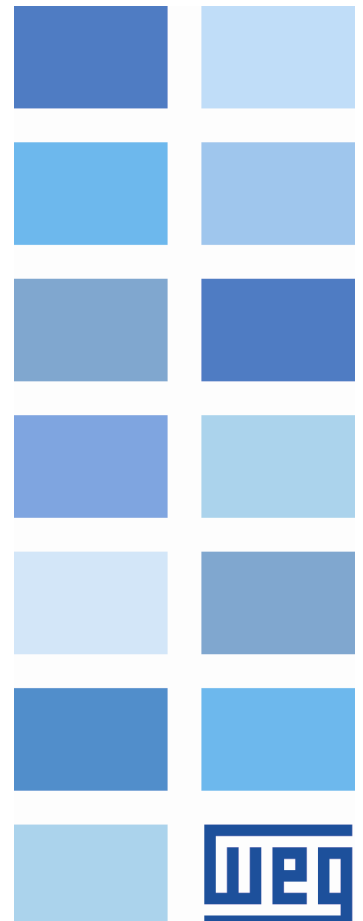


Software

WPS v2.3X

Manual do Usuário





Manual do Usuário

Série: WPS

Idioma: Português

Número do Documento: 10001027753 / R10

Data de Publicação: 05/2016

Sumário

1	WPS v2.3X.....	17
2	Introdução.....	18
2.1	Bem-vindo	18
2.2	Requisitos do Sistema	18
2.3	Equipamentos Suportados	19
2.4	Aviso de Copyright	19
2.5	Aviso de Segurança	19
3	Novidades.....	20
3.1	Novidades - Esta Versão	20
3.2	Novidades - Versões Anteriores	20
4	Instalação/Desinstalação.....	27
4.1	Antes de Instalar	27
4.2	Instalando	27
4.3	Desinstalando	27
4.4	Driver USB WEG	28
4.5	Driver USB FTDI	30
5	Conhecendo o Ambiente.....	35
5.1	Ambiente	35
5.2	Estrutura de uma Configuração	35
6	Início Rápido.....	37
6.1	Janela de Boas Vindas	37
6.2	Criando Nova Configuração	52
6.3	Nova Configuração - Equipamento Online	53
6.4	Nova Configuração - Equipamento Offline	61
6.5	Criando Novo Recurso	69
6.6	Novo Recurso - Equipo Online	69
6.7	Menu Popup	78
6.8	Menu Popup - Configuração	78
6.9	Menu Popup - Recurso	93
7	Comunicação.....	102
7.1	Parametrização do Equipamento	102
7.2	Estabelecendo Comunicação - Porta Serial USB	103
7.3	Estabelecendo Comunicação - RS232	106
7.4	Estabelecendo Comunicação - RS485	111

7.5	Cabos	116
7.6	Conversor USB/Serial	118
8	Ladder.....	120
8.1	Conceitos	120
8.1.1	Introdução	120
8.1.2	Lógica com Contatos	122
8.1.3	Terminologia	123
8.1.4	Tipo de Dados	125
8.1.5	Representação Direta	125
8.2	Editor	126
8.2.1	Área de Trabalho	126
8.2.2	Menu Ladder	127
8.2.3	Rungs	128
8.2.3.1	Visão Geral.....	129
8.2.3.2	Editando.....	129
8.2.3.3	Título e Comentário.....	130
8.2.3.4	Inserindo Elementos.....	130
8.2.3.4.1	Visão Geral.....	130
8.2.3.4.2	Em Série.....	132
8.2.3.4.3	Em Paralelo.....	132
8.2.3.5	Navegando.....	132
8.2.3.5.1	Com o Teclado.....	132
8.2.3.5.2	Com o Mouse	134
8.2.3.6	Copiar/Colar	136
8.2.4	Variáveis	139
8.2.4.1	Visão Geral.....	139
8.2.4.2	Campos.....	141
8.2.4.3	Editando na Rung.....	142
8.2.4.4	Literais na Rung.....	144
8.2.4.5	Arrays na Rung.....	145
8.2.4.6	Instâncias e Estruturas na Rung.....	146
8.2.4.7	Instâncias Voláteis e Retentivas.....	147
8.2.5	Compilação	150
8.2.6	Download	152
8.2.7	Monitoração Online	152
8.3	Trabalhando com USERFBs	156
8.3.1	Criando USERFBs	156
8.3.2	Adicionando entradas e saídas	159
8.3.3	Editando a rotina	161
8.3.4	Utilizando o USERFB	162
9	Trend.....	164
9.1	Visão Geral	164
9.2	Configuração	165
10	Equipamentos.....	167
10.1	PLC300	167
10.1.1	Descrição	167
10.1.2	Novidades e Correções do PLC300	168
10.1.3	I/O's	173
10.1.4	Marcadores de Sistema	177

10.1.5	Ladder	185
10.1.5.1	Componentes e Blocos.....	185
10.1.5.1.1	Contact.....	185
10.1.5.1.1.1	NOCONTACT.....	185
10.1.5.1.1.2	NCCONTACT.....	186
10.1.5.1.1.3	PTSCONTACT.....	187
10.1.5.1.1.4	NTSCONTACT.....	189
10.1.5.1.2	Coil	190
10.1.5.1.2.1	DIRECTCOIL.....	190
10.1.5.1.2.2	INVERTEDCOIL.....	191
10.1.5.1.2.3	SETCOIL.....	192
10.1.5.1.2.4	RESETCOIL.....	193
10.1.5.1.2.5	TOGGLECOIL.....	194
10.1.5.1.2.6	IMMEDIATECOIL.....	196
10.1.5.1.3	Logic Bit.....	197
10.1.5.1.3.1	RESETBIT.....	197
10.1.5.1.3.2	SETBIT.....	199
10.1.5.1.3.3	TESTBIT.....	201
10.1.5.1.4	Logic Boolean.....	202
10.1.5.1.4.1	NOT.....	202
10.1.5.1.4.2	AND.....	204
10.1.5.1.4.3	OR.....	205
10.1.5.1.4.4	XOR.....	207
10.1.5.1.4.5	XNOR.....	208
10.1.5.1.5	Logic Shift.....	210
10.1.5.1.5.1	ASHL.....	210
10.1.5.1.5.2	ASHR.....	212
10.1.5.1.5.3	SHL.....	214
10.1.5.1.5.4	SHR.....	215
10.1.5.1.6	Logic Rotate.....	218
10.1.5.1.6.1	ROL.....	218
10.1.5.1.6.2	ROR.....	220
10.1.5.1.7	Math Basic	222
10.1.5.1.7.1	ABS.....	222
10.1.5.1.7.2	NEG.....	223
10.1.5.1.7.3	ADD.....	225
10.1.5.1.7.4	SUB.....	227
10.1.5.1.7.5	MUL.....	229
10.1.5.1.7.6	DIV.....	231
10.1.5.1.7.7	MOD.....	233
10.1.5.1.8	Math Extended.....	235
10.1.5.1.8.1	LN.....	235
10.1.5.1.8.2	EXP.....	236
10.1.5.1.8.3	LOG10.....	239
10.1.5.1.8.4	ALOG10.....	240
10.1.5.1.8.5	POW.....	242
10.1.5.1.8.6	SQRT.....	244
10.1.5.1.8.7	ROUND.....	245
10.1.5.1.8.8	TRUNC.....	247
10.1.5.1.9	Math Trigonometry	248
10.1.5.1.9.1	SIN.....	248
10.1.5.1.9.2	COS.....	250
10.1.5.1.9.3	TAN.....	251
10.1.5.1.9.4	ASIN.....	253

10.1.5.1.9.5	ACOS.....	254
10.1.5.1.9.6	ATAN.....	256
10.1.5.1.9.7	ATAN2.....	257
10.1.5.1.10	Math Util.....	260
10.1.5.1.10.1	MAX.....	260
10.1.5.1.10.2	MIN.....	262
10.1.5.1.10.3	SAT.....	264
10.1.5.1.11	Module.....	267
10.1.5.1.11.1	USERFB.....	267
10.1.5.1.12	Compare.....	268
10.1.5.1.12.1	COMP_EQ.....	268
10.1.5.1.12.2	COMP_GE.....	270
10.1.5.1.12.3	COMP_GT.....	272
10.1.5.1.12.4	COMP_LE.....	273
10.1.5.1.12.5	COMP_LT.....	275
10.1.5.1.12.6	COMP_NE.....	277
10.1.5.1.13	Conversion.....	278
10.1.5.1.13.1	BCD.....	278
10.1.5.1.13.1	BCD_TO_WORD.....	278
10.1.5.1.13.2	WORD_TO_BCD.....	279
10.1.5.1.13.2	BOOL.....	281
10.1.5.1.13.1	BYTE_TO_BOOL.....	281
10.1.5.1.13.2	REAL_TO_BOOL.....	282
10.1.5.1.13.3	DWORD_TO_BOOL.....	284
10.1.5.1.13.4	WORD_TO_BOOL.....	285
10.1.5.1.13.3	BYTE.....	287
10.1.5.1.13.1	BOOL_TO_BYTE.....	287
10.1.5.1.13.2	REAL_TO_BYTE.....	288
10.1.5.1.13.3	DWORD_TO_BYTE.....	290
10.1.5.1.13.4	WORD_TO_BYTE.....	291
10.1.5.1.13.4	WORD.....	293
10.1.5.1.13.1	BOOL_TO_WORD.....	293
10.1.5.1.13.2	BYTE_TO_WORD.....	294
10.1.5.1.13.3	DWORD_TO_WORD.....	296
10.1.5.1.13.4	REAL_TO_WORD.....	297
10.1.5.1.13.5	REAL.....	299
10.1.5.1.13.1	BOOL_TO_REAL.....	299
10.1.5.1.13.2	DWORD_TO_REAL.....	300
10.1.5.1.13.3	WORD_TO_REAL.....	302
10.1.5.1.13.4	BYTE_TO_REAL.....	303
10.1.5.1.13.5	STRING_TO_REAL.....	305
10.1.5.1.13.6	DWORD.....	308
10.1.5.1.13.1	BOOL_TO_DWORD.....	308
10.1.5.1.13.2	REAL_TO_DWORD.....	309
10.1.5.1.13.3	WORD_TO_DWORD.....	310
10.1.5.1.13.4	BYTE_TO_DWORD.....	312
10.1.5.1.13.5	STRING_TO_DWORD.....	313
10.1.5.1.13.7	Rad-Deg.....	316
10.1.5.1.13.1	RAD_TO_DEG.....	316
10.1.5.1.13.2	DEG_TO_RAD.....	318
10.1.5.1.13.8	STRING.....	319
10.1.5.1.13.1	DWORD_TO_STRING.....	319
10.1.5.1.13.2	REAL_TO_STRING.....	321
10.1.5.1.14	Data Transfer.....	323

10.1.5.1.14.1	STORE.....	323
10.1.5.1.14.2	ILOADBOOL.....	325
10.1.5.1.14.3	ILOAD.....	326
10.1.5.1.14.4	ISTOREBOOL.....	328
10.1.5.1.14.5	ISTORE.....	330
10.1.5.1.14.6	MUX.....	331
10.1.5.1.14.7	DEMUX.....	334
10.1.5.1.14.8	SEL.....	336
10.1.5.1.14.9	SWAP.....	338
10.1.5.1.14.10	ReadRecipe.....	340
10.1.5.1.14.11	WriteRecipe.....	342
10.1.5.1.15	Counter.....	346
10.1.5.1.15.1	CTD.....	346
10.1.5.1.15.2	CTU.....	348
10.1.5.1.15.3	CTUD.....	350
10.1.5.1.16	Timer.....	354
10.1.5.1.16.1	TON.....	354
10.1.5.1.16.2	TOF.....	357
10.1.5.1.16.3	TP.....	359
10.1.5.1.17	Control.....	360
10.1.5.1.17.1	PID.....	360
10.1.5.1.18	Filter.....	365
10.1.5.1.18.1	LOWPASS.....	365
10.1.5.1.19	Hardware.....	367
10.1.5.1.19.1	READENC.....	367
10.1.5.1.19.2	READENC2.....	368
10.1.5.1.19.3	READENC3.....	371
10.1.5.1.19.4	READENC4.....	372
10.1.5.1.19.5	PWM.....	374
10.1.5.1.19.6	P_RAMP.....	376
10.1.5.1.19.7	IMMEDIATE_INPUT.....	380
10.1.5.1.19.8	IMMEDIATE_OUTPUT.....	382
10.1.5.1.20	RTC.....	385
10.1.5.1.20.1	INTIME.....	385
10.1.5.1.20.2	INWEEKDAY.....	388
10.1.5.1.21	Screen.....	392
10.1.5.1.21.1	SETSCREEN.....	392
10.1.5.1.22	Communication Network.....	394
10.1.5.1.22.1	CANopen.....	394
10.1.5.1.22.1	Visão Geral do CANopen.....	394
10.1.5.1.22.2	CO_SDORead.....	396
10.1.5.1.22.3	CO_SDOWrite.....	400
10.1.5.1.22.4	CO_MasterControlStatus.....	403
10.1.5.1.22.5	CO_SlaveStatus.....	406
10.1.5.1.22.2	Modbus RTU.....	407
10.1.5.1.22.1	Visão Geral do Modbus RTU.....	407
10.1.5.1.22.2	MB_ReadBinary.....	408
10.1.5.1.22.3	MB_ReadRegister.....	412
10.1.5.1.22.4	MB_WriteBinary.....	416
10.1.5.1.22.5	MB_WriteRegister.....	420
10.1.5.1.22.6	MB_MasterControlStatus.....	424
10.1.5.1.22.7	MB_SlaveStatus.....	426
10.1.5.1.22.3	Modbus TCP.....	428
10.1.5.1.22.1	Visão Geral do Modbus TCP.....	428

10.1.5.1.22.2	MBTCP_ReadBinary.....	429
10.1.5.1.22.3	MBTCP_ReadRegister.....	433
10.1.5.1.22.4	MBTCP_WriteBinary.....	437
10.1.5.1.22.5	MBTCP_WriteRegister.....	441
10.1.5.1.22.6	MBTCP_ServerStatus.....	445
10.1.5.1.22.7	MBTCP_ClientControlStatus.....	448
10.1.5.1.23	String.....	451
10.1.5.1.23.1	STR_COMPARE.....	451
10.1.5.1.23.2	STR_CONCAT.....	454
10.1.5.1.23.3	STR_COPY.....	455
10.1.5.1.23.4	STR_COPY_LAST.....	458
10.1.5.1.23.5	STR_DELETE.....	461
10.1.5.1.23.6	STR_FIND.....	464
10.1.5.1.23.7	STR_FIND_LAST.....	467
10.1.5.1.23.8	STR_INSERT.....	470
10.1.5.1.23.9	STR_LENGTH.....	473
10.1.5.1.23.10	STR_REPLACE.....	475
10.1.5.2	Tarefas.....	478
10.1.5.3	Estruturas.....	485
10.1.5.4	Receitas.....	488
10.1.6	Tela.....	491
10.1.6.1	Alarmes.....	492
10.1.6.2	Editor de Telas.....	497
10.1.7	Log de Eventos.....	509
10.1.8	Setup.....	518
10.1.8.1	Configuração.....	518
10.1.8.2	Janelas de Configuração.....	519
10.1.8.2.1	Display.....	519
10.1.8.2.2	Analógicas.....	520
10.1.8.2.3	Encoder.....	520
10.1.8.2.4	RS232.....	521
10.1.8.2.5	RS485.....	521
10.1.8.2.6	CAN.....	522
10.1.8.2.7	LAN.....	522
10.1.8.2.8	Modbus TCP.....	523
10.1.8.2.9	Ajustes do Relógio.....	523
10.1.8.2.10	Idioma.....	524
10.1.8.2.11	Watchdog.....	524
10.1.9	Comunicação.....	525
10.1.9.1	Comandos online.....	525
10.1.9.2	Forçar E/S.....	528
10.1.9.3	Dow nload.....	530
10.1.9.4	Hot Dow nload.....	533
10.1.9.4.1	Visão Geral.....	533
10.1.9.4.2	Habilitar/Desabilitar Hot Dow nload.....	533
10.1.9.4.3	Restrições.....	536
10.1.9.4.4	Funcionamento.....	538
10.1.9.5	Upload.....	540
10.1.9.6	Comparação do recurso com o dispositivo.....	541
10.1.9.7	Gerenciador de Arquivos Modbus.....	543
10.1.9.8	Comunicação RS232.....	545
10.1.9.9	Comunicação RS485.....	548
10.2	SCA06.....	549
10.2.1	Descrição.....	550

10.2.2	Marcadores de Sistema	550
10.2.3	Start-up Orientado	554
10.2.4	Auto-Tuning	558
10.2.5	Importar do WLP	560
10.2.6	Parâmetros	563
10.2.6.1	Visão Geral.....	563
10.2.6.2	Configuração.....	565
10.2.6.3	Leitura e Escrita de Parâmetros.....	568
10.2.6.4	Ocultar/Desocultar Parâmetros e Grupos de Parâmetros	571
10.2.6.5	Parâmetros do Usuário.....	581
10.2.7	Ladder	583
10.2.7.1	Componentes e Blocos.....	583
10.2.7.1.1	Contact.....	583
10.2.7.1.1.1	NOCONTACT	583
10.2.7.1.1.2	NCONTACT.....	585
10.2.7.1.1.3	PTSCONTACT.....	586
10.2.7.1.1.4	NTSCONTACT.....	587
10.2.7.1.2	Coil	589
10.2.7.1.2.1	DIRECTCOIL	589
10.2.7.1.2.2	INVERTEDCOIL.....	590
10.2.7.1.2.3	SETCOIL	591
10.2.7.1.2.4	RESETCOIL	592
10.2.7.1.2.5	TOGGLECOIL.....	593
10.2.7.1.2.6	IMMEDIATECOIL	594
10.2.7.1.3	Logic Bit.....	596
10.2.7.1.3.1	RESETBIT.....	596
10.2.7.1.3.2	SETBIT.....	598
10.2.7.1.3.3	TESTBIT	600
10.2.7.1.4	Logic Boolean.....	602
10.2.7.1.4.1	NOT.....	602
10.2.7.1.4.2	AND.....	603
10.2.7.1.4.3	OR.....	605
10.2.7.1.4.4	XOR.....	606
10.2.7.1.4.5	XNOR.....	608
10.2.7.1.5	Logic Shift.....	609
10.2.7.1.5.1	ASHL	609
10.2.7.1.5.2	ASHR.....	612
10.2.7.1.5.3	SHL.....	614
10.2.7.1.5.4	SHR.....	615
10.2.7.1.6	Logic Rotate.....	618
10.2.7.1.6.1	ROL.....	618
10.2.7.1.6.2	ROR.....	620
10.2.7.1.7	Math Basic	622
10.2.7.1.7.1	ABS.....	622
10.2.7.1.7.2	NEG.....	623
10.2.7.1.7.3	ADD.....	625
10.2.7.1.7.4	SUB.....	627
10.2.7.1.7.5	MUL.....	629
10.2.7.1.7.6	DIV.....	631
10.2.7.1.7.7	MOD.....	633
10.2.7.1.8	Math Extended.....	635
10.2.7.1.8.1	LN.....	635
10.2.7.1.8.2	EXP.....	636
10.2.7.1.8.3	LOG10.....	639

10.2.7.1.8.4	ALOG10.....	640
10.2.7.1.8.5	POW.....	642
10.2.7.1.8.6	SQRT.....	644
10.2.7.1.8.7	ROUND.....	645
10.2.7.1.8.8	TRUNC.....	647
10.2.7.1.9	Math Trigonometry	648
10.2.7.1.9.1	SIN.....	648
10.2.7.1.9.2	COS.....	650
10.2.7.1.9.3	TAN.....	651
10.2.7.1.9.4	ASIN.....	653
10.2.7.1.9.5	ACOS.....	654
10.2.7.1.9.6	ATAN.....	656
10.2.7.1.9.7	ATAN2.....	657
10.2.7.1.10	Math Util.....	660
10.2.7.1.10.1	MAX.....	660
10.2.7.1.10.2	MIN.....	662
10.2.7.1.10.3	SAT.....	664
10.2.7.1.11	Module.....	667
10.2.7.1.11.1	USERFB.....	667
10.2.7.1.12	Comparação.....	668
10.2.7.1.12.1	COMP_EQ.....	668
10.2.7.1.12.2	COMP_GE.....	670
10.2.7.1.12.3	COMP_GT.....	672
10.2.7.1.12.4	COMP_LE.....	673
10.2.7.1.12.5	COMP_LT.....	675
10.2.7.1.12.6	COMP_NE.....	677
10.2.7.1.13	Conversão.....	678
10.2.7.1.13.1	BCD.....	678
10.2.7.1.13.1	BCD_TO_WORD.....	678
10.2.7.1.13.2	WORD_TO_BCD.....	679
10.2.7.1.13.2	BOOL.....	681
10.2.7.1.13.1	BYTE_TO_BOOL.....	681
10.2.7.1.13.2	REAL_TO_BOOL.....	682
10.2.7.1.13.3	LREAL_TO_BOOL.....	684
10.2.7.1.13.4	DWORD_TO_BOOL.....	685
10.2.7.1.13.5	WORD_TO_BOOL.....	686
10.2.7.1.13.3	BYTE.....	688
10.2.7.1.13.1	BOOL_TO_BYTE.....	688
10.2.7.1.13.2	REAL_TO_BYTE.....	689
10.2.7.1.13.3	LREAL_TO_BYTE.....	691
10.2.7.1.13.4	DWORD_TO_BYTE.....	692
10.2.7.1.13.5	WORD_TO_BYTE.....	693
10.2.7.1.13.4	WORD.....	695
10.2.7.1.13.1	BOOL_TO_WORD.....	695
10.2.7.1.13.2	BYTE_TO_WORD.....	696
10.2.7.1.13.3	DWORD_TO_WORD.....	698
10.2.7.1.13.4	REAL_TO_WORD.....	699
10.2.7.1.13.5	LREAL_TO_WORD.....	701
10.2.7.1.13.5	REAL.....	702
10.2.7.1.13.1	BOOL_TO_REAL.....	702
10.2.7.1.13.2	DWORD_TO_REAL.....	703
10.2.7.1.13.3	WORD_TO_REAL.....	705
10.2.7.1.13.4	BYTE_TO_REAL.....	706
10.2.7.1.13.5	LREAL_TO_REAL.....	708

10.2.7.1.13.6	DWORD.....	709
10.2.7.1.13.1	BOOL_TO_DWORD.....	709
10.2.7.1.13.2	REAL_TO_DWORD.....	710
10.2.7.1.13.3	LREAL_TO_DWORD.....	712
10.2.7.1.13.4	WORD_TO_DWORD.....	713
10.2.7.1.13.5	BYTE_TO_DWORD.....	714
10.2.7.1.13.7	LREAL.....	716
10.2.7.1.13.1	BOOL_TO_LREAL.....	716
10.2.7.1.13.2	DWORD_TO_LREAL.....	717
10.2.7.1.13.3	WORD_TO_LREAL.....	718
10.2.7.1.13.4	BYTE_TO_LREAL.....	719
10.2.7.1.13.5	REAL_TO_LREAL.....	720
10.2.7.1.13.8	Rad-Deg.....	721
10.2.7.1.13.1	RAD_TO_DEG.....	721
10.2.7.1.13.2	DEG_TO_RAD.....	722
10.2.7.1.14	Data Transfer.....	724
10.2.7.1.14.1	STORE.....	724
10.2.7.1.14.2	ILOADBOOL.....	725
10.2.7.1.14.3	ILOAD.....	727
10.2.7.1.14.4	ISTOREBOOL.....	729
10.2.7.1.14.5	ISTORE.....	731
10.2.7.1.14.6	MUX.....	732
10.2.7.1.14.7	DEMUX.....	735
10.2.7.1.14.8	SEL.....	737
10.2.7.1.14.9	SWAP.....	739
10.2.7.1.14.10	USERERR.....	741
10.2.7.1.15	Counter.....	742
10.2.7.1.15.1	CTD.....	742
10.2.7.1.15.2	CTU.....	744
10.2.7.1.15.3	CTUD.....	747
10.2.7.1.16	Timer.....	751
10.2.7.1.16.1	TON.....	751
10.2.7.1.16.2	TOF.....	754
10.2.7.1.16.3	TP.....	756
10.2.7.1.17	Control.....	757
10.2.7.1.17.1	PID.....	757
10.2.7.1.18	Filter.....	762
10.2.7.1.18.1	LOWPASS.....	762
10.2.7.1.19	Hardware.....	764
10.2.7.1.19.1	READENC5.....	764
10.2.7.1.19.2	IMMEDIATE_INPUT.....	767
10.2.7.1.19.3	IMMEDIATE_OUTPUT.....	769
10.2.7.1.20	RTC.....	772
10.2.7.1.20.1	INTIME.....	772
10.2.7.1.20.2	INWEEKDAY.....	775
10.2.7.1.21	Communication Network.....	779
10.2.7.1.21.1	CANopen.....	779
10.2.7.1.21.1	Visão Geral do CANopen.....	779
10.2.7.1.21.2	CO_SDORead.....	781
10.2.7.1.21.3	CO_SDOWrite.....	784
10.2.7.1.22	Motion Control Cam.....	787
10.2.7.1.22.1	MC_CamTableSelect.....	787
10.2.7.1.22.2	MW_CamCalc.....	789
10.2.7.1.22.3	MC_CamIn.....	794

10.2.7.1.22.4	MC_CamOut.....	799
10.2.7.1.23	Motion Control Command.....	801
10.2.7.1.23.1	MC_Pow er.....	801
10.2.7.1.23.2	MC_Reset.....	804
10.2.7.1.23.3	MC_Stop.....	806
10.2.7.1.23.4	MW_IqControl.....	810
10.2.7.1.24	Motion Control Gear.....	812
10.2.7.1.24.1	MC_GearIn.....	812
10.2.7.1.24.2	MC_GearInPos.....	817
10.2.7.1.24.3	MC_PhasingRelative.....	820
10.2.7.1.24.4	MC_GearOut.....	823
10.2.7.1.25	Motion Control Homing.....	825
10.2.7.1.25.1	MC_StepAbsoluteSw itch.....	825
10.2.7.1.25.2	MC_StepLimitSw itch.....	829
10.2.7.1.25.3	MC_StepReferencePulse.....	832
10.2.7.1.25.4	MC_HomeDirect.....	836
10.2.7.1.25.5	MC_FinishHoming.....	838
10.2.7.1.26	Motion Control Move.....	840
10.2.7.1.26.1	MC_MoveAbsolute.....	840
10.2.7.1.26.2	MC_MoveRelative.....	845
10.2.7.1.26.3	MC_MoveVelocity.....	851
10.2.7.2	Cam Profiles.....	856
10.2.7.3	Estruturas.....	866
10.2.8	Diagnóstico.....	869
10.2.8.1	Painel de Monitoração.....	869
10.2.8.1.1	Main Signals.....	869
10.2.8.2	Log.....	871
10.2.8.2.1	Visão Geral.....	871
10.2.8.2.2	Configuração.....	872
10.2.8.3	Trace.....	873
10.2.8.3.1	Visão Geral.....	873
10.2.8.3.2	Configuração.....	875
10.2.9	Comunicação.....	881
10.2.9.1	Forçar E/S.....	881
10.3	CFW300.....	883
10.3.1	Descrição.....	884
10.3.2	Marcadores de Sistema.....	884
10.3.3	Importar do WLP.....	886
10.3.4	Parâmetros.....	889
10.3.4.1	Visão Geral.....	889
10.3.4.2	Configuração.....	891
10.3.4.3	Leitura e Escrita de Parâmetros.....	894
10.3.4.4	Ocultar/Desocultar Parâmetros e Grupos de Parâmetros.....	897
10.3.4.5	Parâmetros do Usuário.....	907
10.3.5	Ladder.....	909
10.3.5.1	Componentes e Blocos.....	909
10.3.5.1.1	Contact.....	909
10.3.5.1.1.1	NOCONTACT.....	909
10.3.5.1.1.2	NCCONTACT.....	911
10.3.5.1.1.3	PTSCONTACT.....	912
10.3.5.1.1.4	NTSCONTACT.....	913
10.3.5.1.2	Coil.....	915
10.3.5.1.2.1	DIRECTCOIL.....	915
10.3.5.1.2.2	INVERTEDCOIL.....	916

10.3.5.1.2.3	SETCOIL.....	917
10.3.5.1.2.4	RESETCOIL.....	918
10.3.5.1.2.5	TOGGLECOIL.....	919
10.3.5.1.2.6	IMMEDIATECOIL.....	920
10.3.5.1.3	Logic Bit.....	922
10.3.5.1.3.1	RESETBIT.....	922
10.3.5.1.3.2	SETBIT.....	924
10.3.5.1.3.3	TESTBIT.....	926
10.3.5.1.4	Logic Boolean.....	928
10.3.5.1.4.1	NOT.....	928
10.3.5.1.4.2	AND.....	929
10.3.5.1.4.3	OR.....	931
10.3.5.1.4.4	XOR.....	932
10.3.5.1.4.5	XNOR.....	934
10.3.5.1.5	Logic Shift.....	935
10.3.5.1.5.1	ASHL.....	935
10.3.5.1.5.2	ASHR.....	938
10.3.5.1.5.3	SHL.....	940
10.3.5.1.5.4	SHR.....	941
10.3.5.1.6	Logic Rotate.....	944
10.3.5.1.6.1	ROL.....	944
10.3.5.1.6.2	ROR.....	946
10.3.5.1.7	Math Basic.....	948
10.3.5.1.7.1	ABS.....	948
10.3.5.1.7.2	NEG.....	949
10.3.5.1.7.3	ADD.....	951
10.3.5.1.7.4	SUB.....	953
10.3.5.1.7.5	MUL.....	955
10.3.5.1.7.6	DIV.....	957
10.3.5.1.7.7	MOD.....	959
10.3.5.1.8	Math Extended.....	961
10.3.5.1.8.1	LN.....	961
10.3.5.1.8.2	EXP.....	962
10.3.5.1.8.3	LOG10.....	965
10.3.5.1.8.4	ALOG10.....	966
10.3.5.1.8.5	POW.....	968
10.3.5.1.8.6	SQRT.....	970
10.3.5.1.8.7	ROUND.....	971
10.3.5.1.8.8	TRUNC.....	973
10.3.5.1.9	Math Trigonometry.....	974
10.3.5.1.9.1	SIN.....	974
10.3.5.1.9.2	COS.....	976
10.3.5.1.9.3	TAN.....	977
10.3.5.1.9.4	ASIN.....	979
10.3.5.1.9.5	ACOS.....	980
10.3.5.1.9.6	ATAN.....	982
10.3.5.1.9.7	ATAN2.....	983
10.3.5.1.10	Math Util.....	986
10.3.5.1.10.1	MAX.....	986
10.3.5.1.10.2	MIN.....	988
10.3.5.1.10.3	SAT.....	990
10.3.5.1.11	Module.....	993
10.3.5.1.11.1	USERFB.....	993
10.3.5.1.12	Comparação.....	994

10.3.5.1.12.1	COMP_EQ.....	994
10.3.5.1.12.2	COMP_GE.....	996
10.3.5.1.12.3	COMP_GT.....	998
10.3.5.1.12.4	COMP_LE.....	999
10.3.5.1.12.5	COMP_LT.....	1001
10.3.5.1.12.6	COMP_NE.....	1003
10.3.5.1.13	Conversão.....	1004
10.3.5.1.13.1	BCD.....	1004
10.3.5.1.13.1	BCD_TO_WORD.....	1004
10.3.5.1.13.2	WORD_TO_BCD.....	1005
10.3.5.1.13.2	BOOL.....	1007
10.3.5.1.13.1	BYTE_TO_BOOL.....	1007
10.3.5.1.13.2	REAL_TO_BOOL.....	1008
10.3.5.1.13.3	LREAL_TO_BOOL.....	1010
10.3.5.1.13.4	DWORD_TO_BOOL.....	1011
10.3.5.1.13.5	WORD_TO_BOOL.....	1012
10.3.5.1.13.3	BYTE.....	1014
10.3.5.1.13.1	BOOL_TO_BYTE.....	1014
10.3.5.1.13.2	REAL_TO_BYTE.....	1015
10.3.5.1.13.3	LREAL_TO_BYTE.....	1017
10.3.5.1.13.4	DWORD_TO_BYTE.....	1018
10.3.5.1.13.5	WORD_TO_BYTE.....	1019
10.3.5.1.13.4	WORD.....	1021
10.3.5.1.13.1	BOOL_TO_WORD.....	1021
10.3.5.1.13.2	BYTE_TO_WORD.....	1022
10.3.5.1.13.3	DWORD_TO_WORD.....	1024
10.3.5.1.13.4	REAL_TO_WORD.....	1025
10.3.5.1.13.5	LREAL_TO_WORD.....	1027
10.3.5.1.13.5	REAL.....	1028
10.3.5.1.13.1	BOOL_TO_REAL.....	1028
10.3.5.1.13.2	DWORD_TO_REAL.....	1029
10.3.5.1.13.3	WORD_TO_REAL.....	1031
10.3.5.1.13.4	BYTE_TO_REAL.....	1032
10.3.5.1.13.5	LREAL_TO_REAL.....	1034
10.3.5.1.13.6	DWORD.....	1035
10.3.5.1.13.1	BOOL_TO_DWORD.....	1035
10.3.5.1.13.2	REAL_TO_DWORD.....	1036
10.3.5.1.13.3	LREAL_TO_DWORD.....	1038
10.3.5.1.13.4	WORD_TO_DWORD.....	1039
10.3.5.1.13.5	BYTE_TO_DWORD.....	1040
10.3.5.1.13.7	LREAL.....	1042
10.3.5.1.13.1	BOOL_TO_LREAL.....	1042
10.3.5.1.13.2	DWORD_TO_LREAL.....	1043
10.3.5.1.13.3	WORD_TO_LREAL.....	1044
10.3.5.1.13.4	BYTE_TO_LREAL.....	1045
10.3.5.1.13.5	REAL_TO_LREAL.....	1046
10.3.5.1.13.8	Rad-Deg.....	1047
10.3.5.1.13.1	RAD_TO_DEG.....	1047
10.3.5.1.13.2	DEG_TO_RAD.....	1048
10.3.5.1.14	Data Transfer.....	1050
10.3.5.1.14.1	STORE.....	1050
10.3.5.1.14.2	ILOADBOOL.....	1051
10.3.5.1.14.3	ILOAD.....	1053
10.3.5.1.14.4	ISTOREBOOL.....	1055

10.3.5.1.14.5	ISTORE.....	1057
10.3.5.1.14.6	MUX.....	1058
10.3.5.1.14.7	DEMUX.....	1061
10.3.5.1.14.8	SEL.....	1063
10.3.5.1.14.9	SWAP.....	1065
10.3.5.1.14.10	USERERR.....	1067
10.3.5.1.15	Counter.....	1068
10.3.5.1.15.1	CTD.....	1068
10.3.5.1.15.2	CTU.....	1070
10.3.5.1.15.3	CTUD.....	1073
10.3.5.1.16	Timer.....	1077
10.3.5.1.16.1	TON.....	1077
10.3.5.1.16.2	TOF.....	1080
10.3.5.1.16.3	TP.....	1082
10.3.5.1.17	Control.....	1083
10.3.5.1.17.1	PID.....	1083
10.3.5.1.18	Filter.....	1088
10.3.5.1.18.1	LOWPASS.....	1088
10.3.5.1.19	Hardware.....	1090
10.3.5.1.19.1	READENC5.....	1090
10.3.5.1.19.2	IMMEDIATE_INPUT.....	1093
10.3.5.1.19.3	IMMEDIATE_OUTPUT.....	1095
10.3.5.1.20	RTC.....	1098
10.3.5.1.20.1	INTIME.....	1098
10.3.5.1.20.2	INWEEKDAY.....	1101
10.3.5.1.21	Communication Network.....	1105
10.3.5.1.21.1	CANopen.....	1105
10.3.5.1.21.1	Visão Geral do CANopen.....	1105
10.3.5.1.21.2	CO_SDORead.....	1107
10.3.5.1.21.3	CO_SDOWrite.....	1110
10.3.5.1.22	Motion Control Cam.....	1113
10.3.5.1.22.1	MC_CamTableSelect.....	1113
10.3.5.1.22.2	MW_CamCalc.....	1115
10.3.5.1.22.3	MC_CamIn.....	1120
10.3.5.1.22.4	MC_CamOut.....	1125
10.3.5.1.23	Motion Control Command.....	1127
10.3.5.1.23.1	MC_Power.....	1127
10.3.5.1.23.2	MC_Reset.....	1130
10.3.5.1.23.3	MC_Stop.....	1132
10.3.5.1.23.4	MW_IqControl.....	1136
10.3.5.1.24	Motion Control Gear.....	1138
10.3.5.1.24.1	MC_GearIn.....	1138
10.3.5.1.24.2	MC_GearInPos.....	1143
10.3.5.1.24.3	MC_PhasingRelative.....	1146
10.3.5.1.24.4	MC_GearOut.....	1149
10.3.5.1.25	Motion Control Homing.....	1151
10.3.5.1.25.1	MC_StepAbsoluteSwitch.....	1151
10.3.5.1.25.2	MC_StepLimitSwitch.....	1155
10.3.5.1.25.3	MC_StepReferencePulse.....	1158
10.3.5.1.25.4	MC_HomeDirect.....	1162
10.3.5.1.25.5	MC_FinishHoming.....	1164
10.3.5.1.26	Motion Control Move.....	1166
10.3.5.1.26.1	MC_MoveAbsolute.....	1166
10.3.5.1.26.2	MC_MoveRelative.....	1171

10.3.5.1.26.3	MC_MoveVelocity.....	1177
10.3.5.2	Estruturas	1182
10.3.6	Diagnóstico	1184
10.3.6.1	Painel de Monitoração.....	1184
10.3.6.1.1	Main Signals.....	1184
10.3.7	Comunicação	1186
10.3.7.1	Forçar E/S.....	1186

Índice	1189

1 WPS v2.3X



Suporte Técnico: Contate uma filial ou representante.

Contato: <http://www.weg.net>

Data de Publicação: 05/2016

Início

Clique no botão Bem Vindo para iniciar.

Bem Vindo

2 Introdução

[Bem Vindo](#)

[Requisitos do Sistema](#)

[Equipamentos Suportados](#)

[Aviso de Copyright](#)

[Aviso de Segurança](#)

2.1 Bem-vindo

Bem-vindo ao WPS!

O WPS v2.3X é uma ferramenta integrada inovadora para acesso a diversas famílias de produtos WEG, agregando os conceitos de:

- Multi-Produtos, visando a atender ampla gama de produtos da WEG;
- Multi-Utilização, visando a permitir:
 - o Configuração dos equipamentos,
 - o Programação dos equipamentos em linguagem Ladder,
 - o Monitoração dos equipamentos, e
 - o Assistência de criação e configuração de aplicações na área de automação.

O ambiente de programação em linguagem Ladder atende aos requisitos da norma IEC 61131-3.

Recomenda-se fortemente que o usuário esteja ambientado com as diretrizes e nomenclaturas que esta norma apresenta, de forma a utilizar o *software* da melhor forma possível.

2.2 Requisitos do Sistema

Item	Descrição
Plataforma	Windows 7 ou superior
Processador	Mínimo: Pentium IV
	Recomendado: Core i3
Memória	Mínimo: 1 GB
	Recomendado: 2 GB
Resolução de Tela	1024x768 ou superior
Espaço em Disco	500 MB
Comunicação	USB, Ethernet TCP/IP e Porta Serial
Navegador	IE7+ ou Firefox 46+

2.3 Equipamentos Suportados

A seguinte tabela apresenta os equipamentos e versões de firmware que cada base de dados fornecida com o WPS suporta.

WPS	PLC300	SCA06	CFW300	CVW500
v2.30	v1.51 - v3.08	v2.00	v1.0X - v1.1X	v1.0X
v2.20	v1.51 - v3.04	v2.00		
v2.10	v1.51 - v3.04	v2.00		
v2.00	v1.20 - v2.13	v2.00		
v1.80	v1.20 - v2.04			
v1.70	v1.20 - v1.75			
v1.60	v1.20 - v1.51			
v1.50	v1.20 - v1.43			
v1.40	v1.20 - v1.38			
v1.30	v1.20 - v1.33			
v1.20	v1.20			

**NOTA!**

Recomenda-se sempre realizar upgrade do firmware do PLC300 para a versão mais atual.

- Hardware versão 1 (H1): v1.76 ou superior
- Hardware versão 2 (H2): v2.42 ou superior
- Hardware versão 2 c/ 1MB (H2-1M): v3.08 ou superior

2.4 Aviso de Copyright

Este programa de computador está protegido pelas leis dos direitos autorais e tratados internacionais. Sua reprodução ou distribuição, parcial ou total, sem autorização prévia pode resultar em severas penalidades civis e criminais, sujeito às sanções previstas em lei.

2.5 Aviso de Segurança

Uso deste software pode alterar a operação ou desempenho do drive. O usuário é responsável pela adoção de todas as precauções necessárias para garantir a segurança do equipamento e do pessoal envolvido. Antes de aplicar este Software, leia atentamente as Instruções da Ajuda Online. A não observação destas instruções pode causar sérios danos no equipamento e resultar em graves lesões corporais.

3 Novidades

[Novidades - Esta Versão](#)

[Novidades - Versões Anteriores](#)

3.1 Novidades - Esta Versão

WPS V2.30

Novas Funções:

- Incluído CFW300 V1.0X e V1.1X;
- Incluído CVW500 V1.0X;
- Suporte ao Windows 10 (x86 e x64);
- Instalação automática do driver USB WEG;
- Instalação automática do driver FTDI;
- Janela de boas vindas com acesso rápido a algumas funções;
- Melhoria na configuração de comunicação.

Modificações:

- Melhoria no desempenho e na interface do gráfico de tendência;
- Melhoria na organização dos menus.

3.2 Novidades - Versões Anteriores

WPS V2.20

Funcionalidades:

- Função para remover variáveis não utilizadas
- Função para procurar e substituir variáveis no recurso
- Filtro de variáveis na tabela de variáveis

Melhorias:

- JRE (Java) embutido na instalação do WPS
- Melhorias na performance do trend
- Melhorias de design, ícones, janelas etc...
- Definição de recurso principal simplificada
- Reestruturação das ações dos menus e ações sensíveis ao contexto
- Importação de configuração com validação de arquivo .bcp
- Ordenação das configurações e recursos por ordem alfabética
- Suporte a bitfields na tabela de parâmetros
- Monitoração dos parâmetros no ladder

WPS V2.10

Funcionalidades:

- PLC300 v2.30 com suporte a Hot Download e Watchdog
- Importador/Conversor de projetos do WLP para o WPS para o equipamento SCA06
- Impressão completa das configurações e recursos
- Monitoração de variáveis com funcionalidade para fazer backup e restaurar o valor das variáveis
- Criada tela com informações do dispositivo conectado
- Criada tela para informações adicionais do recurso como autor, descrição e cliente

Melhorias:

- Help do WPS reescrito para todos os blocos ladder
- Janela de seleção de variáveis alterada para formato de tabela com opções de filtro
- Mover variáveis de um bloco para outro no editor ladder
- Busca de variáveis com filtros
- Na parte de estruturas foi adicionado suporte a movimentação de várias variáveis simultaneamente e lembrar último tipo de dado adicionado para futuras adições
- Driver USB com suporte ao Windows 8.1

WPS V2.00

Funcionalidades:

- Suporte ao SCA06 a partir da versão de firmware v2.00, agregando as funções dos softwares SuperDrive G2 e WLP
- PLC300 v2.10 com suporte a Strings

WPS V1.80

Funcionalidades:

- PLC300 V2.00 (Hardware versão 2)
- Bloco P_RAMP (gera trem de pulsos com rampa em frequência)

WPS V1.70

Funcionalidades:

- Upload e download dos dados das variáveis, permitindo preservar o valor das variáveis durante o download do programa. Compatível com firmware v1.70 ou superior
- Exportação e download de binários
- Criação de um componente do tipo Text Input no editor de telas

Melhorias:

- Gerenciamento de projetos:
 - o Criação de múltiplas configurações CANopen
 - o Possibilidade de associar o Ladder a uma tarefa na última etapa de sua criação no wizard

- Editor Ladder:
 - o Otimização no editor de diagramas Ladder, diminuindo a utilização de processador e memória
 - o Drag 'n drop de variável da lista de variáveis para os argumentos dos blocos
 - o Aumento do tamanho do título da rung
 - o Aumento do número de elementos por rung
 - o Quebra automática das tags
 - o Exibição do nome das telas ao configurar um botão de função para chamar uma tela
- Otimização na monitoração para diminuição do tempo de resposta
- Otimização no compilador para diminuição do tempo de compilação
- Opção de sincronização do relógio com o do computador no setup

Correções:

- Correção durante exclusão do POU da tarefa, onde as alterações não eram salvas caso outra ação não fosse feita posteriormente
- Correção no compilador quando um mesmo POU estava associado a mais de uma tarefa
- Correção durante o carregamento da tabela de variáveis, que travava quando associada a um tipo de estrutura muito grande
- Correção para aumento de performance durante busca pelos campos das estruturas
- Quando o componente Numeric Input era configurado como inteiro e continha dígitos decimais os valores dos limites não estavam sendo compilados corretamente
- Perda da seleção principal ao mudar versão do equipamento
- Teclas de função não funcionavam quando eram utilizadas com variáveis do tipo BOOL com tamanho maior que 0 (zero). O compilador não estava gerando o endereço de memória correto para acesso as variáveis do tipo BOOL com tamanho maior que 0 (zero)
- A senha do cartão SD estava sendo gravada quando a opção estava desabilitada, o compilador não estava verificando a sinalização da habilitação de senha do cartão SD
- Em erros de compilação onde a variável não existe mais o compilador não mostrava o nome da variável
- Quando dois nós de estruturas de variáveis eram expandidos na tabela de variáveis ao mesmo tempo uma das estruturas exibia o nome das variáveis como ???
- As variáveis amostradas no log de eventos não estavam apresentando os valores corretamente
- Correção de desvio funcional no compilador quando qualquer argumento de um USERFB fosse membro de uma estrutura ou instância. Neste caso o número de bits que estava sendo considerado era o tamanho de toda a estrutura, quando o correto deveria ser somente o tamanho deste membro
- Correção de desvio funcional no compilador quando haviam arquivos Ladder diferentes chamando USERFB com instâncias com mesmo nome, ocasionalmente, o compilador invertia a chamada dos USERFBs. Estava sobreescrevendo o arquivo de argumento do USERFB, quando o nome da instância de um USERFB era o mesmo, estando em arquivos diferentes
- Não permitir criação de pastas e arquivos relacionados a USERFB com nomes inválidos
- Bloqueio na edição do componente Text quando este chega ao limite de seu tamanho
- Na compilação dos alarmes, eram exibidos nome de variáveis errados em caso de erro
- Correção na validação durante criação de nova tarefa
- Correção durante compilação em caso de variáveis duplicadas: o compilador não exibia o nome da variável duplicada
- Correção no recurso de comparação: os arquivos Ladder não estavam sendo comparados de maneira correta, em situações específicas havia pequenas divergências que não eram identificadas

WPS V1.60

Funcionalidades:

- Bloco do usuário USERFB

- Comparação do recurso na aplicação com o recurso do equipamento
- Download do setup do equipamento e da configuração da CANopen
- Novo editor de telas mais rápido e fácil de utilizar
- Opções de habilitar e desabilitar alarmes no editor

Melhorias:

- Gerenciamento de projetos:
 - o Salvar como... na configuração e recurso
 - o Procurar para as variáveis em todo o recurso
 - o Propriedades da pasta de recursos
 - o Edição de membros da estrutura
 - o Indicação nos arquivos abertos no editor para indicar se pertencem à seleção principal
 - o Destaque nos programas não utilizados em nenhuma tarefa
 - o Link na janela de saída do compilador
- Telas e alarmes:
 - o Redimensionamento automático do componente Message
 - o Teclas de atalho na edição dos alarmes
 - o Impressão das telas com suas propriedades
- Editor Ladder:
 - o Permitir alterar tipos de contato ou bobina já inseridas no Ladder
 - o Mudança das variáveis de saída dos contadores de BYTE para BOOL para utilização em contatos
- Variáveis:
 - o Possibilidade de apagar múltiplas variáveis da janela variáveis
 - o Desabilitar a janela de variáveis ao fechar ou alterar seleção do recurso
 - o Seleção do endereço da variável através do endereço Modbus

Correções:

- Aumento do número de tentativas antes da apresentação das mensagens de erro durante a monitoração on-line
- Correção da falha ao gravar as variáveis renomeadas. Após renomear as variáveis, o botão salvar não estava sendo habilitado e as alterações não estavam sendo gravadas
- Alteração do padrão do arquivo de importação de variáveis .csv para ; ao invés de ,
- Bloqueio da ação de renomear no grupo GLOBAL_IO
- Alteração do tamanho da área de memória máxima das telas
- Compilador apresentava erro quando compilava estrutura com elemento inválido.
- Alteração para aceitar variáveis repetidas no grupo GLOBAL_IO
- Habilitado os tipos de dados de 32 bits (DWORD, DINT, UDINT, REAL) para as variáveis de rede
- Falha na compilação das variáveis do watchdog das tarefas Start, Stop e Main
- Falha na compilação de variáveis String utilizando a sintaxe STRING# nos blocos ReadRecipe e WriteRecipe. Foi corrigido para gerar erro nesta situação
- Falha ao utilizar o bloco STORE de uma variável REAL para um membro de uma estrutura REAL
- Correção da falha ao gravar as variáveis de monitoração. As novas variáveis criadas não estavam sendo gravadas
- Correção da ordenação dos campos numéricos na tabela de variáveis. Ao ordenar a coluna da tabela os campos numéricos não ficavam ordenados corretamente
- Bloqueio da edição das variáveis de sistema. Era possível editar a variável e duplicá-la
- Correção da falha na exportação do recurso. Não estavam sendo copiados todos os arquivos internos.
- Correção da falha ao importar recurso com a opção sobrescrever habilitada. A configuração não estava sendo sobrescrita corretamente.
- Correção para remoção da borda de seleção do componente após cópia de componente em diferentes telas

- Correção para diminuir o consumo de memória ao carregar as telas
- Problema no alinhamento das estruturas contendo o tipo de dado BOOL. Foi observado que a monitoração do array de BOOL estava apresentando os valores incorretos
- Correção da edição do array do bloco SDO_Write

WPS V1.50

Funcionalidades:

- Upload de programa
- Forçar E/S
- Configuração de idioma do PLC300 no setup
- Opção de apresentação de variáveis nas telas de alarme
- Opção de preenchimento com zeros nos campos Numeric Input e Numeric Output nas telas de usuário
- Opção de download:
 - o Inicializar variáveis voláteis e de saída
 - o Parar/Iniciar a execução do programa automaticamente
- Menu de comandos on-line para as operações de gravação e carregamento do programa, setup e firmware no SD card, protegidos por senha.

Melhorias:

- Impressão dos diagramas Ladder

Correções:

- Aumento do número de tentativas antes da apresentação das mensagens de erro durante a monitoração on-line
- Correção da falha ao gravar as variáveis renomeadas. Após renomear as variáveis o botão salvar não estava sendo habilitado e as alterações não estavam sendo gravadas
- Alteração do padrão do arquivo de importação de variáveis .csv para ; ao invés de ,.
- Bloqueio da ação de renomear no grupo GLOBAL_IO
- Alteração do tamanho da área de memória máxima das telas
- Compilador apresentava erro quando compila estrutura com elemento inválido.
- Alteração para aceitar variáveis repetidas no grupo GLOBAL_IO
- Habilitado os tipos de dados de 32 bits (DWORD, DINT, UDINT, REAL) para as variáveis de rede
- Falha na compilação das variáveis do watchdog das tarefas Start, Stop e Main
- Falha na compilação de variáveis String utilizando a sintaxe STRING# nos blocos ReadRecipe e WriteRecipe. Foi corrigido para a gerar erro nesta situação
- Falha ao utilizar o bloco STORE de uma variável REAL para um membro de uma estrutura REAL
- Correção da falha ao gravar as variáveis de monitoração. As novas variáveis criadas não estavam sendo gravadas
- Correção da ordenação dos campos numéricos na tabela de variáveis. Ao ordenar a coluna da tabela os campos numéricos não ficavam ordenados corretamente
- Bloqueio da edição das variáveis de sistema. Era possível editar a variável e duplicá-la
- Correção da falha na exportação do recurso. Não estavam sendo copiados todos os arquivos internos.
- Correção da falha ao importar recurso com a opção sobrescrever habilitada. A configuração não estava sendo sobrescrita corretamente.
- Correção para remoção da borda de seleção do componente após cópia de componente em diferentes telas
- Correção para diminuir o consumo de memória ao carregar as telas
- Problema no alinhamento das estruturas contendo o tipo de dado BOOL. Foi observado que a monitoração do array de BOOL estava apresentando os valores incorretos

- Correção da edição do array do bloco SDO_Write

WPS V1.40

Funcionalidades:

- WPS traduzido para os idiomas Inglês e Espanhol
- Central de atualizações

Correções:

- Mover as variáveis entre grupos pelo editor Ladder
- Abrir gerenciador de arquivos Modbus

WPS V1.30

Funcionalidades:

- [Atualização Editor de Variáveis](#)
- Configuração da Ethernet no setup
- Configuração de Estruturas
- Configuração de Receitas
- Configuração de Log de Eventos
- Gerenciador de Arquivos Modbus
- Editor Ladder:
 - o [Mudanças na Área de Trabalho](#)
 - o [Utilização de Literais](#)
 - o [Utilização de Arrays](#)
 - o [Utilização de Instâncias e Estruturas](#)
 - o [Argumentos Opcionais para Blocos Funcionais](#)
 - o [Novas Funcionalidades Copiar/Colar](#)
 - o [Novas Funcionalidades Monitoração On-line](#)

Correções:

- Otimização do editor Ladder de modo a garantir arquivos Ladder com até 200 rungs e consumo de memória e CPU moderados. Na versão 1.20 do WPS, nessa situação ocorriam lentidão e travamentos

PLC300:

- Novidades e Correções do PLC300.

WPS V1.20

- Comunicação com equipamento PLC300 V1.2X
- Programação de tarefas
- Alteração no endereços de marcadores para compatibilidade com o Modbus
- Ferramenta para importar/exportar projetos
- Verificação da versão de firmware do PLC300 no download

- Parar/executar programa como uma ação global
- Persistência do ambiente ao sair
- Melhorias na comunicação
- Opções de download:
 - o Inicializar variáveis retentivas
 - o Limpar histórico de alarmes
 - o Download do código fonte
- Compatível com Windows 7 x64, Windows 7 x86, Windows Vista x64 e Windows Vista x86.

4 Instalação/Desinstalação

[Antes de Instalar](#)

[Instalando](#)

[Desinstalando](#)

[Driver USB WEG](#)

[Driver USB FTDI](#)

4.1 Antes de Instalar

Verifique os seguintes itens antes de instalar o WPS v2.3X:

- Se o microcomputador possui os [requisitos do sistema](#);
- Se a versão do WPS v2.3X é [compatível com seu equipamento](#);
- Se a sua conta de usuário atual possui privilégios de administrador para instalar o software.

4.2 Instalando

Ao instalar uma versão mais nova do WPS é recomendado que primeiro seja [removida](#) a versão anterior e o sistema seja reinicializado, para ter certeza que a nova instalação será executada corretamente.

Feche todos os programas em execução para evitar interferência com o processo de instalação e siga os passos descritos abaixo.

1. Dar um duplo clique no ícone do instalador para iniciar o instalador. O programa de instalação iniciará e uma janela de boas vindas aparecerá.
2. No assistente de instalação, responder à Licença de Uso, então pressionar botão Avançar.
3. Especificar uma pasta vazia dentro da qual será instalada o WPS v2.3X. Certificar-se que a localização da instalação está correta e que existe espaço suficiente em disco para a instalação.
4. Confirmar para criar uma pasta vazia.
5. Selecionar o tipo de instalação que desejar.
6. Selecionar o grupo de programa para criar os atalhos do WPS v2.3X.
7. Selecionar os atalhos de programas adicionais: criar um atalho no desktop e criar um atalho para inicialização rápida.
8. Clicar no botão Instalar para iniciar a instalação.
9. Clicar no botão Concluir para concluir o setup.

4.3 Desinstalando

Se necessário, remover o WPS v2.3X usando os procedimentos abaixo.

**NOTA!**

Sempre utilize o aplicativo Programas e Recursos para remover o WPS. Não apague arquivos e pastas manualmente.

1. Do menu Iniciar, selecionar Painel de Controle.
2. Dar um duplo clique no ícone Programas e Recursos.
3. Selecionar WPS v2.3X da lista e clicar no botão Remover.
4. Seguir as instruções para remover o software.

4.4 Driver USB WEG

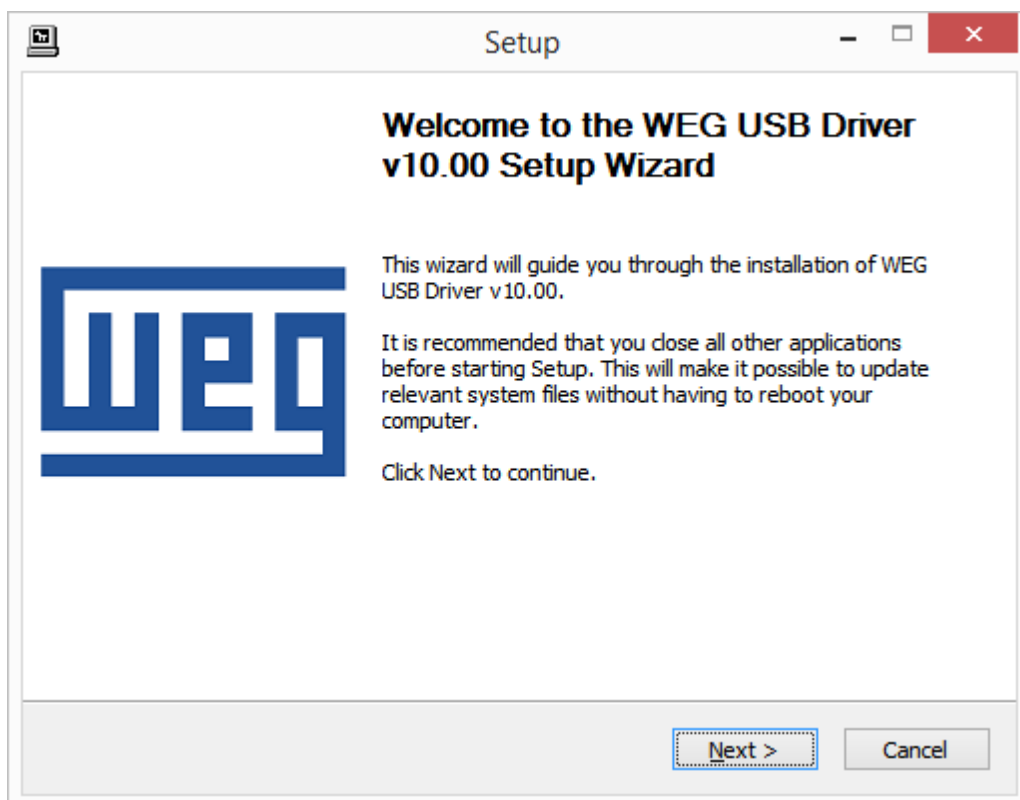
É necessário o driver USB WEG estar instalado no microcomputador para a comunicação com os seguintes equipamentos:

- PLC300, a interface de comunicação é a porta USB denominada XC5;
- SCA06, a interface de comunicação é a porta USB denominada X3.

O driver USB WEG está disponível na pasta USB_Driver\USBIO onde o WPS está instalado.

Instalação

1. Utilize o instalador disponível do WEG USBIO. Você pode fazer *download* deste instalador pelo site <http://www.weg.net>;
2. Feche todas as aplicações na sua máquina. Se você está usando um *software* anti-virus ou *firewall*, feche-os (ou desabilite suas funções);
3. Inicie o instalador e siga os menus;
4. No final da instalação é solicitado ao usuário para desconectar e conectar o equipamento.



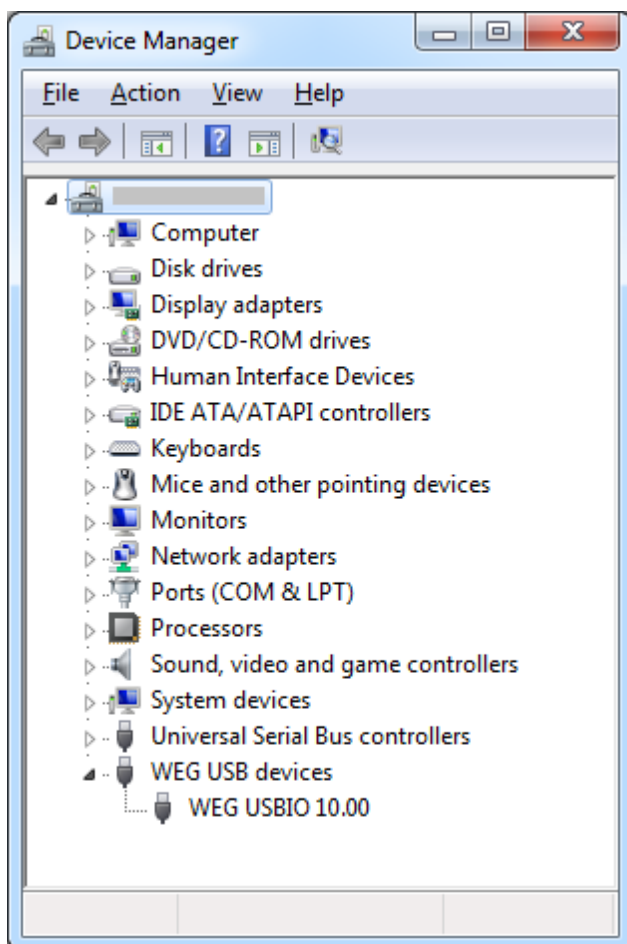
Durante a instalação do *driver*, uma entrada em **Programas e Recursos** no **Painel de Controle** do Windows é criado.

Esta entrada pode ser usada para desinstalar o *software* do *driver*.

Verificando a Instalação

Você pode verificar se a instalação foi bem sucedida no **Gerenciador de Dispositivos do Windows**. Para isso, o equipamento precisa estar conectado ao PC.

No Gerenciador de Dispositivos você deve encontrar, próximo ao fim da lista, a entrada **WEG USB devices** contendo **WEG USBIO V10.00**. Isto indica que a instalação foi realizada com sucesso.



Verifique com o fabricante se existe algum *driver* USB mais novo.

Desinstalação

Se você precisa desinstalar o *driver* USB do PC, siga o procedimento descrito abaixo.

1. Do menu Iniciar, selecionar Painel de Controle;
2. Dar um duplo clique em Programas e Recursos;
3. Selecionar o *software* WEG USB Driver da lista e clicar em Remover;
4. Seguir as instruções para remover o *software*.

Atualização

O procedimento de atualização é semelhante ao de instalação, visto que o instalador detecta automaticamente se já existe uma versão instalada e apenas uma atualização é necessária.

4.5 Driver USB FTDI

É necessário o driver USB FTDI estar instalado no microcomputador para a comunicação com o seguinte equipamento:

- CFW300 com módulo plug-in CFW300-CUSB; a interface de comunicação é a porta serial USB.

O driver USB FTDI está disponível na pasta USB_Driver\FTDI.

Driver Não Instalado

Quando um driver FTDI não está instalado no Windows, o gerenciador de dispositivos indica que o equipamento conectado não foi reconhecido.

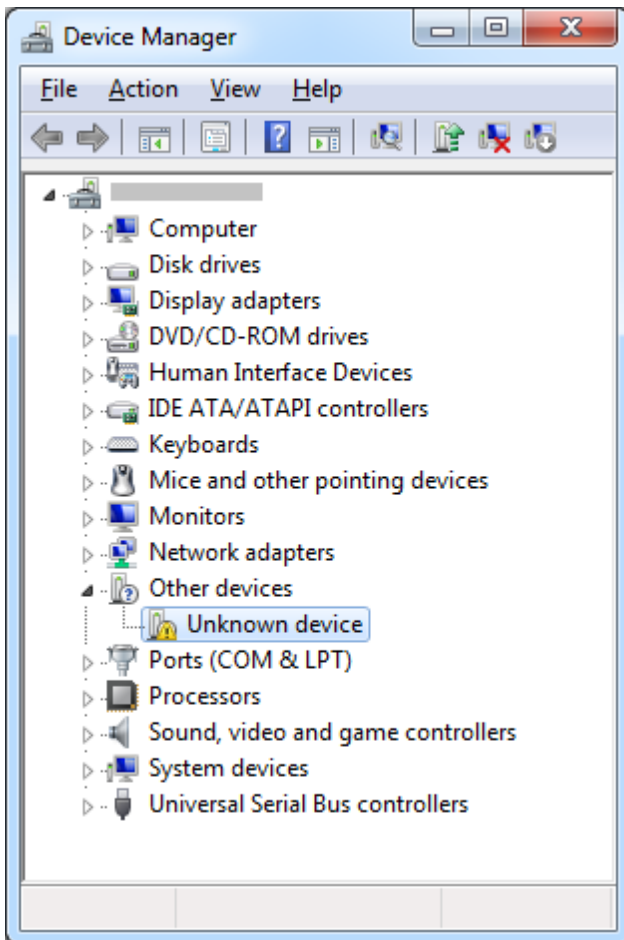
Segue abaixo um exemplo.

Exemplo

Para verificar se um driver USB FTDI está instalado ou não no Windows, execute os seguintes passos:

1. Insira o conector mini-B do cabo usb no conector usb do equipamento;
2. Insira o conector tipo A do cabo usb na porta usb do computador;
3. Quando um driver FTDI não está instalado no Windows, uma entrada informando dispositivo desconhecido aparece no gerenciador de dispositivos do Windows.

O nome do computador no gerenciador de dispositivo está propositalmente ilegível.



Instalação

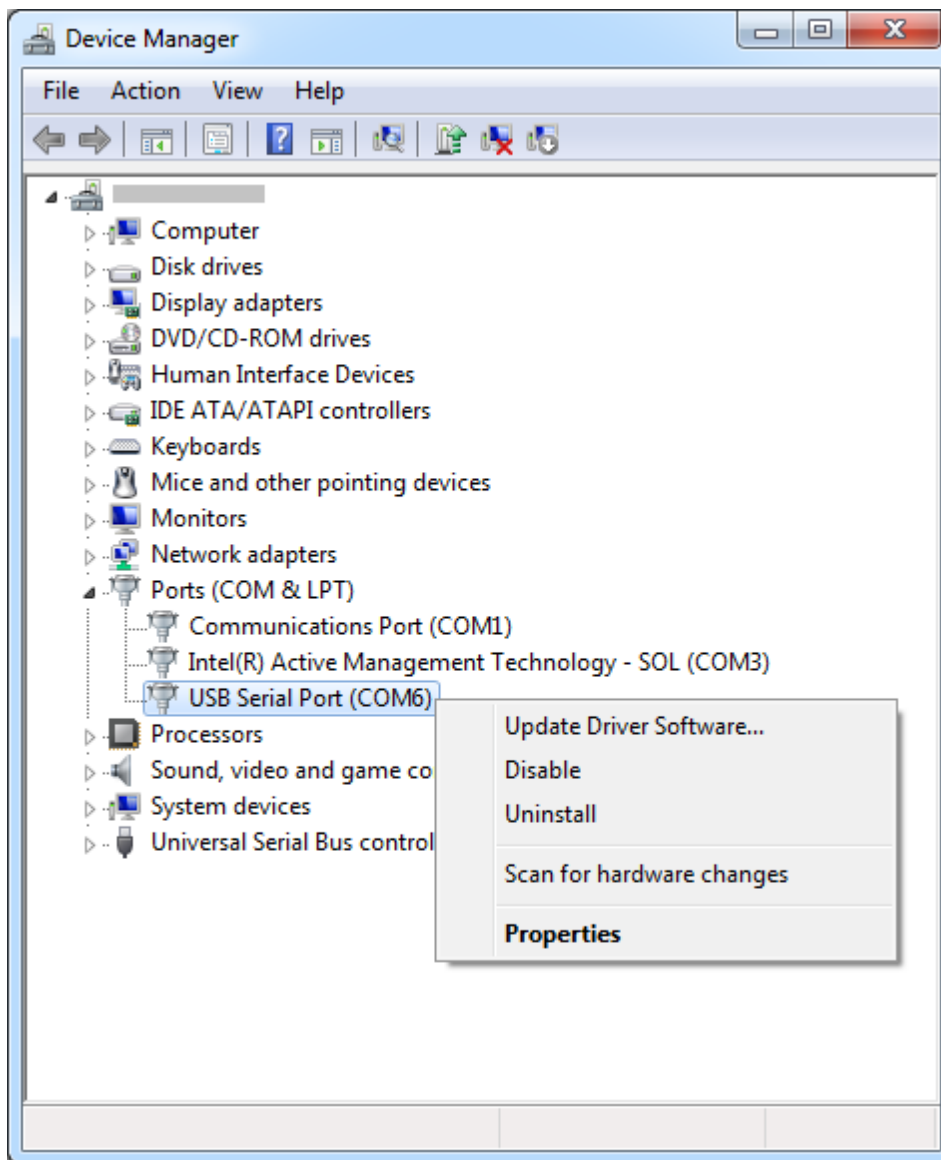
Execute o instalador e siga as instruções de instalação.

Verificando a Instalação

Quando o driver USB FTDI está corretamente instalado, verifique o gerenciador de dispositivo do Windows.

Clique com o botão direito do mouse sobre a porta serial USB, um menu será mostrado.

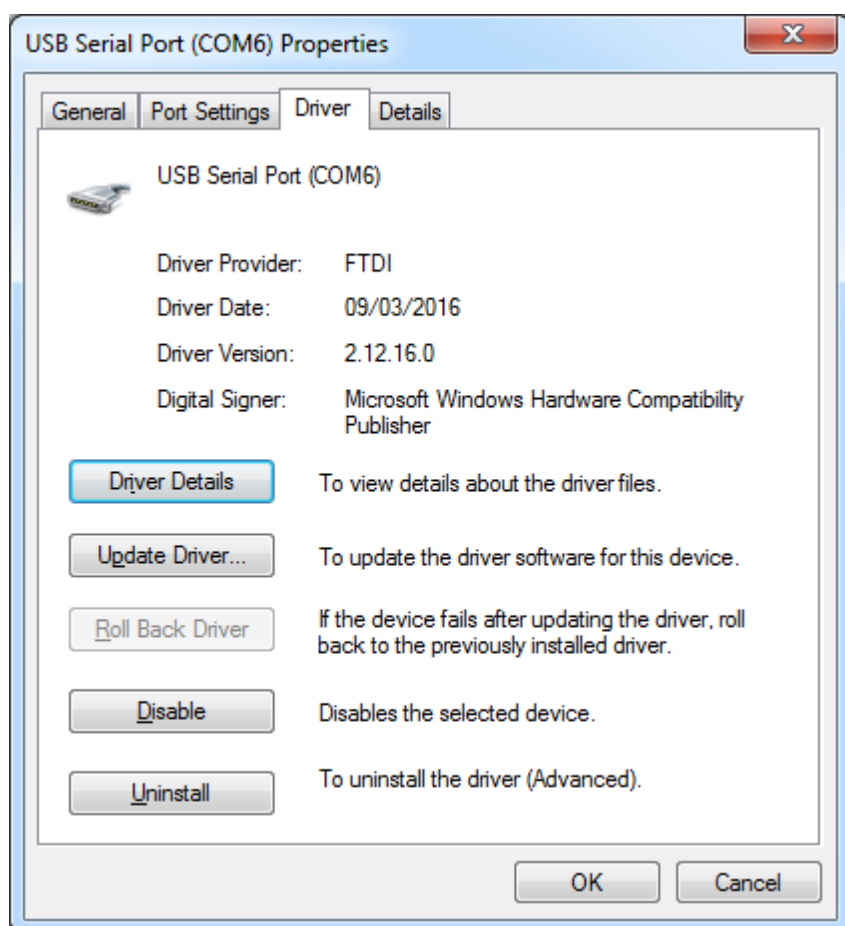
O nome do computador no gerenciador de dispositivo está propositalmente ilegível.



No menu popup, clique em Propriedades

Na janela de propriedades da porta serial USB, clique na aba Driver.

As informações do driver FTDI são mostradas nesta janela.



5 Conhecendo o Ambiente

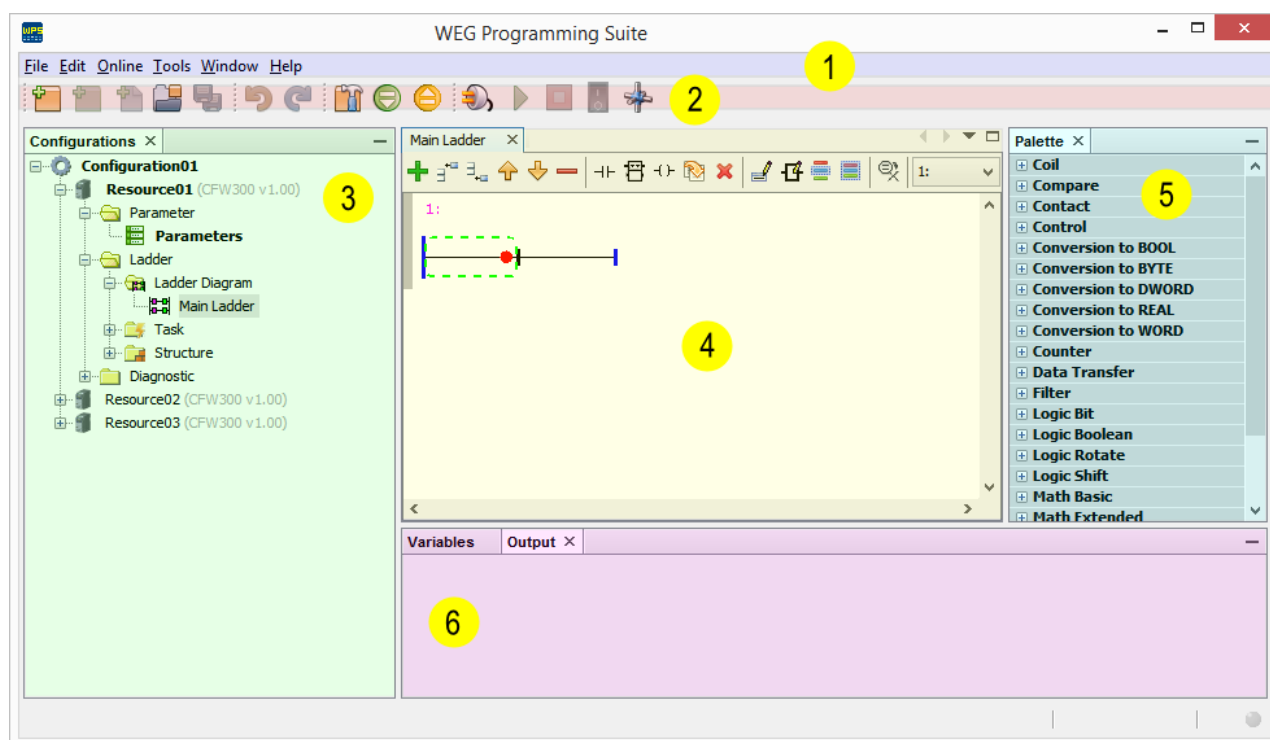
[Ambiente](#)

[Estrutura de uma Configuração](#)

5.1 Ambiente

A janela principal do WPS v2.3X pode ser dividida em 6 partes principais:

1. **Menus:** exibe as diversas opções de edição, visualização, comunicação e ajuda do ambiente de desenvolvimento;
2. **Barra de Ferramentas:** exibe os principais comandos do software;
3. **Janela das Configurações:** exibe as configurações, onde cada configuração contém recursos;
4. **Janela do Editor:** parte principal do ambiente de desenvolvimento, onde ocorre a edição dos componentes que irão compor o arquivo fonte;
5. **Janela da Paleta:** contém os componentes a serem inseridos no editor Ladder através do drag 'n drop;
6. **Janela de Saída:** exibe mensagens de compilação e download.



5.2 Estrutura de uma Configuração

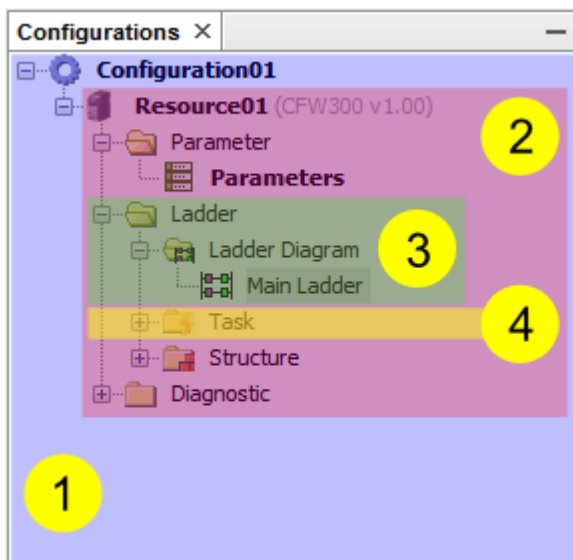
O software WPS segue as deliberações da norma IEC 61131-3.

Sendo assim, toda aplicação desenvolvida nele possui uma configuração com a seguinte estrutura

hierárquica:

- **Configuração:** está no nível mais alto da hierarquia, definindo todos os elementos de software contidos em uma aplicação que interagem entre si para desempenhar as funções de controle.
 - o **Recurso:** segundo nível da hierarquia, representa qualquer elemento com capacidade de processamento para execução de programas. No WPS, os recursos são independentes entre si, e cada um possui um produto atrelado a ele. As variáveis globais de uma aplicação estão neste escopo.
 - **POU (Unidade de Organização de Programas):** descrevem as instruções que serão executadas, em que ordem serão executadas e como estas interagem entre si. Blocos funcionais, funções e programas estão contidos nesta categoria. No WPS, o POU que mais se destaca é o Diagrama Ladder. As variáveis locais de uma aplicação estão neste escopo, cada uma restrita ao seu próprio POU.
 - **Tarefas:** processos que controlam a ordem e o tempo de execução dos POU's.

A figura a seguir apresenta o esquema desta hierarquia no WPS.



Legenda:

1. Configuração;
2. Recurso;
3. POU's;
4. Tarefas;

6 Início Rápido

[Janela Bem Vindo](#)

[Criando Nova Configuração](#)

[Nova Configuração - Equipamento Online](#)

[Nova Configuração - Equipamento Offline](#)

[Criando Novo Recurso](#)

[Novo Recurso - Equipamento Online](#)

[Menu Popup](#)

[Menu Popup - Configuração](#)

[Menu Popup - Recurso](#)

6.1 Janela de Boas Vindas

Esta capítulo fornece algumas informações básicas para você se familiarizar com o software.

Iniciaremos com a janela de boas vindas.

A janela de boas vindas aparece quando o programa é iniciado pela primeira vez.

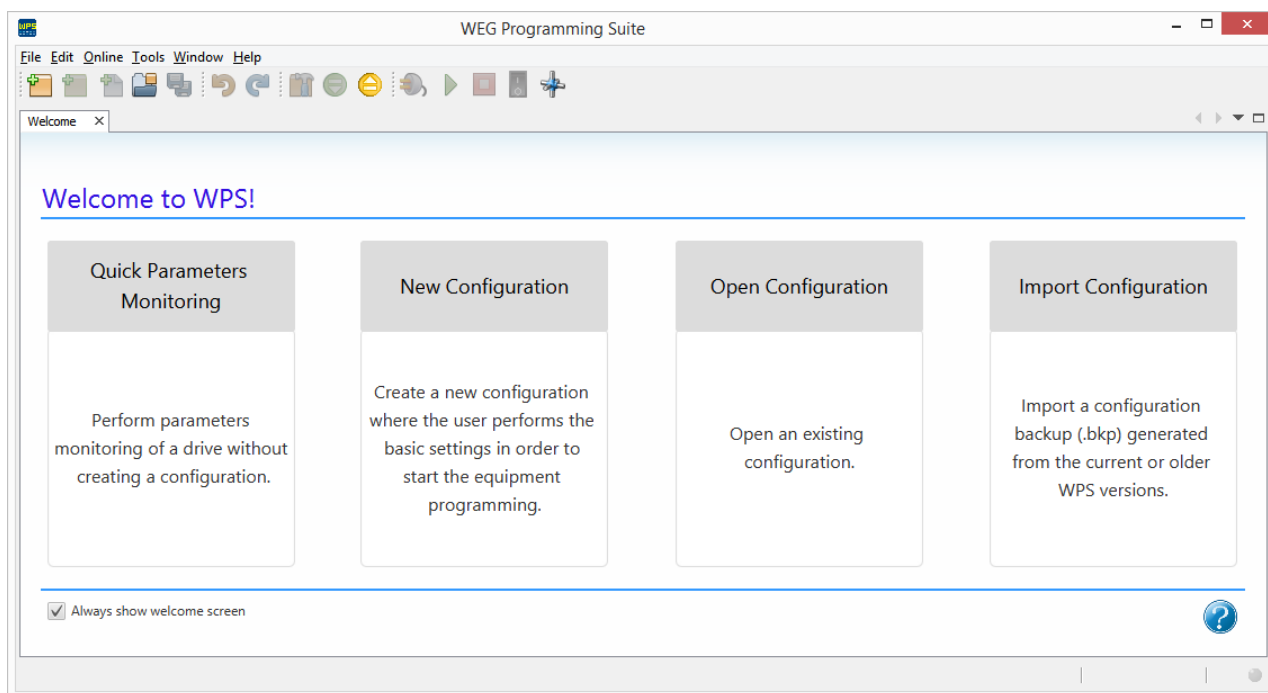
As seguintes opções estão disponíveis nesta janela:

- Monitoração Rápida dos Parâmetros
- Nova Configuração
- Abrir Configuração
- Importar Configuração

Existe ainda uma opção **Sempre mostrar janela bem vindo**, que será explicado abaixo.

O usuário também pode fechar esta janela, não executando nenhuma das opções, e utilizar sua própria metodologia de trabalho.

A figura a seguir apresenta a janela de boas vindas do WPS.

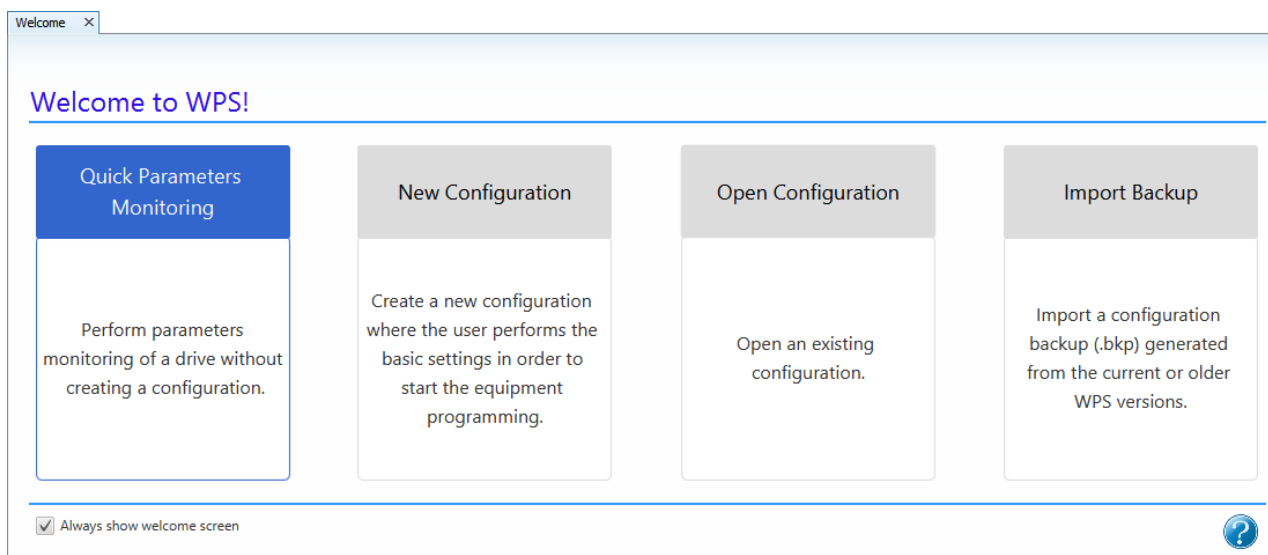


Monitoração Rápida dos Parâmetros

A monitoração rápida dos parâmetros permite visualizar e alterar os parâmetros online, porém não é possível criar uma nova configuração nem salvar os parâmetros no computador.

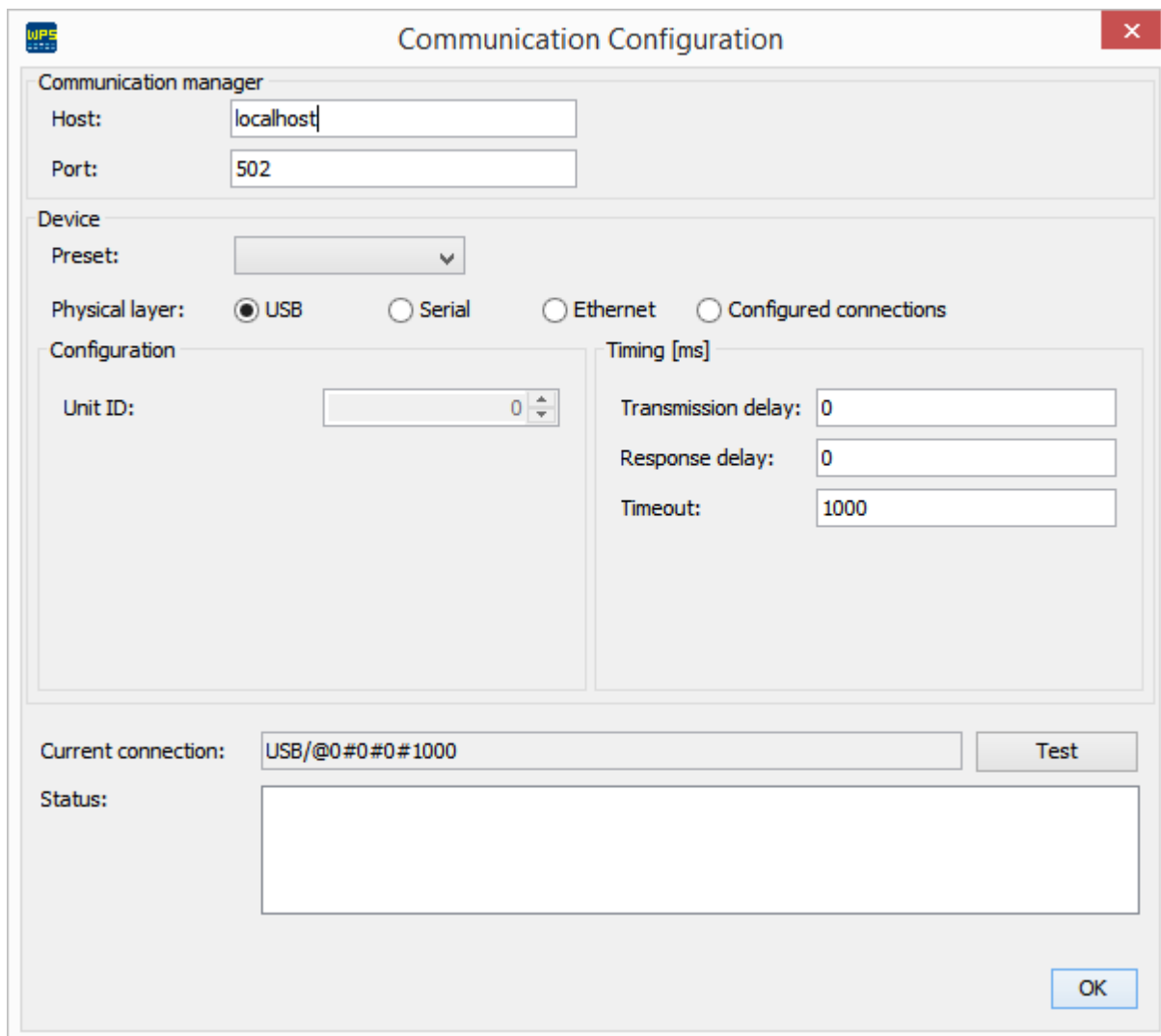
1) Seleção da Função

Clique na opção Monitoração Rápida dos Parâmetros.



2) Configuração da Comunicação

A próxima janela define as opções de comunicação com o equipamento.



The image shows a 'Communication Configuration' dialog box with the following fields and controls:

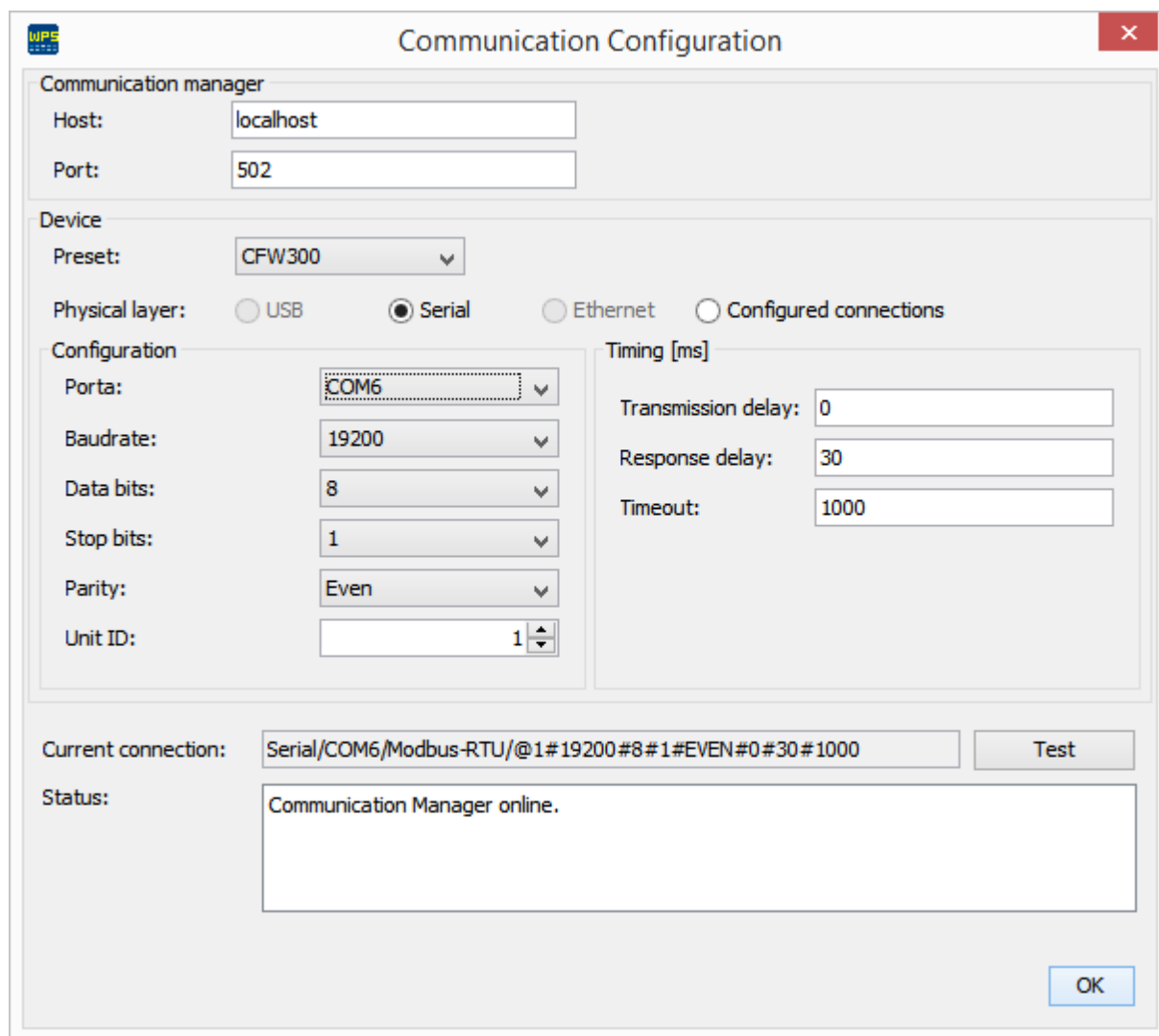
- Communication manager**
 - Host:
 - Port:
- Device**
 - Preset:
 - Physical layer: ☒ USB ☐ Serial ☐ Ethernet ☐ Configured connections
- Configuration**
 - Unit ID:
- Timing [ms]**
 - Transmission delay:
 - Response delay:
 - Timeout:
- Current connection:**
- Status:**
-

3) Seleção das Opções de Comunicação

Escolha as opções corretas de comunicação.

Se você selecionar um equipamento, a configuração padrão de comunicação do equipamento será carregada na janela.

No entanto verifique se a configuração da janela é igual ao do equipamento. Se não for, atualize esta janela de acordo com a configuração do equipamento.



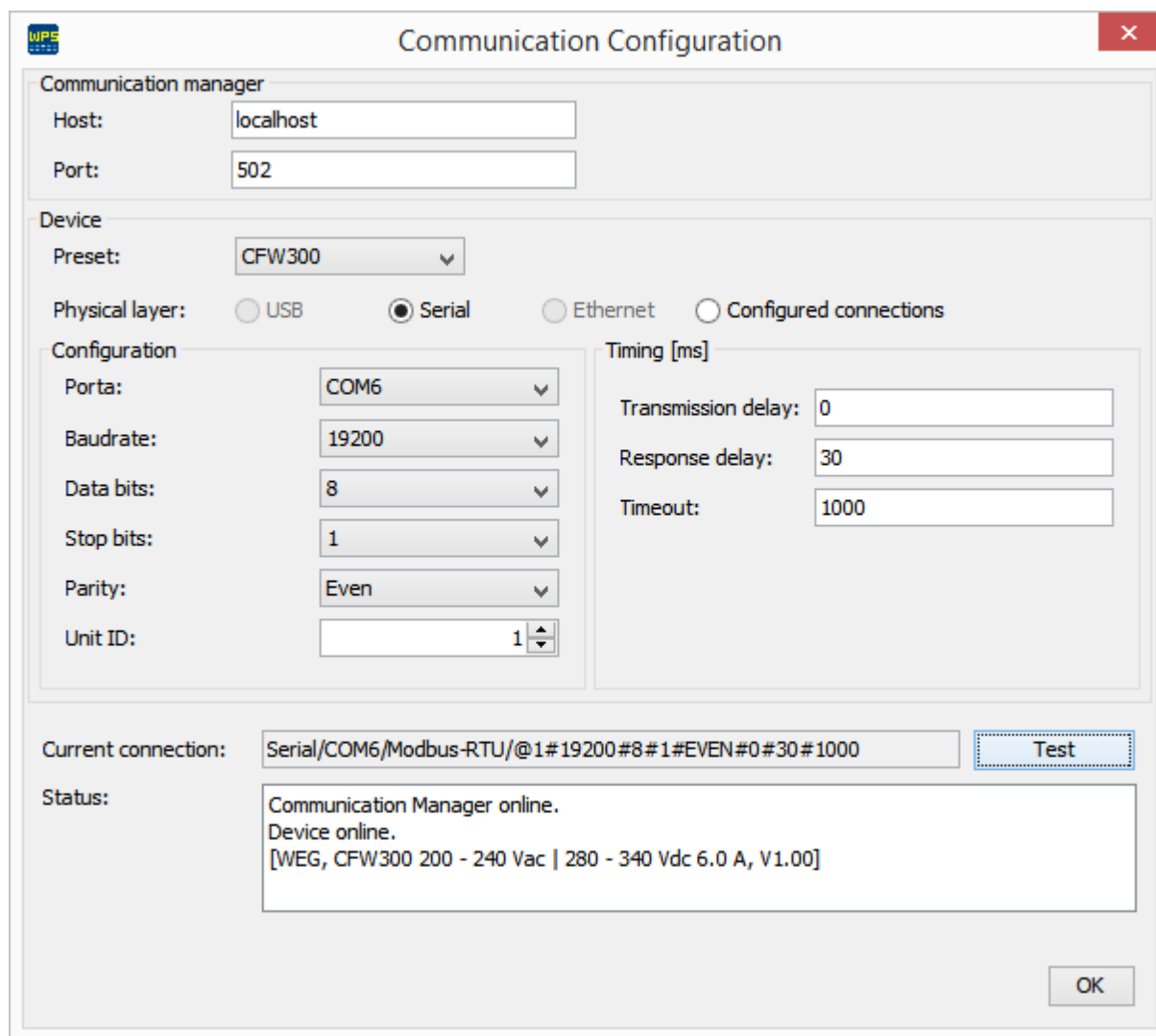
The image shows a 'Communication Configuration' dialog box with the following sections:

- Communication manager:** Host: localhost, Port: 502.
- Device:** Preset: CFW300.
- Physical layer:** Radio buttons for USB, Serial (selected), Ethernet, and Configured connections.
- Configuration:** Porta: COM6, Baudrate: 19200, Data bits: 8, Stop bits: 1, Parity: Even, Unit ID: 1.
- Timing [ms]:** Transmission delay: 0, Response delay: 30, Timeout: 1000.
- Current connection:** Serial/COM6/Modbus-RTU/@1#19200#8#1#EVEN#0#30#1000, with a Test button.
- Status:** Communication Manager online.
- Buttons:** OK at the bottom right.

4) Testando a Comunicação

Você pode verificar se a comunicação está correta clicando no botão Testar desta janela.

O campo estado é atualizado. O equipamento conectado aparece no campo estado.



The image shows a 'Communication Configuration' dialog box with the following sections:

- Communication manager:** Host: localhost, Port: 502.
- Device:** Preset: CFW300.
- Physical layer:** Radio buttons for USB, Serial (selected), Ethernet, and Configured connections.
- Configuration:** Porta: COM6, Baudrate: 19200, Data bits: 8, Stop bits: 1, Parity: Even, Unit ID: 1.
- Timing [ms]:** Transmission delay: 0, Response delay: 30, Timeout: 1000.
- Current connection:** Serial/COM6/Modbus-RTU/@1#19200#8#1#EVEN#0#30#1000. A 'Test' button is next to it.
- Status:** A text box showing 'Communication Manager online.', 'Device online.', and '[WEG, CFW300 200 - 240 Vac | 280 - 340 Vdc 6.0 A, V1.00]'. An 'OK' button is at the bottom right.

Em seguida clique OK.

5) Monitoração em Execução

Uma janela com todos os parâmetros do equipamento é apresentada.

O campo Online apresenta o valor real dos parâmetros no equipamento, sendo que a maioria dos parâmetros permite alteração. Parâmetros de leitura não podem ser alterados.

WEG Programming Suite

File Edit Online Tools Window Help

Welcome x Parameters x

Parâmetros

- HMI
- Frequency Inverter Identification
- Command and References
- Motor Control
- I/O
- Faults and Alarms
- Read
- Communication
- SoftPLC

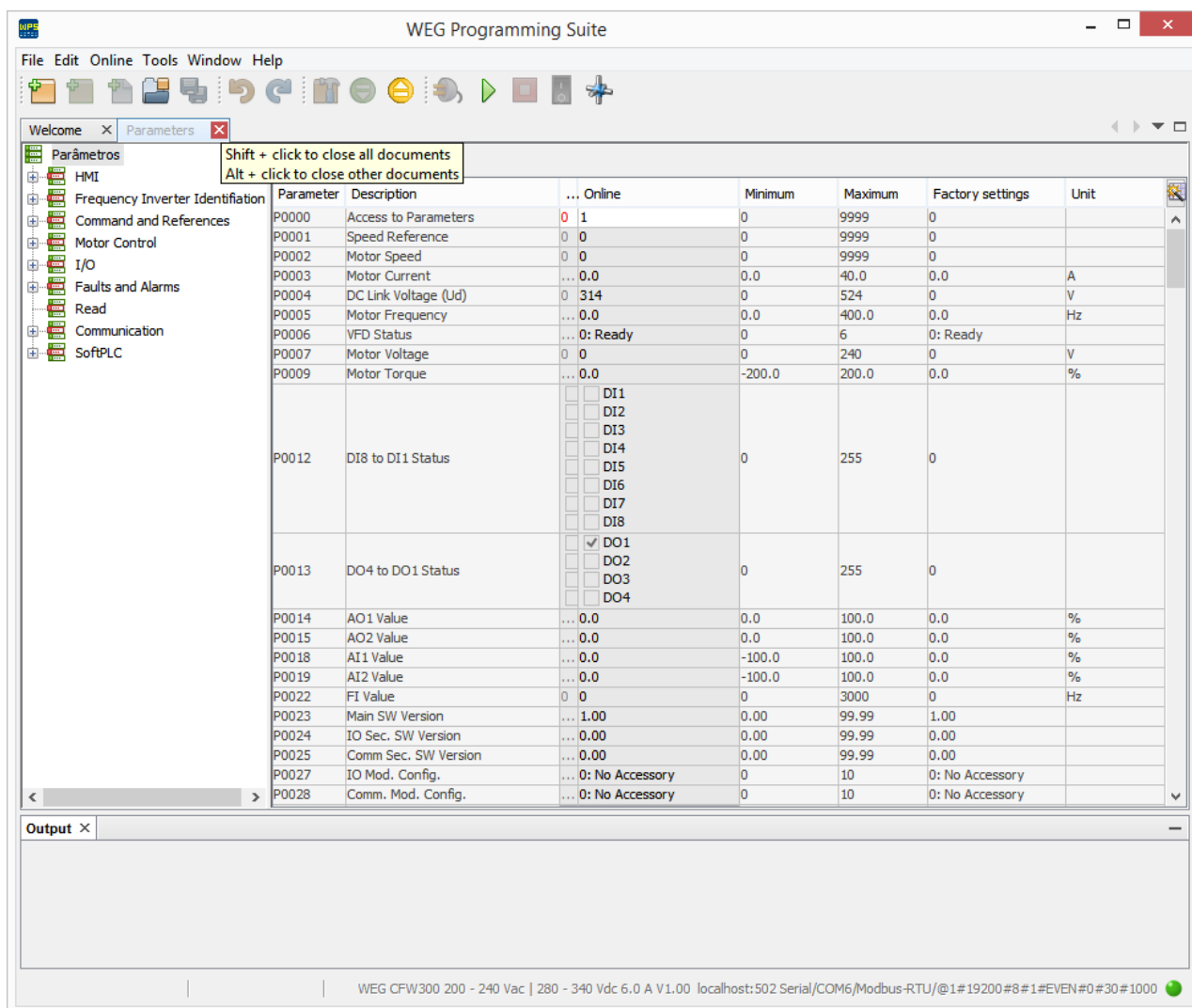
Parameter	Description	Online	Minimum	Maximum	Factory settings	Unit
P0000	Access to Parameters	1	0	9999	0	
P0001	Speed Reference	0	0	9999	0	
P0002	Motor Speed	0	0	9999	0	
P0003	Motor Current	0.0	0.0	40.0	0.0	A
P0004	DC Link Voltage (Ud)	314	0	524	0	V
P0005	Motor Frequency	0.0	0.0	400.0	0.0	Hz
P0006	VFD Status	0: Ready	0	6	0: Ready	
P0007	Motor Voltage	0	0	240	0	V
P0009	Motor Torque	0.0	-200.0	200.0	0.0	%
P0012	DI8 to DI1 Status	☐ DI1	0	255	0	
		☐ DI2				
		☐ DI3				
		☐ DI4				
		☐ DI5				
		☐ DI6				
		☐ DI7				
		☐ DI8				
P0013	DO4 to DO1 Status	☒ DO1	0	255	0	
		☐ DO2				
		☐ DO3				
		☐ DO4				
P0014	AO1 Value	0.0	0.0	100.0	0.0	%
P0015	AO2 Value	0.0	0.0	100.0	0.0	%
P0018	AI1 Value	0.0	-100.0	100.0	0.0	%
P0019	AI2 Value	0.0	-100.0	100.0	0.0	%
P0022	FI Value	0	0	3000	0	Hz
P0023	Main SW Version	1.00	0.00	99.99	1.00	
P0024	IO Sec. SW Version	0.00	0.00	99.99	0.00	
P0025	Comm Sec. SW Version	0.00	0.00	99.99	0.00	
P0027	IO Mod. Config.	0: No Accessory	0	10	0: No Accessory	
P0028	Comm. Mod. Config.	0: No Accessory	0	10	0: No Accessory	

Output x

WEG CFW300 200 - 240 Vac | 280 - 340 Vdc 6.0 A V1.00 localhost:502 Serial/COM6/Modbus-RTU/@1#19200#8#1#EVEN#0#30#1000

6) Fechar a Janela

Para finalizar a monitoração, clique no botão Fechar da janela.

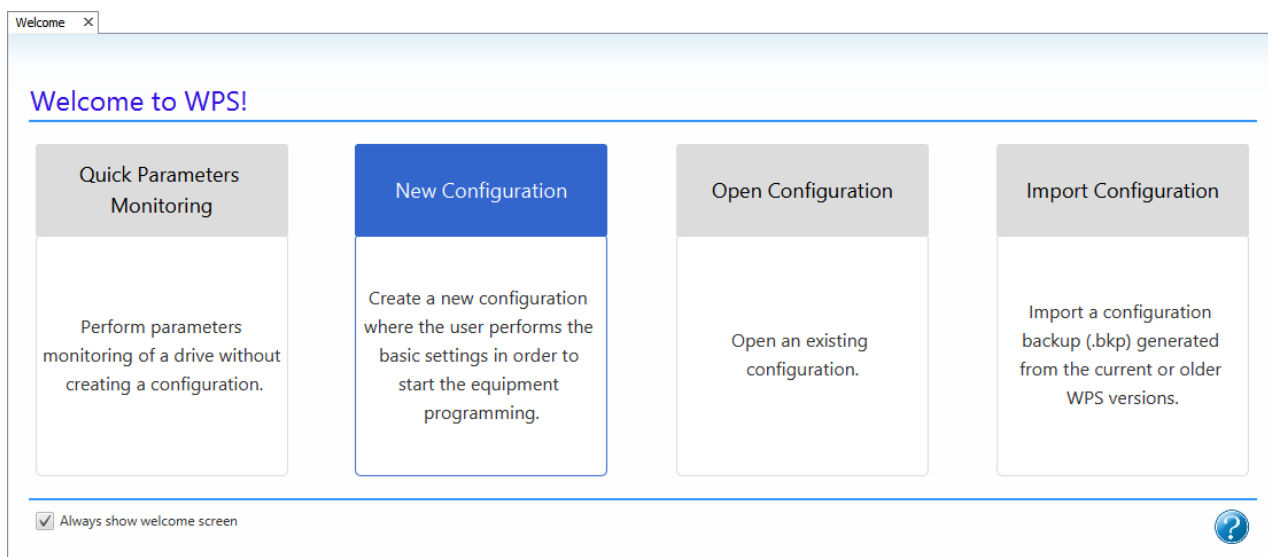


Nova Configuração

Cria uma nova configuração.

1) Seleção da Função

Clique na opção Nova Configuração.



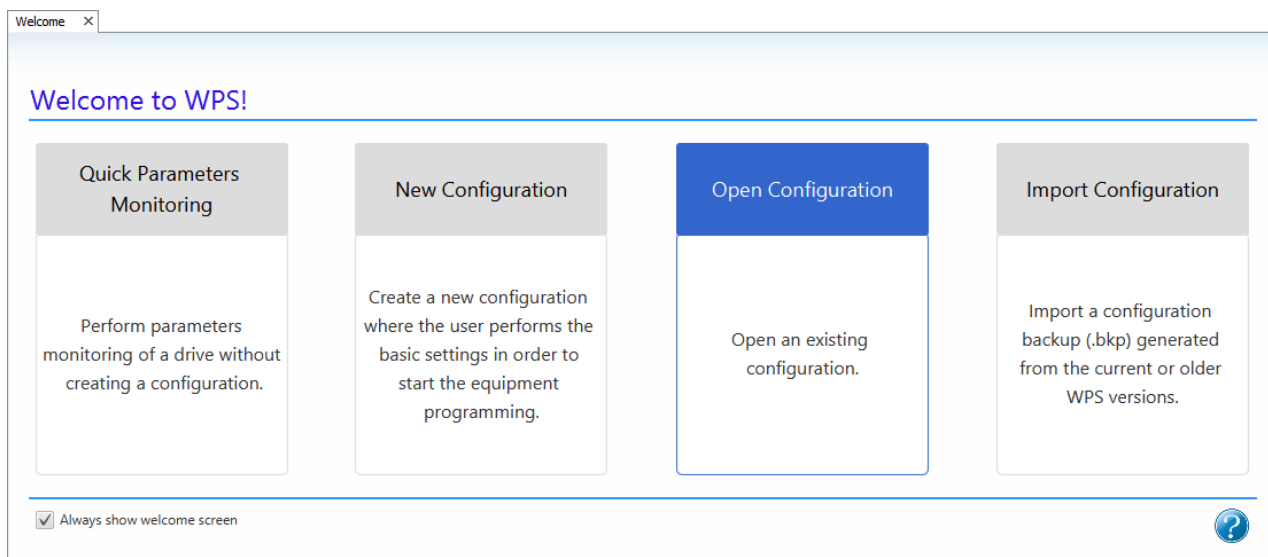
Deste passo em diante, consulte [Criando Nova Configuração](#) para detalhes.

Abrir Configuração

Permite abrir uma configuração previamente criada.

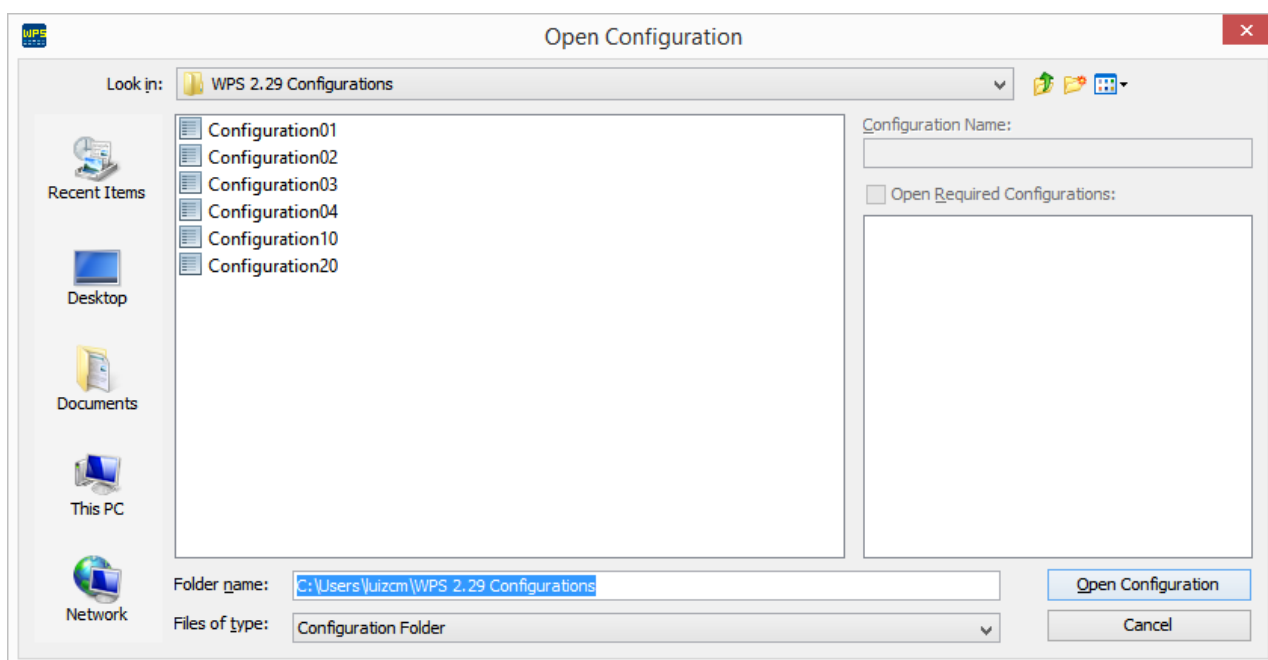
1) Seleção da Função

Clique na opção Abrir Configuração.



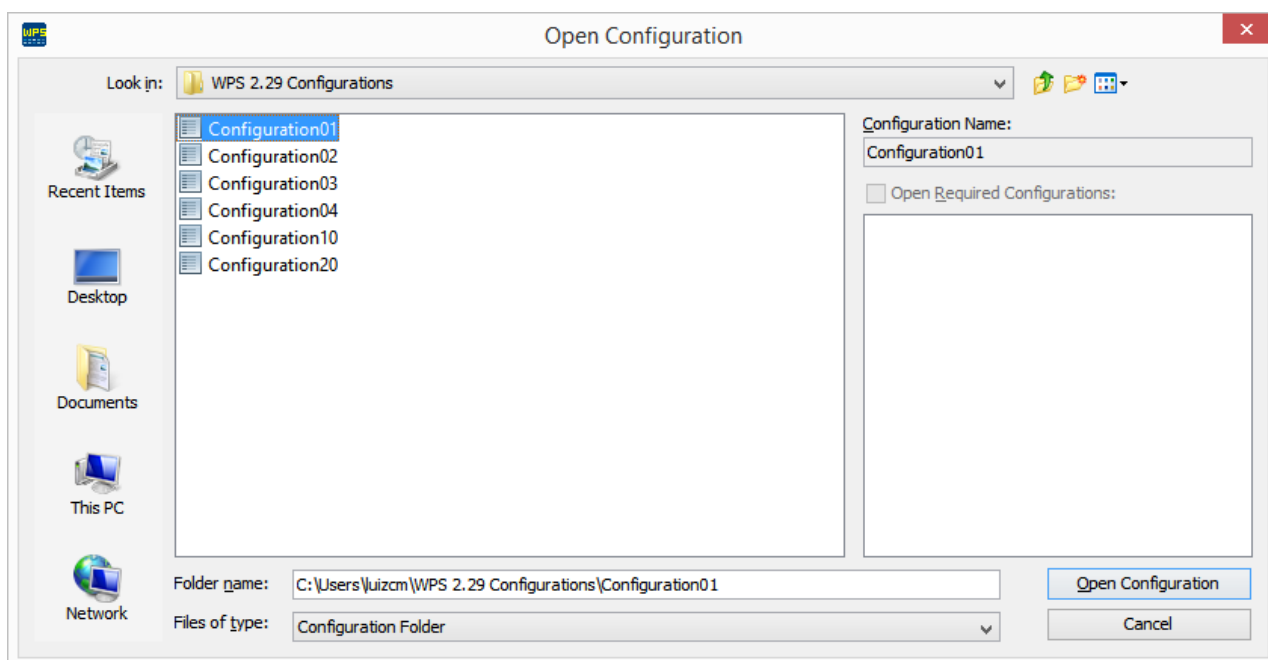
2) Janela Abrir Configuração

Uma janela apresenta todas as configurações criadas.



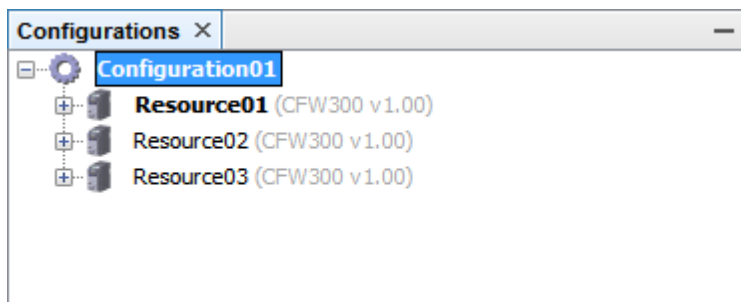
3) Seleção da Configuração

Selecione a configuração desejada e clique no botão Abrir Configuração.



4) Janela das Configurações

Neste momento a configuração é aberta. Na janela de configurações, pode-se visualizar a configuração e os recursos que fazem parte de configuração.

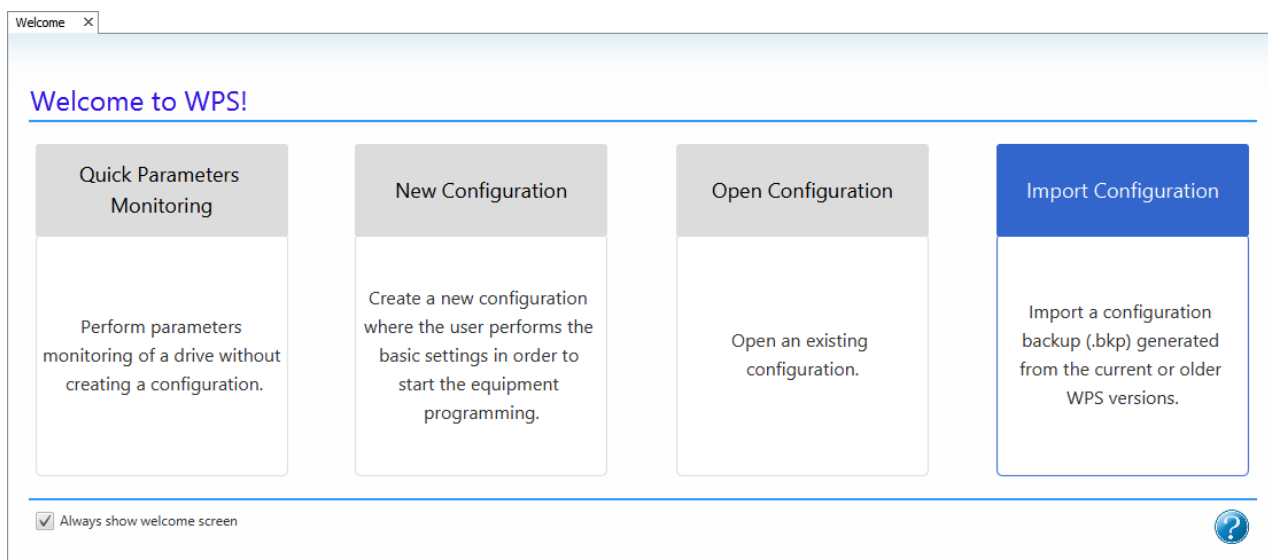


Importar Configuração

Importa uma configuração gerada previamente pelo WPS.

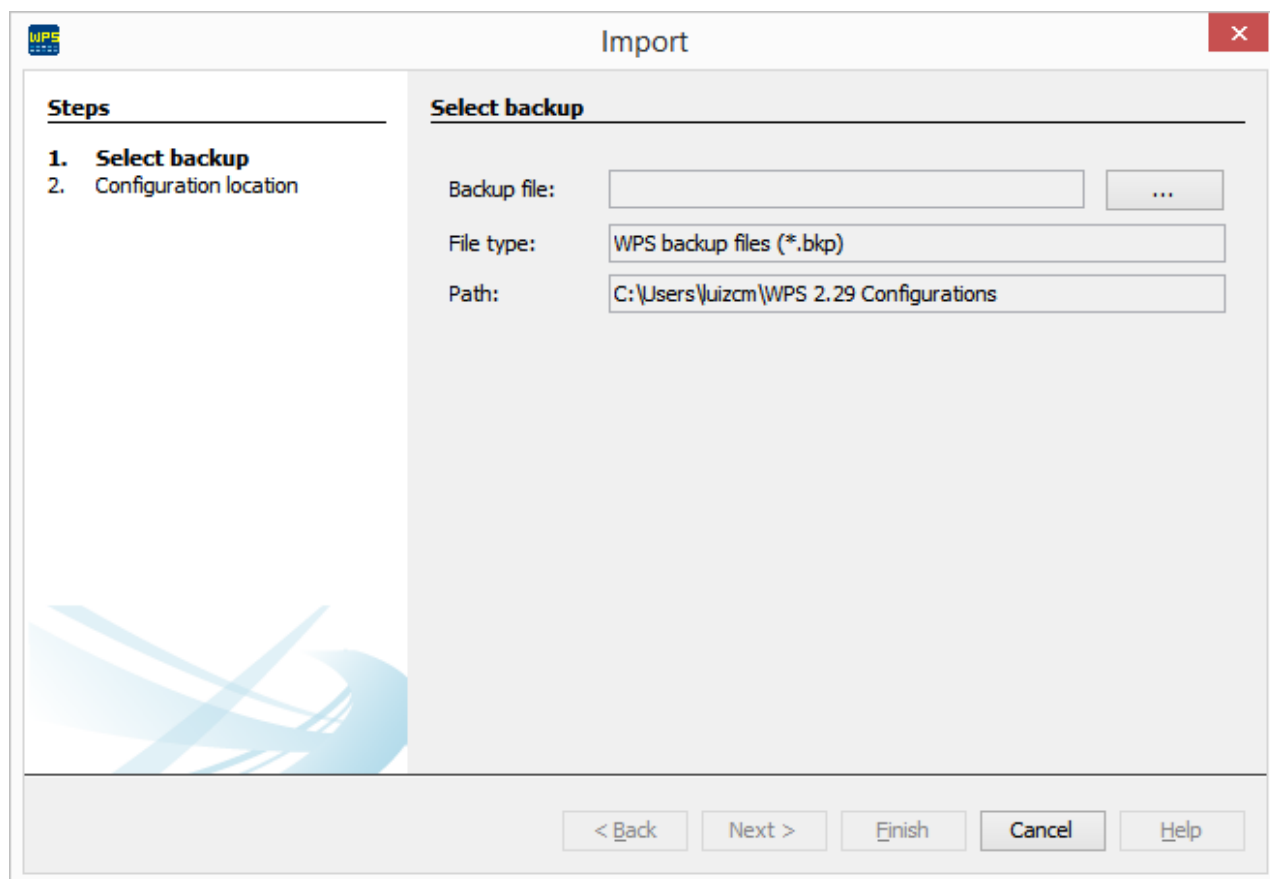
1) Seleção da Função

Clique na opção Importar Configuração.



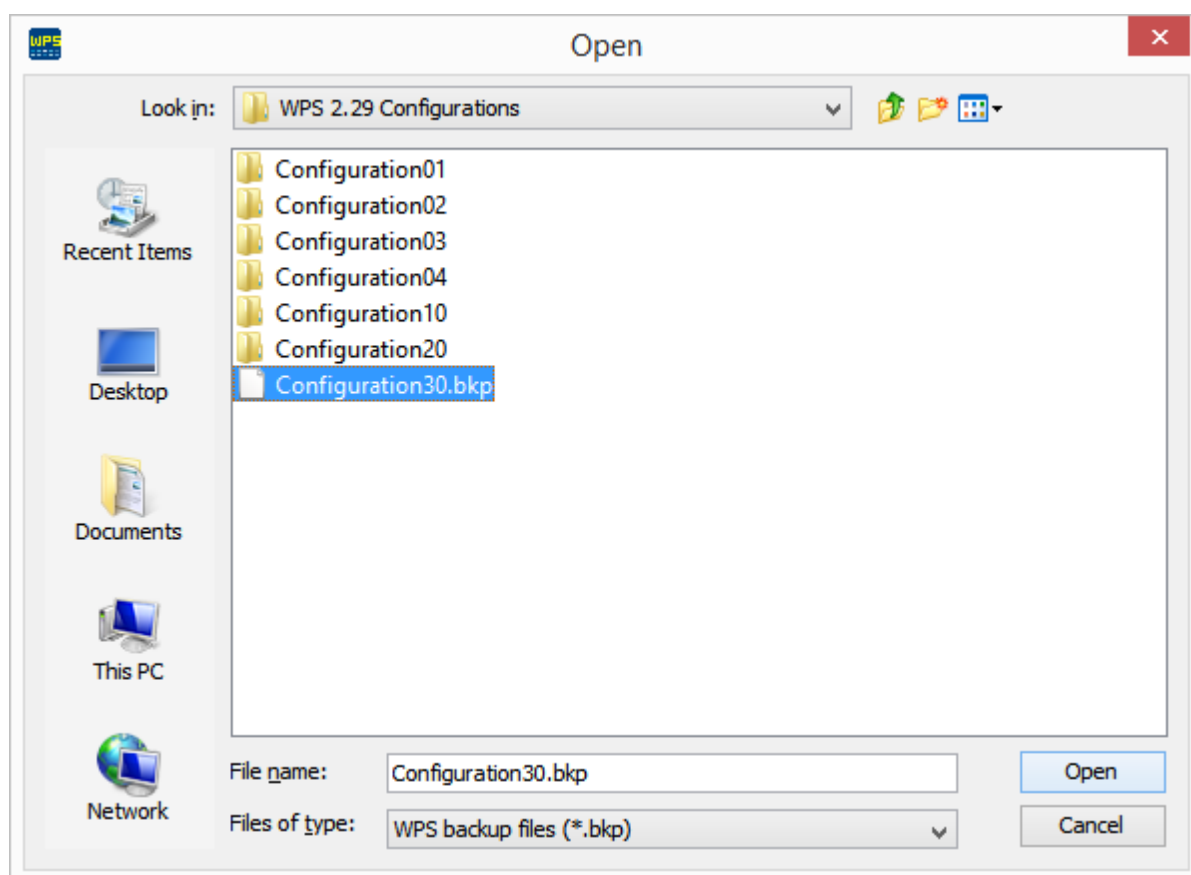
2) Janela Importar

Clique no botão ... para abrir a seleção de arquivo de backup que contem a configuração que deseja importar.



3) Seleção do Arquivo de Backup

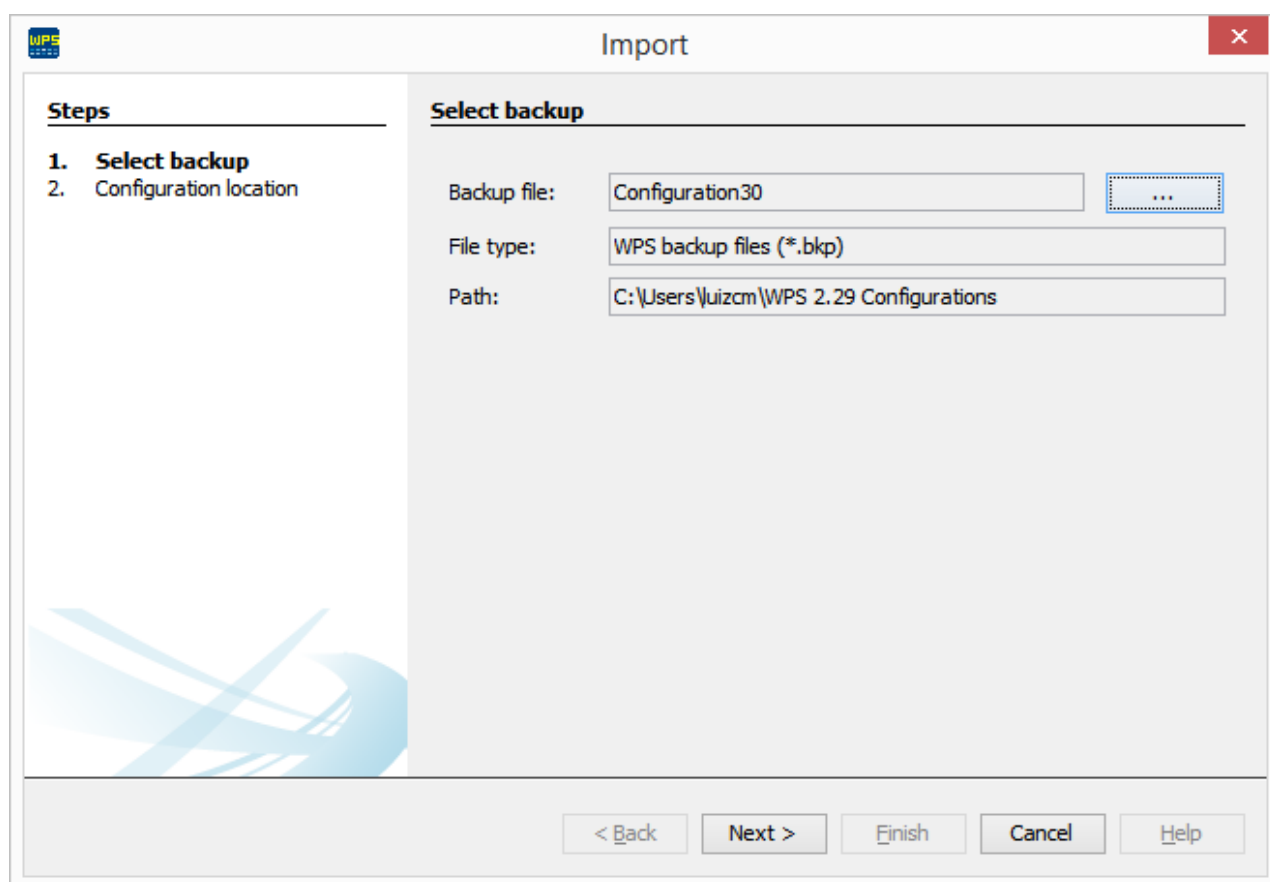
Selecione o arquivo de extensão bkp que deseja importar e clique Abrir.



4) Arquivo BKP a ser Importado

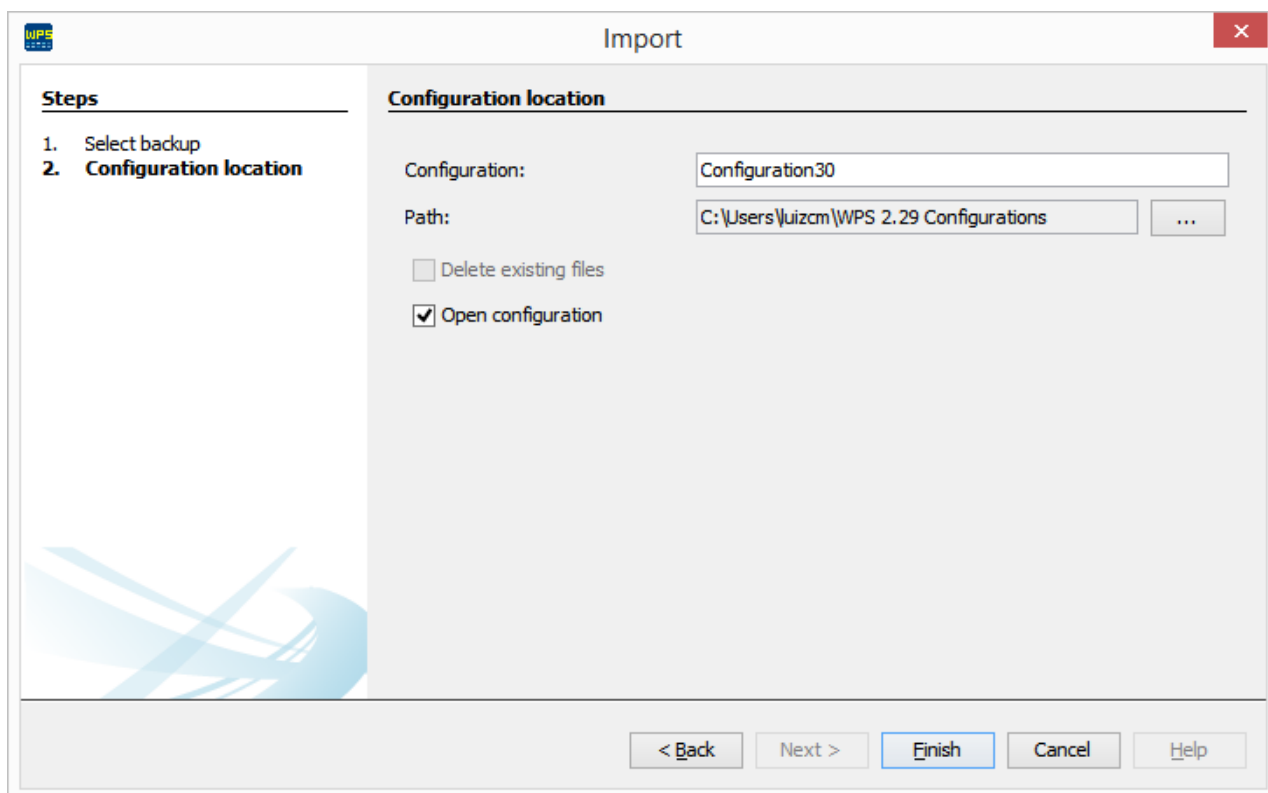
O campo Arquivo de Backup é preenchido com o arquivo selecionado no passo anterior.

Clique Próximo para continuar.



5) Localização da Configuração

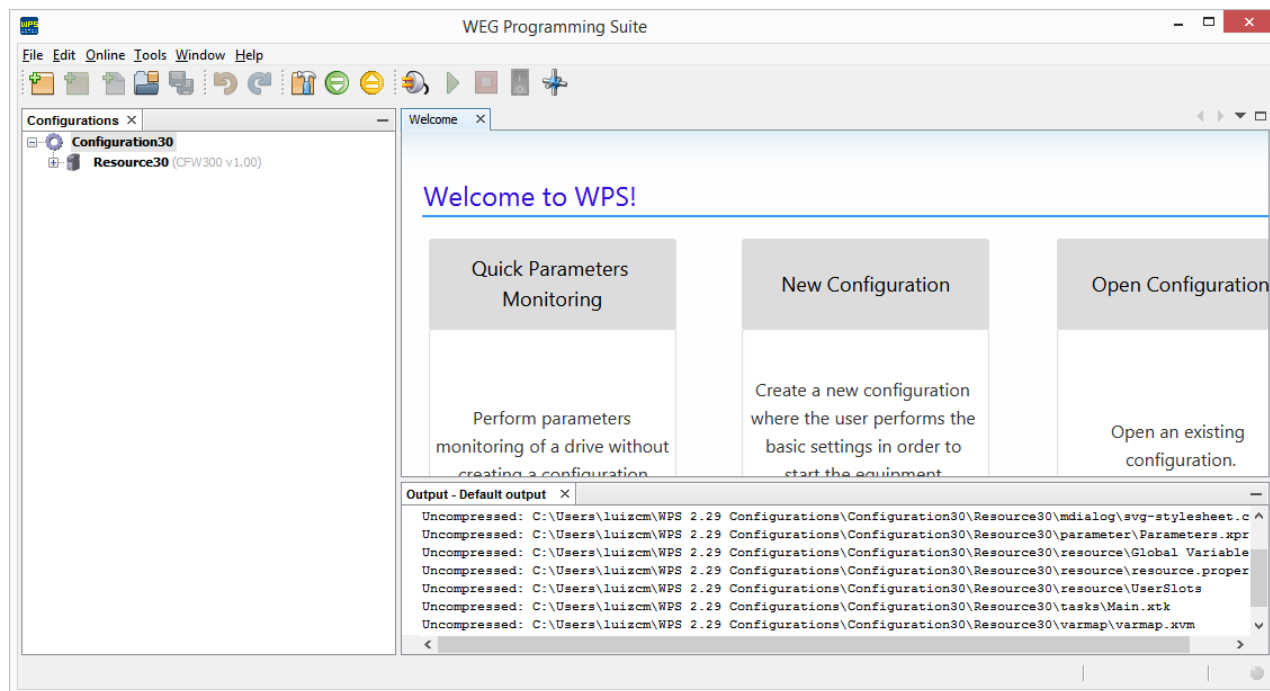
Clique Finalizar para importar e concluir o assistente.



6) Configuração Importada

A importação pode ser visualizada na janela de saída.

Na janela de configurações pode-se visualizar a configuração importada e seus recursos.



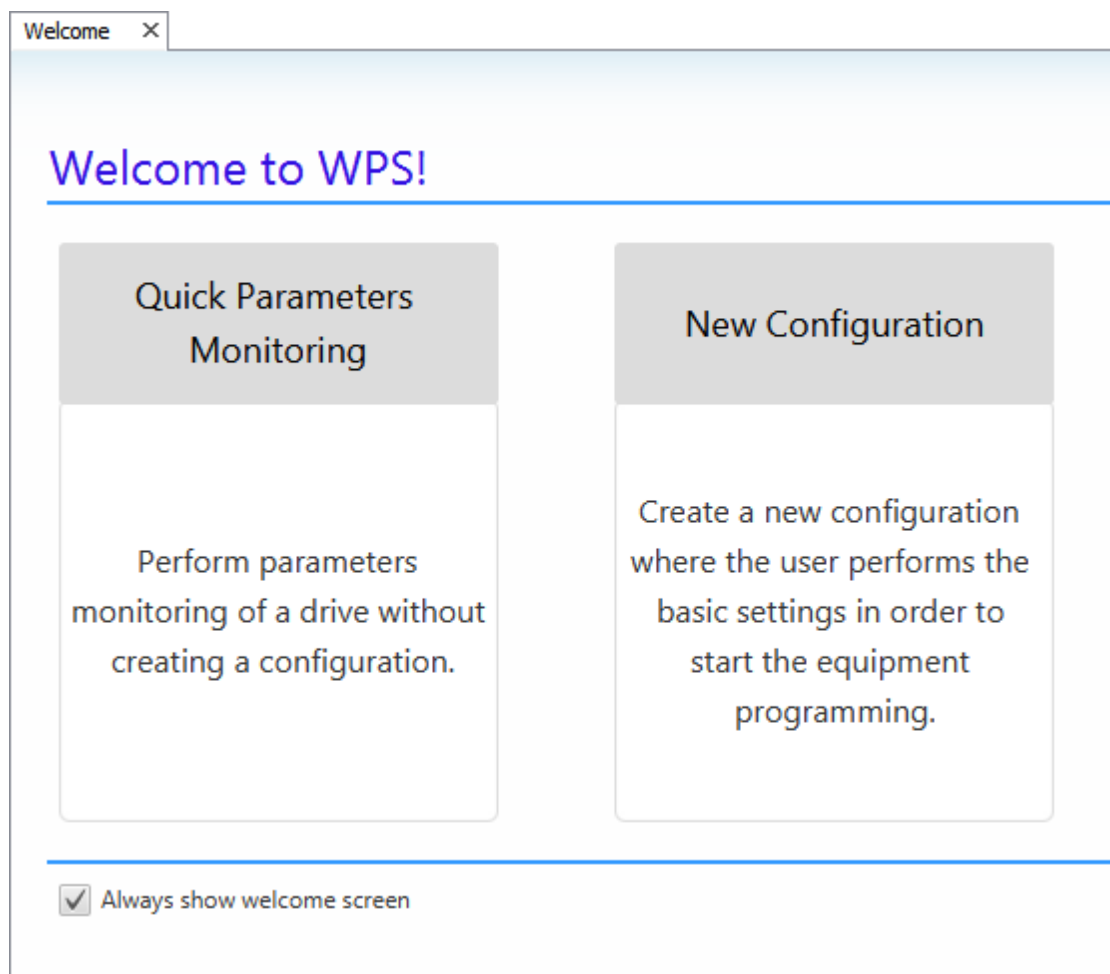
Opção Sempre Mostrar Janela de Boas Vindas

A opção **Sempre mostrar janela de boas vindas** permite que a janela seja sempre apresentada na inicialização do WPS.

Esta opção está localizada na parte inferior da janela Boas Vindas.

Desmarque esta opção para não visualizar a janela de de boas vindas na próxima inicialização do programa.

Você pode habilitar esta opção posteriormente em **Ajuda > Bem Vindo**.



6.2 Criando Nova Configuração

Existem 2 situações para criar uma nova configuração:

- Equipamento está online (conectado ao WPS)
- Equipamento está offline.

Exemplos

Equipamento está online:

- [Nova Configuração - Equipamento Online](#)

Equipamento Offline:

- [Nova Configuração - Equipamento Offline](#)

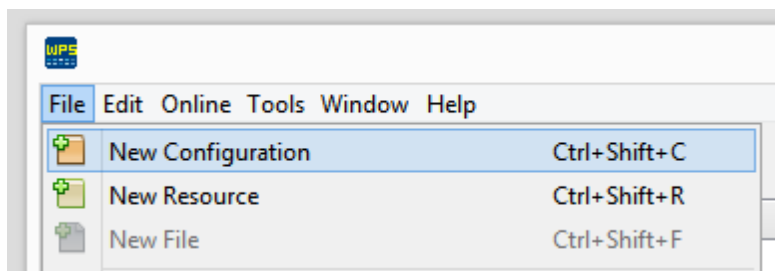
6.3 Nova Configuração - Equipamento Online

No exemplo a seguir, vamos criar uma configuração com CFW300. As janelas poderão ser diferentes se forem configurados outros equipamentos.

CFW300

1) Botão/Item de Menu Nova Configuração

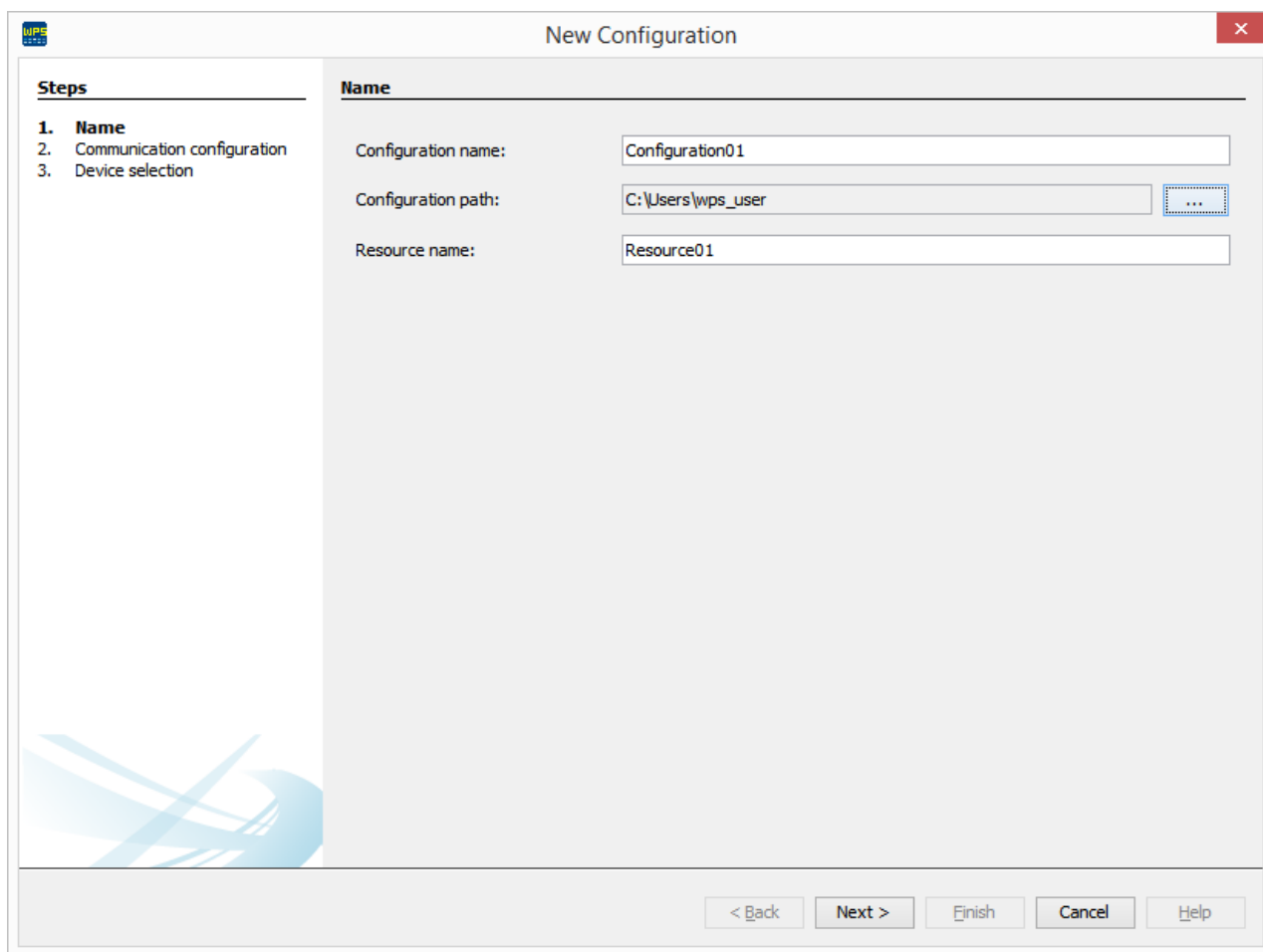
No menu Arquivo, clique em Nova Configuração.



Você também pode utilizar o atalho do teclado (Ctrl+Shift+C) ou o botão Nova Configuração da Barra de Ferramentas: 

2) Janela Nova Configuração

Aparecerá uma janela para inserção do nome da configuração e do primeiro recurso a ser criado. Insira os nomes nos respectivos campos e clique em Próximo.



New Configuration

Steps

1. **Name**
2. Communication configuration
3. Device selection

Name

Configuration name: Configuration01

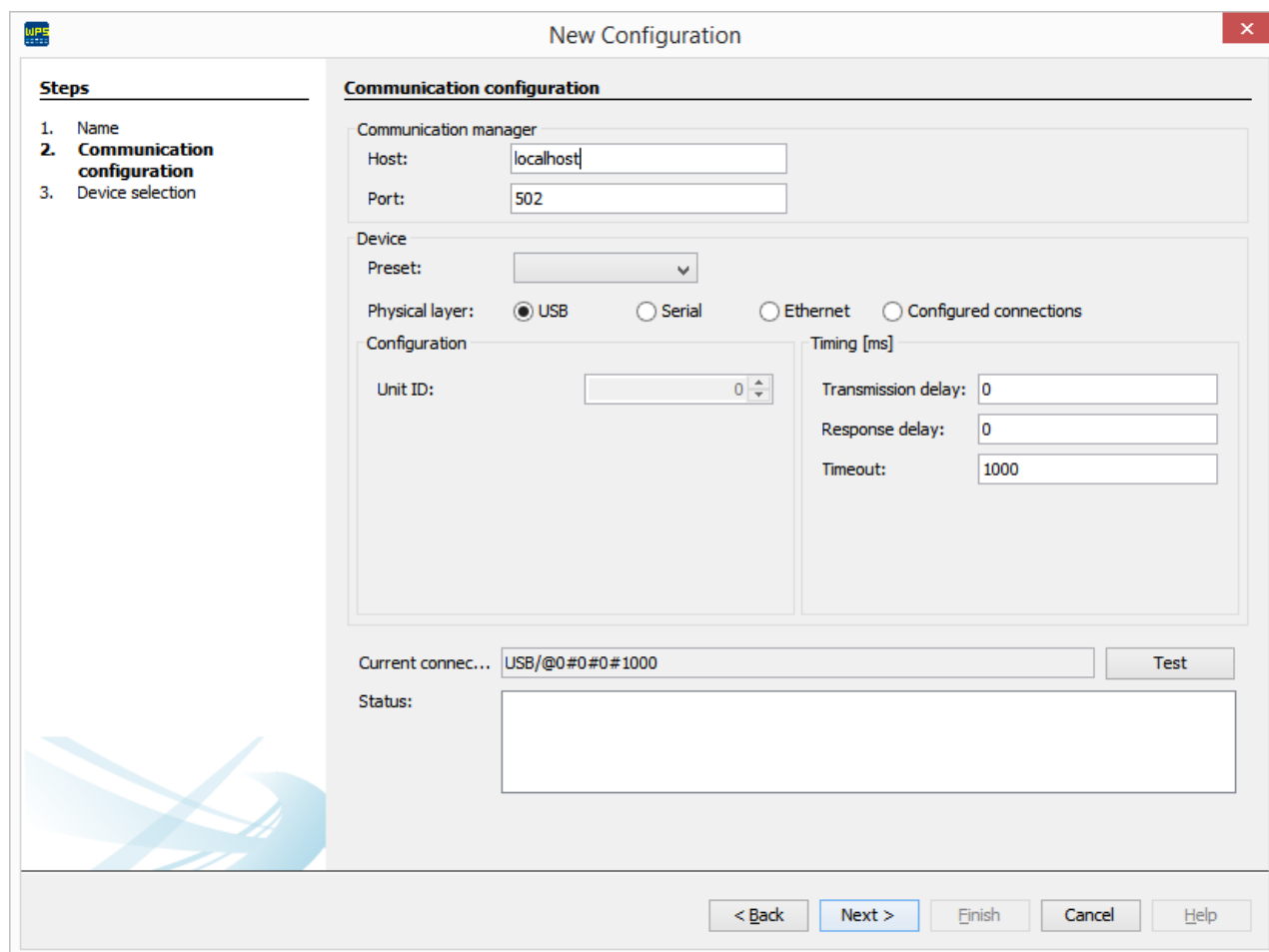
Configuration path: C:\Users\wps_user ...

Resource name: Resource01

< Back Next > Finish Cancel Help

3) Configuração da Comunicação

A próxima janela define as opções de comunicação com o equipamento.

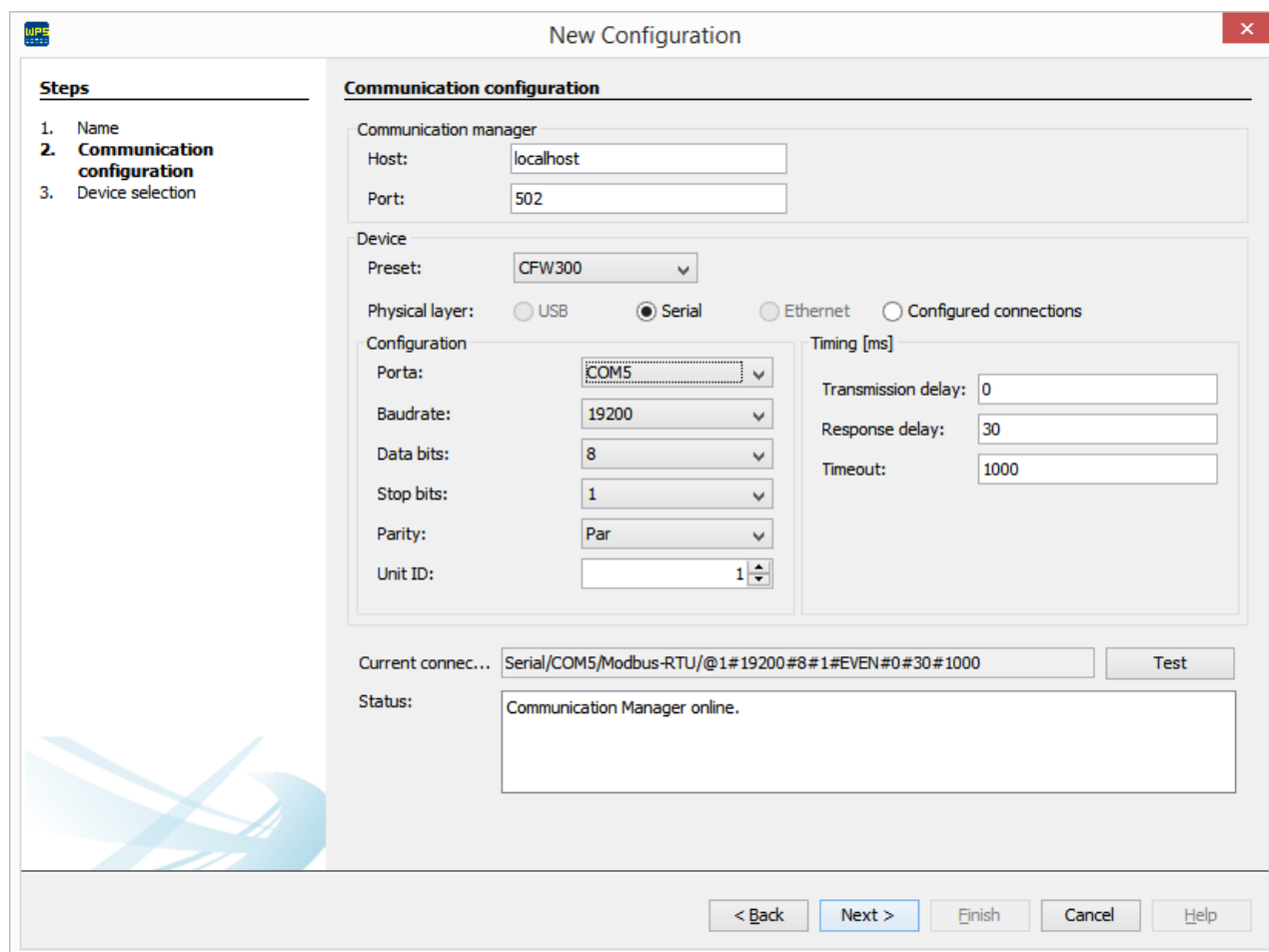


4) Seleção das Opções de Comunicação

Escolha as opções corretas de comunicação.

Se você selecionar um equipamento, a configuração padrão de comunicação do equipamento será carregada na janela.

No entanto verifique se a configuração da janela é igual ao do equipamento. Se não for, atualize esta janela de acordo com a configuração do equipamento.



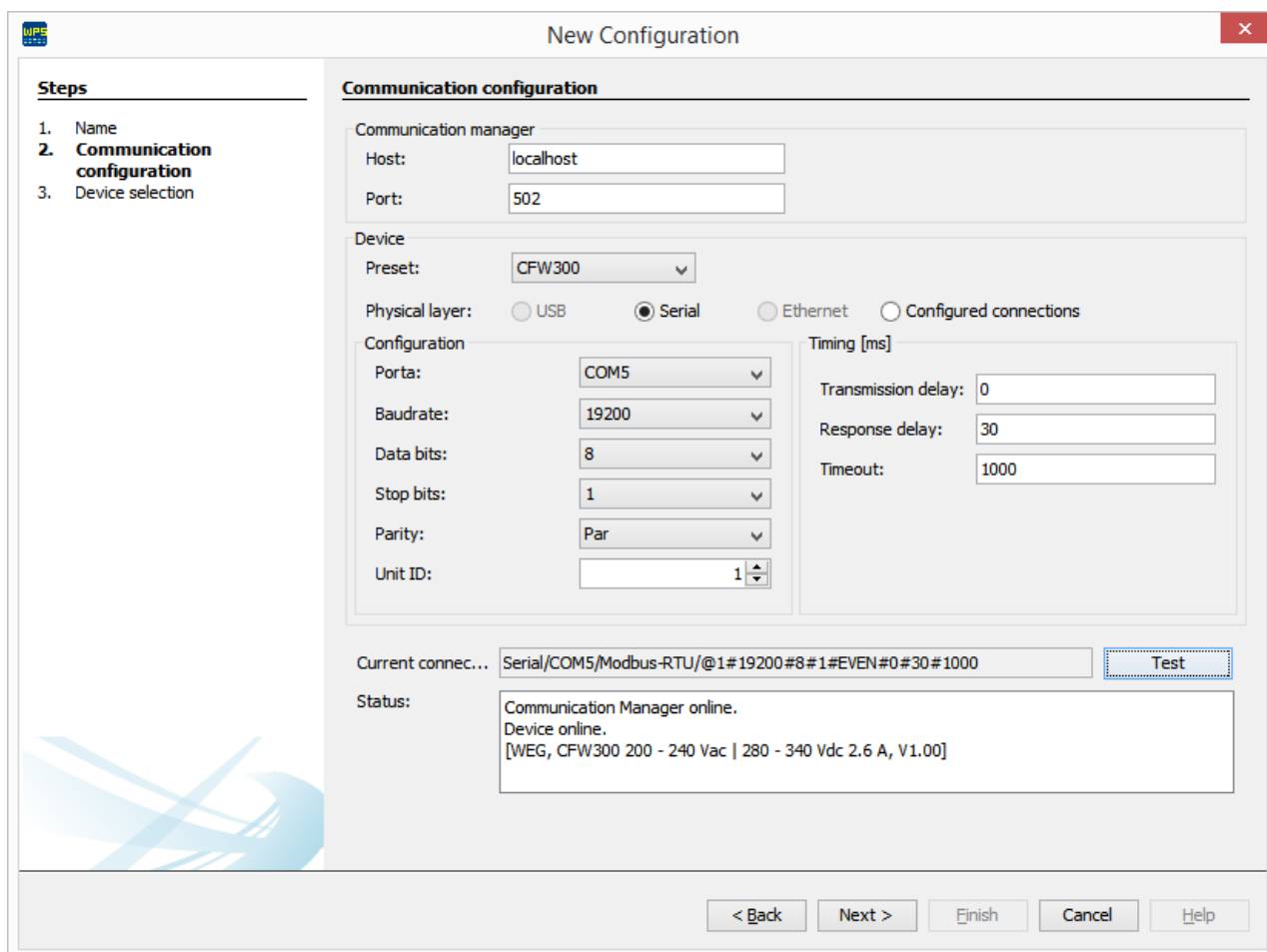

NOTA!

- Neste exemplo, o acessório CFW300-CUSB está sendo utilizado, que no Windows aparece como uma porta serial USB (porta COM virtual);
- Quando o cabo é conectado no microcomputador/equipamento, o gerenciador de dispositivos do Windows informa o nome da porta serial USB criada.

5) Testando a Comunicação

Você pode verificar se a comunicação está correta clicando no botão Testar desta janela.

O campo estado é atualizado. O equipamento conectado aparece no campo estado.



New Configuration

Steps

1. Name
2. **Communication configuration**
3. Device selection

Communication configuration

Communication manager

Host: localhost

Port: 502

Device

Preset: CFW300

Physical layer: ☐ USB ☒ Serial ☐ Ethernet ☐ Configured connections

Configuration

Porta: COM5

Baudrate: 19200

Data bits: 8

Stop bits: 1

Parity: Par

Unit ID: 1

Timing [ms]

Transmission delay: 0

Response delay: 30

Timeout: 1000

Current connec... Serial/COM5/Modbus-RTU/@1#19200#8#1#EVEN#0#30#1000 **Test**

Status:

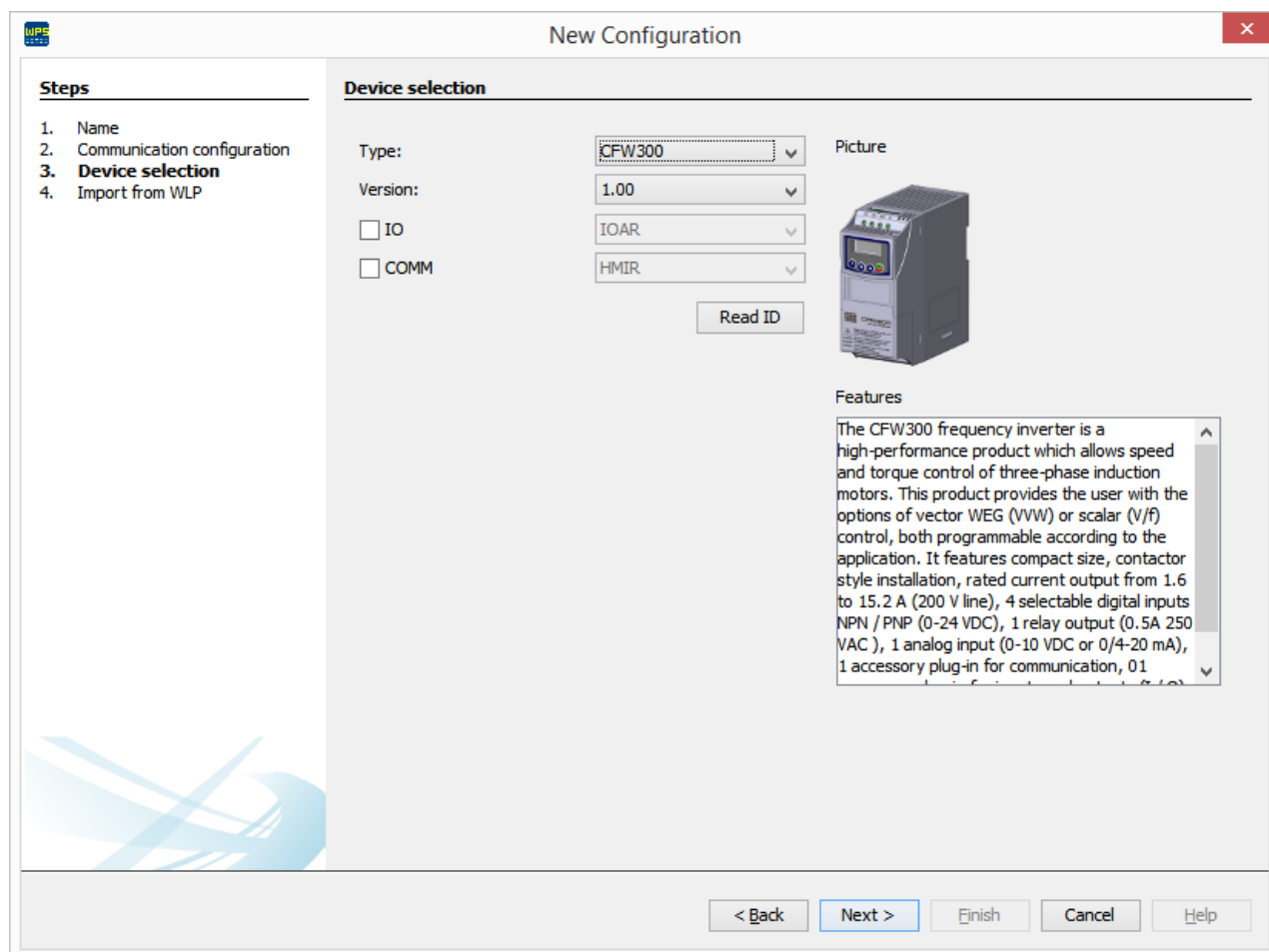
Communication Manager online.
Device online.
[WEG, CFW300 200 - 240 Vac | 280 - 340 Vdc 2.6 A, V1.00]

< Back Next > Finish Cancel Help

Em seguida clique Próximo.

6) Seleção do Equipamento

A janela apresenta o equipamento que está conectado ao recurso criado.



7) Ler ID

Se você clicar no botão Ler ID, uma janela abrirá com as informações do equipamento. Clique OK para fechar.

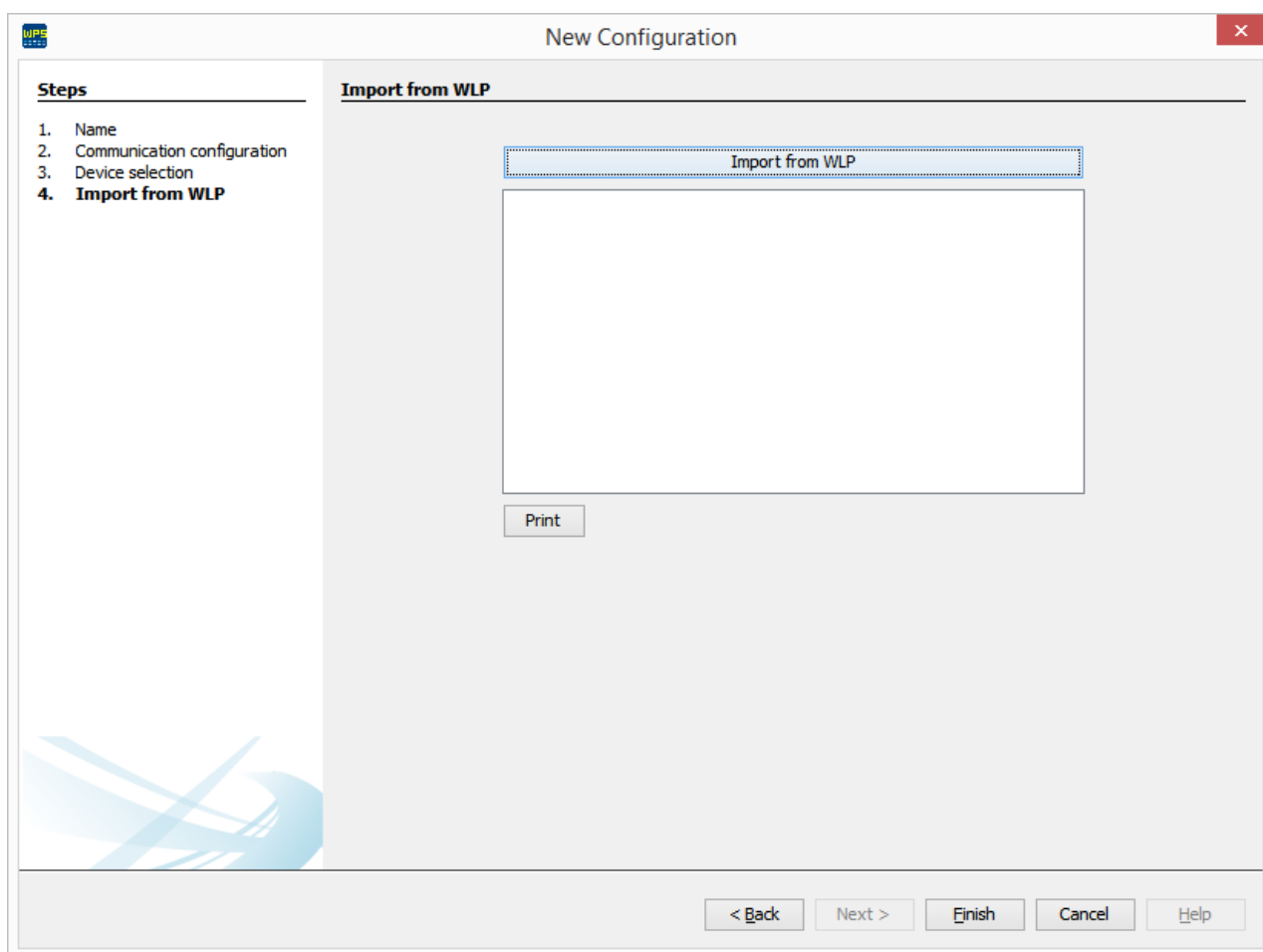


Para ir para a próxima etapa, clique Próximo.

8) Importar do WLP

Uma nova janela aparece onde é possível importar um projeto Ladder do CFW100 desenvolvido no WLP (WEG

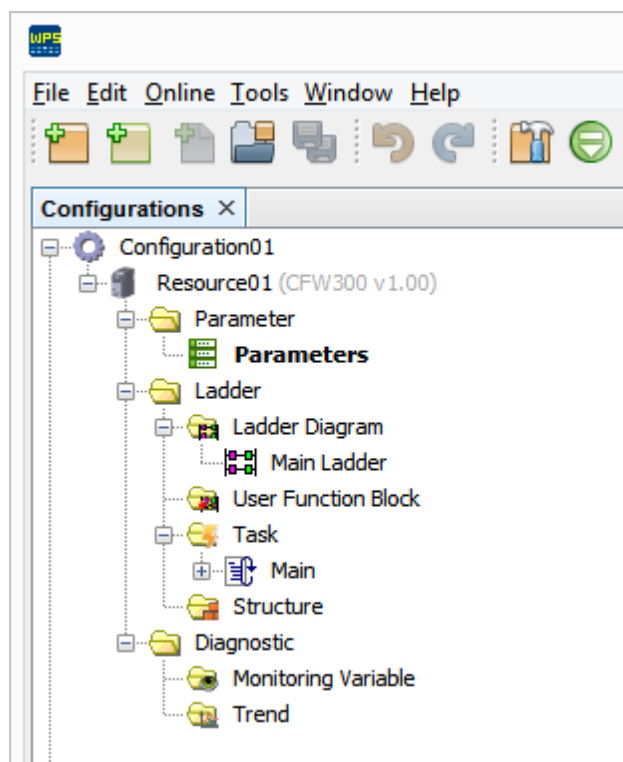
Ladder Programmer).



Clique Finalizar para concluir o assistente.

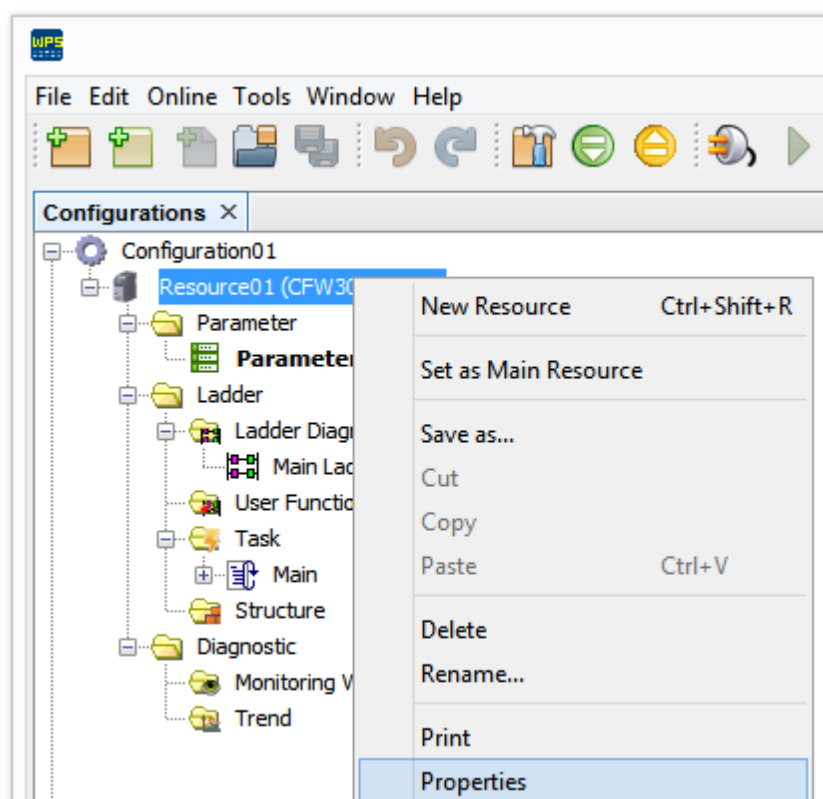
9) Árvore de Configurações

Ao fim destes passos, a árvore com as configurações deve estar parecida com a da imagem a seguir.



10) Propriedades do Recurso

As configurações realizadas no setup podem ser alteradas posteriormente em propriedades do recurso.



NOTA!

- Dependendo do equipamento, novas janelas podem aparecer deste passo em diante, com configurações iniciais e start-ups orientados.

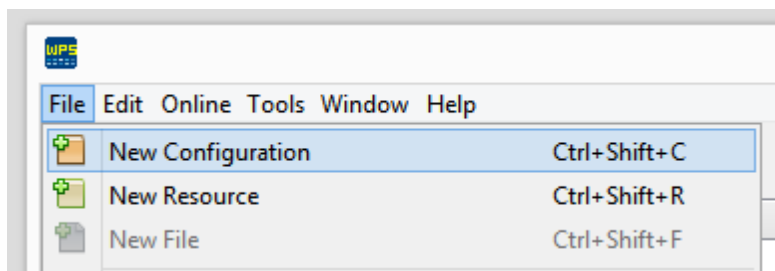
6.4 Nova Configuração - Equipamento Offline

No exemplo a seguir, vamos criar uma configuração com CFW300. As janelas poderão ser diferentes se forem configurados outros equipamentos.

CFW300

1) Botão/Item de Menu Nova Configuração

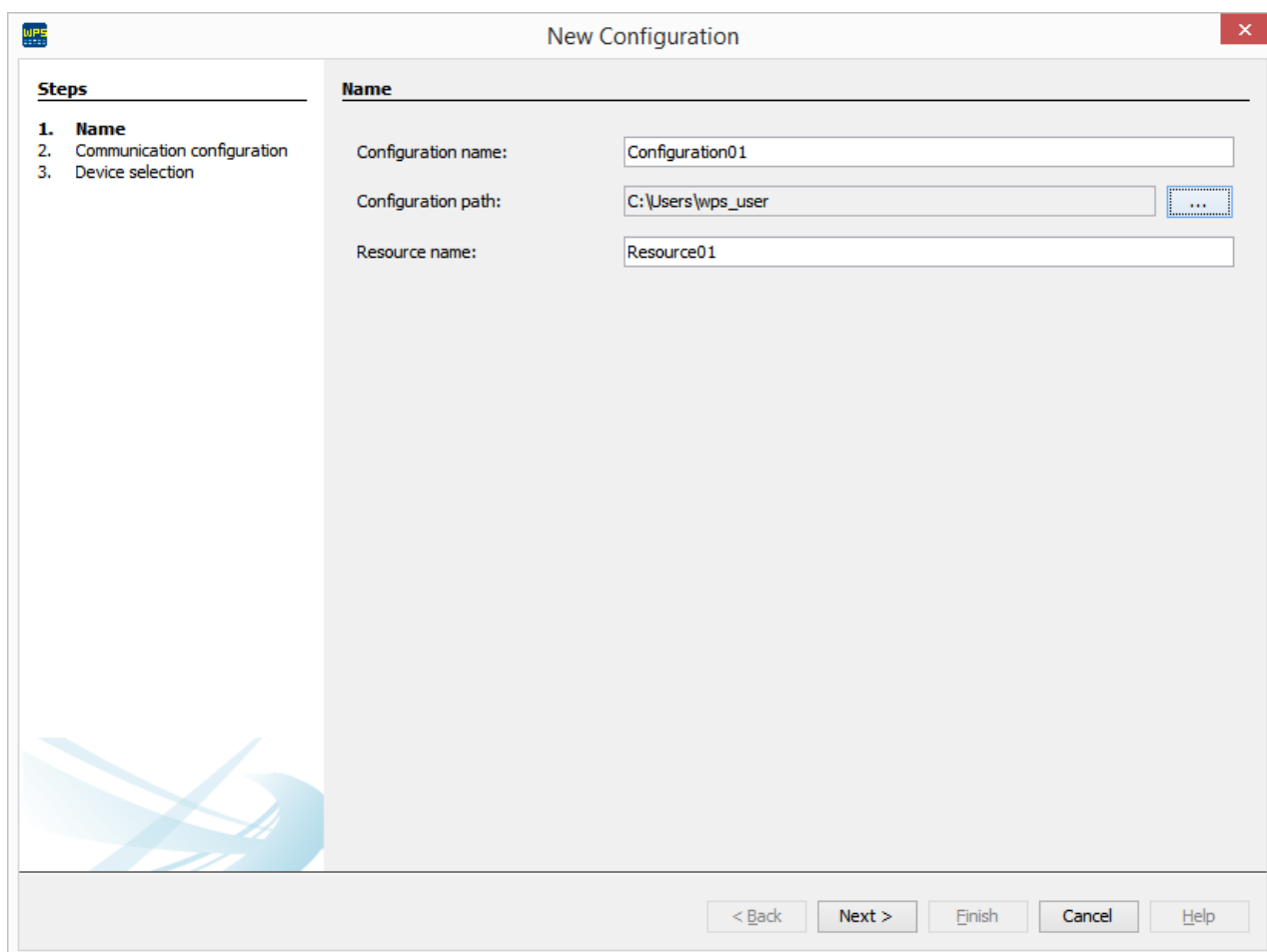
No menu Arquivo, clique em Nova Configuração.



Você também pode utilizar o atalho do teclado (Ctrl+Shift+C) ou o botão Nova Configuração da Barra de Ferramentas: 

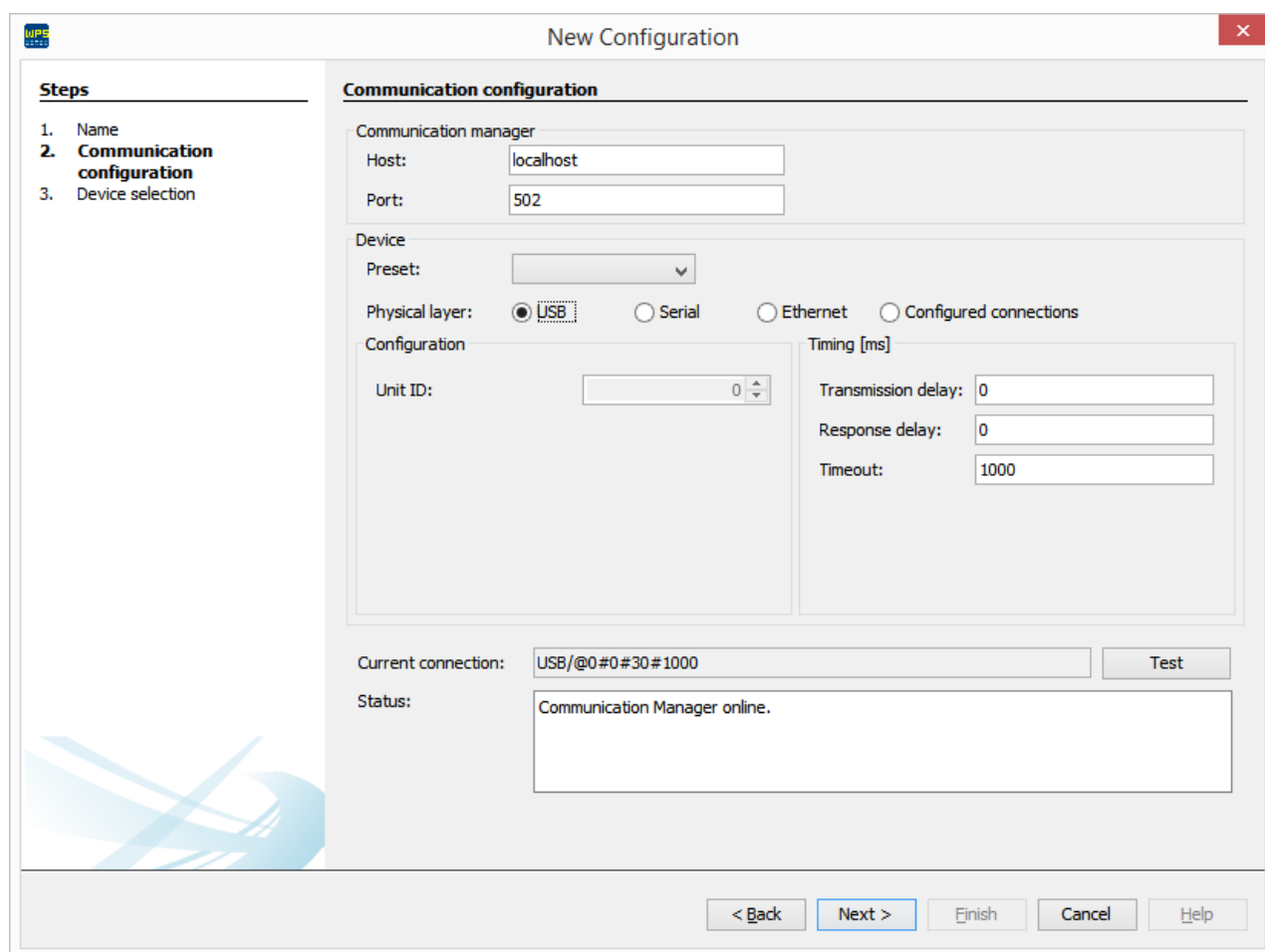
2) Janela Nova Configuração

Aparecerá uma janela para inserção do nome da configuração e do primeiro recurso a ser criado. Insira os nomes nos respectivos campos e clique em Próximo.



3) Configuração da Comunicação

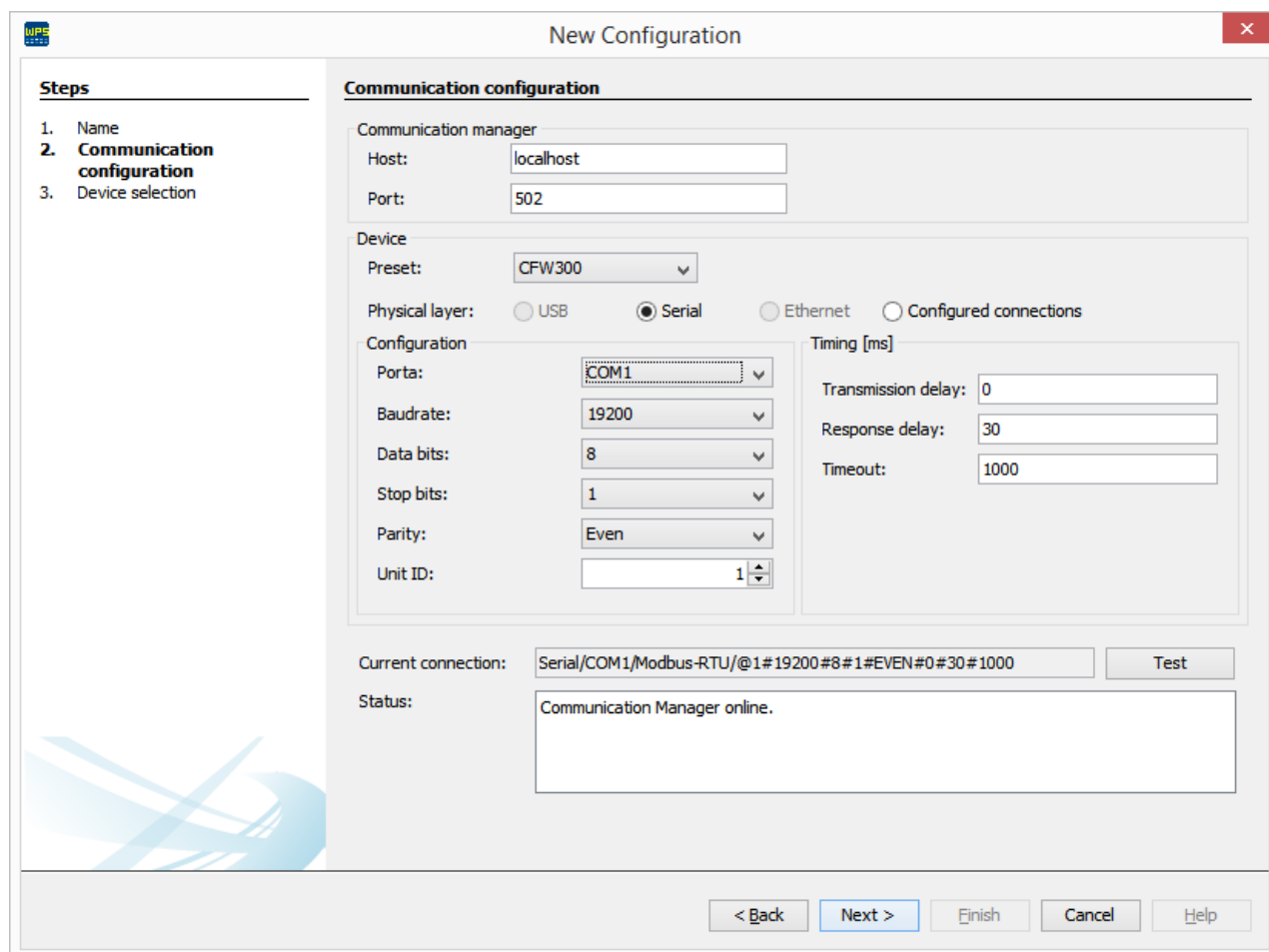
A próxima janela define as opções de comunicação com o equipamento.



4) Seleção das Opções de Comunicação

Escolha as opções corretas de comunicação.

Se você selecionar um equipamento, a configuração padrão de comunicação do equipamento será carregada.



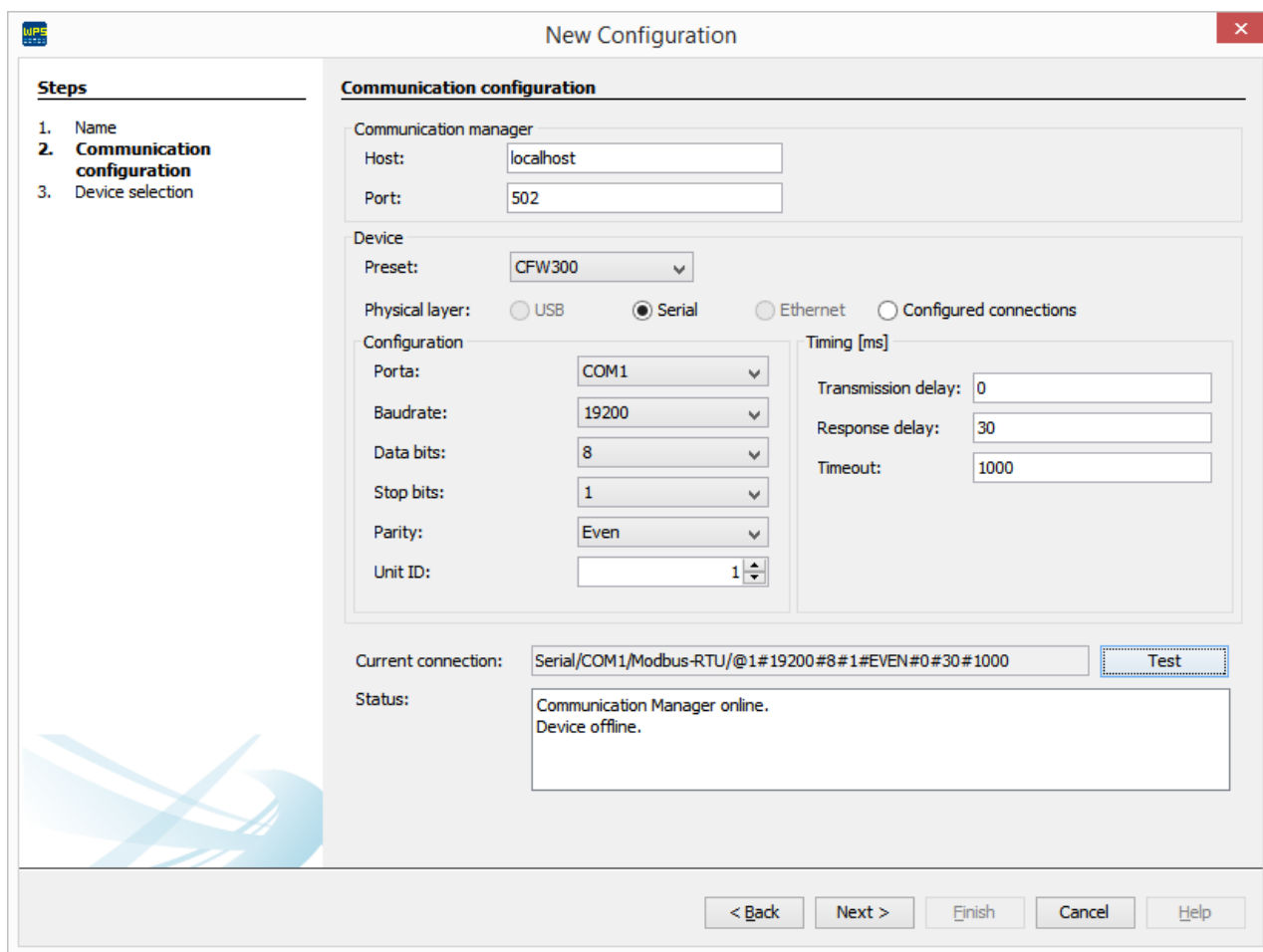
Dependendo da camada física utilizada, será necessário rever esta configuração quando o equipamento for conectado ao WPS, em momento futuro.

Por exemplo, se for utilizado o acessório CFW300-CUSB no CFW300, o Windows cria uma porta serial USB que é conhecida somente no momento da conexão do equipamento e microcomputador.

5) Testando a Comunicação

Neste momento não será possível testar a comunicação (equipamento não está conectado).

Assim, clicando em Testar, o estado aparece como Equipamento offline.



Steps

1. Name
- 2. Communication configuration**
3. Device selection

Communication configuration

Communication manager

Host: localhost

Port: 502

Device

Preset: CFW300

Physical layer: ☐ USB ☒ Serial ☐ Ethernet ☐ Configured connections

Configuration

Porta: COM1

Baudrate: 19200

Data bits: 8

Stop bits: 1

Parity: Even

Unit ID: 1

Timing [ms]

Transmission delay: 0

Response delay: 30

Timeout: 1000

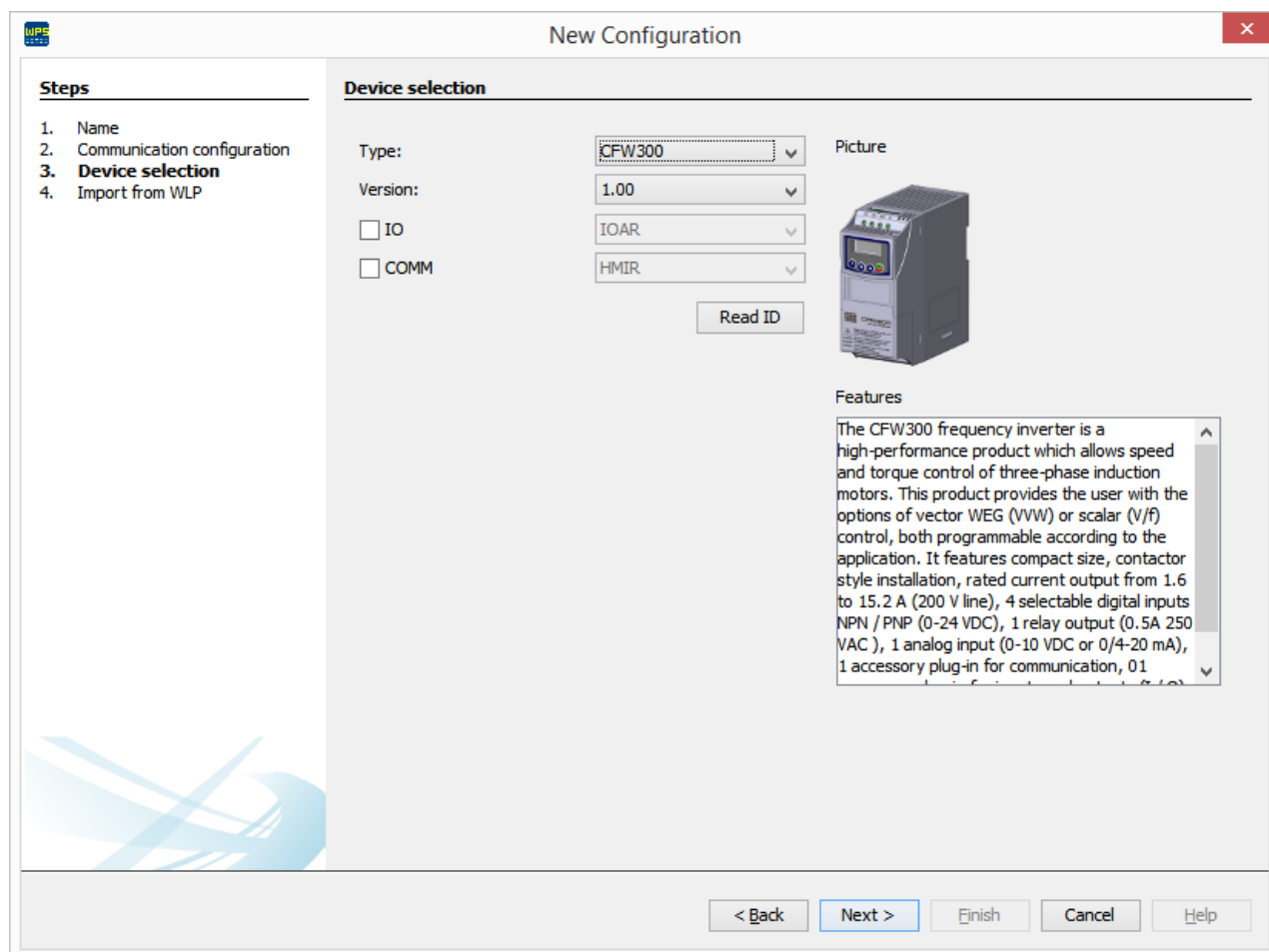
Current connection: Serial/COM1/Modbus-RTU/@1#19200#8#1#EVEN#0#30#1000 **Test**

Status: Communication Manager online.
Device offline.

< Back Next > Finish Cancel Help

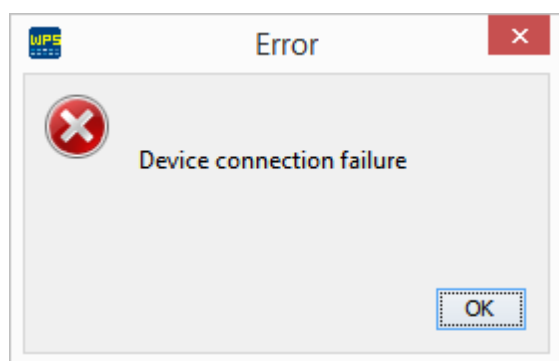
6) Seleção do Equipamento

A janela apresenta o equipamento que está conectado ao recurso criado.



7) Ler ID

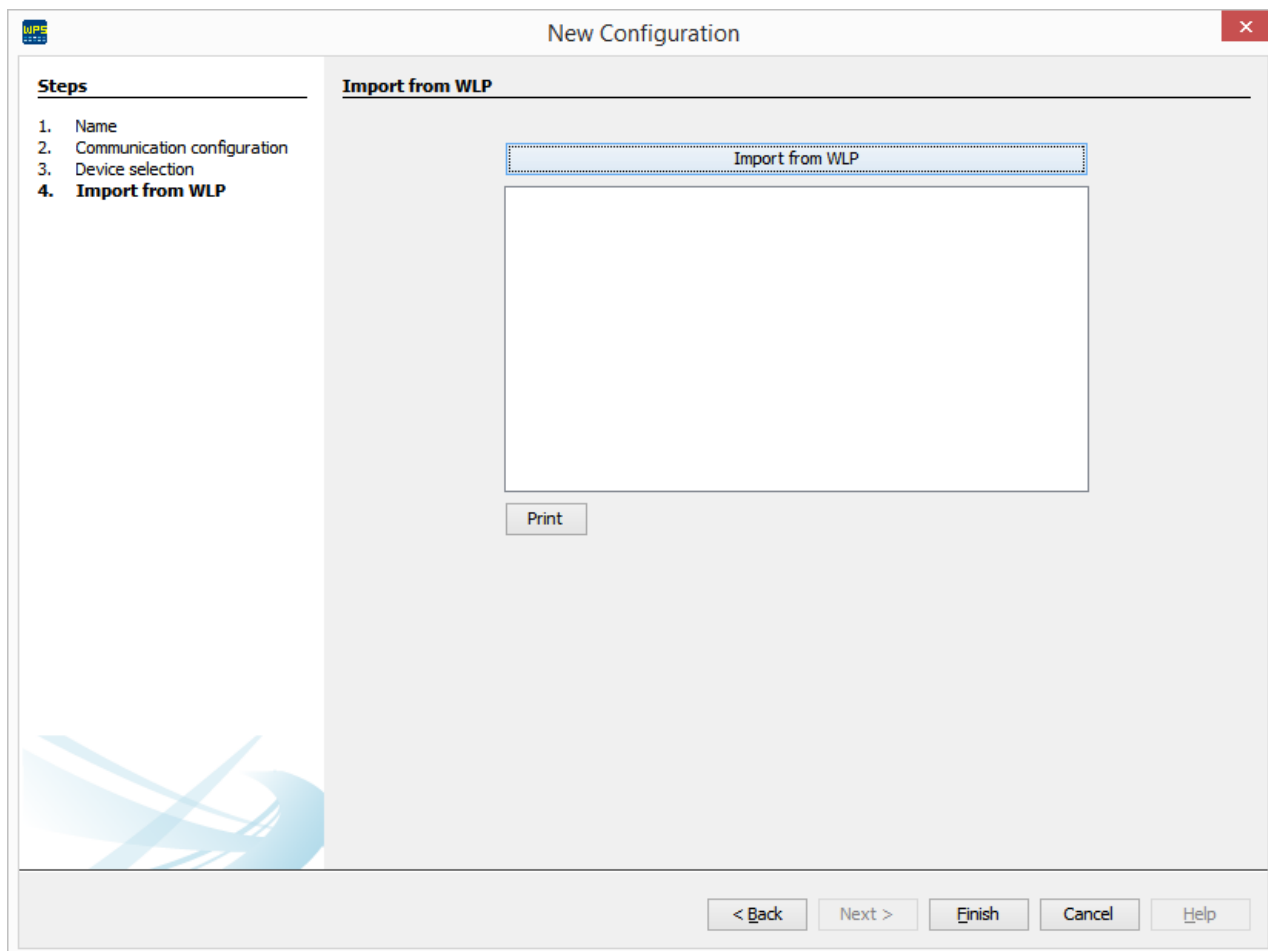
Se você clicar no botão Ler ID e como não há equipamento conectado, uma janela de falha será apresentada (falha de conexão do equipamento). Clique OK para fechar.



Para ir para a próxima etapa, clique Próximo.

8) Importar do WLP

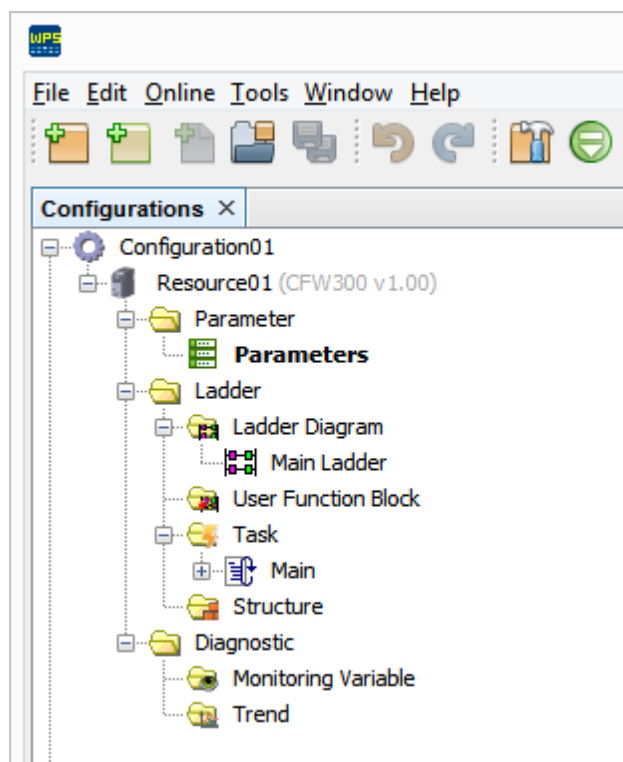
Uma nova janela aparece onde é possível importar um projeto Ladder do CFW100 desenvolvido no WLP (WEG Ladder Programmer).



Clique Finalizar para concluir o assistente.

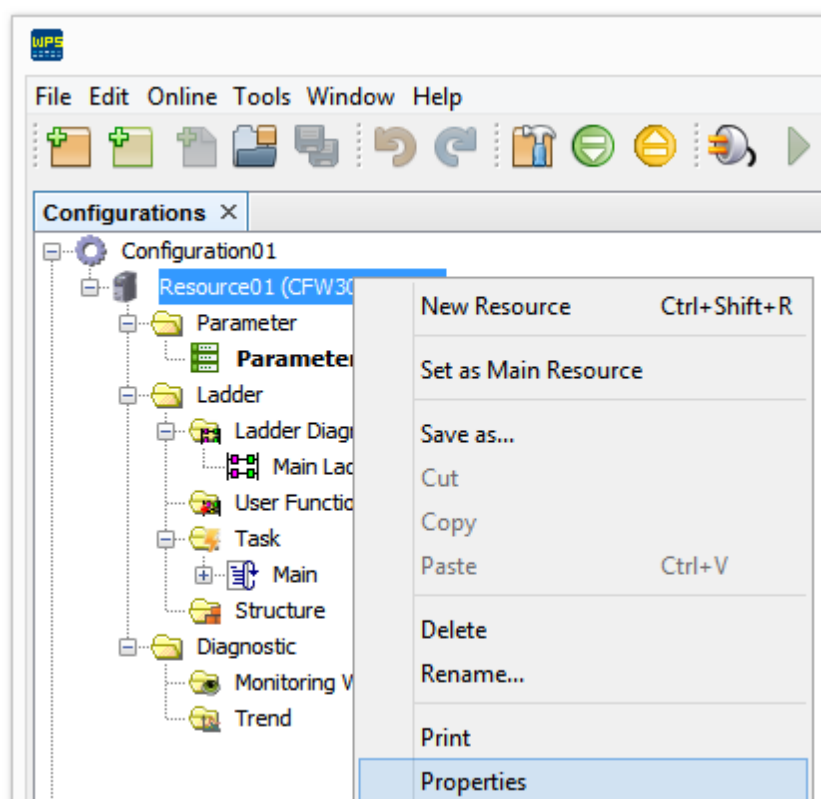
9) Árvore de Configurações

Ao fim destes passos, a árvore com as configurações deve estar parecida com a da imagem a seguir.



10) Propriedades do Recurso

As configurações realizadas no setup podem ser alteradas posteriormente em propriedades do recurso.



6.5 Criando Novo Recurso

Um novo recurso deve ser criado dentro de uma configuração.

Existem 2 situações:

- Equipamento está online (conectado ao WPS)
- Equipamento está offline.

Exemplo

Equipamento está online:

- [Novo Recurso - Equipamento Online](#)

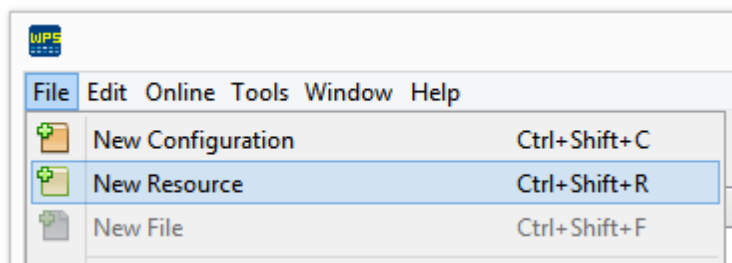
6.6 Novo Recurso - Equipo Online

No exemplo a seguir, vamos criar um novo recurso com CFW300. As janelas poderão ser diferentes se forem configurados outros equipamentos.

CFW300

1) Botão/Item de Menu Novo Recurso

No menu Arquivo, clique em Novo Recurso.

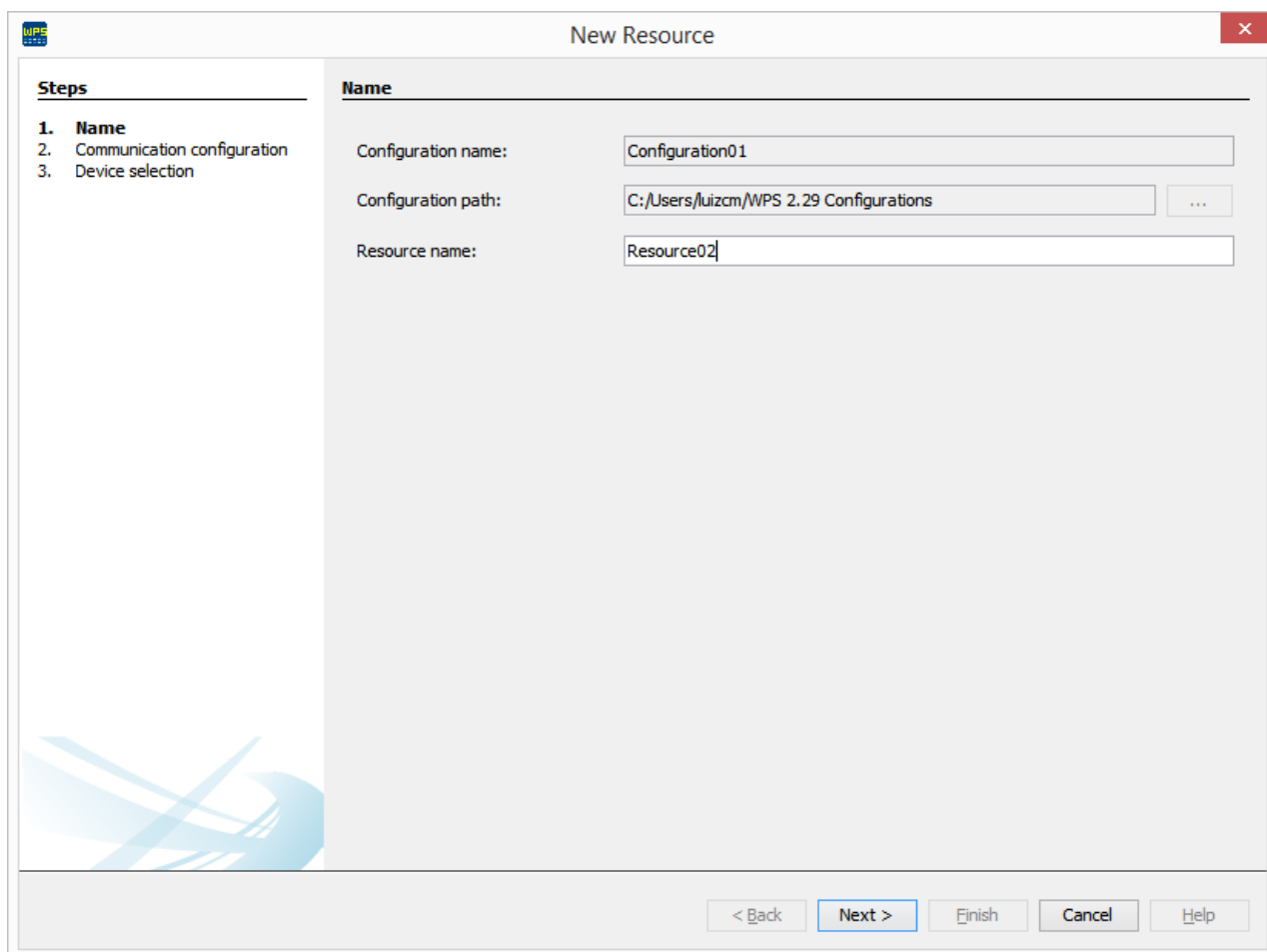


Você também pode utilizar o atalho do teclado (Ctrl+Shift+R) ou o botão Novo Recurso da Barra de

Ferramentas: 

2) Janela Nova Configuração

Aparecerá uma janela para inserção do nome da configuração e do primeiro recurso a ser criado. Insira os nomes nos respectivos campos e clique em Próximo.



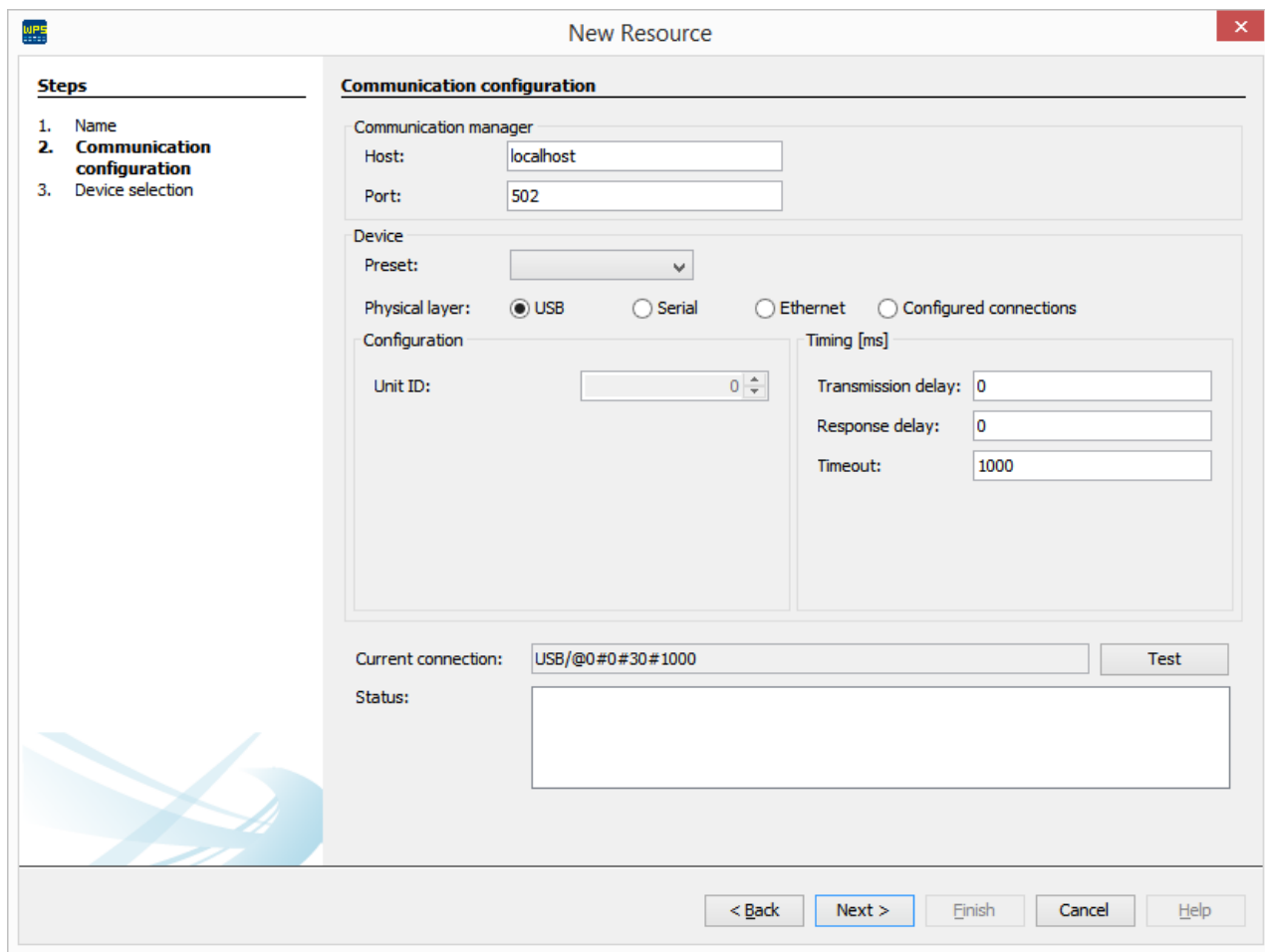
The image shows a 'New Resource' dialog box with a 'Steps' pane on the left and a 'Name' section on the right. The 'Steps' pane lists three steps: 1. Name, 2. Communication configuration, and 3. Device selection. The 'Name' section contains three input fields: 'Configuration name' with the value 'Configuration01', 'Configuration path' with the value 'C:/Users/luizcm/WPS 2.29 Configurations' and a browse button (...), and 'Resource name' with the value 'Resource02'. At the bottom, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

Steps	Name
1. Name	Configuration name: Configuration01
2. Communication configuration	Configuration path: C:/Users/luizcm/WPS 2.29 Configurations ...
3. Device selection	Resource name: Resource02

< Back Next > Finish Cancel Help

3) Configuração da Comunicação

A próxima janela define as opções de comunicação com o equipamento.



The screenshot shows the 'New Resource' dialog box with the 'Communication configuration' step selected. The 'Steps' list on the left includes: 1. Name, 2. Communication configuration (highlighted), and 3. Device selection. The main configuration area is divided into several sections:

- Communication manager:** Host: localhost, Port: 502.
- Device:** Preset: (dropdown menu).
- Physical layer:** Radio buttons for USB (selected), Serial, Ethernet, and Configured connections.
- Configuration:** Unit ID: 0 (spin box).
- Timing [ms]:** Transmission delay: 0, Response delay: 0, Timeout: 1000.
- Current connection:** USB/@0#0#30#1000, with a 'Test' button.
- Status:** (empty text box).

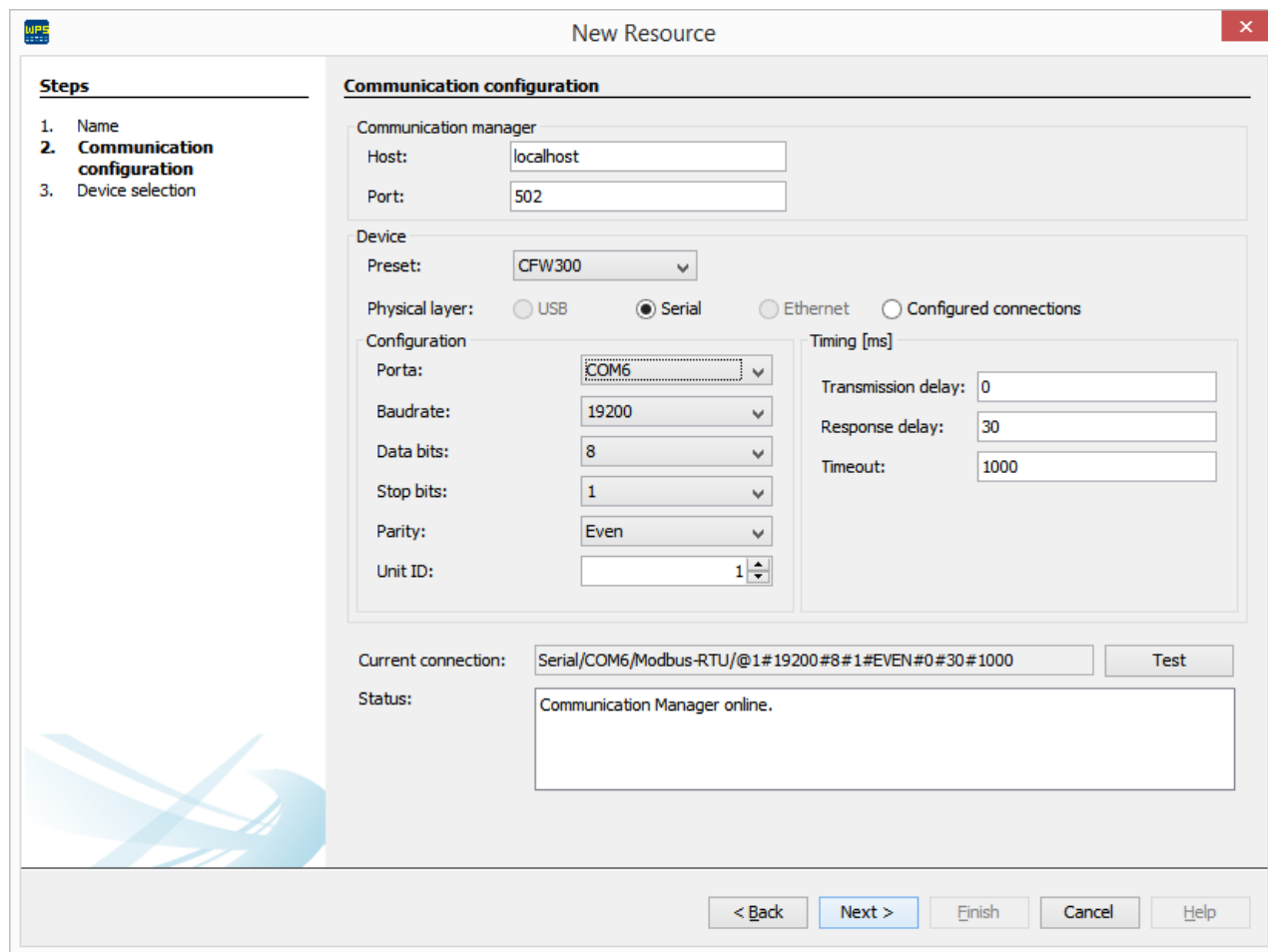
At the bottom, there are navigation buttons: '< Back', 'Next >' (highlighted), 'Finish', 'Cancel', and 'Help'.

4) Seleção das Opções de Comunicação

Escolha as opções corretas de comunicação.

Se você selecionar um equipamento, a configuração padrão de comunicação do equipamento será carregada na janela.

No entanto verifique se a configuração da janela é igual ao do equipamento. Se não for, atualize esta janela de acordo com a configuração do equipamento.



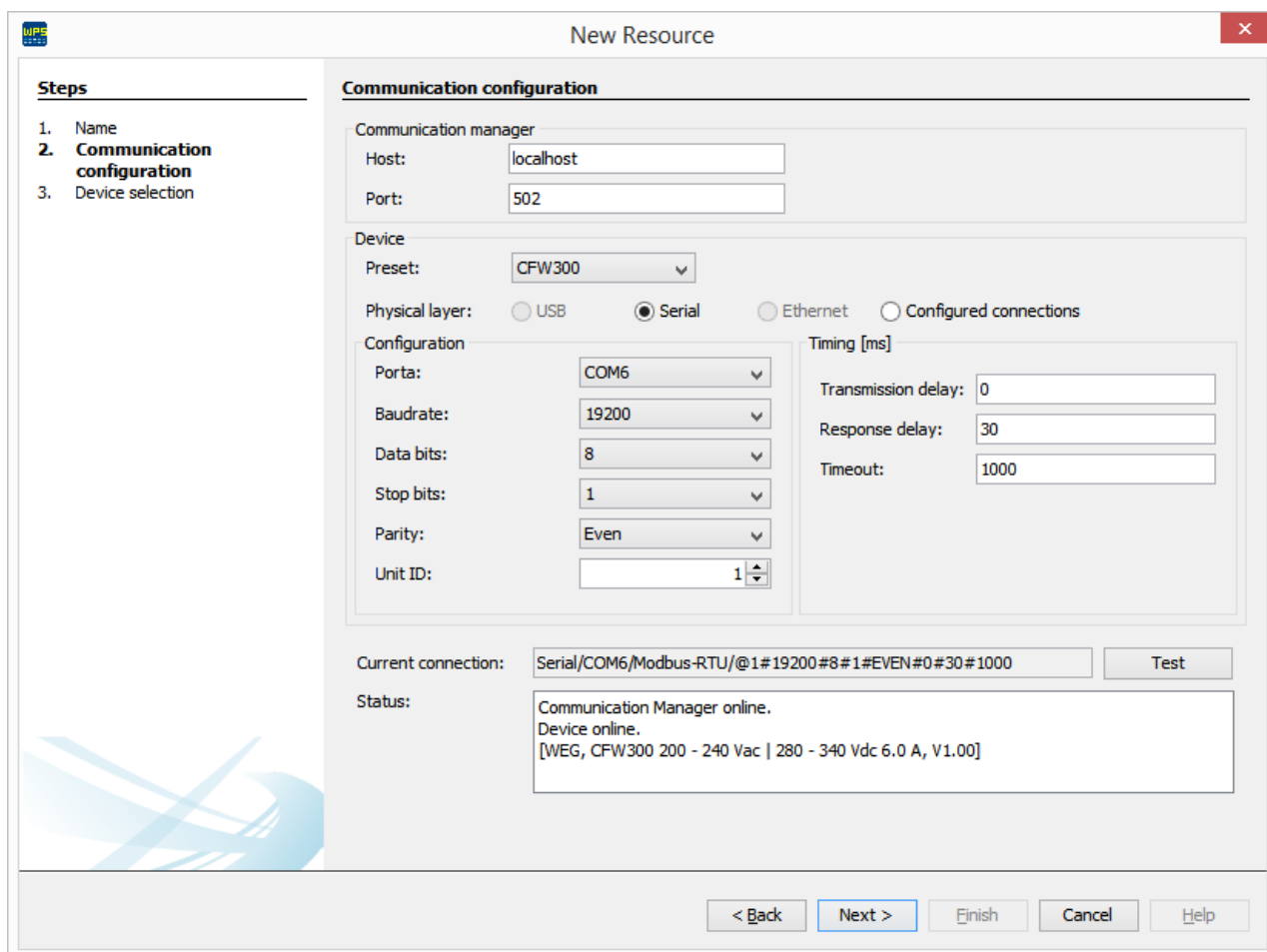

NOTA!

- Neste exemplo, o acessório CFW300-CUSB está sendo utilizado, que no Windows aparece como uma porta serial USB (porta COM virtual);
- Quando o cabo é conectado no microcomputador/equipamento, o gerenciador de dispositivos do Windows informa o nome da porta serial USB criada.

5) Testando a Comunicação

Você pode verificar se a comunicação está correta clicando no botão Testar desta janela.

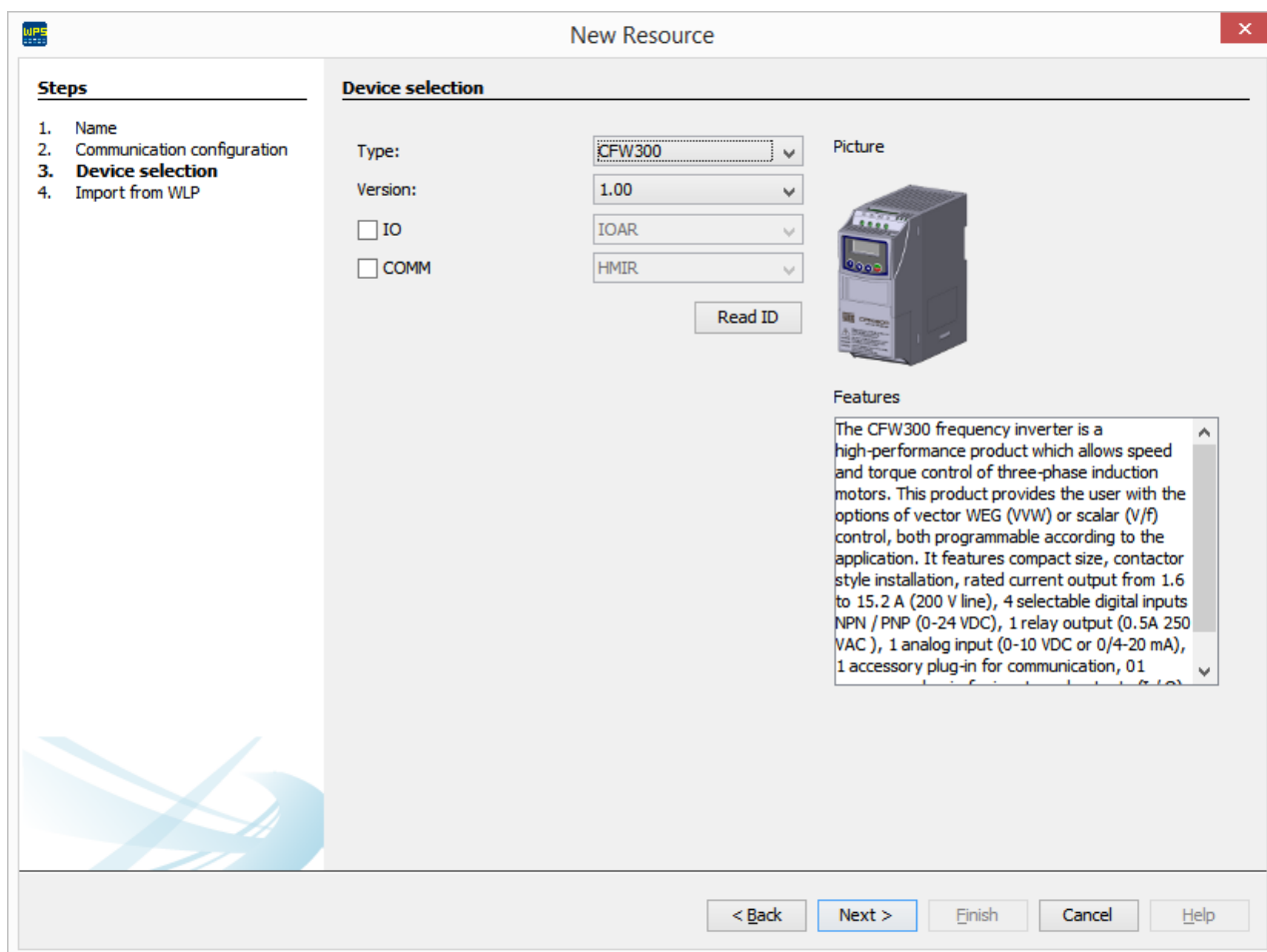
O campo estado é atualizado. O equipamento conectado aparece no campo estado.



Em seguida clique Próximo.

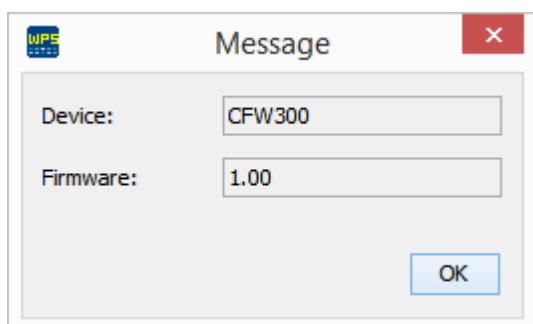
6) Seleção do Equipamento

A janela apresenta o equipamento que está conectado ao recurso criado.



7) Ler ID

Se você clicar no botão Ler ID, uma janela abrirá com as informações do equipamento. Clique OK para fechar.

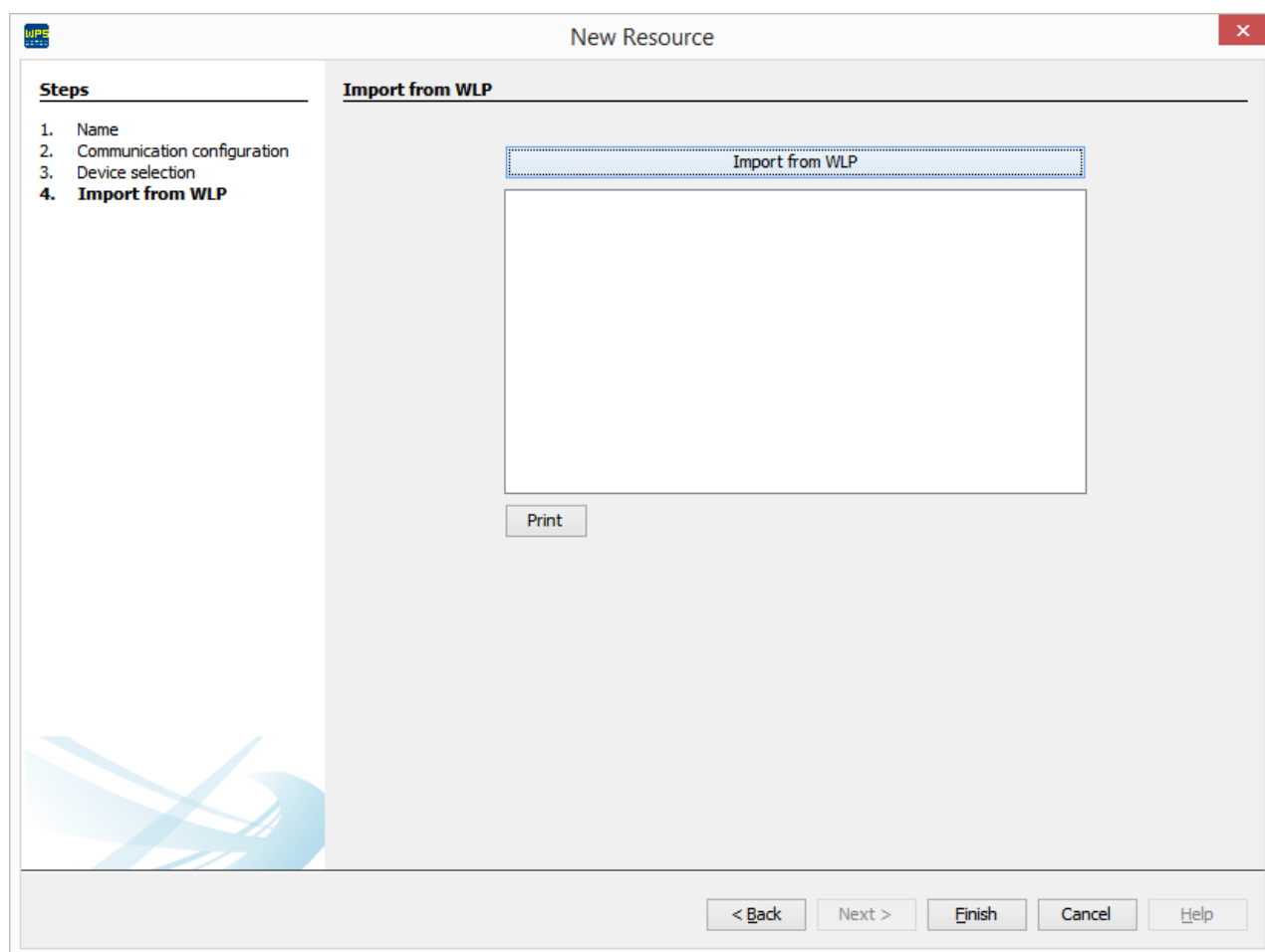


Para ir para a próxima etapa, clique Próximo.

8) Importar do WLP

Uma nova janela aparece onde é possível importar um projeto Ladder do CFW100 desenvolvido no WLP (WEG

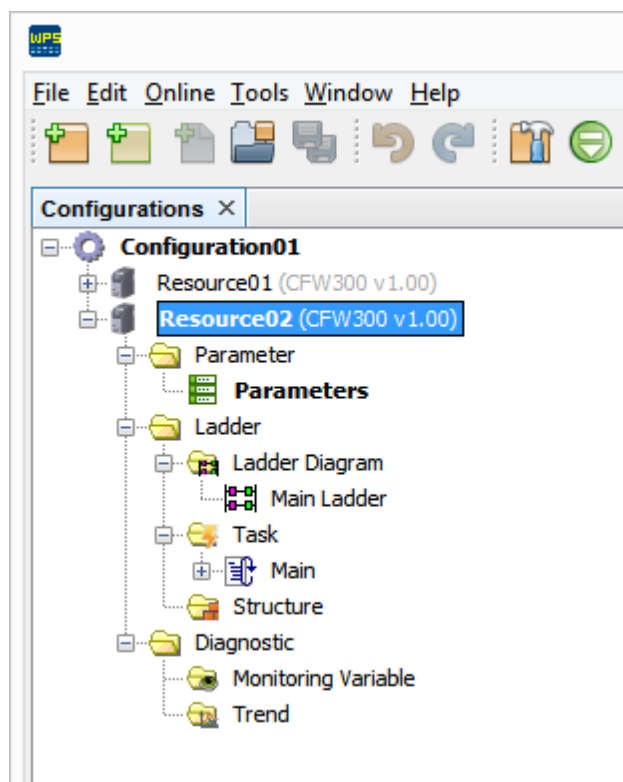
Ladder Programmer).



Clique Finalizar para concluir o assistente.

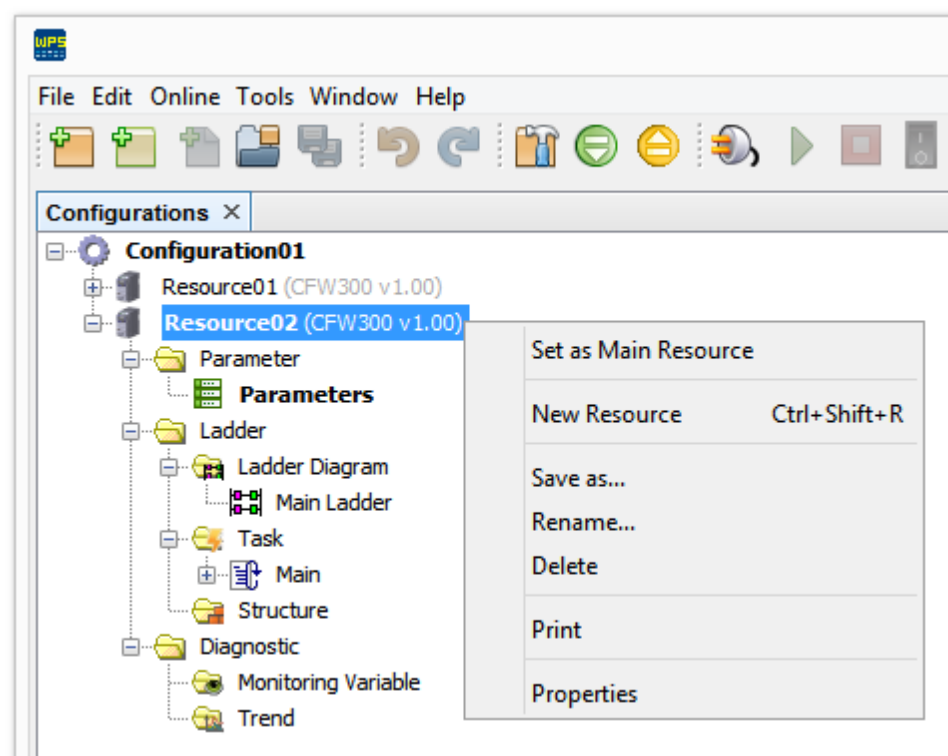
9) Árvore de Configurações

Ao fim destes passos, a árvore com as configurações deve estar parecida com a da imagem a seguir.



10) Propriedades do Recurso

As configurações realizadas no setup podem ser alteradas posteriormente em propriedades do recurso.



NOTA!

- Dependendo do equipamento, novas janelas podem aparecer deste passo em diante, com configurações iniciais e start-ups orientados.

6.7 Menu Popup

O menu popup também é conhecido como menu de contexto.

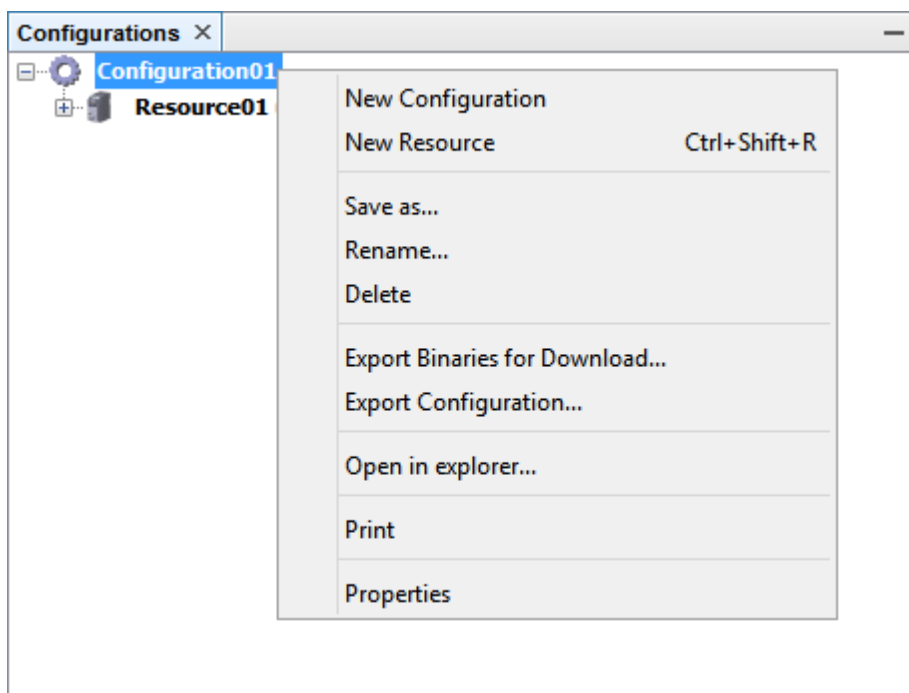
Um menu popup é um menu flutuante acessado com o segundo ou terceiro botão do mouse que permite acessar rapidamente as opções relacionadas ao objeto selecionado pelo cursor.

No WPS, um menu popup adiciona opções para tornarem algumas operações mais simples:

- [Menu Popup - Configuração](#)
- [Menu Popup - Recurso](#)

6.8 Menu Popup - Configuração

O menu popup da configuração disponibiliza algumas funções descritas a seguir.



Clique no link para acesso rápido:

[Nova Configuração](#)

[Novo Recurso](#)

[Salvar Como...](#)

[Renomear...](#)

[Excluir](#)

[Exportar Binários para Download...](#)

[Exportar Configuração](#)

[Abrir no explorer](#)

[Imprimir](#)

[Propriedades](#)

Nova Configuração

Consulte a seção [Criando Nova Configuração](#) para criar uma configuração.

Novo Recurso

Consulte a seção [Criando Novo Recurso](#) para criar um novo recurso.

Salvar Como...

Para duplicar uma configuração, clique no item de menu Salvar Como do menu popup.

No campo Nome digite o novo nome da configuração.

- Clique Ok para aceitar ou
- Clique Cancelar para cancelar a operação.

Renomear...

Para renomear uma configuração, clique no item de menu Renomear do menu popup.

Preencha o campo Novo Nome.

- Clique Ok para aceitar ou
- Clique Cancelar para cancelar a operação.

Excluir

Para excluir uma configuração, clique no item de menu Excluir do menu popup.

Uma mensagem solicita confirmação para apagar a configuração.

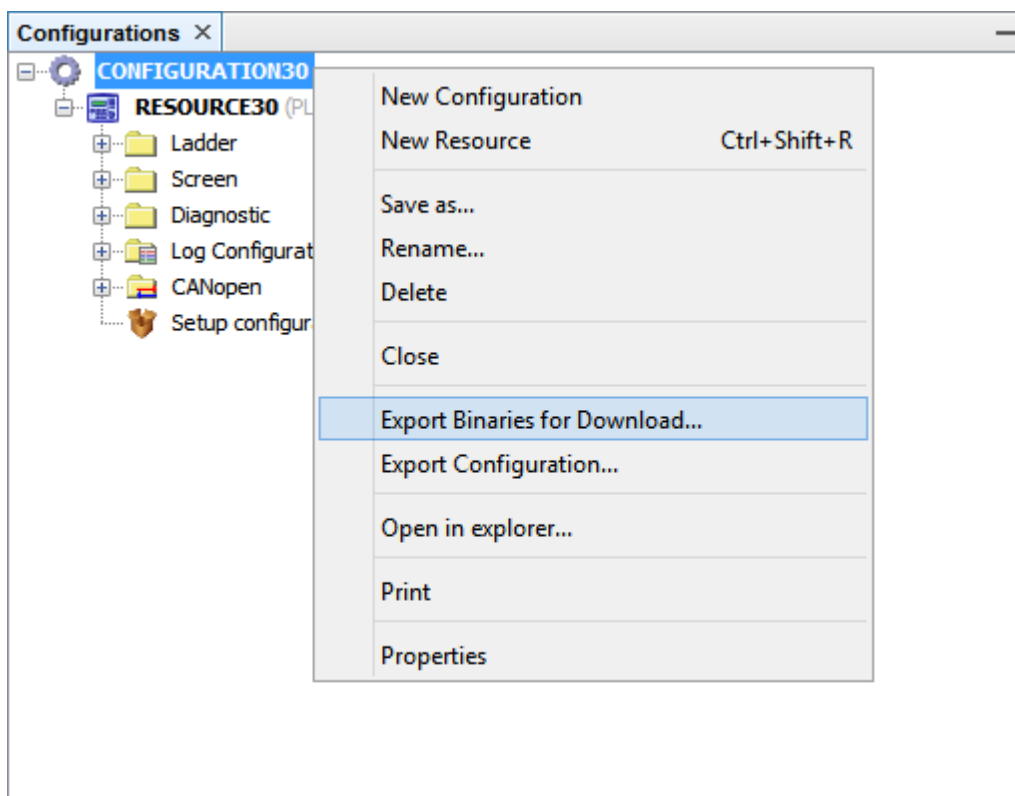
- Clique Ok para apagar ou
- Clique Cancelar para cancelar a operação.

Exportar Binários para Download

O WPS v2.3X permite que os binários de um recurso sejam exportados para um arquivo no formato *.bbp*. Isto permite que um recurso consolidado possa ser encapsulado e redistribuído sem que seu código fonte seja alterado.

1) Exportar da Configuração (Menu Popup)

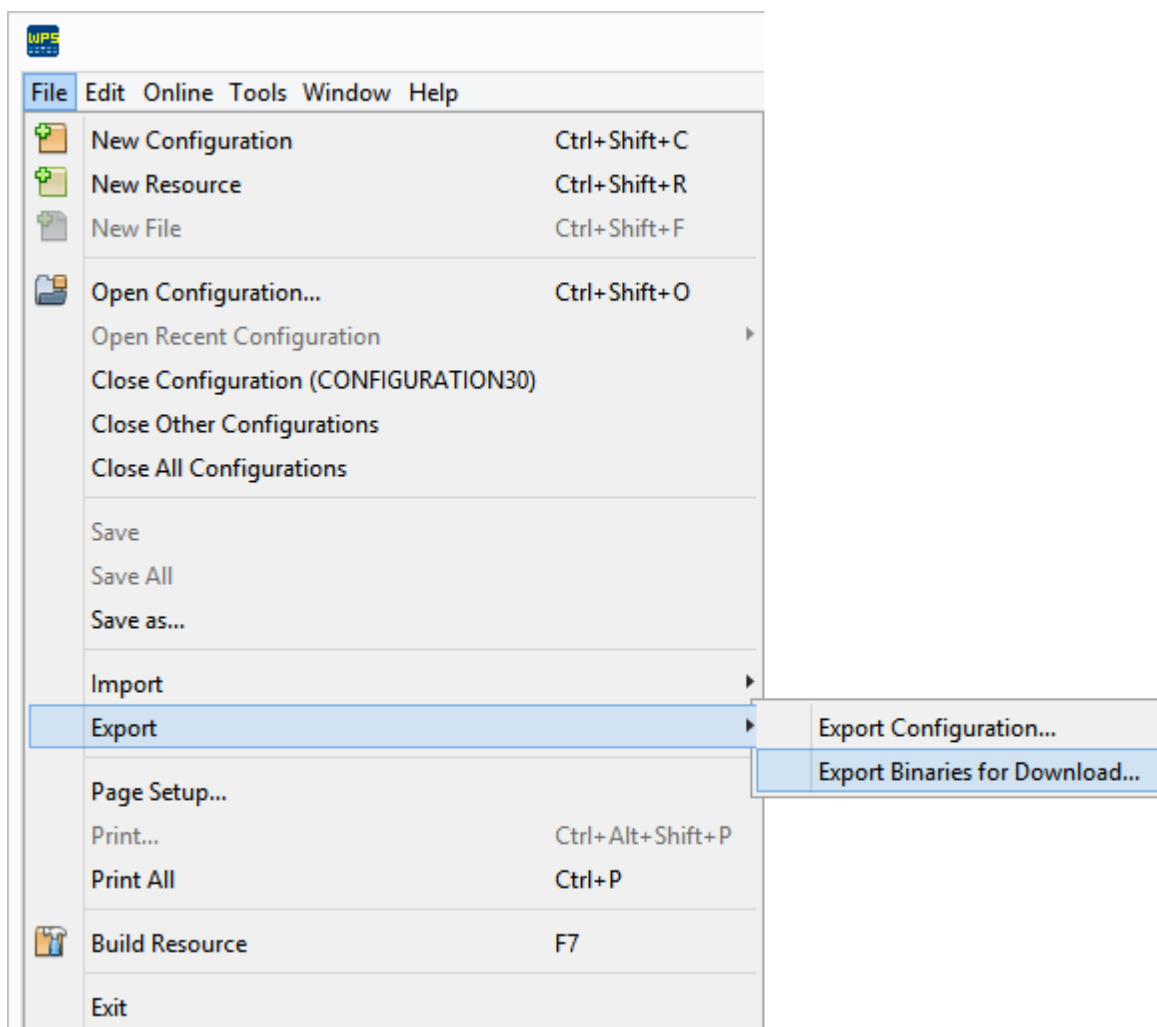
Clique com o botão direito sobre a configuração que contém o recurso a ser exportado e em seguida, no menu popup, clique em **Exportar Binários para Download**.



2) Exportar do Item de Menu

O item de menu Exportar Binários para Download também pode ser utilizado para exportar.

No menu Arquivo, clique em Exportar e depois em Exportar Binários para Download.

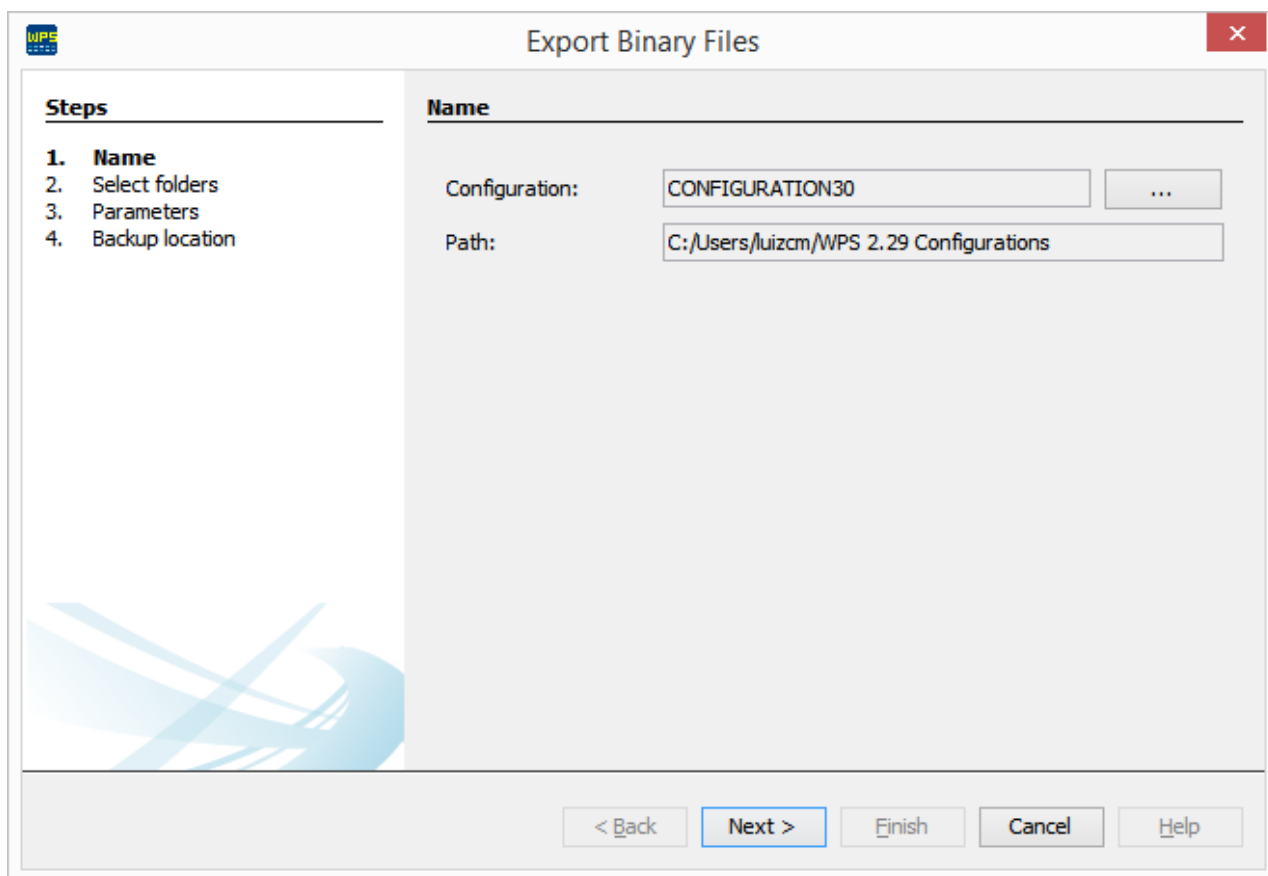


3) Seleção da Configuração

Um janela do assistente de exportação abrirá, exibindo as informações do nome e caminho da configuração que contém o recurso a ser exportado.

Clique no botão ... para modificar esses dados.

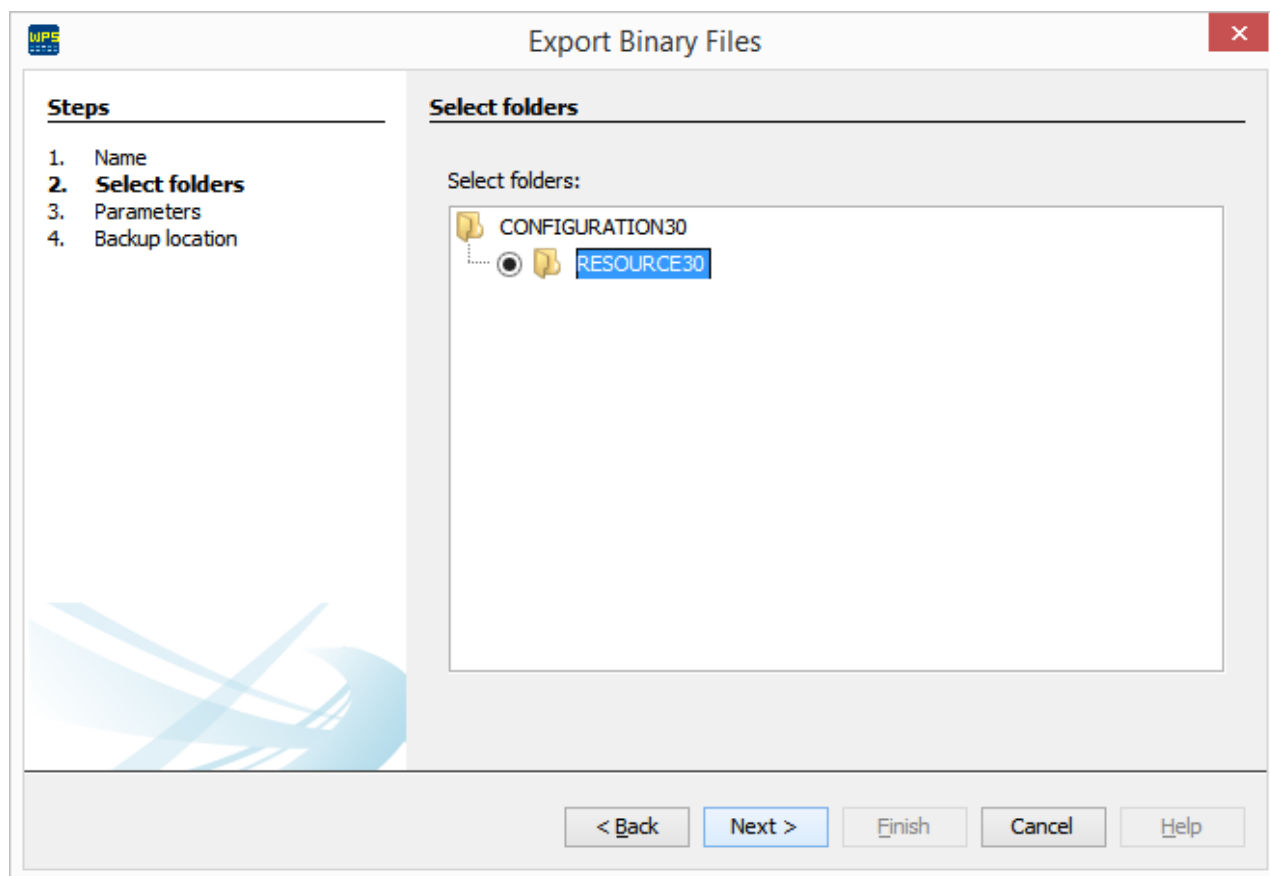
Em seguida clique em **Próximo**.



4) Seleção do Recurso

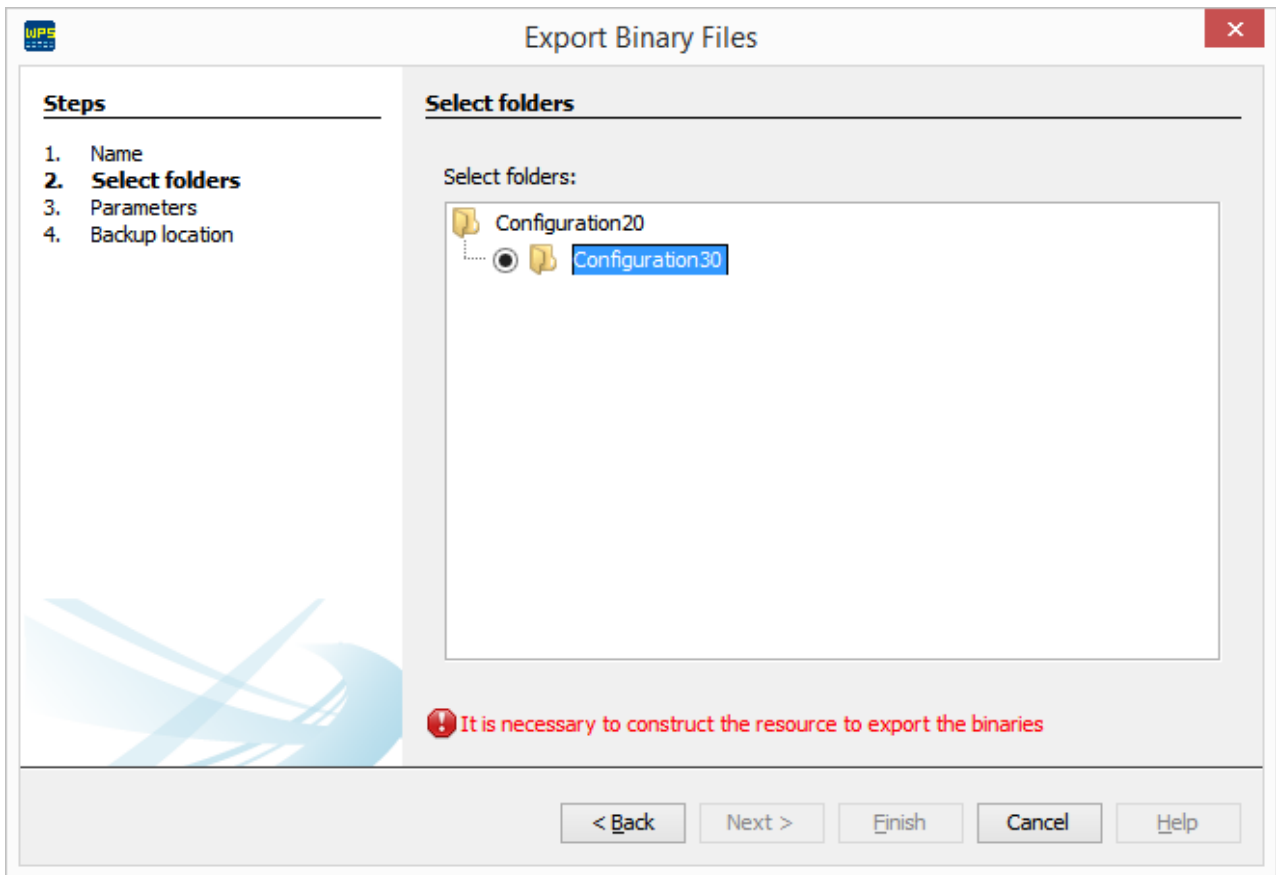
Em seguida aparecerá uma tela de seleção do recurso que possuirá seus binários exportados para o arquivo.

Selecione o recurso desejado e clique em **Próximo**.



5) Erro ao Exportar

Se o recurso a ser exportado não tiver sido compilado anteriormente, uma mensagem de erro aparecerá.



Clique em Cancelar para sair.

Compile o recurso e volte ao passo inicial deste tutorial.

**NOTA!**

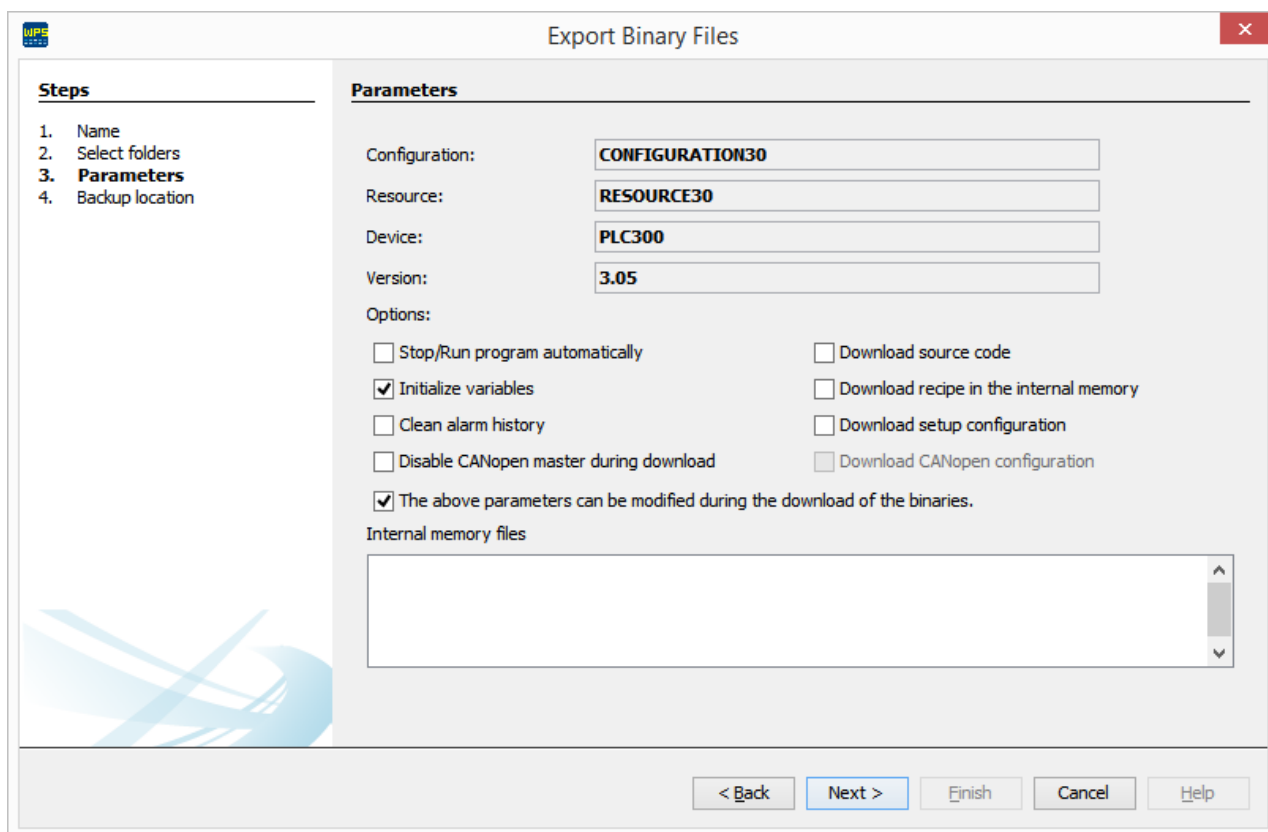
Sempre compile o recurso desejado antes de exportar seus arquivos binários.

6) Selecionar Opções

A próxima tela é semelhante à tela de *download* de recurso, permitindo que diversas opções de *download* sejam configuradas.

Consulte a seção [Download](#) para mais informações.

A última caixa de seleção define se estas configurações podem ser alteradas no momento do *download* dos binários.

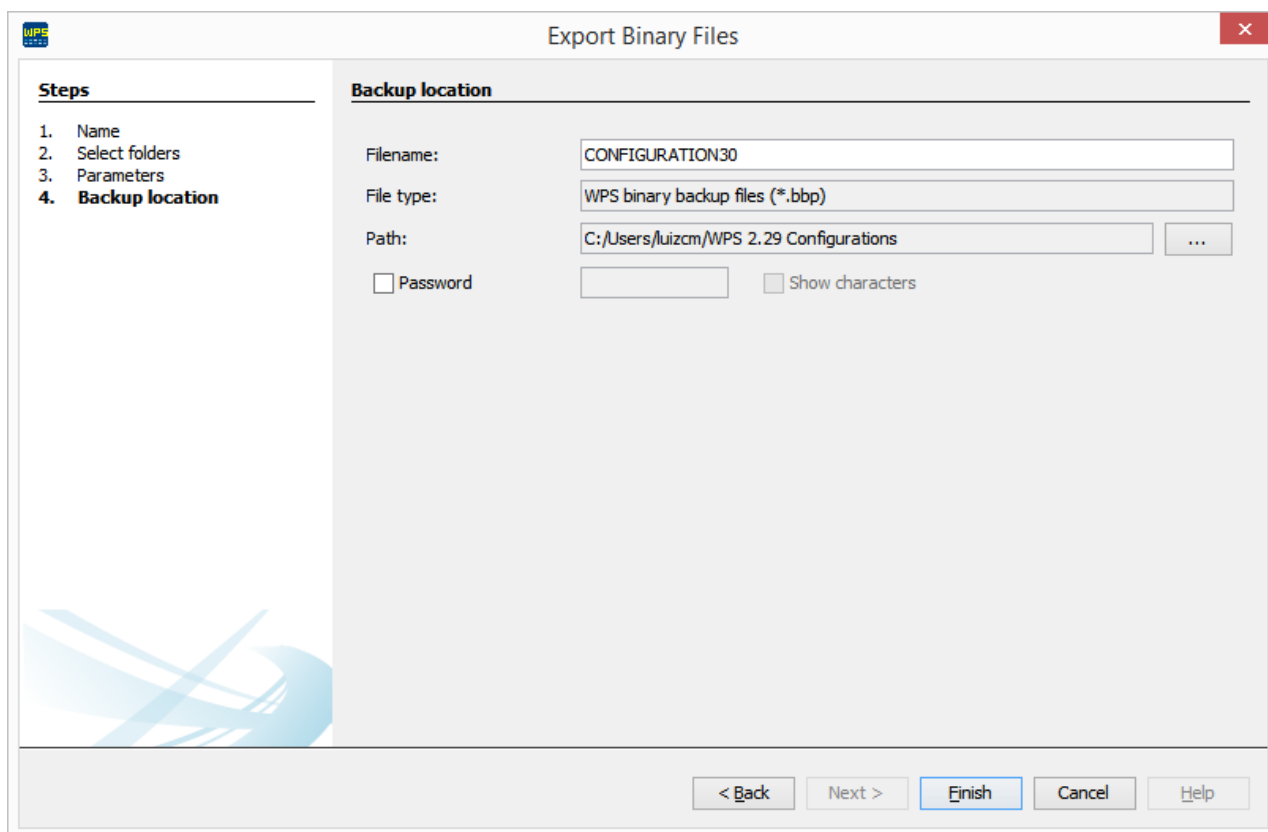


7) Localização do Arquivo Exportado

Por fim, uma última tela permite definir o nome do arquivo e sua localização. Clique no botão ... para modificar esses dados.

Esta tela permite também inserir uma senha de proteção para o arquivo. Caso desejado, marque a caixa de seleção e a insira.

Em seguida clique em **Finalizar**.



Exportar Configuração

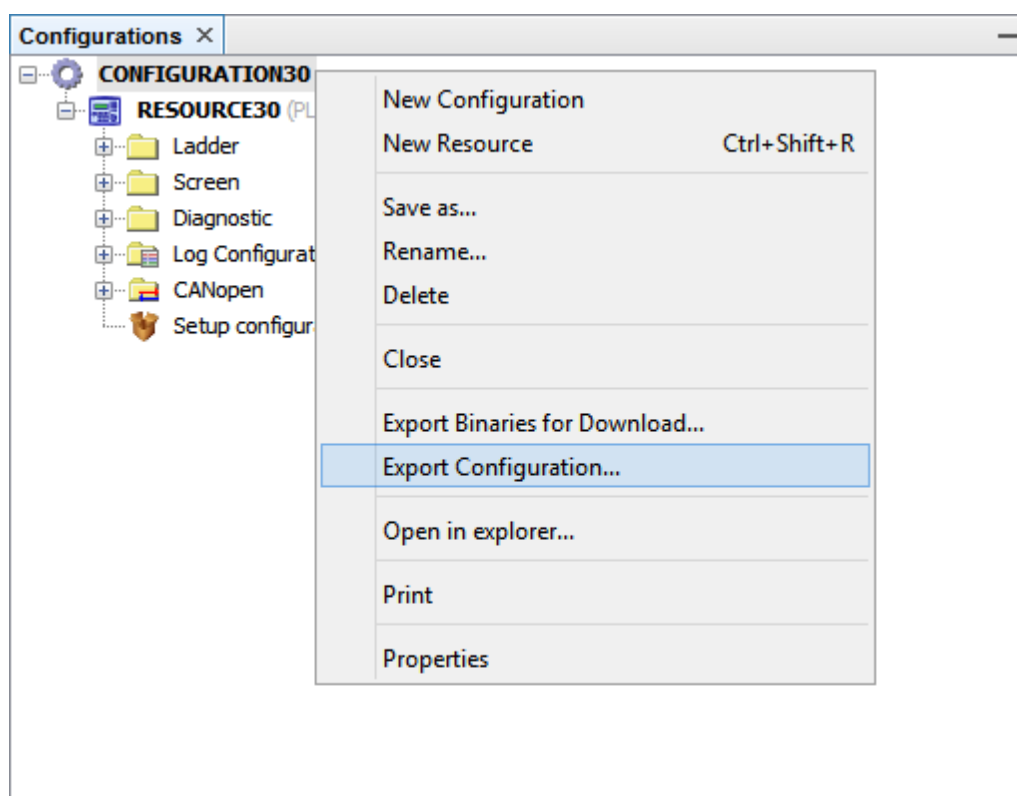
O WPS v2.3X permite que as configurações sejam exportadas para um arquivo de backup no formato *.bbp*.

Isto facilita o envio de uma configuração a outra máquina.

Recomenda-se sempre utilizar este procedimento para evitar que os arquivos sejam corrompidos.

1) Exportar da Configuração (Menu Popup)

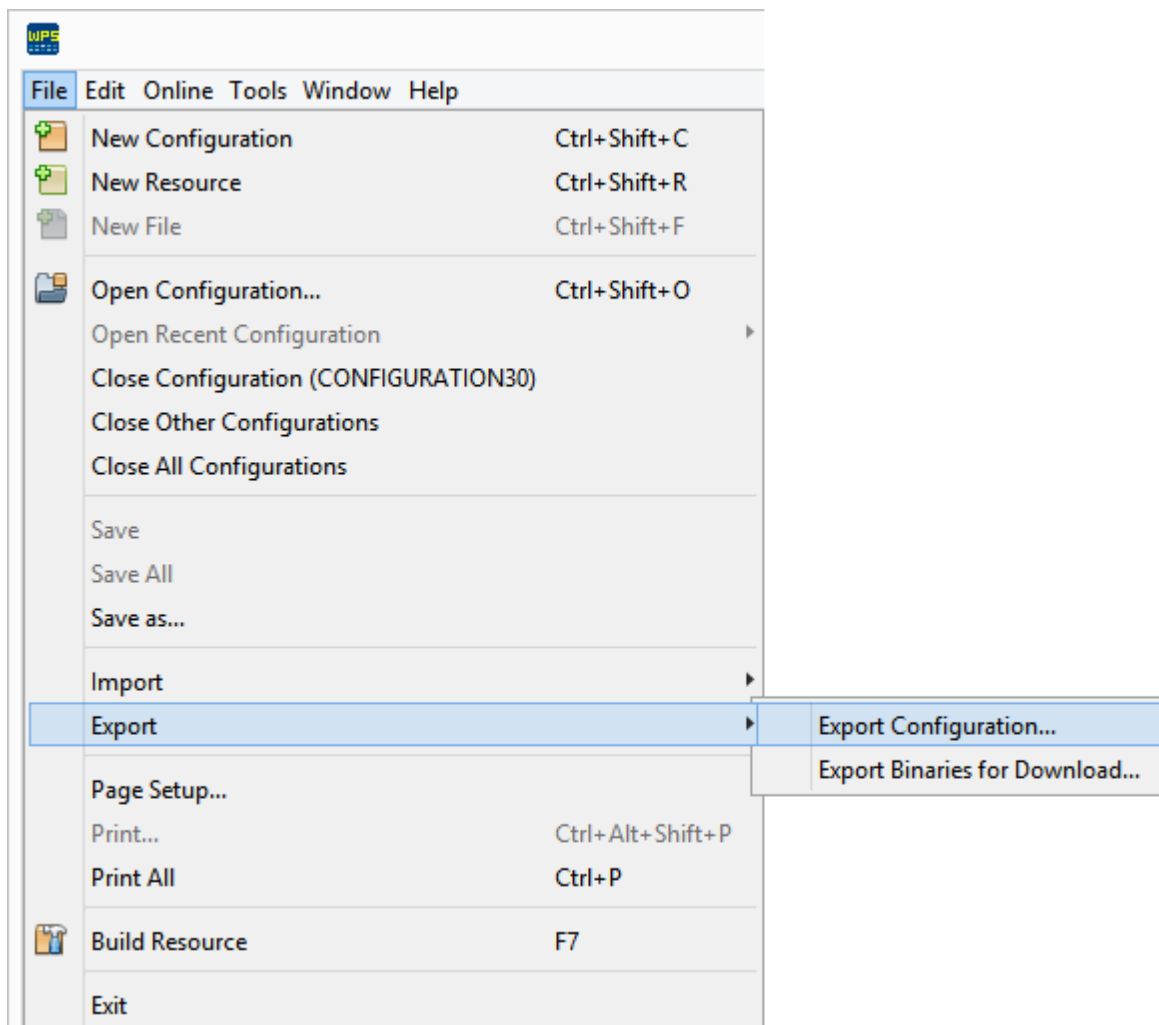
Clique com o botão direito sobre a configuração a ser exportada e em seguida, no menu popup, clique em Exportar Configuração.



2) Exportar do Item de Menu

O item de menu Exportar Configuração também pode ser utilizado para exportar.

No menu Arquivo, clique em Exportar e depois em Exportar Configuração.

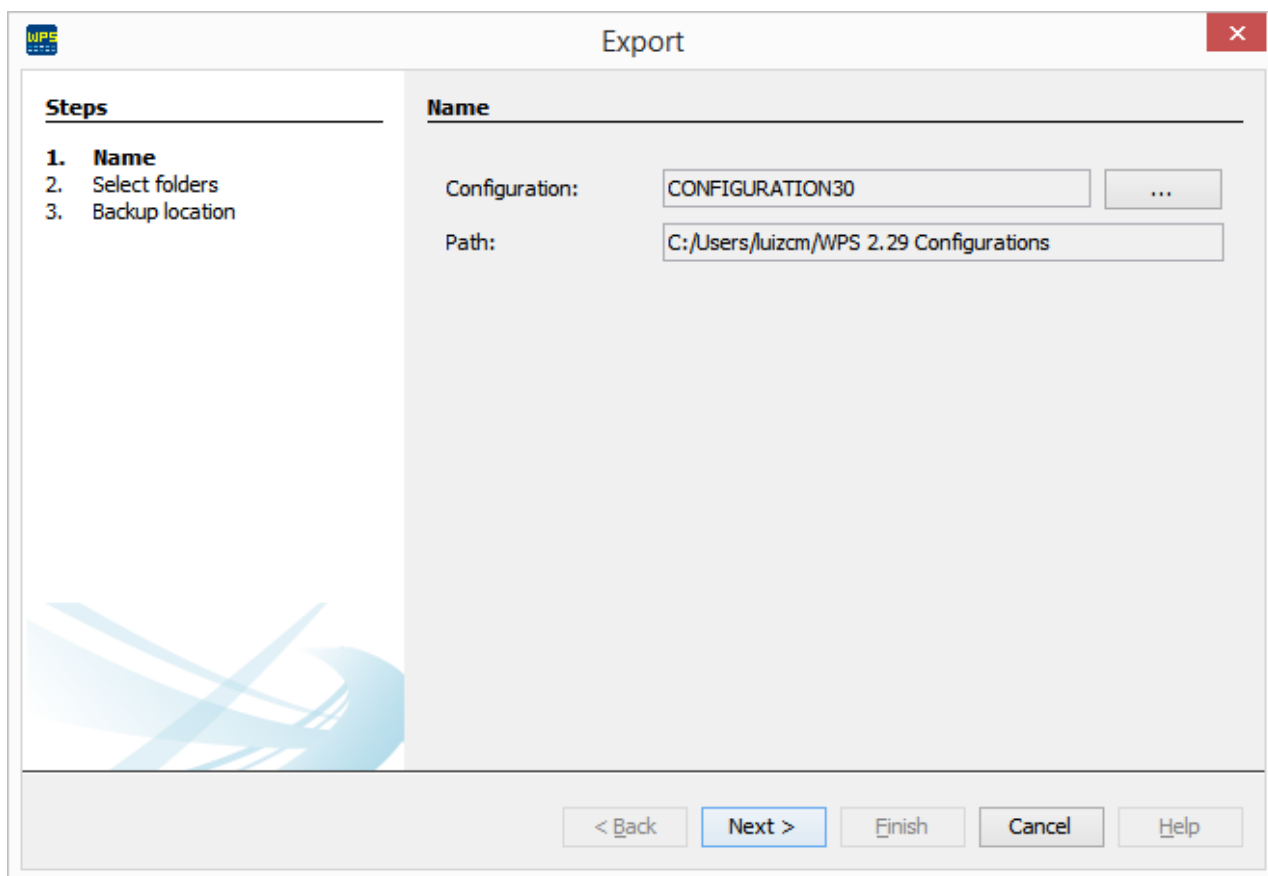


3) Seleção da Configuração

Um janela do assistente de exportação abrirá, exibindo o nome e caminho da configuração a ser exportada.

Clique no botão ... para modificar esses dados.

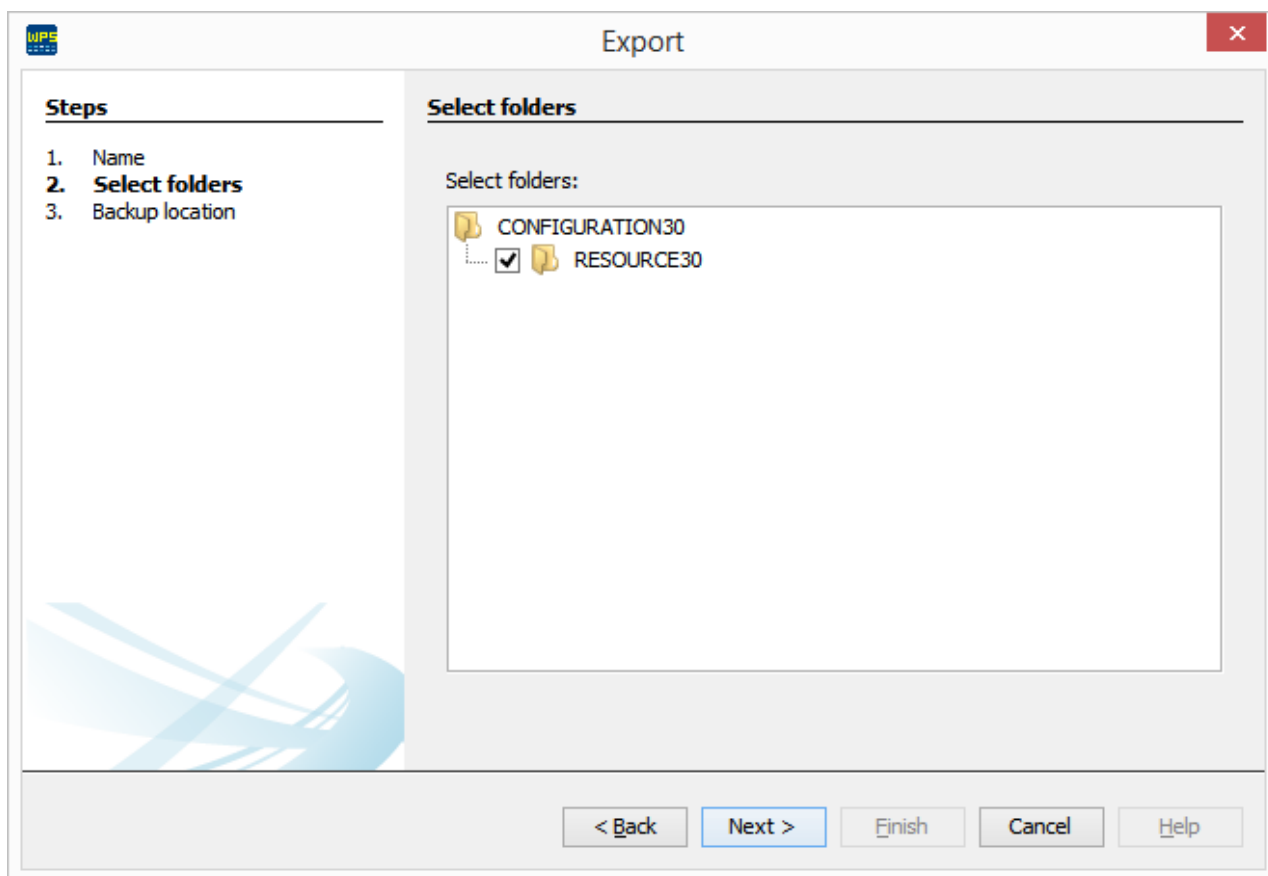
Em seguida clique em Próximo.



4) Seleção do Recurso

Em seguida, aparecerá uma tela de seleção dos recursos que serão exportados para o arquivo.

Selecione os recursos que deseja exportar e clique em **Próximo**.



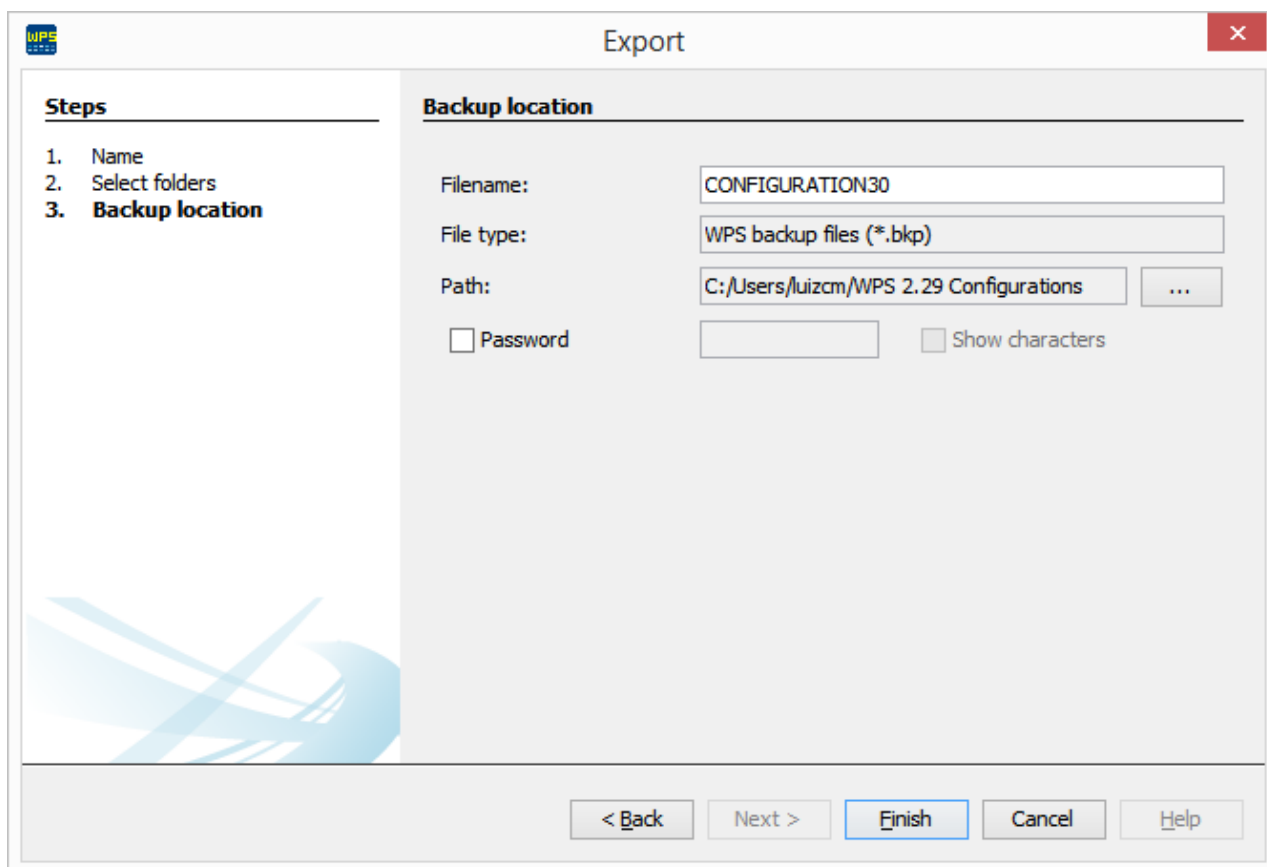
5) Selecionar Opções

Por fim, uma última tela permite definir o nome do arquivo de backup e sua localização.

Clique no botão ... para modificar esses dados.

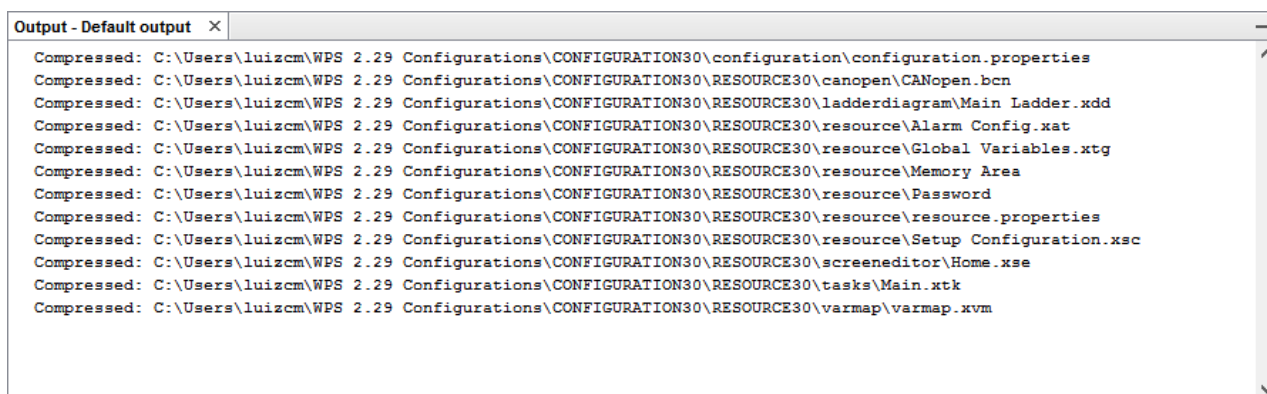
Esta tela permite também inserir uma senha de proteção para o arquivo. Caso desejado, marque a caixa de seleção e a insira.

Em seguida clique em **Finalizar**.



6) Aguarde até Finalizar

Na janela de saída pode-se visualizar mensagens informando a geração da configuração exportada.



Abrir no Explorer

Abre o Windows Explorer na pasta da configuração.

Usuário pode visualizar o conteúdo da pasta no Windows Explorer.

Imprimir

Selecione as opções que deseja imprimir.

- Clique Ok para imprimir ou
- Clique Cancelar para cancelar a operação.

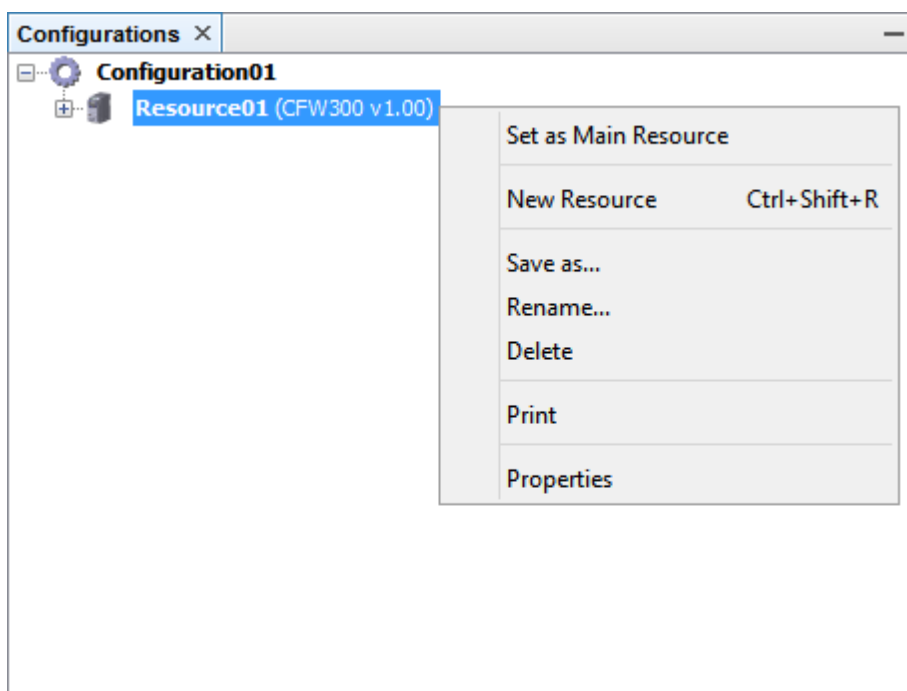
Propriedades

Clique no item de menu Propriedades, uma janela será apresentada com os seguintes itens:

- Name: nome da configuração;
- Path: rota da configuração;
- Size: tamanho da configuração.

6.9 Menu Popup - Recurso

O menu popup do recurso disponibiliza algumas funções descritas a seguir.



Clique no link para acesso rápido:

[Definir como Recurso Principal](#)

[Novo Recurso](#)

- - -

[Salvar Como...](#)

[Renomear...](#)

[Excluir](#)

[Imprimir](#)

[Propriedades](#)

Definir como Recurso Principal

Quando uma configuração possui diversos recursos, o recurso em uso deve ser definido como principal.

Para definir um recurso como principal, clique no item **Definir como Configuração Principal** no menu popup.

Após clicar o item de menu, o recurso aparece em negrito.



NOTA!

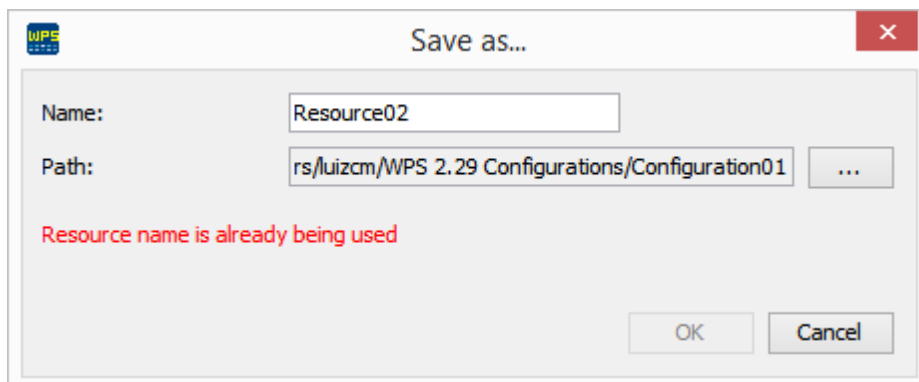
Lembre-se de sempre definir um recurso que se deseja trabalhar como principal, evitando erros de edição e *download*.

Novo Recurso

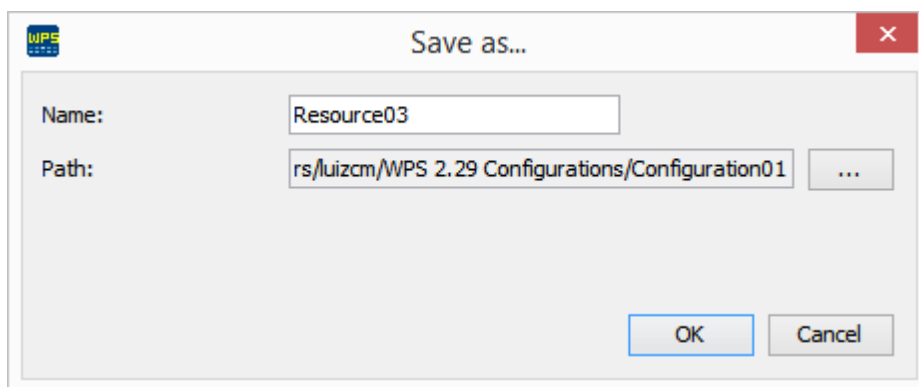
Consulte a seção [Criando Novo Recurso](#) para detalhes de como criar um novo recurso.

Salvar Como...

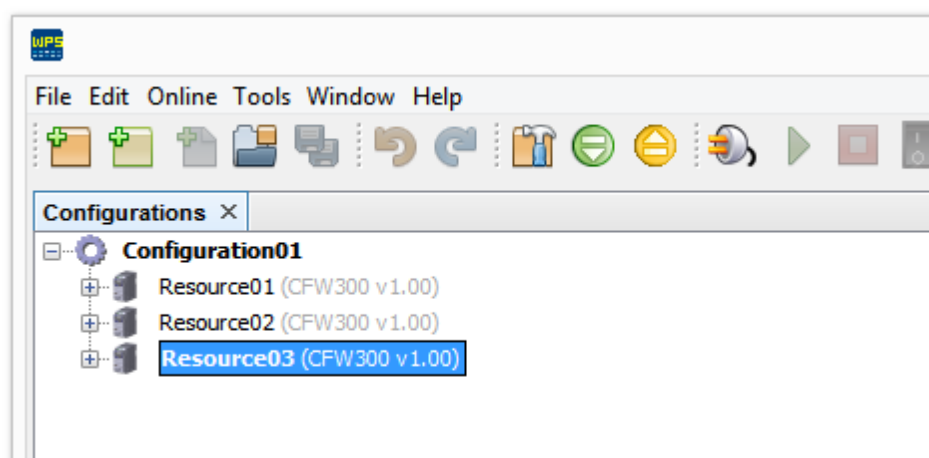
Para duplicar um recurso, clique no item de menu Salvar Como do menu popup.



No campo Nome digite o novo nome do recurso e clique no botão OK.



O novo recurso pode ser visualizado em Configurações.



Renomear

Para renomear um recurso, clique no item de menu Renomear do menu popup.

Preencha o campo Novo Nome.

- Clique Ok para aceitar ou
- Clique Cancelar para cancelar a operação.

Excluir

Para excluir um recurso, clique no item de menu Excluir do menu popup.

Uma mensagem solicita confirmação para apagar o recurso.

- Clique Ok para apagar ou
- Clique Cancelar para cancelar a operação.

Imprimir

Selecione as opções que deseja imprimir.

- Clique Ok para imprimir ou
- Clique Cancelar para cancelar a operação.

Propriedades

Para visualizar/editar as propriedades do recurso, clique no item de menu Propriedades do menu popup.

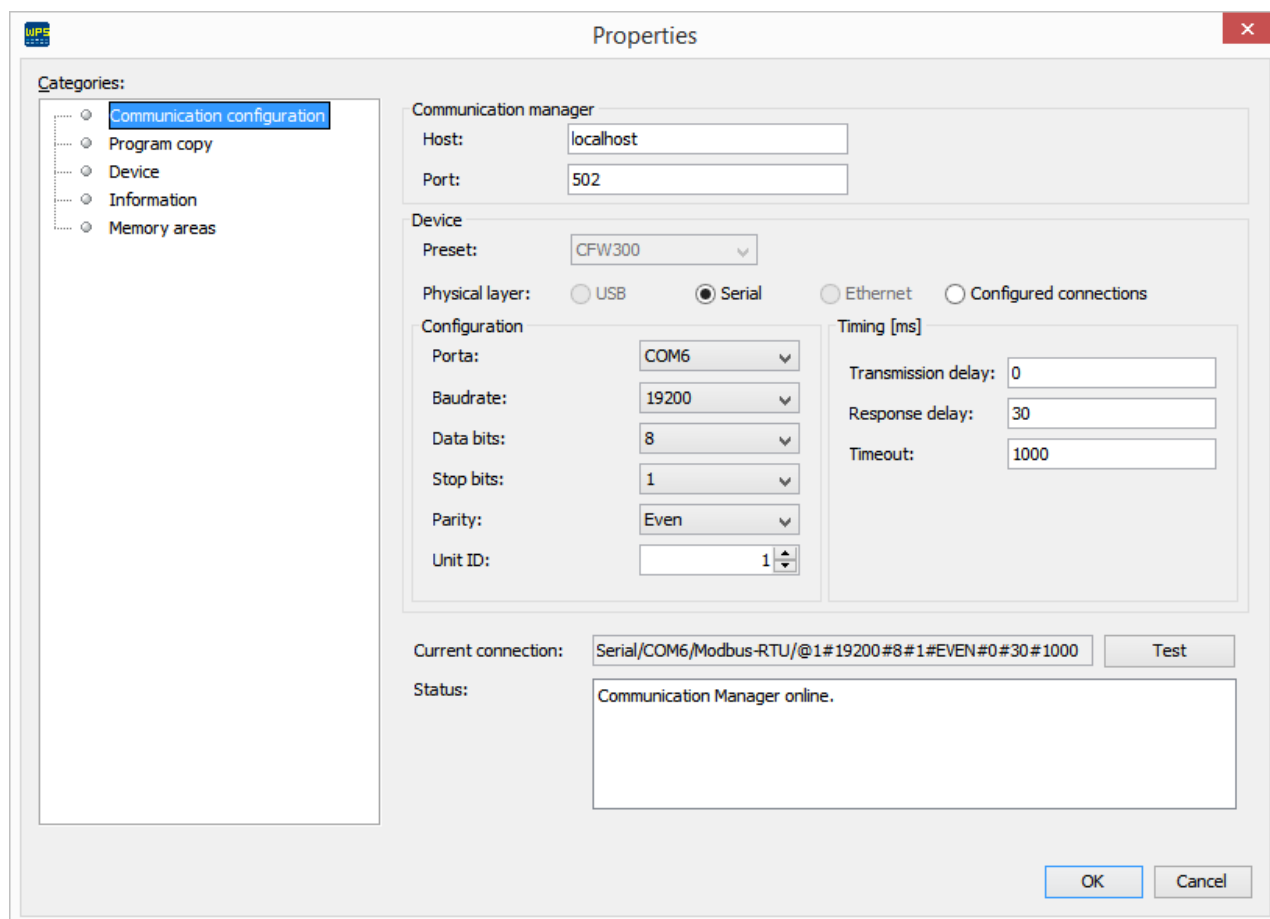
A janela de propriedades apresenta algumas categorias:

- Configuração da Comunicação;
- Cópia do Programa;
- Equipamento;
- Informação;
- Áreas de Memória.

1) Configuração da Comunicação

Na categoria Configuração da Comunicação pode-se alterar diversas variáveis de comunicação:

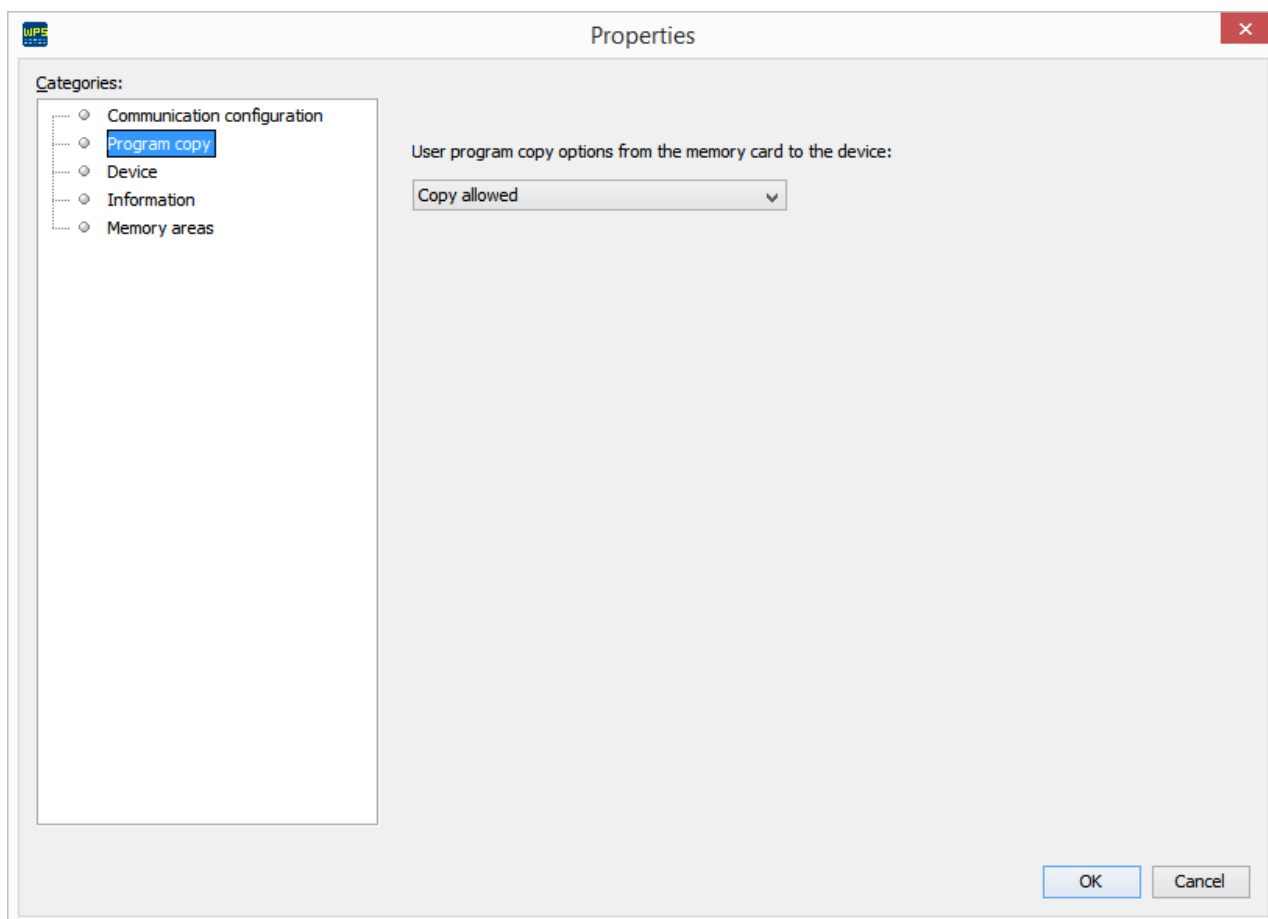
- Nome do host e porta do gerenciador de comunicação;
- Equipamento;
- Camada física;
- Ajustes da porta;
- Tempos;
- Teste de comunicação;
- Estado da comunicação.



2) Cópia do Programa

Na categoria Cópia do Programa, uma caixa de seleção permite selecionar o tipo de cópia do programa do usuário do cartão de memória para o equipamento:

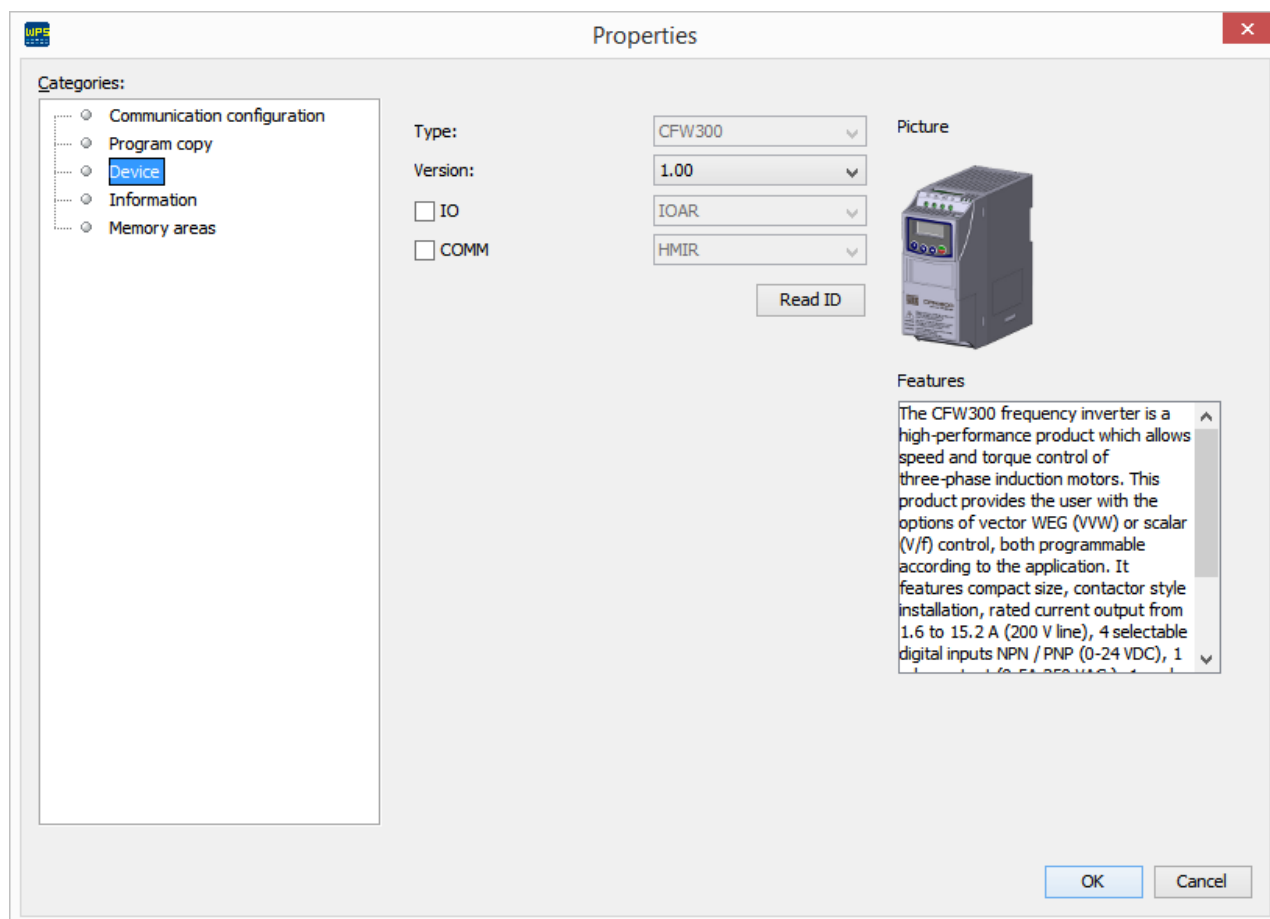
- Permite copiar: permite que o programa ladder possa ser copiado para um módulo de memória flash (CFW300-MMF);
- Permite somente uma cópia: permite que o programa ladder possa ser copiado para um módulo de memória flash (CFW300-MMF), o qual permite carregá-lo em outro equipamento; mas não permite copiar deste equipamento para outro módulo de memória flash; em outras palavras, não permite cópia de cópia;
- Não permite copiar: não permite que o programa ladder possa ser copiado para um módulo de memória flash (CFW300-MMF);



3) Equipamento

Na categoria Equipamento pode-se:

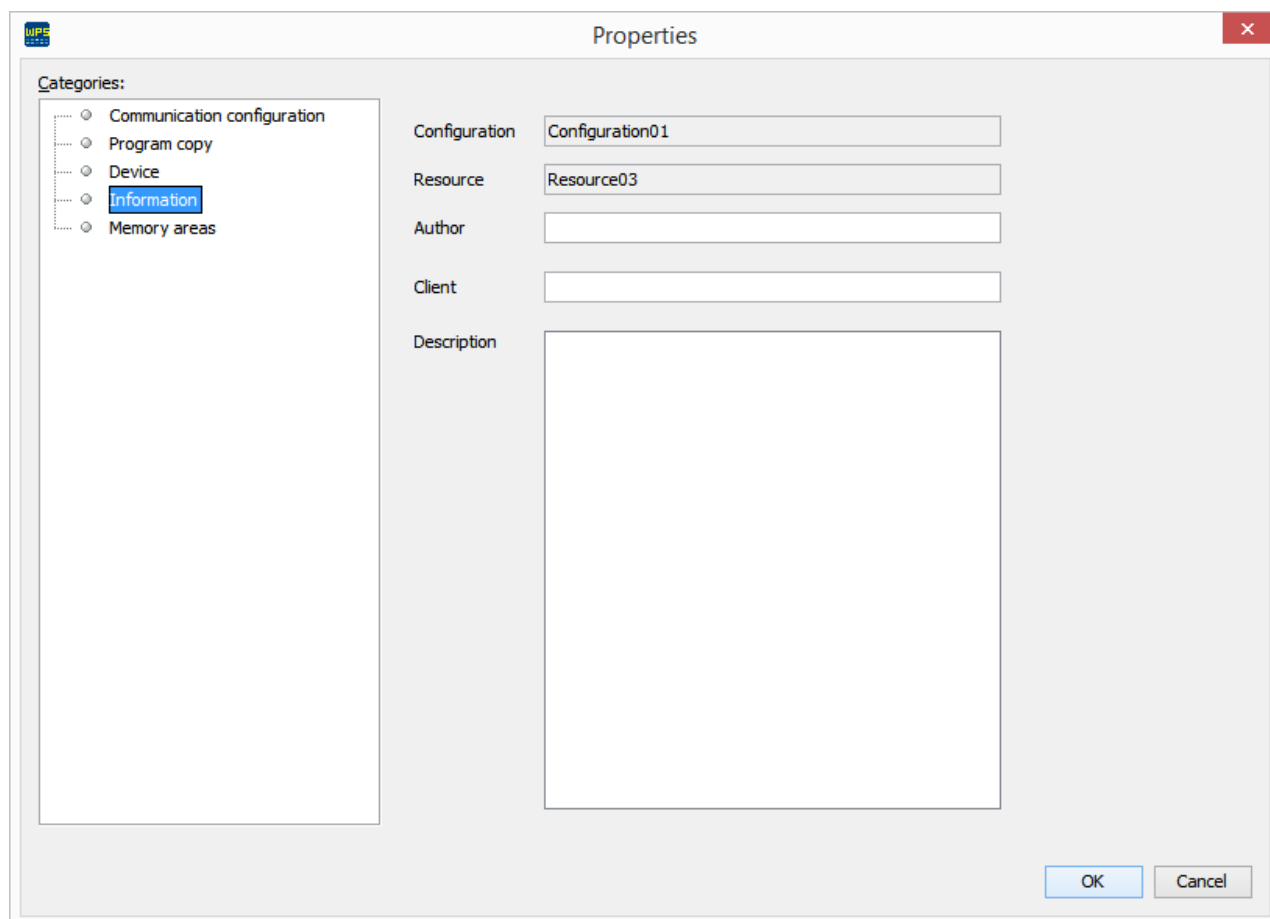
- Alterar versão de firmware;
- Selecionar/remover acessórios;
- Ler a identificação do equipamento, visualizando o modelo e versão de firmware.



4) Informação

Na categoria informação pode-se visualizar/alterar dados da configuração:

- Autor;
- Cliente;
- Descrição.



5) Áreas de Memória

Na categoria áreas de memória é possível alterar as quantidades de memória (em bytes) alocadas para cada seção do recurso.

- Memória alocada: as caixas de texto editáveis exibem as informações de memória alocada, permitindo que estas sejam alteradas pelo usuário;
- Memória ocupada: as caixas de texto não editáveis exibem as informações de memória ocupada atualmente pelo recurso carregado no equipamento;
- Limites: ao lado destas caixas é exibida a faixa de valores de memória alocada aceitáveis para cada campo.

Um clique no botão Alocada exibe um gráfico da memória alocada.

Um clique no botão Ocupada exibe um gráfico da memória ocupada.



Properties

✕

Categories:

- Communication configuration
- Program copy
- Device
- Information
- Memory areas

Memory Areas	Allocated	Occupied	Limits
● Volatile	11776	0	11776 - 11776
● Ladder	11776	0	11776 - 11776
● Information	11776	0	11776 - 11776
● User parameters properties	204	0	204 - 204
● User parameters ranges	204	0	204 - 204
● User parameters values	104	0	104 - 104

Reset to Default

OK

Cancel

7 Comunicação

[Parametrização do Equipamento](#)

[Estabelecendo Comunicação - Porta Serial USB](#)

[Estabelecendo Comunicação - RS232](#)

[Estabelecendo Comunicação - RS485](#)

[Cabos](#)

[Conversor USB/Serial](#)

7.1 Parametrização do Equipamento

Os seguintes parâmetros devem ser ajustados para estabelecer a comunicação com os equipamentos:

Equipamento	Acessório Necessário	Tipo de Conexão	Driver	Parâmetro	Descrição	Recomendado ou Ajuste de Fábrica
CFW300	CFW300 com módulo CFW300-CRS232	RS232	----	0308	Endereço Serial	1
				0310	Taxa Comunic. Serial	1 = 19200 bits/s
				0311	Config. Bytes Serial	1 = 8 bits, paridade par, 1 sb
				0312	Protocolo Serial	2 = Modbus RTU
	CFW300 com módulo CFW300-CRS485	RS485	Drive USB fornecido pelo fabricante do Conversor Isolado USB para 485 (dispositivo externo)	0308	Endereço Serial	1
				0310	Taxa Comunic. Serial	1 = 19200 bits/s
				0311	Config. Bytes Serial	1 = 8 bits, paridade par, 1 sb
				0312	Protocolo Serial	2 = Modbus RTU
	CFW300 com módulo CFW300-CUSB	Porta Serial USB	FTDI	0308	Endereço Serial	1
				0310	Taxa Comunic. Serial	1 = 19200 bits/s
				0311	Config. Bytes Serial	1 = 8 bits, paridade par, 1 sb

Equipamento	Acessório Necessário	Tipo de Conexão	Driver	Parâmetro	Descrição	Recomendado ou Ajuste de Fábrica
				0312	Protocolo Serial	2 = Modbus RTU
CVW500	----	RS232	----	0308	Endereço Serial	1
				0310	Taxa Comunicação Serial	1 = 19200 bits/s
				----	Configuração Bytes Serial	8 bits de dados, paridade par, 1 bit de parada
				----	Protocolo Serial	Modbus RTU
SCA06	----	USB	WEG USB Driver	----	-----	-----
PLC300	----	USB	WEG USB Driver	----	-----	-----

7.2 Estabelecendo Comunicação - Porta Serial USB

A porta serial USB também é conhecida como Porta COM Virtual.

O equipamento deve estar localizado à uma distância menor que 3 m do computador.

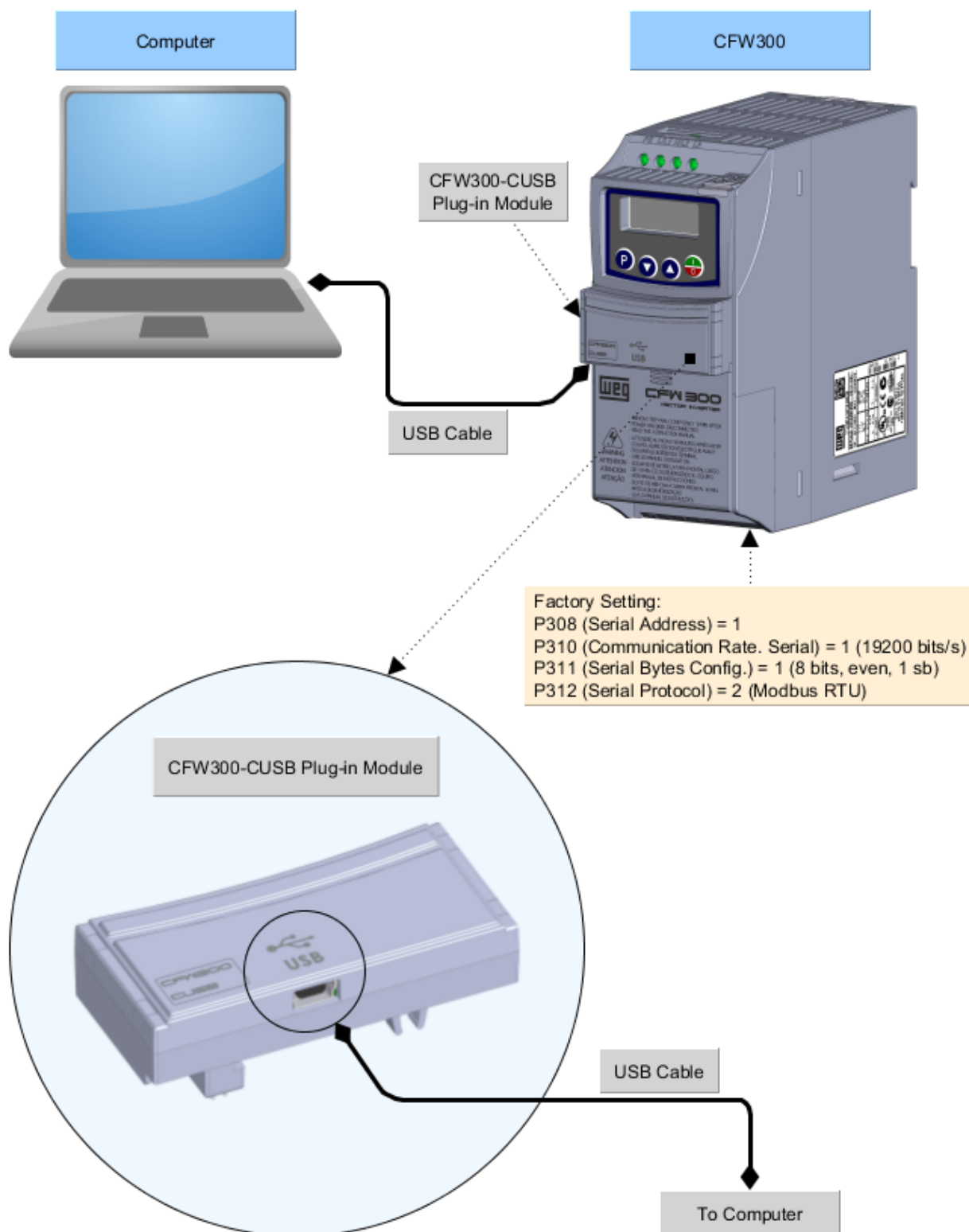
Esta conexão é feita com o Cabo USB.

Driver USB: é necessário instalar o driver USB FTDI disponível na pasta USB_Driver\FTDI.

CFW300

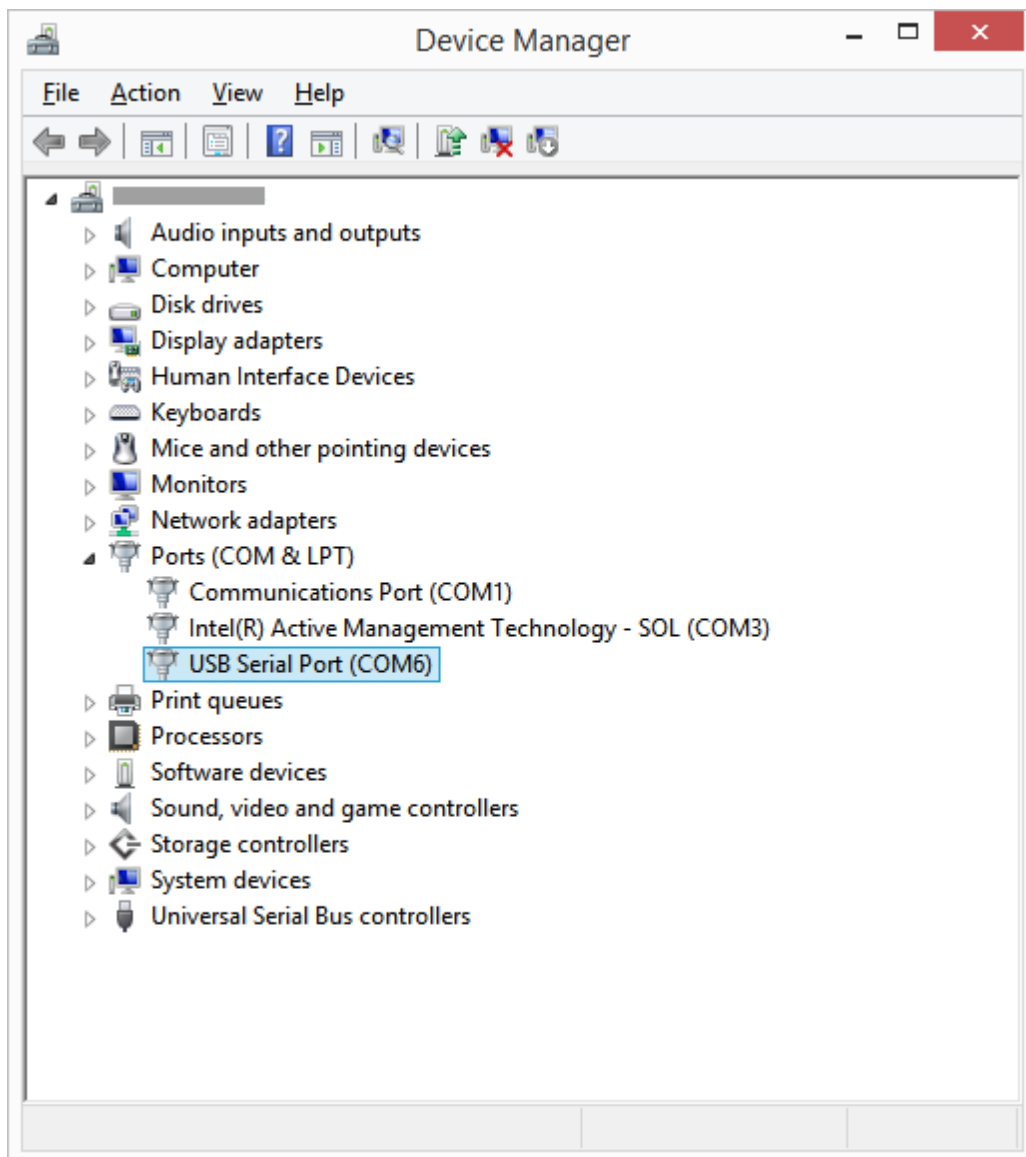
1) Conexão do Computador ao Equipamento

A figura abaixo mostra como conectar um computador ao equipamento via USB.



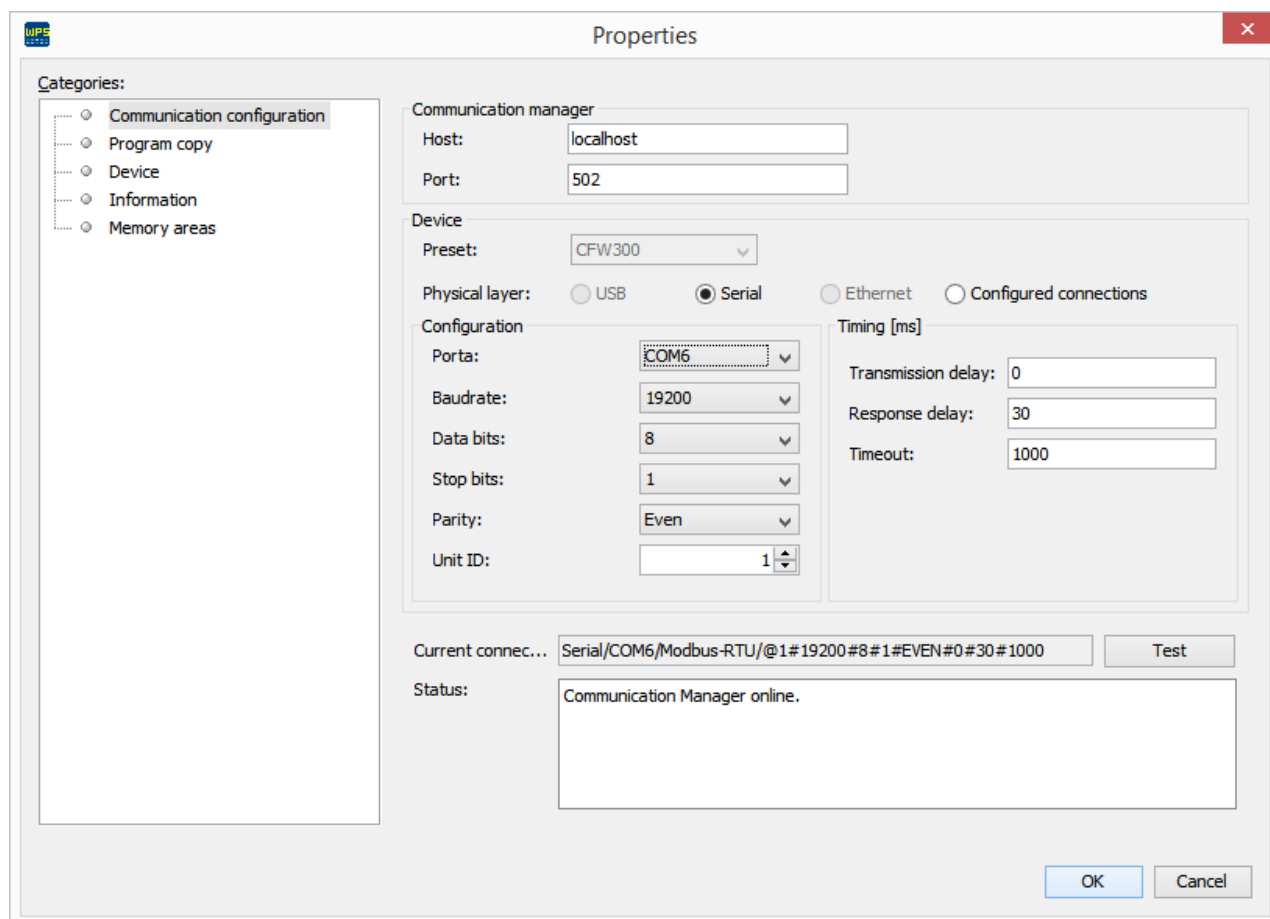
2) Gerenciador de Dispositivo do Windows

Gerenciador de dispositivo do Windows indica a porta serial conectada ao equipamento. O nome do computador no gerenciador de dispositivo está propositalmente ilegível.



3) Configuração da Comunicação no WPS

No WPS, selecione a porta serial corretamente na janela Propriedade > Configuração da Comunicação conforme a seguir.

**NOTA!**

Desligar o equipamento antes de fazer as conexões.

4) Conectando o Equipamento

1. Insira o conector mini-B do cabo usb no conector usb do equipamento;
2. Insira o conector tipo A do cabo usb na porta usb do computador;
3. No gerenciador de dispositivo verifique qual porta serial está conectada;
4. Certifique-se que na categoria Configuração de Comunicação da janela Propriedades do recurso, a conexão serial esteja selecionada;
5. A porta serial e sua configuração do recurso no WPS deve ser a mesma porta serial que aparece no gerenciador de dispositivo do Windows, onde o cabo USB está conectado;
6. Nunca alterar valores nos parâmetros P308, P310, P311 e P312 durante uma conexão. A alteração destes parâmetros causa a perda instantânea de comunicação entre o PC e o equipamento.

7.3 Estabelecendo Comunicação - RS232

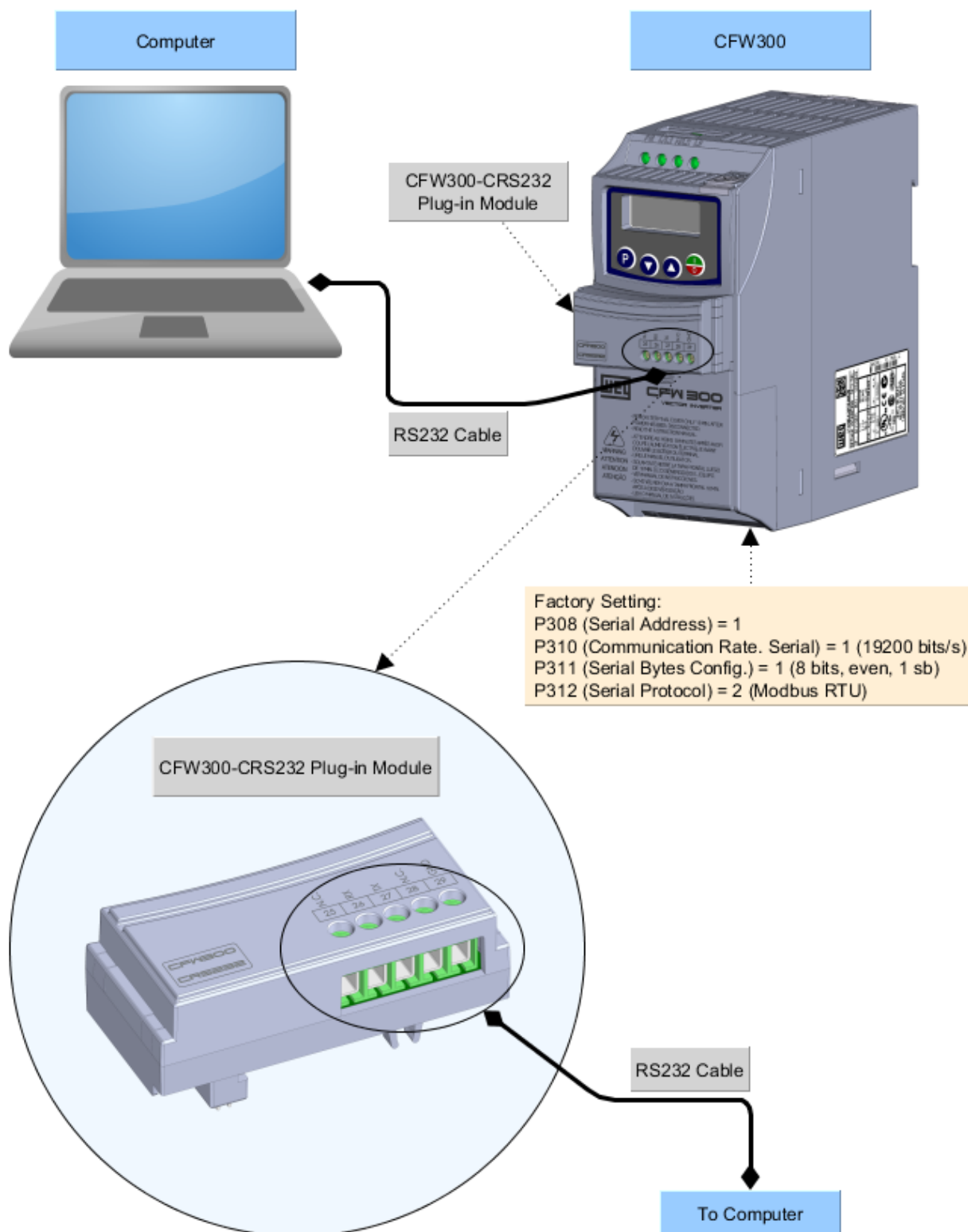
O equipamento deve estar localizado à uma distância do computador conforme tabela abaixo.

Taxa de Transmissão (bps)	Tamanho Máximo do Cabo (pes)	Tamanho Máximo do Cabo (m)
9600	32.81	10
19200	24.93	7.6
38400	12.14	3.7

CFW300

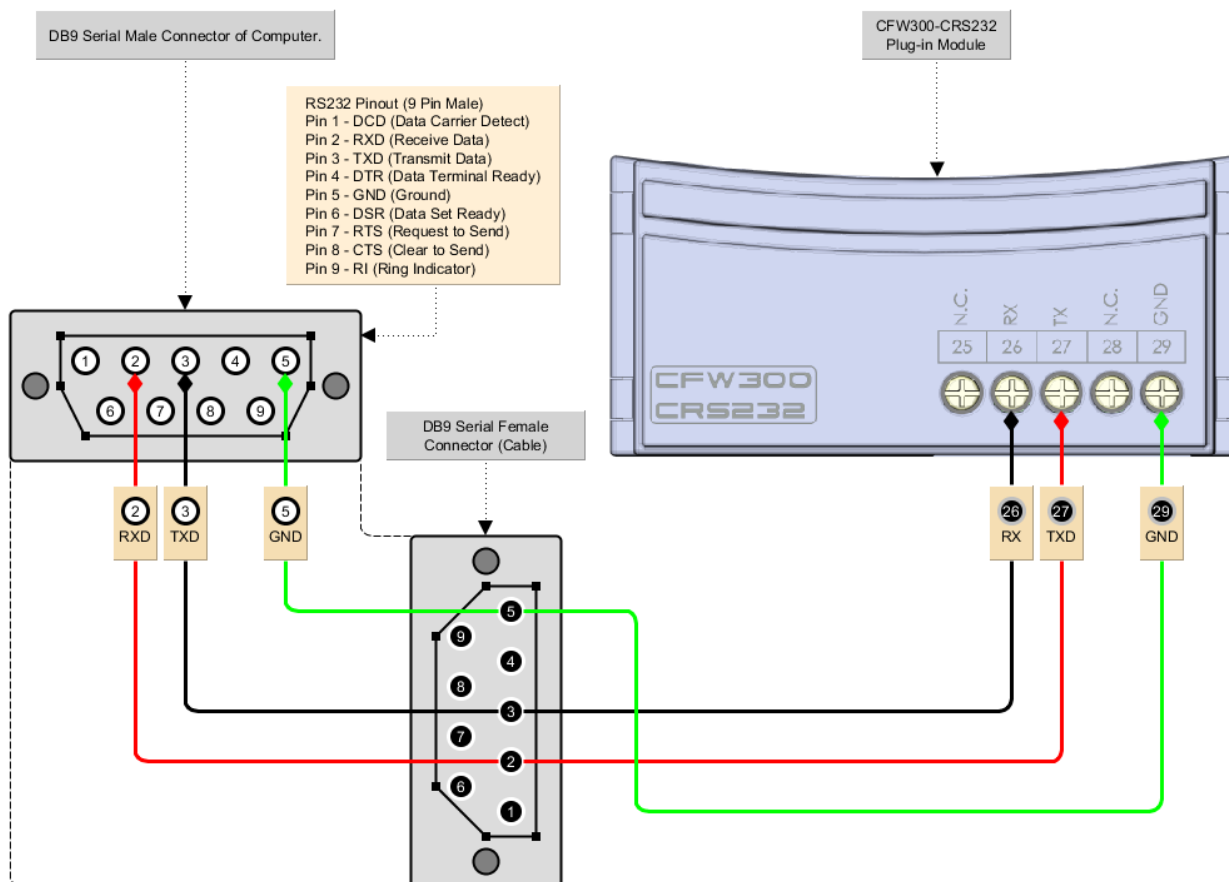
1) Conexão do Computador ao Equipamento

A figura abaixo mostra como conectar um computador ao equipamento via RS232.



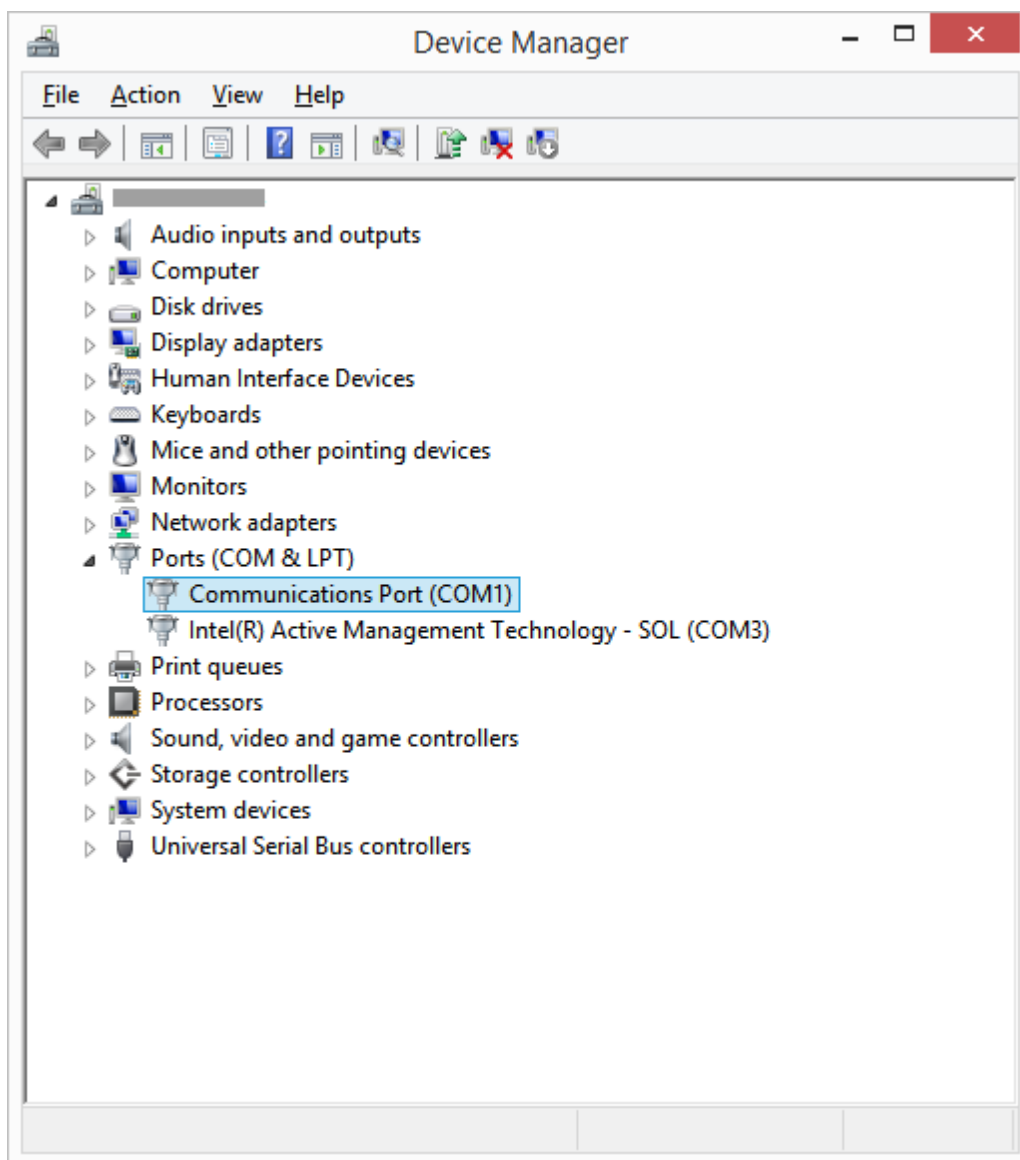
2) Conexão Computador - Módulo Plug-in

A figura abaixo mostra detalhes da conexão.



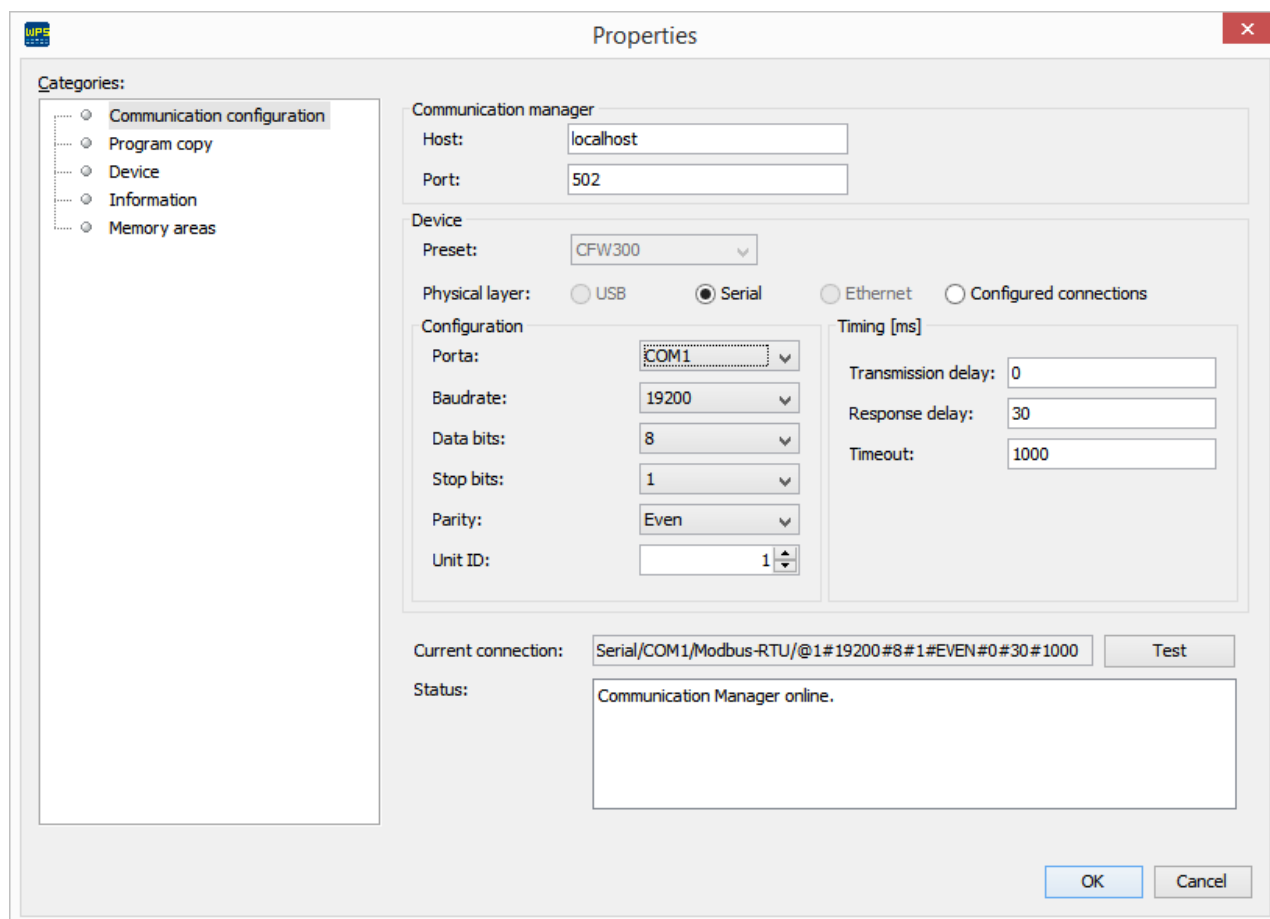
3) Gerenciador de Dispositivo do Windows

Gerenciador de dispositivo do Windows indica a porta serial conectada ao equipamento. O nome do computador no gerenciador de dispositivo está propositalmente ilegível.



4) Configuração da Comunicação no WPS

No WPS, selecione a porta serial corretamente na janela Propriedade > Configuração da Comunicação conforme a seguir.

**NOTA!**

Desligar o equipamento antes de fazer as conexões.

5) Conectando o Equipamento

1. Faça as ligações do conector DB9 fêmea serial do cabo RS232 ao acessório CFW300-RS232 conforme apresentado nas figuras anteriores;
2. Insira o conector DB9 fêmea do cabo RS232 no conector DB9 macho do computador;
3. No gerenciador de dispositivo verifique qual porta serial está conectada;
4. Certifique-se que na categoria Configuração de Comunicação da janela Propriedades do recurso, a conexão serial esteja selecionada;
5. A porta serial e sua configuração do recurso no WPS deve ser a mesma porta serial que aparece no gerenciador de dispositivo do Windows, onde o cabo RS232 está conectado;
6. Nunca alterar valores nos parâmetros P308, P310, P311 e P312 durante uma conexão. A alteração destes parâmetros causa a perda instantânea de comunicação entre o PC e o equipamento.

7.4 Estabelecendo Comunicação - RS485

O equipamento deve estar localizado à uma distância do computador conforme tabela abaixo.

Taxa de Transmissão (bps)	Tamanho Máximo do Cabo (pes)	Tamanho Máximo do Cabo (m)
9600	3280.84	1000
19200	3280.84	1000
38400	3280.84	1000

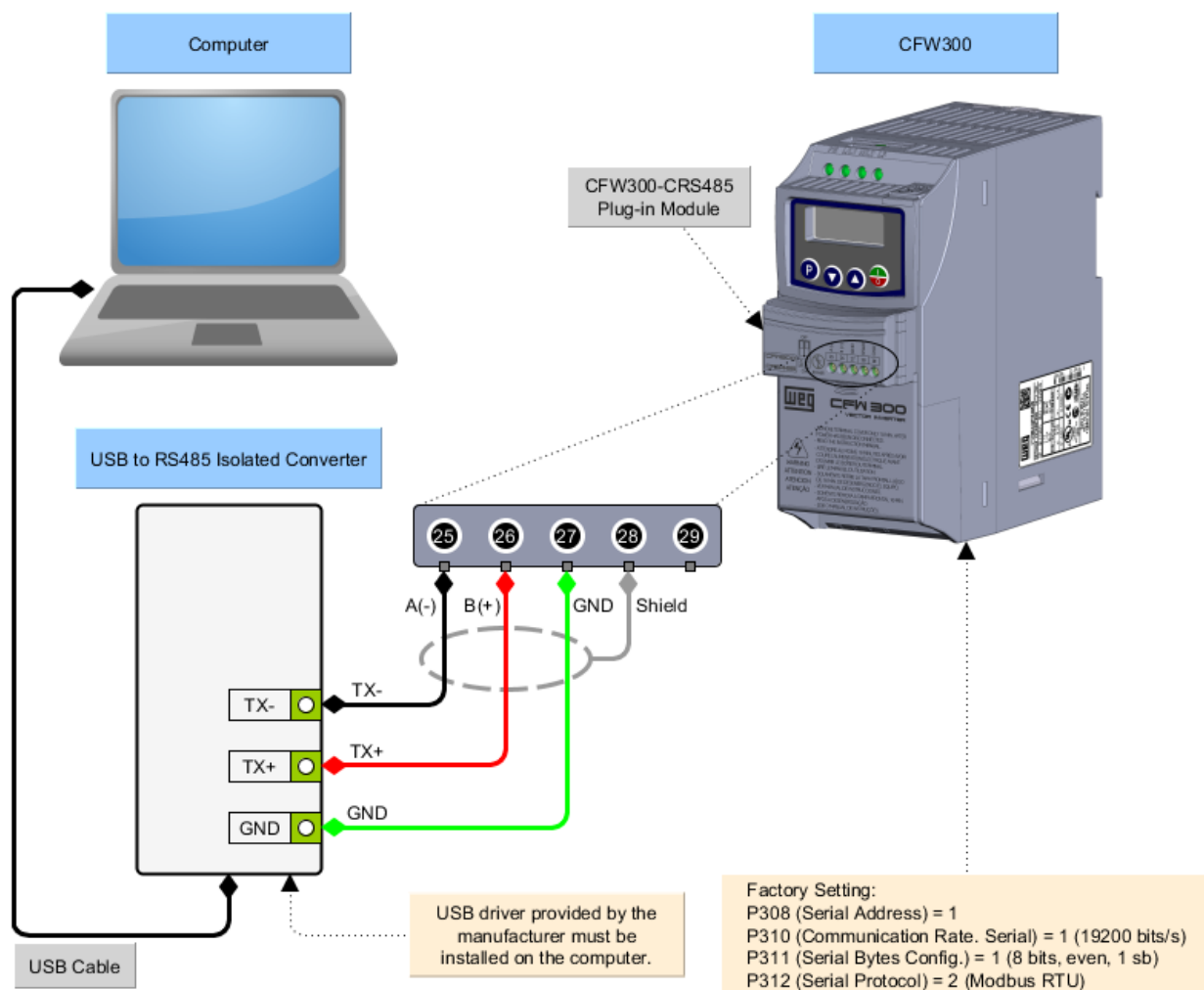
A conexão do computador ao conversor isolado USB para RS485 é feita com o Cabo USB.

Neste caso, é necessário instalar o driver USB do fabricante do conversor isolado, que é normalmente fornecido pelo próprio fabricante.

CFW300

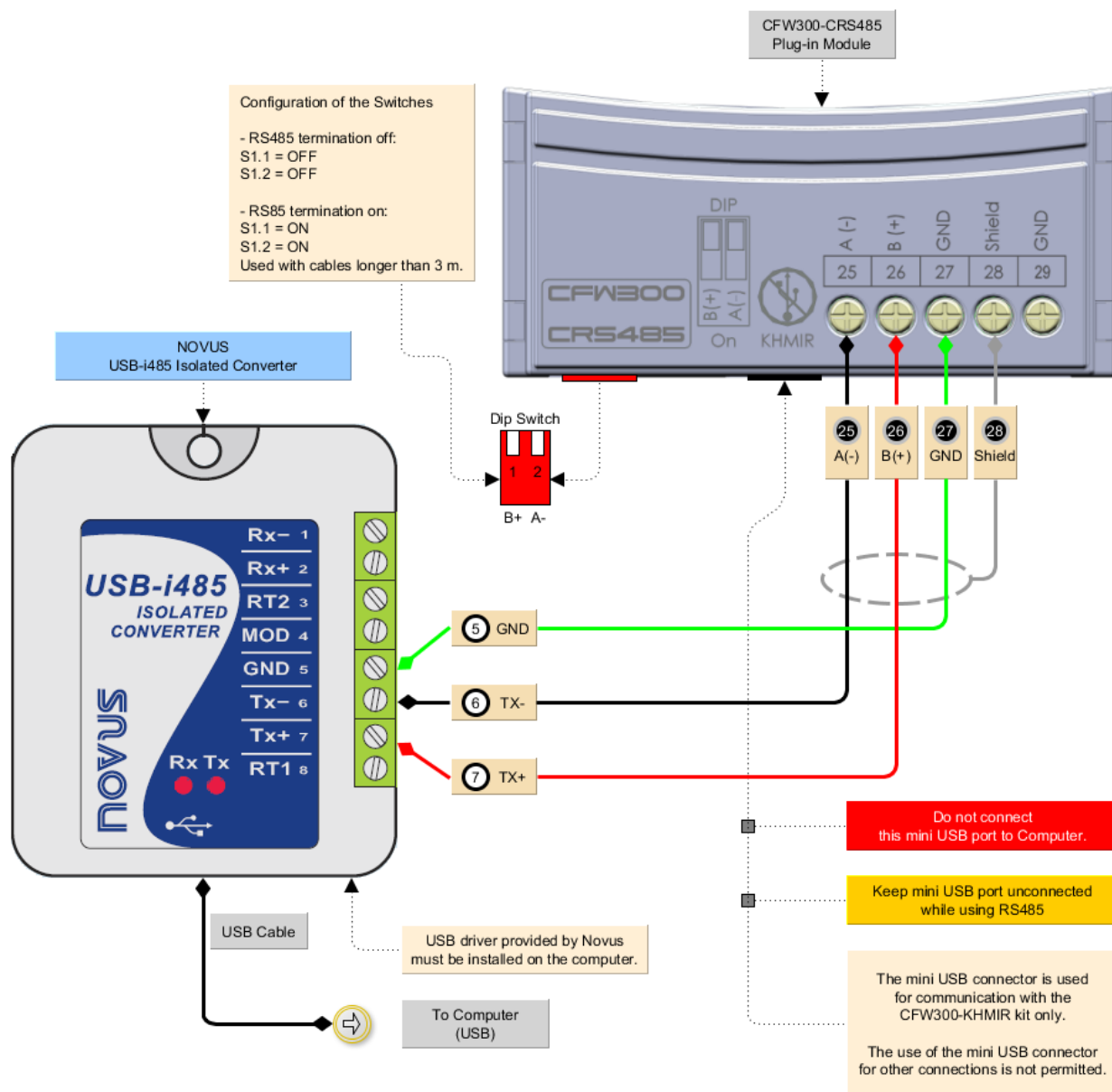
1) Conexão do Computador ao Equipamento

A figura abaixo mostra como conectar um computador ao equipamento via RS485.



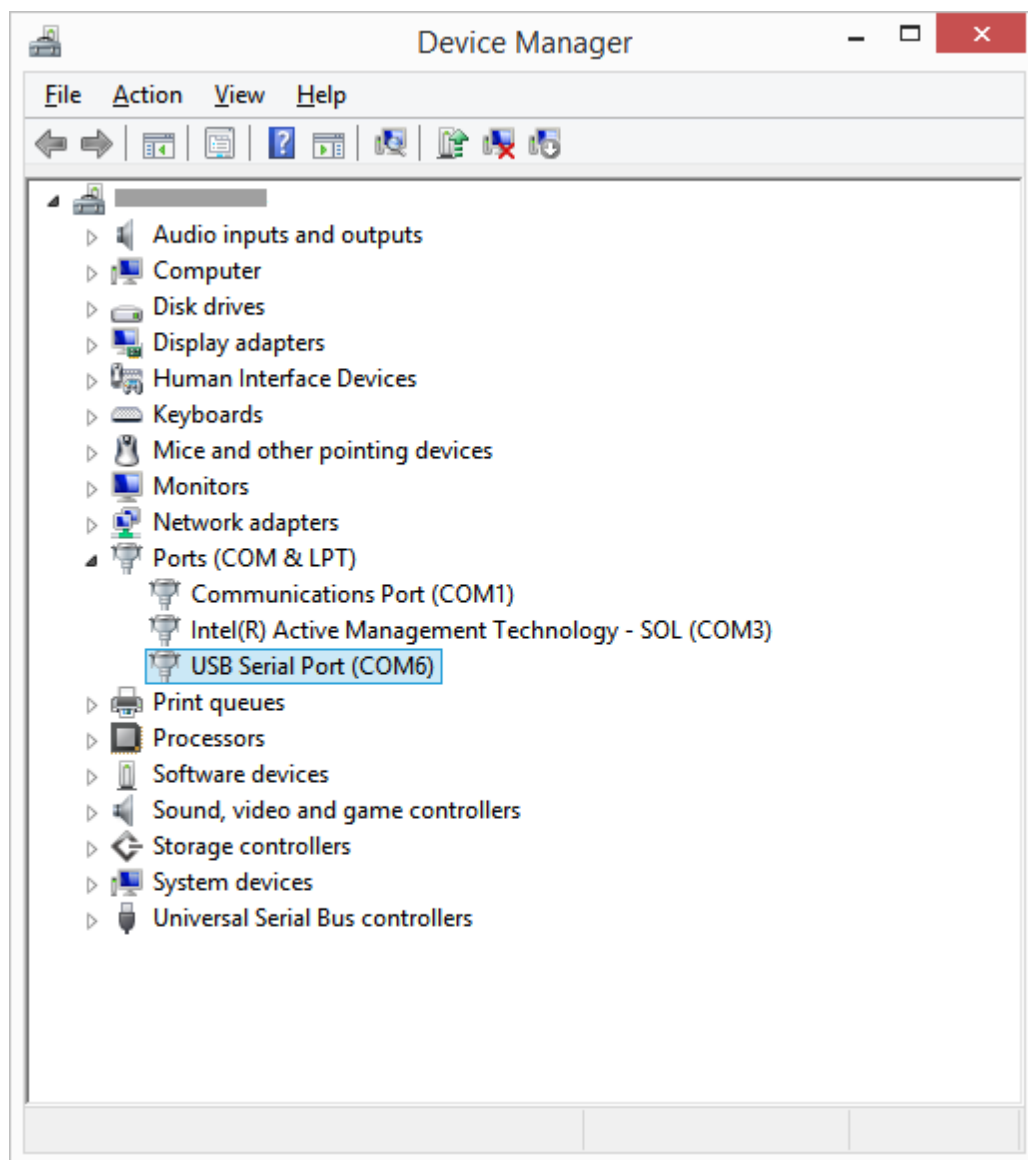
2) Conexão Computador - Módulo Plug-in

A figura abaixo mostra um exemplo utilizando o conversor Novus USB-i485.



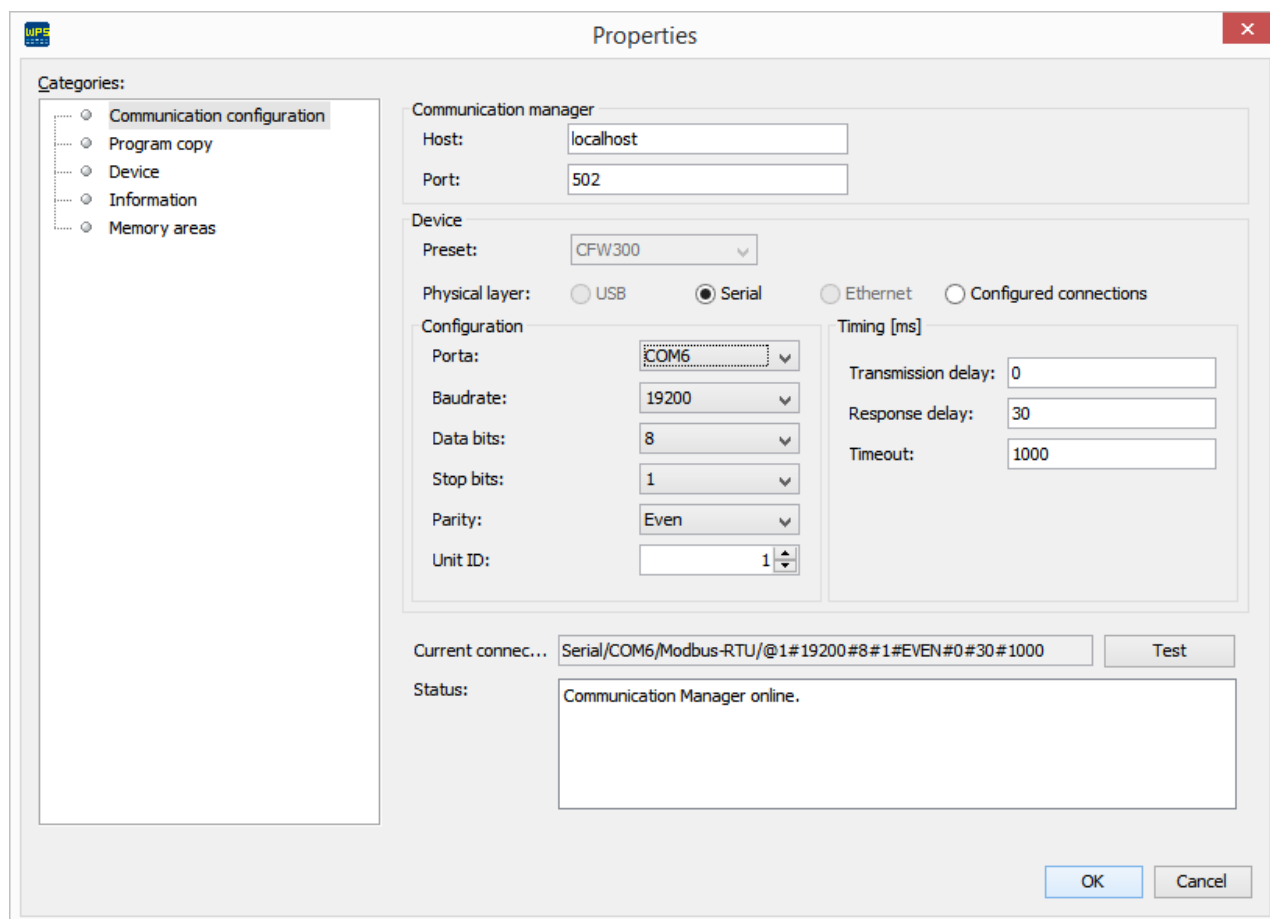
3) Gerenciador de Dispositivo do Windows

Gerenciador de dispositivo do Windows indica a porta serial conectada ao equipamento. O nome do computador no gerenciador de dispositivo está propositalmente ilegível.



4) Configuração da Comunicação no WPS

No WPS, selecione a porta serial corretamente na janela Propriedade > Configuração da Comunicação conforme a seguir.

**NOTA!**

Desligar o equipamento antes de fazer as conexões.

5) Conectando o Equipamento

1. Faça as ligações do conector do conversor isolado USB RS485 ao acessório CFW300-RS485 conforme apresentado nas figuras anteriores;
2. Insira o cabo USB no conector do conversor isolado USB RS85 e na porta USB do computador;
3. No gerenciador de dispositivo verifique qual porta serial está conectada;
4. Certifique-se que na categoria Configuração de Comunicação da janela Propriedades do recurso, a conexão serial esteja selecionada;
5. A porta serial e sua configuração do recurso no WPS deve ser a mesma porta serial que aparece no gerenciador de dispositivo do Windows, onde o cabo USB está conectado;
6. Nunca alterar valores nos parâmetros P308, P310, P311 e P312 durante uma conexão. A alteração destes parâmetros causa a perda instantânea de comunicação entre o PC e o equipamento.

7.5 Cabos

A figura abaixo apresenta o detalhamento do cabo para a conexão USB ponto-a-ponto.

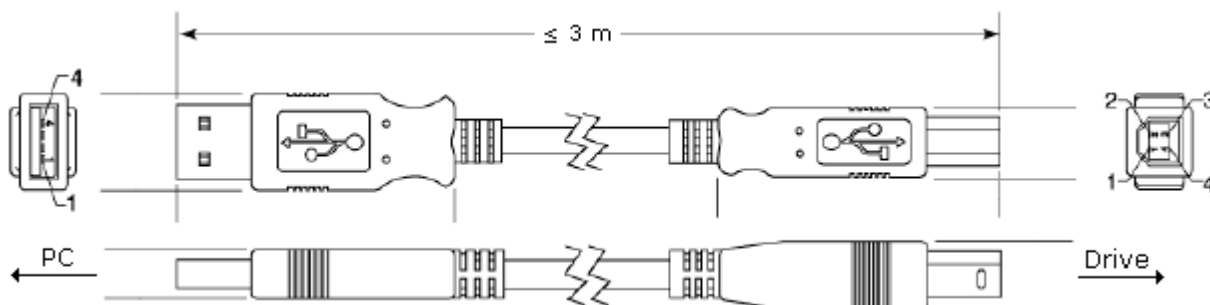


Figura 1 - Cabos USB

A figura abaixo apresenta os conectores.



Figura 2 - Conectores USB

A = Conector Tipo A

B = Conector Tipo B

Tamanho máximo do cabo: 3 metros.



NOTA!

Utilize sempre cabo de interconexão USB blindado padrão (Standard Host/Device Shielded USB Cable). Cabos sem blindagem podem provocar erros de comunicação.



NOTA!

A conexão USB é isolada galvânicamente da rede elétrica de alimentação e de outras tensões elevadas internas ao *drive*. A conexão USB, porém, não é isolada do terra de proteção (PE). Usar laptop isolado para ligação ao conector USB ou desktop com conexão ao mesmo terra de proteção (PE) do *drive*.

Sugestões de Compra

Fabricante:

- Samtec, Inc: <http://www.samtec.com/>

Se você desejar comprar um cabo USB diretamente da Samtec, por favor veja abaixo.

Descrição	Item
Cabo USB blindado de alta velocidade Revisão 2.0, 1 m, Samtec	USBC-AM-MB-B-B-S-1
Cabo USB blindado de alta velocidade Revisão 2.0, 2 m, Samtec	USBC-AM-MB-B-B-S-2
Cabo USB blindado de alta velocidade Revisão 2.0, 3 m, Samtec	USBC-AM-MB-B-B-S-3

No momento em que este manual foi escrito, a especificação era encontrada em <http://www.samtec.com/ftppub/cpdf/USBC-AM-BM-B-B-S-X-MKT.pdf>

7.6 Conversor USB/Serial

O conversor USB/SERIAL é a melhor solução para quem deseja conectar equipamentos seriais (RS232) em portas USB. É uma solução de baixo custo que resolve a necessidade de instalar novas portas seriais em microcomputadores que possuam todos os barramentos ocupados ou em equipamentos (*notebooks*) que não possuem portas RS232.

O conversor USB/SERIAL permite a conexão *plug & play* com seu microcomputador, deixando livre a porta serial existente.

Sugestões de Compra no Brasil

Fabricante: Comm5 Tecnologia

Produto: Conversor de USB para 1 saída serial RS232

Modelo: 1S-USB - conversor USB para 1 serial

Web Site: <http://www.comm5.com.br/1S-USB--CONVERSOR-USB-PARA-1-SERIAL>

Consulte o manual do usuário para mais informações sobre a instalação.



Fabricante: Tripp Lite

Produto: USB High Speed Serial Adapter

Modelo: USA-19HS

Web Site: <http://www.tripplite.com/en/products/model.cfm?txtSeriesID=849&txtModelID=3914>

Consulte o manual do usuário para mais informações sobre a instalação.



8 Ladder

[Conceitos](#)

[Editor](#)

[Trabalhando com USERFBs](#)

8.1 Conceitos

Ver a seguir.

8.1.1 Introdução

Programação Ladder é a representação gráfica de equações booleanas, combinando contatos (argumentos de entrada) com bobinas (resultados de saída).

O programa em Ladder possibilita testar e modificar dados por símbolos gráficos padrão. Estes símbolos são posicionados no diagrama Ladder de maneira semelhante a uma linha de um diagrama lógico com relés. O diagrama Ladder é delimitado na esquerda e na direita por linhas de barramento.

Componentes Gráficos

Os componentes gráficos básicos de um diagrama Ladder são mostrados abaixo.

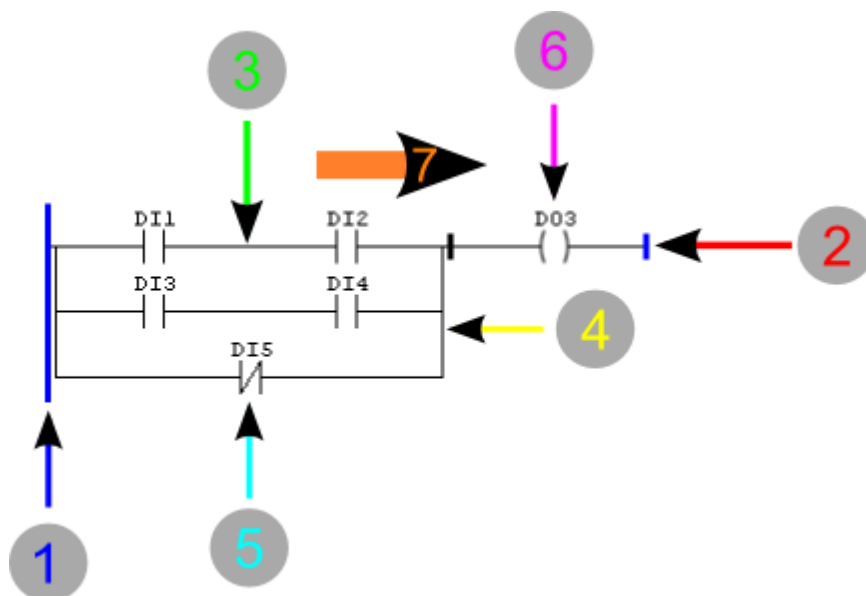


Figura 1 - Fluxo de Funcionamento do Ladder

1. Barramento esquerdo
2. Barramento direito
3. Ligação horizontal

4. Ligação vertical
5. Contato
6. Bobina
7. Sentido do fluxo de potência

Barramentos

O editor é delimitado na esquerda por uma linha vertical conhecida como barramento esquerdo, e na direita por uma linha vertical conhecida como barramento direito.

Elementos de Ligação e Estados

Os elementos de ligação podem ser horizontal ou vertical. O estado dos elementos de ligação podem ser denotados por 1 ou 0, correspondendo ao valor Booleano literal 1 (TRUE) ou 0 (FALSE), respectivamente. O termo estado da ligação tem que ser sinônimo do termo fluxo da potência.

O estado do barramento esquerdo pode ser considerado sempre 1 (fonte de energia), enquanto o barramento direito é definido como estado 0 (barramento de terra).

Um elemento de ligação horizontal é indicado por uma linha horizontal. Um elemento de ligação horizontal transmite o estado do elemento imediatamente a esquerda para o elemento imediatamente a direita.

Um elemento de ligação vertical consiste de linhas verticais intersectadas por uma ou mais ligações horizontais em cada lado.

O estado da ligação vertical representa o OU lógico dos estados das ligações horizontais no lado esquerdo. Isto é, o estado das ligações verticais deverá ser:

- 0, se o estado de todas as ligações horizontais incluídas na sua esquerda são 0,
- 1, se o estado de uma ou mais ligações horizontais incluídas na sua esquerda são 1.

O estado das ligações verticais são copiados para todas as ligações horizontais associados à sua direita. O estado das ligações verticais não é copiado para as ligações horizontais associadas à sua esquerda.

Controle de Execução

A figura a seguir mostra como o programa em Ladder é executado. O processador executa continuamente um ciclo de varredura. O ciclo começa com o sistema de entradas e saídas do *hardware*, compilando os últimos valores de todos os sinais de entrada e gravando seus valores em regiões fixas da memória.

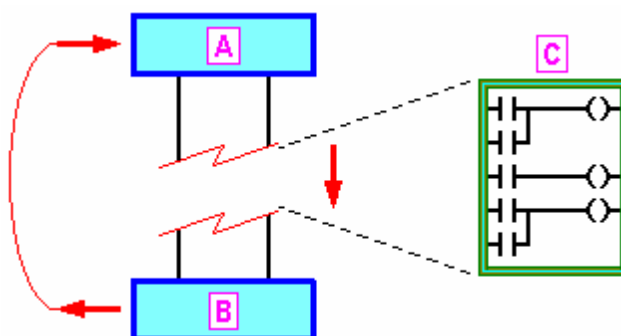


Figura 2 - Controle de Execução

A - Entradas lidas para a memória
 B - Memória escrita nas Saídas
 C - Varredura das linhas do Ladder

As linhas do programa Ladder são então executados num ordem fixa, iniciando com a primeira linha. Durante a varredura do programa, novos valores das saídas físicas, como determinadas a partir da lógica das várias linhas do Ladder, são inicialmente inscritos numa região da memória de saída. Finalmente, quando o programa Ladder terminou a execução, todos os valores de saída retidos na memória são inscritos nas saídas físicas pelo *hardware* numa única operação.

8.1.2 Lógica com Contatos

LÓGICA AND - Contatos em Série

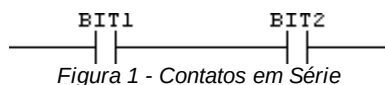


Figura 1 - Contatos em Série

A figura acima executa uma Lógica AND entre os dois últimos elementos carregados no [STACK](#), baixa um nível do [STACK](#) e joga o resultado para o topo do STACK. Isto significa que é executada a seguinte operação Booleana: topo do [STACK](#) = BIT1.BIT2.

Em linguagem IL (Lista de Instruções) fica:

```
LD    BIT1    (* carrega o valor da variável BIT1 para o STACK = | BIT1 | *)
LD    BIT2    (* carrega o valor da variável BIT2 para o STACK = | BIT2 | BIT1 | *)
AND                    (* Executa a Lógica AND entre o BIT1 e BIT2 através do STACK = | BIT1.BIT2 | *)
```

Tabela Verdade

BIT1	BIT2	STACK
0	0	0
0	1	0
1	0	0
1	1	1

LÓGICA OR - Contatos em Paralelo

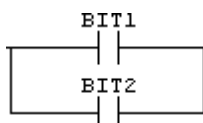


Figura 2 - Contatos em Paralelo

A figura acima executa uma Lógica OR entre os dois últimos elementos carregados no [STACK](#), baixa um nível do [STACK](#) e joga o resultado para o topo do [STACK](#). Isto significa que é executada a seguinte operação Booleana: topo do [STACK](#) = BIT1 + BIT2.

Em linguagem IL (Lista de Instruções) fica:

```
LD    BIT1    (* carrega o valor da variável BIT1 para o STACK = | BIT1 | *)
LD    BIT2    (* carrega o valor da variável BIT2 para o STACK = | BIT 2 | BIT1 | *)
OR                      (* Executa a Lógica OR entre o BIT1 e BIT2 através do STACK = | BIT1+BIT2| *)
```

Tabela Verdade

BIT1	BIT2	STACK
0	0	0
0	1	1
1	0	1
1	1	1

8.1.3 Terminologia

AT: Representação direta de uma variável.

BLOCO FUNCIONAL (FUNCTION BLOCK): Consiste numa Função que necessita de uma Instância.

CONFIGURAÇÃO (CONFIGURATION): Consiste na organização de um *software* em um nível mais elevado. Pode conter vários Recursos dentro de uma Configuração.

FUNÇÃO (FUNCTION): É um bloco responsável por executar um determinado comportamento ou uma ação, baseado em possíveis parâmetros (VAR_IN, VAR_IN_OUT, VAR_OUT).

INSTÂNCIA (INSTANCE): Área de memória ocupada para uso do Bloco Funcional.

LD (Ladder Diagram): Linguagem gráfica de programação baseada nos diagramas elétricos (contatos e bobinas interconectadas), conforme o fluxo de potência entre os elementos.

OVERFLOW: Ocorre quando o resultado de um cálculo matemático ultrapassa os limites permitidos para o tipo de dado do resultado.

POU: Unidade de Organização de Programas. Pode ser: Programa, Bloco Funcional ou Função.

PROGRAMA (PROGRAM): É um agrupamento lógico de todos os elementos de programação e construções

necessários ao processamento de sinais requeridos para o controle de uma máquina ou um processo..

RECURSO (RESOURCE): Consistem em qualquer elemento com capacidade de processamento, responsável pela execução dos programas.

SCAN: Ciclo de varredura de um programa.

STACK: Pilha do programa Ladder. É a área de memória utilizada para realizar as Lógicas dos Programas.

TAG: Nome da Variável.

TAREFA (TASK): Responsável pelo controle de execução de programas, de forma periódica ou engatilhada, com disparo por eventos.

TIPO DE DADO (DATATYPE): Informa ao compilador o espaço ocupado por uma variável e seu respectivo formato (aos Blocos).

VARIÁVEL: Consiste numa posição da memória capaz de reter e representar um valor ou expressão. Pode ter escopo:

- Local: cuja posição automática, é calculada pelo compilador.
- Global: localizada numa área de memória determinada, como entradas e saídas digitais, podendo ser acessada em qualquer ponto da Configuração.

A Variável pode ser:

- Retentiva: armazena o valor após a desenergização do equipamento;
- Volátil: inicia com o valor contido no campo de valor inicial, após a energização do equipamento;
- Constante: não permite alteração do seu conteúdo.

VAR: Variáveis de uso interno de um Bloco do Usuário (USERFB). Equivalente a uma variável de escopo Local.

VAR_INPUT: Argumento de entrada de um Bloco do Usuário (USERFB). As variáveis configuradas neste campo, apenas serão lidas no USERFB.

VAR_IN_OUT: Argumento de entrada e de saída de um Bloco do Usuário (USERFB). Não permite variáveis CONSTANTES, ENTRADAS FÍSICAS (%I_), ENTRADAS DE REDE (%I_) ou MARCADORES DE SISTEMA DE LEITURA (%S_), pois elas serão lidas no início do USERFB e serão atualizadas com novos valores (escritas) no final do USERFB.

VAR_OUTPUT: Argumento de saída de um Bloco do Usuário (USERFB). Não permite variáveis CONSTANTES, ENTRADAS FÍSICAS (%I_), ENTRADAS DE REDE (%I_) ou MARCADORES DE SISTEMA DE LEITURA (%S_), pois elas serão atualizadas com novos valores (escritas) no final do USERFB.

WATCHDOG: É um meio proporcionado pelo fabricante para desempenhar ações específicas, se a integridade do sistema for violado.

8.1.4 Tipo de Dados

Tipo de dado	Tamanho	Sinal	Faixa de valores	Precisão
BOOL	1 bit		0 ou 1	
BYTE	8 bits (1 byte)		0 a 255	
USINT	8 bits (1 byte)		0 a 255	
SINT	8 bits (1 byte)	Sim	-128 a 127	
WORD	16 bits (2 bytes)		0 a 65535	
UINT	16 bits (2 bytes)		0 a 65535	
INT	16 bits (2 bytes)	Sim	-32768 a 32767	
DWORD	32 bits (4 bytes)		0 a ($2^{32} - 1$)	
UDINT	32 bits (4 bytes)		0 a ($2^{32} - 1$)	
DINT	32 bits (4 bytes)	Sim	- 2^{31} a ($2^{31} - 1$)	
LWORD	64 bits (8 bytes)		0 a ($2^{64} - 1$)	
ULINT	64 bits (8 bytes)		0 a ($2^{64} - 1$)	
LINT	64 bits (8 bytes)	Sim	- 2^{63} a ($2^{63} - 1$)	
REAL	32 bits (4 bytes) Ponto flutuante padrão IEEE 559	Sim	$\pm 10^{\pm 38}$; Precisão = 2^{-23}	$1 / 2^{23}$
LREAL	64 bits (8 bytes) Ponto flutuante padrão IEEE 559	Sim	$\pm 10^{\pm 308}$; Precisão = 2^{-52}	$1 / 2^{52}$
STRING	8 bits (1 byte) por posição + 8 bits (1 byte) para o caractere nulo de terminação		1 a 254 caracteres ASCII	



NOTA!

O tipo de dado STRING considera apenas o número de caracteres na configuração do tamanho da variável. O byte ocupado pelo terminador nulo não é adicionado ao tamanho.

8.1.5 Representação Direta

Utilizado para definir a posição de memória de uma variável global.

Sintaxe: %<Formato><Tamanho>

Primeira Letra:

At (Formato)	Descrição
I	Entradas (Input): Recebe valores das variáveis analógicas e discretas, ou variáveis de rede de entrada
Q	Saídas (Output): Armazena valores a serem escritos nas saídas analógicas e discretas, ou variáveis de rede de saída
M	Marcador interno de memória RAM
S	Marcador de status de sistema
C	Marcador de comando de sistema

Segunda Letra:

At (Tamanho)	Descrição
X	Bit
B	Byte (8 bits)
W	Word (16 bits)
D	Double Word (32 bits)
L	Long Word (64 bits)

Exemplos:

%IX: entrada digital, com tamanho de 1 bit.

%MW: marcador de WORD, com tamanho de 16 bits.

%SB: marcador de status de sistema de BYTE, com tamanho de 8 bits.

8.2 Editor

Ver a seguir.

8.2.1 Área de Trabalho

Sempre que uma arquivo Ladder for aberto através do seu atalho na árvore de projetos, o **Editor Ladder** aparecerá, conforme figura a seguir.

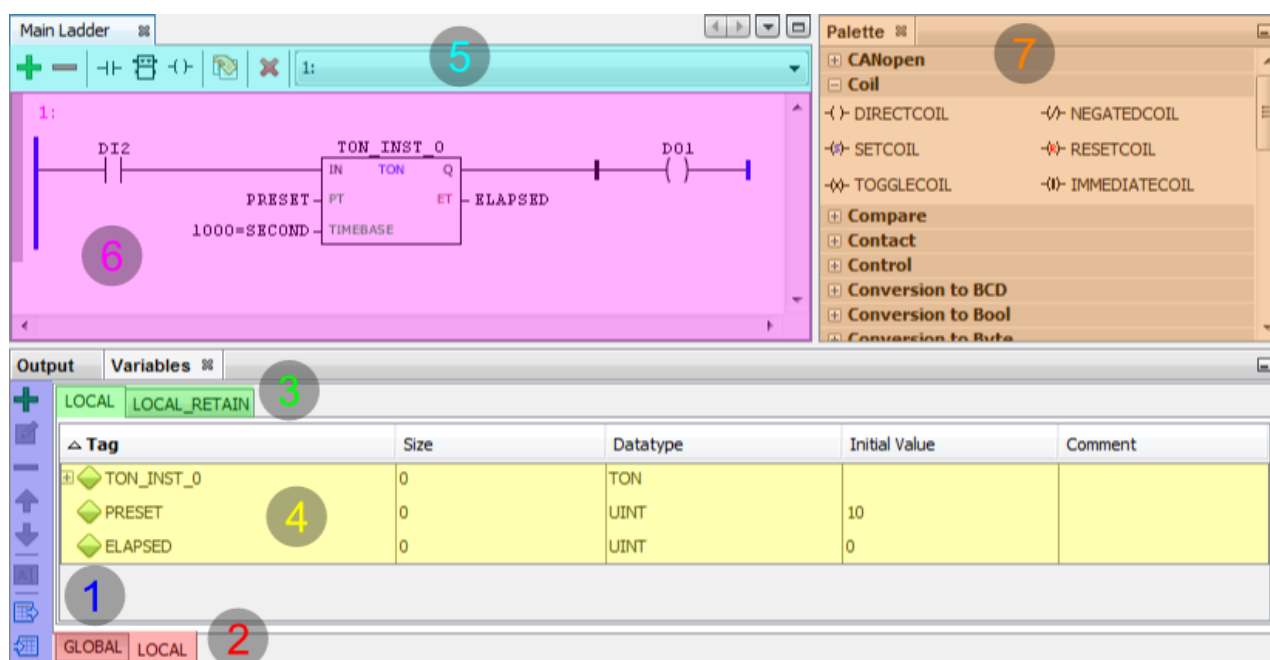


Figura 1 - Área de Trabalho do Editor Ladder do WPS

O editor Ladder possui os seguintes componentes:

1. Barra de ferramentas para edição de variáveis
2. Escopo das variáveis
3. Grupo de variáveis
4. Editor/lista de variáveis
5. Barra de ferramentas de edição de programas
6. Editor Ladder
7. Paleta de componentes Ladder



NOTA!

Caso algum item não esteja visível, ative-o pelo menu Janela.

8.2.2 Menu Ladder

Ao editar um arquivo Ladder do seu recurso, o menu **Ladder** estará ativo, conforme figura a seguir.

Ladder	Communication	Tools	Window	Help
Insert new rung at the end		Ctrl+Alt+N		
Insert new rung below		Ctrl+Alt+B		
Insert new rung above		Ctrl+Alt+A		
Go to rung below		Ctrl+Down		
Go to rung above		Ctrl+Up		
Delete selected rung		Ctrl+Delete		
Select rung below		Shift+Down		
Select rung above		Shift+Up		
Select next element		Right		
Select previous item		Left		
Select element below		Down		
Select item above		Up		
Insert contact		C		
Insert block (Function/Functional)		F		
Insert output		O		
Change ladder element		X		
Select group of elements		G		
Select all elements		Ctrl+A		
Delete element				
Edit arguments		F2		
Edit rung title		Shift+F2		
Replace...				
Cross-reference				
 Force I/O				

Através desse menu é possível fazer todas as operações relativas ao Ladder, bem como conhecer os atalhos de teclado para essas operações.

8.2.3 Rungs

Ver a seguir.

8.2.3.1 Visão Geral

O programa Ladder é editado através de um editor gráfico que organiza o mesmo através de *rungs*. Cada *rung* do programa Ladder corresponde a um intertravamento que relaciona elementos de entrada a elementos de saída. Esse editor é mostrado conforme figura a seguir.

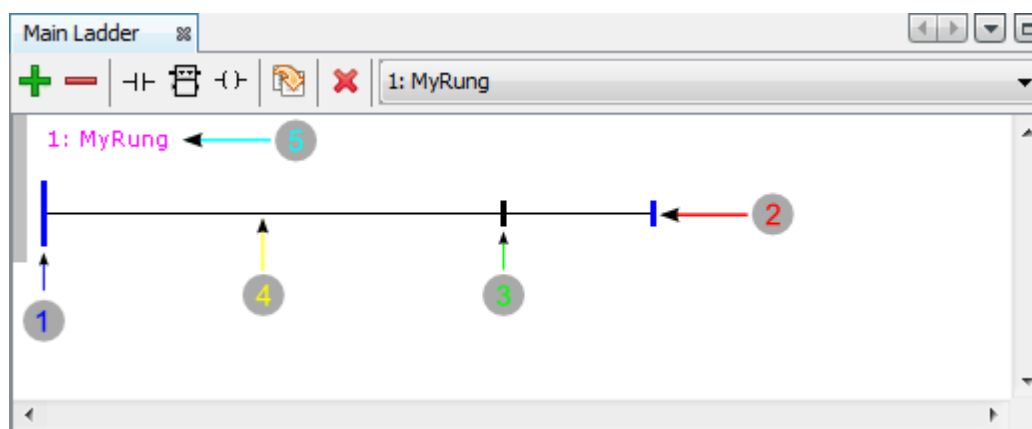


Figura 1 - Elementos da Rung

Os elementos principais de uma *rung* são:

1. Barramento esquerdo
2. Barramento direito
3. Barramento de saídas
4. Ligações entre elementos
5. Título e comentário da *rung*



NOTA!

As saídas sempre estarão conectadas à direita do barramento de saídas.

Para maiores detalhes da programação Ladder, [clique aqui](#).

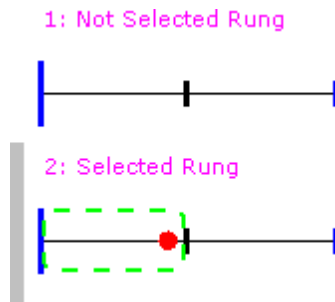
Através da barra de ferramentas do **Editor Ladder** é possível executar as seguintes operações:

-  - Adicionar uma nova *rung*
-  - Remover a *rung* selecionada
-  - Inserir um contato na *rung*
-  - Inserir um bloco funcional na *rung*
-  - Inserir um bobina de saída na *rung*
-  - Troca elemento Ladder
-  - Apagar um elemento da *rung* (contato, bloco funcional ou bobina)
-  - Selecionar a *rung* para edição

8.2.3.2 Editando

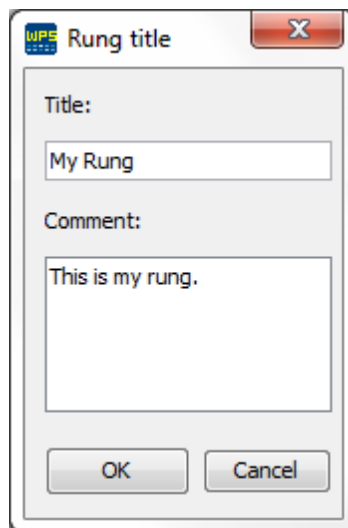
Para editar uma *rung*, ela deve estar selecionada. Para fazer a seleção da *rung*, você pode clicar com o

mouse sobre ela ou utilizar o controle de seleção de *rung* na barra de ferramentas do **Editor Ladder**. Sempre que a *rung* estiver selecionada, existirá uma barra cinza à sua esquerda, conforme figura a seguir.

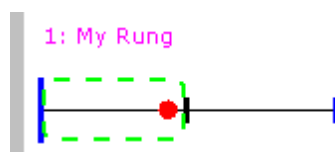


8.2.3.3 Título e Comentário

Para editar o título e comentário da *rung*, basta dar um duplo clique do mouse sobre a região do título e comentário ou utilizar o atalho do teclado (**Shift+F2**). O seguinte diálogo aparecerá.



Após editar o título e comentário, o editor ficará da seguinte maneira.



8.2.3.4 Inserindo Elementos

Ver a seguir.

8.2.3.4.1 Visão Geral

Para inserir um elemento Ladder na *rung* é necessário que um elemento já existente do Ladder esteja selecionado, conforme figura a seguir.



O elemento selecionado estará envolvido por um retângulo verde tracejado. Existirá também um ponto vermelho indicando onde o novo elemento será inserido, chamado de **ponto de inserção**.

A inserção de elementos Ladder pode ser feita através de três maneiras distintas.

1. Pelo teclado, através dos seguintes atalhos

tecla **C** - inserir contato

tecla **F** - inserir bloco funcional

tecla **O** - inserir bobina de saída

2. Pela barra de ferramentas do editor Ladder



- Inserir um contato na *rung*

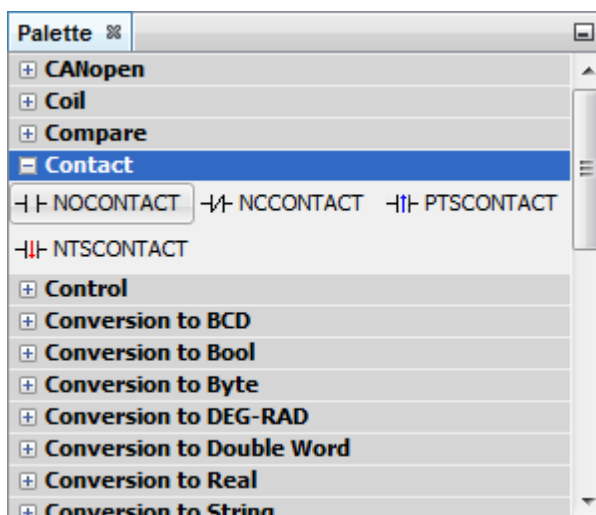


- Inserir um bloco funcional na *rung*



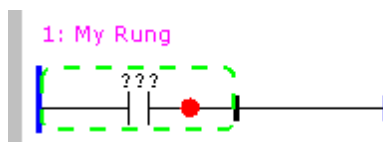
- Inserir um bobina de saída na *rung*

3. Pela paleta de elementos Ladder



Para inserir elementos pela paleta, deve-se utilizar a operação arrastar e soltar do mouse. Para isso, clique sobre o elemento da paleta, mantenha o mouse pressionado, mova o mouse até o ponto de inserção no Ladder e solte o botão do mouse.

Após inserir um elemento (neste caso, um contato), a *rung* ficará da seguinte maneira.



8.2.3.4.2 Em Série

Através do ponto de inserção dos elementos, é possível inserir um elemento em série. Para isso, utilize os seguintes pontos de inserção:

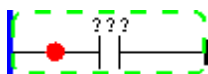


Figura 1 - Insere elemento em série antes do elemento selecionado



Figura 2 - Insere elemento em série depois do elemento selecionado

Inserir elementos em série corresponde à lógica booleana AND.

8.2.3.4.3 Em Paralelo

Através do ponto de inserção dos elementos, é possível inserir um elemento em paralelo. Para isso, utilize os seguintes pontos de inserção:



Figura 1 - Insere elemento em paralelo acima do elemento selecionado



Figura 2 - Insere elemento em paralelo abaixo do elemento selecionado

Inserir elementos em paralelo corresponde à lógica booleana OR.

8.2.3.5 Navegando

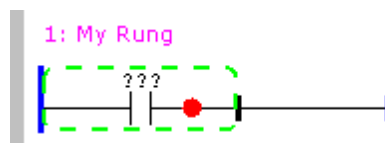
Ver a seguir.

8.2.3.5.1 Com o Teclado

A navegação pelo teclado na *rung* é feito pelas teclas ←, →, ↑, ↓. Através dessas teclas, é possível selecionar os diversos elementos que estão dentro da *rung* e também definir o ponto de inserção dos novos

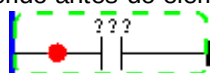
elementos.

Conforme mostrado no item anterior, ao [inserir um elemento](#) do tipo contato na *rung*, esta ficará da seguinte maneira:



Ao pressionar as teclas de navegação podemos modificar o ponto de inserção de um novo elemento, conforme figuras a seguir:

Tecla ← indica que o novo elemento será inserido antes do elemento selecionado.



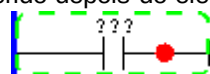
Tecla ↑ indica que o novo elemento será inserido em paralelo acima do elemento selecionado.



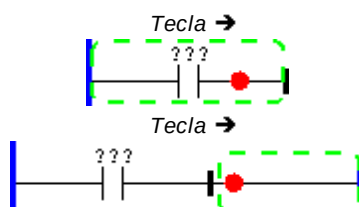
Tecla ↓ indica que o novo elemento será inserido em paralelo abaixo do elemento selecionado.



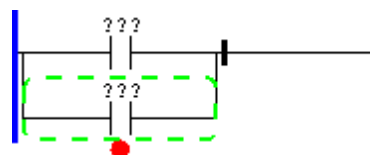
Tecla → indica que o novo elemento será inserido depois do elemento selecionado.



Estando o ponto de inserção já selecionado e ao clicar a mesma tecla da direção correspondente a seleção passará para o próximo elemento, conforme figura a seguir.



Quanto a elementos em paralelo, navegamos com o teclado por cada elemento individualmente, conforme exemplo a seguir.

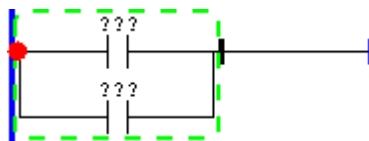


Nessa situação, ao inserirmos um outro elemento, estaremos fazendo essa inserção relacionada a esse único elemento selecionado.

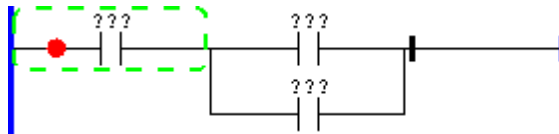
Caso seja necessário inserir um elemento em paralelo a um **grupo** de elementos, devemos utilizar a tecla **G**,

que selecionará o grupo de elementos, e então poderemos definir o ponto de inserção e fazê-la.

Pressionando a tecla **G**:



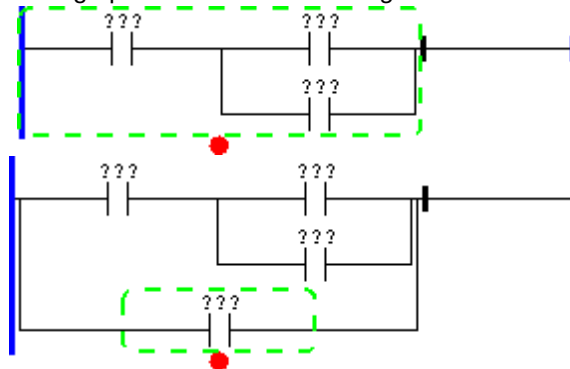
Inserido contato antes do paralelo:



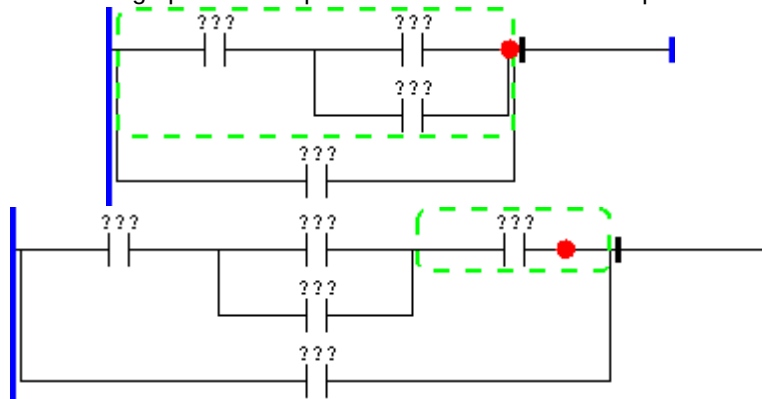
Ao cada toque da tecla **G** selecionamos o grupo imediatamente superior ao elemento/grupo previamente selecionado e, ao final, voltamos para o elemento original.

Sendo assim, em algumas situações, faz-se necessário mais de um toque da tecla **G**. Através da tecla **G** é possível fazer diversas operações de inserção, conforme exemplos a seguir.

Inserir um elemento em paralelo ao grupo de elementos da *rung*:



Inserir um elemento em serie ao grupo formado por elementos internos a um paralelo:



8.2.3.5.2 Com o Mouse

Todas as funcionalidades de navegação pelo teclado na *rung* também estão disponíveis no mouse.

Diretamente pelo clique do mouse é possível selecionar um elemento e o respectivo ponto de inserção.

Cursor	Ação	Representação
	Seleciona elemento e ponto de inserção superior	
	Seleciona elemento e ponto de inserção inferior	
	Seleciona elemento e ponto de inserção anterior	
	Seleciona elemento e ponto de inserção posterior	

Também é possível definir o ponto de inserção durante a operação de inserção de um novo elemento via mouse, através do arrastar e soltar do mouse da paleta.

À medida que arrastamos os elementos em cima da *rung*, a seleção e o ponto de inserção acompanham o cursor do mouse, de modo a determinar o local onde o novo bloco será inserido. A figura a seguir exemplifica a inserção de elementos via mouse.

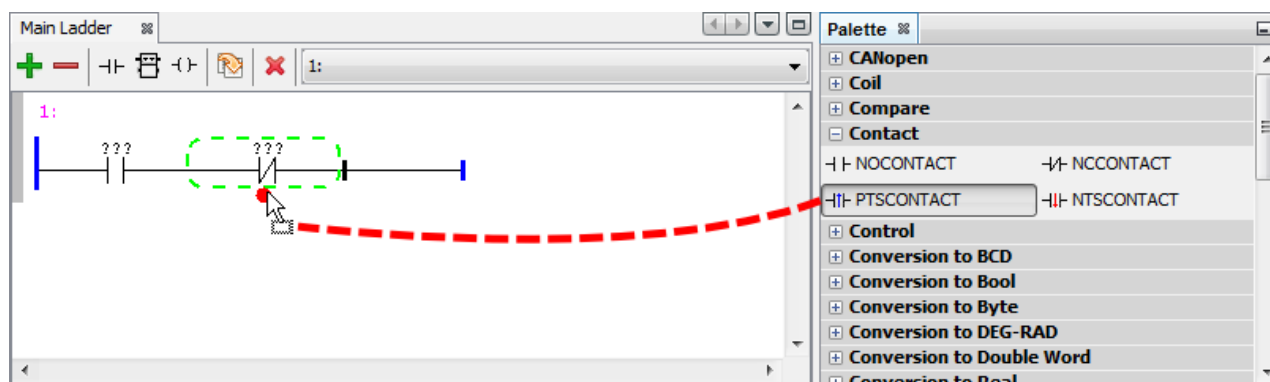


Figura 1 - Operação de arrastar componente com o Mouse

Sequência de inserção de elementos via mouse:

1. Selecione o elemento da paleta que deseja inserir, clicando com o mouse sobre o mesmo e mantendo o botão do mouse pressionado
2. Arraste esse elemento sobre a *rung*
3. Selecione o ponto de inserção através do movimento do cursor e solte o botão do mouse sobre ele

O resultado da inserção nesse exemplo será o seguinte.



Figura 2 - Resultado após a operação de arraste

Durante a inserção via mouse também é possível selecionar um grupo de elementos, de maneira análoga à tecla **G**. Para isso, durante o arraste do elemento previamente selecionado, pressione e mantenha pressionada a tecla **Ctrl**. Enquanto a tecla **Ctrl** estiver pressionada, o mouse irá selecionar pontos de

inserção relativos a grupos.

8.2.3.6 Copiar/Colar

Visão Geral

Todas as funcionalidades de copiar, recortar e colar estão disponíveis no editor Ladder através do menu **Editar** ou dos atalhos de teclado correspondentes.

Para executar essas operações, é necessário ter uma *rung* ou um elemento selecionado.

Copiar/colar um elemento

A seguir um exemplo de copiar/colar de um elemento.

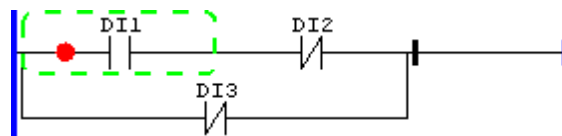


Figura 1 - Antes, elemento selecionado e pressionado Ctrl+C

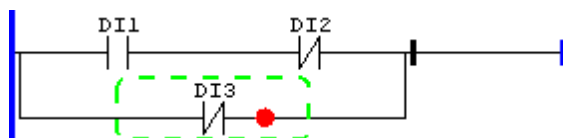


Figura 2 - Selecionado o ponto de inserção de destino

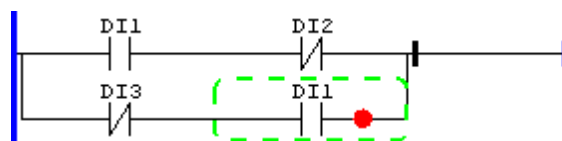


Figura 3 - Depois de pressionado Ctrl+V

Copiar/colar de múltiplos elementos

Também é possível copiar, recortar e colar múltiplos elementos selecionados ou pelo mouse ou pela tecla **G**. A seguir, um exemplo de copiar/colar de múltiplos elementos.

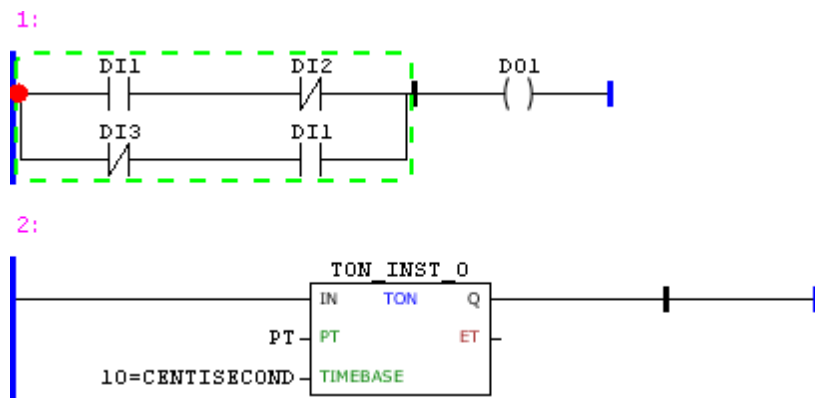


Figura 4 - Antes, grupo selecionado e pressionado Ctrl+C

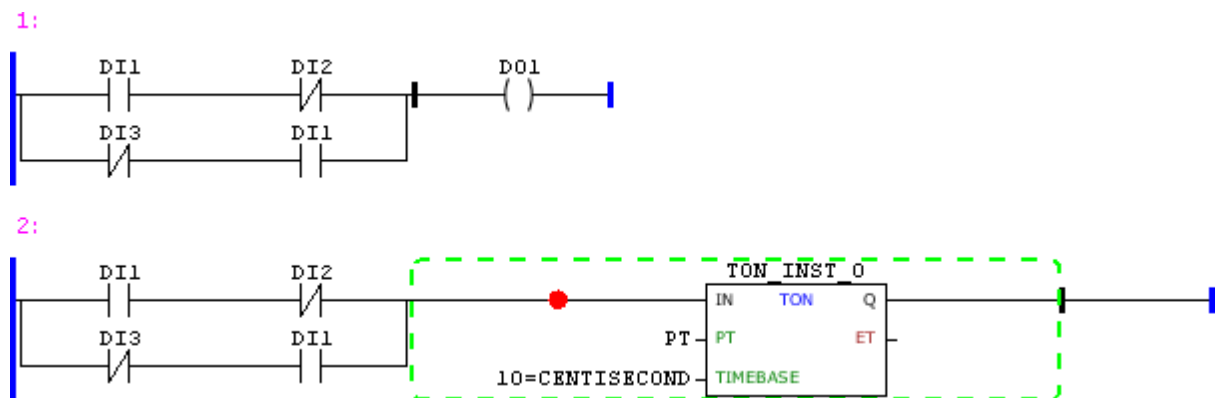


Figura 5 - Depois, temporizador selecionado com ponto de inserção anterior e pressionado Ctrl+V

Poderá haver situações que o colar não será possível, não havendo alteração na *rung* após o comando.

Copiar/colar de rungs

A seleção de *rungs* para as operação de copiar, recortar e colar é feita somente através do mouse conforme figuras a seguir.

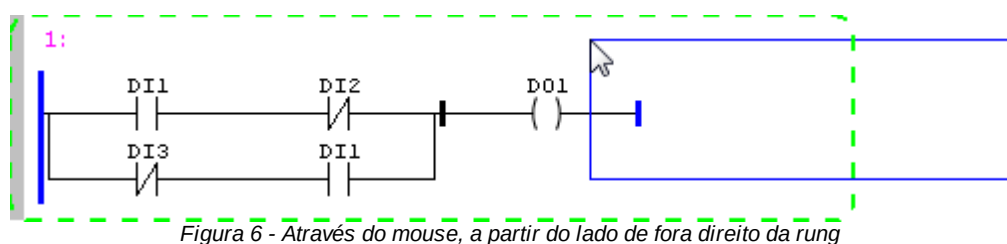


Figura 6 - Através do mouse, a partir do lado de fora direito da rung

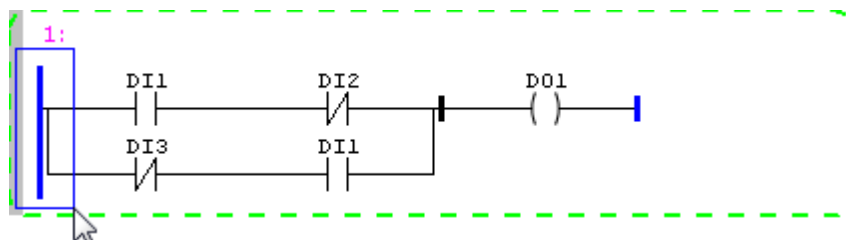


Figura 7 - Através do mouse, a partir do lado esquerdo da rung

Também é possível selecionar múltiplas *rungs* para essas operações.
A seguir, um exemplo de copiar/colar de *rung*.

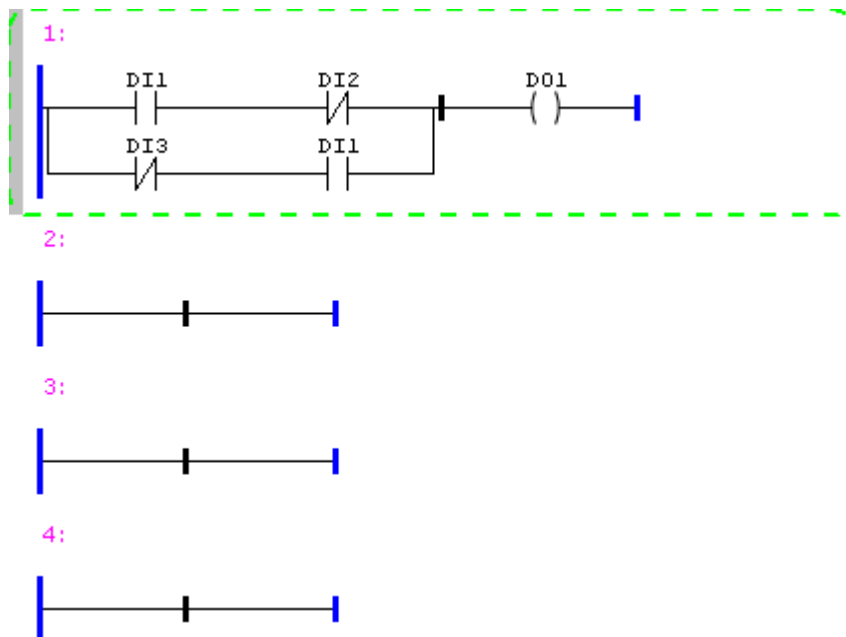


Figura 8 - Rung a ser copiada selecionada e pressionado Ctrl+C

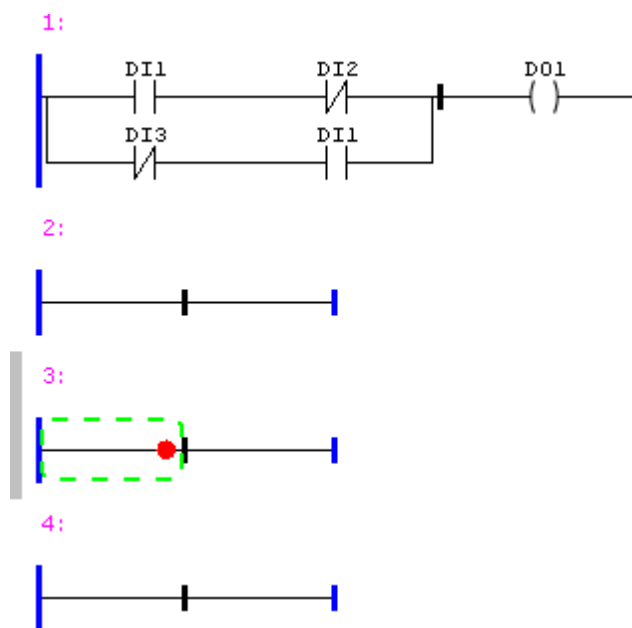


Figura 9 - Rung imediatamente superior à colagem selecionada

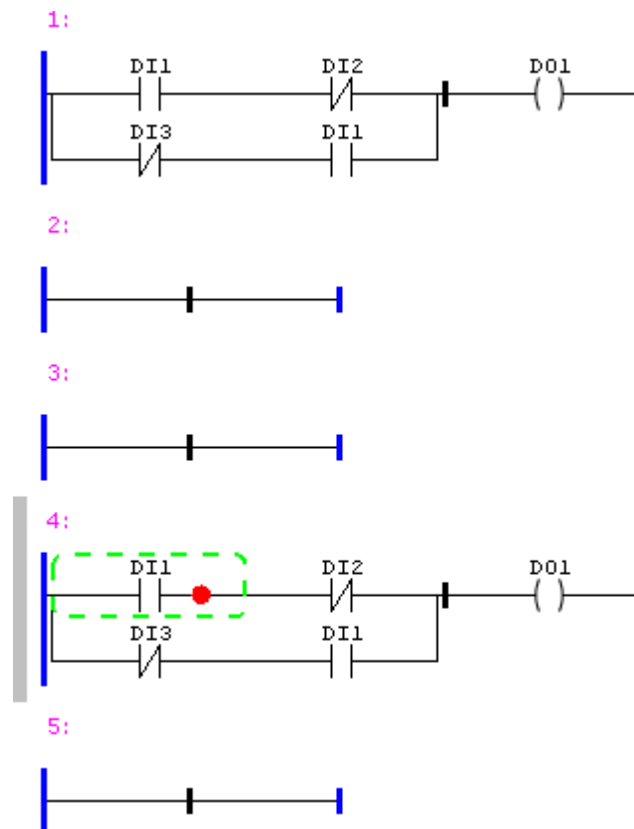


Figura 10 - Depois de pressionado Ctrl+V

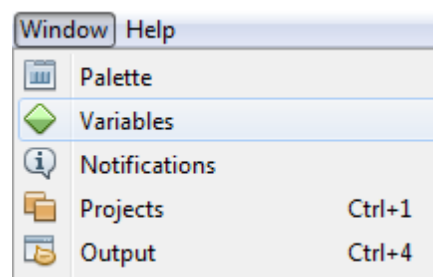
8.2.4 Variáveis

Veja a seguir.

8.2.4.1 Visão Geral

Durante a edição do Ladder, é necessário definir as variáveis utilizadas nos componentes Ladder. Essa definição pode ser feita diretamente no Ladder ou na lista de variáveis.

A tabela de variáveis deve ser ativada através do menu **Janela > Variáveis**, conforme figura a seguir.



Os seguintes comandos estão disponíveis na barra de ferramentas de edição de variáveis:

-  - Adicionar uma nova variável
-  - Edita a variável selecionada
-  - Remover a variável selecionada
-  - Move a variável selecionada uma linha acima na tabela
-  - Move a variável selecionada uma linha abaixo na tabela
-  - Renomeia a variável selecionada
-  - Exporta variáveis deste grupo para um arquivo
-  - Importa variáveis para este grupo de um arquivo

As variáveis do escopo (1) **GLOBAL** apresentam os grupos (2), conforme figura 1.

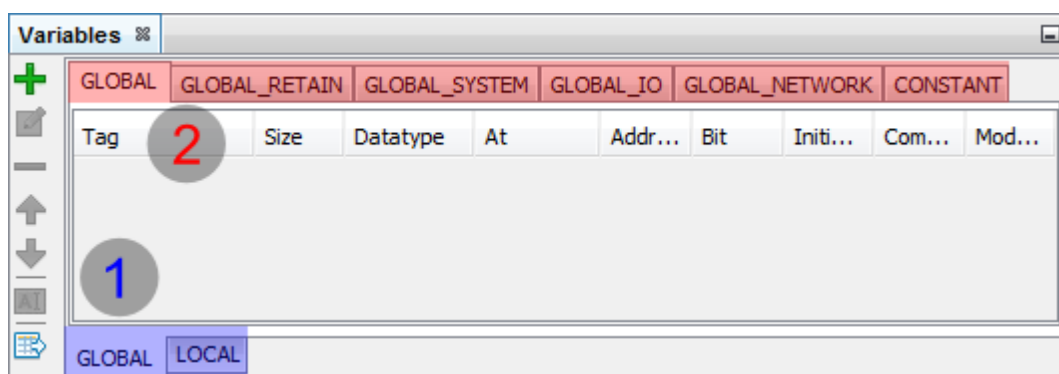


Figura 1 - Editor de Variáveis para o Escopo GLOBAL

Nesta área de variáveis globais, estão disponíveis os seguintes grupos:

- **Constante:** variáveis que armazenam valores constantes
- **Global:** variáveis acessíveis por todos os arquivos Ladder e por todas as telas de HMI
- **Global Retentivo:** semelhante ao grupo global mais com memória retentiva*
- **Global de Sistema:** variáveis previamente definidas com funções do sistema
- **Global de I/O:** entrada e saídas físicas do equipamento
- **Global de Rede:** variáveis previamente definidas para comunicação em rede

As variáveis do escopo (1) **LOCAL** apresentam os grupos (2), conforme figura 2.

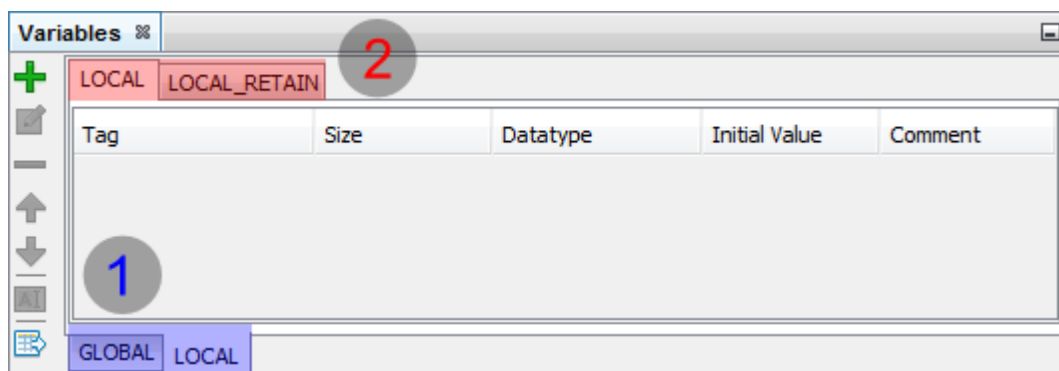


Figura 2 - Editor de Variáveis para o Escopo LOCAL

Nesta área de variáveis locais, estão disponíveis os seguintes grupos:

- **Local:** variáveis acessíveis somente pelo Ladder que está sendo editado
- **Local Retentivo:** semelhante ao grupo local mais com memória retentiva

**NOTA!**

Uma memória retentiva mantém seu valor mesmo com o equipamento desligado.

8.2.4.2 Campos

Ao definir uma variável através da lista de variáveis, alguns dados deverão ser definidos para as variáveis. Nos itens a seguir serão apresentados esses dados de acordo com o grupo que a variável pertence.

Local e Local Retentivo:

Tag	Size	Datatype	Initial Value	Comment
 LOCAL_VAR	0	BOOL	1	Comment

- **Tag:** identificação da variável
- **Tamanho:** número de elementos do array relacionado à variável
- **Tipo de Dado:** tipo numérico da variável
- **Valor Inicial:** valor que será carregado para variável durante a inicialização do equipamento
- **Comentário:** comentário da variável no idioma selecionado

Constante:

Tag	Datatype	Initial Value	Comment
 CONST	REAL	7.89	Comment

- **Tag:** identificação da variável
- **Tipo de Dado:** tipo numérico da variável
- **Valor:** valor constante da variável
- **Comentário:** comentário da variável no idioma selecionado

Global e Global Retentivo:

△ Tag	Size	Datatype	At	Address	Bit	Initial Value	Comment	Modbus
 GLOBAL_VAR	0	BYTE	%MB	100		12	Comment	8050

- **Tag:** identificação da variável
- **Tamanho:** número de elementos do array relacionado à variável
- **Tipo de Dado:** tipo numérico da variável
- **At:** define qual área de memória global a variável acessa
- **Endereço:** endereço relacionado a área de memória global. Caso não seja configurado (vazio), o compilador irá definir automaticamente o seu endereço.
- **Bit:** para tipo de dados booleanos, é necessário definir o bit que esta acessa (0 - 7)

- **Valor Inicial:** valor que será carregado para variável durante a inicialização do equipamento
- **Comentário:** comentário da variável no idioma selecionado

Sistema:

▼ Tag	Size	Datatype	At	Address	Bit	Initial Value	Comment	Modbus
FIRMWARE	0	UINT	%SW	0	0	0	Firmware version	3000

- **Tag:** identificação da variável
- **Tamanho:** número de elementos do array relacionado à variável
- **Tipo de Dado:** tipo numérico da variável
- **At:** define qual área de memória global a variável acessa
- **Endereço:** endereço relacionado a área de memória global
- **Bit:** para tipo de dados booleanos, é necessário definir o bit que esta acessa (0 - 7)
- **Comentário:** comentário da variável no idioma selecionado

I/O:

Tag	Size	Datatype	At	Address	Bit	Initial Value	Comment	Modbus
DI1	0	BOOL	%IB	0	0	0	Digital input 1	16000

- **Tag:** identificação da variável
- **Tamanho:** número de elementos do array relacionado à variável
- **Tipo de Dado:** tipo numérico da variável
- **At:** define qual área de memória global a variável acessa
- **Endereço:** endereço relacionado a área de memória global
- **Bit:** para tipo de dados booleanos, é necessário definir o bit que esta acessa (0 - 7)
- **Comentário:** comentário da variável no idioma selecionado



NOTA!

Quando o tamanho é maior que zero, as variáveis são acessadas no Ladder através do seu índice de array.

8.2.4.3 Editando na Rung

Os elementos Ladder inseridos na *rung* necessitam que variáveis sejam definidas para cada argumento. Veja figura a seguir.

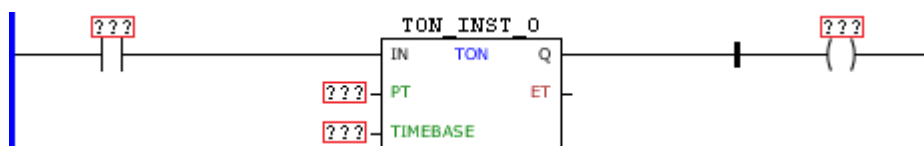


Figura 1 - Variáveis sem declaração nos elementos e blocos



ATENÇÃO!

A partir da versão 1.30 do WPS os argumentos de saída dos blocos funcionais podem ser opcionais.

Os argumentos em questão não serão inicializados com a declaração ??? e poderão ser omitidos caso não necessitem serem utilizados na lógica Ladder.

Para definir uma variável para o argumento é necessário entrar no modo de edição do argumento, feito através de duas maneiras.

1. Pelo mouse

Executando um duplo clique do mouse diretamente no argumento

2. Pelo teclado

Pressionando-se a tecla **F2**.

Para elementos com apenas um argumento, pressionar a tecla **F2** entra no modo de edição de argumento diretamente.

Para outros elementos com mais de um argumento, é necessário selecionar o argumento através das teclas direcionais e então pressionar novamente a tecla **F2**.

Ao entrar no modo de edição, o elemento aparecerá similar à seguinte figura.



Figura 2 - Atribuindo as variáveis

Neste momento, uma caixa de edição estará habilitada para que você entre com o nome da variável.

Ao pressionar o botão **Editar**, uma janela para criar a nova variável será exibida.

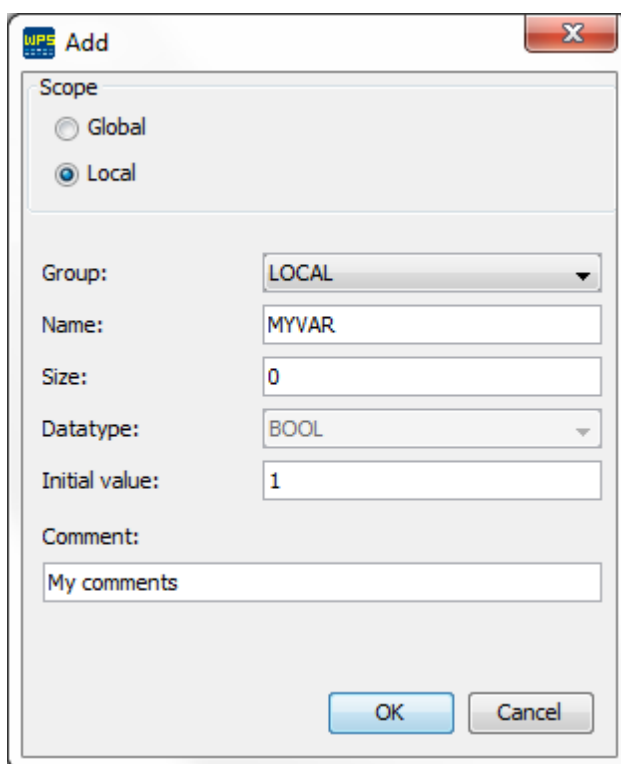
A janela de diálogo "Add" tem um título "WPS Add" e um botão de fechar "X". Ela contém as seguintes opções: "Scope" com radio buttons para "Global" e "Local" (onde "Local" está selecionado); "Group:" com um menu suspenso mostrando "LOCAL"; "Name:" com um campo de texto contendo "MYVAR"; "Size:" com um campo de texto contendo "0"; "Datatype:" com um menu suspenso mostrando "BOOL"; "Initial value:" com um campo de texto contendo "1"; e "Comment:" com um campo de texto contendo "My comments". No rodapé, há botões "OK" e "Cancel".

Figura 3 - Criando a variável se o seu tag não está definido

Nesta janela você deverá definir as seguintes opções para a variável:

- Escopo: se pertence ao escopo **GLOBAL** ou **LOCAL**
- Grupo: a qual grupo do escopo selecionado a variável pertence
- Nome: identificação da variável
- Tamanho: número de elementos do array* relacionado à variável
- Tipo de Dado: tipo numérico da variável

Caso já existam variáveis definidas para o tipo compatível com o elemento Ladder, uma caixa de seleção com essas variáveis aparecerá em conjunto com a caixa de edição. Para selecionar a variável desejada, use as teclas direcionais para baixo e para cima para navegar pelas variáveis e, após estar com a variável desejada selecionada, pressione **Enter**. A figura a seguir mostra essa funcionalidade.

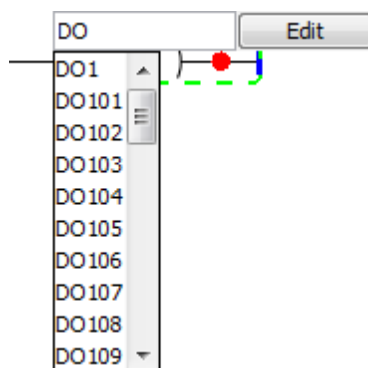


Figura 4 - Selecionando variáveis ao digitar

8.2.4.4 Literais na Rung

Nos blocos funcionais também é possível a entrada de valores literais, conforme figura a seguir.

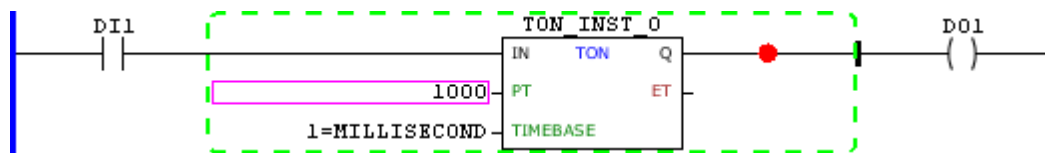


Figura 1 - Exemplo de Programa

Valores literais são aqueles que não estão atrelados a nenhuma variável, sendo estes inseridos no diagrama Ladder com seu valor explícito.

Nesse exemplo foi configurada a entrada PT do bloco TON com o valor literal 1000.

Para a entrada de literais, devem ser usadas as seguintes convenções:

- Número inteiro não tem ponto.
o Ex. 12, 1000, 1555
- Números reais devem obrigatoriamente ter ponto.
o Ex. 1.5, 2.25, 3.0
- Números representados em hexadecimal obrigatoriamente devem definir o tipo do dado.
o Ex. BYTE#16#7F, WORD#16#3CF0, DWORD#16#00FF0088
- Números representados em binário obrigatoriamente devem definir o tipo do dado.
o Ex. BYTE#2#1010_0000, WORD#2#0111_0000_0000_0001

ATENÇÃO!

Em alguns blocos, por questões de verificação de consistência de dados, será necessário definir o tipo do dado do literal através de notação específica, que terá as seguintes opções: BOOL#, BYTE#, INT#, UINT#, DINT#, UDINT#, WORD#, DWORD# e REAL#. Exemplo: WORD#17321

8.2.4.5 Arrays na Rung

Nos contatos, bobinas e blocos funcionais também é possível o acesso de variáveis do tipo array, conforme figuras a seguir.

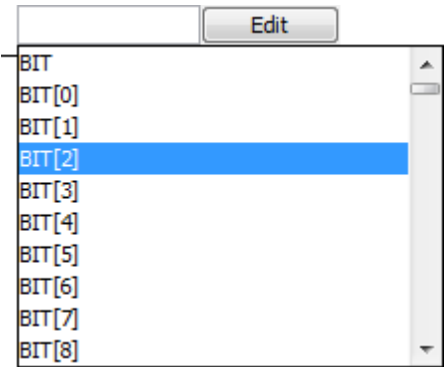


Figura 1 - Acesso de array

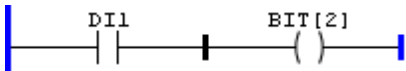


Figura 2 - Array na Rung

Para visualizar os índices de uma variável do tipo array, expanda a variável na janela de variáveis, conforme figura a seguir.

Variables

GLOBAL GLOBAL_RETAIN GLOBAL_SYSTEM GLOBAL_IO GLOBAL_NETWORK CONSTANT

Tag	Size	Datatype	At	Address	Bit	Initial ...	Comment	Modbus
BIT	9	BOOL	%MB					
BIT[0]	0	BOOL	%MB			0		
BIT[1]	0	BOOL	%MB			0		
BIT[2]	0	BOOL	%MB			0		
BIT[3]	0	BOOL	%MB			0		
BIT[4]	0	BOOL	%MB			0		
BIT[5]	0	BOOL	%MB			0		
BIT[6]	0	BOOL	%MB			0		
BIT[7]	0	BOOL	%MB			0		
BIT[8]	0	BOOL	%MB			0		

GLOBAL LOCAL

Figura 3 - Exibindo índices de uma variável do tipo array

8.2.4.6 Instâncias e Estruturas na Rung

Nos contatos, bobinas e blocos funcionais também é possível o acesso às variáveis internas de instâncias e estruturas, conforme figuras a seguir.

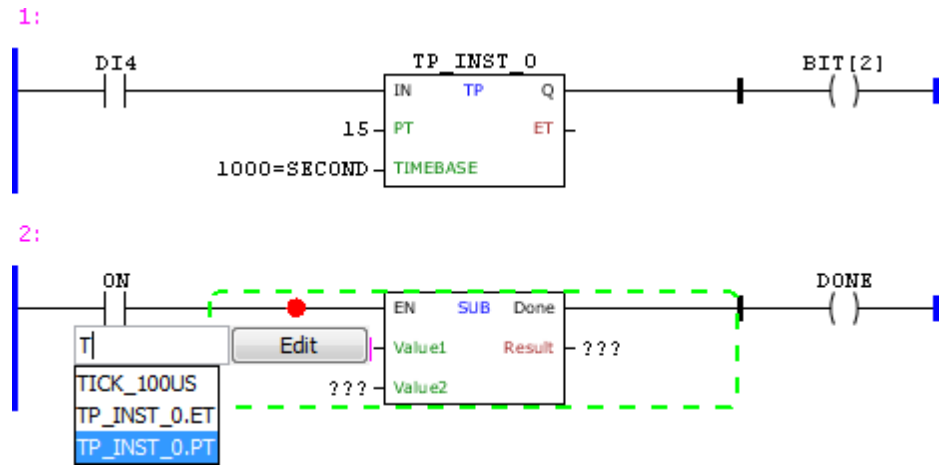


Figura 1 - Acesso à variável interna de instância

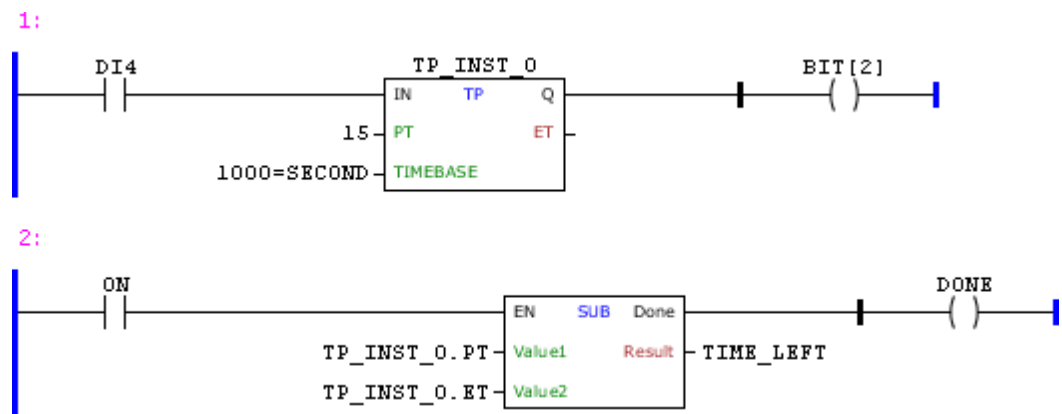


Figura 2 - Variável interna de instância na rung

Para visualizar as variáveis internas das instâncias e estruturas, expanda a variável na janela de variáveis, conforme figura a seguir.

Tag	Size	Datatype	Initial Value	Comment
TP_INST_0	0	TP		
TP_INST_0.PT	0	DWORD		
TP_INST_0.ET	0	DWORD	0	
TP_INST_0.IN	0	BOOL	0	
TP_INST_0.Q	0	BOOL	0	

Figura 3 - Exibindo variáveis internas de instância ou estrutura

8.2.4.7 Instâncias Voláteis e Retentivas

Blocos Funcionais (FBs)

Os FBs possuem variáveis internas, que armazenam seus dados durante os ciclos consecutivos de execução. Conforme a necessidade da aplicação, estes FBs podem ter suas instâncias configuradas como retentivas (**LOCAL_RETAIN** ou **GLOBAL_RETAIN**) ou voláteis (**LOCAL** ou **GLOBAL**). As variáveis de entrada e saída associadas ao FB também podem ser configuradas como retentivas ou voláteis. As variáveis retentivas retêm seus valores após o desligamento do equipamento, enquanto as voláteis carregam seus valores iniciais após uma reinicialização.

Quando queremos que um FB mantenha os valores após a reinicialização do equipamento, é necessário que a instância do FB e as variáveis associadas às suas entradas sejam configuradas como retentivas. Isso fará com que as variáveis internas do FB e as variáveis de entrada associadas mantenham o valor anterior ao desligamento.

No exemplo abaixo temos a utilização do bloco TON (**Temporizador de Habilitação em Atraso**) com instância e variáveis retentivas:

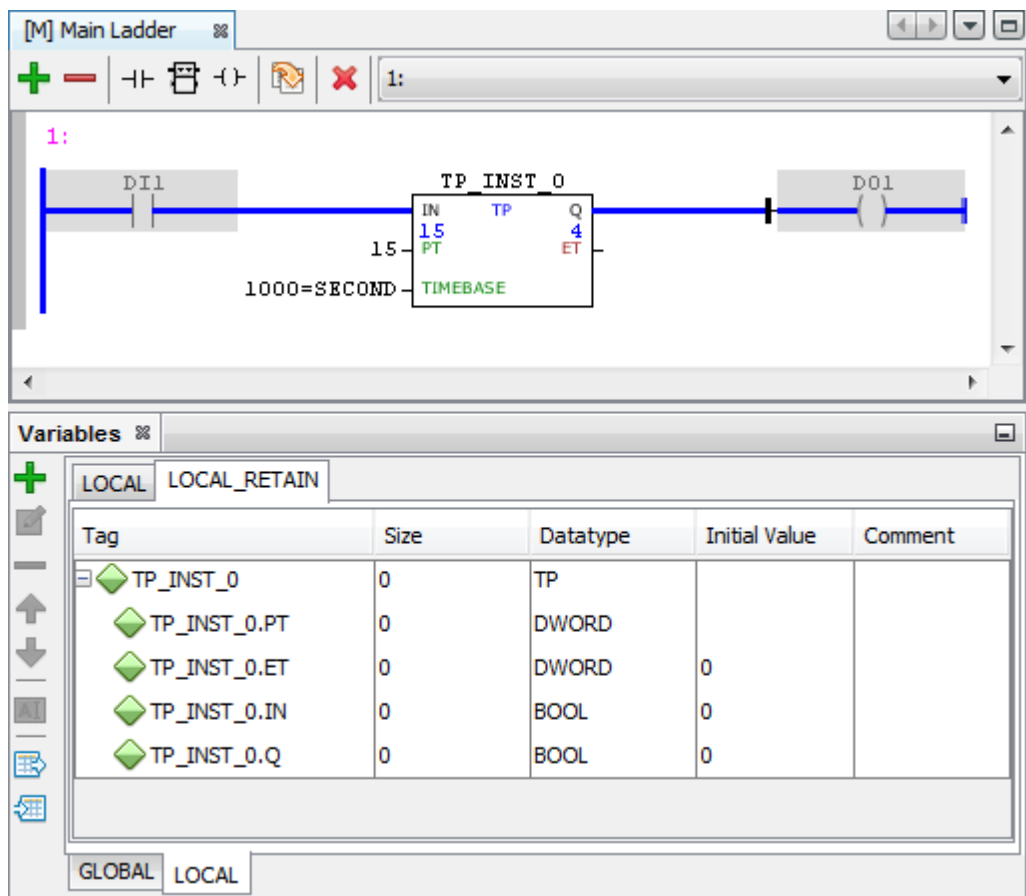


Figura 1 - Bloco TP com instância e variáveis retentivas antes da reinicialização.

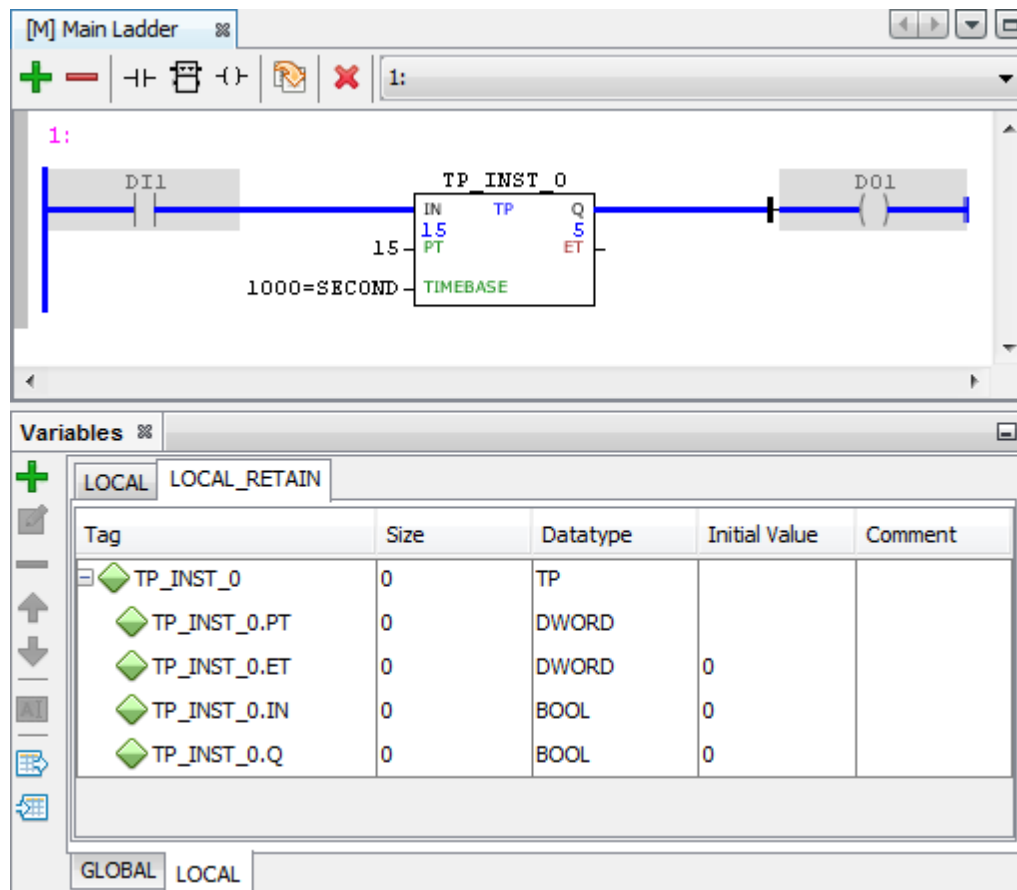


Figura 2 - Bloco TP com instância e variáveis retentivas 1 segundo após a reinicialização.

Quando queremos que um FB reinicialize seus valores após o desligamento do equipamento, é necessário que a instância do FB e as variáveis associadas às suas entradas sejam configuradas como voláteis. Isso fará com que as variáveis internas do FB e as variáveis de entrada associadas reiniciem o valor anterior ao desligamento.

Abaixo, segue um exemplo da utilização do bloco CTU (**Contador Incremental**) com instância e variáveis voláteis:

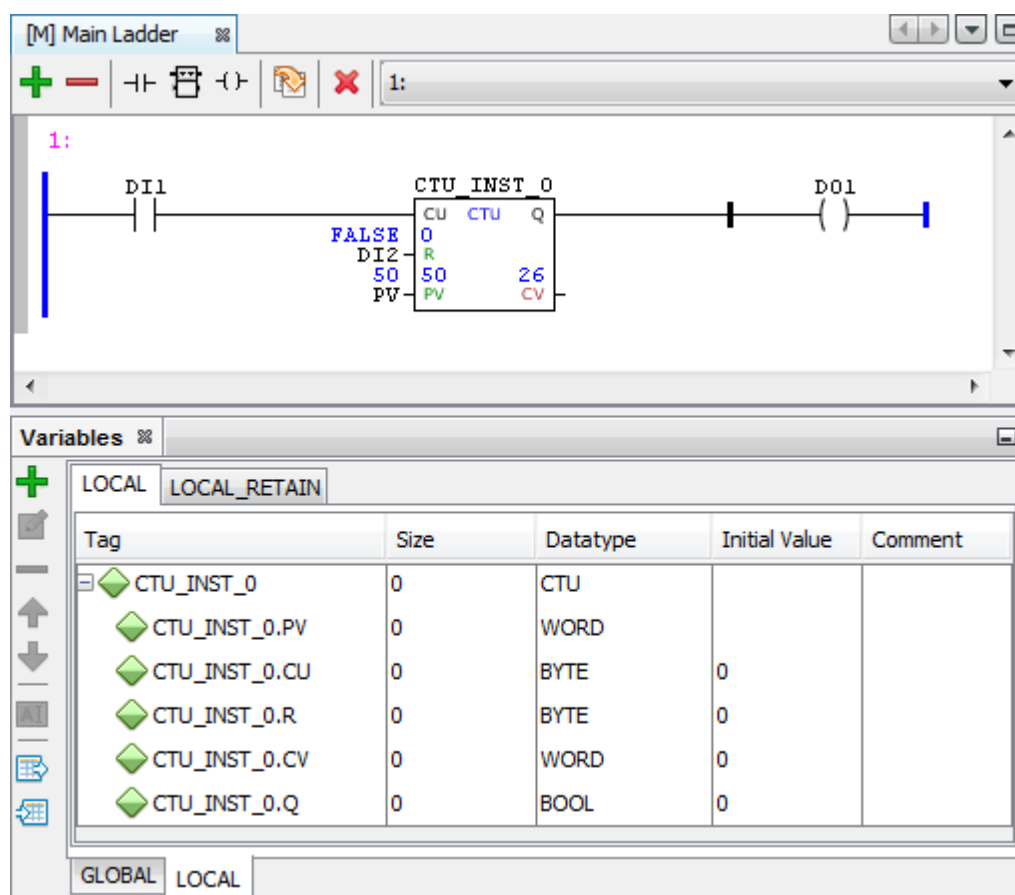


Figura 3 - Bloco CTU com instância e variáveis voláteis antes da reinicialização.

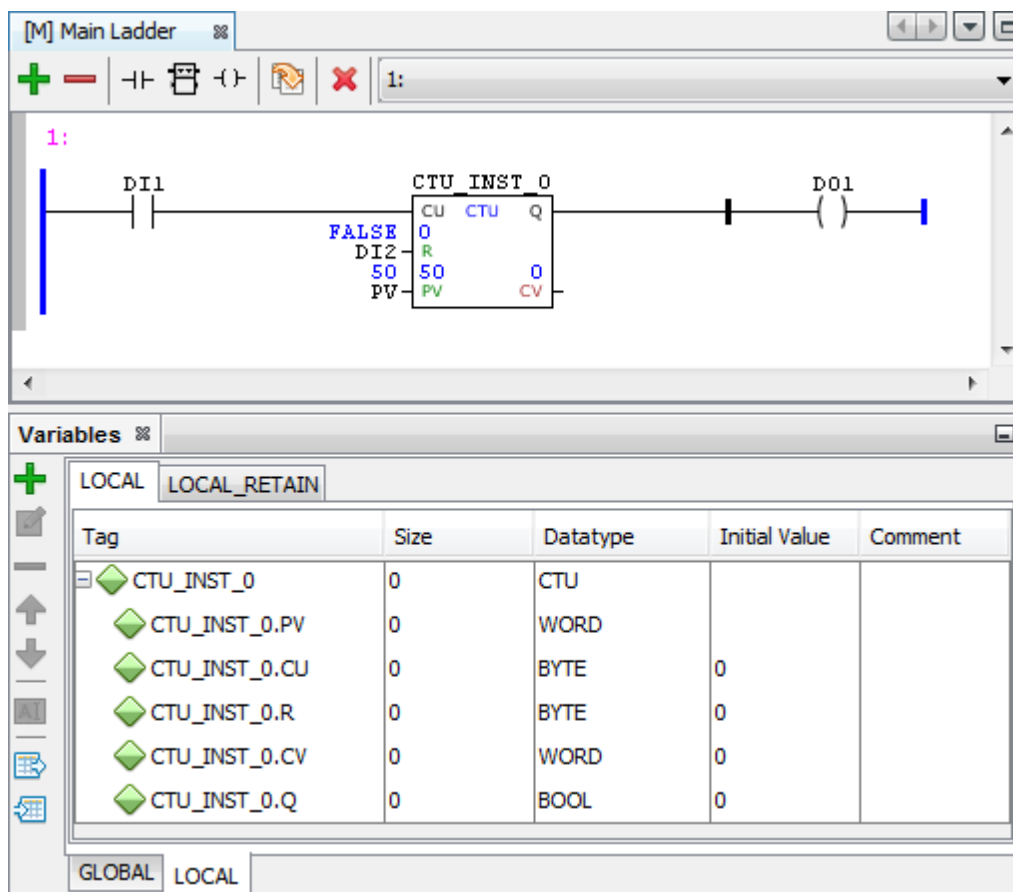


Figura 4 - Bloco CTU com instância e variáveis voláteis após a reinicialização.

Bloco do usuário (USERFB)

Na utilização do USERFB é possível definir variáveis do tipo **LOCAL**, **LOCAL_RETAIN**, **VAR_IN**, **VAR_OUT** e **VAR_IN_OUT**. As variáveis internas definidas como **LOCAL**, serão sempre voláteis e as de tipo **LOCAL_RETAIN** serão sempre retentivas. As variáveis internas definidas como **VAR_IN**, **VAR_OUT** e **VAR_IN_OUT** serão voláteis caso a instância do USERFB seja associado ao grupo **LOCAL** ou **GLOBAL** e retentivas caso seja associado ao grupo **LOCAL_RETAIN** ou **GLOBAL_RETAIN**.

8.2.5 Compilação

Para compilar um POU, há 4 opções:

1. Através do menu **Configuração > Contruir Recurso Principal**.

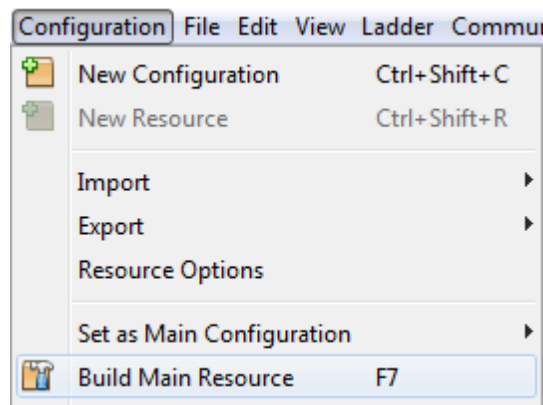


Figura 1 - Compilar a partir do Menu

2. Através da tecla de atalho **F7**.
3. Através do botão na **Barra de Ferramentas**.



Figura 2 - Compilar a partir da Barra de Ferramentas

4. Clicando-se com o botão direito do mouse sobre o nome do recurso.

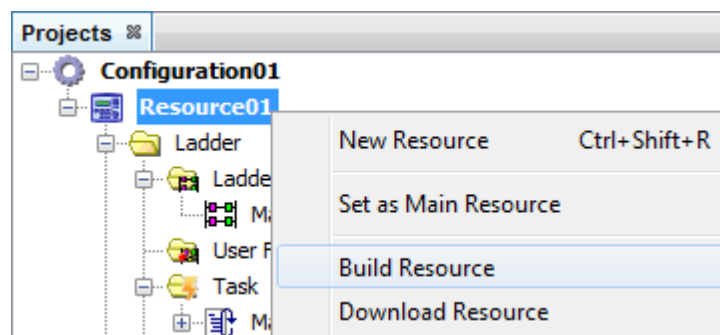


Figura 3 - Botão direito do mouse sobre o recurso

Os resultados da compilação, indicando os erros e avisos, podem ser visualizados através da janela **Saída Padrão**.

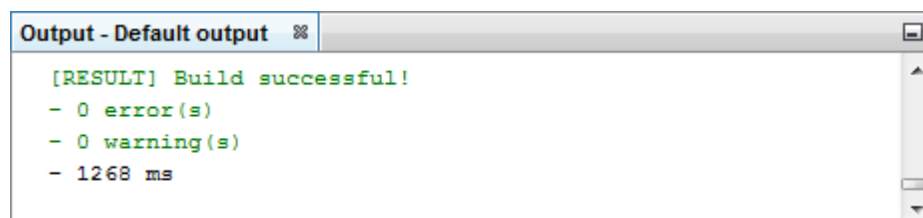


Figura 4 - Resultados da compilação

8.2.6 Download

Para fazer o *download* de um recurso, há 4 opções:

1. Através do menu **Comunicação > Download do Recurso Principal**.

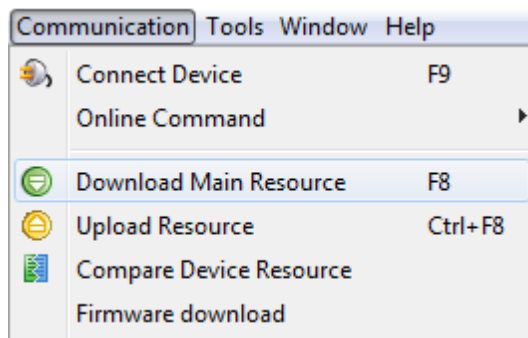


Figura 1 - Download a partir do Menu

2. Através da tecla de atalho **F8**.
3. Através do botão na **Barra de Ferramentas**.



Figura 2 - Download a partir da Barra de Ferramentas

4. Clicando-se com o botão direito do mouse sobre o nome do recurso.

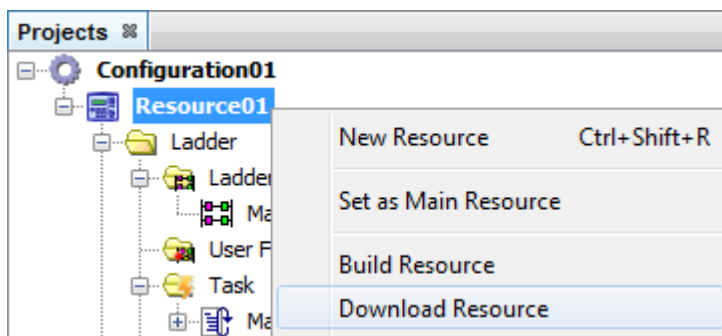


Figura 3 - Botão direito do mouse sobre o recurso

8.2.7 Monitoração Online

Visão Geral

Após o programa Ladder ser compilado e carregado no equipamento, é possível monitorar o Ladder

pressionando o botão **Conectar Equipamento** .

Neste momento, o WPS v2.3X tentará estabelecer a comunicação com o equipamento, testando a conectividade com o mesmo.

A monitoração on-line representará graficamente o estado lógico do programa Ladder. Um exemplo de monitoração on-line pode ser visto na figura a seguir.

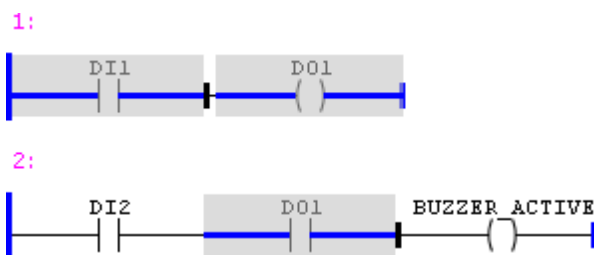


Figura 1 - Exemplo de monitoração on-line

Para blocos funcionais, os valores serão apresentados conforme figura a seguir:

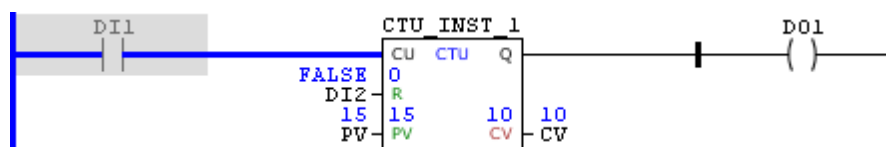


Figura 2 - Valores de monitoração dos blocos funcionais

Os valores das variáveis são mostrados em cima de suas respectivas variáveis, os valores internos da instância são mostrados em cima do nome do argumento respectivo.

Escrita de variáveis

Para a escrita de variáveis, basta dar um clique duplo sobre a variável que se deseja escrever. Então, uma caixa de escrita de valores abrirá, conforme figura a seguir.

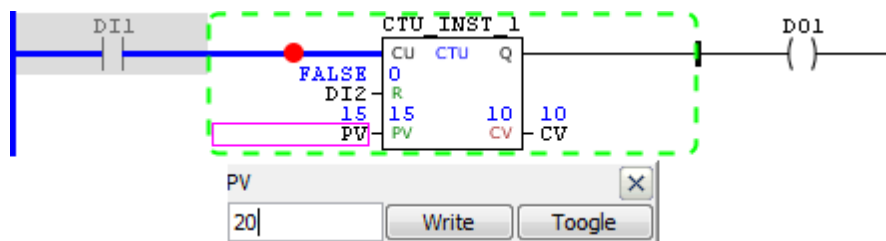


Figura 3 - Escrita de variáveis

Nesta caixa, digite o valor desejado e pressione o botão **Write** para escrever o valor. O botão **Toggle** serve para alternar o valor escrito de 0 para o valor atual e vice-versa.

Monitoração de instâncias

Para monitorar uma instância de um bloco funcional, basta dar um duplo clique sobre a variável da instância. Então, uma caixa relativa à monitoração de instância abrirá, conforme figura a seguir.

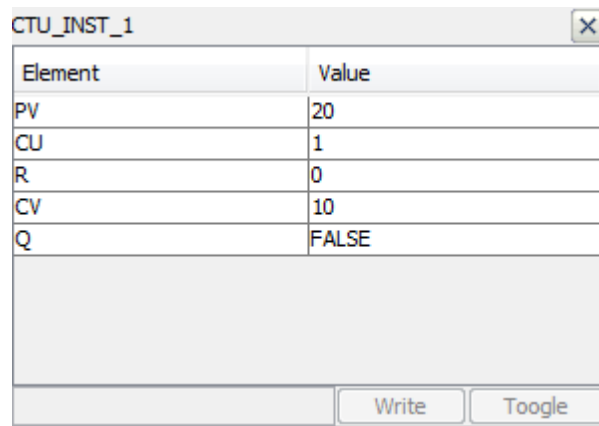


Figura 4 - Monitoração de instâncias

Para escrever em variáveis internas da instância basta clicar na linha correspondente e utilizar a caixa de edição de valores e os botões **Write** e **Toogle**, conforme mencionado no item anterior.

Monitoração de estruturas

Para as variáveis criadas a partir de estruturas definidas no recurso, basta dar um duplo clique sobre a variável correspondente. Uma caixa similar àquela de monitoração de instâncias abrirá. Veja o exemplo a seguir.

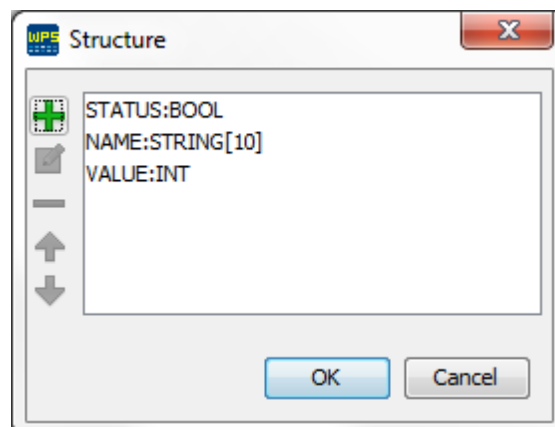


Figura 5 - Estrutura definida no recurso

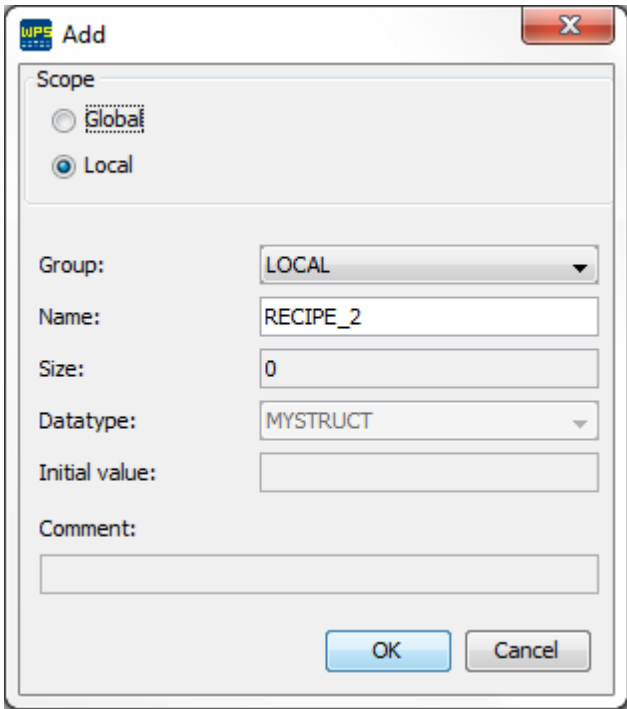


Figura 6 - Variável criada com tipo de dado da estrutura definida no recurso

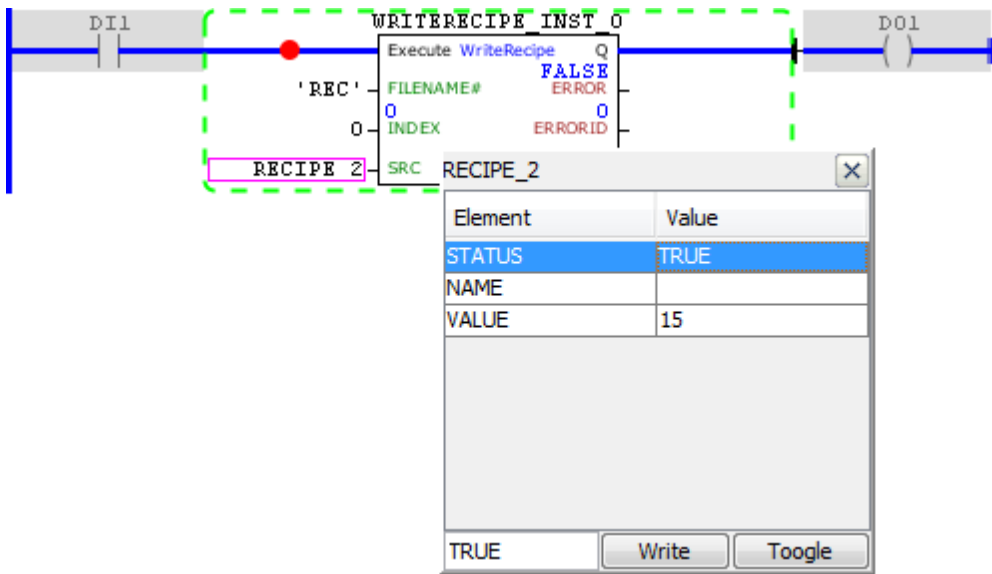


Figura 7 - Monitoração da variável utilizado no bloco WriteRecipe

Monitoração de Arrays

Para as variáveis criadas com tamanho maior que zero, é possível monitorar todos os dados contidos nela. Para isso, basta clicar sobre a variável correspondente que uma caixa de monitoração abrirá. Veja exemplo a seguir.

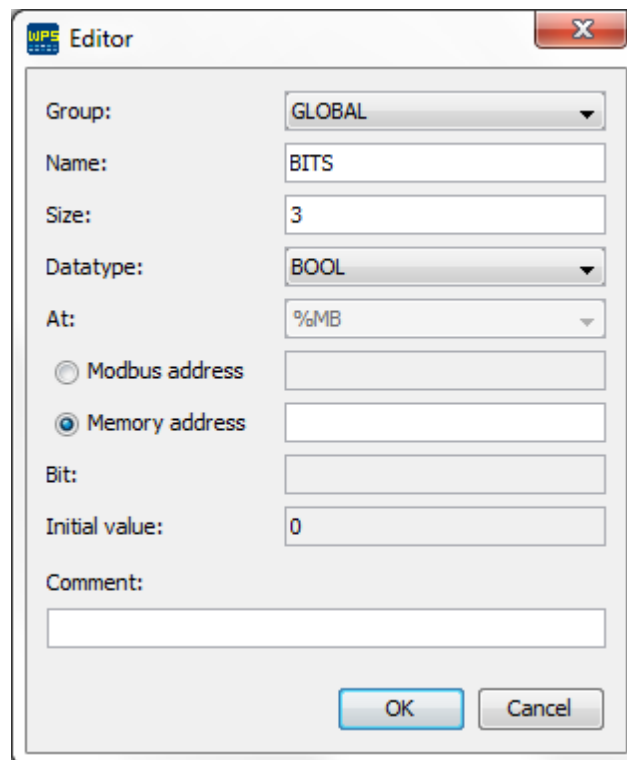


Figura 8 - Variável criada com size maior que zero, array

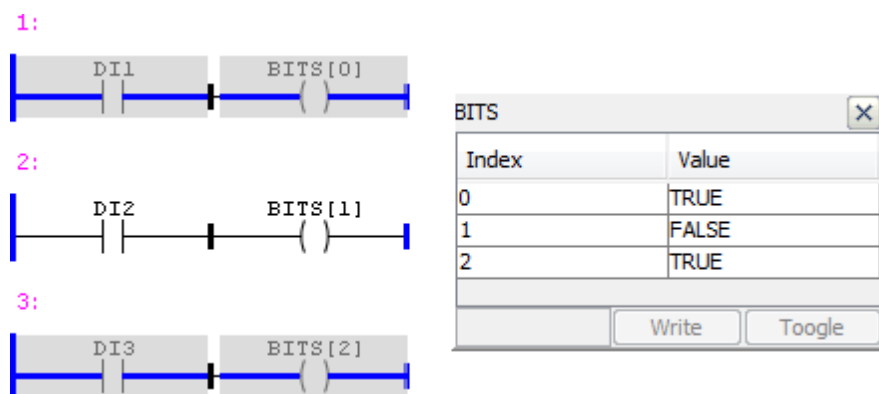


Figura 9 - Monitoração da variável utilizado no bloco MB_ReadRegister

8.3 Trabalhando com USERFBs

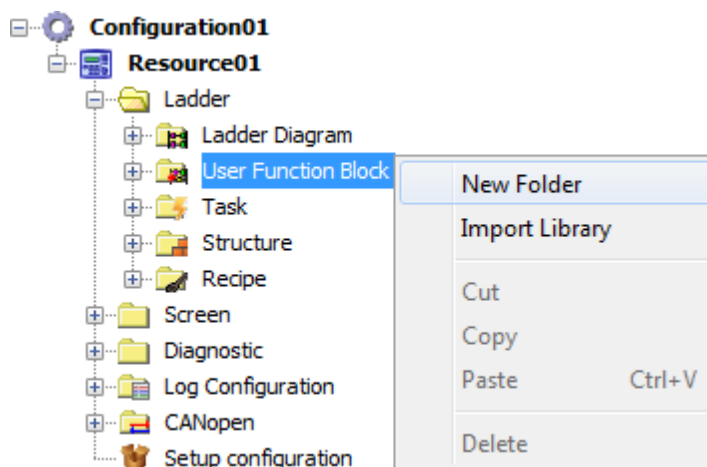
Ver a seguir.

8.3.1 Criando USERFBs

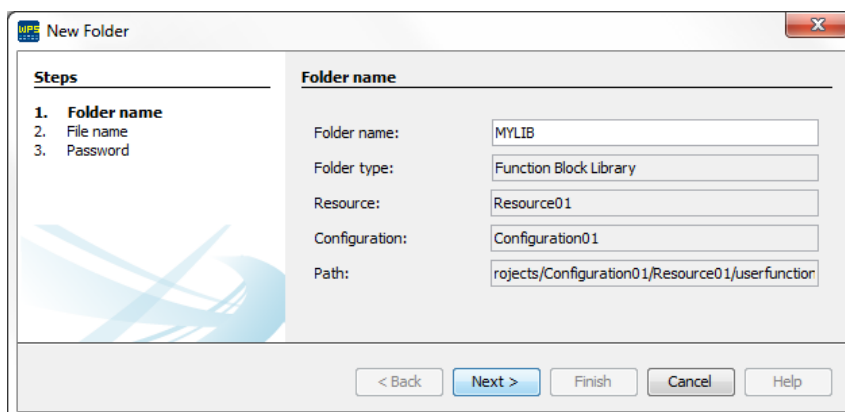
Os USERFBs são blocos funcionais personalizáveis pelo usuário. A sua utilização é encorajada para tornar o programa Ladder menos volumoso e poluído, abstraindo informações com as quais não se quer trabalhar frequentemente e sistematizando tarefas complexas.

Nestes blocos, suas entradas e saídas são definidas pelo próprio usuário, que as manipula no diagrama Ladder associado ao bloco. Veja a seguir como criar seu USERFB.

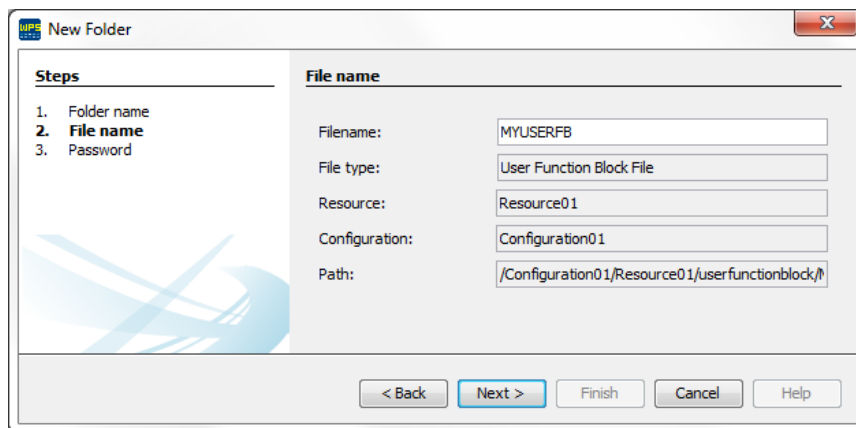
1. Na janela de **Projetos**, no recurso em que se deseja criar o USERFB, clique com o botão direito em **Bloco de Função do Usuário** e clique em **Nova pasta**.



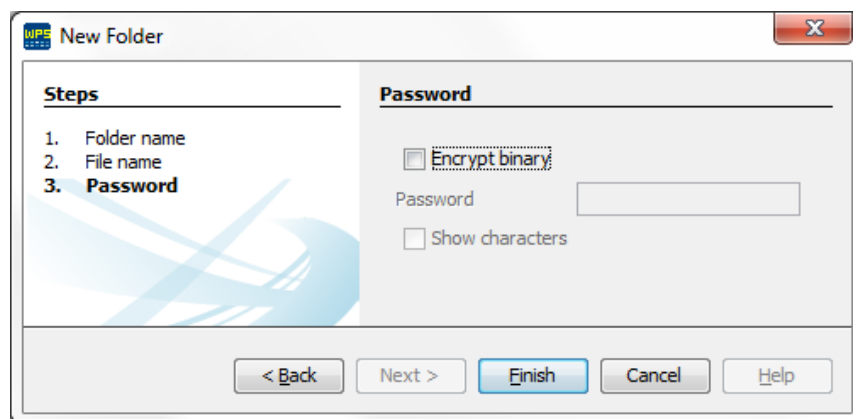
2. No assistente de configuração, insira um nome para a biblioteca da qual fará parte o USERFB e clique **Próximo**.



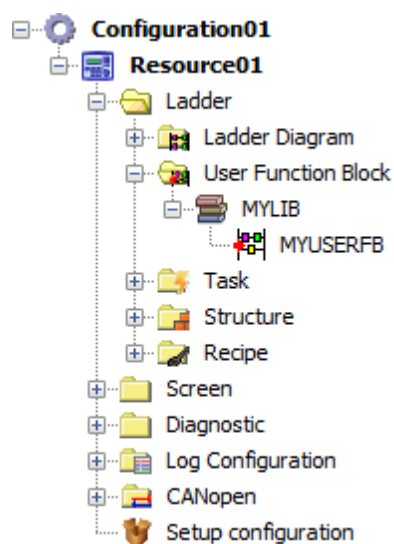
3. Insira um nome válido para o USERFB e clique **Próximo**.



4. Caso você queira inserir uma senha para proteger o bloco, marque a caixa de seleção **Criptografar binário do código fonte** e insira uma senha. Caso contrário, desmarque a caixa de seleção. Clique **Finalizar**.



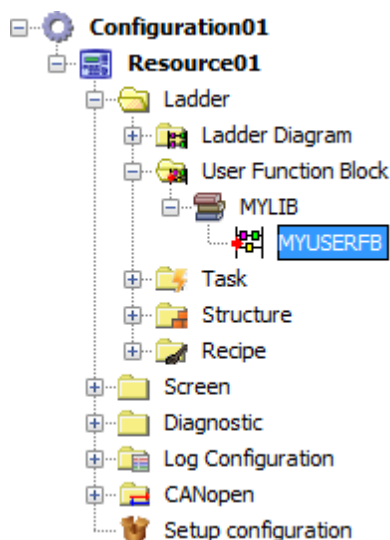
Pronto! O USERFB já foi criado com sucesso. Você deverá ver o seguinte menu na janela de **Projetos**:



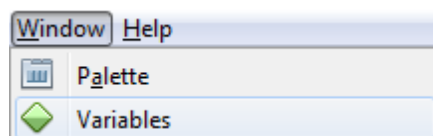
8.3.2 Adicionando entradas e saídas

Agora veremos como criar entradas e saídas para o bloco USERFB.

1. Na janela de **Projetos**, dê um duplo clique no arquivo do USERFB para abrir o editor Ladder do bloco.

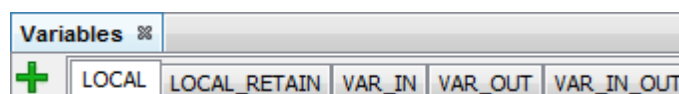


2. No menu **Janela**, clique em **Variáveis**.

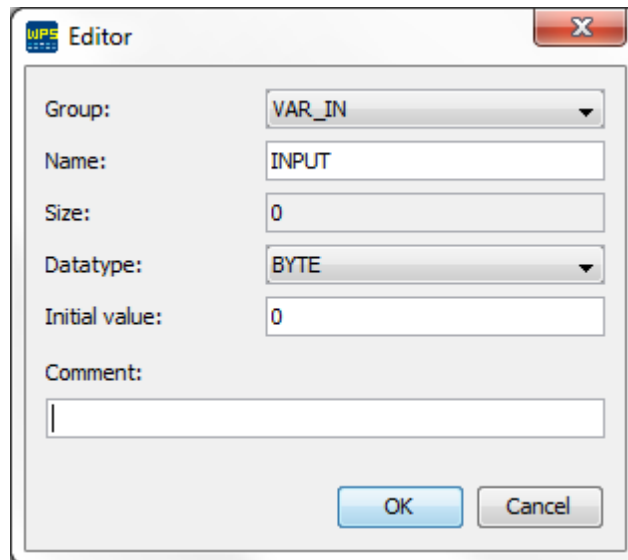


Analisando a figura abaixo, vemos que a janela de **Variáveis** do USERFB é diferente dos outros arquivos Ladder. Ele possui apenas variáveis voláteis e retentivas de escopo **LOCAL**, que são as variáveis internas do bloco utilizadas na subrotina. Além destas, ele possui outras três categorias: **VAR_IN**, **VAR_OUT** e **VAR_IN_OUT**.

- **VAR_IN**: variáveis internas que serão os argumentos de entrada desse bloco.
- **VAR_OUT**: variáveis internas que serão os argumentos de saída desse bloco.
- **VAR_IN_OUT**: variáveis internas que serão os argumentos de entrada e saída desse bloco.



3. Para criar um entrada, clique na aba **VAR_IN** e clique no símbolo . Na janela que aparecer, atribua um nome e um tipo a esta variável e clique em **OK**.

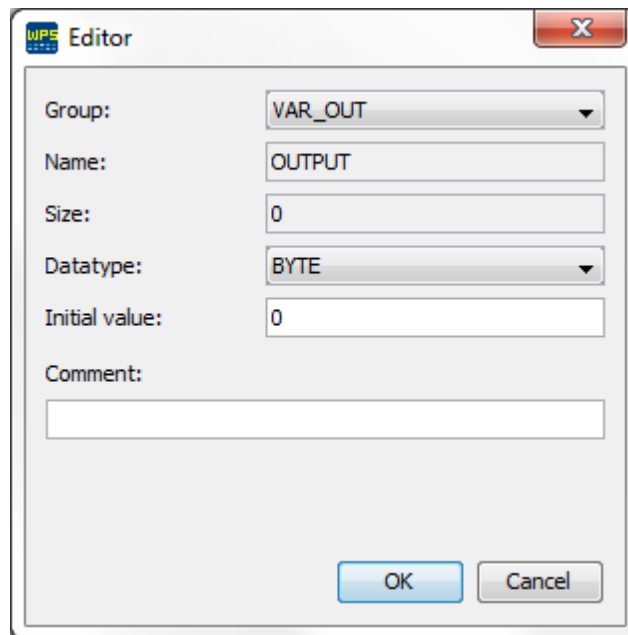


The image shows a dialog box titled "WPS Editor" with a close button (X) in the top right corner. The dialog contains the following fields:

- Group: VAR_IN (dropdown menu)
- Name: INPUT (text field)
- Size: 0 (text field)
- Datatype: BYTE (dropdown menu)
- Initial value: 0 (text field)
- Comment: (empty text area)

At the bottom right, there are two buttons: "OK" and "Cancel".

4. Para criar uma saída, clique na aba **VAR_OUT** e clique no símbolo . Na janela que aparecer, atribua um nome e um tipo a esta variável e clique em **OK**.

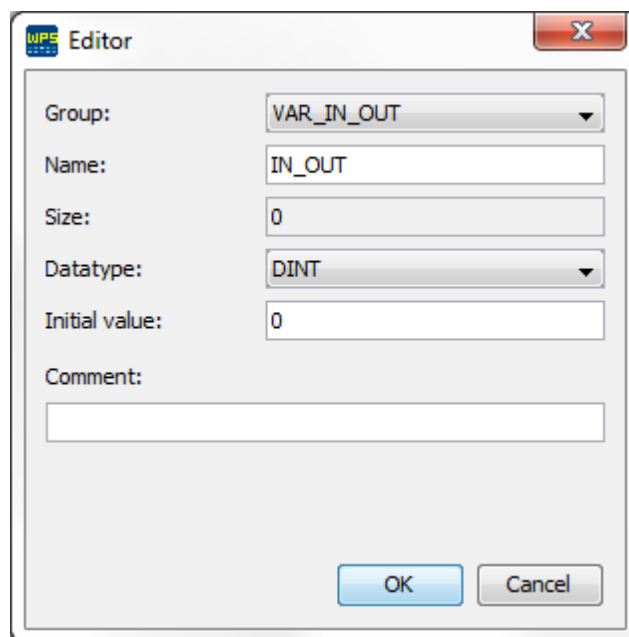


The image shows a dialog box titled "WPS Editor" with a close button (X) in the top right corner. The dialog contains the following fields:

- Group: VAR_OUT (dropdown menu)
- Name: OUTPUT (text field)
- Size: 0 (text field)
- Datatype: BYTE (dropdown menu)
- Initial value: 0 (text field)
- Comment: (empty text area)

At the bottom right, there are two buttons: "OK" and "Cancel".

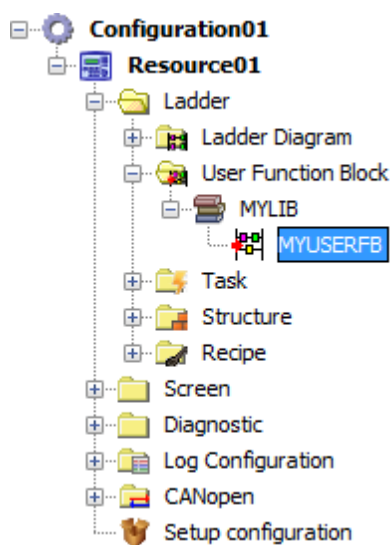
5. Para criar um argumento de entrada e saída, clique na aba **VAR_IN_OUT** e clique no símbolo . Na janela que aparecer, atribua um nome e um tipo a esta variável e clique em **OK**.



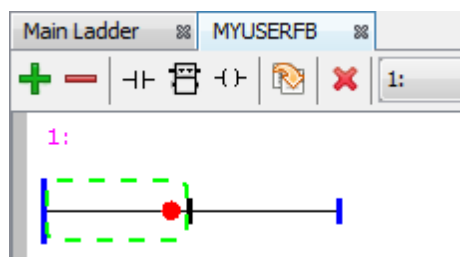
8.3.3 Editando a rotina

Agora veremos como editar a rotina do bloco USERFB.

1. Na janela de **Projetos**, dê um duplo clique no arquivo do USERFB para abrir o editor Ladder do bloco.



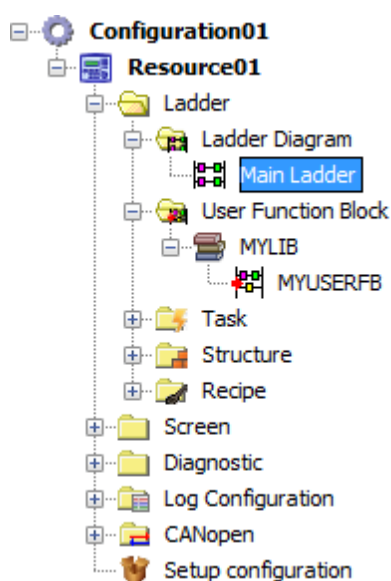
O editor Ladder abrirá, como em qualquer outro diagrama Ladder. Neste, é possível adicionar qualquer tipo de bloco, inclusive outros USERFBs. Atente, porém, que todas as variáveis utilizadas são locais.



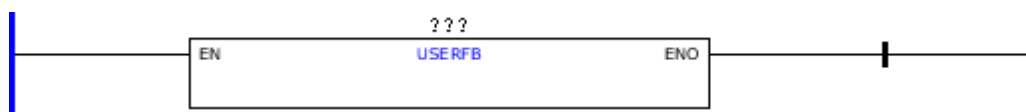
8.3.4 Utilizando o USERFB

Por fim, veremos como utilizar o bloco USERFB, chamando-o em outros diagramas.

1. Na janela **Projetos**, dê um duplo clique no arquivo em que deseja utilizar o USERFB para abrir o editor Ladder deste arquivo.



2. Na janela **Paleta**, selecione o bloco USERFB da categoria **Module** e arraste-o até a posição onde deseja utilizá-lo no diagrama Ladder.

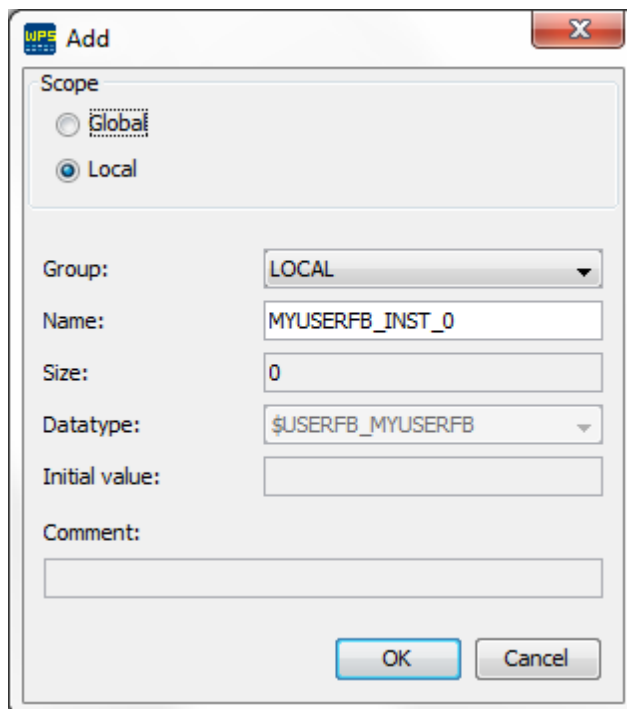


3. Dê um duplo clique nos pontos de interrogação sobre o bloco (???) para inserir variável de instância do USERFB. Digite o nome da variável e clique em **Editar**. Na janela de confirmação, clique **Sim** para criar a nova variável.

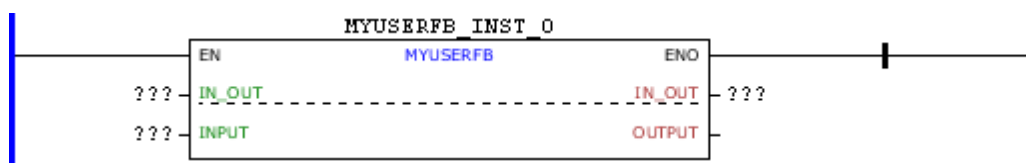


4. Na janela que aparecer, digite um nome para a variável e selecione seus parâmetros. No campo **Tipo**, escolha o nome do seu USERFB (caso você tenha apenas um, o campo não poderá ser editado).

Por exemplo, caso o nome do seu USERFB seja MYUSERFB, o tipo correto a ser selecionado é \$USERFB_MYUSERFB.



Finalizado! Seu USERFB já está inserido no diagrama e pronto para funcionar!



9 Trend

[Visão Geral](#)

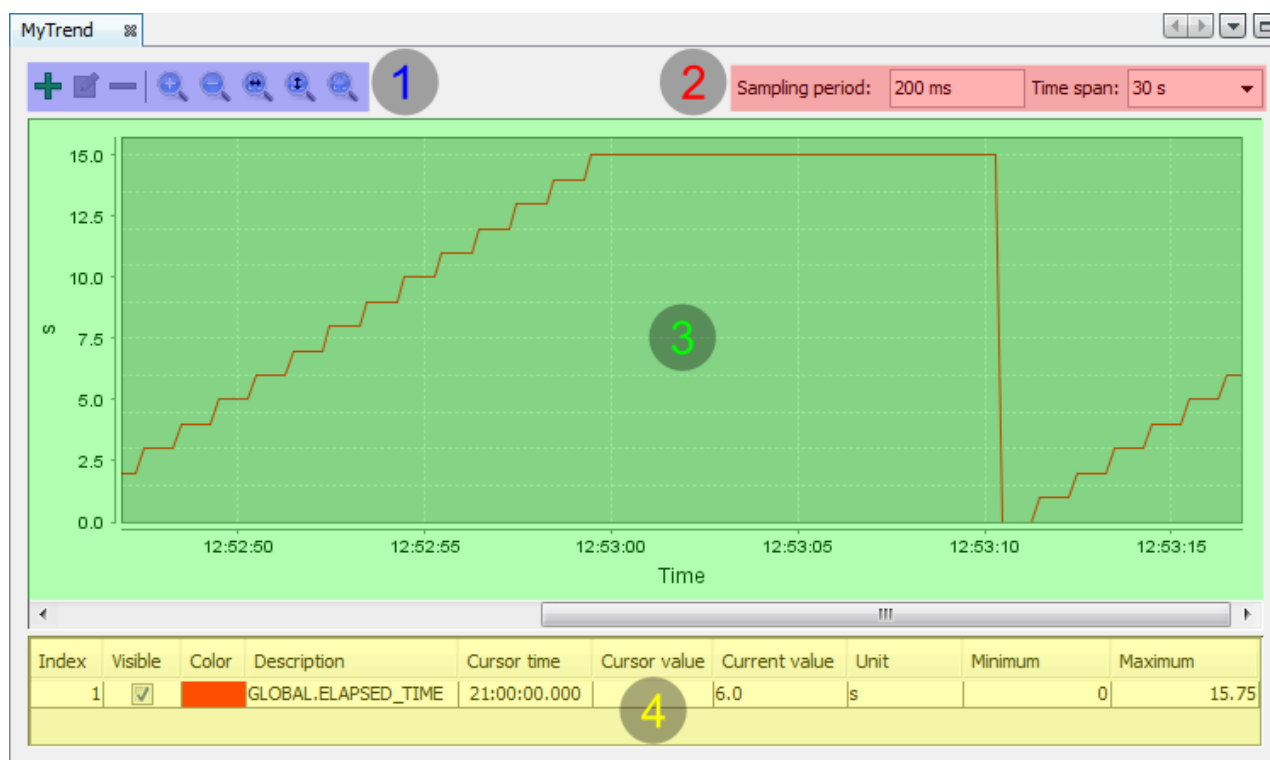
[Configuração](#)

9.1 Visão Geral

Trend é um gráfico de monitoração de variáveis em função do tempo.

A função **Trend** do WPS v2.3X possui dez canais de monitoração, ou seja, é possível a monitoração de até dez variáveis ou parâmetros ao mesmo tempo.

Abaixo segue uma visão geral sobre a tela de configuração da função **Trend**.

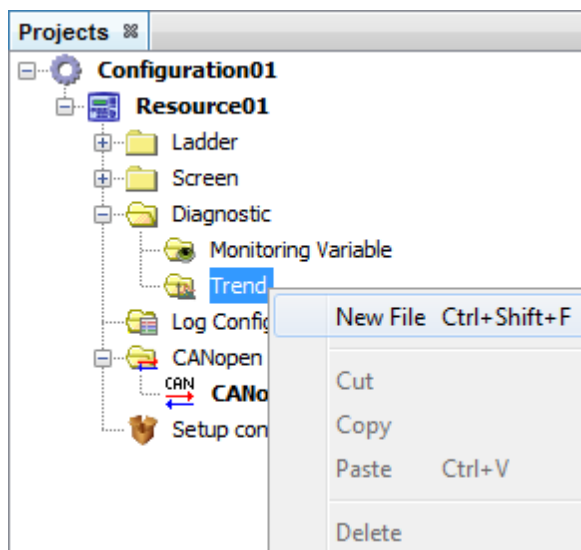


1. **Canais e Zoom Gráfico:** Nesta barra estão as opções para adição, edição e remoção de canais, além das opções para controle de zoom do gráfico.
2. **Tempo:** Nesta barra estão as opções de configuração dos períodos de amostragem e da faixa de tempo a ser exibida no gráfico.
3. **Gráfico:** Nesta tela são exibidos os valores monitorados dos canais em forma gráfica. Na eixo horizontal visualiza-se o horário de coleta dos dados e no eixo vertical é exibida a faixa de valores por unidade de medida dos canais.
4. **Tabela de Canais:** Nesta tabela é possível visualizar os dados dos canais escolhidos na posição em que o cursor se encontra, além de permitir ocultar canais, alterar sua cor e ajustar os limites do gráfico por unidade de medida.

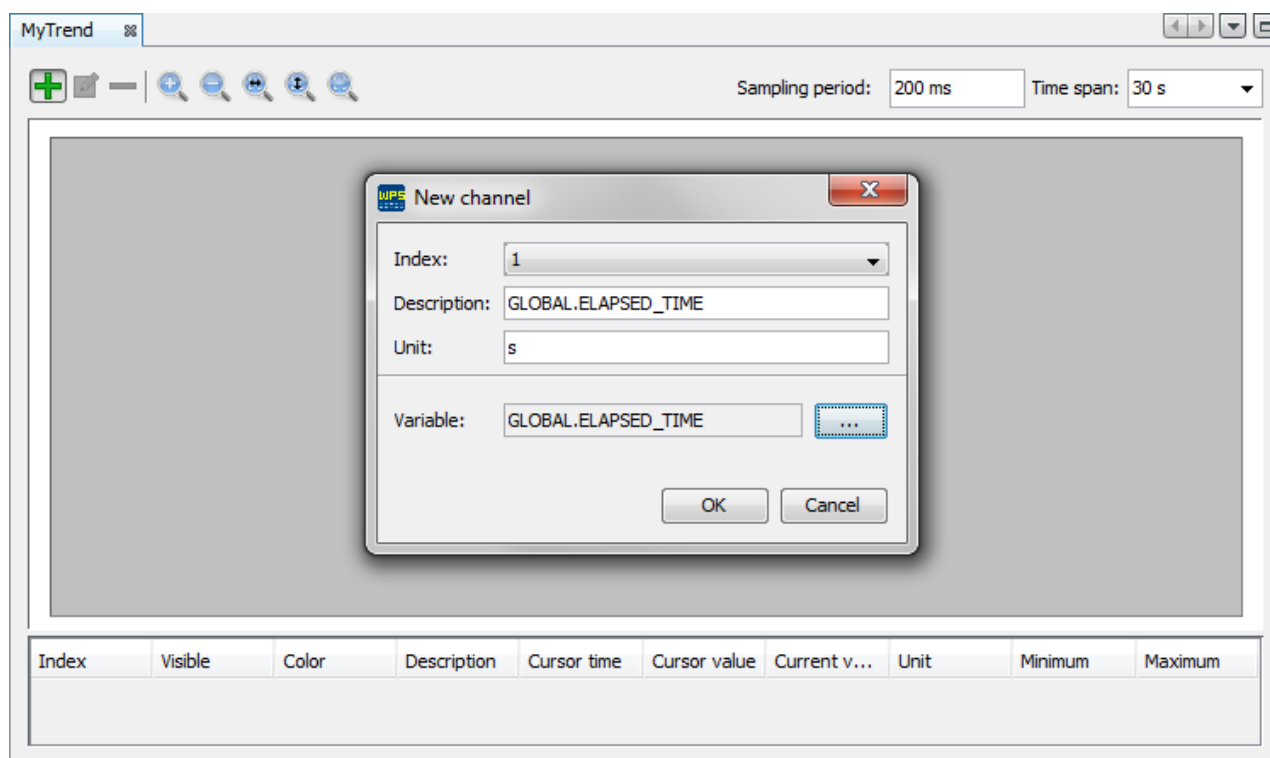
9.2 Configuração

Abaixo segue a lista dos passos necessários para criação de uma configuração de Trend.

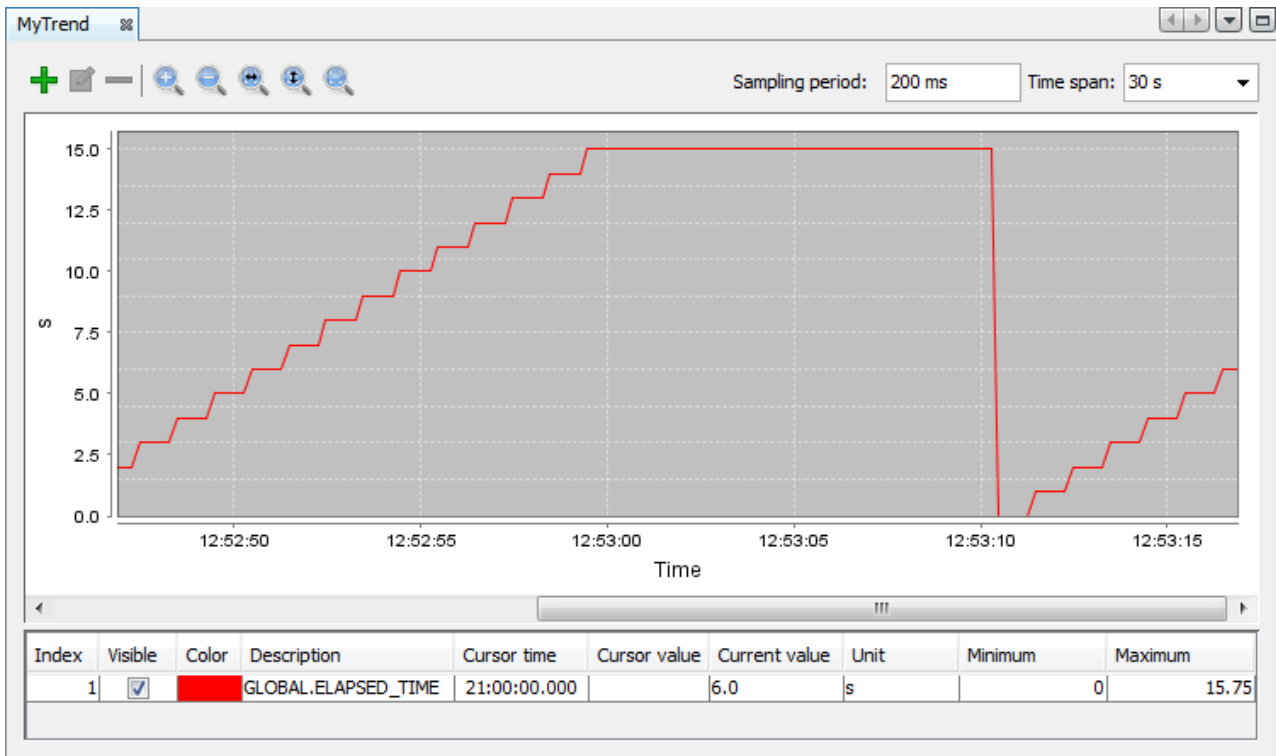
1. Criação de um novo arquivo Trend.



2. Adição e configuração dos canais no botão  no canto superior esquerdo.



3. Após adicionar o canal, basta clicar em **Conectar Dispositivo**  que o Trend já se iniciará automaticamente.



É importante certificar-se de que o recurso esteja corretamente carregado no equipamento para evitar leituras incorretas.

10 Equipamentos

[PLC300](#)

[SCA06](#)

[CFW300](#)

10.1 PLC300

[Descrição](#)

[Novidades e Correções do PLC300](#)

[I/O's](#)

[Marcadores de Sistema](#)

[Ladder](#)

[Tela](#)

[Log de Eventos](#)

[Setup](#)

[Comunicação](#)

10.1.1 Descrição

O PLC300 é um CLP com HMI integrada, desenvolvido para atender a necessidade de interface com o usuário em painéis e máquinas.

Ao mesmo tempo, ele é um completo CLP expansível, rápido e com várias portas de comunicação, possibilitando que o produto seja mestre de redes CANopen, Modbus RTU (rede RS-485) e Modbus TCP (rede Ethernet).

Descrição dos Modelos

O PLC300 está disponível em seis modelos diferentes, em que o usuário poderá personalizar a aparência do PLC, seguindo especificações presentes no CD do produto.

A identificação do PLC300 é feita por um sufixo de duas ou três letras. As letras têm o seguinte significado:

- **H:** HMI instalada
- **B:** Blind (sem HMI)
- **P:** Plus
- **S:** Standard
- **C:** Custom (sem membrana)

Especificações	PLC300HP	PLC300BP	PLC300HS	PLC300BS	PLC300HPC	PLC300HSC
Código WEG	12358193	12358194	12358192	12358195	12358196	12358197
IHM	✓	✗	✓	✗	✓	✓
Membrana	✓	✗	✓	✗	✗	✗
Ethernet	✓	✓	✗	✗	✓	✗
Encoder	✓	✓	✗	✗	✓	✗
Expansão	✓	✓	✗	✗	✓	✗
SD Card	✓	✓	✓	✓	✓	✓
CAN	✓	✓	✓	✓	✓	✓
RS 485	✓	✓	✓	✓	✓	✓
RS 232	✓	✓	✓	✓	✓	✓
USB	✓	✓	✓	✓	✓	✓
Saída PWM	✓	✓	✓	✓	✓	✓
I/O Digital	✓	✓	✓	✓	✓	✓
I/O Analógico	✓	✓	✓	✓	✓	✓

Consulte o manual do usuário do equipamento PLC300 para mais informações sobre o produto.

10.1.2 Novidades e Correções do PLC300

PLC300 V3.00

Novas funcionalidades:

- Aumento de memória interna para 1MB.

Correções de desvio funcional:

- Após o download de firmware o alarme do watchdog era ativado.

PLC300 V2.40

Correções de desvio funcional:

- Travamento dos blocos Modbus RTU quando utilizados em USERFB. Ao desabilitar o USERFB com um bloco Modbus RTU ativo (que reserva o recurso do mestre modbus) outros blocos Modbus RTU não podiam mais executar;
- Backup geral na versão 2.3x apresentava problemas durante o backup dos arquivos no SD Card; e
- O Watchdog estava sendo ativado em alguns produtos durante a inicialização causando o travamento do produto.

PLC300 V2.30

Novas funcionalidades:

- Hot Download: Implementação que permite a realização de carga a quente do recurso, ou seja, é possível alterar o programa ladder, telas, alarmes e logs com o programa atual rodando e após a conclusão do download o novo programa é executado automaticamente;
- Watchdog: Implementação de um Watchdog configurável pelo usuário com um tempo mínimo de 300ms. O estado das saídas pode ser configurado em caso de Watchdog, bem como uma saída para uso exclusivo;
- Marcadores para monitoração do tempo máximo e mínimo do ciclo de scan; e
- Marcadores para desabilitar teclas HOME, SETUP e ALARM.

Modificações de funções existentes:

- Alteração do valor padrão do encoder de 12V para 5V.
-

PLC300 V2.10

Novas funcionalidades:

- Criado blocos para suporte a Strings:
 - o STR_COMPARE
 - o STR_COPY
 - o STR_COPY_LAST
 - o STR_DELETE
 - o STR_FIND
 - o STR_FIND_LAST
 - o STR_INSERT
 - o STR_LENGTH
 - o STR_REPLACE
 - o DWORD_TO_STRING
 - o REAL_TO_STRING
 - o STRING_TO_DWORD
 - o STRING_TO_REAL
- Criado bloco READENC4, efetua a leitura de encoder, calculando o posicionamento e velocidade do mesmo, permite utilização de filtro;
- Tarefa de evento externo e de contagem utilizando DI1 a DI8;
- Criado bloco TRUNC, realiza truncamento de variáveis;
- Criado bloco ROUND, realiza arredondamento de variáveis;
- Incremento e decremento dos campos de "Numeric Input" utilizando seta (na edição do campo);
- Log de variáveis do tipo STRING;
- Bloco STORE com variáveis do tipo STRING;
- Bloco SEL com variáveis do tipo STRING;
- Bloco ISTORE com variáveis do tipo STRING;
- Bloco ILOAD com variáveis do tipo STRING;
- Utilização de variáveis do tipo STRING nos campos de "Text Input";
- Criação de estruturas e Receitas com variáveis do tipo STRING;

Modificações de funções existentes:

- Endereço base alterado de 3000 para 0.
-

PLC300 V2.00

Novas funcionalidades:

- Criação de bloco P_RAMP, que gera um trem de pulsos com frequência na forma de rampa;
 - Entradas DI9 e DI10 como entrada de encoder;
 - DI10 pode ser usada como contagem rápida, sendo que DI9 determina o sentido de contagem;
 - Mudança na entrada analógica de 10bits para 12bits;
 - Recuperação automática (ASR) do aplicativo, *setup*, *firmware*... através de memória FLASH, não necessitando mais do SD Card para esta funcionalidade.
-

PLC300 V1.70

Novas funcionalidades:

- Criação de um componente do tipo **Text Input**;
 - Criação de uma nova área de memória para preservação do valor das variáveis durante o *download*
-

PLC300 V1.60

Modificações de funções existentes:

- Criação de novas instruções para diminuição do tamanho do programa e ciclo de scan.
-

PLC300 V1.50

Novas funcionalidades:

- Marcadores de Sistema de Status (%S_):
 - o STS_ASR_OCC
 - o STS_INPUT
- Marcadores de Sistema de Comando (%C_):
 - o RS232_TIMEOUT_MS
 - o RS232_RX_END_CHARACTER
 - o RS232_TX_INITIAL_ADDRESS
 - o RS232_RX_INITIAL_ADDRESS
 - o RS232_TX_BUFFER_LENGTH
 - o RS232_MAX_RX_BUFFER_LENGTH
 - o RS232_ENABLE_END_CHARACTER
 - o RS232_START_TX
 - o RS485_TIMEOUT_MS
 - o RS485_RX_END_CHARACTER
 - o RS485_TX_INITIAL_ADDRESS
 - o RS485_RX_INITIAL_ADDRESS
 - o RS485_TX_BUFFER_LENGTH
 - o RS485_MAX_RX_BUFFER_LENGTH

- o RS485_ENABLE_END_CHARACTER
- o RS485_START_TX
- Função de recuperação automática de *Software* (ASR)

Modificações de funções existentes:

- Alteração nos blocos Modbus RTU para suportar a leitura e escrita de até 16 registradores
 - Alteração nos blocos temporizadores (TON, TOF e TP) com configuração da base de tempo (milissegundo, centissegundo, segundo e minuto)
 - Aumento da capacidade do número de telas do usuário
-

PLC300 V1.40

Novas funcionalidades:

- Marcadores de Sistema de Status (%S_):
 - o KEY_NUMERIC
 - o KEY_HOME
 - o KEY_ESC
 - o KEY_DEL
 - o KEY_ALARM
 - o KEY_SETUP
 - o KEY_SHIFT
 - o KEY_UP
 - o KEY_DOWN
 - o KEY_LEFT
 - o KEY_RIGHT
 - o KEY_ENTER
 - o KEY_F1 ... KEY_F12
 - *Upload* de programa
 - Forçar E/S
 - Idioma inglês no PLC300
 - Apresentação de variáveis nas telas de alarme
 - Zeros à esquerda com zeros nos campos **Numeric Input** e **Numeric Output** nas telas de usuário
 - Opção de *download*:
 - o Inicializar variáveis voláteis e de saída
 - o Parar/Iniciar a execução do programa automaticamente
 - Proteção por senha dos comandos de gravação e carregamento dos arquivos de programa, *setup* e *firmware* no SD card.
-

PLC300 V1.30

Novas funcionalidades:

- Marcadores de Sistema de Status (%S_):
 - o BOOTLOADER
 - o INTERVAL_TASK9_WATCHDOG ... INTERVAL_TASK16_WATCHDOG
 - o SINGLE_TASK9_WATCHDOG ... SINGLE_TASK16_WATCHDOG

- o COUNT_TASK9_WATCHDOG ... COUNT_TASK16_WATCHDOG
 - o STS_SD_INVALID
 - Marcadores de Sistema de Comando (%C_):
 - o INTERVAL_TASK9_DISABLE ... INTERVAL_TASK16_DISABLE
 - o SINGLE_TASK9_DISABLE ... SINGLE_TASK16_DISABLE
 - o COUNT_TASK9_DISABLE ... COUNT_TASK9_DISABLE
 - Blocos Modbus TCP
 - o MBTCP_ReadBinary
 - o MBTCP_WriteBinary
 - o MBTCP_ReadRegister
 - o MBTCP_WriteRegister
 - o MBTCP_ServerStatus
 - o MBTCP_ClientControlStatus
 - Novos Blocos Data Transfer (Receitas)
 - o ReadRecipe
 - o WriteRecipe
 - Log de
 - o Alarmes
 - o Eventos
 - Backup no SD Card do
 - o *Firmware*
 - o Recurso
 - o *Setup*
 - Ethernet
 - o Conexão do gateway do WPS com o PLC300
-

PLC300 V1.20

Novas funcionalidades:

- Tarefas
 - o INTERVAL
 - o SINGLE
 - o EXTERN EVENT (DI9, DI10 e pulso Z)
 - o COUNT (DI9, DI10, pulsos A, B, Z e quadratura AB)
 - o SYSTEM (start e stop)
- Marcadores de Sistema de Status (%S_)
 - o TICK_100US
 - o INTERVAL_TASK1_WATCHDOG...INTERVAL_TASK8_WATCHDOG
 - o SINGLE_TASK1_WATCHDOG...SINGLE_TASK8_WATCHDOG
 - o EXT_EVENT_TASK1_WATCHDOG...EXT_EVENT_TASK3_WATCHDOG
 - o COUNT_TASK1_WATCHDOG...COUNT_TASK8_WATCHDOG
 - o MAIN_TASK_WATCHDOG
 - o START_TASK_WATCHDOG
 - o STOP_TASK_WATCHDOG
- Marcadores de Sistema de Comando (%C_)
 - o INTERVAL_TASK1_DISABLE...INTERVAL_TASK8_DISABLE
 - o SINGLE_TASK1_DISABLE...SINGLE_TASK8_DISABLE
 - o EXT_EVENT_TASK1_WATCHDOG...EXT_EVENT_TASK3_WATCHDOG

- o COUNT_TASK1_DISABLE...COUNT_TASK8_DISABLE
- Blocos *Hardware*
 - o IMMEDIATEINPUT
 - o IMMEDIATEOUTPUT
 - o READENC3
- Bloco Coil
 - o IMMEDIATECOIL

Modificações de funções existentes:

- Opção de inicializar ou não variáveis retentivas e histórico de alarmes no *download*.
- Bloco PWM - permitido valor de frequência de 0 Hz
- Blocos ReadEnc e ReadEnc2 - Tipo de dado de Value pode ser DINT quando os pulso contados forem Quadratura_AB, permitindo valores negativos conforme o sentido de giro do encoder.

Correções de desvio funcional:

- Bloco MB_WriteBinary - em versões anteriores, o bloco sempre escrevia o valor 1.
- Alterados os endereços das variáveis para acesso via Modbus.

PLC300 Versões Anteriores a V1.20

Recomendamos fazer atualização do *firmware*.

PLC300 V1.10

Nova funcionalidade:

- Protocolo ASCII RS232
-

PLC300 V1.00

Versão inicial.

10.1.3 I/O's

Nas variáveis abaixo, contidas no grupo **I/O** da tabela de variáveis, o tag não pode ser alterado.

Informações de *hardware* podem ser encontradas no manual do PLC300 através do site www.weg.net.

Entradas Digitais

Endereço	Bit	Modbus	Tag	Descrição
%IB0	0	16000	DI1	Entrada digital 1
%IB0	1	16001	DI2	Entrada digital 2

%IB0	2	16002	DI3	Entrada digital 3
%IB0	3	16003	DI4	Entrada digital 4
%IB0	4	16004	DI5	Entrada digital 5
%IB0	5	16005	DI6	Entrada digital 6
%IB0	6	16006	DI7	Entrada digital 7
%IB0	7	16007	DI8	Entrada digital 8
%IB1	0	16008	DI9	Entrada digital 9
%IB1	1	16009	DI10	Entrada digital 10
%IB2	0	16016	DI101	Entrada digital 1 - Slot 1
%IB2	1	16017	DI102	Entrada digital 2 - Slot 1
%IB2	2	16018	DI103	Entrada digital 3 - Slot 1
%IB2	3	16019	DI104	Entrada digital 4 - Slot 1
%IB2	4	16020	DI105	Entrada digital 5 - Slot 1
%IB2	5	16021	DI106	Entrada digital 6 - Slot 1
%IB2	6	16022	DI107	Entrada digital 7 - Slot 1
%IB2	7	16023	DI108	Entrada digital 8 - Slot 1
%IB3	0	16024	DI109	Entrada digital 9 - Slot 1
%IB3	1	16025	DI110	Entrada digital 10 - Slot 1
%IB3	2	16026	DI111	Entrada digital 11 - Slot 1
%IB3	3	16027	DI112	Entrada digital 12 - Slot 1
%IB3	4	16028	DI113	Entrada digital 13 - Slot 1
%IB3	5	16029	DI114	Entrada digital 14 - Slot 1
%IB3	6	16030	DI115	Entrada digital 15 - Slot 1
%IB3	7	16031	DI116	Entrada digital 16 - Slot 1
%IB4	0	16032	DI201	Entrada digital 1 - Slot 2
%IB4	1	16033	DI202	Entrada digital 2 - Slot 2
%IB4	2	16034	DI203	Entrada digital 3 - Slot 2
%IB4	3	16035	DI204	Entrada digital 4 - Slot 2
%IB4	4	16036	DI205	Entrada digital 5 - Slot 2
%IB4	5	16037	DI206	Entrada digital 6 - Slot 2
%IB4	6	16038	DI207	Entrada digital 7 - Slot 2
%IB4	7	16039	DI208	Entrada digital 8 - Slot 2
%IB5	0	16040	DI209	Entrada digital 9 - Slot 2
%IB5	1	16041	DI210	Entrada digital 10 - Slot 2
%IB5	2	16042	DI211	Entrada digital 11 - Slot 2
%IB5	3	16043	DI212	Entrada digital 12 - Slot 2
%IB5	4	16044	DI213	Entrada digital 13 - Slot 2
%IB5	5	16045	DI214	Entrada digital 14 - Slot 2
%IB5	6	16046	DI215	Entrada digital 15 - Slot 2
%IB5	7	16047	DI216	Entrada digital 16 - Slot 2

Entradas Analógicas

Endereço	Bit	Modbus	Tag	Descrição
%IW6	--	5003	AI1	Entrada analógica 1
%IW8	--	5004	AI101	Entrada analógica 1 - Slot 1
%IW10	--	5005	AI102	Entrada analógica 2 - Slot 1
%IW12	--	5006	AI103	Entrada analógica 3 - Slot 1
%IW14	--	5007	AI104	Entrada analógica 4 - Slot 1
%IW16	--	5008	AI105	Entrada analógica 5 - Slot 1
%IW18	--	5009	AI201	Entrada analógica 1 - Slot 2
%IW20	--	5010	AI202	Entrada analógica 2 - Slot 2
%IW22	--	5011	AI203	Entrada analógica 3 - Slot 2
%IW24	--	5012	AI204	Entrada analógica 4 - Slot 2
%IW26	--	5013	AI205	Entrada analógica 5 - Slot 2

Saídas Digitais

Endereço	Bit	Modbus	Tag	Descrição
%QB0	0	16000	DO1	Saída digital 1
%QB0	1	16001	DO2	Saída digital 2
%QB0	2	16002	DO3	Saída digital 3
%QB0	3	16003	DO4	Saída digital 4
%QB0	4	16004	DO5	Saída digital 5
%QB0	5	16005	DO6	Saída digital 6
%QB0	6	16006	DO7	Saída digital 7
%QB0	7	16007	DO8	Saída digital 8
%QB1	0	16008	DO9	Saída digital 9
%QB2	0	16016	DO101	Saída digital 1 - Slot 1
%QB2	1	16017	DO102	Saída digital 2 - Slot 1
%QB2	2	16018	DO103	Saída digital 3 - Slot 1
%QB2	3	16019	DO104	Saída digital 4 - Slot 1
%QB2	4	16020	DO105	Saída digital 5 - Slot 1
%QB2	5	16021	DO106	Saída digital 6 - Slot 1
%QB2	6	16022	DO107	Saída digital 7 - Slot 1
%QB2	7	16023	DO108	Saída digital 8 - Slot 1
%QB3	0	16024	DO109	Saída digital 9 - Slot 1
%QB3	1	16025	DO110	Saída digital 10 - Slot 1
%QB3	2	16026	DO111	Saída digital 11 - Slot 1
%QB3	3	16027	DO112	Saída digital 12 - Slot 1
%QB3	4	16028	DO113	Saída digital 13 - Slot 1
%QB3	5	16029	DO114	Saída digital 14 - Slot 1
%QB3	6	16030	DO115	Saída digital 15 - Slot 1
%QB3	7	16031	DO116	Saída digital 16 - Slot 1
%QB4	0	16032	DO201	Saída digital 1 - Slot 2
%QB4	1	16033	DO202	Saída digital 2 - Slot 2
%QB4	2	16034	DO203	Saída digital 3 - Slot 2
%QB4	3	16035	DO204	Saída digital 4 - Slot 2
%QB4	4	16036	DO205	Saída digital 5 - Slot 2
%QB4	5	16037	DO206	Saída digital 6 - Slot 2
%QB4	6	16038	DO207	Saída digital 7 - Slot 2
%QB4	7	16039	DO208	Saída digital 8 - Slot 2
%QB5	0	16040	DO209	Saída digital 9 - Slot 2
%QB5	1	16041	DO210	Saída digital 10 - Slot 2
%QB5	2	16042	DO211	Saída digital 11 - Slot 2
%QB5	3	16043	DO212	Saída digital 12 - Slot 2
%QB5	4	16044	DO113	Saída digital 13 - Slot 2
%QB5	5	16045	DO214	Saída digital 14 - Slot 2
%QB5	6	16046	DO215	Saída digital 15 - Slot 2
%QB5	7	16047	DO216	Saída digital 16 - Slot 2

Saídas Analógicas

Endereço	Bit	Modbus	Tag	Descrição
%QW6	--	5003	AO1	Saída analógica 1
%QW8	--	5004	AO101	Saída analógica 1 - Slot 1
%QW10	--	5005	AO102	Saída analógica 2 - Slot 1
%QW12	--	5006	AO201	Saída analógica 1 - Slot 2
%QW14	--	5007	AO202	Saída analógica 2 - Slot 2

10.1.4 Marcadores de Sistema

As variáveis abaixo, contidas no grupo **GLOBAL_SYSTEM** da tabela de variáveis, possuem o tag fixo.

Alguns marcadores do tipo BYTE (%SB ou %CB), estão alocados no mesmo endereço Modbus para registradores. Neste caso:

- (L): indica que é o byte menos significativo da WORD;
- (H): indica que é o byte mais significativo da WORD.



NOTA!

A partir da versão de firmware 2.10, o endereço base foi alterado de 3000 para 0.

Marcadores de Sistema de Leitura (Status)

Endereço	Bit	Modbus	Tag	Descrição
Ladder				
%SB0004	0	32	FALSE	O seu valor sempre permanecerá 0 (zero)
%SB0004	1	33	TRUE	O seu valor sempre permanecerá 1 (um)
%SB0004	2	34	FREQ_2HZ	Oscila a 2 Hz
%SB0004	3	35	PULSE_1SCAN	Setado por um ciclo de scan na transição de STOP para RUN do programa do usuário
%SB0004	4	36	ENC_DIR	Sentido de giro do encoder: 0 - Horário e 1 - Anti-horário
Estado PLC300				
%SW0000	--	3000	FIRMWARE	Versão de Firmware do PLC300
%SW0002	--	3001	SCAN_CYCLE	Tempo do ciclo de scan em ms com uma casa decimal (ex.: 123 = 12.3ms)
%SB0006	0	48	STS_BAT	Alarme ativo - Bateria Fraca - Trocar bateria
%SB0006	1	49	STS_DOS	Alarme ativo - Falha DOS - Indica sobre temperatura nas saídas digitais
%SB0006	2	50	STS_ENC	Alarme Ativo - Falha Encoder - Indica que algum sinal do encoder está faltando
%SB0006	3	51	STS_A11	Alarme Ativo - Fio Partido A11 - Corrente abaixo de 2mA na entrada analógica A11, no modo 4 a 20mA

%SB0006	4	52	STS_SD_INVALID	SD card inexistente ou inválido.
%SB0006	5	53	STS_DOS_SLOT1	Falha saída digital slot 1
%SB0006	6	54	STS_DOS_SLOT2	Falha saída digital slot 2
%SB0006	7	55	STS_ASR_OCC	Ocorreu recuperação automática do software (ASR)
%SD0008	--	3004	ENC_FREQ	Frequência do Encoder - Frequência em Hz do Encoder
%SD0012	--	3006	TICK_100US	Marcador incrementado a cada 100 microssegundos
%SW0016	--	3008	BOOTLOADER	Versão do módulo de atualização de firmware
%SB0540	0	3270	WATCHDOG_OCC	Ocorreu watchdog
%SW0542	--	3271	WATCHDOG_DATE	Data do watchdog
%SD0556	--	3278	WATCHDOG_CODE	Código do watchdog
Telas e Teclas da IHM				
%SB0020	--	3010	SCREEN	Tela - Mostra o número da tela atual na IHM
%SB0022	--	3011	KEY_NUMERIC	Código ASCII da tecla numérica
%SB0024	0	192	KEY_HOME	Tecla HOME pressionada
%SB0024	1	193	KEY_ESC	Tecla ESC pressionada
%SB0024	2	194	KEY_DEL	Tecla DEL pressionada
%SB0024	3	195	KEY_ALARM	Tecla ALARM pressionada
%SB0024	4	196	KEY_SETUP	Tecla SETUP pressionada
%SB0024	5	197	KEY_SHIFT	Tecla SHIFT pressionada
%SB0024	6	198	KEY_UP	Tecla UP pressionada
%SB0024	7	199	KEY_DOWN	Tecla DOWN pressionada
%SB0025	0	200	KEY_LEFT	Tecla LEFT pressionada
%SB0025	1	201	KEY_RIGHT	Tecla RIGHT pressionada
%SB0025	2	202	KEY_ENTER	Tecla ENTER pressionada
%SB0025	3	203	STS_INPUT	Edição do elemento 'Numeric Input' na tela
%SB0026	0	208	KEY_F1	Tecla F1 pressionada
%SB0026	1	209	KEY_F2	Tecla F2 pressionada
%SB0026	2	210	KEY_F3	Tecla F3 pressionada
%SB0026	3	2011	KEY_F4	Tecla F4 pressionada
%SB0026	4	212	KEY_F5	Tecla F5 pressionada
%SB0026	5	213	KEY_F6	Tecla F6 pressionada
%SB0026	6	214	KEY_F7	Tecla F7 pressionada
%SB0026	7	215	KEY_F8	Tecla F8 pressionada
%SB0027	0	216	KEY_F9	Tecla F9 pressionada
%SB0027	1	217	KEY_F10	Tecla F10 pressionada
%SB0027	2	218	KEY_F11	Tecla F11 pressionada
%SB0027	3	219	KEY_F12	Tecla F12 pressionada
Relógio de Tempo Real (RTC)				
%SW0030	--	3015	HOUR	Hora do RTC
%SW0032	--	3016	MINUTE	Minuto do RTC
%SW0034	--	3017	SECOND	Segundo do RTC
%SW0036	--	3018	DAY	Dia do RTC

%SW0038	--	3019	MONTH	Mês do RTC
%SW0040	--	3020	YEAR	Ano do RTC
%SW0042	--	3021	WEEKDAY	Dia da semana: 0 = domingo, 1 = segunda ... 6 = sábado
Watchdog de tarefas				
%SB0050	0	400	INTERVAL_TASK1_WATCHDOG	Watchdog tarefa 1 por tempo
%SB0050	1	401	INTERVAL_TASK2_WATCHDOG	Watchdog tarefa 2 por tempo
%SB0050	2	402	INTERVAL_TASK3_WATCHDOG	Watchdog tarefa 3 por tempo
%SB0050	3	403	INTERVAL_TASK4_WATCHDOG	Watchdog tarefa 4 por tempo
%SB0050	4	404	INTERVAL_TASK5_WATCHDOG	Watchdog tarefa 5 por tempo
%SB0050	5	405	INTERVAL_TASK6_WATCHDOG	Watchdog tarefa 6 por tempo
%SB0050	6	406	INTERVAL_TASK7_WATCHDOG	Watchdog tarefa 7 por tempo
%SB0050	7	407	INTERVAL_TASK8_WATCHDOG	Watchdog tarefa 8 por tempo
%SB0051	0	408	INTERVAL_TASK9_WATCHDOG	Watchdog tarefa 9 por tempo
%SB0051	1	409	INTERVAL_TASK10_WATCHDOG	Watchdog tarefa 10 por tempo
%SB0051	2	410	INTERVAL_TASK11_WATCHDOG	Watchdog tarefa 11 por tempo
%SB0051	3	411	INTERVAL_TASK12_WATCHDOG	Watchdog tarefa 12 por tempo
%SB0051	4	412	INTERVAL_TASK13_WATCHDOG	Watchdog tarefa 13 por tempo
%SB0051	5	413	INTERVAL_TASK14_WATCHDOG	Watchdog tarefa 14 por tempo
%SB0051	6	414	INTERVAL_TASK15_WATCHDOG	Watchdog tarefa 15 por tempo
%SB0051	7	415	INTERVAL_TASK16_WATCHDOG	Watchdog tarefa 16 por tempo
%SB0052	0	416	SINGLE_TASK1_WATCHDOG	Watchdog tarefa 1 por evento
%SB0052	1	417	SINGLE_TASK2_WATCHDOG	Watchdog tarefa 2 por evento
%SB0052	2	418	SINGLE_TASK3_WATCHDOG	Watchdog tarefa 3 por evento
%SB0052	3	419	SINGLE_TASK4_WATCHDOG	Watchdog tarefa 4 por evento
%SB0052	4	420	SINGLE_TASK5_WATCHDOG	Watchdog tarefa 5 por evento
%SB0052	5	421	SINGLE_TASK6_WATCHDOG	Watchdog tarefa 6 por evento
%SB0052	6	422	SINGLE_TASK7_WATCHDOG	Watchdog tarefa 7 por evento
%SB0052	7	423	SINGLE_TASK8_WATCHDOG	Watchdog tarefa 8 por evento
%SB0053	0	424	SINGLE_TASK9_WATCHDOG	Watchdog tarefa 9 por evento
%SB0053	1	425	SINGLE_TASK10_WATCHDOG	Watchdog tarefa10 por evento
%SB0053	2	426	SINGLE_TASK11_WATCHDOG	Watchdog tarefa 11 por evento
%SB0053	3	427	SINGLE_TASK12_WATCHDOG	Watchdog tarefa 12 por evento
%SB0053	4	428	SINGLE_TASK13_WATCHDOG	Watchdog tarefa 13 por evento
%SB0053	5	429	SINGLE_TASK14_WATCHDOG	Watchdog tarefa 14 por evento
%SB0053	6	430	SINGLE_TASK15_WATCHDOG	Watchdog tarefa 15 por evento
%SB0053	7	431	SINGLE_TASK16_WATCHDOG	Watchdog tarefa 16 por evento
%SB0054	0	3027	EXT_EVENT_TASK1_WATCHDOG	Watchdog tarefa 1 por evento externo
%SB0054	1	3027	EXT_EVENT_TASK2_WATCHDOG	Watchdog tarefa 2 por evento externo
%SB0054	2	3027	EXT_EVENT_TASK3_WATCHDOG	Watchdog tarefa 3 por evento externo
%SB0056	0	3028	COUNT_TASK1_WATCHDOG	Watchdog tarefa 1 por contagem
%SB0056	1	3028	COUNT_TASK2_WATCHDOG	Watchdog tarefa 2 por contagem
%SB0056	2	3028	COUNT_TASK3_WATCHDOG	Watchdog tarefa 3 por contagem

%SB0056	3	3028	COUNT_TASK4_WATCHDOG	Watchdog tarefa 4 por contagem
%SB0056	4	3028	COUNT_TASK5_WATCHDOG	Watchdog tarefa 5 por contagem
%SB0056	5	3028	COUNT_TASK6_WATCHDOG	Watchdog tarefa 6 por contagem
%SB0056	6	3028	COUNT_TASK7_WATCHDOG	Watchdog tarefa 7 por contagem
%SB0056	7	3028	COUNT_TASK8_WATCHDOG	Watchdog tarefa 8 por contagem
%SB0057	0	3028	COUNT_TASK9_WATCHDOG	Watchdog tarefa 9 por contagem
%SB0057	1	3028	COUNT_TASK10_WATCHDOG	Watchdog tarefa 10 por contagem
%SB0057	2	3028	COUNT_TASK11_WATCHDOG	Watchdog tarefa 11 por contagem
%SB0057	3	3028	COUNT_TASK12_WATCHDOG	Watchdog tarefa 12 por contagem
%SB0057	4	3028	COUNT_TASK13_WATCHDOG	Watchdog tarefa 13 por contagem
%SB0057	5	3028	COUNT_TASK14_WATCHDOG	Watchdog tarefa 14 por contagem
%SB0057	6	3028	COUNT_TASK15_WATCHDOG	Watchdog tarefa 15 por contagem
%SB0057	7	3028	COUNT_TASK16_WATCHDOG	Watchdog tarefa 16 por contagem
%SB0058	0	3029	MAIN_TASK_WATCHDOG	Watchdog tarefa MAIN
%SB0058	1	3029	START_TASK_WATCHDOG	Watchdog tarefa START
%SB0058	2	3029	STOP_TASK_WATCHDOG	Watchdog tarefa STOP
RS232				
%SW0068	--	3034	RS232_TX_TELEGRAM_COUNTER	Contador de telegramas enviados
%SW0070	--	3035	RS232_RX_TELEGRAM_COUNTER	Contador de telegramas recebidos
%SW0072	--	3036	RS232_RX_BYTE_COUNTER	Contador de bytes recebidos por telegrama
%SB0078	0	624	RS232_TIMEOUT	Timeout na resposta
%SB0078	1	625	RS232_RX_TELEGRAM_FINISHED	Término de recebimento de telegrama
%SB0078	2	626	RS232_TX_TELEGRAM_FINISHED	Término de envio de telegrama
%SB0114	0	912	RS232_RX_FINISHED	Recepção ASCII concluída
RS485				
%SW0084	--	3042	RS485_TX_TELEGRAM_COUNTER	Contador de telegramas enviados
%SW0086	--	3043	RS485_RX_TELEGRAM_COUNTER	Contador de telegramas recebidos
%SW0088	--	3044	RS485_RX_BYTE_COUNTER	Contador de bytes recebidos por telegrama
%SB0094	0	752	RS485_TIMEOUT	Timeout na resposta
%SB0094	1	753	RS485_RX_TELEGRAM_FINISHED	Término de recebimento de telegrama
%SB0094	2	754	RS485_TX_TELEGRAM_FINISHED	Término de envio de telegrama
Modbus RTU				
%SB0100	--	3050	MBUS_INTERFACE_DISABLED	Interface do mestre Modbus RTU desabilitada.
%SW0102	--	3051	MBUS_REQUEST_COUNT	Contador de requisições feitas pelo mestre Modbus RTU
%SW0104	--	3052	MBUS_RESPONSE_COUNT	Contador de respostas recebidas com sucesso pelo mestre Modbus RTU
%SW0106	--	3053	MBUS_NO_ANSWER_COUNT	Contador de requisições sem resposta recebida pelo mestre Modbus RTU (timeout)
%SW0108	--	3054	MBUS_RESP_ERROR_COUNT	Contador de respostas de erro recebidas pelo mestre Modbus RTU
%SB0110	--	3055 (L)	MBUS_LAST_ERROR_ADDR	Último erro detectado: endereço do escravo Modbus RTU
%SB0111	--	3055 (H)	MBUS_LAST_ERROR_TYPE	Último erro detectado: tipo de erro

%SB0112	--	3056	MBUS_LAST_ERROR_CODE	Último erro detectado: código do erro
%SW0120	--	3060	MBUS_SLAVE_REQUEST_COUNT	Contador de telegramas recebidos com sucesso pelo PLC300 como escravo Modbus RTU
%SW0122	--	3061	MBUS_SLAVE_RESPONSE_COUNT	Contador de respostas transmitidas com sucesso pelo PLC300 como escravo Modbus RTU
CAN				
%SB0150	--	3075	CAN_STATUS	Estado da interface CAN
%SB0151	0	1208	CAN_BUS_POWER	Alimentação da interface CAN
%SW0152	--	3076	CAN_RX_COUNTER	Contador de telegramas CAN recebidos
%SW0154	--	3077	CAN_TX_COUNTER	Contador de telegramas CAN transmitidos
%SW0156	--	3078	CAN_BUS_OFF_COUNTER	Contador de erros de bus off detectados
%SW0158	--	3079	CAN_OVERRUN_COUNTER	Contador de telegramas CAN perdidos (overrun)
CANopen				
%SB0180	--	3090 (L)	CO_STATUS	Estado da comunicação CANopen
%SB0181	--	3090 (H)	CO_NODE_STATE	Estado do nó CANopen (pré-operacional, operacional, parado)
%SB0200	0	1600	CO_STS_MASTER_CONTACTED	Mestre CANopen contactou todos os escravos
%SB0200	1	1601	CO_STS_MASTER_CONFIG_OK	Mestre CANopen fez download das configurações dos escravos
%SB0200	2	1602	CO_STS_MASTER_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/ heartbeat) iniciado com os escravos
%SB0200	3	1603	CO_STS_MASTER_INIT_FINISHED	Inicialização de todos os escravos realizada pelo mestre CANopen
%SB0200	4	1604	CO_STS_MASTER_INIT_ERROR	Algum escravo apresentou erro de inicialização
%SB0200	5	1605	CO_STS_MASTER_ERROR_CTRL	Detectado falha em algum escravo no protocolo de detecção de erros pelo mestre CANopen
%SB0200	6	1606	CO_STS_MASTER_EMCY	Algum escravo reportou EMCY
%SB0201	0	1608	CO_STS_MASTER_NMT_TOGGLE	Feedback do toggle bit do comando NMT
%SB0206	5	1613	CO_STS_MASTER_BUS_OFF	Mestre CANopen está em bus off
%SB0201	6	1614	CO_STS_MASTER_POWER_OFF	Mestre CANopen não possui alimentação na interface CAN
%SB0201	7	1615	CO_STS_MASTER_COMM_DISABLED	Comunicação do mestre CANopen está desabilitada
%SB0202	0	1616	CO_STS_SLAVE1_CONTACTED	Escravo no endereço indicado foi contactado com sucesso pelo mestre CANopen
%SB0202	1	1617	CO_STS_SLAVE1_CONFIG_OK	Configuração do escravo foi feita com sucesso pelo mestre CANopen
%SB0202	2	1618	CO_STS_SLAVE1_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/ heartbeat) iniciado com o escravo
%SB0202	3	1619	CO_STS_SLAVE1_INIT_FINISHED	Inicialização do escravo concluída
%SB0202	4	1620	CO_STS_SLAVE1_INIT_ERROR	Erro na inicialização do escravo no endereço indicado
%SB0202	5	1621	CO_STS_SLAVE1_ERROR_CTRL_FAIL	Detectado falha em algum escravo no protocolo de detecção de erro pelo mestre CANopen
%SB0202	6	1632	CO_STS_SLAVE1_EMCY	Escravo no endereço indicado reportou erro por EMCY
%SB0204	0	1633	CO_STS_SLAVE2_CONTACTED	Escravo no endereço indicado foi contactado com sucesso pelo mestre CANopen

%SB0204	1	1634	CO_STS_SLAVE2_CONFIG_OK	Configuração do escravo foi feita com sucesso pelo mestre CANopen
%SB0204	2	1635	CO_STS_SLAVE2_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/ heartbeat) iniciado com o escravo
%SB0204	3	1636	CO_STS_SLAVE2_INIT_FINISHED	Inicialização do escravo concluída
%SB0204	4	1637	CO_STS_SLAVE2_INIT_ERROR	Erro na inicialização do escravo no endereço indicado
%SB0204	5	1638	CO_STS_SLAVE2_ERROR_CTRL_FAIL	Detectado falha em algum escravo no protocolo de detecção de erro pelo mestre CANopen
%SB0204	6	1639	CO_STS_SLAVE2_EMCY	Escravo no endereço indicado reportou erro por EMCY
...		...	...	...
%SB0454	0	3632	CO_STS_SLAVE127_CONTACTED	Escravo no endereço indicado foi contactado com sucesso pelo mestre CANopen
%SB0454	1	3633	CO_STS_SLAVE127_CONFIG_OK	Configuração do escravo foi feita com sucesso pelo mestre CANopen
%SB0454	2	3634	CO_STS_SLAVE127_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/ heartbeat) iniciado com o escravo
%SB0454	3	3635	CO_STS_SLAVE127_INIT_FINISHED	Inicialização do escravo concluída
%SB0454	4	3636	CO_STS_SLAVE127_INIT_ERROR	Erro na inicialização do escravo no endereço indicado
%SB0454	5	3637	CO_STS_SLAVE127_ERROR_CTRL_FAIL	Detectado falha em algum escravo no protocolo de detecção de erro pelo mestre CANopen
%SB0454	6	3638	CO_STS_SLAVE127_EMCY	Escravo no endereço indicado reportou erro por EMCY
%SW0460	--	3230	CO_SDO_ERROR_NODE_ID	Erro de SDO: endereço do escravo do último erro de SDO detectado
%SW0462	--	3231	CO_SDO_ERROR_OBJECT_INDEX	Erro de SDO: índice do objeto
%SW0464	--	3232	CO_SDO_ERROR_OBJECT_SUBINDEX	Erro de SDO: sub-índice do objeto
%SW0466	--	3233	CO_SDO_ERROR_FUNCTION	Erro de SDO: função (leitura/escrita)
%SW0468	--	3234	CO_SDO_ERROR_VALUE	Erro de SDO: valor
%SW0472	--	3236	CO_SDO_ERROR_CODE	Erro de SDO: código do erro
%SW0480	--	3240	CO_EMCY_SLAVE_ID	Último EMCY reportado: endereço do escravo
%SW0482	--	3241	CO_EMCY_DATA	Último EMCY reportado: dados do objeto
Ethernet				
%SB0492	--	3246	ETH_MAC[6]	Endereço físico
%SB0498	--	3249	ETH_STS_SPD_DUP	Modo de comunicação Ethernet
%SD0500	--	3250	ETH_STS_IP	Endereço IP
%SD0504	--	3252	ETH_STS_MASK	Máscara de sub-rede
%SD0508	--	3254	ETH_STS_GW	Gateway padrão
Modbus TCP				
%SW0512	--	3256	MBTCP_SERVER_REQUEST_COUNT	Contador de telegramas recebidos com sucesso pelo PLC300 como servidor Modbus TCP
%SW0514	--	3257	MBTCP_SERVER_RESPONSE_COUNT	Contador de respostas transmitidas com sucesso pelo PLC300 como servidor Modbus TCP*
%SW0516	--	3258	MBTCP_SERVER_CNXNS	Número de conexões do servidor Modbus TCP ativas
%SB0520	--	3260	MBTCP_CLIENT_DISABLED	Cliente Modbus TCP desabilitado
%SW0522	--	3261	MBTCP_REQUEST_COUNT	Contador de requisições feitas pelo cliente Modbus

				TCP
%SW0524	--	3262	MBTCP_RESPONSE_COUNT	Contador de respostas recebidas com sucesso pelo cliente Modbus TCP
%SW0526	--	3263	MBTCP_NO_ANSWER_COUNT	Contador de requisições sem resposta recebida pelo cliente Modbus TCP (Timeout)
%SW0528	--	3264	MBTCP_RESP_ERROR_COUNT	Contador de respostas de erro recebidas pelo cliente Modbus TCP
%SW0530	--	3265	MBTCP_LAST_ERROR_TCP_PORT	Último erro detectado: porta TCP do servidor Modbus
%SD0532	--	3266	MBTCP_LAST_ERROR_IP	Último erro detectado: endereço IP do servidor modbus TCP
%SB0536	--	3268 (L)	MBTCP_LAST_ERROR_UNITID	Último erro detectado: Unit ID do servidor Modbus TCP
%SB0537	--	3268 (H)	MBTCP_LAST_ERROR_TYPE	Último erro detectado: tipo de erro
%SB0538	--	3269	MBTCP_LAST_ERROR_CODE	Último erro detectado: código do erro

Marcadores de Sistema de Escrita / Leitura (Comando)

Endereço	Bit	Modbus	Tag	Descrição
PLC300				
%CB0000	0	0	ERASE_RET	Zera Retentivos - Zera marcadores retentivos na transição positiva
%CB0000	1	2	CLEAR_ENC_ALARM	Limpa alarme de Falha de Encoder
%CB0000	2	3	BUZZER_ACTIVE	Buzzer ativo
Relógio de Tempo Real (RTC)				
%CW0030	--	3015	WR_HOUR	Hora do RTC - Ajuste da Hora do RTC
%CW0032	--	3016	WR_MINUTE	Minuto do RTC - Ajuste do Minuto do RTC
%CW0034	--	3017	WR_SECOND	Segundo do RTC - Ajuste do Segundo do RTC
%CW0036	--	3018	WR_DAY	Dia do RTC - Ajuste do Dia do RTC
%CW0038	--	3019	WR_MONTH	Mês do RTC - Ajuste do Mês do RTC
%CW0040	--	3020	WR_YEAR	Ano do RTC - Ajuste do Ano do RTC
CAN				
%CB0052	--	3026	CAN_ADDRESS	Endereço do PLC300 para interface CAN
%CB0055	--	3027	CAN_BAUDRATE	Taxa de comunicação para interface CAN
RS232				
%CB0061	--	3030	RS232_MODE	Modo de operação da interface RS232: 0=Modbus RTU 2=ASCII
%CB0062	--	3031 (L)	RS232_BYTE_FORMAT	Paridade e stop bits para interface RS232
%CB0063	--	3031 (H)	RS232_BAUDRATE	Taxa de comunicação para interface RS232
%CB0114	0	912	RS232_RX_CLEAR	Inicia nova recepção ASCII via RS232
%CB0124	--	3062	RS232_ASCII_STRING	Buffer de recepção do protocolo ASCII
%CB0124	--	3062	RS232_ASCII_BYTEBUFFER	Buffer de recepção do protocolo ASCII
RS485				
%CB0068	--	3034 (L)	RS485_ADDRESS	Endereço do PLC300 para interface RS485
%CB0069	--	3034 (H)	RS485_MODE	Modo de operação (mestre ou escravo) para interface RS485
%CB0070	--	3035 (L)	RS485_BYTE_FORMAT	Paridade e stop bits para interface RS485
%CB0071	--	3035 (H)	RS485_BAUDRATE	Taxa de comunicação para interface RS485
Modbus RTU				
%CW0100	--	3050	MBUS_DISABLE_COMM	Desabilita interface do mestre Modbus RTU
CANopen				
%CB0120	--	3060	CO_NMT_COMMAND	Envio de comando NMT pelo mestre CANopen: código do comando
%CB0121	0	968	CO_NMT_TOGGLE	Envio de comando NMT pelo mestre CANopen: toggle bit
%CB0121	7	975	CO_DISABLE	Desabilita comunicação CANopen
%CB0122	--	3061	CO_NMT_SLAVE_ADDR	Envio de comando NMT pelo mestre CANopen: endereço do escravo
Interrupções de tempo, evento e contagem				
%CB0400	0	3200	INTERVAL_TASK1_DISABLE	Desabilita interrupção 1 de tempo
%CB0400	1	3201	INTERVAL_TASK2_DISABLE	Desabilita interrupção 2 de tempo
%CB0400	2	3202	INTERVAL_TASK3_DISABLE	Desabilita interrupção 3 de tempo
%CB0400	3	3203	INTERVAL_TASK4_DISABLE	Desabilita interrupção 4 de tempo
%CB0400	4	3204	INTERVAL_TASK5_DISABLE	Desabilita interrupção 5 de tempo
%CB0400	5	3205	INTERVAL_TASK6_DISABLE	Desabilita interrupção 6 de tempo
%CB0400	6	3206	INTERVAL_TASK7_DISABLE	Desabilita interrupção 7 de tempo
%CB0400	7	3207	INTERVAL_TASK8_DISABLE	Desabilita interrupção 8 de tempo
%CB0401	0	3208	INTERVAL_TASK9_DISABLE	Desabilita interrupção 9 de tempo
%CB0401	1	3209	INTERVAL_TASK10_DISABLE	Desabilita interrupção 10 de tempo
%CB0401	2	3210	INTERVAL_TASK11_DISABLE	Desabilita interrupção 11 de tempo

10.1.5 Ladder

[Componentes e Blocos](#)

[Tarefas](#)

[Estruturas](#)

[Receitas](#)

10.1.5.1 Componentes e Blocos

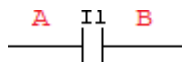
Ver a seguir.

10.1.5.1.1 Contact

10.1.5.1.1.1 NOCONTACT

Contato normalmente aberto.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	I1	BOOL	Entrada de controle do bloco

Funcionamento

Quando a variável I1 está com o valor FALSE, B recebe FALSE.

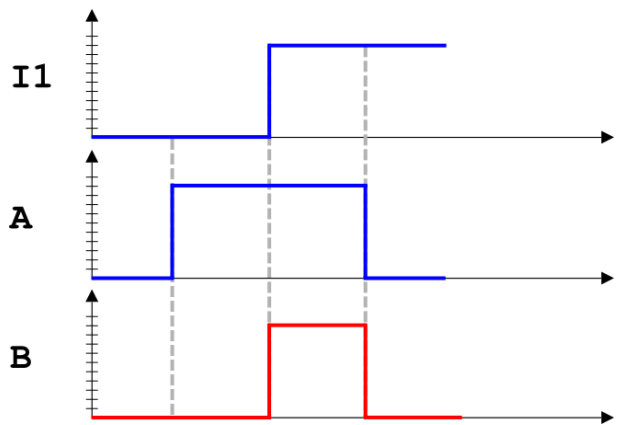
Quando a variável I1 está com o valor TRUE, B recebe o valor de A.



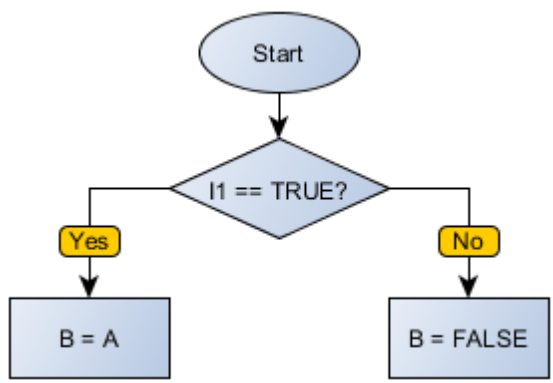
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

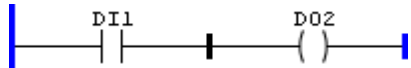
Diagrama



Fluxograma do Bloco



Exemplo

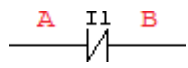


O exemplo acima realiza a transferência do valor da entrada digital DI1 para a saída digital DO2.

10.1.5.1.1.2 NCCONTACT

Contato normalmente fechado.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	I1	BOOL	Entrada de controle do bloco

Funcionamento

Quando a variável I1 está com o valor TRUE, B recebe FALSE.

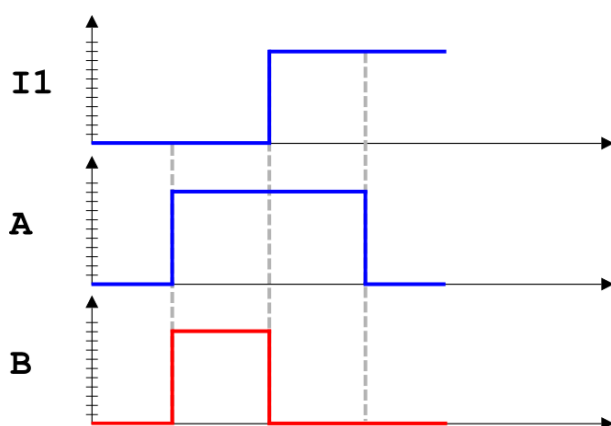
Quando a variável I1 está com o valor FALSE, B recebe o valor de A.



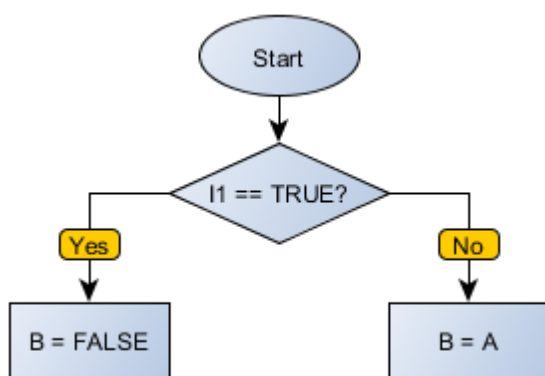
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

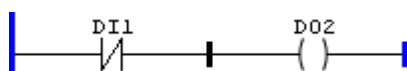
Diagrama



Fluxograma do Bloco



Exemplo

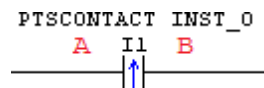


O exemplo acima realiza a transferência do oposto do valor da entrada digital DI1 para a saída digital DO2.

10.1.5.1.1.3 PTSCONTACT

Contato de transição de borda de subida.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	I1	BOOL	Entrada de controle do bloco
VAR	PTSCONTACT_INST_0	PTSCONTACT	Instância de acesso à estrutura do bloco

Funcionamento

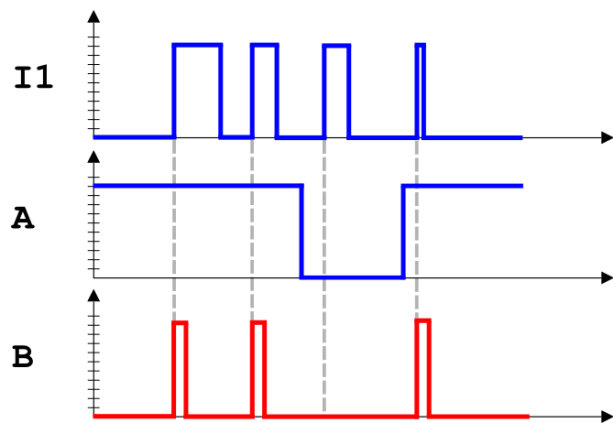
No instante em que a variável I1 transita de FALSE para TRUE (borda de subida ou borda de transição positiva), B recebe o valor de A por um ciclo de scan.
Em todos os outros momentos, B recebe o valor FALSE.



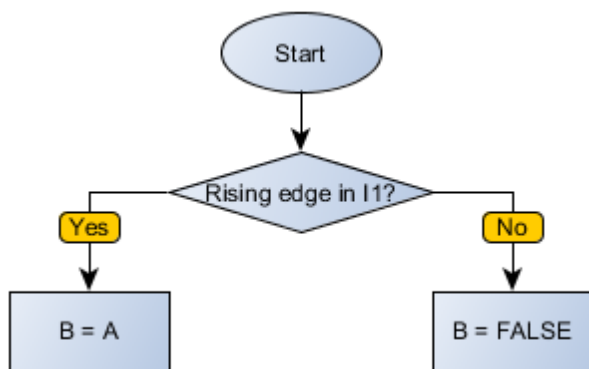
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

Diagrama



Fluxograma do Bloco



Exemplo



O exemplo acima reseta a saída digital DO1 caso a tecla SHIFT esteja pressionada e seja dado um pulso positivo na entrada digital DI2.

10.1.5.1.1.4 NTSCONTACT

Contato de transição de borda de descida.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	I1	BOOL	Entrada de controle do bloco
VAR	NTSCONTACT_INST_0	NTSCONTACT	Instância de acesso à estrutura do bloco

Funcionamento

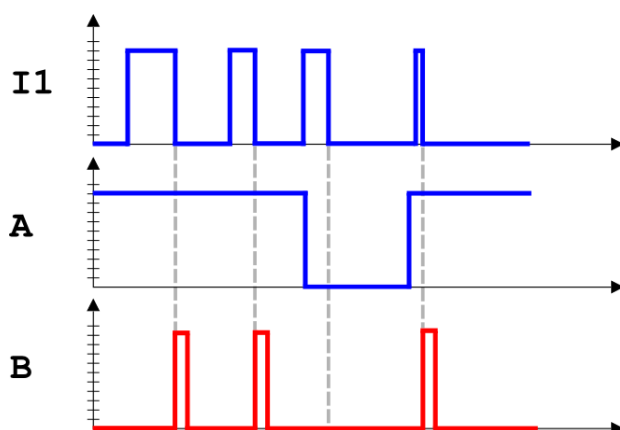
No instante em que a variável I1 transita de TRUE para FALSE (borda de descida ou borda de transição negativa), B recebe o valor de A por um ciclo de scan. Em todos os outros momentos, B recebe o valor FALSE.



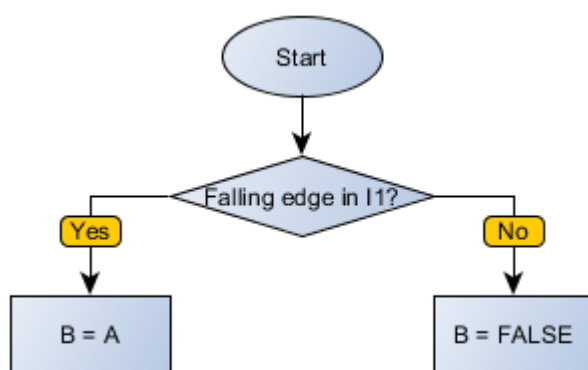
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

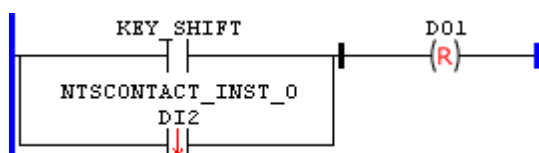
Diagrama



Fluxograma do Bloco



Exemplo



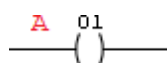
O exemplo acima reseta a saída digital DO1 caso a tecla SHIFT esteja pressionada ou seja dado um pulso negativo na entrada digital DI2.

10.1.5.1.2 Coil

10.1.5.1.2.1 DIRECTCOIL

Bloco lógico utilizado para atribuição de valores diretos a variáveis de saída.

Representação Ladder



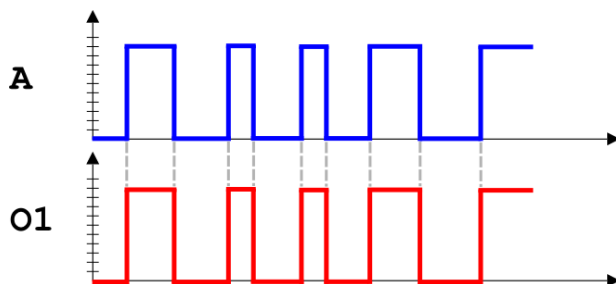
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco

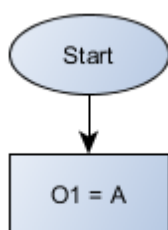
Funcionamento

Este bloco repassa o valor de A para o endereço de memória correspondente a O1.

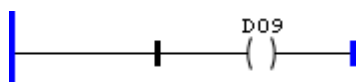
Diagrama



Fluxograma do Bloco



Exemplo

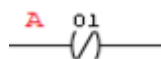


O exemplo acima deixa a saída digital DO9 permanentemente ligada, pois o valor de A, neste caso, é o valor do barramento esquerdo, que é sempre considerado nível lógico alto (TRUE).

10.1.5.1.2.2 INVERTEDCOIL

Bloco lógico utilizado para atribuição de valores negados a variáveis de saída.

Representação Ladder



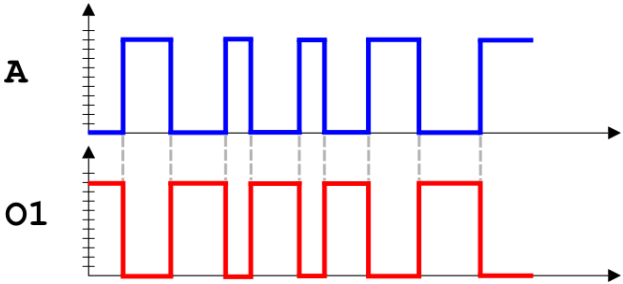
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco

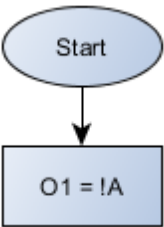
Funcionamento

Este bloco repassa o valor negado de A para o endereço de memória correspondente a O1.

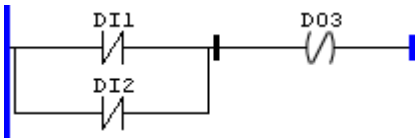
Diagrama



Fluxograma do Bloco



Exemplo

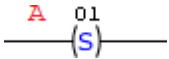


O exemplo acima desativa a saída digital DO3 quando algumas das entradas digitais DI1 ou DI2 estão com valor FALSE. Quando ambas entradas estão com valor TRUE, DO3 ativa.

10.1.5.1.2.3 SETCOIL

Bloco lógico utilizado para habilitação indefinida de variáveis de saída.

Representação Ladder



Estrutura do Bloco

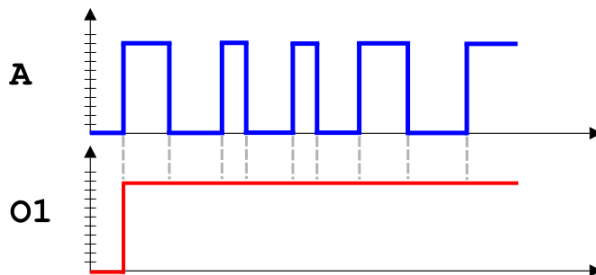
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

Funcionamento

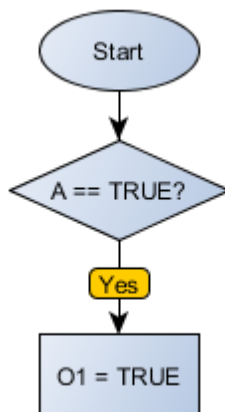
Ao identificar um valor TRUE em A, este bloco repassa um valor TRUE para o endereço de memória correspondente a O1.

Ao identificar um valor FALSE em A, este bloco não realiza nenhuma operação.

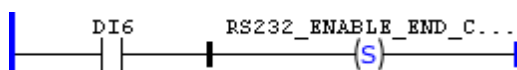
Diagrama



Fluxograma do Bloco



Exemplo

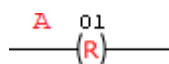


O exemplo acima ativa permanentemente o marcador de controle do sistema habilita caractere de fim de mensagem na comunicação RS232 ao identificar um nível TRUE na entrada digital DI6.

10.1.5.1.2.4 RESETCOIL

Bloco lógico utilizado para desabilitação indefinida de variáveis de saída.

Representação Ladder



Estrutura do Bloco

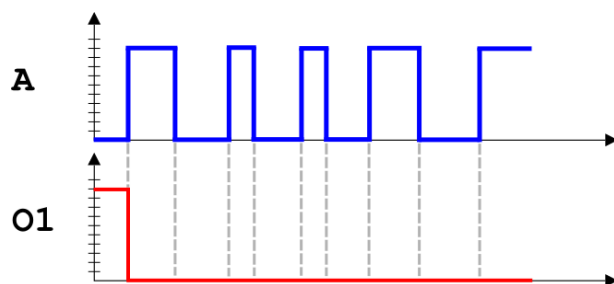
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco

Funcionamento

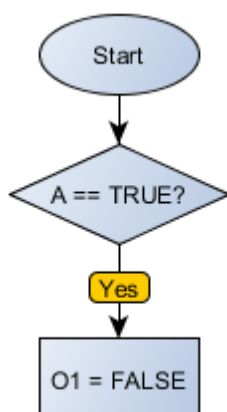
Ao identificar um valor TRUE em A, este bloco repassa um valor FALSE para o endereço de memória correspondente a O1.

Ao identificar um valor FALSE em A, este bloco não realiza nenhuma operação.

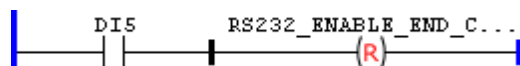
Diagrama



Fluxograma do Bloco



Exemplo

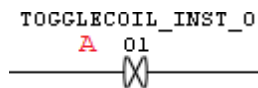


O exemplo acima desativa permanentemente o marcador de controle do sistema que habilita caractere de fim de mensagem na comunicação RS232 ao identificar um nível TRUE na entrada digital DI5.

10.1.5.1.2.5 TOGGLECOIL

Bloco lógico utilizado para alternância de variáveis de saída.

Representação Ladder



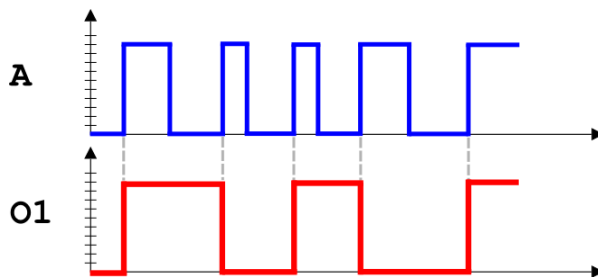
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco
VAR	TOGGLECOIL_INST_0	TOGGLECOIL	Instância de acesso à estrutura do bloco

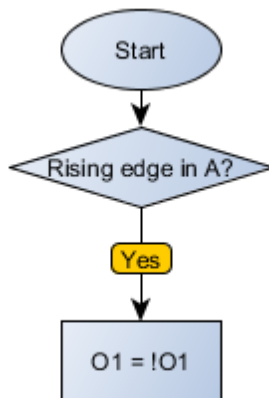
Funcionamento

Ao identificar uma transição de FALSE para TRUE (borda de subida) em A, o bloco inverte o estado de O1.

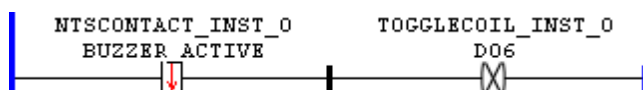
Diagrama



Fluxograma do Bloco



Exemplo

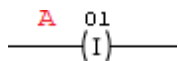


O exemplo acima inverte o estado da saída digital DO6 a cada desativação do buzzer interno.

10.1.5.1.2.6 IMMEDIATECOIL

Bloco lógico utilizado para atribuição de valores a saídas digitais padrão instantaneamente.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco

Funcionamento

Este bloco repassa o valor de A para a saída digital correspondente a O1. Diferentemente da bobina direta, este bloco não aguarda até o fim do ciclo de scan para que o valor da saída seja atualizado; isto é feito no mesmo momento em que o bloco é ativado.

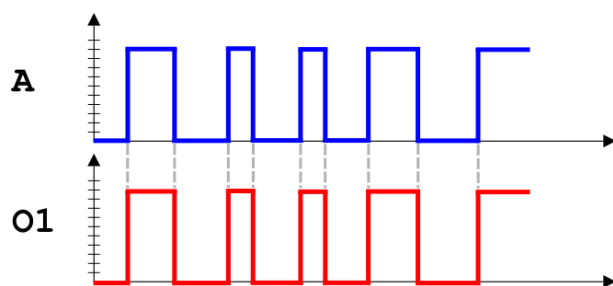
**NOTA!**

Este bloco apenas funciona com as saídas digitais padrão do produto.

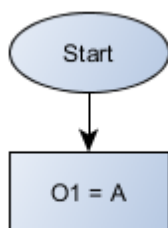
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

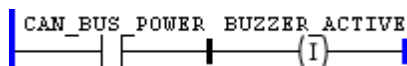
Diagrama



Fluxograma do Bloco



Exemplo



O exemplo acima ativa imediatamente o buzzer interno assim que detectar que a alimentação do barramento da rede CANopen foi interrompida, permanecendo ligado até que a alimentação seja restaurada.

10.1.5.1.3 Logic Bit

10.1.5.1.3.1 RESETBIT

Bloco lógico utilizado para realizar reset de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será alterado
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	DONE	BOOL	Sucesso da operação

Funcionamento

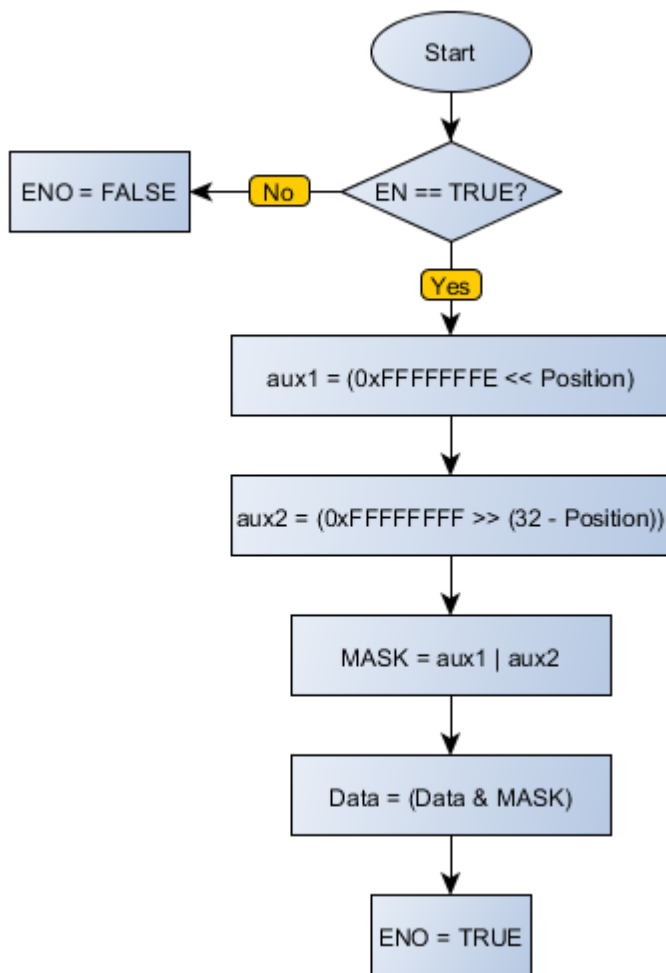
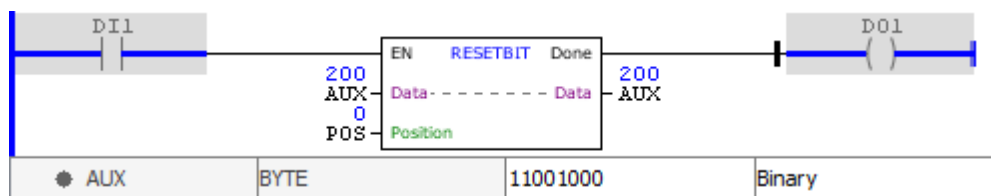
Este bloco, quando tem um valor TRUE em EN, reseta o bit indicado em Position na variável Data, que é repassada para a saída já com o seu valor atualizado.

Quando EN possui valor FALSE, Data permanece inalterado.

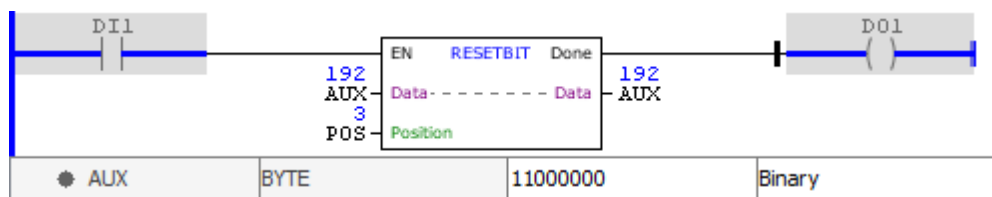
A variável DONE recebe o mesmo valor de EN, exceto quando há erro no reset do bit, recebendo então um valor FALSE.

**NOTA!**

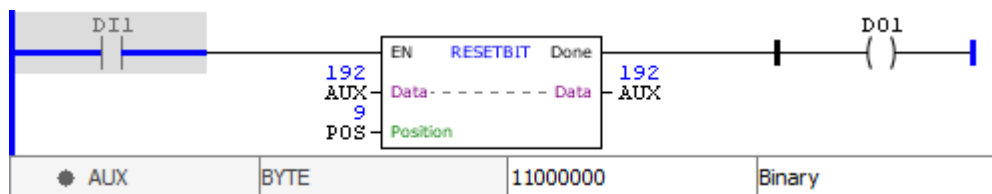
É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

Fluxograma do Bloco**Exemplo**

O exemplo acima reseta o bit da posição zero de AUX, cujo valor inicial é 200 (1100 1000, em binário). Como este bit já possuía valor FALSE, nada foi alterado.



O exemplo acima reseta o bit da posição três de AUX, alterando seu valor binário e, consequentemente, sua representação decimal.

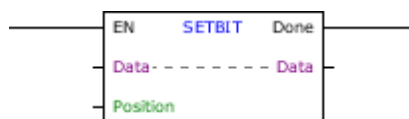


O exemplo acima reseta o bit da posição nove de AUX. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída não é habilitada.

10.1.5.1.3.2 SETBIT

Bloco lógico utilizado para realizar set de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será alterado
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	DONE	BOOL	Sucesso da operação

Funcionamento

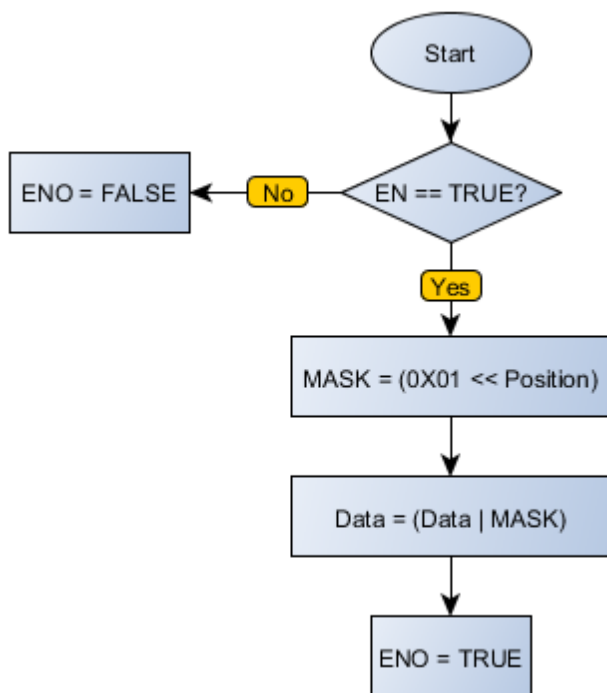
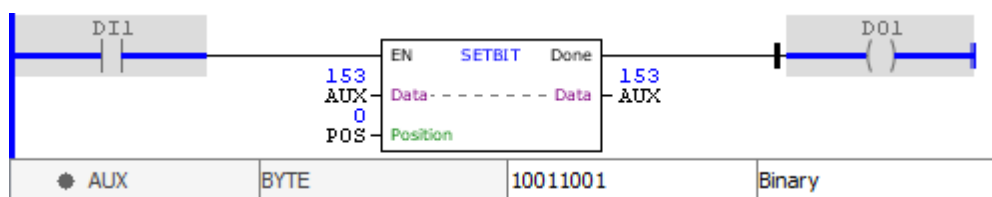
Este bloco, quando tem um valor TRUE em EN, seta o bit indicado em Position na variável Data, que é repassada para a saída já com o seu valor atualizado.

Quando EN possui valor FALSE, Data permanece inalterado.

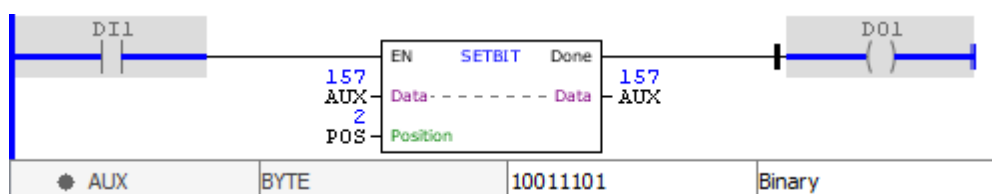
A variável DONE recebe o mesmo valor de EN, exceto quando há erro no set do bit, recebendo então um valor FALSE.

**NOTA!**

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

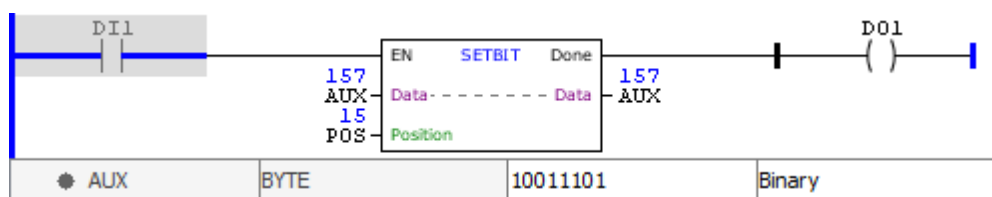
Fluxograma do Bloco**Exemplo**

O exemplo acima seta o bit da posição zero de AUX, cujo valor inicial é 153 (1001 1001, em binário). Como este bit já possuía valor TRUE, nada foi alterado.



O exemplo acima seta o bit da posição dois de AUX, alterando seu valor binário e,

consequentemente, sua representação decimal.



O exemplo acima seta o bit da posição quinze de AUX. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída não é habilitada.

10.1.5.1.3.3 TESTBIT

Bloco lógico que retorna o valor de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será testado
	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	Q	BOOL	Valor do bit testado

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor do bit indicado em Position na variável Data.

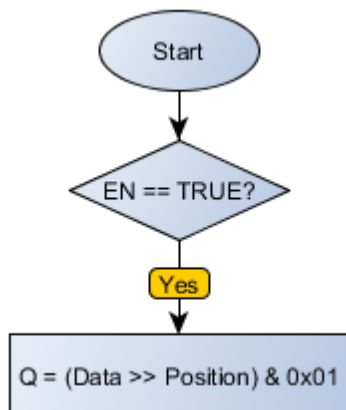
Quando EN possui valor FALSE, Q também recebe FALSE.



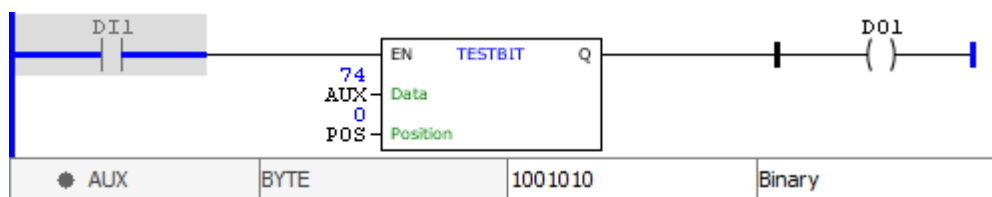
NOTA!

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

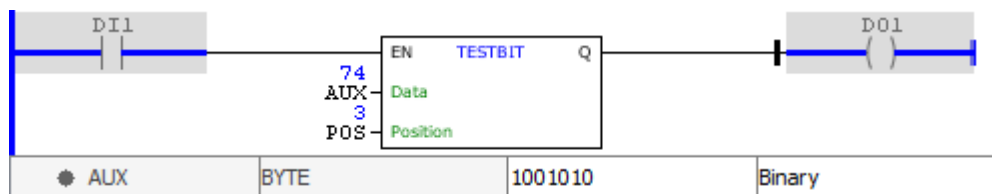
Fluxograma do Bloco



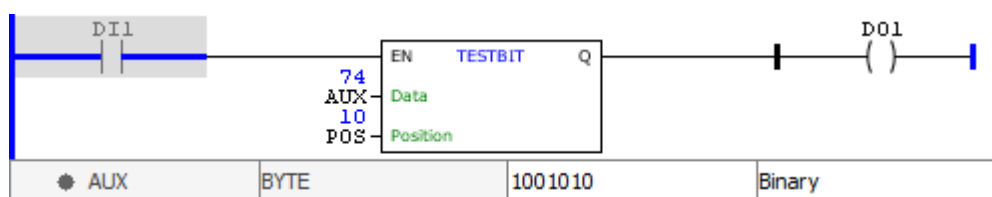
Exemplo



O exemplo acima joga o valor do bit da posição zero de AUX, cujo valor inicial é 74 (0100 1010 em binário) para a saída Q. Como este bit tem valor 0, a saída fica desabilitada.



O exemplo acima joga o valor do bit da posição três de AUX para a saída Q. Como este bit tem valor 1, a saída fica habilitada.



O exemplo acima joga o valor do bit da posição dez de AUX para a saída Q. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída fica desabilitada.

10.1.5.1.4 Logic Boolean

10.1.5.1.4.1 NOT

Bloco lógico que executa uma operação de negado booleano em uma variável, armazenando o

resultado em outra.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável de referência para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

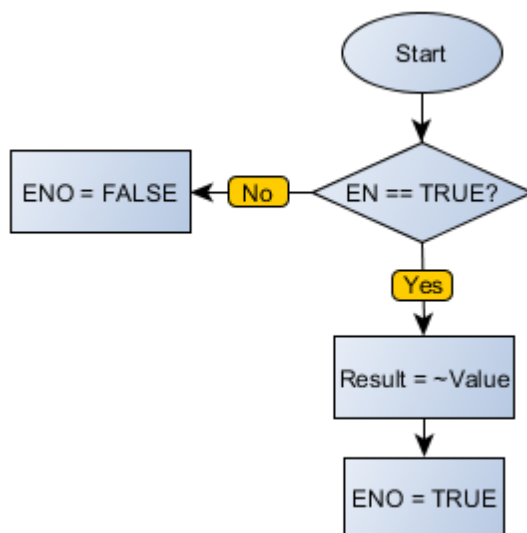
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor booleano negado da variável de entrada Value.

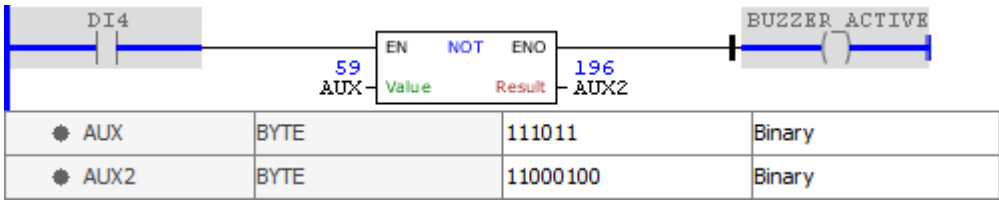
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação negado booleano em AUX, armazenando o resultado em AUX2.

10.1.5.1.4.2 AND

Bloco lógico que executa uma operação de “e” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

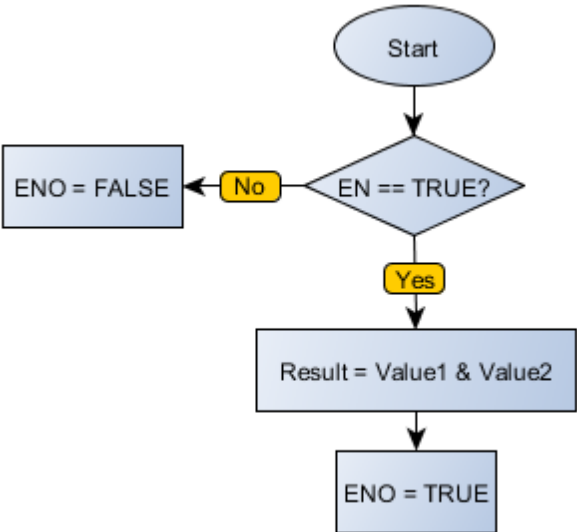
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “e” das variáveis de entrada Value1 e Value2.

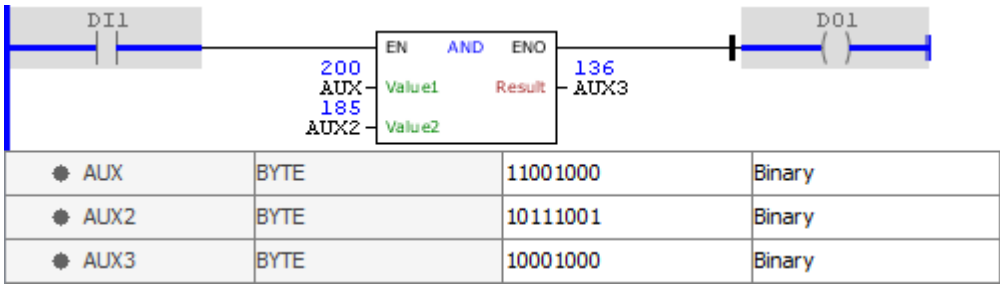
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação e booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.1.5.1.4.3 OR

Bloco lógico que executa uma operação de “ou” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

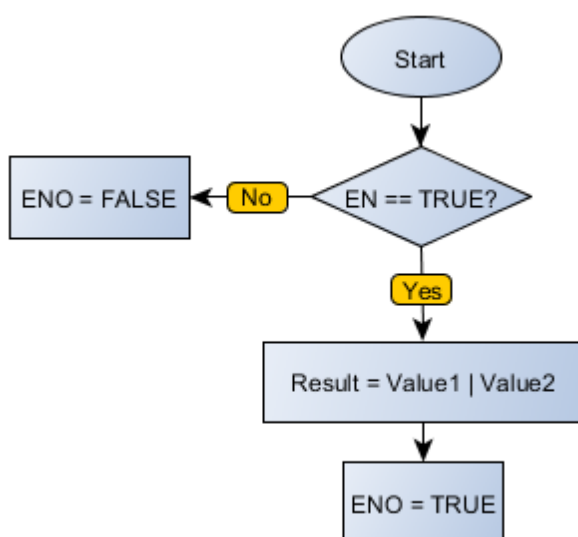
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou” das variáveis de entrada Value1 e Value2.

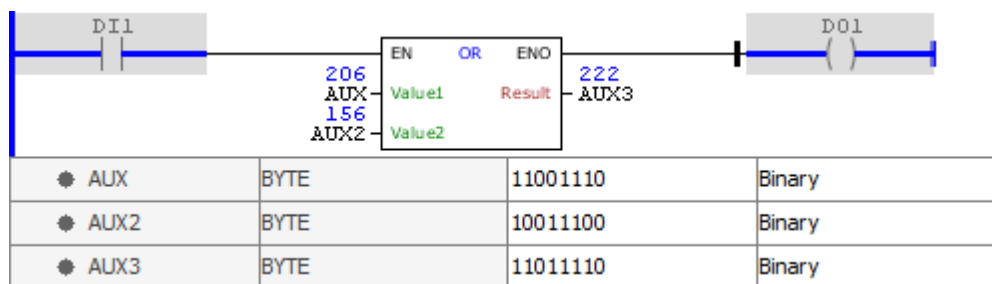
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação ou booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.1.5.1.4.4 XOR

Bloco lógico que executa uma operação de “ou exclusivo” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

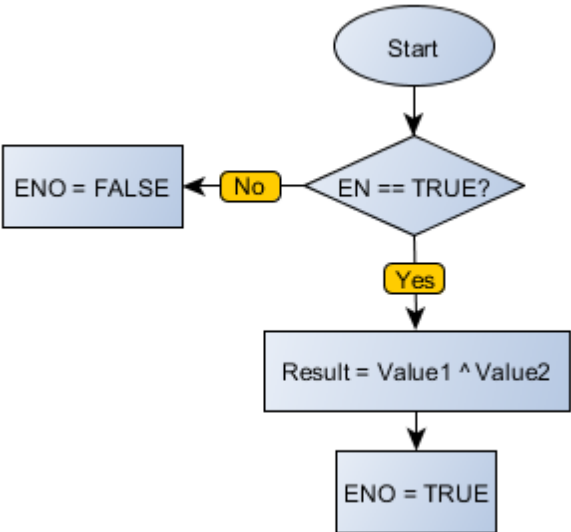
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou exclusivo” das variáveis de entrada Value1 e Value2.

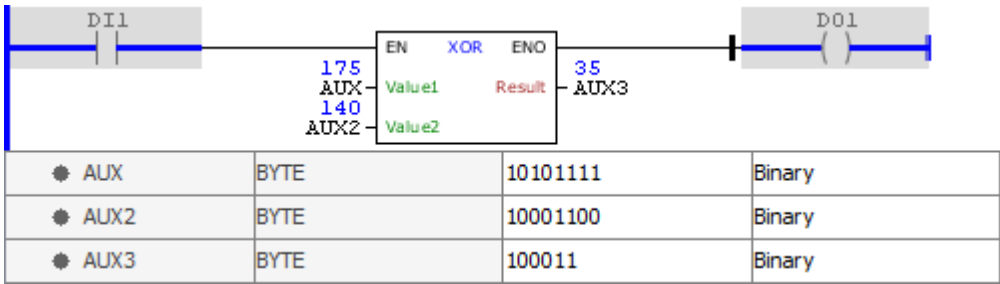
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação ou exclusivo booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.1.5.1.4.5 XNOR

Bloco lógico que executa uma operação de “ou exclusivo negado” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

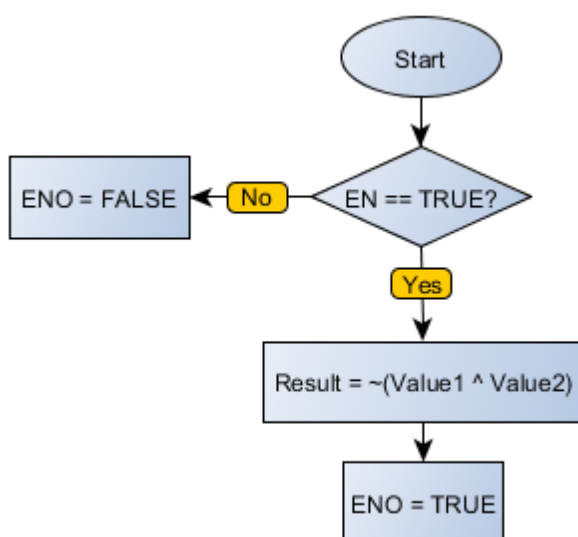
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou exclusivo negado” das variáveis de entrada Value1 e Value2.

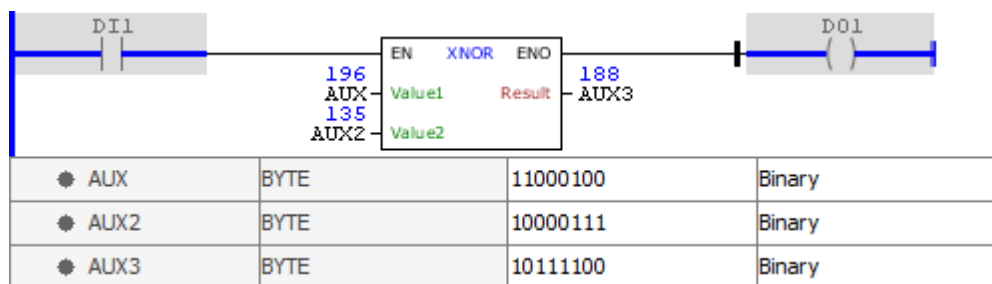
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação ou exclusivo negado booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.1.5.1.5 Logic Shift

10.1.5.1.5.1 ASHL

Bloco que executa uma operação de deslocamento binário à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	SINT INT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	SINT INT DINT	Variável que armazena o resultado da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos aritméticos à esquerda, de acordo com o valor de Shift.



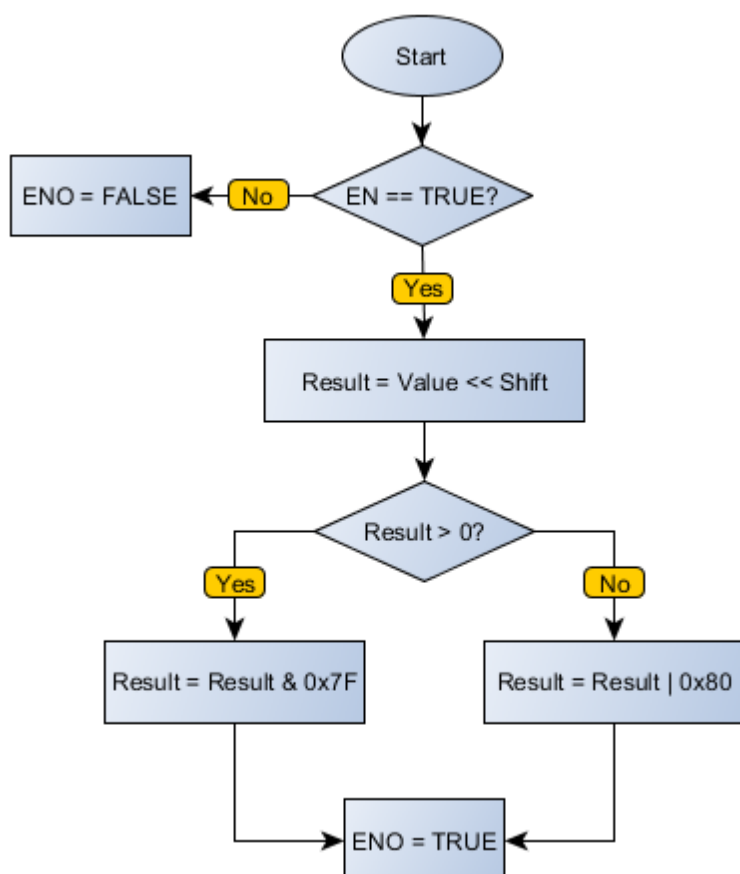
NOTA!

Todos os deslocamentos aritméticos implementados mantém o sinal da variável

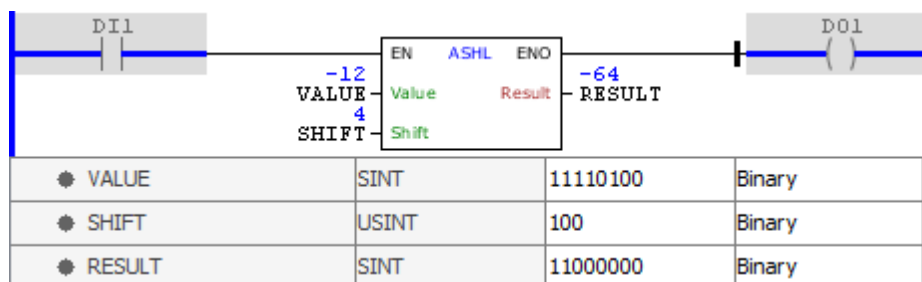
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

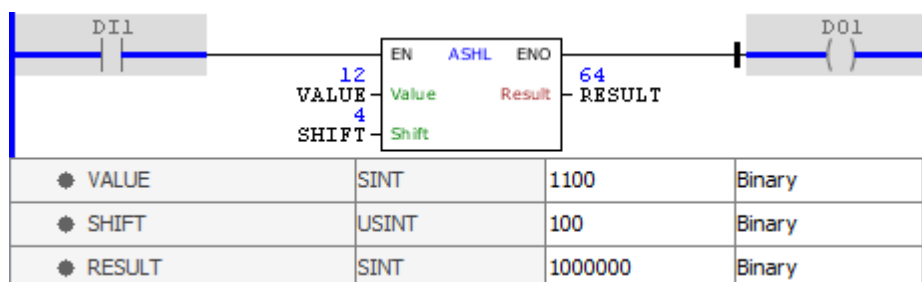
Fluxograma do Bloco



Exemplo



Descrição do exemplo.



Descrição do exemplo.

10.1.5.1.5.2 ASHR

Bloco que executa uma operação de deslocamento aritmético à direita em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	SINT INT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	SINT INT DINT	Variável que armazena o resultado da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos aritméticos à direita, de acordo com o valor de Shift.



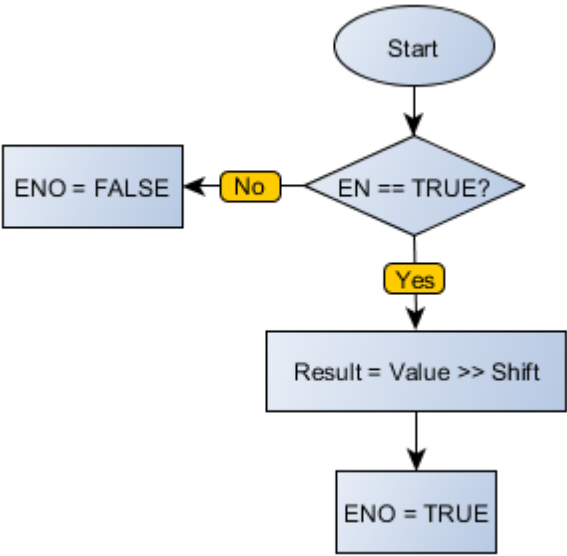
NOTA!

Todos os deslocamentos aritméticos implementados mantém o sinal da variável

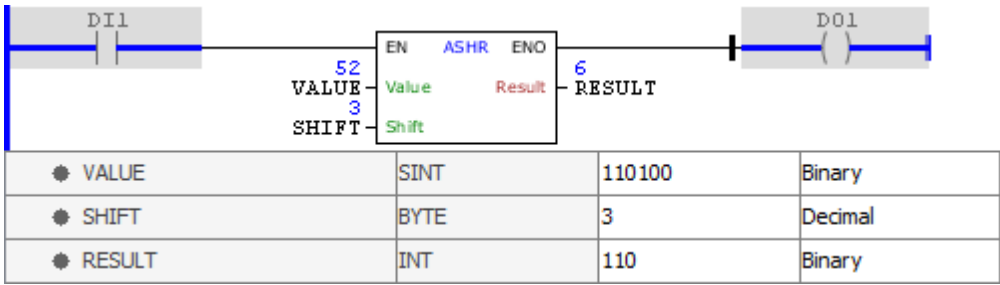
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

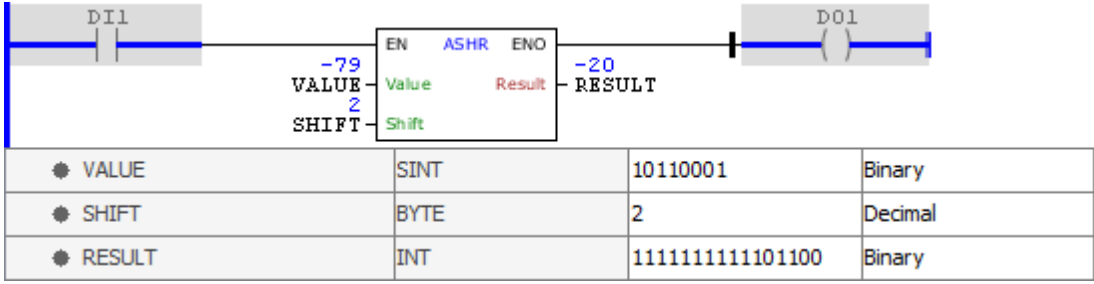
Fluxograma do Bloco



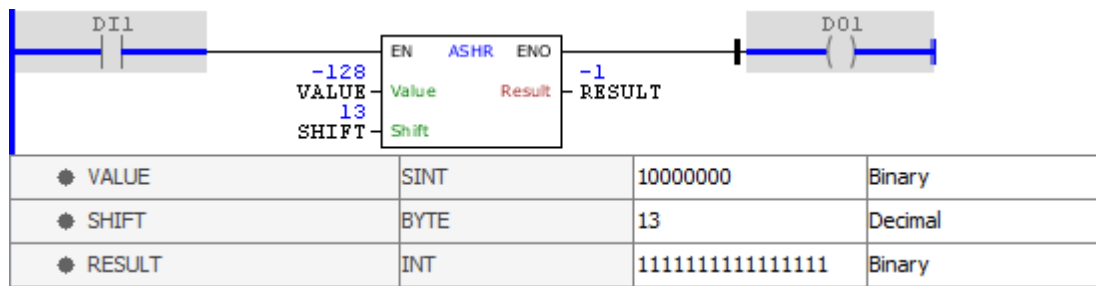
Exemplo



O exemplo acima realiza um deslocamento aritmético à direita de três posições na variável VALUE, cujo valor inicial é 52 (0011 0100 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0000 0110 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento aritmético à direita de duas posições na variável VALUE, cujo valor inicial é -79 (1011 0001 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos uns, visto que os deslocamentos aritméticos à direita preservam o sinal da variável. O resultado final (1111 0110 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento aritmético à direita de treze posições na variável VALUE, cujo valor inicial é -128 (1000 0000 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos uns. O resultado final (1111 1111 em binário) é armazenado em RESULT.

10.1.5.1.5.3 SHL

Bloco que executa uma operação de deslocamento lógico binário à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

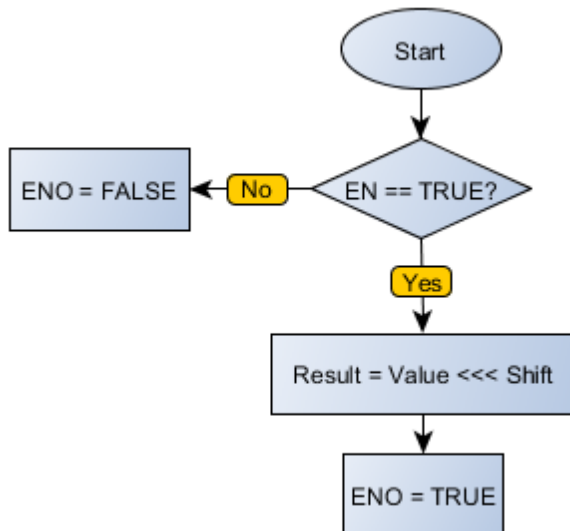
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à esquerda, de acordo com o valor de Shift.

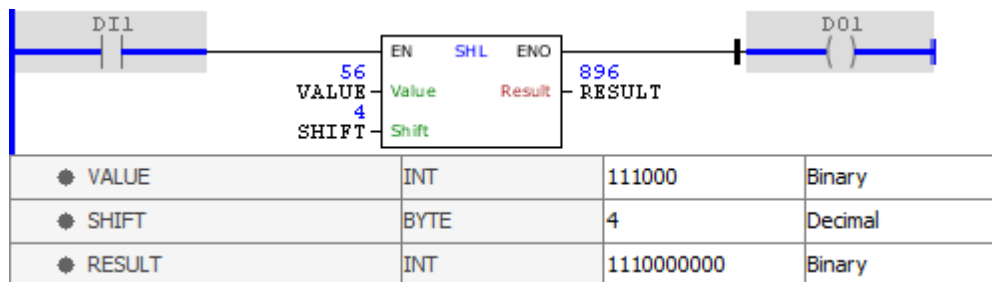
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

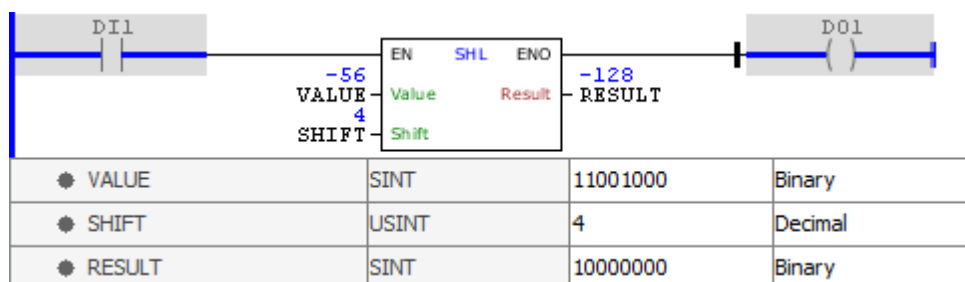
Fluxograma do Bloco



Exemplo



O exemplo acima realiza um deslocamento lógico à esquerda de quatro posições na variável VALUE, cujo valor inicial é 56 (0011 1000 em binário). Os bits à esquerda vão sendo descartados, e à direita são inseridos novos zeros. O resultado final (0011 1000 0000 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento lógico à esquerda de quatro posições na variável VALUE, cujo valor inicial é -56 (1100 1000 em binário). Os bits à esquerda vão sendo descartados, e à direita são inseridos novos zeros. O resultado final (1100 1000 0000 em binário) é armazenado em RESULT. Como RESULT é do tipo SINT, ela só aceita os primeiros oito bits (1000 0000).

10.1.5.1.5.4 SHR

Bloco que executa uma operação de deslocamento lógico binário à direita em um valor passado por

Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

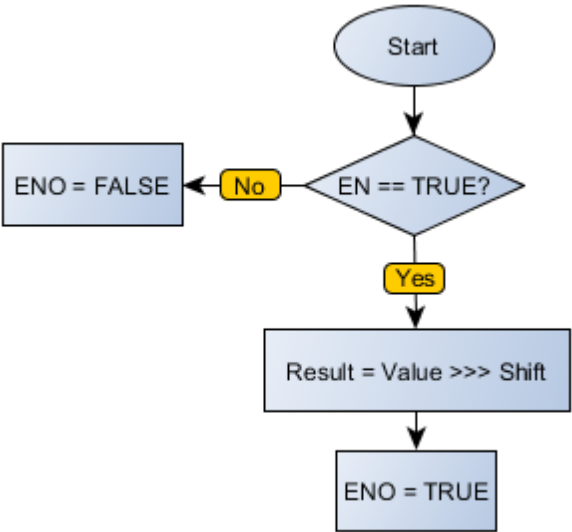
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à direita, de acordo com o valor de Shift.

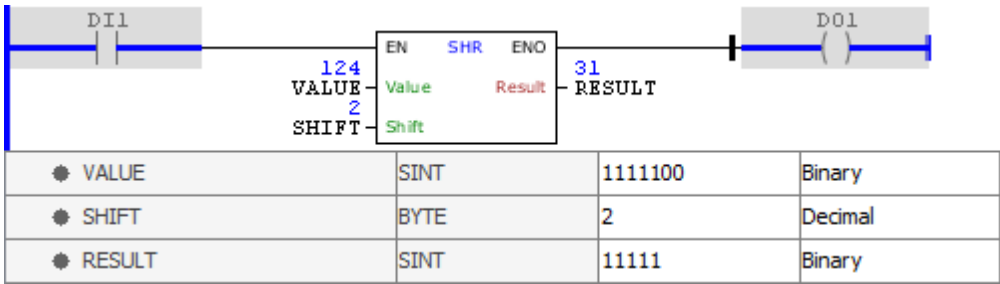
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

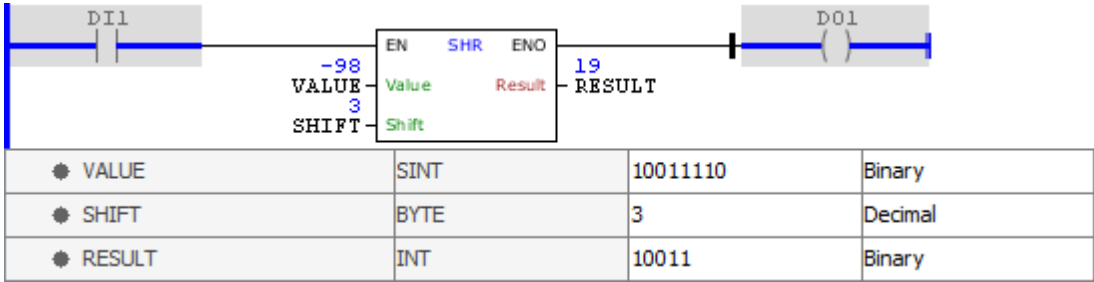
Fluxograma do Bloco



Exemplo



O exemplo acima realiza um deslocamento lógico à direita de duas posições na variável VALUE, cujo valor inicial é 124 (0111 1100 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0001 1111 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento lógico à direita de três posições na variável VALUE, cujo valor inicial é -98 (1001 1110 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0001 0011 em binário) é armazenado em RESULT.

10.1.5.1.6 Logic Rotate

10.1.5.1.6.1 ROL

Bloco que executa uma operação de rotação lógica à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer rotação
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

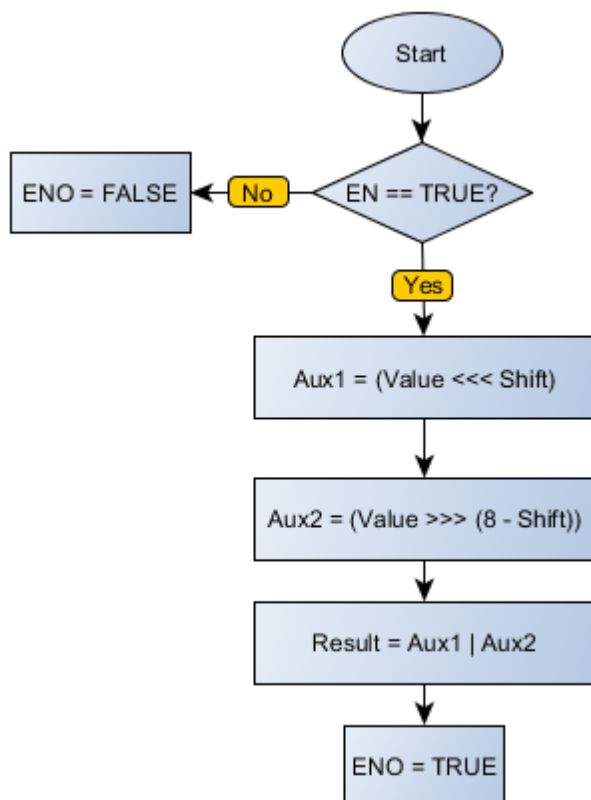
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à esquerda, de acordo com o valor de Shift. Os bits mais significativos que forem sendo descartados são reenviados aos bits menos significativos, caracterizando a rotação.

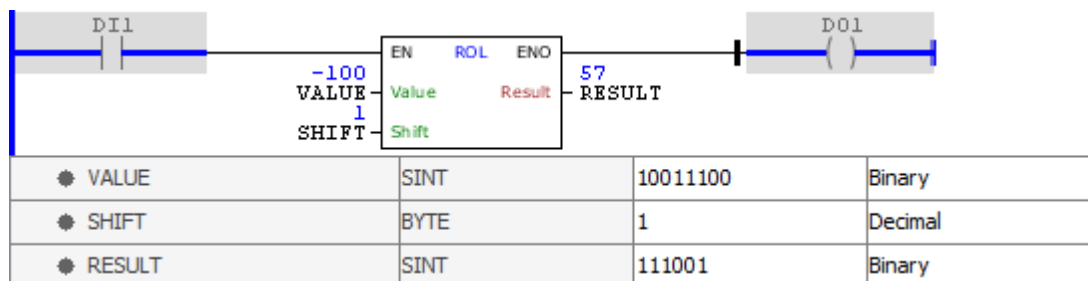
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

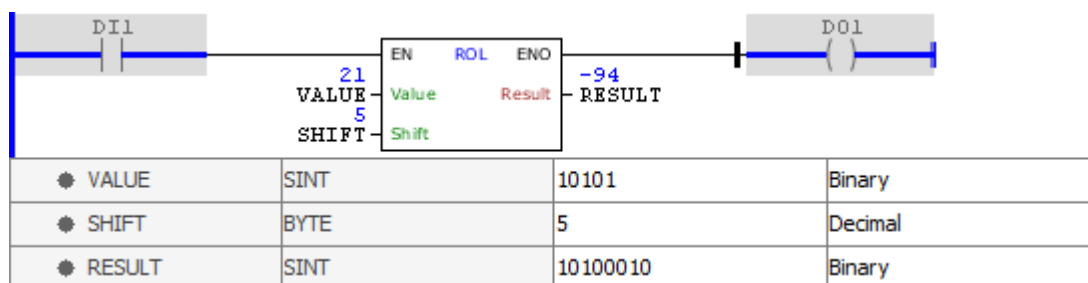
Fluxograma do Bloco



Exemplo



O exemplo acima realiza uma rotação lógica à esquerda de uma posição na variável VALUE, cujo valor inicial é -100 (1001 1100 em binário). Os bits descartados à esquerda são reinseridos à direita. O resultado final (0011 1001 em binário) é armazenado em RESULT.



O exemplo acima realiza uma rotação lógica à esquerda de cinco posições na variável VALUE, cujo valor inicial é 21 (0001 0101 em binário). Os bits descartados à esquerda são reinseridos à direita. O resultado final (1010 0010 em binário) é armazenado em RESULT.

10.1.5.1.6.2 ROR

Bloco que executa uma operação de rotação lógica à direita em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer rotação
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

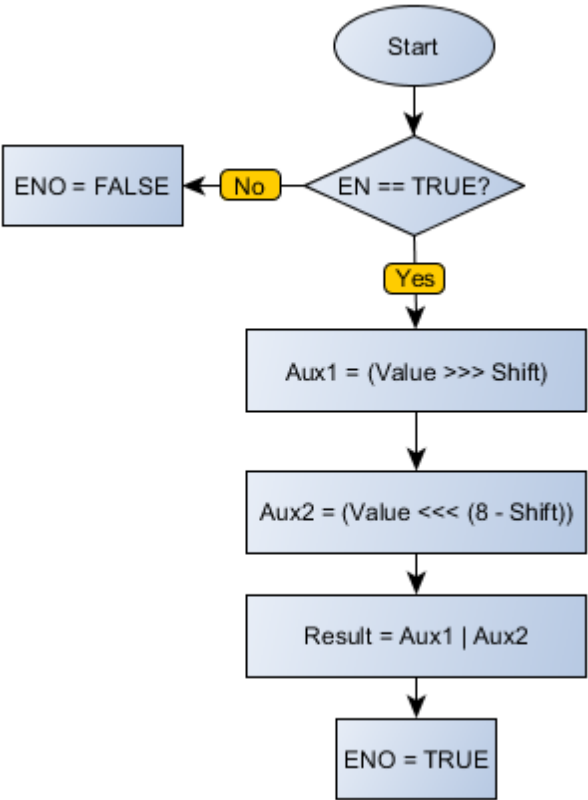
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à direita, de acordo com o valor de Shift. Os bits menos significativos que forem sendo descartados são reenviados aos bits mais significativos, caracterizando a rotação.

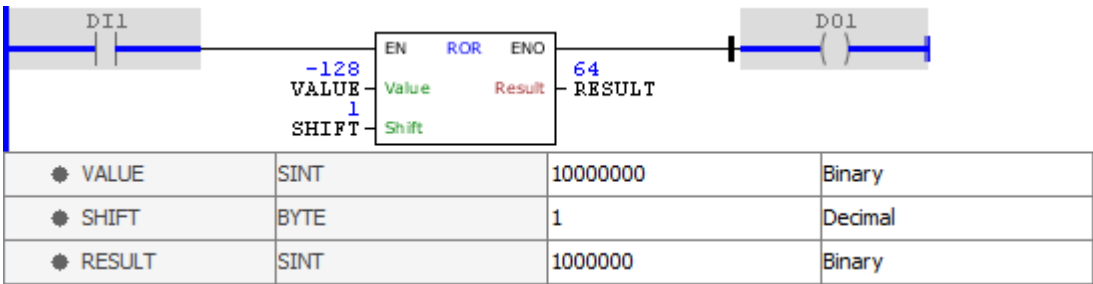
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

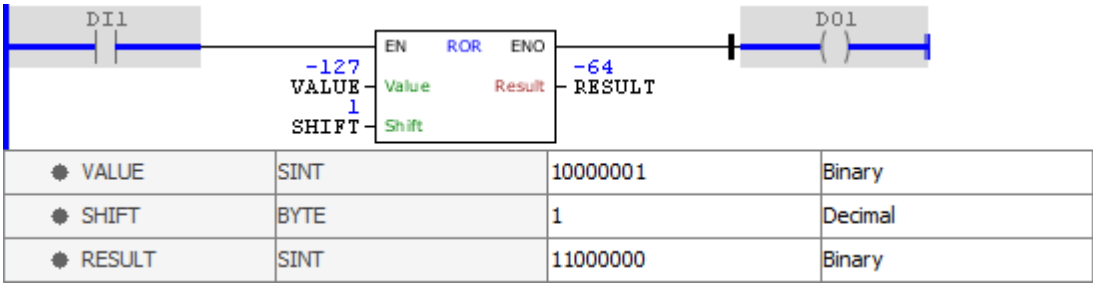
Fluxograma do Bloco



Exemplo



O exemplo acima realiza uma rotação lógica à direita de uma posição na variável VALUE, cujo valor inicial é -128 (1000 0000 em binário). Os bits descartados à direita são reinseridos à esquerda. O resultado final (0100 0000 em binário) é armazenado em RESULT. Observe que o sinal não é preservado nessa operação.



O exemplo acima realiza uma rotação lógica à direita de uma posição na variável VALUE, cujo valor inicial é -127 (1000 0001 em binário). Os bits descartados à direita são reinseridos à esquerda. O resultado final (1100 0000 em binário) é armazenado em RESULT.

10.1.5.1.7 Math Basic

10.1.5.1.7.1 ABS

Bloco que calcula o módulo de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

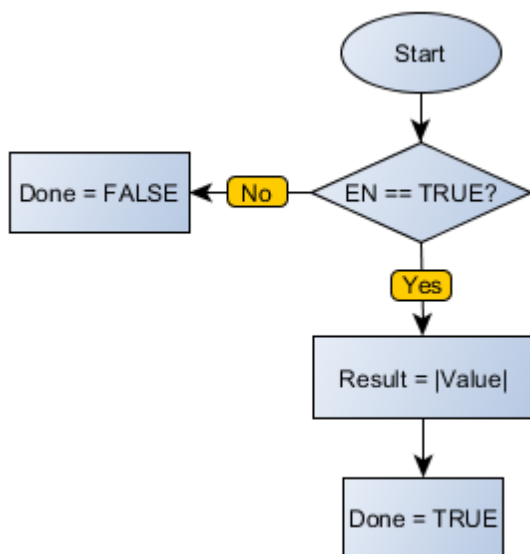
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

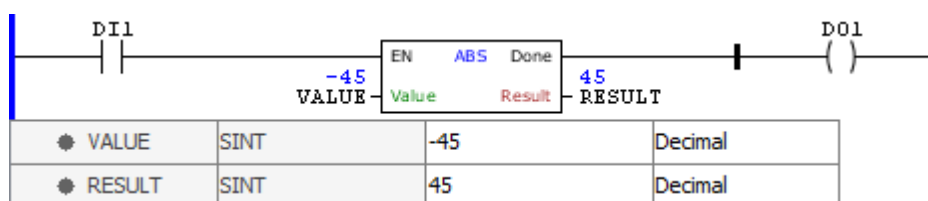
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor absoluto da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

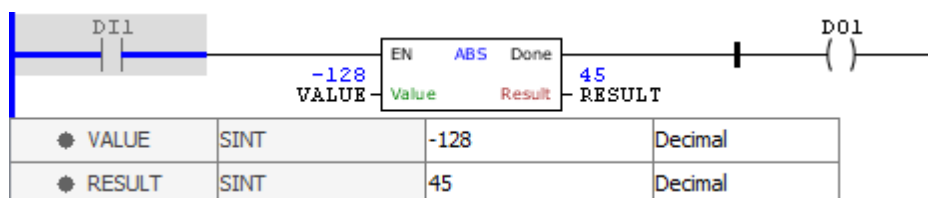
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o valor absoluto da variável VALUE, cujo valor inicial é -45, armazenando o resultado final, 45, em RESULT.

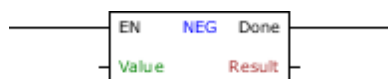


O exemplo acima calcula o valor absoluto da variável VALUE, cujo valor inicial é -45. O resultado final, 128, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.1.5.1.7.2 NEG

Bloco que calcula o oposto (ou seja, o produto com -1) de um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

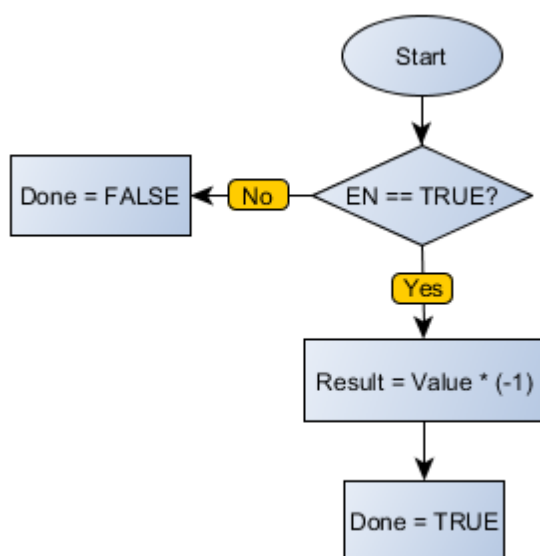
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

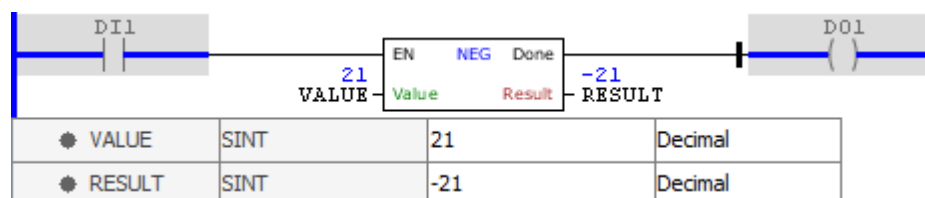
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o oposto da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

Fluxograma do Bloco

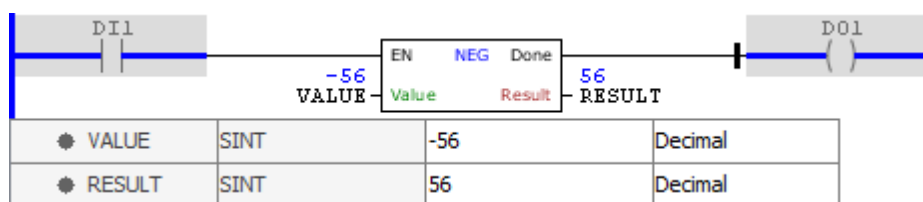


Exemplo

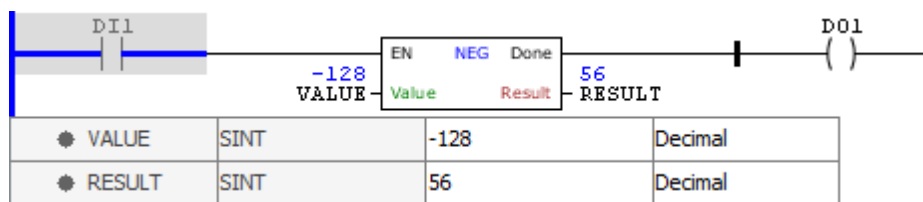


O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é 21, armazenando o

resultado final, -21, em RESULT.



O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é -56, armazenando o resultado final, 56, em RESULT.



O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é -128. O resultado final, 128, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.1.5.1.7.3 ADD

Bloco que calcula a soma dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeira parcela da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segunda parcela da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

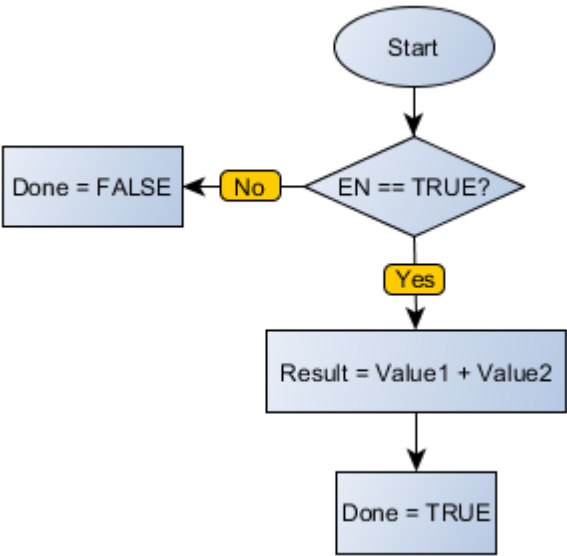
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a soma das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor

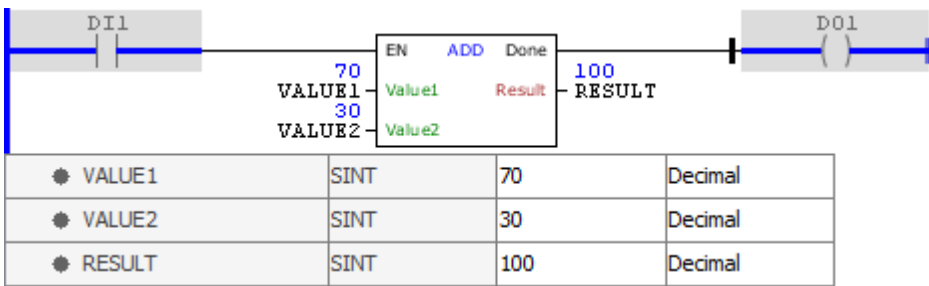
inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

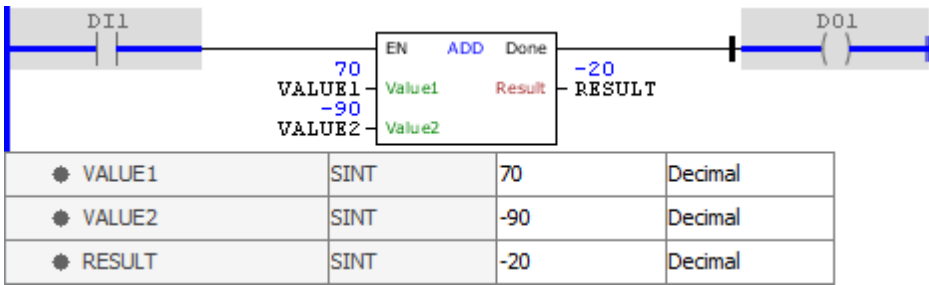
Fluxograma do Bloco



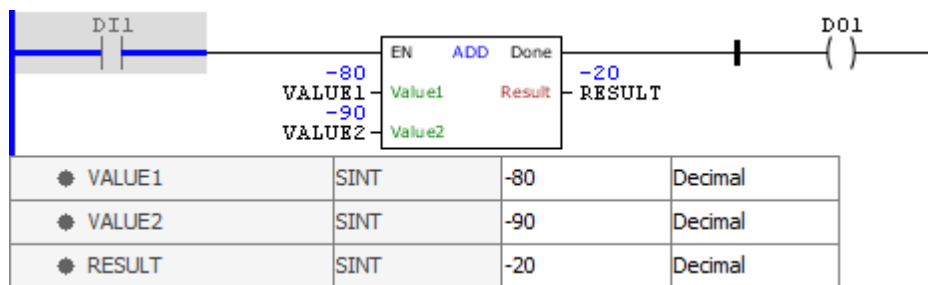
Exemplo



O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.

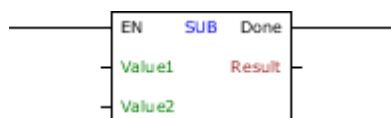


O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2. O resultado final, -170, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.1.5.1.7.4 SUB

Bloco que calcula a subtração entre os valores Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

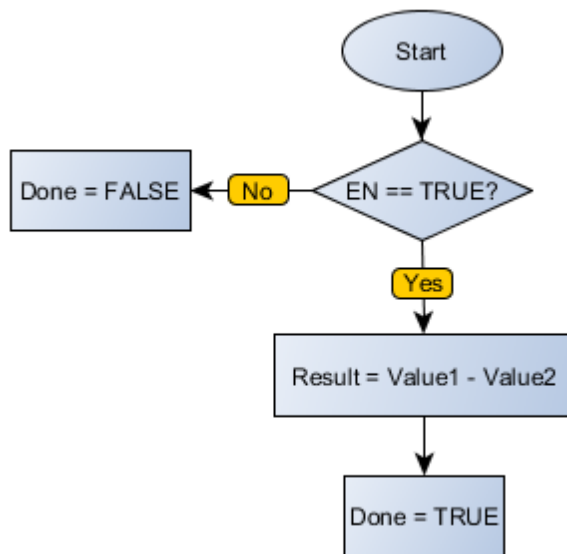
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Minuendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Subtraendo da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

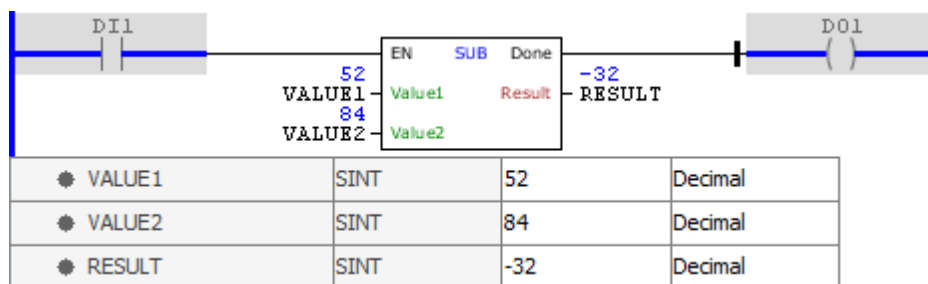
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a subtração das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

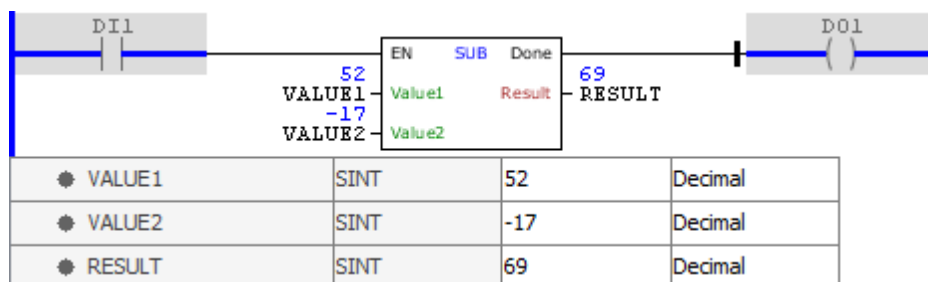
Fluxograma do Bloco



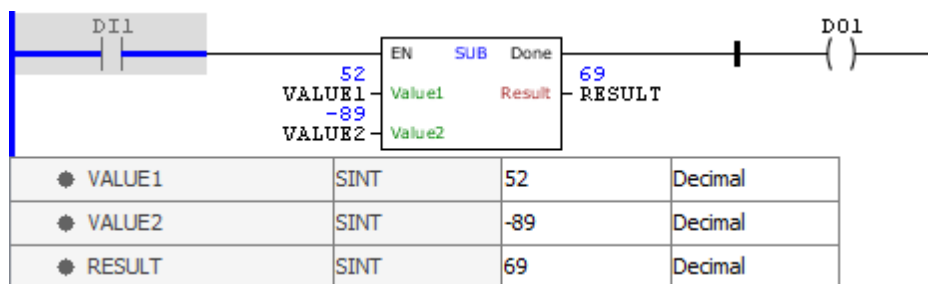
Exemplo



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2. O resultado final, 141, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.1.5.1.7.5 MUL

Bloco que calcula a multiplicação dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

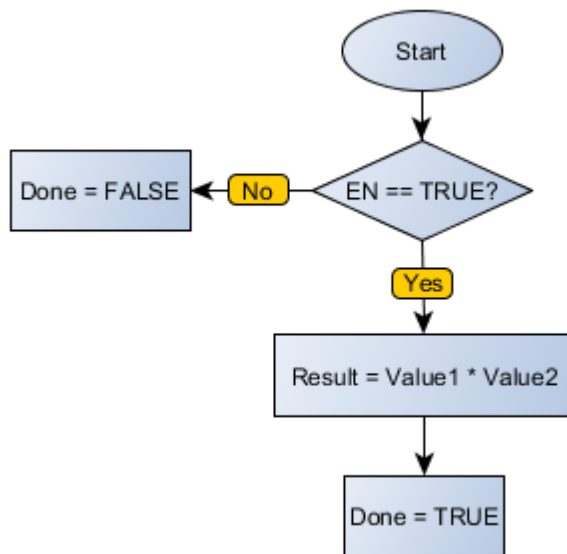
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro fator da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo fator da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

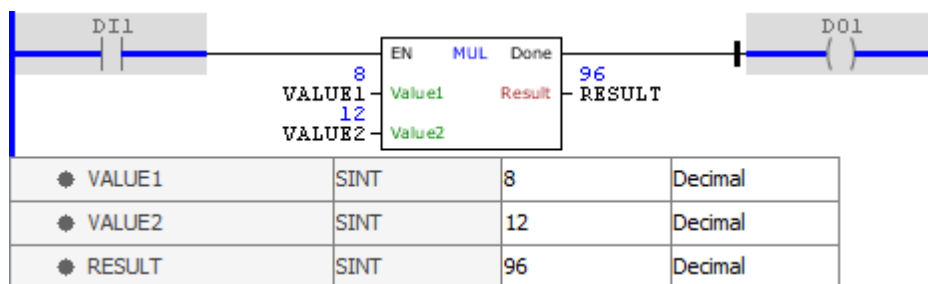
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a multiplicação das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

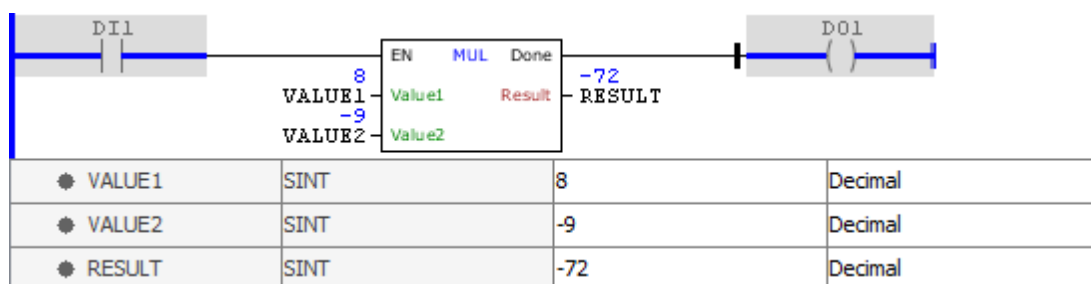
Fluxograma do Bloco



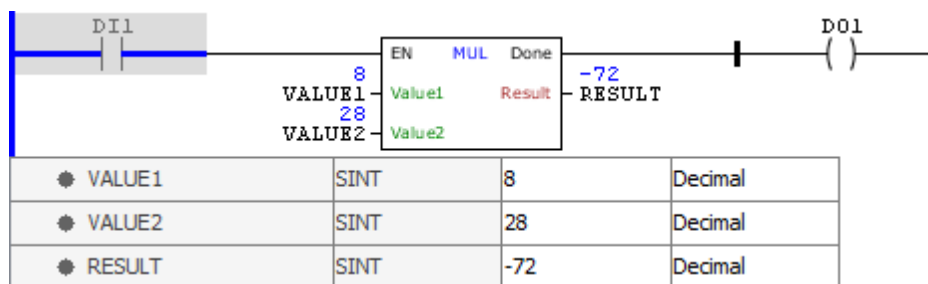
Exemplo



O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.

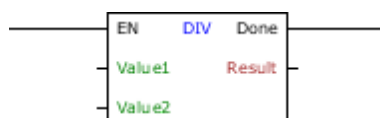


O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2. O resultado final, 224, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.1.5.1.7.6 DIV

Bloco que calcula a divisão dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

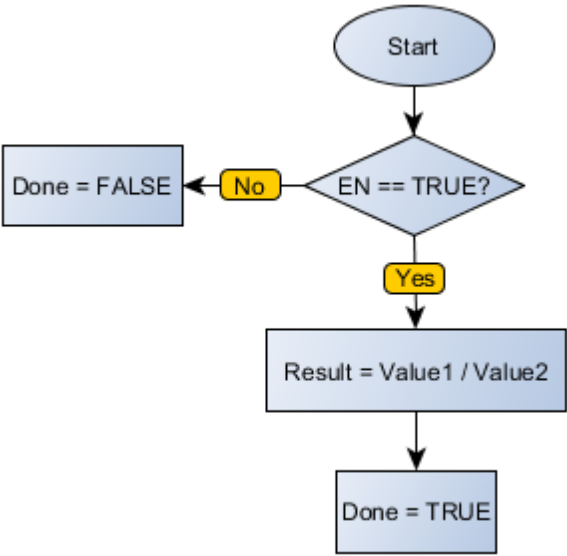
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Dividendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Divisor da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

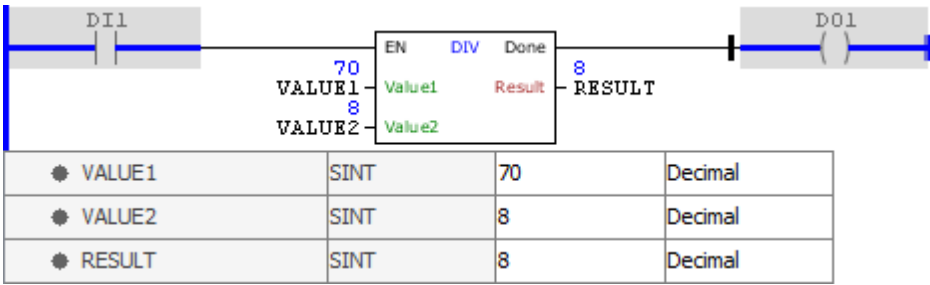
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a divisão das variáveis Value1 e Value2. O valor armazenado será a divisão exata caso Result seja REAL, ou, nos demais casos, apenas o quociente da divisão. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

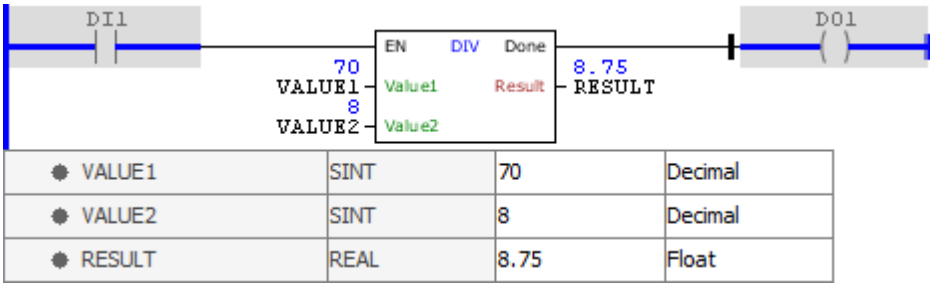
Fluxograma do Bloco



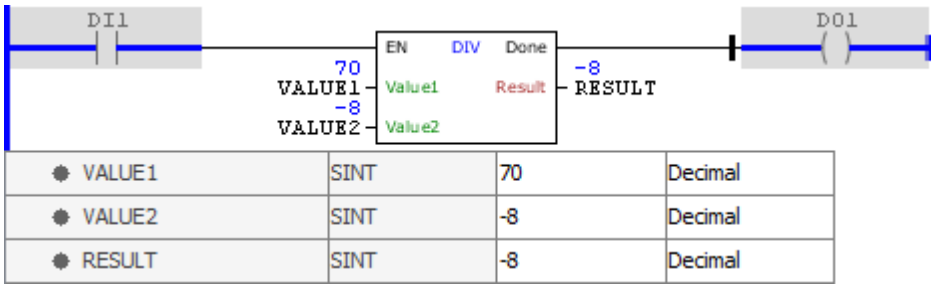
Exemplo



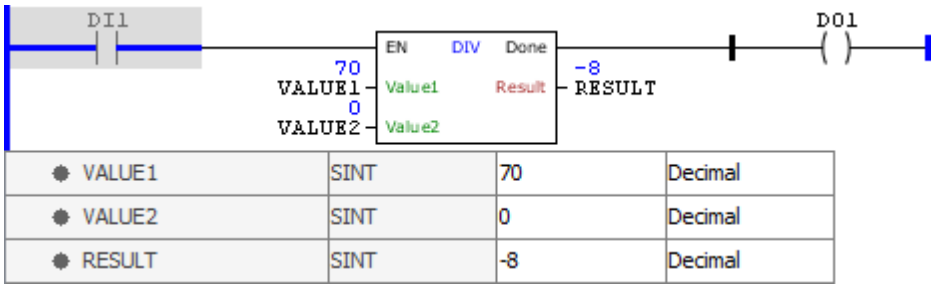
O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo SINT, apenas o quociente da divisão é armazenado nela.



O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo REAL, o valor exato da divisão é armazenado nela.



O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo SINT, apenas o quociente da divisão é armazenado nela. Observe que o bloco aceita argumentos de ambos os sinais.

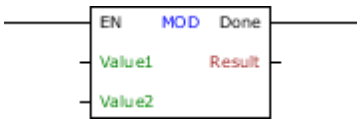


O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como VALUE2 é zero, o bloco gera um erro de execução, RESULT permanece inalterado e a saída é desabilitada.

10.1.5.1.7.7 MOD

Bloco que calcula o resto da divisão dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

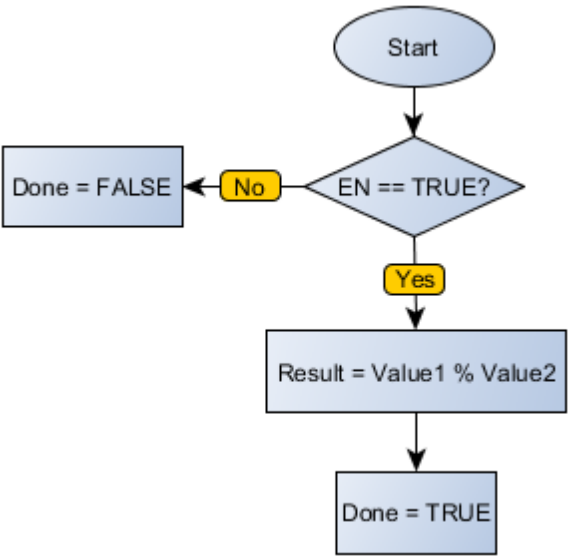
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Dividendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Divisor da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

Funcionamento

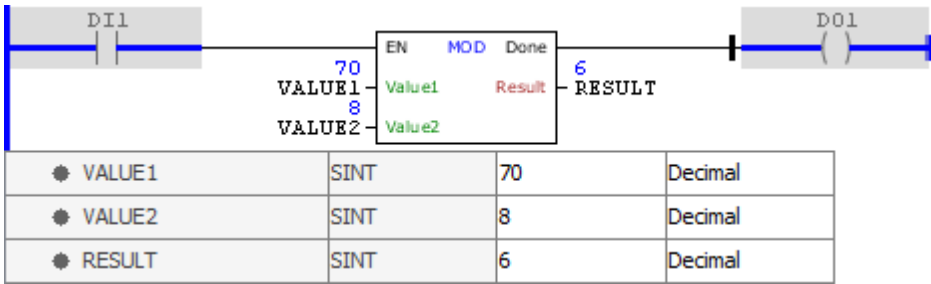
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o resto da divisão das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

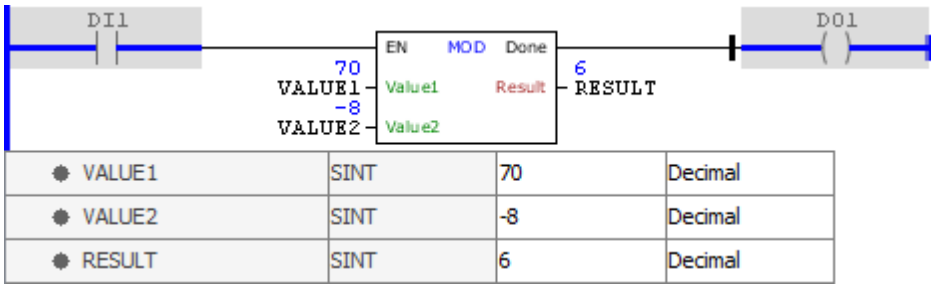
Fluxograma do Bloco



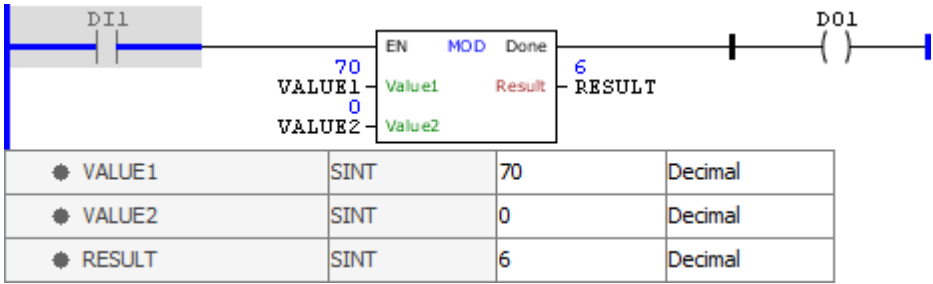
Exemplo



O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.



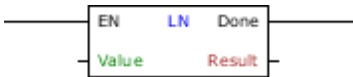
O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como VALUE2 é zero, o bloco gera um erro de execução, RESULT permanece inalterado e a saída é desabilitada.

10.1.5.1.8 Math Extended

10.1.5.1.8.1 LN

Bloco que calcula o logaritmo natural do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

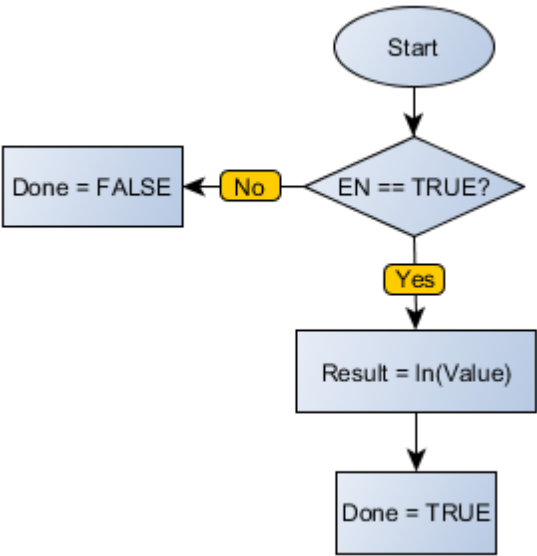
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o logaritmo natural da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor

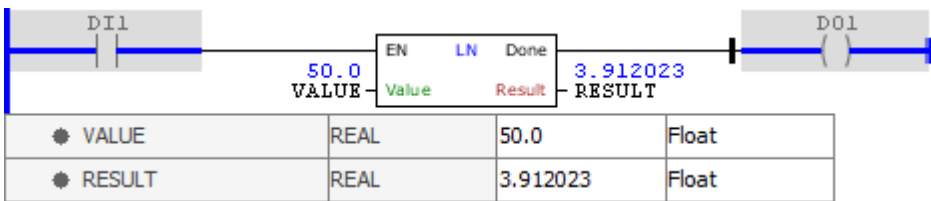
inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

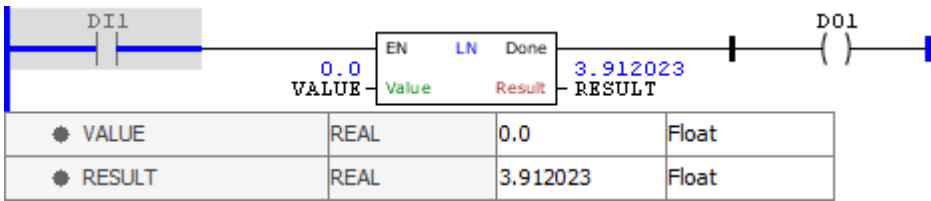
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o logaritmo natural da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



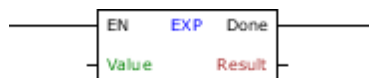
O exemplo acima calcula o logaritmo natural da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor zero, e a saída Done é desativada.

10.1.5.1.8.2 EXP

Bloco que calcula a exponencial do número de Euler “e” elevado ao valor de Value, armazenando o

resultado em Result.

Representação Ladder



Estrutura do Bloco

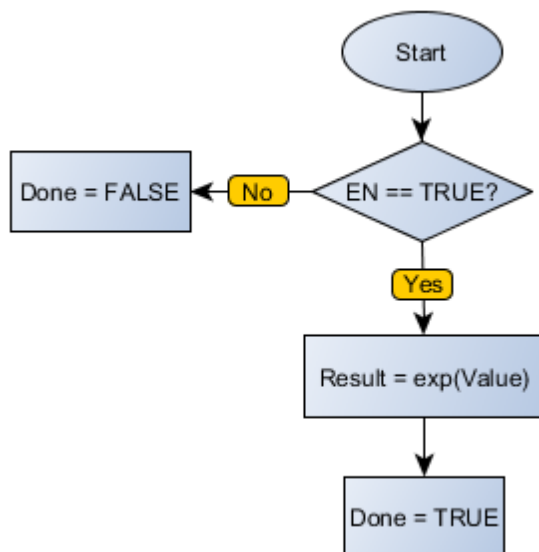
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

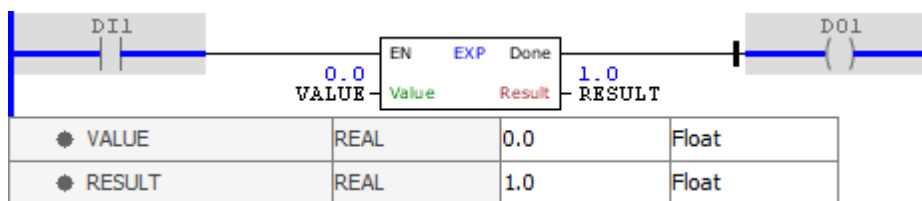
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a exponencial do número de Euler “e” elevado à variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

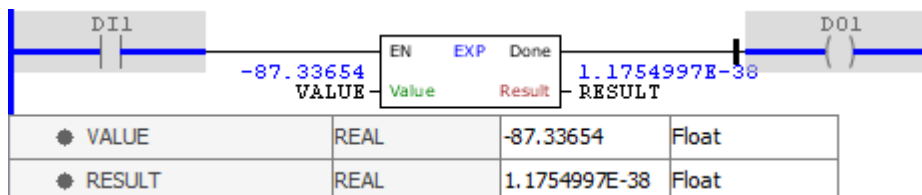
Fluxograma do Bloco



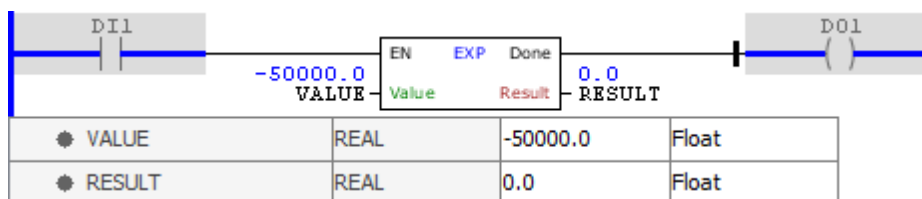
Exemplo



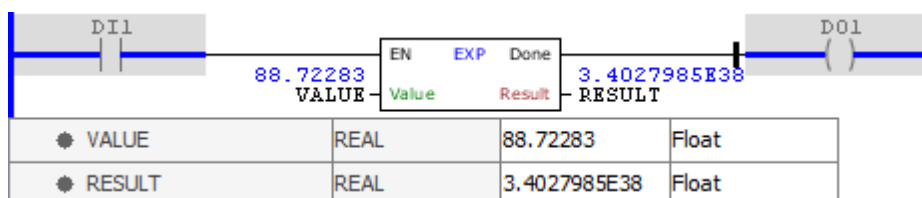
O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



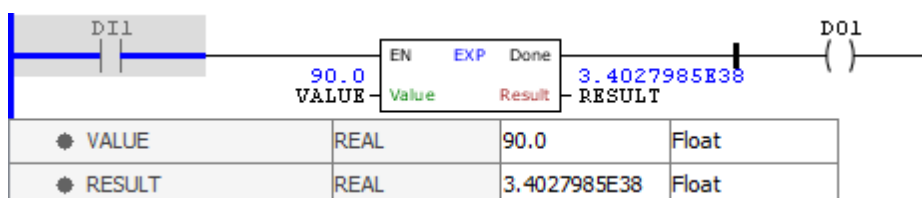
O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor mínimo de entrada para o qual o bloco retorna um resultado não nulo. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. Valores inferiores ao mínimo fazem com que o bloco retorne valor nulo. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor máximo de entrada para o qual o bloco retorna um resultado válido. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. Valores superiores ao máximo fazem com que o bloco gere um erro, a saída RESULT permanece inalterada e a saída Done seja desativada.

10.1.5.1.8.3 LOG10

Bloco que calcula o logaritmo comum (base 10) do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

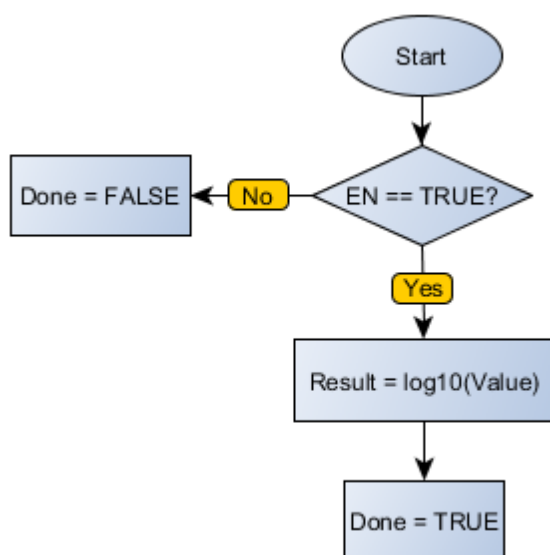
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

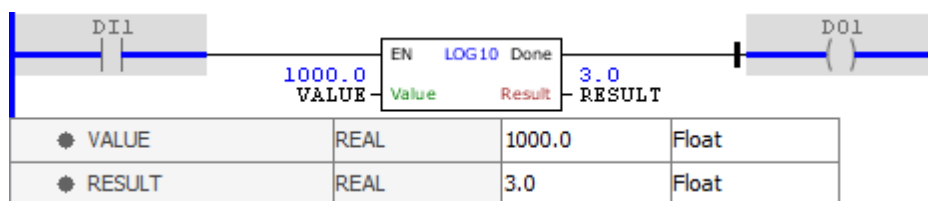
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o logaritmo comum da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

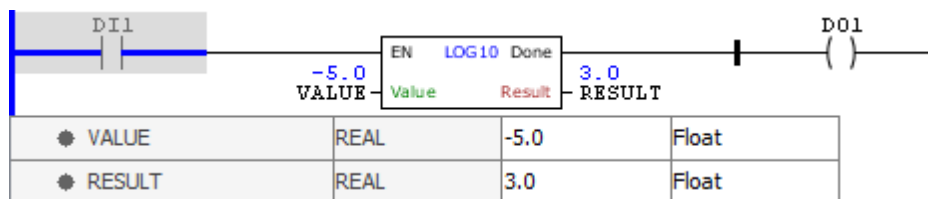
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o logaritmo comum da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o logaritmo comum da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor negativo, e a saída Done é desativada.

10.1.5.1.8.4 ALOG10

Bloco que calcula o antilogaritmo (exponencial com base 10) do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

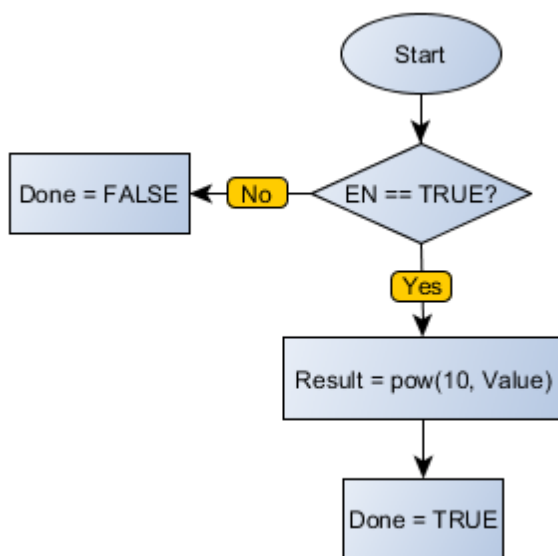
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

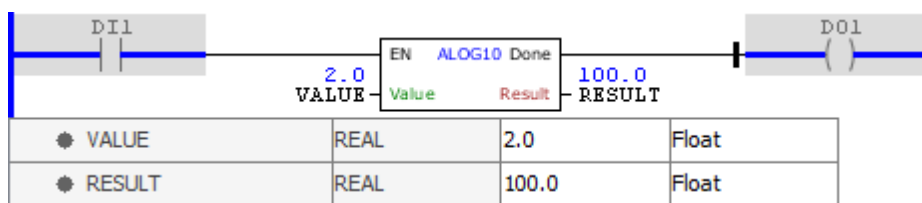
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o antilogaritmo da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

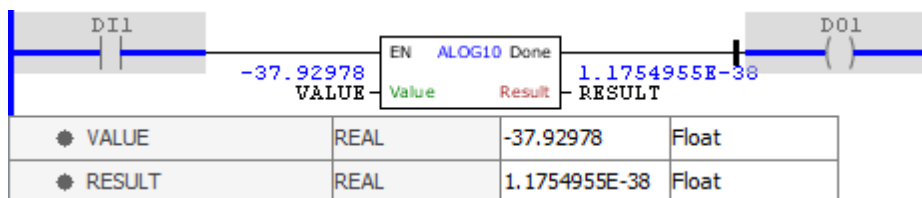
Fluxograma do Bloco



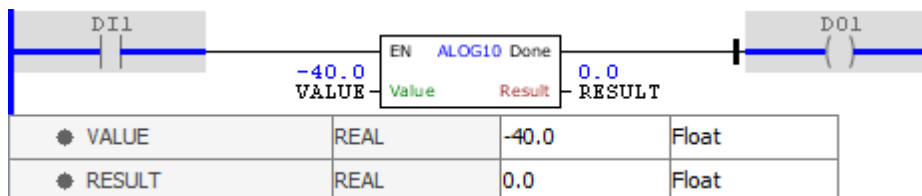
Exemplo



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.

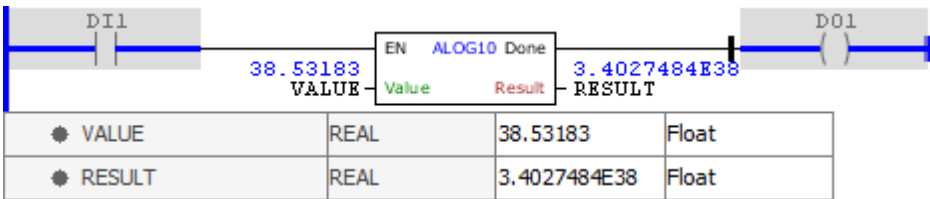


O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor mínimo de entrada para o qual o bloco retorna um resultado não nulo. O bloco finaliza com sucesso e a saída Done é ativada.

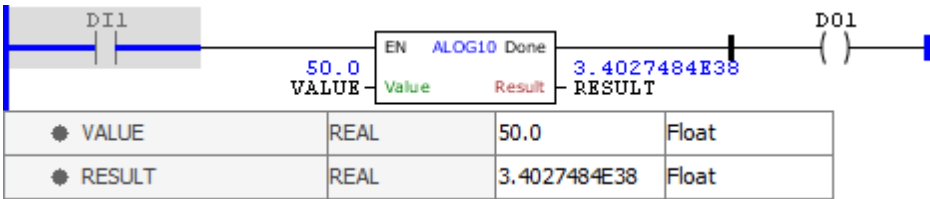


O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. Valores inferiores ao mínimo fazem com que o bloco retorne valor nulo. O bloco finaliza

com sucesso e a saída Done é ativada.



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor máximo de entrada para o qual o bloco retorna um resultado válido. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. Valores superiores ao máximo fazem com que o bloco gere um erro, a saída RESULT permanece inalterada e a saída Done seja desativada.

10.1.5.1.8.5 POW

Bloco que calcula o valor de Value elevado à potência Power, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Base da operação
	Power	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Potência da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

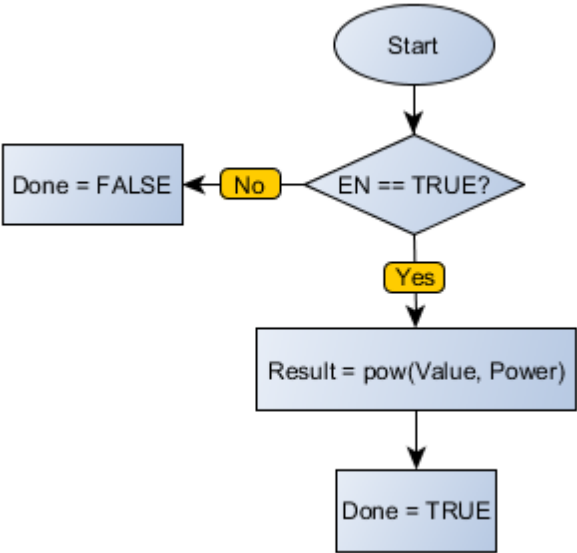
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor de Value elevado à potência Power. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação,

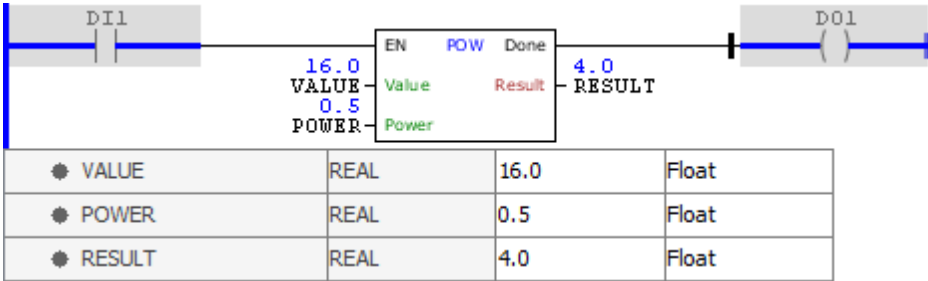
Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

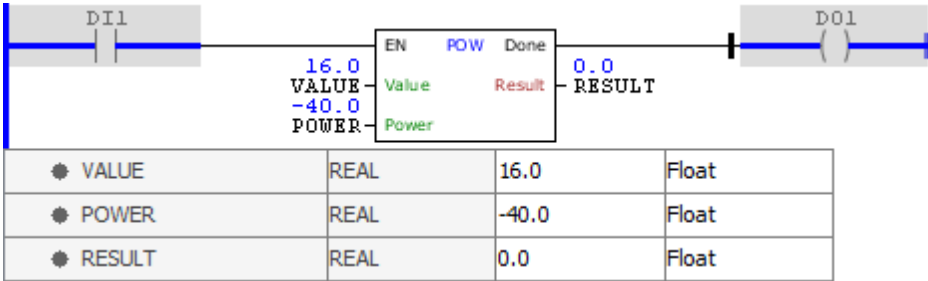
Fluxograma do Bloco



Exemplo

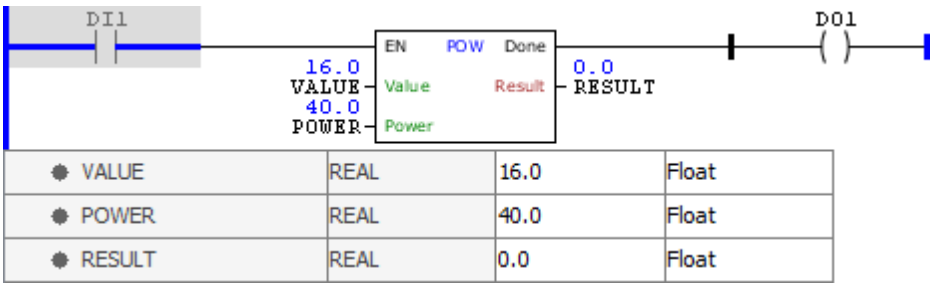


O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER,

armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER, armazenando o resultado final em RESULT. Como o resultado é superior ao valor máximo suportado pelo tipo REAL, o bloco gera um erro e a saída Done é desativada.

10.1.5.1.8.6 Sqrt

Bloco que calcula o valor da raiz quadrada de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

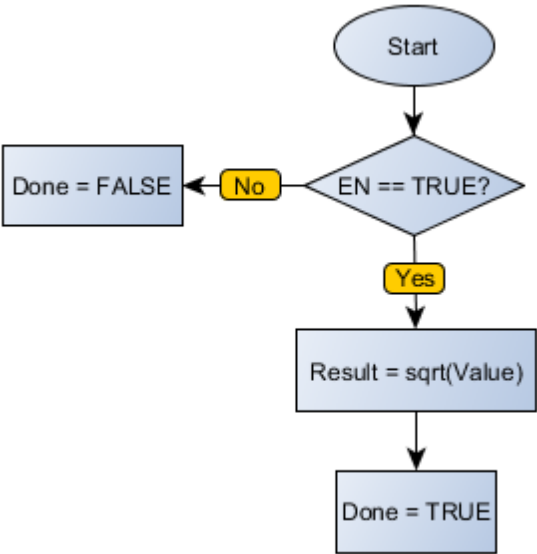
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

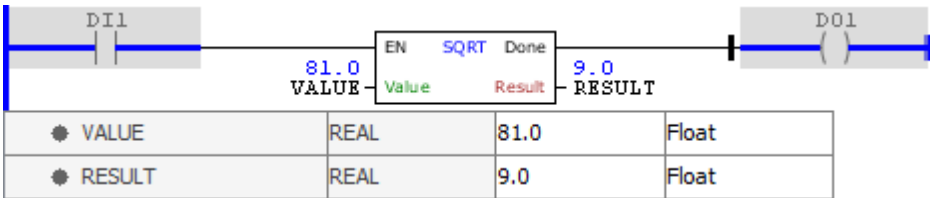
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da raiz quadrada de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

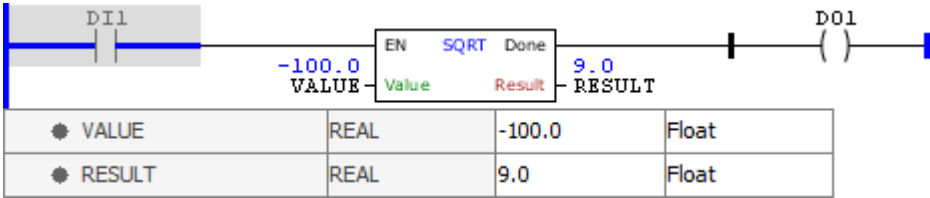
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o valor da raiz quadrada da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da raiz quadrada da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor negativo, e a saída Done é desativada.

10.1.5.1.8.7 ROUND

Bloco que arredonda o valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

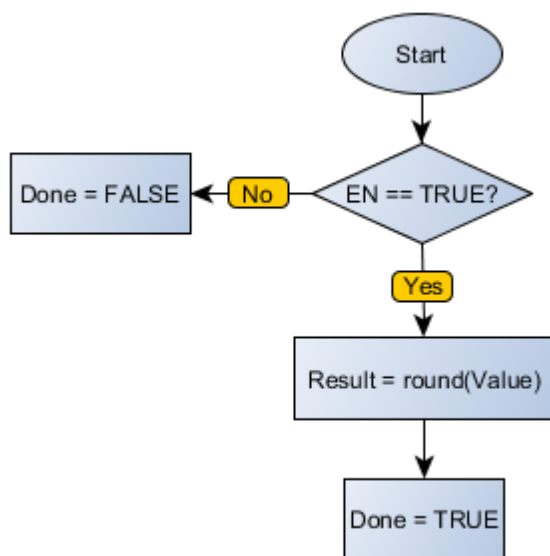
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor arredondado de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

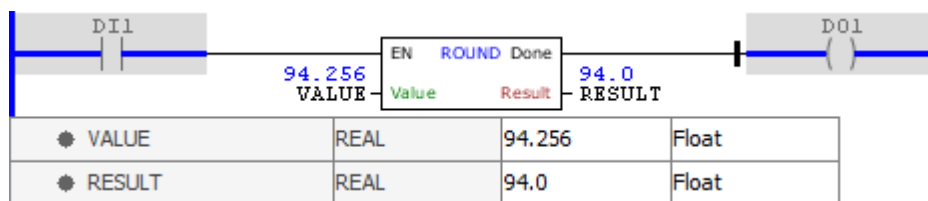
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior
SCA06	2.00 ou superior

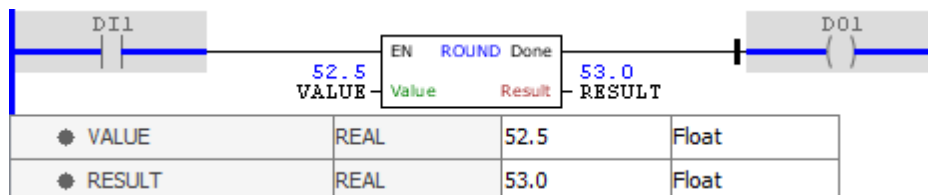
Fluxograma do Bloco



Exemplo



O exemplo acima arredonda o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais inferiores a 0,5 são descartados. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima arredonda o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais iguais ou superiores a 0,5 promovem a unidade ao valor imediatamente acima. O bloco finaliza com sucesso e a saída Done é ativada.

10.1.5.1.8.8 TRUNC

Bloco que trunca o valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

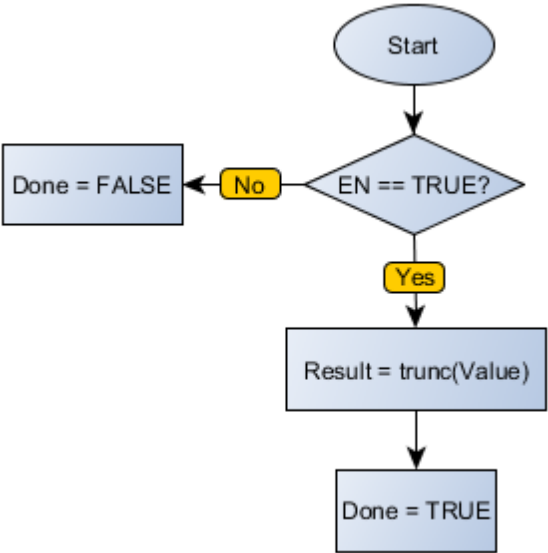
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor truncado de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

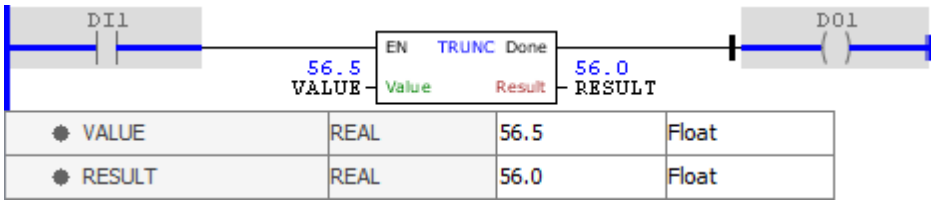
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



O exemplo acima trunca o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais são descartados. O bloco finaliza com sucesso e a saída Done é ativada.

10.1.5.1.9 Math Trigonometry

10.1.5.1.9.1 SIN

Bloco que calcula o seno de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

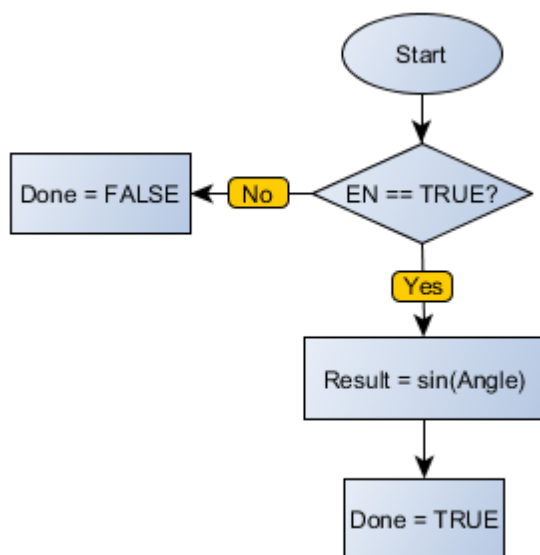
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

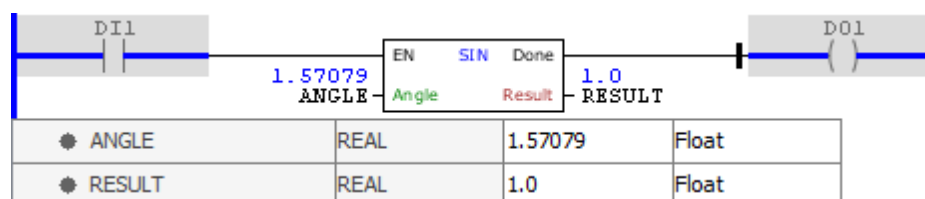
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o seno de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

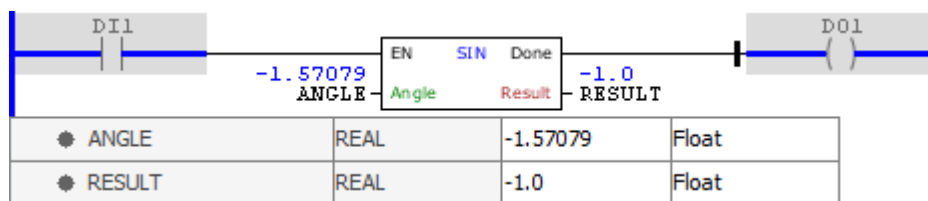
Fluxograma do Bloco



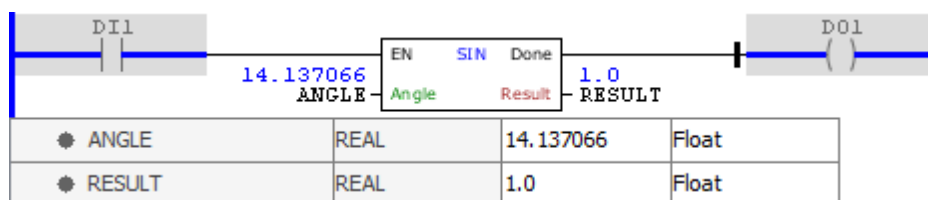
Exemplo



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores negativos de entrada.



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores superiores a uma volta completa.

10.1.5.1.9.2 COS

Bloco que calcula o cosseno de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

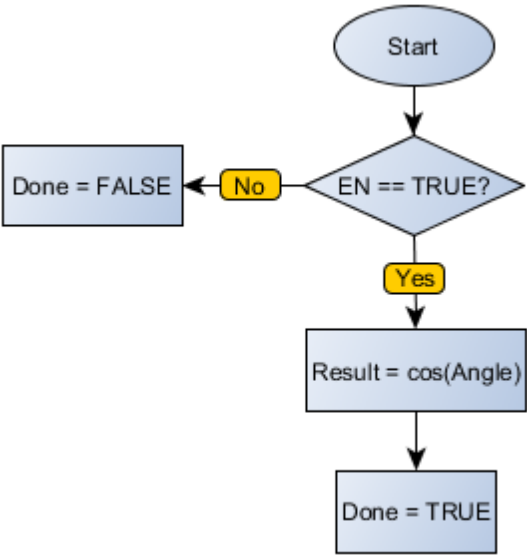
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

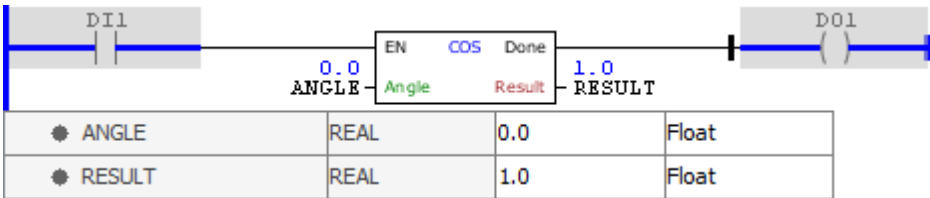
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o cosseno de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

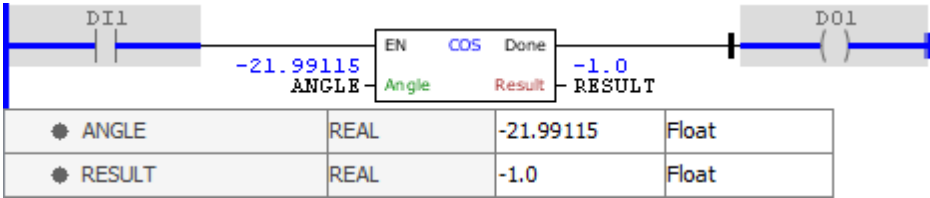
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o cosseno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o cosseno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores de entrada negativos e superiores a uma volta.

10.1.5.1.9.3 TAN

Bloco que calcula a tangente de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

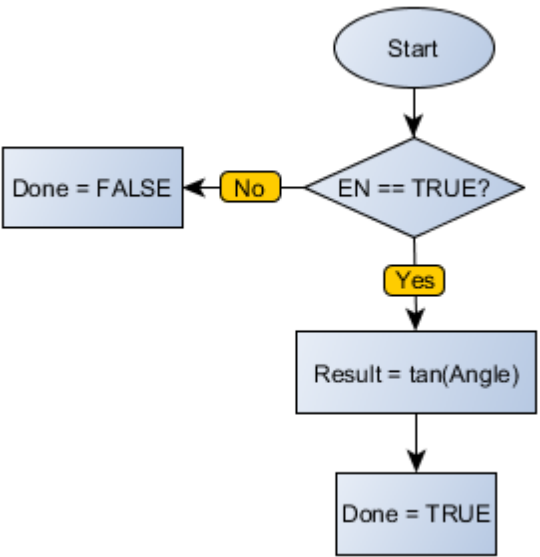
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

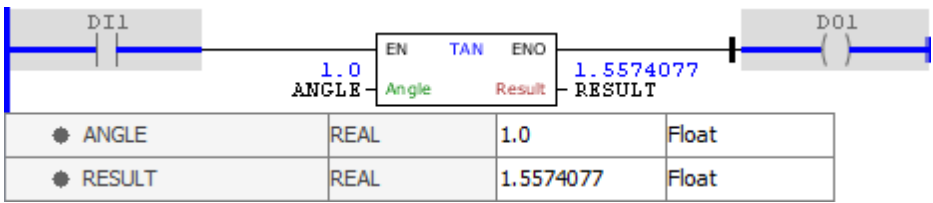
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a tangente de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

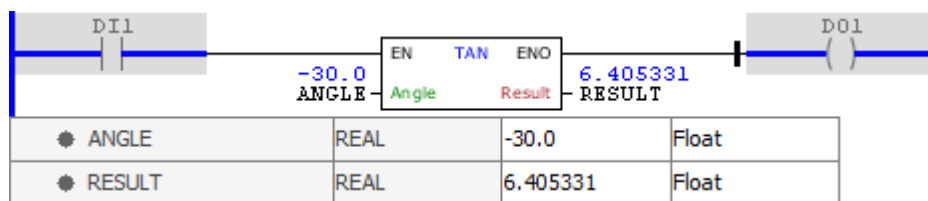
Fluxograma do Bloco



Exemplo



O exemplo acima calcula a tangente da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a tangente da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores de entrada negativos e superiores a uma volta.

10.1.5.1.9.4 ASIN

Bloco que calcula o arco seno de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

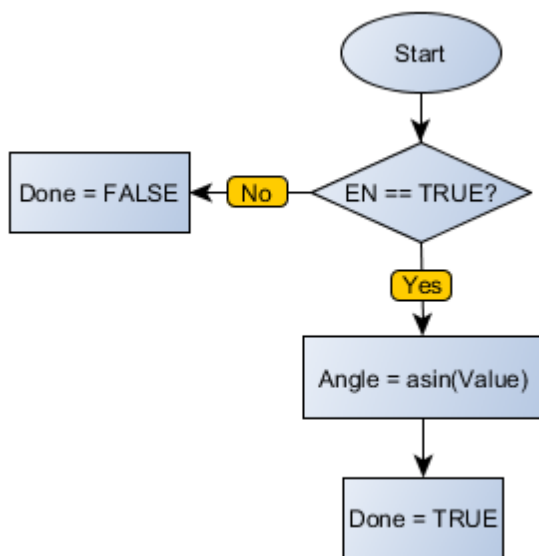
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor do seno
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cujo seno é igual a Value (em radianos)

Funcionamento

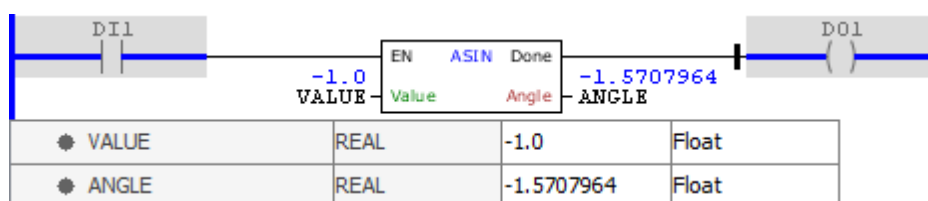
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco seno de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

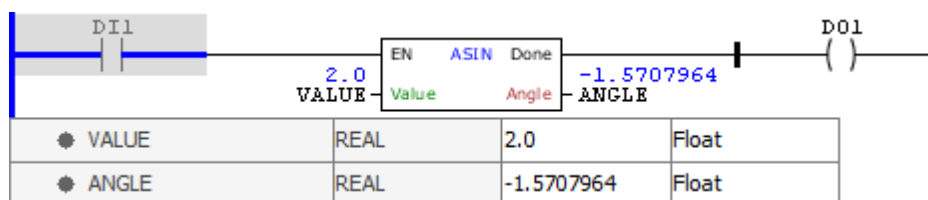
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cujo seno é a variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.

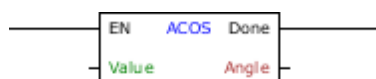


O exemplo acima calcula o arco, em radianos, cujo seno é a variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor superior a 1, e a saída Done é desativada.

10.1.5.1.9.5 ACOS

Bloco que calcula o arco cosseno de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

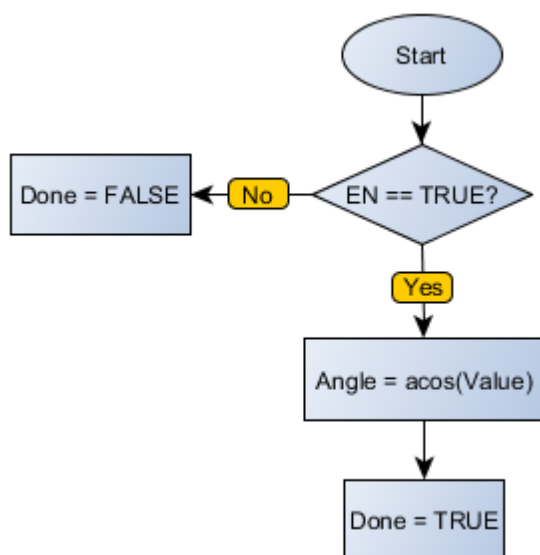
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor do cosseno
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cujo cosseno é igual a Value (em radianos)

Funcionamento

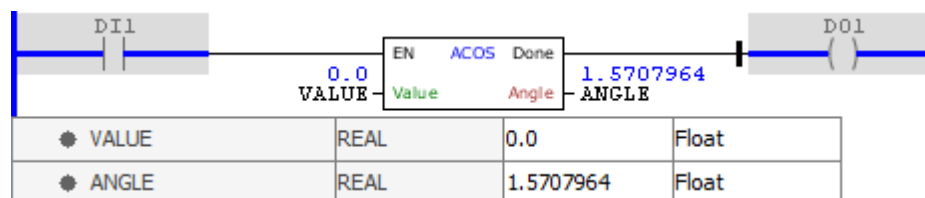
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco cosseno de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

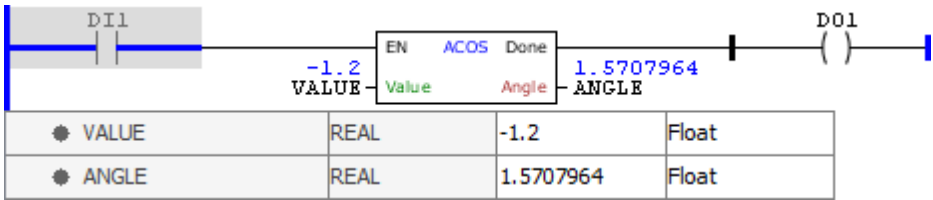
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cujo cosseno é a variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cujo cosseno é a variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor inferior a 1, e a saída Done é desativada.

10.1.5.1.9.6 ATAN

Bloco que calcula o arco tangente de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

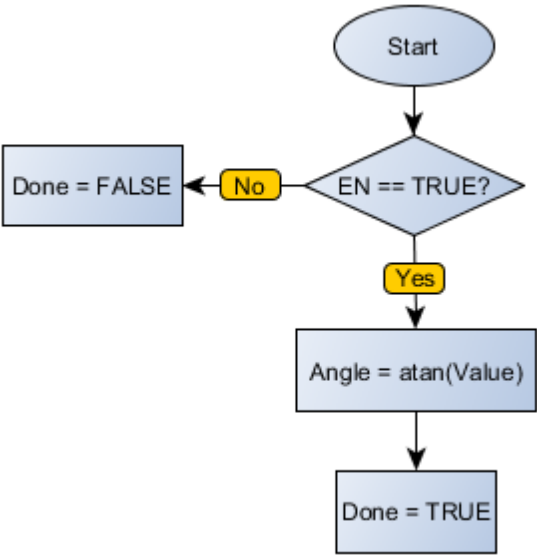
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor da tangente
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cuja tangente é igual a Value (em radianos)

Funcionamento

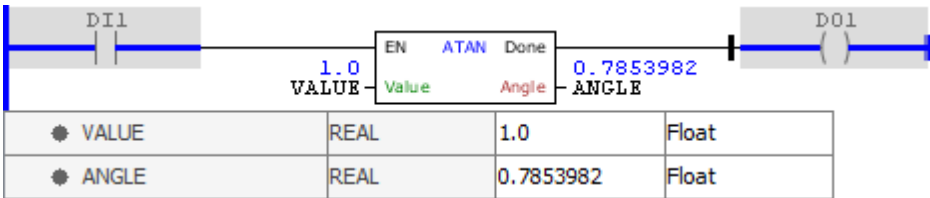
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco tangente de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

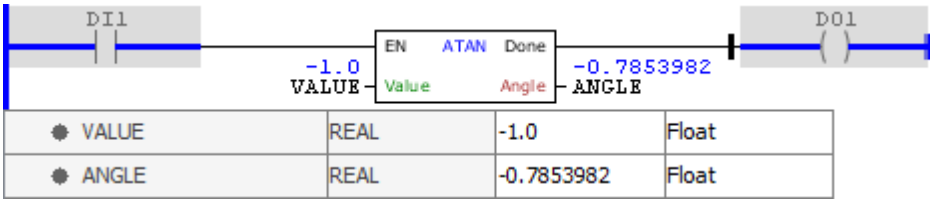
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cuja tangente é a variável VALUE, armazenando o resultado final em RESULT. O arco, para valores positivos, sempre se encontra no primeiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é a variável VALUE, armazenando o resultado final em RESULT. O arco, para valores negativos, sempre se encontra no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

10.1.5.1.9.7 ATAN2

Bloco que calcula o arco tangente de Y/X, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

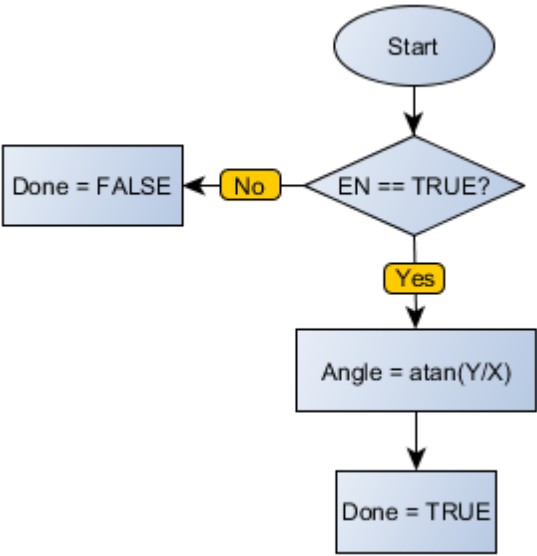
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	X	REAL	Parâmetro X da função
	Y	REAL	Parâmetro Y da função
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cuja tangente é igual a (Y/X) (em radianos)

Funcionamento

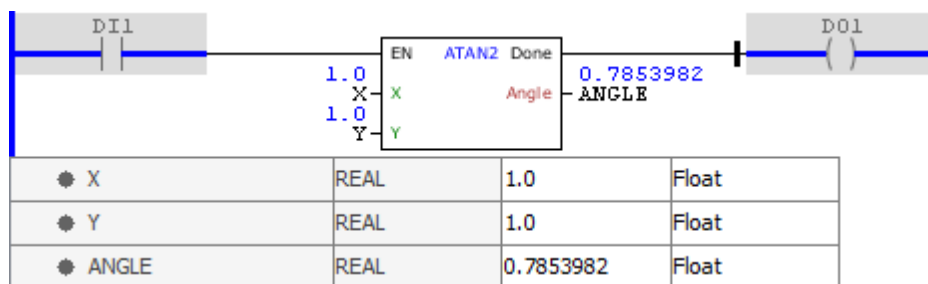
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco tangente de Y/X. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

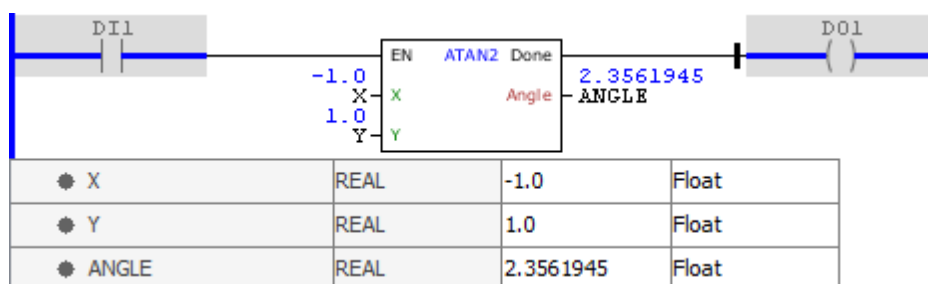
Fluxograma do Bloco



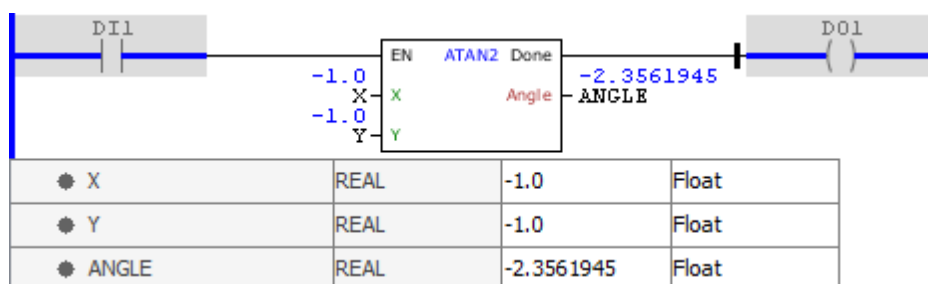
Exemplo



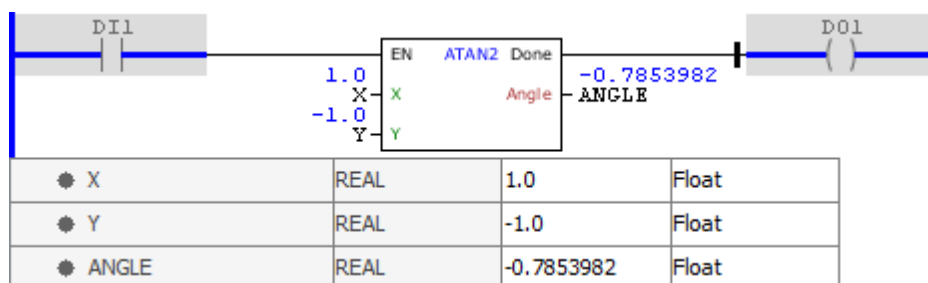
O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores positivos de X e Y, sempre se encontra no primeiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores negativos de X e positivos de Y, sempre se encontra no segundo quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores negativos de X e Y, sempre se encontra no terceiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores positivos de X e negativos de Y, sempre se encontra no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

10.1.5.1.10 Math Util

10.1.5.1.10.1 MAX

Bloco que compara os valores de Value1 e Value2 e armazena o maior deles em Result.

Representação Ladder



Estrutura do Bloco

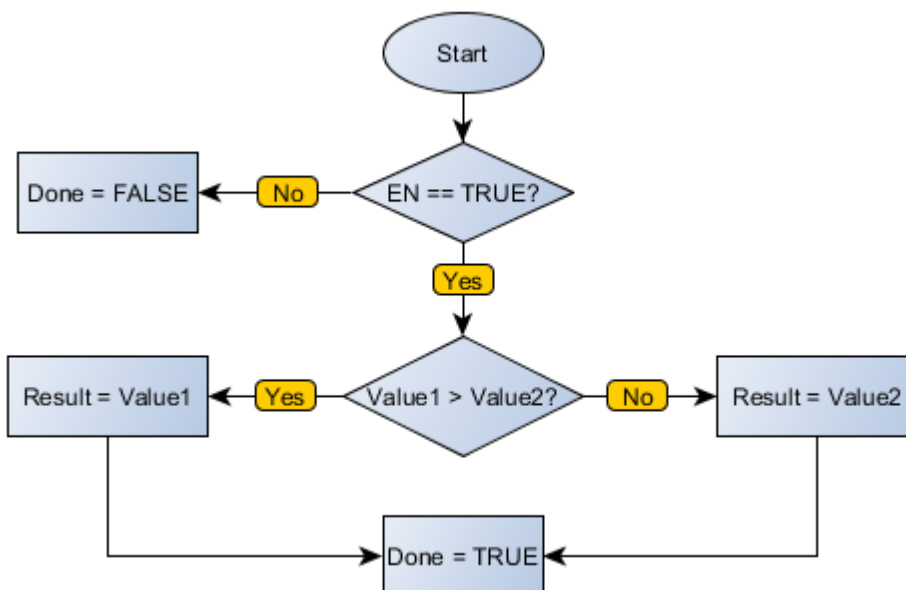
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Maior dos valores comparados

Funcionamento

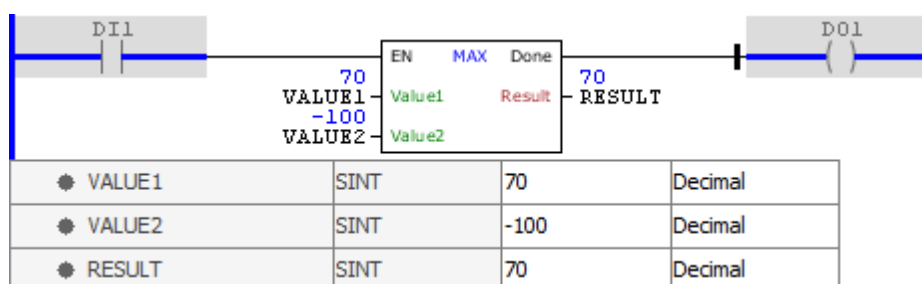
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o maior valor na comparação entre Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

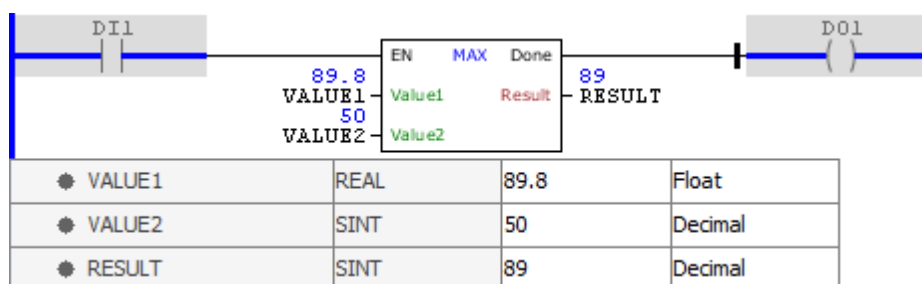
Fluxograma do Bloco



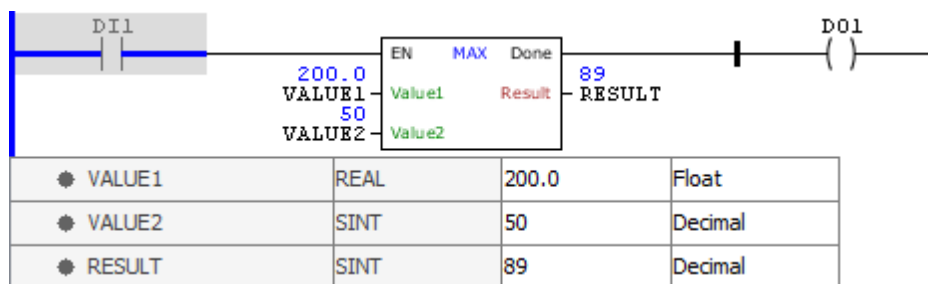
Exemplo



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como o resultado é superior ao valor máximo suportado pelo tipo SINT, o bloco gera um erro e a saída Done é desativada.

10.1.5.1.10.2 MIN

Bloco que compara os valores de Value1 e Value2 e armazena o menor deles em Result.

Representação Ladder



Estrutura do Bloco

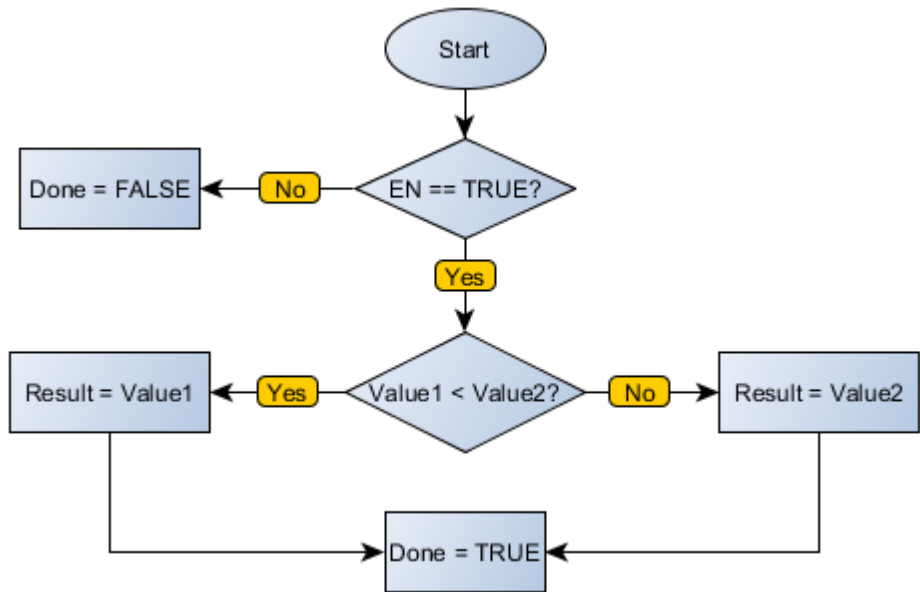
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Menor dos valores comparados

Funcionamento

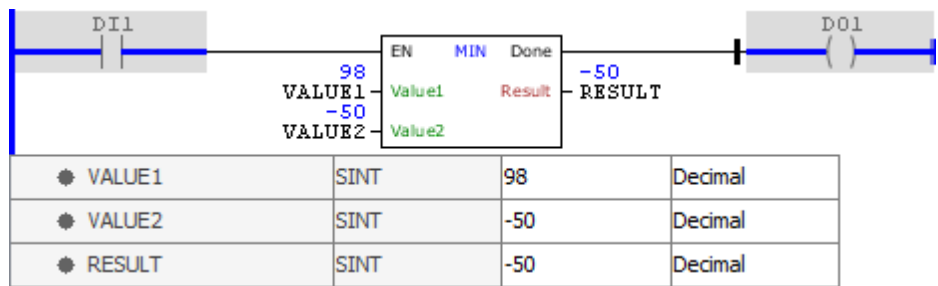
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o menor valor na comparação entre Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

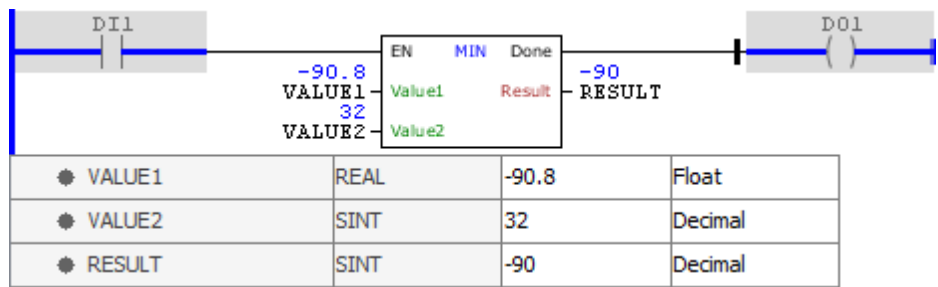
Fluxograma do Bloco



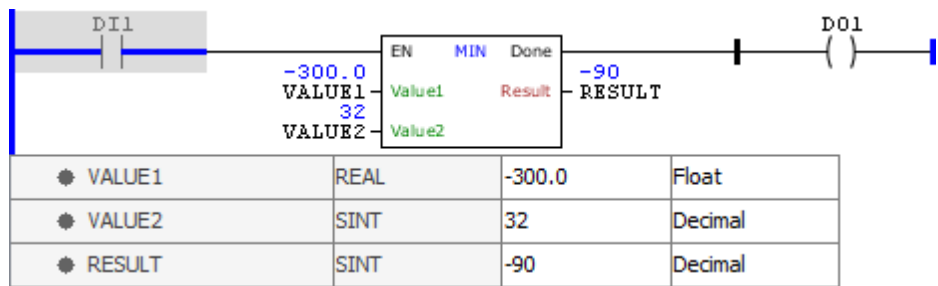
Exemplo



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como o resultado é inferior ao valor mínimo suportado pelo tipo SINT, o bloco gera um erro e a saída Done é desativada.

10.1.5.1.10.3 SAT

Bloco que realiza uma rotina de saturação do valor encontrado em Value de acordo com os limites estipulados em Minimum e Maximum, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

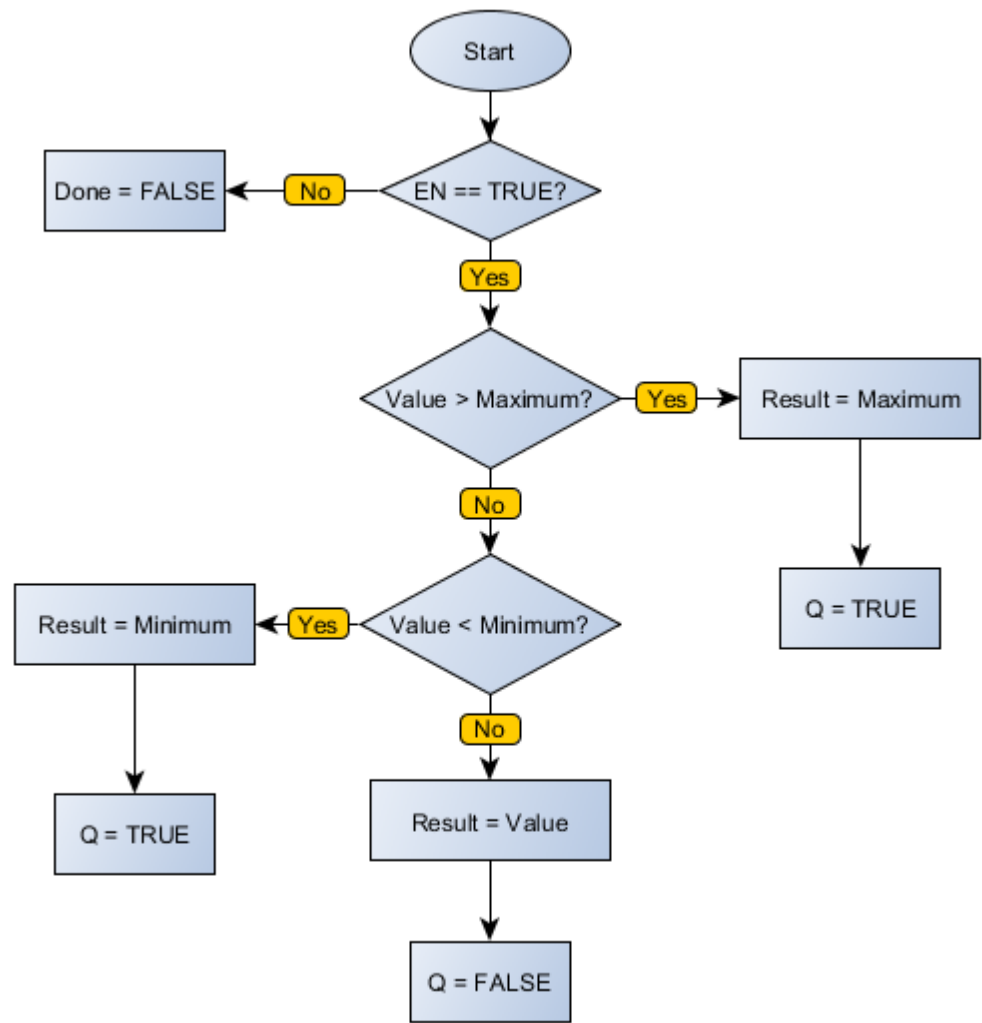
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de referência
	Minimum	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saturação inferior
	Maximum	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saturação superior
VAR_OUTPUT	Q	BOOL	Indicador de que houve saturação no processo
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Resultado da operação

Funcionamento

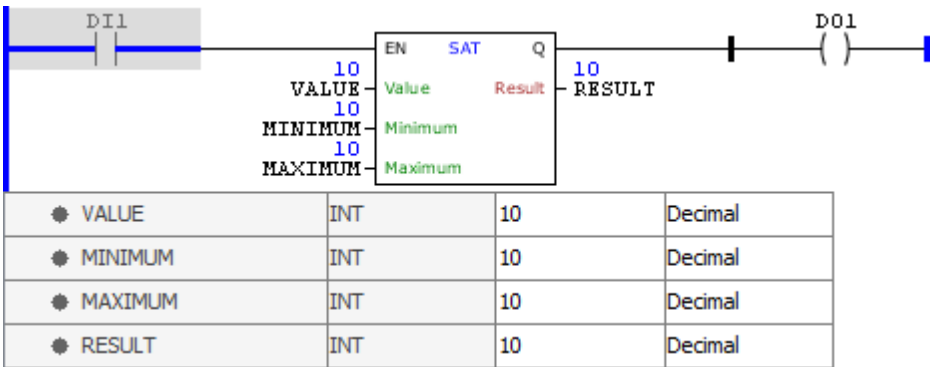
Este bloco, quando tem um valor TRUE em EN, realiza a comparação de Value com Minimum e Maximum. Se Value estiver na faixa entre Minimum e Maximum, Result recebe o valor de Value e Q permanece FALSE. Caso Value seja maior que Maximum, Result recebe Maximum e Q recebe TRUE. Caso Value seja menor que Minimum, Result recebe Minimum e Q recebe TRUE. Havendo algum erro na operação, Q não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Q permanece em FALSE.

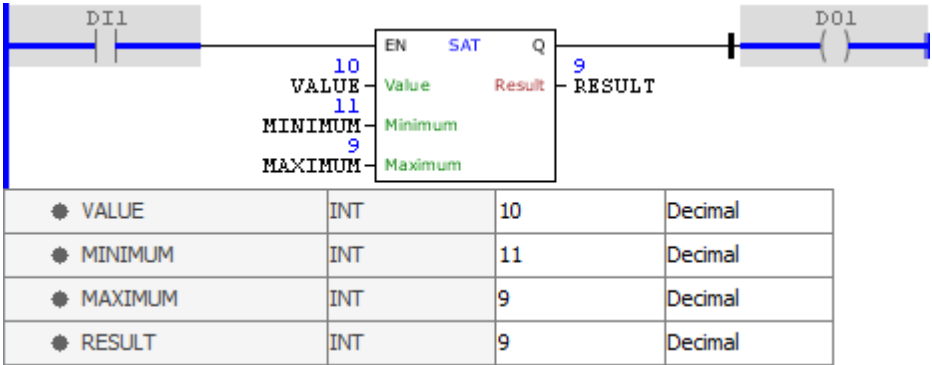
Fluxograma do Bloco



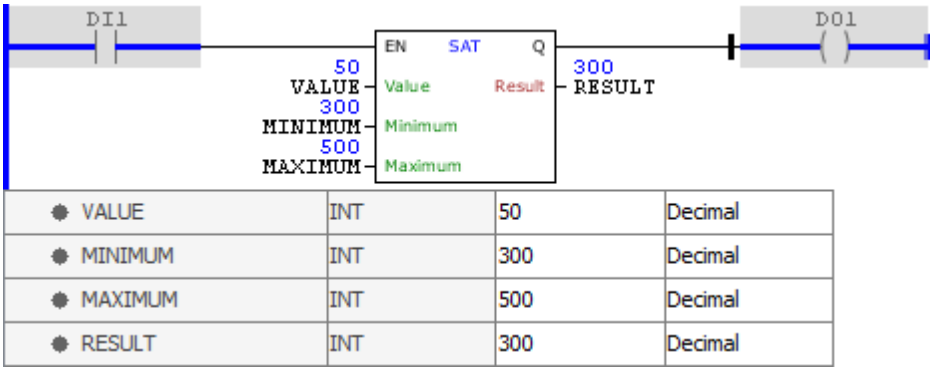
Exemplo



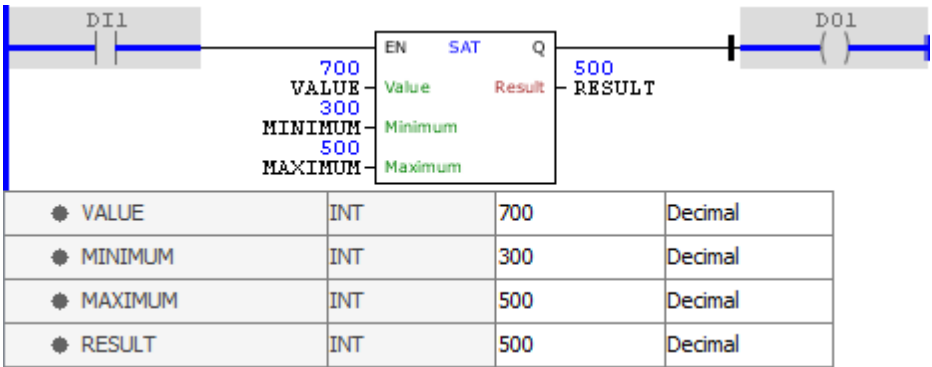
O exemplo acima repassa o valor de VALUE para RESULT, visto que ele não é inferior a MINIMUM ou superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é desativada, pois não houve saturação.



O exemplo acima repassa o valor de MAXIMUM para RESULT, visto que VALUE é superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.



O exemplo acima repassa o valor de MINIMUM para RESULT, visto que VALUE é inferior a MINIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.



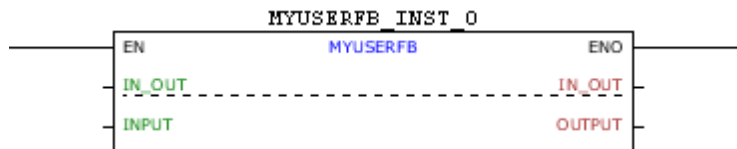
O exemplo acima repassa o valor de MAXIMUM para RESULT, visto que VALUE é superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.

10.1.5.1.11 Module

10.1.5.1.11.1 USERFB

Bloco que realiza uma subrotina programada pelo próprio usuário.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	INPUT	Conforme programação do usuário	Entradas do bloco
VAR_OUTPUT	ENO	BOOL	Fim de operação
	OUTPUT	Conforme programação do usuário	Saídas do bloco
VAR_IN_OUT	IN_OUT	Conforme programação do usuário	Variáveis utilizadas como entrada e saída do bloco
VAR	MYUSERFB_INST_0	MYUSERFB	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, atualiza os valores dos campos internos com as variáveis de entrada, executa a rotina Ladder programada pelo usuário e atualiza os valores das saídas após finalizada a rotina.

Quando EN possui valor FALSE, as saídas permanecem inalteradas.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

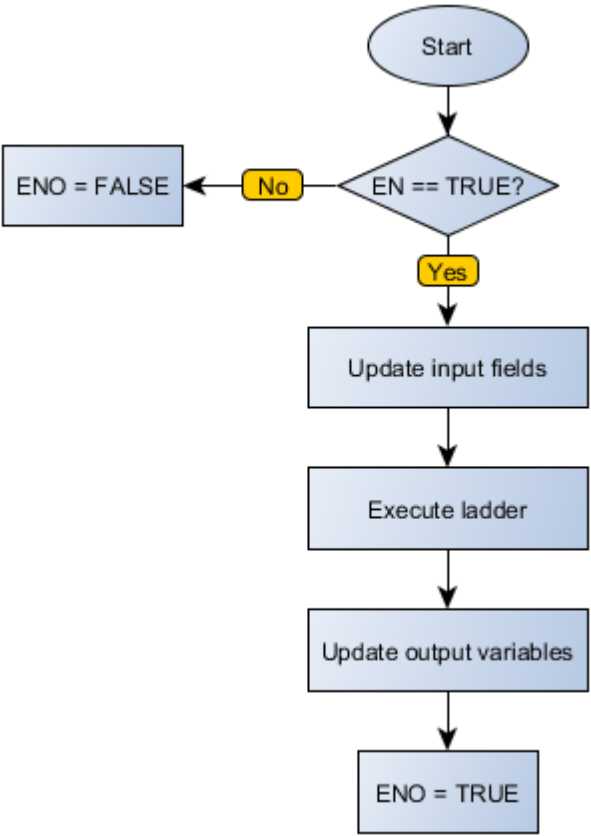
**NOTA!**

Consulte a seção [Trabalhando com USERFBs](#) para mais informações.

Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco

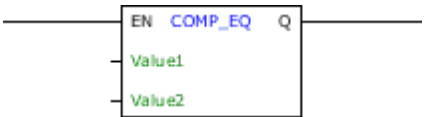


10.1.5.1.12 Compare

10.1.5.1.12.1 COMP_EQ

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso ambos sejam iguais.

Representação Ladder



Estrutura do Bloco

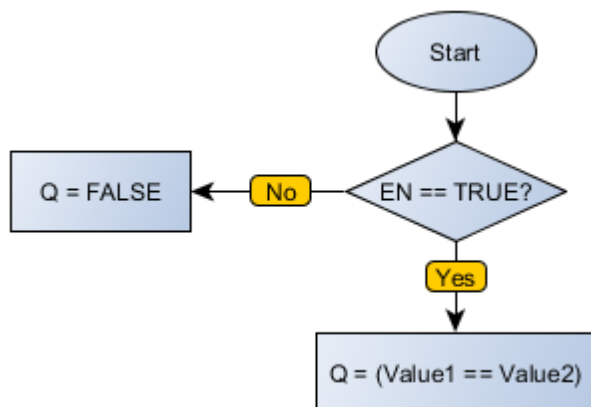
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade

Funcionamento

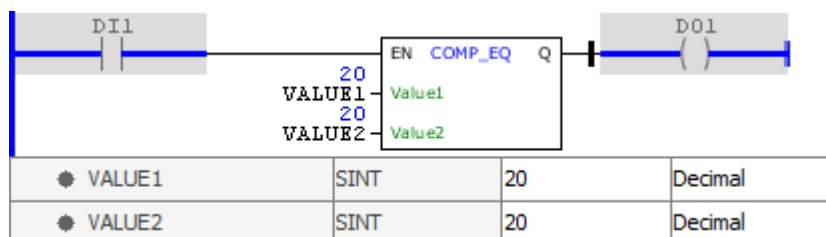
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 e Value2 sejam iguais. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

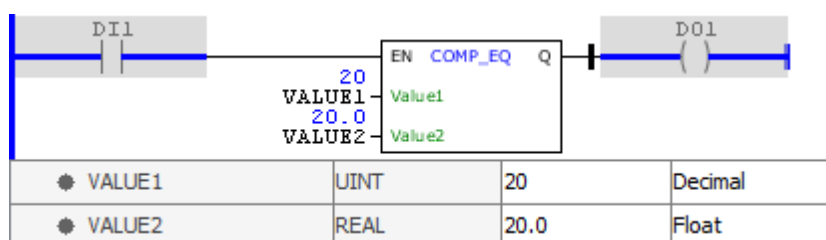
Fluxograma do Bloco



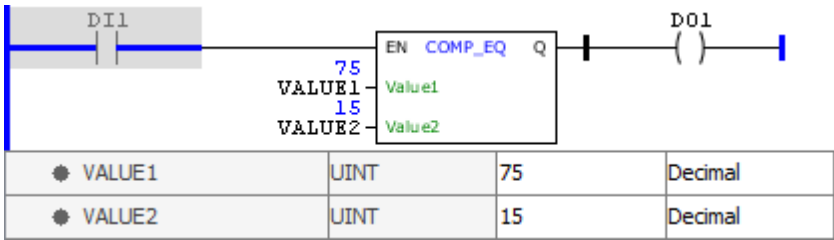
Exemplo



O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.



O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.



O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como as variáveis possuem valores diferentes, a saída Q é desativada.

10.1.5.1.12.2 COMP_GE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja maior ou igual a Value2.

Representação Ladder



Estrutura do Bloco

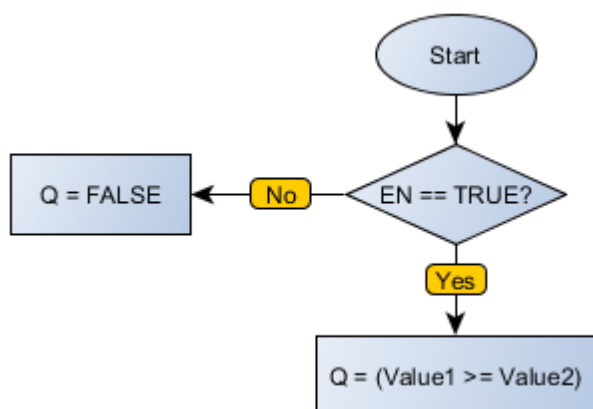
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade ou maioridade de Value1

Funcionamento

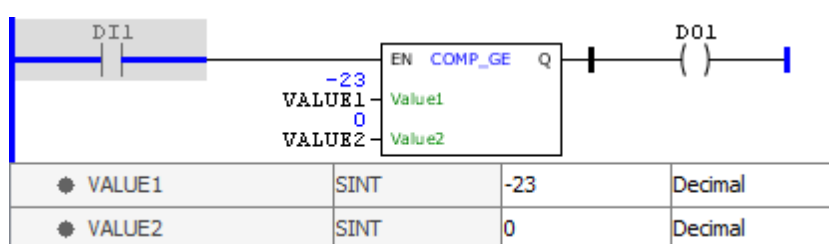
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja maior ou igual a Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

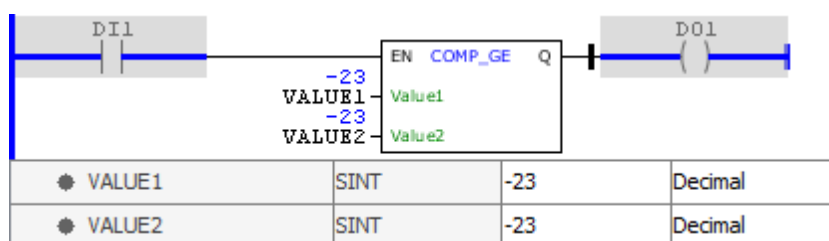
Fluxograma do Bloco



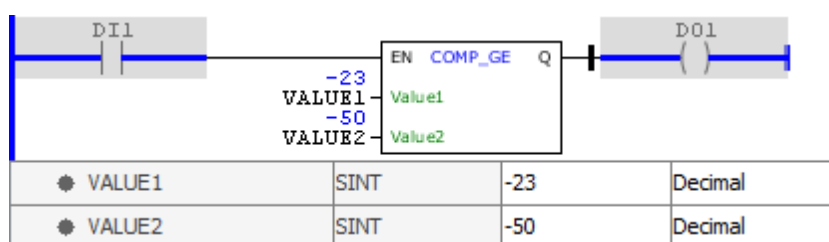
Exemplo



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é desativada.



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é ativada.

10.1.5.1.12.3 COMP_GT

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja maior que Value2.

Representação Ladder



Estrutura do Bloco

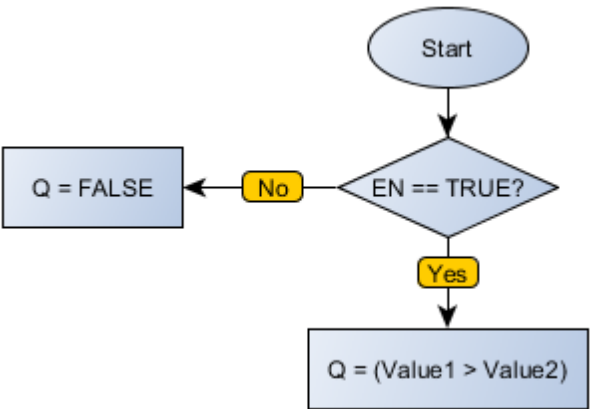
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de maioridade de Value1

Funcionamento

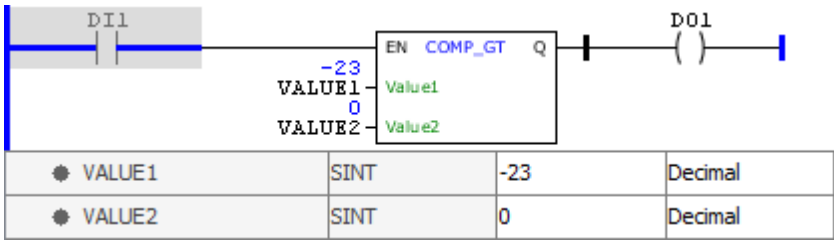
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja maior que Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

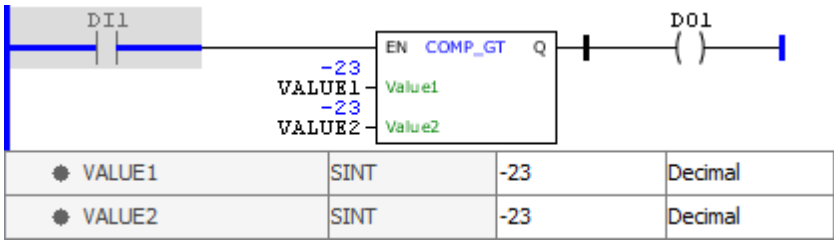
Fluxograma do Bloco



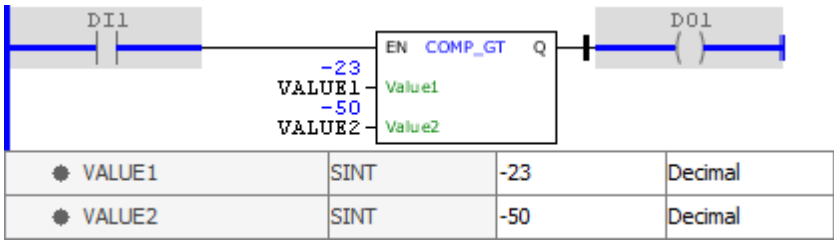
Exemplo



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é desativada.



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é ativada.

10.1.5.1.12.4 COMP_LE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja menor ou igual a Value2.

Representação Ladder



Estrutura do Bloco

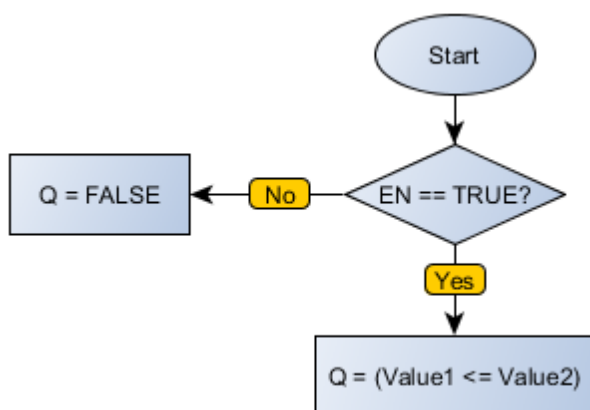
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade ou minoridade de Value1

Funcionamento

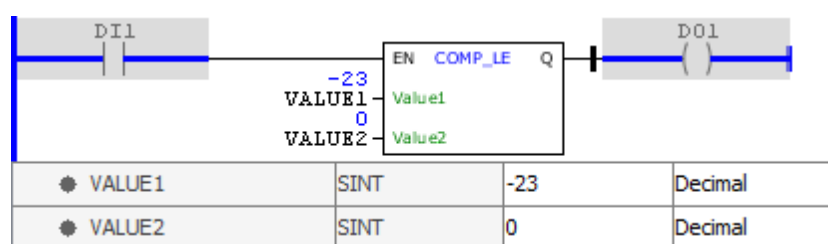
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja menor ou igual a Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

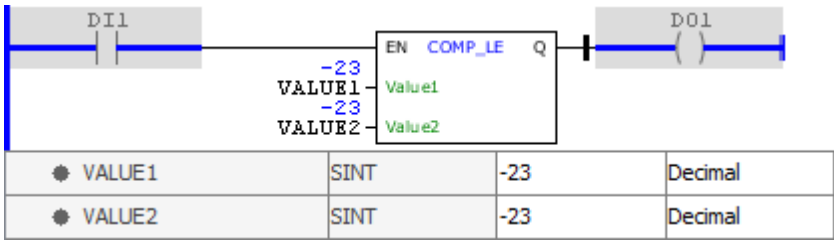
Fluxograma do Bloco



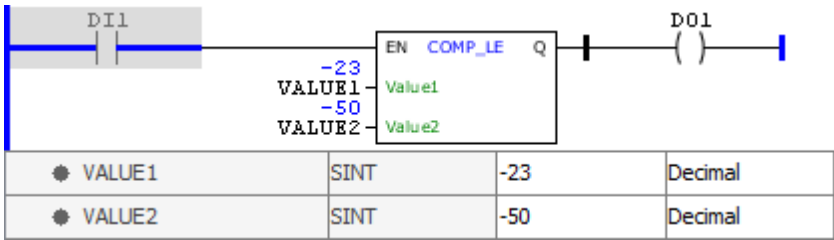
Exemplo



O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é ativada.



O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.

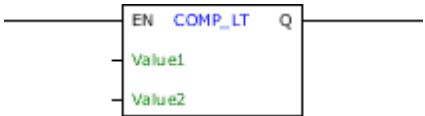


O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é desativada.

10.1.5.1.12.5 COMP_LT

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja menor que Value2.

Representação Ladder



Estrutura do Bloco

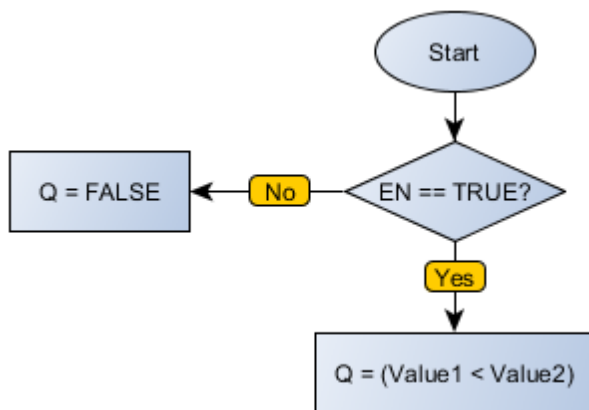
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de minoridade de Value1

Funcionamento

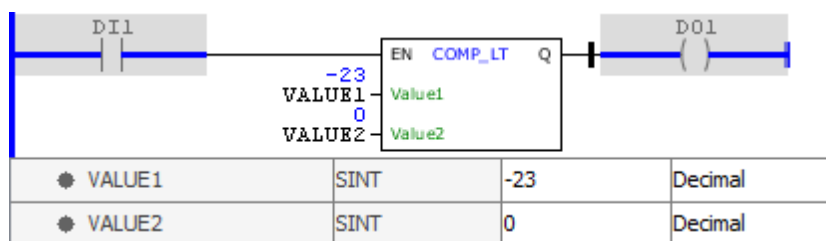
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja menor que Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

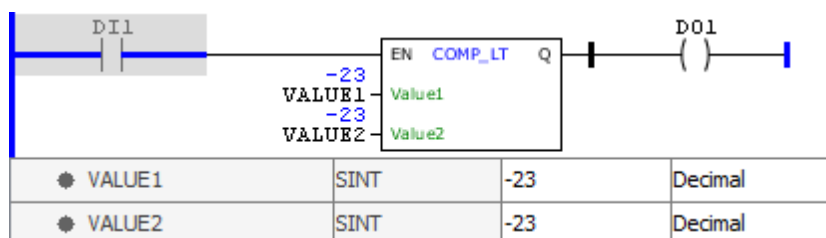
Fluxograma do Bloco



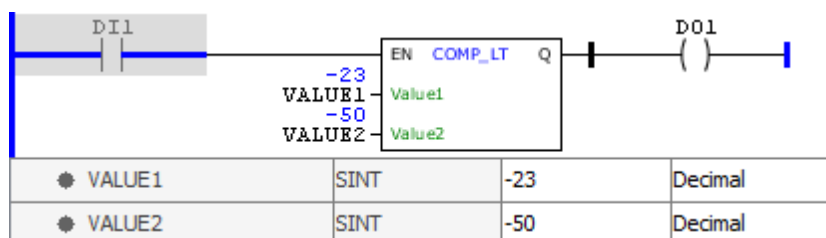
Exemplo



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é ativada.



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é desativada.

10.1.5.1.12.6 COMP_NE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja diferente de Value2.

Representação Ladder



Estrutura do Bloco

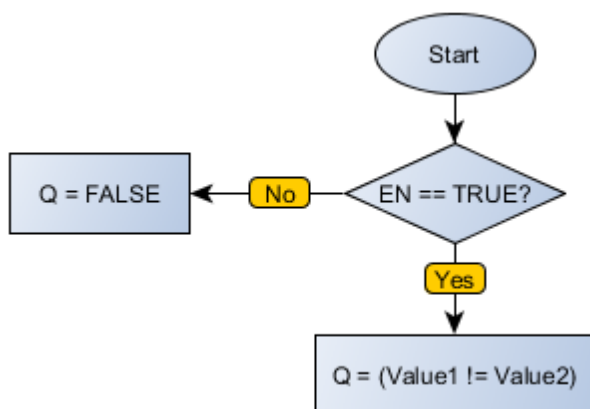
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de desigualdade

Funcionamento

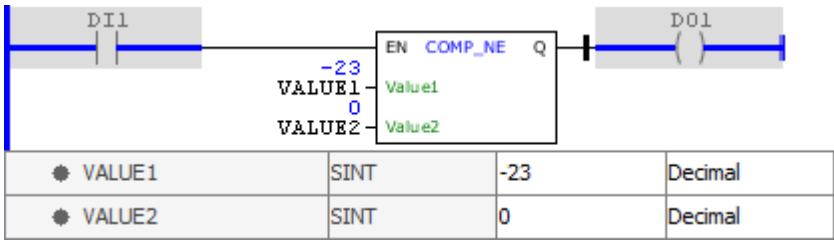
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja diferente de Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

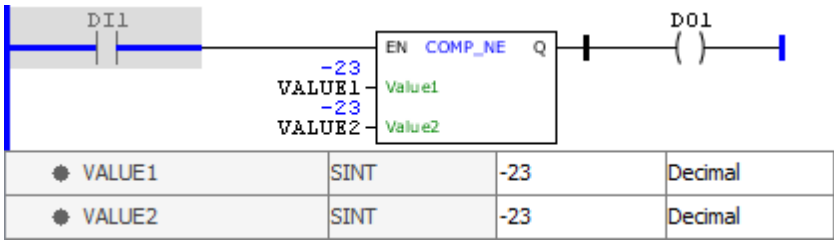
Fluxograma do Bloco



Exemplo



O exemplo acima verifica a desigualdade entre VALUE1 e VALUE2. Como as variáveis possuem valores diferentes, a saída Q é ativada.



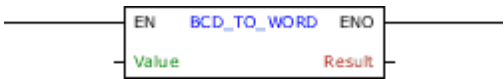
O exemplo acima verifica a desigualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.

10.1.5.1.13 Conversion

10.1.5.1.13.1 BCD

Bloco que realiza a conversão de um código BCD para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em BCD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

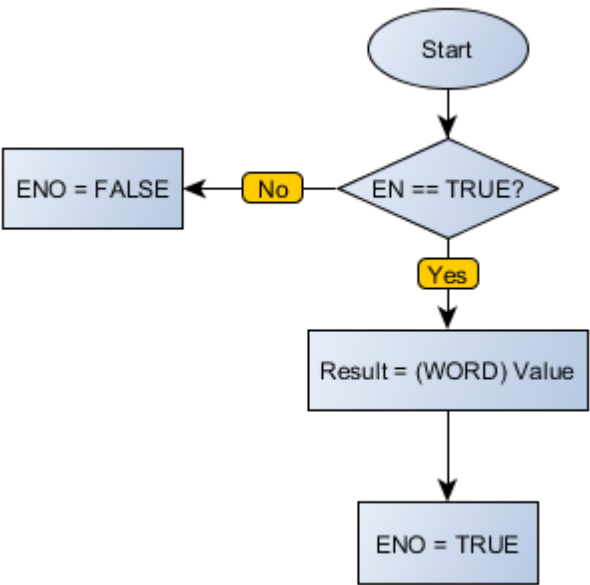
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BCD e realiza a conversão para WORD, armazenando em Result.

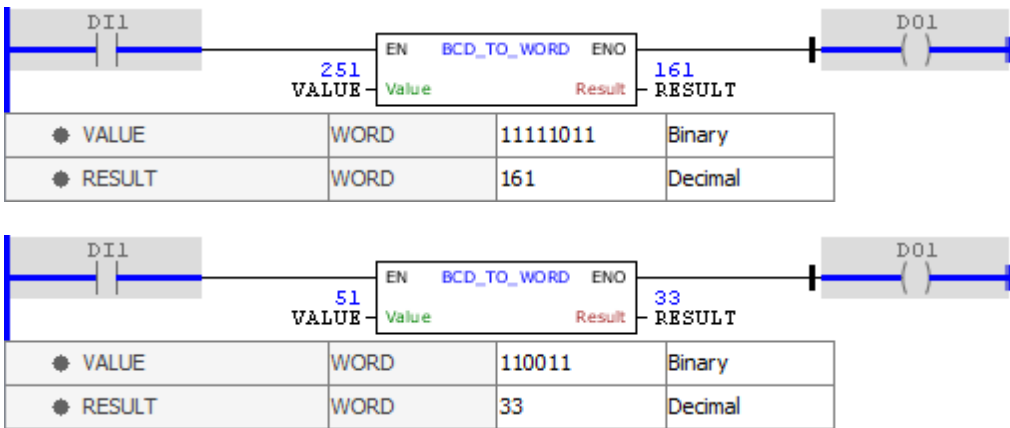
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



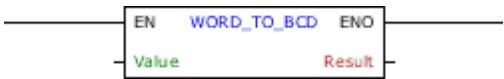
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BCD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um código BCD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em BCD

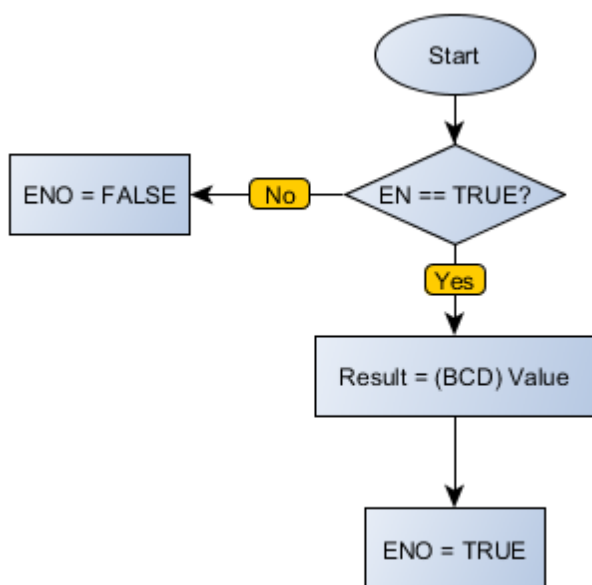
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BCD, armazenando em Result.

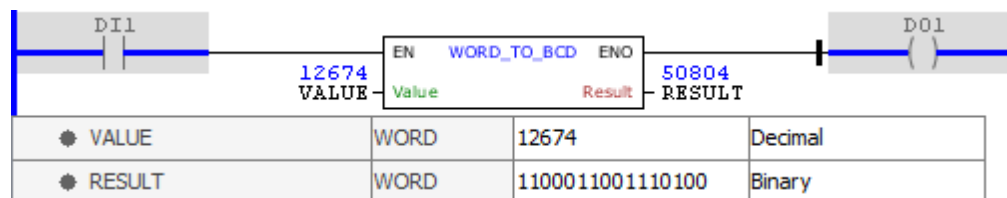
Quando EN possui valor FALSE, Result permanece inalterado.

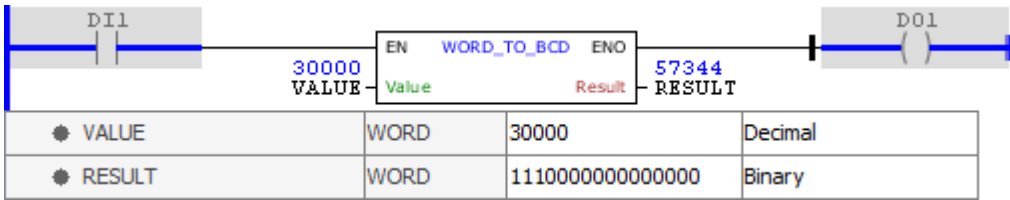
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



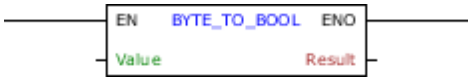


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BCD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.13.2 BOOL

Bloco que realiza a conversão de um valor BYTE para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

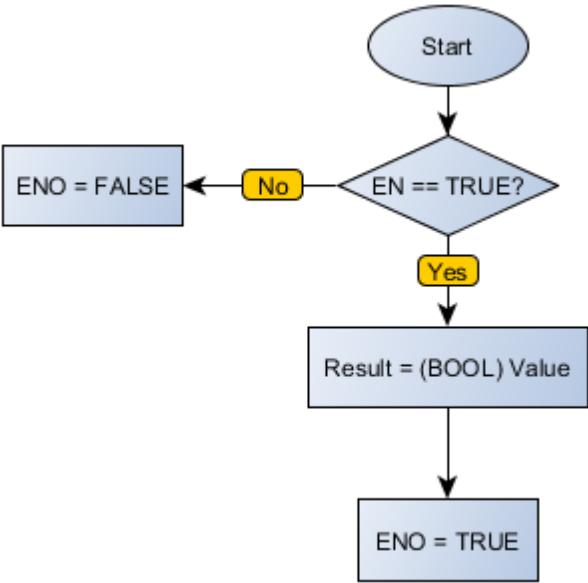
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para BOOL, armazenando em Result.

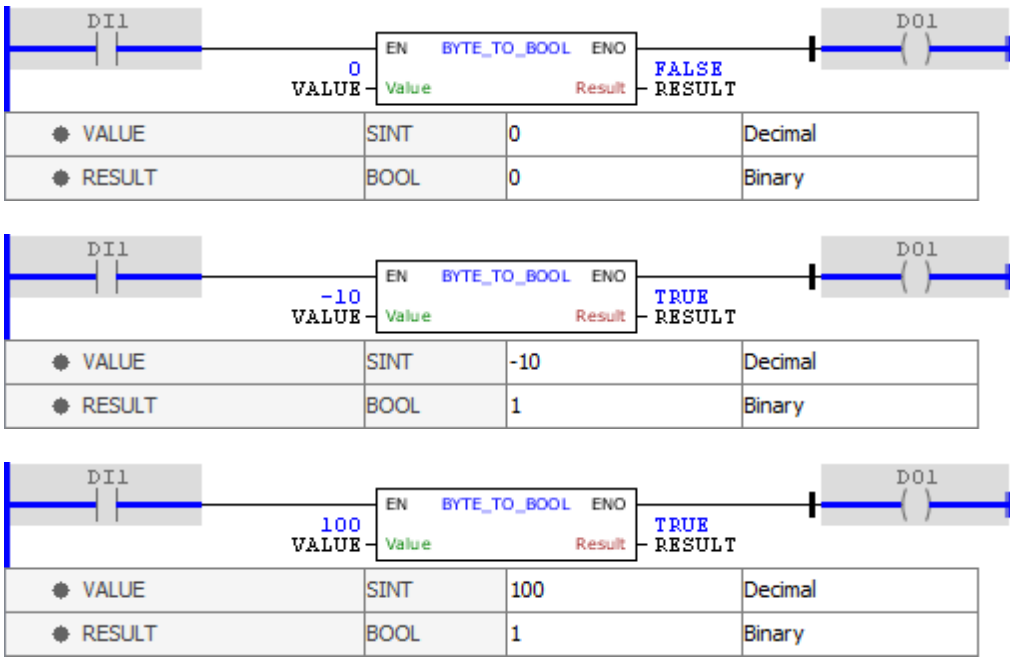
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



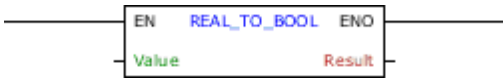
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

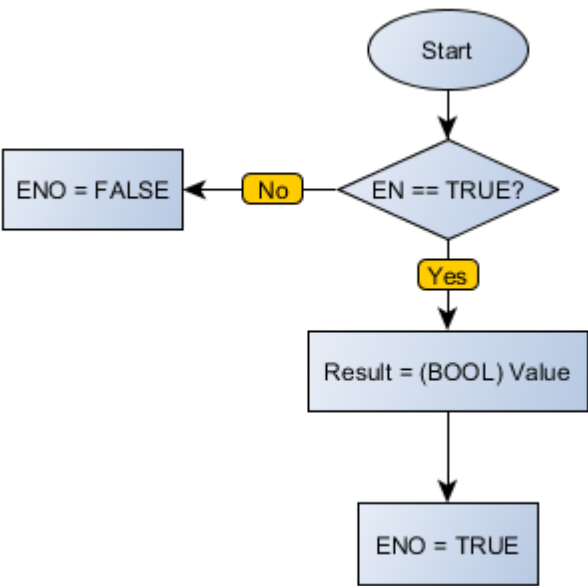
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para BOOL, armazenando em Result.

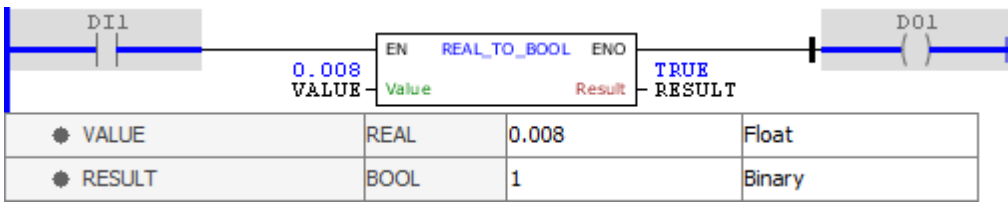
Quando EN possui valor FALSE, Result permanece inalterado.

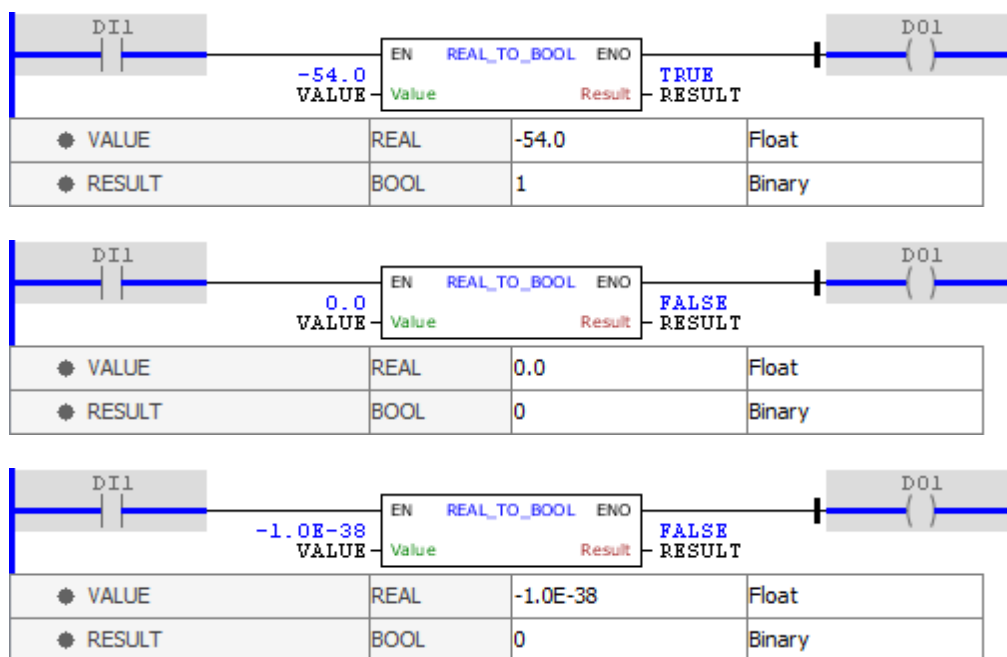
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

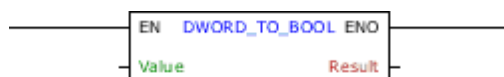




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe no último exemplo que valores muito próximos do epsilon da máquina podem acarretar numa interpretação de valor FALSE.

Bloco que realiza a conversão de um valor DWORD para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

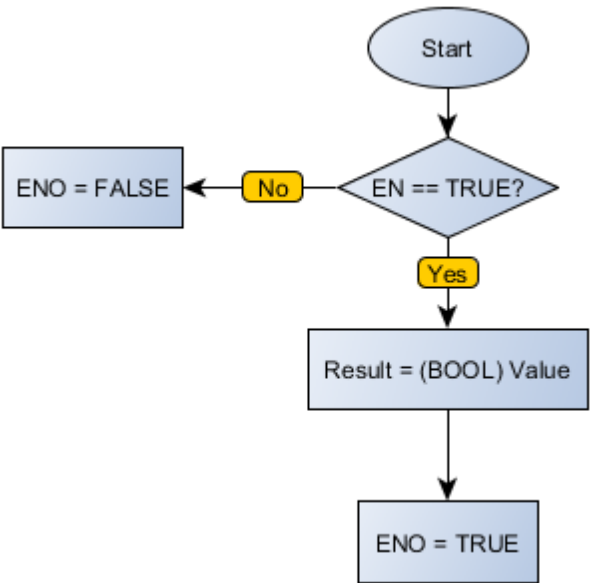
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para BOOL, armazenando em Result.

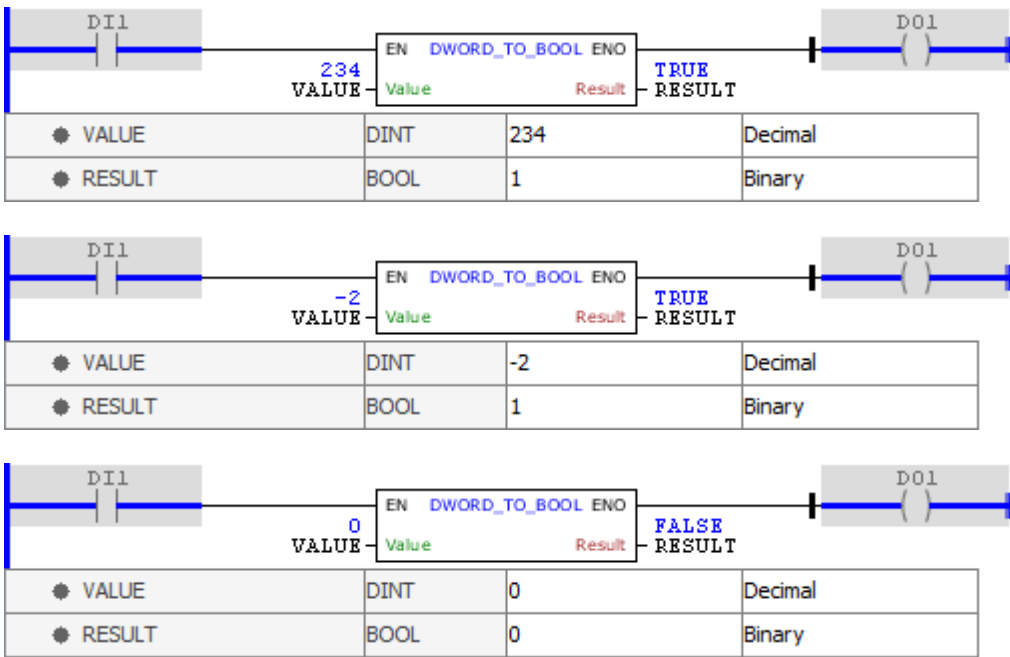
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



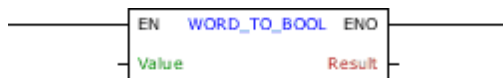
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

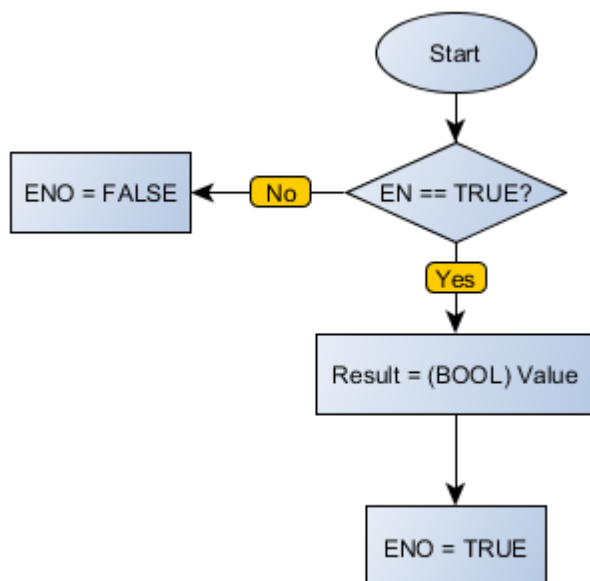
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BOOL, armazenando em Result.

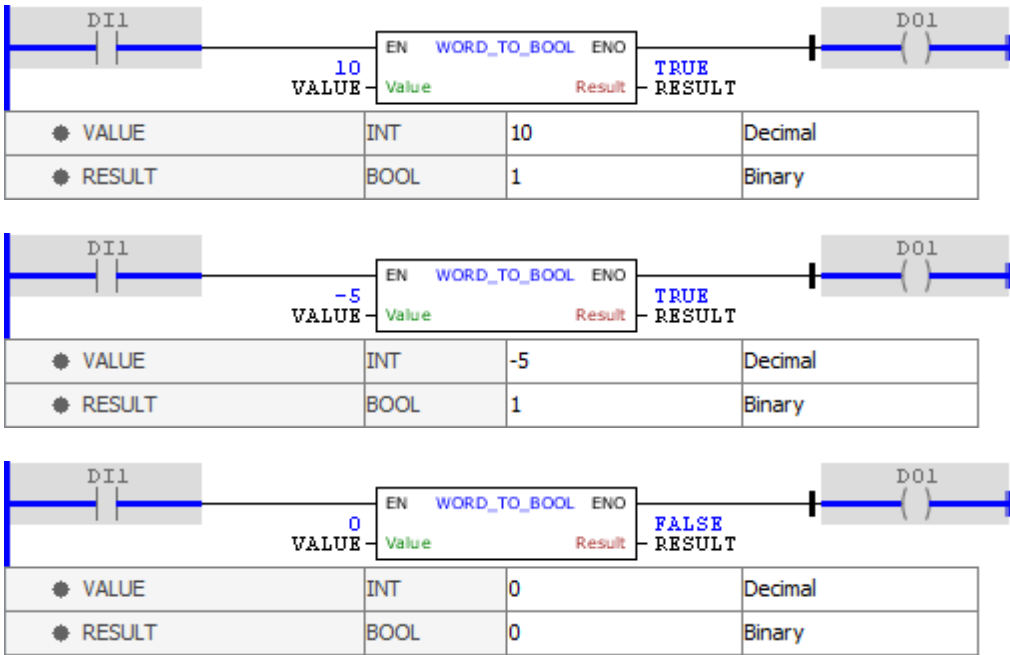
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

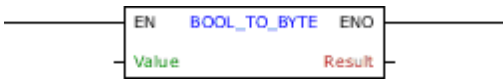


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.13.3 BYTE

Bloco que realiza a conversão de um valor BOOL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

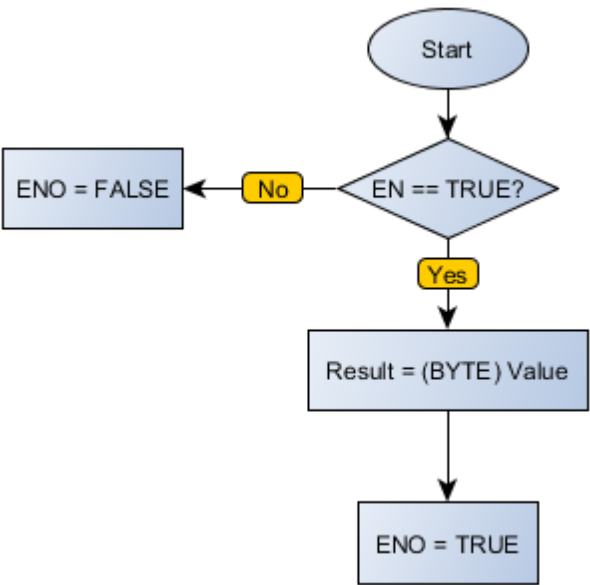
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para BYTE, armazenando em Result.

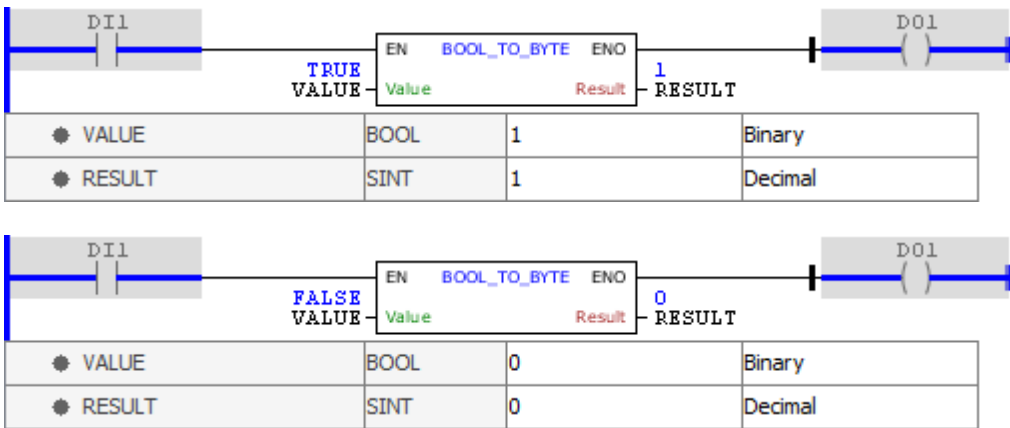
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



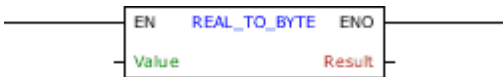
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

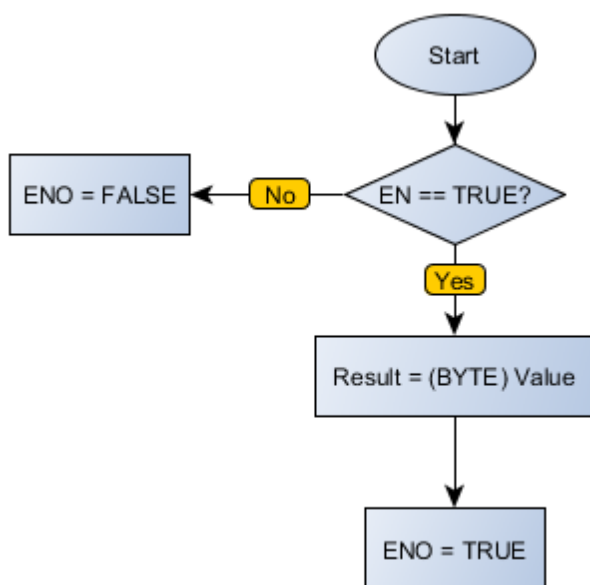
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para BYTE, armazenando em Result.

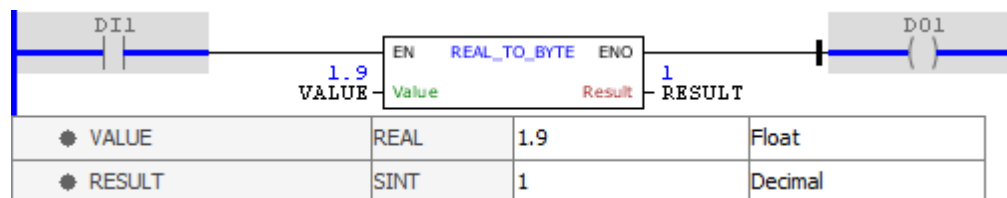
Quando EN possui valor FALSE, Result permanece inalterado.

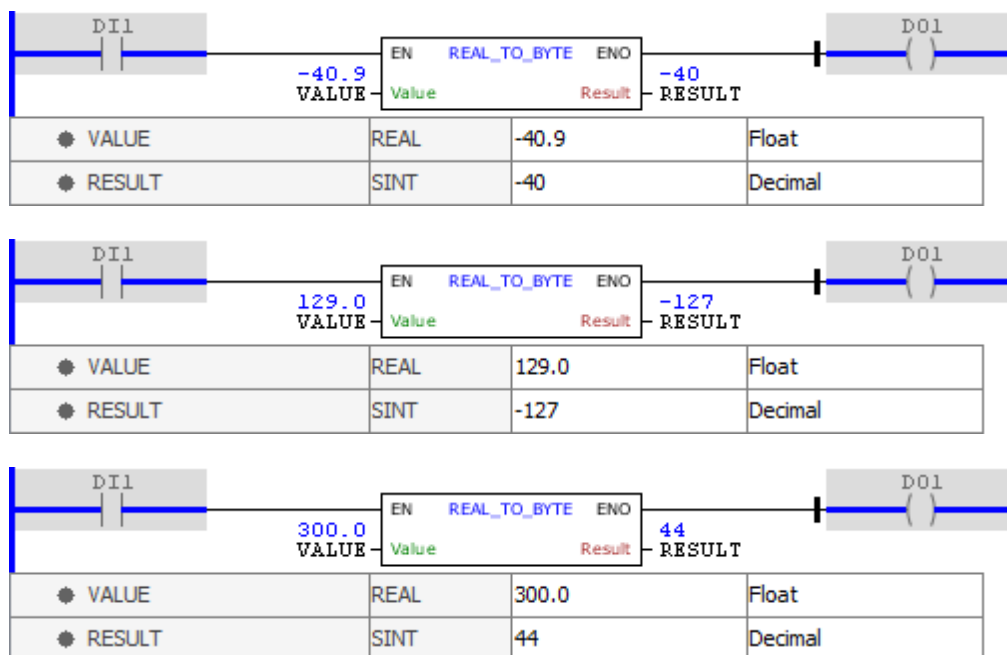
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

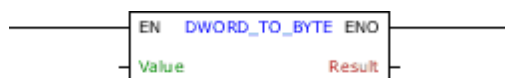




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os oito bits menos significativos.

Bloco que realiza a conversão de um valor DWORD para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

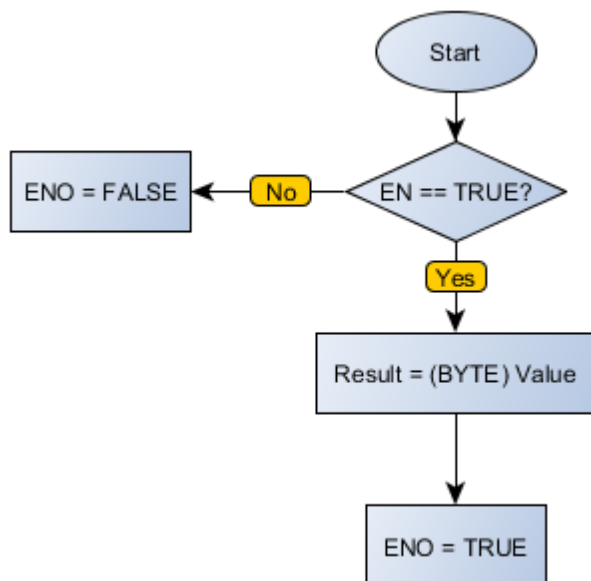
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para BYTE, armazenando em Result.

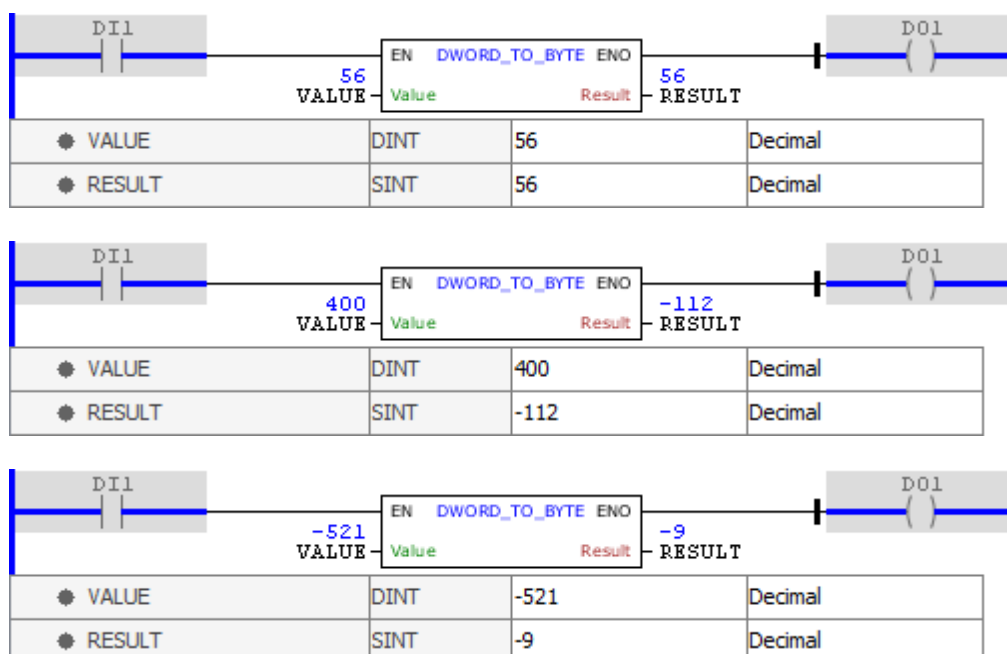
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



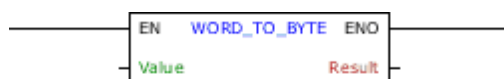
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os oito bits menos significativos.

Bloco que realiza a conversão de um valor WORD para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

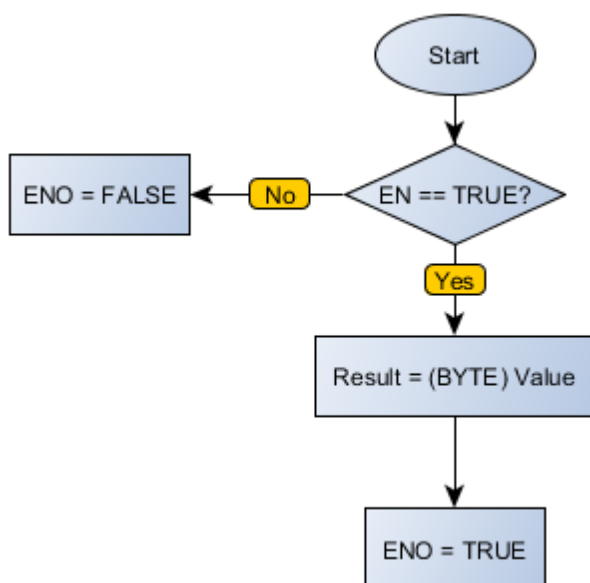
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BYTE, armazenando em Result.

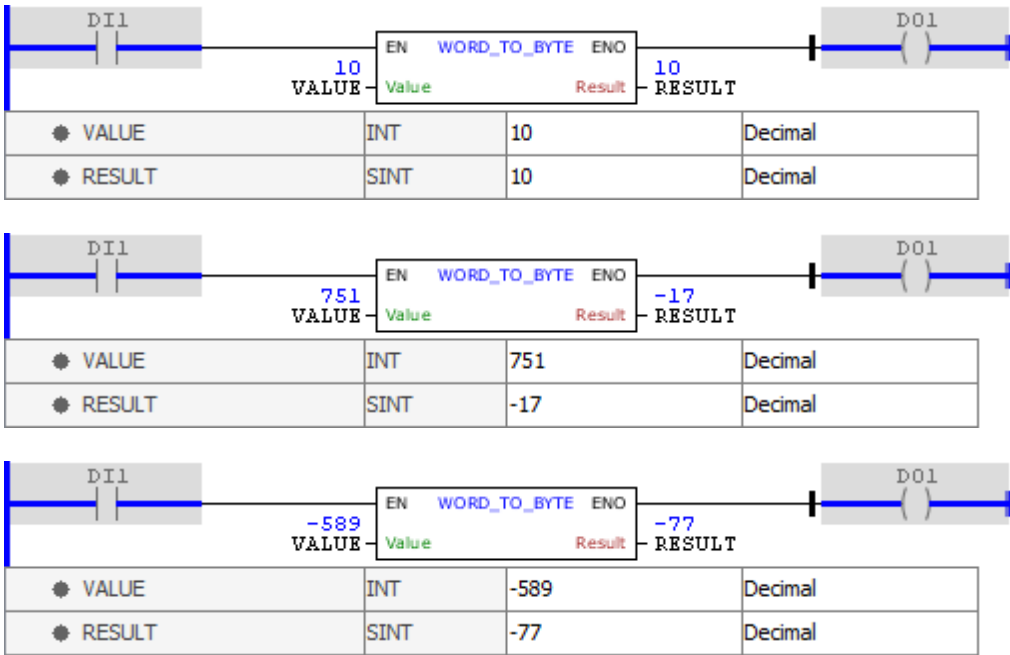
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

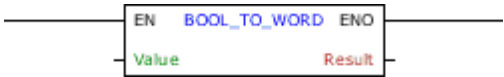


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os oito bits menos significativos.

10.1.5.1.13.4 WORD

Bloco que realiza a conversão de um valor BOOL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

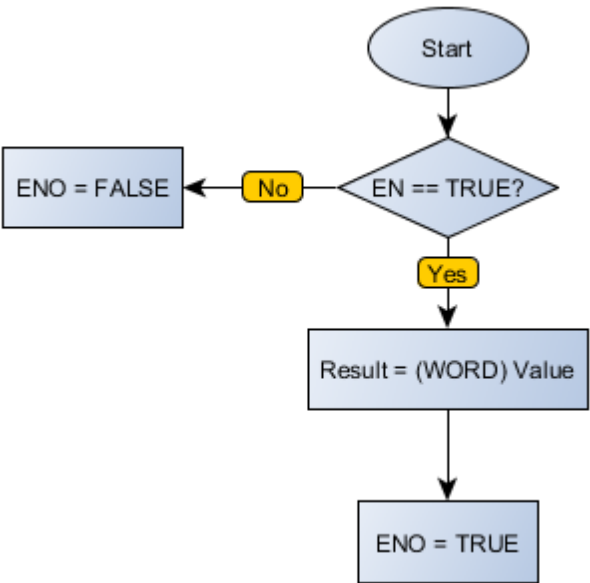
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para WORD, armazenando em Result.

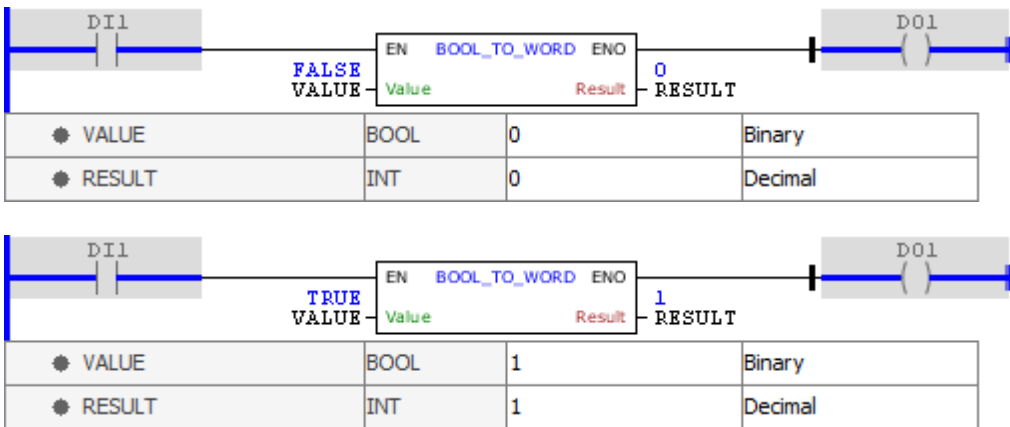
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



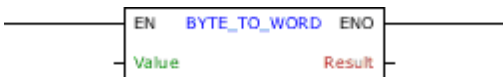
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

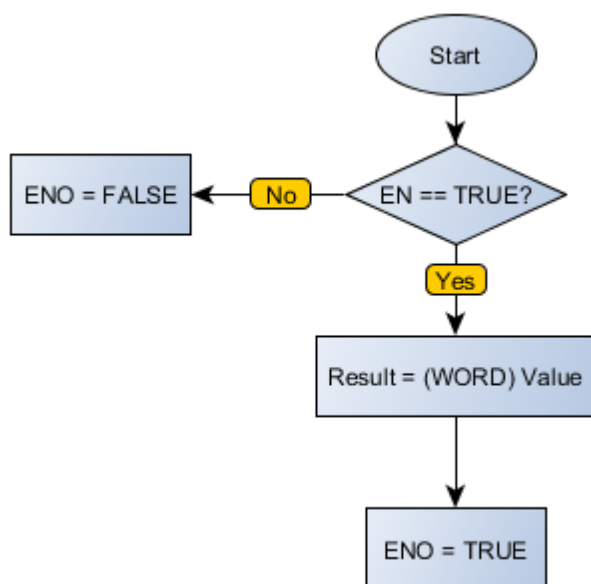
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para WORD, armazenando em Result.

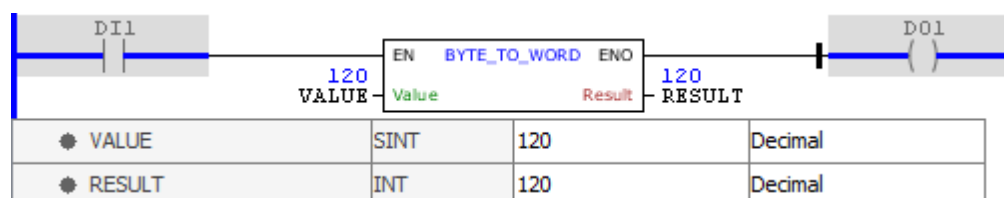
Quando EN possui valor FALSE, Result permanece inalterado.

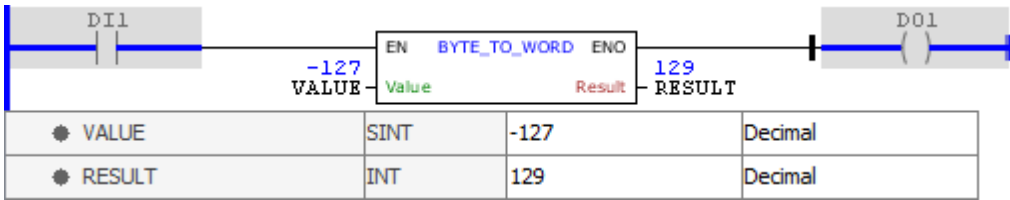
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

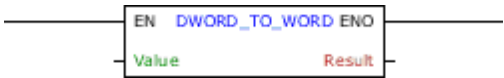




Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor DWORD para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

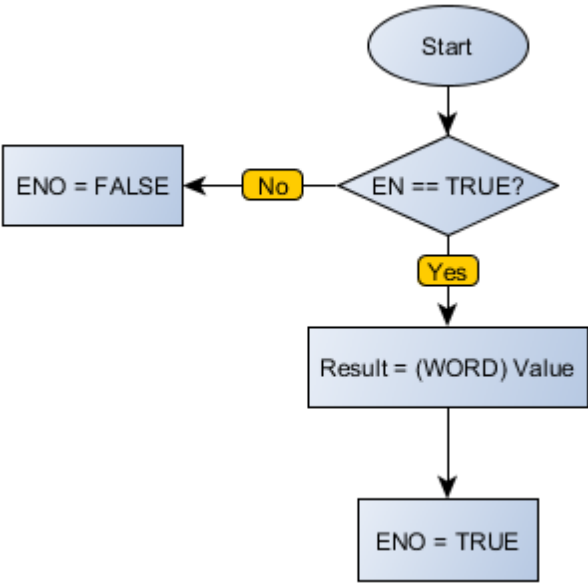
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para WORD, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

VALUE	DINT	30654	Decimal
RESULT	INT	30654	Decimal

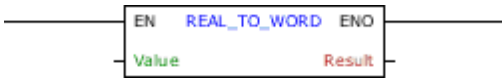
VALUE	DINT	38412	Decimal
RESULT	INT	-27124	Decimal

VALUE	DINT	-45896	Decimal
RESULT	INT	19640	Decimal

Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os dezesseis bits menos significativos.

Bloco que realiza a conversão de um valor REAL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

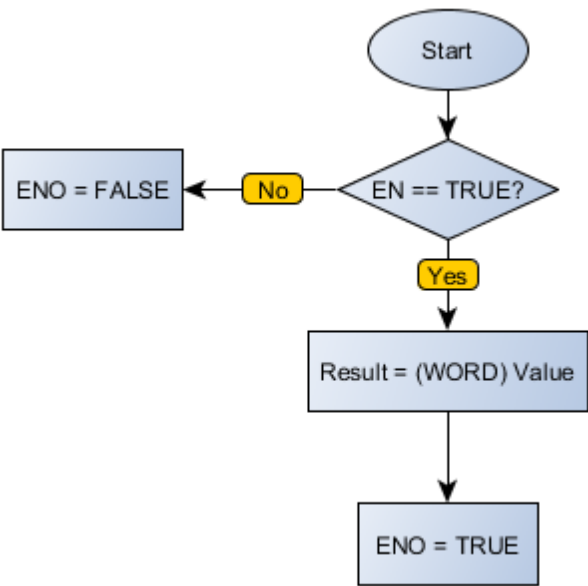
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para WORD, armazenando em Result.

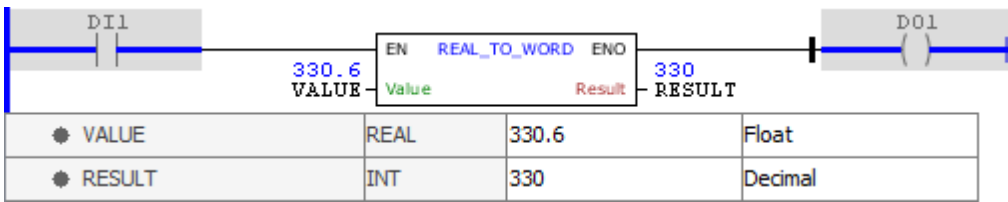
Quando EN possui valor FALSE, Result permanece inalterado.

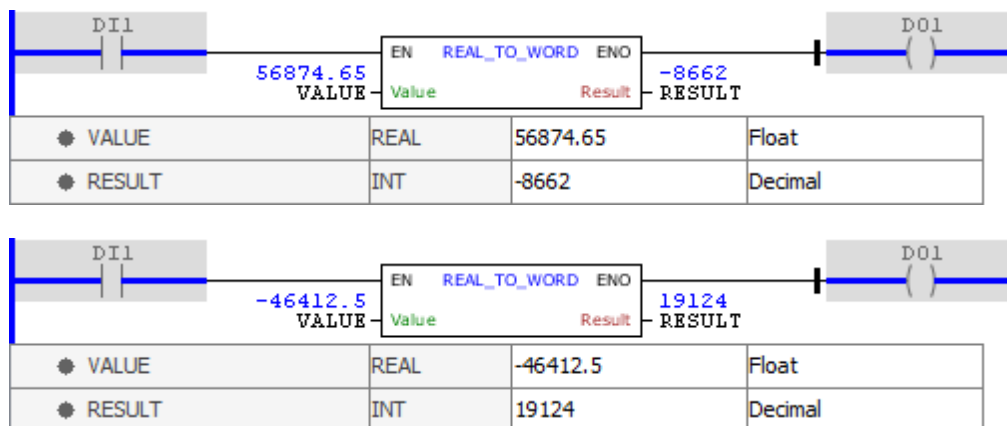
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



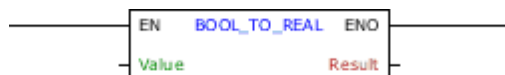


Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os dezesseis bits menos significativos.

10.1.5.1.13.5 REAL

Bloco que realiza a conversão de um valor BOOL para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

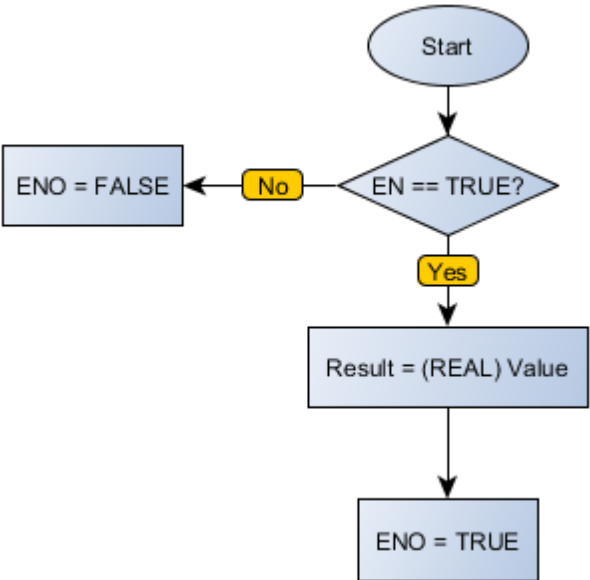
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para REAL, armazenando em Result.

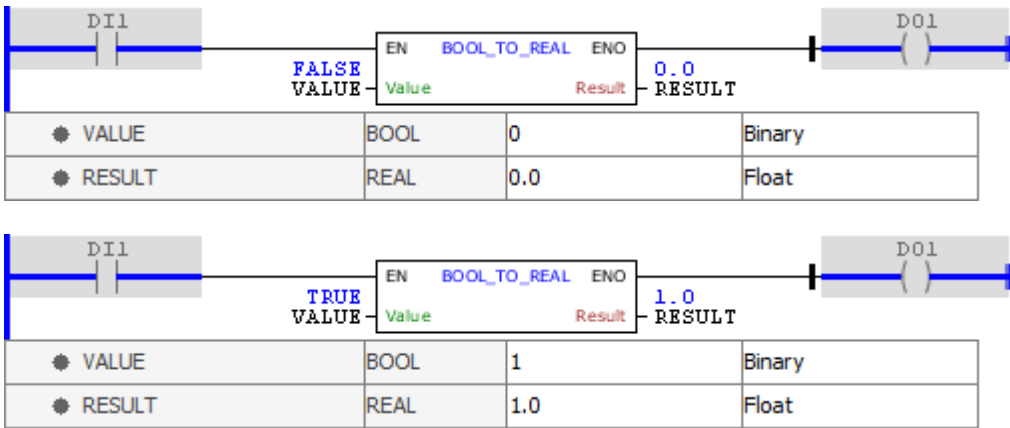
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



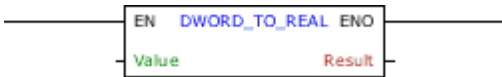
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor DWORD para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

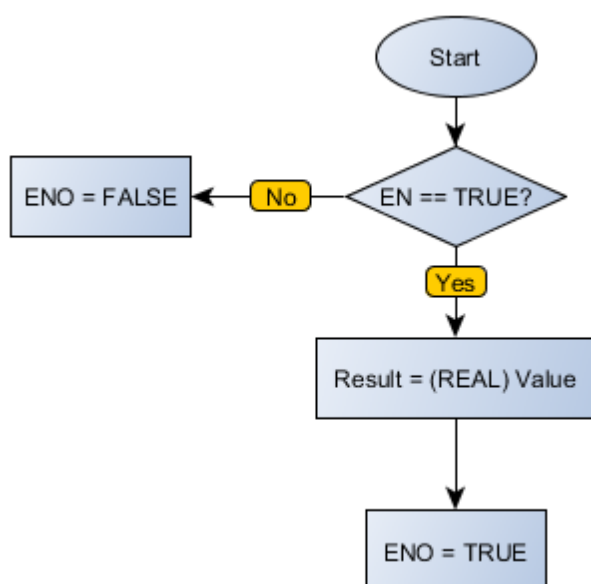
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para REAL, armazenando em Result.

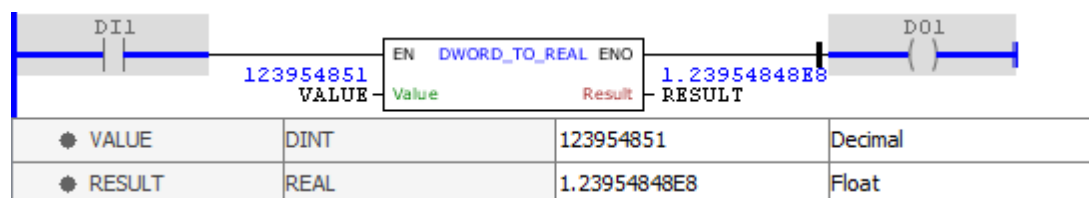
Quando EN possui valor FALSE, Result permanece inalterado.

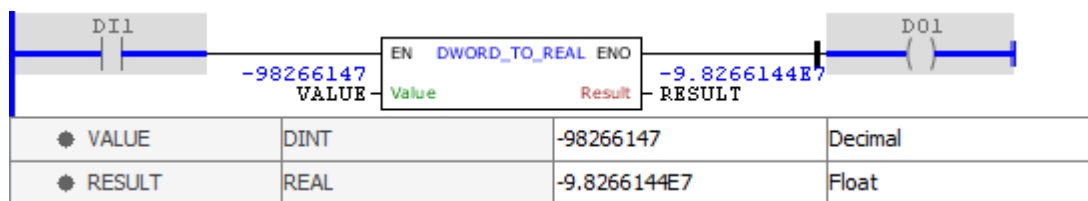
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

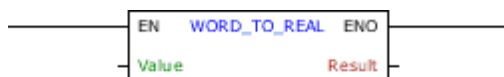




Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

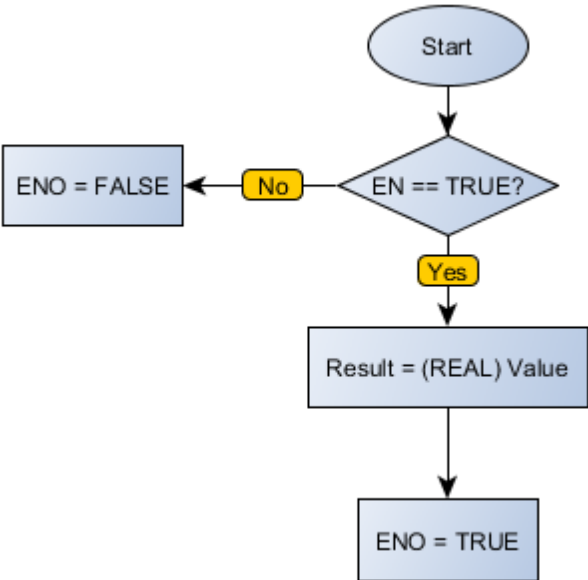
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para REAL, armazenando em Result.

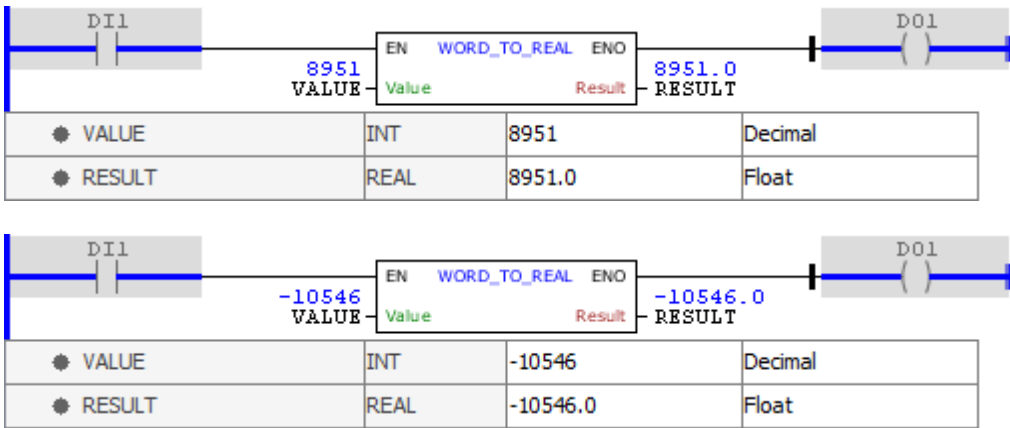
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



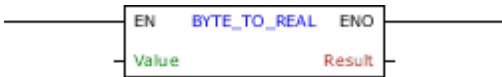
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

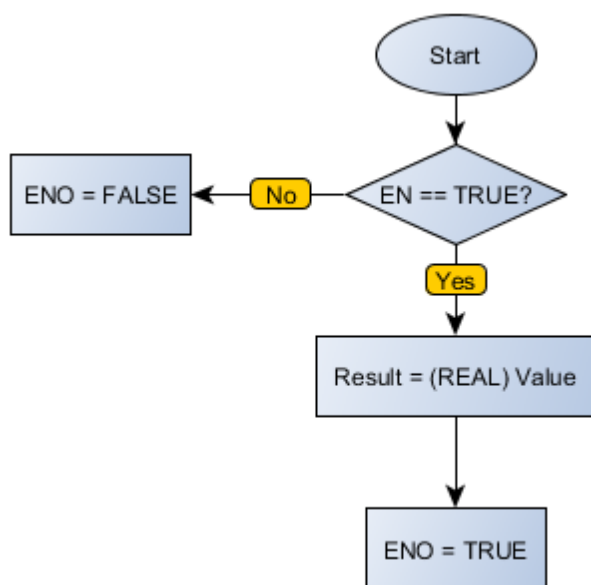
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para REAL, armazenando em Result.

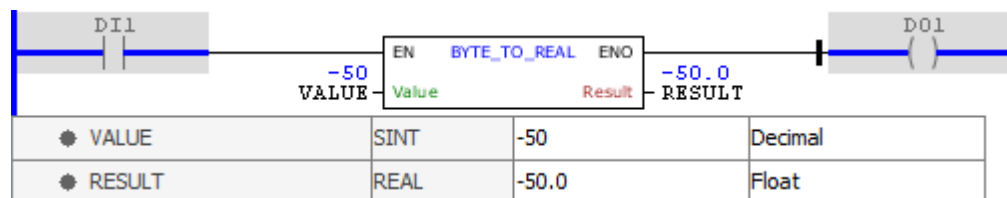
Quando EN possui valor FALSE, Result permanece inalterado.

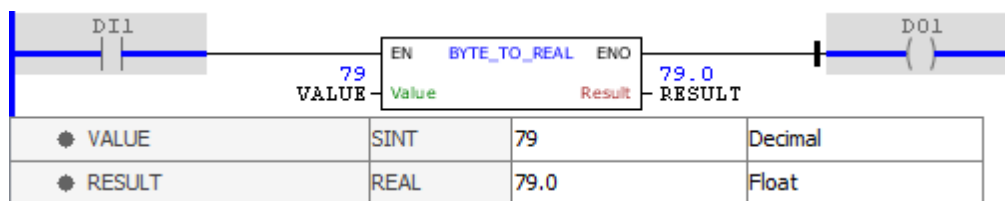
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

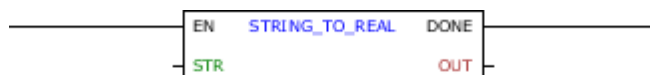




Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor STRING para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR	STRING	Valor em STRING
VAR_OUTPUT	DONE	BOOL	Fim de operação
	OUT	REAL	Valor em REAL

Funcionamento

Este bloco, quando tem um valor TRUE em EN, lê o valor de STR como STRING caractere a caractere, realizando a conversão para REAL e armazenando em DONE. Caso o primeiro caractere não seja matematicamente válido, a saída OUT recebe zero. Caso haja caracteres válidos, esses caracteres serão convertidos até o fim da STRING, ou até encontrar um caractere não válido.



NOTA!

O bloco interpreta o ponto (.) como separador de decimais, e não a vírgula (,).

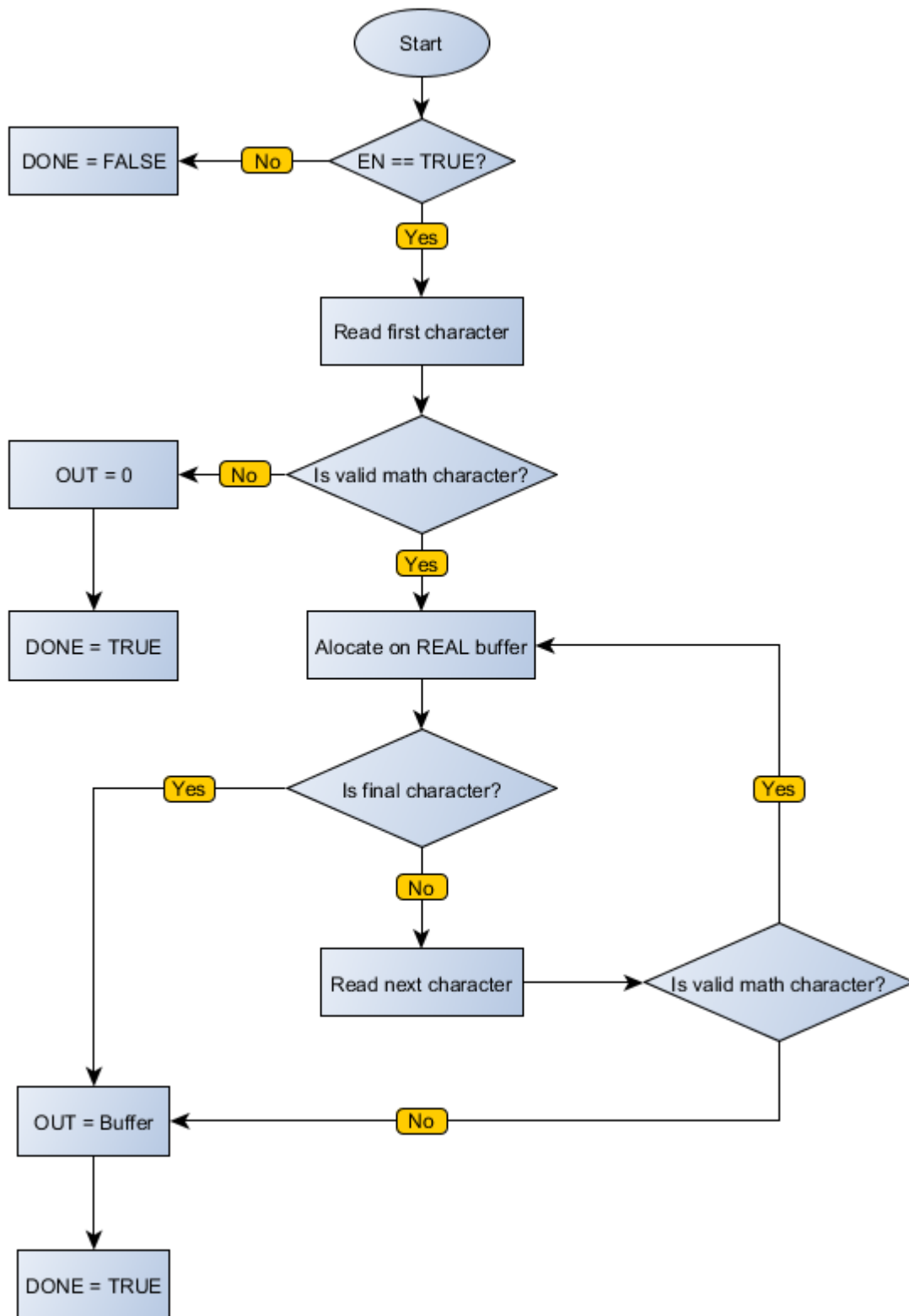
Quando EN possui valor FALSE, OUT permanece inalterado e DONE permanece FALSE.

O valor de DONE repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Compatibilidade

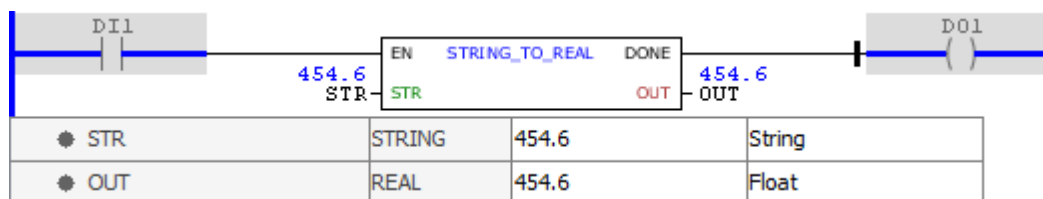
Equipamento	Versão
PLC300	2.10 ou superior

Fluxograma do Bloco

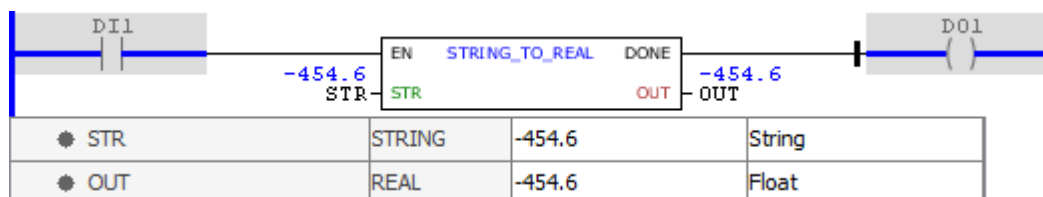


Exemplo

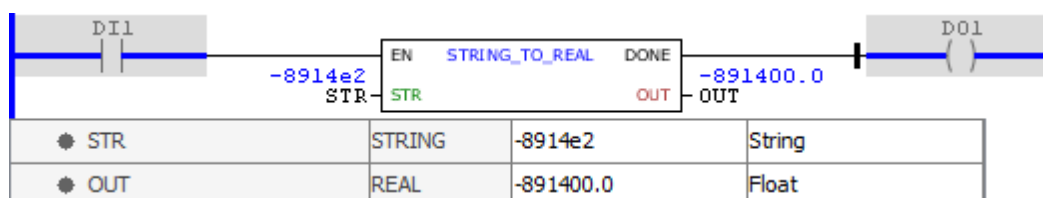
Os exemplos a seguir exibem diversas conversões de STRING em valores do tipo REAL. Todas as conversões habilitam a saída DONE ao fim da operação.



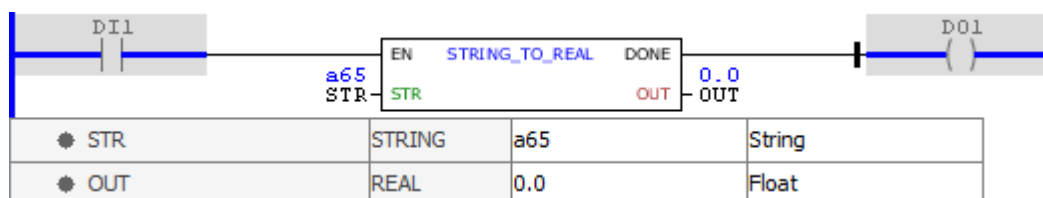
A conversão acima foi realizada com sucesso. O ponto decimal é um caractere matemático válido.



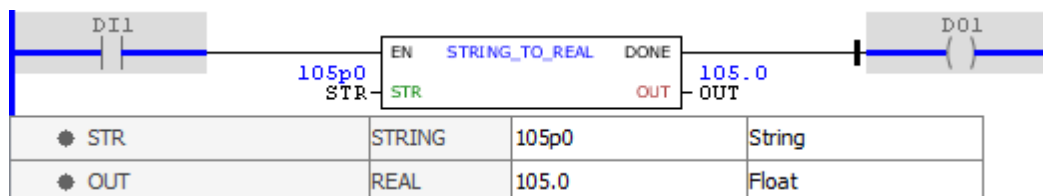
A conversão acima foi realizada com sucesso. O traço e ponto decimal são caracteres matemáticos válidos.



A conversão acima foi realizada com sucesso. O traço e o indicador de potência de 10 "e" são caracteres matemáticos válidos.



A conversão acima não foi realizada com sucesso. O primeiro caractere não foi identificado como matematicamente válido, e a saída foi zerada.

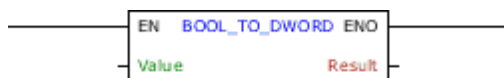


A conversão acima foi realizada com sucesso. O caractere "p" não é matematicamente válido e encerra a conversão, truncando o resultado ao que já havia sido convertido.

10.1.5.1.13.6 DWORD

Bloco que realiza a conversão de um valor BOOL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

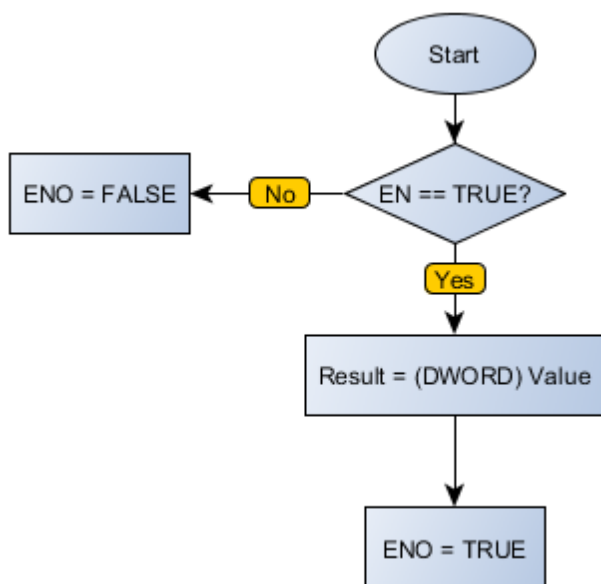
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para DWORD, armazenando em Result.

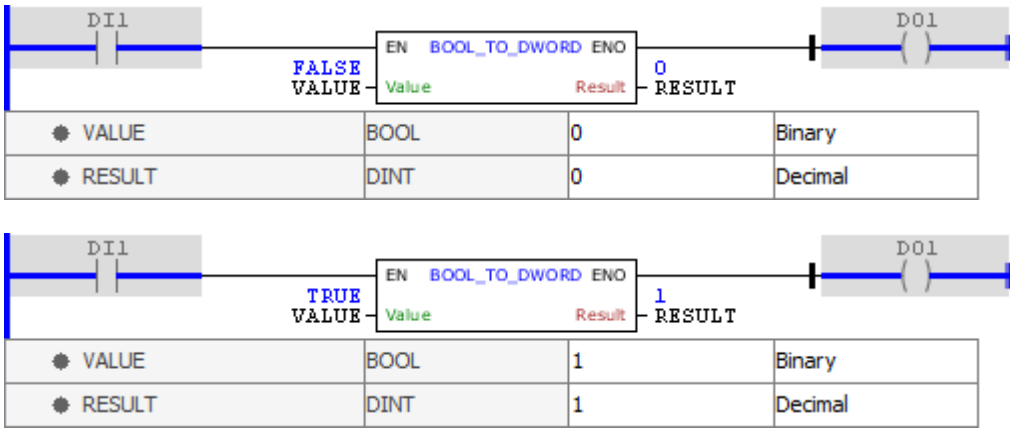
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



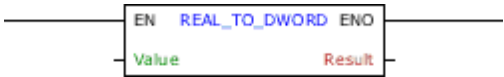
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

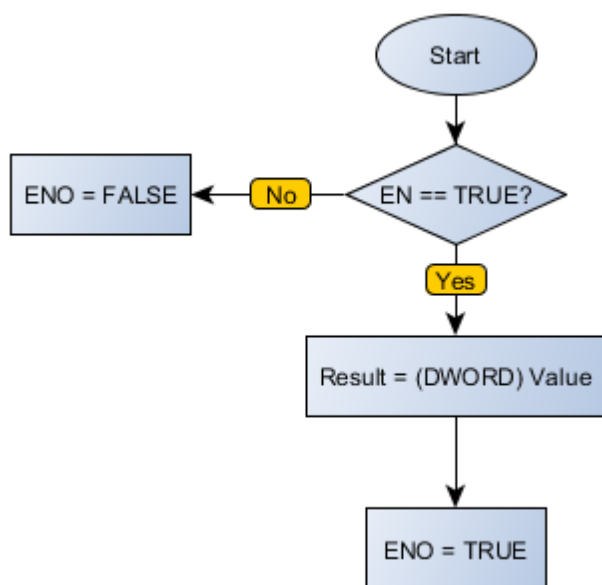
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para DWORD, armazenando em Result.

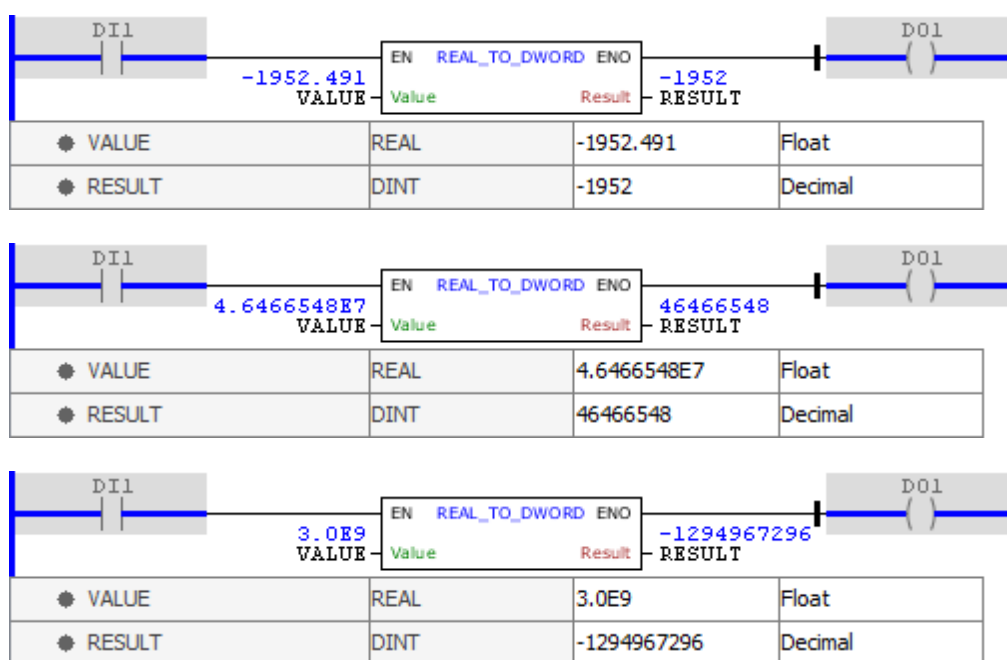
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



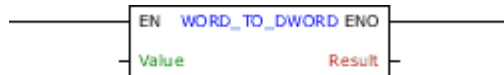
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os trinta e dois bits menos significativos.

Bloco que realiza a conversão de um valor WORD para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

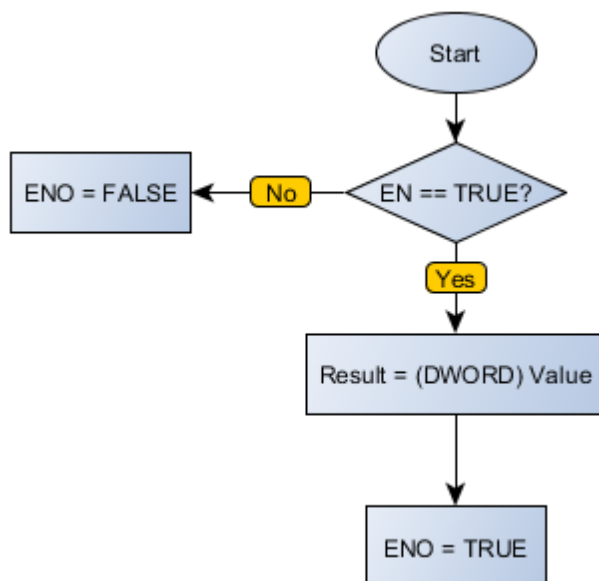
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para DWORD, armazenando em Result.

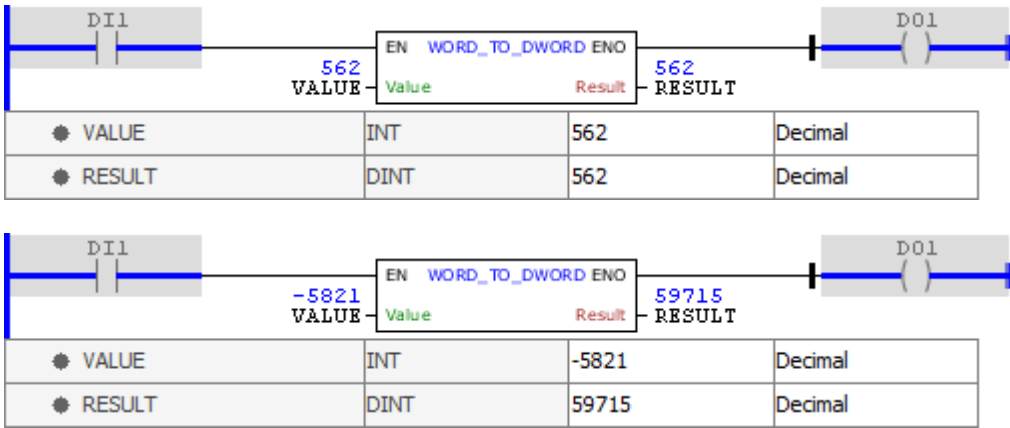
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



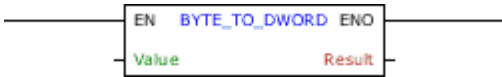
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

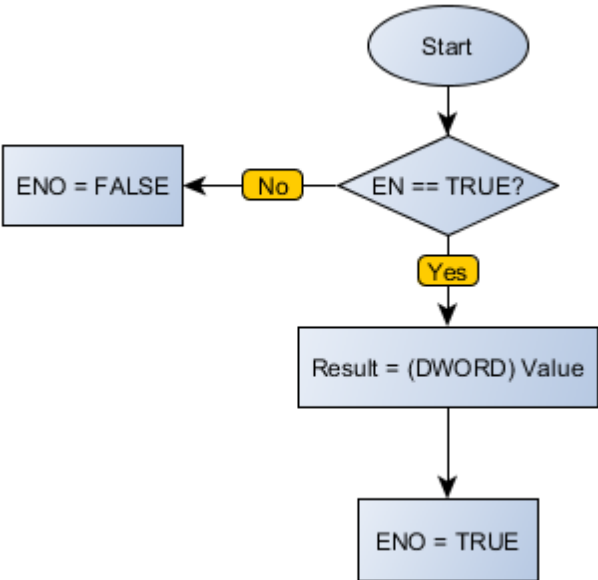
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para DWORD, armazenando em Result.

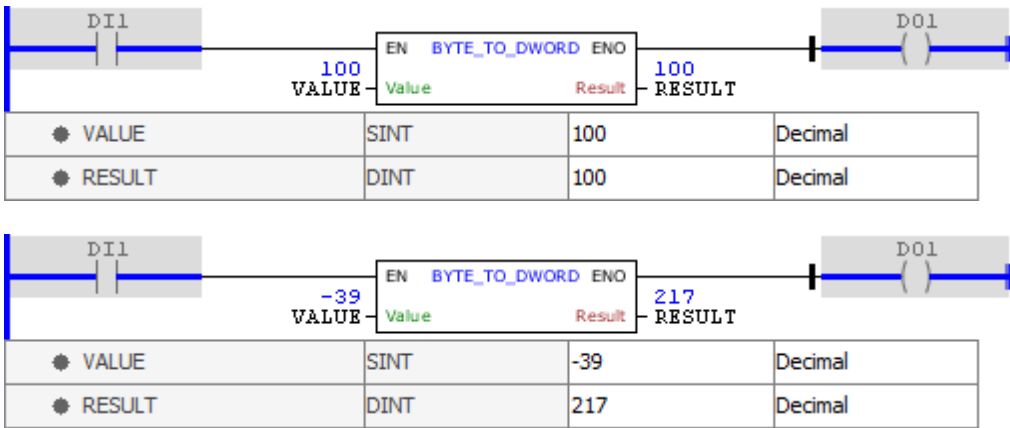
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



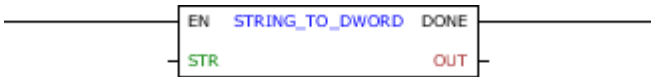
Exemplo



Os exemplos acima realizam a conversão da variável **VALUE**, em **BYTE**, para um valor **DWORD**, armazenando o resultado final em **RESULT**. O bloco finaliza com sucesso e a saída **ENO** é ativada.

Bloco que realiza a conversão de um valor **STRING** para um valor **DWORD**.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR	STRING	Valor em STRING
VAR_OUTPUT	DONE	BOOL	Fim de operação
	OUT	DWORD UDINT DINT	Valor em DWORD

Funcionamento

Este bloco, quando tem um valor TRUE em EN, lê o valor de STR como STRING caractere a caractere, realizando a conversão para REAL e armazenando em OUT. Caso o primeiro caractere não seja matematicamente válido, a saída OUT recebe zero. Caso haja caracteres válidos, esses caracteres serão convertidos até o fim da STRING, ou até encontrar um caractere não válido.



NOTA!

Caso o número representado na STRING seja superior ao máximo suportado por uma DWORD, o valor de Result satura neste valor máximo.

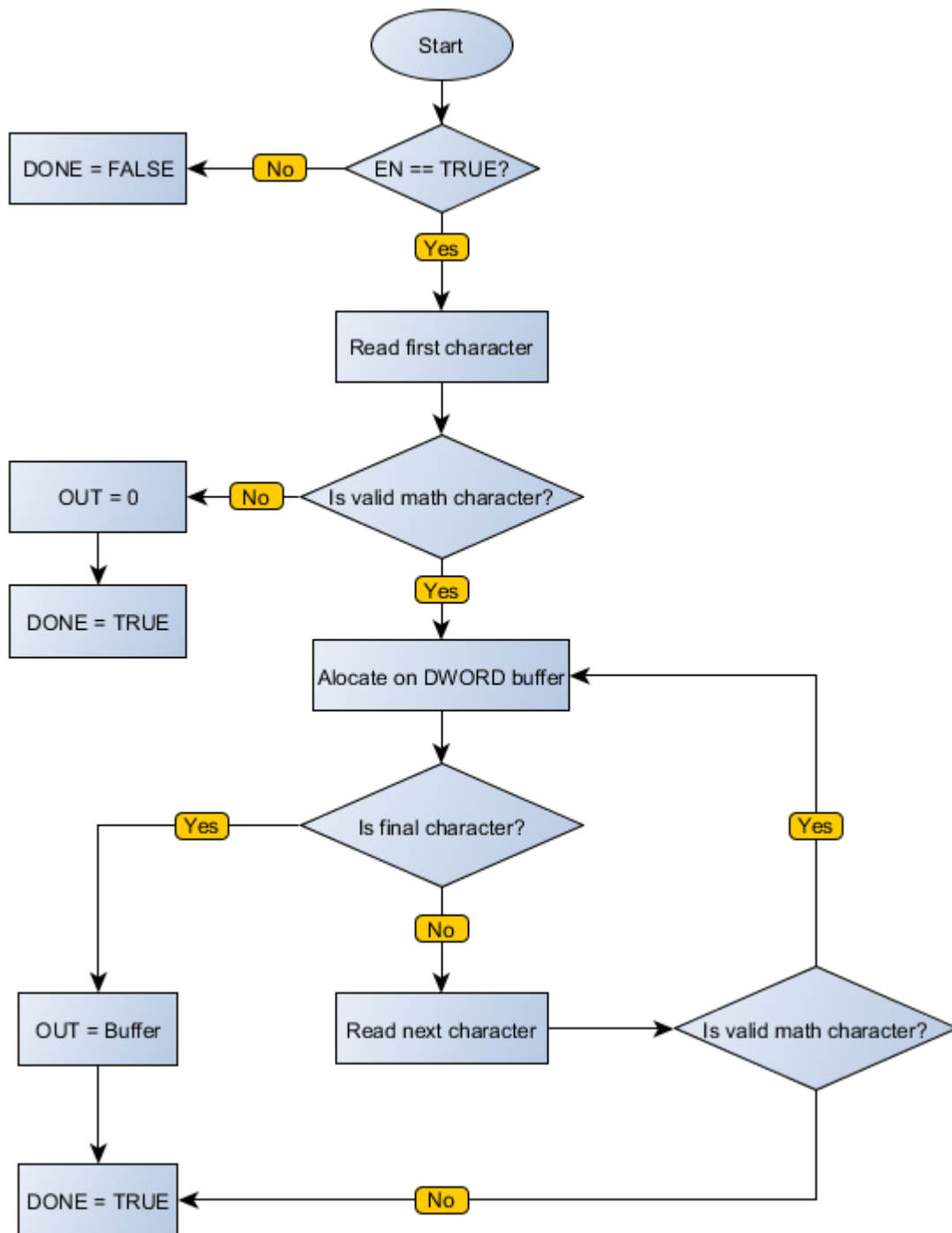
Quando EN possui valor FALSE, OUT permanece inalterado e DONE permanece FALSE.

O valor de DONE repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Compatibilidade

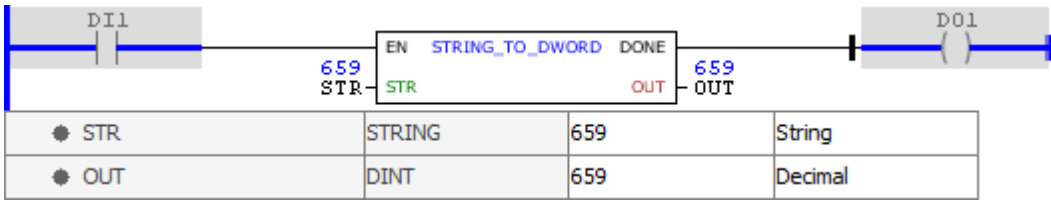
Equipamento	Versão
PLC300	2.10 ou superior

Fluxograma do Bloco

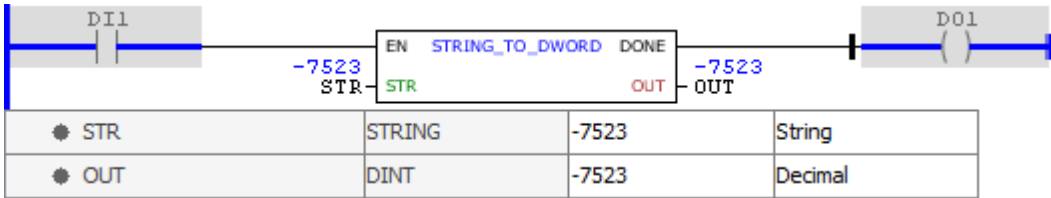


Exemplo

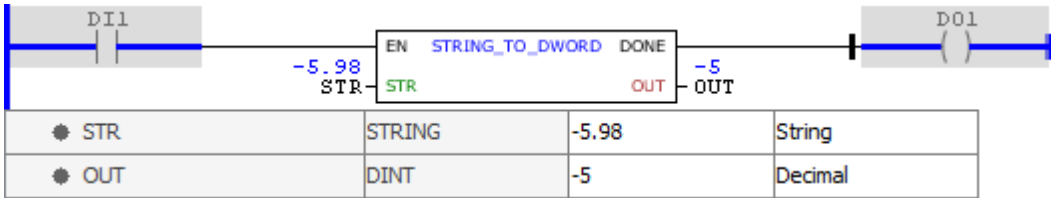
Os exemplos a seguir exibem diversas conversões de STRING em valores do tipo DWORD. Todas as conversões habilitam a saída DONE ao fim da operação.



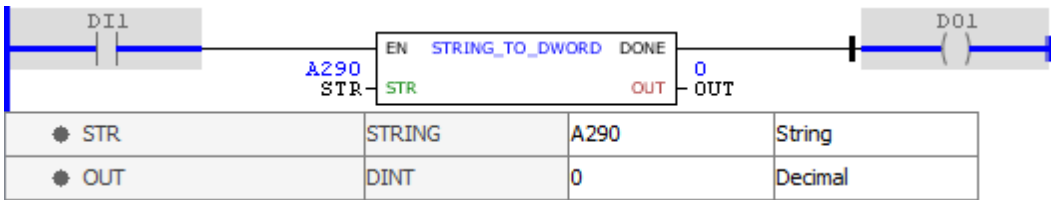
A conversão acima foi realizada com sucesso.



A conversão acima foi realizada com sucesso. O traço é um caractere matemático válido.



A conversão acima foi realizada com sucesso. O ponto decimal não é matematicamente válido para decimais e encerra a conversão, truncando o resultado ao que já havia sido convertido.

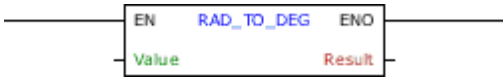


A conversão acima não foi realizada com sucesso. O primeiro caractere não foi identificado como matematicamente válido, e a saída foi zerada.

10.1.5.1.13.7 Rad-Deg

Bloco que realiza a conversão de um valor em radianos para um valor em graus.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em radianos
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em graus

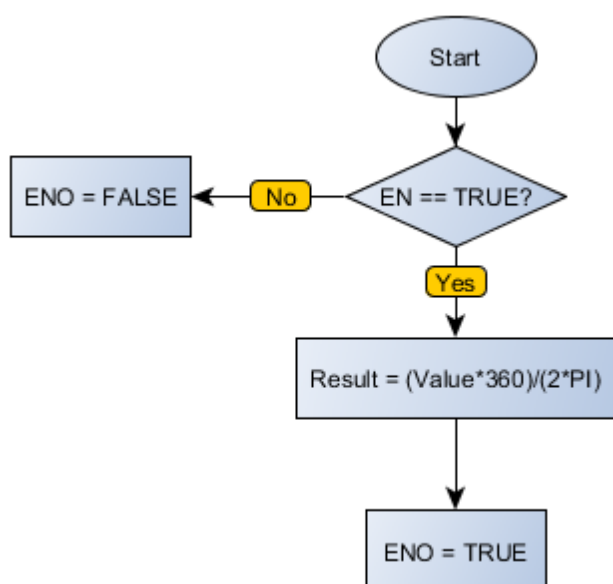
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como em radianos e realiza a conversão para graus, armazenando em Result.

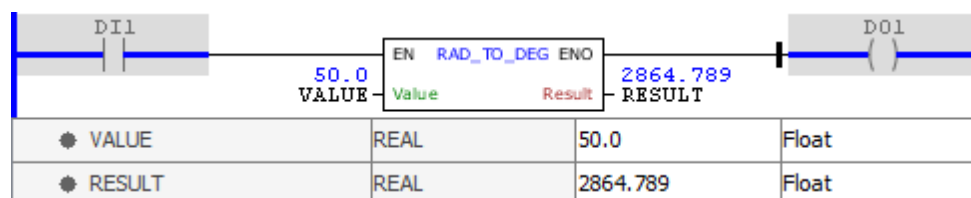
Quando EN possui valor FALSE, Result permanece inalterado.

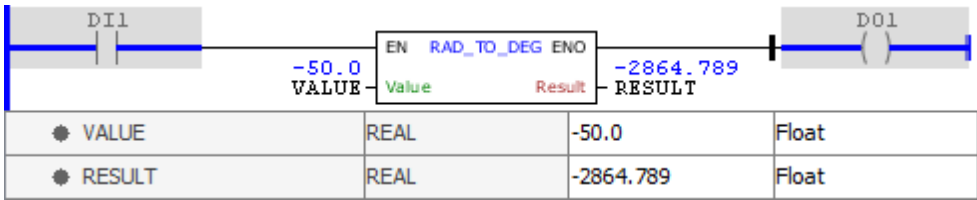
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

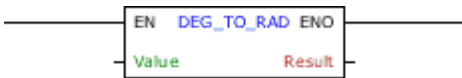




Os exemplos acima realizam a conversão da variável VALUE, em radianos, para um valor correspondente em graus, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor em graus para um valor em radianos.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em graus
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em radianos

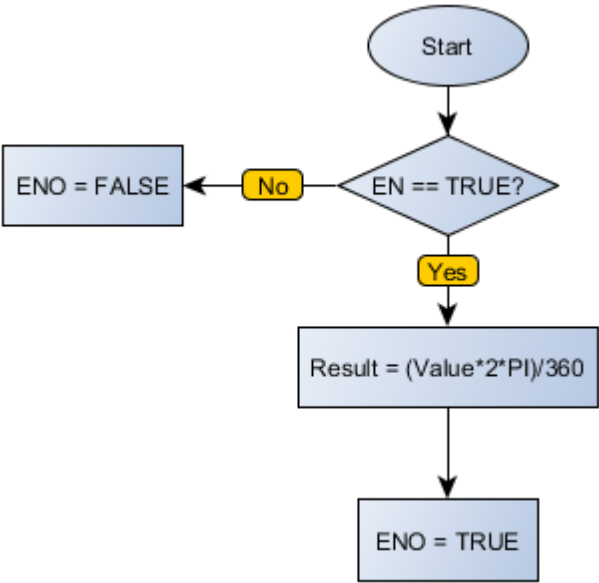
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como em graus e realiza a conversão para radianos, armazenando em Result.

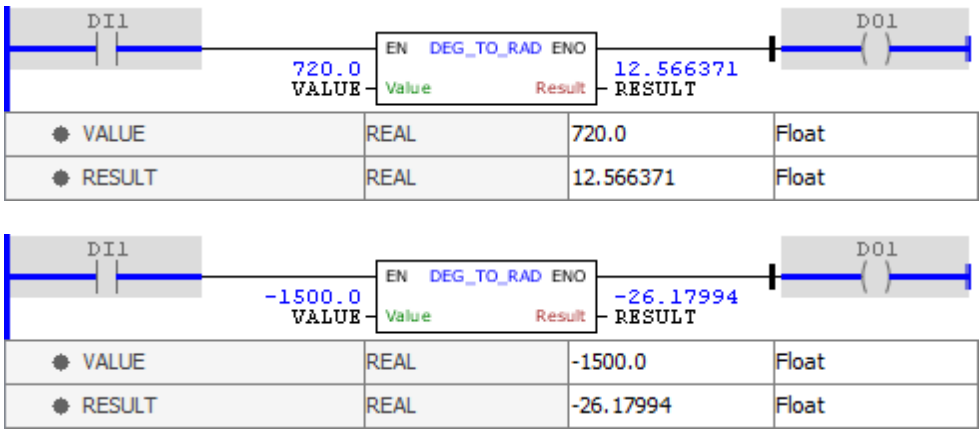
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

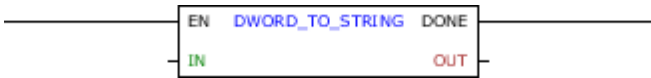


Os exemplos acima realizam a conversão da variável VALUE, em graus, para um valor correspondente em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.13.8 STRING

Bloco que realiza a conversão de um valor DWORD para um valor STRING.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	IN	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	DONE	BOOL	Fim de operação
	OUT	STRING	Valor em STRING

Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de IN como DWORD e realiza a conversão para STRING, armazenando em OUT e enviando TRUE à saída DONE.



NOTA!

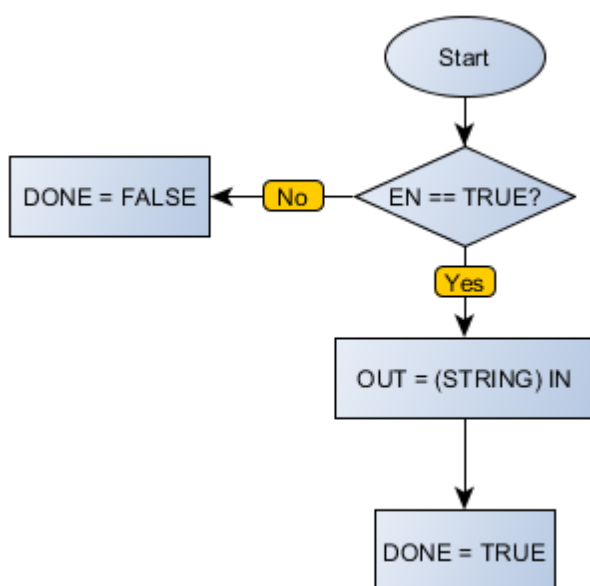
Caso o número representado possua mais dígitos que a capacidade da STRING, o valor será truncado.

Quando EN possui valor FALSE, OUT permanece inalterado e DONE permanece FALSE.

Compatibilidade

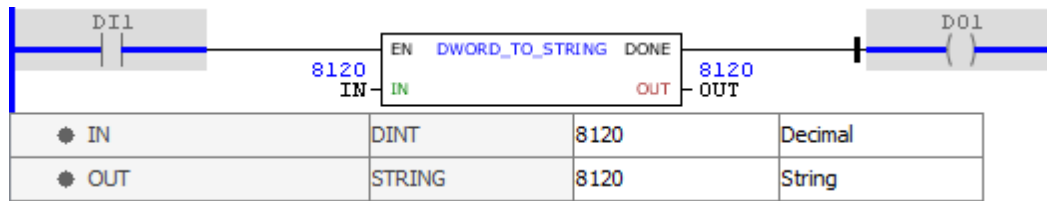
Equipamento	Versão
PLC300	2.10 ou superior

Fluxograma do Bloco

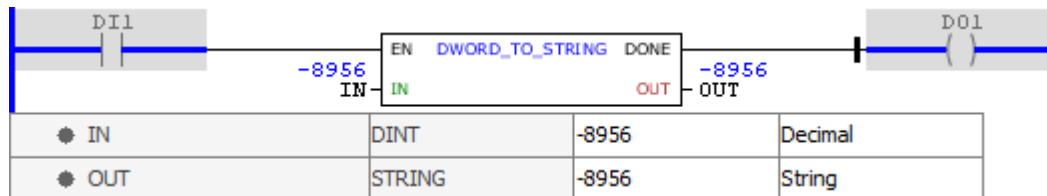


Exemplo

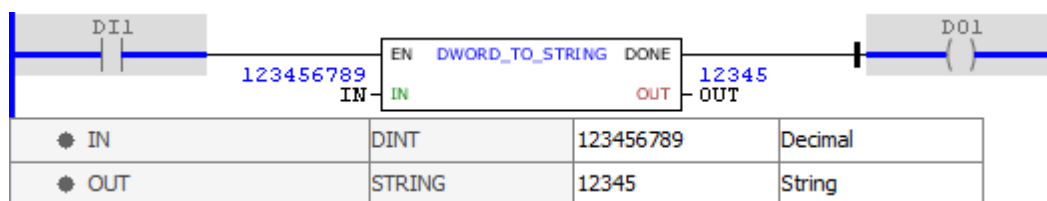
Os exemplos a seguir exibem diversas conversões de valores do tipo DWORD em STRING. Todas as conversões habilitam a saída DONE ao fim da operação.



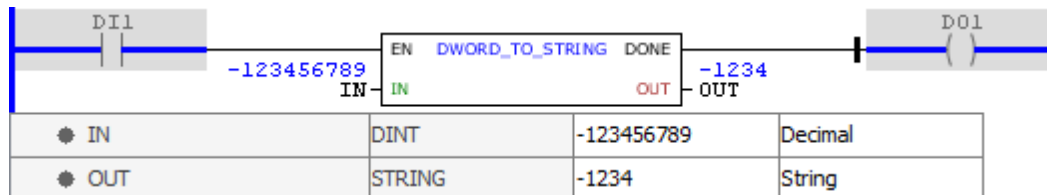
A conversão acima foi realizada com sucesso.



A conversão acima foi realizada com sucesso.



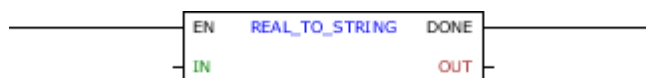
A conversão acima foi realizada com sucesso, porém o tamanho do resultado da conversão é superior ao tamanho de OUT, e este foi truncado.



A conversão acima foi realizada com sucesso, porém o tamanho do resultado da conversão é superior ao tamanho de OUT, e este foi truncado.

Bloco que realiza a conversão de um valor REAL para um valor STRING.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	IN	REAL	Valor em REAL
VAR_OUTPUT	DONE	BOOL	Fim de operação
	OUT	STRING	Valor em STRING

Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de IN como REAL e realiza a conversão para STRING, armazenando em OUT e enviando TRUE à saída DONE.



NOTA!

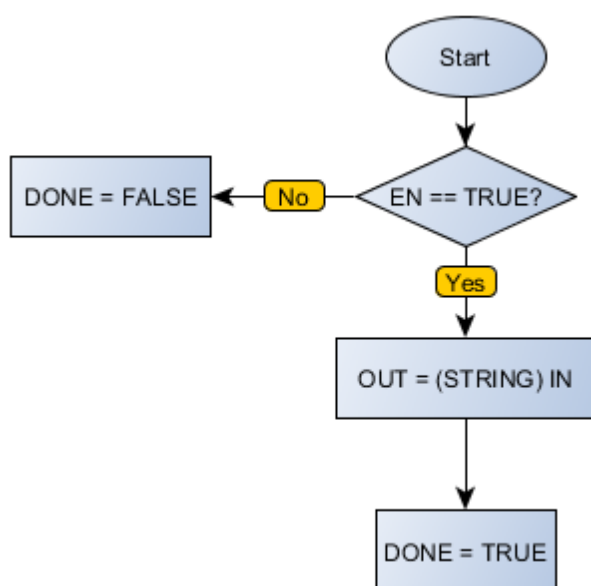
Caso o número representado possua mais dígitos que a capacidade da STRING, o valor será truncado.

Quando EN possui valor FALSE, OUT permanece inalterado e DONE permanece FALSE.

Compatibilidade

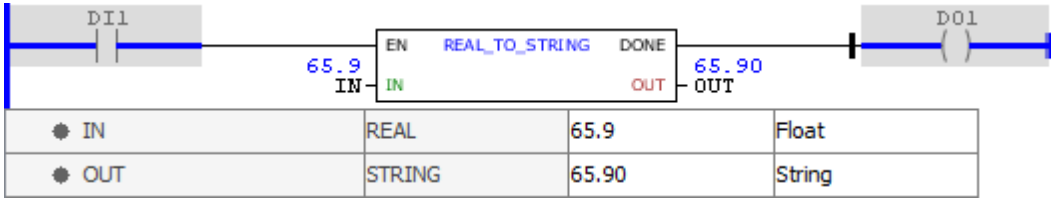
Equipamento	Versão
PLC300	2.10 ou superior

Fluxograma do Bloco

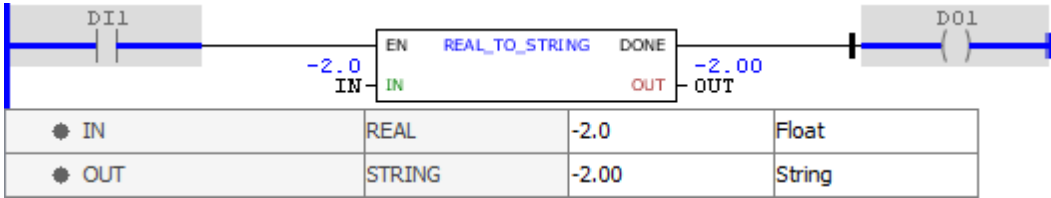


Exemplo

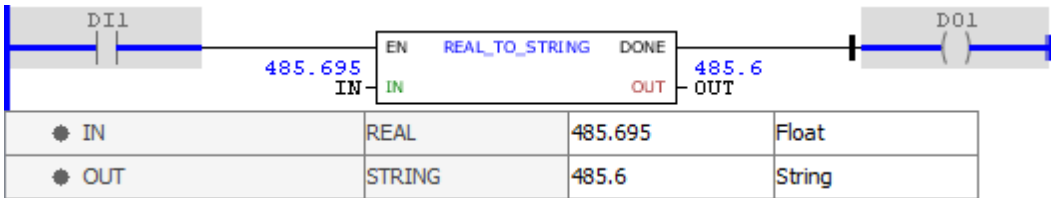
Os exemplos a seguir exibem diversas conversões de valores do tipo REAL em STRING. Todas as conversões habilitam a saída DONE ao fim da operação.



A conversão acima foi realizada com sucesso.



A conversão acima foi realizada com sucesso.



A conversão acima foi realizada com sucesso, porém o tamanho do resultado da conversão é superior ao tamanho de OUT, e este foi truncado.

10.1.5.1.14 Data Transfer

10.1.5.1.14.1 STORE

Bloco que realiza o armazenamento direto de dados provenientes de uma fonte em um destino.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SRC	BYTE USINT SINT WORD UINT INT DWORD DINT REAL	Fonte de dados
VAR_OUTPUT	ENO	BOOL	Fim de operação
	DST	BYTE USINT SINT WORD UINT INT DWORD DINT REAL	Destino de dados

Funcionamento

Este bloco, quando tem um valor TRUE em EN, armazena o conteúdo de SRC em DST.

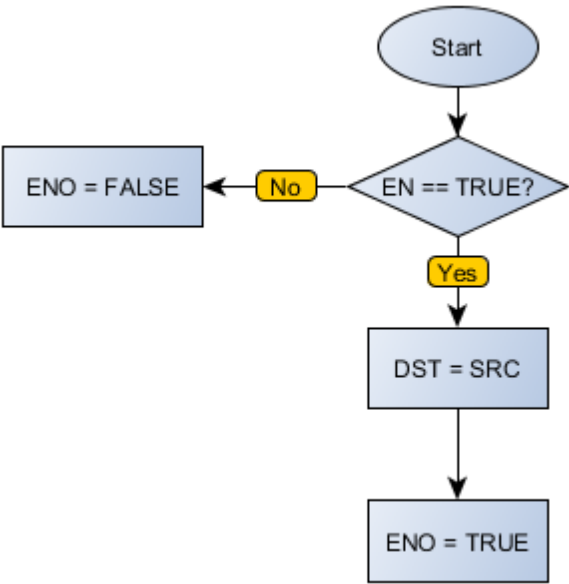


NOTA!
SRC e DST devem possuir tipos de dados de mesmo tamanho.

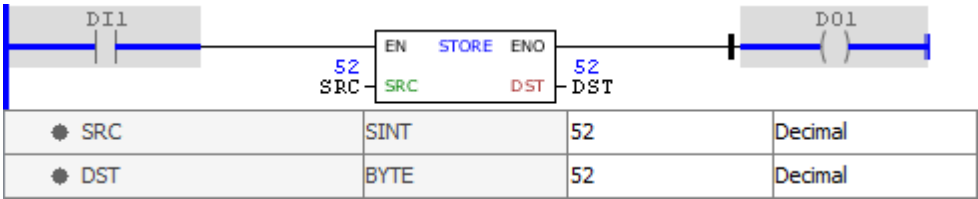
Quando EN possui valor FALSE, DST permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

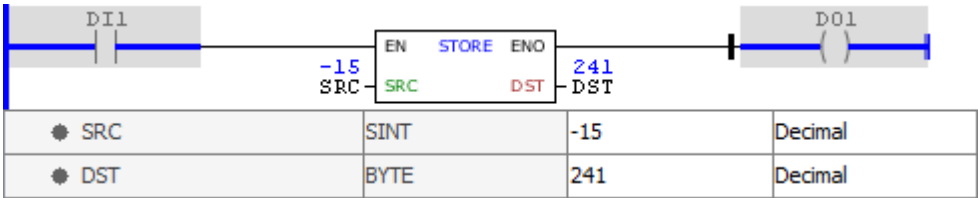
Fluxograma do Bloco



Exemplo



O exemplo acima armazena o valor da variável SRC em DST. O bloco finaliza com sucesso e a saída ENO é ativada.



O exemplo acima armazena o valor da variável SRC em DST. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário é mantido entre variáveis de tipos diferentes.

10.1.5.1.14.2 ILOADBOOL

Bloco que carrega indiretamente o valor de um bit em um endereço de variável global.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Bit	BYTE USINT SINT	Posição do bit a ser verificado
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Value	BOOL	Valor do bit selecionado pelos argumentos de entrada

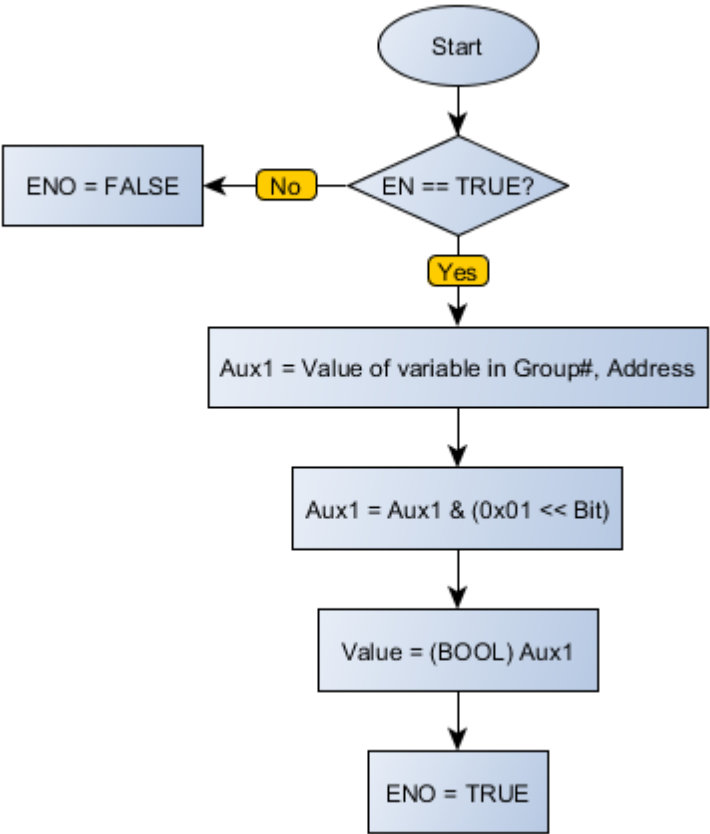
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega, em Value, o conteúdo do Bit da variável Address pertencente ao grupo Group#.

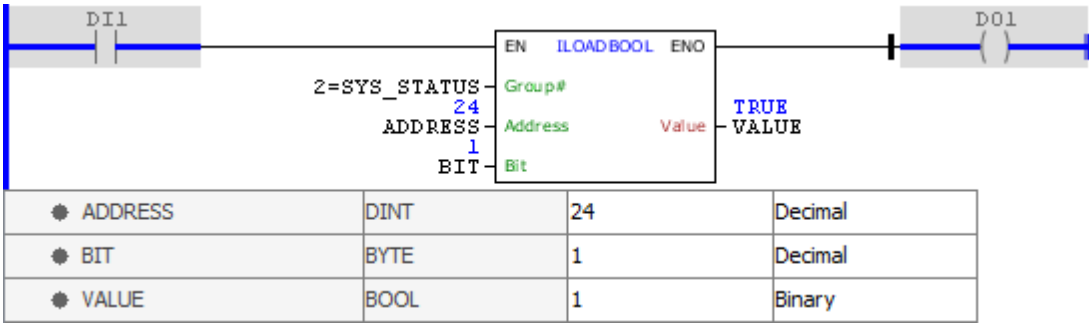
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima carrega o valor do bit 1 do endereço 24 do grupo 2 (GLOBAL_SYSTEM %S) para a variável VALUE. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.14.3 ILOAD

Bloco que carrega indiretamente o valor de uma variável e transfere para Value.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	DataType#	BYTE	Tipo de dado da variável selecionada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Value	Conforme selecionado em DataType#	Valor da variável selecionada

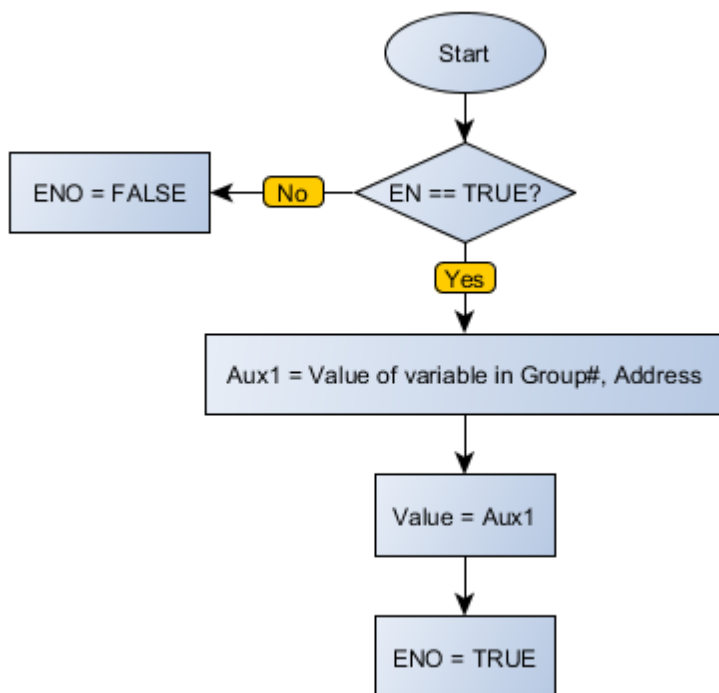
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega, em Value, o conteúdo da variável Address pertencente ao grupo Group#, segundo o DataType# escolhido.

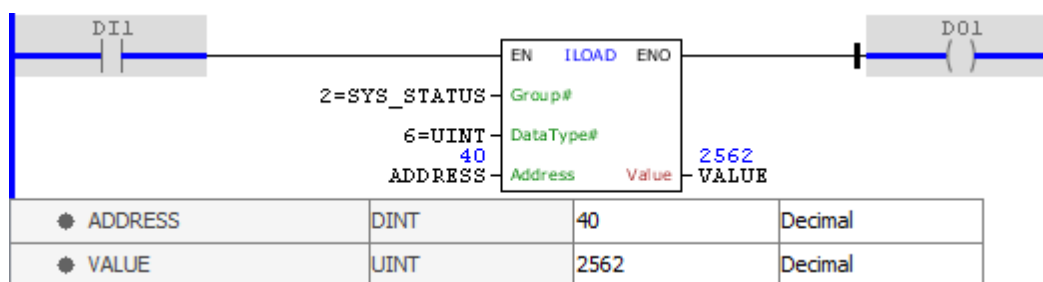
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima carrega o valor do endereço 40 do grupo 2 (GLOBAL_SYSTEM %S) em formato UINT para a variável VALUE. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.14.4 ISTOREBOOL

Bloco que armazena indiretamente o valor de Value em um bit em um endereço de variável global.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Bit	BYTE USINT SINT	Posição do bit a ser modificado
	Value	BOOL	Novo valor do bit selecionado
VAR_OUTPUT	ENO	BOOL	Fim de operação

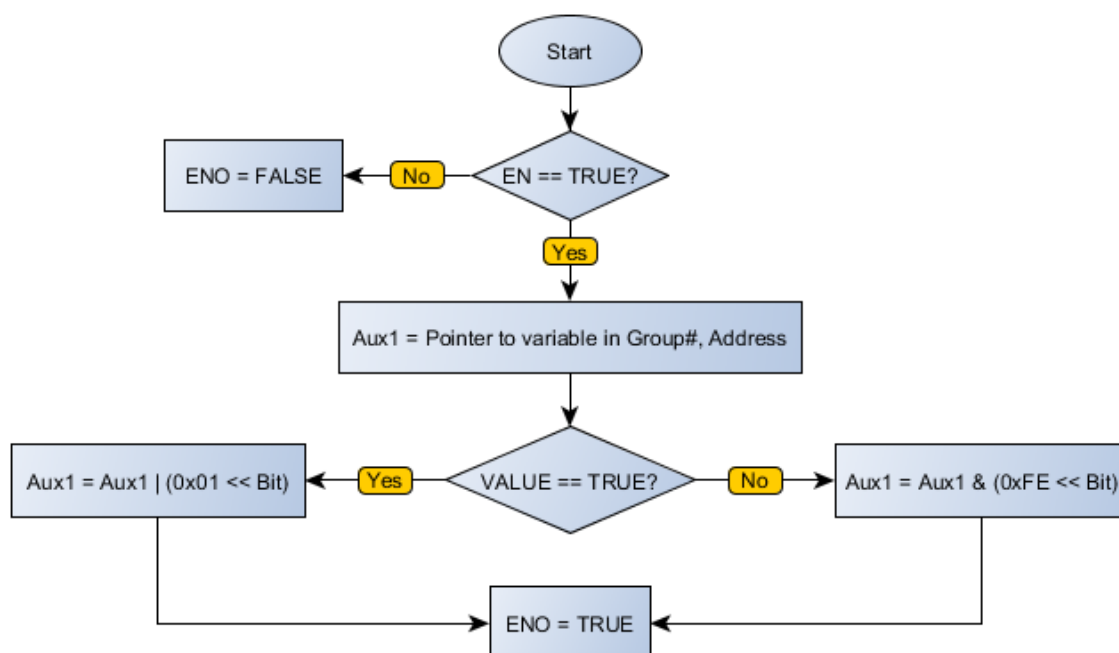
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega o valor de Value no conteúdo do Bit da variável Address pertencente ao grupo Group#.

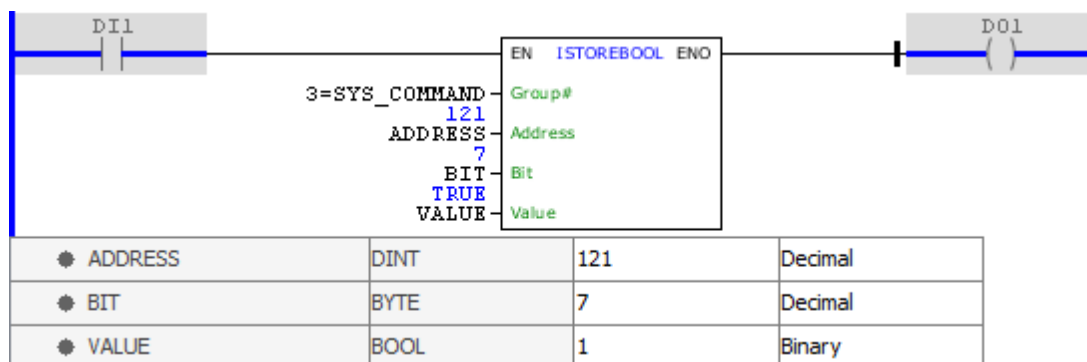
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

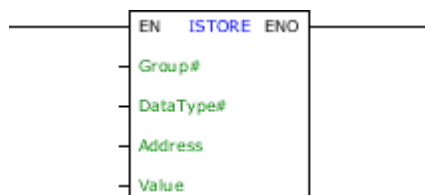


O exemplo acima armazena o valor de VALUE no bit 7 do endereço 121 do grupo 3 (GLOBAL_SYSTEM %C). O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.14.5 ISTORE

Bloco que armazena indiretamente o valor de Value em uma variável.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	DataType#	BYTE	Tipo de dado da variável selecionada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Value	Conforme selecionado em DataType#	Valor a ser escrito na variável selecionada
VAR_OUTPUT	ENO	BOOL	Fim de operação

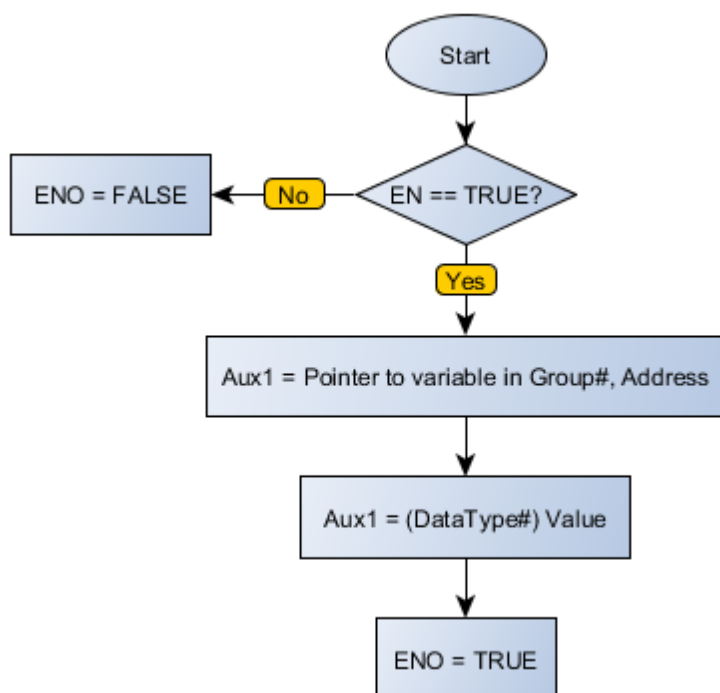
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega o valor de Value no conteúdo da variável Address pertencente ao grupo Group#, segundo o DataType# escolhido.

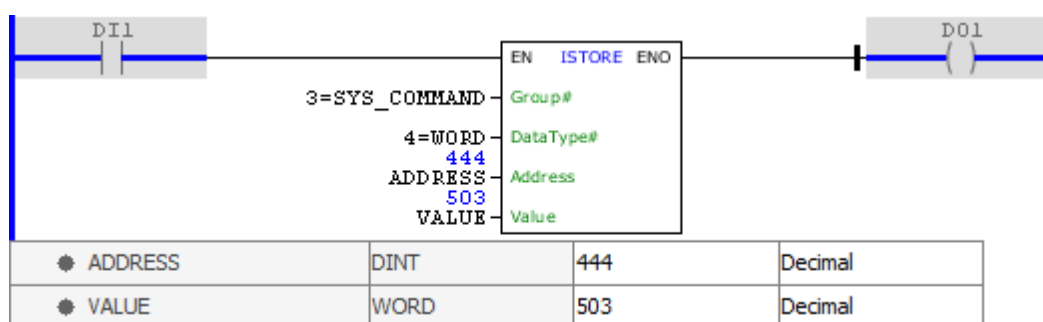
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

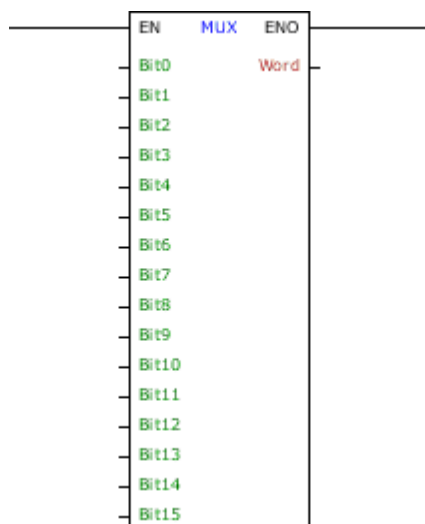


O exemplo acima armazena o valor de VALUE em formato WORD no endereço 444 do grupo 3 (GLOBAL_SYSTEM %C). O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.14.6 MUX

Bloco que cria uma nova variável WORD a partir da concatenação de 15 variáveis BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Bit0 - Bit15	BOOL	Bit da posição correspondente na nova palavra
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Word	WORD UINT INT	Nova palavra formada a partir dos bits de entrada

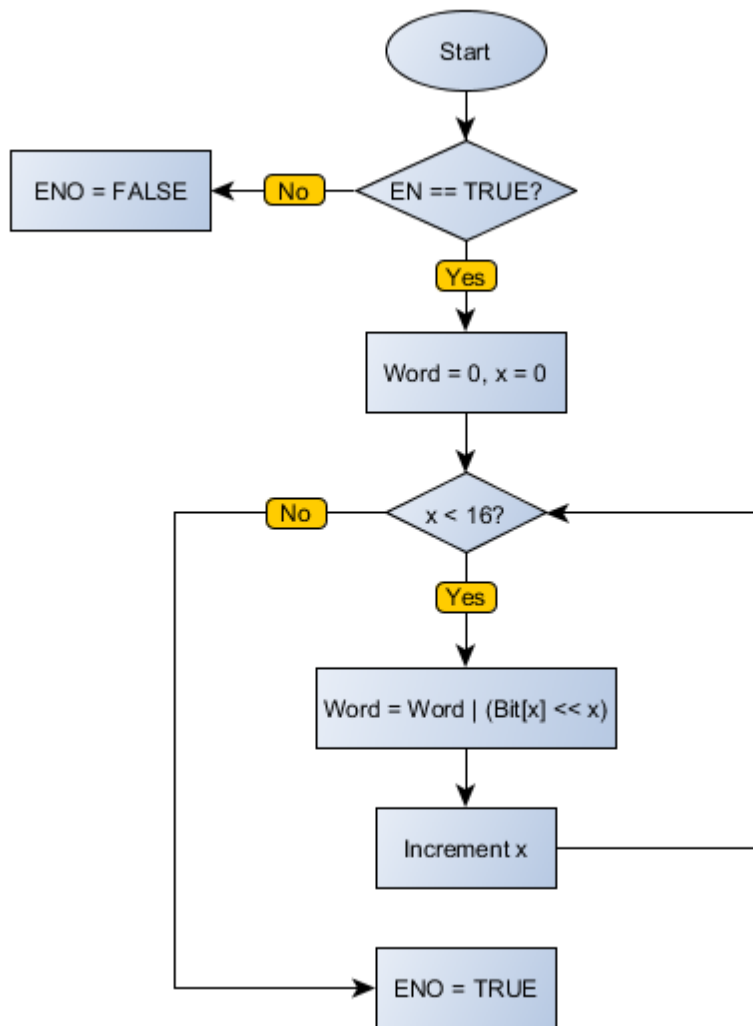
Funcionamento

Este bloco, quando tem um valor TRUE em EN, concatena os valores booleanos das variáveis de entrada e armazena este valor na variável Word. O Bit0 corresponde ao LSB (bit menos significativo) e Bit15 corresponde ao MSB (bit mais significativo).

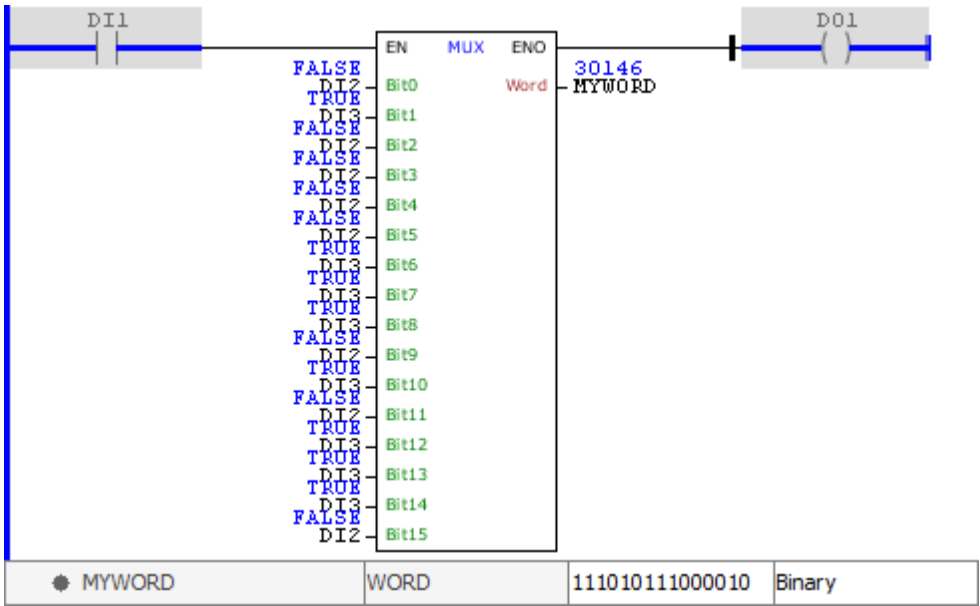
Quando EN possui valor FALSE, Word permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

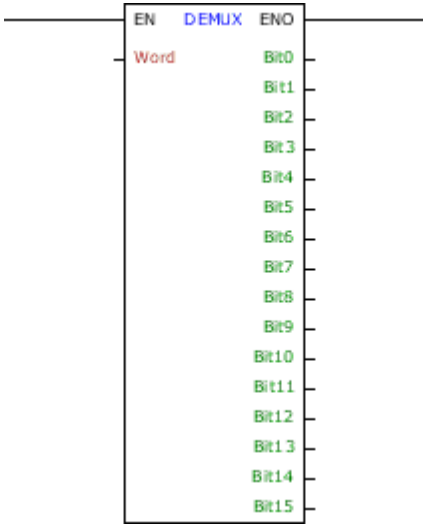


O exemplo acima concatena os valores booleanos dos bits de entrada do bloco, formando a palavra de saída armazenada em MYWORD. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.14.7 DEMUX

Bloco que cria 15 novas variáveis BOOL a partir da decomposição de uma variável WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Word	WORD UINT INT	Variável de entrada de 15 bits
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Bit0 - Bit15	BOOL	Bit da posição correspondente de Word

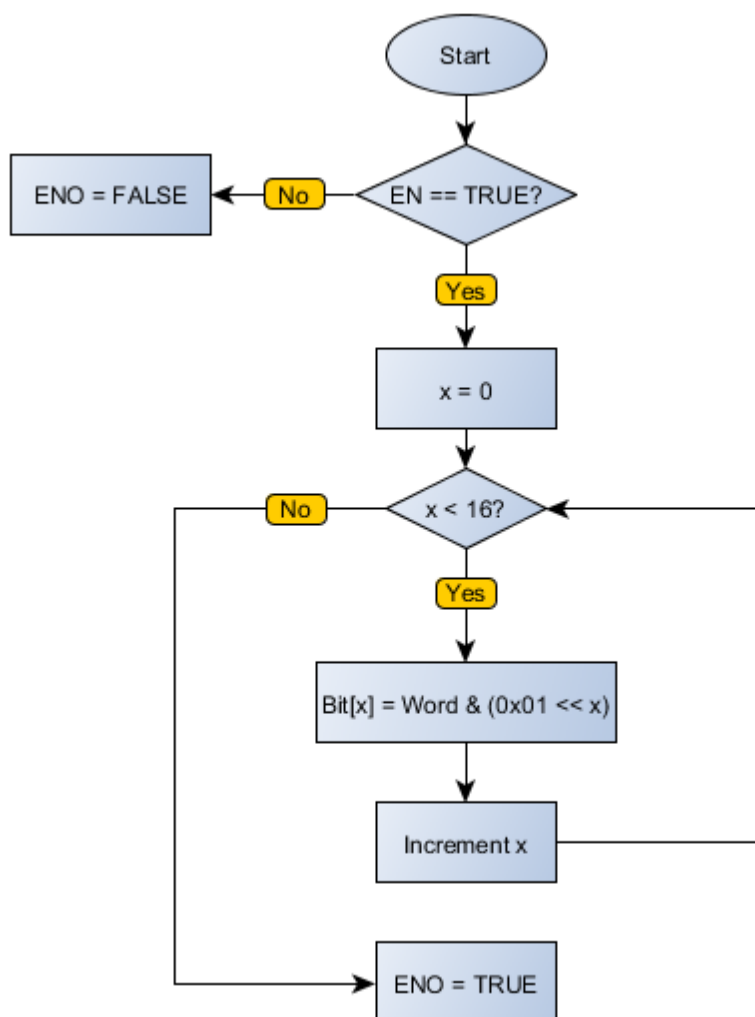
Funcionamento

Este bloco, quando tem um valor TRUE em EN, decompõe a variável de entrada Word em 15 valores booleanos armazenados nas variáveis Bit0 a Bit15. O Bit0 corresponde ao LSB (bit menos significativo) e Bit15 corresponde ao MSB (bit mais significativo).

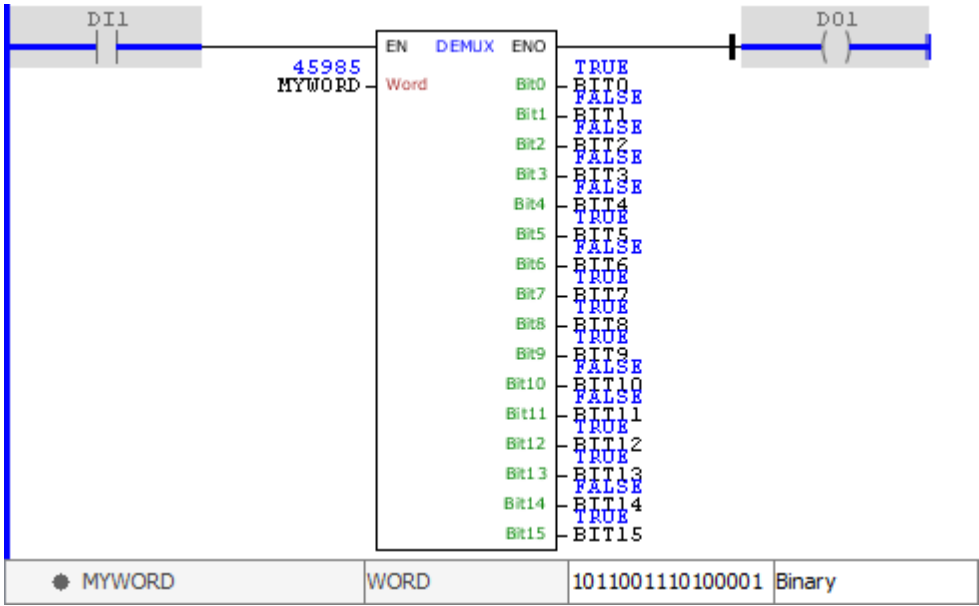
Quando EN possui valor FALSE, as variáveis de saída permanecem inalteradas.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima decompõe o valor de MYWORD em valores booleanos, sendo estes armazenados nas variáveis de saída BIT0 a BIT15. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.14.8 SEL

Bloco que replica para a saída o valor de uma das variáveis de entrada (Value0 ou Value1) de acordo com a seleção de Selector.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Selector	BOOL	Variável que seleciona a entrada
	Value0	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Entrada multiplexada número 1
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Entrada multiplexada número 2
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saída selecionado

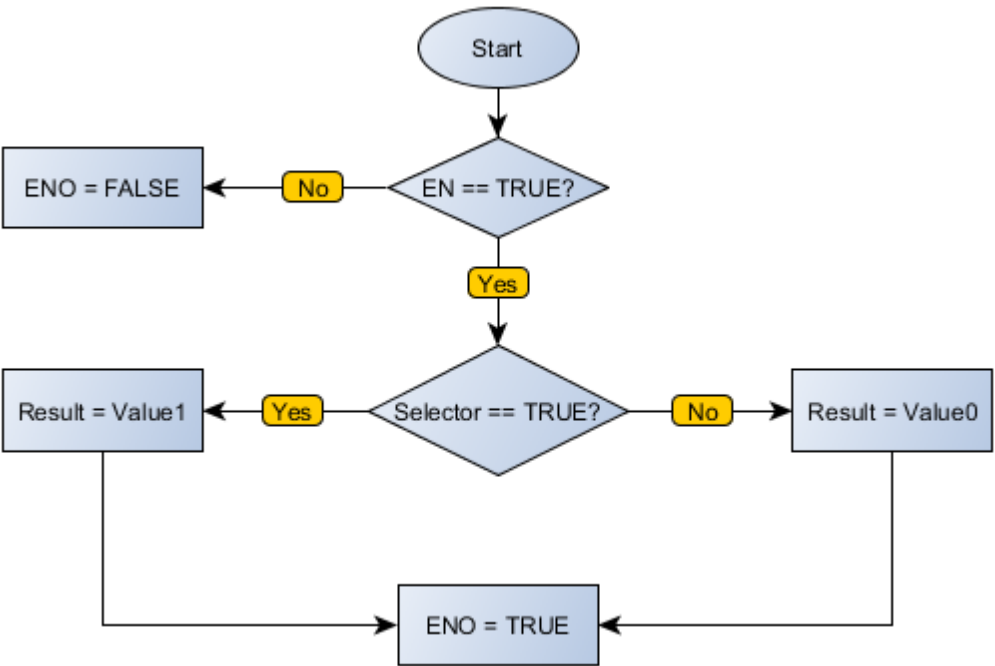
Funcionamento

Este bloco, quando tem um valor TRUE em EN, replica para a variável Result o valor de Value0 caso Selector seja FALSE ou o valor de Value1 caso Selector seja TRUE.

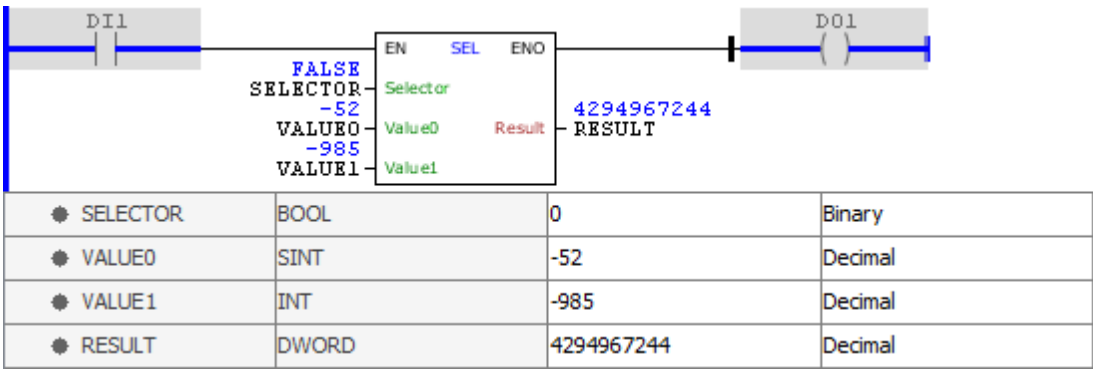
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco

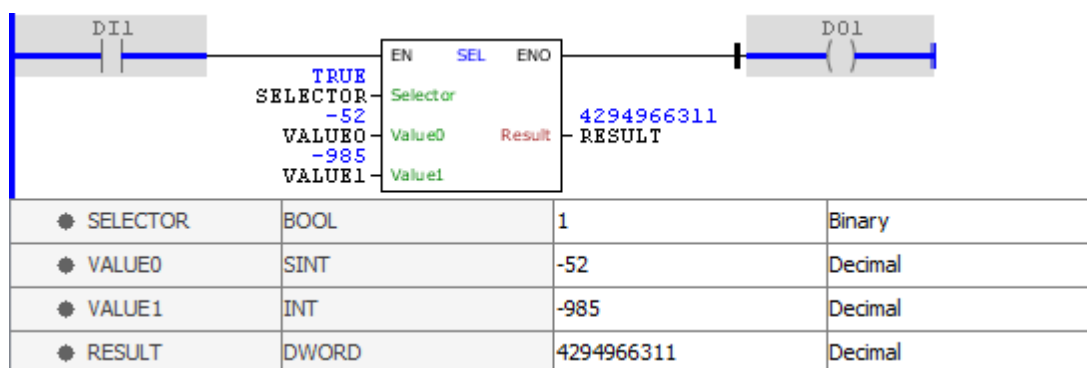


Exemplo



O exemplo acima utiliza a varável SELECTOR como seletor do canal de multiplexação. Quando ela está em nível FALSE, a saída RESULT recebe o valor de VALUE0. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário foi mantido, apesar de a representação decimal

estar corrompida, dado que DWORD não aceita valores negativos.



O exemplo acima utiliza a variável SELECTOR como seletor do canal de multiplexação. Quando ela está em nível TRUE, a saída RESULT recebe o valor de VALUE1. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário foi mantido, apesar de a representação decimal estar corrompida, dado que DWORD não aceita valores negativos.

10.1.5.1.14.9 SWAP

Bloco que realiza uma permuta entre os bytes ímpares e os bytes pares consecutivos em Value e envia o valor para Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT DWORD UDINT DINT	Variável de entrada a ser permutada
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saída

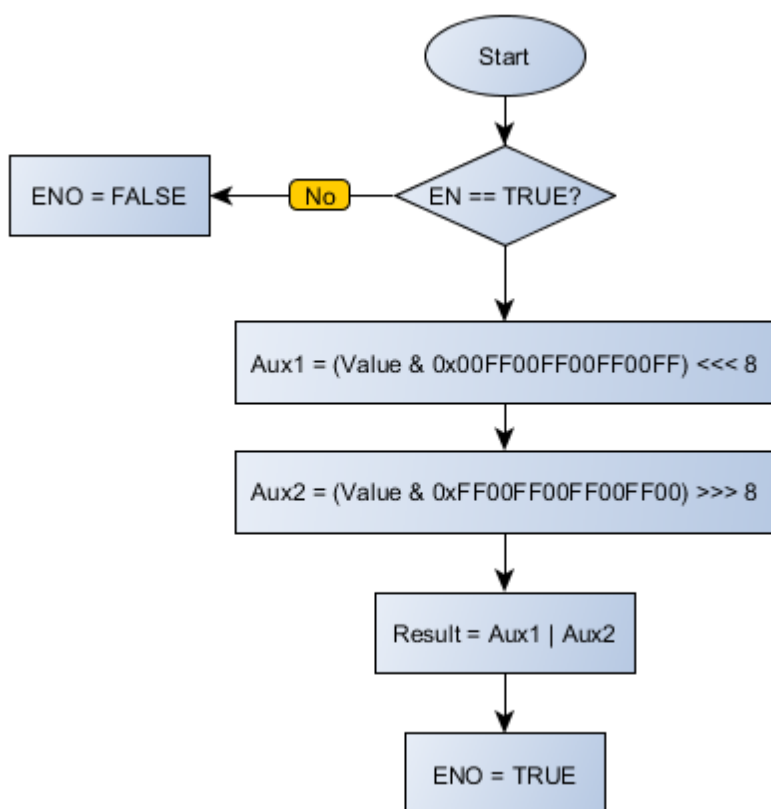
Funcionamento

Este bloco, quando tem um valor TRUE em EN, troca os valores entre os bytes ímpares (1, 3, 5 e 7) e os bytes pares consecutivos (2, 4, 6 e 8, respectivamente) da variável Value, armazenando o resultado em Result.

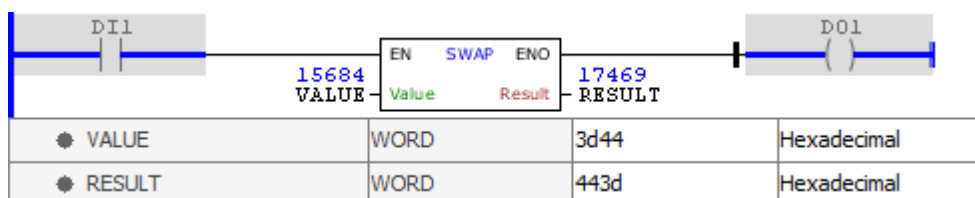
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

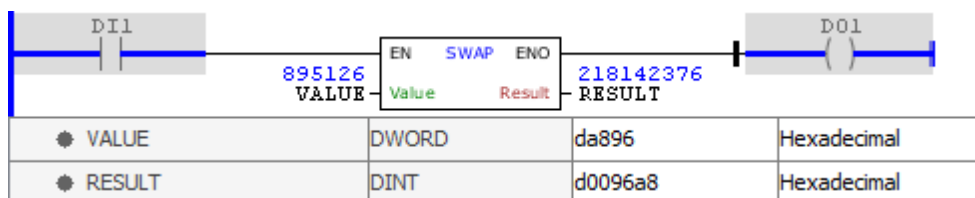
Fluxograma do Bloco



Exemplo



O exemplo troca a posição do valor do byte 1 de VALUE (0x44) com o byte 2 de VALUE (0x3D), armazenando o resultado final (0x44_3D) em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.



O exemplo troca a posição do valor do byte 1 de VALUE (0x96) com o byte 2 de VALUE (0xA8) e do byte 3 de VALUE (0x0D) com o byte 4 de VALUE (0x00), armazenando o resultado final (0x0D_00_96_A8) em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.1.5.1.14.10 ReadRecipe

Bloco que obtém a receita de um arquivo de receitas e o envia a uma variável.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	FILENAME#	STRING	Nome do arquivo de receitas
	INDEX	WORD UINT	Índice da receita a ser lida
VAR_OUTPUT	Q	BOOL	Fim de operação
	ERROR	BOOL	Sinalizador de ocorrência de erro
	ERRORID	BYTE USINT	Identificador do erro ocorrido
	DST	STRUCT	Variável onde serão salvos os dados lidos
VAR	READRECIPE_INST_0	READRECIPE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando identifica uma borda de subida em Execute, obtém os dados da receita selecionada pelo índice INDEX no arquivo FILENAME# e os envia à DST. Caso tudo ocorra com sucesso, Q recebe TRUE e permanece assim enquanto Execute for TRUE.

**NOTA!**

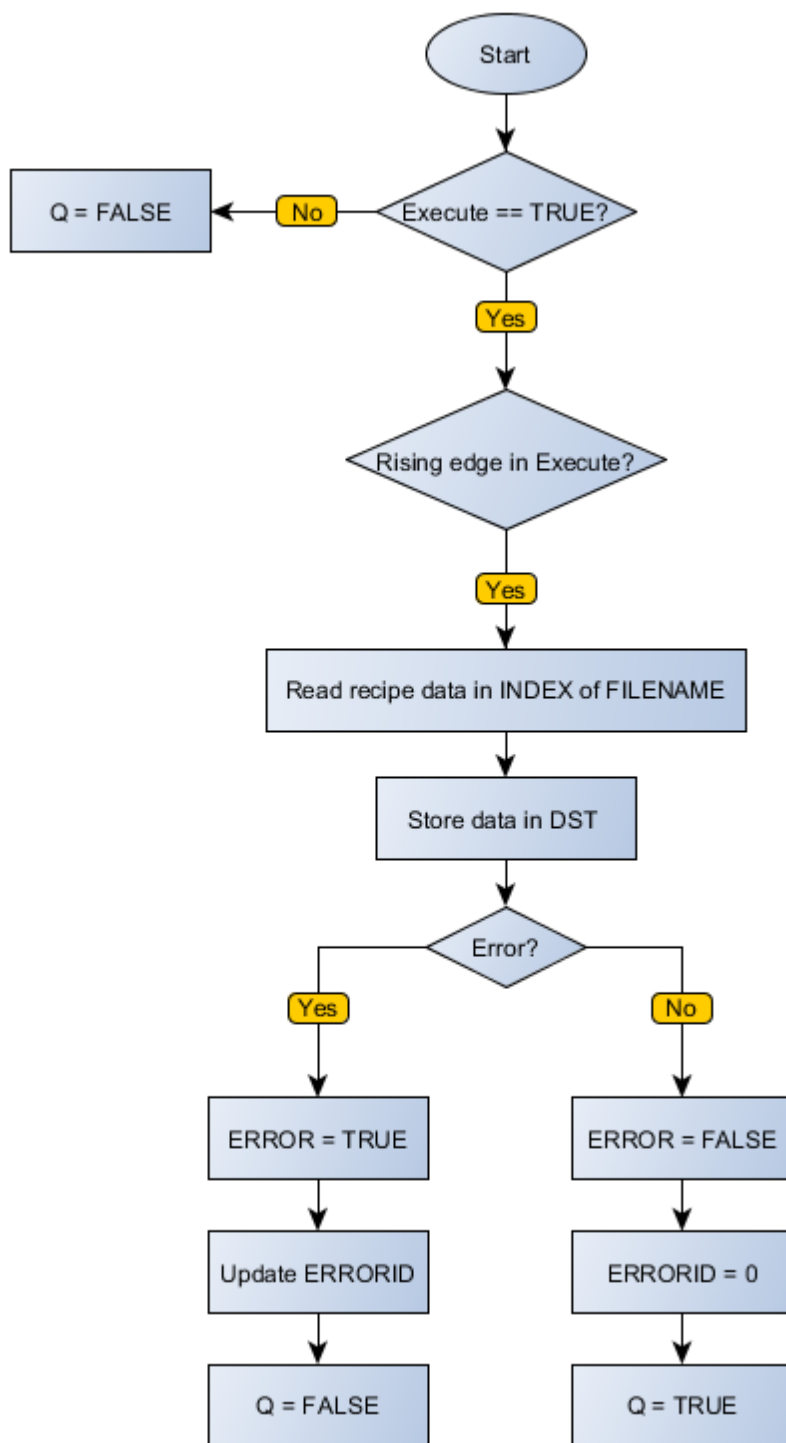
Receitas armazenadas na RAM são identificadas por 'NOME_DA_RECEITA'. Receitas armazenadas no cartão SD são identificadas por 'NOME_DA_RECEITA.CSV'.

Quando Execute possui valor FALSE, DST permanece inalterado.

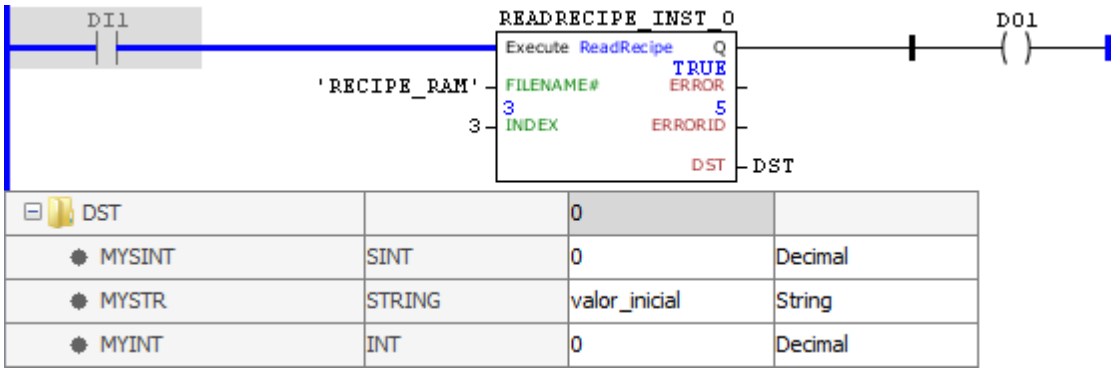
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
1	Receita incompleta
2	Estrutura inválida
3	Receita inexistente
4	Arquivo inválido
5	Arquivo inexistente ou SD card inválido
6	SD card bloqueado para escrita

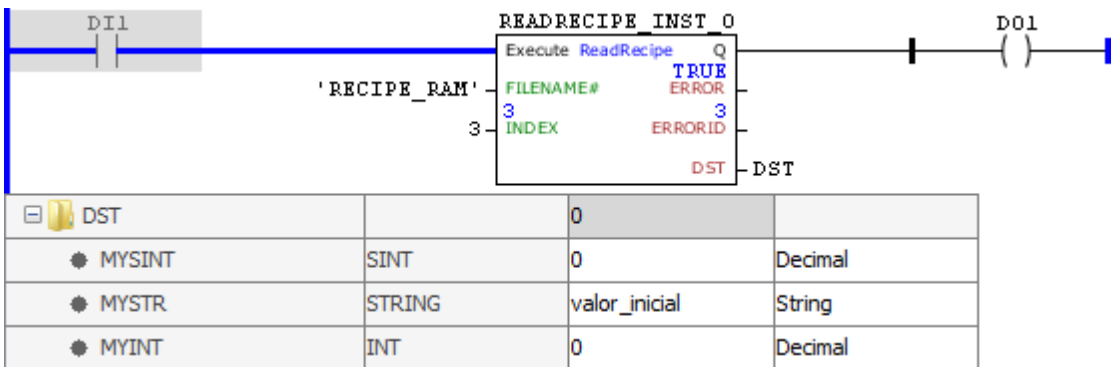
Fluxograma do Bloco



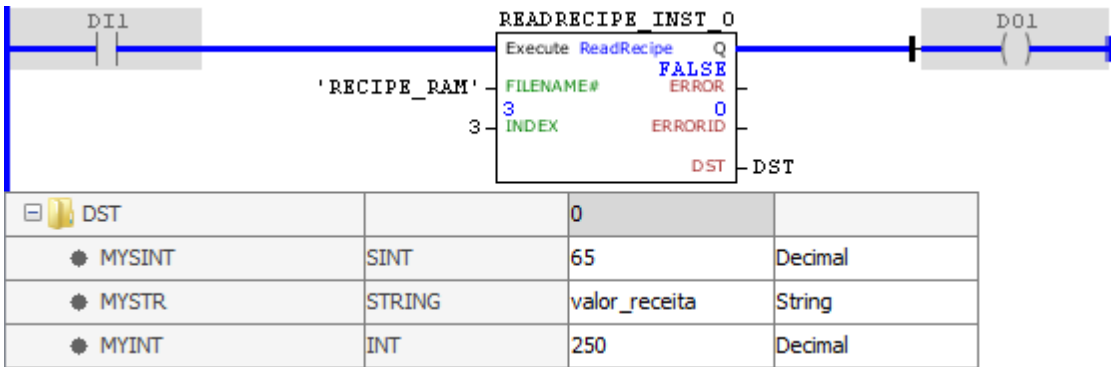
Exemplo



O exemplo acima busca o índice 3 do arquivo de receitas armazenado na RAM 'RECIPE_RAM'. O bloco não encontra o arquivo especificado, habilitando a saída ERROR com ERRORID de valor 5 e desativando a saída Q.



O exemplo acima busca o índice 3 do arquivo de receitas armazenado na RAM 'RECIPE_RAM'. O bloco encontra o arquivo especificado, porém não encontra o índice 3, habilitando a saída ERROR com ERRORID de valor 3 e desativando a saída Q.



O exemplo acima busca o índice 3 do arquivo de receitas armazenado na RAM 'RECIPE_RAM'. O bloco encontra o arquivo e o índice especificado, armazena os valores em DST, desabilita a saída ERROR e ativa a saída Q.

10.1.5.1.14.11 WriteRecipe

Bloco que escreve uma receita em um arquivo de receitas a partir de uma variável.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	FILENAME#	STRING	Nome do arquivo de receitas
	INDEX	WORD UINT	Índice da receita a ser escrita
	SRC	STRUCT	Variável de onde são lidos os dados
VAR_OUTPUT	Q	BOOL	Fim de operação
	ERROR	BOOL	Sinalizador de ocorrência de erro
	ERRORID	BYTE USINT	Identificador do erro ocorrido
VAR	WRITERECIPE_INST_0	WRITERECIPE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando identifica uma borda de subida em Execute, obtém os dados de SRC e os envia ao arquivo de receitas selecionado pelo índice INDEX no arquivo FILENAME#. Caso tudo ocorra com sucesso, Q recebe TRUE e permanece assim enquanto Execute for TRUE.



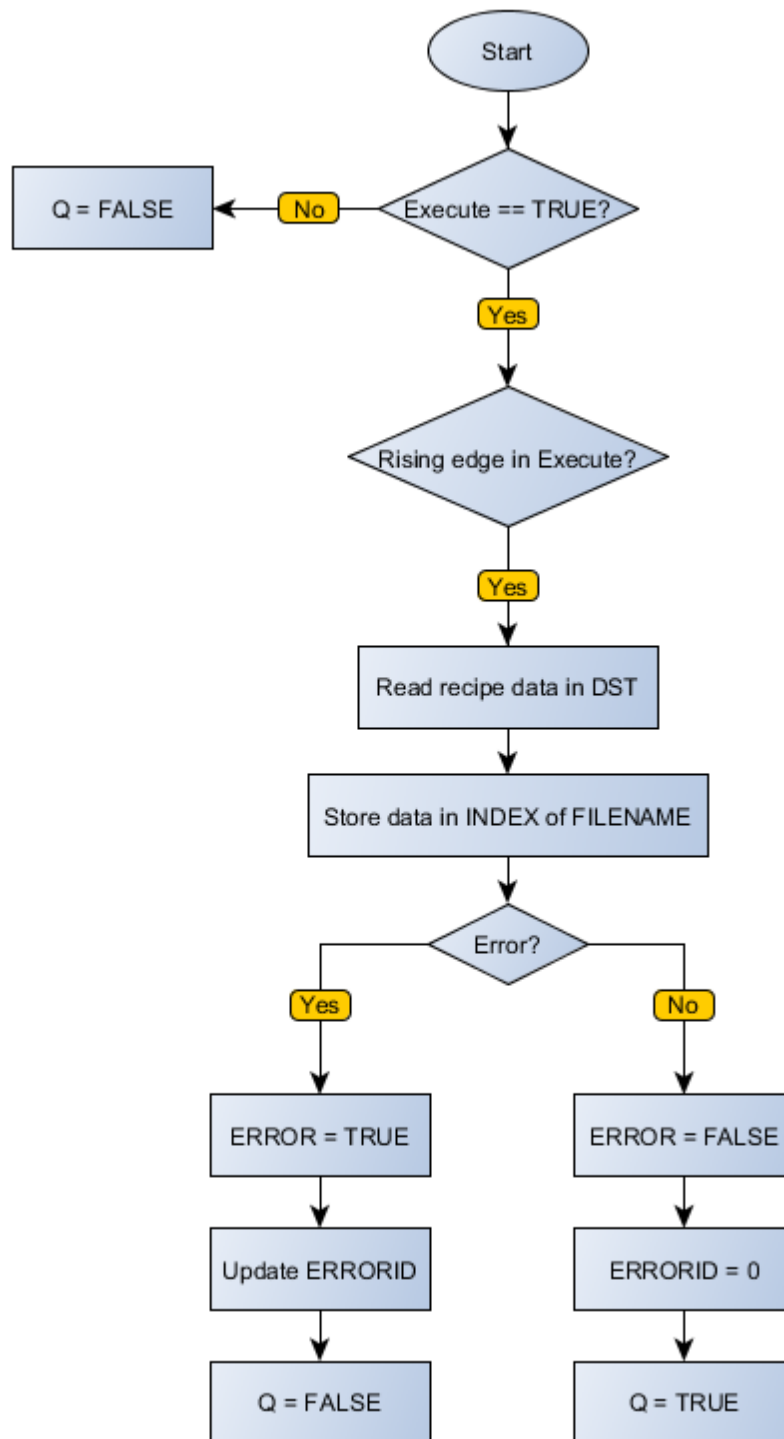
NOTA!

Receitas armazenadas na RAM são identificadas por 'NOME_DA_RECEITA'. Receitas armazenadas no cartão SD são identificadas por 'NOME_DA_RECEITA.CSV'.

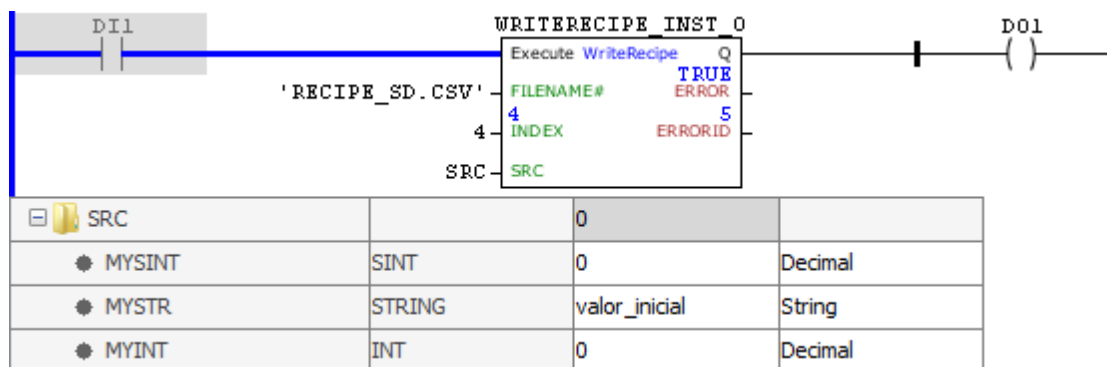
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
1	Receita incompleta
2	Estrutura inválida
3	Receita inexistente
4	Arquivo inválido
5	Arquivo inexistente ou SD card inválido
6	SD card bloqueado para escrita

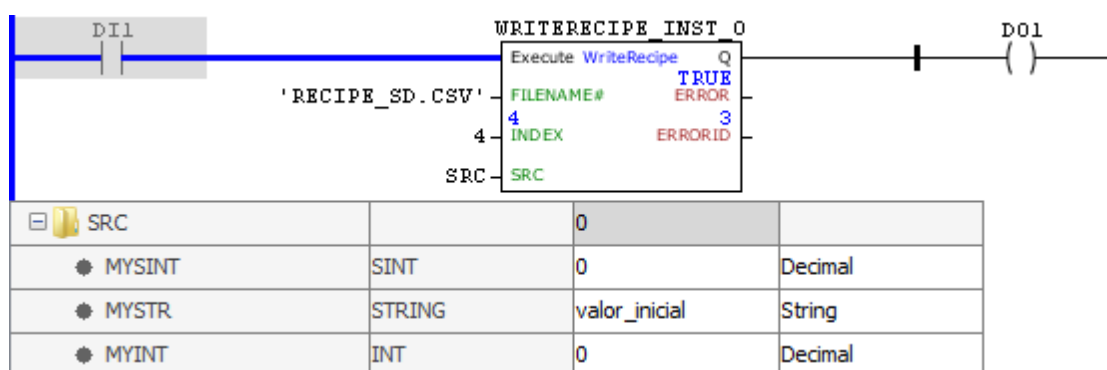
Fluxograma do Bloco



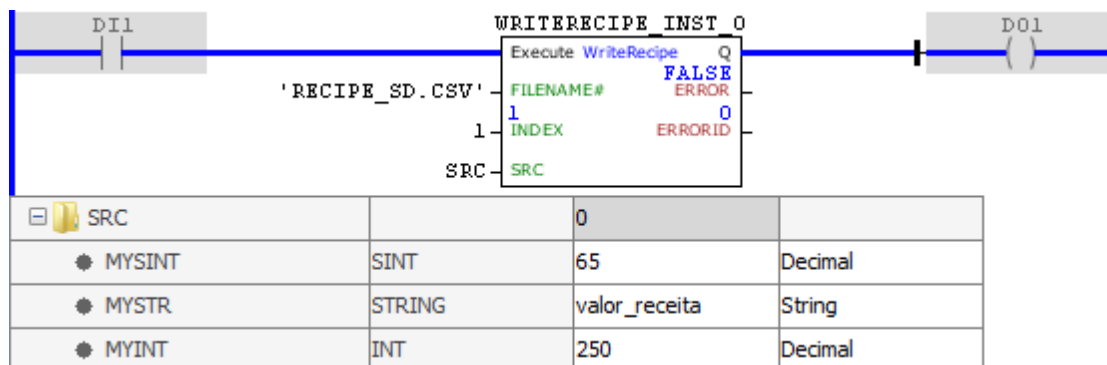
Exemplo



O exemplo acima tenta escrever o conteúdo de SRC no índice 4 do arquivo de receitas armazenado no cartão SD 'RECIPE_SD.CSV'. O bloco não encontra o arquivo especificado, habilitando a saída ERROR com ERRORID de valor 5 e desativando a saída Q.



O exemplo acima tenta escrever o conteúdo de SRC no índice 4 do arquivo de receitas armazenado no cartão SD 'RECIPE_SD.CSV'. O bloco encontra o arquivo especificado, porém não encontra o índice 4, habilitando a saída ERROR com ERRORID de valor 3 e desativando a saída Q.



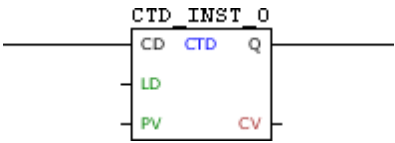
O exemplo acima tenta escrever o conteúdo de SRC no índice 4 do arquivo de receitas armazenado no cartão SD 'RECIPE_SD.CSV'. O bloco encontra o arquivo e o índice especificado, armazena os valores na receita, desabilita a saída ERROR e ativa a saída Q.

10.1.5.1.15 Counter

10.1.5.1.15.1 CTD

Bloco de contagem regressiva de pulsos de entrada.

Representação Ladder



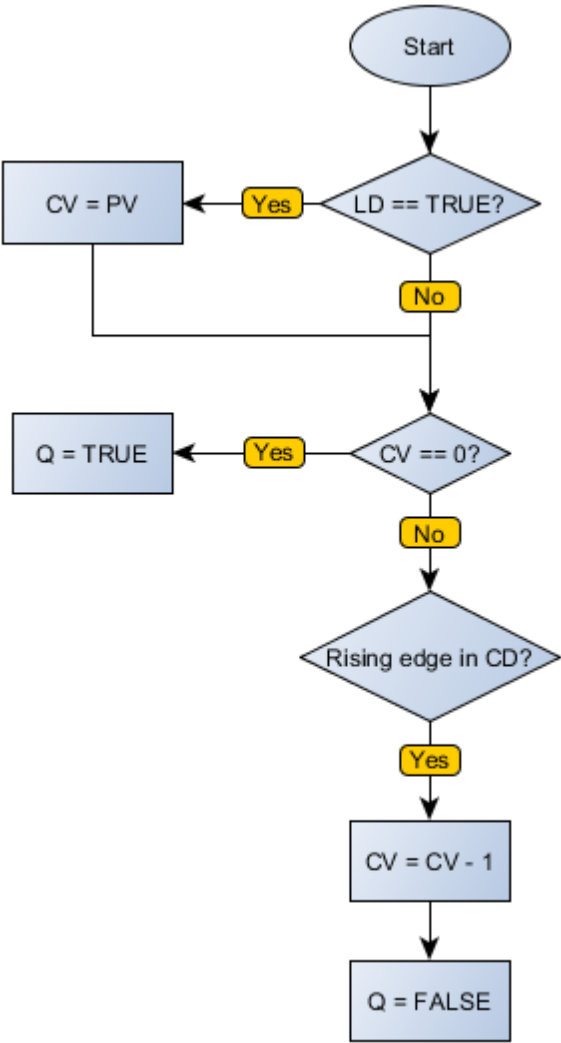
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	CD	BOOL	Identificador de pulsos
	LD	BOOL	Carrega o valor de PV em CV
	PV	WORD UINT	Valor de configuração inicial
VAR_OUTPUT	Q	BOOL	Sinalizador de contador zerado
	CV	WORD UINT	Valor de contagem atual
VAR	CTD_INST_0	CTD	Instância de acesso à estrutura do bloco

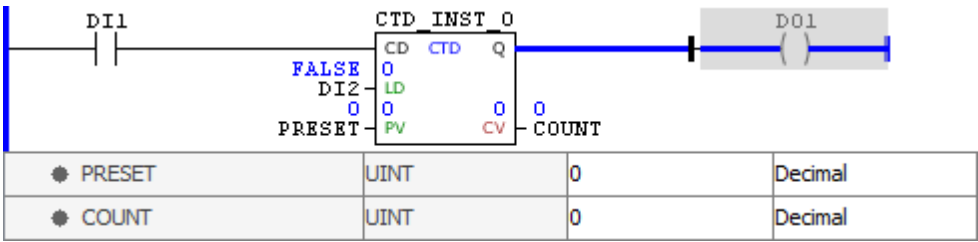
Funcionamento

Este bloco, quando identifica uma borda de subida em CD, decrementa a variável CV até que a mesma seja zero. Enquanto CV for igual a zero, a saída Q permanece em nível TRUE. Ao detectar nível alto em LD, o bloco carrega o valor de PV em CV.

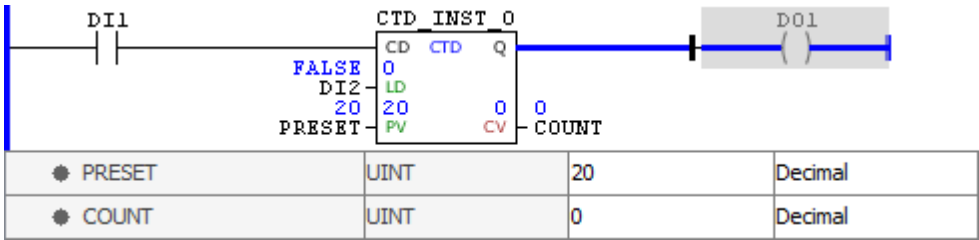
Fluxograma do Bloco



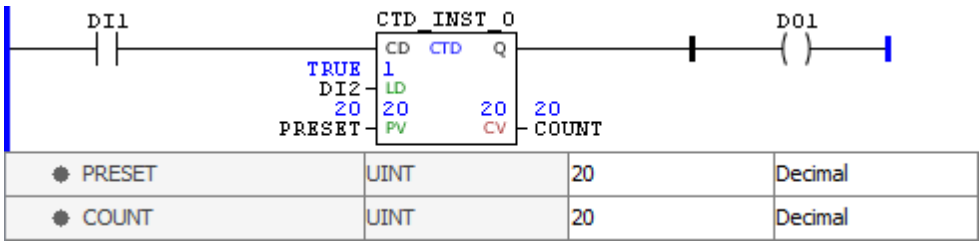
Exemplo



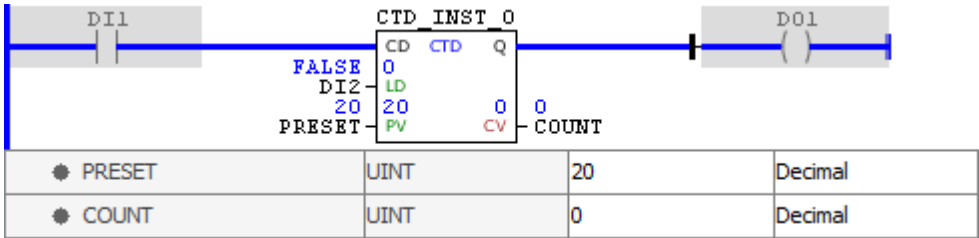
O exemplo acima exibe as condições iniciais da rotina. Como CV possui o valor zero, a saída Q é habilitada.



O valor da variável PV foi alterado para 20, porém ainda não foi carregado.



Ao identificar o nível TRUE em LD, o bloco carrega o valor de PV para CV. Como este valor é maior que zero, a saída Q é desabilitada.

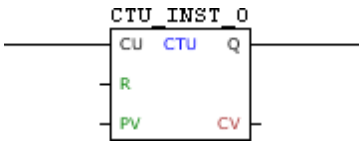


A cada borda de subida identificada em CD, o valor de COUNT é decrementado até chegar a zero, quando a saída Q é habilitada.

10.1.5.1.15.2 CTU

Bloco de contagem progressiva de pulsos de entrada.

Representação Ladder



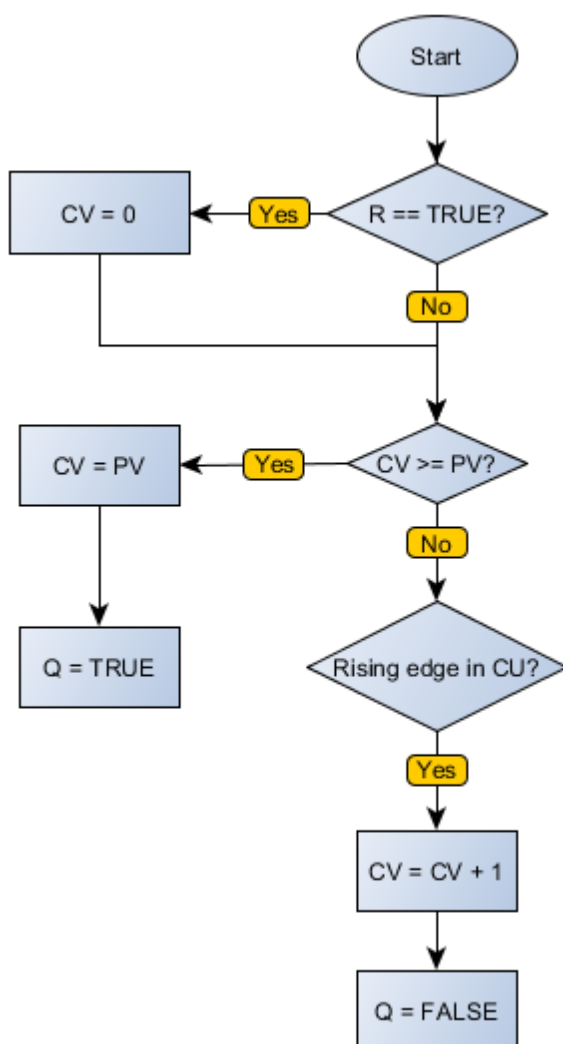
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	CU	BOOL	Identificador de pulsos
	R	BOOL	Carrega o valor zero em CV
	PV	WORD UINT	Valor máximo de contagem
VAR_OUTPUT	Q	BOOL	Sinalizador de contador estourado
	CV	WORD UINT	Valor de contagem atual
VAR	CTU_INST_0	CTU	Instância de acesso à estrutura do bloco

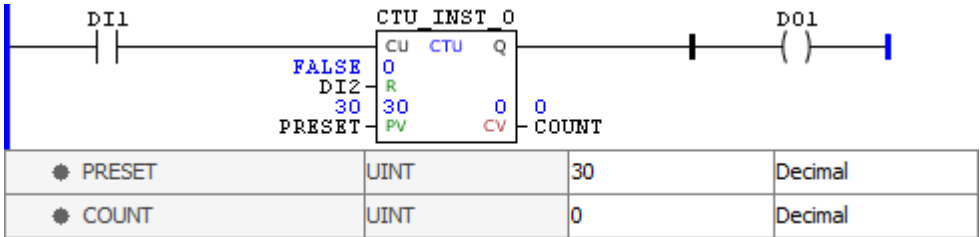
Funcionamento

Este bloco, quando identifica uma borda de subida em CU, incrementa a variável CV até que a mesma seja igual a PV. Enquanto CV for igual a PV, a saída Q permanece em nível TRUE. Ao detectar nível alto em R, o bloco carrega o valor zero em CV.

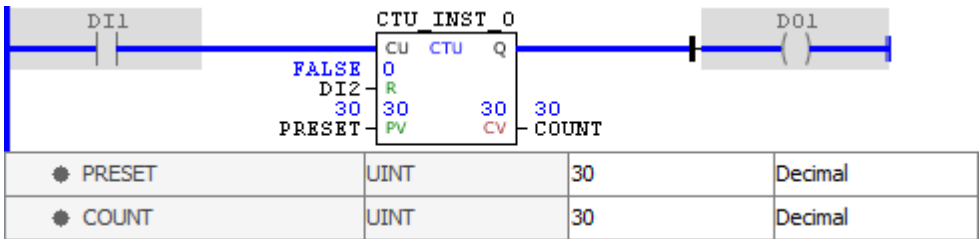
Fluxograma do Bloco



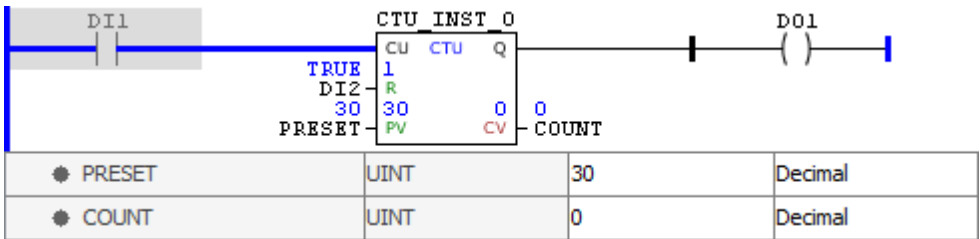
Exemplo



O exemplo acima exibe as condições iniciais da rotina. Como CV possui um valor inferior ao de PV, a saída Q é desabilitada.



A cada borda de subida identificada em CU, o valor de CV é incrementado até chegar ao valor de PV, quando a saída Q é habilitada.

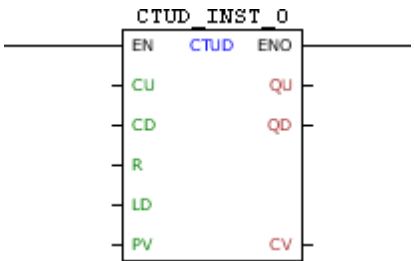


Ao identificar o nível TRUE em R, o bloco carrega o valor zero para CV. Como este valor é inferior ao de PV, a saída Q é desabilitada.

10.1.5.1.15.3 CTUD

Bloco de contagem progressiva e regressiva de pulsos de entrada.

Representação Ladder



Estrutura do Bloco

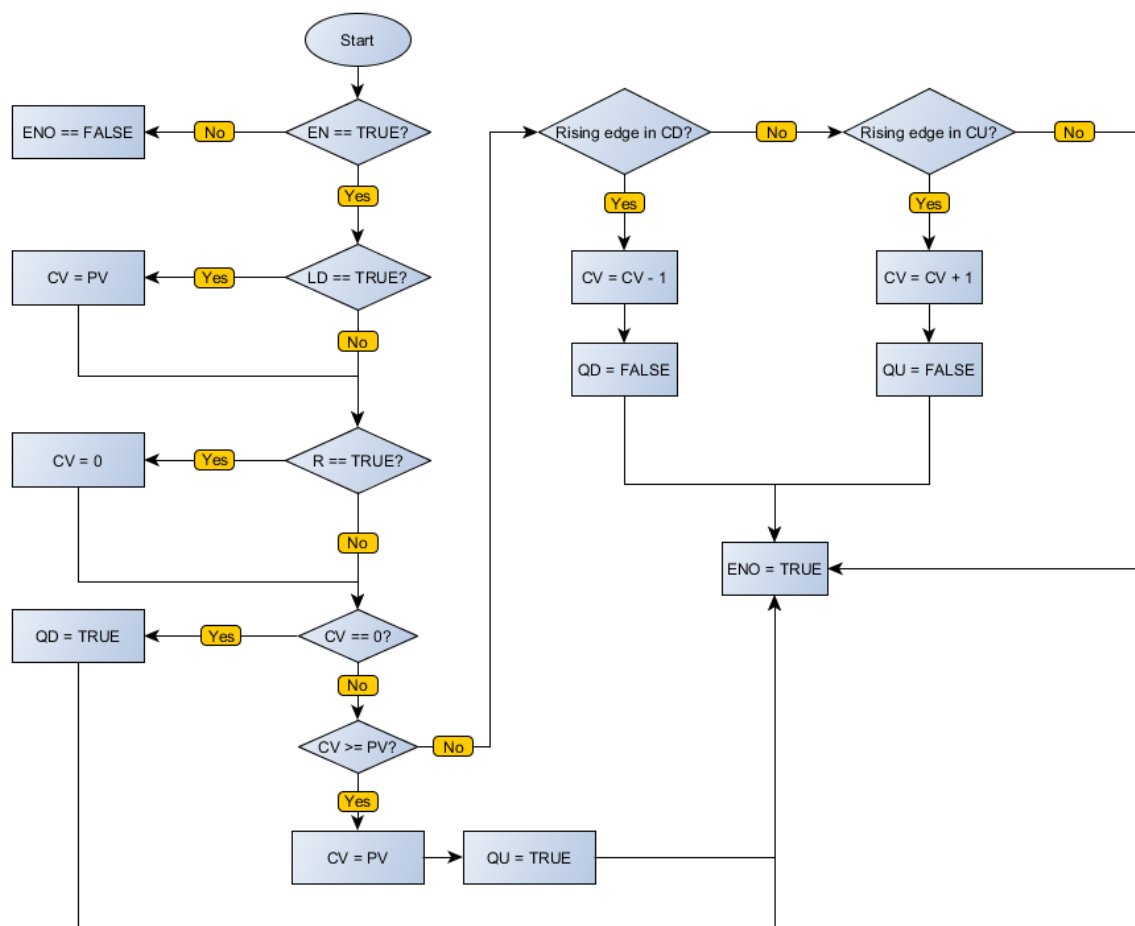
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	CU	BOOL	Identificador de pulsos para incrementador
	CD	BOOL	Identificador de pulsos para decrementador
	R	BOOL	Carrega o valor zero em CV
	LD	BOOL	Carrega o valor de PV em CV
	PV	WORD UINT	Valor de referência
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	QU	BOOL	Sinalizador de contador estourado
	QD	BOOL	Sinalizador de contador zerado
	CV	WORD UINT	Valor de contagem atual
VAR	CTUD_INST_0	CTUD	Instância de acesso à estrutura do bloco

Funcionamento

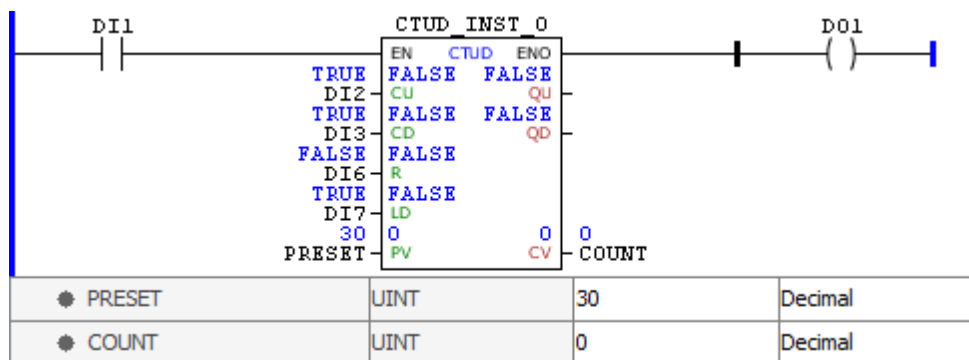
Este bloco, quando tem um valor TRUE em EN, atua como um bloco CTD e um bloco CTU ao mesmo tempo, atuando sobre o mesmo contador CV. Ou seja: incrementa CV até chegar a PV para bordas de subida em CU e decrementa CV até chegar a zero para bordas de subida em CD. Uma transição positiva em R carrega zero em CV, enquanto uma borda de subida em LD carrega o valor de PV em CV. Caso CV tenha valor zero, QD recebe TRUE, e caso CV tenha valor igual a PV, QU recebe TRUE.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN.

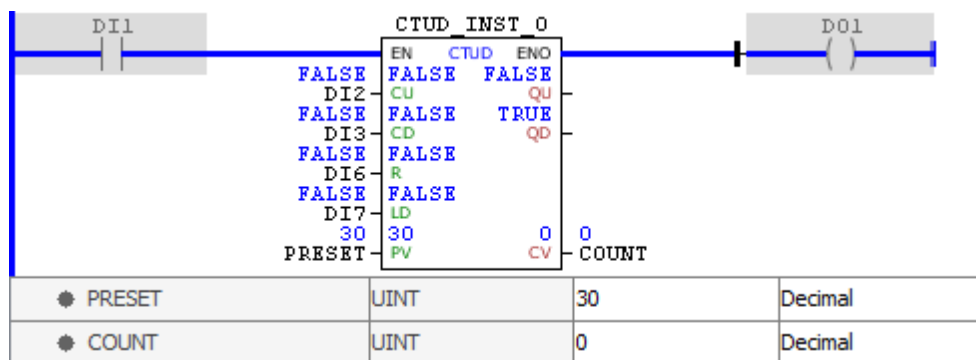
Fluxograma do Bloco



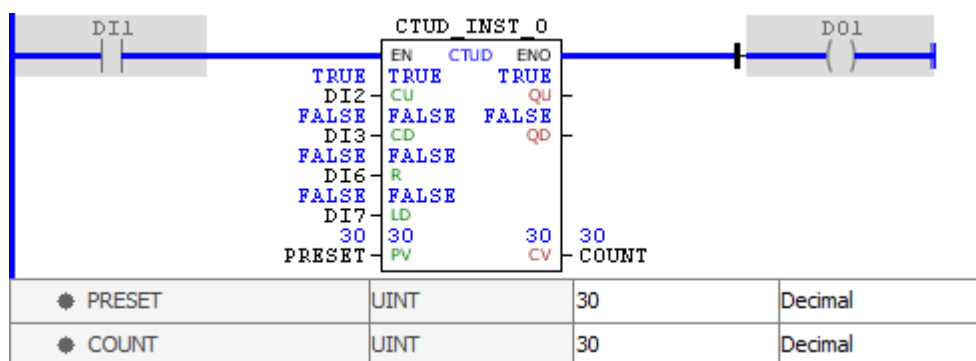
Exemplo



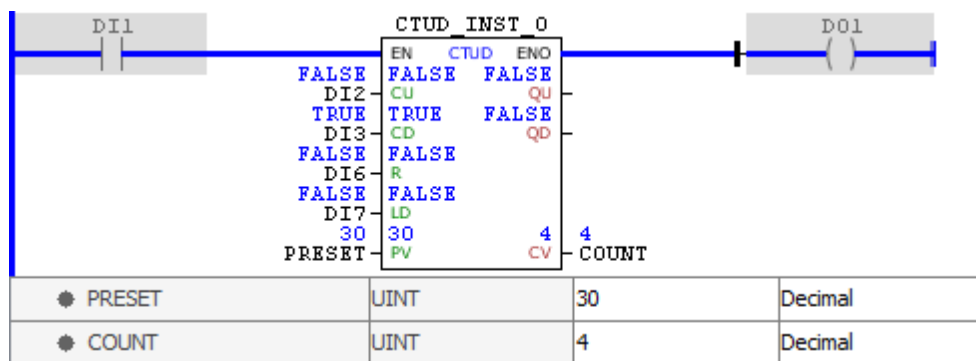
O exemplo acima exibe o bloco desabilitado, com suas variáveis internas todas zeradas. Ainda que os controles externos estejam ativados, esses valores não são repassados para a instância do bloco.



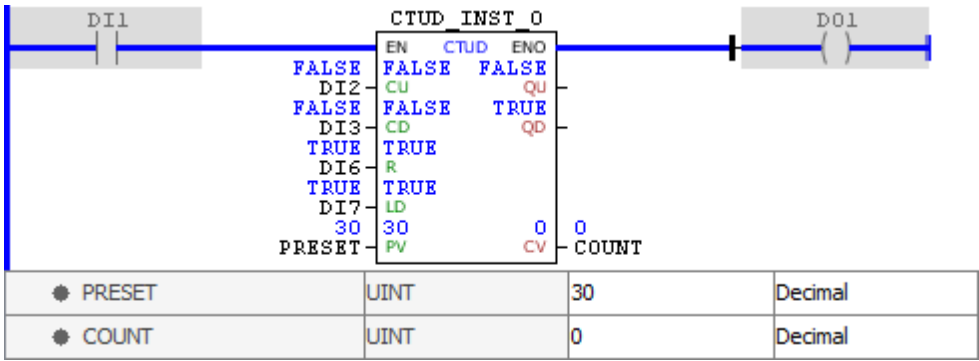
Ao ser ativado, o bloco identifica o valor de PRESET, carregando-o em PV, e identifica que a saída está com o valor zero, habilitando a saída QD. Finalizada a execução, a saída ENO é ativada.



A cada borda de subida identificada em CU, o valor de CV é incrementado até chegar ao valor de PV, quando a saída QU é habilitada. Finalizada a execução, a saída ENO é ativada.



A cada borda de subida identificada em CD, o valor de CV é decrementado. Sendo CV de um valor intermediário a zero e PV, ambas as saídas QD e QU são desativadas. Finalizada a execução, a saída ENO é ativada.



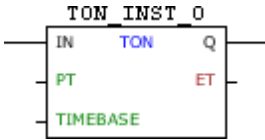
Um valor TRUE em R reseta CV, enquanto um valor TRUE em LD carrega o valor de PV para CV. Como podemos notar, R prevalece sobre LD, restando CV e habilitando a saída QD. Finalizada a execução, a saída ENO é ativada.

10.1.5.1.16 Timer

10.1.5.1.16.1 TON

Bloco de temporizador que, ao ser energizado, habilita a saída após um atraso definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Atraso de acionamento da saída
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TON_INST_0	TON	Instância de acesso à estrutura do bloco

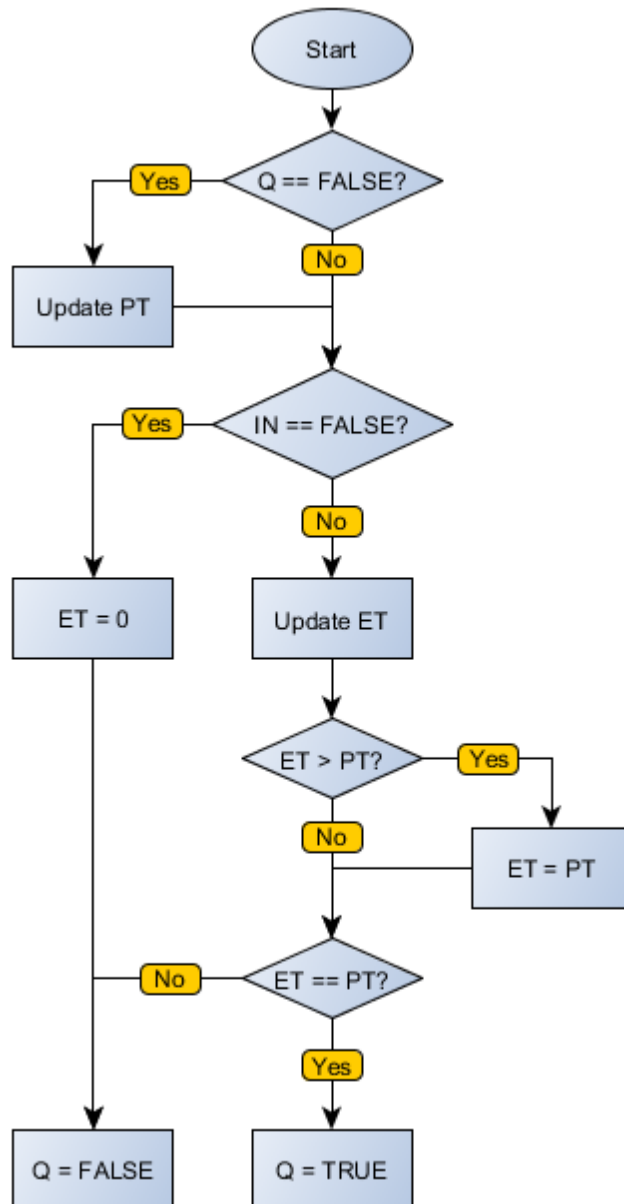
Funcionamento

Enquanto a entrada IN for FALSE, a saída Q também é FALSE e ET recebe o valor zero. Na transição positiva de borda em IN, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado TRUE, até que IN volte a ser FALSE.

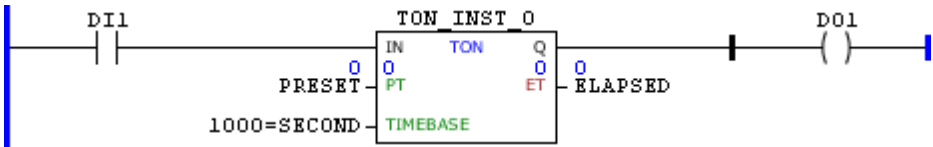
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

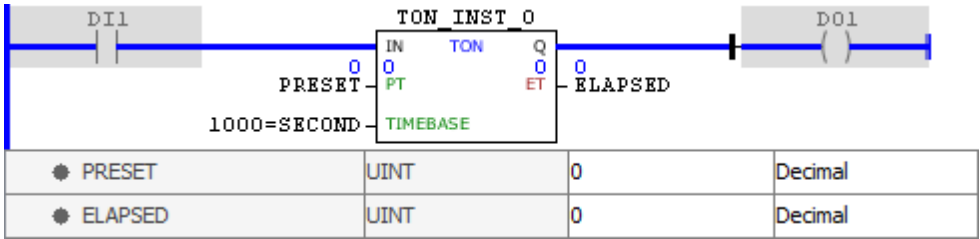
Fluxograma do Bloco



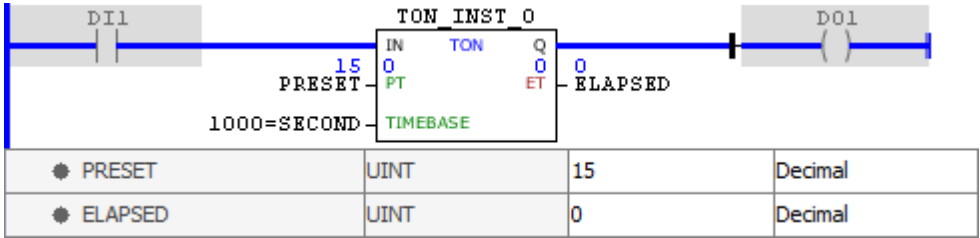
Exemplo



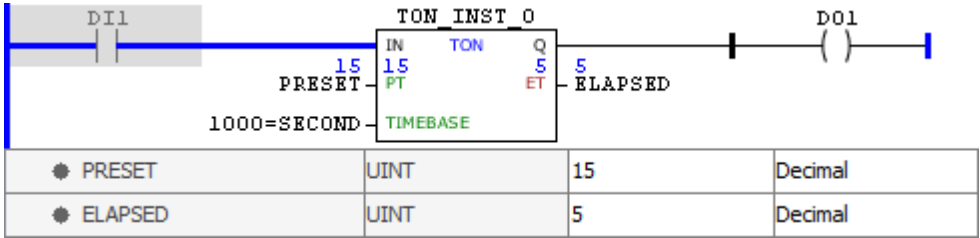
O exemplo acima exibe as condições iniciais do bloco e das variáveis da rotina.



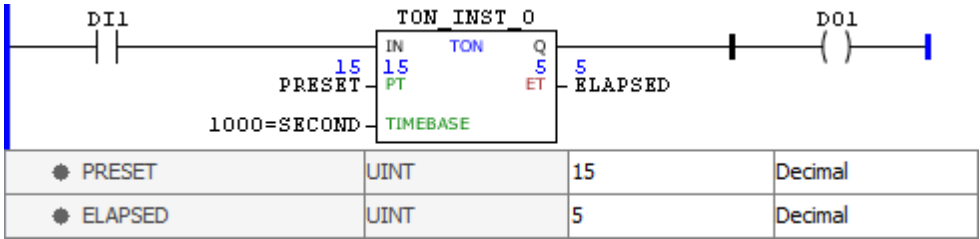
Ao ser ativada a entrada IN, a contagem dispara. Como ET é igual a PT, a saída Q é habilitada.



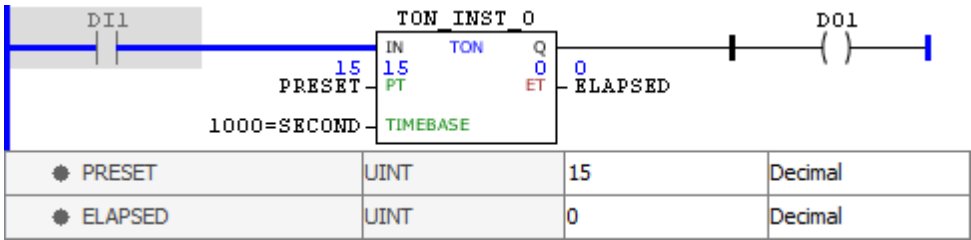
Observe que uma alteração na variável PRESET não é repassada ao campo PT enquanto a entrada IN permanece habilitada.



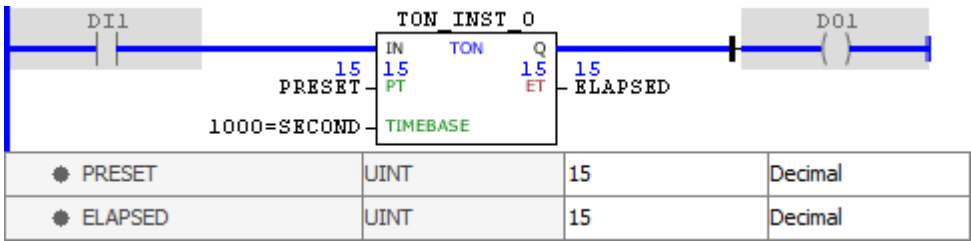
Desabilitando a entrada IN, o valor de PT é atualizado e a saída Q é desabilitada. Ao ativá-la novamente, a contagem dispara.



Desabilitando a entrada IN, o valor de ET permanece salvo.



Habilitando a entrada IN, o valor de ET é resetado e a contagem dispara.



Quando ET atinge o valor de PT, a saída Q é habilitada e permanece assim enquanto IN estiver em nível TRUE.

10.1.5.1.16.2 TOF

Bloco de temporizador que, ao ser desenergizado, desabilita a saída após um atraso definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Atraso de desacionamento da saída
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TOF_INST_0	TOF	Instância de acesso à estrutura do bloco

Funcionamento

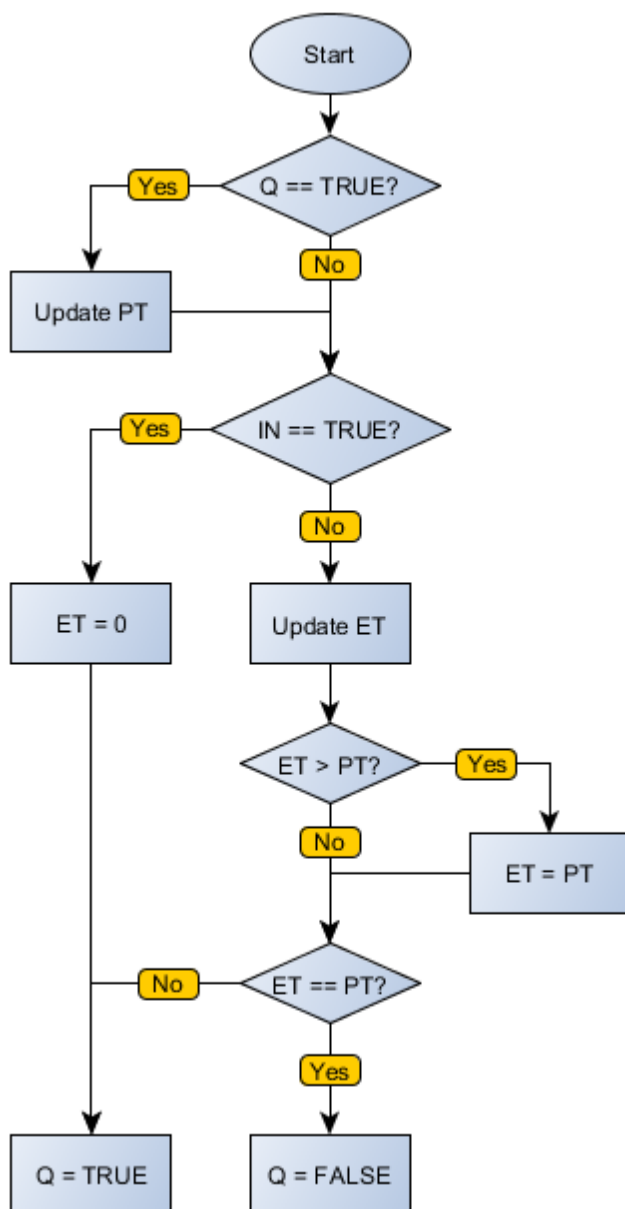
Enquanto a entrada IN for TRUE, a saída Q também é TRUE e ET recebe o valor zero. Na transição negativa de borda em IN, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado FALSE, até que IN volte a ser

TRUE.

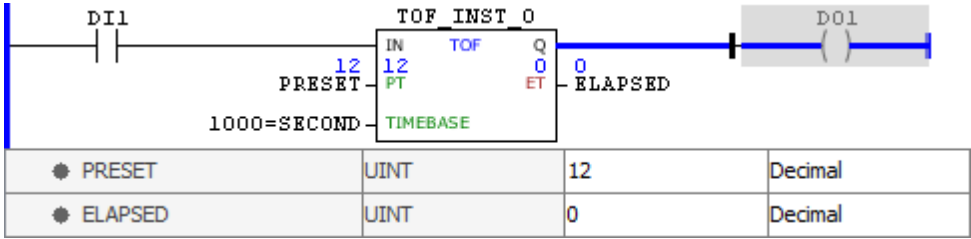
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo

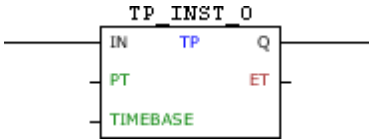


O exemplo acima desabilita a saída DO1 ao identificar um nível baixo em DI1 por 12 segundos, permanecendo desabilitada até que DI1 volte a ser TRUE.

10.1.5.1.16.3 TP

Bloco de temporizador que, ao identificar ser energizado, habilita a saída por um tempo definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Tempo em que a saída fica habilitada
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TP_INST_0	TP	Instância de acesso à estrutura do bloco

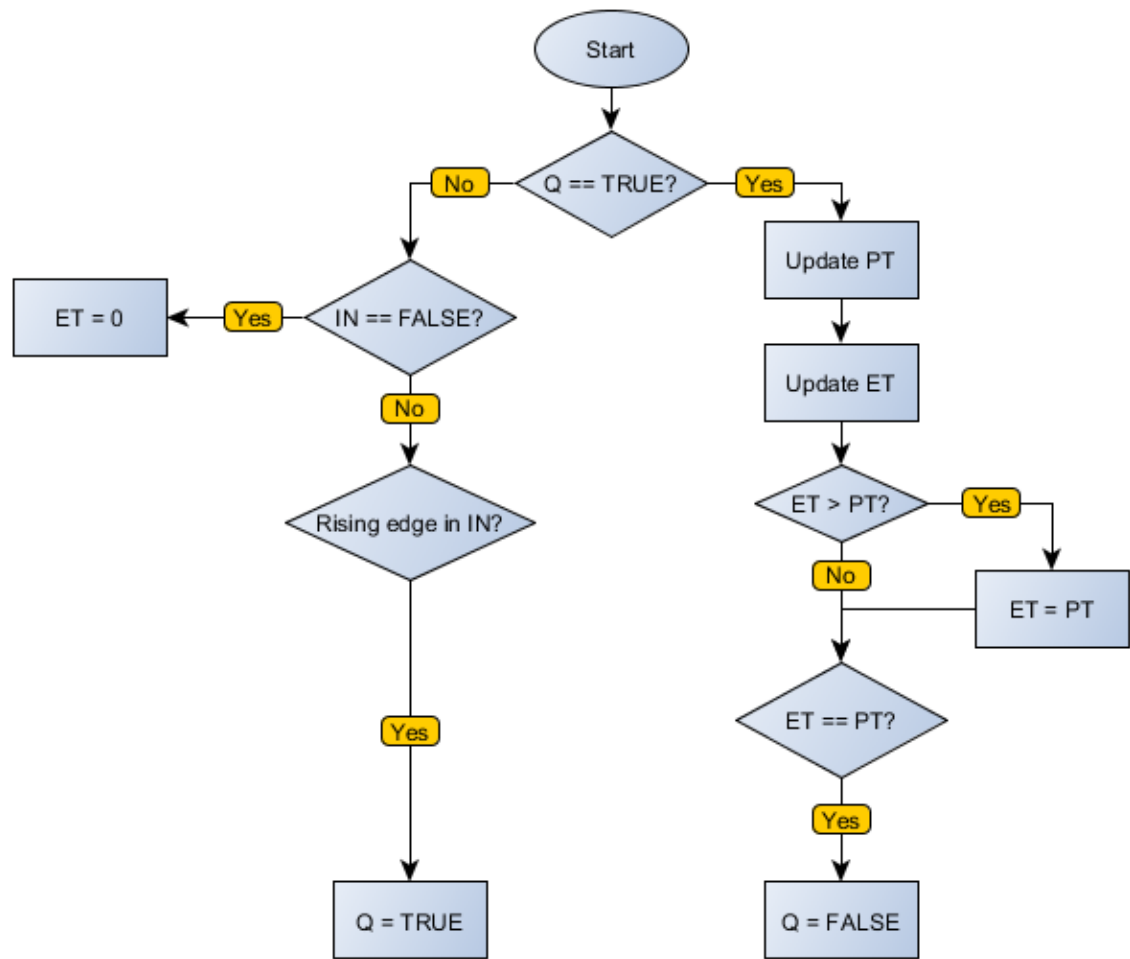
Funcionamento

Na transição positiva de borda em IN, Q recebe valor TRUE, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado FALSE, até que IN volte a ser TRUE. Nesse momento, se IN estiver em nível TRUE, nada acontece. Se IN estiver em nível FALSE, ET é resetado automaticamente.

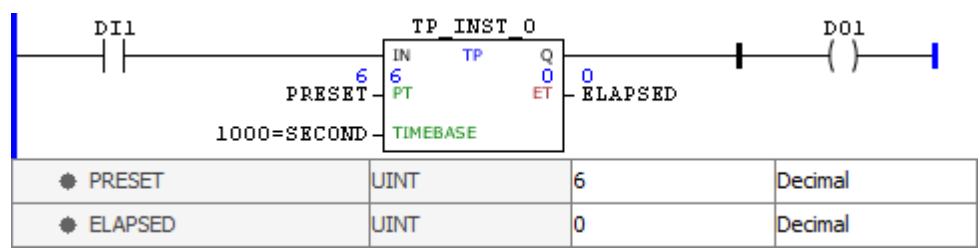
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



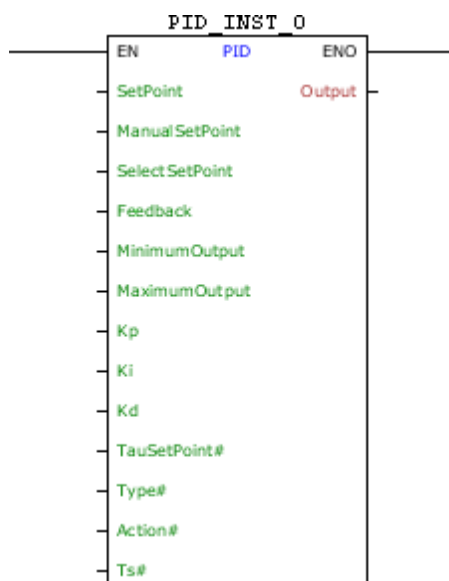
O exemplo acima habilita a saída DO1 por seis segundos a cada transição positiva de DI1.

10.1.5.1.17 Control

10.1.5.1.17.1 PID

Bloco que executa a função de um controlador PID discreto. A partir das variáveis de entrada, ele calcula a saída correspondente do controlador.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SetPoint	REAL	Referência automática (pré-controle)
	ManualSetPoint	REAL	Referência forçada (pós-controle)
	SelectSetPoint	BOOL	Seleciona qual referência utilizar
	Feedback	REAL	Variável de realimentação da malha
	MinimumOutput	REAL	Valor mínimo da saída do controlador
	MaximumOutput	REAL	Valor máximo da saída do controlador
	Kp	REAL	Ganho proporcional
	Ki	REAL	Ganho integral
	Kd	REAL	Ganho derivativo
	TauSetPoint#	REAL	Constante de tempo do filtro de entrada de referência automática
	Type#	BYTE	Tipo de controlador
	Action#	BYTE	Ação de controle
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Output	REAL	Saída do controlador
VAR	PID_INST_0	PID	Instância de acesso à estrutura do bloco

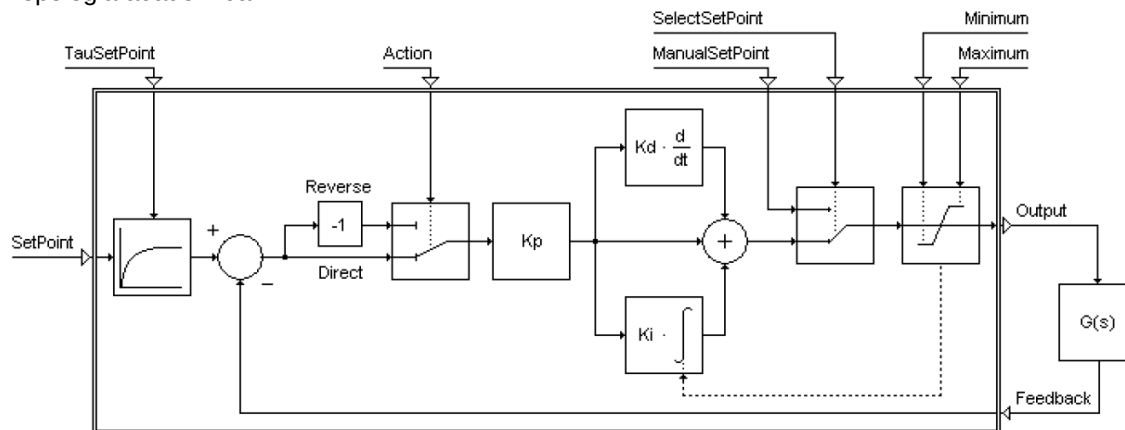
Funcionamento

Na transição positiva de borda em EN, Output recebe valor zero, e o bloco executa sua

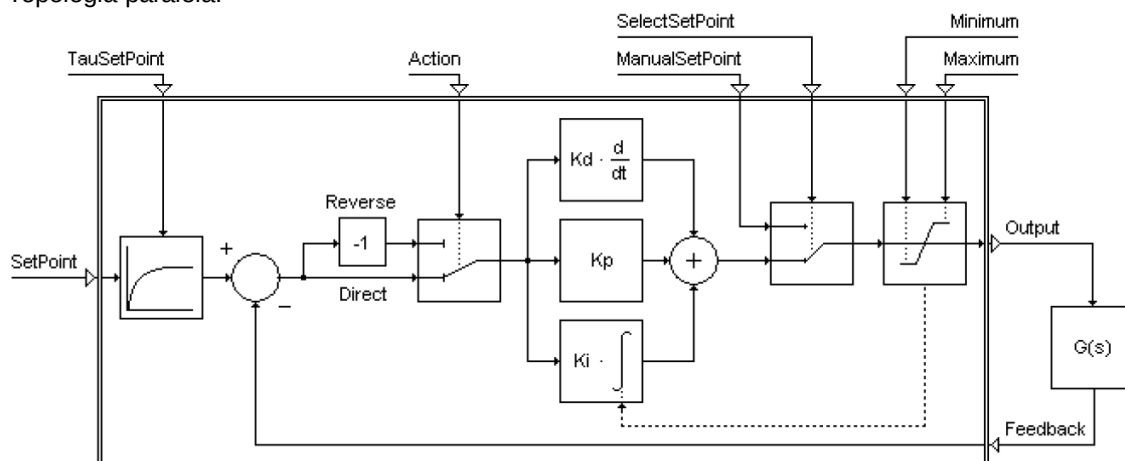
funcionalidade enquanto EN estiver em nível TRUE.

Quando habilitado, este bloco executa uma rotina de controle PID com os parâmetros K_p , K_i e K_d escolhidos. A topologia PID utilizada poderá ser a Acadêmica ou a Paralela, dependendo do que for escolhido em Type#.

Topologia acadêmica:



Topologia paralela:



Os níveis do sinal da saída do controlador são saturados pelo valor de MinimumOutput e MaximumOutput. A entrada SelectSetPoint em nível FALSE faz com que a referência adotada seja SetPoint, permitindo que o controlador exerça domínio sobre o processo. Quando SelectSetPoint vai para nível TRUE, o controlador não exerce mais domínio, sendo que ManualSetPoint passa a ser considerado o sinal de saída do controlador.

Action# definirá a operação de realimentação. Se Action# for DIRECT, a operação será SetPoint – Feedback. Se Action# for REVERSE, a operação será Feedback – SetPoint.

Feedback recebe a variável de processo considerada como saída da planta. Ts# recebe o período de amostragem para o controlador e TauSetPoint# recebe a constante de tempo para o filtro de entrada da referência automática.

Quando EN possui valor FALSE, Output permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

**NOTA!**

Efeito da alteração dos ganhos sobre o processo:

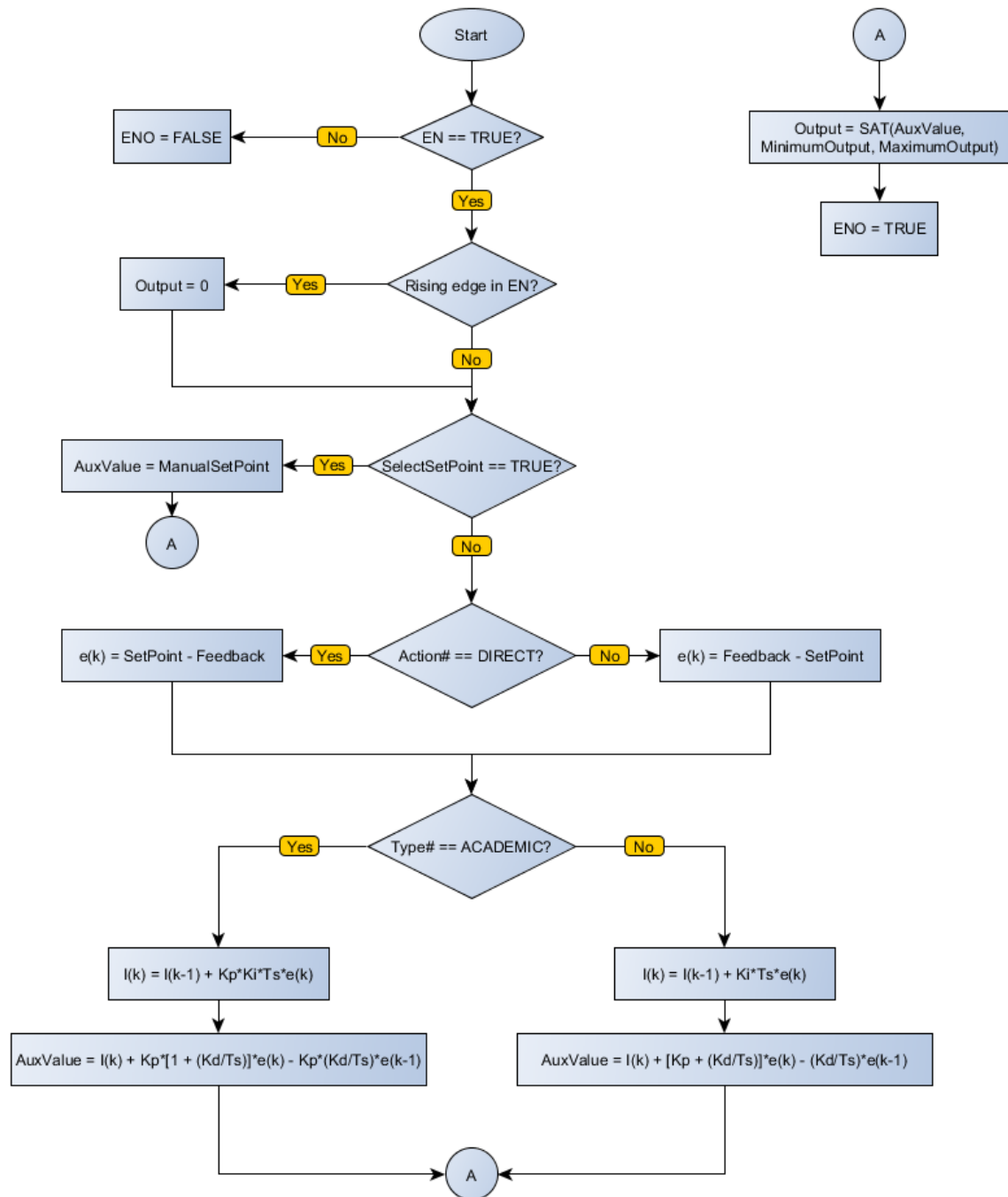
- Se diminuir Kp, o processo torna-se mais lento; geralmente mais estável ou menos oscilante; tem menos sobre-sinal (overshoot).
- Se aumentar Kp, o processo responde mais rápido; pode ficar instável ou mais oscilante; tem mais sobre-sinal (overshoot).
- Se diminuir Ki, o processo torna-se mais lento, demorando para atingir o "SetPoint"; fica mais estável ou menos oscilante; tem menos sobre-sinal (overshoot).
- Se aumentar Ki, o processo torna-se mais rápido, atingindo rapidamente o "SetPoint"; fica mais instável ou mais oscilante; tem mais sobre-sinal (overshoot).
- Se diminuir Kd, o processo torna-se mais lento; tem menos sobre-sinal (overshoot).
- Se aumentar Kd, tem mais sobre-sinal (overshoot).

**NOTA!**

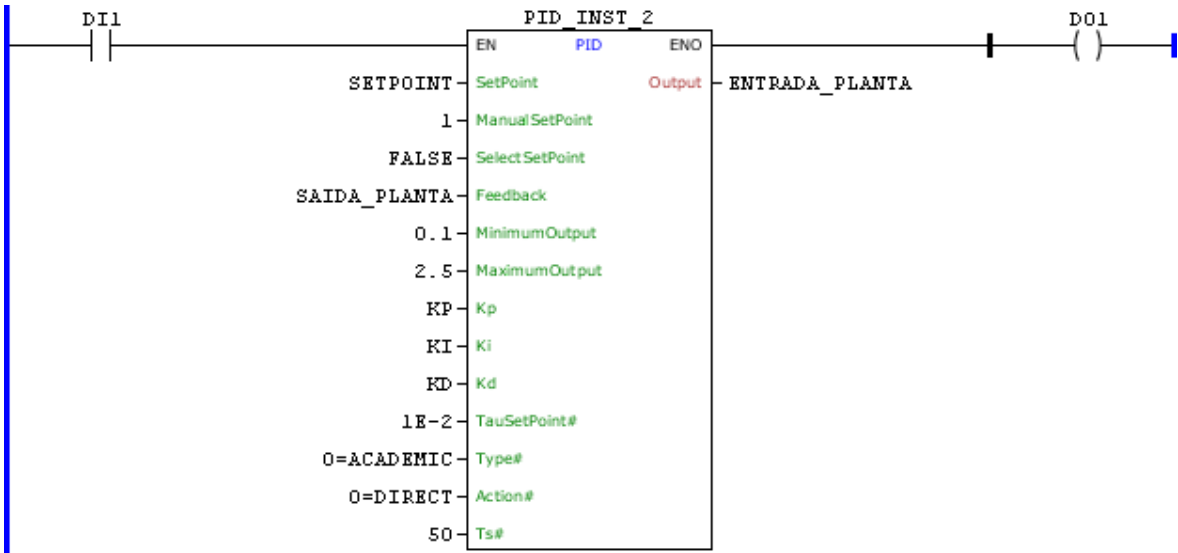
Como melhorar o desempenho do processo pelo ajuste dos ganhos (válido para o PID Acadêmico)

- Se o desempenho do processo está quase bom, mas o sobre-sinal (overshoot) está um pouco alto, tente: (1) diminuir o Kp em 20%, (2) diminuir Ki em 20% e/ou (3) diminuir Kd em 50%.
- Se o desempenho do processo está quase bom, mas não tem o sobre-sinal (overshoot) e demora para atingir o "SetPoint", tente: (1) aumentar Kp em 20%, (2) aumentar Ki em 20%, (3) aumentar Kd em 50%.
- Se o desempenho do processo está bom, mas a saída do processo está variando demais, tente: (1) aumentar Kd em 50%, (2) diminuir Kp em 20%.
- Se o desempenho do processo está ruim, ou seja, após a partida, o transitório dura vários períodos de oscilação, que reduz muito lentamente ou não reduz, tente: (1) diminuir Kp em 50%.
- Se o desempenho do processo está ruim, ou seja, após a partida avança lentamente em direção ao "SetPoint", sem sobre-sinal (overshoot), mas ainda está muito longe e a saída do processo é menor que o valor nominal, tente: (1) aumentar Kp em 50%, (2) aumentar Ki em 50%, (3) aumentar Kd em 70%.

Fluxograma do Bloco



Exemplo



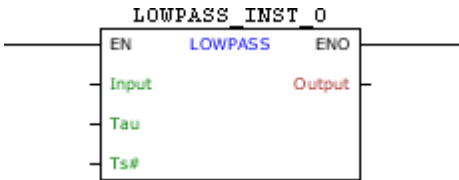
O exemplo acima cria uma malha de um PID acadêmico digital, com tempo de amostragem de 50 ms, utilizando as constantes KP, KI e KD para o controle. É utilizada a referência automática SETPOINT, filtrada por um filtro de primeira ordem com constante de tempo 0.01. O sinal de erro é calculado como a diferença entre a referência filtrada e a variável SAIDA_PLANTA. A saída do controlador é saturada entre os valores 0.1 e 2.5 e enviada para a variável ENTRADA_PLANTA.

10.1.5.1.18 Filter

10.1.5.1.18.1 LOWPASS

Bloco que filtra a entrada utilizando um filtro passa baixas de primeira ordem e insere o resultado na saída.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Input	REAL	Sinal de entrada
	Tau	REAL	Constante de tempo do filtro
	Ts#	UINT	Período de amostragem [ms]
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Output	REAL	Saída do filtro
VAR	LOWPASS_INST_0	LOWPASS	Instância de acesso à estrutura do bloco

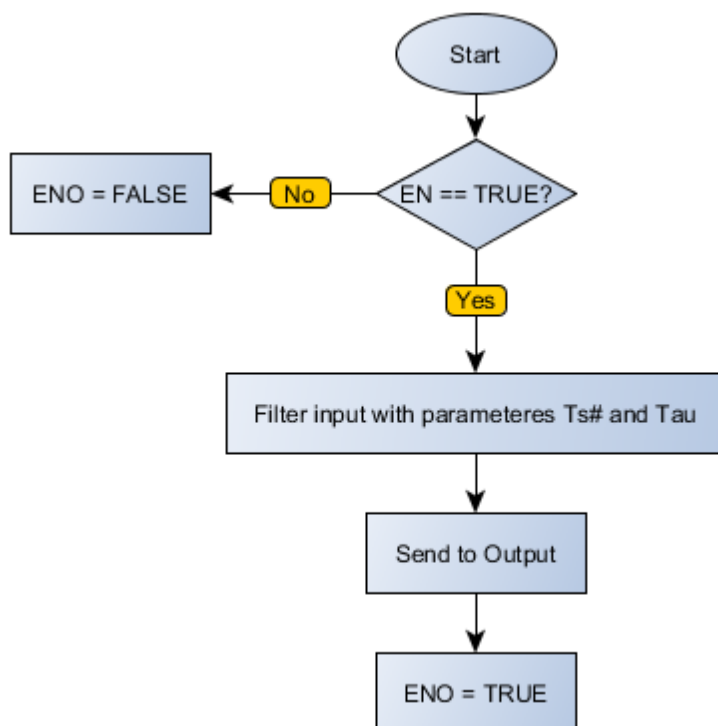
Funcionamento

Este bloco, quando tem um valor TRUE em EN, filtra o valor de entrada Input utilizando um filtro passa baixas de primeira ordem descrito por Tau e Ts#, inserindo o resultado em Output. Na borda de subida de EN, Output recebe zero.

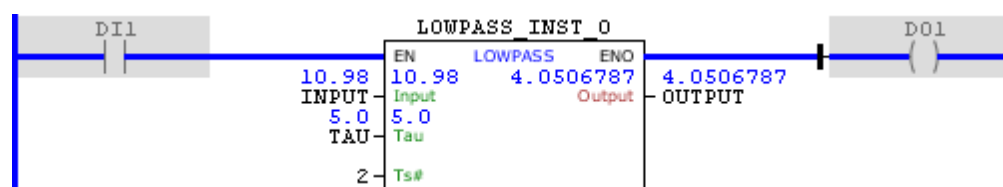
Quando EN possui valor FALSE, Output permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

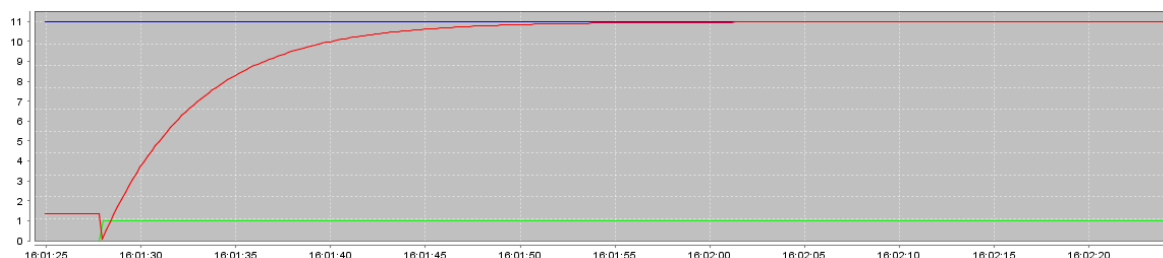
Fluxograma do Bloco



Exemplo



O exemplo acima faz com que a saída OUTPUT, ao identificar uma borda de subida em EN, exiba um comportamento de primeira ordem, com constante de tempo igual a TAU e tempo de amostragem de 2 ms, visando a atingir a referência setada em INPUT. A cada cálculo finalizado com sucesso, a saída ENO é ativada.

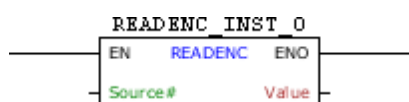


10.1.5.1.19 Hardware

10.1.5.1.19.1 READENC

Bloco que lê continuamente o valor de pulsos de um encoder de acordo com um tipo de leitura escolhido.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source#	BYTE	Fonte de contagem
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Value	DINT DWORD UDINT	Valor de contagem
VAR	READENC_INST_0	READENC	Instância de acesso à estrutura do bloco

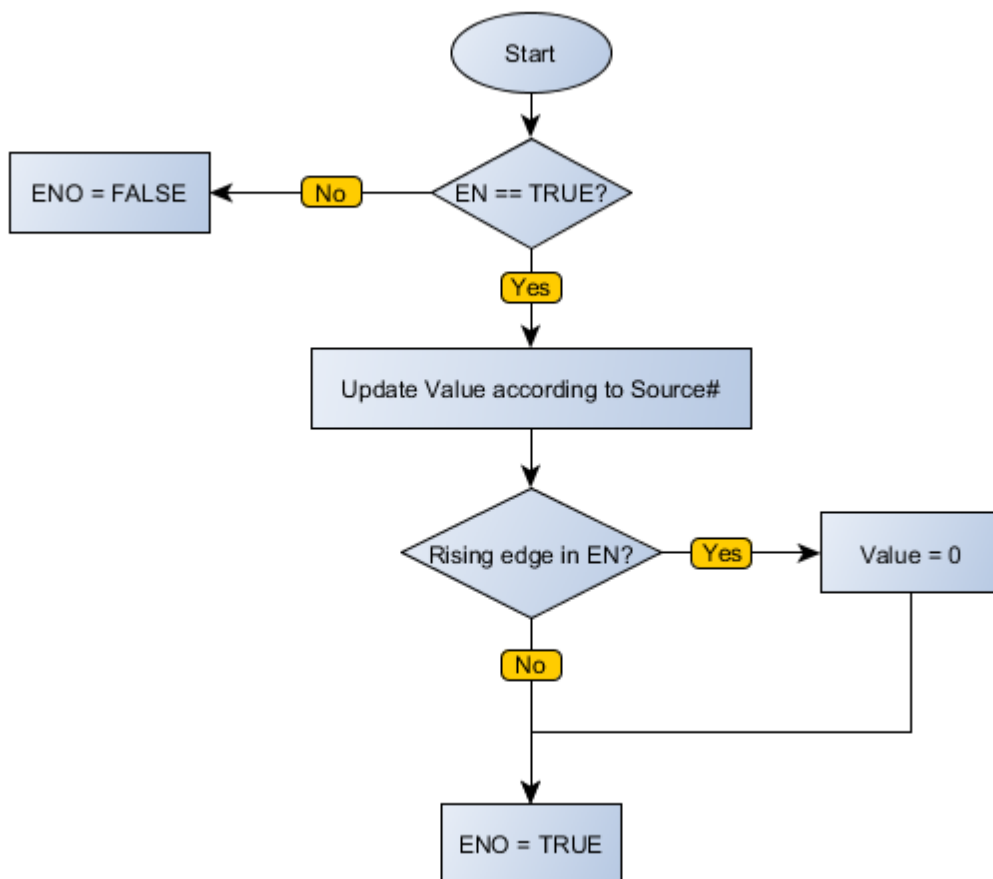
Funcionamento

Este bloco, quando tem um valor TRUE em EN, obtém o número de pulsos contados no encoder, conforme selecionado em Source#, e os transfere para Value. A cada borda de subida, Value é reinicializado em zero.

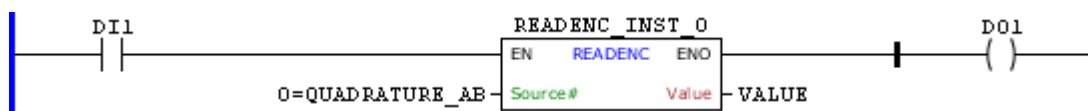
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

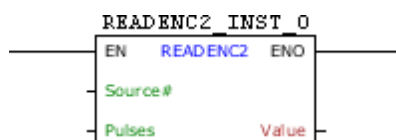


O exemplo acima, ao identificar uma borda de subida em DI1, começa a contar o número de pulsos no encoder ligado em conformação de quadratura AB, armazenando o valor em VALUE. Finalizado o bloco com sucesso, a saída ENO é ativada.

10.1.5.1.19.2 READENC2

Bloco que lê o valor de pulsos de um encoder de acordo com um tipo de leitura escolhido, interpretando-o como fração de voltas.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source#	BYTE	Fonte de contagem
	Pulses	DWORD UDINT	Número de pulsos máximo, relativo ao fim de volta
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Value	DINT DWORD UDINT	Valor de contagem
VAR	READENC2_INST_0	READENC2	Instância de acesso à estrutura do bloco

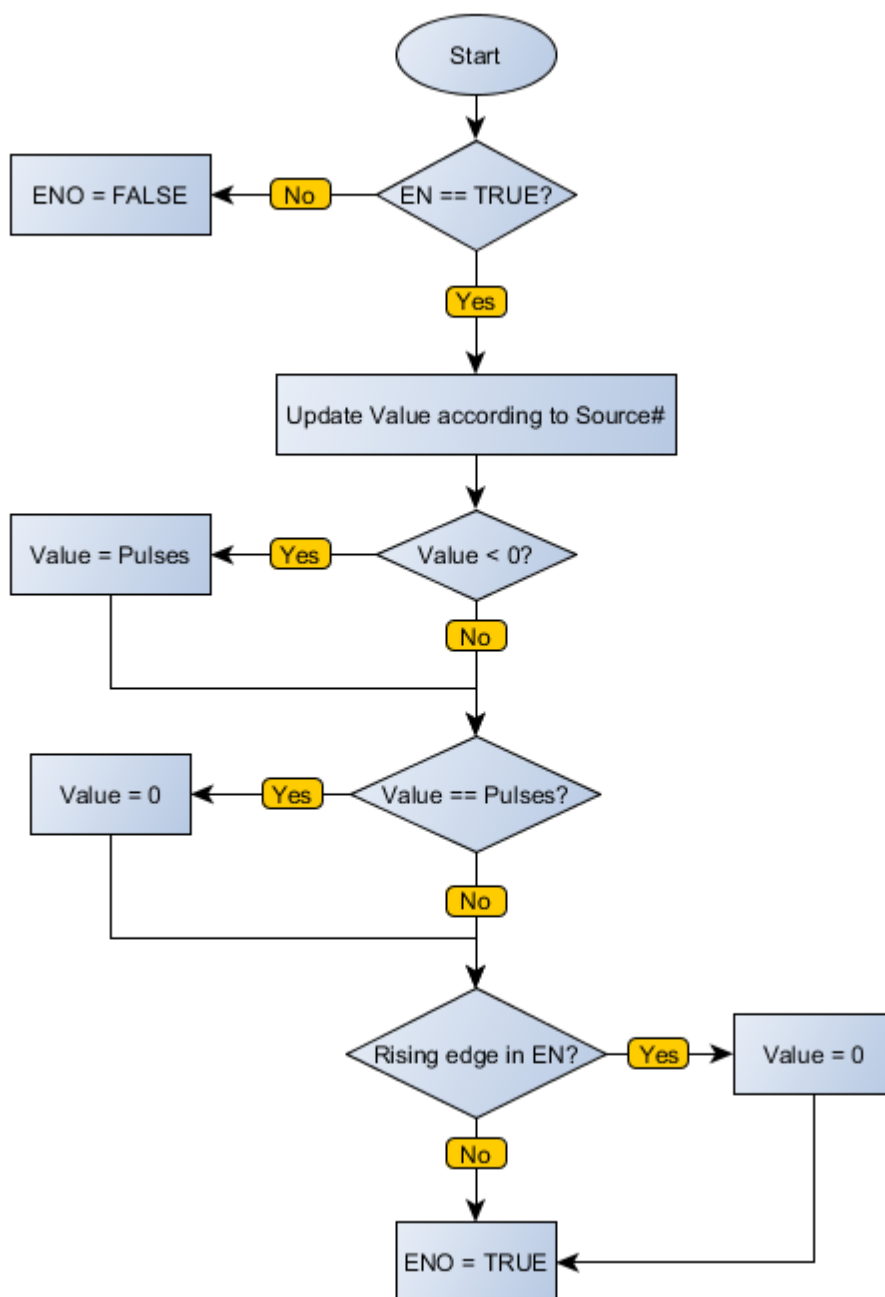
Funcionamento

Este bloco, quando tem um valor TRUE em EN, obtém o número de pulsos contados no encoder, conforme selecionado em Source#, e os transfere para Value. A cada borda de subida, Value é reinicializado em zero. Quando Value incrementa e atinge o valor de Pulses, Value recebe zero. Quando Value decrementa e tende a atingir o valor de -1, Value recebe o valor em Pulses. Dessa forma, Value sempre apresenta o resultado em fração de voltas.

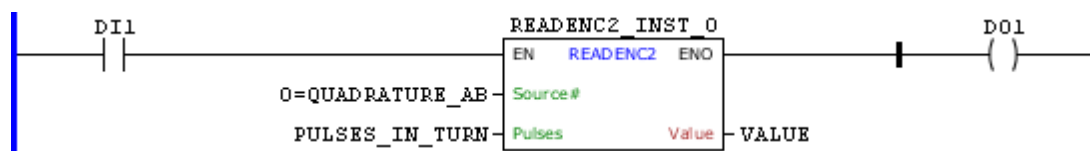
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



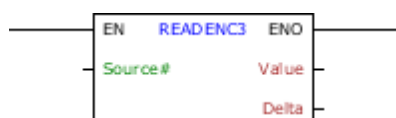
O exemplo acima, ao identificar uma borda de subida em DI1, começa a contar o número de pulsos no encoder ligado em conformação de quadratura AB. PULSES_IN_TURN é o valor de pulsos para

completar uma volta. Caso o valor lido seja superior, retorna a contagem a partir de zero. Caso o valor lido seja negativo, retorna a contagem a partir de PULSES_IN_TURN. O valor resultante é armazenado em VALUE. Finalizado o bloco com sucesso, a saída ENO é ativada.

10.1.5.1.19.3 READENC3

Bloco que lê o valor de pulsos de um encoder de acordo com um tipo de leitura escolhido, calculando diferenciais entre chamadas ao bloco.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source#	BYTE	Fonte de contagem
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Value	DINT DWORD UDINT	Valor de contagem
	Delta	DINT DWORD UDINT	Diferencial entre contagem atual e contagem anterior

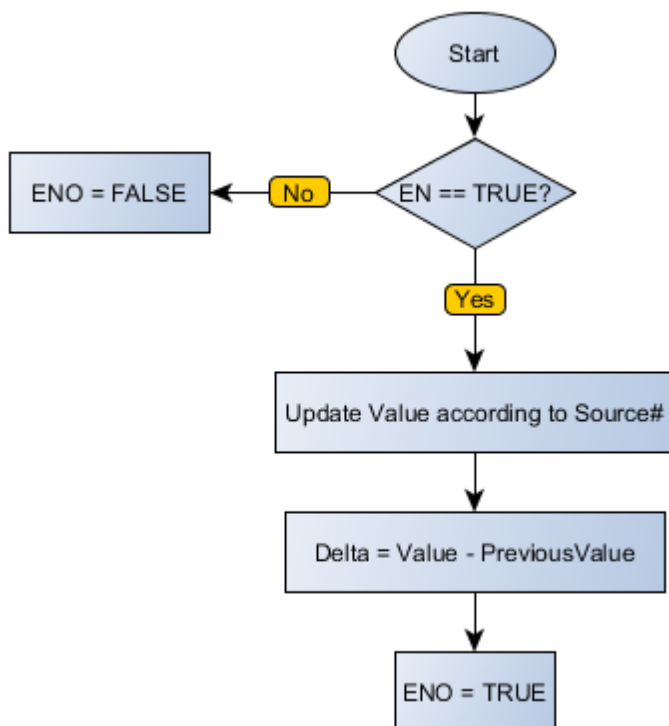
Funcionamento

Este bloco, quando tem um valor TRUE em EN, obtém o número de pulsos contados no encoder desde a energização do PLC300, conforme selecionado em Source#, e os transfere para Value. O valor de Delta será a diferença entre os pulsos contados nessa execução e na execução anterior.

Quando EN possui valor FALSE, Value e Delta permanecem inalterados.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

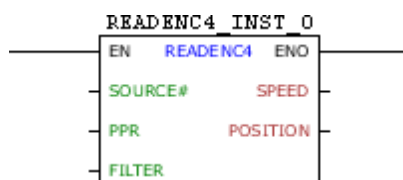


O exemplo acima, ao identificar uma borda de subida em DI1, obtém o número de pulsos na entrada A desde a energização do equipamento, armazenando o valor em VALUE. A diferença entre o atual valor e o valor lido anteriormente é armazenado em DIFFERENCE. Finalizado o bloco com sucesso, a saída ENO é ativada.

10.1.5.1.19.4 READENC4

Bloco que lê o valor de pulsos de um encoder de acordo com um tipo de leitura escolhido, exibindo para o usuário a posição dele atual em voltas e a velocidade em RPM.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SOURCE#	BYTE	Fonte de contagem
	PPR	WORD	Número de pulsos correspondente a uma volta
	FILTER	REAL	Constante de tempo do filtro de entrada
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	SPEED	REAL	Velocidade de rotação em RPM
	POSITION	REAL	Posição atual do encoder, em voltas
VAR	READENC4_INST_0	READENC4	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, obtém o número de pulsos contados no encoder desde a energização do PLC300, conforme selecionado em SOURCE#, e calcula o número de voltas absolutas através do argumento PPR, inserindo o resultado em POSITION. O valor de SPEED é calculado por meio de uma base interna de tempo do bloco. Além disso, o bloco permite a filtragem do sinal com um filtro de saída determinado por FILTER.

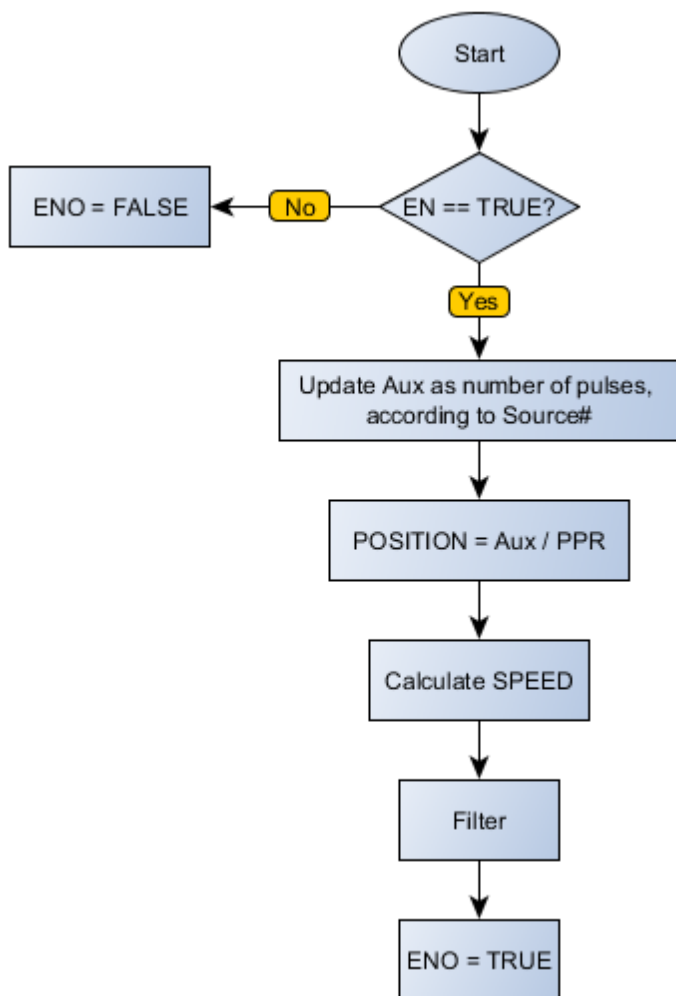
Quando EN possui valor FALSE, POSITION e SPEED permanecem inalterados.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

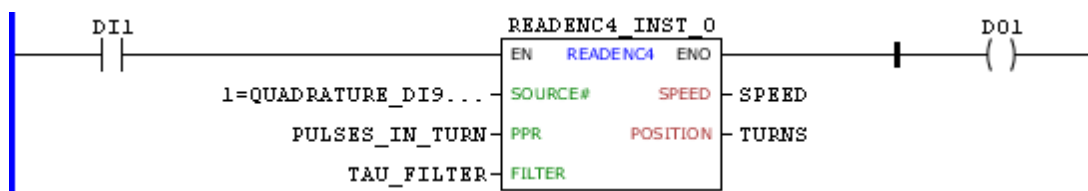
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

Fluxograma do Bloco



Exemplo

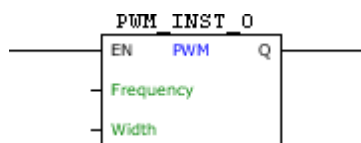


O exemplo acima, ao identificar uma borda de subida em DI1, obtém o número de pulsos no encoder ligado em conformação de quadratura nas entradas rápidas desde a energização do equipamento. PULSES_IN_TURN é o valor de pulsos para completar uma volta. TURNS armazenada o valor dos pulsos interpretado como voltas, de acordo com PULSES_IN_TURN, e SPEED armzena a velocidade atual do encoder. Finalizado o bloco com sucesso, a saída ENO é ativada.

10.1.5.1.19.5 PWM

Bloco que insere um sinal PWM na saída digital DO9.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Frequency	DWORD UDINT	Frequência do PWM
	Width	WORD	Largura do pulso
VAR_OUTPUT	Q	BOOL	Habilitação de saída
VAR	PWM_INST_0	PWM	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, insere em DO9 um sinal PWM com frequência determinada em Frequency (entre 0 e 300 kHz) e largura de pulso determinada por Width (entre 0 e 1000, onde 1000 seria 100% de ciclo ativo).

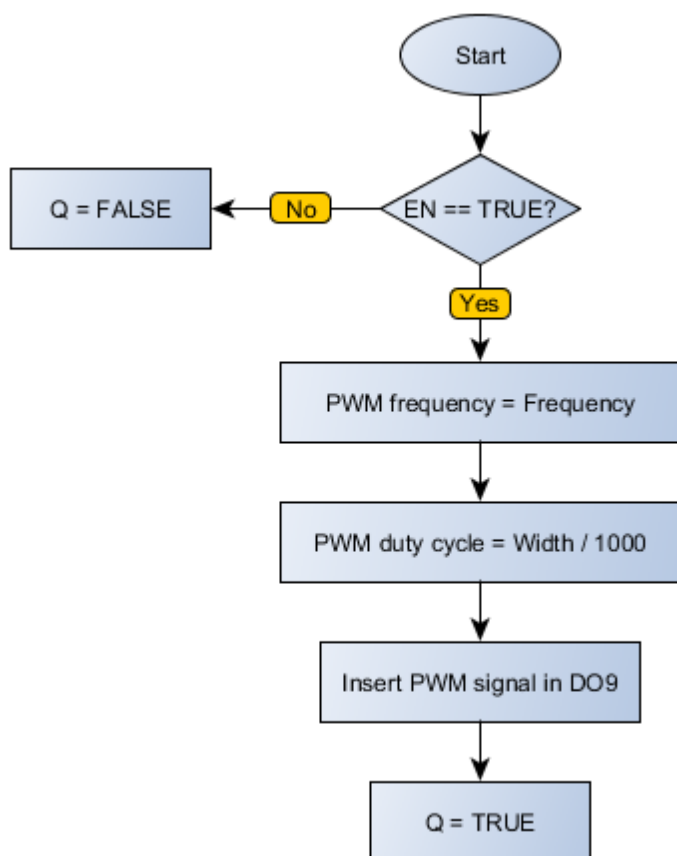


NOTA!

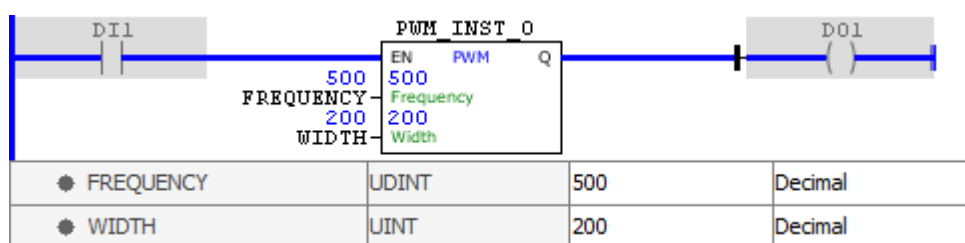
Este bloco tem prioridade sobre qualquer bobina que esteja escrevendo dados em DO9.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

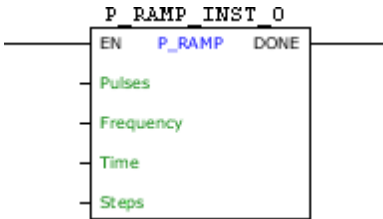


O exemplo acima habilita um sinal PWM na saída DO9 com frequência de 500 Hz e duty cycle de 20%. Finalizado o bloco com sucesso, a saída Q é ativada.

10.1.5.1.19.6 P_RAMP

Bloco que insere uma rampa PWM na saída digital DO9.

Representação Ladder

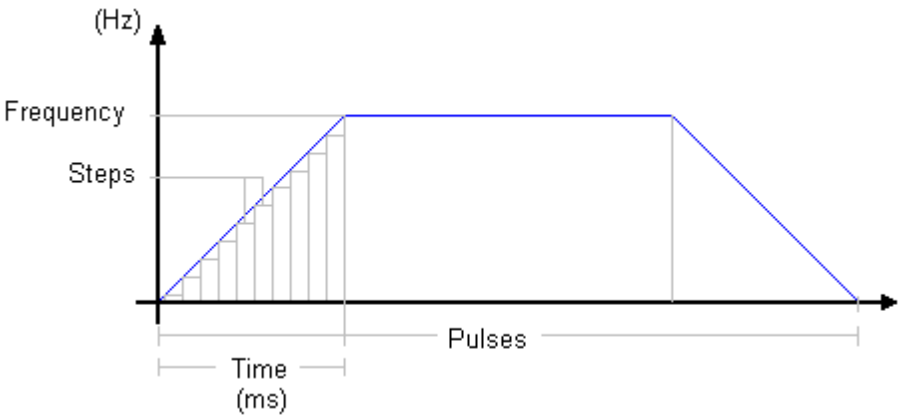


Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Pulses	DWORD UDINT	Número de pulsos total da execução do bloco
	Frequency	DWORD UDINT	Frequência máxima que se deseja atingir
	Time	DWORD UDINT	Tempo de rampa [ms]
	Steps	BYTE USINT	Passos de incremento de frequência
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
VAR	P_RAMP_INST_0	P_RAMP	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, insere em DO9 um sinal PWM de frequência nula, incrementando-a passo a passo, como determinado em Steps, até atingir o valor de Frequency (entre 0 e 200 kHz). A frequência começará a decrementar até zerar novamente no fim de execução do bloco, para que a rampa de aceleração seja igual à de desaceleração.



O valor de Frequency deve ser sempre maior ou igual a 4 vezes o valor de Steps. Caso contrário, não será gerada rampa, apenas o número de pulsos determinado em Pulses na frequência máxima.

O duty cycle do sinal mantém-se constante em 0.5 durante toda a execução do bloco.

Quando EN recebe FALSE, a saída DO9 é zerada imediatamente.

O valor de DONE repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

**NOTA!**

Havendo outros blocos PWM em execução, terá prioridade aquele que for chamado primeiro.

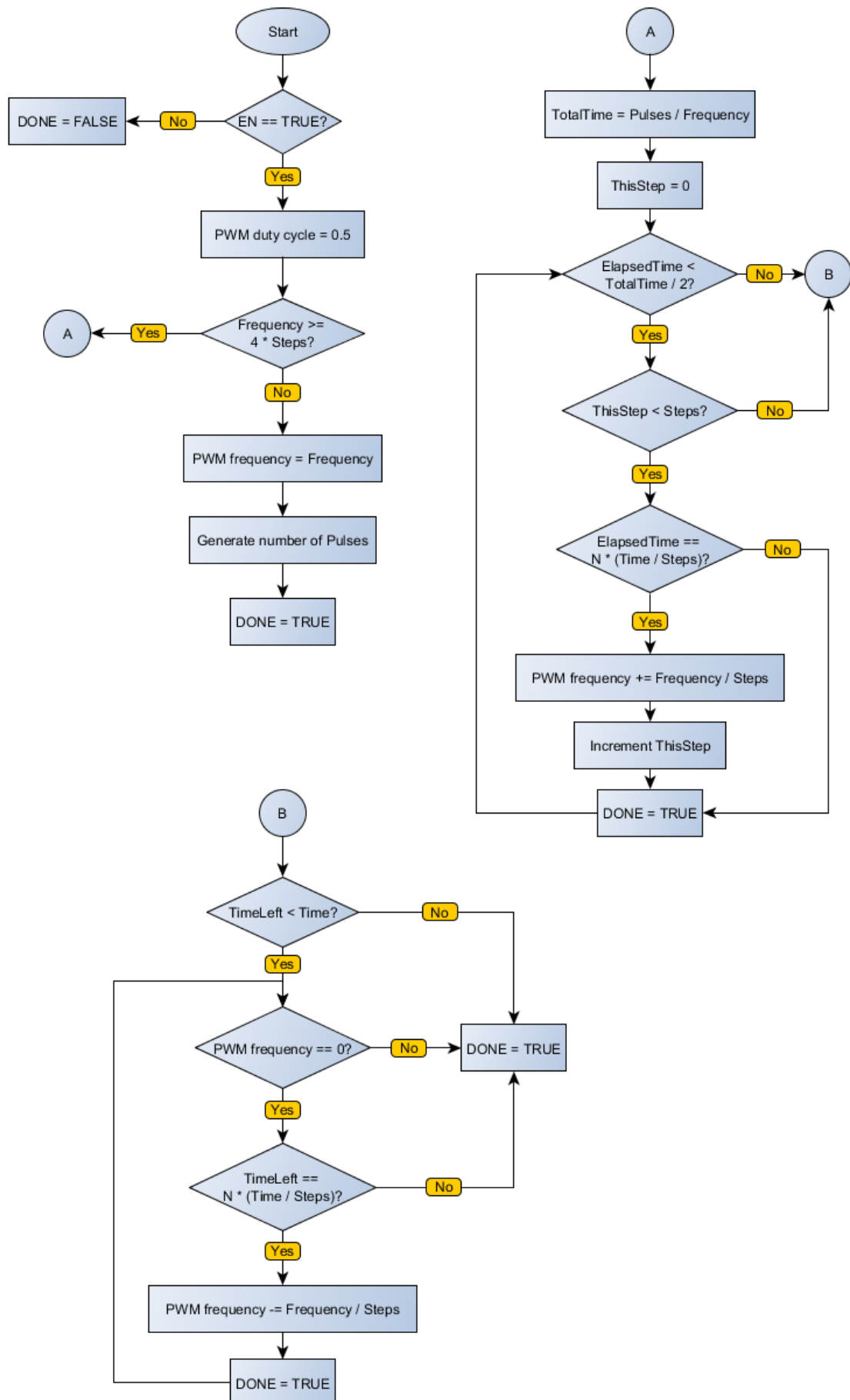
**NOTA!**

Este bloco tem prioridade sobre qualquer bobina que esteja escrevendo dados em DO9.

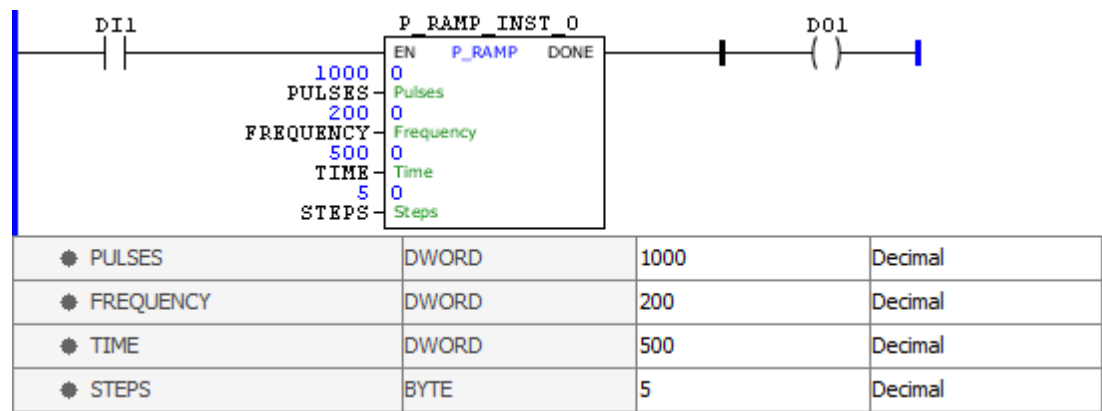
Compatibilidade

Equipamento	Versão
PLC300	2.00 ou superior
SCA06	2.00 ou superior

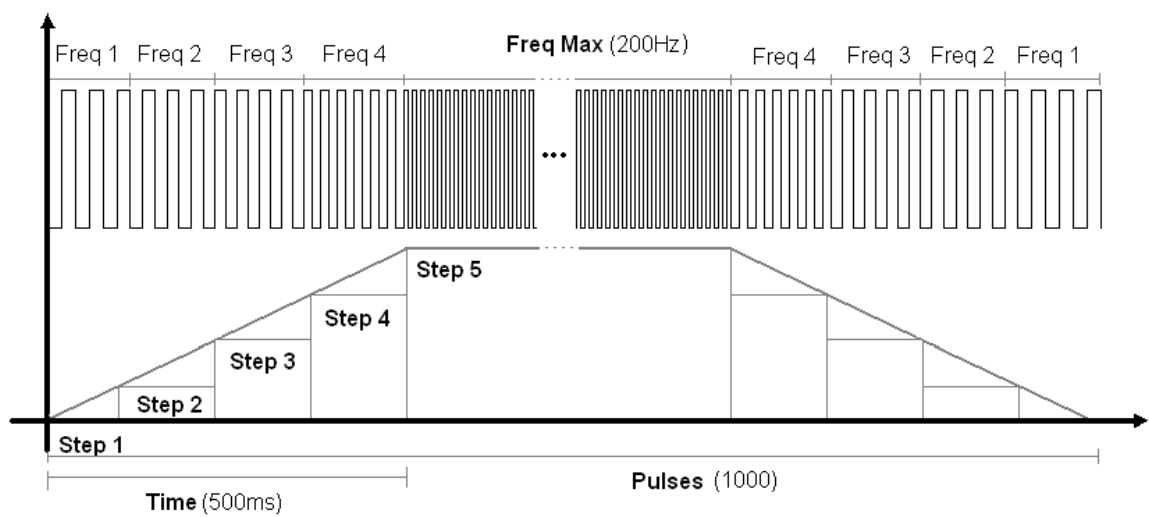
Fluxograma do Bloco



Exemplo



O exemplo acima habilita um sinal PWM na saída DO9 com frequência inicial de 0 Hz e duty cycle de 50%. O tempo da rampa de subida de frequência é de 500 ms, e a rampa possui 5 passos de frequência (0, 50, 100, 150 e 200 Hz). No total da execução do bloco, 1000 pulsos serão enviado à saída. Finalizado o bloco com sucesso, a saída Q é ativada.



10.1.5.1.19.7 IMMEDIATE_INPUT

Bloco que realiza uma leitura instantânea do valor da entrada selecionada, sem alterar o valor das imagens (variáveis GLOBAL_IO).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source#	BYTE	Entradas a serem lidas (digitais ou analógicas)
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Destination	WORD INT UINT	Variável mapeada com os valores das entradas selecionadas

Funcionamento

Este bloco, quando tem um valor TRUE em EN, obtém o valor imediato da entrada selecionada em Source#. Caso seja selecionada a entrada analógica AI1, seu valor é repassado a Destination. Caso sejam selecionadas as entradas digitais, seus bits são concatenados de forma a DI1 ser o bit menos significativo e DI10 ser o bit mais significativo, sendo o resultado enviado a Destination.

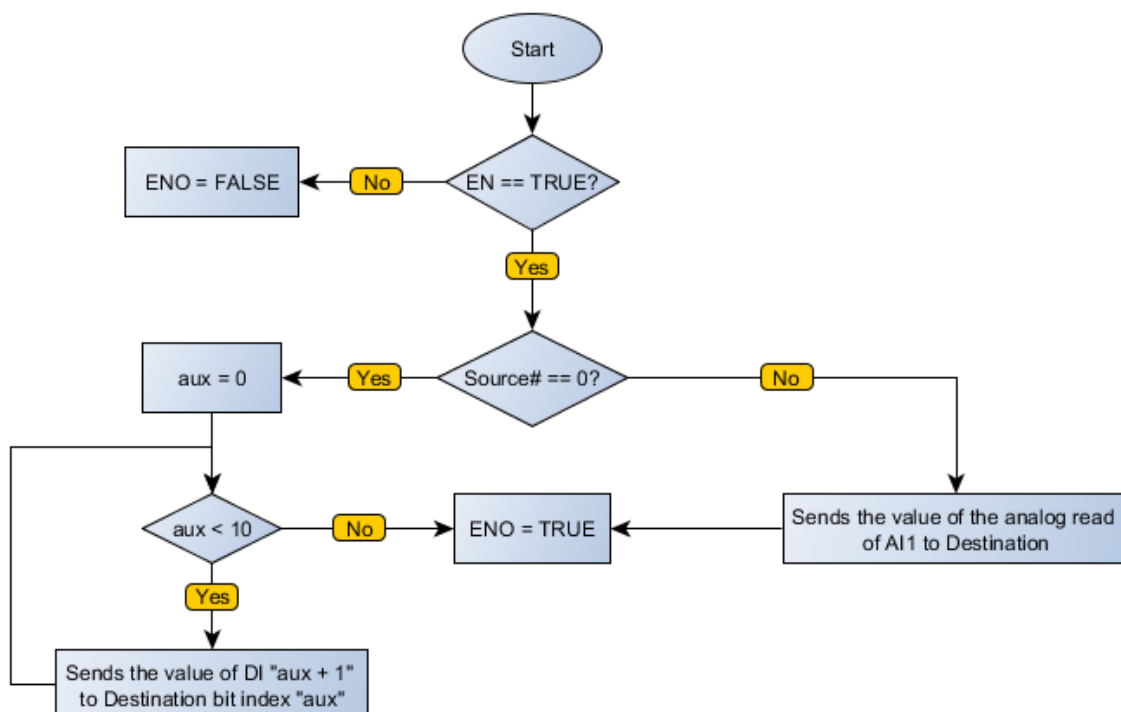
Quando EN possui valor FALSE, Destination permanece inalterada.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

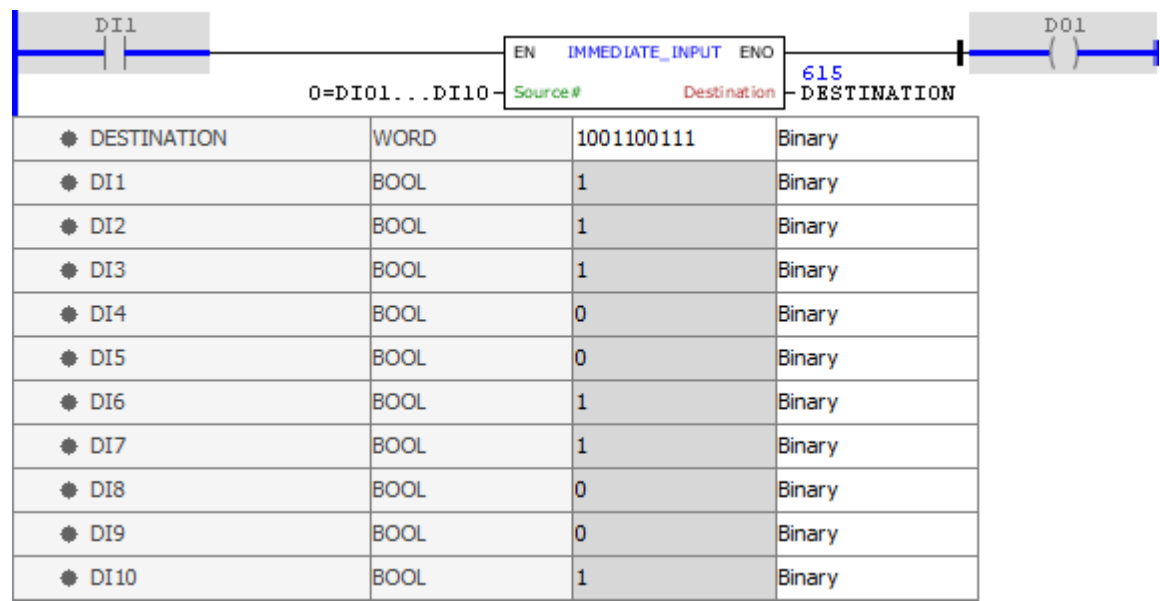
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo

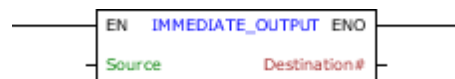


O exemplo acima faz uma leitura imediata dos sinais das entradas digitais DI1 a DI10 do PLC300. Essa leitura é então interpretada como uma sequência de binários, com DI1 sendo o bit menos significativo, e o resultado é enviado à variável DESTINATION. Finalizado o bloco com sucesso, a saída ENO é ativada.

10.1.5.1.19.8 IMMEDIATE_OUTPUT

Bloco que realiza uma escrita instantânea na porta da saída selecionada, sem alterar o valor das imagens (variáveis GLOBAL_IO).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source	WORD INT UINT	Variável mapeada com os valores a serem enviados às saídas selecionadas
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Destination#	BYTE	Saídas a serem escritas (digitais ou analógicas)

Funcionamento

Este bloco, quando tem um valor TRUE em EN, escreve imediatamente na saída selecionada o valor de Source. Caso seja selecionada a saída analógica AO1, o valor de Source é repassado à esta. Caso sejam selecionadas as saídas digitais, DO1 receberá o bit zero de Source, DO2 o bit um, DO3

o bit dois, e assim sucessivamente.

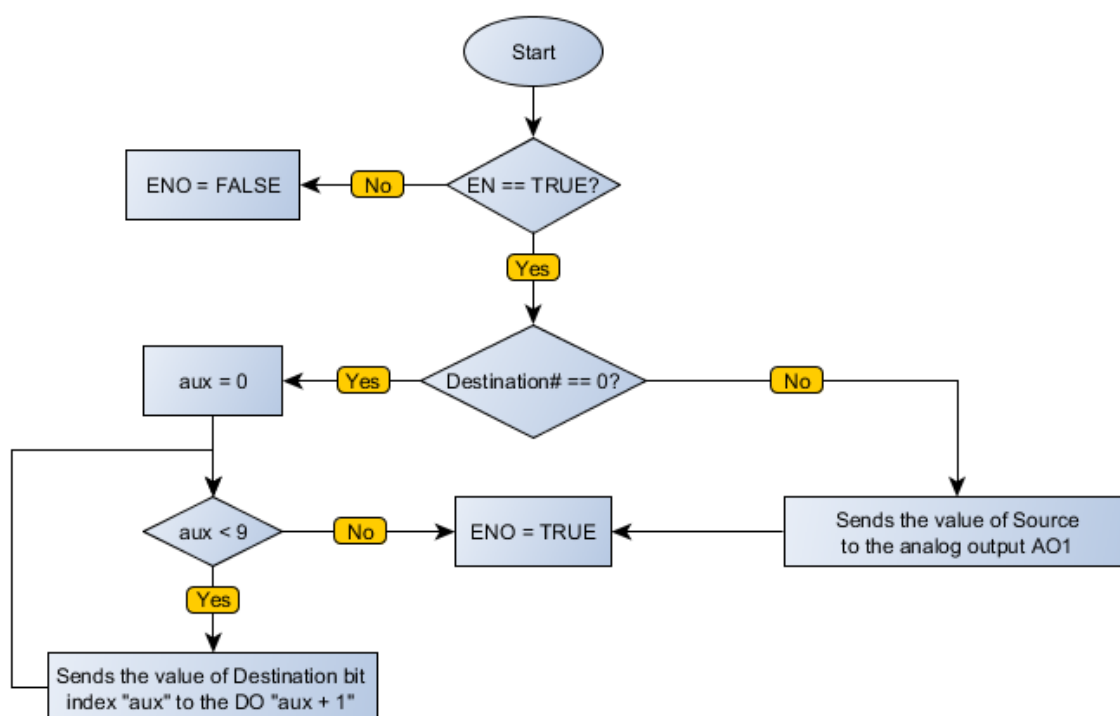
Quando EN possui valor FALSE, Destination# permanece inalterada.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

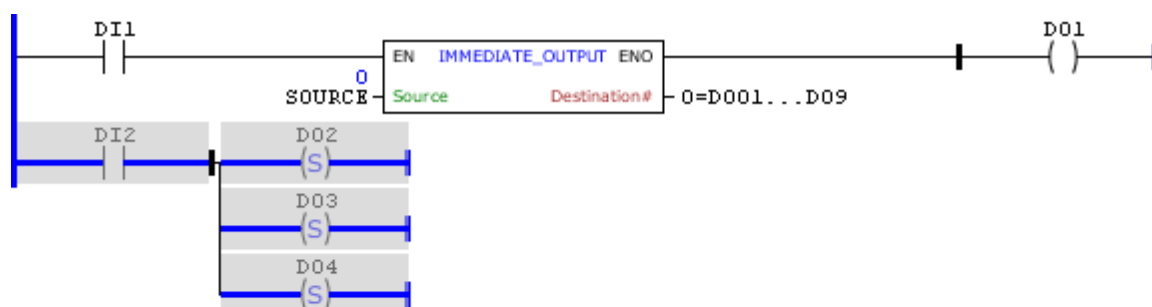
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

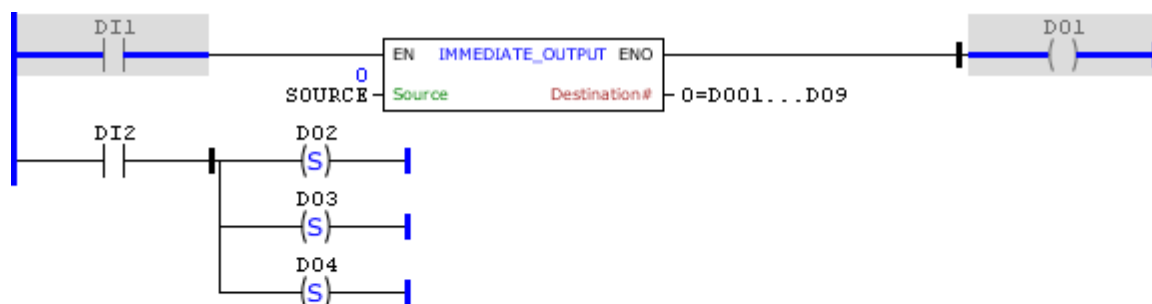
Fluxograma do Bloco



Exemplo



● SOURCE	WORD	0	Binary
● DO1	BOOL	0	Binary
● DO2	BOOL	1	Binary
● DO3	BOOL	1	Binary
● DO4	BOOL	1	Binary
● DO5	BOOL	0	Binary
● DO6	BOOL	0	Binary
● DO7	BOOL	0	Binary
● DO8	BOOL	0	Binary
● DO9	BOOL	0	Binary



● SOURCE	WORD	0	Binary
● DO1	BOOL	1	Binary
● DO2	BOOL	0	Binary
● DO3	BOOL	0	Binary
● DO4	BOOL	0	Binary
● DO5	BOOL	0	Binary
● DO6	BOOL	0	Binary
● DO7	BOOL	0	Binary
● DO8	BOOL	0	Binary
● DO9	BOOL	0	Binary

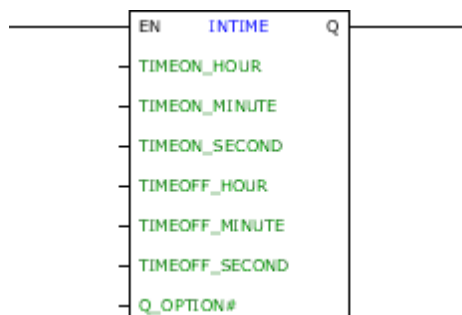
O exemplo acima faz uma escrita imediata do valor de SOURCE, interpretada como uma sequência de binários, nas saídas digitais DO1 a DO9 do PLC300, com DO1 recebendo o valor do bit menos significativo. Finalizado o bloco com sucesso, a saída ENO é ativada. Observe que a escrita imediata não prevalece sobre a bobina direta DO1 ou sobre as bobinas de habilitação em DO2, DO3 e DO4.

10.1.5.1.20 RTC

10.1.5.1.20.1 INTIME

Bloco que realiza uma habilitação programada para um horário baseado em RTC (Real Time Clock).

Representação Ladder



Estrutura do Bloco

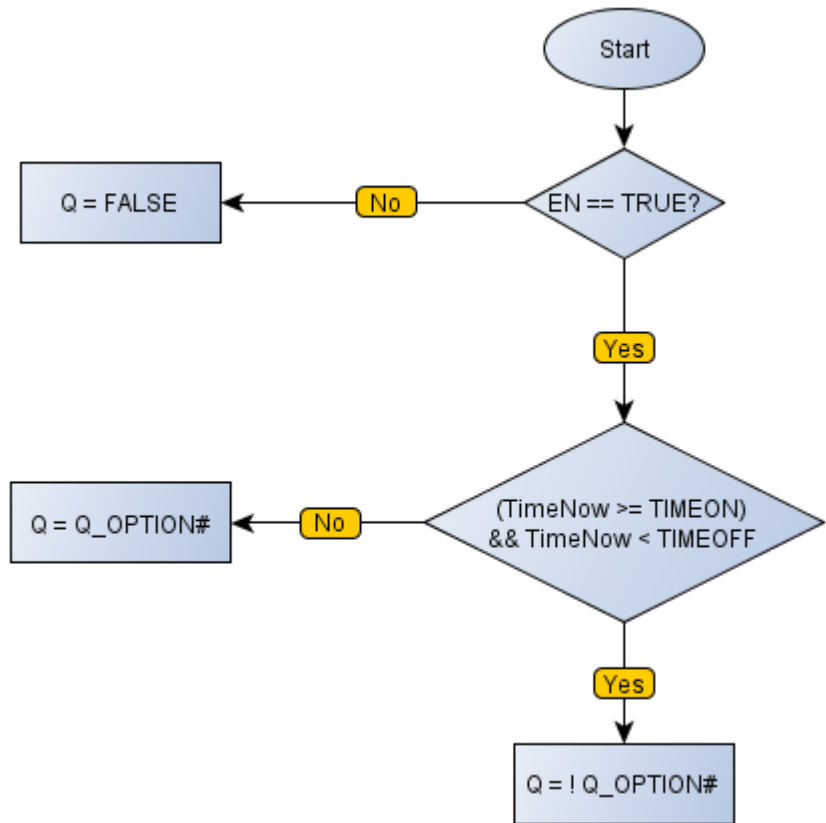
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	TIMEON_HOUR	WORD UINT	Hora de habilitação
	TIMEON_MINUTE	WORD UINT	Minutos de habilitação
	TIMEON_SECOND	WORD UINT	Segundo de habilitação
	TIMEOFF_HOUR	WORD UINT	Hora de desabilitação
	TIMEOFF_MINUTE	WORD UINT	Minutos de desabilitação
	TIMEOFF_SECOND	WORD UINT	Segundo de desabilitação
	Q_OPTION#	BYTE	Funcionamento da saída
VAR_OUTPUT	Q	BOOL	Saída do bloco

Funcionamento

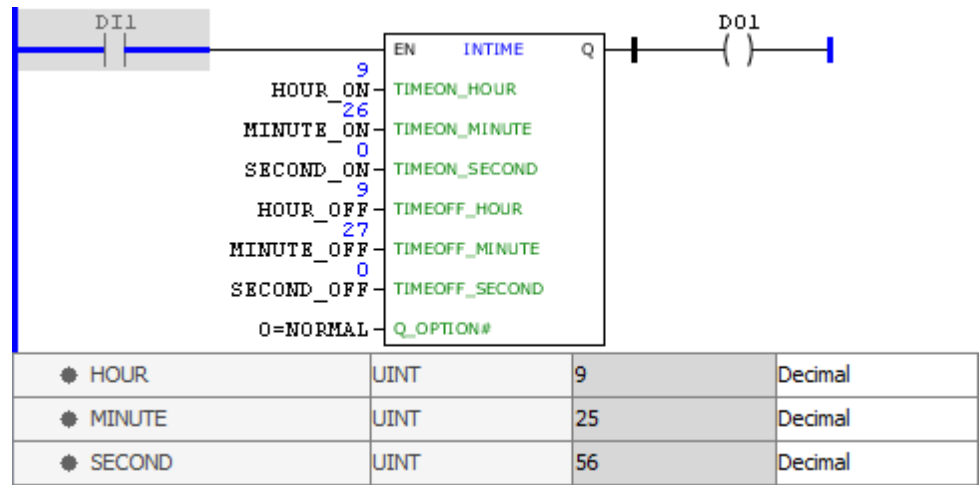
Este bloco, quando tem um valor TRUE em EN, possui dois modos de operação. Se Q_OPTION# for Normal, Q é habilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEON e desabilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEOFF. Se Q_OPTION# for Inverted, Q é desabilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEON e habilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEOFF.

Quando EN possui valor FALSE, Q permanece FALSE.

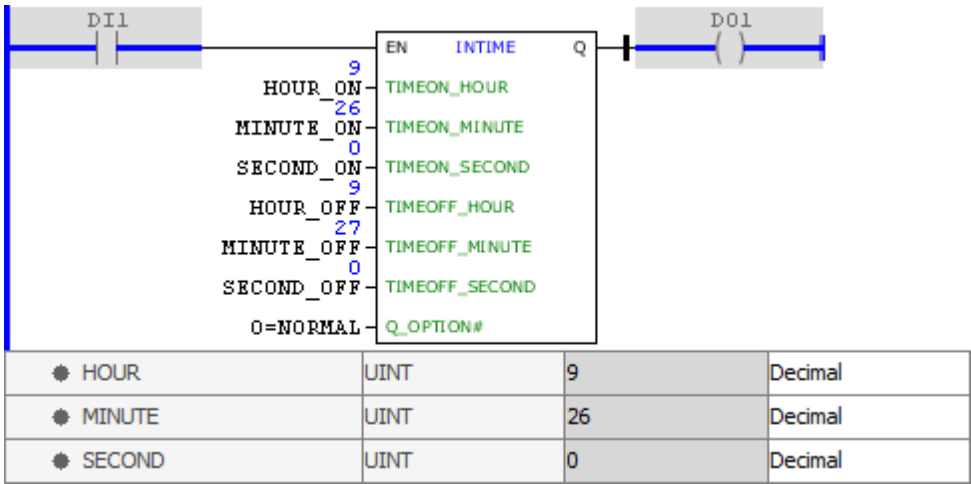
Fluxograma do Bloco



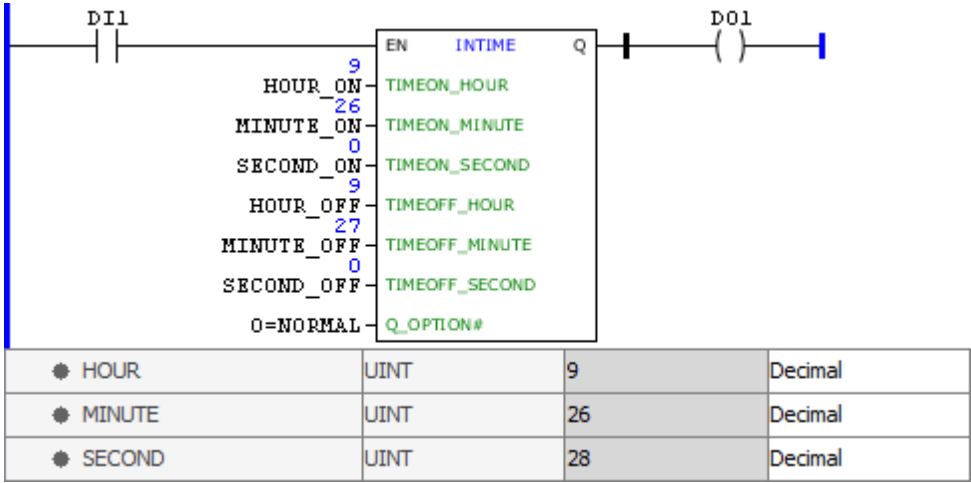
Exemplo



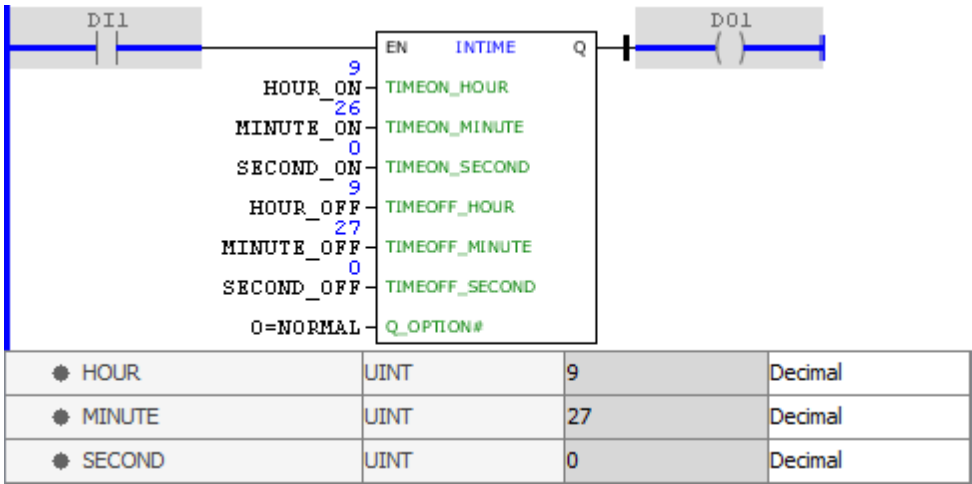
No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é inferior ao cadastrado nas entradas de habilitação do bloco (HOUR_ON, MINUTE_ON e SECOND_ON). Dessa forma, a saída Q é desabilitada.



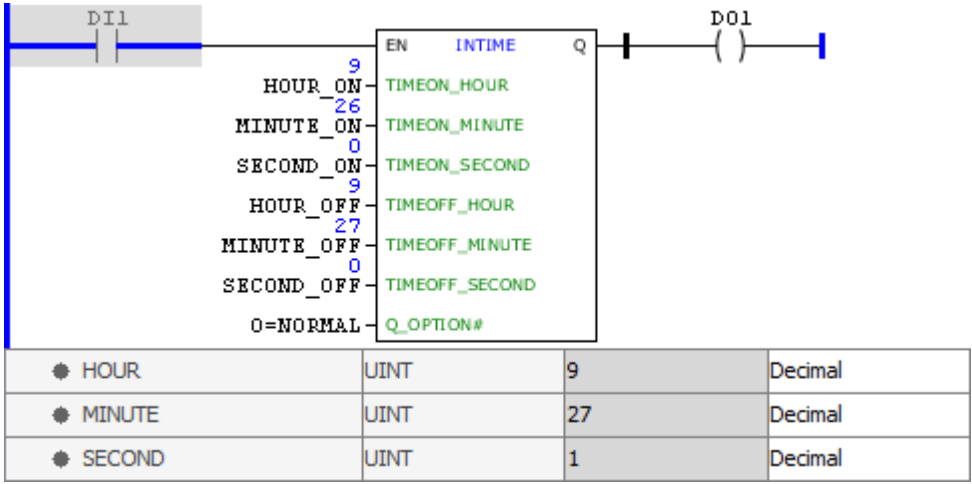
No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é igual ao cadastrado nas entradas de habilitação do bloco (HOUR_ON, MINUTE_ON e SECOND_ON). Dessa forma, a saída Q é habilitada.



No exemplo acima, o bloco INTIME está desabilitado. Dessa forma, independente das entradas, a saída Q é desabilitada.



No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é igual ao cadastrado nas entradas de desabilitação do bloco (HOUR_OFF, MINUTE_OFF e SECOND_OFF). Dessa forma, a saída Q é habilitada.

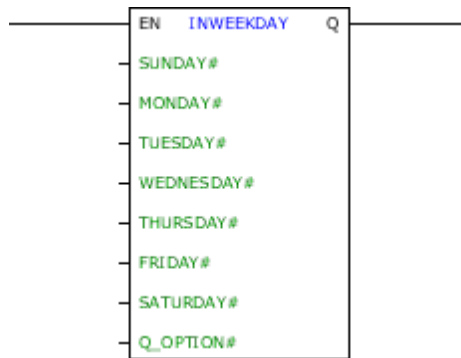


No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é superior ao cadastrado nas entradas de desabilitação do bloco (HOUR_OFF, MINUTE_OFF e SECOND_OFF). Dessa forma, a saída Q é desabilitada.

10.1.5.1.20.2 INWEEKDAY

Bloco que realiza uma habilitação programada para dias da semana baseado em RTC (Real Time Clock).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SUNDAY#	BOOL	Habilitado no domingo
	MONDAY#	BOOL	Habilitado na segunda-feira
	TUESDAY#	BOOL	Habilitado na terça-feira
	WEDNESDAY#	BOOL	Habilitado na quarta-feira
	THURSDAY#	BOOL	Habilitado na quinta-feira
	FRIDAY#	BOOL	Habilitado na sexta-feira
	SATURDAY#	BOOL	Habilitado no sábado
	Q_OPTION#	BYTE	Funcionamento da saída
VAR_OUTPUT	Q	BOOL	Saída do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, possui dois modos de operação. Se Q_OPTION# for Normal, Q é habilitada se o dia da semana do relógio interno possuir parâmetro Enabled no bloco. Se Q_OPTION# for Inverted, Q é desabilitada se o dia da semana do relógio interno possuir parâmetro Enabled no bloco.

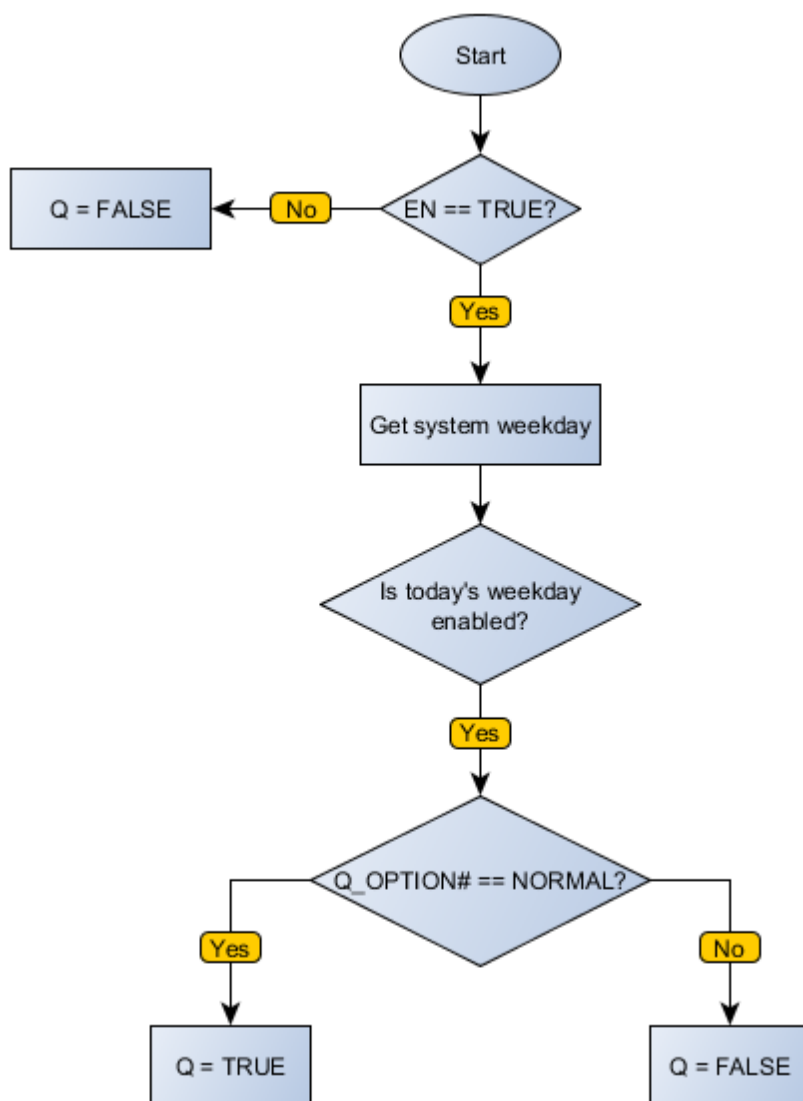
Quando EN possui valor FALSE, Q permanece FALSE.



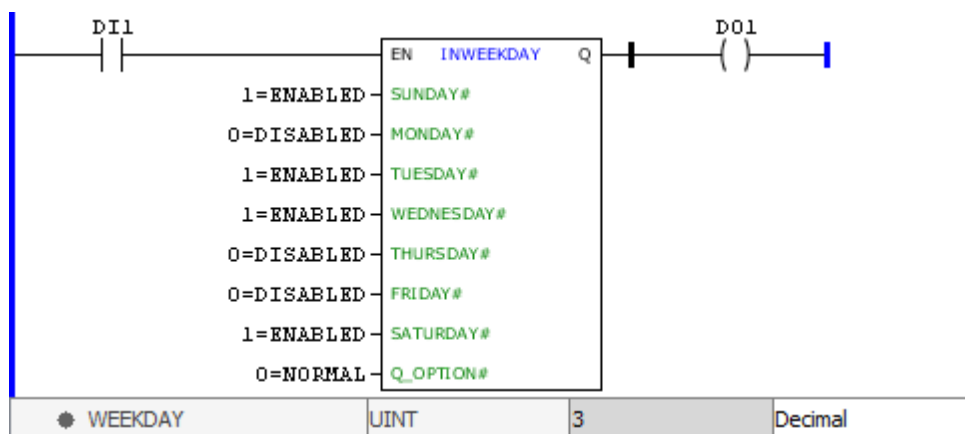
NOTA!

Os dias da semana são identificados por números, sendo Domingo o dia 0 e Sábado o dia 6.

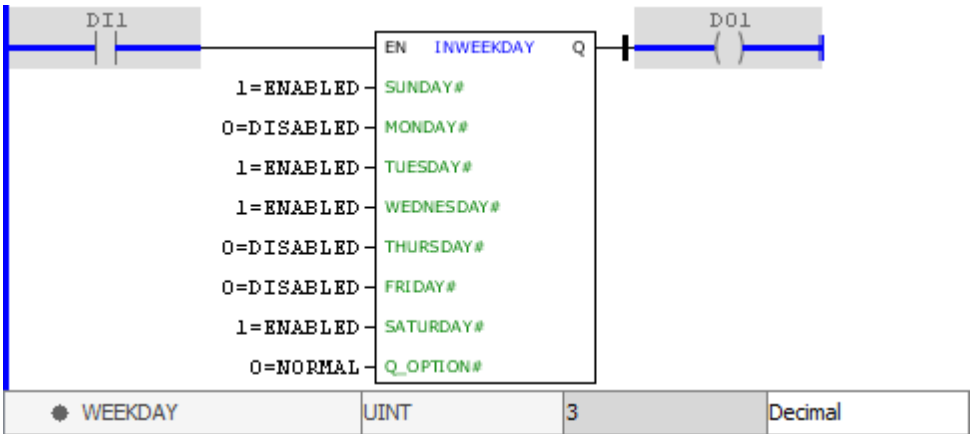
Fluxograma do Bloco



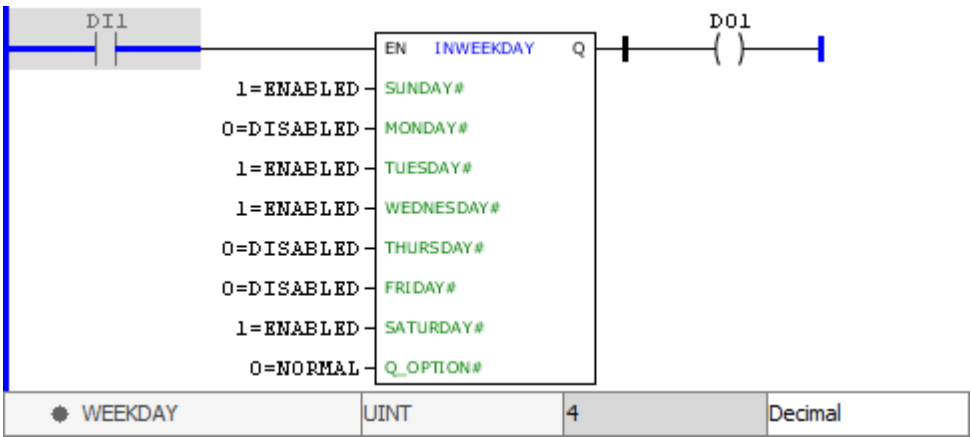
Exemplo



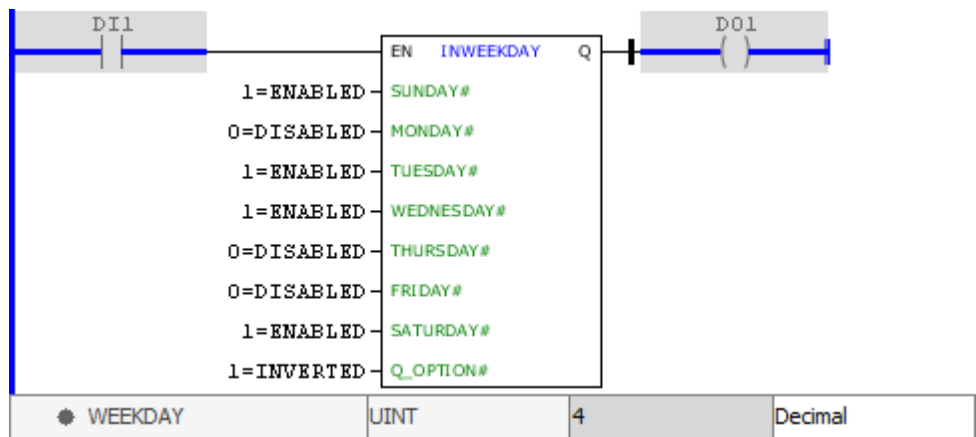
No exemplo acima, o bloco INWEEKDAY está desabilitado. Dessa forma, independente das entradas, a saída Q é desabilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação NORMAL. O dia da semana atual do relógio interno do equipamento é quarta-feira (valor 3), que possui estado ENABLED na programação. Dessa forma, a saída Q é habilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação NORMAL. O dia da semana atual do relógio interno do equipamento é quinta-feira (valor 4), que possui estado DISABLED na programação. Dessa forma, a saída Q é desabilitada.



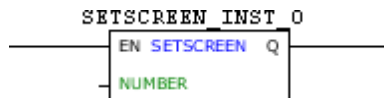
No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação INVERTED. O dia da semana atual do relógio interno do equipamento é quinta-feira (valor 4), que possui estado DISABLED na programação. Dessa forma, a saída Q é habilitada.

10.1.5.1.21 Screen

10.1.5.1.21.1 SETSCREEN

Bloco que exibe uma determinada tela na IHM.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	NUMBER	BYTE UINT USINT WORD	Número da tela a ser exibida
VAR_OUTPUT	Q	BOOL	Saída do bloco
VAR	SETSCREEN_INST_0	SETSCREEN	Instância de acesso à estrutura do bloco

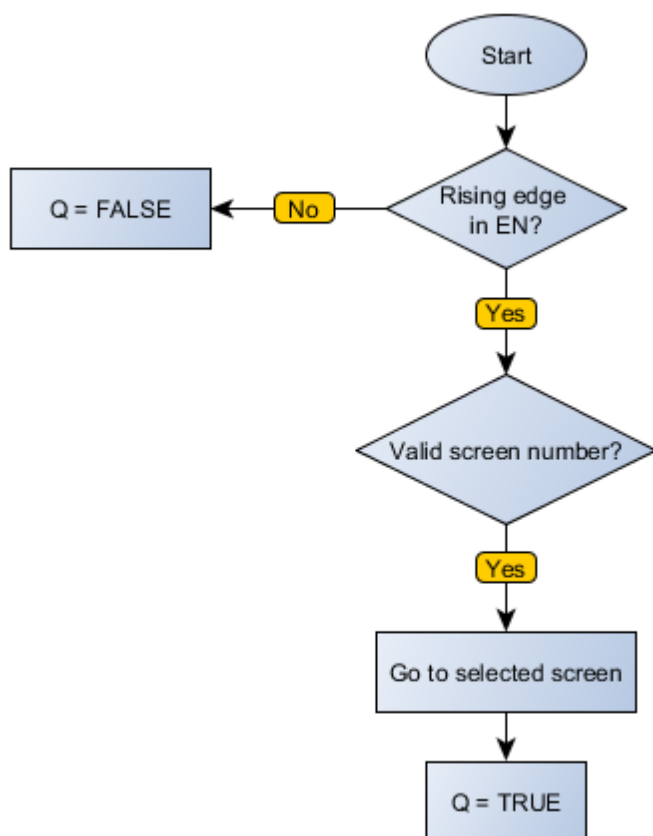
Funcionamento

Este bloco, quando detecta uma borda de subida em EN, exibe a tela representada por NUMBER na IHM.

Q recebe TRUE por um ciclo de scan, caso o número da tela seja válido.

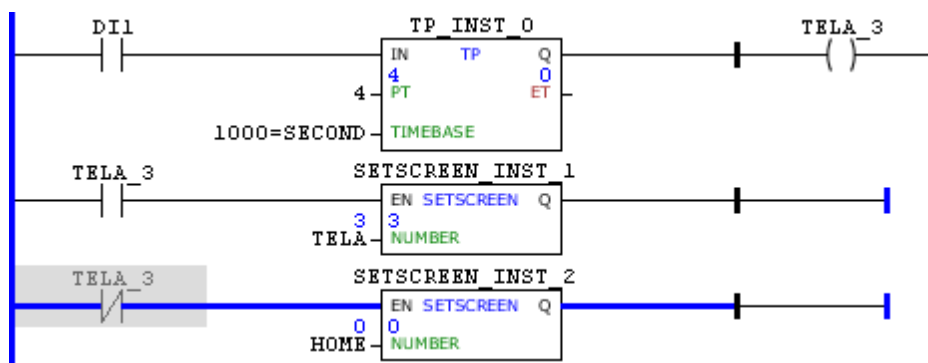
Quando EN possui valor FALSE, Q permanece FALSE.

Fluxograma do Bloco

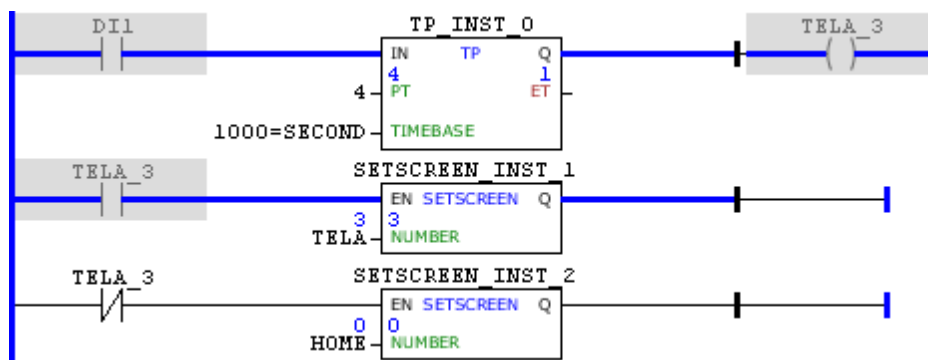


Exemplo

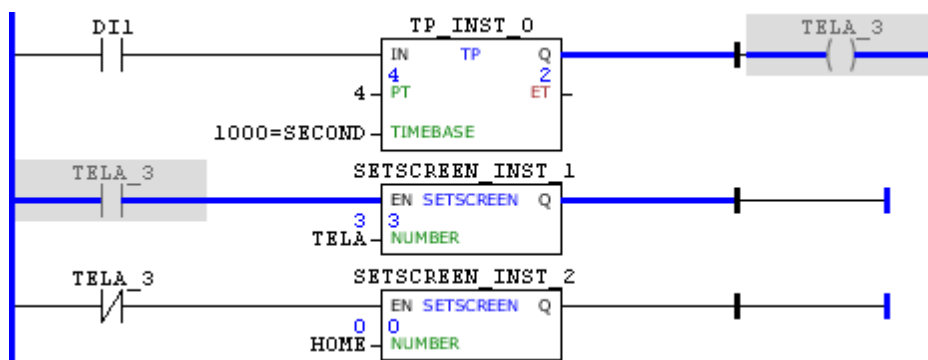
O exemplo a seguir habilita a tela 3 por 4 segundos a cada pulso em DI1.



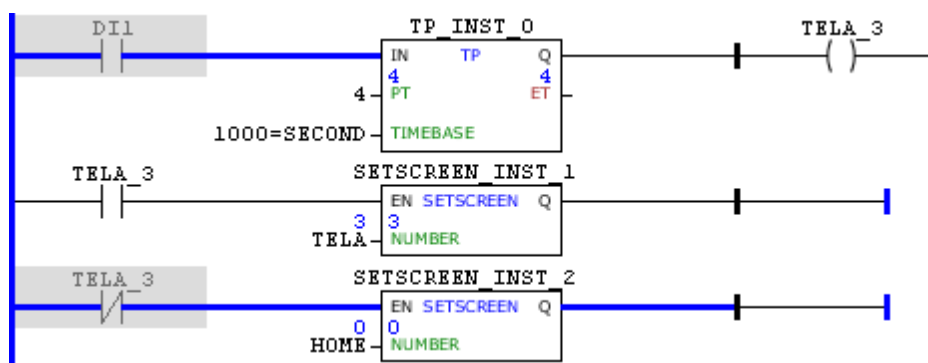
No início, considera-se TELA_3 com valor FALSE, ativando a entrada do bloco SETSCREEN_INST_0, que exibe a tela HOME.



Ao identificar um pulso em DI1, TELA_3 recebe valor TRUE, ativando a entrada do bloco SETSCREEN_INST_1, que exibe a tela 3.



Mesmo com o sinal retirado de DI1, TELA_3 continua com o valor TRUE, ou seja, a tela 3 continua sendo mostrada.



Passados os quatro segundos, caso DI1 ainda seja TRUE, a tela HOME ainda assim é exibida. Só será exibida a tela 3 quando houver uma nova borda de subida em DI1.

10.1.5.1.22 Communication Network

10.1.5.1.22.1 CANopen

Operação na Rede CANopen - Modo Mestre

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

rede CANopen.

Habilitação da Função CANopen Mestre

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

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



NOTA!

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

Características do Mestre CANopen

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

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

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



NOTA!

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

Operação do Mestre

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

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

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

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



NOTA!

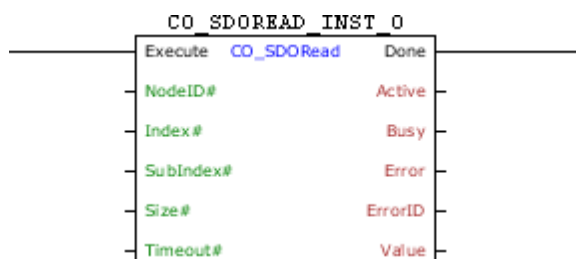
O estado da comunicação e o estado de cada escravo podem ser observados em marcadores de sistema de entrada.

Blocos para o Mestre CANopen

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

Bloco que executa uma leitura de dados via SDO de um escravo remoto em rede CANopen.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NodeID#	BYTE	Endereço do escravo
	Index#	WORD	Índice do objeto a ser acessado no escravo
	SubIndex#	BYTE	Sub-índice do objeto
	Size#	BYTE	Tamanho do dado acessado, em bytes
	Timeout#	WORD	Tempo de espera máximo para chegada do dado, a partir do início da solicitação [ms]
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que o cliente SDO está ocupado com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE USINT	Identificador do erro ocorrido
	Value	BYTE USINT	Variável que armazena os dados recebidos
VAR	CO_SDOREAD_INST_0	CO_SDOREAD	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o cliente SDO no endereço especificado em NodeID# está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de leitura para o objeto de tamanho Size# localizado em Index# e SubIndex# e seta a saída Active, resetando-a ao receber a resposta do escravo. Os dados recebidos são armazenados na variável Value. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.



NOTA!

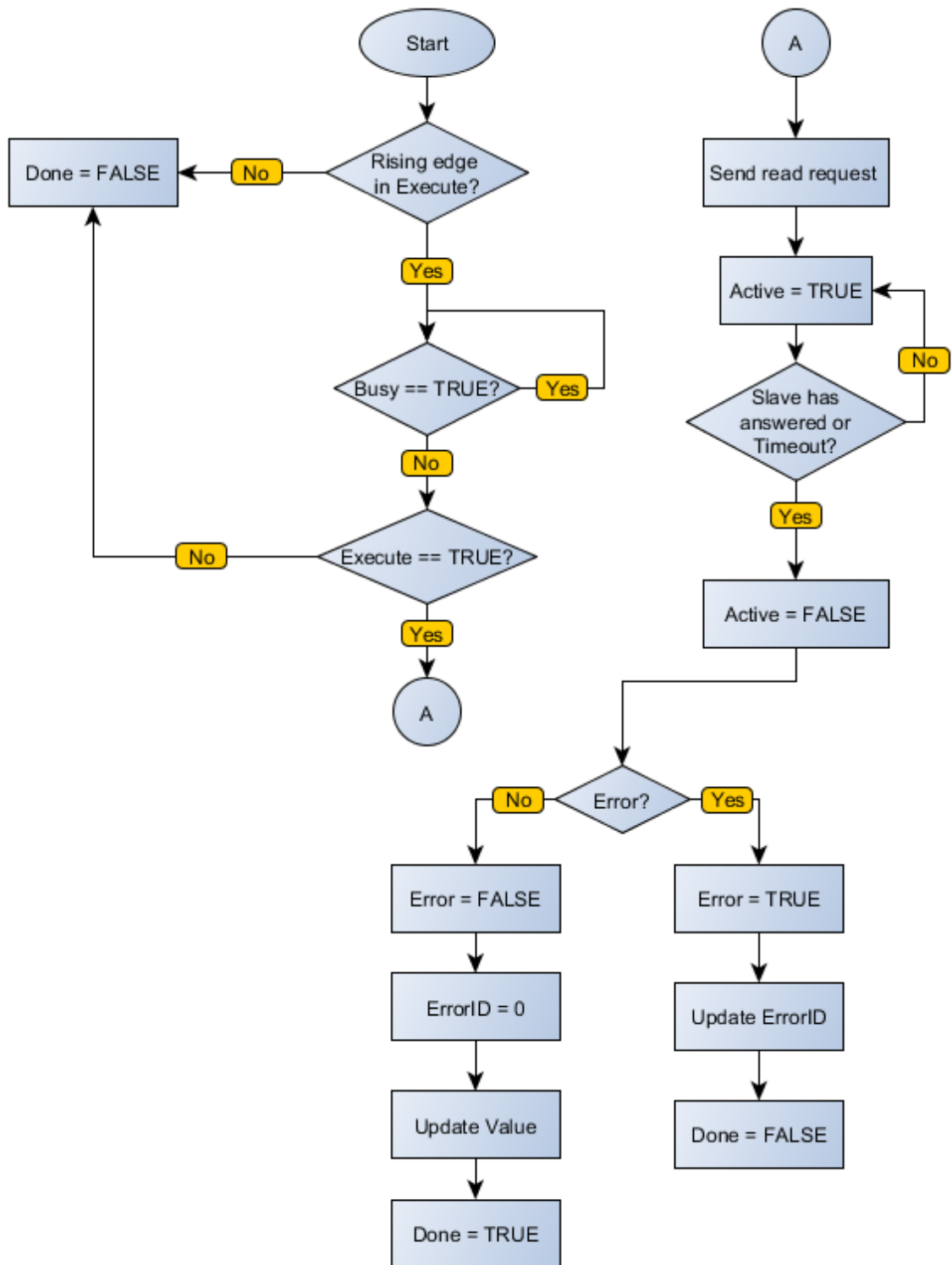
Value é um array de tamanho igual a Size#. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

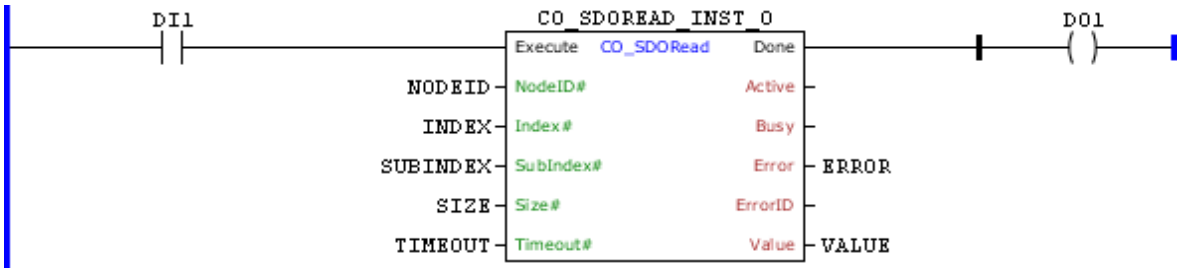
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Cartão não pode executar a função
2	Timeout na resposta do escravo
3	Escravo retornou erro

Fluxograma do Bloco



Exemplo



O exemplo acima requisita leitura do dado de tamanho SIZE, localizado em INDEX - SUBINDEX, do equipamento NODEID. Esse dado é repassado a VALUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma escrita de dados via SDO de um escravo remoto em rede CANopen.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NodeID#	BYTE	Endereço do escravo
	Index#	WORD	Índice do objeto a ser acessado no escravo
	SubIndex#	BYTE	Sub-índice do objeto
	Size#	BYTE	Tamanho do dado acessado, em bytes
	Timeout#	WORD	Tempo de espera máximo para chegada do dado, a partir do início da solicitação [ms]
	Value	BYTE USINT	Variável que possui os dados a serem escritos
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que o cliente SDO está ocupado com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE USINT	Identificador do erro ocorrido
VAR	CO_SDOWRITE_INST_0	CO_SDOWRITE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o cliente SDO no endereço especificado em NodeID# está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de escrita para o objeto de tamanho Size# localizado em Index# e SubIndex# e seta a saída Active, resetando-a ao receber a resposta do escravo. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.

**NOTA!**

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.

**NOTA!**

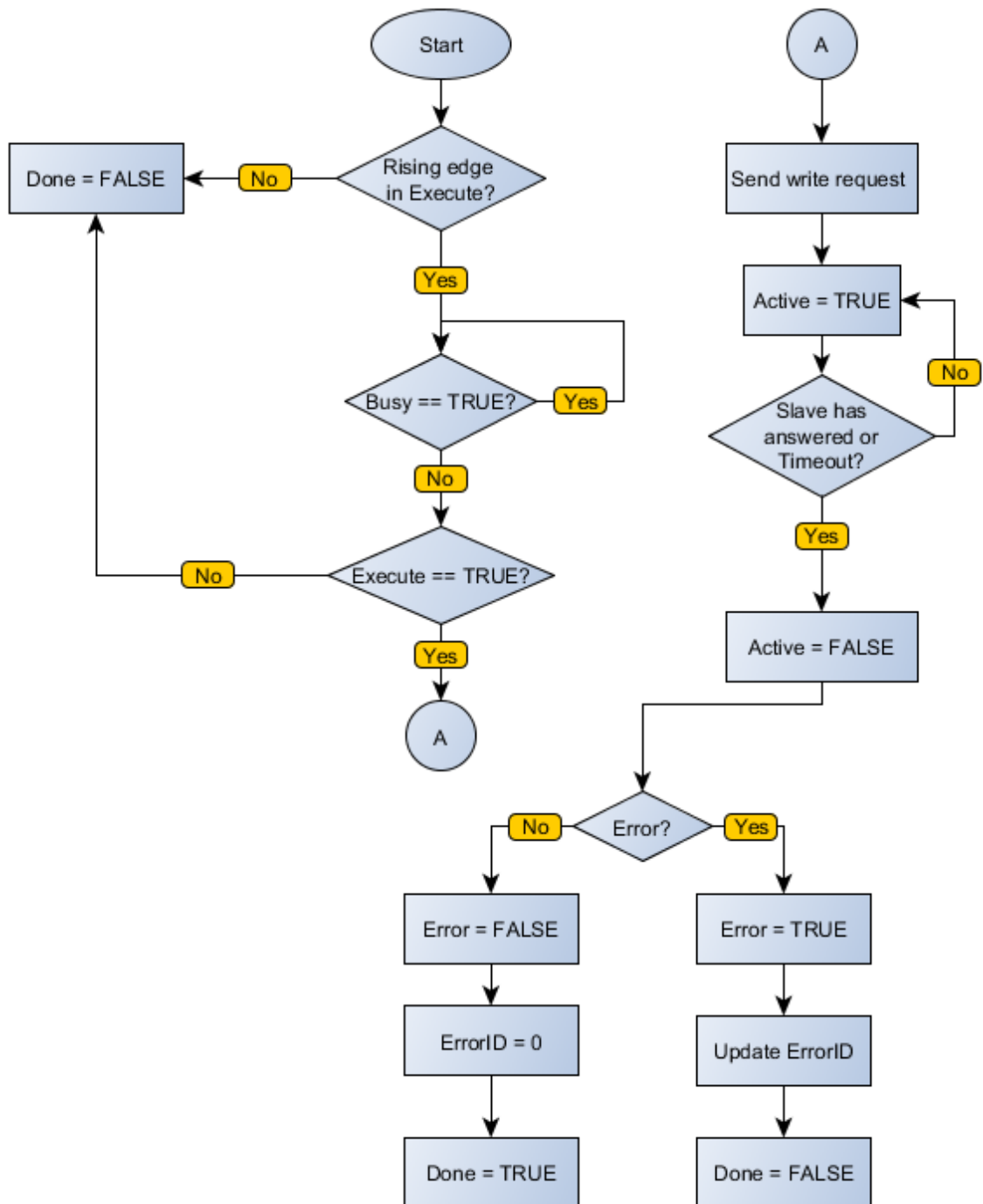
Value é um array de tamanho igual a Size#. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

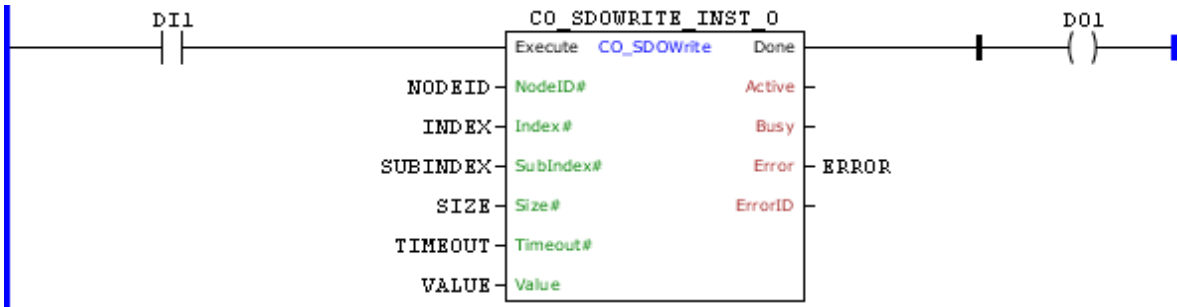
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Cartão não pode executar a função
2	Timeout na resposta do escravo
3	Escravo retornou erro

Fluxograma do Bloco



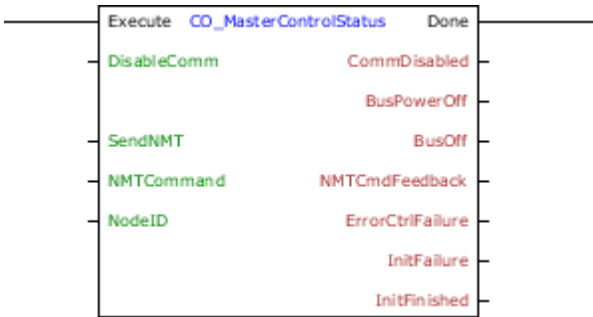
Exemplo



O exemplo acima requisita escrita do dado contido em VALUE, de tamanho SIZE, no local INDEX - SUBINDEX do equipamento NODEID. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que permite monitorar diversos estados do mestre da rede CANopen.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	B00L	Habilitação do bloco
	DisableComm	B00L	Desabilita a comunicação CANopen
	SendNMT	B00L	Dispara envio de comando de gerenciamento
	NMTCommand	BYTE	Código do comando de gerenciamento a ser enviado
	NodeID	BYTE USINT	Endereço do escravo para envio do NMT
VAR_OUTPUT	Done	B00L	Habilitação de saída
	CommDisabled	B00L	Sinalizador de comunicação desabilitada
	BusPowerOff	B00L	Sinalizador de falha na alimentação da interface CAN
	BusOff	B00L	Sinalizador de erro de BusOff na interface CAN
	NMTCmdFeedback	B00L	Sinalizador de comando NMT enviado pelo mestre
	ErrorCtrlFailure	B00L	Sinalizador de erro de nodeguarding ou heartbeat em um dos escravos da rede
	InitFailure	B00L	Sinalizador de erro na inicialização de um dos escravos da rede
	InitFinished	B00L	Sinalizador de que o procedimento de inicialização dos escravos foi concluído

Funcionamento

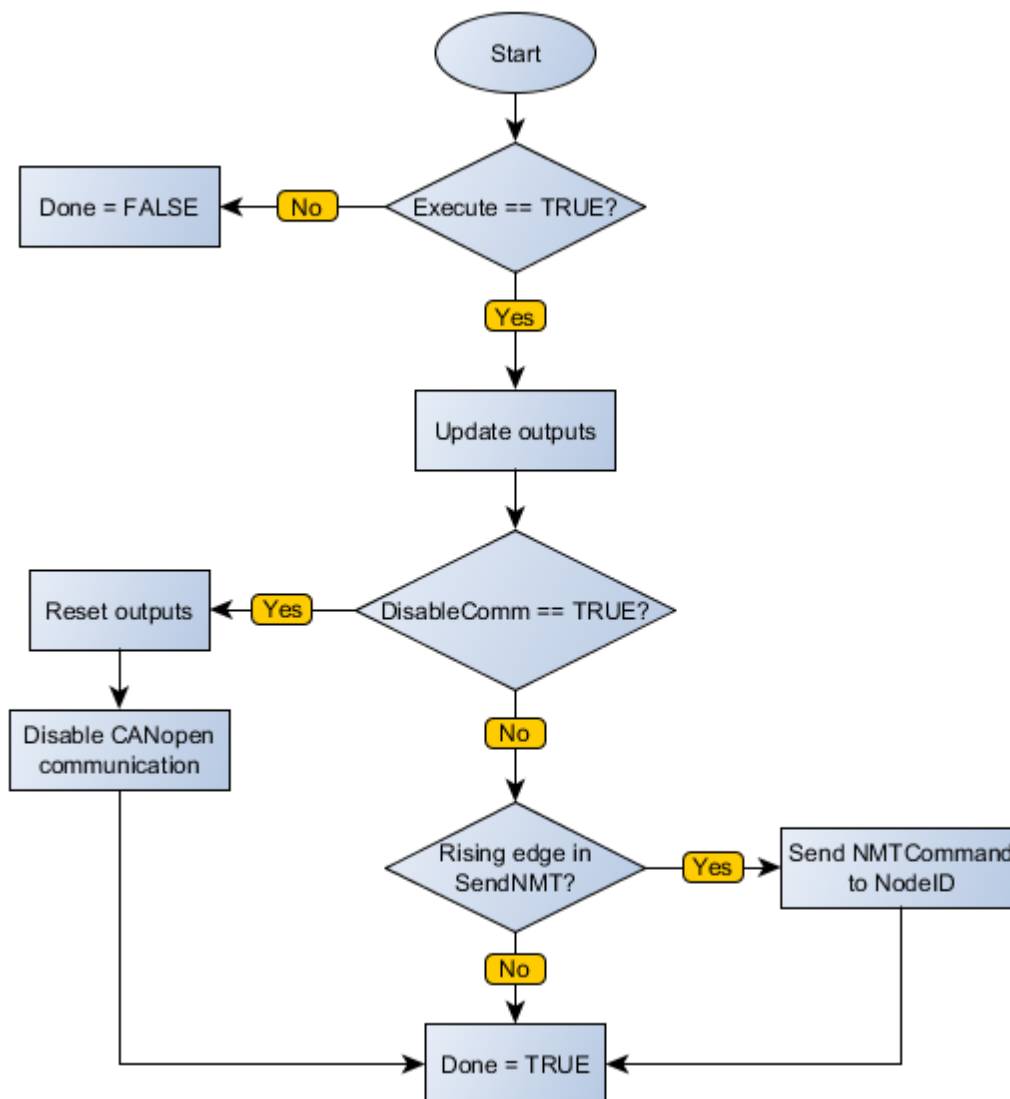
Este bloco permanece ativo enquanto Execute está em nível TRUE, atualizando suas saídas de acordo com a monitoração do mestre e com as requisições de entrada. Quando Execute recebe nível FALSE, as entradas são ignoradas e as saídas são zeradas. A saída Done recebe nível TRUE quando Execute possui nível TRUE e o bloco finalizou sua execução.

Um nível TRUE em DisableComm desabilita a comunicação CANopen e zera os contadores e marcadores de status do mestre.

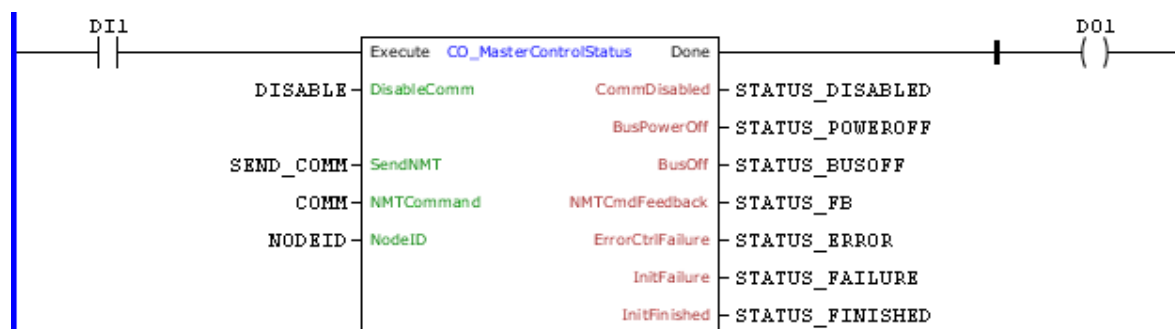
Uma borda de subida em SendNMT envia um comando de gerenciamento (NMT) indicado em NMTCommand ao escravo de endereço NodeID.

As saídas recebem nível TRUE quando o status do mestre CANopen corresponde à sua descrição (comunicação desabilitada, falha na alimentação, etc).

Fluxograma do Bloco



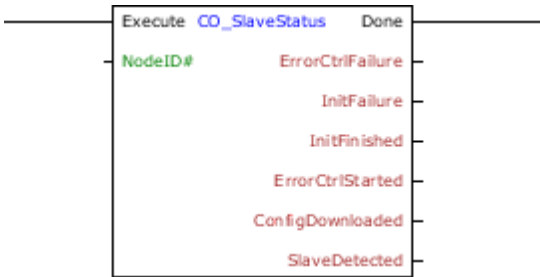
Exemplo



O exemplo acima requisita dados de status do mestre da rede CANopen, além de permitir desabilitar a comunicação através de DISABLE e enviar comandos NMT para NODEID através de SEND_COMM. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que permite monitorar diversos estados de um determinado escravo da rede CANopen.

Representação Ladder



Estrutura do Bloco

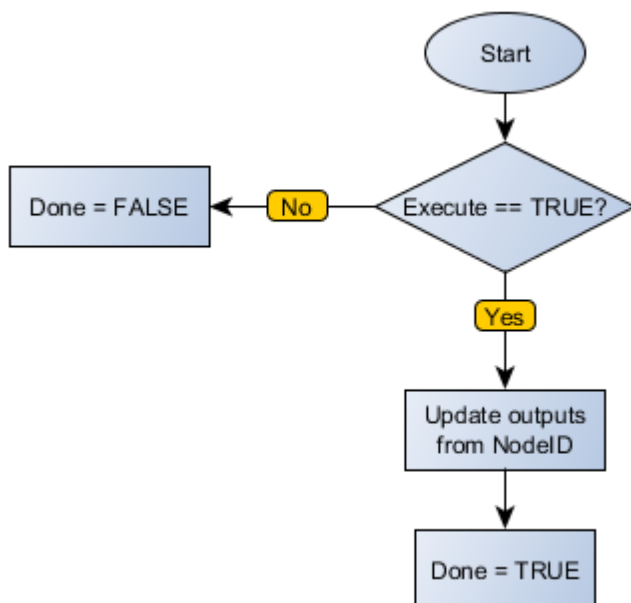
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	B00L	Habilitação do bloco
	NodeID	BYTE USINT	Endereço do escravo para monitoração
VAR_OUTPUT	Done	B00L	Habilitação de saída
	ErrorCtrlFailure	B00L	Sinalizador de erro de nodeguarding ou heartbeat no escravo indicado
	InitFailure	B00L	Sinalizador de erro na inicialização no escravo indicado
	InitFinished	B00L	Sinalizador de que o procedimento de inicialização do escravo foi concluído
	ErrorCtrlStarted	B00L	Sinalizador de início do serviço de controle de erros
	ConfigDownloaded	B00L	Sinalizador de sucesso no dow nload das configurações SDO no escravo
	SlaveDetected	B00L	Sinalizador de sucesso na leitura de identificação do escravo via SDO

Funcionamento

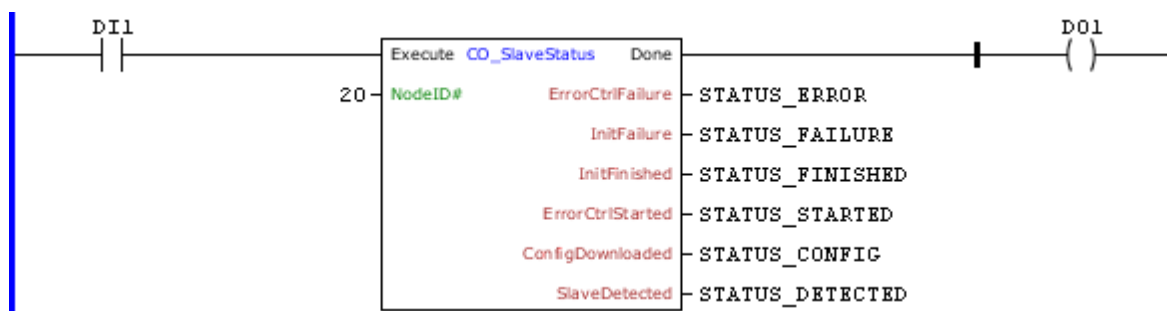
Este bloco permanece ativo enquanto Execute está em nível TRUE, atualizando suas saídas de acordo com a monitoração do escravo. Quando Execute recebe nível FALSE, as entradas são ignoradas e as saídas são zeradas. A saída Done recebe nível TRUE quando Execute possui nível TRUE e o bloco finalizou sua execução.

As saídas recebem nível TRUE quando o status do escravo CANopen indicado pelo endereço NodeID corresponde à sua descrição (erro de inicialização, sucesso de *download*, etc).

Fluxograma do Bloco



Exemplo



O exemplo acima requisita dados do escravo 20 da rede CANopen. Finalizado o bloco com sucesso, a saída Done é ativada.

10.1.5.1.22.2 Modbus RTU

Operação na Rede Modbus RTU - Modo mestre

Além da operação como escravo, o controlador programável PLC300 também permite a operação como mestre da rede Modbus RTU. Para esta operação, é necessário observar os seguintes pontos:

- Somente a interface RS485 permite operação como mestre da rede.
- É necessário programar, nas configurações do produto, o modo de operação como Mestre, além da taxa de comunicação, paridade e stop bits, que devem ser as mesmas para todos os equipamentos da rede.
- O mestre da rede Modbus RTU não possui endereço, logo o endereço configurado no PLC300 não é utilizado.
- O envio e recepção de telegramas via interface RS485 utilizando o protocolo Modbus RTU é programado utilizando blocos em linguagem de programação Ladder. É necessário conhecer os blocos disponíveis e o *software* de programação em Ladder para poder programar o mestre da rede.
- As seguintes funções estão disponíveis para envio de requisições pelo mestre Modbus:

- o Função 01: Read Coils
- o Função 02: Read Discrete Inputs
- o Função 03: Read Holding Registers
- o Função 04: Read Input Registers
- o Função 05: Write Single Coil
- o Função 06: Write Single Register
- o Função 15: Write Multiple Coils
- o Função 16: Write Multiple Registers

Blocos para a programação do mestre

Para o controle e monitoração da comunicação Modbus RTU utilizando o controlador programável PLC300, foram desenvolvidos os seguintes blocos, que devem ser utilizados durante a programação em Ladder.

Bloco que executa uma leitura de até 128 dados binários (via Read Coils ou Read Discrete Inputs) de um escravo em rede Modbus RTU.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	SlaveAddress	BYTE	Endereço do escravo
	Function#	BYTE	Código da função de leitura
	InitialDataAddress	WORD	Endereço do bit inicial dos dados a serem lidos
	NumberOfData	BYTE	Número de bits a serem lidos (1 a 128)
	Timeout#	WORD	Tempo máximo de espera pela resposta do escravo [ms]
	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a interface RS485 está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE	Identificador do erro ocorrido
	Value	BOOL	Variável que armazena os dados recebidos
VAR	MB_READBINARY_INST_0	MB_READBINARY	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o escravo Modbus RTU no endereço especificado em SlaveAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de leitura de um número de bits indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do escravo. Os dados recebidos são armazenados na variável Value. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.



NOTA!

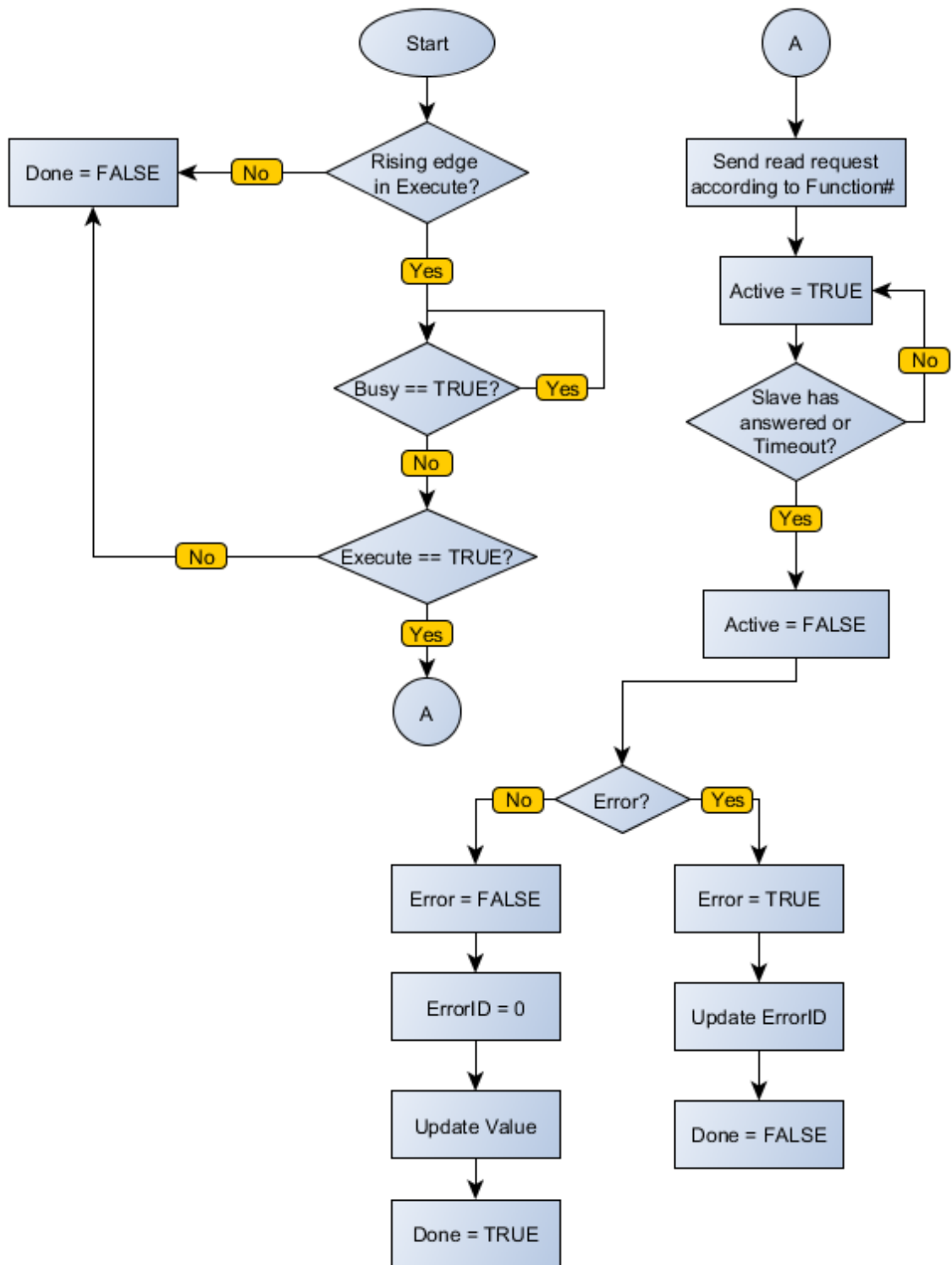
Value é um array de tamanho igual a NumberOfData. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

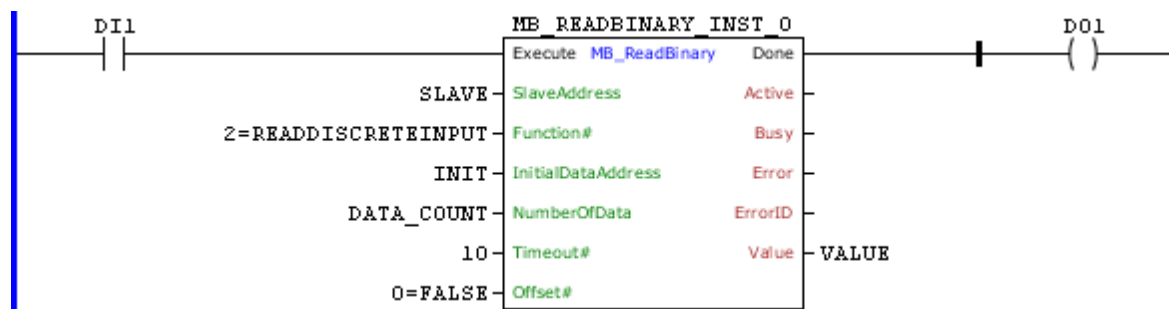
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Mestre não habilitado
4	Timeout na resposta do escravo
5	Escravo retornou erro

Fluxograma do Bloco



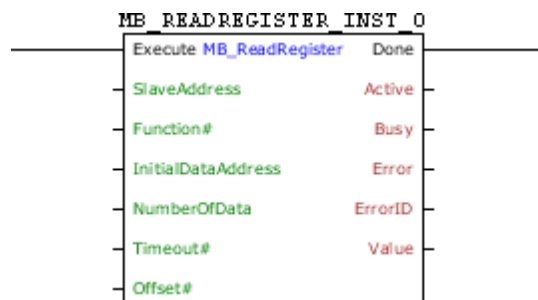
Exemplo



O exemplo acima requisita leitura de um número de dados binários descrito por DATA_COUNT, posicionados em INIT no escravo Modbus RTU de endereço SLAVE através da função Read Discrete Input. Esses dados são repassados a VALUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma leitura de até 16 registradores de 16 bits (via Read Holding Registers ou Read Input Registers) de um escravo em rede Modbus RTU.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	SlaveAddress	BYTE	Endereço do escravo
	Function#	BYTE	Código da função de leitura
	InitialDataAddress	WORD	Endereço do registrador inicial a ser lido
	NumberOfData	BYTE	Número de registradores a serem lidos (1 a 16)
	Timeout#	WORD	Tempo máximo de espera pela resposta do escravo [ms]
	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a interface RS485 está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE	Identificador do erro ocorrido
	Value	BYTE SINT USINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena os dados recebidos
VAR	MB_READREGISTER_INST_0	MB_READREGISTER	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o escravo Modbus RTU no endereço especificado em SlaveAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de leitura de um número de registradores indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do escravo. Os dados recebidos são armazenados na variável Value. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.



NOTA!

Value é um array de número de bits a NumberOfData multiplicado por 16. Ou seja, caso NumberOfData seja 16, Value pode ser um array de 32 posições BYTE, 16 posições WORD ou 8 posições DWORD. É importante verificar esta compatibilidade para não gerar erros no bloco.

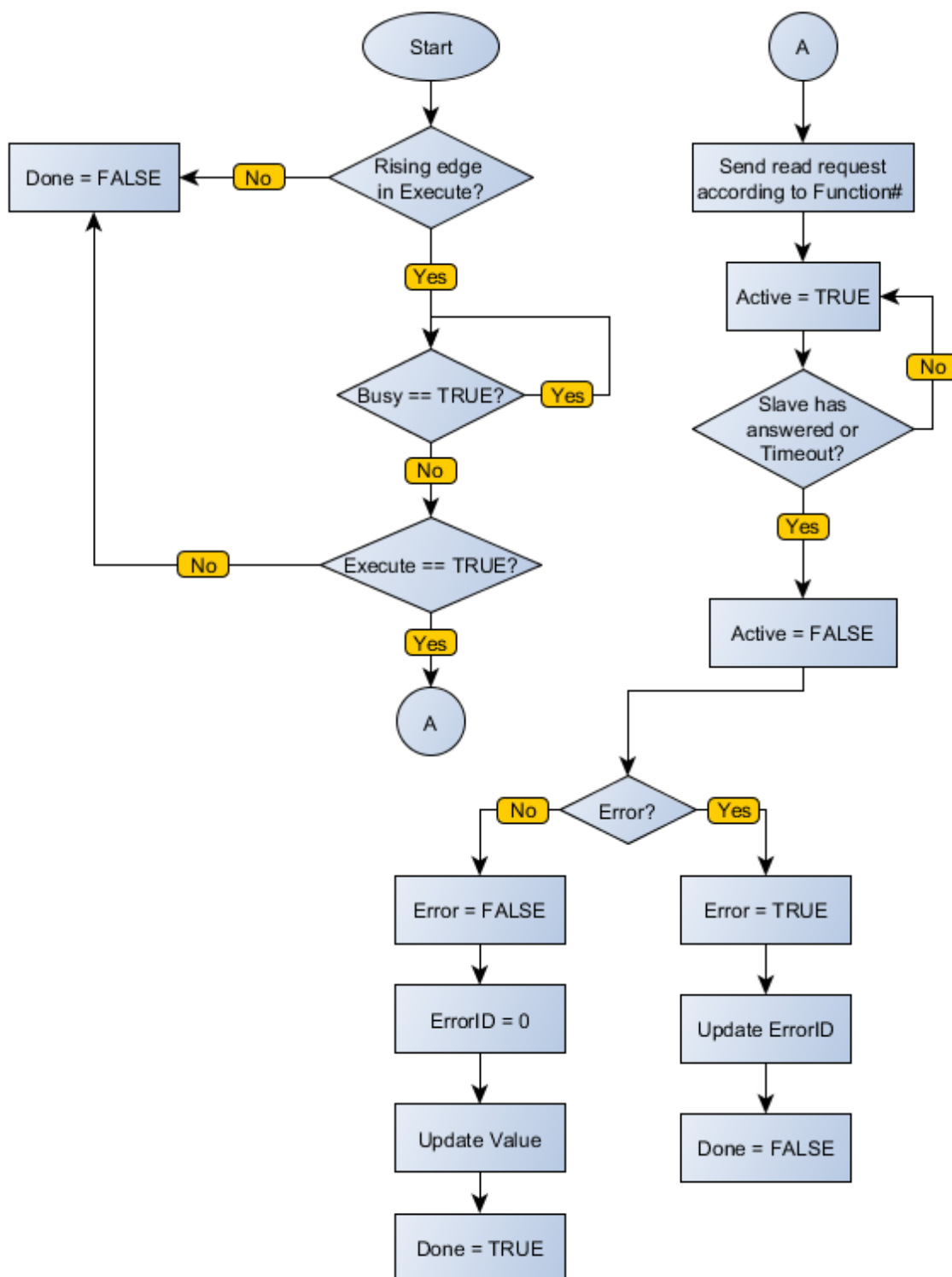
Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo

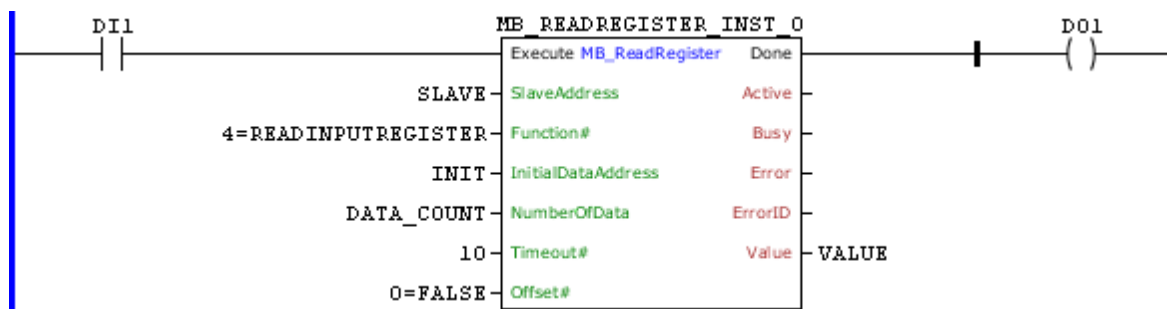
a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Mestre não habilitado
4	Timeout na resposta do escravo
5	Escravo retornou erro

Fluxograma do Bloco



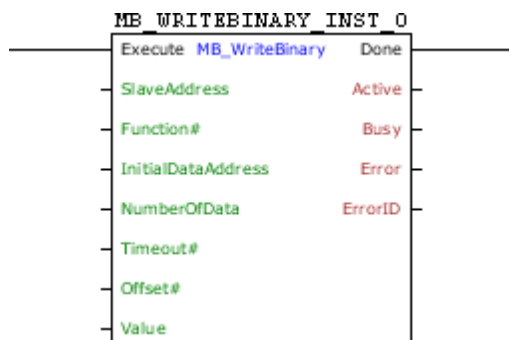
Exemplo



O exemplo acima requisita leitura de um número de dados de registradores descrito por DATA_COUNT, posicionados em INIT no escravo Modbus RTU de endereço SLAVE através da função Read Input Register. Esses dados são repassados a VALUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma escrita de até 128 dados binários (via Write Single Coil ou Write Multiple Coils) em um escravo em rede Modbus RTU.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	SlaveAddress	BYTE	Endereço do escravo
	Function#	BYTE	Código da função de escrita
	InitialDataAddress	WORD	Endereço do bit inicial onde os dados serão escritos
	NumberOfData	BYTE	Número de bits a serem escritos (1 a 128)
	Timeout#	WORD	Tempo máximo de espera pela resposta do escravo [ms]
	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
	Value	BOOL	Variável que armazena os dados a serem escritos
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a interface RS485 está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE	Identificador do erro ocorrido
VAR	MB_WRITEBINARY_INST_0	MB_WRITEBINARY	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o escravo Modbus RTU no endereço especificado em SlaveAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de escrita dos valores de Value em um número de bits indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do escravo. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.



NOTA!

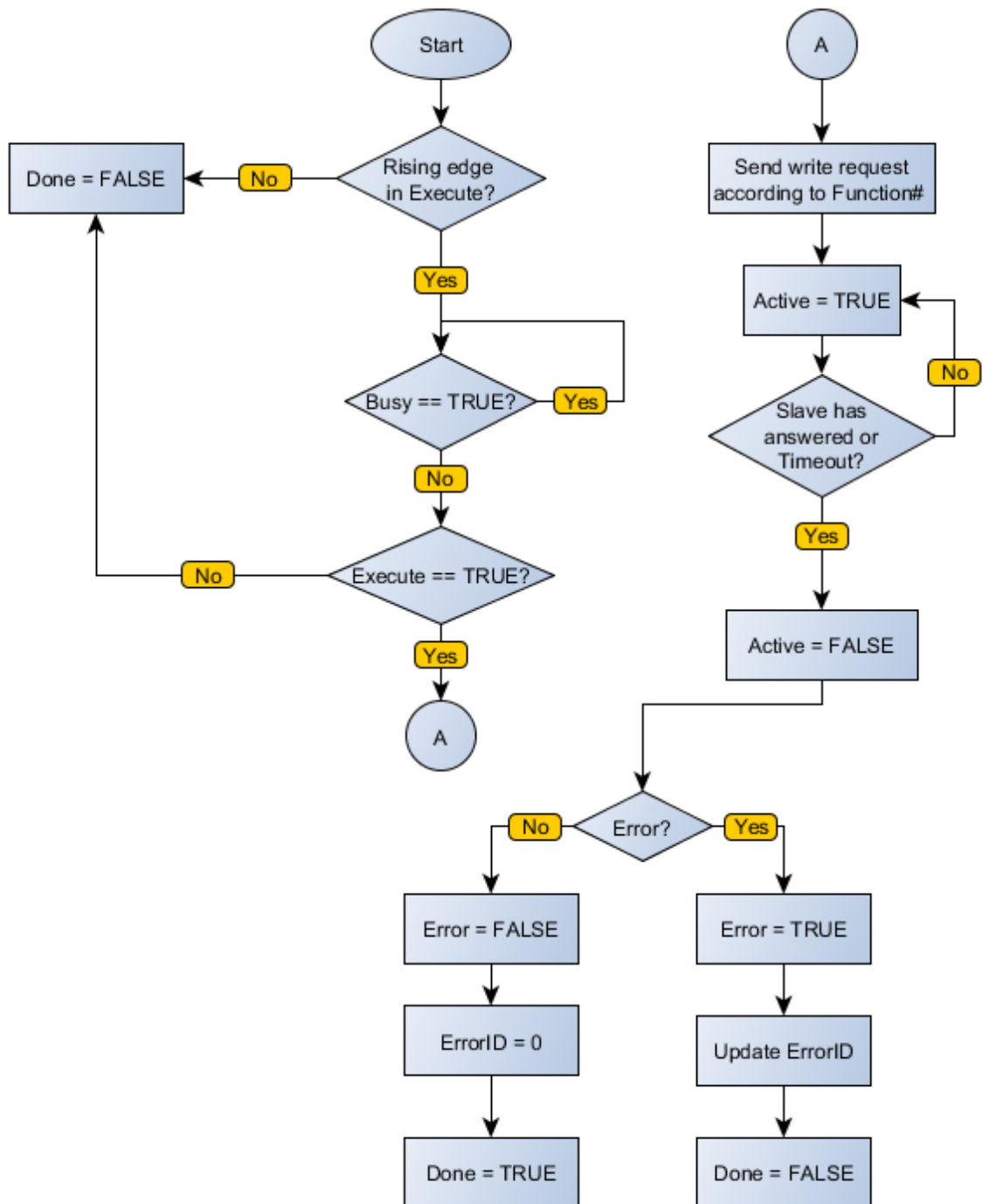
Value é um array de tamanho igual a NumberOfData. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

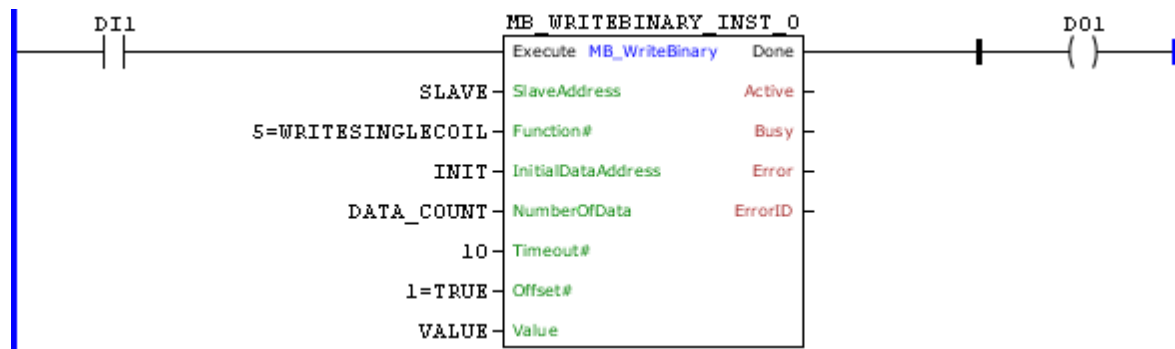
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Mestre não habilitado
4	Timeout na resposta do escravo
5	Escravo retornou erro

Fluxograma do Bloco



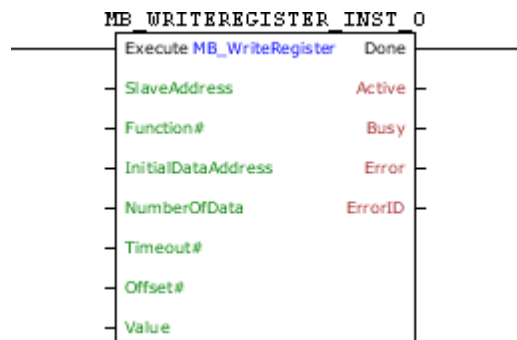
Exemplo



O exemplo acima requisita escrita do dado contido em VALUE, de tamanho descrito por DATA_COUNT, nos endereços posicionados a partir de INIT no escravo Modbus RTU de endereço SLAVE através da função Write Single Coil. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma escrita de até 16 registradores de 16 bits (via Write Single Register ou Write Multiple Registers) de um escravo em rede Modbus RTU.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	SlaveAddress	BYTE	Endereço do escravo
	Function#	BYTE	Código da função de escrita
	InitialDataAddress	WORD	Endereço do registrador inicial a ser escrito
	NumberOfData	BYTE	Número de registradores a serem escritos (1 a 16)
	Timeout#	WORD	Tempo máximo de espera pela resposta do escravo [ms]
	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
VAR_OUTPUT	Value	BYTE SINT USINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena os dados a serem escritos
	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a interface RS485 está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
VAR	ErrorID	BYTE	Identificador do erro ocorrido
	MB_WRITEREGISTER_INST_0	MB_WRITEREGISTER	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o escravo Modbus RTU no endereço especificado em SlaveAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de escrita dos valores de Value em um número de registradores indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do escravo. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.



NOTA!

Value é um array de número de bits a NumberOfData multiplicado por 16. Ou seja, caso NumberOfData seja 16, Value pode ser um array de 32 posições BYTE, 16 posições WORD ou 8 posições DWORD. É importante verificar esta compatibilidade para não gerar erros no bloco.

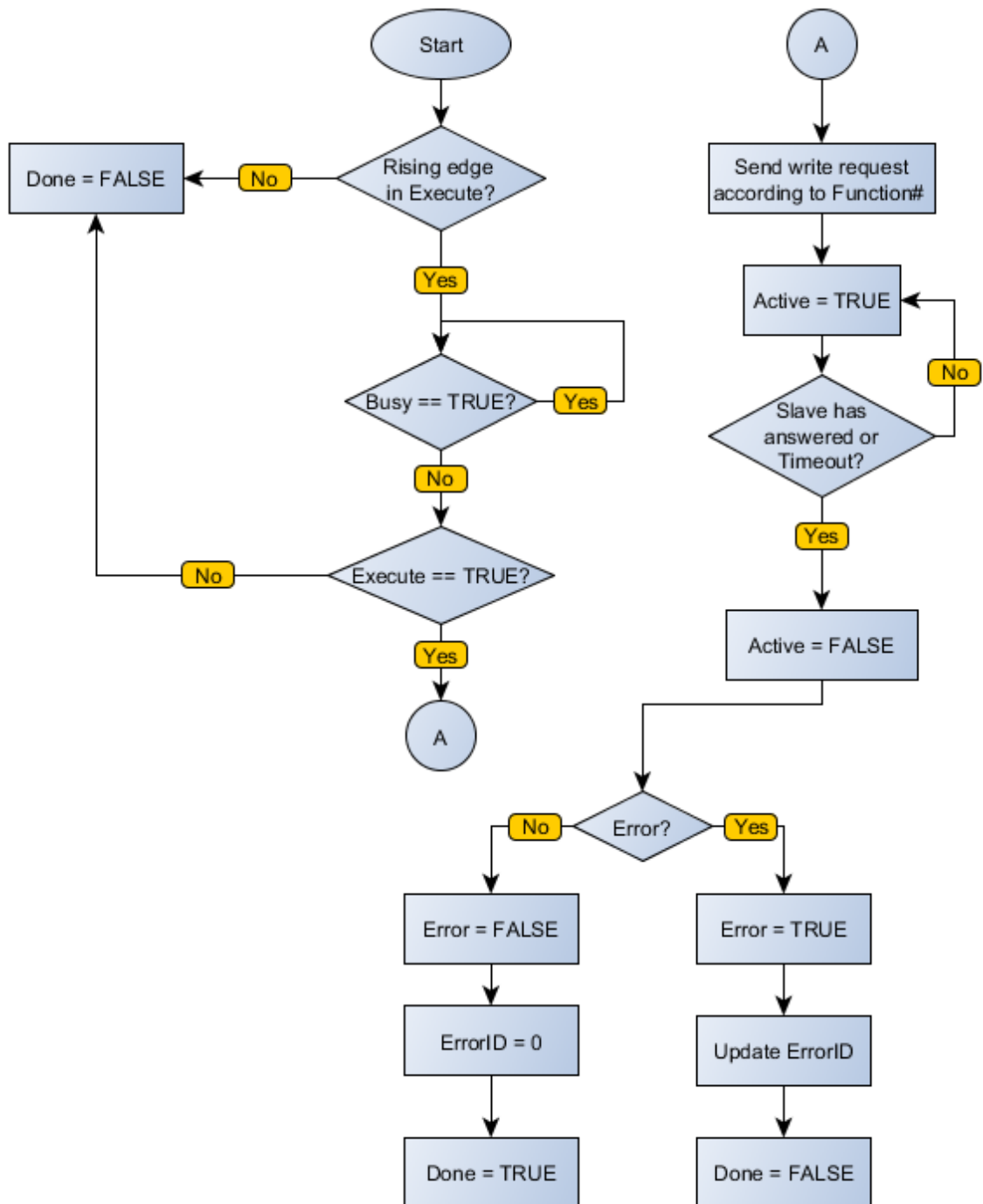
Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo

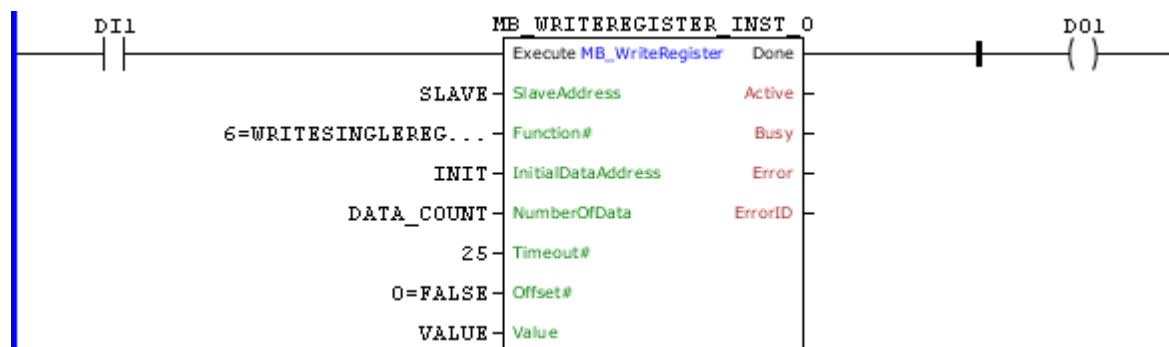
a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Mestre não habilitado
4	Timeout na resposta do escravo
5	Escravo retornou erro

Fluxograma do Bloco



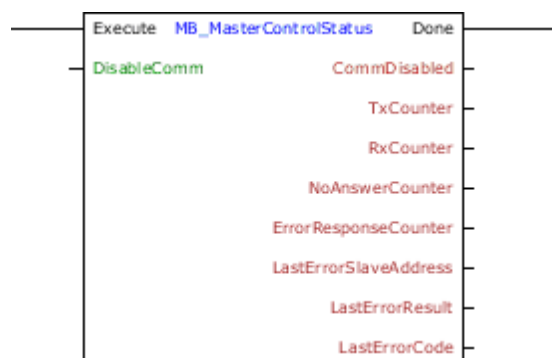
Exemplo



O exemplo acima requisita escrita do dado contido em VALUE, de tamanho descrito por DATA_COUNT, nos endereços posicionados a partir de INIT no escravo Modbus RTU de endereço SLAVE através da função Write Single Register. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que permite monitorar diversos estados do mestre da rede Modbus RTU.

Representação Ladder



Estrutura do Bloco

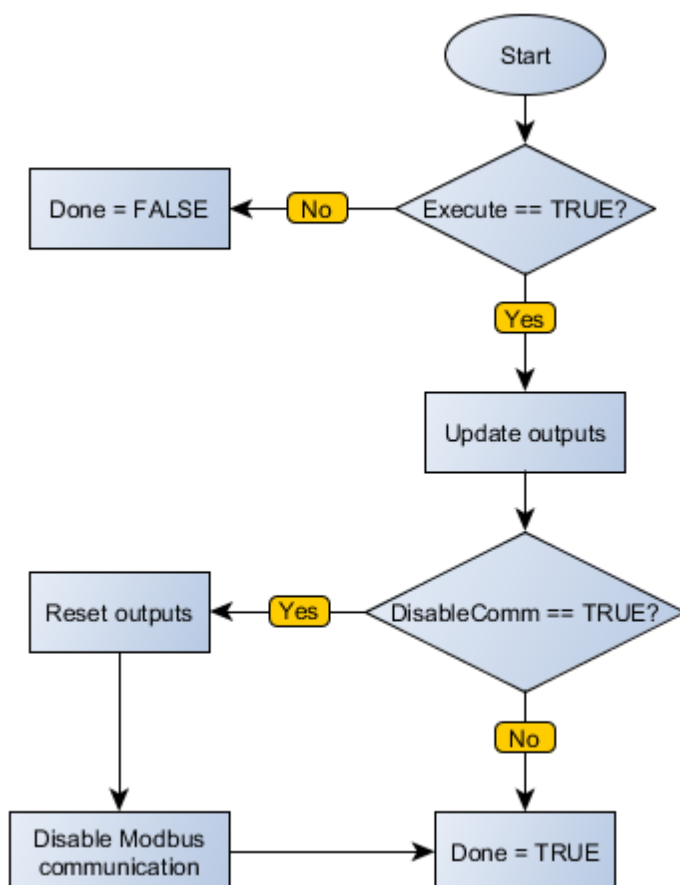
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	DisableComm	BOOL	Desabilita a comunicação Modbus RTU
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	CommDisabled	BOOL	Sinalizador de comunicação desabilitada
	TxCounter	WORD UINT	Contador de requisições enviadas
	RxCounter	WORD UINT	Contador de telegramas recebidos
	NoAnswerCounter	WORD UINT	Contador de requisições não respondidas
	ErrorResponseCounter	WORD UINT	Contador de respostas recebidas com informação de erro
	LastErrorSlaveAddress	BYTE USINT	Endereço do escravo no qual foi detectado o último erro de comunicação
	LastErrorResult	BYTE USINT	Resultado de operação do último erro de comunicação recebido (0 – Sem erro) (4 – Timeout de resposta) (5 – Escravo retornou erro)
	LastErrorCode	BYTE USINT	Código do último erro de comunicação recebido

Funcionamento

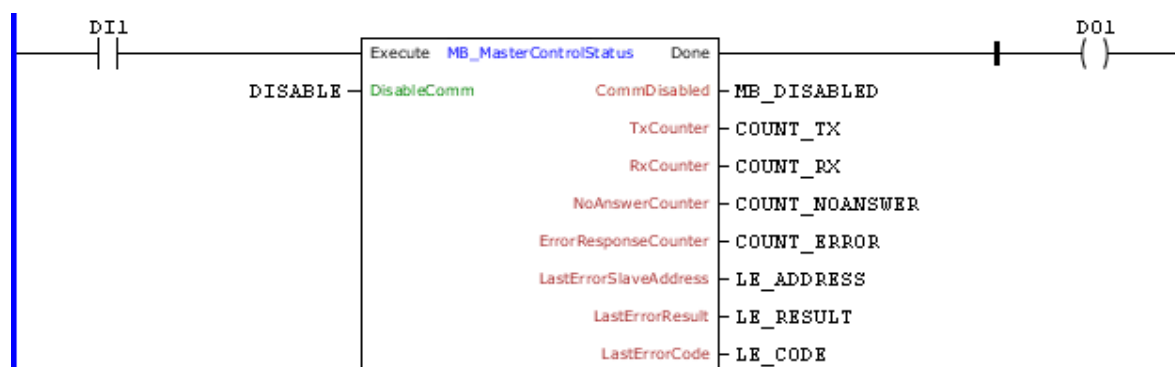
Este bloco permanece ativo enquanto Execute está em nível TRUE, atualizando suas saídas de acordo com a monitoração do mestre e com as requisições de entrada. Quando Execute recebe nível FALSE, as entradas são ignoradas e as saídas são zeradas. A saída Done recebe nível TRUE quando Execute possui nível TRUE e o bloco finalizou sua execução.

Um nível TRUE em DisableComm desabilita a comunicação Modbus RTU e zera os contadores e marcadores de status do mestre. Estes marcadores e contadores são exibidos nas saídas do bloco, cada um contendo um dado correspondendo à sua descrição. Seus valores também são zerados no desligamento do mestre.

Fluxograma do Bloco



Exemplo



O exemplo acima requisita dados de status do mestre da rede Modbus RTU, além de permitir desabilitar a comunicação através de DISABLE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que permite monitorar o estado de 4 escravos da rede Modbus RTU.

Representação Ladder



Estrutura do Bloco

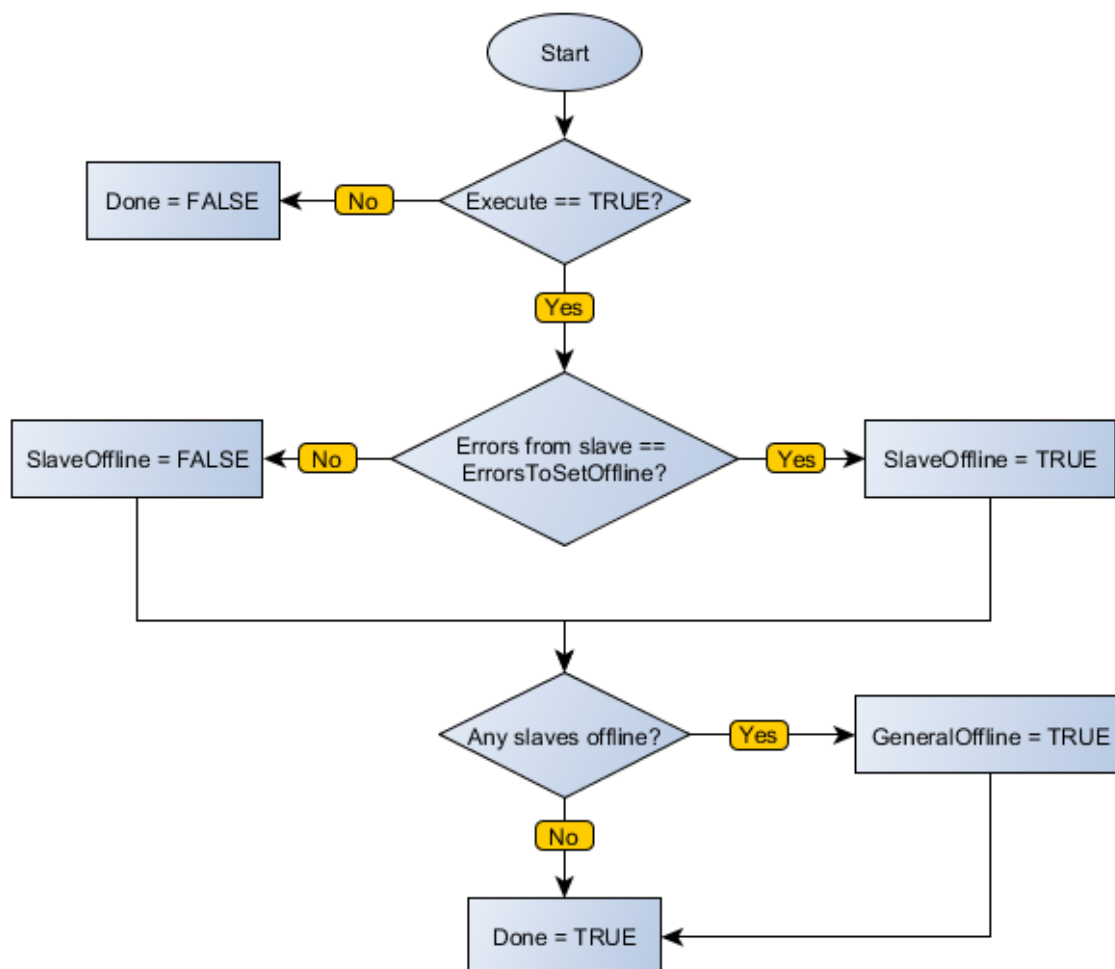
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ErrorsToSetOffline#	BYTE	Quantidade de erros que o mestre deve identificar até considerar a comunicação com um escravo off-line
	AddressSlave1#	BYTE	Endereço do escravo 1 a ser monitorado
	AddressSlave2#	BYTE	Endereço do escravo 2 a ser monitorado
	AddressSlave3#	BYTE	Endereço do escravo 3 a ser monitorado
	AddressSlave4#	BYTE	Endereço do escravo 4 a ser monitorado
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	GeneralOffline	BOOL	Sinalizador de que qualquer uma das comunicações monitoradas está off-line
	Slave1Offline	BOOL	Sinalizador de estado off-line do escravo 1
	Slave2Offline	BOOL	Sinalizador de estado off-line do escravo 2
	Slave3Offline	BOOL	Sinalizador de estado off-line do escravo 3
	Slave4Offline	BOOL	Sinalizador de estado off-line do escravo 4

Funcionamento

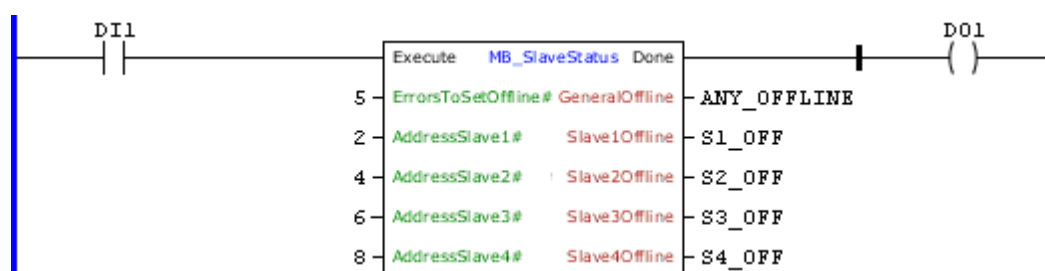
Este bloco permanece ativo enquanto Execute está em nível TRUE, atualizando suas saídas de acordo com a monitoração do número de erros registrado para cada escravo. Quando Execute recebe nível FALSE, as entradas são ignoradas e as saídas são zeradas. A saída Done recebe nível TRUE quando Execute possui nível TRUE e o bloco finalizou sua execução.

A entrada ErrorsToSetOffline# permite cadastrar o número de erros identificados em um escravo que caracterizará uma comunicação off-line. As entradas AddressSlave permitem inserir quatro endereços de escravos a serem monitorados. Quando esse escravo monitorado relatar o número de erros programado, a sua saída SlaveOffline correspondente é colocada em nível TRUE. Se qualquer uma das saídas SlaveOffline estiver em nível TRUE, GeneralOffline também recebe nível TRUE.

Fluxograma do Bloco



Exemplo



O exemplo acima verifica o número de respostas de erro enviado pelos escravos 2, 4, 6 e 8 da rede Modbus RTU. Caso algum deles seja superior a 5, seu status SX_OFF é levado a nível TRUE. Finalizado o bloco com sucesso, a saída Done é ativada.

10.1.5.1.22.3 Modbus TCP

Operação na Rede Modbus TCP - Modo Cliente

Além da operação como servidor, o controlador programável PLC300 também permite a operação como cliente da rede Modbus TCP. Para esta operação, é necessário observar os seguintes pontos:

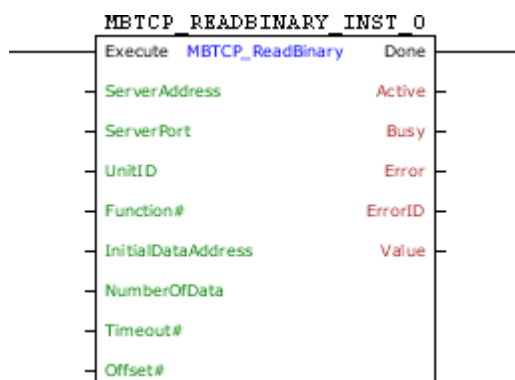
- O envio e recepção de telegramas via interface Ethernet utilizando o protocolo Modbus TCP é programado utilizando blocos em linguagem de programação Ladder. É necessário conhecer os blocos disponíveis e o *software* de programação em Ladder para poder programar o cliente da rede.
- As seguintes funções estão disponíveis para envio de requisições pelo cliente Modbus TCP:
 - o Função 01: Read Coils
 - o Função 02: Read Discrete Inputs
 - o Função 03: Read Holding Registers
 - o Função 04: Read Input Registers
 - o Função 05: Write Single Coil
 - o Função 06: Write Single Register
 - o Função 15: Write Multiple Coils
 - o Função 16: Write Multiple Registers

Blocos para a Programação do Cliente

Para o controle e monitoração da comunicação Modbus TCP utilizando o controlador programável PLC300, foram desenvolvidos os seguintes blocos, que devem ser utilizados durante a programação em Ladder.

Bloco que executa uma leitura de até 128 dados binários (via Read Coils ou Read Discrete Inputs) de um servidor em rede Modbus TCP.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ServerAddress	DWORD	Endereço IP do servidor (Ex: 192.168.0.1)
	ServerPort	WORD	Porta Modbus TCP do servidor (Padrão: 502)
	UnitID	BYTE	UnitID do servidor (Padrão: 255)
	Function#	BYTE	Código da função de leitura
	InitialDataAddress	WORD	Endereço do bit inicial dos dados a serem lidos
	NumberOfData	BYTE	Número de bits a serem lidos (1 a 128)
	Timeout#	WORD	Tempo máximo de espera pela resposta do servidor [ms]
VAR_OUTPUT	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a conexão está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE	Identificador do erro ocorrido
VAR	Value	BOOL	Variável que armazena os dados recebidos
	MBTCP_READBINARY_INST_0	MBTCP_READBINARY	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o servidor Modbus TCP no endereço especificado em ServerAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de leitura de um número de bits indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do servidor. Os dados recebidos são armazenados na variável Value. Caso o servidor não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.



NOTA!

Value é um array de tamanho igual a NumberOfData. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

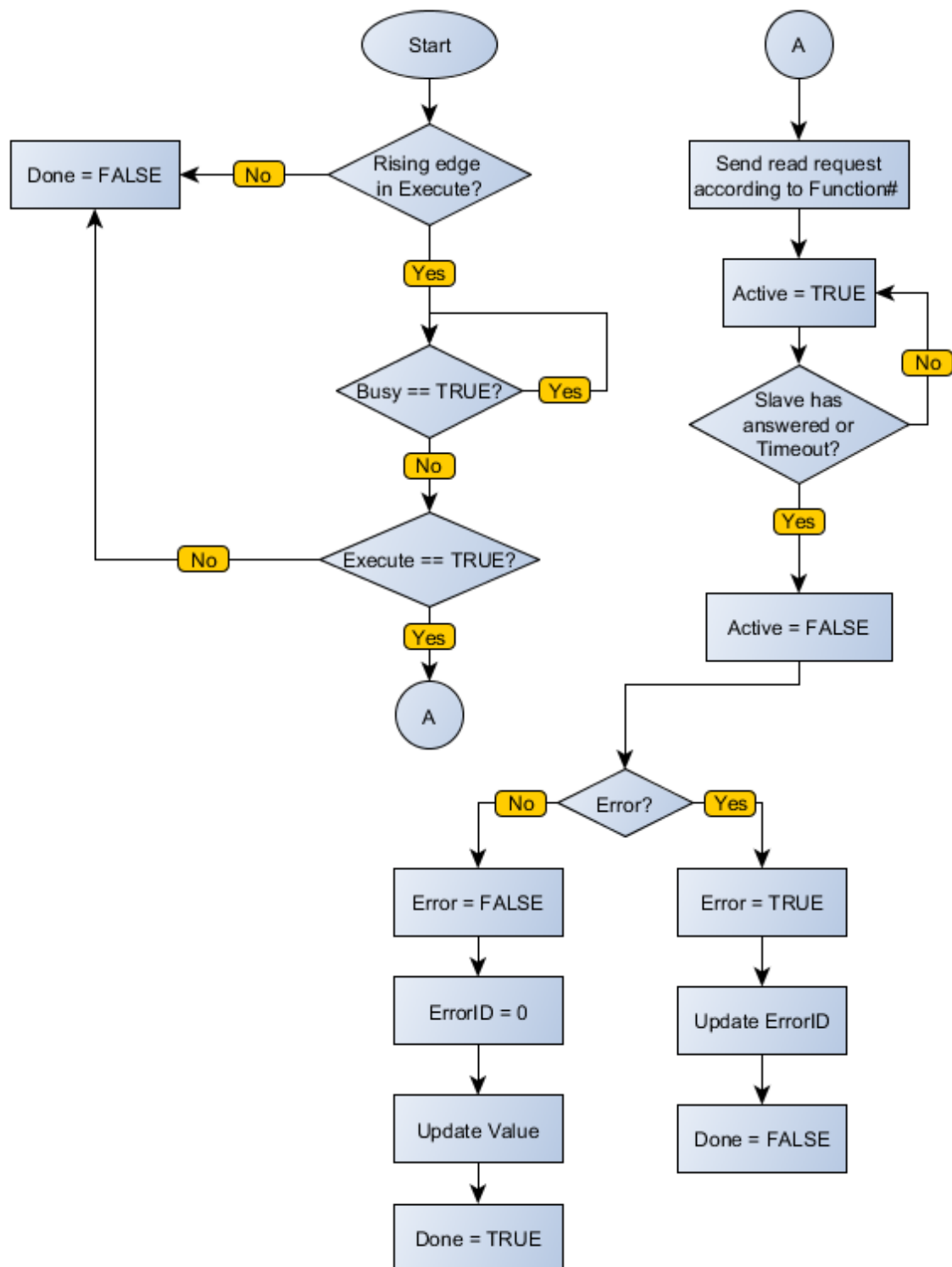
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Cliente não habilitado
4	Timeout na resposta do servidor
5	Servidor retornou erro
6	Falha ao conectar com o servidor
7	Conexão TCP/IP terminada prematuramente

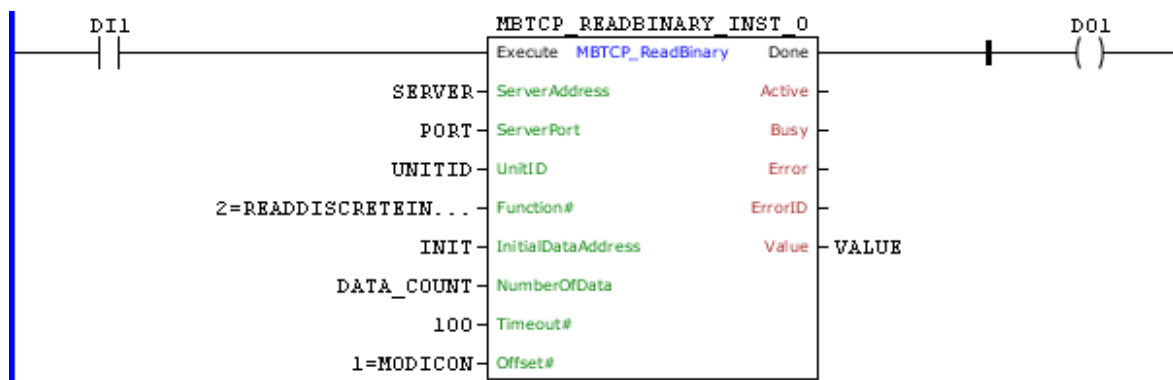
Compatibilidade

Equipamento	Versão
PLC300	1.30 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



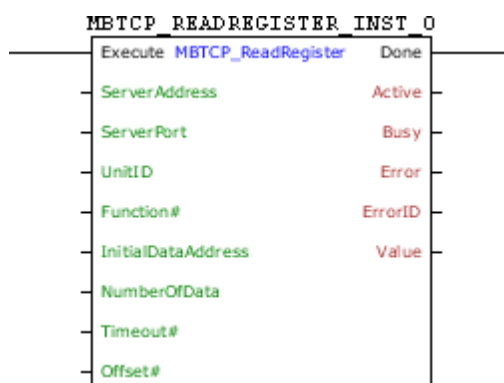
Exemplo



O exemplo acima requisita leitura de um número de dados binários descrito por DATA_COUNT, posicionados em INIT - 1 no servidor Modbus TCP de endereço SERVER:PORT através da função Read Discrete Input. Esses dados são repassados a VALUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma leitura de até 16 registradores de 16 bits (via Read Holding Registers ou Read Input Registers) de um servidor em rede Modbus TCP.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ServerAddress	DWORD	Endereço IP do servidor (Ex: 192.168.0.1)
	ServerPort	WORD	Porta Modbus TCP do servidor (Padrão: 502)
	UnitID	BYTE	UnitID do servidor (Padrão: 255)
	Function#	BYTE	Código da função de leitura
	InitialDataAddress	WORD	Endereço do registrador inicial a ser lido
	NumberOfData	BYTE	Número de registradores a serem lidos (1 a 16)
	Timeout#	WORD	Tempo máximo de espera pela resposta do servidor [ms]
	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a conexão está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE	Identificador do erro ocorrido
	Value	BYTE SINT USINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena os dados recebidos
VAR	MBTCP_READREGISTER_INST_0	MBTCP_READREGISTER	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o servidor Modbus TCP no endereço especificado em ServerAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de leitura de um número de registradores indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do servidor. Os dados recebidos são armazenados na variável Value. Caso o servidor não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.

**NOTA!**

Value é um array de número de bits a NumberOfData multiplicado por 16. Ou seja, caso NumberOfData seja 16, Value pode ser um array de 32 posições BYTE, 16 posições WORD ou 8 posições DWORD. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

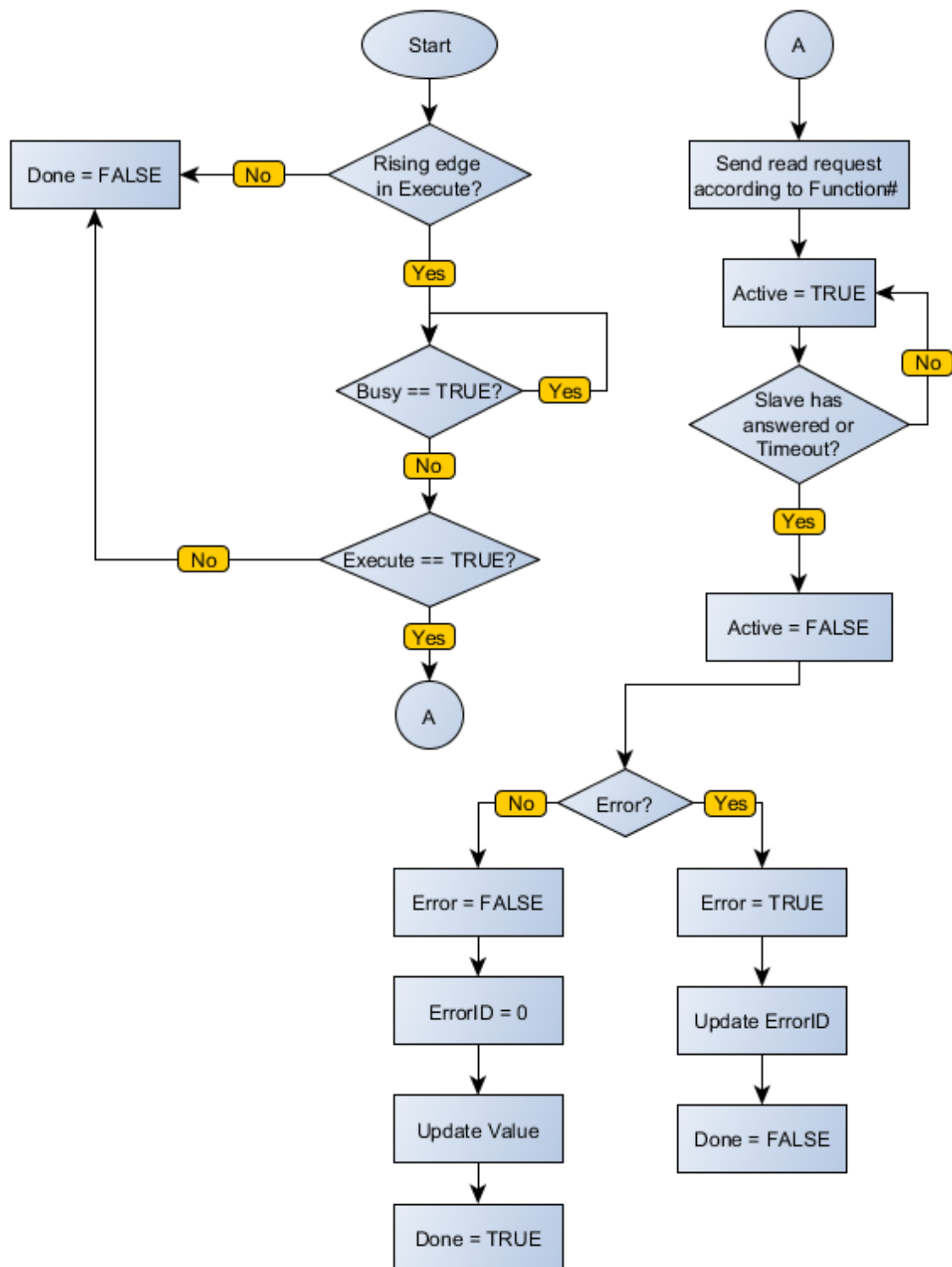
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Cliente não habilitado
4	Timeout na resposta do servidor
5	Servidor retornou erro
6	Falha ao conectar com o servidor
7	Conexão TCP/IP terminada prematuramente

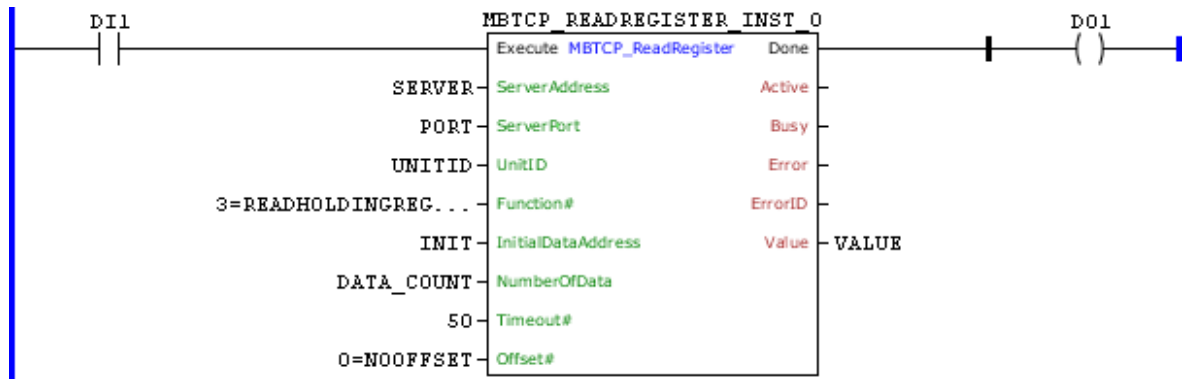
Compatibilidade

Equipamento	Versão
PLC300	1.30 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



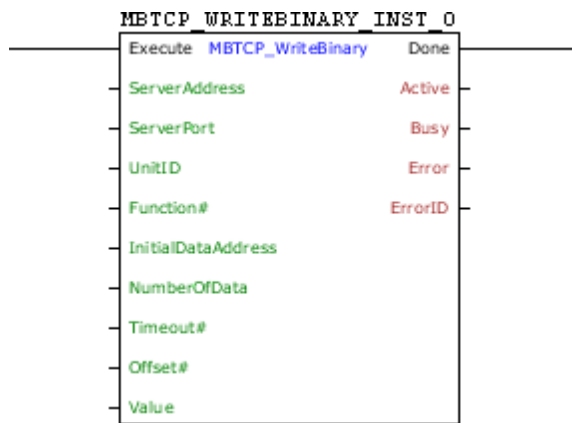
Exemplo



O exemplo acima requisita leitura de um número de dados de registradores descrito por DATA_COUNT, posicionados em INIT no servidor Modbus TCP de endereço SERVER:PORT através da função Read Holding Register. Esses dados são repassados a VALUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma escrita de até 128 dados binários (via Write Single Coil ou Write Multiple Coils) em um servidor em rede Modbus TCP.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ServerAddress	DWORD	Endereço IP do servidor (Ex: 192.168.0.1)
	ServerPort	WORD	Porta Modbus TCP do servidor (Padrão: 502)
	UnitID	BYTE	UnitID do servidor (Padrão: 255)
	Function#	BYTE	Código da função de escrita
	InitialDataAddress	WORD	Endereço do bit inicial onde os dados serão escritos
	NumberOfData	BYTE	Número de bits a serem escritos (1 a 128)
	Timeout#	WORD	Tempo máximo de espera pela resposta do servidor [ms]
	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
	Value	BOOL	Variável que armazena os dados a serem escritos
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a conexão está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE	Identificador do erro ocorrido
VAR	MBTCP_WRITEBINARY_INST_0	MBTCP_WRITEBINARY	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o servidor Modbus TCP no endereço especificado em ServerAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de escrita dos valores de Value em um número de bits indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do servidor. Caso o servidor não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.



NOTA!

Value é um array de tamanho igual a NumberOfData. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

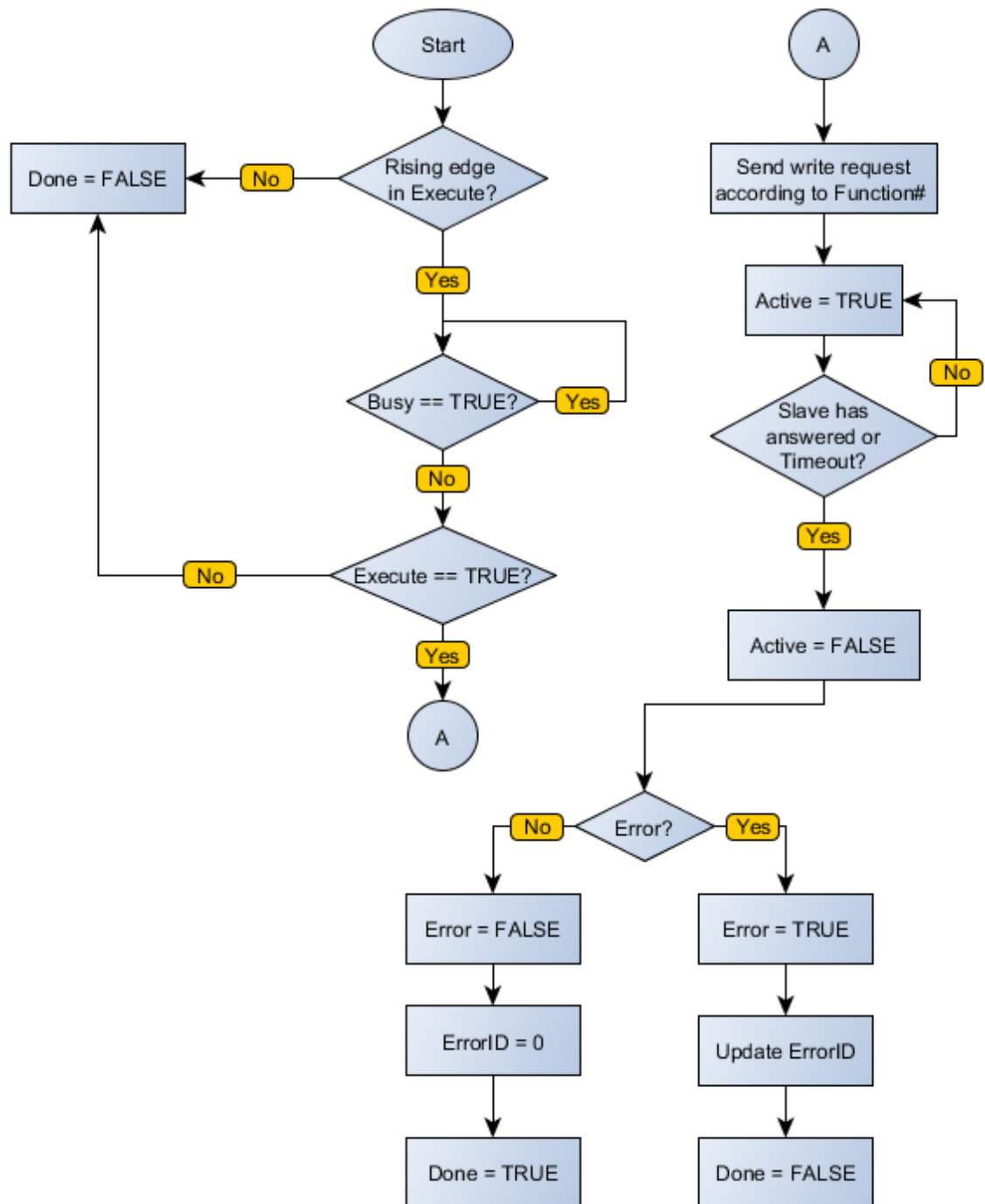
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Cliente não habilitado
4	Timeout na resposta do servidor
5	Servidor retornou erro
6	Falha ao conectar com o servidor
7	Conexão TCP/IP terminada prematuramente

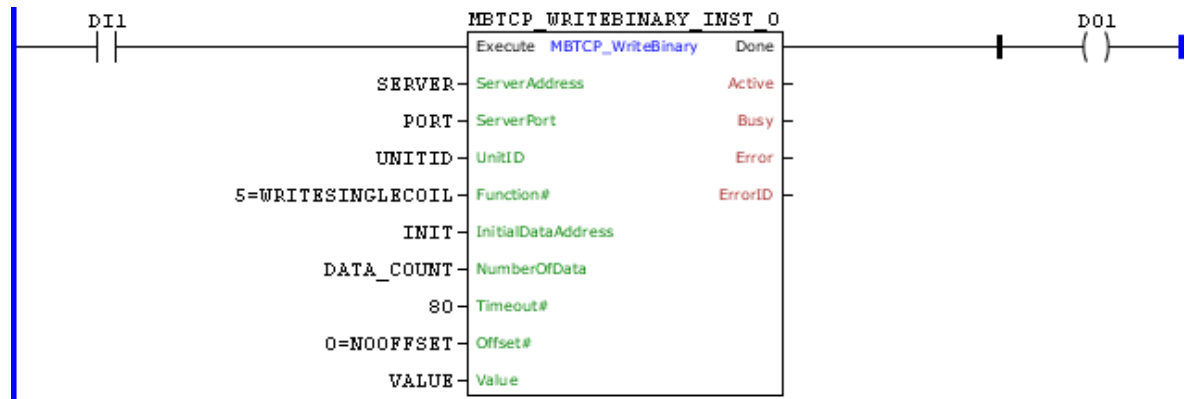
Compatibilidade

Equipamento	Versão
PLC300	1.30 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



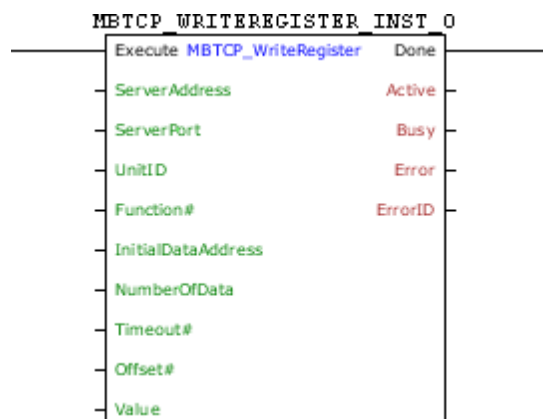
Exemplo



O exemplo acima requisita escrita do dado contido em VALUE, de tamanho descrito por DATA_COUNT, nos endereços posicionados a partir de INIT no servidor Modbus TCP de endereço SERVER:PORT através da função Write Single Coil. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma escrita de até 16 registradores de 16 bits (via Write Single Register ou Write Multiple Registers) de um servidor em rede Modbus TCP.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ServerAddress	DWORD	Endereço IP do servidor (Ex: 192.168.0.1)
	ServerPort	WORD	Porta Modbus TCP do servidor (Padrão: 502)
	UnitID	BYTE	UnitID do servidor (Padrão: 255)
	Function#	BYTE	Código da função de escrita
	InitialDataAddress	WORD	Endereço do registrador inicial a ser escrito
	NumberOfData	BYTE	Número de registradores a serem escritos (1 a 16)
	Timeout#	WORD	Tempo máximo de espera pela resposta do servidor [ms]
	Offset#	BOOL	Indicação de offset em InitialDataAddress, ou seja, necessidade de subtrair 1 deste número
	Value	BYTE SINT USINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena os dados a serem escritos
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que a conexão está ocupada com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE	Identificador do erro ocorrido
VAR	MB_TCPWRITEREGISTER_INST_0	MB_TCPWRITEREGISTER	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o servidor Modbus TCP no endereço especificado em ServerAddress está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de escrita dos valores de Value em um número de registradores indicado por NumberOfData no endereço InitialDataAddress utilizando a função escolhida em Function# e seta a saída Active, resetando-a ao receber a resposta do servidor. Caso o servidor não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.



NOTA!

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.

**NOTA!**

Value é um array de número de bits a NumberOfData multiplicado por 16. Ou seja, caso NumberOfData seja 16, Value pode ser um array de 32 posições BYTE, 16 posições WORD ou 8 posições DWORD. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

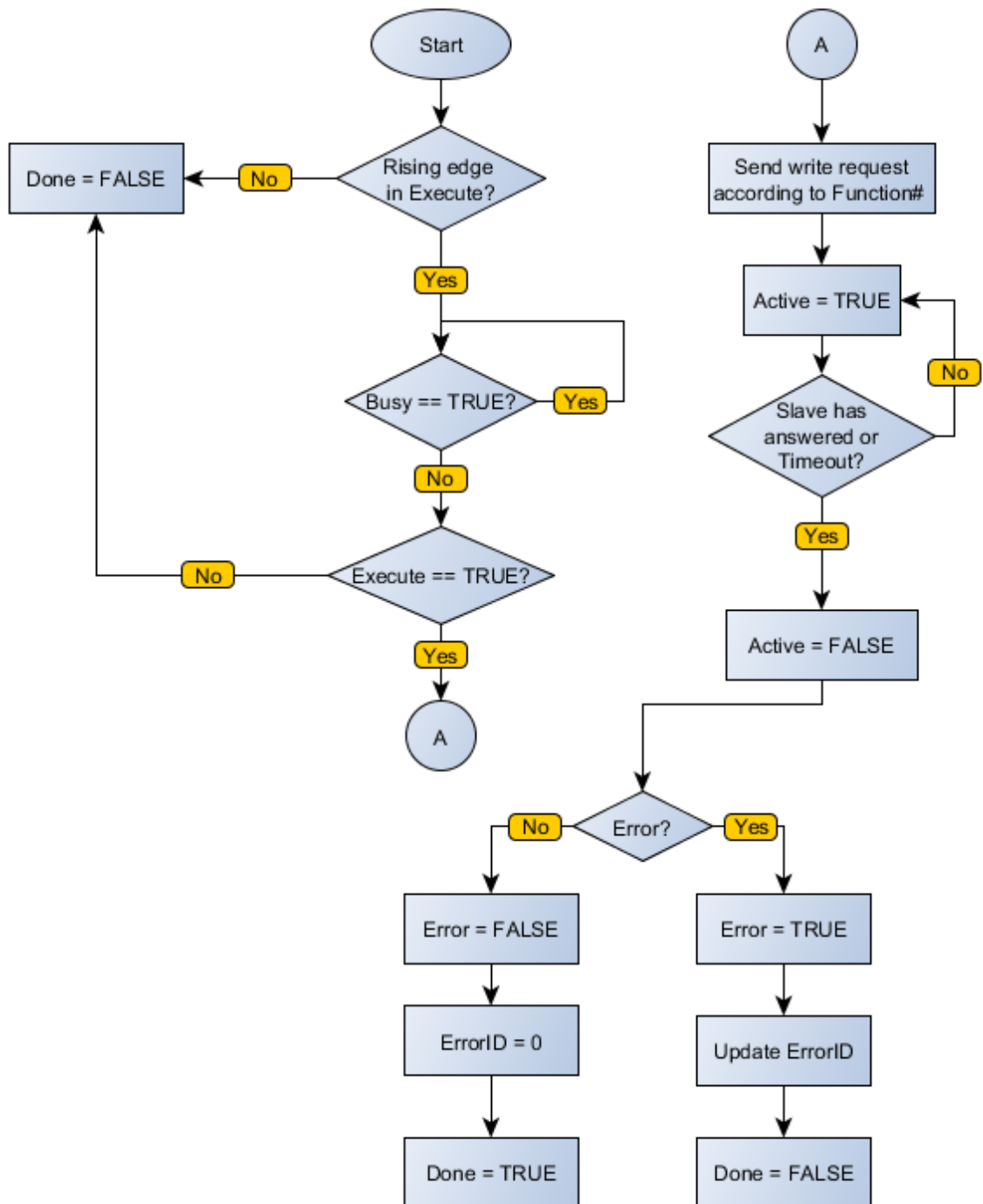
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Algum dado de entrada inválido
2	Cliente não habilitado
4	Timeout na resposta do servidor
5	Servidor retornou erro
6	Falha ao conectar com o servidor
7	Conexão TCP/IP terminada prematuramente

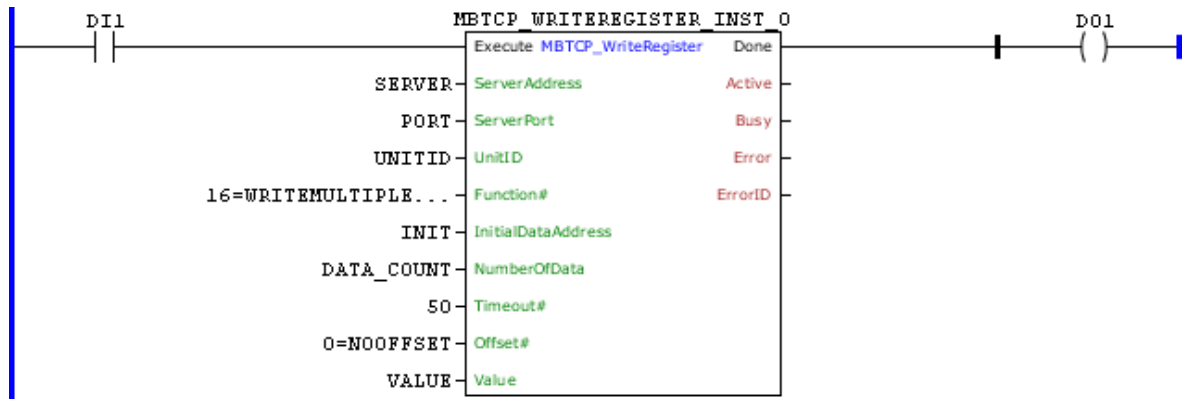
Compatibilidade

Equipamento	Versão
PLC300	1.30 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



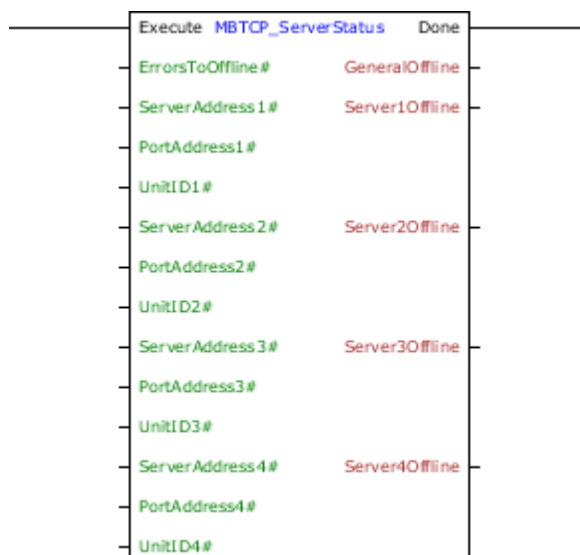
Exemplo



O exemplo acima requisita escrita do dado contido em VALUE, de tamanho descrito por DATA_COUNT, nos endereços posicionados a partir de INIT no servidor Modbus TCP de endereço SERVER:PORT através da função Write Multiple Registers. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que permite monitorar o estado de 4 servidores da rede Modbus TCP.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ErrorsToSetOffline#	BYTE	Quantidade de erros que se deve identificar até considerar a comunicação com um servidor off-line
	ServerAddress1#	DWORD	Endereço do servidor 1 a ser monitorado (Ex: 192.168.0.1)
	PortAddress1#	WORD	Porta do servidor 1 a ser monitorado (Padrão: 502)
	UnitID1#	BYTE	UnitID do servidor 1 a ser monitorado (Padrão: 255)
	ServerAddress2#	DWORD	Endereço do servidor 2 a ser monitorado (Ex: 192.168.0.1)
	PortAddress2#	WORD	Porta do servidor 2 a ser monitorado (Padrão: 502)
	UnitID2#	BYTE	UnitID do servidor 2 a ser monitorado (Padrão: 255)
	ServerAddress3#	DWORD	Endereço do servidor 3 a ser monitorado (Ex: 192.168.0.1)
	PortAddress3#	WORD	Porta do servidor 3 a ser monitorado (Padrão: 502)
	UnitID3#	BYTE	UnitID do servidor 3 a ser monitorado (Padrão: 255)
	ServerAddress4#	DWORD	Endereço do servidor 4 a ser monitorado (Ex: 192.168.0.1)
	PortAddress4#	WORD	Porta do servidor 4 a ser monitorado (Padrão: 502)
	UnitID4#	BYTE	UnitID do servidor 4 a ser monitorado (Padrão: 255)
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	GeneralOffline	BOOL	Sinalizador de que qualquer uma das comunicações monitoradas está off-line
	Server1Offline	BOOL	Sinalizador de estado off-line do servidor 1
	Server2Offline	BOOL	Sinalizador de estado off-line do servidor 2
	Server3Offline	BOOL	Sinalizador de estado off-line do servidor 3
	Server4Offline	BOOL	Sinalizador de estado off-line do servidor 4

Funcionamento

Este bloco permanece ativo enquanto Execute está em nível TRUE, atualizando suas saídas de acordo com a monitoração do número de erros registrado para cada servidor. Quando Execute recebe nível FALSE, as entradas são ignoradas e as saídas são zeradas. A saída Done recebe nível TRUE quando Execute possui nível TRUE e o bloco finalizou sua execução.

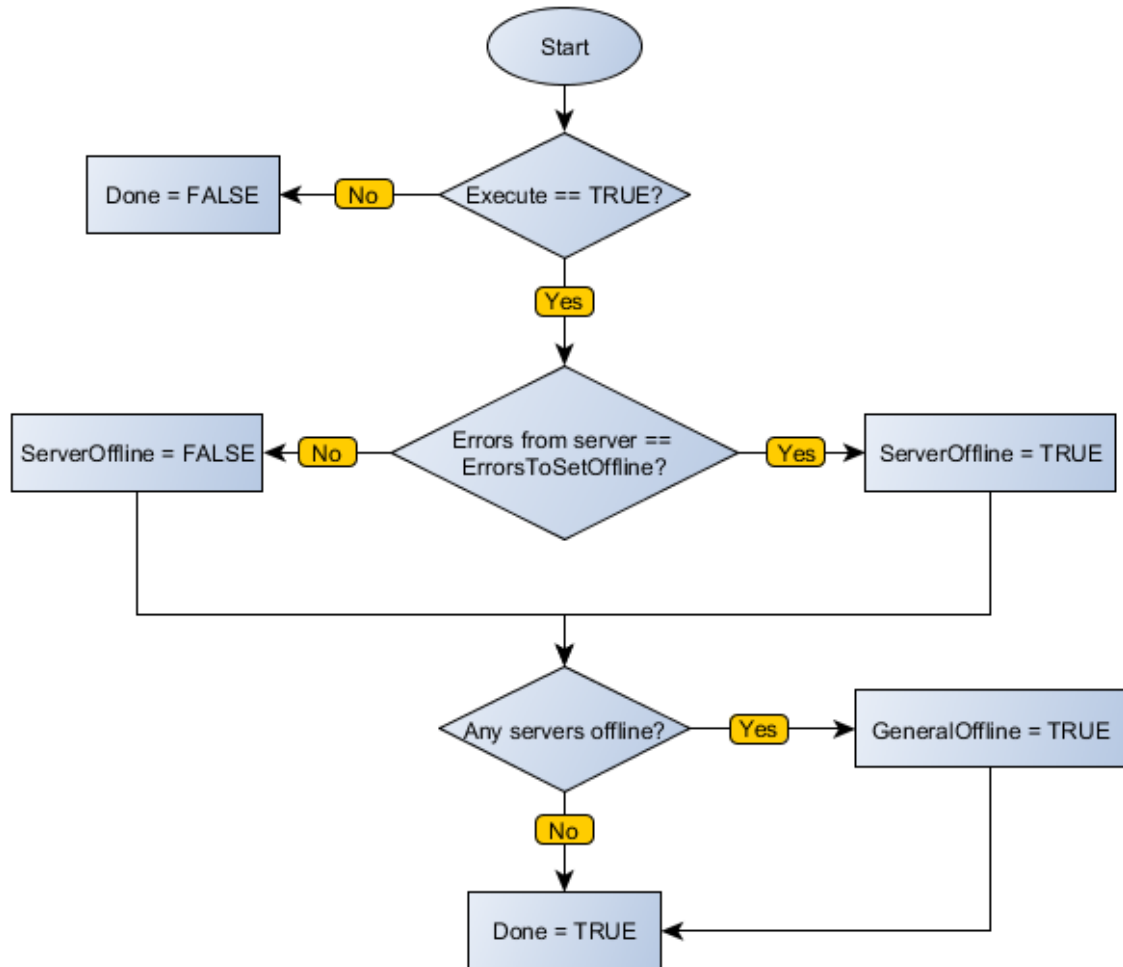
A entrada ErrorsToSetOffline# permite cadastrar o número de erros identificados em um servidor que caracterizará uma comunicação off-line. As entradas AddressServer permitem inserir quatro endereços de servidores a serem monitorados. Caso queira deixar um canal ignorado, insira o valor 0 no endereço do servidor. Quando esse servidor monitorado relatar o número de erros programado, a sua saída ServerOffline correspondente é colocada em nível TRUE. Se qualquer uma das saídas

ServerOffline esteve em nível TRUE, GeneralOffline também recebe nível TRUE.

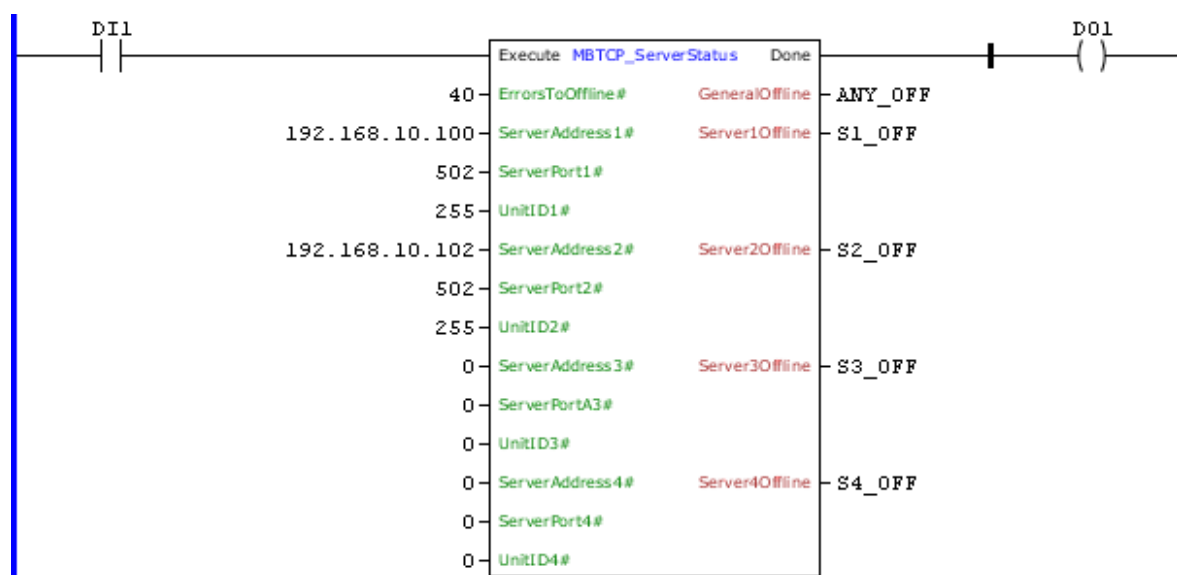
Compatibilidade

Equipamento	Versão
PLC300	1.30 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



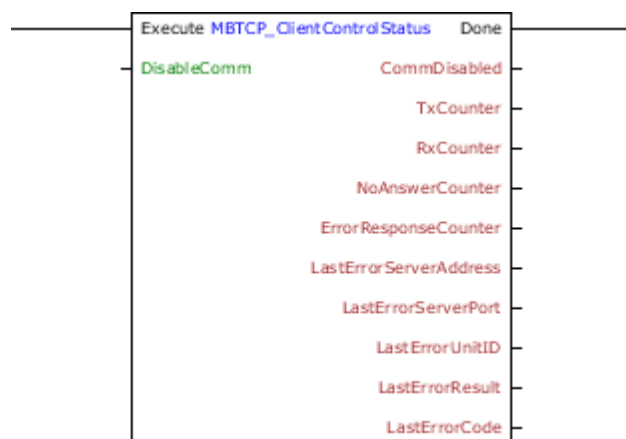
Exemplo



O exemplo acima verifica o número de respostas de erro enviado pelos servidores 192.168.10.100:502 e 192.168.10.102:502 da rede Modbus TCP. Caso algum deles seja superior a 40, seu status SX_OFF é levado a nível TRUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que permite monitorar diversos estados do cliente da rede Modbus TCP.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	DisableComm	BOOL	Desabilita a comunicação Modbus TCP
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	CommDisabled	BOOL	Sinalizador de comunicação desabilitada
	TxCounter	WORD UINT	Contador de requisições enviadas
	RxCounter	WORD UINT	Contador de telegramas recebidos
	NoAnswerCounter	WORD UINT	Contador de requisições não respondidas
	ErrorResponseCounter	WORD UINT	Contador de respostas recebidas com informação de erro
	LastErrorServerAddress	DWORD	Endereço do servidor no qual foi detectado o último erro de comunicação
	LastErrorServerPort	WORD UINT	Porta do servidor no qual foi detectado o último erro de comunicação
	LastErrorUnitID	BYTE USINT	UnitID do servidor no qual foi detectado o último erro de comunicação
	LastErrorResult	BYTE USINT	Resultado de operação do último erro de comunicação recebido (0 – Sem erro) (4 – Timeout de resposta) (5 – Servidor retornou erro)
	LastErrorCode	BYTE USINT	Código do último erro de comunicação recebido

Funcionamento

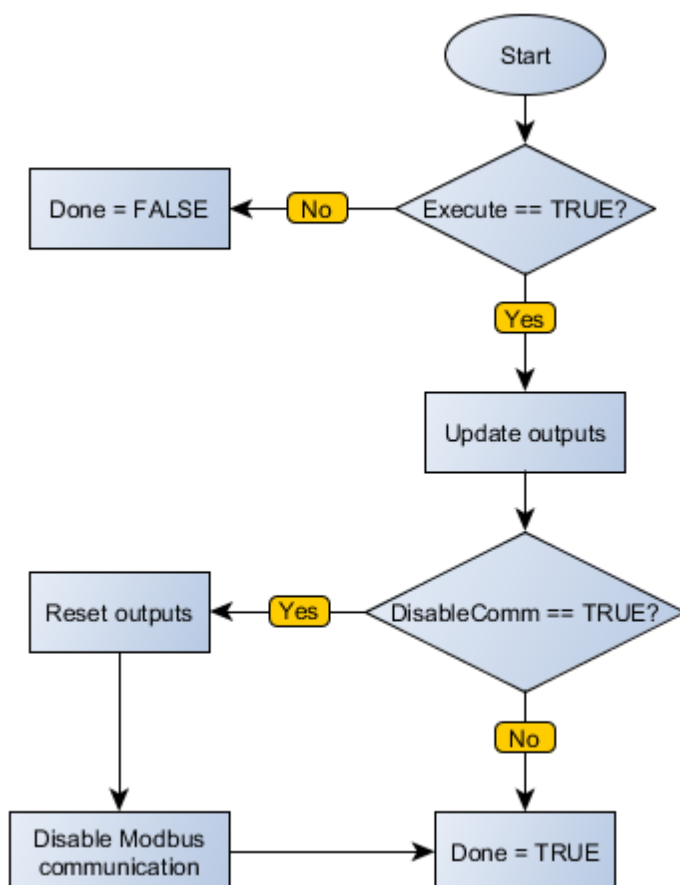
Este bloco permanece ativo enquanto Execute está em nível TRUE, atualizando suas saídas de acordo com a monitoração do mestre e com as requisições de entrada. Quando Execute recebe nível FALSE, as entradas são ignoradas e as saídas são zeradas. A saída Done recebe nível TRUE quando Execute possui nível TRUE e o bloco finalizou sua execução.

Um nível TRUE em DisableComm desabilita a comunicação Modbus TCP e zera os contadores e marcadores de status do cliente. Estes marcadores e contadores são exibidos nas saídas do bloco, cada um contendo um dado correspondendo à sua descrição. Seus valores também são zerados no desligamento do cliente.

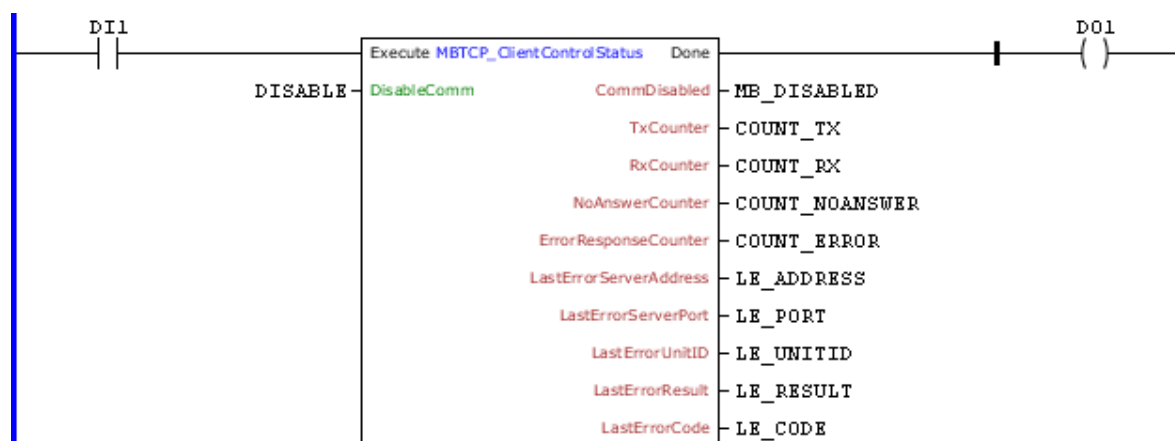
Compatibilidade

Equipamento	Versão
PLC300	1.30 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



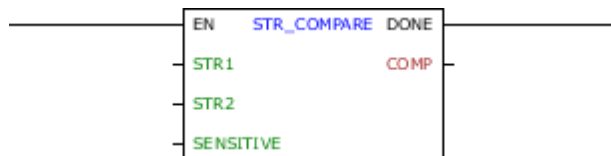
O exemplo acima requisita dados de status do cliente da rede Modbus TCP, além de permitir desabilitar a comunicação através de DISABLE. Finalizado o bloco com sucesso, a saída Done é ativada.

10.1.5.1.23 String

10.1.5.1.23.1 STR_COMPARE

Bloco que realiza comparação entre STRINGS.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR1	STRING	Primeira STRING da comparação
	STR2	STRING	Segunda STRING da comparação
	SENSITIVE	BOOL	Seleciona se a comparação diferenciará maiúsculas e minúsculas ou não
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	COMP	BYTE	Valor da comparação

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de COMP de acordo com as STRINGS de entrada. A entrada SENSITIVE com valor TRUE força a comparação a levar em consideração letras maiúsculas e minúsculas, enquanto um valor FALSE desconsidera essa diferenciação. Os valores que COMP pode assumir são:

- 0, se ambas forem iguais;
- -1, se STR1 vem antes na ordem alfabética;
- 1, se STR1 vem depois na ordem alfabética.

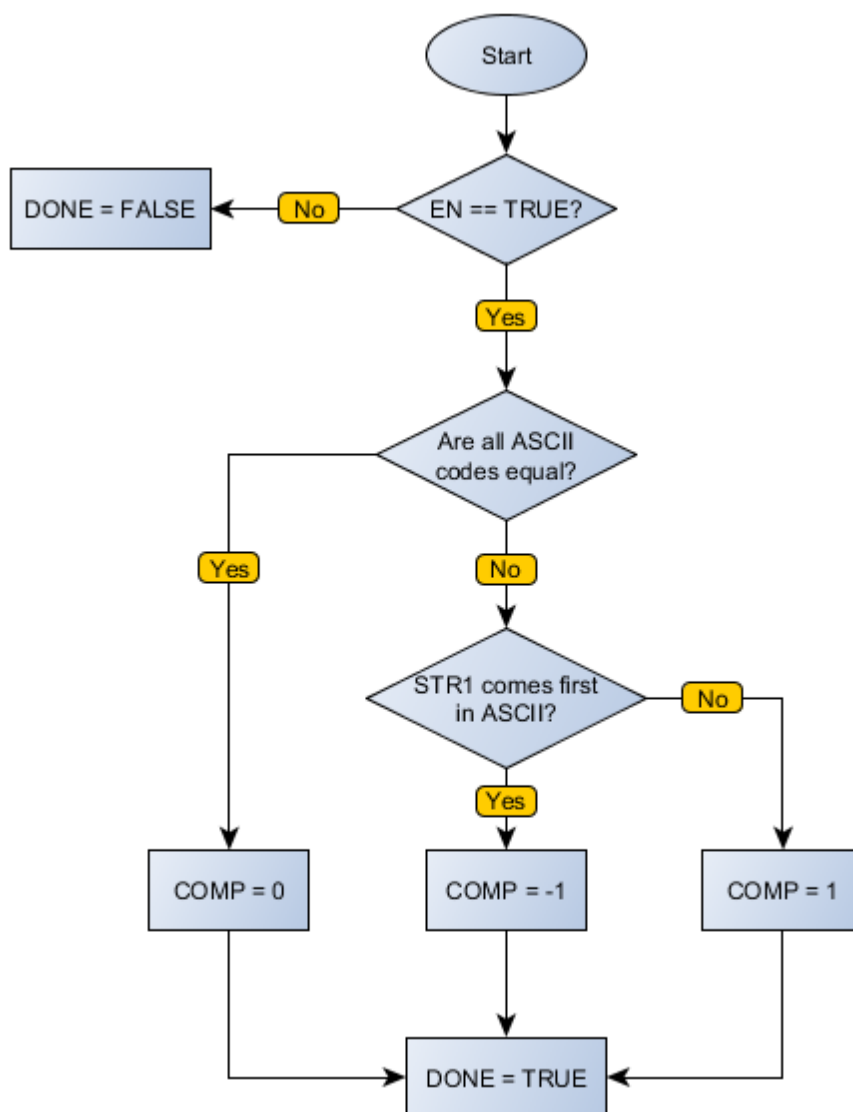
A comparação leva em consideração os valores da tabela ASCII para os caracteres.

O valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

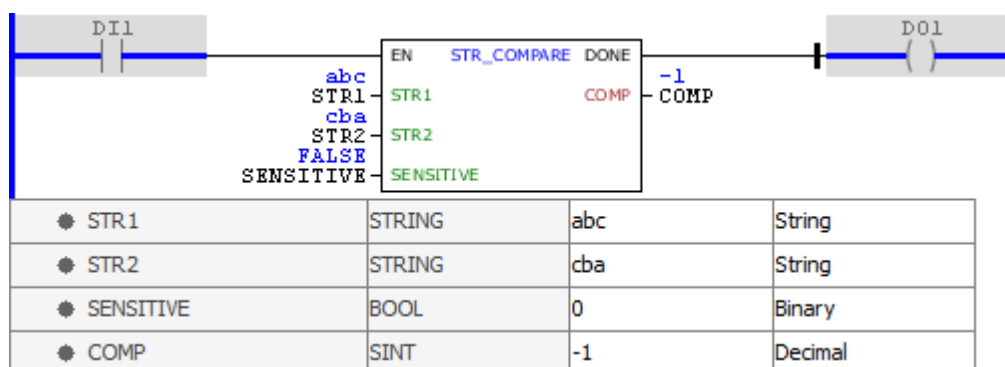
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

Fluxograma do Bloco

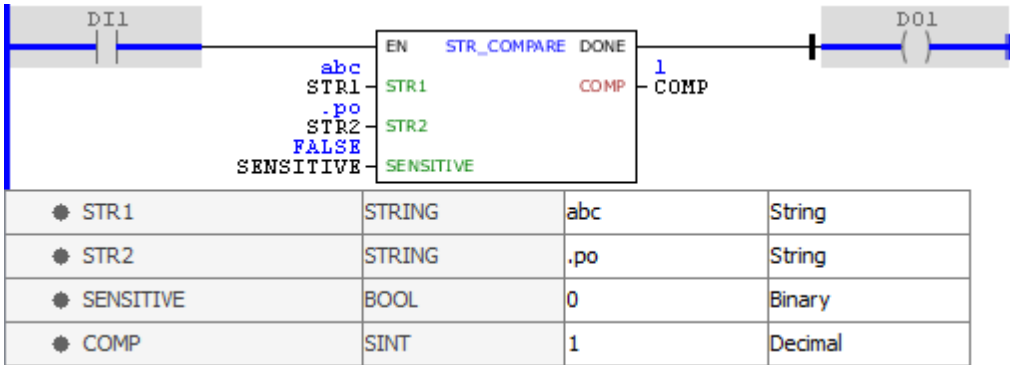


Exemplo

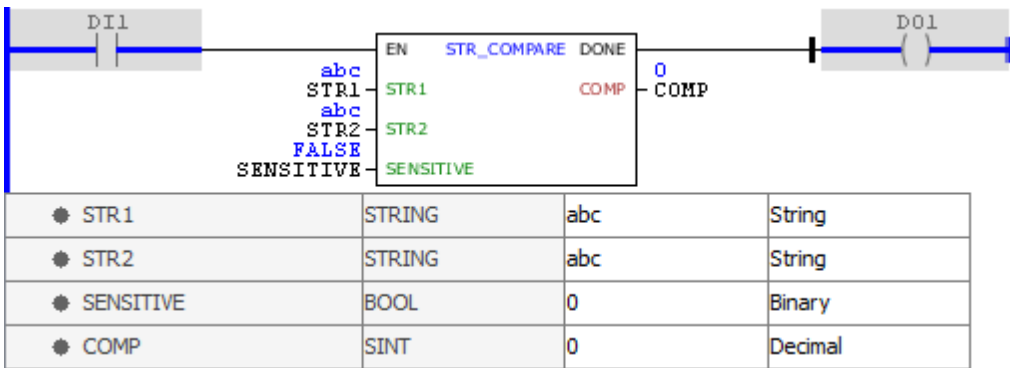


No exemplo acima, o código ASCII de 'a', primeiro caractere de STR1, é menor que o de 'c', primeiro

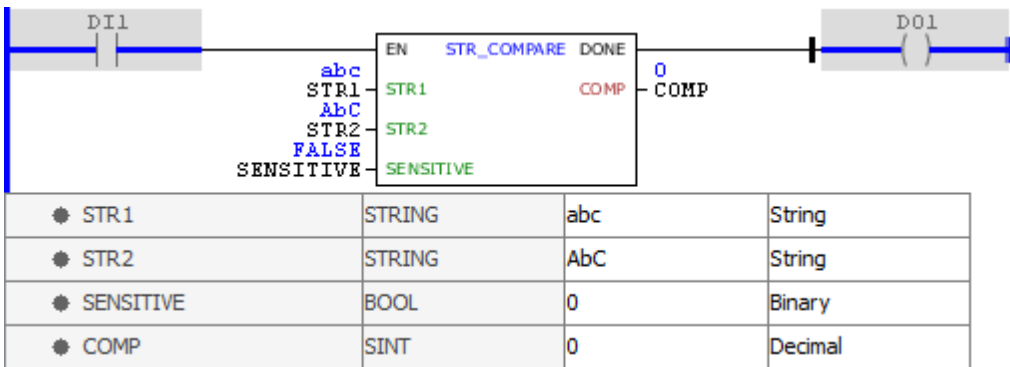
caractere de STR2. Por isso, COMP recebe o valor de -1. Finalizado o bloco com sucesso, a saída DONE é ativada.



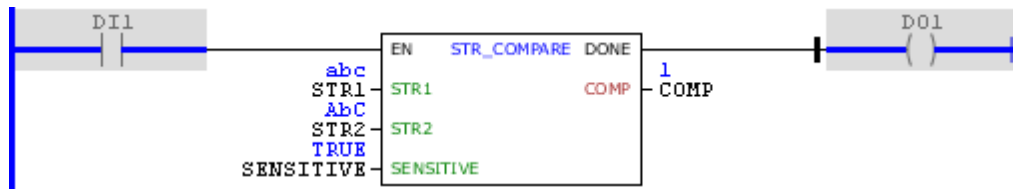
No exemplo acima, o código ASCII de 'a', primeiro caractere de STR1, é maior que o de '.', primeiro caractere de STR2. Por isso, COMP recebe o valor de 1. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, STR1 e STR2 são iguais. Por isso, COMP recebe o valor de 0. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, STR1 e STR2 são iguais, desconsiderando maiúsculas e minúsculas (SENSITIVE com valor FALSE). Por isso, COMP recebe o valor de 0. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, STR1 e STR2 são diferentes, considerando maiúsculas e minúsculas (SENSITIVE com valor TRUE). O código ASCII de 'a', primeiro caractere de STR1, é maior que o de 'A', primeiro caractere de STR2. Por isso, COMP recebe o valor de 1. Finalizado o bloco com sucesso, a saída DONE é ativada.

10.1.5.1.23.2 STR_CONCAT

Bloco que realiza concatenação de STR1 e STR2, armazenando o resultado em DST.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR1	STRING	Primeira STRING
	STR2	STRING	Segunda STRING
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	DST	BYTE	Variável que recebe a nova STRING formada da junção de STR1 e STR2

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de DST de acordo com as STRINGS de entrada. DST recebe o valor de STR1 concatenado com valor de STR2 em seu fim.

O valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.



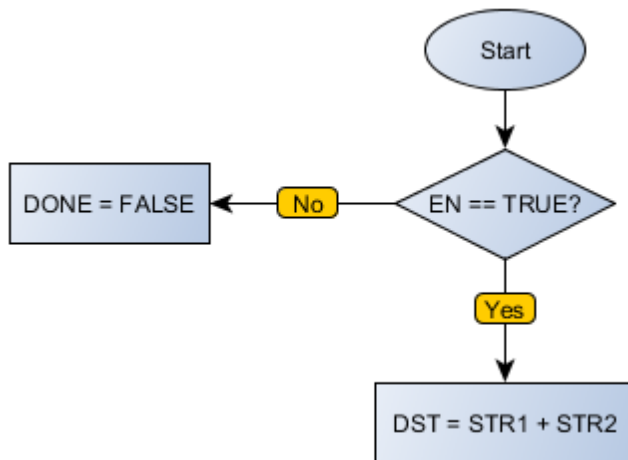
NOTA!

Caso o tamanho de DST seja menor que a soma do número de caracteres de STR1 e STR2, o valor resultante será truncado.

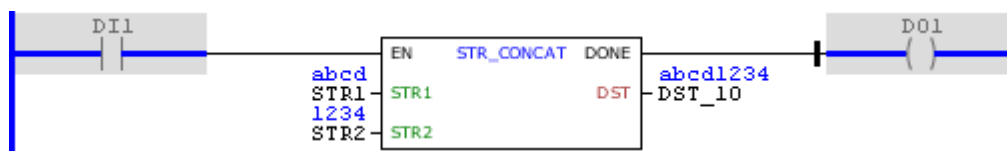
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

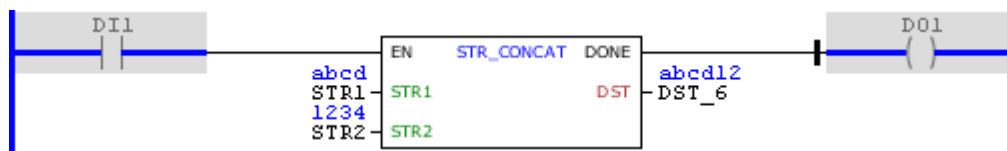
Fluxograma do Bloco



Exemplo



No exemplo acima, STR1 e STR2 são concatenadas, e o resultado é enviado a DST_10. Finalizado o bloco com sucesso, a saída DONE é ativada.

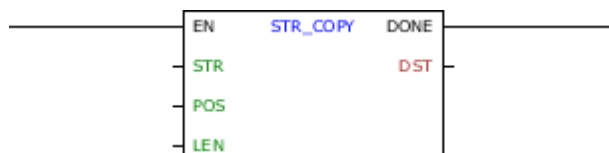


No exemplo acima, STR1 e STR2 são concatenadas, e o resultado é enviado a DST_6. Como o tamanho de DST_6 é 6, os dois últimos caracteres da concatenação são descartados. Finalizado o bloco com sucesso, a saída DONE é ativada.

10.1.5.1.23.3 STR_COPY

Bloco que realiza uma cópia de uma seção de STR, armazenando o resultado em DST.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR	STRING	STRING original
	POS	BYTE	Posição a partir da qual a cópia se iniciará
	LEN	BYTE	Número de caracteres que serão copiados a partir de POS
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	DST	BYTE	Variável que recebe a nova STRING

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de DST de acordo com os parâmetros de entrada. DST recebe um número de caracteres de STR1 definido por LEN a partir da posição inserida em POS.

Caso POS esteja fora da faixa de valores permitida (entre 1 e o tamanho de STR), DONE recebe FALSE e DST permanece inalterada.

Em caso de sucesso, o valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

**NOTA!**

POS é tratada com índice "base um". Ou seja, POS = 1 referencia a primeira posição de STR.

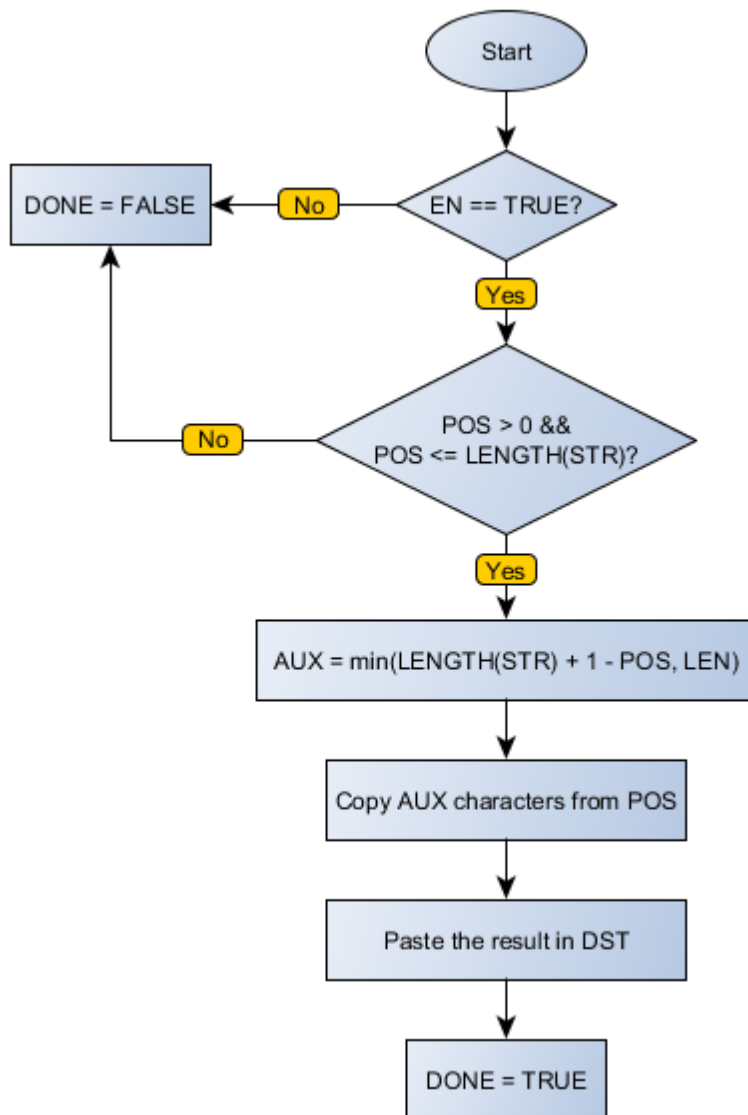
**NOTA!**

Caso o tamanho de DST seja menor que o número de caracteres copiados de STR, o valor resultante será truncado.

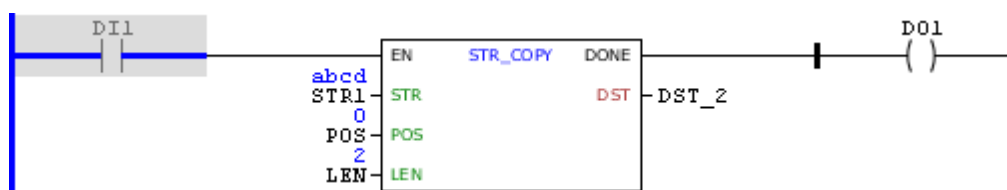
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

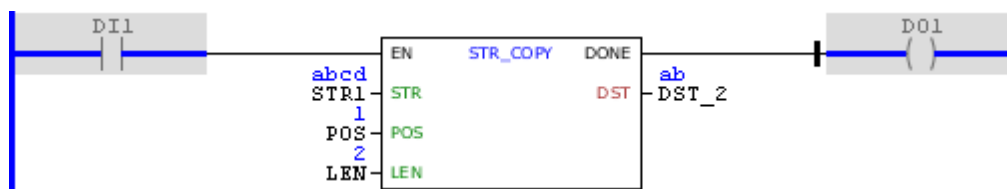
Fluxograma do Bloco



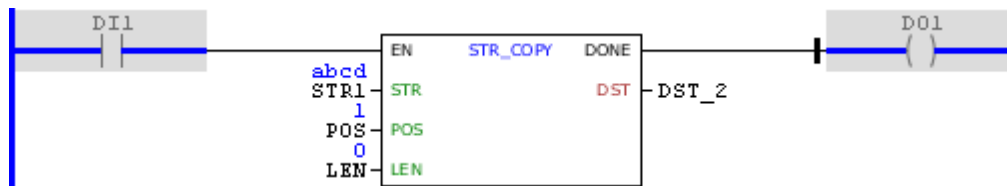
Exemplo



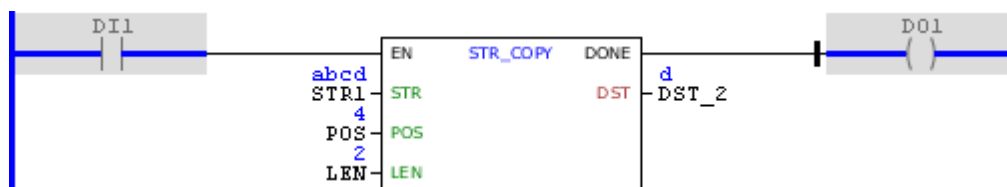
No exemplo acima, como o valor de POS é inválido, o bloco não é finalizado com sucesso, e a saída DONE é desativada.



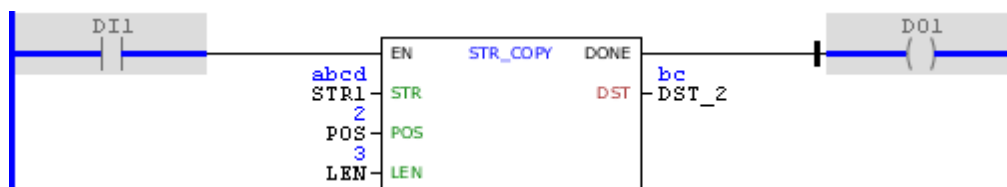
No exemplo acima, são copiados dois caracteres a partir da posição 1 de STR1, e o resultado é enviado a DST_2. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, são copiados zero caracteres a partir da posição 1 de STR1, e o resultado é enviado a DST_2. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que, quando LEN é zero, a saída é sempre construída como uma STRING nula.



No exemplo acima, são copiados dois caracteres a partir da posição 4 de STR1, e o resultado é enviado a DST_2. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que, se POS + LEN for maior que o tamanho de STR1, são copiados apenas os caracteres restantes de STR1, sem gerar erro.

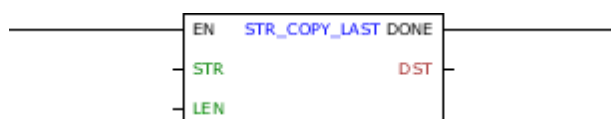


No exemplo acima, são copiados três caracteres a partir da posição 2 de STR1, e o resultado é enviado a DST_2. Como o tamanho de DST_2 é 2, o último caractere copiado é descartado. Finalizado o bloco com sucesso, a saída DONE é ativada.

10.1.5.1.23.4 STR_COPY_LAST

Bloco que realiza uma cópia de uma seção final de STR, armazenando o resultado em DST.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR	STRING	STRING original
	LEN	BYTE	Número de caracteres que serão copiados a partir do fim de STR
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	DST	BYTE	Variável que recebe a nova STRING

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de DST de acordo com os parâmetros de entrada. DST recebe um número de caracteres de STR1 definido por LEN a partir da posição final de STR, voltando em direção ao início.

O valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.



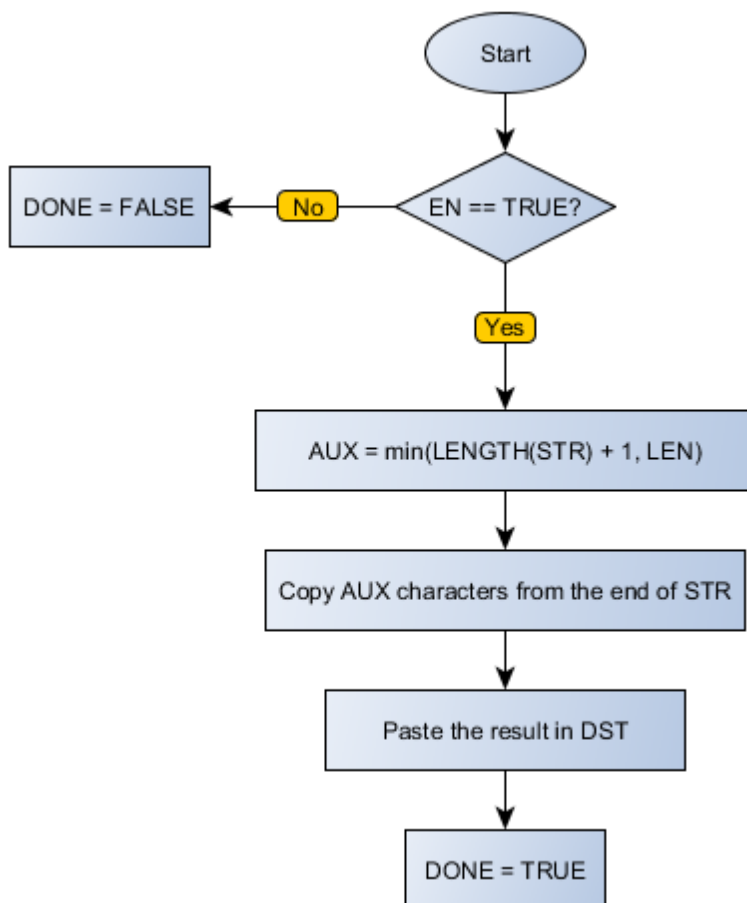
NOTA!

Caso o tamanho de DST seja menor que o número de caracteres copiados de STR, o valor resultante será truncado.

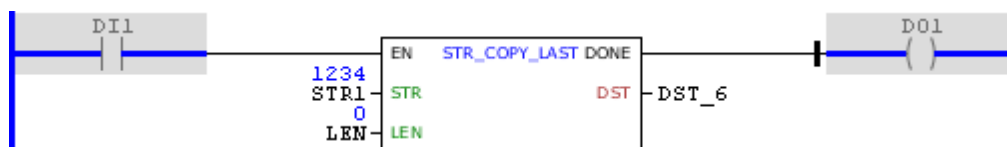
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

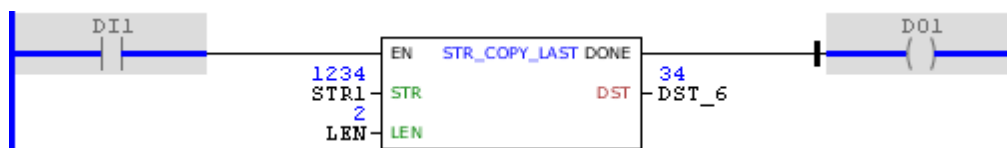
Fluxograma do Bloco



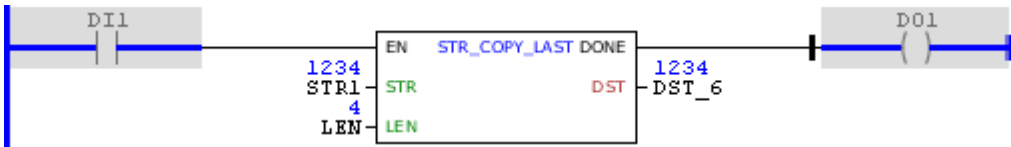
Exemplo



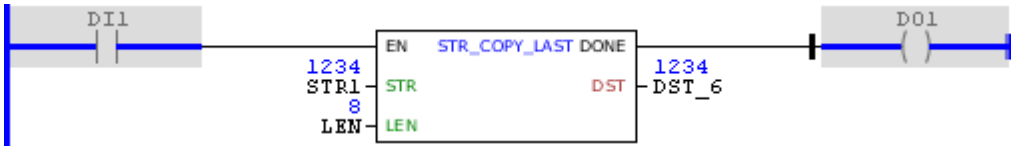
No exemplo acima, são copiados zero caracteres a partir da posição final de `STR1`, e o resultado é enviado a `DST_6`. Finalizado o bloco com sucesso, a saída `DONE` é ativada. Observe que, quando `LEN` é zero, a saída é sempre construída como uma `STRING` nula.



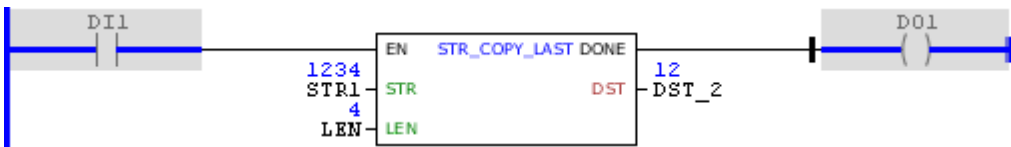
No exemplo acima, são copiados dois caracteres a partir da posição final de `STR1`, e o resultado é enviado a `DST_6`. Finalizado o bloco com sucesso, a saída `DONE` é ativada.



No exemplo acima, são copiados quatro caracteres a partir da posição final de STR1, e o resultado é enviado a DST_6. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, são copiados oito caracteres a partir da posição final de STR1, e o resultado é enviado a DST_6. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que, se LEN for maior que o tamanho de STR1, STR1 é copiada por completo, sem gerar erro.



No exemplo acima, são copiados quatro caracteres a partir da posição final de STR1, e o resultado é enviado a DST_2. Como o tamanho de DST_2 é 2, os dois últimos caracteres copiados são descartados. Finalizado o bloco com sucesso, a saída DONE é ativada.

10.1.5.1.23.5 STR_DELETE

Bloco que deleta parte de uma seção de STR, armazenando o restante em DST.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR	STRING	STRING original
	POS	BYTE	Posição a partir da qual a retirada se iniciará
	LEN	BYTE	Número de caracteres que serão retirados a partir de POS
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	DST	BYTE	Variável que recebe a nova STRING

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de DST de acordo com os parâmetros de entrada. É retirada de STR uma seção definida pela posição inicial POS e um tamanho LEN, sendo o resultado final armazenado em DST.

Caso POS esteja fora da faixa de valores permitida (entre 1 e o tamanho de STR), DONE recebe FALSE e DST permanece inalterada.

Em caso de sucesso, o valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

**NOTA!**

POS é tratada com índice "base um". Ou seja, POS = 1 referencia a primeira posição de STR.

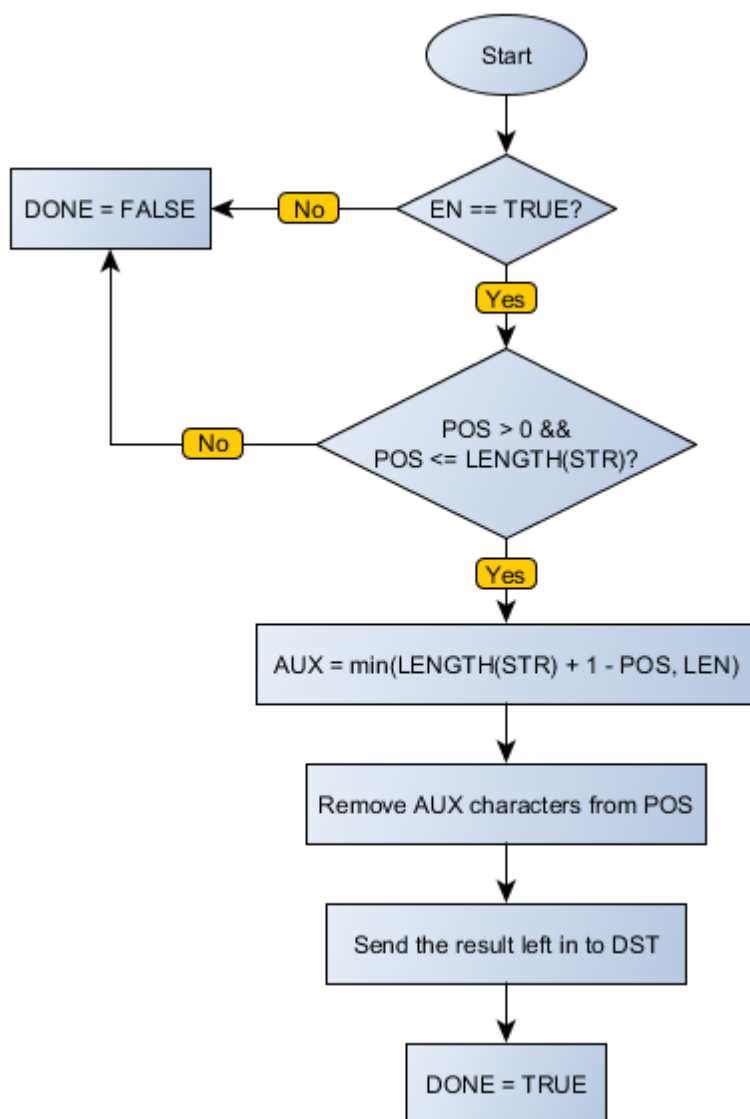
**NOTA!**

Caso o tamanho de DST seja menor que o número de caracteres restantes em STR, o valor resultante será truncado.

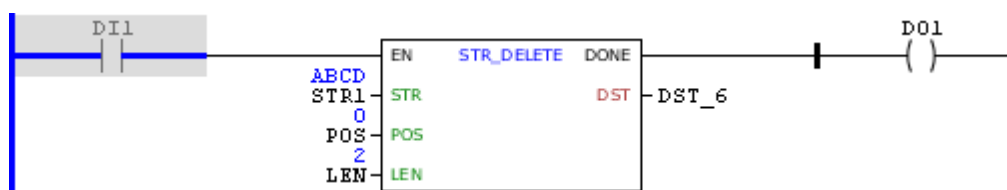
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

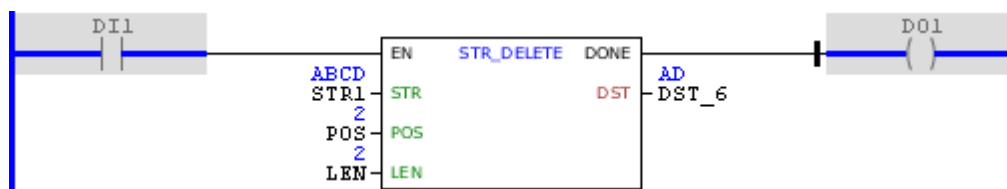
Fluxograma do Bloco



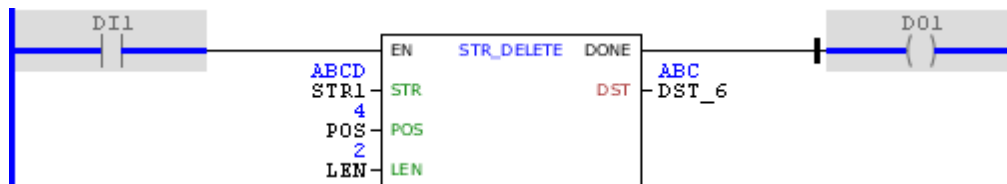
Exemplo



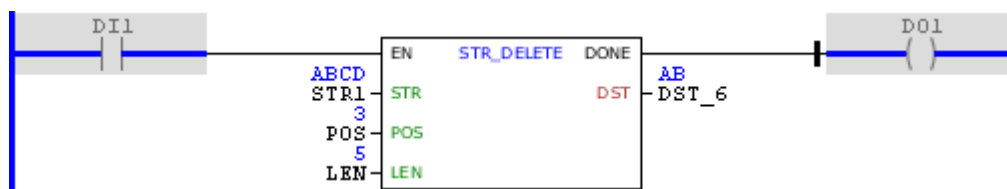
No exemplo acima, como o valor de POS é inválido, o bloco não é finalizado com sucesso, e a saída DONE é desativada.



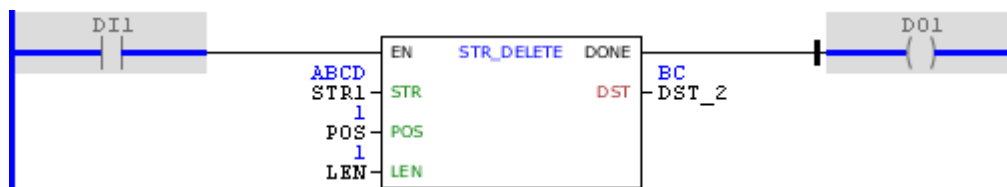
No exemplo acima, são deletados dois caracteres a partir da posição 2 de STR1, e os caracteres remanescentes são enviados a DST_6. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, são deletados dois caracteres a partir da posição 4 de STR1, e os caracteres remanescentes são enviados a DST_6. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que, se POS + LEN for maior que o tamanho de STR1, são deletados apenas os caracteres restantes de STR1, sem gerar erro.



No exemplo acima, são deletados cinco caracteres a partir da posição 3 de STR1, e os caracteres remanescentes são enviados a DST_6. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que, se POS + LEN for maior que o tamanho de STR1, são deletados apenas os caracteres restantes de STR1, sem gerar erro.

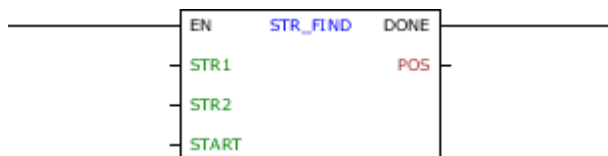


No exemplo acima, é deletado um caractere a partir da posição 1 de STR1, e os caracteres remanescentes são enviados a DST_2. Como o tamanho de DST_2 é 2, o último caractere copiado é descartado. Finalizado o bloco com sucesso, a saída DONE é ativada.

10.1.5.1.23.6 STR_FIND

Bloco que busca a primeira ocorrência de uma STRING em outra, retornando a posição dessa ocorrência.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR1	STRING	STRING onde será realizada a busca
	STR2	STRING	STRING a ser buscada
	START	BYTE	Posição inicial de busca
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	POS	BYTE	Posição onde STR2 foi encontrada

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de POS de acordo com os parâmetros de entrada.

É realizada uma busca em STR1, a partir da posição START, por uma ocorrência de STR2.

Caso ela seja encontrada, insere em POS a posição da ocorrência, enquanto DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

Caso ela não seja encontrada, ou START receba valor zero, DONE permanece em FALSE e POS permanece inalterada.



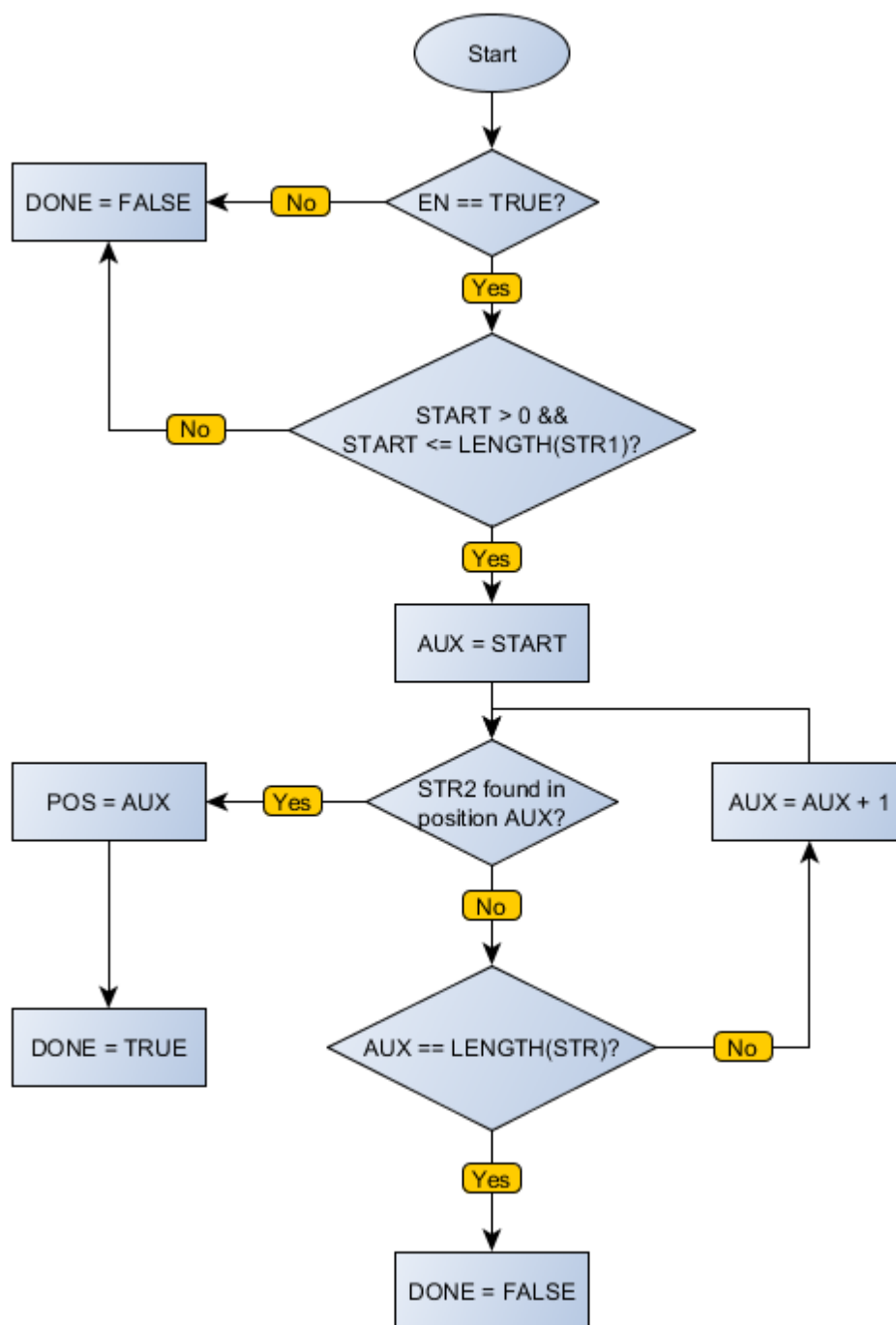
NOTA!

POS é tratada com índice "base um". Ou seja, POS = 1 referencia a primeira posição de STR.

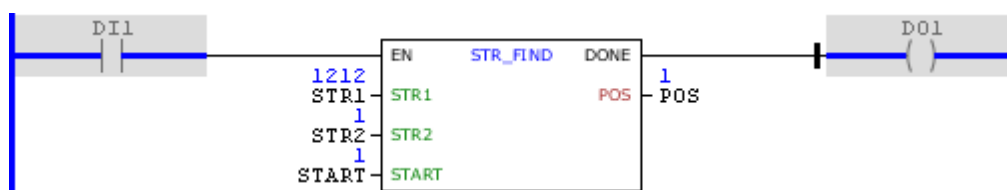
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

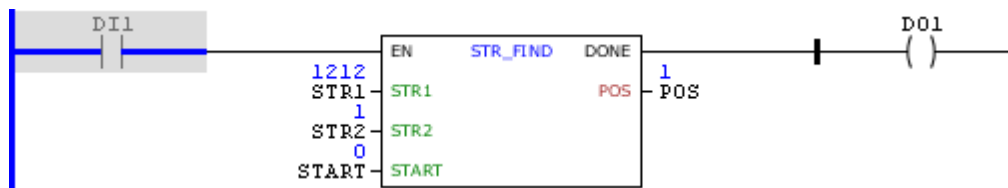
Fluxograma do Bloco



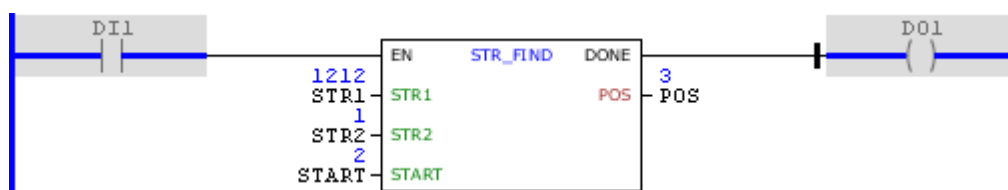
Exemplo



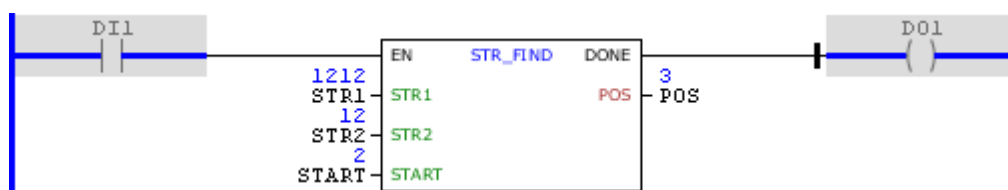
No exemplo acima, é feita uma busca por STR2 em STR1 a partir da posição 1. Como STR2 é encontrada na posição 1, POS recebe 1. Finalizado o bloco com sucesso, a saída DONE é ativada.



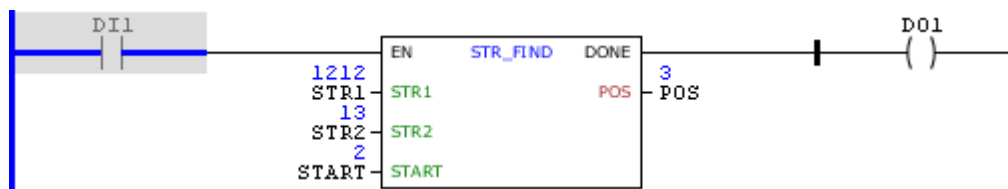
No exemplo acima, como o valor de START é inválido, o bloco não é finalizado com sucesso, e a saída DONE é desativada.



No exemplo acima, é feita uma busca por STR2 em STR1 a partir da posição 2. Como STR2 é encontrada na posição 3, POS recebe 3. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, é feita uma busca por STR2 em STR1 a partir da posição 2. Como STR2 é encontrada na posição 3, POS recebe 3. Finalizado o bloco com sucesso, a saída DONE é ativada.

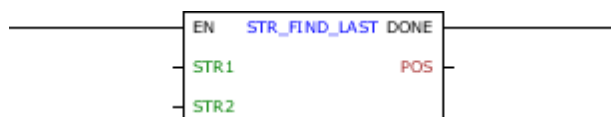


No exemplo acima, é feita uma busca por STR2 em STR1 a partir da posição 2. Como STR2 não é encontrada, POS permanece inalterada e a saída DONE é desativada.

10.1.5.1.23.7 STR_FIND_LAST

Bloco que busca a última ocorrência de uma STRING em outra, retornando a posição dessa ocorrência.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR1	STRING	STRING onde será realizada a busca
	STR2	STRING	STRING a ser buscada
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	POS	BYTE	Posição onde STR2 foi encontrada

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de POS de acordo com os parâmetros de entrada.

É realizada uma busca em STR1, a partir da sua última posição, por uma ocorrência de STR2.

Caso ela seja encontrada, insere em POS a posição dessa ocorrência, enquanto DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

Caso ela não seja encontrada, DONE permanece em FALSE e POS permanece inalterada.



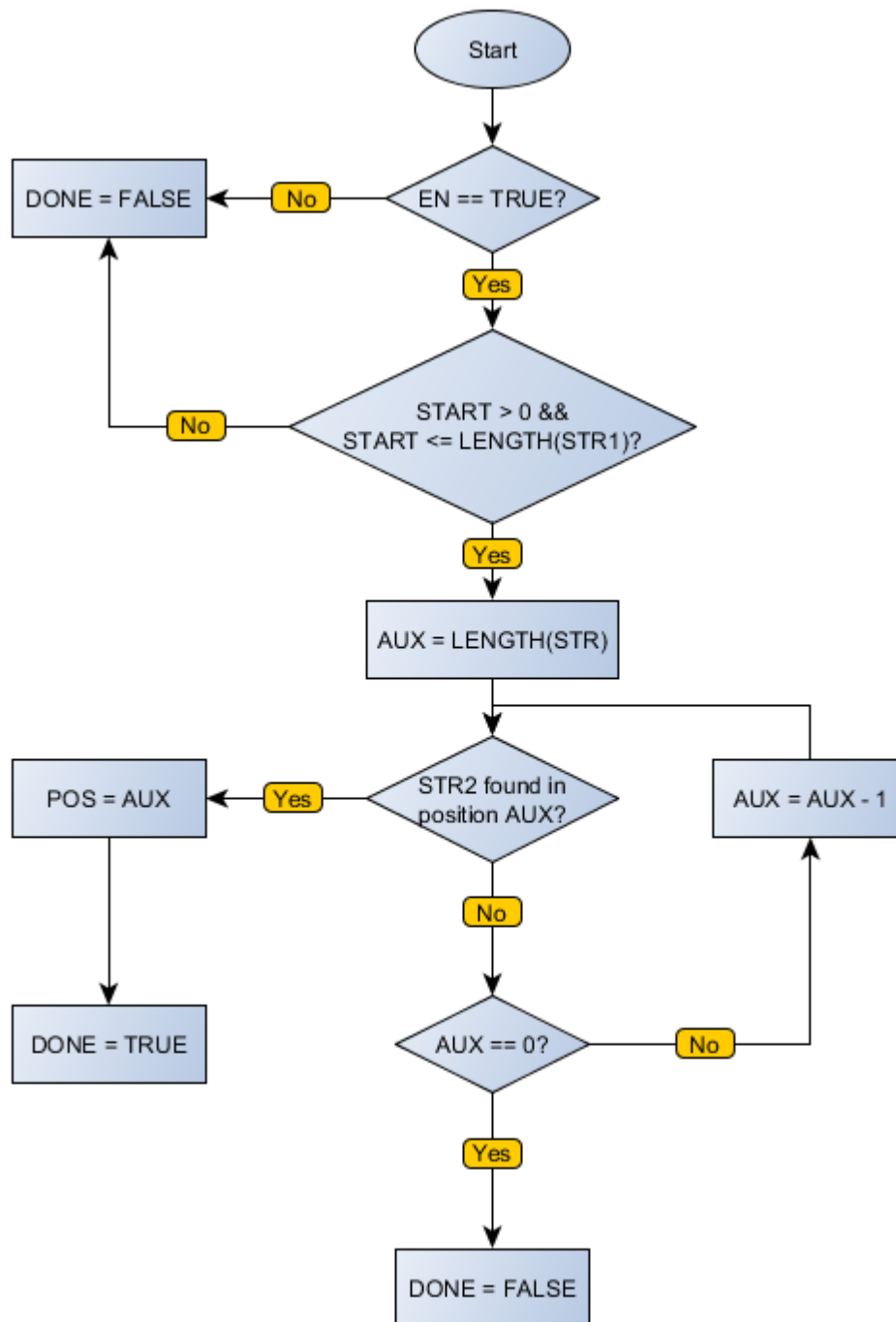
NOTA!

POS é tratada com índice "base um". Ou seja, POS = 1 referencia a primeira posição de STR.

Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

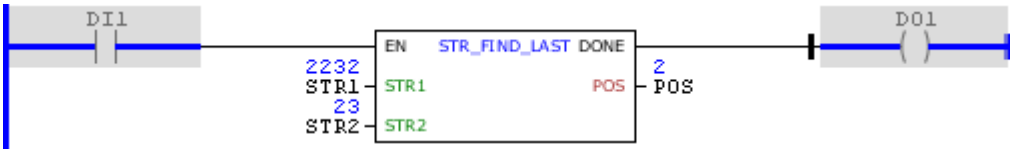
Fluxograma do Bloco



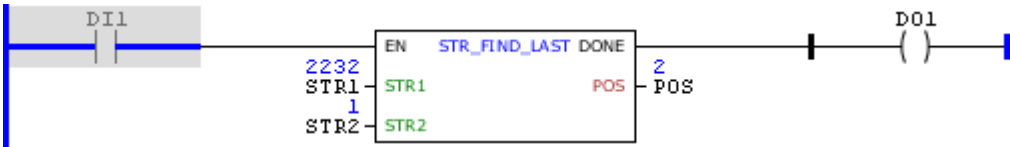
Exemplo



No exemplo acima, é feita uma busca por STR2 em STR1 a partir da sua posição final. Como STR2 é encontrada na posição 4, POS recebe 4. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, é feita uma busca por STR2 em STR1 a partir da sua posição final. Como STR2 é encontrada na posição 2, POS recebe 4. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, é feita uma busca por STR2 em STR1 a partir da sua posição final. Como STR2 não é encontrada, POS permanece inalterada e a saída DONE é desativada.

10.1.5.1.23.8 STR_INSERT

Bloco que insere uma STRING em determinada posição de outra, retornando a STRING resultante.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	STR1	STRING	STRING onde será realizada a inserção
	STR2	STRING	STRING a ser inserida
	POS	BYTE	Posição onde STR2 será inserida
VAR_OUTPUT	DONE	B00L	Habilitação de saída
	DST	STRING	STRING resultante

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de DST de acordo com os parâmetros de entrada.

É realizada uma inserção em STR1, a partir da posição definida em POS, do conteúdo de STR2.

Caso POS esteja fora da faixa de valores permitida (entre 1 e o tamanho de STR mais um), DONE recebe FALSE e DST permanece inalterada.

Em caso de sucesso, o valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

**NOTA!**

POS é tratada com índice "base um". Ou seja, POS = 1 referencia a primeira posição de STR.

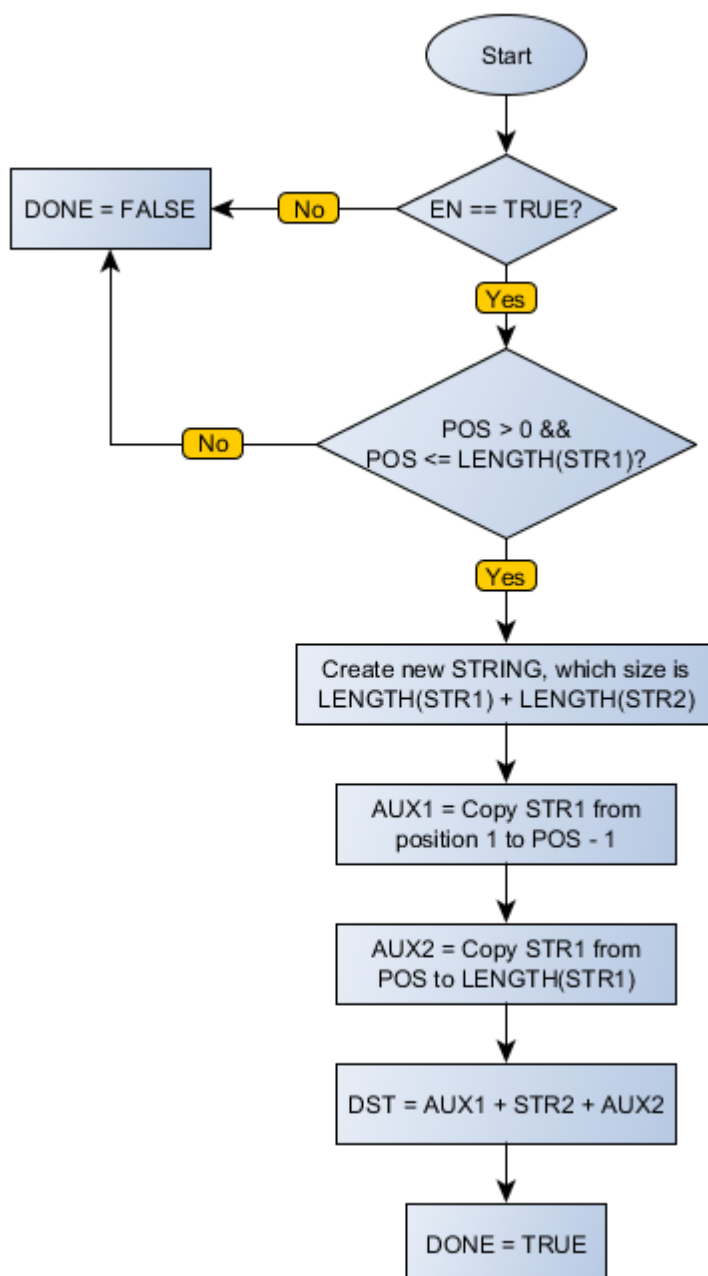
**NOTA!**

Caso o tamanho de DST seja menor que a soma do número de caracteres de STR1 e STR2, o valor resultante será truncado.

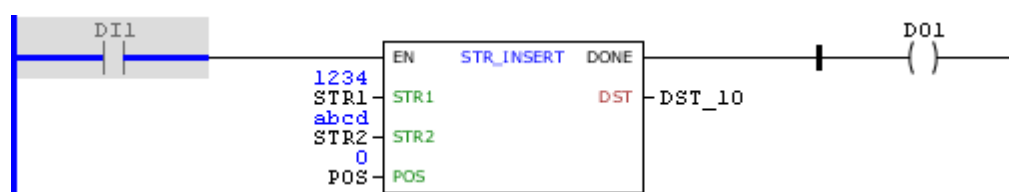
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

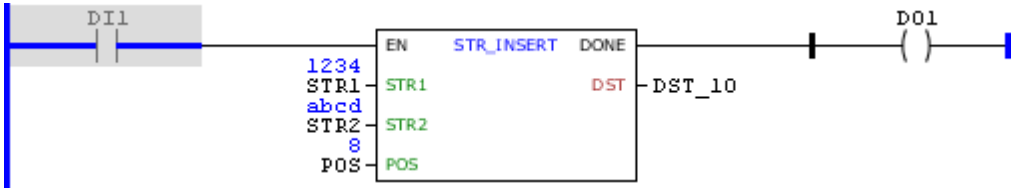
Fluxograma do Bloco



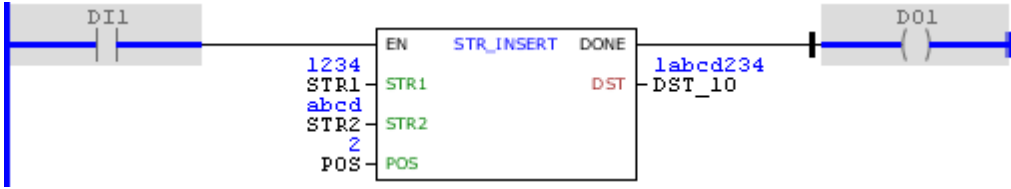
Exemplo



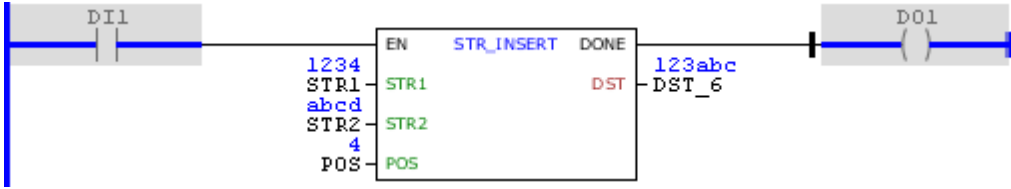
No exemplo acima, como o valor de POS é inválido, o bloco não é finalizado com sucesso, e a saída DONE é desativada.



No exemplo acima, como o valor de POS é inválido, o bloco não é finalizado com sucesso, e a saída DONE é desativada.



No exemplo acima, é feita uma inserção de STR2 em STR1 a partir da posição 2, e o resultado é enviado a DST_10. Finalizado o bloco com sucesso, a saída DONE é ativada.

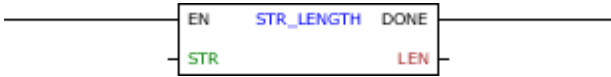


No exemplo acima, é feita uma inserção de STR2 em STR1 a partir da posição 4, e o resultado é enviado a DST_6. Como o tamanho de DST_6 é 6, os últimos dois caracteres copiados são descartados. Finalizado o bloco com sucesso, a saída DONE é ativada.

10.1.5.1.23.9 STR_LENGTH

Bloco que calcula o tamanho de uma STRING.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR1	STRING	STRING de entrada
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	LEN	BYTE	Tamanho da STRING

Funcionamento

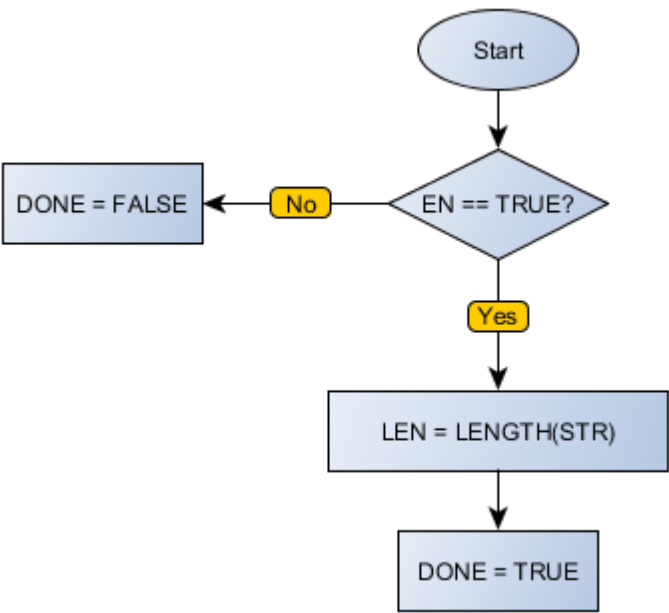
Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de LEN de acordo com o tamanho identificado de STR.

O valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

Fluxograma do Bloco



Exemplo



No exemplo acima, é calculado o comprimento da STRING em STR e o resultado é enviado a LEN. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, é calculado o comprimento da STRING em STR e o resultado é enviado a LEN. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que STRINGS nulas possuem comprimento zero.

10.1.5.1.23.10 STR_REPLACE

Bloco que substituiu uma seção de uma STRING por outra STRING, retornando a STRING resultante.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	STR1	STRING	STRING onde será realizada a substituição
	STR2	STRING	STRING a ser inserida
	POS	BYTE	Posição onde STR2 será inserida
	LEN	BYTE	Número de caracteres a serem substituídos por STR2 em STR1
VAR_OUTPUT	DONE	BOOL	Habilitação de saída
	DST	STRING	STRING resultante

Funcionamento

Este bloco permanece ativo enquanto EN está em nível TRUE, atualizando o valor de DST de acordo com os parâmetros de entrada.

A partir da posição definida em POS, são deletados um número de caracteres definidos por LEN. Após isso, nesta mesma posição, é inserido o conteúdo de STR2.

Caso POS esteja fora da faixa de valores permitida (entre 1 e o tamanho de STR mais um), DONE recebe FALSE e DST permanece inalterada.

Em caso de sucesso, o valor de DONE repassa para o próximo bloco Ladder o valor de EN ao finalizar a operação.

**NOTA!**

POS é tratada com índice "base um". Ou seja, POS = 1 referencia a primeira posição de STR.

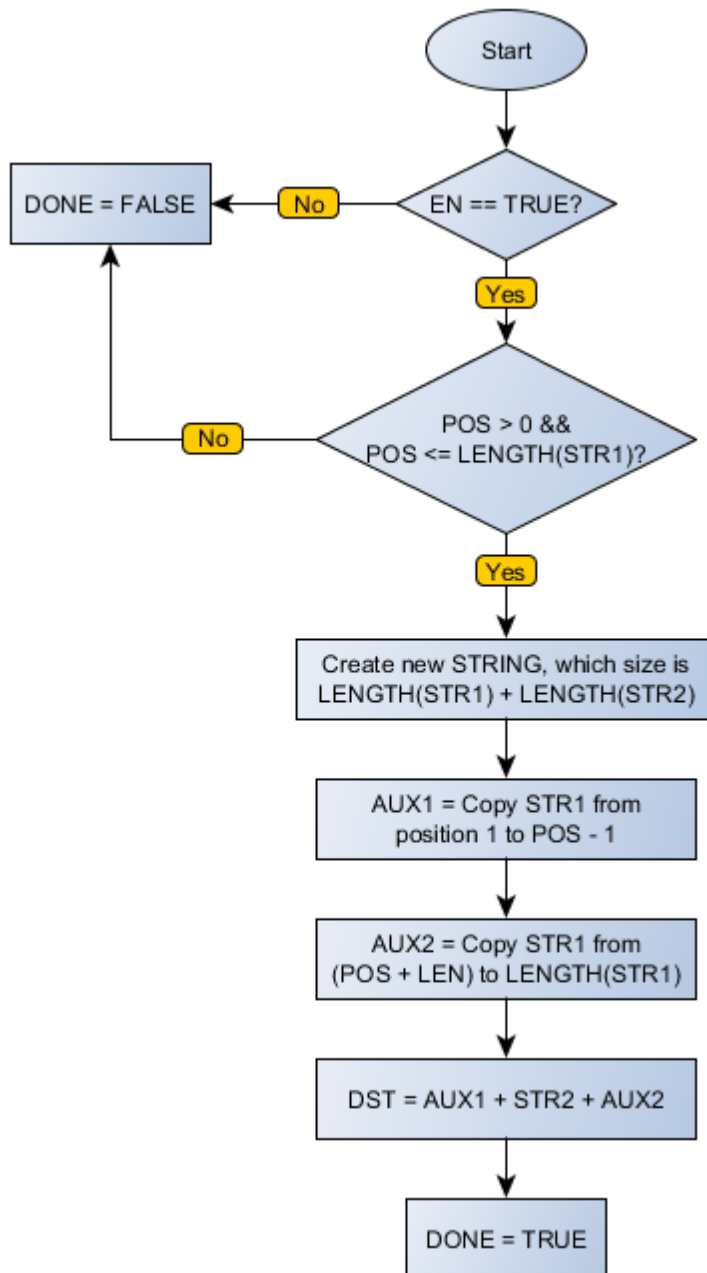
**NOTA!**

Caso o tamanho de DST seja menor que a soma do número de caracteres restantes de STR1 e STR2, o valor resultante será truncado.

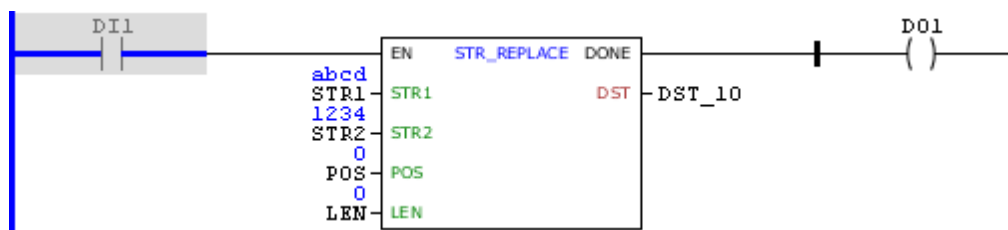
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior

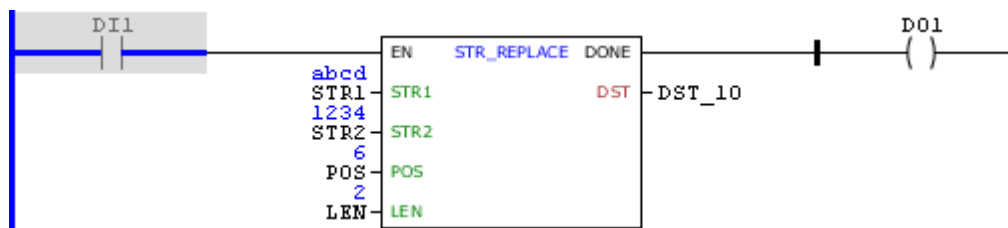
Fluxograma do Bloco



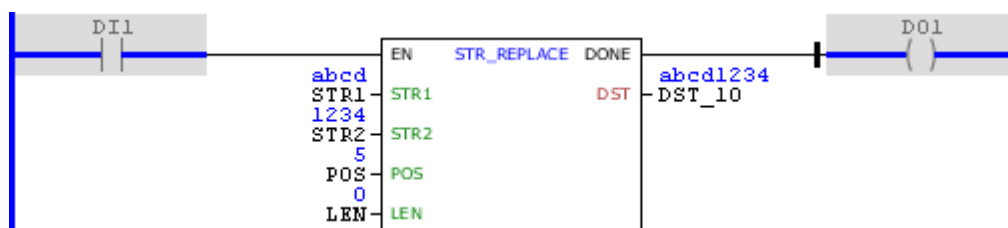
Exemplo



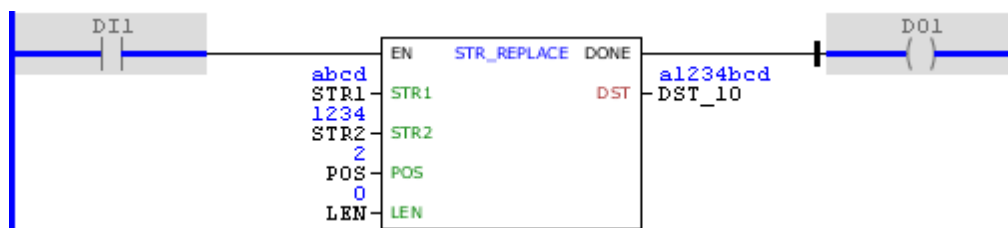
No exemplo acima, como o valor de POS é inválido, o bloco não é finalizado com sucesso, e a saída DONE é desativada.



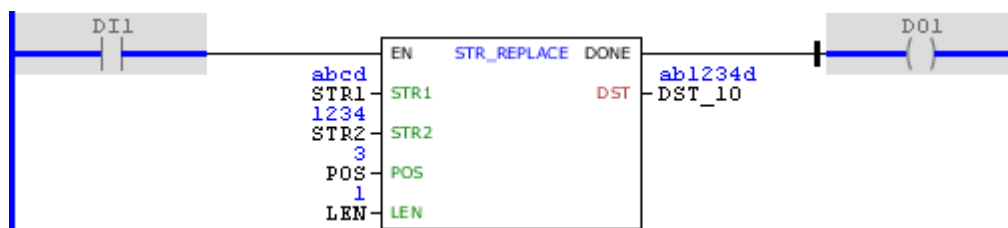
No exemplo acima, como o valor de POS é inválido, o bloco não é finalizado com sucesso, e a saída DONE é desativada.



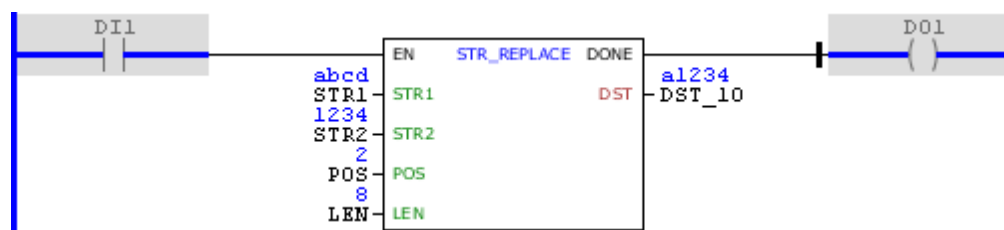
No exemplo acima, é feita uma substituição de zero caracteres a partir da posição 5 de STR1 pelo conteúdo de STR2, e o resultado é enviado a DST_10. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que este bloco atuou como um STR_INSERT.



No exemplo acima, é feita uma substituição de zero caracteres a partir da posição 2 de STR1 pelo conteúdo de STR2, e o resultado é enviado a DST_10. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que este bloco atuou como um STR_INSERT.



No exemplo acima, é feita uma substituição de um caractere a partir da posição 3 de STR1 pelo conteúdo de STR2, e o resultado é enviado a DST_10. Finalizado o bloco com sucesso, a saída DONE é ativada.



No exemplo acima, é feita uma substituição de oito caracteres a partir da posição 2 de STR1 pelo conteúdo de STR2, e o resultado é enviado a DST_10. Finalizado o bloco com sucesso, a saída DONE é ativada. Observe que, se POS + LEN for maior que o tamanho de STR1, são substituídos apenas os caracteres restantes de STR1, sem gerar erro.

10.1.5.2 Tarefas

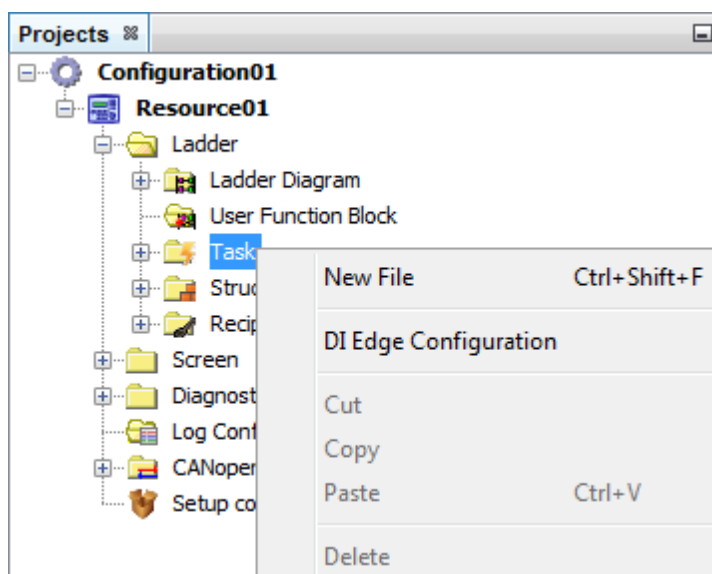
Visão Geral

A tarefa é um mecanismo de escalonamento muito útil em sistemas de tempo real para controlar a execução de programas, de forma periódica ou disparada por eventos.

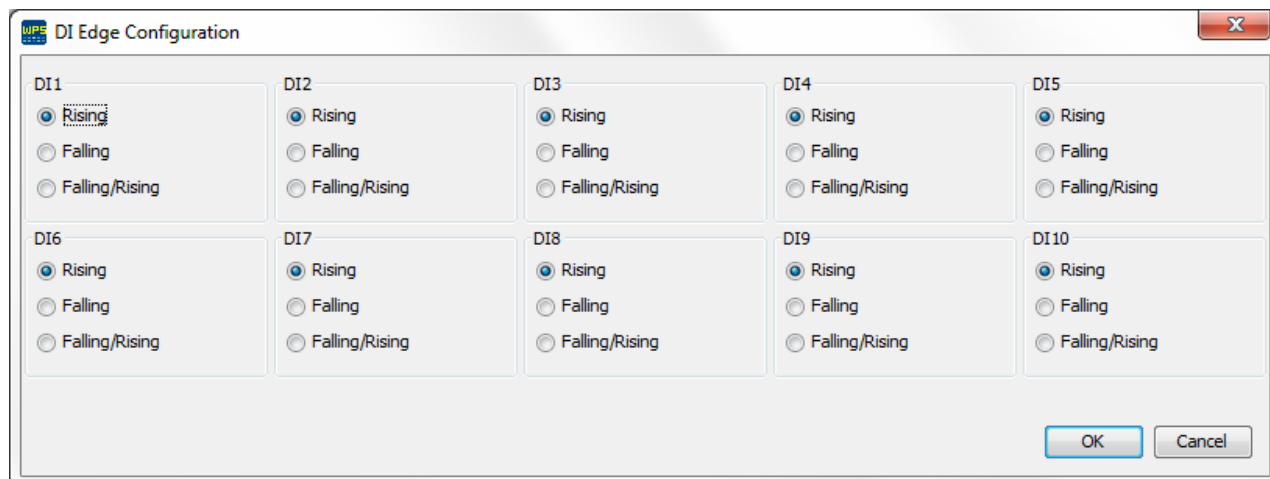
A configuração das tarefas é realizada adicionando arquivos de tarefas à pasta **Tarefa**, dentro do recurso. A estes arquivos de tarefa, são então associados programas que executam sua lógica, quando a tarefa se torna ativa. Para cada tarefa pode ser configurado um watchdog (ver seção *Watchdog*). Cada tarefa possui um marcador de sistema associado que pode ser habilitado ou desabilitado através do programa.

Dentro de um recurso, existe uma única tarefa chamada **Main**, que não pode ser apagada, na qual o programa principal (**Main Ladder**) é executado.

No menu de opções da pasta de tarefas, é possível realizar as seguintes ações:

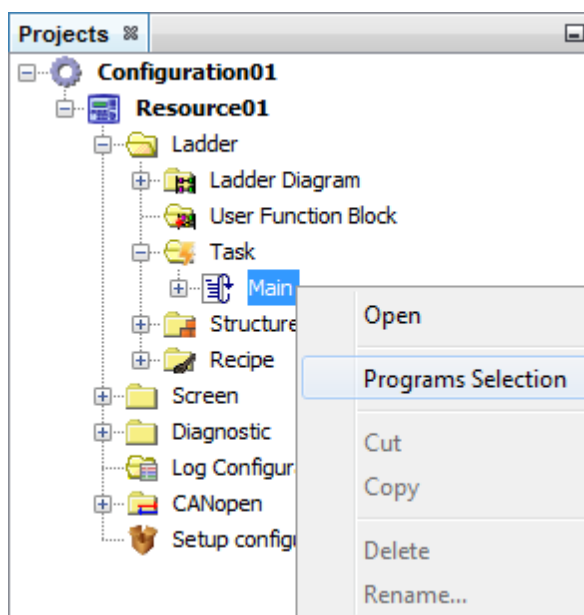


- **Adicionar uma nova tarefa:** para adicionar uma nova tarefa, é preciso selecionar a opção **Novo arquivo** da pasta **Tarefa**.
- **Configuração da borda de transição das entradas digitais (DIs):** as entradas digitais DI9 e DI10 possuem a opção de disparar eventos através da transição da borda de subida, descida ou subida e descida. A configuração destas bordas de transição é feita através da janela de configuração das DIs, que é acessada selecionando a opção **Configura transição DI** da pasta de **Tarefa**.



Estas DIs são utilizadas nas tarefas do tipo evento externo e contagem.

Através do menu de opções dos arquivos das tarefas, é possível realizar as seguintes ações:



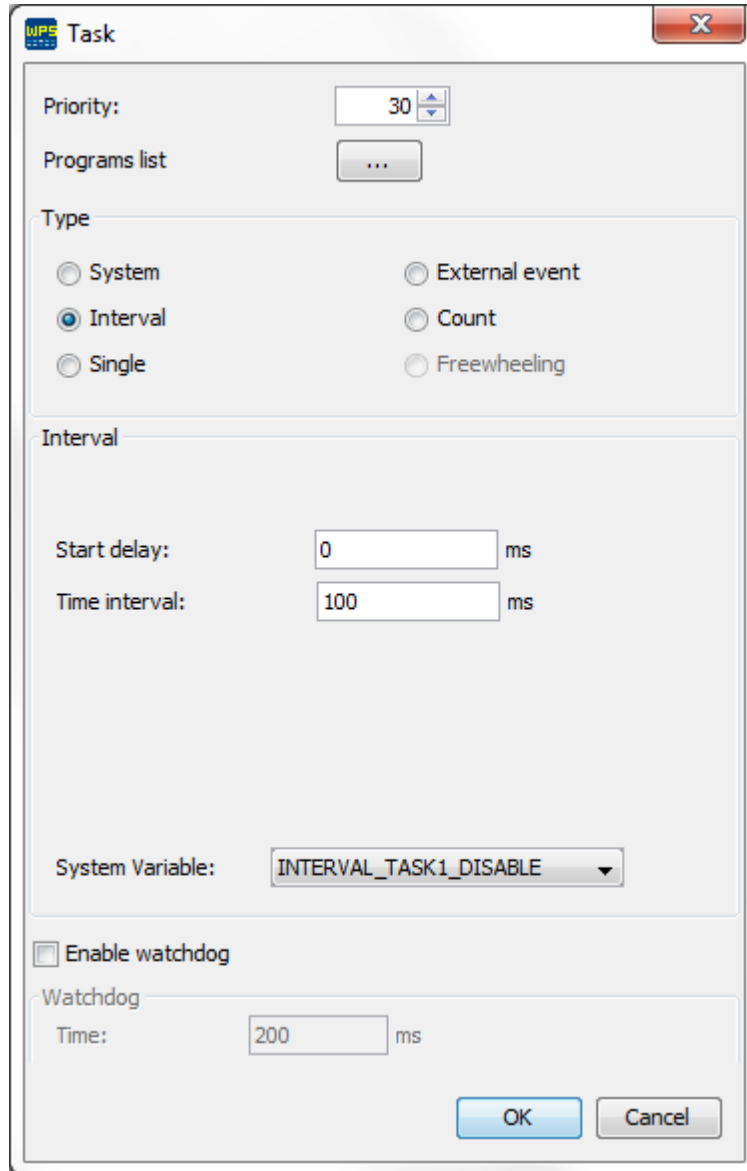
- **Configurar tarefas:** para configurar a tarefa, é preciso selecionar a opção **Abrir**, no arquivo de tarefas que se deseja configurar. Para mais informações de como configurar as tarefas, ver seção *Configuração das tarefas*.
- **Adicionar, remover ou ordenar programas da tarefa:** para adicionar, remover ou ordenar programas de uma determinada tarefa, é preciso selecionar a opção **Seleciona programas** no arquivo de tarefas que

deseja configurar. Para mais informações de como selecionar os programas, ver na seção *Configuração das tarefas* o item *Lista de programas*.

Configuração de Tarefas

Na janela de configuração das tarefas, são configurados:

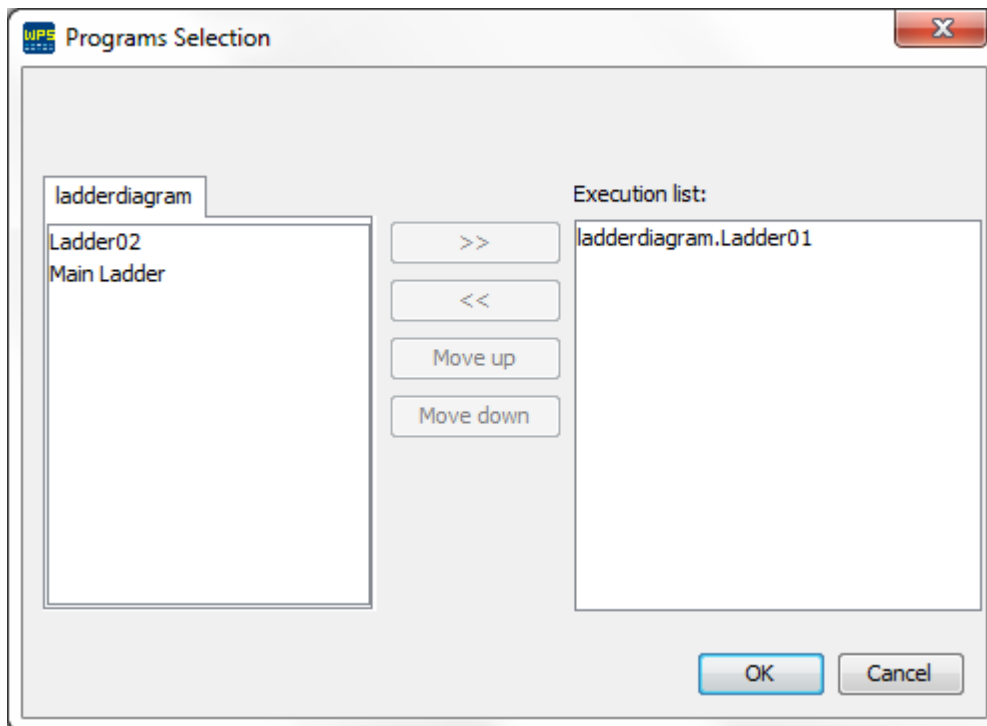
- Prioridade,
- Sequência dos programas,
- Tipo de tarefa, e
- Opções de watchdog da tarefa.



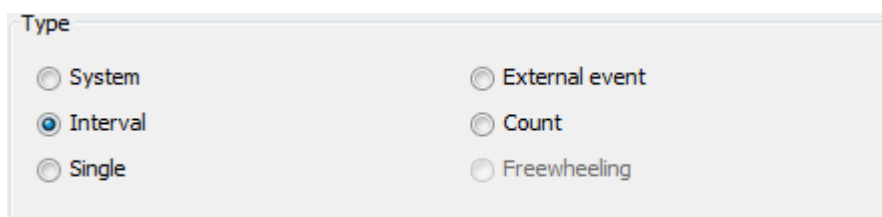
- **Prioridade:** configurada com um número entre 0 e 30 (0 - prioridade mais alta, 30 - mais baixa) que define a prioridade em que a tarefa será executada. Caso uma tarefa de prioridade mais alta do que a tarefa em execução seja acionada, esta imediatamente entrará em execução. De outra forma, caso uma tarefa de

prioridade mais baixa do que a atual seja acionada, esta aguarda o final da execução da tarefa atual para entrar em execução.

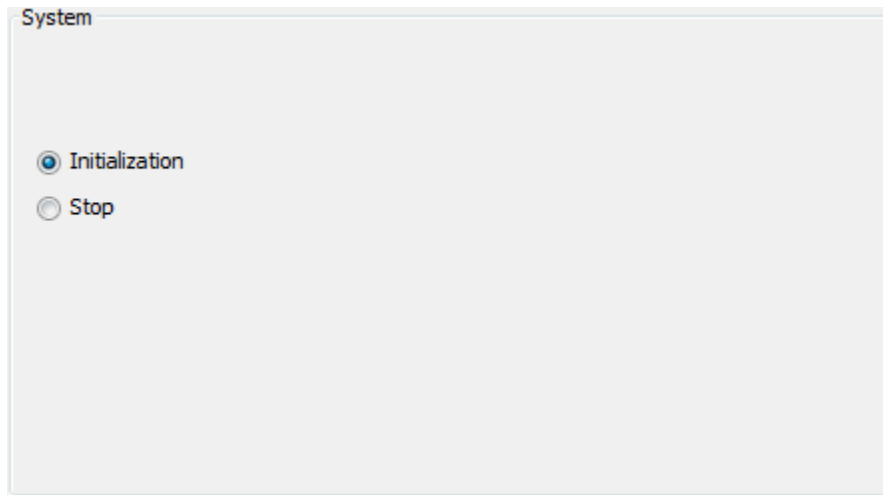
- **Lista de programas:** a janela de seleção da sequência dos programas serve para adicionar, remover ou alterar a sequência dos programas associados a determinada tarefa. Na lista do lado esquerdo, estão todos os programas disponíveis no recurso e no lado direito, os programas selecionados para esta tarefa. A ordem que os programas entrarão em execução será a mesma ordem definida nesta lista.



- **Tipos de Tarefa:** As tarefas estão divididas em: *system*, *interval*, *single*, *external event*, *count* e *freewheeling*.

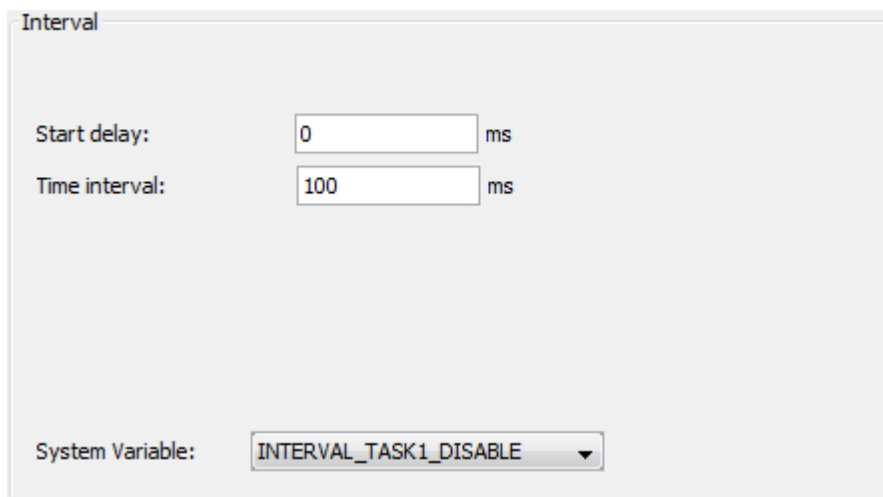


- o **System:** Esta tarefa pode ser do tipo Inicialização ou Parada. Quando selecionada a opção **Inicialização**, a tarefa entrará em execução logo que o programa do usuário for executado pela primeira vez. Caso a opção **Parada** seja selecionada, os programas associados a esta tarefa, entram em execução logo após a parada do programa do usuário.



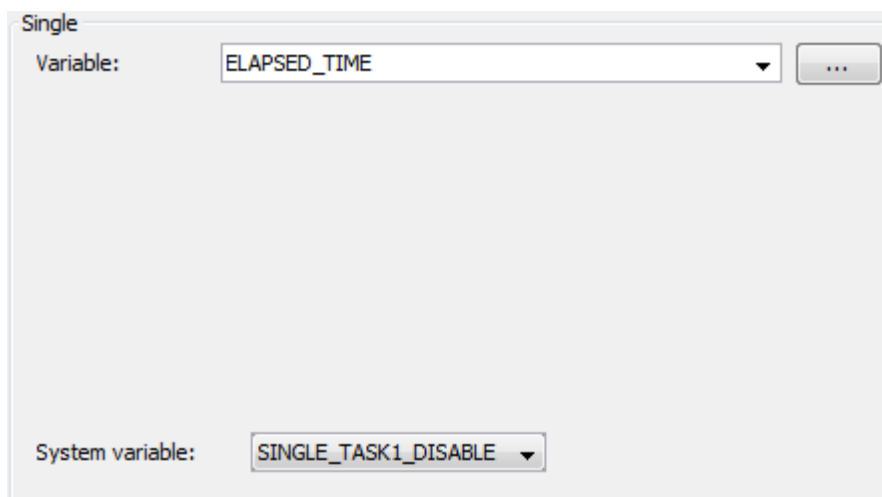
The 'System' dialog box contains two radio buttons. The 'Initialization' radio button is selected, indicated by a filled circle. The 'Stop' radio button is unselected, indicated by an empty circle.

- o **Interval:** Os programas associados a esta tarefa, são executados repetidas vezes, no intervalo de tempo definido no campo **Intervalo de tempo**, com o tempo de atraso para a primeira execução definido no campo **Atraso inicial**. A esta tarefa é associada uma variável de sistema que permite que a tarefa seja desabilitada através do programa.



The 'Interval' dialog box contains two input fields and a dropdown menu. The 'Start delay:' field has the value '0' and is followed by 'ms'. The 'Time interval:' field has the value '100' and is followed by 'ms'. The 'System Variable:' dropdown menu is set to 'INTERVAL_TASK1_DISABLE'.

- o **Single:** Os programas associados a esta tarefa, são executados uma única vez, sempre que a variável selecionada sofrer uma transição positiva, ou seja, de zero para algum outro valor. A esta tarefa é associada uma variável de sistema que permite que a tarefa seja desabilitada através do programa.

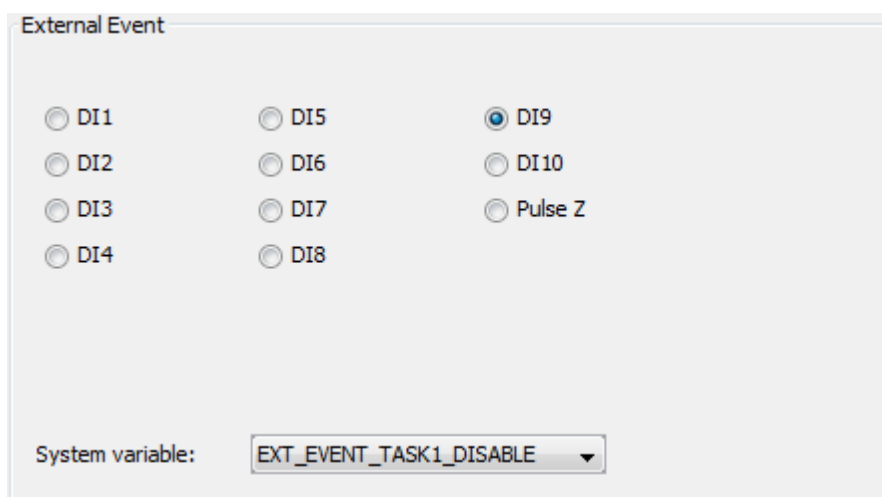


Single

Variable:

System variable:

- o **External Event:** Os programas associados a esta tarefa, são executados uma única vez, sempre que o evento selecionado for ativado. Os eventos das entradas digitais possuem a opção de selecionar a transição que ativa a tarefa (veja *Configuração da borda de transição das entradas digitais* na seção *Visão Geral*). A esta tarefa é associada uma variável de sistema que permite que a tarefa seja desabilitada através do programa.



External Event

☐ DI1 ☐ DI5 ☒ DI9
☐ DI2 ☐ DI6 ☐ DI10
☐ DI3 ☐ DI7 ☐ Pulse Z
☐ DI4 ☐ DI8

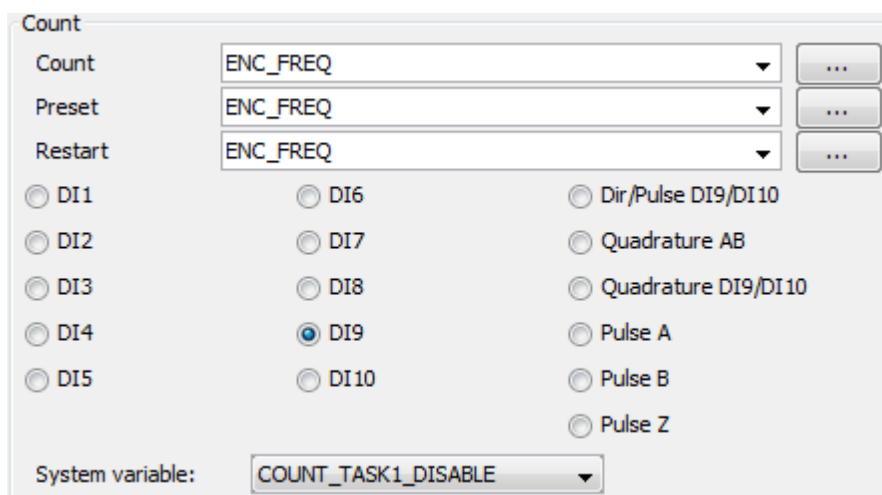
System variable:

**NOTA!**

As opções DI1 a DI8 estão disponíveis a partir da versão 2.10.

- o **Count:** Os programas associados a esta tarefa, são executados toda vez que a contagem de pulsos na entrada selecionada (Entrada Digital, Pulso A, Pulso B, Pulso Z, Quadratura AB, Quadratura DI9/DI10, Dir/Pulso DI9/DI10) for maior do que o valor da variável definida no campo **Predefinido**. O valor de contagem dos pulsos é armazenado na variável **Contagem**. O valor da contagem dos pulsos é reiniciado quando este valor ultrapassar o valor definido no campo **Reinício**. As variáveis configuradas nestes campos podem ser do tipo DWORD e UDINT para as Entradas Digitais, Pulso A, Pulso B e Pulso Z e DINT para entrada quadratura AB. A esta tarefa é associada uma variável de sistema que permite que a tarefa seja desabilitada através do programa. A opção Dir/Pulso_DI9/DI10 pode ser utilizada como

contagem rápida de até 100KHz.



Importante:

As funções Quadrature_DI9/DI10 e Dir/Pulse_DI9/DI10 não poderão ser utilizadas simultaneamente em tarefas ou em tarefa e bloco;

As funções Quadrature_DI9/DI10 e Dir/Pulse_DI9/DI10 poderão ser utilizadas em blocos diferentes, desde que não sejam utilizadas em tarefas.



NOTA!

Estas opções após selecionadas não causarão erro caso as outras (DI9, DI10, Pulso A...) também sejam selecionadas.

Exemplos que não geram erro de compilação:

- Dois blocos em Quadrature_DI9/DI10;
- Dois blocos, sendo um bloco Quadrature_DI9/DI10 e outro Dir/Pulse_DI9/DI10;
- Duas tarefas Quadrature_DI9/DI10 ou duas tarefas Dir/Pulse_DI9/DI10;

Exemplos que geram erro de compilação:

- Duas tarefas, sendo uma Quadrature_DI9/DI10 e outra Dir/Pulse_DI9/DI10;
- Tarefa Quadrature_DI9/DI10 e bloco Dir/Pulse_DI9/DI10;
- Tarefa Dir/Pulse_DI9/DI10 e bloco Quadrature_DI9/DI10.



NOTA!

As opções Quadrature_DI9/DI10 e Dir/Pulse_DI9/DI10 estão disponíveis a partir da versão 2.00.

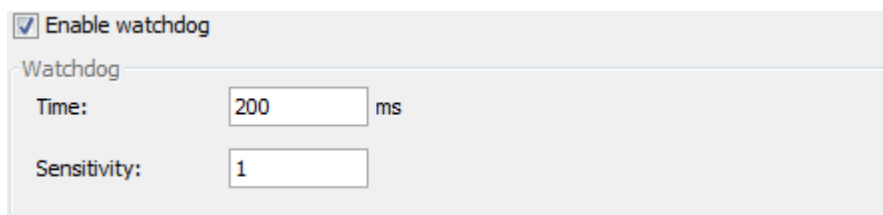


NOTA!

As opções DI1 a DI8 estão disponíveis a partir da versão 2.10.

- o **Freewheeling:** Os programas associados a esta tarefa são executados de forma cíclica. No final da execução da lista de programas, a lista de programas é reiniciada, até que o programa do usuário seja parado. O programa principal (Main Ladder) está associado a esta tarefa e não é possível removê-lo ou associá-lo a outra tarefa.

- **Watchdog:** Quando a opção watchdog está ativa, o programa do usuário é parado com alarme de erro, caso o tempo de execução da tarefa demore mais do que o tempo definido no campo **Tempo**, por um número de vezes maior que o definido no campo **Sensibilidade**.



- o **Tempo:** Tempo máximo para execução da tarefa, sem que ocorra o erro de watchdog.
- o **Sensibilidade:** Número de erros de watchdog permitidos, antes da geração do alarme de erro e parada do programa do usuário.

Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior

10.1.5.3 Estruturas

Uma estrutura consiste num agrupamento de dados utilizada para definir uma receita ou um objeto.

No programa Ladder é possível criar variáveis do tipo estrutura e utilizá-las nos blocos. Para acesso aos membros internos da estrutura, utiliza-se o '.' seguido do seu respectivo membro.

Criando uma Estrutura

1. Com o botão direito do mouse sobre a pasta **Estrutura**, clique em **Novo Arquivo**.

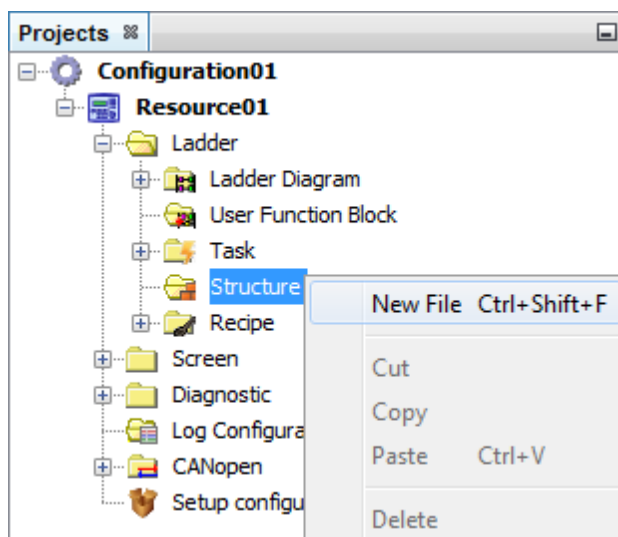


Figura 1 - Criando uma Estrutura

2. Defina o nome do arquivo e aperte o botão **Próximo**.

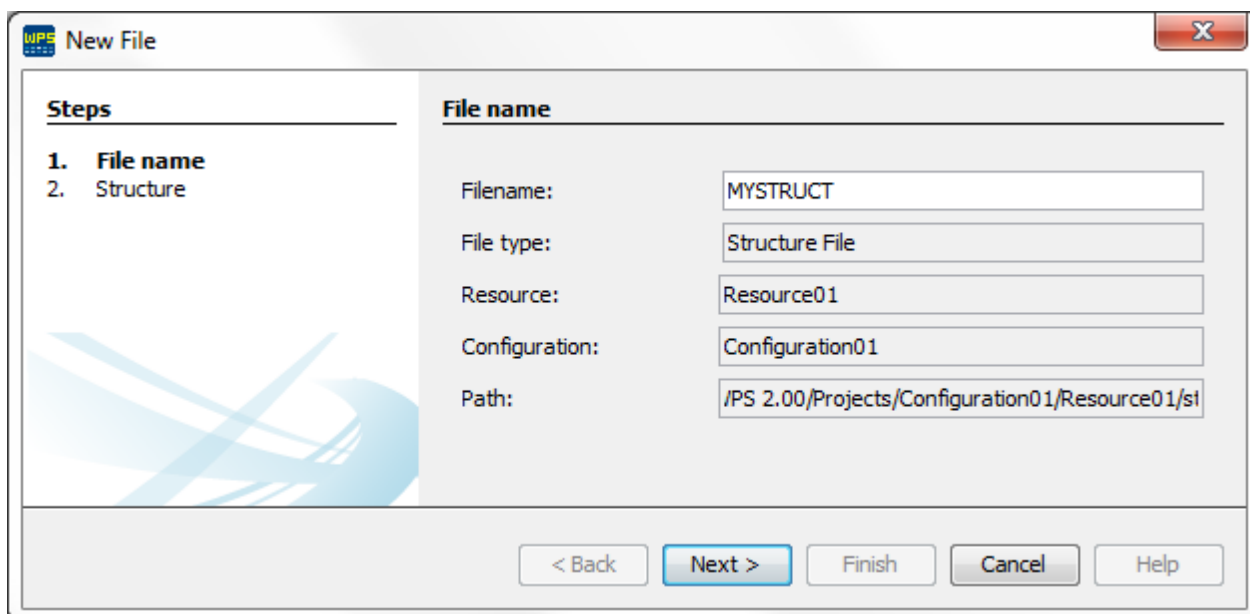


Figura 2 - Definindo o nome da Estrutura

3. Configure a estrutura utilizando os botões apresentados na figura abaixo.

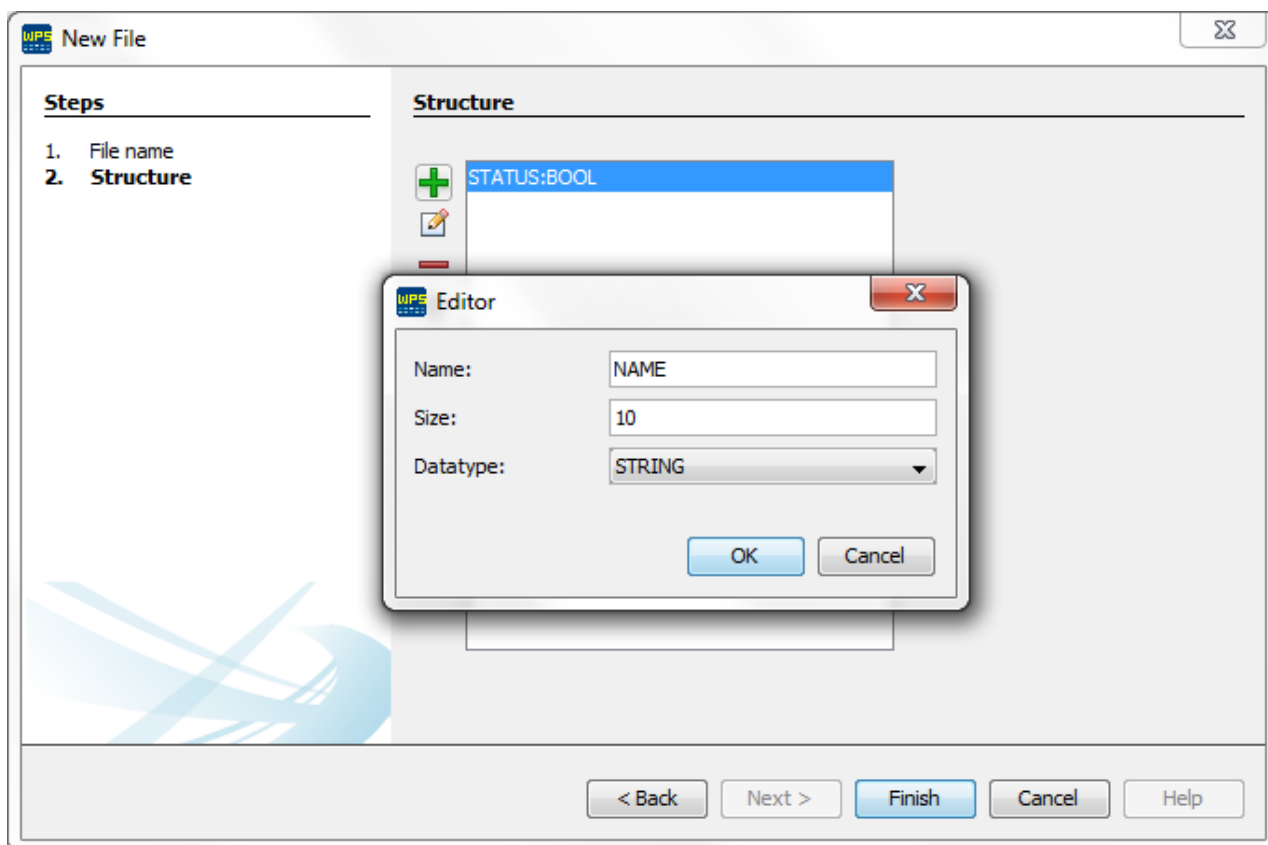


Figura 3 - Editando a Estrutura

4. Após concluir a edição da estrutura, clique no botão **Finalizar**.

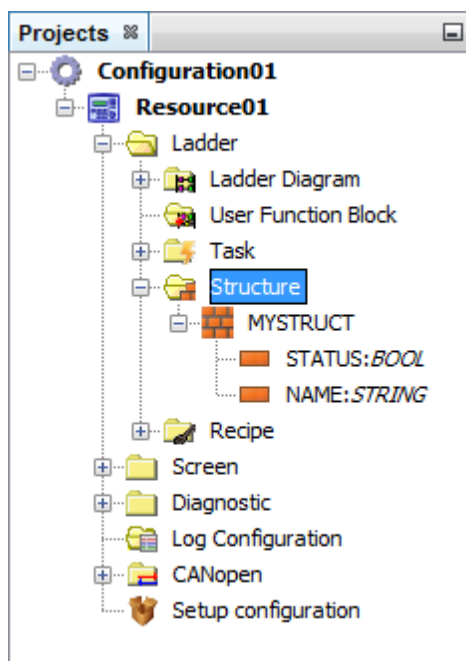


Figura 4 - Estrutura criada no projeto

Editando uma Estrutura

Para editar uma estrutura é necessário dar um duplo-clique na estrutura desejada. Uma janela abrirá, similar à da figura 3, permitindo inserir, apagar ou mover dados.

10.1.5.4 Receitas

Uma receita constitui um conjunto de dados organizados na memória, que definem determinadas configurações para um processo, como por exemplo: tempo de cada etapa, valores mínimos e máximos, setpoint, número de repetições, etc.

Para se criar uma tabela de receitas, primeiramente é necessário definir os dados que a compõem, através de uma estrutura de dados. Para criar uma estrutura de dados, consulte a seção [Estruturas](#).

Criando uma Receita

1. Com o botão direito do mouse sobre a pasta **Receita**, clique em **Novo Arquivo**.

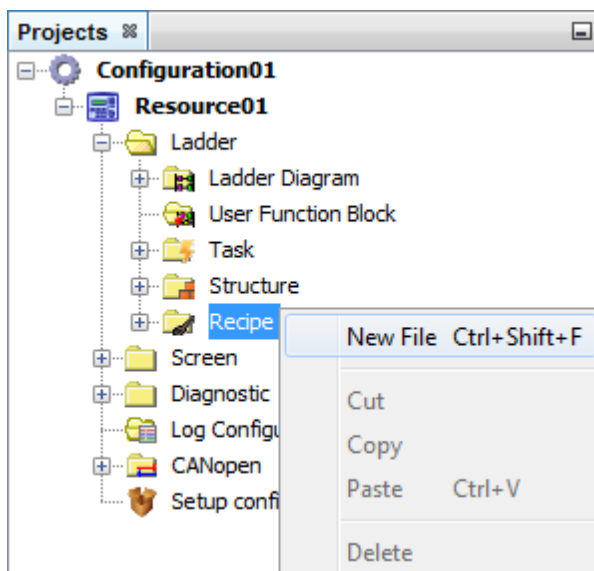


Figura 1 - Criando uma Receita

2. Defina o nome do arquivo e aperte o botão **Próximo**.

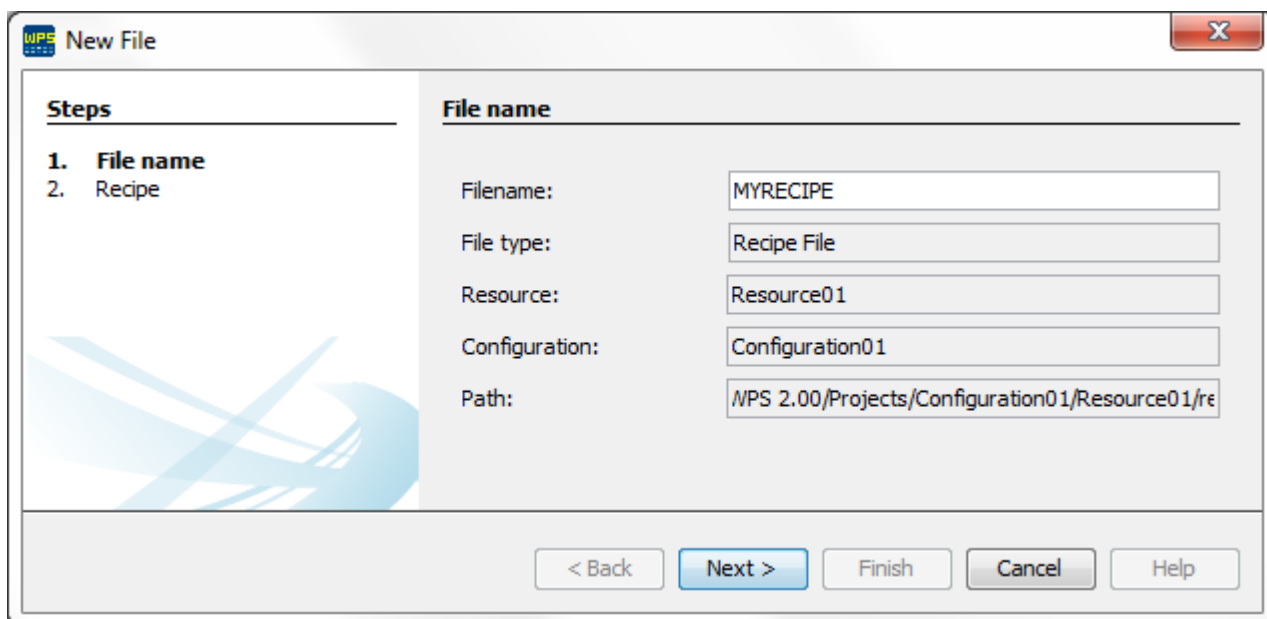


Figura 2 - Definindo o nome da Receita

3. Configure a receita, editando os campos exibidos na figura abaixo.

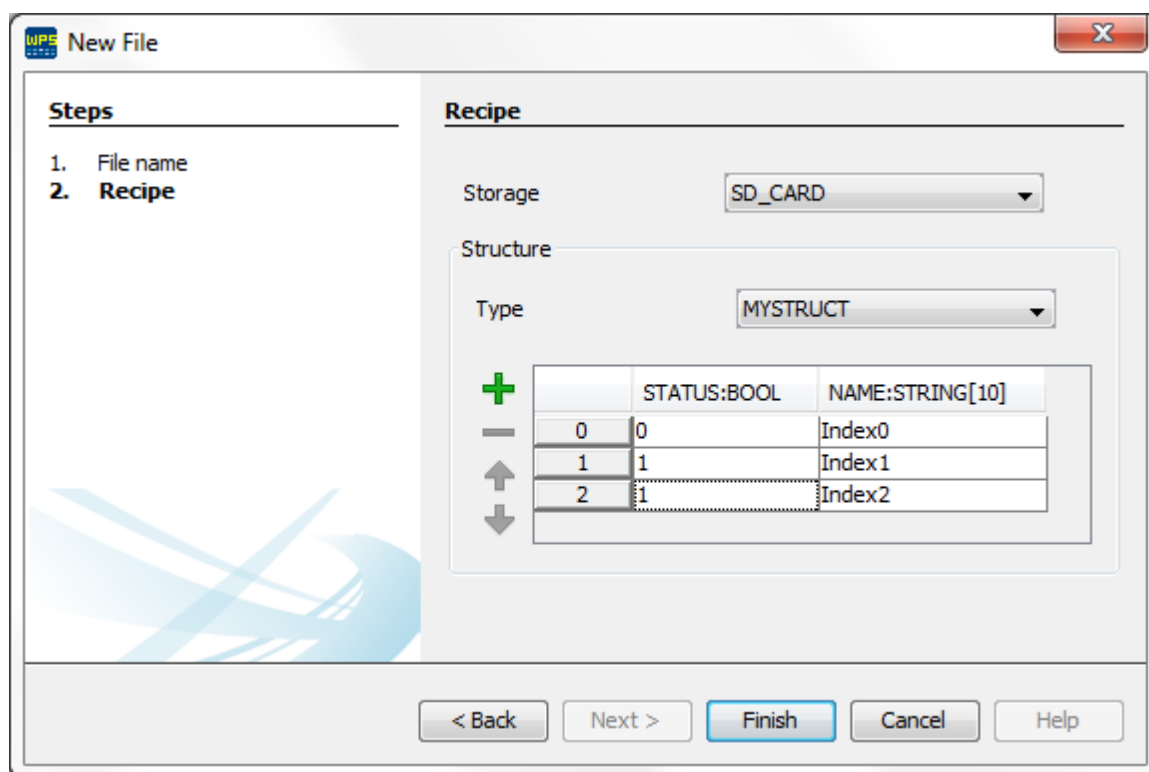


Figura 3 - Editando a receita

- Storage
 - o **RAM Memory:** os dados da receita são armazenados na memória RAM, tornando a sua manipulação mais rápida, porém com capacidade de armazenamento limitada a memória do PLC300.

- o **SD Card**: os dados da receita são armazenados em arquivos no cartão de memória, tornando a sua manipulação bem mais lenta, mas com capacidade de armazenamento de acordo com o cartão SD.
 - **Type**
 - o Permite selecionar todas as estruturas que foram definidas pelo usuário no projeto.
 - **Tabela de Dados**
 - o As colunas representam os elementos da estrutura. Observe que além do nome membro, também é apresentado após os ':' o seu respectivo tipo de dado;
 - o As linhas representam cada receita.
4. Após concluir a edição da estrutura, clique no botão **Finalizar**.

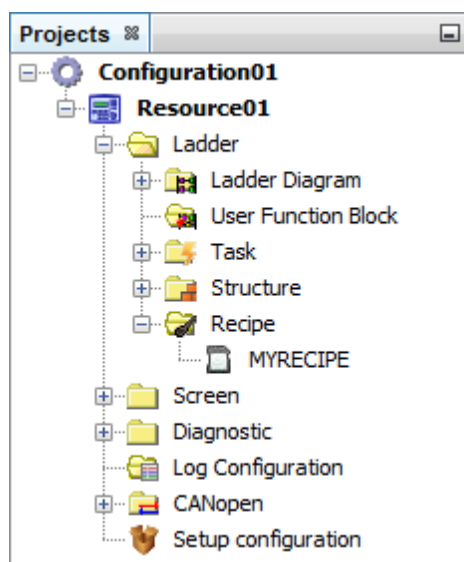


Figura 4 - Receita criada no Projeto

Editando uma Receita

Para editar uma receita é necessário dar um duplo-clique na receita desejada. Uma janela abrirá, similar à da figura 3, permitindo inserir, apagar ou mover dados.

Utilizando as Receitas

Para utilizar os dados da receita, deve-se criar uma variável do tipo de estrutura desejado:

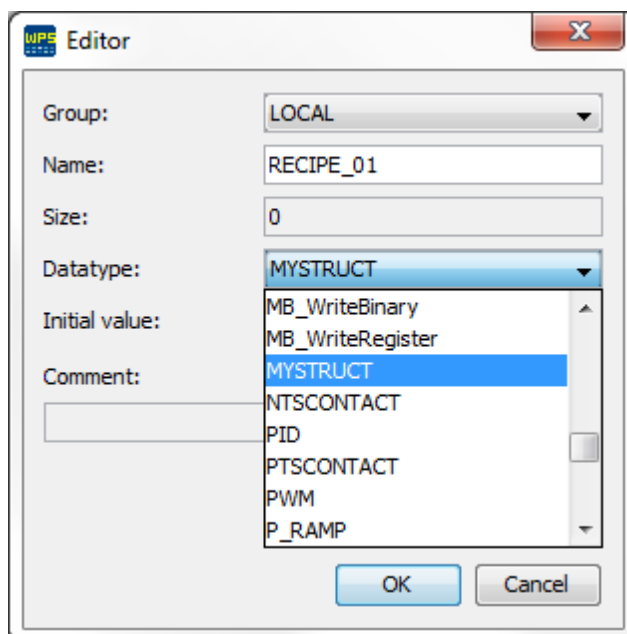


Figura 5 - Criação da variável da estrutura

Variables %				
LOCAL LOCAL_RETAIN				
Tag	Size	Datatype	Initial Value	Comment
RECIPE_01	0	MYSTRUCT		
RECIPE_01.STATUS	0	BOOL		
RECIPE_01.NAME	10	STRING	0	

Figura 6 - Tabela contendo a variável criada

Após a criação da variável deve-se utilizar o bloco ReadRecipe para carregar os dados da receita para a variável ou WriteRecipe para gravar os dados contidos na variável.

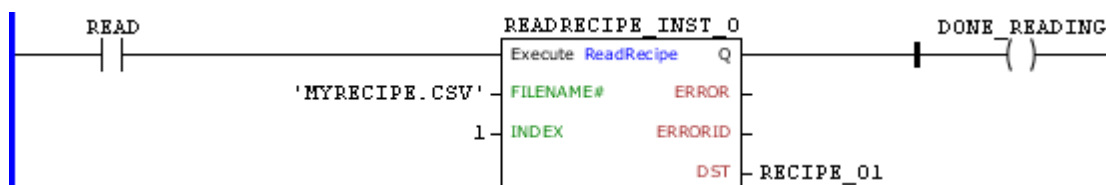


Figura 7 - Bloco ReadRecipe configurado

10.1.6 Tela

[Alarmes](#)

[Editor de Telas](#)

10.1.6.1 Alarmes

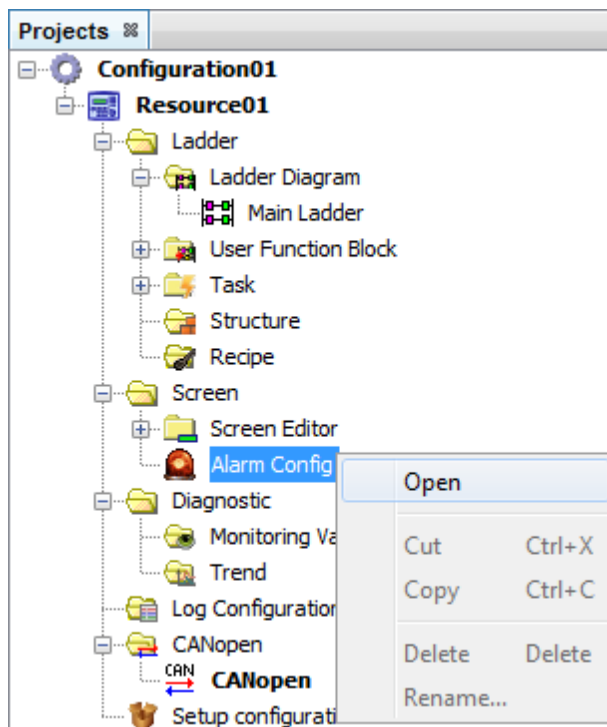
Visão Geral

Os alarmes são uma ferramenta importante na automação de processos, permitindo ao usuário monitorar sua planta, checando pontos críticos e sinalizando ao operador.

No PLC300 os alarmes configuráveis são programados pelo usuário, sendo acionados por um marcador de bit que pode ser ativado pelo programa.

Os alarmes internos são alarmes relativos a alguns componentes de *hardware*.

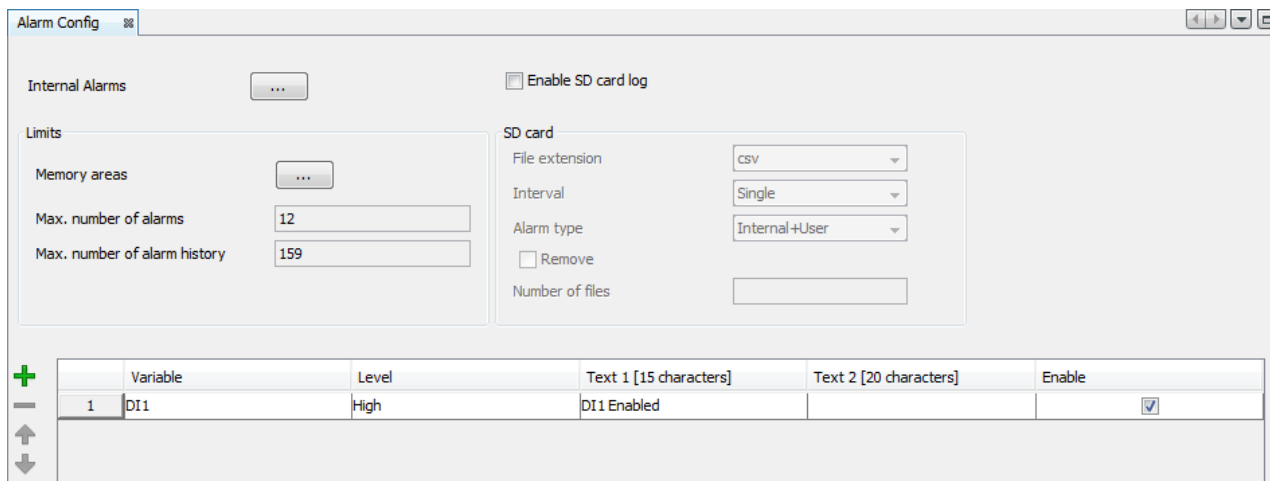
Os alarmes são configurados através da “Configuração de Alarme” acessada através do recurso:



Configuração dos Alarmes

Na janela de configuração dos alarmes são configurados:

- **Alarmes internos:** Alarmes gerados pelo equipamento que podem ser habilitados ou desabilitados pelo usuário;
- **Limites:** Área máxima ocupada pelos alarmes do usuário e pelo histórico de alarmes;
- **SD card:** Configurações de armazenamento dos alarmes no SD card;
- **Tabela de alarmes de usuário:** Tabela para configuração dos alarmes acionados através de variáveis do equipamento.



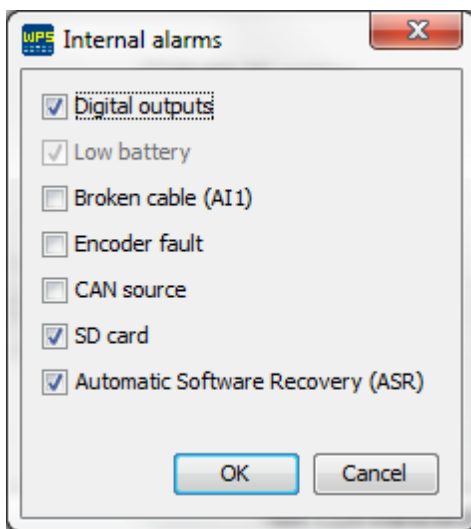
The 'Alarm Config' window is divided into several sections. At the top, there's a tab labeled 'Alarm Config'. Below it, on the left, is the 'Internal Alarms' section with a button to expand it. To the right of this is a checkbox for 'Enable SD card log'. Below 'Internal Alarms' is the 'Limits' section, which includes 'Memory areas' (with an expand button), 'Max. number of alarms' (set to 12), and 'Max. number of alarm history' (set to 159). To the right of 'Limits' is the 'SD card' section, which includes 'File extension' (set to CSV), 'Interval' (set to Single), 'Alarm type' (set to Internal+User), a 'Remove' checkbox, and 'Number of files' (empty). At the bottom, there is a table with columns: Variable, Level, Text 1 [15 characters], Text 2 [20 characters], and Enable. The table contains one row with '1' in the first column, 'DI1' in the second, 'High' in the third, 'DI1 Enabled' in the fourth, and a checked checkbox in the fifth.

	Variable	Level	Text 1 [15 characters]	Text 2 [20 characters]	Enable
1	DI1	High	DI1 Enabled		☒

Alarmes Internos

Os alarmes internos são alarmes gerados pelo equipamento que podem ser habilitados ou desabilitados pelo usuário. Para ter acesso à janela de configuração de alarmes internos é necessário pressionar o botão  na janela de alarmes.

O PLC 300 possui sete alarmes internos, seis dos quais podem ser alterados pelo usuário:



The 'Internal alarms' dialog box shows a list of seven internal alarms with checkboxes. The first three are checked: 'Digital outputs', 'Low battery', and 'SD card'. The last four are unchecked: 'Broken cable (AI1)', 'Encoder fault', 'CAN source', and 'Automatic Software Recovery (ASR)'. At the bottom are 'OK' and 'Cancel' buttons.

- **Falha Saídas Digitais:** Indica que algum problema está ocorrendo em alguma das saídas DO1 a DO8. Pode ser sobretensão, causada por um curto-circuito, ou sobrecarga nas saídas; o alarme também é acionado se os 24 V que alimentam as saídas digitais estiver faltando.
- **Bateria Fraca:** Indica que a bateria que mantém os dados na memória deve ser trocada. Este alarme não pode ser desabilitado pelo usuário. O alarme é gerado quando a tensão da bateria está com seu valor menor que 2,0 V e é desativado quando a tensão é maior que 2,5 V.
- **Fio Partido:** Quando a entrada analógica estiver em modo corrente, 4 a 20 mA, e a corrente estiver abaixo de aproximadamente 2 mA, é acionado o alarme. A normalização deste alarme se dá quando a corrente volta a subir acima dos 3 mA.
- **Falha Encoder:** Caso algum dos sinais do encoder estiver faltando este alarme é ativado. Após a

normalização da falha, o equipamento deve ser reinicializado.

- **Alimentação da CAN:** A interface CAN necessita de uma tensão entre 11 e 30 Vcc para seu correto funcionamento. Na ausência desta tensão, este alarme é acionado.
- **SD card:** Erro na tentativa de escrita ou leitura do SD card através do Log de Evento, Log de Alarmes ou Receitas. Os problemas mais comuns são: SD card ausente, cartão protegido contra escrita e formatação do sistema de arquivos diferente do FAT32.
- **Automatic Software Recovery (ASR):** Indica que o recurso do usuário estava inválido e foi recuperado da memória Flash.

Limites



The 'Limits' configuration window contains the following fields:

Field	Value
Memory areas	...
Max. number of alarms	12
Max. number of alarm history	159

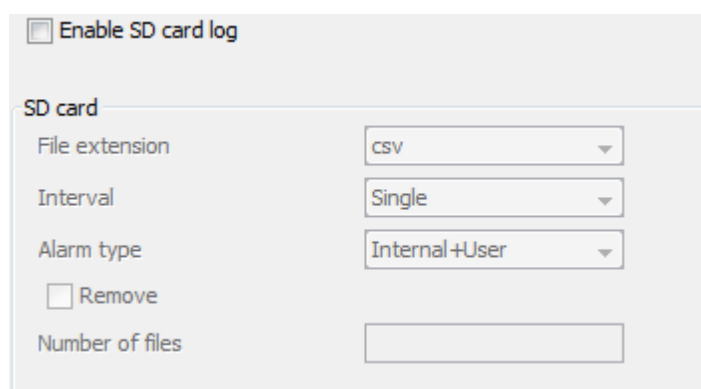
No campo limites é configurada a área máxima ocupada pelos alarmes do usuário e pelo histórico de alarmes. Através do botão  da área de limites dos alarmes, é aberta a janela de configuração das áreas de memória. Através desta janela é possível configurar o tamanho da área de memória que os alarmes de usuário e o histórico de usuário deverão ocupar.

Tamanho Alarme (bytes) = 32 + (80 x numero de alarmes)

O tamanho que o histórico de alarme tem na memória é calculado por meio da formula:

Tamanho do histórico (bytes) = 32 x número de históricos

SD Card



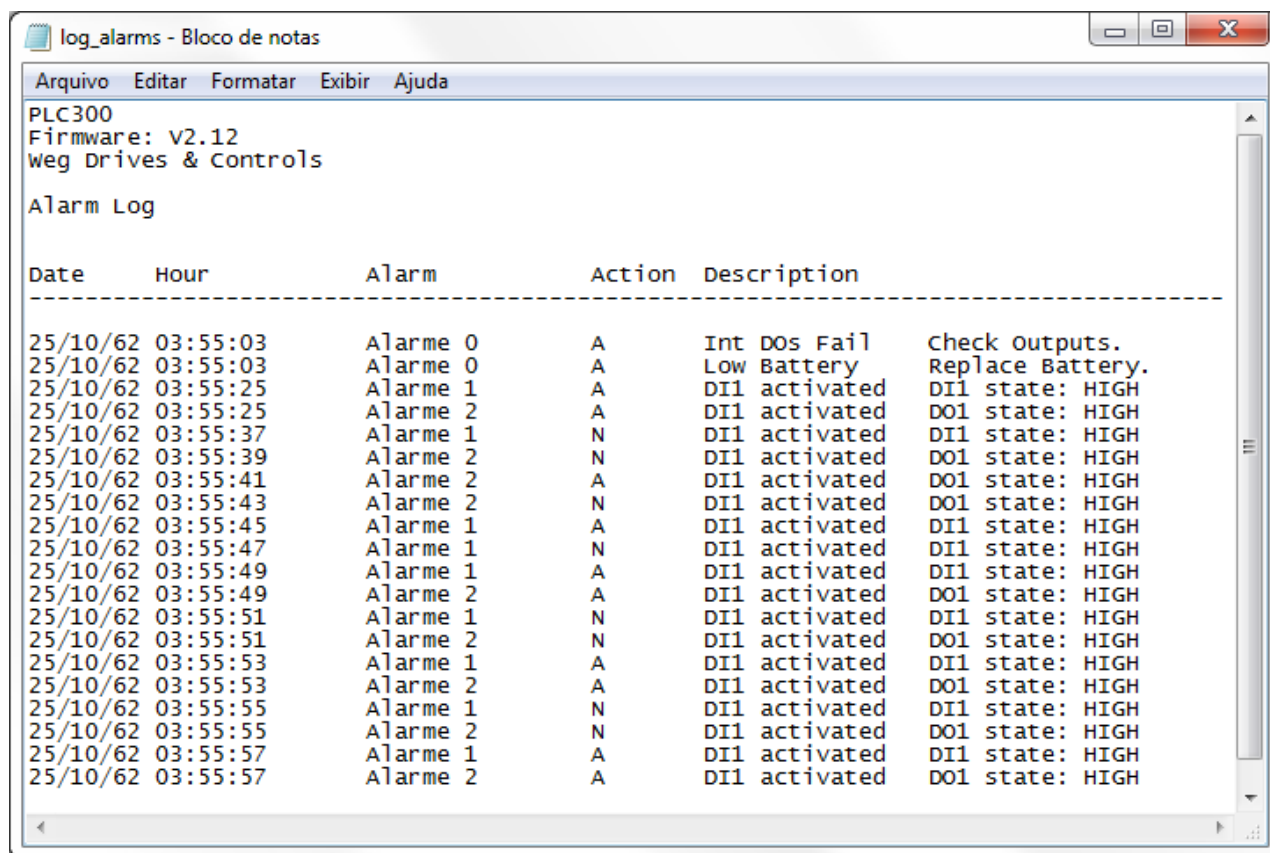
The 'SD card' configuration window includes the following settings:

Field	Value
☐ Enable SD card log	
File extension	CSV
Interval	Single
Alarm type	Internal+User
☐ Remove	
Number of files	

A opção SD card, quando habilitada, configura as características de armazenamento dos alarmes no SD card. O armazenamento dos arquivos de alarmes possui as seguintes opções:

- **Extensão do arquivo:** Formato no qual os alarmes serão armazenados no SD card. As opções são:

o **txt**: texto com formatação de fácil compreensão para os usuários.



log_alarms - Bloco de notas

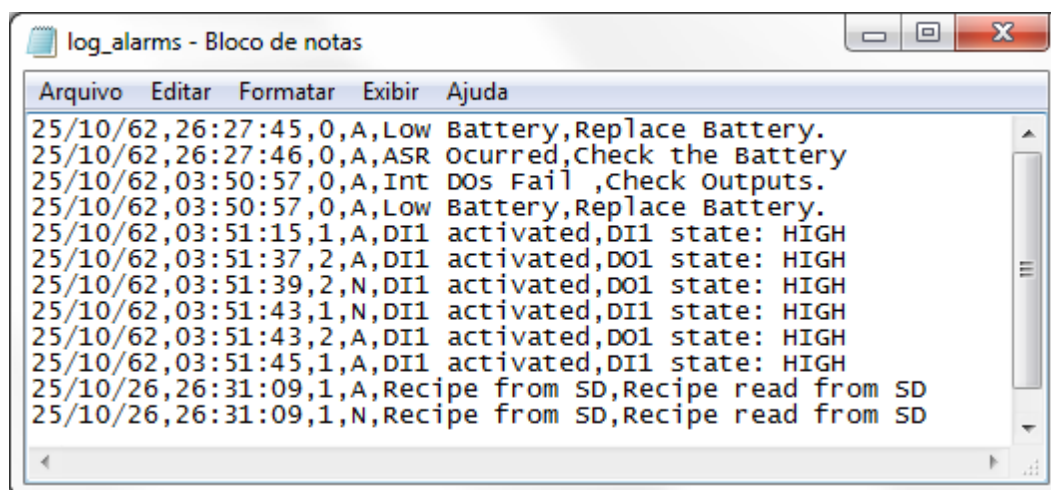
Arquivo Editar Formatar Exibir Ajuda

PLC300
Firmware: V2.12
weg Drives & Controls

Alarm Log

Date	Hour	Alarm	Action	Description
25/10/62	03:55:03	Alarma 0	A	Int D0s Fail
25/10/62	03:55:03	Alarma 0	A	Low Battery
25/10/62	03:55:25	Alarma 1	A	DI1 activated
25/10/62	03:55:25	Alarma 2	A	DI1 activated
25/10/62	03:55:37	Alarma 1	N	DI1 activated
25/10/62	03:55:39	Alarma 2	N	DI1 activated
25/10/62	03:55:41	Alarma 2	A	DI1 activated
25/10/62	03:55:43	Alarma 2	N	DI1 activated
25/10/62	03:55:45	Alarma 1	A	DI1 activated
25/10/62	03:55:47	Alarma 1	N	DI1 activated
25/10/62	03:55:49	Alarma 1	A	DI1 activated
25/10/62	03:55:49	Alarma 2	A	DI1 activated
25/10/62	03:55:51	Alarma 1	N	DI1 activated
25/10/62	03:55:51	Alarma 2	N	DI1 activated
25/10/62	03:55:53	Alarma 1	A	DI1 activated
25/10/62	03:55:53	Alarma 2	A	DI1 activated
25/10/62	03:55:55	Alarma 1	N	DI1 activated
25/10/62	03:55:55	Alarma 2	N	DI1 activated
25/10/62	03:55:57	Alarma 1	A	DI1 activated
25/10/62	03:55:57	Alarma 2	A	DI1 activated

o **csv (comma separated value)**: valores separados por vírgula, geralmente utilizado em planilhas eletrônicas.



log_alarms - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

25/10/62,26:27:45,0,A,Low Battery,Replace Battery.
25/10/62,26:27:46,0,A,ASR Occurred,Check the Battery
25/10/62,03:50:57,0,A,Int D0s Fail ,check outputs.
25/10/62,03:50:57,0,A,Low Battery,Replace Battery.
25/10/62,03:51:15,1,A,DI1 activated,DI1 state: HIGH
25/10/62,03:51:37,2,A,DI1 activated,DO1 state: HIGH
25/10/62,03:51:39,2,N,DI1 activated,DO1 state: HIGH
25/10/62,03:51:43,1,N,DI1 activated,DI1 state: HIGH
25/10/62,03:51:43,2,A,DI1 activated,DO1 state: HIGH
25/10/62,03:51:45,1,A,DI1 activated,DI1 state: HIGH
25/10/26,26:31:09,1,A,Recipe from SD,Recipe read from SD
25/10/26,26:31:09,1,N,Recipe from SD,Recipe read from SD

Os valores armazenados são a data, hora, alarme (0 – interno, 1 – alarme do usuário), ação (A – atuado e N – normalizado) e descrição.

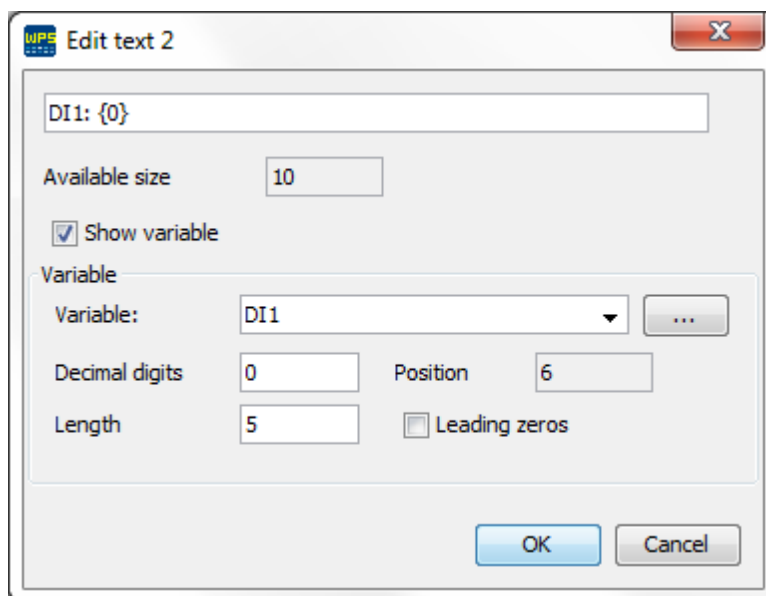
- **Intervalo:** No campo intervalo é configurado o tempo de duração da gravação dos dados em um único arquivo. As opções deste campo são as seguintes:
 - o **Único:** Os dados serão gravados em um único arquivo.
 - o **Diário:** Os dados são gravados em um arquivo por dia. A gravação de um novo arquivo é iniciada sempre que ocorrer a mudança do dia no relógio do equipamento. O arquivo é gravado com um sufixo contendo o dia, mês e ano em que iniciou sua gravação.
 - o **Mensal:** Os dados são gravados em um arquivo por mês. A gravação de um novo arquivo é iniciada sempre que ocorrer a mudança do mês no relógio do equipamento. O arquivo é gravado com um sufixo contendo o mês e o ano em que iniciou sua gravação.
 - o **Anual:** Os dados são gravados em um arquivo por ano. A gravação de um novo arquivo é iniciada sempre que ocorrer a mudança do ano no relógio do equipamento. O arquivo é gravado com um sufixo contendo o ano em que iniciou sua gravação.
- **Tipo do alarme:** Valores de alarme que serão armazenados no SD card. Podem ser dos tipos “Usuário” para armazenar somente os alarmes configurados pelo usuário ou “Usuário+Internos” para armazenar os alarmes configurados pelo usuário e os alarmes internos.
- **Remover:** Quando selecionada esta opção a caixa de diálogo “Número de arquivos” se torna habilitada, permitindo a entrada de um número inteiro. Este número representa o número de arquivos que serão mantidos no SD card. Sempre que um arquivo é criado diariamente, mensalmente ou anualmente é verificado o número de arquivos criados para este alarme e então removidos os arquivos mais antigos.

Tabela de Alarmes de Usuário

	Variable	Level	Text 1 [15 characters]	Text 2 [20 characters]	Enable
1	DI1	High	DI1 Enabled		☒

Na tabela de alarmes de usuário é realizada a configuração dos alarmes acionados através de variáveis do equipamento com textos editados pelo usuário. A tabela dos alarmes possui quatro campos a serem preenchidos:

- **Marcador:** Variável global do tipo booleana que ativa o alarme.
- **Borda:** Borda de transição em que o alarme será ativado. Os valores possíveis são positiva (de 0 para 1) ou negativa (de 1 para 0).
- **Habilita:** Opção que habilita/desabilita o alarme.
- **Texto 1:** Texto contendo a mensagem de alarme. Este campo poderá conter no máximo 15 caracteres e será visualizado na lista de alarmes ativos e histórico de alarmes.
- **Texto 2:** Texto com o descritivo da mensagem de alarme. Este campo poderá conter no máximo 20 caracteres e será visualizado na descrição detalhada dos alarmes ativos e histórico de alarmes. A partir da versão 1.50 do aplicativo é possível configurar uma variável para ser mostrada no texto do alarme. Esta configuração é feita selecionando a célula do texto 2. Ao selecionar a célula, a seguinte janela irá aparecer:



Os campos de visualização e configuração do texto 2 são os seguintes:

- **Espaço disponível:** Número de caracteres disponíveis para a edição do texto. O número máximo de caracteres é 20 (caso a opção mostrar variável seja selecionada, o espaço ocupado pela variável é adicionado aos caracteres ocupados pelo texto).
- **Mostrar variável:** Habilita a opção de mostrar variável no texto 2 do alarme. Caso o texto não possua a marcação da localização da variável "{0}", ela é automaticamente adicionada.
- **Variável:** Variável que será apresentada no texto, na posição e formatação definidos nos campos: posição, dígitos decimais, comprimento e preenchimento com zeros.
- **Dígitos decimais:** Número de dígitos decimais para a apresentação da variável.
- **Posição:** Posição em que a variável será inserida. Este campo é somente leitura e é atualizado a cada mudança de posição.
- **Comprimento:** Espaço que será reservado para a apresentação da variável. O usuário deverá ter o cuidado de reservar o espaço suficiente para apresentação do sinal e ponto decimal, caso necessário.
- **Zeros à esquerda:** preenche os espaços que estiverem vazios, entre o comprimento configurado e o tamanho da variável, com zeros.

10.1.6.2 Editor de Telas

Visão geral

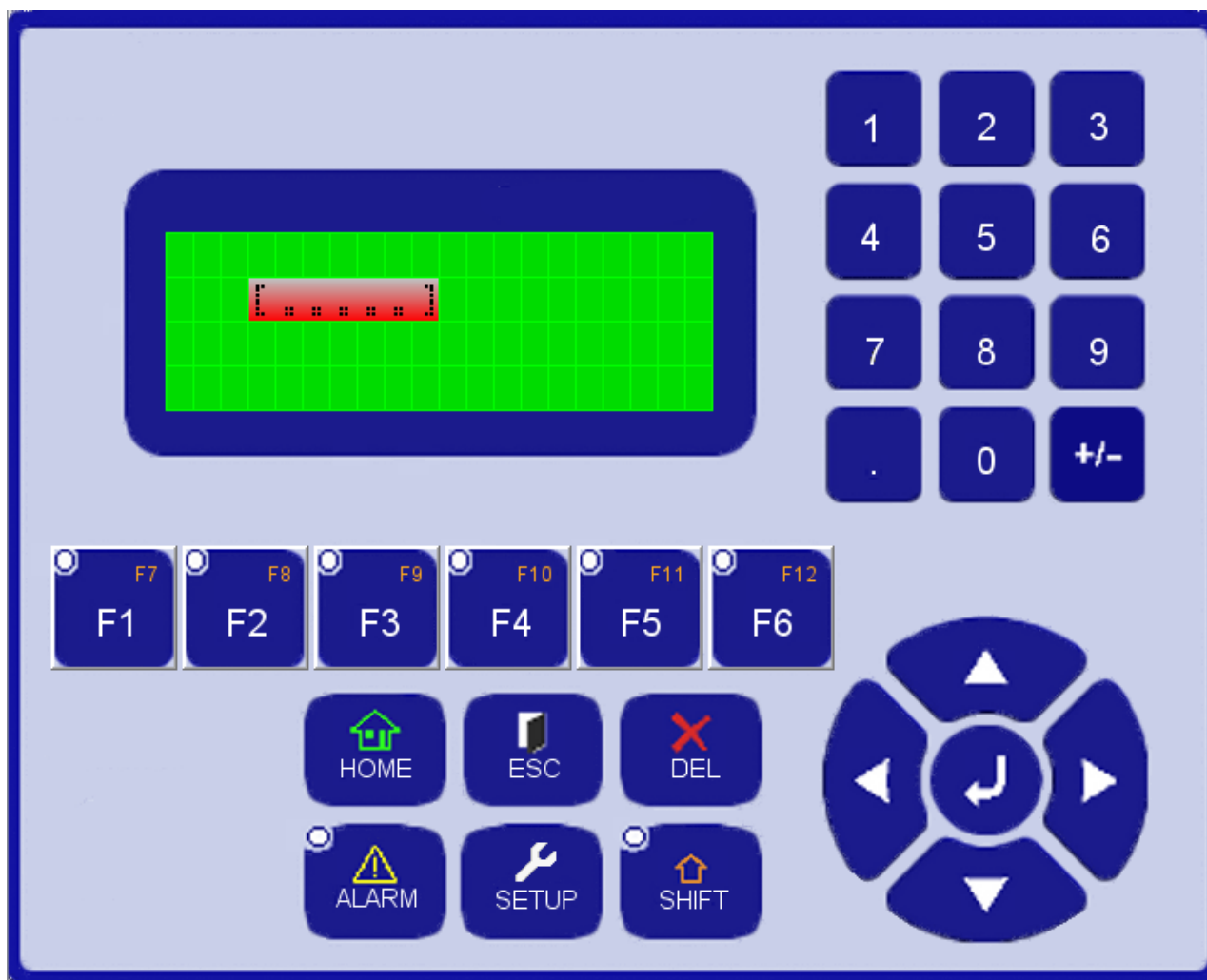
O editor de telas permite que telas sejam configuradas pelo usuário, adicionando e removendo componentes, para que valores sejam lidos e escritos no programa ou apresentados no display do equipamento.

O editor possui teclas de função que permitem a navegação entre as telas ou a manipulação de bits. Para cada uma das telas é permitida configuração de um nível de acesso para permitir que somente usuários com privilégio acessem o conteúdo.



NOTA!

O PLC300 permite que sejam editadas no máximo 512 telas.



Os componentes utilizados para a edição das telas são os seguintes:



Bargraph: Apresenta o valor de uma variável de forma gráfica;



Text: Apresenta o texto na tela;



Numeric Input: Realiza a entrada de valores numéricos para uma variável;



Numeric Output: Apresenta o valor numérico de uma variável;



Message: Apresenta textos configurados em uma tabela, selecionados através de um índice;



Text Input: Realiza a entrada de variáveis do tipo STRING;



Text Output: Apresenta o valor de variáveis do tipo STRING;

Teclas de Função

O navegação de telas e a manipulação de bits são feitos através da configuração das teclas de função.



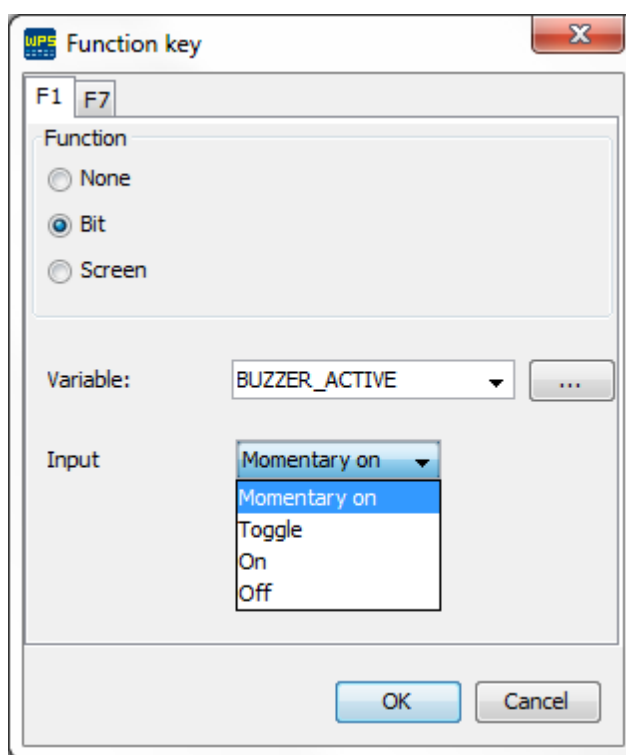
As opções disponíveis para a configuração das teclas são as seguintes:

- **Nenhuma:** Esta tecla de função não irá realizar nenhuma ação;
- **Bit:** Seleciona uma das funções de manipulação de bits;
- **Screen:** Seleciona uma tela para navegação.

Manipulação de Bits

A função de manipulação de bits permite que uma variável booleana seja ligada, desligada, tenha seu estado invertido ou seja ligada temporariamente.

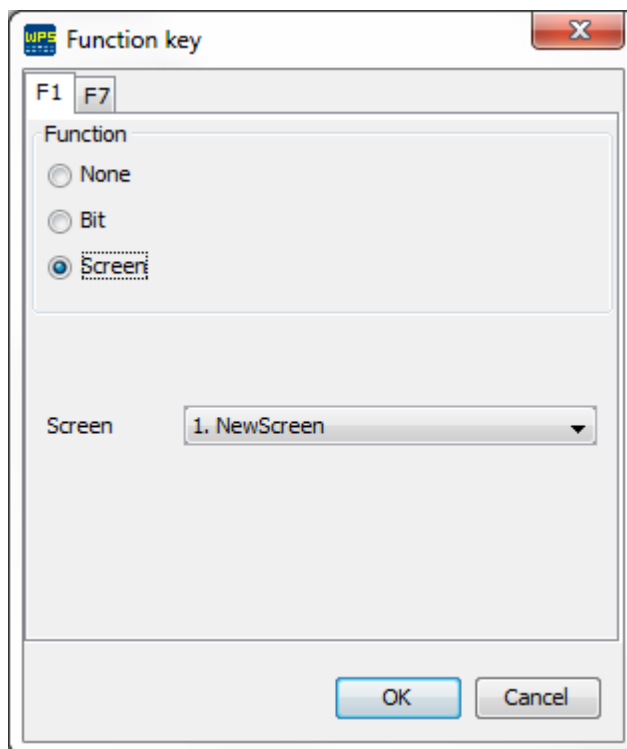
As opções disponíveis para seleção são as seguintes:



- **Momentary On:** Liga uma variável booleana selecionada (escreve valor 1) enquanto a tecla estiver pressionada. Ao soltar a tecla, o marcador volta a zero.
- **Toggle:** Inverte o estado da variável booleana selecionada;
- **On:** Liga uma variável booleana selecionada (escreve valor 1);
- **Off:** Desliga uma variável booleana selecionada (escreve valor 0).

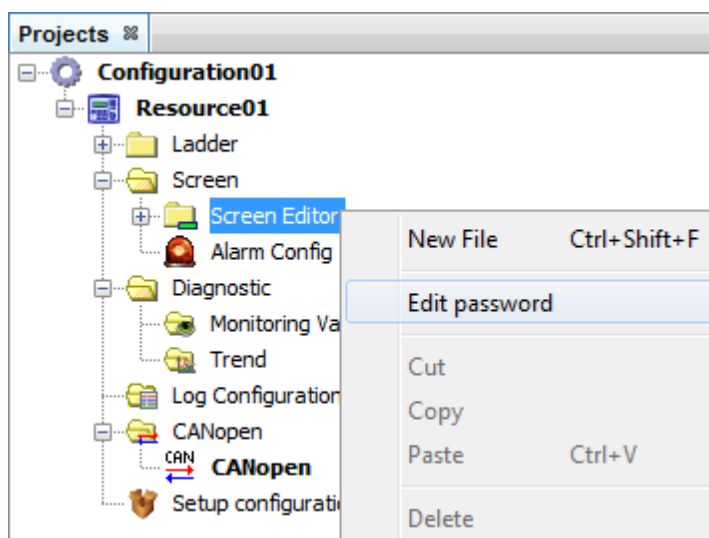
Navegação das Telas

A navegação entre as telas é feita através da edição da opção **Screen**. No exemplo abaixo, a tela 2 será acessada ao pressionar a tecla **F1**.

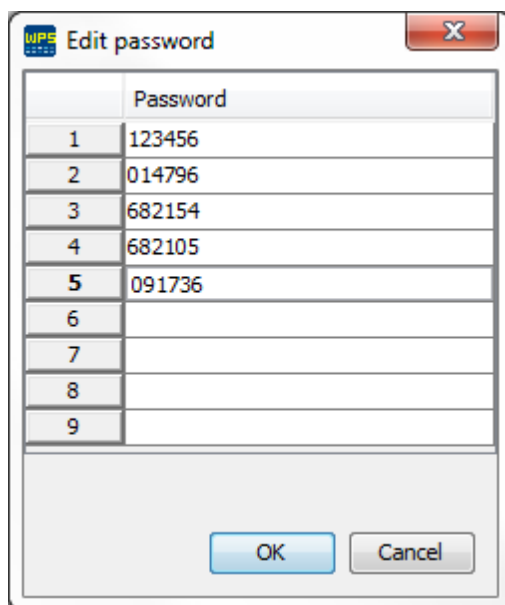


Senhas e níveis de acesso

O PLC300 possui dez níveis de senha (0 - 9). O nível zero (valor padrão) é escolhido quando é desejado o acesso livre à tela. Com exceção da tela número zero (Home), que sempre possui nível de acesso zero, as demais telas podem ser programadas com um nível de acesso. A janela de configuração das senhas é aberta ao selecionar as opções da pasta **Editor de telas**.



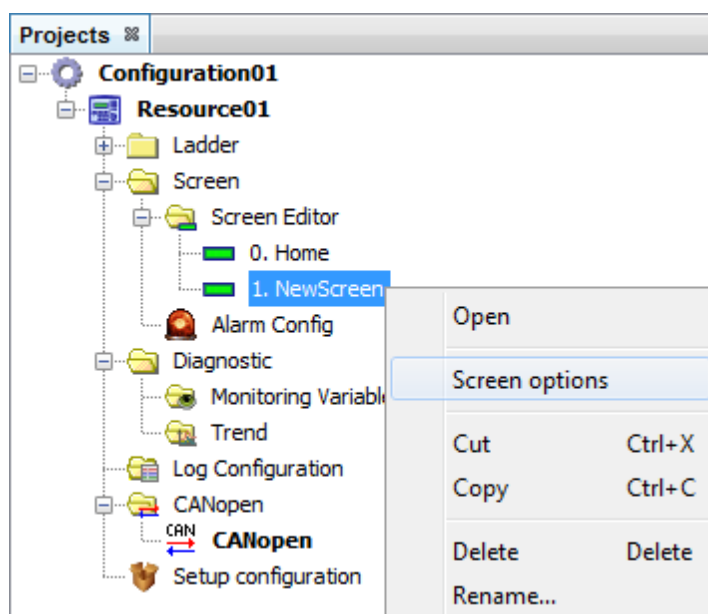
As senhas configuradas devem possuir seis caracteres decimais.



Para configurar o nível de acesso às telas é necessário configurar esta opção nas opções de tela.

Opções da tela

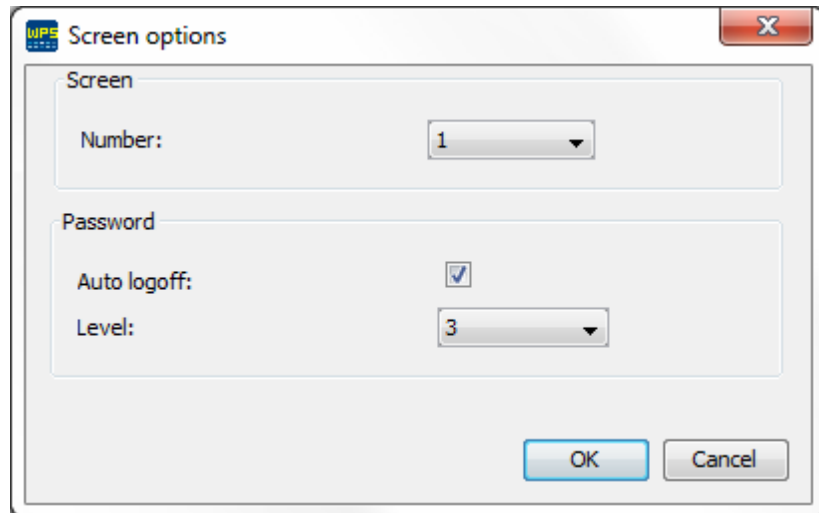
Nas opções de tela é possível configurar o número atual da tela e as propriedades de senha da tela. Para abrir estas opções é necessário selecionar as opções do arquivo de telas no recurso.



As opções de senha são as seguintes:

- **Auto Logoff:** Quando uma tela com nível de acesso é visualizada, caso o nível do acesso da tela seja superior o atual, o nível de acesso do equipamento é tornado igual ao nível atual da tela. Caso a opção de auto logoff esteja habilitada, após o acesso à tela, o nível de acesso do equipamento não será alterado.
- **Nível:** Seleciona o nível de acesso da tela. Quanto maior o nível de acesso, maior o privilégio. Até nove

níveis de acesso podem ser programados, de modo a permitir diferentes tipos de usuários com acesso a determinadas telas. O nível de acesso zero significa que a tela não precisará de senha para ser acessada.

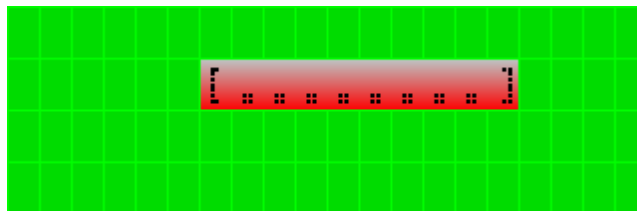


Edição das telas

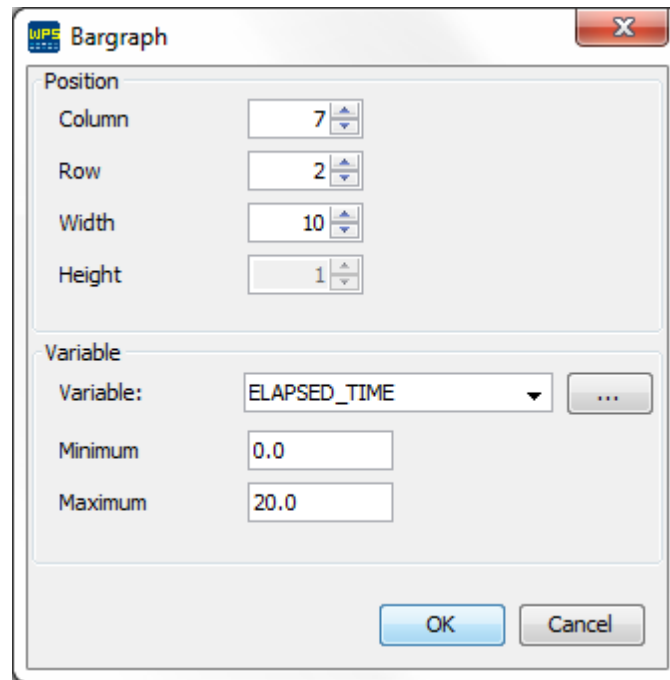
As telas são compostas por componentes do tipo Bargraph, Numeric Input, Numeric Output, Text, Message e Text Output. Para a programação das telas os componentes são inseridos no display arrastando os componentes da paleta para o display ou clicando com o botão direito do mouse sobre o display. Os componentes utilizados para a edição das telas são os seguintes:

Bargraph

O componente Bargraph mostra um gráfico de barras no display, com caracteres tipo bloco, proporcional ao valor da variável selecionada.



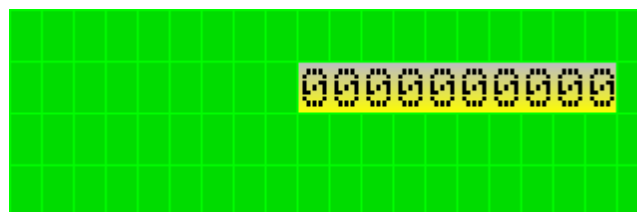
As propriedades deste componente são:



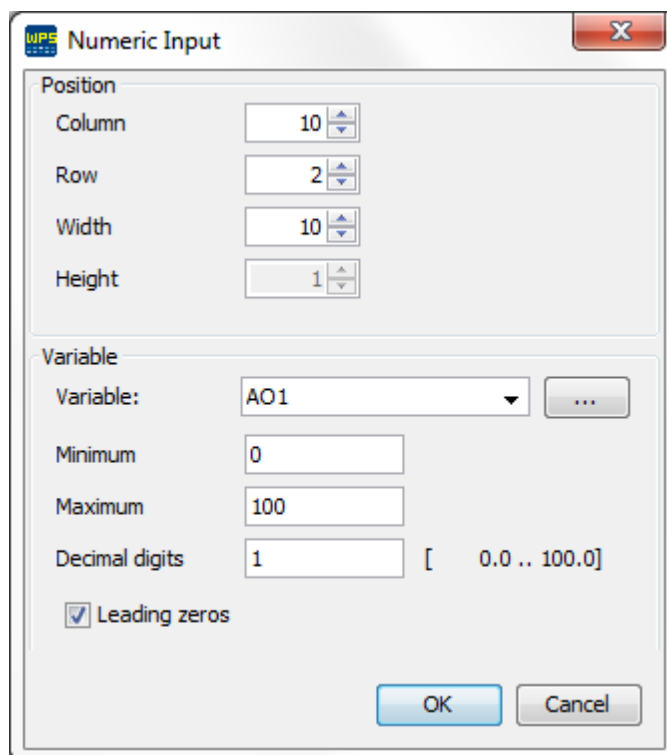
- **Variável:** Variável utilizada para a apresentação do gráfico de barras.
- **Máximo:** Valor máximo apresentado pelo gráfico. Caso o valor da variável seja igual ou superior ao valor máximo, o gráfico é apresentado com todas as barras preenchidas.
- **Mínimo:** Valor mínimo apresentado pelo gráfico. Caso o valor da variável seja igual ou inferior ao valor mínimo, o gráfico não apresenta nenhuma barra preenchida.

Numeric Input

O componente Numeric Input possibilita a entrada de um valor numérico pelo usuário dentro de uma faixa. O conteúdo digitado é armazenado em uma variável.



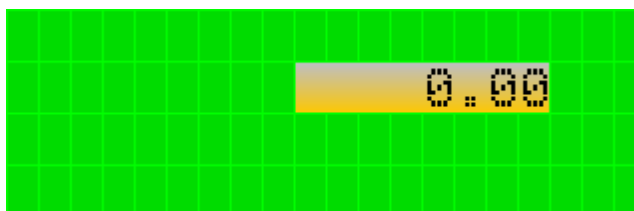
As propriedades deste componente são:



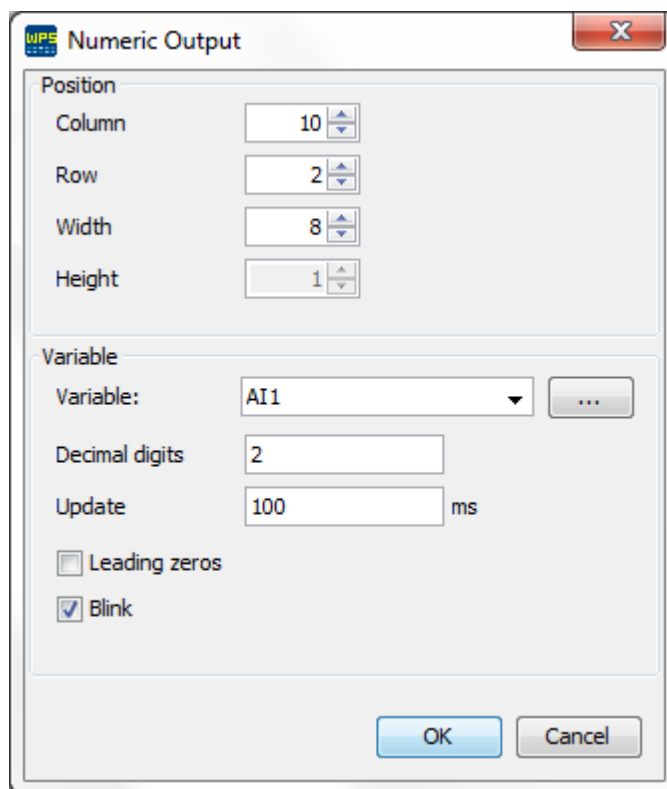
- **Variável:** Variável utilizada para entrada de dados no display;
- **Máximo:** Valor máximo permitido como entrada para este campo. Caso o valor digitado seja superior a este valor, será atribuído o valor máximo;
- **Mínimo:** Valor mínimo permitido como entrada para este campo. Caso o valor digitado seja superior a este valor, será atribuído o valor mínimo;
- **Dígitos decimais:** Número de casas decimais na apresentação da variável no display;
- **Zeros à esquerda:** Preenche os espaços que estiverem vazios, entre o tamanho do componente configurado e o tamanho da variável, com zeros.

Numeric Output

O componente Numeric Output apresenta o valor formatado (dígitos decimais, zeros à esquerda ou piscante) da variável selecionada no display.



As propriedades deste componente são:



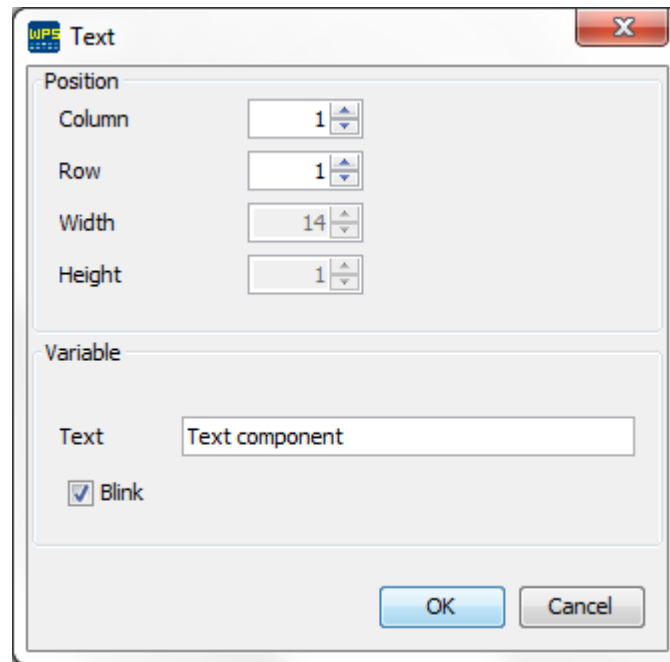
- **Variável:** Variável utilizada para apresentação dos dados no display;
- **Dígitos decimais:** Número de casas decimais na apresentação da variável no display;
- **Atualização:** Tempo (em milissegundos) utilizado para a atualização do campo no display. Este valor é apenas uma aproximação pois depende do varredura do equipamento;
- **Zeros à esquerda:** Preenche os espaços que estiverem vazios, entre o tamanho do componente configurado e o tamanho da variável, com zeros;
- **Piscante:** Habilita a opção de texto piscante.

Text

O componente Text é utilizado para inserir textos fixos na tela.



As propriedades deste componente são:



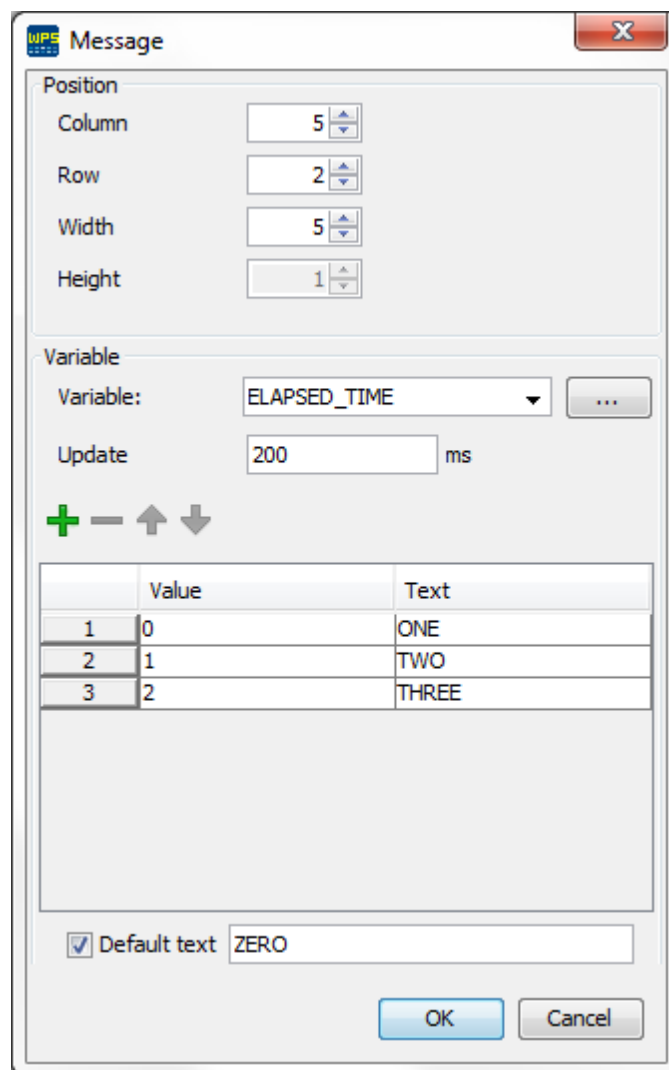
- **Texto:** Texto apresentado na tela;
- **Pisca:** Habilita a opção de texto piscante.

Message

O componente Message apresenta textos configurados em uma tabela, selecionados através de um índice.



As propriedades deste componente são:



The image shows a 'WPS Message' dialog box with the following fields and controls:

- Position:**
 - Column: 5
 - Row: 2
 - Width: 5
 - Height: 1
- Variable:**
 - Variable: ELAPSED_TIME (dropdown menu)
 - Update: 200 ms
- Table:**

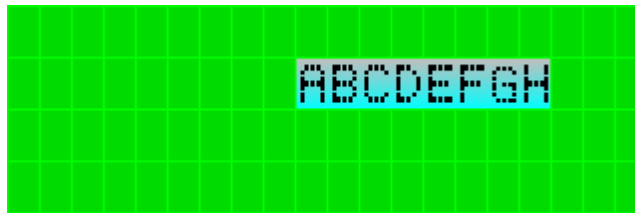
Controls: +, -, ↑, ↓

	Value	Text
1	0	ONE
2	1	TWO
3	2	THREE
- Default text:** ☒ Default text ZERO
- Buttons:** OK, Cancel

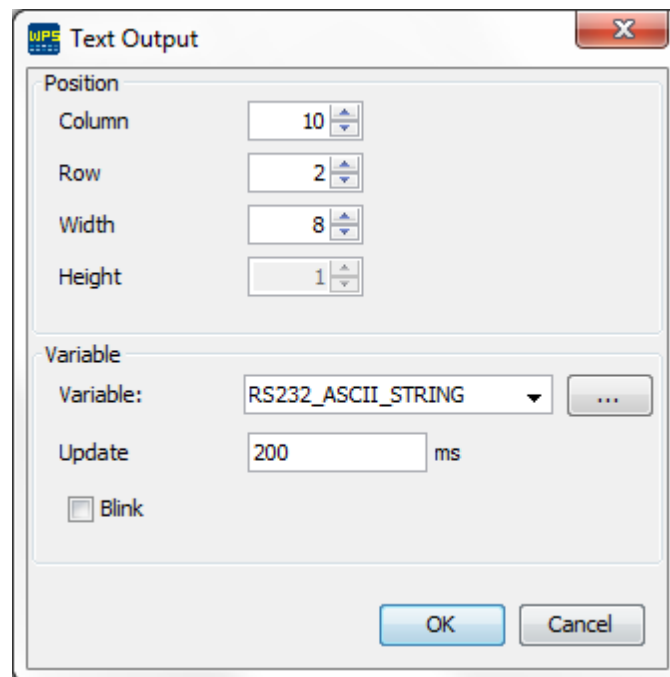
- **Variável:** Variável utilizada como índice para apresentação do texto na tela;
- **Atualização:** Tempo (em milissegundos) utilizado para a atualização do campo no display. Este valor é apenas uma aproximação pois depende do varredura do equipamento;
- **Tabela:**
 - o **Valor:** Índice da tabela;
 - o **Texto:** Texto apresentado quando o valor da variável equivale ao índice da tabela;
- **Texto padrão:** Texto apresentado quando a variável assume um valor não configurado na coluna valor. Quando esta opção está desabilitada, o texto apresentado é o valor da variável.

Text Output

O componente Text Output apresenta o valor armazenado em uma variável do tipo STRING.



As propriedades deste componente são:



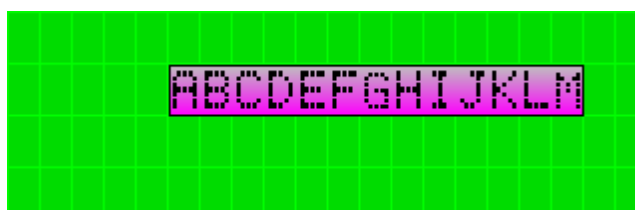
- **Variável:** Variável do tipo STRING;
- **Atualização:** Tempo (em milissegundos) utilizado para a atualização do campo no display. Este valor é apenas uma aproximação pois depende do varredura do equipamento;
- **Pisca:** Habilita a opção de texto piscante.

**NOTA!**

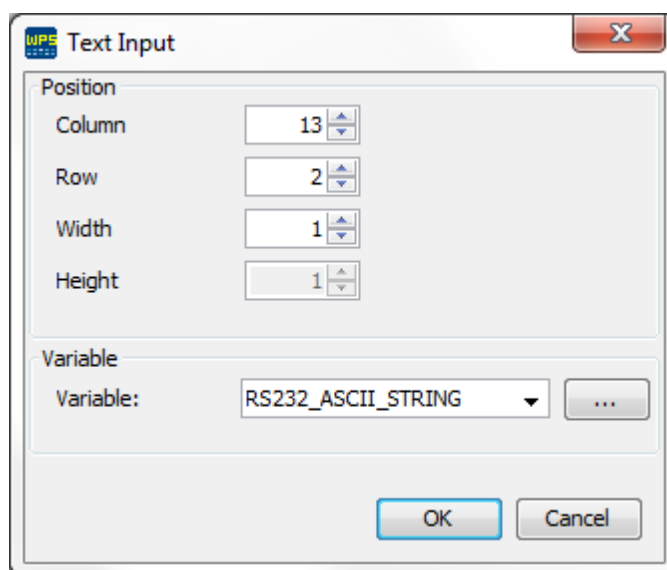
Uma variável do tipo BYTE era utilizada em versões de *firmware* inferiores à 2.10 em vez da variável do tipo STRING. O tamanho do vetor era definido pelo tamanho do componente.

Text Input

O componente Text Input realiza a entrada de valores em uma variável do tipo STRING.



As propriedades deste componente são:



- **Variável:** Variável do tipo STRING.

**NOTA!**

Uma variável do tipo BYTE era utilizada em versões de *firmware* inferiores à 2.10 em vez da variável do tipo STRING. O tamanho do vetor era definido pelo tamanho do componente.

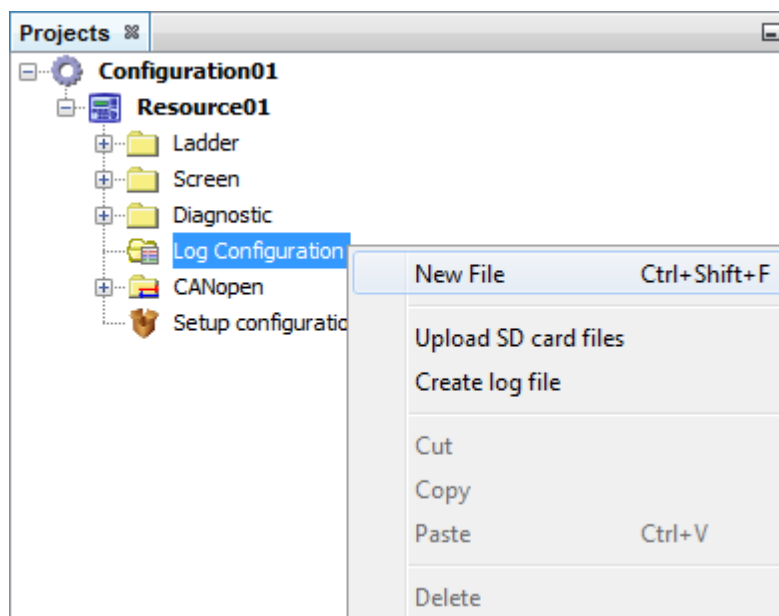
10.1.7 Log de Eventos

Visão Geral

O *log* de eventos é um conjunto valores de variáveis que são armazenados, com data e hora, no SD card do equipamento, no formato csv (comma separated value). Estes valores são gravados após a ocorrência de eventos que podem ser dos tipos: time, change of state ou trigger. Para cada arquivo de *log* de eventos é configurado o intervalo de gravação do arquivo e é associada uma variável booleana responsável por habilitar ou desabilitar o *log* através do programa.

Para cada arquivo de configuração de *log* pode ser associado um texto que será apresentado juntamente com as variáveis assim que o *upload* do *log* de eventos for realizado (Ver seção configuração do campo texto).

No menu de opções do *log* de eventos é possível realizar as seguintes ações:

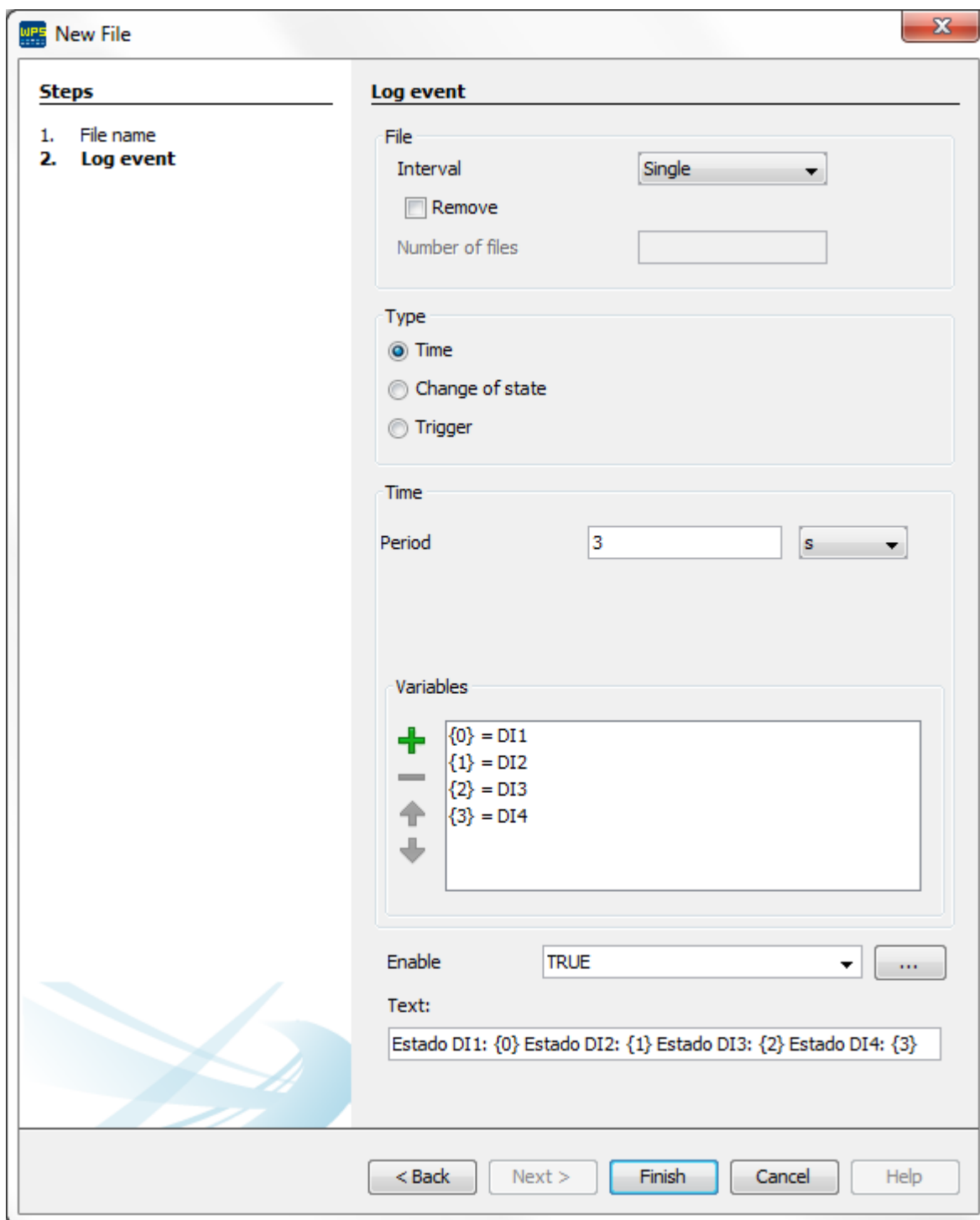


- **Adicionar um novo log de eventos:** para adicionar um novo *log* de eventos é preciso selecionar a opção **Novo arquivo** da pasta **Configuração de Log**. Para mais informações de como configurar o *log* de eventos ver seção *Configuração do log de eventos*.
- **Upload arquivos SD card:** após o *download* dos arquivos de *log* de eventos configurados para o equipamento, é possível fazer o *upload* destes arquivos para visualizar os eventos ocorridos. Mais detalhes ver seção *Upload arquivos SD card*.
- **Criar arquivo de log:** esta funcionalidade permite que o usuário crie um arquivo de *log* dentro de um período definido com os arquivos de *log* carregados no projeto. Mais detalhes ver seção *Criação arquivos de log*.

Configuração de Log de Eventos

Na janela de configuração de *log* de eventos são configurados: o intervalo de gravação do arquivo de *log* de eventos, o tipo do *log* de eventos, a variável booleana que habilita o *log* eventos e o texto a ser exibido no *log* de eventos.

- **Intervalo do arquivo:** No campo intervalo do arquivo é configurado o tempo de duração da gravação dos dados em um único arquivo. As opções deste campo são as seguintes:
 - o **Único:** Os dados serão gravados em um único arquivo.
 - o **Diário:** Os dados são gravados em um arquivo por dia. A gravação de um novo arquivo é iniciada sempre que ocorrer a mudança do dia no relógio do equipamento. O arquivo é gravado com um sufixo contendo o dia, mês e ano em que iniciou sua gravação.
 - o **Mensal:** Os dados são gravados em um arquivo por mês. A gravação de um novo arquivo é iniciada sempre que ocorrer a mudança do mês no relógio do equipamento. O arquivo é gravado com um sufixo contendo o mês e o ano em que iniciou sua gravação.
 - o **Anual:** Os dados são gravados em um arquivo por ano. A gravação de um novo arquivo é iniciada sempre que ocorrer a mudança do ano no relógio do equipamento. O arquivo é gravado com um sufixo contendo o ano em que iniciou sua gravação.



The image shows a 'New File' dialog box from the WPS software. It has a 'Steps' panel on the left and a 'Log event' panel on the right. The 'Steps' panel lists two steps: '1. File name' and '2. Log event', with the second step being the active one. The 'Log event' panel contains several sections: 'File' with an 'Interval' dropdown set to 'Single', a 'Remove' checkbox, and a 'Number of files' input field; 'Type' with three radio buttons: 'Time' (selected), 'Change of state', and 'Trigger'; 'Time' with a 'Period' input field set to '3' and a unit dropdown set to 's'; 'Variables' with a list of four variables: '{0} = DI1', '{1} = DI2', '{2} = DI3', and '{3} = DI4'; 'Enable' with a dropdown set to 'TRUE' and a button with three dots; and 'Text:' with a text field containing 'Estado DI1: {0} Estado DI2: {1} Estado DI3: {2} Estado DI4: {3}'. At the bottom, there are five buttons: '< Back', 'Next >', 'Finish' (highlighted in blue), 'Cancel', and 'Help'.

Steps

1. File name
2. **Log event**

Log event

File

Interval: Single

☐ Remove

Number of files:

Type

☒ Time

☐ Change of state

☐ Trigger

Time

Period: 3 s

Variables

+ {0} = DI1
- {1} = DI2
↑ {2} = DI3
↓ {3} = DI4

Enable: TRUE ...

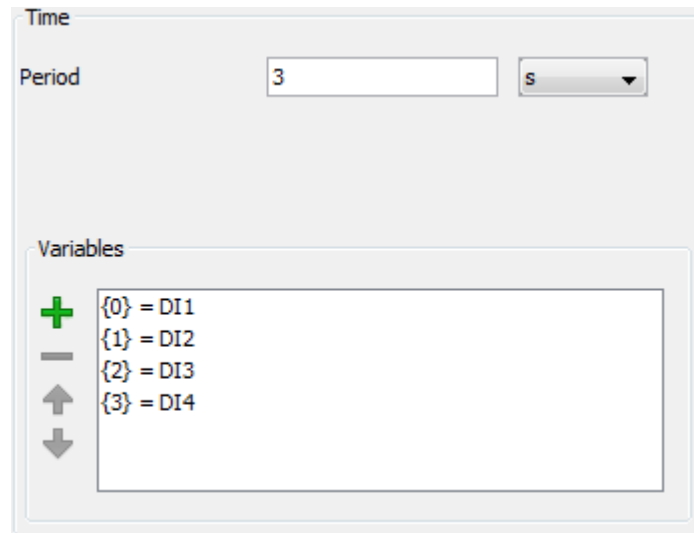
Text: Estado DI1: {0} Estado DI2: {1} Estado DI3: {2} Estado DI4: {3}

< Back Next > Finish Cancel Help

Tipos de Log de Eventos

Os *log* de eventos podem ser dos tipos **time**, **change of state** e **trigger**.

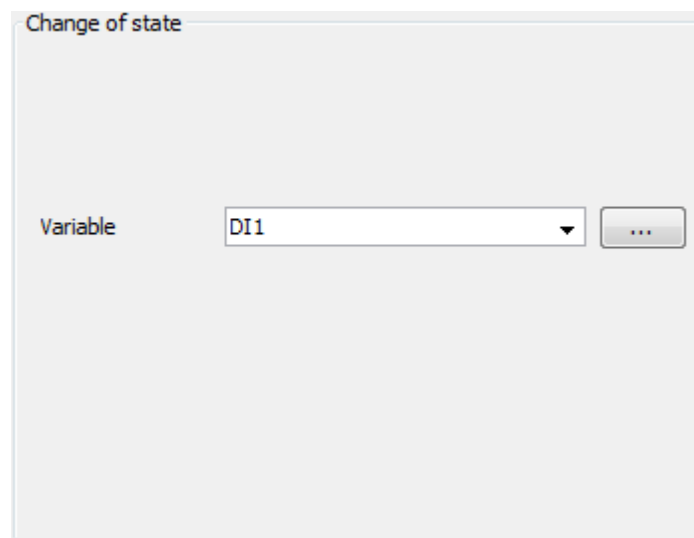
- **Time**



O *log* de eventos é gravado em intervalos de tempo definidos no campo **Período**. O valor configurado no campo **Período** tem que ser um valor inteiro maior que 0 e menor que 4294967295 s. A unidade de tempo pode ser selecionada entre segundos, minutos ou horas.

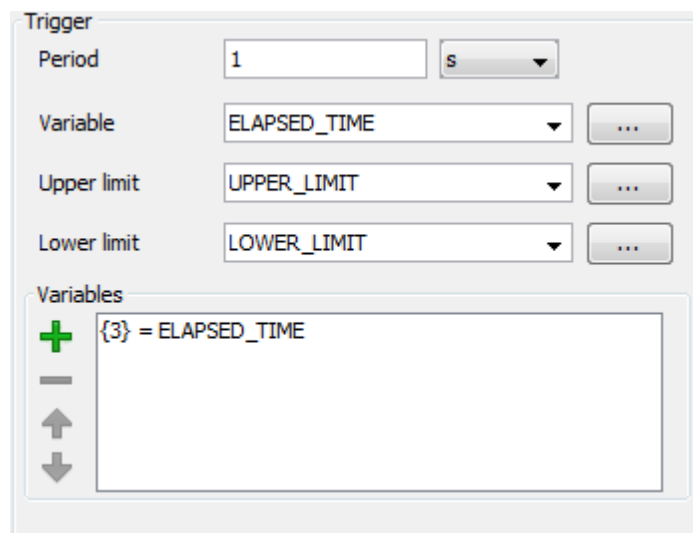
No campo **Variáveis** são selecionadas as variáveis que serão amostradas no período configurado.

- **Change of State**



O *log* de eventos é gravado após a mudança do valor da variável selecionada. Apenas o valor da variável selecionada é gravado no *log* de eventos.

- **Trigger**



The screenshot shows a configuration window titled 'Trigger'. It contains the following fields and controls:

- Period:** A text box with the value '1' and a dropdown menu set to 's'.
- Variable:** A dropdown menu with 'ELAPSED_TIME' selected and a button with three dots.
- Upper limit:** A dropdown menu with 'UPPER_LIMIT' selected and a button with three dots.
- Lower limit:** A dropdown menu with 'LOWER_LIMIT' selected and a button with three dots.
- Variables:** A list box containing the entry '{3} = ELAPSED_TIME'. To the left of the list are icons for adding (+), removing (-), and moving (up/down arrows).

Assim como no evento **Time**, o *log* de eventos é gravado em intervalos de tempo definidos no campo **Período**, mas somente quando o valor da variável configurada no campo **Trigger** estiver dentro dos limites selecionados nas variáveis dos campos **Limite superior** e **Limite inferior**. O valor configurado no campo **Período** tem que ser um valor inteiro maior que 0 e menor que 4294967295 s. A unidade de tempo pode ser selecionada entre segundos, minutos ou horas.

No campo **Variáveis** são selecionadas as variáveis que serão amostradas no período configurado.

Configuração do Campo Texto

No campo **Texto** é possível adicionar um texto, para ser apresentado juntamente com as variáveis. Este texto será adicionado assim que o arquivo de *log* de eventos for carregado (através do *upload* de arquivos de *log*) para o WPS.

Para que os valores das variáveis possam ser inseridos em qualquer posição do texto, são adicionados marcadores no texto que serão substituídos pelas variáveis.

Conforme o tipo de *log* de eventos selecionado, os marcadores devem ser adicionados da seguinte forma:

- **Time:** a primeira variável da lista é representada pelo marcador {0}, a segunda pelo marcador {1} e assim sucessivamente.
- **Change of state:** a única variável selecionada será representada pelo marcador {0}.
- **Trigger:** a variável **Trigger** é representada pelo marcador {0}, **Limite superior** pelo marcador {1}, **Limite inferior** pelo marcador {2}, a primeira variável da lista pelo marcador {3}, a segunda pelo marcador {4} e assim sucessivamente. Como exemplo segue abaixo a configuração de um *log* de eventos do tipo **Trigger** e seu arquivo de *log*.

WPS New File

Steps

1. File name
2. **Log event**

Log event

File

Interval

☐ Remove

Number of files

Type

☒ Time

☐ Change of state

☐ Trigger

Time

Period

Variables

{0} = DI1
{1} = DI2
{2} = DI3
{3} = DI4

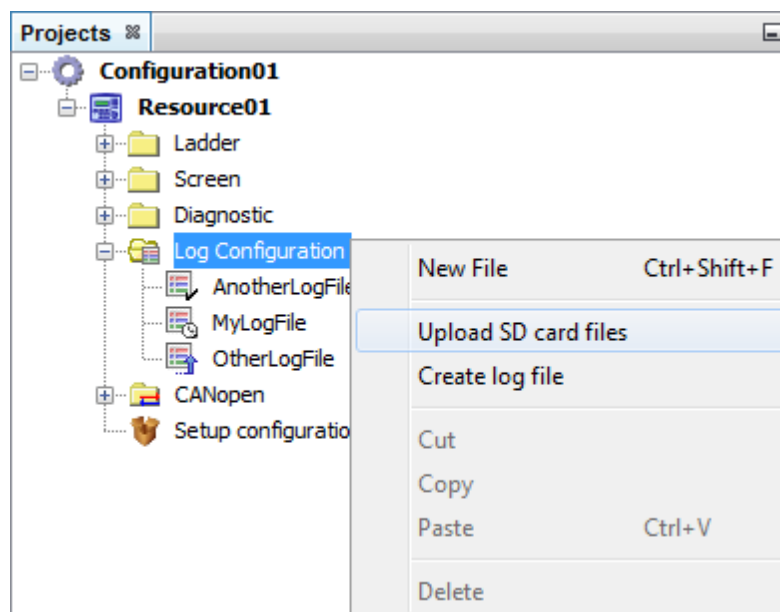
Enable

Text:

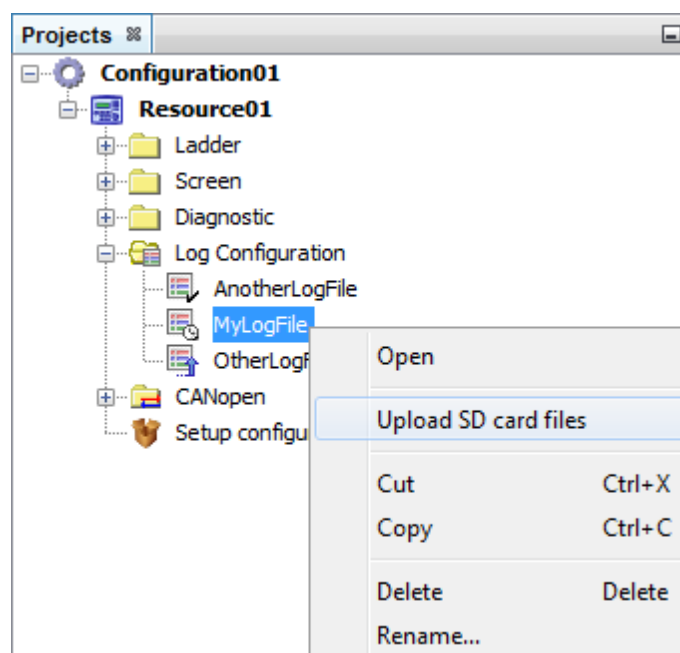
MyLogFile		
25/10/2562	03:03:20	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:03:31	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:03:34	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:03:37	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:03:40	Estado DI1: 1 Estado DI2: 1 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:03:43	Estado DI1: 1 Estado DI2: 1 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:03:46	Estado DI1: 0 Estado DI2: 1 Estado DI3: 1 Estado DI4: 0
25/10/2562	03:03:49	Estado DI1: 0 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:03:52	Estado DI1: 0 Estado DI2: 0 Estado DI3: 1 Estado DI4: 0
25/10/2562	03:03:55	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:03:58	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:04:01	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:04:04	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:04:07	Estado DI1: 0 Estado DI2: 1 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:04:10	Estado DI1: 0 Estado DI2: 1 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:13	Estado DI1: 0 Estado DI2: 1 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:16	Estado DI1: 1 Estado DI2: 0 Estado DI3: 1 Estado DI4: 0
25/10/2562	03:04:19	Estado DI1: 1 Estado DI2: 0 Estado DI3: 1 Estado DI4: 0
25/10/2562	03:04:22	Estado DI1: 1 Estado DI2: 0 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:04:25	Estado DI1: 1 Estado DI2: 1 Estado DI3: 1 Estado DI4: 0
25/10/2562	03:04:28	Estado DI1: 1 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:31	Estado DI1: 1 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:34	Estado DI1: 1 Estado DI2: 1 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:04:37	Estado DI1: 1 Estado DI2: 1 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:04:40	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:04:43	Estado DI1: 0 Estado DI2: 1 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:46	Estado DI1: 0 Estado DI2: 0 Estado DI3: 0 Estado DI4: 1
25/10/2562	03:04:49	Estado DI1: 0 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:52	Estado DI1: 0 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:55	Estado DI1: 0 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:04:58	Estado DI1: 0 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:05:01	Estado DI1: 0 Estado DI2: 0 Estado DI3: 1 Estado DI4: 1
25/10/2562	03:05:04	Estado DI1: 0 Estado DI2: 1 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:05:07	Estado DI1: 0 Estado DI2: 1 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:05:10	Estado DI1: 0 Estado DI2: 1 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:05:13	Estado DI1: 0 Estado DI2: 1 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:05:16	Estado DI1: 0 Estado DI2: 1 Estado DI3: 0 Estado DI4: 0
25/10/2562	03:05:19	Estado DI1: 0 Estado DI2: 1 Estado DI3: 0 Estado DI4: 0

Upload de Arquivos de Log

Existem duas formas de realizar o *upload* dos arquivos de log: *upload* de todos os arquivos do projeto ou *upload* dos arquivos individualmente.



Para realizar o *upload* de todos os arquivos de *log* configurados no projeto, é preciso selecionar a pasta **Log de evento** com o botão direito e selecionar a opção **Upload arquivos SD card**.



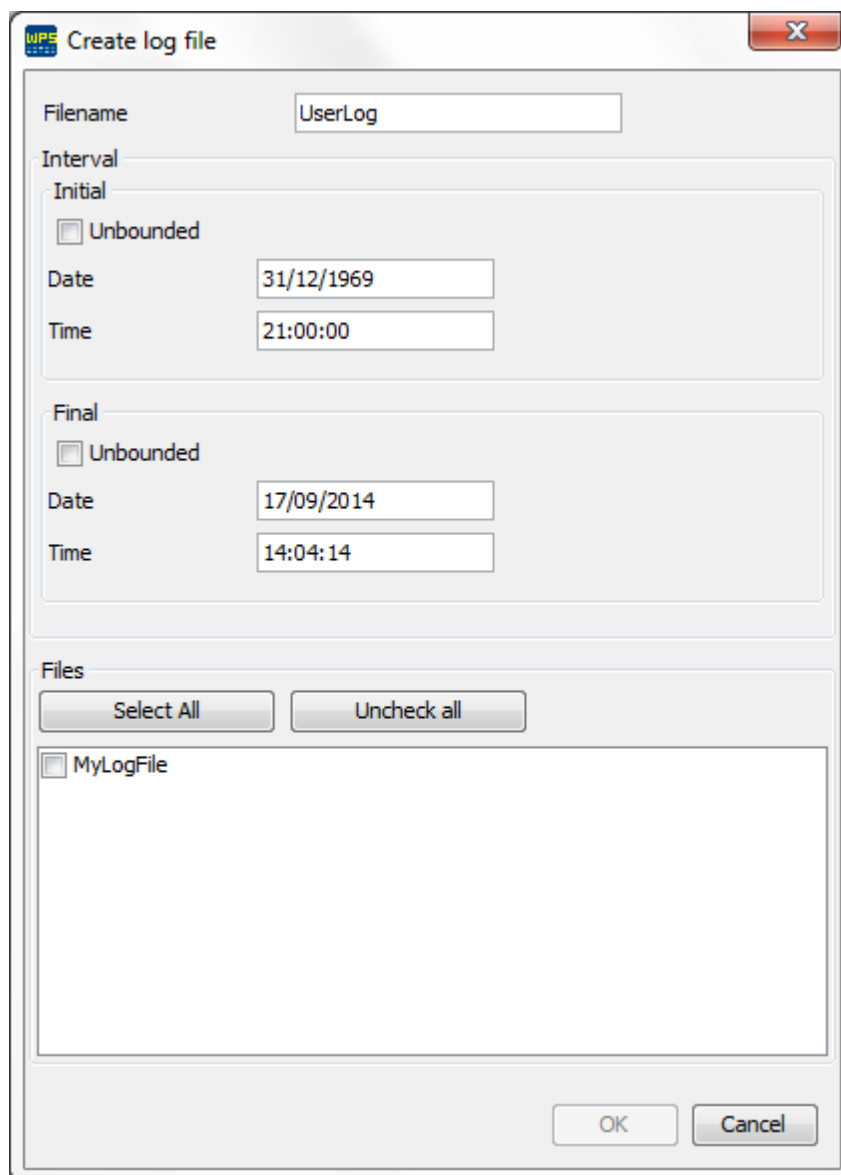
Para realizar o *upload* dos arquivos associados a apenas um arquivo de configuração de *log* de eventos é preciso clicar com o botão direito no arquivo de configuração do *log* de eventos e selecionar a opção **Upload arquivos SD card**.

**ATENÇÃO!**

Ao fazer o *upload* de arquivos os arquivos anteriores serão sobrescritos.

Criação de Arquivos de Log

Através da ferramenta de criação de arquivos de *log* é possível criar novos arquivos de *log* de eventos utilizando os arquivos de *log* previamente carregados no projeto. Com isso é possível definir o intervalo inicial e final do *log* e quais os arquivos de *log* que devem ser utilizados. Para iniciar a criação de um novo arquivo de *log*, é preciso clicar com o botão direito do mouse na pasta **Log de evento** e selecionar a opção **Criar arquivo de log**. As opções para configuração do novo arquivo são as seguintes:



- **Nome do arquivo:** Nome que será utilizado no arquivo de *log* criado. Caso o nome já exista, os dados serão sobrescritos.
- **Intervalo inicial:** Determina o dia e hora inicial dos *logs* do arquivo. Caso a opção **Ilimitado** seja selecionada, não haverá um valor mínimo para a data.
- **Intervalo final:** Determina o dia e hora final dos *logs* do arquivo. Caso a opção **Ilimitado** seja selecionada, não haverá um valor máximo para a data.

- **Arquivos:** Seleciona os arquivos de *log* que serão utilizados para gerar o novo arquivo de *log*.

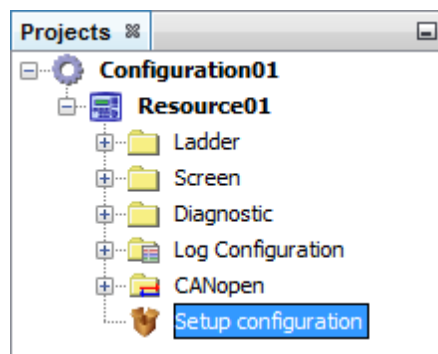
10.1.8 Setup

[Configuração](#)

[Janelas de Configuração](#)

10.1.8.1 Configuração

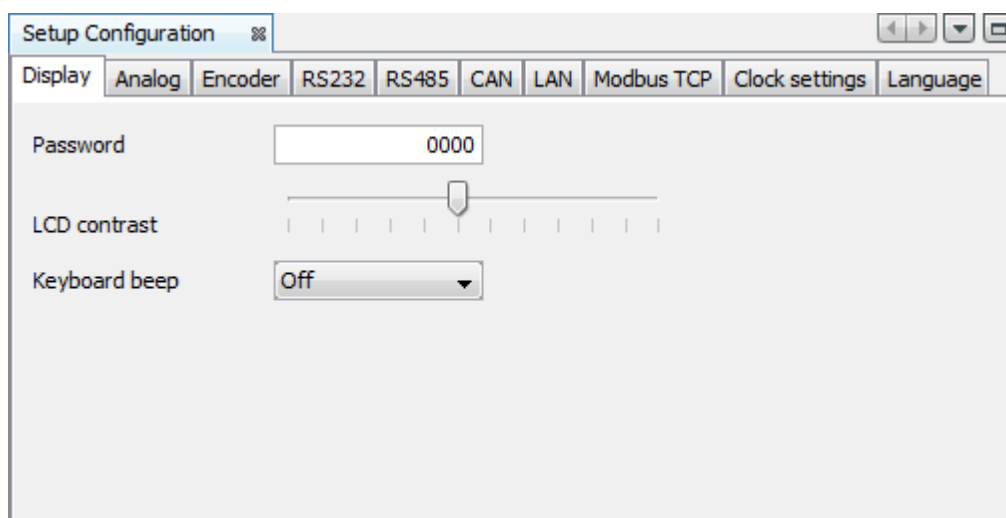
A configuração do *setup* do PLC300 é acessada através do clique duplo no atalho disponível no recurso conforme figura a seguir.



ATENÇÃO!

Ao criar um novo recurso no WPS os valores da configuração de *setup* serão os valores padrões do PLC300, recomendamos a revisão desses valores conforme a necessidade da aplicação e o envio desses ajustes conforme explicação a seguir.

A janela de configuração de *setup* é dividida em abas e possui dois botões para acesso ao PLC300 conforme figura a seguir.



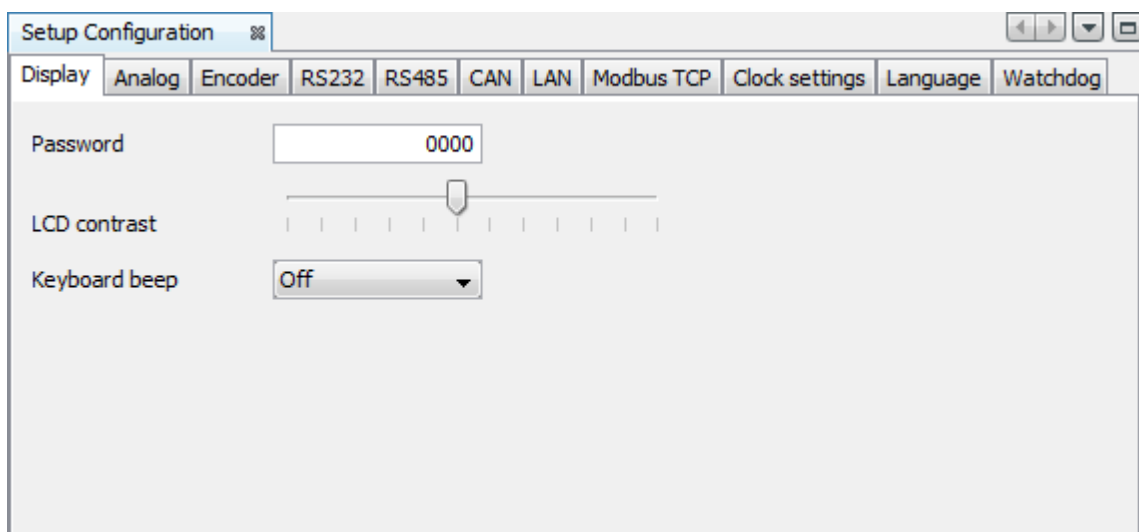
O botões de **Escreve Configuração** e **Lê Configuração** só estarão ativos quando o WPS estiver conectado ao PLC300 através do comando **Conectar Equipamento** .

- **Escreve Configuração:** Envia todas as configurações ajustadas nas telas da configuração de *setup* para o equipamento.
- **Lê Configuração:** Recebe todas as configurações ajustadas no equipamento e configura as telas conforme os valores recebidos.

10.1.8.2 Janelas de Configuração

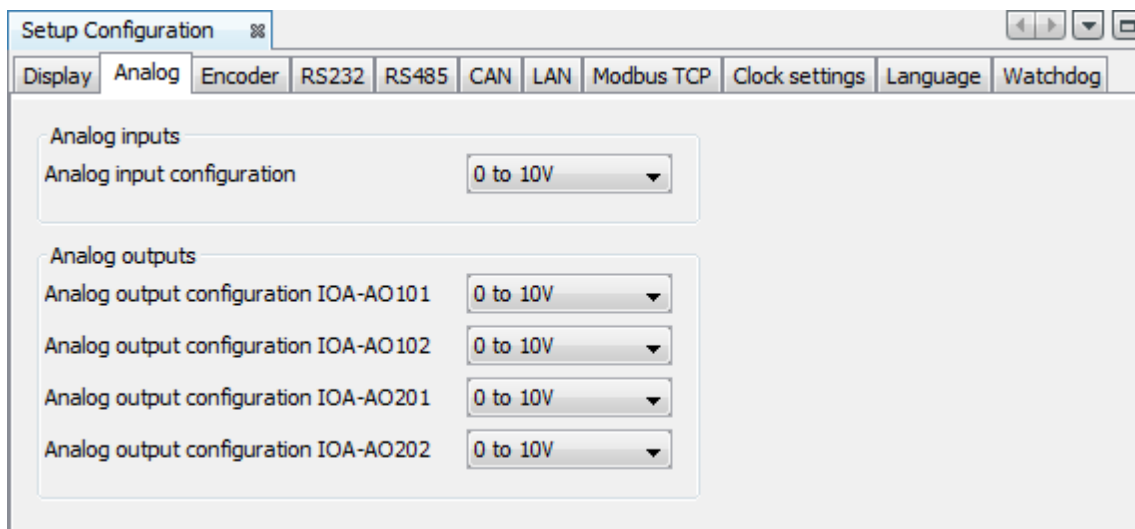
Ver a seguir.

10.1.8.2.1 Display



- **Senha:** Permite alterar a senha da função *setup* do PLC300. A senha padrão é '0000'. A nova senha deve ser um número com quatro algarismos.
- **Contraste LCD:** ajuste do contraste do LCD do PLC300.
- **Beep do Teclado:** Habilita o beep das teclas do PLC300.

10.1.8.2.2 Analógicas



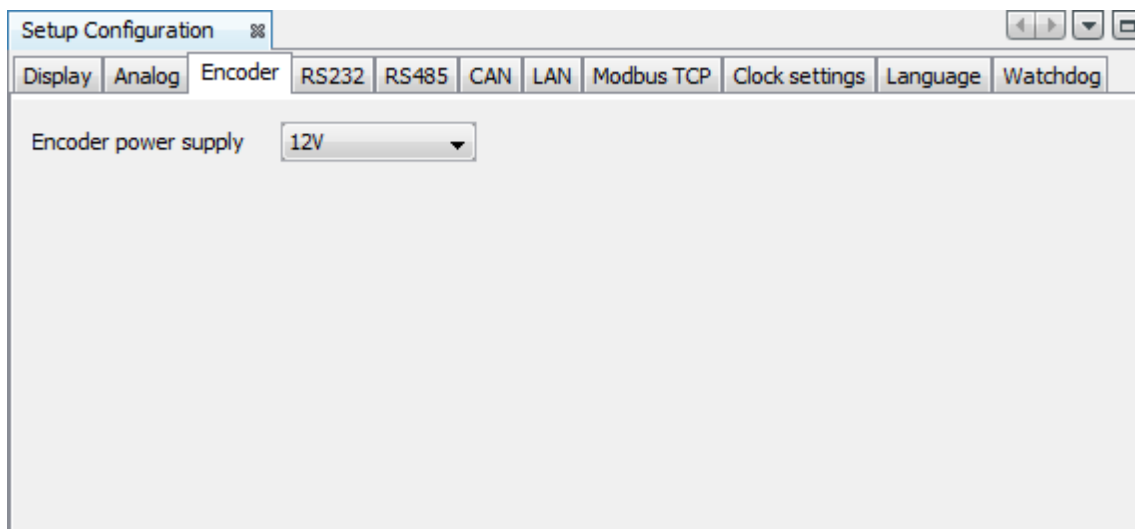
- **Entradas Analógicas:** Seleciona um dos três modos de operação da entrada analógica AI1 do PLC300: 'Tensão 0 a 10 V', 'Corrente 0 a 20 mA' ou 'Corrente 4 a 20 mA'.

**NOTA!**

Na opção 4 a 20 mA, o valor que o Ladder enxerga é um valor proporcional, normalizado, ou seja 4 a 20 mA ' 0 a 32767.

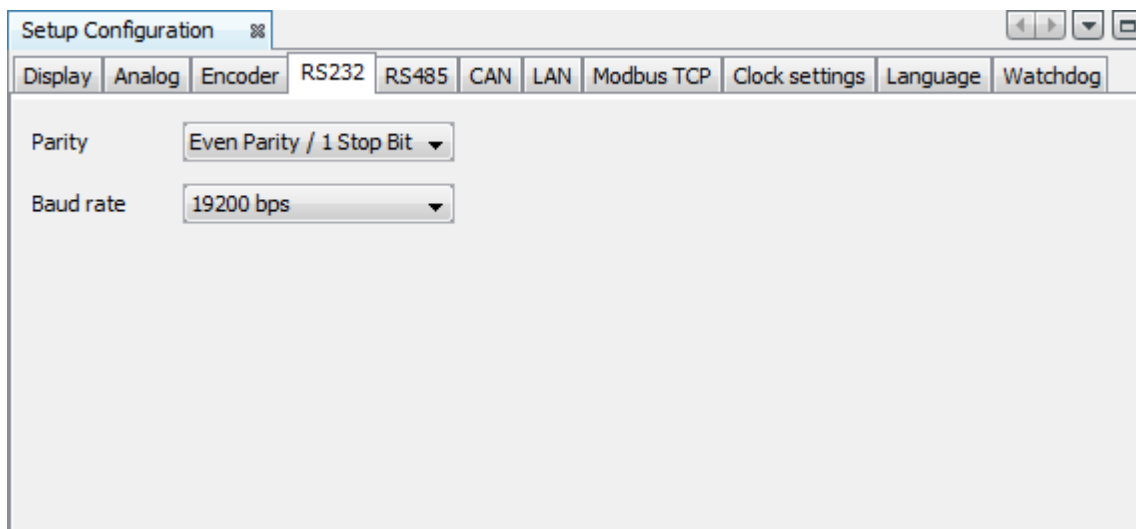
- **Saídas Analógicas:** Seleciona um dos quatro modos de operação das saídas analógicas (AOs) dos acessórios IOA instalados no PLC300, sendo que as AOs 101 e 102, são as AOs 1 e 2 do cartão IOA instalado no slot 1 do PLC300 e as AOs 201 e 202, são as AOs 1 e 2 do cartão IOA instalado no slot 2.

10.1.8.2.3 Encoder



Seleciona a tensão de alimentação do encoder do PLC300 entre 5 ou 12V.

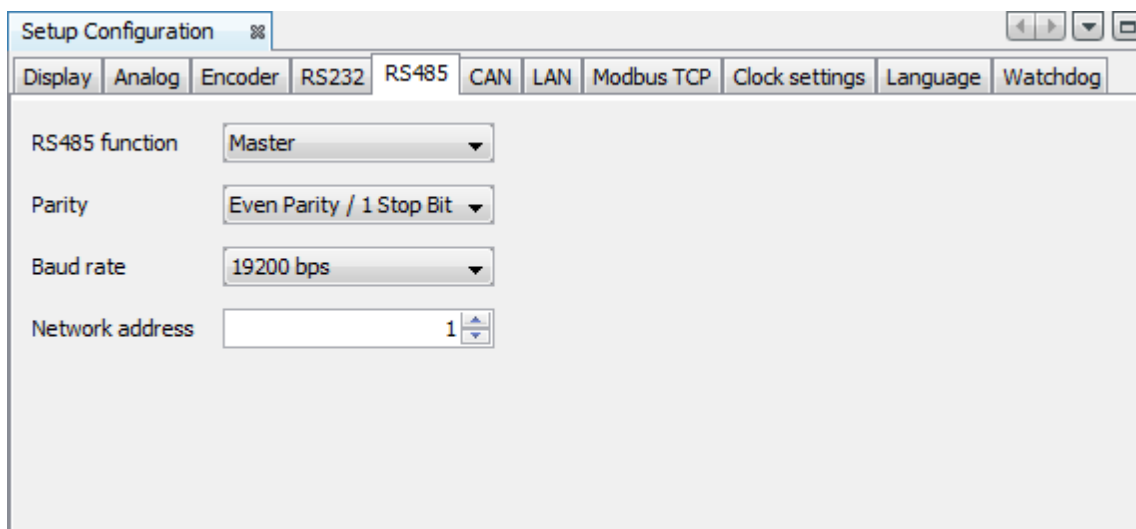
10.1.8.2.4 RS232



The screenshot shows the 'Setup Configuration' window with the 'RS232' tab selected. The window has a title bar with a close button and navigation arrows. Below the title bar is a tabbed interface with tabs for 'Display', 'Analog', 'Encoder', 'RS232', 'RS485', 'CAN', 'LAN', 'Modbus TCP', 'Clock settings', 'Language', and 'Watchdog'. The 'RS232' tab is active, showing two settings: 'Parity' set to 'Even Parity / 1 Stop Bit' and 'Baud rate' set to '19200 bps'.

É possível configurar o baud rate, a paridade e o número de stop bits da interface serial RS232 do PLC300.

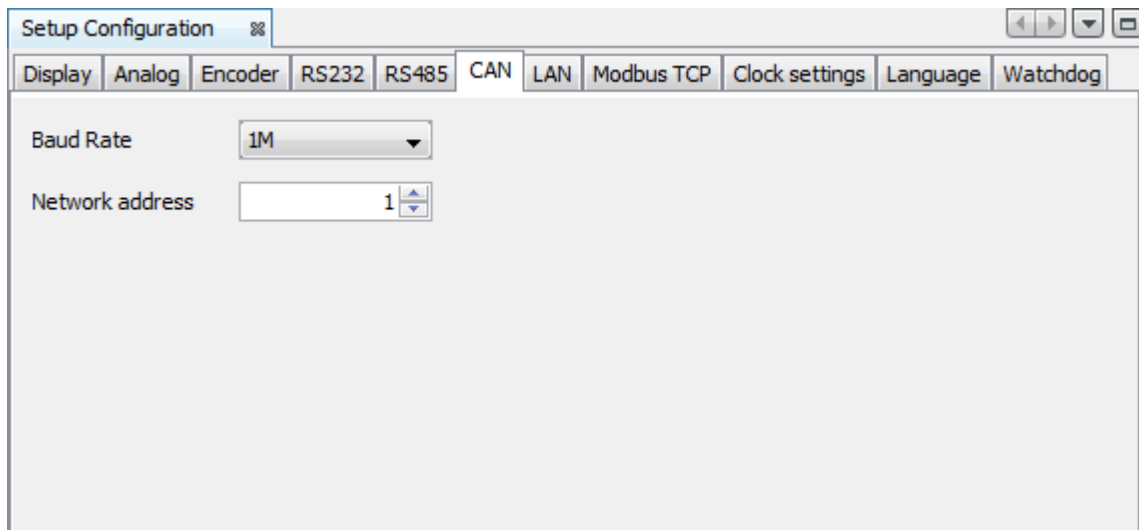
10.1.8.2.5 RS485



The screenshot shows the 'Setup Configuration' window with the 'RS485' tab selected. The window has a title bar with a close button and navigation arrows. Below the title bar is a tabbed interface with tabs for 'Display', 'Analog', 'Encoder', 'RS232', 'RS485', 'CAN', 'LAN', 'Modbus TCP', 'Clock settings', 'Language', and 'Watchdog'. The 'RS485' tab is active, showing four settings: 'RS485 function' set to 'Master', 'Parity' set to 'Even Parity / 1 Stop Bit', 'Baud rate' set to '19200 bps', and 'Network address' set to '1'.

É possível configurar o baud rate, a paridade o número de stop bits, o modo (mestre/escravo) e o endereço do PLC300 numa rede ModbusRTU, através interface serial RS485 do PLC300.

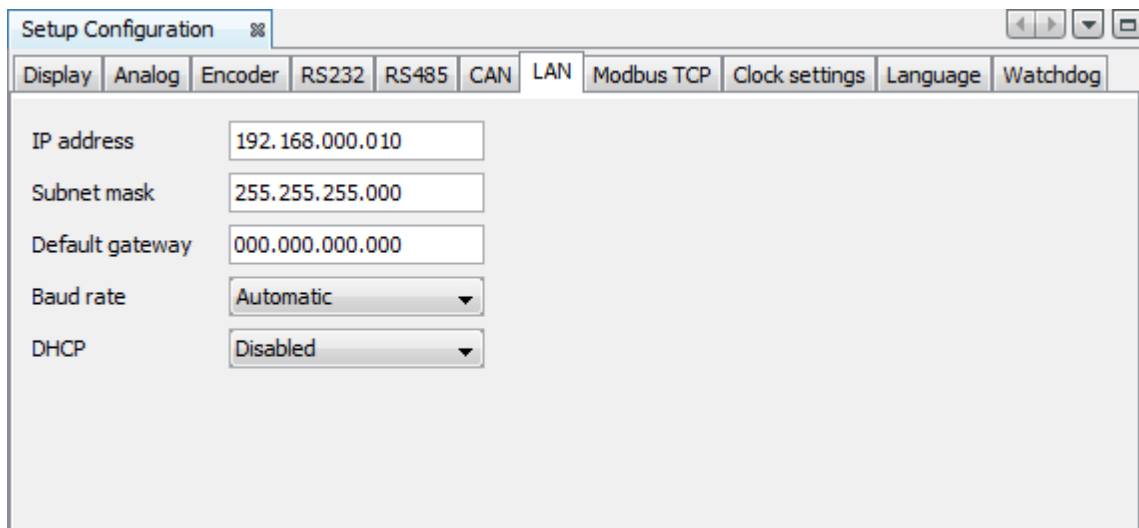
10.1.8.2.6 CAN



The screenshot shows the 'Setup Configuration' window with the 'CAN' tab selected. The window has a title bar with a close button and navigation arrows. Below the title bar is a tabbed interface with tabs for Display, Analog, Encoder, RS232, RS485, CAN, LAN, Modbus TCP, Clock settings, Language, and Watchdog. The CAN tab is active, showing two configuration fields: 'Baud Rate' with a dropdown menu set to '1M', and 'Network address' with a numeric input field set to '1'.

É possível configurar o baud rate e o endereço do PLC300 numa rede CANopen, através da interface CAN do PLC300.

10.1.8.2.7 LAN

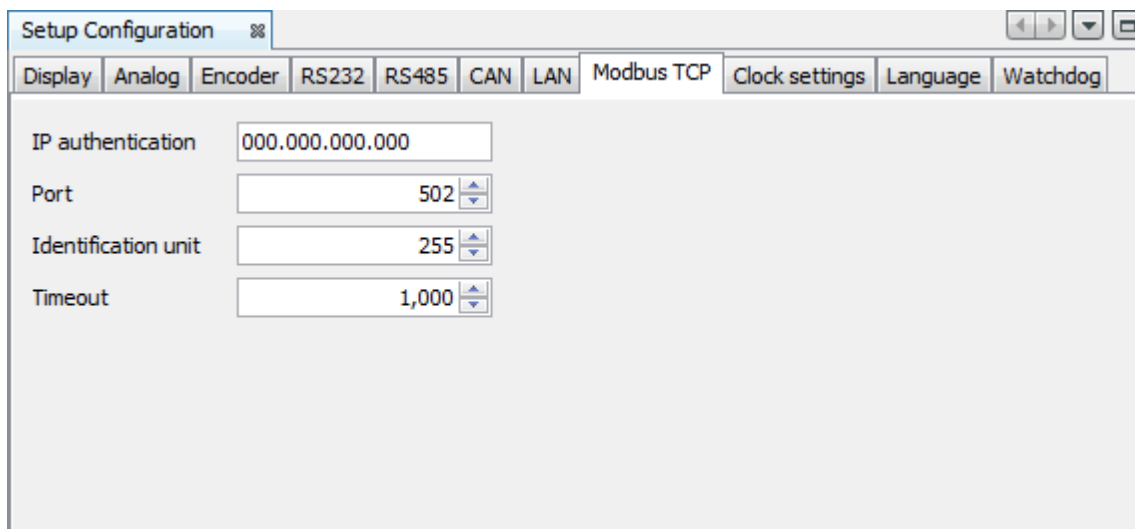


The screenshot shows the 'Setup Configuration' window with the 'LAN' tab selected. The window has a title bar with a close button and navigation arrows. Below the title bar is a tabbed interface with tabs for Display, Analog, Encoder, RS232, RS485, CAN, LAN, Modbus TCP, Clock settings, Language, and Watchdog. The LAN tab is active, showing five configuration fields: 'IP address' (192.168.000.010), 'Subnet mask' (255.255.255.000), 'Default gateway' (000.000.000.000), 'Baud rate' (Automatic), and 'DHCP' (Disabled).

É possível configurar endereço IP, máscara de sub-rede, gateway padrão, DHCP, velocidade e modo duplex do PLC300 numa rede Ethernet.

- **Endereço IP:** 4 bytes de endereço que identificam o PLC300 na rede IP;
- **Máscara de Sub-rede:** 4 bytes que identificam a sub-rede ao qual pertence o PLC300 na rede IP;
- **Gateway:** 4 bytes de endereço que identificam o gateway padrão para acesso a outras sub-redes na rede IP;
- **DHCP:** Desabilitado, Habilitado;
- **Speed/Duplex:** Auto, 10MBps Full Duplex, 10MBps Half Duplex, 100MBps Full Duplex, 100MBps Half Duplex;

10.1.8.2.8 Modbus TCP



Setup Configuration ⓘ

Display Analog Encoder RS232 RS485 CAN LAN Modbus TCP Clock settings Language Watchdog

IP authentication 000.000.000.000

Port 502

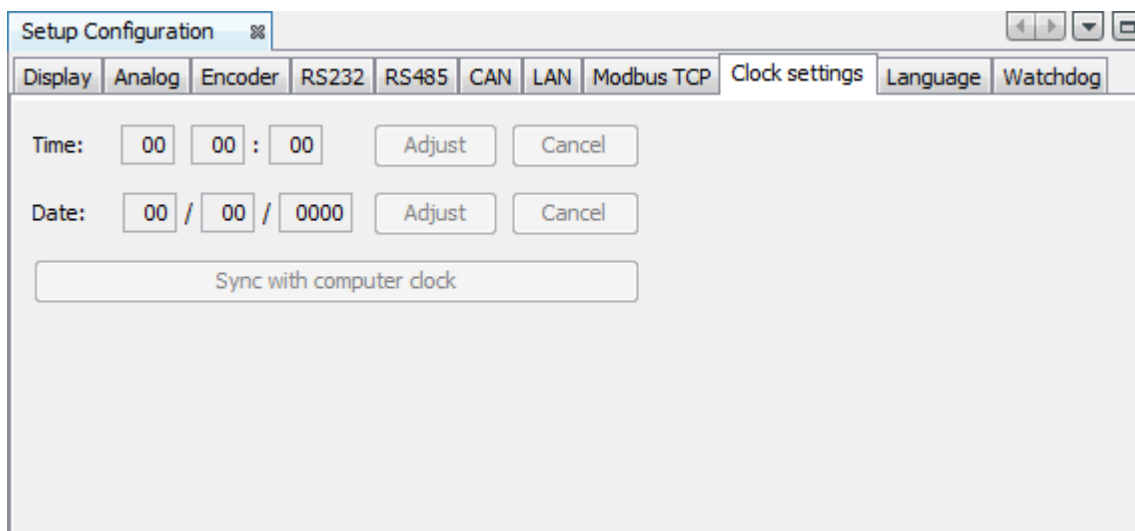
Identification unit 255

Timeout 1,000

É possível configurar porta TCP, Unit ID, autenticação de IP e timeout do Gateway Modbus TCP/RTU do PLC300 numa rede Ethernet utilizando o protocolo ModbusTCP.

- **Autenticação IP:** 4 bytes de endereço que identificam o único endereço IP remoto que pode se conectar ao PLC300. Todos os campos em zero desabilitam a autenticação de IP e qualquer endereço remoto pode se conectar ao PLC300;
- **Porta TCP:** 0 a 65535,
- **Unit ID:** 1 a 255,
- **Gateway Timeout:** 20 a 5000 ms.

10.1.8.2.9 Ajustes do Relógio



Setup Configuration ⓘ

Display Analog Encoder RS232 RS485 CAN LAN Modbus TCP Clock settings Language Watchdog

Time: 00 : 00 : 00 Adjust Cancel

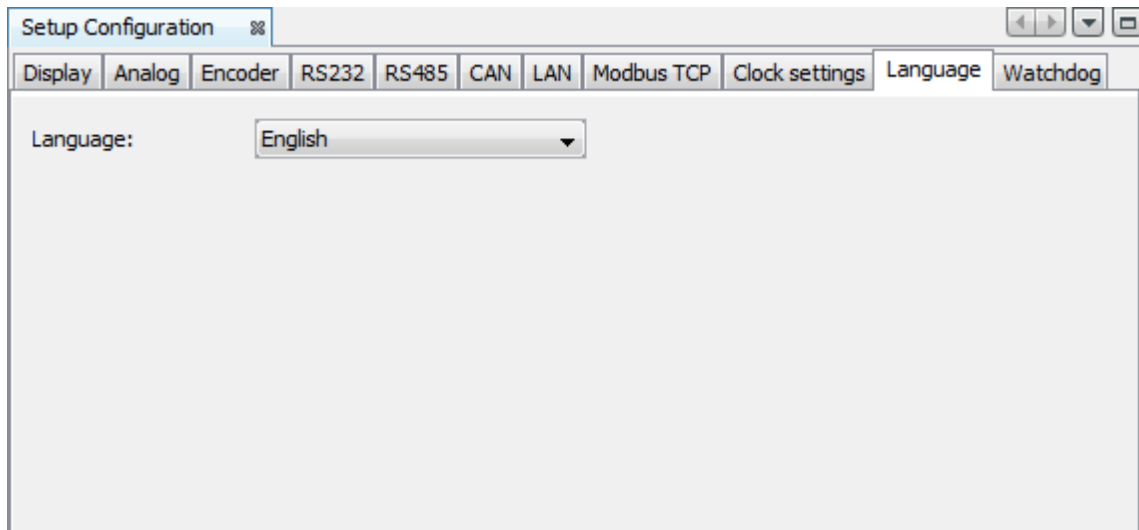
Date: 00 / 00 / 0000 Adjust Cancel

Sync with computer clock

Os ajustes do relógio só estarão ativos quando o WPS estiver conectado ao PLC300 através do comando **Conectar Equipamento**.

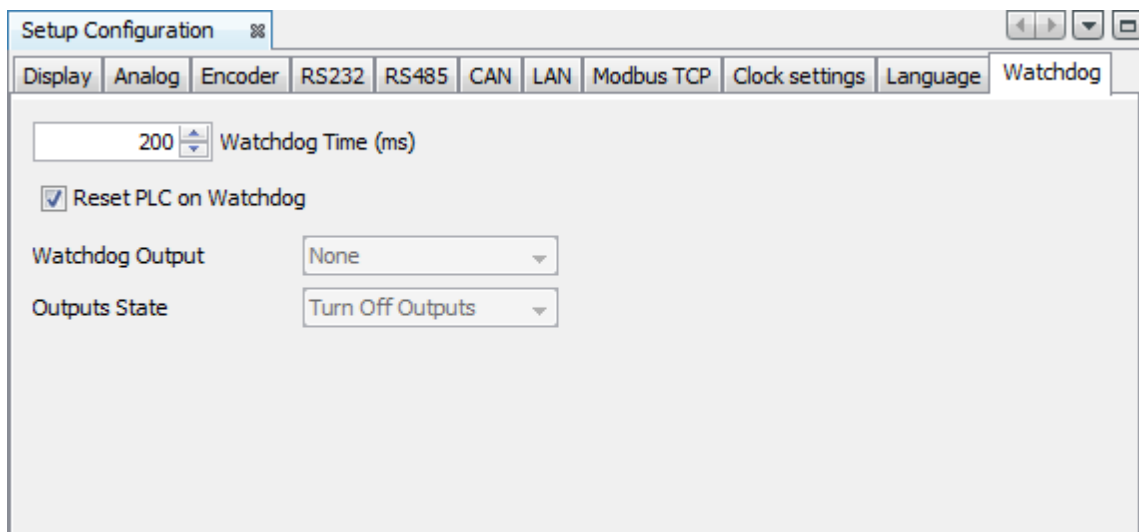
Permite o ajuste do relógio RTC do PLC300.

10.1.8.2.10 Idioma



Permite alterar o idioma configurado no PLC300.

10.1.8.2.11 Watchdog



Permite configurar *watchdog* personalizado no PLC300.

- **Tempo do Watchdog (ms):** tempo de espera do *watchdog*. Valor mínimo: 300ms.
- **Resetar o PLC em caso de Watchdog:** se marcada, esta opção reseta o PLC300 em caso de *watchdog* e roda novamente o aplicativo do usuário após a reinicialização. Marcando esta opção, as opções 3 e 4 serão desabilitadas automaticamente;
- **Saída Watchdog:** permite selecionar uma saída para uso exclusivo como *flag* do *watchdog*.
 - o A saída selecionada (DO8 ou DO9) não poderá ser escrita via Ladder. Ela permanecerá em nível alto depois de configurada para esta opção e, em caso de ocorrência do *watchdog*, ela será desligada.
- **Estado das Saídas:** permite selecionar se as saídas digitais manterão seu estado em caso de ativação do

watchdog ou se elas serão desligadas.

- o Esta opção não terá ação sobre a saída selecionada na opção 3. Por exemplo: se na opção 2 for selecionada a DO9, mesmo que na opção 3 esteja marcado **Manter o estado**, no momento do *watchdog* ela será desligada.

Compatibilidade

Equipamento	Versão
PLC300	2.30 ou superior

10.1.9 Comunicação

Ver a seguir.

10.1.9.1 Comandos online

Visão Geral

Os comandos on-line são comandos realizados quando o WPS estiver conectado ao PLC300 através do comando **Conectar Equipamento**.

Nos comandos on-line é possível:

- Realizar os comandos de gravação e carregamento dos arquivos do recurso, *setup* e *firmware*.
- Configurar o número do equipamento.
- Parar e executar o programa e apagar o recurso.

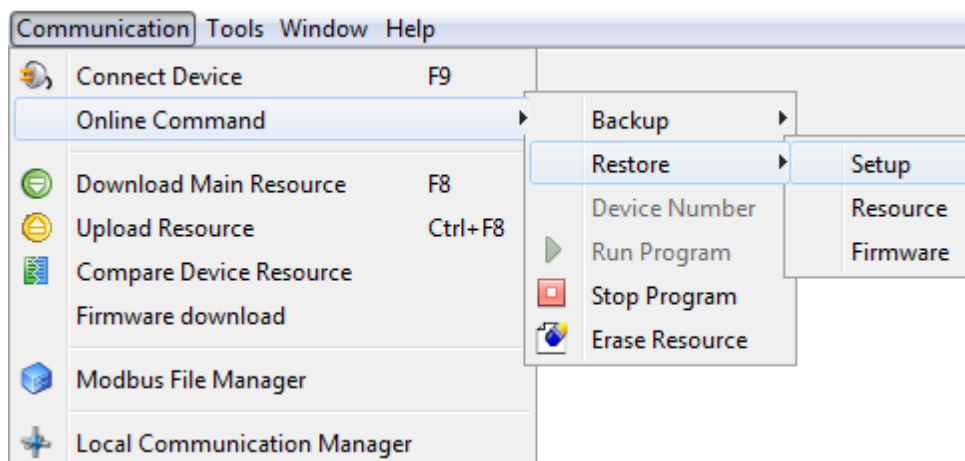
Configuração da gravação e carregamento

Os comandos de gravação dos arquivos do recurso, *setup* e *firmware* fazem a cópia dos arquivos binários do equipamento para o SD card. O carregamento dos arquivos fazem a cópia dos arquivos binários do SD card para o equipamento.

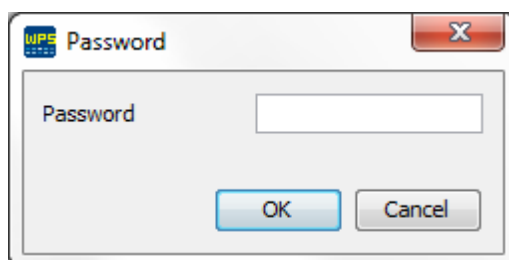
Para segurança dos arquivos, é possível configurar uma senha de proteção. É possível também configurar o número do equipamento para que os arquivos de backup sejam copiados para diferentes pastas.

Comando de gravação e carregamento

Para realizar os comandos de gravação e carregamento do recurso, *setup* e *firmware* é necessário selecionar o menu **Comunicação > Comandos on-line** > escolher uma das opções listadas **Grava/Carrega** e depois **Recurso/Setup/Firmware**.

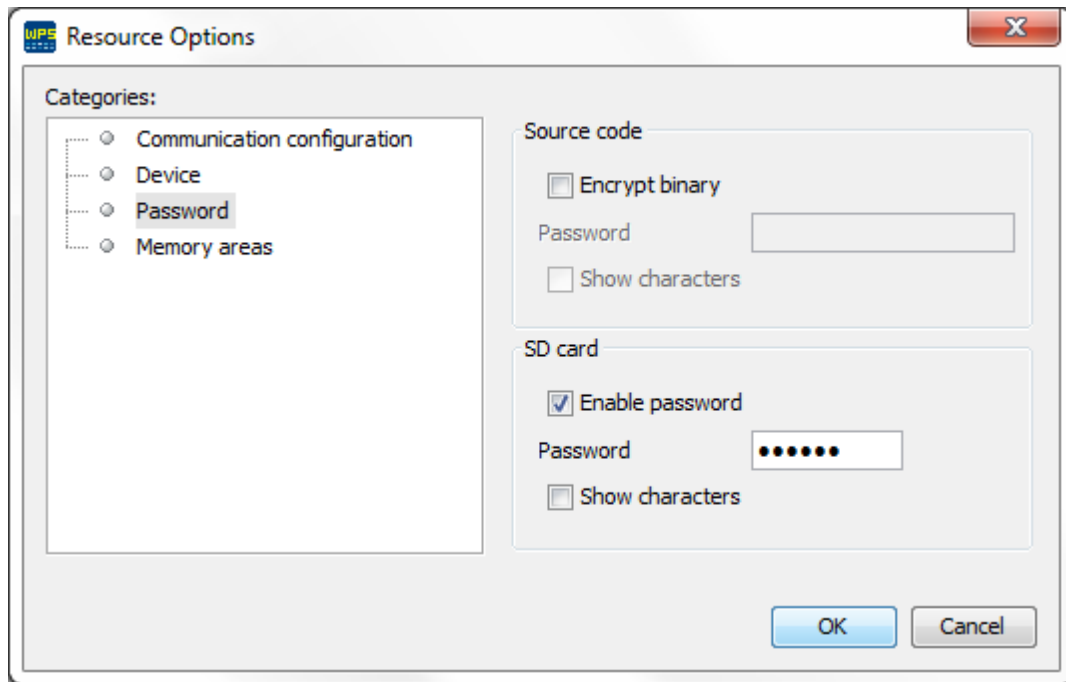


Caso uma senha tenha sido configurada para o recurso, uma janela de diálogo solicitando a senha deverá aparecer. A senha possui de 1 a 8 dígitos decimais.



Configuração de senha

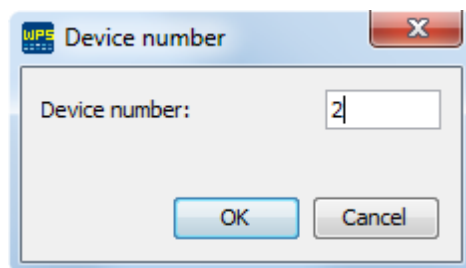
A configuração da senha para os comandos on-line é feita através da configuração das propriedades do recurso, na opção senhas.



Após digitar a senha (de 1 - 8 dígitos decimais), é preciso construir o recurso e enviá-la ao equipamento. A senha ficará armazenada no equipamento e, quando um dos comandos on-line é realizado, a senha é solicitada. A senha não será solicitada novamente até que a monitoração on-line seja finalizada.

Número do equipamento

O número do equipamento é configurado para a gravação do backup de diferentes equipamentos em um mesmo SD card. Ao selecionar as funções de gravação de recurso, *setup* ou *firmware*, os arquivos são gravados nas seguintes pastas, de acordo com o número do equipamento (1 no exemplo).



\\PLC300\\0001\\Resource
\\PLC300\\0001\\Setup
\\PLC300\\0001\\Firmware



ATENÇÃO!

Os arquivos existentes nas pastas de destino serão sobrescritos ao executar as funções de gravação.

Executar programa 

Executa o programa do usuário.

Para realizar o comandos de executar programa é necessário selecionar o menu **Comunicação > Comandos on-line > Executar programa.**

Parar programa



Para o programa do usuário.

Para realizar o comandos de executar programa é necessário selecionar o menu **Comunicação > Comandos on-line > Parar programa.**

Apagar recurso



Apaga o recurso gravado no equipamento.

Para realizar o comandos de executar programa é necessário selecionar o menu **Comunicação > Comandos on-line > Apagar programa.**

10.1.9.2 Forçar E/S

Visão Geral

A janela de forçar entradas e saídas é utilizada para que os valores das entradas digitais e analógicas sejam lidos pelo programa, por valores manipulados pelo usuário, independente dos seus estados físicos. Também permite a manipulação dos estados físicos das saídas digitais e analógicas pelo usuário, de forma independente aos valores calculados pelo programa.

Para forçar as entradas e saídas do equipamento, é necessário que a monitoração on-line esteja ativa e a opção de **Rodar Ciclicamente** esteja habilitada. Os dados são enviados ao equipamento a cada 2 segundos.

Os valores podem ser editados sem que o equipamento esteja conectado. As configurações ficam armazenadas no recurso e são gravadas sempre que a seleção de recurso principal é alterada.

Os dados apresentados na janela de forçar E/S contêm os valores pertencentes ao recurso (e configuração) selecionado como principal.

A janela de forçar E/S é aberta através do menu **Ladder > Forçar E/S:**

Ladder	Communication	Tools	Window	Help
Insert new rung at the end		Ctrl+Alt+N		
Insert new rung below		Ctrl+Alt+B		
Insert new rung above		Ctrl+Alt+A		
Go to rung below		Ctrl+Down		
Go to rung above		Ctrl+Up		
Delete selected rung		Ctrl+Delete		
Select rung below		Shift+Down		
Select rung above		Shift+Up		
Select next element		Right		
Select previous item		Left		
Select element below		Down		
Select item above		Up		
Insert contact		C		
Insert block (Function/Functional)		F		
Insert output		O		
Change ladder element		X		
Select group of elements		G		
Select all elements		Ctrl+A		
Delete element				
Edit arguments		F2		
Edit rung title		Shift+F2		
Replace...				
Cross-reference				
 Force I/O				

Barra de ferramentas

A barra de ferramentas da janela de forçar possui as opções de rodar ciclicamente, *upload* da configuração de forçar do equipamento, habilitar todos e desabilitar todos:



Rodar ciclicamente: Envia os as configurações do usuário para o equipamento e atualiza o estado das entradas e saídas de maneira cíclica.



Upload da configuração: Permite que a configuração atual do equipamento seja lida. Para esta opção estar habilitada, é necessário que a monitoração on-line esteja ativa e a opção rodar ciclicamente desativada.

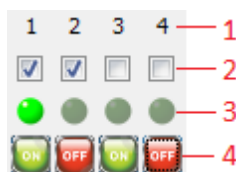


Habilitar todos: Habilita o forçar E/S de todas as entradas e saídas do equipamento.

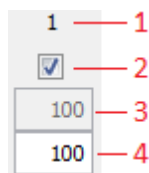
✖ Desabilitar todos: Desabilita o forçar E/S de todas as entradas e saídas do equipamento.

Comandos das entradas e saídas

Para cada entrada e saída digital e analógica estão associados uma caixa de seleção para habilitar o forçar, um campo de status e um campo para edição.



1. Número da entrada/saída digital
2. Habilita/desabilita o forçar E/S
3. Estado atual da E/S: Possui três estados: 1. led verde claro: acionado; 2. led verde escuro: desacionado; 3. led cinza: o valor não está sendo lido.
4. Habilita/desabilita a entrada/saída



1. Número da entrada/saída analógica
2. Habilita/desabilita o forçar E/S
3. Valor atual da entrada/saída
4. Valor da entrada/saída configurada pelo usuário

10.1.9.3 Download

O *download* do recurso realiza o *download* dos arquivos de configuração das áreas de memória, dos dados voláteis, dos dados retentivos, dos programas, das telas, alarmes, código fonte e receitas para a memória interna do equipamento, configuração do *setup* e configuração da rede CANopen.

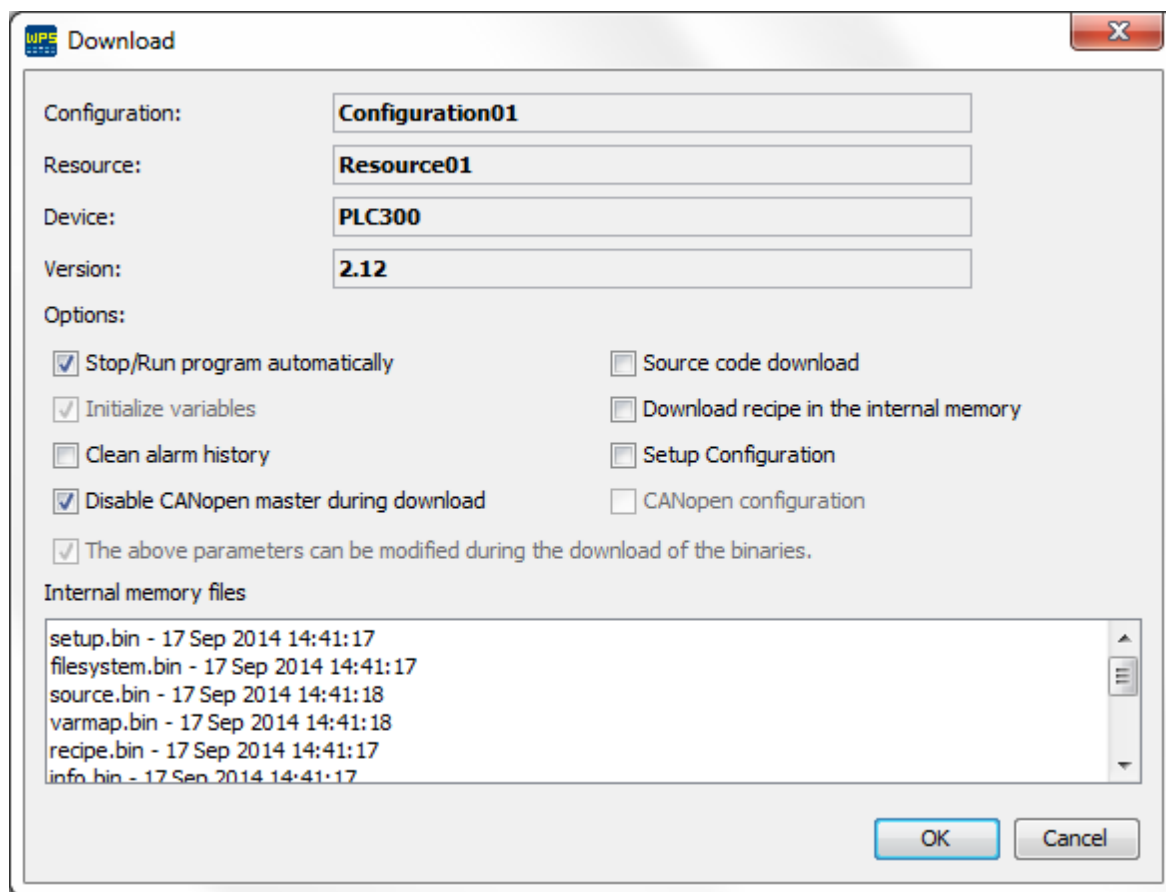


Figura 1 - Janela de download do recurso

É possível também realizar as seguintes operações durante o *download* do recurso:

- **Inicializar variáveis:** Inicializa as variáveis voláteis, retentivas e de saída. Esta opção somente está disponível a partir da versão 1.70 do *firmware*, nas versões mais antigas as variáveis sempre são inicializadas a cada *download*.
- **Limpar histórico de alarmes:** Exclui os dados armazenados do histórico de alarmes. É recomendado limpar esta área sempre que as áreas de memória são alteradas ou a configuração dos alarmes tenha sido alterada.
- **Download do código fonte:** Realiza o *download* do código fonte do recurso.
- **Download de receitas na memória interna:** Realiza o *download* do arquivo contendo os dados das receitas configuradas com a opção de armazenamento na RAM.
- **Desabilitar mestre CANopen durante o download:** Desabilita o mestre CANopen durante o *download*.
- **Parar/Executar o programa automaticamente:** Desabilita as janelas de aviso de que o programa será parado (Figura 2) e a janela de pergunta se é desejado executar o programa (Figura 3).
- **Configuração de setup:** Realiza o *download* dos dados da configuração *setup*.
- **Configuração CANopen:** Realiza o *download* da configuração da rede CANopen.

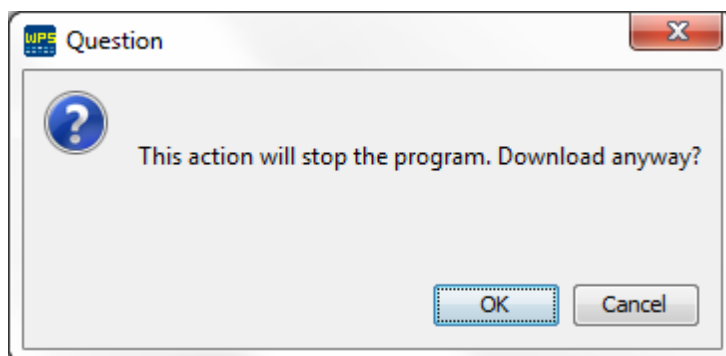


Figura 2 - Janela de aviso que o programa será parado

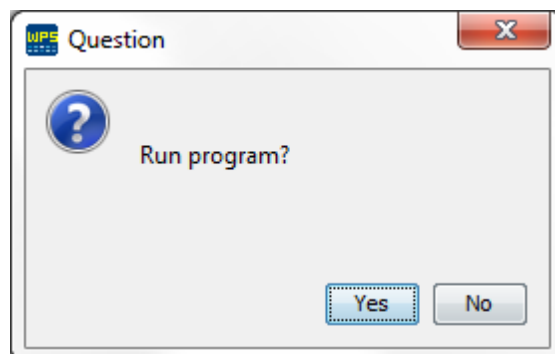


Figura 3 - Janela de pergunta executar o programa

Download dos arquivos SD card

Caso o usuário tenha configurado uma ou mais receitas com a opção de armazenamento no SD card, a janela de confirmação de *download* de arquivos no SD card irá aparecer (Figura 4). Para que a janela apareça, é necessário que o SD card esteja conectado ao equipamento.

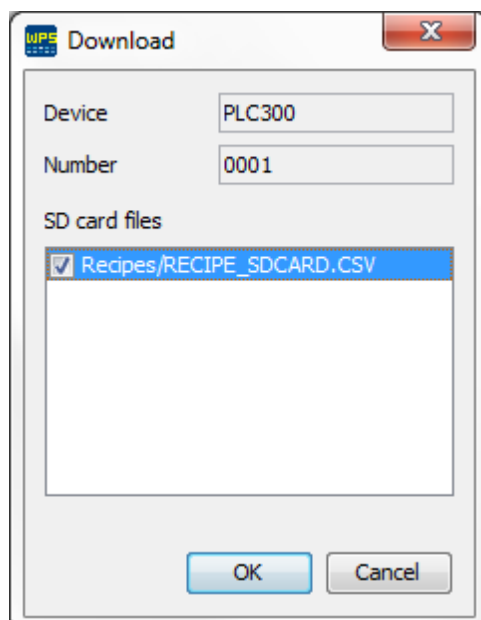


Figura 4 - Janela de confirmação de download no SD card

10.1.9.4 Hot Download

Ver a seguir.

10.1.9.4.1 Visão Geral

O Hot Download é uma funcionalidade que permite o carregamento das alterações de lógica e configurações de um recurso em um equipamento, sem que ocorra a parada de sua rotina de execução. Os valores atuais das variáveis são mantidos.

No Hot Download, as alterações de lógica e configurações devem ser realizadas *off-line*, ou seja, com a monitoração desabilitada, sendo então transferidas ao equipamento em uma única etapa. Isto exige que toda e qualquer modificação realizada necessite ser enviada ao equipamento através de um comando realizado pelo próprio usuário, evitando assim que edições incompletas provoquem um funcionamento diferente do esperado.



ATENÇÃO!

Tenha muito cuidado ao realizar o Hot Download: erros podem resultar em ferimentos as pessoas envolvidas e danificar o equipamento. Antes de fazer o Hot Download, avalie as consequências das alterações e as notifique aos envolvidos.

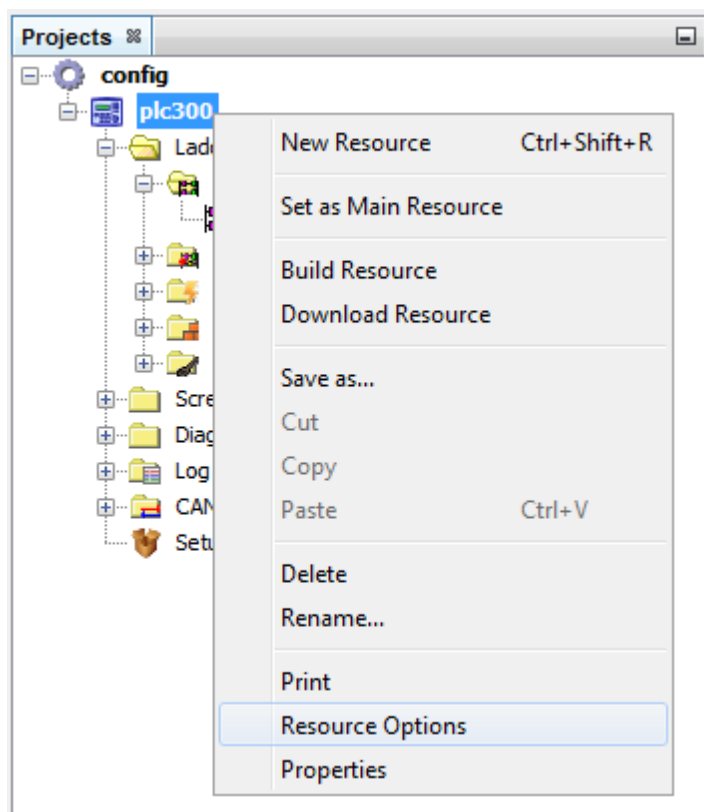
Compatibilidade

Equipamento	Versão
PLC300	2.30 ou superior

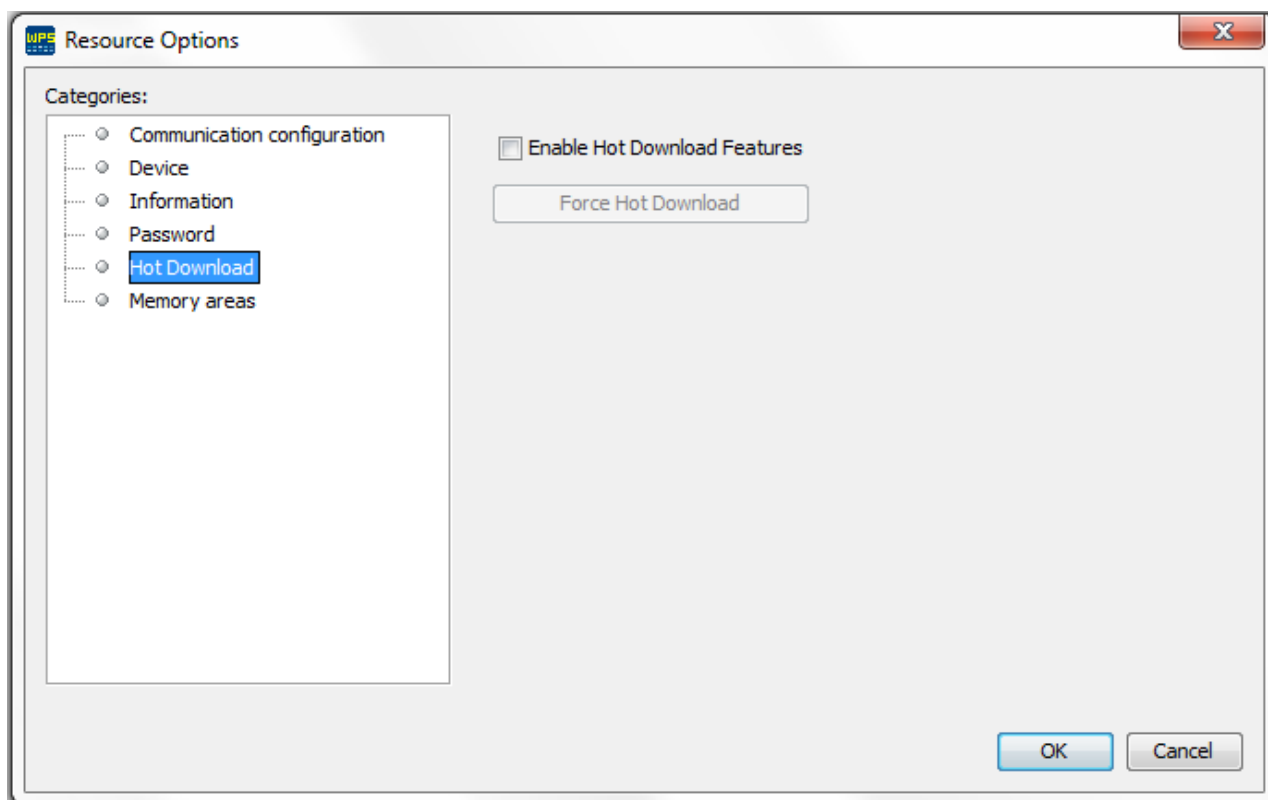
10.1.9.4.2 Habilitar/Desabilitar Hot Download

A opção de Hot Download vem desabilitada no recurso por padrão, para habilitá-la basta seguir os passos abaixo:

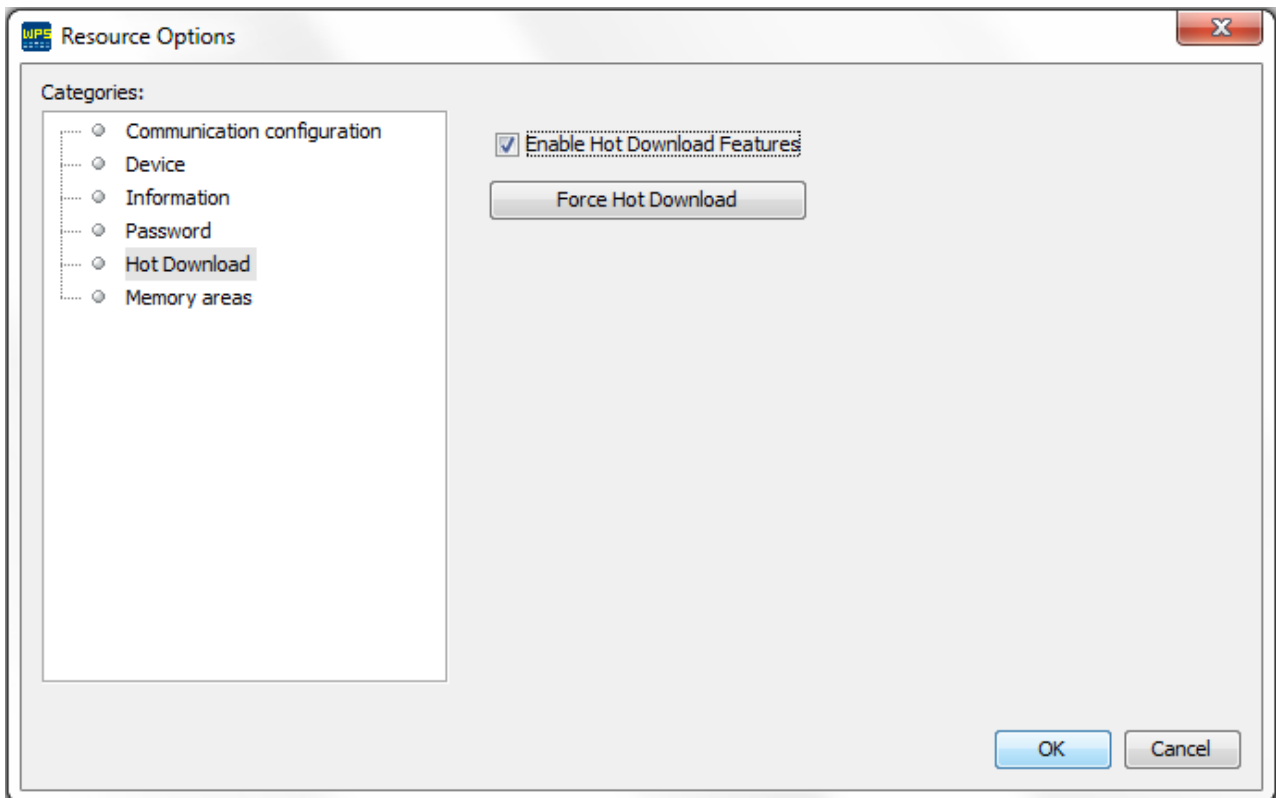
- 1) Abra o menu de contexto do recurso e seleciona a opção "Opções do Recurso".



2) Selecione a opção "Hot Download" no menu lateral, irá abrir uma janela com as configurações de Hot Download.



3) Na janela de configuração do Hot Download basta seleccionar a opção "Habilitar Recursos de Hot Download" e confirmar.



4) A opção "Forçar Hot Download" é utilizada quando se deseja que o sistema reabilite o Hot Download para o recurso, após o usuário realizar uma operação que o desabilite.
Exemplo: O usuário cria uma variável local após o sistema ter informado que a criação da mesma iria desabilitar o Hot Download, caso o usuário exclua esta variável ele pode então utilizar a opção de "Forçar Hot Download" para que o sistema continue permitindo o Hot Download da mesma forma que permitia antes da criação da variável.

10.1.9.4.3 Restrições

Para que seja possível a realização do Hot Download no equipamento, é necessário que, após o *download* completo do recurso, algumas restrições não sejam violadas.

Restrições são limitações impostas às operações que o usuário executa em um recurso, pois tais operações inviabilizam a rotina de Hot Download.

A lista abaixo exhibe as restrições para o usuário:

- **Áreas de memória**
 - o Não podem ser alteradas quaisquer das áreas de memória.
- **Programas**
 - o Não podem ser removidos;
 - o Não podem ser adicionados na lista de execução das tarefas.

- **Tarefas**
 - o Não podem ser editadas;
 - o Não podem ser removidas;
 - o Não podem ter suas listas de execução modificadas. Também não pode ser alterada a ordem de execução dos programas.
- **Variáveis**
 - o Variáveis de escopo LOCAL não poderão ser criadas, excluídas ou editadas;
 - o Variáveis de escopo GLOBAL sem endereço e do grupo GLOBAL_RETAIN sem endereço não poderão ser criadas, excluídas ou editadas.
 - o Variáveis de escopo GLOBAL com endereço e CONSTANT poderão ser criadas, excluídas ou editadas.
 - o Variáveis do USERFB do tipo VAR_IN, VAR_OUT, VAR_IN_OUT, LOCAL, LOCAL_RETAIN não poderão ser criadas, excluídas ou editadas.
- **USERFBs**
 - o Não podem ser adicionados nem removidos dos programas.
- **Estruturas**
 - o Não podem ser editadas ou removidas.
- **Receitas**
 - o Não podem ser transferidas ao equipamento.
- **Telas**
 - o Sem restrições.
 - o Durante o carregamento das modificações no dispositivo será mostrada uma tela indicando a operação. Caso ocorra alteração nas telas, será chamada a tela HOME.
- **Alarme**
 - o Sem restrições.
 - o Caso seja removido algum alarme armazenado no histórico de alarmes, o histórico se torna inválido.
- **Log de evento**
 - o Sem restrições.

Caso alguma das restrições seja violada, uma mensagem de alerta será mostrada.

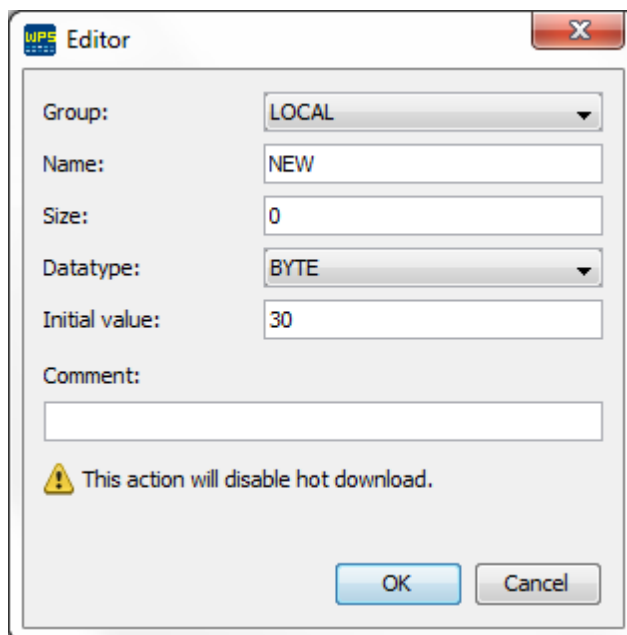


Figura 1 - Alerta na janela de edição

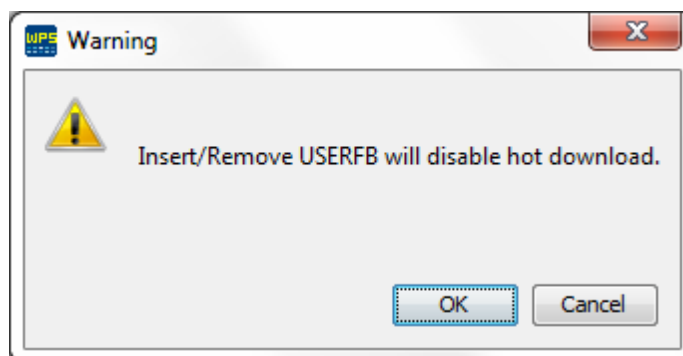


Figura 2 - Alerta na caixa de diálogo



NOTA!

Os alertas exibidos durante a edição não são garantia de que o Hot Download estará desabilitado quando o usuário requisitá-lo. Estes alertas servem apenas como parâmetro para que o usuário tenha a consciência de que a alteração que ele está realizando irá desabilitar o Hot Download caso ela não venha a ser revertida.

10.1.9.4.4 Funcionamento

Após a realização de um *download* completo de um recurso, o Hot Download é habilitado para os próximos *downloads*.

Caso seja desejável realizar o Hot Download, é importante verificar na janela de saída da compilação se existe espaço disponível para o *download* do arquivo Ladder e do mapa de variáveis.

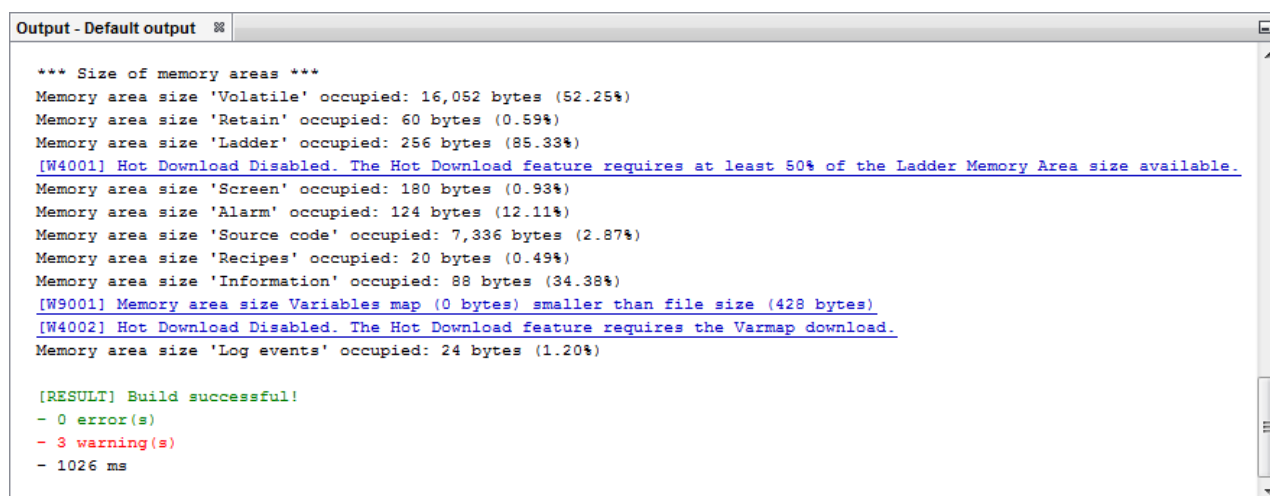


Figura 1 - Avisos de memória insuficiente para HotDownload exibidos na janela de compilação

Após realizar as alterações desejadas no recurso, o início da [rotina de download](#) verificará se o recurso editado é igual ao recurso em execução no equipamento.

Caso sejam iguais, é apresentada a janela para seleção entre **Download Completo** ou **Hot Download**.

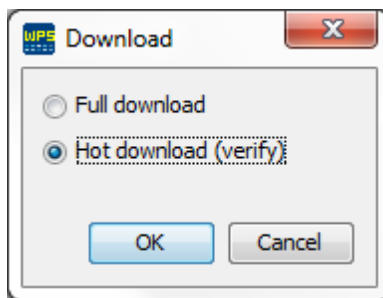


Figura 2 - Diálogo de escolha entre Download Completo e HotDownload

Caso a opção de Hot Download seja selecionada, é realizada uma verificação para saber se alguma das [restrições](#) foi violada. Caso tenha sido, uma mensagem de erro é exibida, informando o impedimento. Validada a verificação, é feita a comparação das áreas de memória, sendo que os arquivos que foram alterados são apresentados em vermelho.



NOTA!

O mapa de variáveis sempre é apresentado na cor cinza, visto que seu *download* é obrigatório no caso de alterações.

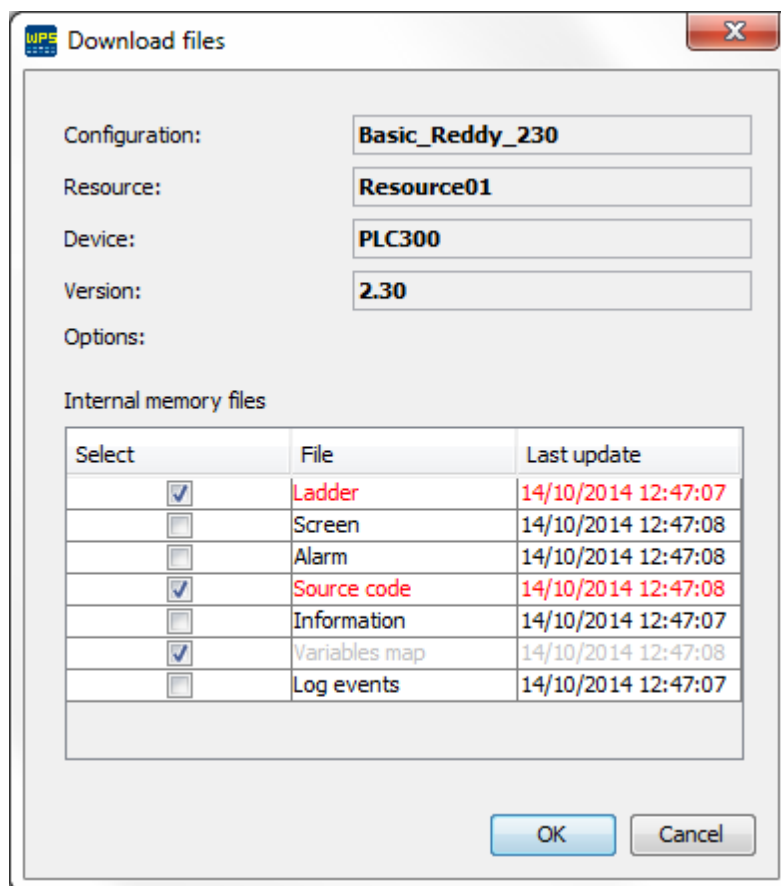


Figura 3 - Janela de escolha de arquivos para Hot Download

Caso algum arquivo seja enviado ao equipamento sem o envio do código fonte, este se tornará inválido, impossibilitando assim o seu *upload*.

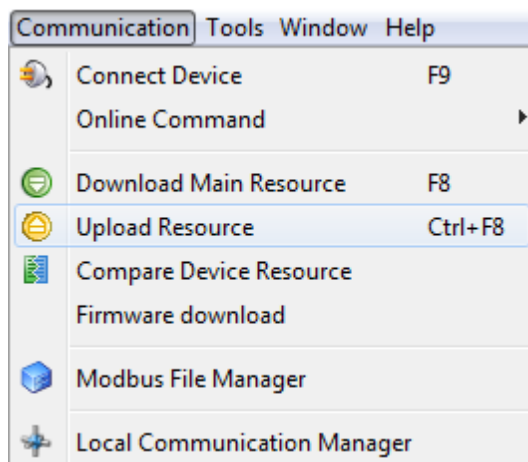
Este procedimento é realizado porque o código fonte não mais representa os arquivos presentes nas áreas de memória.

**NOTA!**

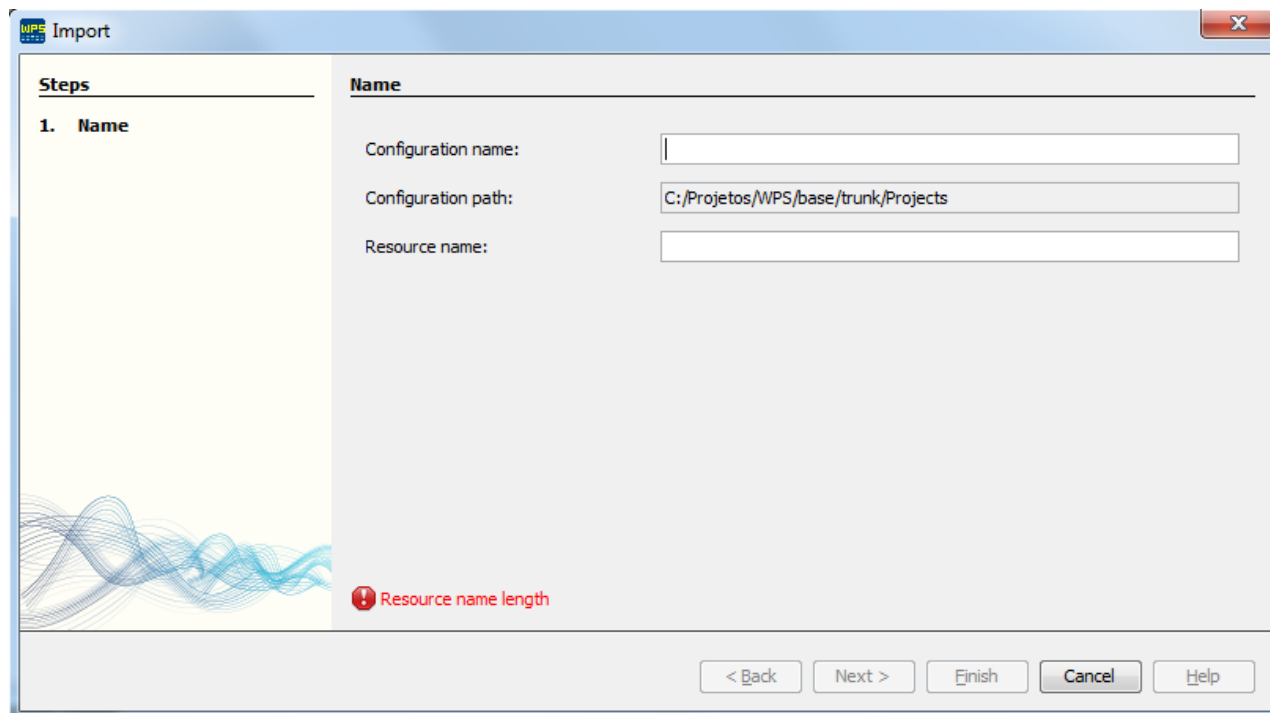
O código fonte se torna inválido mesmo que ele seja igual ao previamente salvo no equipamento.

10.1.9.5 Upload

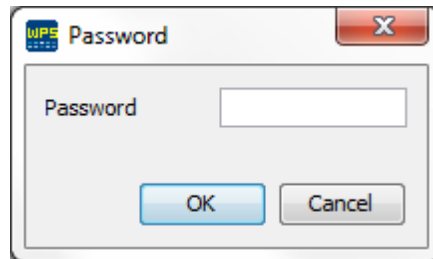
Realiza o *upload* do código fonte do recurso. Para isso, é necessário que o código fonte tenha sido gravado anteriormente (durante o *download*) na memória do equipamento. Para iniciar o *upload*, é necessário acessar o menu Comunicação > Upload do recurso.



Caso o *upload* tenha sido realizado com sucesso, a janela para seleção do nome da configuração será aberta.



Caso alguma senha para proteção do código fonte tenha sido configurada, ela será solicitada:



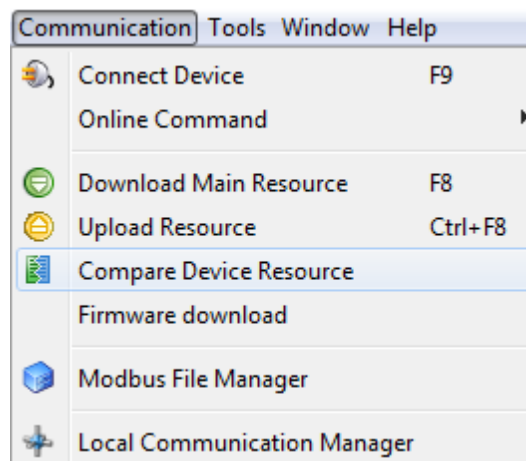
10.1.9.6 Comparação do recurso com o dispositivo



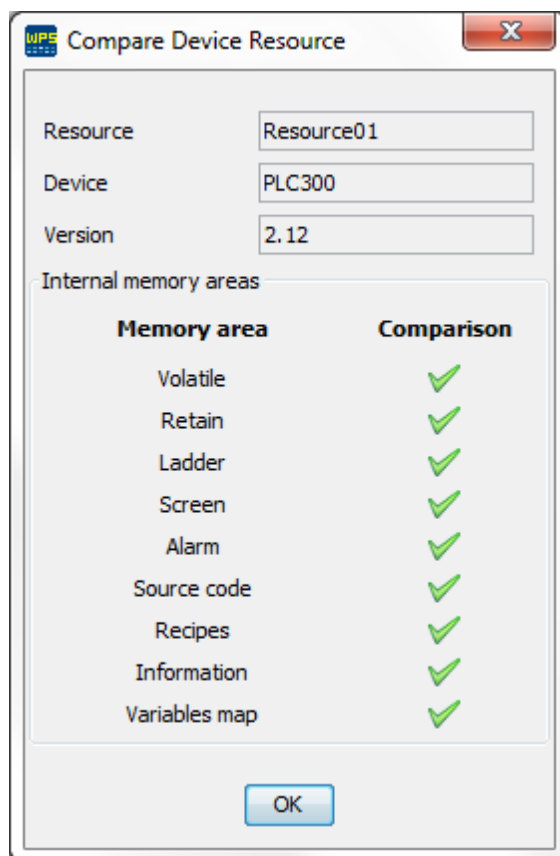
NOTA!

A comparação do recurso com o equipamento funciona somente para versões superiores à versão 1.50 de *firmware*.

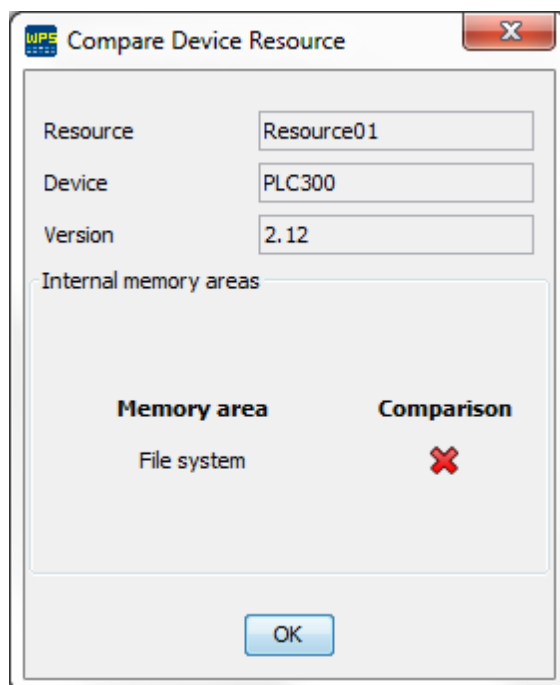
A comparação do recurso do equipamento (arquivos binários) permite ao usuário comparar o recurso principal da aplicação com o recurso que está sendo executado no equipamento. Para iniciar a comparação, é necessário acessar o menu **Comunicação > Compara recurso do equipamento**.



Após selecionada a opção de comparação, é verificado se o recurso está compilado para que as informações de tamanho e CRC dos arquivos binários sejam lidas. Então, uma comparação de cada uma das áreas de memória é realizada:



Caso as áreas de memória tenham sido alteradas, um erro de comparação do sistema de arquivos é mostrado:

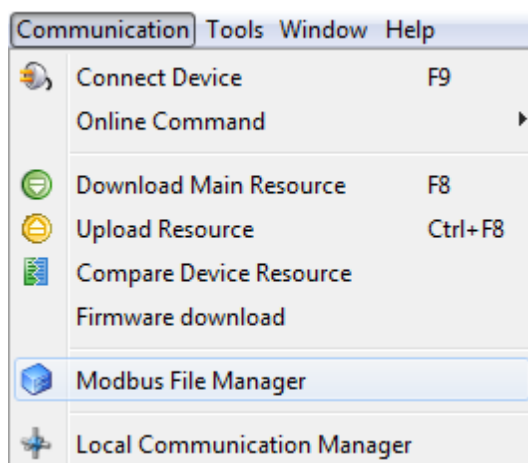


Observações:

- Os arquivos de informação e código fonte são alterados a cada compilação.
- Os arquivos de voláteis, retentivos, código fonte e receitas são optativos no *download*. Caso o *download* destes arquivos não tenha sido realizado, a comparação aparecerá sinalizada como falsa.

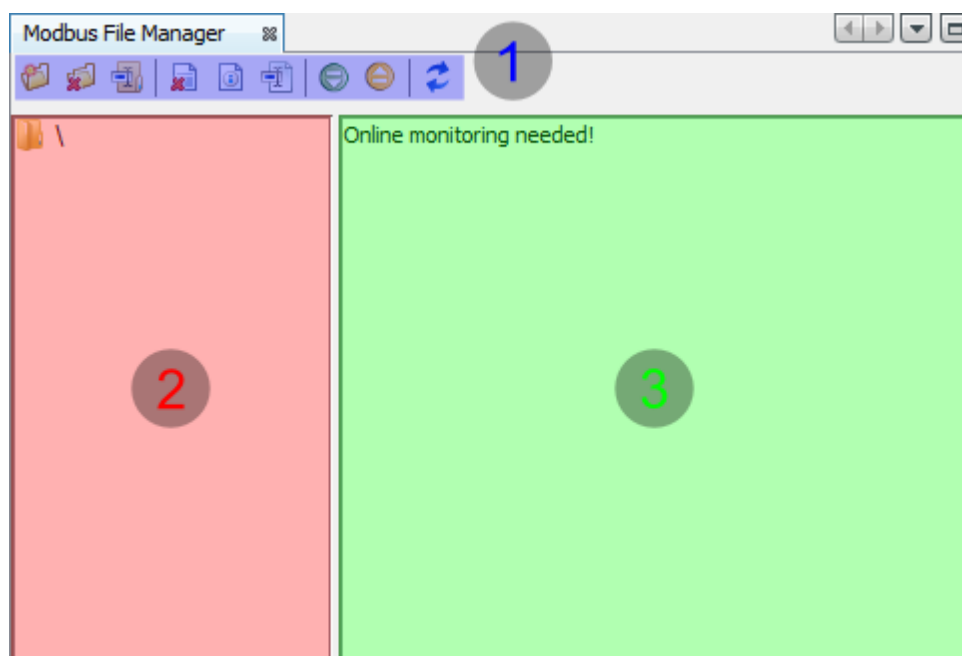
10.1.9.7 Gerenciador de Arquivos Modbus

O Gerenciador de Arquivos Modbus do PLC300 é acessado por meio do menu **Comunicação**, conforme figura a seguir.



A janela do Gerenciador de Arquivos Modbus é composta das seguintes partes:

1. Barra de ferramentas do gerenciador.
2. Lista de diretórios do cartão SD.
3. Lista de arquivos do diretório selecionado.



Só é possível visualizar diretórios e arquivos do cartão SD quando o WPS está conectado ao PLC300 através do comando **Conectar Equipamento** .

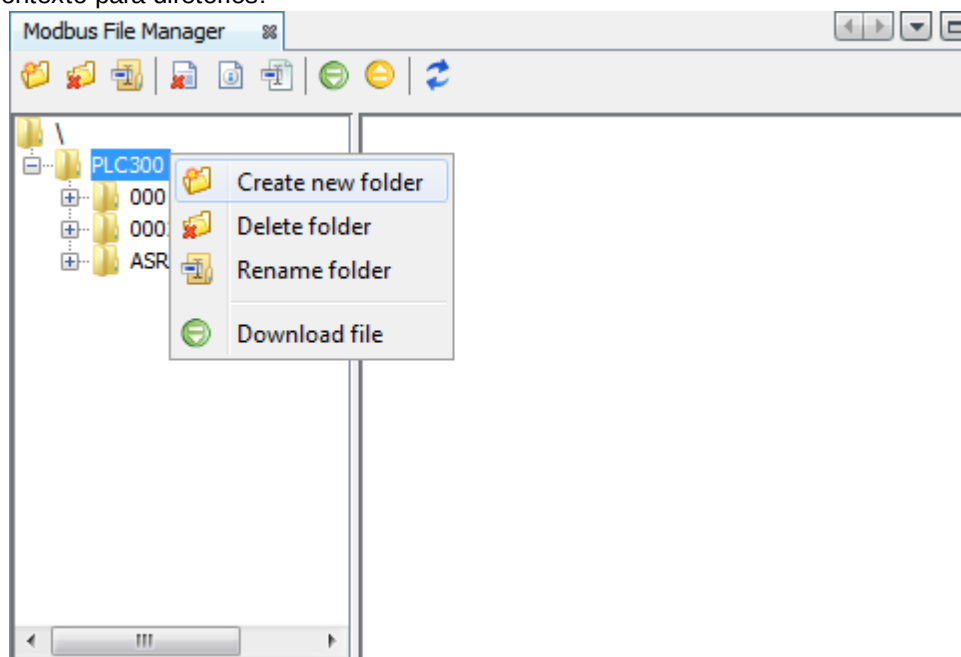
Barra de ferramentas

-  Cria um novo diretório
-  Apaga o diretório selecionado
-  Renomeia o diretório selecionado
-  Apaga o arquivo selecionado
-  Exibe as propriedades do arquivo selecionado (caminho, tamanho e data)
-  Renomeia o arquivo selecionado
-  Envia o arquivo selecionado para o PLC300
-  Recebe o arquivo selecionado do PLC300

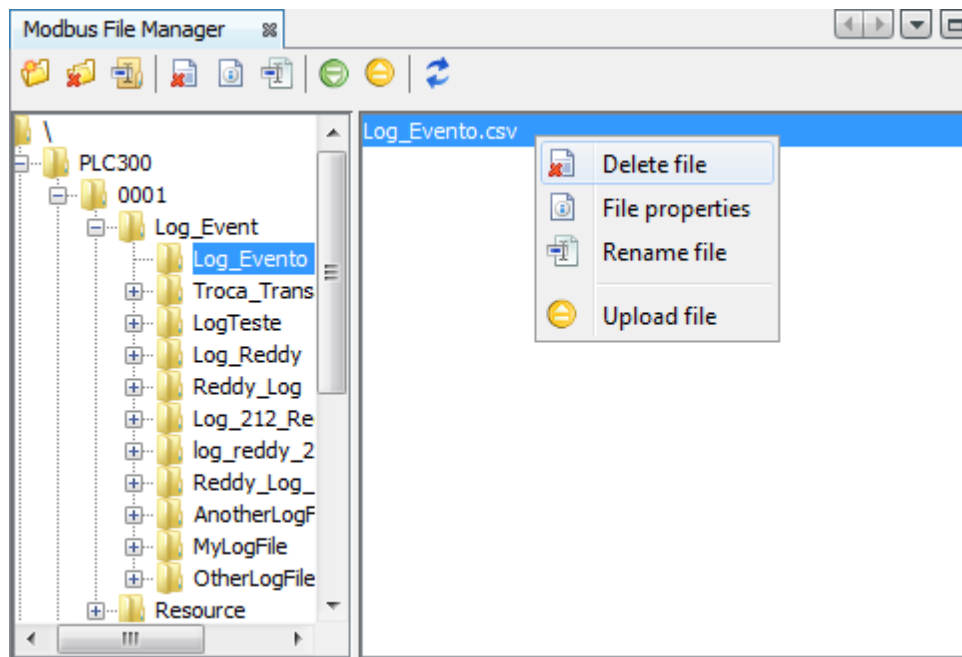
Menu de contexto

Para acessar o menu de contexto, basta clicar com o botão direito em um diretório ou em um arquivo e ele será exibido, conforme figuras a seguir.

- Menu de contexto para diretórios:



- Menu de contexto para arquivos:



10.1.9.8 Comunicação RS232

Visão Geral

Em uma interface serial, os bits de dados são enviados sequencialmente através de um canal de comunicação ou barramento. Diversas tecnologias utilizam comunicação serial para transferência de dados, incluindo as interfaces RS232.

Configuração

Alguns parâmetros precisam ser configurados para realizar a comunicação RS232, para isso, verifique o tópico [RS232](#).

Existem 3 modos de operação da comunicação RS232.

- **Modo 0:** Escravo Modbus;
- **Modo 2:** Protocolo ASCII;
- **Modo 4:** Telegramas genéricos.



NOTA!

Os modos 1 e 3 são reservados.

Modo 0: Escravo Modbus

Verificar manual **PLC300 - Comunicação Modbus RTU**, disponível em <http://www.weg.net/br>.

Modo 2: Protocolo ASCII

O protocolo ASCII, via RS232, foi desenvolvido para leitura de códigos de barras.

Variáveis envolvidas:

- RS232_MODE: marcador de sistema tipo BYTE que define o modo de funcionamento da RS232:
 - o 2: protocolo ASCII;
- RS232_RX_CLEAR: marcador tipo BIT; limpa o buffer de dados (RS232_ASCII_BYTEBUFFER) e o flag RS232_RX_FINISHED;
- RS232_RX_FINISHED: marcador tipo BIT; indica que um pacote de dados está disponível no buffer RS232_ASCII_BYTEBUFFER;
- RS232_ASCII_BYTEBUFFER: buffer de 256 bytes que armazena os caracteres recebidos;
- RS232_ASCII_STRING: buffer tipo STRING que pode ser mostrado na tela do PLC300 pelo componente 'Text Output'.

Como utilizar a RS232 no modo ASCII:

1. Selecionar o protocolo ASCII fazendo: RS232_MODE = 2, no Ladder;
2. Dar um pulso no marcador RS232_RX_CLEAR, para limpar o buffer RS232_ASCII_BYTEBUFFER, e o flag RS232_RX_FINISHED;
3. Ao receber um pacote, o flag: RS232_RX_FINISHED vai para TRUE;
4. Os dados estão disponíveis para o Ladder através do array: RS232_ASCII_BYTEBUFFER, de 256 bytes;
5. Para utilizar com a função 'Text Output', da IHM, utilizar o marcador tipo STRING, RS232_ASCII_STRING.

**NOTA!**

Mesmo sem ligar RS232_RX_CLEAR, o sistema pode receber outra leitura, colocando os dados por cima da leitura anterior.

**NOTA!**

A finalização do pacote é feita quando recebe os caracteres CR/LF (0x0d/0x0a).

**NOTA!**

Os caracteres CR/LF não são armazenados no buffer.

**NOTA!**

O buffer é terminado com caractere NULL (0x00).

A configuração da RS232 deve ser feita pela configuração normal do PLC300;

Nas telas, foi acrescentado o componente 'Text Output' que aceita somente o tipo STRING, para mostrar RS232_ASCII_STRING, que é a string que mostra o valor lido pelo protocolo, limitado a 20 caracteres (limitação da função output = 1 linha).

Modo 4: Telegramas genéricos

Funcionalidade desenvolvida para envio e recepção de telegramas através da comunicação serial RS232.

Variáveis de comando envolvidas:

- RS232_MODE: Marcador do tipo BYTE; define o modo de funcionamento da RS232:
 - o 4: telegramas via RS232.
- RS232_TIMEOUT: marcador tipo WORD; indica o tempo máximo (em ms) de espera de uma resposta, impedindo que novo telegrama seja enviado antes que esse tempo transcorra.
- RS232_END_CHARACTER: define um caractere que pode terminar um telegrama, por exemplo, ETX (03H). Ao receber esse caractere, o PLC300 considera que os bytes recebidos até ele são os necessários, os demais serão ignorados.
- RS232_ENABLE_END_CHARACTER: marcador tipo BIT; marcador de sistema que habilita a utilização do caractere de finalização de telegrama especificado em RS232_END_CHARACTER;
 - o 0: desabilitado;
 - o 1: habilitado.
- RS232_START_TX: marcador tipo BIT; o telegrama é enviado na borda de subida desse marcador;
- RS232_TX_ADDRESS: marcador tipo WORD; indica o endereço inicial dos dados a serem transferidos;
- RS232_TX_LENGTH: marcador do tipo BYTE; indica o número de bytes da informação a ser enviada;
- RS232_RX_ADDRESS: marcador tipo WORD; indica o endereço inicial dos dados recebidos;
- RS232_MAX_RX_BUFFER_LENGTH: marcador do tipo BYTE; indica o número máximo de bytes que pode ser recebido;

Variáveis de status envolvidas:

- RS232_TX_TELEGRAM_COUNTER: marcador tipo WORD; conta o número de telegramas enviados;
- RS232_RX_TELEGRAM_FINISHED: marcador tipo BIT; indica que um pacote de dados está disponível na memória a partir do endereço indicado em RS232_RX_ADDRESS.
- RS232_TX_FINISHED: marcador tipo BIT; indica que um pacote de dados foi enviado completamente.
- RS232_TIMEOUT_INDICATOR: marcador tipo BIT; indica se transcorreu o tempo especificado em RS232_TIMEOUT.
 - o 0: não ocorreu estouro;
 - o 1: ocorreu estouro de tempo.
- RS232_RX_TELEGRAM_COUNTER (status): marcador tipo WORD; conta o número de telegramas recebidos;
- RS232_RX_BYTE_COUNTER (status): marcador tipo WORD; conta o número de bytes que foram recebidos.

Como utilizar a RS232 para envio e recepção de telegramas:

1. Selecionar “envio e recepção de telegramas via RS232” fazendo RS232_MODE = 4 no Ladder;
2. Configurar o endereço inicial dos telegramas a serem enviados e recebidos em, respectivamente, RS232_TX_ADDRESS e RS232_RX_ADDRESS;
3. Especificar o tamanho (em bytes) do telegrama a ser enviado através do marcador RS232_TX_LENGTH;
4. Especificar o tamanho máximo (em bytes) do telegrama que pode ser recebido, através do marcador RS232_RX_LENGTH;
5. Especificar o timeout em RS232_TIMEOUT;
6. Se for o caso, configurar os caracteres de finalização de telegramas através, respectivamente, dos marcadores: RS232_ENABLE_END_CHARACTER e RS232_END_CHARACTER;
7. Resetar o flag RS232_RX_TELEGRAM_FINISHED para ficar pronto para receber um telegrama;
8. Dar um pulso em RS232_START_TX para enviar um telegrama.

**NOTA!**

Será considerado fim de recepção de telegrama, o recebimento do caractere especial, se configurado em RS232_END_CHARACTER, com o flag RS232_ENABLE_END_CHARACTER habilitado, ou um timeout igual a duas vezes o tempo de transmissão de um byte.

Compatibilidade

Equipamento	Versão
PLC300	Modo 0: 1.00 ou superior
	Modo 2: 1.11 ou superior
	Modo 4: 1.50 ou superior

10.1.9.9 Comunicação RS485

Visão Geral

Em uma interface serial, os bits de dados são enviados sequencialmente através de um canal de comunicação ou barramento. Diversas tecnologias utilizam comunicação serial para transferência de dados, incluindo as interfaces RS485.

Configuração

Alguns parâmetros precisam ser configurados para realizar a comunicação RS485, para isso, verifique o item [RS485](#).

Existem 3 modos de operação da comunicação RS485.

- **Modo 0:** Escravo Modbus;
- **Modo 1:** Mestre Modbus;
- **Modo 4:** Telegramas genéricos.

**NOTA!**

Os modos 2 e 3 são reservados.

Modo 0: Escravo Modbus

Verificar manual **PLC300 - Comunicação Modbus RTU**, disponível em <http://www.weg.net/br>.

Modo 1: Mestre Modbus

Verificar manual **PLC300 - Comunicação Modbus RTU**, disponível em <http://www.weg.net/br>.

Modo 4: Telegramas genéricos

Funcionalidade desenvolvida para envio e recepção de telegramas através da comunicação RS485.

Variáveis envolvidas:

- RS485_MODE: Marcador do tipo BYTE; define o modo de funcionamento da RS485:
 - o 4: Telegramas via RS485.
- RS485_TIMEOUT: marcador tipo WORD; indica o tempo máximo (em ms) de espera de uma resposta.
- RS485_TIMEOUT_INDICATOR: marcador tipo BIT; indica se houve estouro de tempo.
 - o 0: não ocorreu estouro;
 - o 1: ocorreu estouro de tempo.
- RS485_ENABLE_END_CHARACTER: marcador tipo BIT; caractere de fim de transmissão;
 - o 0: desabilitado;

o 1: habilitado.

- RS485_END_CHARACTER: define o caractere que pode terminar um bloco, por exemplo, ETX (03H). RS485_ENABLE_END_CHARACTER deve estar habilitado. Nesse caso, tanto timeout (RS485_TIMEOUT) quanto o caractere de final terminam a recepção do telegrama, o que ocorrer primeiro.
- RS485_START_TX: marcador tipo BIT; deve ser colocado em TRUE para enviar o telegrama;
- RS485_TX_ADDRESS: marcador tipo WORD; indica o endereço inicial dos dados a serem transferidos;
- RS485_TX_LENGTH: marcador do tipo BYTE; indica o número de bytes da informação a ser enviada;
- RS485_TX_TELEGRAM_COUNTER: marcador tipo WORD; conta o número de telegramas enviados;
- RS485_RX_ADDRESS: marcador tipo WORD; indica o endereço inicial dos dados recebidos;
- RS232_MAX_RX_BUFFER_LENGTH: marcador do tipo BYTE; indica o número máximo de bytes que pode ser recebido;
- RS485_RX_TELEGRAM_FINISHED: marcador tipo BIT; indica que um pacote de dados está disponível na memória, a partir do endereço indicado em RS485_RX_ADDRESS.
- RS485_TX_FINISHED (status): marcador tipo BIT; indica que um pacote de dados foi enviado completamente.
- RS485_RX_TELEGRAM_COUNTER: marcador tipo WORD; conta o número de telegramas recebidos;
- RS485_RX_BYTE_COUNTER: marcador tipo WORD; conta o número de bytes que foram recebidos.

Como utilizar a RS485 para envio e recepção de telegramas:

1. Selecionar “envio e recepção de telegramas via RS485” fazendo RS485_MODE = 4 no Ladder;
2. Configurar o endereço inicial dos telegramas a serem enviados e recebidos em, respectivamente, RS485_TX_ADDRESS e RS485_RX_ADDRESS;
3. Especificar o tamanho (em bytes) do telegrama a ser enviado, através do marcador RS485_TX_LENGTH;
4. Especificar o tamanho máximo (em bytes) do telegrama que pode ser recebido através do marcador RS485_RX_LENGTH;
5. Especificar o timeout em RS485_TIMEOUT;
6. Se for o caso, configurar os caracteres de finalização de telegramas através, respectivamente, dos marcadores: RS485_ENABLE_END_CHARACTER e RS485_END_CHARACTER;
7. Resetar o flag RS485_RX_TELEGRAM_FINISHED para ficar pronto para receber um telegrama;
8. Dar um pulso em RS485_START_TX para enviar um telegrama.



NOTA!

Será considerado fim de recepção de telegrama, o recebimento do caractere especial, se configurado em RS485_END_CHARACTER, com o flag RS485_ENABLE_END_CHARACTER habilitado, ou um timeout igual a duas vezes o tempo de transmissão de um byte.

Compatibilidade

Equipamento	Versão
PLC300	Modo 0: 1.00 ou superior
	Modo 2: 1.11 ou superior
	Modo 4: 1.50 ou superior

10.2 SCA06

[Descrição](#)

[Marcadores de Sistema](#)

[Start-up Orientado](#)

[Auto-Tuning](#)

[Importar do WLP](#)

[Parâmetros](#)

[Ladder](#)

[Dianóstico](#)

[Comunicação](#)

10.2.1 Descrição

O servoconversor SCA06 é um produto de alta performance que permite o controle de velocidade, torque e posição de servomotores de corrente alternada senoidal trifásicos. A característica central deste produto é o alto desempenho e alta precisão de controle do movimento do eixo do servomotor devido à operação em malha fechada através da realimentação de posição dada por um sensor dentro do servomotor.

O SCA06 possui alimentações de controle e potência independentes, permitindo, por exemplo, que as redes de comunicação do produto continuem funcionando normalmente mesmo que circuito de potência tenha que ser desligado por algum motivo.

O uso de resistores de frenagem possibilita tempos de frenagem muito reduzidos otimizando processos que exigem alta performance.

Várias funções especiais estão disponíveis, tais como a programação em linguagem Ladder com blocos de posicionamento que proporciona extrema flexibilidade e integração ao acionamento.

As mais variadas aplicações podem ser atendidas com a ampla quantidade de cabos disponível, seja para aplicações simples ou aplicações complexas como movimentação, ambientes com óleo, etc.

10.2.2 Marcadores de Sistema

As variáveis abaixo, contidas no grupo **GLOBAL_SYSTEM** da tabela de variáveis, possuem o tag fixo. O tag dos marcadores de sistema foram divididos em grupos e subgrupos sendo:

Grupos:

- SCA: variáveis de leitura e escrita do servoconversor SCA06;
- CO: variáveis de leitura e escrita da rede de comunicação CANopen.

Subgrupos:

- STS: variável de leitura (status);
- CMD: variável de escrita (comando).

Marcadores de Sistema de Leitura (Status)

Endereço	Bit	Modbus	Tag	Descrição
Ladder				
%SB6000	0	0	FREQ_2HZ	Oscilador com frequência de 2 Hz
%SB6000	1	1	PULSE_1SCAN	Pulso durante o primeiro ciclo de scan
%SB6000	2	2	FALSE	Sempre em 0
%SB6000	3	3	TRUE	Sempre em 1
%SW6002	--	3001	ELAPSED_SCAN_CYCLES	Ciclos de scan transcorridos
Eixo Real				
%SW6004	--	3002	SCA_STS_REAL_AXIS_STATUS	Estado do eixo real (ver nota)
%SD6008	--	3004	SCA_STS_REAL_AXIS_VELOCITY	Velocidade do eixo real
%SL6024	--	3012	SCA_STS_REAL_AXIS_POSITION	Posição do eixo real
Eixo Virtual				
%SW6006	--	3003	SCA_STS_VIRTUAL_AXIS_STATUS	Estado do eixo virtual (ver nota)
%SD6012	--	3006	SCA_STS_VIRTUAL_AXIS_VELOCITY	Velocidade do eixo virtual
%SL6032	--	3016	SCA_STS_VIRTUAL_AXIS_POSITION	Posição do eixo virtual
Corrente				
%SD6016	--	3008	SCA_STS_MOTOR_CURRENT	Corrente no motor
Posição na transição de Dí's				
%SD6040	--	3020	SCA_STS_DI1_POSITION_STORED	Posição armazenada na transição da DI1
%SD6048	--	3024	SCA_STS_DI2_POSITION_STORED	Posição armazenada na transição da DI2
%SD6056	--	3028	SCA_STS_DI3_POSITION_STORED	Posição armazenada na transição da DI3
Contadores				
%SD6064	--	3032	SCA_STS_BUILT_IN_COUNTER	Valor do contador padrão
%SD6068	--	3034	SCA_STS_BUILT_IN_COUNTER_DI3	Valor do contador padrão armazenado na transição da DI3
%SD6072	--	3036	SCA_STS_ENC1_COUNTER	Valor do contador do encoder 1
%SD6076	--	3038	SCA_STS_ENC2_COUNTER	Valor do contador do encoder 2
%SD6080	--	3040	SCA_STS_ENC_COUNTER_Z1	Contador de encoder armazenado na transição do Z1 conforme definido em P00511
%SD6084	--	3042	SCA_STS_ENC_COUNTER_Z2	Contador de encoder armazenado na transição do Z2 conforme definido em P00521
CANopen				
%SB6100	0	800	CO_STS_MASTER_CONTACTED	Mestre CANopen contactou todos os escravos
%SB6100	1	801	CO_STS_MASTER_CONFIG_OK	Mestre CANopen fez download das configurações dos escravos
%SB6100	2	802	CO_STS_MASTER_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/heartbeat) iniciado com os escravos

%SB6100	3	803	CO_STS_MASTER_INIT_FINISHED	Inicialização de todos os escravos realizada pelo mestre CANopen
%SB6100	4	804	CO_STS_MASTER_INIT_ERROR	Algum escravo apresentou erro de inicialização
%SB6100	5	805	CO_STS_MASTER_ERROR_CTRL	Detectado falha em algum escravo no protocolo de detecção de erros pelo mestre CANopen
%SB6100	6	806	CO_STS_MASTER_EMCY	Algum escravo reportou EMCY
%SB6101	0	808	CO_STS_MASTER_NMT_TOGGLE	Feedback do toggle bit do comando NMT
%SB6101	5	813	CO_STS_MASTER_BUS_OFF	Mestre CANopen está em bus off
%SB6101	6	814	CO_STS_MASTER_POWER_OFF	Mestre CANopen não possui alimentação na interface CAN
%SB6101	7	815	CO_STS_MASTER_COMM_DISABLED	Comunicação do mestre CANopen está desabilitada
%SB6102	0	816	CO_STS_SLAVE1_CONTACTED	Escravo no endereço indicado foi contactado com sucesso pelo mestre CANopen
%SB6102	1	817	CO_STS_SLAVE1_CONFIG_OK	Configuração do escravo foi feita com sucesso pelo mestre CANopen
%SB6102	2	818	CO_STS_SLAVE1_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/heartbeat) iniciado com o escravo
%SB6102	3	819	CO_STS_SLAVE1_INIT_FINISHED	Inicialização do escravo concluída
%SB6102	4	820	CO_STS_SLAVE1_INIT_ERROR	Erro na inicialização do escravo no endereço indicado
%SB6102	5	821	CO_STS_SLAVE1_ERROR_CTRL_FAIL	Detectado falha em algum escravo no protocolo de detecção de erro pelo mestre CANopen
%SB6102	6	822	CO_STS_SLAVE1_EMCY	Escravo no endereço indicado reportou erro por EMCY
%SB6104	0	832	CO_STS_SLAVE2_CONTACTED	Escravo no endereço indicado foi contactado com sucesso pelo mestre CANopen
%SB6104	1	833	CO_STS_SLAVE2_CONFIG_OK	Configuração do escravo foi feita com sucesso pelo mestre CANopen
%SB6104	2	834	CO_STS_SLAVE2_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/heartbeat) iniciado com o escravo
%SB6104	3	835	CO_STS_SLAVE2_INIT_FINISHED	Inicialização do escravo concluída
%SB6104	4	836	CO_STS_SLAVE2_INIT_ERROR	Erro na inicialização do escravo no endereço indicado
%SB6104	5	837	CO_STS_SLAVE2_ERROR_CTRL_FAIL	Detectado falha em algum escravo no protocolo de detecção de erro pelo mestre CANopen
%SB6104	6	838	CO_STS_SLAVE2_EMCY	Escravo no endereço indicado reportou erro por EMCY
...	...	...	...	...
%SB6354	0	2832	CO_STS_SLAVE127_CONTACTED	Escravo no endereço indicado foi contactado com sucesso pelo mestre CANopen
%SB6354	1	2833	CO_STS_SLAVE127_CONFIG_OK	Configuração do escravo foi feita com sucesso pelo mestre CANopen
%SB6354	2	2834	CO_STS_SLAVE127_ERROR_CTRL_OK	Protocolo de controle de erros (node guarding/heartbeat) iniciado com o escravo
%SB6354	3	2835	CO_STS_SLAVE127_INIT_FINISHED	Inicialização do escravo concluída
%SB6354	4	2836	CO_STS_SLAVE127_INIT_ERROR	Erro na inicialização do escravo no endereço indicado
%SB6354	5	2837	CO_STS_SLAVE127_ERROR_CTRL_FAIL	Detectado falha em algum escravo no protocolo de detecção de erro pelo mestre CANopen

%SB6354	6	2838	CO_STS_SLAVE127_EMCI	Escravo no endereço indicado reportou erro por EMCY
%SW6360	--	3180	CO_SDO_ERROR_NODE_ID	Erro de SDO: endereço do escravo do último erro de SDO detectado
%SW6362	--	3181	CO_SDO_ERROR_OBJECT_INDEX	Erro de SDO: índice do objeto
%SW6364	--	3182	CO_SDO_ERROR_OBJECT_SUBINDEX	Erro de SDO: sub-índice do objeto
%SW6366	--	3183	CO_SDO_ERROR_FUNCTION	Erro de SDO: função (leitura/escrita)
%SD6368	--	3184	CO_SDO_ERROR_VALUE	Erro de SDO: valor
%SD6372	--	3186	CO_SDO_ERROR_CODE	Erro de SDO: código do erro
%SB6380	--	3190	CO_EMCI_SLAVE_ID	Último EMCY reportado: endereço do escravo
%SB6382	--	3191	CO_EMCI_DATA	Último EMCY reportado: dados do objeto

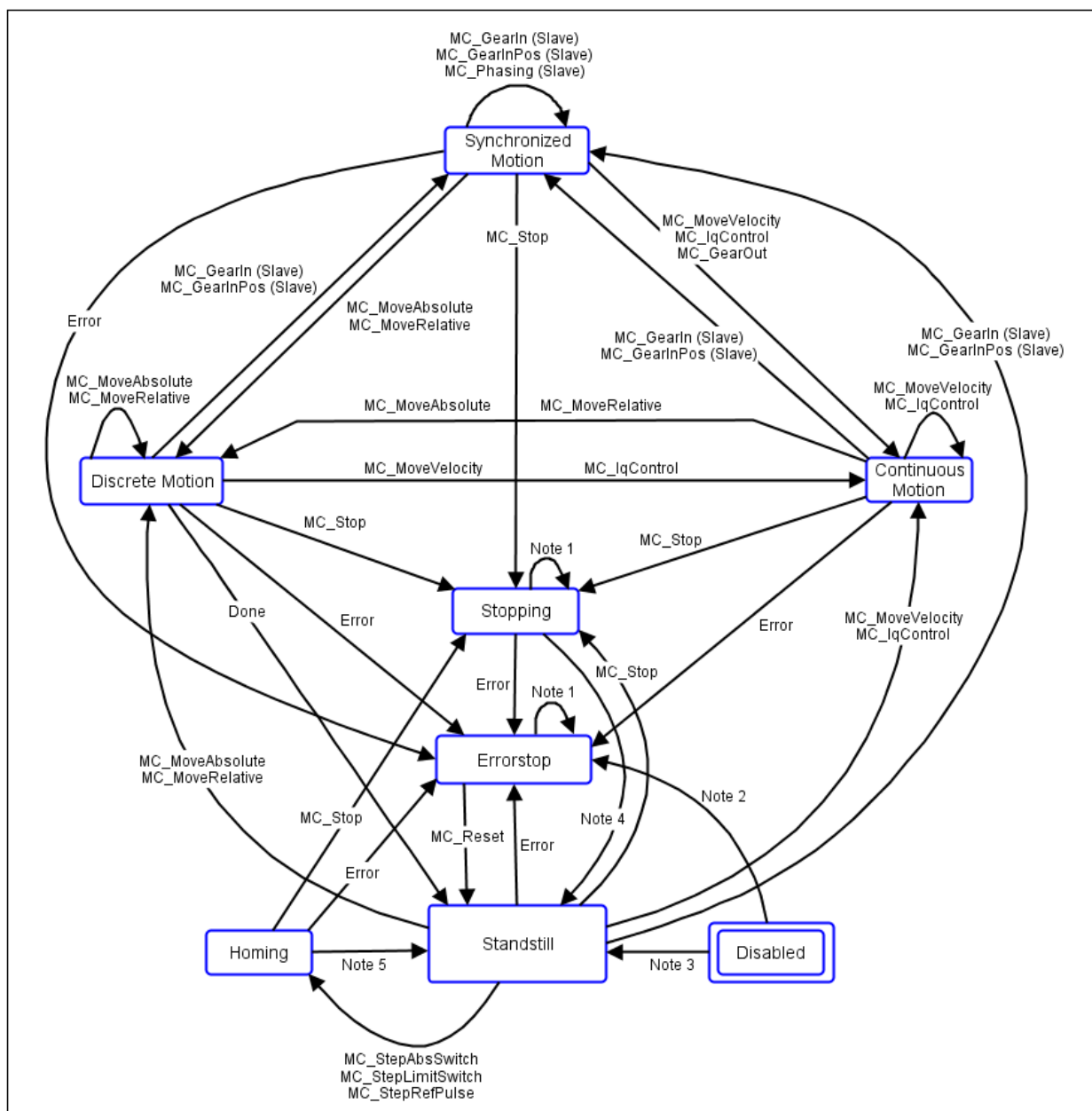
Marcadores de Sistema de Escrita / Leitura (Comando)

Endereço	Bit	Modbus	Tag	Descrição
CANopen				
%CB6000	--	3000	CO_CMD_NMT_COMMAND	Envio de comando NMT pelo mestre CANopen: código do comando
%CB6001	0	8	CO_CMD_NMT_TOGGLE	Envio de comando NMT pelo mestre CANopen: toggle bit
%CB6001	7	15	CO_CMD_DISABLE	Desabilita comunicação CANopen
%CB6002	--	3001	CO_CMD_NMT_SLAVE_ADDR	Envio de comando NMT pelo mestre CANopen: endereço do escravo

NOTA!

Abaixo descrição dos estados do eixo real e virtual:

0. Disabled (Desabilitado).
1. Errorstop (Eixo com falha).
2. Standstill (Habilitado e nenhum bloco de movimento ativo).
3. Stopping (Stop em execução).
4. Homing (Referenciando).
5. Continuous Motion (Movimento contínuo).
6. Discrete Motion (Movimento discreto).
7. Synchronized Motion (Movimento sincronizado).



Note 1: Quando o drive estiver no estado "Stopping" ou "Errorstop" todos os blocos podem ser chamados, mas somente o bloco MC_Reset será executado;

Note 2: Tentativa de habilitar o drive, mas o drive está em falha;

Note 3: Habilitação do drive e o drive não está em falha;

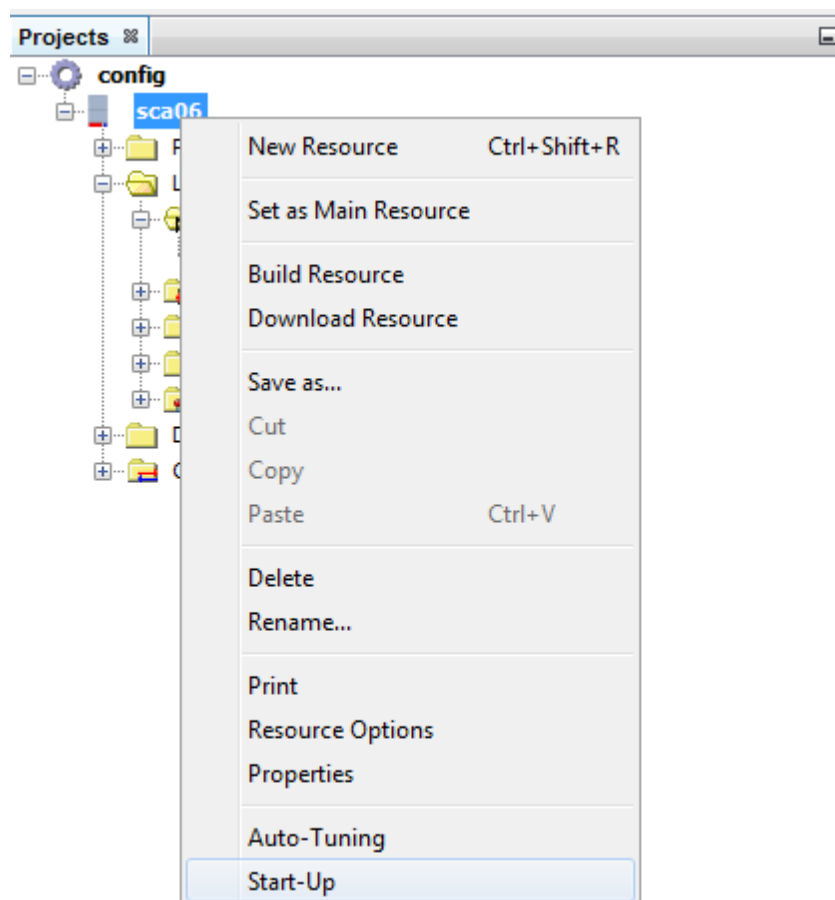
Note 4: MC_Stop.Done é verdadeiro e MC_Stop.Execute é falso;

Note 5: MC_StopDirect, MC_StopRefPulse ou MC_FinishHoming.

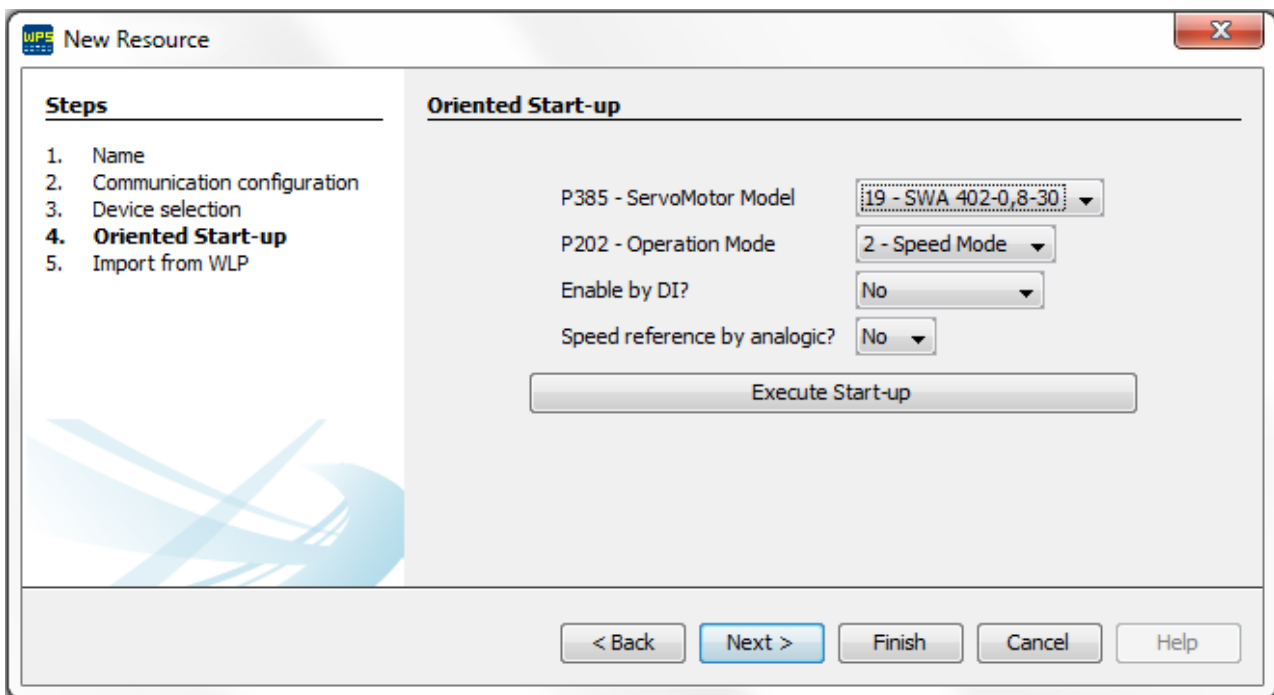
10.2.3 Start-up Orientado

A função de Start-Up Orientado é utilizada para a realização das configurações mínimas necessárias para se colocar o SCA-06 em operação.

O Start-Up Orientado pode ser executado durante a [criação do recurso](#), ou através do menu de contexto do recurso e seleção da opção "Start-Up Orientado".



1. Na tela de Start-Up as principais opções que devem ser configuradas são os parâmetros P385 (Modelo do Servomotor) e P202 (Modo de Operação).

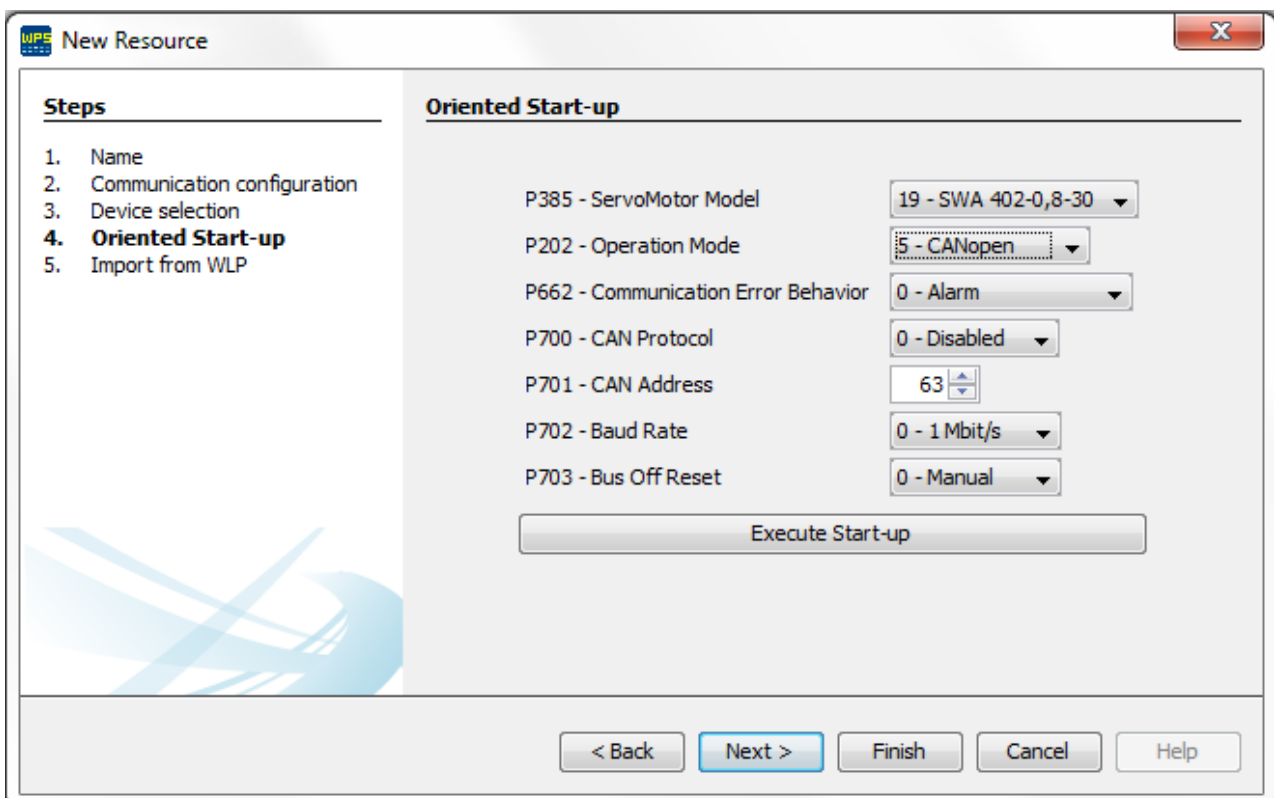


The image shows the 'New Resource' window in the WPS software, specifically the 'Oriented Start-up' step. On the left, a 'Steps' list shows: 1. Name, 2. Communication configuration, 3. Device selection, 4. **Oriented Start-up**, and 5. Import from WLP. The main area contains the following configuration options:

Parameter	Value
P385 - ServoMotor Model	19 - SWA 402-0,8-30
P202 - Operation Mode	2 - Speed Mode
Enable by DI?	No
Speed reference by analogic?	No

Below these options is an 'Execute Start-up' button. At the bottom of the window are navigation buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

2. Conforme o modo de operação selecionado o configurador irá se ajustar habilitando ou desabilitando opções, segue abaixo um exemplo das opções caso o modo de operação selecionado seja "5 - CANopen".



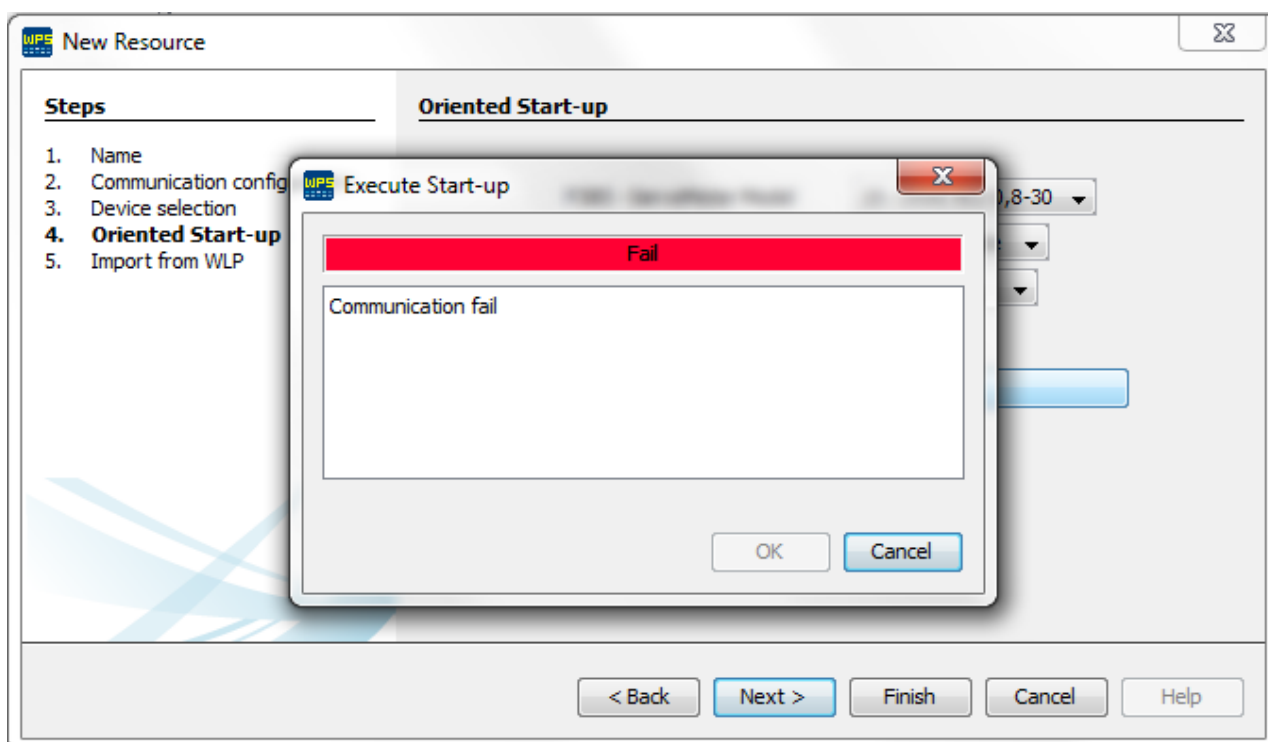
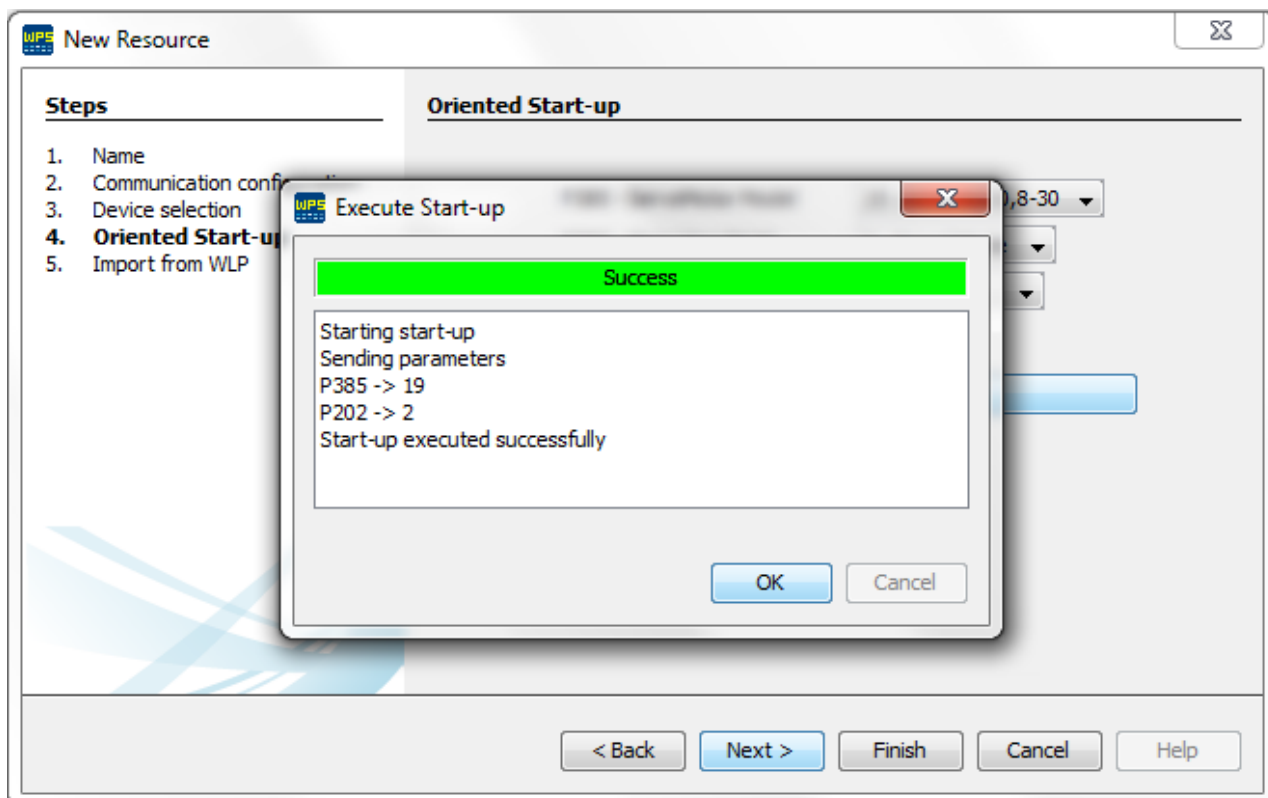
The image shows the 'New Resource' window in the WPS software, specifically the 'Oriented Start-up' step, configured for CANopen mode. The 'Steps' list on the left is identical to the previous window. The main area contains the following configuration options:

Parameter	Value
P385 - ServoMotor Model	19 - SWA 402-0,8-30
P202 - Operation Mode	5 - CANopen
P662 - Communication Error Behavior	0 - Alarm
P700 - CAN Protocol	0 - Disabled
P701 - CAN Address	63
P702 - Baud Rate	0 - 1 Mbit/s
P703 - Bus Off Reset	0 - Manual

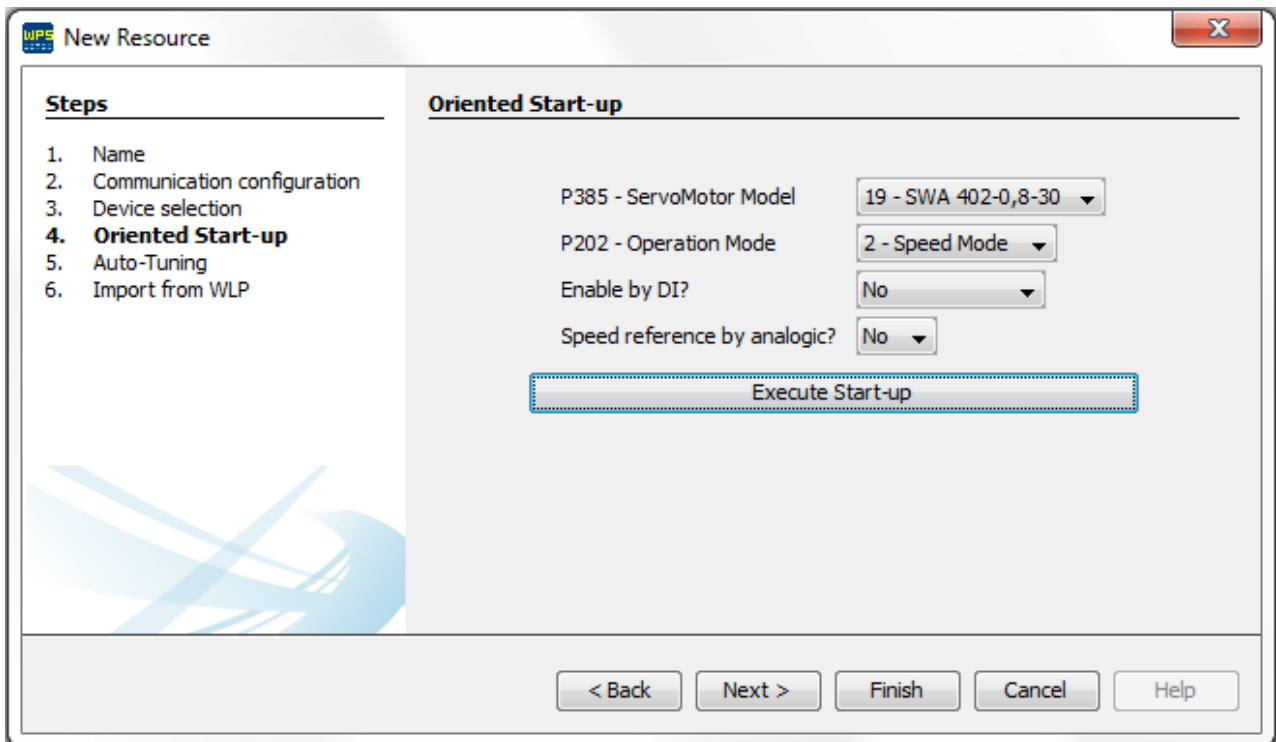
Below these options is an 'Execute Start-up' button. At the bottom of the window are navigation buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

3. Após definido o modo de operação e demais opções, basta clicar em "Executar Start-up", caso este seja

executado corretamente será exibida uma mensagem informando sucesso, caso contrário uma mensagem informando falha.



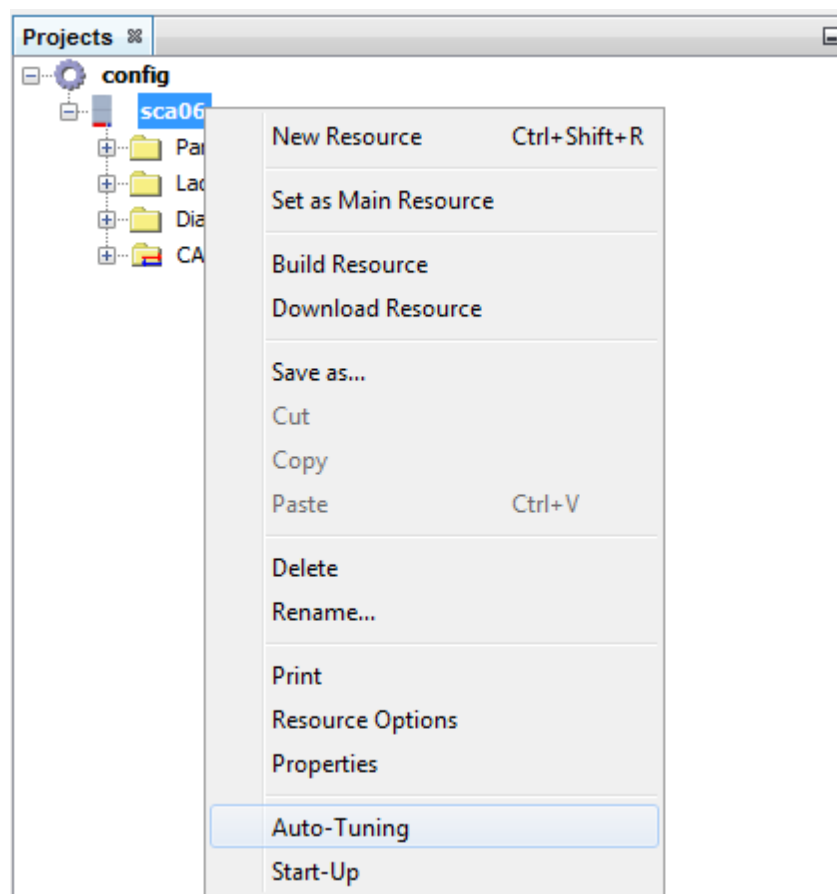
4. Após o Start-Up ser executado com sucesso durante a criação do recurso, o sistema irá habilitar a etapa de [Auto-Tuning](#) no assistente, que fica a critério do usuário se deseja ou não executá-la.



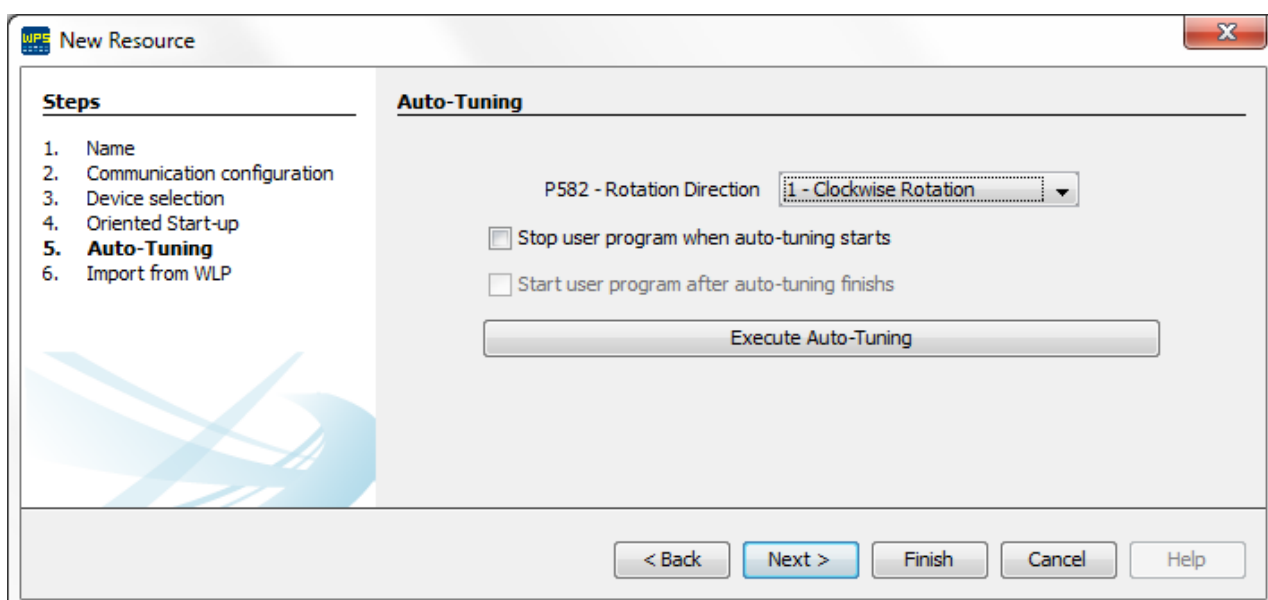
10.2.4 Auto-Tuning

A função de Auto-Tuning é utilizada para a realização de ajustes automáticos no SCA-06 para obtenção de uma melhor performance.

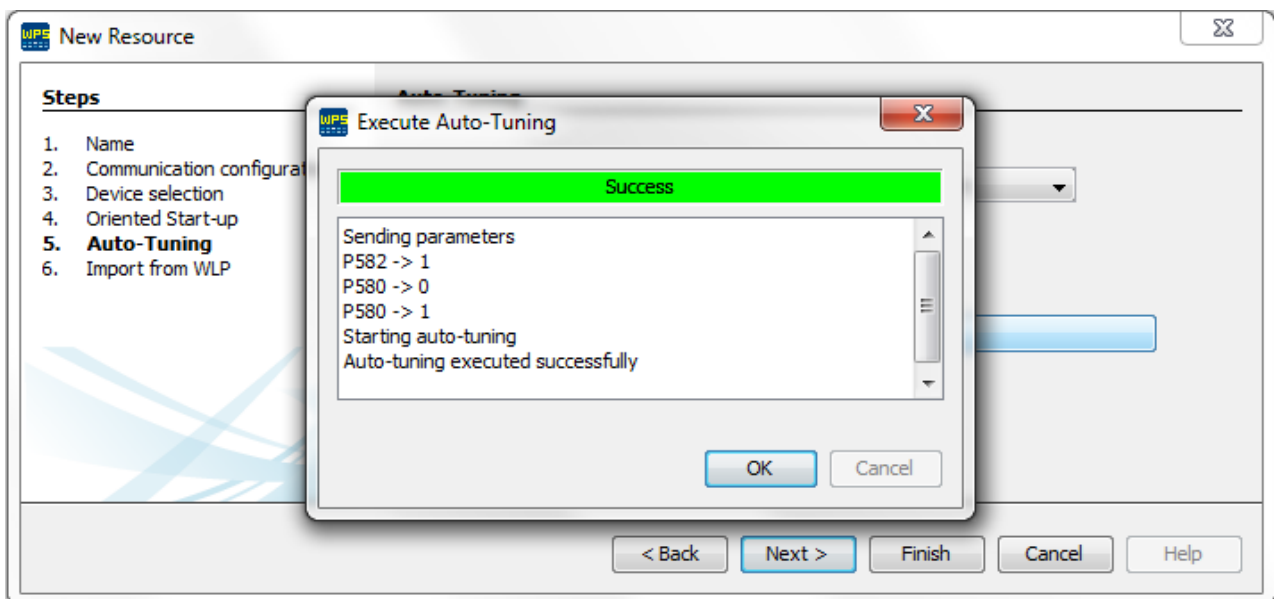
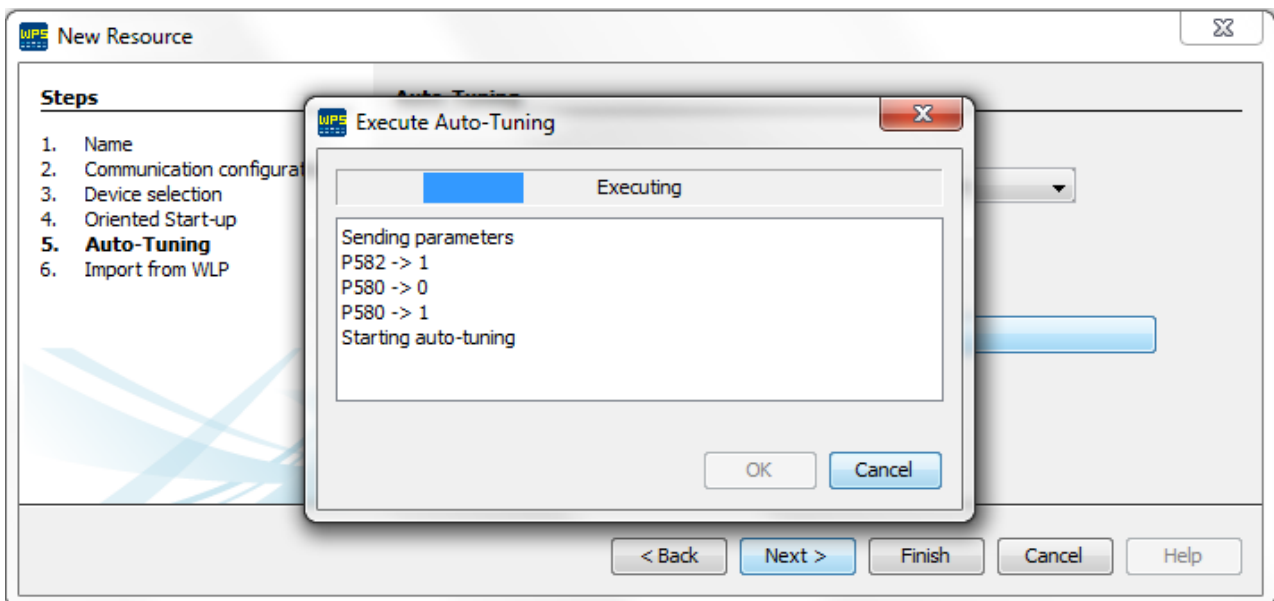
O Auto-Tuning pode ser executado durante a [criação do recurso](#), ou através do menu de contexto do recurso e seleção da opção "Auto-Tuning".



1. Na tela de Auto-Tuning as opções que precisam ser configuradas são P582 (Sentido de giro do motor) e se o programa do usuário deve ser parado antes da execução e iniciado novamente após a conclusão do Auto-Tuning.



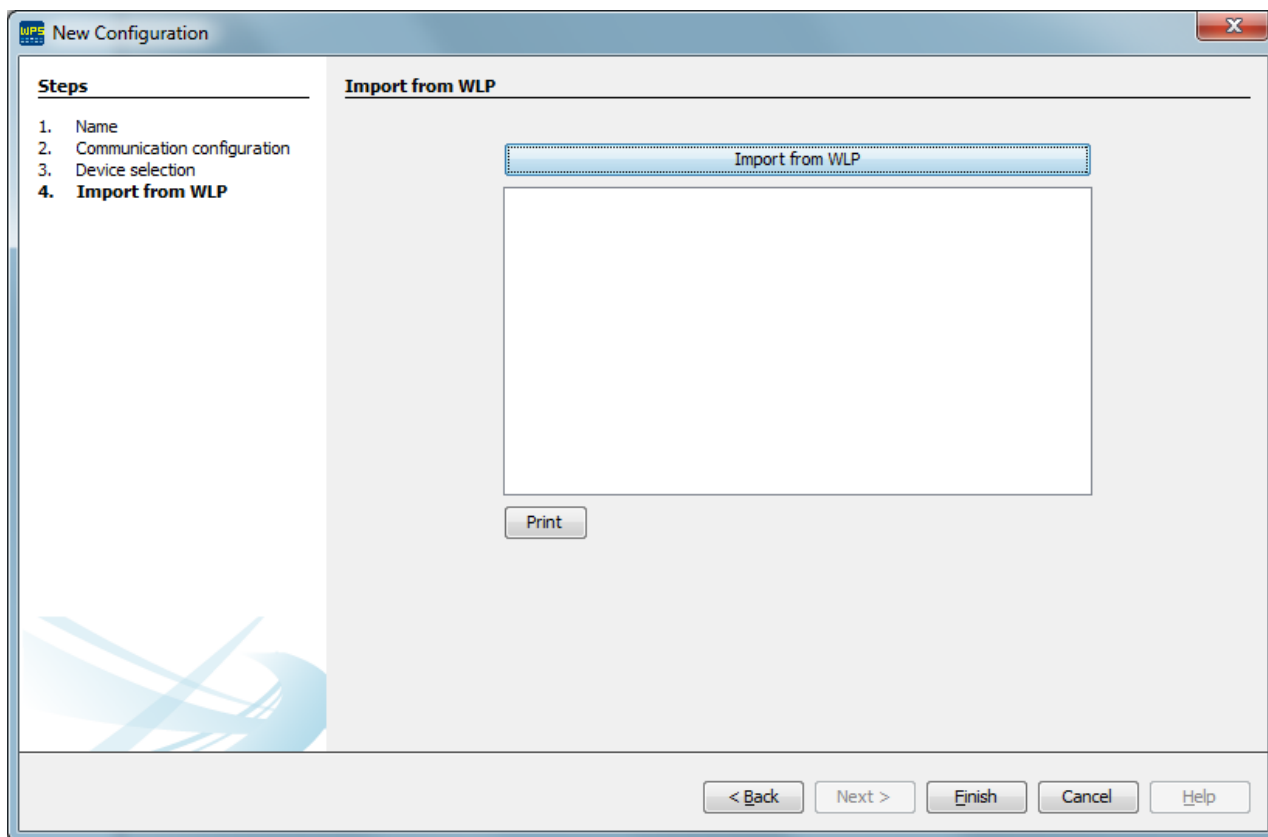
2. Após escolher as opções desejadas basta clicar em "Executar Auto-Tuning" para iniciar o processo que demora menos de 1 minuto, caso este seja executado corretamente será exibida uma mensagem informando sucesso, caso contrário uma mensagem informando falha.



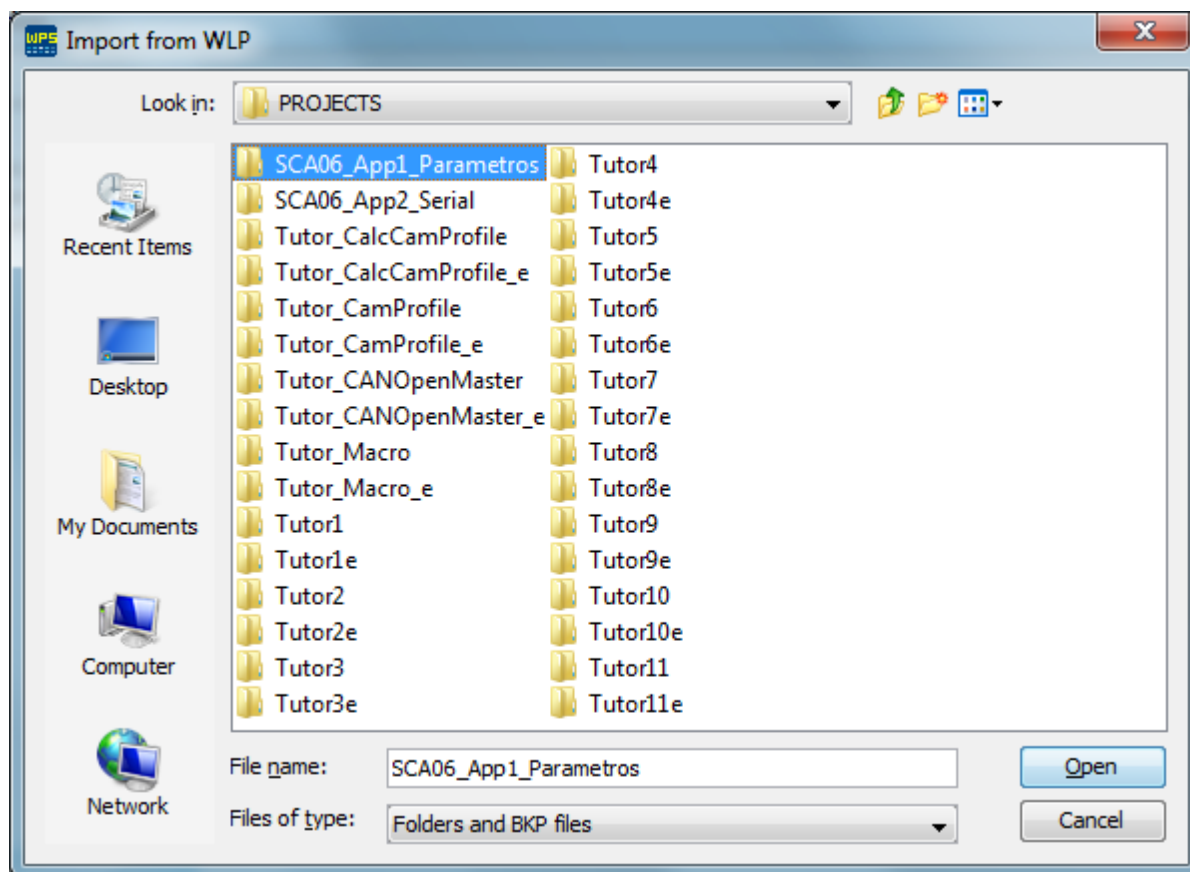
10.2.5 Importar do WLP

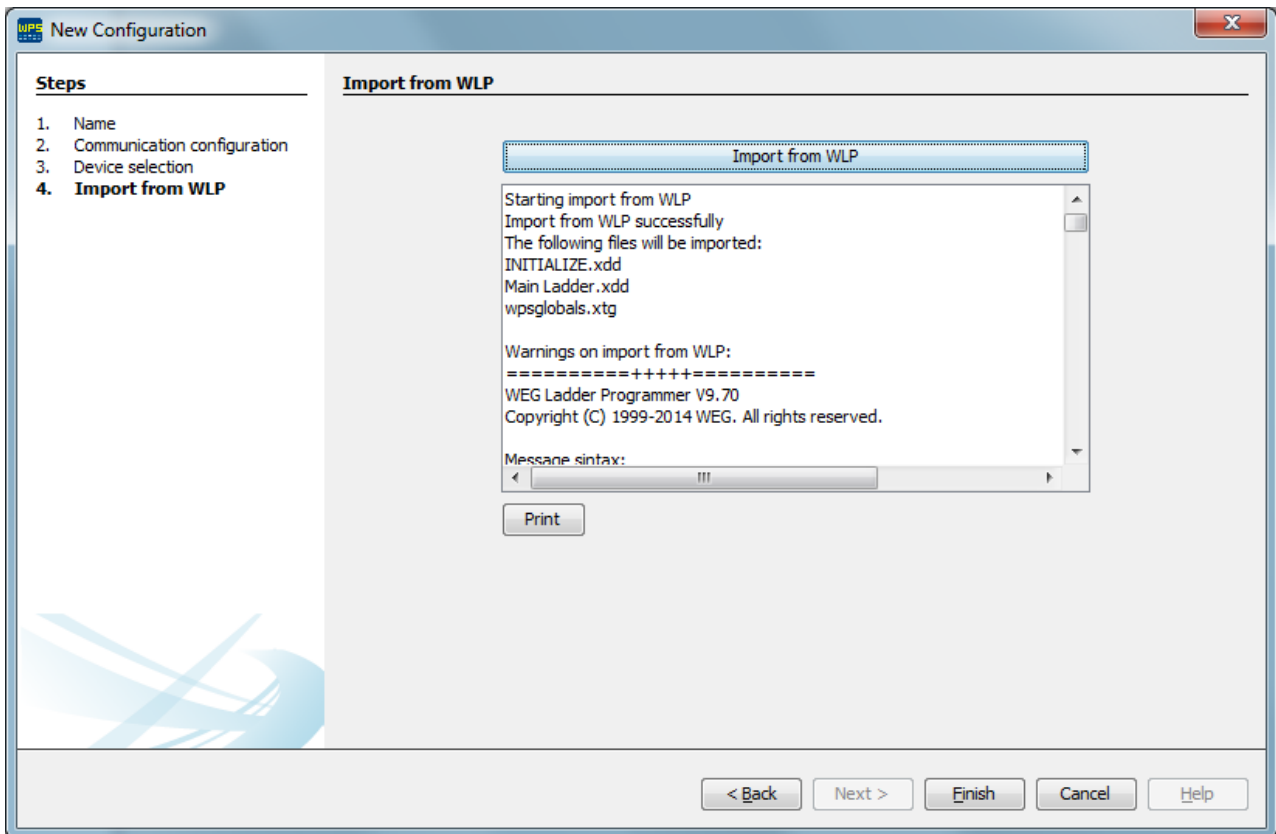
A função importar do WLP é utilizada para importar o Ladder desenvolvido no software WLP para o SCA06.

A importação do WLP deve ser executada durante a [criação do recurso](#).



1. Para executar a importação clique no botão Importar do WLP e selecione a pasta do projeto do WLP ou o arquivo BKP do WLP.





2. Após a importação finalizada com sucesso clique no botão Finalizar para copiar os arquivos importados para o novo recurso.

10.2.6 Parâmetros

[Visão Geral](#)

[Configuração](#)

[Leitura e Escrita de Parâmetros](#)

[Ocultar/Desocultar Parâmetros e Grupos de Parâmetros](#)

[Parâmetros do Usuário](#)

10.2.6.1 Visão Geral

A tela de configuração de parâmetros é utilizada para a configuração e monitoração de todos os parâmetros do equipamento, inclusive os parâmetros de usuário.



NOTA!

A leitura e escrita dos parâmetros é realizada através desta tela, somente a configuração dos parâmetros do usuário deve ser enviada a primeira vez ou sempre que modificada através da rotina de *download* do recurso.

Abaixo segue uma visão geral sobre a tela de configuração de parâmetros.

The screenshot shows the WEG SCA06 parameter configuration interface. It features a 'Projects' tree on the left, a 'Parameters' tree in the center, and a large table of parameters on the right. Red boxes and numbers 1 through 7 highlight specific features:

- 1. Projects tree
- 2. Parameters tree
- 3. Parameter groups
- 4. Action icons
- 5. Parameter table
- 6. Parameter table rows
- 7. Output window

Parameter	Description	User	Monitored	Minimum	Maximum	Factory setti...	Unit
P0099	Enable	0: Disable	0: Disable	0	2	0: Disable	
P0105	STOP Function Ramp	200	200	1	32767	200	ms/rpm
P0111	Rotation Direction	0: standart	0: standart	0	1	0: standart	
P0119	Current Reference	0.0	0.0	-3276.7	3276.7	0.0	A
P0121	Speed Reference	0	0	-9999	9999	0	rpm
P0126	Limit position Enable	0: Disabled	0: Disabled	0	1	0: Disabled	
P0127	Position lower limit - fractions of revolution	-16383	-16383	-16383	16383	-16383	
P0128	Position lower limit - number of revolutions	-32768	-32768	-32768	32767	-32768	
P0129	Position higher limit - fractions of revolution	16383	16383	-16383	16383	16383	
P0130	Position higher limit - number of revolutions	32767	32767	-32768	32767	32767	
P0131	Current lower limit	-140.0	-140.0	-140.0	0.0	-140.0	A
P0132	Current higher limit	140.0	140.0	0.0	140.0	140.0	A
P0133	Speed lower limit	-9999	-9999	-9999	0	-9999	rpm
P0134	Speed higher limit	9999	9999	0	9999	9999	rpm
P0136	Idynamic/Nominal	300	300	0	400	300	%
P0154	Breaking Resistor	0	0	0	1000	0	ohm
P0155	Breaking Resistor Average Power	200	200	0	10000	200	W
P0156	Breaking Resistor Maximum Energy	2200	2200	0	10000	2200	J
P0159	Kp Position Regulator	50	50	0	32767	50	
P0161	Kp PID Speed	4000	4000	0	32767	4000	
P0162	Ki PID Speed	50	50	0	32767	50	
P0163	Kd PID Speed	0	0	0	32767	0	
P0192	RTC Update	0	0	0	1	0	
P0193	Day of week	0	0	0	6	0	
P0194	Day	1	1	1	31	1	
P0195	Month	1	1	1	12	1	
P0196	Year	2011	2011	0	4095	2011	
P0197	Hous	0	0	0	23	0	
P0198	Minutes	0	0	0	59	0	
P0199	Seconds	0	0	0	59	0	
P0200	Password	1: Active	1: Active	0	2	1: Active	
P0202	Operation Mode	1: Torque Mode	2: Speed Mode	1	6	2: Speed Mode	
P0203	Flash backup Enable	1: Enable	1: Enable	0	1	1: Enable	
P0204	Load/Save Parameters	0: Disable	0: Disable	0	13	0: Disable	
P0209	Position/speed feedback source	0: internal motor sensor	0: internal motor sensor	0	3	0: internal mo...	
P0210	External feedback gear ratio:Numerator	1	1	1	32767	1	
P0211	External feedback gear ratio:Denominator	1	1	1	32767	1	
P0213	External feedback Rotation Direction	0: direct	0: direct	0	1	0: direct	
P0214	External feedback loss protection	1000	1000	0	9999	1000	

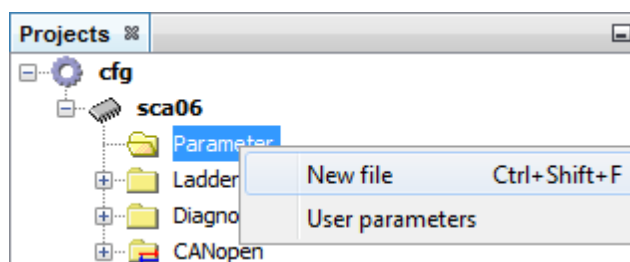
- Arquivos de parâmetros:** Nesta parte ficam todos os arquivos de configurações de parâmetros criados pelo usuário. Observar que quando o arquivo tiver um boneco desenhado sobre a tabela, significa que esta tabela de parâmetros possui parâmetros/grupo de parâmetros ocultos.
- Grupos de parâmetros:** Nesta árvore são mostrados todos os grupos de parâmetros, observar que um mesmo parâmetro pode encontrar-se em mais de um grupo, porém quando seu valor for modificado este será modificado em todos os grupos que pertencer.
- Grupos de parâmetros modificados:** Grupos de parâmetros que contenham o boneco sobre a tabela significam que estes possuem parâmetros ocultos.
- Comandos:** Segue abaixo descrição dos comandos por ordem de exibição:
 - Desocultar parâmetro: Caso algum parâmetro tenha sido oculto este botão permite torná-los visíveis novamente.
 - Ocultar parâmetro: Basta selecionar um ou mais parâmetros na tabela e disparar esta ação para que os parâmetros sejam ocultos.
 - Gravar tabela: Grava os valores dos parâmetros exibidos na tela no equipamento, os valores enviados são os presentes na coluna Usuário. O fluxo é Usuário -> Monitorado (equipamento)
 - Ler tabela: Lê os valores dos parâmetros do equipamento exibidos na coluna Monitorado e grava no arquivo de parâmetros na coluna Usuário. O fluxo é Monitorado (equipamento) -> Usuário
 - Parâmetros do usuário: Abre uma tela para edição dos parâmetros do usuário.
 - Filtro: Abre uma opção de filtro de parâmetros, podendo filtrar por número do parâmetro ou descrição.

5. **Parâmetros do Usuário e Parâmetros Monitorado:** Nestas duas colunas ocorre a visualização off-line e on-line dos parâmetros pode-se assim dizer, a coluna **Usuário** mostra os valores que estão no arquivo localizado no computador e a coluna **Monitorado** mostra os valores que estão efetivamente gravados dentro do equipamento. Sempre que você usar a opção de **Gravar Parâmetro**, os valores enviados serão da coluna **Usuário** para a coluna **Monitorado**, ou seja, do Arquivo para o Equipamento, no caso da leitura o fluxo se inverte sendo da coluna **Monitorado** para **Usuário**, ou seja, do Equipamento para o Arquivo. Caso deseja alterar os valores diretamente no equipamento sem alterar no arquivo, basta clicar sobre a coluna monitorado e alterar os valores, que a alteração ocorrerá de forma on-line.
6. **Parâmetros modificados:** Sempre que um valor de um parâmetro estiver diferente entre as colunas **Usuário** e **Monitorado**, este ficará em vermelho.
7. **Saída:** Nesta tela são exibidas informações de erro, caso estes ocorram durante a escrita e ou leitura dos parâmetros.

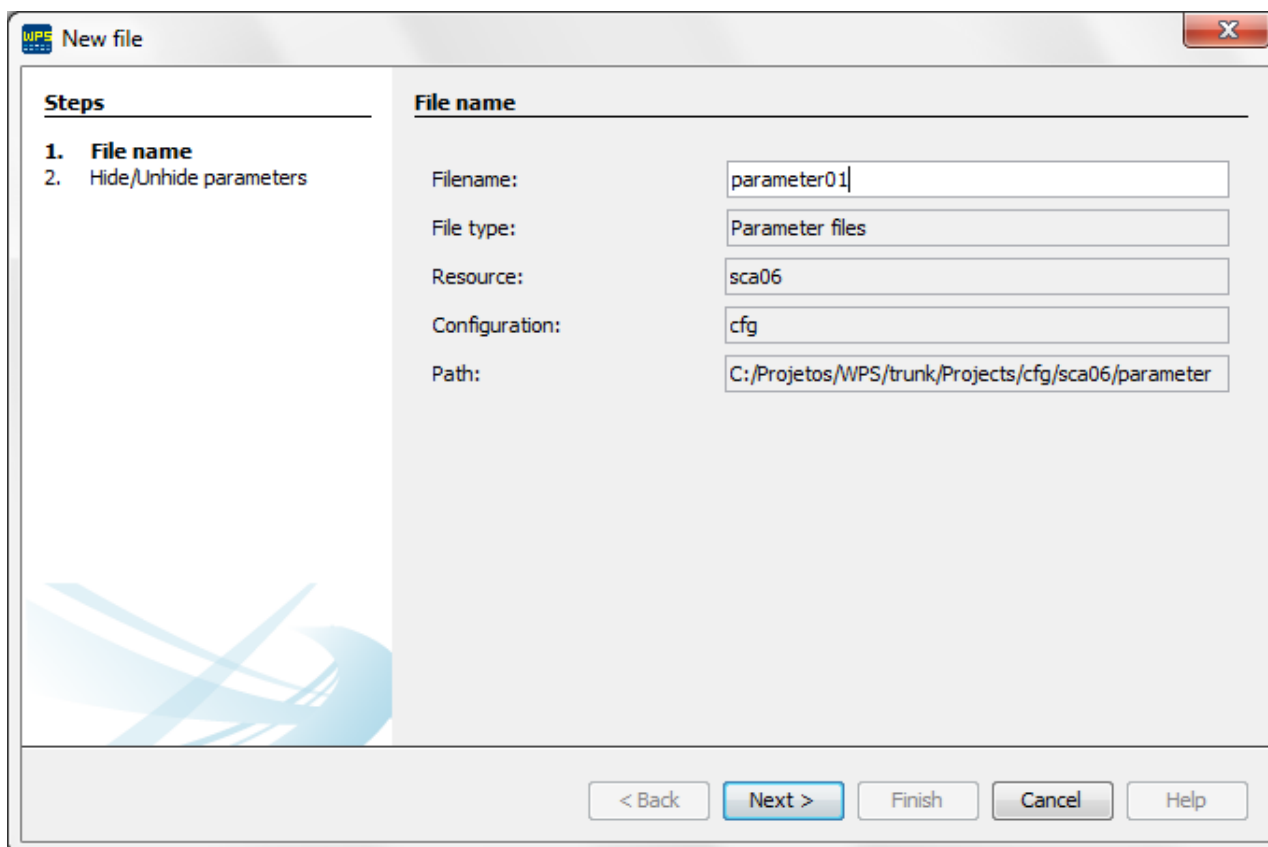
10.2.6.2 Configuração

Abaixo segue a lista dos passos necessários para criação de um arquivo de parâmetros.

1. Criação de um novo arquivo de parâmetros.



2. Definir nome para o arquivo de parâmetros

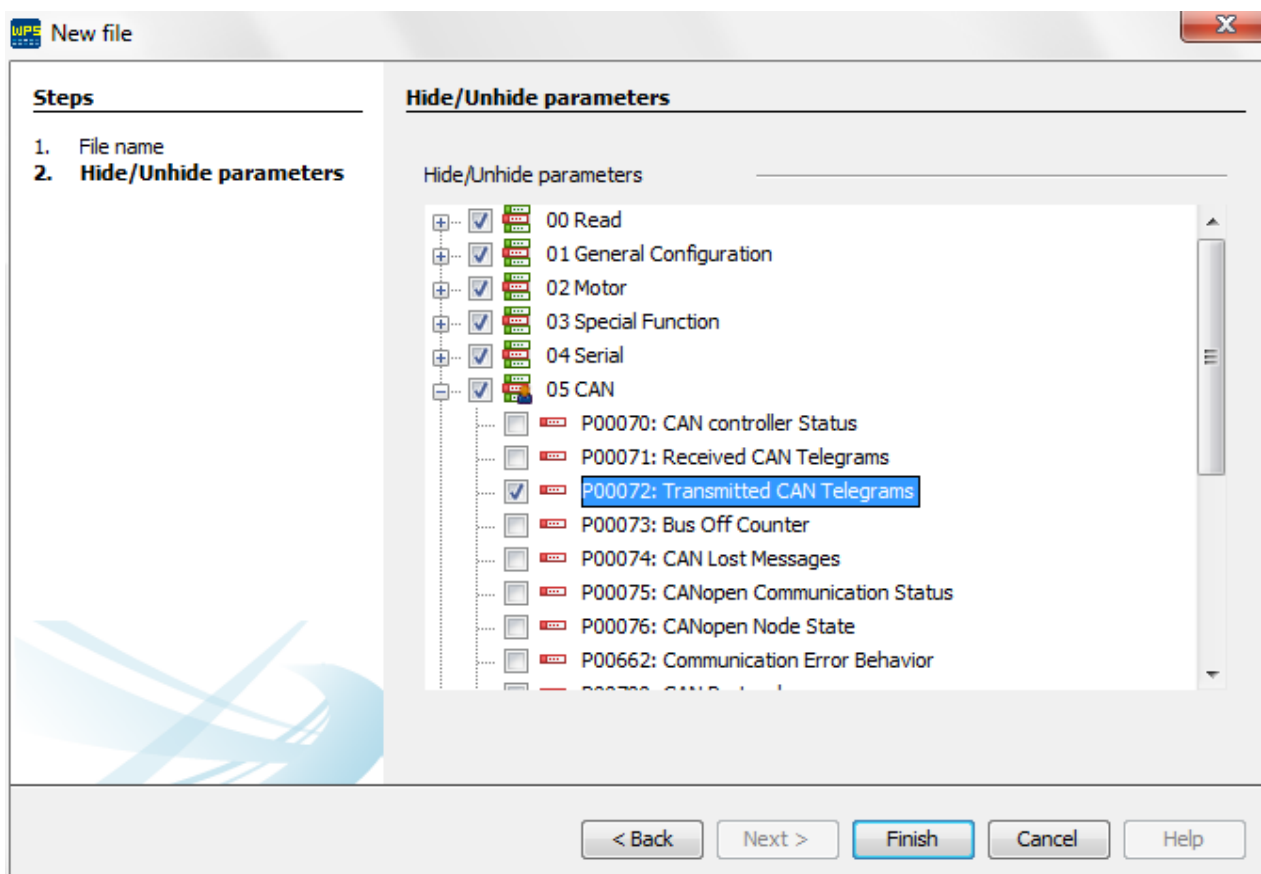


The image shows a 'New file' dialog box from the WPS software. It has a title bar with the WPS logo and a close button. The dialog is divided into two main sections: 'Steps' on the left and 'File name' on the right. The 'Steps' section lists two steps: '1. File name' and '2. Hide/Unhide parameters'. The 'File name' section contains five input fields: 'Filename:' with the text 'parameter01', 'File type:' with 'Parameter files', 'Resource:' with 'sca06', 'Configuration:' with 'cfg', and 'Path:' with 'C:/Projetos/WPS/trunk/Projects/cfg/sca06/parameter'. At the bottom of the dialog, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'. The 'Next >' button is highlighted with a blue border.

Steps	File name
1. File name	Filename: parameter01
2. Hide/Unhide parameters	File type: Parameter files
	Resource: sca06
	Configuration: cfg
	Path: C:/Projetos/WPS/trunk/Projects/cfg/sca06/parameter

< Back Next > Finish Cancel Help

3. Configurar quais parâmetros você deseja visualizar em sua tabela de parâmetros



4. Após a realização das etapas acima o arquivo de parâmetros estará criado e a parametrização do equipamento já poderá ser feita.

parameter01 38

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	...	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0		-9999	9999	0	rpm
P0003	Motor Current	0.0		-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	0		0	999	0	V
P0006	Drive Status	0: Ready		0	5	0: Ready	
P0008	DI3 to DI1 Status	0		0	7	0	
P0009	DI101 to DI106 Status	0		0	63	0	
P0010	DI107 to DI112 Status	0		0	63	0	
P0011	DI201 to DI206 Status	0		0	63	0	
P0012	DI207 to DI212 Status	0		0	63	0	
P0013	DI301 to DI306 Status	0		0	63	0	
P0014	DI307 to DI312 Status	0		0	63	0	
P0015	DO1 Status	0		0	1	0	
P0016	DO101 to DO106 Status	0		0	63	0	
P0017	DO201 to DO206 Status	0		0	63	0	
P0018	DO301 to DO306 Status	0		0	63	0	
P0021	Internal Air Temperature	0		0	1000	0	°C
P0022	Dissipator Temperature	0		0	1000	0	°C
P0023	Software Version	9.99		0.00	655.35	9.99	
P0024	Bootloader Version	0.00		0.00	655.35	0.00	
P0025	FPGA Project Version	0.00		0.00	655.35	0.00	
P0030	Present Alarm	0		0	2000	0	
P0031	Last Alarm	0		0	2000	0	
P0032	Last alarm Day.Month	0.00		0.00	31.12	0.00	
P0033	Last alarm year	0		0	4096	0	
P0034	Last alarm Hour.Min	0.00		0.00	23.59	0.00	
P0035	Present Fault	0		0	2000	0	
P0036	Last Fault	0		0	2000	0	
P0037	Last Fault Day.Month	0.00		0.00	31.12	0.00	
P0038	Last fault year	0		0	4096	0	
P0039	Last fault Hour.Min	0.00		0.00	23.59	0.00	
P0040	Second fault	0		0	2000	0	
P0041	Second fault Day.Month	0.00		0.00	31.12	0.00	
P0042	Second fault year	0		0	4096	0	
P0043	Second fault Hour.Min	0.00		0.00	23.59	0.00	
P0044	Third fault	0		0	2000	0	
P0045	Third fault Day.Month	0.00		0.00	31.12	0.00	
P0046	Third fault year	0		0	4096	0	
P0047	Third fault Hour.Min	0.00		0.00	23.59	0.00	
P0048	Lag fault	0		0	65535	0	
P0050	Real Axis: Actual Position	0		0	16383	0	
P0052	fractions of revolution	0		-16383	16383	0	
P0053	number of revolutions	0		-32768	32767	0	
P0056	Standart Counter - Low	0		0	65535	0	
P0057	Standart Counter - High	0		0	65535	0	
P0058	Counter 1 - Low	0		0	65535	0	
P0059	Counter 1 - High	0		0	65535	0	

10.2.6.3 Leitura e Escrita de Parâmetros

Existem três formas de se realizar a leitura e escrita dos parâmetros, sendo estas por tabela, seleção e grupo.

1. **Escrita da tabela.** O comando de escrever a tabela irá enviar todos os parâmetros visíveis na tela ao equipamento, caso ocorra algum erro no envio de algum parâmetro específico será exibida uma mensagem na janela de saída informando o erro. É importante observar que somente os parâmetros visíveis serão enviados, então é necessária atenção a qual nó da árvore de grupos de parâmetros você está visualizando. Exemplo: Se deseja escrever todos sem filtrar por grupo basta deixar selecionada a raiz da árvore.

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	M...	Minimum	Maxim...	Factory settings	Unit
P0002	Motor speed	0	0	-9999	9999	0	rpm
P0003	Motor Current	0.0	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	0	308	0	999	0	V
P0006	Drive Status	0: Ready	0: ...	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 Status	0	0	0	63	0	
P0012	DI207 to DI212 Status	0	0	0	63	0	
P0013	DI301 to DI306 Status	0	0	0	63	0	
P0014	DI307 to DI312 Status	0	0	0	63	0	
P0015	DO1 Status	0	0	0	1	0	
P0016	DO101 to DO106 Status	0	0	0	63	0	
P0017	DO201 to DO206 Status	0	0	0	63	0	
P0018	DO301 to DO306 Status	0	0	0	63	0	
P0021	Internal Air Temperature	0	41	0	1000	0	°C
P0022	Dissipator Temperature	0	33	0	1000	0	°C
P0023	Software Version	9.99	1.40	0.00	655.35	9.99	
P0024	Bootloader Version	0.00	20.03	0.00	655.35	0.00	
P0025	FPGA Project Version	0.00	0.03	0.00	655.35	0.00	
P0030	Present Alarm	0	0	0	2000	0	
P0031	Last Alarm	0	826	0	2000	0	
P0032	Last alarm Day..Month	0.00	3.01	0.00	31.12	0.00	
P0033	Last alarm year	0	1586	0	4096	0	
P0034	Last alarm Hour..Min	0.00	17.01	0.00	23.59	0.00	
P0035	Present Fault	0	0	0	2000	0	
P0036	Last Fault	0	2	0	2000	0	
P0037	Last Fault Day..Month	0.00	3.01	0.00	31.12	0.00	
P0038	Last fault year	0	1594	0	4096	0	
P0039	Last fault Hour..Min	0.00	17.13	0.00	23.59	0.00	

Output - Default output

*** Writing parameter ***

P1035 31%

WEG SCA-06 V1.40 localhost:502 USB/@0

2. **Leitura da tabela.** O comando de ler a tabela irá ler todos os parâmetros do equipamento, caso ocorra algum erro na leitura de algum parâmetro específico será exibida uma mensagem na janela de saída informando o erro. É importante observar que somente os parâmetros visíveis serão lidos, então é necessária atenção a qual nó da árvore de grupos de parâmetros você está visualizado. Exemplo: Se deseja ler todos sem filtrar por grupo basta deixar selecionada a raiz da árvore.

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	...	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0	0	-9999	9999	0	rpm
P0003	Motor Current	-0.1	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	308	308	0	999	0	V
P0006	Drive Status	0: Ready	0: ...	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 Status	0	0	0	63	0	
P0012	DI207 to DI212 Status	0	0	0	63	0	
P0013	DI301 to DI306 Status	0	0	0	63	0	
P0014	DI307 to DI312 Status	0	0	0	63	0	
P0015	DO1 Status	0	0	0	1	0	
P0016	DO101 to DO106 Status	0	0	0	63	0	
P0017	DO201 to DO206 Status	0	0	0	63	0	
P0018	DO301 to DO306 Status	0	0	0	63	0	
P0021	Internal Air Temperature	41	41	0	1000	0	°C
P0022	Dissipator Temperature	33	33	0	1000	0	°C
P0023	Software Version	1.40	1.40	0.00	655.35	9.99	
P0024	Bootloader Version	20.03	20...	0.00	655.35	0.00	
P0025	FPGA Project Version	0.03	0.03	0.00	655.35	0.00	
P0030	Present Alarm	0	0	0	2000	0	
P0031	Last Alarm	826	826	0	2000	0	
P0032	Last alarm Day.Month	3.01	3.01	0.00	31.12	0.00	
P0033	Last alarm year	1586	1586	0	4096	0	
P0034	Last alarm Hour.Min	17.01	17...	0.00	23.59	0.00	
P0035	Present Fault	0	0	0	2000	0	
P0036	Last Fault	2	2	0	2000	0	
P0037	Last Fault Day.Month	3.01	3.01	0.00	31.12	0.00	
P0038	Last fault year	1594	1594	0	4096	0	
P0039	Last fault Hour.Min	17.13	17...	0.00	23.59	0.00	
P0040	Second fault	2	2	0	2000	0	
P0041	Second fault Day.Month	3.01	3.01	0.00	31.12	0.00	
P0042	Second fault year	1594	1594	0	4096	0	
P0043	Second fault Hour.Min	17.12	17...	0.00	23.59	0.00	
P0044	Third fault	32	32	0	2000	0	
P0045	Third fault Day.Month	1.01	1.01	0.00	31.12	0.00	

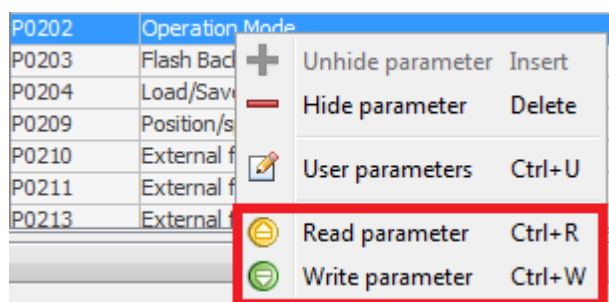
Output - Default output

*** Reading parameter ***

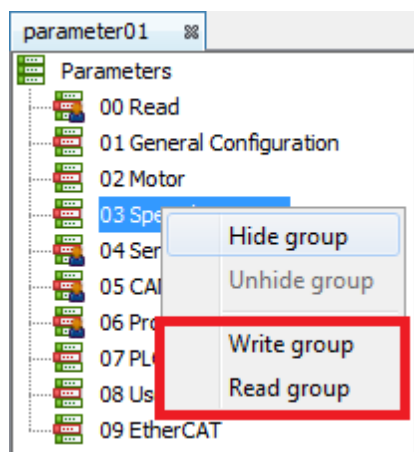
P0918 43%

WEG SCA-06 V1.40 localhost:502 USB/@0

3. **Leitura/Escrita de parâmetros específicos.** Para ler/escrever somente alguns parâmetros um ou mais, basta selecioná-los na tabela e clicar com o botão direito do mouse escolhendo a opção desejada, sendo estas ler ou escrever parâmetro.



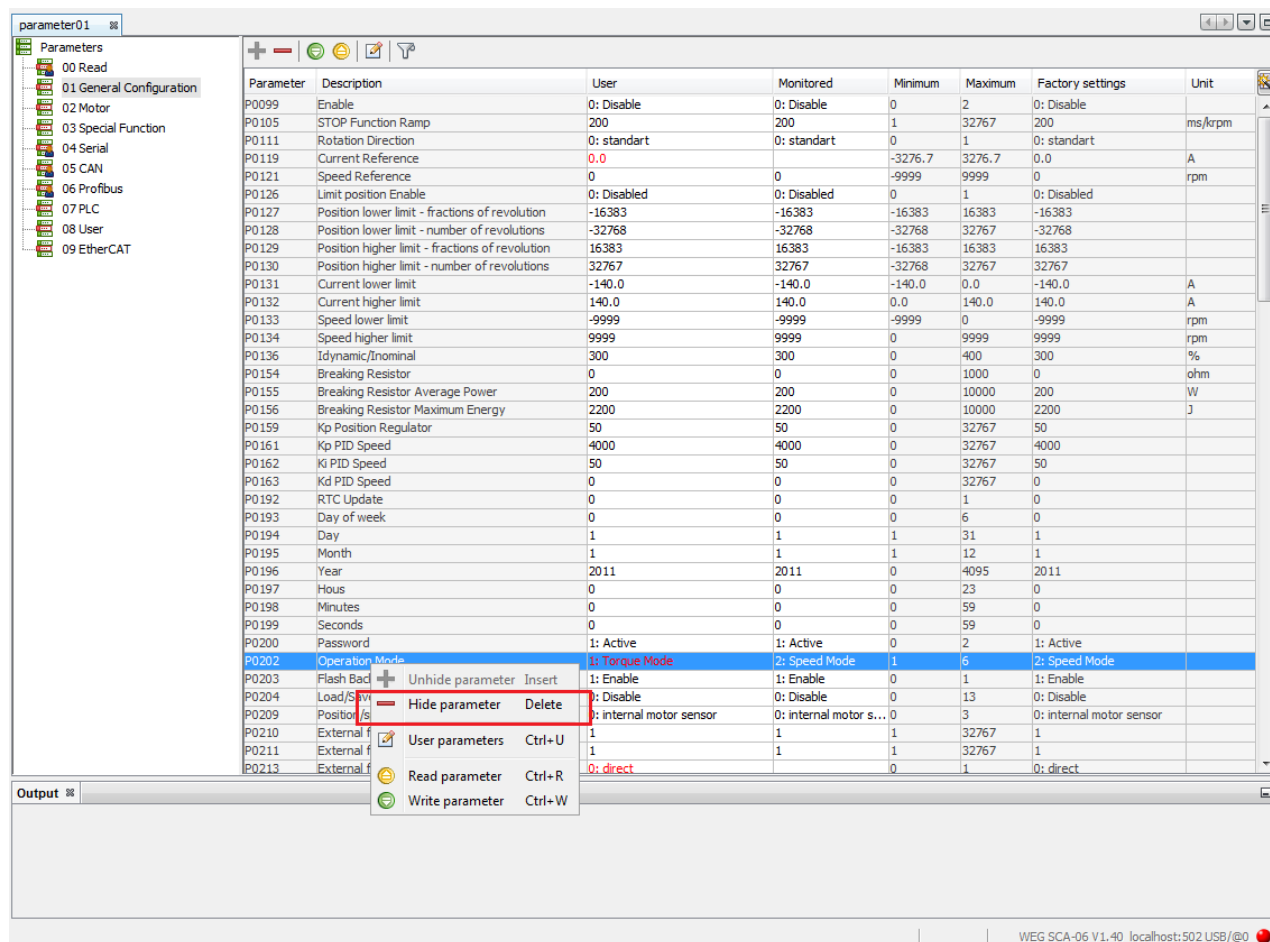
4. **Leitura/escrita de grupo de parâmetros.** Para ler/escrever somente um grupo de parâmetros, basta selecioná-los na árvore de grupos e clicar com o botão direito do mouse escolhendo a opção desejada, sendo estas ler ou escrever grupo.



10.2.6.4 Ocultar/Desocultar Parâmetros e Grupos de Parâmetros

Os parâmetros podem ser ocultos/desocultos de duas formas, sendo estas de forma individual ou por grupo.

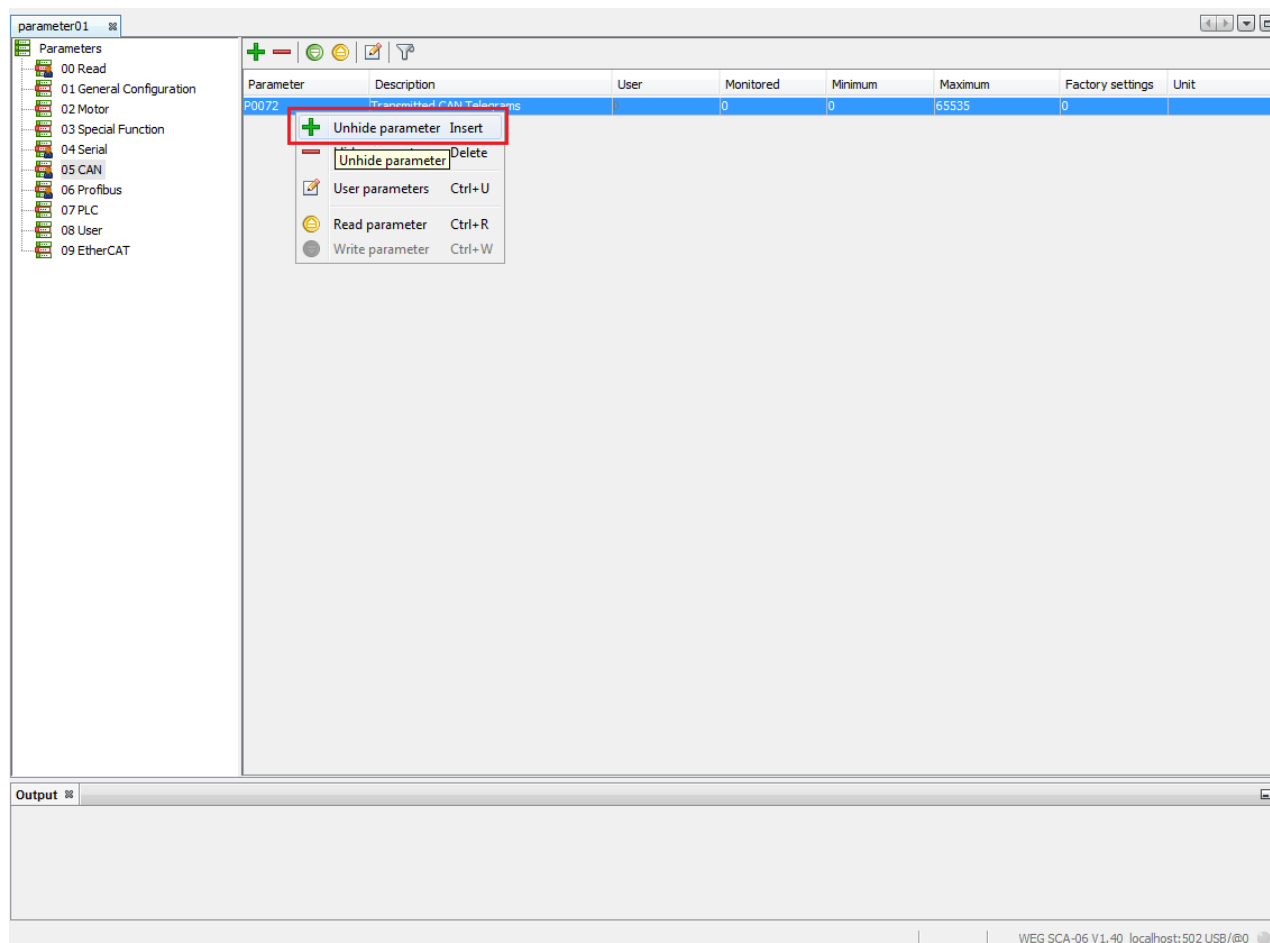
1. **Ocultar Parâmetros.** Para ocultar um parâmetro individualmente basta clicar sobre o parâmetro desejado com o botão direito do mouse e selecionar a opção **Ocultar Parâmetro**, se preferir também pode apenas pressionar a tecla **Delete**.



2. **Desocultar Parâmetros.** Para exibir os parâmetros que foram ocultos clique com o botão direito do mouse e escolha a opção **Desocultar parâmetros** ou pressione a tecla **Insert**, após isto uma nova janela se abrirá exibindo os parâmetros que estão ocultos então basta selecionar os desejados e confirmar.

**NOTA!**

Os parâmetros exibidos nesta nova janela são somente os que pertencem ao filtro atual conforme seleção na árvore de grupos de parâmetros, nas imagens abaixo o grupo CAN está selecionado isto significa que somente os parâmetros ocultos deste grupo serão exibidos.



parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams		0	0	65535	0	

Select parameters

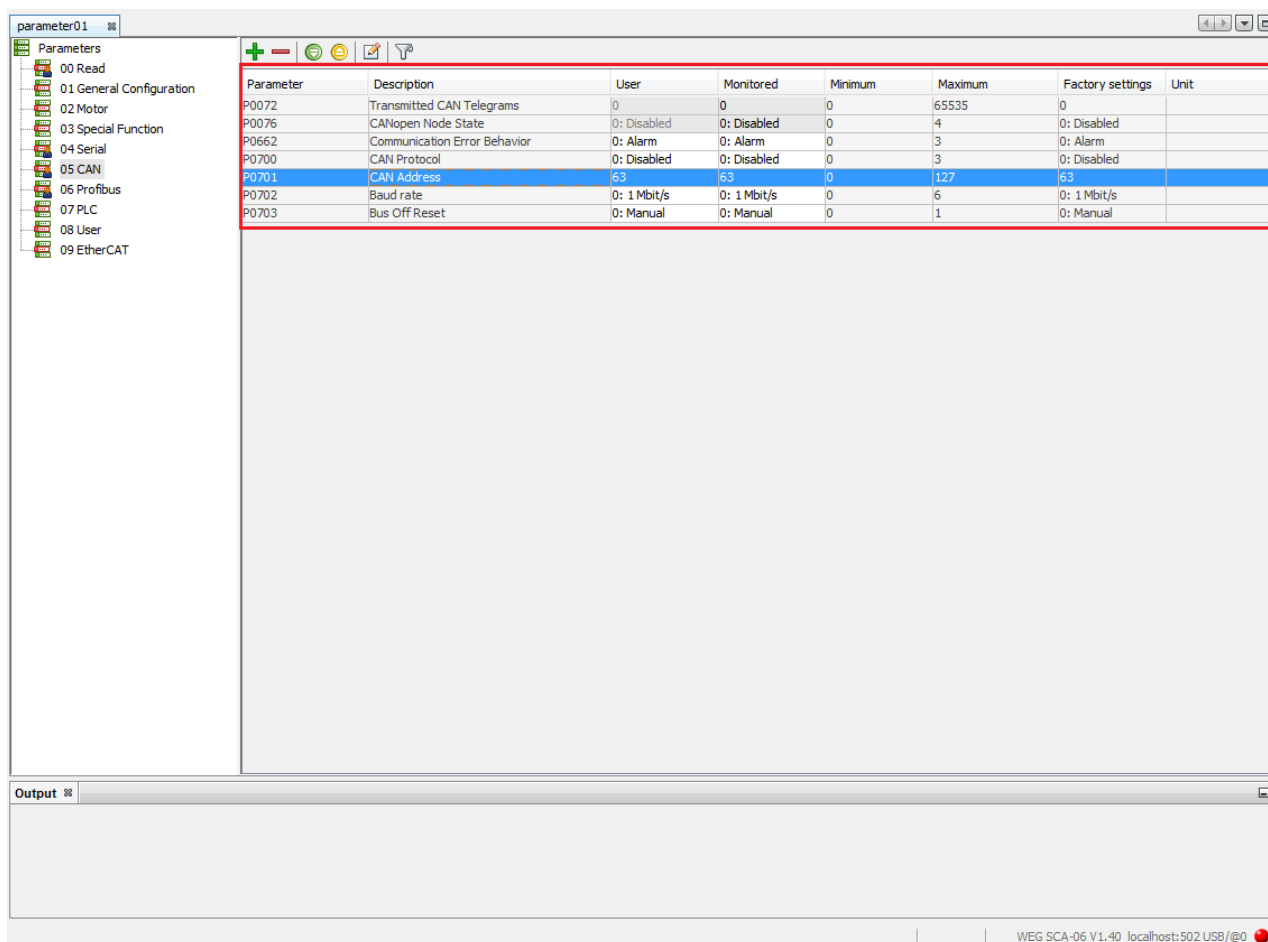
Select to unhide:

- CAN controller Status
- Received CAN Telegrams
- Bus Off Counter
- CAN Lost Messages
- CANopen Communication Status
- CANopen Node State
- Communication Error Behavior
- CAN Protocol
- CAN Address
- Baud rate
- Bus Off Reset
- Follow Type
- Follow COB ID
- Follow Period

OK Cancel

Output

WEG SCA-06 V1.40 localhost:502.USB/@0



The screenshot displays the 'parameter01' configuration window. On the left, a tree view lists parameter groups: 00 Read, 01 General Configuration, 02 Motor, 03 Special Function, 04 Serial, 05 CAN, 06 Profibus, 07 PLC, 08 User, and 09 EtherCAT. The main area shows a table of parameters for the selected group. The table has columns for Parameter, Description, User, Monitored, Minimum, Maximum, Factory settings, and Unit. The 'CAN' group parameters are highlighted in blue.

Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams	0	0	0	65535	0	
P0075	CANopen Node State	0: Disabled	0: Disabled	0	4	0: Disabled	
P0662	Communication Error Behavior	0: Alarm	0: Alarm	0	3	0: Alarm	
P0700	CAN Protocol	0: Disabled	0: Disabled	0	3	0: Disabled	
P0701	CAN Address	63	63	0	127	63	
P0702	Baud rate	0: 1 Mbit/s	0: 1 Mbit/s	0	6	0: 1 Mbit/s	
P0703	Bus Off Reset	0: Manual	0: Manual	0	1	0: Manual	

At the bottom of the window, the status bar shows 'WEG SCA-06 V1.40 localhost:502.USB/@0'.

3. **Ocultar Grupos de Parâmetros.** Para ocultar um grupo de parâmetros basta selecionar o grupo desejado na árvore e usar a opção **Ocultar Grupo**.

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Sp **Hide group**
- 04 Set
- 05 CA Unhide group
- 06 Pro
- 07 PLC
- 08 Use
- 09 EtherCAT

Write group

Read group

Param...	Description	User	Mo...	Minimum	Maximum	Factory settings	Unit
P0365	Encoder temperature	0.0		-3276.7	3276.7	0.0	
P0368	Manufacture password	0	0	0	65535	0	
P0369	Electrical position	0	50828	0	65535	0	
P0370	Encoder temperature	0		-3276.7	3276.7	0	
P0371	Read Encoder filter	3000	3000	3000	37500	3000	
P0372	Encoder filter	3000	3000	3000	37500	3000	
P0375	Encoder Offset	0	0	-3276.7	3276.7	0	
P0490	Load Absolute position	0	0	0	1	0	
P0492	User reference: fractions of revolution	0	0	-16383	16383	0	
P0493	User reference: number of revolutions	0	0	-32768	32767	0	
P0500	Count Mode: Standart Counter	0: Disabled	0: Di...	0	4	0: Disabled	
P0502	Load counter	0	0	0	1	0	
P0503	Counter Value - Low	0	0	0	65535	0	
P0504	Counter Value - High	0	0	0	65535	0	
P0506	Counter number of pulses/turns	1024	1024	1	65535	1024	
P0507	Counter speed filter	500	500	1	4000	500	Hz
P0510	Count Mode: Counter 1	0: Disabled	0: Di...	0	3	0: Disabled	
P0511	Null Pulse 1 options	0: Disabled	0: Di...	0	5	0: Disabled	
P0512	Load Counter 1 value	0	0	0	1	0	
P0513	Counter 1 value - Low	0	0	0	65535	0	
P0514	Counter 1 value - High	0	0	0	65535	0	
P0516	Counter 1 number of pulses/turns	1024	1024	1	65535	1024	
P0517	Counter 1 speed filter	500	500	1	4000	500	Hz
P0519	Error Enable	0: Disable	0: Di...	0	2	0: Disable	
P0520	Count Mode: Counter 2	0: Disabled	0: Di...	0	3	0: Disabled	
P0521	Null Pulse 2 options	0: Disabled	0: Di...	0	5	0: Disabled	
P0522	Load Counter 2 value	0	0	0	1	0	
P0523	Counter 2 value - Low	0	0	0	65535	0	
P0524	Counter 2 value - High	0	0	0	65535	0	
P0526	Counter 2 number of pulses/turns	1024	1024	1	65535	1024	
P0527	Counter 2 speed filter	500	500	1	4000	500	Hz
P0529	Error Enable	0: Disabled	0: Di...	0	2	0: Disabled	
P0531	Stop function: level or edge	0: stop activation by level	0: st...	0	1	0: stop activation by level	
P0550	Trigger 1 Signal Source	0: Disabled	0: Di...	0	48	0: Disabled	
P0551	Trigger 1 Value	0	0	-32768	32767	0	
P0552	Trigger 1 Condition	0: Higher or equal than reference value	0: Hi...	0	1	0: Higher or equal than reference value	
P0553	Trigger 2 Signal Source	0: Disable	0: Di...	0	48	0: Disable	
P0554	Trioper 2 Value	0		-32768	32767	0	

Output

WEG SCA-06 V1.40 localhost:502.USB/@0

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 04 Serial**
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Parameter	Description	User	Monito...	Minimum	Maximum	Factory settings	Unit
P0650	Serial Address 1	1	1	1	247	1	
P0652	Bit Rate of Serial 1	1: 9600 bits/s	1: 9600 ...	0	11	1: 9600 bits/s	
P0653	Data bits, parity and stop bit of Serial 1	3: 8bits, no parity, 2stop bit	3: 8bits,...	0	11	3: 8bits, no parity, 2stop bit	
P0654	Serial 1 Protocol	2: ModBus	2: ModBus	1	2	2: ModBus	
P0656	Serial Address 2	1	1	0	247	1	
P0658	Bit Rate of Serial 2	1: 9600 bits/s	1: 9600 ...	0	11	1: 9600 bits/s	
P0659	Data bits, parity and stop bit of Serial 2	3: 8bits, no parity, 2stop bit	3: 8bits,...	0	11	3: 8bits, no parity, 2stop bit	
P0660	Serial 2 Protocol	2: ModBus	2: ModBus	1	2	2: ModBus	
P0663	Timeout for communication	0.0	0.0	0.0	999.9	0.0	s
P0664	Save parameters in non volatile memory	1: Save parameters	1: Save ...	0	1	1: Save parameters	
P0667	Parameters serial access remapping	0: Read/Write in parameters	0: Read/...	0	1	0: Read/Write in parameters	

Output

WEG SCA-06 V1.40 localhost:502.USB/@0

4. **Desocultar Grupos de Parâmetros.** Para exibir um grupo de parâmetros que foi oculto, basta selecionar a raiz da árvore de grupos e selecionar a opção **Desocultar grupo**, na sequência uma janela se abrirá exibindo os grupos que estão ocultos então basta selecionar o grupo que deseja desocultar.

parameter01

Parameters

- 00 Hide group
- 01 Unhide group
- 02 Write group
- 03 Read group
- 06
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	Monitored	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0	0	-9999	9999	0	rpm
P0003	Motor Current	0.0	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	0	307	0	999	0	V
P0006	Drive Status	0: Ready	0: Ready	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 Status	0	0	0	63	0	
P0012	DI207 to DI212 Status	0	0	0	63	0	
P0013	DI301 to DI306 Status	0	0	0	63	0	
P0014	DI307 to DI312 Status	0	0	0	63	0	
P0015	DO1 Status	0	0	0	1	0	
P0016	DO101 to DO106 Status	0	0	0	63	0	
P0017	DO201 to DO206 Status	0	0	0	63	0	
P0018	DO301 to DO306 Status	0	0	0	63	0	
P0021	Internal Air Temperature	0	41	0	1000	0	°C
P0022	Dissipator Temperature	0	33	0	1000	0	°C
P0023	Software Version	9.99	1.40	0.00	655.35	9.99	
P0024	Bootloader Version	0.00	20.03	0.00	655.35	0.00	
P0025	FPGA Project Version	0.00	0.03	0.00	655.35	0.00	
P0030	Present Alarm	0	0	0	2000	0	
P0031	Last Alarm	0	826	0	2000	0	
P0032	Last alarm Day,Month	0.00	3.01	0.00	31.12	0.00	
P0033	Last alarm year	0	1586	0	4096	0	
P0034	Last alarm Hour,Min	0.00	17.01	0.00	23.59	0.00	
P0035	Present Fault	0	0	0	2000	0	
P0036	Last Fault	0	2	0	2000	0	
P0037	Last Fault Day,Month	0.00	3.01	0.00	31.12	0.00	
P0038	Last fault year	0	1594	0	4096	0	
P0039	Last fault Hour,Min	0.00	17.13	0.00	23.59	0.00	
P0040	Second fault	0	2	0	2000	0	
P0041	Second fault Day,Month	0.00	3.01	0.00	31.12	0.00	
P0042	Second fault year	0	1594	0	4096	0	
P0043	Second fault Hour,Min	0.00	17.12	0.00	23.59	0.00	
P0044	Third fault	0	32	0	2000	0	
P0045	Third fault Day,Month	0.00	1.01	0.00	31.12	0.00	
P0046	Third fault year	0	1577	0	4096	0	
P0047	Third fault Hour,Min	0.00		0.00	23.59	0.00	

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	Monitored	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0	0	-9999	9999	0	/rpm
P0003	Motor Current	0.0	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Ud)	0	308	0	999	0	V
P0006	Drive Status	0: Ready	0: Ready	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 S			0	63	0	
P0012	DI207 to DI212 S			0	63	0	
P0013	DI301 to DI306 S			0	63	0	
P0014	DI307 to DI312 S			0	63	0	
P0015	DO1 Status			0	1	0	
P0016	DO101 to DO106			0	63	0	
P0017	DO201 to DO206			0	63	0	
P0018	DO301 to DO306			0	63	0	
P0021	Internal Air Temp		1	0	1000	0	°C
P0022	Dissipator Tempe		3	0	1000	0	°C
P0023	Software Version		.40	0.00	655.35	9.99	
P0024	Bootloader Version		0.03	0.00	655.35	0.00	
P0025	FPGA Project Vers		.03	0.00	655.35	0.00	
P0030	Present Alarm			0	2000	0	
P0031	Last Alarm		26	0	2000	0	
P0032	Last alarm Day,Mo		.01	0.00	31.12	0.00	
P0033	Last alarm year		586	0	4096	0	
P0034	Last alarm Hour,Min		7.01	0.00	23.59	0.00	
P0035	Present Fault			0	2000	0	
P0036	Last Fault			0	2000	0	
P0037	Last Fault Day,Mo		.01	0.00	31.12	0.00	
P0038	Last fault year		594	0	4096	0	
P0039	Last fault Hour,Min		7.13	0.00	23.59	0.00	
P0040	Second fault		2	0	2000	0	
P0041	Second fault Day,Month	0.00	3.01	0.00	31.12	0.00	
P0042	Second fault year	0	1594	0	4096	0	
P0043	Second fault Hour,Min	0.00	17.12	0.00	23.59	0.00	
P0044	Third fault	0	32	0	2000	0	
P0045	Third fault Day,Month	0.00	1.01	0.00	31.12	0.00	
P0046	Third fault year	0	1577	0	4096	0	
P0047	Third fault Hour,Min	0.00		0.00	23.59	0.00	

Select groups

Select to unhide:

03 Special Function

OK Cancel

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	M...	Minimum	Maxim...	Factory settings	Unit
P0002	Motor Speed	0		-9999	9999	0	rpm
P0003	Motor Current	0.0		-999.9	999.9	0.0	A
P0004	DC Link Voltage (Ud)	0		0	999	0	V
P0006	Drive Status	0: Ready		0	5	0: Ready	
P0008	DI3 to DI1 Status	0		0	7	0	
P0009	DI101 to DI106 Status	0		0	63	0	
P0010	DI107 to DI112 Status	0		0	63	0	
P0011	DI201 to DI206 Status	0		0	63	0	
P0012	DI207 to DI212 Status	0		0	63	0	
P0013	DI301 to DI306 Status	0		0	63	0	
P0014	DI307 to DI312 Status	0		0	63	0	
P0015	DO1 Status	0		0	1	0	
P0016	DO101 to DO106 Status	0		0	63	0	
P0017	DO201 to DO206 Status	0		0	63	0	
P0018	DO301 to DO306 Status	0		0	63	0	
P0021	Internal Air Temperature	0		0	1000	0	°C
P0022	Dissipator Temperature	0		0	1000	0	°C
P0023	Software Version	9.99		0.00	655.35	9.99	
P0024	Bootloader Version	0.00		0.00	655.35	0.00	
P0025	FPGA Project Version	0.00		0.00	655.35	0.00	
P0030	Present Alarm	0		0	2000	0	
P0031	Last Alarm	0		0	2000	0	
P0032	Last alarm Day.Month	0.00		0.00	31.12	0.00	
P0033	Last alarm year	0		0	4096	0	
P0034	Last alarm Hour.Min	0.00		0.00	23.59	0.00	
P0035	Present Fault	0		0	2000	0	
P0036	Last Fault	0		0	2000	0	
P0037	Last Fault Day.Month	0.00		0.00	31.12	0.00	
P0038	Last fault year	0		0	4096	0	
P0039	Last fault Hour.Min	0.00		0.00	23.59	0.00	
P0040	Second fault	0		0	2000	0	
P0041	Second fault Day.Month	0.00		0.00	31.12	0.00	
P0042	Second fault year	0		0	4096	0	
P0043	Second fault Hour.Min	0.00		0.00	23.59	0.00	
P0044	Third fault	0		0	2000	0	
P0045	Third fault Day.Month	0.00		0.00	31.12	0.00	
P0046	Third fault year	0		0	4096	0	

Output

5. **Ocultar e Exibir Parâmetros e Grupos de Parâmetros.** Através desta opção se tem o controle total dos parâmetros e grupos de parâmetros, é possível ocultar e desocultar parâmetros individuais, múltiplos parâmetros, grupos individuais e múltiplos grupos em uma mesma ação.

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor

Context Menu:

- Open
- Hide/unhide parameters
- Cut (Ctrl+X)
- Copy (Ctrl+C)
- Delete
- Rename...

Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams	U	U	U	65535	U	
P0080	Virtual Axis Speed	0	0	-9999	9999	0	rpm
P0082	Virtual Axis - fractions of revolution	0	0	-16383	16383	0	
P0083	Virtual Axis - numbers of revolutions	0	0	-32768	32767	0	
P0084	Day of week	0	1	0	6	0	
P0085	Day	1	3	1	31	1	
P0086	Month	1	1	1	12	1	
P0087	Year	2009	1594	0	4095	2009	
P0088	Hours	0	20	0	23	0	
P0089	Minutes	0	22	0	59	0	
P0090	Seconds	0	52	0	59	0	
P0091	Slot 1 ID	0	0	0	65535	0	
P0092	Slot 2 ID	0	0	0	65535	0	
P0093	Slot 3 ID	0	512	0	65535	0	
P0095	Optional Card	0	0	0	1000	0	
P0097	Nominal Current	0.0	5.0	0.0	999.9	0.0	
P0098	Line Rated Voltage	0	2: 220 V	0	10	0	
P0228	Iq RMS Cicle	0.0	0.0	-3276.8	3276.7	0.0	A
P0740	Profibus Comm. Status	0: Disabled	0: Disabled	0	6	0: Disabled	
P0850	EtherCAT: FW Revision	0	0	0	65535	0	
P0851	ECAT: AccessoryStatus	0: Inactive	0: Inactive	0	4	0: Inactive	
P0852	ECAT: Link Status	0	0	0	65535	0	
P0853	ECAT: Slave Status	0: Inactive	0: Inactive	0	8	0: Inactive	
P0854	EtherCAT: Reserved	0	0	0	65535	0	
P0855	ECAT: TxPDO Config	0	0	0	65535	0	
P0856	ECAT: TxPDO Data Size	0	0	0	32	0	
P0857	ECAT: RxPDO Config	0	0	0	65535	0	
P0858	ECAT: RxPDO Data Size	0	0	0	32	0	
P0859	ECAT: Data Update	0.0	0.0	0.0	1000.0	0.0	ms
P0944	Fault Message Counter	0	0	0	65535	0	
P0947	Fault Number	0	0	0	65535	0	
P0963	Profibus Baud Rate	0: 9.6 kbit/s	5: Not Detected	0	11	0: 9.6 kbit/s	
P0964	Drive Unit Ident.	0	367	0	65535	0	
P0965	Profile Ident. Number	0	809	0	65535	0	
P0967	Control Word 1	0	0	0	65535	0	
P0968	Status Word 1	0	0	0	65535	0	
P1000	PLC State	0: No Program	4: Stopped Prog.	0	5	0: No Program	
P1001	Scan Time	0.0	0.0	0.0	6553.5	0.0	ms

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Hide/unhide parameters dialog:

- ☒ 00 Read
- ☒ 01 General Configuration
- ☒ 02 Motor
- ☒ 03 Special Function
- ☒ 04 Serial
- ☒ 05 CAN
- ☒ 06 Profibus
- ☒ 07 PLC
- ☒ 08 User
- ☒ 09 EtherCAT

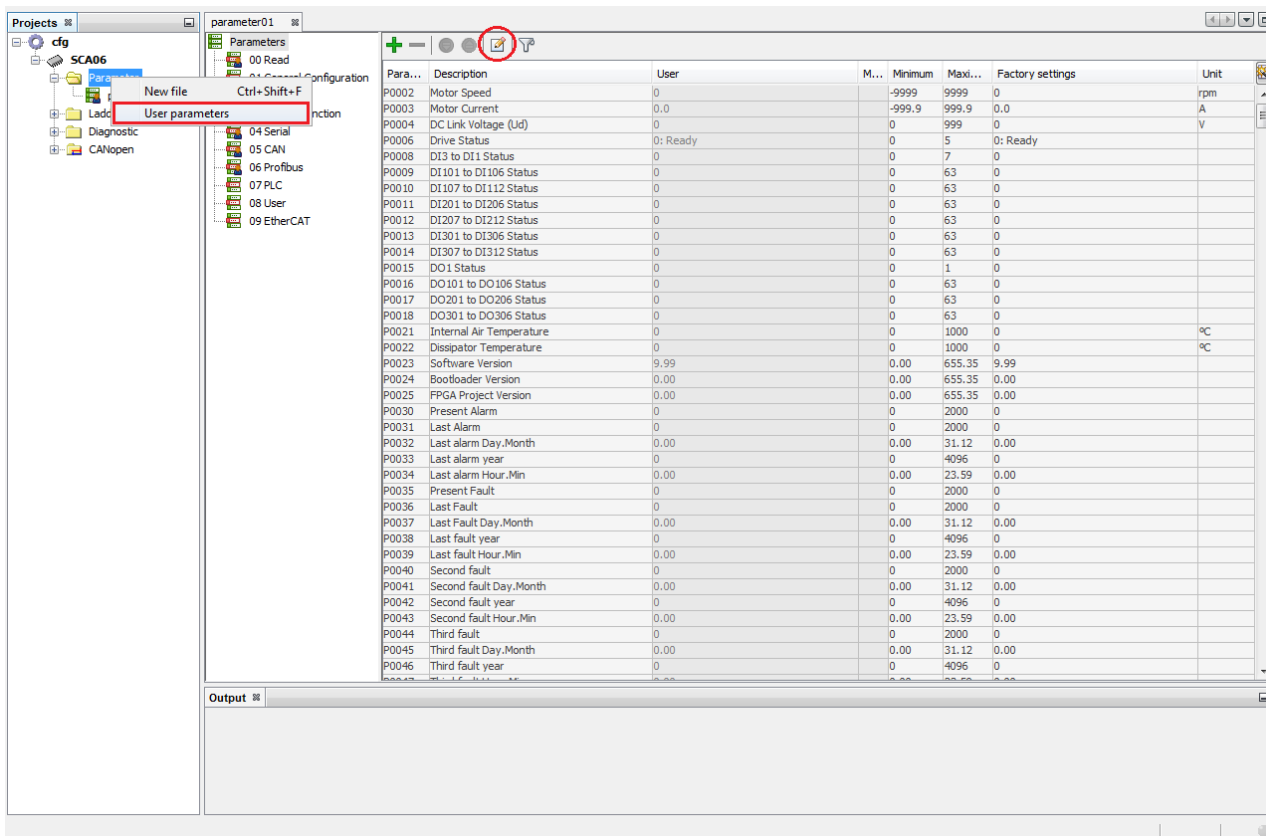
Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams	U	U	U	65535	U	
P0080	Virtual Axis Speed	0	0	-9999	9999	0	rpm
P0082	Virtual Axis - fractions of revolution	0	0	-16383	16383	0	
P0083	Virtual Axis - numbers of revolutions	0	0	-32768	32767	0	
P0084	Day of week	0	1	0	6	0	
P0085	Day	1	3	1	31	1	
P0086	Month	1	1	1	12	1	
P0087	Year	2009	1594	0	4095	2009	
P0088	Hours	0	20	0	23	0	
P0089	Minutes	0	22	0	59	0	
P0090	Seconds	0	52	0	59	0	
P0091	Slot 1 ID	0	0	0	65535	0	
P0092	Slot 2 ID	0	0	0	65535	0	
P0093	Slot 3 ID	0	512	0	65535	0	
P0095	Optional Card	0	0	0	1000	0	
P0097	Nominal Current	0.0	5.0	0.0	999.9	0.0	
P0098	Line Rated Voltage	0	2: 220 V	0	10	0	
P0228	Iq RMS Cicle	0.0	0.0	-3276.8	3276.7	0.0	A
P0740	Profibus Comm. Status	0: Disabled	0: Disabled	0	6	0: Disabled	
P0850	EtherCAT: FW Revision	0	0	0	65535	0	
P0851	ECAT: AccessoryStatus	0: Inactive	0: Inactive	0	4	0: Inactive	
P0852	ECAT: Link Status	0	0	0	65535	0	
P0853	ECAT: Slave Status	0: Inactive	0: Inactive	0	8	0: Inactive	
P0854	EtherCAT: Reserved	0	0	0	65535	0	
P0855	ECAT: TxPDO Config	0	0	0	65535	0	
P0856	ECAT: TxPDO Data Size	0	0	0	32	0	
P0857	ECAT: RxPDO Config	0	0	0	65535	0	
P0858	ECAT: RxPDO Data Size	0	0	0	32	0	
P0859	ECAT: Data Update	0.0	0.0	0.0	1000.0	0.0	ms
P0944	Fault Message Counter	0	0	0	65535	0	
P0947	Fault Number	0	0	0	65535	0	
P0963	Profibus Baud Rate	0: 9.6 kbit/s	5: Not Detected	0	11	0: 9.6 kbit/s	
P0964	Drive Unit Ident.	0	367	0	65535	0	
P0965	Profile Ident. Number	0	809	0	65535	0	
P0967	Control Word 1	0	0	0	65535	0	
P0968	Status Word 1	0	0	0	65535	0	
P1000	PLC State	0: No Program	4: Stopped Prog.	0	5	0: No Program	
P1001	Scan Time	0.0	0.1	0.0	6553.5	0.0	ms

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

10.2.6.5 Parâmetros do Usuário

Para abrir a tela de configuração dos parâmetros de usuário basta clicar na opção **Parâmetros do Usuário** no nó **Parâmetros** da árvore do projeto ou clicar no ícone indicado na barra de ferramentas do arquivo de parâmetros.



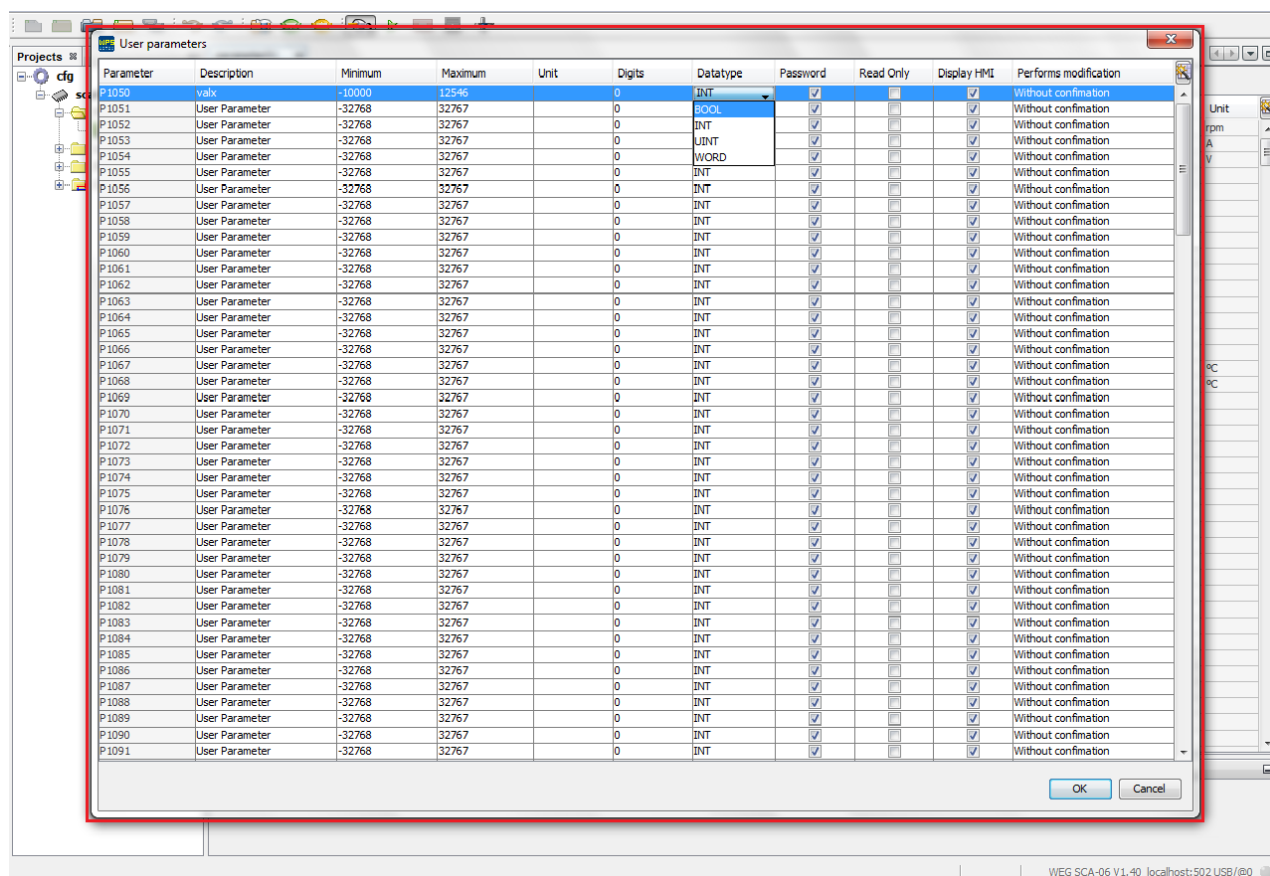
Tela de Configuração

Na tela de configuração dos parâmetros dos usuários é possível definir vários atributos dos parâmetros como descrição, valores mínimos e máximos, unidade, dígitos, tipo de dados entre outros.



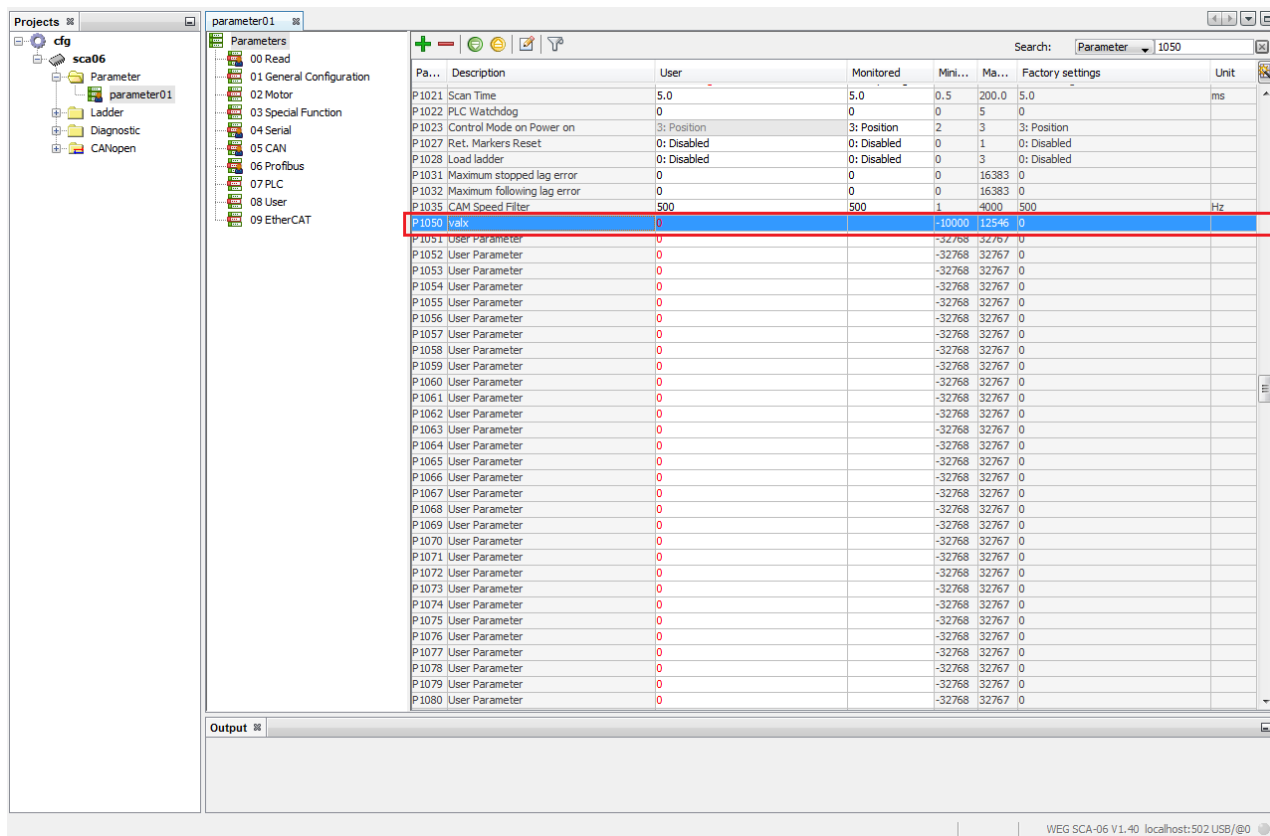
NOTA!

Estas configurações serão exibidas automaticamente no arquivo de parâmetros, porém para que sejam enviados ao equipamento é necessário fazer o *download* do recurso.



Exibição do parâmetro do usuário

Na tabela de parâmetros os parâmetros do usuário serão exibidos conforme configurados na tela de configuração.



Pa...	Description	User	Monitored	Mini...	Ma...	Factory settings	Unit
P1021	Scan Time	5.0	5.0	0.5	200.0	5.0	ms
P1022	PLC Watchdog	0	0	0	5	0	
P1023	Control Mode on Power on	3: Position	3: Position	2	3	3: Position	
P1027	Ret. Markers Reset	0: Disabled	0: Disabled	0	1	0: Disabled	
P1028	Load ladder	0: Disabled	0: Disabled	0	3	0: Disabled	
P1031	Maximum stopped lag error	0	0	0	16383	0	
P1032	Maximum following lag error	0	0	0	16383	0	
P1035	CAM Speed Filter	500	500	1	4000	500	Hz
P1050	Ivalx	0		-10000	12546	0	
P1051	User Parameter	0		-32768	32767	0	
P1052	User Parameter	0		-32768	32767	0	
P1053	User Parameter	0		-32768	32767	0	
P1054	User Parameter	0		-32768	32767	0	
P1055	User Parameter	0		-32768	32767	0	
P1056	User Parameter	0		-32768	32767	0	
P1057	User Parameter	0		-32768	32767	0	
P1058	User Parameter	0		-32768	32767	0	
P1059	User Parameter	0		-32768	32767	0	
P1060	User Parameter	0		-32768	32767	0	
P1061	User Parameter	0		-32768	32767	0	
P1062	User Parameter	0		-32768	32767	0	
P1063	User Parameter	0		-32768	32767	0	
P1064	User Parameter	0		-32768	32767	0	
P1065	User Parameter	0		-32768	32767	0	
P1066	User Parameter	0		-32768	32767	0	
P1067	User Parameter	0		-32768	32767	0	
P1068	User Parameter	0		-32768	32767	0	
P1069	User Parameter	0		-32768	32767	0	
P1070	User Parameter	0		-32768	32767	0	
P1071	User Parameter	0		-32768	32767	0	
P1072	User Parameter	0		-32768	32767	0	
P1073	User Parameter	0		-32768	32767	0	
P1074	User Parameter	0		-32768	32767	0	
P1075	User Parameter	0		-32768	32767	0	
P1076	User Parameter	0		-32768	32767	0	
P1077	User Parameter	0		-32768	32767	0	
P1078	User Parameter	0		-32768	32767	0	
P1079	User Parameter	0		-32768	32767	0	
P1080	User Parameter	0		-32768	32767	0	

10.2.7 Ladder

[Componentes e Blocos](#)

[Cam Profiles](#)

[Estruturas](#)

10.2.7.1 Componentes e Blocos

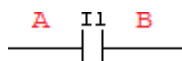
Ver a seguir.

10.2.7.1.1 Contact

10.2.7.1.1.1 NOCONTACT

Contato normalmente aberto.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	I1	BOOL	Entrada de controle do bloco

Funcionamento

Quando a variável I1 está com o valor FALSE, B recebe FALSE.

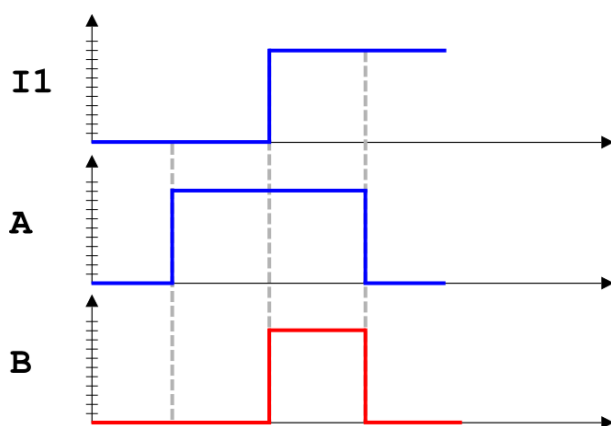
Quando a variável I1 está com o valor TRUE, B recebe o valor de A.



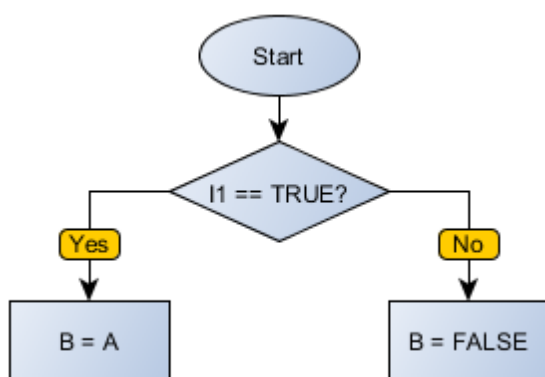
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

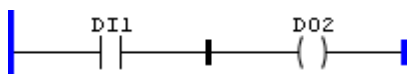
Diagrama



Fluxograma do Bloco



Exemplo



O exemplo acima realiza a transferência do valor da entrada digital DI1 para a saída digital DO2.

10.2.7.1.1.2 NCCONTACT

Contato normalmente fechado.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	I1	BOOL	Entrada de controle do bloco

Funcionamento

Quando a variável I1 está com o valor TRUE, B recebe FALSE.

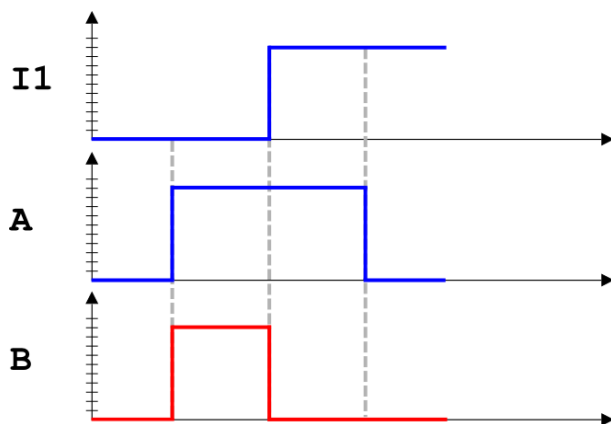
Quando a variável I1 está com o valor FALSE, B recebe o valor de A.



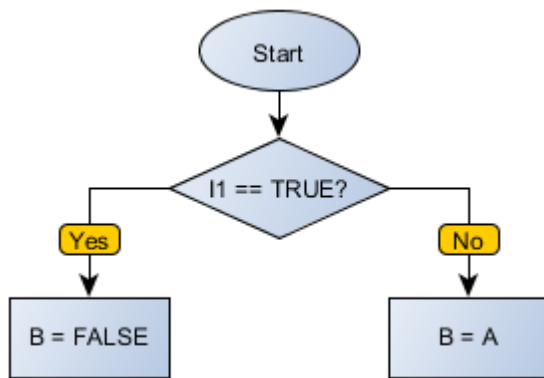
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

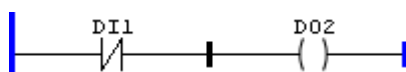
Diagrama



Fluxograma do Bloco



Exemplo



O exemplo acima realiza a transferência do oposto do valor da entrada digital DI1 para a saída digital DO2.

10.2.7.1.1.3 PTSCONTACT

Contato de transição de borda de subida.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	I1	BOOL	Entrada de controle do bloco
VAR	PTSCONTACT_INST_0	PTSCONTACT	Instância de acesso à estrutura do bloco

Funcionamento

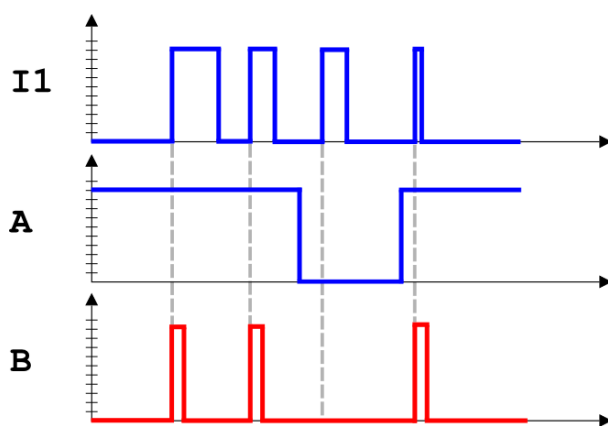
No instante em que a variável I1 transita de FALSE para TRUE (borda de subida ou borda de transição positiva), B recebe o valor de A por um ciclo de scan. Em todos os outros momentos, B recebe o valor FALSE.



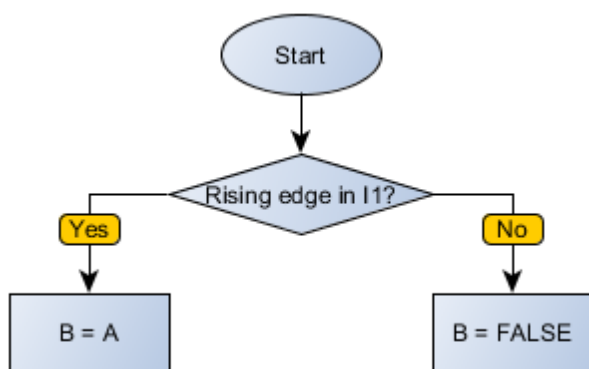
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

Diagrama



Fluxograma do Bloco



Exemplo

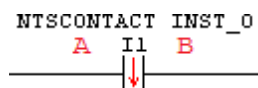


O exemplo acima reseta a saída digital DO1 caso a tecla SHIFT esteja pressionada e seja dado um pulso positivo na entrada digital DI2.

10.2.7.1.1.4 NTSCONTACT

Contato de transição de borda de descida.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	I1	BOOL	Entrada de controle do bloco
VAR	NTSCONTACT_INST_0	NTSCONTACT	Instância de acesso à estrutura do bloco

Funcionamento

No instante em que a variável I1 transita de TRUE para FALSE (borda de descida ou borda de transição negativa), B recebe o valor de A por um ciclo de scan.

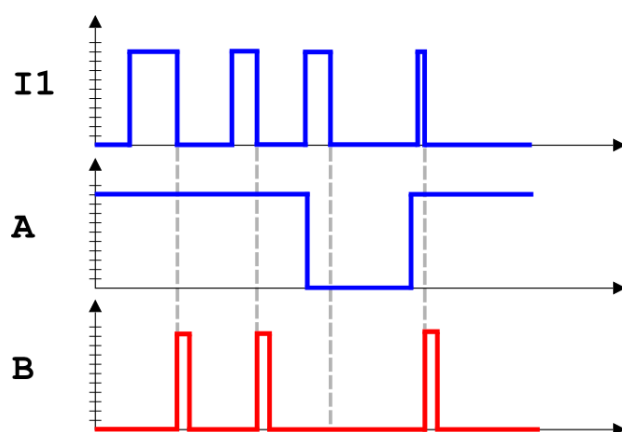
Em todos os outros momentos, B recebe o valor FALSE.



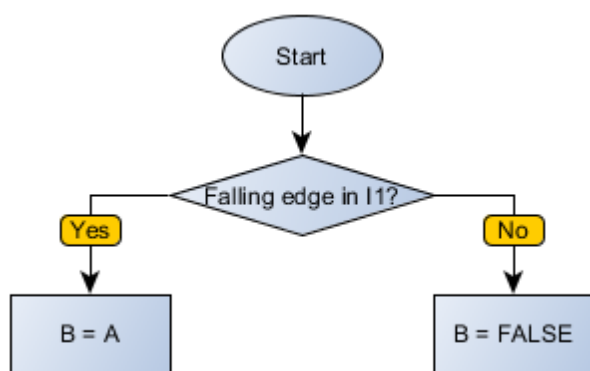
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

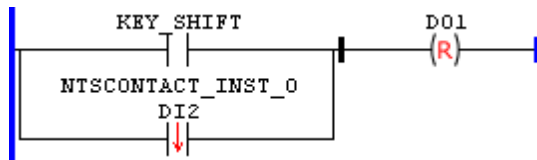
Diagrama



Fluxograma do Bloco



Exemplo



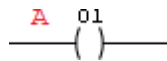
O exemplo acima reseta a saída digital DO1 caso a tecla SHIFT esteja pressionada ou seja dado um pulso negativo na entrada digital DI2.

10.2.7.1.2 Coil

10.2.7.1.2.1 DIRECTCOIL

Bloco lógico utilizado para atribuição de valores diretos a variáveis de saída.

Representação Ladder



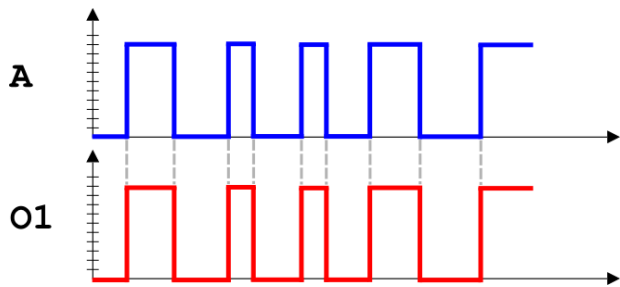
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	01	B00L	Saída de registro do bloco

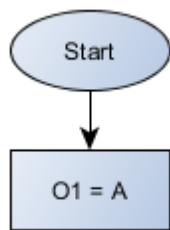
Funcionamento

Este bloco repassa o valor de A para o endereço de memória correspondente a O1.

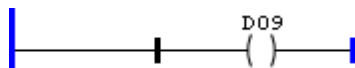
Diagrama



Fluxograma do Bloco



Exemplo

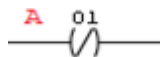


O exemplo acima deixa a saída digital DO9 permanentemente ligada, pois o valor de A, neste caso, é o valor do barramento esquerdo, que é sempre considerado nível lógico alto (TRUE).

10.2.7.1.2.2 INVERTEDCOIL

Bloco lógico utilizado para atribuição de valores negados a variáveis de saída.

Representação Ladder



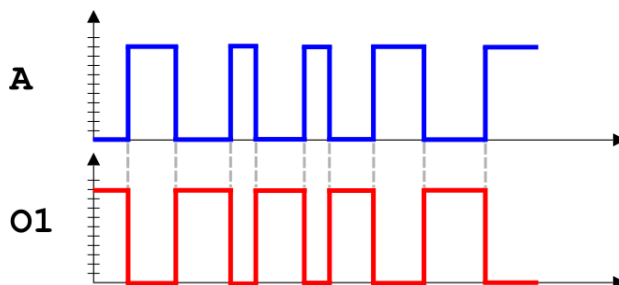
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

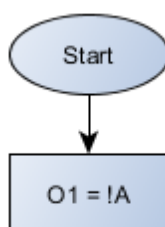
Funcionamento

Este bloco repassa o valor negado de A para o endereço de memória correspondente a O1.

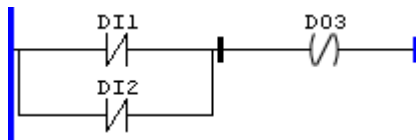
Diagrama



Fluxograma do Bloco



Exemplo

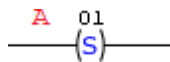


O exemplo acima desativa a saída digital DO3 quando algumas das entradas digitais DI1 ou DI2 estão com valor FALSE. Quando ambas entradas estão com valor TRUE, DO3 ativa.

10.2.7.1.2.3 SETCOIL

Bloco lógico utilizado para habilitação indefinida de variáveis de saída.

Representação Ladder



Estrutura do Bloco

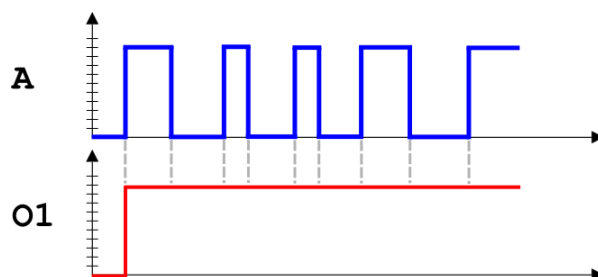
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

Funcionamento

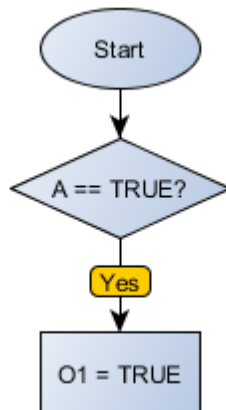
Ao identificar um valor TRUE em A, este bloco repassa um valor TRUE para o endereço de memória correspondente a O1.

Ao identificar um valor FALSE em A, este bloco não realiza nenhuma operação.

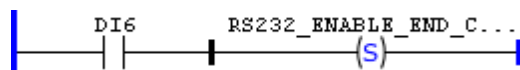
Diagrama



Fluxograma do Bloco



Exemplo

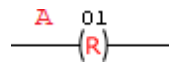


O exemplo acima ativa permanentemente o marcador de controle do sistema habilita caractere de fim de mensagem na comunicação RS232 ao identificar um nível TRUE na entrada digital DI6.

10.2.7.1.2.4 RESETCOIL

Bloco lógico utilizado para desabilitação indefinida de variáveis de saída.

Representação Ladder



Estrutura do Bloco

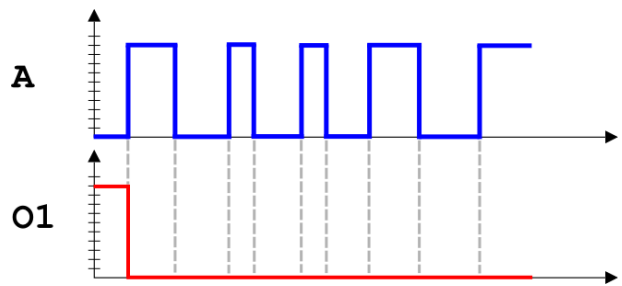
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

Funcionamento

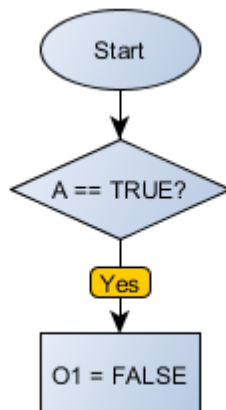
Ao identificar um valor TRUE em A, este bloco repassa um valor FALSE para o endereço de memória correspondente a O1.

Ao identificar um valor FALSE em A, este bloco não realiza nenhuma operação.

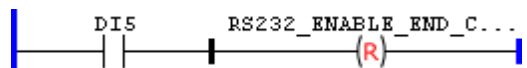
Diagrama



Fluxograma do Bloco



Exemplo

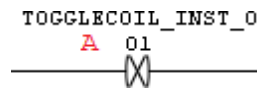


O exemplo acima desativa permanentemente o marcador de controle do sistema que habilita caractere de fim de mensagem na comunicação RS232 ao identificar um nível TRUE na entrada digital DI5.

10.2.7.1.2.5 TOGGLECOIL

Bloco lógico utilizado para alternância de variáveis de saída.

Representação Ladder



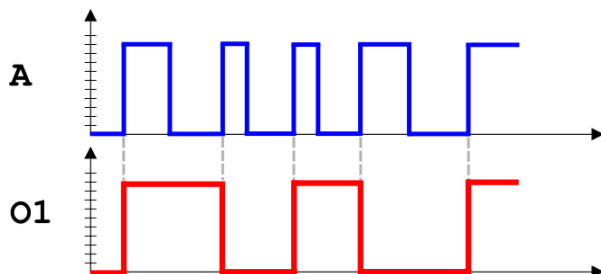
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco
VAR	TOGGLECOIL_INST_0	TOGGLECOIL	Instância de acesso à estrutura do bloco

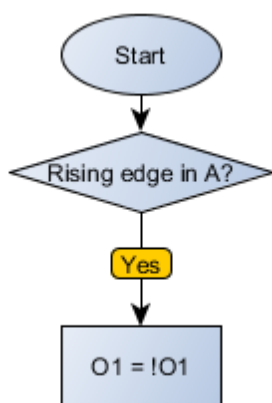
Funcionamento

Ao identificar uma transição de FALSE para TRUE (borda de subida) em A, o bloco inverte o estado de O1.

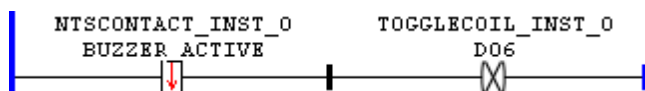
Diagrama



Fluxograma do Bloco



Exemplo

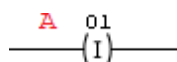


O exemplo acima inverte o estado da saída digital DO6 a cada desativação do buzzer interno.

10.2.7.1.2.6 IMMEDIATECOIL

Bloco lógico utilizado para atribuição de valores a saídas digitais padrão instantaneamente.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco

Funcionamento

Este bloco repassa o valor de A para a saída digital correspondente a O1. Diferentemente da bobina direta, este bloco não aguarda até o fim do ciclo de scan para que o valor da saída seja atualizado; isto é feito no mesmo momento em que o bloco é ativado.



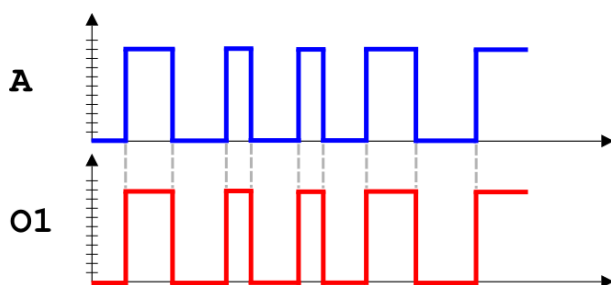
NOTA!

Este bloco apenas funciona com as saídas digitais padrão do produto.

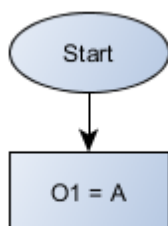
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

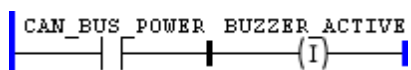
Diagrama



Fluxograma do Bloco



Exemplo



O exemplo acima ativa imediatamente o buzzer interno assim que detectar que a alimentação do barramento da rede CANopen foi interrompida, permanecendo ligado até que a alimentação seja restaurada.

10.2.7.1.3 Logic Bit

10.2.7.1.3.1 RESETBIT

Bloco lógico utilizado para realizar reset de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será alterado
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	DONE	BOOL	Sucesso da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, reseta o bit indicado em Position na variável Data, que é repassada para a saída já com o seu valor atualizado.

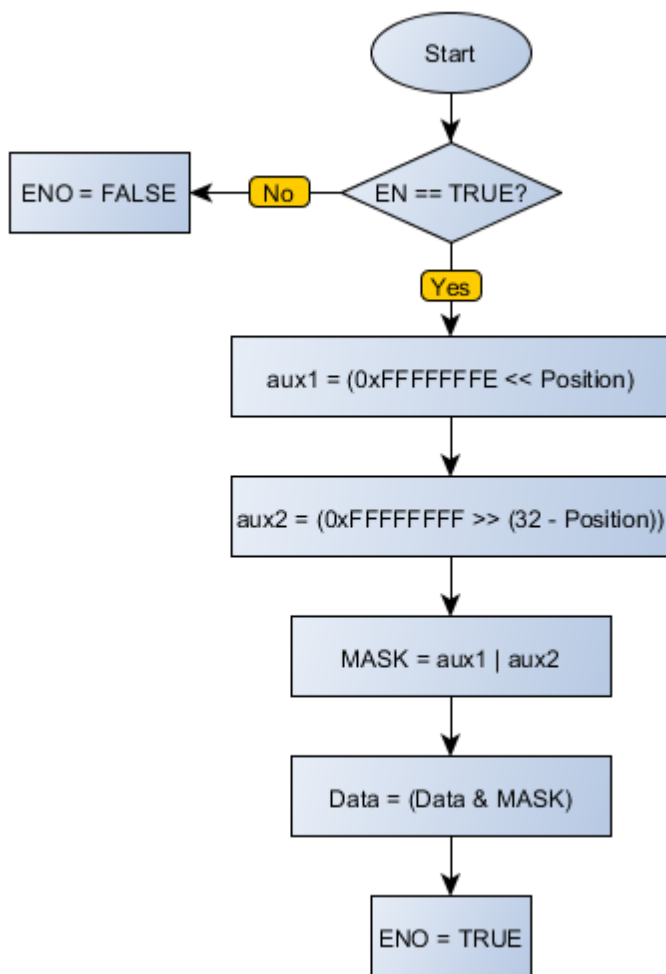
Quando EN possui valor FALSE, Data permanece inalterado.

A variável DONE recebe o mesmo valor de EN, exceto quando há erro no reset do bit, recebendo então um valor FALSE.

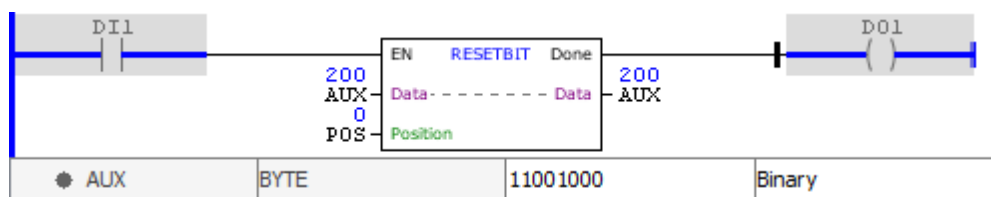
**NOTA!**

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

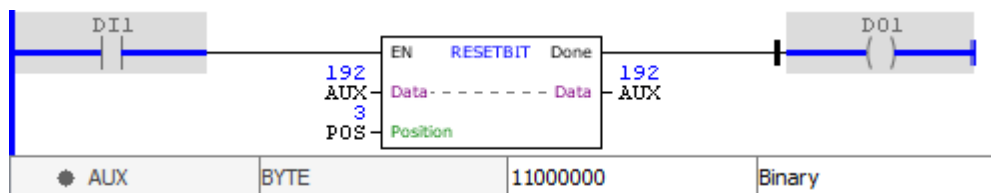
Fluxograma do Bloco



Exemplo

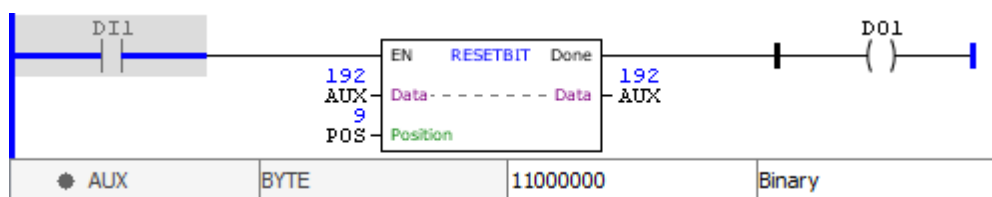


O exemplo acima reseta o bit da posição zero de AUX, cujo valor inicial é 200 (1100 1000, em binário). Como este bit já possuía valor FALSE, nada foi alterado.



O exemplo acima reseta o bit da posição três de AUX, alterando seu valor binário e,

consequentemente, sua representação decimal.

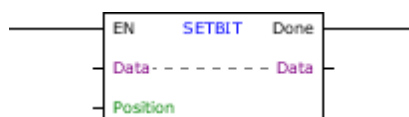


O exemplo acima reseta o bit da posição nove de AUX. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída não é habilitada.

10.2.7.1.3.2 SETBIT

Bloco lógico utilizado para realizar set de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será alterado
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	DONE	BOOL	Sucesso da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, seta o bit indicado em Position na variável Data, que é repassada para a saída já com o seu valor atualizado.

Quando EN possui valor FALSE, Data permanece inalterado.

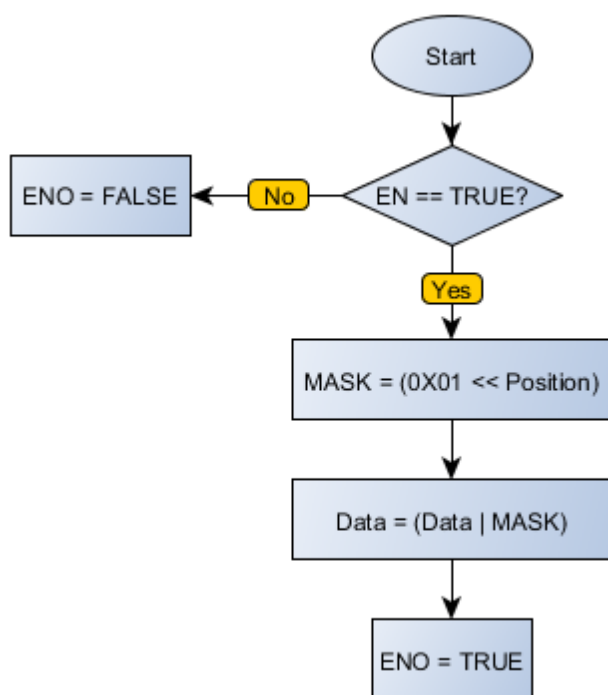
A variável DONE recebe o mesmo valor de EN, exceto quando há erro no set do bit, recebendo então um valor FALSE.



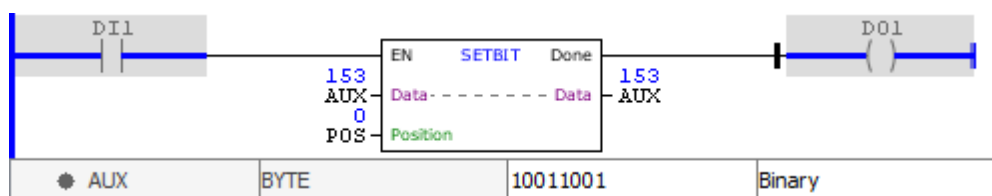
NOTA!

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

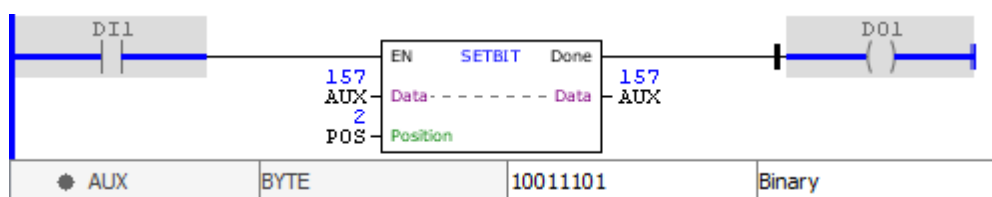
Fluxograma do Bloco



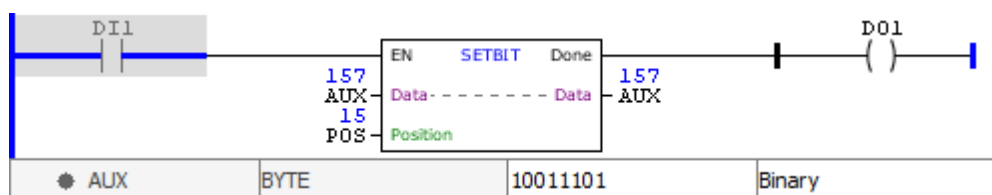
Exemplo



O exemplo acima seta o bit da posição zero de AUX, cujo valor inicial é 153 (1001 1001, em binário). Como este bit já possuía valor TRUE, nada foi alterado.



O exemplo acima seta o bit da posição dois de AUX, alterando seu valor binário e, consequentemente, sua representação decimal.



O exemplo acima seta o bit da posição quinze de AUX. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída não é habilitada.

10.2.7.1.3.3 TESTBIT

Bloco lógico que retorna o valor de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será testado
	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	Q	BOOL	Valor do bit testado

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor do bit indicado em Position na variável Data.

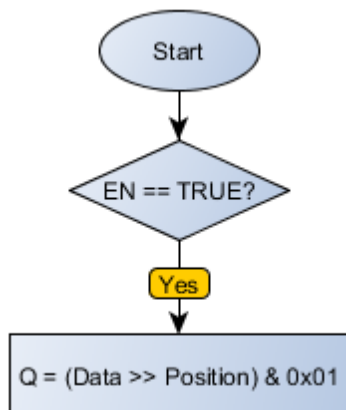
Quando EN possui valor FALSE, Q também recebe FALSE.



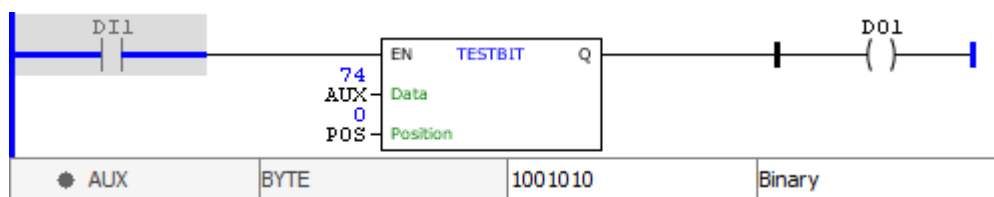
NOTA!

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

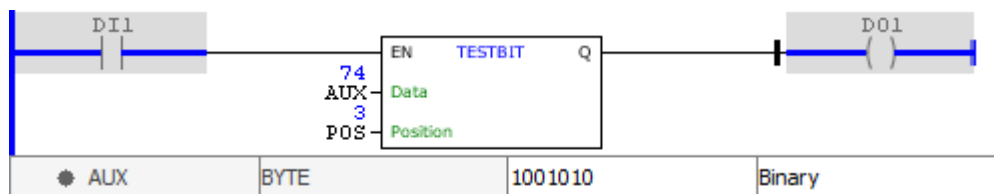
Fluxograma do Bloco



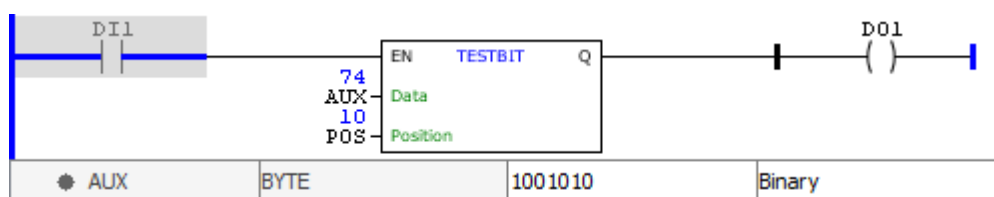
Exemplo



O exemplo acima joga o valor do bit da posição zero de AUX, cujo valor inicial é 74 (0100 1010 em binário) para a saída Q. Como este bit tem valor 0, a saída fica desabilitada.



O exemplo acima joga o valor do bit da posição três de AUX para a saída Q. Como este bit tem valor 1, a saída fica habilitada.



O exemplo acima joga o valor do bit da posição dez de AUX para a saída Q. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída fica desabilitada.

10.2.7.1.4 Logic Boolean

10.2.7.1.4.1 NOT

Bloco lógico que executa uma operação de negado booleano em uma variável, armazenando o resultado em outra.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável de referência para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

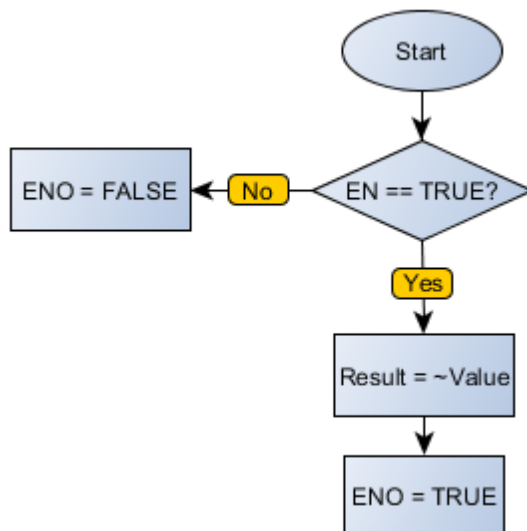
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor booleano negado da variável de entrada Value.

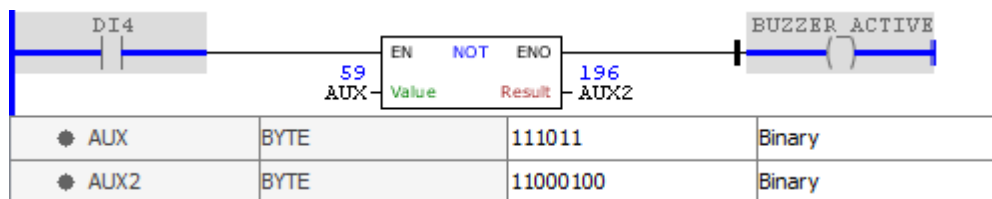
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação negado booleano em AUX, armazenando o resultado em AUX2.

10.2.7.1.4.2 AND

Bloco lógico que executa uma operação de “e” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

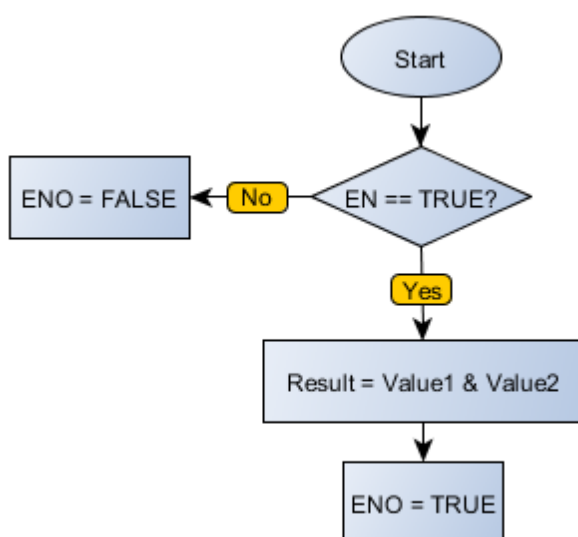
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “e” das variáveis de entrada Value1 e Value2.

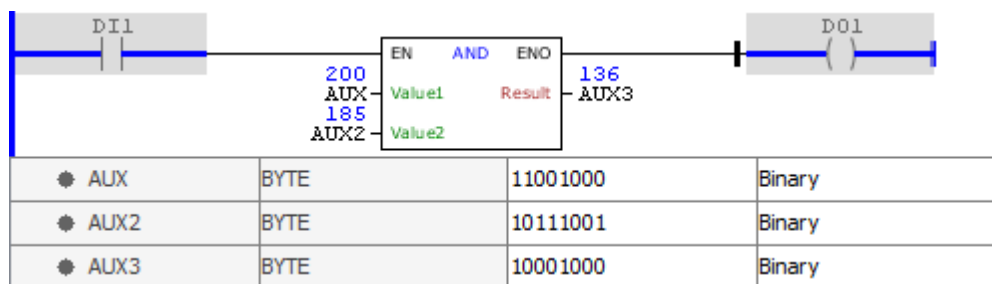
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação e booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.2.7.1.4.3 OR

Bloco lógico que executa uma operação de “ou” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

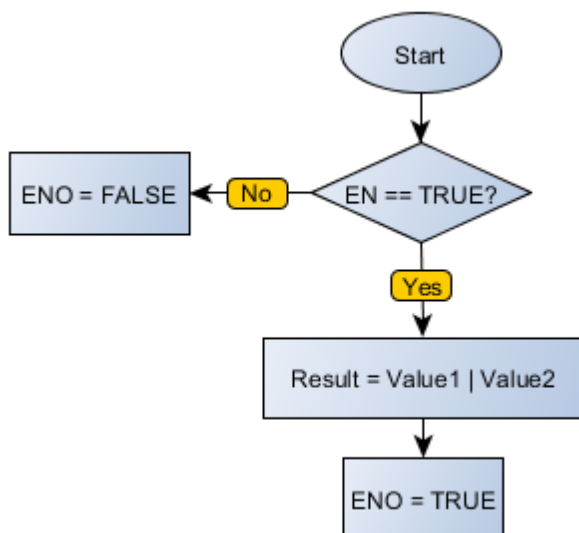
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou” das variáveis de entrada Value1 e Value2.

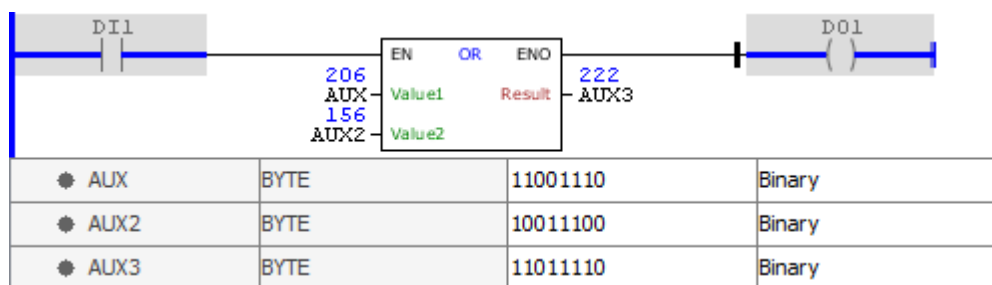
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação ou booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.2.7.1.4.4 XOR

Bloco lógico que executa uma operação de “ou exclusivo” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

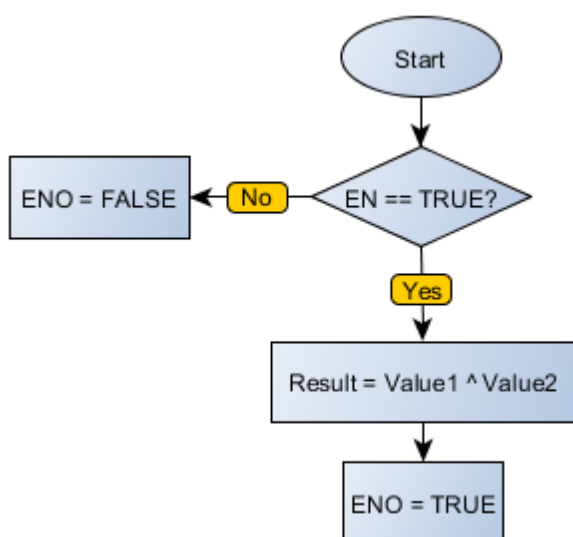
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou exclusivo” das variáveis de entrada Value1 e Value2.

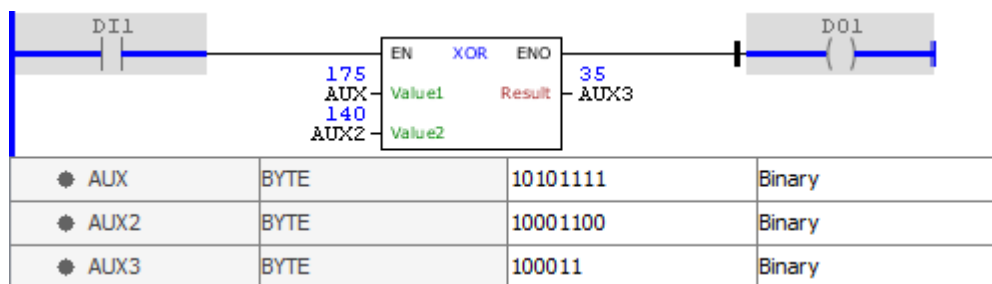
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

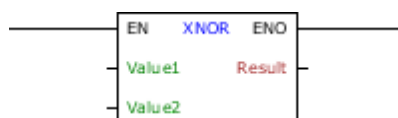


O exemplo acima realiza a operação ou exclusivo booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.2.7.1.4.5 XNOR

Bloco lógico que executa uma operação de “ou exclusivo negado” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

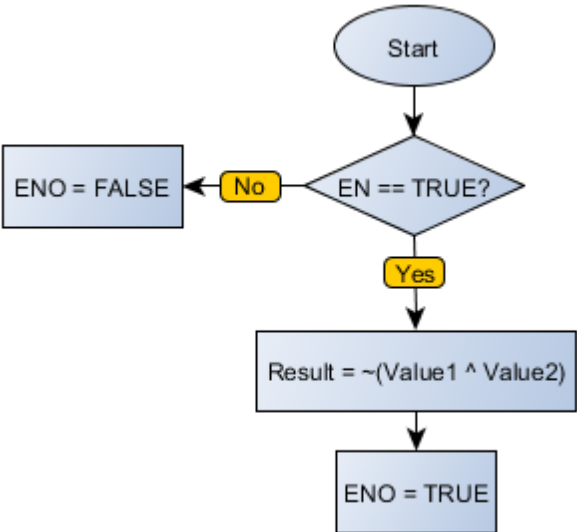
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou exclusivo negado” das variáveis de entrada Value1 e Value2.

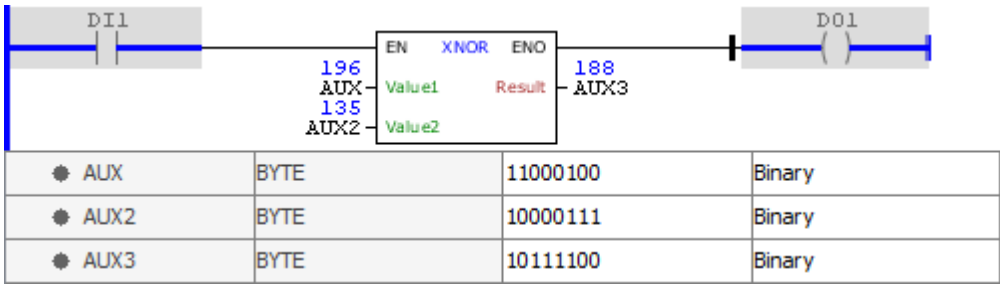
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação ou exclusivo negado booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.2.7.1.5 Logic Shift

10.2.7.1.5.1 ASHL

Bloco que executa uma operação de deslocamento binário à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	SINT INT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	SINT INT DINT	Variável que armazena o resultado da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos aritméticos à esquerda, de acordo com o valor de Shift.



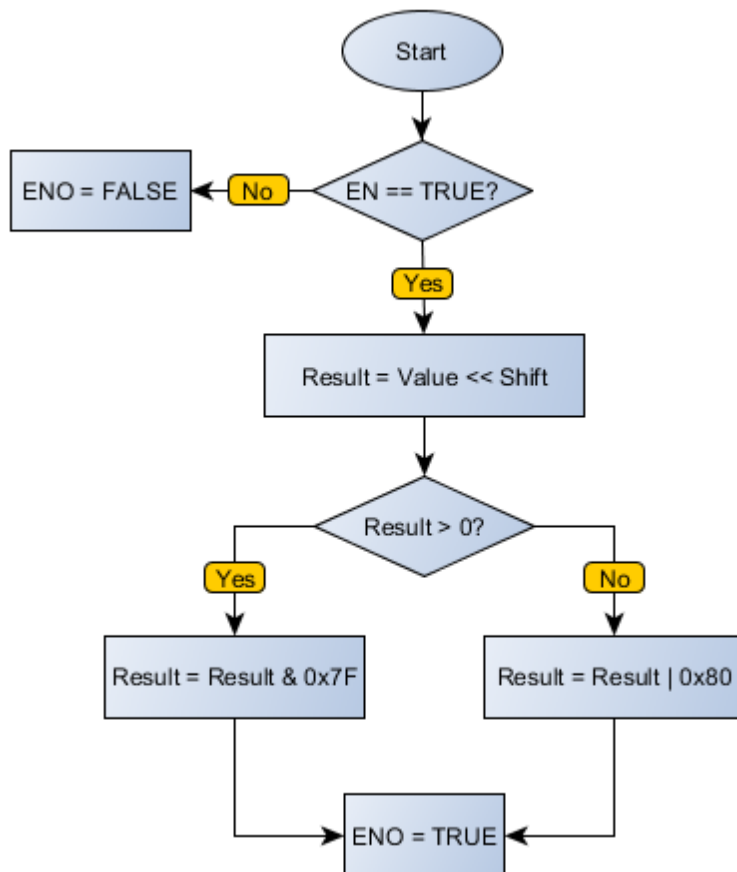
NOTA!

Todos os deslocamentos aritméticos implementados mantêm o sinal da variável

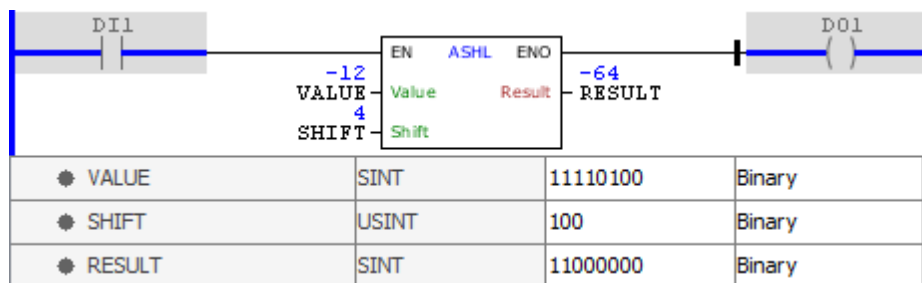
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

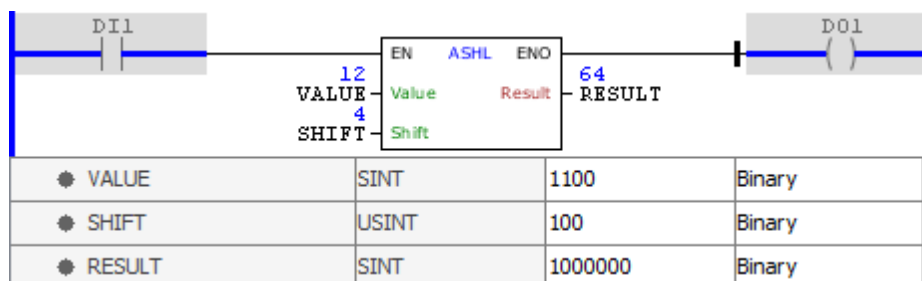
Fluxograma do Bloco



Exemplo



Descrição do exemplo.



Descrição do exemplo.

10.2.7.1.5.2 ASHR

Bloco que executa uma operação de deslocamento aritmético à direita em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	SINT INT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	SINT INT DINT	Variável que armazena o resultado da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos aritméticos à direita, de acordo com o valor de Shift.



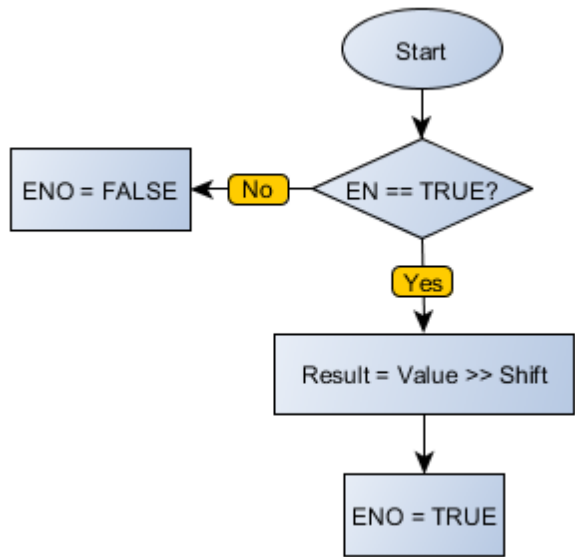
NOTA!

Todos os deslocamentos aritméticos implementados mantém o sinal da variável

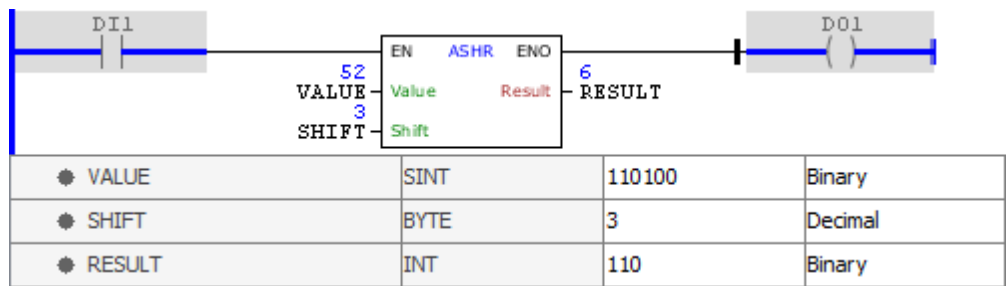
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

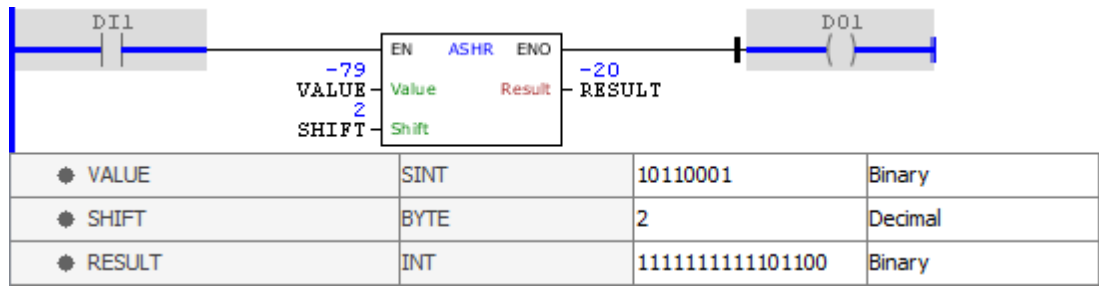
Fluxograma do Bloco



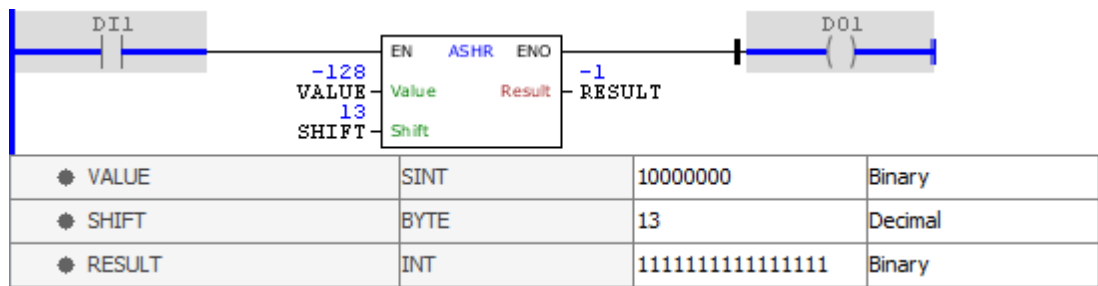
Exemplo



O exemplo acima realiza um deslocamento aritmético à direita de três posições na variável VALUE, cujo valor inicial é 52 (0011 0100 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0000 0110 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento aritmético à direita de duas posições na variável VALUE, cujo valor inicial é -79 (1011 0001 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos uns, visto que os deslocamentos aritméticos à direita preservam o sinal da variável. O resultado final (1111 0110 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento aritmético à direita de treze posições na variável VALUE, cujo valor inicial é -128 (1000 0000 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos uns. O resultado final (1111 1111 em binário) é armazenado em RESULT.

10.2.7.1.5.3 SHL

Bloco que executa uma operação de deslocamento lógico binário à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

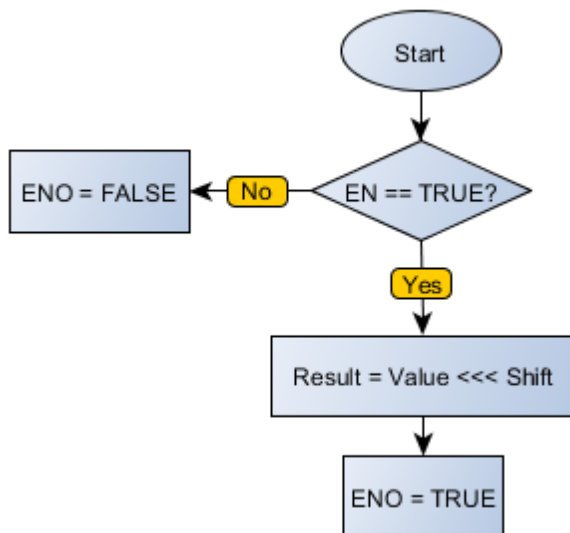
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à esquerda, de acordo com o valor de Shift.

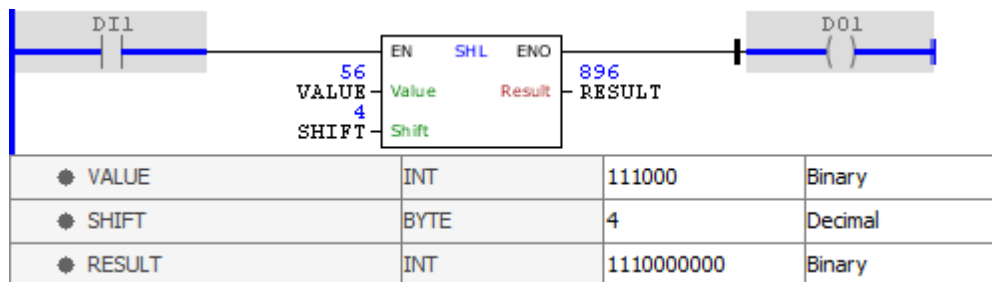
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

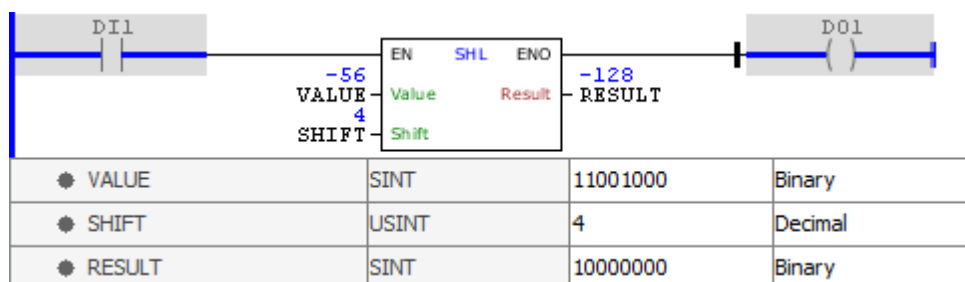
Fluxograma do Bloco



Exemplo



O exemplo acima realiza um deslocamento lógico à esquerda de quatro posições na variável VALUE, cujo valor inicial é 56 (0011 1000 em binário). Os bits à esquerda vão sendo descartados, e à direita são inseridos novos zeros. O resultado final (0011 1000 0000 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento lógico à esquerda de quatro posições na variável VALUE, cujo valor inicial é -56 (1100 1000 em binário). Os bits à esquerda vão sendo descartados, e à direita são inseridos novos zeros. O resultado final (1100 1000 0000 em binário) é armazenado em RESULT. Como RESULT é do tipo SINT, ela só aceita os primeiros oito bits (1000 0000).

10.2.7.1.5.4 SHR

Bloco que executa uma operação de deslocamento lógico binário à direita em um valor passado por

Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

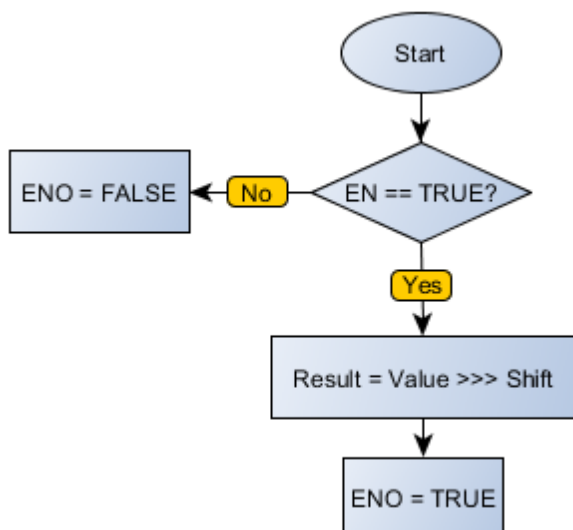
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à direita, de acordo com o valor de Shift.

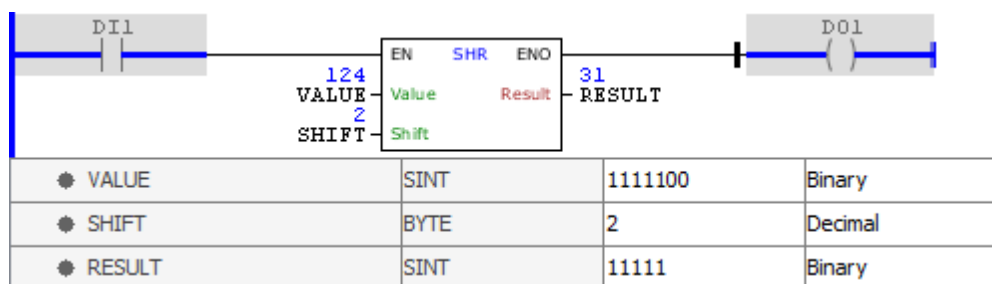
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

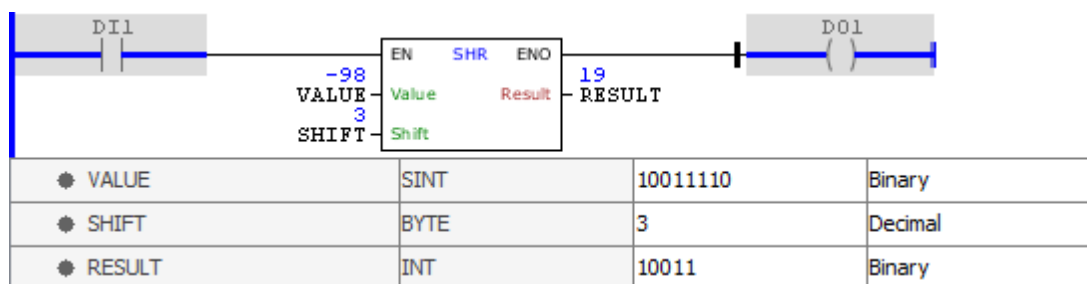
Fluxograma do Bloco



Exemplo



O exemplo acima realiza um deslocamento lógico à direita de duas posições na variável VALUE, cujo valor inicial é 124 (0111 1100 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0001 1111 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento lógico à direita de três posições na variável VALUE, cujo valor inicial é -98 (1001 1110 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0001 0011 em binário) é armazenado em RESULT.

10.2.7.1.6 Logic Rotate

10.2.7.1.6.1 ROL

Bloco que executa uma operação de rotação lógica à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer rotação
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

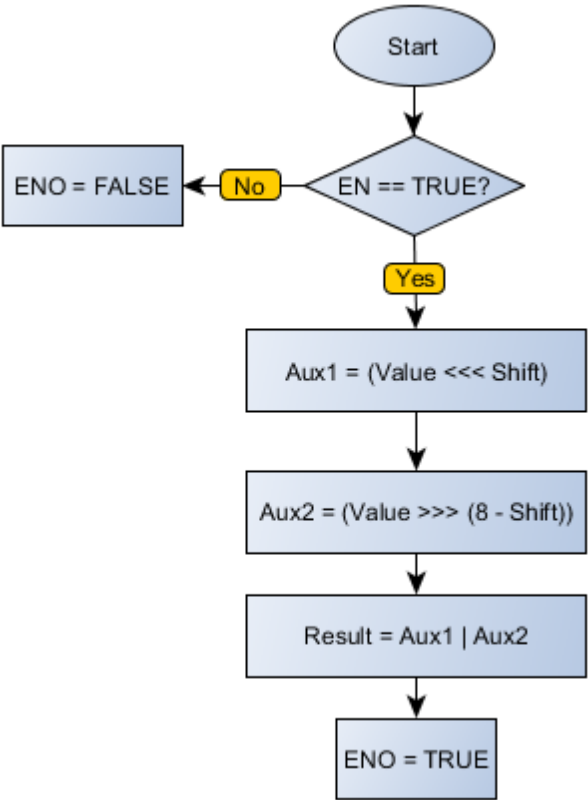
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à esquerda, de acordo com o valor de Shift. Os bits mais significativos que forem sendo descartados são reenviados aos bits menos significativos, caracterizando a rotação.

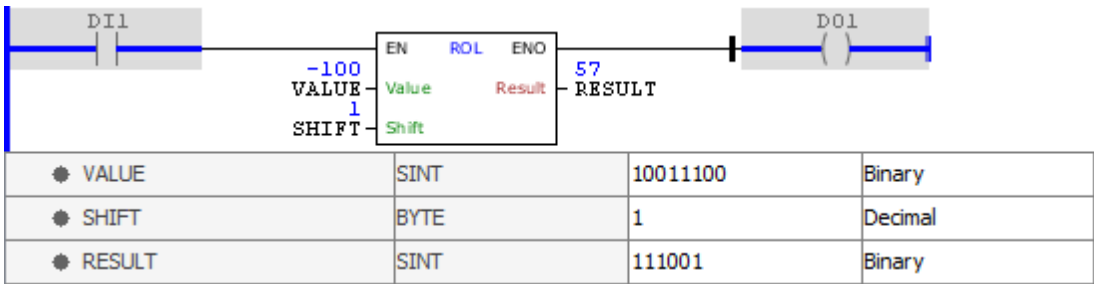
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

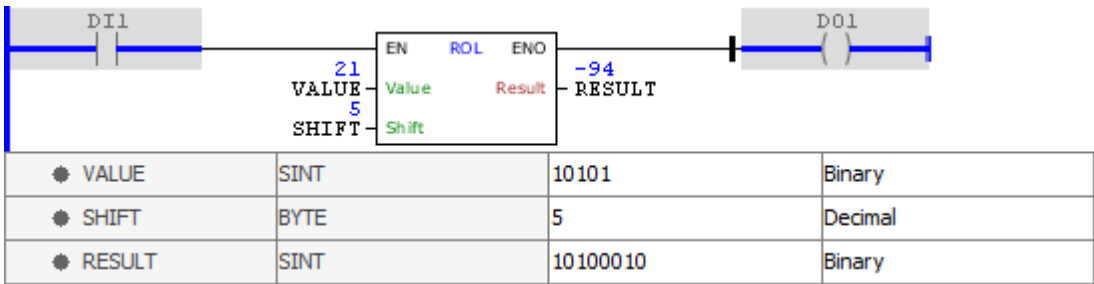
Fluxograma do Bloco



Exemplo



O exemplo acima realiza uma rotação lógica à esquerda de uma posição na variável VALUE, cujo valor inicial é -100 (1001 1100 em binário). Os bits descartados à esquerda são reinseridos à direita. O resultado final (0011 1001 em binário) é armazenado em RESULT.



O exemplo acima realiza uma rotação lógica à esquerda de cinco posições na variável VALUE, cujo valor inicial é 21 (0001 0101 em binário). Os bits descartados à esquerda são reinseridos à direita. O resultado final (1010 0010 em binário) é armazenado em RESULT.

10.2.7.1.6.2 ROR

Bloco que executa uma operação de rotação lógica à direita em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer rotação
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

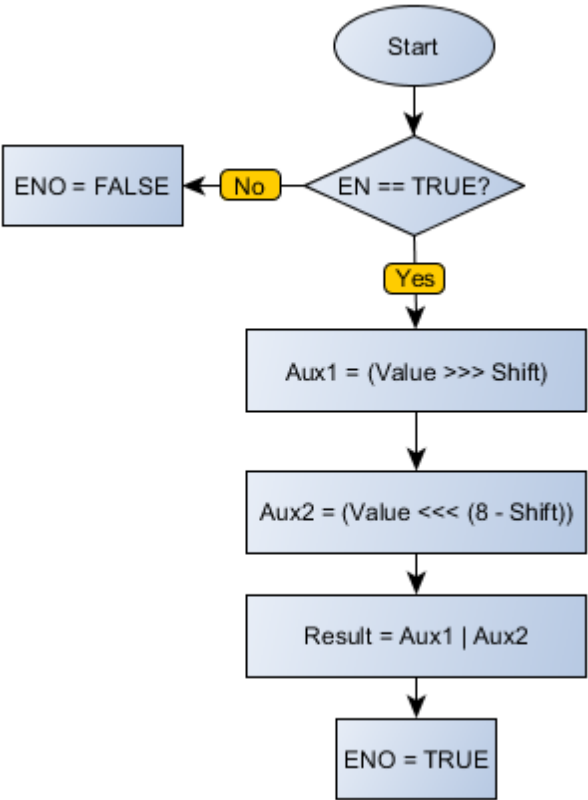
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à direita, de acordo com o valor de Shift. Os bits menos significativos que forem sendo descartados são reenviados aos bits mais significativos, caracterizando a rotação.

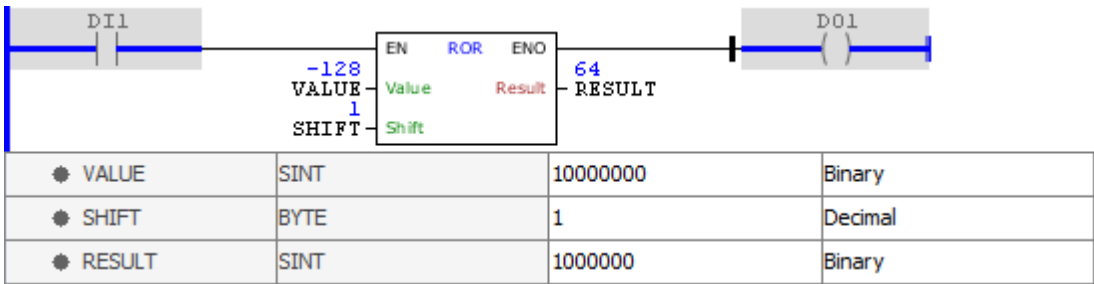
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

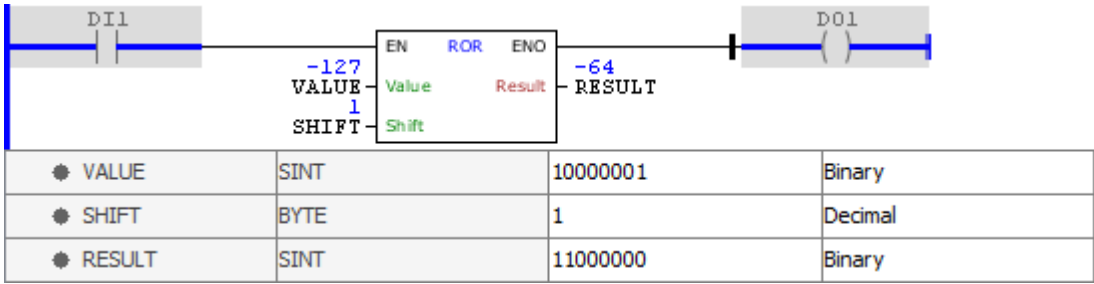
Fluxograma do Bloco



Exemplo



O exemplo acima realiza uma rotação lógica à direita de uma posição na variável VALUE, cujo valor inicial é -128 (1000 0000 em binário). Os bits descartados à direita são reinseridos à esquerda. O resultado final (0100 0000 em binário) é armazenado em RESULT. Observe que o sinal não é preservado nessa operação.



O exemplo acima realiza uma rotação lógica à direita de uma posição na variável VALUE, cujo valor inicial é -127 (1000 0001 em binário). Os bits descartados à direita são reinseridos à esquerda. O resultado final (1100 0000 em binário) é armazenado em RESULT.

10.2.7.1.7 Math Basic

10.2.7.1.7.1 ABS

Bloco que calcula o módulo de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

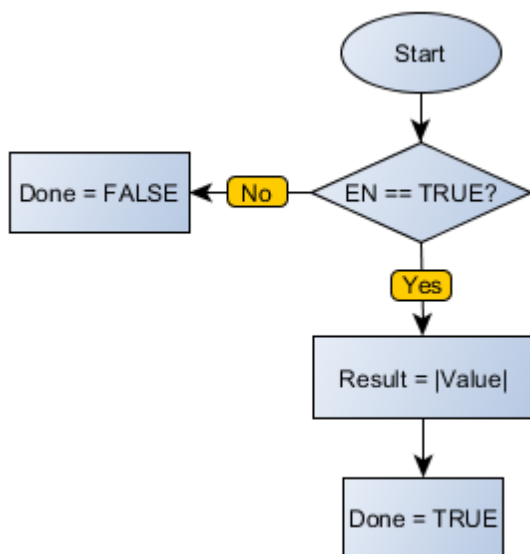
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

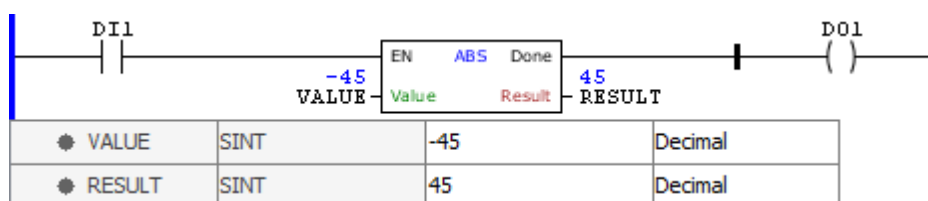
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor absoluto da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

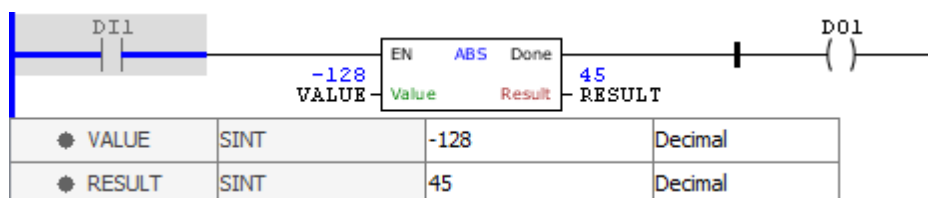
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o valor absoluto da variável VALUE, cujo valor inicial é -45, armazenando o resultado final, 45, em RESULT.

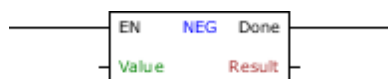


O exemplo acima calcula o valor absoluto da variável VALUE, cujo valor inicial é -45. O resultado final, 128, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.2.7.1.7.2 NEG

Bloco que calcula o oposto (ou seja, o produto com -1) de um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

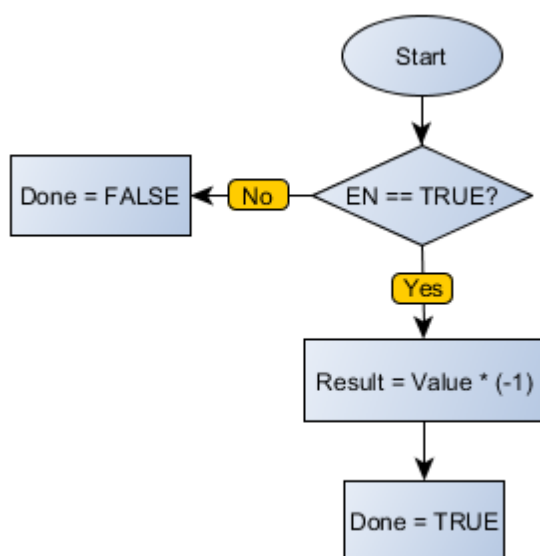
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

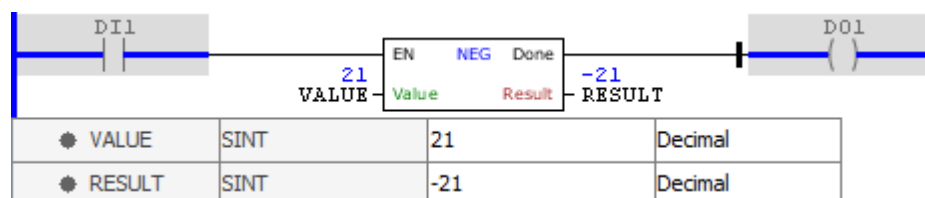
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o oposto da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

Fluxograma do Bloco

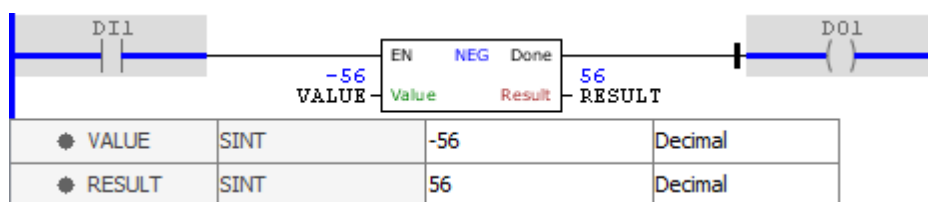


Exemplo

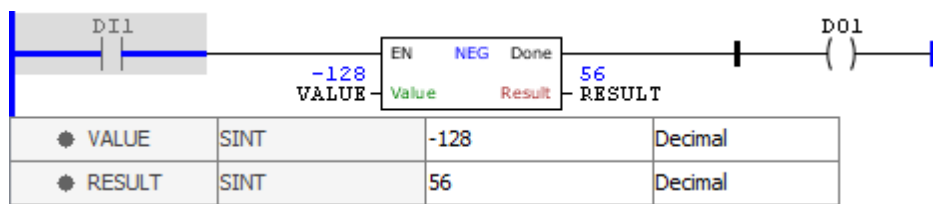


O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é 21, armazenando o

resultado final, -21, em RESULT.



O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é -56, armazenando o resultado final, 56, em RESULT.



O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é -128. O resultado final, 128, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.2.7.1.7.3 ADD

Bloco que calcula a soma dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeira parcela da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segunda parcela da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

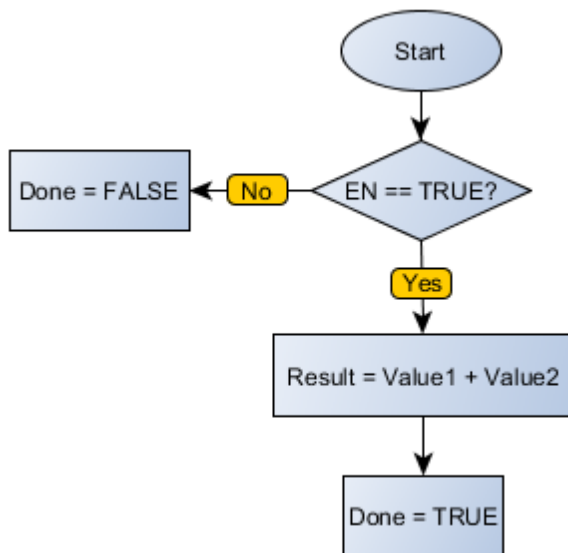
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a soma das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor

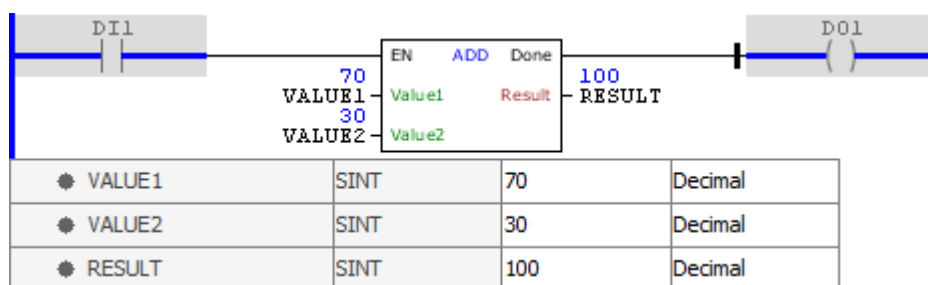
inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

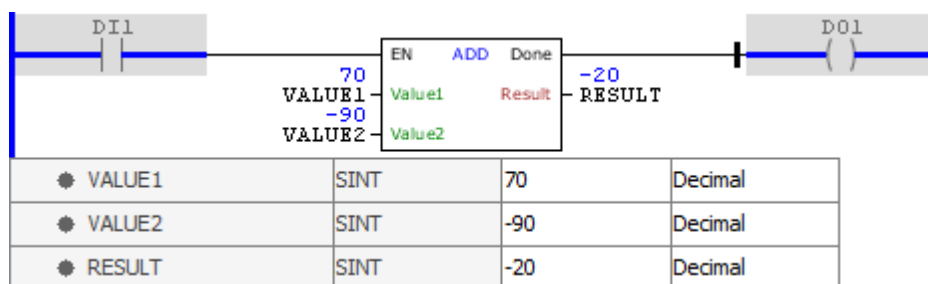
Fluxograma do Bloco



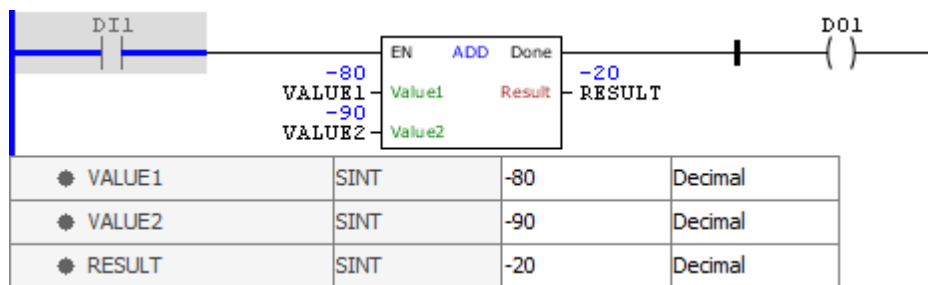
Exemplo



O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.

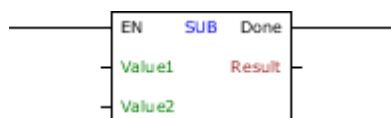


O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2. O resultado final, -170, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.2.7.1.7.4 SUB

Bloco que calcula a subtração entre os valores Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

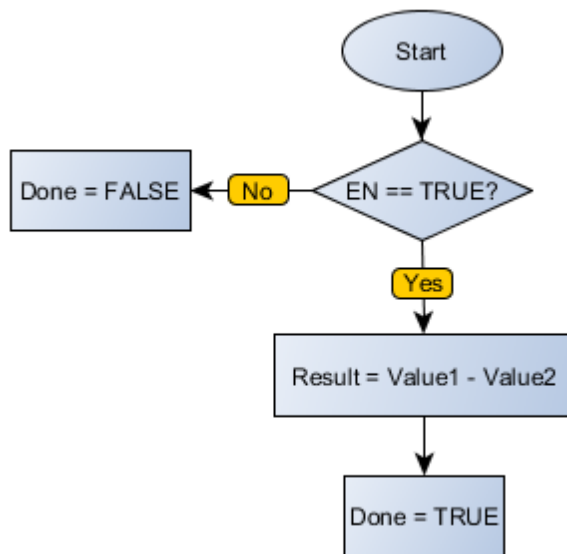
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Minuendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Subtraendo da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

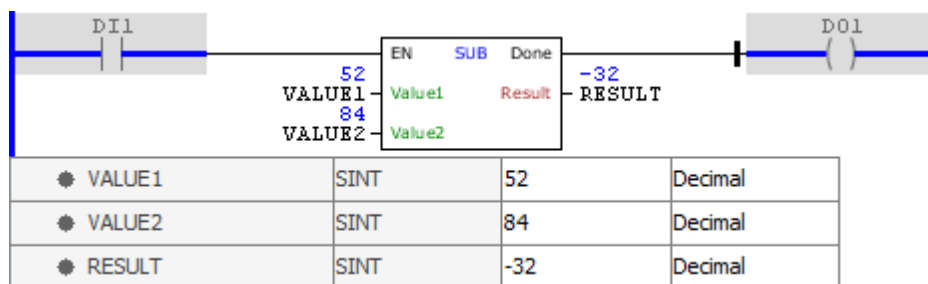
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a subtração das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

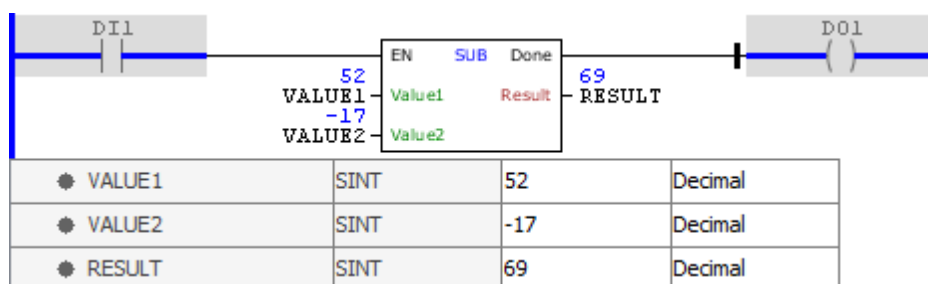
Fluxograma do Bloco



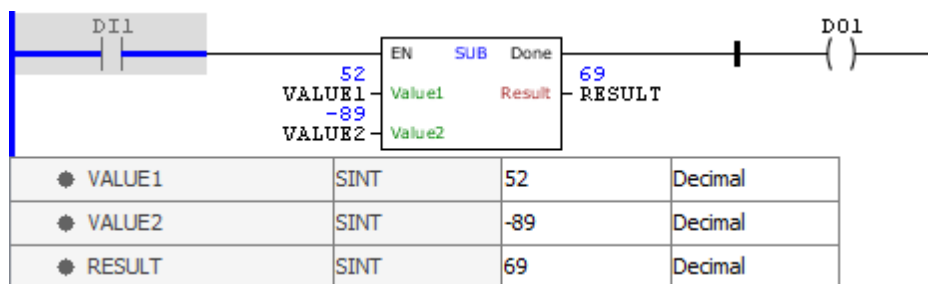
Exemplo



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2. O resultado final, 141, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.2.7.1.7.5 MUL

Bloco que calcula a multiplicação dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

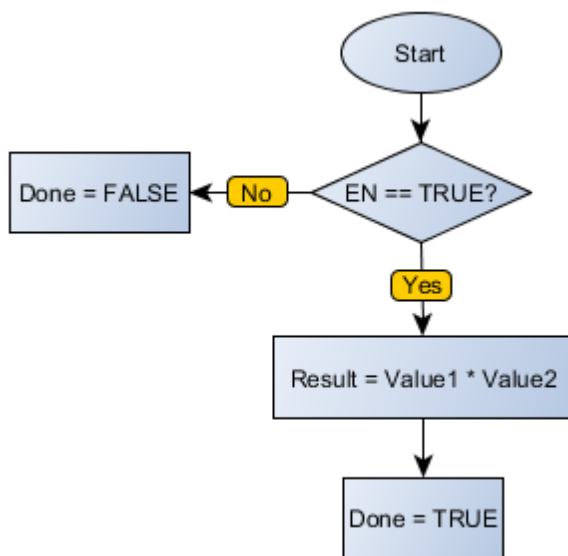
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro fator da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo fator da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

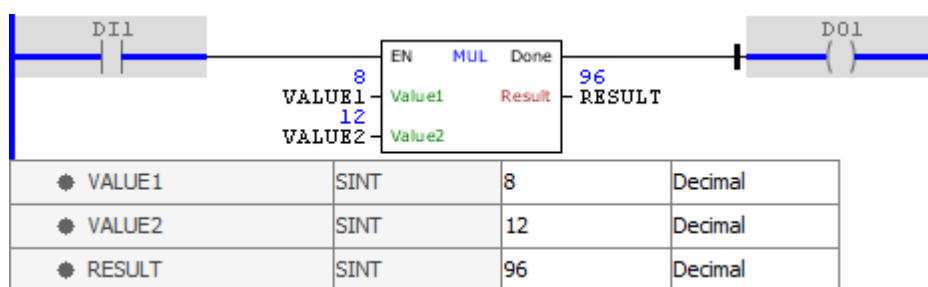
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a multiplicação das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

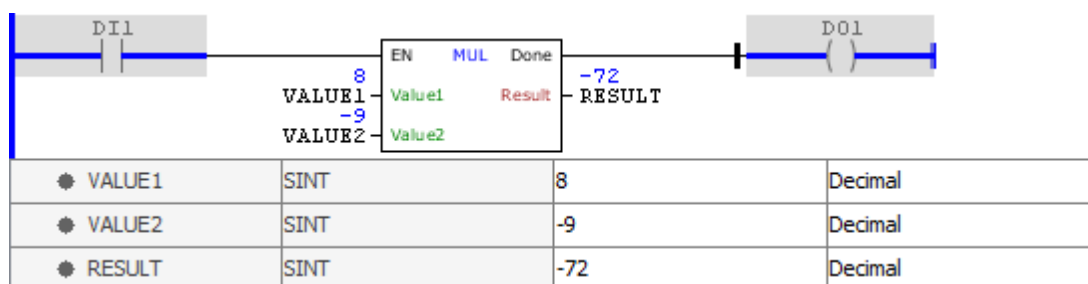
Fluxograma do Bloco



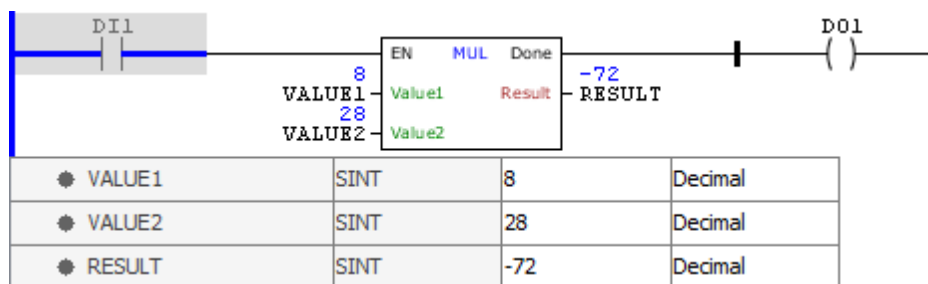
Exemplo



O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.

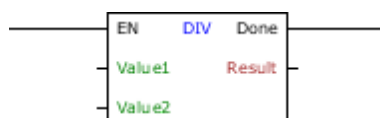


O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2. O resultado final, 224, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.2.7.1.7.6 DIV

Bloco que calcula a divisão dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

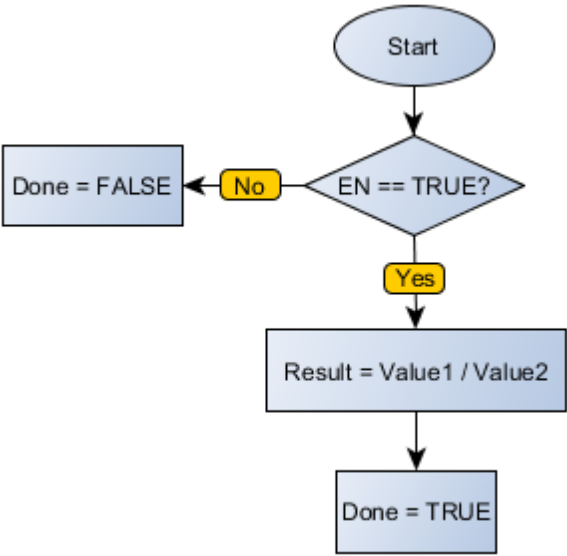
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Dividendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Divisor da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

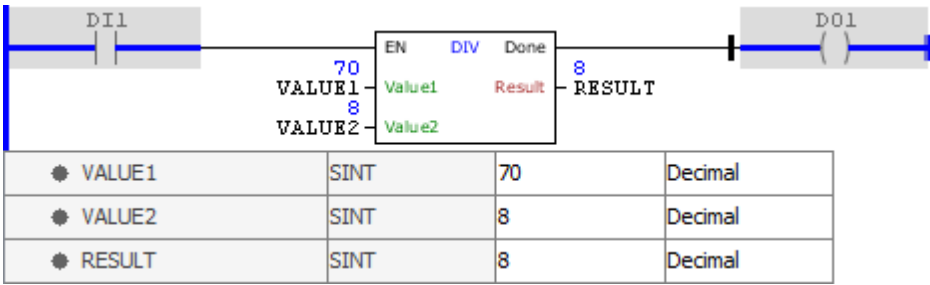
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a divisão das variáveis Value1 e Value2. O valor armazenado será a divisão exata caso Result seja REAL, ou, nos demais casos, apenas o quociente da divisão. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

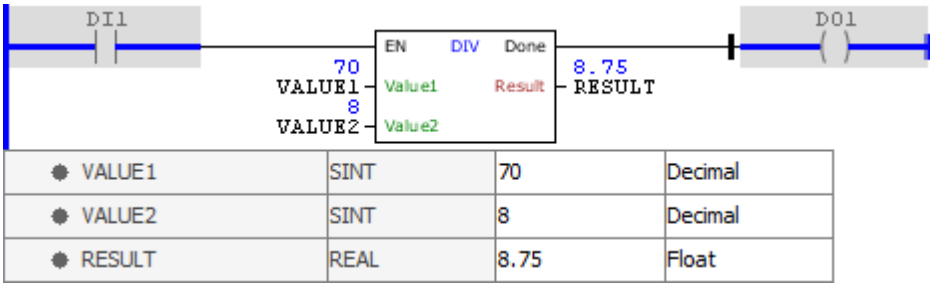
Fluxograma do Bloco



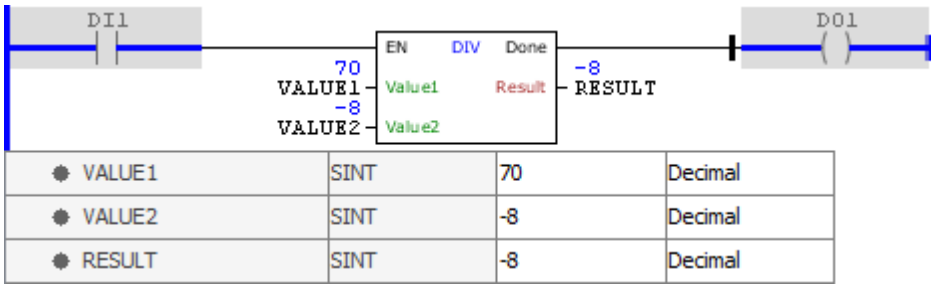
Exemplo



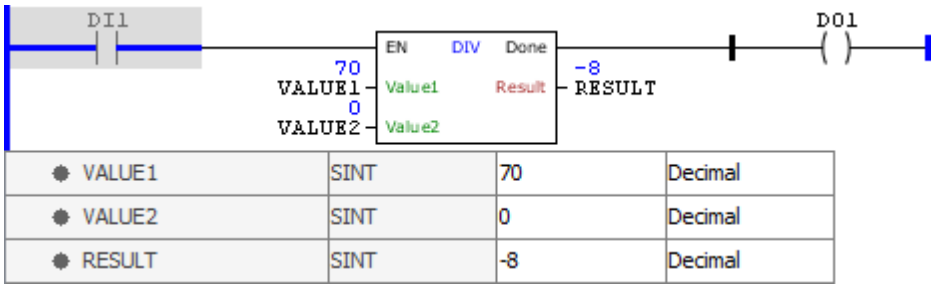
O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo SINT, apenas o quociente da divisão é armazenado nela.



O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo REAL, o valor exato da divisão é armazenado nela.



O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo SINT, apenas o quociente da divisão é armazenado nela. Observe que o bloco aceita argumentos de ambos os sinais.

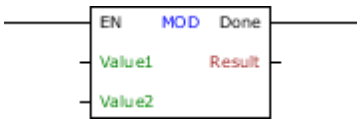


O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como VALUE2 é zero, o bloco gera um erro de execução, RESULT permanece inalterado e a saída é desabilitada.

10.2.7.1.7.7 MOD

Bloco que calcula o resto da divisão dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

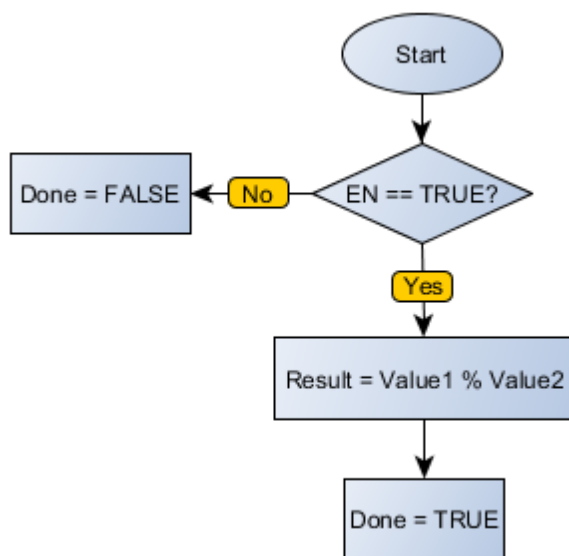
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Dividendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Divisor da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

Funcionamento

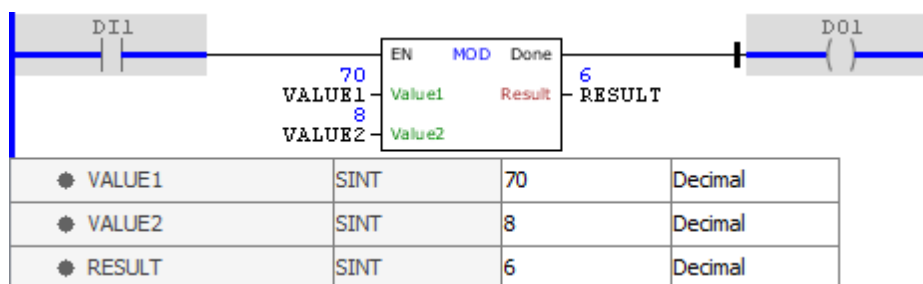
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o resto da divisão das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

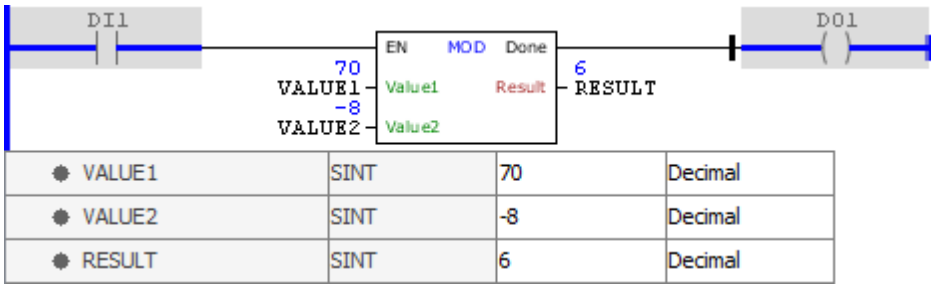
Fluxograma do Bloco



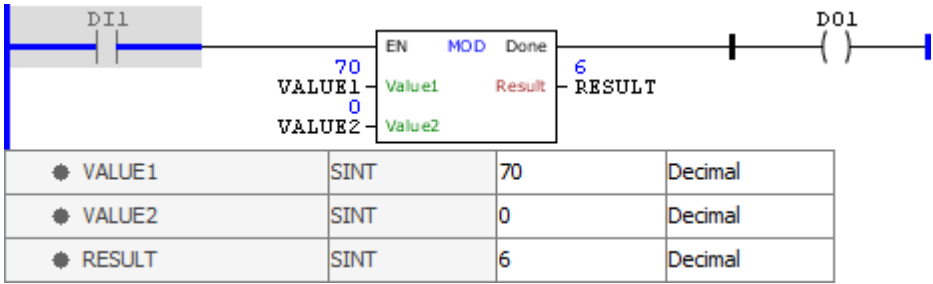
Exemplo



O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.



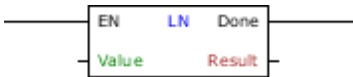
O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como VALUE2 é zero, o bloco gera um erro de execução, RESULT permanece inalterado e a saída é desabilitada.

10.2.7.1.8 Math Extended

10.2.7.1.8.1 LN

Bloco que calcula o logaritmo natural do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

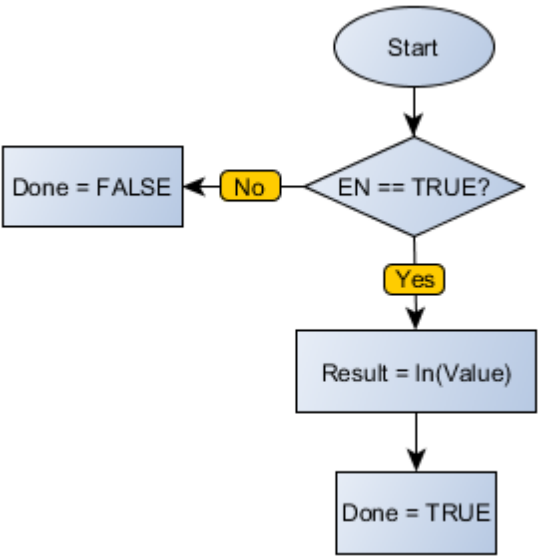
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o logaritmo natural da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor

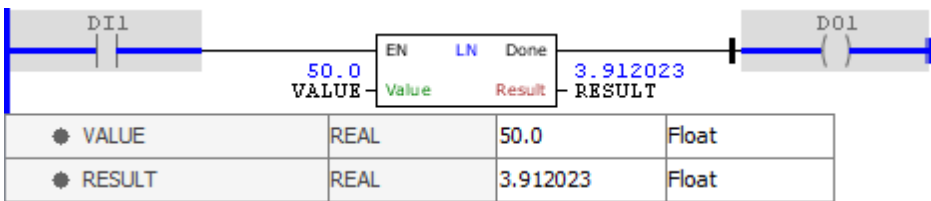
inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

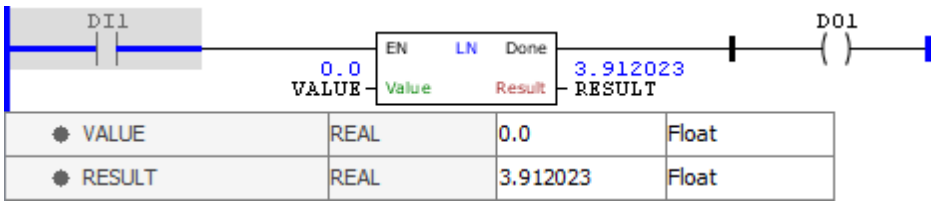
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o logaritmo natural da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



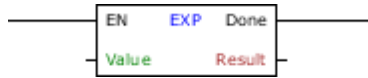
O exemplo acima calcula o logaritmo natural da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor zero, e a saída Done é desativada.

10.2.7.1.8.2 EXP

Bloco que calcula a exponencial do número de Euler “e” elevado ao valor de Value, armazenando o

resultado em Result.

Representação Ladder



Estrutura do Bloco

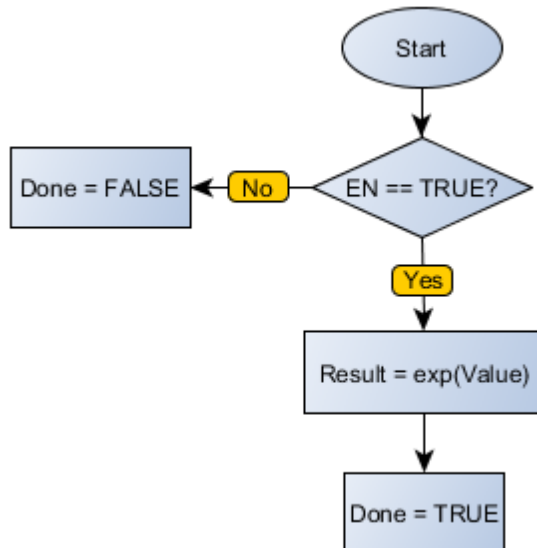
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

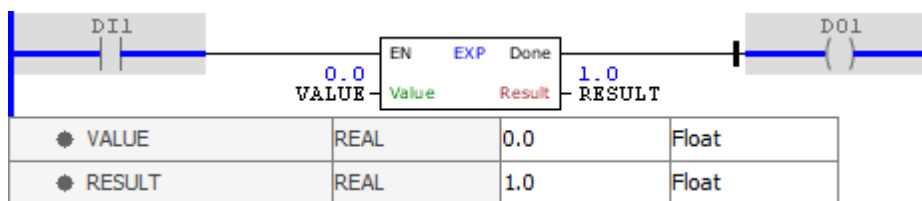
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a exponencial do número de Euler “e” elevado à variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

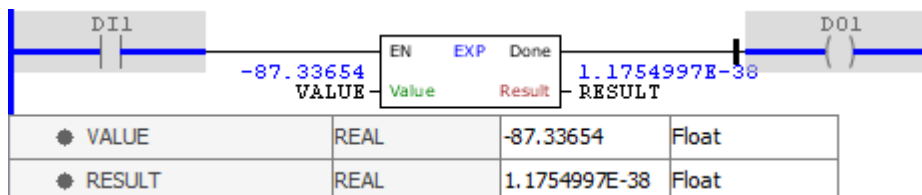
Fluxograma do Bloco



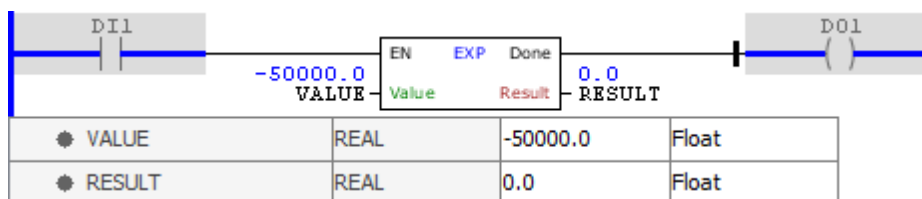
Exemplo



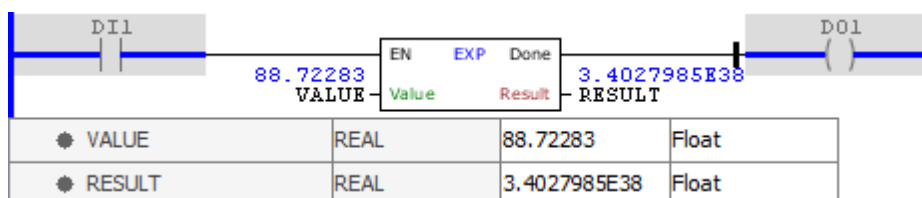
O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



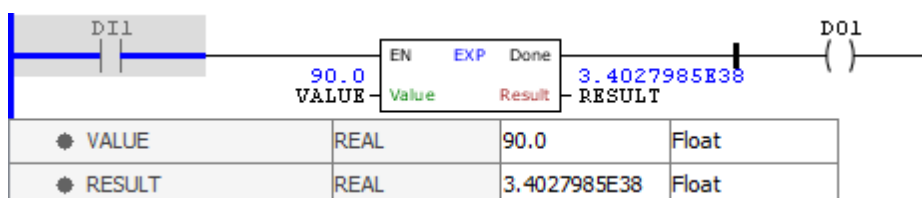
O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor mínimo de entrada para o qual o bloco retorna um resultado não nulo. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. Valores inferiores ao mínimo fazem com que o bloco retorne valor nulo. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor máximo de entrada para o qual o bloco retorna um resultado válido. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. Valores superiores ao máximo fazem com que o bloco gere um erro, a saída RESULT permanece inalterada e a saída Done seja desativada.

10.2.7.1.8.3 LOG10

Bloco que calcula o logaritmo comum (base 10) do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

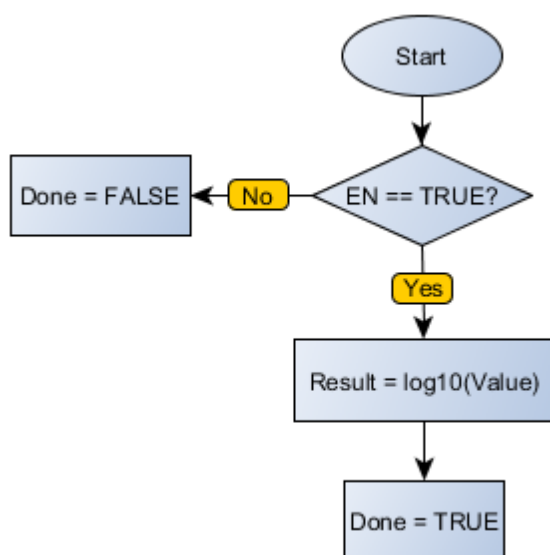
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

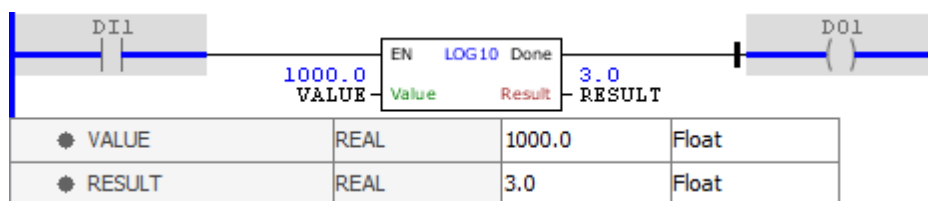
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o logaritmo comum da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

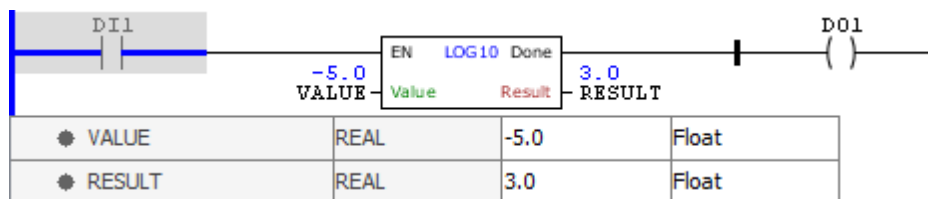
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o logaritmo comum da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o logaritmo comum da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor negativo, e a saída Done é desativada.

10.2.7.1.8.4 ALOG10

Bloco que calcula o antilogaritmo (exponencial com base 10) do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

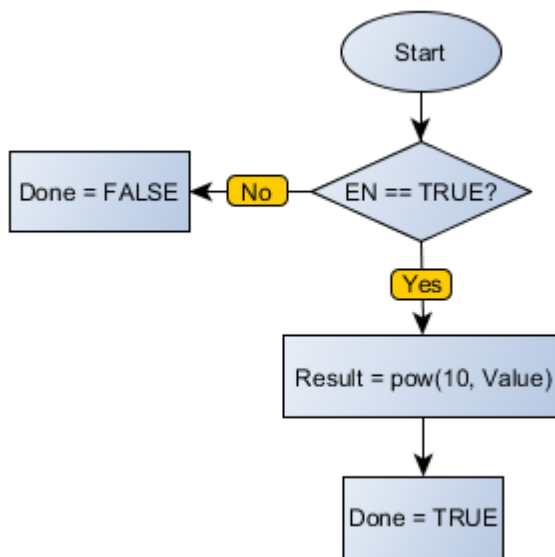
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

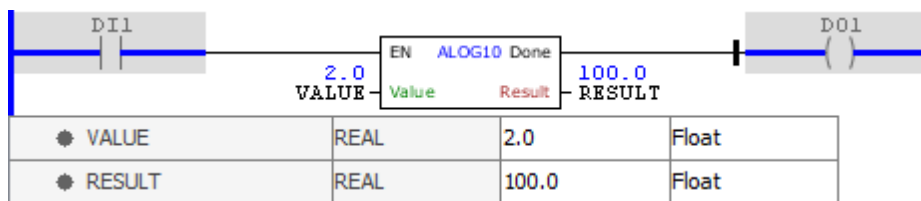
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o antilogaritmo da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

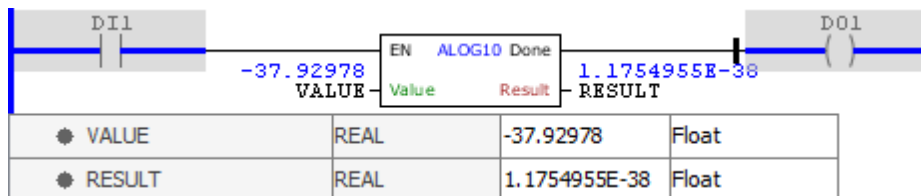
Fluxograma do Bloco



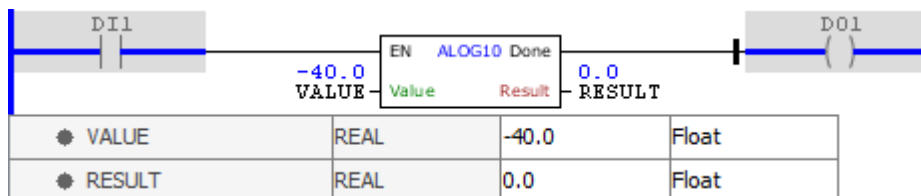
Exemplo



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.

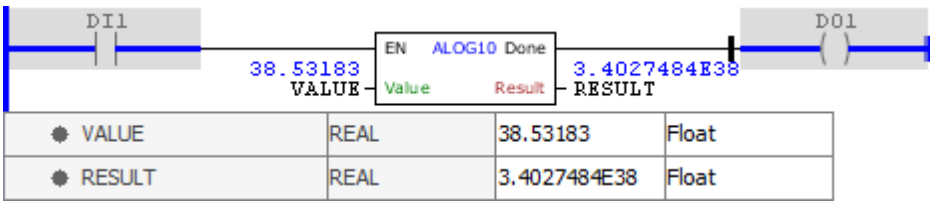


O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor mínimo de entrada para o qual o bloco retorna um resultado não nulo. O bloco finaliza com sucesso e a saída Done é ativada.

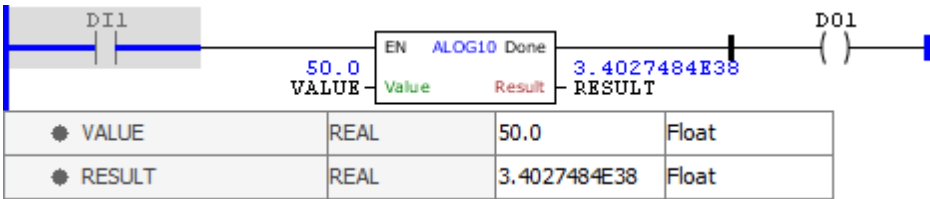


O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. Valores inferiores ao mínimo fazem com que o bloco retorne valor nulo. O bloco finaliza

com sucesso e a saída Done é ativada.



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor máximo de entrada para o qual o bloco retorna um resultado válido. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. Valores superiores ao máximo fazem com que o bloco gere um erro, a saída RESULT permanece inalterada e a saída Done seja desativada.

10.2.7.1.8.5 POW

Bloco que calcula o valor de Value elevado à potência Power, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Base da operação
	Power	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Potência da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

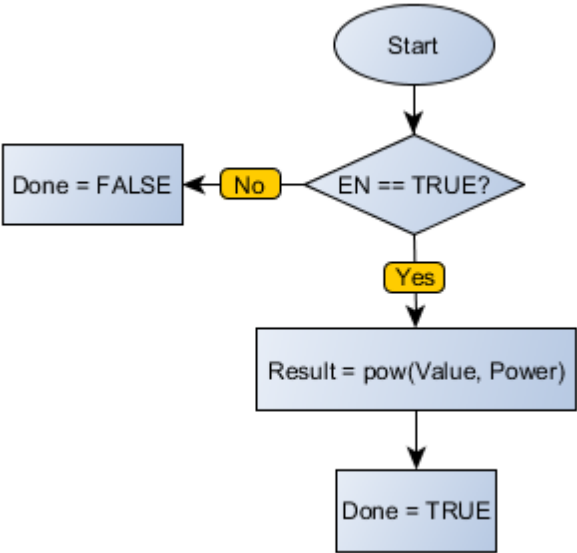
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor de Value elevado à potência Power. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação,

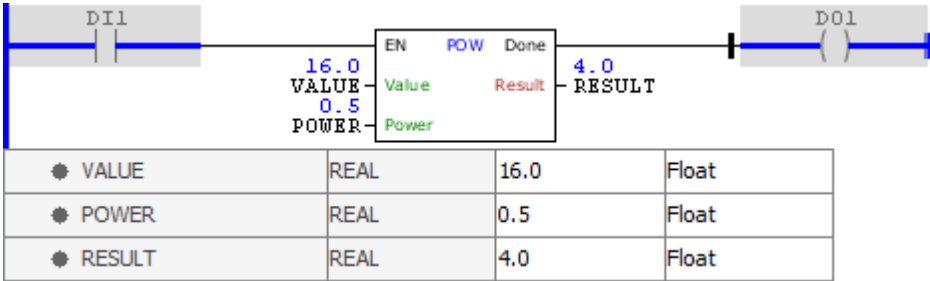
Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

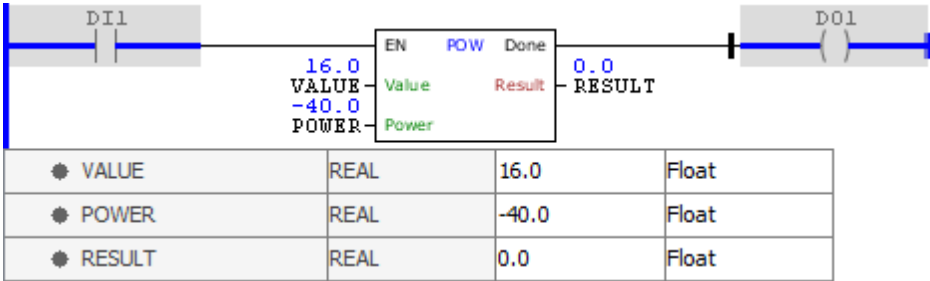
Fluxograma do Bloco



Exemplo

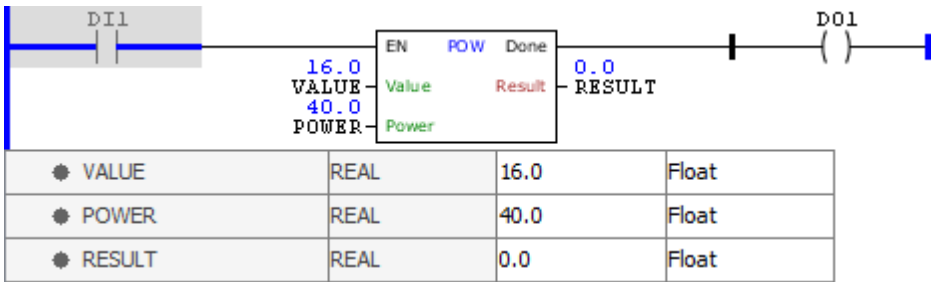


O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER,

armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER, armazenando o resultado final em RESULT. Como o resultado é superior ao valor máximo suportado pelo tipo REAL, o bloco gera um erro e a saída Done é desativada.

10.2.7.1.8.6 Sqrt

Bloco que calcula o valor da raiz quadrada de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

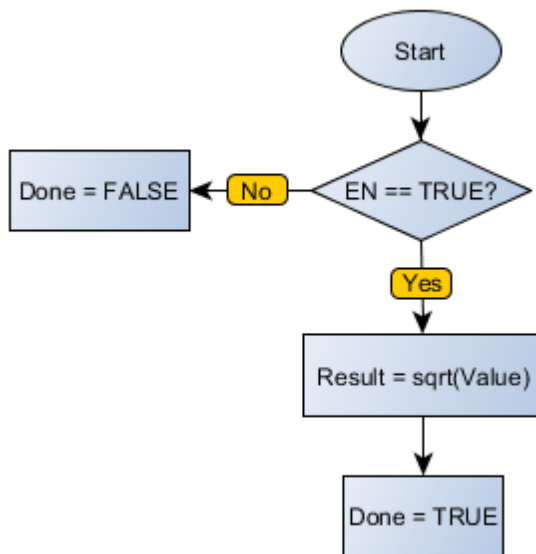
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

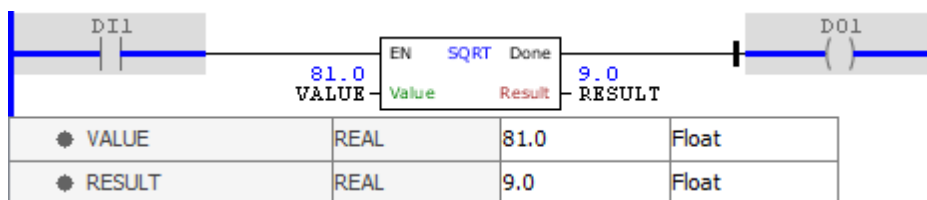
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da raiz quadrada de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

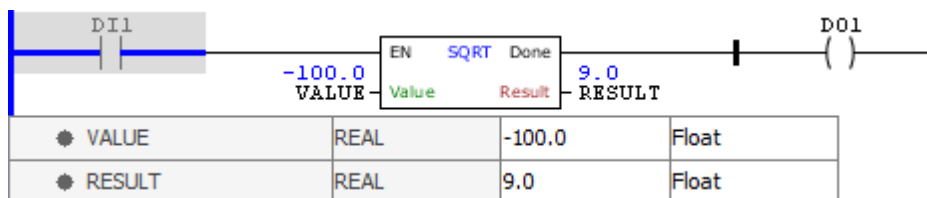
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o valor da raiz quadrada da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da raiz quadrada da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor negativo, e a saída Done é desativada.

10.2.7.1.8.7 ROUND

Bloco que arredonda o valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

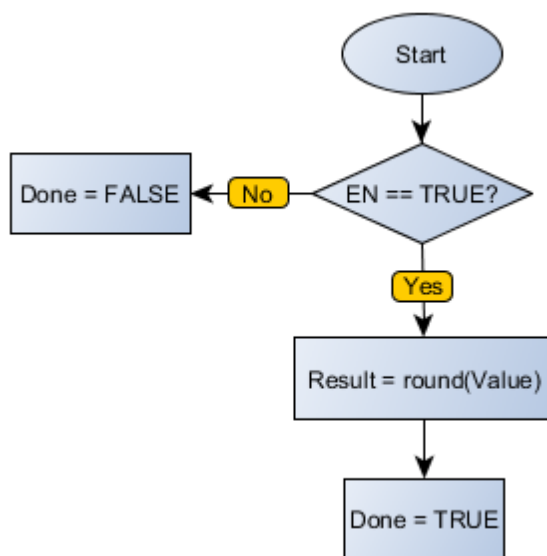
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor arredondado de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

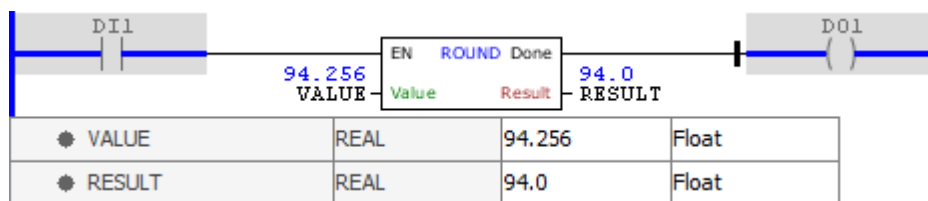
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior
SCA06	2.00 ou superior

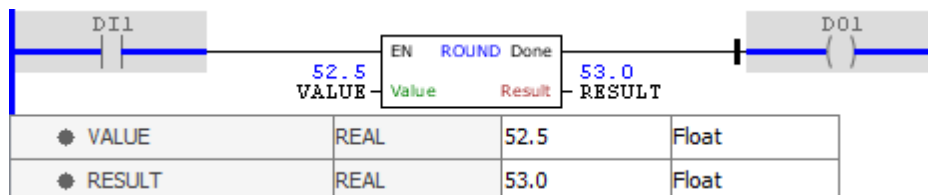
Fluxograma do Bloco



Exemplo



O exemplo acima arredonda o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais inferiores a 0,5 são descartados. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima arredonda o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais iguais ou superiores a 0,5 promovem a unidade ao valor imediatamente acima. O bloco finaliza com sucesso e a saída Done é ativada.

10.2.7.1.8.8 TRUNC

Bloco que trunca o valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

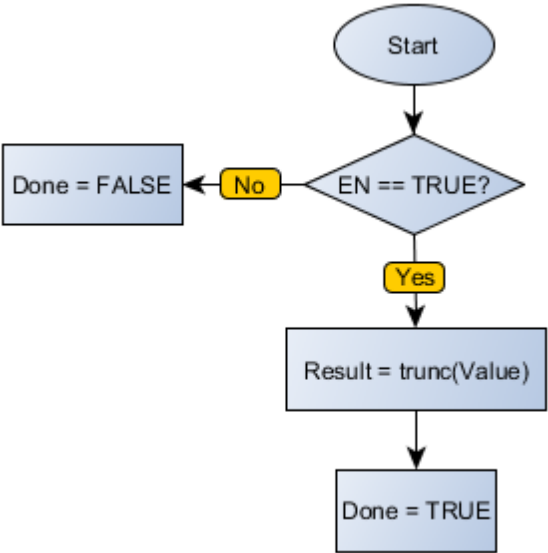
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor truncado de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

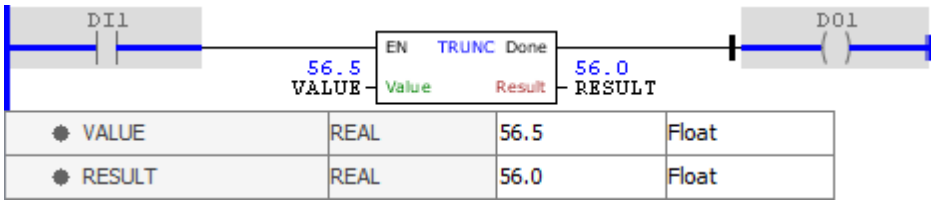
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



O exemplo acima trunca o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais são descartados. O bloco finaliza com sucesso e a saída Done é ativada.

10.2.7.1.9 Math Trigonometry

10.2.7.1.9.1 SIN

Bloco que calcula o seno de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

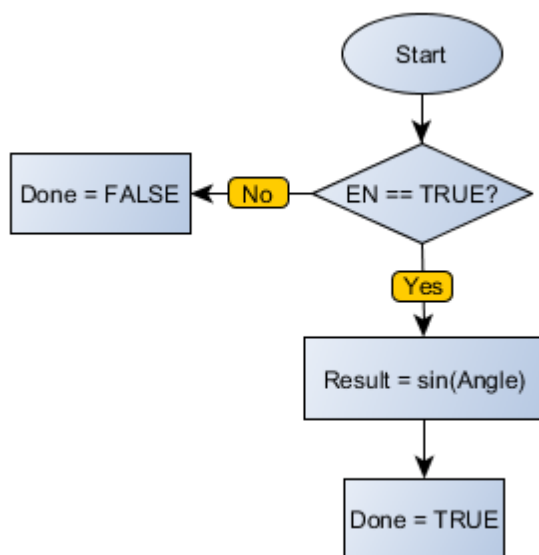
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

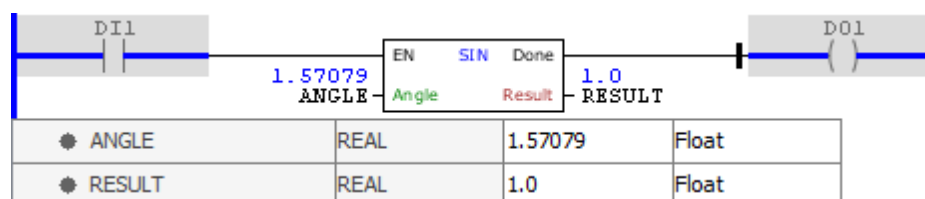
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o seno de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

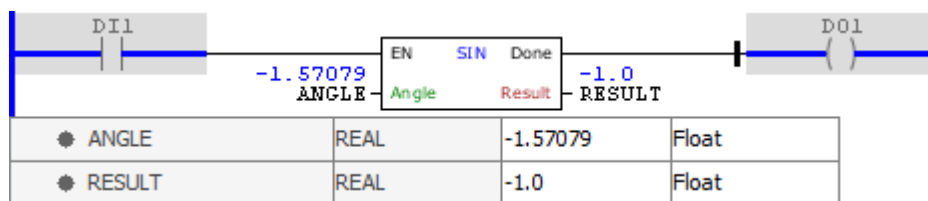
Fluxograma do Bloco



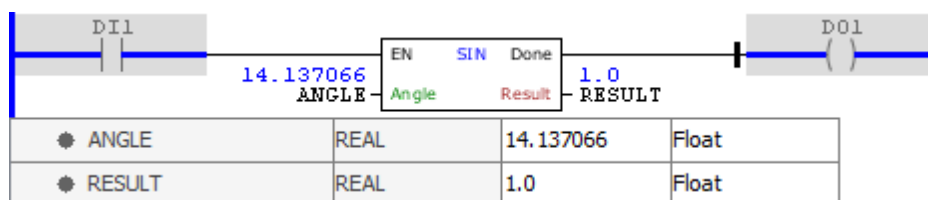
Exemplo



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores negativos de entrada.



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores superiores a uma volta completa.

10.2.7.1.9.2 COS

Bloco que calcula o cosseno de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

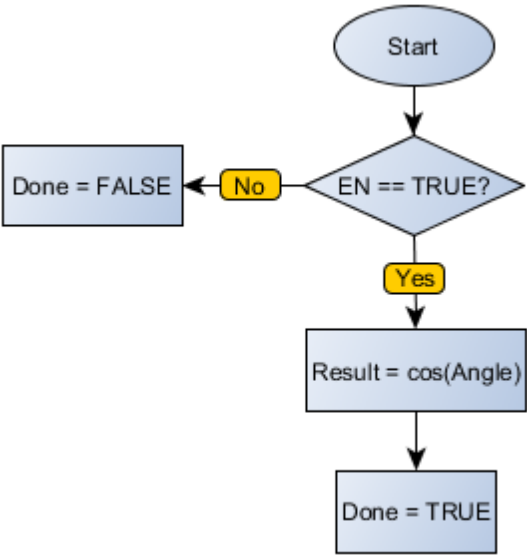
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

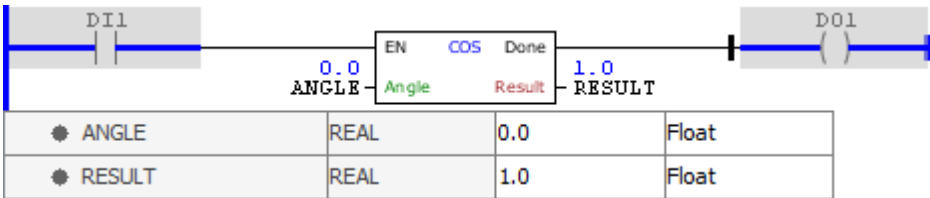
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o cosseno de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

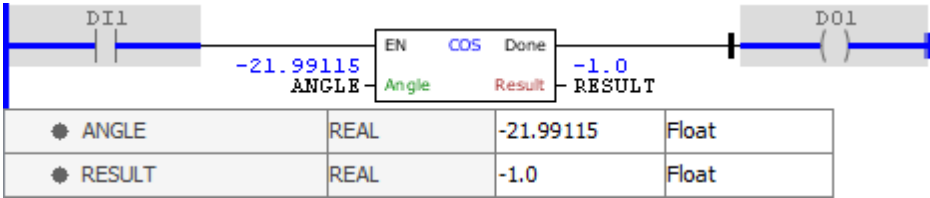
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o cosseno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.

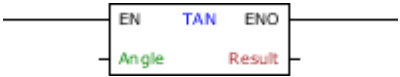


O exemplo acima calcula o cosseno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores de entrada negativos e superiores a uma volta.

10.2.7.1.9.3 TAN

Bloco que calcula a tangente de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

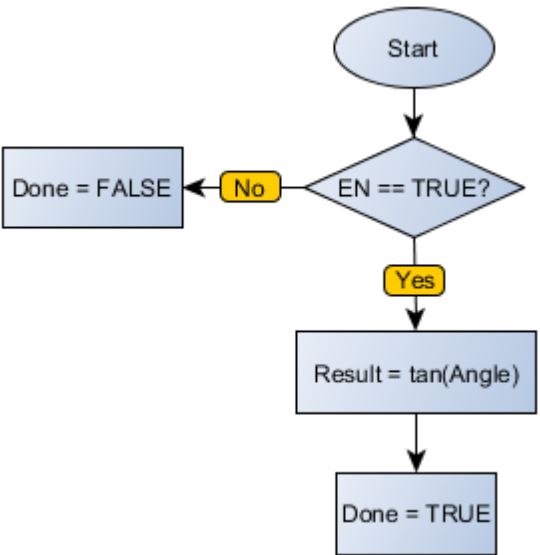
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

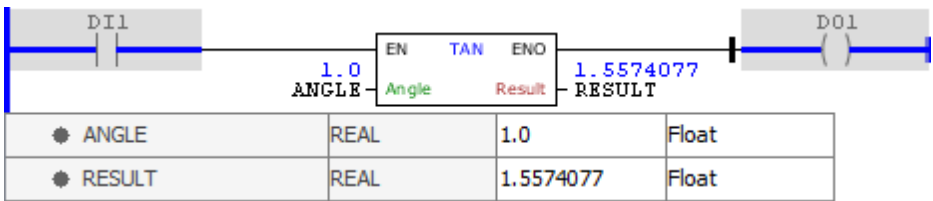
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a tangente de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

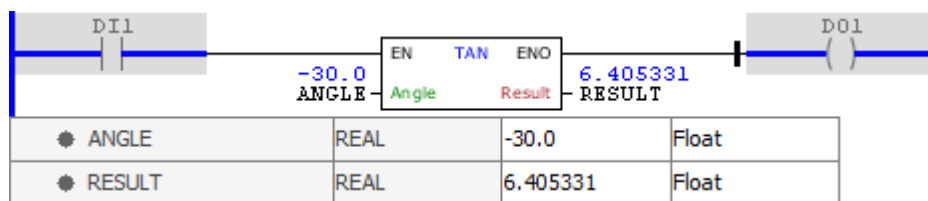
Fluxograma do Bloco



Exemplo



O exemplo acima calcula a tangente da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a tangente da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores de entrada negativos e superiores a uma volta.

10.2.7.1.9.4 ASIN

Bloco que calcula o arco seno de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

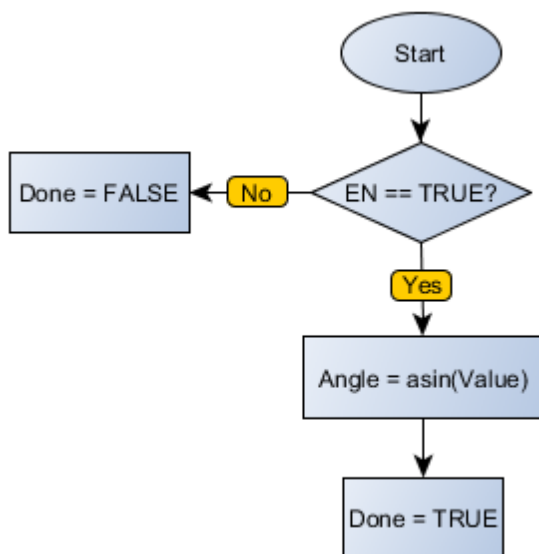
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor do seno
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cujo seno é igual a Value (em radianos)

Funcionamento

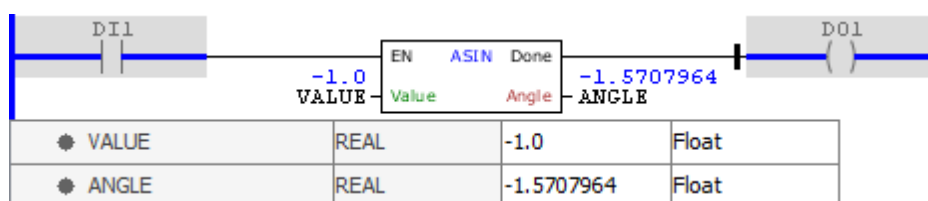
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco seno de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

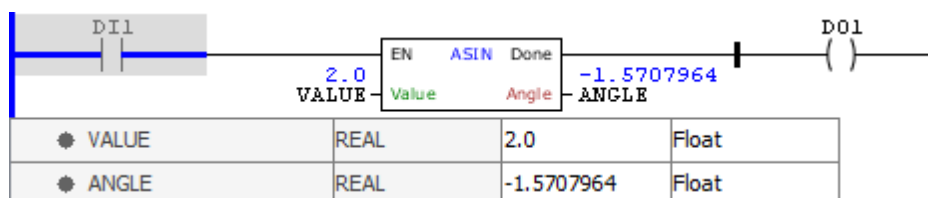
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cujo seno é a variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cujo seno é a variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor superior a 1, e a saída Done é desativada.

10.2.7.1.9.5 ACOS

Bloco que calcula o arco cosseno de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

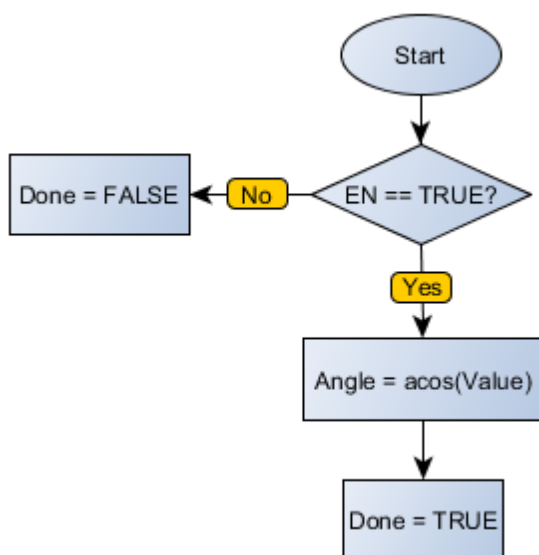
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor do cosseno
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cujo cosseno é igual a Value (em radianos)

Funcionamento

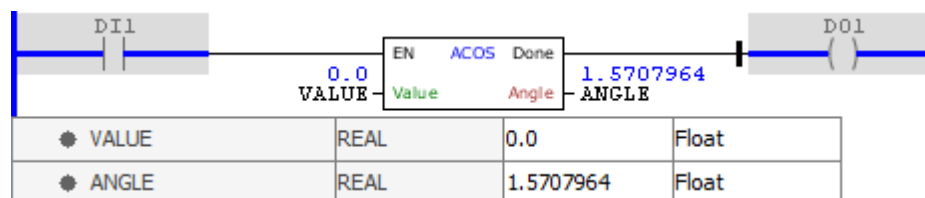
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco cosseno de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

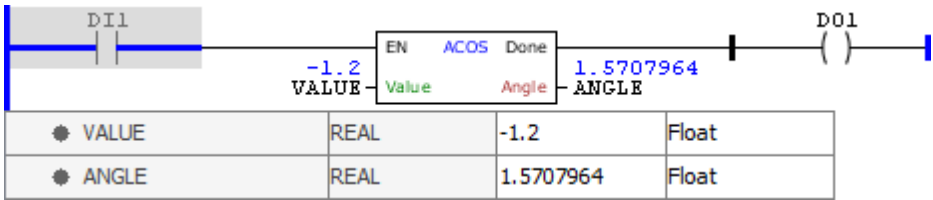
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cujo cosseno é a variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cujo cosseno é a variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor inferior a 1, e a saída Done é desativada.

10.2.7.1.9.6 ATAN

Bloco que calcula o arco tangente de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

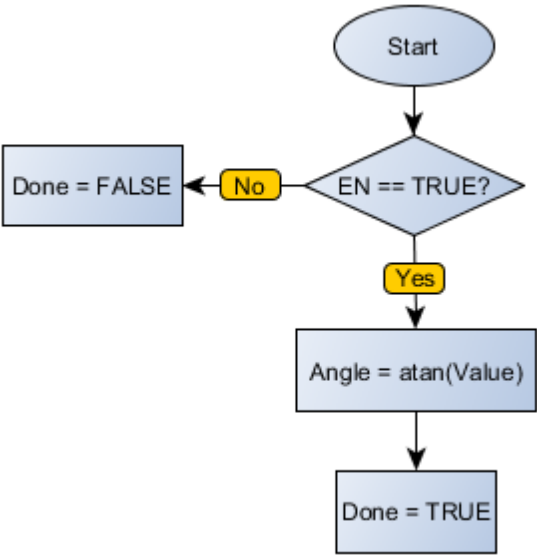
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor da tangente
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cuja tangente é igual a Value (em radianos)

Funcionamento

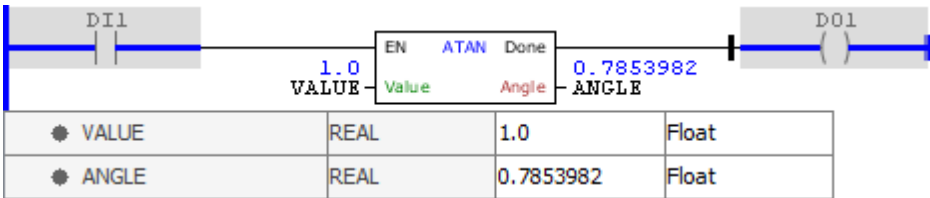
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco tangente de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

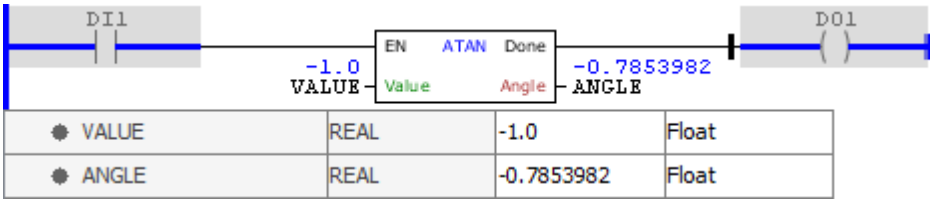
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cuja tangente é a variável VALUE, armazenando o resultado final em RESULT. O arco, para valores positivos, sempre se encontra no primeiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é a variável VALUE, armazenando o resultado final em RESULT. O arco, para valores negativos, sempre se encontra no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

10.2.7.1.9.7 ATAN2

Bloco que calcula o arco tangente de Y/X, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

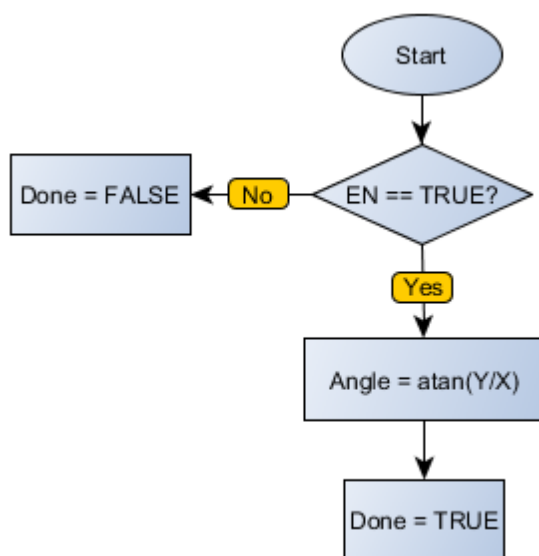
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	X	REAL	Parâmetro X da função
	Y	REAL	Parâmetro Y da função
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cuja tangente é igual a (Y/X) (em radianos)

Funcionamento

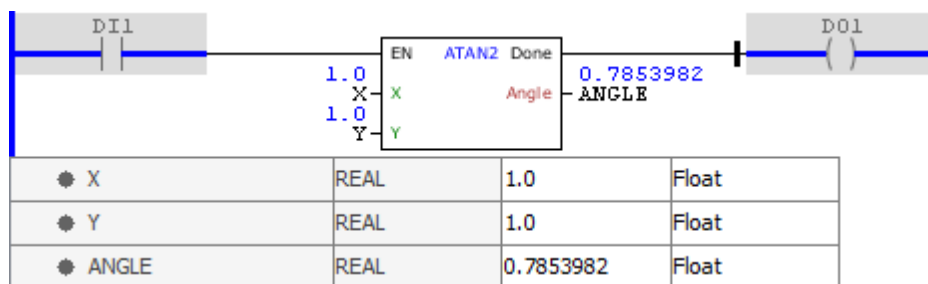
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco tangente de Y/X. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

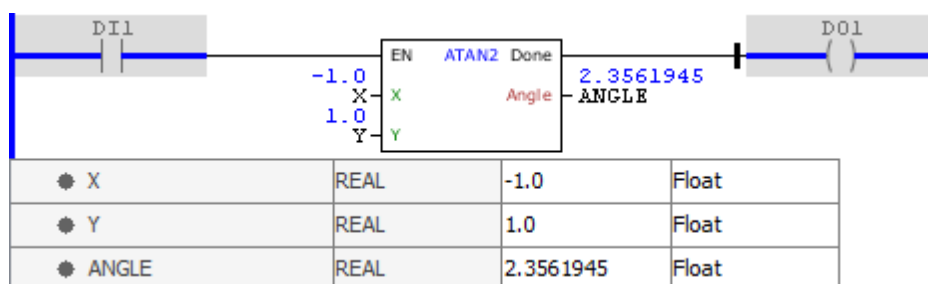
Fluxograma do Bloco



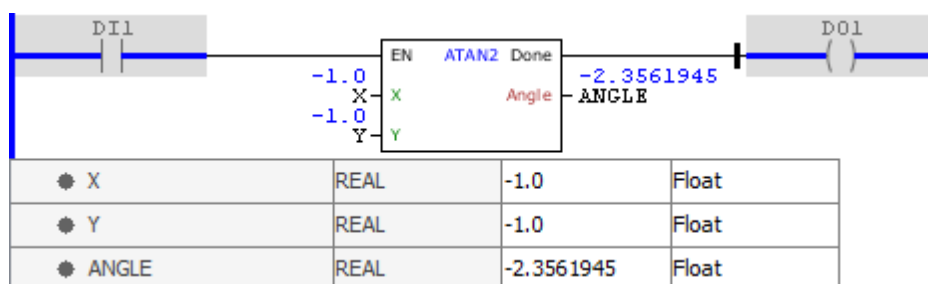
Exemplo



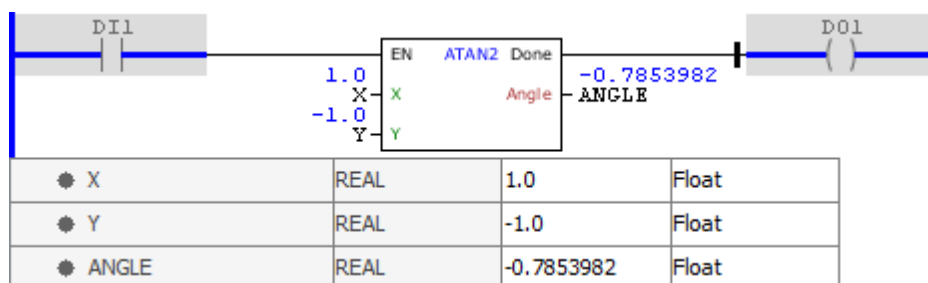
O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores positivos de X e Y, sempre se encontra no primeiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores negativos de X e positivos de Y, sempre se encontra no segundo quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores negativos de X e Y, sempre se encontra no terceiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores positivos de X e negativos de Y, sempre se encontra no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

10.2.7.1.10 Math Util

10.2.7.1.10.1 MAX

Bloco que compara os valores de Value1 e Value2 e armazena o maior deles em Result.

Representação Ladder



Estrutura do Bloco

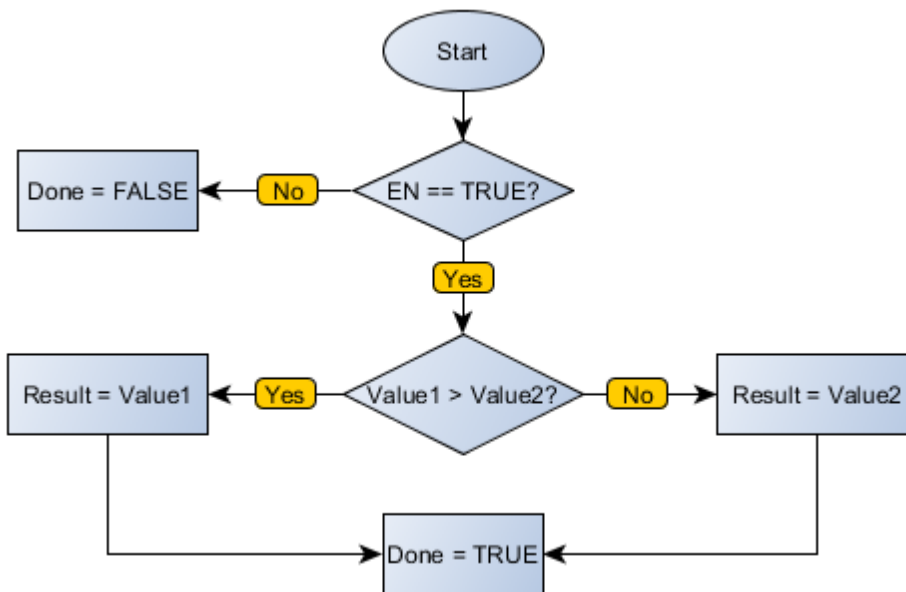
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Maior dos valores comparados

Funcionamento

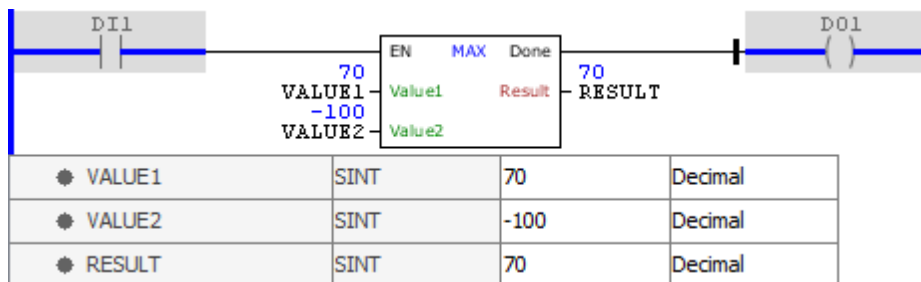
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o maior valor na comparação entre Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

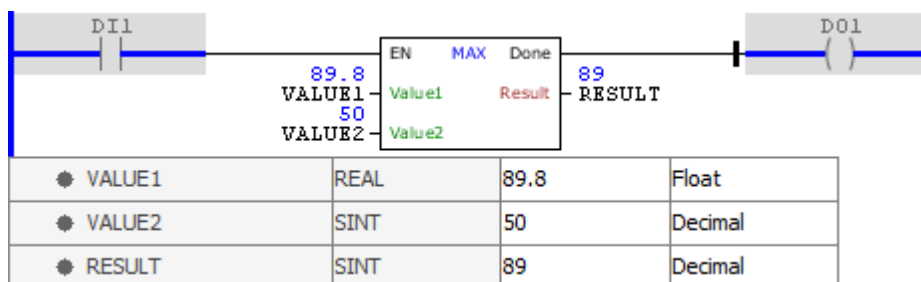
Fluxograma do Bloco



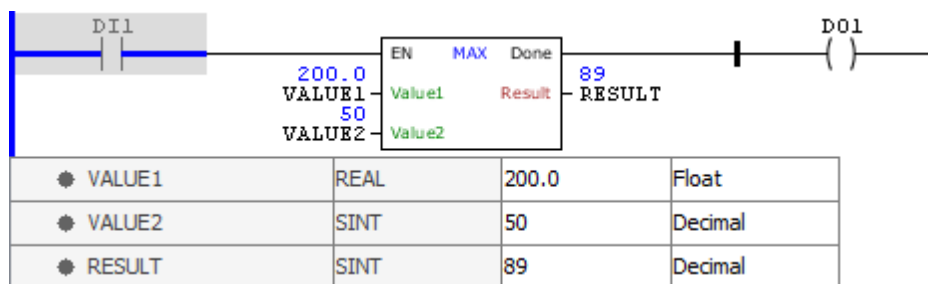
Exemplo



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como o resultado é superior ao valor máximo suportado pelo tipo SINT, o bloco gera um erro e a saída Done é desativada.

10.2.7.1.10.2 MIN

Bloco que compara os valores de Value1 e Value2 e armazena o menor deles em Result.

Representação Ladder



Estrutura do Bloco

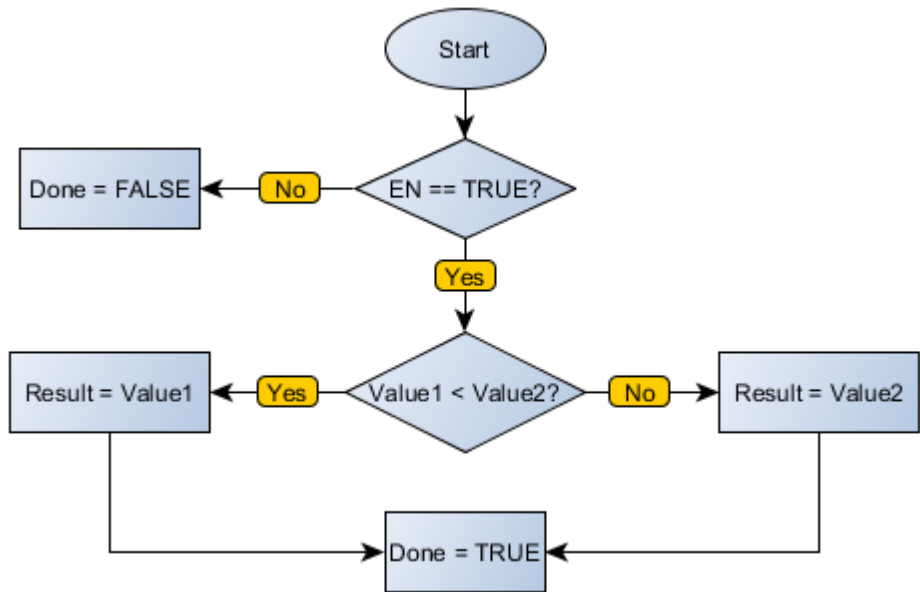
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Menor dos valores comparados

Funcionamento

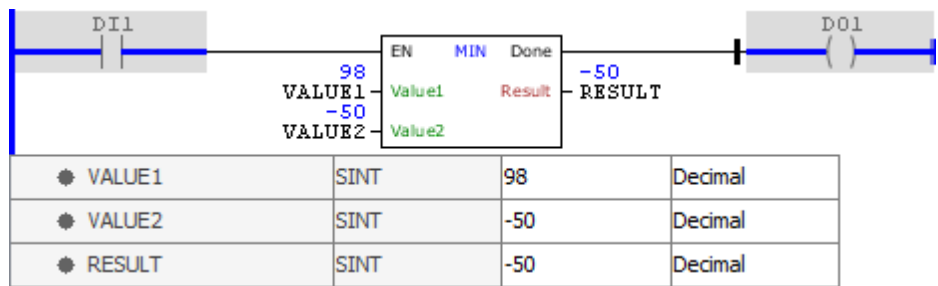
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o menor valor na comparação entre Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

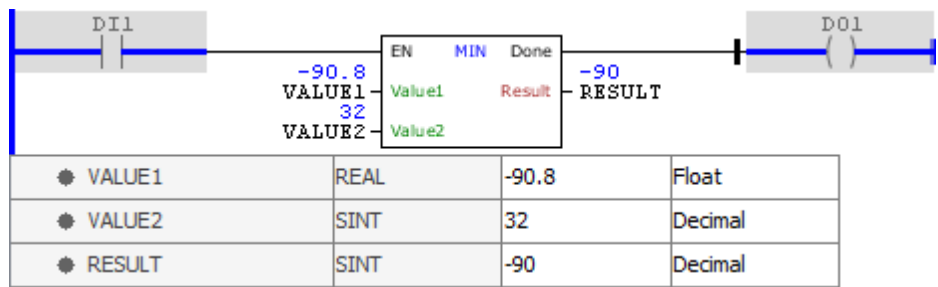
Fluxograma do Bloco



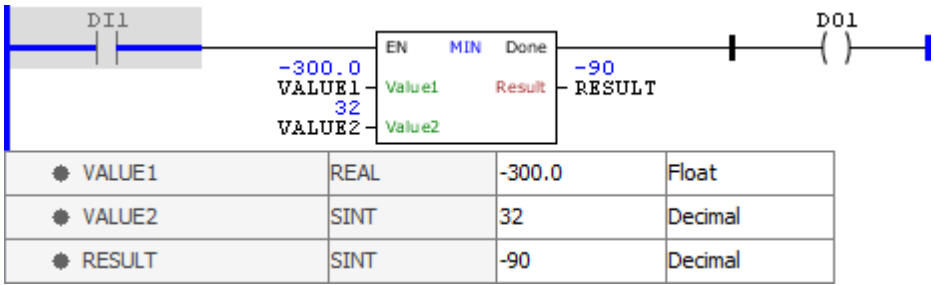
Exemplo



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como o resultado é inferior ao valor mínimo suportado pelo tipo SINT, o bloco gera um erro e a saída Done é desativada.

10.2.7.1.10.3 SAT

Bloco que realiza uma rotina de saturação do valor encontrado em Value de acordo com os limites estipulados em Minimum e Maximum, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

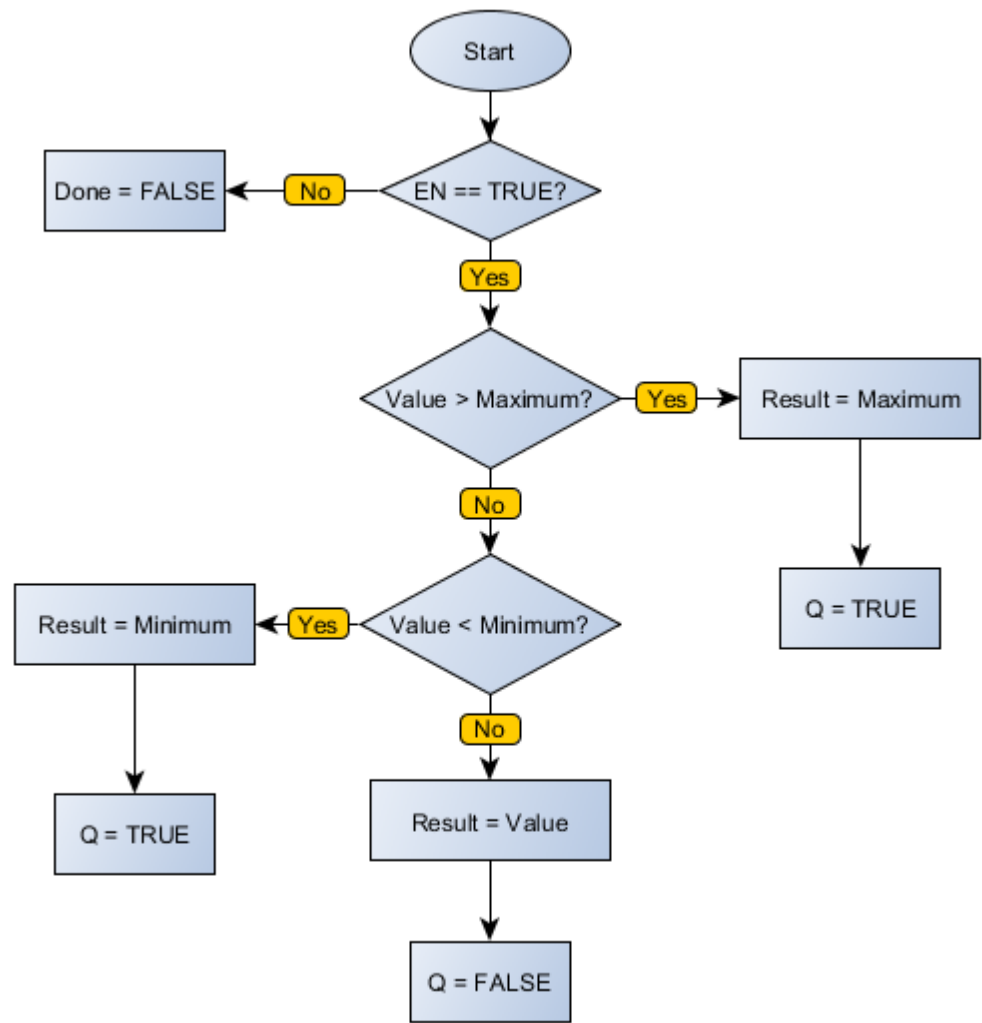
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de referência
	Minimum	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saturação inferior
	Maximum	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saturação superior
VAR_OUTPUT	Q	BOOL	Indicador de que houve saturação no processo
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Resultado da operação

Funcionamento

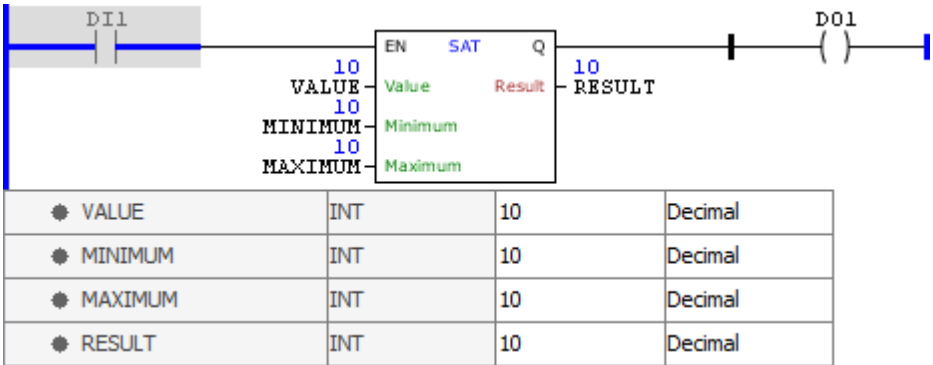
Este bloco, quando tem um valor TRUE em EN, realiza a comparação de Value com Minimum e Maximum. Se Value estiver na faixa entre Minimum e Maximum, Result recebe o valor de Value e Q permanece FALSE. Caso Value seja maior que Maximum, Result recebe Maximum e Q recebe TRUE. Caso Value seja menor que Minimum, Result recebe Minimum e Q recebe TRUE. Havendo algum erro na operação, Q não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Q permanece em FALSE.

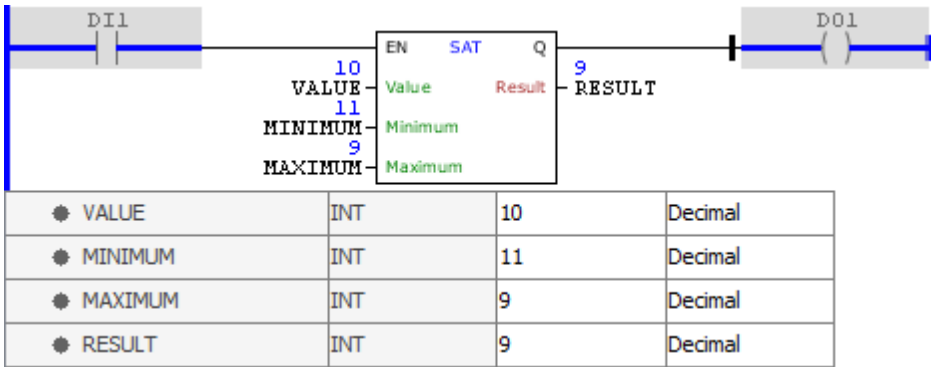
Fluxograma do Bloco



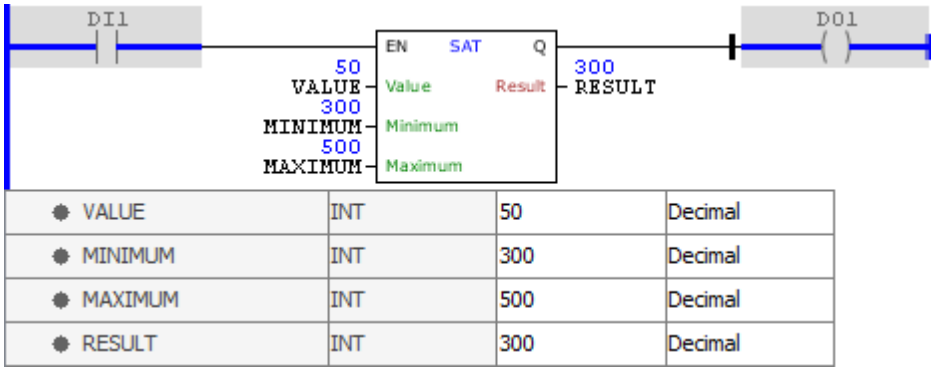
Exemplo



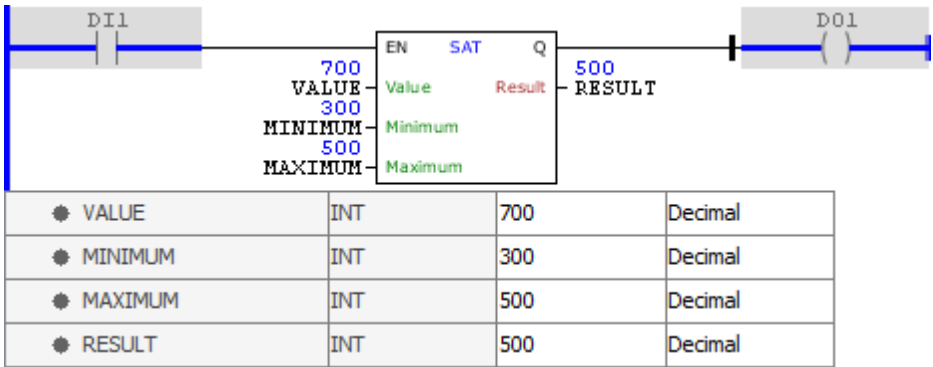
O exemplo acima repassa o valor de VALUE para RESULT, visto que ele não é inferior a MINIMUM ou superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é desativada, pois não houve saturação.



O exemplo acima repassa o valor de MAXIMUM para RESULT, visto que VALUE é superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.



O exemplo acima repassa o valor de MINIMUM para RESULT, visto que VALUE é inferior a MINIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.



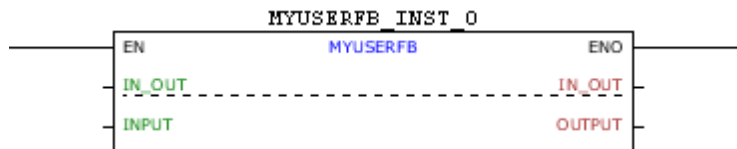
O exemplo acima repassa o valor de MAXIMUM para RESULT, visto que VALUE é superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.

10.2.7.1.11 Module

10.2.7.1.11.1 USERFB

Bloco que realiza as instruções programadas pelo próprio usuário.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	INPUT	Conforme programação do usuário	Entradas do bloco
VAR_OUTPUT	ENO	BOOL	Fim de operação
	OUTPUT	Conforme programação do usuário	Saídas do bloco
VAR_IN_OUT	IN_OUT	Conforme programação do usuário	Variáveis utilizadas como entrada e saída do bloco
VAR	MYUSERFB_INST_0	MYUSERFB	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, atualiza os valores dos campos internos com as variáveis de entrada, executa a rotina Ladder programada pelo usuário e atualiza os valores das saídas após finalizada a rotina.

Quando EN possui valor FALSE, as saídas permanecem inalteradas.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

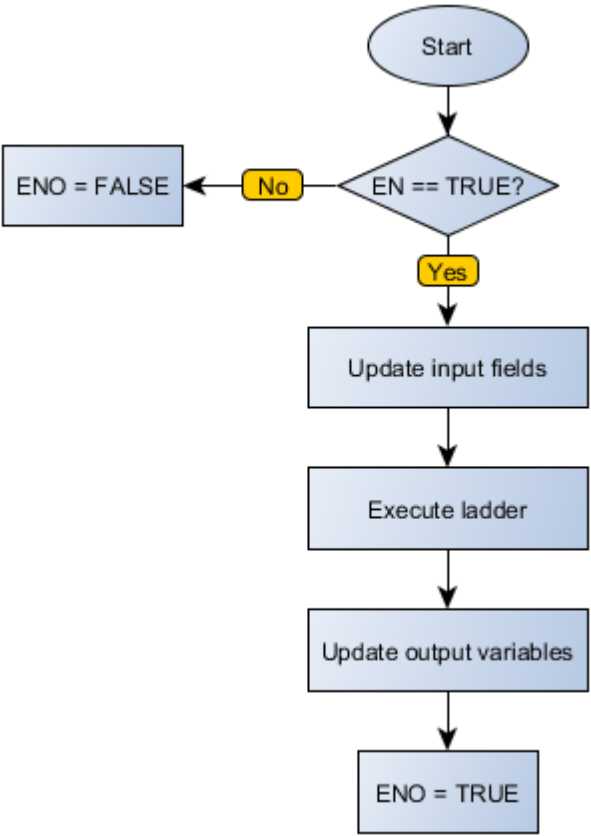
**NOTA!**

Consulte a seção [Trabalhando com USERFBs](#) para mais informações.

Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco

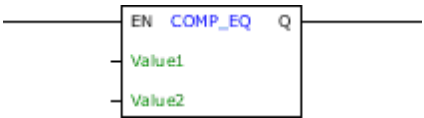


10.2.7.1.12 Comparação

10.2.7.1.12.1 COMP_EQ

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso ambos sejam iguais.

Representação Ladder



Estrutura do Bloco

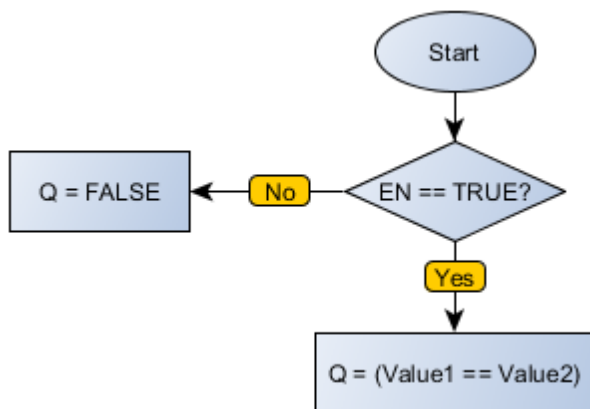
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade

Funcionamento

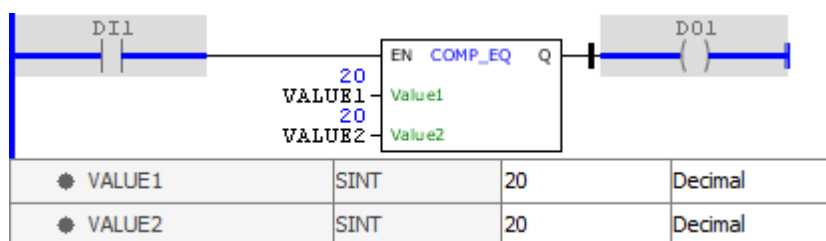
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 e Value2 sejam iguais. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

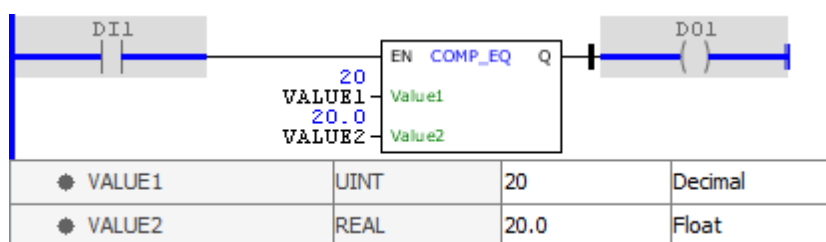
Fluxograma do Bloco



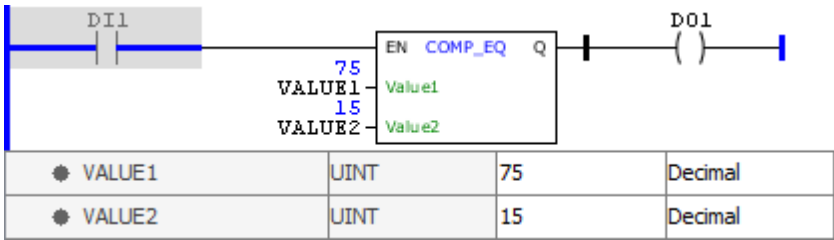
Exemplo



O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.



O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.

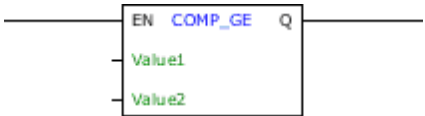


O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como as variáveis possuem valores diferentes, a saída Q é desativada.

10.2.7.1.12.2 COMP_GE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja maior ou igual a Value2.

Representação Ladder



Estrutura do Bloco

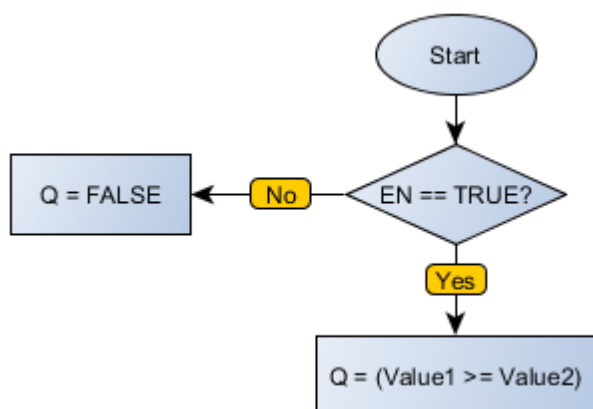
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade ou maioridade de Value1

Funcionamento

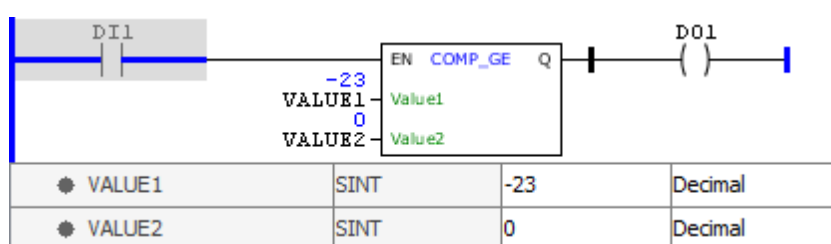
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja maior ou igual a Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

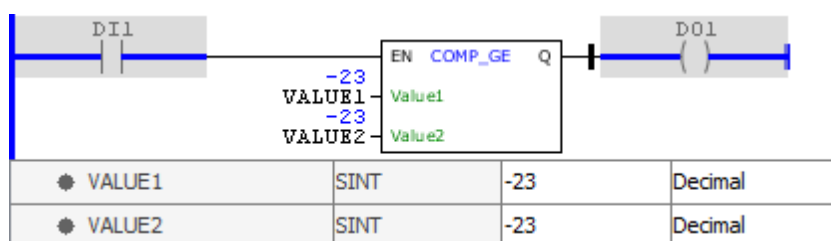
Fluxograma do Bloco



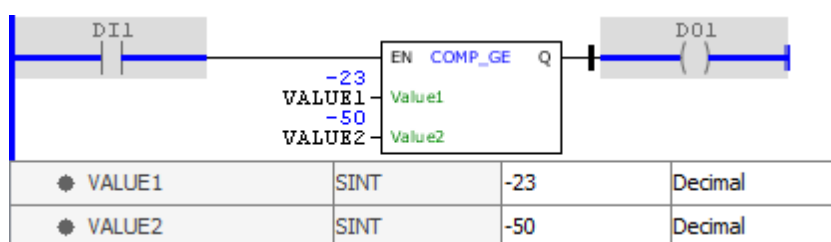
Exemplo



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é desativada.



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.

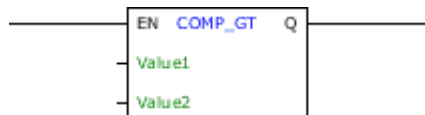


O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é ativada.

10.2.7.1.12.3 COMP_GT

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja maior que Value2.

Representação Ladder



Estrutura do Bloco

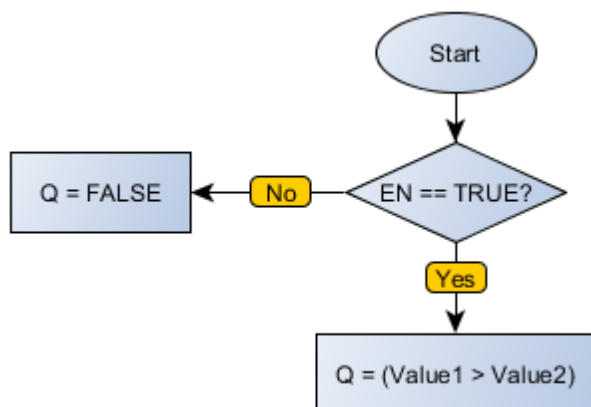
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de maioridade de Value1

Funcionamento

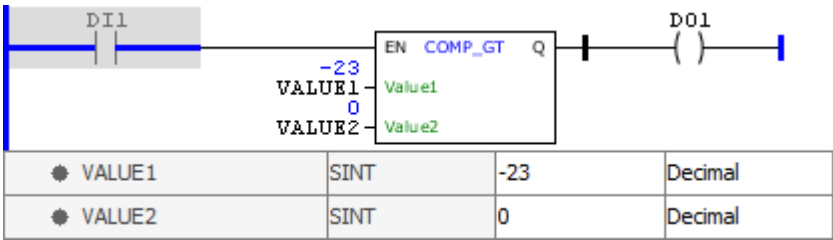
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja maior que Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

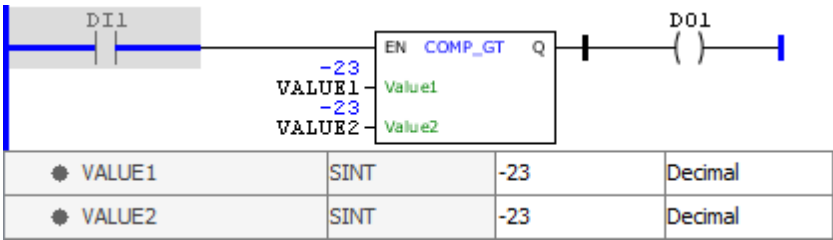
Fluxograma do Bloco



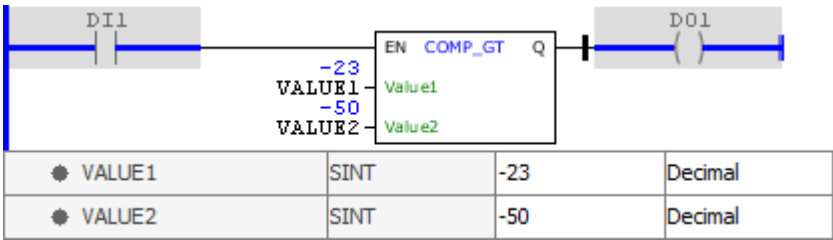
Exemplo



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é desativada.



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é ativada.

10.2.7.1.12.4 COMP_LE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja menor ou igual a Value2.

Representação Ladder



Estrutura do Bloco

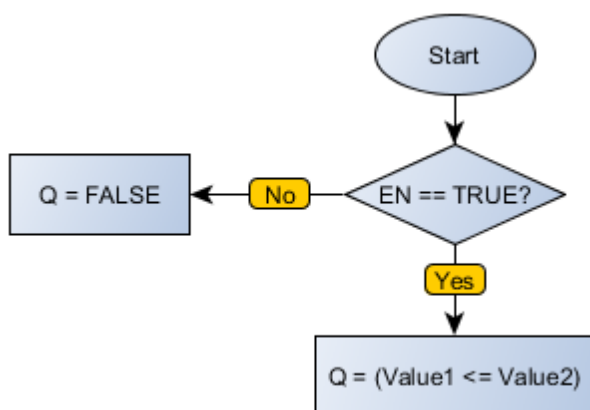
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade ou minoridade de Value1

Funcionamento

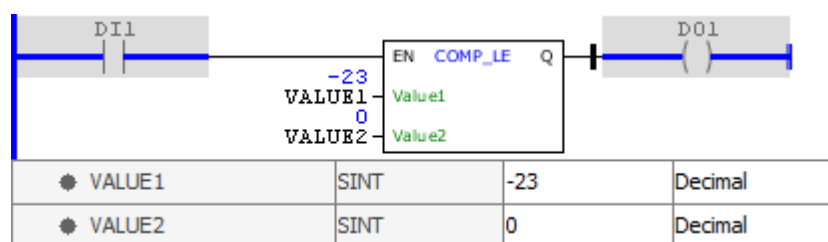
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja menor ou igual a Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

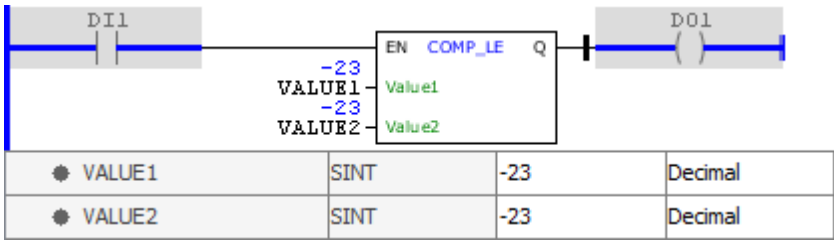
Fluxograma do Bloco



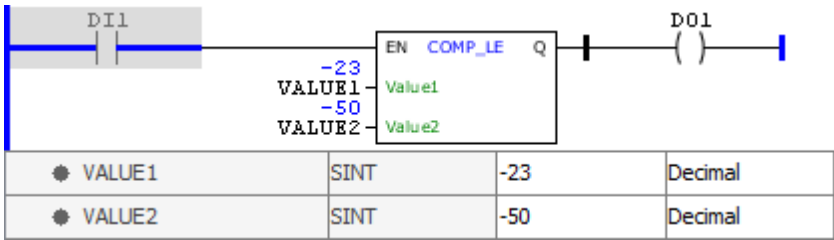
Exemplo



O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é ativada.



O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.

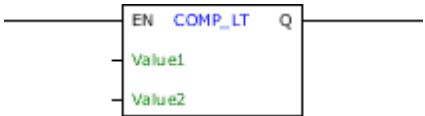


O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é desativada.

10.2.7.1.12.5 COMP_LT

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja menor que Value2.

Representação Ladder



Estrutura do Bloco

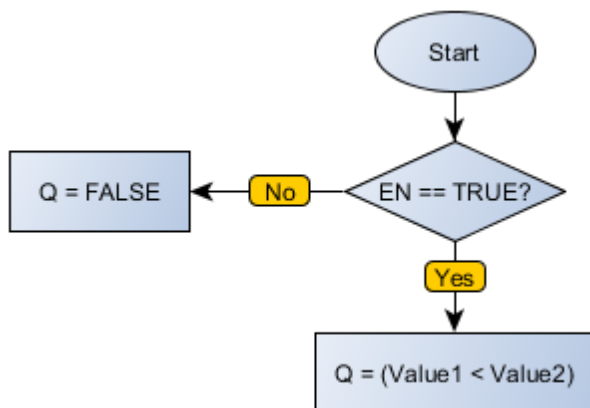
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de minoridade de Value1

Funcionamento

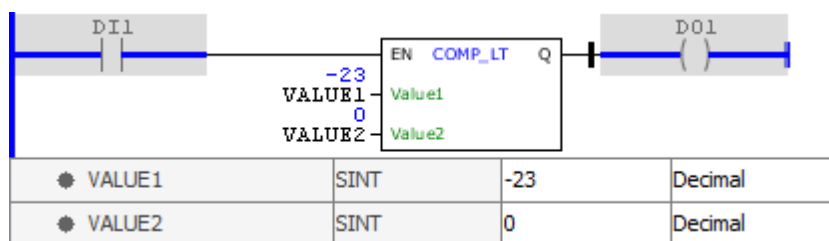
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja menor que Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

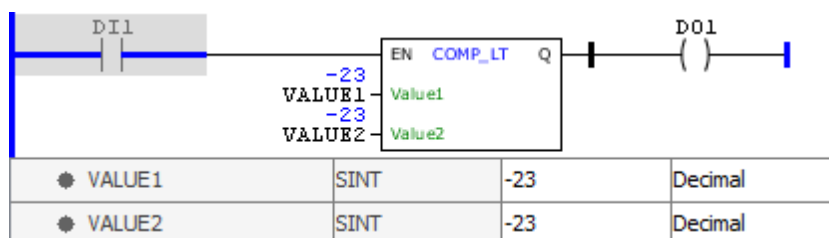
Fluxograma do Bloco



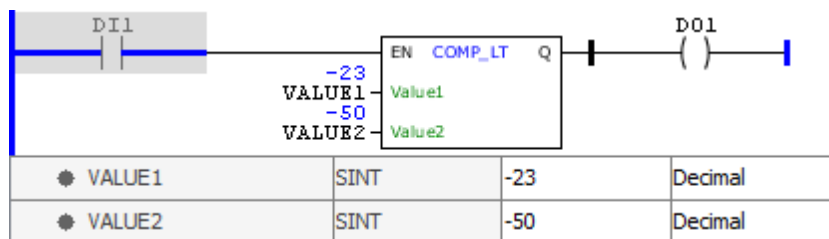
Exemplo



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é ativada.



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é desativada.

10.2.7.1.12.6 COMP_NE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja diferente de Value2.

Representação Ladder



Estrutura do Bloco

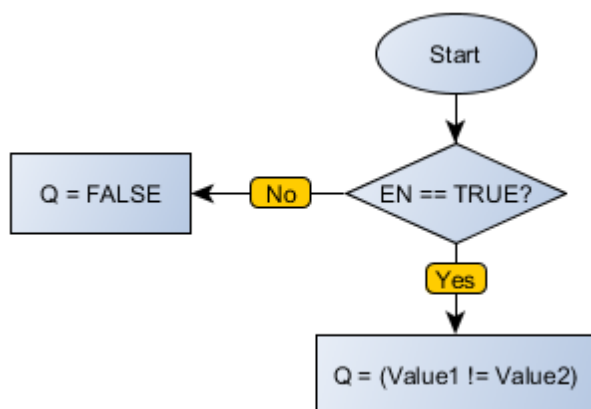
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de desigualdade

Funcionamento

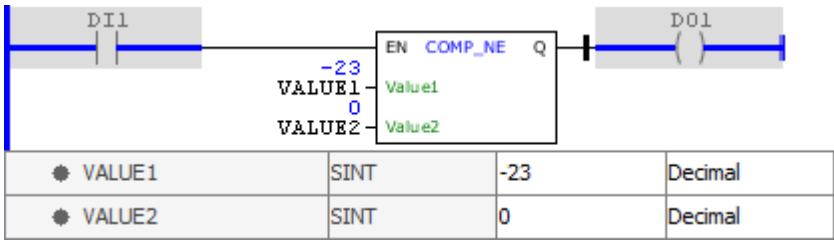
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja diferente de Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

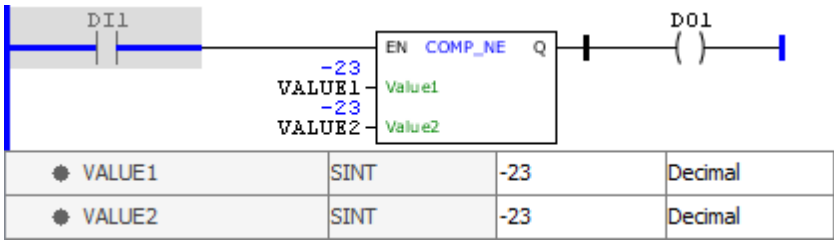
Fluxograma do Bloco



Exemplo



O exemplo acima verifica a desigualdade entre VALUE1 e VALUE2. Como as variáveis possuem valores diferentes, a saída Q é ativada.



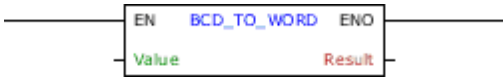
O exemplo acima verifica a desigualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.

10.2.7.1.13 Conversão

10.2.7.1.13.1 BCD

Bloco que realiza a conversão de um código BCD para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em BCD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

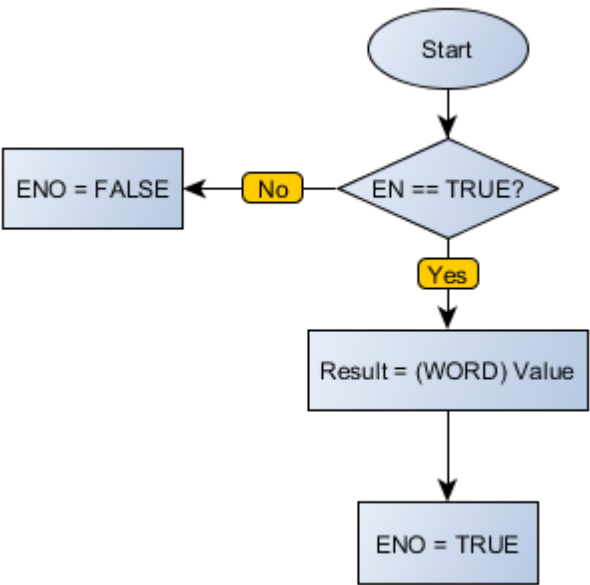
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BCD e realiza a conversão para WORD, armazenando em Result.

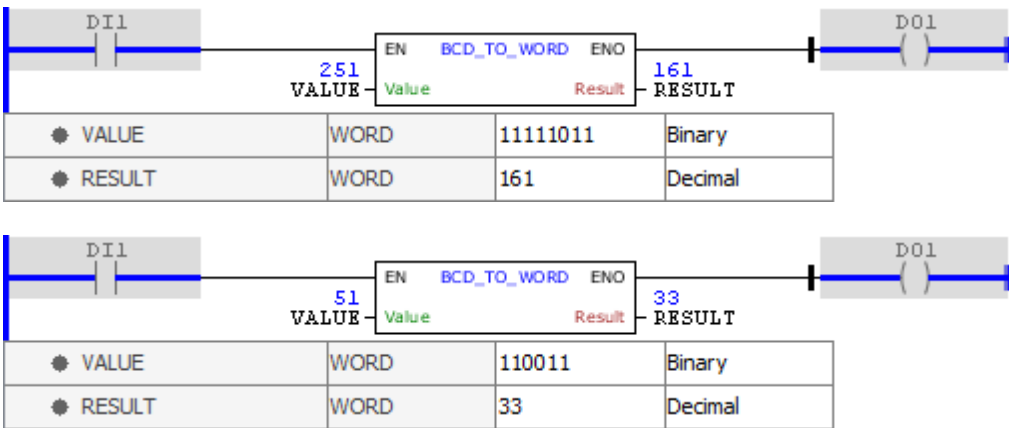
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



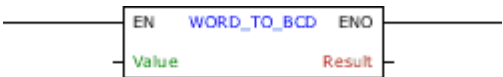
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BCD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um código BCD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em BCD

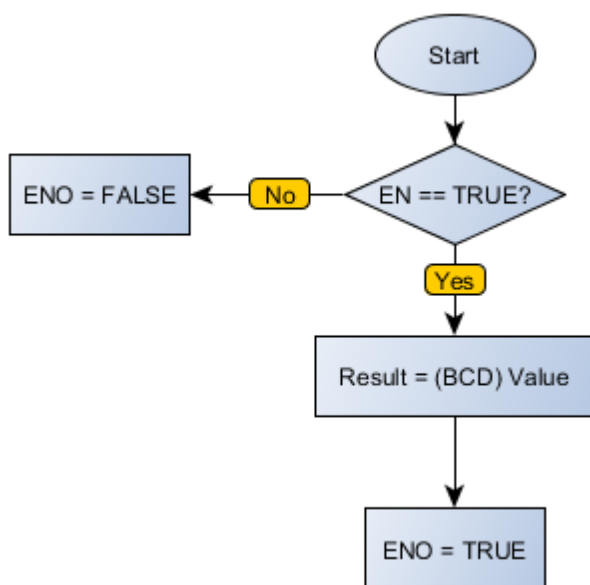
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BCD, armazenando em Result.

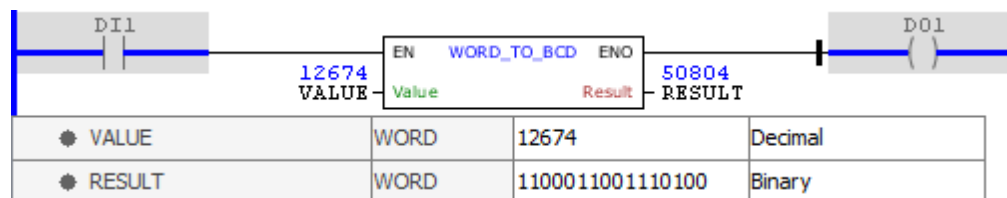
Quando EN possui valor FALSE, Result permanece inalterado.

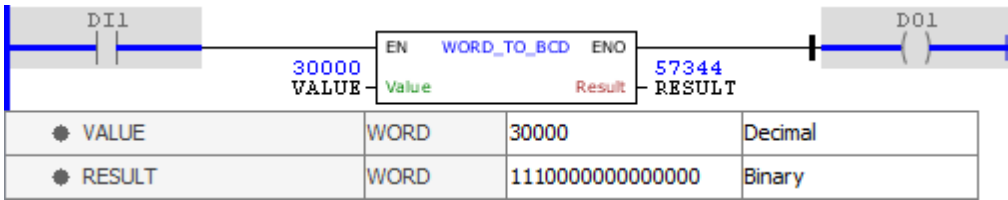
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



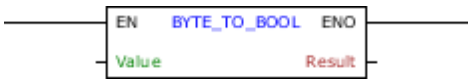


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BCD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.13.2 BOOL

Bloco que realiza a conversão de um valor BYTE para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

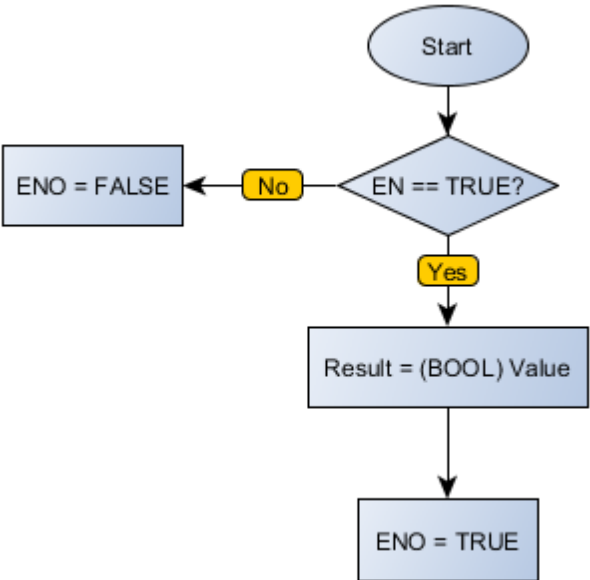
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para BOOL, armazenando em Result.

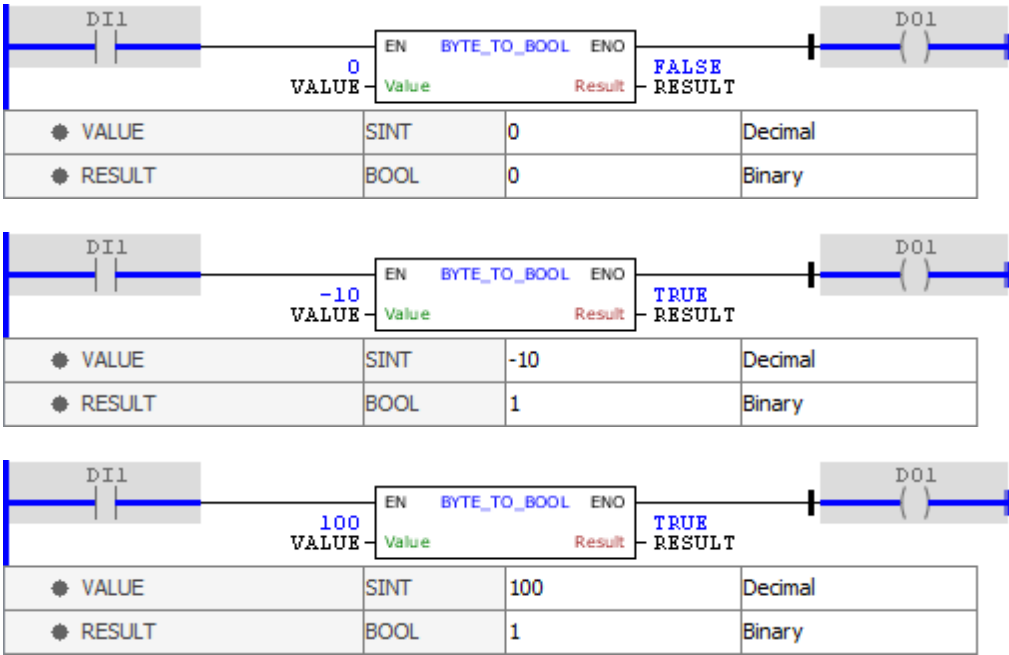
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



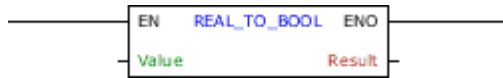
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

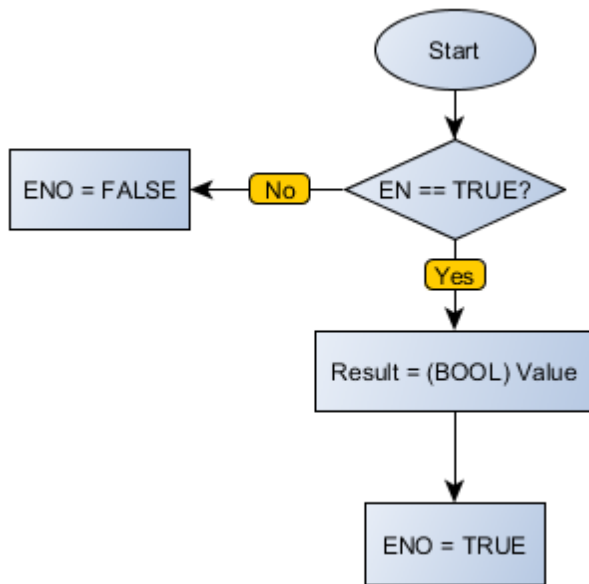
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para BOOL, armazenando em Result.

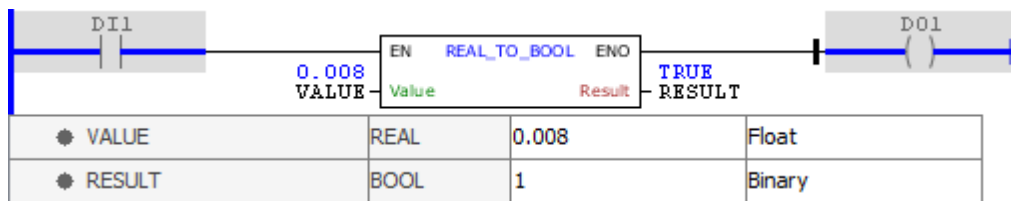
Quando EN possui valor FALSE, Result permanece inalterado.

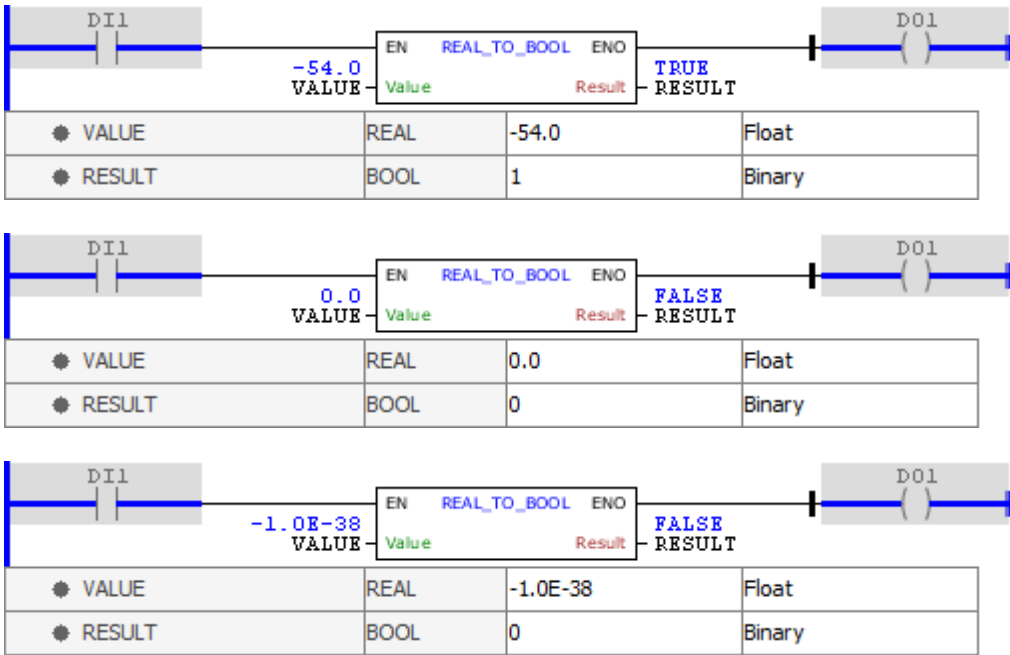
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

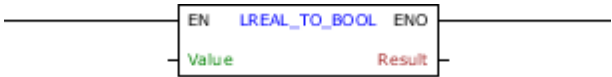




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe no último exemplo que valores muito próximos do epsilon da máquina podem acarretar numa interpretação de valor FALSE.

Bloco que realiza a conversão de um valor LREAL para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

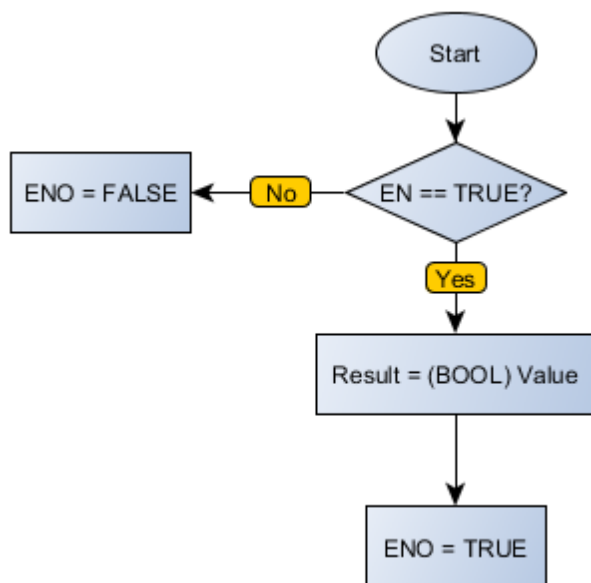
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para BOOL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

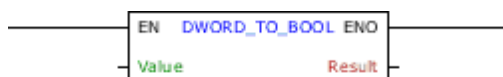
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor DWORD para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

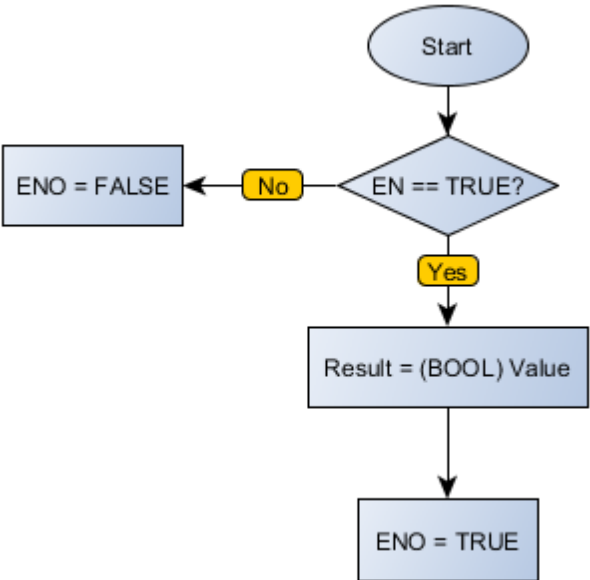
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para BOOL, armazenando em Result.

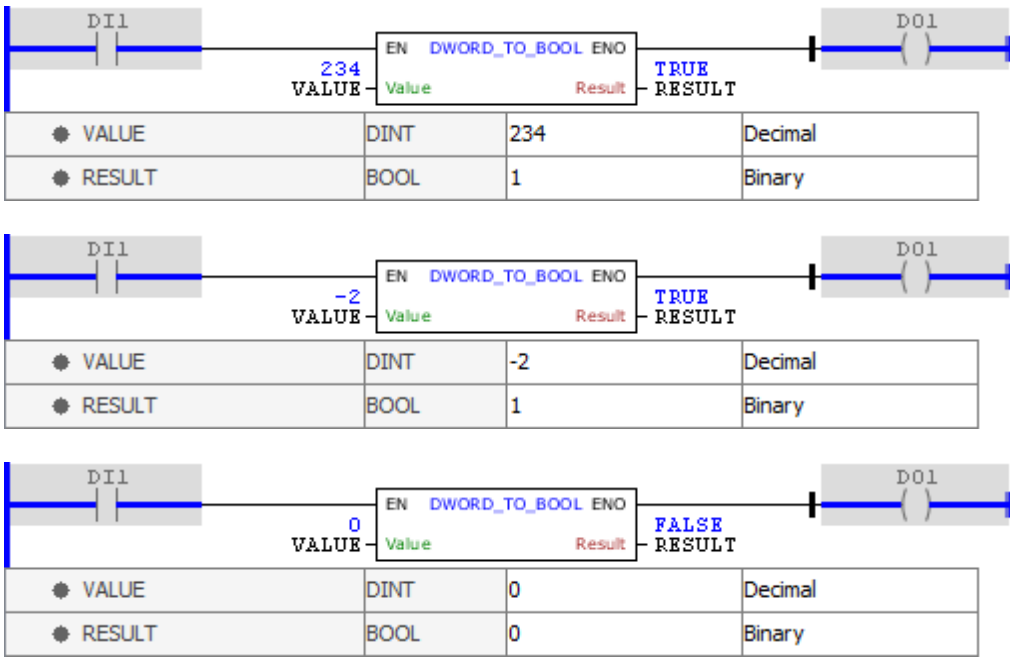
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



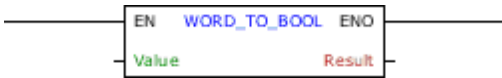
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

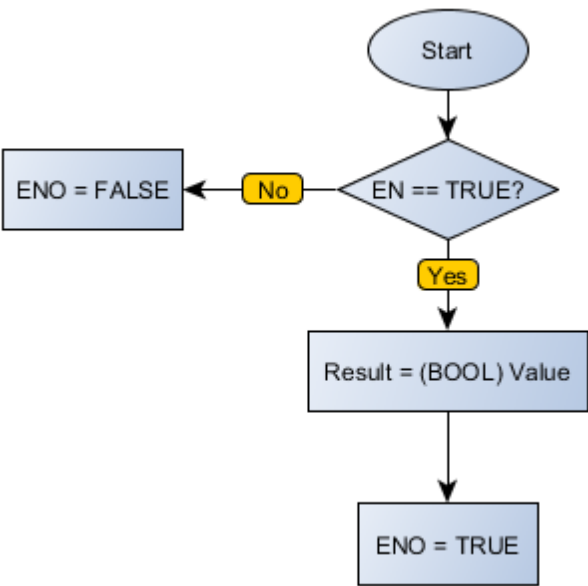
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BOOL, armazenando em Result.

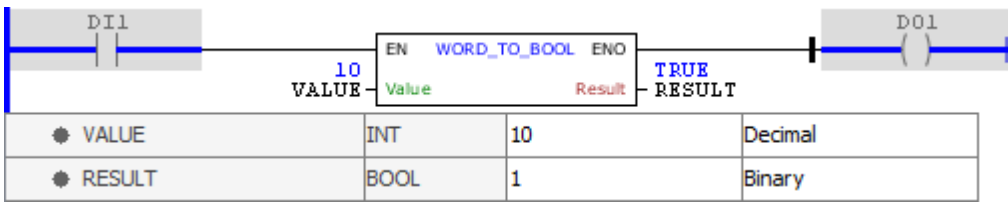
Quando EN possui valor FALSE, Result permanece inalterado.

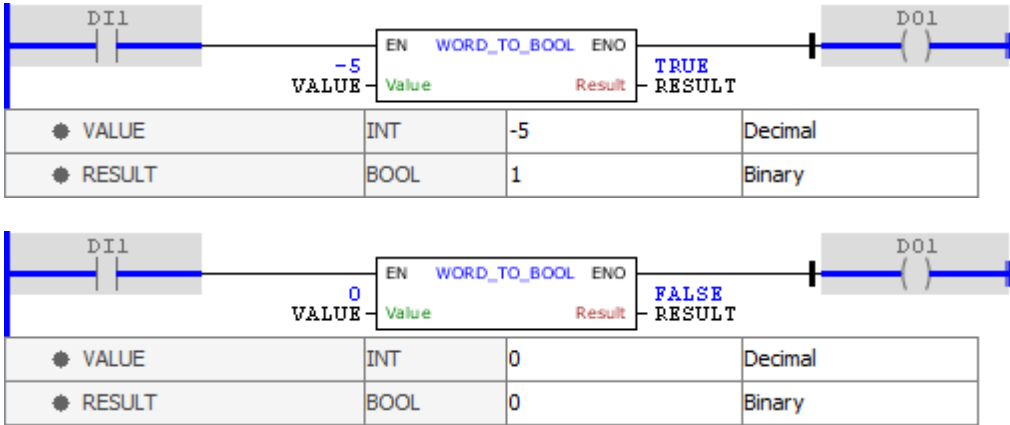
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



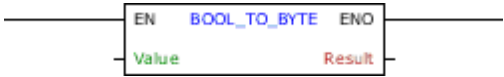


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.13.3 BYTE

Bloco que realiza a conversão de um valor BOOL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

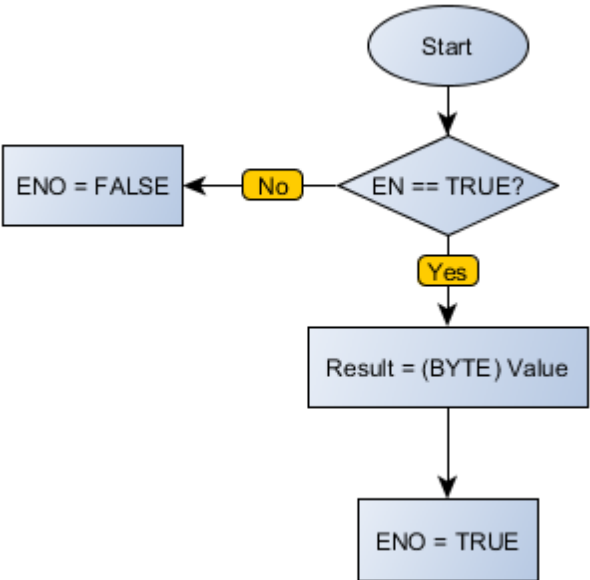
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para BYTE, armazenando em Result.

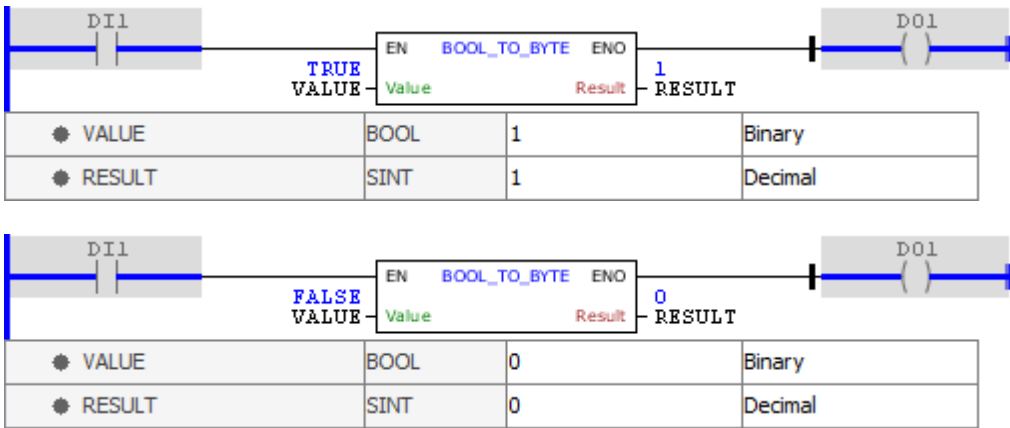
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



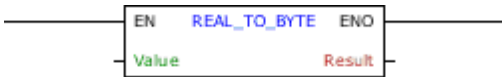
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

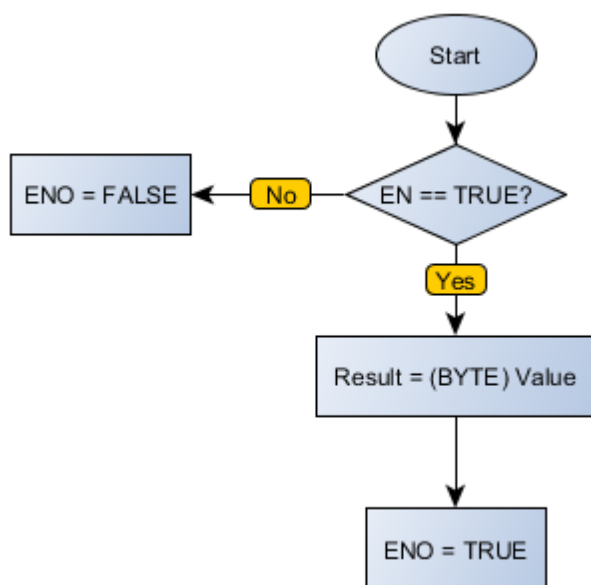
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para BYTE, armazenando em Result.

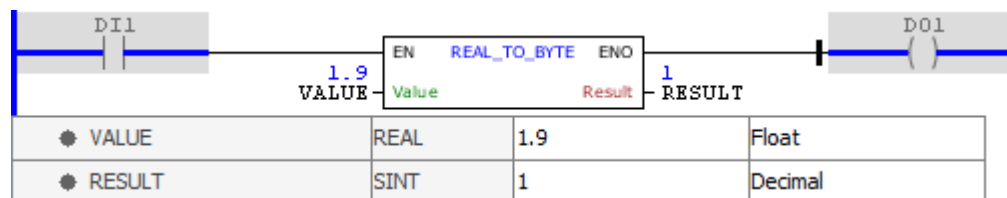
Quando EN possui valor FALSE, Result permanece inalterado.

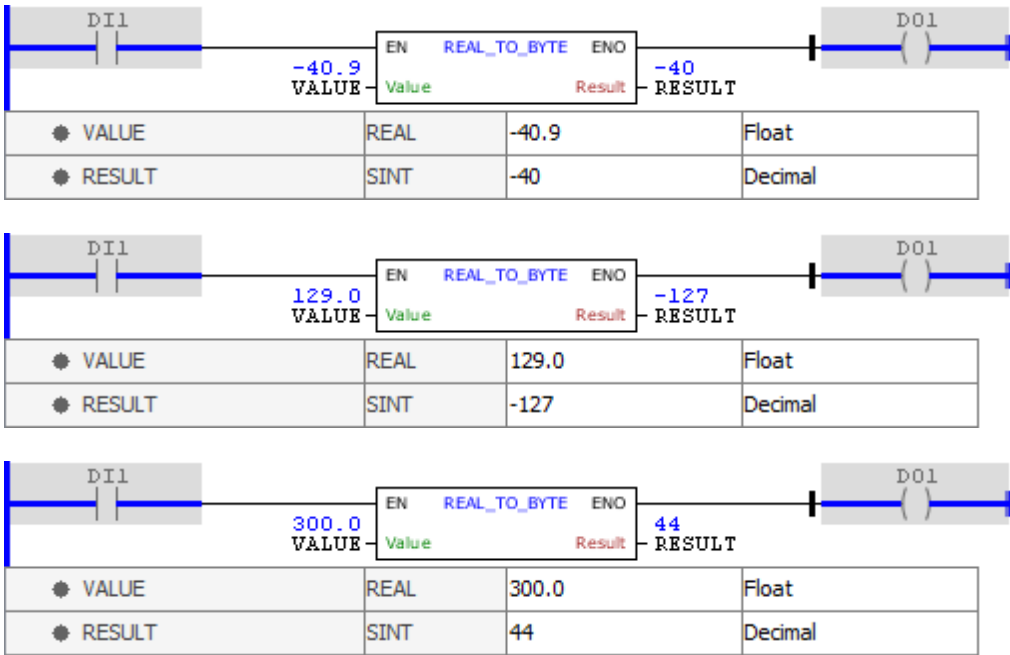
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

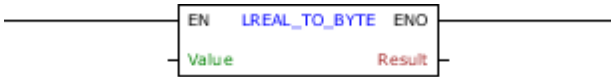




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os oito bits menos significativos.

Bloco que realiza a conversão de um valor LREAL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

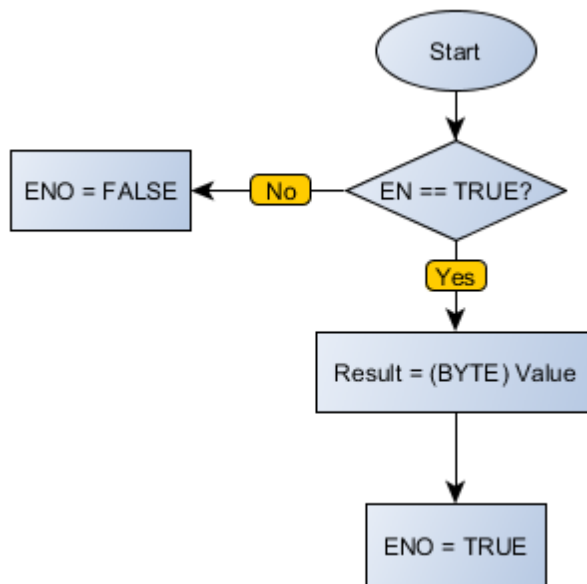
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para BYTE, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

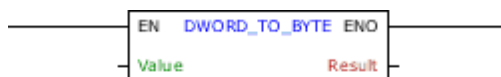
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor DWORD para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

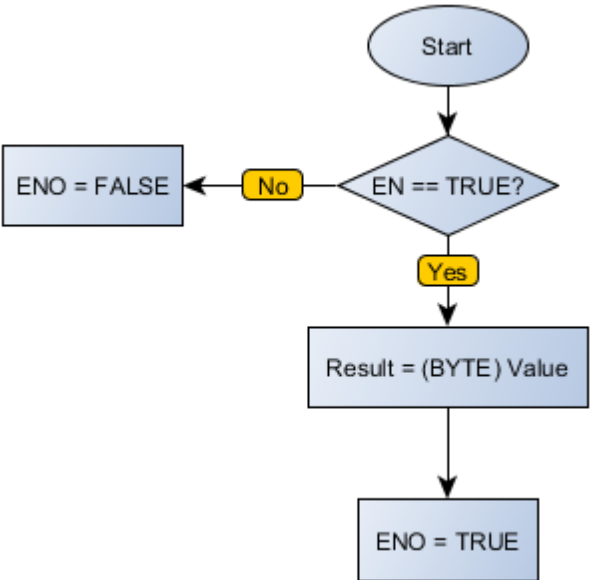
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para BYTE, armazenando em Result.

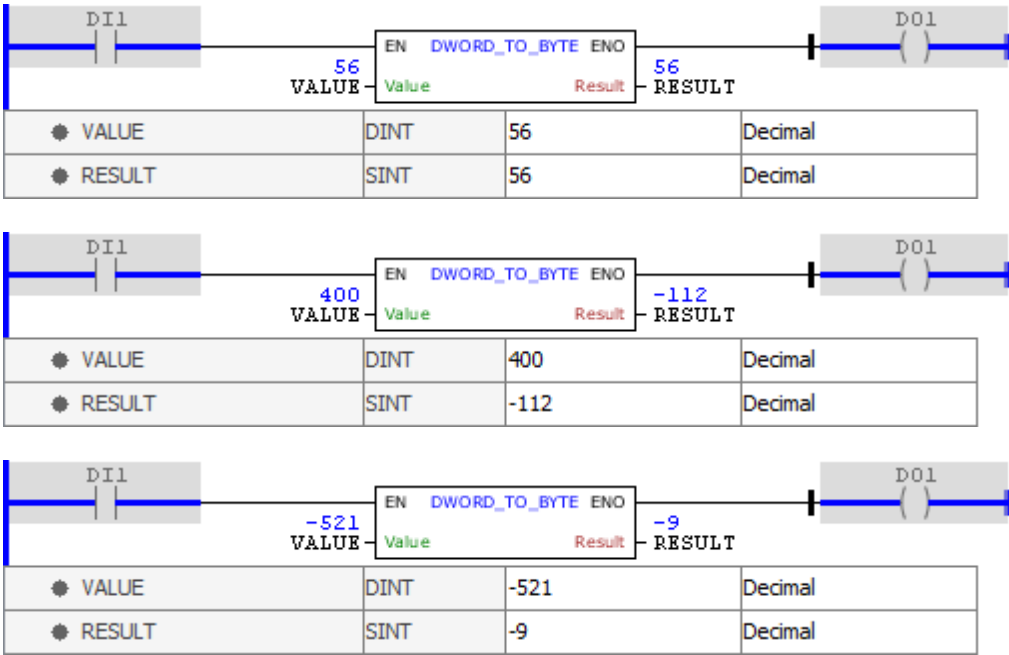
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



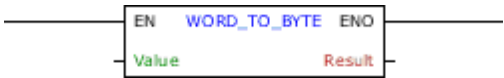
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os oito bits menos significativos.

Bloco que realiza a conversão de um valor WORD para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

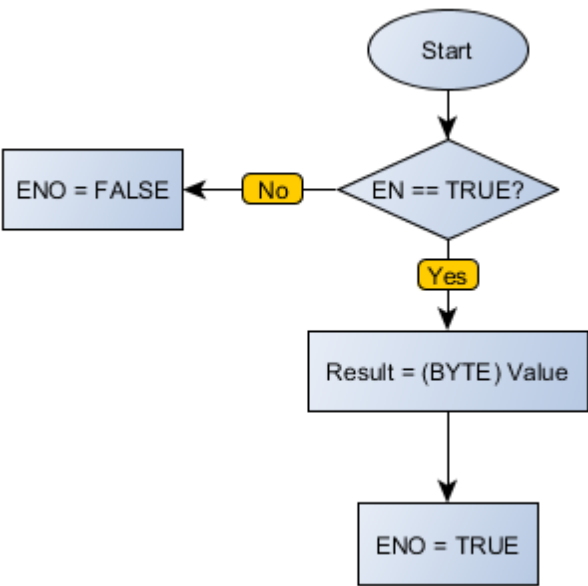
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BYTE, armazenando em Result.

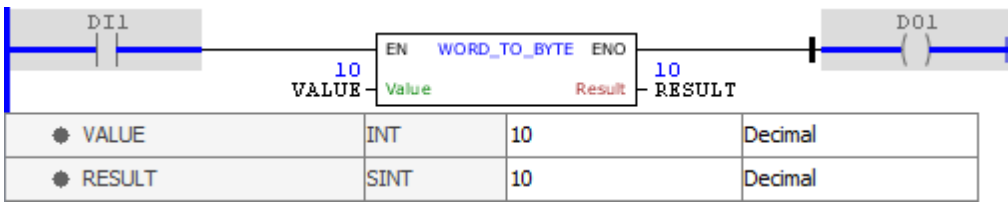
Quando EN possui valor FALSE, Result permanece inalterado.

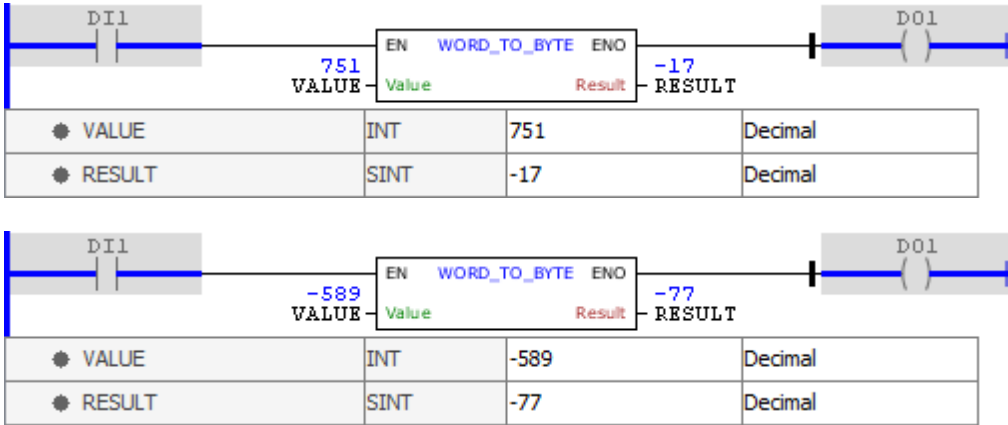
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



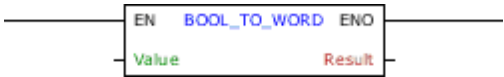


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os oito bits menos significativos.

10.2.7.1.13.4 WORD

Bloco que realiza a conversão de um valor BOOL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

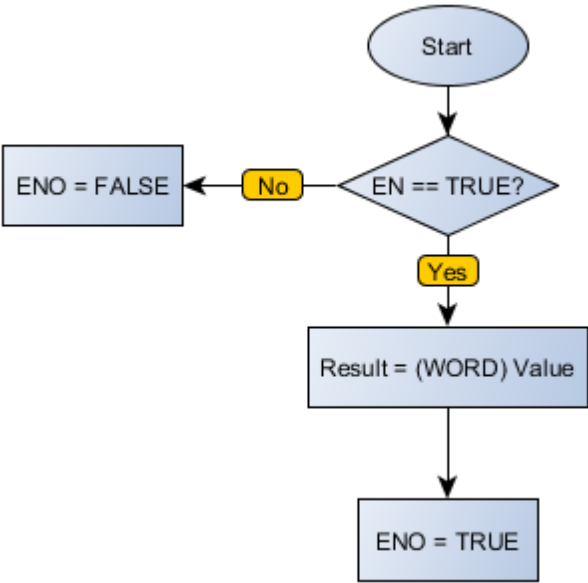
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para WORD, armazenando em Result.

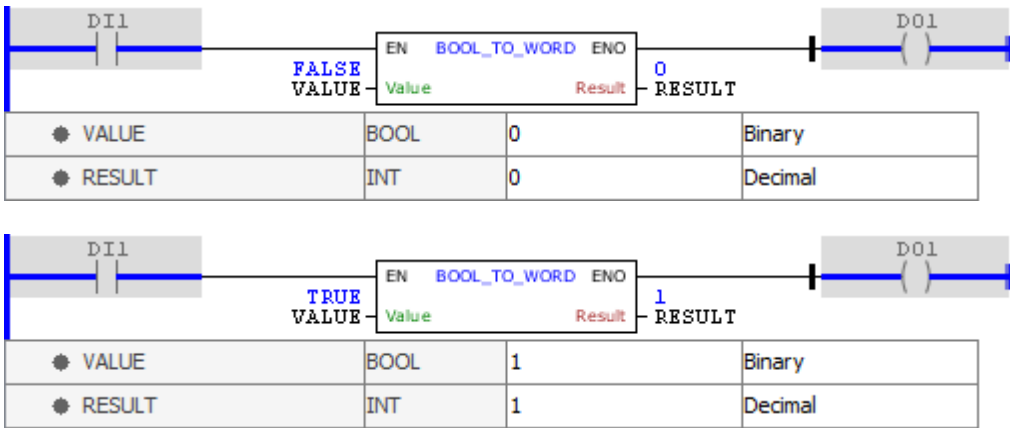
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



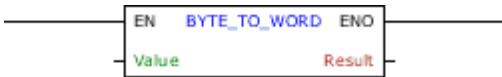
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

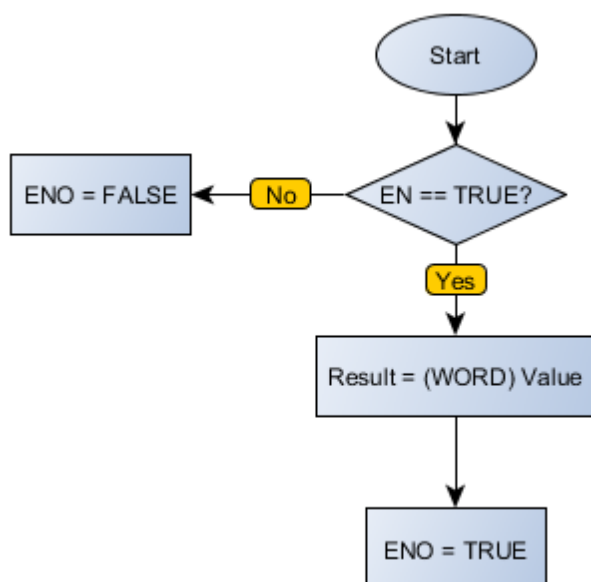
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para WORD, armazenando em Result.

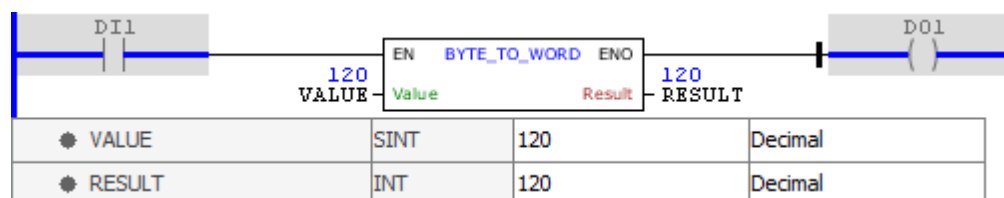
Quando EN possui valor FALSE, Result permanece inalterado.

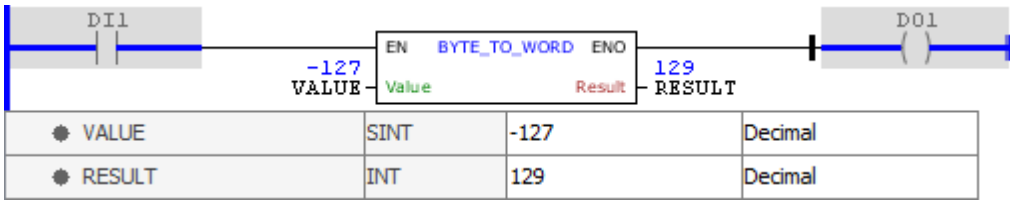
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

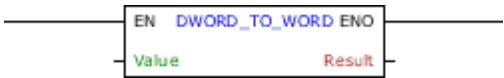




Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor DWORD para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

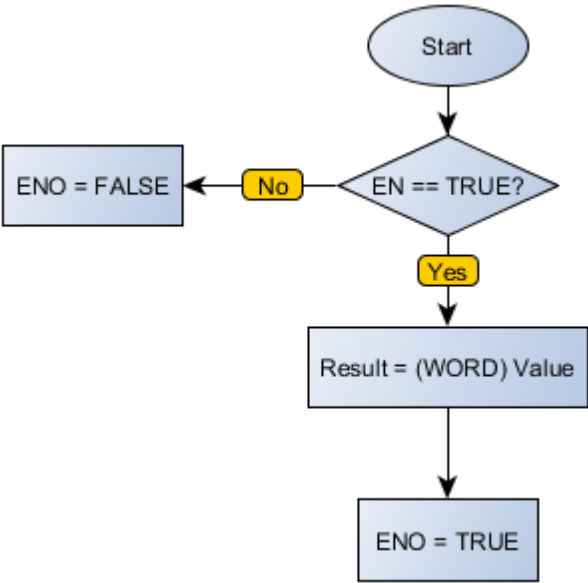
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para WORD, armazenando em Result.

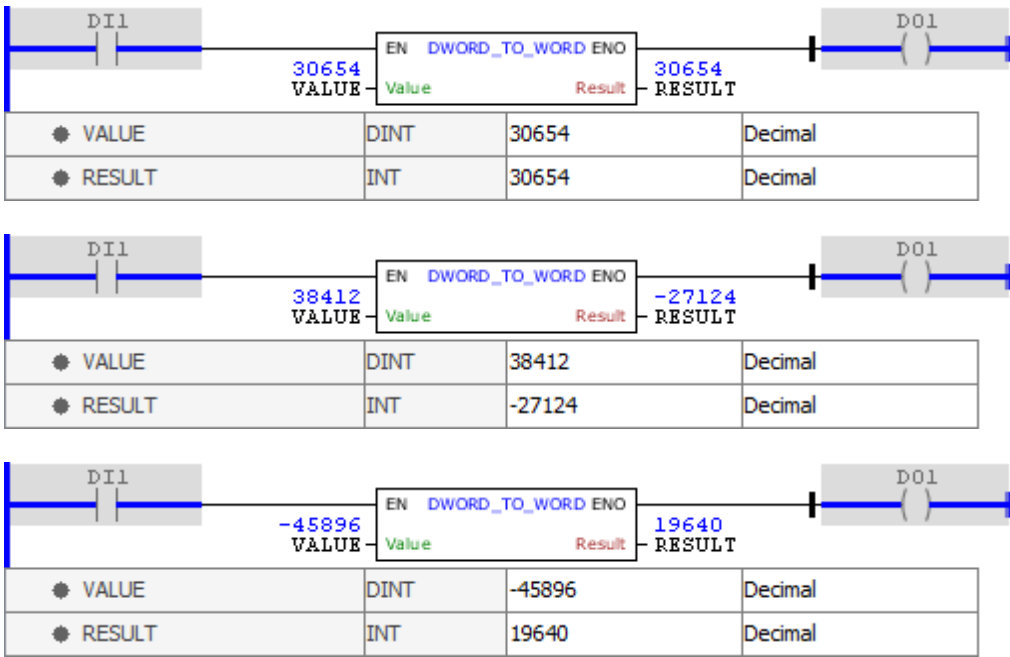
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



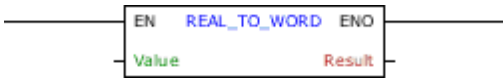
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os dezesseis bits menos significativos.

Bloco que realiza a conversão de um valor REAL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

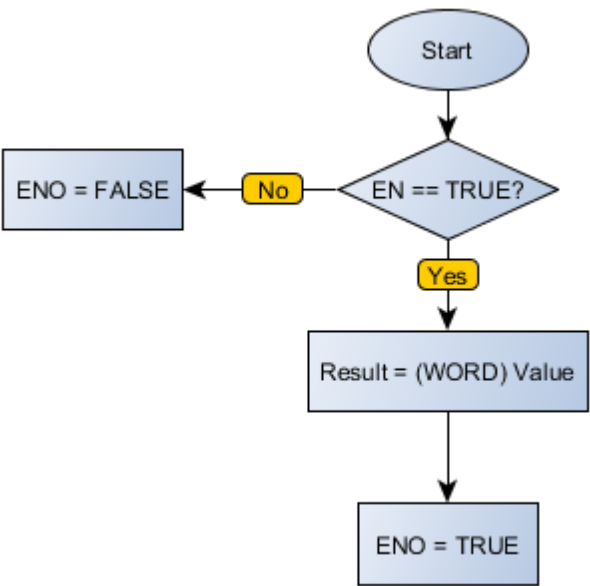
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para WORD, armazenando em Result.

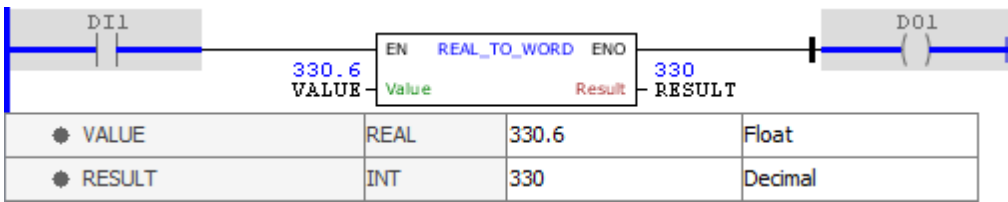
Quando EN possui valor FALSE, Result permanece inalterado.

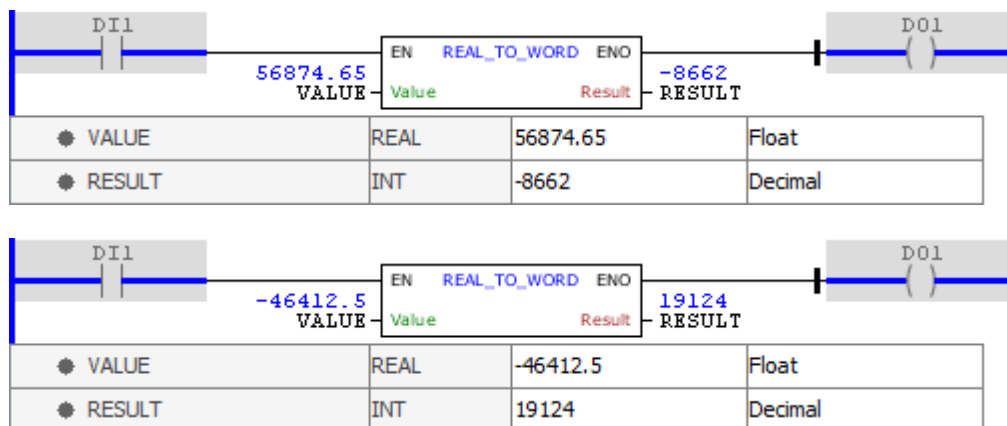
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

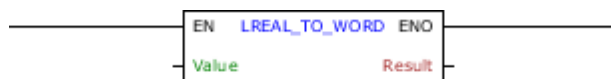




Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os dezesseis bits menos significativos.

Bloco que realiza a conversão de um valor LREAL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

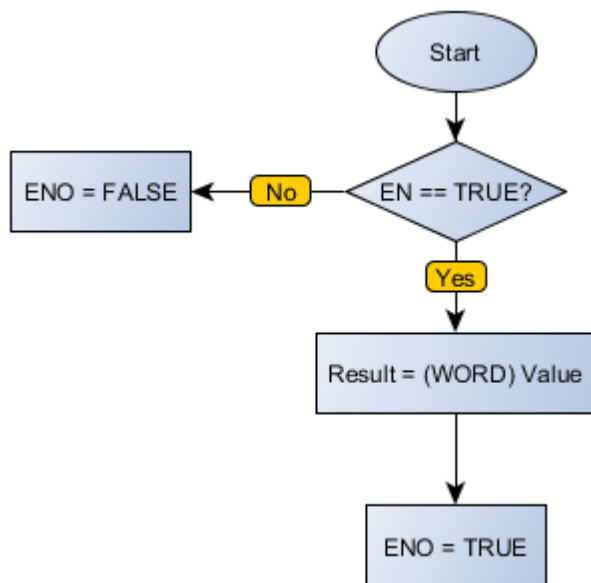
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para WORD, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

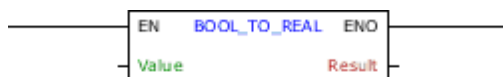
Fluxograma do Bloco



10.2.7.1.13.5 REAL

Bloco que realiza a conversão de um valor BOOL para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

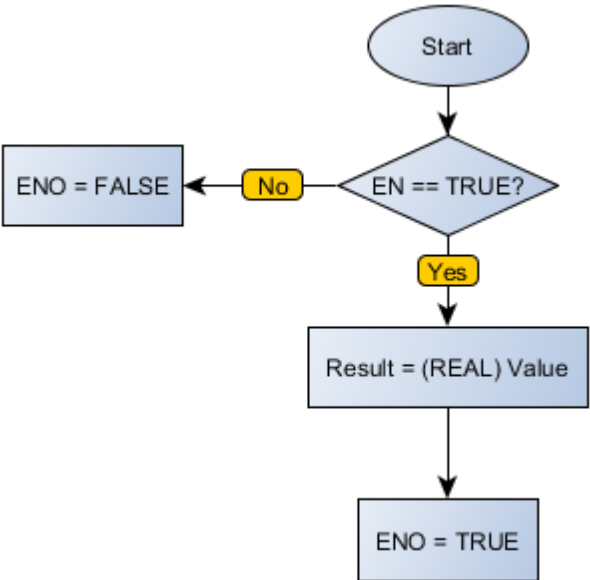
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para REAL, armazenando em Result.

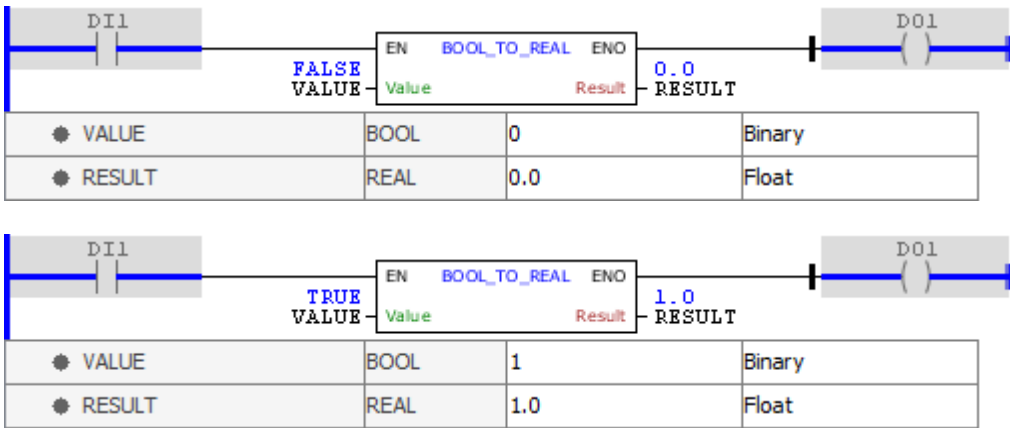
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



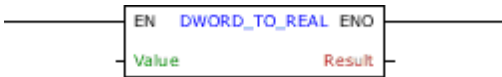
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor DWORD para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

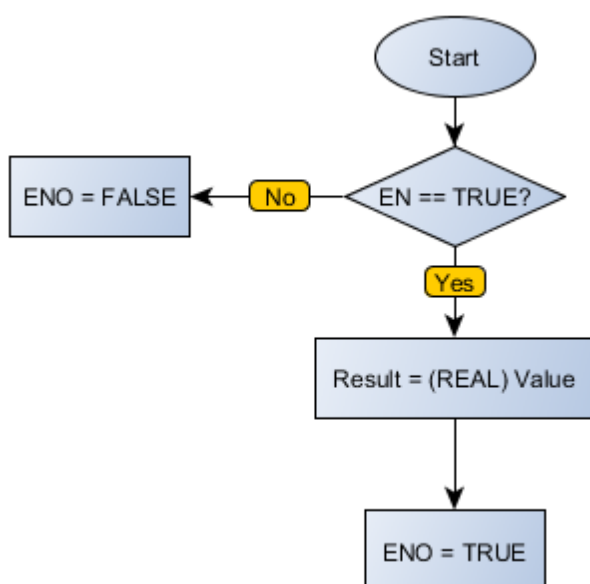
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para REAL, armazenando em Result.

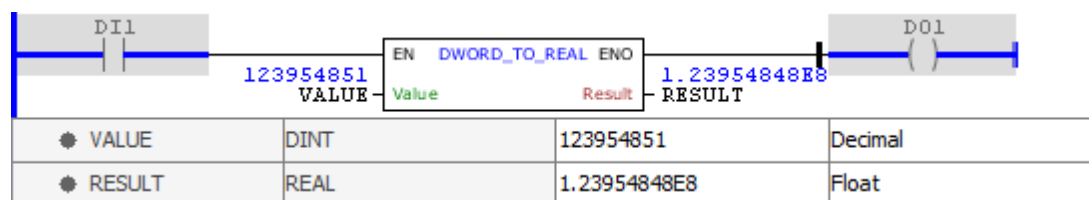
Quando EN possui valor FALSE, Result permanece inalterado.

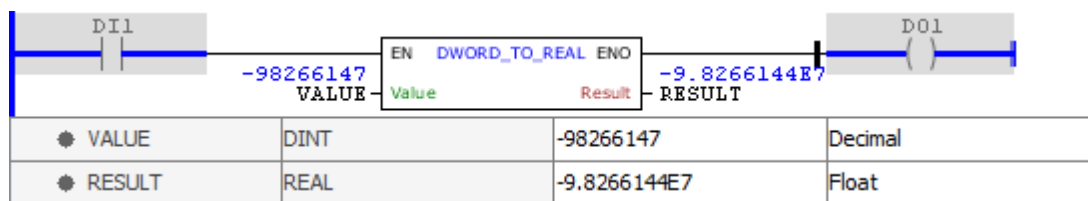
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

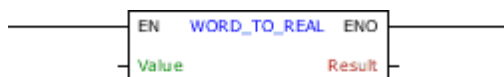




Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

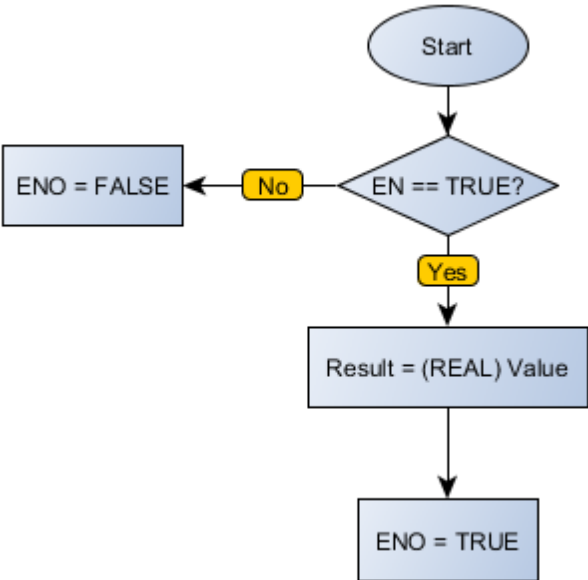
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para REAL, armazenando em Result.

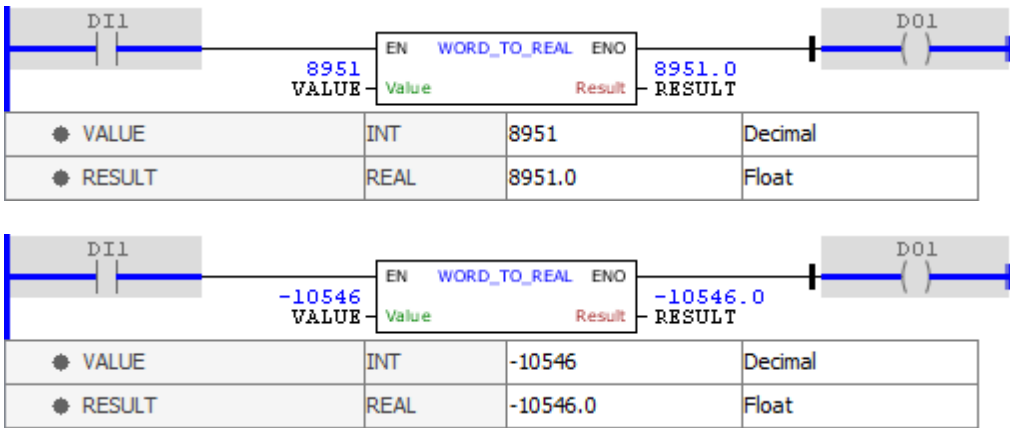
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



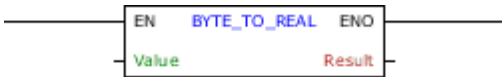
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

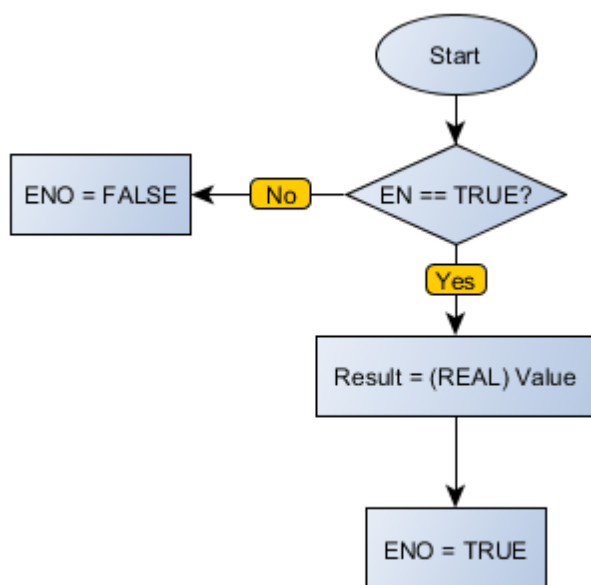
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para REAL, armazenando em Result.

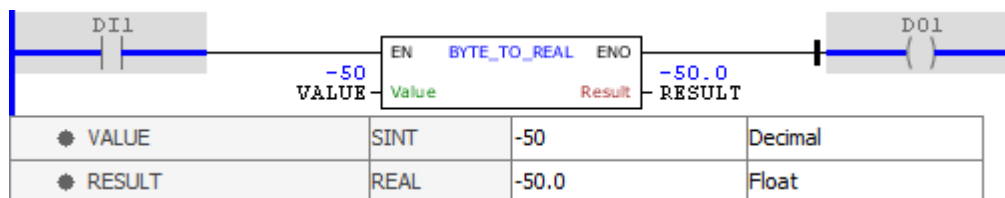
Quando EN possui valor FALSE, Result permanece inalterado.

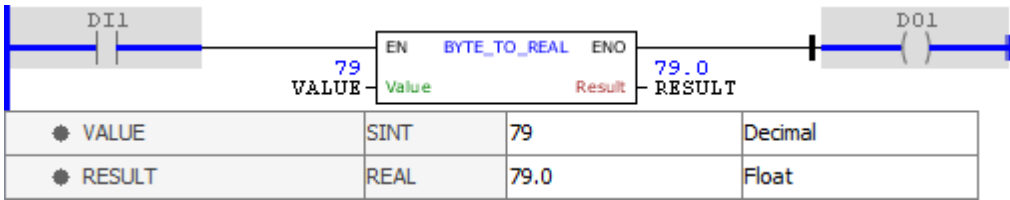
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

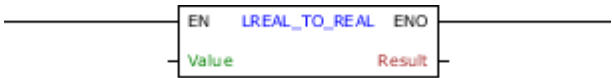




Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor LREAL para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

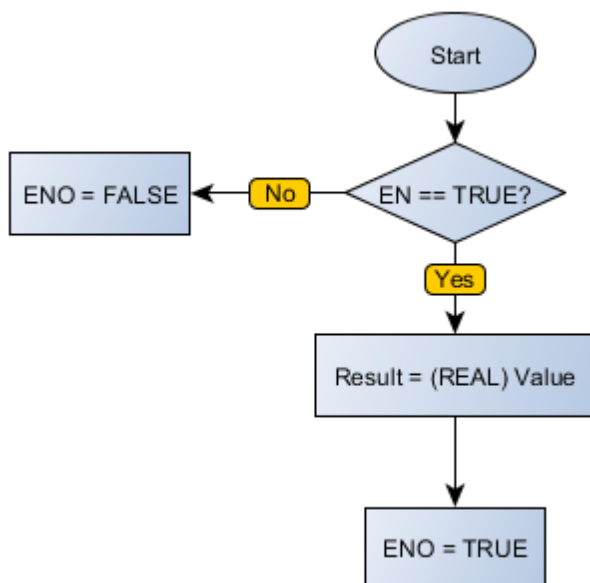
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para REAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

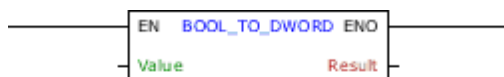
Fluxograma do Bloco



10.2.7.1.13.6 DWORD

Bloco que realiza a conversão de um valor BOOL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

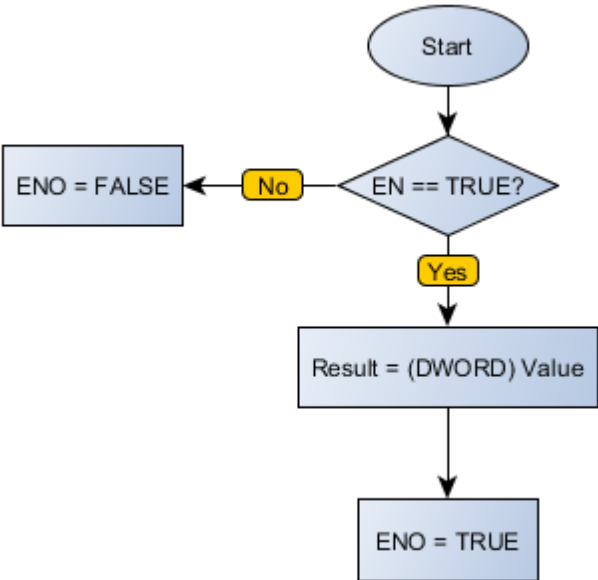
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para DWORD, armazenando em Result.

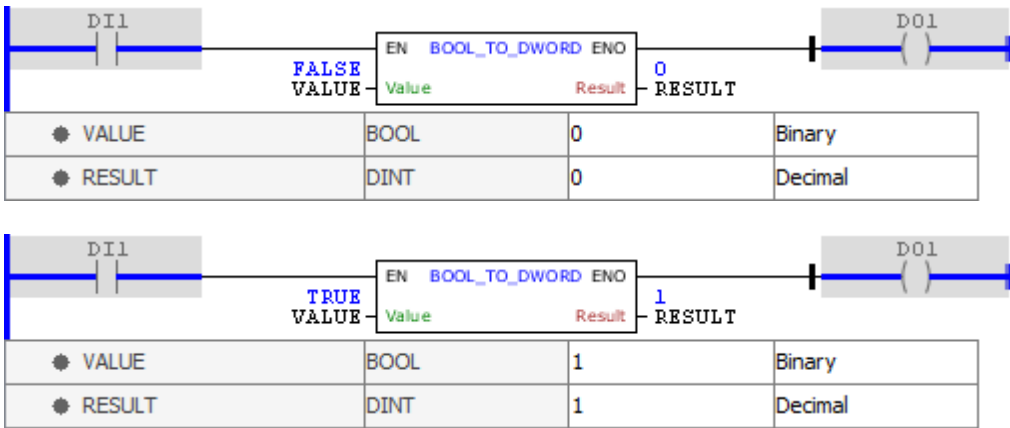
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



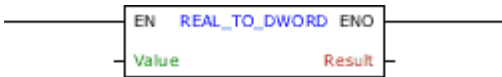
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

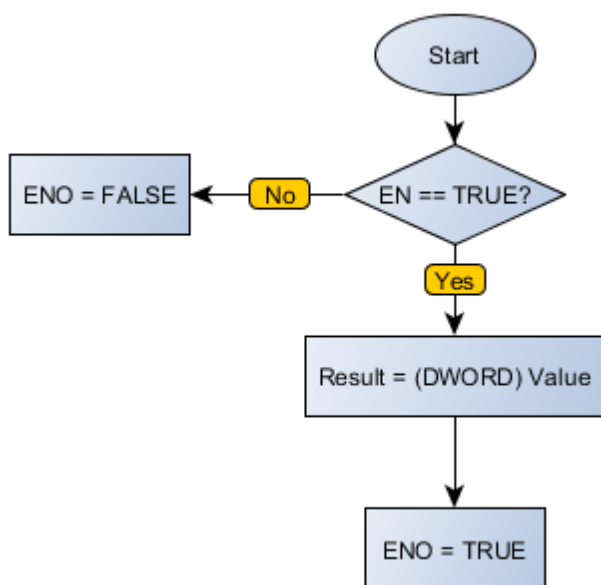
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para DWORD, armazenando em Result.

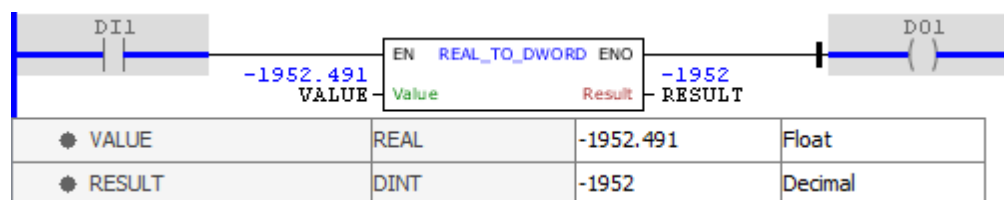
Quando EN possui valor FALSE, Result permanece inalterado.

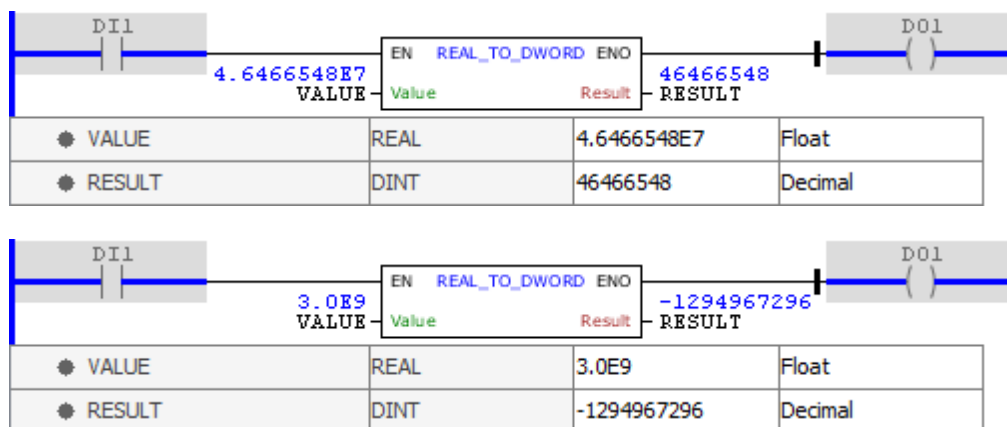
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

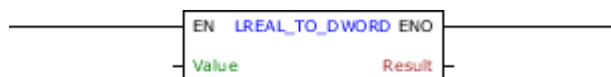




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os trinta e dois bits menos significativos.

Bloco que realiza a conversão de um valor LREAL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

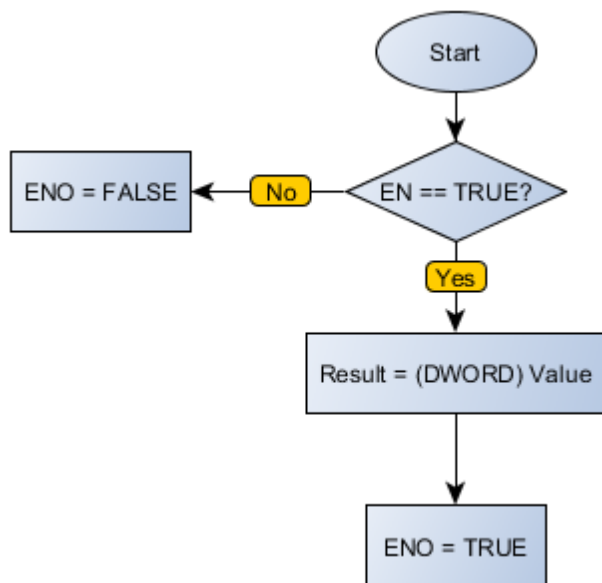
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para DWORD, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

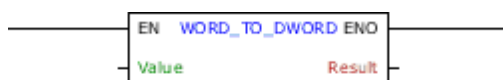
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor WORD para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

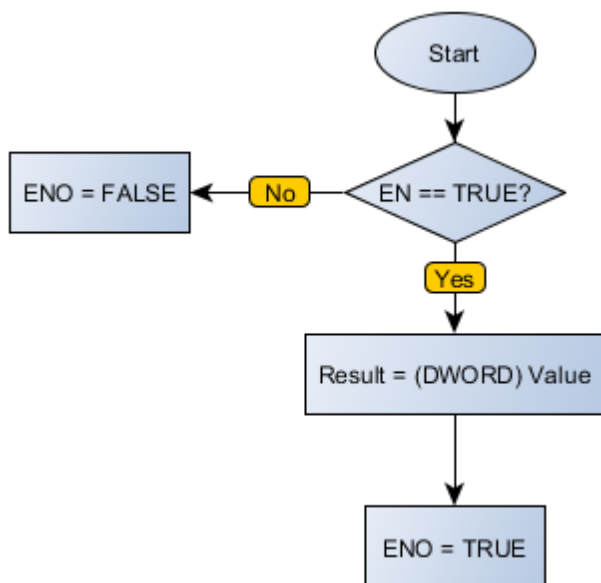
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para DWORD, armazenando em Result.

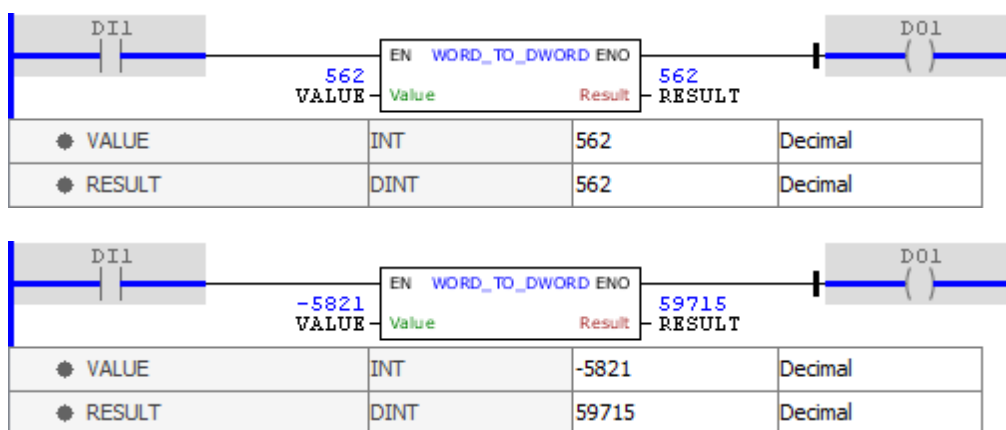
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



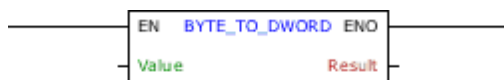
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

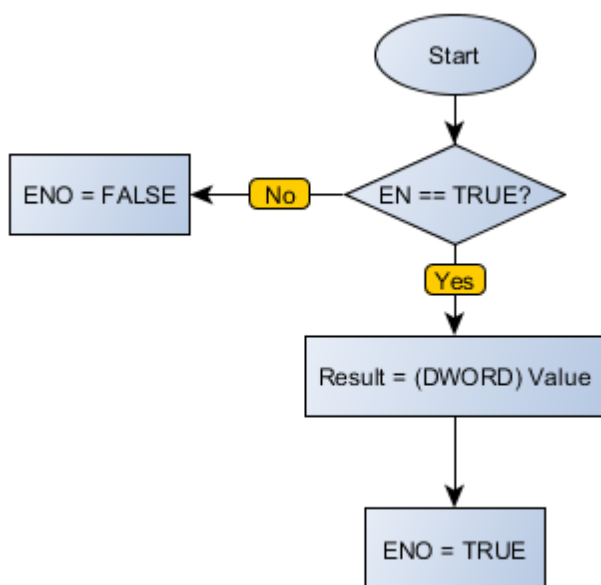
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para DWORD, armazenando em Result.

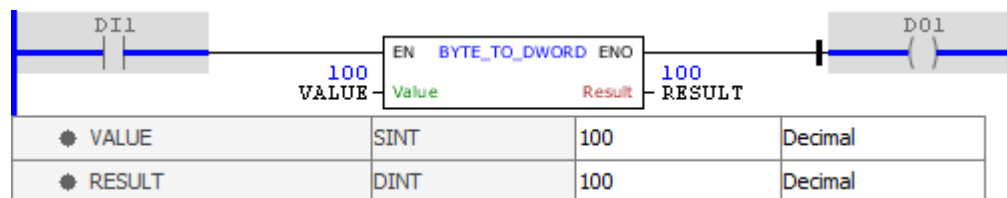
Quando EN possui valor FALSE, Result permanece inalterado.

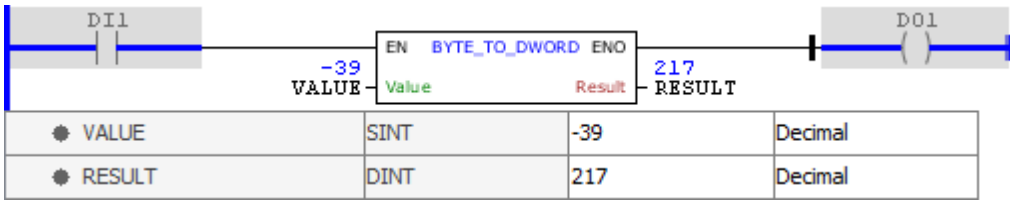
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



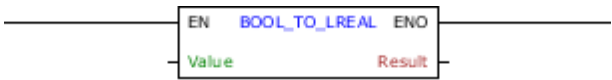


Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.13.7 LREAL

Bloco que realiza a conversão de um valor BOOL para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

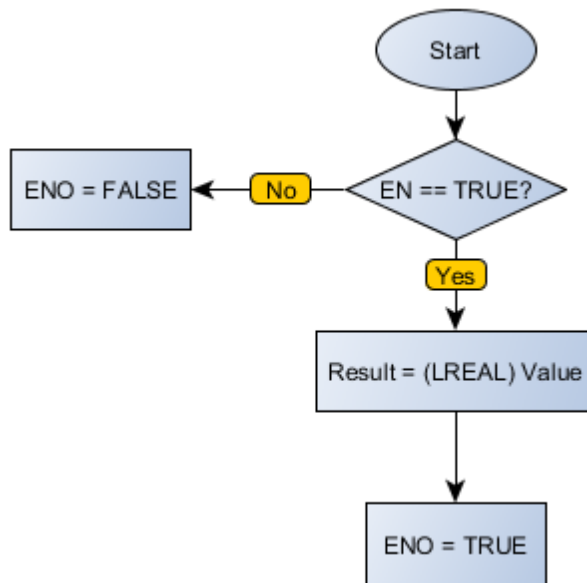
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

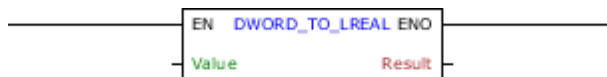
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor DWORD para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

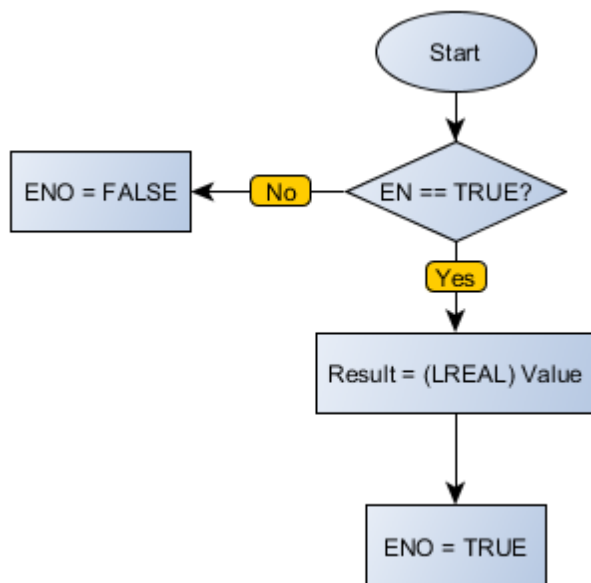
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

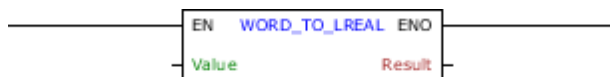
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor WORD para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

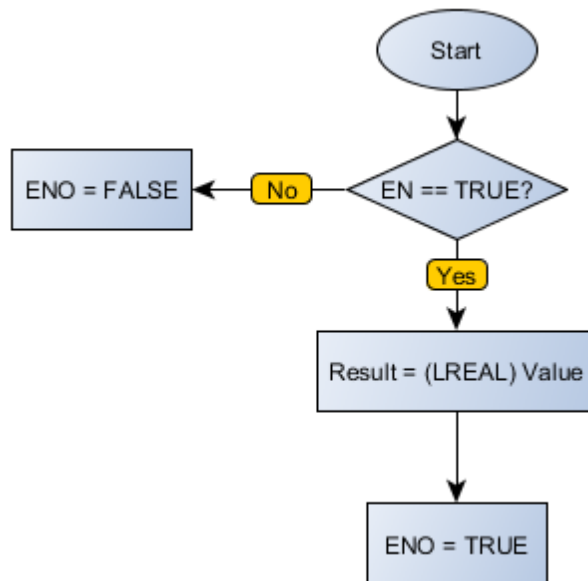
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

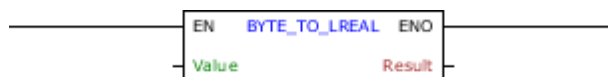
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor BYTE para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

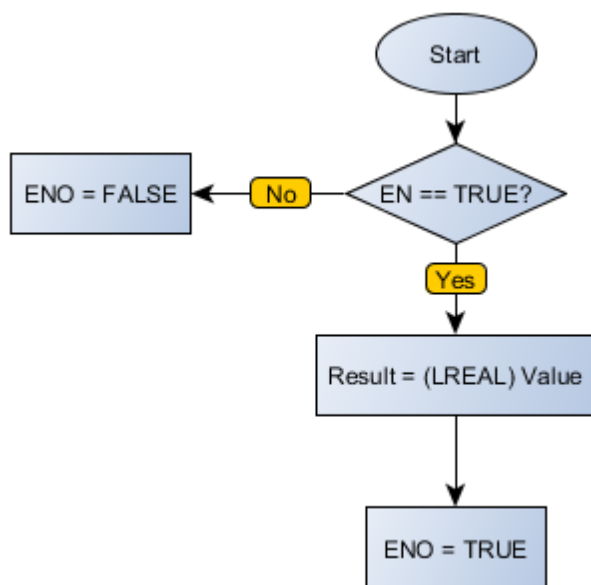
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

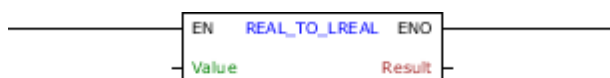
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor REAL para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

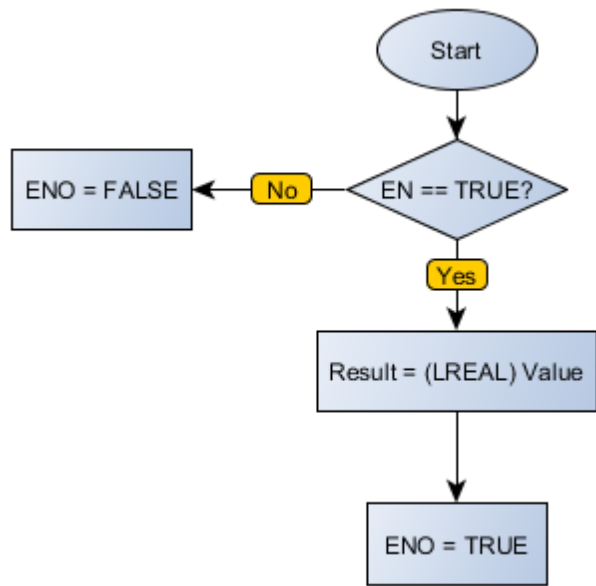
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



10.2.7.1.13.8 Rad-Deg

Bloco que realiza a conversão de um valor em radianos para um valor em graus.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Valor em radianos
VAR_OUTPUT	ENO	B00L	Fim de operação
	Result	REAL	Valor em graus

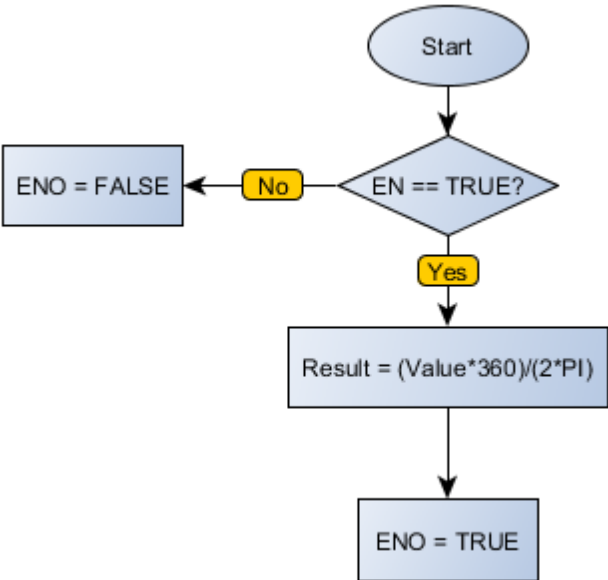
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como em radianos e realiza a conversão para graus, armazenando em Result.

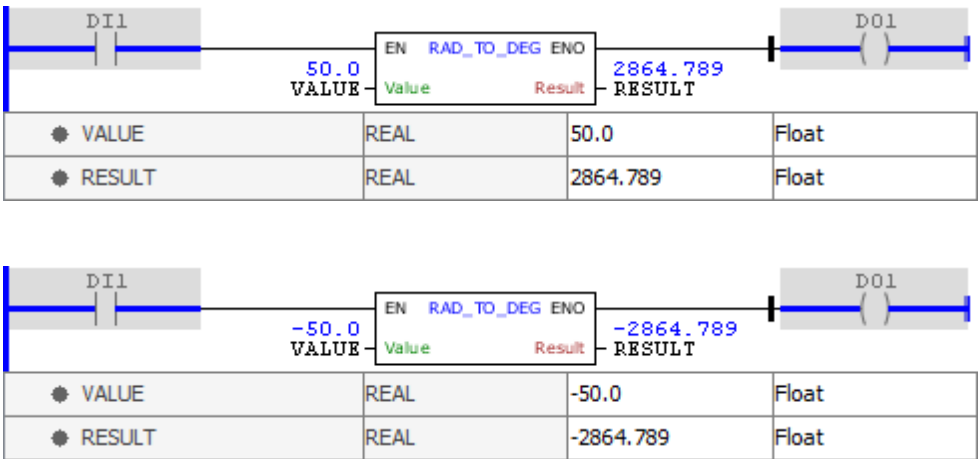
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



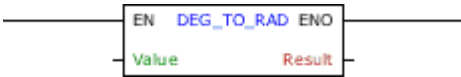
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em radianos, para um valor correspondente em graus, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor em graus para um valor em radianos.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em graus
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em radianos

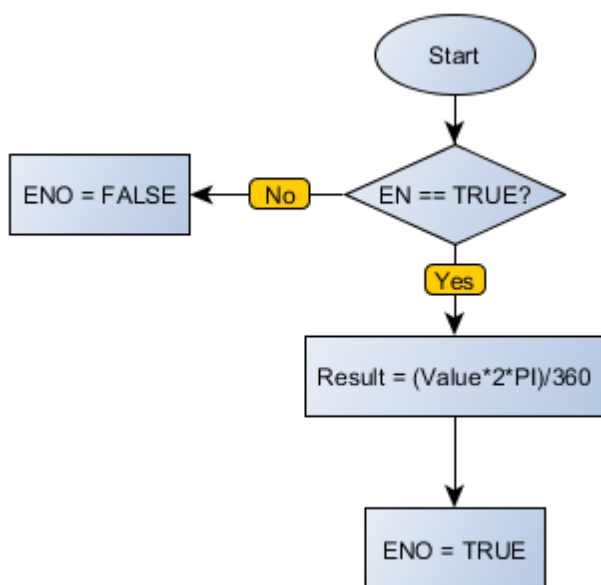
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como em graus e realiza a conversão para radianos, armazenando em Result.

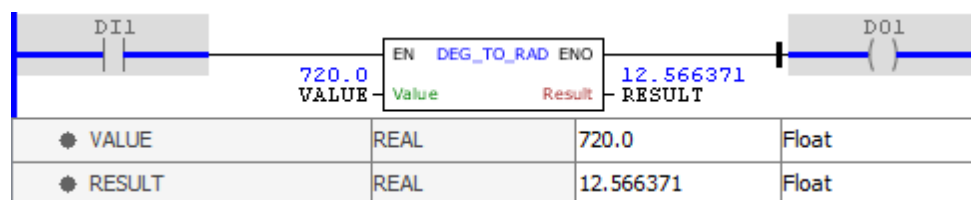
Quando EN possui valor FALSE, Result permanece inalterado.

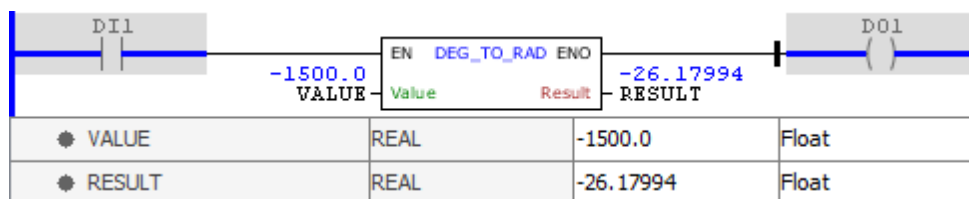
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo





Os exemplos acima realizam a conversão da variável VALUE, em graus, para um valor correspondente em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14 Data Transfer

10.2.7.1.14.1 STORE

Bloco que realiza o armazenamento direto de dados provenientes de uma fonte em um destino.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SRC	BYTE USINT SINT WORD UINT INT DWORD DINT REAL	Fonte de dados
VAR_OUTPUT	ENO	BOOL	Fim de operação
	DST	BYTE USINT SINT WORD UINT INT DWORD DINT REAL	Destino de dados

Funcionamento

Este bloco, quando tem um valor TRUE em EN, armazena o conteúdo de SRC em DST.



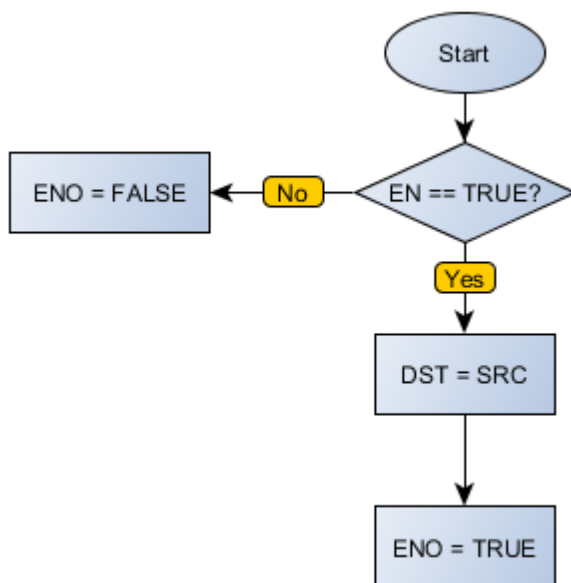
NOTA!

SRC e DST devem possuir tipos de dados de mesmo tamanho.

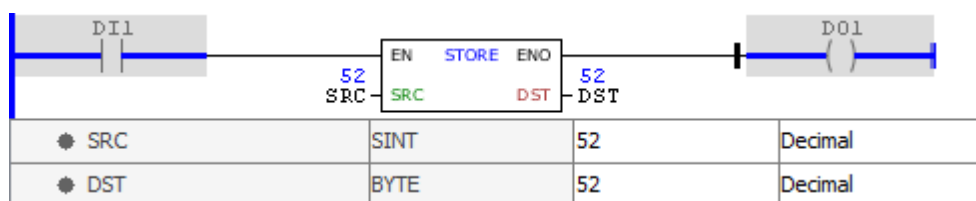
Quando EN possui valor FALSE, DST permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

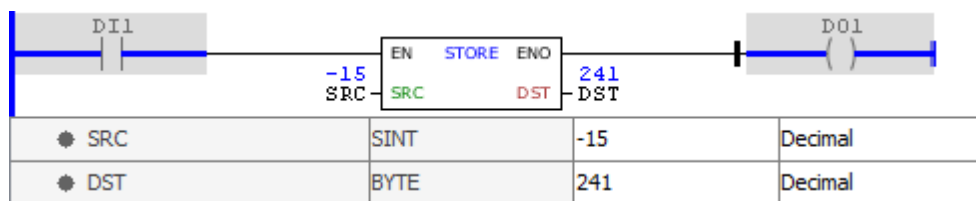
Fluxograma do Bloco



Exemplo



O exemplo acima armazena o valor da variável SRC em DST. O bloco finaliza com sucesso e a saída ENO é ativada.



O exemplo acima armazena o valor da variável SRC em DST. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário é mantido entre variáveis de tipos diferentes.

10.2.7.1.14.2 ILOADBOOL

Bloco que carrega indiretamente o valor de um bit em um endereço de variável global.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Bit	BYTE USINT SINT	Posição do bit a ser verificado
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Value	BOOL	Valor do bit selecionado pelos argumentos de entrada

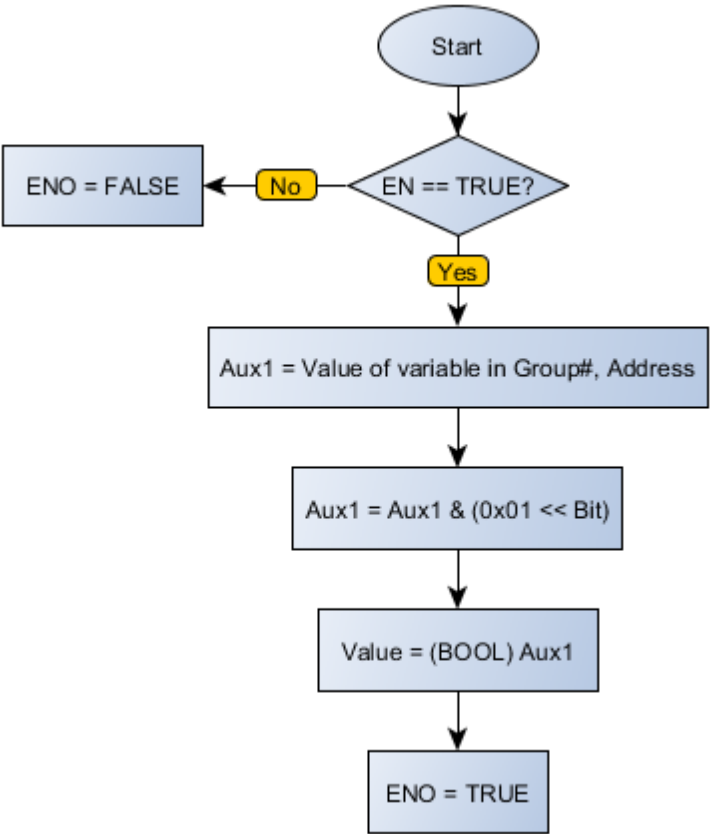
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega, em Value, o conteúdo do Bit da variável Address pertencente ao grupo Group#.

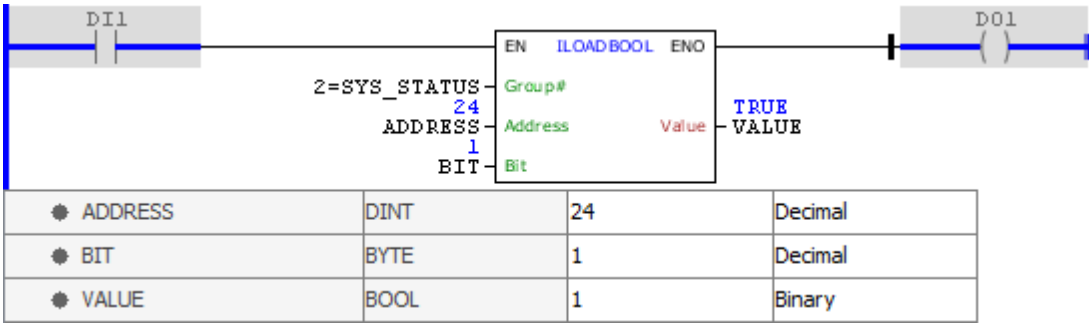
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima carrega o valor do bit 1 do endereço 24 do grupo 2 (GLOBAL_SYSTEM %S) para a variável VALUE. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14.3 ILOAD

Bloco que carrega indiretamente o valor de uma variável e transfere para Value.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	DataType#	BYTE	Tipo de dado da variável selecionada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Value	Conforme selecionado em DataType#	Valor da variável selecionada

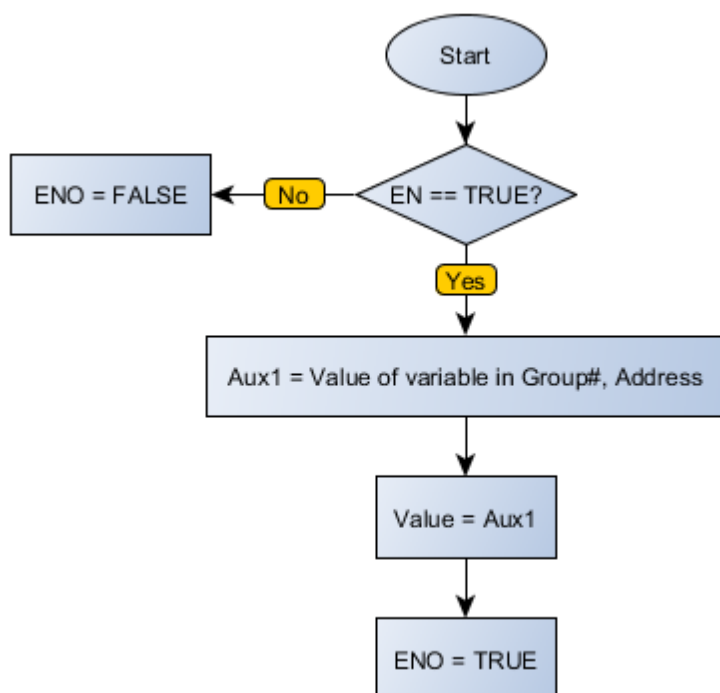
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega, em Value, o conteúdo da variável Address pertencente ao grupo Group#, segundo o DataType# escolhido.

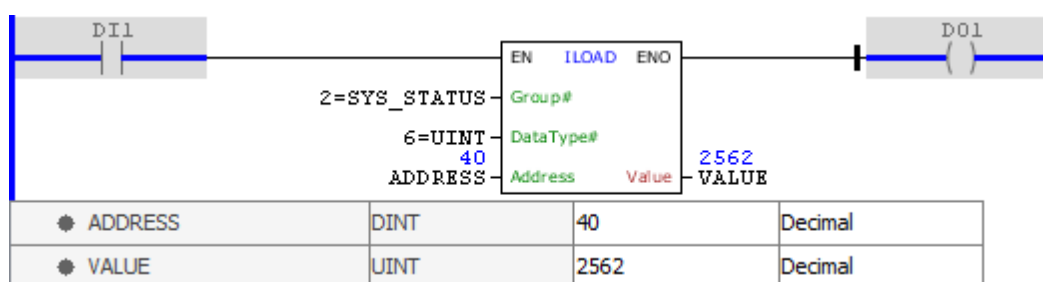
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima carrega o valor do endereço 40 do grupo 2 (GLOBAL_SYSTEM %S) em formato UINT para a variável VALUE. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14.4 ISTOREBOOL

Bloco que armazena indiretamente o valor de Value em um bit em um endereço de variável global.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Bit	BYTE USINT SINT	Posição do bit a ser modificado
	Value	BOOL	Novo valor do bit selecionado
VAR_OUTPUT	ENO	BOOL	Fim de operação

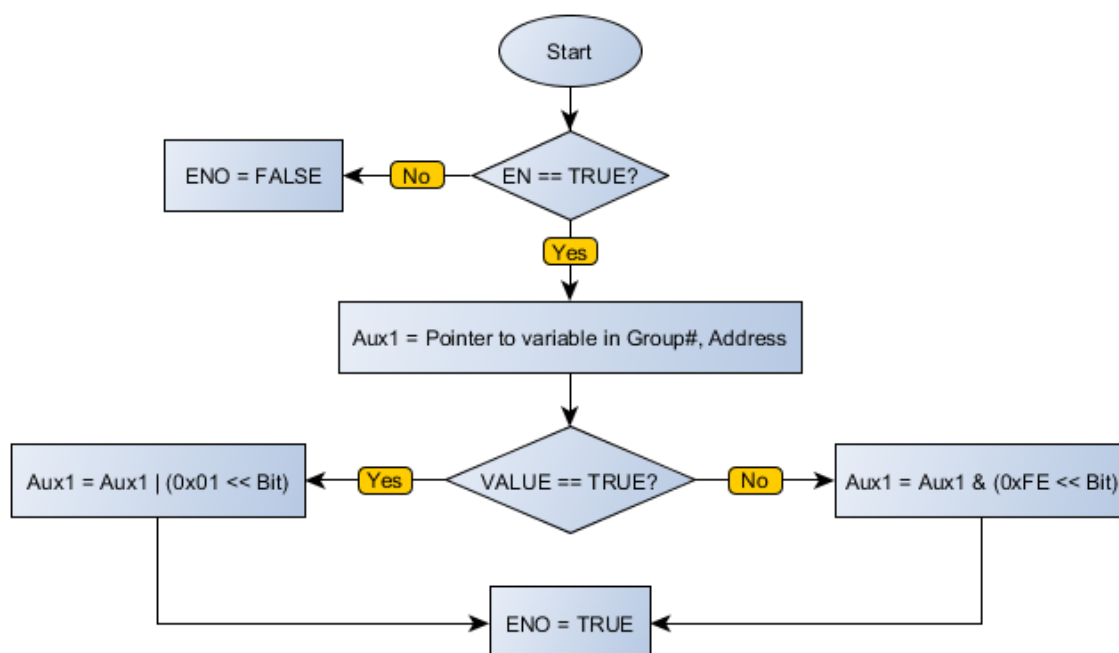
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega o valor de Value no conteúdo do Bit da variável Address pertencente ao grupo Group#.

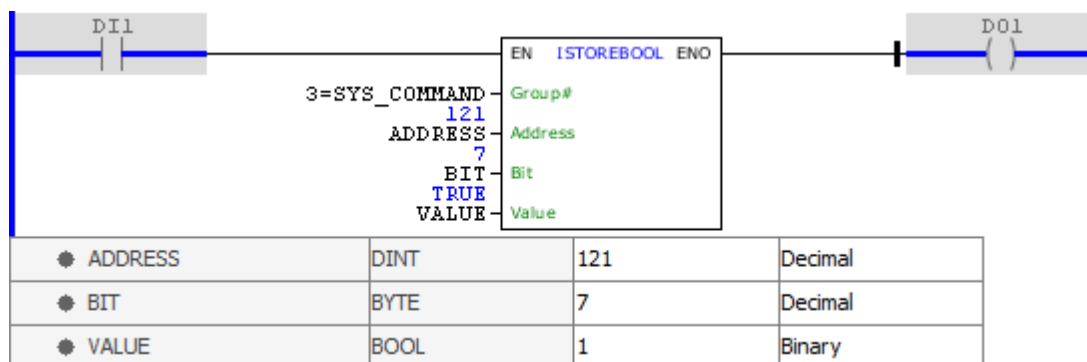
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

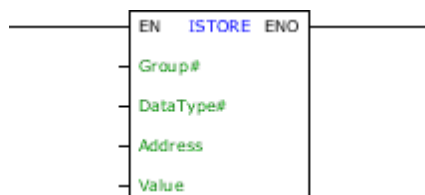


O exemplo acima armazena o valor de VALUE no bit 7 do endereço 121 do grupo 3 (GLOBAL_SYSTEM %C). O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14.5 ISTORE

Bloco que armazena indiretamente o valor de Value em uma variável.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	DataType#	BYTE	Tipo de dado da variável selecionada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Value	Conforme selecionado em DataType#	Valor a ser escrito na variável selecionada
VAR_OUTPUT	ENO	BOOL	Fim de operação

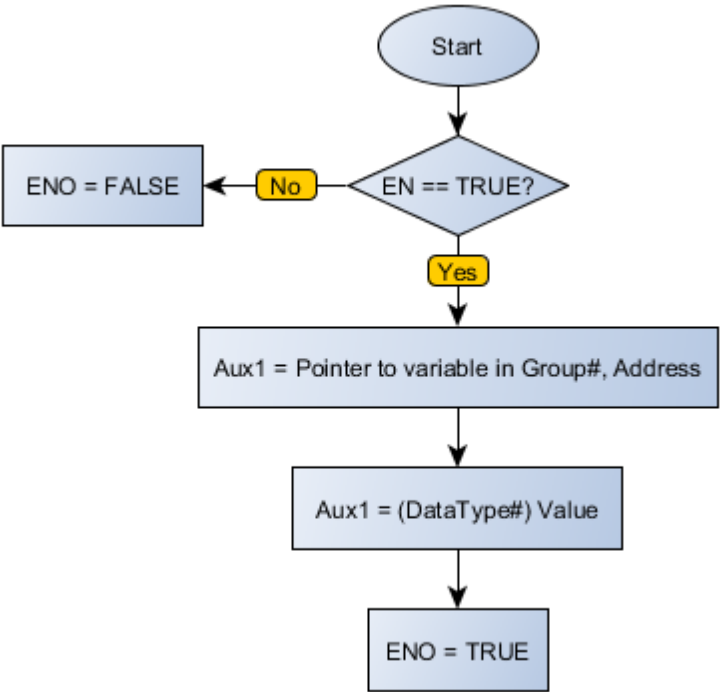
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega o valor de Value no conteúdo da variável Address pertencente ao grupo Group#, segundo o DataType# escolhido.

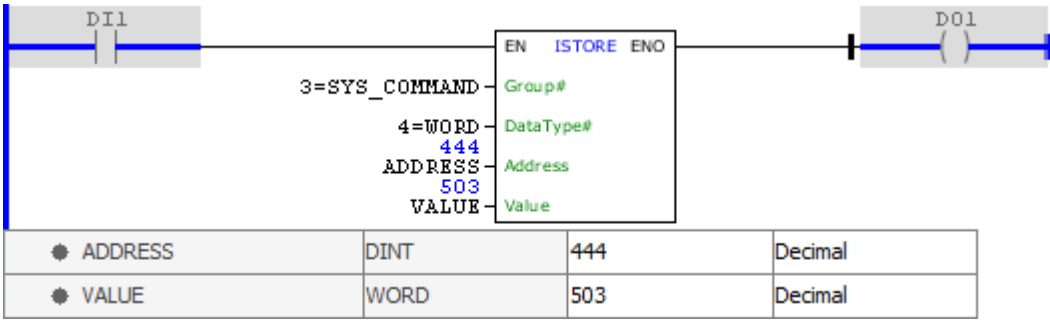
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

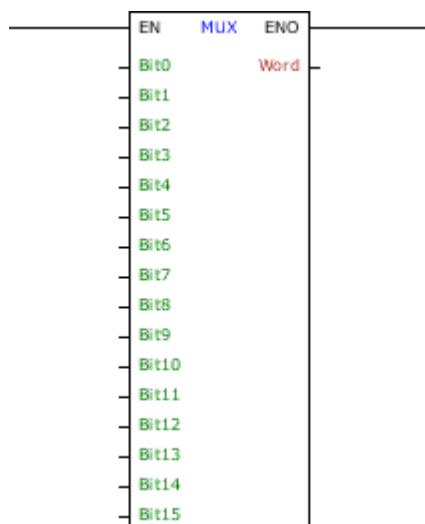


O exemplo acima armazena o valor de VALUE em formato WORD no endereço 444 do grupo 3 (GLOBAL_SYSTEM %C). O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14.6 MUX

Bloco que cria uma nova variável WORD a partir da concatenação de 15 variáveis BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Bit0 - Bit15	BOOL	Bit da posição correspondente na nova palavra
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Word	WORD UINT INT	Nova palavra formada a partir dos bits de entrada

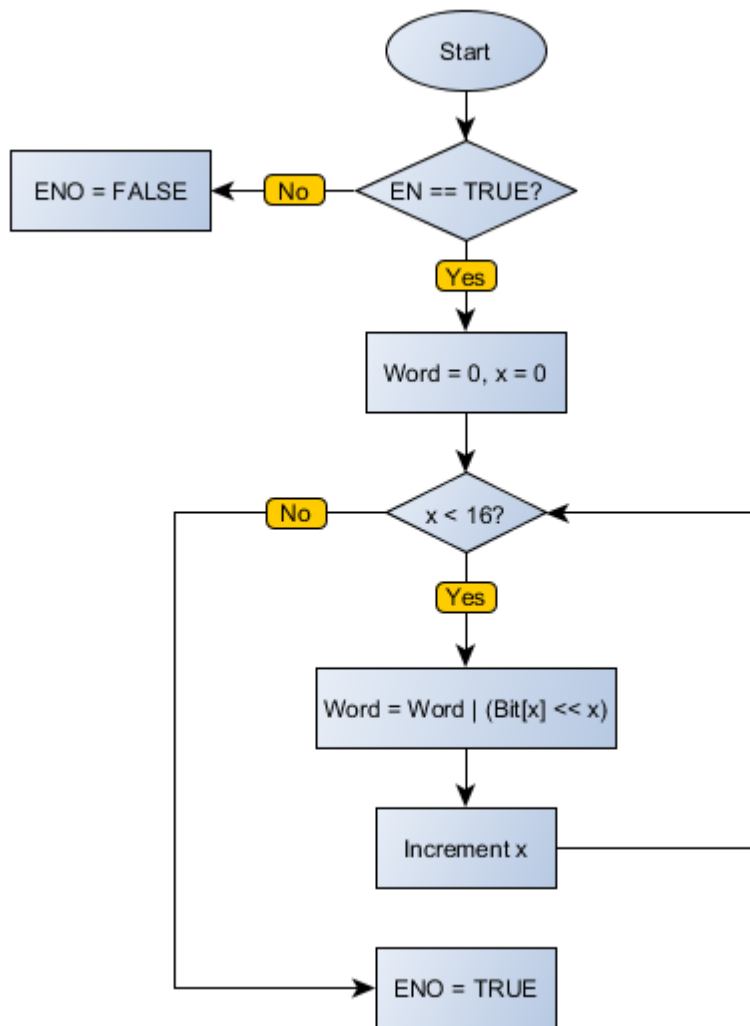
Funcionamento

Este bloco, quando tem um valor TRUE em EN, concatena os valores booleanos das variáveis de entrada e armazena este valor na variável Word. O Bit0 corresponde ao LSB (bit menos significativo) e Bit15 corresponde ao MSB (bit mais significativo).

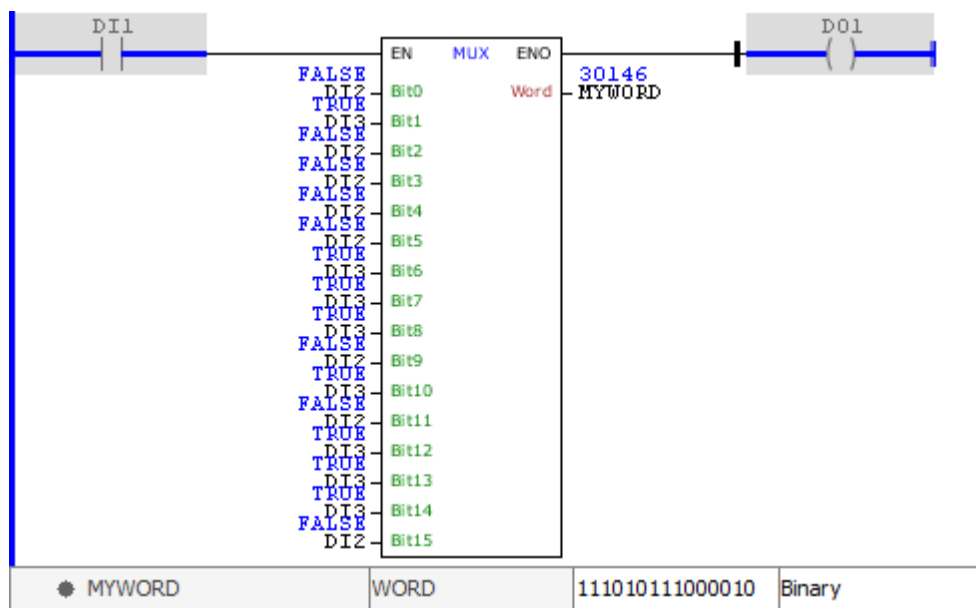
Quando EN possui valor FALSE, Word permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

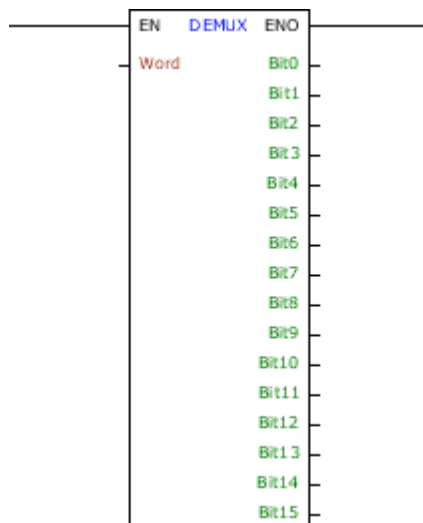


O exemplo acima concatena os valores booleanos dos bits de entrada do bloco, formando a palavra de saída armazenada em MYWORD. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14.7 DEMUX

Bloco que cria 15 novas variáveis BOOL a partir da decomposição de uma variável WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Word	WORD UINT INT	Variável de entrada de 15 bits
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Bit0 - Bit15	BOOL	Bit da posição correspondente de Word

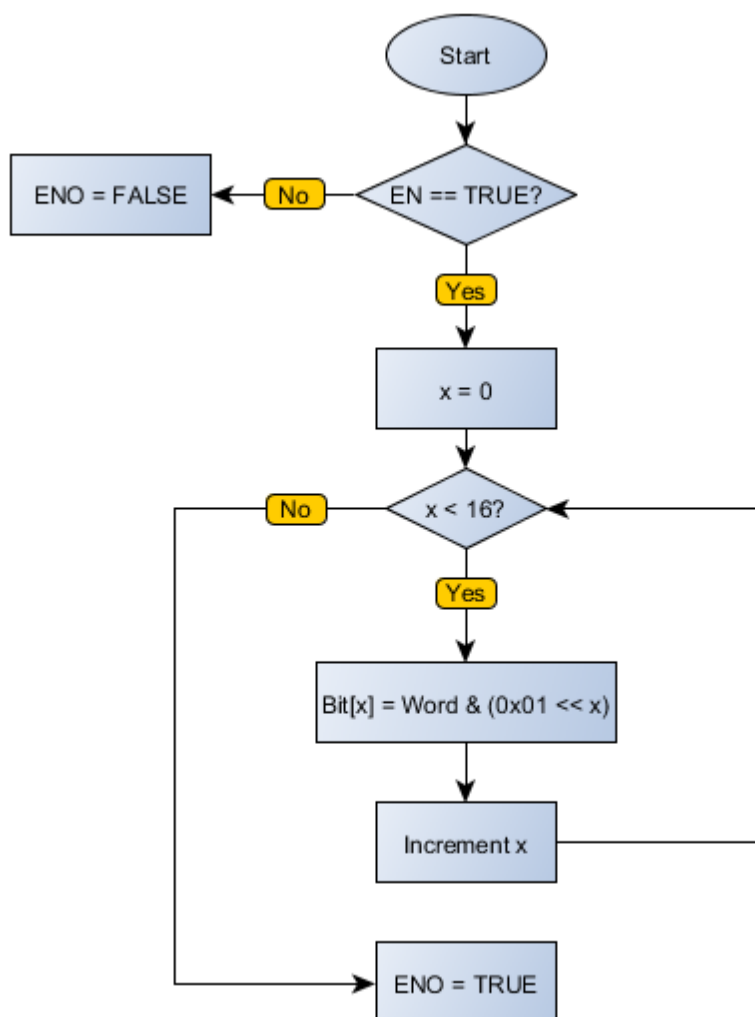
Funcionamento

Este bloco, quando tem um valor TRUE em EN, decompõe a variável de entrada Word em 15 valores booleanos armazenados nas variáveis Bit0 a Bit15. O Bit0 corresponde ao LSB (bit menos significativo) e Bit15 corresponde ao MSB (bit mais significativo).

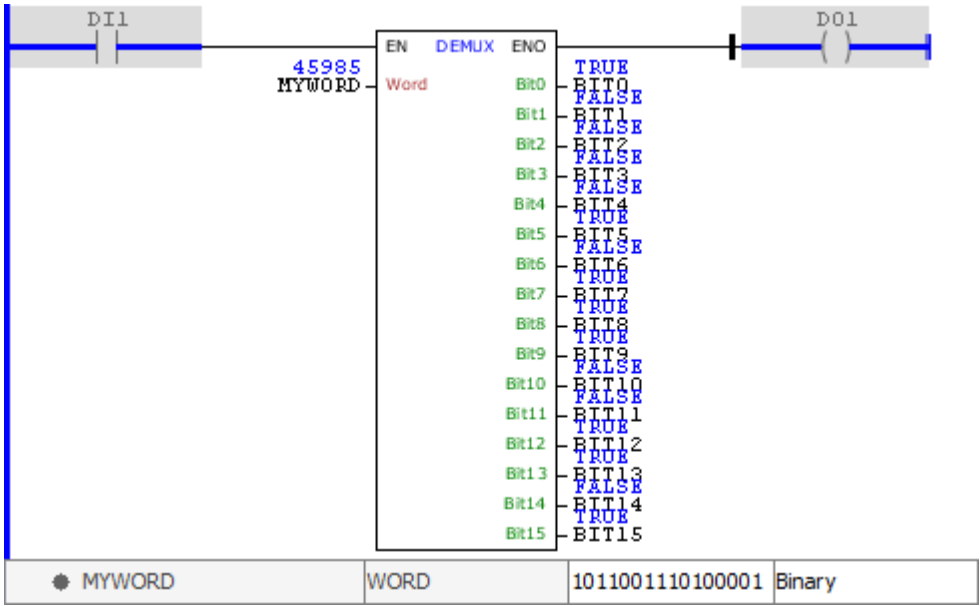
Quando EN possui valor FALSE, as variáveis de saída permanecem inalteradas.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima decompõe o valor de MYWORD em valores booleanos, sendo estes armazenados nas variáveis de saída BIT0 a BIT15. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14.8 SEL

Bloco que replica para a saída o valor de uma das variáveis de entrada (Value0 ou Value1) de acordo com a seleção de Selector.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Selector	BOOL	Variável que seleciona a entrada
	Value0	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Entrada multiplexada número 1
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Entrada multiplexada número 2
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saída selecionado

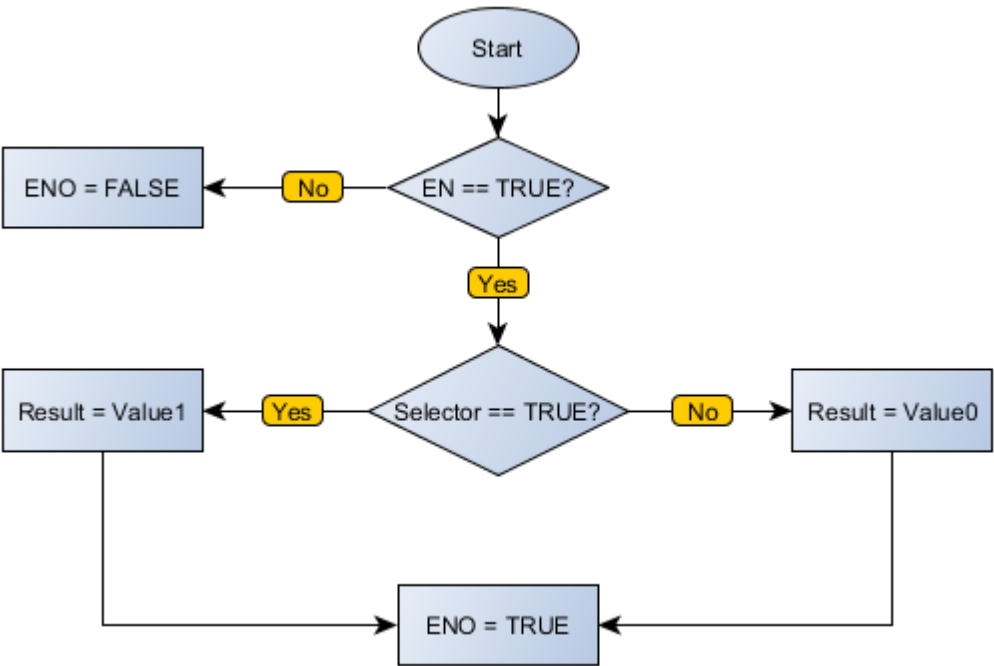
Funcionamento

Este bloco, quando tem um valor TRUE em EN, replica para a variável Result o valor de Value0 caso Selector seja FALSE ou o valor de Value1 caso Selector seja TRUE.

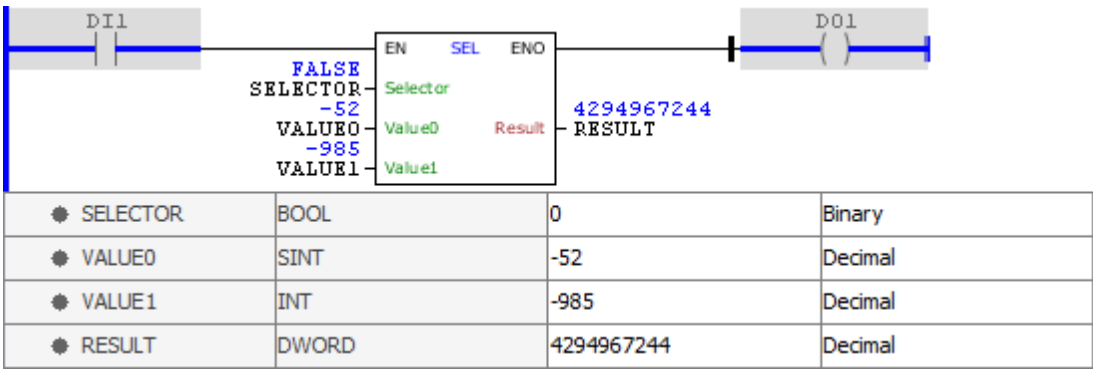
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco

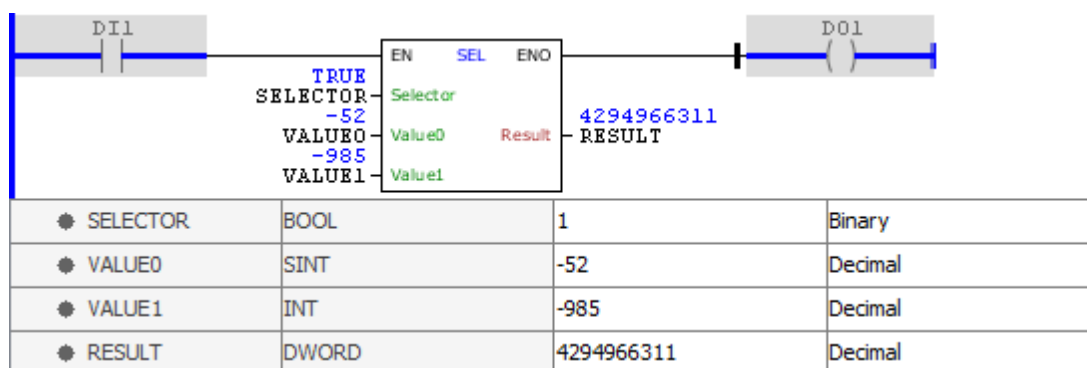


Exemplo



O exemplo acima utiliza a varável SELECTOR como seletor do canal de multiplexação. Quando ela está em nível FALSE, a saída RESULT recebe o valor de VALUE0. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário foi mantido, apesar de a representação decimal

estar corrompida, dado que DWORD não aceita valores negativos.



O exemplo acima utiliza a variável SELECTOR como seletor do canal de multiplexação. Quando ela está em nível TRUE, a saída RESULT recebe o valor de VALUE1. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário foi mantido, apesar de a representação decimal estar corrompida, dado que DWORD não aceita valores negativos.

10.2.7.1.14.9 SWAP

Bloco que realiza uma permuta entre os bytes ímpares e os bytes pares consecutivos em Value e envia o valor para Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT DWORD UDINT DINT	Variável de entrada a ser permutada
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saída

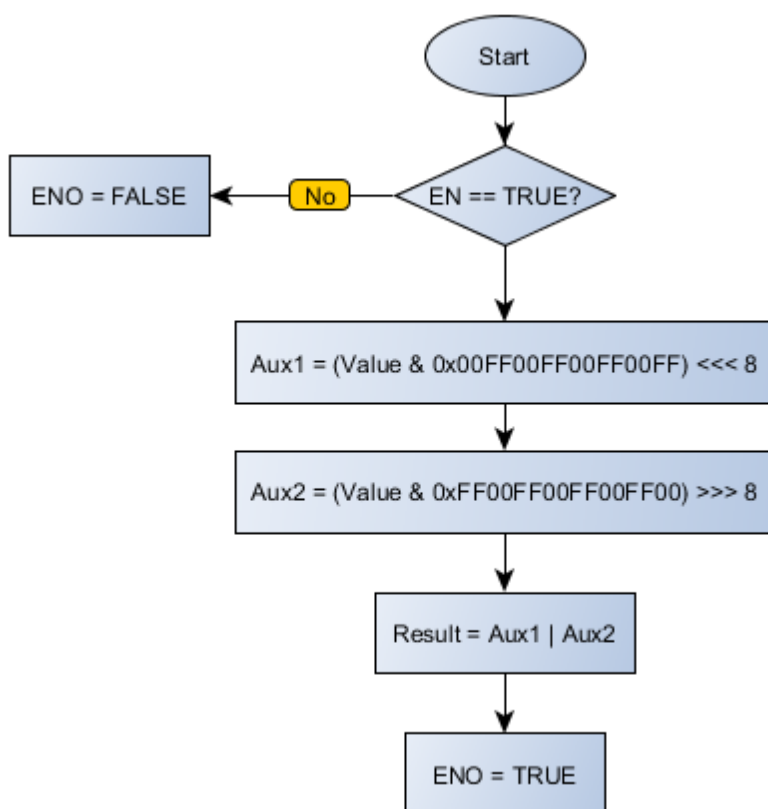
Funcionamento

Este bloco, quando tem um valor TRUE em EN, troca os valores entre os bytes ímpares (1, 3, 5 e 7) e os bytes pares consecutivos (2, 4, 6 e 8, respectivamente) da variável Value, armazenando o resultado em Result.

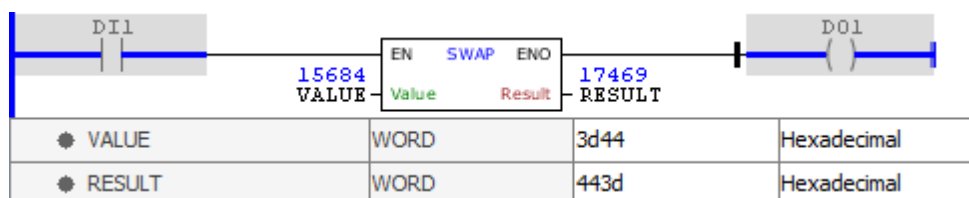
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

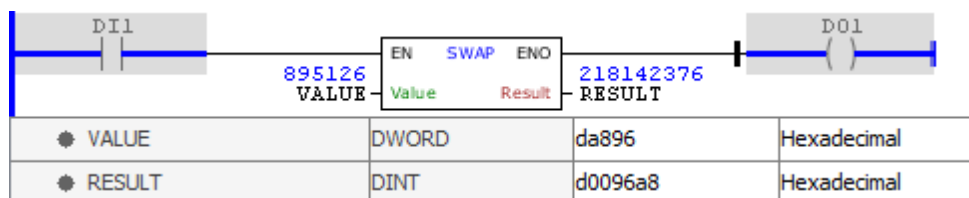
Fluxograma do Bloco



Exemplo



O exemplo troca a posição do valor do byte 1 de VALUE (0x44) com o byte 2 de VALUE (0x3D), armazenando o resultado final (0x44_3D) em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.



O exemplo troca a posição do valor do byte 1 de VALUE (0x96) com o byte 2 de VALUE (0xA8) e do byte 3 de VALUE (0x0D) com o byte 4 de VALUE (0x00), armazenando o resultado final (0x0D_00_96_A8) em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.2.7.1.14.10 USERERR

Bloco que gera um alarme ou falha com o número programado pelo usuário.

Representação Ladder



Estrutura do Bloco

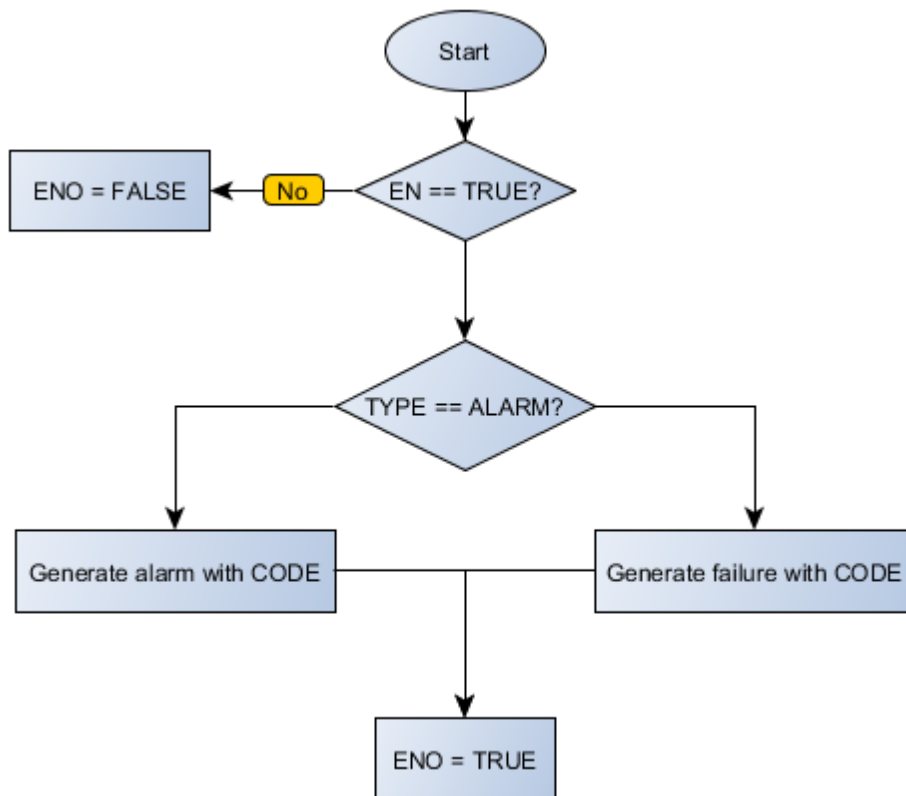
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	CODE	WORD UINT	Código do erro gerado (950 - 999)
	TYPE	BYTE	Tipo de erro gerado (0 - Alarme) (1 - Falha)
VAR_OUTPUT	ENO	BOOL	Sucesso na geração do erro
VAR	USERERR_INST_0	USERERR	Instância de acesso à estrutura do bloco

Funcionamento

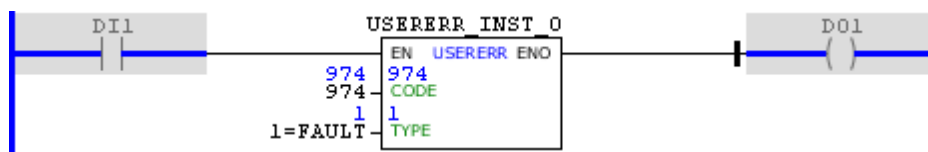
Este bloco, quando tem um valor TRUE em EN, gera um alarme ou falha no equipamento, dependendo do tipo definido em TYPE com o código CODE.

O valor de ENO informa se a geração do alarme ou falha foi executada com sucesso.

Fluxograma do Bloco



Exemplo



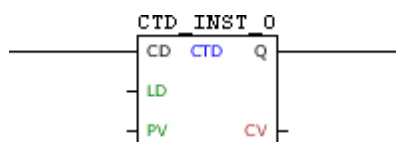
O exemplo acima, ao identificar nível TRUE em DI1, gera uma falha com o código 974 e seta a saída DO1.

10.2.7.1.15 Counter

10.2.7.1.15.1 CTD

Bloco de contagem regressiva de pulsos de entrada.

Representação Ladder



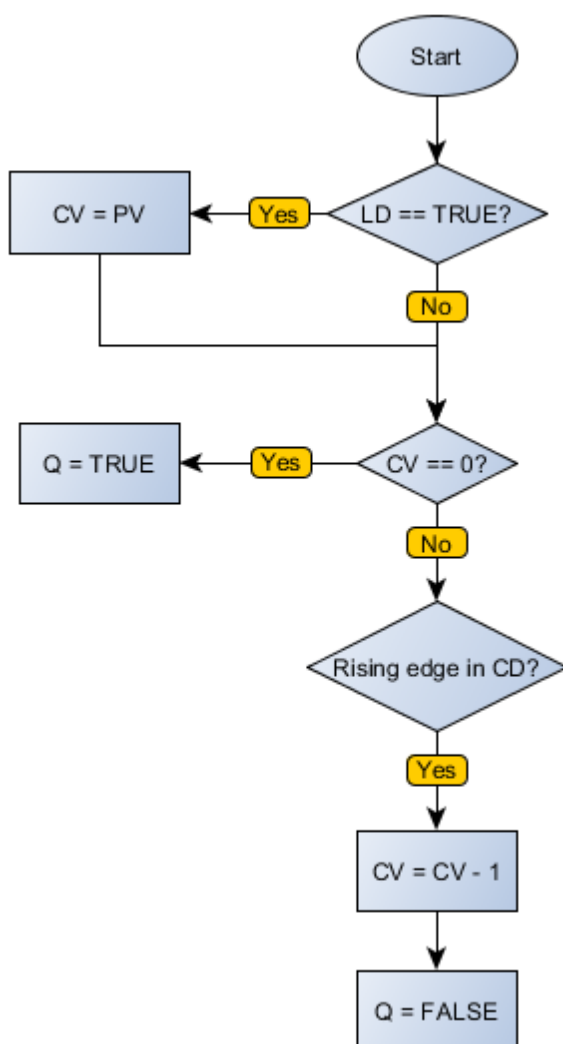
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	CD	BOOL	Identificador de pulsos
	LD	BOOL	Carrega o valor de PV em CV
	PV	WORD UINT	Valor de configuração inicial
VAR_OUTPUT	Q	BOOL	Sinalizador de contador zerado
	CV	WORD UINT	Valor de contagem atual
VAR	CTD_INST_0	CTD	Instância de acesso à estrutura do bloco

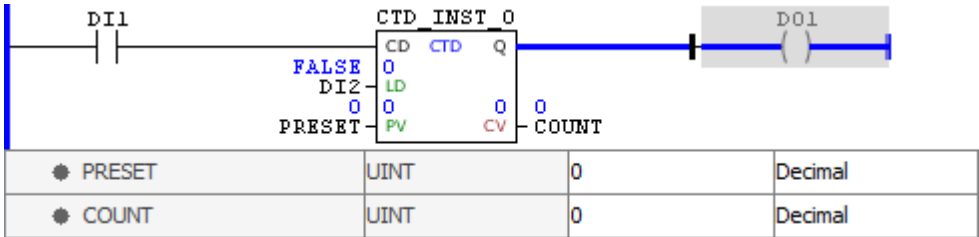
Funcionamento

Este bloco, quando identifica uma borda de subida em CD, decrementa a variável CV até que a mesma seja zero. Enquanto CV for igual a zero, a saída Q permanece em nível TRUE. Ao detectar nível alto em LD, o bloco carrega o valor de PV em CV.

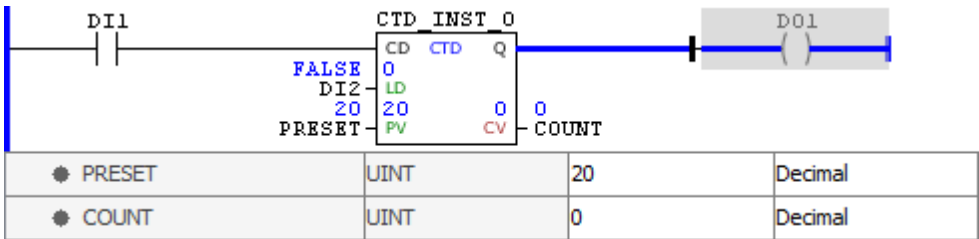
Fluxograma do Bloco



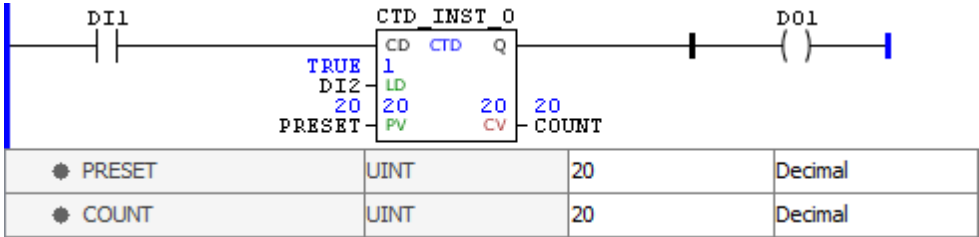
Exemplo



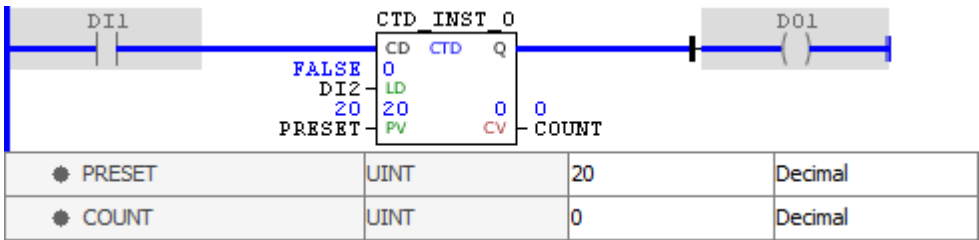
O exemplo acima exibe as condições iniciais da rotina. Como CV possui o valor zero, a saída Q é habilitada.



O valor da variável PV foi alterado para 20, porém ainda não foi carregado.



Ao identificar o nível TRUE em LD, o bloco carrega o valor de PV para CV. Como este valor é maior que zero, a saída Q é desabilitada.

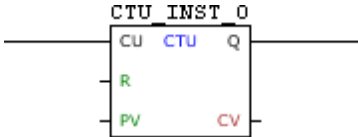


A cada borda de subida identificada em CD, o valor de COUNT é decrementado até chegar a zero, quando a saída Q é habilitada.

10.2.7.1.15.2 CTU

Bloco de contagem progressiva de pulsos de entrada.

Representação Ladder



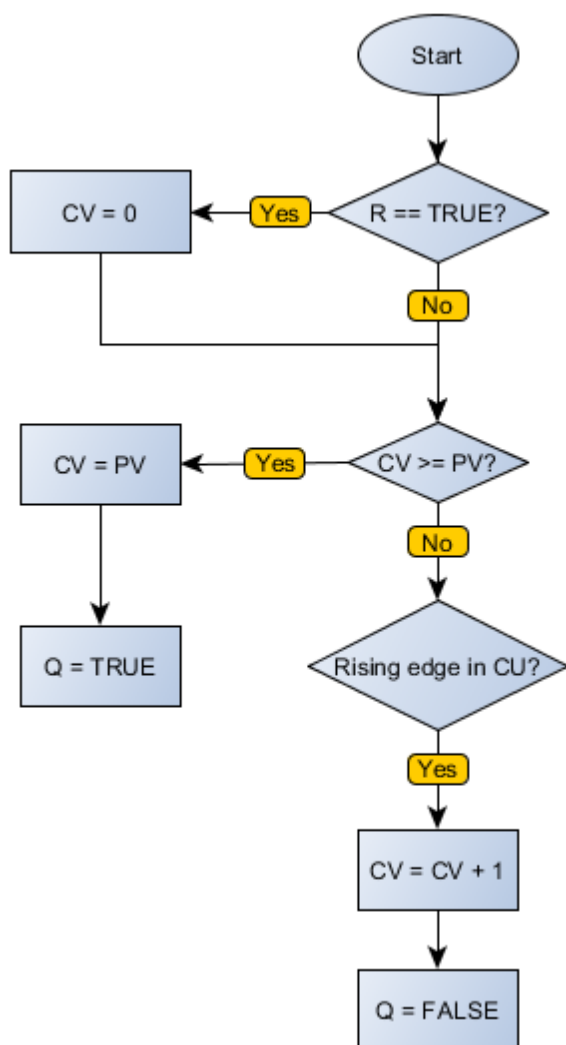
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	CU	BOOL	Identificador de pulsos
	R	BOOL	Carrega o valor zero em CV
	PV	WORD UINT	Valor máximo de contagem
VAR_OUTPUT	Q	BOOL	Sinalizador de contador estourado
	CV	WORD UINT	Valor de contagem atual
VAR	CTU_INST_0	CTU	Instância de acesso à estrutura do bloco

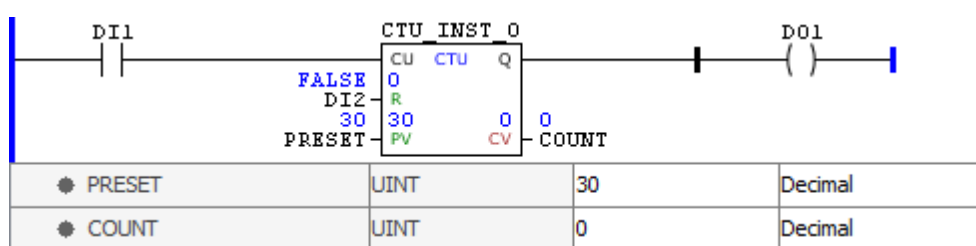
Funcionamento

Este bloco, quando identifica uma borda de subida em CD, incrementa a variável CV até que a mesma seja igual a PV. Enquanto CV for igual a PV, a saída Q permanece em nível TRUE. Ao detectar nível alto em R, o bloco carrega o valor zero em CV.

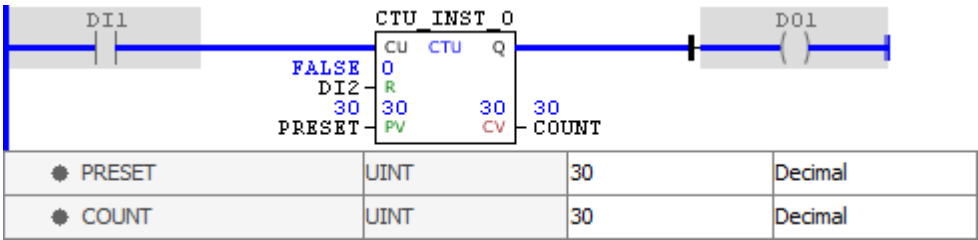
Fluxograma do Bloco



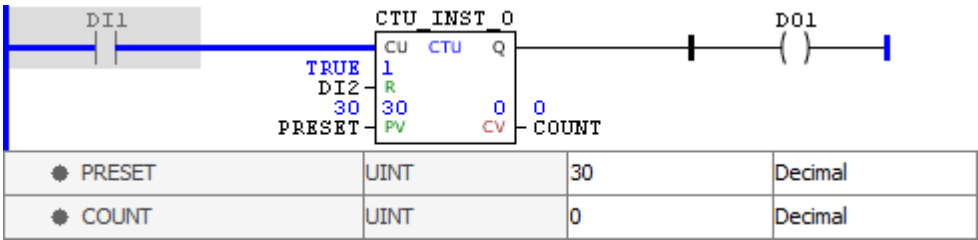
Exemplo



O exemplo acima exibe as condições iniciais da rotina. Como CV possui um valor inferior ao de PV, a saída Q é desabilitada.



A cada borda de subida identificada em CU, o valor de CV é incrementado até chegar ao valor de PV, quando a saída Q é habilitada.

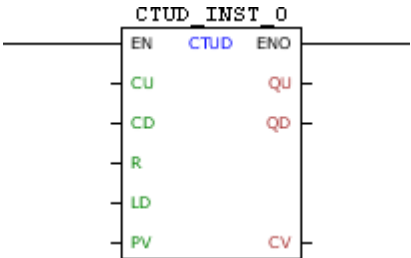


Ao identificar o nível TRUE em R, o bloco carrega o valor zero para CV. Como este valor é inferior ao de PV, a saída Q é desabilitada.

10.2.7.1.15.3 CTUD

Bloco de contagem progressiva e regressiva de pulsos de entrada.

Representação Ladder



Estrutura do Bloco

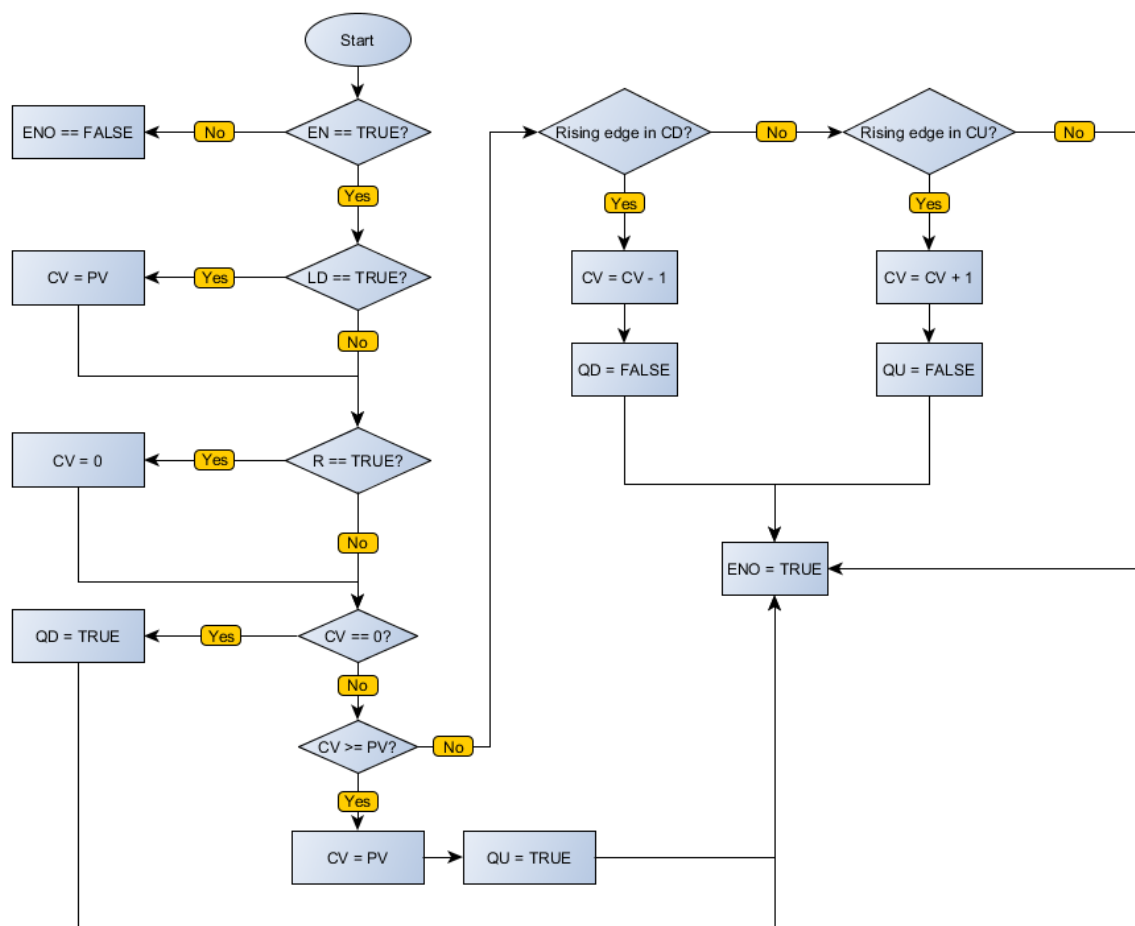
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	CU	BOOL	Identificador de pulsos para incrementador
	CD	BOOL	Identificador de pulsos para decrementador
	R	BOOL	Carrega o valor zero em CV
	LD	BOOL	Carrega o valor de PV em CV
	PV	WORD UINT	Valor de referência
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	QU	BOOL	Sinalizador de contador estourado
	QD	BOOL	Sinalizador de contador zerado
	CV	WORD UINT	Valor de contagem atual
VAR	CTUD_INST_0	CTUD	Instância de acesso à estrutura do bloco

Funcionamento

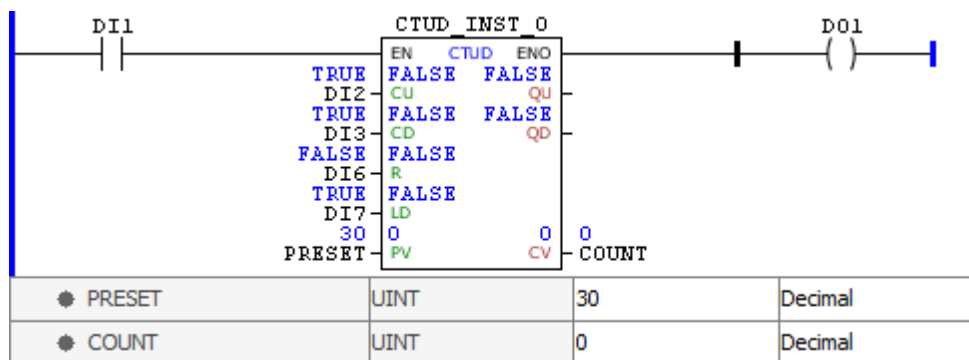
Este bloco, quando tem um valor TRUE em EN, atua como um bloco CTD e um bloco CTU ao mesmo tempo, atuando sobre o mesmo contador CV. Ou seja: incrementa CV até chegar a PV para bordas de subida em CU e decrementa CV até chegar a zero para bordas de subida em CD. Uma transição positiva em R carrega zero em CV, enquanto uma borda de subida em LD carrega o valor de PV em CV. Caso CV tenha valor zero, QD recebe TRUE, e caso CV tenha valor igual a PV, QU recebe TRUE.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN.

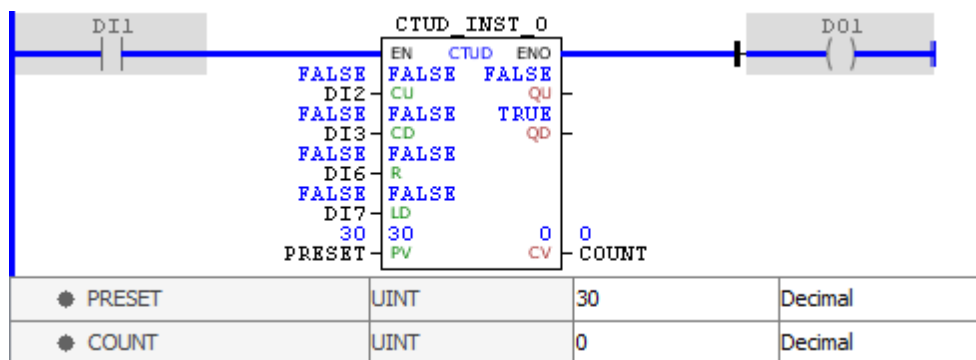
Fluxograma do Bloco



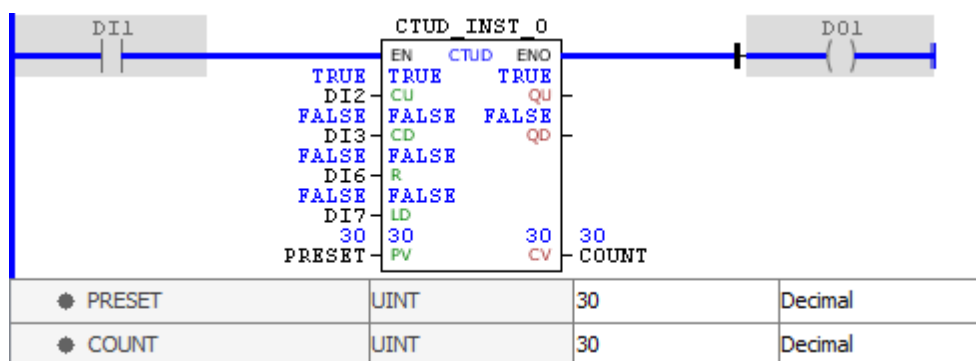
Exemplo



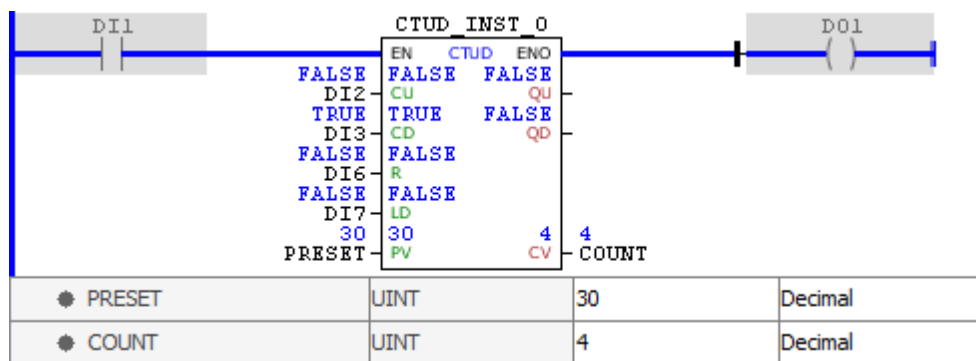
O exemplo acima exibe o bloco desabilitado, com suas variáveis internas todas zeradas. Ainda que os controles externos estejam ativados, esses valores não são repassados para a instância do bloco.



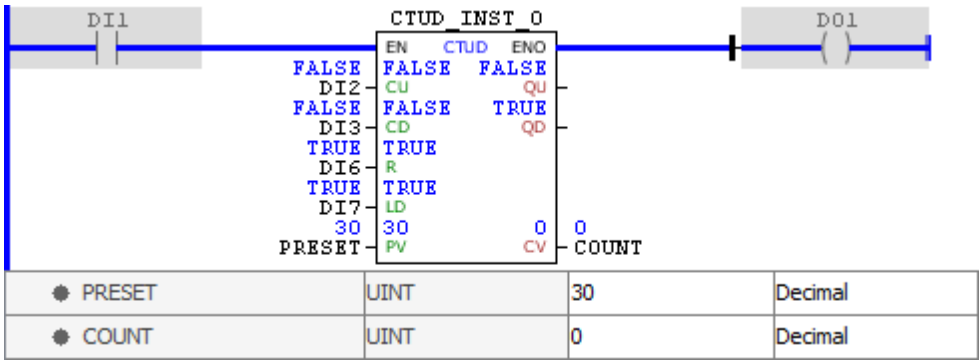
Ao ser ativado, o bloco identifica o valor de PRESET, carregando-o em PV, e identifica que a saída está com o valor zero, habilitando a saída QD. Finalizada a execução, a saída ENO é ativada.



A cada borda de subida identificada em CU, o valor de CV é incrementado até chegar ao valor de PV, quando a saída QU é habilitada. Finalizada a execução, a saída ENO é ativada.



A cada borda de subida identificada em CD, o valor de CV é decrementado. Sendo CV de um valor intermediário a zero e PV, ambas as saídas QD e QU são desativadas. Finalizada a execução, a saída ENO é ativada.



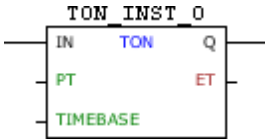
Um valor TRUE em R reseta CV, enquanto um valor TRUE em LD carrega o valor de PV para CV. Como podemos notar, R prevalece sobre LD, restando CV e habilitando a saída QD. Finalizada a execução, a saída ENO é ativada.

10.2.7.1.16 Timer

10.2.7.1.16.1 TON

Bloco de temporizador que, ao ser energizado, habilita a saída após um atraso definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Atraso de acionamento da saída
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TON_INST_0	TON	Instância de acesso à estrutura do bloco

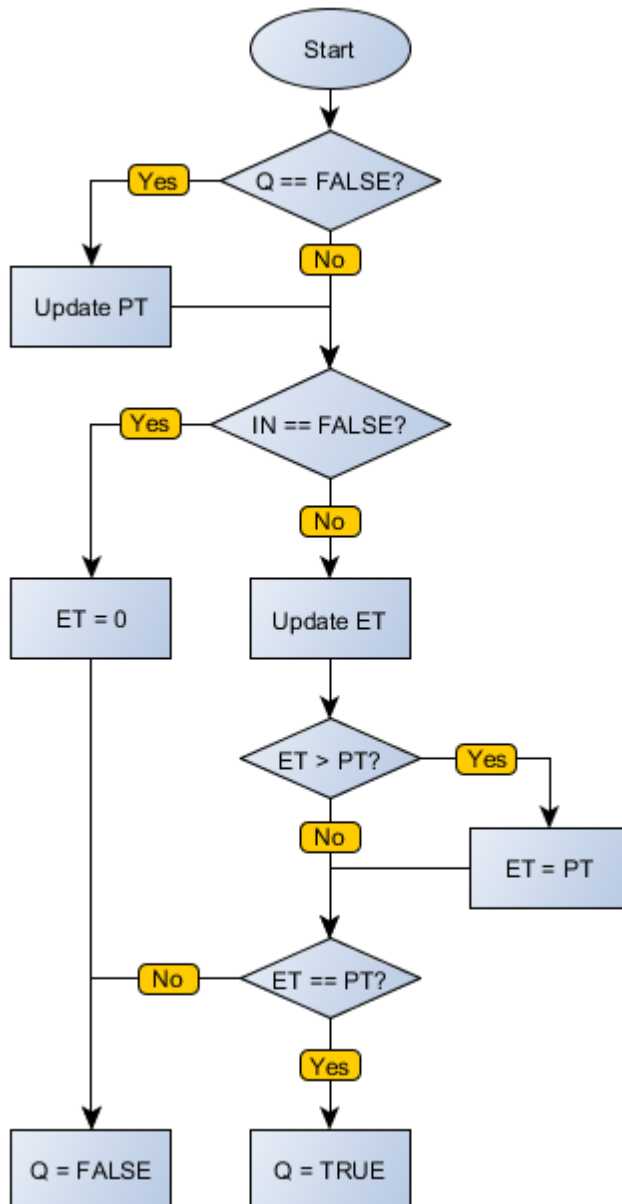
Funcionamento

Enquanto a entrada IN for FALSE, a saída Q também é FALSE e ET recebe o valor zero. Na transição positiva de borda em IN, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado TRUE, até que IN volte a ser FALSE.

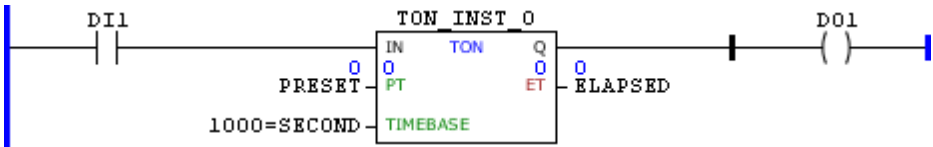
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

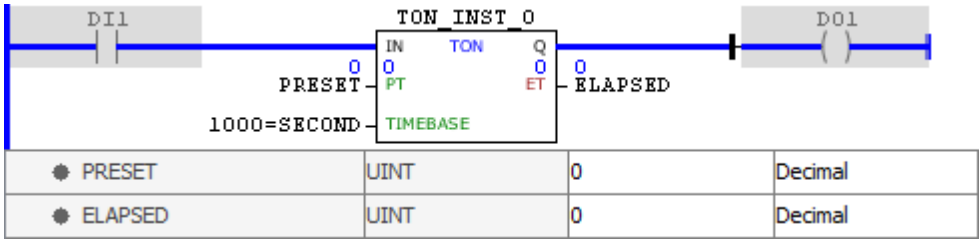
Fluxograma do Bloco



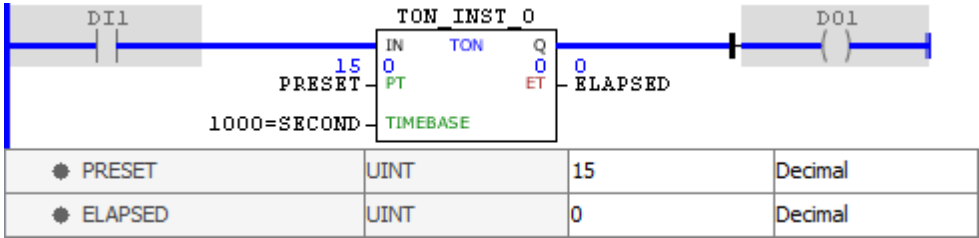
Exemplo



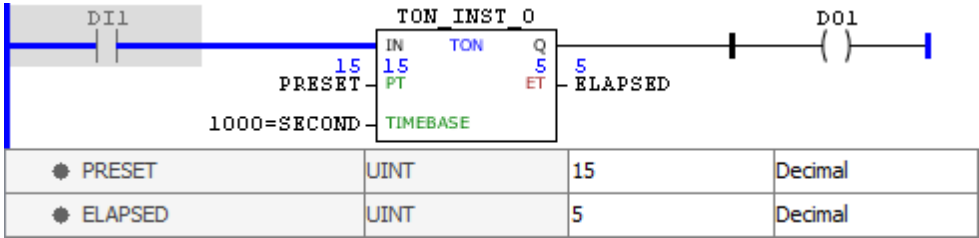
O exemplo acima exibe as condições iniciais do bloco e das variáveis da rotina.



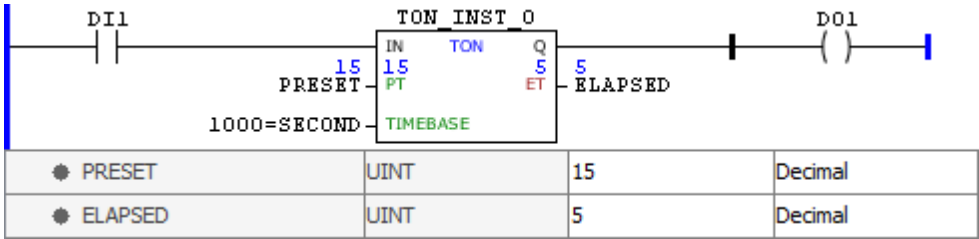
Ao ser ativada a entrada IN, a contagem dispara. Como ET é igual a PT, a saída Q é habilitada.



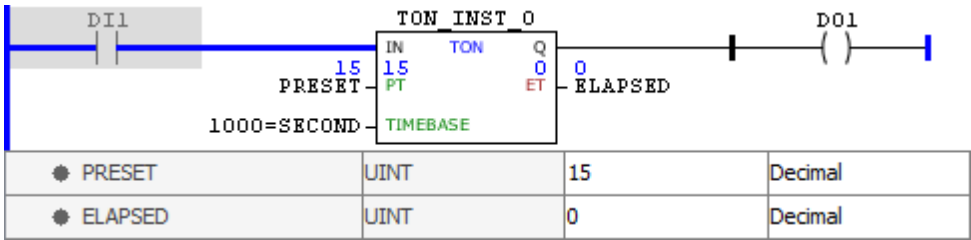
Observe que uma alteração na variável PRESET não é repassada ao campo PT enquanto a entrada IN permanece habilitada.



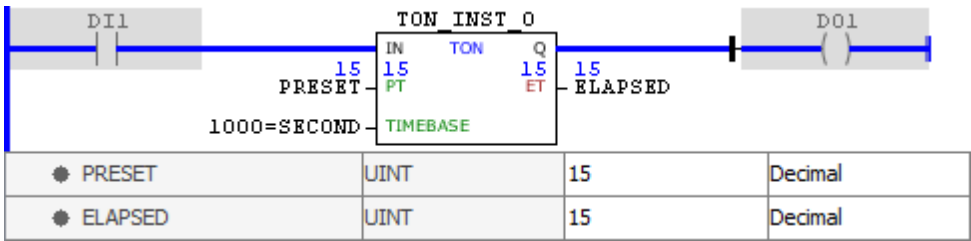
Desabilitando a entrada IN, o valor de PT é atualizado e a saída Q é desabilitada. Ao ativá-la novamente, a contagem dispara.



Desabilitando a entrada IN, o valor de ET permanece salvo.



Habilitando a entrada IN, o valor de ET é resetado e a contagem dispara.



Quando ET atinge o valor de PT, a saída Q é habilitada e permanece assim enquanto IN estiver em nível TRUE.

10.2.7.1.16.2 TOF

Bloco de temporizador que, ao ser desenergizado, desabilita a saída após um atraso definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Atraso de desacionamento da saída
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TOF_INST_0	TOF	Instância de acesso à estrutura do bloco

Funcionamento

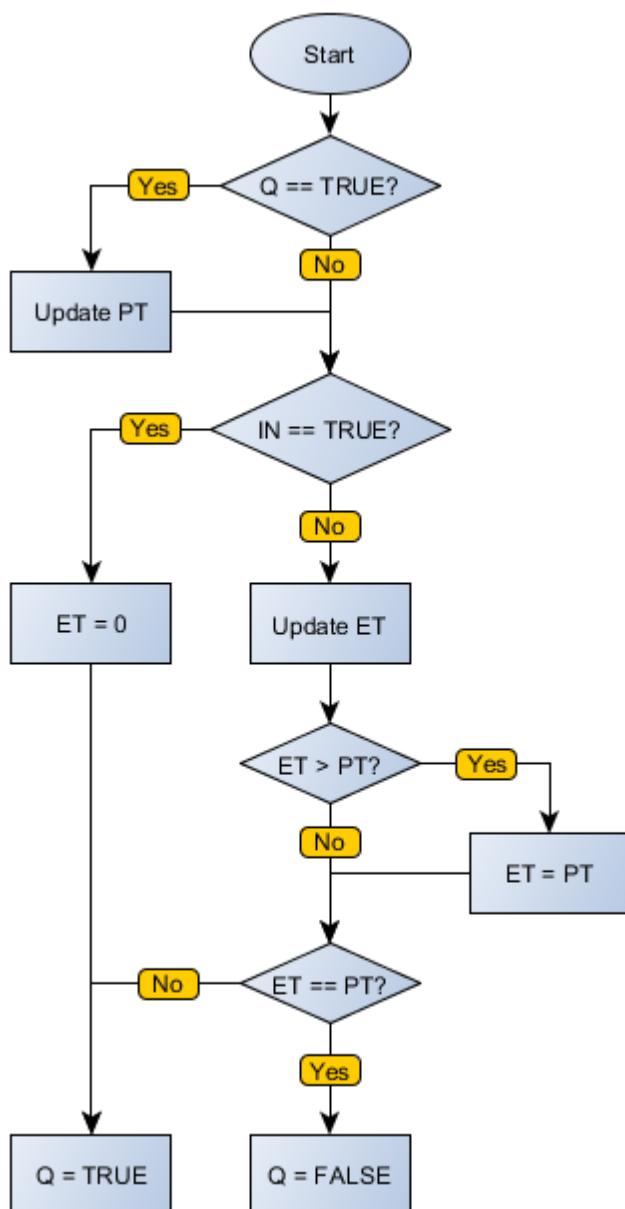
Enquanto a entrada IN for TRUE, a saída Q também é TRUE e ET recebe o valor zero. Na transição negativa de borda em IN, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado FALSE, até que IN volte a ser

TRUE.

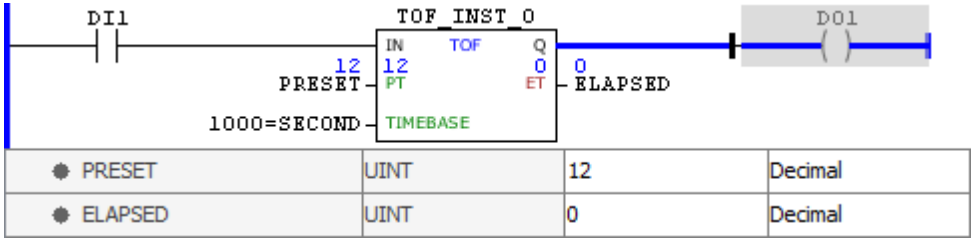
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo

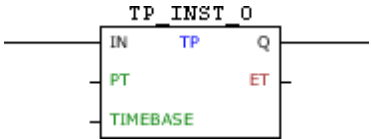


O exemplo acima desabilita a saída DO1 ao identificar um nível baixo em DI1 por 12 segundos, permanecendo desabilitada até que DI1 volte a ser TRUE.

10.2.7.1.16.3 TP

Bloco de temporizador que, ao identificar ser energizado, habilita a saída por um tempo definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Tempo em que a saída fica habilitada
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TP_INST_0	TP	Instância de acesso à estrutura do bloco

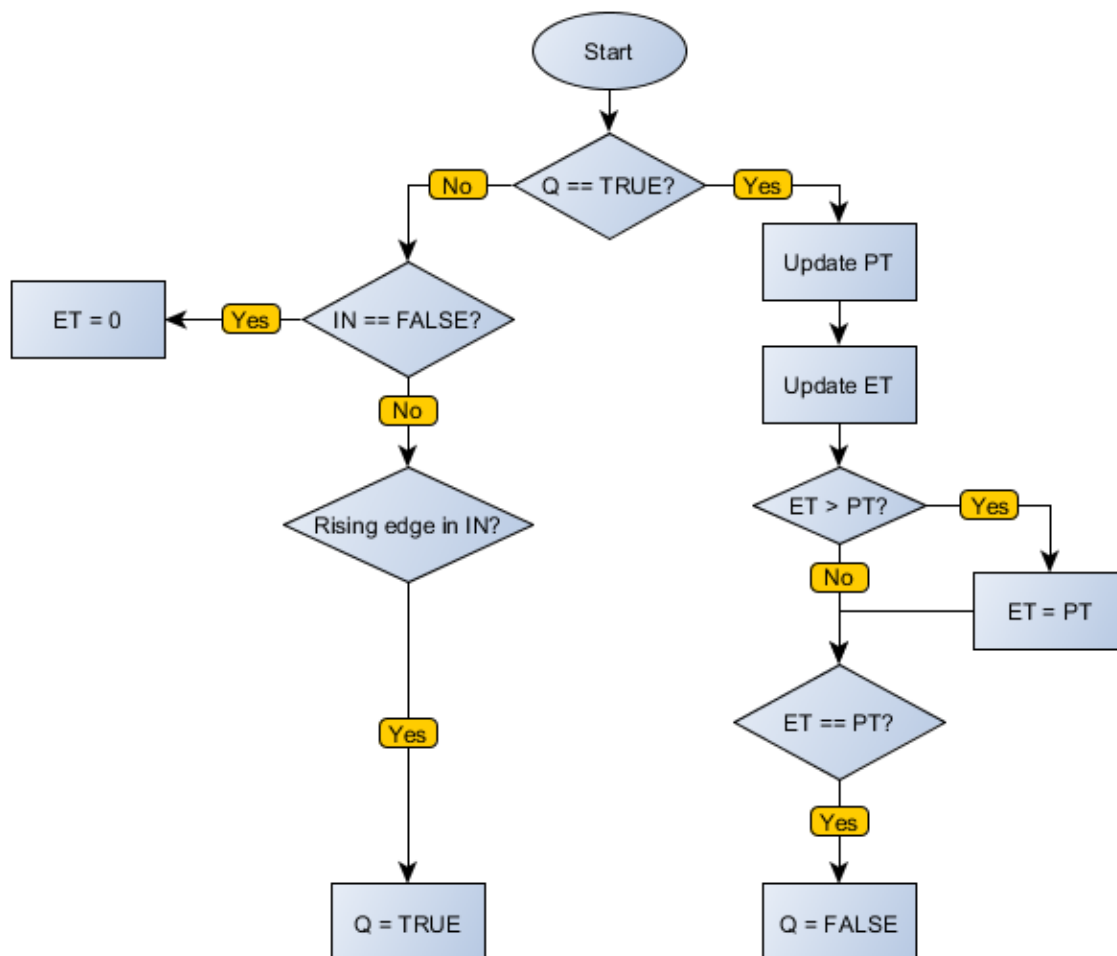
Funcionamento

Na transição positiva de borda em IN, Q recebe valor TRUE, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado FALSE, até que IN volte a ser TRUE. Nesse momento, se IN estiver em nível TRUE, nada acontece. Se IN estiver em nível FALSE, ET é resetado automaticamente.

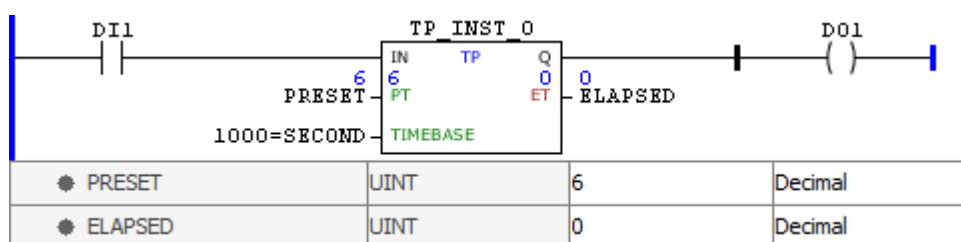
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



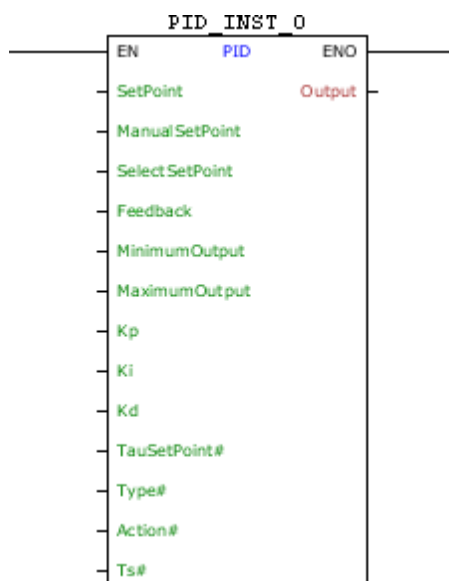
O exemplo acima habilita a saída DO1 por seis segundos a cada transição positiva de DI1.

10.2.7.1.17 Control

10.2.7.1.17.1 PID

Bloco que executa a função de um controlador PID discreto. A partir das variáveis de entrada, ele calcula a saída correspondente do controlador.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SetPoint	REAL	Referência automática (pré-controle)
	ManualSetPoint	REAL	Referência forçada (pós-controle)
	SelectSetPoint	BOOL	Seleciona qual referência utilizar
	Feedback	REAL	Variável de realimentação da malha
	MinimumOutput	REAL	Valor mínimo da saída do controlador
	MaximumOutput	REAL	Valor máximo da saída do controlador
	Kp	REAL	Ganho proporcional
	Ki	REAL	Ganho integral
	Kd	REAL	Ganho derivativo
	TauSetPoint#	REAL	Constante de tempo do filtro de entrada de referência automática
	Type#	BYTE	Tipo de controlador
	Action#	BYTE	Ação de controle
	Ts#	UINT	Período de amostragem [ms]
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Output	REAL	Saída do controlador
VAR	PID_INST_0	PID	Instância de acesso à estrutura do bloco

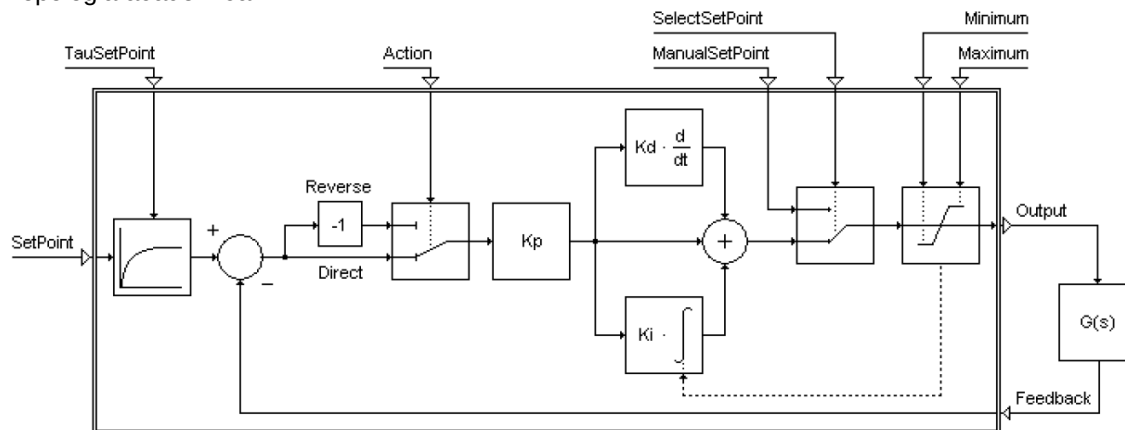
Funcionamento

Na transição positiva de borda em EN, Output recebe valor zero, e o bloco executa sua

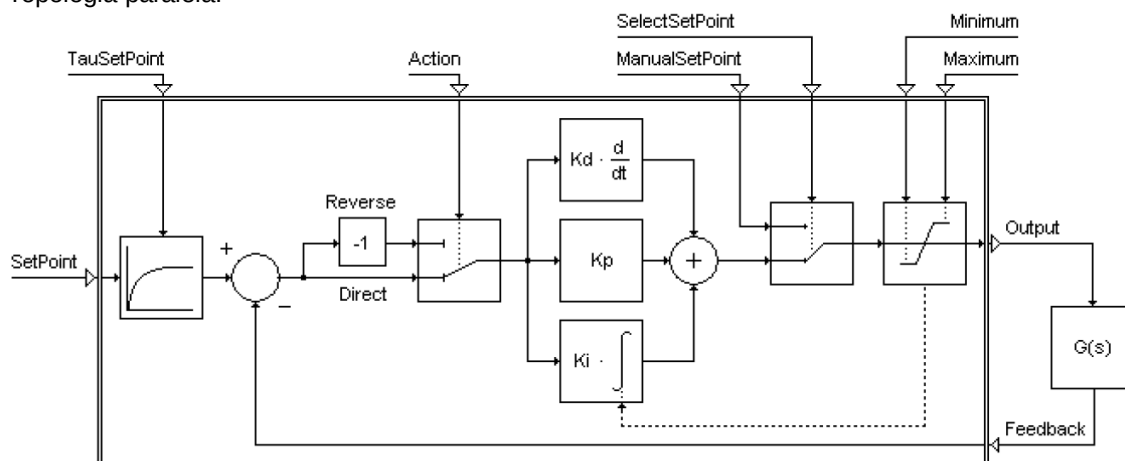
funcionalidade enquanto EN estiver em nível TRUE.

Quando habilitado, este bloco executa uma rotina de controle PID com os parâmetros K_p , K_i e K_d escolhidos. A topologia PID utilizada poderá ser a Acadêmica ou a Paralela, dependendo do que for escolhido em Type#.

Topologia acadêmica:



Topologia paralela:



Os níveis do sinal da saída do controlador são saturados pelo valor de MinimumOutput e MaximumOutput. A entrada SelectSetPoint em nível FALSE faz com que a referência adotada seja SetPoint, permitindo que o controlador exerça domínio sobre o processo. Quando SelectSetPoint vai para nível TRUE, o controlador não exerce mais domínio, sendo que ManualSetPoint passa a ser considerado o sinal de saída do controlador.

Action# definirá a operação de realimentação. Se Action# for DIRECT, a operação será SetPoint – Feedback. Se Action# for REVERSE, a operação será Feedback – SetPoint.

Feedback recebe a variável de processo considerada como saída da planta. Ts# recebe o período de amostragem para o controlador e TauSetPoint# recebe a constante de tempo para o filtro de entrada da referência automática.

Quando EN possui valor FALSE, Output permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

**NOTA!**

Efeito da alteração dos ganhos sobre o processo:

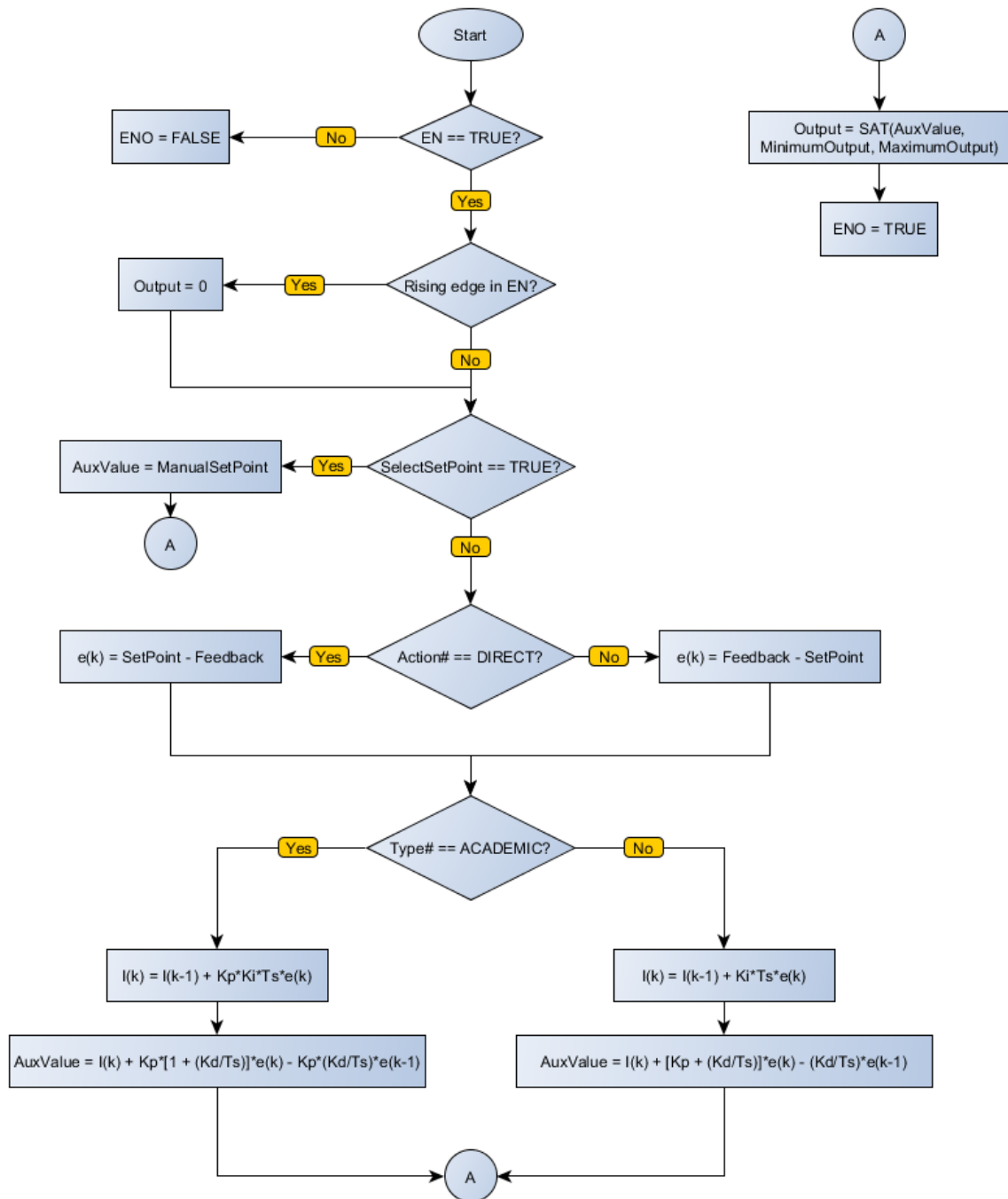
- Se diminuir K_p , o processo torna-se mais lento; geralmente mais estável ou menos oscilante; tem menos sobre-sinal (overshoot).
- Se aumentar K_p , o processo responde mais rápido; pode ficar instável ou mais oscilante; tem mais sobre-sinal (overshoot).
- Se diminuir K_i , o processo torna-se mais lento, demorando para atingir o "SetPoint"; fica mais estável ou menos oscilante; tem menos sobre-sinal (overshoot).
- Se aumentar K_i , o processo torna-se mais rápido, atingindo rapidamente o "SetPoint"; fica mais instável ou mais oscilante; tem mais sobre-sinal (overshoot).
- Se diminuir K_d , o processo torna-se mais lento; tem menos sobre-sinal (overshoot).
- Se aumentar K_d , tem mais sobre-sinal (overshoot).

**NOTA!**

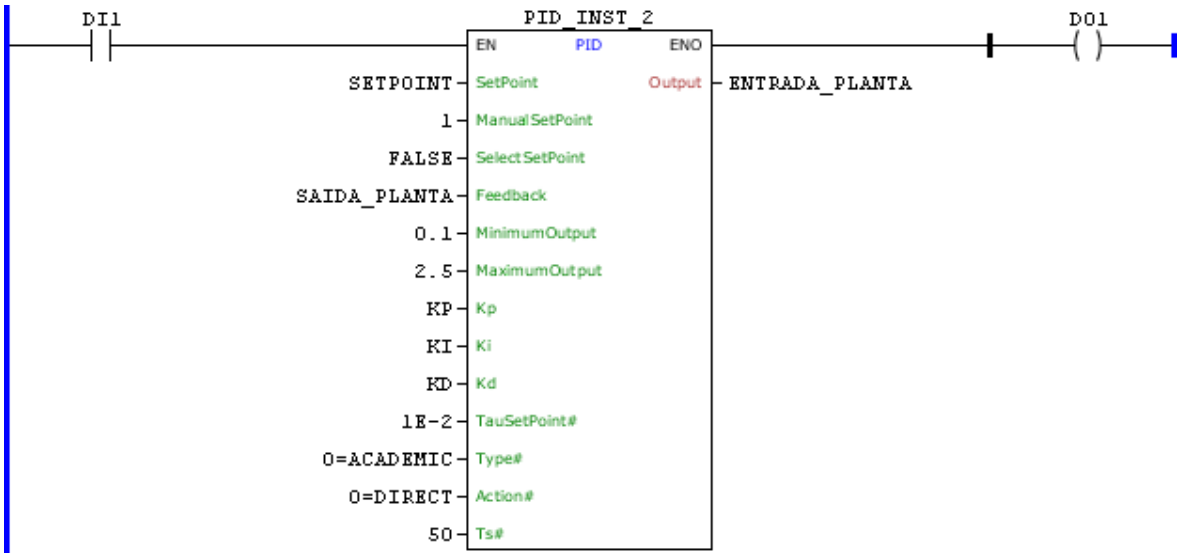
Como melhorar o desempenho do processo pelo ajuste dos ganhos (válido para o PID Acadêmico)

- Se o desempenho do processo está quase bom, mas o sobre-sinal (overshoot) está um pouco alto, tente: (1) diminuir o K_p em 20%, (2) diminuir K_i em 20% e/ou (3) diminuir K_d em 50%.
- Se o desempenho do processo está quase bom, mas não tem o sobre-sinal (overshoot) e demora para atingir o "SetPoint", tente: (1) aumentar K_p em 20%, (2) aumentar K_i em 20%, (3) aumentar K_d em 50%.
- Se o desempenho do processo está bom, mas a saída do processo está variando demais, tente: (1) aumentar K_d em 50%, (2) diminuir K_p em 20%.
- Se o desempenho do processo está ruim, ou seja, após a partida, o transitório dura vários períodos de oscilação, que reduz muito lentamente ou não reduz, tente: (1) diminuir K_p em 50%.
- Se o desempenho do processo está ruim, ou seja, após a partida avança lentamente em direção ao "SetPoint", sem sobre-sinal (overshoot), mas ainda está muito longe e a saída do processo é menor que o valor nominal, tente: (1) aumentar K_p em 50%, (2) aumentar K_i em 50%, (3) aumentar K_d em 70%.

Fluxograma do Bloco



Exemplo



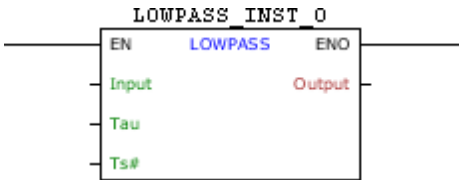
O exemplo acima cria uma malha de um PID acadêmico digital, com tempo de amostragem de 50 ms, utilizando as constantes KP, KI e KD para o controle. É utilizada a referência automática SETPOINT, filtrada por um filtro de primeira ordem com constante de tempo 0.01. O sinal de erro é calculado como a diferença entre a referência filtrada e a variável SAIDA_PLANTA. A saída do controlador é saturada entre os valores 0.1 e 2.5 e enviada para a variável ENTRADA_PLANTA.

10.2.7.1.18 Filter

10.2.7.1.18.1 LOWPASS

Bloco que filtra a entrada utilizando um filtro passa baixas de primeira ordem e insere o resultado na saída.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Input	REAL	Sinal de entrada
	Tau	REAL	Constante de tempo do filtro
	Ts#	UINT	Período de amostragem [ms]
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Output	REAL	Saída do filtro
VAR	LOWPASS_INST_0	LOWPASS	Instância de acesso à estrutura do bloco

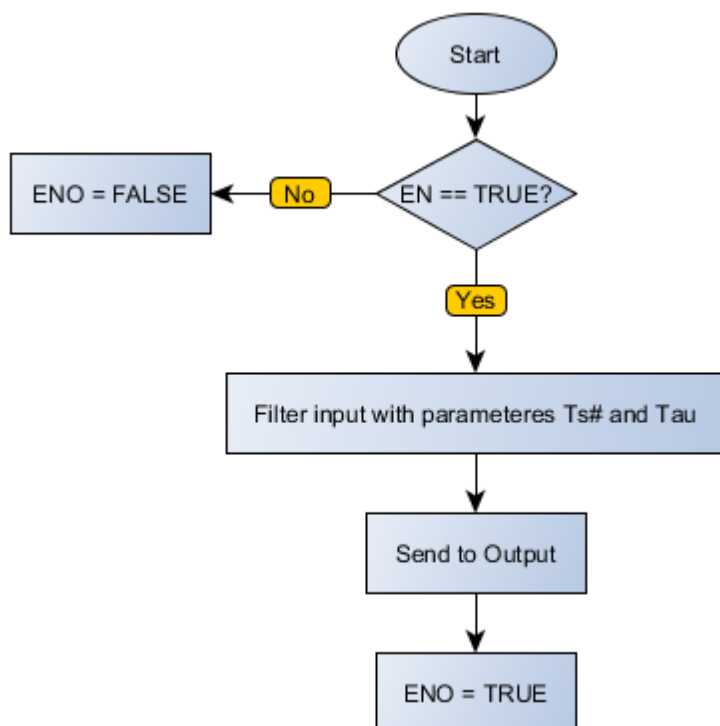
Funcionamento

Este bloco, quando tem um valor TRUE em EN, filtra o valor de entrada Input utilizando um filtro passa baixas de primeira ordem descrito por Tau e Ts#, inserindo o resultado em Output. Na borda de subida de EN, Output recebe zero.

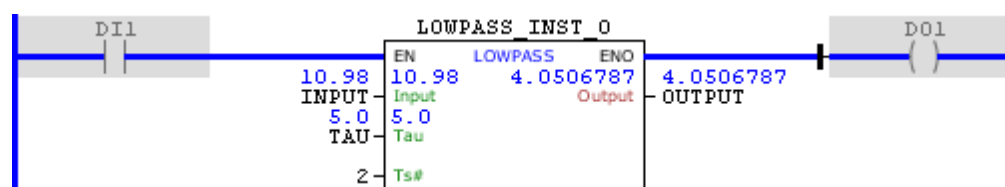
Quando EN possui valor FALSE, Output permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

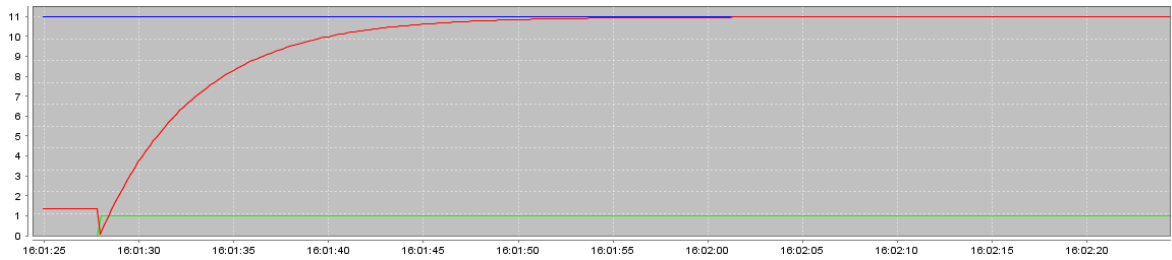
Fluxograma do Bloco



Exemplo



O exemplo acima faz com que a saída OUTPUT, ao identificar uma borda de subida em EN, exiba um comportamento de primeira ordem, com constante de tempo igual a TAU e tempo de amostragem de 2 ms, visando a atingir a referência setada em INPUT. A cada cálculo finalizado com sucesso, a saída ENO é ativada.

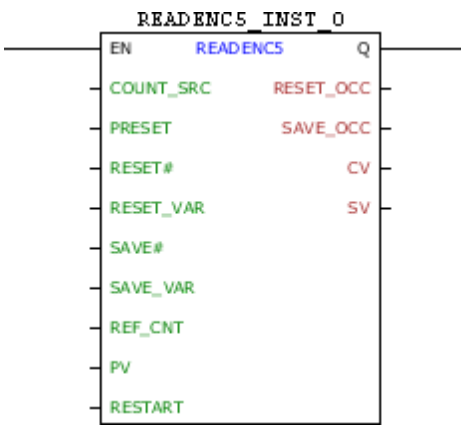


10.2.7.1.19 Hardware

10.2.7.1.19.1 READENC5

Bloco que executa contagem dos pulsos de encoder.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	COUNT_SRC	BYTE	Determina qual o encoder será utilizado na contagem dos pulsos
	PRESET	BOOL	Atribui o valor de PV a CV
	RESET#	BYTE	Escolhe o controle do reset de CV
	RESET_VAR	BOOL	Caso a escolha de RESET# seja por variável, realiza o reset de CV quando em valor TRUE
	SAVE#	BYTE	Escolhe o controle do salvamento do contador
	SAVE_VAR	BOOL	Caso a escolha de SAVE# seja por variável, realiza o salvamento de CV em SV quando em valor TRUE
	REF_CNT	LREAL	Valor de pulso de referência para habilitação de saída
	PV	LREAL	Valor de configuração inicial
	RESTART	LREAL	Valor de referência para reset automático de CV
VAR_OUTPUT	Q	BOOL	Habilitação da saída
	RESET_OCC	BOOL	Sinalizador de reset
	SAVE_OCC	BOOL	Sinalizador de salvamento
	CV	LREAL	Valor do contador de pulsos
	SV	LREAL	Valor do contador salvo pela última vez
VAR	READENC5_INST_0	READENC5	Instância de acesso à estrutura do bloco

Funcionamento

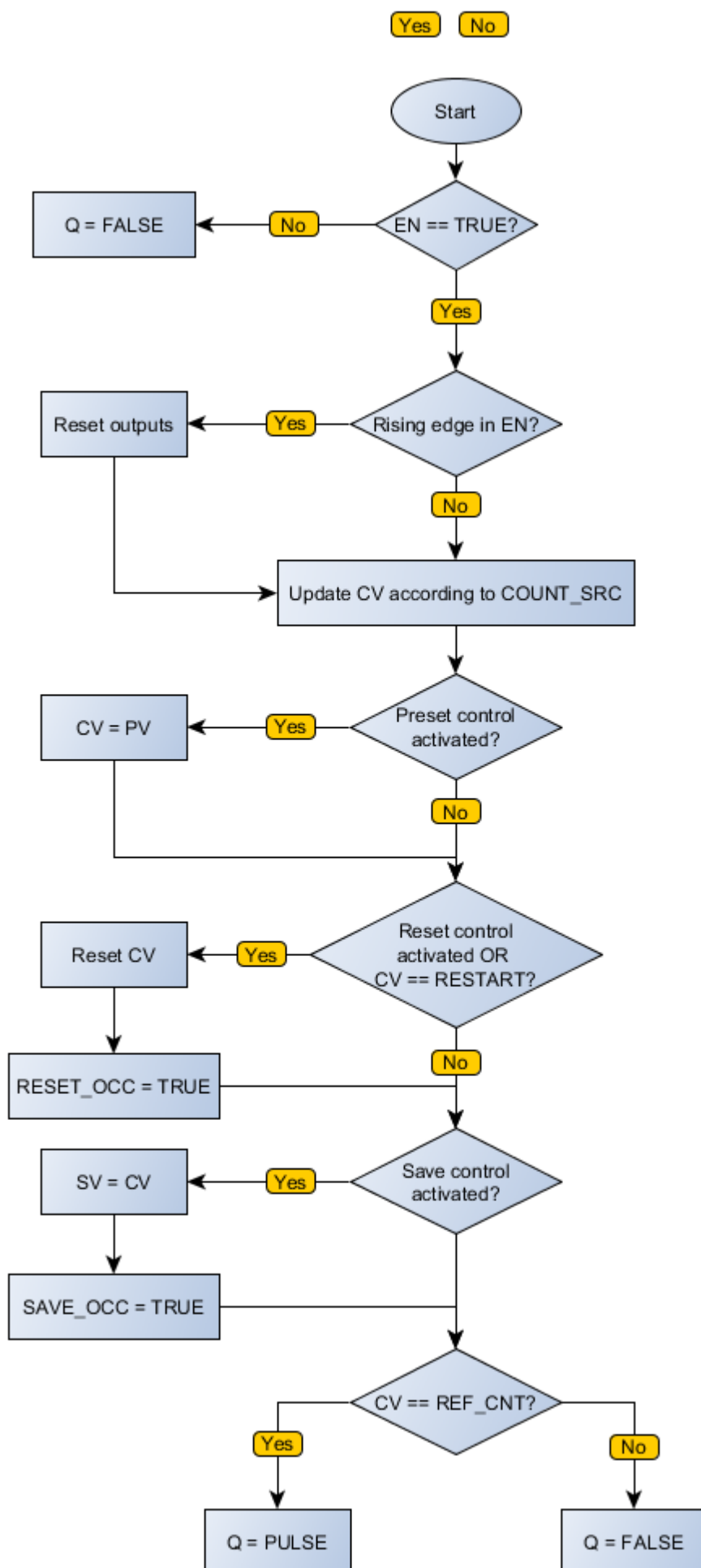
Este bloco, quando identifica uma borda de subida em EN, zera suas saídas e realiza a contagem dos pulsos do encoder especificado em COUNT_SRC enquanto habilitado. Esse valor de contagem é armazenado em CV.

O controle especificado em RESET# zera o contador quando for ativado, enquanto o controle especificado em SAVE#, quando ativado, armazena o valor de CV em SV. Cada um desses controles seta seu respectivo sinalizador (RESET_OCC ou SAVE_OCC) por um ciclo de scan.

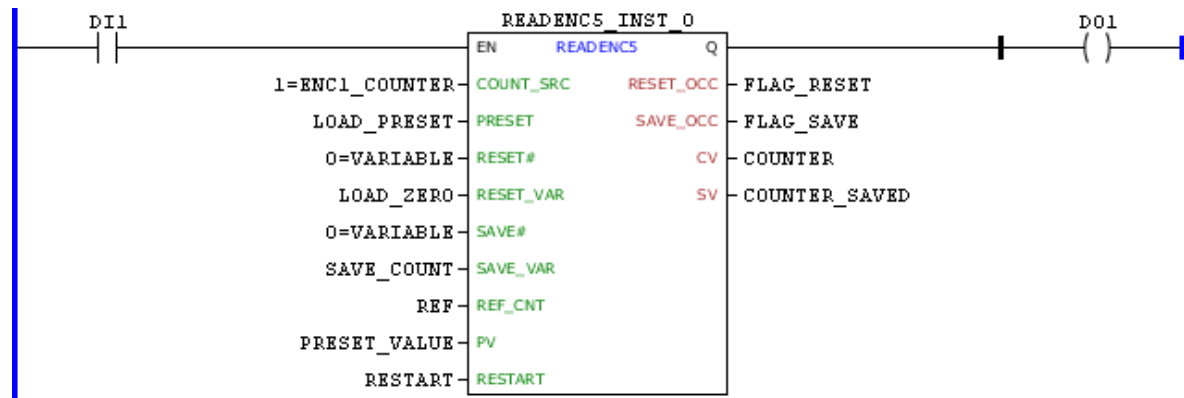
O bloco também permite configurar um reset automático, quando CV atinge o valor de RESTART.

O valor de Q é ativado por um ciclo de scan quando CV atinge o valor de REF_CNT.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a leitura do encoder 1, armazenando o valor em COUNTER, com referência de contagem REF e controles de RESET e SAVE realizados pelas variáveis LOAD_ZERO e SAVE_COUNT, respectivamente. Quando COUNTER for igual a REF, a saída Q é ativada por um ciclo de scan.

10.2.7.1.19.2 IMMEDIATE_INPUT

Bloco que realiza uma leitura instantânea do valor da entrada selecionada, sem alterar o valor das imagens (variáveis GLOBAL_IO).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source#	BYTE	Entradas a serem lidas (digitais ou analógicas)
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Destination	WORD INT UINT	Variável mapeada com os valores das entradas selecionadas

Funcionamento

Este bloco, quando tem um valor TRUE em EN, obtém o valor imediato da entrada selecionada em Source#. Caso seja selecionada a entrada analógica AI1, seu valor é repassado a Destination. Caso sejam selecionadas as entradas digitais, seus bits são concatenados de forma a DI1 ser o bit menos significativo e DI10 ser o bit mais significativo, sendo o resultado enviado a Destination.

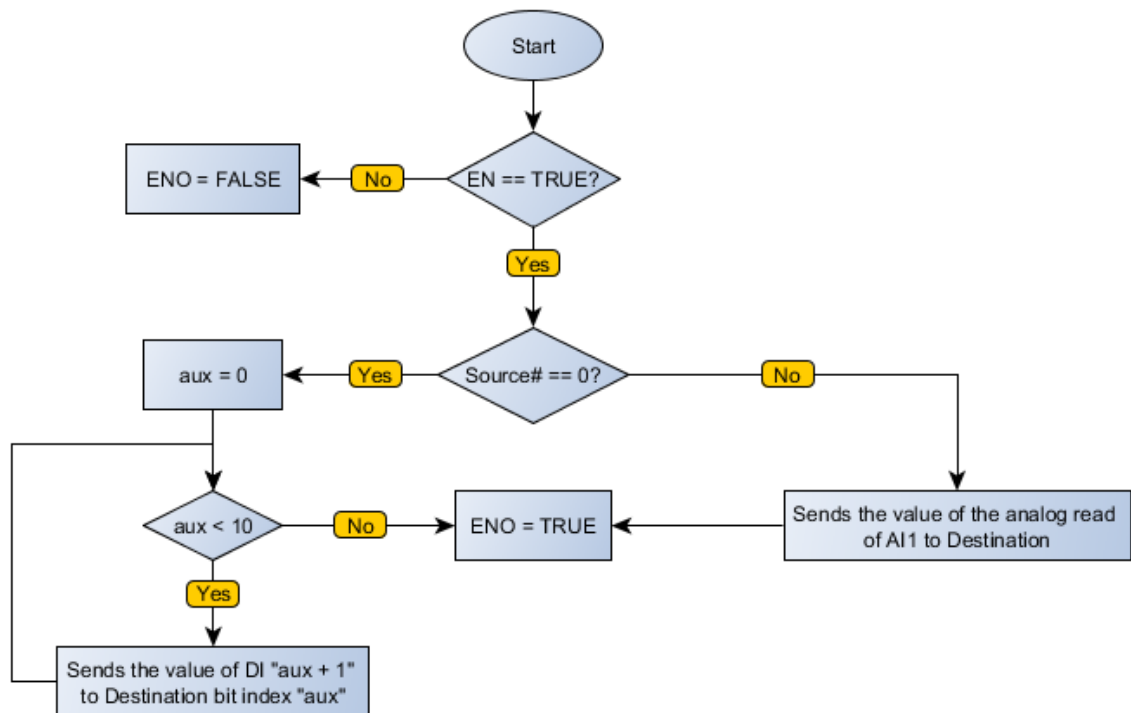
Quando EN possui valor FALSE, Destination permanece inalterada.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

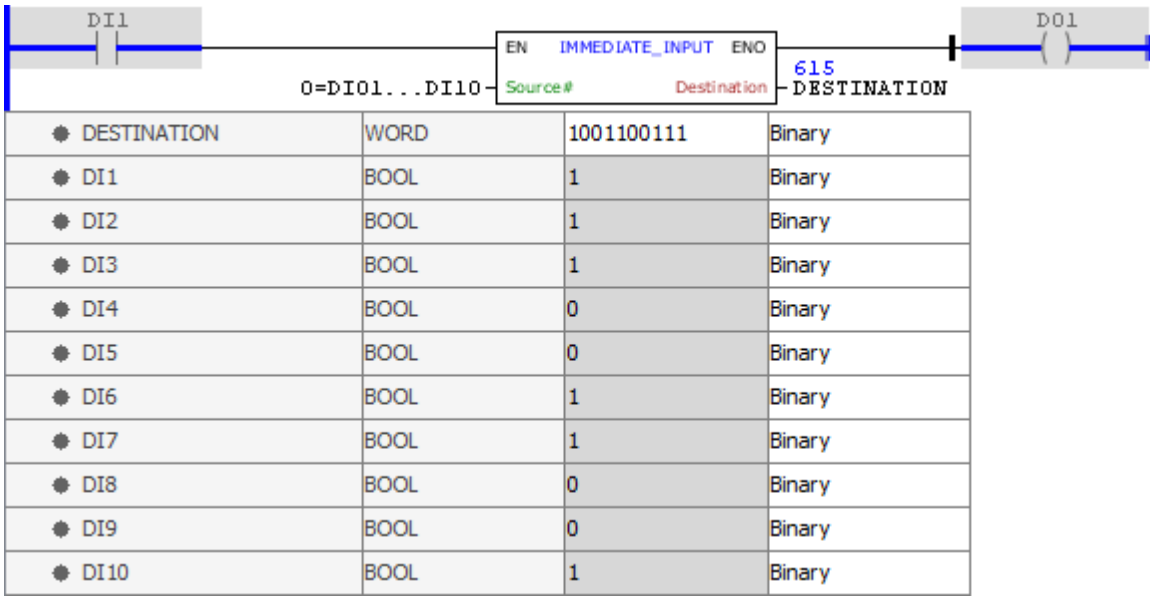
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



O exemplo acima faz uma leitura imediata dos sinais das entradas digitais DI1 a DI10 do PLC300. Essa leitura é então interpretada como uma sequência de binários, com DI1 sendo o bit menos significativo, e o resultado é enviado à variável DESTINATION. Finalizado o bloco com sucesso, a saída ENO é ativada.

10.2.7.1.19.3 IMMEDIATE_OUTPUT

Bloco que realiza uma escrita instantânea na porta da saída selecionada, sem alterar o valor das imagens (variáveis GLOBAL_IO).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source	WORD INT UINT	Variável mapeada com os valores a serem enviados às saídas selecionadas
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Destination#	BYTE	Saídas a serem escritas (digitais ou analógicas)

Funcionamento

Este bloco, quando tem um valor TRUE em EN, escreve imediatamente na saída selecionada o valor de Source. Caso seja selecionada a saída analógica AO1, o valor de Source é repassado à esta. Caso sejam selecionadas as saídas digitais, DO1 receberá o bit zero de Source, DO2 o bit um, DO3 o bit dois, e assim sucessivamente.

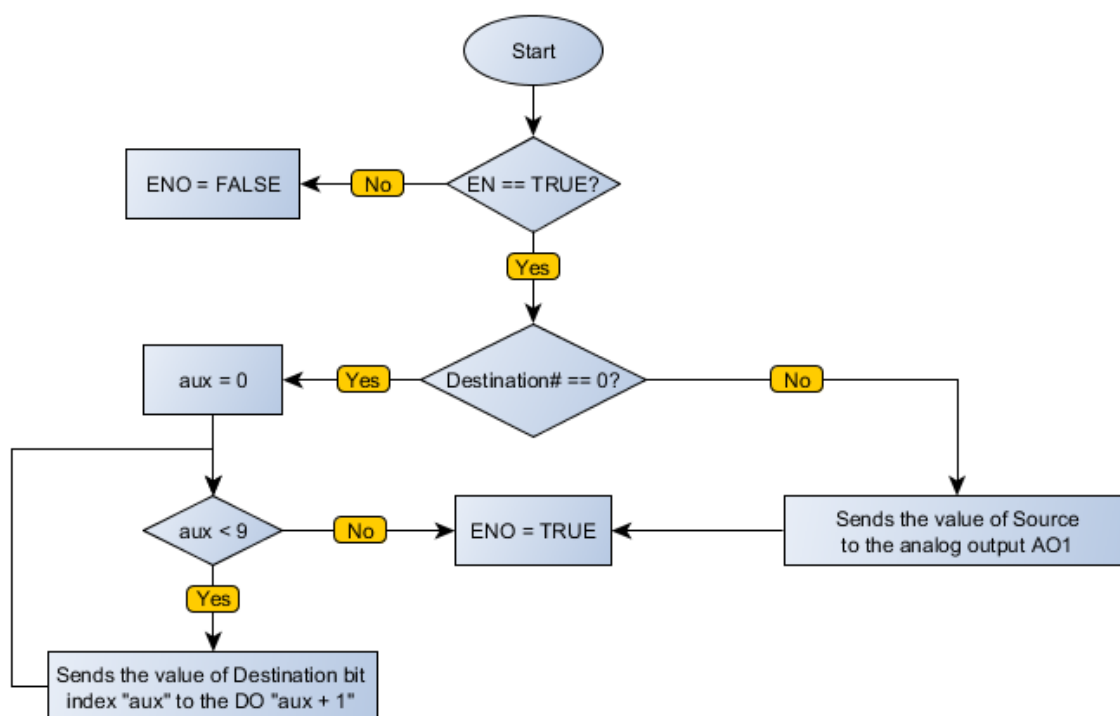
Quando EN possui valor FALSE, Destination# permanece inalterada.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

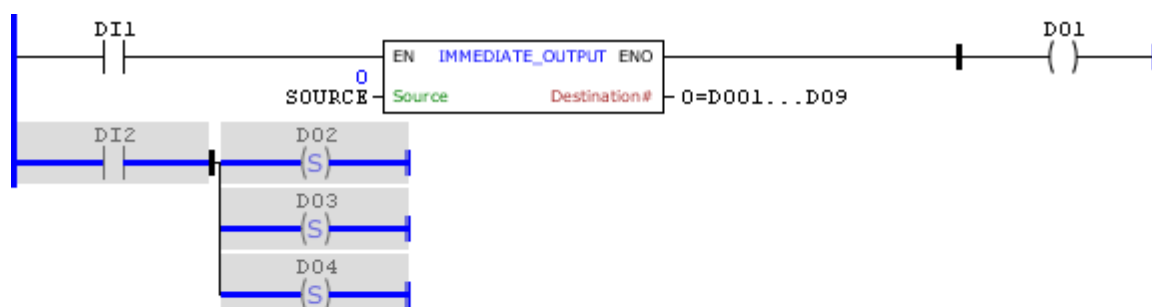
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

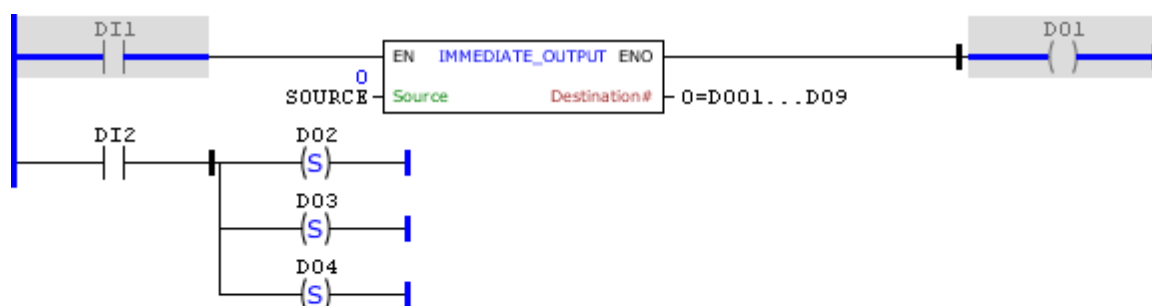
Fluxograma do Bloco



Exemplo



● SOURCE	WORD	0	Binary
● DO1	BOOL	0	Binary
● DO2	BOOL	1	Binary
● DO3	BOOL	1	Binary
● DO4	BOOL	1	Binary
● DO5	BOOL	0	Binary
● DO6	BOOL	0	Binary
● DO7	BOOL	0	Binary
● DO8	BOOL	0	Binary
● DO9	BOOL	0	Binary



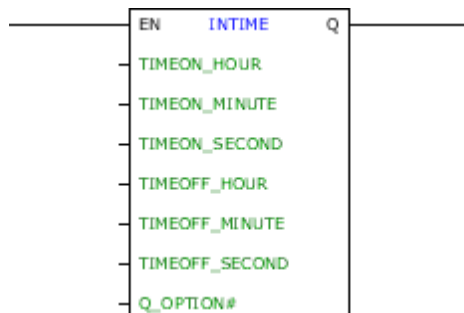
● SOURCE	WORD	0	Binary
● DO1	BOOL	1	Binary
● DO2	BOOL	0	Binary
● DO3	BOOL	0	Binary
● DO4	BOOL	0	Binary
● DO5	BOOL	0	Binary
● DO6	BOOL	0	Binary
● DO7	BOOL	0	Binary
● DO8	BOOL	0	Binary
● DO9	BOOL	0	Binary

O exemplo acima faz uma escrita imediata do valor de SOURCE, interpretada como uma sequência de binários, nas saídas digitais DO1 a DO9 do PLC300, com DO1 recebendo o valor do bit menos significativo. Finalizado o bloco com sucesso, a saída ENO é ativada. Observe que a escrita imediata não prevalece sobre a bobina direta DO1 ou sobre as bobinas de habilitação em DO2, DO3 e DO4.

10.2.7.1.20 RTC

10.2.7.1.20.1 INTIME

Bloco que realiza uma habilitação programada para um horário baseado em RTC (Real Time Clock).

Representação Ladder**Estrutura do Bloco**

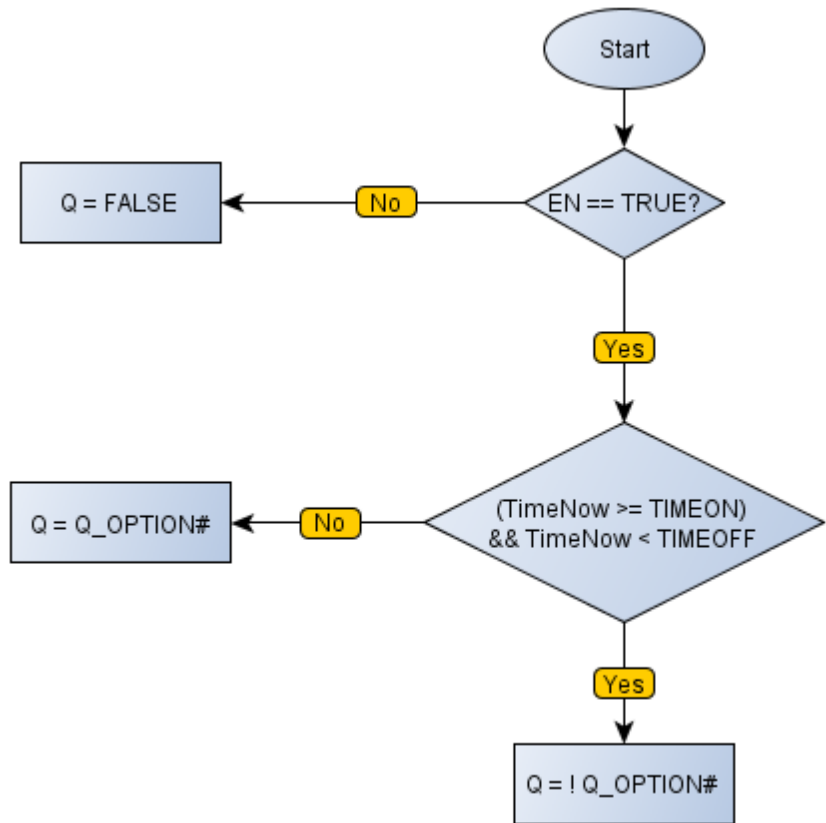
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	TIMEON_HOUR	WORD UINT	Hora de habilitação
	TIMEON_MINUTE	WORD UINT	Minutos de habilitação
	TIMEON_SECOND	WORD UINT	Segundo de habilitação
	TIMEOFF_HOUR	WORD UINT	Hora de desabilitação
	TIMEOFF_MINUTE	WORD UINT	Minutos de desabilitação
	TIMEOFF_SECOND	WORD UINT	Segundo de desabilitação
	Q_OPTION#	BYTE	Funcionamento da saída
VAR_OUTPUT	Q	BOOL	Saída do bloco

Funcionamento

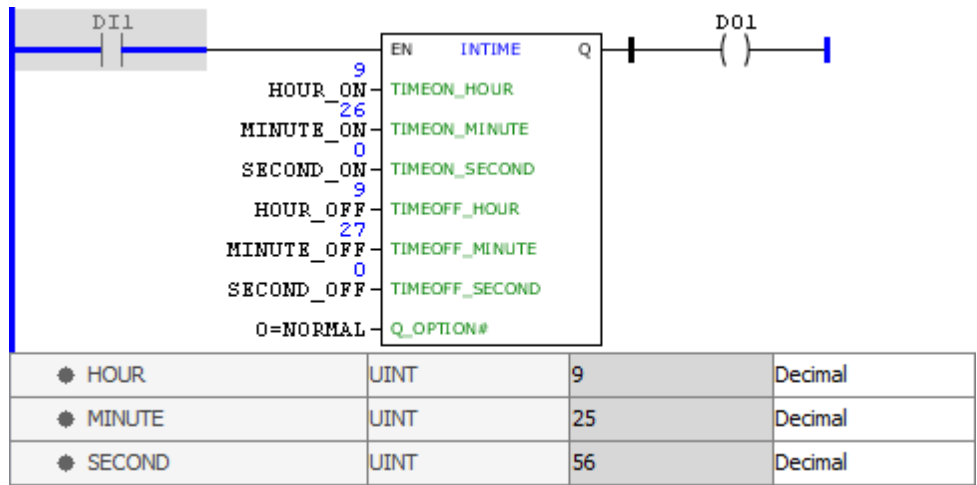
Este bloco, quando tem um valor TRUE em EN, possui dois modos de operação. Se Q_OPTION# for Normal, Q é habilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEON e desabilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEOFF. Se Q_OPTION# for Inverted, Q é desabilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEON e habilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEOFF.

Quando EN possui valor FALSE, Q permanece FALSE.

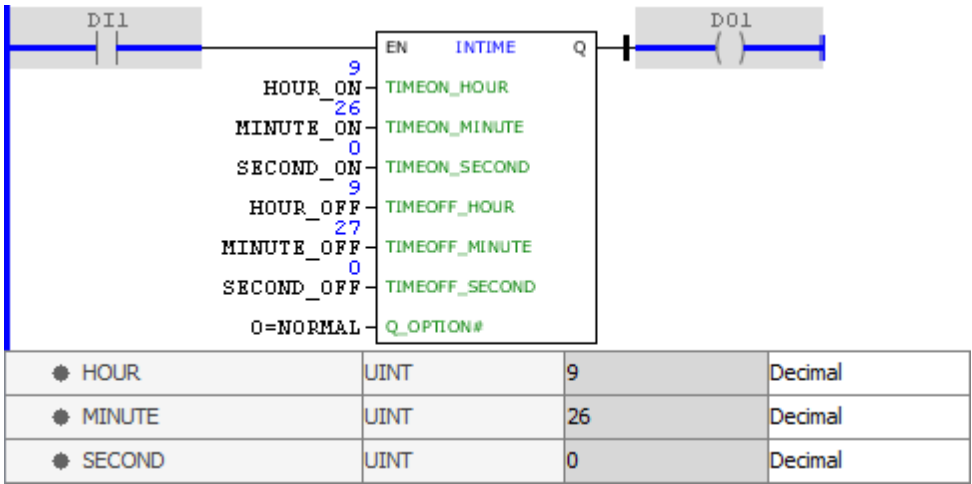
Fluxograma do Bloco



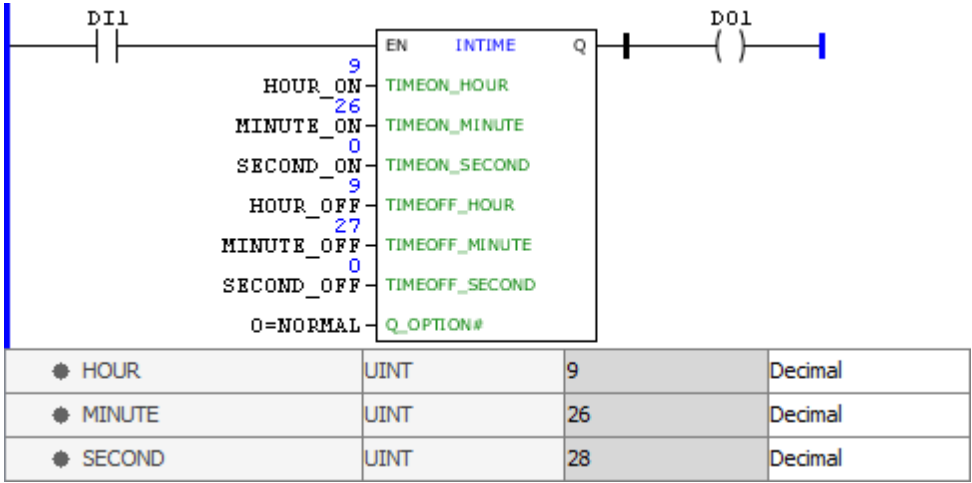
Exemplo



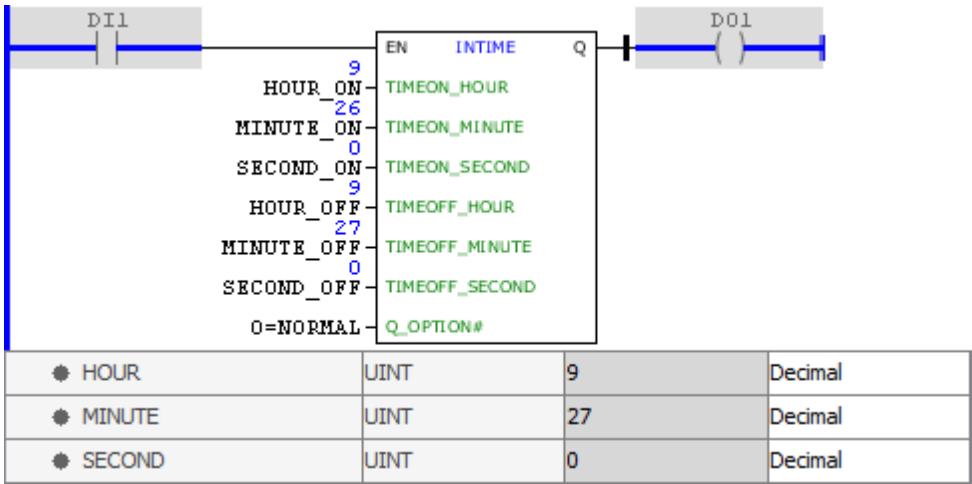
No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é inferior ao cadastrado nas entradas de habilitação do bloco (HOUR_ON, MINUTE_ON e SECOND_ON). Dessa forma, a saída Q é desabilitada.



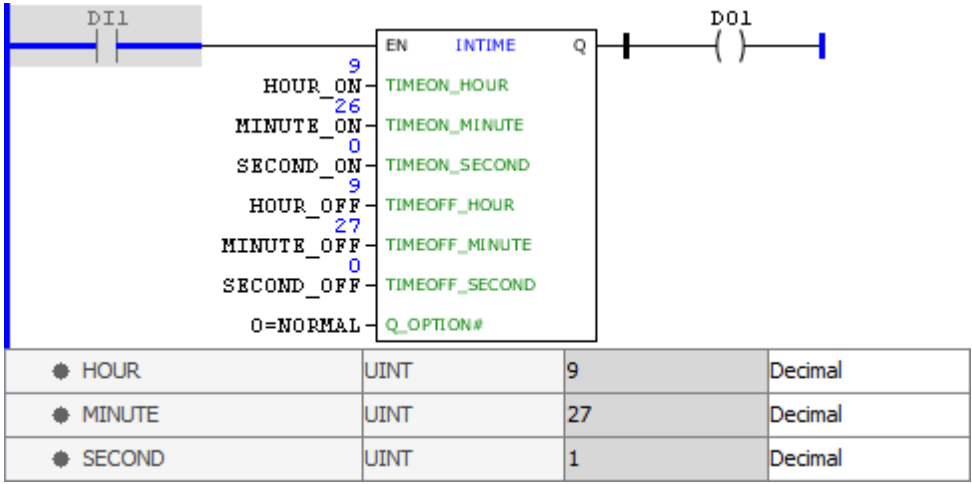
No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é igual ao cadastrado nas entradas de habilitação do bloco (HOUR_ON, MINUTE_ON e SECOND_ON). Dessa forma, a saída Q é habilitada.



No exemplo acima, o bloco INTIME está desabilitado. Dessa forma, independente das entradas, a saída Q é desabilitada.



No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é igual ao cadastrado nas entradas de desabilitação do bloco (HOUR_OFF, MINUTE_OFF e SECOND_OFF). Dessa forma, a saída Q é habilitada.

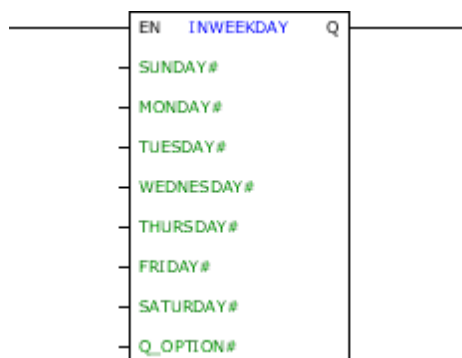


No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é superior ao cadastrado nas entradas de desabilitação do bloco (HOUR_OFF, MINUTE_OFF e SECOND_OFF). Dessa forma, a saída Q é desabilitada.

10.2.7.1.20.2 INWEEKDAY

Bloco que realiza uma habilitação programada para dias da semana baseado em RTC (Real Time Clock).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SUNDAY#	BOOL	Habilitado no domingo
	MONDAY#	BOOL	Habilitado na segunda-feira
	TUESDAY#	BOOL	Habilitado na terça-feira
	WEDNESDAY#	BOOL	Habilitado na quarta-feira
	THURSDAY#	BOOL	Habilitado na quinta-feira
	FRIDAY#	BOOL	Habilitado na sexta-feira
	SATURDAY#	BOOL	Habilitado no sábado
	Q_OPTION#	BYTE	Funcionamento da saída
VAR_OUTPUT	Q	BOOL	Saída do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, possui dois modos de operação. Se Q_OPTION# for Normal, Q é habilitada se o dia da semana do relógio interno possuir parâmetro Enabled no bloco. Se Q_OPTION# for Inverted, Q é desabilitada se o dia da semana do relógio interno possuir parâmetro Enabled no bloco.

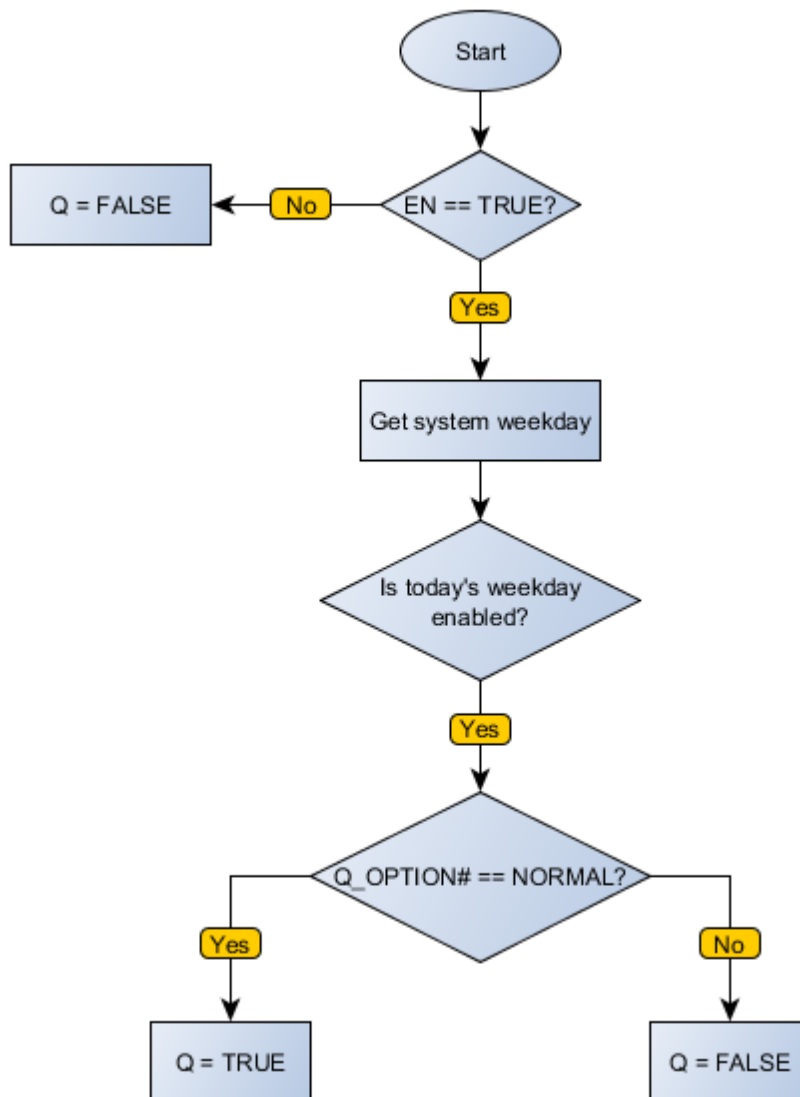
Quando EN possui valor FALSE, Q permanece FALSE.



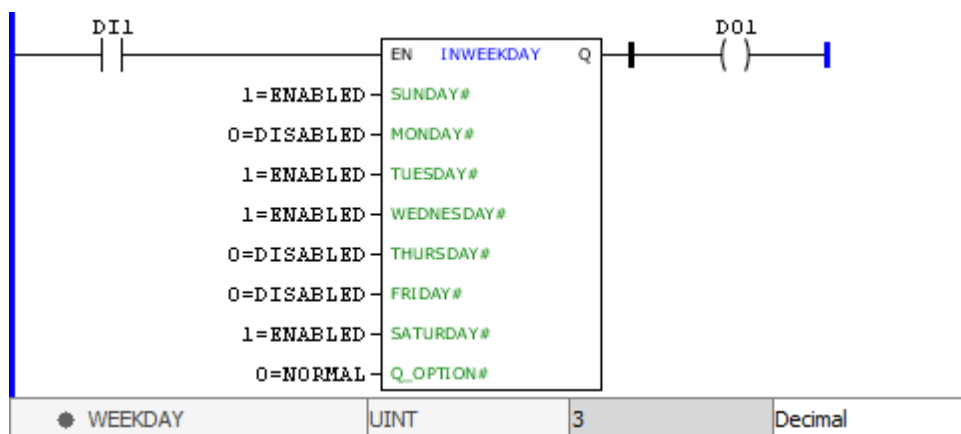
NOTA!

Os dias da semana são identificados por números, sendo Domingo o dia 0 e Sábado o dia 6.

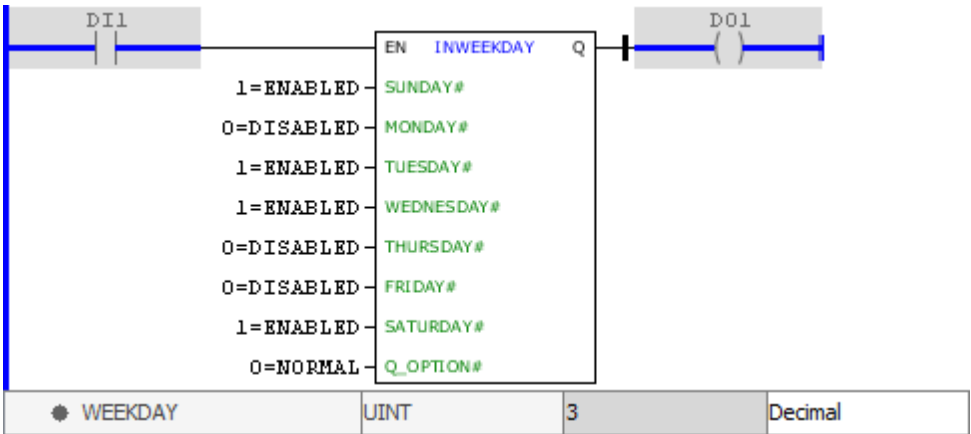
Fluxograma do Bloco



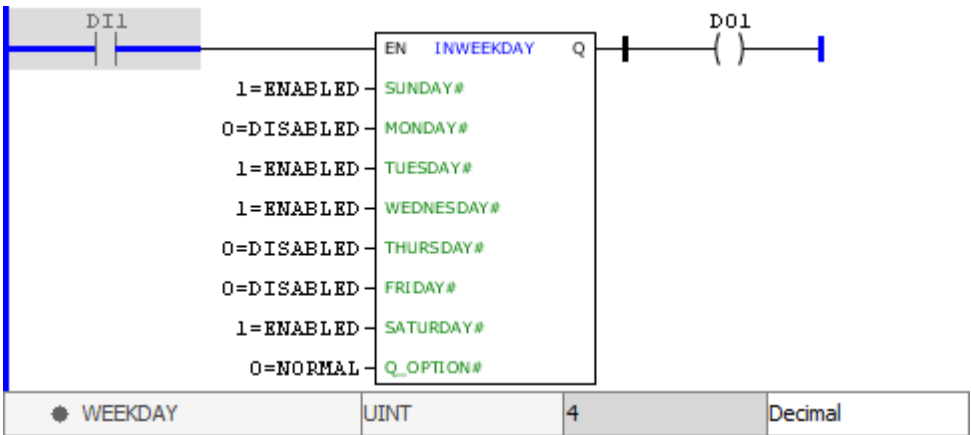
Exemplo



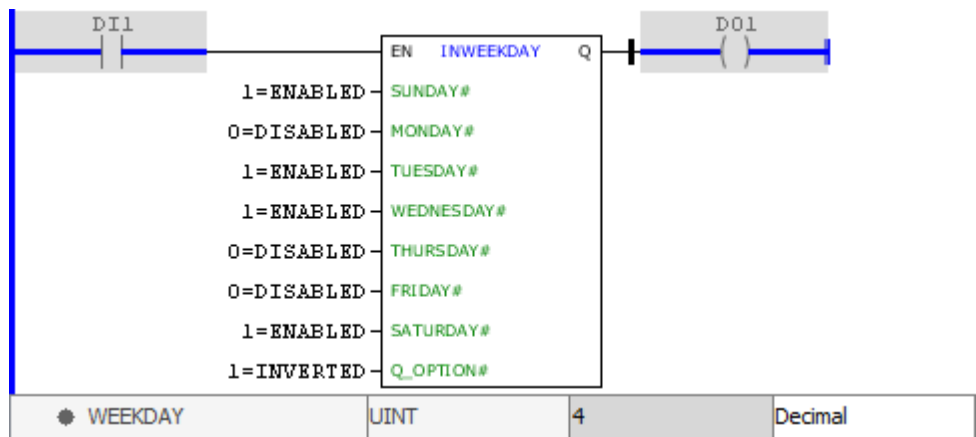
No exemplo acima, o bloco INWEEKDAY está desabilitado. Dessa forma, independente das entradas, a saída Q é desabilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação NORMAL. O dia da semana atual do relógio interno do equipamento é quarta-feira (valor 3), que possui estado ENABLED na programação. Dessa forma, a saída Q é habilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação NORMAL. O dia da semana atual do relógio interno do equipamento é quinta-feira (valor 4), que possui estado DISABLED na programação. Dessa forma, a saída Q é desabilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação INVERTED. O dia da semana atual do relógio interno do equipamento é quinta-feira (valor 4), que possui estado DISABLED na programação. Dessa forma, a saída Q é habilitada.

10.2.7.1.21 Communication Network

10.2.7.1.21.1 CANopen

Operação na Rede CANopen - Modo Mestre

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

Habilitação da Função CANopen Mestre

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

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



NOTA!

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

Características do Mestre CANopen

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

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

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



NOTA!

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

Operação do Mestre

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

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

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

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

**NOTA!**

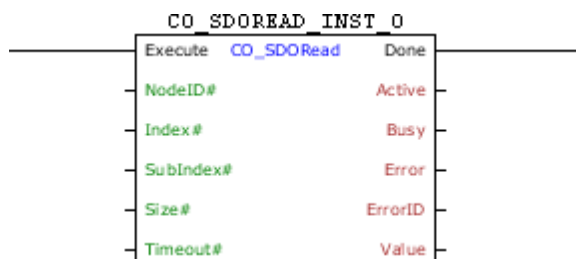
O estado da comunicação e o estado de cada escravo podem ser observados em marcadores de sistema de entrada.

Blocos para o Mestre CANopen

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

Bloco que executa uma leitura de dados via SDO de um escravo remoto em rede CANopen.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NodeID#	BYTE	Endereço do escravo
	Index#	WORD	Índice do objeto a ser acessado no escravo
	SubIndex#	BYTE	Sub-índice do objeto
	Size#	BYTE	Tamanho do dado acessado, em bytes
	Timeout#	WORD	Tempo de espera máximo para chegada do dado, a partir do início da solicitação [ms]
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que o cliente SDO está ocupado com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE USINT	Identificador do erro ocorrido
	Value	BYTE USINT	Variável que armazena os dados recebidos
VAR	CO_SDOREAD_INST_0	CO_SDOREAD	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o cliente SDO no endereço especificado em NodeID# está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de leitura para o objeto de tamanho Size# localizado em Index# e SubIndex# e seta a saída Active, resetando-a ao receber a resposta do escravo. Os dados recebidos são armazenados na variável Value. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.

**NOTA!**

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.

**NOTA!**

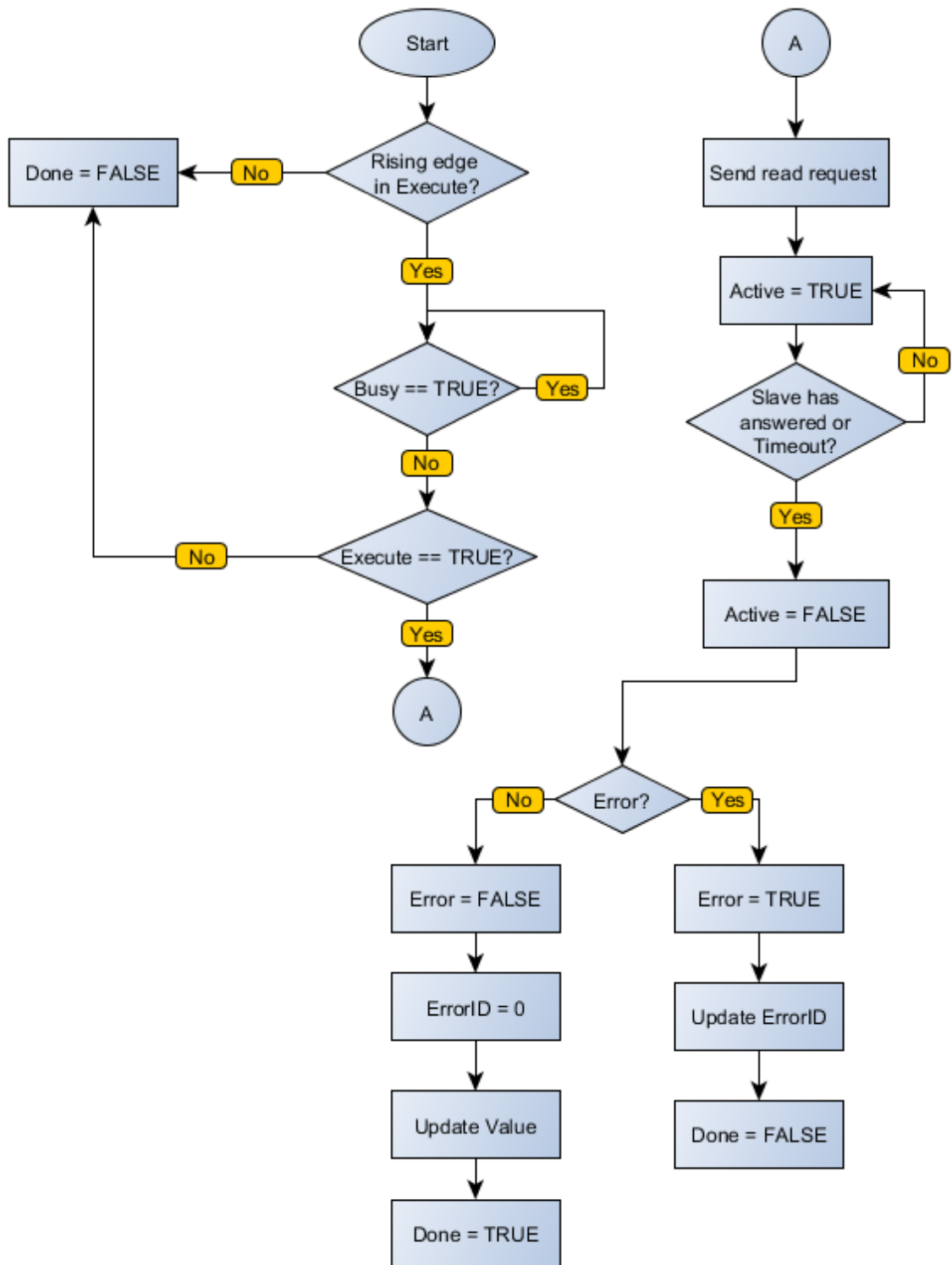
Value é um array de tamanho igual a Size#. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

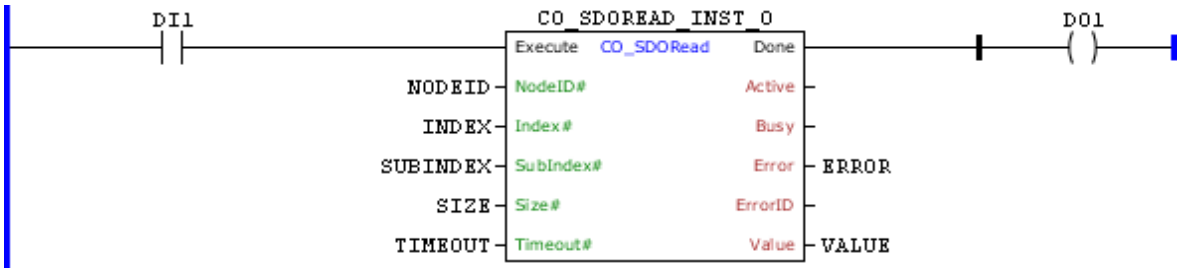
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Cartão não pode executar a função
2	Timeout na resposta do escravo
3	Escravo retornou erro

Fluxograma do Bloco



Exemplo



O exemplo acima requisita leitura do dado de tamanho SIZE, localizado em INDEX - SUBINDEX, do equipamento NODEID. Esse dado é repassado a VALUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma escrita de dados via SDO de um escravo remoto em rede CANopen.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NodeID#	BYTE	Endereço do escravo
	Index#	WORD	Índice do objeto a ser acessado no escravo
	SubIndex#	BYTE	Sub-índice do objeto
	Size#	BYTE	Tamanho do dado acessado, em bytes
	Timeout#	WORD	Tempo de espera máximo para chegada do dado, a partir do início da solicitação [ms]
	Value	BYTE USINT	Variável que possui os dados a serem escritos
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que o cliente SDO está ocupado com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE USINT	Identificador do erro ocorrido
VAR	CO_SDOWRITE_INST_0	CO_SDOWRITE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o cliente SDO no endereço especificado em NodeID# está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de escrita para o objeto de tamanho Size# localizado em Index# e SubIndex# e seta a saída Active, resetando-a ao receber a resposta do escravo. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.

**NOTA!**

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.

**NOTA!**

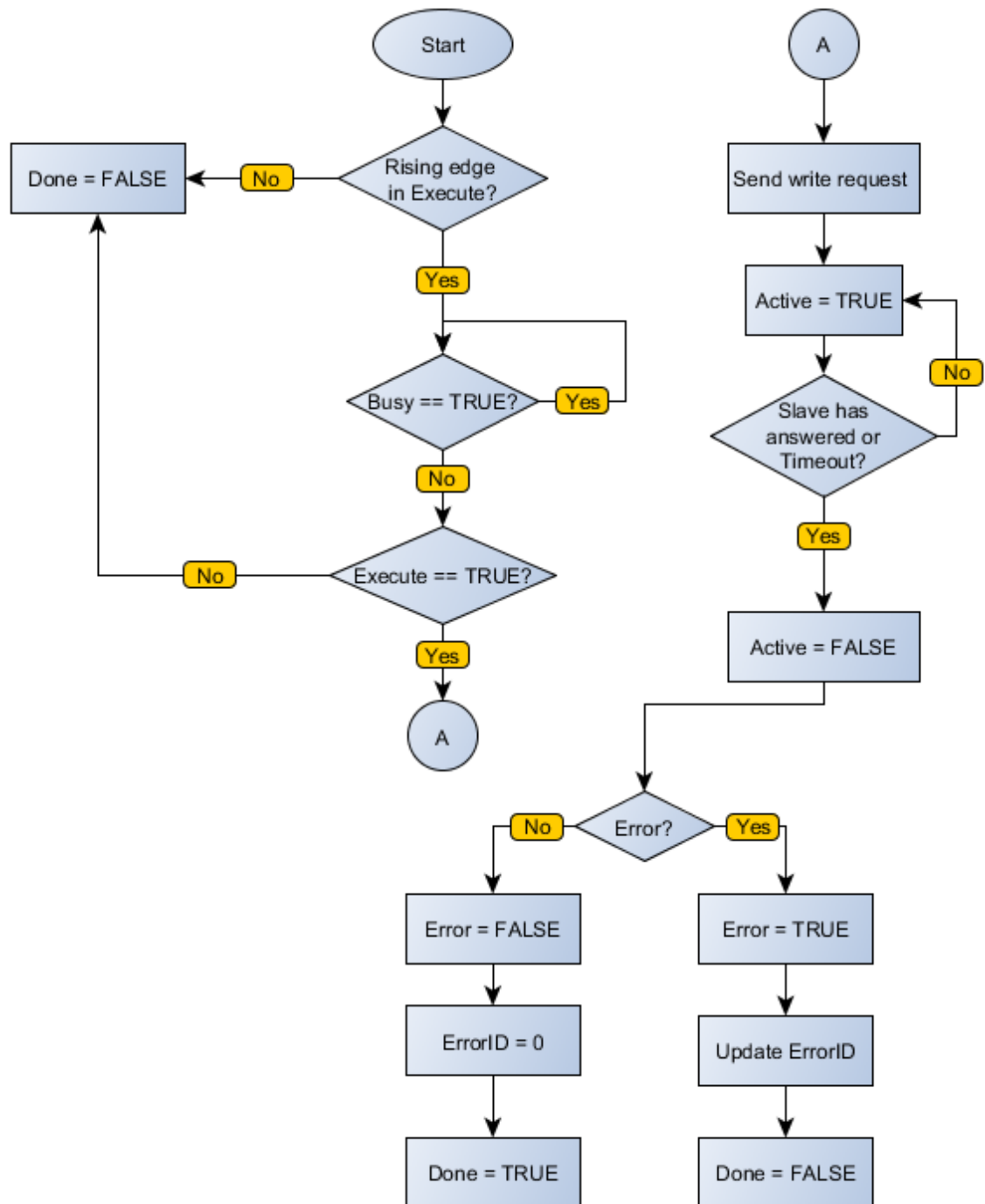
Value é um array de tamanho igual a Size#. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

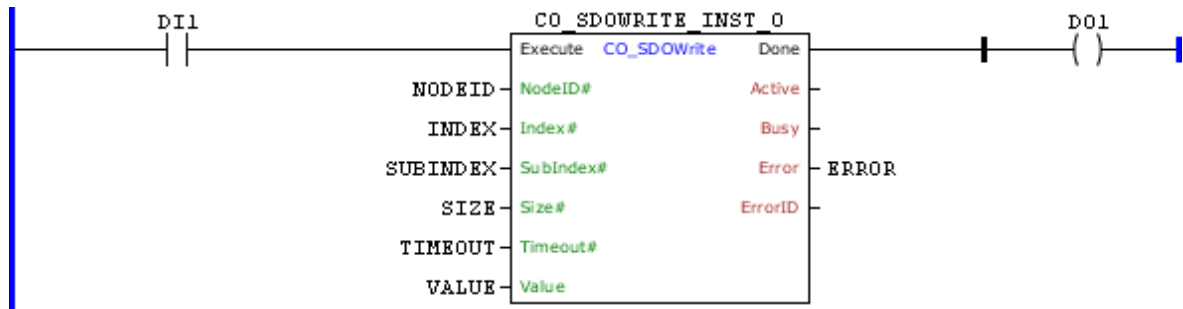
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Cartão não pode executar a função
2	Timeout na resposta do escravo
3	Escravo retornou erro

Fluxograma do Bloco



Exemplo



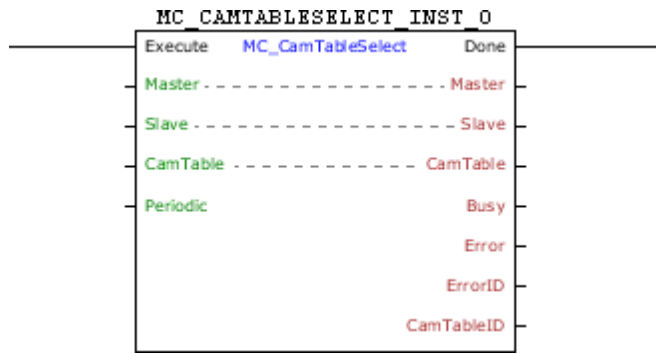
O exemplo acima requisita escrita do dado contido em VALUE, de tamanho SIZE, no local INDEX - SUBINDEX do equipamento NODEID. Finalizado o bloco com sucesso, a saída Done é ativada.

10.2.7.1.22 Motion Control Cam

10.2.7.1.22.1 MC_CamTableSelect

Bloco que seleciona uma tabela de pontos de uma curva CAM previamente programada através da ferramenta CAM Profiles.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	44 Bytes
Tamanho de Memória de Dados	16 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
	CamTable	WORD	Código da tabela CAM
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Periodic	BOOL	Controle de execução da requisição (0 - Execução única) (1 - Execução periódica)
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
	CamTableID	WORD	Tabela CAM selecionada
VAR	MC_CAMTABLESELECT_INST_0	MC_CAMTABLESELECT	Instância de acesso à estrutura do bloco

Funcionamento

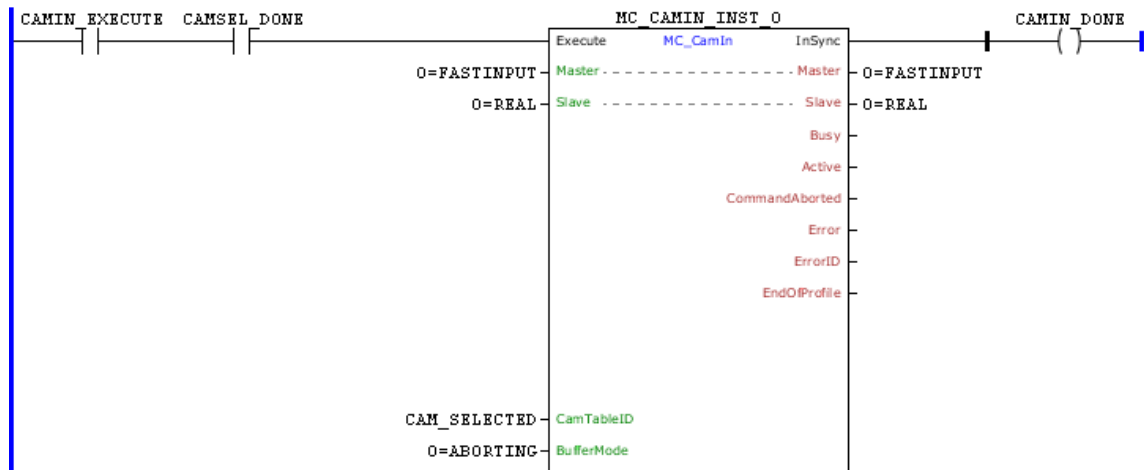
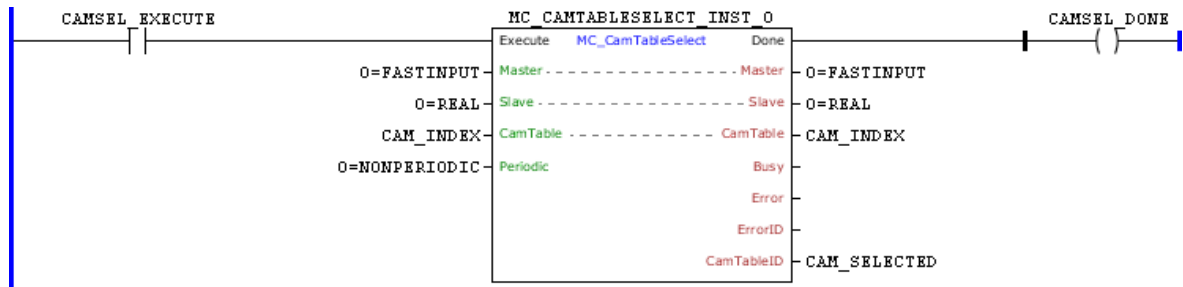
Este bloco, quando detecta uma borda de subida em Execute, busca a tabela especificada em CamTable para que esta possa ser utilizada pelo bloco MC_CamIn.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
83	Arquivo das tabelas de pontos da curva CAM inválido.
84	CAM Table inválido. CAM Table deve ser de 1 à 10.

Exemplo



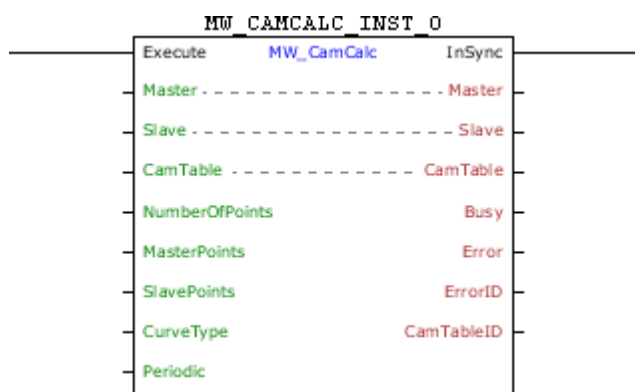
LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_CAMTABLESELECT_INST_0	0	MC_CamTableSelect		
CAMSEL_EXECUTE	0	BOOL	0	
CAM_INDEX	0	WORD	3	
CAM_SELECTED	0	WORD	3	
CAMSEL_DONE	0	BOOL	0	
MC_CAMIN_INST_0	0	MC_CamIn		
CAMIN_EXECUTE	0	BOOL	0	
CAMIN_DONE	0	BOOL	0	

Na transição de subida de CAMSEL_EXECUTE, o bloco MC_CamTableSelect é executado e, com isso, a tabela de pontos selecionada em CAM_INDEX poderá ser utilizada pelo bloco MC_CamIn. Ao executar o bloco, a saída Done e CAMSEL_DONE são setados e permanecem em nível TRUE enquanto a entrada Execute está setada. Nesse exemplo, CAMSEL_DONE garante que o bloco MC_CamIn não será acionado antes do bloco MC_CamTableSelect ser executado com sucesso.

10.2.7.1.22.2 MW_CamCalc

Bloco que calcula uma tabela de pontos de uma curva CAM.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	66 Bytes
Tamanho de Memória de Dados	24 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
	CamTable	WORD	Código da tabela CAM
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NumberOfPoints	WORD	Número de pontos da tabela
	MasterPoints	LREAL	Pontos do mestre
	SlavePoints	LREAL	Pontos do escravo
	CurveType	WORD	Tipo de curva
	Periodic	BOOL	Controle de execução da requisição (0 - Execução única) (1 - Execução periódica)
VAR_OUTPUT	InSync	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
	CamTableID	WORD	Tabela CAM selecionada
VAR	MW_CAMCALC_INST_0	MW_CAMCALC	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, constrói a tabela especificada em

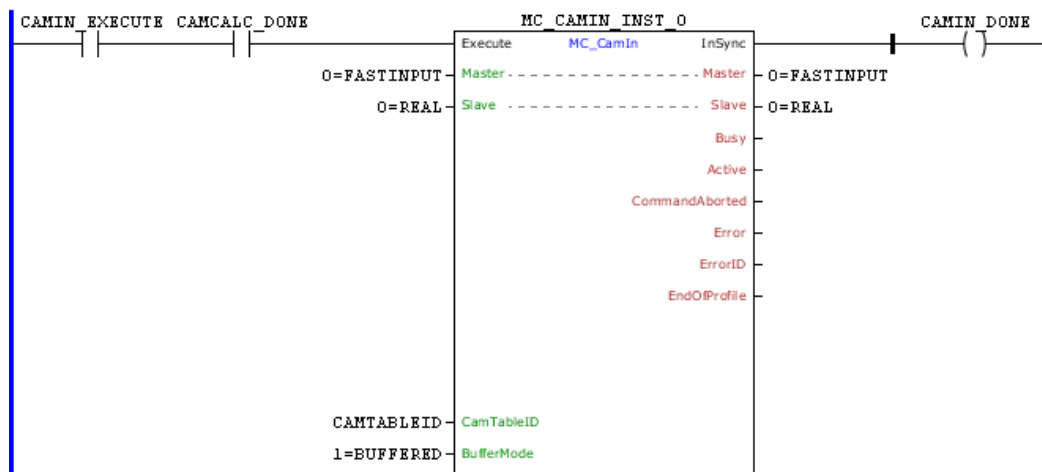
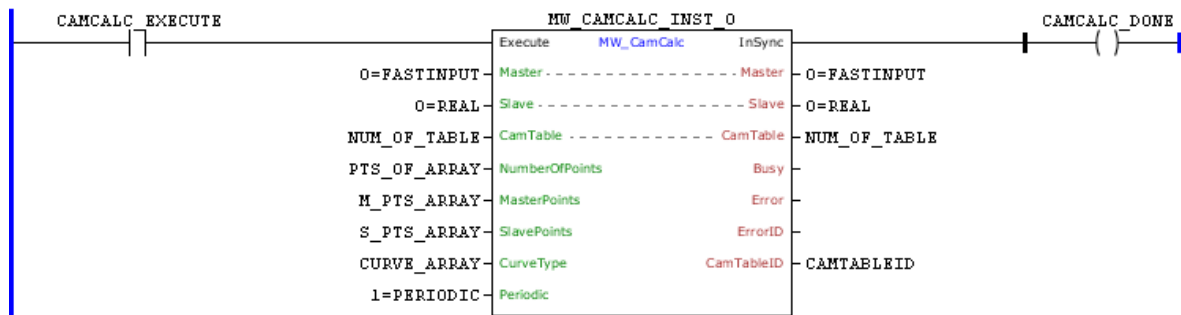
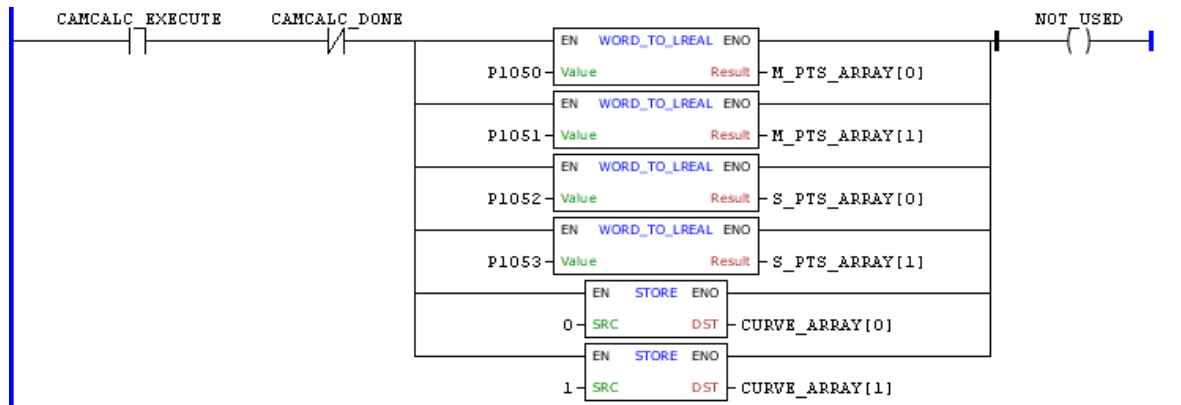
CamTable com os dados programados para que esta possa ser utilizada pelo bloco MC_CamIn.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
83	Arquivo das tabelas de pontos da curva CAM inválido.
84	CAM Table inválido. CAM Table deve ser de 11 à 20.
86	Número de pontos maior que o programado no configurador CAM PROFILES.
87	Posição do eixo mestre inválida. A posição do eixo mestre deve ser maior que a posição do ponto anterior.
88	Bloco MW_CamCalc em execução. Somente é permitido a execução de um bloco MW_CamCalc de cada vez.
89	Tabela de pontos em uso pelo bloco MC_CamIn.
90	Marcador de double com posição do eixo mestre inexistente.
91	Marcador de double com posição do eixo escravo inexistente.
92	Marcador de word com tipo de curva inexistente.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
NOT_USED	0	BOOL	0	
MW_CAMCALC_INST_0	0	MW_CamCalc		
CAMCALC_EXECUTE	0	BOOL	0	
CAMCALC_DONE	0	BOOL	0	
CAMTABLEID	0	WORD	11	
PTS_OF_ARRAY	0	WORD	2	
M_PTS_ARRAY	2	LREAL		
M_PTS_ARRAY[0]	0	LREAL	3.0	
M_PTS_ARRAY[1]	0	LREAL	7.0	
S_PTS_ARRAY	2	LREAL		
S_PTS_ARRAY[0]	0	LREAL	10.0	
S_PTS_ARRAY[1]	0	LREAL	-5.0	
CURVE_ARRAY	2	WORD		
CURVE_ARRAY[0]	0	WORD	0	
CURVE_ARRAY[1]	0	WORD	1	
NUM_OF_TABLE	0	WORD	11	
MC_CAMIN_INST_0	0	MC_CamIn		
CAMIN_EXECUTE	0	BOOL	0	
CAMIN_DONE	0	BOOL	0	

GLOBAL LOCAL

CAM Profile List			
Cam Table	Cam Type	Cam File	Max Points
1	Fixed	---	0
2	Fixed	---	0
3	Fixed	---	0
4	Fixed	---	0
5	Fixed	---	0
6	Fixed	---	0
7	Fixed	---	0
8	Fixed	---	0
9	Fixed	---	0
10	Fixed	---	0
11	Calculable		2
12	Calculable		0
13	Calculable		0
14	Calculable		0
15	Calculable		0
16	Calculable		0
17	Calculable		0
18	Calculable		0
19	Calculable		0
20	Calculable		0

Editar Perfil

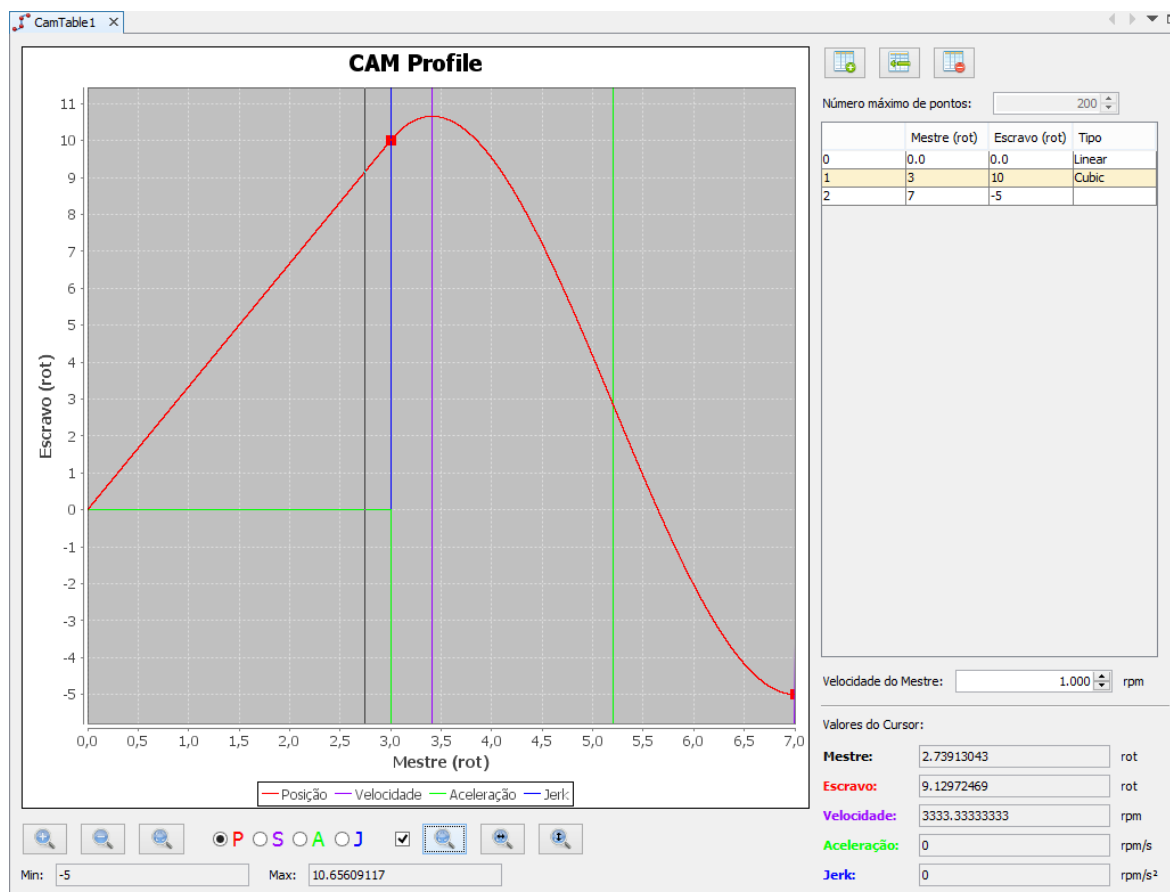
Selecionar Perfil

Desconsiderar Perfil

Na transição de subida do marcador de bit CAMCALC_EXECUTE, o bloco MW_CamCalc é executado e a tabela de pontos de TABLE será calculada de acordo com os argumentos do bloco.

Nesse exemplo, o número de pontos da curva será o conteúdo de NUM_POINTS, a posição do eixo mestre será de acordo com os conteúdos de MASTER_POINTS[0] e MASTER_POINTS[1], a posição do eixo escravo será de acordo com os conteúdos de SLAVE_POINTS[0] e SLAVE_POINTS[1] e o tipo da curva será de acordo com os conteúdos de CURVE_TYPE[0] e CURVE_TYPE[1].

Colocando os mesmos valores na ferramenta CAM Profiles podemos observar a curva abaixo:



Ao finalizar o cálculo da tabela de pontos, a saída InSync é setada enquanto a entrada Execute permanece setada.

Com CAMCALC_DONE setado, o bloco MC_CamIn poderá ser executado.

10.2.7.1.22.3 MC_CamIn

Bloco responsável pela execução de um posicionamento definido por uma tabela de pontos de uma curva CAM.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	92 Bytes
Tamanho de Memória de Dados	52 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	CamTableID	WORD	Tabela CAM selecionada
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InSync	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
	EndOfProfile	BOOL	Sinalizador de fim do perfil CAM
VAR	MC_CAMIN_INST_0	MC_CAMIN	Instância de acesso à estrutura do bloco

Funcionamento

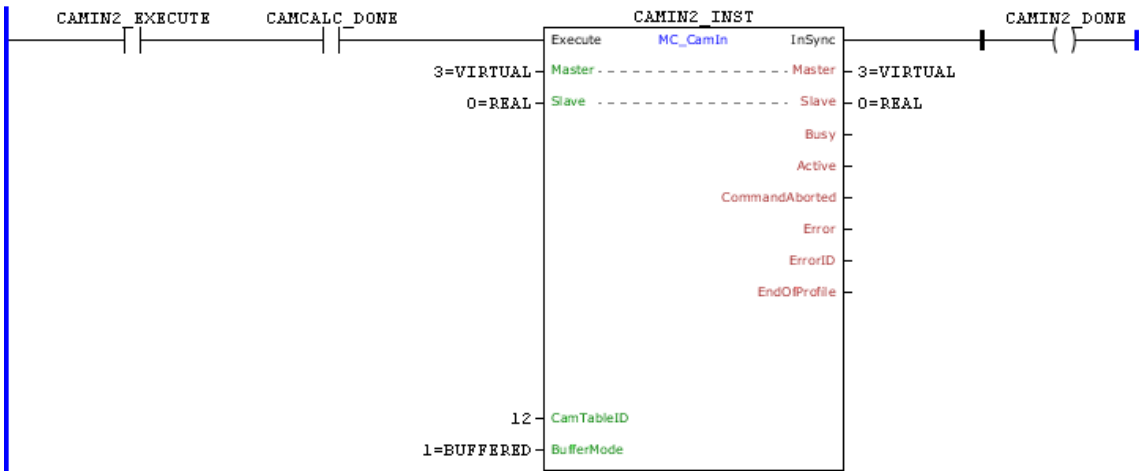
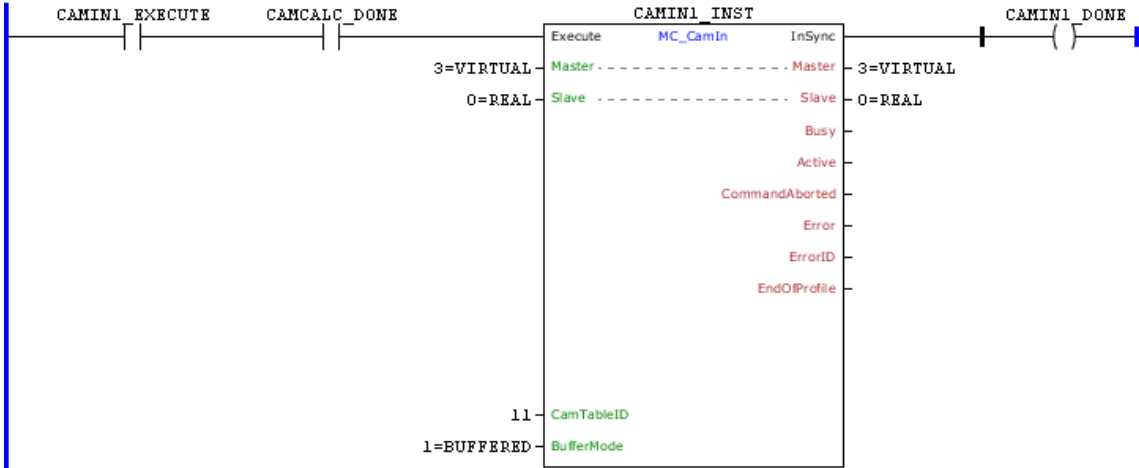
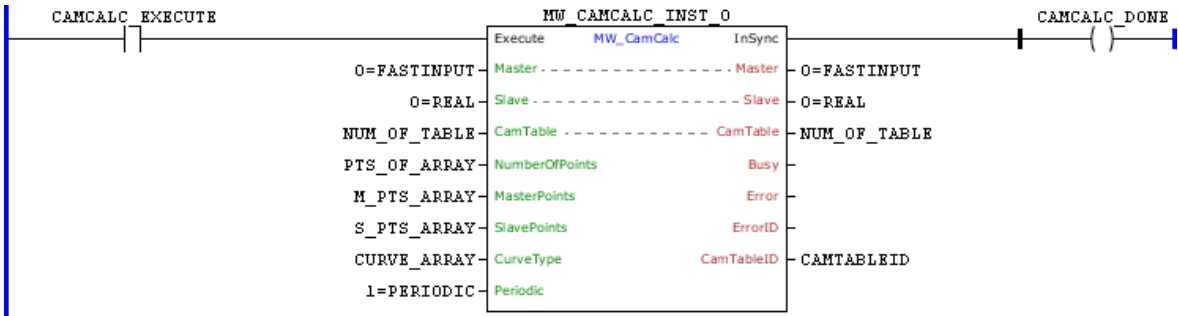
Este bloco, quando detecta uma borda de subida em Execute, envia um comando para o *drive* executar um posicionamento definido por CamTableID.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
85	CamTableID inválido.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
◆ CAMIN1_INST	0	MC_CamIn		
◆ CAMIN1_EXECUTE	0	BOOL	0	
◆ CAMIN1_DONE	0	BOOL	0	
◆ CAMIN2_INST	0	MC_CamIn		
◆ CAMIN2_EXECUTE	0	BOOL	0	
◆ CAMIN2_DONE	0	BOOL	0	
◆ MW_CAMCALC_INST_0	0	MW_CamCalc		
◆ NUM_OF_TABLE	0	WORD	11	
◆ PTS_OF_ARRAY	0	WORD	2	
◆ M_PTS_ARRAY	2	LREAL		
◆ M_PTS_ARRAY[0]	0	LREAL	3.0	
◆ M_PTS_ARRAY[1]	0	LREAL	7.0	
◆ S_PTS_ARRAY	2	LREAL		
◆ S_PTS_ARRAY[0]	0	LREAL	10.0	
◆ S_PTS_ARRAY[1]	0	LREAL	-5.0	
◆ CURVE_ARRAY	2	WORD		
◆ CURVE_ARRAY[0]	0	WORD	0	
◆ CURVE_ARRAY[1]	0	WORD	1	
◆ CAMTABLEID	0	WORD	11	
GLOBAL LOCAL				

Na transição de subida de CAMCALC_EXECUTE, o bloco MW_CamCalc é executado e a tabela de pontos NUM_OF_TABLE será calculada de acordo com os argumentos do bloco.

Ao finalizar o cálculo da tabela de pontos, a saída InSync é setada enquanto a entrada Execute permanece setada.

Com CAMCALC_DONE setado, o bloco MC_CamIn poderá ser executado.

Na transição de subida de CAMIN1_EXECUTE, o primeiro bloco MC_CamIn é executado.

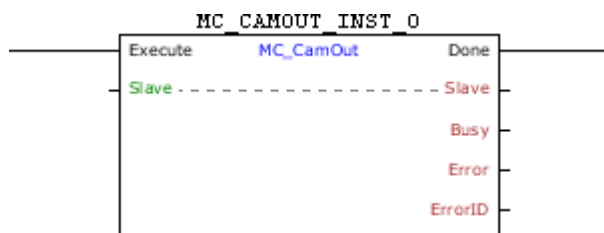
Caso houver a necessidade de algum ajuste na tabela de pontos da curva CAM, basta fazer o ajuste em MASTER_PT[0], MASTER_PT[1], SLAVE_PT[0] e SLAVE_PT[1], mudar o conteúdo de NUM_OF_TABLE para 12 e executar novamente o bloco MW_CamCalc.

Na transição de subida de CAMIN2_EXECUTE, o segundo bloco MC_CamIn será executado (sem perda de posição do eixo mestre) assim que o primeiro bloco MC_CamIn terminar de executar a curva em execução.

10.2.7.1.22.4 MC_CamOut

Bloco responsável pela finalização de uma sincronia estabelecida por um bloco MC_CamIn.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	28 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Slave	BYTE	Seleção de escravo da operação (0 - Exo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	CamTableID	WORD	Tabela CAM selecionada
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_CAMOUT_INST_0	MC_CAMOUT	Instância de acesso à estrutura do bloco

Funcionamento

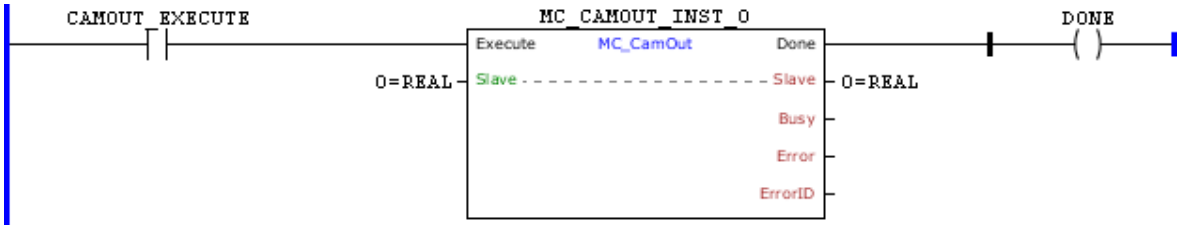
Este bloco, quando detecta uma borda de subida em Execute, finaliza o sincronismo existente pela última execução de um bloco MC_CamIn. O eixo manterá a velocidade do instante em que o bloco é executado.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
73	Drive não está no estado "Synchronized Motion"
78	Bloco MC não executado - Falha interna.

Exemplo



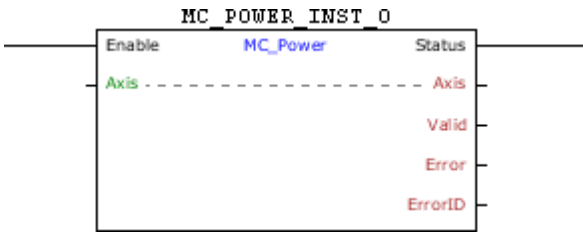
Na transição de subida de CAMOUT_EXECUTE, o bloco MC_CamOut é executado. Com isso, os sinais Busy e Active deste bloco são setados e finaliza-se o sincronismo iniciado por outros blocos CAM. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.2.7.1.23 Motion Control Command

10.2.7.1.23.1 MC_Power

Bloco responsável pela habilitação/desabilitação de eixo do *drive*.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	40 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Enable	BOOL	Habilitação do bloco
VAR_OUTPUT	Status	BOOL	Habilitação da saída
	Valid	BOOL	Sinalizador de validade dos sinais de saída
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_POWER_INST_0	MC_POWER	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco realiza um comando de habilitação/deshabilitação do eixo Axis de acordo com a entrada Enable, desabilitando se Enable for FALSE e habilitando se Enable for TRUE.



NOTA!

Quando o bloco MC_Power for usado para habilitação/deshabilitação do eixo real, nenhuma entrada digital deverá estar programada para a função de Habilitação (opção 1), ou o Alarme A0120 poderá ocorrer.

Ao habilitar o eixo real pela primeira vez, o *drive* poderá operar em malha de posição, dependendo do valor do parâmetro P0773. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

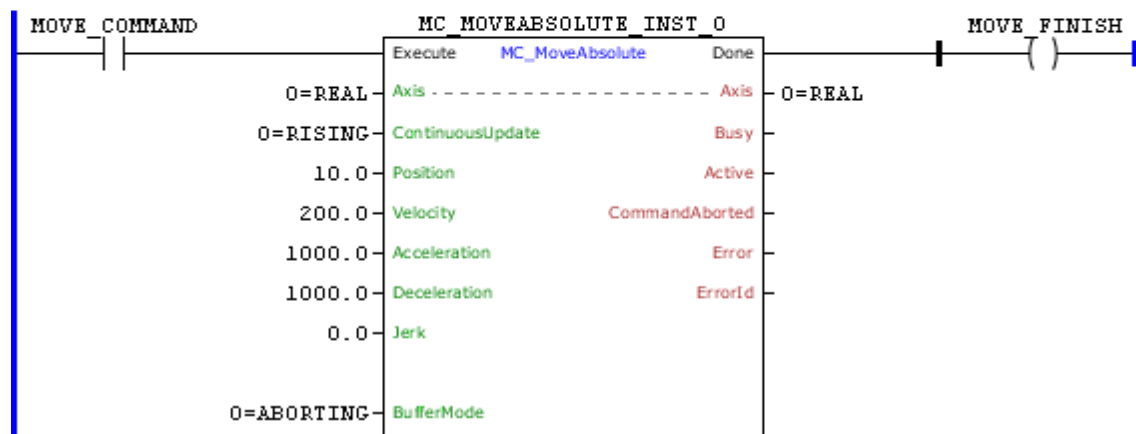
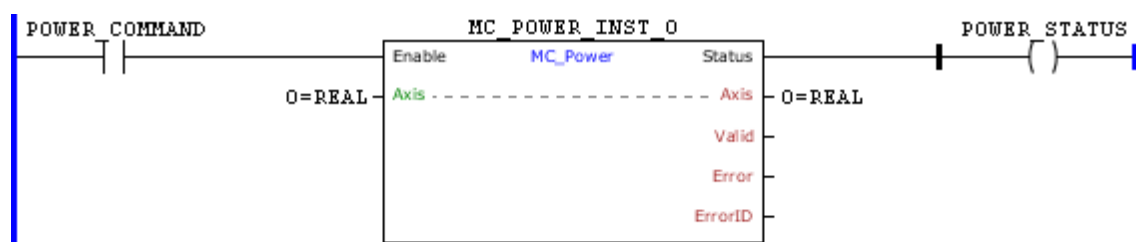
Quando o eixo real estiver desabilitado, o estado do eixo será Disabled. Ao habilitar o eixo real, o estado do eixo mudará para Standstill.

Quando Enable possui valor FALSE, Status permanece FALSE. A saída Status é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Enable receba FALSE.

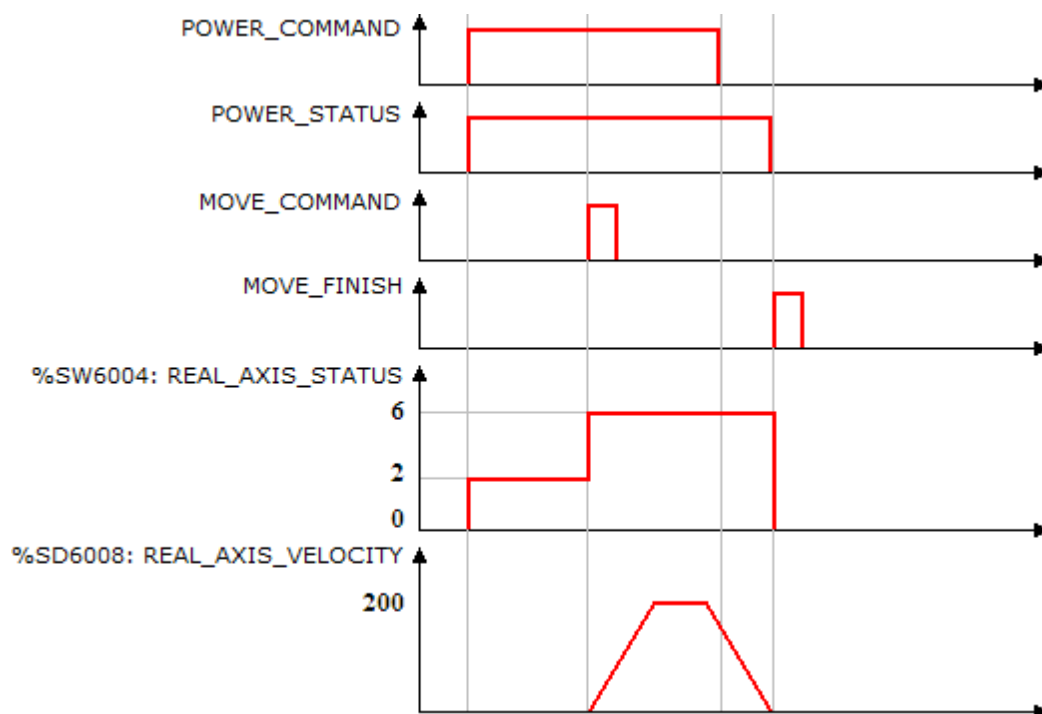
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
66	Drive no estado "ErrorStop".
71	P202 diferente de 4.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_POWER_INST_0	0	MC_Power		
POWER_COMMAND	0	BOOL	0	
MC_MOVEABSOLUTE_INST_0	0	MC_MoveAbsolute		
POWER_STATUS	0	BOOL	0	
MOVE_COMMAND	0	BOOL	0	
MOVE_FINISH	0	BOOL	0	



Na borda de subida de POWER_COMMAND, o eixo real é habilitado e o seu estado, o marcador do sistema REAL_AXIS_STATUS (%SW6004), é alterado para Standstill (%SW6004 = 2). A saída Status é setada.

Tendo-se a transição de subida de MOVE_COMMAND, o bloco MC_MoveAbsolute é executado e inicia-se o posicionamento para a posição 10 voltas. O estado do eixo é alterado para Discrete Motion (%SW3406 = 6).

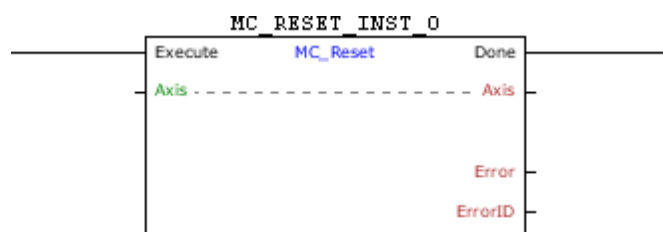
Enquanto o posicionamento é executado, POWER_COMMAND é resetado. Como o BufferMode do MC_Power está configurado como Buffered, o eixo somente será desabilitado na conclusão do posicionamento.

Ao finalizar o posicionamento, a saída Done do bloco MC_MoveAbsolute é setada por 1 ciclo de scan e o eixo é desabilitado (MC_Power.Enable = 0). O estado do eixo é alterado para Disabled (%SW6004 = 0).

10.2.7.1.23.2 MC_Reset

Bloco responsável pela limpeza do estado de falha do *drive*.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	28 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_RESET_INST_0	MC_RESET	Instância de acesso à estrutura do bloco

Funcionamento

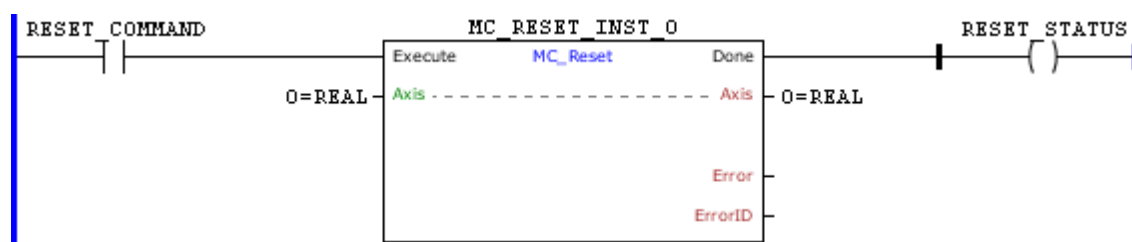
Este bloco, quando detecta uma borda de subida em Execute, realiza uma limpeza no estado do *drive*, alterando-o de Errorstop para Disabled.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

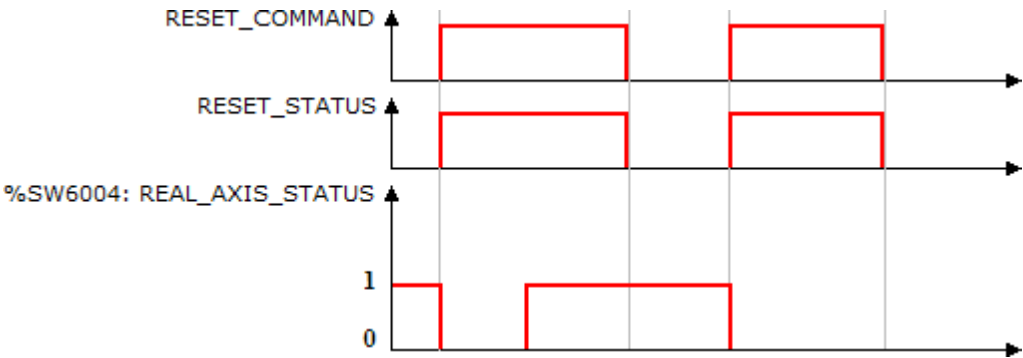
Código	Descrição
71	P202 diferente de 4.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_RESET_INST_0	0	MC_Reset		
RESET_COMMAND	0	BOOL	0	
RESET_STATUS	0	BOOL	0	

GLOBAL LOCAL				

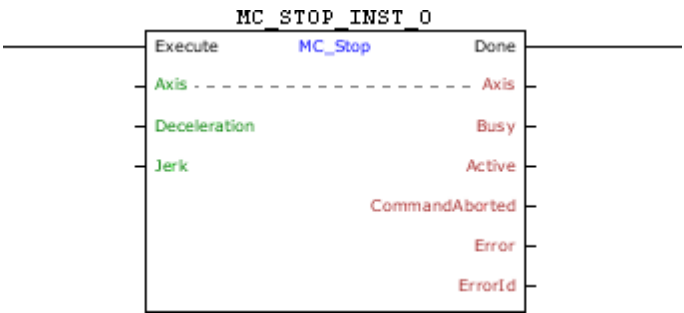


Ocorrendo alguma falha no *drive*, o estado do eixo mudará para Errorstop (%SW6004 = 1). Quando ocorrer uma transição de subida de RESET_COMMAND, o bloco MC_Reset será executado e o estado do eixo mudará para Disabled (%SW6004 = 0). A saída Done permanecerá setada enquanto a entrada Execute estiver em nível TRUE.

10.2.7.1.23.3 MC_Stop

Bloco responsável pela execução de uma parada controlada.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	52 Bytes
Tamanho de Memória de Dados	12 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Deceleration	REAL	Desaceleração da parada
	Jerk	REAL	Jerk
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STOP_INST_0	MC_STOP	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para uma parada controlada do eixo. Enquanto Execute possuir nível TRUE, nenhum outro bloco MC é executado.

Ao executar o bloco MC_Stop, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

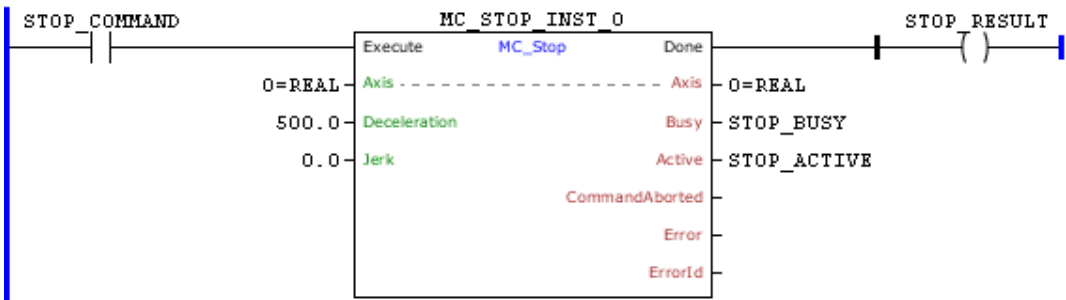
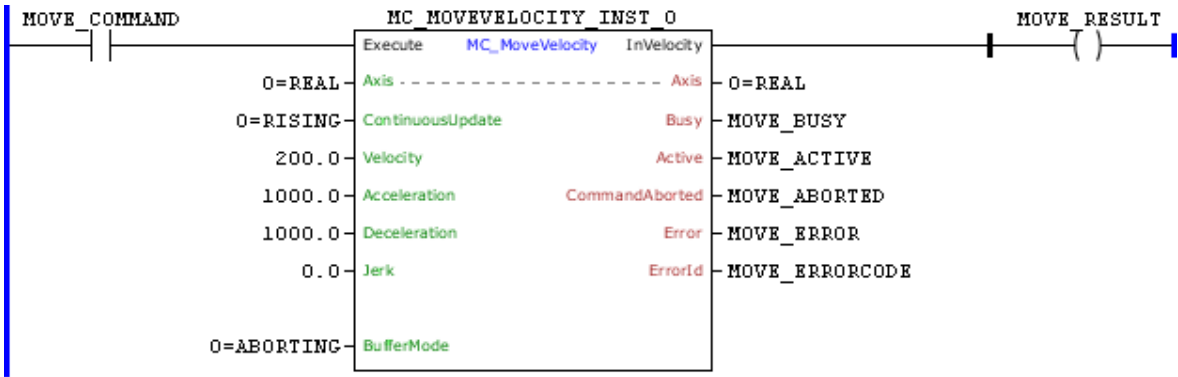
Na execução do bloco o estado do eixo mudará para Stopping. Ao finalizar a parada e o bloco não estiver mais ativo, o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

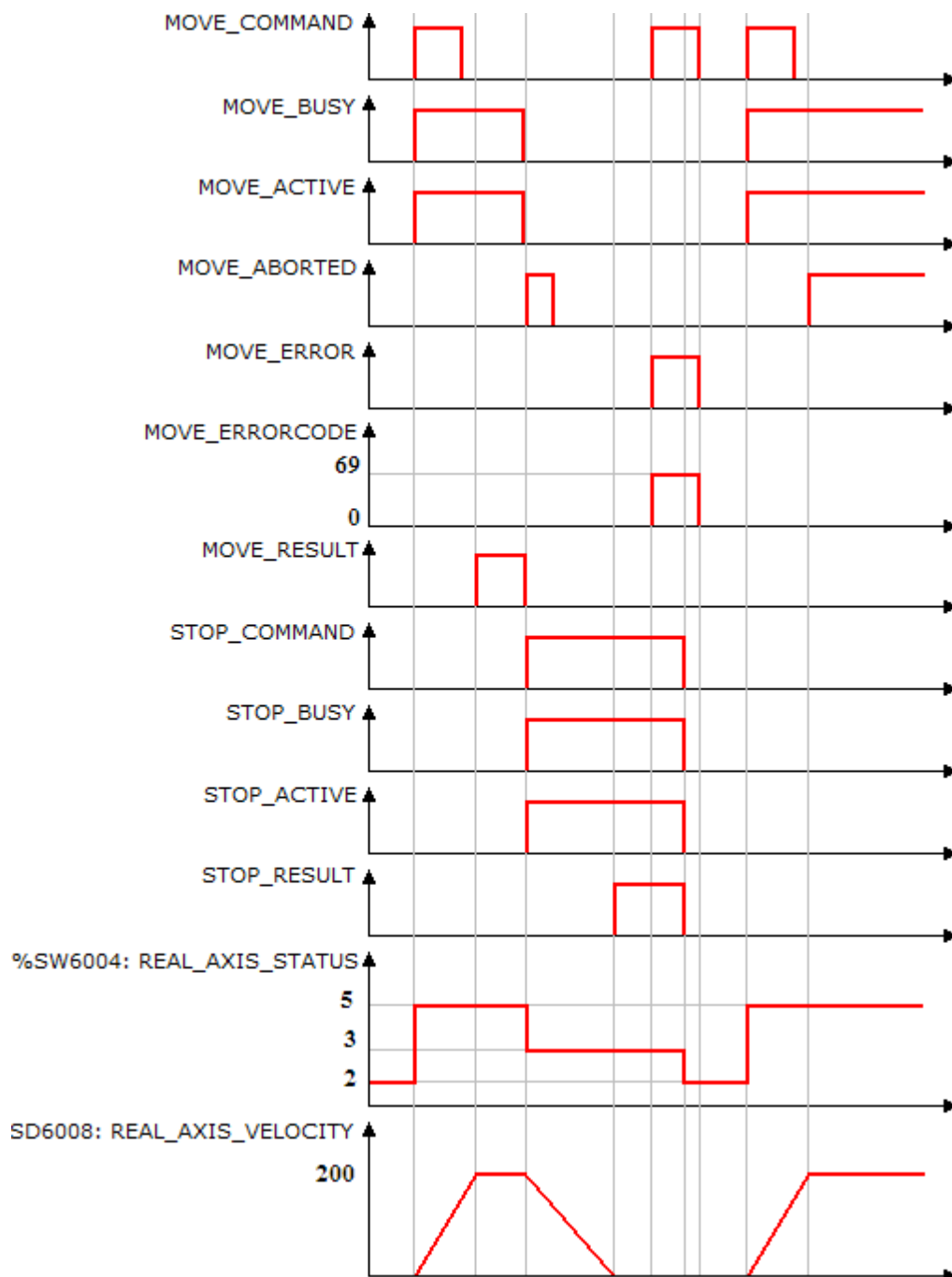
Código	Descrição
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_MOVEVELOCITY_INST_0	0	MC_MoveVelocity		
MC_STOP_INST_0	0	MC_Stop		
MOVE_COMMAND	0	BOOL	0	
MOVE_RESULT	0	BOOL	0	
MOVE_BUSY	0	BOOL	0	
MOVE_ACTIVE	0	BOOL	0	
MOVE_ABORTED	0	BOOL	0	
MOVE_ERROR	0	BOOL	0	
MOVE_ERRORCODE	0	WORD	0	
STOP_COMMAND	0	BOOL	0	
STOP_RESULT	0	BOOL	0	
STOP_BUSY	0	BOOL	0	
STOP_ACTIVE	0	BOOL	0	

GLOBAL LOCAL



Na transição de subida de MOVE_COMMAND, o bloco MC_MoveVelocity é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o movimento para chegar à velocidade de 200 RPM. O estado do eixo (%SW6004) muda de Standstill (2) para Continuous Motion (5). No instante em que a velocidade atinge 200 RPM, MOVE_RESULT é setado.

Tendo-se a transição de subida de STOP_COMMAND, o bloco MC_Stop é instantaneamente executado, com isso os sinais Busy e Active deste bloco são setados e se inicia a parada. Ao mesmo tempo os sinais Busy, Active e InVelocity do bloco MC_MoveVelocity são resetados e o sinal CommandAborted é setado por 1 ciclo de scan. O estado do eixo (%SW6004) muda de ContinuousMotion (5) para Stopping (3).

Ao finalizar a parada, a saída Done do bloco MC_Stop é setada e permanece até a entrada Execute seja resetada. O estado do eixo (%SW6004) permanece igual a Stopping (3) e nenhum outro bloco MC será executado.

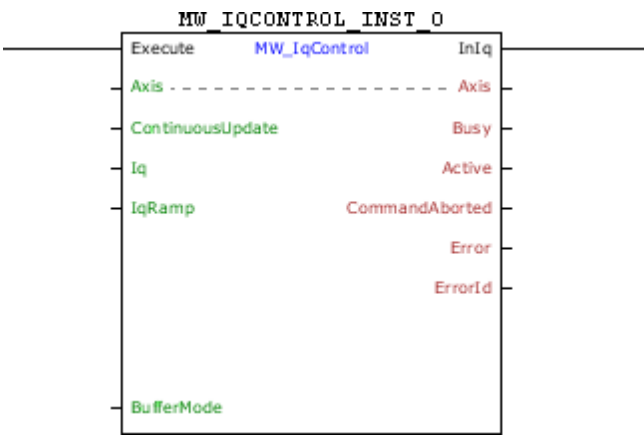
Tendo-se a transição de subida de MOVE_COMMAND, o bloco MC_MoveVelocity é iniciado, mas como o bloco MC_Stop está ativo, ocorrerá erro e o sinal Error será setado, inserindo em ErrorID o valor 69.

Quando a entrada Execute do bloco MC_Stop é resetada, os sinais Busy, Active e Done são resetados. O estado do eixo (%SW6004) muda de Stopping (3) para Standstill (2) e outros blocos MC poderão ser executados.

10.2.7.1.23.4 MW_IqControl

Bloco responsável pela realização do controle de Iq programado.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	78 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Iq	REAL	Valor de Iq
	IqRamp	REAL	Valor da rampa de Iq
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InIq	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MW_IQCONTROL_INST_0	MW_IQCONTROL	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando execução do controle de Iq de acordo com os parâmetros programados.

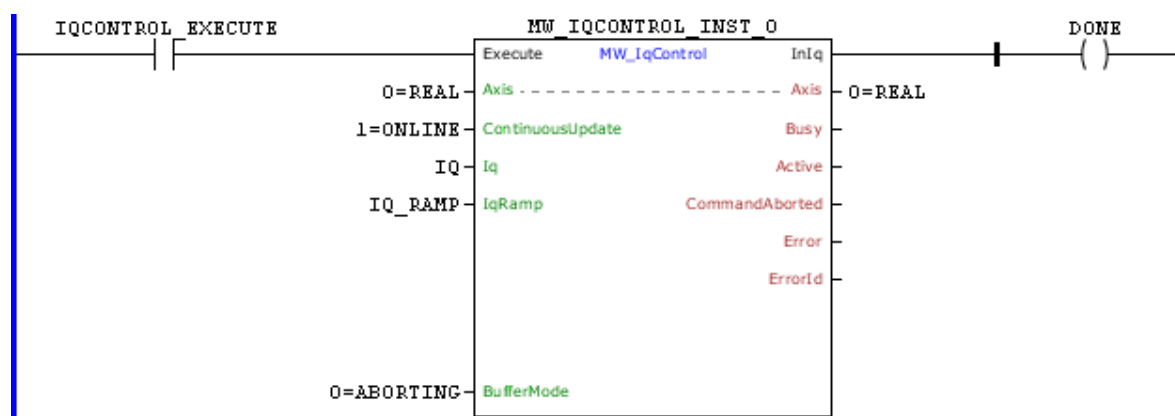
Na execução do bloco o estado do eixo mudará para Continuous Motion. Para finalizar o bloco, é necessária a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
80	Iq programado maior que o máximo permitido.
81	IqRamp programado menor que o mínimo permitido.
82	IqRamp programado maior que o máximo permitido.

Exemplo



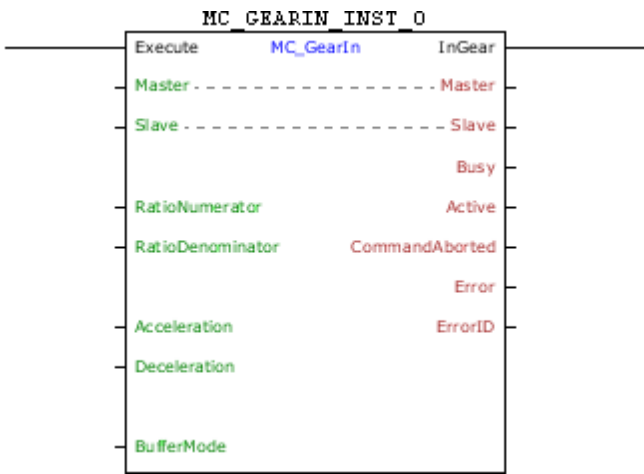
Na transição de subida de IQCONTROL_EXECUTE, o bloco MW_IqControl é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se a execução do controle de Iq de acordo com os parâmetros configurados. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.2.7.1.24 Motion Control Gear

10.2.7.1.24.1 MC_GearIn

Bloco responsável pela execução de sincronismo em velocidade entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	74 Bytes
Tamanho de Memória de Dados	28 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	RatioNumerator	INT	Numerador da relação de sincronismo
	RatioDenominator	WORD	Denominador da relação de sincronismo
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InGear	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEARIN_INST_0	MC_GEARIN	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para sincronismo em velocidade entre os eixos programados.

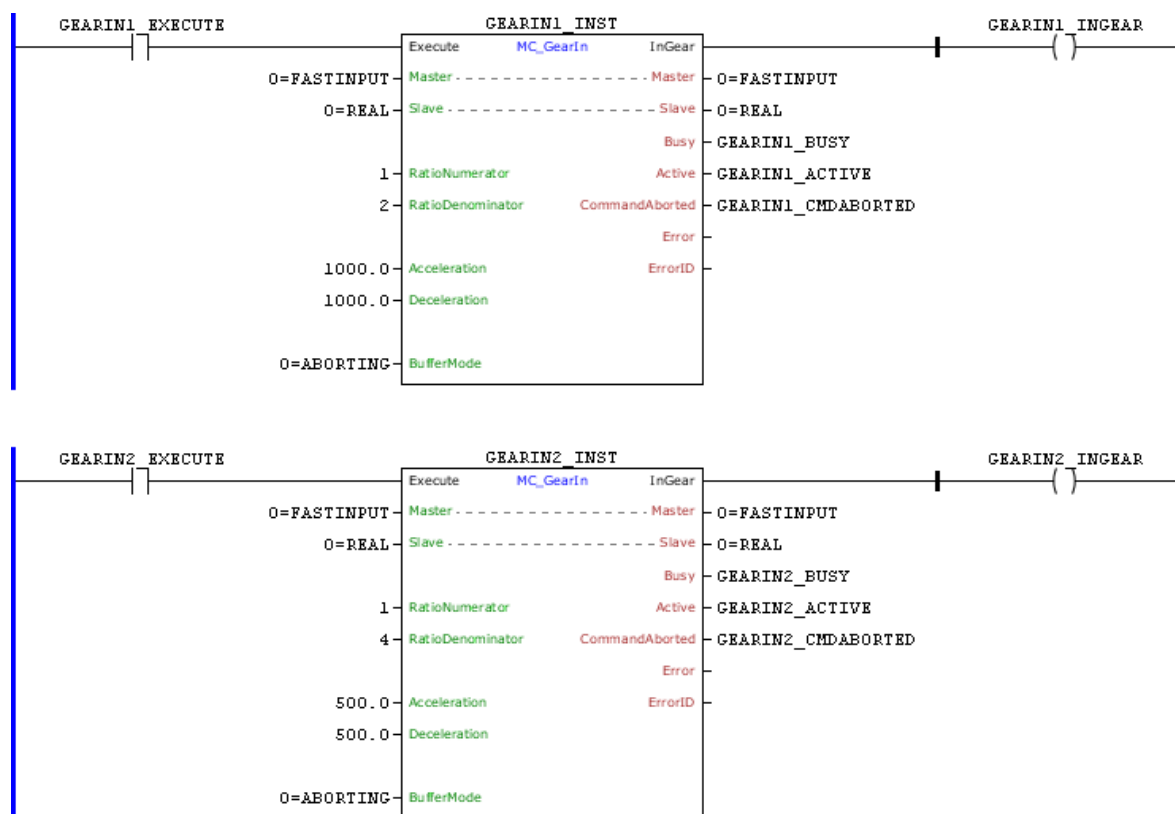
Para o eixo escravo atingir a velocidade do eixo mestre, será realizado um movimento com uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. A direção do movimento dependerá do sinal do RatioNumerator. Se RatioNumerator for maior que zero, o movimento será na mesma direção do eixo mestre e se RatioNumerator for menor que zero, o movimento será na direção oposta ao eixo mestre.

A saída InGear é ativada quando a sincronia é atingida. Para finalizar o bloco, é necessário a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

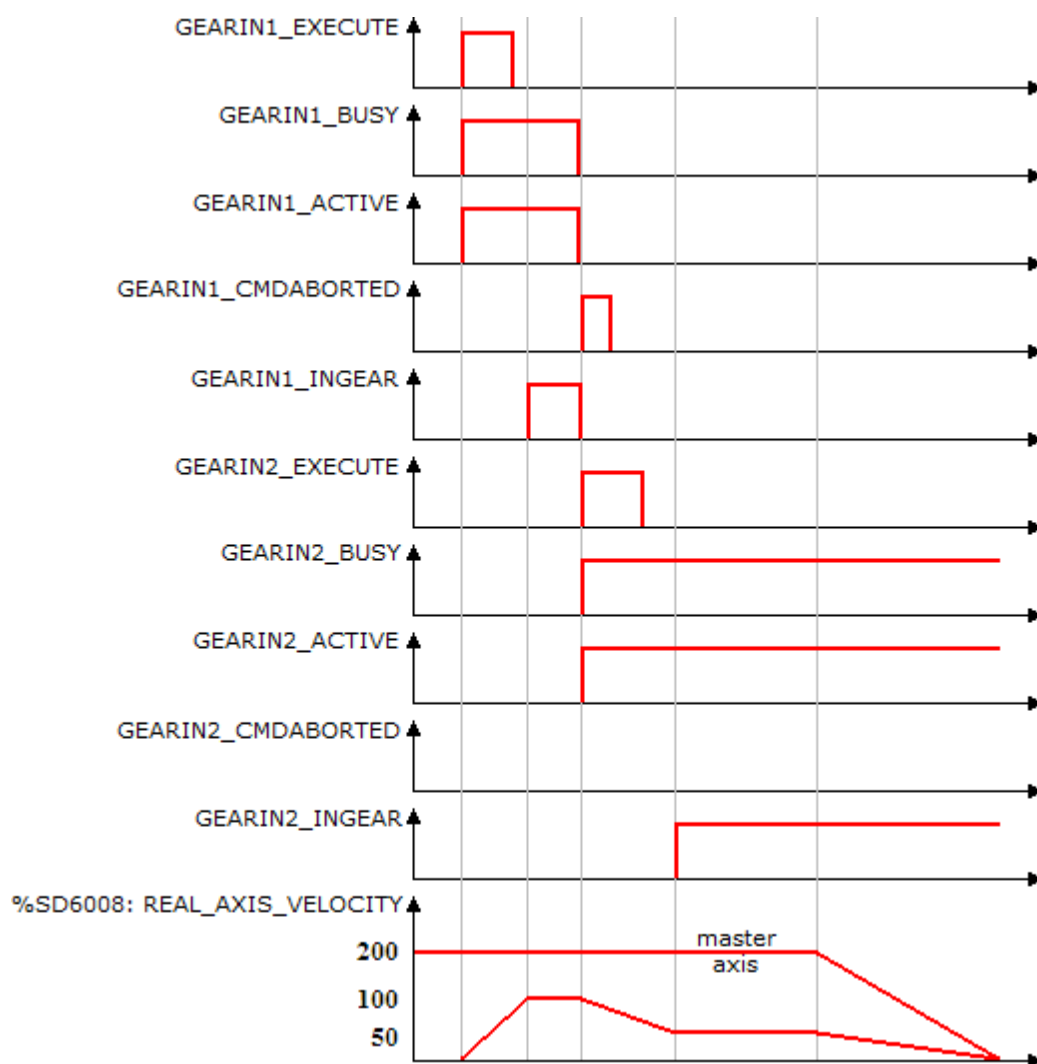
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
72	Relação de sincronismo inválida.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
GEARIN1_INST	0	MC_GearIn		
GEARIN1_EXECUTE	0	BOOL	0	
GEARIN1_INGEAR	0	BOOL	0	
GEARIN1_BUSY	0	BOOL	0	
GEARIN1_ACTIVE	0	BOOL	0	
GEARIN1_CMDABORTED	0	BOOL	0	
GEARIN2_INST	0	MC_GearIn		
GEARIN2_EXECUTE	0	BOOL	0	
GEARIN2_INGEAR	0	BOOL	0	
GEARIN2_BUSY	0	BOOL	0	
GEARIN2_ACTIVE	0	BOOL	0	
GEARIN2_CMDABORTED	0	BOOL	0	



Na transição de subida de GEARIN1_EXECUTE, o primeiro bloco MC_GearIn é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca do sincronismo com a

aceleração configurada. Como a relação configurada é 1:2 e o eixo mestre está a 200 RPM, o eixo escravo deverá atingir 100 RPM para estabelecer o sincronismo. No instante em que a velocidade atinge 100 RPM, a saída InGear é setada.

Tendo-se a transição de subida de GEARIN2_EXECUTE, o segundo bloco MC_GearIn é instantaneamente executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca do sincronismo com a desaceleração configurada. Como a relação configurada é 1:4 e o eixo mestre está a 200 RPM, o eixo escravo deverá atingir 50 RPM para estabelecer o sincronismo. Ao mesmo tempo os sinais Busy, Active e InGear do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

Ao atingir a velocidade de 50 RPM, a saída InGear do segundo bloco é setada e permanece até a execução de outro bloco.

10.2.7.1.24.2 MC_GearInPos

Bloco responsável pela execução de sincronismo em posição entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	104 Bytes
Tamanho de Memória de Dados	56 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	RatioNumerator	INT	Numerador da relação de sincronismo
	RatioDenominator	WORD	Denominador da relação de sincronismo
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InSync	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEARINPOS_INST_0	MC_GEARINPOS	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para sincronismo em posição entre os eixos programados.

Para o eixo escravo atingir a velocidade do eixo mestre, será realizado um movimento com uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. A direção do movimento dependerá do sinal do RatioNumerator. Se RatioNumerator for maior que zero, o movimento será na mesma direção do eixo mestre e se RatioNumerator for menor que zero, o movimento será na direção oposta ao eixo mestre.

Ao executar o bloco MC_GearInPos, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

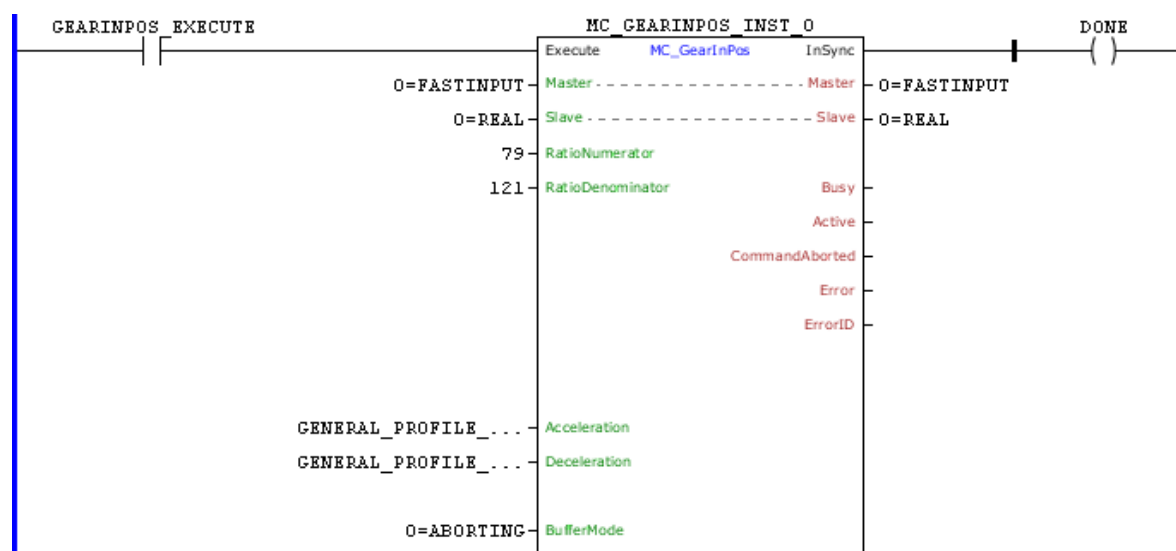
Na execução do bloco o estado do eixo mudará para Synchronized Motion.

A saída InSync é ativada quando a sincronia é atingida. Para finalizar o bloco, é necessário a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
72	Relação de sincronismo inválida.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.

Exemplo



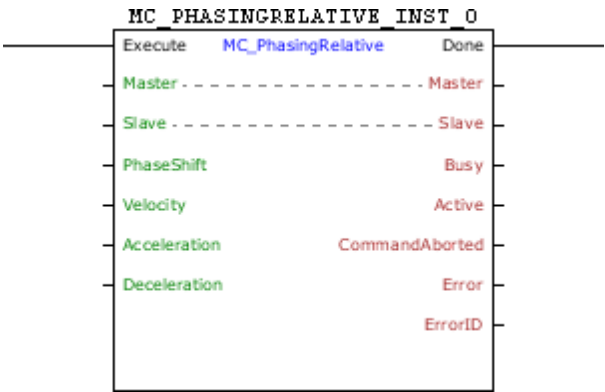
Na transição de subida de GEARINPOS_EXECUTE, o bloco MC_GearInPos é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca do sincronismo com a aceleração configurada. Como a relação configurada é 79:121, se o eixo mestre está a 200 RPM, o

eixo escravo deverá atingir 131 RPM para estabelecer o sincronismo.
No instante em que a velocidade atinge 131 RPM, a saída InSync é setada.

10.2.7.1.24.3 MC_PhasingRelative

Bloco responsável pela execução de um defasamento em posição entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	78 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	PhaseShift	REAL	Diferença de fase entre mestre e escravo
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEARIN_INST_0	MC_GEARIN	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica o sincronismo entre mestre e escravo e envia um comando para deslocar o eixo mestre de forma a deixá-lo defasado do escravo na magnitude de PhaseShift.

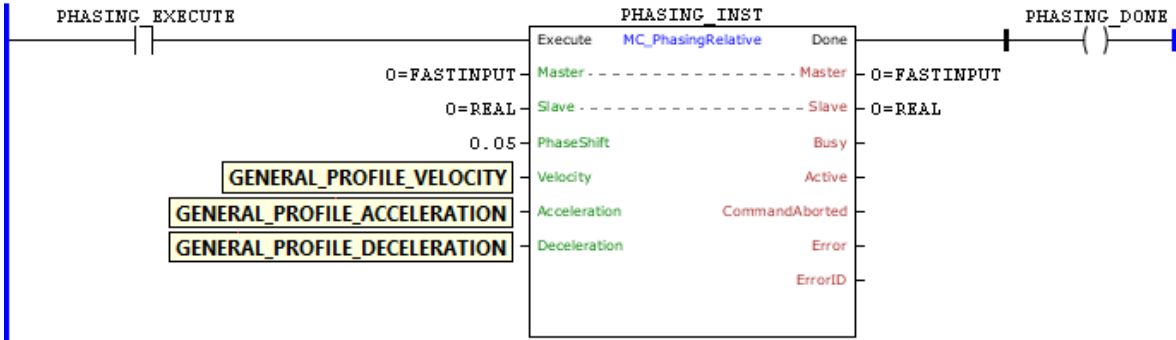
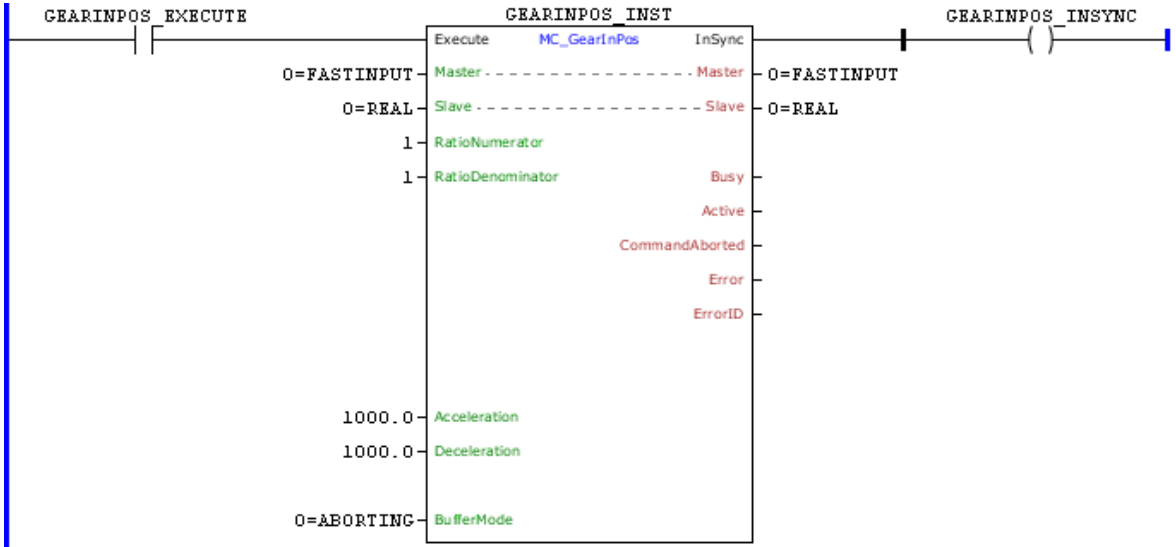
Ao executar o bloco MC_PhasingRelative, o *drive* não altera o modo de operação atual. Na execução do bloco o estado do eixo não mudará.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
73	Drive não está no estado "Synchronized Motion".
78	Bloco MC não executado - Falha interna.
79	Eixo Mestre não está em sincronismo.
96	Bloco MC_PhasingRelative em execução.

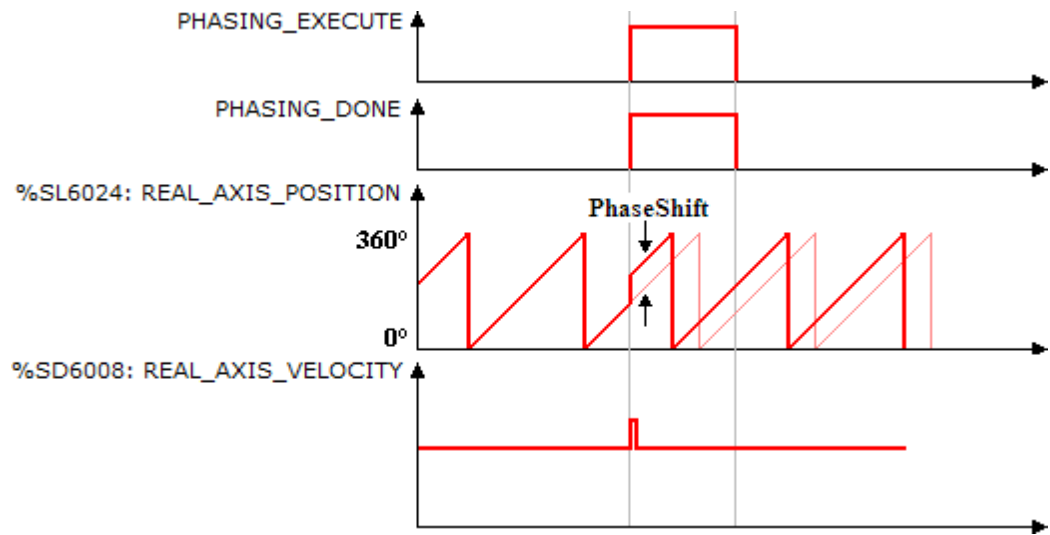
Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
GEARINPOS_INST	0	MC_GearInPos		
GEARINPOS_EXECUTE	0	BOOL	0	
GEARINPOS_INSYNC	0	BOOL	0	
PHASING_INST	0	MC_PhasingRelative		
PHASING_EXECUTE	0	BOOL	0	
PHASING_DONE	0	BOOL	0	

GLOBAL

LOCAL

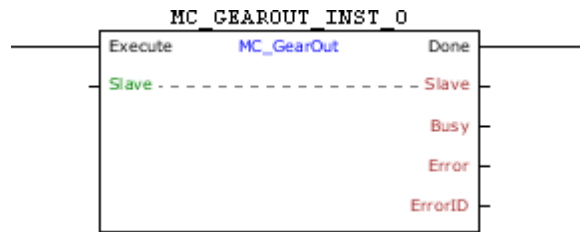


Tendo-se o sincronismo de posição do Eixo Real com o Contador Rápido através do bloco MC_GearInPos e ocorrendo uma transição de subida em PHASING_EXECUTE, o bloco MC_PhasingRelative é executado e um deslocamento de 0,05 volta é aplicado ao eixo mestre, resultando em um pulso na velocidade. A saída Done é setado enquanto a entrada Execute está setada.

10.2.7.1.24.4 MC_GearOut

Bloco responsável pela finalização de sincronismo entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	28 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEAROUT_INST_0	MC_GEAROUT	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, finaliza o sincronismo dos blocos MC_GearIn ou MC_GearInPos no eixo programado. O eixo manterá a velocidade do instante em que o bloco é executado.

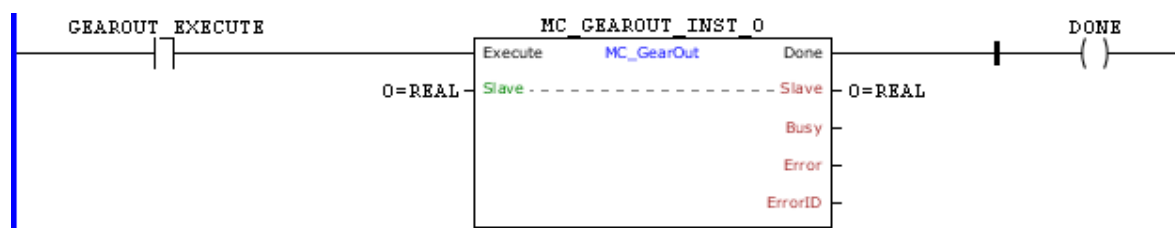
Ao executar o bloco MC_GearOut, o *drive* não opera em malha de posição. Na execução do bloco o estado do eixo mudará para ContinuousMotion.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
73	Drive não está no estado "Synchronized Motion".
78	Bloco MC não executado - Falha interna.

Exemplo



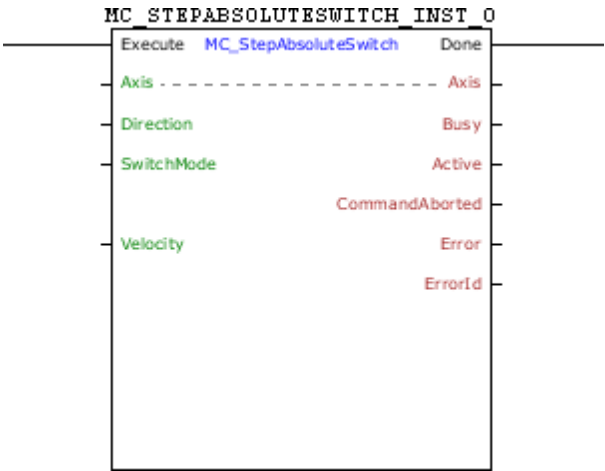
Na transição de subida de GEAROUT_EXECUTE, o bloco MC_GearOut é executado. Com isso, os sinais Busy e Active deste bloco são setados e finaliza-se o sincronismo iniciado por outros blocos de sincronismo entre mestre e escravo. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.2.7.1.25 Motion Control Homing

10.2.7.1.25.1 MC_StepAbsoluteSwitch

Bloco responsável pela busca da posição da AbsoluteSwitch.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	74 Bytes
Tamanho de Memória de Dados	40 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Direction	BYTE	Direção de busca (0 - Direção positiva) (1 - Direção negativa) (5 - MC_SwitchPositive: Direção positiva, se AbsoluteSwitch não acionada) (6 - MC_SwitchNegative: Direção negativa, se AbsoluteSwitch não acionada)
	SwitchMode	BYTE	Modo de busca (3 - MC_EdgeOn) (4 - MC_EdgeOff)
	Velocity	REAL	Velocidade
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STEPABSOLUTESWITCH_INST_0	MC_STEPABSOLUTESWITCH	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para busca da posição da AbsoluteSwitch.

A AbsoluteSwitch somente poderá ser ligada nas entradas digitais 1, 2 ou 3, sendo que a função programada da entrada digital deverá ser de acordo com o argumento SwitchMode. Se SwitchMode for configurado como MC_EdgeOn (borda de subida), a função da entrada digital (P0300, P0301 ou P0302) deverá ser "armazena posição - borda de subida" (opção 8). Se SwitchMode for configurado como MC_EdgeOff (borda de descida), a função da entrada digital (P0300, P0301 ou P0302) deverá ser "armazena posição - borda de descida" (opção 9). Será considerado AbsoluteSwitch a primeira entrada digital configurada conforme SwitchMode a partir da entrada digital 1. Caso nenhuma entrada digital esteja configurada conforme SwitchMode, ocorrerá o erro 77 no bloco e ele não será executado.

Se ao buscar a posição da AbsoluteSwitch e atingir a posição de LimitSwitch (fim de curso), o movimento mudará de sentido até a posição da AbsoluteSwitch.

A busca será executada com a velocidade configurada no argumento Velocity e uma aceleração/desaceleração configurado no Perfil Padrão (General Profile).

Com a execução do bloco MC_StepAbsoluteSwitch, a posição de referência do usuário (P0051, P0052 e P0053) não é alterada. O *drive* passará a operar em malha de posição e permanece assim

mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

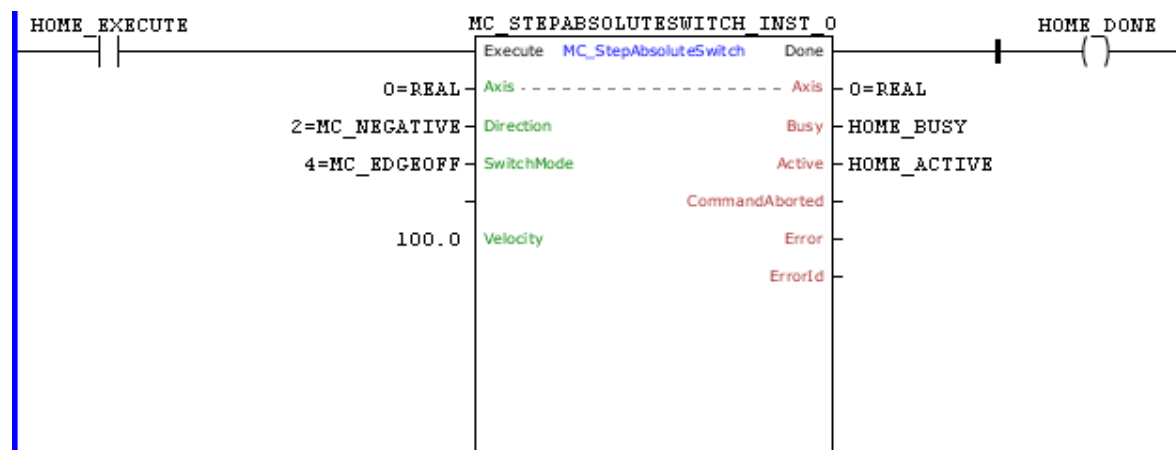
Na execução do bloco o estado do eixo mudará para Homing e permanecerá assim até a execução dos blocos MC_StepRefPulse, MC_StepDirect ou MC_FinishHoming.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

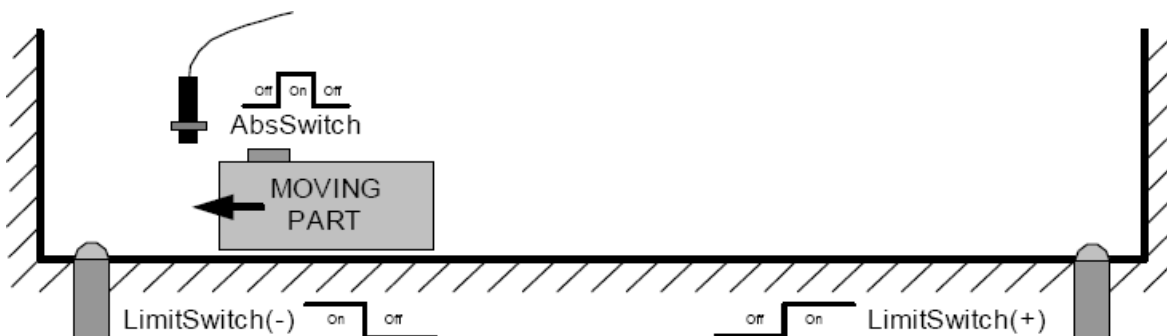
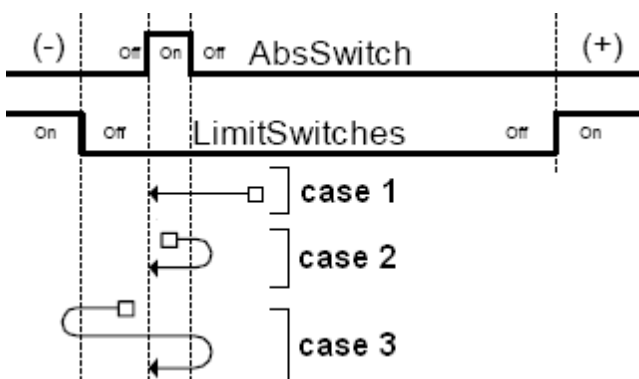
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".
77	Entradas digitais 1, 2 e 3 não configuradas conforme "SwitchMode".

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_STEPABSOLUTESWITCH_INST_0	0	MC_StepAbsoluteSwitch		
HOME_EXECUTE	0	BOOL	0	
HOME_BUSY	0	BOOL	0	
HOME_ACTIVE	0	BOOL	0	
HOME_DONE	0	BOOL	0	



Na transição de subida de HOME_EXECUTE, o bloco MC_StepAbsoluteSwitch é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se a busca da AbsoluteSwitch.

No caso 1, ao executar o bloco, a AbsoluteSwitch não está acionada. Como o argumento Direction está configurado como MC_SwitchNegative, o movimento será na direção negativa. Quando ocorrer uma borda de descida em AbsoluteSwitch (SwitchMode = MC_EdgeOff), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco a AbsoluteSwitch está acionada. Como o argumento Direction está configurado como MC_SwitchPositive, o movimento será na direção positiva e, ao sair da AbsoluteSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer uma borda de descida em AbsoluteSwitch (SwitchMode = MC_EdgeOff), o motor para e volta para a posição em que a borda ocorreu.

No caso 3, ao executar o bloco a AbsoluteSwitch não está acionada. Como o argumento Direction está configurado como MC_SwitchNegative, o movimento será na direção negativa. Mas ao encontrar a LimitSwitch (fim de curso) o motor para e muda o movimento para a direção positiva. Ao

sair da AbsoluteSwitch o motor para novamente e muda o movimento para a direção negativa. Quando ocorrer uma borda de descida em AbsoluteSwitch (SwitchMode = MC_EdgeOff), o motor para e volta para a posição em que a borda ocorreu.

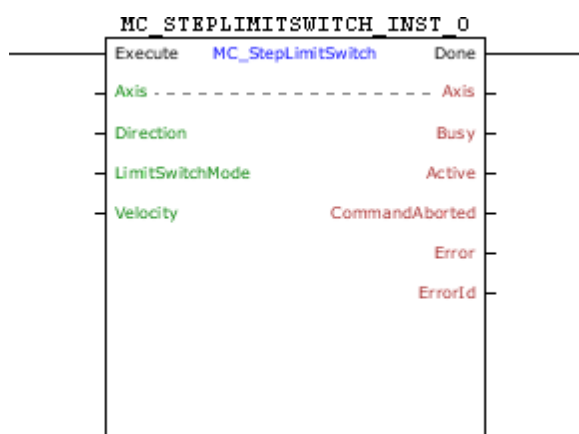
Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101, exceto ao encontrar a LimitSwitch (fim de curso), onde o motor para instantaneamente.

Ao voltar a posição de borda de descida da AbsoluteSwitch, a saída Done do bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

10.2.7.1.25.2 MC_StepLimitSwitch

Bloco responsável pela busca da posição da LimitSwitch.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	40 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Direction	BYTE	Direção de busca (0 - Direção positiva) (1 - Direção negativa) (5 - MC_SwitchPositive: Direção positiva, se AbsoluteSwitch não acionada) (6 - MC_SwitchNegative: Direção negativa, se AbsoluteSwitch não acionada)
	LimitSwitchMode	BYTE	Modo de busca (3 - MC_EdgeOn) (4 - MC_EdgeOff)
	Velocity	REAL	Velocidade
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STEPLIMITSWITCH_INST_0	MC_STEPLIMITSWITCH	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para busca da posição da LimitSwitch.

A LimitSwitch somente poderá ser ligada nas entradas digitais 1, 2 ou 3, sendo que a função programada da entrada digital deverá ser de acordo com o argumento "LimitSwitchMode" e o argumento "Direction". Será considerado LimitSwitch a primeira entrada digital configurada conforme a tabela, a partir da entrada digital 1. Caso nenhuma entrada digital esteja configurada conforme LimitSwitchMode e Direction, ocorrerá o erro 77 no bloco e ele não será executado.

A busca será executada com a velocidade configurada no argumento Velocity e uma aceleração/desaceleração configurado no Perfil Padrão (General Profile).

Com a execução do bloco MC_StepLimitSwitch, a posição de referência do usuário (P0051, P0052 e P0053) não é alterada. O *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

Na execução do bloco o estado do eixo mudará para Homing e permanecerá assim até a execução dos blocos MC_StepRefPulse, MC_StepDirect ou MC_FinishHoming.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba

FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".
77	Entradas digitais 1, 2 e 3 não configuradas conforme "Sw itchMode".

Exemplo

HOME_EXECUTE

MC STEPLIMITSWITCH INST 0

HOME_DONE

Execute MC_StepLimitSwitch Done

Axis ----- Axis

Direction Busy

LimitSwitchMode Active

Velocity CommandAborted

Error

ErrorId

O=REAL

2=MC_NEGATIVE

3=MC_EDGEON

100.0

O=REAL

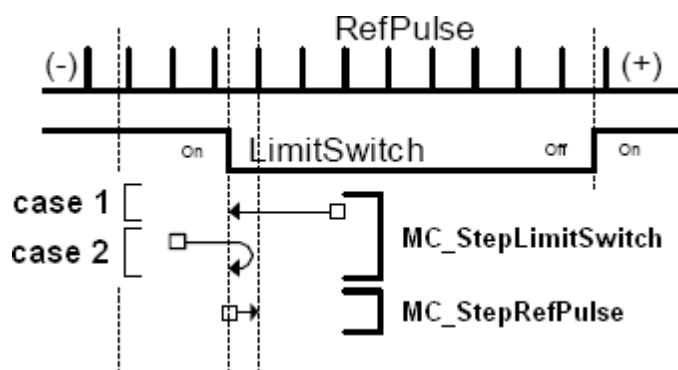
HOME_BUSY

HOME_ACTIVE

LOCAL LOCAL_RETAIN

Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_STEPLIMITSWITCH_INST_0	0	MC_StepLimitSwitch		
HOME_EXECUTE	0	BOOL	0	
HOME_BUSY	0	BOOL	0	
HOME_ACTIVE	0	BOOL	0	
HOME_DONE	0	BOOL	0	

GLOBAL LOCAL



Na transição de subida de HOME_EXECUTE, o bloco MC_StepLimitSwitch é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca da LimitSwitch.

No caso 1, ao executar o bloco, a LimitSwitch não está acionada. Como o argumento Direction está configurado como MC_Negative, o movimento será na direção negativa. Quando ocorrer uma borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco, a LimitSwitch está acionada. Mesmo com o argumento Direction configurado como MC_Negative, o movimento será na direção positiva e, ao sair da LimitSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer a borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

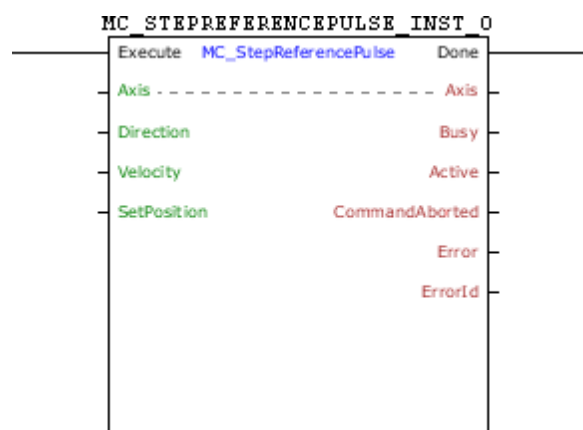
Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101, exceto ao encontrar a LimitSwitch (fim de curso), onde o motor para instantaneamente.

Ao voltar a posição de borda de subida da LimitSwitch, a saída Done do bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

10.2.7.1.25.3 MC_StepReferencePulse

Bloco responsável pela busca da posição do pulso nulo.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	40 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Direction	BYTE	Direção de busca (0 - Direção positiva) (1 - Direção negativa) (5 - MC_SwitchPositive: Direção positiva, se AbsoluteSwitch não acionada) (6 - MC_SwitchNegative: Direção negativa, se AbsoluteSwitch não acionada)
	Velocity	REAL	Velocidade
	SetPosition	LREAL	Nova posição de referência do usuário
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STEPREFERENCEPULSE_INST_0	MC_STEPREFERENCEPULSE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para busca da posição do pulso nulo.

A busca será executada com a velocidade configurada no argumento Velocity e uma aceleração/desaceleração configurado no Perfil Padrão (General Profile).

Com a execução do bloco MC_StepReferencePulse, a posição de referência do usuário (P0051, P0052 e P0053) é alterada para o valor do argumento SetPosition. O *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

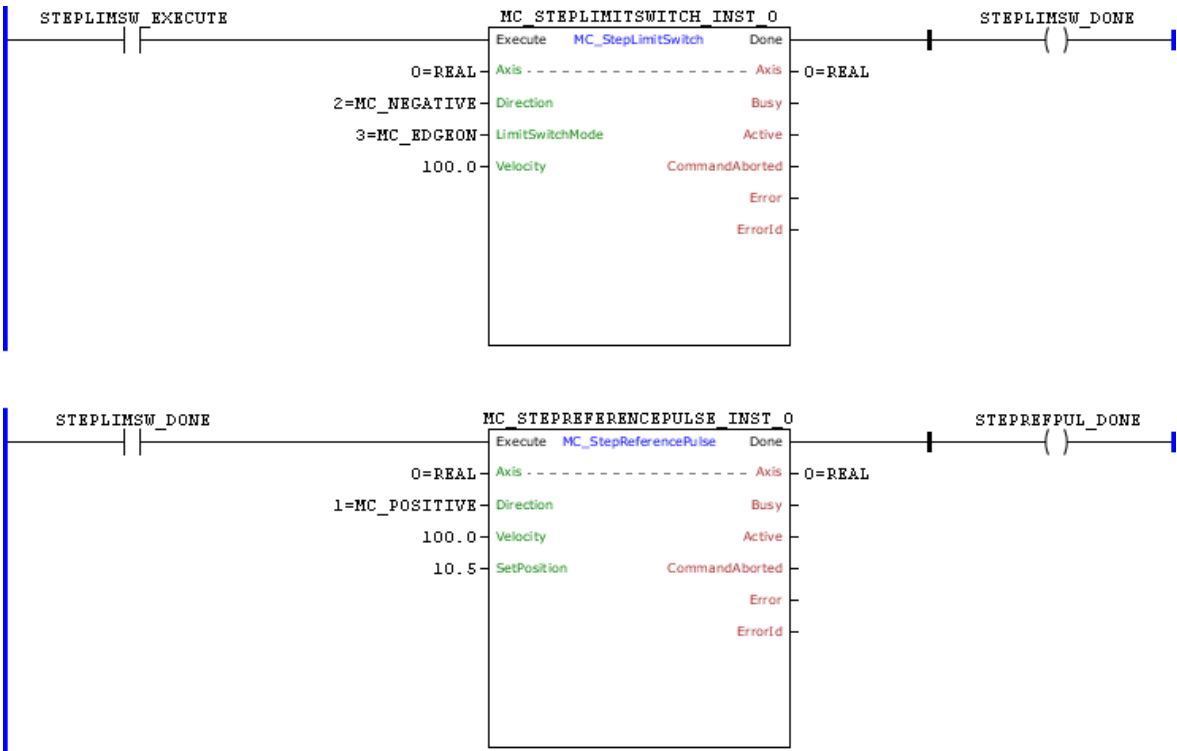
Na execução do bloco o estado do eixo mudará para Homing. Ao concluir a busca o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

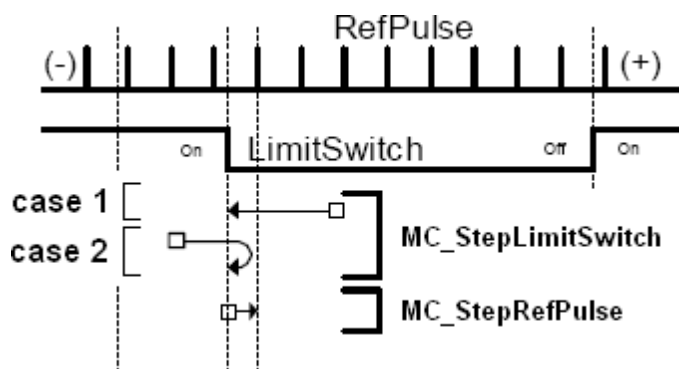
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_STEPLIMITSWITCH_INST_0	0	MC_StepLimitSwitch		
STEPLIMSW_EXECUTE	0	BOOL	0	
STEPLIMSW_DONE	0	BOOL	0	
MC_STEPPREFERENCEPULSE_INST_0	0	MC_StepReferencePulse		
STEPPREFPUL_DONE	0	BOOL	0	



Na transição de subida de STEPLIMSW_EXECUTE, o bloco MC_StepLimitSwitch é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca da LimitSwitch.

No caso 1, ao executar o bloco, a LimitSwitch não está acionada. Como o argumento Direction está configurado como MC_Negative, o movimento será na direção negativa. Quando ocorrer uma borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco, a LimitSwitch está acionada. Mesmo com o argumento Direction configurado como MC_Negative, o movimento será na direção positiva e, ao sair da LimitSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer a borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101, exceto ao encontrar a LimitSwitch (fim de curso), onde o motor para instantaneamente.

Ao voltar a posição de borda de subida da LimitSwitch, a saída Done do bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

Na transição de subida de STEPLIMSW_DONE, o bloco MC_StepRefPulse é executado e iniciasse a busca do pulso nulo.

O movimento será na direção positiva e ao encontrar o pulso nulo, o motor para e volta para a posição do pulso nulo.

Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101.

Ao voltar a posição do pulso nulo, a saída Done do bloco é setada e permanece TRUE enquanto a

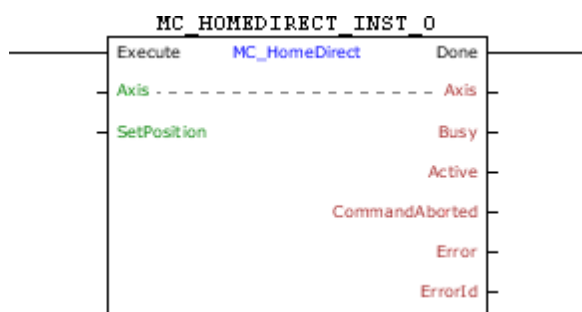
entrada Execute está setada. A posição de referência do usuário (P0051, P0052 e P0053) é alterada para 10,5 voltas (P0051 = 8192, P0052 = 10 e P0053 = 0).

Quando STEPLIMSW_EXECUTE é resetado, STEPLIMSW_DONE e STEPREFPUL_DONE também são resetados.

10.2.7.1.25.4 MC_HomeDirect

Bloco responsável pela mudança da posição de referência do usuário.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	46 Bytes
Tamanho de Memória de Dados	16 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	SetPosition	LREAL	Nova posição de referência do usuário
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_HOMEDIRECT_INST_0	MC_HOMEDIRECT	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, faz com que a posição de referência do usuário (P0051, P0052 e P0053) seja alterada para o valor do argumento SetPosition.

Na execução do bloco, se o estado do eixo é Homing, o estado do eixo mudará para Standstill. Caso

contrário, permanecerá no estado atual.

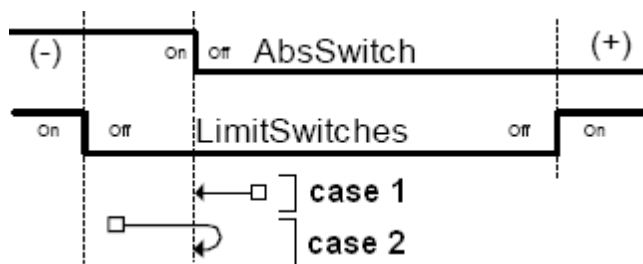
Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".

Exemplo





Na transição de subida de STEPABSSW_EXECUTE, o bloco MC_StepAbsoluteSwitch é executado e inicia-se a busca da AbsoluteSwitch. O estado do eixo é alterado para Homing.

No caso 1, ao executar o bloco, a AbsoluteSwitch não está acionada. Como o argumento Direction está configurado como MC_SwitchNegative, o movimento será na direção negativa. Quando ocorrer uma borda de subida em AbsoluteSwitch (SwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco a AbsoluteSwitch está acionada. Como o argumento Direction está configurado como MC_SwitchPositive, o movimento será na direção positiva e, ao sair da AbsoluteSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer uma borda de subida em AbsoluteSwitch (SwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101.

Ao voltar a posição de borda de subida da AbsoluteSwitch, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

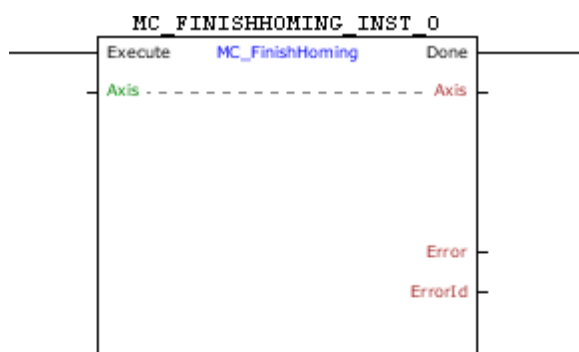
Na transição de subida de STEPABSSW_DONE, o bloco MC_StepDirect é executado e a posição de referência do usuário (P0051, P0052 e P0053) é alterada para 3,5 voltas (P0051 = 8192, P0052 = 3 e P0053 = 0). O estado do eixo é alterado para Standstill.

Quando STEPABSSW_EXECUTE é resetado, STEPABSSW_DONE e STEPDIR_DONE também são resetados.

10.2.7.1.25.5 MC_FinishHoming

Bloco responsável pela mudança do estado do eixo de Homing para Standstill.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	70 Bytes
Tamanho de Memória de Dados	8 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_HOMEDIRECT_INST_0	MC_HOMEDIRECT	Instância de acesso à estrutura do bloco

Funcionamento

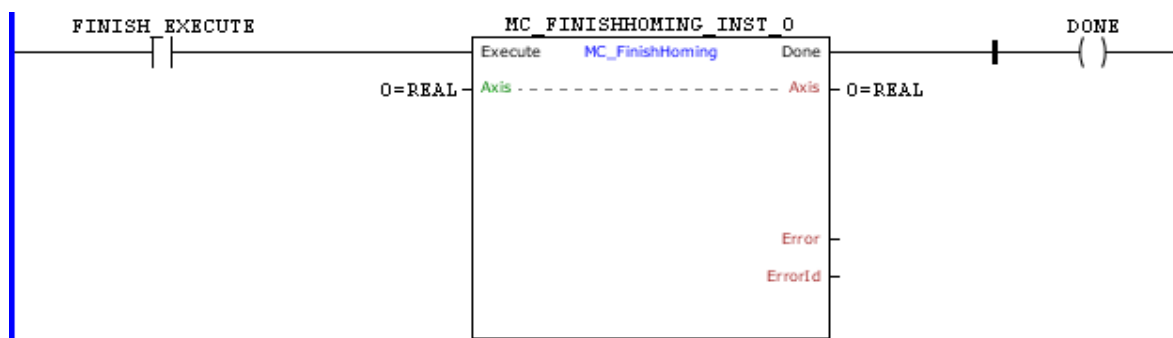
Este bloco, quando detecta uma borda de subida em Execute, se o estado do eixo é Homing, o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
71	P202 diferente de 4.
75	Estado do Drive diferente de "Homing".

Exemplo



Na transição de subida de FINISH_EXECUTE, o bloco MC_FinishHoming é executado. Com isso, os sinais Busy e Active deste bloco são setados e muda-se o estado do eixo selecionado para

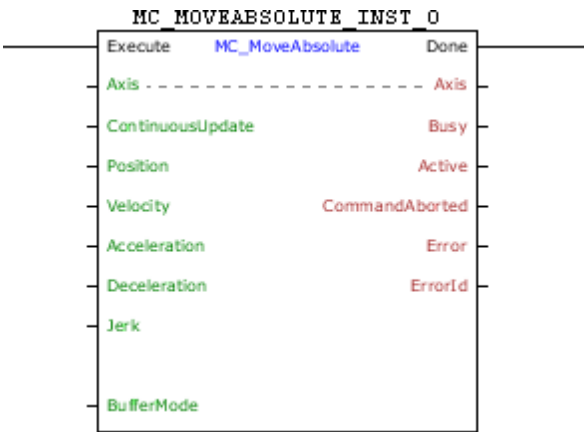
Standstill. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.2.7.1.26 Motion Control Move

10.2.7.1.26.1 MC_MoveAbsolute

Bloco responsável por realizar um posicionamento para a posição absoluta programada.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Position	LREAL	Posição
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	Jerk	REAL	Jerk
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_MOVEABSOLUTE_INST_0	MC_MOVEABSOLUTE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, executará um posicionamento para a posição absoluta configurada no argumento Position, com uma velocidade máxima configurada no argumento Velocity e uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. Dependendo da distância do posicionamento e dos valores de aceleração e desaceleração, a velocidade do motor não atingirá a velocidade máxima configurada.



NOTA!

Caso o valor do argumento Jerk for diferente de zero:

- 1) o valor da desaceleração será o mesmo do valor configurado na aceleração;
- 2) o argumento ContinuousUpdate On-line não terá efeito, sendo considerado os valores dos argumentos no instante da transição positiva de Execute;
- 3) não é permitido executar o posicionamento com outro bloco ativo, ocorrendo ErrorID 95.

Ao executar o bloco MC_MoveAbsolute, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

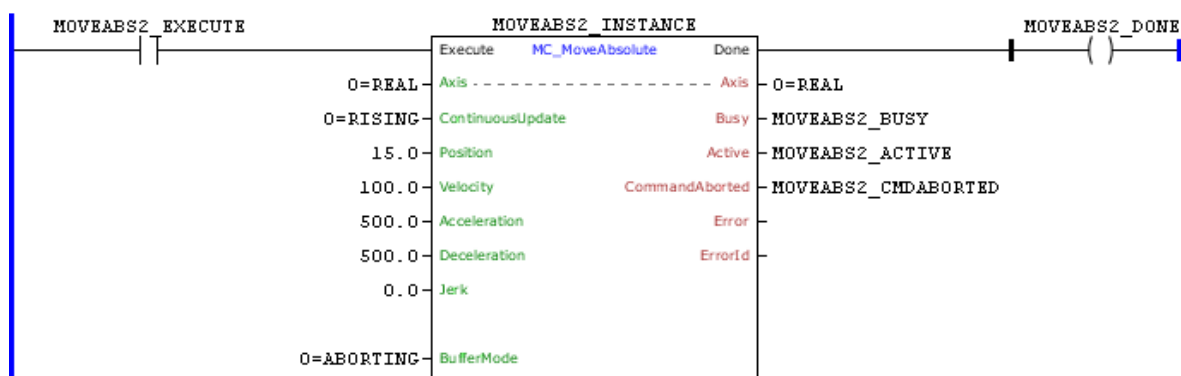
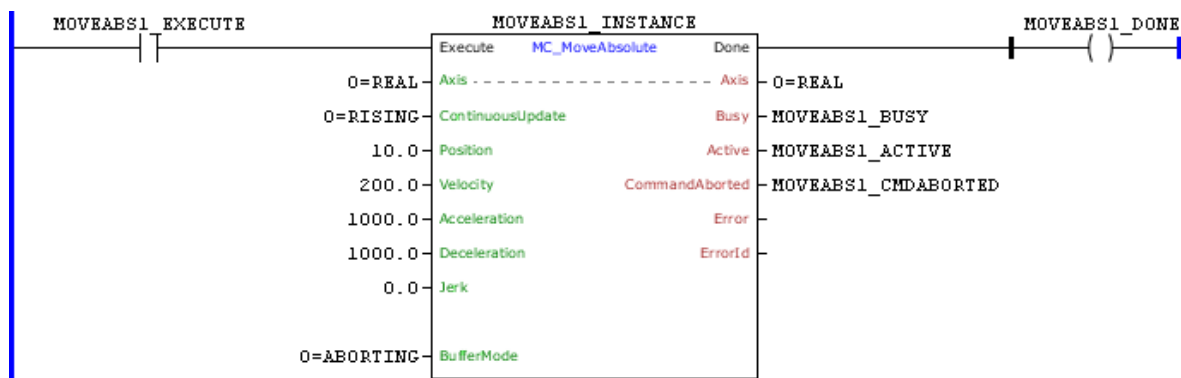
Na execução do posicionamento o estado do eixo mudará para Discrete Motion. Ao concluir o posicionamento o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

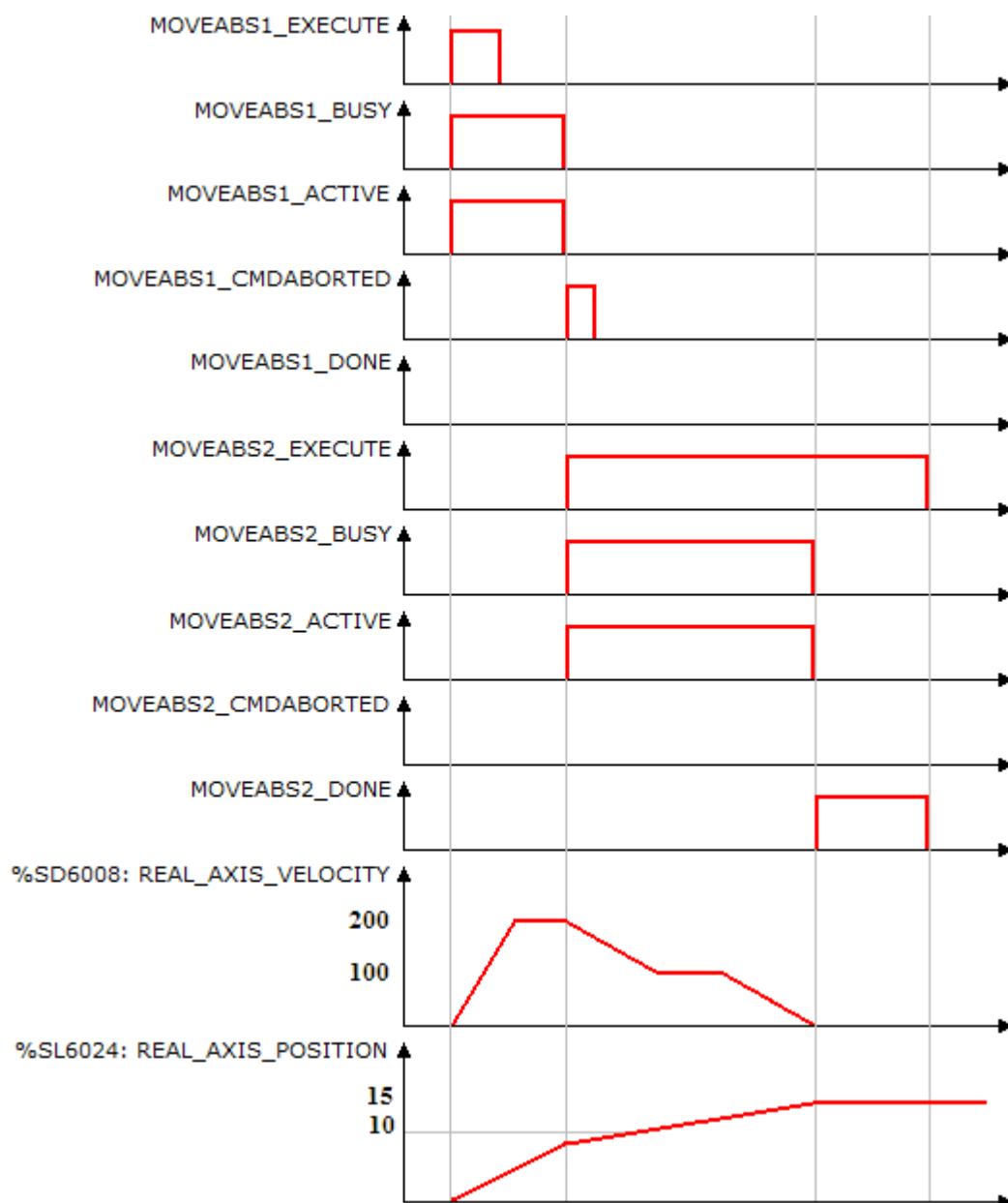
Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.
95	Não é permitido executar posicionamento com Jerk quando outro bloco ativo.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MOVEABS1_INSTANCE	0	MC_MoveAbsolute		
MOVEABS2_INSTANCE	0	MC_MoveAbsolute		
MOVEABS1_EXECUTE	0	BOOL	0	
MOVEABS1_DONE	0	BOOL	0	
MOVEABS1_BUSY	0	BOOL	0	
MOVEABS1_ACTIVE	0	BOOL	0	
MOVEABS1_CMDABORTED	0	BOOL	0	
MOVEABS2_EXECUTE	0	BOOL	0	
MOVEABS2_DONE	0	BOOL	0	
MOVEABS2_BUSY	0	BOOL	0	
MOVEABS2_ACTIVE	0	BOOL	0	
MOVEABS2_CMDABORTED	0	BOOL	0	

GLOBAL LOCAL



Na transição de subida de MOVEABS1_EXECUTE, o primeiro bloco MC_MoveAbsolute é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento para a posição absoluta "10 voltas".

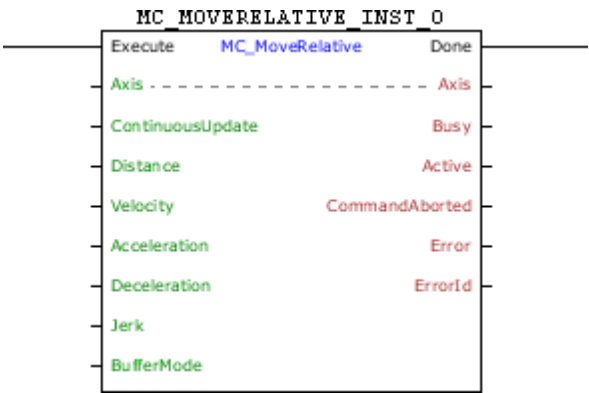
Tendo-se a transição de subida de MOVEABS2_EXECUTE, o segundo bloco MC_MoveAbsolute é instantaneamente executado (BufferMode - Aborting). Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento para a posição 15 voltas. Ao mesmo tempo os sinais Busy e Active do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

Ao atingir a posição 15 voltas, a saída Done do segundo bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

10.2.7.1.26.2 MC_MoveRelative

Bloco responsável por realizar um posicionamento para a posição relativa programada.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Distance	LREAL	Distância de deslocamento
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	Jerk	REAL	Jerk
VAR_OUTPUT	BufferMode	REAL	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_MOVERELATIVE_INST_0	MC_MOVERELATIVE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, executará um posicionamento de acordo com a distância relativa à posição atual, configurada no argumento Distance, com uma velocidade máxima configurada no argumento Velocity e uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. Dependendo da distância do posicionamento e dos valores de aceleração e desaceleração, a velocidade do motor não atingirá a velocidade máxima configurada.



NOTA!

Caso o valor do argumento Jerk for diferente de zero:

- 1) o valor da desaceleração será o mesmo do valor configurado na aceleração;
- 2) o argumento ContinuousUpdate On-line não terá efeito, sendo considerado os valores dos argumentos no instante da transição positiva de Execute;
- 3) não é permitido executar o posicionamento com outro bloco ativo, ocorrendo ErrorID 95.

Ao executar o bloco MC_MoveRelative, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

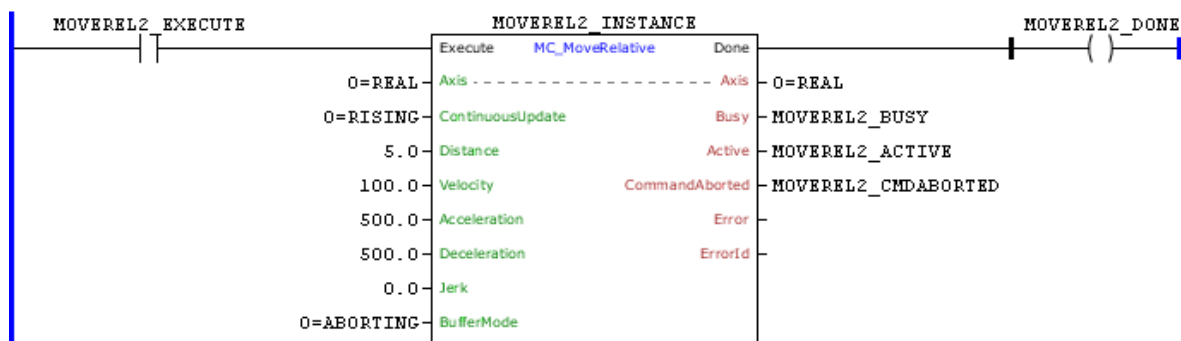
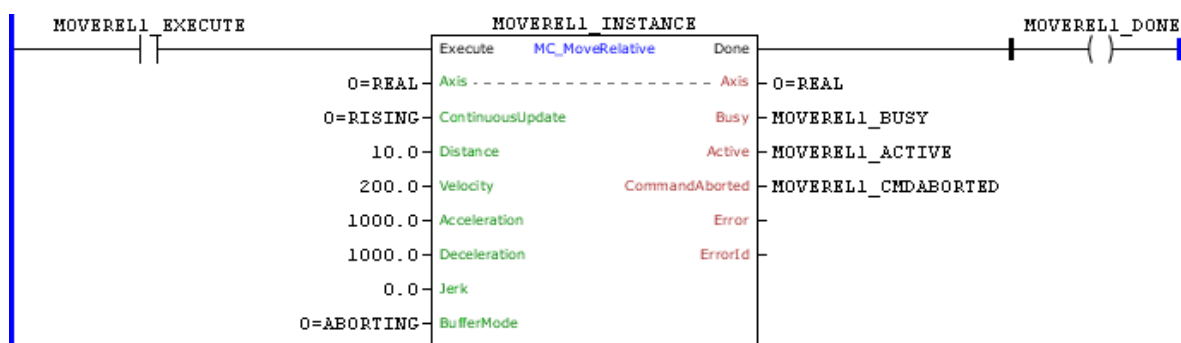
Na execução do posicionamento o estado do eixo mudará para Discrete Motion. Ao concluir o posicionamento o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

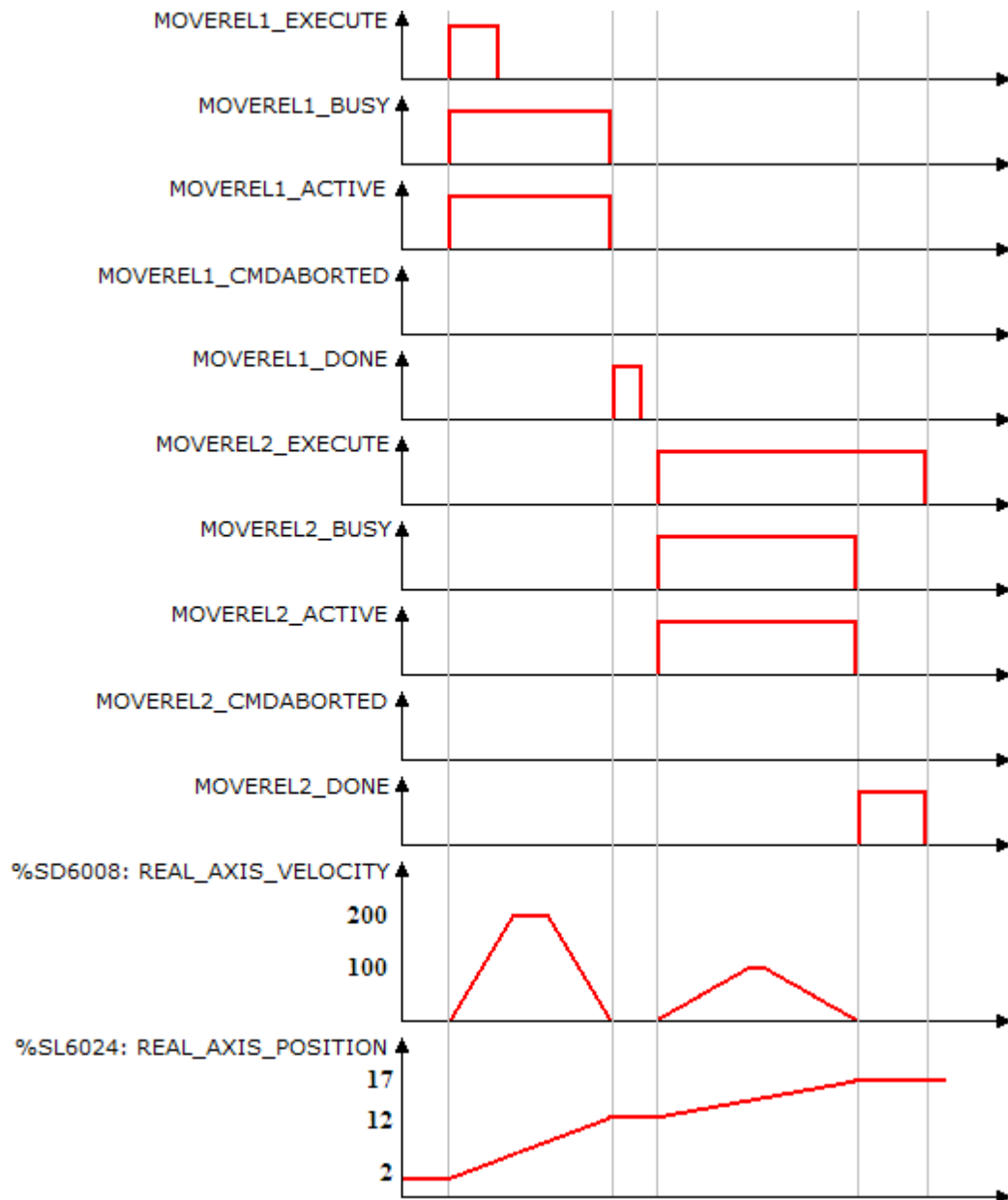
Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.
95	Não é permitido executar posicionamento com Jerk quando outro bloco ativo.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
 MOVEREL1_INSTANCE	0	MC_MoveRelative		
 MOVEREL2_INSTANCE	0	MC_MoveRelative		
 MOVEREL1_EXECUTE	0	BOOL	0	
 MOVEREL1_DONE	0	BOOL	0	
 MOVEREL1_BUSY	0	BOOL	0	
 MOVEREL1_ACTIVE	0	BOOL	0	
 MOVEREL1_CMDABORTED	0	BOOL	0	
 MOVEREL2_EXECUTE	0	BOOL	0	
 MOVEREL2_DONE	0	BOOL	0	
 MOVEREL2_BUSY	0	BOOL	0	
 MOVEREL2_ACTIVE	0	BOOL	0	
 MOVEREL2_CMDABORTED	0	BOOL	0	

GLOBAL LOCAL

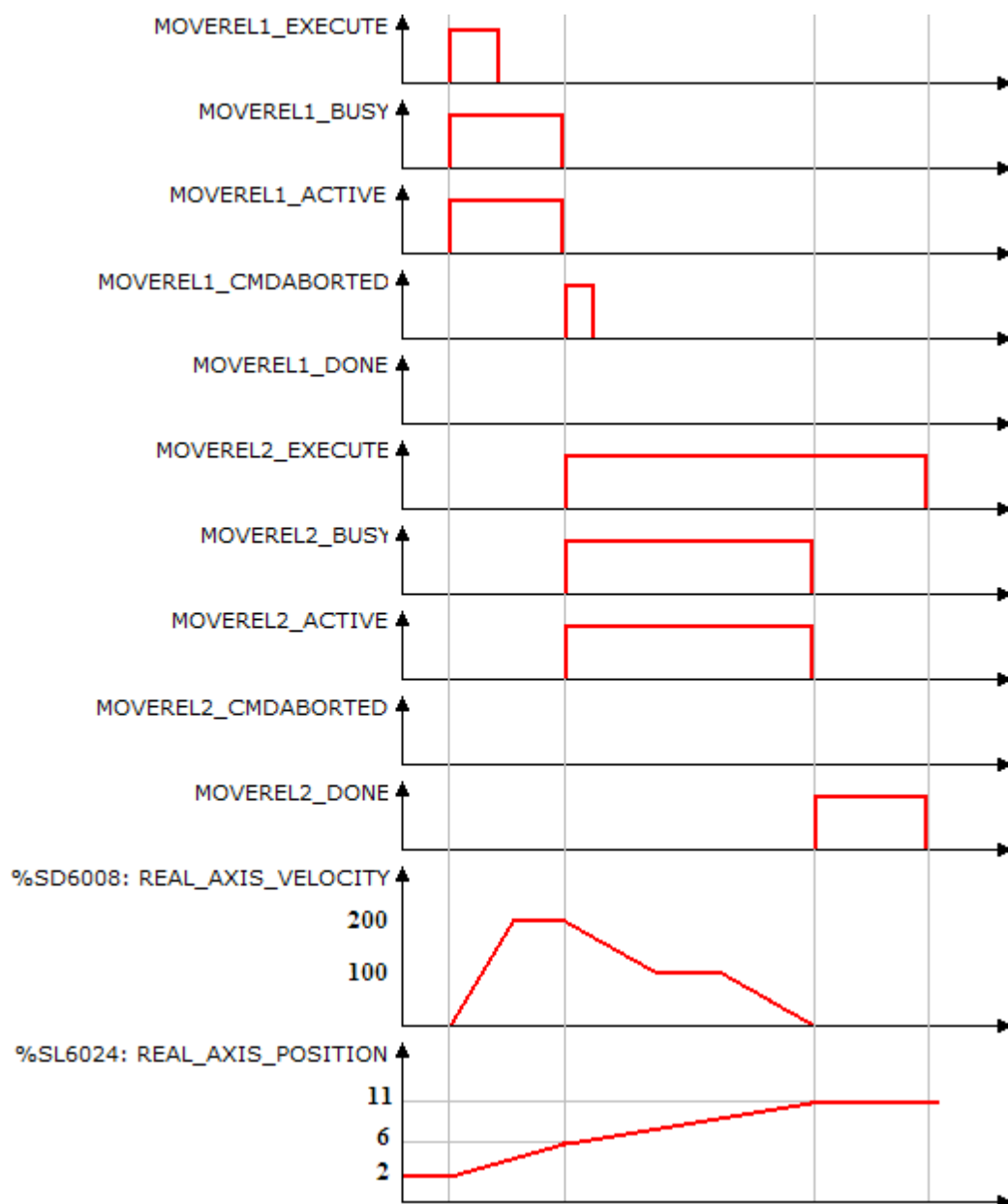


Na transição de subida de MOVEREL1_EXECUTE, o primeiro bloco MC_MoveRelative é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 10 voltas positivas a partir da posição atual.

Ao terminar o posicionamento, o primeiro bloco é concluído. Com isso, os sinais Busy e Active deste bloco são resetados e a saída Done é setada por 1 ciclo de scan.

Com uma transição de subida de MOVEREL2_EXECUTE, o segundo bloco MC_MoveRelative é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 5 voltas positivas a partir da posição atual.

Ao terminar o posicionamento, a saída Done do segundo bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.



Na transição de subida de MOVEREL1_EXECUTE, o primeiro bloco MC_MoveRelative é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 10 voltas positivas a partir da posição atual.

Tendo-se a transição de subida de MOVEREL2_EXECUTE, o segundo bloco MC_MoveRelative é instantaneamente executado (BufferMode - Aborting). Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 5 voltas positivas a partir da posição atual. Ao mesmo tempo, os sinais Busy e Active do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

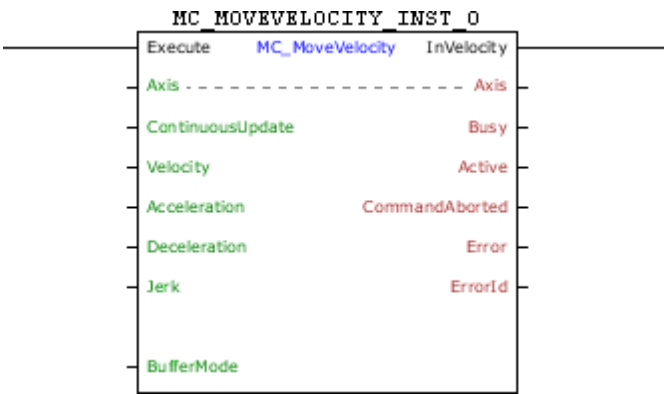
Ao terminar o posicionamento, a saída Done do segundo bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está

setada.

10.2.7.1.26.3 MC_MoveVelocity

Bloco responsável por realizar um movimento para a velocidade programada.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	66 Bytes
Tamanho de Memória de Dados	24 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	Jerk	REAL	Jerk
	BufferMode	REAL	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InVelocity	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_MOVEVELOCITY_INST_0	MC_MOVEVELOCITY	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, executará um movimento para a velocidade configurada no argumento Velocity com uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. A direção do movimento dependerá do sinal da velocidade: se a velocidade for maior que zero, o movimento será na direção positiva (sentido horário) e se a velocidade for menor que zero, o movimento será na direção negativa (sentido anti horário).



NOTA!

Caso o valor do argumento Jerk for diferente de zero, o argumento ContinuousUpdate On-line não terá efeito, sendo considerado os valores dos argumentos no instante da transição positiva de Execute.

Ao executar o bloco MC_MoveVelocity, o *drive* não opera em malha de posição.

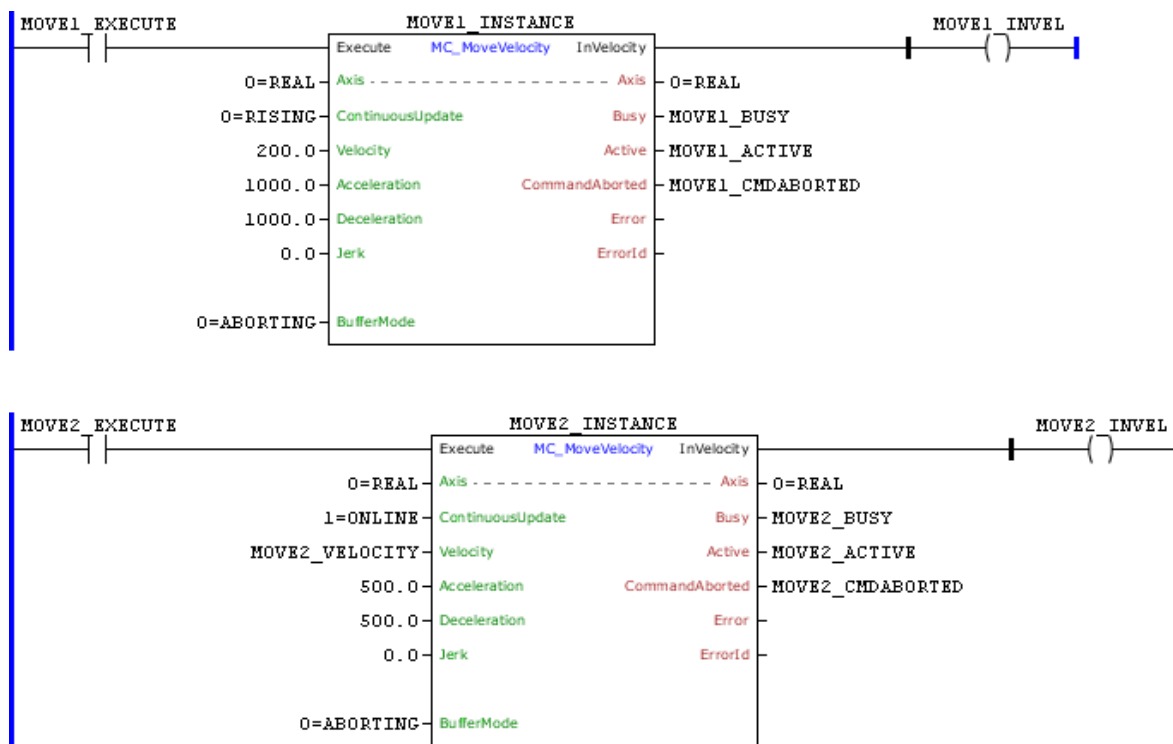
Na execução do movimento o estado do eixo mudará para Continuous Motion.

Quando Execute possui valor FALSE, InVelocity permanece FALSE. A saída InVelocity é ativada quando a velocidade programada é atingida com sucesso, permanecendo em nível TRUE enquanto o bloco estiver ativo. Para finalizar o bloco, é necessária a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.
95	Não é permitido executar posicionamento com Jerk quando outro bloco ativo.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MOVE1_INSTANCE	0	MC_MoveVelocity		
MOVE1_EXECUTE	0	BOOL	0	
MOVE1_INVEL	0	BOOL	0	
MOVE1_BUSY	0	BOOL	0	
MOVE1_ACTIVE	0	BOOL	0	
MOVE1_CMDABORTED	0	BOOL	0	
MOVE2_INSTANCE	0	MC_MoveVelocity		
MOVE2_EXECUTE	0	BOOL	0	
MOVE2_INVEL	0	BOOL	0	
MOVE2_VELOCITY	0	REAL	200.0	
MOVE2_BUSY	0	BOOL	0	
MOVE2_ACTIVE	0	BOOL	0	
MOVE2_CMDABORTED	0	BOOL	0	

GLOBAL LOCAL



Na transição de subida de MOVE1_EXECUTE, o primeiro bloco MC_MoveVelocity é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o movimento para chegar a velocidade de 200 RPM.

No instante em que a velocidade atinge 200 RPM, a saída InVelocity é setada.

Tendo-se a transição de subida de MOVE2_EXECUTE, o segundo bloco MC_MoveVelocity é instantaneamente executado (BufferMode - Aborting). Com isso os sinais Busy e Active deste bloco são setados e inicia-se o movimento para a velocidade de 100 RPM (neste instante MOVE2_VELOCITY contém o valor 100). Ao mesmo tempo os sinais Busy, Active e InVelocity do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

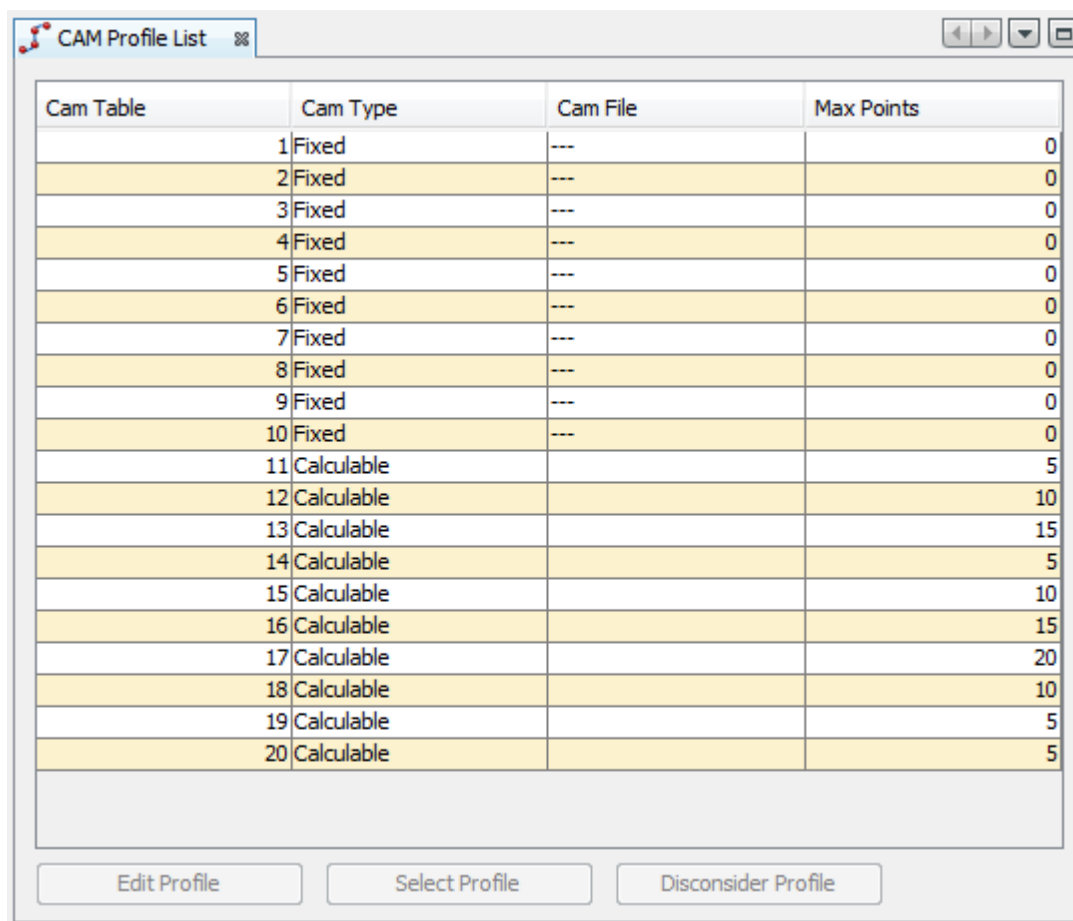
Ao atingir a velocidade de 100 RPM, a saída InVelocity do segundo bloco é setada e permanece até a execução de outro bloco.

Como o argumento ContinuousUpdate está configurado como On-line, com a mudança do valor de MOVE2_VELOCITY para 80, a velocidade muda imediatamente para 80 RPM, sem executar uma rampa de aceleração/desaceleração.

10.2.7.2 Cam Profiles

Permite carregar e editar as tabelas de pontos das curvas CAM.

Acessado pelo comando **CAM Profile List** com o botão direito do mouse na pasta **CAM Profiles** do recurso.



Descrição

As tabelas de pontos (Cam Table) de 1 à 10 são tabelas de pontos fixos, que serão transmitidos no momento do *download* do aplicativo. Para usar as tabelas de 1 à 10, primeiramente o bloco MC_CamTableSelect deve ser executado com a tabela desejada e após o bloco MC_CamIn.

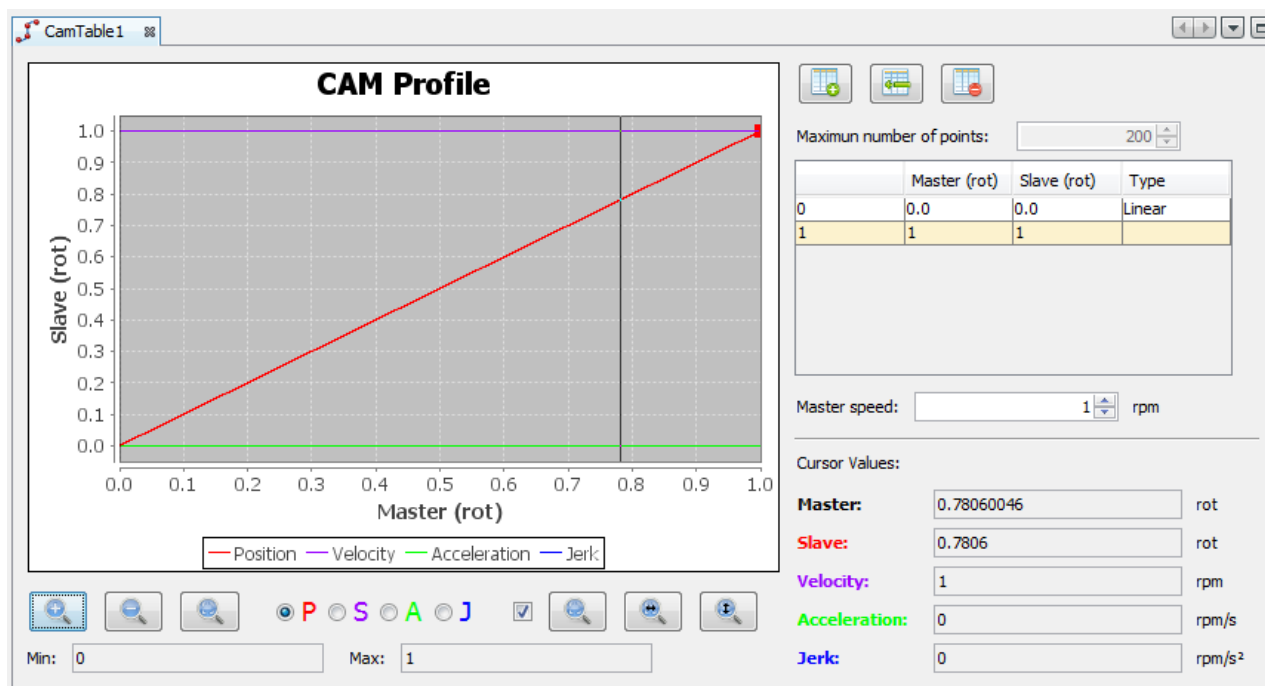
As tabelas de pontos de 11 à 20, são tabelas de pontos variáveis. Para usar as tabelas de 11 à 20, primeiramente o bloco MC_CamCalc deve ser executado com a tabela desejada e após o bloco MC_CamIn.

Para o equipamento SCA06 é permitida a programação de, no máximo, 200 pontos fixos e 100 pontos variáveis, sendo que o número máximo de pontos variáveis de cada tabela deve ser configurado na coluna Max Points, conforme abaixo:

CAM Profile List

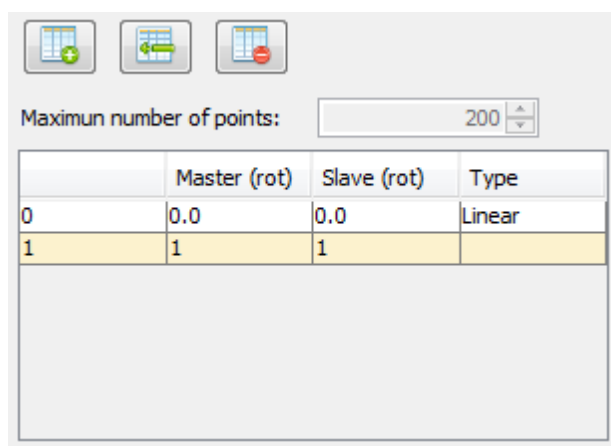
Cam Table	Cam Type	Cam File	Max Points
1	Fixed	---	0
2	Fixed	---	0
3	Fixed	---	0
4	Fixed	---	0
5	Fixed	---	0
6	Fixed	---	0
7	Fixed	---	0
8	Fixed	---	0
9	Fixed	---	0
10	Fixed	---	0
11	Calculable		5
12	Calculable		10
13	Calculable		15
14	Calculable		5
15	Calculable		10
16	Calculable		15
17	Calculable		20
18	Calculable		10
19	Calculable		5
20	Calculable		5

Para editar a tabela cam clique no botão **Editar**, o editor de perfil cam abrirá, conforme figura a seguir:



Nessa janela existe os seguintes controles:

Tabela de pontos:



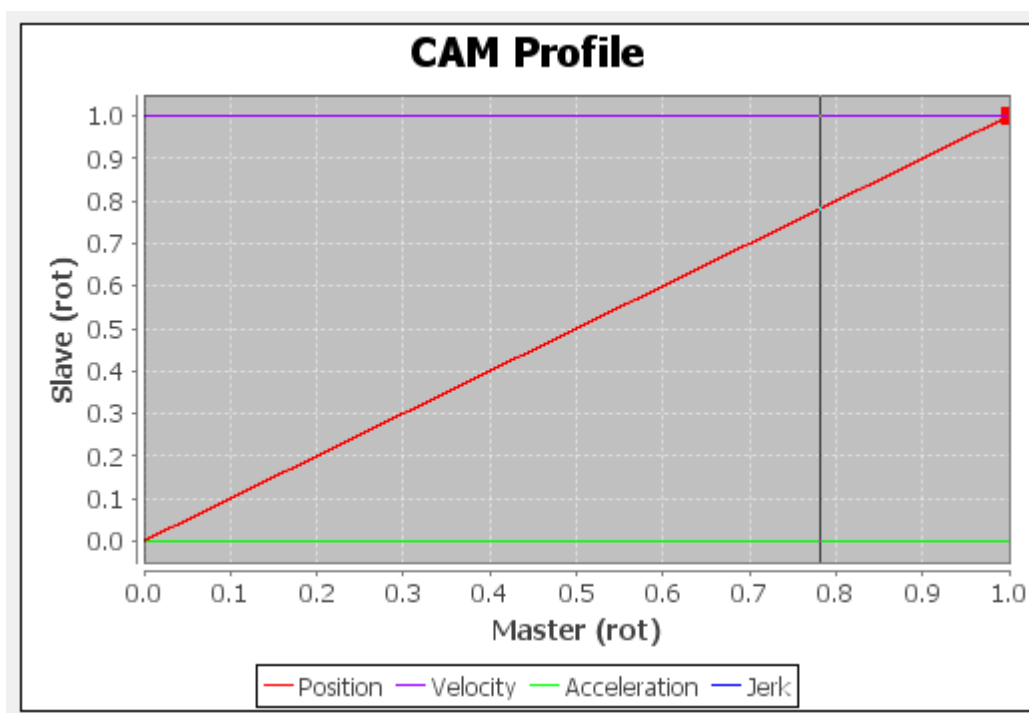
	Master (rot)	Slave (rot)	Type
0	0.0	0.0	Linear
1	1	1	



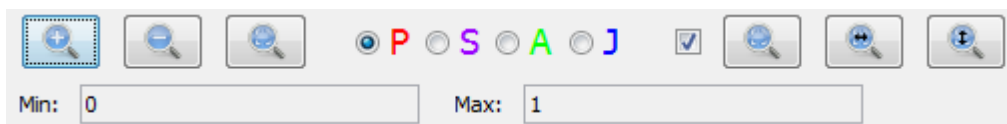
NOTA!

O bloco CAM é sempre relativo, logo o primeiro ponto da tabela de pontos sempre será Mestre (eixo virtual) = 0 e escravo (eixo real do *drive*) = 0.

Gráfico do perfil:



Ferramentas de controle do gráfico:



Min: 0 Max: 1

Valores do cursor:

Valores relativos ao ponto selecionado do cursor.

Cursor Values:		
Master:	0.78060046	rot
Slave:	0.7806	rot
Velocity:	1	rpm
Acceleration:	0	rpm/s
Jerk:	0	rpm/s ²

Velocidade do mestre:

Velocidade utilizada para cálculo da velocidade, aceleração e jerk do escravo.

Master speed: rpm

**NOTA!**

A velocidade, aceleração e jerk do escravo devem ser utilizados como referência para o desenvolvimento do perfil cam, onde os mesmos são calculados numericamente e não levam em consideração carga, inércia, torque e a dinâmica do *drive*.

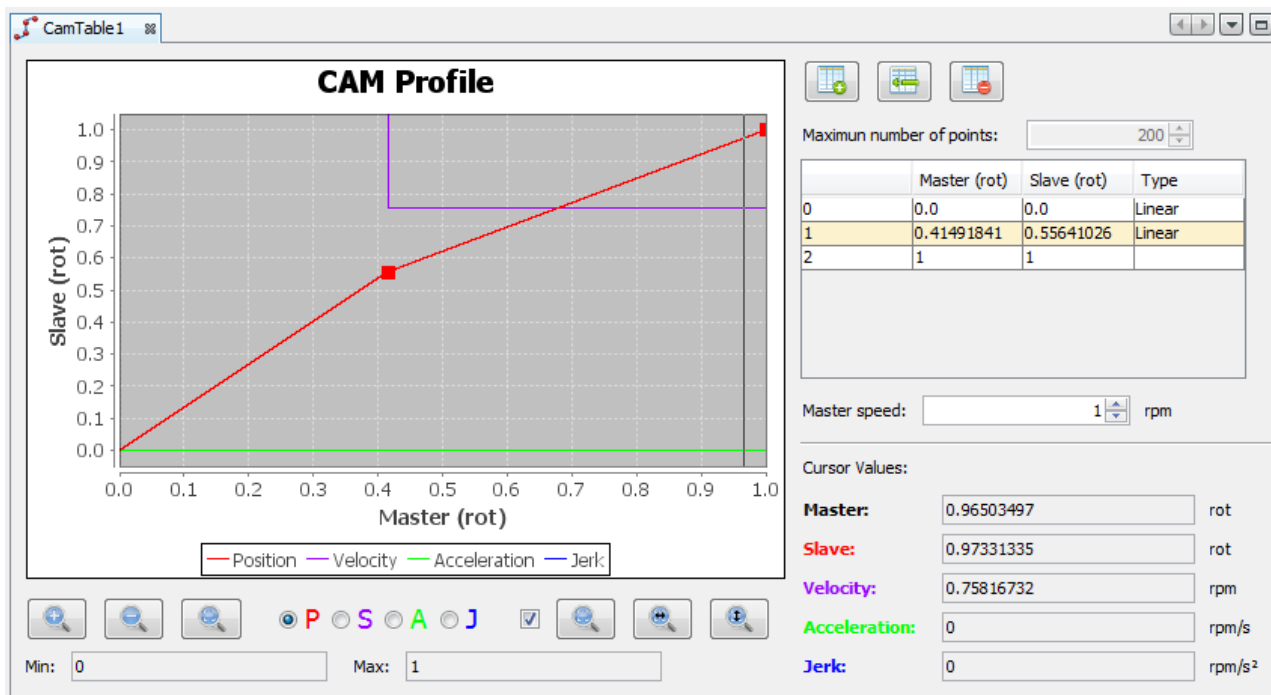
Adicionando um novo ponto no perfil CAM

Um ponto pode ser adicionado através dos botões adicionar ou inserir ponto ou através de um duplo clique do mouse no gráfico na posição onde deseja-se adicionar o ponto. O duplo clique pode ser feito em qualquer região do gráfico. Caso já exista uma interpolação nessa região o editor irá inserir esse ponto entre os dois pontos da interpolação.

O ponto é sempre adicionado como interpolação do tipo linear.

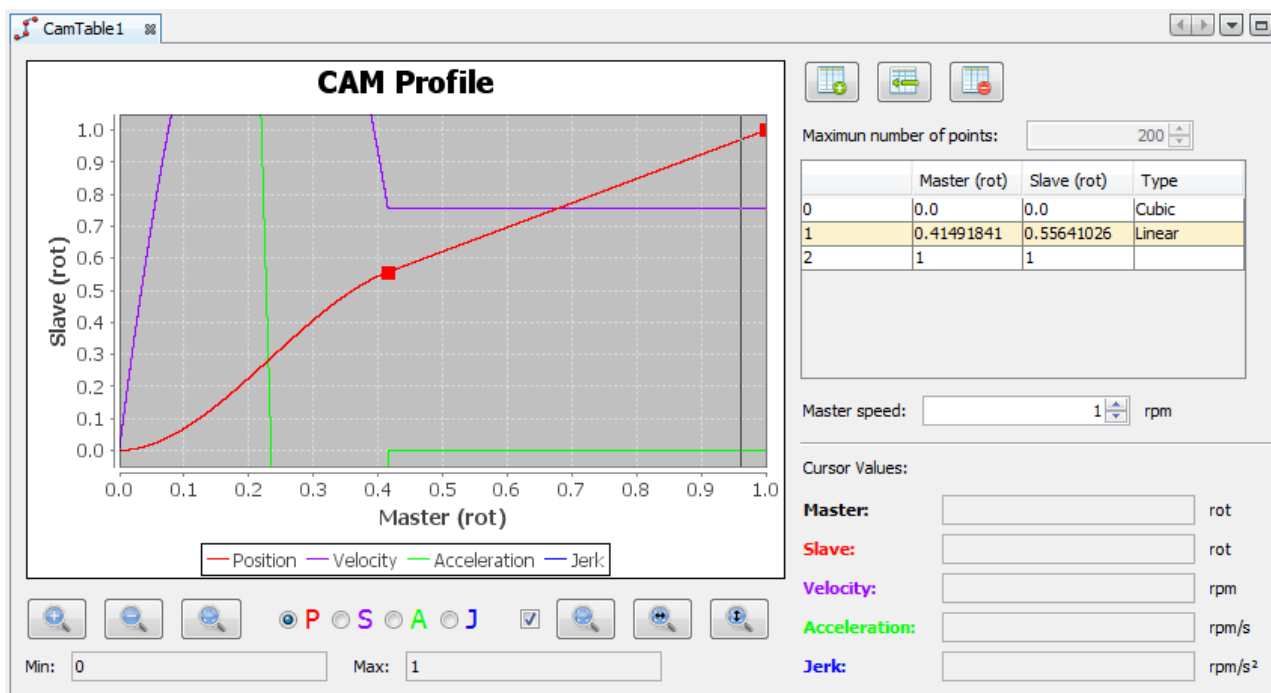
Quando é adicionado ou inserido um ponto através dos respectivos botões os valores de mestre e escravo vem zerados. No caso da inserção de ponto isso pode ocasionar uma interrupção do perfil, pois a posição do mestre deve sempre crescer em relação a origem, então, deve-se editar o valor do mestre e escravo clicando sobre suas células na tabela de pontos.

Na figura a seguir foi inserido um ponto através do duplo clique do mouse:

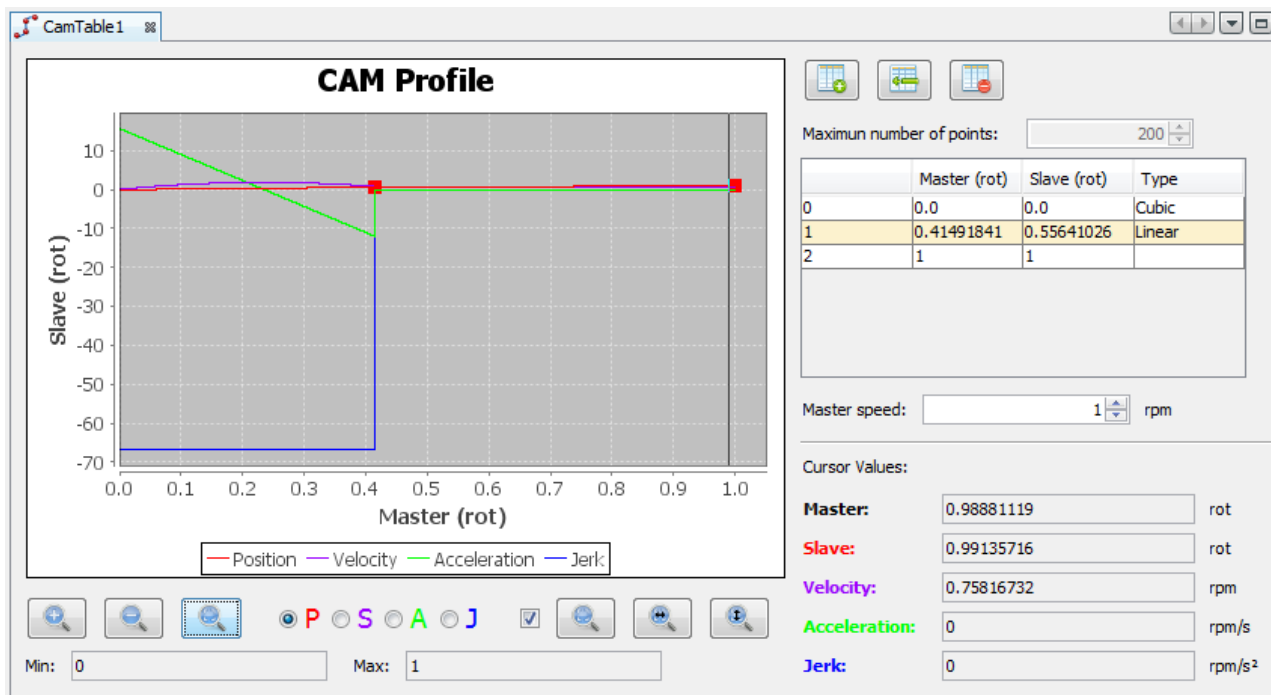


Para alterar o tipo da interpolação clique na célula de tipo na linha correspondente a origem da interpolação e selecione a desejada.

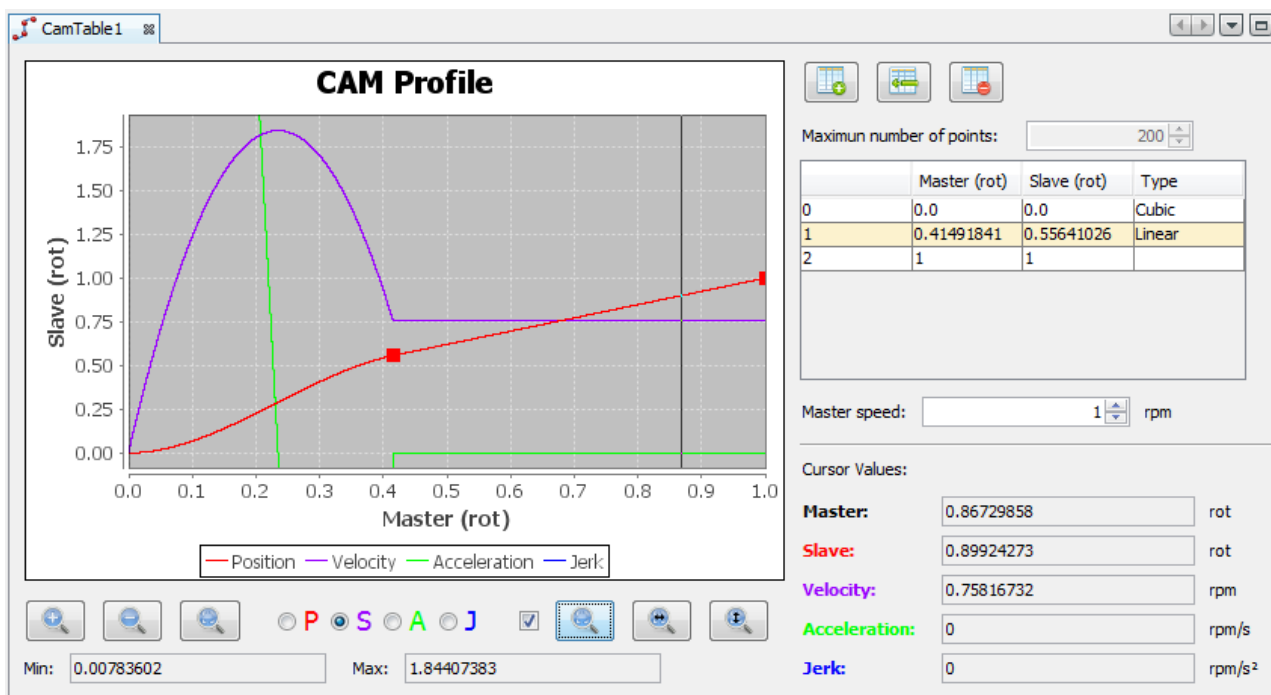
Na figura a seguir foi alterado o ponto para interpolação tipo cúbica.



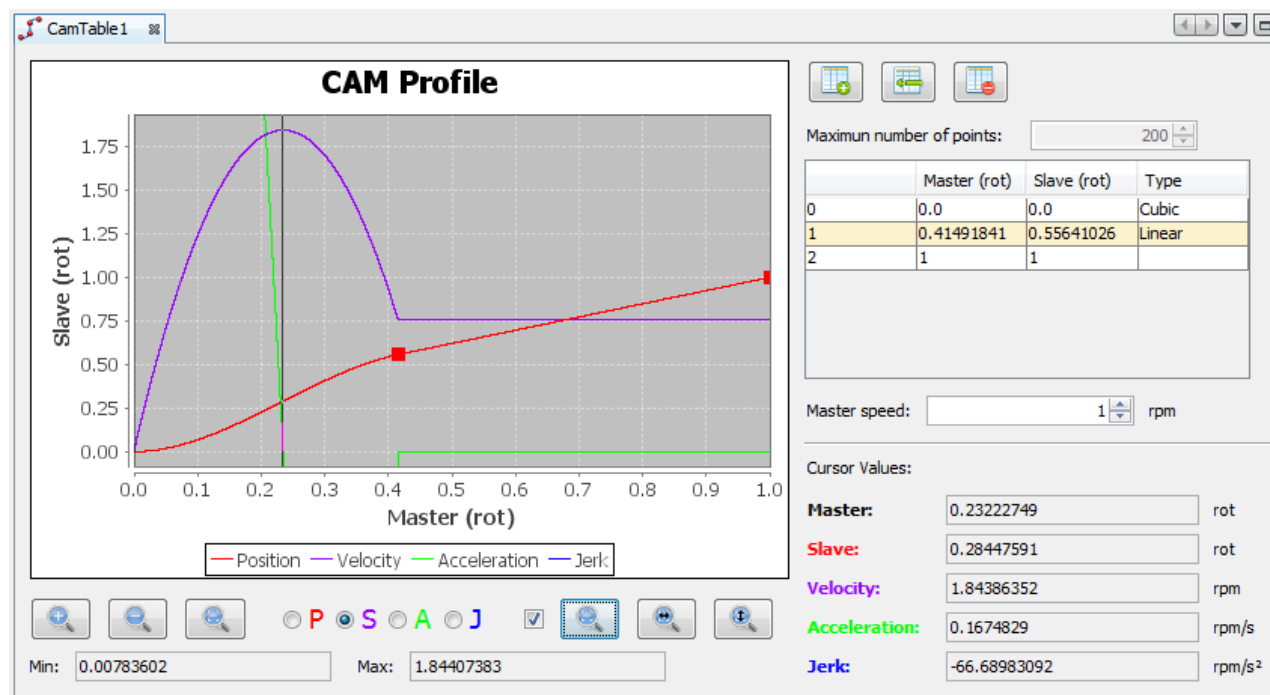
Agora nessa curva já é possível observar outras grandezas além da posição como velocidade, aceleração e jerk. Para uma melhor visualização de todas as grandezas podemos utilizar o botão **Ajusta Zoom Tudo** conforme figura a seguir.



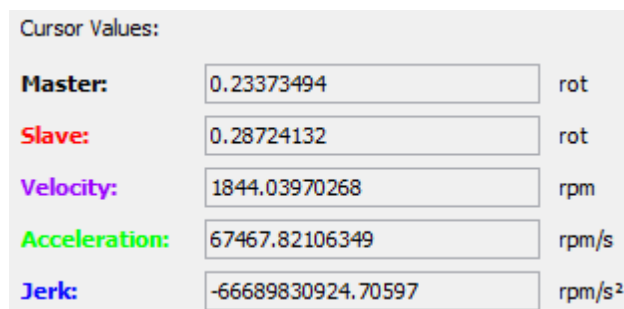
Da mesma maneira podemos escolher uma das grandeza e utilizar o botão **Aplica Zoom Selecionado**. No exemplo a seguir foi efetuado um zoom na velocidade.



Outra ferramenta interessante de ser citada é o cursor. No exemplo a seguir posicionaremos o cursor no ponto de máxima velocidade.

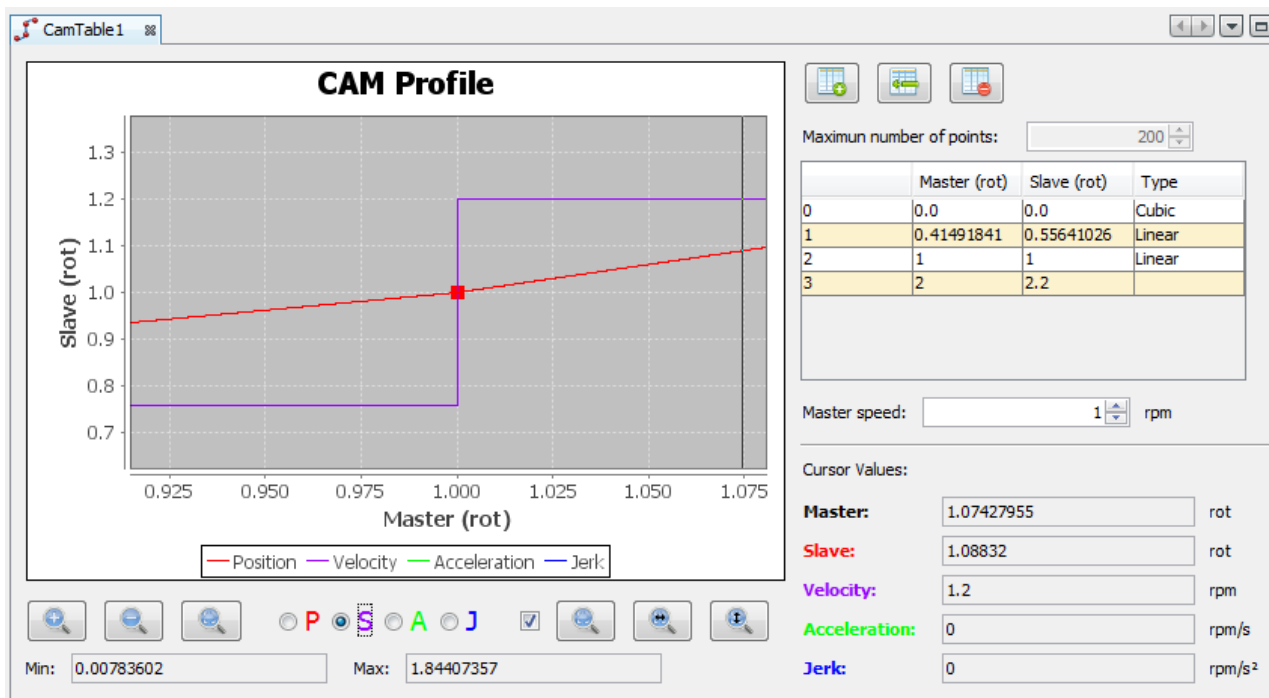


Deve-se lembrar que as grandezas velocidade, aceleração e jerk do escravo são dependentes da velocidade do mestre, então é interessante altera-la de modo a simular algo muito próximo ao real. Na figura a seguir a velocidade do mestre será alterada para 1000 rpm e analisaremos a mesma posição do cursor.



Durante o projeto do perfil cam todas essas grandezas devem ser observadas pois as mesmas poderão ou não ser cumpridas em função de limitações mecânicas, elétricas e eletrônicas dos equipamentos envolvidos.

Como os gráficos de aceleração e jerk são calculados levando em consideração a interpolação entre dois pontos, nas junções entre interpolações lineares a aceleração e jerk serão mostrados como iguais a zero. Mas sabemos que teoricamente num degrau de velocidade a aceleração e jerk são infinitos, na prática a aceleração e jerk nesse momento dependerá também das limitações mecânicas, elétricas e eletrônicas dos equipamentos envolvidos. Esses degraus de velocidade devem ser observados e considerados no projeto do perfil cam. Na figura a seguir é exemplificado esta situação.



O bloco CAM tem disponível dois tipos de interpolação, linear e cúbica. Sendo utilizada as seguintes equações :

Linear:

$$pe = pie * \left(\frac{p_{fm} - pm}{p_{fm} - pim} \right) + p_{fe} * \left(\frac{pm - pim}{p_{fm} - pim} \right)$$

$$ve = \left(\frac{-pie}{p_{fm} - pim} + \frac{p_{fe}}{p_{fm} - pim} \right) * vm$$

$$ae = 0$$

$$je = 0$$

Cúbica:

$$pe = a * (pm - pim)^3 + b * (pm - pim)^2 + c * (pm - pim) + pie$$

$$ve = (3 * a * (pm - pim)^2 + 2 * b * (pm - pim) + c) * vm$$

$$ae = (6 * a * (pm - pim) + 2 * b) * vm^2$$

$$je = 6 * a * vm^3$$

Onde:

pe = posição do escravo

ve = velocidade do escravo

ae = aceleração do escravo

je = jerk do escravo

pm = posição do mestre

vm = velocidade do mestre

pim = posição inicial do mestre
pfm = posição final do mestre
pie = posição inicial do escravo
pfe = posição final do escravo
a = coeficiente calculado pelo editor CAM
b = coeficiente calculado pelo editor CAM
c = coeficiente calculado pelo editor CAM

Alterando um ponto no perfil CAM

Um ponto pode ser alterado através da tabela de pontos pela edição direta ou movendo o ponto no gráfico. Para mover o ponto no gráfico leve o mouse até o ponto em questão que é marcado com um quadrado vermelho, clique sobre o mesmo e mantenha o mouse pressionado e arraste o mesmo para a nova posição. Ao clicar sobre o ponto a tabela de pontos será deslocada para o ponto em questão, selecionando a célula relacionada.

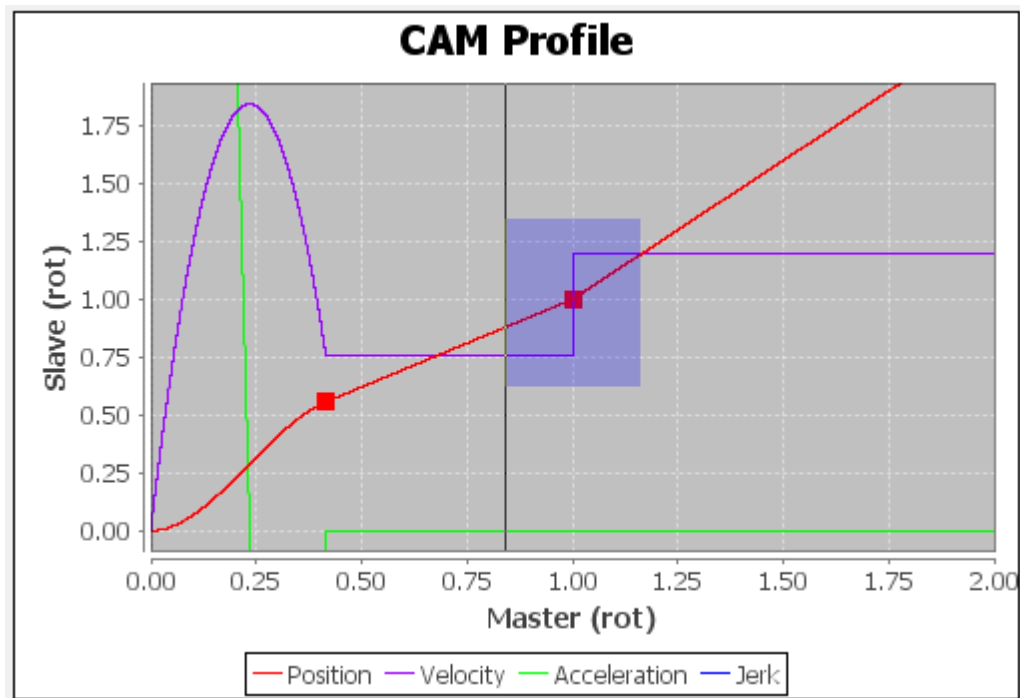
A operação de mover o ponto no gráfico é interativa e calcula todo o perfil a cada mudança do ponto em questão. O novo ponto pode ser visto na tabela de pontos.

Removendo um ponto no perfil CAM

O mesmo é removido diretamente na tabela de pontos. Para isso selecione uma das células respectiva ao ponto e clique no botão **Remover Ponto**.

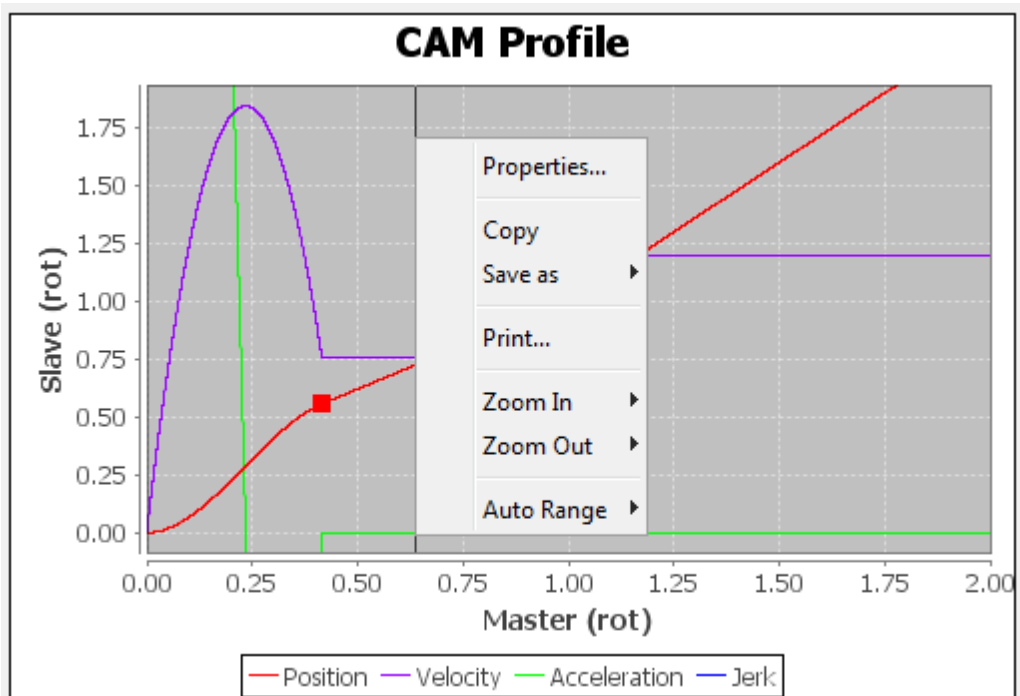
Zoom de uma área determinada do gráfico

Clique com o mouse sobre um dos cantos da região que deseja executar o zoom e mantenha o mouse pressionado, mova o mouse de modo a marcar uma região. Nesse momento um retângulo aparecerá no gráfico, solte o botão do mouse. Na figura a seguir um exemplo desse zoom.

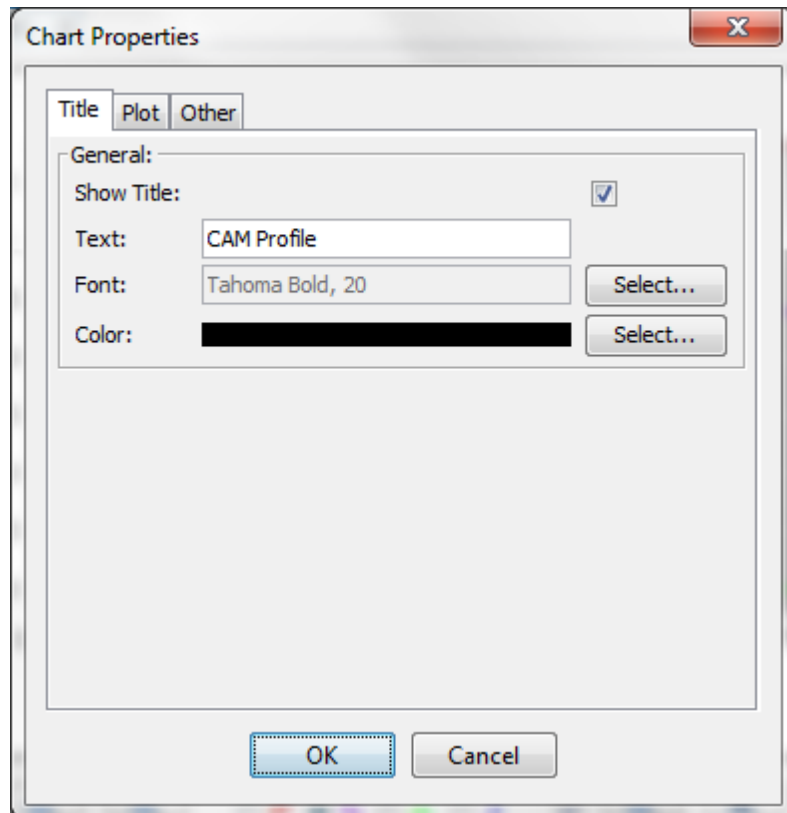


Menu gráfico

Para ter acesso ao menu do gráfico clique com o botão direito do mouse sobre a área do gráfico, após o seguinte menu aparecerá.



Na figura a seguir é mostrada a caixa de propriedades do gráfico.



10.2.7.3 Estruturas

Uma estrutura consiste num agrupamento de dados utilizada para definir uma receita ou um objeto.

No programa Ladder é possível criar variáveis do tipo estrutura e utilizá-las nos blocos. Para acesso aos membros internos da estrutura, utiliza-se o '.' seguido do seu respectivo membro.

Criando uma Estrutura

1. Com o botão direito do mouse sobre a pasta **Estrutura**, clique em **Novo Arquivo**.

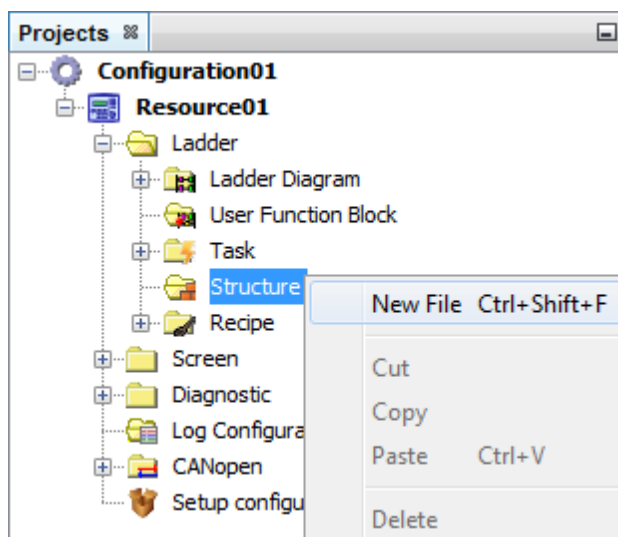


Figura 1 - Criando uma Estrutura

2. Defina o nome do arquivo e aperte o botão **Próximo**.

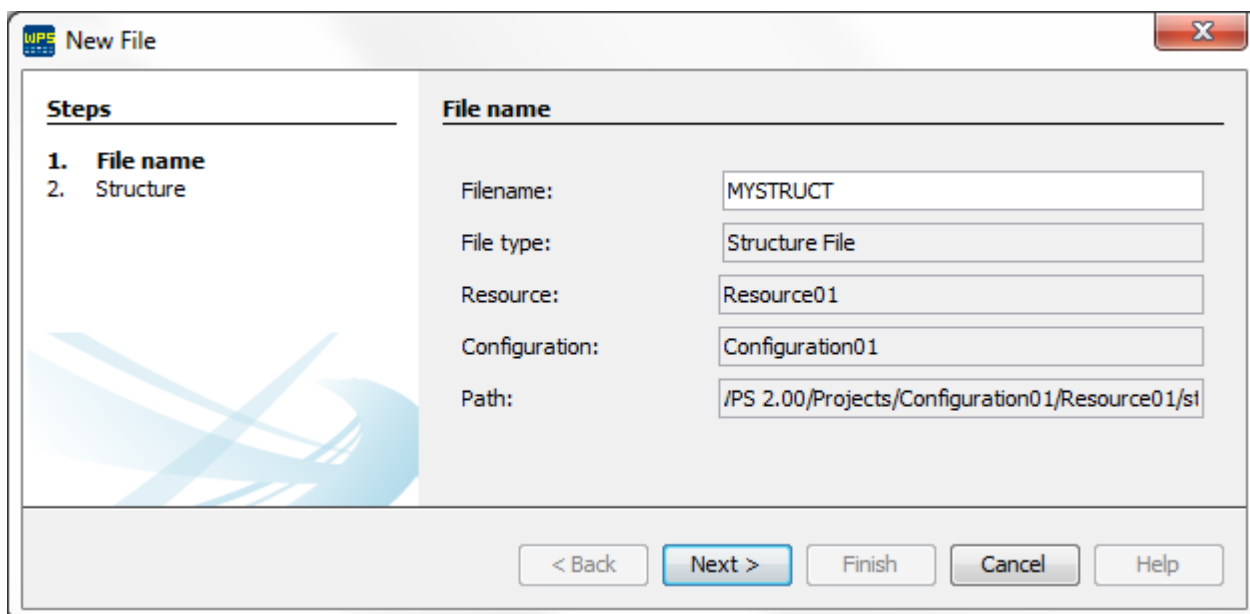


Figura 2 - Definindo o nome da Estrutura

3. Configure a estrutura utilizando os botões apresentados na figura abaixo.

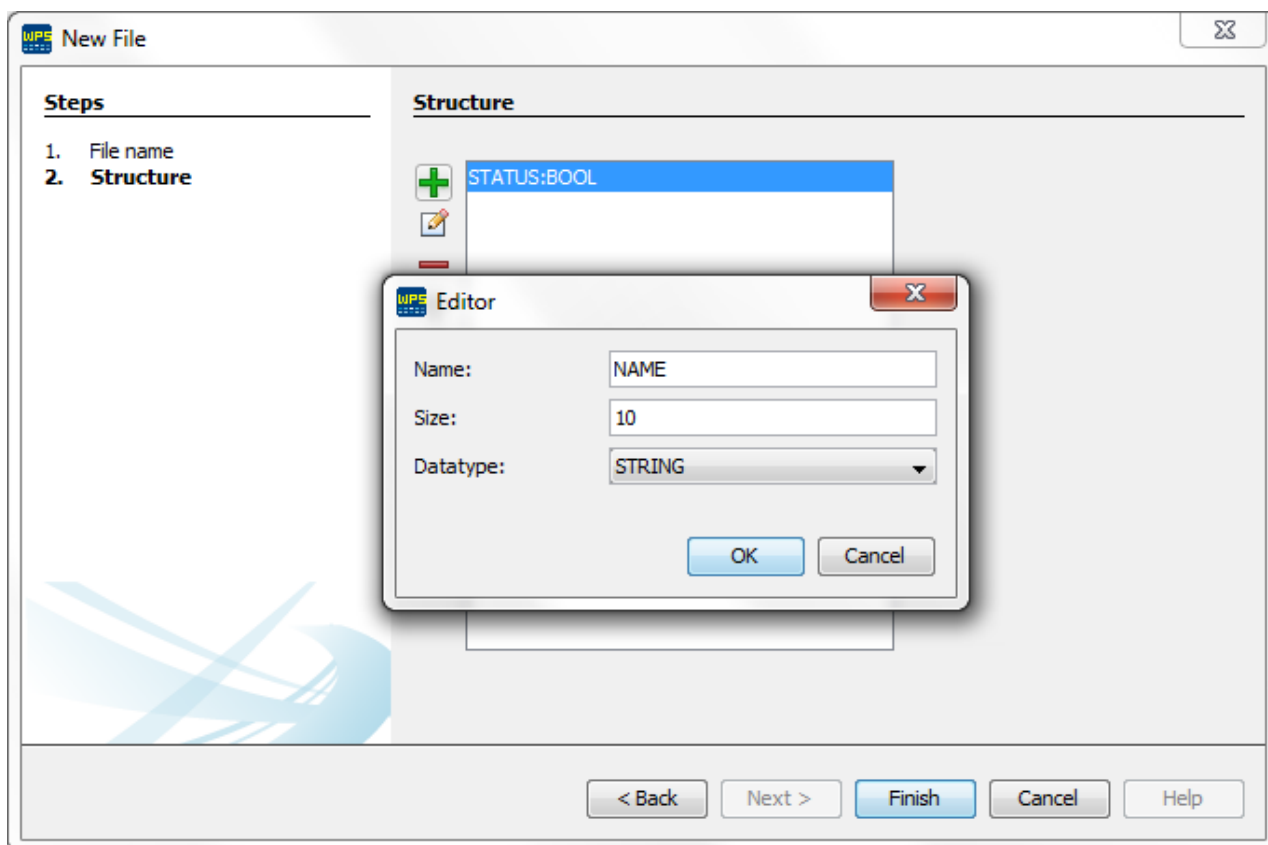


Figura 3 - Editando a Estrutura

4. Após concluir a edição da estrutura, clique no botão **Finalizar**.

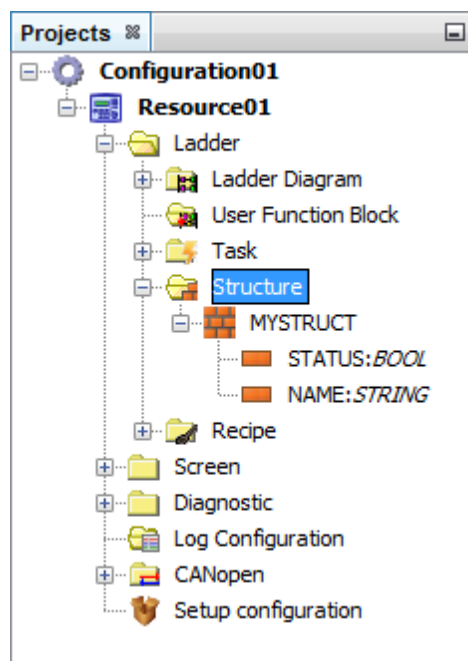


Figura 4 - Estrutura criada no projeto

Editando uma Estrutura

Para editar uma estrutura é necessário dar um duplo-clique na estrutura desejada. Uma janela abrirá, similar à da figura 3, permitindo inserir, apagar ou mover dados.

10.2.8 Diagnóstico

[Painel de Monitoração](#)

[Log](#)

[Trace](#)

10.2.8.1 Painel de Monitoração

[Main Signals](#)

10.2.8.1.1 Main Signals

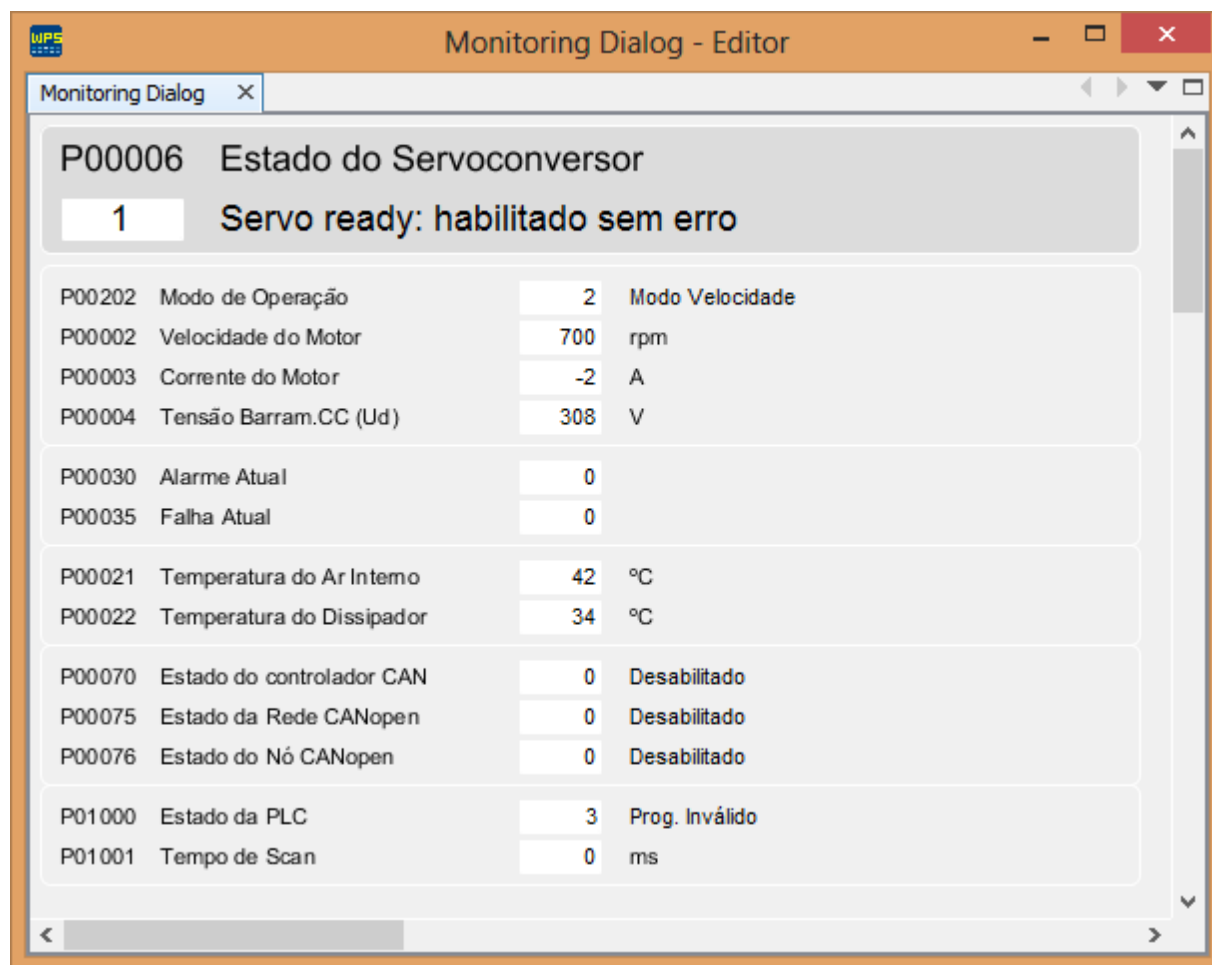
A janela Principais Sinais fornece uma visão geral dos principais sinais do equipamento:

- Estado do *drive* (P00006),
- Indicações gerais (P00202, P00002, P00003 e P00004),
- Alarme (P00030) e falha (P00035),
- Temperaturas (P00021 e P00022),
- Estados das redes (P00070, P00075 e P00076), e
- Informações do PLC (P01000 e P01001).

Os principais sinais são detalhados abaixo.

Parâmetro	Função	Descrição
P00006	Estado do Servoconversor	Indica o estado atual do servoconversor
P00202	Modo de Operação	Define o modo de operação do servoconversor, ou seja, qual a variável que se deseja controlar: Torque, Velocidade ou se o controle será feito via Ladder, CANopen ou Profibus
P00002	Velocidade do Motor	Indica o valor da velocidade real em rpm do servomotor, exceto quando for programado para receber realimentação de posição/velocidade externa
P00003	Corrente do Motor	Indica o valor da corrente Iq de saída, em amperes rms, do servoconversor
P00004	Tensão do Barramento CC	Indica a tensão atual no barramento CC em volts (V)
P00030	Alarme Atual	Indicam o número do alarme que eventualmente esteja presente no servoconversor
P00035	Falha Atual	Indicam o número da falha que eventualmente esteja presentes no servoconversor
P00021	Temperatura do ar interno	Este parâmetro apresenta, em graus Celsius, a temperatura do ar interno
P00022	Temperatura do dissipador	Este parâmetro apresenta, em graus Celsius, a temperatura do dissipador.
P00070	Estado do Controlador CAN	Indica o estado do controlador CAN, responsável por enviar e receber telegramas CAN
P00075	Estado da Rede CANopen	Indica o estado da comunicação CANopen, informando se o protocolo foi inicializado corretamente e o estado do serviço de guarda do escravo
P00076	Estado do Nó CANopen	Cada equipamento na rede CANopen possui um estado associado. É possível ver o estado atual do servoconversor através deste parâmetro
P01000	Estado do PLC	Permite ao usuário visualizar o status do programa
P01001	Tempo de Scan	Permite ao usuário monitorar o tempo do ciclo de varredura do programa em milissegundos

A janela pode ser visualizada abaixo.



10.2.8.2 Log

[Visão Geral](#)

[Configuração](#)

10.2.8.2.1 Visão Geral

Função *log* permite a visualização dos alarmes/falhas atuais e também dos últimos alarmes/falhas de uma forma mais amigável e centralizada do que a HMI do equipamento, exibindo os dados de forma tabulada.

Todos alarmes/falhas descarregados são gravados em um arquivo a fim de manter um histórico para consulta posterior, é possível também exportar os dados registrados para um arquivo csv.

Abaixo segue uma visão geral sobre a tela de *logs* do SCA06.

Log 2014-03-04 12:58:16

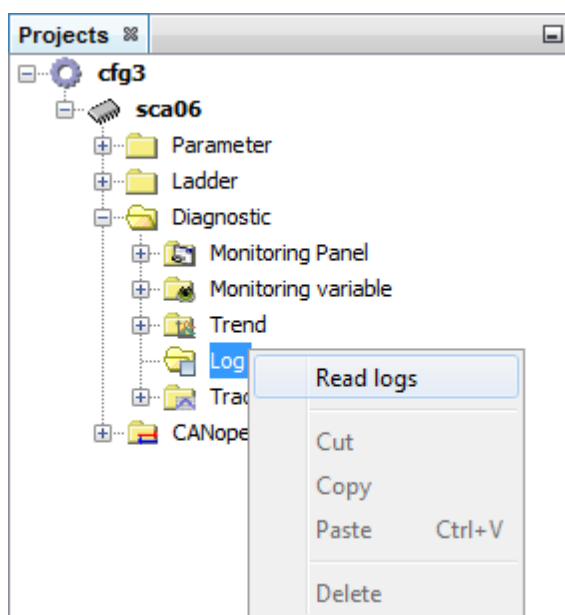
1		Current Fail: OK	2	3
		Current Alarm: OK		
Type	Date	Code	Description	
FAIL	Jan 3, 1594 05:15:00	1105	Defect on the internal circuit of the control card	
ALARM	Apr 27, 1577 09:36:00	127	Initial value of the parameter out of the limits	
FAIL	Dec 22, 1576 04:11:00	32	Resolver Cable disconnected, or overtemperature on the servomotor	

1. **Ações:** segue abaixo a descrição de cada ação em sequência:
 - 1.1. Leitura On-Line: Esta ação fica buscando novas falhas/alarmes constantemente no equipamento caso alguma ocorra a tabela é atualizada automaticamente.
 - 1.2. Ler Logs: Esta ação lê todas as falhas/alarmes uma única vez e atualiza a tabela.
 - 1.3. Exportar: Exportação dos dados exibidos na tabela para um arquivo CSV.
2. **Falha e Alarme Atual:** nestes campos são mostrados a falha e a alarme atual do equipamento caso existam, se não houverem é exibido a mensagem **OK**.
3. **Tabela de Falhas e Alarmes:** nesta tabela são mostradas todas falhas e alarmes assim como a data em que ocorreram, código e descrição.

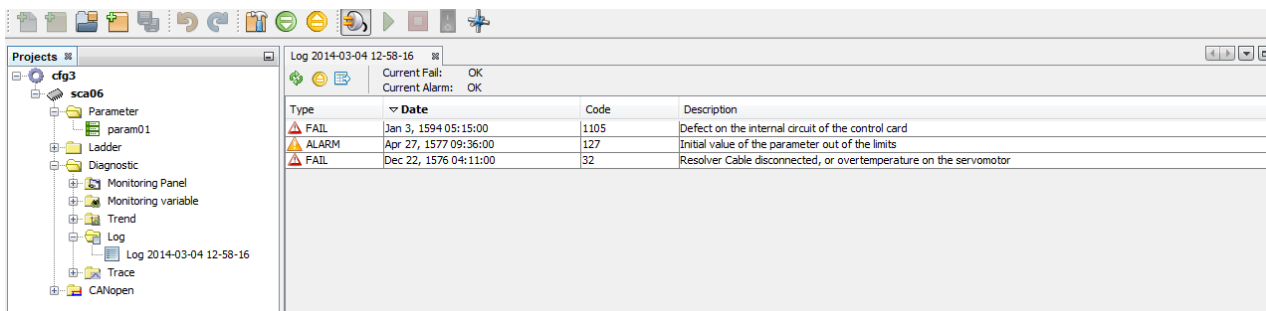
10.2.8.2.2 Configuração

Para criar um novo arquivo de *logs* pasta executar a operação abaixo:

1. Criar um novo arquivo de *logs* selecionando a opção **Ler Logs**, é necessário estar conectado ao equipamento para executar esta operação.



2. Após executada a etapa de ler *logs*, o arquivo de *logs* será automaticamente criado com um nome composto com a data e hora atual e com todos alarmes e falhas presentes no equipamento lidos.



10.2.8.3 Trace

[Visão Geral](#)

[Configuração](#)

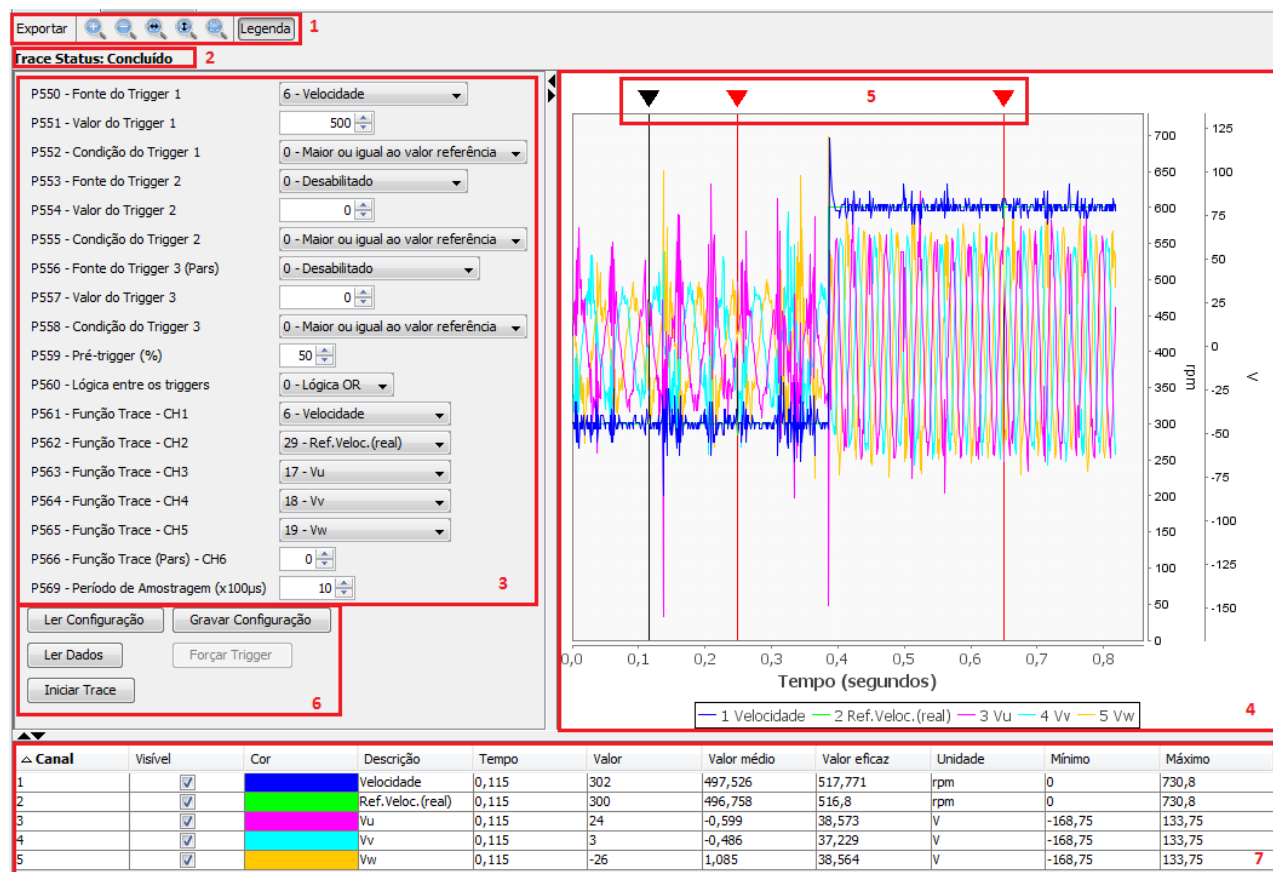
10.2.8.3.1 Visão Geral

A função trace é utilizada para registrar até 6 (seis) variáveis de interesse do SCA06 (como corrente, tensão, velocidade, etc) quando ocorre um determinado evento no sistema. Este evento no sistema, por desencadear o

processo de armazenamento das variáveis, é chamado de “trigger” (disparo) e o usuário pode definir até 3 (três) condições de trigger e a lógica a ser usada nos mesmos (lógica AND ou OR).

As variáveis armazenadas podem ser vistas sob a forma de gráficos utilizando o WPS executado em um PC conectado via USB ou via serial ao SCA06.

Abaixo segue uma visão geral sobre a tela de configuração da função trace do SCA06.



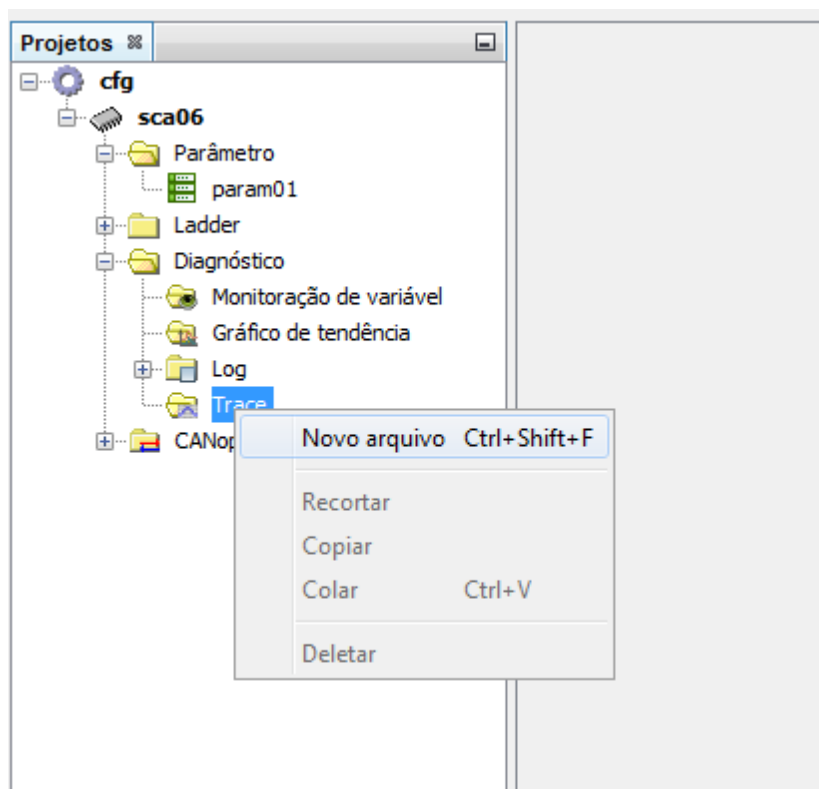
- Zoom Gráfico.** Nesta barra estão as opções para controle do gráfico como exportar para arquivo de imagens, aumentar zoom, diminuir zoom, ajustar largura, ajustar altura, ajustar tudo e exibir ou não exibir as legendas do gráfico.
- Estado Trace.** Neste item é mostrado o estado atual da função trace, pode estes serem não iniciado, ocorreu trigger e concluído.
- Parâmetros.** Nesta parte ficam todos os parâmetros que podem ser configurados na rotina trace, como triggers, condições, canais a serem monitorados e período de amostragem.
- Gráfico.** Nesta área é exibido o gráfico após a conclusão do trace, na parte inferior fica a linha do tempo e na parte lateral direita ficam os valores separados por unidade de medida.
- Marcadores.** Os marcadores ficam dentro da área do gráfico, após o gráfico montado basta clicar com o mouse em cima do marcador preto para criar marcadores vermelhos (fixos). É possível a adição de dois marcadores fixos, estes marcadores fixos são utilizados para calcular os valores médios e eficazes entre os dois pontos.
- Comandos do trace.** Segue abaixo descrição das funções dos comandos:
 - 6.1.Ler configuração: Lê os parâmetros de configuração do trace e atualiza os parâmetros na tela (item 3).
 - 6.2.Gravar configuração: Envia os parâmetros de configuração (item 3) do trace ao equipamento.
 - 6.3.Ler Dados: Comando utilizado somente quando o estado do trace estiver concluído, ou seja, já existe um trace concluído no equipamento e você quer apenas descarregar os dados sem iniciar um novo trace.
 - 6.4.Força Trigger: Força o disparo do trigger independente das condições.
 - 6.5.Iniciar Trace: Inicia a função trace.

7. **Tabela de Canais.** Nesta tabela é possível visualizar os dados dos canais escolhidos, além de poder ocultar canais (Visível), alterar cor dos canais (Cor) e ajustar os limites do gráfico por unidade de medida (Máximo).

10.2.8.3.2 Configuração

Abaixo segue a lista dos passos necessários para criação de uma configuração trace no SCA06.

1. Criação de um novo arquivo trace.



WPS Novo arquivo

Etapas

1. File name

File name

Nome do arquivo: trace01

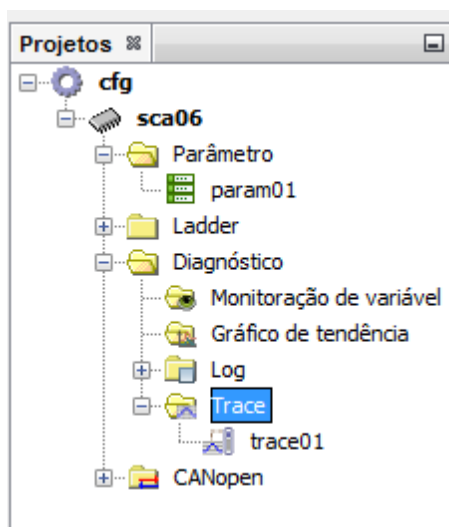
Tipo do arquivo: Trace

Recurso: sca06

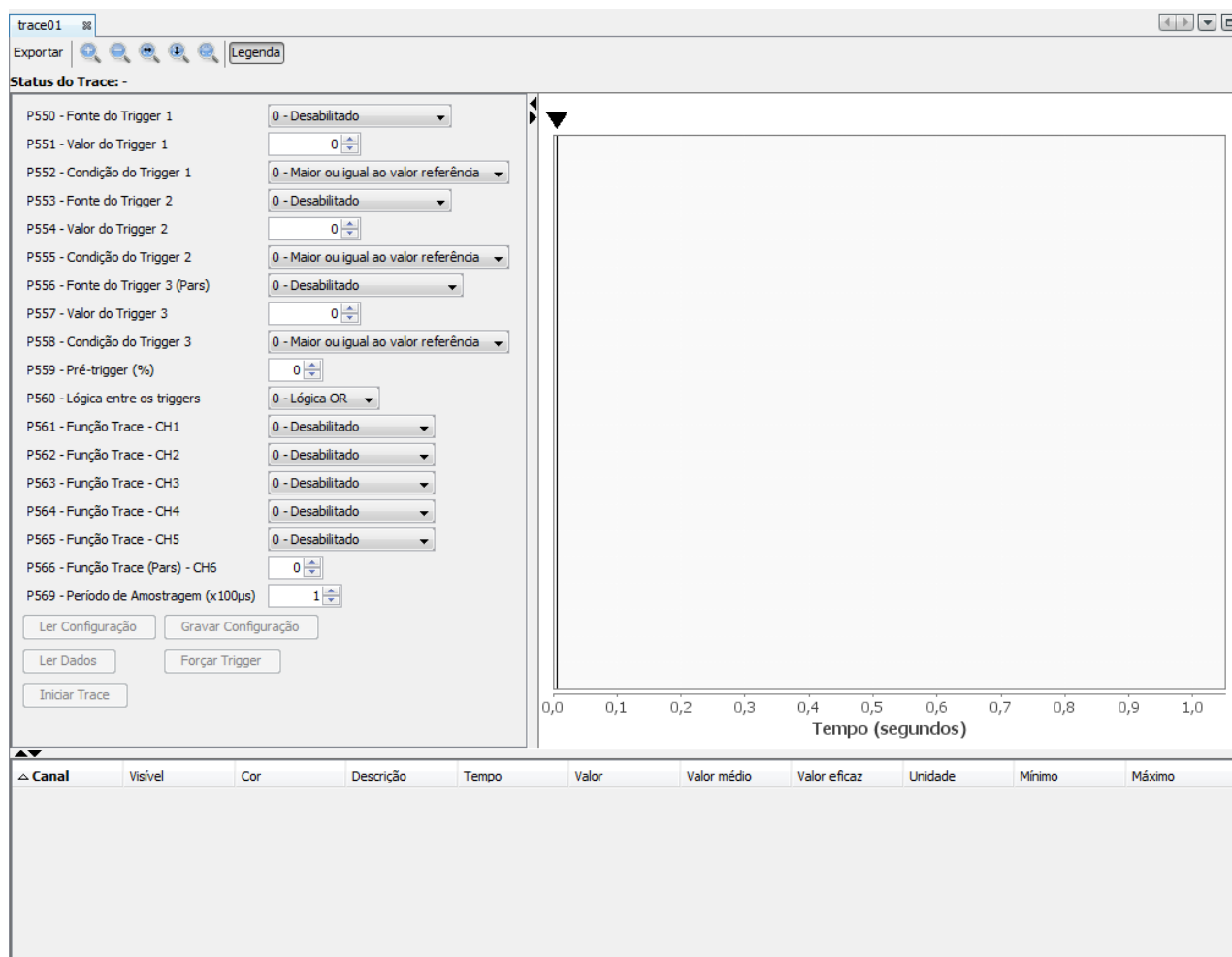
Configuração: cfg

Caminho: C:/Projetos/WPS/trunk/Projects/cfg/sca06/trace

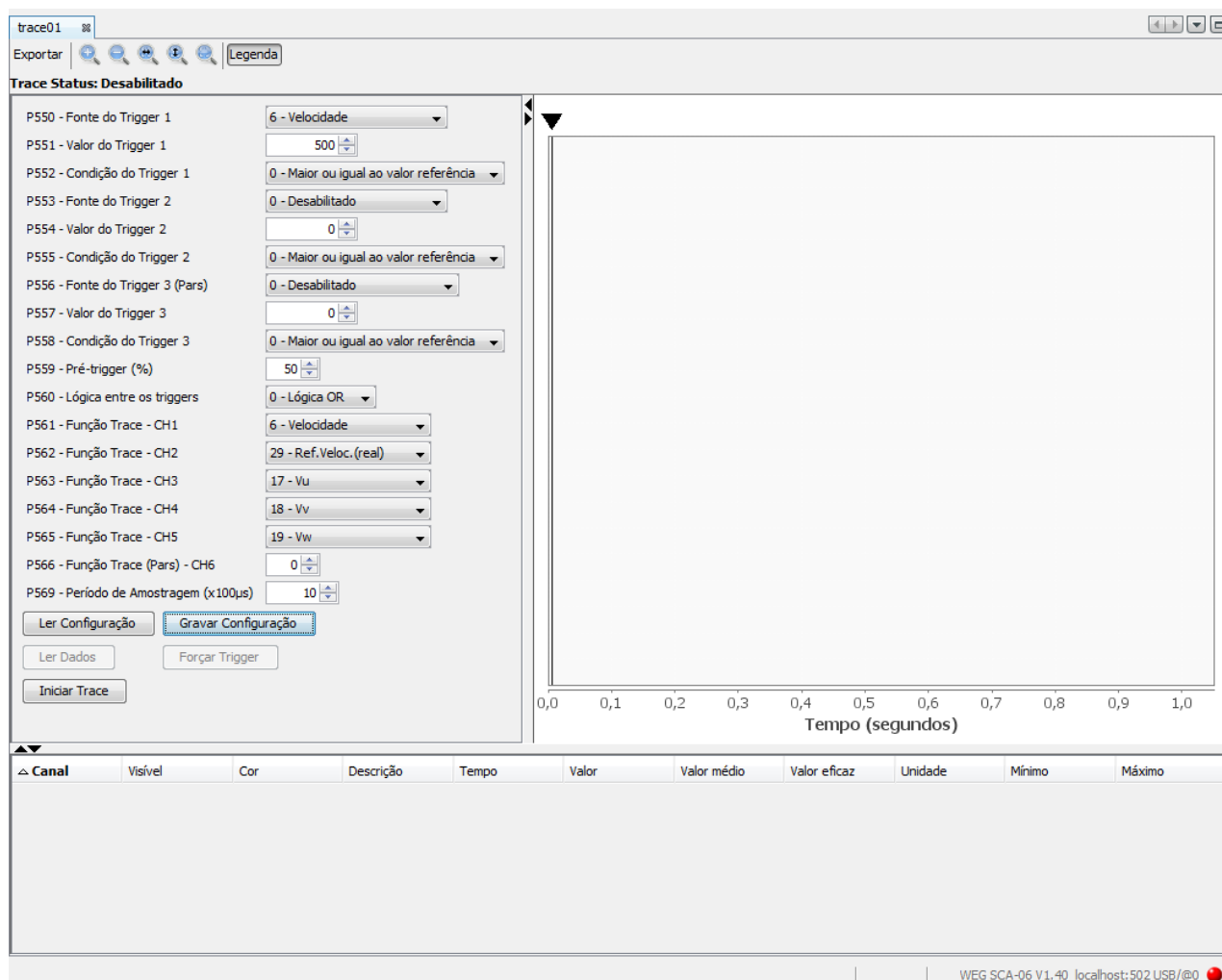
< Voltar Próximo > Finalizar Cancelar Ajuda



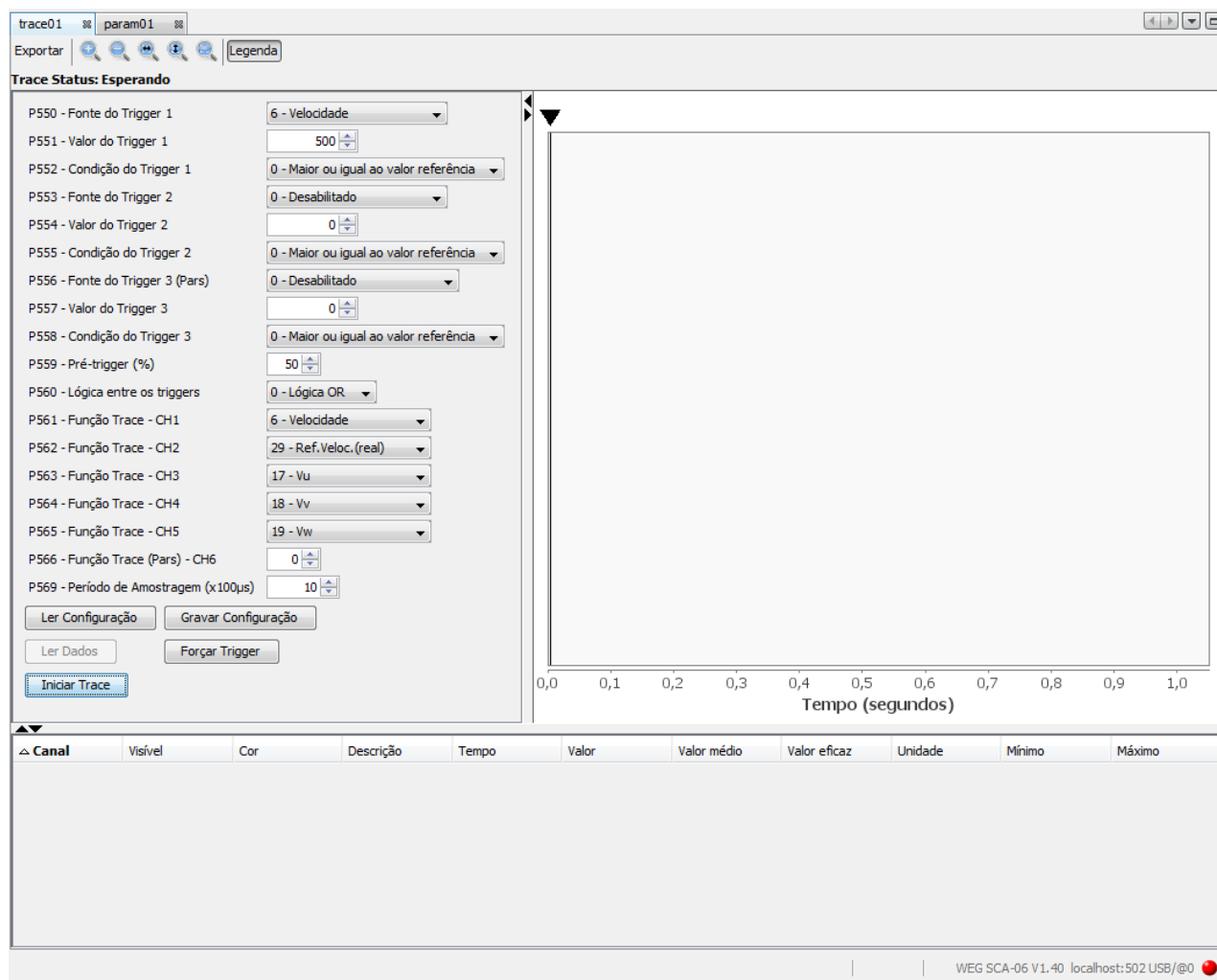
2. Após a criação do arquivo trace, é necessário configurá-lo com as configurações desejadas na parte de parâmetros.



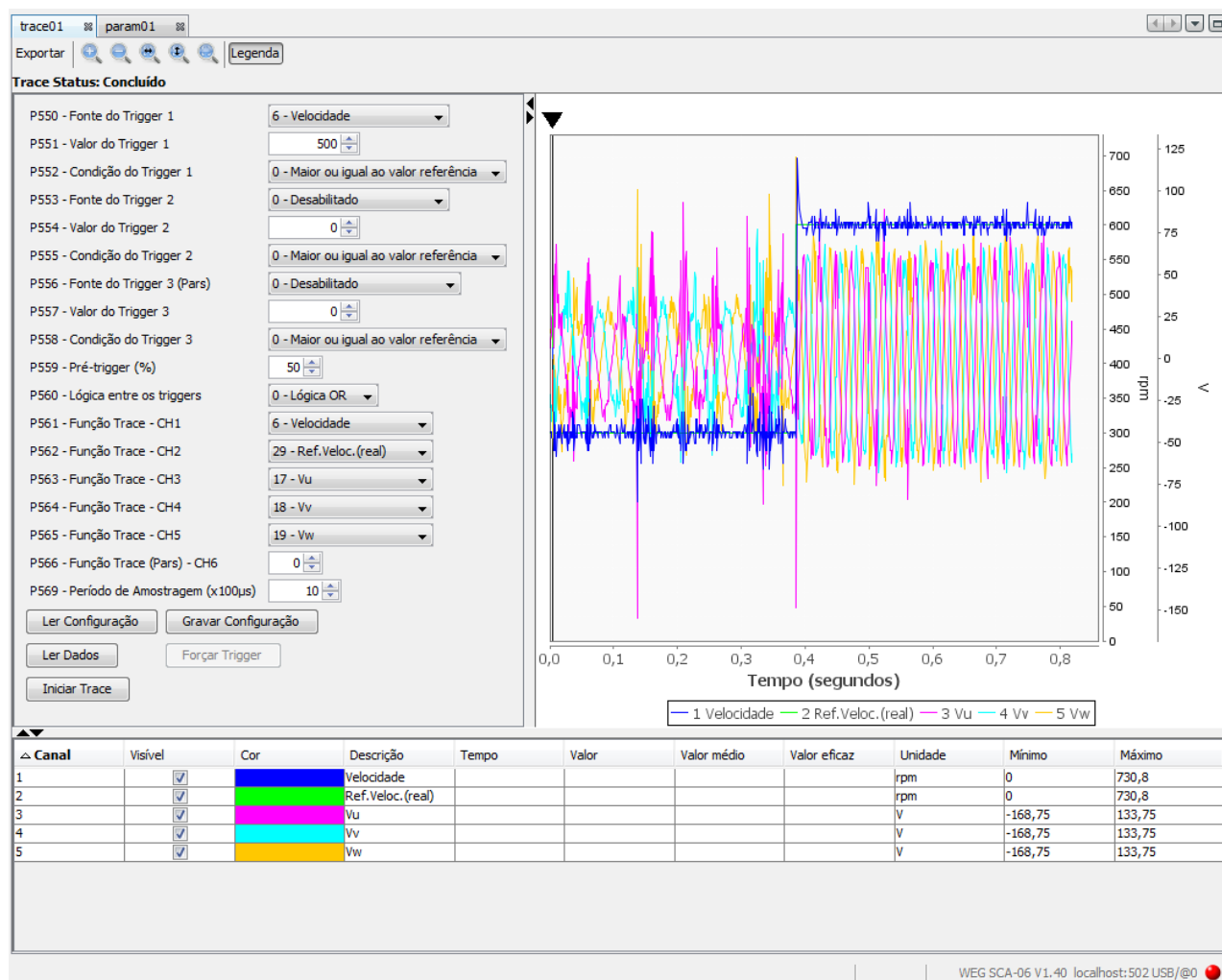
3. Depois de realizar as configurações desejadas basta clicar em gravar configuração para enviar estas ao equipamento. Observe que é necessário estar conectado ao equipamento com a opção **Conectar equipamento** do WPS.



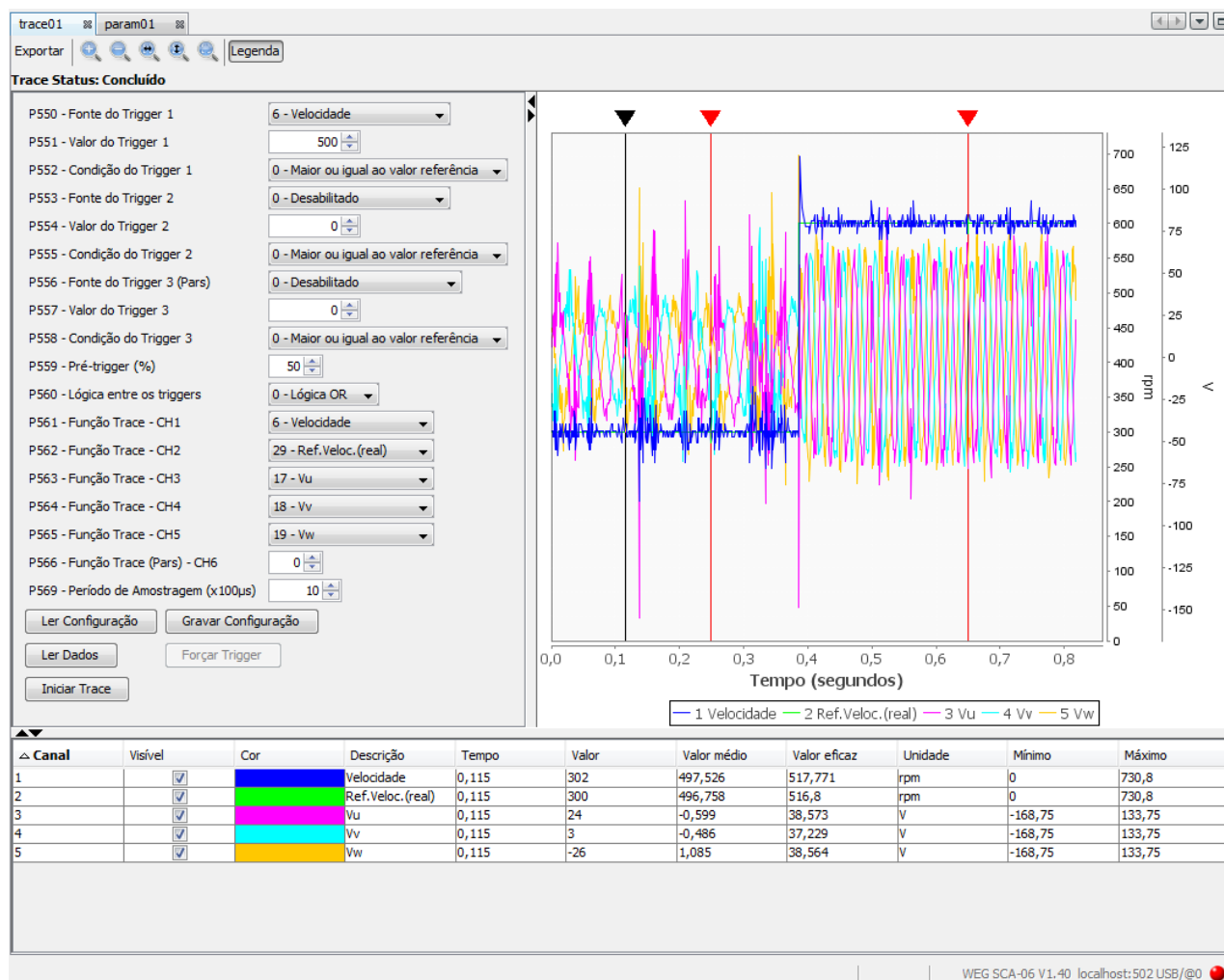
4. Depois das configurações gravadas, basta clicar em **Iniciar Trace**. Observe que o estado da função trace mudou para **Esperando** ou seja a ferramenta agora está aguardando a execução da trigger para montar o gráfico e exibir os valores do trace.



5. Após a trigger ocorrer, o gráfico e os valores serão exibidos na tabela e o estado da função trace será **Concluído**.



6. Se desejar você pode clicar com o mouse no curso preto do gráfico e adicionar cursores fixos para que o cálculo dos valores médios e eficazes sejam realizados para os canais nas faixas definidas.



10.2.9 Comunicação

Forçar E/S

10.2.9.1 Forçar E/S

Visão Geral

A janela de forçar entradas e saídas é utilizada para que os valores das entradas digitais e analógicas sejam lidos pelo programa, por valores manipulados pelo usuário, independente dos seus estados físicos. Também permite a manipulação dos estados físicos das saídas digitais e analógicas pelo usuário, de forma independente aos valores calculados pelo programa.

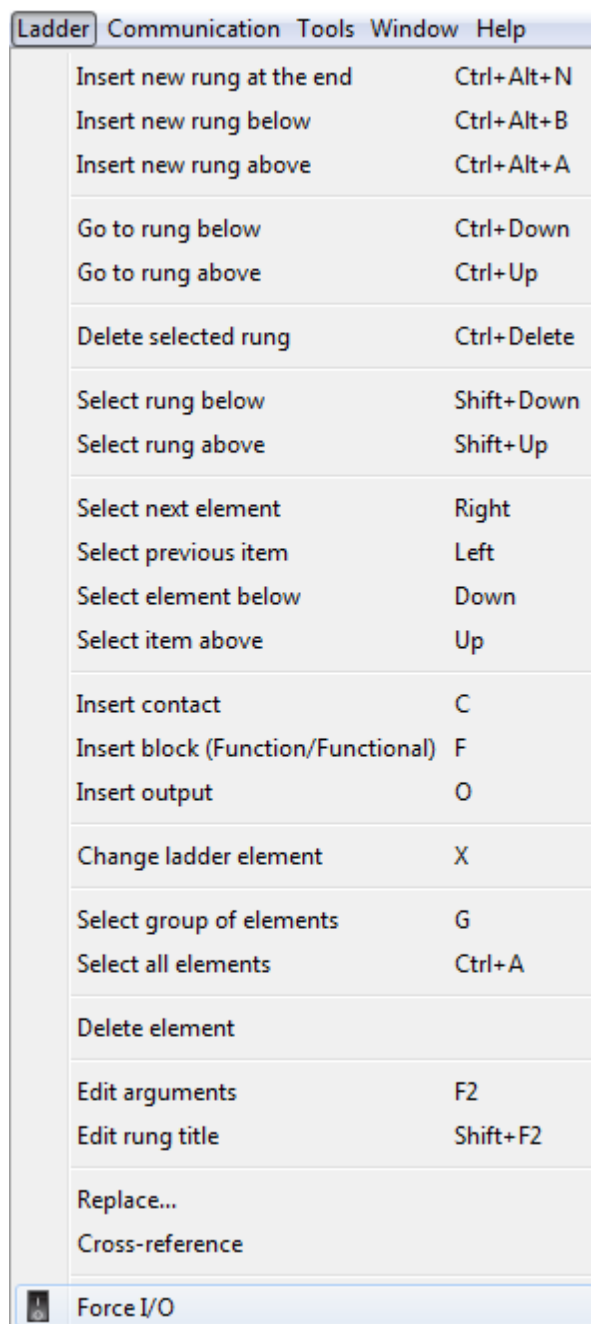
Para forçar as entradas e saídas do equipamento, é necessário que a monitoração on-line esteja ativa e a opção de **Rodar Ciclicamente** esteja habilitada. Os dados são enviados ao equipamento a cada 2 segundos.

Os valores podem ser editados sem que o equipamento esteja conectado. As configurações ficam armazenadas no recurso e são gravadas sempre que a seleção de recurso principal é alterada.

Os dados apresentados na janela de forçar E/S contêm os valores pertencentes ao recurso (e configuração)

selecionado como principal.

A janela de forçar E/S é aberta através do menu **Ladder > Forçar E/S**:



Barra de ferramentas

A barra de ferramentas da janela de forçar possui as opções de rodar ciclicamente, *upload* da configuração de forçar do equipamento, habilitar todos e desabilitar todos:

 **Rodar ciclicamente:** Envia os as configurações do usuário para o equipamento e atualiza o estado das

entradas e saídas de maneira cíclica.

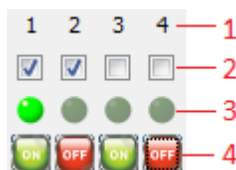
 **Upload da configuração:** Permite que a configuração atual do equipamento seja lida. Para esta opção estar habilitada, é necessário que a monitoração on-line esteja ativa e a opção rodar ciclicamente desativada.

 **Habilitar todos:** Habilita o forçar E/S de todas as entradas e saídas do equipamento.

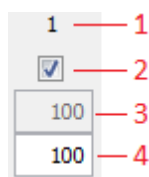
 **Desabilitar todos:** Desabilita o forçar E/S de todas as entradas e saídas do equipamento.

Comandos das entradas e saídas

Para cada entrada e saída digital e analógica estão associados uma caixa de seleção para habilitar o forçar, um campo de status e um campo para edição.



1. Número da entrada/saída digital
2. Habilita/desabilita o forçar E/S
3. Estado atual da E/S: Possui três estados: 1. led verde claro: acionado; 2. led verde escuro: desacionado; 3. led cinza: o valor não está sendo lido.
4. Habilita/desabilita a entrada/saída



1. Número da entrada/saída analógica
2. Habilita/desabilita o forçar E/S
3. Valor atual da entrada/saída
4. Valor da entrada/saída configurada pelo usuário

10.3 CFW300

[Descrição](#)

[Marcadores de Sistema](#)

[Importar do WLP](#)

[Parâmetros](#)

[Ladder](#)

[Diagnóstico](#)

[Comunicação](#)

10.3.1 Descrição

O inversor de frequência CFW300 é um produto de alta performance que permite o controle de velocidade de motores de indução trifásicos. Este produto proporciona ao usuário as opções de controle vetorial (VVW) ou escalar (V/f), ambos programáveis de acordo com a aplicação.

O modo escalar (V/f) é recomendado para aplicações mais simples como o acionamento da maioria das bombas e ventiladores. Nesses casos é possível reduzir as perdas no motor e no inversor utilizando a opção "V/f Quadrática", o que resulta em economia de energia. O modo V/f também é utilizado quando mais de um motor é acionado por um inversor simultaneamente (aplicações multimotores). No modo vetorial (VVW) a operação é otimizada para o motor em uso, obtendo-se um melhor desempenho em termos de regulação de velocidade.

O inversor de frequência CFW300 também possui funções de CLP (Controlador Lógico Programável) através do recurso SoftPLC (integrado). Possui dois "slots" para conexão simultânea de acessórios: Slot 1 - Acessório de comunicação ou HMI externa e Slot 2 - Acessório de expansão de entradas e saídas (I/Os).

Consulte o manual do usuário do CFW300 para mais informações sobre o produto.

10.3.2 Marcadores de Sistema

As variáveis abaixo, contidas no grupo **GLOBAL_SYSTEM** da tabela de variáveis, possuem o tag fixo. O tag dos marcadores de sistema foram divididos em grupos e subgrupos sendo:

Grupos:

- CFW: variáveis de leitura e escrita do inversor de frequência CFW300.

Subgrupos:

- STS: variável de leitura (status);
- CMD: variável de escrita (comando).

Marcadores de Sistema de Leitura (Status)

Leitura - Função Modbus 02 "Read Discrete Inputs"

Endereço	Bit	Modbus	Tag	Descrição
Ladder				
%SB6000	0	0	SYS_FREQ_2HZ	Oscilador com frequência de 2 Hz
%SB6000	1	1	SYS_PULSE_1SCAN	Pulso durante o primeiro ciclo de scan
%SB6000	2	2	SYS_FALSE	Sempre em 0
%SB6000	3	3	SYS_TRUE	Sempre em 1
Estado Lógico				
%SB6002	5	21	CFW_STS_SEC_RAMP_ACTIVE	O inversor está configurado para usar os valores da primeira ou segunda rampa (0-Primeira, 1-Segunda)
%SB6002	6	22	CFW_STS_CONFIG_MODE	O inversor está em modo de configuração
%SB6002	7	23	CFW_STS_ALARM_ACTIVE	O inversor está com um alarme ativo
%SB6003	0	24	CFW_STS_MOTOR_RUNNING	O inversor está girando o motor na referência de velocidade, ou executando rampa de aceleração ou desaceleração
%SB6003	1	25	CFW_STS_GENERAL_ENABLED	Inversor está habilitado geral e pronto para girar motor
%SB6003	2	26	CFW_STS_FWD_REV_DIRECTION	O motor está girando no sentido reverso ou direto (0-Reverso, 1-Direto)
%SB6003	3	27	CFW_STS_JOG_ACTIVE	A função JOG está ativa
%SB6003	4	28	CFW_STS_LOC_REM_MODE	O inversor está em modo local ou remote (0-Local, 1-Remoto)
%SB6003	5	29	CFW_STS_UNDERVOLTAGE	O inversor está em subtensão
%SB6003	6	30	CFW_STS_RUNCOMMAND	O comando gira o motor está ativo no inversor
%SB6003	7	31	CFW_STS_FAULT_ACTIVE	O inversor está no estado de falha
%SB6004	0	32	CFW_STS_AI1_BROKEN_CABLE	Indica que o sinal da entrada analógica AI1 em 4 a 20 mA ou 20 a 4 mA está abaixo de 2 mA
%SB6004	1	33	CFW_STS_AI2_BROKEN_CABLE	Indica que o sinal da entrada analógica AI2 em 4 a 20 mA ou 20 a 4 mA está abaixo de 2 mA
Teclas da HMI				
%SB6006	0	48	CFW_STS_KEY_START_STOP	Tecla START/STOP (I)/(O) pressionada
%SB6006	2	50	CFW_STS_KEY_UP	Tecla UP pressionada
%SB6006	3	51	CFW_STS_KEY_DOWN	Tecla DOWN pressionada

Leitura - Função Modbus 04 "Read Input Registers"

Velocidade				
% SW6200	--	3100	CFW_STS_MOTOR_SPEED_13BITS	Velocidade do motor em 13 bits (8192)
% SW6202	--	3101	CFW_STS_MOTOR_SYNC_SPEED	Velocidade síncrona do motor em rpm
% SW6204	--	3102	CFW_STS_MOTOR_SPEED_RPM	Velocidade do motor em rpm
% SW6206	--	3103	CFW_STS_SPEED_REFERENCE	Referência de velocidade após a rampa em rpm
Alarme e Falha				
% SW6208	--	3104	CFW_STS_PRES_ALARM	Número do alarme que eventualmente está presente no inversor
% SW6210	--	3105	CFW_STS_PRES_FAULT	Número da falha que eventualmente está presente no inversor
Corrente e Torque				
% SW6212	--	3106	CFW_STS_RATED_CURRENT	Corrente nominal (HD) do inversor em A (x10)
% SW6214	--	3107	CFW_STS_MOTOR_CURRENT	Corrente do motor sem filtro em A (x10)

Marcadores de Sistema de Escrita / Leitura (Comando)

Leitura - Função Modbus 01 "Read Coils"

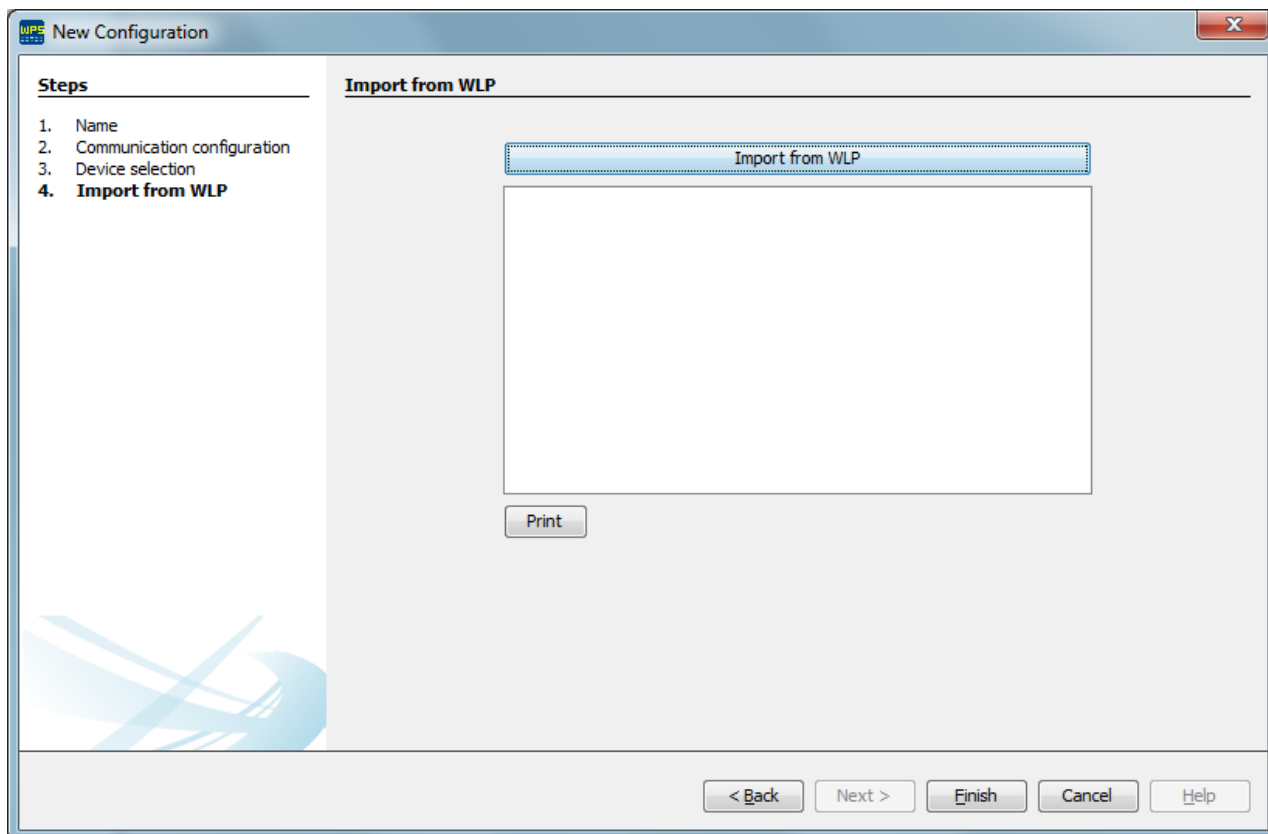
Escrita - Função Modbus 05 "Write Single Coil" e 15 "Write Multiple Coils"

Endereço	Bit	Modbus	Tag	Descrição
Comando Lógico				
%CB6008	0	16	CFW_CMD_RUN_STOP	Gira o motor de acordo com o valor da referência de velocidade (0-Para, 1-Gira)
%CB6008	1	17	CFW_CMD_GENERAL_ENABLE	Habilita geral o inversor permitindo a operação do motor (0-Desabilita, 1-Habilita)
%CB6008	2	18	CFW_CMD_SPEED_DIRECTION	O motor gira na direção indicada pela referência de velocidade (0-Reverso, 1-Direto)
%CB6008	3	19	CFW_CMD_JOG	Habilita a função JOG (0-Desabilita, 1-Habilita)
%CB6008	4	20	CFW_CMD_LOC_REM	Seleciona o modo de operação do inversor (0-Local, 1-Remoto)
%CB6008	5	21	CFW_CMD_SECOND_RAMP	Seleciona a rampa para acelerar e desacelerar o motor (0-Primeira, 1-Segunda)
%CB6008	6	22	CFW_CMD_FORCE_RUN_STOP_SPLC	Permite que o comando CFW_CMD_RUN_STOP da SoftPLC altere o comando Gira/Para do inversor independentemente da fonte programada para Gira/Para via P224 ou P227
%CB6008	7	23	CFW_CMD_FAULT_RESET	Executa o comando para reset da falha

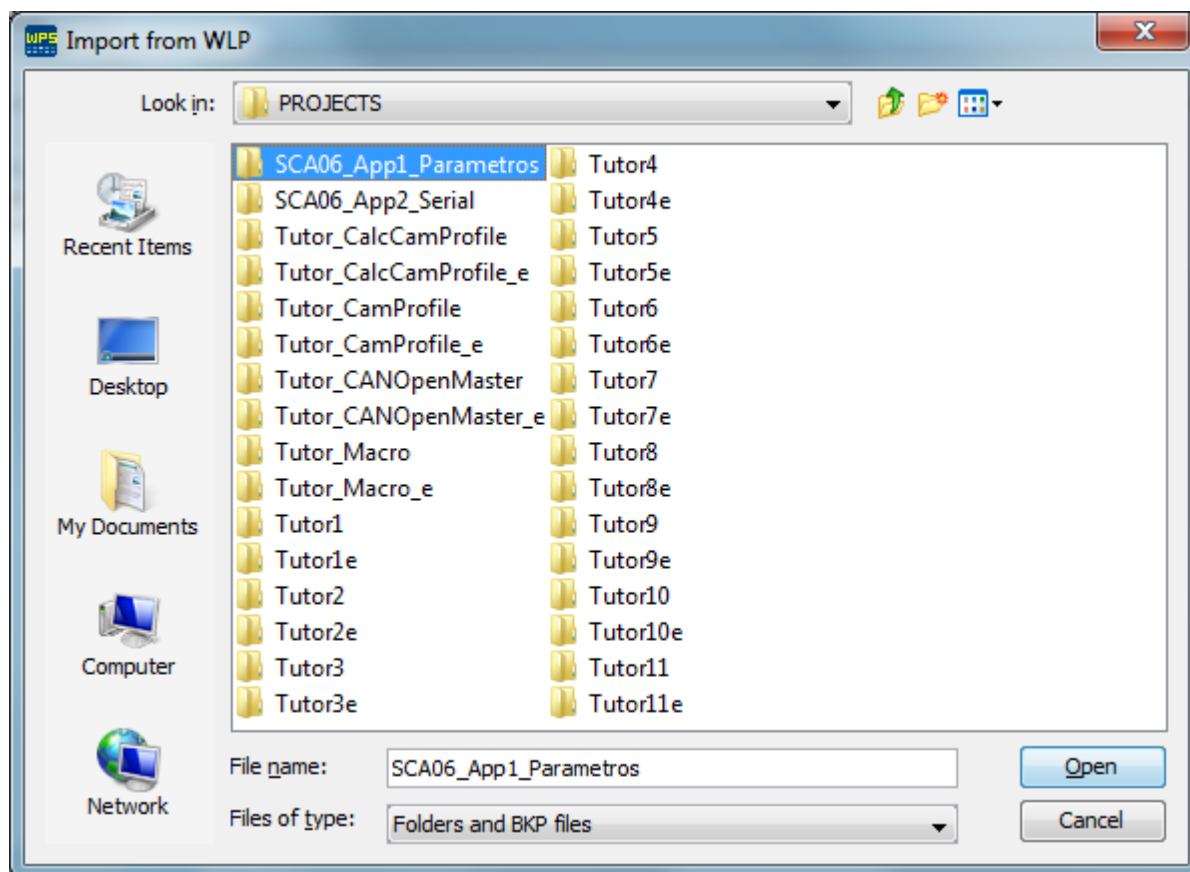
10.3.3 Importar do WLP

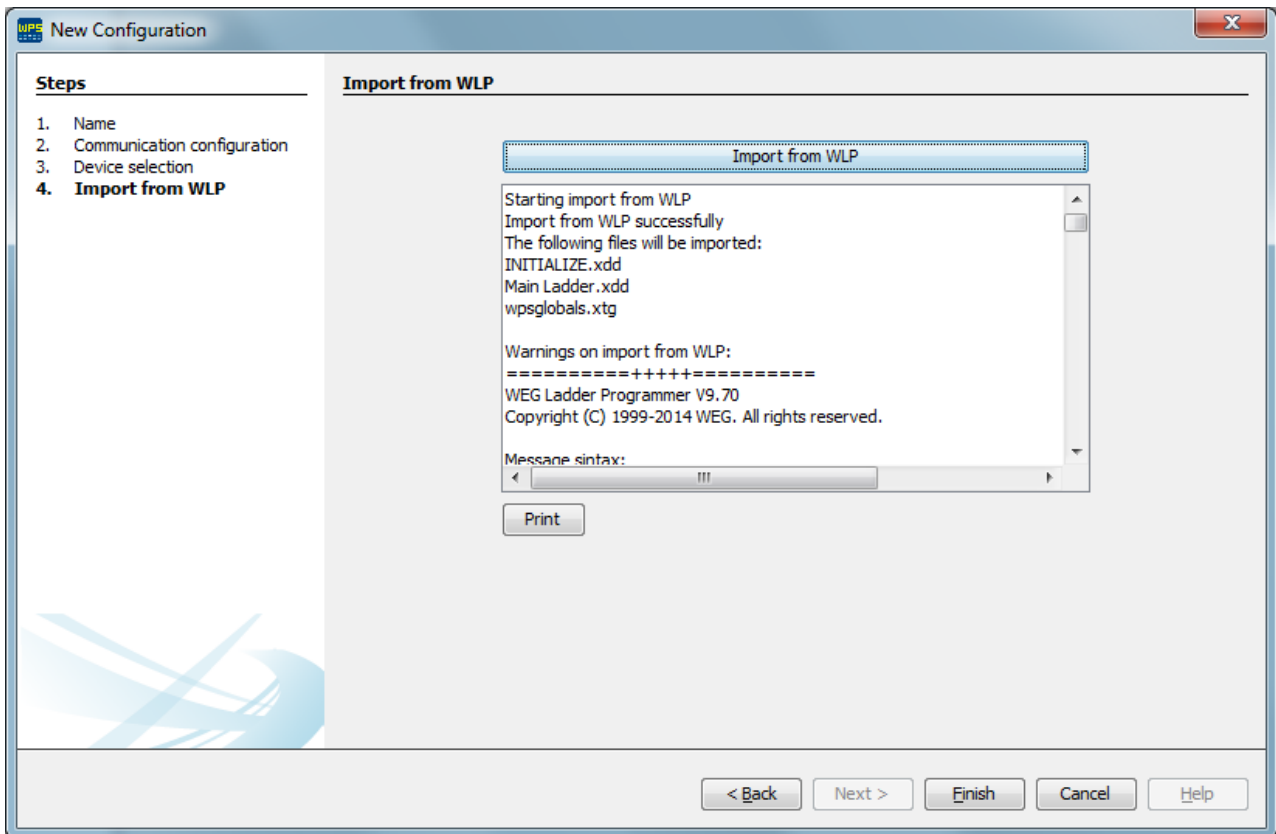
A função importar do WLP é utilizada para importar o Ladder desenvolvido no software WLP para o SCA06.

A importação do WLP deve ser executada durante a [criação do recurso](#).



1. Para executar a importação clique no botão Importar do WLP e selecione a pasta do projeto do WLP ou o arquivo BKP do WLP.





2. Após a importação finalizada com sucesso clique no botão Finalizar para copiar os arquivos importados para o novo recurso.

10.3.4 Parâmetros

[Visão Geral](#)

[Configuração](#)

[Leitura e Escrita de Parâmetros](#)

[Ocultar/Desocultar Parâmetros e Grupos de Parâmetros](#)

[Parâmetros do Usuário](#)

10.3.4.1 Visão Geral

A tela de configuração de parâmetros é utilizada para a configuração e monitoração de todos os parâmetros do equipamento, inclusive os parâmetros de usuário.



NOTA!

A leitura e escrita dos parâmetros é realizada através desta tela, somente a configuração dos parâmetros do usuário deve ser enviada a primeira vez ou sempre que modificada através da rotina de *download* do recurso.

Abaixo segue uma visão geral sobre a tela de configuração de parâmetros.

The screenshot shows the WEG SCA06 parameter configuration interface. It features a 'Projects' tree on the left, a 'Parameters' tree in the center, and a large table of parameters on the right. Red boxes and numbers 1 through 7 highlight specific features:

- 1. Projects tree
- 2. Parameters tree
- 3. Parameter groups
- 4. Action icons
- 5. Parameter table
- 6. Parameter table rows
- 7. Output window

Parameter	Description	User	Monitored	Minimum	Maximum	Factory setti...	Unit
P0099	Enable	0: Disable	0: Disable	0	2	0: Disable	
P0105	STOP Function Ramp	200	200	1	32767	200	ms/rpm
P0111	Rotation Direction	0: standard	0: standard	0	1	0: standard	
P0119	Current Reference	0.0	0.0	-3276.7	3276.7	0.0	A
P0121	Speed Reference	0	0	-9999	9999	0	rpm
P0126	Limit position Enable	0: Disabled	0: Disabled	0	1	0: Disabled	
P0127	Position lower limit - fractions of revolution	-16383	-16383	-16383	16383	-16383	
P0128	Position lower limit - number of revolutions	-32768	-32768	-32768	32767	-32768	
P0129	Position higher limit - fractions of revolution	16383	16383	-16383	16383	16383	
P0130	Position higher limit - number of revolutions	32767	32767	-32768	32767	32767	
P0131	Current lower limit	-140.0	-140.0	-140.0	0.0	-140.0	A
P0132	Current higher limit	140.0	140.0	0.0	140.0	140.0	A
P0133	Speed lower limit	-9999	-9999	-9999	0	-9999	rpm
P0134	Speed higher limit	9999	9999	0	9999	9999	rpm
P0136	Idynamic/Nominal	300	300	0	400	300	%
P0154	Breaking Resistor	0	0	0	1000	0	ohm
P0155	Breaking Resistor Average Power	200	200	0	10000	200	W
P0156	Breaking Resistor Maximum Energy	2200	2200	0	10000	2200	J
P0159	Kp Position Regulator	50	50	0	32767	50	
P0161	Kp PID Speed	4000	4000	0	32767	4000	
P0162	Ki PID Speed	50	50	0	32767	50	
P0163	Kd PID Speed	0	0	0	32767	0	
P0192	RTC Update	0	0	0	1	0	
P0193	Day of week	0	0	0	6	0	
P0194	Day	1	1	1	31	1	
P0195	Month	1	1	1	12	1	
P0196	Year	2011	2011	0	4095	2011	
P0197	Hours	0	0	0	23	0	
P0198	Minutes	0	0	0	59	0	
P0199	Seconds	0	0	0	59	0	
P0200	Password	1: Active	1: Active	0	2	1: Active	
P0202	Operation Mode	1: Torque Mode	2: Speed Mode	1	6	2: Speed Mode	
P0203	Flash backup Enable	1: Enable	1: Enable	0	1	1: Enable	
P0204	Load/Save Parameters	0: Disable	0: Disable	0	13	0: Disable	
P0209	Position/speed feedback source	0: internal motor sensor	0: internal motor sensor	0	3	0: internal mo...	
P0210	External feedback gear ratio:Numerator	1	1	1	32767	1	
P0211	External feedback gear ratio:Denominator	1	1	1	32767	1	
P0213	External feedback Rotation Direction	0: direct	0: direct	0	1	0: direct	
P0214	External feedback loss protection	1000	1000	0	9999	1000	

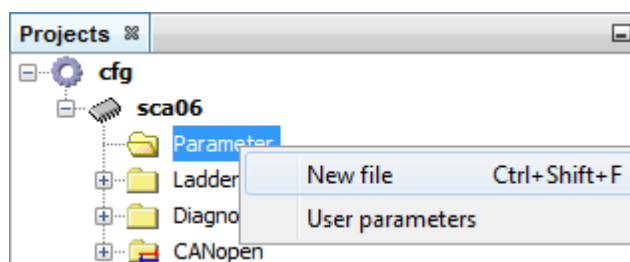
- Arquivos de parâmetros:** Nesta parte ficam todos os arquivos de configurações de parâmetros criados pelo usuário. Observar que quando o arquivo tiver um boneco desenhado sobre a tabela, significa que esta tabela de parâmetros possui parâmetros/grupo de parâmetros ocultos.
- Grupos de parâmetros:** Nesta árvore são mostrados todos os grupos de parâmetros, observar que um mesmo parâmetro pode encontrar-se em mais de um grupo, porém quando seu valor for modificado este será modificado em todos os grupos que pertencer.
- Grupos de parâmetros modificados:** Grupos de parâmetros que contenham o boneco sobre a tabela significam que estes possuem parâmetros ocultos.
- Comandos:** Segue abaixo descrição dos comandos por ordem de exibição:
 - Desocultar parâmetro: Caso algum parâmetro tenha sido oculto este botão permite torná-los visíveis novamente.
 - Ocultar parâmetro: Basta selecionar um ou mais parâmetros na tabela e disparar esta ação para que os parâmetros sejam ocultos.
 - Gravar tabela: Grava os valores dos parâmetros exibidos na tela no equipamento, os valores enviados são os presentes na coluna Usuário. O fluxo é Usuário -> Monitorado (equipamento)
 - Ler tabela: Lê os valores dos parâmetros do equipamento exibidos na coluna Monitorado e grava no arquivo de parâmetros na coluna Usuário. O fluxo é Monitorado (equipamento) -> Usuário
 - Parâmetros do usuário: Abre uma tela para edição dos parâmetros do usuário.
 - Filtro: Abre uma opção de filtro de parâmetros, podendo filtrar por número do parâmetro ou descrição.

5. **Parâmetros do Usuário e Parâmetros Monitorado:** Nestas duas colunas ocorre a visualização off-line e on-line dos parâmetros pode-se assim dizer, a coluna **Usuário** mostra os valores que estão no arquivo localizado no computador e a coluna **Monitorado** mostra os valores que estão efetivamente gravados dentro do equipamento. Sempre que você usar a opção de **Gravar Parâmetro**, os valores enviados serão da coluna **Usuário** para a coluna **Monitorado**, ou seja, do Arquivo para o Equipamento, no caso da leitura o fluxo se inverte sendo da coluna **Monitorado** para **Usuário**, ou seja, do Equipamento para o Arquivo. Caso deseja alterar os valores diretamente no equipamento sem alterar no arquivo, basta clicar sobre a coluna monitorado e alterar os valores, que a alteração ocorrerá de forma on-line.
6. **Parâmetros modificados:** Sempre que um valor de um parâmetro estiver diferente entre as colunas **Usuário** e **Monitorado**, este ficará em vermelho.
7. **Saída:** Nesta tela são exibidas informações de erro, caso estes ocorram durante a escrita e ou leitura dos parâmetros.

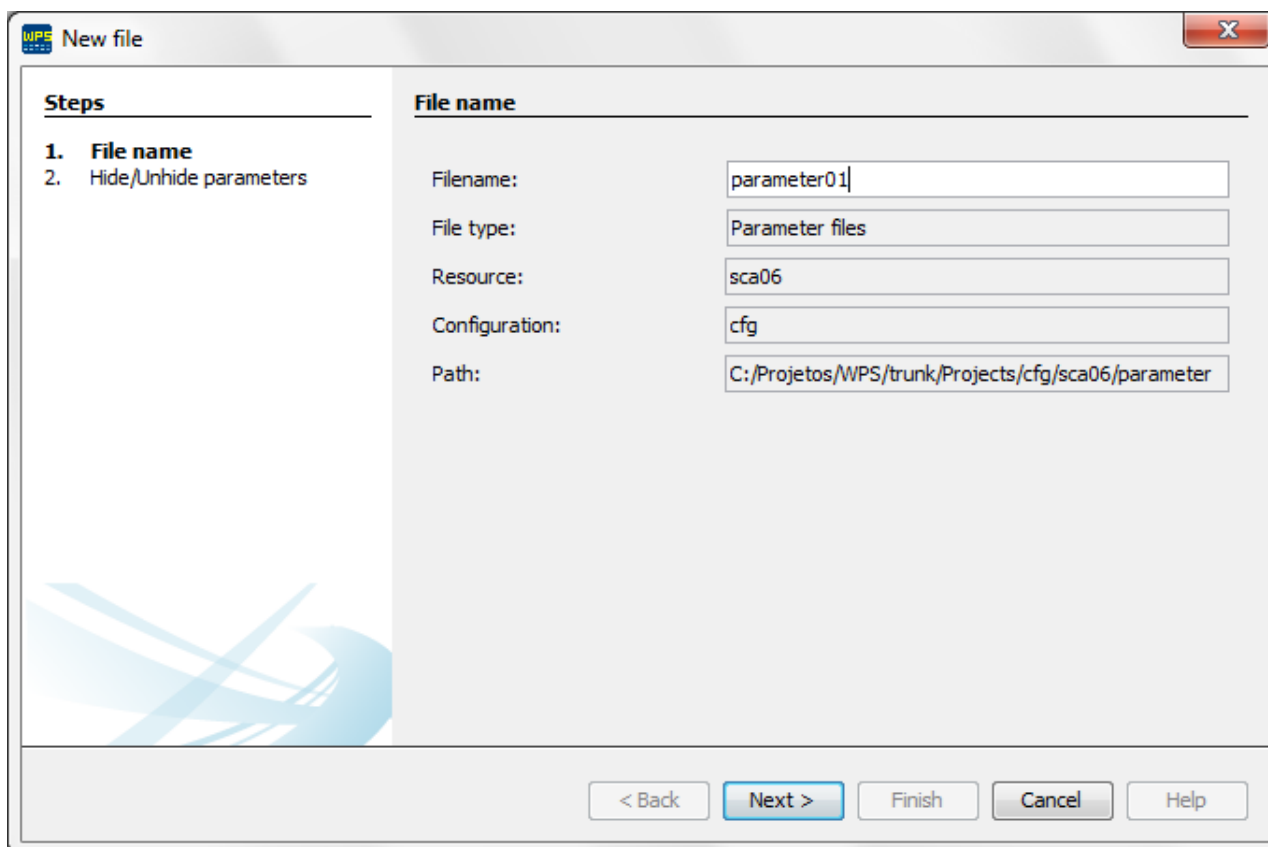
10.3.4.2 Configuração

Abaixo segue a lista dos passos necessários para criação de um arquivo de parâmetros.

1. Criação de um novo arquivo de parâmetros.



2. Definir nome para o arquivo de parâmetros

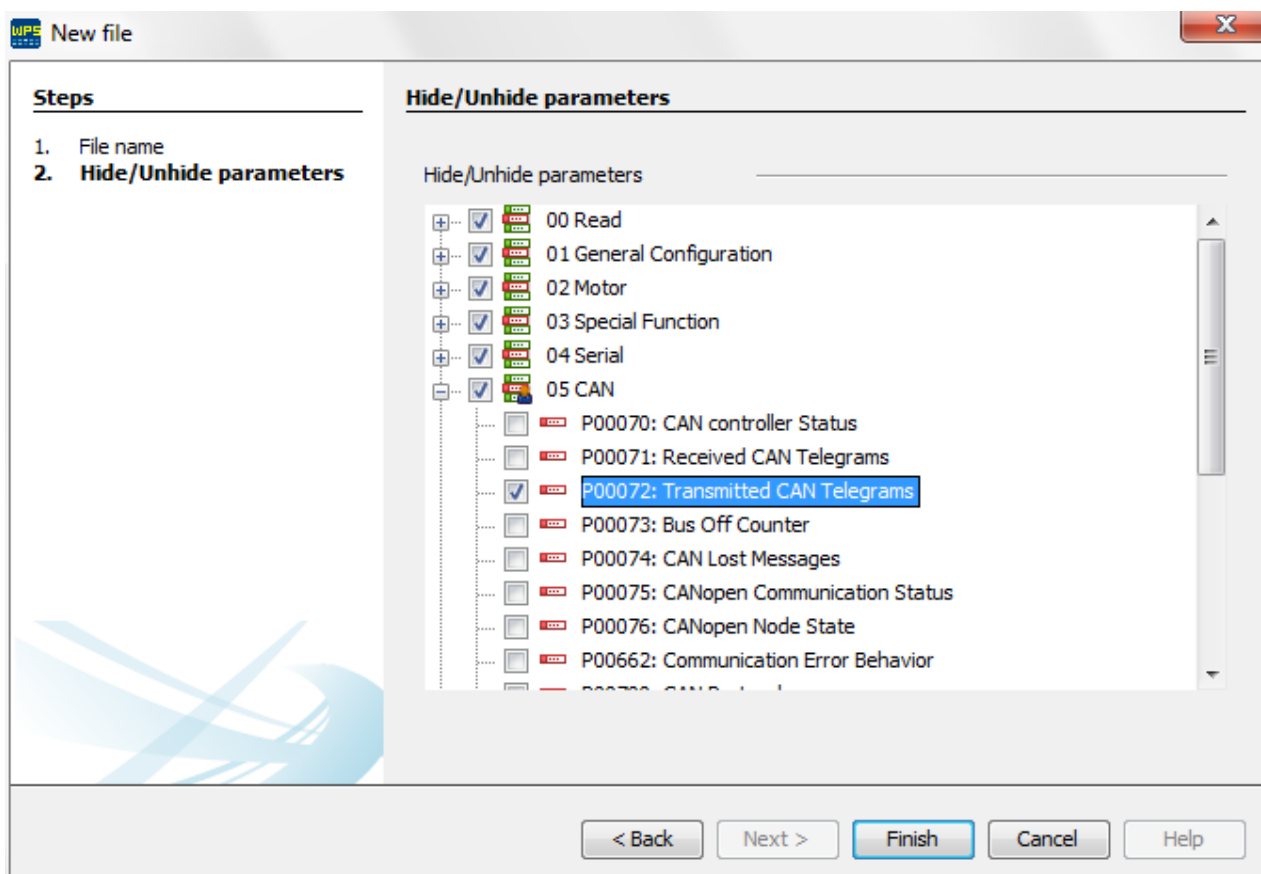


The image shows a 'New file' dialog box from the WPS software. It has a title bar with the WPS logo and a close button. The dialog is divided into two main sections: 'Steps' on the left and 'File name' on the right. The 'Steps' section lists two steps: '1. File name' and '2. Hide/Unhide parameters'. The 'File name' section contains five input fields: 'Filename:' with the text 'parameter01', 'File type:' with 'Parameter files', 'Resource:' with 'sca06', 'Configuration:' with 'cfg', and 'Path:' with 'C:/Projetos/WPS/trunk/Projects/cfg/sca06/parameter'. At the bottom of the dialog, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'. The 'Next >' button is highlighted with a blue border.

Steps	File name
1. File name	Filename: parameter01
2. Hide/Unhide parameters	File type: Parameter files
	Resource: sca06
	Configuration: cfg
	Path: C:/Projetos/WPS/trunk/Projects/cfg/sca06/parameter

< Back Next > Finish Cancel Help

3. Configurar quais parâmetros você deseja visualizar em sua tabela de parâmetros



4. Após a realização das etapas acima o arquivo de parâmetros estará criado e a parametrização do equipamento já poderá ser feita.

parameter01 38

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	...	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0		-9999	9999	0	rpm
P0003	Motor Current	0.0		-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	0		0	999	0	V
P0006	Drive Status	0: Ready		0	5	0: Ready	
P0008	DI3 to DI1 Status	0		0	7	0	
P0009	DI101 to DI106 Status	0		0	63	0	
P0010	DI107 to DI112 Status	0		0	63	0	
P0011	DI201 to DI206 Status	0		0	63	0	
P0012	DI207 to DI212 Status	0		0	63	0	
P0013	DI301 to DI306 Status	0		0	63	0	
P0014	DI307 to DI312 Status	0		0	63	0	
P0015	DO1 Status	0		0	1	0	
P0016	DO101 to DO106 Status	0		0	63	0	
P0017	DO201 to DO206 Status	0		0	63	0	
P0018	DO301 to DO306 Status	0		0	63	0	
P0021	Internal Air Temperature	0		0	1000	0	°C
P0022	Dissipator Temperature	0		0	1000	0	°C
P0023	Software Version	9.99		0.00	655.35	9.99	
P0024	Bootloader Version	0.00		0.00	655.35	0.00	
P0025	FPGA Project Version	0.00		0.00	655.35	0.00	
P0030	Present Alarm	0		0	2000	0	
P0031	Last Alarm	0		0	2000	0	
P0032	Last alarm Day.Month	0.00		0.00	31.12	0.00	
P0033	Last alarm year	0		0	4096	0	
P0034	Last alarm Hour.Min	0.00		0.00	23.59	0.00	
P0035	Present Fault	0		0	2000	0	
P0036	Last Fault	0		0	2000	0	
P0037	Last Fault Day.Month	0.00		0.00	31.12	0.00	
P0038	Last fault year	0		0	4096	0	
P0039	Last fault Hour.Min	0.00		0.00	23.59	0.00	
P0040	Second fault	0		0	2000	0	
P0041	Second fault Day.Month	0.00		0.00	31.12	0.00	
P0042	Second fault year	0		0	4096	0	
P0043	Second fault Hour.Min	0.00		0.00	23.59	0.00	
P0044	Third fault	0		0	2000	0	
P0045	Third fault Day.Month	0.00		0.00	31.12	0.00	
P0046	Third fault year	0		0	4096	0	
P0047	Third fault Hour.Min	0.00		0.00	23.59	0.00	
P0048	Lag fault	0		0	65535	0	
P0050	Real Axis: Actual Position	0		0	16383	0	
P0052	fractions of revolution	0		-16383	16383	0	
P0053	number of revolutions	0		-32768	32767	0	
P0056	Standart Counter - Low	0		0	65535	0	
P0057	Standart Counter - High	0		0	65535	0	
P0058	Counter 1 - Low	0		0	65535	0	
P0059	Counter 1 - High	0		0	65535	0	

10.3.4.3 Leitura e Escrita de Parâmetros

Existem três formas de se realizar a leitura e escrita dos parâmetros, sendo estas por tabela, seleção e grupo.

1. **Escrita da tabela.** O comando de escrever a tabela irá enviar todos os parâmetros visíveis na tela ao equipamento, caso ocorra algum erro no envio de algum parâmetro específico será exibida uma mensagem na janela de saída informando o erro. É importante observar que somente os parâmetros visíveis serão enviados, então é necessária atenção a qual nó da árvore de grupos de parâmetros você está visualizando. Exemplo: Se deseja escrever todos sem filtrar por grupo basta deixar selecionada a raiz da árvore.

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	M...	Minimum	Maxim...	Factory settings	Unit
P0002	Motor speed	0	0	-9999	9999	0	rpm
P0003	Motor Current	0.0	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	0	308	0	999	0	V
P0006	Drive Status	0: Ready	0: ...	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 Status	0	0	0	63	0	
P0012	DI207 to DI212 Status	0	0	0	63	0	
P0013	DI301 to DI306 Status	0	0	0	63	0	
P0014	DI307 to DI312 Status	0	0	0	63	0	
P0015	DO1 Status	0	0	0	1	0	
P0016	DO101 to DO106 Status	0	0	0	63	0	
P0017	DO201 to DO206 Status	0	0	0	63	0	
P0018	DO301 to DO306 Status	0	0	0	63	0	
P0021	Internal Air Temperature	0	41	0	1000	0	°C
P0022	Dissipator Temperature	0	33	0	1000	0	°C
P0023	Software Version	9.99	1.40	0.00	655.35	9.99	
P0024	Bootloader Version	0.00	20.03	0.00	655.35	0.00	
P0025	FPGA Project Version	0.00	0.03	0.00	655.35	0.00	
P0030	Present Alarm	0	0	0	2000	0	
P0031	Last Alarm	0	826	0	2000	0	
P0032	Last alarm Day.Month	0.00	3.01	0.00	31.12	0.00	
P0033	Last alarm year	0	1586	0	4096	0	
P0034	Last alarm Hour.Min	0.00	17.01	0.00	23.59	0.00	
P0035	Present Fault	0	0	0	2000	0	
P0036	Last Fault	0	2	0	2000	0	
P0037	Last Fault Day.Month	0.00	3.01	0.00	31.12	0.00	
P0038	Last fault year	0	1594	0	4096	0	
P0039	Last fault Hour.Min	0.00	17.13	0.00	23.59	0.00	

Output - Default output

*** Writing parameter ***

P1035 31% WEG SCA-06 V1.40 localhost:502 USB/@0

2. **Leitura da tabela.** O comando de ler a tabela irá ler todos os parâmetros do equipamento, caso ocorra algum erro na leitura de algum parâmetro específico será exibida uma mensagem na janela de saída informando o erro. É importante observar que somente os parâmetros visíveis serão lidos, então é necessária atenção a qual nó da árvore de grupos de parâmetros você está visualizado. Exemplo: Se deseja ler todos sem filtrar por grupo basta deixar selecionada a raiz da árvore.

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	...	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0	0	-9999	9999	0	rpm
P0003	Motor Current	-0.1	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	308	308	0	999	0	V
P0006	Drive Status	0: Ready	0: ...	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 Status	0	0	0	63	0	
P0012	DI207 to DI212 Status	0	0	0	63	0	
P0013	DI301 to DI306 Status	0	0	0	63	0	
P0014	DI307 to DI312 Status	0	0	0	63	0	
P0015	DO1 Status	0	0	0	1	0	
P0016	DO101 to DO106 Status	0	0	0	63	0	
P0017	DO201 to DO206 Status	0	0	0	63	0	
P0018	DO301 to DO306 Status	0	0	0	63	0	
P0021	Internal Air Temperature	41	41	0	1000	0	°C
P0022	Dissipator Temperature	33	33	0	1000	0	°C
P0023	Software Version	1.40	1.40	0.00	655.35	9.99	
P0024	Bootloader Version	20.03	20...	0.00	655.35	0.00	
P0025	FPGA Project Version	0.03	0.03	0.00	655.35	0.00	
P0030	Present Alarm	0	0	0	2000	0	
P0031	Last Alarm	826	826	0	2000	0	
P0032	Last alarm Day.Month	3.01	3.01	0.00	31.12	0.00	
P0033	Last alarm year	1586	1586	0	4096	0	
P0034	Last alarm Hour.Min	17.01	17...	0.00	23.59	0.00	
P0035	Present Fault	0	0	0	2000	0	
P0036	Last Fault	2	2	0	2000	0	
P0037	Last Fault Day.Month	3.01	3.01	0.00	31.12	0.00	
P0038	Last fault year	1594	1594	0	4096	0	
P0039	Last fault Hour.Min	17.13	17...	0.00	23.59	0.00	
P0040	Second fault	2	2	0	2000	0	
P0041	Second fault Day.Month	3.01	3.01	0.00	31.12	0.00	
P0042	Second fault year	1594	1594	0	4096	0	
P0043	Second fault Hour.Min	17.12	17...	0.00	23.59	0.00	
P0044	Third fault	32	32	0	2000	0	
P0045	Third fault Day.Month	1.01	1.01	0.00	31.12	0.00	

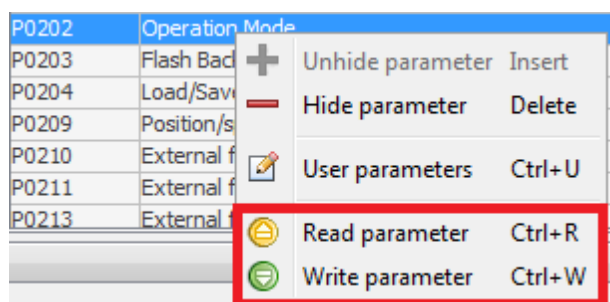
Output - Default output

*** Reading parameter ***

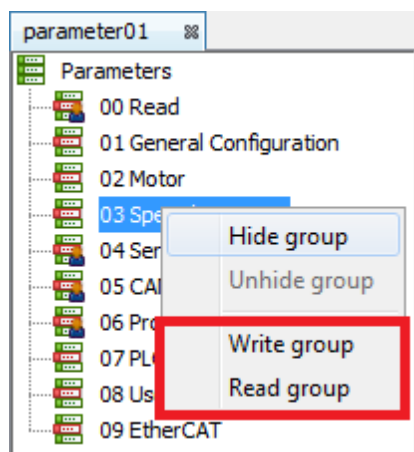
P0918 43%

WEG SCA-06 V1.40 localhost:502 USB/@0

3. **Leitura/Escrita de parâmetros específicos.** Para ler/escrever somente alguns parâmetros um ou mais, basta selecioná-los na tabela e clicar com o botão direito do mouse escolhendo a opção desejada, sendo estas ler ou escrever parâmetro.



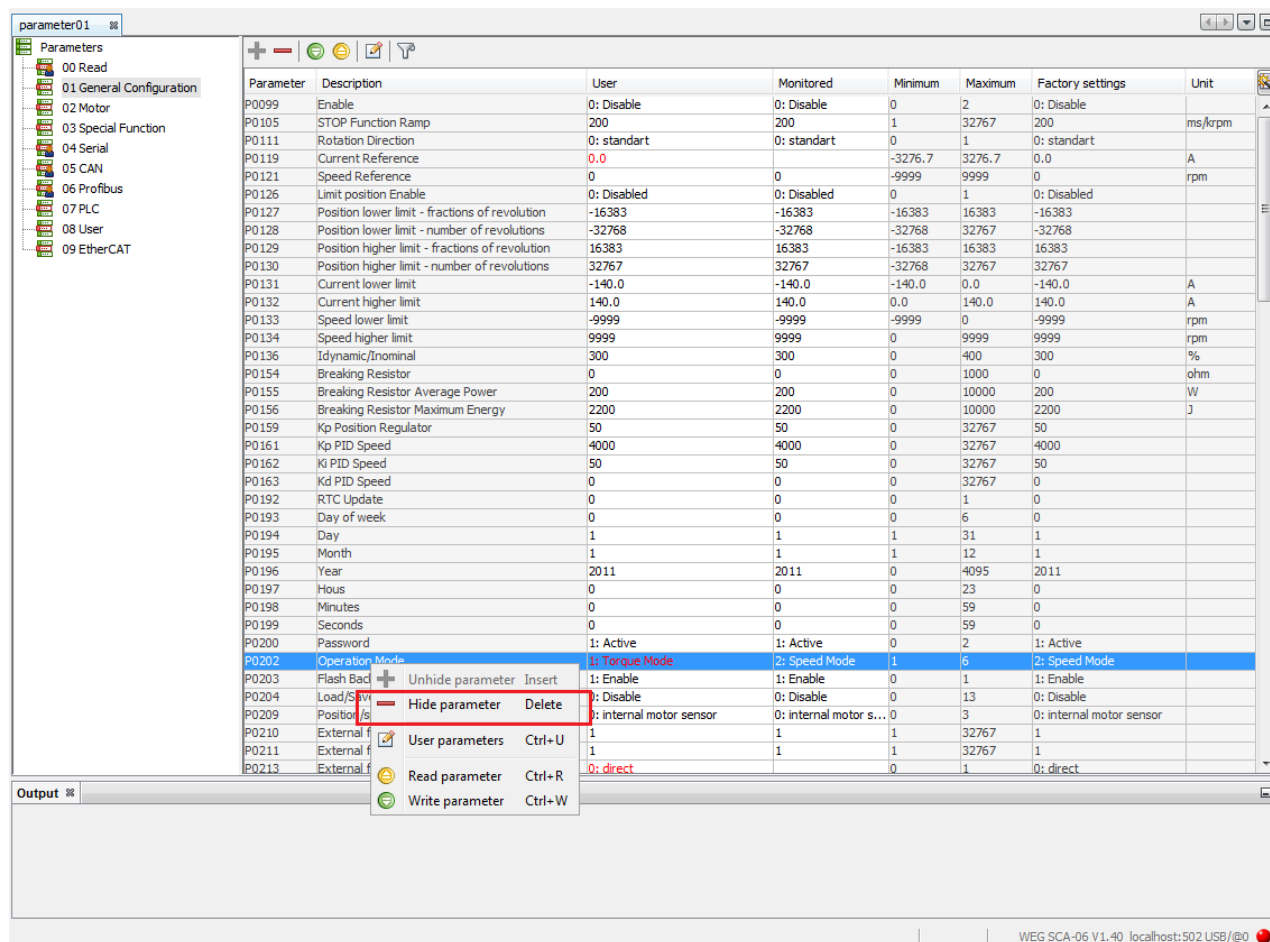
4. **Leitura/escrita de grupo de parâmetros.** Para ler/escrever somente um grupo de parâmetros, basta selecioná-los na árvore de grupos e clicar com o botão direito do mouse escolhendo a opção desejada, sendo estas ler ou escrever grupo.



10.3.4.4 Ocultar/Desocultar Parâmetros e Grupos de Parâmetros

Os parâmetros podem ser ocultos/desocultos de duas formas, sendo estas de forma individual ou por grupo.

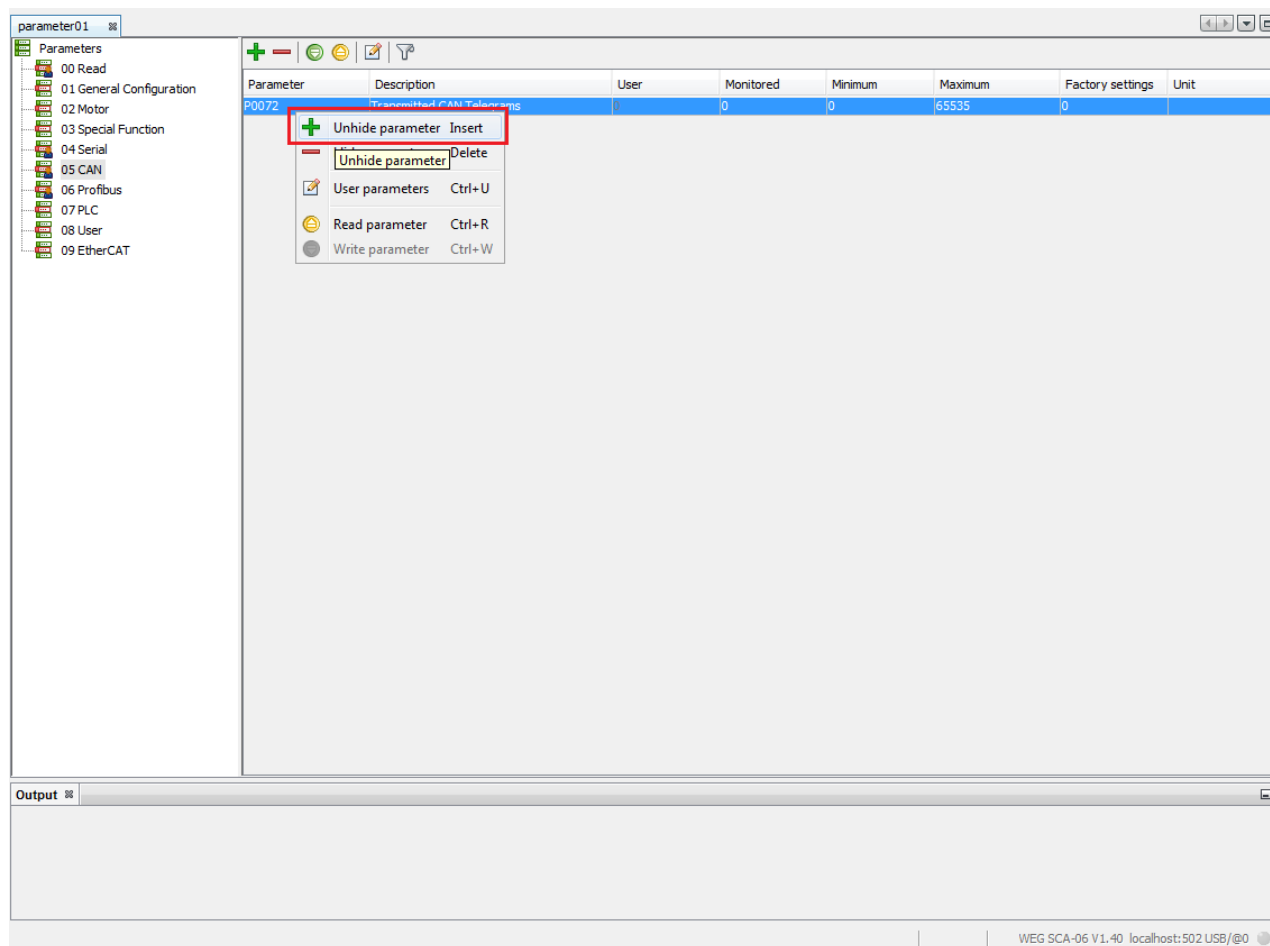
1. **Ocultar Parâmetros.** Para ocultar um parâmetro individualmente basta clicar sobre o parâmetro desejado com o botão direito do mouse e selecionar a opção **Ocultar Parâmetro**, se preferir também pode apenas pressionar a tecla **Delete**.



2. **Desocultar Parâmetros.** Para exibir os parâmetros que foram ocultos clique com o botão direito do mouse e escolha a opção **Desocultar parâmetros** ou pressione a tecla **Insert**, após isto uma nova janela se abrirá exibindo os parâmetros que estão ocultos então basta selecionar os desejados e confirmar.

**NOTA!**

Os parâmetros exibidos nesta nova janela são somente os que pertencem ao filtro atual conforme seleção na árvore de grupos de parâmetros, nas imagens abaixo o grupo CAN está selecionado isto significa que somente os parâmetros ocultos deste grupo serão exibidos.



parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams		0	0	65535	0	

Select parameters

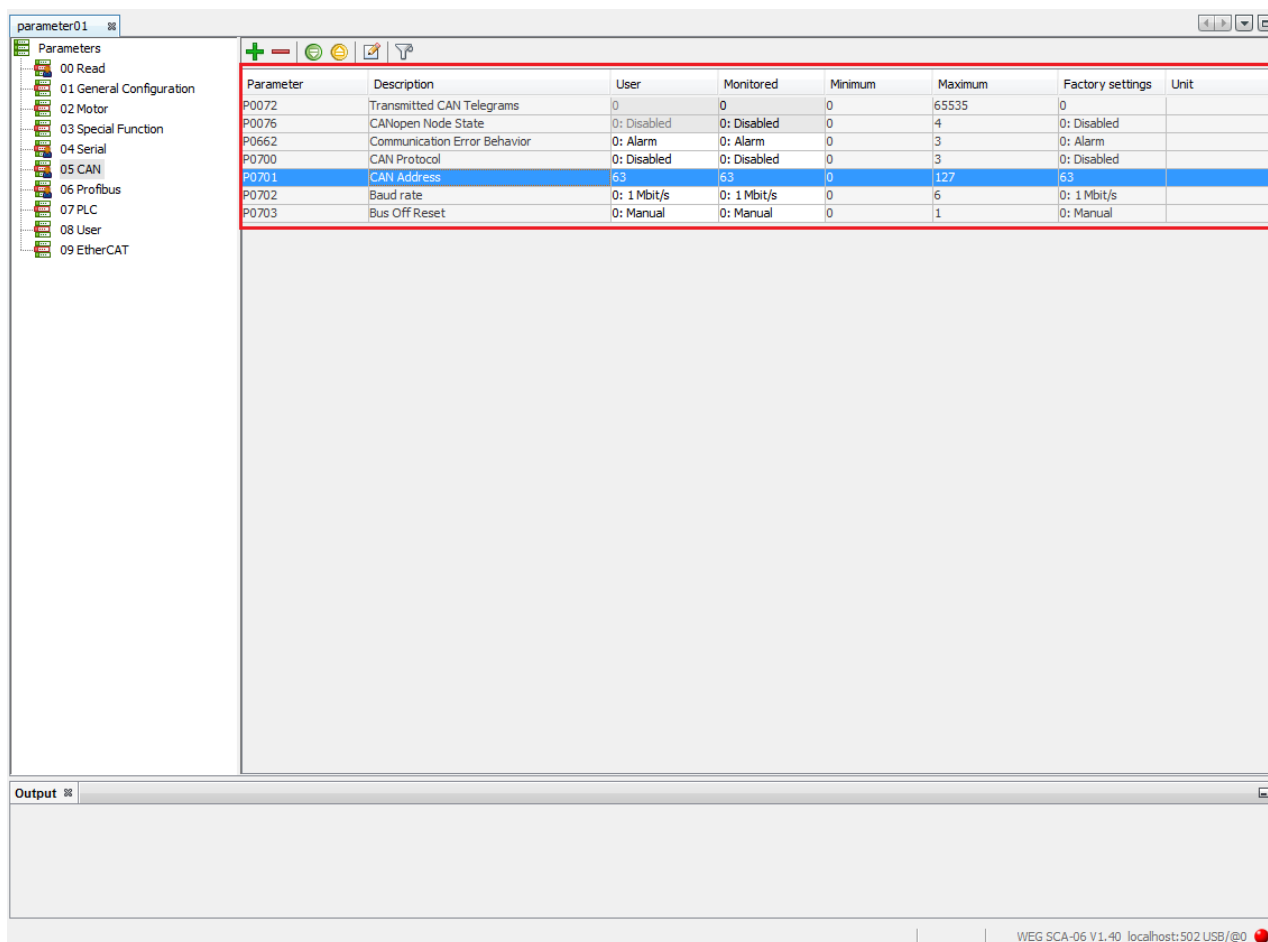
Select to unhide:

- CAN controller Status
- Received CAN Telegrams
- Bus Off Counter
- CAN Lost Messages
- CANopen Communication Status
- CANopen Node State
- Communication Error Behavior
- CAN Protocol
- CAN Address
- Baud rate
- Bus Off Reset
- Follow Type
- Follow COB ID
- Follow Period

OK Cancel

Output

WEG SCA-06 V1.40 localhost:502.USB/@0



Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams	0	0	0	65535	0	
P0075	CANopen Node State	0: Disabled	0: Disabled	0	4	0: Disabled	
P0662	Communication Error Behavior	0: Alarm	0: Alarm	0	3	0: Alarm	
P0700	CAN Protocol	0: Disabled	0: Disabled	0	3	0: Disabled	
P0701	CAN Address	63	63	0	127	63	
P0702	Baud rate	0: 1 Mbit/s	0: 1 Mbit/s	0	6	0: 1 Mbit/s	
P0703	Bus Off Reset	0: Manual	0: Manual	0	1	0: Manual	

3. **Ocultar Grupos de Parâmetros.** Para ocultar um grupo de parâmetros basta selecionar o grupo desejado na árvore e usar a opção **Ocultar Grupo**.

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Sp **Hide group**
- 04 Set
- 05 CA Unhide group
- 06 Pro
- 07 PLC
- 08 Use
- 09 EtherCAT

Write group

Read group

Param...	Description	User	Mo...	Minimum	Maximum	Factory settings	Unit
P0365	Encoder temperature	0.0		-3276.7	3276.7	0.0	
P0368	Manufacture password	0	0	0	65535	0	
P0369	Electrical position	0	50828	0	65535	0	
P0370	Encoder temperature	0		-3276.7	3276.7	0	
P0371	Read Encoder filter	3000	3000	3000	37500	3000	
P0372	Encoder filter	3000	3000	3000	37500	3000	
P0375	Encoder Offset	0	0	-3276.7	3276.7	0	
P0490	Load Absolute position	0	0	0	1	0	
P0492	User reference: fractions of revolution	0	0	-16383	16383	0	
P0493	User reference: number of revolutions	0	0	-32768	32767	0	
P0500	Count Mode: Standart Counter	0: Disabled	0: Di...	0	4	0: Disabled	
P0502	Load counter	0	0	0	1	0	
P0503	Counter Value - Low	0	0	0	65535	0	
P0504	Counter Value - High	0	0	0	65535	0	
P0506	Counter number of pulses/turns	1024	1024	1	65535	1024	
P0507	Counter speed filter	500	500	1	4000	500	Hz
P0510	Count Mode: Counter 1	0: Disabled	0: Di...	0	3	0: Disabled	
P0511	Null Pulse 1 options	0: Disabled	0: Di...	0	5	0: Disabled	
P0512	Load Counter 1 value	0	0	0	1	0	
P0513	Counter 1 value - Low	0	0	0	65535	0	
P0514	Counter 1 value - High	0	0	0	65535	0	
P0516	Counter 1 number of pulses/turns	1024	1024	1	65535	1024	
P0517	Counter 1 speed filter	500	500	1	4000	500	Hz
P0519	Error Enable	0: Disable	0: Di...	0	2	0: Disable	
P0520	Count Mode: Counter 2	0: Disabled	0: Di...	0	3	0: Disabled	
P0521	Null Pulse 2 options	0: Disabled	0: Di...	0	5	0: Disabled	
P0522	Load Counter 2 value	0	0	0	1	0	
P0523	Counter 2 value - Low	0	0	0	65535	0	
P0524	Counter 2 value - High	0	0	0	65535	0	
P0526	Counter 2 number of pulses/turns	1024	1024	1	65535	1024	
P0527	Counter 2 speed filter	500	500	1	4000	500	Hz
P0529	Error Enable	0: Disabled	0: Di...	0	2	0: Disabled	
P0531	Stop function: level or edge	0: stop activation by level	0: st...	0	1	0: stop activation by level	
P0550	Trigger 1 Signal Source	0: Disabled	0: Di...	0	48	0: Disabled	
P0551	Trigger 1 Value	0	0	-32768	32767	0	
P0552	Trigger 1 Condition	0: Higher or equal than reference value	0: Hi...	0	1	0: Higher or equal than reference value	
P0553	Trigger 2 Signal Source	0: Disable	0: Di...	0	48	0: Disable	
P0554	Trioper 2 Value	0		-32768	32767	0	

Output

WEG SCA-06 V1.40 localhost:502.USB/@0

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 04 Serial**
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Parameter	Description	User	Monito...	Minimum	Maximum	Factory settings	Unit
P0650	Serial Address 1	1	1	1	247	1	
P0652	Bit Rate of Serial 1	1: 9600 bits/s	1: 9600 ...	0	11	1: 9600 bits/s	
P0653	Data bits, parity and stop bit of Serial 1	3: 8bits, no parity, 2stop bit	3: 8bits,...	0	11	3: 8bits, no parity, 2stop bit	
P0654	Serial 1 Protocol	2: ModBus	2: ModBus	1	2	2: ModBus	
P0656	Serial Address 2	1	1	0	247	1	
P0658	Bit Rate of Serial 2	1: 9600 bits/s	1: 9600 ...	0	11	1: 9600 bits/s	
P0659	Data bits, parity and stop bit of Serial 2	3: 8bits, no parity, 2stop bit	3: 8bits,...	0	11	3: 8bits, no parity, 2stop bit	
P0660	Serial 2 Protocol	2: ModBus	2: ModBus	1	2	2: ModBus	
P0663	Timeout for communication	0.0	0.0	0.0	999.9	0.0	s
P0664	Save parameters in non volatile memory	1: Save parameters	1: Save ...	0	1	1: Save parameters	
P0667	Parameters serial access remapping	0: Read/Write in parameters	0: Read/...	0	1	0: Read/Write in parameters	

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

4. **Desocultar Grupos de Parâmetros.** Para exibir um grupo de parâmetros que foi oculto, basta selecionar a raiz da árvore de grupos e selecionar a opção **Desocultar grupo**, na sequência uma janela se abrirá exibindo os grupos que estão ocultos então basta selecionar o grupo que deseja desocultar.

parameter01

Parameters

- 00 Hide group
- 01 Unhide group
- 02 Write group
- 03 Read group
- 06
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	Monitored	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0	0	-9999	9999	0	rpm
P0003	Motor Current	0.0	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	0	307	0	999	0	V
P0006	Drive Status	0: Ready	0: Ready	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 Status	0	0	0	63	0	
P0012	DI207 to DI212 Status	0	0	0	63	0	
P0013	DI301 to DI306 Status	0	0	0	63	0	
P0014	DI307 to DI312 Status	0	0	0	63	0	
P0015	DO1 Status	0	0	0	1	0	
P0016	DO101 to DO106 Status	0	0	0	63	0	
P0017	DO201 to DO206 Status	0	0	0	63	0	
P0018	DO301 to DO306 Status	0	0	0	63	0	
P0021	Internal Air Temperature	0	41	0	1000	0	°C
P0022	Dissipator Temperature	0	33	0	1000	0	°C
P0023	Software Version	9.99	1.40	0.00	655.35	9.99	
P0024	Bootloader Version	0.00	20.03	0.00	655.35	0.00	
P0025	FPGA Project Version	0.00	0.03	0.00	655.35	0.00	
P0030	Present Alarm	0	0	0	2000	0	
P0031	Last Alarm	0	826	0	2000	0	
P0032	Last alarm Day,Month	0.00	3.01	0.00	31.12	0.00	
P0033	Last alarm year	0	1586	0	4096	0	
P0034	Last alarm Hour,Min	0.00	17.01	0.00	23.59	0.00	
P0035	Present Fault	0	0	0	2000	0	
P0036	Last Fault	0	2	0	2000	0	
P0037	Last Fault Day,Month	0.00	3.01	0.00	31.12	0.00	
P0038	Last fault year	0	1594	0	4096	0	
P0039	Last fault Hour,Min	0.00	17.13	0.00	23.59	0.00	
P0040	Second fault	0	2	0	2000	0	
P0041	Second fault Day,Month	0.00	3.01	0.00	31.12	0.00	
P0042	Second fault year	0	1594	0	4096	0	
P0043	Second fault Hour,Min	0.00	17.12	0.00	23.59	0.00	
P0044	Third fault	0	32	0	2000	0	
P0045	Third fault Day,Month	0.00	1.01	0.00	31.12	0.00	
P0046	Third fault year	0	1577	0	4096	0	
P0047	Third fault Hour,Min	0.00		0.00	23.59	0.00	

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

parameter01

Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	Monitored	Minimum	Maxi...	Factory settings	Unit
P0002	Motor Speed	0	0	-9999	9999	0	/rpm
P0003	Motor Current	0.0	0.0	-999.9	999.9	0.0	A
P0004	DC Link Voltage (Vd)	0	308	0	999	0	V
P0006	Drive Status	0: Ready	0: Ready	0	5	0: Ready	
P0008	DI3 to DI1 Status	0	0	0	7	0	
P0009	DI101 to DI106 Status	0	0	0	63	0	
P0010	DI107 to DI112 Status	0	0	0	63	0	
P0011	DI201 to DI206 S			0	63	0	
P0012	DI207 to DI212 S			0	63	0	
P0013	DI301 to DI306 S			0	63	0	
P0014	DI307 to DI312 S			0	63	0	
P0015	DO1 Status			0	1	0	
P0016	DO101 to DO106			0	63	0	
P0017	DO201 to DO206			0	63	0	
P0018	DO301 to DO306			0	63	0	
P0021	Internal Air Temp		1	0	1000	0	°C
P0022	Dissipator Tempe		3	0	1000	0	°C
P0023	Software Version		.40	0.00	655.35	9.99	
P0024	Bootloader Version		0.03	0.00	655.35	0.00	
P0025	FPGA Project Vers		.03	0.00	655.35	0.00	
P0030	Present Alarm			0	2000	0	
P0031	Last Alarm		26	0	2000	0	
P0032	Last alarm Day,Mo		.01	0.00	31.12	0.00	
P0033	Last alarm year		586	0	4096	0	
P0034	Last alarm Hour,Min		7.01	0.00	23.59	0.00	
P0035	Present Fault			0	2000	0	
P0036	Last Fault			0	2000	0	
P0037	Last Fault Day,Mo		.01	0.00	31.12	0.00	
P0038	Last fault year		594	0	4096	0	
P0039	Last fault Hour,Min		7.13	0.00	23.59	0.00	
P0040	Second fault		2	0	2000	0	
P0041	Second fault Day,Month	0.00	3.01	0.00	31.12	0.00	
P0042	Second fault year	0	1594	0	4096	0	
P0043	Second fault Hour,Min	0.00	17.12	0.00	23.59	0.00	
P0044	Third fault	0	32	0	2000	0	
P0045	Third fault Day,Month	0.00	1.01	0.00	31.12	0.00	
P0046	Third fault year	0	1577	0	4096	0	
P0047	Third fault Hour,Min	0.00		0.00	23.59	0.00	

Select groups

Select to unhide:

03 Special Function

OK Cancel

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

parameter01

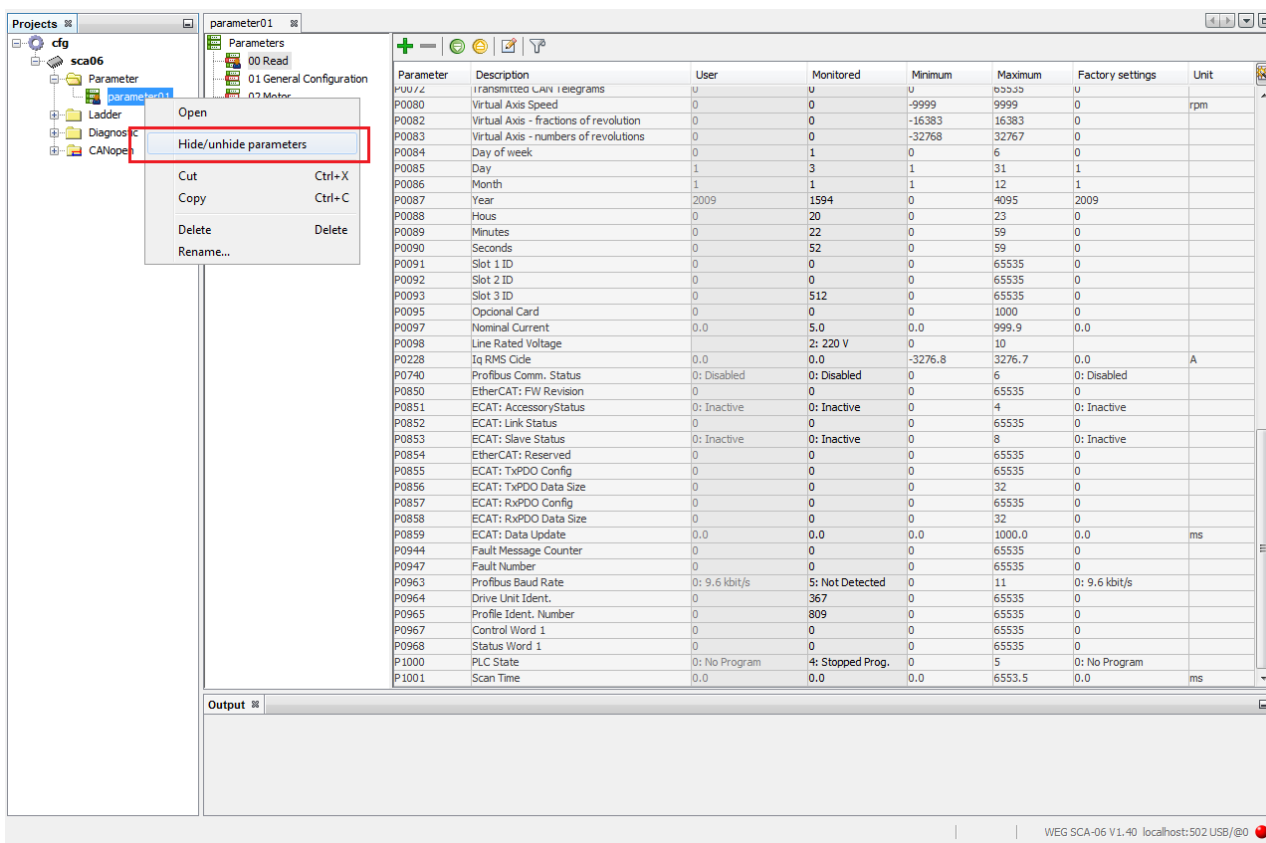
Parameters

- 00 Read
- 01 General Configuration
- 02 Motor
- 03 Special Function
- 04 Serial
- 05 CAN
- 06 Profibus
- 07 PLC
- 08 User
- 09 EtherCAT

Para...	Description	User	M...	Minimum	Maxim...	Factory settings	Unit
P0002	Motor Speed	0		-9999	9999	0	rpm
P0003	Motor Current	0.0		-999.9	999.9	0.0	A
P0004	DC Link Voltage (Ud)	0		0	999	0	V
P0006	Drive Status	0: Ready		0	5	0: Ready	
P0008	DI3 to DI1 Status	0		0	7	0	
P0009	DI101 to DI106 Status	0		0	63	0	
P0010	DI107 to DI112 Status	0		0	63	0	
P0011	DI201 to DI206 Status	0		0	63	0	
P0012	DI207 to DI212 Status	0		0	63	0	
P0013	DI301 to DI306 Status	0		0	63	0	
P0014	DI307 to DI312 Status	0		0	63	0	
P0015	DO1 Status	0		0	1	0	
P0016	DO101 to DO106 Status	0		0	63	0	
P0017	DO201 to DO206 Status	0		0	63	0	
P0018	DO301 to DO306 Status	0		0	63	0	
P0021	Internal Air Temperature	0		0	1000	0	°C
P0022	Dissipator Temperature	0		0	1000	0	°C
P0023	Software Version	9.99		0.00	655.35	9.99	
P0024	Bootloader Version	0.00		0.00	655.35	0.00	
P0025	FPGA Project Version	0.00		0.00	655.35	0.00	
P0030	Present Alarm	0		0	2000	0	
P0031	Last Alarm	0		0	2000	0	
P0032	Last alarm Day.Month	0.00		0.00	31.12	0.00	
P0033	Last alarm year	0		0	4096	0	
P0034	Last alarm Hour.Min	0.00		0.00	23.59	0.00	
P0035	Present Fault	0		0	2000	0	
P0036	Last Fault	0		0	2000	0	
P0037	Last Fault Day.Month	0.00		0.00	31.12	0.00	
P0038	Last fault year	0		0	4096	0	
P0039	Last fault Hour.Min	0.00		0.00	23.59	0.00	
P0040	Second fault	0		0	2000	0	
P0041	Second fault Day.Month	0.00		0.00	31.12	0.00	
P0042	Second fault year	0		0	4096	0	
P0043	Second fault Hour.Min	0.00		0.00	23.59	0.00	
P0044	Third fault	0		0	2000	0	
P0045	Third fault Day.Month	0.00		0.00	31.12	0.00	
P0046	Third fault year	0		0	4096	0	

Output

5. **Ocultar e Exibir Parâmetros e Grupos de Parâmetros.** Através desta opção se tem o controle total dos parâmetros e grupos de parâmetros, é possível ocultar e desocultar parâmetros individuais, múltiplos parâmetros, grupos individuais e múltiplos grupos em uma mesma ação.



parameter01

Parameters

00 Read

01 General Configuration

02 Motor

03 Special Function

04 Serial

05 CAN

06 Profibus

07 PLC

08 User

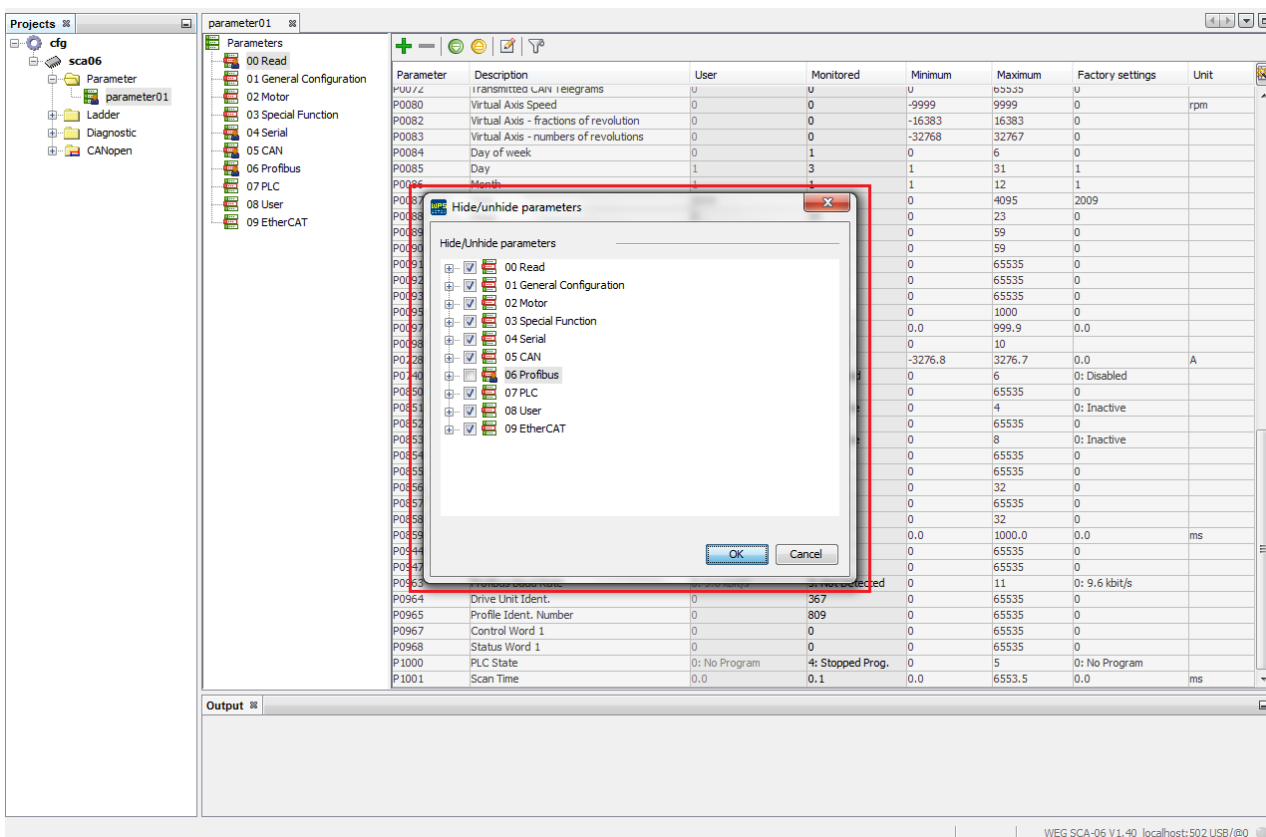
09 EtherCAT

Parameter Description User Monitored Minimum Maximum Factory settings Unit

Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams	U	U	U	65535	U	
P0080	Virtual Axis Speed	0	0	-9999	9999	0	rpm
P0082	Virtual Axis - fractions of revolution	0	0	-16383	16383	0	
P0083	Virtual Axis - numbers of revolutions	0	0	-32768	32767	0	
P0084	Day of week	0	1	0	6	0	
P0085	Day	1	3	1	31	1	
P0086	Month	1	1	1	12	1	
P0087	Year	2009	1594	0	4095	2009	
P0088	Hours	0	20	0	23	0	
P0089	Minutes	0	22	0	59	0	
P0090	Seconds	0	52	0	59	0	
P0091	Slot 1 ID	0	0	0	65535	0	
P0092	Slot 2 ID	0	0	0	65535	0	
P0093	Slot 3 ID	0	512	0	65535	0	
P0095	Optional Card	0	0	0	1000	0	
P0097	Nominal Current	0.0	5.0	0.0	999.9	0.0	
P0098	Line Rated Voltage	0	2: 220 V	0	10	0	
P0228	Iq RMS Cicle	0.0	0.0	-3276.8	3276.7	0.0	A
P0740	Profibus Comm. Status	0: Disabled	0: Disabled	0	6	0: Disabled	
P0850	EtherCAT: FW Revision	0	0	0	65535	0	
P0851	ECAT: AccessoryStatus	0: Inactive	0: Inactive	0	4	0: Inactive	
P0852	ECAT: Link Status	0	0	0	65535	0	
P0853	ECAT: Slave Status	0: Inactive	0: Inactive	0	8	0: Inactive	
P0854	EtherCAT: Reserved	0	0	0	65535	0	
P0855	ECAT: TxPDO Config	0	0	0	65535	0	
P0856	ECAT: TxPDO Data Size	0	0	0	32	0	
P0857	ECAT: RxPDO Config	0	0	0	65535	0	
P0858	ECAT: RxPDO Data Size	0	0	0	32	0	
P0859	ECAT: Data Update	0.0	0.0	0.0	1000.0	0.0	ms
P0944	Fault Message Counter	0	0	0	65535	0	
P0947	Fault Number	0	0	0	65535	0	
P0963	Profibus Baud Rate	0: 9.6 kbit/s	5: Not Detected	0	11	0: 9.6 kbit/s	
P0964	Drive Unit Ident.	0	367	0	65535	0	
P0965	Profile Ident. Number	0	809	0	65535	0	
P0967	Control Word 1	0	0	0	65535	0	
P0968	Status Word 1	0	0	0	65535	0	
P1000	PLC State	0: No Program	4: Stopped Prog.	0	5	0: No Program	
P1001	Scan Time	0.0	0.0	0.0	6553.5	0.0	ms

Output

WEG SCA-06 V1.40 localhost:502 USB/@0



parameter01

Parameters

00 Read

01 General Configuration

02 Motor

03 Special Function

04 Serial

05 CAN

06 Profibus

07 PLC

08 User

09 EtherCAT

Parameter Description User Monitored Minimum Maximum Factory settings Unit

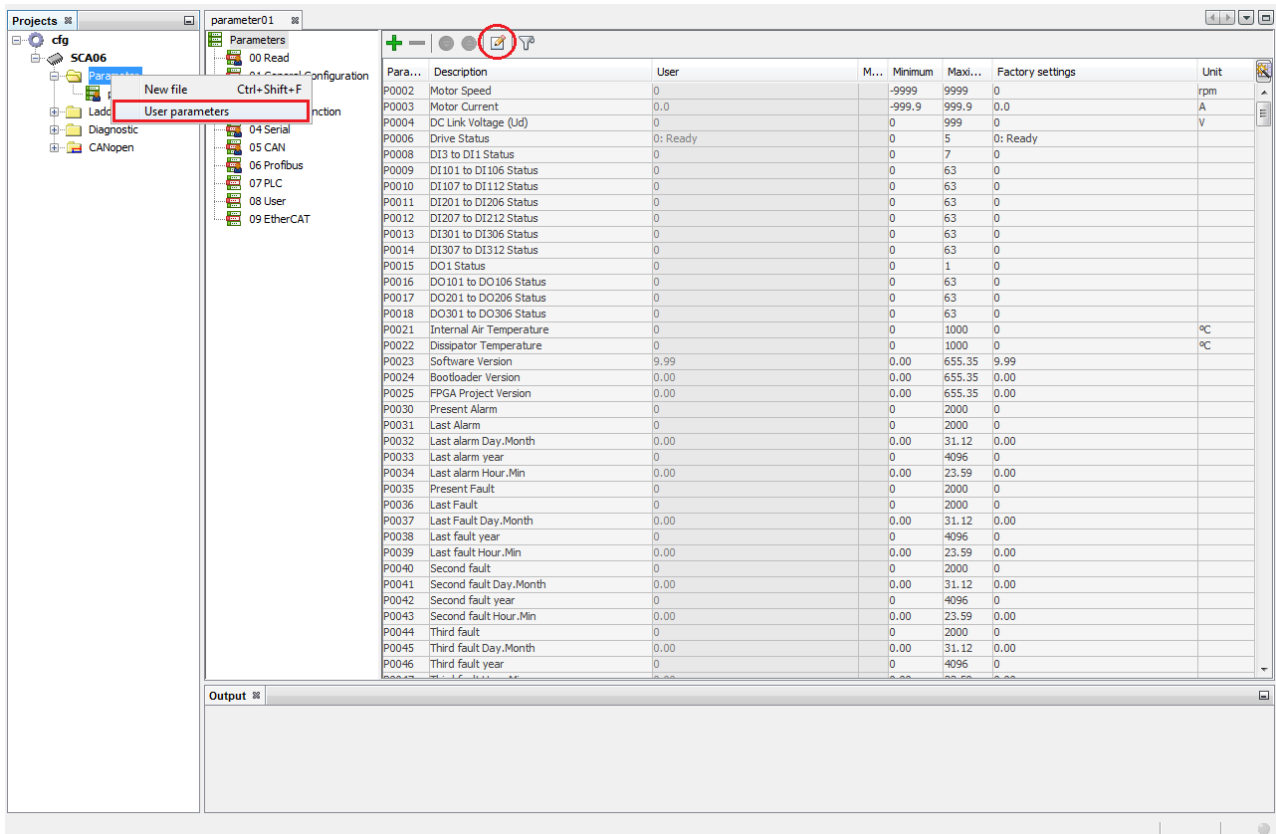
Parameter	Description	User	Monitored	Minimum	Maximum	Factory settings	Unit
P0072	Transmitted CAN Telegrams	U	U	U	65535	U	
P0080	Virtual Axis Speed	0	0	-9999	9999	0	rpm
P0082	Virtual Axis - fractions of revolution	0	0	-16383	16383	0	
P0083	Virtual Axis - numbers of revolutions	0	0	-32768	32767	0	
P0084	Day of week	0	1	0	6	0	
P0085	Day	1	3	1	31	1	
P0086	Month	1	1	1	12	1	
P0087	Year	2009	1594	0	4095	2009	
P0088	Hours	0	20	0	23	0	
P0089	Minutes	0	22	0	59	0	
P0090	Seconds	0	52	0	59	0	
P0091	Slot 1 ID	0	0	0	65535	0	
P0092	Slot 2 ID	0	0	0	65535	0	
P0093	Slot 3 ID	0	512	0	65535	0	
P0095	Optional Card	0	0	0	1000	0	
P0097	Nominal Current	0.0	5.0	0.0	999.9	0.0	
P0098	Line Rated Voltage	0	2: 220 V	0	10	0	
P0228	Iq RMS Cicle	0.0	0.0	-3276.8	3276.7	0.0	A
P0740	Profibus Comm. Status	0: Disabled	0: Disabled	0	6	0: Disabled	
P0850	EtherCAT: FW Revision	0	0	0	65535	0	
P0851	ECAT: AccessoryStatus	0: Inactive	0: Inactive	0	4	0: Inactive	
P0852	ECAT: Link Status	0	0	0	65535	0	
P0853	ECAT: Slave Status	0: Inactive	0: Inactive	0	8	0: Inactive	
P0854	EtherCAT: Reserved	0	0	0	65535	0	
P0855	ECAT: TxPDO Config	0	0	0	65535	0	
P0856	ECAT: TxPDO Data Size	0	0	0	32	0	
P0857	ECAT: RxPDO Config	0	0	0	65535	0	
P0858	ECAT: RxPDO Data Size	0	0	0	32	0	
P0859	ECAT: Data Update	0.0	0.0	0.0	1000.0	0.0	ms
P0944	Fault Message Counter	0	0	0	65535	0	
P0947	Fault Number	0	0	0	65535	0	
P0963	Profibus Baud Rate	0: 9.6 kbit/s	5: Not Detected	0	11	0: 9.6 kbit/s	
P0964	Drive Unit Ident.	0	367	0	65535	0	
P0965	Profile Ident. Number	0	809	0	65535	0	
P0967	Control Word 1	0	0	0	65535	0	
P0968	Status Word 1	0	0	0	65535	0	
P1000	PLC State	0: No Program	4: Stopped Prog.	0	5	0: No Program	
P1001	Scan Time	0.0	0.1	0.0	6553.5	0.0	ms

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

10.3.4.5 Parâmetros do Usuário

Para abrir a tela de configuração dos parâmetros de usuário basta clicar na opção **Parâmetros do Usuário** no nó **Parâmetros** da árvore do projeto ou clicar no ícone indicado na barra de ferramentas do arquivo de parâmetros.



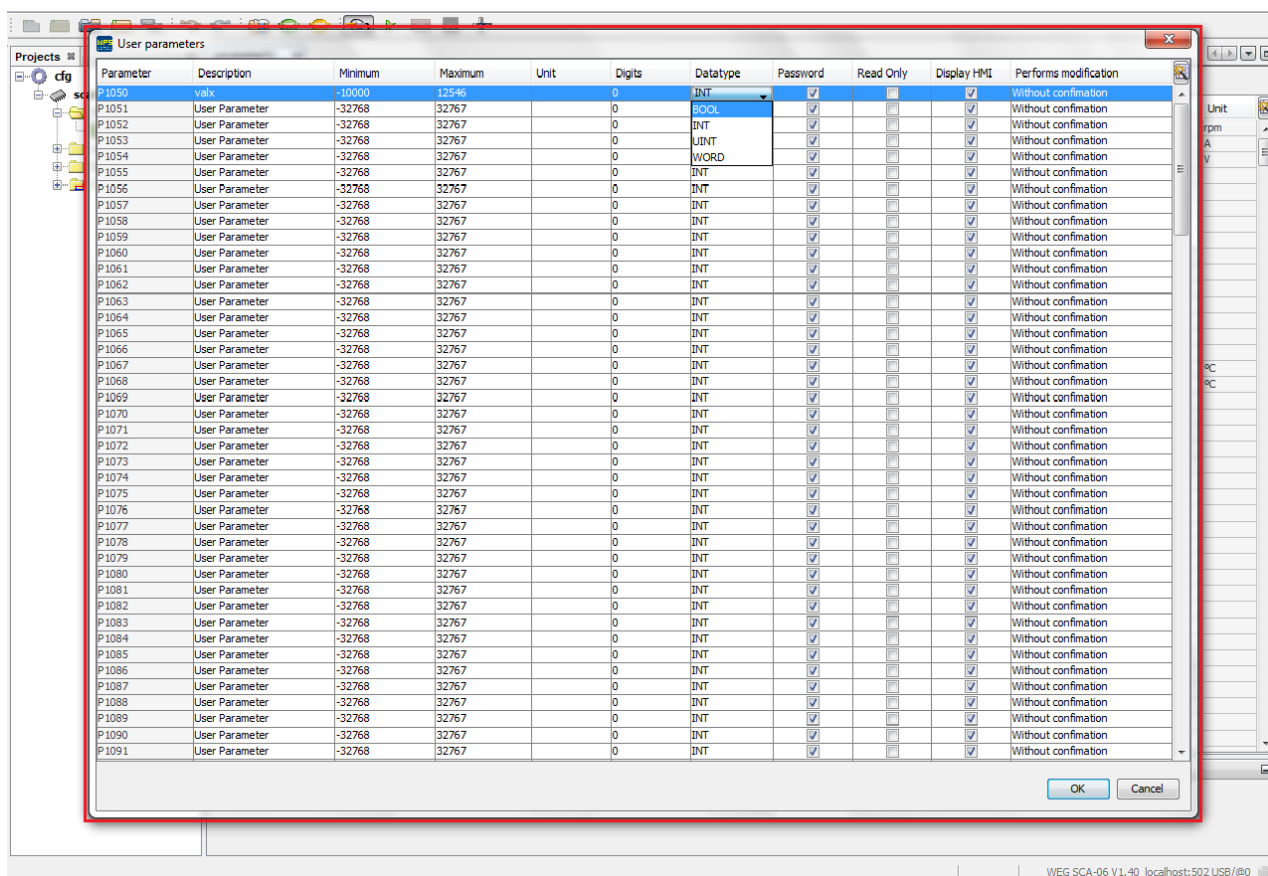
Tela de Configuração

Na tela de configuração dos parâmetros dos usuários é possível definir vários atributos dos parâmetros como descrição, valores mínimos e máximos, unidade, dígitos, tipo de dados entre outros.



NOT A!

Estas configurações serão exibidas automaticamente no arquivo de parâmetros, porém para que sejam enviados ao equipamento é necessário fazer o *download* do recurso.



Exibição do parâmetro do usuário

Na tabela de parâmetros os parâmetros do usuário serão exibidos conforme configurados na tela de configuração.

Projects: **cfg** | **sca06** | Parameter | **parameter01**

parameter01

- Parameters
 - 00 Read
 - 01 General Configuration
 - 02 Motor
 - 03 Special Function
 - 04 Serial
 - 05 CAN
 - 06 Profibus
 - 07 PLC
 - 08 User
 - 09 EtherCAT

Search: Parameter | 1050

Pa...	Description	User	Monitored	Mini...	Ma...	Factory settings	Unit
P1021	Scan Time	5.0	5.0	0.5	200.0	5.0	ms
P1022	PLC Watchdog	0	0	0	5	0	
P1023	Control Mode on Power on	3: Position	3: Position	2	3	3: Position	
P1027	Ret. Markers Reset	0: Disabled	0: Disabled	0	1	0: Disabled	
P1028	Load ladder	0: Disabled	0: Disabled	0	3	0: Disabled	
P1031	Maximum stopped lag error	0	0	0	16383	0	
P1032	Maximum following lag error	0	0	0	16383	0	
P1035	CAM Speed Filter	500	500	1	4000	500	Hz
P1050	ivalx	0		-10000	12546	0	
P1051	User Parameter	0		-32768	32767	0	
P1052	User Parameter	0		-32768	32767	0	
P1053	User Parameter	0		-32768	32767	0	
P1054	User Parameter	0		-32768	32767	0	
P1055	User Parameter	0		-32768	32767	0	
P1056	User Parameter	0		-32768	32767	0	
P1057	User Parameter	0		-32768	32767	0	
P1058	User Parameter	0		-32768	32767	0	
P1059	User Parameter	0		-32768	32767	0	
P1060	User Parameter	0		-32768	32767	0	
P1061	User Parameter	0		-32768	32767	0	
P1062	User Parameter	0		-32768	32767	0	
P1063	User Parameter	0		-32768	32767	0	
P1064	User Parameter	0		-32768	32767	0	
P1065	User Parameter	0		-32768	32767	0	
P1066	User Parameter	0		-32768	32767	0	
P1067	User Parameter	0		-32768	32767	0	
P1068	User Parameter	0		-32768	32767	0	
P1069	User Parameter	0		-32768	32767	0	
P1070	User Parameter	0		-32768	32767	0	
P1071	User Parameter	0		-32768	32767	0	
P1072	User Parameter	0		-32768	32767	0	
P1073	User Parameter	0		-32768	32767	0	
P1074	User Parameter	0		-32768	32767	0	
P1075	User Parameter	0		-32768	32767	0	
P1076	User Parameter	0		-32768	32767	0	
P1077	User Parameter	0		-32768	32767	0	
P1078	User Parameter	0		-32768	32767	0	
P1079	User Parameter	0		-32768	32767	0	
P1080	User Parameter	0		-32768	32767	0	

Output

WEG SCA-06 V1.40 localhost:502 USB/@0

10.3.5 Ladder

[Componentes e Blocos](#)

[Estruturas](#)

10.3.5.1 Componentes e Blocos

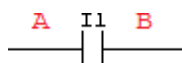
Ver a seguir.

10.3.5.1.1 Contact

10.3.5.1.1.1 NOCONTACT

Contato normalmente aberto.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	I1	BOOL	Entrada de controle do bloco

Funcionamento

Quando a variável I1 está com o valor FALSE, B recebe FALSE.

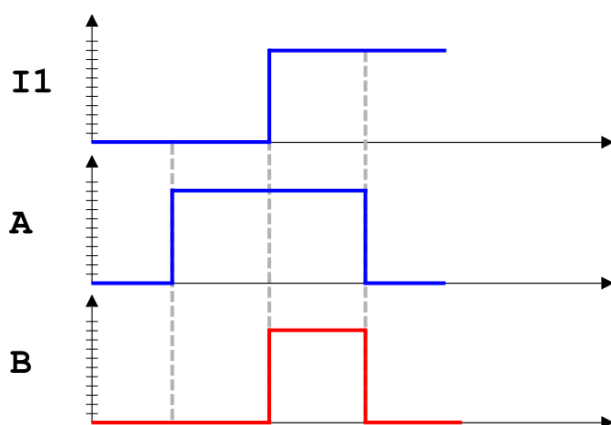
Quando a variável I1 está com o valor TRUE, B recebe o valor de A.



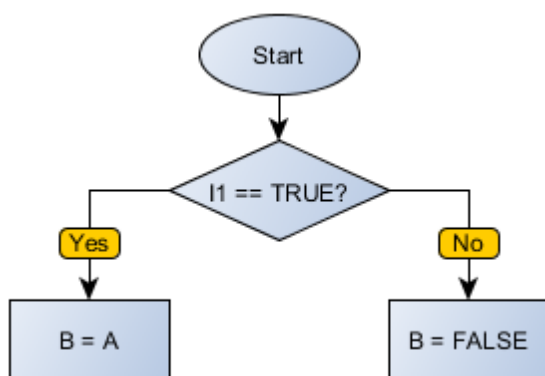
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

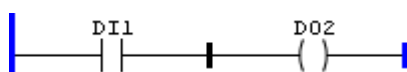
Diagrama



Fluxograma do Bloco



Exemplo

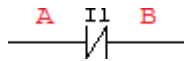


O exemplo acima realiza a transferência do valor da entrada digital DI1 para a saída digital DO2.

10.3.5.1.1.2 NCCONTACT

Contato normalmente fechado.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	I1	BOOL	Entrada de controle do bloco

Funcionamento

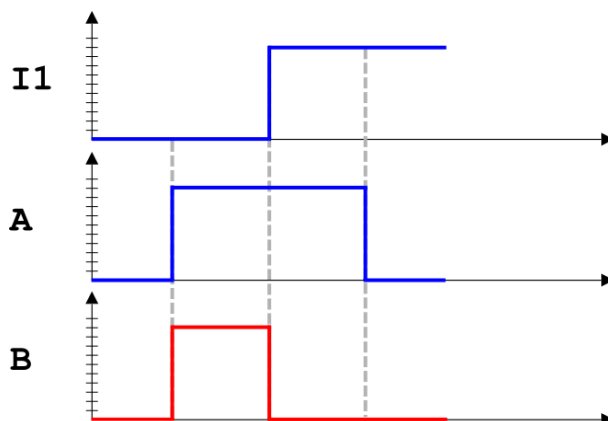
Quando a variável I1 está com o valor TRUE, B recebe FALSE.

Quando a variável I1 está com o valor FALSE, B recebe o valor de A.

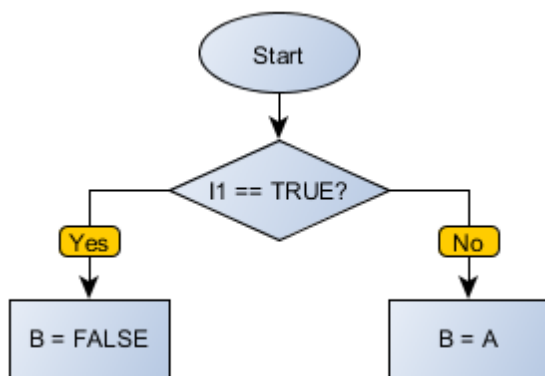
**NOTA!**

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

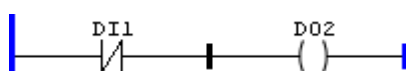
Diagrama



Fluxograma do Bloco



Exemplo

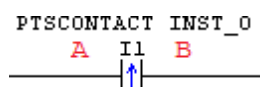


O exemplo acima realiza a transferência do oposto do valor da entrada digital DI1 para a saída digital DO2.

10.3.5.1.1.3 PTSCONTACT

Contato de transição de borda de subida.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	I1	BOOL	Entrada de controle do bloco
VAR	PTSCONTACT_INST_0	PTSCONTACT	Instância de acesso à estrutura do bloco

Funcionamento

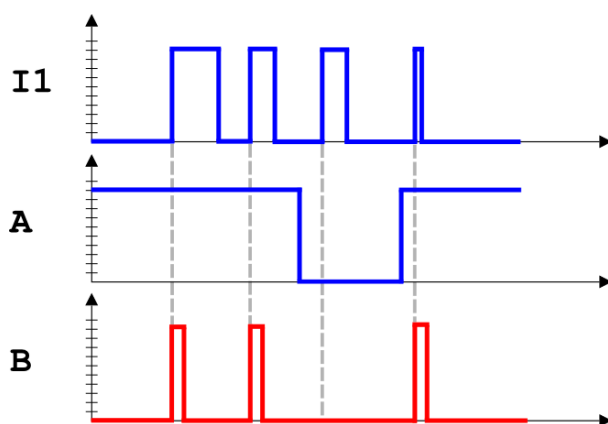
No instante em que a variável I1 transita de FALSE para TRUE (borda de subida ou borda de transição positiva), B recebe o valor de A por um ciclo de scan. Em todos os outros momentos, B recebe o valor FALSE.



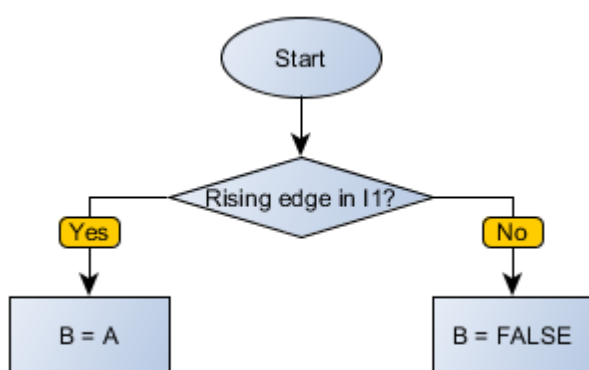
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

Diagrama



Fluxograma do Bloco



Exemplo

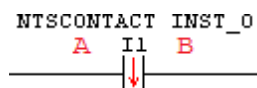


O exemplo acima reseta a saída digital DO1 caso a tecla SHIFT esteja pressionada e seja dado um pulso positivo na entrada digital DI2.

10.3.5.1.1.4 NTSCONTACT

Contato de transição de borda de descida.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	I1	BOOL	Entrada de controle do bloco
VAR	NTSCONTACT_INST_0	NTSCONTACT	Instância de acesso à estrutura do bloco

Funcionamento

No instante em que a variável I1 transita de TRUE para FALSE (borda de descida ou borda de transição negativa), B recebe o valor de A por um ciclo de scan.

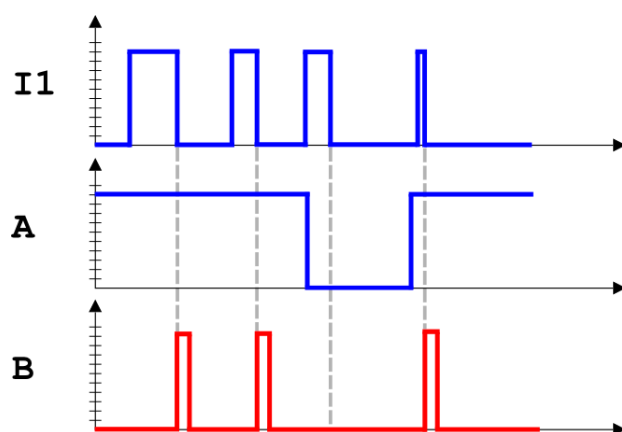
Em todos os outros momentos, B recebe o valor FALSE.



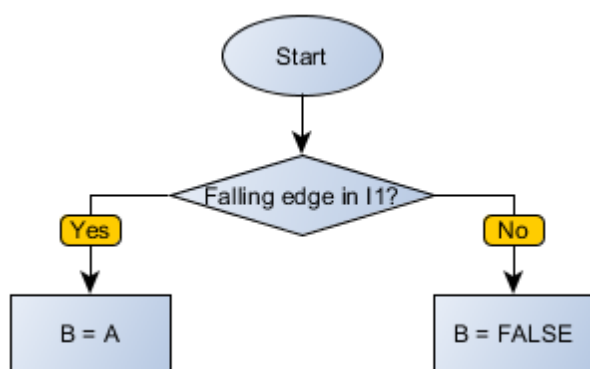
NOTA!

Atente para as associações série e paralelo dos contatos. Veja a seção [Lógica com Contatos](#) para mais informações.

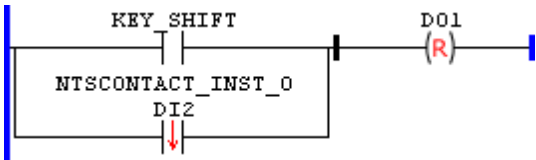
Diagrama



Fluxograma do Bloco



Exemplo



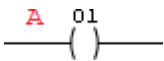
O exemplo acima reseta a saída digital DO1 caso a tecla SHIFT esteja pressionada ou seja dado um pulso negativo na entrada digital DI2.

10.3.5.1.2 Coil

10.3.5.1.2.1 DIRECTCOIL

Bloco lógico utilizado para atribuição de valores diretos a variáveis de saída.

Representação Ladder



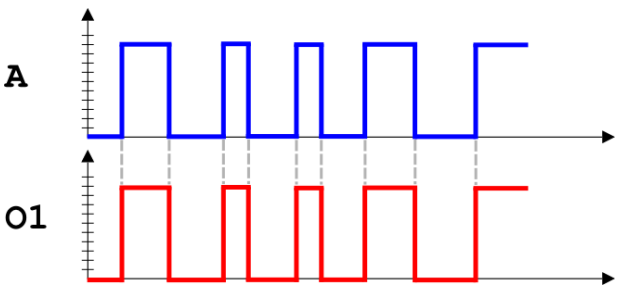
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

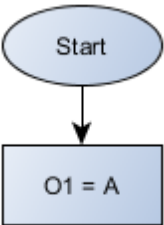
Funcionamento

Este bloco repassa o valor de A para o endereço de memória correspondente a O1.

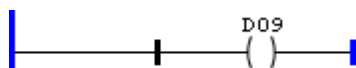
Diagrama



Fluxograma do Bloco



Exemplo

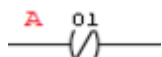


O exemplo acima deixa a saída digital DO9 permanentemente ligada, pois o valor de A, neste caso, é o valor do barramento esquerdo, que é sempre considerado nível lógico alto (TRUE).

10.3.5.1.2.2 INVERTEDCOIL

Bloco lógico utilizado para atribuição de valores negados a variáveis de saída.

Representação Ladder



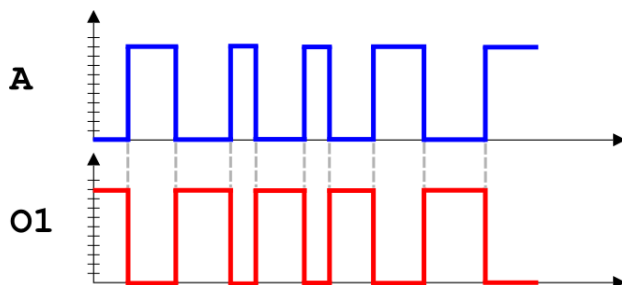
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

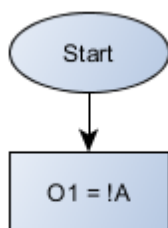
Funcionamento

Este bloco repassa o valor negado de A para o endereço de memória correspondente a O1.

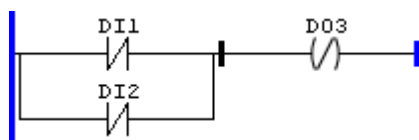
Diagrama



Fluxograma do Bloco



Exemplo

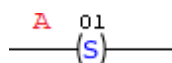


O exemplo acima desativa a saída digital DO3 quando algumas das entradas digitais DI1 ou DI2 estão com valor FALSE. Quando ambas entradas estão com valor TRUE, DO3 ativa.

10.3.5.1.2.3 SETCOIL

Bloco lógico utilizado para habilitação indefinida de variáveis de saída.

Representação Ladder



Estrutura do Bloco

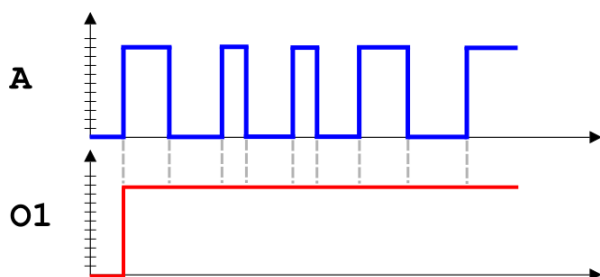
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

Funcionamento

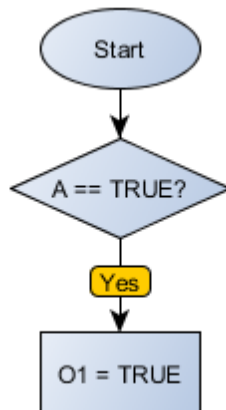
Ao identificar um valor TRUE em A, este bloco repassa um valor TRUE para o endereço de memória correspondente a O1.

Ao identificar um valor FALSE em A, este bloco não realiza nenhuma operação.

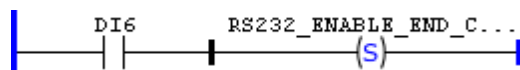
Diagrama



Fluxograma do Bloco



Exemplo

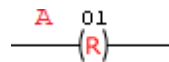


O exemplo acima ativa permanentemente o marcador de controle do sistema habilita caractere de fim de mensagem na comunicação RS232 ao identificar um nível TRUE na entrada digital DI6.

10.3.5.1.2.4 RESETCOIL

Bloco lógico utilizado para desabilitação indefinida de variáveis de saída.

Representação Ladder



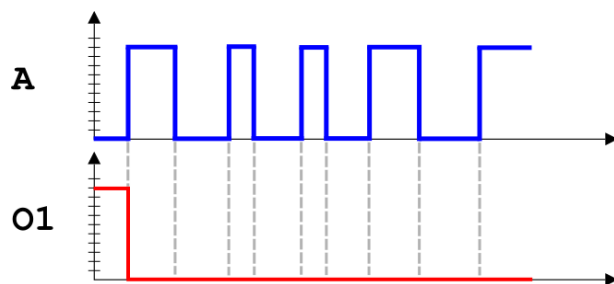
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	B00L	Saída de registro do bloco

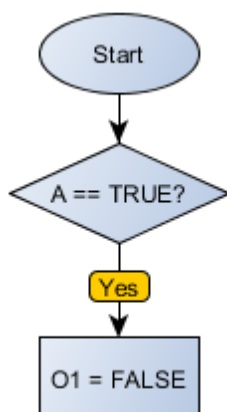
Funcionamento

Ao identificar um valor TRUE em A, este bloco repassa um valor FALSE para o endereço de memória correspondente a O1.
Ao identificar um valor FALSE em A, este bloco não realiza nenhuma operação.

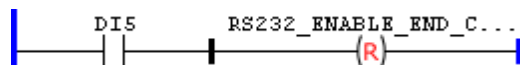
Diagrama



Fluxograma do Bloco



Exemplo

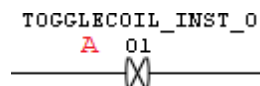


O exemplo acima desativa permanentemente o marcador de controle do sistema que habilita caractere de fim de mensagem na comunicação RS232 ao identificar um nível TRUE na entrada digital DI5.

10.3.5.1.2.5 TOGGLECOIL

Bloco lógico utilizado para alternância de variáveis de saída.

Representação Ladder



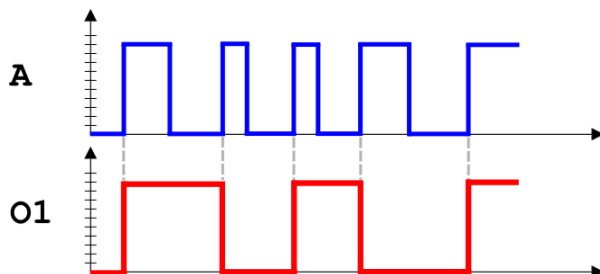
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco
VAR	TOGGLECOIL_INST_0	TOGGLECOIL	Instância de acesso à estrutura do bloco

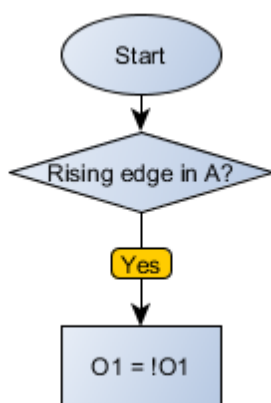
Funcionamento

Ao identificar uma transição de FALSE para TRUE (borda de subida) em A, o bloco inverte o estado de O1.

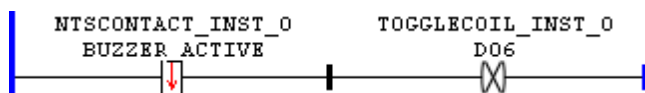
Diagrama



Fluxograma do Bloco



Exemplo

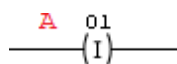


O exemplo acima inverte o estado da saída digital DO6 a cada desativação do buzzer interno.

10.3.5.1.2.6 IMMEDIATECOIL

Bloco lógico utilizado para atribuição de valores a saídas digitais padrão instantaneamente.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_OUTPUT	O1	BOOL	Saída de registro do bloco

Funcionamento

Este bloco repassa o valor de A para a saída digital correspondente a O1. Diferentemente da bobina direta, este bloco não aguarda até o fim do ciclo de scan para que o valor da saída seja atualizado; isto é feito no mesmo momento em que o bloco é ativado.



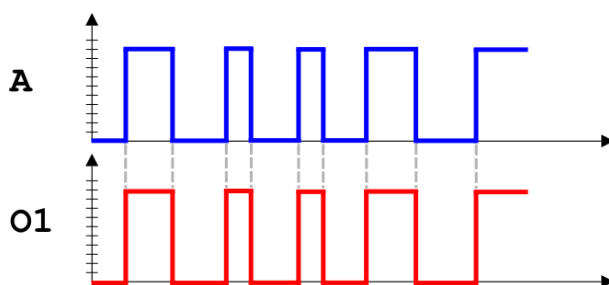
NOTA!

Este bloco apenas funciona com as saídas digitais padrão do produto.

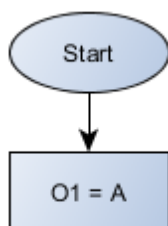
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

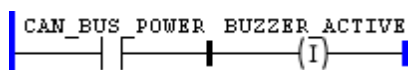
Diagrama



Fluxograma do Bloco



Exemplo



O exemplo acima ativa imediatamente o buzzer interno assim que detectar que a alimentação do barramento da rede CANopen foi interrompida, permanecendo ligado até que a alimentação seja restaurada.

10.3.5.1.3 Logic Bit

10.3.5.1.3.1 RESETBIT

Bloco lógico utilizado para realizar reset de um bit específico em um campo.

Representação Ladder**Estrutura do Bloco**

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será alterado
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	DONE	BOOL	Sucesso da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, reseta o bit indicado em Position na variável Data, que é repassada para a saída já com o seu valor atualizado.

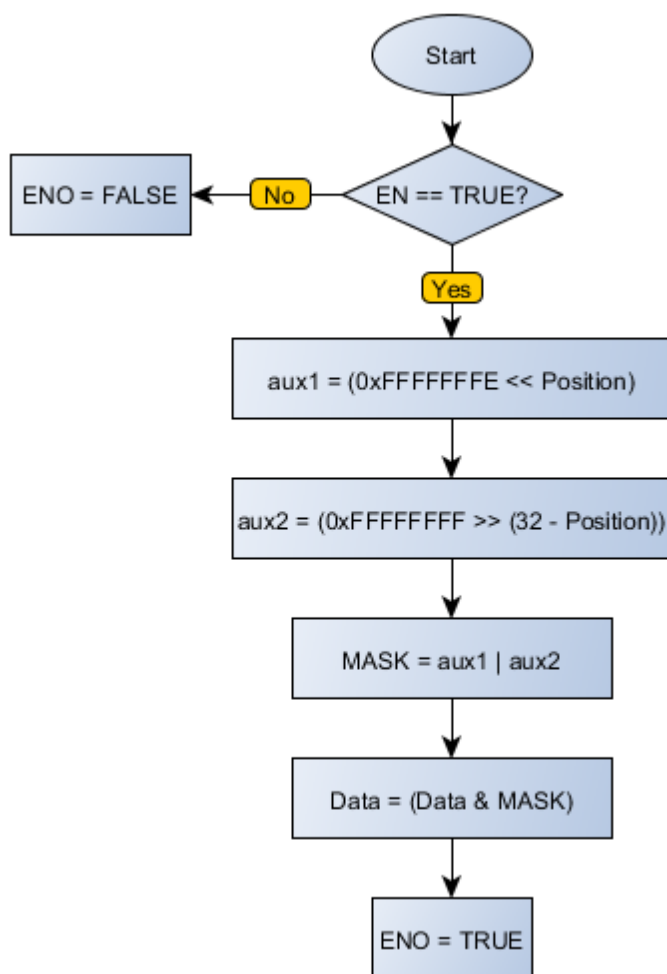
Quando EN possui valor FALSE, Data permanece inalterado.

A variável DONE recebe o mesmo valor de EN, exceto quando há erro no reset do bit, recebendo então um valor FALSE.

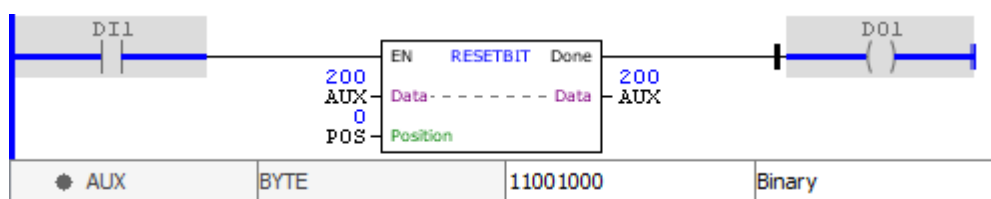
**NOTA!**

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

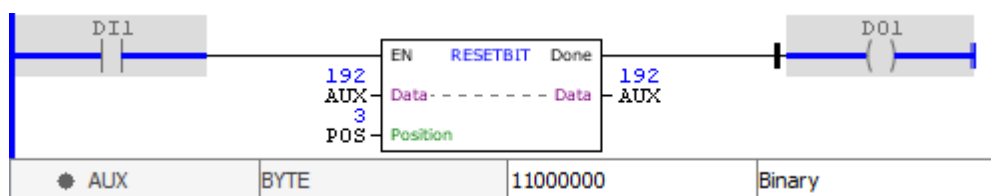
Fluxograma do Bloco



Exemplo

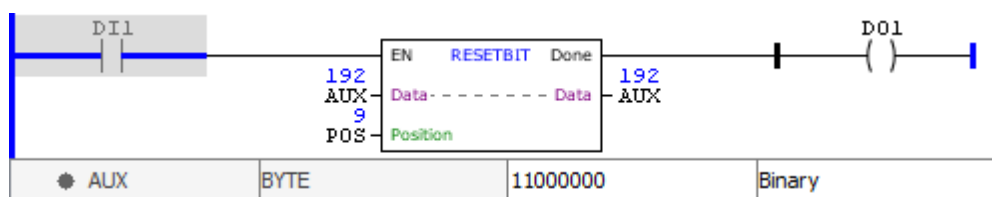


O exemplo acima reseta o bit da posição zero de AUX, cujo valor inicial é 200 (1100 1000, em binário). Como este bit já possuía valor FALSE, nada foi alterado.



O exemplo acima reseta o bit da posição três de AUX, alterando seu valor binário e,

consequentemente, sua representação decimal.

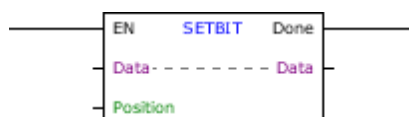


O exemplo acima reseta o bit da posição nove de AUX. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída não é habilitada.

10.3.5.1.3.2 SETBIT

Bloco lógico utilizado para realizar set de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será alterado
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	DONE	BOOL	Sucesso da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, seta o bit indicado em Position na variável Data, que é repassada para a saída já com o seu valor atualizado.

Quando EN possui valor FALSE, Data permanece inalterado.

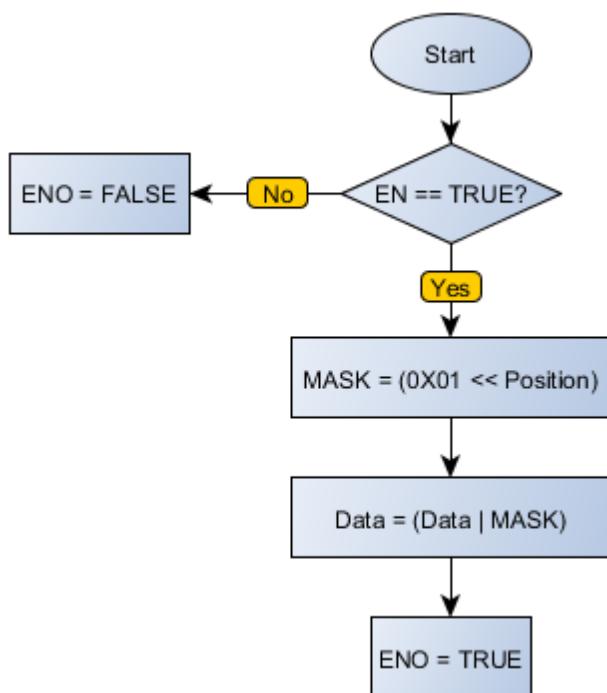
A variável DONE recebe o mesmo valor de EN, exceto quando há erro no set do bit, recebendo então um valor FALSE.



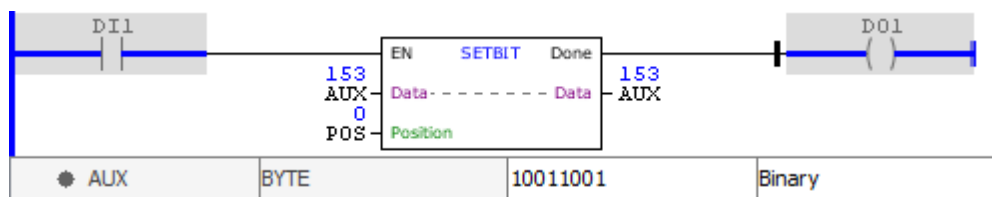
NOTA!

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

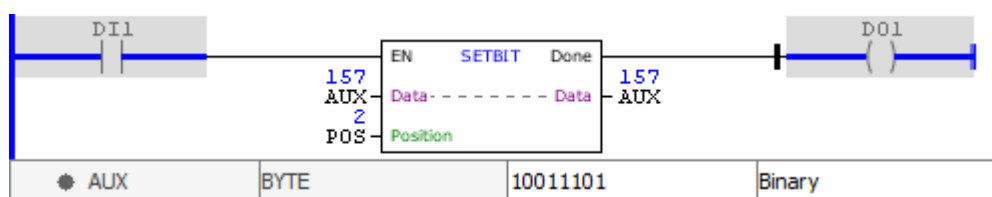
Fluxograma do Bloco



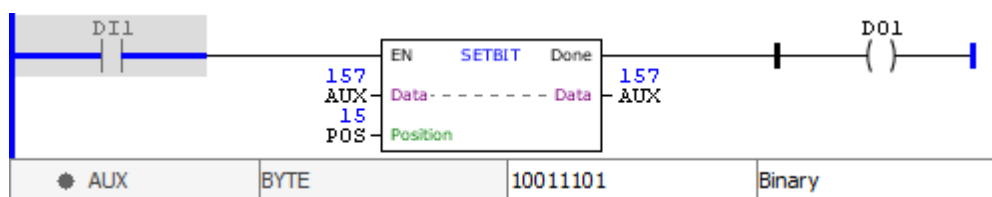
Exemplo



O exemplo acima seta o bit da posição zero de AUX, cujo valor inicial é 153 (1001 1001, em binário). Como este bit já possuía valor TRUE, nada foi alterado.



O exemplo acima seta o bit da posição dois de AUX, alterando seu valor binário e, consequentemente, sua representação decimal.



O exemplo acima seta o bit da posição quinze de AUX. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída não é habilitada.

10.3.5.1.3.3 TESTBIT

Bloco lógico que retorna o valor de um bit específico em um campo.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Data	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável cujo bit será testado
	EN	BOOL	Habilitação do bloco
	Position	BYTE USINT	Posição do bit que será alterado
VAR_OUTPUT	Q	BOOL	Valor do bit testado

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor do bit indicado em Position na variável Data.

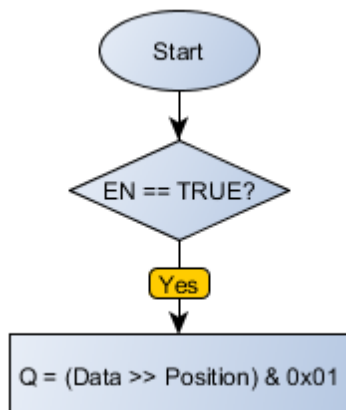
Quando EN possui valor FALSE, Q também recebe FALSE.



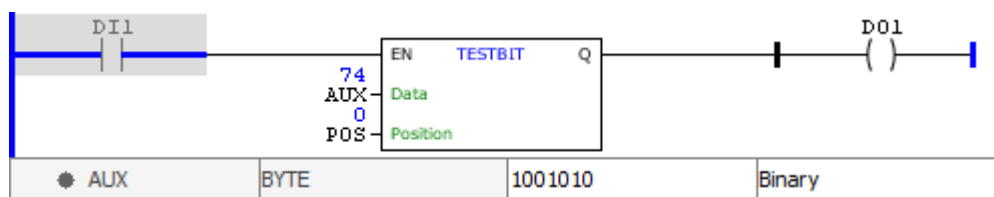
NOTA!

É importante atentar para que Position esteja dentro da faixa de valores de bits correspondente ao tipo de variável em Data. Por exemplo: se Data é um BYTE, logo possui 8 bits, e Position deve compreender um valor entre 0 e 7.

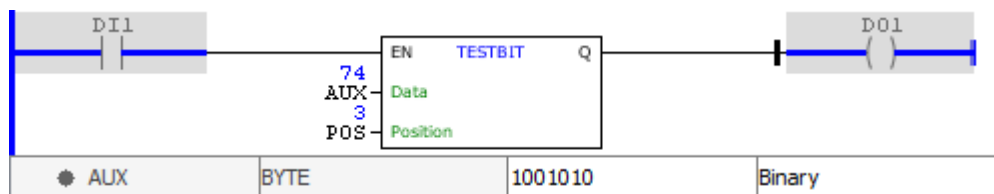
Fluxograma do Bloco



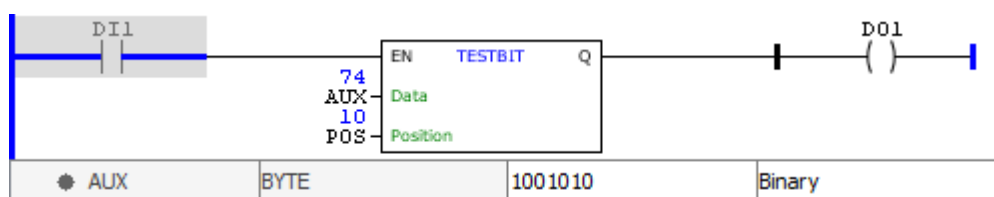
Exemplo



O exemplo acima joga o valor do bit da posição zero de AUX, cujo valor inicial é 74 (0100 1010 em binário) para a saída Q. Como este bit tem valor 0, a saída fica desabilitada.



O exemplo acima joga o valor do bit da posição três de AUX para a saída Q. Como este bit tem valor 1, a saída fica habilitada.



O exemplo acima joga o valor do bit da posição dez de AUX para a saída Q. Como AUX é uma variável do tipo BYTE, ela possui apenas oito bits. Sendo assim, o exemplo acima gera um erro de execução no bloco e, por isso, a saída fica desabilitada.

10.3.5.1.4 Logic Boolean

10.3.5.1.4.1 NOT

Bloco lógico que executa uma operação de negado booleano em uma variável, armazenando o resultado em outra.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável de referência para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

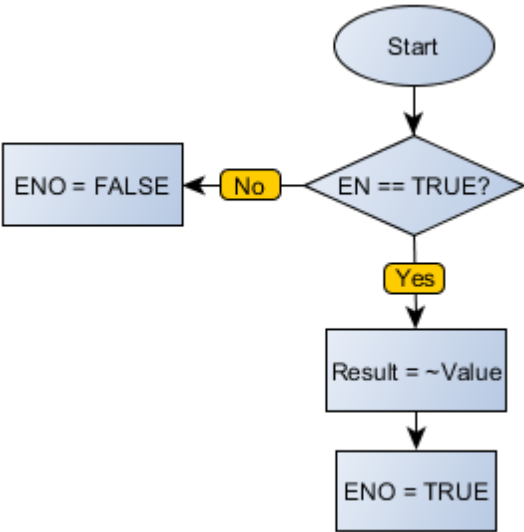
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor booleano negado da variável de entrada Value.

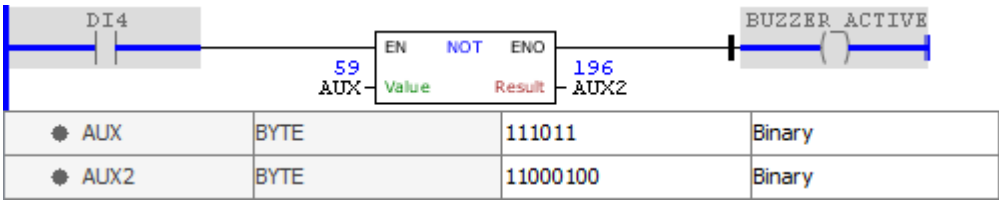
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

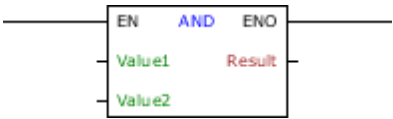


O exemplo acima realiza a operação negado booleano em AUX, armazenando o resultado em AUX2.

10.3.5.1.4.2 AND

Bloco lógico que executa uma operação de “e” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

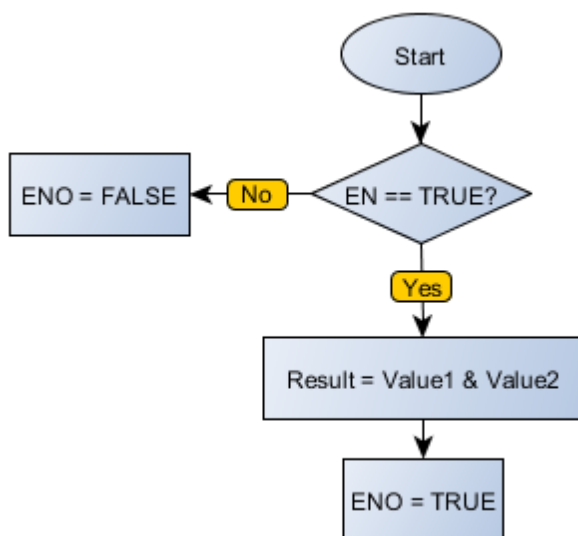
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “e” das variáveis de entrada Value1 e Value2.

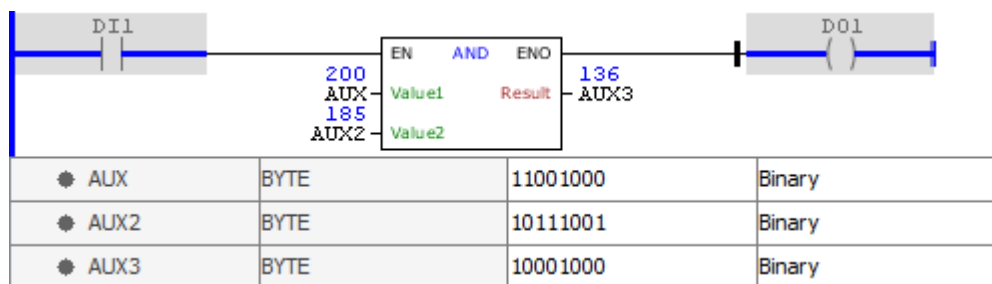
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação e booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.3.5.1.4.3 OR

Bloco lógico que executa uma operação de “ou” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

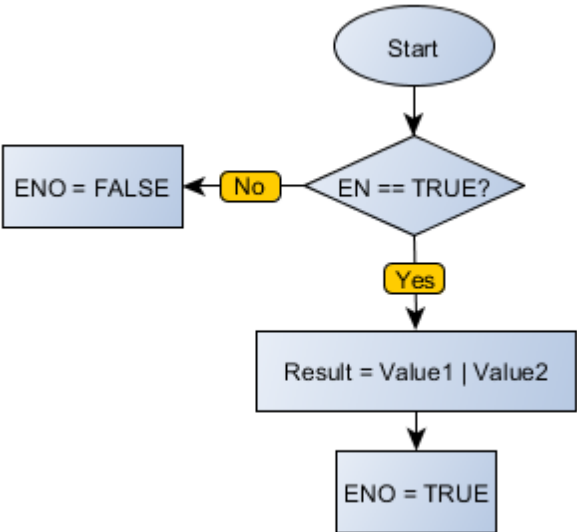
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou” das variáveis de entrada Value1 e Value2.

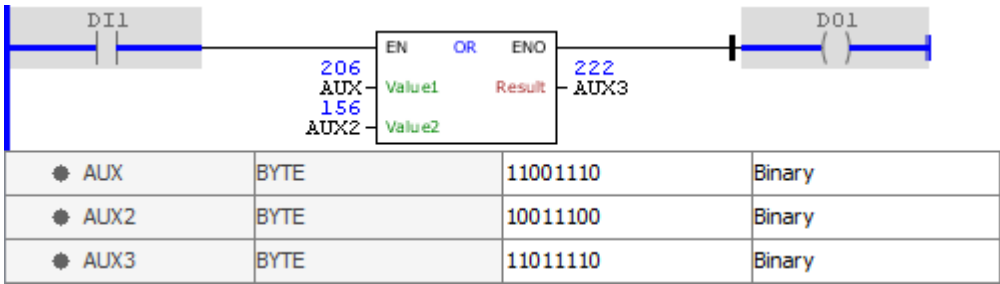
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação ou booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.3.5.1.4.4 XOR

Bloco lógico que executa uma operação de “ou exclusivo” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

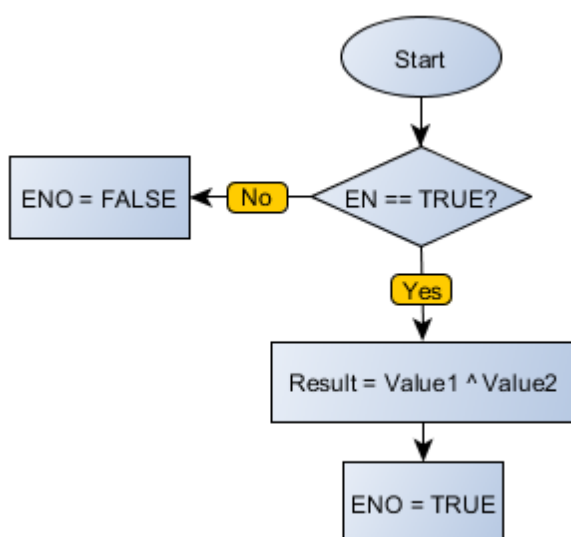
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou exclusivo” das variáveis de entrada Value1 e Value2.

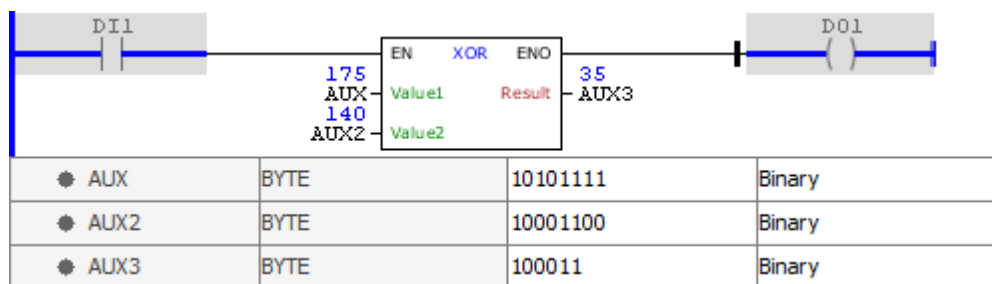
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

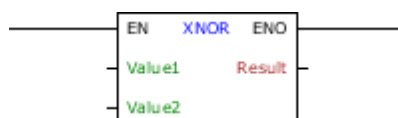


O exemplo acima realiza a operação ou exclusivo booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.3.5.1.4.5 XNOR

Bloco lógico que executa uma operação de “ou exclusivo negado” booleano entre duas variáveis, armazenando o resultado em uma terceira.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável para a operação
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

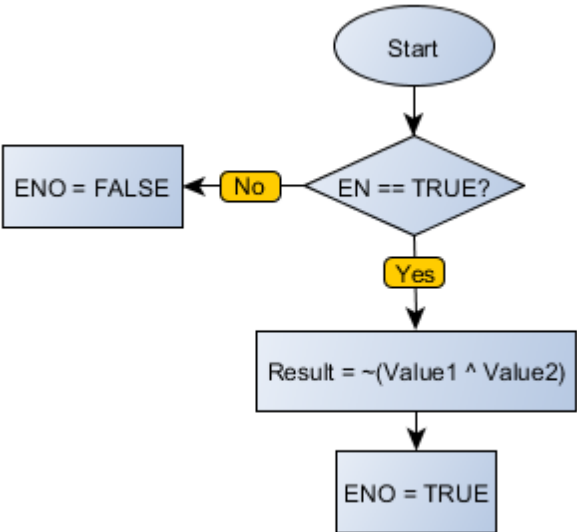
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da operação booleana “ou exclusivo negado” das variáveis de entrada Value1 e Value2.

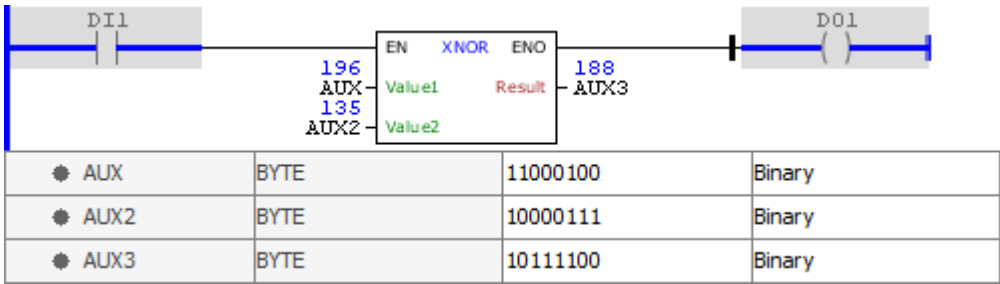
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a operação ou exclusivo negado booleano entre AUX e AUX2, armazenando o resultado em AUX3.

10.3.5.1.5 Logic Shift

10.3.5.1.5.1 ASHL

Bloco que executa uma operação de deslocamento binário à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	SINT INT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	SINT INT DINT	Variável que armazena o resultado da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos aritméticos à esquerda, de acordo com o valor de Shift.



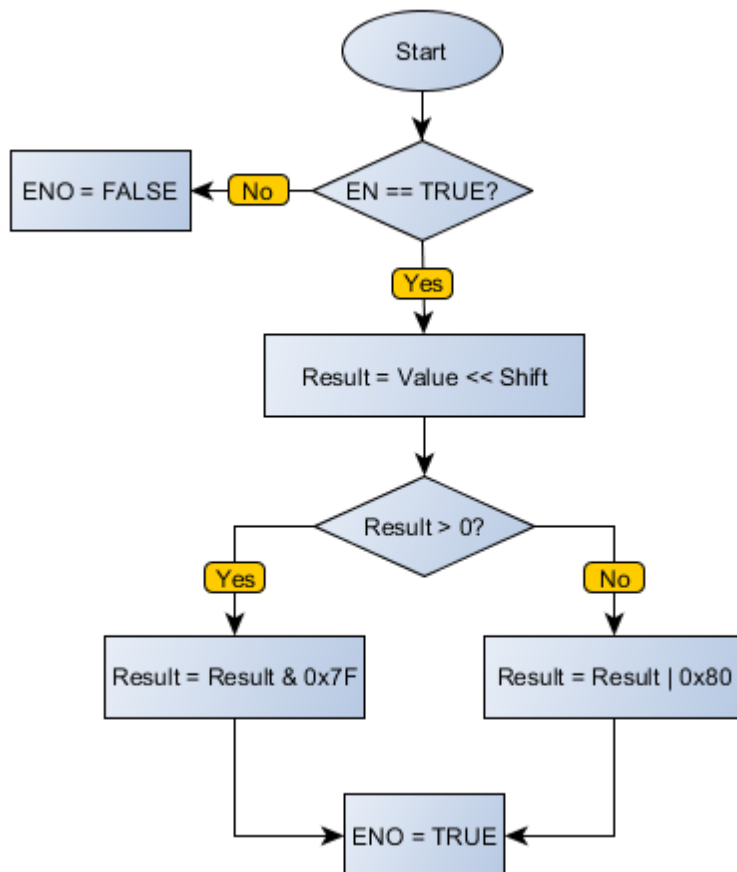
NOTA!

Todos os deslocamentos aritméticos implementados mantêm o sinal da variável

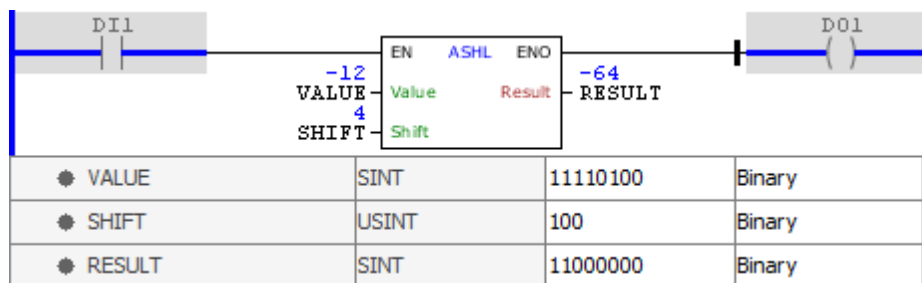
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

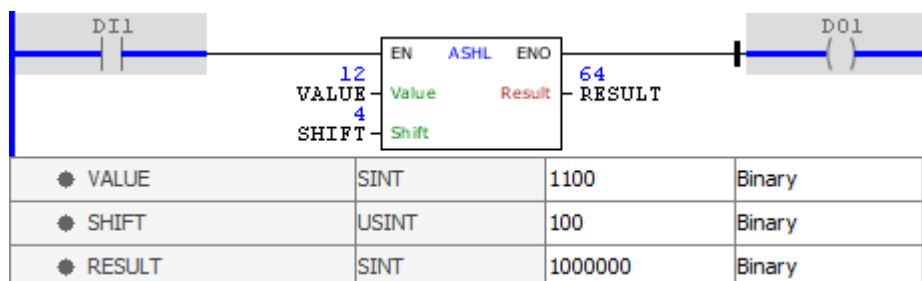
Fluxograma do Bloco



Exemplo



Descrição do exemplo.



Descrição do exemplo.

10.3.5.1.5.2 ASHR

Bloco que executa uma operação de deslocamento aritmético à direita em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	SINT INT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	SINT INT DINT	Variável que armazena o resultado da operação

Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos aritméticos à direita, de acordo com o valor de Shift.



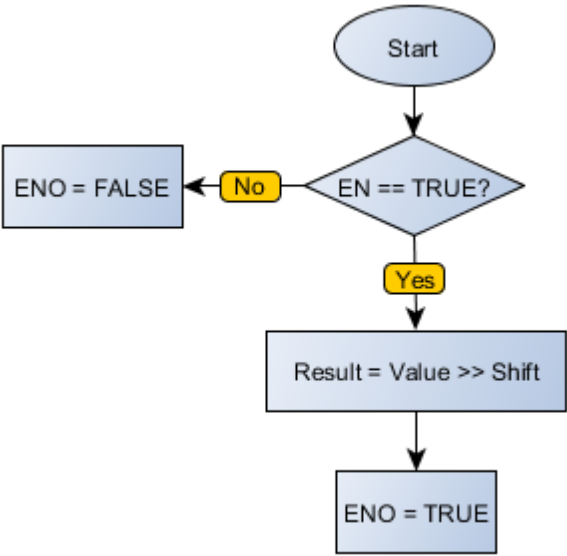
NOTA!

Todos os deslocamentos aritméticos implementados mantêm o sinal da variável

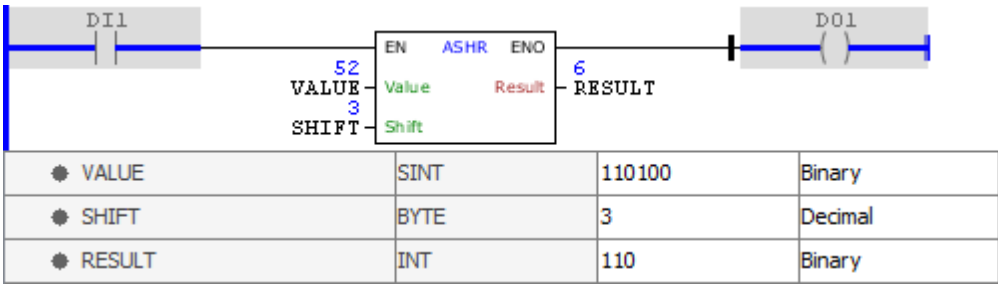
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

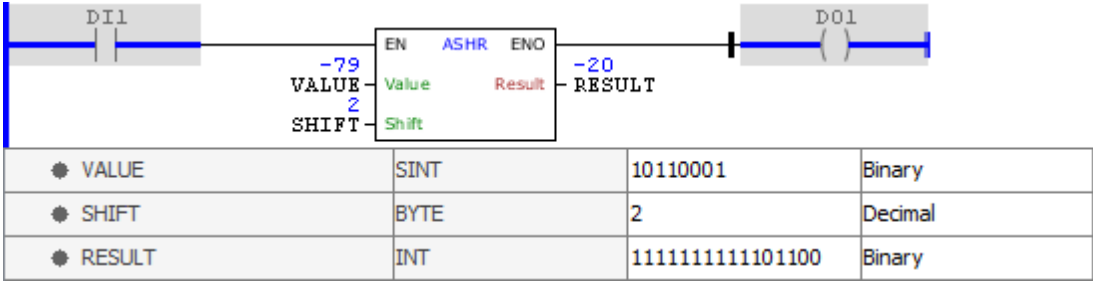
Fluxograma do Bloco



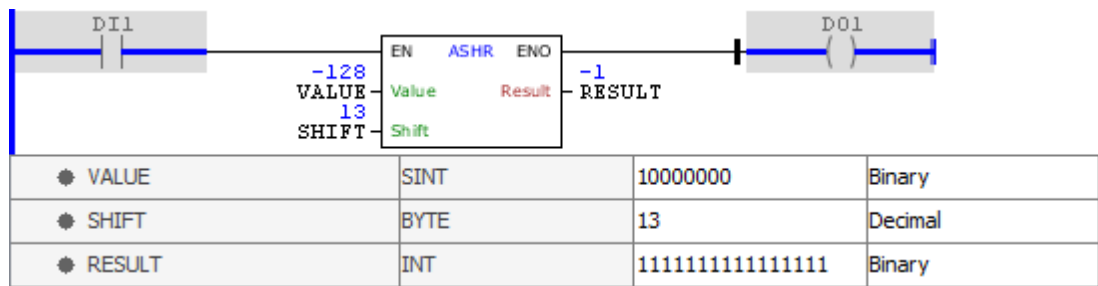
Exemplo



O exemplo acima realiza um deslocamento aritmético à direita de três posições na variável VALUE, cujo valor inicial é 52 (0011 0100 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0000 0110 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento aritmético à direita de duas posições na variável VALUE, cujo valor inicial é -79 (1011 0001 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos uns, visto que os deslocamentos aritméticos à direita preservam o sinal da variável. O resultado final (1111 0110 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento aritmético à direita de treze posições na variável VALUE, cujo valor inicial é -128 (1000 0000 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos uns. O resultado final (1111 1111 em binário) é armazenado em RESULT.

10.3.5.1.5.3 SHL

Bloco que executa uma operação de deslocamento lógico binário à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

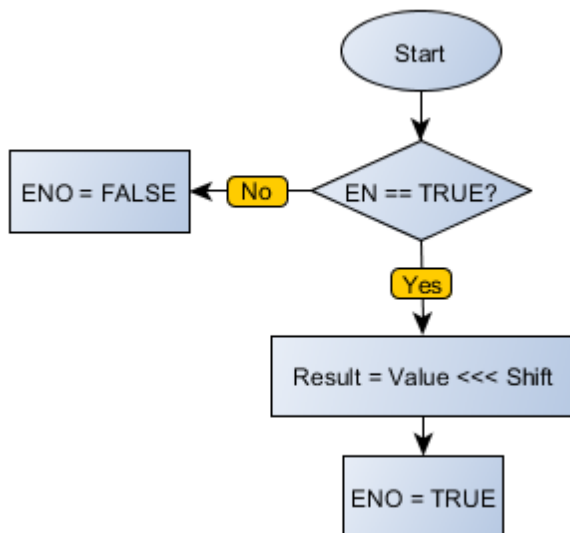
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à esquerda, de acordo com o valor de Shift.

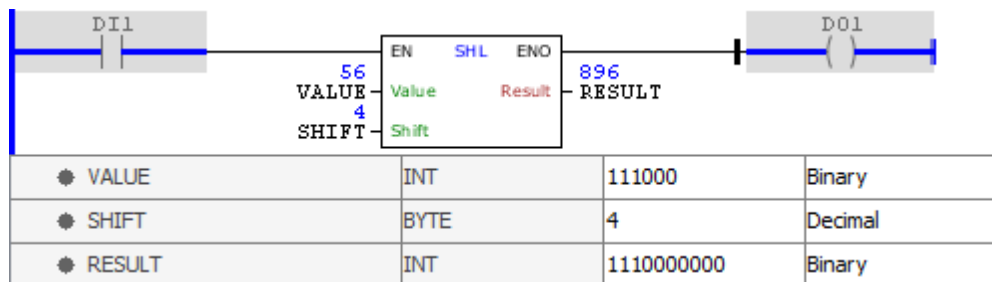
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

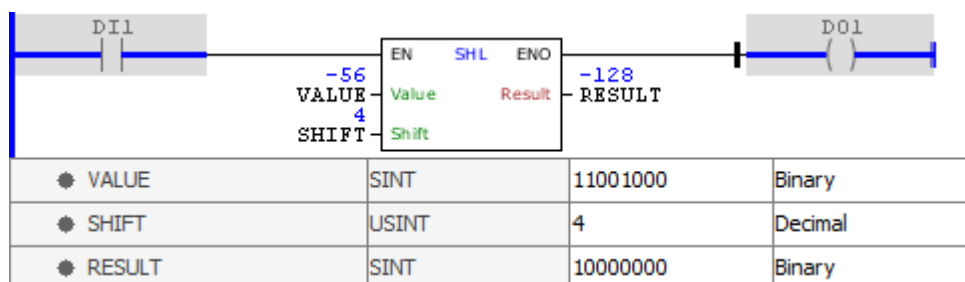
Fluxograma do Bloco



Exemplo



O exemplo acima realiza um deslocamento lógico à esquerda de quatro posições na variável VALUE, cujo valor inicial é 56 (0011 1000 em binário). Os bits à esquerda vão sendo descartados, e à direita são inseridos novos zeros. O resultado final (0011 1000 0000 em binário) é armazenado em RESULT.



O exemplo acima realiza um deslocamento lógico à esquerda de quatro posições na variável VALUE, cujo valor inicial é -56 (1100 1000 em binário). Os bits à esquerda vão sendo descartados, e à direita são inseridos novos zeros. O resultado final (1100 1000 0000 em binário) é armazenado em RESULT. Como RESULT é do tipo SINT, ela só aceita os primeiros oito bits (1000 0000).

10.3.5.1.5.4 SHR

Bloco que executa uma operação de deslocamento lógico binário à direita em um valor passado por

Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer deslocamento
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

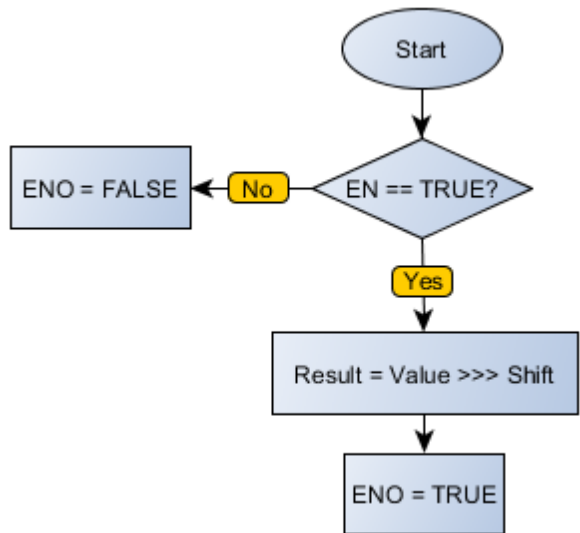
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à direita, de acordo com o valor de Shift.

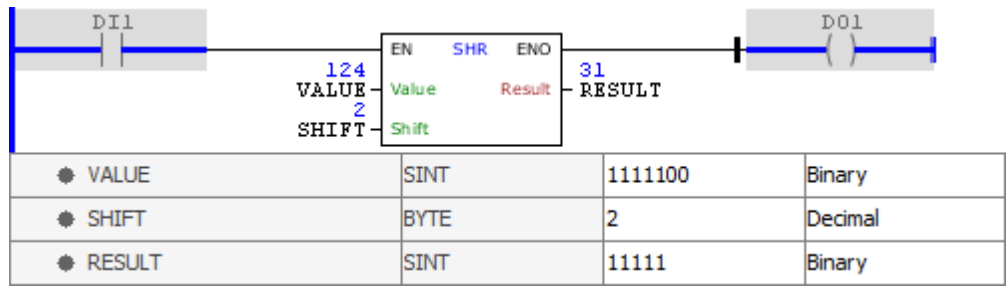
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

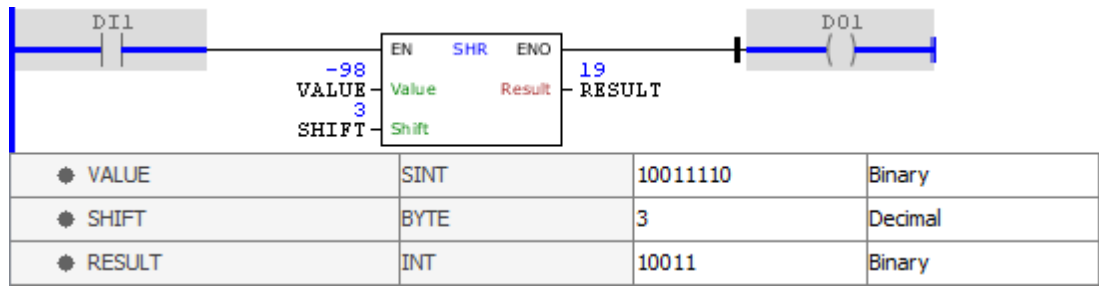
Fluxograma do Bloco



Exemplo



O exemplo acima realiza um deslocamento lógico à direita de duas posições na variável VALUE, cujo valor inicial é 124 (0111 1100 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0001 1111 em binário) é armazenado em RESULT.



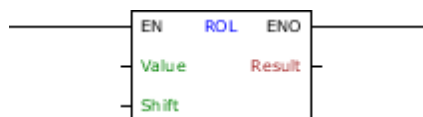
O exemplo acima realiza um deslocamento lógico à direita de três posições na variável VALUE, cujo valor inicial é -98 (1001 1110 em binário). Os bits à direita vão sendo descartados, e à esquerda são inseridos novos zeros. O resultado final (0001 0011 em binário) é armazenado em RESULT.

10.3.5.1.6 Logic Rotate

10.3.5.1.6.1 ROL

Bloco que executa uma operação de rotação lógica à esquerda em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer rotação
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

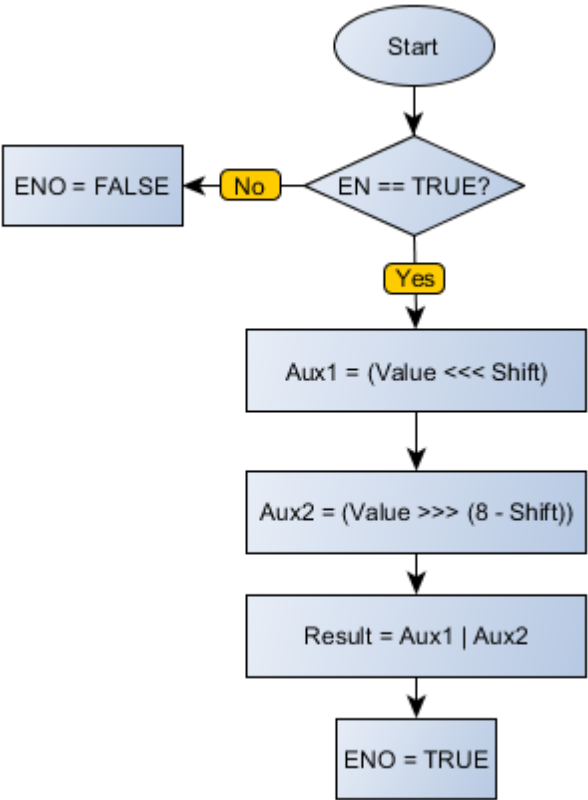
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à esquerda, de acordo com o valor de Shift. Os bits mais significativos que forem sendo descartados são reenviados aos bits menos significativos, caracterizando a rotação.

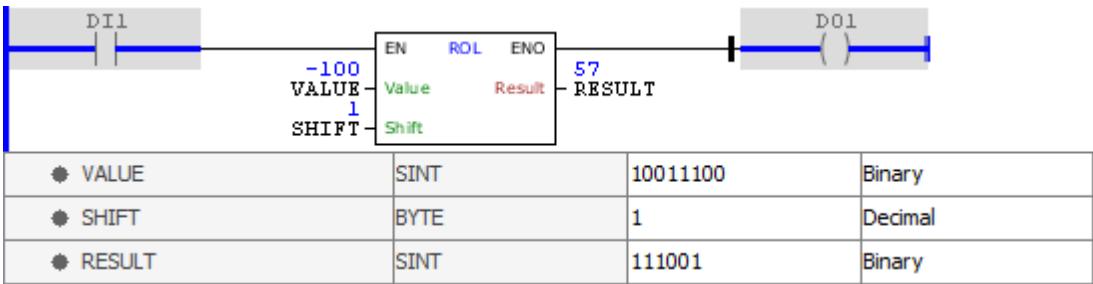
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

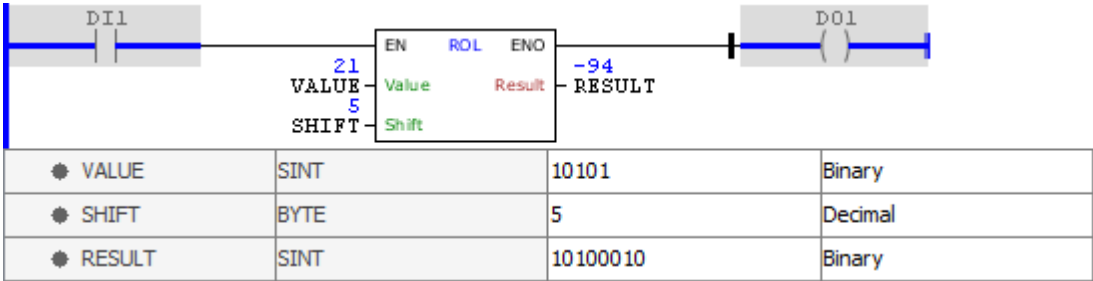
Fluxograma do Bloco



Exemplo



O exemplo acima realiza uma rotação lógica à esquerda de uma posição na variável VALUE, cujo valor inicial é -100 (1001 1100 em binário). Os bits descartados à esquerda são reinseridos à direita. O resultado final (0011 1001 em binário) é armazenado em RESULT.

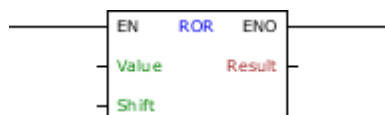


O exemplo acima realiza uma rotação lógica à esquerda de cinco posições na variável VALUE, cujo valor inicial é 21 (0001 0101 em binário). Os bits descartados à esquerda são reinseridos à direita. O resultado final (1010 0010 em binário) é armazenado em RESULT.

10.3.5.1.6.2 ROR

Bloco que executa uma operação de rotação lógica à direita em um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável a sofrer rotação
	Shift	BYTE USINT	Índice de deslocamento
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

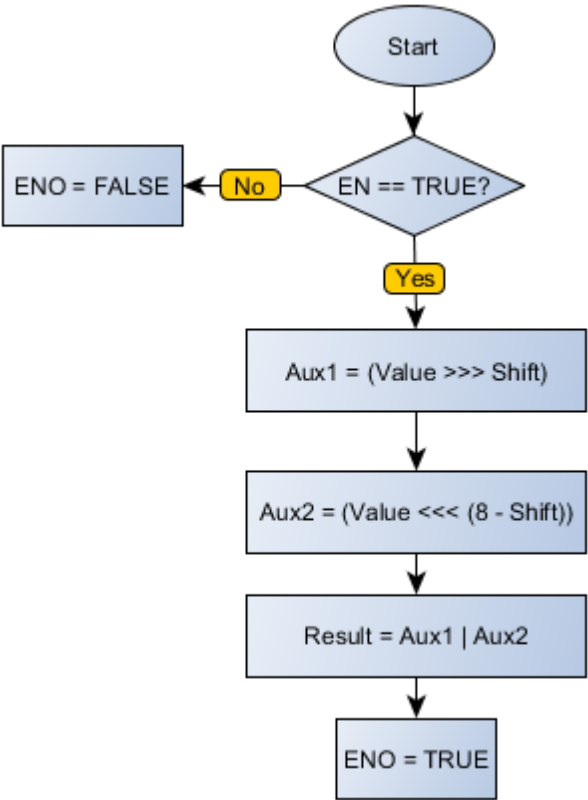
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da variável Value após realizar um determinado número de deslocamentos lógicos à direita, de acordo com o valor de Shift. Os bits menos significativos que forem sendo descartados são reenviados aos bits mais significativos, caracterizando a rotação.

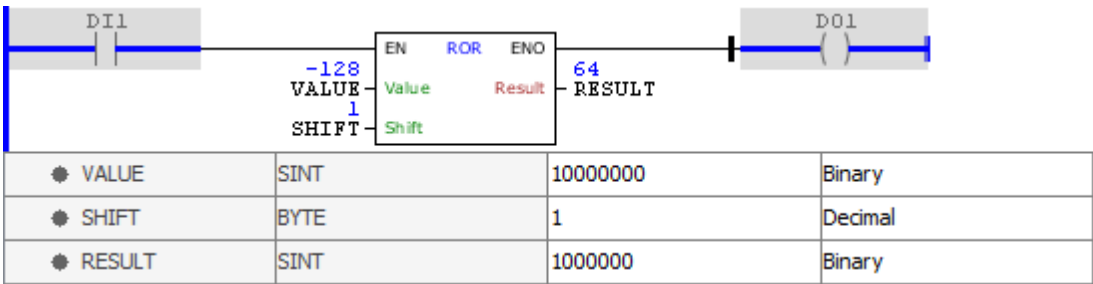
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

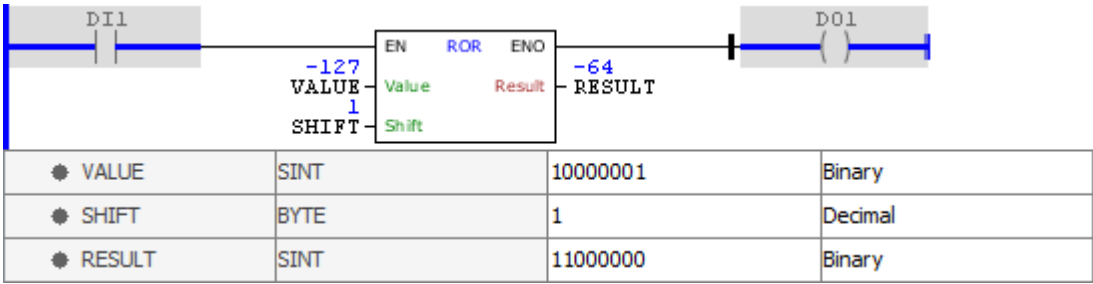
Fluxograma do Bloco



Exemplo



O exemplo acima realiza uma rotação lógica à direita de uma posição na variável VALUE, cujo valor inicial é -128 (1000 0000 em binário). Os bits descartados à direita são reinseridos à esquerda. O resultado final (0100 0000 em binário) é armazenado em RESULT. Observe que o sinal não é preservado nessa operação.



O exemplo acima realiza uma rotação lógica à direita de uma posição na variável VALUE, cujo valor inicial é -127 (1000 0001 em binário). Os bits descartados à direita são reinseridos à esquerda. O resultado final (1100 0000 em binário) é armazenado em RESULT.

10.3.5.1.7 Math Basic

10.3.5.1.7.1 ABS

Bloco que calcula o módulo de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

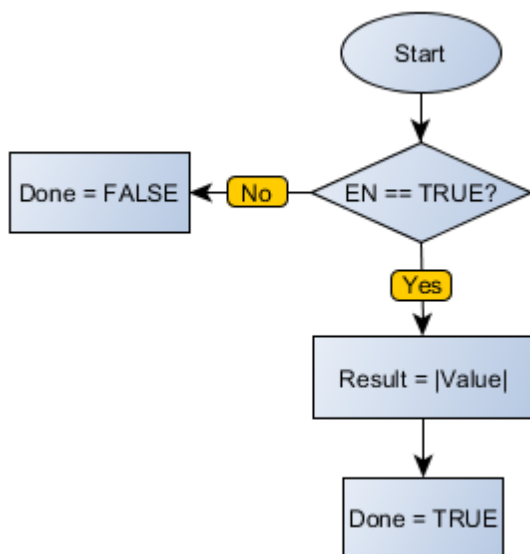
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

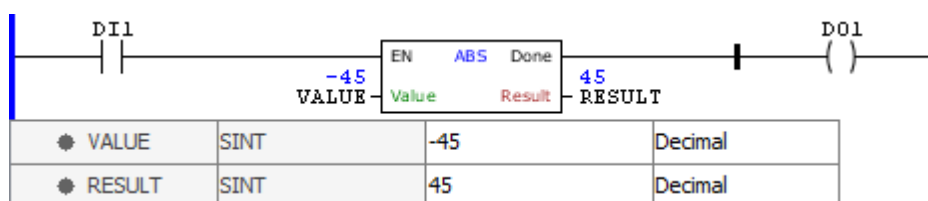
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor absoluto da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

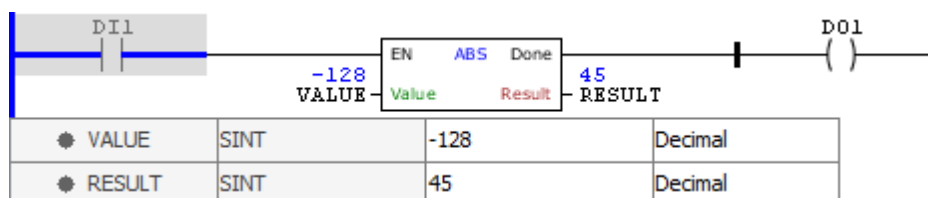
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o valor absoluto da variável VALUE, cujo valor inicial é -45, armazenando o resultado final, 45, em RESULT.

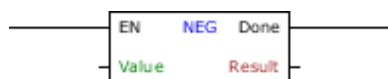


O exemplo acima calcula o valor absoluto da variável VALUE, cujo valor inicial é -45. O resultado final, 128, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.3.5.1.7.2 NEG

Bloco que calcula o oposto (ou seja, o produto com -1) de um valor passado por Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

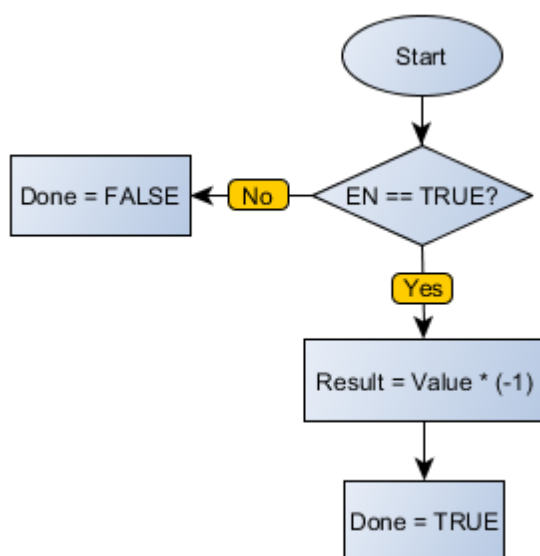
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

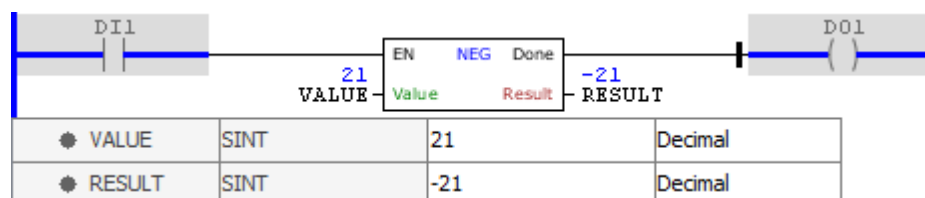
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o oposto da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

Fluxograma do Bloco

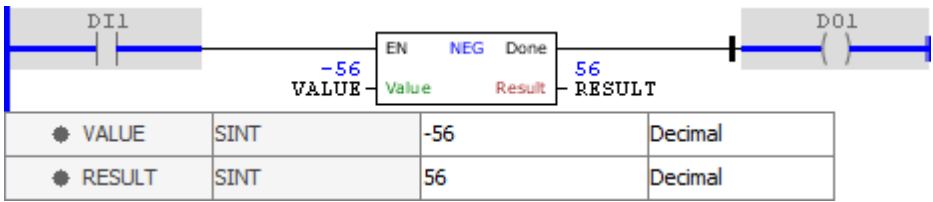


Exemplo

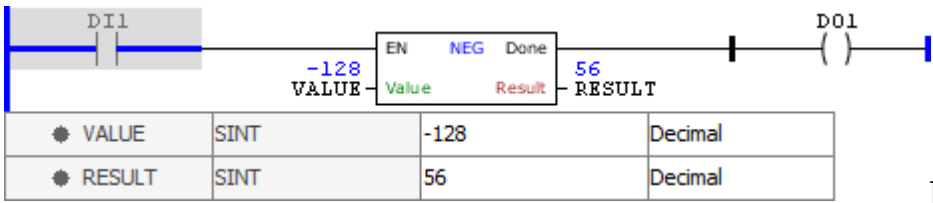


O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é 21, armazenando o

resultado final, -21, em RESULT.



O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é -56, armazenando o resultado final, 56, em RESULT.



O exemplo acima calcula o oposto da variável VALUE, cujo valor inicial é -128. O resultado final, 128, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.3.5.1.7.3 ADD

Bloco que calcula a soma dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeira parcela da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segunda parcela da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

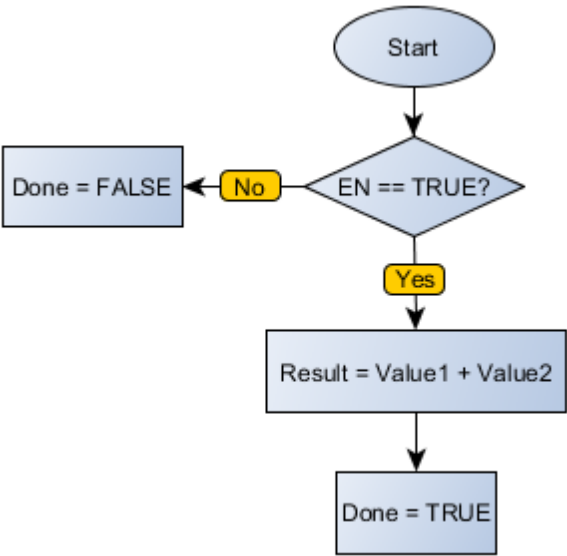
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a soma das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor

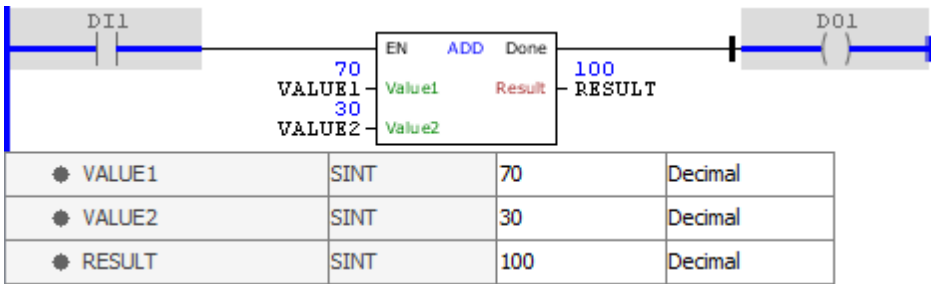
inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

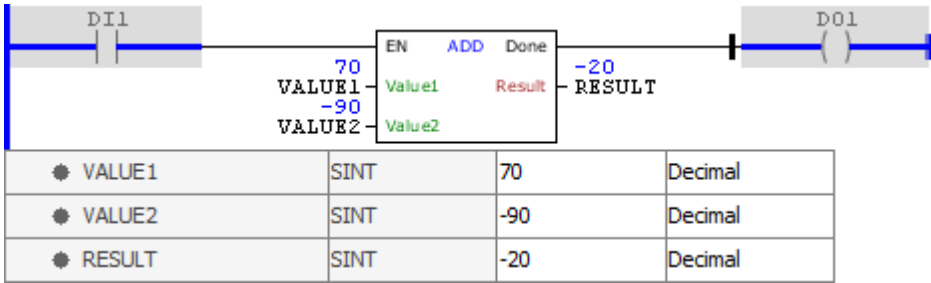
Fluxograma do Bloco



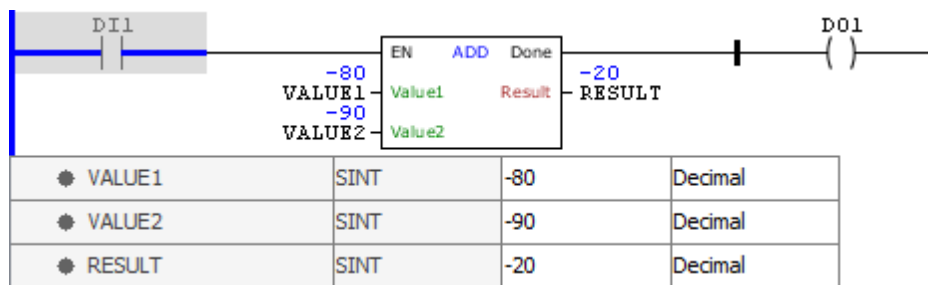
Exemplo



O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.

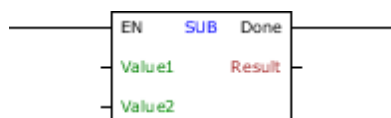


O exemplo acima calcula a soma das variáveis VALUE1 e VALUE2. O resultado final, -170, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.3.5.1.7.4 SUB

Bloco que calcula a subtração entre os valores Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

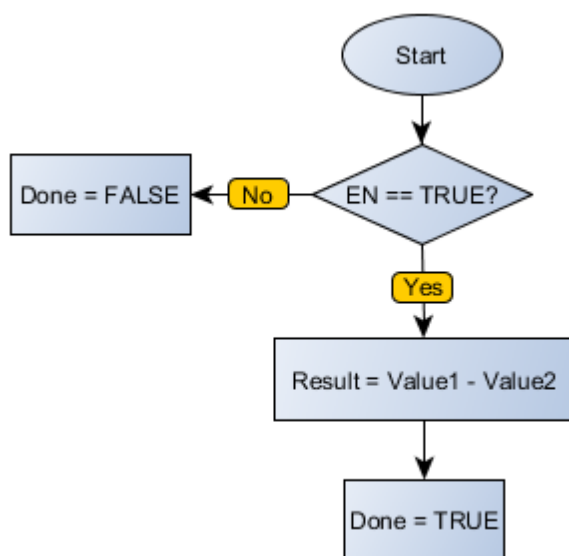
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Minuendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Subtraendo da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

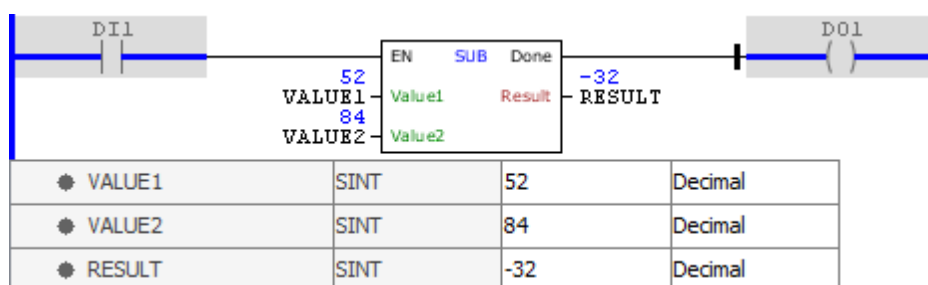
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a subtração das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

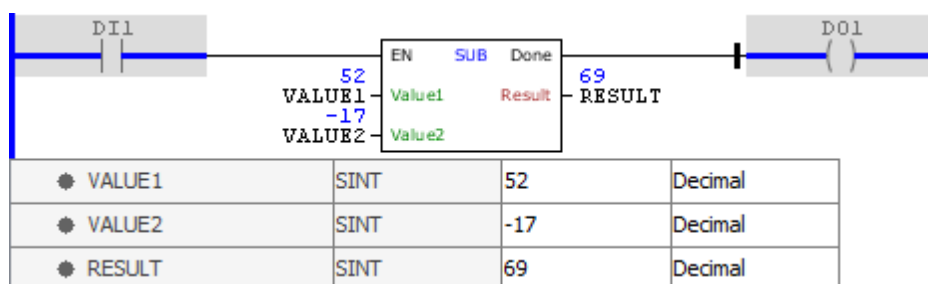
Fluxograma do Bloco



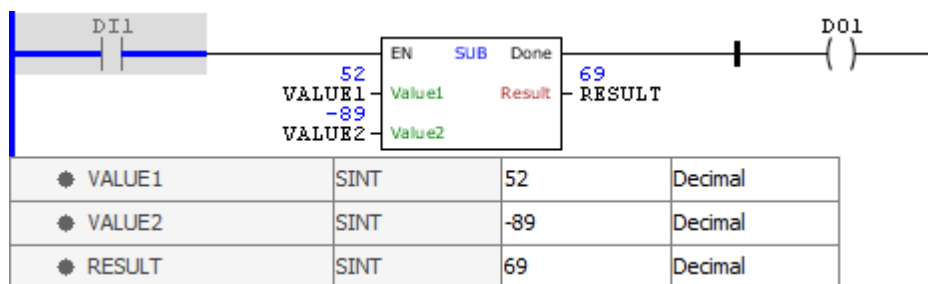
Exemplo



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.



O exemplo acima calcula a subtração das variáveis VALUE1 e VALUE2. O resultado final, 141, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.3.5.1.7.5 MUL

Bloco que calcula a multiplicação dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

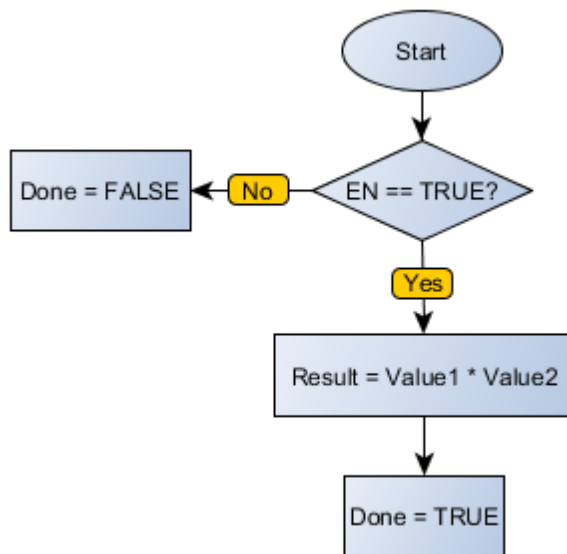
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro fator da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo fator da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

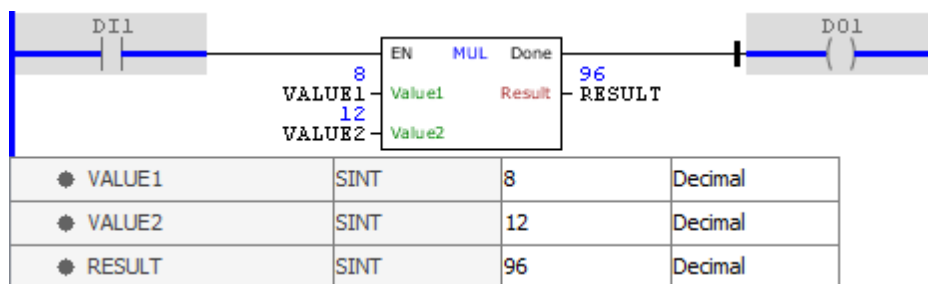
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a multiplicação das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

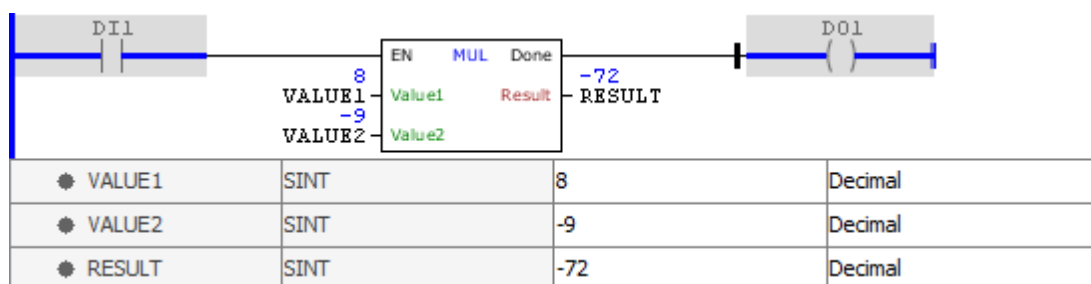
Fluxograma do Bloco



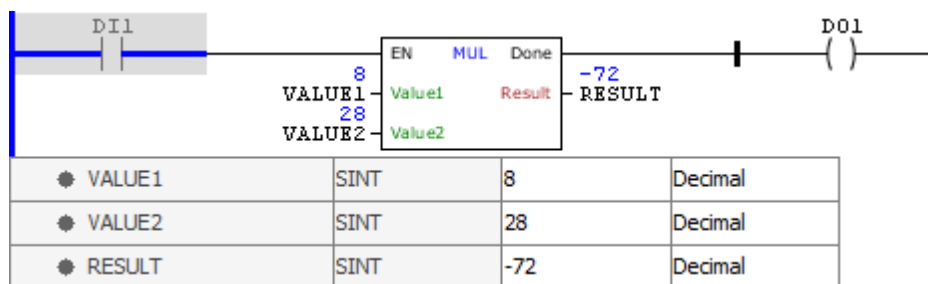
Exemplo



O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.

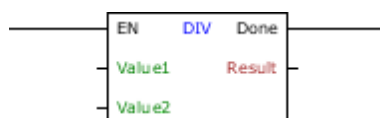


O exemplo acima calcula o produto das variáveis VALUE1 e VALUE2. O resultado final, 224, não pode ser armazenado em RESULT, pois ele está fora dos limites de valores aceitos pelo tipo SINT. Por isso, RESULT permanece inalterado e a saída fica desabilitada.

10.3.5.1.7.6 DIV

Bloco que calcula a divisão dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

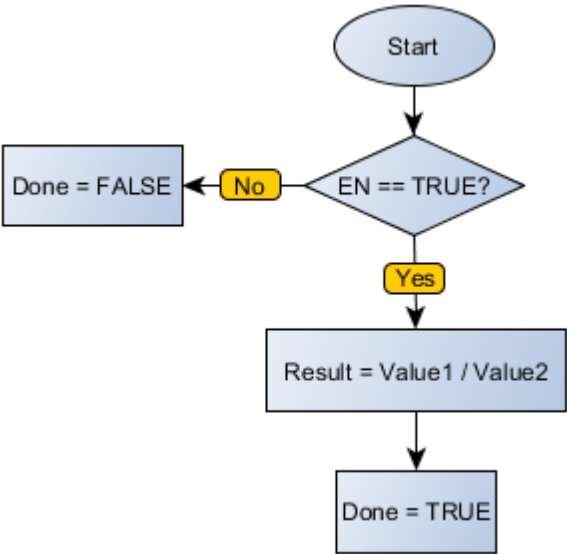
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Dividendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Divisor da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

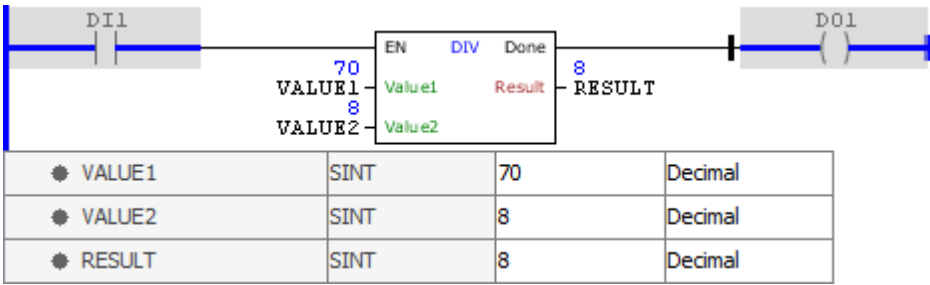
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a divisão das variáveis Value1 e Value2. O valor armazenado será a divisão exata caso Result seja REAL, ou, nos demais casos, apenas o quociente da divisão. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

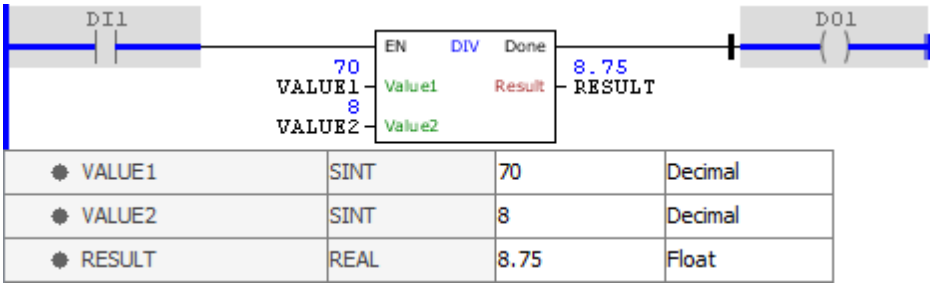
Fluxograma do Bloco



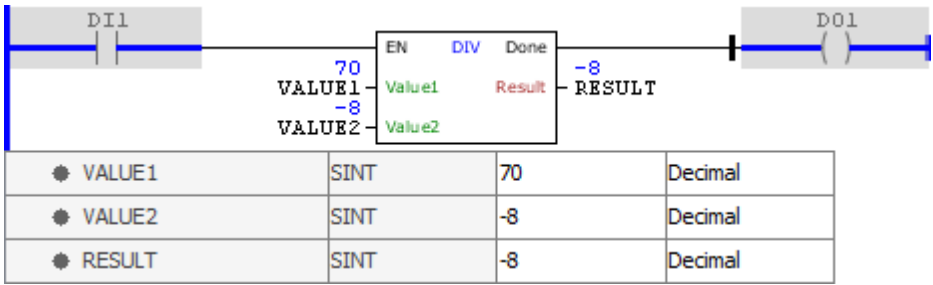
Exemplo



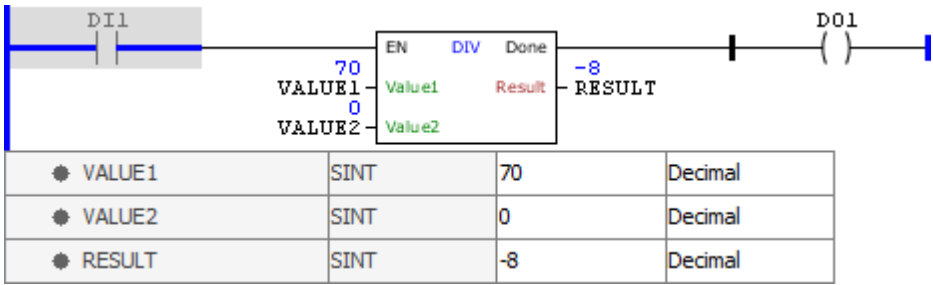
O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo SINT, apenas o quociente da divisão é armazenado nela.



O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo REAL, o valor exato da divisão é armazenado nela.



O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como RESULT é do tipo SINT, apenas o quociente da divisão é armazenado nela. Observe que o bloco aceita argumentos de ambos os sinais.



O exemplo acima calcula a divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como VALUE2 é zero, o bloco gera um erro de execução, RESULT permanece inalterado e a saída é desabilitada.

10.3.5.1.7.7 MOD

Bloco que calcula o resto da divisão dos valores de Value1 e Value2, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

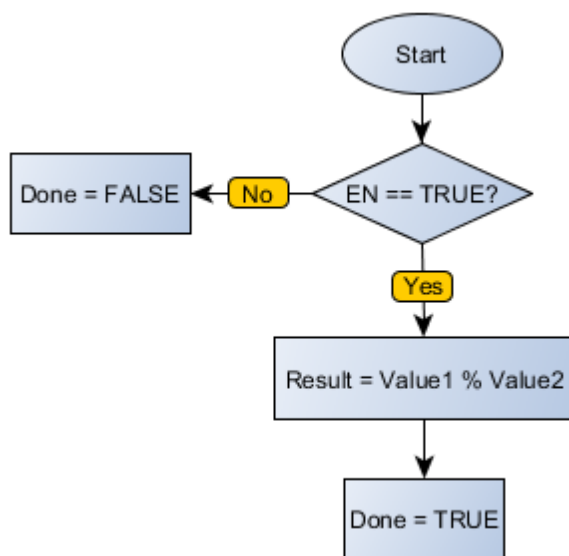
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Dividendo da operação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Divisor da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT	Variável que armazena o resultado da operação

Funcionamento

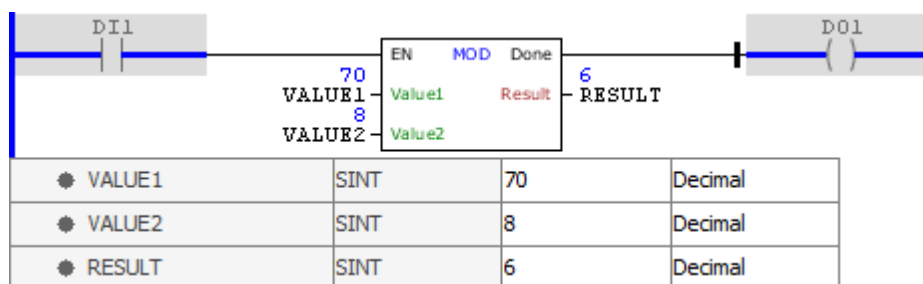
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o resto da divisão das variáveis Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

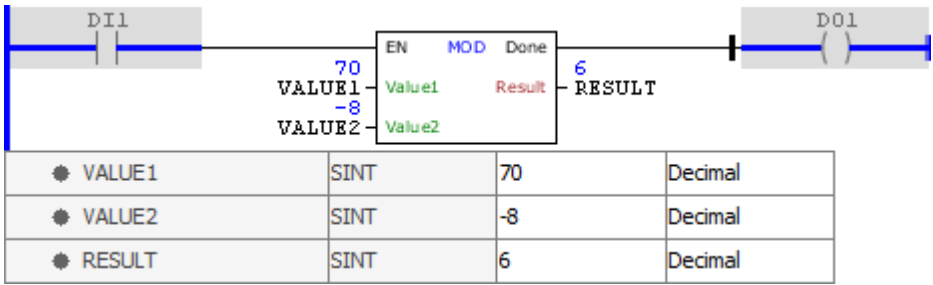
Fluxograma do Bloco



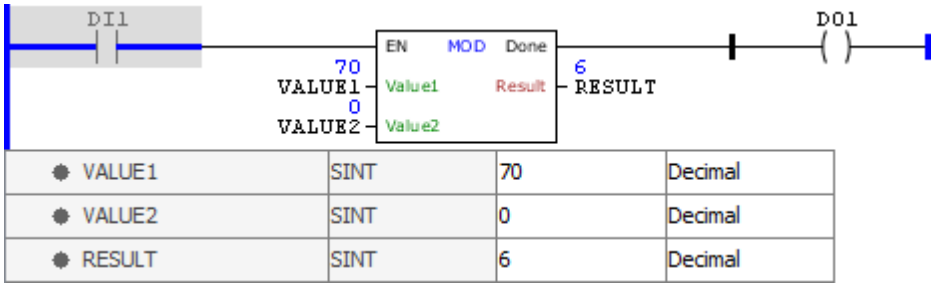
Exemplo



O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT.



O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Observe que o bloco aceita argumentos de ambos os sinais.



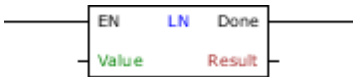
O exemplo acima calcula o resto da divisão das variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como VALUE2 é zero, o bloco gera um erro de execução, RESULT permanece inalterado e a saída é desabilitada.

10.3.5.1.8 Math Extended

10.3.5.1.8.1 LN

Bloco que calcula o logaritmo natural do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

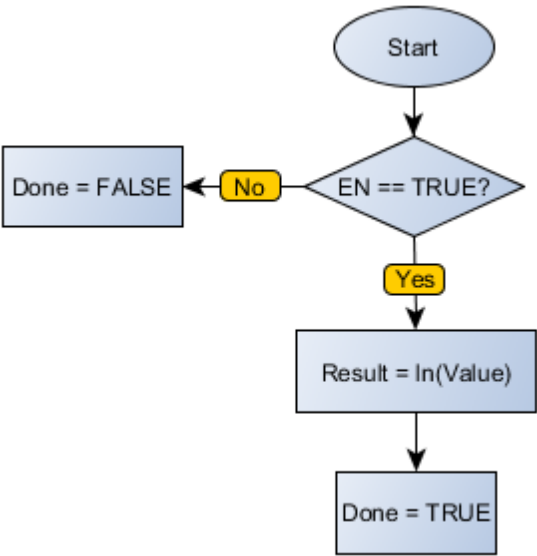
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o logaritmo natural da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor

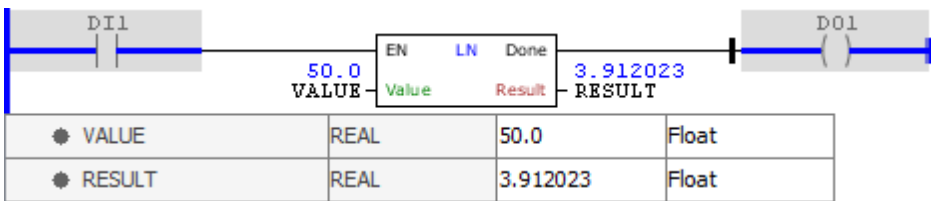
inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

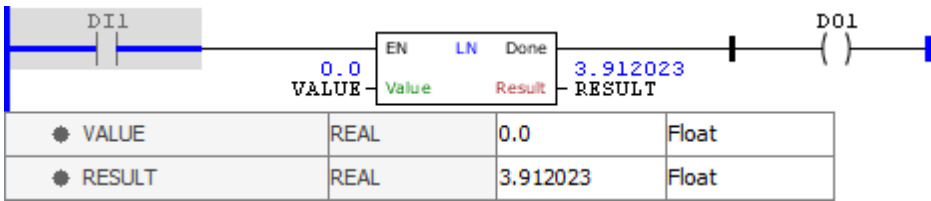
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o logaritmo natural da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



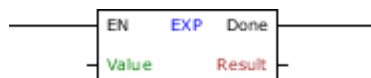
O exemplo acima calcula o logaritmo natural da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor zero, e a saída Done é desativada.

10.3.5.1.8.2 EXP

Bloco que calcula a exponencial do número de Euler “e” elevado ao valor de Value, armazenando o

resultado em Result.

Representação Ladder



Estrutura do Bloco

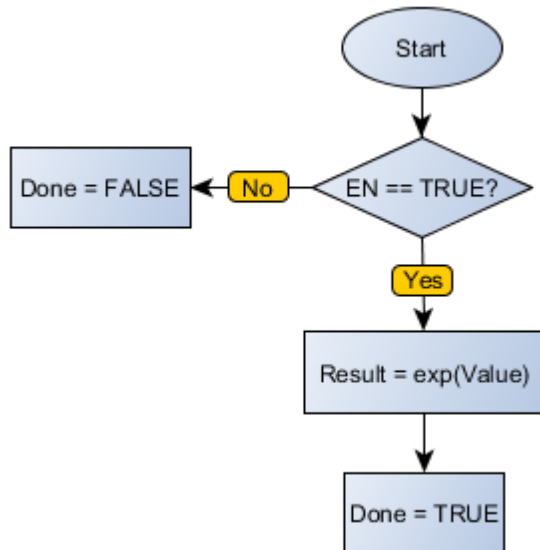
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

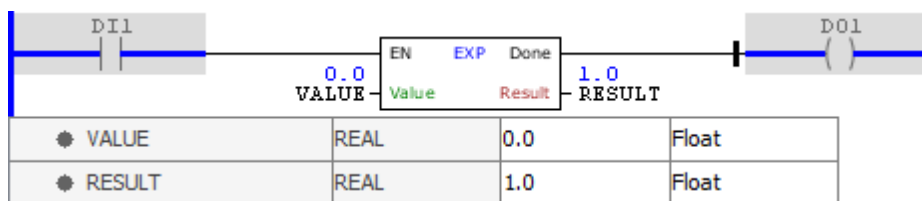
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a exponencial do número de Euler “e” elevado à variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

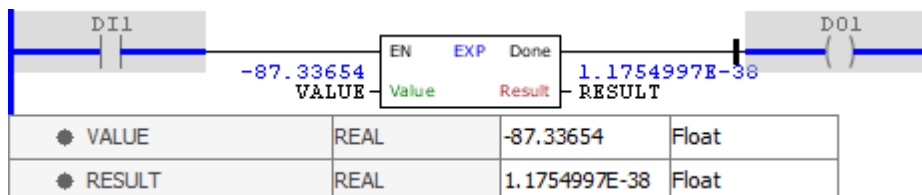
Fluxograma do Bloco



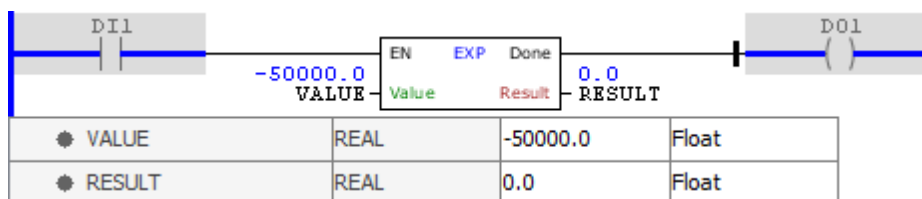
Exemplo



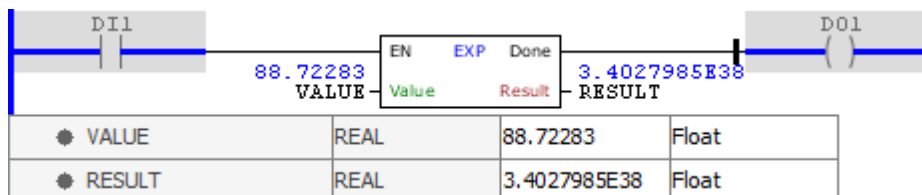
O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



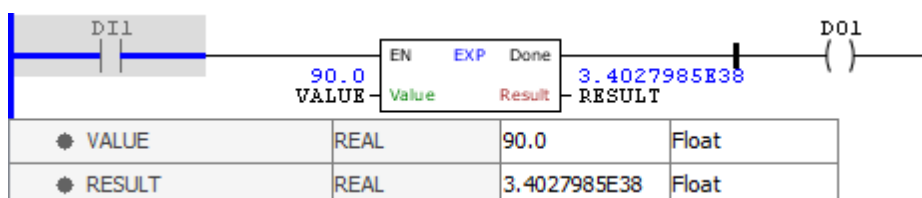
O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor mínimo de entrada para o qual o bloco retorna um resultado não nulo. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. Valores inferiores ao mínimo fazem com que o bloco retorne valor nulo. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor máximo de entrada para o qual o bloco retorna um resultado válido. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a exponencial da variável VALUE, armazenando o resultado final em RESULT. Valores superiores ao máximo fazem com que o bloco gere um erro, a saída RESULT permanece inalterada e a saída Done seja desativada.

10.3.5.1.8.3 LOG10

Bloco que calcula o logaritmo comum (base 10) do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

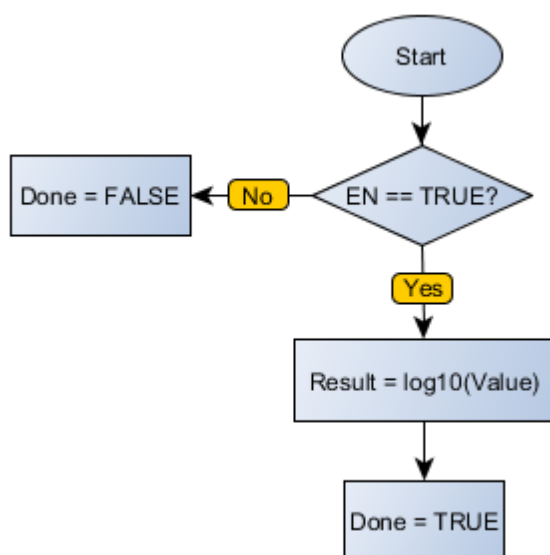
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

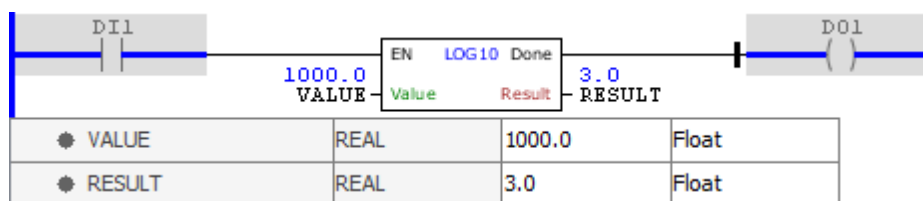
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o logaritmo comum da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

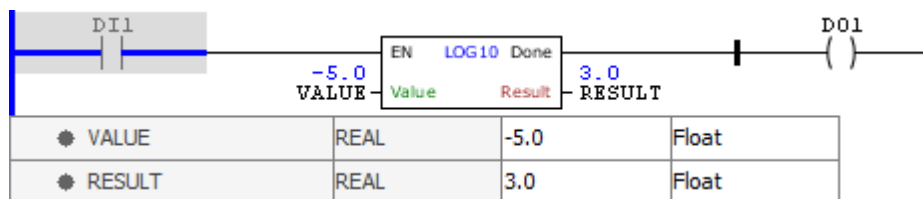
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o logaritmo comum da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o logaritmo comum da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor negativo, e a saída Done é desativada.

10.3.5.1.8.4 ALOG10

Bloco que calcula o antilogaritmo (exponencial com base 10) do valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

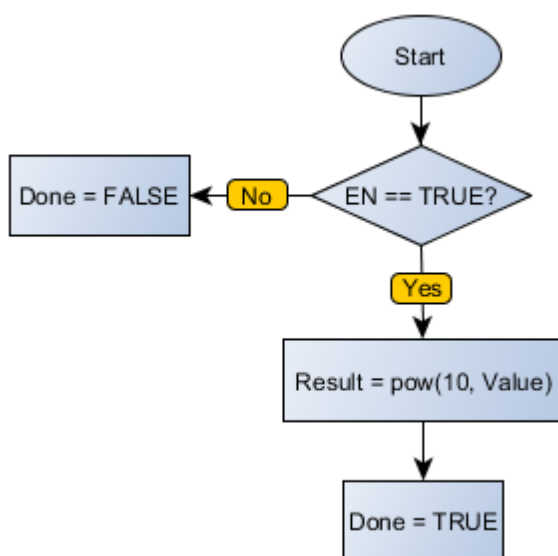
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

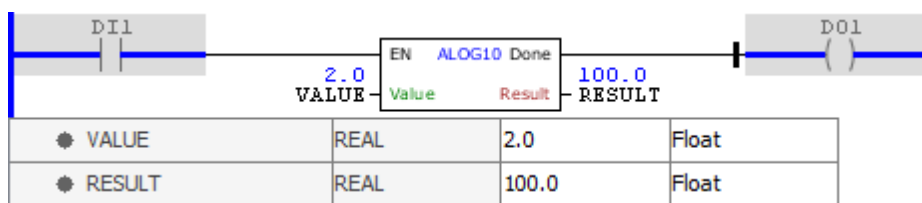
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o antilogaritmo da variável Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

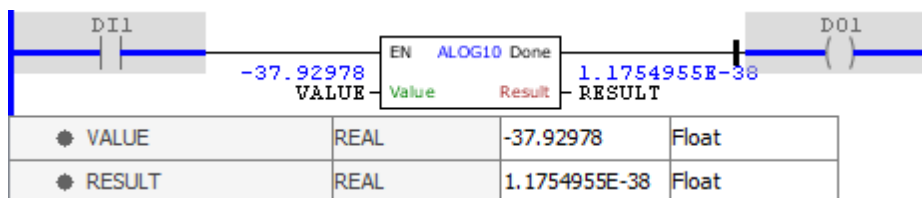
Fluxograma do Bloco



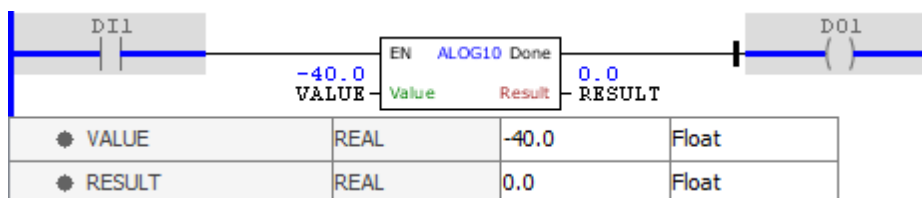
Exemplo



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.

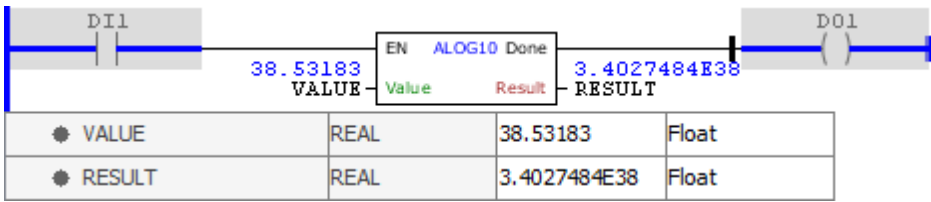


O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor mínimo de entrada para o qual o bloco retorna um resultado não nulo. O bloco finaliza com sucesso e a saída Done é ativada.

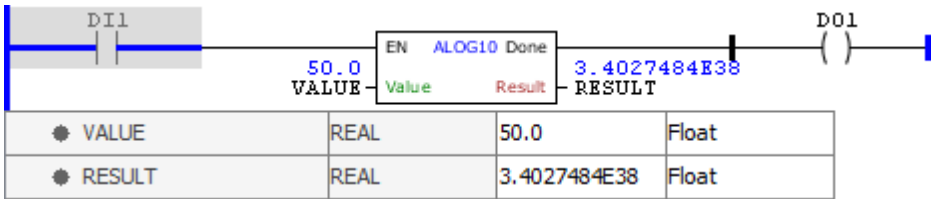


O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. Valores inferiores ao mínimo fazem com que o bloco retorne valor nulo. O bloco finaliza

com sucesso e a saída Done é ativada.



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. O valor indicado é o valor máximo de entrada para o qual o bloco retorna um resultado válido. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o antilogaritmo da variável VALUE, armazenando o resultado final em RESULT. Valores superiores ao máximo fazem com que o bloco gere um erro, a saída RESULT permanece inalterada e a saída Done seja desativada.

10.3.5.1.8.5 POW

Bloco que calcula o valor de Value elevado à potência Power, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Base da operação
	Power	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Potência da operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

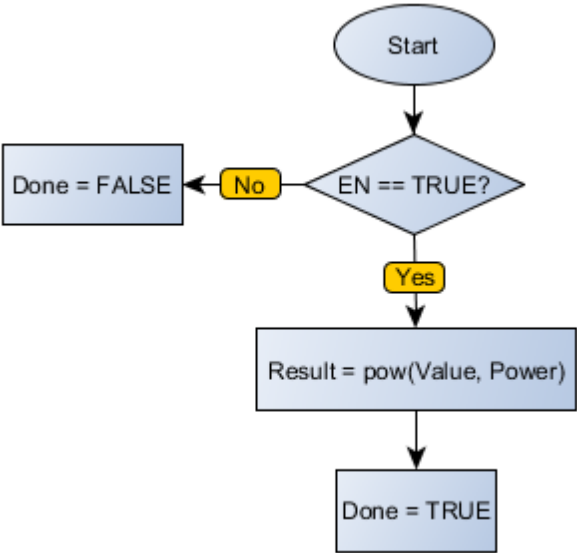
Funcionamento

Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor de Value elevado à potência Power. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação,

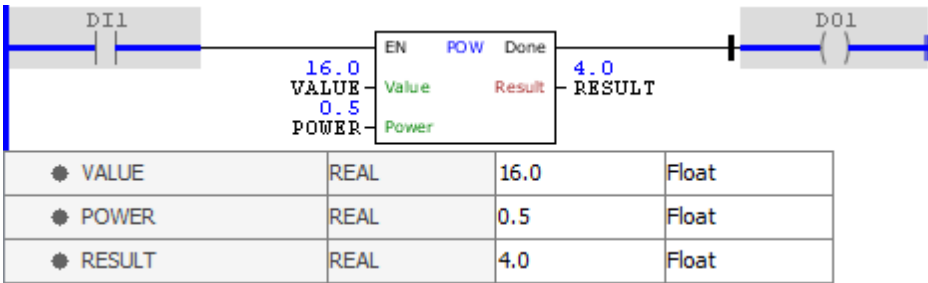
Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

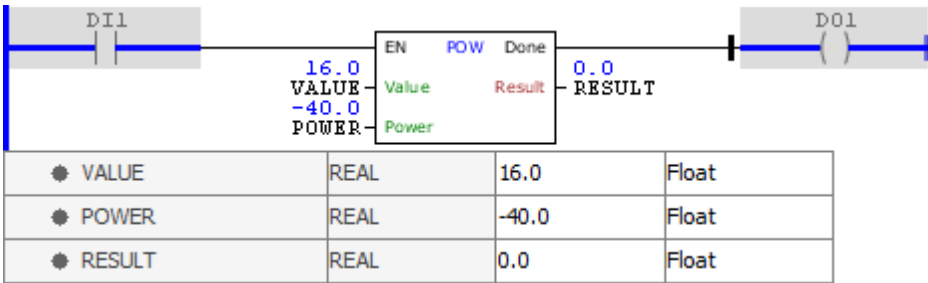
Fluxograma do Bloco



Exemplo

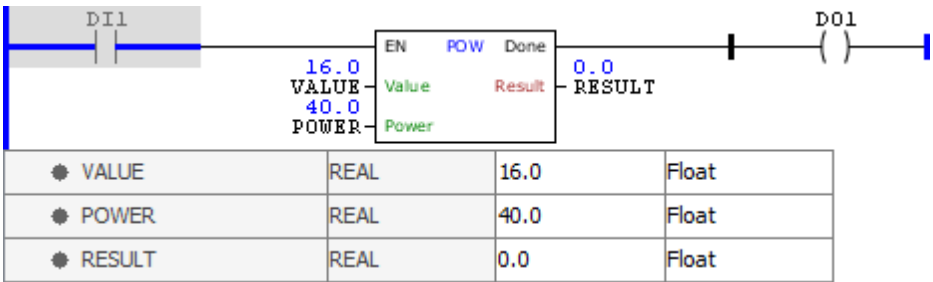


O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER,

armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da variável VALUE elevada à potência da variável POWER, armazenando o resultado final em RESULT. Como o resultado é superior ao valor máximo suportado pelo tipo REAL, o bloco gera um erro e a saída Done é desativada.

10.3.5.1.8.6 Sqrt

Bloco que calcula o valor da raiz quadrada de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

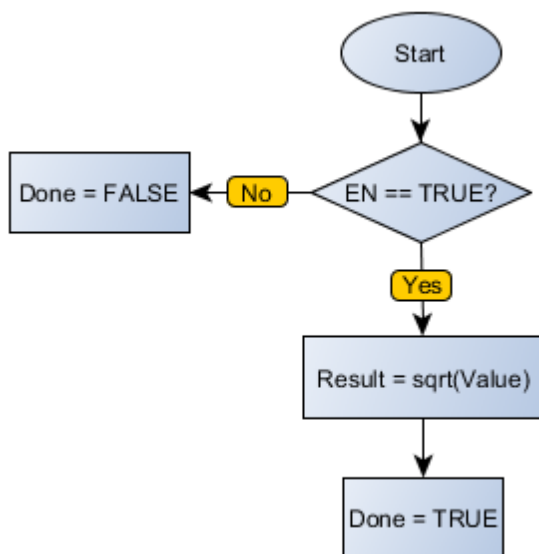
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Variável de referência para a operação
VAR_OUTPUT	Done	B00L	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

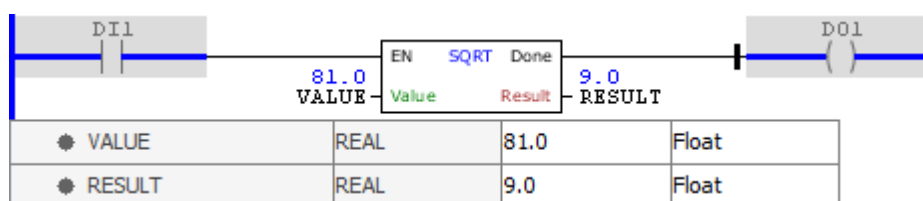
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor da raiz quadrada de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

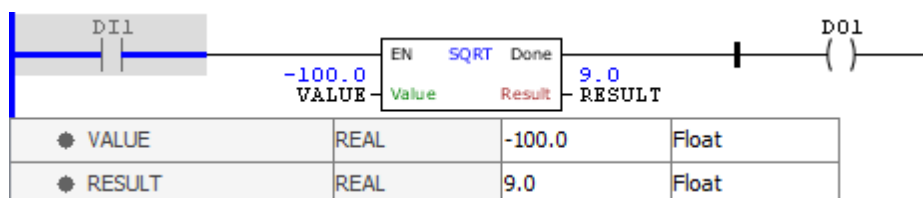
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o valor da raiz quadrada da variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor da raiz quadrada da variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor negativo, e a saída Done é desativada.

10.3.5.1.8.7 ROUND

Bloco que arredonda o valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

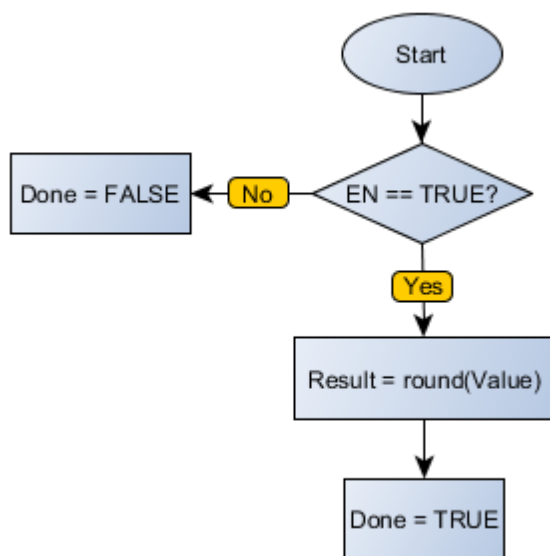
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor arredondado de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

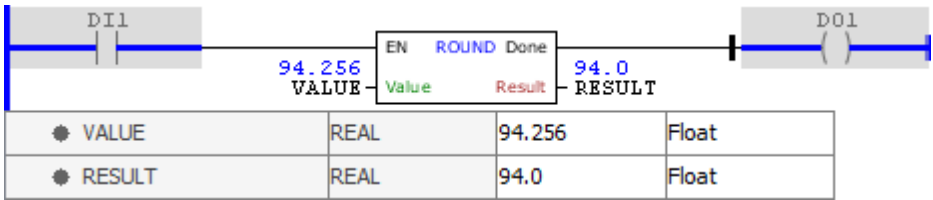
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior
SCA06	2.00 ou superior

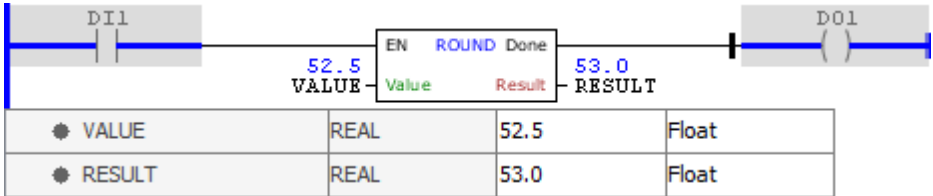
Fluxograma do Bloco



Exemplo



O exemplo acima arredonda o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais inferiores a 0,5 são descartados. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima arredonda o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais iguais ou superiores a 0,5 promovem a unidade ao valor imediatamente acima. O bloco finaliza com sucesso e a saída Done é ativada.

10.3.5.1.8.8 TRUNC

Bloco que trunca o valor de Value, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável de referência para a operação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Variável que armazena o resultado da operação

Funcionamento

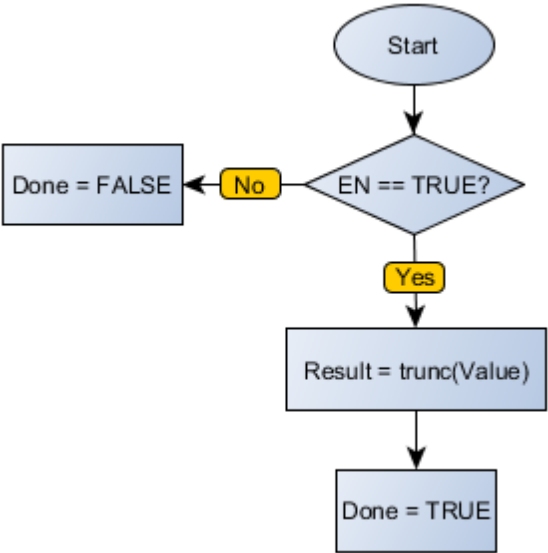
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o valor truncado de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

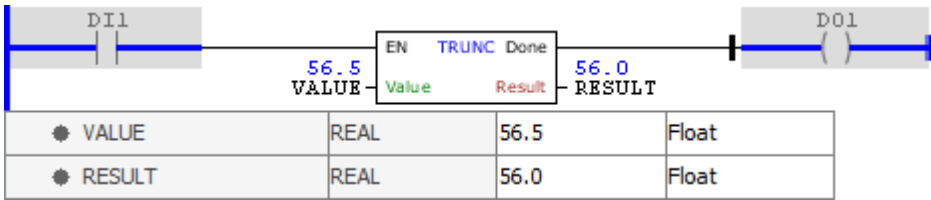
Compatibilidade

Equipamento	Versão
PLC300	2.10 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



O exemplo acima trunca o valor da variável VALUE, armazenando o resultado final em RESULT. Decimais são descartados. O bloco finaliza com sucesso e a saída Done é ativada.

10.3.5.1.9 Math Trigonometry

10.3.5.1.9.1 SIN

Bloco que calcula o seno de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

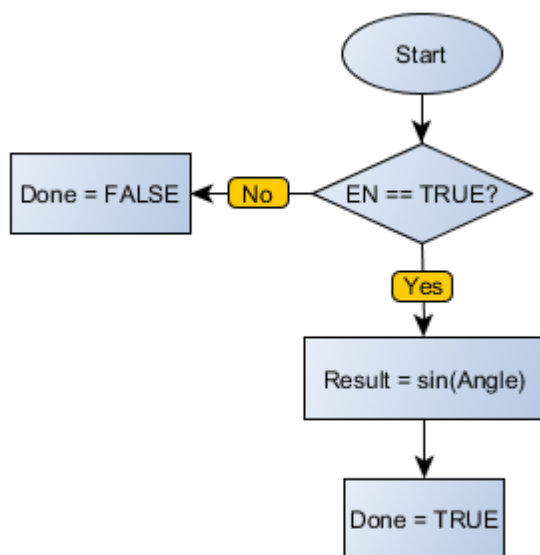
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

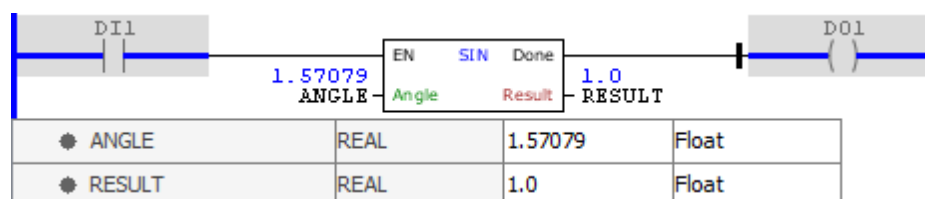
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o seno de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

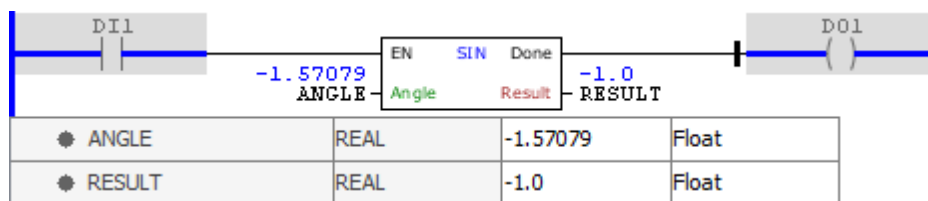
Fluxograma do Bloco



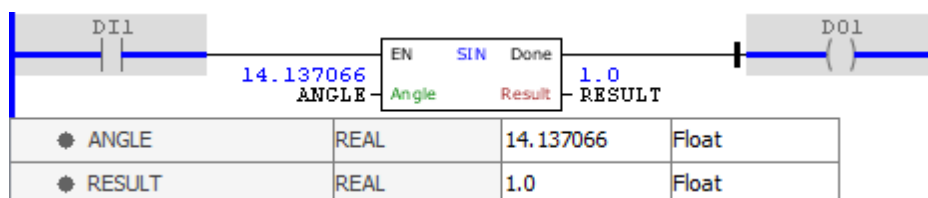
Exemplo



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores negativos de entrada.



O exemplo acima calcula o seno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores superiores a uma volta completa.

10.3.5.1.9.2 COS

Bloco que calcula o cosseno de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

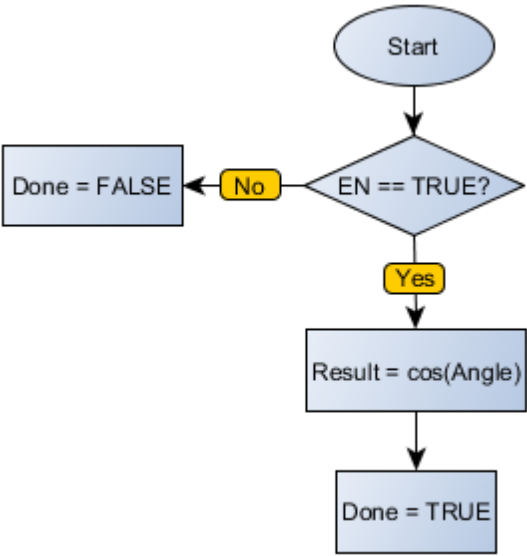
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

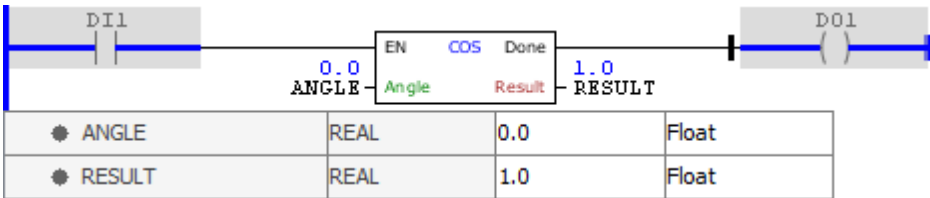
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o cosseno de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

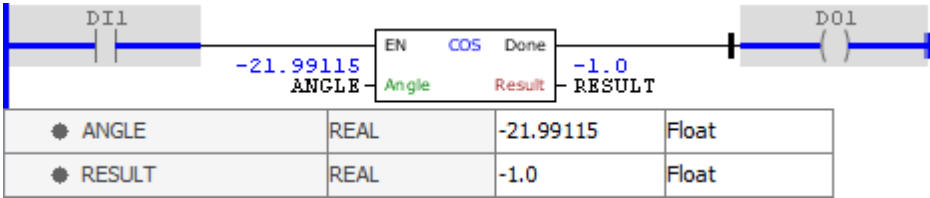
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o cosseno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o cosseno da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores de entrada negativos e superiores a uma volta.

10.3.5.1.9.3 TAN

Bloco que calcula a tangente de Angle, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

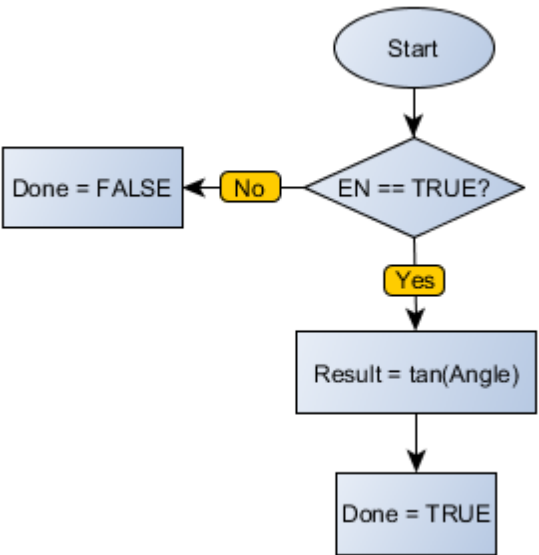
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Angle	REAL	Ângulo (em radianos)
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	REAL	Variável que armazena o resultado da operação

Funcionamento

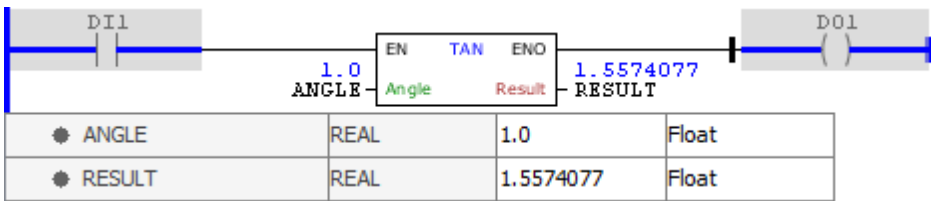
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result a tangente de Angle. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

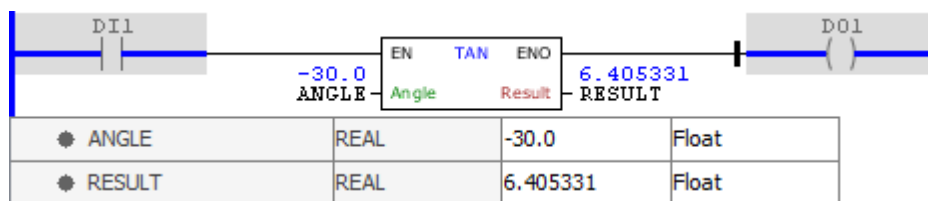
Fluxograma do Bloco



Exemplo



O exemplo acima calcula a tangente da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula a tangente da variável VALUE, interpretada em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que o bloco aceita valores de entrada negativos e superiores a uma volta.

10.3.5.1.9.4 ASIN

Bloco que calcula o arco seno de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

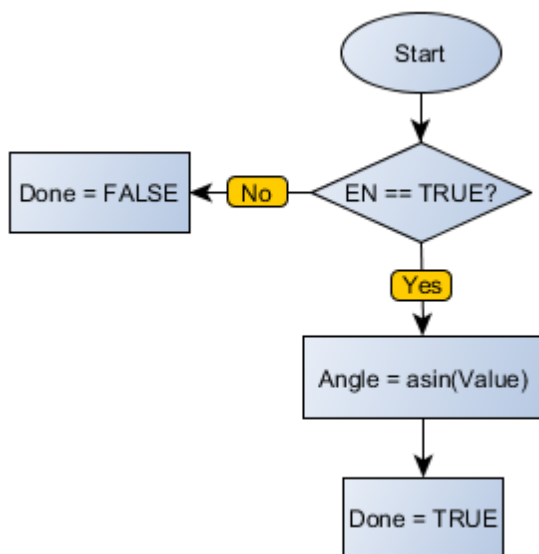
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor do seno
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cujo seno é igual a Value (em radianos)

Funcionamento

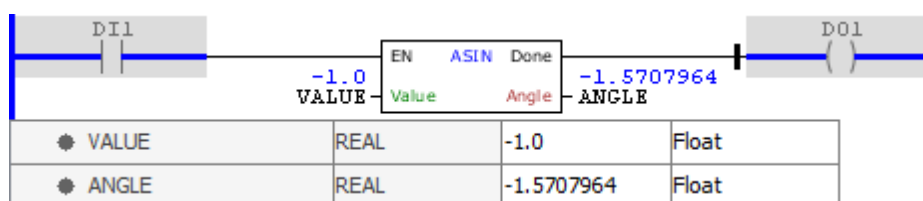
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco seno de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

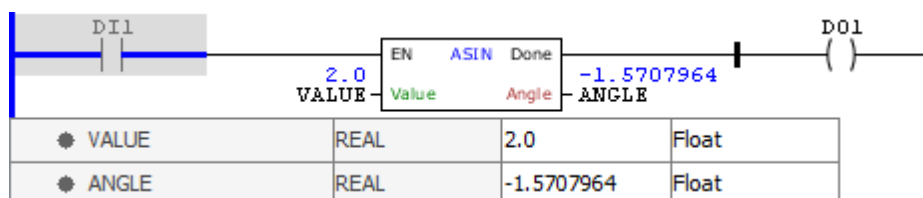
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cujo seno é a variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cujo seno é a variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor superior a 1, e a saída Done é desativada.

10.3.5.1.9.5 ACOS

Bloco que calcula o arco cosseno de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

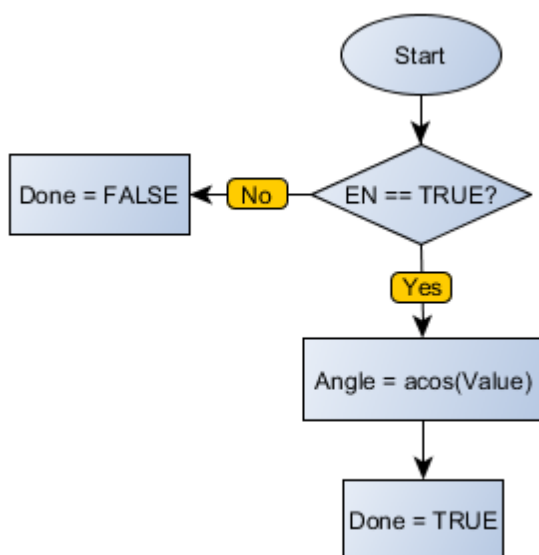
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor do cosseno
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cujo cosseno é igual a Value (em radianos)

Funcionamento

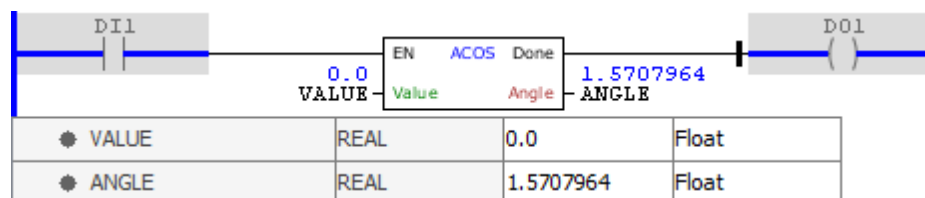
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco cosseno de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

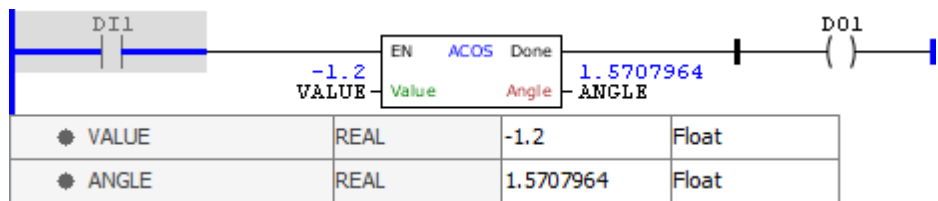
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cujo cosseno é a variável VALUE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cujo cosseno é a variável VALUE, armazenando o resultado final em RESULT. O bloco gera um erro de execução, visto que VALUE possui valor inferior a 1, e a saída Done é desativada.

10.3.5.1.9.6 ATAN

Bloco que calcula o arco tangente de Value, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

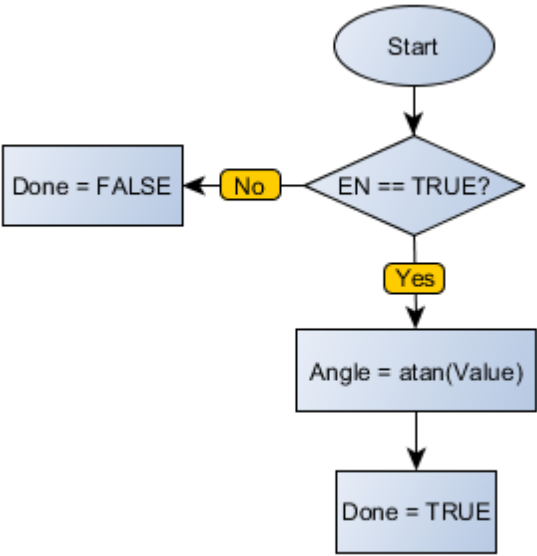
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor da tangente
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cuja tangente é igual a Value (em radianos)

Funcionamento

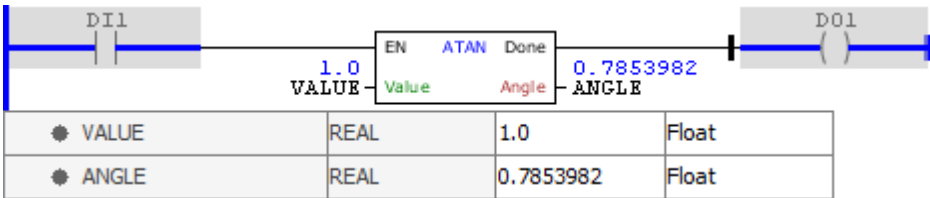
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco tangente de Value. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

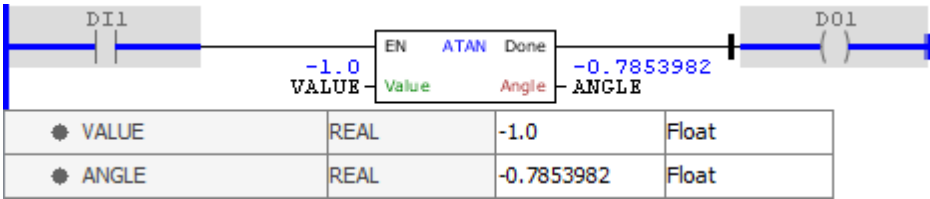
Fluxograma do Bloco



Exemplo



O exemplo acima calcula o arco, em radianos, cuja tangente é a variável VALUE, armazenando o resultado final em RESULT. O arco, para valores positivos, sempre se encontra no primeiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é a variável VALUE, armazenando o resultado final em RESULT. O arco, para valores negativos, sempre se encontra no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

10.3.5.1.9.7 ATAN2

Bloco que calcula o arco tangente de Y/X, armazenando o resultado em Angle.

Representação Ladder



Estrutura do Bloco

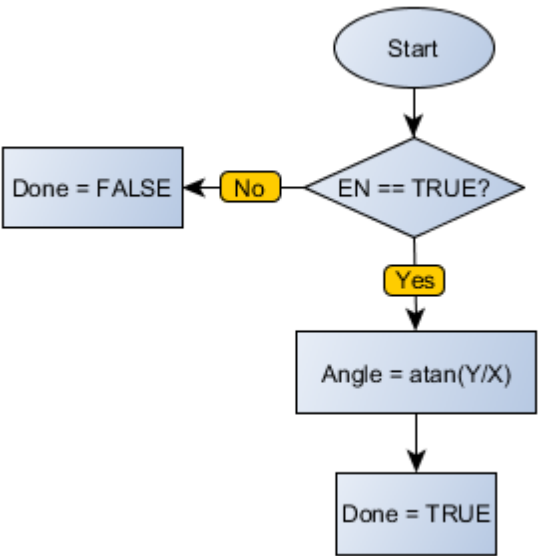
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	X	REAL	Parâmetro X da função
	Y	REAL	Parâmetro Y da função
VAR_OUTPUT	Done	BOOL	Fim de operação
	Angle	REAL	Valor do ângulo cuja tangente é igual a (Y/X) (em radianos)

Funcionamento

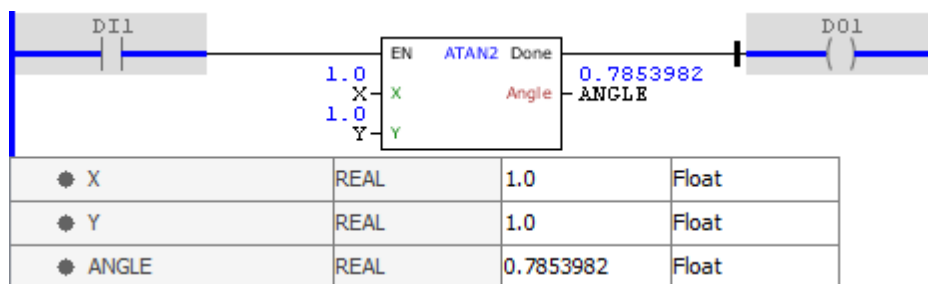
Este bloco, quando tem um valor TRUE em EN, envia para a saída Angle o arco tangente de Y/X. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Angle permanece com seu valor inalterado.

Quando EN possui valor FALSE, Angle permanece inalterado e Done permanece em FALSE.

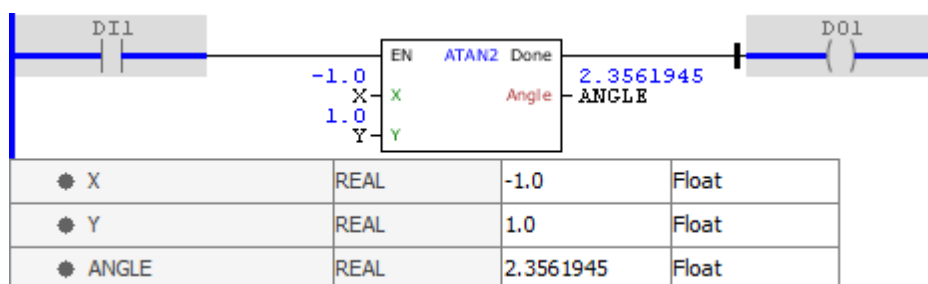
Fluxograma do Bloco



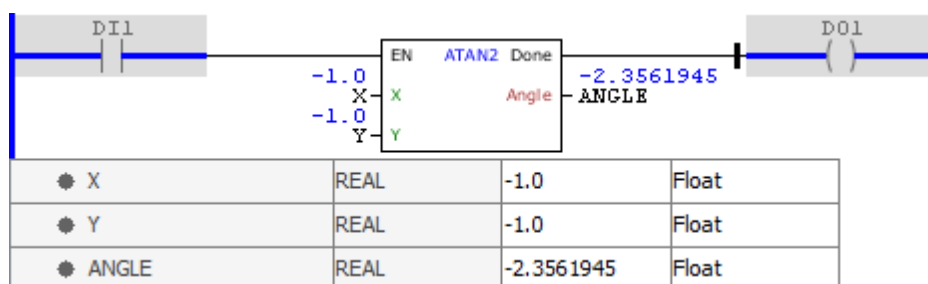
Exemplo



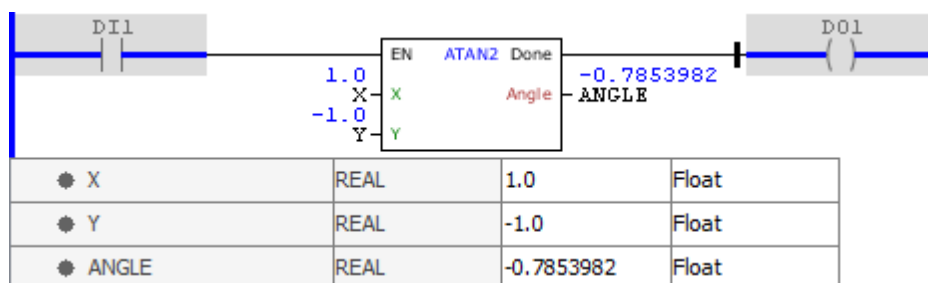
O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores positivos de X e Y, sempre se encontra no primeiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores negativos de X e positivos de Y, sempre se encontra no segundo quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores negativos de X e Y, sempre se encontra no terceiro quadrante. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o arco, em radianos, cuja tangente é o valor de Y/X, armazenando o resultado final em RESULT. O arco, para valores positivos de X e negativos de Y, sempre se encontra no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

no quarto quadrante. O bloco finaliza com sucesso e a saída Done é ativada.

10.3.5.1.10 Math Util

10.3.5.1.10.1 MAX

Bloco que compara os valores de Value1 e Value2 e armazena o maior deles em Result.

Representação Ladder



Estrutura do Bloco

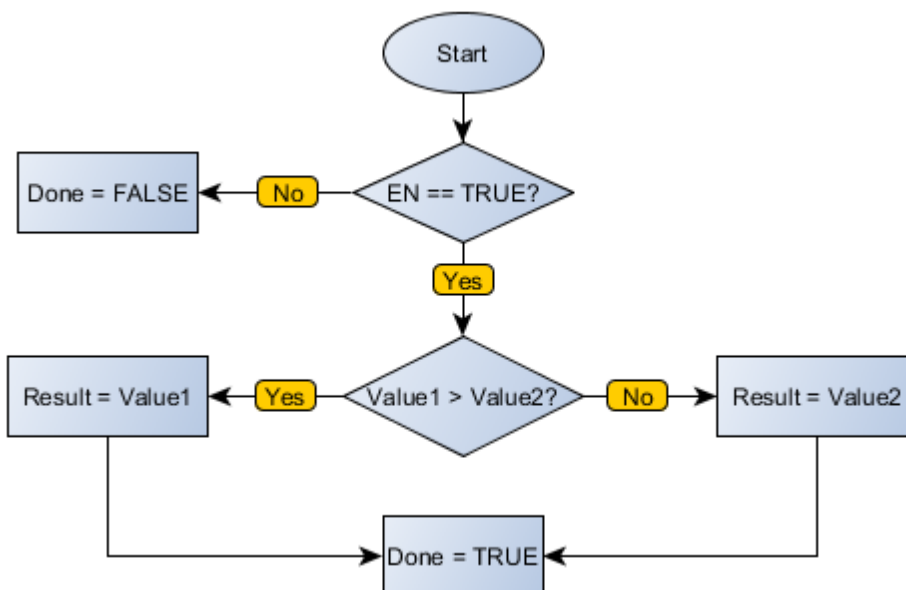
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Maior dos valores comparados

Funcionamento

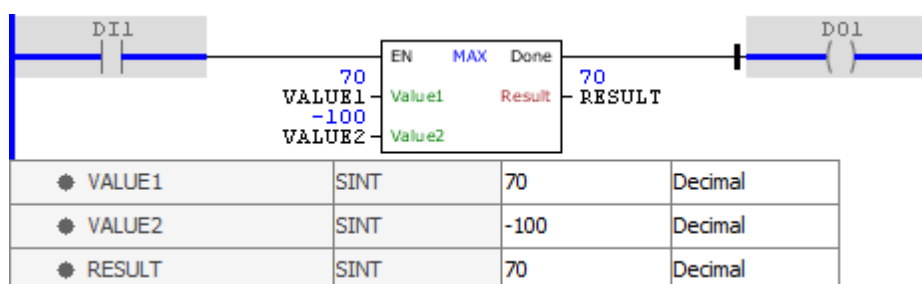
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o maior valor na comparação entre Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

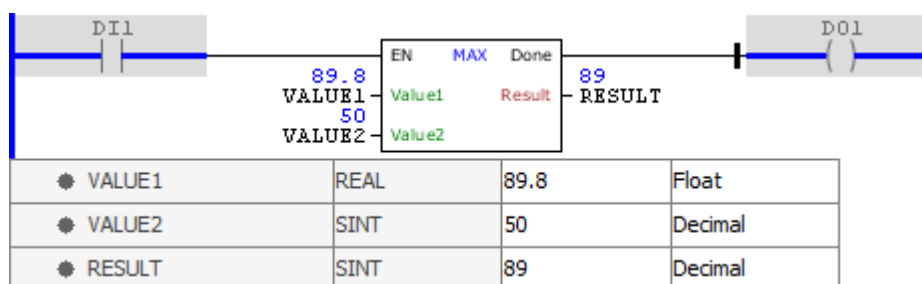
Fluxograma do Bloco



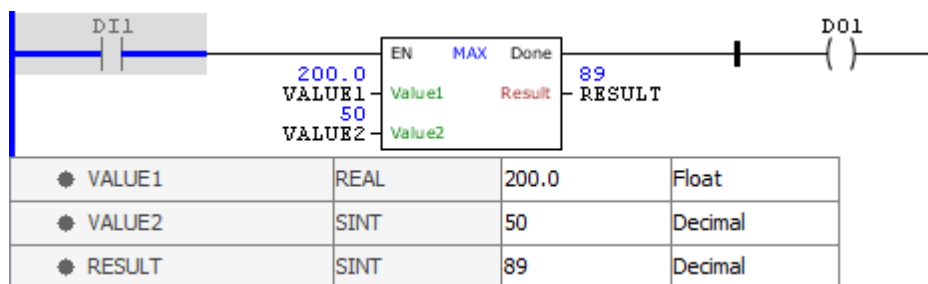
Exemplo



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.



O exemplo acima calcula o valor máximo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como o resultado é superior ao valor máximo suportado pelo tipo SINT, o bloco gera um erro e a saída Done é desativada.

10.3.5.1.10.2 MIN

Bloco que compara os valores de Value1 e Value2 e armazena o menor deles em Result.

Representação Ladder



Estrutura do Bloco

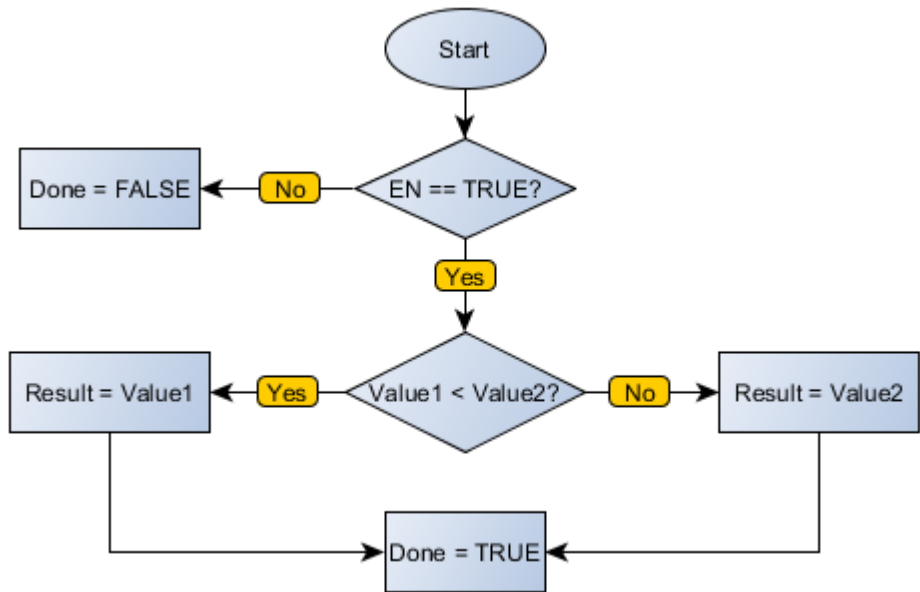
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Done	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Menor dos valores comparados

Funcionamento

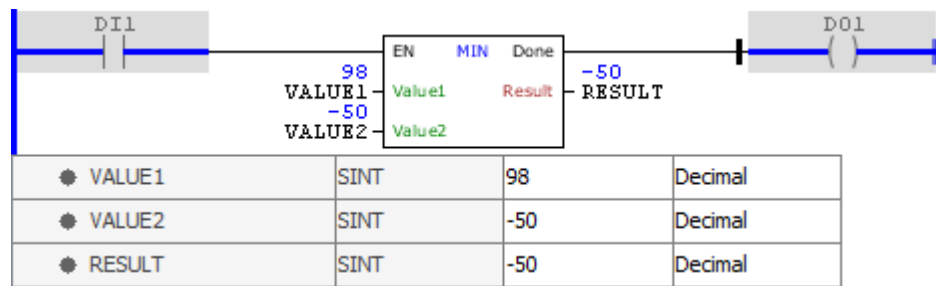
Este bloco, quando tem um valor TRUE em EN, envia para a saída Result o menor valor na comparação entre Value1 e Value2. Não havendo erros, a variável Done é setada. Caso haja algum erro na operação, Done não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Done permanece em FALSE.

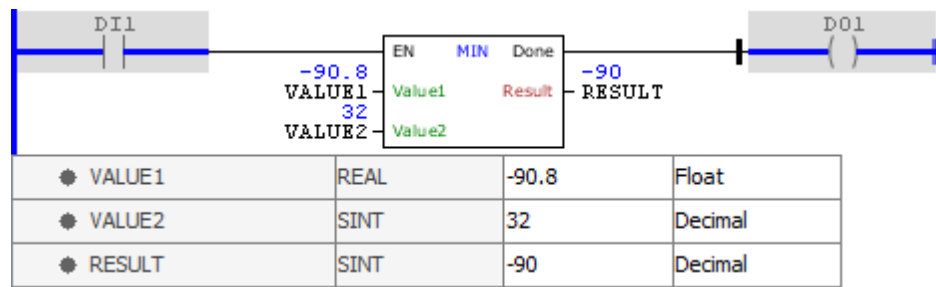
Fluxograma do Bloco



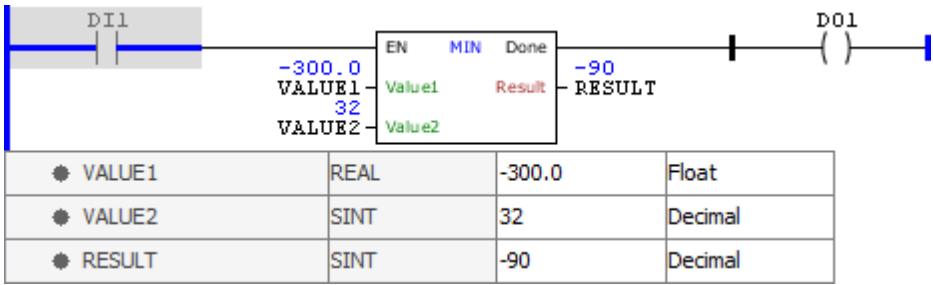
Exemplo



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada.



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída Done é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.



O exemplo acima calcula o valor mínimo entre as variáveis VALUE1 e VALUE2, armazenando o resultado final em RESULT. Como o resultado é inferior ao valor mínimo suportado pelo tipo SINT, o bloco gera um erro e a saída Done é desativada.

10.3.5.1.10.3 SAT

Bloco que realiza uma rotina de saturação do valor encontrado em Value de acordo com os limites estipulados em Minimum e Maximum, armazenando o resultado em Result.

Representação Ladder



Estrutura do Bloco

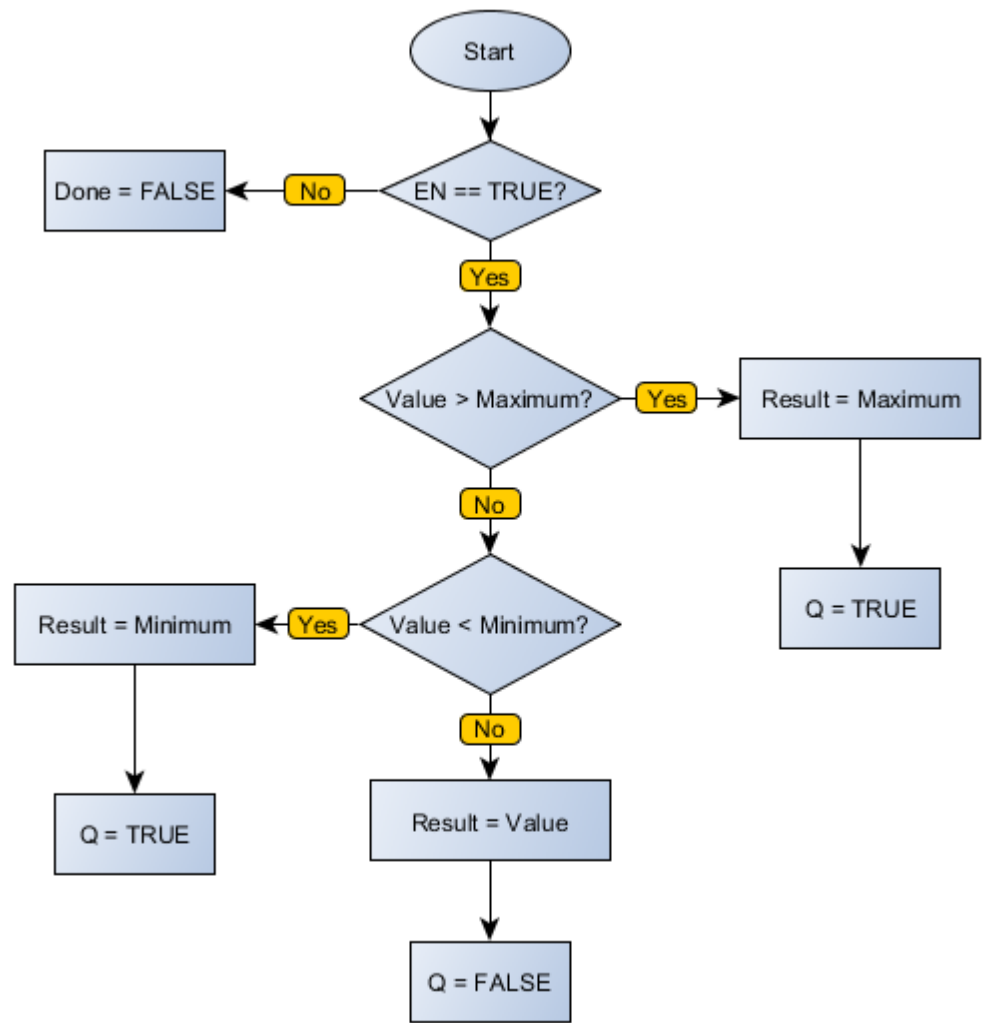
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de referência
	Minimum	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saturação inferior
	Maximum	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saturação superior
VAR_OUTPUT	Q	BOOL	Indicador de que houve saturação no processo
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Resultado da operação

Funcionamento

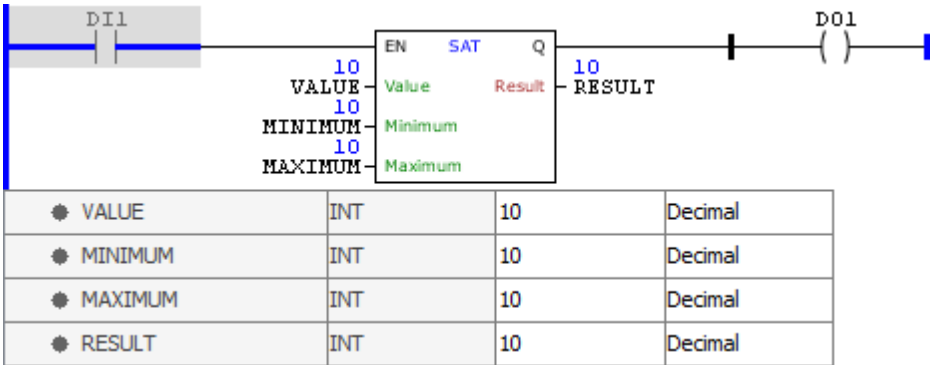
Este bloco, quando tem um valor TRUE em EN, realiza a comparação de Value com Minimum e Maximum. Se Value estiver na faixa entre Minimum e Maximum, Result recebe o valor de Value e Q permanece FALSE. Caso Value seja maior que Maximum, Result recebe Maximum e Q recebe TRUE. Caso Value seja menor que Minimum, Result recebe Minimum e Q recebe TRUE. Havendo algum erro na operação, Q não é setado, permanecendo em estado FALSE, enquanto Result permanece com seu valor inalterado.

Quando EN possui valor FALSE, Result permanece inalterado e Q permanece em FALSE.

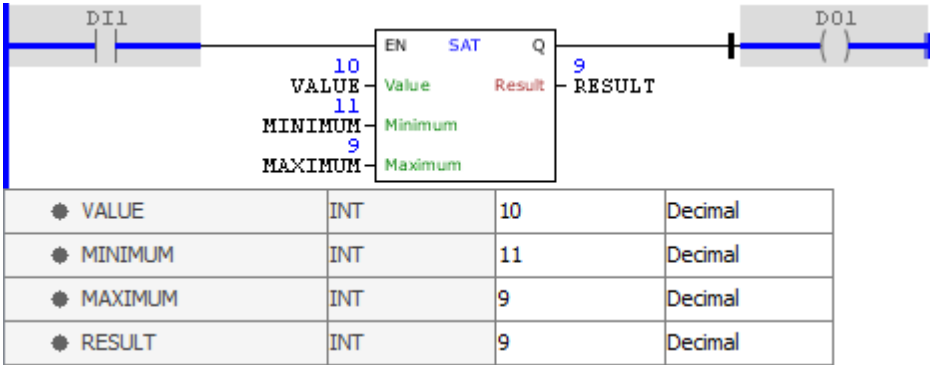
Fluxograma do Bloco



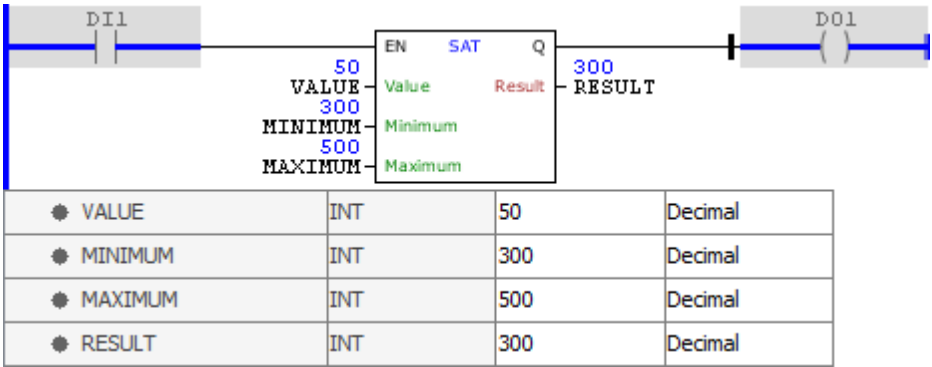
Exemplo



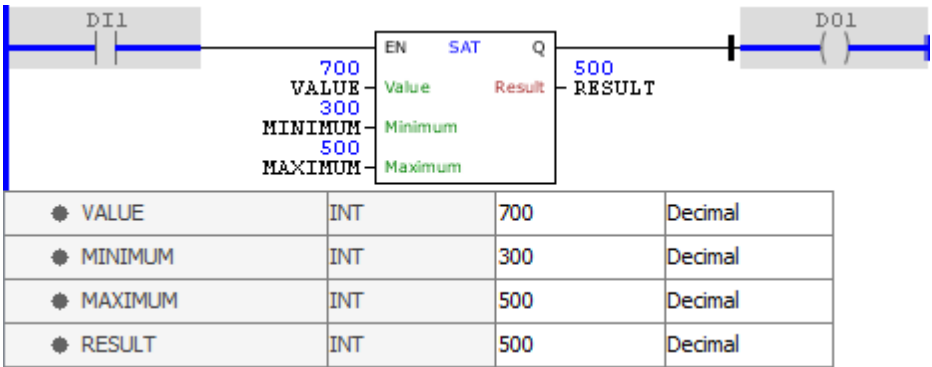
O exemplo acima repassa o valor de VALUE para RESULT, visto que ele não é inferior a MINIMUM ou superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é desativada, pois não houve saturação.



O exemplo acima repassa o valor de MAXIMUM para RESULT, visto que VALUE é superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.



O exemplo acima repassa o valor de MINIMUM para RESULT, visto que VALUE é inferior a MINIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.



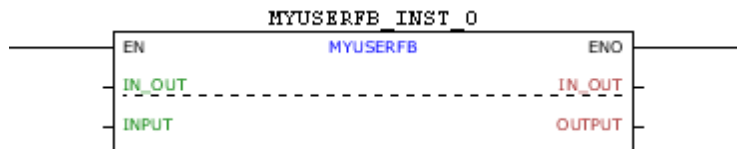
O exemplo acima repassa o valor de MAXIMUM para RESULT, visto que VALUE é superior a MAXIMUM. O bloco finaliza com sucesso e a saída Q é ativada, pois houve saturação.

10.3.5.1.11 Module

10.3.5.1.11.1 USERFB

Bloco que realiza uma subrotina programada pelo próprio usuário.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	INPUT	Conforme programação do usuário	Entradas do bloco
VAR_OUTPUT	ENO	BOOL	Fim de operação
	OUTPUT	Conforme programação do usuário	Saídas do bloco
VAR_IN_OUT	IN_OUT	Conforme programação do usuário	Variáveis utilizadas como entrada e saída do bloco
VAR	MYUSERFB_INST_0	MYUSERFB	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, atualiza os valores dos campos internos com as variáveis de entrada, executa a rotina Ladder programada pelo usuário e atualiza os valores das saídas após finalizada a rotina.

Quando EN possui valor FALSE, as saídas permanecem inalteradas.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

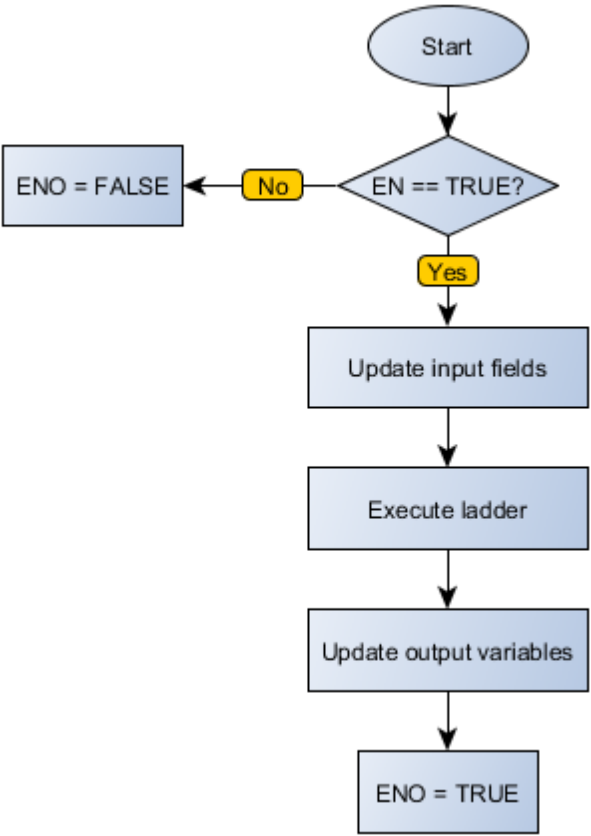
**NOTA!**

Consulte a seção [Trabalhando com USERFBs](#) para mais informações.

Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



10.3.5.1.12 Comparação

10.3.5.1.12.1 COMP_EQ

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso ambos sejam iguais.

Representação Ladder



Estrutura do Bloco

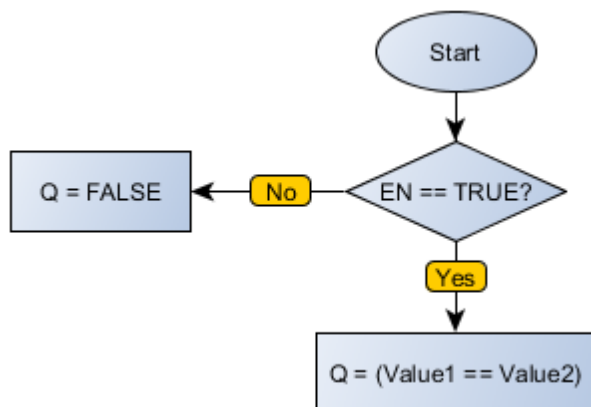
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade

Funcionamento

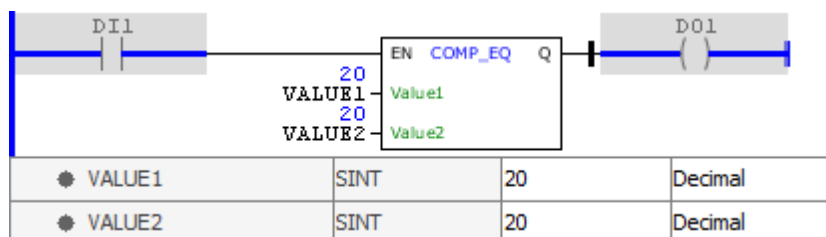
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 e Value2 sejam iguais. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

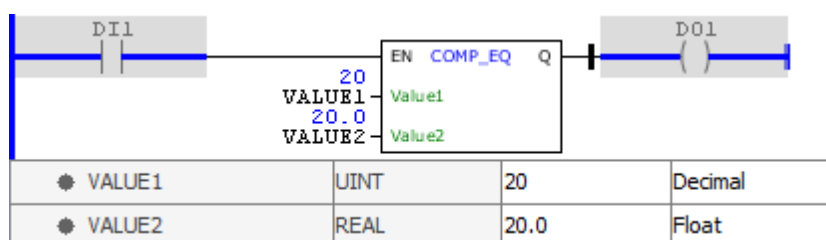
Fluxograma do Bloco



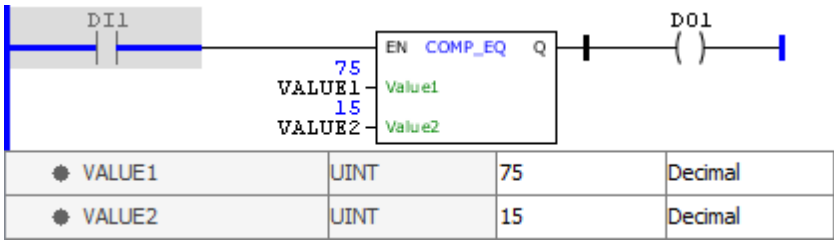
Exemplo



O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.



O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada. Observe que os tipos das variáveis de entrada podem ser diferentes, sem causar problemas de execução.

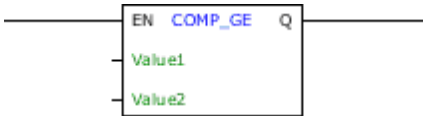


O exemplo acima verifica a igualdade entre VALUE1 e VALUE2. Como as variáveis possuem valores diferentes, a saída Q é desativada.

10.3.5.1.12.2 COMP_GE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja maior ou igual a Value2.

Representação Ladder



Estrutura do Bloco

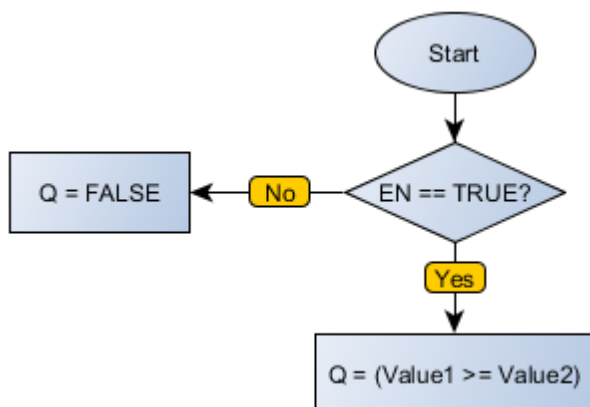
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade ou maioridade de Value1

Funcionamento

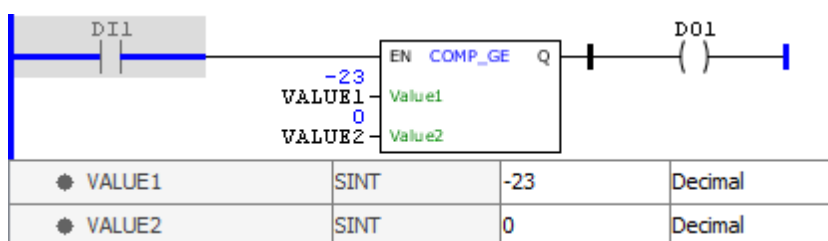
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja maior ou igual a Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

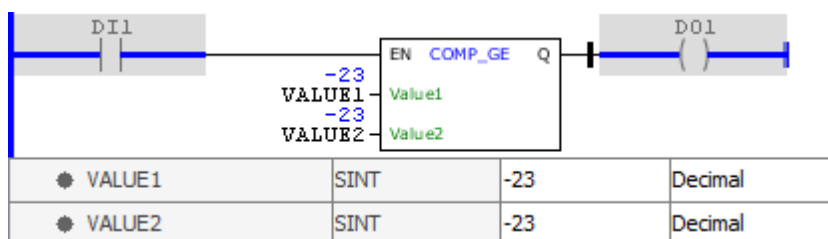
Fluxograma do Bloco



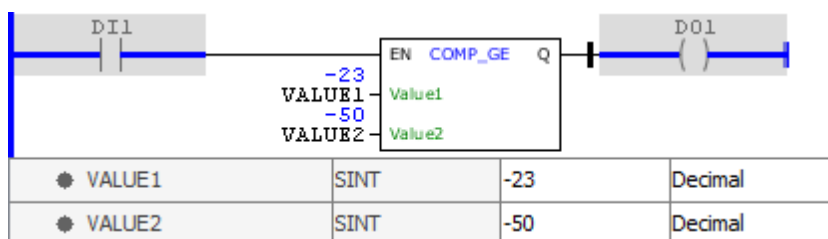
Exemplo



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é desativada.



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.



O exemplo acima verifica a igualdade ou maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é ativada.

10.3.5.1.12.3 COMP_GT

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja maior que Value2.

Representação Ladder



Estrutura do Bloco

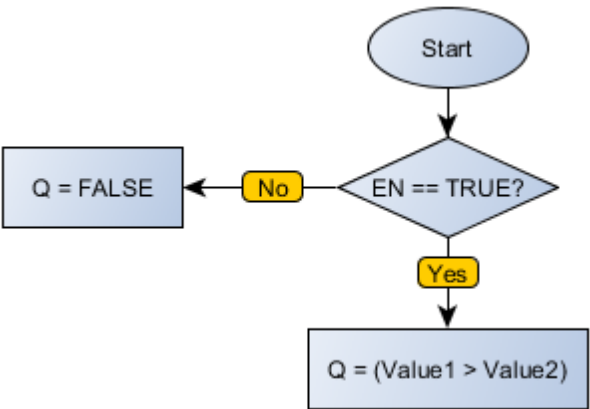
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de maioridade de Value1

Funcionamento

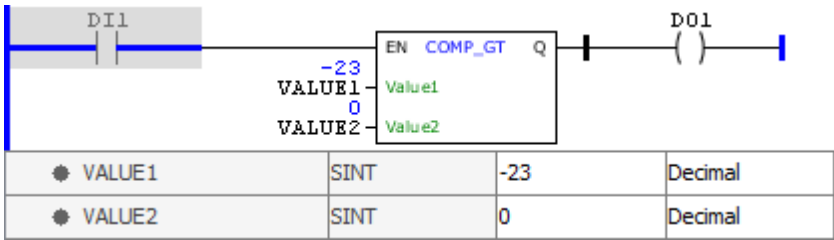
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja maior que Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

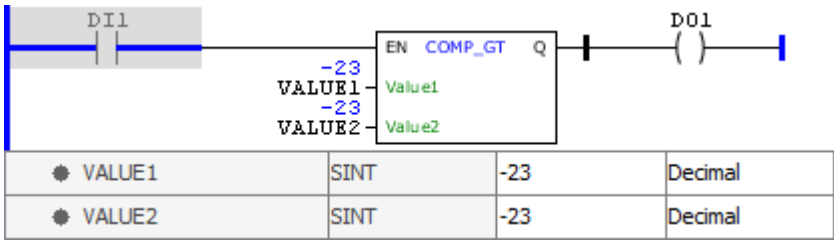
Fluxograma do Bloco



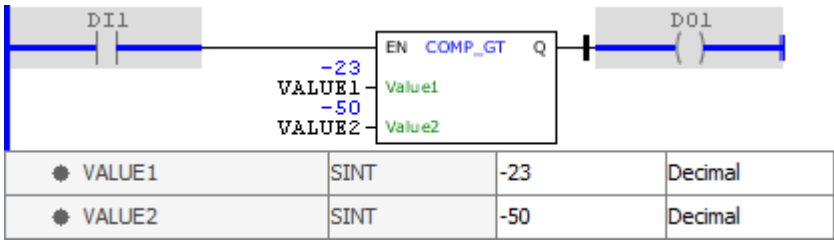
Exemplo



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é desativada.



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.



O exemplo acima verifica a maioridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é ativada.

10.3.5.1.12.4 COMP_LE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja menor ou igual a Value2.

Representação Ladder



Estrutura do Bloco

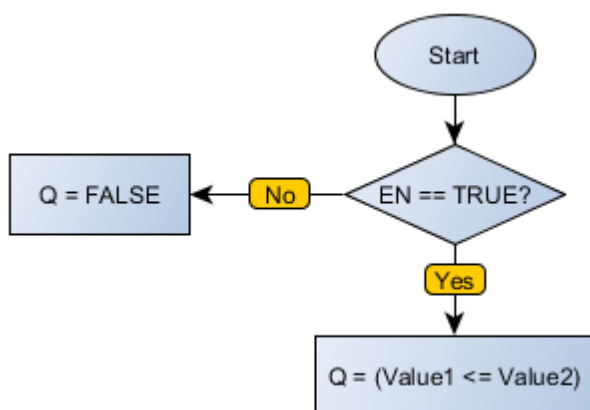
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de igualdade ou minoridade de Value1

Funcionamento

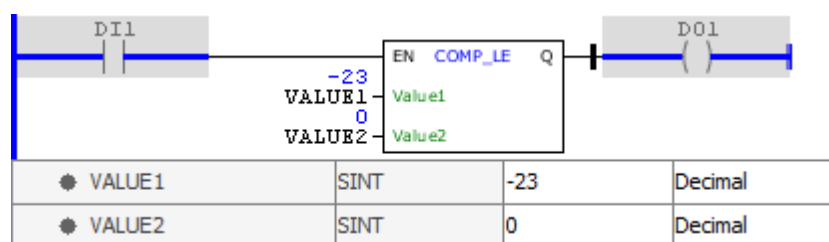
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja menor ou igual a Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

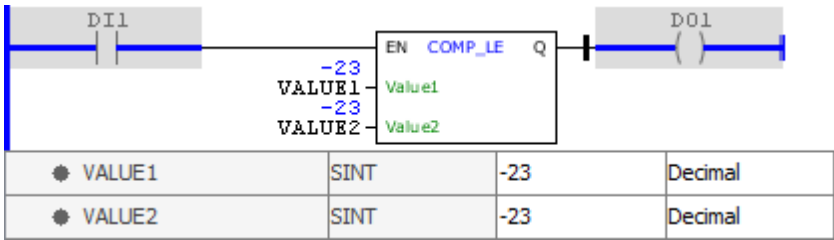
Fluxograma do Bloco



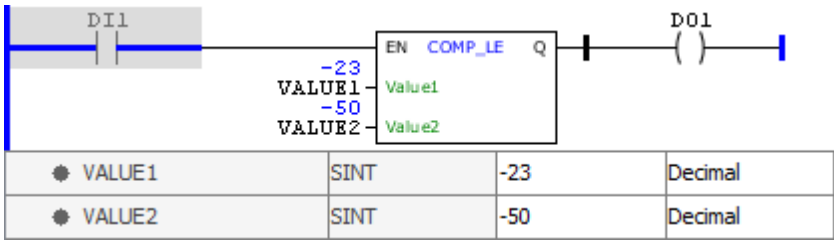
Exemplo



O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é ativada.



O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é ativada.

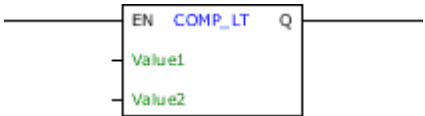


O exemplo acima verifica a igualdade ou minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é desativada.

10.3.5.1.12.5 COMP_LT

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja menor que Value2.

Representação Ladder



Estrutura do Bloco

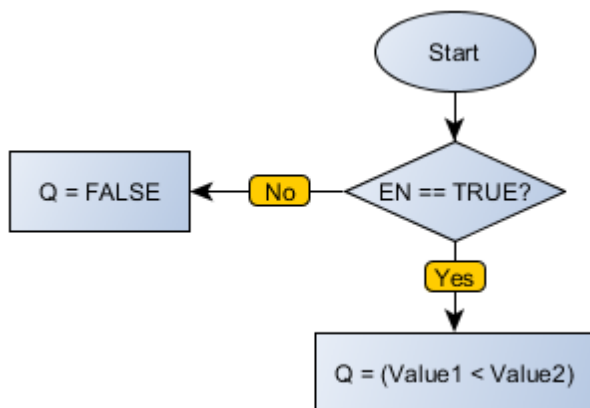
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de minoridade de Value1

Funcionamento

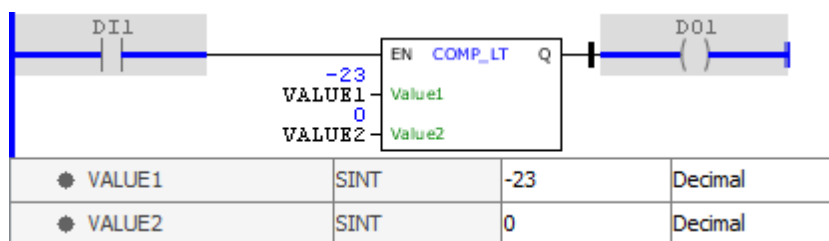
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja menor que Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

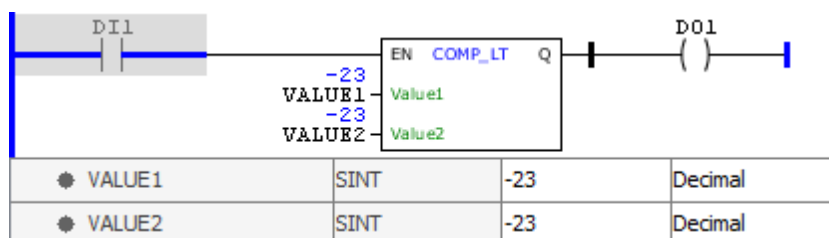
Fluxograma do Bloco



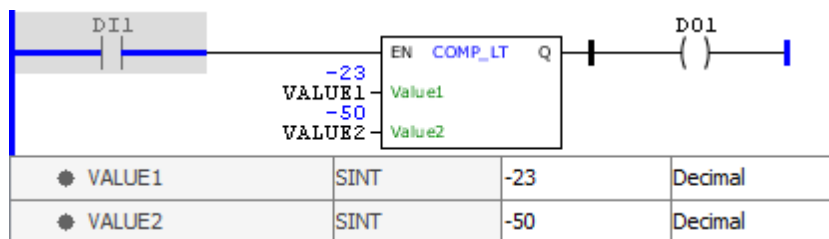
Exemplo



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor inferior a VALUE2, a saída Q é ativada.



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.



O exemplo acima verifica a minoridade de VALUE1 em relação a VALUE2. Como VALUE1 possui valor superior a VALUE2, a saída Q é desativada.

10.3.5.1.12.6 COMP_NE

Bloco que compara os valores de Value1 e Value2, habilitando a saída Q caso Value1 seja diferente de Value2.

Representação Ladder



Estrutura do Bloco

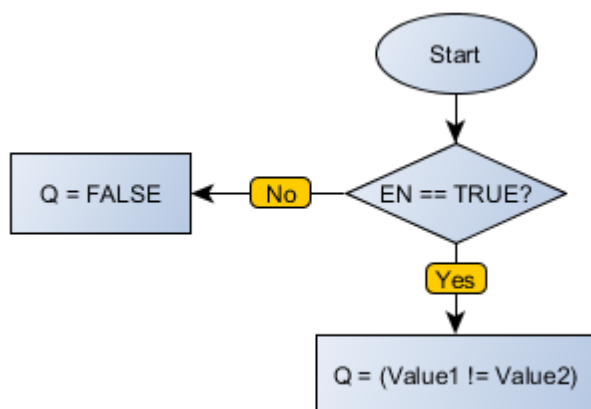
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Primeiro valor da comparação
	Value2	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Segundo valor da comparação
VAR_OUTPUT	Q	BOOL	Indicador de desigualdade

Funcionamento

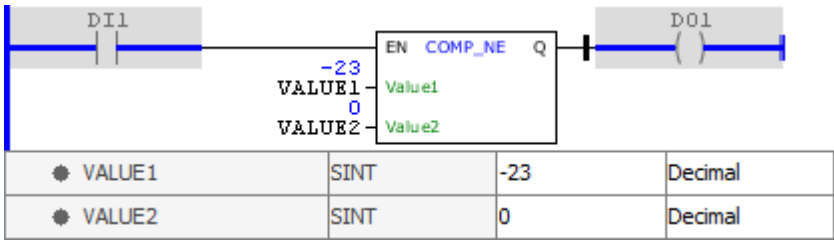
Este bloco, quando tem um valor TRUE em EN, envia para a saída Q o valor TRUE caso Value1 seja diferente de Value2. Caso contrário, Q recebe FALSE.

Quando EN possui valor FALSE, Q permanece em FALSE.

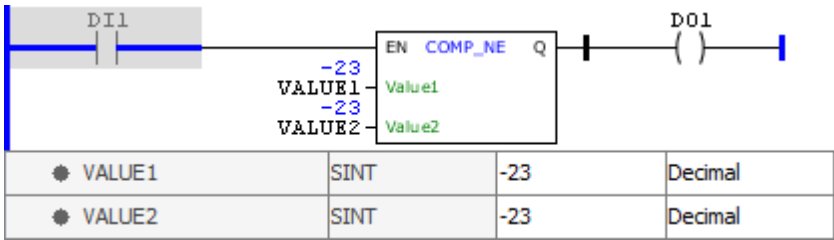
Fluxograma do Bloco



Exemplo



O exemplo acima verifica a desigualdade entre VALUE1 e VALUE2. Como as variáveis possuem valores diferentes, a saída Q é ativada.



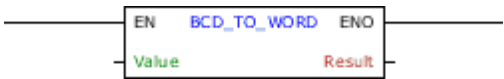
O exemplo acima verifica a desigualdade entre VALUE1 e VALUE2. Como ambas as variáveis possuem o mesmo valor, a saída Q é desativada.

10.3.5.1.13 Conversão

10.3.5.1.13.1 BCD

Bloco que realiza a conversão de um código BCD para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em BCD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

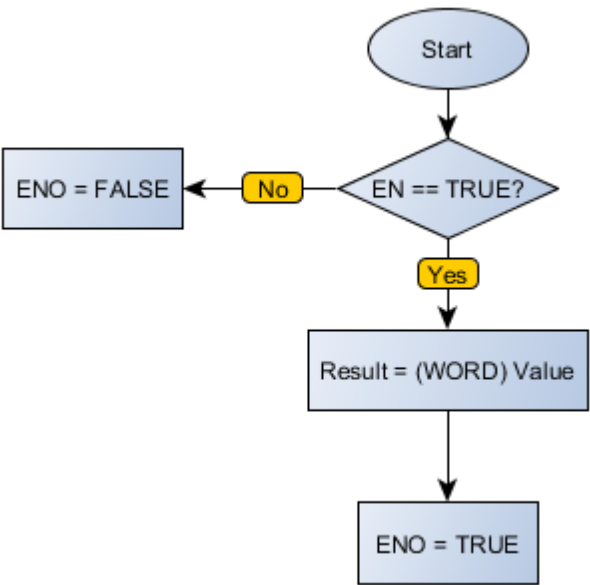
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BCD e realiza a conversão para WORD, armazenando em Result.

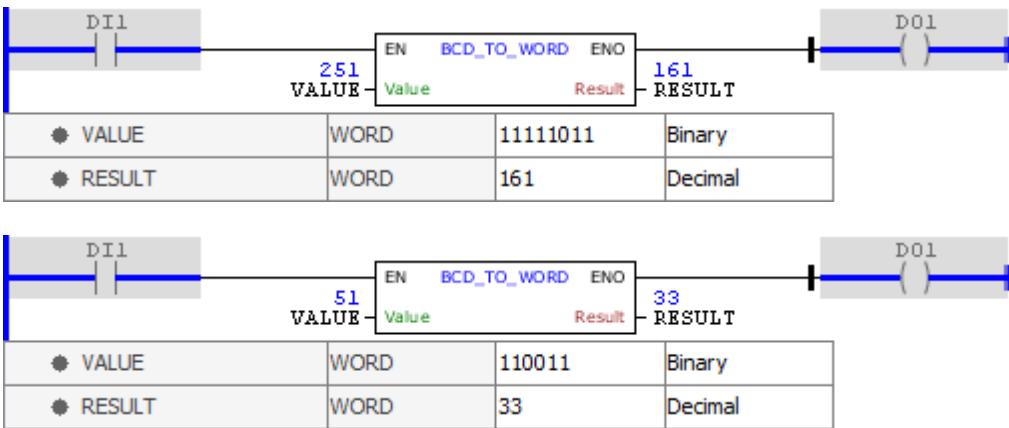
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



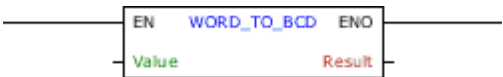
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BCD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um código BCD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em BCD

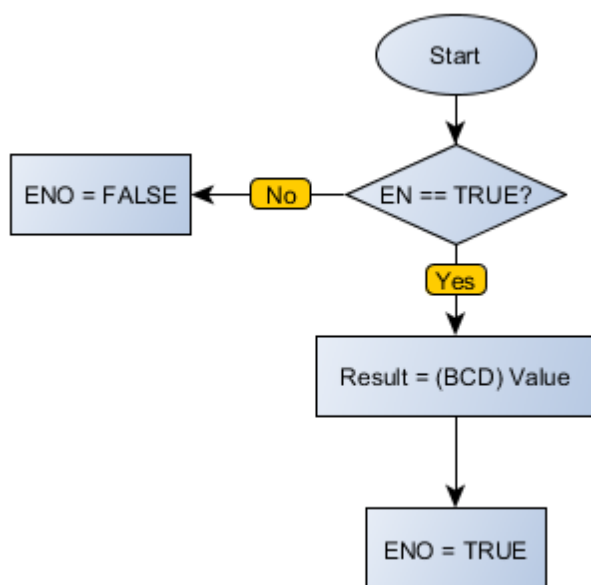
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BCD, armazenando em Result.

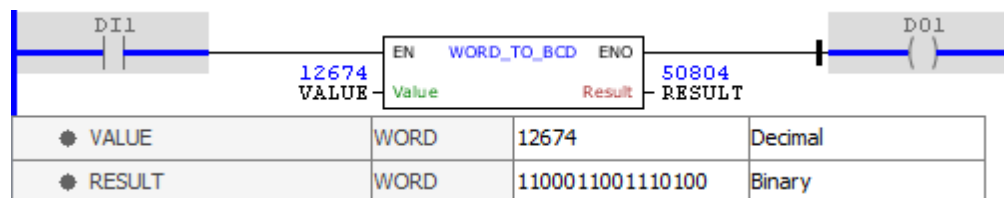
Quando EN possui valor FALSE, Result permanece inalterado.

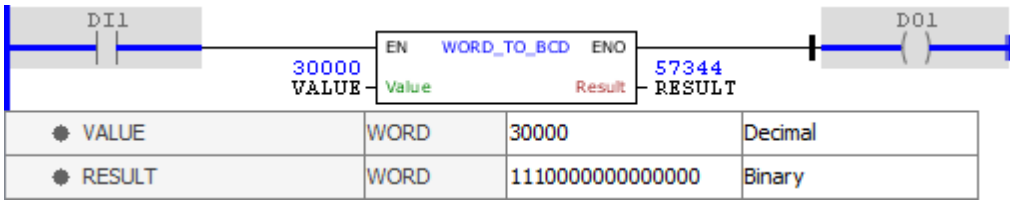
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



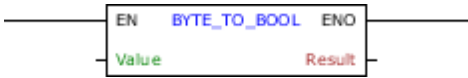


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BCD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.13.2 BOOL

Bloco que realiza a conversão de um valor BYTE para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

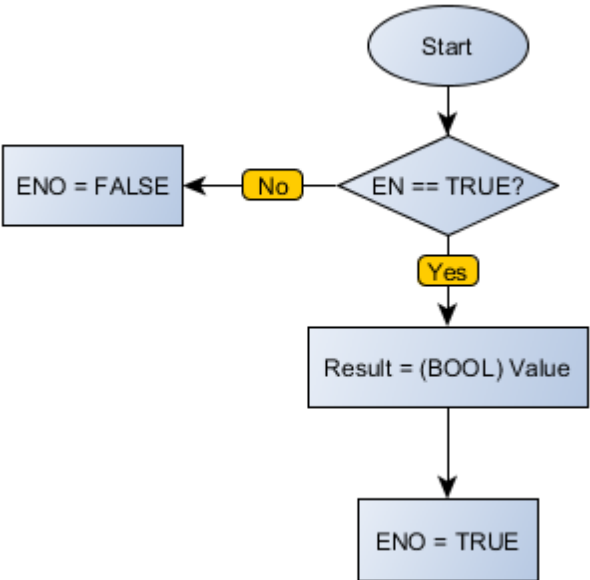
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para BOOL, armazenando em Result.

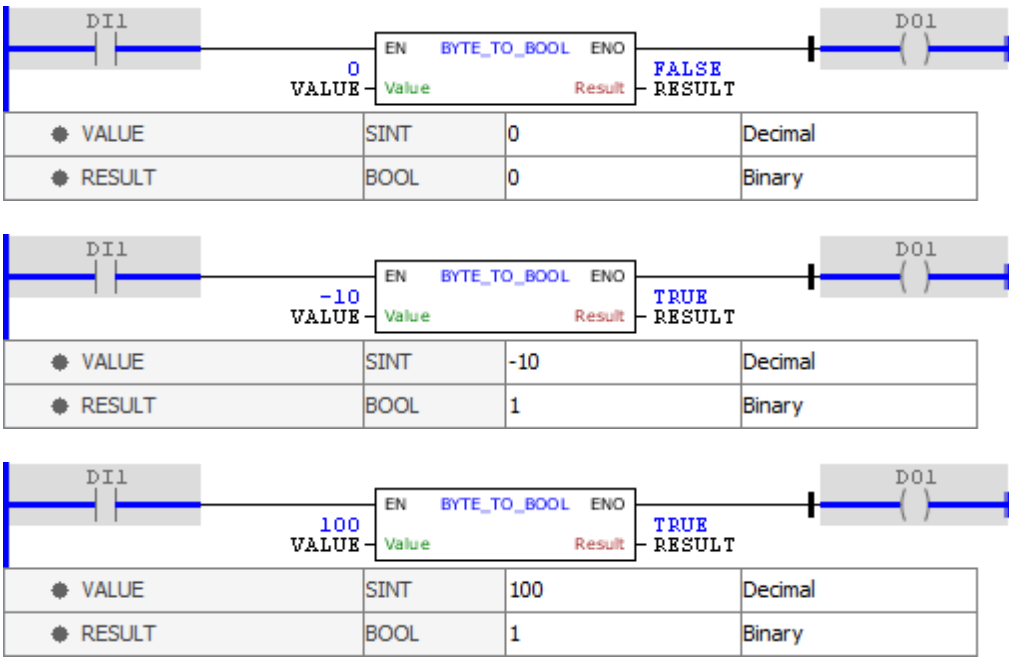
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



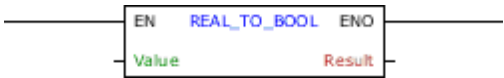
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

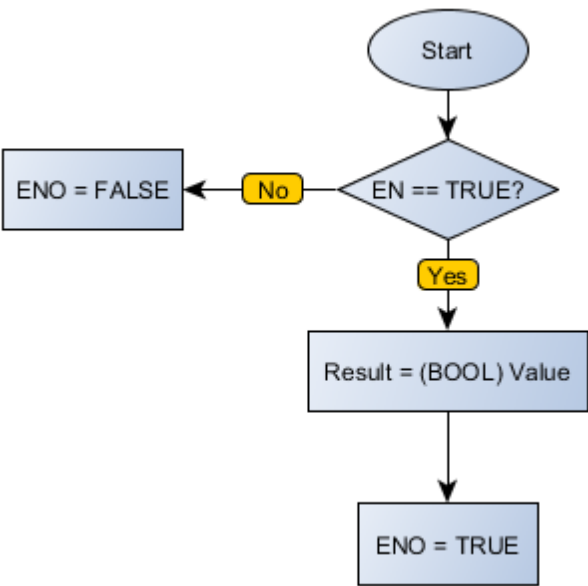
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para BOOL, armazenando em Result.

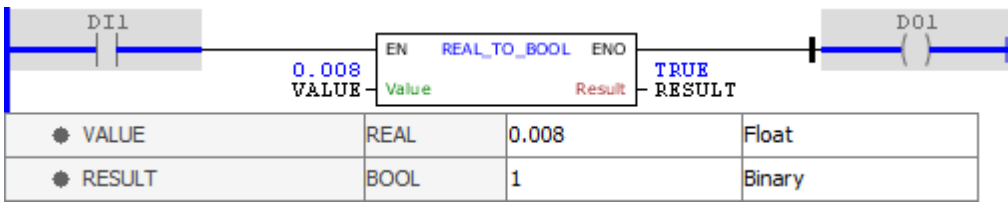
Quando EN possui valor FALSE, Result permanece inalterado.

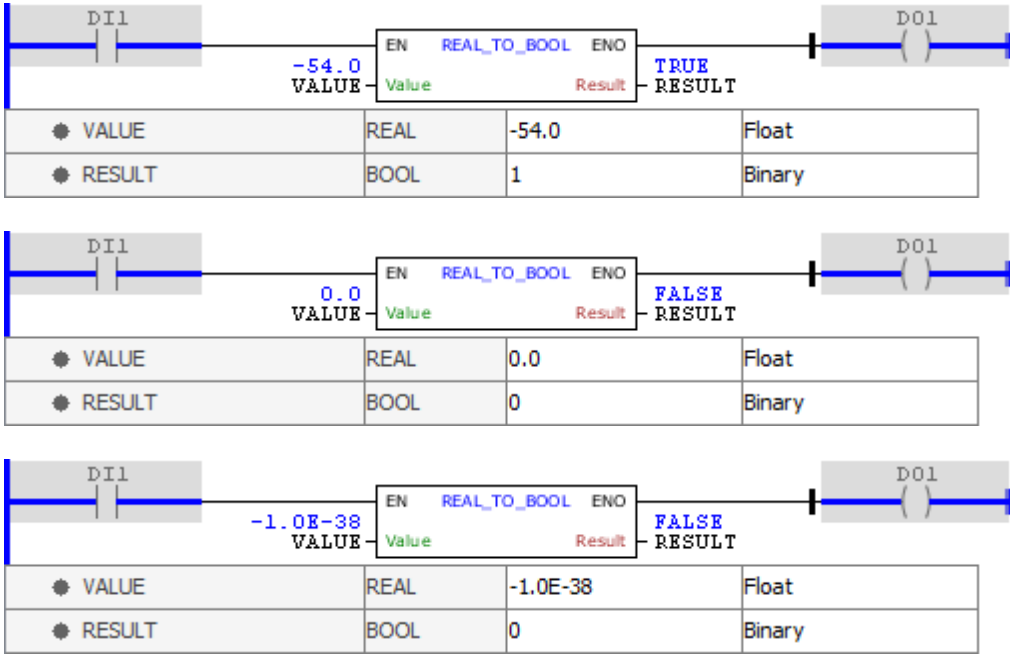
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

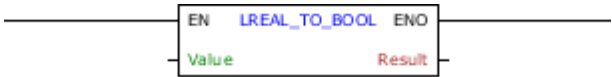




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe no último exemplo que valores muito próximos do epsilon da máquina podem acarretar numa interpretação de valor FALSE.

Bloco que realiza a conversão de um valor LREAL para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

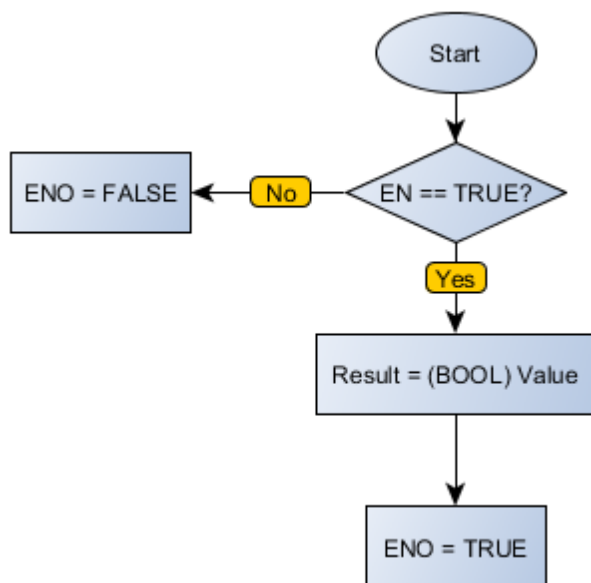
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para BOOL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

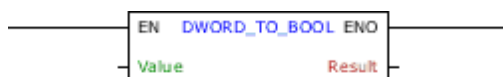
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor DWORD para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

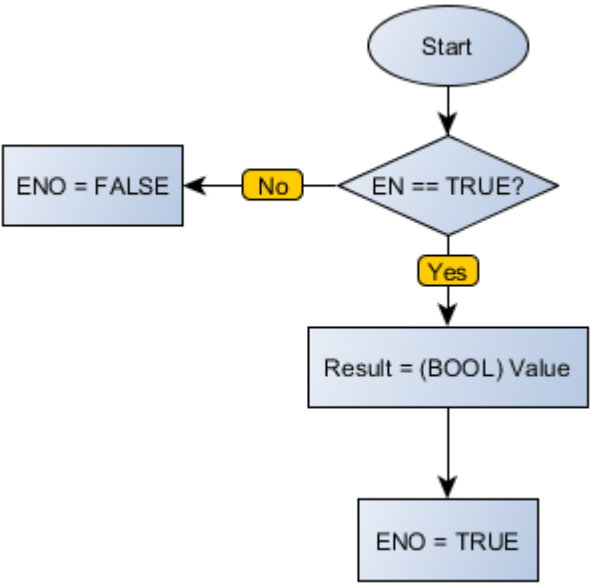
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para BOOL, armazenando em Result.

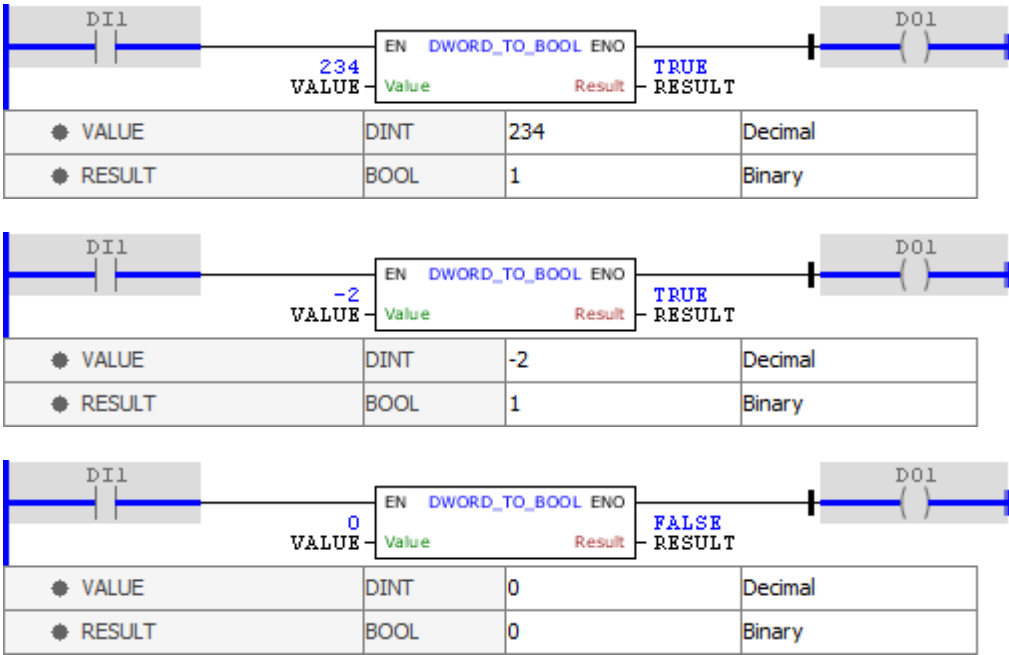
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



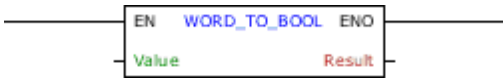
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um valor BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BOOL	Valor em BOOL

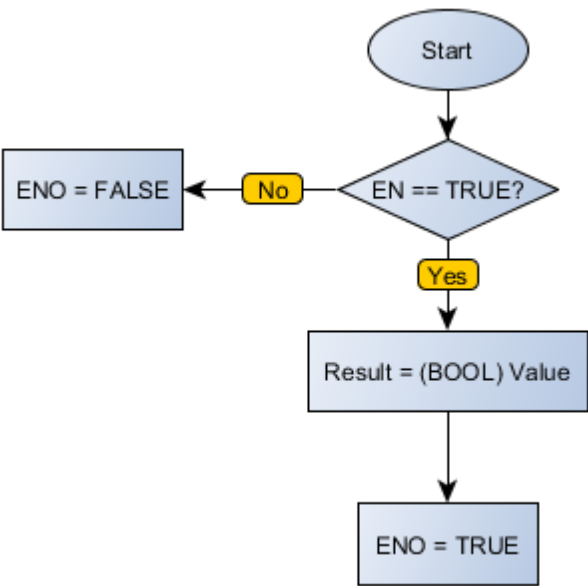
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BOOL, armazenando em Result.

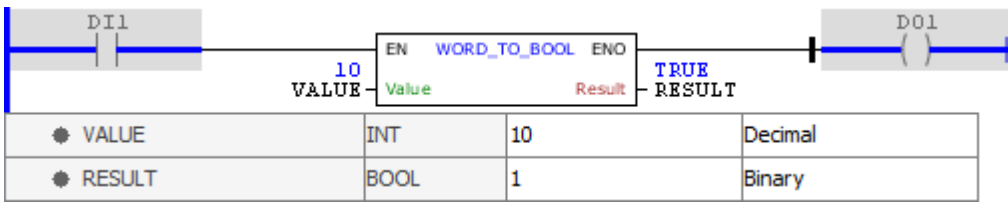
Quando EN possui valor FALSE, Result permanece inalterado.

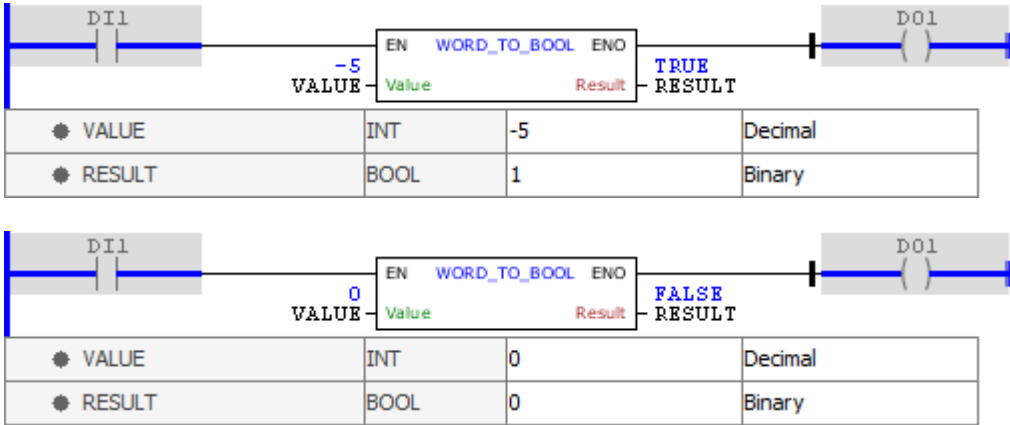
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



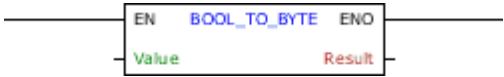


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BOOL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.13.3 BYTE

Bloco que realiza a conversão de um valor BOOL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

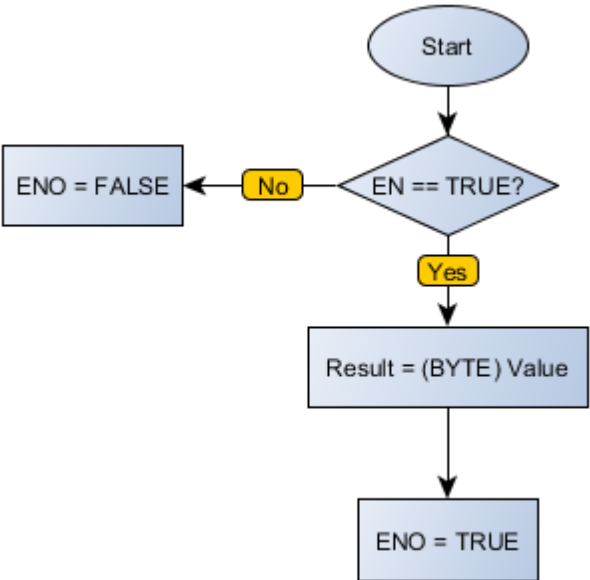
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para BYTE, armazenando em Result.

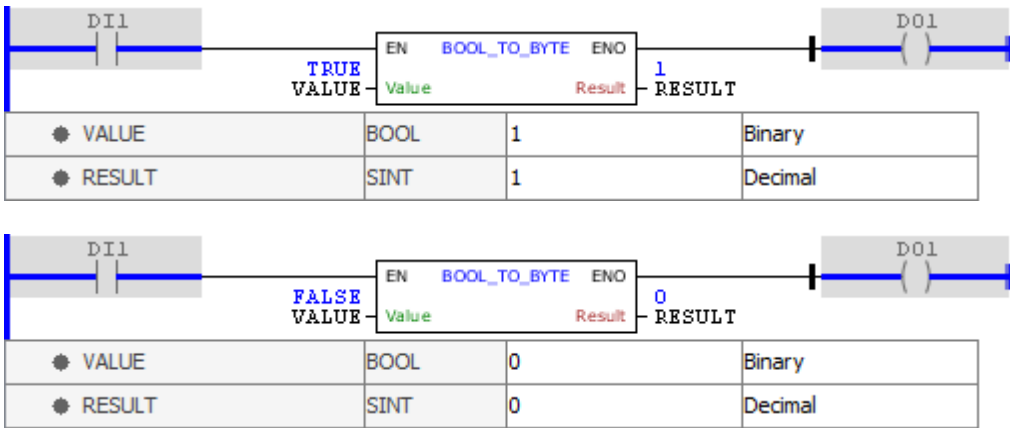
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



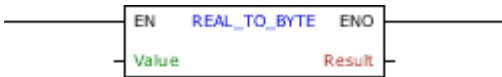
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

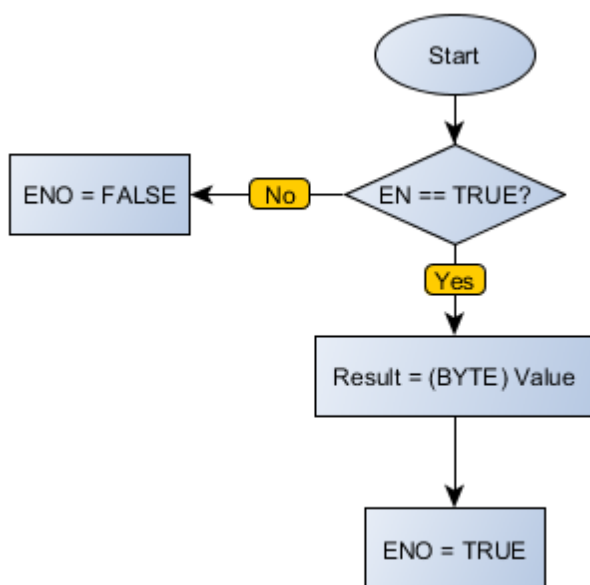
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para BYTE, armazenando em Result.

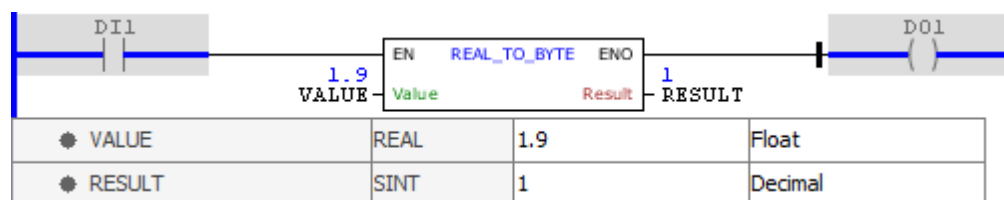
Quando EN possui valor FALSE, Result permanece inalterado.

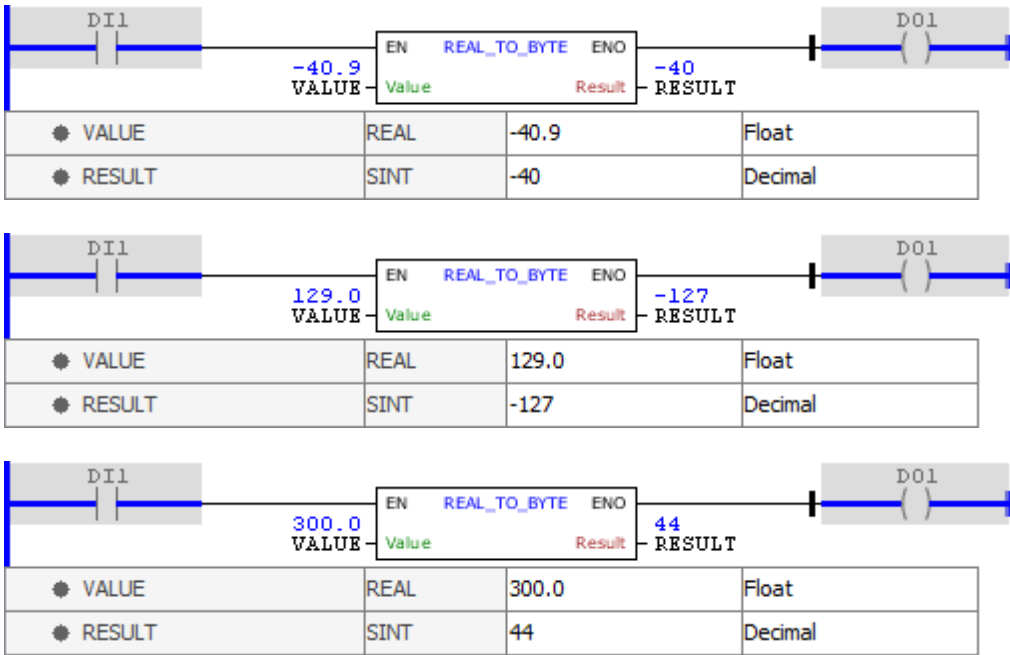
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

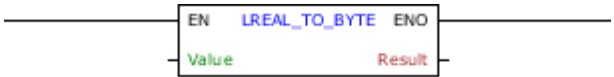




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os oito bits menos significativos.

Bloco que realiza a conversão de um valor LREAL para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	B00L	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

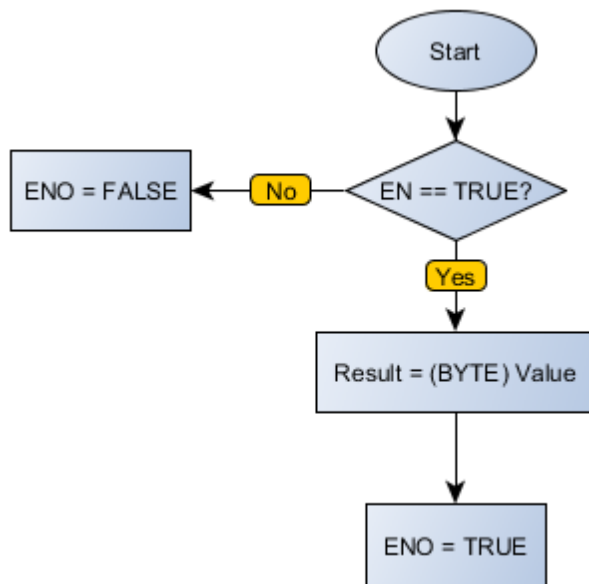
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para BYTE, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

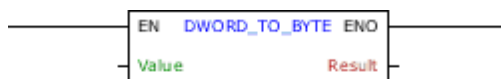
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor DWORD para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

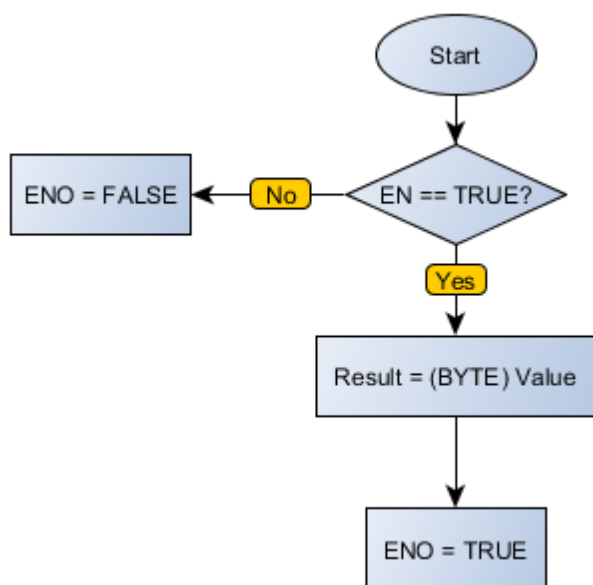
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para BYTE, armazenando em Result.

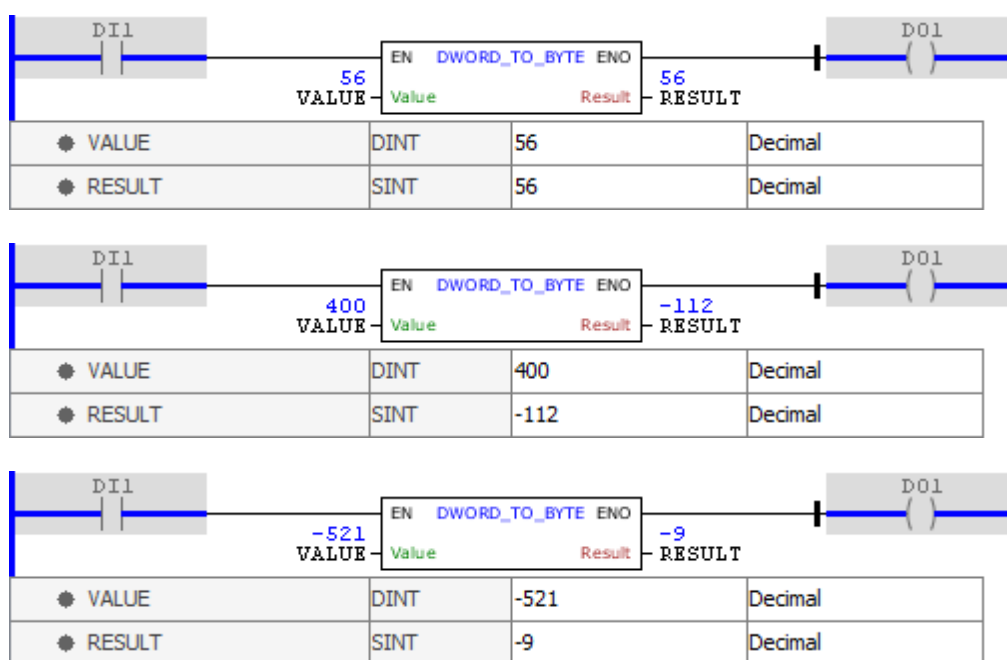
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



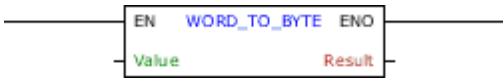
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os oito bits menos significativos.

Bloco que realiza a conversão de um valor WORD para um valor BYTE.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT	Valor em BYTE

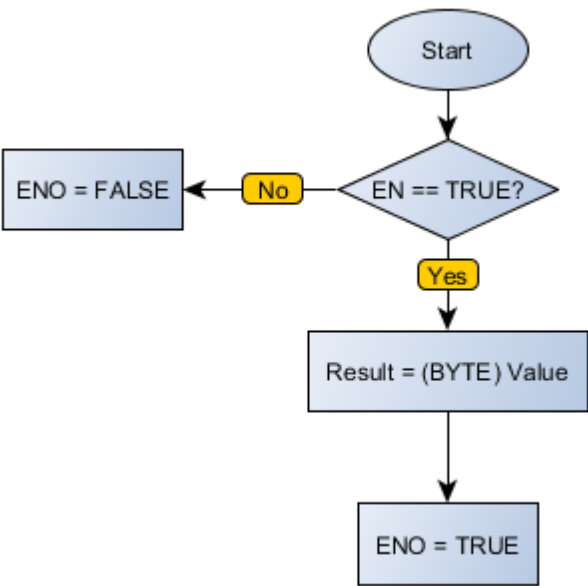
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para BYTE, armazenando em Result.

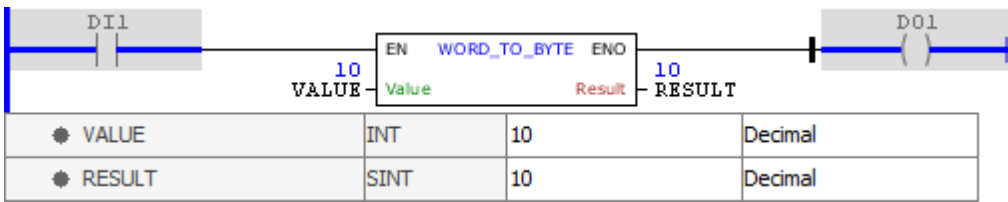
Quando EN possui valor FALSE, Result permanece inalterado.

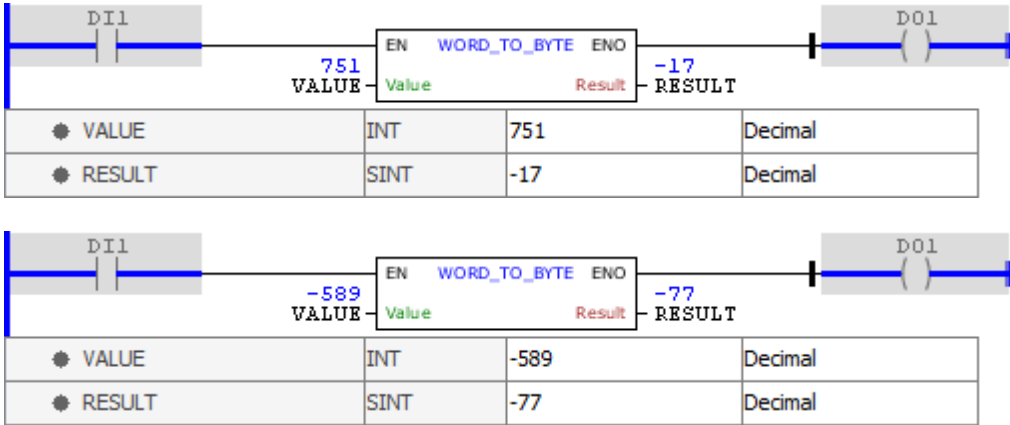
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



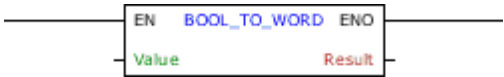


Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor BYTE, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os oito bits menos significativos.

10.3.5.1.13.4 WORD

Bloco que realiza a conversão de um valor BOOL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

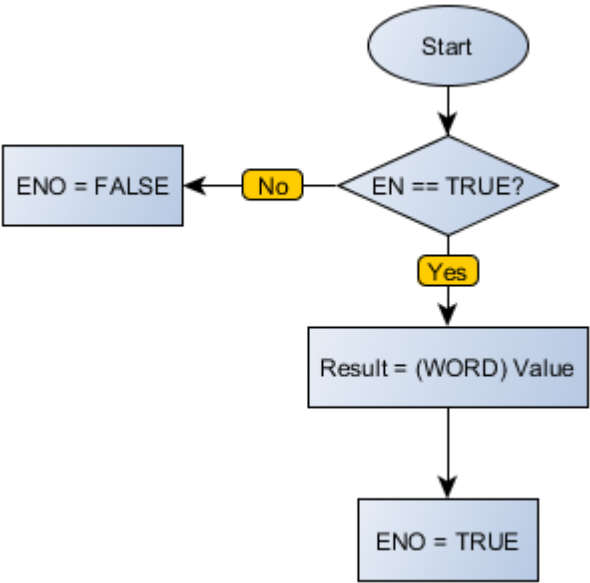
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para WORD, armazenando em Result.

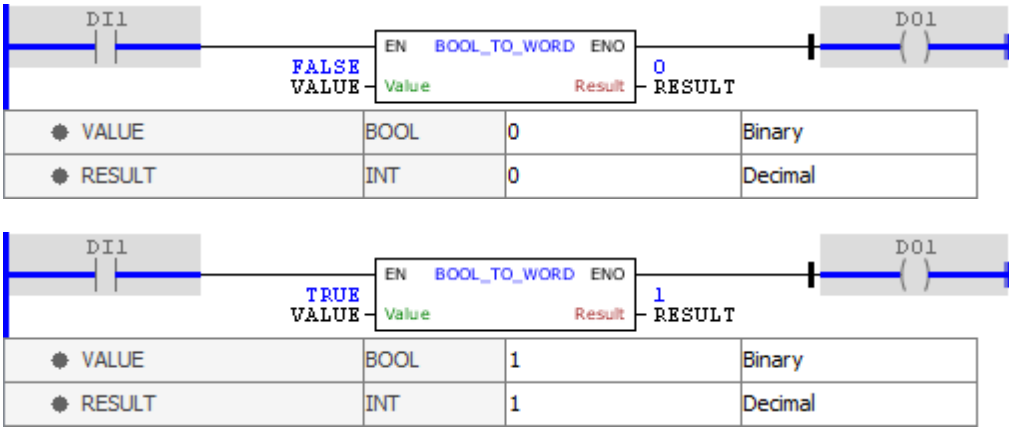
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



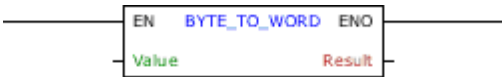
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

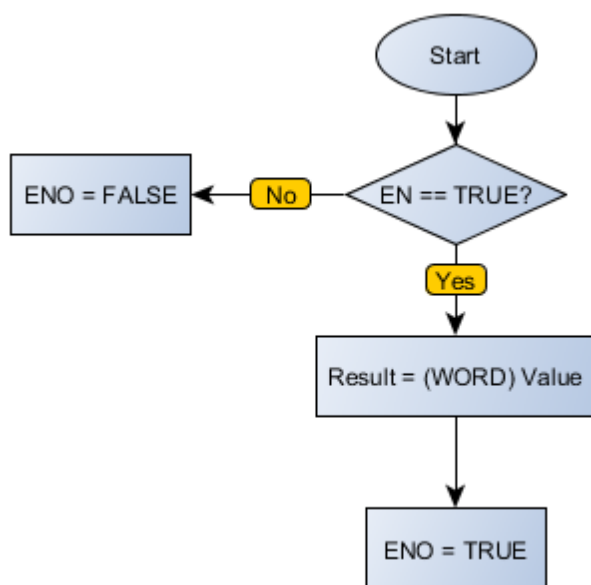
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para WORD, armazenando em Result.

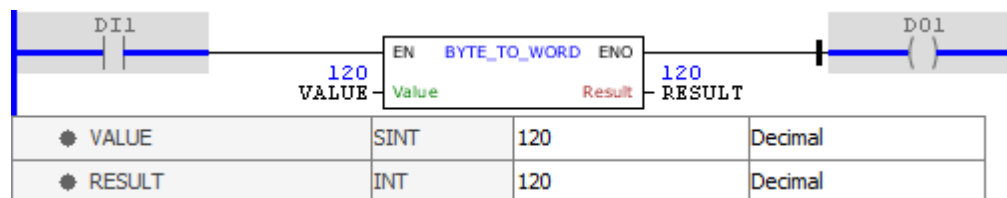
Quando EN possui valor FALSE, Result permanece inalterado.

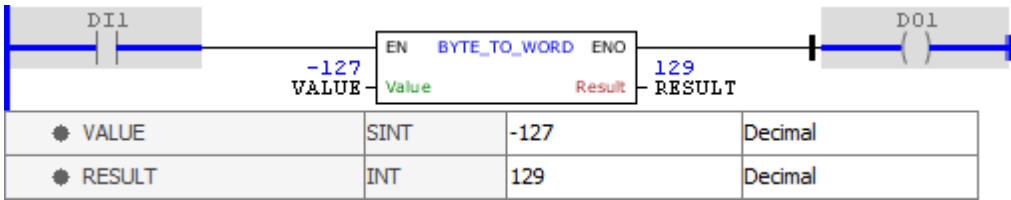
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

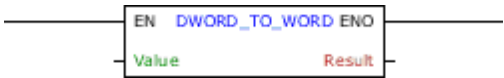




Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor DWORD para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

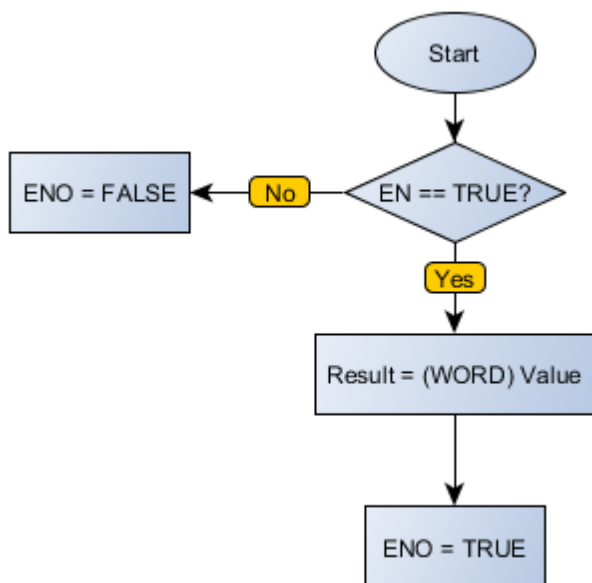
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para WORD, armazenando em Result.

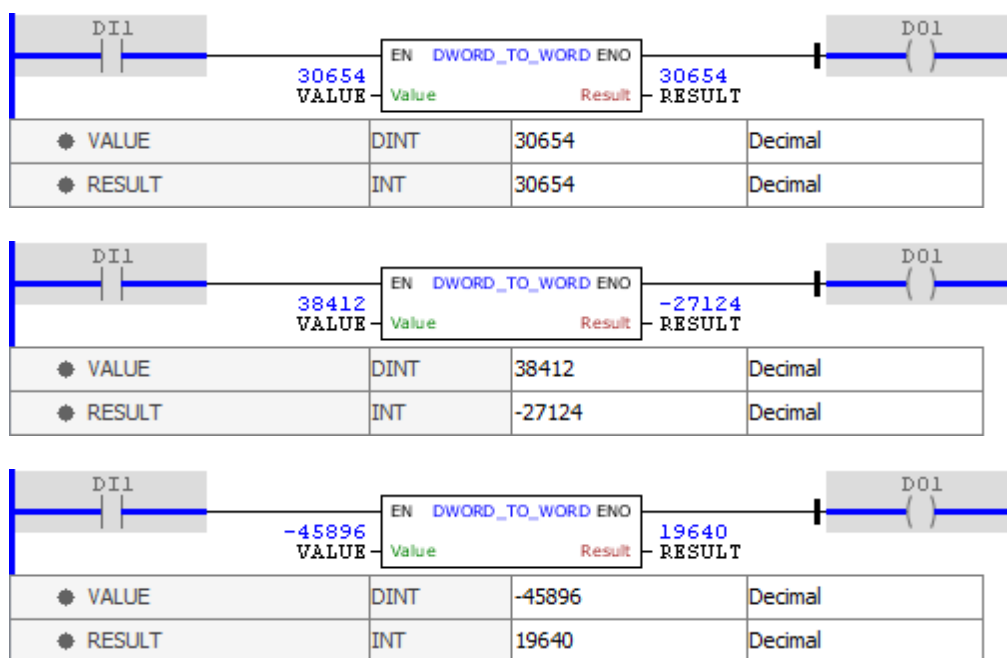
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



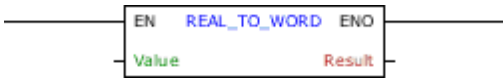
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que são levados em consideração apenas os dezesseis bits menos significativos.

Bloco que realiza a conversão de um valor REAL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

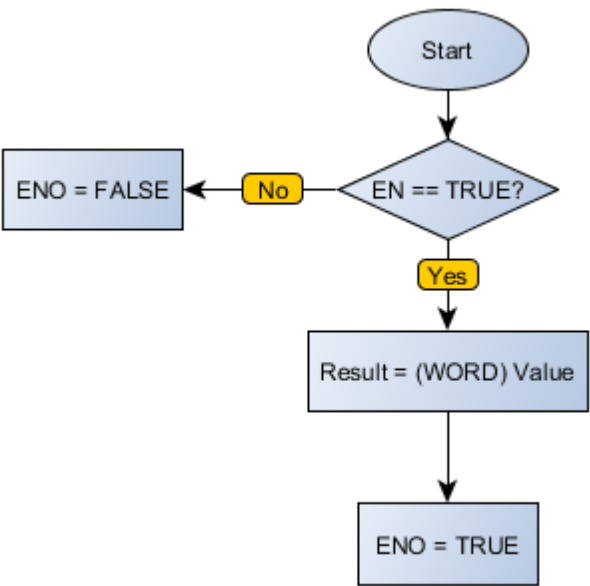
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para WORD, armazenando em Result.

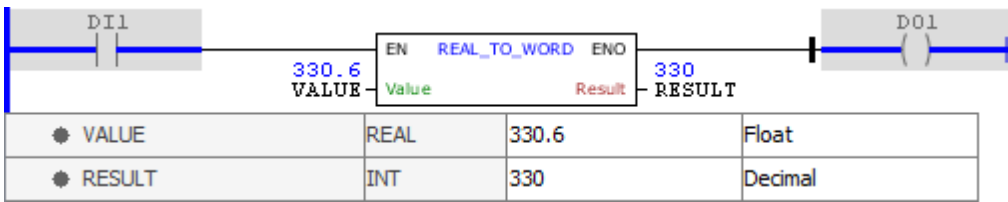
Quando EN possui valor FALSE, Result permanece inalterado.

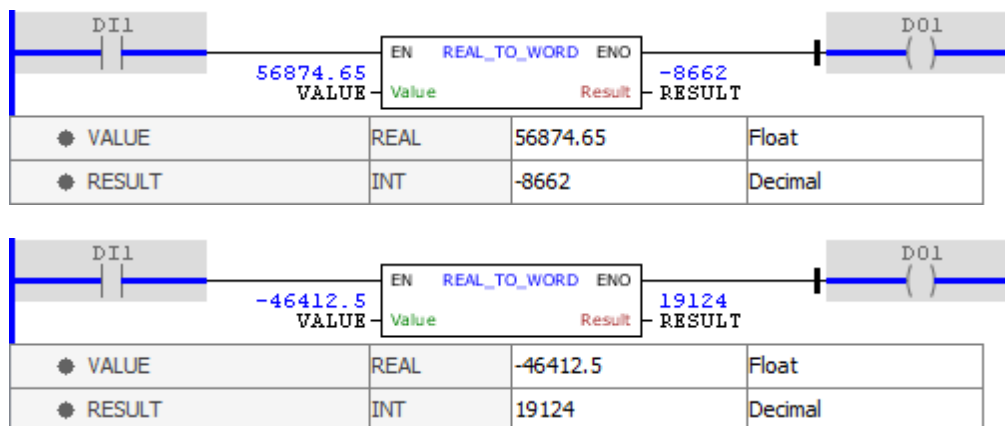
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

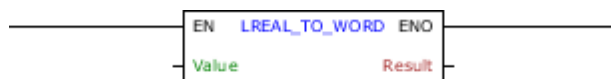




Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor WORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os dezesseis bits menos significativos.

Bloco que realiza a conversão de um valor LREAL para um valor WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	WORD UINT INT	Valor em WORD

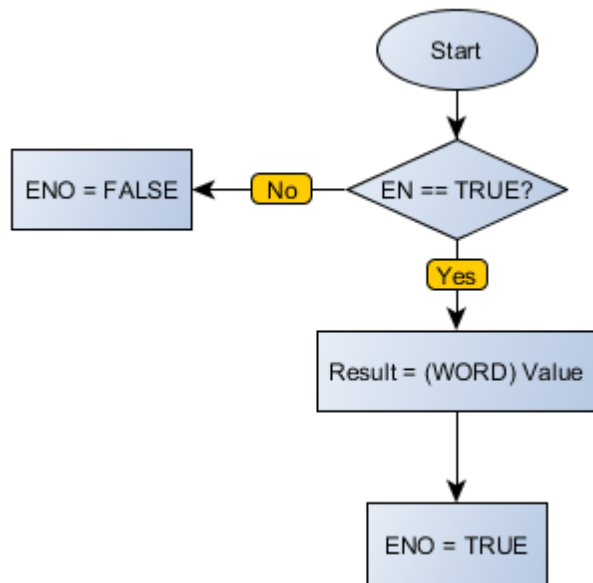
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para WORD, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

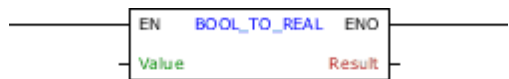
Fluxograma do Bloco



10.3.5.1.13.5 REAL

Bloco que realiza a conversão de um valor BOOL para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

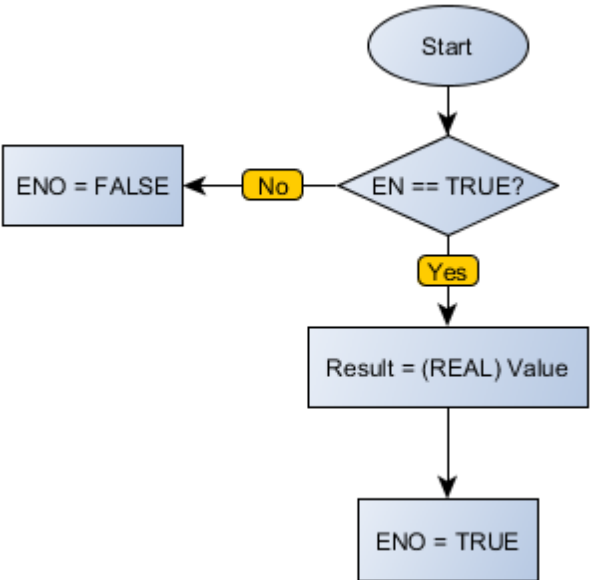
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para REAL, armazenando em Result.

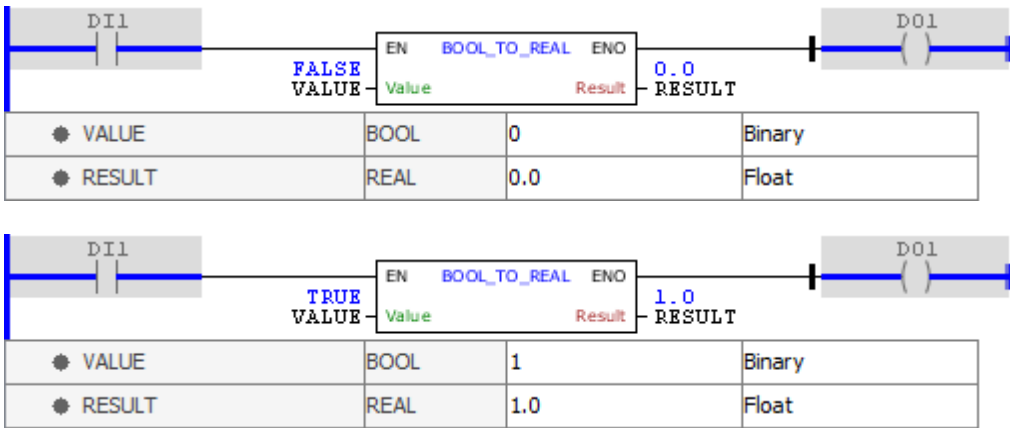
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



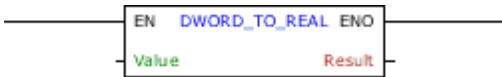
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor DWORD para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

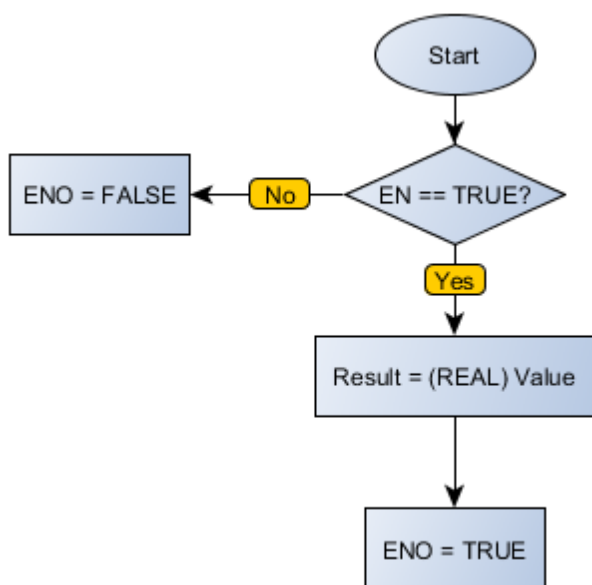
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para REAL, armazenando em Result.

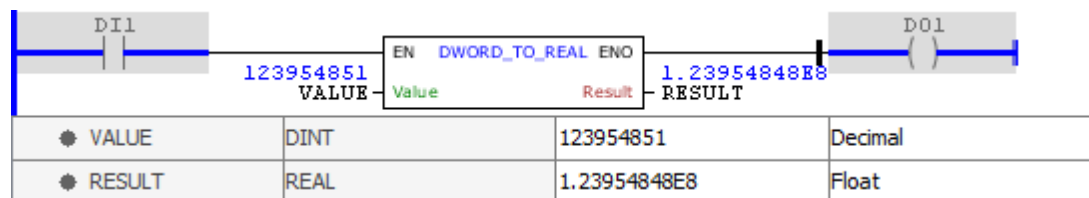
Quando EN possui valor FALSE, Result permanece inalterado.

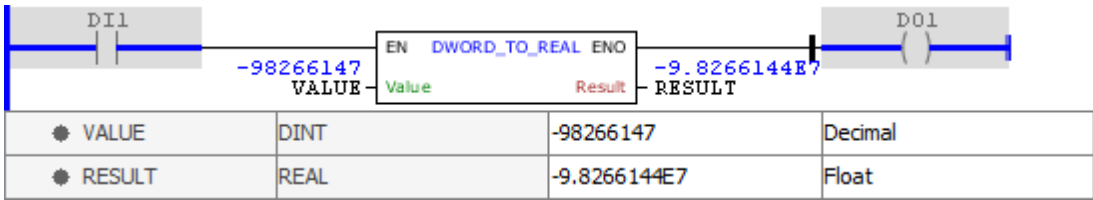
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

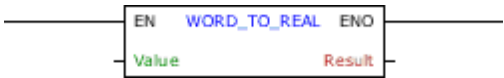




Os exemplos acima realizam a conversão da variável VALUE, em DWORD, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor WORD para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

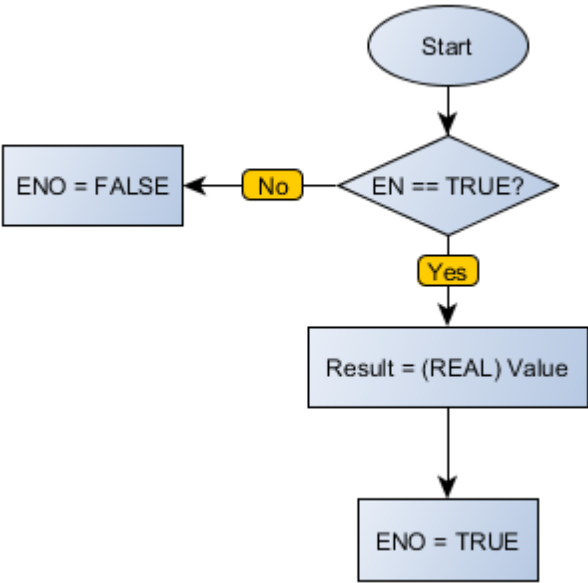
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para REAL, armazenando em Result.

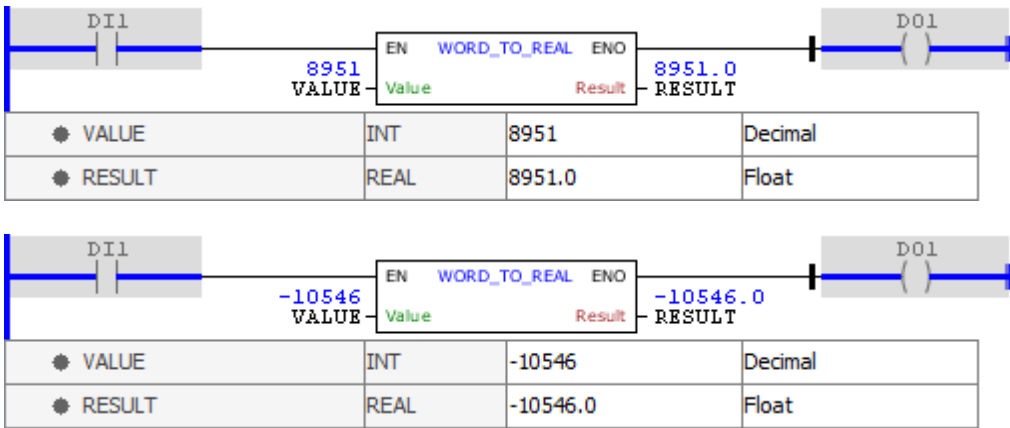
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



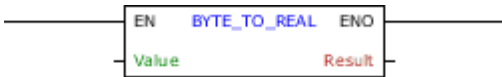
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

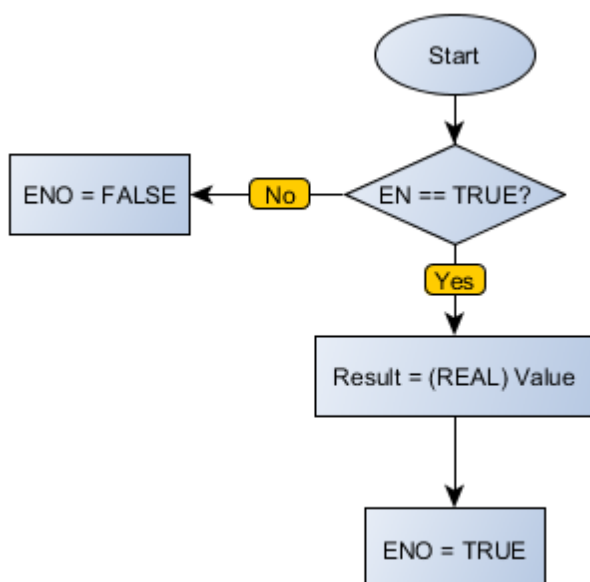
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para REAL, armazenando em Result.

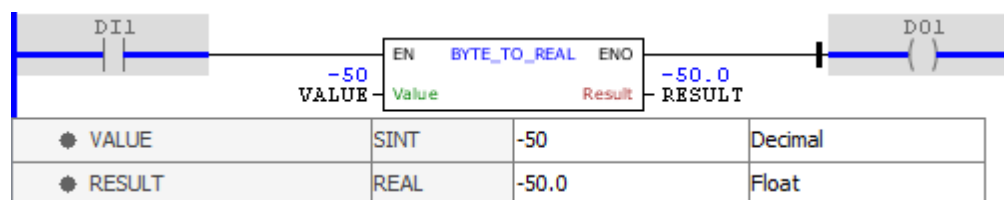
Quando EN possui valor FALSE, Result permanece inalterado.

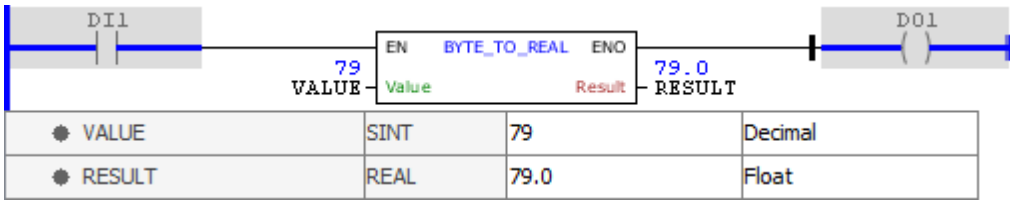
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

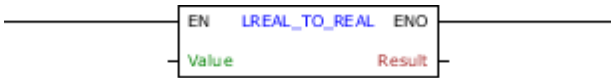




Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor REAL, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor LREAL para um valor REAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em REAL

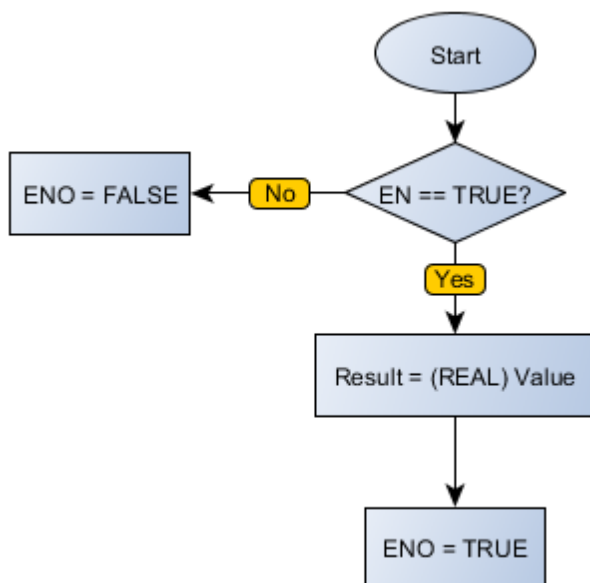
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para REAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

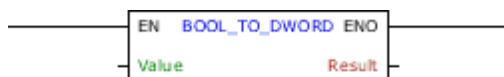
Fluxograma do Bloco



10.3.5.1.13.6 DWORD

Bloco que realiza a conversão de um valor BOOL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

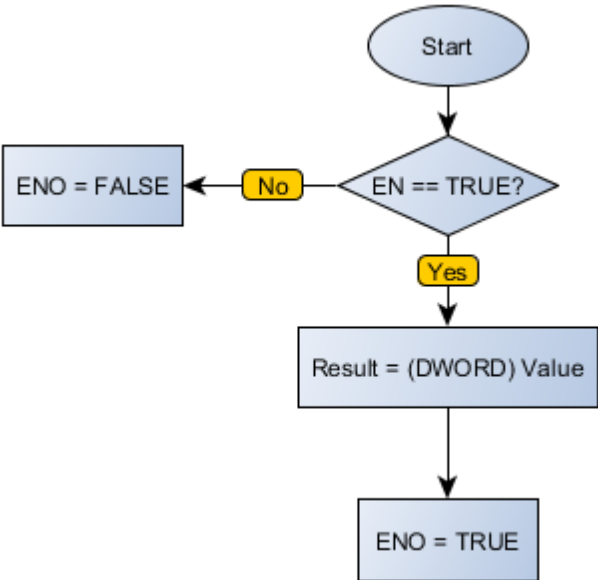
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para DWORD, armazenando em Result.

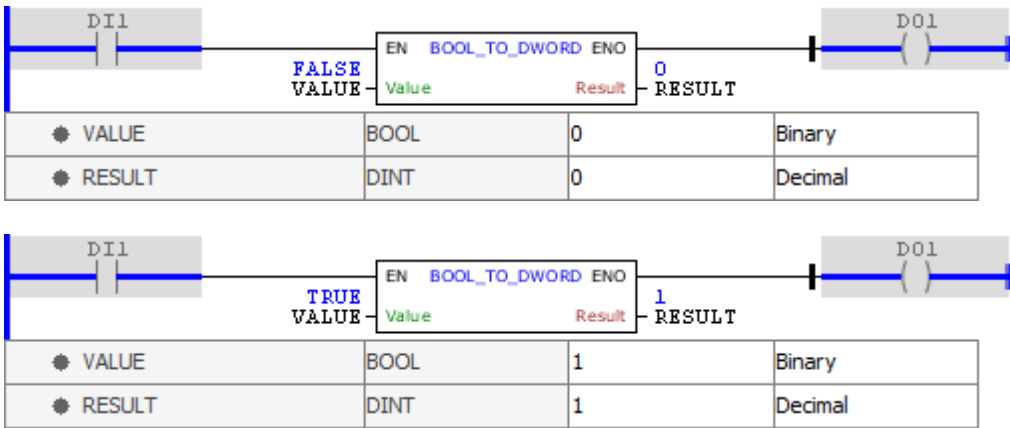
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



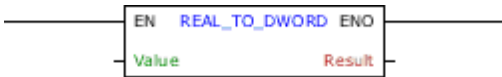
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em BOOL, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor REAL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

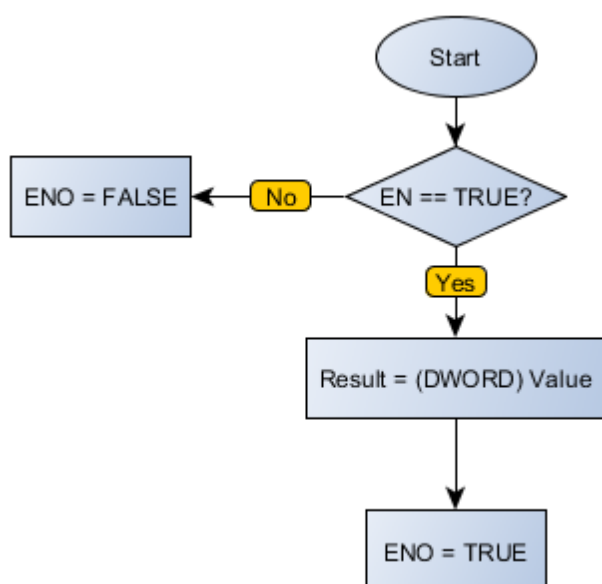
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para DWORD, armazenando em Result.

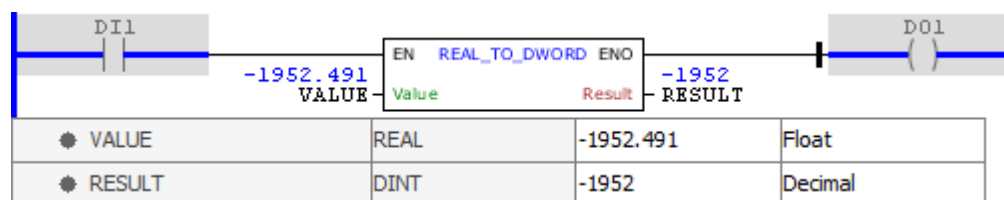
Quando EN possui valor FALSE, Result permanece inalterado.

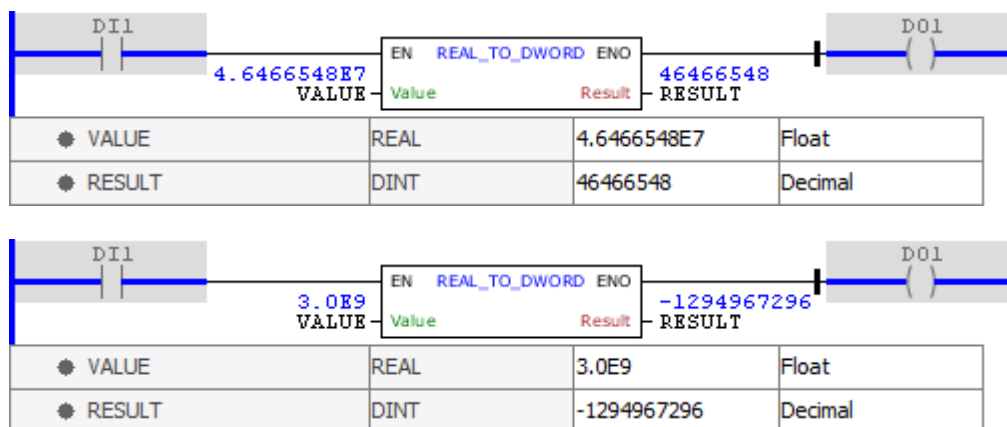
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

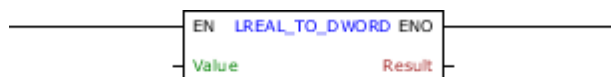




Os exemplos acima realizam a conversão da variável VALUE, em REAL, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que os resultados são truncados em decimal e são levados em consideração apenas os trinta e dois bits menos significativos.

Bloco que realiza a conversão de um valor LREAL para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	LREAL	Valor em LREAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

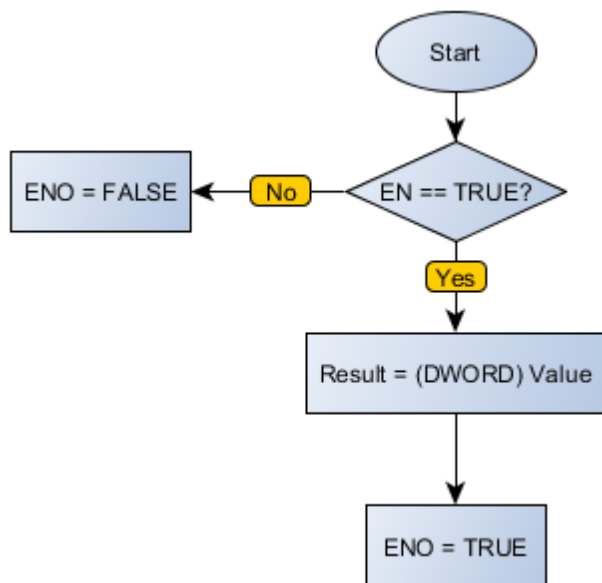
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como LREAL e realiza a conversão para DWORD, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

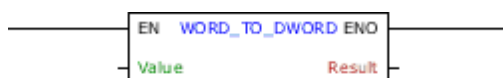
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor WORD para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

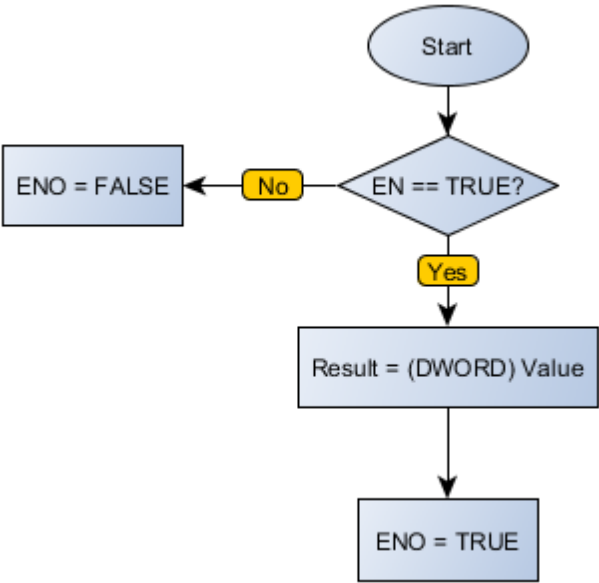
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para DWORD, armazenando em Result.

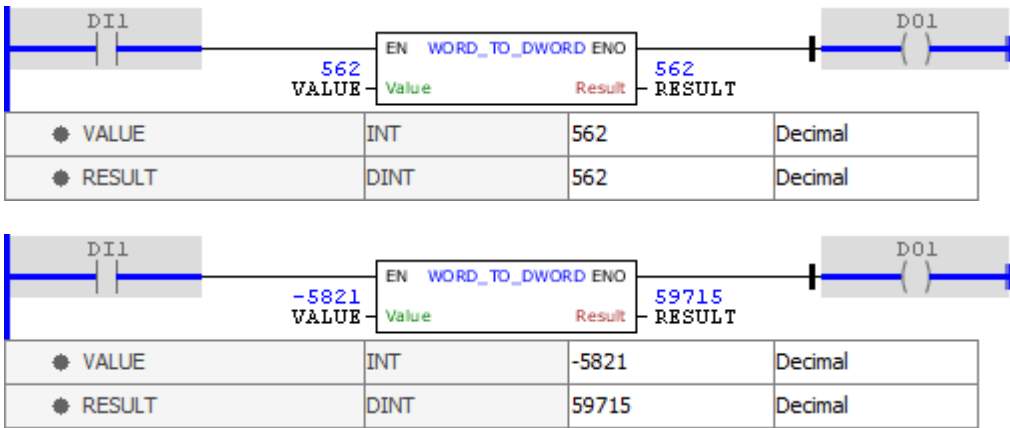
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



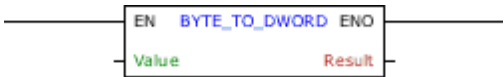
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em WORD, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor BYTE para um valor DWORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	DWORD UDINT DINT	Valor em DWORD

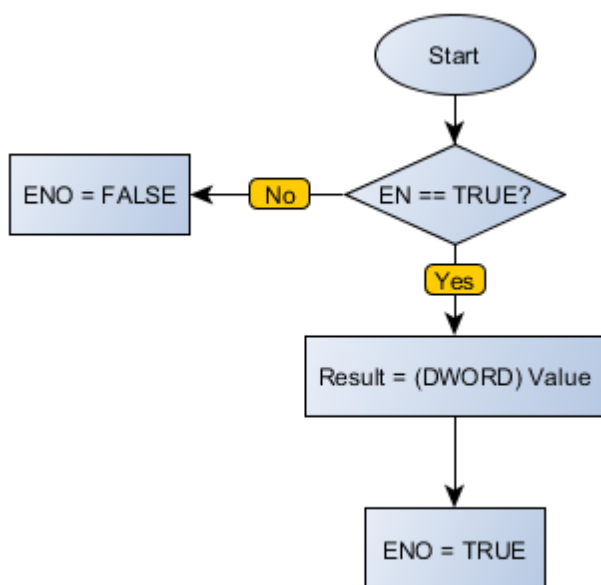
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para DWORD, armazenando em Result.

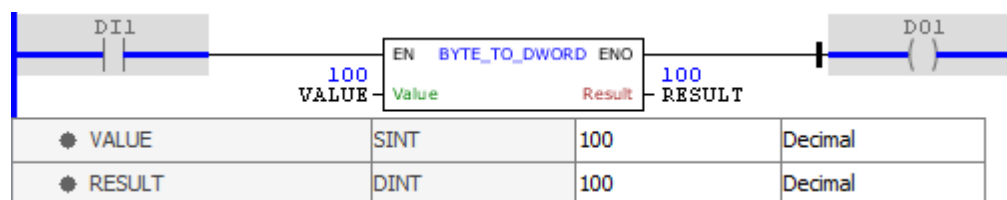
Quando EN possui valor FALSE, Result permanece inalterado.

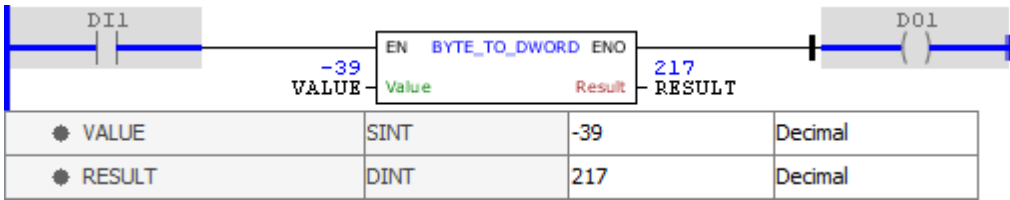
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



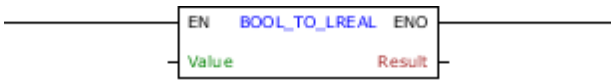


Os exemplos acima realizam a conversão da variável VALUE, em BYTE, para um valor DWORD, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.13.7 LREAL

Bloco que realiza a conversão de um valor BOOL para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BOOL	Valor em BOOL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

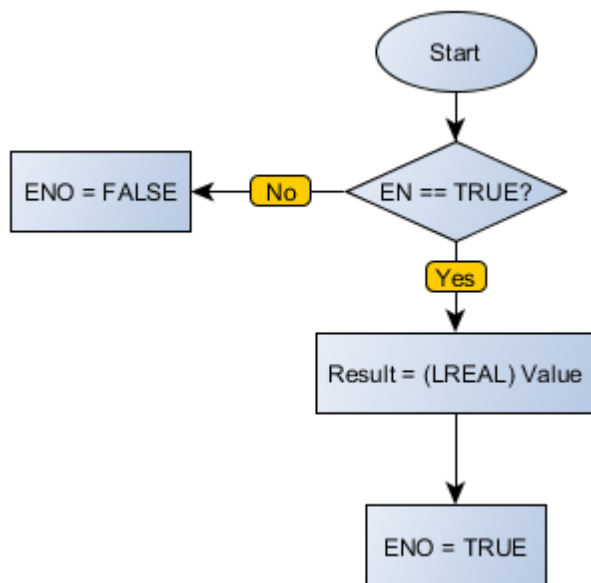
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BOOL e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

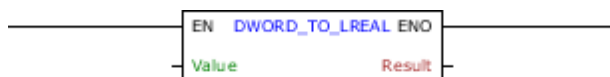
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor DWORD para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	DWORD UDINT DINT	Valor em DWORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

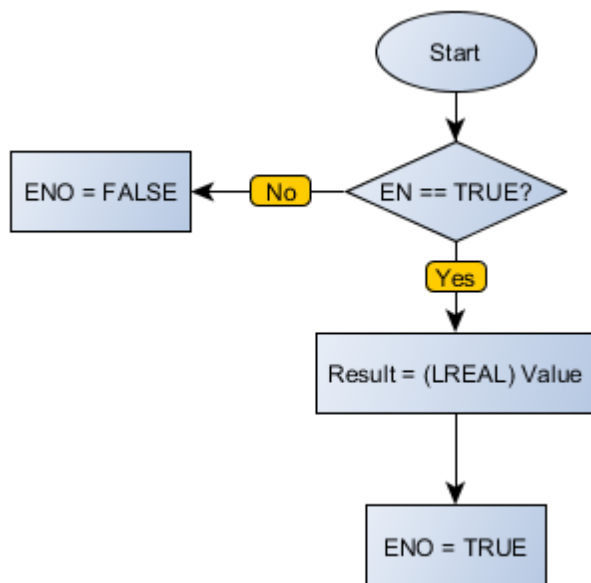
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como DWORD e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

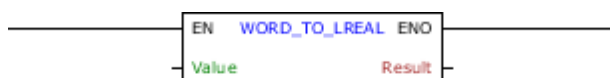
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor WORD para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT	Valor em WORD
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

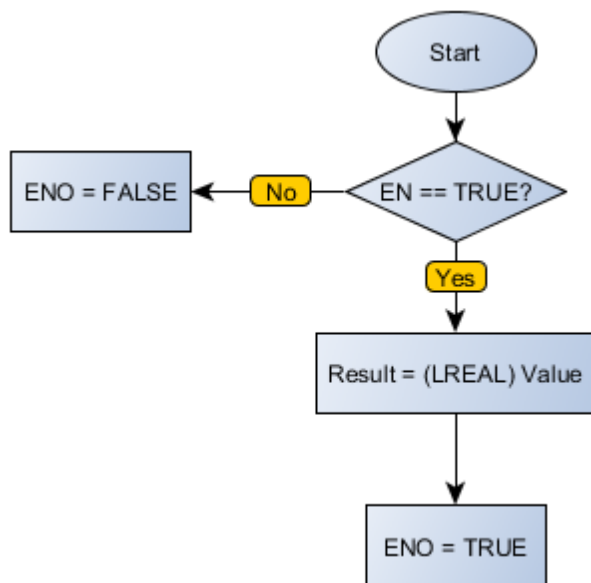
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como WORD e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

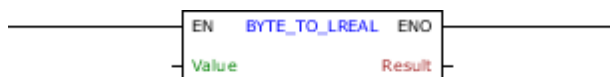
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor BYTE para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	BYTE USINT SINT	Valor em BYTE
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

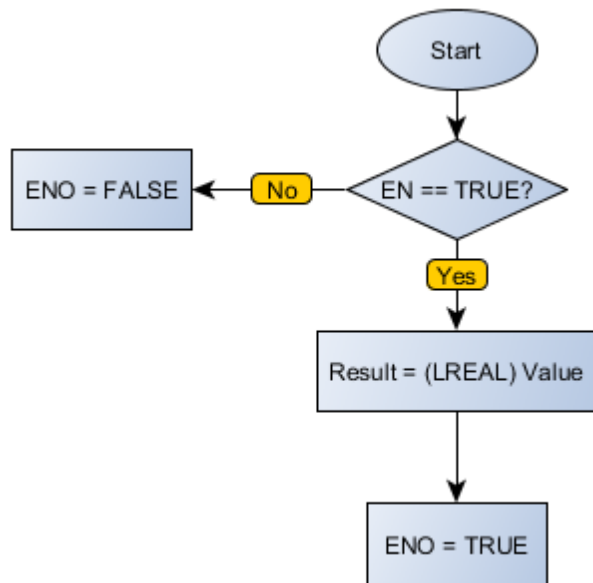
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como BYTE e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

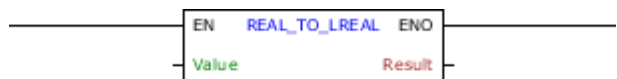
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Bloco que realiza a conversão de um valor REAL para um valor LREAL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em REAL
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	LREAL	Valor em LREAL

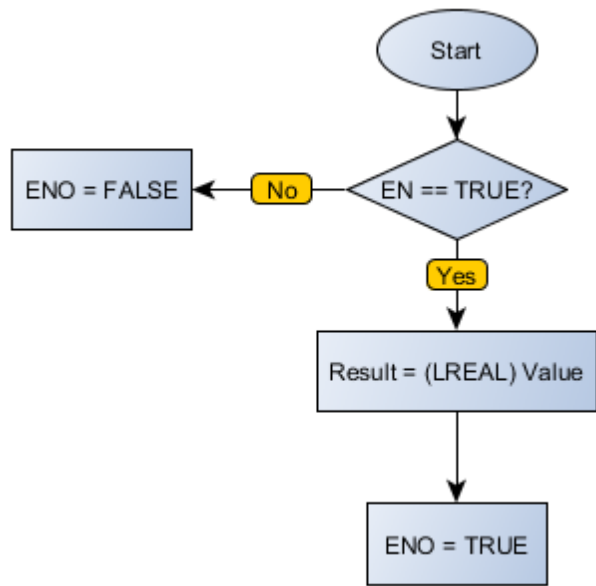
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como REAL e realiza a conversão para LREAL, armazenando em Result.

Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



10.3.5.1.13.8 Rad-Deg

Bloco que realiza a conversão de um valor em radianos para um valor em graus.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	B00L	Habilitação do bloco
	Value	REAL	Valor em radianos
VAR_OUTPUT	ENO	B00L	Fim de operação
	Result	REAL	Valor em graus

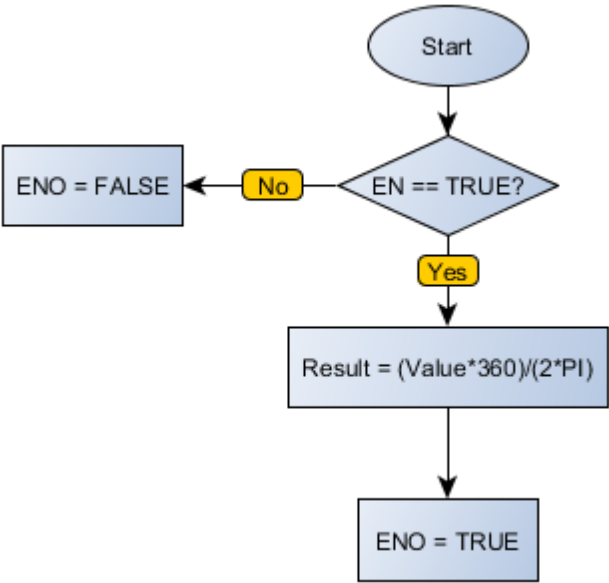
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como em radianos e realiza a conversão para graus, armazenando em Result.

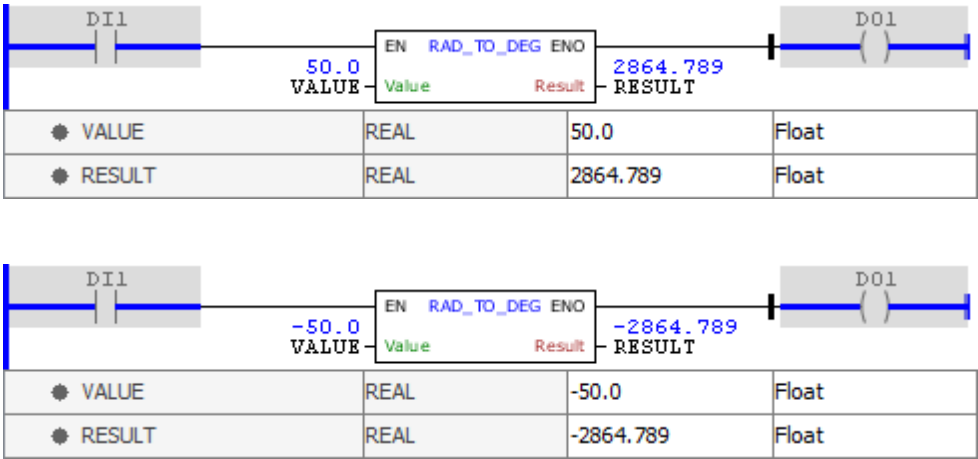
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



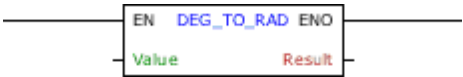
Exemplo



Os exemplos acima realizam a conversão da variável VALUE, em radianos, para um valor correspondente em graus, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

Bloco que realiza a conversão de um valor em graus para um valor em radianos.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	REAL	Valor em graus
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	REAL	Valor em radianos

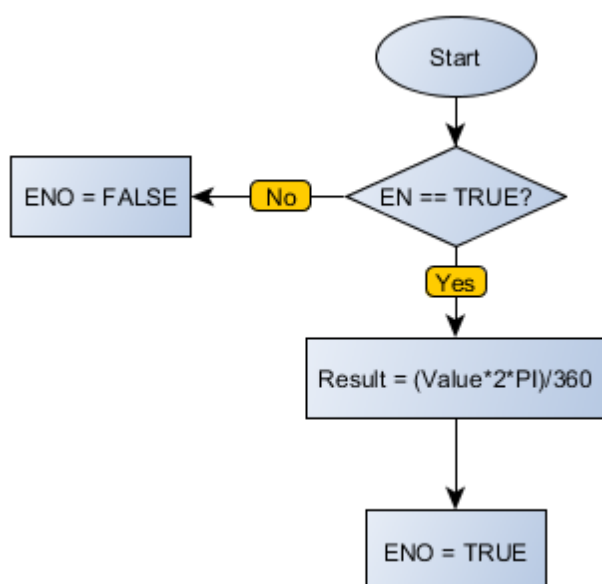
Funcionamento

Este bloco, quando tem um valor TRUE em EN, interpreta o valor de Value como em graus e realiza a conversão para radianos, armazenando em Result.

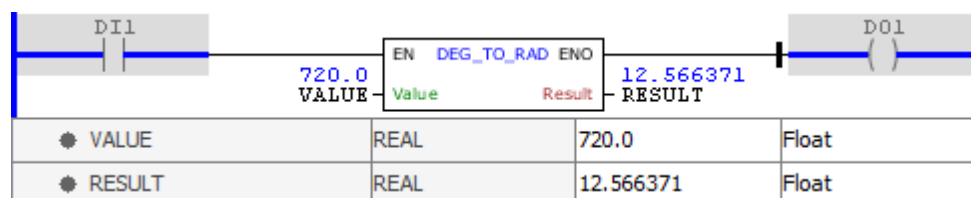
Quando EN possui valor FALSE, Result permanece inalterado.

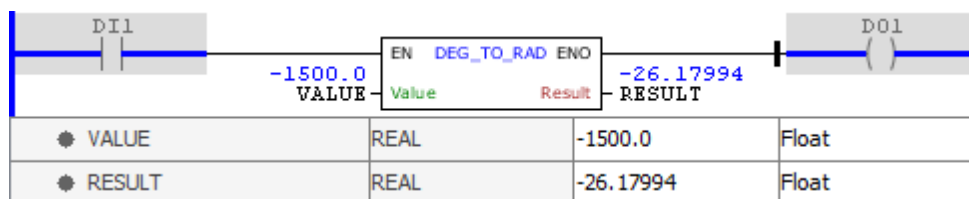
O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo





Os exemplos acima realizam a conversão da variável VALUE, em graus, para um valor correspondente em radianos, armazenando o resultado final em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14 Data Transfer

10.3.5.1.14.1 STORE

Bloco que realiza o armazenamento direto de dados provenientes de uma fonte em um destino.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SRC	BYTE USINT SINT WORD UINT INT DWORD DINT REAL	Fonte de dados
VAR_OUTPUT	ENO	BOOL	Fim de operação
	DST	BYTE USINT SINT WORD UINT INT DWORD DINT REAL	Destino de dados

Funcionamento

Este bloco, quando tem um valor TRUE em EN, armazena o conteúdo de SRC em DST.



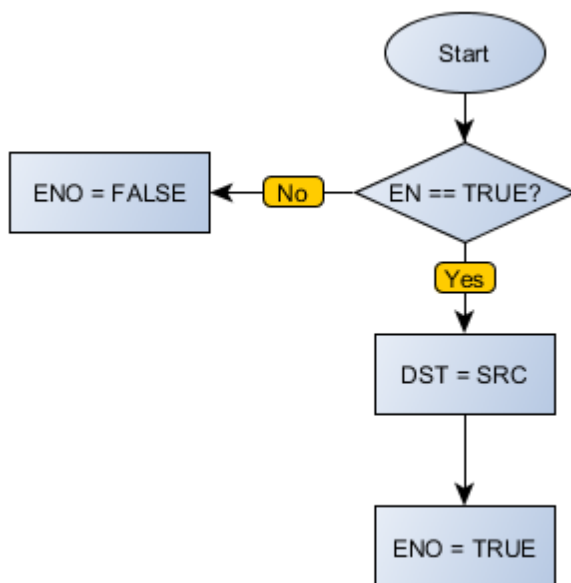
NOTA!

SRC e DST devem possuir tipos de dados de mesmo tamanho.

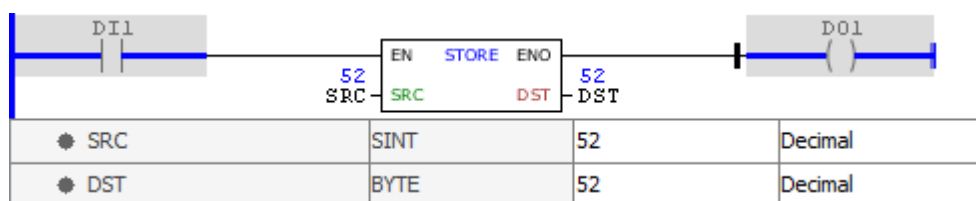
Quando EN possui valor FALSE, DST permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

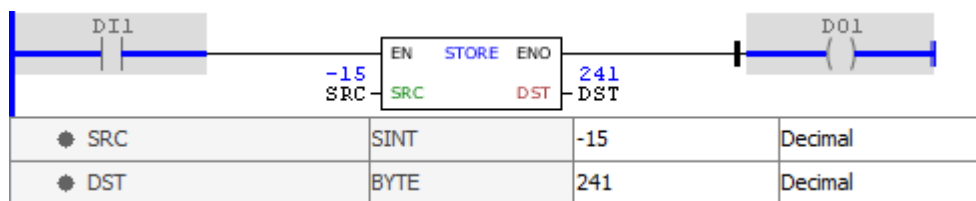
Fluxograma do Bloco



Exemplo



O exemplo acima armazena o valor da variável SRC em DST. O bloco finaliza com sucesso e a saída ENO é ativada.



O exemplo acima armazena o valor da variável SRC em DST. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário é mantido entre variáveis de tipos diferentes.

10.3.5.1.14.2 ILOADBOOL

Bloco que carrega indiretamente o valor de um bit em um endereço de variável global.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Bit	BYTE USINT SINT	Posição do bit a ser verificado
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Value	BOOL	Valor do bit selecionado pelos argumentos de entrada

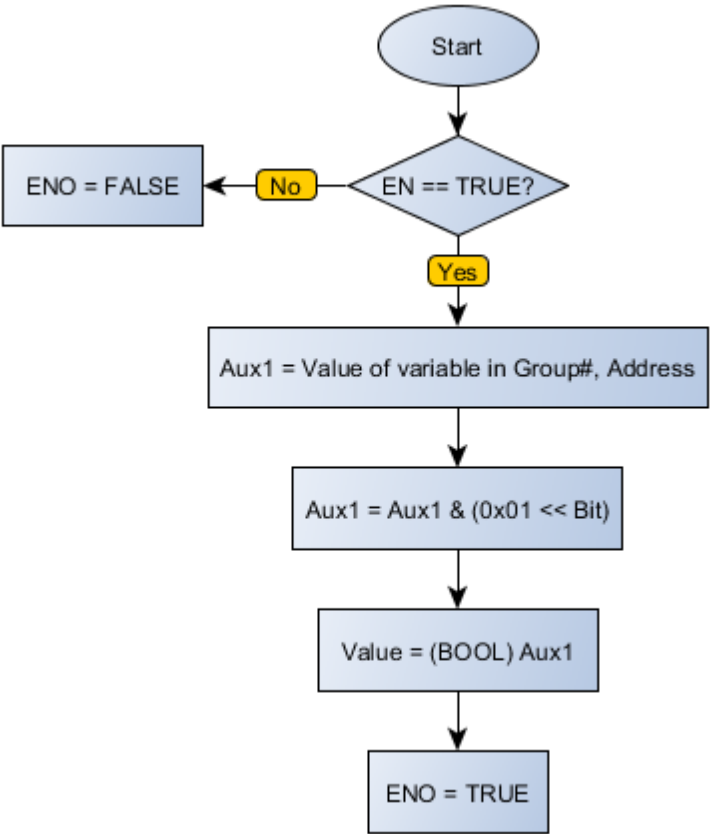
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega, em Value, o conteúdo do Bit da variável Address pertencente ao grupo Group#.

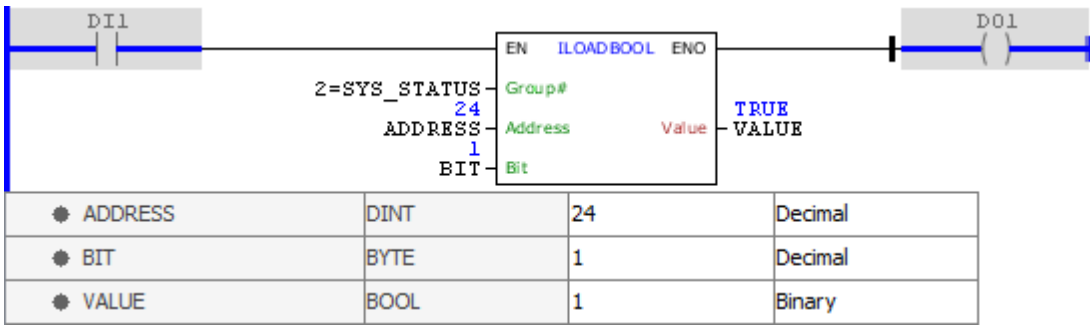
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

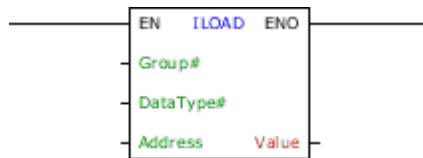


O exemplo acima carrega o valor do bit 1 do endereço 24 do grupo 2 (GLOBAL_SYSTEM %S) para a variável VALUE. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14.3 ILOAD

Bloco que carrega indiretamente o valor de uma variável e transfere para Value.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	DataType#	BYTE	Tipo de dado da variável selecionada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Value	Conforme selecionado em DataType#	Valor da variável selecionada

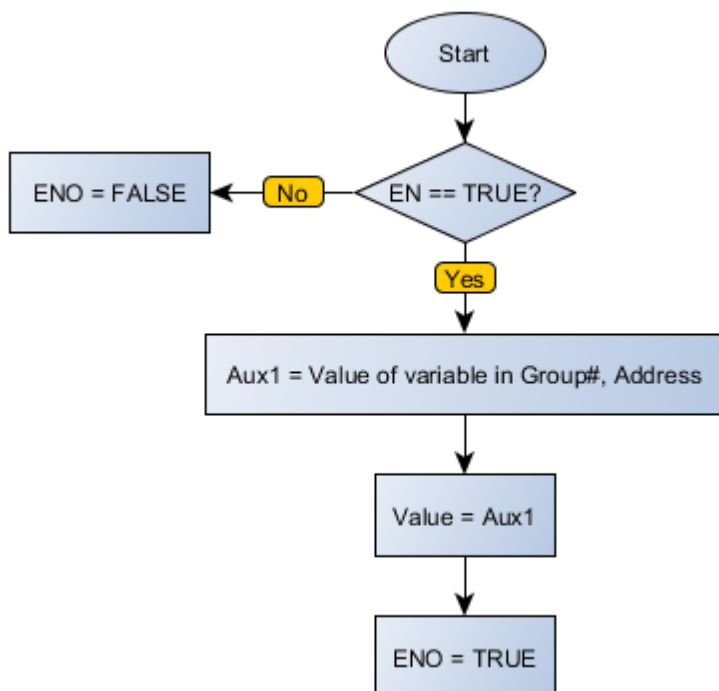
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega, em Value, o conteúdo da variável Address pertencente ao grupo Group#, segundo o DataType# escolhido.

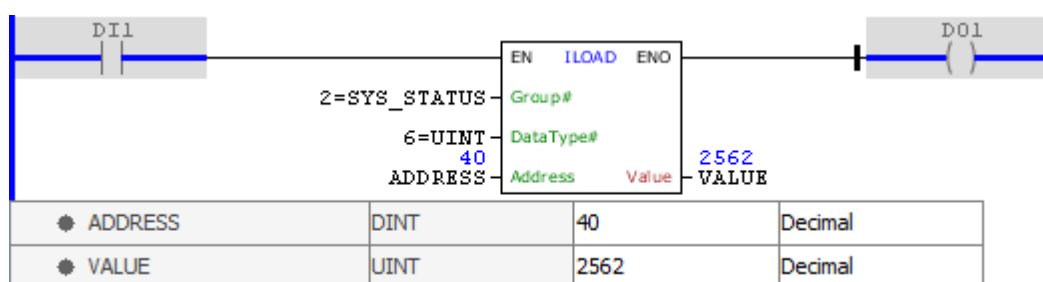
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima carrega o valor do endereço 40 do grupo 2 (GLOBAL_SYSTEM %S) em formato UINT para a variável VALUE. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14.4 ISTOREBOOL

Bloco que armazena indiretamente o valor de Value em um bit em um endereço de variável global.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Bit	BYTE USINT SINT	Posição do bit a ser modificado
	Value	BOOL	Novo valor do bit selecionado
VAR_OUTPUT	ENO	BOOL	Fim de operação

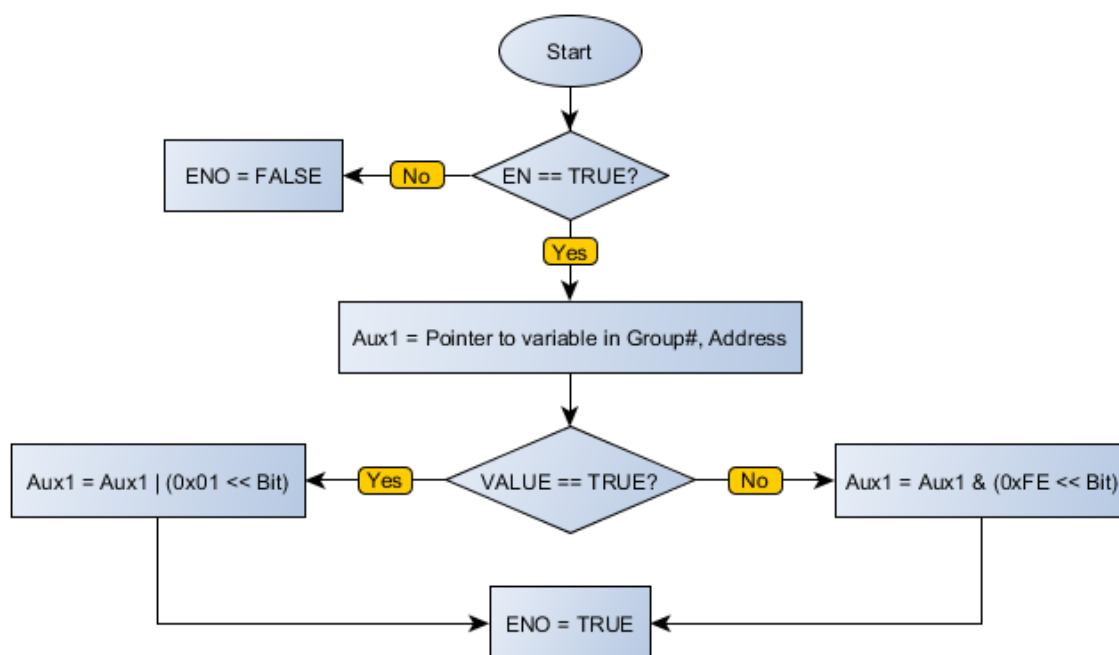
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega o valor de Value no conteúdo do Bit da variável Address pertencente ao grupo Group#.

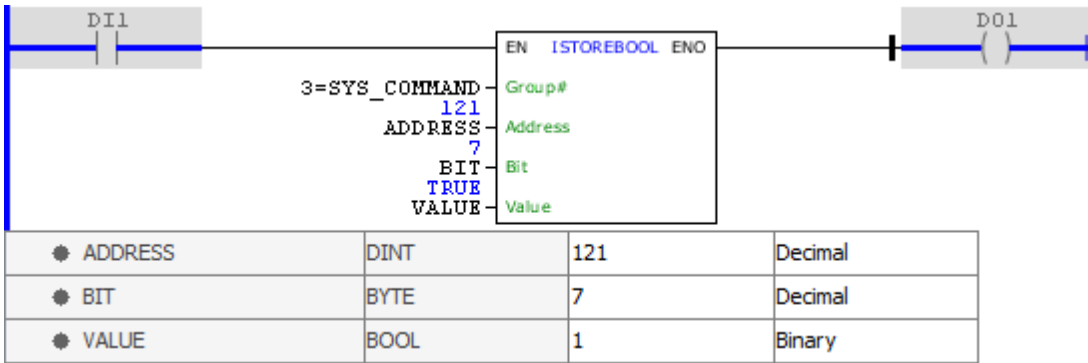
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

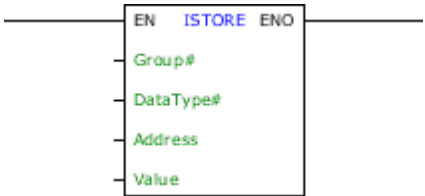


O exemplo acima armazena o valor de VALUE no bit 7 do endereço 121 do grupo 3 (GLOBAL_SYSTEM %C). O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14.5 ISTORE

Bloco que armazena indiretamente o valor de Value em uma variável.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Group#	BYTE	Grupo em que a variável está armazenada
	DataType#	BYTE	Tipo de dado da variável selecionada
	Address	DWORD UDINT DINT	Endereço da variável global, conforme seu grupo
	Value	Conforme selecionado em DataType#	Valor a ser escrito na variável selecionada
VAR_OUTPUT	ENO	BOOL	Fim de operação

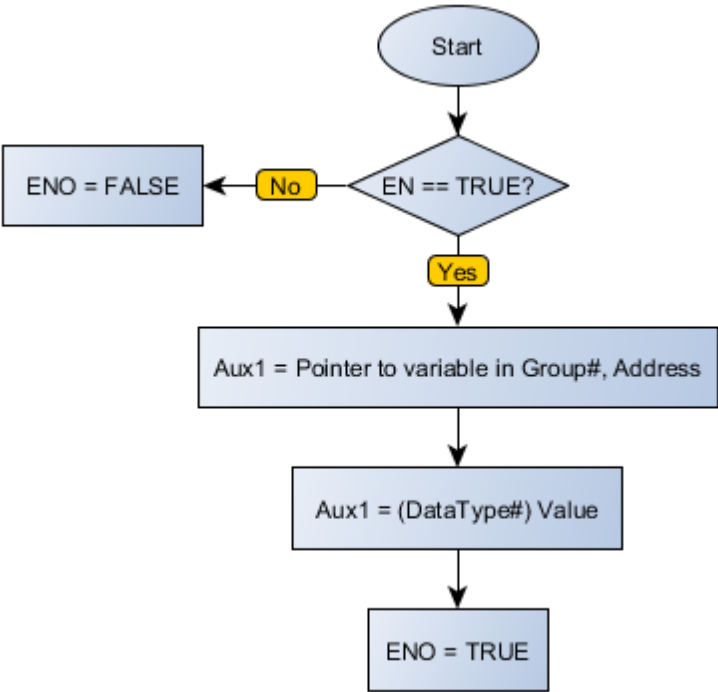
Funcionamento

Este bloco, quando tem um valor TRUE em EN, carrega o valor de Value no conteúdo da variável Address pertencente ao grupo Group#, segundo o DataType# escolhido.

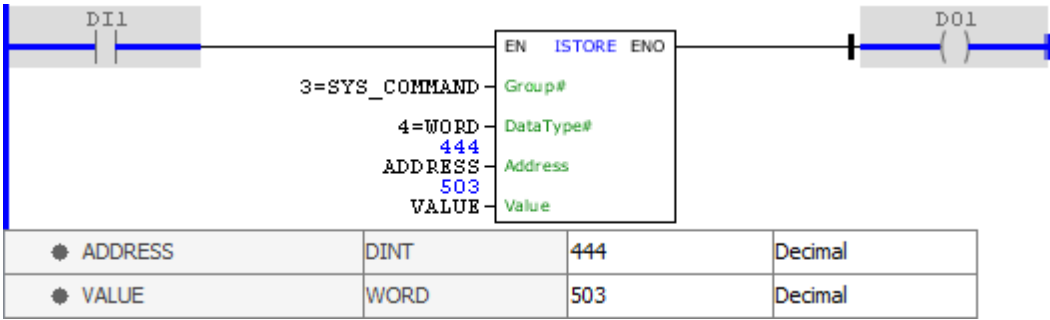
Quando EN possui valor FALSE, Value permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

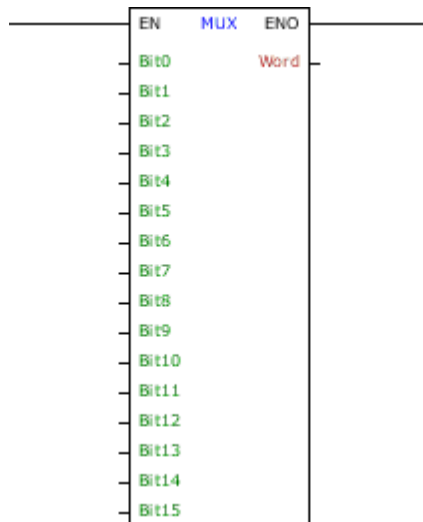


O exemplo acima armazena o valor de VALUE em formato WORD no endereço 444 do grupo 3 (GLOBAL_SYSTEM %C). O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14.6 MUX

Bloco que cria uma nova variável WORD a partir da concatenação de 15 variáveis BOOL.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Bit0 - Bit15	BOOL	Bit da posição correspondente na nova palavra
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Word	WORD UINT INT	Nova palavra formada a partir dos bits de entrada

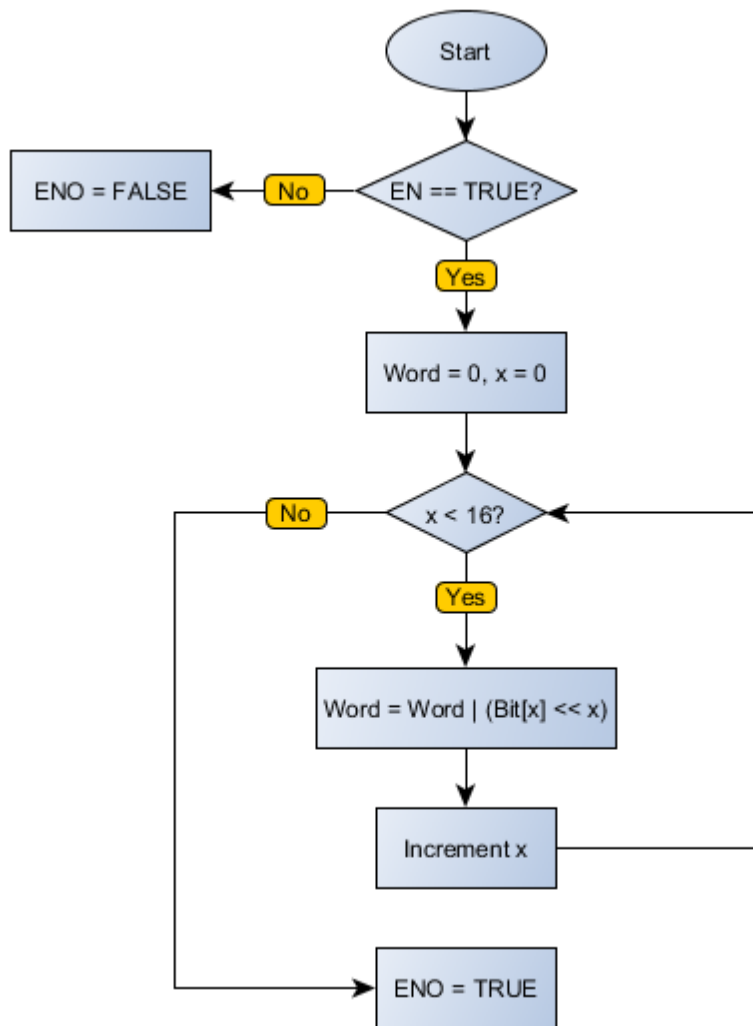
Funcionamento

Este bloco, quando tem um valor TRUE em EN, concatena os valores booleanos das variáveis de entrada e armazena este valor na variável Word. O Bit0 corresponde ao LSB (bit menos significativo) e Bit15 corresponde ao MSB (bit mais significativo).

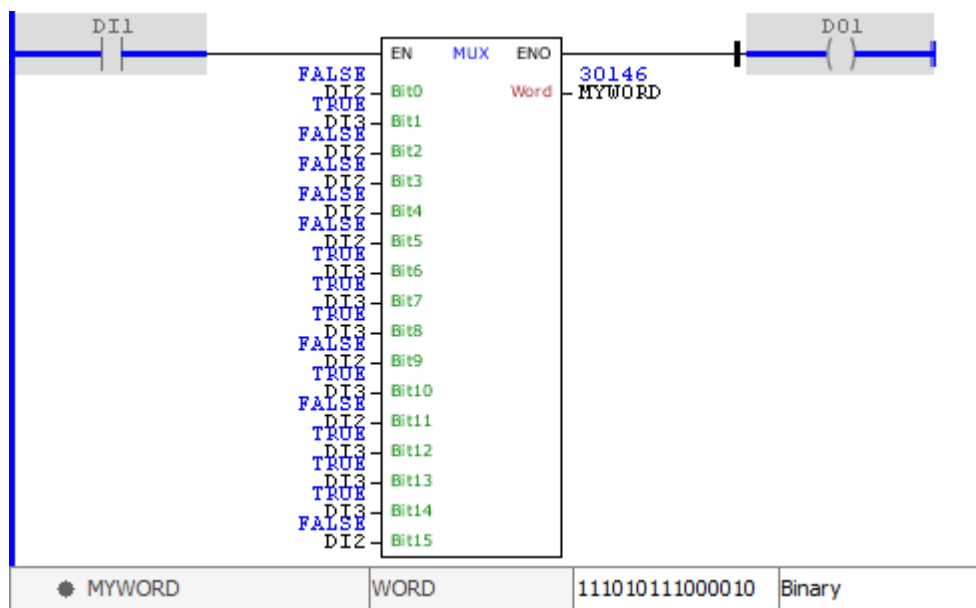
Quando EN possui valor FALSE, Word permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo

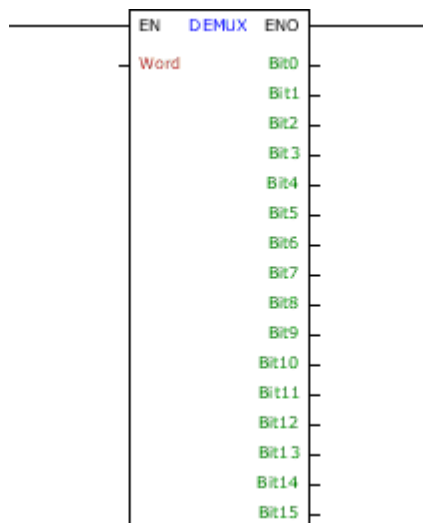


O exemplo acima concatena os valores booleanos dos bits de entrada do bloco, formando a palavra de saída armazenada em MYWORD. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14.7 DEMUX

Bloco que cria 15 novas variáveis BOOL a partir da decomposição de uma variável WORD.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Word	WORD UINT INT	Variável de entrada de 15 bits
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Bit0 - Bit15	BOOL	Bit da posição correspondente de Word

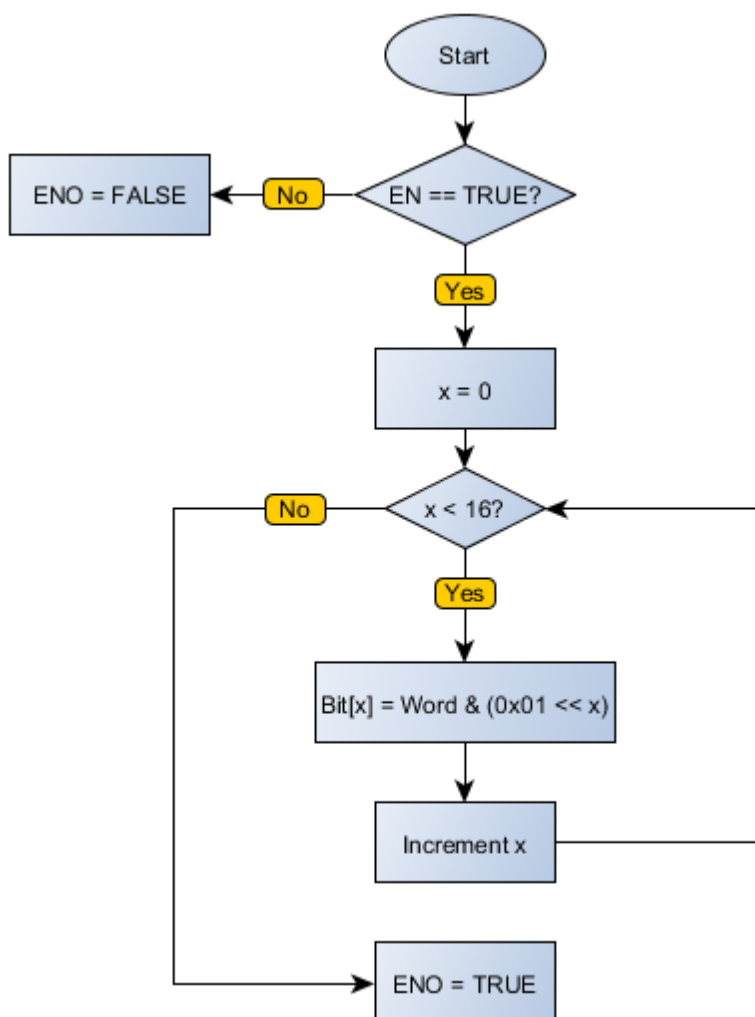
Funcionamento

Este bloco, quando tem um valor TRUE em EN, decompõe a variável de entrada Word em 15 valores booleanos armazenados nas variáveis Bit0 a Bit15. O Bit0 corresponde ao LSB (bit menos significativo) e Bit15 corresponde ao MSB (bit mais significativo).

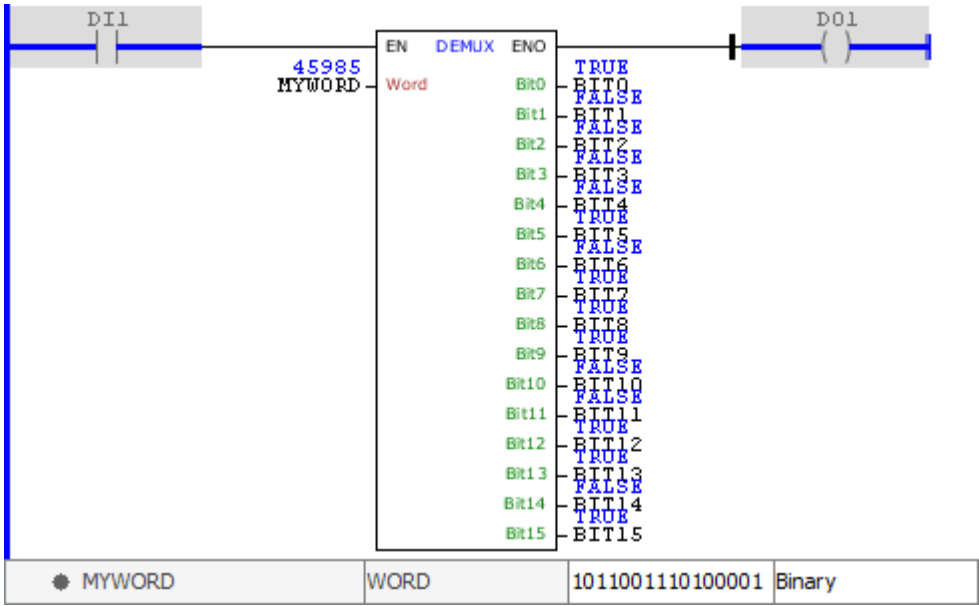
Quando EN possui valor FALSE, as variáveis de saída permanecem inalteradas.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco



Exemplo



O exemplo acima decompõe o valor de MYWORD em valores booleanos, sendo estes armazenados nas variáveis de saída BIT0 a BIT15. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14.8 SEL

Bloco que replica para a saída o valor de uma das variáveis de entrada (Value0 ou Value1) de acordo com a seleção de Selector.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Selector	BOOL	Variável que seleciona a entrada
	Value0	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Entrada multiplexada número 1
	Value1	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Entrada multiplexada número 2
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saída selecionado

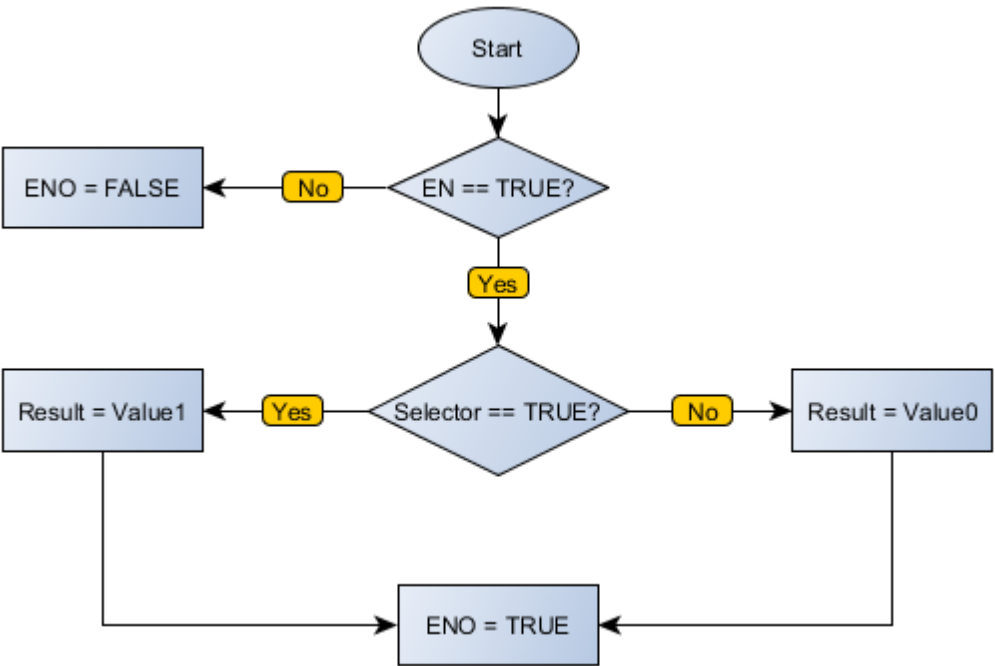
Funcionamento

Este bloco, quando tem um valor TRUE em EN, replica para a variável Result o valor de Value0 caso Selector seja FALSE ou o valor de Value1 caso Selector seja TRUE.

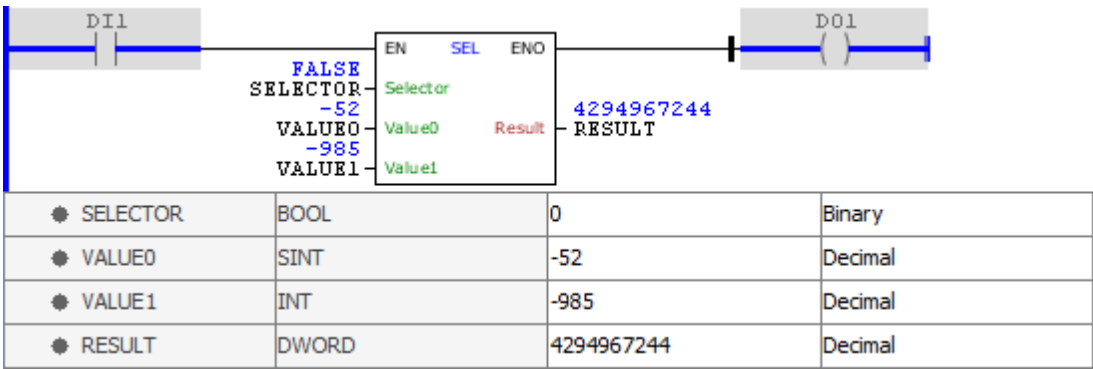
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Fluxograma do Bloco

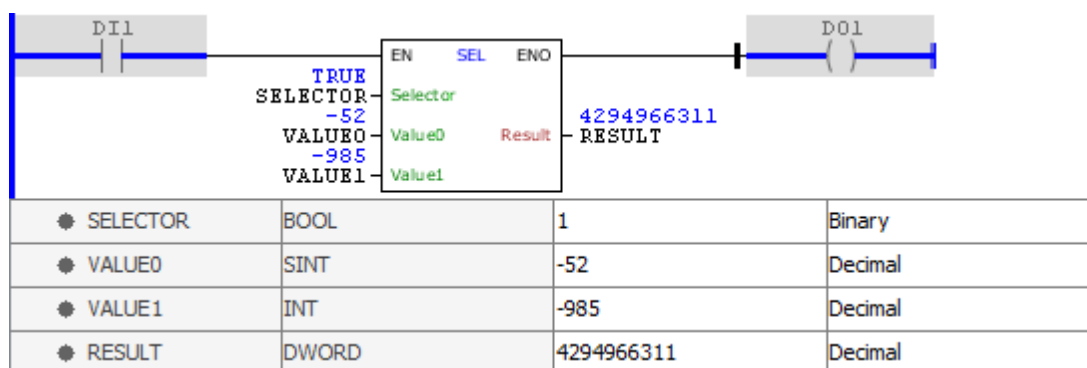


Exemplo



O exemplo acima utiliza a varável SELECTOR como seletor do canal de multiplexação. Quando ela está em nível FALSE, a saída RESULT recebe o valor de VALUE0. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário foi mantido, apesar de a representação decimal

estar corrompida, dado que DWORD não aceita valores negativos.



O exemplo acima utiliza a variável SELECTOR como seletor do canal de multiplexação. Quando ela está em nível TRUE, a saída RESULT recebe o valor de VALUE1. O bloco finaliza com sucesso e a saída ENO é ativada. Observe que o padrão binário foi mantido, apesar de a representação decimal estar corrompida, dado que DWORD não aceita valores negativos.

10.3.5.1.14.9 SWAP

Bloco que realiza uma permuta entre os bytes ímpares e os bytes pares consecutivos em Value e envia o valor para Result.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Value	WORD UINT INT DWORD UDINT DINT	Variável de entrada a ser permutada
VAR_OUTPUT	ENO	BOOL	Fim de operação
	Result	BYTE USINT SINT WORD UINT INT DWORD UDINT DINT REAL	Valor de saída

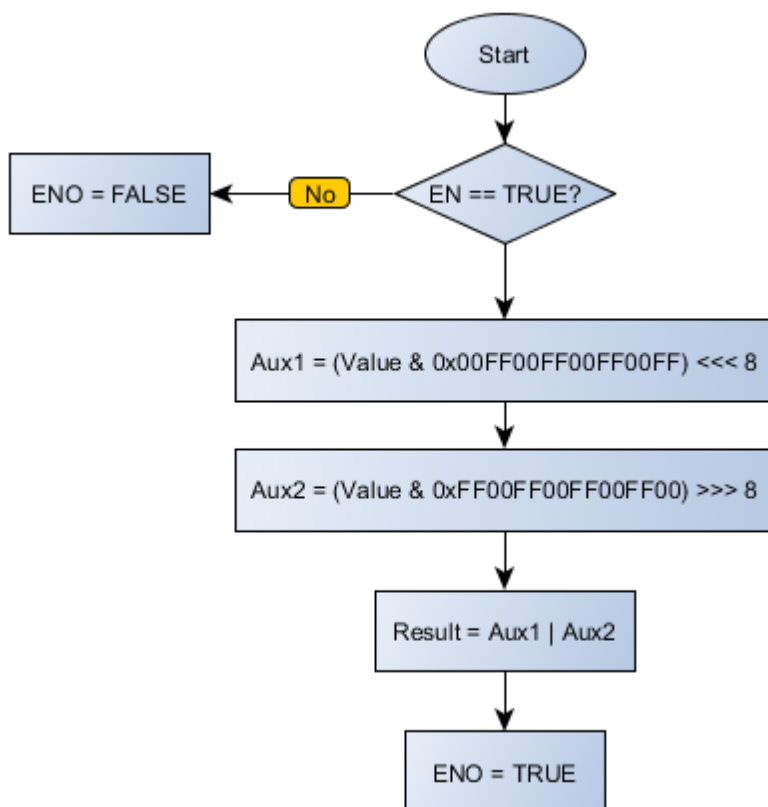
Funcionamento

Este bloco, quando tem um valor TRUE em EN, troca os valores entre os bytes ímpares (1, 3, 5 e 7) e os bytes pares consecutivos (2, 4, 6 e 8, respectivamente) da variável Value, armazenando o resultado em Result.

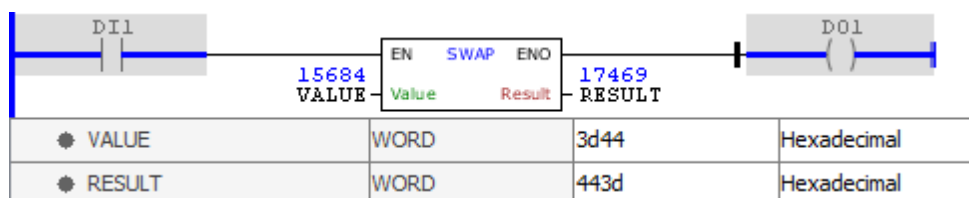
Quando EN possui valor FALSE, Result permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

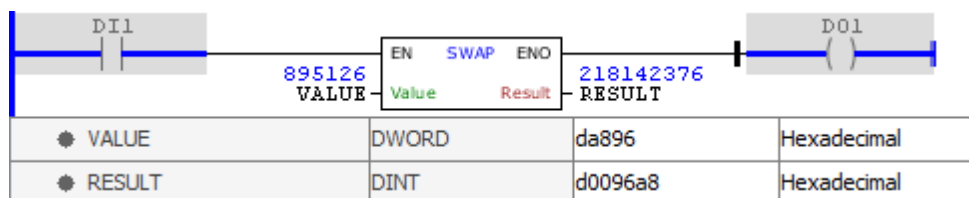
Fluxograma do Bloco



Exemplo



O exemplo troca a posição do valor do byte 1 de VALUE (0x44) com o byte 2 de VALUE (0x3D), armazenando o resultado final (0x44_3D) em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

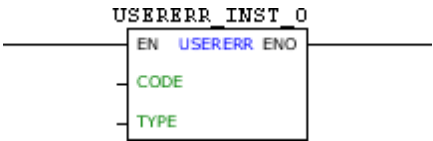


O exemplo troca a posição do valor do byte 1 de VALUE (0x96) com o byte 2 de VALUE (0xA8) e do byte 3 de VALUE (0x0D) com o byte 4 de VALUE (0x00), armazenando o resultado final (0x0D_00_96_A8) em RESULT. O bloco finaliza com sucesso e a saída ENO é ativada.

10.3.5.1.14.10 USERERR

Bloco que gera um alarme ou falha com o número programado pelo usuário.

Representação Ladder



Estrutura do Bloco

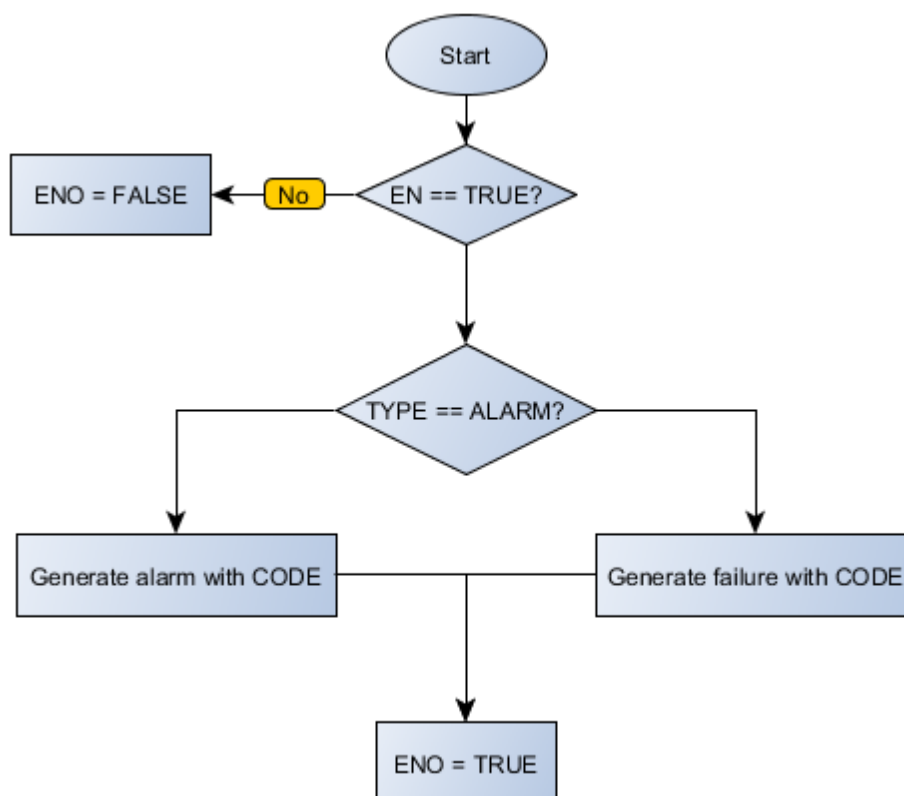
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	CODE	WORD UINT	Código do erro gerado (950 - 999)
	TYPE	BYTE	Tipo de erro gerado (0 - Alarme) (1 - Falha)
VAR_OUTPUT	ENO	BOOL	Sucesso na geração do erro
VAR	USERERR_INST_0	USERERR	Instância de acesso à estrutura do bloco

Funcionamento

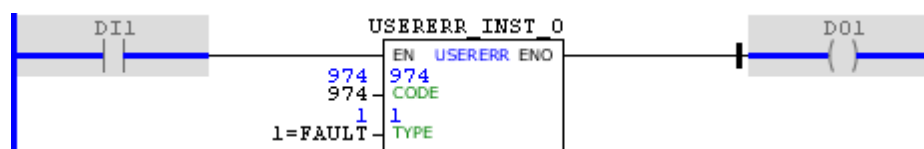
Este bloco, quando tem um valor TRUE em EN, gera um alarme ou falha no equipamento, dependendo do tipo definido em TYPE com o código CODE.

O valor de ENO informa se a geração do alarme ou falha foi executada com sucesso.

Fluxograma do Bloco



Exemplo



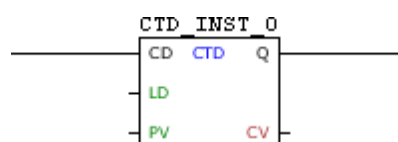
O exemplo acima, ao identificar nível TRUE em DI1, gera uma falha com o código 974 e seta a saída DO1.

10.3.5.1.15 Counter

10.3.5.1.15.1 CTD

Bloco de contagem regressiva de pulsos de entrada.

Representação Ladder



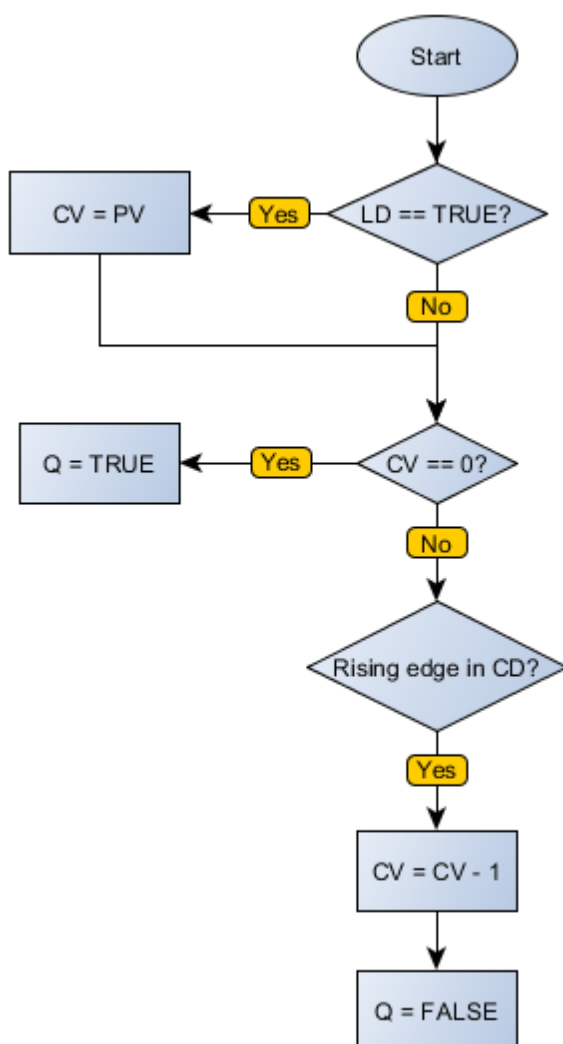
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	CD	BOOL	Identificador de pulsos
	LD	BOOL	Carrega o valor de PV em CV
	PV	WORD UINT	Valor de configuração inicial
VAR_OUTPUT	Q	BOOL	Sinalizador de contador zerado
	CV	WORD UINT	Valor de contagem atual
VAR	CTD_INST_0	CTD	Instância de acesso à estrutura do bloco

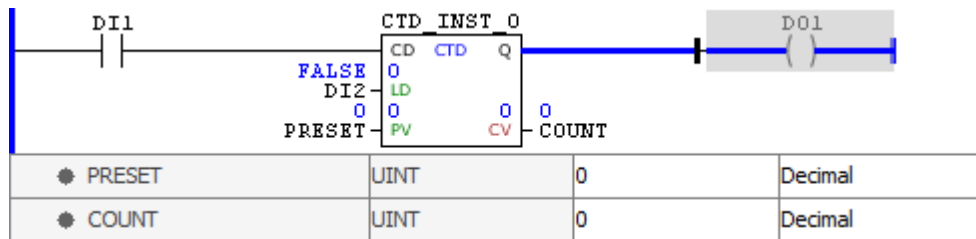
Funcionamento

Este bloco, quando identifica uma borda de subida em CD, decrementa a variável CV até que a mesma seja zero. Enquanto CV for igual a zero, a saída Q permanece em nível TRUE. Ao detectar nível alto em LD, o bloco carrega o valor de PV em CV.

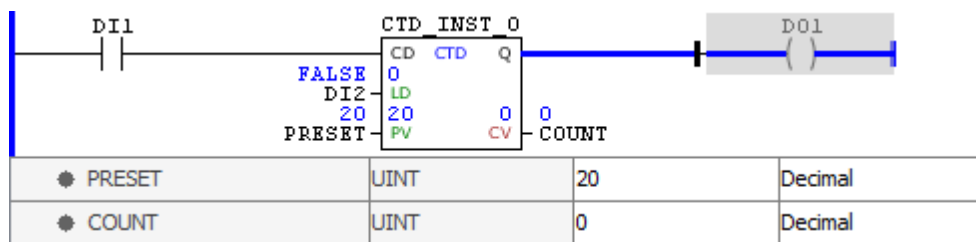
Fluxograma do Bloco



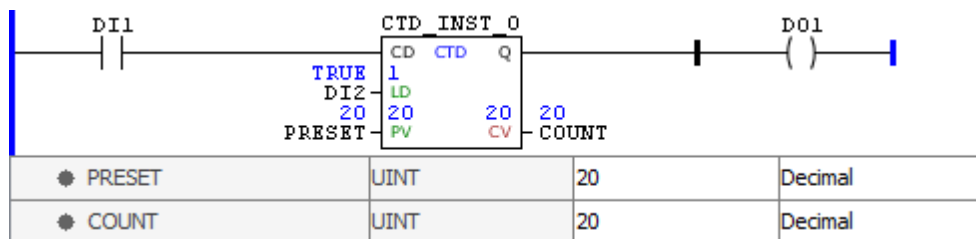
Exemplo



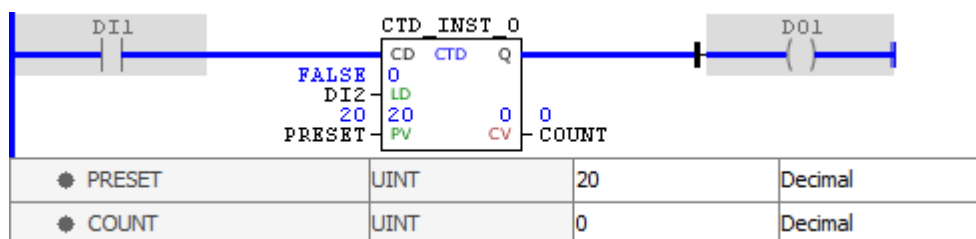
O exemplo acima exibe as condições iniciais da rotina. Como CV possui o valor zero, a saída Q é habilitada.



O valor da variável PV foi alterado para 20, porém ainda não foi carregado.



Ao identificar o nível TRUE em LD, o bloco carrega o valor de PV para CV. Como este valor é maior que zero, a saída Q é desabilitada.

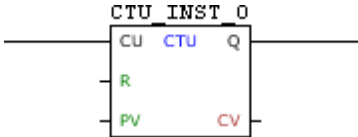


A cada borda de subida identificada em CD, o valor de COUNT é decrementado até chegar a zero, quando a saída Q é habilitada.

10.3.5.1.15.2 CTU

Bloco de contagem progressiva de pulsos de entrada.

Representação Ladder



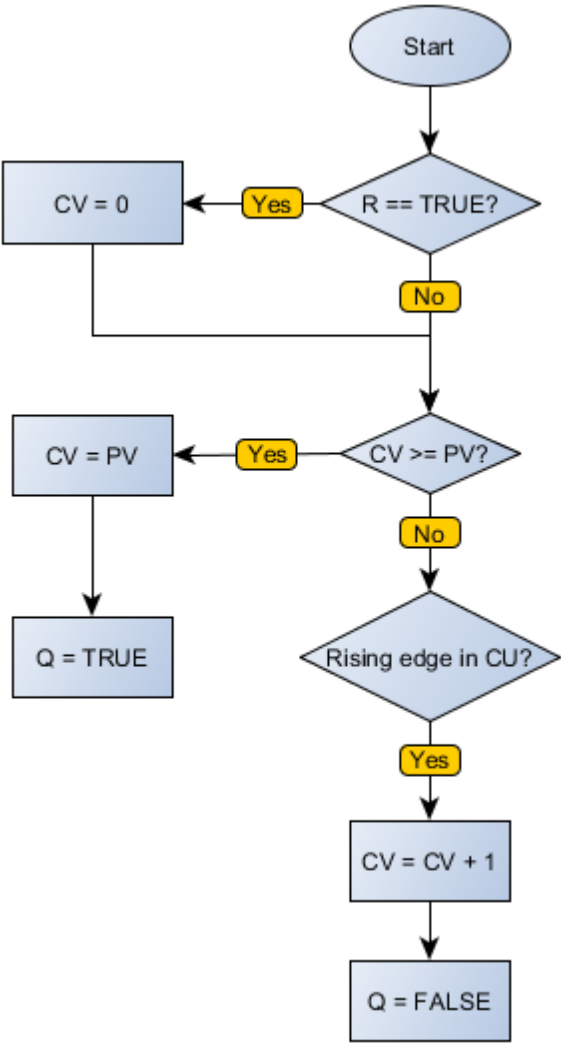
Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	CU	BOOL	Identificador de pulsos
	R	BOOL	Carrega o valor zero em CV
	PV	WORD UINT	Valor máximo de contagem
VAR_OUTPUT	Q	BOOL	Sinalizador de contador estourado
	CV	WORD UINT	Valor de contagem atual
VAR	CTU_INST_0	CTU	Instância de acesso à estrutura do bloco

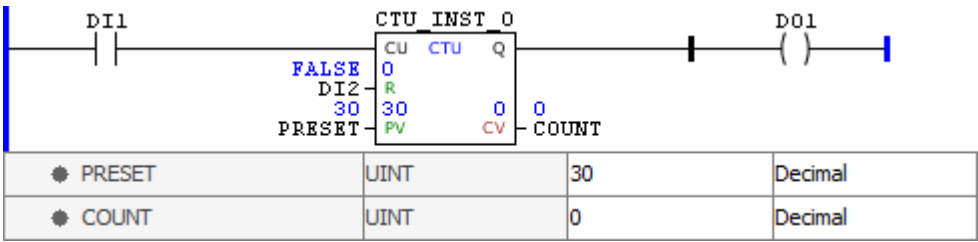
Funcionamento

Este bloco, quando identifica uma borda de subida em CD, incrementa a variável CV até que a mesma seja igual a PV. Enquanto CV for igual a PV, a saída Q permanece em nível TRUE. Ao detectar nível alto em R, o bloco carrega o valor zero em CV.

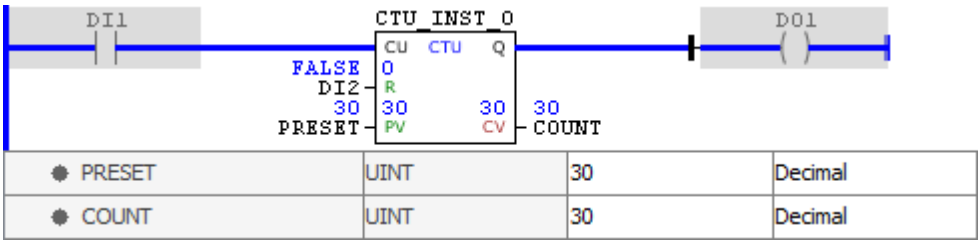
Fluxograma do Bloco



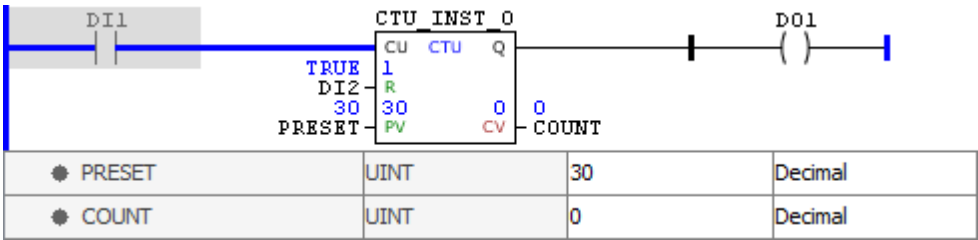
Exemplo



O exemplo acima exibe as condições iniciais da rotina. Como CV possui um valor inferior ao de PV, a saída Q é desabilitada.



A cada borda de subida identificada em CU, o valor de CV é incrementado até chegar ao valor de PV, quando a saída Q é habilitada.

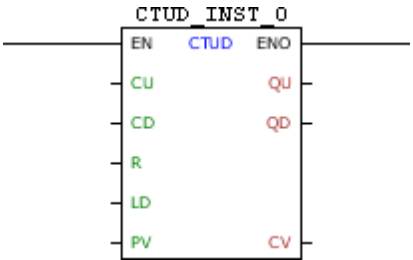


Ao identificar o nível TRUE em R, o bloco carrega o valor zero para CV. Como este valor é inferior ao de PV, a saída Q é desabilitada.

10.3.5.1.15.3 CTUD

Bloco de contagem progressiva e regressiva de pulsos de entrada.

Representação Ladder



Estrutura do Bloco

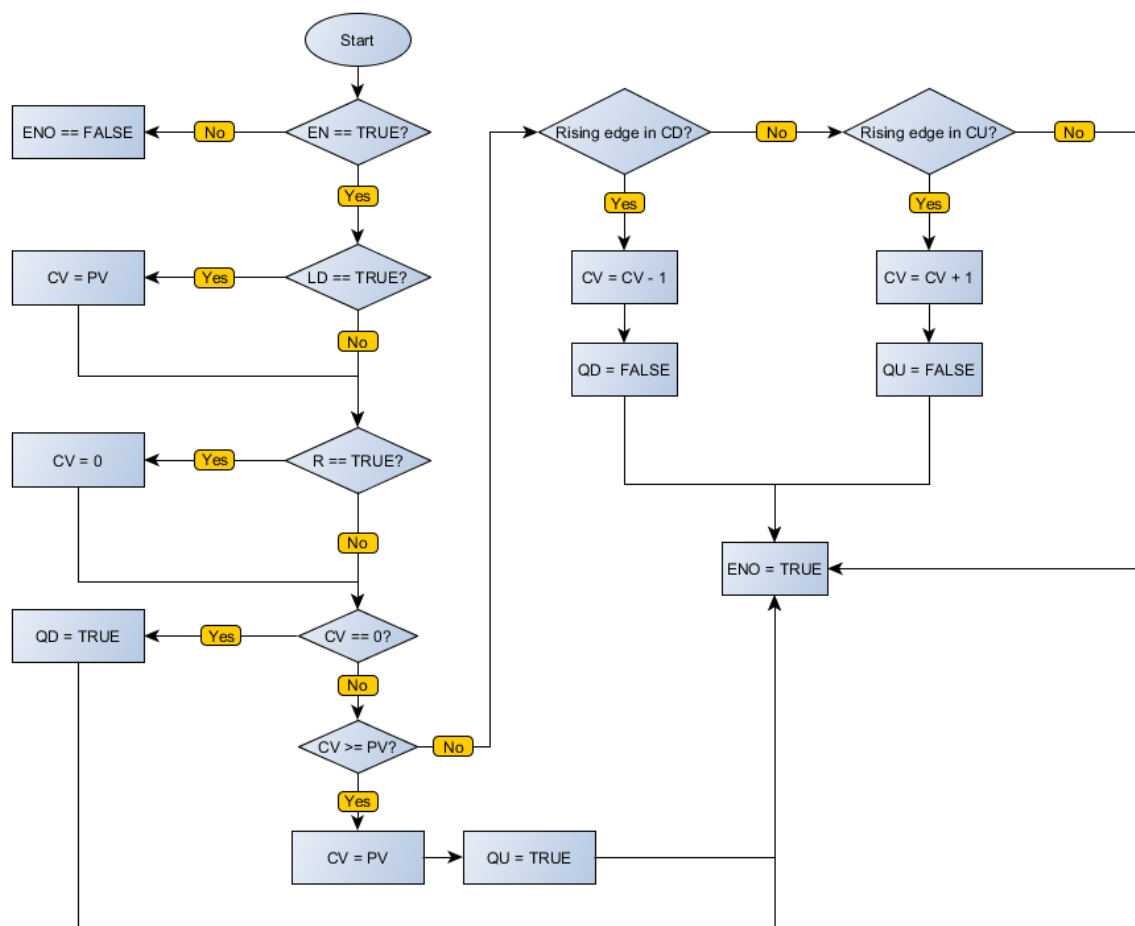
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	CU	BOOL	Identificador de pulsos para incrementador
	CD	BOOL	Identificador de pulsos para decrementador
	R	BOOL	Carrega o valor zero em CV
	LD	BOOL	Carrega o valor de PV em CV
	PV	WORD UINT	Valor de referência
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	QU	BOOL	Sinalizador de contador estourado
	QD	BOOL	Sinalizador de contador zerado
	CV	WORD UINT	Valor de contagem atual
VAR	CTUD_INST_0	CTUD	Instância de acesso à estrutura do bloco

Funcionamento

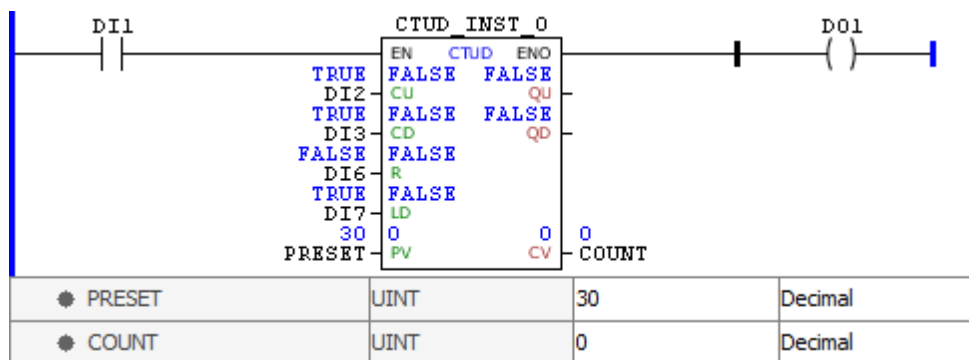
Este bloco, quando tem um valor TRUE em EN, atua como um bloco CTD e um bloco CTU ao mesmo tempo, atuando sobre o mesmo contador CV. Ou seja: incrementa CV até chegar a PV para bordas de subida em CU e decrementa CV até chegar a zero para bordas de subida em CD. Uma transição positiva em R carrega zero em CV, enquanto uma borda de subida em LD carrega o valor de PV em CV. Caso CV tenha valor zero, QD recebe TRUE, e caso CV tenha valor igual a PV, QU recebe TRUE.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN.

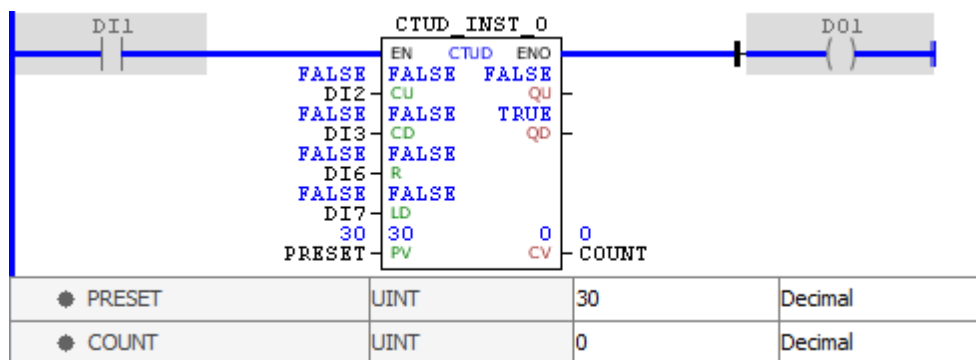
Fluxograma do Bloco



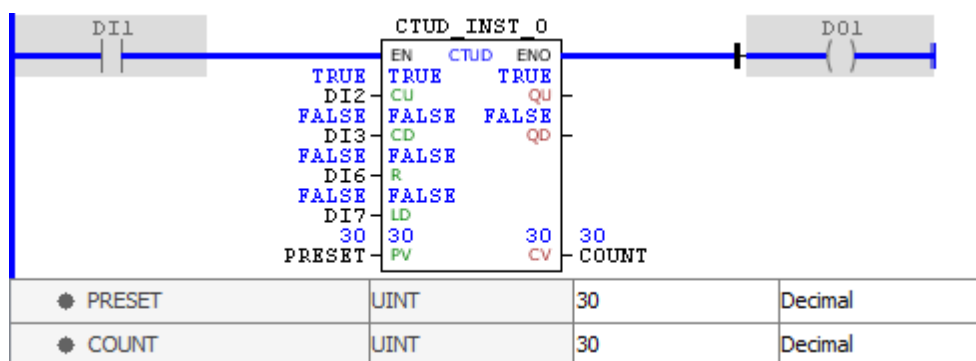
Exemplo



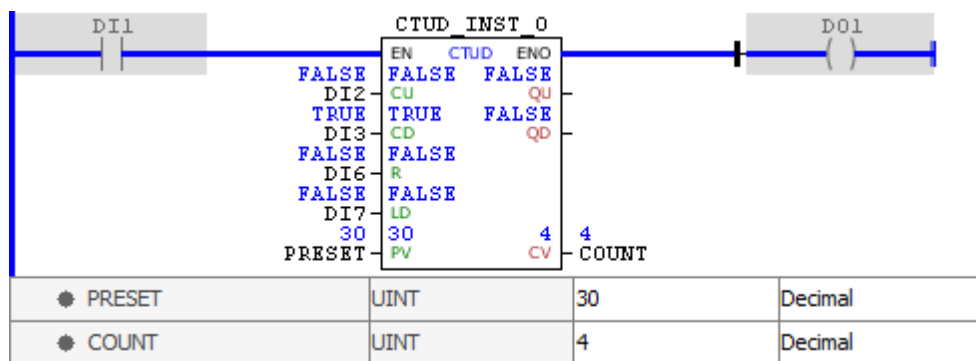
O exemplo acima exibe o bloco desabilitado, com suas variáveis internas todas zeradas. Ainda que os controles externos estejam ativados, esses valores não são repassados para a instância do bloco.



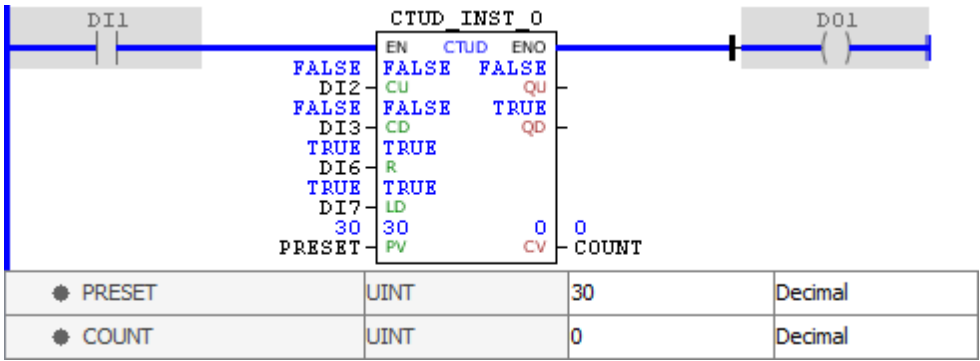
Ao ser ativado, o bloco identifica o valor de PRESET, carregando-o em PV, e identifica que a saída está com o valor zero, habilitando a saída QD. Finalizada a execução, a saída ENO é ativada.



A cada borda de subida identificada em CU, o valor de CV é incrementado até chegar ao valor de PV, quando a saída QU é habilitada. Finalizada a execução, a saída ENO é ativada.



A cada borda de subida identificada em CD, o valor de CV é decrementado. Sendo CV de um valor intermediário a zero e PV, ambas as saídas QD e QU são desativadas. Finalizada a execução, a saída ENO é ativada.



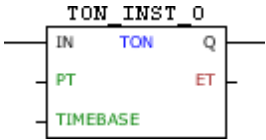
Um valor TRUE em R reseta CV, enquanto um valor TRUE em LD carrega o valor de PV para CV. Como podemos notar, R prevalece sobre LD, restando CV e habilitando a saída QD. Finalizada a execução, a saída ENO é ativada.

10.3.5.1.16 Timer

10.3.5.1.16.1 TON

Bloco de temporizador que, ao ser energizado, habilita a saída após um atraso definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Atraso de acionamento da saída
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TON_INST_0	TON	Instância de acesso à estrutura do bloco

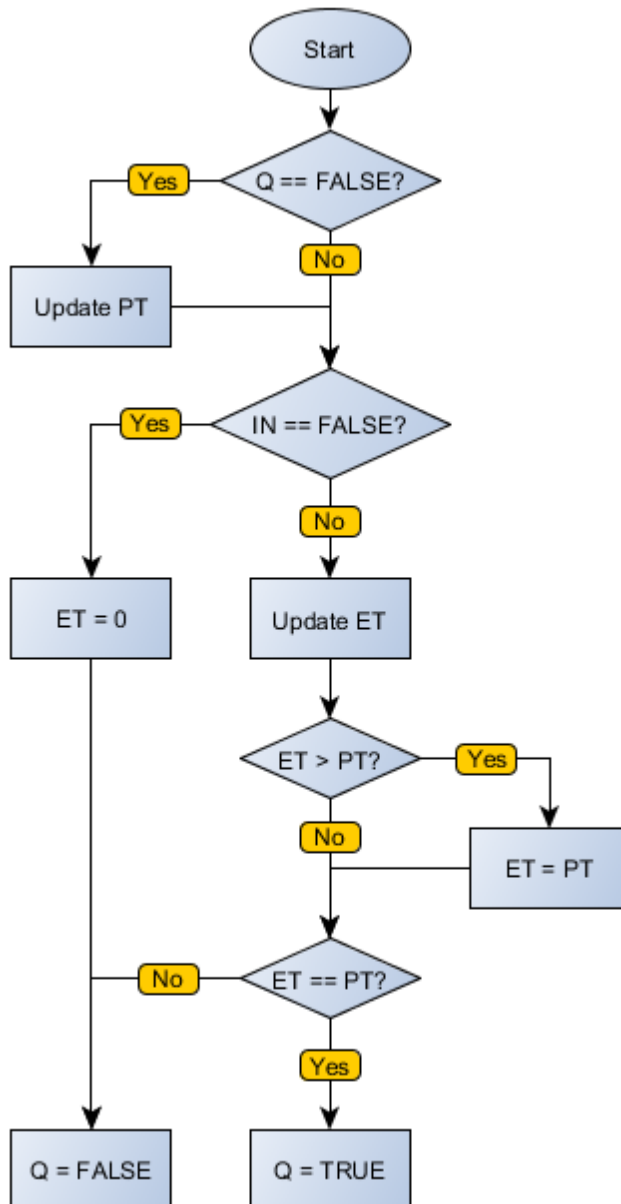
Funcionamento

Enquanto a entrada IN for FALSE, a saída Q também é FALSE e ET recebe o valor zero. Na transição positiva de borda em IN, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado TRUE, até que IN volte a ser FALSE.

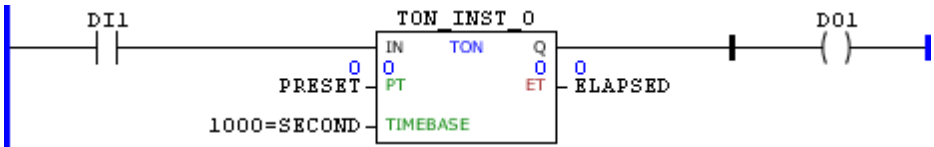
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

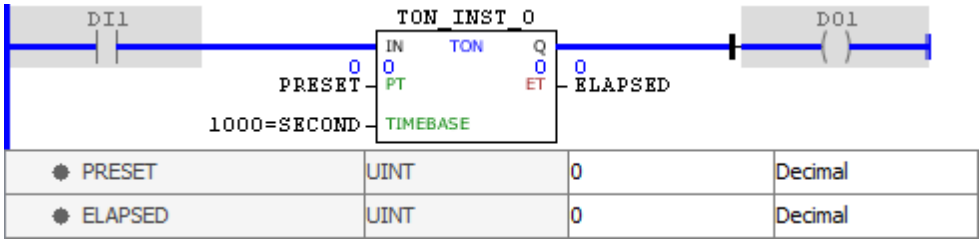
Fluxograma do Bloco



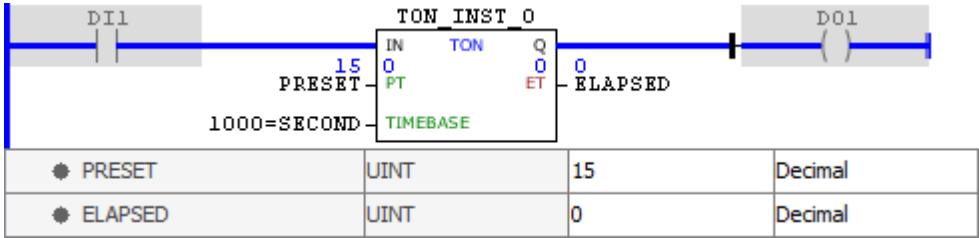
Exemplo



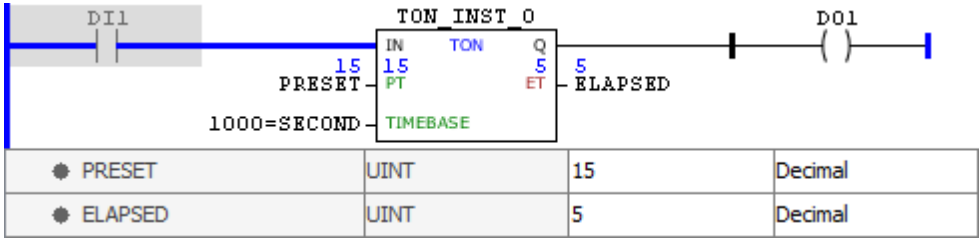
O exemplo acima exibe as condições iniciais do bloco e das variáveis da rotina.



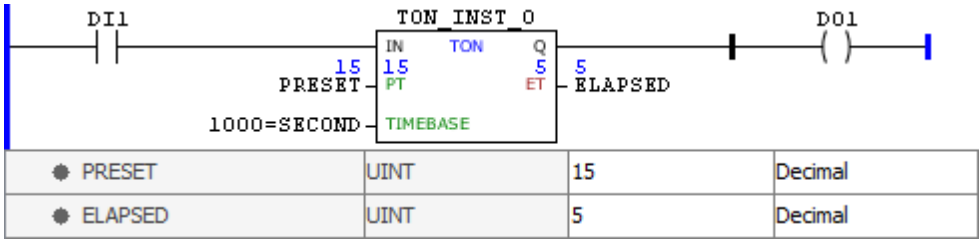
Ao ser ativada a entrada IN, a contagem dispara. Como ET é igual a PT, a saída Q é habilitada.



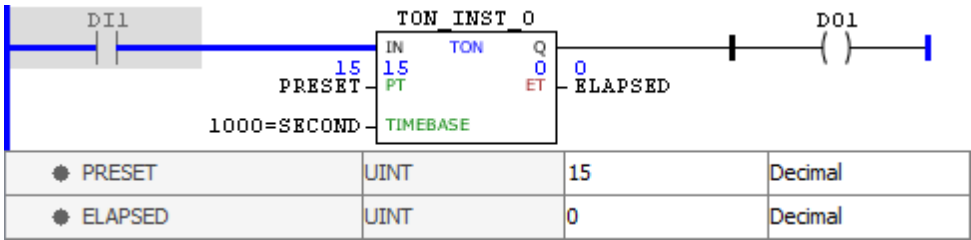
Observe que uma alteração na variável PRESET não é repassada ao campo PT enquanto a entrada IN permanece habilitada.



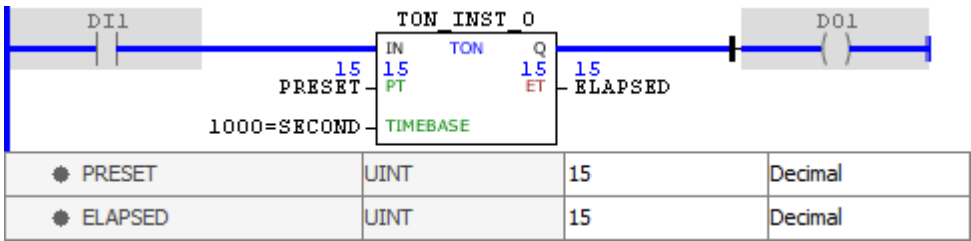
Desabilitando a entrada IN, o valor de PT é atualizado e a saída Q é desabilitada. Ao ativá-la novamente, a contagem dispara.



Desabilitando a entrada IN, o valor de ET permanece salvo.



Habilitando a entrada IN, o valor de ET é resetado e a contagem dispara.



Quando ET atinge o valor de PT, a saída Q é habilitada e permanece assim enquanto IN estiver em nível TRUE.

10.3.5.1.16.2 TOF

Bloco de temporizador que, ao ser desenergizado, desabilita a saída após um atraso definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Atraso de desacionamento da saída
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TOF_INST_0	TOF	Instância de acesso à estrutura do bloco

Funcionamento

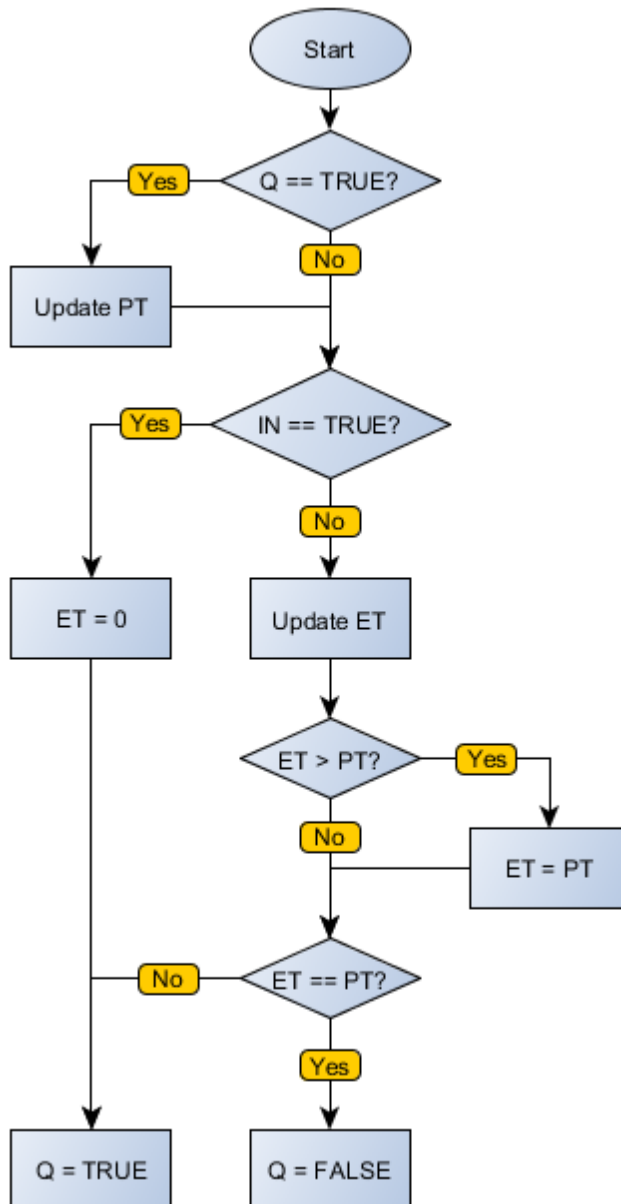
Enquanto a entrada IN for TRUE, a saída Q também é TRUE e ET recebe o valor zero. Na transição negativa de borda em IN, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado FALSE, até que IN volte a ser

TRUE.

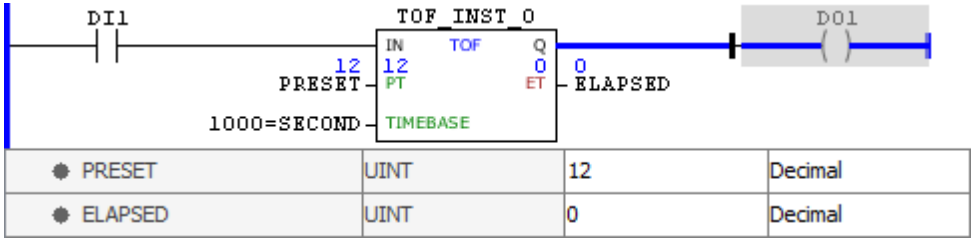
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo

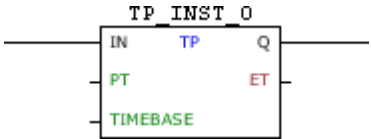


O exemplo acima desabilita a saída DO1 ao identificar um nível baixo em DI1 por 12 segundos, permanecendo desabilitada até que DI1 volte a ser TRUE.

10.3.5.1.16.3 TP

Bloco de temporizador que, ao identificar ser energizado, habilita a saída por um tempo definido por PT.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	IN	BOOL	Habilitação do bloco
	PT	WORD UINT DWORD UDINT	Tempo em que a saída fica habilitada
	TIMEBASE	WORD	Base de tempo para PT e ET
VAR_OUTPUT	Q	BOOL	Saída do bloco
	ET	WORD UINT DWORD UDINT	Tempo decorrido do contador
VAR	TP_INST_0	TP	Instância de acesso à estrutura do bloco

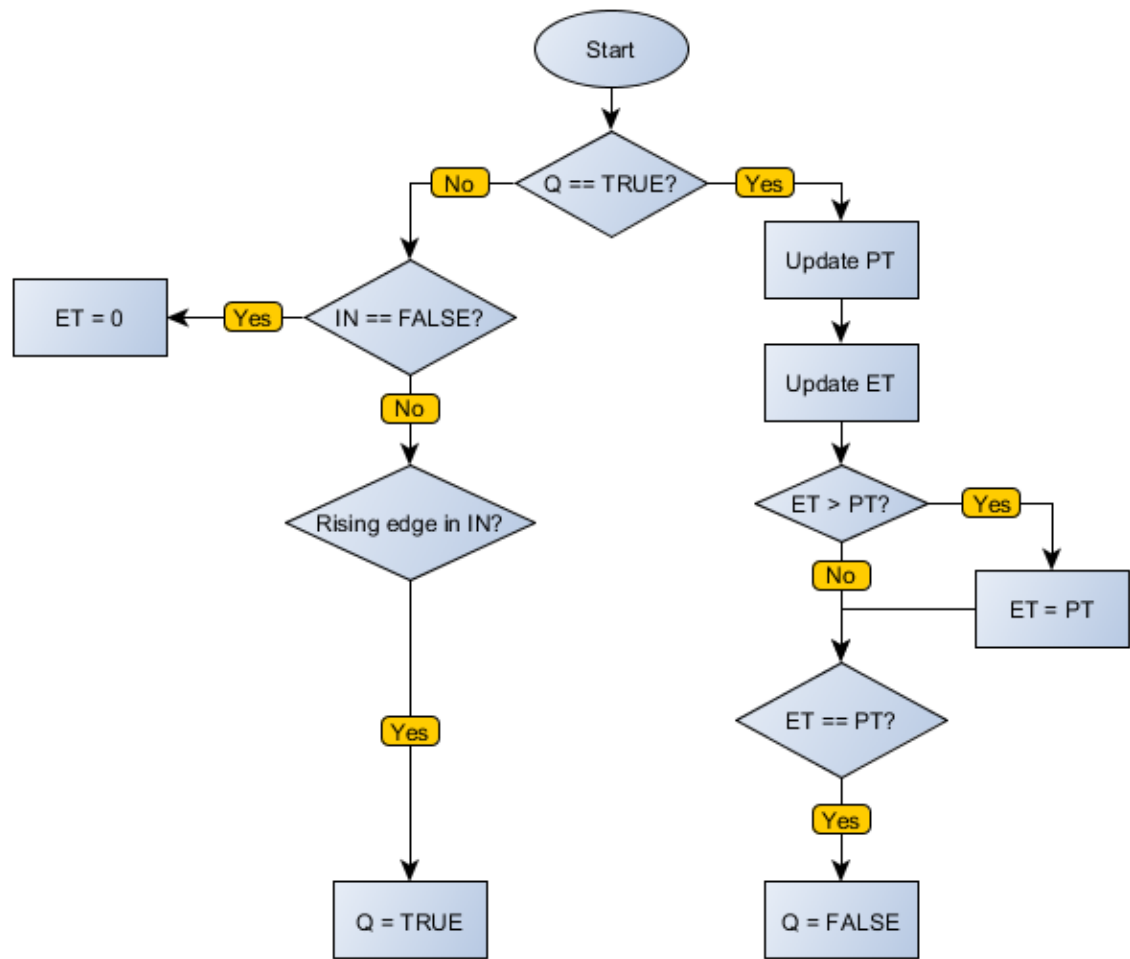
Funcionamento

Na transição positiva de borda em IN, Q recebe valor TRUE, a contagem dispara e ET é incrementado de acordo com TIMEBASE. Quando ET for igual a PT, a saída Q vai para estado FALSE, até que IN volte a ser TRUE. Nesse momento, se IN estiver em nível TRUE, nada acontece. Se IN estiver em nível FALSE, ET é resetado automaticamente.

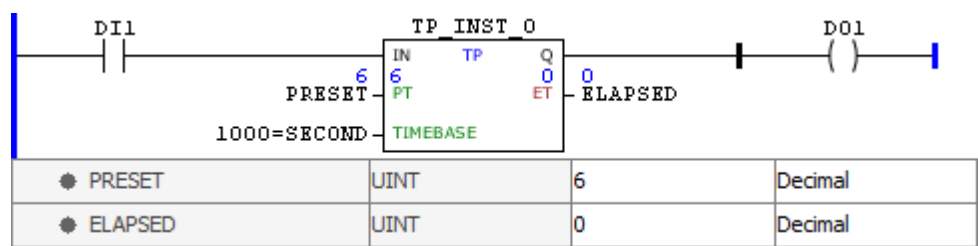
Compatibilidade

Equipamento	Versão
PLC300	1.50 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco



Exemplo



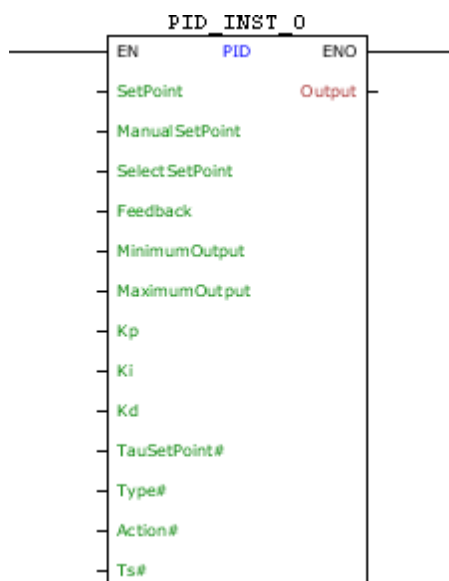
O exemplo acima habilita a saída DO1 por seis segundos a cada transição positiva de DI1.

10.3.5.1.17 Control

10.3.5.1.17.1 PID

Bloco que executa a função de um controlador PID discreto. A partir das variáveis de entrada, ele calcula a saída correspondente do controlador.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SetPoint	REAL	Referência automática (pré-controle)
	ManualSetPoint	REAL	Referência forçada (pós-controle)
	SelectSetPoint	BOOL	Seleciona qual referência utilizar
	Feedback	REAL	Variável de realimentação da malha
	MinimumOutput	REAL	Valor mínimo da saída do controlador
	MaximumOutput	REAL	Valor máximo da saída do controlador
	Kp	REAL	Ganho proporcional
	Ki	REAL	Ganho integral
	Kd	REAL	Ganho derivativo
	TauSetPoint#	REAL	Constante de tempo do filtro de entrada de referência automática
	Type#	BYTE	Tipo de controlador
	Action#	BYTE	Ação de controle
	Ts#	UINT	Período de amostragem [ms]
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Output	REAL	Saída do controlador
VAR	PID_INST_0	PID	Instância de acesso à estrutura do bloco

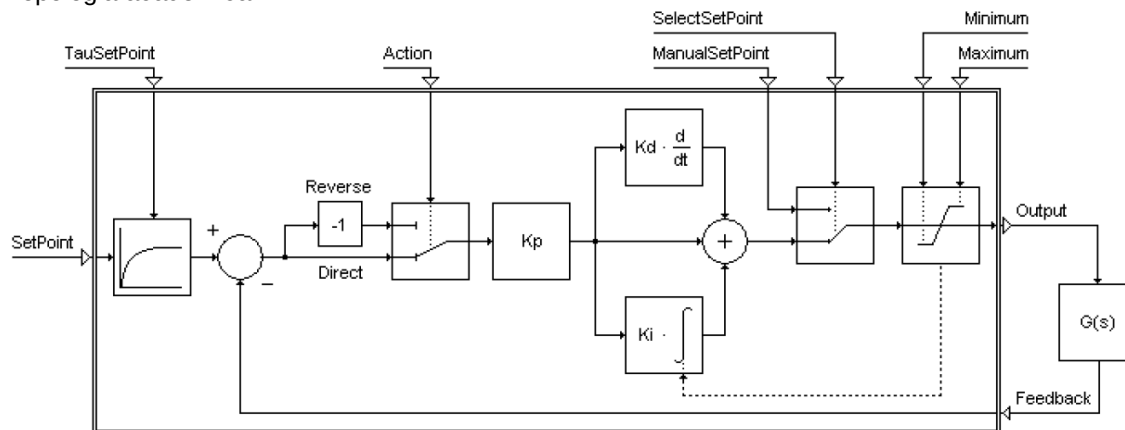
Funcionamento

Na transição positiva de borda em EN, Output recebe valor zero, e o bloco executa sua

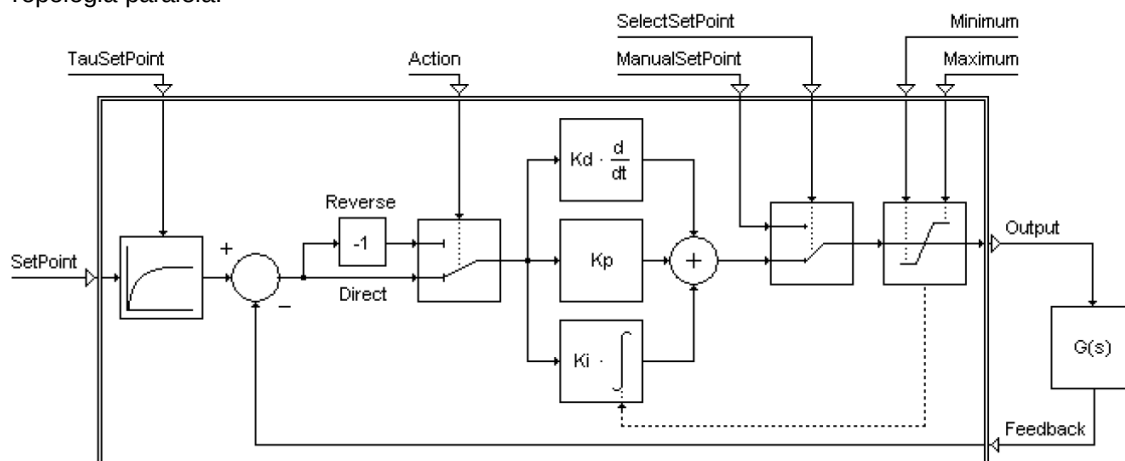
funcionalidade enquanto EN estiver em nível TRUE.

Quando habilitado, este bloco executa uma rotina de controle PID com os parâmetros K_p , K_i e K_d escolhidos. A topologia PID utilizada poderá ser a Acadêmica ou a Paralela, dependendo do que for escolhido em Type#.

Topologia acadêmica:



Topologia paralela:



Os níveis do sinal da saída do controlador são saturados pelo valor de MinimumOutput e MaximumOutput. A entrada SelectSetPoint em nível FALSE faz com que a referência adotada seja SetPoint, permitindo que o controlador exerça domínio sobre o processo. Quando SelectSetPoint vai para nível TRUE, o controlador não exerce mais domínio, sendo que ManualSetPoint passa a ser considerado o sinal de saída do controlador.

Action# definirá a operação de realimentação. Se Action# for DIRECT, a operação será SetPoint – Feedback. Se Action# for REVERSE, a operação será Feedback – SetPoint.

Feedback recebe a variável de processo considerada como saída da planta. Ts# recebe o período de amostragem para o controlador e TauSetPoint# recebe a constante de tempo para o filtro de entrada da referência automática.

Quando EN possui valor FALSE, Output permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

**NOTA!**

Efeito da alteração dos ganhos sobre o processo:

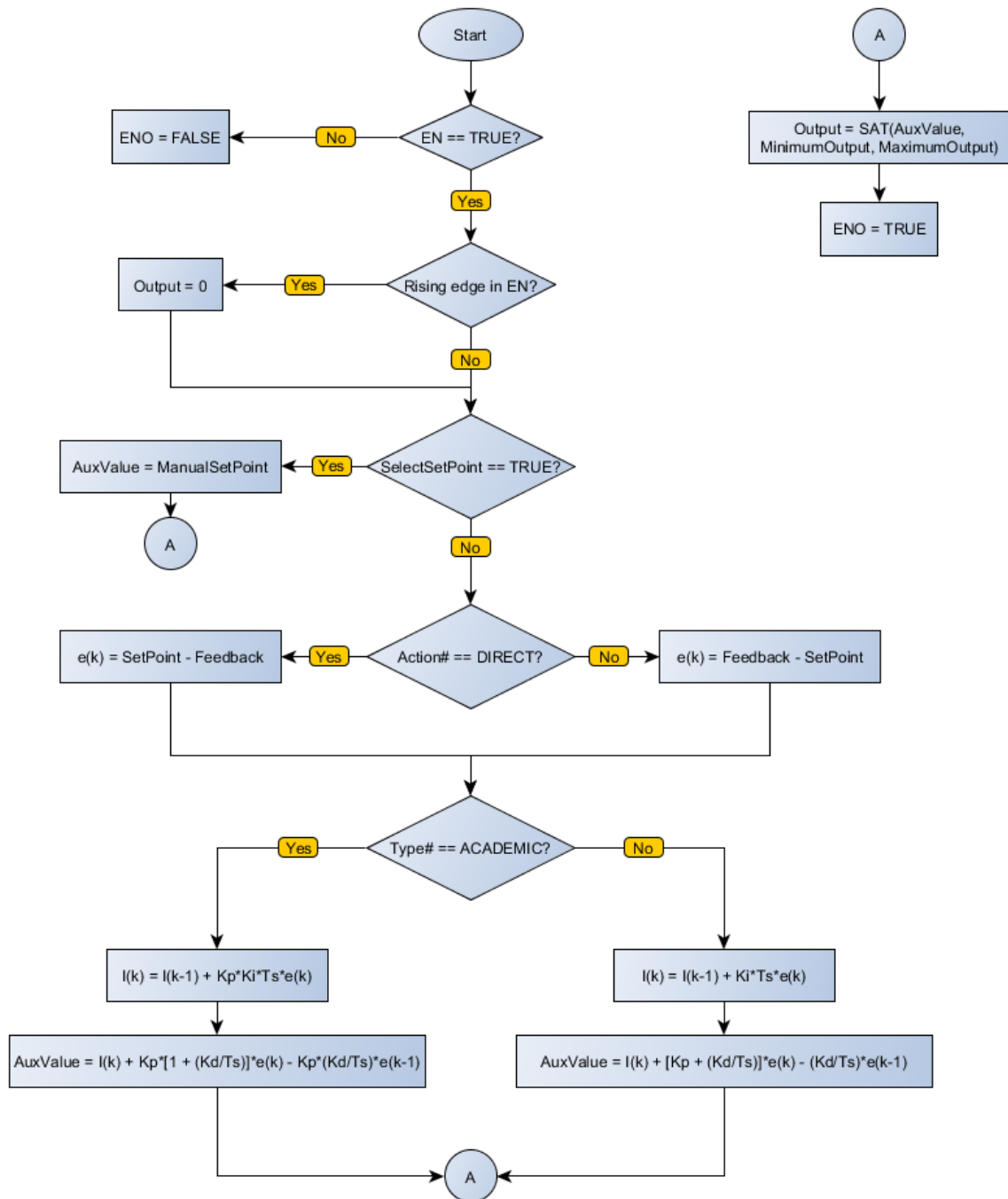
- Se diminuir K_p , o processo torna-se mais lento; geralmente mais estável ou menos oscilante; tem menos sobre-sinal (overshoot).
- Se aumentar K_p , o processo responde mais rápido; pode ficar instável ou mais oscilante; tem mais sobre-sinal (overshoot).
- Se diminuir K_i , o processo torna-se mais lento, demorando para atingir o "SetPoint"; fica mais estável ou menos oscilante; tem menos sobre-sinal (overshoot).
- Se aumentar K_i , o processo torna-se mais rápido, atingindo rapidamente o "SetPoint"; fica mais instável ou mais oscilante; tem mais sobre-sinal (overshoot).
- Se diminuir K_d , o processo torna-se mais lento; tem menos sobre-sinal (overshoot).
- Se aumentar K_d , tem mais sobre-sinal (overshoot).

**NOTA!**

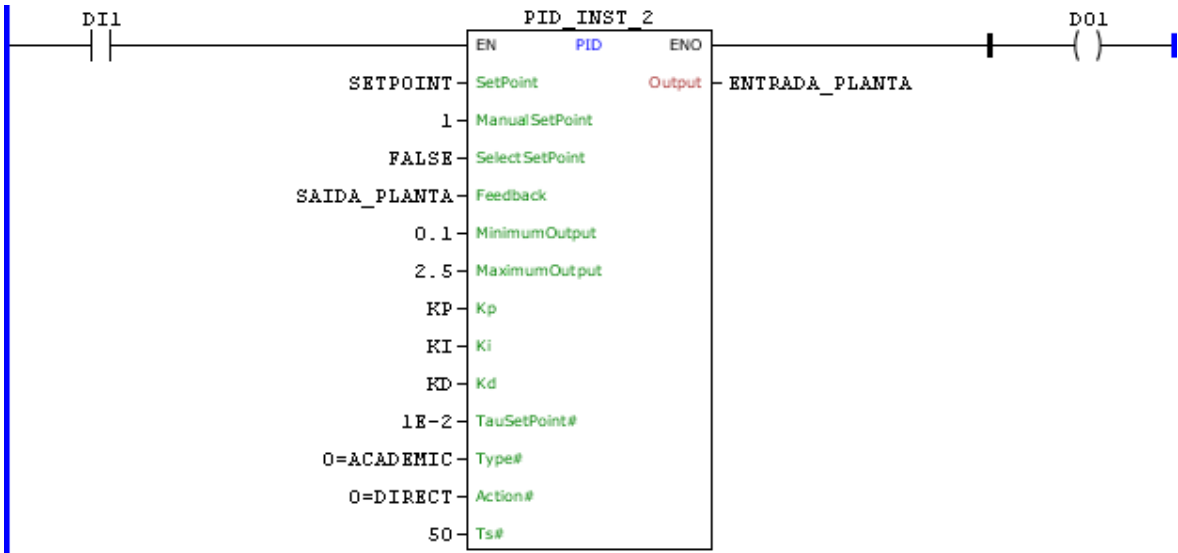
Como melhorar o desempenho do processo pelo ajuste dos ganhos (válido para o PID Acadêmico)

- Se o desempenho do processo está quase bom, mas o sobre-sinal (overshoot) está um pouco alto, tente: (1) diminuir o K_p em 20%, (2) diminuir K_i em 20% e/ou (3) diminuir K_d em 50%.
- Se o desempenho do processo está quase bom, mas não tem o sobre-sinal (overshoot) e demora para atingir o "SetPoint", tente: (1) aumentar K_p em 20%, (2) aumentar K_i em 20%, (3) aumentar K_d em 50%.
- Se o desempenho do processo está bom, mas a saída do processo está variando demais, tente: (1) aumentar K_d em 50%, (2) diminuir K_p em 20%.
- Se o desempenho do processo está ruim, ou seja, após a partida, o transitório dura vários períodos de oscilação, que reduz muito lentamente ou não reduz, tente: (1) diminuir K_p em 50%.
- Se o desempenho do processo está ruim, ou seja, após a partida avança lentamente em direção ao "SetPoint", sem sobre-sinal (overshoot), mas ainda está muito longe e a saída do processo é menor que o valor nominal, tente: (1) aumentar K_p em 50%, (2) aumentar K_i em 50%, (3) aumentar K_d em 70%.

Fluxograma do Bloco



Exemplo



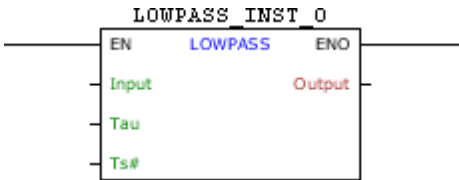
O exemplo acima cria uma malha de um PID acadêmico digital, com tempo de amostragem de 50 ms, utilizando as constantes KP, KI e KD para o controle. É utilizada a referência automática SETPOINT, filtrada por um filtro de primeira ordem com constante de tempo 0.01. O sinal de erro é calculado como a diferença entre a referência filtrada e a variável SAIDA_PLANTA. A saída do controlador é saturada entre os valores 0.1 e 2.5 e enviada para a variável ENTRADA_PLANTA.

10.3.5.1.18 Filter

10.3.5.1.18.1 LOWPASS

Bloco que filtra a entrada utilizando um filtro passa baixas de primeira ordem e insere o resultado na saída.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Input	REAL	Sinal de entrada
	Tau	REAL	Constante de tempo do filtro
	Ts#	UINT	Período de amostragem [ms]
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Output	REAL	Saída do filtro
VAR	LOWPASS_INST_0	LOWPASS	Instância de acesso à estrutura do bloco

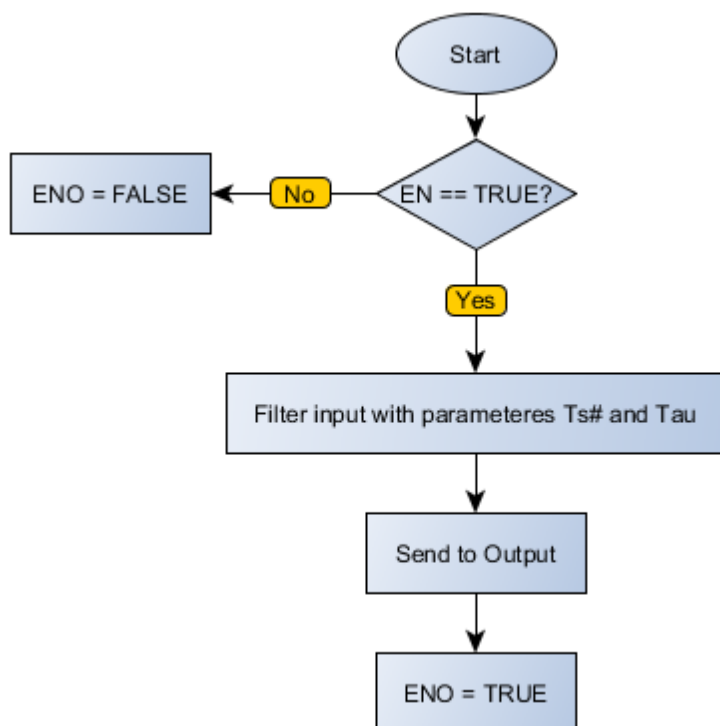
Funcionamento

Este bloco, quando tem um valor TRUE em EN, filtra o valor de entrada Input utilizando um filtro passa baixas de primeira ordem descrito por Tau e Ts#, inserindo o resultado em Output. Na borda de subida de EN, Output recebe zero.

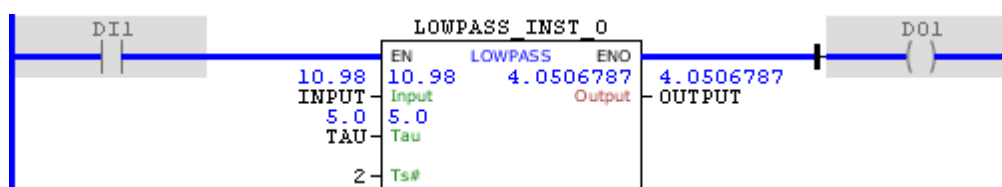
Quando EN possui valor FALSE, Output permanece inalterado.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

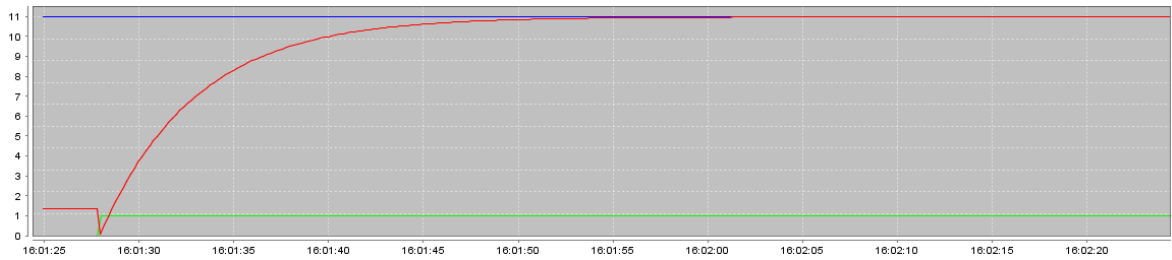
Fluxograma do Bloco



Exemplo



O exemplo acima faz com que a saída OUTPUT, ao identificar uma borda de subida em EN, exiba um comportamento de primeira ordem, com constante de tempo igual a TAU e tempo de amostragem de 2 ms, visando a atingir a referência setada em INPUT. A cada cálculo finalizado com sucesso, a saída ENO é ativada.

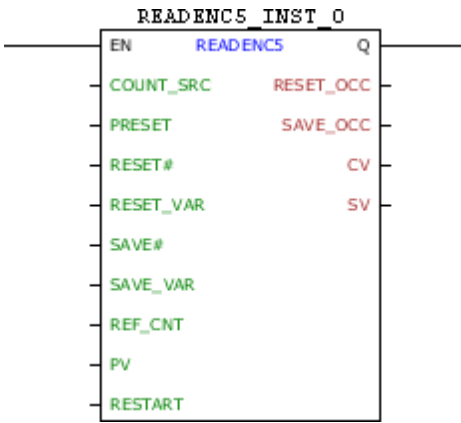


10.3.5.1.19 Hardware

10.3.5.1.19.1 READENC5

Bloco que executa contagem dos pulsos de encoder.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	COUNT_SRC	BYTE	Determina qual o encoder será utilizado na contagem dos pulsos
	PRESET	BOOL	Atribui o valor de PV a CV
	RESET#	BYTE	Escolhe o controle do reset de CV
	RESET_VAR	BOOL	Caso a escolha de RESET# seja por variável, realiza o reset de CV quando em valor TRUE
	SAVE#	BYTE	Escolhe o controle do salvamento do contador
	SAVE_VAR	BOOL	Caso a escolha de SAVE# seja por variável, realiza o salvamento de CV em SV quando em valor TRUE
	REF_CNT	LREAL	Valor de pulso de referência para habilitação de saída
	PV	LREAL	Valor de configuração inicial
	RESTART	LREAL	Valor de referência para reset automático de CV
VAR_OUTPUT	Q	BOOL	Habilitação da saída
	RESET_OCC	BOOL	Sinalizador de reset
	SAVE_OCC	BOOL	Sinalizador de salvamento
	CV	LREAL	Valor do contador de pulsos
	SV	LREAL	Valor do contador salvo pela última vez
VAR	READENC5_INST_0	READENC5	Instância de acesso à estrutura do bloco

Funcionamento

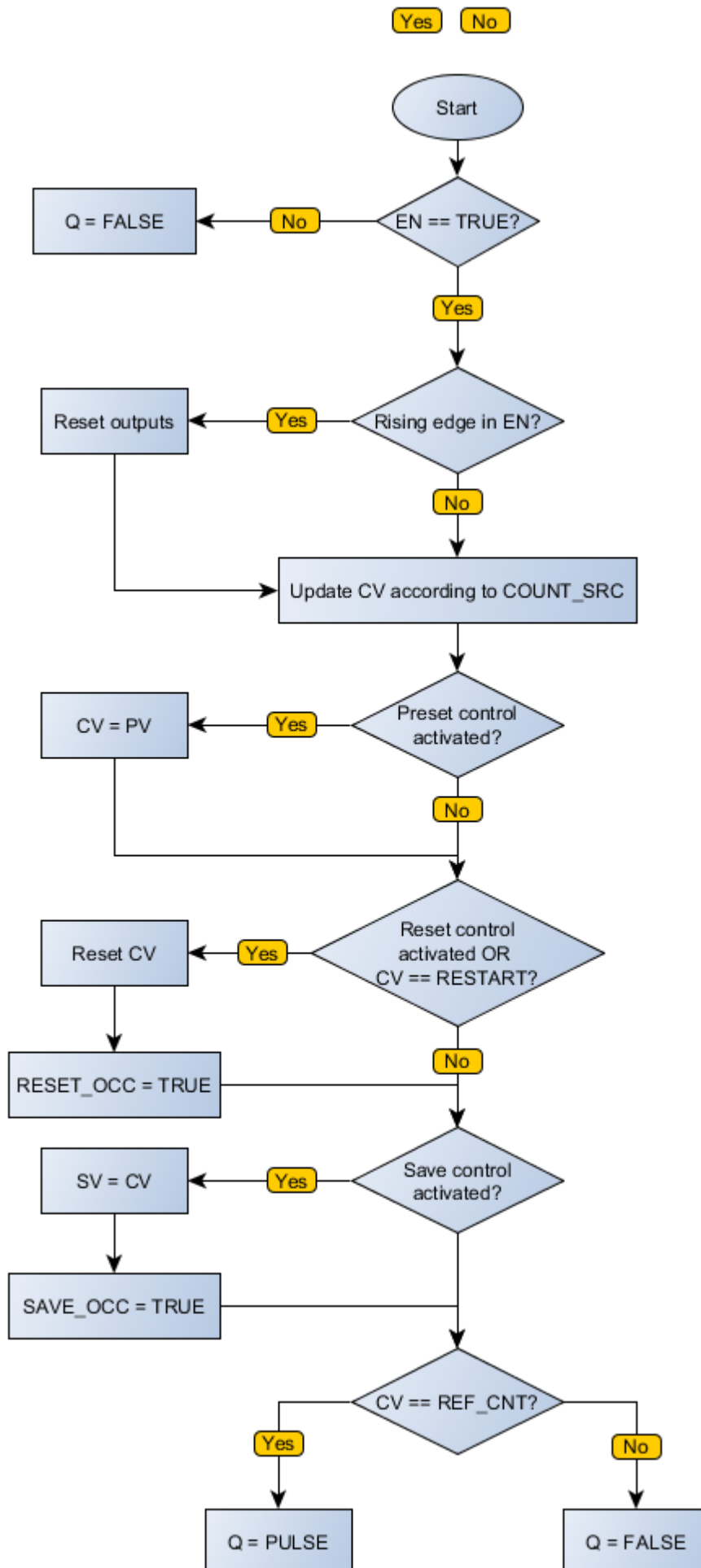
Este bloco, quando identifica uma borda de subida em EN, zera suas saídas e realiza a contagem dos pulsos do encoder especificado em COUNT_SRC enquanto habilitado. Esse valor de contagem é armazenado em CV.

O controle especificado em RESET# zera o contador quando for ativado, enquanto o controle especificado em SAVE#, quando ativado, armazena o valor de CV em SV. Cada um desses controles seta seu respectivo sinalizador (RESET_OCC ou SAVE_OCC) por um ciclo de scan.

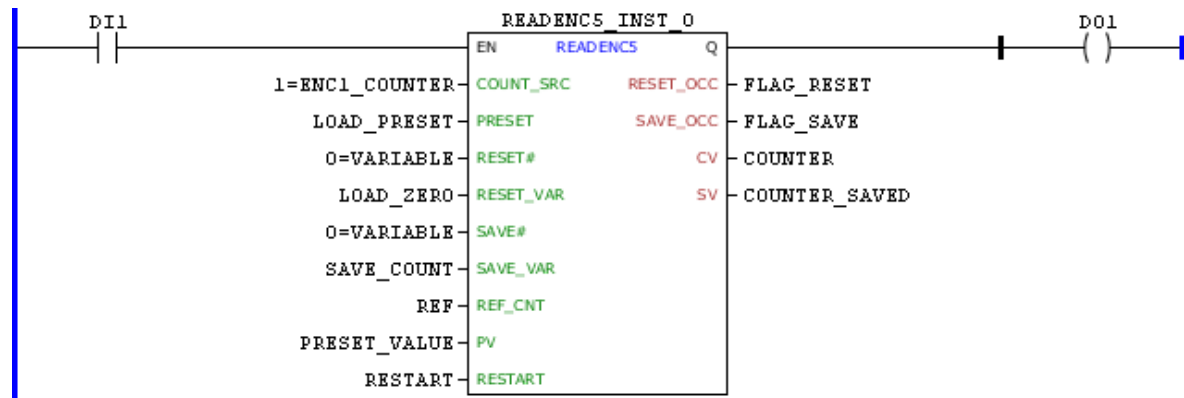
O bloco também permite configurar um reset automático, quando CV atinge o valor de RESTART.

O valor de Q é ativado por um ciclo de scan quando CV atinge o valor de REF_CNT.

Fluxograma do Bloco



Exemplo



O exemplo acima realiza a leitura do encoder 1, armazenando o valor em COUNTER, com referência de contagem REF e controles de RESET e SAVE realizados pelas variáveis LOAD_ZERO e SAVE_COUNT, respectivamente. Quando COUNTER for igual a REF, a saída Q é ativada por um ciclo de scan.

10.3.5.1.19.2 IMMEDIATE_INPUT

Bloco que realiza uma leitura instantânea do valor da entrada selecionada, sem alterar o valor das imagens (variáveis GLOBAL_IO).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source#	BYTE	Entradas a serem lidas (digitais ou analógicas)
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Destination	WORD INT UINT	Variável mapeada com os valores das entradas selecionadas

Funcionamento

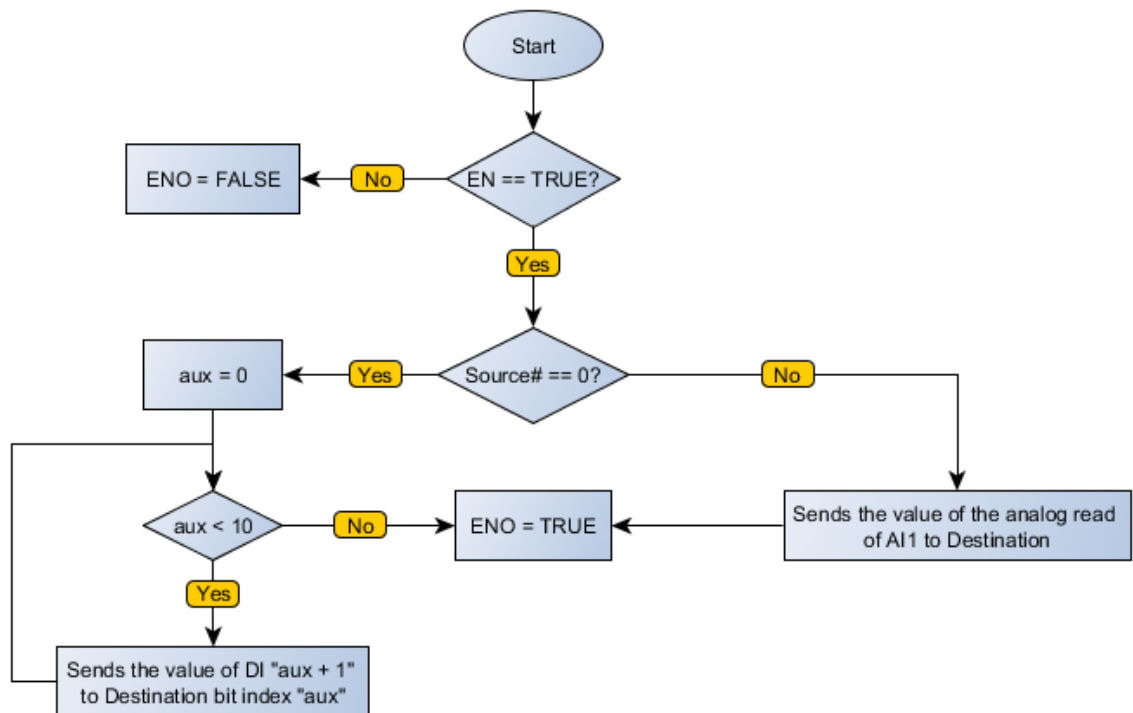
Este bloco, quando tem um valor TRUE em EN, obtém o valor imediato da entrada selecionada em Source#. Caso seja selecionada a entrada analógica AI1, seu valor é repassado a Destination. Caso sejam selecionadas as entradas digitais, seus bits são concatenados de forma a DI1 ser o bit menos significativo e DI10 ser o bit mais significativo, sendo o resultado enviado a Destination.

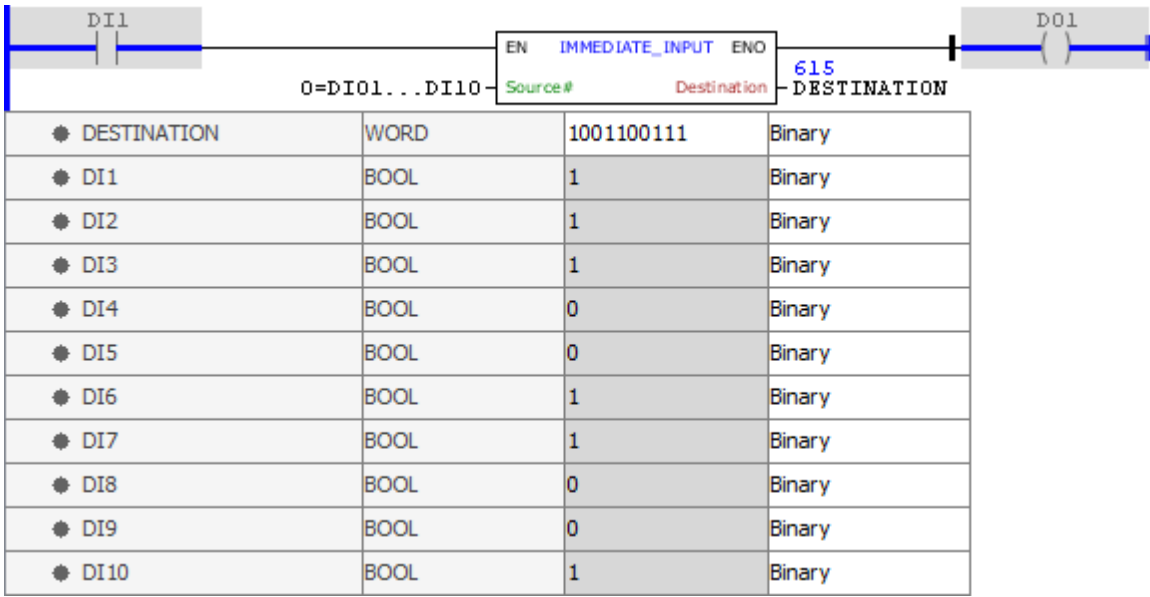
Quando EN possui valor FALSE, Destination permanece inalterada.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

Fluxograma do Bloco**Exemplo**



O exemplo acima faz uma leitura imediata dos sinais das entradas digitais DI1 a DI10 do PLC300. Essa leitura é então interpretada como uma sequência de binários, com DI1 sendo o bit menos significativo, e o resultado é enviado à variável DESTINATION. Finalizado o bloco com sucesso, a saída ENO é ativada.

10.3.5.1.19.3 IMMEDIATE_OUTPUT

Bloco que realiza uma escrita instantânea na porta da saída selecionada, sem alterar o valor das imagens (variáveis GLOBAL_IO).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	Source	WORD INT UINT	Variável mapeada com os valores a serem enviados às saídas selecionadas
VAR_OUTPUT	ENO	BOOL	Habilitação de saída
	Destination#	BYTE	Saídas a serem escritas (digitais ou analógicas)

Funcionamento

Este bloco, quando tem um valor TRUE em EN, escreve imediatamente na saída selecionada o valor de Source. Caso seja selecionada a saída analógica AO1, o valor de Source é repassado à esta. Caso sejam selecionadas as saídas digitais, DO1 receberá o bit zero de Source, DO2 o bit um, DO3 o bit dois, e assim sucessivamente.

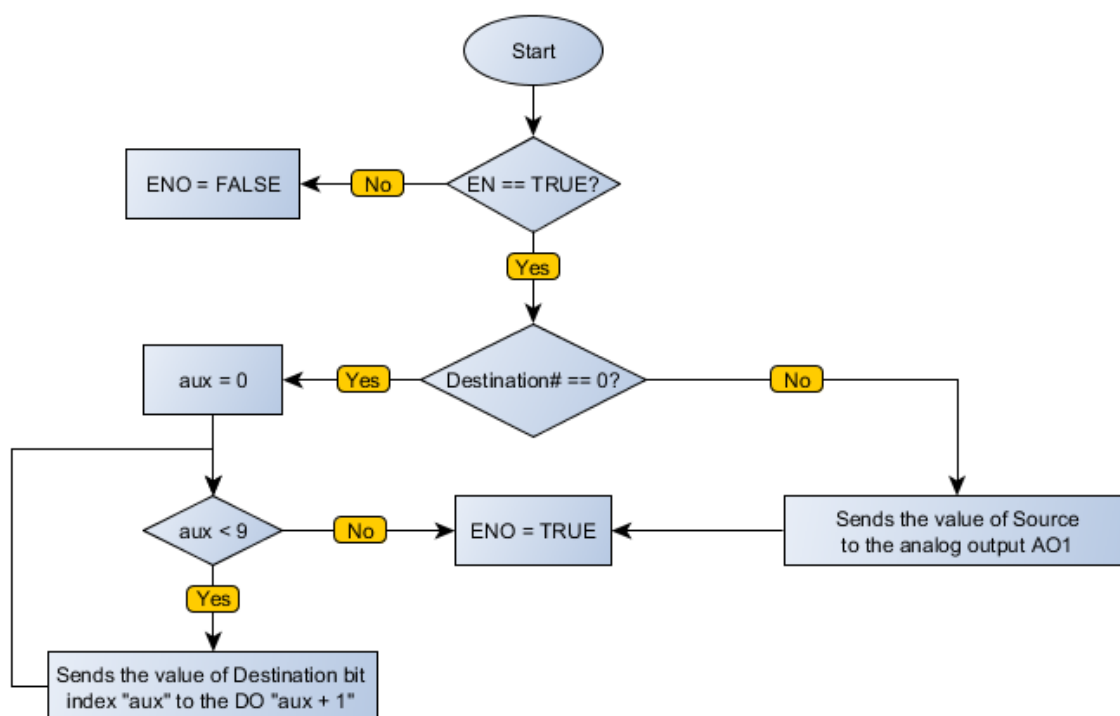
Quando EN possui valor FALSE, Destination# permanece inalterada.

O valor de ENO repassa para o próximo bloco Ladder o valor de EN após finalizada a operação.

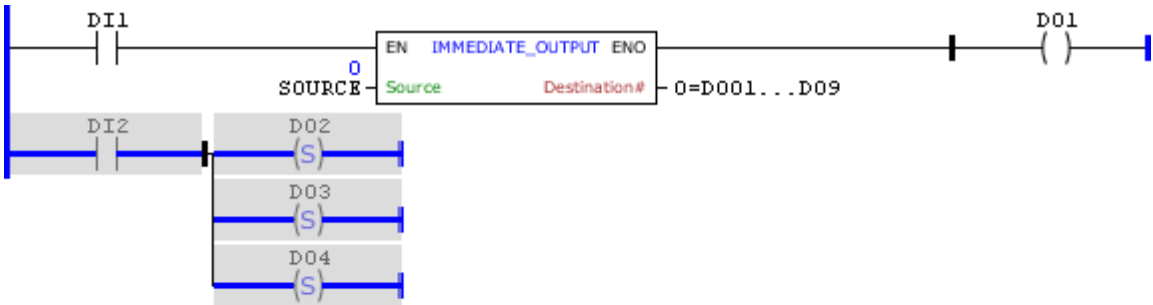
Compatibilidade

Equipamento	Versão
PLC300	1.20 ou superior
SCA06	2.00 ou superior

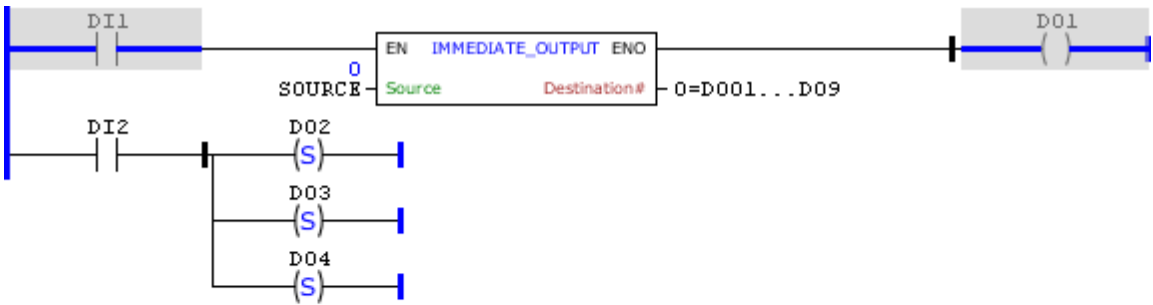
Fluxograma do Bloco



Exemplo



● SOURCE	WORD	0	Binary
● DO1	BOOL	0	Binary
● DO2	BOOL	1	Binary
● DO3	BOOL	1	Binary
● DO4	BOOL	1	Binary
● DO5	BOOL	0	Binary
● DO6	BOOL	0	Binary
● DO7	BOOL	0	Binary
● DO8	BOOL	0	Binary
● DO9	BOOL	0	Binary



● SOURCE	WORD	0	Binary
● DO1	BOOL	1	Binary
● DO2	BOOL	0	Binary
● DO3	BOOL	0	Binary
● DO4	BOOL	0	Binary
● DO5	BOOL	0	Binary
● DO6	BOOL	0	Binary
● DO7	BOOL	0	Binary
● DO8	BOOL	0	Binary
● DO9	BOOL	0	Binary

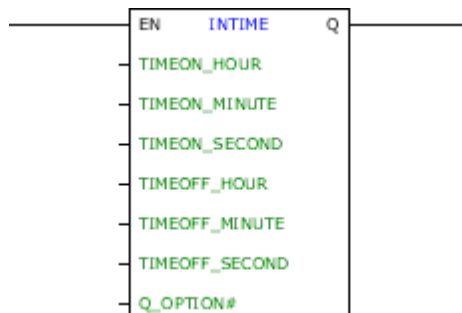
O exemplo acima faz uma escrita imediata do valor de SOURCE, interpretada como uma sequência de binários, nas saídas digitais DO1 a DO9 do PLC300, com DO1 recebendo o valor do bit menos significativo. Finalizado o bloco com sucesso, a saída ENO é ativada. Observe que a escrita imediata não prevalece sobre a bobina direta DO1 ou sobre as bobinas de habilitação em DO2, DO3 e DO4.

10.3.5.1.20 RTC

10.3.5.1.20.1 INTIME

Bloco que realiza uma habilitação programada para um horário baseado em RTC (Real Time Clock).

Representação Ladder



Estrutura do Bloco

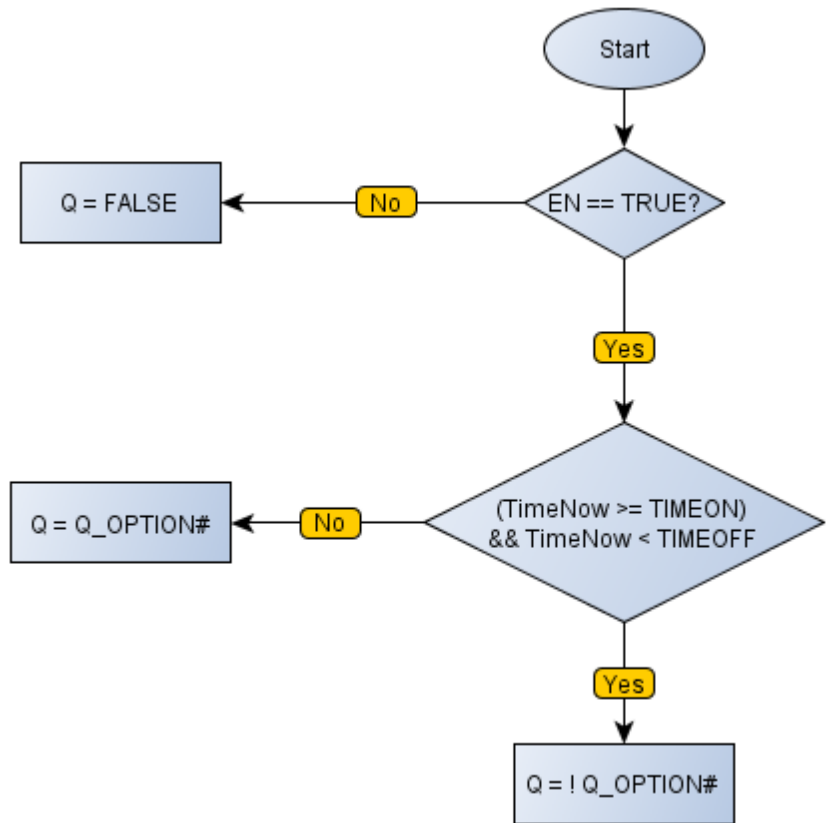
Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	TIMEON_HOUR	WORD UINT	Hora de habilitação
	TIMEON_MINUTE	WORD UINT	Minutos de habilitação
	TIMEON_SECOND	WORD UINT	Segundo de habilitação
	TIMEOFF_HOUR	WORD UINT	Hora de desabilitação
	TIMEOFF_MINUTE	WORD UINT	Minutos de desabilitação
	TIMEOFF_SECOND	WORD UINT	Segundo de desabilitação
	Q_OPTION#	BYTE	Funcionamento da saída
VAR_OUTPUT	Q	BOOL	Saída do bloco

Funcionamento

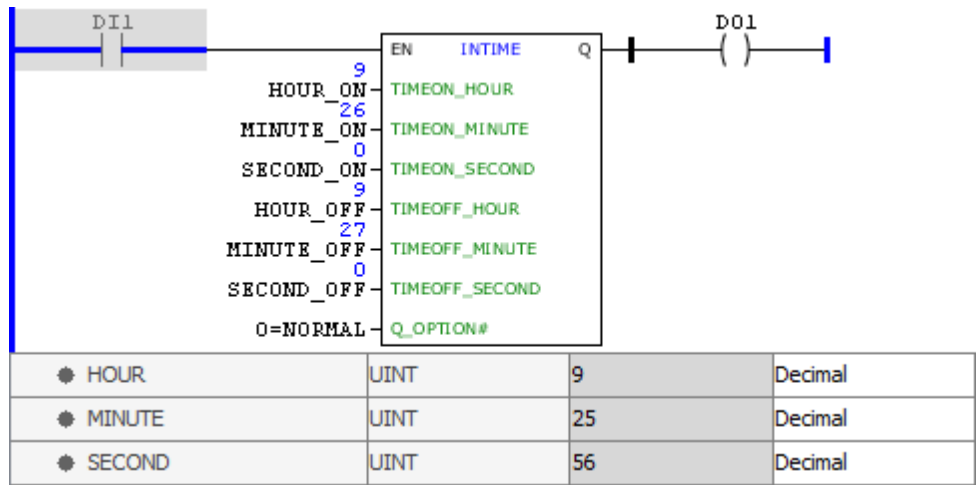
Este bloco, quando tem um valor TRUE em EN, possui dois modos de operação. Se Q_OPTION# for Normal, Q é habilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEON e desabilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEOFF. Se Q_OPTION# for Inverted, Q é desabilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEON e habilitada quando a hora do relógio interno for igual à definida pelos parâmetros TIMEOFF.

Quando EN possui valor FALSE, Q permanece FALSE.

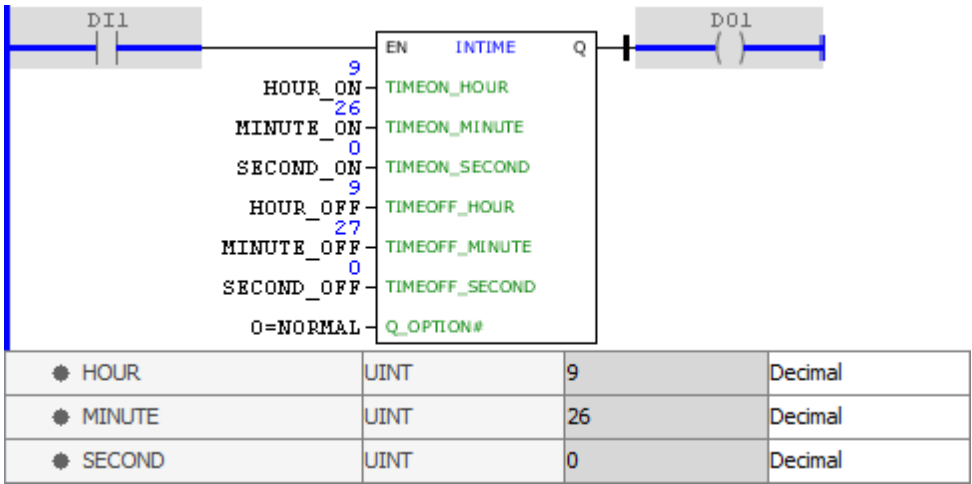
Fluxograma do Bloco



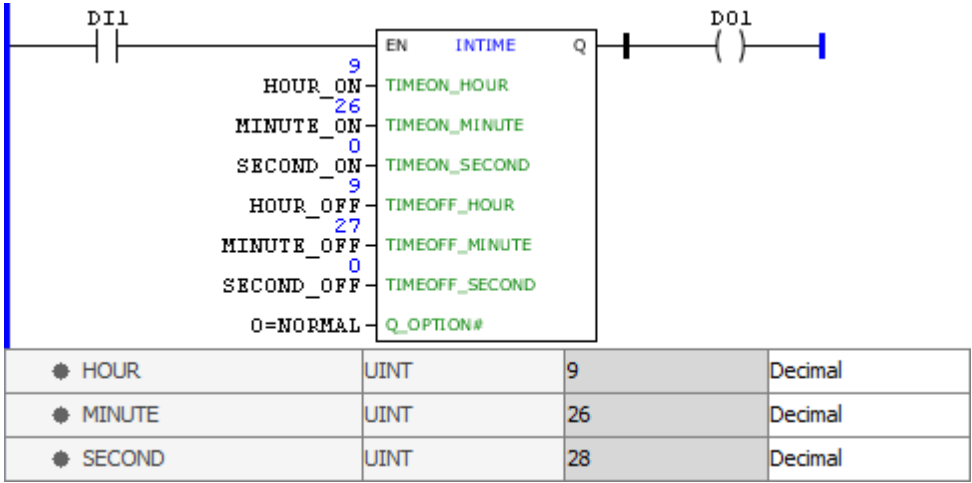
Exemplo



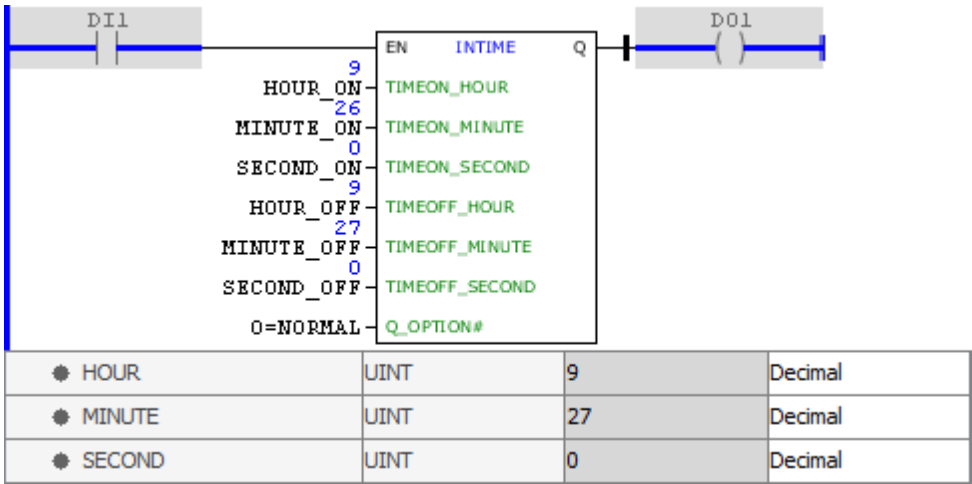
No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é inferior ao cadastrado nas entradas de habilitação do bloco (HOUR_ON, MINUTE_ON e SECOND_ON). Dessa forma, a saída Q é desabilitada.



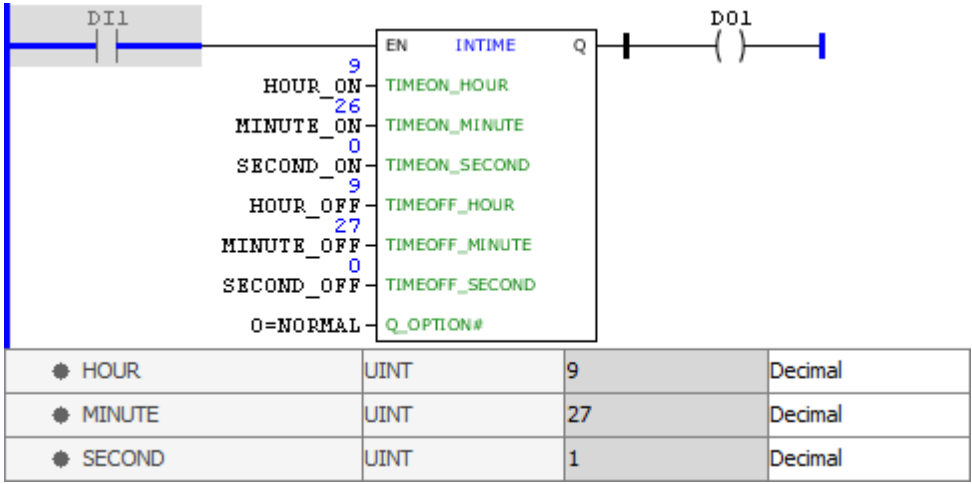
No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é igual ao cadastrado nas entradas de habilitação do bloco (HOUR_ON, MINUTE_ON e SECOND_ON). Dessa forma, a saída Q é habilitada.



No exemplo acima, o bloco INTIME está desabilitado. Dessa forma, independente das entradas, a saída Q é desabilitada.



No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é igual ao cadastrado nas entradas de desabilitação do bloco (HOUR_OFF, MINUTE_OFF e SECOND_OFF). Dessa forma, a saída Q é habilitada.

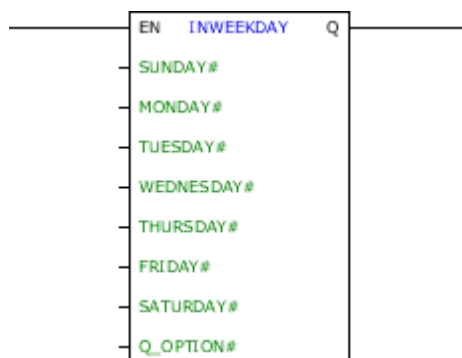


No exemplo acima, o bloco INTIME está habilitado, a entrada Q_OPTION# está habilitada para operação NORMAL e o horário atual do relógio interno do equipamento é superior ao cadastrado nas entradas de desabilitação do bloco (HOUR_OFF, MINUTE_OFF e SECOND_OFF). Dessa forma, a saída Q é desabilitada.

10.3.5.1.20.2 INWEEKDAY

Bloco que realiza uma habilitação programada para dias da semana baseado em RTC (Real Time Clock).

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	EN	BOOL	Habilitação do bloco
	SUNDAY#	BOOL	Habilitado no domingo
	MONDAY#	BOOL	Habilitado na segunda-feira
	TUESDAY#	BOOL	Habilitado na terça-feira
	WEDNESDAY#	BOOL	Habilitado na quarta-feira
	THURSDAY#	BOOL	Habilitado na quinta-feira
	FRIDAY#	BOOL	Habilitado na sexta-feira
	SATURDAY#	BOOL	Habilitado no sábado
	Q_OPTION#	BYTE	Funcionamento da saída
VAR_OUTPUT	Q	BOOL	Saída do bloco

Funcionamento

Este bloco, quando tem um valor TRUE em EN, possui dois modos de operação. Se Q_OPTION# for Normal, Q é habilitada se o dia da semana do relógio interno possuir parâmetro Enabled no bloco. Se Q_OPTION# for Inverted, Q é desabilitada se o dia da semana do relógio interno possuir parâmetro Enabled no bloco.

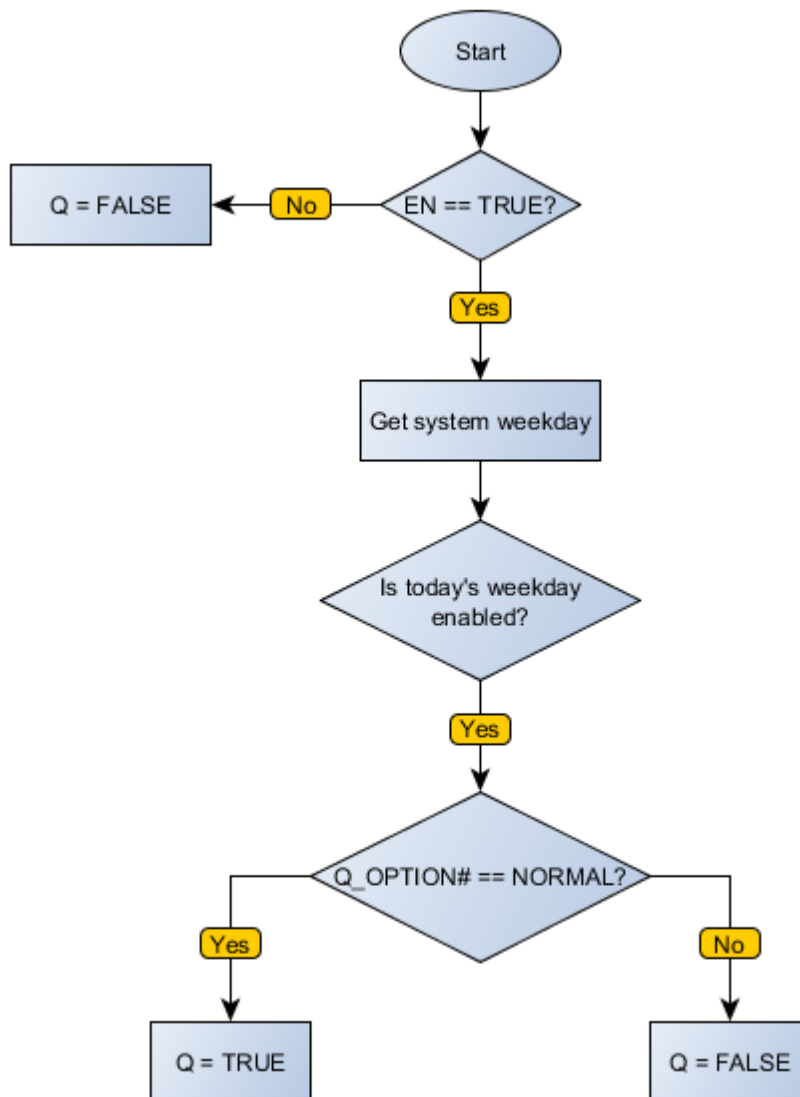
Quando EN possui valor FALSE, Q permanece FALSE.



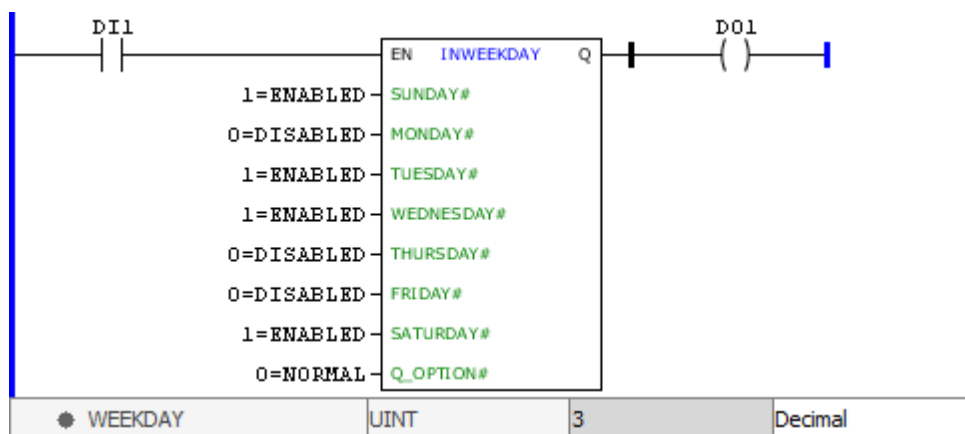
NOTA!

Os dias da semana são identificados por números, sendo Domingo o dia 0 e Sábado o dia 6.

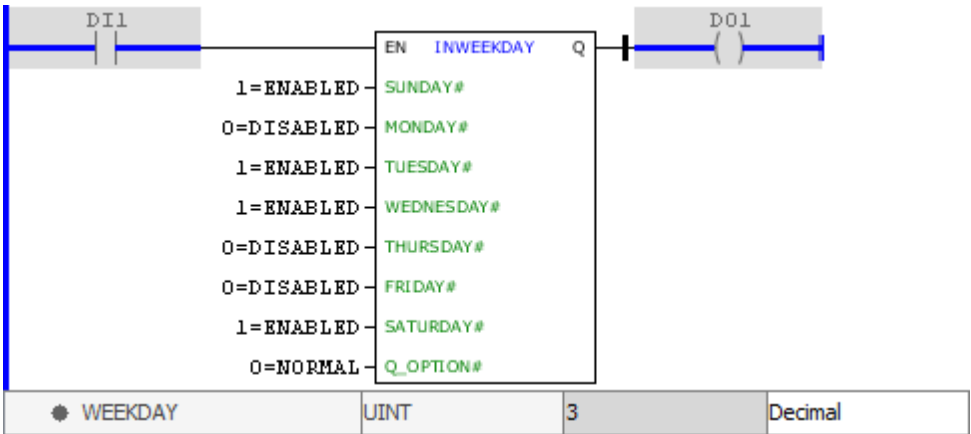
Fluxograma do Bloco



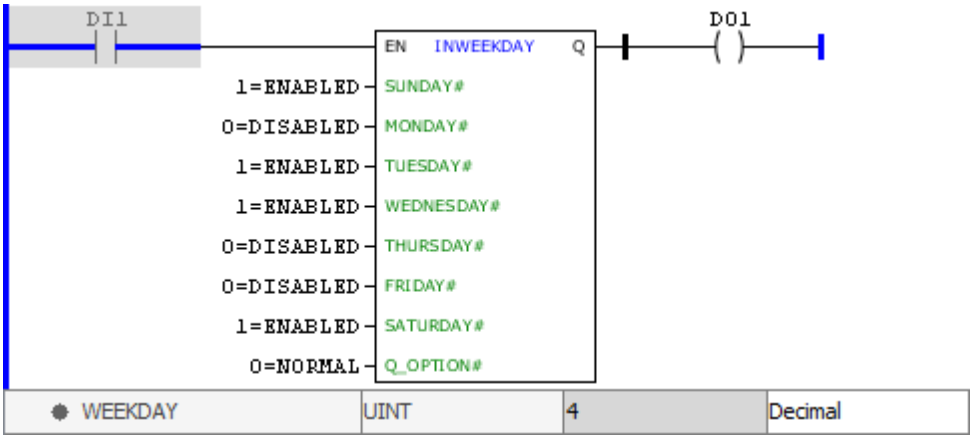
Exemplo



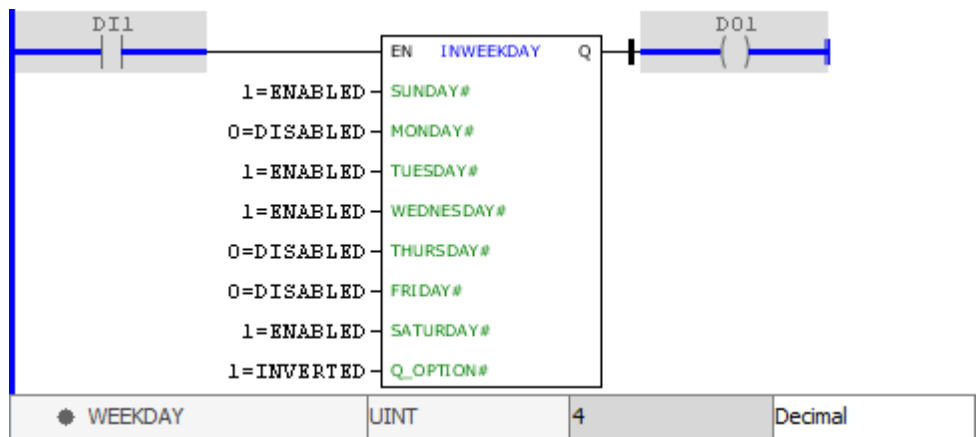
No exemplo acima, o bloco INWEEKDAY está desabilitado. Dessa forma, independente das entradas, a saída Q é desabilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação NORMAL. O dia da semana atual do relógio interno do equipamento é quarta-feira (valor 3), que possui estado ENABLED na programação. Dessa forma, a saída Q é habilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação NORMAL. O dia da semana atual do relógio interno do equipamento é quinta-feira (valor 4), que possui estado DISABLED na programação. Dessa forma, a saída Q é desabilitada.



No exemplo acima, o bloco INWEEKDAY está habilitado e a entrada Q_OPTION# está habilitada para operação INVERTED. O dia da semana atual do relógio interno do equipamento é quinta-feira (valor 4), que possui estado DISABLED na programação. Dessa forma, a saída Q é habilitada.

10.3.5.1.21 Communication Network

10.3.5.1.21.1 CANopen

Operação na Rede CANopen - Modo Mestre

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

Habilitação da Função CANopen Mestre

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

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



NOTA!

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

Características do Mestre CANopen

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

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

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



NOTA!

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

Operação do Mestre

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

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

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

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

**NOTA!**

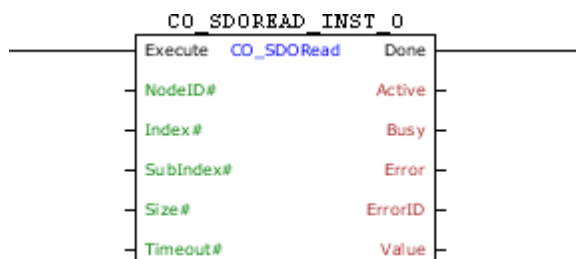
O estado da comunicação e o estado de cada escravo podem ser observados em marcadores de sistema de entrada.

Blocos para o Mestre CANopen

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

Bloco que executa uma leitura de dados via SDO de um escravo remoto em rede CANopen.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NodeID#	BYTE	Endereço do escravo
	Index#	WORD	Índice do objeto a ser acessado no escravo
	SubIndex#	BYTE	Sub-índice do objeto
	Size#	BYTE	Tamanho do dado acessado, em bytes
	Timeout#	WORD	Tempo de espera máximo para chegada do dado, a partir do início da solicitação [ms]
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que o cliente SDO está ocupado com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE USINT	Identificador do erro ocorrido
	Value	BYTE USINT	Variável que armazena os dados recebidos
VAR	CO_SDOREAD_INST_0	CO_SDOREAD	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o cliente SDO no endereço especificado em NodeID# está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de leitura para o objeto de tamanho Size# localizado em Index# e SubIndex# e seta a saída Active, resetando-a ao receber a resposta do escravo. Os dados recebidos são armazenados na variável Value. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.

**NOTA!**

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.

**NOTA!**

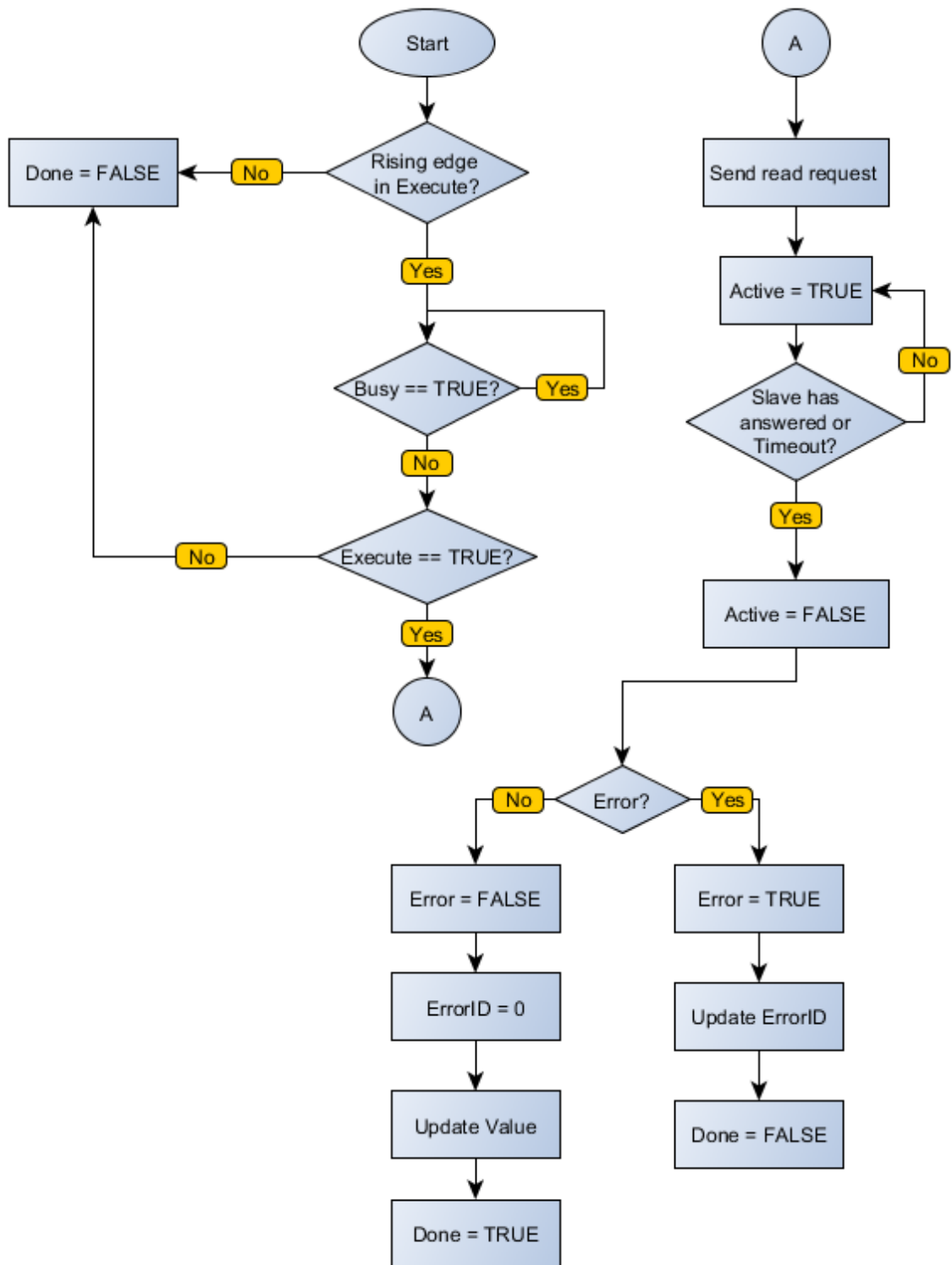
Value é um array de tamanho igual a Size#. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

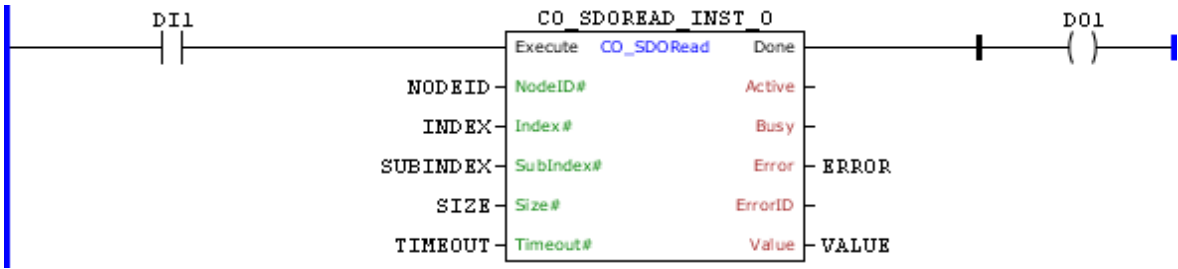
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Cartão não pode executar a função
2	Timeout na resposta do escravo
3	Escravo retornou erro

Fluxograma do Bloco



Exemplo



O exemplo acima requisita leitura do dado de tamanho SIZE, localizado em INDEX - SUBINDEX, do equipamento NODEID. Esse dado é repassado a VALUE. Finalizado o bloco com sucesso, a saída Done é ativada.

Bloco que executa uma escrita de dados via SDO de um escravo remoto em rede CANopen.

Representação Ladder



Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NodeID#	BYTE	Endereço do escravo
	Index#	WORD	Índice do objeto a ser acessado no escravo
	SubIndex#	BYTE	Sub-índice do objeto
	Size#	BYTE	Tamanho do dado acessado, em bytes
	Timeout#	WORD	Tempo de espera máximo para chegada do dado, a partir do início da solicitação [ms]
	Value	BYTE USINT	Variável que possui os dados a serem escritos
VAR_OUTPUT	Done	BOOL	Habilitação de saída
	Active	BOOL	Sinalizador de aguardo de resposta
	Busy	BOOL	Sinalizador de que o cliente SDO está ocupado com outra requisição
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	BYTE USINT	Identificador do erro ocorrido
VAR	CO_SDOWRITE_INST_0	CO_SDOWRITE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica se o cliente SDO no endereço especificado em NodeID# está livre para envio de dados (variável Busy em nível FALSE). Caso esteja, envia a requisição de escrita para o objeto de tamanho Size# localizado em Index# e SubIndex# e seta a saída Active, resetando-a ao receber a resposta do escravo. Caso o escravo não esteja livre, o bloco aguarda Busy ir para nível FALSE para reenviar a solicitação.

**NOTA!**

Caso Execute vá para nível FALSE e Busy ainda esteja em nível TRUE, a requisição é cancelada.

**NOTA!**

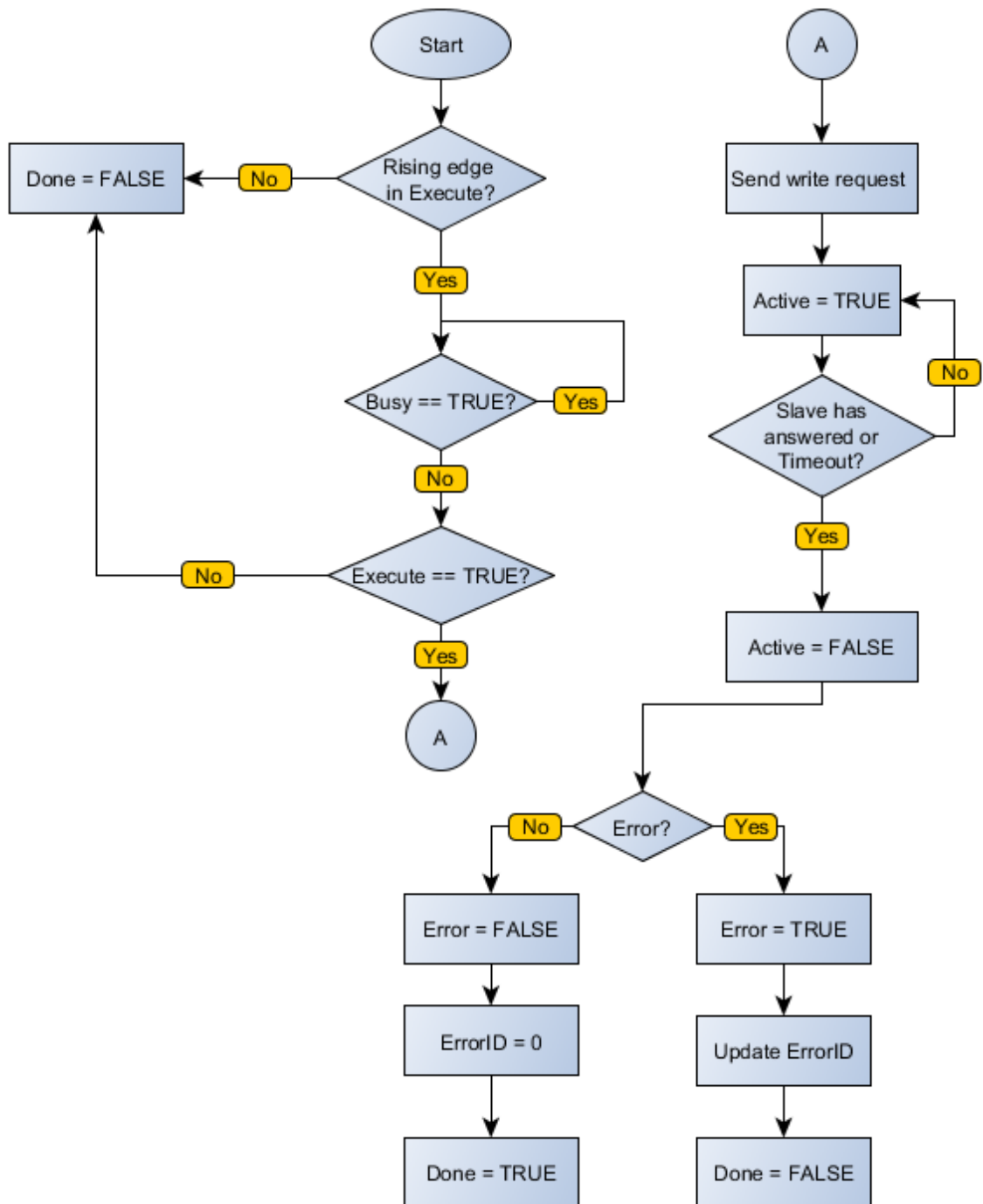
Value é um array de tamanho igual a Size#. É importante verificar esta compatibilidade para não gerar erros no bloco.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done só é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

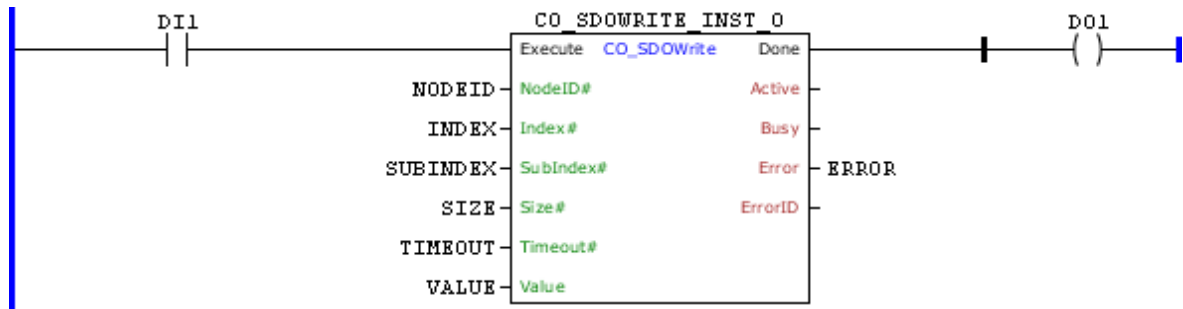
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
0	Executado com sucesso
1	Cartão não pode executar a função
2	Timeout na resposta do escravo
3	Escravo retornou erro

Fluxograma do Bloco



Exemplo



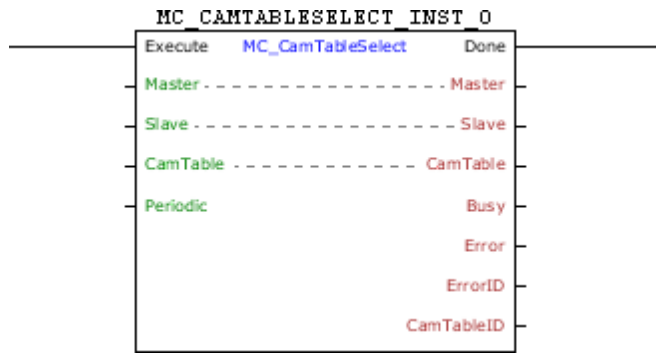
O exemplo acima requisita escrita do dado contido em VALUE, de tamanho SIZE, no local INDEX - SUBINDEX do equipamento NODEID. Finalizado o bloco com sucesso, a saída Done é ativada.

10.3.5.1.22 Motion Control Cam

10.3.5.1.22.1 MC_CamTableSelect

Bloco que seleciona uma tabela de pontos de uma curva CAM previamente programada através da ferramenta CAM Profiles.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	44 Bytes
Tamanho de Memória de Dados	16 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
	CamTable	WORD	Código da tabela CAM
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Periodic	BOOL	Controle de execução da requisição (0 - Execução única) (1 - Execução periódica)
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
	CamTableID	WORD	Tabela CAM selecionada
VAR	MC_CAMTABLESELECT_INST_0	MC_CAMTABLESELECT	Instância de acesso à estrutura do bloco

Funcionamento

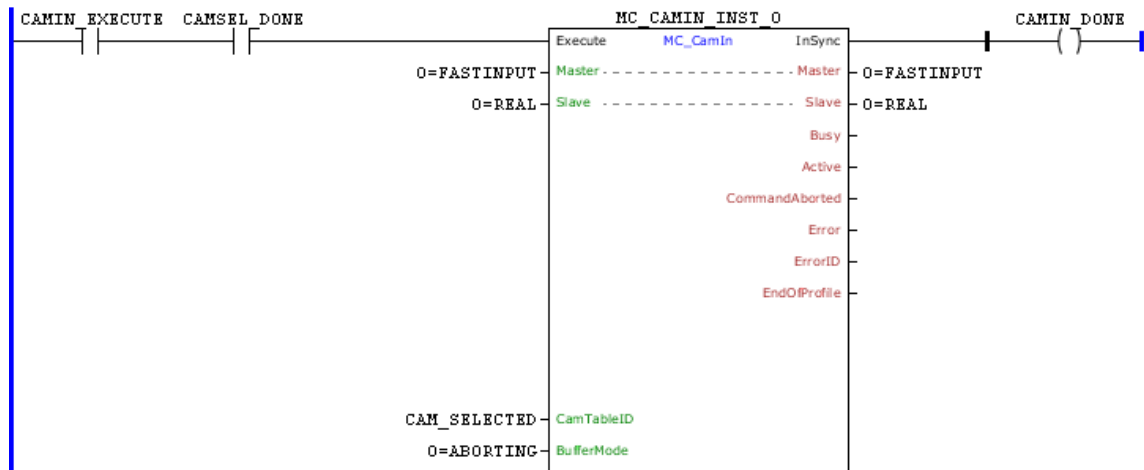
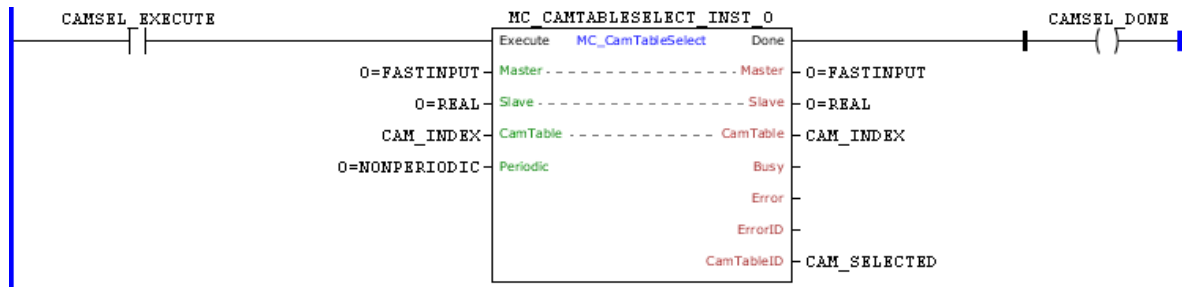
Este bloco, quando detecta uma borda de subida em Execute, busca a tabela especificada em CamTable para que esta possa ser utilizada pelo bloco MC_CamIn.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
83	Arquivo das tabelas de pontos da curva CAM inválido.
84	CAM Table inválido. CAM Table deve ser de 1 à 10.

Exemplo



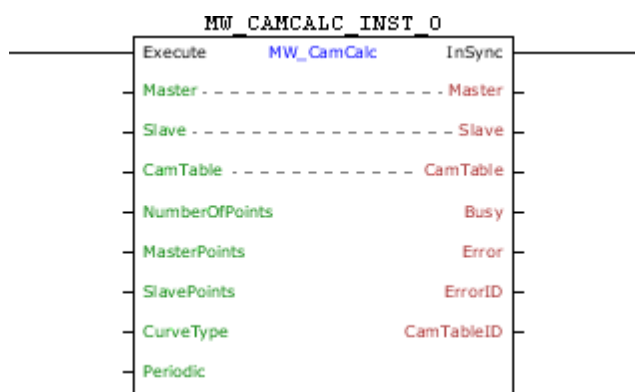
LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_CAMTABLESELECT_INST_0	0	MC_CamTableSelect		
CAMSEL_EXECUTE	0	BOOL	0	
CAM_INDEX	0	WORD	3	
CAM_SELECTED	0	WORD	3	
CAMSEL_DONE	0	BOOL	0	
MC_CAMIN_INST_0	0	MC_CamIn		
CAMIN_EXECUTE	0	BOOL	0	
CAMIN_DONE	0	BOOL	0	

Na transição de subida de CAMSEL_EXECUTE, o bloco MC_CamTableSelect é executado e, com isso, a tabela de pontos selecionada em CAM_INDEX poderá ser utilizada pelo bloco MC_CamIn. Ao executar o bloco, a saída Done e CAMSEL_DONE são setados e permanecem em nível TRUE enquanto a entrada Execute está setada. Nesse exemplo, CAMSEL_DONE garante que o bloco MC_CamIn não será acionado antes do bloco MC_CamTableSelect ser executado com sucesso.

10.3.5.1.22.2 MW_CamCalc

Bloco que calcula uma tabela de pontos de uma curva CAM.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	66 Bytes
Tamanho de Memória de Dados	24 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
	CamTable	WORD	Código da tabela CAM
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	NumberOfPoints	WORD	Número de pontos da tabela
	MasterPoints	LREAL	Pontos do mestre
	SlavePoints	LREAL	Pontos do escravo
	CurveType	WORD	Tipo de curva
	Periodic	BOOL	Controle de execução da requisição (0 - Execução única) (1 - Execução periódica)
VAR_OUTPUT	InSync	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
	CamTableID	WORD	Tabela CAM selecionada
VAR	MW_CAMCALC_INST_0	MW_CAMCALC	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, constrói a tabela especificada em

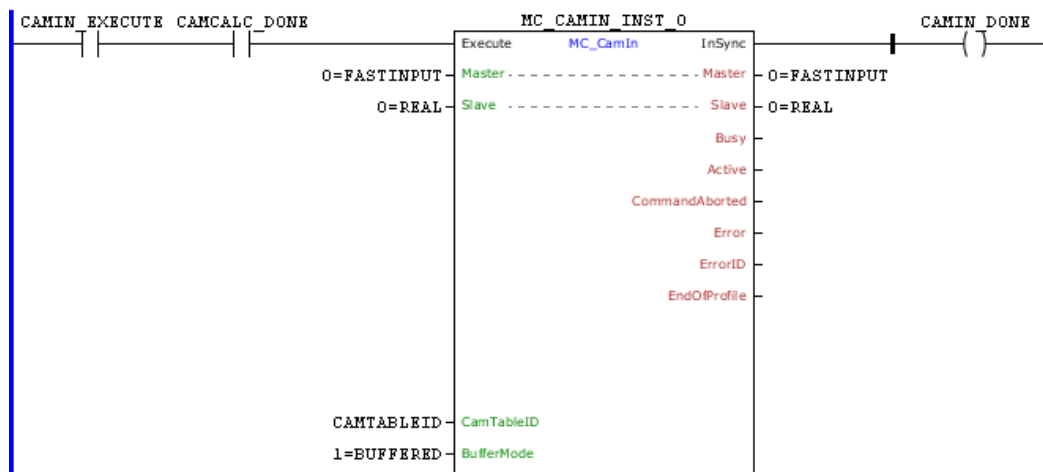
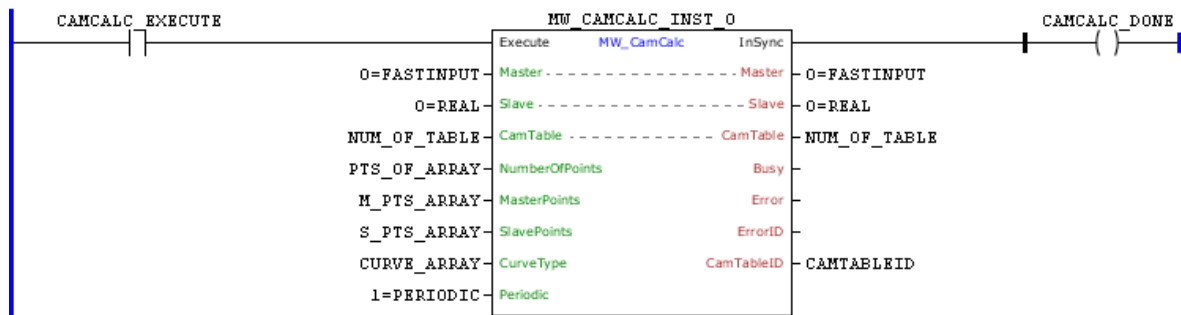
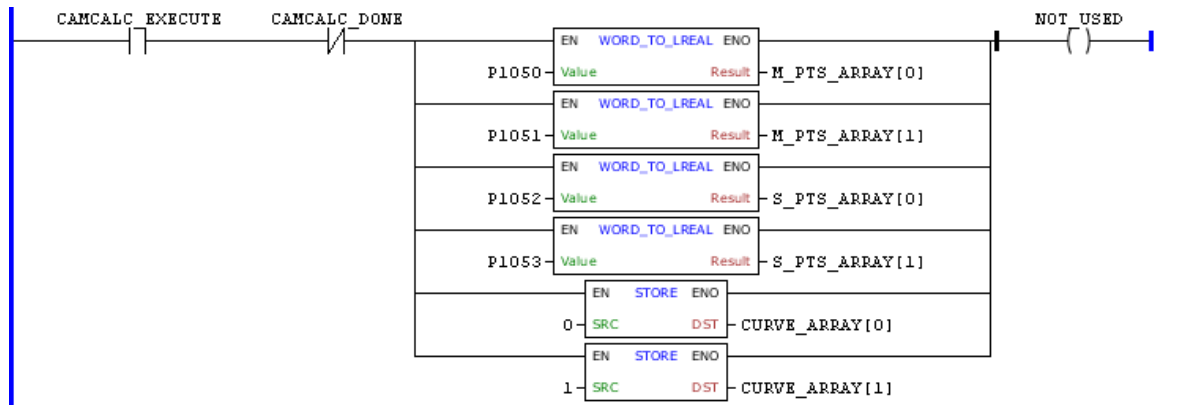
CamTable com os dados programados para que esta possa ser utilizada pelo bloco MC_CamIn.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
83	Arquivo das tabelas de pontos da curva CAM inválido.
84	CAM Table inválido. CAM Table deve ser de 11 à 20.
86	Número de pontos maior que o programado no configurador CAM PROFILES.
87	Posição do eixo mestre inválida. A posição do eixo mestre deve ser maior que a posição do ponto anterior.
88	Bloco MW_CamCalc em execução. Somente é permitido a execução de um bloco MW_CamCalc de cada vez.
89	Tabela de pontos em uso pelo bloco MC_CamIn.
90	Marcador de double com posição do eixo mestre inexistente.
91	Marcador de double com posição do eixo escravo inexistente.
92	Marcador de word com tipo de curva inexistente.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
NOT_USED	0	BOOL	0	
MW_CAMCALC_INST_0	0	MW_CamCalc		
CAMCALC_EXECUTE	0	BOOL	0	
CAMCALC_DONE	0	BOOL	0	
CAMTABLEID	0	WORD	11	
PTS_OF_ARRAY	0	WORD	2	
M_PTS_ARRAY	2	LREAL		
M_PTS_ARRAY[0]	0	LREAL	3.0	
M_PTS_ARRAY[1]	0	LREAL	7.0	
S_PTS_ARRAY	2	LREAL		
S_PTS_ARRAY[0]	0	LREAL	10.0	
S_PTS_ARRAY[1]	0	LREAL	-5.0	
CURVE_ARRAY	2	WORD		
CURVE_ARRAY[0]	0	WORD	0	
CURVE_ARRAY[1]	0	WORD	1	
NUM_OF_TABLE	0	WORD	11	
MC_CAMIN_INST_0	0	MC_CamIn		
CAMIN_EXECUTE	0	BOOL	0	
CAMIN_DONE	0	BOOL	0	

GLOBAL LOCAL

CAM Profile List			
Cam Table	Cam Type	Cam File	Max Points
1	Fixed	---	0
2	Fixed	---	0
3	Fixed	---	0
4	Fixed	---	0
5	Fixed	---	0
6	Fixed	---	0
7	Fixed	---	0
8	Fixed	---	0
9	Fixed	---	0
10	Fixed	---	0
11	Calculable		2
12	Calculable		0
13	Calculable		0
14	Calculable		0
15	Calculable		0
16	Calculable		0
17	Calculable		0
18	Calculable		0
19	Calculable		0
20	Calculable		0

Editar Perfil

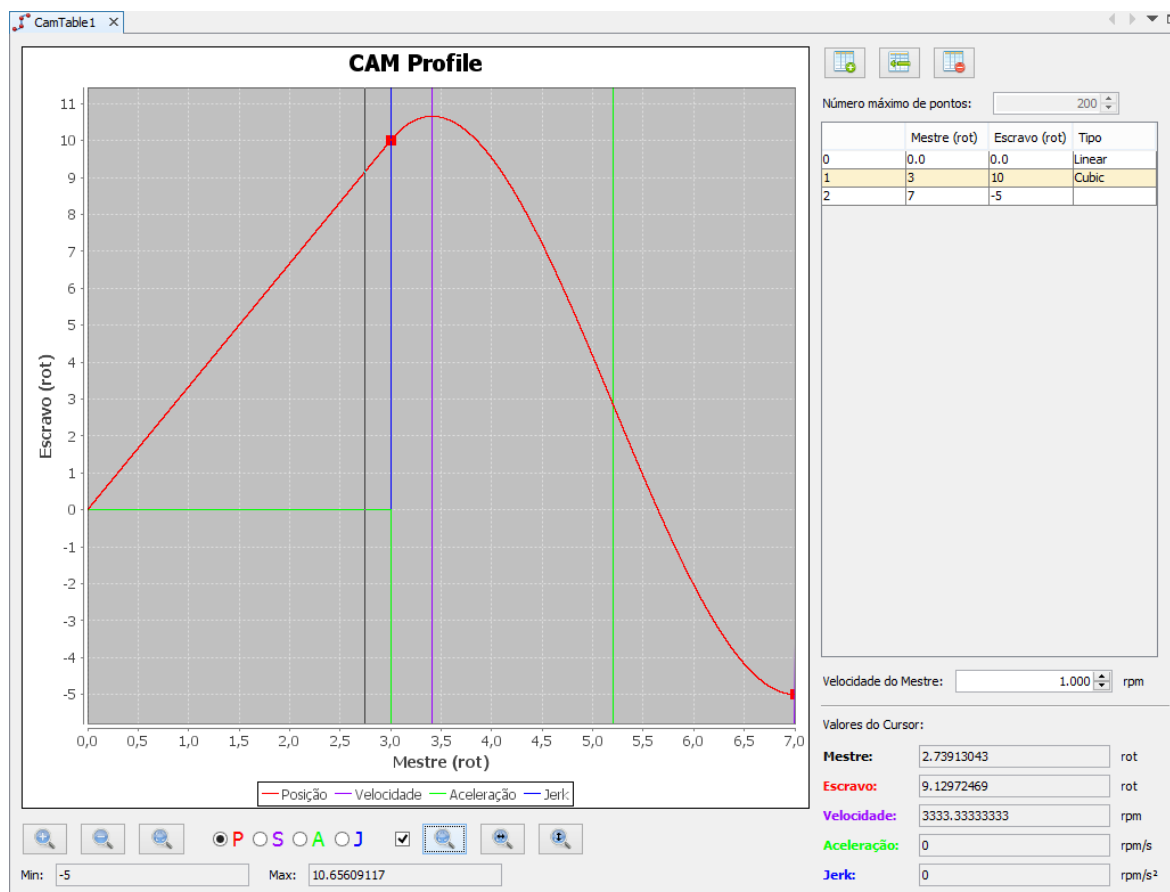
Selecionar Perfil

Desconsiderar Perfil

Na transição de subida do marcador de bit CAMCALC_EXECUTE, o bloco MW_CamCalc é executado e a tabela de pontos de TABLE será calculada de acordo com os argumentos do bloco.

Nesse exemplo, o número de pontos da curva será o conteúdo de NUM_POINTS, a posição do eixo mestre será de acordo com os conteúdos de MASTER_POINTS[0] e MASTER_POINTS[1], a posição do eixo escravo será de acordo com os conteúdos de SLAVE_POINTS[0] e SLAVE_POINTS[1] e o tipo da curva será de acordo com os conteúdos de CURVE_TYPE[0] e CURVE_TYPE[1].

Colocando os mesmos valores na ferramenta CAM Profiles podemos observar a curva abaixo:



Ao finalizar o cálculo da tabela de pontos, a saída InSync é setada enquanto a entrada Execute permanece setada.

Com CAMCALC_DONE setado, o bloco MC_CamIn poderá ser executado.

10.3.5.1.22.3 MC_CamIn

Bloco responsável pela execução de um posicionamento definido por uma tabela de pontos de uma curva CAM.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	92 Bytes
Tamanho de Memória de Dados	52 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	CamTableID	WORD	Tabela CAM selecionada
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InSync	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
	EndOfProfile	BOOL	Sinalizador de fim do perfil CAM
VAR	MC_CAMIN_INST_0	MC_CAMIN	Instância de acesso à estrutura do bloco

Funcionamento

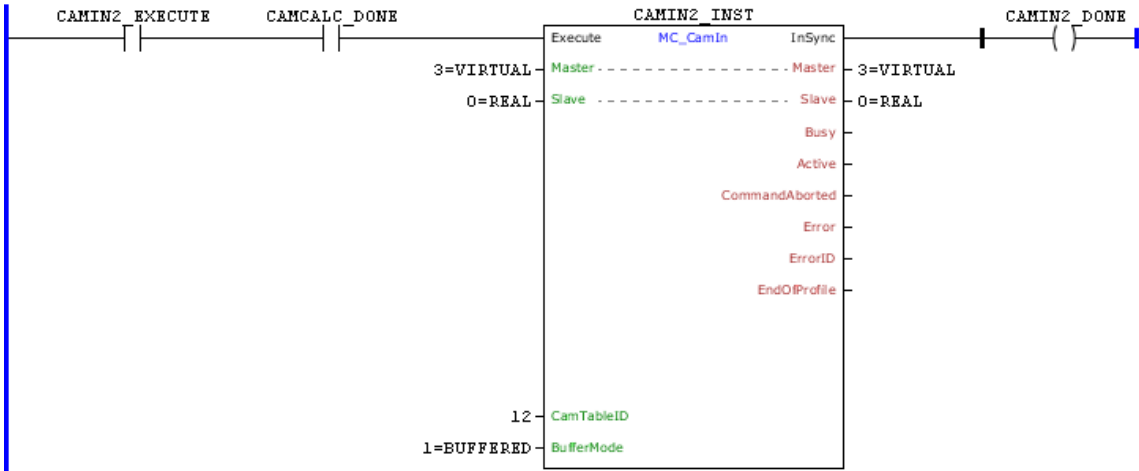
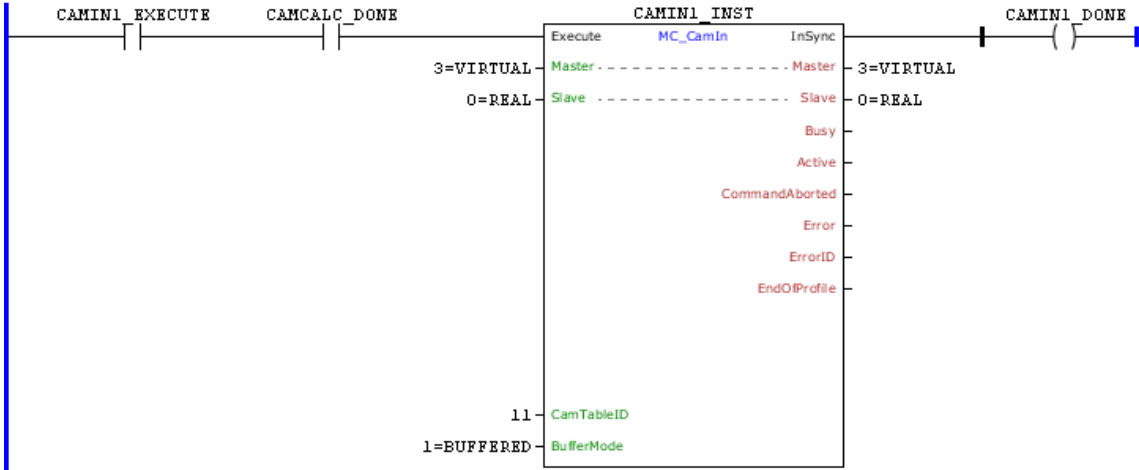
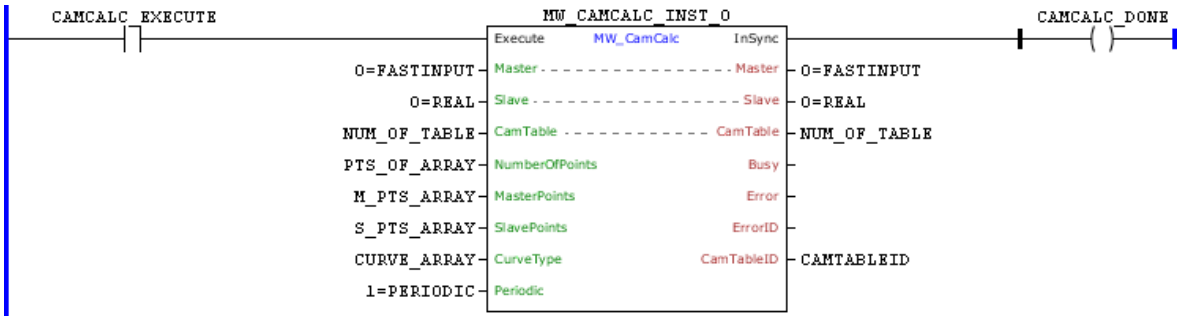
Este bloco, quando detecta uma borda de subida em Execute, envia um comando para o *drive* executar um posicionamento definido por CamTableID.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
85	CamTableID inválido.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
◆ CAMIN1_INST	0	MC_CamIn		
◆ CAMIN1_EXECUTE	0	BOOL	0	
◆ CAMIN1_DONE	0	BOOL	0	
◆ CAMIN2_INST	0	MC_CamIn		
◆ CAMIN2_EXECUTE	0	BOOL	0	
◆ CAMIN2_DONE	0	BOOL	0	
◆ MW_CAMCALC_INST_0	0	MW_CamCalc		
◆ NUM_OF_TABLE	0	WORD	11	
◆ PTS_OF_ARRAY	0	WORD	2	
◆ M_PTS_ARRAY	2	LREAL		
◆ M_PTS_ARRAY[0]	0	LREAL	3.0	
◆ M_PTS_ARRAY[1]	0	LREAL	7.0	
◆ S_PTS_ARRAY	2	LREAL		
◆ S_PTS_ARRAY[0]	0	LREAL	10.0	
◆ S_PTS_ARRAY[1]	0	LREAL	-5.0	
◆ CURVE_ARRAY	2	WORD		
◆ CURVE_ARRAY[0]	0	WORD	0	
◆ CURVE_ARRAY[1]	0	WORD	1	
◆ CAMTABLEID	0	WORD	11	
GLOBAL LOCAL				

Na transição de subida de CAMCALC_EXECUTE, o bloco MW_CamCalc é executado e a tabela de pontos NUM_OF_TABLE será calculada de acordo com os argumentos do bloco.

Ao finalizar o cálculo da tabela de pontos, a saída InSync é setada enquanto a entrada Execute permanece setada.

Com CAMCALC_DONE setado, o bloco MC_CamIn poderá ser executado.

Na transição de subida de CAMIN1_EXECUTE, o primeiro bloco MC_CamIn é executado.

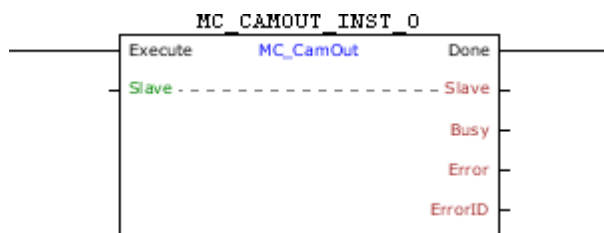
Caso houver a necessidade de algum ajuste na tabela de pontos da curva CAM, basta fazer o ajuste em MASTER_PT[0], MASTER_PT[1], SLAVE_PT[0] e SLAVE_PT[1], mudar o conteúdo de NUM_OF_TABLE para 12 e executar novamente o bloco MW_CamCalc.

Na transição de subida de CAMIN2_EXECUTE, o segundo bloco MC_CamIn será executado (sem perda de posição do eixo mestre) assim que o primeiro bloco MC_CamIn terminar de executar a curva em execução.

10.3.5.1.22.4 MC_CamOut

Bloco responsável pela finalização de uma sincronia estabelecida por um bloco MC_CamIn.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	28 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Slave	BYTE	Seleção de escravo da operação (0 - Exo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	CamTableID	WORD	Tabela CAM selecionada
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_CAMOUT_INST_0	MC_CAMOUT	Instância de acesso à estrutura do bloco

Funcionamento

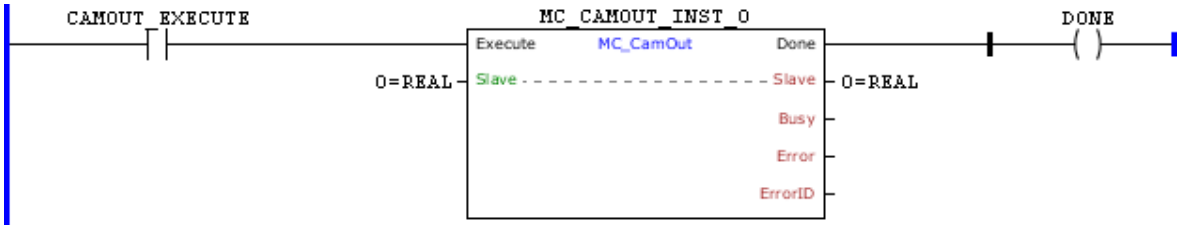
Este bloco, quando detecta uma borda de subida em Execute, finaliza o sincronismo existente pela última execução de um bloco MC_CamIn. O eixo manterá a velocidade do instante em que o bloco é executado.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
73	Drive não está no estado "Synchronized Motion"
78	Bloco MC não executado - Falha interna.

Exemplo



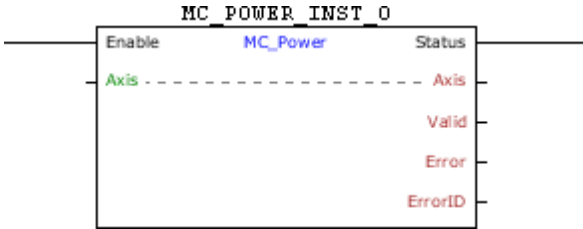
Na transição de subida de CAMOUT_EXECUTE, o bloco MC_CamOut é executado. Com isso, os sinais Busy e Active deste bloco são setados e finaliza-se o sincronismo iniciado por outros blocos CAM. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.3.5.1.23 Motion Control Command

10.3.5.1.23.1 MC_Power

Bloco responsável pela habilitação/desabilitação de eixo do *drive*.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	40 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Enable	BOOL	Habilitação do bloco
VAR_OUTPUT	Status	BOOL	Habilitação da saída
	Valid	BOOL	Sinalizador de validade dos sinais de saída
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_POWER_INST_0	MC_POWER	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco realiza um comando de habilitação/deshabilitação do eixo Axis de acordo com a entrada Enable, desabilitando se Enable for FALSE e habilitando se Enable for TRUE.



NOTA!

Quando o bloco MC_Power for usado para habilitação/deshabilitação do eixo real, nenhuma entrada digital deverá estar programada para a função de Habilitação (opção 1), ou o Alarme A0120 poderá ocorrer.

Ao habilitar o eixo real pela primeira vez, o *drive* poderá operar em malha de posição, dependendo do valor do parâmetro P0773. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

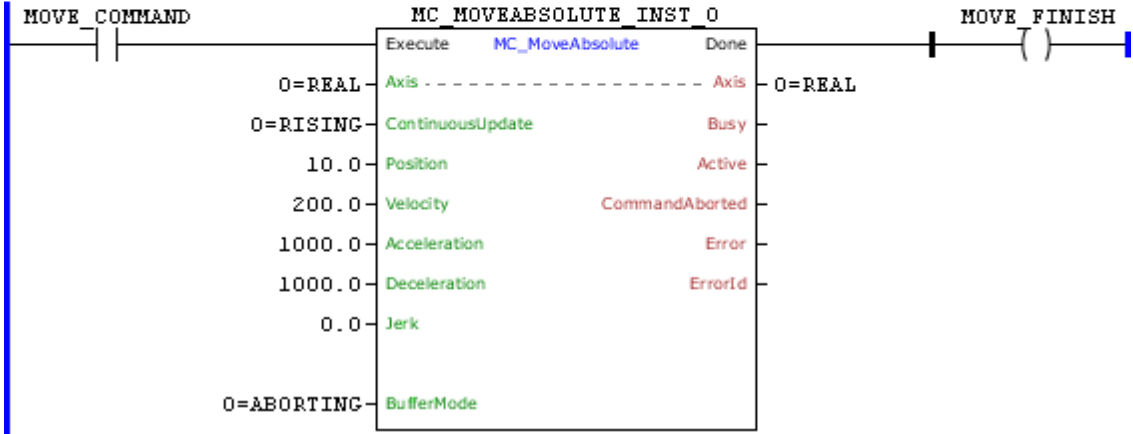
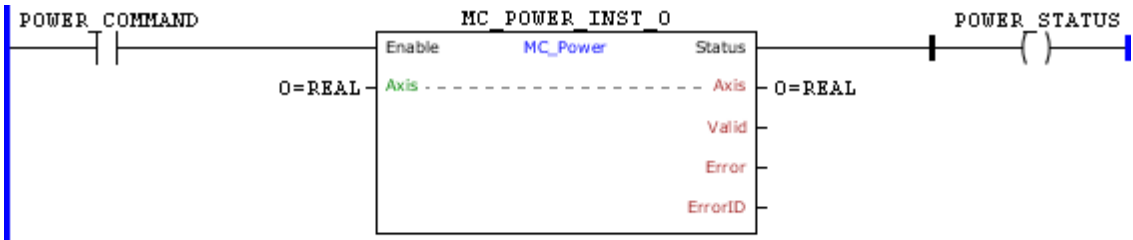
Quando o eixo real estiver desabilitado, o estado do eixo será Disabled. Ao habilitar o eixo real, o estado do eixo mudará para Standstill.

Quando Enable possui valor FALSE, Status permanece FALSE. A saída Status é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Enable receba FALSE.

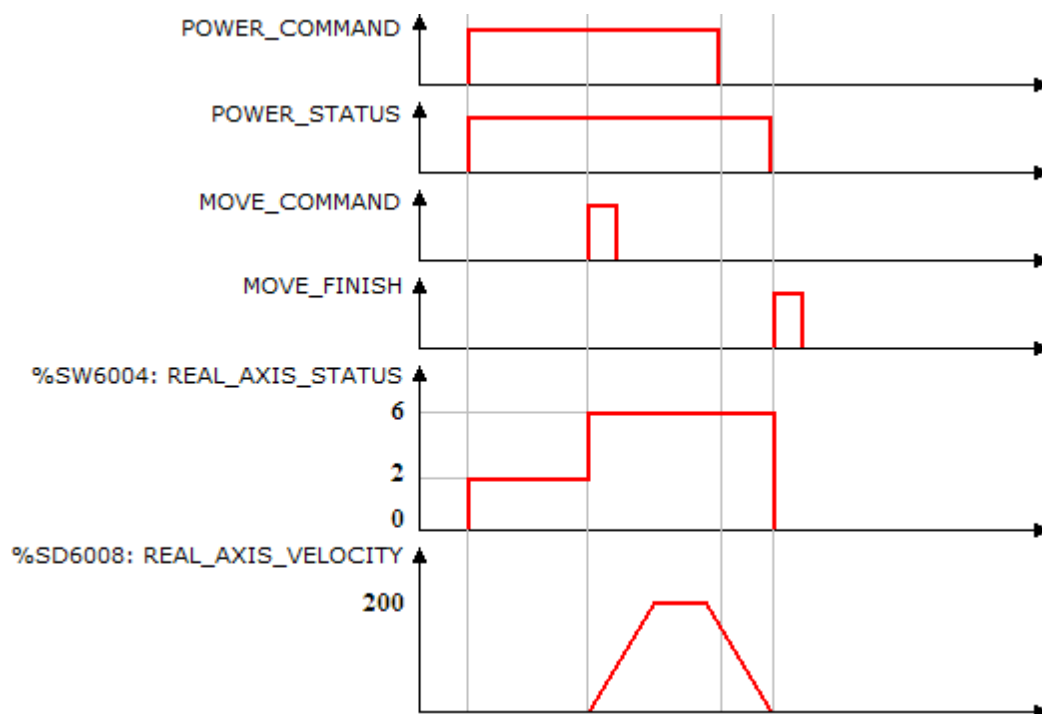
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
66	Drive no estado "ErrorStop".
71	P202 diferente de 4.

Exemplo



LOCAL				
LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_POWER_INST_0	0	MC_Power		
POWER_COMMAND	0	BOOL	0	
MC_MOVEABSOLUTE_INST_0	0	MC_MoveAbsolute		
POWER_STATUS	0	BOOL	0	
MOVE_COMMAND	0	BOOL	0	
MOVE_FINISH	0	BOOL	0	



Na borda de subida de `POWER_COMMAND`, o eixo real é habilitado e o seu estado, o marcador do sistema `REAL_AXIS_STATUS` (%SW6004), é alterado para Standstill (%SW6004 = 2). A saída Status é setada.

Tendo-se a transição de subida de `MOVE_COMMAND`, o bloco `MC_MoveAbsolute` é executado e inicia-se o posicionamento para a posição 10 voltas. O estado do eixo é alterado para Discrete Motion (%SW3406 = 6).

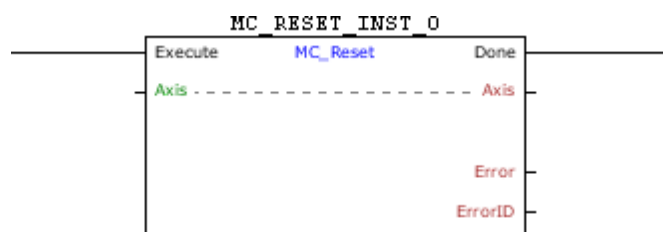
Enquanto o posicionamento é executado, `POWER_COMMAND` é resetado. Como o `BufferMode` do `MC_Power` está configurado como `Buffered`, o eixo somente será desabilitado na conclusão do posicionamento.

Ao finalizar o posicionamento, a saída `Done` do bloco `MC_MoveAbsolute` é setada por 1 ciclo de scan e o eixo é desabilitado (`MC_Power.Enable` = 0). O estado do eixo é alterado para Disabled (%SW6004 = 0).

10.3.5.1.23.2 MC_Reset

Bloco responsável pela limpeza do estado de falha do *drive*.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	28 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_RESET_INST_0	MC_RESET	Instância de acesso à estrutura do bloco

Funcionamento

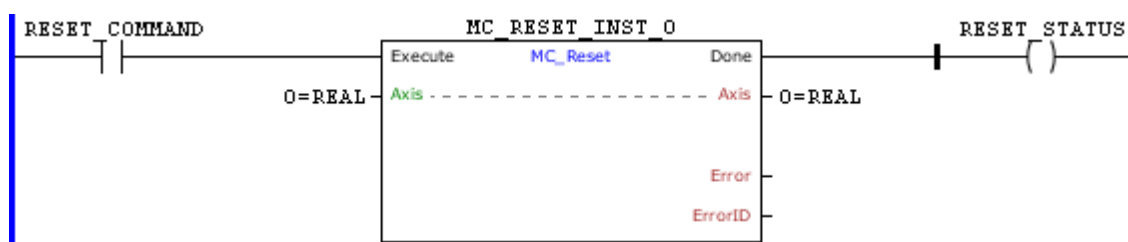
Este bloco, quando detecta uma borda de subida em Execute, realiza uma limpeza no estado do *drive*, alterando-o de Errorstop para Disabled.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
71	P202 diferente de 4.

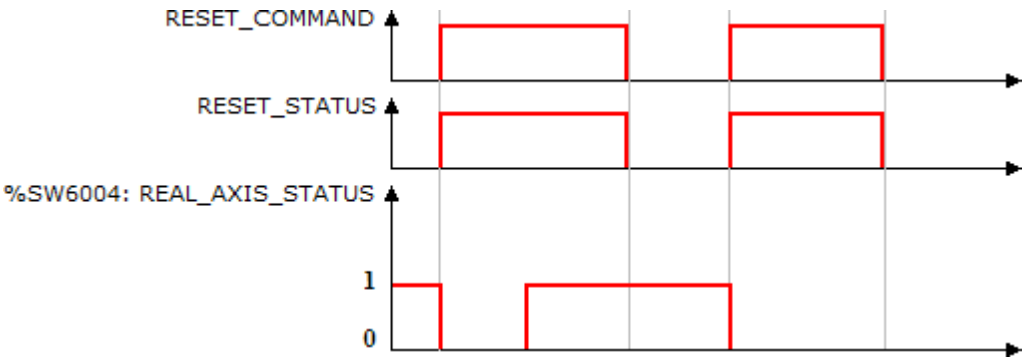
Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_RESET_INST_0	0	MC_Reset		
RESET_COMMAND	0	BOOL	0	
RESET_STATUS	0	BOOL	0	

GLOBAL

LOCAL

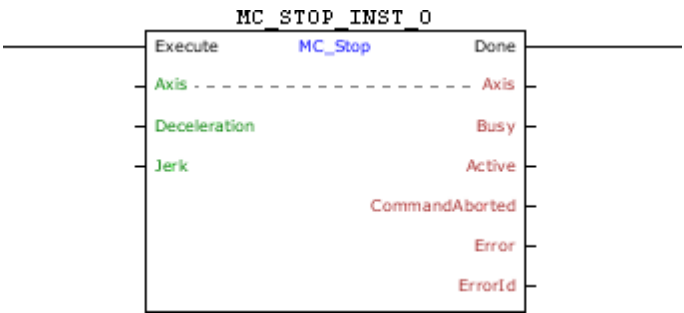


Ocorrendo alguma falha no *drive*, o estado do eixo mudará para Errorstop (%SW6004 = 1). Quando ocorrer uma transição de subida de RESET_COMMAND, o bloco MC_Reset será executado e o estado do eixo mudará para Disabled (%SW6004 = 0). A saída Done permanecerá setada enquanto a entrada Execute estiver em nível TRUE.

10.3.5.1.23.3 MC_Stop

Bloco responsável pela execução de uma parada controlada.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	52 Bytes
Tamanho de Memória de Dados	12 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Deceleration	REAL	Desaceleração da parada
	Jerk	REAL	Jerk
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STOP_INST_0	MC_STOP	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para uma parada controlada do eixo. Enquanto Execute possuir nível TRUE, nenhum outro bloco MC é executado.

Ao executar o bloco MC_Stop, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

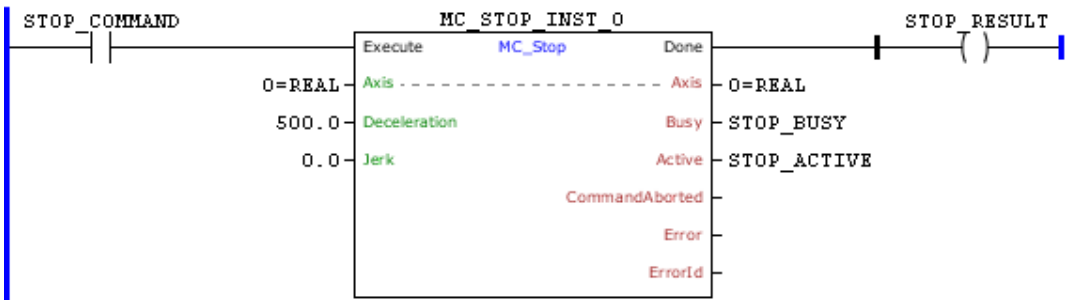
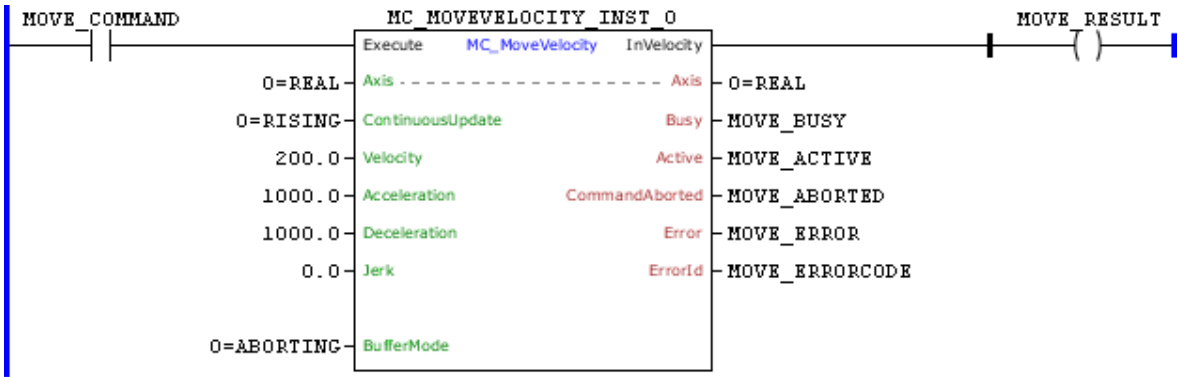
Na execução do bloco o estado do eixo mudará para Stopping. Ao finalizar a parada e o bloco não estiver mais ativo, o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

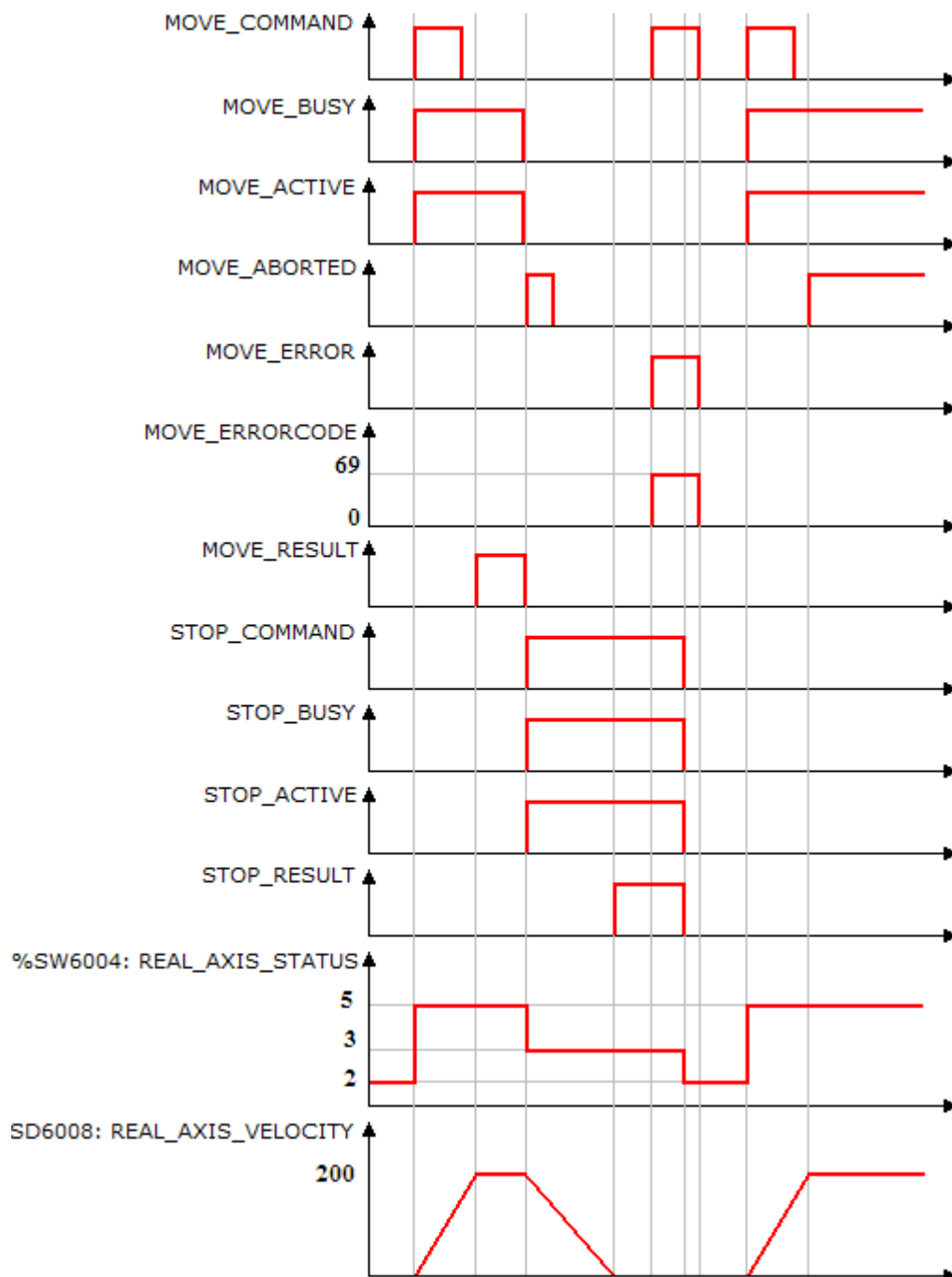
Código	Descrição
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_MOVEVELOCITY_INST_0	0	MC_MoveVelocity		
MC_STOP_INST_0	0	MC_Stop		
MOVE_COMMAND	0	BOOL	0	
MOVE_RESULT	0	BOOL	0	
MOVE_BUSY	0	BOOL	0	
MOVE_ACTIVE	0	BOOL	0	
MOVE_ABORTED	0	BOOL	0	
MOVE_ERROR	0	BOOL	0	
MOVE_ERRORCODE	0	WORD	0	
STOP_COMMAND	0	BOOL	0	
STOP_RESULT	0	BOOL	0	
STOP_BUSY	0	BOOL	0	
STOP_ACTIVE	0	BOOL	0	

GLOBAL LOCAL



Na transição de subida de **MOVE_COMMAND**, o bloco **MC_MoveVelocity** é executado. Com isso, os sinais **Busy** e **Active** deste bloco são setados e inicia-se o movimento para chegar à velocidade de 200 RPM. O estado do eixo (**%SW6004**) muda de Standstill (2) para Continuous Motion (5). No instante em que a velocidade atinge 200 RPM, **MOVE_RESULT** é setado.

Tendo-se a transição de subida de **STOP_COMMAND**, o bloco **MC_Stop** é instantaneamente executado, com isso os sinais **Busy** e **Active** deste bloco são setados e se inicia a parada. Ao mesmo tempo os sinais **Busy**, **Active** e **InVelocity** do bloco **MC_MoveVelocity** são resetados e o sinal **CommandAborted** é setado por 1 ciclo de scan. O estado do eixo (**%SW6004**) muda de ContinuousMotion (5) para Stopping (3).

Ao finalizar a parada, a saída Done do bloco MC_Stop é setada e permanece até a entrada Execute seja resetada. O estado do eixo (%SW6004) permanece igual a Stopping (3) e nenhum outro bloco MC será executado.

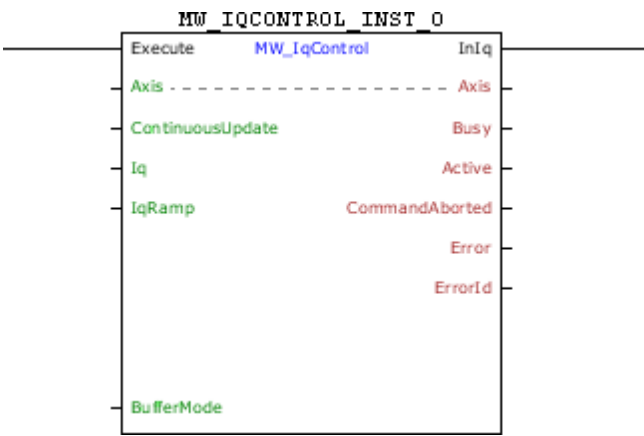
Tendo-se a transição de subida de MOVE_COMMAND, o bloco MC_MoveVelocity é iniciado, mas como o bloco MC_Stop está ativo, ocorrerá erro e o sinal Error será setado, inserindo em ErrorID o valor 69.

Quando a entrada Execute do bloco MC_Stop é resetada, os sinais Busy, Active e Done são resetados. O estado do eixo (%SW6004) muda de Stopping (3) para Standstill (2) e outros blocos MC poderão ser executados.

10.3.5.1.23.4 MW_IqControl

Bloco responsável pela realização do controle de Iq programado.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	78 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Iq	REAL	Valor de Iq
	IqRamp	REAL	Valor da rampa de Iq
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InIq	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MW_IQCONTROL_INST_0	MW_IQCONTROL	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando execução do controle de Iq de acordo com os parâmetros programados.

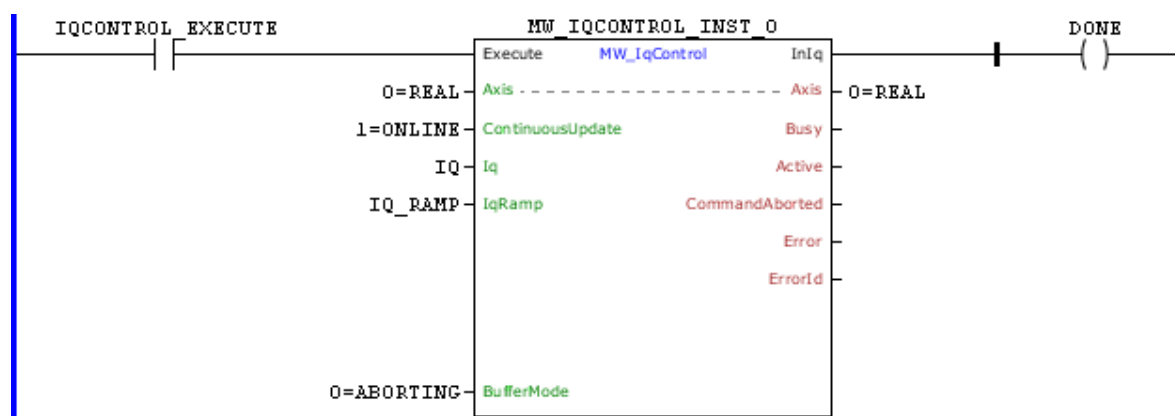
Na execução do bloco o estado do eixo mudará para Continuous Motion. Para finalizar o bloco, é necessária a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
80	Iq programado maior que o máximo permitido.
81	IqRamp programado menor que o mínimo permitido.
82	IqRamp programado maior que o máximo permitido.

Exemplo



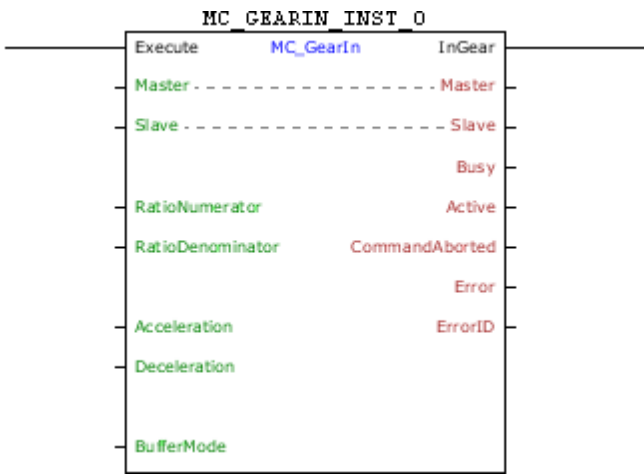
Na transição de subida de IQCONTROL_EXECUTE, o bloco MW_IqControl é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se a execução do controle de Iq de acordo com os parâmetros configurados. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.3.5.1.24 Motion Control Gear

10.3.5.1.24.1 MC_GearIn

Bloco responsável pela execução de sincronismo em velocidade entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	74 Bytes
Tamanho de Memória de Dados	28 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	RatioNumerator	INT	Numerador da relação de sincronismo
	RatioDenominator	WORD	Denominador da relação de sincronismo
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InGear	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEARIN_INST_0	MC_GEARIN	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para sincronismo em velocidade entre os eixos programados.

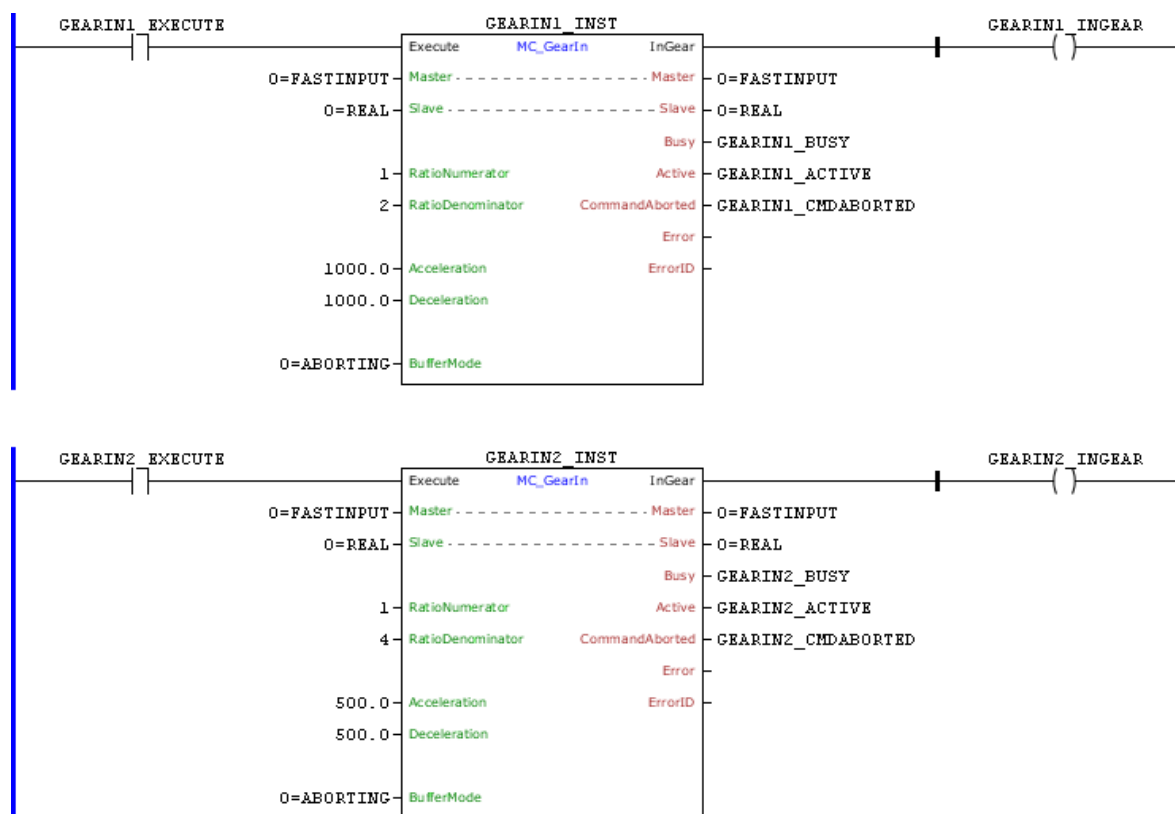
Para o eixo escravo atingir a velocidade do eixo mestre, será realizado um movimento com uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. A direção do movimento dependerá do sinal do RatioNumerator. Se RatioNumerator for maior que zero, o movimento será na mesma direção do eixo mestre e se RatioNumerator for menor que zero, o movimento será na direção oposta ao eixo mestre.

A saída InGear é ativada quando a sincronia é atingida. Para finalizar o bloco, é necessário a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

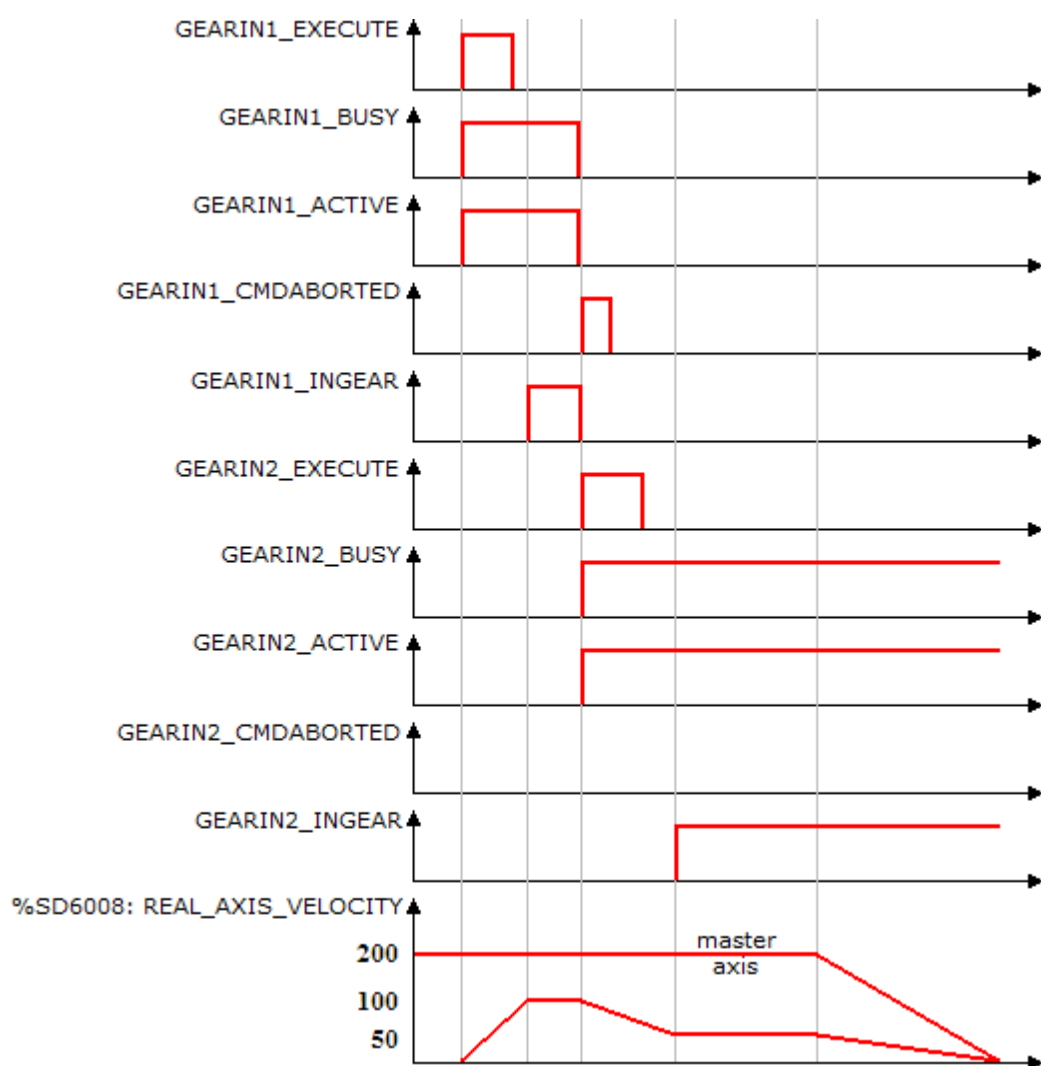
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
72	Relação de sincronismo inválida.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
GEARIN1_INST	0	MC_GearIn		
GEARIN1_EXECUTE	0	BOOL	0	
GEARIN1_INGEAR	0	BOOL	0	
GEARIN1_BUSY	0	BOOL	0	
GEARIN1_ACTIVE	0	BOOL	0	
GEARIN1_CMDABORTED	0	BOOL	0	
GEARIN2_INST	0	MC_GearIn		
GEARIN2_EXECUTE	0	BOOL	0	
GEARIN2_INGEAR	0	BOOL	0	
GEARIN2_BUSY	0	BOOL	0	
GEARIN2_ACTIVE	0	BOOL	0	
GEARIN2_CMDABORTED	0	BOOL	0	



Na transição de subida de GEARIN1_EXECUTE, o primeiro bloco MC_GearIn é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca do sincronismo com a

aceleração configurada. Como a relação configurada é 1:2 e o eixo mestre está a 200 RPM, o eixo escravo deverá atingir 100 RPM para estabelecer o sincronismo. No instante em que a velocidade atinge 100 RPM, a saída InGear é setada.

Tendo-se a transição de subida de GEARIN2_EXECUTE, o segundo bloco MC_GearIn é instantaneamente executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca do sincronismo com a desaceleração configurada. Como a relação configurada é 1:4 e o eixo mestre está a 200 RPM, o eixo escravo deverá atingir 50 RPM para estabelecer o sincronismo. Ao mesmo tempo os sinais Busy, Active e InGear do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

Ao atingir a velocidade de 50 RPM, a saída InGear do segundo bloco é setada e permanece até a execução de outro bloco.

10.3.5.1.24.2 MC_GearInPos

Bloco responsável pela execução de sincronismo em posição entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	104 Bytes
Tamanho de Memória de Dados	56 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	RatioNumerator	INT	Numerador da relação de sincronismo
	RatioDenominator	WORD	Denominador da relação de sincronismo
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	BufferMode	BYTE	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InSync	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEARINPOS_INST_0	MC_GEARINPOS	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para sincronismo em posição entre os eixos programados.

Para o eixo escravo atingir a velocidade do eixo mestre, será realizado um movimento com uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. A direção do movimento dependerá do sinal do RatioNumerator. Se RatioNumerator for maior que zero, o movimento será na mesma direção do eixo mestre e se RatioNumerator for menor que zero, o movimento será na direção oposta ao eixo mestre.

Ao executar o bloco MC_GearInPos, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

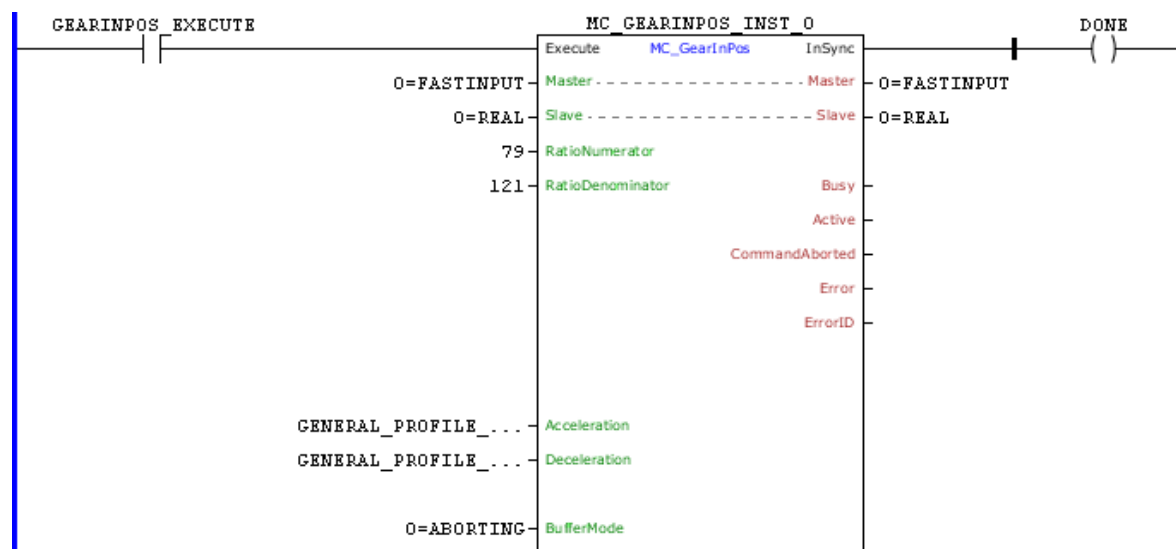
Na execução do bloco o estado do eixo mudará para Synchronized Motion.

A saída InSync é ativada quando a sincronia é atingida. Para finalizar o bloco, é necessário a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
72	Relação de sincronismo inválida.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.

Exemplo



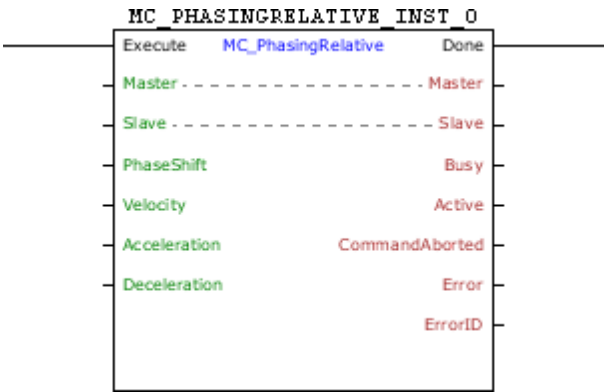
Na transição de subida de GEARINPOS_EXECUTE, o bloco MC_GearInPos é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca do sincronismo com a aceleração configurada. Como a relação configurada é 79:121, se o eixo mestre está a 200 RPM, o

eixo escravo deverá atingir 131 RPM para estabelecer o sincronismo.
No instante em que a velocidade atinge 131 RPM, a saída InSync é setada.

10.3.5.1.24.3 MC_PhasingRelative

Bloco responsável pela execução de um defasamento em posição entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	78 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Master	BYTE	Seleção de mestre da operação (0 - Entradas digitais rápidas) (1 - CANopen) (2 - Encoder 1) (3 - Eixo virtual) (4 - Encoder 2)
	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	PhaseShift	REAL	Diferença de fase entre mestre e escravo
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEARIN_INST_0	MC_GEARIN	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, verifica o sincronismo entre mestre e escravo e envia um comando para deslocar o eixo mestre de forma a deixá-lo defasado do escravo na magnitude de PhaseShift.

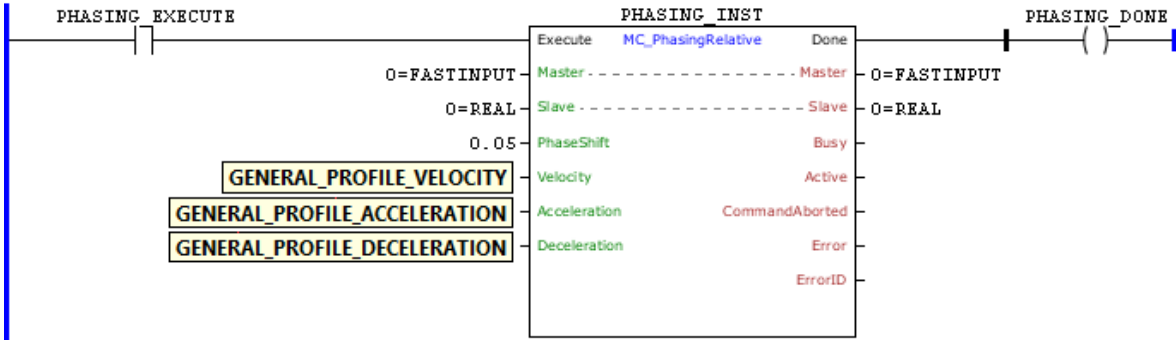
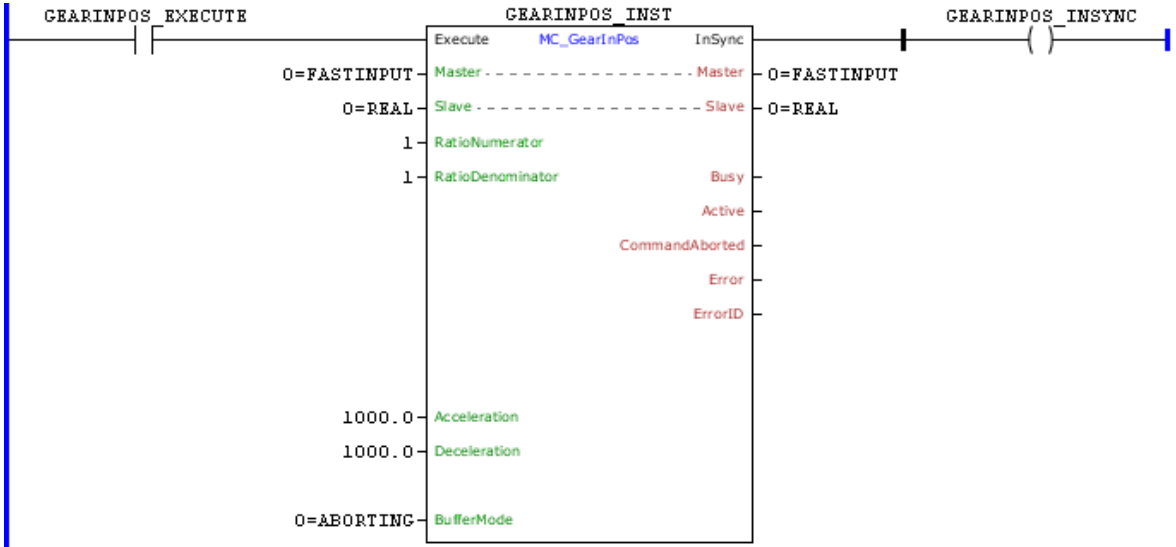
Ao executar o bloco MC_PhasingRelative, o *drive* não altera o modo de operação atual. Na execução do bloco o estado do eixo não mudará.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

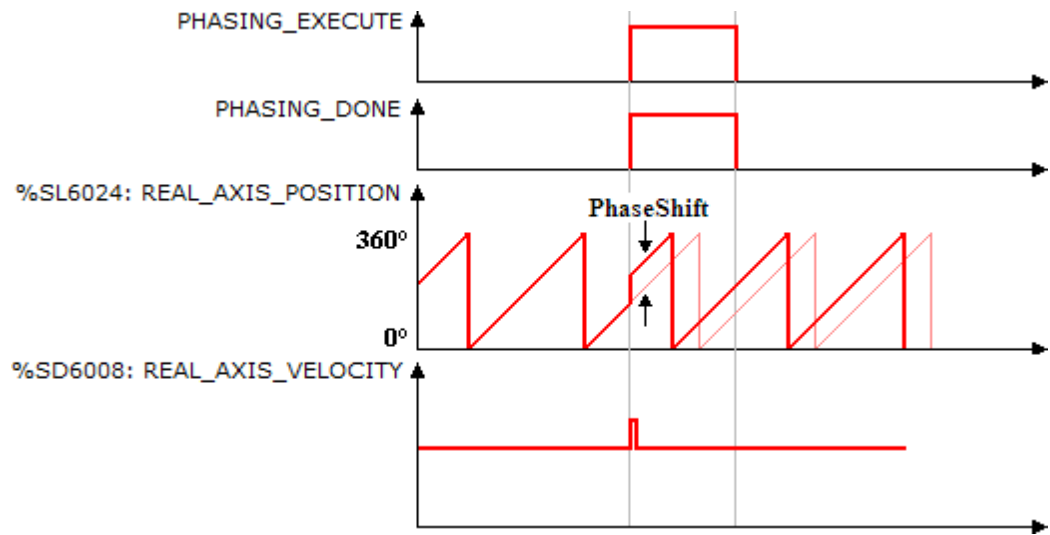
Código	Descrição
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
73	Drive não está no estado "Synchronized Motion".
78	Bloco MC não executado - Falha interna.
79	Eixo Mestre não está em sincronismo.
96	Bloco MC_PhasingRelative em execução.

Exemplo



LOCAL		LOCAL_RETAIN			
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário	
GEARINPOS_INST	0	MC_GearInPos			
GEARINPOS_EXECUTE	0	BOOL	0		
GEARINPOS_INSYNC	0	BOOL	0		
PHASING_INST	0	MC_PhasingRelative			
PHASING_EXECUTE	0	BOOL	0		
PHASING_DONE	0	BOOL	0		

GLOBAL LOCAL

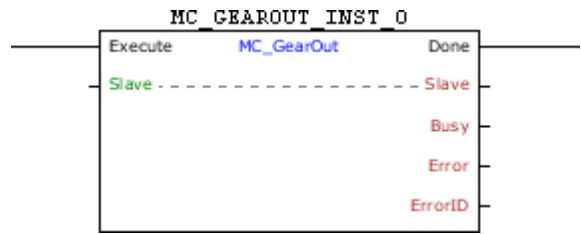


Tendo-se o sincronismo de posição do Eixo Real com o Contador Rápido através do bloco MC_GearInPos e ocorrendo uma transição de subida em PHASING_EXECUTE, o bloco MC_PhasingRelative é executado e um deslocamento de 0,05 volta é aplicado ao eixo mestre, resultando em um pulso na velocidade. A saída Done é setado enquanto a entrada Execute está setada.

10.3.5.1.24.4 MC_GearOut

Bloco responsável pela finalização de sincronismo entre os eixos programados.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	28 Bytes
Tamanho de Memória de Dados	4 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Slave	BYTE	Seleção de escravo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_GEAROUT_INST_0	MC_GEAROUT	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, finaliza o sincronismo dos blocos MC_GearIn ou MC_GearInPos no eixo programado. O eixo manterá a velocidade do instante em que o bloco é executado.

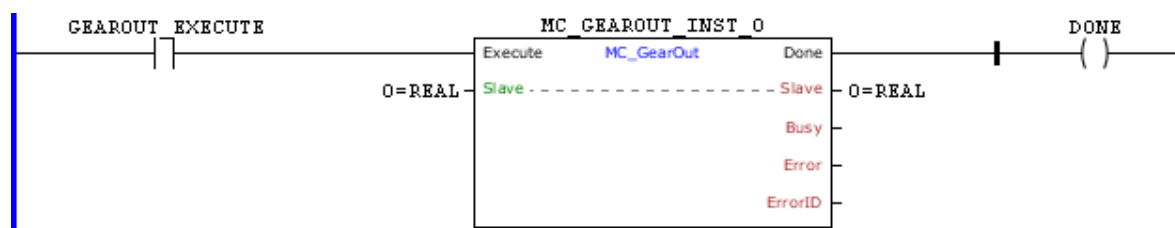
Ao executar o bloco MC_GearOut, o *drive* não opera em malha de posição. Na execução do bloco o estado do eixo mudará para ContinuousMotion.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
67	Drive no estado "Disabled" ou "ErrorStop".
71	P202 diferente de 4.
73	Drive não está no estado "Synchronized Motion".
78	Bloco MC não executado - Falha interna.

Exemplo



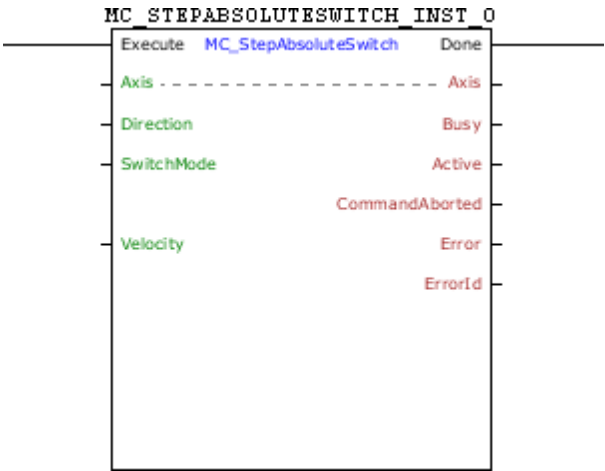
Na transição de subida de GEAROUT_EXECUTE, o bloco MC_GearOut é executado. Com isso, os sinais Busy e Active deste bloco são setados e finaliza-se o sincronismo iniciado por outros blocos de sincronismo entre mestre e escravo. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.3.5.1.25 Motion Control Homing

10.3.5.1.25.1 MC_StepAbsoluteSwitch

Bloco responsável pela busca da posição da AbsoluteSwitch.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	74 Bytes
Tamanho de Memória de Dados	40 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Direction	BYTE	Direção de busca (0 - Direção positiva) (1 - Direção negativa) (5 - MC_SwitchPositive: Direção positiva, se AbsoluteSwitch não acionada) (6 - MC_SwitchNegative: Direção negativa, se AbsoluteSwitch não acionada)
	SwitchMode	BYTE	Modo de busca (3 - MC_EdgeOn) (4 - MC_EdgeOff)
	Velocity	REAL	Velocidade
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STEPABSOLUTESWITCH_INST_0	MC_STEPABSOLUTESWITCH	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para busca da posição da AbsoluteSwitch.

A AbsoluteSwitch somente poderá ser ligada nas entradas digitais 1, 2 ou 3, sendo que a função programada da entrada digital deverá ser de acordo com o argumento SwitchMode. Se SwitchMode for configurado como MC_EdgeOn (borda de subida), a função da entrada digital (P0300, P0301 ou P0302) deverá ser "armazena posição - borda de subida" (opção 8). Se SwitchMode for configurado como MC_EdgeOff (borda de descida), a função da entrada digital (P0300, P0301 ou P0302) deverá ser "armazena posição - borda de descida" (opção 9). Será considerado AbsoluteSwitch a primeira entrada digital configurada conforme SwitchMode a partir da entrada digital 1. Caso nenhuma entrada digital esteja configurada conforme SwitchMode, ocorrerá o erro 77 no bloco e ele não será executado.

Se ao buscar a posição da AbsoluteSwitch e atingir a posição de LimitSwitch (fim de curso), o movimento mudará de sentido até a posição da AbsoluteSwitch.

A busca será executada com a velocidade configurada no argumento Velocity e uma aceleração/desaceleração configurado no Perfil Padrão (General Profile).

Com a execução do bloco MC_StepAbsoluteSwitch, a posição de referência do usuário (P0051, P0052 e P0053) não é alterada. O *drive* passará a operar em malha de posição e permanece assim

mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

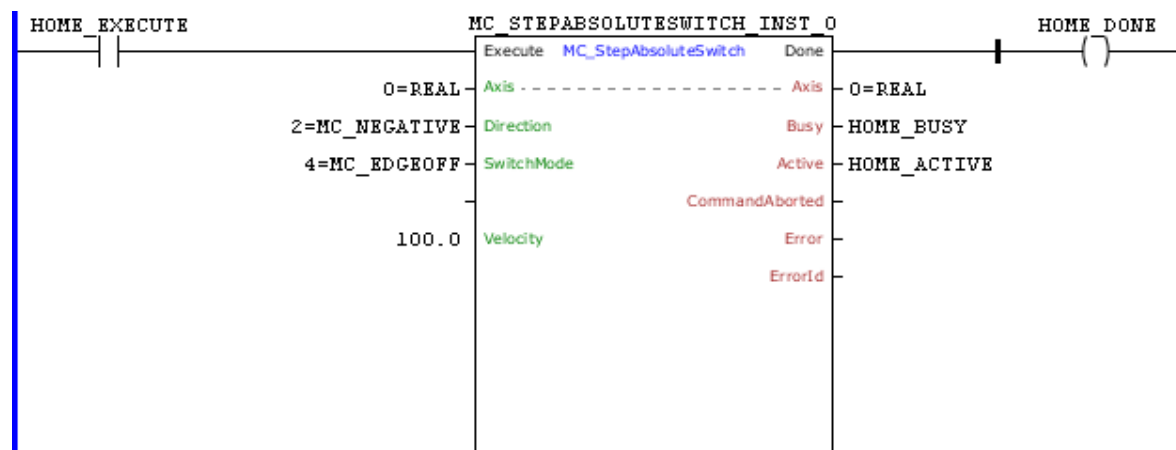
Na execução do bloco o estado do eixo mudará para Homing e permanecerá assim até a execução dos blocos MC_StepRefPulse, MC_StepDirect ou MC_FinishHoming.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

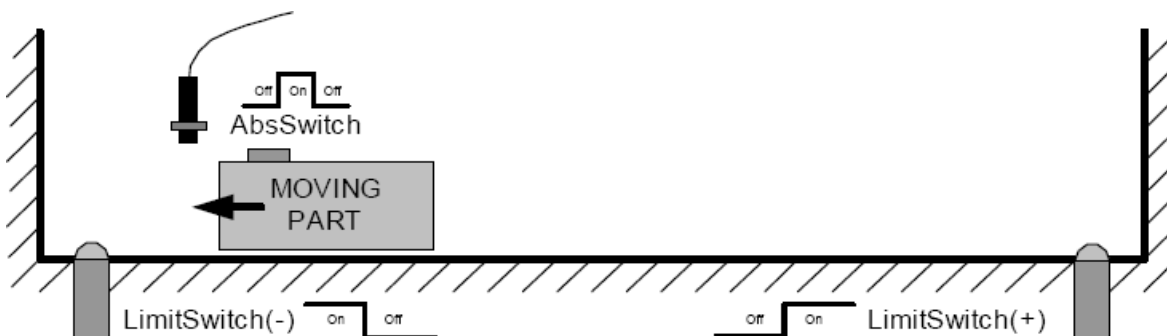
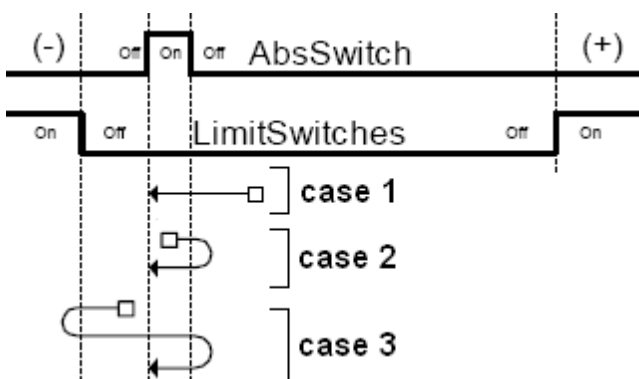
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".
77	Entradas digitais 1, 2 e 3 não configuradas conforme "SwitchMode".

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_STEPABSOLUTESWITCH_INST_0	0	MC_StepAbsoluteSwitch		
HOME_EXECUTE	0	BOOL	0	
HOME_BUSY	0	BOOL	0	
HOME_ACTIVE	0	BOOL	0	
HOME_DONE	0	BOOL	0	



Na transição de subida de HOME_EXECUTE, o bloco MC_StepAbsoluteSwitch é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se a busca da AbsoluteSwitch.

No caso 1, ao executar o bloco, a AbsoluteSwitch não está acionada. Como o argumento Direction está configurado como MC_SwitchNegative, o movimento será na direção negativa. Quando ocorrer uma borda de descida em AbsoluteSwitch (SwitchMode = MC_EdgeOff), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco a AbsoluteSwitch está acionada. Como o argumento Direction está configurado como MC_SwitchPositive, o movimento será na direção positiva e, ao sair da AbsoluteSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer uma borda de descida em AbsoluteSwitch (SwitchMode = MC_EdgeOff), o motor para e volta para a posição em que a borda ocorreu.

No caso 3, ao executar o bloco a AbsoluteSwitch não está acionada. Como o argumento Direction está configurado como MC_SwitchNegative, o movimento será na direção negativa. Mas ao encontrar a LimitSwitch (fim de curso) o motor para e muda o movimento para a direção positiva. Ao

sair da AbsoluteSwitch o motor para novamente e muda o movimento para a direção negativa. Quando ocorrer uma borda de descida em AbsoluteSwitch (SwitchMode = MC_EdgeOff), o motor para e volta para a posição em que a borda ocorreu.

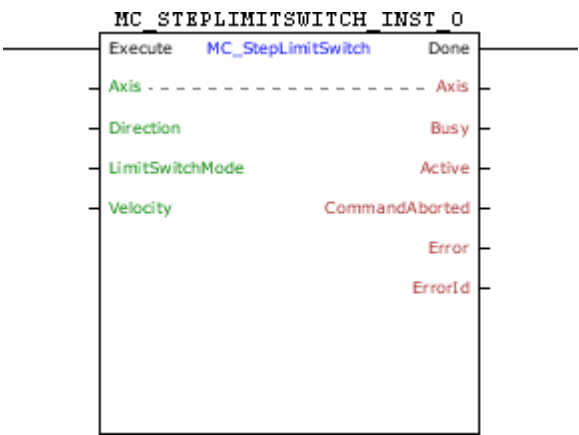
Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101, exceto ao encontrar a LimitSwitch (fim de curso), onde o motor para instantaneamente.

Ao voltar a posição de borda de descida da AbsoluteSwitch, a saída Done do bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

10.3.5.1.25.2 MC_StepLimitSwitch

Bloco responsável pela busca da posição da LimitSwitch.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	40 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Direction	BYTE	Direção de busca (0 - Direção positiva) (1 - Direção negativa) (5 - MC_SwitchPositive: Direção positiva, se AbsoluteSwitch não acionada) (6 - MC_SwitchNegative: Direção negativa, se AbsoluteSwitch não acionada)
	LimitSwitchMode	BYTE	Modo de busca (3 - MC_EdgeOn) (4 - MC_EdgeOff)
	Velocity	REAL	Velocidade
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STEPLIMITSWITCH_INST_0	MC_STEPLIMITSWITCH	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para busca da posição da LimitSwitch.

A LimitSwitch somente poderá ser ligada nas entradas digitais 1, 2 ou 3, sendo que a função programada da entrada digital deverá ser de acordo com o argumento "LimitSwitchMode" e o argumento "Direction". Será considerado LimitSwitch a primeira entrada digital configurada conforme a tabela, a partir da entrada digital 1. Caso nenhuma entrada digital esteja configurada conforme LimitSwitchMode e Direction, ocorrerá o erro 77 no bloco e ele não será executado.

A busca será executada com a velocidade configurada no argumento Velocity e uma aceleração/desaceleração configurado no Perfil Padrão (General Profile).

Com a execução do bloco MC_StepLimitSwitch, a posição de referência do usuário (P0051, P0052 e P0053) não é alterada. O *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

Na execução do bloco o estado do eixo mudará para Homing e permanecerá assim até a execução dos blocos MC_StepRefPulse, MC_StepDirect ou MC_FinishHoming.

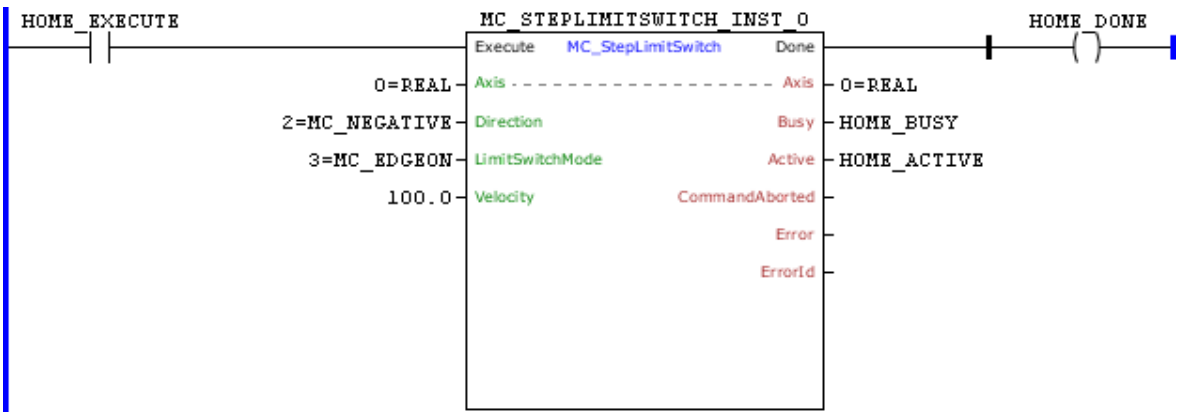
Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba

FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

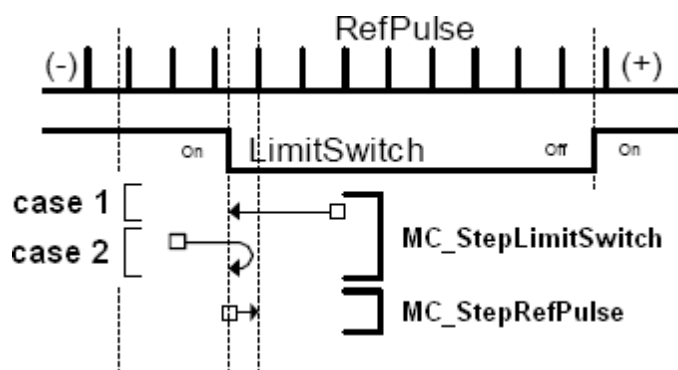
Código	Descrição
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".
77	Entradas digitais 1, 2 e 3 não configuradas conforme "Sw itchMode".

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_STEPLIMITSWITCH_INST_0	0	MC_StepLimitSwitch		
HOME_EXECUTE	0	BOOL	0	
HOME_BUSY	0	BOOL	0	
HOME_ACTIVE	0	BOOL	0	
HOME_DONE	0	BOOL	0	

GLOBAL LOCAL



Na transição de subida de HOME_EXECUTE, o bloco MC_StepLimitSwitch é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca da LimitSwitch.

No caso 1, ao executar o bloco, a LimitSwitch não está acionada. Como o argumento Direction está configurado como MC_Negative, o movimento será na direção negativa. Quando ocorrer uma borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco, a LimitSwitch está acionada. Mesmo com o argumento Direction configurado como MC_Negative, o movimento será na direção positiva e, ao sair da LimitSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer a borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

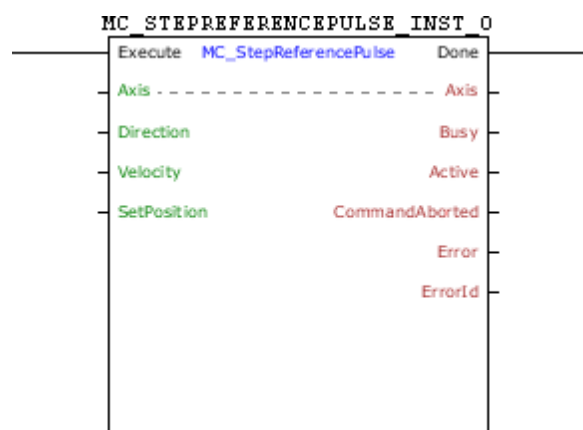
Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101, exceto ao encontrar a LimitSwitch (fim de curso), onde o motor para instantaneamente.

Ao voltar a posição de borda de subida da LimitSwitch, a saída Done do bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

10.3.5.1.25.3 MC_StepReferencePulse

Bloco responsável pela busca da posição do pulso nulo.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	40 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	Direction	BYTE	Direção de busca (0 - Direção positiva) (1 - Direção negativa) (5 - MC_SwitchPositive: Direção positiva, se AbsoluteSwitch não acionada) (6 - MC_SwitchNegative: Direção negativa, se AbsoluteSwitch não acionada)
	Velocity	REAL	Velocidade
	SetPosition	LREAL	Nova posição de referência do usuário
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_STEPREFERENCEPULSE_INST_0	MC_STEPREFERENCEPULSE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, envia um comando para busca da posição do pulso nulo.

A busca será executada com a velocidade configurada no argumento Velocity e uma aceleração/desaceleração configurado no Perfil Padrão (General Profile).

Com a execução do bloco MC_StepReferencePulse, a posição de referência do usuário (P0051, P0052 e P0053) é alterada para o valor do argumento SetPosition. O *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

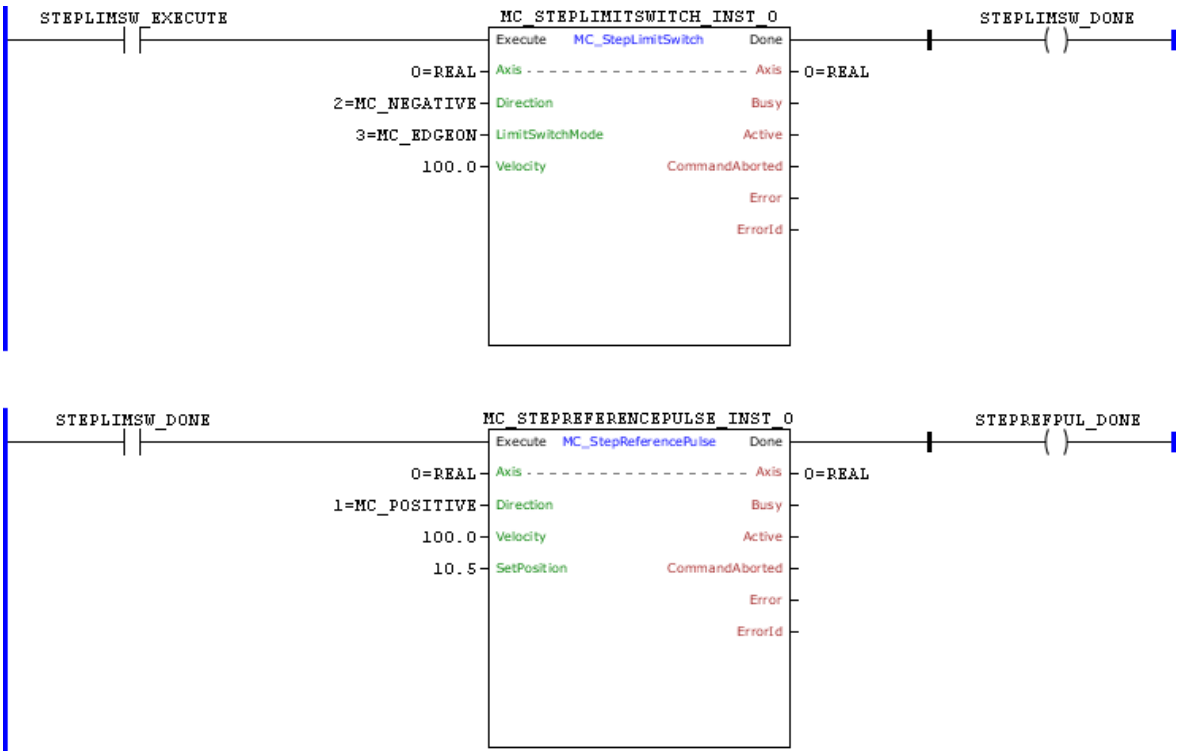
Na execução do bloco o estado do eixo mudará para Homing. Ao concluir a busca o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

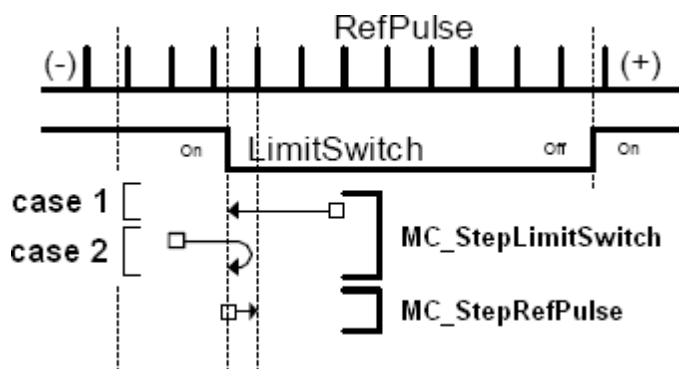
Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MC_STEPLIMITSWITCH_INST_0	0	MC_StepLimitSwitch		
STEPLIMSW_EXECUTE	0	BOOL	0	
STEPLIMSW_DONE	0	BOOL	0	
MC_STEPPREFERENCEPULSE_INST_0	0	MC_StepReferencePulse		
STEPPREFPUL_DONE	0	BOOL	0	



Na transição de subida de STEPLIMSW_EXECUTE, o bloco MC_StepLimitSwitch é executado. Com isso os sinais Busy e Active deste bloco são setados e inicia-se a busca da LimitSwitch.

No caso 1, ao executar o bloco, a LimitSwitch não está acionada. Como o argumento Direction está configurado como MC_Negative, o movimento será na direção negativa. Quando ocorrer uma borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco, a LimitSwitch está acionada. Mesmo com o argumento Direction configurado como MC_Negative, o movimento será na direção positiva e, ao sair da LimitSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer a borda de subida em LimitSwitch (LimitSwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101, exceto ao encontrar a LimitSwitch (fim de curso), onde o motor para instantaneamente.

Ao voltar a posição de borda de subida da LimitSwitch, a saída Done do bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

Na transição de subida de STEPLIMSW_DONE, o bloco MC_StepRefPulse é executado e iniciasse a busca do pulso nulo.

O movimento será na direção positiva e ao encontrar o pulso nulo, o motor para e volta para a posição do pulso nulo.

Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101.

Ao voltar a posição do pulso nulo, a saída Done do bloco é setada e permanece TRUE enquanto a

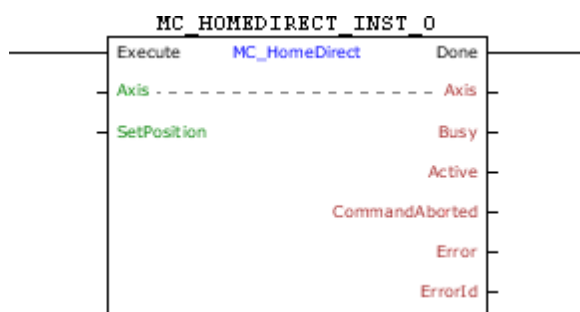
entrada Execute está setada. A posição de referência do usuário (P0051, P0052 e P0053) é alterada para 10,5 voltas (P0051 = 8192, P0052 = 10 e P0053 = 0).

Quando STEPLIMSW_EXECUTE é resetado, STEPLIMSW_DONE e STEPREFPUL_DONE também são resetados.

10.3.5.1.25.4 MC_HomeDirect

Bloco responsável pela mudança da posição de referência do usuário.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	46 Bytes
Tamanho de Memória de Dados	16 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	SetPosition	LREAL	Nova posição de referência do usuário
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_HOMEDIRECT_INST_0	MC_HOMEDIRECT	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, faz com que a posição de referência do usuário (P0051, P0052 e P0053) seja alterada para o valor do argumento SetPosition.

Na execução do bloco, se o estado do eixo é Homing, o estado do eixo mudará para Standstill. Caso

contrário, permanecerá no estado atual.

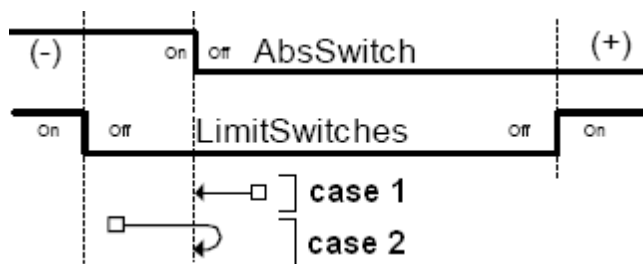
Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
71	P202 diferente de 4.
76	Estado do Drive diferente de "StandStill" ou "Homing".

Exemplo





Na transição de subida de STEPABSSW_EXECUTE, o bloco MC_StepAbsoluteSwitch é executado e inicia-se a busca da AbsoluteSwitch. O estado do eixo é alterado para Homing.

No caso 1, ao executar o bloco, a AbsoluteSwitch não está acionada. Como o argumento Direction está configurado como MC_SwitchNegative, o movimento será na direção negativa. Quando ocorrer uma borda de subida em AbsoluteSwitch (SwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

No caso 2, ao executar o bloco a AbsoluteSwitch está acionada. Como o argumento Direction está configurado como MC_SwitchPositive, o movimento será na direção positiva e, ao sair da AbsoluteSwitch, o motor para e muda o movimento para a direção negativa. Quando ocorrer uma borda de subida em AbsoluteSwitch (SwitchMode = MC_EdgeOn), o motor para e volta para a posição em que a borda ocorreu.

Todos os movimentos serão realizados com uma aceleração/desaceleração programados em P0100 e P0101.

Ao voltar a posição de borda de subida da AbsoluteSwitch, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

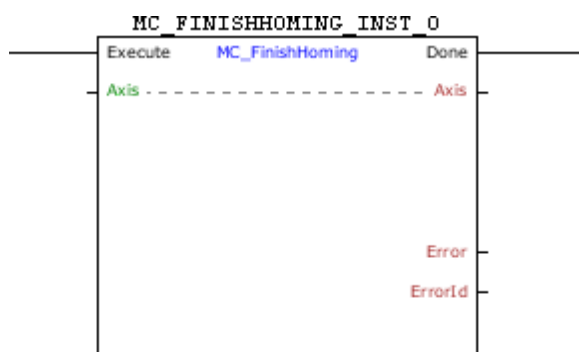
Na transição de subida de STEPABSSW_DONE, o bloco MC_StepDirect é executado e a posição de referência do usuário (P0051, P0052 e P0053) é alterada para 3,5 voltas (P0051 = 8192, P0052 = 3 e P0053 = 0). O estado do eixo é alterado para Standstill.

Quando STEPABSSW_EXECUTE é resetado, STEPABSSW_DONE e STEPDIR_DONE também são resetados.

10.3.5.1.25.5 MC_FinishHoming

Bloco responsável pela mudança do estado do eixo de Homing para Standstill.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	70 Bytes
Tamanho de Memória de Dados	8 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_HOMEDIRECT_INST_0	MC_HOMEDIRECT	Instância de acesso à estrutura do bloco

Funcionamento

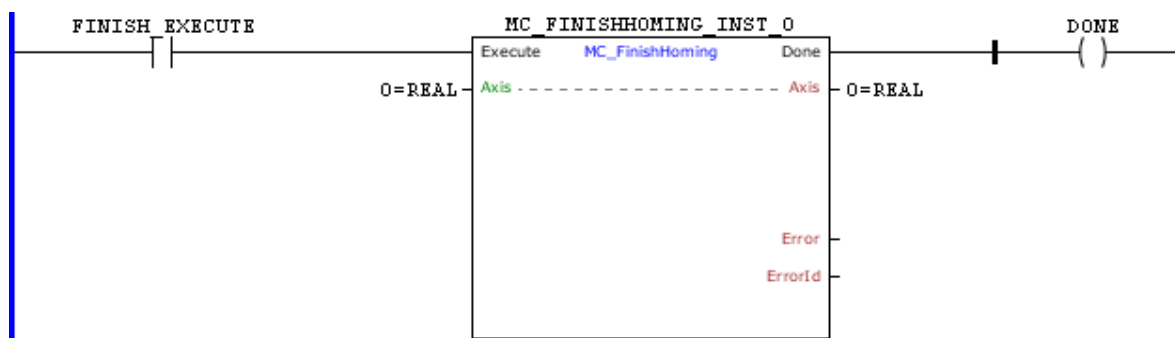
Este bloco, quando detecta uma borda de subida em Execute, se o estado do eixo é Homing, o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
71	P202 diferente de 4.
75	Estado do Drive diferente de "Homing".

Exemplo



Na transição de subida de FINISH_EXECUTE, o bloco MC_FinishHoming é executado. Com isso, os sinais Busy e Active deste bloco são setados e muda-se o estado do eixo selecionado para

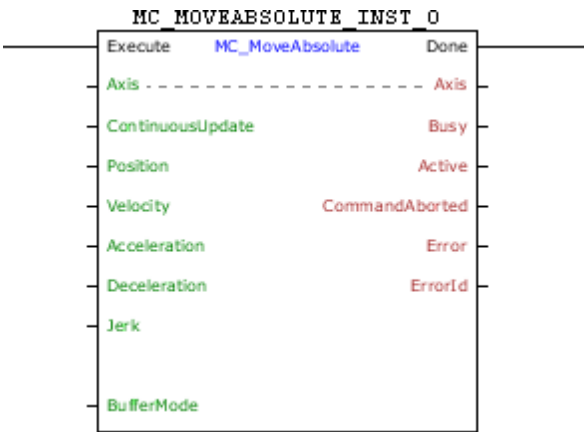
Standstill. Ao encerrar o processo, a saída Done do bloco é setada e permanece TRUE enquanto a entrada Execute está setada.

10.3.5.1.26 Motion Control Move

10.3.5.1.26.1 MC_MoveAbsolute

Bloco responsável por realizar um posicionamento para a posição absoluta programada.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Position	LREAL	Posição
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	Jerk	REAL	Jerk
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
	BufferMode	REAL	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR	MC_MOVEABSOLUTE_INST_0	MC_MOVEABSOLUTE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, executará um posicionamento para a posição absoluta configurada no argumento Position, com uma velocidade máxima configurada no argumento Velocity e uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. Dependendo da distância do posicionamento e dos valores de aceleração e desaceleração, a velocidade do motor não atingirá a velocidade máxima configurada.



NOTA!

Caso o valor do argumento Jerk for diferente de zero:

- 1) o valor da desaceleração será o mesmo do valor configurado na aceleração;
- 2) o argumento ContinuousUpdate On-line não terá efeito, sendo considerado os valores dos argumentos no instante da transição positiva de Execute;
- 3) não é permitido executar o posicionamento com outro bloco ativo, ocorrendo ErrorID 95.

Ao executar o bloco MC_MoveAbsolute, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

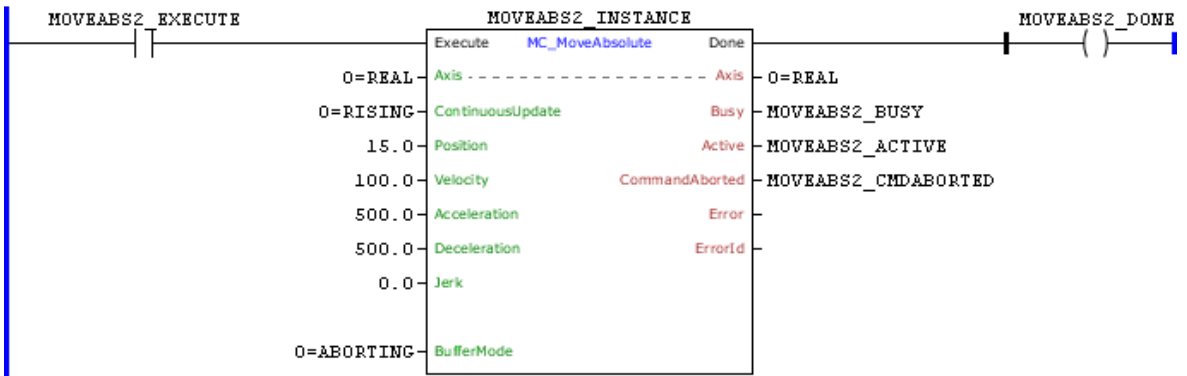
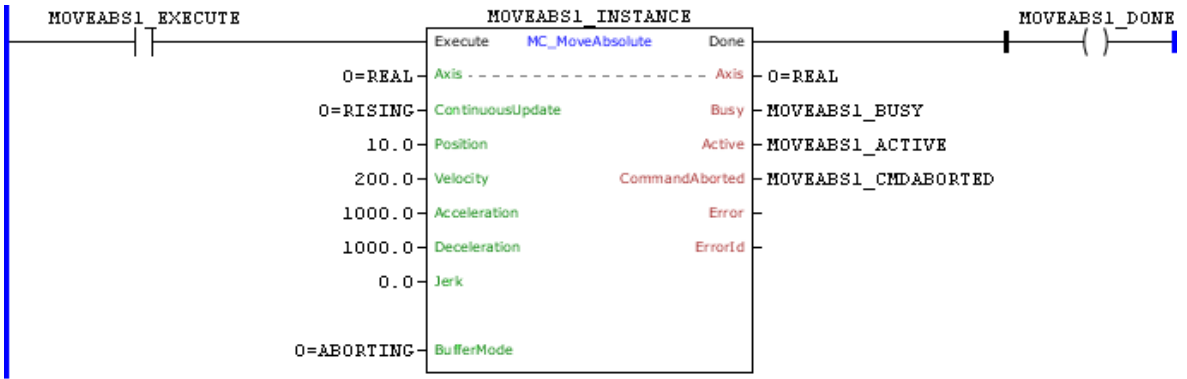
Na execução do posicionamento o estado do eixo mudará para Discrete Motion. Ao concluir o posicionamento o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

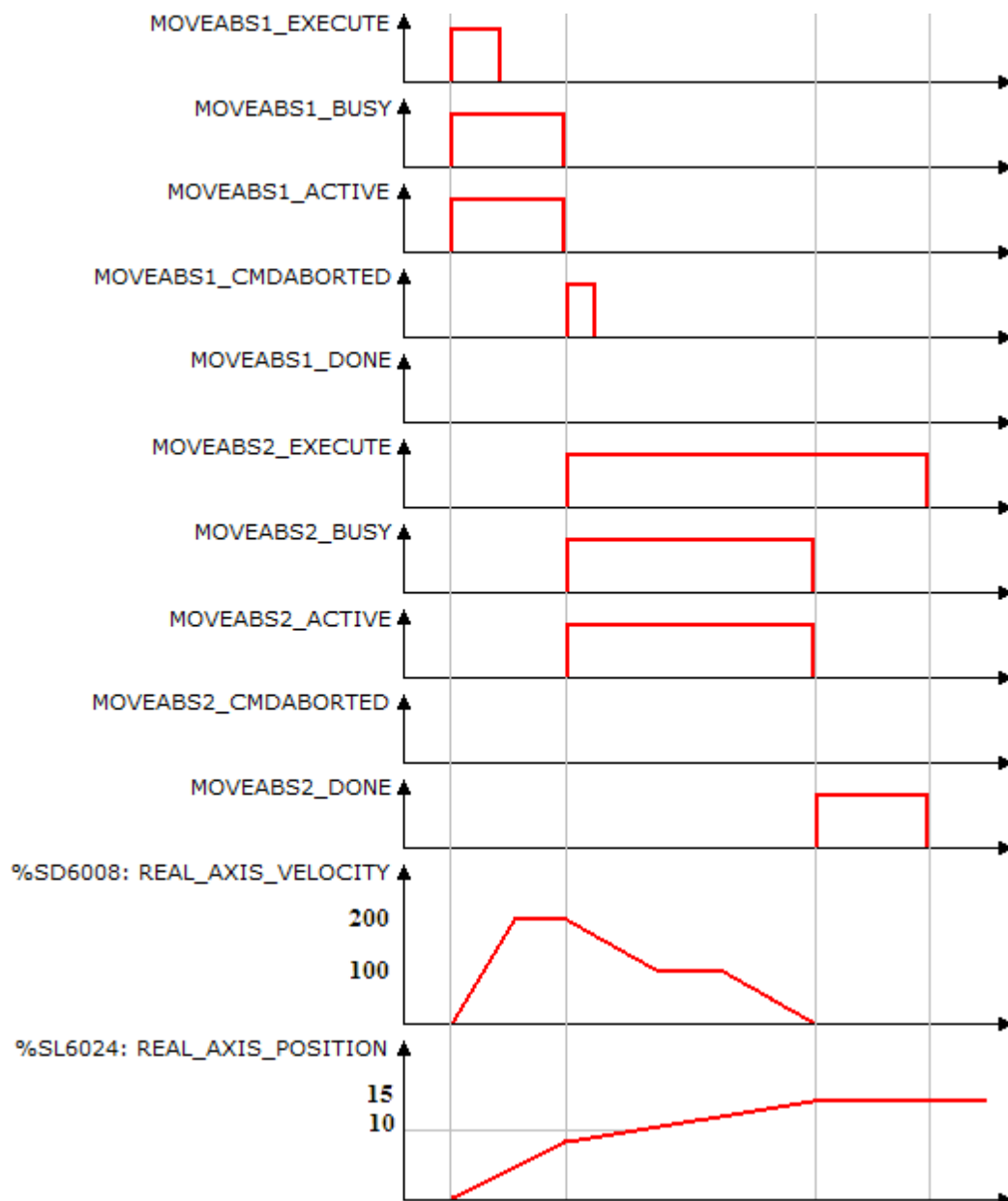
Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.
95	Não é permitido executar posicionamento com Jerk quando outro bloco ativo.

Exemplo



LOCAL		LOCAL_RETAIN			
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário	
MOVEABS1_INSTANCE	0	MC_MoveAbsolute			
MOVEABS2_INSTANCE	0	MC_MoveAbsolute			
MOVEABS1_EXECUTE	0	BOOL	0		
MOVEABS1_DONE	0	BOOL	0		
MOVEABS1_BUSY	0	BOOL	0		
MOVEABS1_ACTIVE	0	BOOL	0		
MOVEABS1_CMDABORTED	0	BOOL	0		
MOVEABS2_EXECUTE	0	BOOL	0		
MOVEABS2_DONE	0	BOOL	0		
MOVEABS2_BUSY	0	BOOL	0		
MOVEABS2_ACTIVE	0	BOOL	0		
MOVEABS2_CMDABORTED	0	BOOL	0		

GLOBAL LOCAL



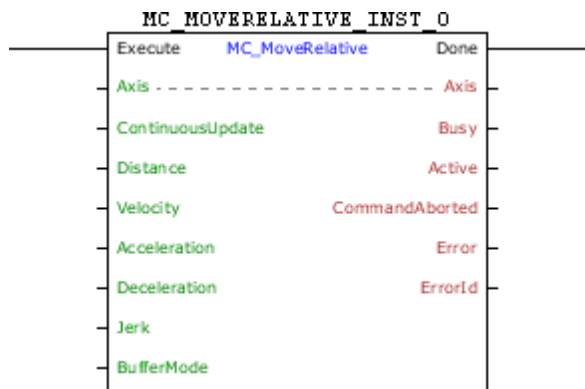
Na transição de subida de MOVEABS1_EXECUTE, o primeiro bloco MC_MoveAbsolute é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento para a posição absoluta "10 voltas".

Tendo-se a transição de subida de MOVEABS2_EXECUTE, o segundo bloco MC_MoveAbsolute é instantaneamente executado (BufferMode - Aborting). Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento para a posição 15 voltas. Ao mesmo tempo os sinais Busy e Active do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

Ao atingir a posição 15 voltas, a saída Done do segundo bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.

10.3.5.1.26.2 MC_MoveRelative

Bloco responsável por realizar um posicionamento para a posição relativa programada.

Representação Ladder**Características de Execução**

Tamanho de Memória de Programa	72 Bytes
Tamanho de Memória de Dados	32 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Distance	LREAL	Distância de deslocamento
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	Jerk	REAL	Jerk
	BufferMode	REAL	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	Done	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_MOVERELATIVE_INST_0	MC_MOVERELATIVE	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, executará um posicionamento de acordo com a distância relativa à posição atual, configurada no argumento Distance, com uma velocidade máxima configurada no argumento Velocity e uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. Dependendo da distância do posicionamento e dos valores de aceleração e desaceleração, a velocidade do motor não atingirá a velocidade máxima configurada.



NOTA!

Caso o valor do argumento Jerk for diferente de zero:

- 1) o valor da desaceleração será o mesmo do valor configurado na aceleração;
- 2) o argumento ContinuousUpdate On-line não terá efeito, sendo considerado os valores dos argumentos no instante da transição positiva de Execute;
- 3) não é permitido executar o posicionamento com outro bloco ativo, ocorrendo ErrorID 95.

Ao executar o bloco MC_MoveRelative, o *drive* passará a operar em malha de posição e permanece assim mesmo após a conclusão do bloco. Deve-se ajustar o ganho proporcional de posição (P0159) para obter um melhor desempenho do *drive*.

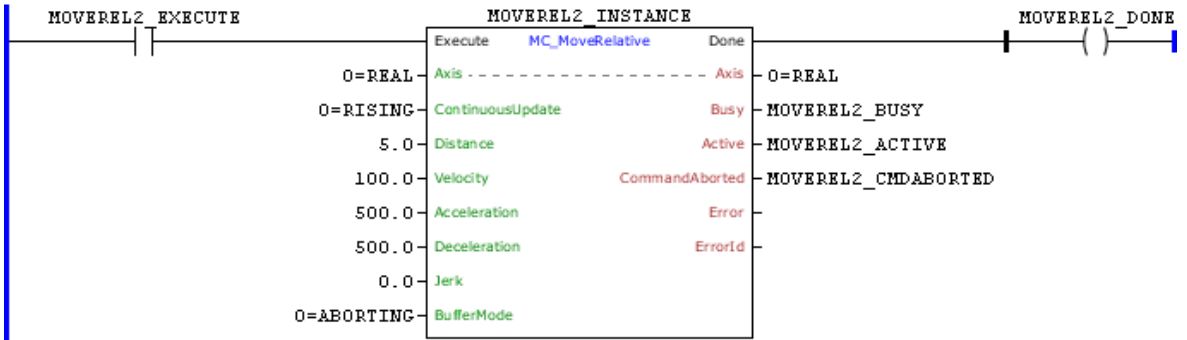
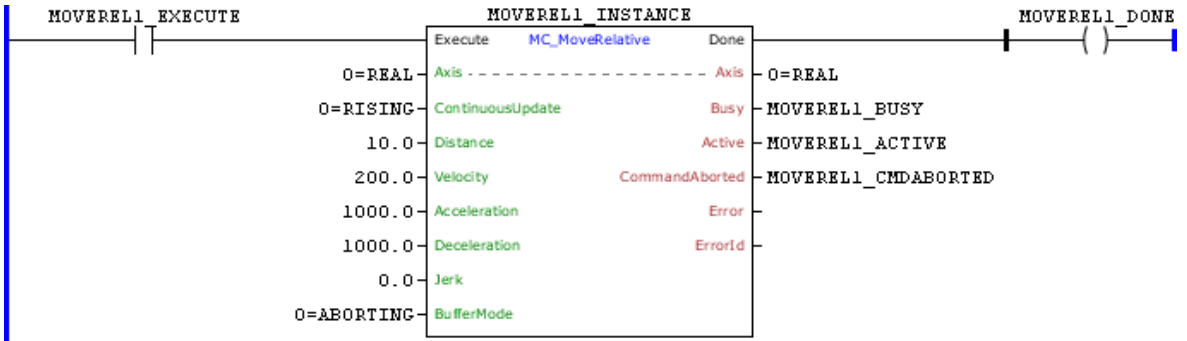
Na execução do posicionamento o estado do eixo mudará para Discrete Motion. Ao concluir o posicionamento o estado do eixo mudará para Standstill.

Quando Execute possui valor FALSE, Done permanece FALSE. A saída Done é ativada quando o bloco termina a execução com sucesso, permanecendo em nível TRUE até que Execute receba FALSE.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

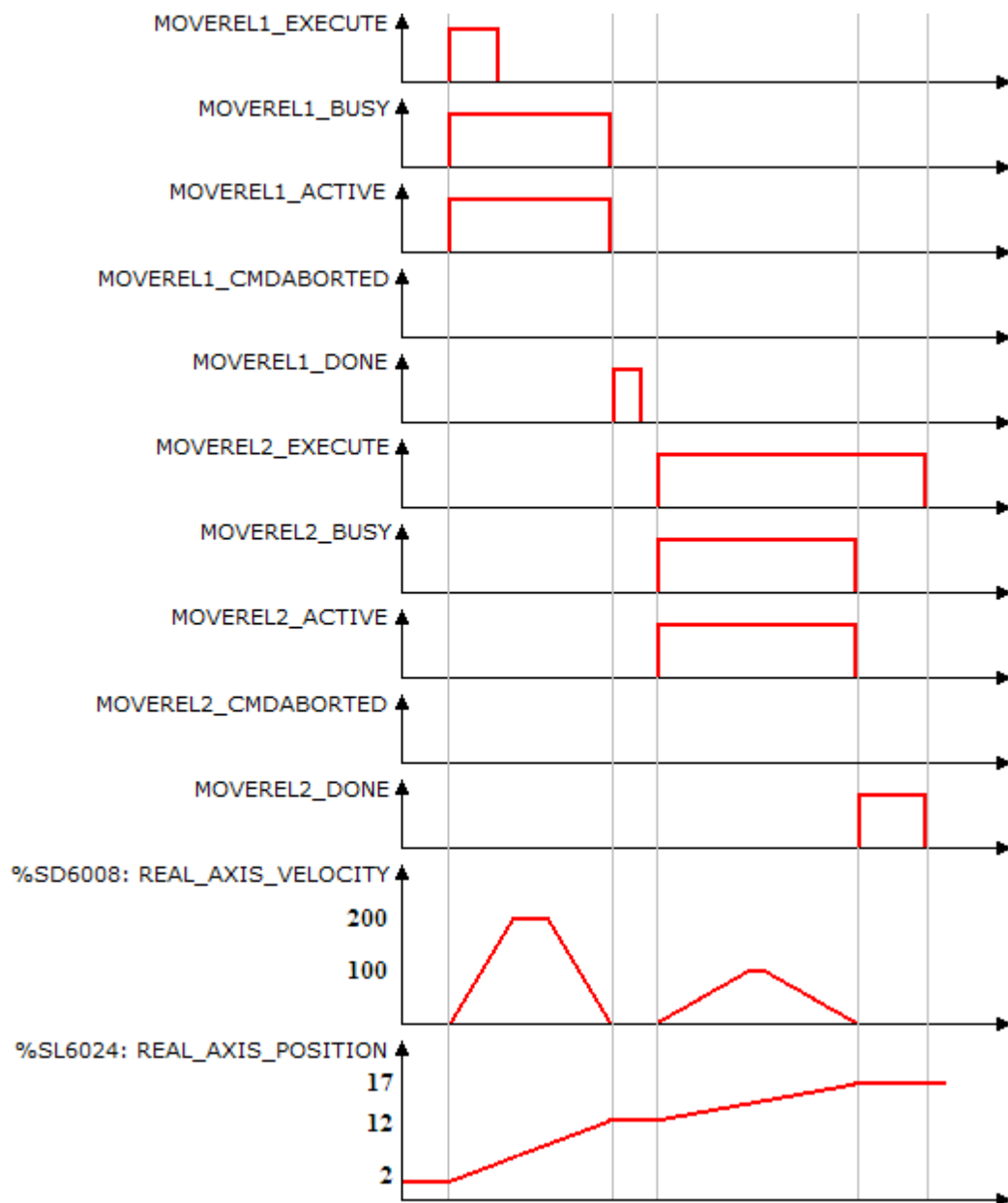
Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.
95	Não é permitido executar posicionamento com Jerk quando outro bloco ativo.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MOVEREL1_INSTANCE	0	MC_MoveRelative		
MOVEREL2_INSTANCE	0	MC_MoveRelative		
MOVEREL1_EXECUTE	0	BOOL	0	
MOVEREL1_DONE	0	BOOL	0	
MOVEREL1_BUSY	0	BOOL	0	
MOVEREL1_ACTIVE	0	BOOL	0	
MOVEREL1_CMDABORTED	0	BOOL	0	
MOVEREL2_EXECUTE	0	BOOL	0	
MOVEREL2_DONE	0	BOOL	0	
MOVEREL2_BUSY	0	BOOL	0	
MOVEREL2_ACTIVE	0	BOOL	0	
MOVEREL2_CMDABORTED	0	BOOL	0	

GLOBAL LOCAL

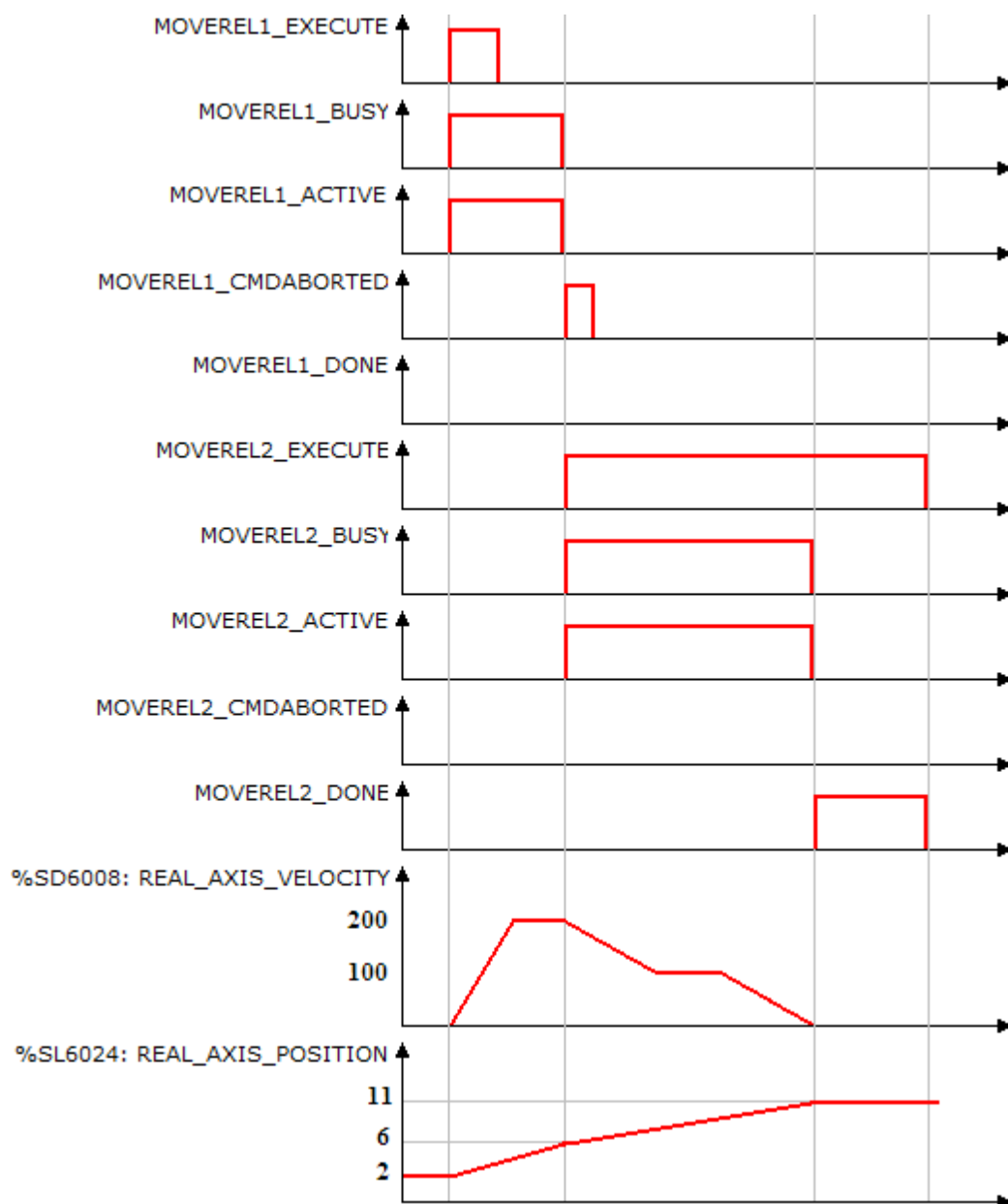


Na transição de subida de MOVEREL1_EXECUTE, o primeiro bloco MC_MoveRelative é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 10 voltas positivas a partir da posição atual.

Ao terminar o posicionamento, o primeiro bloco é concluído. Com isso, os sinais Busy e Active deste bloco são resetados e a saída Done é setada por 1 ciclo de scan.

Com uma transição de subida de MOVEREL2_EXECUTE, o segundo bloco MC_MoveRelative é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 5 voltas positivas a partir da posição atual.

Ao terminar o posicionamento, a saída Done do segundo bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está setada.



Na transição de subida de MOVEREL1_EXECUTE, o primeiro bloco MC_MoveRelative é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 10 voltas positivas a partir da posição atual.

Tendo-se a transição de subida de MOVEREL2_EXECUTE, o segundo bloco MC_MoveRelative é instantaneamente executado (BufferMode - Aborting). Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o posicionamento de 5 voltas positivas a partir da posição atual. Ao mesmo tempo, os sinais Busy e Active do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

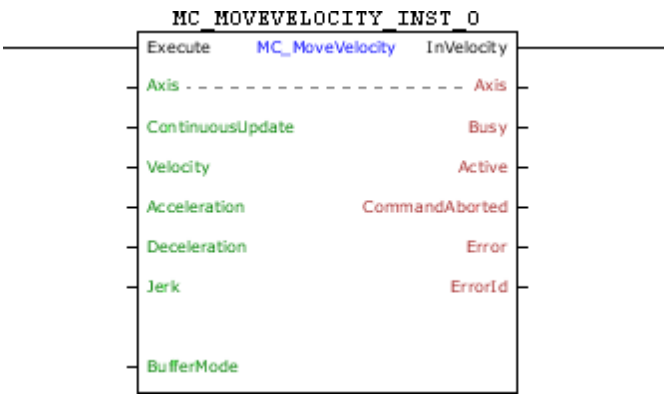
Ao terminar o posicionamento, a saída Done do segundo bloco é setada e os sinais Busy e Active deste bloco são resetados. A saída Done permanece TRUE enquanto a entrada Execute está

setada.

10.3.5.1.26.3 MC_MoveVelocity

Bloco responsável por realizar um movimento para a velocidade programada.

Representação Ladder



Características de Execução

Tamanho de Memória de Programa	66 Bytes
Tamanho de Memória de Dados	24 Bytes

Estrutura do Bloco

Tipo de Variável	Nome	Tipo de Dado	Descrição
VAR_IN_OUT	Axis	BYTE	Seleção de eixo da operação (0 - Eixo real)
VAR_INPUT	Execute	BOOL	Habilitação do bloco
	ContinuousUpdate	BYTE	Modo de atualização automática (0 - Borda de subida) (1 - Tempo real)
	Velocity	REAL	Velocidade
	Acceleration	REAL	Aceleração
	Deceleration	REAL	Desaceleração
	Jerk	REAL	Jerk
	BufferMode	REAL	Modo de início de execução (0 - Inicia bloco imediatamente, se houver outro bloco em execução o mesmo será abortado) (1 - Se outro bloco estiver em execução, o bloco em execução continuará seu movimento até concluí-lo e esse novo bloco aguardará para então ser executado.) (6 - Se algum outro bloco estiver em execução, este bloco entrará em erro 52 e não será executado. Será mostrado na IHM o alarme A00052.)
VAR_OUTPUT	InVelocity	BOOL	Habilitação da saída
	Busy	BOOL	Sinalizador de que o bloco ainda não foi finalizado
	Active	BOOL	Sinalizador de bloco com controle sobre o eixo
	CommandAborted	BOOL	Sinalizador de comando abortado
	Error	BOOL	Sinalizador de erro na execução
	ErrorID	WORD	Identificador do erro ocorrido
VAR	MC_MOVEVELOCITY_INST_0	MC_MOVEVELOCITY	Instância de acesso à estrutura do bloco

Funcionamento

Este bloco, quando detecta uma borda de subida em Execute, executará um movimento para a velocidade configurada no argumento Velocity com uma aceleração/desaceleração configurada nos argumentos Acceleration e Deceleration. A direção do movimento dependerá do sinal da velocidade: se a velocidade for maior que zero, o movimento será na direção positiva (sentido horário) e se a velocidade for menor que zero, o movimento será na direção negativa (sentido anti horário).



NOTA!

Caso o valor do argumento Jerk for diferente de zero, o argumento ContinuousUpdate On-line não terá efeito, sendo considerado os valores dos argumentos no instante da transição positiva de Execute.

Ao executar o bloco MC_MoveVelocity, o *drive* não opera em malha de posição.

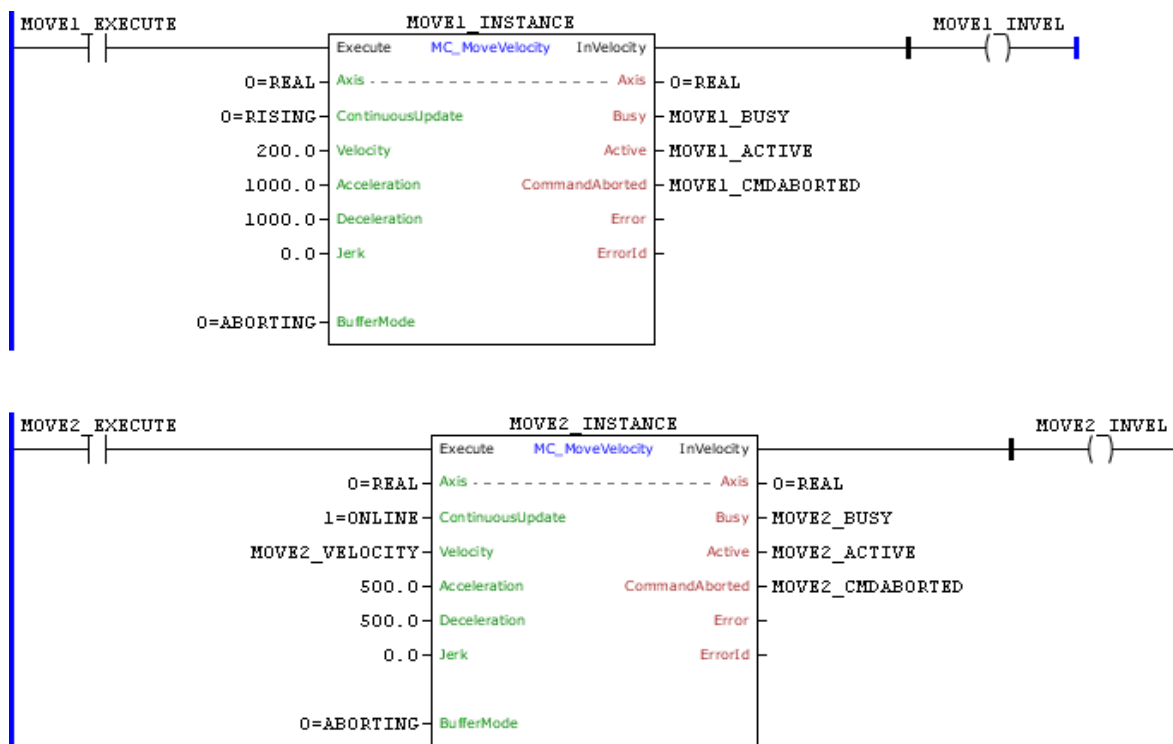
Na execução do movimento o estado do eixo mudará para Continuous Motion.

Quando Execute possui valor FALSE, InVelocity permanece FALSE. A saída InVelocity é ativada quando a velocidade programada é atingida com sucesso, permanecendo em nível TRUE enquanto o bloco estiver ativo. Para finalizar o bloco, é necessária a execução de outro bloco ou o *drive* passar para o estado Disabled ou Errorstop.

Caso haja algum erro na execução, a saída Error é ativada e ErrorID exibe o código do erro segundo a tabela abaixo.

Código	Descrição
52	Tentativa de executar bloco com BufferMode em Single quando outro bloco ativo.
60	Velocidade programada menor que a mínima permitida.
61	Velocidade programada maior que a máxima permitida.
62	Aceleração programada menor que a mínima permitida.
63	Aceleração programada maior que a máxima permitida.
64	Desaceleração programada menor que a mínima permitida.
65	Desaceleração programada maior que a máxima permitida.
67	Drive no estado "Disabled" ou "ErrorStop".
69	Drive no estado "Stopping".
70	Tentativa de executar bloco com BufferMode em Buffered quando outro bloco ativo e outro bloco aguardando.
71	P202 diferente de 4.
74	Drive no estado "Homing".
78	Bloco MC não executado - Falha interna.
93	Jerk programado menor que o mínimo permitido.
94	Jerk programado maior que o máximo permitido.
95	Não é permitido executar posicionamento com Jerk quando outro bloco ativo.

Exemplo



LOCAL LOCAL_RETAIN				
Tag	Tamanho	Tipo de Dado	Valor Inicial	Comentário
MOVE1_INSTANCE	0	MC_MoveVelocity		
MOVE1_EXECUTE	0	BOOL	0	
MOVE1_INVEL	0	BOOL	0	
MOVE1_BUSY	0	BOOL	0	
MOVE1_ACTIVE	0	BOOL	0	
MOVE1_CMDABORTED	0	BOOL	0	
MOVE2_INSTANCE	0	MC_MoveVelocity		
MOVE2_EXECUTE	0	BOOL	0	
MOVE2_INVEL	0	BOOL	0	
MOVE2_VELOCITY	0	REAL	200.0	
MOVE2_BUSY	0	BOOL	0	
MOVE2_ACTIVE	0	BOOL	0	
MOVE2_CMDABORTED	0	BOOL	0	

GLOBAL LOCAL



Na transição de subida de MOVE1_EXECUTE, o primeiro bloco MC_MoveVelocity é executado. Com isso, os sinais Busy e Active deste bloco são setados e inicia-se o movimento para chegar a velocidade de 200 RPM.

No instante em que a velocidade atinge 200 RPM, a saída InVelocity é setada.

Tendo-se a transição de subida de MOVE2_EXECUTE, o segundo bloco MC_MoveVelocity é instantaneamente executado (BufferMode - Aborting). Com isso os sinais Busy e Active deste bloco são setados e inicia-se o movimento para a velocidade de 100 RPM (neste instante MOVE2_VELOCITY contém o valor 100). Ao mesmo tempo os sinais Busy, Active e InVelocity do primeiro bloco são resetados e o sinal CommandAborted é setado por 1 ciclo de scan.

Ao atingir a velocidade de 100 RPM, a saída InVelocity do segundo bloco é setada e permanece até a execução de outro bloco.

Como o argumento ContinuousUpdate está configurado como On-line, com a mudança do valor de MOVE2_VELOCITY para 80, a velocidade muda imediatamente para 80 RPM, sem executar uma rampa de aceleração/desaceleração.

10.3.5.2 Estruturas

Uma estrutura consiste num agrupamento de dados utilizada para definir uma receita ou um objeto.

No programa Ladder é possível criar variáveis do tipo estrutura e utilizá-las nos blocos. Para acesso aos membros internos da estrutura, utiliza-se o '.' seguido do seu respectivo membro.

Criando uma Estrutura

1. Com o botão direito do mouse sobre a pasta **Estrutura**, clique em **Novo Arquivo**.

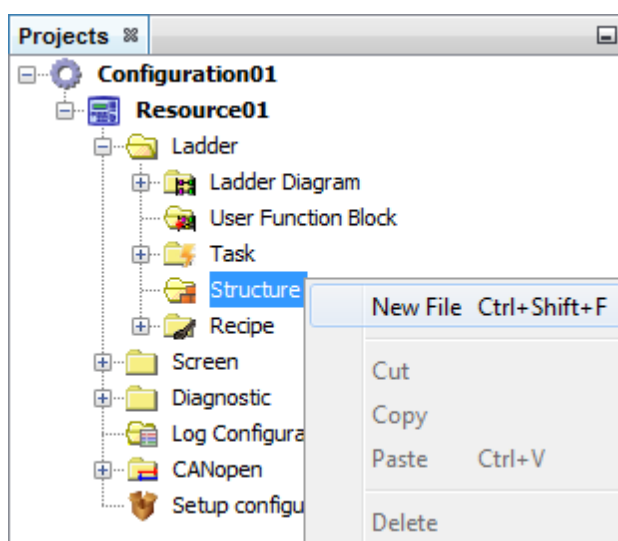


Figura 1 - Criando uma Estrutura

2. Defina o nome do arquivo e aperte o botão **Próximo**.

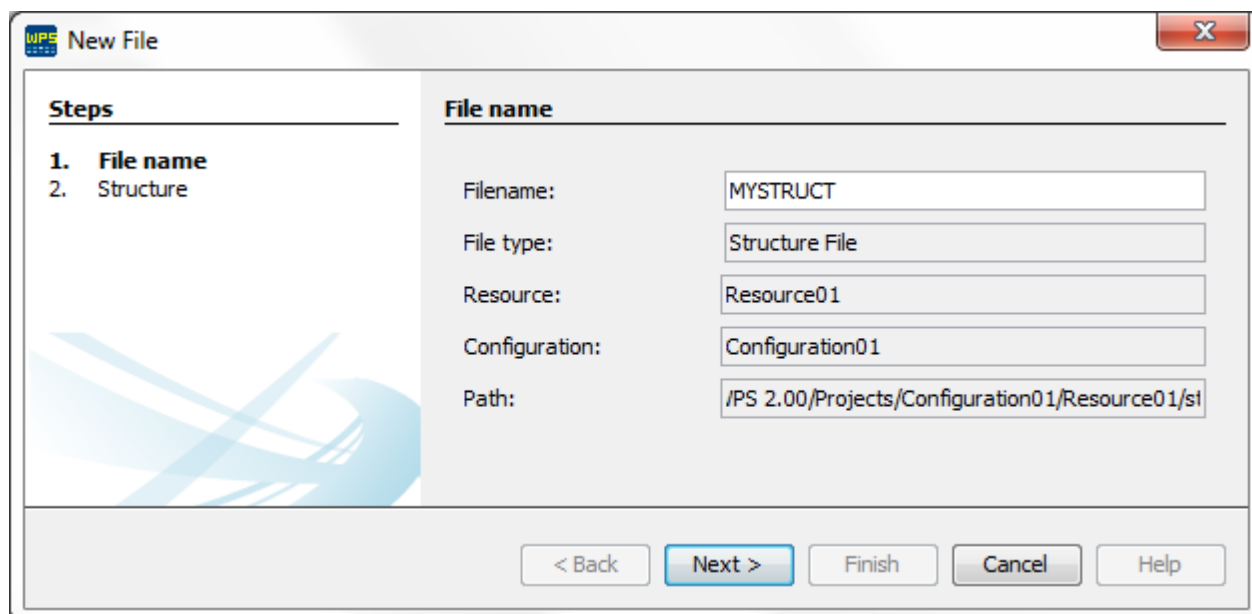


Figura 2 - Definindo o nome da Estrutura

3. Configure a estrutura utilizando os botões apresentados na figura abaixo.

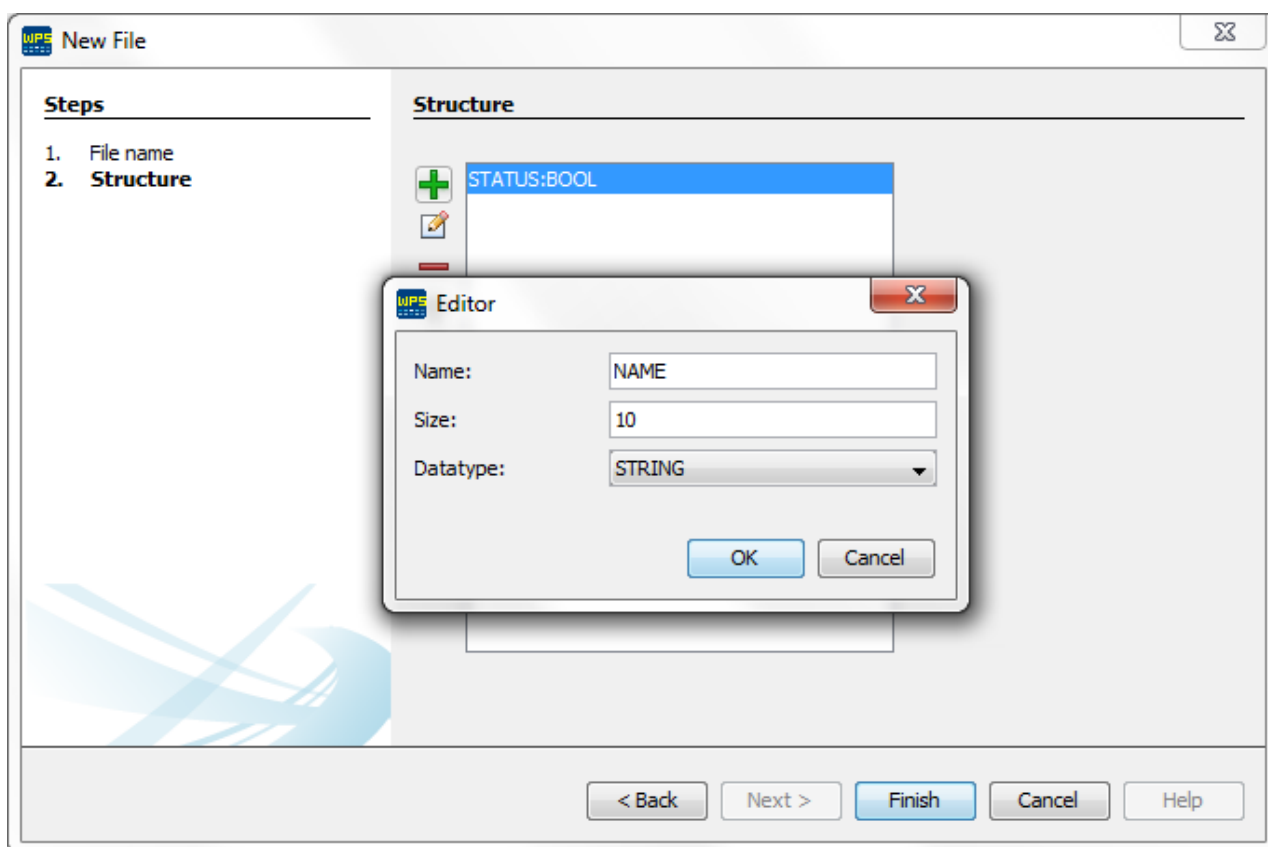


Figura 3 - Editando a Estrutura

4. Após concluir a edição da estrutura, clique no botão **Finalizar**.

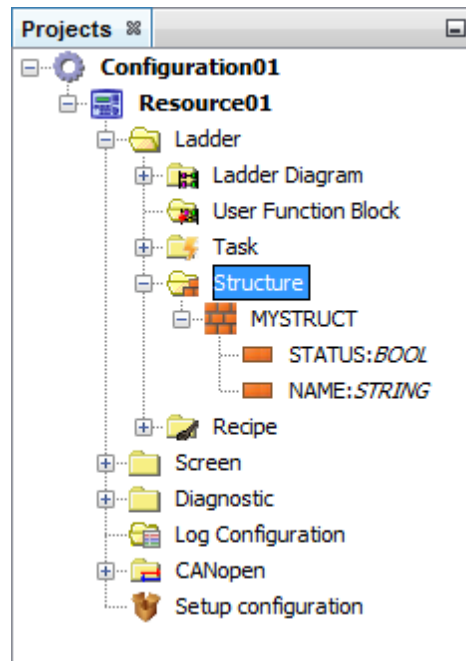


Figura 4 - Estrutura criada no projeto

Editando uma Estrutura

Para editar uma estrutura é necessário dar um duplo-clique na estrutura desejada. Uma janela abrirá, similar à da figura 3, permitindo inserir, apagar ou mover dados.

10.3.6 Diagnóstico

[Painel de Monitoração](#)

10.3.6.1 Painel de Monitoração

[Main Signals](#)

10.3.6.1.1 Main Signals

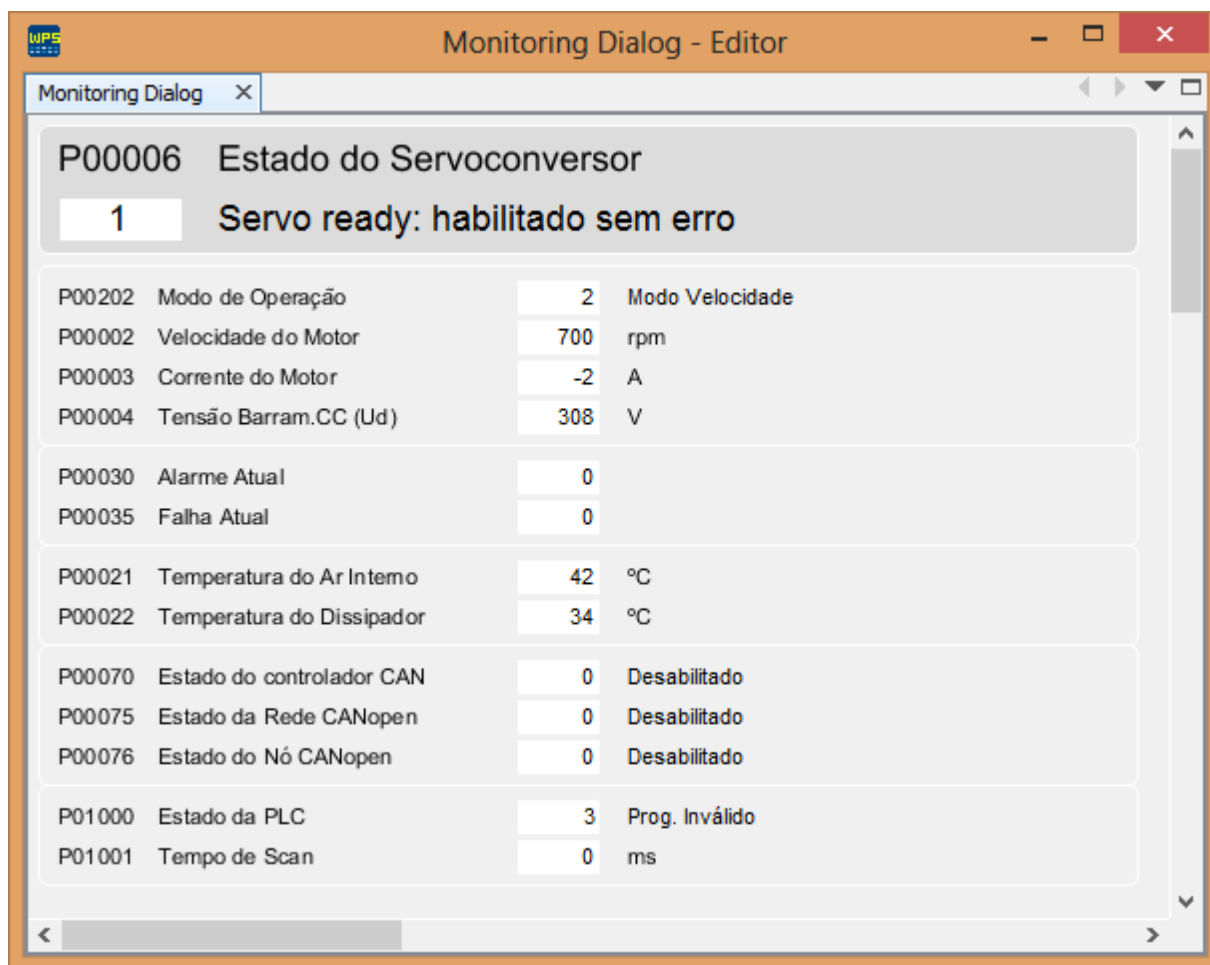
A janela Principais Sinais fornece uma visão geral dos principais sinais do equipamento:

- Estado do *drive* (P00006),
- Indicações gerais (P00202, P00002, P00003 e P00004),
- Alarme (P00030) e falha (P00035),
- Temperaturas (P00021 e P00022),
- Estados das redes (P00070, P00075 e P00076), e
- Informações do PLC (P01000 e P01001).

Os principais sinais são detalhados abaixo.

Parâmetro	Função	Descrição
P00006	Estado do Servoconversor	Indica o estado atual do servoconversor
P00202	Modo de Operação	Define o modo de operação do servoconversor, ou seja, qual a variável que se deseja controlar: Torque, Velocidade ou se o controle será feito via Ladder, CANopen ou Profibus
P00002	Velocidade do Motor	Indica o valor da velocidade real em rpm do servomotor, exceto quando for programado para receber realimentação de posição/velocidade externa
P00003	Corrente do Motor	Indica o valor da corrente Iq de saída, em amperes rms, do servoconversor
P00004	Tensão do Barramento CC	Indica a tensão atual no barramento CC em volts (V)
P00030	Alarme Atual	Indicam o número do alarme que eventualmente esteja presente no servoconversor
P00035	Falha Atual	Indicam o número da falha que eventualmente esteja presentes no servoconversor
P00021	Temperatura do ar interno	Este parâmetro apresenta, em graus Celsius, a temperatura do ar interno
P00022	Temperatura do dissipador	Este parâmetro apresenta, em graus Celsius, a temperatura do dissipador.
P00070	Estado do Controlador CAN	Indica o estado do controlador CAN, responsável por enviar e receber telegramas CAN
P00075	Estado da Rede CANopen	Indica o estado da comunicação CANopen, informando se o protocolo foi inicializado corretamente e o estado do serviço de guarda do escravo
P00076	Estado do Nó CANopen	Cada equipamento na rede CANopen possui um estado associado. É possível ver o estado atual do servoconversor através deste parâmetro
P01000	Estado do PLC	Permite ao usuário visualizar o status do programa
P01001	Tempo de Scan	Permite ao usuário monitorar o tempo do ciclo de varredura do programa em milissegundos

A janela pode ser visualizada abaixo.



10.3.7 Comunicação

[Forçar E/S](#)

10.3.7.1 Forçar E/S

Visão Geral

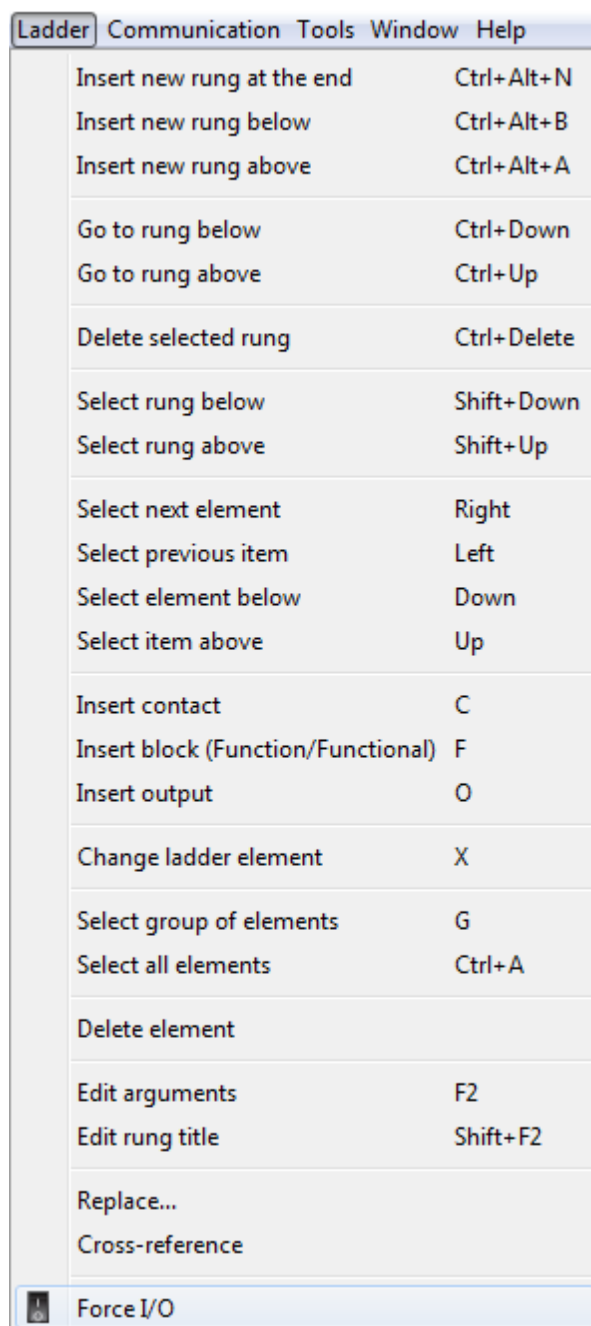
A janela de forçar entradas e saídas é utilizada para que os valores das entradas digitais e analógicas sejam lidos pelo programa, por valores manipulados pelo usuário, independente dos seus estados físicos. Também permite a manipulação dos estados físicos das saídas digitais e analógicas pelo usuário, de forma independente aos valores calculados pelo programa.

Para forçar as entradas e saídas do equipamento, é necessário que a monitoração on-line esteja ativa e a opção de **Rodar Ciclicamente** esteja habilitada. Os dados são enviados ao equipamento a cada 2 segundos.

Os valores podem ser editados sem que o equipamento esteja conectado. As configurações ficam armazenadas no recurso e são gravadas sempre que a seleção de recurso principal é alterada.

Os dados apresentados na janela de forçar E/S contêm os valores pertencentes ao recurso (e configuração) selecionado como principal.

A janela de forçar E/S é aberta através do menu **Ladder > Forçar E/S**:



Barra de ferramentas

A barra de ferramentas da janela de forçar possui as opções de rodar ciclicamente, *upload* da configuração de forçar do equipamento, habilitar todos e desabilitar todos:

 **Rodar ciclicamente:** Envia os as configurações do usuário para o equipamento e atualiza o estado das entradas e saídas de maneira cíclica.

 **Upload da configuração:** Permite que a configuração atual do equipamento seja lida. Para esta opção

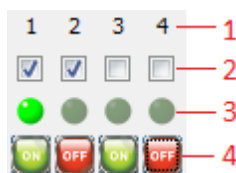
estar habilitada, é necessário que a monitoração on-line esteja ativa e a opção rodar ciclicamente desativada.

✓ **Habilitar todos:** Habilita o forçar E/S de todas as entradas e saídas do equipamento.

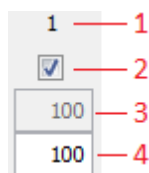
✗ **Desabilitar todos:** Desabilita o forçar E/S de todas as entradas e saídas do equipamento.

Comandos das entradas e saídas

Para cada entrada e saída digital e analógica estão associados uma caixa de seleção para habilitar o forçar, um campo de status e um campo para edição.



1. Número da entrada/saída digital
2. Habilita/desabilita o forçar E/S
3. Estado atual da E/S: Possui três estados: 1. led verde claro: acionado; 2. led verde escuro: desacionado; 3. led cinza: o valor não está sendo lido.
4. Habilita/desabilita a entrada/saída



1. Número da entrada/saída analógica
2. Habilita/desabilita o forçar E/S
3. Valor atual da entrada/saída
4. Valor da entrada/saída configurada pelo usuário

Índice

- A -

At; Representação Direta 123

- B -

Bloco Funcional; Function Block 123

- C -

comandos online 525, 545, 548

configuração 518, 519, 520, 521, 522, 523, 524

Configuração; Config; Configuration 123

- D -

download 530

- F -

forçar E/S 528, 881, 1186

Função; Bloco de Função 123

- M -

MC_CamTableSelect 787, 789, 794, 799, 1113, 1115, 1120, 1125

MC_Power 801, 1127

MC_Reset 804, 806, 810, 812, 817, 820, 823, 825, 829, 832, 836, 838, 840, 845, 851, 1130, 1132, 1136, 1138, 1143, 1146, 1149, 1151, 1155, 1158, 1162, 1164, 1166, 1171, 1177

- P -

Programa; Programa; Aplicativo 123

- R -

Recurso; Resource; Equipamento 124

- S -

Seleciona Tabela Cam 787, 789, 794, 799, 1113, 1115, 1120, 1125

setup 518, 519, 520, 521, 522, 523, 524

- T -

Tarefa; Task 124

Tipo de Dado; Data Type 124

- U -

upload 540

- V -

Variável 124

- W -

Watchdog; Estouro 124

WPS v2.3X 17