os blocos funcionais S&R TCP.

A troca de dados cíclica e não cíclica não podem ser utilizadas juntas na mesma conexão.

##### Informações gerais

Name	Descrição
Type	Send/Receive over TCP
Name	Nome para o protocolo Send/Receive-over-TCP. No máximo 31 caracteres.
Module	Seleção do módulo COM no qual o protocolo é processado.
Activate Max. $\mu$ P Budget	Ativado: Transferir o limite do $\mu$ P Budget do campo Max. $\mu$ P-Budget in [%].  Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P-Budget in [%]	Valor máximo de $\mu$ P-Budget do módulo que pode ser produzido ao processar o protocolo.  Faixa de valores: 1...100% Valor padrão: 30%
Behavior on CPU/COM Connection Loss	No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. (P. ex., se o módulo de comunicação é puxado para fora durante a comunicação). <div style="display: flex; justify-content: space-between;"> <div>Adopt Initial Data</div> <div>As variáveis de entrada são resetadas para os valores iniciais.</div> </div> <div style="display: flex; justify-content: space-between;"> <div>Retain Last Value</div> <div>As variáveis de entrada mantêm o último valor.</div> </div>

Tabela 220: Características gerais S&R TCP

## CPU/COM

Os parâmetros pré-definidos garantem a troca de dados mais rápida possível dos dados S&R TCP entre o módulo COM (COM) e o módulo CPU (CPU) no sistema de comando. Estes parâmetros apenas devem ser alterados se uma redução da carga COM e/ou CPU é necessária para uma aplicação e se o processo permitir.

- 
- i** A alteração dos parâmetros apenas se recomenda para programadores experientes. Um aumento do tempo de atualização COM e CPU também significa que o tempo de atualização real dos dados S&R TCP aumenta. As requisições de tempo do sistema devem ser verificadas.
- 

Name	Descrição
Process data refresh rate [ms]	Tempo de atualização em milissegundos com o qual os dados do protocolo S&R TCP são trocados entre COM e CPU. Se o Refresh Rate for zero ou menor do que o tempo de ciclo do sistema de comando, a troca de dados ocorre o mais rápido possível. Faixa de valores: 0...( $2^{31} - 1$ ) Valor padrão: 0
Force Process Data Consistency	Ativado: Transferência dos dados S&R TCP da CPU para COM dentro de um ciclo de CPU.  Desativado: Transferência dos dados S&R TCP (no máximo 1100 Byte por direção de dados) da CPU para COM, distribuídos sobre vários ciclos de CPU.

Tabela 221: Parametros COM/CPU



## 8.4 Funções de menu conexão TCP

### 8.4.1 Edit

Mediante a função de menu **Edit** podem ser acessados os registro de variáveis de processo e variáveis de sistema.

#### Variáveis de processo

##### Input Signals

As variáveis para a troca de dados cíclica que devem ser recebidas por este sistema de comando são introduzidas na área *Input Signals*.

No registro *Input Signals* pode ser criado um número livre de variáveis. Os offsets e tipos de variáveis, porém, devem ser idênticos com os offsets e tipos de variáveis (dados de envio) do parceiro de comunicação.

##### Output Signals

As variáveis para a troca de dados cíclica que devem ser enviadas por este sistema de comando são introduzidas na área *Output Signals*.

No registro *Output Signals* pode ser criado um número livre de variáveis. Os offsets e tipos de variáveis, porém, devem ser idênticos com os offsets e tipos de variáveis (dados de recepção) do parceiro de comunicação.

### 8.4.2 Variáveis de sistema

Com as variáveis no registro *System variables*, é possível avaliar o estado da conexão TCP no programa de aplicação.

Name	Descrição
Bytes received	Quantidade de bytes recebidos até agora
Bytes sent	Quantidade de bytes enviados até agora
Error Code	Código de erro da conexão TCP. Veja Capítulo 8.8.3.
Error Code Timestamp [ms]	Fração de milissegundos do carimbo de tempo. Momento da ocorrência do erro.
Error Code Timestamp [s]	Fração de segundos do carimbo de hora. Momento da ocorrência do erro.
Partner Request Timeout	Com transmissão de dados cíclica: Tempo de timeout dentro do qual após um envio de dados deve ser recebido no mínimo um envio de dados pelo parceiro de comunicação.  0 = Desliga 1...2 <sup>31</sup> - 1 [ms]
Partner Connection State	Se dentro do tempo do timeout não for recebido um envio de dados, o status estado de conexão parceiro é colocado em <i>Not Connected</i> e a conexão é novamente estabelecida.  0 = Sem conexão 1 = Conexão OK
Status	Status da conexão TCP. Veja Capítulo 8.8.5.

Tabela 222: Variáveis de sistema

## 8.4.3 Características

A troca de dados via conexão TCP ocorre de forma cíclica ou acíclica. Para a troca de dados acíclica, são necessários os blocos funcionais S&R TCP. Na comunicação de dados cíclica, a operação de blocos funcionais S&R TCP não é possível.

Name	Descrição
Type	Conexão TCP
Name	Nome livre inequívoco para uma conexão TCP. No máximo 31 caracteres.
Id	Número de identificação livre, mas inequívoco para cada conexão TCP. O ID também é necessário como referência nos blocos funcionais S&R TCP. Faixa de valores 0..255 Valor padrão: 0
Mode	Server (Valor padrão): Esta estação trabalha como servidor, ou seja, no modo passivo. A conexão deve ser iniciada pelo parceiro de comunicação (Client). Depois de estabelecer a conexão pela primeira vez, porém, ambas as estações tem direitos iguais e podem enviar dados a qualquer momento. A introdução da própria porta é necessária.
	Servidor com parceiro definido: Esta estação trabalha como servidor, ou seja, no modo passivo. A conexão deve ser iniciada pelo parceiro de comunicação (Client). Depois de estabelecer a conexão pela primeira vez, porém, ambas as estações tem direitos iguais e podem enviar dados a qualquer momento. Aqui é introduzido o endereço IP e/ou a porta do parceiro de comunicação, assim, apenas o parceiro de comunicação definido pode estabelecer uma comunicação. Todas as outras estações são ignoradas. Se um dos parâmetros (endereço IP ou porta) é colocado a zero, não haverá verificação para este parâmetro.
	Client: Esta estação trabalha como cliente, ou seja, a estação inicia o estabelecimento da conexão com o parceiro de comunicação. Necessita a indicação do endereço IP e da porta do parceiro de comunicação. Opcionalmente também pode ser indicada uma porta própria.
Partner IP Address	Endereço IP do parceiro de comunicação. 0.0.0.0: endereço IP livre permitido. Área válida: 1.0.0.0...223.255.255.255, exceto: 127.x.x.x Valor padrão: 0
Partner Port	Porta do parceiro de comunicação. 0: Porta livre. Portas reservadas ou já ocupadas (1...1024) são recusadas pelo sistema operacional COM. Faixa de valores 0..65535 Valor padrão: 0
Own Port	Porta própria. 0: Porta livre. Portas reservadas ou já ocupadas (1...1024) são recusadas pelo sistema operacional COM. Faixa de valores 0..65535 Valor padrão: 0

Name	Descrição
Cyclic data transfer	Desativado (Valor padrão): Envio de dados cíclico está desativado. A troca de dados através desta conexão TCP deve ser programada com blocos funcionais. Nesta conexão não podem ser definidos dados de E/S cíclicos.
	Ativado: Envio de dados cíclico está ativo. Os dados são definidos no diálogo variáveis de processo da conexão TCP. Dados de recepção devem estar definidos. Nesta conexão, não podem ser operados blocos funcionais.
Send interval [ms]	Apenas editável com envio de dados cíclico. Aqui é ajustado o intervalo de envio. Faixa de valores 10...2147483647 ms (valores menores são arredondados para 10 ms) Valor padrão: 0
Keep Alive Interval [s]	É o tempo até a supervisão de conexão disponibilizada pelo TCP se tornar ativa. Zero desativa a supervisão da conexão. Se dentro do KeepAlive Interval ajustado não forem trocados dados, são enviadas amostras teste KeepAlive ao parceiro de comunicação. Se a conexão ainda estiver ativa, as amostras teste KeepAlive são confirmadas pelo parceiro de comunicação. Se durante um tempo > 10 intervalos KeepAlive não houver nenhuma troca de dados entre os parceiros, a conexão é encerrada. Se após envio de um pacote de dados não for recebida nenhuma resposta, o pacote de dados é repetido em determinados intervalos. A conexão é interrompida após 12 tentativas sem êxito (aprox. 7 minutos). Faixa de valores 1...65535s Valor padrão: 0 = Desativado
Partner Request Timeout [ms]	Com transmissão de dados cíclica: Tempo de timeout dentro do qual após um envio de dados deve ser recebido no mínimo um envio de dados pelo parceiro de comunicação. Se dentro do tempo do timeout não for recebido um envio de dados, o status estado <i>Partner connection</i> é colocado em <i>Not Connected</i> e a conexão é novamente estabelecida. Depois de fechar a conexão por um timeout ou um outro erro, o lado ativo restabelece a conexão com um retardo de 10 x PartnerRequestTimeout ou 10 segundos, se o PartnerRequestTimeout = 0. O lado passivo abre a porta já depois da metade do tempo. 0 = Desliga Faixa de valores $1 \dots 2^{31} - 1$ [ms] Valor padrão: 0

Tabela 223: Características conexão S&amp;R TCP

## 8.5 Troca de dados

O S&R TCP trabalha conforme o princípio de cliente/servidor. A conexão deve ser iniciada pelo parceiro de comunicação que está configurado como cliente. Depois de estabelecer a conexão pela primeira vez, porém, ambos os parceiros de comunicação tem direitos iguais e podem enviar dados a qualquer momento.

O S&R TCP não possui um protocolo próprio de proteção dos dados, mas usa para isso o protocolo TCP/IP. Como o TCP envia os dados como "Data Stream", deve ser garantido que os offsets e os tipos das variáveis a serem trocadas sejam idênticos do lado de recepção e do lado de envio.

O S&R TCP é compatível à interface Siemens SEND/RECEIVE e permite a troca de dados cíclica com os blocos funcionais Siemens S7 AG\_SEND (FC5) e AG\_RECV (FC6) (veja Capítulo 8.2, Exemplo de configuração S&R TCP).

Além disso, a HIMA disponibiliza cinco blocos funcionais S&R TCP com os quais a comunicação pode ser controlada e individualmente adaptada pelo programa de aplicação. Mediante os blocos funcionais S&R TCP, é possível receber e enviar quaisquer protocolos (p. ex., Modbus) que são transmitidos via TCP.

### 8.5.1 Conexões TCP

Para cada conexão via S&R TCP com um parceiro de comunicação deve ser criada no mínimo uma conexão TCP no sistema de comando HIMax.

Nas *Properties* da conexão TCP devem ser introduzidos o número de identificação da conexão TCP e os endereços/ports do próprio sistema de comando e do parceiro de comunicação.

No máximo 32 conexões TCP podem ser criadas num sistema de comando HIMax.

As conexões TCP criadas devem ter números de identificação diferentes e endereços/portas diferentes.

#### **Assim cria-se uma nova conexão TCP:**

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **Protocols** e selecionar no menu de contexto **New**.
3. Selecionar **Send/Receive over TCP** e introduzir um nome para o protocolo.
4. Confirmar com OK para criar um novo protocolo.
5. Clicar com o botão direito em **Send/Receive over TCP** e selecionar no menu de contexto **Properties**.
6. Selecionar **COM-Module**. Os demais parâmetros mantêm os valores padrão.

---

**DICA** O sistema de comando HIMax/HIMatrix e o sistema de outros fabricantes devem estar na mesma subrede e no caso de usar um roteador, devem possuir as respectivas entradas de roteamento.

---

### 8.5.2 Troca de dados cíclica

Se a troca de dados cíclica é usada, um intervalo de envio deve ser definido no sistema de comando HIMax/HIMatrix e no parceiro de comunicação.

O intervalo de envio define o intervalo cíclico dentro do qual o parceiro de comunicação remetente envia as suas variáveis ao parceiro de comunicação destinatário.

- Para garantir uma troca de dados contínua, deve ser definido aproximadamente o mesmo intervalo de envio em ambos os parceiros de comunicação (veja Capítulo 8.5.5, Controle de fluxo).
- A opção *Cyclic Data Transfer* deve estar ativada na conexão TCP usada para a troca de dados cíclica.
- Numa conexão TCP em que a opção *Cyclic Data Transfer* está ativada, não podem ser usados blocos funcionais.
- As variáveis a serem enviadas e recebidas são atribuídas na janela de diálogo *Process Variable* da conexão TCP. Dados de recepção **devem** estar presentes, dados de envio são opcionais.



As mesmas variáveis (mesmos offsets e tipos) que são definidas em uma das estações como dados de envio, devem ser definidas como dados de recepção na outra estação.

---

### 8.5.3 Troca de dados acíclica com blocos funcionais

A troca de dados acíclica é controlada no sistema de comando HIMax/HIMatrix pelo programa de aplicação via blocos funcionais S&R TCP.

Assim, é possível controlar a troca de dados com um temporizador ou interruptor mecânico numa entrada física do sistema de comando HIMax/HIMatrix.

- A opção *Cyclic Data Transfer* deve estar desativada na conexão TCP usada.
- Sempre só um bloco funcional TCP S/R pode estar enviando de cada vez.
- As variáveis a serem enviadas e recebidas são atribuídas na janela de diálogo *Process Variable* dos blocos funcionais S&R TCP (todos exceto *Reset*).



As mesmas variáveis (mesmos offsets e tipos) que são definidas em uma das estações como dados de envio, devem ser definidas como dados de recepção na outra estação.

---

### 8.5.4 Troca de dados cíclica e acíclica simultaneamente

Para este fim, deve ser configurada uma conexão TCP para dados cíclicos e uma segunda para dados acíclicos. Ambas as conexões TCP devem usar diferentes *partner IP Addresses* e *Partner Ports*.

Uma única conexão TCP não pode ser usada para troca de dados cíclica e acíclica de forma compartilhada.

### 8.5.5 Controle de fluxo

O controle de fluxo é um componente do TCP e monitora a comunicação de dados contínua entre dois parceiros de comunicação.

Na transmissão de dados cíclica deve ser recebido pacote no máximo após 3 a 5 pacotes enviados, caso contrário, o envio é bloqueado até um novo pacote ser recebido ou até a supervisão conexão encerrar a conexão.

A quantidade (3..5) de envios possíveis sem receber um pacote depende do tamanho dos pacotes a serem enviados.

Quantidade = 5 para pacotes pequenos < 4kB.

Quantidade = 3 para pacotes grandes  $\geq 4$  kB.

- Na elaboração do projeto deve ser observado que nenhuma das duas estações envia mais dados do que a outra pode processar sincronicamente.
- Para a troca de dados cíclica, deve ser ajustado aproximadamente o mesmo intervalo de envio nos dois parceiros de comunicação.

## 8.6 Sistemas de outros fabricantes com Pad Bytes

Na troca de dados cíclica e acíclica deve ser observado que alguns sistemas de comando (p. ex., SIMATIC 300) inserem os chamados *pad bytes*. Isso garante que todos os tipos de dados que são maiores do que um Byte sempre iniciam num offset par e que o comprimento total dos pacotes (em Byte) também sempre seja um valor par.

No sistema de comando HIMax/HIMatrix devem ser inseridos *bytes vazios (dummy bytes)* ao invés dos *pad bytes* nos respectivos lugares.

Address	Name	Type	Initial value
0.0		STRUCT	
+0.0	InOut_1	BYTE	B#16#0
+2.0	InOut_3	WORD	W#16#0
=4.0		END_STRUCT	

Figura 71: Lista de variáveis Siemens

No sistema de comando Siemens é inserido um *pad byte* (invisível) para que a variável *InOut\_3* comece em um offset par.

Output Signals				
F	Name	Data type	Offset	Global Variable
1	InOut_1	BYTE	0	InOut_1
2	Dummy	BYTE	1	Dummy
3	InOut_3	WORD	2	InOut_3

Figura 72: Lista de variáveis HIMax/HIMatrix

No sistema de comando HIMax/HIMatrix deve ser inserido um *bytes vazios (dummy bytes)* para que a variável *InOut\_3* tenha o mesmo offset como no sistema de comando Siemens.

## 8.7 Blocos funcionais S&R TCP

Se a transmissão de dados cíclica for inflexível demais, os dados também podem ser enviados e recebidos mediante os blocos funcionais S&R TCP. A opção *Cyclic Data Transfer* deve ser desativada na conexão TCP usada.

Mediante os blocos funcionais S&R TCP, o usuário podem adaptar a transmissão de dados via TCP/IP aos requisitos do seu projeto da melhor forma possível.

Os blocos funcionais são parametrizados no programa de aplicação. Assim é possível atribuir e avaliar as funções (enviar, receber, reset) do sistema de comando HIMax/HIMatrix no programa de aplicação.

Os blocos funcionais S&R TCP apenas são necessários para a troca de dados acíclica. Para a troca de dados cíclica entre servidor e cliente, estes blocos funcionais não são necessários!

### i

A configuração dos blocos funcionais S&R TCP é descrita no Capítulo 13.1.

Os seguintes blocos funcionais estão à disposição:

Bloco funcional	Descrição da função
TCP_Reset (veja Capítulo 8.7.1)	Resetar a conexão TCP
TCP_Send (veja Capítulo 8.7.2)	Enviar dados
TCP_Receive (veja Capítulo 8.7.3)	Receber pacotes de dados de comprimento fixo
TCP_ReceiveLine (veja Capítulo 8.7.4)	Recepção de uma linha ASCII
TCP_ReceiveVar (veja Capítulo 8.7.5)	Receber pacotes de dados de comprimento variável (com campo de comprimento)
LATCH	É usado apenas dentro de outros blocos funcionais
PIG	É usado apenas dentro de outros blocos funcionais
PIGII	É usado apenas dentro de outros blocos funcionais

Tabela 224: Blocos funcionais para conexões S&R TCP

8.7.1 TCP\_Reset

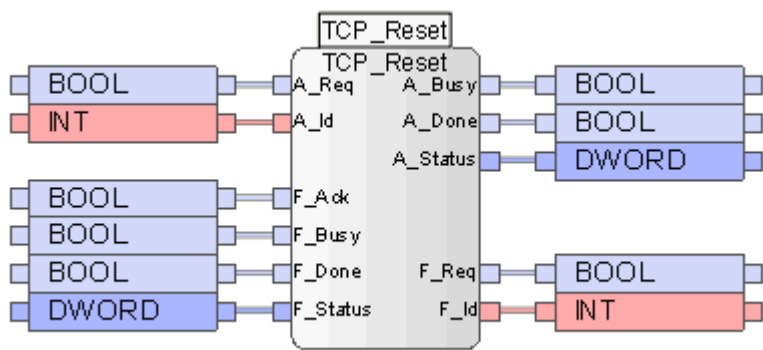


Figura 73: Bloco funcional TCP\_Reset

Com o bloco funcional **TCP\_Reset** é possível restabelecer uma conexão avariada se um bloco funcional Send ou Receive comunicar um erro de TIMEOUT (16#8A).

**i** Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Flanco positivo inicia o bloco	BOOL
A_Id	Número de identificação <i>ID</i> da conexão TCP avariada que deve ser resetada.	INT

Tabela 225: Entradas A bloco funcional TCP\_Reset

Saídas A	Descrição	Type
A_Busy	TRUE: O reset do bloco funcional ainda não encerrou.	BOOL
A_Done	TRUE: O processo de envio foi encerrado sem erros	BOOL
A_Status	Na saída <i>A_Status</i> é emitido o status e código de erro do bloco funcional e da conexão TCP.	DWORD

Tabela 226: Saídas A do bloco funcional TCP\_Reset



**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **Reset** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **Reset** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **TCP\_Reset** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas pelo usuário no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **TCP\_Reset** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **Reset** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Busy	BOOL
F_Done	BOOL
F_Status	DWORD

Tabela 227: Entradas F do bloco funcional TCP\_Reset

Conectar as *F-Outputs* do bloco funcional **TCP\_Reset** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **Reset** na árvore de estrutura também.

Saídas F	Tipo
F_Req	BOOL
F_Id	DWORD

Tabela 228: Saídas F do bloco funcional TCP\_Reset

**Criar o bloco funcional Reset correspondente na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, Send Receive over TCP, Function Blocks, New**.
2. Selecionar o bloco funcional **Reset**.
3. Clicar com o botão direito em bloco funcional **Reset** e selecionar **Edit**
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional **Reset** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **TCP\_Reset** no programa de aplicação.

Entradas	Tipo
ID	DWORD
REQ	BOOL

Tabela 229: Variáveis de sistema de entrada

Conectar as seguintes saídas do bloco funcional **Reset** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **TCP\_Reset** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
DONE	BOOL
STATUS	DWORD

Tabela 230: Variáveis de sistema de saída

**Para a operação do bloco funcional TCP\_Reset são necessários os seguintes passos:**

1. Atribuir no programa de aplicação na entrada *A\_ID* o número de identificação da conexão TCP avariada.
2. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.

---

**i**

O bloco funcional reage a uma mudança de flanco positiva em *A\_Req*.

---

A saída *A\_Busy* é TRUE até um Reset ter sido enviado para a conexão TCP definida. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Done* para TRUE.

8.7.2 TCP\_Send

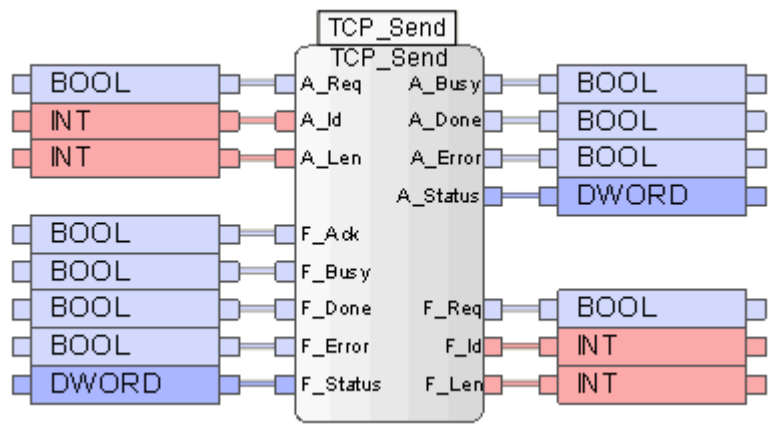


Figura 74: Bloco funcional TCP\_Send

O bloco funcional **TCP\_Send** serve para o envio acíclico de variáveis a um parceiro de comunicação. No parceiro de comunicação deve ser configurado um bloco funcional, p. ex., *Receive*, com as mesmas variáveis e offsets.

i

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Flanco positivo inicia o bloco.	BOOL
A_Id	O número de identificação da conexão TCP configurada ao parceiro de comunicação para o qual os dados devem ser enviados.	INT
A_Len	Quantidade das variáveis a serem enviadas em Bytes. A_Len deve ser maior do que zero e não pode terminar dentro de uma variável.	INT

Tabela 231: Entradas A do bloco funcional TCP\_Send

Saídas A	Descrição	Type
A_Busy	TRUE: O processo de envio ainda não terminou.	BOOL
A_Done	TRUE: O processo de envio foi encerrado sem erros	BOOL
A_Error	TRUE: Ocorreu um erro FALSE: Sem erro	BOOL
A_Status	Na saída A_Status é emitido o status e código de erro do bloco funcional e da conexão TCP.	DWORD

Tabela 232: Saídas A do bloco funcional TCP\_Send

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **Send** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **Send** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **TCP\_Send** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas pelo usuário no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **TCP\_Send** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **Send** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Busy	BOOL
F_Done	BOOL
F_Error	BOOL
F_Status	DWORD

Tabela 233: Entradas F do bloco funcional TCP\_Send

Conectar as *F-Outputs* do bloco funcional **TCP\_Send** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **Send** na árvore de estrutura também.

Saídas F	Tipo
F_Id	DWORD
F_Len	INT
F_Req	BOOL

Tabela 234: Saídas F do bloco funcional TCP\_Send

**Criar o bloco funcional Send correspondente na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, Send Receive over TCP, Function Blocks, New**.
2. Selecionar o bloco funcional **Send**.
3. Clicar com o botão direito em bloco funcional **Send** e selecionar **Edit**
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **Send** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **TCP\_Send** no programa de aplicação.

Entradas	Tipo
ID	DWORD
LEN	INT
REQ	BOOL

Tabela 235: Variáveis de sistema de entrada

Conectar as seguintes saídas do bloco funcional do bloco funcional **Send** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **TCP\_Send** no programa de aplicação.

Saídas	Tipo
Ack	BOOL
Busy	BOOL
Done	BOOL
ERROR	BOOL
STATUS	DWORD

Tabela 236: Variáveis de sistema de saída

Dados	Descrição
Dados de envio	No registro <i>Process Variables</i> pode ser criado um número livre de variáveis. Os offsets e tipos de variáveis, porém, devem ser idênticos com os offsets e tipos de variáveis de envio do parceiro de comunicação.

Tabela 237: Dados de envio

**Para a operação do bloco funcional TCP\_Send são necessários os seguintes passos:**

---

**i** As variáveis a serem enviadas devem ser criadas no registro *Process Variables* do diálogo *Send*. Os offsets e tipos das variáveis devem ser idênticos com os offsets e tipos das variáveis do parceiro de comunicação.

---

1. Colocar no programa de aplicação o ID da conexão TCP na entrada *A\_ID*.
  2. Colocar no programa de aplicação o comprimento das variáveis a serem enviadas em Byte na entrada *A\_Len*.
  3. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.
- 

**i** O bloco funcional reage a uma mudança de flanco positiva em *A\_Req*.

---

A saída *A\_Busy* vai para TRUE até as variáveis terem sido enviadas. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Done* para TRUE.

Se não foi possível executar o processo de envio com êxito, a saída *A\_Error* muda para TRUE e um código de erro é emitido na saída *A\_Status*.

8.7.3 TCP\_Receive

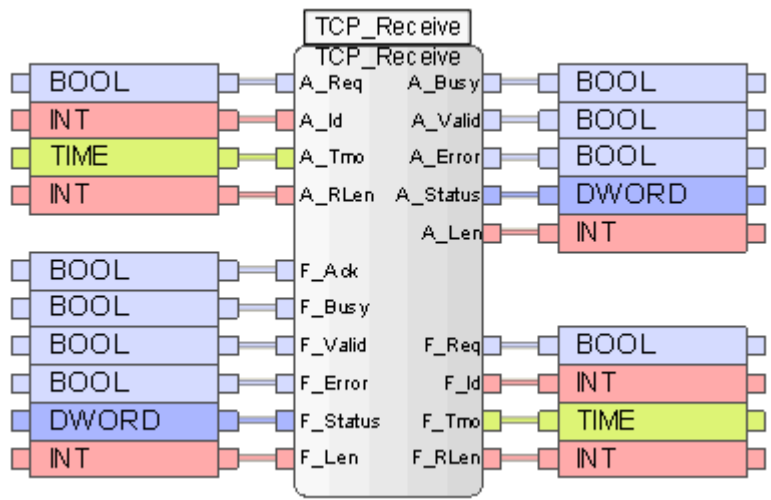


Figura 75: Bloco funcional TCP\_Receive

Com o bloco funcional **TCP\_Receive** é possível receber variáveis definidas de um parceiro de comunicação.

No parceiro de comunicação deve ser configurado um bloco funcional, p. ex., TCP\_Send, com as mesmas variáveis e offsets.

**i** Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Um flanco positivo inicia o bloco.	BOOL
A_Id	O número de identificação da conexão TCP configurada ao parceiro de comunicação do qual os dados devem ser recebidos.	INT
A_Tmo	Timeout de recepção. Se dentro deste intervalo de tempo não foram recebido dados, o bloco é encerrado com uma mensagem de erro. Se a entrada A_Tmo não é atribuída ou ajustada para zero, o timeout está desativado.	TIME
A_RLen	A_RLen é o comprimento esperado em bytes das variáveis a serem recebidas. A_RLen deve ser maior do que zero e não pode terminar dentro de uma variável.	INT

Tabela 238: Entradas A do bloco funcional TCP\_Receive

Saídas A	Descrição	Type
A_Busy	TRUE: A recepção dos dados ainda não terminou.	BOOL
A_Valid	TRUE: A recepção foi encerrada sem erros.	BOOL
A_Error	TRUE: Ocorreu um erro FALSE: Sem erro	BOOL
A_Status	Na saída <i>A_Status</i> são emitidos o status e código de erro do bloco funcional e da conexão TCP.	DWORD
A_Len	Quantidade de bytes recebidos.	INT

Tabela 239: Saídas A do bloco funcional TCP\_Receive

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **Receive** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **Receive** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **TCP\_Receive** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas pelo usuário no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **TCP\_Receive** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **Receive** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Busy	BOOL
F_Valid	BOOL
F_Error	BOOL
F_Status	DWORD
F_Len	INT

Tabela 240: Entradas A do bloco funcional TCP\_Receive

Conectar as *F-Outputs* do bloco funcional **TCP\_Receive** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **Receive** na árvore de estrutura também.

Saídas F	Tipo
F_Req	BOOL
F_Id	DWORD
F_Tmo	INT
F_RLen	INT

Tabela 241: Saídas F do bloco funcional TCP\_Receive

**Criar o bloco funcional Receive correspondente na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, Send Receive over TCP, Function Blocks, New**.
2. Selecionar o bloco funcional **Receive**.
3. Clicar com o botão direito em bloco funcional **Receive** e selecionar **Edit**
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **Receive** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **TCP\_Receive** no programa de aplicação.

Entradas	Tipo
ID	INT
REQ	BOOL
RLEN	INT
TIMEOUT	TIME

Tabela 242: Variáveis de sistema de entrada

Conectar as seguintes saídas do bloco funcional **Receive** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **TCP\_Receive** no programa de aplicação.

Saídas	Tipo
Ack	BOOL
Busy	BOOL
ERROR	BOOL
LEN	INT
STATUS	DWORD
VALID	BOOL

Tabela 243: Variáveis de sistema de saída

Dados	Descrição
Variáveis de recepção	No registro <i>Process Variables</i> pode ser criado um número livre de variáveis. Os offsets e tipos de variáveis, porém, devem ser idênticos com os offsets e tipos de variáveis de envio do parceiro de comunicação.

Tabela 244: Variáveis de recepção



**Para a operação do bloco funcional TCP\_Receive são necessários os seguintes passos:**

---

**i**

As variáveis de recepção devem ser criadas no registro *Process Variables* do diálogo *Receive*. Os offsets e tipos das variáveis de recepção devem ser idênticos com os offsets e tipos das variáveis de envio do parceiro de comunicação.

---

1. Atribuir no programa de aplicação na entrada *A\_ID* o número de identificação da conexão TCP.
  2. Colocar no programa de aplicação o timeout de recepção na entrada *A\_Tmo*.
  3. Colocar no programa de aplicação o comprimento das variáveis a serem recebidas em Byte na entrada *A\_RLen*.
  4. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.
- 

**i**

O bloco funcional inicia com uma mudança de flanco positiva em *A\_Req*.

---

A saída *A\_Busy* é TRUE até variáveis serem recebidas ou o tempo do timeout de recepção se esgotar. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Valid* ou *A\_Error* para TRUE.

Se a recepção for livre de erros, a saída *A\_Valid* muda para TRUE. As variáveis que foram definidas no registro Data podem ser avaliadas.

Se a recepção não for livre de erros, a saída *A\_Error* muda para TRUE e na saída *A\_Status* é emitido um código de erro.

8.7.4 TCP\_ReceiveLine

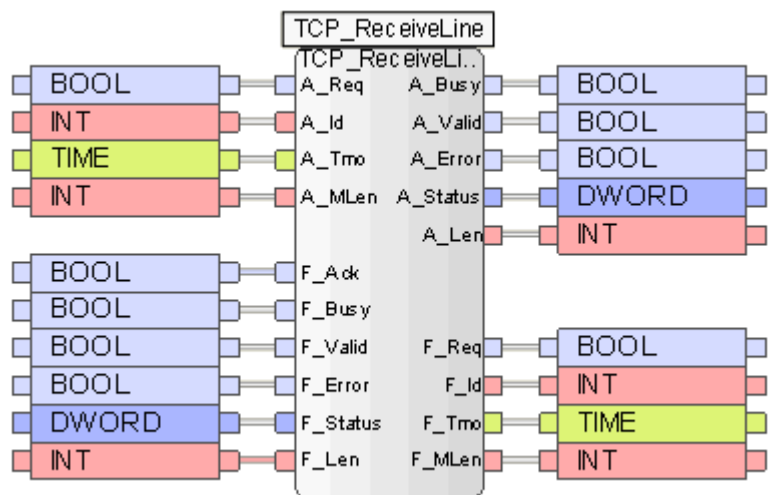


Figura 76: Bloco funcional TCP\_ReceiveLine

O bloco funcional **TCP\_ReceiveLine** é utilizado para a recepção de uma sequência de caracteres ASCII inclusive LineFeed (16#0A) de um parceiro de comunicação.

i

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

**Entradas e saídas do bloco funcional com o prefixo A:**

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Um flanco positivo inicia o bloco e coloca a conexão em estado pronto para recepção	BOOL
A_Id	O número de identificação da conexão TCP configurada ao parceiro de comunicação do qual os dados devem ser recebidos.	INT
A_Tmo	Timeout de recepção Se dentro deste intervalo de tempo não foram recebido dados, o bloco é encerrado com uma mensagem de erro. Se esta entrada permanecer em aberto ou é ajustada para zero, o timeout está desligado.	TIME
A_MLen	<i>A_MLen</i> é o comprimento máximo em bytes da linha a ser recebida. As variáveis de recepção devem ser criadas no registro <i>Data</i> no bloco funcional Com. Bytes transmitidos = Min ( <i>A_MLen</i> , comprimento da linha, comprimento da área de dados).	INT

Tabela 245: Entradas A do bloco funcional TCP\_ReceiveLine

Saídas A	Descrição	Type
A_Busy	TRUE: A recepção dos dados ainda não terminou.	BOOL
A_Valid	TRUE: A recepção foi encerrada sem erros.	BOOL
A_Error	TRUE: Ocorreu um erro FALSE: Sem erro	BOOL
A_Status	Na saída A_Status é emitido o status e código de erro do bloco funcional e da conexão TCP.	DWORD
A_Len	Quantidade de bytes recebidos.	INT

Tabela 246: Saídas A do bloco funcional TCP\_ReceiveLine

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **ReceiveLine** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **ReceiveLine** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **TCP\_ReceiveLine** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas pelo usuário no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **TCP\_ReceiveLine** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **ReceiveLine** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Busy	BOOL
F_Valid	BOOL
F_Error	BOOL
F_Status	DWORD
F_Len	INT

Tabela 247: Entradas F do bloco funcional TCP\_ReceiveLine

Conectar as *F-Outputs* do bloco funcional **TCP\_ReceiveLine** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **ReceiveLine** na árvore de estrutura também.

Saídas F	Tipo
A_Req	BOOL
A_Id	INT
A_Tmo	TIME
A_MLen	INT

Tabela 248: Saídas F do bloco funcional TCP\_ReceiveLine

**Criar o bloco funcional ReceiveLine correspondente na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, Send Receive over TCP, Function Blocks, New**.
2. Selecionar o bloco funcional **ReceiveLine**.
3. Clicar com o botão direito em bloco funcional **ReceiveLine** e selecionar **Edit**
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **ReceiveLine** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **TCP\_ReceiveLine** no programa de aplicação.

Entradas	Tipo
ID	INT
MLEN	INT
REQ	BOOL
TIMEOUT	TIME

Tabela 249: Variáveis de sistema de entrada

Conectar as seguintes saídas do bloco funcional **ReceiveLine** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **TCP\_ReceiveLine** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
ERROR	BOOL
LEN	INT
STATUS	DWORD
VALID	BOOL

Tabela 250: Variáveis de sistema de saída

Dados	Descrição
Variáveis de recepção	No registro <i>Process Variables</i> só faz sentido criar variáveis do tipo BYTE. Os offsets das variáveis devem ser idênticos com os offsets das variáveis do parceiro de comunicação.

Tabela 251: Variáveis de recepção

**Para a operação do bloco funcional TCP\_ReceiveLine são necessários os seguintes passos:**

---

**i**

As variáveis de recepção do tipo Byte devem ser criadas no registro *Process Variables* do diálogo *ReceiveLine*. Os offsets das variáveis de recepção devem ser idênticos com os offsets das variáveis de envio do parceiro de comunicação.

---

1. Atribuir no programa de aplicação na entrada *A\_ID* o número de identificação da conexão TCP.
  2. Colocar no programa de aplicação o timeout de recepção na entrada *A\_Tmo*.
  3. Colocar no programa de aplicação o comprimento máximo da linha a ser recebida na entrada *A\_MLen*.
- 

**i**

*A\_MLen* deve ser maior do que zero e define o tamanho do buffer de recepção em Byte. Se o buffer estiver cheio e ainda não ocorreu o fim da linha, o processo de leitura é encerrado sem mensagem de erro.

Na saída *A\_Len* é disponibilizada a quantidade de bytes recebidos:

Bytes recebidos = Min (*A\_MLen*, comprimento da linha, comprimento da área de dados).

---

4. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.
- 

**i**

O bloco funcional reage a uma mudança de flanco positiva em *A\_Req*.

---

A saída *A\_Busy* é TRUE até o buffer de recepção encher ou o fim de linha *LineFeed* ser recebido ou o tempo do timeout de recepção se esgotar. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Valid* ou *A\_Error* para TRUE.

Se a recepção for livre de erros, a saída *A\_Valid* muda para TRUE. As variáveis que foram definidas no registro Data podem ser avaliadas.

Se a recepção não for livre de erros, a saída *A\_Error* muda para TRUE e na saída *A\_Status* é emitido um código de erro.

## 8.7.5 TCP\_ReceiveVar

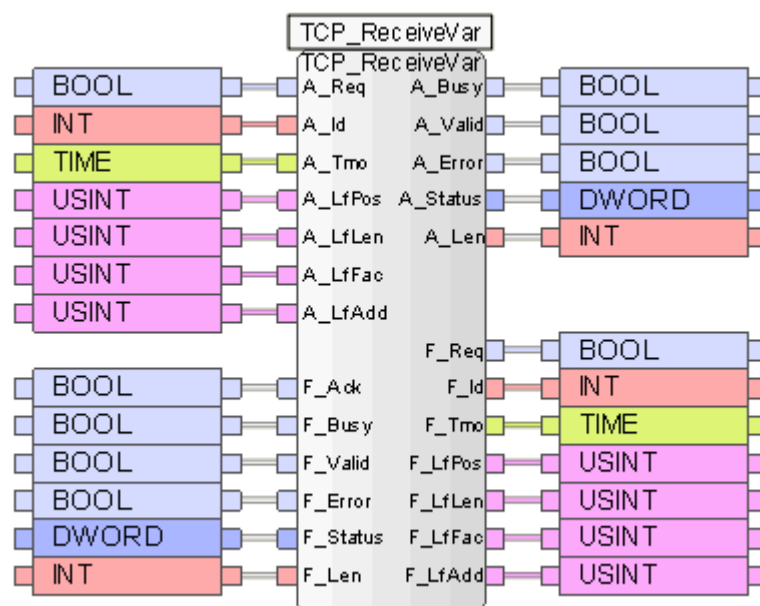


Figura 77: Bloco funcional TCP\_ReceiveVar

Com o bloco funcional **TCP\_ReceiveVar** é possível avaliar pacotes de dados de comprimento variável equipados com um campo de comprimento.

**i**

Para a configuração, puxar o bloco funcional mediante Drag&Drop da biblioteca de blocos funcionais para o programa de aplicação (veja também Capítulo 13.1).

### Descrição da função

Os pacotes de dados recebidos devem ter a estrutura representada na figura abaixo (p. ex., protocolo Modbus). A adaptação a qualquer formato de protocolo ocorre mediante ajuste dos parâmetros de introdução *A\_LfPos*, *A\_LfLen*, *A\_LfFac*, *A\_LfLen*.

O pacote de dados recebido consiste numa área de cabeçalho e numa área de dados de trabalho. A área de cabeçalho contém dados como endereço do participante, função de telegrama, campo de comprimento, etc., necessários para a conexão de comunicação. Para avaliar a área de dados de trabalho, a área de cabeçalho deve ser separada e o campo de comprimento deve ser lido.

O tamanho da área de cabeçalho é introduzido no parâmetro *A\_LfAdd*.

O comprimento da área de dados de trabalho deve ser lido do campo de comprimento do pacote de dados atualmente lido. A posição do campo de comprimento é introduzida no parâmetro *A\_LfPos*. O tamanho do campo de comprimento em Byte é introduzido no parâmetro *A\_LfLen*. Se o comprimento não for indicado em Byte deve ser introduzido o fator de conversão para isso em *A\_LfFac* (p. ex., 2 para Word ou 4 para Double Word).

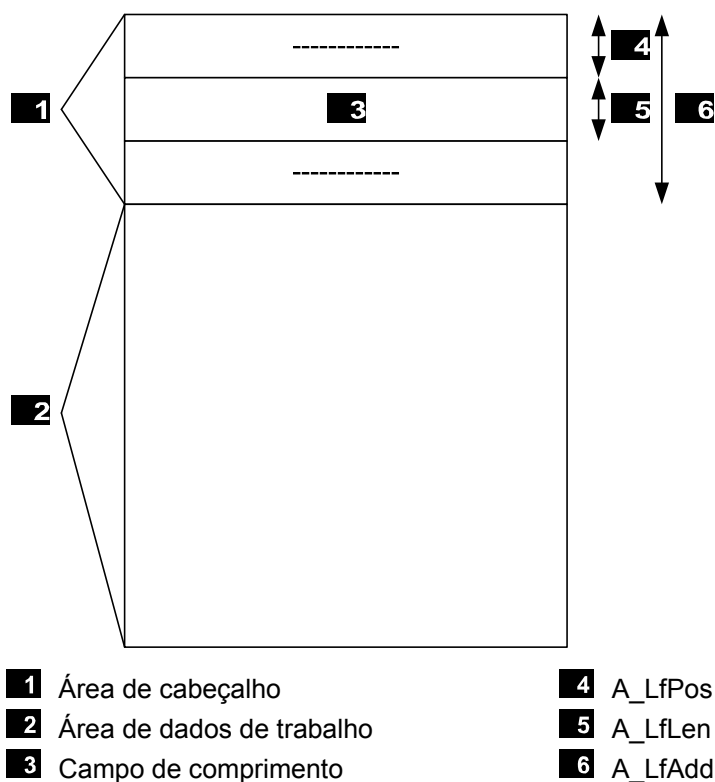


Figura 78: Estrutura dos pacotes de dados

### Entradas e saídas do bloco funcional com o prefixo A

Através destas entradas e saídas, é possível controlar e avaliar o bloco funcional com ajuda do programa de aplicação. O prefixo “A” significa “Application” – Aplicação.

Entradas A	Descrição	Type
A_Req	Um flanco positivo inicia o bloco funcional CPU.	BOOL
A_Id	O número de identificação <i>ID</i> da conexão TCP configurada a um parceiro de comunicação do qual o pacote de dados deve ser recebido.	DWORD
A_Tmo	Timeout de recepção Se dentro deste intervalo de tempo não foram recebido dados, o bloco é encerrado com uma mensagem de erro. Se esta entrada permanecer em aberto ou é ajustada para zero, o timeout está desligado.	INT
A_LfPos	Posição inicial do campo de comprimento no pacote de dados; a numeração inicia com zero (medido em Bytes).	USINT
A_LfLen	Tamanho do campo de comprimento <i>A_LfLen</i> em Bytes. 1, 2 ou 4 Bytes são admissíveis.	USINT
A_LfFac	Fator de conversão em Bytes se a introdução no campo de comprimento não for em Bytes. Se a entrada permanecer em aberto ou se for atribuído zero, o valor padrão 1 é assumido.	USINT
A_LfAdd	Tamanho do campo de cabeçalho em Bytes	USINT

Tabela 252: Entradas A do bloco funcional TCP\_ReceiveVar

Saídas A	Descrição	Type
A_Busy	TRUE: A recepção dos dados ainda não terminou.	BOOL
A_Valid	TRUE: A recepção de dados foi encerrada sem erros.	BOOL
A_Error	TRUE: Durante a leitura ocorreu um erro FALSE: Sem erro	BOOL
A_Status	Na saída <i>A_Status</i> é emitido o status e código de erro do bloco funcional e da conexão TCP.	DWORD
A_Len	Quantidade de bytes recebidos.	INT

Tabela 253: Saídas A do bloco funcional TCP\_ReceiveVar

**Entradas e saídas do bloco funcional com o prefixo F:**

Estas entradas e saídas do bloco funcional estabelecem a conexão co bloco funcional **ReceiveVar** na árvore de estrutura. O prefixo “F” significa “Field” – Campo.

**i**

A conexão do bloco funcional **ReceiveVar** na árvore de estrutura (na pasta Blocos funcionais) com o bloco funcional **TCP\_ReceiveVar** (no programa de aplicação) ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas pelo usuário no editor de variáveis.

Conectar as *F-Inputs* do bloco funcional **TCP\_ReceiveVar** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as saídas do bloco funcional **ReceiveVar** na árvore de estrutura também.

Entradas F	Tipo
F_Ack	BOOL
F_Busy	BOOL
F_Valid	BOOL
F_Error	BOOL
A_Status	DWORD
A_Len	INT

Tabela 254: Entradas F do bloco funcional TCP\_ReceiveVar

Conectar as *F-Outputs* do bloco funcional **TCP\_ReceiveVar** no programa de aplicação com as mesmas variáveis com as quais mais tarde serão conectadas as entradas do bloco funcional **ReceiveVar** na árvore de estrutura também.

Saídas F	Tipo
F_Req	BOOL
F_Id	INT
F_Tmo	TIME
F_LfPos	USINT
A_LfLen	USINT
A_LfFac	USINT
A_LfAdd	USINT

Tabela 255: Saídas F do bloco funcional TCP\_ReceiveVar



**Criar o bloco funcional ReceiveVar correspondente na árvore de estrutura:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, Send Receive over TCP, Function Blocks, New**.
2. Selecionar o bloco funcional **ReceiveVar**.
3. Clicar com o botão direito em bloco funcional **ReceiveVar** e selecionar **Edit**
  - ☒ A atribuição de variáveis para o bloco funcional se abre.

Conectar as entradas do bloco funcional do bloco funcional **ReceiveVar** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional **TCP\_ReceiveVar** no programa de aplicação.

Entradas	Tipo
ID	INT
Lf Add	USINT
Lf Fac	USINT
Lf Len	USINT
Lf Pos	USINT
REQ	BOOL
TIMEOUT	TIME

Tabela 256: Variáveis de sistema de entrada

Conectar as seguintes saídas do bloco funcional **ReceiveVar** na árvore de estrutura com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional **TCP\_ReceiveVar** no programa de aplicação.

Saídas	Tipo
ACK	BOOL
BUSY	BOOL
ERROR	BOOL
LEN	INT
STATUS	DWORD
VALID	BOOL

Tabela 257: Variáveis de sistema de saída

Dados	Descrição
Variáveis de recepção	No registro <i>Process Variables</i> pode ser criado um número livre de variáveis. Os offsets e tipos de variáveis, porém, devem ser idênticos com os offsets e tipos de variáveis de envio do parceiro de comunicação.

Tabela 258: Variáveis de recepção

**Para a operação do bloco funcional TCP\_ReceiveVar são necessários os seguintes passos:**

---

**i** As variáveis de recepção devem ser criadas no registro *Process Variables* do diálogo *Variable Receive*. Os offsets e tipos das variáveis de recepção devem ser idênticos com os offsets e tipos das variáveis de envio do parceiro de comunicação.

---

1. Atribuir no programa de aplicação na entrada *A\_ID* o número de identificação da conexão TCP.
2. Colocar no programa de aplicação o timeout de recepção na entrada *A\_Tmo*.
3. Atribuir no programa de aplicação os parâmetros *A\_LfPos*, *A\_LfLen*, *A\_LfFac* e *A\_LfAdd*.
4. Colocar no programa de aplicação a entrada *A\_Req* em TRUE.

---

**i** O bloco funcional inicia com uma mudança de flanco positiva em *A\_Req*.

---

A saída *A\_Busy* é TRUE até variáveis serem recebidas ou o tempo do timeout de recepção se esgotar. Depois disso, as saídas *A\_Busy* mudam para FALSE e *A\_Valid* ou *A\_Error* para TRUE.

Se a recepção for livre de erros, a saída *A\_Valid* muda para TRUE. As variáveis que foram definidas no registro Data podem ser avaliadas. A saída *A\_Len* contém a quantidade de bytes que realmente foram lidos.

Se a recepção não for livre de erros, a saída *A\_Error* muda para TRUE e na saída *A\_Status* é emitido um código de erro.

## 8.8 Control Panel (Send/Receive over TCP)

No Control Panel, o usuário pode verificar e controlar os ajustes do protocolo Send/Receive. Além disso, informações de status atuais (p. ex., conexões avariadas) do protocolo Send/Receive são exibidas.

**Assim abre-se o Control Panel para a supervisão do protocolo Send/Receive:**

1. Selecionar na árvore de estrutura **Resource**.
2. Selecionar da **Action Bar** Online.
3. Introduzir no **System-Login** os dados de acesso para abrir do Control Panel do recurso.
4. Selecionar na árvore de estrutura do Control Panel **Send/Receive Protocol**.

### 8.8.1 Campo de exibição parâmetros gerais

No campo de exibição são mostrados os seguintes valores do protocolo Send/Receive.

Elemento	Descrição
Name	Protocolo TCP SR
µP Load (planned) [%]	Veja Capítulo 7.2.3.2.
µP Load (actual) [%]	
Undisturbed Connections	Quantidade conexões não avariadas
Disturbed Connections	Disturbed Connection Count

Tabela 259: Campo de exibição protocolo S&R

### 8.8.2 Campo de exibição conexões TCP

No campo de exibição são mostrados os seguintes valores das conexões TCP selecionadas.

Elemento	Descrição
Name	Conexão TCP
Partner Timeout	Sim: Partner Request Timeout esgotou Não: Partner Request Timeout não esgotou
Connection State	Estado atual desta conexão 0x00: Conexão OK 0x01: Conexão encerrada 0x02: Servidor aguarda início de conexão 0x04: Client tenta estabelecer a conexão 0x08: Conexão está bloqueada
Peer Address	Endereço IP do parceiro de comunicação.
Peer Port	Porta do parceiro de comunicação.
Own Port	Porta deste sistema de comando.
Watchdog Time [ms]	Tempo de Partner Request Timeout atual dentro do qual após um envio de dados deve ser recebido no mínimo um envio de dados pelo parceiro de comunicação.
Error Code	Código de erro (veja Capítulo 8.8.3)
Timestamp Error Code [ms]	Carimbo de hora do último erro comunicado Faixa de valores: Segundos desde 1/1/1970 em milissegundos
Received Bytes [Byte]	Quantidade de bytes recebidos desta conexão TCP
Transmitted Bytes [Byte]	Quantidade de bytes enviados desta conexão TCP

Tabela 260: Campo de exibição Modbus Slaves

## 8.8.3 Código de erro da conexão TCP

Os códigos de erro podem ser lidos da variável *Error Code*.

Para cada conexão configurada: O status da conexão é composto do estado de conexão e do código de erro da última operação.

Código de erro decimal	Código de erro hexadecimal	Descrição
0	16#00	OK
4	16#04	Chamada de sistema interrompida
5	16#05	Erro I/O
6	16#06	Equipamento desconhecido
9	16#09	Descritor de soquete inválido
12	16#0C	Não existe mais memória
13	16#0D	Acesso recusado
14	16#0E	Endereço inválido
16	16#10	Equipamento está ocupado
22	16#16	Valor inválido (p. ex., no campo de comprimento)
23	16#17	A tabela de descritores está cheia
32	16#20	Conexão interrompida
35	16#23	Operação bloqueada
36	16#24	Operação em execução agora
37	16#25	Operação já em execução
38	16#27	Endereço de destino necessário
39	16#28	Mensagem longa demais
40	16#29	Tipo de protocolo incorreto para soquete
42	16#2A	Protocolo não disponível
43	16#2B	Protocolo não suportado
45	16#2D	Operação não suportada no soquete
47	16#2F	Endereço não suportado pelo protocolo
48	16#30	Endereço já está em uso
49	16#31	Endereço não pode ser atribuído
50	16#32	Rede não opera
53	16#35	Software interrompeu a conexão
54	16#36	Conexão resetada pelo parceiro
55	16#37	Não há memória de buffer disponível
56	16#38	Soquete já está conectado
57	16#39	Soquete não está conectado
58	16#3A	Soquete está fechado
60	16#3C	Tempo para operação esgotou
61	16#3D	Conexão recusada (pelo parceiro)
65	16#41	Não há entrada de roteamento ao parceiro
78	16#4E	Função não existe
254	16#FE	Ocorreu um Timeout
255	16#FF	Conexão fechada pelo parceiro

Tabela 261: Códigos de erro da conexão TCP

#### 8.8.4 Códigos de erro adicionais dos blocos funcionais

Os códigos de erro dos blocos funcionais apenas são emitidos em A\_Status dos blocos funcionais S&R TCP.

Código de erro decimal	Código de erro hexadecimal	Descrição
129	16#81	Não existe nenhuma conexão com este identificador
130	16#82	Comprimento é zero ou grande demais
131	16#83	Nesta conexão apenas dados cíclicos são permitidos
132	16#84	Estado incorreto
133	16#85	Valor do timeout é excessivo
134	16#86	Erro interno de programa
135	16#87	Erro de configuração
136	16#88	Dados transmitidos não cabem na área de dados
137	16#89	Bloco funcional parado
138	16#8A	Ocorreu um Timeout ou envio bloqueado
139	16#8B	Um bloco funcional deste tipo já está ativo nesta conexão

Tabela 262: Códigos de erro adicionais

#### 8.8.5 Estado da conexão

Código de erro hexadecimal	Descrição
16#00	Conexão OK
16#01	Conexão encerrada
16#02	Servidor aguarda incício de conexão
16#04	Client tenta estabelecer a conexão
16#08	Conexão está bloqueada

Tabela 263: Estado da conexão

#### 8.8.6 Estado da conexão parceiro

Estado do protocolo decimal	Descrição
0	Sem conexão
1	Conexão OK

Tabela 264: Estado da conexão parceiro

## 9 SNTP Protocol

(Simple Network Time Protocol)

Com o protocolo SNTP, o horário dos clientes SNTP é sincronizado através do servidor SNTP via Ethernet.

Os sistemas de comando HIMax/HIMatrix podem ser configurados e utilizados como **SNTP Server** e/ou **SNTP Client**. Vale o padrão SNTP conforme RFC 2030 (SNTP Versão 4) com a restrição que apenas o modo Unicast é suportado.

### Equipamentos necessários e requisitos de sistema:

Elemento	Descrição
Sistema de comando	HIMax com módulo COM ou apenas módulo CPU HIMatrix a partir de CPU OS V7 e COM OS V12
Ativação	Esta função está liberada por padrão em todos os sistemas HIMax/HIMatrix.
Interface	Ethernet 10/100/1000BaseT

Tabela 265: Equipamentos necessários e requisitos de sistema S&R TCP

### 9.1 SNTP Client

O cliente SNTP usa para a sua sincronização de tempo sempre apenas o servidor SNTP com a prioridade máxima.

Em cada recurso pode ser configurado um cliente SNTP para a sincronização do tempo.

·  
1

Sincronização de tempo de um Remote I/O através de um sistema de comando HIMax. Se um cliente SNTP é configurado num sistema de comando HIMax, o servidor SNTP interno do sistema de comando HIMax é desligado.

Para garantir ainda assim a sincronização de tempo do Remote I/O pelo sistema de comando HIMax, deve ser configurado um servidor SNTP no módulo de comunicação HIMax ao qual o Remote I/O está conectado.

#### Assim cria-se um novo cliente SNTP:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Clicar com o botão direito em **Protocols** e seleccionar no menu de contexto **New, SNTP Client**.  
☒ Um novo cliente SNTP é adicionado.
3. No menu de contexto do SNTP Client **Properties**, seleccionar o **COM-Module**.

A janela de diálogo do cliente SNTP contém os seguintes parâmetros.

Elemento	Descrição				
Type	Cliente SNTP				
Name	Nome para o cliente SNTP, no máximo 32 caracteres.				
Module	Seleção do módulo CPU COM no qual este protocolo é processado.				
Behavior on CPU/COM Connection Loss	<p>No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. (P. ex., se o módulo de comunicação é puxado para fora durante a comunicação).</p> <table> <tr> <td>Adopt Initial Data</td><td>As variáveis de entrada são resetadas para os valores iniciais.</td></tr> <tr> <td>Retain Last Value</td><td>As variáveis de entrada mantêm o último valor.</td></tr> </table>	Adopt Initial Data	As variáveis de entrada são resetadas para os valores iniciais.	Retain Last Value	As variáveis de entrada mantêm o último valor.
Adopt Initial Data	As variáveis de entrada são resetadas para os valores iniciais.				
Retain Last Value	As variáveis de entrada mantêm o último valor.				
Activate Max. $\mu$ P Budget	<p>Não é considerado pelo sistema operacional do módulo.</p> <p>O parâmetro foi mantido por causa da estabilidade de CRC e Reload.</p>				
Max. $\mu$ P Budget in [%]	<p>Não é considerado pelo sistema operacional do módulo.</p> <p>O parâmetro foi mantido por causa da estabilidade de CRC e Reload.</p>				
Description	Descrição livre inequívoca para o SNTP				
Current SNTP Version	Exibição da versão atual do SNTP				
Reference Stratum	<p>O Stratum de um cliente SNTP reproduz a precisão da sua hora local. Quanto menor o stratum, mais precisão tem a sua hora local. Zero significa um Stratum não especificado ou não disponível (não válido). O servidor SNTP atualmente usado de um cliente SNTP é aquele que é acessível e possui a maior prioridade.</p> <p>Se o Stratum do servidor SNTP atual for menor do que o do cliente SNTP, então, o recurso assume a hora do servidor SNTP atual.</p> <p>Se o Stratum do servidor SNTP atual for maior do que o do cliente SNTP, então, o recurso não assume a hora do servidor SNTP atual.</p> <p>Se o Stratum do servidor SNTP atual for igual ao do cliente SNTP, então, devem ser diferenciados dois casos:</p> <ul style="list-style-type: none"> <li>Se o cliente SNTP (recurso) trabalhar exclusivamente como cliente SNTP, então, o recurso assume a hora do servidor SNTP atual.</li> <li>Se o cliente SNTP (recurso) trabalhar simultaneamente como servidor SNTP também, para cada requisição do cliente SNTP é assumida a metade da diferença de hora ao servidor SNTP atual no recurso (a hora se aproxima aos poucos).</li> </ul> <p>Faixa de valores: 1...16 Valor padrão: 15</p>				
Client Time Request Interval [s]	<p>Intervalo de tempo no qual a sincronização da hora pelo servidor SNTP atual ocorre.</p> <p>O intervalo de requisição de hora do cliente no cliente SNTP deve ser maior do que o timeout no servidor SNTP.</p> <p>Faixa de valores: 16 s...16384 s Valor padrão: 16</p>				

Tabela 266: Características do cliente SNTP

## 9.2 SNTP Client (Server Info)

Nas informações do servidor SNTP é configurada a conexão a um servidor SNTP.

Abaixo de um cliente SNTP podem ser configuradas 1 a 4 informações de servidor SNTP.

### Assim cria-se um novo SNTP Server Info:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, SNTP Client**.
2. Clicar com o botão direito em **Protocols** e selecionar no menu de contexto **New, SNTP Server Info**.  
☒ Um novo SNTP Server Info é adicionado.
3. No menu de contexto do SNTP Sever Info **Properties**, selecionar o **COM-Module**.

A janela de diálogo SNTP Server Info contém os seguintes parâmetros.

Elemento	Descrição
Type	SNTP Server Info
Name	Nome para o SNTP Server Info. No máximo 31 caracteres.
Descrição	Descrição do servidor SNTP. No máximo 31 caracteres.
IP Address	Endereço IP do recurso ou do PC onde o servidor SNTP está configurado. Valor padrão: 0.0.0.0
SNTP Server Priority	Prioridade com a qual o cliente SNTP trata este servidor SNTP. Os servidores SNTP configurados para um cliente SNTP deveriam ter diferente prioridade. Faixa de valores: 0 (menor prioridade) até 4294967295 (prioridade máxima.) Valor padrão: 1
SNTP Server Timeout [s]	O timeout no servidor SNTP deve ser ajustado menor do que o <i>Time Request Interval</i> no cliente SNTP. Faixa de valores: 1 s...16384 s Valor padrão: 1

Tabela 267: Características SNTP Server Info

## 9.3 SNTP Server

O servidor SNTP recebe a requisição de um cliente SNTP e envia a sua hora atual de volta ao cliente SNTP.

### Assim cria-se um novo servidor SNTP:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Clicar com o botão direito em **Protocols** e selecionar no menu de contexto **New, SNTP Server**.  
☒ Um novo servidor SNTP é adicionado.
3. No menu de contexto do SNTP Server **Properties**, selecionar o **COM-Module**.



A janela de diálogo do servidor SNTP contém os seguintes parâmetros.

Elemento	Descrição
Type	SNTP Server
Name	Nome para o servidor SNTP, no máximo 31 caracteres.
Module	Seleção do módulo CPU COM no qual este protocolo é processado.
Activate Max. $\mu$ P Budget	Ativado: Transferir o limite do $\mu$ P Budget do campo Max. $\mu$ P-Budget in [%].  Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P Budget in [%]	Carga máxima $\mu$ P do módulo que pode ser produzida ao processar o protocolo.  Faixa de valores: 1...100% Valor padrão: 30%
Behavior on CPU/COM Connection Loss	No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. (P. ex., se o módulo de comunicação é puxado para fora durante a comunicação). Adopt Initial Data      As variáveis de entrada são resetadas para os valores iniciais. Retain Last Value      As variáveis de entrada mantêm o último valor.
Description	Descrição do SNTP
Current SNTP Version	Exibição da versão atual do SNTP
Stratum of Timeserver	O Stratum de um servidor SNTP reproduz a precisão da sua hora local. Quanto menor o stratum, mais precisão tem a sua hora local. Zero significa um Stratum não especificado ou não disponível (não válido). O Stratum do servidor SNTP deve ser menor ou igual ao Stratum do cliente SNTP que efetua a requisição. Caso contrário, a hora do servidor SNTP não é aplicada pelo cliente SNTP.  Faixa de valores: 1...15 Valor padrão: 14

Tabela 268: Características servidor SNTP

## 10 X-OPC Server

O HIMA X-OPC Server serve como interface de transmissão entre sistemas de comando HIMax/HIMatrix e sistemas de outros fabricantes que disponham de uma interface OPC.

OPC significa Openess, Productivity & Collaboration e é baseado na tecnologia desenvolvida pela Microsoft (COM/DCOM). Com ela podem ser conectados o sistema de gestão de processos, sistemas de visualização e sistemas de comando dos mais variados fornecedores para a troca de dados, veja também [www.opcfoundation.org](http://www.opcfoundation.org).

O servidor OPC da HIMA é executado num PC como serviço Windows de pois de ser instalado.

- 
- i** A configuração inteira e a operação do servidor X-OPC é efetuada no SILworX. No Control Panel do SILworX é possível carregar, iniciar e parar o servidor X-OPC como um sistema de comando.
- 

O servidor X-OPC suporta as seguintes especificações:

- **Data Access (DA) versões 1.0, 2.05a e 3.0**

DA é usado para a transmissão de dados de processo do sistema de comando HIMax/HIMatrix ao cliente OPC.

Cada variável global do sistema de comando HIMax/HIMatrix pode ser transmitida ao cliente OPC.

- **Alarm & Event (A&E) Versão 1.10**

A&E é usado para a transmissão de alarmes e eventos do sistema de comando HIMax/HIMatrix ao cliente OPC. Cada variável global do sistema de comando HIMax/HIMatrix pode ser monitorada com o registro de eventos.

Eventos são alterações do estado de uma variável pela instalação ou pelo sistema de comando que recebem um carimbo de hora.

Alarmes são eventos que assinalam um aumento do potencial de perigo.

Diferencia-se entre eventos booleanos e escalares, veja Capítulo 10.7.

## 10.1 Equipamentos necessários e requisitos de sistema

Elemento	Descrição
Ativação	<p>A liberação ocorre mediante código de liberação do software, veja Capítulo 3.4. As seguintes licenças podem ser ativadas individualmente:</p> <ul style="list-style-type: none"> <li>▪ Data Access (DA) Server</li> <li>▪ Alarm and Events (A&amp;E) Server</li> </ul>
Sistema operacional do PC	<p>O servidor X-OPC pode ser executado num PC x86 com os seguintes sistemas operacionais:</p> <ul style="list-style-type: none"> <li>▪ Windows XP Professional (mín. Service Pack 2) (32Bit)</li> <li>▪ Windows Server 2003 (32Bit)</li> <li>▪ Windows Vista Ultimate (32Bit)</li> <li>▪ Windows Vista Business (32Bit)</li> </ul>
Requisitos ao Host PC	<p>Requisitos mínimos ao Host PC:</p> <ul style="list-style-type: none"> <li>▪ Pentium 4</li> <li>▪ 1 GByte (XP) ou 2,5 GByte RAM (Vista)</li> <li>▪ A placa de rede deve ser dimensionada de acordo com o volume de dados, 100Mbit/s ou 1Gbit/s</li> </ul> <hr/> <p><b>i</b> Os requisitos mínimos apenas valem para a operação do servidor X-OPC, se não forem executadas outras aplicações (p. ex., SILworX, Word, etc.) no Host PC.</p>

Tabela 269: Equipamentos necessários e requisitos de sistema do servidor X-OPC

## 10.2 Properties X-OPC Server

Elemento	Descrição
OPC Server	O servidor X-OPC suporta as seguintes funções <ul style="list-style-type: none"> <li>▪ OPC Data Access Custom Interface, nas versões 1.0, 2.05a e 3.0.</li> <li>▪ OPC Alarm &amp; Event Interfaces 1.10</li> </ul>
Direcionado à segurança	O servidor X-OPC roda num PC e não é direcionado à segurança.
Interface	Recomendado: Ethernet 1 GBit/s
Troca de dados	Troca de dados via <b>safeethernet</b> .
Rede Ethernet	A velocidade de rede da rede Ethernet base deve ser dimensionada de acordo com o volume de dados, (mín. 100 Mbit/s, recomendado 1 GBit/s).
Variáveis globais	Podem ser usadas variáveis globais do contexto da configuração.
Tipos de variáveis admissíveis	Todos os tipos de dados que podem ser criados no SILworX são admissíveis.
Caracteres ASCII não permitidos	Os seguintes caracteres são reservados e não podem ser usados (p. ex., para variáveis globais): ! " # ' , . / \ ` :
Sistemas de comando HIMax/HIMatrix	No máximo 255 sistemas de comando HIMax/HIMatrix podem ser suportados por um servidor X-OPC.
Conexão safeethernet	O servidor X-OPC pode trocar 128 kB em cada conexão <b>safeethernet</b> .
X-OPC Server	10 X-OPC Server podem ser operados em um Host PC.
X-OPC Clients	Um servidor X-OPC pode suportar 10 clientes OPC.
Data Access Tags	Um Data Access Server suporta no máximo 100 000 DA Tags. Definição: Tags: Dados disponibilizados pelo servidor X-OPC. Tags correspondem às variáveis globais definidas. Items: Dados requisitados pelo cliente OPC.
Alarm & Event Definições de eventos	Um X-OPC Alarm & Event Server suporta no máximo 100 000 definições de eventos.

Tabela 270: Características do servidor X-OPC

### 10.3 Características do sistema de comando HIMA para a conexão X-OPC

Elemento	HIMax	HIMatrix L3	HIMatrix L2	Descrição
Volume de dados de processo	128 kB	128 kB	16 kB	O volume de dados de processo que um sistema de comando HIMA pode trocar no máximo com um servidor X-OPC por conexão <b>safeethernet</b> .
Tamanho do fragmento	1100 Byte	1100 Bytes	900 Byte	Por ciclo HIMax, apenas um fragmento é enviado a um servidor X-OPC.
Interfaces Ethernet	10/100/1000BaseT	10/100BaseT	10/100BaseT	As interfaces Ethernet em uso podem ser usadas simultaneamente para outros protocolos.
Quantidade máx. de eventos de sistema (CPU Event)	20 000	4000	n. a.	Os eventos definidos como CPU Event são formados no módulo processador. Ele executa a formação de eventos por completo em cada um de seus ciclos. Assim, o valor de cada variável global pode ser determinado e avaliado como evento.
Quantidade máx. de eventos de E/S (I/O Event)	6000	n. a.	n. a.	Os eventos definidos como I/O Event apenas podem ser formados em módulos de E/S SOE (p. ex., AI 32 02 ou DI 32 04). Ele executa a formação de eventos por completo em cada um de seus ciclos.
Tamanho da memória de eventos	5000	1000 (só com licença SER liberada)	n. a.	Buffer de eventos não-volátil do módulo processador HIMax. Se o buffer de eventos estiver cheio, não são mais armazenados eventos novos, até uma entrada de evento foi lido pelo menos por um X-OPC A&E Server e, assim, liberado para sobrescrever.
Quantidade máx. X-OPC Server	4	4	4	A quantidade máxima de X-OPC Server que podem acessar o sistema de comando HIMA e ler eventos do buffer de eventos do módulo processador simultaneamente.
n. a.: não aplicável				

Tabela 271: Características do sistema de comando HIMA para a conexão X-OPC

Faixa de valores do carimbo de hora UTC (Universal Time Coordinated):  
 Porção em seg. desde 1970 como [udword]  
 proporção em ms como [udword] de 0–999  
 Valor padrão: 01.01.2000 / 00:00:00 Uhr  
 Comutação automática de horário de verão não é suportada.

## 10.4 Ações necessárias no caso de alterações

A seguinte tabela mostra as ações que devem ser efetuadas após uma alteração em sistemas individuais.

Tipo de alteração	Alterações em		
	HIMax	HIMatrix	X-OPC
<b>DA</b>			
Adicionar Tags	C+D	C+D	C+D
Nomes de Tags (alteração do nome de GV)	C+D	C+D	C+D
Excluir Tags	C+D	C+D	C+D
Alterar fragmentos (adicionar/excluir e parâmetros)	C+D	C+D	C+D
<b>A&amp;E</b>			
Adicionar Event Definition	C+D	n. a.	C+D
Excluir Event Definition	C+D	n. a.	C+D
Alterar Event Source	C+D	n. a.	C+D
Alterar Alarm Texts	-	n. a.	C+D
Alterar Alarm Severity	-	n. a.	C+D
Alterar parâmetro "ACK Required"	-	n. a.	C+D
Alterar "Alarm Values" para evento escalar	C+D	n. a.	C+D
Alterar parâmetro "Alarm at False" para eventos booleanos	C+D	n. a.	C+D
Alterar nome	C+D	n. a.	C+D
Conectar canal de de E/S com GV	C+R	n. a.	-
Conectar variáveis de estado com GV	C+R	n. a.	-
<b>Em geral</b>			
Alterar parâmetros safeethernet	C+D	C+D	C+D

Tabela 272: Ações necessárias no caso de alterações

- C: Código precisa ser gerado
- R: Reload necessário
- D: Download necessário
- n. a.: não aplicável
- : Nenhuma ação necessária

## 10.5 Forcing de variáveis globais em módulos de E/S

**i**

Se as variáveis globais conectadas ao valor de processo são forçadas por um módulo de E/S, as mesmas não têm efeito sobre as variáveis globais conectadas com o parâmetros -> **estado -LL, -L, -N, -H, -HH**.

Isso vale mesmo quando estes alarmes são introduzidos no Alarm & Event Editor. No caso de testes, estas variáveis globais devem ser forçadas individualmente.

## 10.6 Configuração de uma conexão OPC Server

Neste exemplo é configurada uma conexão redundante de servidor X-OPC com ajuda de um sistema de comando HIMax.

Os servidores X-OPC disponibilizam as variáveis de processo e valores de evento do sistema de comando HIMax aos clientes OPC. Os clientes OPC acessam as variáveis de processo e os valores de eventos disponibilizados e os representam na sua interface de usuário.

### 10.6.1 Software necessário:

- SILworX
- X-OPC Server
- OPC Client

**i**

A configuração inteira e a operação do servidor X-OPC é efetuada no SILworX. No Control Panel do SILworX é possível carregar, iniciar e parar o servidor X-OPC como um sistema de comando.

### 10.6.2 Requisitos para a operação do servidor X-OPC:

- A rede Ethernet deve ter uma largura de banda de no mínimo 100 Mbit/s (melhor 1Gbit/s).
- A hora de sistema dos computadores/servidores deve ser sincronizada, p. ex., mediante SNTP.
- Garantir que os conjuntos de dados para Data Access e Alarm & Events no sistema de comando, nos servidores X-OPC e nos clientes OPC sejam compatíveis.

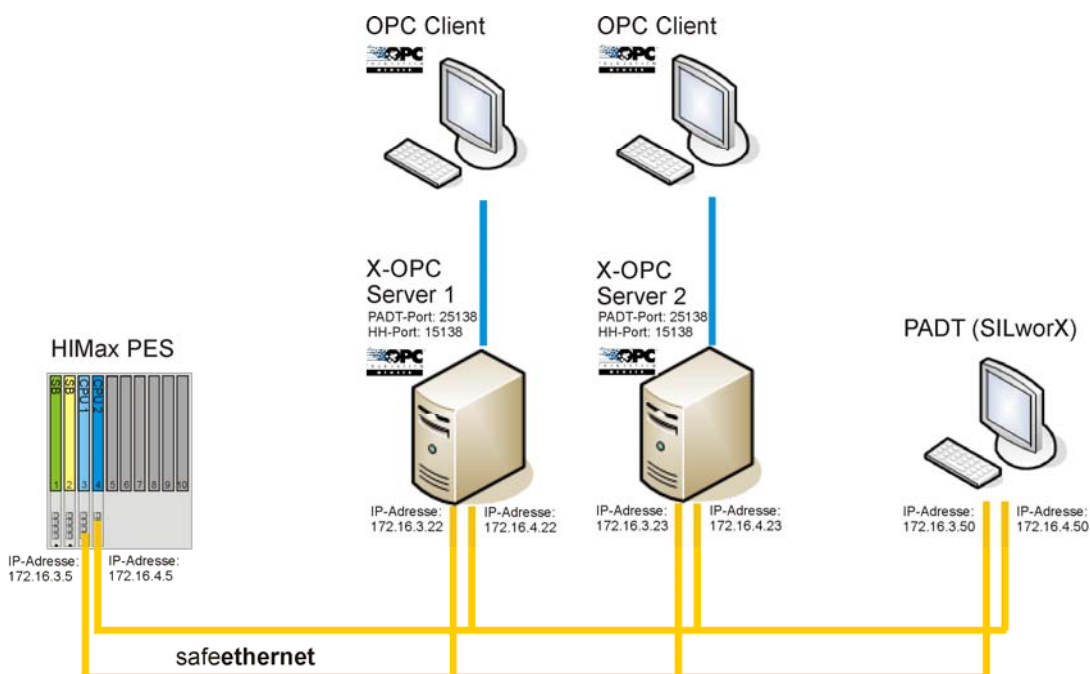


Figura 79: Operação redundante X-OPC

### 10.6.3 Instalação no Host PC

O servidor X-OPC deve ser instalado no respectivo Host PC.

i

Anotar o ID de sistema e o número da porta PADT. São necessários para gerar a chave de licença!

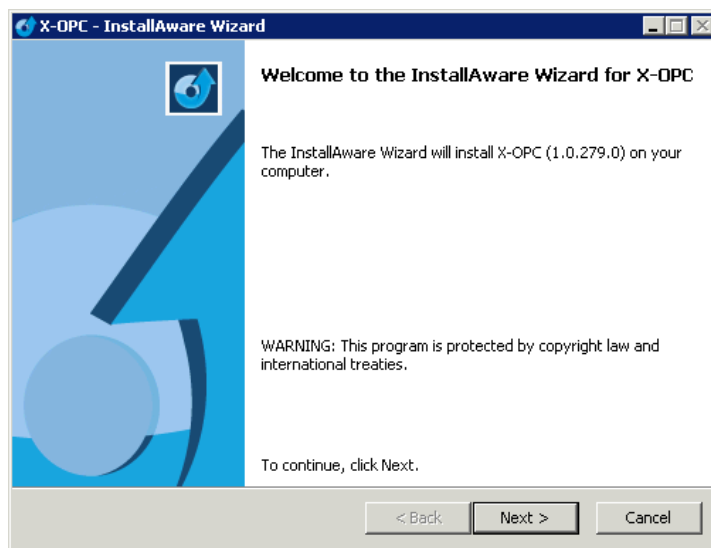


Figura 80: Rotina de instalação do servidor X-OPC

#### Assim instala-se o servidor X-OPC no primeiro Host PC:

Iniciar o arquivo *X-OPC.exe* no respectivo Host PC e seguir a rotina de instalação.

1. Introduzir os seguintes dados para o servidor X-OPC:
  - System ID: 100
  - PADT Port: 25138
  - Nome livre do servidor X-OPC (é exibido no cliente OPC).
2. Para instalar o servidor X-OPC, clicar em **Next >**.

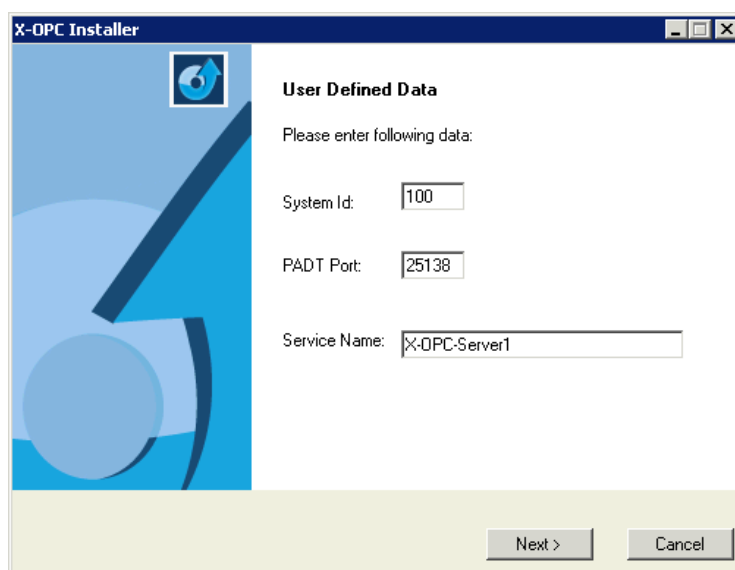


Figura 81: Rotina de instalação do servidor X-OPC



**Assim instala-se o servidor X-OPC no segundo Host PC:****i**

Primeiramente determinar o ClassID do primeiro servidor X-OPC, antes de instalar o segundo servidor X-OPC!

No caso da conexão redundante de um cliente OPC com dois servidores X-OPC, alguns sistemas de cliente OPC esperam que o ClassID dos dois servidores X-OPC seja igual. Determinar primeiramente o ClassID do primeiro servidor X-OPC (p. ex., com ajuda do cliente OPC) e anotar.

Iniciar o arquivo X-OPC.exe no segundo Host PC e seguir a rotina de instalação.

1. Introduzir os seguintes dados para o servidor X-OPC:

- System ID: 110
- PADT Port: 25138
- Nome livre do servidor X-OPC (é exibido no cliente OPC).

**i**

A porta PADT e a porta HH do segundo servidor X-OPC podem ser iguais ao primeiro se os servidores X-OPC são operados em PCs diferentes.

2. Clicar **Continue >** para confirmar.

**Assim ajusta-se o ClassID igual no segundo Host PC:**

1. Selecionar ajuste CLSID **manual** para DA und AE.
2. Introduzir o ClassID do primeiro servidor X-OPC nos campos **CLSID**.
3. Para instalar o servidor X-OPC, clicar em **Next >**.

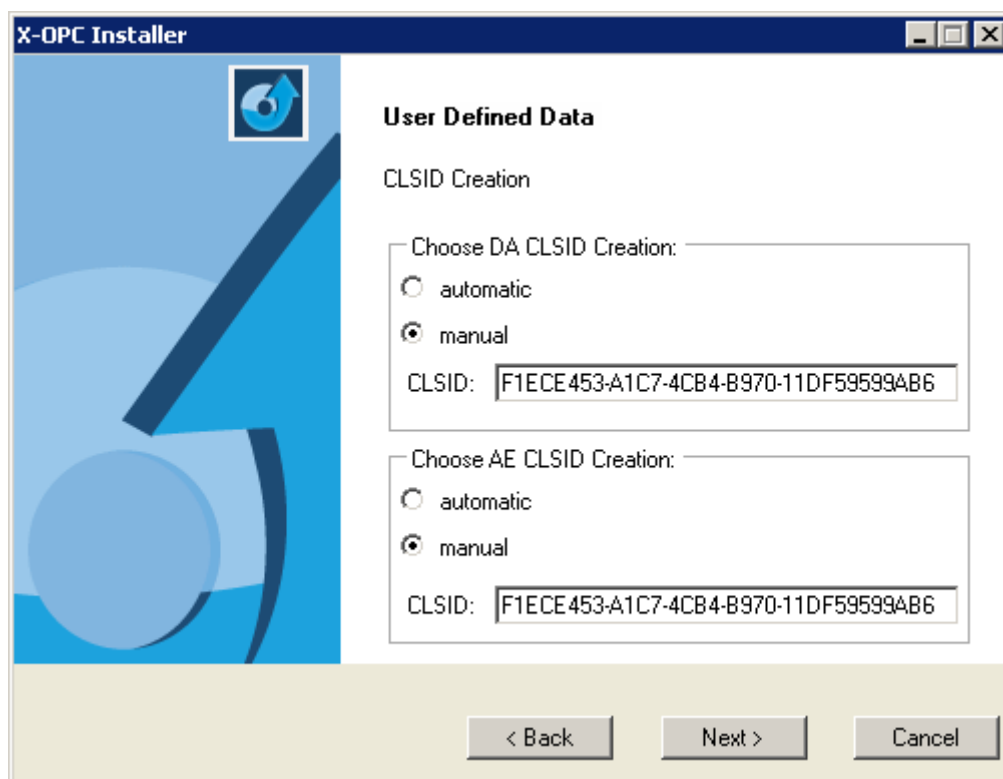


Figura 82: Ajuste manual do ClassID do segundo servidor X-OPC

**Iniciar automaticamente os servidores X-OPC após reinicialização do PC:**

1. Abrir no Windows **Start, Settings, Control Panel, Administration, Services** e seleccionar **X-OPC Server** na lista.
2. No menu de contexto do servidor X-OPC, seleccionar **Properties**.
3. No registo **General**, seleccionar o tipo de iniciar **Automatic**.

**Assim verifica-se se o servidor X-OPC está rodando no PC:**

1. Abrir o *Windows Task-Manager* e seleccionar o registo *Processes*.
2. Verificar se o processo *X-OPC.exe* está rodando no PC.

---

**i**

Se o cliente OPC e o servidor OPC não estão rodando no mesmo PC, precisa adaptar a interface DCOM.

Para este fim devem ser executados os passos descritos no manual da OPC Foundation *Using OPC via DCOM with Microsoft Windows XP Service Pack 2 Version 1.10* (veja [www.opcfoundation.org](http://www.opcfoundation.org)).

---

#### 10.6.4 Configurar o servidor OPC no SILworX

**Assim cria-se um novo servidor OPC no SILworX:**

1. Abrir na árvore de estrutura **Configuration**.
2. Clicar com o botão direito em **Configuration** e selecionar no menu de contexto **New, OPC Server-Set**.
  - ☒ Um novo OPC Server-Set é adicionado.
3. Selecionar no menu de contexto do OPC Server-Set **Properties** e aplicar valores padrão.

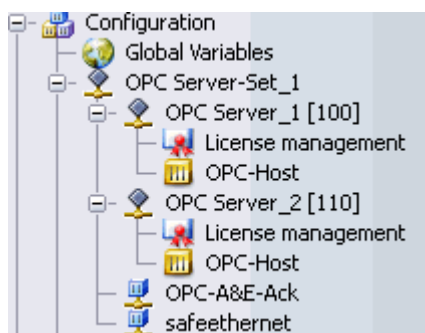


Figura 83: Operação redundante X-OPC

**Assim configura-se o primeiro OPC Server no SILworX:**

1. Selecionar na árvore de estrutura **Configuration, OPC Server-Set**.
2. Clicar com o botão direito em **OPC Server-Set** e selecionar no menu de contexto **New, OPC Server**.
  - ☒ Um novo OPC Server é adicionado.
3. Clicar com o botão direito em **OPC Server** e selecionar no menu de contexto **Properties**.
  - Introduzir System ID [SRS] (p. ex., 100).
  - Aplicar os ajustes padrão.
3. Clicar com o botão direito em **OPC Host** e selecionar no menu de contexto **Edit**.
  - ☒ O OPC Host Dialog para a configuração das interfaces IP se abre.
4. Clicar com o botão direito em um lugar vazio do diálogo OPC Host e selecionar no menu de contexto **New IP Device**.
  - Ajustar a porta PADT (p. ex., 25138).
  - Endereço IP do PC onde o X-OPC Server está instalado (p. ex., 172.16.3.22).
  - Endereço IP do PC onde o X-OPC Server está instalado (p. ex., 172.16.4.22).
  - Marcar como interface padrão.
  - Ajustar a porta HH (p. ex., 15138).

No mesmo OPC Server-Set configura-se o OPC Server redundante.

**Assim configura-se o segundo OPC Server:**

1. Selecionar na árvore de estrutura **Configuration, OPC Server-Set**.
2. Clicar com o botão direito em **OPC Server-Set** e selecionar no menu de contexto **New, OPC Server**.
  - ☒ Um novo OPC Server é adicionado.
3. Clicar com o botão direito em **OPC Server** e selecionar no menu de contexto **Properties**.
  - Introduzir System ID [SRS] (p. ex., 110).
  - Aplicar os ajustes padrão.
5. Clicar com o botão direito em **OPC Host** e selecionar no menu de contexto **Edit**.
  - ☒ O OPC Host Dialog para a configuração das interfaces IP se abre.
6. Clicar com o botão direito em um lugar vazio do diálogo OPC Host e selecionar no menu de contexto **New IP Device**.
  - Ajustar a porta PADT (p. ex., 25138).
  - Endereço IP do PC onde o X-OPC Server está instalado (p. ex., 172.16.3.23).
  - Endereço IP do PC onde o X-OPC Server está instalado (p. ex., 172.16.4.23).
  - Marcar como interface padrão.
  - Ajustar a porta HH (p. ex., 15138).

**i**

Se no PC estiver instalado um Firewall, as portas TCP/UDP PADT e HH do servidor X-OPC devem ser introduzidas como exceções na configuração do Firewall.

## 10.6.5 Ajustes do OPC Server no Editor safeethernet

**Assim cria-se a conexão safeethernet entre o OPC Server e o recurso (sistema de comando HIMax):**

1. Abrir no OPC Server-Set o **safeethernet Editor**
2. Na seleção de objetos, clicar no **Resource** e puxar mediante Drag&Drop para um local livre na área de trabalho do **safeethernet Editor**.
3. Para Alarm & Events, ativar o parâmetro *Activate SOE* por padrão.

	IF CH 1 (local)	IF CH 2 (local)	IF CH 1 (target)	IF CH 2 (target)	Profile	Sync/Async
1					Fast & Noisy	ASYN
2	110.x.x (122.16.4.23:15138)	110.x.x (122.16.4.23:15138)	2.0.15 (172.16.4.5:6010)	2.0.15 (172.16.4.5:6010)		
3	100.x.x (172.16.3.22:15138)	100.x.x (172.16.3.22:15138)	2.0.5 (172.16.3.5:6010)	2.0.5 (172.16.3.5:6010)		

Figura 84: Operação redundante X-OPC

**i**

As interfaces Ethernet dos PCs são representadas na coluna **IF CH1 (local)**. As interfaces Ethernet do sistema de comando HIMax devem ser selecionadas na coluna **IF CH1 (target)**.

Os valores padrão dos parâmetros **safeethernet** para a comunicação servidor X-OPC são ajustados para máxima disponibilidade.

Receive Timeout = 1000 ms, Response Time = 500 ms etc.

Informações sobre os parâmetros de **safeethernet**, veja Capítulo 4.6.

### 10.6.6 Configurar servidor OPC Data Access no SILworX

**Assim criam-se as definições de fragmentos da conexão safeethernet:**

Requisito: O Editor safeethernet do OPC Server deve estar aberto.

1. Clicar com o botão direito na linha do **Resource** para abrir o menu de contexto do recurso.
2. Selecionar no menu de contexto **Detail View** para abrir a visualização de detalhes da conexão safeethernet.
3. Clicar no registro **Fragment Definitions**.
4. Clicar com o botão direito em sobre um lugar livre na área de trabalho e selecionar **New Fragment Definition**.  
Na coluna Priority é ajustado quantas vezes este fragmento é enviado em relação ao outros fragmentos (Um fragmento possui o tamanho de 1100 Byte).  
Usar para as definições de fragmentos primeiramente o ajuste padrão com prioridade 1, veja também Capítulo 10.6.8.
5. Clicar no registro **OPC Server Set <-> Resource**.

**Assim se adicionam as variáveis de recepção OPC:**

As variáveis de recepção OPC são enviadas do recurso ao OPC Server.

1. Abrir a visualização de detalhes do Editor safeethernet X-OPC e selecionar o registro **OPC Server-Set <-> Resource**.
2. Na seleção de objetos, selecionar uma **Global Variable** e puxar via Drag&Drop para a área **OPC Server Set <- Resource**.
3. Clique duplo na coluna **Fragment Name** e selecionar a **Fragment Definition** anteriormente criada.
4. Repetir este passo para outras variáveis de recepção OPC.

**Assim se adicionam as variáveis de envio OPC:**

As variáveis de envio OPC são enviadas do OPC Server ao recurso

1. Abrir a visualização de detalhes do Editor safeethernet X-OPC e selecionar o registro **OPC Server-Set <-> Resource**.
2. Na seleção de objetos, selecionar uma **Global Variable** e puxar via Drag&Drop para a área **OPC Server Set -> Resource**.
3. Clique duplo na coluna **Fragment Name** e selecionar a **Fragment Definition** anteriormente criada.
4. Repetir este passo para outras variáveis de envio OPC.

---

**i**

As variáveis de envio e recepção OPC apenas precisam ser criadas uma vez no OPC-Server-Set. As mesmas são usadas automaticamente no OPC Server-Set pelos dois servidores X-OPC.

---

**Gerar o código e carregar o recurso:**

1. Selecionar na árvore de estrutura **Configuration, Resource**.
2. Clicar em **Code Generation** na **Action Bar** e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.
4. Carregar o código gerado para o recurso.

**Gerar o código e verificar o OPC Server-Set:**

1. Selecionar na árvore de estrutura **Configuration, OPC Server-Set**.
2. Clicar em **Code Generation** na **Action Bar** e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.

**Carregar o código gerado para o servidor X-OPC:**

1. Clicar com o botão direito em **OPC Server** e selecionar no menu de contexto **Online** para o **System Login**.
2. Introduzir os dados de acesso:
  - Endereço IP do PC onde o X-OPC Server está instalado (p. ex., 172.16.3.23).
  - Nome de usuário: Administrator
  - Senha: sem
  - Direitos: Administrator
3. Clicar em **Login** para abrir o Control Panel.
4. Na barra de menu do SILworX, clicar no sistema **Resource Download**.
  - ☒ O código é carregado no servidor X-OPC.
5. Na barra de menu do SILworX, clicar no sistema **Resource Cold Start**.
  - ☒ O servidor X-OPC está rodando.

**Abrir o cliente OPC:**

O nome do servidor X-OPC exibido no cliente OPC é composto por:  
**HIMA** (Manufacturer).**Service name** (veja Capítulo 10.6.3)-**DA** (Data Access).

Estabelecer a conexão ao servidor X-OPC. Os dados Data Access configurados devem ser transmitidos agora ao cliente OPC.

### 10.6.7 Configurar o X-OPC Alarm & Event Server no SILworX

Neste exemplo é configurado um X-OPC A&E Server para um sistema de comando HIMax. O X-OPC A&E Server detecta os eventos via **safeethernet** do sistema de comando HIMax e os disponibiliza ao cliente OPC. O cliente OPC acessa as variáveis de evento disponibilizados e os representa na sua interface de usuário.

#### Assim cria-se um Alarm & Event Editor:

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **Resource** e selecionar no menu de contexto **New, Alarm & Events**.
  - ☒ O Alarm & Event Editor novo é adicionado.

#### Assim criam-se os Alarm & Events:

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **Alarm & Events** e selecionar **Edit**.
3. Selecionar o registro **Event Definition Bool** para eventos booleanos, veja Capítulo 10.7.1.
4. Selecionar o registro **Event Definition Scalar** para eventos escalares, veja Capítulo 10.7.2.
5. Na seleção de objetos, clicar na **Global Variable** e puxar mediante Drag&Drop para um local livre na área de trabalho do Alarm & Event Editor.
6. A prioridade dos eventos é introduzida no **safeethernet** Editor, veja Capítulo 10.6.8.

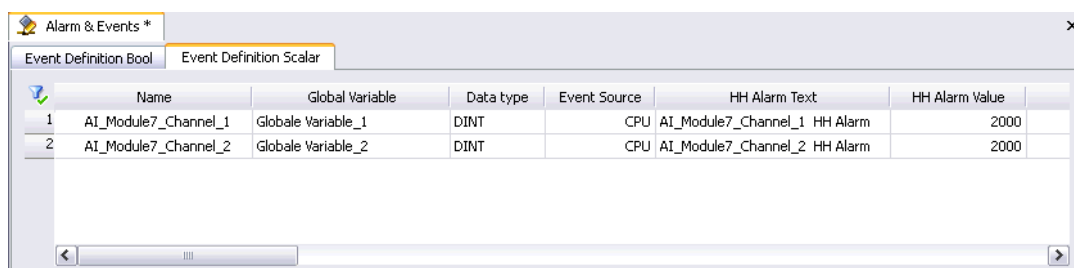


Figura 85: Alarm & Event Editor

### Assim cria-se a conexão Acknowledge entre os dois Alarm & Event X-OPC Servers:

i

Se dois Alarm & Event X-OPC Server redundantes são operados, é possível sincronizar os Acknowledgements (confirmação) dos alarmes nos dois servidores X-OPC. Para isso, cria-se uma conexão Acknowledge.

1. Selecionar na árvore de estrutura **Configuration, OPC Server-Set, New**.
2. Clicar com o botão direito em **OPC Server-Set** e selecionar no menu de contexto **New, OPC A&E Ack**.
3. Selecionar no diálogo OPC A&E-Ack as seguintes conexões IP.
  - IF CH1 (OPC Server 1, p. ex., 172.16.3.22).
  - IF CH2 (OPC Server 1, p. ex., 172.16.4.22).
  - IF CH1 (OPC Server 2, p. ex., 172.16.3.23).
  - IF CH2 (OPC Server 2, p. ex., 172.16.4.23).

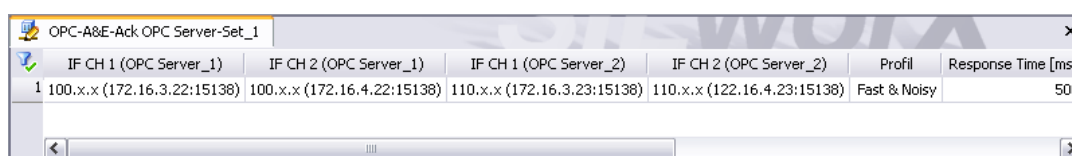


Figura 86: Operação redundante X-OPC

### Gerar o código e carregar o recurso:

1. Selecionar na árvore de estrutura **Configuration, Resource**.
2. Clicar em **Code Generation** na **Action Bar** e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.
4. Carregar o código gerado para o recurso.

### Gerar o código e verificar o OPC Server-Set:

1. Selecionar na árvore de estrutura **Configuration, OPC Server-Set**.
2. Clicar em **Code Generation** na **Action Bar** e confirmar com **OK**.
3. Verificar cuidadosamente as entradas no log, corrigir se for necessário.

### Carregar o código gerado para o servidor X-OPC:

1. Clicar com o botão direito em **OPC Server** e selecionar no menu de contexto **Online** para o System Login.
2. Introduzir os dados de acesso:
  - Endereço IP do PC onde o X-OPC Server está instalado (p. ex., 172.16.3.23).
  - Nome de usuário: Administrator
  - Senha: sem
  - Direitos: Administrator
3. Clicar em **Login** para abrir o Control Panel.
4. Na barra de menu do SILworX, clicar no sistema **Resource Download**.
  - ☒ O código é carregado no servidor OPC.
5. Na barra de menu do SILworX, clicar no sistema **Resource Cold Start**.
  - ☒ O servidor OPC está rodando.



**Abrir o cliente OPC:**

O nome do servidor X-OPC exibido no cliente OPC é composto por:

**HIMA** (Manufacturer).**Service name** (veja Capítulo 10.6.3)-**AE (Alarm & Event)**.

Estabelecer a conexão ao servidor X-OPC. Os dados Alarms & Events configurados devem ser transmitidos agora ao cliente OPC.

---

**i**

Se um sistema de comando e um X-OPC A&E Server são conectados, primeiramente o X-OPC A&E Server precisa se sincronizar. Para este fim, o X-OPC A&E Server lê o estado atual de todas as variáveis que estão definidas como evento e transfere os alarmes ativos ao cliente OPC. No cliente OPC pode ser formada uma imagem sobre o estado do sistema de comando. Somente a partir deste momento os eventos são lidos.

Depois de sincronizar o servidor X-OPC para o sistema de comando, todos os eventos são atualizados no cliente OPC. Entradas de eventos com um carimbo de hora mais antigo são sobrescritos com os estados atuais lidos das variáveis de eventos.

---

### 10.6.8 Parametrização de fragmentos e prioridades no SILworX

Um sistema de comando HIMA, de acordo com o tipo, pode enviar 128 kB ou 16 kB por conexão **safeethernet** a um servidor X-OPC, porém, apenas um fragmento (1100 Byte ou 900 Byte) por ciclo da CPU HIMA. Para enviar mais dados via uma conexão **safeethernet**, os dados são fragmentados. Mediante o parâmetro prioridade destes fragmentos, é possível determinar com que frequência estes fragmentos devem ser atualizados.

#### i

Fragmentos com a prioridade **n** e fragmentos com a prioridade **m** são enviados na proporção **n** a **m** vezes.

Para o tempo de reação do sistema de comando ao servidor X-OPC deve ser considerado adicionalmente a quantidade de fragmentos e comandos (p. ex., Stop, Start) do SOE.

$T_R = t_1 + t_2 + t_3 + t_4$ ; apenas vale se a prioridade de todos os fragmentos para dados de estado for 1

$T_R$	Worst Case Reaction Time
$t_1$	Tempo de segurança do PES 1
$t_2$	<i>Number of Fragments * Receive TMO</i>
$t_3$	Tempo de segurança do servidor X-OPC
$t_4$	Retardo de tempo pela função SOE; depende do volume de eventos e inicialização da conexão

Para a direção inversa, o tempo de reação pode ser determinado com a mesma fórmula, apenas que aqui normalmente apenas um fragmento é considerado, pois o servidor X-OPC apenas transfere os dados escritos por clientes OPC.

Quantidade máxima de fragmentos: 1024

Tamanho máximo de um fragmento: 1100 Byte ou 900 Byte

Faixa de valores das prioridades: 1 (mais alta) até 65 535 (mais baixa)

Denominação	Valor padrão da prioridade
Priority of events (Alarm&Event)	1
Priority of state values (Alarm&Event)	10
Priority of the fragment definitions (Data Access)	1

Tabela 273: Valores padrão das prioridades

#### 10.6.8.1 Prioridade dos fragmentos para eventos (Alarm & Event)

Os eventos que foram criados no Alarm & Event Editor são automaticamente fragmentados e transmitidos em fragmentos.

A prioridade para os eventos é introduzida no safeethernet Editor, nas colunas **Priority of Events** e **Priority of State Values**, estas prioridades passam a valer, então, para todos os fragmentos de Alarm & Event desta conexão safeethernet.

##### Assim ajustam-se as prioridades para os fragmentos de Alarm & Event

1. Abrir no OPC Server-Set o safeethernet Editor

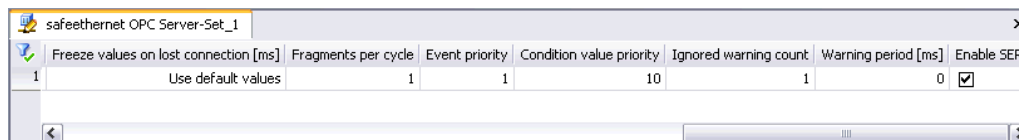


Figura 87: safeethernet Editor

2. Clique duplo em **Priority of Events**, para alterar a prioridade dos eventos.  
Os fragmentos dos eventos recebem todos a prioridade introduzida na coluna **Priority of Events** (p. ex., 1). Assim, define-se com que prioridade o X-OPC Server solicita eventos do sistema de comando. Se neste momento no sistema de comando não houver eventos, não serão transmitidos eventos.
3. Clique duplo em **Priority of State Values** para alterar a prioridade dos valores de estado dos eventos. Os fragmentos dos valores de estado dos eventos contêm todos a prioridade introduzida na coluna **Priority of State Values** (p. ex., 10).

---

**i**

Os valores de estado dos eventos apenas são necessários para a sincronização (p. ex., ao estabelecer uma conexão) e por isso podem ser transmitidos numa distância de tempo maior de tempo do que os eventos.

---

### 10.6.8.2 Prioridade dos fragmentos Data Access

Para cada fragmento Data Access, é possível atribuir variáveis globais e ajustar a prioridade do fragmento com a qual se determina com quanta frequência estas variáveis são atualizadas.

#### Assim ajustam-se as prioridades para os fragmentos de Data Access

Requisito: O Editor **safeethernet** do OPC Server deve estar aberto.

1. Clicar com o botão direito na linha do **Resource** para abrir o menu de contexto do recurso.
2. Selecionar no menu de contexto **Detail View** para abrir a visualização de detalhes da conexão **safeethernet**.
3. Clicar no registro **Fragment Definitions**.
4. Na coluna prioridade ajusta-se a prioridade dos fragmentos Data Access.
  - Atribuir aos Data Access Fragments com variáveis globais que devem ser atualizados frequentemente uma prioridade alta (p. ex., 1);
  - Atribuir aos Data Access Fragments com variáveis globais que devem ser atualizados mais raramente uma prioridade mais baixa (p. ex., 10);

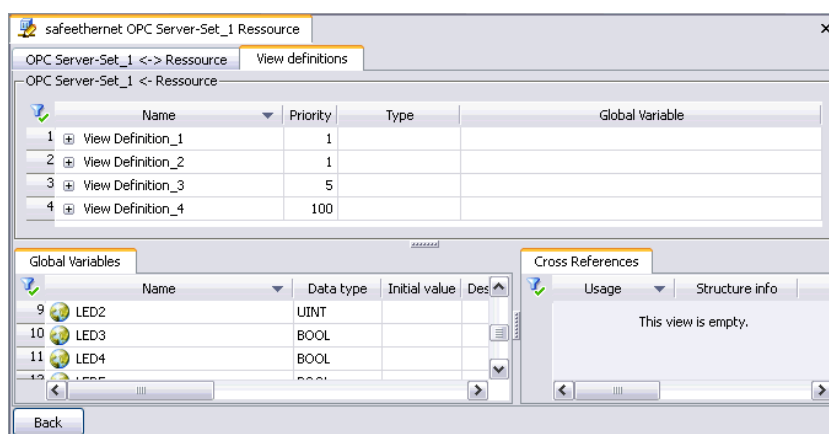


Figura 88: Visualização de detalhes da conexão **safeethernet**

Adicionalmente, o estado e o carimbo de hora de cada Data Access Fragment pode ser lido pelas seguintes variáveis.

Nome	Descrição	
Fragment timestamp [ms]	Fração de milissegundos do carimbo de hora (hora de sistema atual).	
Fragment timestamp [s]	Fração de segundos do carimbo de hora (hora de sistema atual).	
Fragment status	Status	Descrição
	0	CLOSED: Conexão está encerrada.
	1	TRY OPEN: Tentativa de abrir a conexão, porém, ainda não está aberta.
	2	CONNECTED: A conexão foi estabelecida e os dados de fragmento atuais foram recebidos (v. carimbo de hora). Enquanto não são recebidos dados de fragmento, o estado do fragmento ao estabelecer a conexão permanece em TRY_OPEN.
<p><b>i</b> O estado da conexão do <b>safeethernet</b> Editor (veja Capítulo 4.4) é colocado em CONNECTED logo que a conexão estiver aberta. Ao contrário do estado de fragmento, neste caso ainda não precisam ter sido trocados dados.</p>		

Tabela 274: Estado e carimbo de hora dos fragmentos Data Access

## 10.7 Alarm & Event Editor

O Alarm & Event Editor serve para a parametrização dos alarmes e eventos do sistema de comando HIMax.

### Assim cria-se um Alarm & Event Editor:

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **Resource** e selecionar no menu de contexto **New, Alarm & Events**.  
☒ Um novo objeto Alarm & Events é adicionado.

### Assim criam-se os Alarm & Events:

1. Abrir na árvore de estrutura **Configuration, Resource**.
2. Clicar com o botão direito em **Alarm & Events** e selecionar **Edit**.
3. Selecionar o registro **Event Definition Bool** para eventos booleanos, veja Capítulo 10.7.1.
4. Selecionar o registro **Event Definition Scalar** para eventos escalares, veja Capítulo 10.7.2.
5. Na seleção de objetos, clicar na **Global Variable** e puxar mediante Drag&Drop para um local livre na área de trabalho do Alarm & Event Editor.
6. A prioridade dos eventos é introduzida no **safeethernet** Editor, veja Capítulo 10.6.8.

O HIMax diferencia entre eventos booleanos e escalares.

### 10.7.1 Eventos booleanos

- Alterações de variáveis booleanas, p. ex., entradas digitais.
- Estado de alarme e estado normal, estes podem ser atribuídos aos estados de variáveis de forma livre.

Os parâmetros dos eventos do tipo **Boolean** são introduzidos no Alarm & Event Editor do recurso que contém as seguintes colunas:

Coluna	Descrição	Faixa de valores
Name	Nome da definição de evento	Texto, máx. 31 caracteres
Global Variable	Nome da variável global atribuída (p. ex., inserido por Drag&Drop)	
Data Type	Tipo de dados da variável global, não pode ser alterado	BOOL
Event Source	<p><b>CPU Event</b> O carimbo de hora é formado num módulo processador. Ele executa a formação de eventos por completo em cada um de seus ciclos.</p> <p><b>IO Event</b> O carimbo de hora é formado num módulo de E/S adequado (p. ex., DI 32 04).</p> <p><b>Auto Event</b> É formado um CPU Event e, se presente, IO Events dos módulos de E/S.</p> <p>Valor padrão: CPU Event</p>	CPU Event, IO Event, Auto Event
Alarm when FALSE	<p><b>Ativado</b> A alteração do valor TRUE -&gt; FALSE da variável global dispara um evento</p> <p><b>Desativado</b> A alteração do valor FALSE -&gt; TRUE da variável global dispara um evento</p> <p>Valor padrão: Desativado</p>	Caixinha de controle ativada, desativada
Alarm Text	Texto que identifica o estado de alarme	Texto
Alarm Priority	Prioridade do estado de alarme Valor padrão: 1	1...1000

Coluna	Descrição	Faixa de valores
Alarm Acknowledgment Required	Ativado      Confirmação do estado de alarme pelo usuário é necessária (acknowledge) Desativado      Confirmação do estado de alarme pelo usuário não é necessária Valor padrão: Desativado	Caixinha de controle ativada, desativada
Return to Normal Text	Texto que identifica o estado de alarme	Texto
Return to Normal Severity	Prioridade do estado normal	1...1000
Return to Normal Ack Required	Confirmação do estado normal pelo usuário é necessária (acknowledge) Valor padrão: Desativado	Caixinha de controle ativada, desativada

Tabela 275: Parâmetros para eventos booleanos

### 10.7.2 Eventos escalares

- Ultrapassagem de valores limite definidos para uma variável escalar, p. ex., uma entrada analógica.
- Dois limites superiores e inferiores são possíveis.  
Para os valores limite deve valer:  
Limite máximo  $\geq$  limite superior  $\geq$  faixa normal  $\geq$  limite inferior  $\geq$  limite mínimo.  
Uma histerese terá efeito nos seguintes casos:
  - Ao ultrapassar um limite superior para baixo
  - Ao ultrapassar um limite inferior
 Mediante indicação de uma histerese, é possível evitar uma quantidade desnecessariamente elevada de eventos se a variável global oscilar muito em torno a um valor limite.

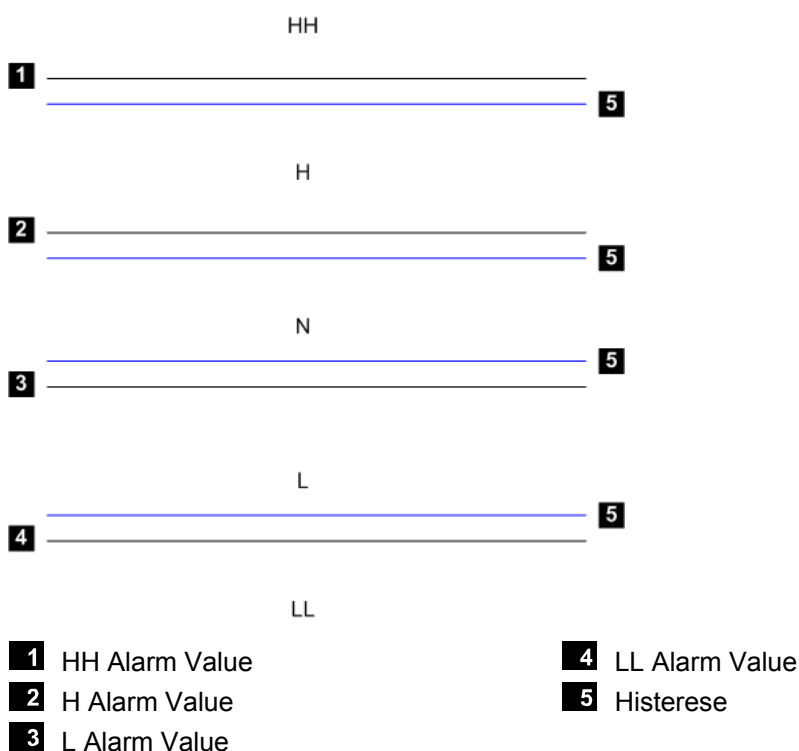


Figura 89: Cinco áreas de um evento escalar

Os parâmetros dos eventos do tipo **Scalar** são introduzidos no Alarm & Event Editor do recurso que contém as seguintes colunas:

Coluna	Descrição	Faixa de valores
Name	Nome da definição de evento	Texto, máx. 31 caracteres
Global Variable	Nome da variável global atribuída (p. ex., inserido por Drag&Drop)	
Data Type	Tipo de dados da variável global, não pode ser alterado.	Depende do tipo da variável global
Event Source	<p>CPU Event O carimbo de hora é formado num módulo processador. Ele executa a formação de eventos por completo em cada um de seus ciclos.</p> <p>IO Event O carimbo de hora é formado num módulo de E/S adequado (p. ex., AI 32 02).</p> <p>Auto Event É formado um CPU Event e, se presente, IO Events dos módulos de E/S.</p> <p>Valor padrão: CPU Event</p>	CPU Event, IO Event, Auto Event
HH Alarm text	Texto que identifica o estado de alarme do valor limite máximo	Texto
HH Alarm Value	Valor limite superior máximo que dispara um evento. Condição: (HH Alarm Value - Histerese) > H Alarm Value ou HH Alarm Value = H Alarm Value	Depende do tipo da variável global
HH Alarm Priority	Prioridade do valor limite máximo, valor padrão: 1	1...1000
HH Alarm Acknowledgment Required	<p>Ativado O operador deve confirmar a ultrapassagem do valor limite máximo (acknowledge).</p> <p>Desativado O operador não precisa confirmar a ultrapassagem do valor limite máximo.</p> <p>Valor padrão: Desativado</p>	Caixinha de controle ativada, desativada
H Alarm text	Texto que identifica o estado de alarme do valor limite superior	Texto
H Alarm Value	Valor limite superior que dispara um evento. Condição: (H Alarm Value - Histerese) > (L Alarm Value + Histerese) ou H Alarm Value = L Alarm Value	Depende do tipo da variável global
H Alarm Priority	Prioridade do valor limite superior, valor padrão: 1	1...1000
H Alarm Acknowledgment Required	<p>Ativado O operador deve confirmar a ultrapassagem do valor limite superior (acknowledge).</p> <p>Desativado O operador não precisa confirmar a ultrapassagem do valor limite superior.</p> <p>Valor padrão: Desativado</p>	Caixinha de controle ativada, desativada
Return to Normal Text	Texto que identifica o estado de alarme	Texto
Return to Normal Severity	Prioridade do estado normal, valor padrão: 1	1...1000
Return to Normal Ack Required	Confirmação do estado normal pelo usuário é necessária (acknowledge), valor padrão: Desativado	Caixinha de controle ativada, desativada
L Alarm text	Texto que identifica o estado de alarme do valor limite inferior	Texto
L Alarm Value	Valor limite inferior que dispara um evento. Condição: (L Alarm Value + Histerese) < (H Alarm Value - Histerese) ou L Alarm Value = H Alarm Value	Depende do tipo da variável global
L Alarm Priority	Prioridade do valor limite inferior, valor padrão: 1	1...1000

Coluna	Descrição	Faixa de valores
L Alarm Acknowledgment Required	<p>Ativado O operador deve confirmar não alcançar o valor limite inferior (acknowledge).</p> <p>Desativado O operador não precisa confirmar não alcançar o valor limite inferior.</p> <p>Valor padrão: Desativado</p>	Caixinha de controle ativada, desativada
LL Alarm text	Texto que identifica o estado de alarme do valor limite mínimo	Texto
LL Alarm Value	Valor limite mínimo que dispara um evento. Condição: (LL Alarm Value + Histerese) < (L Alarm Value) ou LL Alarm Value = L Alarm Value	Depende do tipo da variável global
LL Alarm Priority	Prioridade do valor limite mínimo, valor padrão: 1	1...1000
LL Alarm Acknowledgment Required	<p>Ativado O operador deve confirmar não alcançar o valor limite mínimo (acknowledge).</p> <p>Desativado O operador não precisa confirmar não alcançar o valor limite mínimo.</p> <p>Valor padrão: Desativado</p>	Caixinha de controle ativada, desativada
Alarm Hysteresis	A histerese impede a criação permanente de muitos eventos quando o valor de processo oscila muito em torno de um valor limite.	Depende do tipo da variável global

Tabela 276: Parâmetros para eventos escalares



## 10.8 Parâmetros das características do servidor X-OPC



A configuração inteira e a operação do servidor X-OPC é efetuada no SILworX. No Control Panel do SILworX é possível carregar, iniciar e parar o servidor X-OPC como um sistema de comando.

### 10.8.1 OPC Server-Set

O OPC Server-Set serve como base conjunta de parametrização para até dois servidores OPC.

As características do OPC Server-Set automaticamente são idênticas para os dois servidores X-OPC redundantes.

#### Assim cria-se um novo OPC Server-Set:

1. Abrir na árvore de estrutura **Configuration**.
2. Selecionar no menu de contexto da configuração **New, OPC Server-Set** para adicionar um novo OPC Server-Set.
3. Aceitar os valores padrão no menu de contexto das OPC Server-Set **Properties**.

A janela de diálogo características do OPC Server-Set contém os seguintes parâmetros.

Elemento	Descrição
Name	Nome do OPC Server-Set. No máximo 31 caracteres.
Safety Time [ms]	<p>O tempo de segurança é o tempo em milissegundos dentro do qual o servidor X-OPC precisa reagir a um erro.</p> <p>Condição: Tempo de segurança <math>\geq 2 \times</math> tempo de Watchdog</p> <p>Faixa de valores: 2000...400 000 ms</p> <p>Valor padrão: 20 000 ms</p>
Watchdog Time [ms]	<p>O tempo de Watchdog é o tempo em milissegundos que o servidor X-OPC pode levar no máximo para executar um ciclo de programa. Se o tempo de Watchdog definido for ultrapassado (execução do ciclo de programa demora demais), o servidor X-OPC é encerrado.</p> <p>Condição: <math>WDT \geq 1000 \text{ ms}</math> e <math>\leq 0,5 \times</math> tempo de segurança</p> <p>Faixa de valores: 1000...200 000 ms</p> <p>Valor padrão: 10 000 ms</p>
Main Enable	<p>O ajuste do interruptor OPC liberação principal influencia a função dos demais interruptores OPC.</p> <p>Se a liberação principal estiver desligada, os ajustes dos demais interruptores OPC não podem ser alterados enquanto o programa de aplicação é processado (sistema de comando em RUN).</p> <p>Valor padrão: Ativado</p>

Elemento	Descrição
Autostart	<p>Autostart define se depois de ligar ou depois de dar o Boot no servidor OPC as configurações OPC podem ser iniciadas automaticamente via arranque frio, arranque quente ou não (Desliga).</p> <p>Se Autostart está desativado, o servidor X-OPC entra no estado STOP/VALID CONFIGURATION depois do Boot. Valor padrão: Desativado</p>
Start Allowed	<p>Apenas se <i>Start Allowed</i> estiver ativado o servidor X-OPC pode ser iniciado pelo aparelho de programação.</p> <p>Se <i>Start Allowed</i> estiver desativado o servidor X-OPC não pode ser iniciado pelo aparelho de programação. Neste caso, apenas é possível iniciar o servidor X-OPC se <i>Autostart</i> estiver ativado e se ligar o Host PC ou der boot nele.</p> <p>Se <i>Autostart</i> nem <i>Start Allowed</i> estiverem ativados, o servidor X-OPC não pode mais iniciar. Isso pode ser necessário, p. ex., no caso de trabalhos de manutenção, para impedir o arranque de uma instalação. Valor padrão: Ativado</p>
Load Allowed	<p>Se <i>Load Allowed</i> está desativado, não é possível carregar uma (nova) configuração OPC para o sistema de comando.</p> <p>Desativar <i>Load Allowed</i> se quiser impedir que a configuração OPC carregada para o servidor X-OPC possa ser sobrescrita. Valor padrão: Ativado</p>
Reload Allowed	Ainda sem função!
Global Forcing Allowed	<p>Apenas se <i>Global Forcing Allowed</i> está ativado é possível iniciar <i>Global forcing</i>.</p> <hr/> <p>• <b>1</b> O Force Editor pode ser chamado para a exibição do conteúdo de variáveis mesmo se <i>Global Forcing Allowed</i> estiver desativado.</p> <hr/> <p>Valor padrão: Desativado</p>
Global Force Timeout Reaction	<p>Se <i>Global Force Timeout Reaction, Stop Resource</i> estiver ajustado, o servidor X-OPC entra no estado STOP depois de esgotar o tempo de forcing pré-ajustado. Todas as saídas do servidor X-OPC são colocadas em LOW.</p> <p>Se <i>Global Force Timeout Reaction Stop Forcing Only</i> estiver ajustado, o servidor X-OPC continua a execução da configuração OPC, mesmo depois do tempo de forcing ter esgotado.</p> <hr/> <p>• <b>1</b> Se Forcing estiver permitido, verificar o ajuste para "Stop at Force Timeout" cuidadosamente. Observar a este respeito também os comentários no Manual de segurança.</p> <hr/> <p>Valor padrão: Parar recurso</p>

Elemento	Descrição
Max. Com. Timeslice ASYNC [ms]	O <i>Max. Com. Time Slice ASYNC [ms]</i> é o tempo em milissegundos reservado por ciclo do OPC Server para processar todas as tarefas de comunicação ativas para a comunicação Peer-to-Peer. Valor padrão: 500 ms
Target cycle time [ms]	Tempo de ciclo nominal do servidor X-OPC Valor padrão: 50 ms
Safeethernet-CRC	- Versão atual - SILworX V2.36
Namespace Separator	Ponto . Barra / Dois pontos : Barra invertida \  Valor padrão: Ponto
Namespace Type	De acordo com os requisitos do cliente OPC pode ser ajustado um tipo para o espaço de nome: - Espaço de nome hierárquico - Espaço de nome plano  Valor padrão: Espaço de nome hierárquico
Changeless update	De acordo com os requisitos do cliente OPC.  Ativado: Se <i>Changeless Update</i> estiver ativado, depois de esgotar o OPC Group-UpdateRate, o servidor X-OPC entrega sempre todos os itens ao cliente OPC.  Desativado: Se <i>Changeless Update</i> estiver desativado, apenas os valores alterados são entregues (este comportamento corresponde à especificação OPC).
Cycle delay [ms]	O retardo do ciclo limita a carga da CPU do PC gerada pelo servidor X-OPC para que outros programas também possam ser processados.  Faixa de valores: 1...100 ms  Valor padrão: 5 ms
Short Tag Names for DA	Apenas se <i>Flat Namespace</i> foi selecionado é possível ativar este parâmetro. Trata-se de uma opção onde dados e eventos são oferecidos ao cliente OPC sem outro contexto (nome de caminho). Valor padrão: Desativado
Simple-Events for CPU I/O-Events	Nunca Apenas ao iniciar Sempre
Short Tag Names for A&E	Apenas se <i>Flat Namespace</i> foi selecionado é possível ativar este parâmetro. Trata-se de uma opção onde fontes de dados e eventos são oferecidos ao cliente OPC sem outro contexto (nome de caminho). Valor padrão: Desativado

Tabela 277: Características

### 10.8.2 OPC Server

#### Assim cria-se um novo OPC Server:

1. Abrir na árvore de estrutura **Configuration, OPC Server-Set**.
2. Selecionar no menu de contexto do OPC Server Set **New, OPC Server** para adicionar um novo OPC Server.
3. No menu de contexto do servidor OPC, selecionar **Properties**.

A janela de diálogo características do OPC Server contém os seguintes parâmetros.

Elemento	Descrição
Name	Nome do OPC Server. No máximo 31 caracteres.
System ID [SRS]	Valor padrão: 60000
Namespace Prefix	

Tabela 278: Características

#### Assim abre-se o OPC Host:

1. Abrir na árvore de estrutura **Configuration, OPC Server-Set, OPC Server**.
2. Selecionar no menu de contexto do **OPC Host Edit** para abrir a visão geral das interfaces IP.

A janela de diálogo do OPC Host contém os seguintes parâmetros.

Elemento	Descrição
PADT-Port	Valor padrão: 25138
Name	Nome do OPC Server-Set. No máximo 31 caracteres.
IP Address	Endereço IP do Host PC. Valor padrão: 192.168.0.1
Standard Interface	Deve ser ativado se o Host PC possuir mais do que uma porta Ethernet. Valor padrão: Ativado
HH Port	Valor padrão: 15138

Tabela 279: Edit

## 10.9 Desinstalação do servidor X-OPC

#### Assim desinstala-se o servidor X-OPC:

1. Abrir no Windows **Start, Settings, Control Panel, Software**.
2. Selecionar na lista o servidor X-OPC que deve ser desinstalado e clicar em **Remove**.
3. Seguir as instruções da rotina de desinstalação.

11 Interface serial síncrona

A interface serial síncrona não direcionada à segurança (SSI) é uma interface para absolute encoders (sistemas de medição de ângulos).  
O submódulo SSI permite a ligação de até três absolute encoders a um ciclo comum para receber simultaneamente a posição X-Y-Z.

Para a colocação em funcionamento do submódulo SSI, é necessária a configuração do COM mediante o ComUserTask e a criação da lógica no programa de aplicação com ajuda da ferramenta de programação SILworX.

11.1 Requisitos de sistema

Equipamentos necessários e requisitos de sistema:

Elemento	Descrição
Sistema de comando	HIMax com módulo COM HIMatrix a partir de CPU OS V7.24 e COM OS V12.30
Módulo CPU	As interfaces do módulo processador não podem ser usadas para SSI.
Módulo COM	Se a interface de barramento de campo serial (FB1 ou FB2) é usada, as mesmas precisam ser equipadas com o submódulo SSI, veja Capítulo 3.6.
Ativação	A liberação ocorre mediante código de liberação do software, veja Capítulo 3.4.

Tabela 280: Equipamentos necessários e requisitos de sistema ComUserTask

11.2 Diagrama de blocos

O submódulo SSI é galvanicamente separado do sistema de comando HIMA.

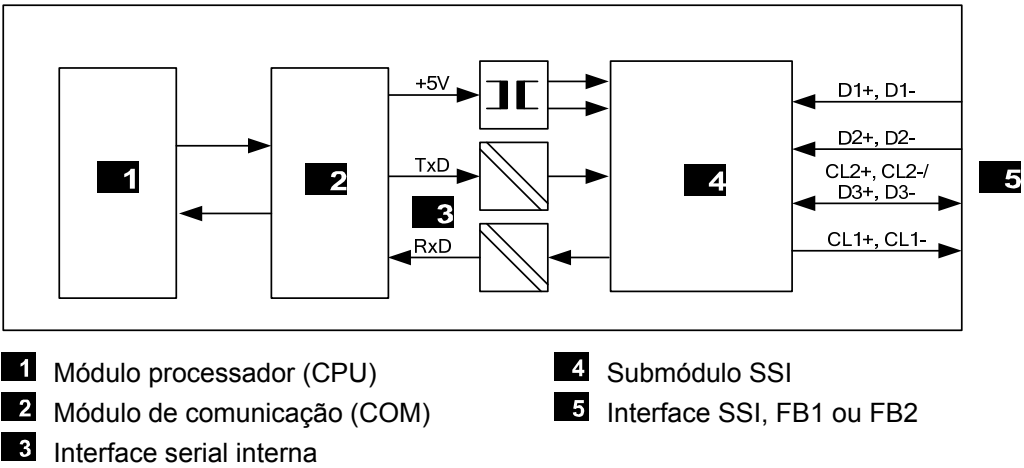


Figura 90: Diagrama de blocos

### 11.3 Tomadas D-Sub FB1 e FB2

Ao usar o submódulo SSI vale a pinagem das tomadas D-Sub FB1 e FB2 como descrito no Capítulo 3.6.

### 11.4 Configuração entre COM e submódulo SSI

A troca de dados entre o módulo de comunicação (COM) **2** e o submódulo SSI **4** ocorre pela interface serial interna **3** e deve ser realizada mediante *ComUserTask*, veja Capítulo 12 *ComUserTask*.

O protocolo de dados criado na *ComUserTask* para a troca de dados entre COM e submódulo SSI deve ser ajustado como segue:

- Baud rate 115,2 kBit/s
- Comprimento de dados 8 Bit,
- Parity even,
- 1 Stop Bit

### 11.5 Configuração da interface SSI

A interface SSI **5** é configurada pelo byte de comando inicial SSISTART, veja Tabela 281.

Para a requisição de um novo conjunto de dados de transdutor (posição atual), o byte de comando inicial SSISTART deve ser atribuído sempre de novo e repassado ao submódulo SSI **4**.

O byte de comando inicial SSISTART a ser enviado possui o seguinte formato:

Bit	Descrição
7	Auxiliary Bit Este Bit define a atribuição do byte de comando inicial.
Se Bit 7 = 1	
6...4	Os seguintes ajustes são possíveis para o pulso de deslocamento SSI 000 62,5 kHz 001 125 kHz 010 250 kHz 011 500 kHz
3	Se pino 3 e 8 são ligados como entrada de dados ou saída de pulso. 0: D3+, D3- ligado como entrada 1: CL2+, CL2- ligado como saída de pulso
2	Saída de pulso CL1 0: Ativado 1: Desativado
1	Não usado
0	Saída de pulso CL2 0: Ativado 1: Desativado
Se Bit 7 = 0	
6 ... 0	Não usado

Tabela 281: Formato de dados do byte de comando inicial SSISTART

Os dados de sensores determinados via interface SSI são passados pela interface serial interna **3** do submódulo SSI **4** ao **2** no seguinte formato e sequência.

No programa de aplicação é possível avaliar estes dados de sensores para a determinação da posição X-Y-Z (Observação: os bits de dados são repassados ao programa de aplicação na mesma sequência como são fornecidos pelo sensor).

Nº.	Canal	Bit de dados
1	Canal1	D47...D40
2		D39...D32
3		D31...D24
4		D23...D16
5		D15...D8
6		D7...D0
7	Canal2	D47...D40
8		D39...D32
9		D31...D24
10		D23...D16
11		D15...D8
12		D7...D0
13	Canal3	D47...D40
14		D39...D32
15		D31...D24
16		D23...D16
17		D15...D8
18		D7...D0

Tabela 282: Formato e sequência dos dados de sensores

#### 11.5.1 Comprimento de condutores e frequências de relógio recomendadas

A seguinte tabela mostra as frequências de relógio recomendadas para a interface SSI dependendo do comprimento do condutor de campo.

Comprimento do condutor/m	Clockrate/kHz
< 25	≤ 500
< 50	< 400
< 100	< 300
< 200	< 200
< 400	< 100

Tabela 283: Frequências de relógio recomendadas dependendo do comprimento de condutores de campo

## 11.6 Avisos de aplicação

O submódulo SSI é galvanicamente separado do sistema de comando HIMA, por isso os sensores SSI no campo devem ser alimentados por uma alimentação com tensão externa.

As seguintes aplicações são possíveis com um submódulo SSI:

- Ligação de 3 sensores para a determinação simultânea de coordenadas X-Y-Z  
1 pulso de deslocamento SSI (CL1+, CL1-) e 3 canais de dados (D1+, D1-, D2+, D2-, D3+, D3-) para determinar coordenadas XYZ simultaneamente.  
Todos os 3 sensores são alimentados pelo mesmo pulso (CL1+, CL1-) e, assim, com a mesma frequência de relógio.
- Ligação de 2 sensores para a determinação simultânea de coordenadas X-Y  
2 pulsos de deslocamento SSI (CL1+, CL1-, CL2+, CL2-) e 2 canais de dados (D1+, D1-, D2+, D2-) para determinar coordenadas XY simultaneamente.  
Ambos os sensores são controlados por pulsos separados (CL1+, CL1-, CL2+, CL2-). Ambos tem a mesma frequência de relógio.
- Ligação de 1 sensor  
1 pulso de deslocamento SSI (CL1+, CL1- ou CL2+, CL2-) e 1 canal de dados (D1+, D1- ou D2+, D2-).

---

**i**

A montagem do submódulo SSI na zona 2 (Diretiva CE 94/9/CE, ATEX) é permitida se requisitos especiais X são observados.

---



## 12 ComUserTask

Além do programa de lógica que é criado com o SILworX, é possível rodar adicionalmente um programa em C no sistema de comando.

Este programa em C não seguro roda como ComUserTask no módulo de comunicação do sistema de comando, sem retroalimentação ao módulo processador seguro.

A ComUserTask possui um ciclo próprio, independente do ciclo da CPU.

Assim, é possível criar quaisquer aplicações em C e implementar as mesmas como ComUserTask, p. ex.:

- Interfaces de comunicação para protocolos especiais (TCP, UDP etc.).
- Função de gateway entre TCP/UDP e comunicação serial.

### 12.1 Requisitos de sistema

Equipamentos necessários e requisitos de sistema:

Elemento	Descrição
Sistema de comando	HIMax com módulo COM HIMatrix a partir de CPU OS V7 e COM OS V12
Módulo CPU	As interfaces Ethernet do módulo processador não podem ser usadas para ComUserTask.
Módulo COM	Ethernet 10/100BaseT Conexões D-Sub FB1 e FB2, p. ex., para RS232 Se a interface de barramento de campo serial (FB1 ou FB2) é usada, as mesmas precisam ser equipadas com o submódulo SSI HIMA opcional, veja Capítulo 3.6.
Ativação	A liberação ocorre mediante código de liberação do software, veja Capítulo 3.4.

Tabela 284: Equipamentos necessários e requisitos de sistema ComUserTask

Características da ComUserTask:

Elemento	Descrição
ComUserTask	Para cada sistema de comando HIMax pode ser configurada uma ComUserTask.
Direcionado à segurança	Não
Troca de dados	Configurável
Área de código e de dados	Veja Capítulo 12.5.4
Stack	O Stack está numa memória para isso reservada fora da área de código/de dados. Veja Capítulo 12.5.4

Tabela 285: Características ComUserTask

#### 12.1.1 Criar uma ComUserTask

**Assim cria-se uma nova ComUserTask:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. No menu de contexto de protocolos, seleccionar **New, ComUserTask** para adicionar uma nova ComUserTask.
3. No menu de contexto **ComUserTask Properties**, seleccionar o **COM-Module**.  
Os ajustes padrão podem ser mantidos para a primeira configuração.

## 12.2 Requisitos

Para a programação da ComUserTask existe, adicionalmente à gama de comandos C, uma biblioteca própria à disposição (veja Capítulo 12.4) com funções definidas.

### Ambiente de desenvolvimento

Pertencem ao ambiente de desenvolvimento o GNU C Compiler e Cygwin, disponíveis num CD de instalação (não incluído no SILworX) e sujeitos às condições de licenciamento GNU General Public License (veja [www.gnu.org](http://www.gnu.org)).

As mais novas versões e documentações sobre o ambiente de desenvolvimento podem ser descarregados das respectivas páginas na Internet: [www.cygwin.com](http://www.cygwin.com) e [www.gnu.org](http://www.gnu.org).

### Sistema de comando

No caso dos sistemas de comando HIMax, a ComUserTask não tem acesso às entradas e saídas seguras do hardware. Se o acesso às entradas e saídas de hardware for necessário, um programa de aplicação da CPU para a conexão das variáveis é necessário (veja Capítulo 12.4.5).

## 12.3 Abreviações

Abreviação	Significado
CUCB	COM User Callback (Funções CUCB são chamadas pelo módulo COM)
CUIT	COM User IRQ Task
CUL	COM User Library (Funções CUL são chamadas pela CUT)
CUT	ComUserTask
GNU	Projeto GNU
IF	InterFace
FB	Fieldbus Interface – interface do barramento de campo do sistema de comando
FIFO	First In First Out (primeiro a entrar, primeiro a sair – memória de dados)
NVRam	Non Volatile Random Access Memory, não volátil Speicher
SSI	<b>S</b> ynchron <b>S</b> erial <b>I</b> nterface

Tabela 286: Abreviações

## 12.4 Interface CUT no SILworX

A comunicação de dados de processo de dados de processo da ComUserTask ocorre entre os módulos COM e CPU.

### ⚠ ATENÇÃO



A ComUserTask carregada não pode usar comandos privilegiados do módulo COM. O código da CUT roda no COM sem retroalimentação para CPU. Assim, o módulo CPU seguro está protegido contra o código da CUT. Porém, deve ser observado que erros no código CUT podem interferir com a função do COM como um todo e, assim, influenciar ou até impedir o funcionamento do sistema de comando. As funções de segurança do módulo CPU não são afetadas por isso.

### 12.4.1 Schedule Interval [ms]

A ComUserTask é chamada num Schedule Interval [ms] parametrizado nos estados RUN e STOP\_VALID\_CONFIG do sistema de comando (módulo COM).

O Schedule Interval [ms] é ajustado no SILworX, nas *Properties* da ComUserTask.

Schedule Interval [ms]	
Faixa de valores:	10...255 ms
Valor padrão:	15 ms

Tabela 287: Schedule Interval [ms]

·  
1

O tempo de processador disponível à CUT depende das outras funções parametrizadas do COM, como, p. ex., **safeethernet**, Modbus TCP, etc. Se a CUT não terminar dentro do Schedule Interval, cada chamada para reiniciar a CUT é recusada até a CUT estiver processada.

### 12.4.2 Pré-processamento do Scheduling

#### No modo operacional RUN do sistema de comando:

Antes de cada chamada da CUT, COM disponibiliza os dados de processo do módulo CPU seguro da CUT numa área de memória definida pela CUT.

#### No modo operacional STOP do sistema de comando:

Não ocorre troca de dados de processo do COM para o módulo CPU seguro.

### 12.4.3 Pós-processamento do Scheduling

#### No modo operacional RUN do sistema de comando:

Após cada chamada da CUT, COM disponibiliza os dados de processo da CUT numa área de memória definida pela CUT.

#### No modo operacional STOP do sistema de comando:

Não ocorre troca de dados de processo do COM para o módulo CPU seguro.

12.4.4 STOP\_INVALID\_CONFIG

Se o COM estiver no estado STOP\_INVALID\_CONFIG, a CUT não é executada.

Se o COM mudar para o estado STOP\_INVALID\_CONFIG e está executando a CUT ou a CUIT, as mesmas são terminadas.

12.4.5 Variáveis da interface CUT (CPU<->CUT)

Parametrização da comunicação de dados de processo não direcionada à segurança entre módulo CPU seguro e COM (CUT).

Direção de transmissão	Tamanho máximo dos dados de processo
COM->CPU	16375 Bytes de dados (16384 Bytes – 9 bytes de status)
CPU->COM	16382 Bytes de dados (16384 Bytes – 2 bytes de comando)

Troca de dados de processo com ComUserTask

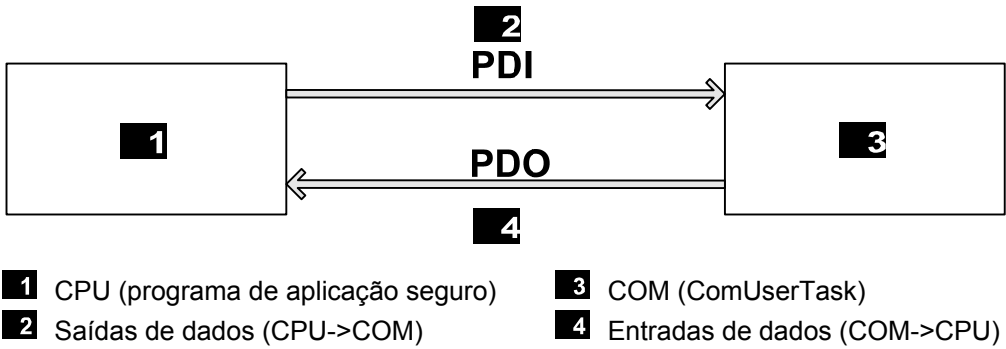


Figura 91: Troca de dados de processo entre CPU e COM (CUT)

Todos os tipos de dados que são usados no SILworX podem ser trocados.

A estrutura dos dados deve ser parametrizada no SILworX.

O tamanho das estruturas de dados CUT\_PDI e CUT\_PDO (no C-Code compilado da CUT) devem corresponder ao mesmo tamanho da estrutura de dados configurada no SILworX.

---

**i** Se no código C compilado as estruturas de dados CUT\_PDI e CUT\_PDO não existirem ou não tiverem o mesmo tamanho como a estrutura de dados dos dados de processo parametrizados no SILworX, então, a configuração é inválida e COM assume o estado STOP\_INVALID\_CONFIG.

A comunicação de dados de processo apenas ocorre no modo de operação RUN.

---

### 12.4.6 Função de menu características

A função de menu **Properties** do menu de contexto da ComUserTask abre o diálogo **Properties**.

Elemento	Descrição
Type	ComUserTask
Name	Nome livre inequívoco para uma ComUserTask
Force Process Data Consistency	Ativado: Transferência dos dados completos do protocolo de CPU para COM dentro de um ciclo de CPU. Desativado: Transferência dos dados completos do protocolo de CPU para COM, distribuídos sobre vários ciclos de CPU cada vez 1100 Byte por direção de dados. Com isso, eventualmente também é possível reduzir o tempo de ciclo do sistema de comando. Valor padrão: Ativado
Module	Seleção do módulo COM no qual o protocolo é processado.
Activate Max. $\mu$ P-Budget	Ativado: Transferir o limite do $\mu$ P-Budget do campo <i>Max. <math>\mu</math>P-Budget in [%]</i> . Desativado: Não usar limite do $\mu$ P-Budget para este protocolo.
Max. $\mu$ P-Budget in [%]	Valor máximo de $\mu$ P-Budget do módulo que pode ser produzido ao processar o protocolo. Faixa de valores: 1...100% Valor padrão: 30%
Behavior on CPU/COM Connection Loss	No caso da perda de conexão do módulo processador ao módulo de comunicação, dependendo deste parâmetro, as variáveis de entrada são inicializadas ou usadas de forma inalterada no módulo processador. (P. ex., se o módulo de comunicação é puxado para fora durante a comunicação). Adopt Initial Data      As variáveis de entrada são resetadas para os valores iniciais. Retain Last Value      As variáveis de entrada mantêm o último valor.
Schedule Interval [ms]	A ComUserTask é chamada num Schedule Interval [ms] parametrizado do sistema de comando (módulo COM), veja Capítulo 12.4.1. Faixa de valores: 10. 255 ms Valor padrão: 15 ms
User Task	Caminho para loadable, se já foi carregado.

Tabela 288: Características gerais do dispositivo PROFINET-IO

### 12.4.7 Função de menu Edit

Mediante a função de menu **Edit** podem ser acessados os registro de variáveis de processo e variáveis de sistema.

#### 12.4.7.1 Variáveis de sistema

O registro **System Variables** contém os seguintes parâmetros de sistema para supervisão e controle da CUT:

Name	Função										
Execution Time [DWORD]	Tempo de execução da ComUserTask em µs										
Real schedule interval [DWORD]	Distância de tempo entre das execuções da ComUserTask em ms										
User task state control [WORD]	<p>A seguinte tabela mostra as opções com as quais o usuário pode controlar a ComUserTask com o parâmetro <i>User Task State Control</i>:</p> <table> <tr> <th>Função</th><th>Descrição</th></tr> <tr> <td>DISABLED 0x8000</td><td>O programa de aplicação bloqueia a CUT (ou seja, a CUT não é iniciada).</td></tr> <tr> <td>AUTOSTART 0 (padrão)</td><td>Após terminação da CUT a CUT inicia automaticamente depois de eliminar a avaria ou o erro.</td></tr> <tr> <td>TOGGLE_MODE_0 0x0100</td><td>Após a terminação da CUT apenas é permitido reiniciar a CUT depois de colocar TOGGLE_MODE_1.</td></tr> <tr> <td>TOGGLE_MODE_1 0x0101</td><td>Após a terminação da CUT apenas é permitido reiniciar a CUT depois de colocar TOGGLE_MODE_0.</td></tr> </table>	Função	Descrição	DISABLED 0x8000	O programa de aplicação bloqueia a CUT (ou seja, a CUT não é iniciada).	AUTOSTART 0 (padrão)	Após terminação da CUT a CUT inicia automaticamente depois de eliminar a avaria ou o erro.	TOGGLE_MODE_0 0x0100	Após a terminação da CUT apenas é permitido reiniciar a CUT depois de colocar TOGGLE_MODE_1.	TOGGLE_MODE_1 0x0101	Após a terminação da CUT apenas é permitido reiniciar a CUT depois de colocar TOGGLE_MODE_0.
Função	Descrição										
DISABLED 0x8000	O programa de aplicação bloqueia a CUT (ou seja, a CUT não é iniciada).										
AUTOSTART 0 (padrão)	Após terminação da CUT a CUT inicia automaticamente depois de eliminar a avaria ou o erro.										
TOGGLE_MODE_0 0x0100	Após a terminação da CUT apenas é permitido reiniciar a CUT depois de colocar TOGGLE_MODE_1.										
TOGGLE_MODE_1 0x0101	Após a terminação da CUT apenas é permitido reiniciar a CUT depois de colocar TOGGLE_MODE_0.										
State of the User Task [BYTE]	1 = RUNNING (CUT rodando) 0 = ERROR (CUT não roda devido a um erro)										

Tabela 289: Variáveis de sistema da ComUserTask

## 12.4.7.2 Variáveis de processo

## Sinais de entrada (COM-&gt;CPU)

No registro **Inputs Signals** são introduzidas as variáveis que devem ser transmitidas de COM (CUT) para CPU (área de entrada de CPU).

**! CUIDADO**

**Dados não seguros da ComUserTask!**

**As variáveis não seguras da ComUserTask não podem afetar as funções de segurança do programa de aplicação de CPU.**

## Sinais de entrada da ComUserTask

Name	Tipo de dados	Offset
CUT_Counter	DWORD	0
Time_Stamp	DWORD	4

Tabela 290: Sinais de entrada da ComUserTask

## Entrada necessária no código C

O código C da ComUserTask deve conter para as saídas de COM (área de entrada de CPU) a seguinte estrutura de dados CUT\_PDO:

```
/* SILworX Input Records (COM->CPU) */
uword CUT_PDO[1] __attribute__((section("CUT_PD_OUT_SECT"), aligned(1)));
```

O tamanho da estrutura de dados CUT-PDO deve corresponder ao tamanho das entradas de dados configurads no SILworX.

## Sinais de saída (CPU-&gt;COM)

No registro **Output Signals** são introduzidas as variáveis que devem ser transmitidas de CPU (área de saída de CPU) para COM (CUT).

## Sinais de saída da ComUserTask

Name	Tipo de dados	Offset
CPU_COM_1	WORD	0
CPU_COM_2	WORD	2

Tabela 291: Sinais de saída da ComUserTask

## Entrada necessária no código C

O código C da ComUserTask deve conter para as entradas de COM (área de saída de CPU) a seguinte estrutura de dados CUT\_PDI:

```
/* SILworX Output Records (CPU->COM) */
uword CUT_PDI[1] __attribute__((section("CUT_PD_IN_SECT"), aligned(1)));
```

O tamanho da estrutura de dados CUT-PDI deve corresponder ao tamanho das saídas de dados configurads no SILworX.

## 12.5 Funções CUT

### 12.5.1 Funções COM User Callback

As funções COM User Callback possuem todas o prefixo **CUCB\_** e são chamadas diretamente pelo COM no caso de eventos.

**i**

---

Todas as funções COM User Callback devem ser definidas no código C do usuário! Também a função CUCB\_IrqService não suportada para HIMax e HIMatrix deve ser definida no código C do usuário por motivos de compatibilidade.  
Protótipo da função: void CUCB\_IrqService(udword devNo) {}

---

As funções COM User Callback (CUCB) e COM User Library (CUL) compartilham a mesma memória de código e de dados bem como o Stack. Estas funções garantem a consistência mútua dos dados (variáveis) compartilhados.

### 12.5.2 Funções COM User Library

Todas as funções e variáveis COM User Library têm o prefixo **CUL\_** e são chamados na CUT.

Estas funções CUL estão disponíveis pelo arquivo de objetos **libcut.a**.

### 12.5.3 Header Files

Os dois arquivos de cabeçalho **cut.h** e **cut\_types.h** contêm todos os protótipos de função para CUL/CUCB e os tipos de dados e constantes correspondentes.

Para encurtar a forma de escrever, os seguintes tipos de dados são definidos no Header File **cut\_types.h**:

```
typedef unsigned long    udword;
typedef unsigned short   uword;
typedef unsigned char     ubyte;
typedef signed long      dword;
typedef signed short     word;
typedef signed char      sbyte;
#ifndef HAS_BOOL
typedef unsigned char    bool; // com 0=FALSE, senão TRUE
#endif
```



#### 12.5.4 Área de código e de dados e Stack para a CUT

A área de código/dados é uma área de memória adjacente que inicia com o segmento de código e o segmento de dados iniciais e continua com os segmentos de dados. No arquivo de controle de link (**makeinc.inc.app** e **section.dld**), é definida a sequência descrita de segmentos e a quantidade de memória disponível (vale para HIMax e HIMatrix).

A ComUserTask distribui a área de memória disponível de forma ideal entre o código e os dados com ajuda do arquivo de linker da HIMA.

Elemento	HIMax/HIMatrix L2	HIMatrix L3
Endereço inicial	0x790000	0x800000
Comprimento	448 kByte	4 MB

O Stack está numa memória para isso reservada que é definida durante o tempo de execução do sistema operacional COM.

Elemento	HIMax/HIMatrix L2	HIMatrix L3
Endereço final	Dinâmico do ponto de vista da CUT	Dinâmico do ponto de vista da CUT
Comprimento	64 kByte	500 kByte

#### 12.5.5 Função inicial CUCB\_TaskLoop

A função `CUCB_TaskLoop( )` é a função inicial para a ComUserTask.

A execução do programa da ComUserTask iniciar com a chamada desta função (veja *Schedule-Interval[ms]*, Capítulo 12.4.1).

##### Protótipo da função:

```
void CUCB_TaskLoop(udword mode)
```

##### Parâmetros:

A função tem o seguinte parâmetro

Parâmetro	Descrição
mode	1 = MODE_STOP corresponde ao modo STOP_VALID_CONFIG 2 = MODE_RUN operação normal do sistema de comando

Tabela 292: Parâmetro

### 12.5.6 Interfaces seriais RS485/RS232 IF

As interfaces de barramento de campo usadas devem ser equipadas com os respectivos submódulos de barramento de campo (hardware).

---

**i**

Para cada sistema de comando HIMax, a respectiva documentação de sistema está à disposição.

---

#### 12.5.6.1 CUL\_AscOpen

A função `CUL_AscOpen( )` inicializa a interface serial introduzida (*comId*) com os parâmetros transferidos. Depois de chamar a função `CUL_AscOpen( )`, o COM inicia imediatamente com a recepção de dados por esta interface.

Os dados recebidos são armazenados com um FIFO de software do tamanho 1kByte para **cada** interface serial inicializada.

Os dados são armazenados até serem lidos com a função `CUL_AscRcv( )` pela CUT.

---

**i**

Se a leitura dos dados do FIFO for mais lenta do que a recepção de novos dados, os novos dados recebidos são descartados.

---

#### Protótipo da função:

```
udword          Udword comId,  
CUL_AscOpen(    Ubyte duplex,  
                udword baudRate,  
                ubyte parity,  
                ubyte stopBits)
```

#### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
comId	Fieldbus Interface (RS485, RS232) 1 = FB1 2 = FB2 3 = FB3 4 = FB4_SERVICE
duplex	0 = Full-Duplex (só para FB4 se RS232 for permitido) 1 = Half-Duplex
baudRate	1 = 1200 Bit 2 = 2400 Bit 3 = 4800 Bit 4 = 9600 Bit 5 = 19200 Bit 6 = 38400 Bit (baud rate máximo HIMax) 7 = 57600 Bit (só HIMatrix) 8 = 115000 Bit (só HIMatrix)
O comprimento de bits de dados está ajustado de forma fixa a 8 bits de dados. A estes 8 bits de dados são adicionados os bits parametrizados para Parity e Stopbits e um Startbit.	
parity	0 = NONE 1 = EVEN 2 = ODD
stopBits	1 = 1 Bit 2 = 2 Bits

Tabela 293: Parâmetros

**Valor de retorno:**

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header cut.h.

Código de erro	Descrição
CUL_OKAY	A inicialização foi executada com êxito.
CUL_ALREADY_IN_USE	A interface está em uso por outras funções de COM ou já está aberta.
CUL_INVALID_PARAM	Foram passados parâmetros ou combinações de parâmetros inadmissíveis.
CUL_DEVICE_ERROR	Outros erros

Tabela 294: Valor de retorno

### 12.5.6.2 CUL\_AscClose

A função `CUL_AscClose()` fecha a interface serial introduzida em *comId*.

Neste caso, os dados recebidos, mas ainda não lidos pela função `CUL_AscRcv()` são excluídos da memória FIFO.

**Protótipo da função:**

Udword `CUL_AscClose(udword comId)`

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
comId	Fieldbus Interface (RS485, RS232) 1 = FB1 2 = FB2 3 = FB3 4 = FB4_SERVICE

Tabela 295: Parâmetro

**Valor de retorno:**

É retornado um código de erro (udword).

Os códigos de erro são definidos no Header File `cut.h`.

Código de erro	Descrição
CUL_OKAY	A interface foi fechada com êxito.
CUL_NOT_OPENED	A interface não estava aberta (pela CUT).
CUL_INVALID_PARAM	Foram passados parâmetros ou combinações de parâmetros inadmissíveis.
CUL_DEVICE_ERROR	Outros erros

Tabela 296: Valor de retorno

### 12.5.6.3 CUL\_AscRcv

A função `CUL_AscRcv()` pede à COM disponibilizar uma quantidade definida de dados da FIFO.

Logo que a quantidade de dados solicitada estiver disponível (e CUL ou o Scheduling permitirem), COM chama a função `CUCB_AscRcvReady()`.

Se não houver dados suficiente na memória FIFO, a função `CUL_AscRcv()` retorna imediatamente.

O pedido para a recepção de dados permanece na memória até:

- O pedido ter sido processado por completo ou
- a função `CUL_AscClose()` ser chamada ou
- redefinida por um novo pedido.

**i**

Ate o pedido terminar, o conteúdo de `*pBuf` apenas pode ser alterado pela função `CUCB_AscRcvReady()`.

#### Protótipo da função:

Udword `CUL_AscRcv`(udword `comId`, CUCB\_ASC\_BUFFER `*pBuf`)

```
typedef struct CUCB_AscBuffer {

    bool  bAscState;        // for using by CUT/CUCB
    bool  bError;           // for using by CUT/CUCB
    uword align;            // COM is 4 aligned, long's are higher-performance
    udword mDataIdx;        // Byte offset in aData from there on the data are
                           // located
    udword mDataMax;        // max. Byte offset: (mDataMax-mDataIdx)
                           // indicates,
    udword aData[1];        // how many bytes in aData have to be sent or
                           // received
}CUCB_ASC_BUFFER;        // Start point of the data copy range
```

#### Parâmetros

A função tem os seguintes parâmetros:

Parâmetro	Descrição
<code>comId</code>	Fieldbus Interface (RS485, RS232) 1 = FB1 2 = FB2 3 = FB3 4 = FB4_SERVICE
<code>pBuf</code>	Define o volume de dados requisitado e o local para o qual devem ser copiados antes de chamar <code>CUCB_AscReady()</code> . Se já houver dados suficientes na memória FIFO, <code>CUCB_AscRcvReady()</code> é chamada durante <code>CUL_AscRcv()</code> .

Tabela 297: Parâmetro

**Valor de retorno:**

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header cut.h.

Código de erro	Descrição
CUL_OKAY	Se o pedido terminou com êxito, senão, código de erro.
CUL_NOT_OPENED	Se a interface não foi aberta pela CUT.
CUL_INVALID_PARAM	Foram passados parâmetros ou combinações de parâmetros inadmissíveis.
CUL_DEVICE_ERROR	Outros erros

Tabela 298: Valor de retorno

**Restrições:**

- Se a área de memória definida por CUCB\_ASC\_BUFFER não está no segmento de dados da CUT, CUIT e CUT são terminados.
- É possível solicitar 1024 Bytes de dados, no máximo.
- É possível solicitar 1 Byte de dados, no mínimo.

#### 12.5.6.4 CUCB\_AscRcvReady

Se o COM chamar a função `CUCB_AscRcvReady()`, então o volume de dados solicitado está disponível na FIFO (dados da interface serial definida no parâmetro `comId`).

Os dados foram anteriormente solicitados com a função `CUL_AscRcv()`.

A chamada da função `CUCB_AscRcvReady()` pode ocorrer dentro e fora da chamada da função `CUL_AscRcv()`. O contexto de tarefa sempre é da CUT.

A função `CUCB_AscRcvReady()` pode chamar todas as funções CUT Library.

Também é permitido

- o aumento de `mDataMax`, ou
- a nova parametrização de `mDataIdx` e `mDataMax` de dados `*pBuf` atribuídos à `comId` (para continuar a leitura).

O elemento de estrutura de `CUCB_ASC_BUFFER.mDataIdx` tem o valor de `CUCB_ASC_BUFFER.mDataMax`.

##### Protótipo da função:

```
void CUCB_AscRcvReady(udword comId)
```

##### Parâmetros:

A função tem o seguinte parâmetro:

Parâmetro	Descrição
comId	Fieldbus Interface (RS485, RS232) 1 = FB1 2 = FB2 3 = FB3 4 = FB4_SERVICE

Tabela 299: Parâmetro

##### Restrições:

Se a área de memória definida por `CUCB_ASC_BUFFER` não está no segmento de dados da CUT, CUIT e CUT são terminados.

### 12.5.6.5 CUL\_AscSend

A função `CUCB_AscSend` envia a quantidade de dados definida pelo parâmetro `pBuf` pela interface serial `comId`.

A quantidade de dados definida deve ser  $\geq 1$  Byte e  $\leq 1$ kByte.

Depois de enviar, a função `CUCB_AscSendReady()` é chamada.

No caso de erro

- não é enviado e
- a função `CUCB_AscSendReady()` não é chamada.

Protótipo da função:

Udword `CUL_AscSend`(udword `comId`, `CUCB_ASC_BUFFER` \*`pBuf`)

#### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
<code>comId</code>	Fieldbus Interface (RS485, RS232) 1 = FB1 2 = FB2 3 = FB3 4 = FB4_SERVICE
<code>pBuf</code>	Define a quantidade de dados a ser enviada

Tabela 300: Parâmetro

#### Valor de retorno:

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header `cut.h`.

Código de erro	Descrição
<code>CUL_OKAY</code>	Se o envio foi efetuado com êxito
<code>CUL_WOULDBLOCK</code>	Se uma mensagem anteriormente enviada ainda não foi enviada
<code>CUL_NOT_OPENED</code>	Se a interface não foi aberta pela CUT
<code>CUL_INVALID_PARAM</code>	Foram passados parâmetros ou combinações de parâmetros inadmissíveis.
<code>CUL_DEVICE_ERROR</code>	Outros erros

Tabela 301: Valor de retorno

#### Restrições:

Se a área de memória definida por `CUCB_ASC_BUFFER` não está no segmento de dados da CUT, CUIT e CUT são terminados.



### 12.5.6.6 CUCB\_AscSendReady

Se COM chama a função `CUCB_AscSendReady( )` o envio dos dados com a função `CUCB_AscSend( )` pela interface serial está encerrado.

O contexto de tarefa sempre é da CUT. A função `CUCB_AscSendReady( )` pode chamar todas as funções CUT Library.

**Protótipo da função:**

```
void CUCB_AscSendReady(udword comId)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
comId	Fieldbus Interface (RS485, RS232) 1 = FB1 2 = FB2 3 = FB3 4 = FB4_SERVICE

Tabela 302: Parâmetro

### 12.5.7 UDP/TCP-Socket-IF

No máximo 8 soquetes estão à disposição para utilização simultaneamente, independente do protocolo usado.

A conexão física ocorre pelas interfaces 10/100BaseT Ethernet do sistema de comando.

#### 12.5.7.1 CUL\_SocketOpenUdpBind

A função `CUL_SocketOpenUdpBind()` cria um soquete do tipo UDP e vincula o soquete à porta selecionada.

O endereço para vincular sempre é `INADDR_ANY`, ou seja, todas as mensagens endereçadas para COM para UDP/Port são recebidas. Soquetes sempre são operados no modo non-blocking, ou seja, esta função não gera bloqueios.

##### Protótipo da função:

`dword CUL_SocketOpenUdpBind (uword port, uword *assigned_port_ptr)`

##### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
port	Um número de porta livre, não ocupado por COM $\geq 0$ . Se o parâmetro port = 0, então, o soquete é vinculado à primeira porta livre.
assigned_port_ptr	Endereço para o qual o número de porta vinculado deve ser copiado se port = 0, ou ZERO, se não for o caso

Tabela 303: Parâmetro

##### Valor de retorno:

É retornado um código de erro (uword).

Os códigos de erro são definidos no arquivo Header cut.h.

Código de erro	Descrição
Socket number	SocketNumber atribuído para UDP se $> 0$ ; Os códigos de erro são $< 0$
CUL_ALREADY_BOUND	Vincular a uma porta para UDP não é possível
CUL_NO_MORE_SOCKETS	Não há recursos disponíveis para Socket
CUL_SOCK_ERROR	Outros erros de Socket

Tabela 304: Valor de retorno

##### Restrições:

Se `assigned_port_ptr` não for propriedade de CUT, então, CUT/CUIT são terminadas.

### 12.5.7.2 CUL\_SocketOpenUdp

A função `CUL_SocketOpenUdp( )` cria um soquete do tipo UDP sem vinculação a uma porta. Depois disso, as mensagens podem ser apenas enviadas pelo soquete, não há recepção.

**Protótipo da função:**

`dword CUL_SocketOpenUdp (void)`

**Parâmetros:**

Sem

**Valor de retorno:**

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header cut.h.

Código de erro	Descrição
Socket number	SocketNumber atribuído para UDP se > 0 Códigos de erro são < 0
CUL_NO_MORE_SOCKETS	Não há recursos disponíveis para Socket
CUL SOCK_ERROR	Outros erros de Socket

Tabela 305: Valor de retorno

### 12.5.7.3 CUL\_NetMessageAlloc

A função `CULMessageAlloc()` aloca memória de mensagem para a utilização de

- `CUL_SocketSendTo()` com UDP e
- `CUL_SocketSend()` com TCP

No máximo 10 mensagens podem ser usadas simultaneamente na CUT.

#### Protótipo da função:

```
void *CUL_NetMessageAlloc(udword size, ubyte proto)
```

#### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
size	Quantidade de memória necessária em Bytes, deve ser $\geq 1$ Byte e $\leq 1400$ Byte
proto	0 = TCP 1 = UDP

Tabela 306: Parâmetros

#### Retorno:

Endereço de buffer para o qual os dados de trabalho a serem enviados devem ser copiados. Nunca as áreas de memória fora da área alocada podem ser usadas para escrita. Não há áreas à disposição para os protocolos de transporte usados (Ethernet/IP/UDP ou TCP).

#### Restrições:

Se não tiver outros recursos de memória à disposição ou se o tamanho de parâmetro for excessivo ou se `proto > 1`, CUT e CUIT são terminadas.

#### 12.5.7.4 CUL\_SocketSendTo

A função `CUL_SocketSendTo()` envia a mensagem anteriormente alocada e enche com `CULNetMessageAlloc()` como pacote UDP ao endereço de destino `destIp/destPort`.

Depois do envio, a memória de mensagens `pMsg` é liberada automaticamente de novo.

Para cada envio, primeiramente precisa ser alocada da memória de mensagem com a função `CULMessageAlloc()`.

##### Protótipo da função:

```
DWORD CUL_SocketSendTo(
    DWORD socket,
    void *pMsg,
    UDWORD size,
    UDWORD destIp,
    UDWORD destPort)
```

##### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
Socket	Soquete anteriormente criado com <code>CUL_SocketOpenUdp()</code>
pMsg	Memória dos dados de trabalho anteriormente reservada com <code>CULNetMessageAlloc()</code>
Size	Quantidade de memória em Bytes, deve ser $\leq$ quantidade anteriormente alocada
destIp	Endereço de destino $\neq 0$ , também <code>0xffffffff</code> permitido como Broadcast
destPort	Porta de destino $\neq 0$

Tabela 307: Parâmetros

##### Valor de retorno:

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header `cut.h`.

Código de erro	Descrição
CUL_OKAY	Mensagem enviada com êxito
CUL_NO_ROUTE	Não há roteamento para alcançar <code>destIp</code>
CUL_WRONG SOCK	Tipo de soquete incorreto ou soquete não existe
CUL_SOCKET_ERROR	Outros erros de Socket

Tabela 308: Valor de retorno

##### Restrições:

Se `pMsg` não for uma mensagem da propriedade de CUT ou se o tamanho for excessivo para `pMsg`, então CUT/CUIT são terminadas.

#### 12.5.7.5 CUCB\_SocketUdpRcv

Com chama a função CUCB\_SocketUdpRcv() se tiver dados do soquete disponíveis. No Callback, os dados devem ser copiados de \*pMsg para CUT Data, se necessário. Depois do retorno da função, não é mais permitido acessar \*pMsg.

##### Protótipo da função:

```
void CUCB_SocketUdpRcv(      dword socket,
                             void *pMsg,
                             udword packetLength,
                             udword dataLength)
```

##### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
socket	Soquete anteriormente criado com CUL_SocketOpenUdp()
pMsg	pMsg aponta para o início do pacote UDP inclusive o EthernetHeader. Pelo cabeçalho Ethernet, o remetente da mensagem pode ser determinado.
packetLength	O comprimento do pacote está em packetLength, neste caso, o comprimento do Header está incluído.
dataLength	O comprimento da proporção de dados de trabalho UDP está em dataLength.

Tabela 309: Parâmetros

### 12.5.7.6 CUL\_NetMessageFree

A função `CUL_NetMessageFree()` libera a mensagem anteriormente alocada com `CUL_NetMessageAlloc()`.

Esta função normalmente não é necessária, pois pela chamada da função `CUL_SocketSendTo()` automaticamente ocorre uma liberação.

**Protótipo da função:**

```
void CUL_NetMessageFree(void *pMsg)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetros	Descrição
<code>pMsg</code>	Memória anteriormente reservada com <code>CUL_NetMessageAlloc()</code>

Tabela 310: Parâmetro

**Restrições:**

Se `pMsg` não for uma mensagem da propriedade de CUT, então CUT/CUIT são terminadas.

### 12.5.7.7 CUL\_SocketOpenTcpServer

A função `CUL_SocketOpenServer()` cria um soquete do tipo TCP e vincula o soquete à porta selecionada.

O endereço para vincular sempre é `INADDR_ANY`. Adicionalmente, COM recebe o pedido de executar *listen* no Stream Socket. Soquetes sempre são operados no modo non-blocking, ou seja, esta função não gera bloqueios.

Para a operação posterior do soquete, veja `CUCB_SocketTryAccept()` e `CUL_SocketAccept()`.

#### Protótipo da função:

`dword CUL_SocketOpenTcpServer(udword port, udword backlog)`

#### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
port	Um número de porta não ocupado por COM > 0
backlog	Quantidade máxima de estabelecimentos de conexão em espera para o soquete

Tabela 311: Parâmetro

#### Valor de retorno:

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header cut.h.

Código de erro	Descrição
Socket number	SocketNumber atribuído para UDP se > 0 Os códigos de erro são < 0
CUL_ALREADY_BOUND	Vincular a port/proto não é possível
CUL_NO_MORE_SOCKETS	Não há recursos disponíveis para Socket
CUL_SOCK_ERROR	Outros erros de Socket

Tabela 312: Valor de retorno

#### Restrições:

No caso de executar com êxito, soquete 1 é consumido.



### 12.5.7.8 CUCB\_SocketTryAccept

COM chama a função `CUL_SocketTryAccept ( )` se houver uma requisição de conexão TCP ativa.

Com esta requisição é possível criar, então, um soquete com a função `CUL_SocketAccept ( )`.

**Protótipo da função:**

```
void CUCB_SocketTryAccept(dword serverSocket)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
serverSocket	Soquete anteriormente criado com <code>CUL_SocketOpenTcpServer()</code> .

Tabela 313: Parâmetro

### 12.5.7.9 CUL\_SocketAccept

A função `CUL_SocketAccept ( )` cria um novo soquete para a requisição de conexão anteriormente sinalizada com `CUCB_SocketTryAccept()`.

#### Protótipo da função:

```

dword          dword serverSocket ,
CUL_SocketAccept (    udword *pIpAddr ,
                      uword *pTcpPort )

```

#### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
serverSocket	Soquete assinalado imediatamente antes com <code>CUCB_SocketTryAccept()</code>
pIpAddr	Endereço para o qual o endereço IP do Peer deve ser copiado, ou 0 se não for o caso
pTcpPort	Endereço para o qual o número da porta TCP do Peer deve ser copiado, ou 0 se não for o caso.

Tabela 314: Parâmetros

#### Valor de retorno:

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header cut.h.

Código de erro	Descrição
Socket	Se > 0, soquete criado como novo
CUL_WRONG_SOCKET	Tipo de soquete incorreto ou soquete não existe
CUL_NO_MORE_SOCKETS	Não há mais recursos de soquete disponíveis
CUL_SOCKET_ERROR	Outros erros de Socket

Tabela 315: Valor de retorno

#### Restrições:

Se pIpAddr und pTcpPort não forem propriedade de CUT, então, CUT/CUIT são terminadas.

### 12.5.7.10 CUL\_SocketOpenTcpClient

A função `CUL_SocketOpenTcpClient()` cria um soquete do tipo TCP com porta local livre e solicita uma conexão para `destIp` e `destPort`. Soquetes sempre são operados no modo non-blocking, ou seja, esta função não gera bloqueios. Logo que a conexão estiver sido estabelecida é chamado `CUCB_SocketConnected()`.

**Protótipo da função:**

`dword CUL_SocketOpenTcpClient(udword destIp, uword destPort)`

**Parâmetros:**

A função tem os seguintes parâmetros:

Parâmetro	Descrição
destIp	Endereço IP do parceiro de comunicação
destPort	Número de porta do parceiro de comunicação

Tabela 316: Parâmetro

**Valor de retorno:**

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header cut.h.

Código de erro	Descrição
Socket number	Se > 0; os códigos de erro são < 0
CUL_NO_MORE_SOCKETS	Não há recursos disponíveis para Socket
CUL_NO_ROUTE	Não há roteamento para alcançar destIp
CUL_SOCK_ERROR	Outros erros de Socket

Tabela 317: Valor de retorno

#### 12.5.7.11 CUCB\_SocketConnected

A função `CUCB_SocketConnected()` é chamada por COM se foi estabelecida uma conexão TCP com a função `CUL_SocketOpenTcpClient()`.

**Protótipo da função:**

```
void CUCB_SocketConnected(dword socket, bool successfully)
```

**Parâmetros:**

A função tem os seguintes parâmetros:

Parâmetro	Descrição
socket	Soquete anteriormente criado com <code>CUL_SocketOpenTcpClient()</code>
successfully	TRUE, se a tentativa de conexão teve êxito, senão, FALSE

Tabela 318: Parâmetro

### 12.5.7.12 CUL\_SocketSend

A função `CUL_SocketSend()` envia a mensagem anteriormente alocada e enchida com `CULNetMessageAlloc()` como pacote TCP.

Depois do envio, a memória de mensagens `pMsg` é liberada automaticamente de novo.

Para cada envio, primeiramente precisa ser alocada da memória de mensagem com a função `CULMessageAlloc()`.

#### Protótipo da função:

```
dword CUL_SocketSend(dword socket, void *pMsg, udword size)
```

#### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
socket	Soquete anteriormente criado com <code>CUL_SocketAccept()</code> / <code>CUL_SocketOpenTcpClient()</code>
pMsg	Memória dos dados de trabalho TCP anteriormente reservada com <code>CULNetMessageAlloc()</code>
size	Quantidade de memória em Bytes, deve ser $\leq$ quantidade anteriormente alocada

Tabela 319: Parâmetros

#### Valor de retorno:

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header `cut.h`.

Código de erro	Descrição
CUL_OKAY	Mensagem enviada com êxito
CUL_WRONG SOCK	Tipo de soquete incorreto ou soquete não existe
CUL_WOULD_BLOCK	A mensagem não pode ser enviada, senão o soquete seria bloqueado
CUL_SOCKET_ERROR	Outros erros de Socket

Tabela 320: Valor de retorno

#### Restrições:

Se `pMsg` não for uma mensagem da propriedade de CUT ou se o tamanho for excessivo para `pMsg`, então CUT/CUIT são terminadas.

## 12.5.7.13 CUCB\_SocketTcpRcv

A função `CUCB_SocketTcpRcv()` é chamada por COM se os dados de trabalho do soquete estão disponíveis.

Depois de sair da função `CUCB_SocketTcpRcv()`, não é mais permitido acessar `*pMsg`.

Se os dados de trabalho são necessários também fora da função `CUCB_SocketTcpRcv()`, os dados de trabalho devem ser copiados de `*pMsg` para uma área criada para este fim.

.  
1

Se a conexão TCP for separada assincronamente (após erro ou Request do outro lado), é chamado `CUCB_SocketTcpRcv()` com `dataLength = 0`.

Com esta chamada assinala-se à CUT que deve fechar o soquete para sincronizar comunicação de novo.

#### Protótipo da função:

```
void CUCB_SocketTcpRcv( dword socket,
                        void *pMsg,
                        udword dataLength)
```

#### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
socket	Soquete pelo qual os dados de trabalho foram recebidos.
pMsg	O parâmetro <code>pMsg</code> aponta para o início dos dados de trabalho <b>sem</b> o cabeçalho Ethernet/IP/TCP.
dataLength	O comprimento dos dados de trabalho em Bytes

Tabela 321: Parâmetro

#### 12.5.7.14 CUL\_SocketClose

A função `CUL_SocketClose()` fecha um soquete anteriormente criado.

O soquete é fechado dentro de 90 segundos. A função `SocketOpen` apenas pode ser executada de novo depois de ter fechado o soquete.

**Protótipo da função:**

`dword CUL_SocketClose(dword socket)`

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
socket	Soquete anteriormente criado

Tabela 322: Parâmetros

**Valor de retorno:**

É retornado um código de erro (dword).

Os códigos de erro são definidos no arquivo Header `cut.h`.

Código de erro	Descrição
CUL_OKAY	Soquete fechado e um recurso de soquete novamente livre.
CUL_WRONG_SOCKET	Soquete não existe

Tabela 323: Valor de retorno

## 12.5.8 Timer IF

### 12.5.8.1 CUL\_GetTimeStampMS

A função `CUL_GetTimeStampMS()` fornece um tique de milissegundo. Este é adequado para implementar na CUT/CUIT temporizadores próprios. O contador é derivado do quartzo do processador COM e, assim, possui a mesma precisão.

**Protótipo da função:**

udword CUL\_GetTimeStampMS(void)

### 12.5.8.2 CUL\_GetDateAndTime

A função `CUL_GetDateAndTime()` fornece ao local de memória indicado `*pSec` os segundos depois de 1º de Janeiro de 1970, 00:00 e em `*pMsec` os milissegundos correspondentes. Os valores são comparados ao módulo CPU seguro e podem ser sincronizados externamente via SNTP de acordo com a parametrização (veja Capítulo 9).

Os valores de `CUL_GetDateAndTime()` **não** devem ser usados para medição de tempo, temporizadores etc., pois podem ser ajustados em funcionamento pela sincronização e/ou pelo usuário.

**Protótipo da função:**

void CUL\_GetDateAndTime(udword \*pSec, udword \*pMsec)

**Restrições:**

Se a memória de `pSec` ou `pMsec` não estiver no segmento de dados CUT, CUT/CUIT são terminados.



### 12.5.9 Diagnóstico

A função `CUL_DiagEntry()` insere um evento no diagnóstico de curto prazo de COM que pode ser lido pelo PADT.

**Protótipo da função:**

```
void                                udword severity,  
CUL_DiagEntry(                     udword code,  
                                   udword param1,  
                                   udword param2)
```

**Parâmetros:**

A função tem os seguintes parâmetros:

Parâmetro	Descrição
severity	O valor severity serve para a classificação do evento 0x45 ('E') == Erro, 0x57 ('W') == Atenção, 0x49 ('I') == Informação
code	O usuário define o código de parâmetro com um número livre para os respectivos eventos. Quando o evento ocorrer, o número é exibido no diagnóstico.
param1, param2	Informações adicionais sobre o evento

Tabela 324: Parâmetro

## 12.6 Funções para SEW

As seguintes funções valem apenas para os sistemas de comando HM30/ PFF-HM31 da SEW.

### 12.6.1 COM User IRQ Task

A COM User IRQ Task (CUT) precisa compartilhar a memória de código e de dados com a CUT. Possui o seu próprio Stack e como no caso do Stack da CUT, é determinado pelo SO de COM (do ponto de vista da CUIT, dinâmico) e tem o tamanho de 8 kByte. Apenas há uma CUIT no COM. Inicialmente, os IrqServices estão disabled, ou seja, após um Power On ou depois de carregar a configuração.

#### Restrições:

Da CUIT apenas podem ser chamadas as funções CUL para o Semaphore-Handling.

#### 12.6.1.1 CUCB\_IrqService

A função `CUCB_IrqService()` é chamada pelo COM depois de acionar um dos dois possíveis CAN IRQs.

Esta função é responsável por servir a fonte IRQ `devNo` do respectivo chip CAN e deve assegurar que o chip CAN retome sua requisição de IRQ novamente.

#### Protótipo da função:

```
void CUCB_IrqService(udword devNo)
```

#### Restrições:

A gestão de IRQ do processador COM é efetuada pelo SO de COM e não pode ser assumida pela função `CUCB_IrqService()`.

---

**i**

A função `CUCB_IrqService()` deve ser implementada de forma muito eficiente para minimizar latências desnecessárias de outras funções do processador de COM. Caso contrário, é possível que com carga alta do processador de COM certas funções não possam mais ser executadas, o que, p. ex., afetaria também a comunicação segura da CPU segura.

---

A biblioteca CUT permite a liberação e o desligamento do canal IRQ COM no qual o controlador CAN está ligado.

#### CUL\_IrqServiceEnable

A função `CUL_IrqServiceEnable()` libera o canal COM-IRQ para o controlador CAN selecionado `devNo`. A partir de agora, CAN-IRQs acionam a chamada da CUT-IRQ-Task.

#### Protótipo da função:

```
void CUL_IrqServiceEnable(udword devNo)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
devNo	1 = CAN Controller A 2 = CAN Controller B

Tabela 325: Parâmetro

**Restrições:**

Se para devNo são usados valores não igual a 1 ou 2, então, CUIT/CUT são terminadas.

### 12.6.1.2 CUL\_IrqServiceDisable

A função `CUL_IrqServiceDisable()` bloqueia o canal COM-IRQ para o controlador CAN *devNo*. A partir de agora, CAN-IRQs não acionam mais a chamada da CUT-IRQ-Task. Porém, processamento de IRQ ainda não encerrados são executados ainda.

**Protótipo da função:**

```
void CUL_IrqServiceDisable(udword devNo)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
devNo	1 = CAN Controller A 2 = CAN Controller B

Tabela 326: Parâmetro

**Restrições:**

Se para *devNo* são usados valores desigual 1 ou 2, então, CUIT/CUT são terminadas.

### 12.6.1.3 CUL\_DeviceBaseAddr

A função `CUL_DeviceBaseAddr()` fornece o endereço base de 32 Bit dos controladores CAN.

**Protótipo da função:**

```
void* CUL_DeviceBaseAddr(udword devNo)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
devNo	1 = CAN Controller A 2 = CAN Controller B

Tabela 327: Parâmetro

**Restrições:**

Se para *devNo* são usados valores desigual 1 ou 2, então, CUIT/CUT são terminadas.

### 12.6.2 NVRam-IF

A CUT e a CUIT podem escrever e ler a área disponível.

O NVRam disponível para estas funções possui o tamanho de 9 kBytes.

---

**i**

O COM **não** assegura a consistência dos dados no caso de queda da tensão de operação durante um acesso e também não durante acessos a partir de duas tarefas simultaneamente.

---

#### 12.6.2.1 CUL\_NVRamWrite

A função `CUL_NVRamWrite()` escreve dados no NVRam.

##### Protótipo da função:

`void CUL_NVRamWrite(udword offset, void *source, udword size)`

##### Parâmetros:

A função tem os seguintes parâmetros:

Parâmetro	Descrição
offset	Na área do NVRam, valores válidos são 0–9215
source	Área de memória no segmento de dados CUT que deve ser copiada para o NVRam.
size	Quantidade de Bytes que devem ser copiados

Tabela 328: Parâmetro

##### Restrições:

No caso de parâmetros inválidos, CUT e CUIT são terminadas, ou seja com:

- `offset ≥ 9216`
- `offset+size > 9216`
- `source` não no segmento de dados CUT
- `source+size` não no segmento de dados CUT

#### 12.6.2.2 CUL\_NVRamRead

A função `CUL_NVRamRead()` lê dados do NVRam.

##### Protótipo da função:

`void CUL_NVRamRead(udword offset, void *destination, udword size)`

**Parâmetros:**

A função tem os seguintes parâmetros:

Parâmetro	Descrição
offset	Na área do NVRam, valores válidos são 0–9215
destination	Área de memória no segmento de dados CUT que deve ser copiada para o NVRam.
size	Quantidade de Bytes que devem ser copiados

Tabela 329: Parâmetro

**Restrições:**

No caso de parâmetros inválidos, CUT e CUIT são terminadas:

- $\text{offset} \geq 9216$
- $\text{offset} + \text{size} > 9216$
- destination não no segmento de dados CUT
- $\text{destination} + \text{size}$  não no segmento de dados CUT

### 12.6.3 Semaphore-IF

A CUT e a CUIT possuem em conjunto **um** Semaphor para a sincronização de processo.

Os dados das funções CUCB e CUL que são usadas em conjunto com a função `CUCB_IrqService()` devem ser protegidas por um Semaphor. Isso garante a consistência dos dados usados em conjunto com a função `CUCB_IrqService()`.

#### 12.6.3.1 CUL\_SemaRequest

A função `CUL_SemaRequest()` solicita o Semaphore da CUT/UIT.

Se o Semaphore estiver

- estiver livre, a função retorna com o valor `pContext`.
- não estiver livre, a tarefa que chamou será bloqueada até o Semaphor ser liberado por uma outra tarefa e retorna com o valor `pContext`.

O contexto referenciado pelo parâmetro `pContext` é apenas usado pelas funções CUL para a tarefa que está chamando e não pode ser alterado entre Request e Release.

**Protótipo da função:**

```
void CUL_SemaRequest(udword *pContext)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
pContext	Apenas é usado por CUL dentro da tarefa que está chamando. O contexto é retornado via <code>pContext</code> e deve ser novamente indicado com a função <code>CUL_SemaRelease()</code> .

Tabela 330: Parâmetro

**Restrições:**

Se a quantidade de recursões admissíveis for ultrapassada, CUT/CUIT são terminadas.

Se a CUT estiver bloqueada por um Semaphor, não são mais executadas CUCB\_'s, exceto CUCB\_IrqService().

**12.6.3.2 CUL\_SemaRelease**

A função CUL\_SemaRelease() libera novamente o Semaphor definido por \*pContext.

**Protótipo da função:**

```
void CUL_SemaRelease(udword *pContext)
```

**Parâmetros:**

A função tem o seguinte parâmetro:

Parâmetro	Descrição
pContext	Com o mesmo valor para *pContext como ele foi escrito por CUL_SemaRequest ou CUL_SemaTry.

Tabela 331: Parâmetro

**Restrições:**

Se Release é chamada mais frequentemente do que Request/Try, CUT/CUIL são terminadas.

### 12.6.3.3 CUL\_SemaTry

A função `CUL_SemaTry( )` tenha solicitar o Semaphor da CUT/CUIT.

Se o Semaphore estiver

- livre, a função retorna com TRUE e ocupa o Semaphor.
- ocupado, a função retorna com FALSE e não ocupa o Semaphor.

A udword referenciada pelo `pContext` é apenas usado pelas funções CUL para a tarefa que está chamando e não pode ser alterado entre Request e Release.

#### Protótipo da função:

```
bool CUL_SemaTry(udword *pContext)
```

#### Parâmetros:

A função tem o seguinte parâmetro:

Parâmetro	Descrição
<code>pContext</code>	Apenas é usado por CUL dentro da tarefa que está chamando.

Tabela 332: Parâmetro

#### Valor de retorno:

É retornado um código de erro (udword).

Os códigos de erro são definidos no arquivo Header `cut.h`.

Valor de retorno	Descrição
TRUE	Semaphor pôde ser ocupado.
FALSE	Semaphor não pôde ser ocupado.

Tabela 333: Valor de retorno

O contexto é retornado via `pContext` e deve ser novamente indicado com a função `CUL_SemaRelease( )`.

#### Restrições:

Se a quantidade de recursões admissíveis for ultrapassada, CUT/CUIT são terminadas.

Se a CUT estiver bloqueada por um Semaphor, não são mais executadas CUCB\_'s, exceto `CUCB_IrqService( )`.

## i

As funções `CUL_SemaTry( )` e `CUL_SemaRequest( )` podem ser chamadas sem bloqueio mesmo se a tarefa que está chamando já ocupou o Semaphor; apenas que neste caso deve ocorrer o mesmo número de `CUL_SemaReleases`, até o Semaphor ficar livre de novo. A recursão permite no mínimo 32000 passos. Se mais passos são possíveis depende da respectiva versão de COM.

### 12.6.4 COM-IO-IF (Só HM 30)

Interface para a operação das E/S padrão de COM. A filtragem dos canais DI não é parametrizável; ao invés disso, sempre estão disponíveis valores filtrados e não filtrados.

```
struct CUL_IoBufferIn {
    uword mDIRaw;           //valores DI não filtrados,
                           //com Bit[0]=DI[1]...Bit[11]=DI[12]
                           //Bit[12]=DI[13]/DO[1]...Bit[15]=DI[16]/DO[4]

    uword mDIFilter;        //valores DI filtrados,
                           //com Bit[0]=DI[1]...Bit[15]=DI[16]
}CUL_IO_BUFFER_IN;

struct CUL_IoBufferOut {
    uword mDO;              // valor D 1=ON, 0=OFF,
                           // com Bit[0]=DO[1]...Bit[3]=DO[4]
}CUL_IO_BUFFER_OUT; // compatível a mDIDODir e Bit[4].Bit[15]=0

struct CUL_IoBufferCfg {
    uword mDIDODir;         // valor 1=saída, 0=entrada,
                           // com Bit[0]=DI/DO[1]...Bit[3]=DI/DO[4]
                           // Bit[4..15] indefinidos, devem ser 0
} CUL_IO_BUFFER_CFG;
```

Os canais DI/DO parametrizáveis são os canais 1 a 4. Os números de canal valem tanto para DI quando para DO nas estruturas acima. Se um canal DI/DO estiver parametrizado como saída, então, seu mDIRaw e mDIFilter não são válidos.

#### 12.6.4.1 CUL\_IORead (Só HM 30)

Lê as entradas. As características das DI ou dos valores filtrados / crus são definidas pela especificação HW.

##### Protótipo da função:

```
void CUL_IORead(CUL_IO_BUFFER_IN *pIN)
```

##### Restrições:

Se o ponteiro transmitido não estiver na propriedade da CUT, então, CUT/CUIT são terminadas.



#### 12.6.4.2 CUL\_IOWrite (Só HM 30)

Escreve as saídas. Antes de chamar `CUL_IOWrite()` pela primeira vez, são emitidos valores 0 nas saídas.

**Protótipo da função:**

```
void CUL_IOWrite(CUL_IO_BUFFER_OUT *pOUT)
```

**Restrições:**

Se o ponteiro transmitido não estiver na propriedade da CUT, então, CUT/CUIT são terminadas.

#### 12.6.4.3 CUL\_IOConfigure (Só HM 30)

A função `CUL_IOConfigure()` configura as entradas/saídas.

Antes de chamar `CUL_IOConfigure()` pela primeira vez, os canais DI/DO estão definidos como entrada. `CUL_IOConfigure()` pode ser chamada várias vezes para reconfigurar os canais DI/DO.

Canais DI/DO cuja parametrização muda de

- saída para entrada não emitem mais energia,
- entrada para saída emitem o valor anteriormente definido com `CUL_IOWrite()`.

As funções `COM_IO` também podem ser chamadas várias vezes por CUIT/CUT durante uma chamada CUCB.

**Protótipo da função:**

```
void CUL_IOConfigure(CUL_IO_BUFFER_CFG *pCFG)
```

**Restrições:**

Se o ponteiro transmitido não estiver na propriedade da CUT, então, CUT/CUIT são terminadas.

## 12.7 Instalação do ambiente de desenvolvimento

Neste Capítulo é descrita a instalação do ambiente de desenvolvimento e a criação de uma ComUserTask.

O ambiente de desenvolvimento encontra-se no CD de instalação, veja Capítulo 12.2.

### 12.7.1 Instalação do ambiente Cygwin

O ambiente Cygwin é necessário pois as ferramentas de compilação GNU C apenas são executáveis no ambiente Cygwin.

O ambiente Cygwin deve ser instalado sob Windows 2000/XP/Vista.

**i**

Observar os requisitos para instalação no Capítulo 12.2. **Desativar o anti-vírus** no PC onde Cygwin deve ser instalado, para evitar problemas durante a instalação de Cygwin.

Executar os seguintes passos para instalar o ambiente Cygwin:

#### Iniciar o programa de setup para a instalação do Cygwin:

1. Copiar o arquivo de instalador Cygwin `cygwin-1.7.5-1` do CD de instalação para o seu HD local (e.g., drive `C:\`).
2. Abrir no Windows-Explorer a pasta Cygwin  
`C:\ cygwin-1.7.5-1`.
3. Iniciar a instalação do Cygwin com um clique duplo no arquivo **setup-2.697.exe**.
4. Clicar na janela de diálogo Cygwin no botão **Next** para executar o setup.

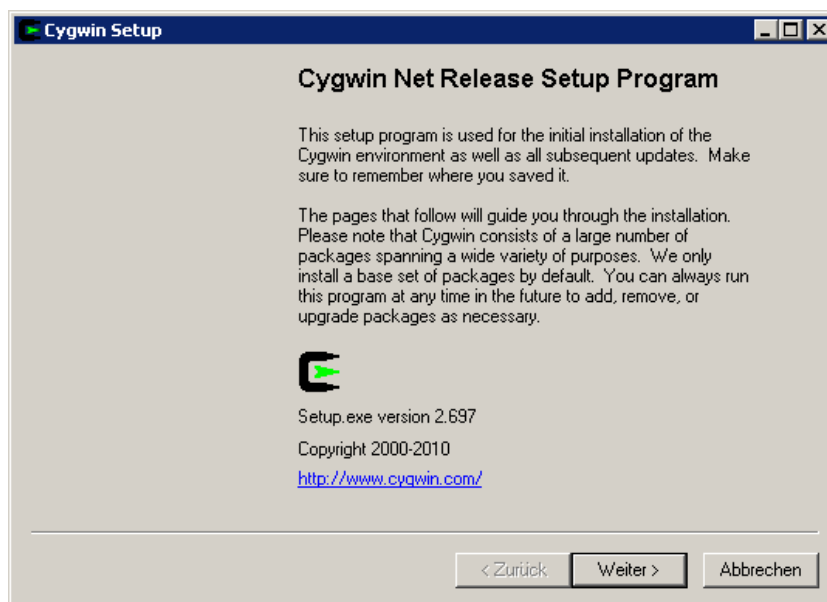


Figura 92: Diálogo de configuração do Cygwin, *Cygwin Setup*

O diálogo “**Disable Virus Scanner**” aparece se o anti-vírus não foi desativado.

Executar este passo para desativar o anti-vírus para a instalação do Cygwin.

---

i

Desativar o anti-vírus antes da instalação Cygwin, pois dependendo do anti-vírus usado pode ocorrer que esta janela não apareça, apesar do anti-vírus estar rodando.

---

1. Selecionar **Disable Virus scanner** para evitar problemas causados pelo anti-vírus durante a instalação.
2. Clicar no botão **Next** para confirmar a introdução.

**Selecionar no diálogo *Choose Installation Type* a origem de instalação do Cygwin:**

1. Selecionar como origem da instalação **Install from Local Directory**.
2. Clicar no botão **Next** para confirmar a introdução.

**Selecionar no diálogo *Choose Installation Directory* o destino de instalação do Cygwin:**

1. Indicar o diretório para o qual Cygwin deve ser instalado.
2. Aceitar todos os demais ajustes padrão do diálogo.
3. Clicar no botão **Next** para confirmar a introdução.

**Selecionar no diálogo *Select Local Package Directory* o arquivo de instalação do Cygwin:**

1. Indicar no campo *Local Package Directory* o arquivo de instalação do Cygwin no qual se encontram os dados de instalação.
2. Clicar no botão **Next** para confirmar a introdução.

**Selecionar no diálogo *Select Packages* todos os pacotes para a instalação:**

1. Selecionar o botão de opção **Curr**.
2. Clicar no campo de exibição várias vezes lentamente na opção de instalação ao lado de **All** até **Install** para a instalação completa de todos os pacotes seja exibido (cerca de 1,86 MB de memória livre necessária para a instalação).

---

i

Assegurar que atrás de cada pacote apareça **Install**.  
Se os pacotes não são instalados completamente, funções importantes estão faltando para compilar o código C da CUT!

---

3. Clicar no botão **Next** para confirmar a introdução.

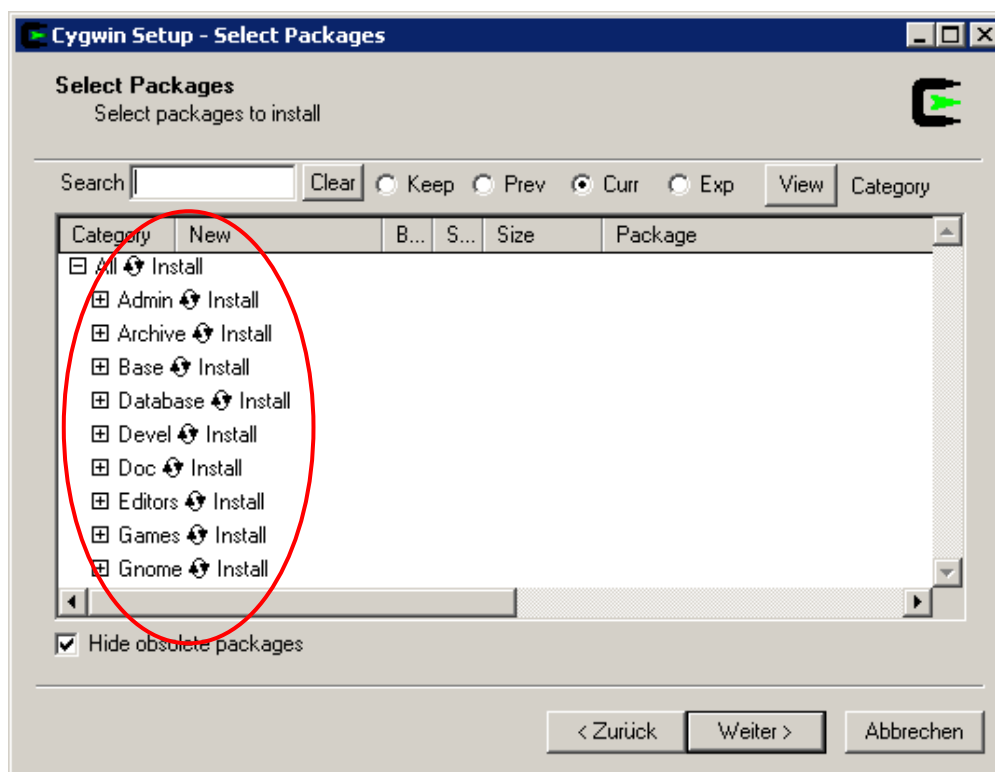


Figura 93: Diálogo de configuração do Cygwin, *Select Packages*

**Fechar a instalação do Cygwin com as seguintes entradas:**

1. Selecionar a entrada no **Menu Iniciar**.
2. Selecionar a entrada **Desktop-Icon**.
3. Clicar no botão **Finish** para finalizar a instalação do Cygwin.

Comandos Cygwin	Descrição
<b>cd (nome de diretório)</b>	Mudar o diretório
<b>cd.</b>	Mudar para o diretório superior
<b>ls -l</b>	Exibir todos os arquivos de um diretório
<b>help</b>	Visão geral dos comandos Bash Shell

Tabela 334: Comandos no Cygwin (Bash Shell)

### 12.7.2 Instalação do compilador GNU

**Executar os seguintes passos para instalar o compilador GNU:**

1. Abrir no Windows-Explorer o diretório do CD de instalação.
2. Clicar duas vezes no arquivo zipado `gcc-ppc-v3.3.2_binutils-v2.15.zip`.
3. Extrair todos os arquivos para o diretório Cygwin (p. ex., `C:\cygwin\...`).  
O compilador GNU é extraído para o diretório Cygwin, dentro de **gcc-ppc**.

4. Introduzir no sistema de comando as variáveis de ambiente:
  - Abrir as características do sistema pelo menu Iniciar do Windows **Settings->System Control->System**.
  - Selecionar o registro **Advanced**.
  - Clicar no botão Environment Variables.
  - Selecionar no campo *System Variables* a variável de sistema **Path** e complementar a variável de sistema com a entrada: `C:\cygwin\gcc-ppc\bin`.

Copiar do CD de instalação a pasta `cut_src` para o diretório `root`.

A pasta `cut_src` contém todos os diretórios “include” e “lib” necessários para a criação de uma ComUserTask. O arquivo de código fonte `cut_cbf.c` fornecido está por padrão no diretório `...\cut_src\cutapp\`.

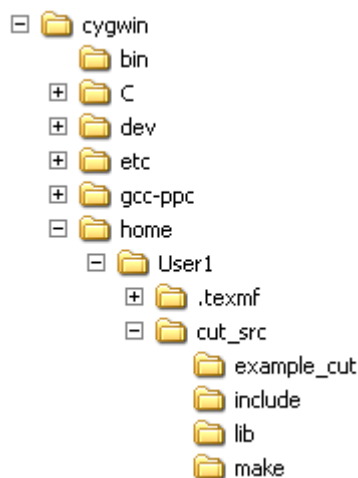


Figura 94: Árvore de estrutura do Cygwin

**i**

Se o diretório `root` não foi criado automaticamente, criar o diretório `root` com o Windows-Explorer an (e.g. `C:\cygwin\home\User1`).

Se quiser criar um outro diretório `root` para Cygwin Bash Shell precisa complementar o arquivo Batch `cygwin.bat` com o comando `set Home`.

```
@echo off
```

```
C:
```

```
chdir C:\cygwin\bin
```

```
set Home=C:\User1
```

```
bash --login -i
```

Figura 95: Arquivo Batch *Cygwin.bat*

**i**

Para gerar código executável para o programa `example_cut` fornecido no DVD, veja Capítulo 12.8.2.4.

## 12.8 Criar novo projeto CUT

Este Capítulo mostra como um novo projeto CUT é criado e quais arquivos devem ser adaptados.

i

O projeto CUT **example\_cut** encontra-se pronto e adaptado no CD CUT.

Ao criar outros projetos CUT novos, recomenda-se criar para cada projeto CUT um novo diretório abaixo de ...`\cut_src\`.

### Exemplo:

Para fins de teste é criado um diretório *example\_cut*, a fonte em C se chama **example\_cut.c**, o arquivo ldb criado no diretório *make* neste caso se chama **example\_cut.ldb**.

**Criar para uma nova ComUserTask a pasta *example\_cut*.**

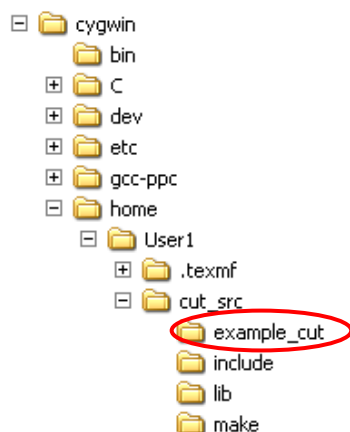


Figura 96: Árvore de estrutura do Cygwin

1. Copiar os arquivos
  - cutapp.c
  - cutapp.mke
  - makefile
 do diretório *cutapp* para o diretório *example\_cut*.
2. Os nomes de arquivo devem ser renomeados para nomes Loadable (p. ex., de **cutapp** para **example\_cut**).

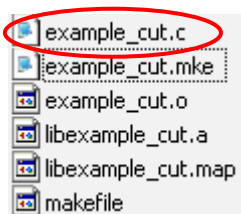


Figura 97: Arquivo de código C na pasta *example\_cut*

As alterações no arquivo mke e no makefile devem ser efetuadas para um projeto novo, como descrito nos próximos capítulos.

### 12.8.1 CUT Makefiles

A configuração dos CUT Makefiles para diferentes Sourcefiles e arquivos ldb  
No total devem ser adaptados três Makefiles, como descrito nos seguintes parágrafos.

#### 12.8.1.1 Makefile com a extensão “.mke”

O arquivo mke encontra-se no respectivo diretório de código fonte  
p. ex., *cut\_src\example\_cut\example\_cut.mke*.

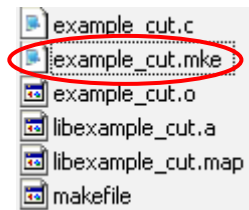


Figura 98: Arquivo mke na pasta example\_cut

#### Efetuar no arquivo mke as seguintes alterações:

1. À variável **module** deve ser atribuído o mesmo nome Loadable que ao arquivo.mke (p. ex., *example\_cut*).
2. À variável **c\_sources** podem ser atribuídos um ou vários arquivos C que são necessários para a criação do código de destino (arquivo Loadable).

```
#####
#
# make file (DOS/NT)
# $Id: cutapp.mke 58869 2005-10-11 12:35:46Z es_fp $
#####
#
# assign name of module here (e.g. nl for NetworkLayer)
module= example_cut
#
# assign module sources here
sources=

c_sources= $(module).c
asm_sources=
```

Figura 99: Arquivo mke a partir da linha 1

## Makefile

O arquivo makefile encontra-se no respectivo diretório de código fonte  
p. ex., cut\_src\example\_cut\makefile.

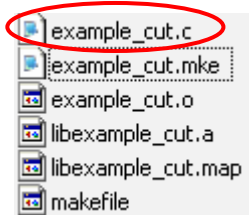


Figura 100: makefile na pasta *example\_cut*

### Efetuar no arquivo makefile as seguintes alterações:

1. Puxar a linha Include para o arquivo mke para cima e alterar o arquivo.mke para o nome atual.
2. Ampliar a chamada Make com as duas variáveis **SUBMOD\_DIRS** e **CUT\_NAME**.

all\_objects:

**include example\_cut.mke**

cut:

\$(MAKE) -C./make elf **SUBMOD\_DIRS=cut\_src/\$(module)** **CUT\_NAME=\$(module)**

all\_objects: \$(c\_objects) \$(asm\_objects) \$(objects)

Figura 101: makefile a partir da linha 34

### 12.8.1.2 Makefile com a extensão “makeinc.inc.app”

Como alteração única para este e todos os demais projetos CUT, o nome do Loadable CUT é tornado alterável mediante uma variável Make.

O arquivo makeinc.inc.app encontra-se no diretório cut\_src,  
p. ex. cut\_src\makeinc.inc.app.

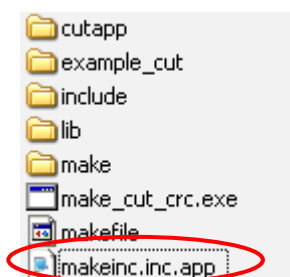


Figura 102: makeinc.inc.app arquivo na pasta *example\_cut*



**Efetuar no arquivo makeinc.inc.app as seguintes alterações:**

1. Complementar o arquivo com a variável **CUT\_NAME**

```
all : lib$(module).$(LIBEXT)
@echo 'did make for module ["lib$(module).$(LIBEXT)"]

lib$(module).$(LIBEXT) : $(objects) $(c_objects) $(asm_objects) $(libraries)

SUBMOD2_LIBS=$(foreach lib,$(SUBMOD2_LIBS),./$(lib))

CUT_NAME=cut

makeAllLibs:
$(MAKE) -C./cut_src cut_src

makeLoadable:
@echo; \
BGTYPE=" $(CUT_NAME)"; \
if [ ! -f $$BGTYPE.map ] ; then \
echo "Error: MAP file $$BGTYPE.map does not exist"; \
exit 1; \
fi; \
OS_LENGTH=$(gawk '/__OS_LENGTH/ {print substr($$1,3,8)}' $$BGTYPE.map); \
echo; \
$(OBJCOPY) --strip-all --strip-debug -O binary $$BGTYPE.elf $$BGTYPE.bin;\
echo; \
echo "Building C3-Loadable-Binary ..."; \
$(MCR)C) $$BGTYPE.bin 0 $$OS_LENGTH $$OS_LENGTH $$BGTYPE.ldb; \
echo; \

$(CUT_NAME).elf: makeAllLibs $(SUBMOD2_LIBS)

elf:
@echo; test -f section.dld && $(MAKE) $(CUT_NAME).elf && $(MAKE) makeLoadable \
|| { echo "ERROR: Wrong subdir. Please invoke elf target only from make/ subdirectory."
&& echo && false ; } ;

# end of file: makeinc.inc
```

Figura 103: makeinc.inc.app a partir da linha 247

## 12.8.2 Editar código fonte em C

**Efetuar os seguintes passos para abrir o arquivo de código fonte:**

1. Abrir o diretório de projeto *cut\_src\example\_cut* o qual criou e configurou nos passos anteriores.
2. Abrir o arquivo de código fonte com a extensão **.c** com um editor (p. ex., Notepad – Bloco de notas).

### 12.8.2.1 Configurar variável de entrada e saída

**Efetuar os seguintes passos para configurar as variáveis de entrada e saída no arquivo de código fonte:**

1. O tamanho de dados das variáveis que devem ser criadas no registro **Data Outputs** no SILworX deve ser criado no arquivo de código fonte no array **CUT\_PDI[X]**.
2. O tamanho de dados das variáveis que devem ser criadas no registro **Data Inputs** no SILworX deve ser criado no arquivo de código fonte no array **CUT\_PDO[X]**.

### 12.8.2.2 Função inicial na CUT

A função C `void CUCB_TaskLoop (udword mode)` é a função inicial e é chamada pelo programa de aplicação ciclicamente.

### 12.8.2.3 Código de exemplo “example\_cut.c”

O seguinte código C copia o valor da entrada **CUT\_PDI[0]** para a saída **CUT\_PDO[0]** e retorna o valor sem alteração ao programa de aplicação SILworX.

i

O código C **example\_cut.c** encontra-se no CD de instalação.

```
/* Example for the CUT implemetation */
#include "include/cut_types.h"
#include "include/cut.h"
#ifdef __cplusplus
extern "C" {
#endif
/*****
/* SILworX Output Records (CPU->COM) */
uword CUT_PDI[1] __attribute__ ((section("CUT_PD_IN_SECT"), aligned(1)));
/* SILworX Input Records (COM->CPU) */
uword CUT_PDO[1] __attribute__ ((section("CUT_PD_OUT_SECT"), aligned(1)));
*****/

/* Callback function for starting the CUT */
void CUCB_TaskLoop(udword mode)
{
    if (CUT_PDI[0] > CUT_PDO[0]) /*This is executed only, if the          */
    {                               /*SILworX application program      */
                                    /*was processed.                  */
                                    /*The SILworX application program*/
                                    /*adds the value 1 to CUT_PDO[0] and */
                                    /*writes the result into CUT_PDI[0]   */
    CUT_PDO[0] = CUT_PDI[0]; /*Copies the value from input CUT_PDI[0] */
                            /*into output CUT_PDO[0] of the SPS        */
    if (CUT_PDO[0] == 65535)
        {CUT_PDO[0] = 0;}
    }
}
/*****/
```

Figura 104: Legenda da figura

```

/*****
void CUCB_AscRcvReady(udword comId)
{
    CUL_DiagEntry(0x49, 1, comId, 0);
}
*****/
/*****
void CUCB_AscSendReady(udword comId)
{
    CUL_DiagEntry(0x49, 2, comId, 0);
}
*****/
/*****
void CUCB_SocketTryAccept(dword serverSocket)
{
    CUL_DiagEntry(0x49, 3, serverSocket, 0);
}
*****/
/*****
void CUCB_SocketConnected(dword socket, bool Okay)
{
    CUL_DiagEntry(0x49, 4, socket, Okay);
}
*****/
/*****
void CUCB_SocketTcpRcv(dword socket, void *pMsg, udword dataLength)
{
    CUL_DiagEntry(0x49, 5, socket, dataLength);
}
*****/
/*****
void CUCB_SocketUdpRcv(dword socket, void *pMsg, udword packetLength,
                      udword dataLength)
{
    CUL_DiagEntry(0x49, 6, socket, dataLength);
}
*****/
/*****
void CUCB_IrqService(udword devNo)
{
    CUL_DiagEntry(0x49, 7, devNo, 0);
}
*****/

#ifdef __cplusplus
} /* end extern "C" */
#endif

/* end of file */

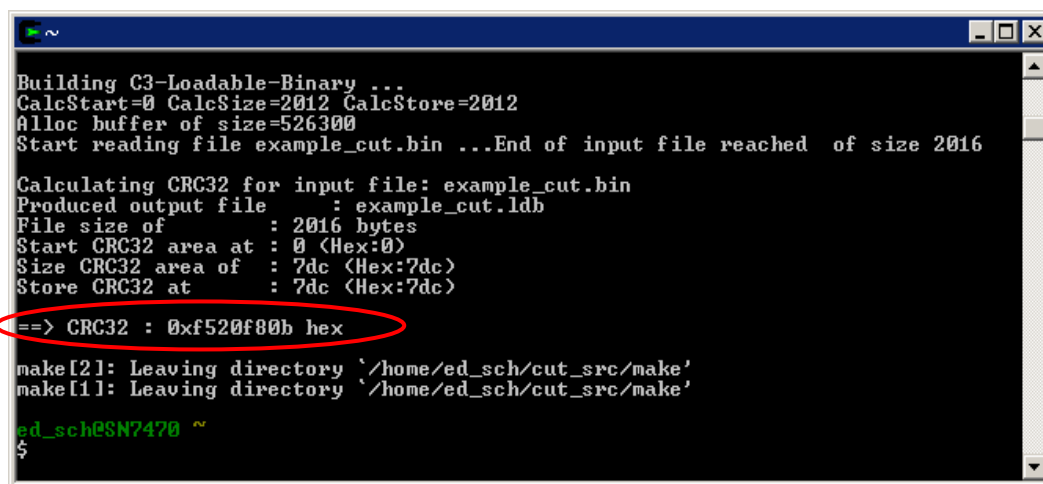
```

Figura 105: Exemplo de código C

#### 12.8.2.4 Criação do código executável (ldb-Datei)

**Executar os seguintes passos para a criação do código executável (arquivo ldb):**

1. Iniciar **Cygwin Bash Shell**.
2. Mudar par ao diretório `.../cut_src/example_cut/`
3. Iniciar a geração de código com a introdução  
`make cut_himax` para HImax  
`make cut_l2` para HIMatrix L2  
`make cut_l3` para HIMatrix L3.  
O arquivo binário **cut.ldb** é criado no diretório `/cut_src/make/` automaticamente.
4. Se o CRC32 foi criado, então, também código executável foi criado (veja marcação verde em Figura 106).



```
Building C3-Loadable-Binary ...
CalcStart=0 CalcSize=2012 CalcStore=2012
Alloc buffer of size=526300
Start reading file example_cut.bin ...End of input file reached of size 2016

Calculating CRC32 for input file: example_cut.bin
Produced output file      : example_cut.ldb
File size of              : 2016 bytes
Start CRC32 area at      : 0 (Hex:0)
Size CRC32 area of       : 7dc (Hex:7dc)
Store CRC32 at           : 7dc (Hex:7dc)
==> CRC32 : 0xf520f80b hex
make[2]: Leaving directory `/home/ed_sch/cut_src/make'
make[1]: Leaving directory `/home/ed_sch/cut_src/make'

ed_sch@SN7470 ~
$
```

Figura 106: Cygwin Bash Shell

Este código executável (arquivo ldb) deve ser carregado para o projeto como ComUserTask (veja Capítulo 12.8.3).

### 12.8.3 Integrar ComUserTask ao projeto

Executar no SILworX os seguintes passos para integrar a ComUserTask ao projeto:

#### 12.8.3.1 Criar ComUserTask

**Assim cria-se uma nova ComUserTask:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. No menu de contexto de protocolos, seleccionar **New, ComUserTask** para adicionar uma nova ComUserTask.
3. No menu de contexto **ComUserTask Properties**, seleccionar o COM-Module.  
Os ajustes padrão podem ser mantidos para a primeira configuração.



Sempre só uma ComUserTask por recurso pode ser criada.

---

#### 12.8.3.2 Carregar o código de programa no projeto

**Assim carrega-se uma nova ComUserTask para o projeto:**

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols**.
2. Clicar com o botão direito em **ComUserTask** e seleccionar no menu de contexto **Load User Task**. Abrir o diretório.../cut\_src/make/.
3. Seleccionar o **arquivo ldb** que deve ser executada na ComUserTask.



Ao carregar de novo o código executável (arquivo ldb), é possível transferir novas versões do arquivo ldb. O conteúdo do arquivo ldb não é verificado ao carregar para detectar se está correto. A seguir, o arquivo ldb é compilado em conjunto com a combinação de recursos no projeto e pode ser carregado ao sistema de comando. Se o arquivo ldb for alterado, o projeto deve ser compilado e carregado de novo.

---

#### Conectar variáveis com CUT

O usuário pode definir uma comunicação de dados de processo não direcionada à segurança entre módulo CPU seguro e COM (CUT) não segura. Neste caso podem ser no máximo 16kByte em cada duração.

Criar as seguintes variáveis globais:

Variável	Tipo
COM_CPU	UINT
CPU_COM	UINT

### 12.8.3.3 Conectar variáveis de processo

#### Variáveis de processo na ComUserTask:

1. Clicar com o botão direito em **ComUserTask** e selecionar no menu de contexto **Edit**.
2. Selecionar no diálogo **Edit** o registro **Process Variables**.

#### Variáveis de saída (CPU->COM)

No registro **Output Variables** são introduzidas as variáveis que devem ser transmitidas de CPU para COM.

Name	Tipo	Offset	Variáveis globais
CPU_COM	UINT	0	CPU_COM

Tabela 335: Variáveis de saída (CPU->COM)

1. Na seleção de objetos, puxar a Global Variables para o envio com Drag&Drop para o registro **Output Variables**.
2. Clicar com o botão direito em um espaço vazio na área **Output Variables** para abrir o menu de contexto.
3. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

#### Variáveis de entrada (COM->CPU)

No registro **Input Variables** são introduzidas as variáveis que devem ser transmitidas de COM para CPU.

Name	Tipo	Offset	Variáveis globais
COM_CPU	UINT	0	COM_CPU

Tabela 336: Variáveis de entrada (COM->CPU)

1. Na seleção de objetos, puxar a Global Variables para a recepção com Drag&Drop para o registro **Input Variables**.
2. Clicar com o botão direito em um espaço vazio na área **Input Variables** para abrir o menu de contexto.
3. Selecionar no menu de contexto **New Offsets** para que os offsets das variáveis sejam gerados novamente.

#### Verificar a configuração ComUserTask:

1. Abrir na árvore de estrutura **Configuration, Resource, Protocols, ComUserTask**.
2. Clicar com o botão direito em **Verification** para verificar a configuração CUT.
3. Verificar cuidadosamente as entradas no **logbook**, corrigir se for necessário.

### 12.8.3.4 Criar o programa de aplicação SILworX

#### Assim cria-se o programa de aplicação SILworX

1. Selecionar na árvore de estrutura **Configuration, Resource** e no menu de contexto, **Edit**.
2. Puxar da seleção de objetos as variáveis globais **COM\_CPU** e **CPU\_COM** via Drag & Drop da seleção de objetos para o campo de símbolos.
3. Criar o programa de aplicação SILworX como mostrado na seguinte figura.

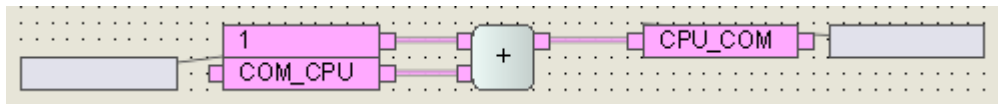


Figura 107: SILworX Programm Editor

#### Assim configura-se o Schedule Interval [ms]

1. Clicar com o botão direito em **ComUserTask** e selecionar no menu de contexto **Properties**.
2. No campo de introdução de dados *Schedule Interval [ms]*, colocar em que intervalos a ComUserTask deve ser chamada.

**i**

A configuração da conexão da ComUserTask deve ser novamente compilada com o programa de aplicação do recurso e transmitida para os sistemas de comando antes de se tornar efetiva para o sistema HIMax.

#### Assim a ComUserTask pode ser testada com um teste online

1. Selecionar na árvore de estrutura **Configuration, Resource, Program**.
2. Clicar com o botão direito em **Program** e selecionar no menu de contexto **Online**. Efetuar o login de sistema.

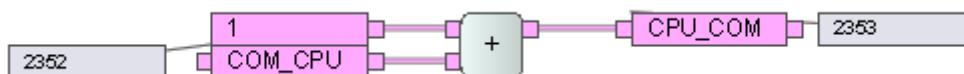


Figura 108: Teste on-line SILworX

#### Função do programa de aplicação:

O programa de aplicação SILworX adiciona ao sinal **COM\_CPU** (entradas de dados) o valor **1** e transmite o resultado ao sinal **CPU\_COM** (saídas de dados).

Na próxima chamada CUT (Schedule Interval [ms]) é transferido o sinal **CPU\_COM** à função CUT (veja código de exemplo, Capítulo 12.8.2).

A ComUserTask recebe o sinal **CPU\_COM** e retorna o valor com o sinal **COM\_CPU** inalterado.

#### 12.8.4 Erro ao carregar uma configuração com CUT Problemas de tempo de execução (p. ex., COM-User-Task em loop infinito):

##### **Causa para problemas de tempo de execução:**

Se no respectivo código fonte CUT foi programado um loop que roda por muito tempo, isso pode levar o processado COM a “empacar”.

Isso tem como resultado que não é mais possível estabelecer uma conexão ao sistema de comando e não é mais possível excluir a configuração de recursos.

**Solução:** Reset do módulo de comunicação HIMax ou do sistema de comando HIMatrix:

- Executar um reset do módulo de comunicação na visão Online do Editor de hardware mediante a função **Maintenance/Service, Module (Restart)** (reset do HIMatrix via tecla Reset, veja folha de dados do sistema de comando).
- Criar uma nova CUT (de preferência sem erro de tempo de execução, loop infinito).
- Carregar a CUT (arquivo ldb) para o projeto.
- Gerar o código.
- Carregar o código para o sistema de comando.



## 13 Informações gerais

### 13.1 Configuração dos blocos funcionais

Os protocolos de barramento de campo e os blocos funcionais correspondentes rodam no módulo COM do sistema de comando HIMax. Por isso, estes blocos funcionais devem ser criados na árvore de estrutura do SILworX em **Configuration, Resource, Protocols...**

Para controlar estes blocos funcionais no módulo COM, é possível criar blocos funcionais no programa de aplicação pelo SILworX (veja Capítulo 13.1.1), que podem ser usados como blocos funcionais padrão.

A conexão dos blocos funcionais no programa de aplicação do SILworX com os blocos funcionais correspondentes na árvore de estrutura do SILworX ocorre através de variáveis conjuntas. Estas devem ser anteriormente criadas pelo usuário no editor de variáveis.

#### 13.1.1 Aquisição das bibliotecas de blocos funcionais

As bibliotecas de blocos funcionais para PROFIBUS DP e TCP Send/Receive devem ser adicionadas ao projeto pela função *Restore..* (menu de contexto do projeto).

A biblioteca de blocos funcionais pode ser obtida pelo suporte da HIMA.

**Tel.: +49-(0)6202-709    -185**  
**-259**  
**-261**

**E-Mail: support@hima.com**

#### 13.1.2 Configuração dos blocos funcionais no programa de aplicação

Os blocos funcionais necessários podem ser copiados com Drag&Drop para o programa de aplicação. Configurar as entradas e saídas de acordo com a descrição do respectivo bloco funcional.

##### Parte superior do bloco funcional

A parte superior do bloco funcional corresponde à interface do usuário pela qual o bloco funcional é controlado pelo programa de aplicação.

Aqui são atribuídas às variáveis de sistema que são utilizadas no programa de aplicação. O prefixo "A" significa "Application" – Aplicação.

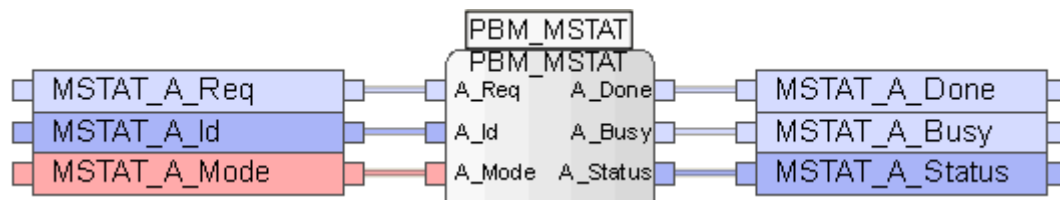


Figura 109: Bloco funcional PNM\_MSTST (parte superior)

### Parte inferior do bloco funcional

A parte inferior do bloco funcional representa a conexão ao bloco funcional (na árvore de estrutura de SILworX).

Aqui são conectadas as variáveis que devem ser conectadas ao bloco funcional na árvore de estrutura do SILworX. O prefixo “F” significa “Field” – Campo.

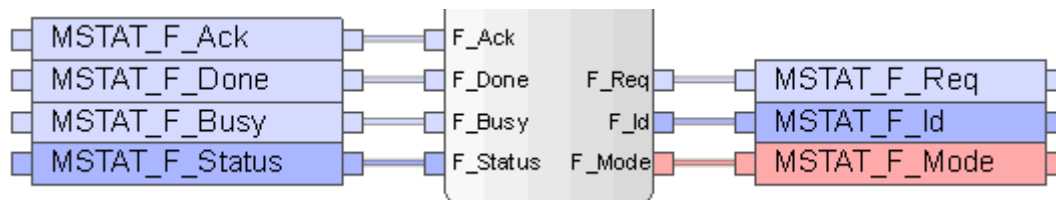


Figura 110: Bloco funcional PNM\_MSTST (parte inferior)

### 13.1.3 Configuração dos blocos funcionais na árvore de estrutura do SILworX

**Assim configura-se o bloco funcional na árvore de estrutura do SILworX:**

1. Selecionar na árvore de estrutura **Configuration, Resource, Protocols, PROFIBUS DP Master**.
2. Clicar com o botão direito do mouse em **Function Blocks** e selecionar **New**.
3. Selecionar o bloco funcional adequado (na árvore de estrutura do SILworX).

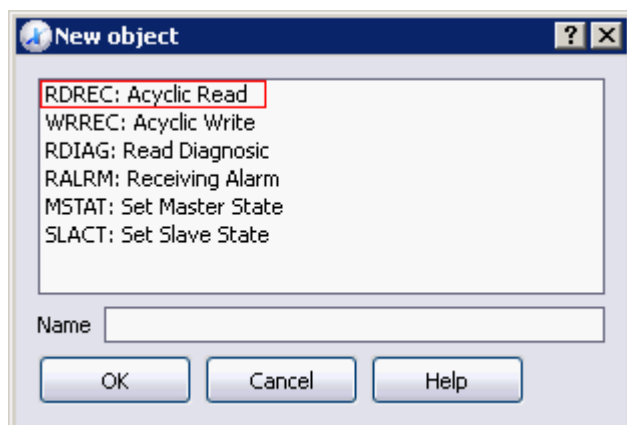


Figura 111: Seleção de blocos funcionais

As entradas do bloco funcional do bloco funcional (ganchinho na coluna variável de entrada) devem ser conectadas com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Outputs* do bloco funcional no programa de aplicação.

As saídas do bloco funcional do bloco funcional (ganchinho na coluna variável de saídas) devem ser conectadas com as mesmas variáveis com as quais anteriormente foram conectadas também as *F-Inputs* do bloco funcional no programa de aplicação.

System Variables				
F	Name	Data type	Input variable	Global Variable
1	ACK	BOOL	<input checked="" type="checkbox"/>	MSTAT_F_Ack
2	BUSY	BOOL	<input checked="" type="checkbox"/>	MSTAT_F_Busy
3	DONE	BOOL	<input checked="" type="checkbox"/>	MSTAT_F_Done
4	M_ID	DWORD	<input type="checkbox"/>	MSTAT_F_Id
5	MODE	INT	<input type="checkbox"/>	MSTAT_F_Mode
6	REQ	BOOL	<input type="checkbox"/>	MSTAT_F_Req
7	STATUS	DWORD	<input checked="" type="checkbox"/>	MSTAT_F_Status

Figura 112: Variáveis de sistema do bloco funcional MSTAT

## 13.2 Fatia de tempo de comunicação máxima

A fatia de tempo de comunicação máxima é o tempo atribuído em milissegundos (ms) por ciclo de CPU dentro do qual o módulo processador está processando as tarefas de comunicação.

Se não for possível executar todas as tarefas de comunicação em um ciclo de CPU, ocorre a transferência completa dos dados de comunicação ao longo de vários ciclos de CPU (quantidade de fatias de tempo de comunicação > 1).

### 13.2.1 Para sistemas de comando HIMax

#### Determinar a fatia de tempo de comunicação máxima

1. Operar o sistema HIMax sob carga total:  
Todos os protocolos de comunicação estão operando (safeethernet e protocolos padrão).
2. Abrir o **Control Panel** e selecionar na árvore de estrutura o diretório **Statistics**.
3. Abrir **Cyc.Async** para ler a quantidade de fatias de tempo de comunicação.
4. Abrir **Time.Async** para ler a duração de uma fatia de tempo de comunicação.

---

i

Para os cálculos dos tempos máximos de reação admissíveis, veja Capítulo 4.7 vale a condição que a quantidade das fatias de tempo de comunicação = 1.

A duração da fatia de tempo de comunicação deve ser ajustada tão alta que o ciclo de CPU não pode ultrapassar o tempo de Watchdog determinado pelo processo, quando aproveita a fatia de tempo de comunicação.

---

## 13.3 Limite de carga

Para cada protocolo de comunicação pode ser definido um limite de tempo de processador em % ( $\mu P$ -Budget). Assim é possível distribuir o tempo de processador disponível entre os protocolos configurados. A soma dos limites de tempos de processador de todos os protocolos de comunicação parametrizados de um módulo CPU ou COM não pode ser maior do que 100%.

Os limites de tempo de processador definidos dos protocolos de comunicação individuais são monitorados. Se um protocolo de comunicação alcançou ou ultrapassou o seu limite de tempo de processador e não há tempo de processador adicional à disposição, o protocolo de comunicação não é completamente processado.

Se ainda houver o suficiente de tempo de processador sobrando, o mesmo é usado para ainda processar um protocolo de comunicação que alcançou ou ultrapassou seu limite de tempo de processador. Desta forma pode ocorrer que um protocolo de comunicação use de fato um limite de tempo de processador mais alto do que lhe foi atribuído.

Eventualmente são exibidos mais do que 100% limites de tempo de processador online. Isso não é um erro, o limite de tempo de processador acima de 100% é o tempo de processador adicional.

---

i

O limite de tempo de processador adicional não é nenhuma garantia para um determinado protocolo de comunicação e pode ser revogado pelo sistema a qualquer momento.

---

## Anexo

### Glossário

Conceito	Descrição
ARP	Address Resolution Protocol: Protocolo de rede para a atribuição de endereços de rede a endereços de hardware
AI	Analog Input, Entrada analógica
Connector Board	Placa de conexão para o módulo HIMax
COM	Módulo de comunicação
CRC	Cyclic Redundancy Check, Soma de verificação
DI	Digital Input, Entrada digital
DO	Digital Output, Saída digital
CEM	Compatibilidade eletromagnética
EN	Normas européias
ESD	ElectroStatic Discharge, descarga eletrostática
FB	Fieldbus, barramento de campo
FBS	Funktionsbausteinsprache, linguagem de bloco funcional
FTA	Field Termination Assembly
FTT	Fault tolerance time, tempo de tolerância de falhas
ICMP	Internet Control Message Protocol: Protocolo de rede para mensagens de status e de falhas
IEC	Normas internacionais para eletrotécnica
Endereço MAC	Endereço de hardware de uma conexão de rede (Media Access Control)
PADT	Programming and Debugging Tool (conforme IEC 61131-3), PC com SILworX
PE	Terra de proteção
PELV	Protective Extra Low Voltage: Extra baixa tensão funcional com separação segura
PES	Programable Electronic System, Sistema eletrônico programável
PFD	Probability of Failure on Demand: Probabilidade de uma falha ao demandar uma função de segurança
PFH	Probability of Failure per Hour: Probabilidade de uma falha perigosa por hora
R	Read, Ler
Rack ID	Identificação de um suporte básico (número)
Livre de efeitos de retro-alimentação	Dois circuitos de entrada estão ligados à mesma fonte (p. ex., transmissor). Uma ligação de entrada é chamada de “sem efeito de retroalimentação” se ela não interferir com os sinais de uma outra ligação de entrada.
R/W	Read/Write, Ler/Escriver
SB	Systembus, (módulo do) barramento de sistema
SELV	Safety Extra Low Voltage: Tensão extra baixa de proteção
SFF	Safe Failure Fraction, Fração de falhas que podem ser controladas com segurança
SIL	Safety Integrity Level (conf. IEC 61508)
SILworX	Software de programação para HIMax e HIMatrix
SNTP	Simple Network Time Protocol (RFC 1769)
SRS	System.Rack.Slot
SW	Software
TMO	Timeout
W	Write
WD	Watchdog
WDT	Watchdog Time

**Lista de figuras**

<b>Figura 1:</b>	<b>Estruturas de sistema</b>	<b>37</b>
<b>Figura 2:</b>	<b>Estrutura para a configuração de uma conexão redundante</b>	<b>39</b>
<b>Figura 3:</b>	<b>Árvore de estrutura do recurso</b>	<b>39</b>
<b>Figura 4:</b>	<b>Valores dos parâmetros de uma conexão safeethernet redundante</b>	<b>40</b>
<b>Figura 5:</b>	<b>Visualização de detalhes Editor safeethernet</b>	<b>40</b>
<b>Figura 6:</b>	<b>Conexão redundante entre dois sistemas de comando HIMax</b>	<b>48</b>
<b>Figura 7:</b>	<b>Conexão redundante entre dois sistemas de comando HIMax através de um condutor</b>	<b>49</b>
<b>Figura 8:</b>	<b>Tempo de reação ao conectar dois sistemas de comando HIMax</b>	<b>55</b>
<b>Figura 9:</b>	<b>Tempo de reação na conexão de um sistema de comando HIMax com um sistema de comando HIMatrix</b>	<b>55</b>
<b>Figura 10:</b>	<b>Tempo de reação com dois Remote I/Os e um sistema de comando HIMax</b>	<b>56</b>
<b>Figura 11:</b>	<b>Tempo de reação com dois sistemas de comando HIMax e um sistema de comando HIMatrix</b>	<b>56</b>
<b>Figura 12:</b>	<b>Tempo de reação ao conectar dois sistemas de comando HIMatrix</b>	<b>57</b>
<b>Figura 13:</b>	<b>Tempo de reação com Remote I/Os</b>	<b>58</b>
<b>Figura 14:</b>	<b>Tempo de reação com dois sistemas de comando HIMatrix e um sistema de comando HIMax</b>	<b>58</b>
<b>Figura 15:</b>	<b>Conexão safeethernet entre recurso A1 no projeto A e recurso B1 no projeto B</b>	<b>63</b>
<b>Figura 16:</b>	<b>Variante Projeto A como projeto local</b>	<b>64</b>
<b>Figura 17:</b>	<b>Variante Projeto B como projeto local</b>	<b>65</b>
<b>Figura 18:</b>	<b>Comunicação entre projetos com SILworX e ELOP II Factory</b>	<b>66</b>
<b>Figura 19:</b>	<b>Recurso Proxy HIMatrix</b>	<b>67</b>
<b>Figura 20:</b>	<b>Parâmetros de uma conexão safeethernet ao recurso Proxy</b>	<b>68</b>
<b>Figura 21:</b>	<b>Exportar uma conexão safeethernet</b>	<b>69</b>
<b>Figura 22:</b>	<b>Importar conexões no ELOP II Factory</b>	<b>70</b>
<b>Figura 23:</b>	<b>Peer-to-Peer-Editor no ELOP II Factory</b>	<b>70</b>
<b>Figura 24:</b>	<b>Atribuir sinais de envio no ELOP II Factory</b>	<b>71</b>
<b>Figura 25:</b>	<b>Atribuir sinais de recepção no ELOP II Factory</b>	<b>71</b>
<b>Figura 26:</b>	<b>Control Panel para a supervisão da conexão</b>	<b>72</b>
<b>Figura 27:</b>	<b>Controlar o Consumer/Provider Status (IOxS)</b>	<b>75</b>
<b>Figura 28:</b>	<b>Control-Byte e Status-Byte no PROFIsafe</b>	<b>77</b>
<b>Figura 29:</b>	<b>Tempo de reação entre um F-Device e um HIMA F-Host</b>	<b>79</b>
<b>Figura 30:</b>	<b>Tempo de reação com um HIMA F-Host e dois F-Devices</b>	<b>79</b>
<b>Figura 31:</b>	<b>Controlador PROFINET-IO na árvore de estrutura HIMax</b>	<b>84</b>
<b>Figura 32:</b>	<b>Device Access Point (DAP) para o dispositivo PROFINET-IO</b>	<b>85</b>
<b>Figura 33:</b>	<b>Árvore de estrutura do controlador PROFINET-IO</b>	<b>89</b>
<b>Figura 34:</b>	<b>Comunicação via PROFINET/PROFIsafe</b>	<b>103</b>
<b>Figura 35:</b>	<b>Dispositivo PROFINET-IO na árvore de estrutura HIMax</b>	<b>109</b>
<b>Figura 36:</b>	<b>Comunicação via PROFIBUS DP</b>	<b>120</b>

<b>Figura 37: Slave HIMax PROFIBUS DP com módulos</b>	<b>123</b>
<b>Figura 38: Campo dos dados de usuário</b>	<b>126</b>
<b>Figura 39: Janela de diálogo verificação</b>	<b>126</b>
<b>Figura 40: Características do master PROFIBUS DP</b>	<b>127</b>
<b>Figura 41: Características do slave PROFIBUS DP</b>	<b>128</b>
<b>Figura 42: Ciclo isocrônico PROFIBUS DP</b>	<b>137</b>
<b>Figura 43: Diálogo <i>Edit User Parameters</i></b>	<b>147</b>
<b>Figura 44: Bloco funcional MSTAT</b>	<b>149</b>
<b>Figura 45: Bloco funcional RALRM</b>	<b>152</b>
<b>Figura 46: Bloco funcional RDIAG</b>	<b>156</b>
<b>Figura 47: Bloco funcional RDREC</b>	<b>160</b>
<b>Figura 48: Bloco funcional SLACT</b>	<b>163</b>
<b>Figura 49: Bloco funcional WRREC</b>	<b>166</b>
<b>Figura 50: Bloco funcional auxiliar ACTIVE</b>	<b>169</b>
<b>Figura 51: Bloco funcional auxiliar ALARM</b>	<b>170</b>
<b>Figura 52: Bloco funcional auxiliar DEID</b>	<b>171</b>
<b>Figura 53: Bloco funcional auxiliar ID</b>	<b>172</b>
<b>Figura 54: Bloco funcional auxiliar NSLOT</b>	<b>173</b>
<b>Figura 55: Bloco funcional auxiliar SLOT</b>	<b>173</b>
<b>Figura 56: Bloco funcional auxiliar STDDIAG</b>	<b>174</b>
<b>Figura 57: Topologia de barramento RS485</b>	<b>188</b>
<b>Figura 58: Comunicação via Modbus TCP</b>	<b>192</b>
<b>Figura 59: Rede Modbus</b>	<b>207</b>
<b>Figura 60: Modbus Gateway</b>	<b>210</b>
<b>Figura 61: Modbus serial</b>	<b>213</b>
<b>Figura 62: Telegrama Modbus</b>	<b>213</b>
<b>Figura 63: Ligação entre HIMax e sistema de comando Siemens</b>	<b>239</b>
<b>Figura 64: Transmissão de dados entre HIMax e sistema de comando Siemens</b>	<b>240</b>
<b>Figura 65: Lista de variáveis no bloco UDT1 da Siemens</b>	<b>241</b>
<b>Figura 66: Lista de variáveis no bloco DB1 da Siemens</b>	<b>242</b>
<b>Figura 67: SIMATIC-Symbol Editor</b>	<b>242</b>
<b>Figura 68: Esquema de função para recepção</b>	<b>243</b>
<b>Figura 69: Esquema de função para envio</b>	<b>244</b>
<b>Figura 70: Características da conexão TCP no SILworX</b>	<b>245</b>
<b>Figura 71: Lista de variáveis Siemens</b>	<b>254</b>
<b>Figura 72: Lista de variáveis HIMax/HIMatrix</b>	<b>254</b>
<b>Figura 73: Bloco funcional TCP_Reset</b>	<b>256</b>
<b>Figura 74: Bloco funcional TCP_Send</b>	<b>259</b>
<b>Figura 75: Bloco funcional TCP_Receive</b>	<b>262</b>
<b>Figura 76: Bloco funcional TCP_ReceiveLine</b>	<b>266</b>

Figura 77: Bloco funcional TCP_ReceiveVar	270
Figura 78: Estrutura dos pacotes de dados	271
Figura 79: Operação redundante X-OPC	287
Figura 80: Rotina de instalação do servidor X-OPC	288
Figura 81: Rotina de instalação do servidor X-OPC	288
Figura 82: Ajuste manual do ClassID do segundo servidor X-OPC	289
Figura 83: Operação redundante X-OPC	291
Figura 84: Operação redundante X-OPC	292
Figura 85: Alarm & Event Editor	295
Figura 86: Operação redundante X-OPC	296
Figura 87: safeethernet Editor	299
Figura 88: Visualização de detalhes da conexão safeethernet	300
Figura 89: Cinco áreas de um evento escalar	302
Figura 90: Diagrama de blocos	309
Figura 91: Troca de dados de processo entre CPU e COM (CUT)	316
Figura 92: Diálogo de configuração do Cygwin, <i>Cygwin Setup</i>	354
Figura 93: Diálogo de configuração do Cygwin, <i>Select Packages</i>	356
Figura 94: Árvore de estrutura do Cygwin	357
Figura 96: Árvore de estrutura do Cygwin	358
Figura 97: Arquivo de código C na pasta <i>example_cut</i>	358
Figura 98: Arquivo mke na pasta <i>example_cut</i>	359
Figura 99: Arquivo mke a partir da linha 1	359
Figura 100: makefile na pasta <i>example_cut</i>	360
Figura 101: makefile a partir da linha 34	360
Figura 102: makeinc.inc.app arquivo na pasta <i>example_cut</i>	360
Figura 103: makeinc.inc.app a partir da linha 247	361
Figura 104: Legenda da figura	362
Figura 105: Exemplo de código C	363
Figura 106: Cygwin Bash Shell	364
Figura 107: SILworX Programm Editor	367
Figura 108: Teste on-line SILworX	367
Figura 109: Bloco funcional PNM_MSTST (parte superior)	369
Figura 110: Bloco funcional PNM_MSTST (parte inferior)	370
Figura 111: Seleção de blocos funcionais	370
Figura 112: Variáveis de sistema do bloco funcional MSTAT	371



**Lista de tabelas**

<b>Tabela 1:</b>	<b>Manuais adicionalmente em vigor</b>	<b>10</b>
<b>Tabela 2:</b>	<b>Normas para requisitos de CEM, climáticas e do meio-ambiente</b>	<b>13</b>
<b>Tabela 3:</b>	<b>Requisitos gerais</b>	<b>13</b>
<b>Tabela 4:</b>	<b>Requisitos climáticos</b>	<b>13</b>
<b>Tabela 5:</b>	<b>Testes mecânicos</b>	<b>14</b>
<b>Tabela 6:</b>	<b>Testes de resistência contra interferência</b>	<b>14</b>
<b>Tabela 7:</b>	<b>Testes de emissão de interferência</b>	<b>14</b>
<b>Tabela 8:</b>	<b>Verificação das características da alimentação com corrente contínua</b>	<b>15</b>
<b>Tabela 9:</b>	<b>Protocolos padrão</b>	<b>18</b>
<b>Tabela 10:</b>	<b>Protocolos padrão disponíveis</b>	<b>19</b>
<b>Tabela 11:</b>	<b>Protocolos num módulo de comunicação</b>	<b>19</b>
<b>Tabela 12:</b>	<b>Protocolos num módulo de comunicação</b>	<b>19</b>
<b>Tabela 13:</b>	<b>Protocolos disponíveis do sistema HIMax/HIMatrix</b>	<b>21</b>
<b>Tabela 14:</b>	<b>Características interfaces Ethernet</b>	<b>22</b>
<b>Tabela 15:</b>	<b>Parâmetros de configuração</b>	<b>26</b>
<b>Tabela 16:</b>	<b>Parâmetros de roteamento</b>	<b>27</b>
<b>Tabela 17:</b>	<b>Parâmetros do switch Ethernet</b>	<b>27</b>
<b>Tabela 18:</b>	<b>Registro VLAN</b>	<b>28</b>
<b>Tabela 19:</b>	<b>Valores para LLDP</b>	<b>28</b>
<b>Tabela 20:</b>	<b>Submódulos de barramento de campo disponíveis</b>	<b>30</b>
<b>Tabela 21:</b>	<b>Opções para interfaces de barramento de campo FB1 e FB2</b>	<b>31</b>
<b>Tabela 22:</b>	<b>Componentes HIMax disponíveis</b>	<b>31</b>
<b>Tabela 23:</b>	<b>Exemplos para números de peça e denominações de módulos COM</b>	<b>31</b>
<b>Tabela 24:</b>	<b>Equipamento de sistemas de comando HIMatrix com sub-módulos de barramento de campo</b>	<b>32</b>
<b>Tabela 25:</b>	<b>Exemplos para números de peça dos sistemas de comando HIMatrix</b>	<b>32</b>
<b>Tabela 26:</b>	<b>Pinagem das conexões D-Sub para RS485</b>	<b>33</b>
<b>Tabela 27:</b>	<b>Pinagem das conexões D-Sub para PROFIBUS DP</b>	<b>33</b>
<b>Tabela 28:</b>	<b>Pinagem das conexões D-Sub para INTERBUS</b>	<b>33</b>
<b>Tabela 29:</b>	<b>Pinagem das conexões D-Sub para RS232</b>	<b>34</b>
<b>Tabela 30:</b>	<b>Pinagem das conexões D-Sub para RS422</b>	<b>34</b>
<b>Tabela 31:</b>	<b>Pinagem das conexões D-Sub para SSI</b>	<b>34</b>
<b>Tabela 32:</b>	<b>Protocolo direcionado à segurança (safeethernet)</b>	<b>35</b>
<b>Tabela 33:</b>	<b>Parâmetros protocolo safeethernet</b>	<b>43</b>
<b>Tabela 34:</b>	<b>Registro variáveis de sistema no safeethernet Editor</b>	<b>46</b>
<b>Tabela 35:</b>	<b>Interfaces Ethernet disponíveis</b>	<b>47</b>
<b>Tabela 36:</b>	<b>Combinações para conexões safeethernet</b>	<b>48</b>
<b>Tabela 37:</b>	<b>Campo de exibição da conexão safeethernet</b>	<b>73</b>
<b>Tabela 38:</b>	<b>Visão geral – Blocos funcionais PROFINET-IO</b>	<b>74</b>

<b>Tabela 39: Requisitos de sistema e equipamentos necessários para PROFINET-IO Controller</b>	<b>83</b>
<b>Tabela 40: Características controlador PROFINET-IO</b>	<b>84</b>
<b>Tabela 41: Registro PROFINET-IO (características)</b>	<b>90</b>
<b>Tabela 42: Registro Parâmetros (características)</b>	<b>91</b>
<b>Tabela 43: Registro Parâmetros (características)</b>	<b>91</b>
<b>Tabela 44: Registro Parâmetros</b>	<b>92</b>
<b>Tabela 45: Registro variáveis de sistema</b>	<b>92</b>
<b>Tabela 46: Registro Parâmetros do módulo I/O PROFINET-IO</b>	<b>93</b>
<b>Tabela 47: Registro Parâmetros</b>	<b>94</b>
<b>Tabela 48: Registro variáveis de sistema</b>	<b>97</b>
<b>Tabela 49: F-Parameters do submódulo Input (características)</b>	<b>98</b>
<b>Tabela 50: Registro Parâmetros</b>	<b>99</b>
<b>Tabela 51: Registro variáveis de sistema</b>	<b>100</b>
<b>Tabela 52: Registro Parâmetros</b>	<b>101</b>
<b>Tabela 53: Registro variáveis de sistema</b>	<b>102</b>
<b>Tabela 54: Application Relation (características)</b>	<b>103</b>
<b>Tabela 55: Alarm CR (características)</b>	<b>104</b>
<b>Tabela 56: Input CR (características)</b>	<b>105</b>
<b>Tabela 57: Input CR (Edit)</b>	<b>106</b>
<b>Tabela 58: Output CR (características)</b>	<b>107</b>
<b>Tabela 59: Requisitos de sistema e equipamentos necessários para PROFINET-IO Controller</b>	<b>108</b>
<b>Tabela 60: Características controlador PROFINET-IO</b>	<b>108</b>
<b>Tabela 61: Características gerais do dispositivo PROFINET-IO</b>	<b>112</b>
<b>Tabela 62: Módulos PROFINET-IO</b>	<b>113</b>
<b>Tabela 63: Módulos PROFIsafe</b>	<b>114</b>
<b>Tabela 64: Características gerais do módulo Device</b>	<b>115</b>
<b>Tabela 65: Diálogo Edit submódulo</b>	<b>117</b>
<b>Tabela 66: Equipamentos necessários e requisitos de sistema</b>	<b>119</b>
<b>Tabela 67: Características do master PROFIBUS DP</b>	<b>119</b>
<b>Tabela 68: Saídas slave HIMA PROFIBUS DP</b>	<b>121</b>
<b>Tabela 69: Entradas slave HIMA PROFIBUS DP</b>	<b>121</b>
<b>Tabela 70: Variáveis do módulo de entrada [000] DP-Input/ELOP-Export: 2 Bytes</b>	<b>124</b>
<b>Tabela 71: Variáveis do módulo de entrada [001] DP-Input/ELOP-Export: 8 Bytes</b>	<b>124</b>
<b>Tabela 72: Variáveis do módulo de entrada [002] DP-Input/ELOP-Export: 1 Byte</b>	<b>125</b>
<b>Tabela 73: Variáveis módulo de saída [003] DP-Output/ELOP-Import: 2 Bytes</b>	<b>125</b>
<b>Tabela 74: Variáveis módulo de saída [004] DP-Output/ELOP-Import: 1 Byte</b>	<b>125</b>
<b>Tabela 75: Variáveis de sistema do master PROFIBUS DP</b>	<b>129</b>
<b>Tabela 76: Características gerais do master PROFIBUS DP</b>	<b>130</b>
<b>Tabela 77: Registro Tempos no diálogo de características do master PROFIBUS DP</b>	<b>132</b>
<b>Tabela 78: Registro CPU/COM no diálogo de características do master PROFIBUS DP</b>	<b>132</b>

<b>Tabela 79: Outras características do master PROFIBUS DP</b>	<b>133</b>
<b>Tabela 80: Valores de orientação para o tempo de rotação do token com diferentes taxas de transmissão</b>	<b>134</b>
<b>Tabela 81: Tempo de transmissão para um caracter com diferentes taxas de transmissão</b>	<b>135</b>
<b>Tabela 82: Elementos para calcular o tempo de rotação do token</b>	<b>135</b>
<b>Tabela 83: Variáveis de sistema do slave PROFIBUS DP</b>	<b>139</b>
<b>Tabela 84: Registro Parâmetros do slave PROFIBUS DP</b>	<b>140</b>
<b>Tabela 85: Registro Grupos no diálogo de características do slave PROFIBUS DP</b>	<b>141</b>
<b>Tabela 86: Registro DP-V1 no diálogo de características do slave PROFIBUS DP</b>	<b>141</b>
<b>Tabela 87: Registro Alarmes no diálogo de características do slave PROFIBUS DP</b>	<b>142</b>
<b>Tabela 88: Registro Dados no diálogo de características do slave PROFIBUS DP</b>	<b>142</b>
<b>Tabela 89: Registro Model no diálogo de características do slave PROFIBUS DP</b>	<b>143</b>
<b>Tabela 90: Registro Features no diálogo de características do slave PROFIBUS DP</b>	<b>143</b>
<b>Tabela 91: Registro Taxas de transmissão no diálogo de características do slave PROFIBUS DP</b>	<b>144</b>
<b>Tabela 92: Registro Acíclico no diálogo de características do slave PROFIBUS DP</b>	<b>144</b>
<b>Tabela 93: Arquivo GSD do Slave HIMax PROFIBUS DP</b>	<b>145</b>
<b>Tabela 94: Exemplo: Blocos 1..4 do campo dos dados de usuário</b>	<b>147</b>
<b>Tabela 95: Exemplo: Blocos 5..8 do campo dos dados de usuário</b>	<b>147</b>
<b>Tabela 96: Visão geral – Blocos funcionais PROFIBUS DP</b>	<b>148</b>
<b>Tabela 97: Entradas A bloco funcional MSTAT</b>	<b>149</b>
<b>Tabela 98: Saídas A bloco funcional MSTAT</b>	<b>149</b>
<b>Tabela 99: Entradas F bloco funcional MSTAT</b>	<b>150</b>
<b>Tabela 100: Saídas F bloco funcional MSTAT</b>	<b>150</b>
<b>Tabela 101: Variáveis de sistema de entrada</b>	<b>150</b>
<b>Tabela 102: Variáveis de sistema de saída</b>	<b>151</b>
<b>Tabela 103: Entradas A bloco funcional RALRM</b>	<b>152</b>
<b>Tabela 104: Saídas A bloco funcional RALRM</b>	<b>153</b>
<b>Tabela 105: Entradas F bloco funcional RALRM</b>	<b>153</b>
<b>Tabela 106: Saídas F bloco funcional RALRM</b>	<b>153</b>
<b>Tabela 107: Variáveis de sistema de entrada</b>	<b>154</b>
<b>Tabela 108: Variáveis de sistema de saída</b>	<b>154</b>
<b>Tabela 109: Dados de alarme</b>	<b>155</b>
<b>Tabela 110: Entradas A bloco funcional RALRM</b>	<b>156</b>
<b>Tabela 111: Saídas A bloco funcional RALRM</b>	<b>156</b>
<b>Tabela 112: Entradas F bloco funcional RDIAG</b>	<b>157</b>
<b>Tabela 113: Saídas F bloco funcional RDIAG</b>	<b>157</b>
<b>Tabela 114: Variáveis de sistema de entrada</b>	<b>157</b>
<b>Tabela 115: Variáveis de sistema de saída</b>	<b>158</b>
<b>Tabela 116: Dados de diagnóstico</b>	<b>158</b>

<b>Tabela 117: Entradas A bloco funcional RDREC</b>	<b>160</b>
<b>Tabela 118: Saídas A bloco funcional RDREC</b>	<b>160</b>
<b>Tabela 119: Entradas F bloco funcional RDREC</b>	<b>161</b>
<b>Tabela 120: Saídas F bloco funcional RDREC</b>	<b>161</b>
<b>Tabela 121: Variáveis de sistema de entrada</b>	<b>161</b>
<b>Tabela 122: Variáveis de sistema de saída</b>	<b>162</b>
<b>Tabela 123: Dados</b>	<b>162</b>
<b>Tabela 124: Entradas A bloco funcional SLACT</b>	<b>163</b>
<b>Tabela 125: Saídas A bloco funcional SLACT</b>	<b>163</b>
<b>Tabela 126: Entradas F bloco funcional SLACT</b>	<b>164</b>
<b>Tabela 127: Saídas F bloco funcional SLACT</b>	<b>164</b>
<b>Tabela 128: Variáveis de sistema de entrada</b>	<b>164</b>
<b>Tabela 129: Variáveis de sistema de saída</b>	<b>165</b>
<b>Tabela 130: Entradas A bloco funcional WRREC</b>	<b>166</b>
<b>Tabela 131: Saídas A bloco funcional WRREC</b>	<b>166</b>
<b>Tabela 132: Entradas F bloco funcional WRREC</b>	<b>167</b>
<b>Tabela 133: Saídas F bloco funcional WRREC</b>	<b>167</b>
<b>Tabela 134: Variáveis de sistema de entrada</b>	<b>167</b>
<b>Tabela 135: Variáveis de sistema de saída</b>	<b>168</b>
<b>Tabela 136: Dados</b>	<b>168</b>
<b>Tabela 137: Visão geral – Blocos funcionais auxiliares</b>	<b>169</b>
<b>Tabela 138: Entradas bloco funcional auxiliar ACTIVE</b>	<b>169</b>
<b>Tabela 139: Saídas bloco funcional auxiliar ACTIVE</b>	<b>169</b>
<b>Tabela 140: Entradas bloco funcional auxiliar ALARM</b>	<b>170</b>
<b>Tabela 141: Saídas bloco funcional auxiliar ALARM</b>	<b>171</b>
<b>Tabela 142: Entradas bloco funcional auxiliar DEID</b>	<b>171</b>
<b>Tabela 143: Saídas bloco funcional auxiliar DEID</b>	<b>171</b>
<b>Tabela 144: Entradas bloco funcional auxiliar ID</b>	<b>172</b>
<b>Tabela 145: Saídas bloco funcional auxiliar ID</b>	<b>172</b>
<b>Tabela 146: Entradas bloco auxiliar NSLOT</b>	<b>173</b>
<b>Tabela 147: Saídas bloco auxiliar NSLOT</b>	<b>173</b>
<b>Tabela 148: Entradas bloco auxiliar SLOT</b>	<b>174</b>
<b>Tabela 149: Saídas bloco auxiliar SLOT</b>	<b>174</b>
<b>Tabela 150: Entradas bloco auxiliar STDDIAG</b>	<b>175</b>
<b>Tabela 151: Saídas bloco auxiliar STDDIAG</b>	<b>175</b>
<b>Tabela 152: Códigos de erro blocos funcionais</b>	<b>176</b>
<b>Tabela 153: Campo de exibição Master PROFIBUS</b>	<b>179</b>
<b>Tabela 154: Estado do master PROFIBUS DP</b>	<b>179</b>
<b>Tabela 155: Comportamento master PROFIBUS DP</b>	<b>179</b>
<b>Tabela 156: LED FBx</b>	<b>180</b>

<b>Anexo</b>	<b>Comunicação</b>
<b>Tabela 157: FAULT FBx</b>	<b>180</b>
<b>Tabela 158: Equipamentos necessários e requisitos de sistema do slave HIMA PROFIBUS DP</b>	<b>181</b>
<b>Tabela 159: Características do slave HIMA PROFIBUS DP</b>	<b>181</b>
<b>Tabela 160: Variáveis de sistema slave PROFIBUS DP</b>	<b>183</b>
<b>Tabela 161: Características do slave: Registro geral</b>	<b>185</b>
<b>Tabela 162: Campo de exibição (Slave PROFIBUS DP)</b>	<b>186</b>
<b>Tabela 163: LED FBx</b>	<b>187</b>
<b>Tabela 164: FAULT FBx</b>	<b>187</b>
<b>Tabela 165: Pinagem H 7506</b>	<b>189</b>
<b>Tabela 166: Cabo de barramento</b>	<b>189</b>
<b>Tabela 167: Características da transmissão RS485</b>	<b>190</b>
<b>Tabela 168: Equipamentos necessários e requisitos de sistema para Modbus Master</b>	<b>191</b>
<b>Tabela 169: Características Modbus Master</b>	<b>192</b>
<b>Tabela 170: Variáveis de sistema master Modbus</b>	<b>197</b>
<b>Tabela 171: Características gerais do Modbus Master</b>	<b>199</b>
<b>Tabela 172: Parâmetros COM/CPU</b>	<b>199</b>
<b>Tabela 173: Códigos de função Modbus</b>	<b>200</b>
<b>Tabela 174: Telegrama de requisição Read Coils</b>	<b>203</b>
<b>Tabela 175: Telegrama de requisição Read Discrete Inputs</b>	<b>203</b>
<b>Tabela 176: Telegrama de requisição Read Holding Registers</b>	<b>203</b>
<b>Tabela 177: Telegrama de requisição Read Input Registers</b>	<b>204</b>
<b>Tabela 178: Registro Read Write Holding</b>	<b>205</b>
<b>Tabela 179: Telegrama de requisição Write Multiple Coils</b>	<b>205</b>
<b>Tabela 180: Telegrama de requisição Write Multiple Registers</b>	<b>206</b>
<b>Tabela 181: Telegrama de requisição Write Single Coil (05)</b>	<b>206</b>
<b>Tabela 182: Telegrama de requisição Write Single Register</b>	<b>206</b>
<b>Tabela 183: Variáveis de sistema slaves TCP/UDP</b>	<b>208</b>
<b>Tabela 184: Parâmetros de configuração</b>	<b>209</b>
<b>Tabela 185: Parâmetros de conexão gateway Modbus</b>	<b>212</b>
<b>Tabela 186: Variáveis de status slave de gateway</b>	<b>212</b>
<b>Tabela 187: Parâmetros de conexão slave de gateway</b>	<b>212</b>
<b>Tabela 188: Parâmetros master Modbus serial</b>	<b>214</b>
<b>Tabela 189: Variáveis de sistema slave Modbus</b>	<b>215</b>
<b>Tabela 190: Parâmetros de conexão master Modbus</b>	<b>215</b>
<b>Tabela 191: Campo de exibição Modbus Master</b>	<b>216</b>
<b>Tabela 192: LED FBx</b>	<b>217</b>
<b>Tabela 193: FAULT FBx</b>	<b>217</b>
<b>Tabela 194: Equipamentos necessários e requisitos de sistema para slave Modbus HIMA</b>	<b>218</b>
<b>Tabela 195: Características slave Modbus</b>	<b>218</b>

<b>Tabela 196: Slots admissíveis dos módulos COM slave Modbus redundantes</b>	<b>220</b>
<b>Tabela 197: Registro Características Slave Set Modbus</b>	<b>222</b>
<b>Tabela 198: Registro Slave Set Modbus</b>	<b>223</b>
<b>Tabela 199: Registro Portas TCP e UDP para slave Modbus HIMA</b>	<b>225</b>
<b>Tabela 200: Registro Variáveis de sistema slave Modbus HIMA</b>	<b>225</b>
<b>Tabela 201: Códigos de função Modbus do slave Modbus HIMA</b>	<b>226</b>
<b>Tabela 202: Variáveis do tipo Register na área Register do slave Modbus</b>	<b>230</b>
<b>Tabela 203: Variáveis Bit na área Bit do slave Modbus</b>	<b>231</b>
<b>Tabela 204: Aba Offsets para slave Modbus HIMA</b>	<b>232</b>
<b>Tabela 205: Variáveis espelhadas da área Register para a área Bit</b>	<b>233</b>
<b>Tabela 206: Variáveis espelhadas da área Bit para a área Register</b>	<b>234</b>
<b>Tabela 207: Campo de exibição Slave Modbus</b>	<b>236</b>
<b>Tabela 208: Campo de exibição dados master</b>	<b>236</b>
<b>Tabela 209: LED FBx</b>	<b>237</b>
<b>Tabela 210: FAULT FBx</b>	<b>237</b>
<b>Tabela 211: Códigos de erro Modbus TCP/IP</b>	<b>237</b>
<b>Tabela 212: Equipamentos necessários e requisitos de sistema S&amp;R TCP</b>	<b>238</b>
<b>Tabela 213: Características S&amp;R TCP</b>	<b>238</b>
<b>Tabela 214: Configuração sistema de comando HIMax</b>	<b>240</b>
<b>Tabela 215: Configuração Siemens SIMATIC 300</b>	<b>240</b>
<b>Tabela 216: Variáveis globais</b>	<b>245</b>
<b>Tabela 217: Variáveis para dados de recepção</b>	<b>246</b>
<b>Tabela 218: Variáveis para dados de envio</b>	<b>246</b>
<b>Tabela 219: Variáveis de sistema S&amp;R TCP</b>	<b>247</b>
<b>Tabela 220: Características gerais S&amp;R TCP</b>	<b>247</b>
<b>Tabela 221: Parametros COM/CPU</b>	<b>248</b>
<b>Tabela 222: Variáveis de sistema</b>	<b>249</b>
<b>Tabela 223: Características conexão S&amp;R TCP</b>	<b>251</b>
<b>Tabela 224: Blocos funcionais para conexões S&amp;R TCP</b>	<b>255</b>
<b>Tabela 225: Entradas A bloco funcional TCP_Reset</b>	<b>256</b>
<b>Tabela 226: Saídas A do bloco funcional TCP_Reset</b>	<b>256</b>
<b>Tabela 227: Entradas F do bloco funcional TCP_Reset</b>	<b>257</b>
<b>Tabela 228: Saídas F do bloco funcional TCP_Reset</b>	<b>257</b>
<b>Tabela 229: Variáveis de sistema de entrada</b>	<b>257</b>
<b>Tabela 230: Variáveis de sistema de saída</b>	<b>258</b>
<b>Tabela 231: Entradas A do bloco funcional TCP_Send</b>	<b>259</b>
<b>Tabela 232: Saídas A do bloco funcional TCP_Send</b>	<b>259</b>
<b>Tabela 233: Entradas F do bloco funcional TCP_Send</b>	<b>260</b>
<b>Tabela 234: Saídas F do bloco funcional TCP_Send</b>	<b>260</b>
<b>Tabela 235: Variáveis de sistema de entrada</b>	<b>260</b>

Tabela 236: Variáveis de sistema de saída	261
Tabela 237: Dados de envio	261
Tabela 238: Entradas A do bloco funcional TCP_Receive	262
Tabela 239: Saídas A do bloco funcional TCP_Receive	263
Tabela 240: Entradas A do bloco funcional TCP_Receive	263
Tabela 241: Saídas F do bloco funcional TCP_Receive	263
Tabela 242: Variáveis de sistema de entrada	264
Tabela 243: Variáveis de sistema de saída	264
Tabela 244: Variáveis de recepção	264
Tabela 245: Entradas A do bloco funcional TCP_ReceiveLine	266
Tabela 246: Saídas A do bloco funcional TCP_ReceiveLine	267
Tabela 247: Entradas F do bloco funcional TCP_ReceiveLine	267
Tabela 248: Saídas F do bloco funcional TCP_ReceiveLine	267
Tabela 249: Variáveis de sistema de entrada	268
Tabela 250: Variáveis de sistema de saída	268
Tabela 251: Variáveis de recepção	268
Tabela 252: Entradas A do bloco funcional TCP_ReceiveVar	271
Tabela 253: Saídas A do bloco funcional TCP_ReceiveVar	272
Tabela 254: Entradas F do bloco funcional TCP_ReceiveVar	272
Tabela 255: Saídas F do bloco funcional TCP_ReceiveVar	272
Tabela 256: Variáveis de sistema de entrada	273
Tabela 257: Variáveis de sistema de saída	273
Tabela 258: Variáveis de recepção	273
Tabela 259: Campo de exibição protocolo S&R	275
Tabela 260: Campo de exibição Modbus Slaves	275
Tabela 261: Códigos de erro da conexão TCP	276
Tabela 262: Códigos de erro adicionais	277
Tabela 263: Estado da conexão	277
Tabela 264: Estado da conexão parceiro	277
Tabela 265: Equipamentos necessários e requisitos de sistema S&R TCP	278
Tabela 266: Características do cliente SNTP	279
Tabela 267: Características SNTP Server Info	280
Tabela 268: Características servidor SNTP	281
Tabela 269: Equipamentos necessários e requisitos de sistema do servidor X-OPC	283
Tabela 270: Características do servidor X-OPC	284
Tabela 271: Características do sistema de comando HIMA para a conexão X-OPC	285
Tabela 272: Ações necessárias no caso de alterações	286
Tabela 273: Valores padrão das prioridades	298
Tabela 274: Estado e carimbo de hora dos fragmentos Data Access	300
Tabela 275: Parâmetros para eventos booleanos	302

<b>Tabela 276: Parâmetros para eventos escalares</b>	<b>304</b>
<b>Tabela 277: Características</b>	<b>307</b>
<b>Tabela 278: Características</b>	<b>308</b>
<b>Tabela 279: Edit</b>	<b>308</b>
<b>Tabela 280: Equipamentos necessários e requisitos de sistema ComUserTask</b>	<b>309</b>
<b>Tabela 281: Formato de dados do byte de comando inicial SSISTART</b>	<b>310</b>
<b>Tabela 282: Formato e sequência dos dados de sensores</b>	<b>311</b>
<b>Tabela 283: Frequências de relógio recomendadas dependendo do comprimento de condutores de campo</b>	<b>311</b>
<b>Tabela 284: Equipamentos necessários e requisitos de sistema ComUserTask</b>	<b>313</b>
<b>Tabela 285: Características ComUserTask</b>	<b>313</b>
<b>Tabela 286: Abreviações</b>	<b>314</b>
<b>Tabela 287: Schedule Interval [ms]</b>	<b>315</b>
<b>Tabela 288: Características gerais do dispositivo PROFINET-IO</b>	<b>317</b>
<b>Tabela 289: Variáveis de sistema da ComUserTask</b>	<b>318</b>
<b>Tabela 290: Sinais de entrada da ComUserTask</b>	<b>319</b>
<b>Tabela 291: Sinais de saída da ComUserTask</b>	<b>319</b>
<b>Tabela 292: Parâmetro</b>	<b>321</b>
<b>Tabela 293: Parâmetros</b>	<b>323</b>
<b>Tabela 294: Valor de retorno</b>	<b>323</b>
<b>Tabela 295: Parâmetro</b>	<b>324</b>
<b>Tabela 296: Valor de retorno</b>	<b>324</b>
<b>Tabela 297: Parâmetro</b>	<b>325</b>
<b>Tabela 298: Valor de retorno</b>	<b>326</b>
<b>Tabela 299: Parâmetro</b>	<b>327</b>
<b>Tabela 300: Parâmetro</b>	<b>328</b>
<b>Tabela 301: Valor de retorno</b>	<b>328</b>
<b>Tabela 302: Parâmetro</b>	<b>329</b>
<b>Tabela 303: Parâmetro</b>	<b>330</b>
<b>Tabela 304: Valor de retorno</b>	<b>330</b>
<b>Tabela 305: Valor de retorno</b>	<b>331</b>
<b>Tabela 306: Parâmetros</b>	<b>332</b>
<b>Tabela 307: Parâmetros</b>	<b>333</b>
<b>Tabela 308: Valor de retorno</b>	<b>333</b>
<b>Tabela 309: Parâmetros</b>	<b>334</b>
<b>Tabela 310: Parâmetro</b>	<b>335</b>
<b>Tabela 311: Parâmetro</b>	<b>336</b>
<b>Tabela 312: Valor de retorno</b>	<b>336</b>
<b>Tabela 313: Parâmetro</b>	<b>337</b>
<b>Tabela 314: Parâmetros</b>	<b>338</b>



<b>Anexo</b>	<b>Comunicação</b>
<b>Tabela 315: Valor de retorno</b>	<b>338</b>
<b>Tabela 316: Parâmetro</b>	<b>339</b>
<b>Tabela 317: Valor de retorno</b>	<b>339</b>
<b>Tabela 318: Parâmetro</b>	<b>340</b>
<b>Tabela 319: Parâmetros</b>	<b>341</b>
<b>Tabela 320: Valor de retorno</b>	<b>341</b>
<b>Tabela 321: Parâmetro</b>	<b>342</b>
<b>Tabela 322: Parâmetros</b>	<b>343</b>
<b>Tabela 323: Valor de retorno</b>	<b>343</b>
<b>Tabela 324: Parâmetro</b>	<b>345</b>
<b>Tabela 325: Parâmetro</b>	<b>347</b>
<b>Tabela 326: Parâmetro</b>	<b>347</b>
<b>Tabela 327: Parâmetro</b>	<b>347</b>
<b>Tabela 328: Parâmetro</b>	<b>348</b>
<b>Tabela 329: Parâmetro</b>	<b>349</b>
<b>Tabela 330: Parâmetro</b>	<b>349</b>
<b>Tabela 331: Parâmetro</b>	<b>350</b>
<b>Tabela 332: Parâmetro</b>	<b>351</b>
<b>Tabela 333: Valor de retorno</b>	<b>351</b>
<b>Tabela 334: Comandos no Cygwin (Bash Shell)</b>	<b>356</b>
<b>Tabela 335: Variáveis de saída (CPU-&gt;COM)</b>	<b>366</b>
<b>Tabela 336: Variáveis de entrada (COM-&gt;CPU)</b>	<b>366</b>

**Índice remissivo**

Ativar .....	32	Proteção contra ESD .....	15
Condições de utilização		Número de peça	
Alimentação com tensão .....	15	HIMatrix.....	32
CEM .....	14	HIMax.....	31
Climáticas .....	13	Pinagens.....	33
Mecânicas .....	14	Registrar .....	32



HI 801 240 P

© 2011 HIMA Paul Hildebrandt GmbH

HIMax, HIMatrix e SILworX são marcas registradas da  
HIMA Paul Hildebrandt GmbH

HIMA Paul Hildebrandt GmbH

Albert-Bassermann-Str. 28 | 68782 Brühl | Alemanha

Telefone+49 6202 709-0 | Fax +49 6202 709-107

info@hima.com | www.hima.com



SAFETY  
NONSTOP



Uma lista detalhada de todas as filiais e representações  
encontra-se em: [www.hima.com/contact](http://www.hima.com/contact)

