

# Software de programação

## Unity Pro



**Schneider**  
 **Electric**

---

Este manual não pode ser reproduzido, total ou parcialmente, sem autorização por escrito da **SCHNEIDER ELECTRIC**.

---

Seu conteúdo tem caráter exclusivamente técnico/informativo e a **SCHNEIDER ELECTRIC** se reserva no direito, sem qualquer aviso prévio, de alterar as informações deste documento.

---

## • **Serviço de Suporte SCHNEIDER ELECTRIC**

---

A **SCHNEIDER ELECTRIC** conta com um grupo de técnicos e engenheiros especializados aptos para fornecer informações e posicionamentos comerciais, esclarecer dúvidas técnicas, facilitar e garantir serviços técnicos com qualidade, rapidez e segurança..

Com o objetivo de criar um canal de comunicação entre a **SCHNEIDER ELECTRIC** e seus usuários, criamos um serviço denominado **AssisT**. Este serviço centraliza as eventuais dúvidas e sugestões, visando a excelência dos produtos e serviços comercializados pela **SCHNEIDER ELECTRIC**.

Este serviço está permanentemente disponível com uma cobertura horária das 7h30m às 18h, com informações sobre plantão de atendimento técnico durante os fins de semana e feriados, tudo que você precisa fazer é ligar para 0800 7289 110. O AssisT apresentará rapidamente a melhor solução, valorizando o seu precioso tempo.



Sua porta de entrada para o novo mundo elétrico

Para contato com a **SCHNEIDER ELECTRIC** utilize o endereço e telefones mostrados atrás deste Manual.

# CONVENÇÕES UTILIZADAS

- Títulos de capítulos estão destacados no índice e aparecem no cabeçalho das páginas.
- Palavras em outras línguas são apresentadas em *italico*, porém algumas palavras são empregadas livremente por causa de sua generalidade e freqüência de uso. Como, por exemplo, às palavras software e hardware.
- Números seguidos da letra h subscrita (ex: 1024<sub>h</sub>) indicam numeração hexadecimal e seguidos da letra b (ex: 10<sub>b</sub>), binário. Qualquer outra numeração presente deve ser interpretada em decimal.
- O destaque de algumas informações é dado através de ícones localizados sempre à esquerda da página. Cada um destes ícones caracteriza um tipo de informação diferente, sendo alguns considerados somente com caráter informativo e outros de extrema importância e cuidado. Eles estão identificados mais abaixo:



**NOTA:** De caráter informativo, mostra dicas de utilização e/ou configuração possíveis, ou ressalta alguma informação relevante no equipamento;



**OBSERVAÇÃO:** De caráter informativo, mostra alguns pontos importantes no comportamento/utilização ou configuração do equipamento. Ressalta tópicos necessários para a correta abrangência do conteúdo deste manual;



**IMPORTANTE:** De caráter informativo, mostrando pontos e trechos importantes do manual. Sempre observe e analise bem o conteúdo das informações que são identificadas por este ícone;



**ATENÇÃO:** Este ícone identifica tópicos que devem ser lidos com extrema atenção, pois afetam no correto funcionamento do equipamento em questão, podendo até causar danos à máquina/processo, ou mesmo ao operador, se não forem observados e obedecidos.

# Índice

---

<b>CAPÍTULO 1 .....</b>	<b>9</b>
<b>.PLATAFORMA PREMIUM.....</b>	<b>9</b>
<b>Plataforma Premium.....</b>	<b>11</b>
Introdução.....	11
Capacidade de expansão.....	12
Expansão de racks - Bus X.....	12
Procedimento para configuração dos racks de expansão.....	13
Bus X (Extensão de Racks) .....	14
Processadores.....	15
Sistema operacional Premium – Unity Pro.....	19
Comunicação.....	20
Ethernet TCP/IP .....	20
Fipio .....	22
CANopen.....	22
Interbus-S (geração 4) .....	22
Profibus DP .....	23
AS-i .....	23
Normas e certificados.....	23
<b>CAPÍTULO 2 .....</b>	<b>25</b>
<b>.PLATAFORMA QUANTUM.....</b>	<b>25</b>
<b>Plataforma Quantum .....</b>	<b>27</b>
Introdução.....	27
Bastidores.....	28
Arquitetura Flexível.....	29
Expansão de Bastidor - Módulo Expansor (XBE).....	30
Características do bus RIO .....	32
Processadores Quantum .....	34
Processadores Modelo Standard (Low End CPU).....	35
Processadores Alta Performance (High End CPU) .....	37
Conectividade Modicon Quantum .....	38
<b>CAPÍTULO 3 .....</b>	<b>39</b>
<b>.PLATAFORMA M340.....</b>	<b>39</b>
<b>Plataforma M340 .....</b>	<b>41</b>
Introdução.....	41
Bastidores “racks” Modicon M340.....	42
Multi-rack.....	43
Alimentação CA/CC.....	45
Unidade Central de Processamento BMX P34 xxxx .....	46
<b>CAPÍTULO 4 .....</b>	<b>51</b>
<b>.APRESENTAÇÃO E DESCRIÇÃO DO SOFTWARE UNITY.....</b>	<b>51</b>
<b>Apresentação e Descrição do Software Unity .....</b>	<b>53</b>
Unity Pro .....	53
Características Gerais do Unity Pro .....	55
Introdução .....	55
Guia de seleção .....	56
Unity Loader.....	56
Funcionalidades .....	57
Biblioteca de funções .....	58
Ferramentas de configuração .....	58
Simulador .....	59
<b>CAPÍTULO 5 .....</b>	<b>61</b>
<b>.AJUSTES DO PROJETO.....</b>	<b>61</b>
<b>Ajustes do Projeto .....</b>	<b>63</b>
Visão Geral da Interface .....	63
Iniciando um novo Projeto .....	65
Ajustes da estação de trabalho - Options .....	66
Project Settings .....	69
Project browser .....	71

<b>CAPÍTULO 6 .....</b>	<b>73</b>
<b>.CONFIGURAÇÃO DE HARDWARE.....</b>	<b>73</b>
Configuração de Hardware .....	75
Escolha o Processador .....	75
Editor de Configuração .....	75
Configuração do Rack.....	76
Configuração do Processador Premium .....	76
Configuração do Processador Quantum .....	76
Configuração do Processador Modicon M340 .....	77
Substituição do Processador.....	77
Configuração de Módulos .....	78
Endereçamento do CLP Quantum .....	83
Exemplos de endereçamento.....	84
Configuração da Rede.....	95
<b>CAPÍTULO 7 .....</b>	<b>97</b>
<b>.VARIÁVEIS.....</b>	<b>97</b>
Variáveis.....	99
Introdução .....	99
Tipos de Variáveis .....	99
Tipos e Faixa de dados comuns .....	99
Variáveis booleanas e tipos de dados permitidos:.....	100
Variáveis de sistema: .....	100
Regras para Entrada de valores Literais .....	100
Regras para Nomeação de Variáveis .....	100
Endereçamento Direto de variáveis.....	101
Acesso as variáveis .....	101
Edição de Variável .....	102
Filtro de variáveis.....	102
Propriedades dos Dados .....	103
Editor do tipo de dado – Data Type Editor.....	104
Criando Variáveis Elementares .....	104
Criando variáveis elementares a partir da pasta "Elementary Variables" .....	104
Criando variáveis elementares a partir da pasta "Configurations" .....	105
Criando variáveis elementares durante a edição do programa.....	110
<b>CAPÍTULO 8 .....</b>	<b>113</b>
<b>.GERENCIAMENTO DA BIBLIOTECA DE FUNÇÕES.....</b>	<b>113</b>
Gerenciamento da Biblioteca de Funções .....	115
Visão Geral .....	115
Gerenciamento do tipo de biblioteca .....	115
Bibliotecas Fornecidas na biblioteca global "Libset" .....	115
Expansões do Bowser Libset.....	116
Transferência para uma biblioteca local .....	117
Transferência para a biblioteca global .....	117
Acesso a biblioteca .....	118
Tipos de Browser de biblioteca.....	118
Armazenar um "DDT" ou um "DFB" em uma biblioteca .....	119
Gerenciamento de Versão.....	120
<b>CAPÍTULO 9 .....</b>	<b>121</b>
<b>.LINGUAGENS DE PROGRAMAÇÃO.....</b>	<b>121</b>
Linguagem de Programação Ladder .....	123
Introdução .....	123
Seqüência de execução em uma Networks/Rung .....	123
Configuração do editor Ladder.....	124
Criação de uma seção de Diagrama Ladder.....	125
Edição do Ladder .....	128
Diagrama de Blocos Funcionais – FBD.....	168
Introdução .....	168
Criando uma Seção FBD .....	168
Inserção de Bloco de Função a partir da barra de ferramentas.....	172
Inserção de FFBs a partir do "Types Library Browser" .....	173
Inserção de Bloco de Função a partir do "FFB input assistant" .....	176
Seqüência de execução de objetos em FBD .....	176
Seqüência de execução em uma rede lógica "Networks" .....	177
Depuração - Debugging .....	181

<b>CAPÍTULO 10 .....</b>	<b>183</b>
<b>.TESTE DE APLICAÇÃO.....</b>	<b>183</b>
<b>Teste da Aplicação.....</b>	<b>185</b>
Análise e Compilação do projeto.....	185
Conectando ao CLP .....	186
Conectando ao Simulador.....	187
Carregando uma aplicação no simulador: .....	188
Simulação do funcionamento do projeto.....	189
Simulação de entradas digitais e analógicas.....	190
Exemplo de aplicação.....	191
<b>CAPÍTULO 11 .....</b>	<b>195</b>
<b>.FUNÇÕES DE DIAGNÓSTICOS .....</b>	<b>195</b>
<b>Funções de Diagnóstico .....</b>	<b>197</b>
Introdução.....	197
Animação Dinâmica.....	197
Caixas de Display.....	197
Tabela Animada .....	198
Diagnóstico.....	200
Bits e Words de sistema.....	204
<b>CAPÍTULO 12 .....</b>	<b>207</b>
<b>.TIPO DE DADOS DERIVADOS - DDT.....</b>	<b>207</b>
<b>Tipo de Dados Derivados - DDT .....</b>	<b>209</b>
Tipo de Dados Derivados - Derived Data Type- DDT .....	209
Variáveis Tipos DDT.....	209
Tipo de Dados Derivados de Entradas/Saídas - I/ODDT:.....	212
<b>CAPÍTULO 13 .....</b>	<b>217</b>
<b>.BLOCO DE FUNÇÃO DERIVADO.....</b>	<b>217</b>
<b>Bloco de Função Derivado .....</b>	<b>219</b>
Introdução.....	219
Características do DFB. ....	220
Proteção do DFB .....	224
Armazenamento do DFB .....	227
Aplicação De DFB Para Controle De Partida Direta De Motor De Indução Trifásico.....	229
Depuração do DFB .....	236
<b>CAPÍTULO 14 .....</b>	<b>239</b>
<b>.TELAS DE OPERAÇÃO – OPERATOR SCREENS.....</b>	<b>239</b>
<b>Telas de Operação – Operator Screens.....</b>	<b>241</b>
Características da Tela de Operação.....	241
Criação de Aplicação de Tela de Operação.....	242
Configuração das Opções “Options” da Tela de Operação.....	242
Criação de Telas.....	245
Criação de Objetos Gráficos.....	248
Inserir Objetos Gráficos Na Tela.....	248
Animação dos objetos da biblioteca .....	250
Desagrupando objetos da biblioteca.....	252
Inserindo objetos a partir da área de edição.....	253
Inserir animações.....	253
Diagramas de Tendência.....	254
<b>CAPÍTULO 15 .....</b>	<b>263</b>
<b>.EXERCÍCIOS.....</b>	<b>263</b>
<b>Exercício 1: Criação de projeto padrão.....</b>	<b>265</b>
<b>Exercício 2: Sistema de Partida Direta de Motor de Indução Trifásico .....</b>	<b>266</b>
<b>Exercício 3: Sistema de Partida Direta com Reversão de Motor de Indução.....</b>	<b>266</b>
<b>Exercício 4: Controle do Sistema de Partida Direta com Reversão de Motor de Indução .....</b>	<b>267</b>
<b>Exercício 5: Controle Básico de Entrada Analógica .....</b>	<b>267</b>
<b>Exercício 6: Controle Básico de Entrada e Saída Analógicas.....</b>	<b>267</b>
<b>Exercício 7: Rampa de aceleração .....</b>	<b>267</b>

Exercício 8: Utilização de Bits e Words de sistema.....	268
Exercício 9: Controle de Nível em Diagrama de Blocos Funcionais – FBD.....	268
Exercício 10: Contagem dos produtos de uma linha de produção .....	268
Exercício 11: Utilização de Variáveis DDT em Controle de Turno .....	269
Exercício 12: Utilização de I/ODDT na supervisão de Entrada e de Saída Digital.....	269
Exercício 13: Aplicação de Bloco de Função Derivado – DFB no Controle de Sobrecarga .....	269
Exercício 14: Utilização de Telas de Operação para Controle de Nível .....	270
Exercício 15: Sistema de Análise de Ph .....	270
Exercício 16: Aplicação das funções de Exportação/Importação .....	270
Exercício 17: Utilização de Recursos de Utilidades do Unity .....	270
<b>CAPÍTULO 16 .....</b>	<b>271</b>
<b>GLOSSÁRIO .....</b>	<b>271</b>
Glossário .....	273

# CAPÍTULO 1

## **Plataforma Premium**

---



# Plataforma Premium

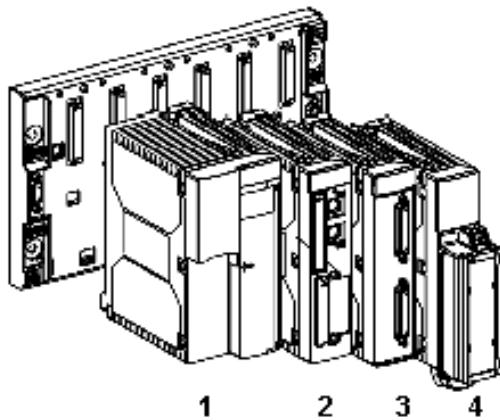
## Introdução

O CLP Premium é um Controlador Programável modular. Atende a uma grande diversidade de aplicações, devido à sua flexibilidade em termos de capacidade de processamento e possibilidades de expansão.



As estações Premium podem ser distribuídas sobre um ou diversos racks conectados ao Bus X ou a uma rede de campo.

Uma estação do Premium é composta por:



- (1) Fonte de Alimentação (formato padrão ou duplo, dependendo da potência da fonte);
- (2) Processador (formato padrão ou duplo, dependendo do modelo do processador);
- (3) Módulo de extensão;
- (4) I/O discreto, I/Os analógicos e módulos especiais.

Com exceção da CPU e fonte de alimentação, todos os módulos E/S e de aplicação específica podem ser instalados em qualquer posição do rack.

Os racks são usados para ligação elétrica e mecânica de todos os módulos do PLC. A modularidade é de 4, 6, 8 ou 12 módulos por rack. Podem existir diversos racks distribuídos por um sistema chamado Bus X, ligados uns aos outros por cabos de extensão. Os cabos estão ligados por dois conectores Db-9 na extremidade de cada rack. A CPU pode estar em qualquer um destes racks.

Os processadores constituem uma gama diversificada em termos de capacidades, de modo a satisfazer as diferentes necessidades de cada aplicação, em conjunto com a capacidade de comunicação integrada, e adicionalmente módulos de comunicação se forem necessários.

Os módulos de comunicação permitem a interligação deste CLP em outras redes de comunicação mais ou menos complexas.

Os módulos para aplicações específicas permitem ao CLP funcionalidades adicionais para um desempenho ótimo em determinadas aplicações, como é o caso do módulo de pesagem industrial ou do módulo de controle de posicionamento, por exemplo.

## Capacidade de expansão

O Premium pode expandir o número de racks através do Bus X ou através de Entradas/Saídas distribuídas (Advantys ou Momentum). Descreve-se aqui apenas o Bus X, remetendo para capítulo próprio as Entradas/Saídas distribuídas.

O Bus X disponibiliza uma velocidade de transferência de 12 Mbps determinísticos e permite instalar até 128 módulos/16 racks ao longo de 100m, sem repetidor.

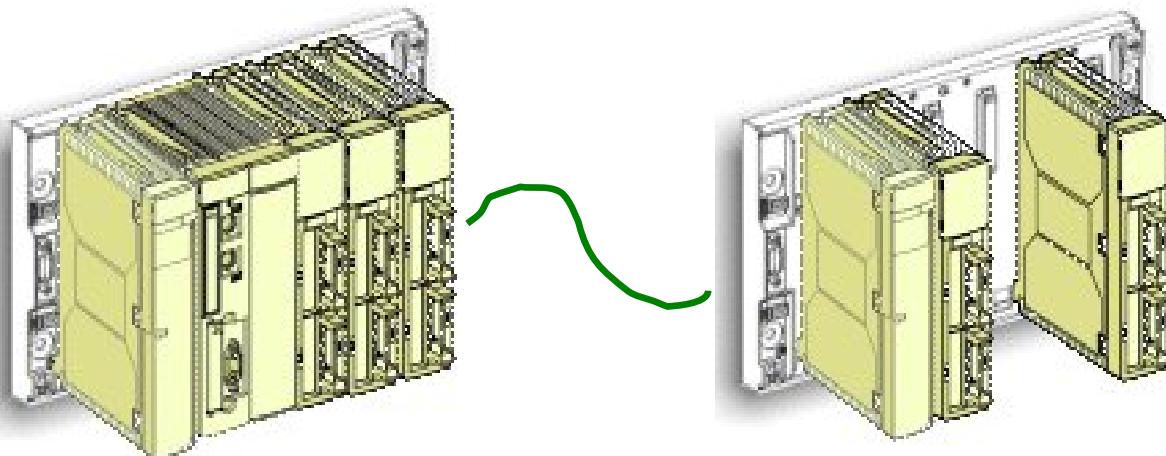
Em caso de aplicações que requerem distâncias entre racks maiores que 100 metros, o módulo remoto Bus X pode ser usado para aumentar a distância. Não altera a desempenho da aquisição e controle das E/S. O módulo mestre do Bus X é instalado no rack que suporta a CPU e um ou dois módulos escravos localizados em cada rack de extensão.

O módulo mestre tem dois canais que permitem 2 segmentos independentes com uma distância máxima de 250 metros. A distância máxima entre dois extremos pode ser 2 x 250m (extensão com módulo remoto) + 2 x 100m (extensão sem módulo remoto), Tendo assim a capacidade de atingir até 700 metros de um extremo ao outro.

A distância máxima permitida, relativamente ao processador, é limitada nos seguintes módulos:

- Módulos Ethernet: não pode ser remoto
- Módulos rede terceiros: não pode ser remoto
- Módulo comunicação serial: não pode ser remoto
- Módulo pesagem, AS-i: 175m
- Módulo E/S rápidas: 175m
- Módulo E/S Analógicas: 175m

### *Expansão de racks - Bus X*



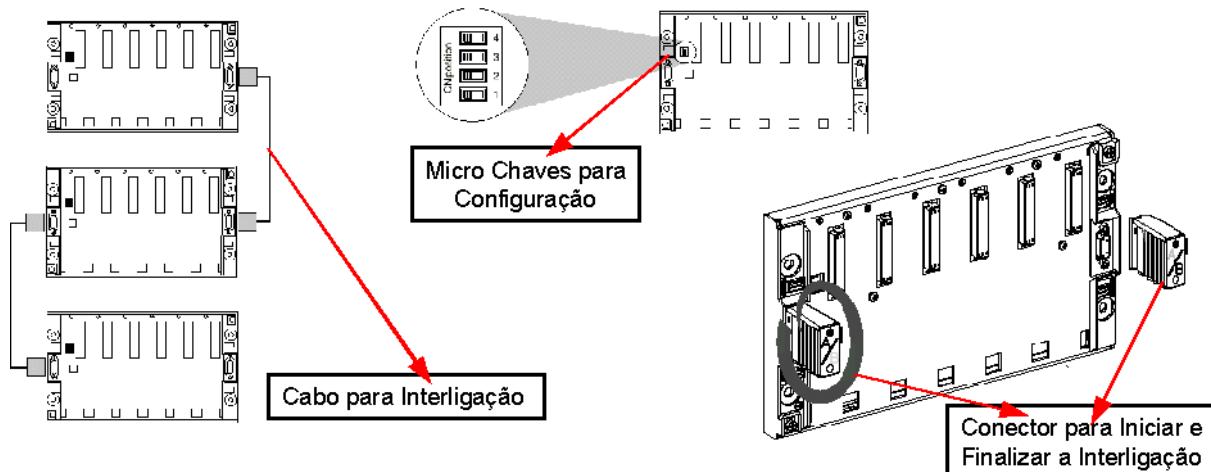
A Linha Premium permite que uma única CPU trabalhe com Módulos distribuídos em até 8 Racks, para isto os Racks devem ser interligados por um cabo especial e configurados através de Micro Chaves (Localizadas na parte inferior do Rack). Precisamos também utilizar um conector especial no início e no final da Interligação.

#### **Características:**

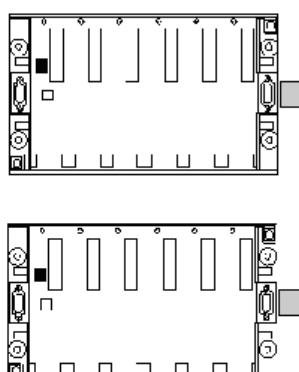
- Cada Rack deve possuir sua própria fonte de alimentação.
- Velocidade: 12Mbps;
- Um único cabo para conectar os racks;
- Comprimento máximo de 100m;
- Até 8 racks de expansão, porém cada rack de expansão pode ter um rack de extensão, sendo assim podemos afirmar que conseguimos um total de até 16 racks;
- Qualquer módulo em qualquer rack, sem restrição ou diminuição de desempenho;

- Saque a quente dos módulos (Somente uma chave de fenda é necessária).

### Conexão entre racks:



### Cabo para interligação dos racks:



TSX CBY 010K	1 METRO
TSX CBY 030K	3 METROS
TSX CBY 050K	5 METROS
TSX CBY 120K	12 METROS
TSX CBY 180K	18 METROS
TSX CBY 280K	28 METROS
TSX CBY 380K	38 METROS
TSX CBY 500K	50 METROS
TSX CBY 720K	72 METROS
TSX CBY 1000K	100 METROS

Distância Máxima entre Racks = 100 m



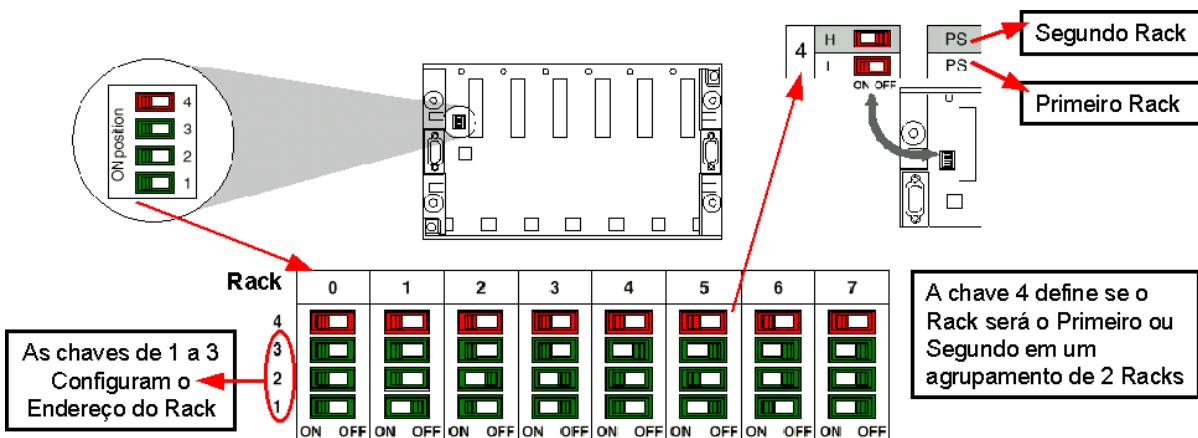
Pode-se de acordo com a necessidade aumentar estas medidas até 750m, usando módulos repetidores.

### Procedimento para configuração dos racks de expansão

Cada rack possui um conjunto de 04 micro-chaves seletoras, onde a combinação formada com o posicionamento delas nos permite configurar o endereço de cada rack de expansão, e, além disto, também nos possibilita definirmos quem é o primeiro rack e quem é o segundo rack em uma extensão (a seguir veremos o que é uma extensão).

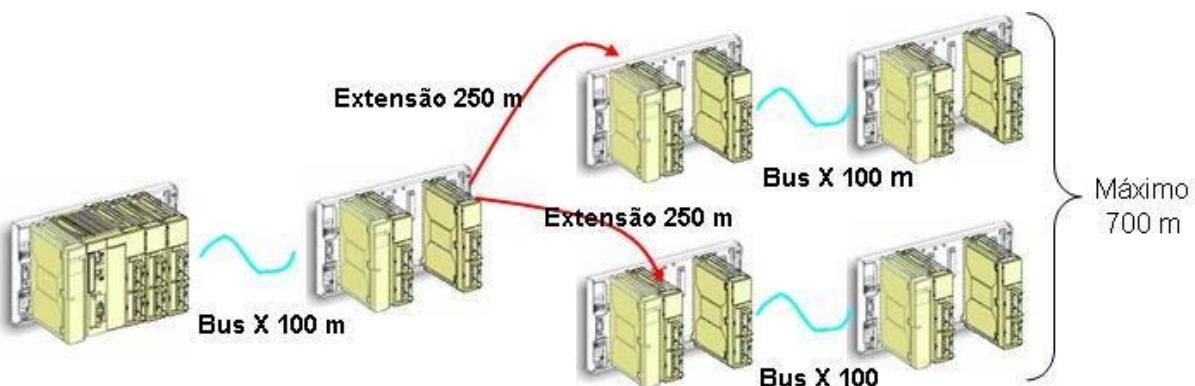
Este endereçamento do rack é utilizado na formação do endereço dos I/Os que estiverem no rack de expansão e extensão que será utilizado no programa que iremos desenvolver.

As quatro chaves presentes no rack na posição da fonte de alimentação determinam o endereçamento do rack.



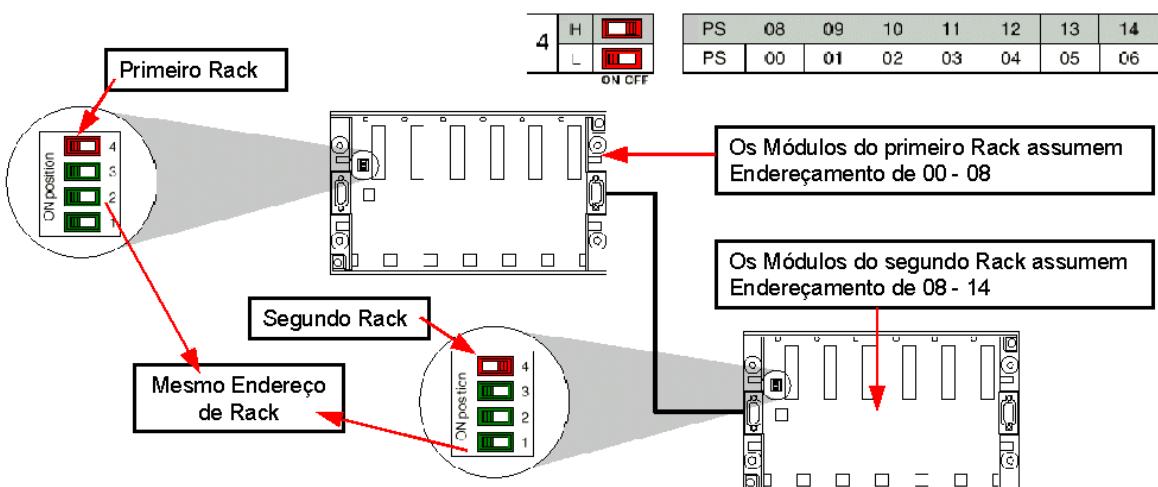
### Bus X (Extensão de Racks)

Os racks de extensão servem para expandir a quantidade de slots de um rack principal, seja ele o rack da CPU, ou um rack de expansão.



Use repetidores para estender o Bus X sem comprometer o desempenho;  
Comprimento máximo por extensão: 250m;  
Distância máxima entre dois racks: 700m.

### CONFIGURANDO RACKS DE EXTENSÃO



## Processadores



Cada processador Premium tem na face frontal dois terminais mini-DIN 8-pin (ligação RS485 não isolada)

- Ligação TER: usada para ligação de um PC ou de um bus UNI-TELWAY. A porta fornece uma alimentação de 5 V a um equipamento periférico, com a limitação de corrente imposta pela fonte de alimentação.
- Ligação AUX: usada para ligar equipamento periférico com alimentação autônoma, terminal, impressora, etc

O protocolo de comunicação nas portas TER e AUX é o protocolo UNI-TELWAY mestre a 19200 bauds; o protocolo UNI-TELWAY slave ou o protocolo ASCII podem ser selecionados por configuração.

Os processadores Modicon Premium podem ter as seguintes portas de comunicação

- Desde o 57-1x, pode ter porta Ethernet TCP/IP integrada
- Os 57-0x e 57-Cx têm uma porta CANopen mestre integrada.
- Desde o 57-1x até 57-5x está disponível uma versão com Fipio (SUB-D 9 pinos) integrada
- Exceto no 57-0x e 57-Cx, cada processador Premium tem um slot para módulos de extensão de memória PCMCIA tipo1 e um slot para cartão de comunicação PCMCIA tipo 3. Os tipos de cartão de comunicação que podem ser integradas nestes processadores são:
  - TSX SCP: cartões multi-protocolo (UNI-TELWAY, Modbus/Jbus, modo caracteres), RS 232D, anel de corrente (20 mA CL), compatível RS 485, RS422 isolado
  - TSX FPP: Agente FIPPIO, rede FIPWAY
  - TSX MBP: cartão de rede Modbus Plus
  - TSX CPP: cartão mestre de rede CANopen

A capacidade de processamento da família Premium varia de acordo com o modelo do processador.

- De 4 a 16 racks.
- De 512 a 2048 E/S digitais locais.
- De 24 a 256 E/S analógicas locais.
- De 8 a 64 canais de aplicações específicas.
- De 1 a 4 módulos rede (Ethernet, Fipway, Modbus Plus).
- De 2 a 8 redes AS-i.
- Redes Fipio.
- Possibilidade de módulos de ligação a redes (Interbus-S, Profibus DP, CANopen)

Os valores de E/S locais e remotas são cumulativas. Um total de mais de 10,000 E/S são possíveis: em rack + Fipio + redes de terceiros + AS-i.

Atualmente a família Premium é composta por 16 processadores, divididos em 6 níveis.

As características do PLC (número de E/S, módulos de aplicação específica, ligações de rede, etc.) são determinadas pelo modelo do processador.

	<b>57-00</b>	<b>57-10</b>	<b>57-20</b>	<b>57-30</b>	<b>57-40</b>	<b>57-50</b>	<b>57-60</b>
Racks	1	4	16	16	16	16	16
E/S Digitais	256	512	1024	1024	2048	2048	2048
E/S Analógicas	12	24	80	128	256	512	512
Posic./Contagem/Pesagem	4	8	24	32	64	64	64
Ethernet TCP/IP	1	1	2	3	4	4	4
CANopen	Built-in	1	1	1	1	1	1
Interbus, Profibus-DP			1	3	4	5	5
AS-i	1	2	4	8	8	8	8
Variáveis Internas Max.	96 KB	96 KB	192 KB	208 KB	440 KB	896 KB	896 KB
Programa Máx.	128 KB	224 KB	768 KB	1.7 MB	2 MB	7 MB	7 MB
Ethernet Incorporada	✓	✓	✓	✓	✓	✓	✓
Fipio Incorporada	✓	✓	✓	✓	✓	✓	✓
USB Incorporada				✓	✓	✓	✓

### Capacidade de processamento:

A capacidade máxima em "K instruções equivalentes" é dada na seguinte tabela:

<b>Max. Capacidade</b>	<b>57-0x</b>	<b>57-1x</b>	<b>57-2x</b>	<b>57-3x</b>	<b>57-4x</b>	<b>57-5x</b>
<b>Booleana</b>	22K	42K	157K	374K	428K	1545K
<b>Numérica</b>	15K	29K	120K	260K	298K	1012K

O tempo de execução para "1 K instruções equivalentes" é dada na seguinte tabela:

<b>Tempo de execução</b>	<b>57-0x</b>	<b>57-1x</b>	<b>57-2x</b>	<b>57-3x</b>	<b>57-4x</b>	<b>57-5x</b>
<b>Booleana</b>	0,32ms	0.32 ms	0.27 ms	0.22 ms	0.063 ms	0.049 ms
<b>65% Booleana 35% Numérica</b>	0,48ms	0.48 ms	0.40 ms	0.32 ms	0.088 ms	0.071 ms

A área de memória do Premium consiste numa RAM interna para guardar a aplicação (dados, programa e constantes) que pode ser extensível através de um cartão de memória PCMCIA para guardar o programa e as constantes. Dependendo do tipo de cartão pode armazenar também dados e símbolos da aplicação.

A memória da aplicação está dividida em diferentes áreas, fisicamente distribuídas na RAM interna e no cartão de memória PCMCIA (se existir):

- A área dos dados da aplicação é sempre na RAM interna.
- A área do programa pode ser na RAM interna ou no cartão de memória PCMCIA
- A área de constantes pode ser na RAM interna ou no cartão de memória PCMCIA

A área de armazenamento de dados é de 256 Kbytes ou 1.2 Mbytes (dependendo do modelo do cartão PCMCIA ) e pode ser utilizada para aplicações distribuídas ou para armazenar informações sobre dados de produção, receitas de fabrica, etc.



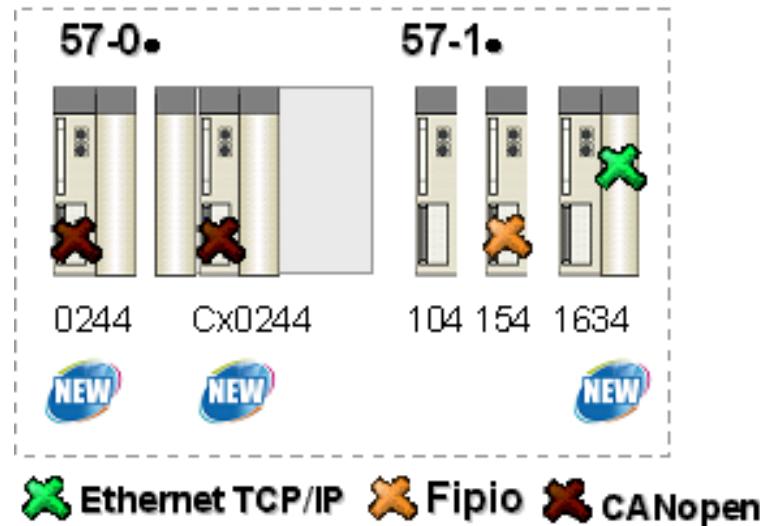
Dois tipos de extensão de memória estão disponíveis:

**Cartão de memória do tipo RAM protegido** – este cartão é utilizado principalmente na criação de programas da aplicação e nas fases de correção de erros. Permite transferências de dados e modificações em modo “online”. A memória é protegida por uma bateria integrada no cartão de memória.

**Cartão de memória do tipo Flash EPROM** – este cartão é utilizado quando a fase de correção de erros termina e permite apenas a transferência da aplicação completa. Com certos modelos de cartões RAM ou Flash EPROM á possível guardar dados (receitas de fabricação, dados de produção). Nos cartões RAM o tamanho desta área é definido pelo usuário. Além disso, é possível utilizar um cartão especial no encaixe PCMCIA de comunicação, ficando com até 8Mb adicionais.

#### Características Gerais:

##### a) CPUs de baixo custo

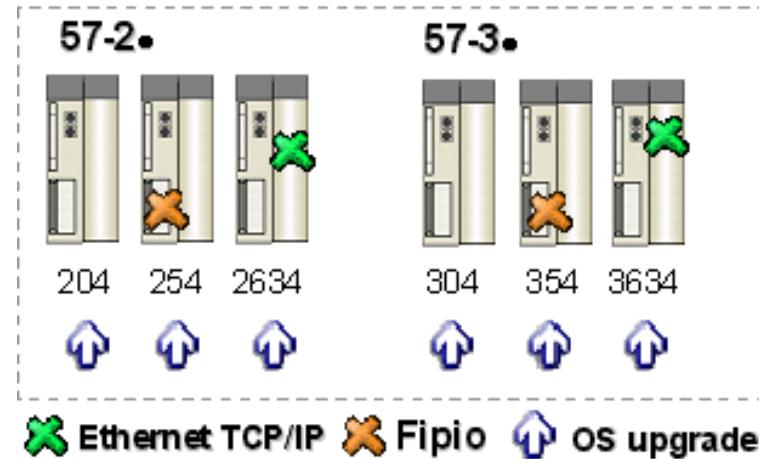


57-0 Com uma porta CANopen mestre incorporada;

57-C Suporta um rack de 6 slots, uma fonte CA ou DC e um módulo de contagem rápida com 2 canais de 40KHz;

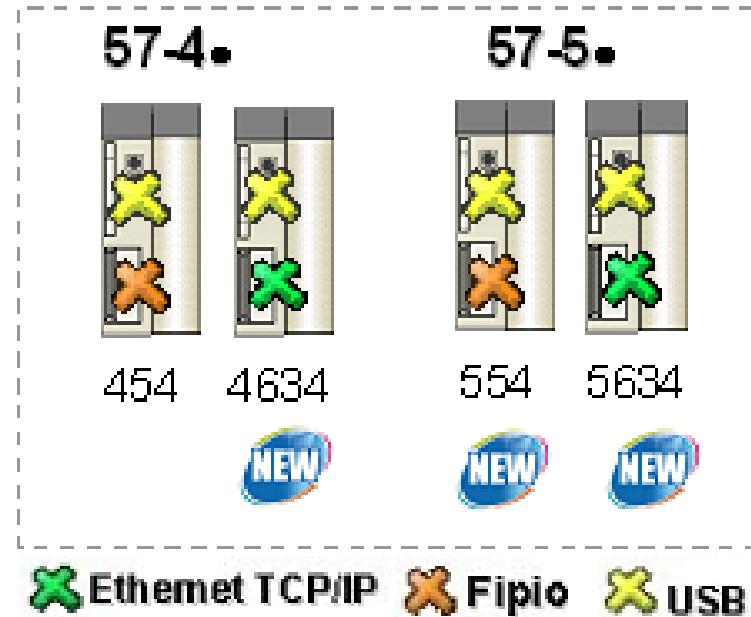
57-1 Com uma porta Ethernet incorporada.

##### b) CPUs standard



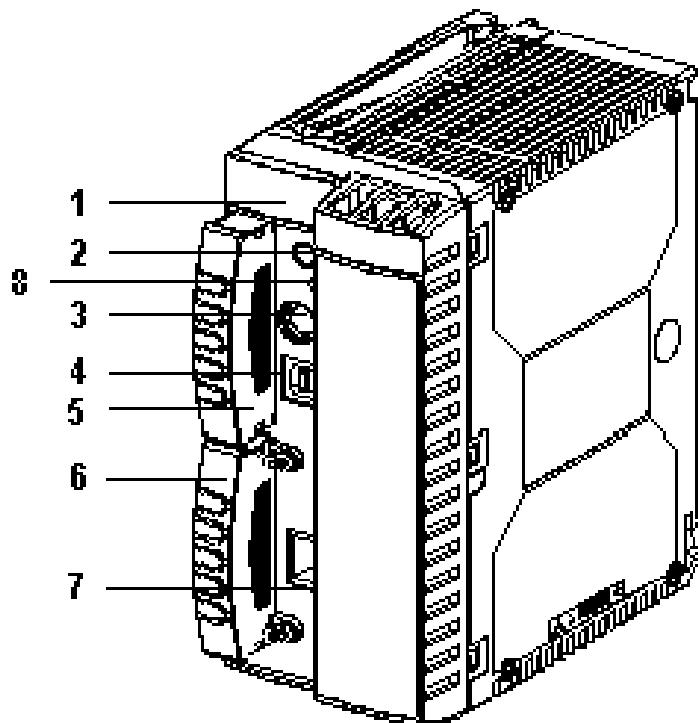
Para se utilizar o Unity, pode ser atualizado o firmware em um processador para o PL7  
 (Somente para família de CPUs L3)  
 Conversão de aplicações do PL7;  
 Aumento no desempenho;  
 Aumento da memória de dados e programa.

### c) CPUs de alta performance



Novo poder de processamento excepcional;  
 Mais memória para aumentar programa e capacidade de dados;  
 Aumento na capacidade de controle de processo e analógico;  
 Uma porta USB para transferência a 12 Mbaud;  
 Uma porta Ethernet TCP/IP incorporada.

### Descrição física dos processadores



Display (1);

Botão para atualização de dados ou para extração do cartão de memória. Aperte este botão para extraír o cartão (2);

Conector Terminal, para se conectar ao PLC (8-pin mini DIN) (3);

Porta USB (4);

Slot PCMCIA para cartão de extensão de memória (5);

Slot PCMCIA para cartão de comunicação (6);

Conector RJ45 para conexão em Ethernet (7);

Botão reset, o acionamento deste botão causa uma partida a frio (8).

## Sistema operacional Premium – Unity Pro

O sistema operacional tem uma estrutura multitarefa.

1 tarefa mestre periódica ou cíclica

1 tarefa rápida periódica com prioridade em relação à tarefa mestre.

4 tarefas auxiliares periódicas, geralmente com um período longo, para operações relacionadas com processamento (apenas no 57-5x)

De 32 a 128 tarefas ativadas por evento, dependendo do nível do CPU. O processamento destas tarefas é prioritário em relação a todas as outras tarefas. São adequadas para processos que requerem um tempo de resposta muito curto em relação à ocorrência do evento.

Tarefas cíclicas podem ter um tempo de resposta mínimo até 1 ms com o 57-5x e 2ms com outros. Cada tarefa é organizada em secções pelo utilizador. Isto permite ao utilizador estruturar o programa e utilizar qualquer uma das 5 linguagens IEC (exceto o SFC que apenas pode ser utilizado na tarefa mestre). O tempo de execução de cada varredura (periódico e cíclico) é supervisionado por um “watchdog”, cujo valor é definido pelo utilizador. Se o valor é excedido, é assinalada uma falha.

Cada canal ou grupo de canais está ligado a uma tarefa na tela de configuração para o módulo correspondente. Dependendo do tipo de módulo, o canal pode ser atribuído à tarefa mestre ou a uma tarefa rápida, de acordo com o pretendido.

Independentemente da estrutura da memória (aplicação em RAM interna ou em PCMCIA) é possível proteger a memória para proibir o acesso online (leitura/escrita do programa) com Unity Pro. Uma entrada física, definida em modo de configuração, está também disponível para proibir qualquer alteração ao programa.

Se a aplicação ficar em STOP, as saídas podem ser definidas para ficarem num estado seguro para a aplicação. Este estado de falha é definido ao configurar as saídas de cada módulo. A configuração é usada para selecionar:

Estado de falha: os canais ficarão a 0 ou 1 de acordo com o valor de estado de falha definido.

Mantém: a saída mantém o seu estado

O sistema memoriza o contexto da aplicação em caso de falha de alimentação. Quando a alimentação volta, o contexto guardado é comparado com o contexto corrente. Isto define que tipo de retorno será feito:

Se o contexto da aplicação se alterou (perda do contexto ou nova aplicação) o PLC faz um retorno “a frio” e inicializa a aplicação.

Se o contexto da aplicação é idêntico, o PLC faz um retorno “morno” mas não inicializa os dados.

Os bits e palavras de sistema indicam os estados do PLC e são utilizados para controlar a sua operação. Alguns são geridos pelo sistema, outros pelo usuário.

Os módulos de alimentação para alimentação dos racks e módulos instalados podem aguentar os seguintes micro-cortes.

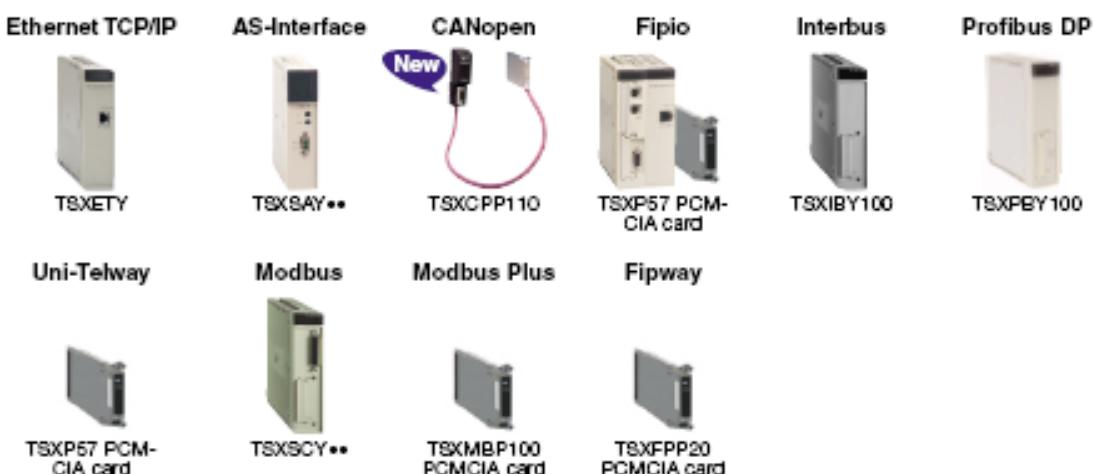
Em corrente alternada: micro cortes menores que 10ms

Em corrente continuada: micro cortes menores que 1 ms

O sistema operacional do Modicon Premium pode ser atualizado, carregando um novo firmware via porta de programação do processador.

O Unity Pro pode acessar um PLC cujo sistema operacional é mais antigo que a versão do software, sem necessidade de carregar o novo firmware.

## Comunicação



### Ethernet TCP/IP

#### Ligação através de módulo de comunicação

O Premium pode ligar-se a uma rede Ethernet TCP/IP através dos módulos TSX ETY (10 / 100 Mbps) com reconhecimento automático de velocidade.

O PLC suporta 1 a 4 módulos Ethernet dependendo do processador.

Os módulos têm no painel frontal:

- Display de Led's indicando o estado do módulo
- Conector RJ45 para 10baseT/100baseTX

Os módulos podem ser instalados em qualquer slot do PLC exceto nos slots dedicados à fonte de alimentação e ao processador. O módulo pode ser usado para trocar 800 mensagens/seg.

### Ligaçāo integrada no processador

O Premium pode ligar-se a uma rede Ethernet TCP/IP através de uma porta Ethernet TCP/IP integrada no processador, utilizando um terminal RJ45.

Estes processadores suportam os mesmos serviços TCP/IP que um cartão de comunicação TSXETY4xxx, mantendo todas as funcionalidades de um processador equivalente sem Ethernet integrada.

### Protocolo de comunicação

Os módulos TSX ETY e os processadores Premium com Ethernet integrada TSXP57xx23M suportam protocolo UNITE em modo cliente/servidor (pedidos de 256 byte e 1 Kbytes) e MODBUS (pedidos de 256 byte).

### Global Data

Os módulos e processadores Ethernet integram o serviço 'Global Data' que permite trocas em tempo real entre estações PLC. Pode-se interligar até 64 PLCs para uma base de dados global de 4K. O período de troca é configurável de 1 a n MAST ciclos.

### SNMP

Os módulos TSX ETY suportam SNMP V1 função agente suportado na norma MIB II base (RFC 1213). Integram também um MIB Transparent Factory que fornece dados relativos a funções específicas Schneider Electric (Global Data, varredura E/S, etc)

### Web Server

Os módulos TSX ETY têm um servidor Web incorporado. As funções disponíveis no servidor Web não necessitam de qualquer programação seja no PLC, seja no PC com o browser. Estas funções não afetam o tempo de varredura do PLC.

As funções principais são as seguintes:

- Diagnóstico de sistema do PLC (configuração, módulo, E/S)
- Acesso às variáveis e dados do PLC (password, leitura/escrita, acesso por endereço e símbolos)
- Visualização, reconhecimento e eliminação de alarmes (proteção por password)
- Editor gráfico com objetos animados ligados a variáveis do PLC (copy/paste de parâmetros, etc.)

Um módulo TSX ETY especializado fornece uma área de memória reservada de 8 Mbytes do tipo Flash EPROM para carregar páginas Web personalizadas. Estas páginas podem ser criadas em qualquer ferramenta standard para edição de páginas em formato HTML (FrontPage, Word 97, PowerPoint, etc.).

### Outros serviços TCP/IP

Os módulos TSX ETY funcionam a 10 / 100 Mbps e suportam um serviço de troca entre E/S remotas sem qualquer programação, incluem ainda:

- Atualização Transparente de E/S usando pedidos leitura/escrita com Modbus sobre TCP/IP.
- Área de palavras na aplicação reservada para leitura/escrita de E/S
- Períodos de atualização independentes da varredura do PLC
- Gestão das ligações TCP/IP para cada equipamento remoto
- Informação das palavras de estado para monitoração da correta operação da aplicação
- Aplicação de valores de emergência pré-configurados, para quando acontece um problema de comunicação
- Tempo de atualização para 64 equipamentos: 20 ms

De maneira a aperfeiçoar os serviços acessíveis pelos módulos ETY TCP/IP, estes têm integrado um mecanismo de controle de largura de banda que pode ser visualizado e modificado através de telas específicas do PL7 (gráficos de barras).

## Fipio



Os processadores e co-processadores 57-x54 têm uma porta SUB-D 9 pinos no painel frontal para ligar o PLC como gestor da rede. Podem ser ligados de 63 a 127 equipamentos, dependendo do modelo do processador.

Comprimento do bus: 15,000 m máximo

Comprimento do segmento: 1000 m (elétrico), 3000 m (fibra óptica)

Método do acesso: produtor/consumidor

Velocidade: 1 Mbps,

Meio físico: par entrancado blindado de 150 ohm, fibra óptica 62.5/125 ou 50/125 utilizando repetidores eletro/óticos

Número de equipamentos: 32 por segmento/128 em todos os segmentos conforme a norma WorldFIP.

## CANopen



Utiliza o cartão de comunicação PCMCIA tipo III, modelo TSX CPP e tem a função mestre de rede.

Características da rede:

Comprimento da rede em função da velocidade: 30m a 1 Mbps até 5000 m a 10 Kbps

Método de acesso: CSMA/CA, produtor/consumidor

Número de escravos: 127 max

O software de programação pode ser utilizado para importar a configuração da rede criada em CANopen para a aplicação do PLC. É também possível carregar a configuração diretamente, inserindo um cartão PCMCIA no slot compatível de um PC.

## Interbus-S (geração 4)



Utiliza o módulo de comunicação TSX IBY que tem a função mestre de rede. O módulo pode ser instalado em qualquer slot do PLC exceto nos encaixes reservados.

Este módulo tem:

Display com Led's para diagnóstico dos módulos e da rede

Terminal dB 9 SUB-D RS232 para ligação do terminal de programação

Terminal dB 9 SUB-D RS485 para ligação a rede

Características da rede:

Interface física: RS485

Método de Acesso: Master/Slave

Taxa de transmissão: 500 Kbps

Número de escravos: 512 max.

O software de programação pode ser utilizado para importar a configuração da rede para a aplicação do PLC.

## *Profibus DP*



Utiliza o módulo de comunicação TSX PBY que tem a função mestre de rede. O módulo pode ser instalado em qualquer slot do PLC exceto nos encaixes reservados.

Características da rede:

Interface física: RS485

Método de acesso: Master/Slave

Comprimento do cabo: 1200m - 9.6 Kbps e 100m - 12 Mbps

Número de escravos: 126 máximo.

## *AS-i*



Utiliza o módulo de comunicação TSX SAY que tem a função mestre de rede. O módulo pode ser instalado em qualquer slot do PLC exceto nos encaixes reservados

O PLC suporta 2, 4 ou 8 módulos AS-i dependendo do processador.

Compatível com sistemas AS-i versão V1 e V2.1

Número máximo de nós = 64

Compatível com interfaces analógicas

Compatível com interfaces de segurança

## **Normas e certificados**

Premium está projetado segundo as seguintes normas:

- IEC/EN 61131-2
- CSA 22-2 (Canadian Standards Association)
- UL 508 (Underwriters Laboratories)
- C-Tick ACA (Australian Communication Authority/Australia)
- Hazardous Location (CSA)
- Ghost CEI

Premium Unity está em curso de certificação pelas seguintes classificações navais:

- BV (Bureau Veritas/France)
- DNV (Det Norske Veritas/Norway)
- GL (Germanischer Lloyd/Germany)
- LR (Lloyd's Register/Unidadeed Kingdom)
- RINA (Registro Italiano Navale/Italy)
- ABS (American Bureau of Shipping)
- RMRS (Russian Maritime Register of Shipping)



## **CAPÍTULO 2**

# **Plataforma Quantum**

---



# Plataforma Quantum

## Introdução

O Quantum é uma plataforma de automação de natureza modular. É composta por um ou mais bastidores com os seguintes tipos de módulos: processador, fonte de alimentação, E/S digitais e analógicas, módulos de aplicação específica. Estes módulos podem ser distribuídos por 64 bastidores no máximo

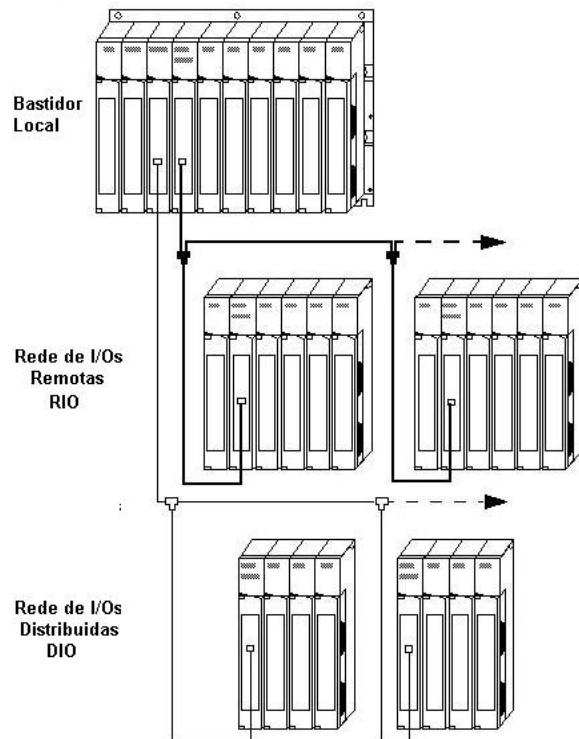


Os novos PLCs Quantum utilizam processadores Pentium;  
As estações Quantum podem ser distribuídas entre vários racks utilizando a tecnologia de Remote I/O ou Distributed I/O;  
Uma estação Quantum é composta de módulos de Fonte, Processador, I/O, Comunicação e especiais;  
Qualquer módulo pode ser inserido em qualquer posição no rack.

A plataforma Quantum apresenta uma grande variedade de módulos sendo dedicado a controle em tempo real em aplicações indústrias em uma arquitetura modular e expansível com os seguintes recursos:

- Controlador (CPU)
- Fonte de Alimentação (CPS)
- Módulos de I/Os Distribuídos (Dxx, Axx)
- Módulos de interface de redes (including Field Bus Modules)
- Módulos Inteligentes e de aplicações especiais.
- Módulos Simuladores (XSM) e baterias (XCP)
- Bastidor padrão (XBP) e expansor (XBE)
- Sistema de Cabeamento rápido (CFx)

Arquitetura básica da plataforma Quantum:



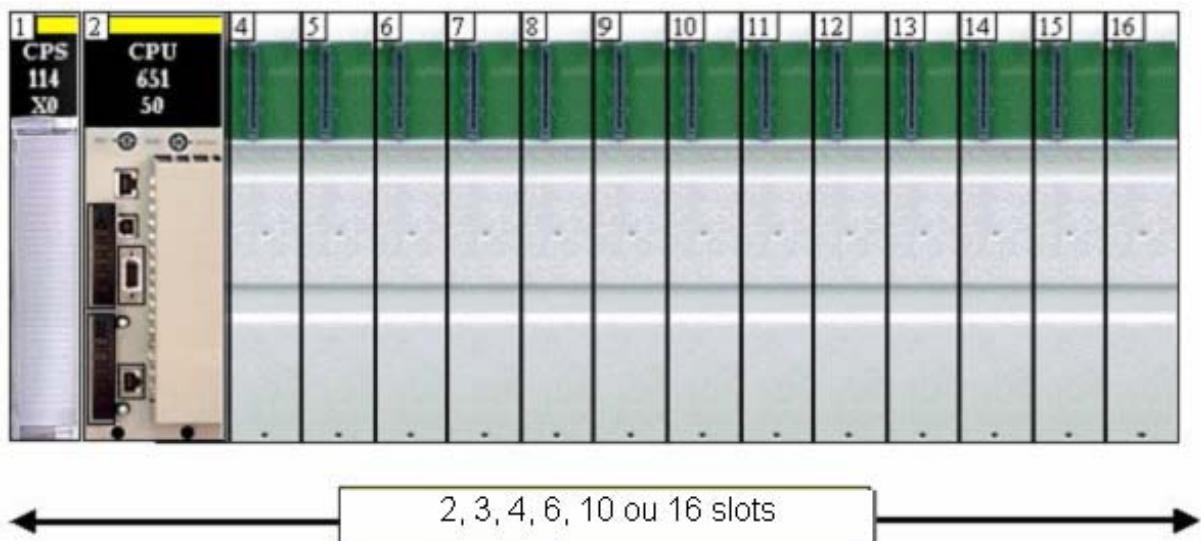
Detalhes da arquitetura de Redes.

Rede	Módulos de Interface de Rede	Mídia
I/Os Remotas (RIO)	RIO Heads: 140 CRP 931 00 140 CRP 932 00  RIO Drops: 140 CRA 931 00 140 CRA 932 00	RIO Cabo coaxial de até 15 Pés
I/Os Distribuídas (DIO)	NOM -Network Option Modules: 140 NOM 211 00: Modbus Plus Option Module 140 NOM 212 00: Modbus Plus Option Module 140 NOM 252 00: Modbus Plus Option Module 10Base-FL  DIO - Distributed I/O: 140 CRA 211 10 140 CRA 212 10 140 CRA 211 20 140 CRA 212 20	Par Trançado

## Bastidores

### Bastidor Padrão (XBP)

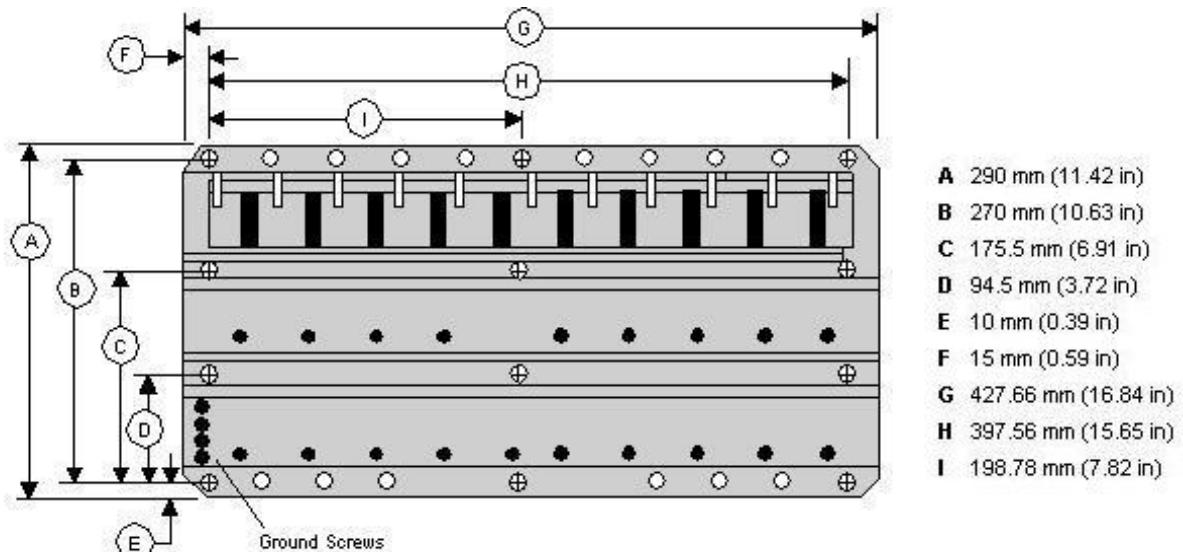
Pode ser usado em todas as aplicações de I/Os locais, remotas ou distribuídas. São disponíveis seis versões sendo 2, 3, 4, 6, 10 e 16 posições. Todas as posições podem ser usadas para qualquer tipo de modulo de I/O.



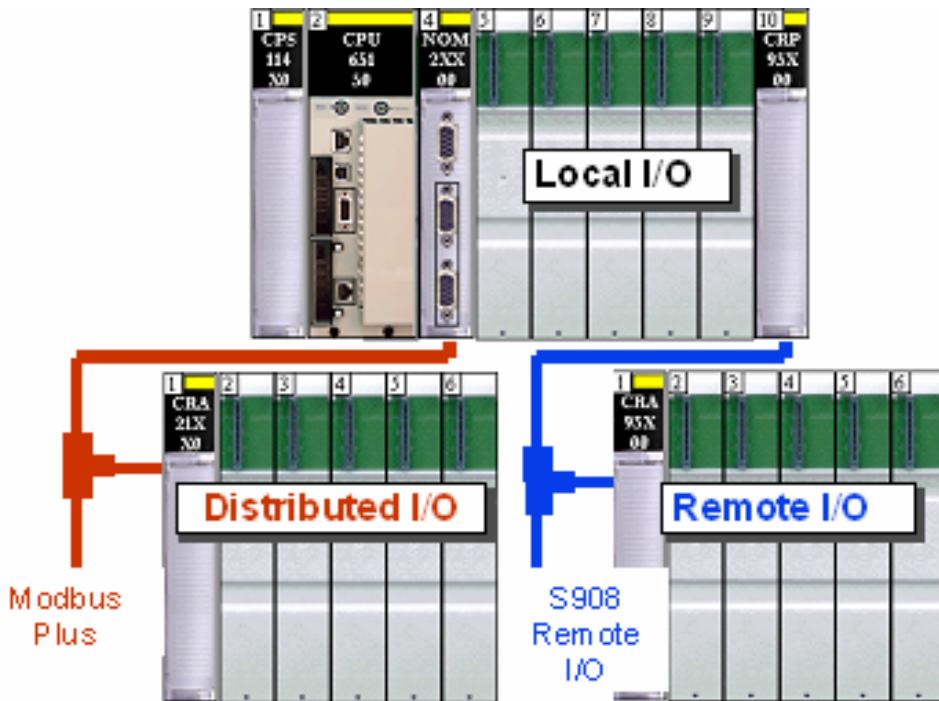
Tipos disponíveis:

- 140 XBP 002 00: Bastidor de Duas Posições
- 140 XBP 003 00: Bastidor de Três Posições
- 140 XBP 004 00: Bastidor de Quatro Posições
- 140 XBP 006 00: Bastidor de Seis Posições
- 140 XBP 010 00: Bastidor de Dez Posições
- 140 XBP 016 00: Bastidor de Dezesseis Posições

Medidas do Bastidor 140 XBP 010 00: Bastidor de Dez Posições.



## Arquitetura Flexível



### Local I/O

Com a arquitetura local, somente se utiliza um único rack contendo uma fonte, CPU, cartões de entradas e saídas, cartões especiais e cartões de comunicação, porém a arquitetura fica limitada ao tamanho do rack.

### Remote I/O

Com a arquitetura Remote I/O é possível distribuir até 31 estações remotas interligadas a estação local. A estação local possuirá a CPU e todas as estações remotas necessitam ter apenas uma fonte para a alimentação do rack, um módulo CRA para que se possa interligar esta estação remota a estação local e os cartões de entradas e saídas de acordo com a necessidade do processo. Todos os pontos de entradas e saídas da estação remota serão controlados pela CPU da estação local. As estações remotas ficam limitadas em no máximo 128 entradas e saídas analógica, sendo assim a sua combinação de cartões a serem inseridos em uma estação remota, não pode exceder este limite.

O protocolo utilizado para a tecnologia Remote I/O é o S908 e sua velocidade é de 1.544 Mbauds.

A interligação dos rack pode ser realizada utilizando cabo coaxial ou fibra ótica. Para cabo coaxial a distância máxima total é de 4500 metros e para fibra ótica é de 15 quilômetros.

### Distributed I/O

Com a arquitetura Distributed I/O é possível distribuir até 63 estações de entradas e saídas interligadas via protocolo Modbus Plus a estação local. Todos os pontos de entradas e saídas tanto da estação local quanto das estações distribuídas serão controlados pela CPU que deverá estar na estação local, e as estações precisaram ter uma fonte, e um cartão NOM para a interligação com a estação local além de pode ter conforme a necessidade da aplicação cartões de entradas e saídas.

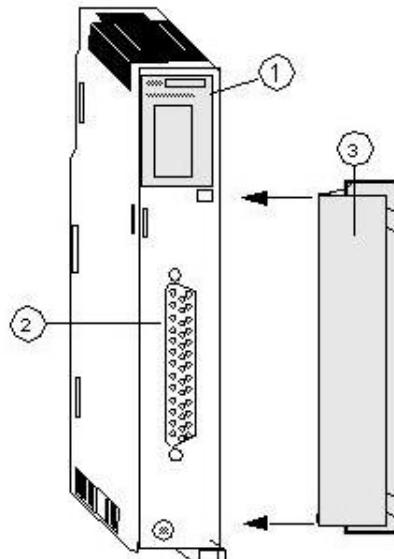
A interligação dos rack pode ser realizada utilizando cabo de par trançado ou fibra ótica. Para cabo de par trançado a distância máxima total é de 1800 metros e para fibra ótica é de 15 quilômetros.

## Expansão de Bastidor - Módulo Expansor (XBE)

O Módulo Expansor 140 XBE 100 00 permite ao Modicom Quantum a capacidade para expandir local ou remotamente as derivações de I/Os para um segundo bastidor.

O Módulo expansor atua como um repetidor para sinais de dados vindos de um bastidor primário Quantum e cada um dos bastidores devem ter módulos de fonte de alimentação próprios. O Módulo expansor não possui LEDs de sinalização de estado, estes são indicados por bits de diagnóstico no funcionamento do mesmo.

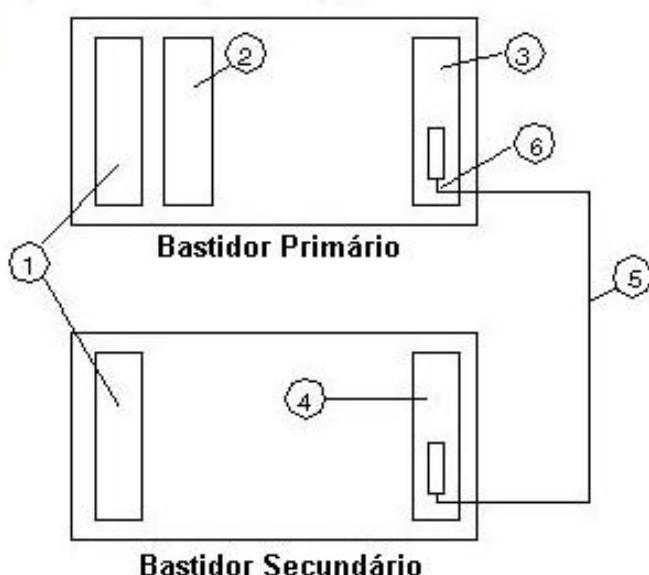
Visão Geral do Módulo Expansor:



- 1 - Número do Módulo, Descrição do Módulo e Código de Cor.
- 2 - Conector
- 3 - Tampa Removível

### Configuração básica:

O Bastidor que contém uma CPU ou um módulo de I/O remota (RIO) é chamado de Bastidor Primário, e o bastidor adjacente é denominado de Bastidor Secundário, cada bastidor requer seu próprio módulo de Fonte de Alimentação.



Onde:

- 1 Fonte de Alimentação
- 2 CPU ou Adaptador de I/O Remota (RIO)
- 3 Primeiro Módulo expansor (140 XBE 100 00)
- 4 Segundo Módulo expansor (140 XBE 100 00)
- 5 Cabo dos Módulos expansores (140 XCA 717 0•)
- 6 Terminal do cabo marcado como "Primary"

### Especificações Gerais:

Número máximo de Bastidores: 2;  
 Distância Máxima: 3m;  
 Número de Módulo Expansor permitido: 1/ Bastidor;  
 LEDs: Nenhum;  
 Endereçamento: Através do I/OMap;  
 Potência Consumida: 2,5W;  
 Corrente requerida do Barramento: 500mA;  
 Conector: 37 Pinos Tipo D

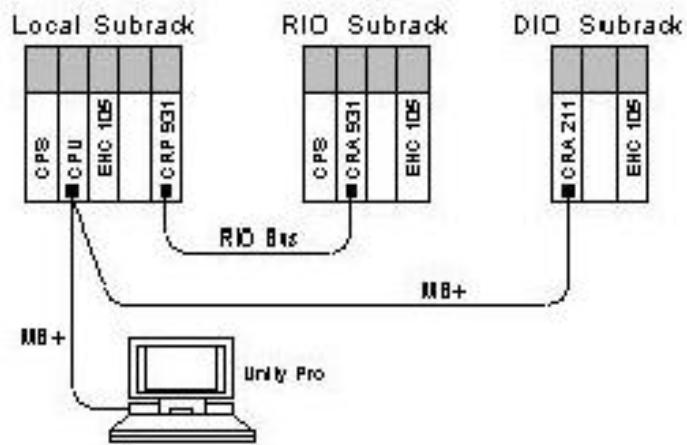
### Compatibilidade:

Bastidor Primário: Sem Restrições  
 Bastidor Secundário: Todos os tipos de Módulos de I/O Quantum podem ser usados no bastidor secundário, até que a documentação não dê informação contrária.

### Máximas Words/Derivação (Words/Drops):

I/O Local: 64 Entradas / 64 Saídas  
 I/O Remota: 64 Entradas / 64 Saídas

### Exemplo de Configuração Típica de Hardware:



### Síntese da Capacidade de Processamento

As capacidades de processamento do Quantum variam conforme o modelo do processador, tendo como limite:

- 32 Estações E/S, cada uma com dois racks: um rack principal e um rack de extensão
- 64.000 bits de E/S remotos, acessíveis através de Modbus
- 1024 bits por módulo com número ilimitado de palavras para o bastidor local,
- 1024 bits de entrada e 1024 bits de saída por cada Unidade Remota (E/S ou aplicação específica),
- 2 ou 6 módulos opcionais (Ethernet, Modbus Plus, Profibus DP, DeviceNet, SERCOS, Motion, Sy/Max Ethernet),

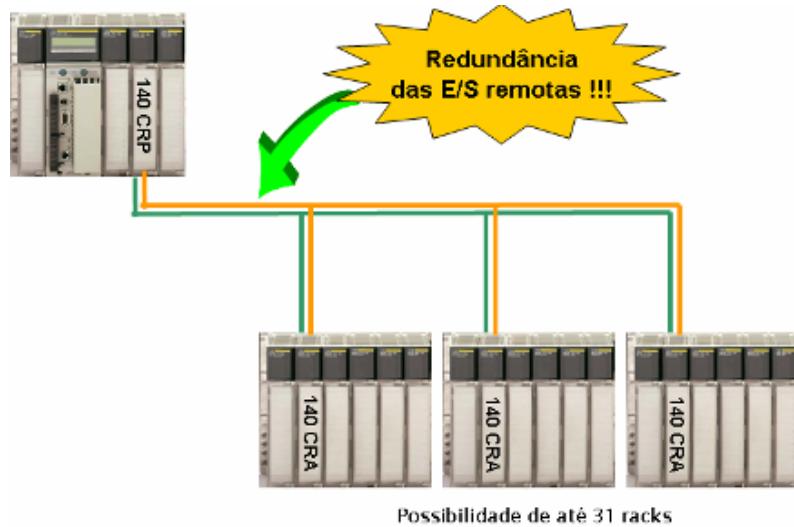
A arquitetura de E/S Remotas (RIO) é necessária se os slots do rack local que contém a CPU e os slots do rack de extensão estiverem ocupados, ou por necessidade de topologia.

Exemplo para cálculo de bits: um módulo de 32 entradas digitais "consome" 64 bits de entrada; um módulo de 8 saídas analógicas "consome" 128 bits de saída; um módulo de 16 entradas analógicas "consome" 17 palavras de entrada (valores & palavras de estado).

## Características do bus RIO

Distância máxima sem repetidores: 4500m com cabo coaxial,  
 Derivação máxima: 30 m,  
 Método de acesso: produtor/consumidor,  
 Velocidade: 1,544Mbit/s,  
 Meio: Cabo coaxial 75 ohm,  
 Tipo de cabo principal: rígido, semi-rígido ou flexível, de acordo com o tamanho da configuração (RG11, RG6)  
 Tipo de cabo de derivação: flexível (RG6)  
 Número de remotos por rede: 31  
 Protocolo proprietário (Codificação Manchester, HDLC)

### Exemplo de arquitetura Remote IO



### Características do módulo de comunicação 140 CRP:

140CRP	931	932
Bastidores por estação	2	
Bits E/S	1024E + 1024S	
Tipo de módulo	Digital e de aplicação específica	
Número de remotos	31	
Medium	Coaxial simples	Coaxial redundante
Tempo de aquisição	Max 5ms/estação	
Firmware	Memória Flash	
Protocolo	HDLC	
Velocidade	1,544mbits/s	
Gama dinâmica	35db	
E/S Suportadas	Séries E/S 200/500/800 e Quantum	
Alimentação	Do rack	

## Processadores Quantum

### Gama de Processadores

140 CPU	311 10	434 12A	534 14A	651 50	65160/67160*
<b>Racks locais</b>	2	2	2	2	2
<b>Unidades E/S remotas</b>	31*2 racks	31*2 racks	31*2 racks	31*2 racks	31*2 racks
<b>Unidades E/S distribuídas</b>	63	63	63	63	63
<b>Palavras E/S Local</b>	Ilimitado	Ilimitado	Ilimitado	Ilimitado	Ilimitado
<b>Palavras E/S por bastidor remoto</b>	64 E / 64 S	64 E / 64 S			
<b>Palavras E/S por bastidor distribuído</b>	30 E / 32 S	30 E / 32 S			
<b>Número de módulos opcionais (em bastidor local)</b>	2	6	6	6	6
<b>Modbus / ASCII integrado</b>	2 RS232	2 RS232	2 RS 232	1 RS232/ RS485	1 RS232/ RS485
<b>Modbus Plus integrado</b>	1	1	1	1	1
<b>Modbus Plus em bastidor local (Max.)</b>	2	6	6	6	6
<b>Ethernet TCP/IP integrado</b>	-	-	-	1	1 / 1 hot standby'
<b>Ethernet TCP/IP rack Local (Max.)</b>	2	6	6	6	6
<b>USB</b>	-	-	-	1	1
<b>Hot Standby</b>	-	-	-	-	- / SIM
<b>Programa e dados não-alocados sem PCMCIA</b>	400 Kb	800 Kb	2716 Kb	512 Kb	768 Kb
<b>Dados alocados &amp; config. (Max.)</b>	148 Kb	256 Kb	256 Kb	256 Kb	256 Kb
<b>Dados alocados (State RAM)</b>	10 K palavras	64 K palavras	64 K palavras	64 K palavras	64 K palavras
<b>Armazenamento de dados e programas com PCMCIA</b>	-	-	-	7168 Kb	7168 Kb
<b>Armazenamento de dados em PCMCIA</b>	-	-	-	8192 Kb	8192 Kb

Os Controladores (CPUs) Quantum, servem como um barramento mestre de controle local, remoto ou I/Os Distribuídos de sistemas Quantum.

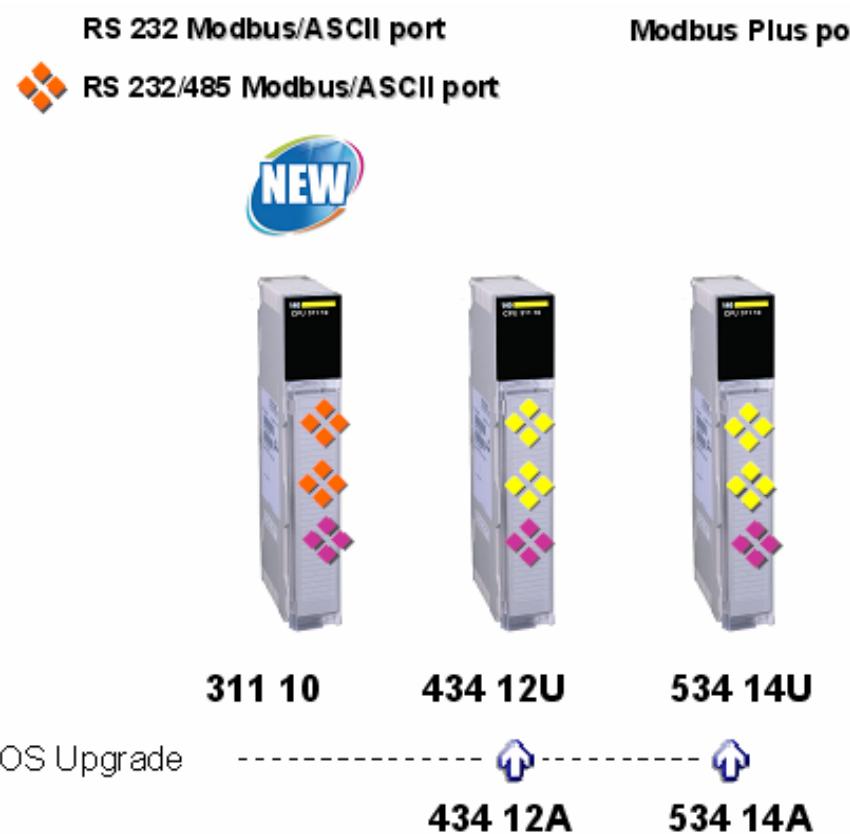
O módulo está instalado em um dos bastidores do sistema local de I/O. Há um sistema eletrônico operado digitalmente, o qual usa a memória programável para armazenamento interno das instruções do usuário. Estas instruções são usadas para funções lógicas, processamento seqüencial, temporização, acoplamentos, instruções aritméticas e outros.

Estas instruções permitem controle de saídas digitais e analógicas, para vários tipos de máquinas e processos.

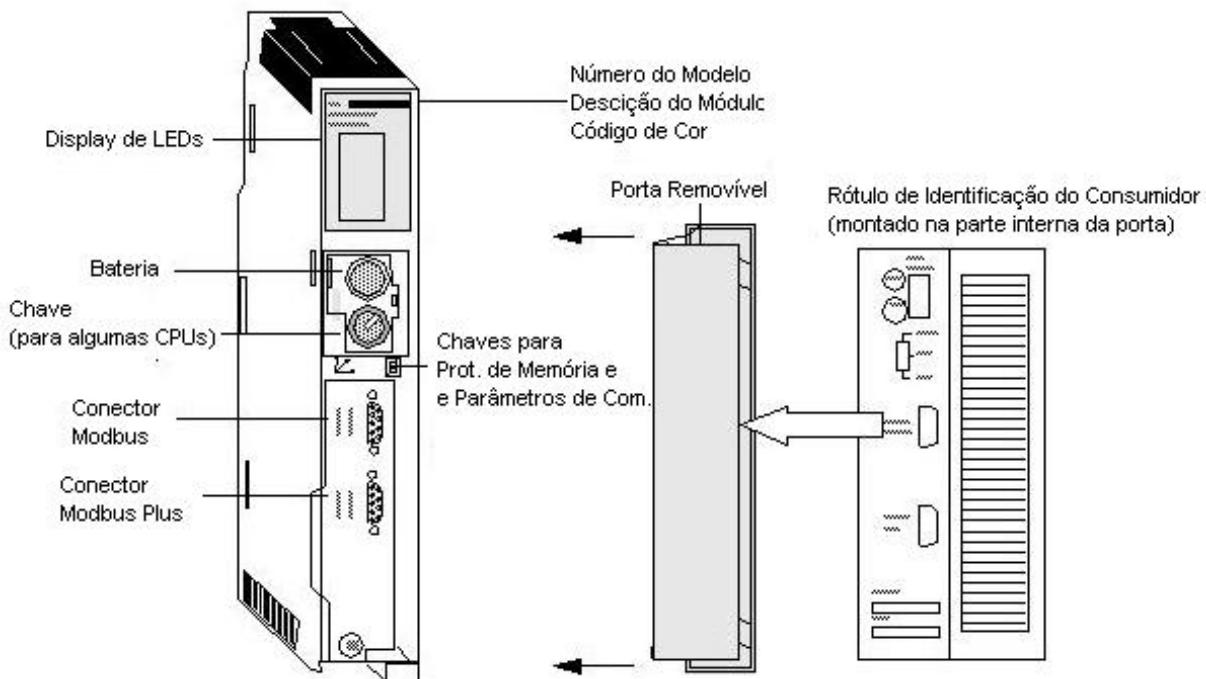
As CPUs do sistema quantum são divididas em dois grupos denominados **Modelo Standard** (Low End CPU) e **Alta Performance** (High End CPU), como mostra a tabela a seguir.

	CPU	SRAM (bytes)	Max IEC Program	Max IEC Program (with PCMCIA)	Key Switch
<b>Low End</b>	140 CPU 311 10	2 MBytes	400 kBytes	NA	No
	140 CPU 434 12A	2 MBytes	800 kBytes	NA	Yes
	140 CPU 534 14A	4 MBytes	2.7 MBytes	NA	Yes
<b>High End</b>	140 CPU 651 50	2 MBytes	512 kBytes	7168 kBytes	Yes
	140 CPU 651 60	2 MBytes	1024 kBytes	7168 kBytes	Yes
	140 CPU 652 60	4 MBytes	3072 kBytes	7168 kBytes	Yes
	140 CPU 671 60 (HSBY)	2 MBytes	1024 kBytes	7168 kBytes	Yes

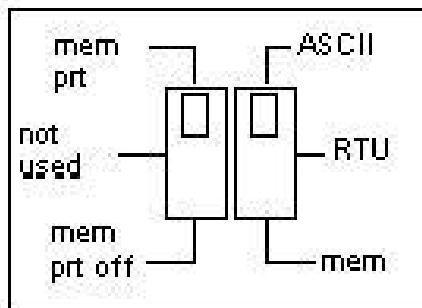
### Processadores Modelo Standard (Low End CPU)



### Visão Geral:

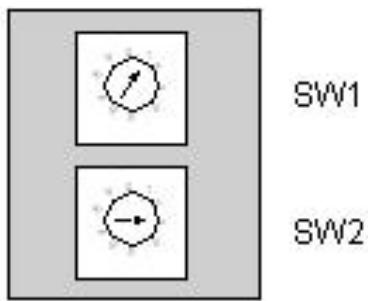


#### Chaves de Três Posições no Painel Frontal:



A chave da esquerda é usada para proteção de memória quando está na posição do topo e sem proteção quando está na posição do meio e embaixo. A chave da direita é usada para selecionar o ajuste dos parâmetros de comunicação para as portas Modbus (RS-232).

#### Chaves Rotativas no painel traseiro:



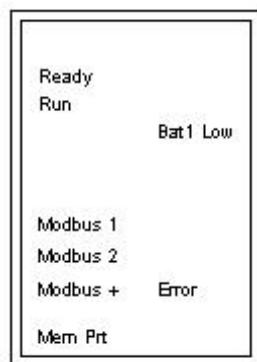
Duas chaves rotativas estão localizadas no painel traseiro da CPU. Estes são usados para ajustar o Número Modbus e endereço da porta Modbus, sendo o número máximo de endereço 64. SW1 (chave do topo) ajusta o dígito alto (dezena) do endereço; SW2 (chave de baixo) ajusta o dígito baixo (Unidade) do endereço.

Exemplos de endereçamento:

Endereço do Nó	SW1	SW2
1 ... 9	0	1 ... 9
10 ... 19	1	0 ... 9
20 ... 29	2	0 ... 9
30 ... 39	3	0 ... 9
40 ... 49	4	0 ... 9
50 ... 59	5	0 ... 9
60 ... 64	6	0 ... 4

Se “0” ou um endereço maior que 64 forem selecionados, o LED “Modbus+LED” estará aceso constante, indicando a seleção de um endereço inválido.

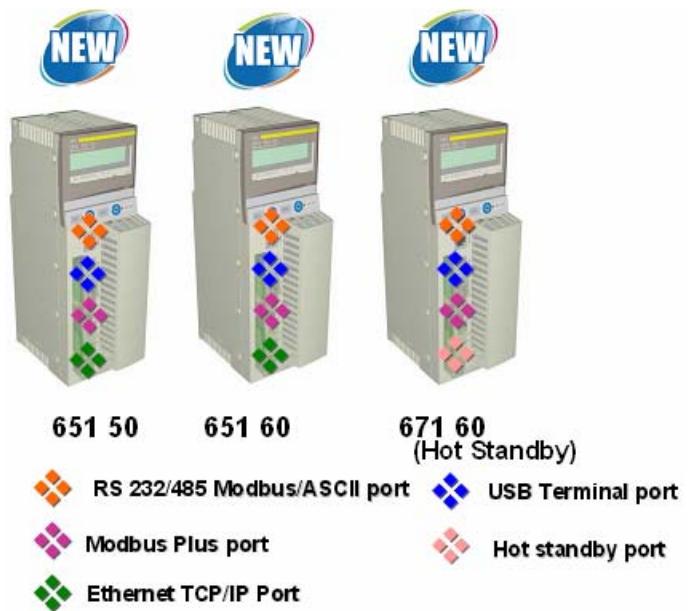
#### LEDs Indicadores:



Função dos LEDs indicadores:

LEDs	Cor	Indication when On
Ready	Verde	A CPU pelo diagnóstico de energização
Run	Verde	A CPU iniciou e está em uma solução lógica. De acordo com a seqüência de acendimento deste LED indica erros de funcionamento em Run. (Run LED Error Codes).
Modbus 1	Verde	Comunicação ativa na Porta Modbus 1 (Modbus port 1).
Modbus 2	Verde	Comunicação ativa na Porta Modbus 2 (Modbus port 2).
Modbus +	Verde	Comunicação ativa na Porta Modbus Plus (Modbus Plus port).
Mem Prt.	Âmbar	Memória Protegida de escrita ( chave de proteção de memória no painel frontal ligado).
Bat 1 Low	Vermelho	Necessidade de troca de bateria.
Error	Vermelho	Indica erro de comunicação na porta Modbus Plus.

#### Processadores Alta Performance (High End CPU)

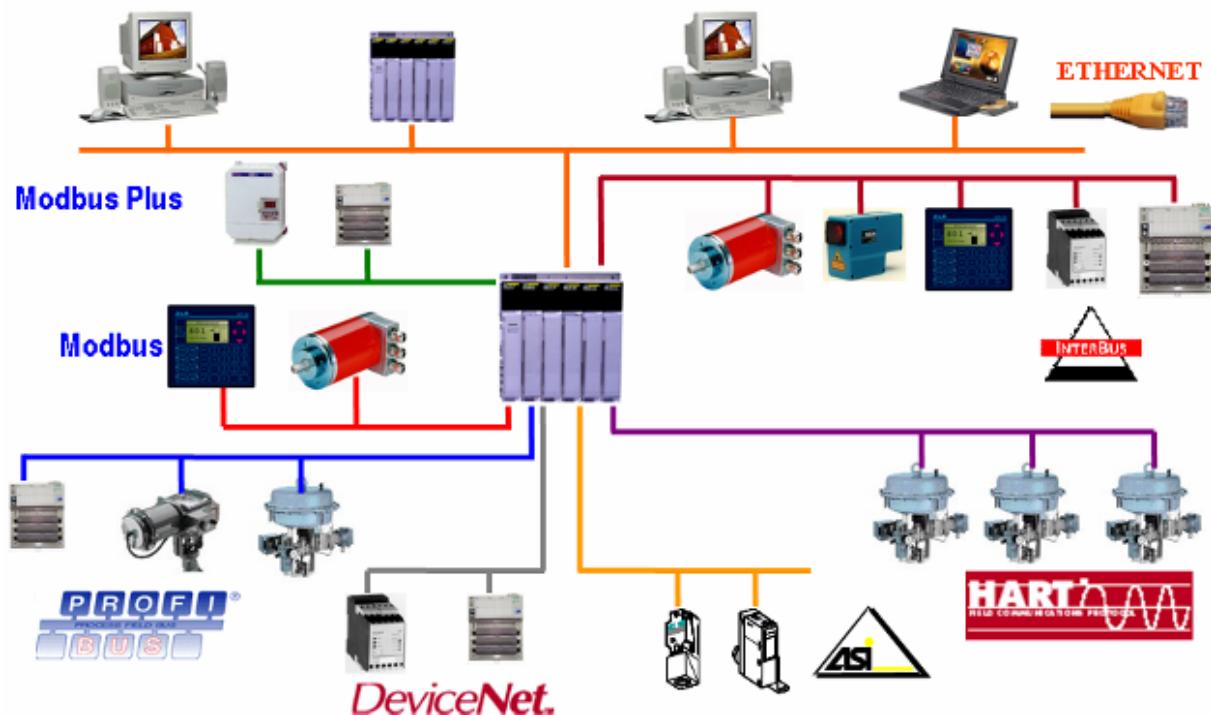


São disponíveis dois tipos de CPU de Alta Performance, a padrão e com Hot Stand By.

### Conectividade Modicon Quantum

Além das redes de transporte acima citadas, a família Quantum também possui conectividade com as redes MODBUS, MODBUS PLUS, HART, ASI, INTERBUS.

### CLP's Modicon Quantum Conectividade = Schneider Modicon



## CAPÍTULO 3

### Plataforma M340

---



# Plataforma M340.

## Introdução

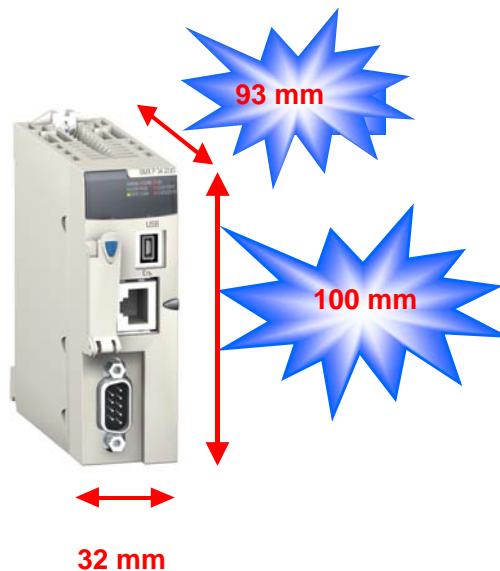
A plataforma M340 é otimizada para médias & pequenas configurações:

- Tamanho
- Alta & Baixa densidade
- Portas de comunicação embarcada
- Alta robustez
- Em todos os módulos pode-se conectar/desconectar com a CPU em RUN



Por ser concebido sob moderna tecnologia de integração, apresenta tamanho reduzido.

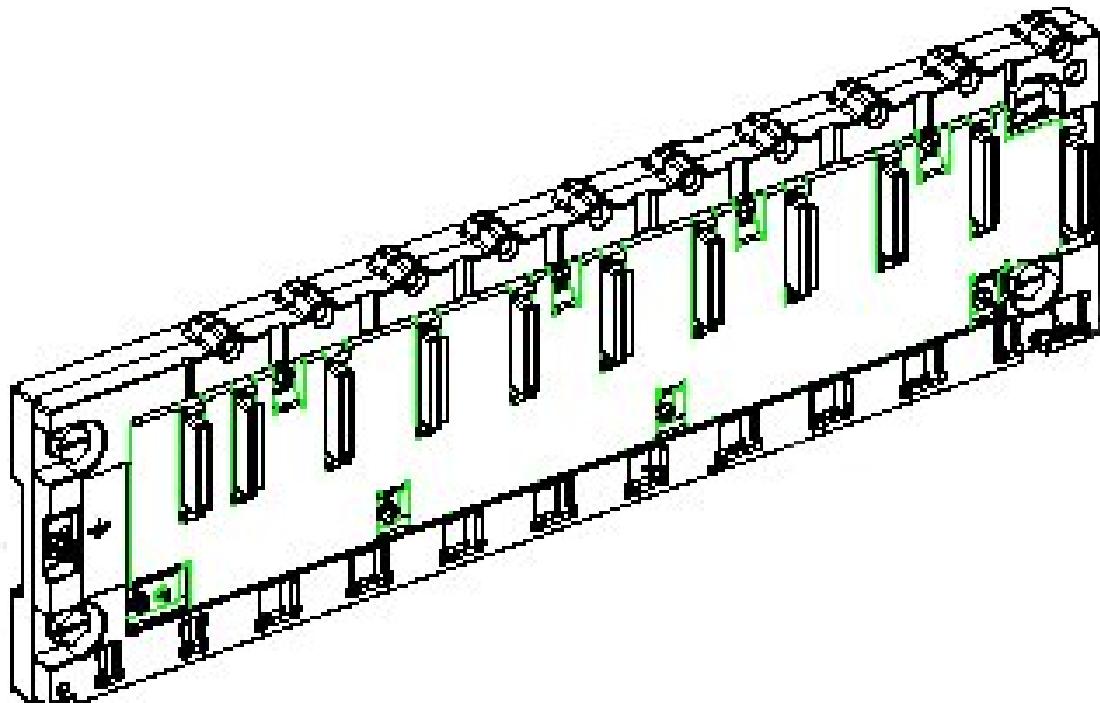
- Montagem em gabinete menores (<150mm)
- Todas as CPU e I/Os tem a mesma largura



### Especificações do Sistema:

- Restrições Mecânicas
  - Choques : 50g
  - Vibração : 5g
- Temperatura
  - Ambiente : 0 to 60 °C (Acima 2000 m limite 55° C )
- Altitude
  - Todas as restrições garantidas para 4000 m

## Bastidores “racks” Modicon M340.



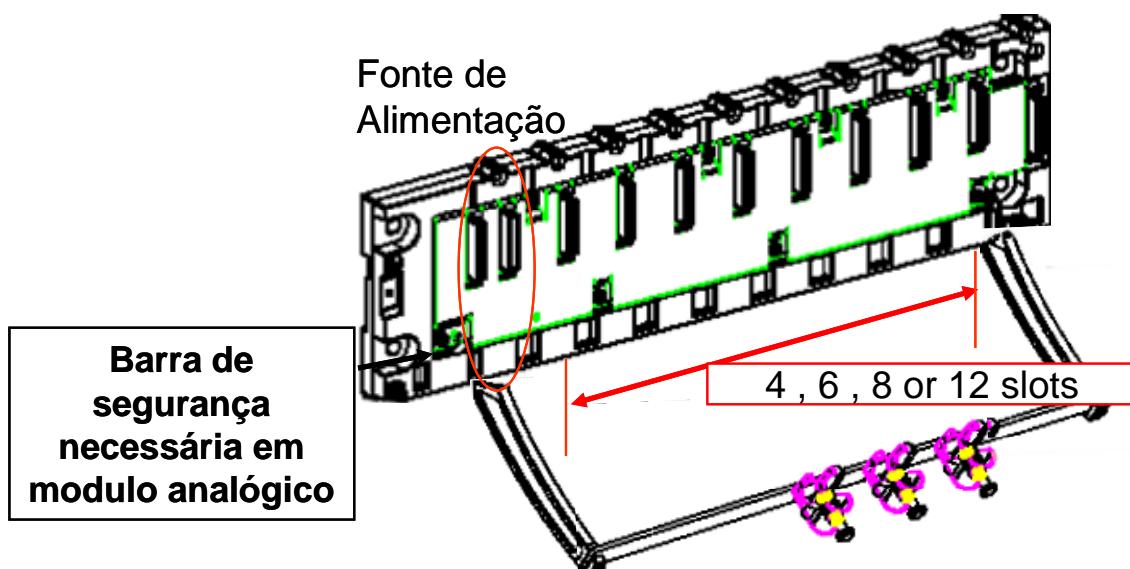
Os Racks BMX XBP xxx formam a unidade de base do M340, estas unidades tem as seguintes funções:

### Funções Mecânicas:

Estes são usados para montagem de um conjunto de módulos para estações PLC (Ex. Módulos de alimentação, processadores, módulo de entradas/saídas discretas/analógicas, módulos de aplicações específicas). Podendo ser montados em cabines, quadros de máquinas ou em painéis.

### Funções Elétricas:

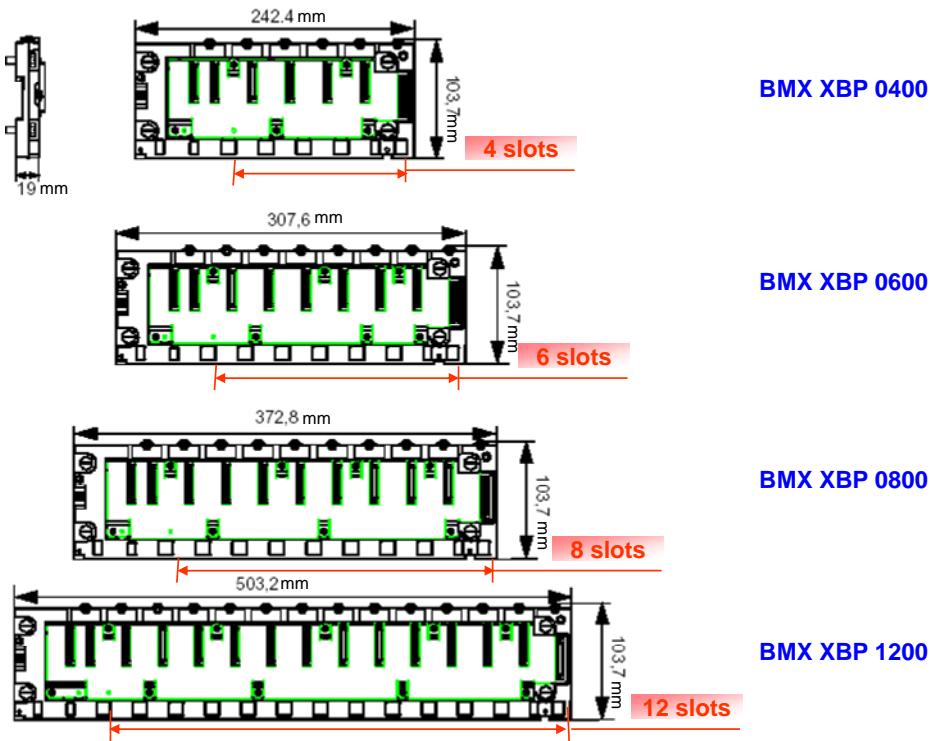
Estes racks são construídos em barramento PLC, os quais distribuem a alimentação necessária para cada modulo num mesmo rack.



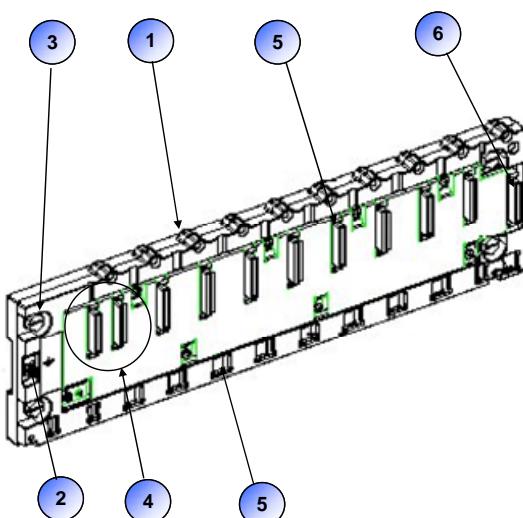
## Modelos Disponíveis:

Características e performance				
Rack	BMX XBP 0400	BMX XBP 0600	BMX XBP 0800	BMX XBP 1200
Descrição simplificada	4 slots na placa principal	6 slots na placa principal	8 slots na placa principal	12 slots na placa principal
Barramento de Segurança	BMX XSP 0400	BMX XSP 0600	BMX XSP 0800	BMX XSP 1200
Descrição simplificada	4 slots de proteção	6 slots de proteção	8 slots de proteção	12 slots de proteção

## Tamanho: Racks BMX XBP xxxx



## Detalhamento do Rack M340

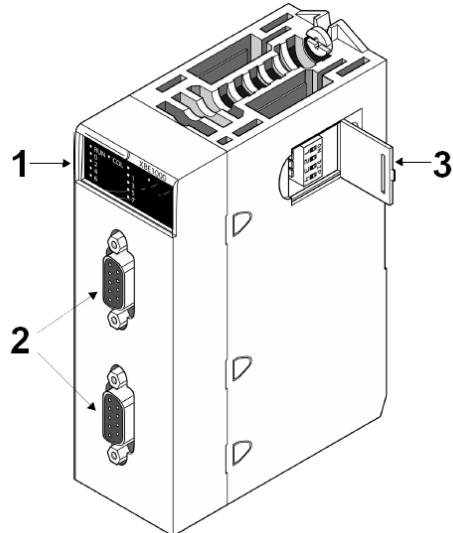


Descrição	
1	Placa de metal: Supoorte a cartão eletrônico X-Bus, e protege de interferências entre EMI e ESD. Montagem em racks rígidos
2	Terminais de terra para aterrramento do rack.
3	Cavidade para montagem no rack. Estas cavidades podem utilizar parafusos M6.
4	2 conectores de 40 pontos dedicados para a conexão do rack/ fonte de alimentação
5	Partes mecânicas dedicadas para acomodação do módulo.
6	Conector de 40 pontos Dedicados para expansão do rack
7	Conector de 40 pts dedicado à conexão rack/modulo

## Multi-rack

### Extensor de rack BMX XBE1000

- Compatível com qualquer rack produzido desde julho de 2007
- Instalado no slot dedicado na extremidade direita do rack marcado como “XBE” (**não reduz os slots para módulos de E/S**)
- Características:
  1. Display utilizado para diagnósticos
  2. Possui 2 conectores DB9 (fêmea) na parte frontal do módulo para ligar cabos e/ou o terminador de linha TSXTLYEX (o mesmo que o Modicon Premium já utiliza)
  3. Chaves para seleção do endereço do rack (0 a 3) no barramento local



### Cabos com conectores angulares (BMX XBCxxxK)

- Reduz a profundidade utilizada no painel
- Todos os cabos do Modicon Premium (TSX CBYxxxK) inferiores a 30m podem ser usados também (distância total 30m)

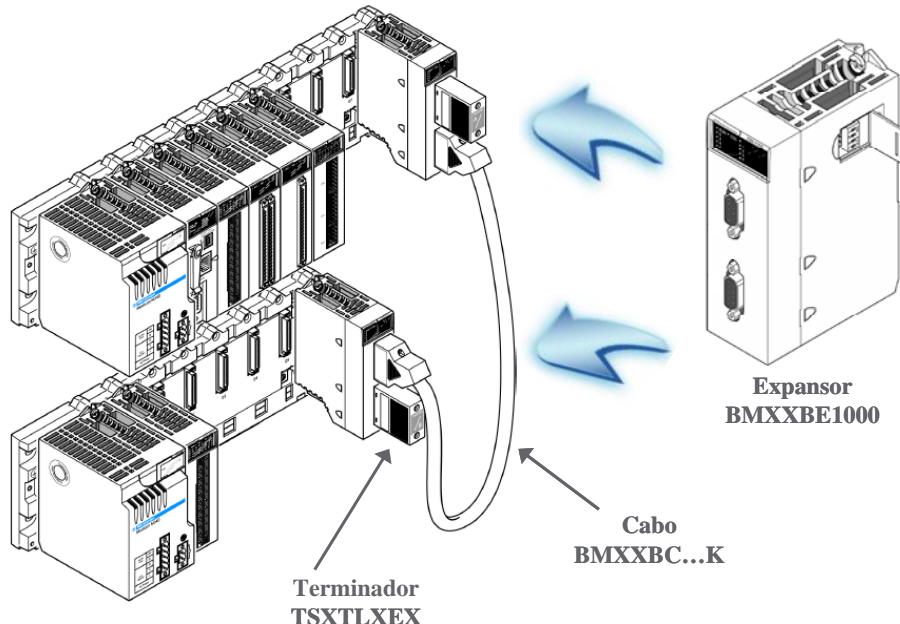
	Length
BMX XBC 008K	0.8 m
BMX XBC 015K	1.5 m
BMX XBC 030K	3 m
BMX XBC 050K	5 m
BMX XBC 120K	12 m

### Opção pelo kit de expansão BMX XBE2005

- Para montar uma configuração mais barata entre 2 racks locais
- Incluindo dois XBE1000, dois TLYEX e um cabo de 0,8m

### Topologia e regras

- Cada rack deve ser fornecido com uma fonte de alimentação BMX CPS xxxx
- O módulo da CPU deve ser instalado no rack 0
- CPU do modelo P341000 pode controlar até 2 racks, enquanto CPUs dos modelos P3420x0 controlam até 4 racks
- Comprimento máximo da rede de ligação entre os racks é de 30 metros
- Em cada extremidade da rede de expansão, o módulo XBE deve ser equipado com 1 terminador de linha (comercializado em pares)



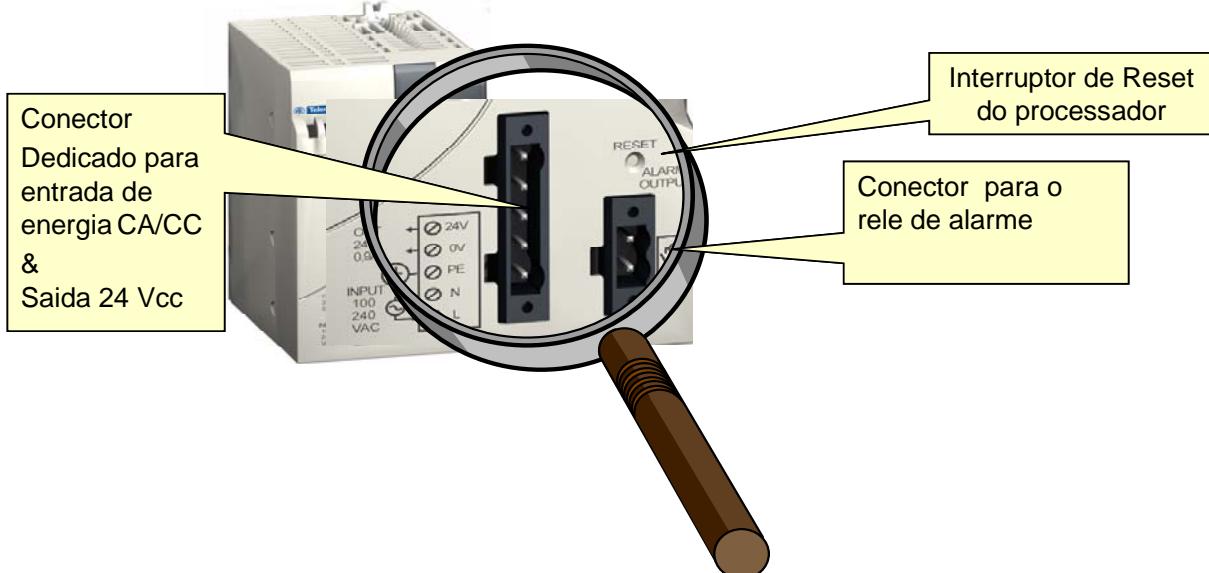
## Alimentação CA/CC



### Modelos Disponíveis:

Características e performance				
Tipos	BMX CPS 2000	BMX CPS 3500	BMX CPS 2010	BMX CPS 3020
Descrição simplificada	20 Watts Alimentação em 115 a 230 Vca	36 Watts Alimentação em 115 a 230 Vca	16 Watts Alimentação em 24 Vdc	31 Watts Alimentação em 24 & 48 Vcc
Proteção	Sobrecarga: Sim curto circuito: Sim sobretenção: Sim			
Saida Maxima 24Vcc para sensor	0.45 Amp.	0.9 Amp.	-	-

## Recursos no Painel Frontal das Fontes

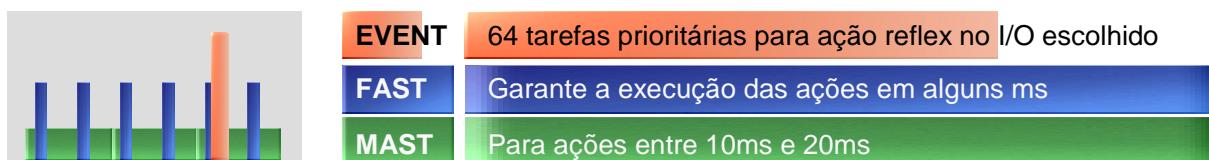


## Unidade Central de Processamento BMX P34 xxxx

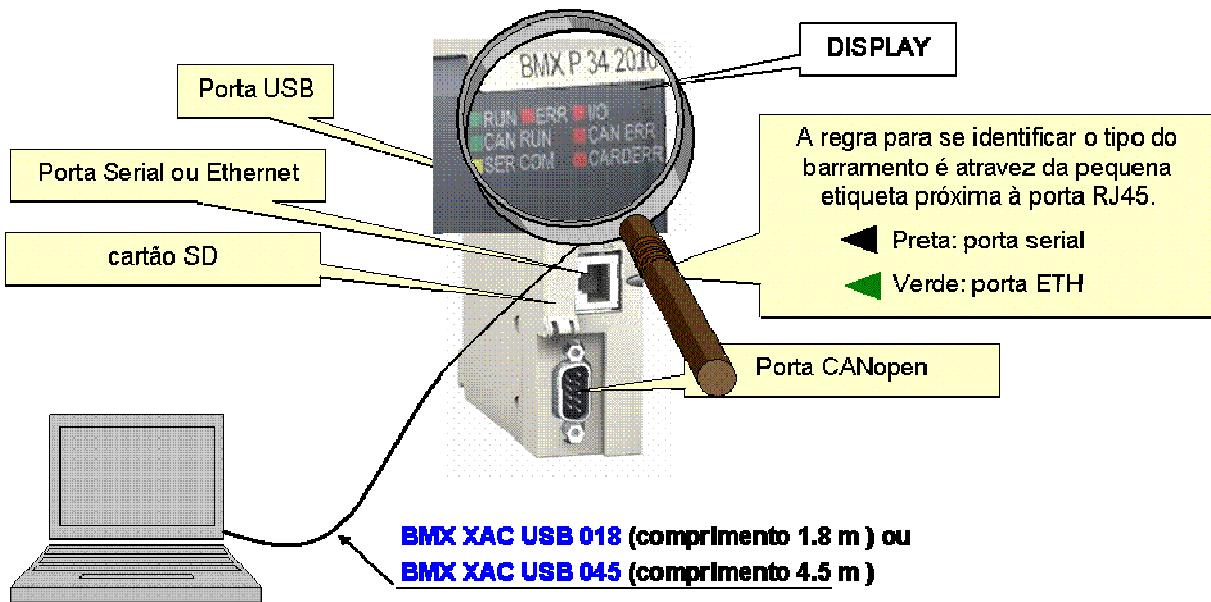
### Características Gerais das CPUs BMX P34 xxxx:

	STANDARD	PERFORMANCE		
Características e performance	BMX P34 1000	BMX P34 2010	BMX P34 2020	BMX P34 2030
Tipos				
Descrição simplificada	Porta USB, Slot de cartão de memória Uma Porta de comunicação: - Serial	Porta USB Slot de cartão de memória Duas portas de comunicação: - Serial - CANopen	Porta USB Slot de cartão de memória Duas portas de comunicação: - Serial - Ethernet	Porta USB Slot de cartão de memória Duas portas de comunicação - CANopen - Ethernet
Número de I/Os digitais	512		1024	
Número de I/Os analógicas	128		256	
Canais Contadores & porta Serial	20		36	
Memória	2 Mb		4 Mb	
Dados de Usuário	128 Kb		256 Kb	

### Operação em sistema Multitarefa



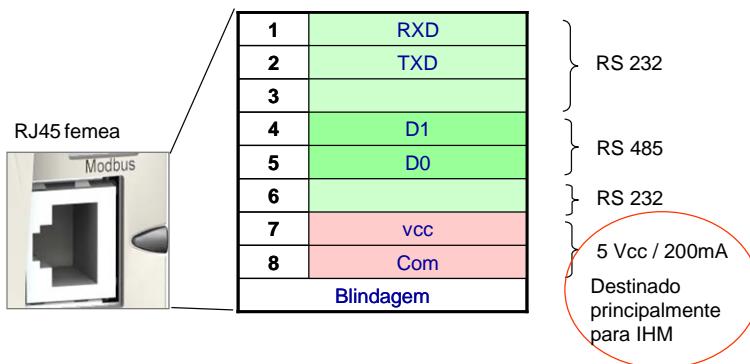
## Detalhamento do Painel Frontal das CPUs BMX P34 xxxx



### Características da porta de comunicação Serial

A porta de comunicação serial integrada permite os protocolos Modbus ( ASCII et RTU) e Character mod

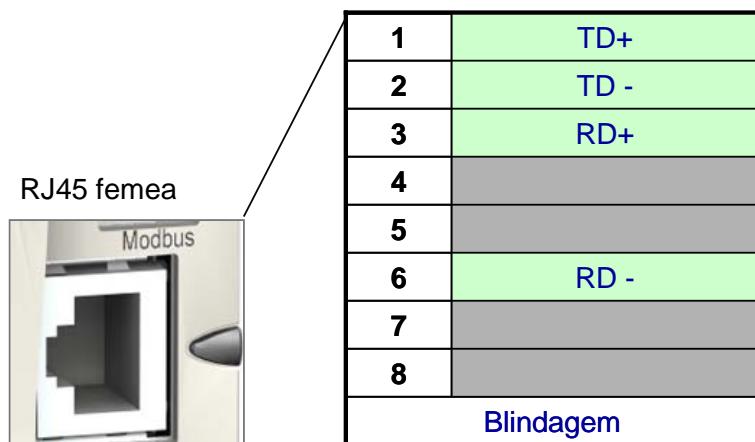
- CPUs BMX P34 1000/2010/2020



### Características da porta de Ethernet

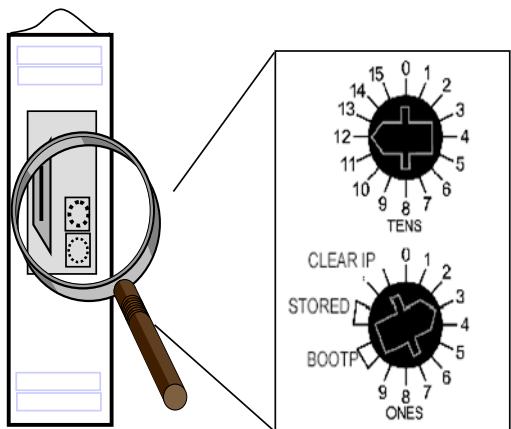
A porta de comunicação Ethernet suporta o Protocolo de Comunicação Ethernet

- CPUs BMX P34 2020/2030



- Endereçamento da porta Ethernet.

Endereçamento: 2 switches usados para “setar” o endereço IP do dispositivo 4 formas possíveis para obter o endereço IP

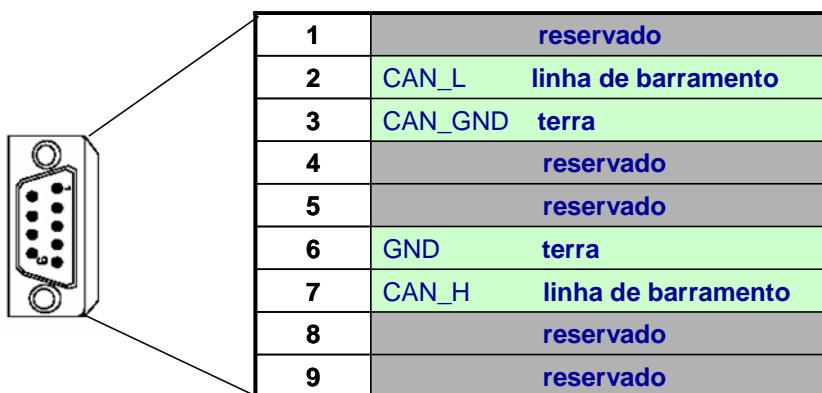


1. Por STORED o dispositivo usa o endereço IP configurado na aplicação.
2. Por BOOTP o dispositivo obtém o endereço IP através de servidor BootP.
3. Por CLEAR IP o dispositivo usa o endereço IP default com base no endereço MAC (84 + os 3 últimos bytes em hexadecimal convertidos em decimal)
4. Tens (dezenas) (xx = 0 a 15) e Ones (Unidades) (y = 0 a 9) dígitos.

### Características da Porta CANopen.

A porta CAN open suporta o protocolo CANopen

- BMX P34 2010/2030



## Cartão de Memória



O MODICON M340 CPU BMX P34 xxx usa o cartão de memória **BMX RMS 008MPx**.

O cartão de memória **BMX RMS 008 MP** é fornecido com cada processador.

O cartão de memória **BMX RMS 008MPx** é formatado para ser utilizado com a serie M340, não sendo possível o uso de um cartão de memória padrão.

### Cartões disponíveis:

Referência do Cartão	Armazenamento da aplicação	Armazenamento de arquivos	Armazenamento de páginas WEB
BMX RMS 008MP	Sim	Não	Sim
BMX RMS 008MPF	Sim	8 Mb	Sim
BMX RMS 128MPF	Sim	128 Mb	Sim

### Standard memory cards

Description	Processor compatibility	Capacity	Reference	Weight kg
Flash memory cards (1)	BMX P34 2020H BMX P34 20302H	8 Mb / 8 Mb files 8 Mb / 128 Mb files	BMX RMS 008MPF BMX RMS 128MPF	0.002 0.002

### Standard replacement parts

Description	Use	Processor compatibility	Reference	Weight kg
Flash memory card 8 Mb	Supplied as standard with each processor, used for: - Backup of program, constants, symbol and data - Activation of class B10 web server	BMX P34 2020H BMX P34 20302H	BMX RMS 008MP	0.002

A tabela seguinte apresenta a compatibilidade dos cartões de memória:

BMX RMS 008MP >> Cartão compatível com todos os processadores.

BMX RMS 008MPF e BMX RMS 128MPF >> Cartão compatível com os seguintes processadores:

- BMX P34 2000,
- BMX P34 2010,
- BMX P34 2020,
- BMX P34 2030.

**Função do cartão de memória:**

O cartão de memória realiza automaticamente o download de aplicações dentro do processador na ligação do mesmo.

Com o bit de sistema %66 é possível forçar a transferência entre o processador e o cartão de memória, caso seja colocado o cartão de memória no processador sem desligar, este cartão será carregado pela aplicação.

**O programa é salvo no cartão de memória de duas maneiras:****1 - Automaticamente, Após:**

**Um download:** Se o cartão de memória estiver presente e não protegido contra gravação.

**Alteração OnLine:** Se o cartão de memória estiver presente e não protegido contra gravação.

**Detecção da borda de subida do bit de sistema %S66.**

**2 - Manualmente, Com o comando PLC ® Project backup ® Backup Save.**

**Obs.:** Se for retirado o cartão de memória enquanto o backup estiver em progresso, o programa no cartão de memória será perdido.

**O programa é copiado do Cartão de Memória para a memória interna de duas maneiras:****1 - Automaticamente após:**

Uma partida a quente, warm start;

Uma partida a frio, cold start.

**2 - Manualmente, Com o comando do Unity Pro para o PLC restaurar o backup do projeto (Unity Pro command PLC Project backup Backup Restore).**

NOTA

Ao inserir o cartão de memória no modo Run ou Stop, deverá ser feito o clico energização para restaurar o projeto no CLP.

# CAPÍTULO 4

## **Apresentação e Descrição do Software Unity**

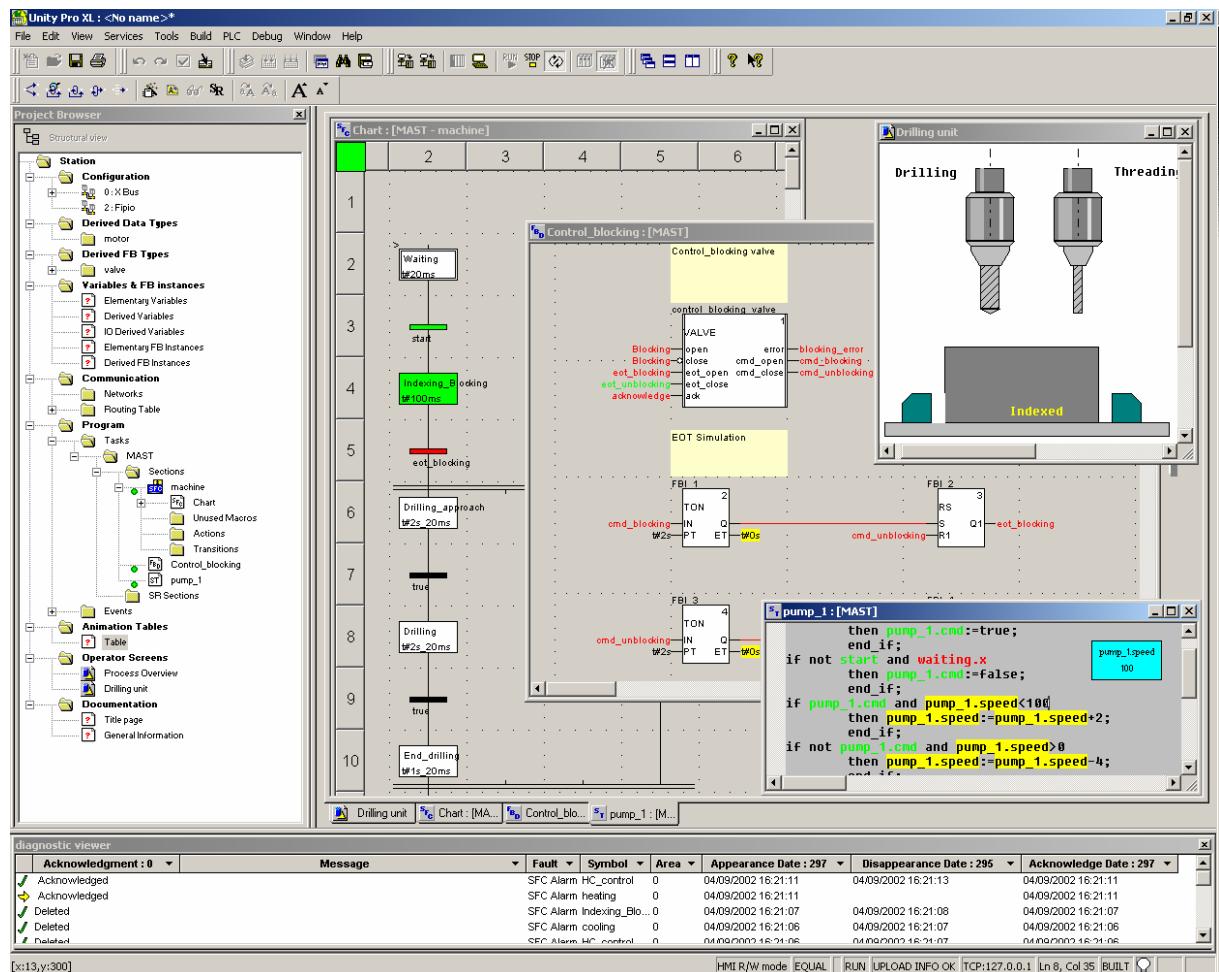
---



# Apresentação e Descrição do Software Unity

## Unity Pro

O **Unity Pro** é uma ferramenta de programação dos CLPs Quantum, Premium, Atrium e M340. Com base na experiência Schneider Electric em software de automação foram reutilizados os melhores recursos dos softwares de programação PL7 e Concept criando novas funcionalidades fornecendo todos os passos para ter uma aplicação mais rápida.



Desenvolvido para atender a norma IEC 61131-3 e C ++ aberto, possibilitando uma programação complexa, porém de forma simplificada e amigável, permitindo a utilização das cinco linguagens de programação descritas pela referida norma:

**LD – Ladder – Linguagem Ladder**

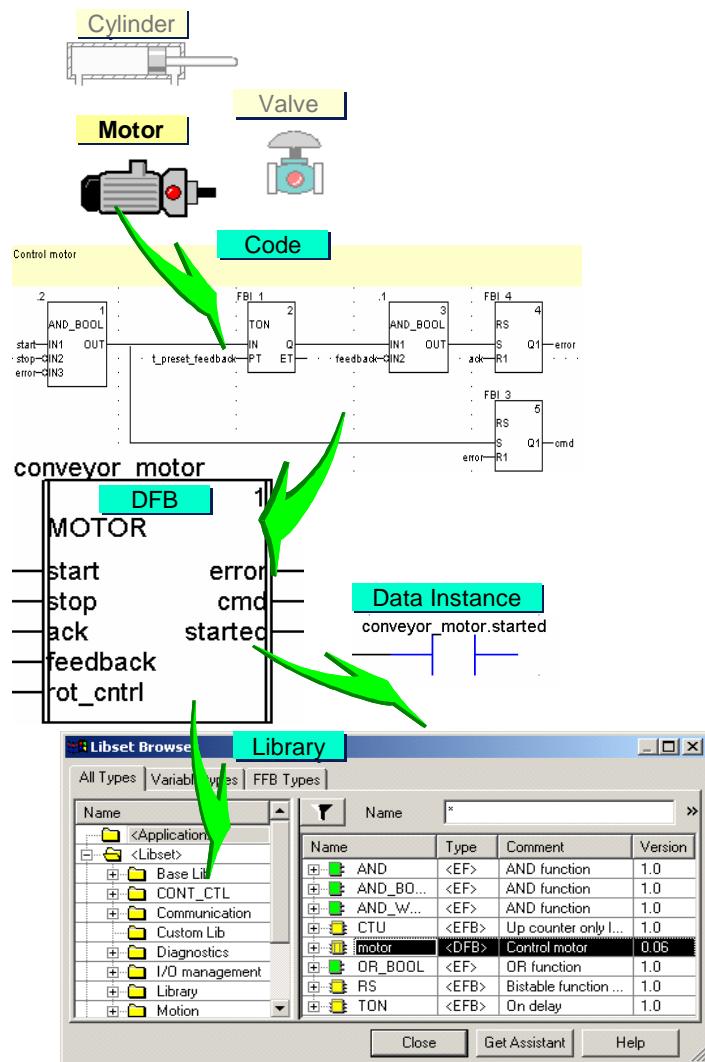
**FBD – Function Blocks Diagram – Diagrama de Blocos Funções**

**SFC – Sequential Function Chart – Gráfico de Seqüência Funcional**

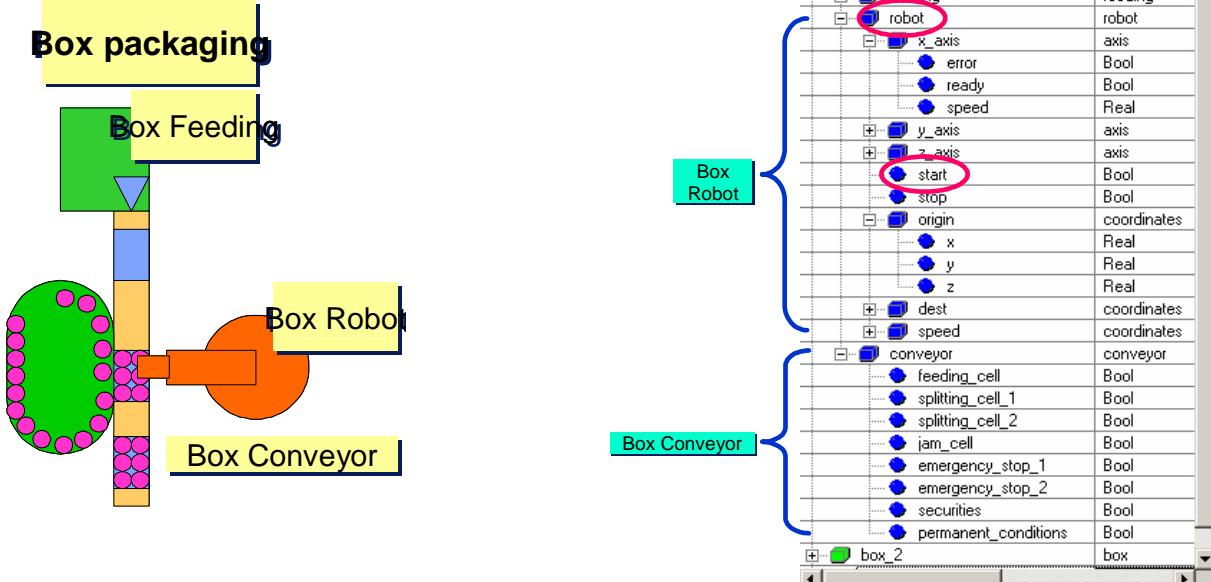
**IL – Instruction List – Lista de Instrução**

**ST – Structure Text – Texto Estruturado**

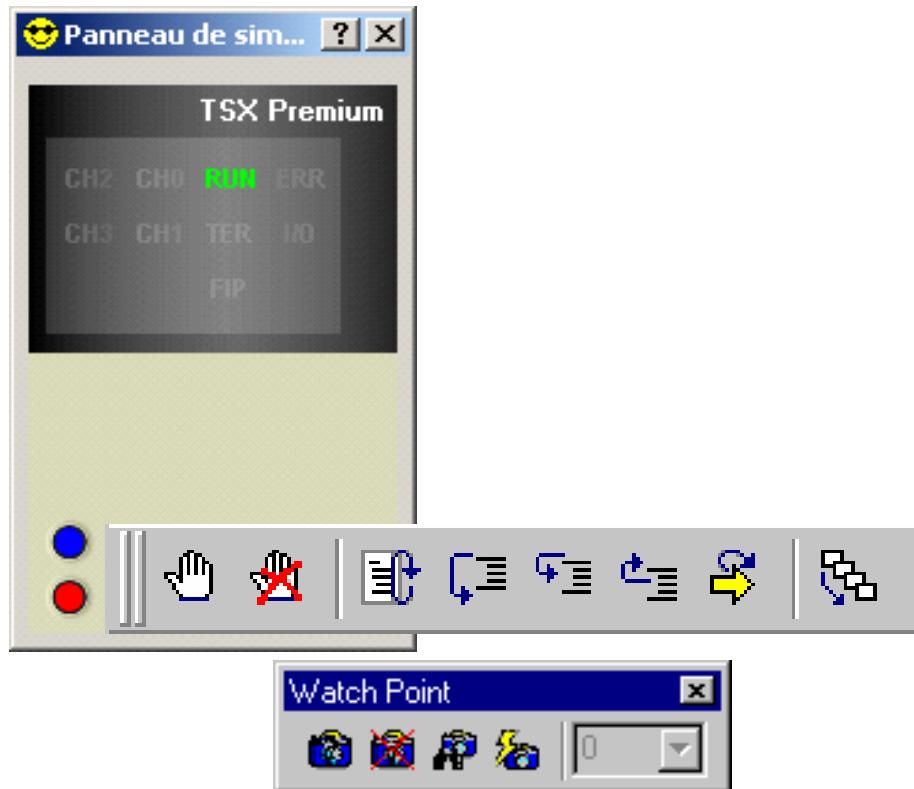
O **Unity Pro** garante uma padronização das funções usadas no processo.



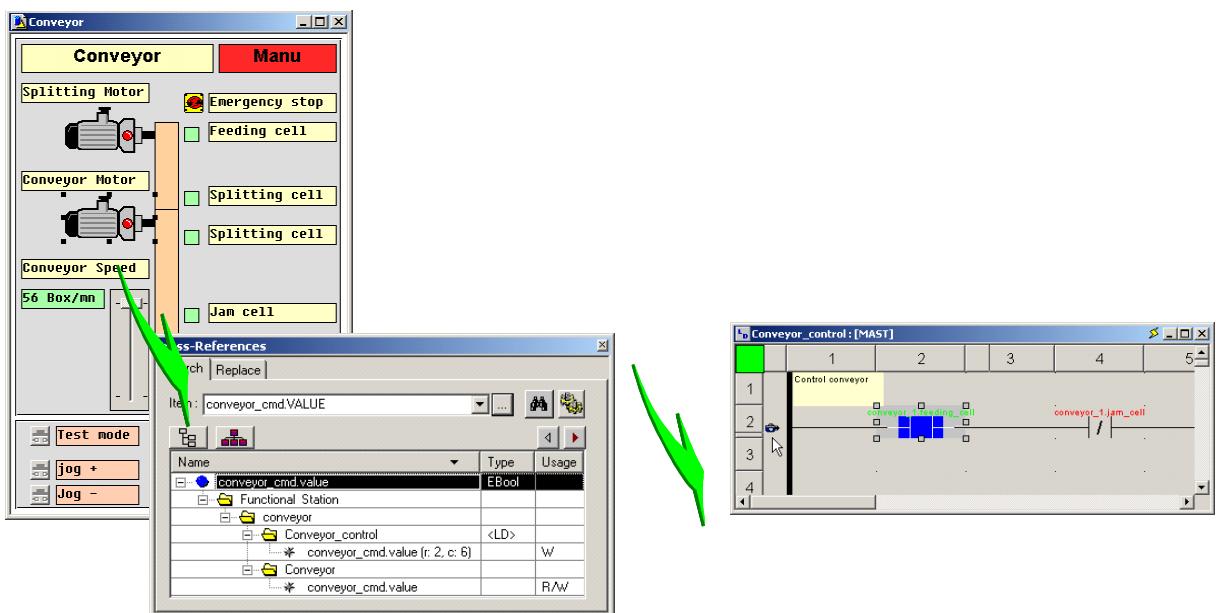
O Unity Pro customiza os dados necessários ao processo



O **Unity Pro** possui o recurso de **Simulação** permitindo a verificação do funcionamento e depuração da aplicação continuamente ou em passos reduzindo assim tempo de “startup” do processo.



O **Unity Pro** possui o recurso de tela animada denominada **“Operator Screens”** que permite a visualização e controle da aplicação através de gráficos e objetos animados.



## Características Gerais do Unity Pro

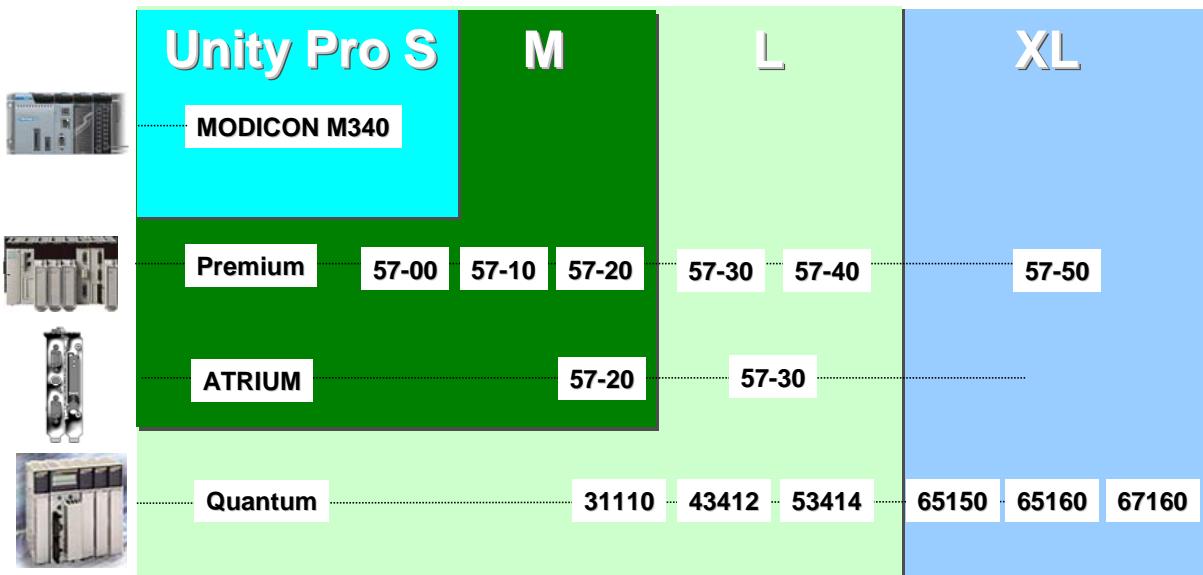
### Introdução

O Unity Pro é a Ferramenta certa para todas as fases de um projeto de automação:



"Tudo em um" software para programar e depurar uma aplicação completa;  
 5 linguagens de programação IEC, multi-tarefas, dados estruturados, biblioteca de funções;  
 Simulador do PLC, Telas de operação e funções de depuração completa;  
 Múltiplas modificações on-line e telas de diagnóstico;  
 Migração completa de aplicações com o Concept e PL7.

## Guia de seleção



## Grau de aplicação:

Simples & Multi-postos	Acima de 3 ..... 10 ..... 100 usuários			
	↓	↓	↓	↓
	Simples	Grupos	Equipes	Facilidades
<b>Unity Pro X Large</b>	✓	✓	✓	✓
<b>Unity Pro Large</b>	✓	✓	✓	✓
<b>Unity Pro Medium</b>	✓	✓	✓	
<b>Unity Pro Small</b>	✓	✓	✓	

O Unity Pro Small acompanha a mesma política das demais versões:

Atualizações

- Facilidade de transição do PL7 Micro, Concept Small e ProWorx Lite  
Somente para clientes com registro ativos

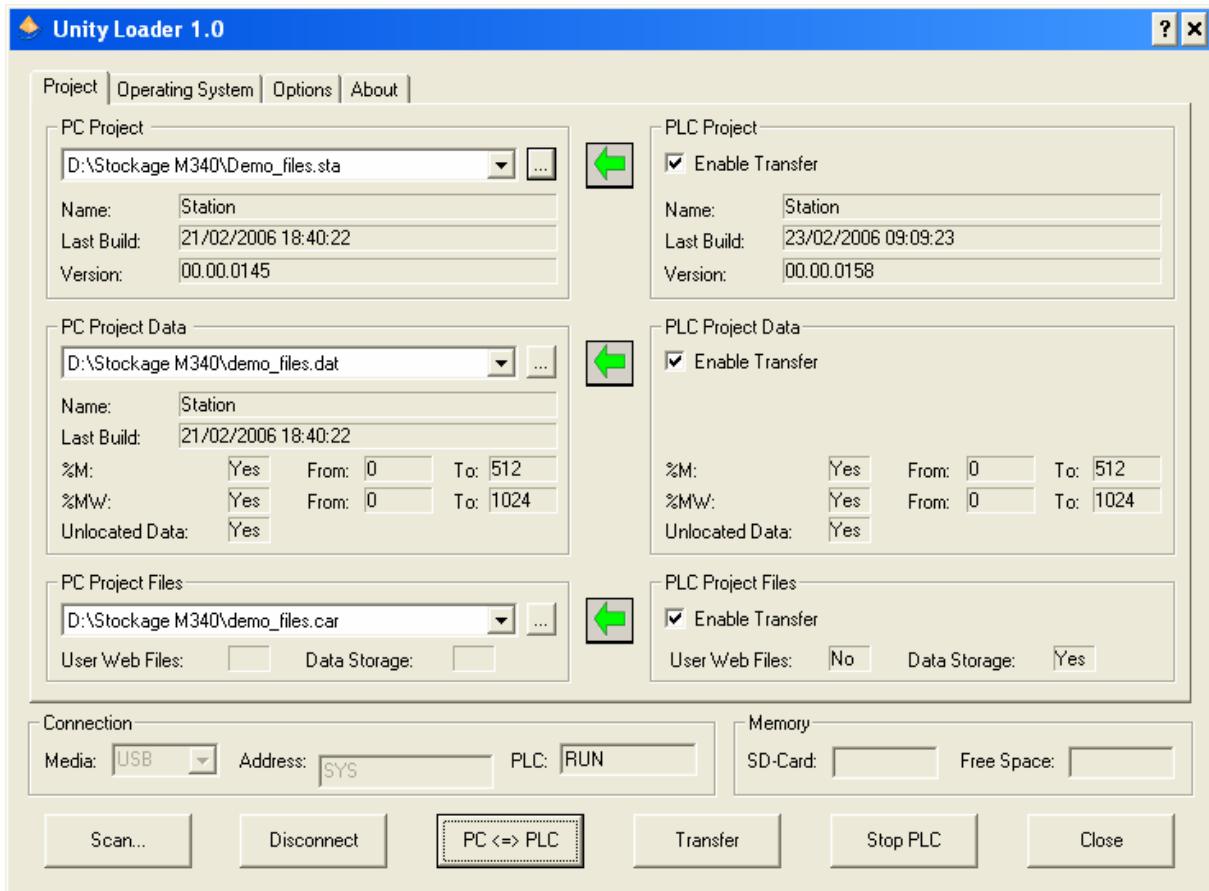
## Unity Loader

É um software independente que permite o carregamento da Unidade para operação e manutenção com os seguintes recursos.

1 - Backup/restore de projetos:

- Programa
- Dados em tempo real
- Armazenamento de dados através de cartucho

## 2 - Atualização de Firmware.



## Funcionalidades

### O Unity Pro apresenta forte estrutura de aplicação.

Project settings;  
Project browser com visualização funcional/estruturada;  
Biblioteca de funções (EF/EFB/DFB);  
Editor de variáveis (estruturadas, alocada/não alocada);  
Tarefas simples ou multi-tarefas;  
Único arquivo de aplicação (.STU).

### Programação flexível

Independente da configuração de hardware;  
Instruções genéricas (compatível com o Premium e plataforma Quantum);  
5 linguagens de programação IEC.

### Configuração gráfica de módulos de I/O e módulos especiais

### Comunicação

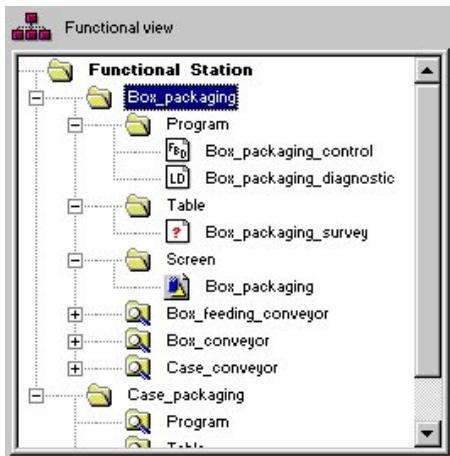
Rede lógica (somente Ethernet, Modbus Plus, FIPWAY) dependendo do hardware;  
Módulos de comunicação específicos.

### Depuração e ajuste fino

Tela de operação e visualização de diagnóstico;  
Simulador do PLC;

Modificação on-line;  
Visualização do consumo da fonte e do uso de memória.

- **Visualização funcional**

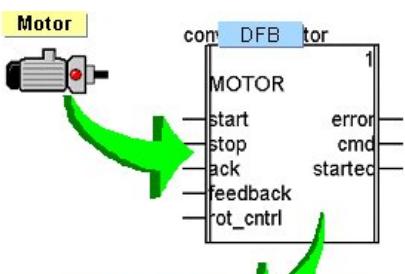


- Mapa da aplicação;
- Acesso direto a uma parte do programa utilizando a estrutura da aplicação.

## Editor de variáveis

Mapeamento dos objetos estruturados

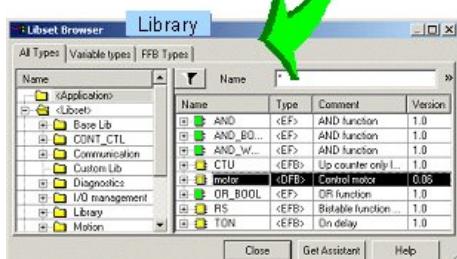
- Visão simples;
- Acesso rápido para os dados usando o nome da estrutura e ampliando os dados;
- Programa automaticamente documentado;
- Se alterado uma estrutura, todas as instâncias são atualizadas automaticamente;
- A declaração de uma instância cria todas as variáveis.



Name	Type
box	box
+ feeding	feeding
robot	robot
x_axis	axis
error	Bool
ready	Bool
speed	Real
y_axis	axis
z_axis	axis
start	Bool
stop	Bool
origin	coordinates
x	Real
y	Real
z	Real
dest	coordinates
speed	coordinates
conveyor	conveyor
feeding_cell	Bool
splitting_cell_1	Bool
splitting_cell_2	Bool
item_cell	Bool
emergency_stop_1	Bool
emergency_stop_2	Bool
securities	Bool
permanent_conditions	Bool
box_2	box

## Biblioteca de funções

Criação e administração de biblioteca de funções próprias do usuário.



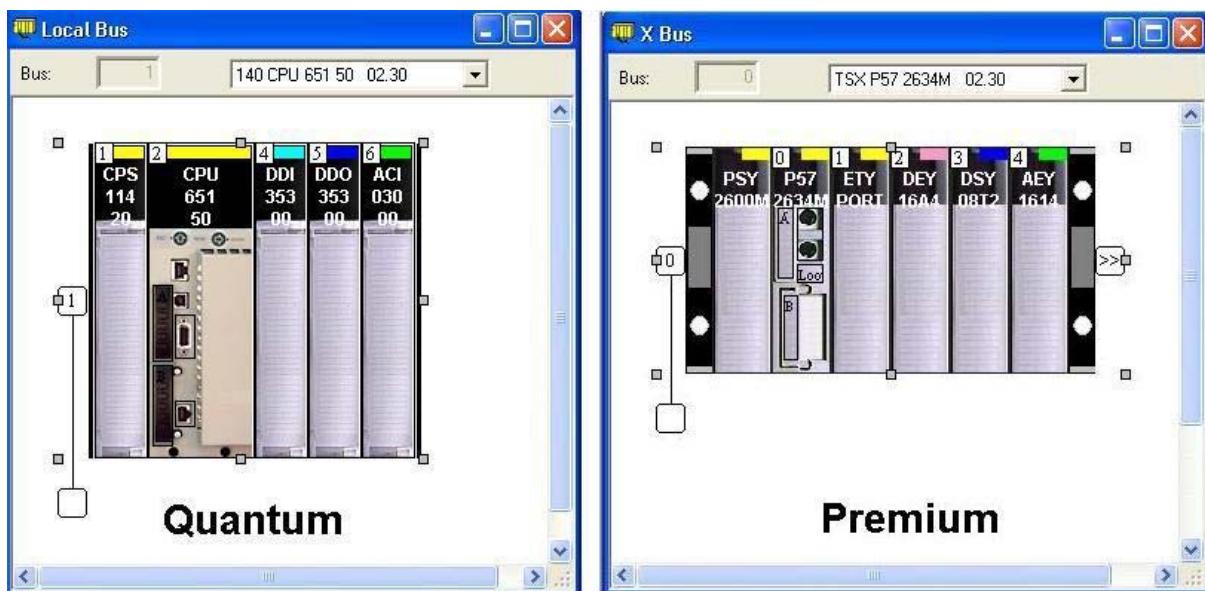
Cada função elementar desenvolvida dentro de um Bloco de Função derivado (DFB)

- Somente um código para desenvolver e administrar;
- Padronização e utilização de componentes elementares.

- O bloco DFB desenvolvido é inserido na biblioteca do software e pode ser utilizado em diferentes projetos
- Fácil de administrar;
- Fácil de usar.

## Ferramentas de configuração

Procedimento de configuração idêntico aos CLPs Quantum, Premium e M340.



### Recursos:

- Configurar o hardware e ajustar os parâmetros de cada módulo;
- Uma ferramenta para configuração e diagnóstico de módulo de I/O e CPU;
- Acesso direto (on-line) para depuração e diagnóstico do processador e dos módulos;
- Descrição gráfica da configuração do hardware;
- Copiar, colar e mover os módulos de I/O;
- Telas dedicadas para configurar os módulos de I/O e entrar nos parâmetros associados.
- Programa e configuração do hardware independente;
- Não é possível trocar uma aplicação com a CPU do Premium por uma com a CPU do Quantum, ou vice-versa;
- Você pode exportar e importar porém sem a configuração de hardware.

### Simulador

- Apresenta o recurso de depuração e ajuste fino do projeto, permitindo a redução de tempo de startup do processo.
- Teste do programa sem o hardware
  - Simulador com as mesmas características de um PLC real;
  - Tabela animada para ajustes, forçar variáveis.

- Simplifica a depuração completa usando a ferramenta completa:
  - Telas gráficas;
  - Visualização de diagnóstico.

### Painel Frontal do Simulador

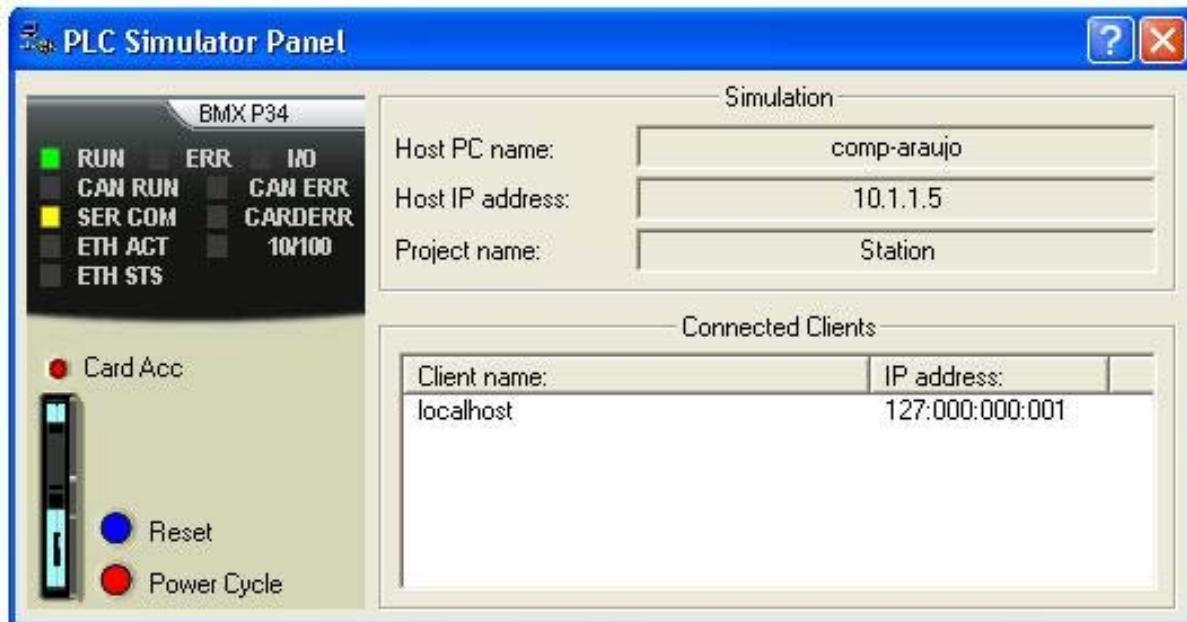
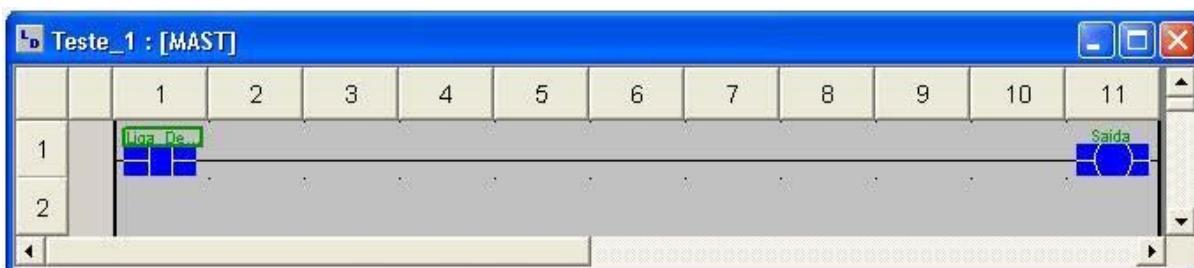


Tabela animada, função Force e modificação de valores.

Table[LD Editor - Teste_1 : [MAST]]				
Modification		Force		
Name		Value	Type	Comment
Liga_Desliga		F1	EBOOL	Chave Liga/Desliga
Saida		1	EBOOL	Contactor 1 - Motor

Verificação Online da lógica.



# **CAPÍTULO 5**

## **Ajustes do Projeto**

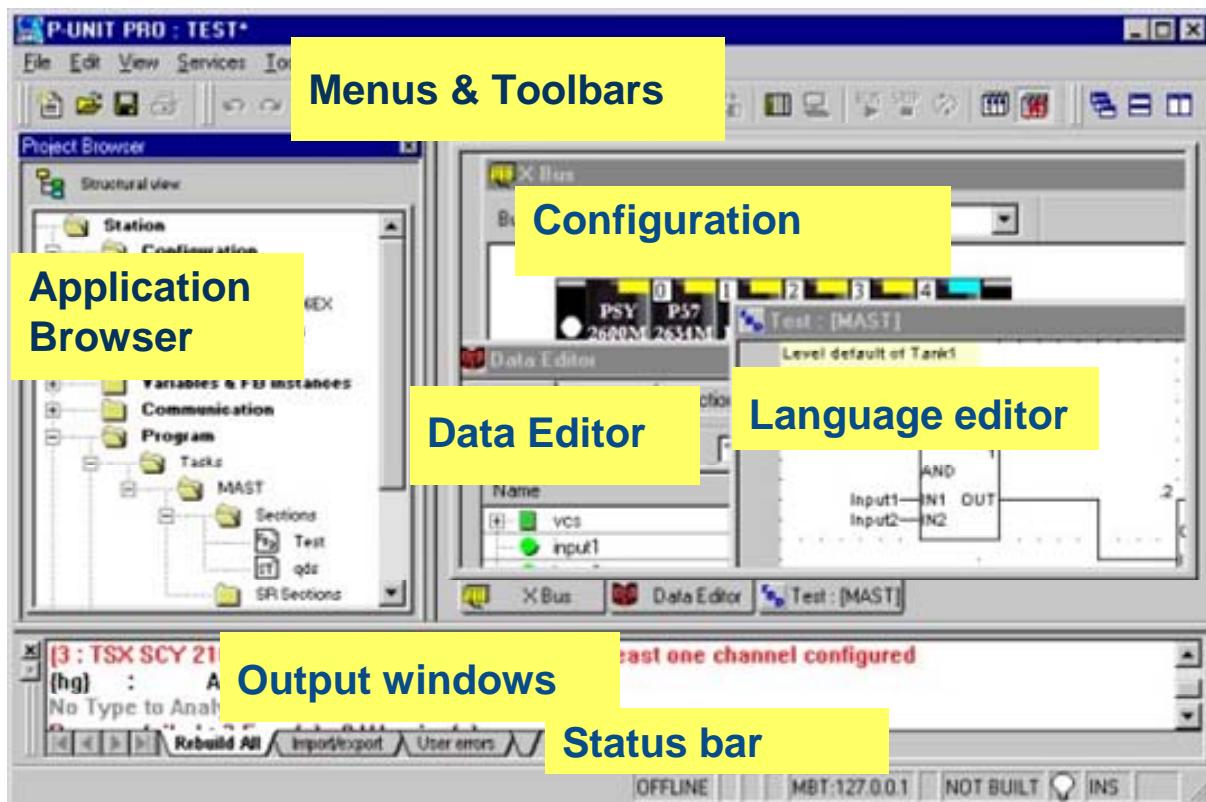
---



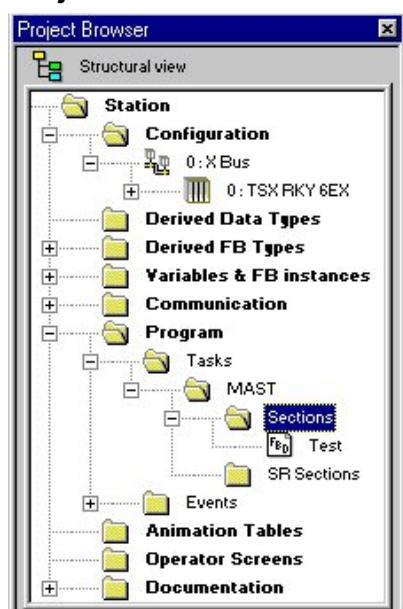
# Ajustes do Projeto

## Visão Geral da Interface

A interface com o usuário é composta por várias janelas de configuração e barra de ferramentas.

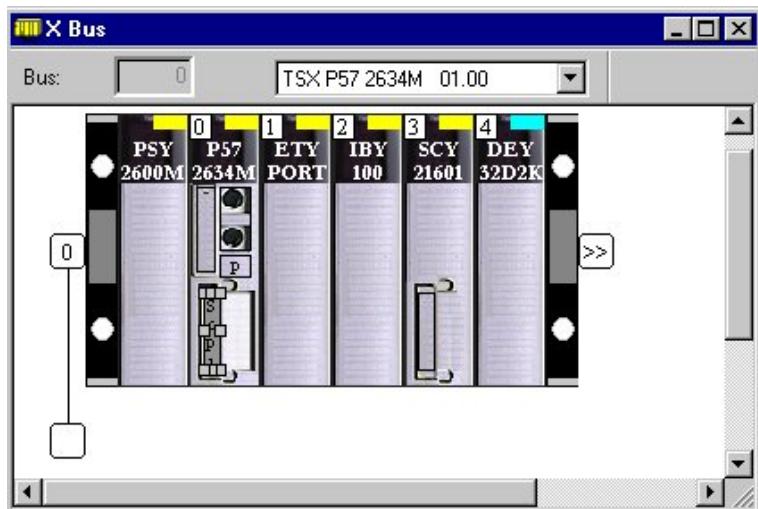


### Project Browser



O Project Browser permite visualizar todo o conteúdo de um projeto do Unity Pro separado por pastas onde:

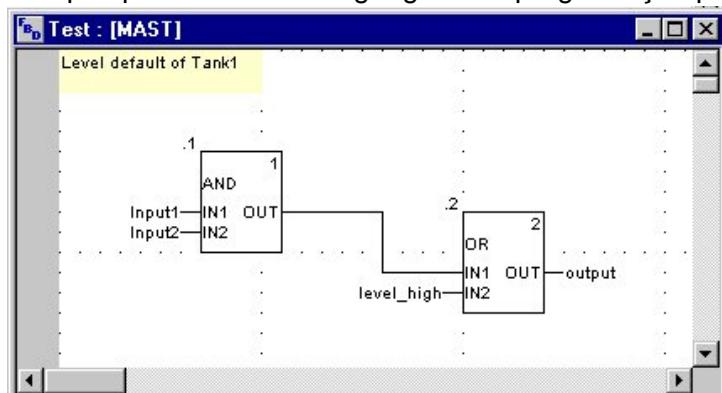
### Configuration Editor



O **Configuration Editor** é a pasta que permite o acesso a área de configuração do hardware e parametrização dos módulos.

### Language editor

O **Language editor** é o editor da linguagem utilizado para o desenvolvimento do programa em qualquer uma das 5 linguagens de programação padronizadas pela IEC 1131-3:



- Ladder;
- Function Block Diagram (Diagrama de Blocos de Funções);
- Sequential Flow Chart (Seqüenciamento Gráfico de Funções);
- Structured Text (Texto Estruturado);
- Instruction List (Lista de Instruções).

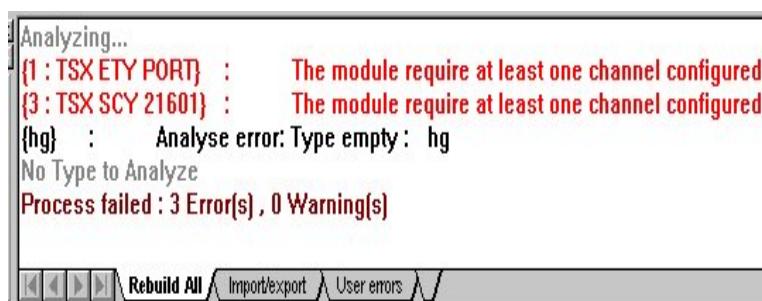
### Data Editor

Data Editor			
Variables   DDT types   Function Blocks   DFB types			
<input type="checkbox"/> Filter <input type="text"/> Name <input checked="" type="checkbox"/> Elementary >			
Name	Type	Address	Va
+ vcs	T_DIS_IN_STD		
input1	Bool		
input2	Bool		
Output	Bool		
level_high	Bool		

O **Data Editor** é a área utilizada para todo o gerenciamento de dados (variáveis) que serão utilizadas no projeto. No Editor de Dados é possível:

- Criar Tipos de Dados;
- Declarar variáveis Elementares;
- Declarar variáveis Estruturadas;
- Declarar Blocos de Funções;
- Criar Blocos DFBs (Derived Function Blocks).

### Output Windows



O **Output Windows** é uma ferramenta utilizada para informar ao usuário sobre erros existentes no programa. Toda vez que o programa for construído ("Build"), e possuir algum erro, este será indicado no Output Windows, ao clicar duas vezes sobre o erro o software indica onde está o mesmo.

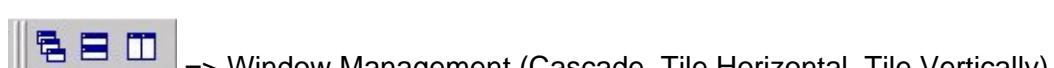
### Status Bar

O **Status Bar** nos mostra uma variedade de informações sobre a operação do software. Ex. Off/Online.



### Barra de Ferramentas Comuns

Todas as funções podem ser acessadas pela barra de ferramentas de menu. As funções freqüentemente usadas podem ser acessadas diretamente pelos ícones existentes na barra de ferramentas, é possível ainda personalizar a barra de ferramenta de acordo com as necessidades do usuário



## Iniciando um novo Projeto

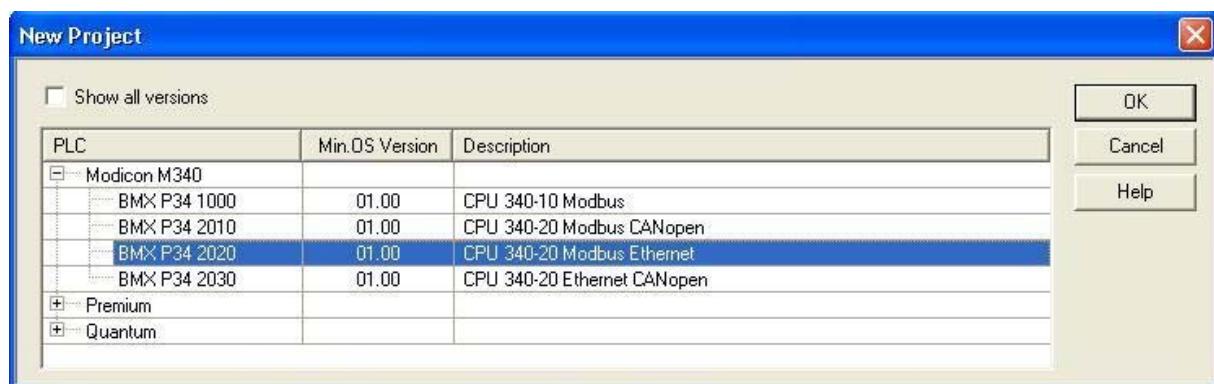


Selecione Unity Pro no grupo de programas Schneider Electric.

O software será aberto e após isto clicar no ícone indicado na figura abaixo para criar um novo projeto.



Após clicar para criar um novo projeto aparecerá a tela abaixo onde deve ser selecionado qual a plataforma do PLC que será utilizado neste projeto (Modicon M340, Premium ou Quantum) assim como o modelo da CPU.



## Ajustes da estação de trabalho - Options

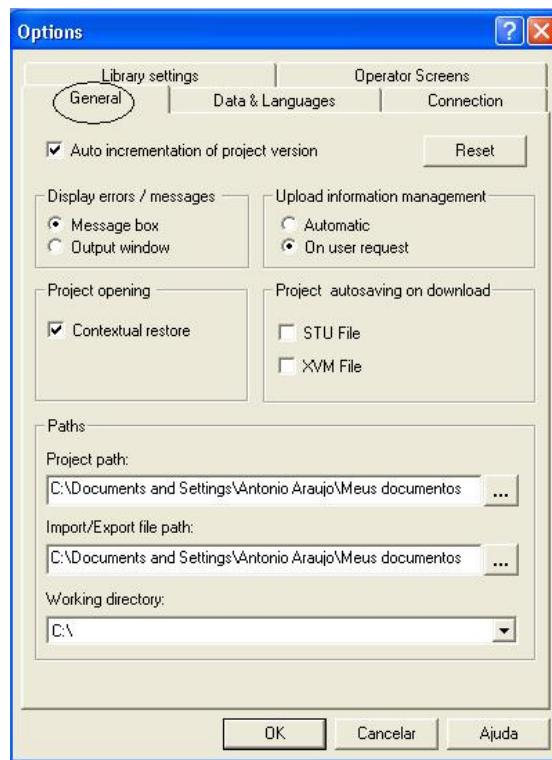
Os ajustes da estação de trabalho podem ser acessados pelo comando:

**Tools => Options.**

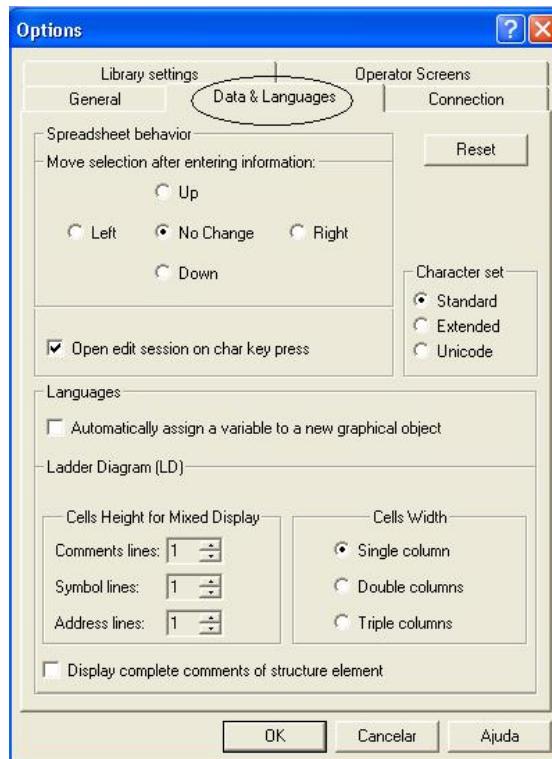
### General:

Versão do projeto, upload e erros.

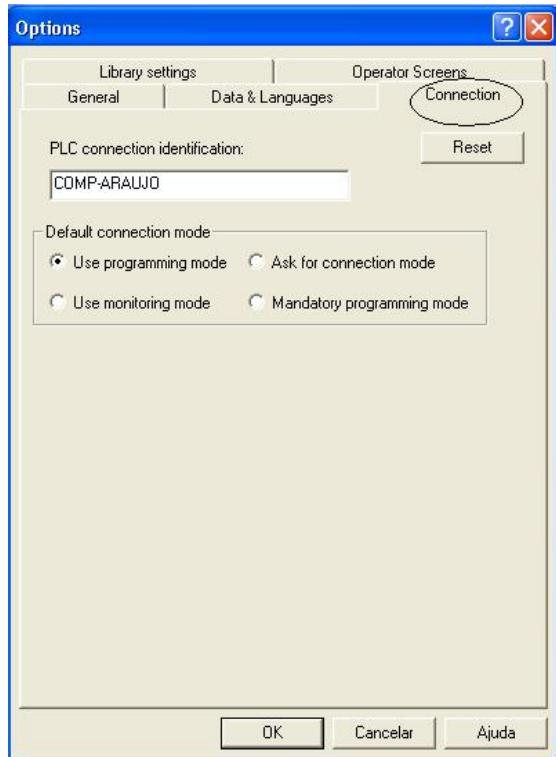
O diretório de trabalho do Unity Pro (padrão).

**Data & Languages:**

Ajustes padrões para entrada de dados e linguagem.

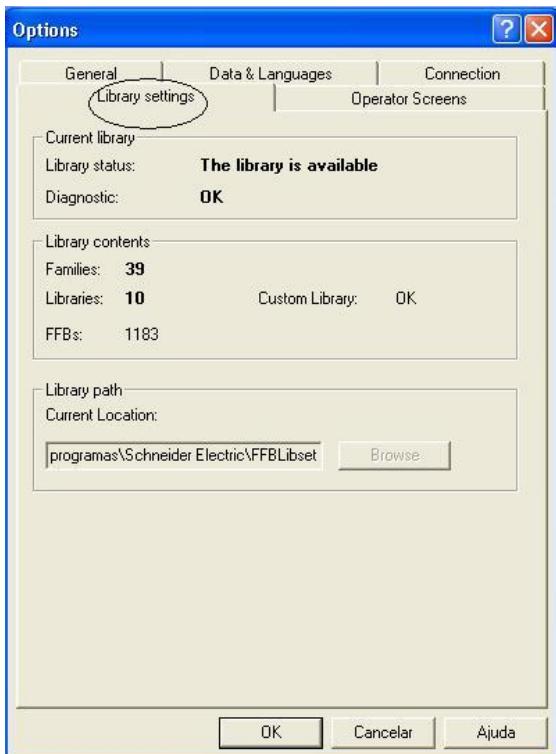
**Connection**

Contém as opções relacionadas à conexão do terminal ao PLC.



## Library settings

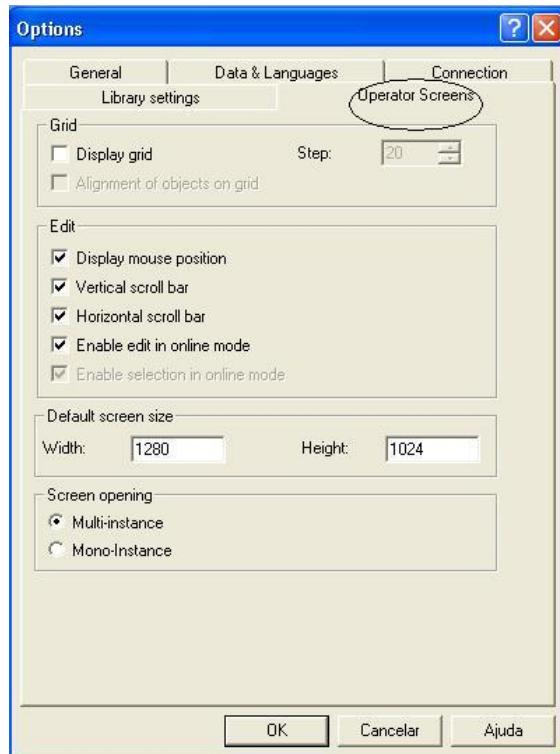
Informações sobre a biblioteca de funções (versão, locação).



## Operator screens

Definir o espaço de trabalho e o tamanho da tela para edição do Operator Screens.  
Habilitar modificações on-line.

## Ajustes do Projeto



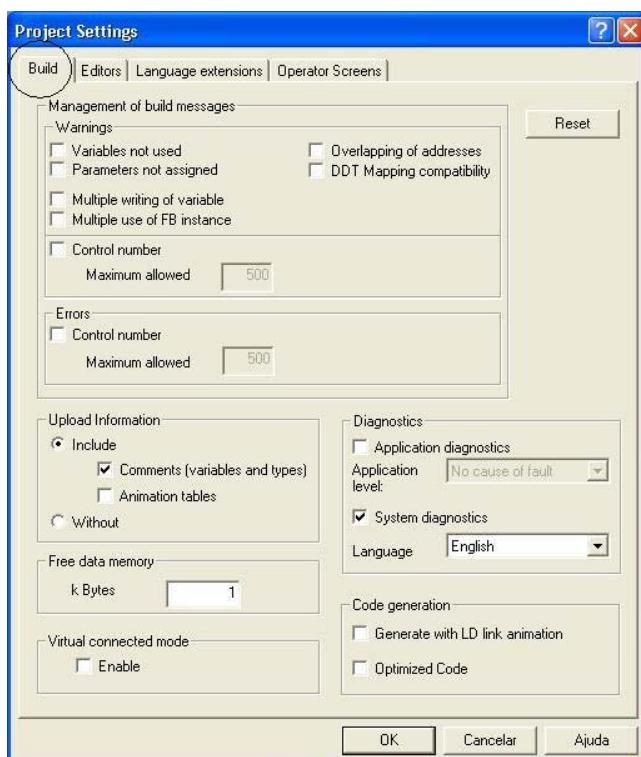
## *Project Settings*

Por serem ajustes do projeto, estes são salvos na aplicação.  
O Project Settings somente pode ser acessível via comando:

### Tools => Project Settings

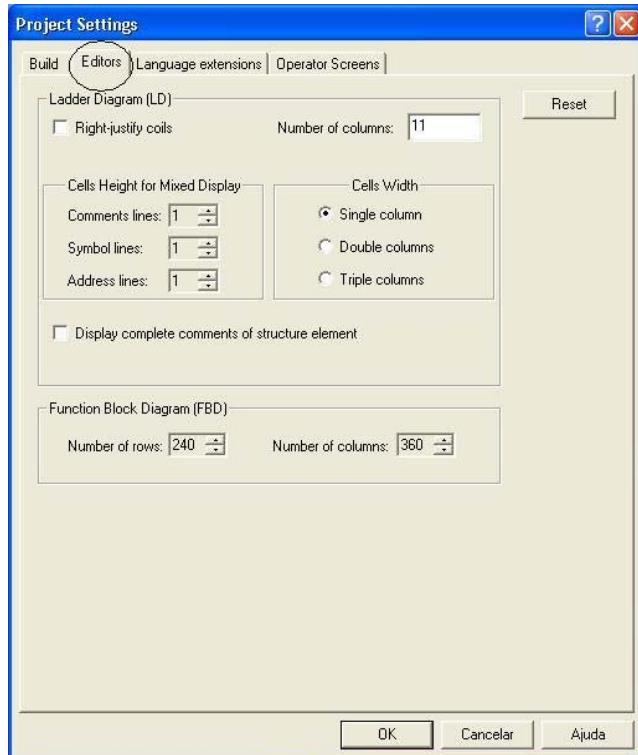
#### Build

Configura a geração do Projeto (analyse, code generation, diagnostic,...)



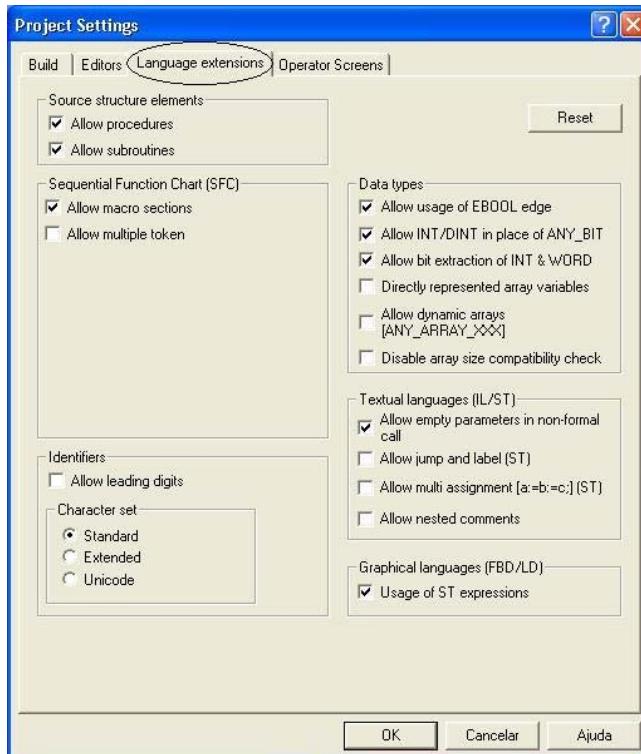
#### Editors

Define os recursos do editor Ladder



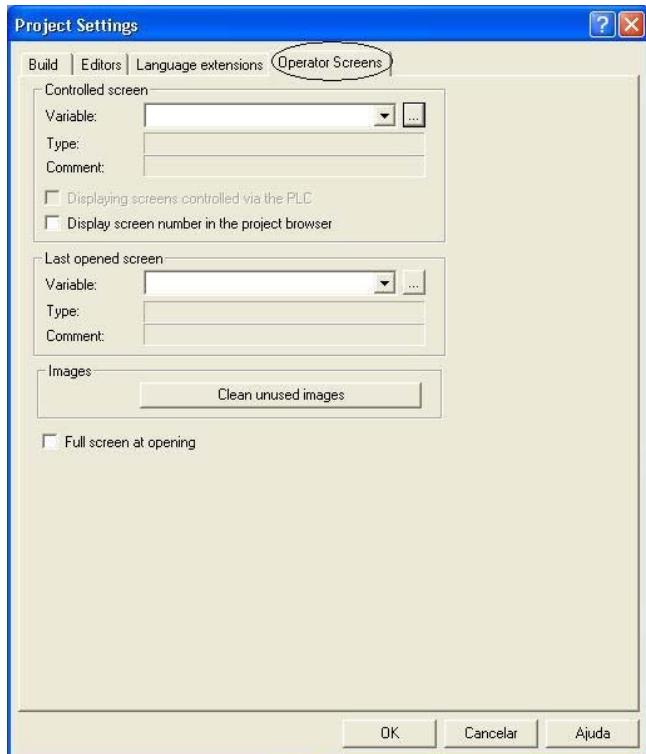
## Language extensions

Contém os ajustes para habilitar as expansões IEC 61131-3 (data type, programming language, structure) para uma aplicação compatível com IEC.



## Operator screens

Permite o gerenciamento da tela (Screen) pelo operador  
A detecção pelo CLP da última tela visualizada.

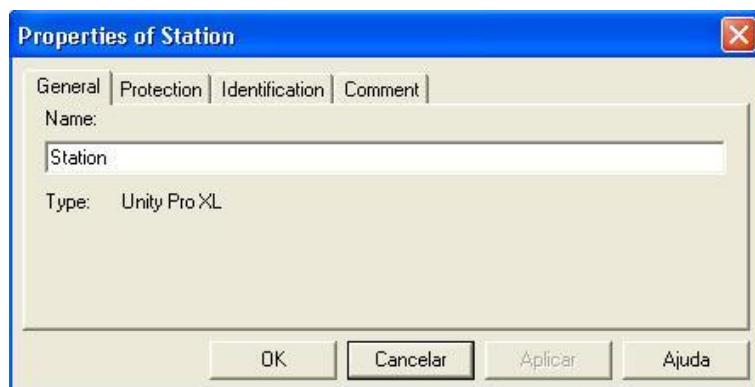


## *Project browser*

Apresenta o conteúdo do projeto em formato de estrutura de árvore e permite o movimento em torno de vários elementos: configuração, dados, programa etc.

## **Propriedades do Projeto**

As propriedades do projeto são acessíveis com um clique com botão direito na pasta station.



Aba **General**: define o nome do projeto;

Aba **Protection**: ativa a proteção das seções do programa;

- Alteração ou apagamento da senha



A alteração não pode ser ativada se não houver senha.

Aba **Identification**: identificação do projeto

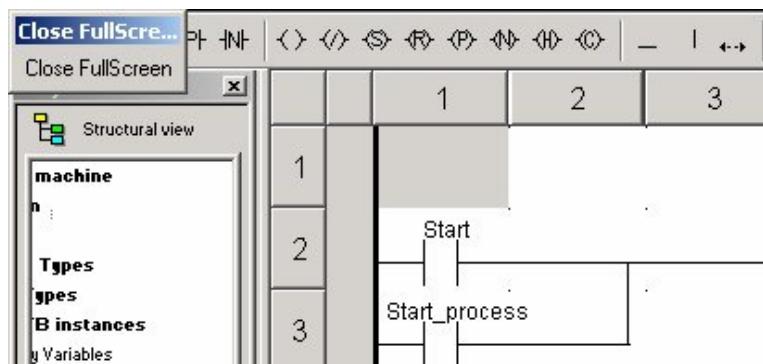
- Versão corrente com a opção de incremento automático
- Data de criação e geração

Aba Comment: Associa um comentário com o projeto.

## Ergonomia

Acessado através da barra de ferramentas View => Full screen

Todo o editor pode ser alterado para modo normal ou modo tela cheia.  
Expandir o espaço de edição.



# CAPÍTULO 6

## **Configuração de Hardware**

---

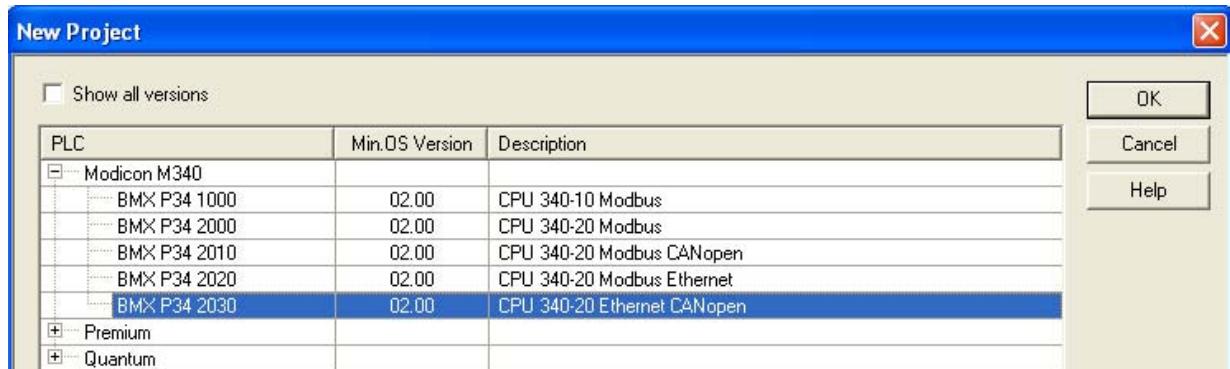


# Configuração de Hardware

## Escolha o Processador

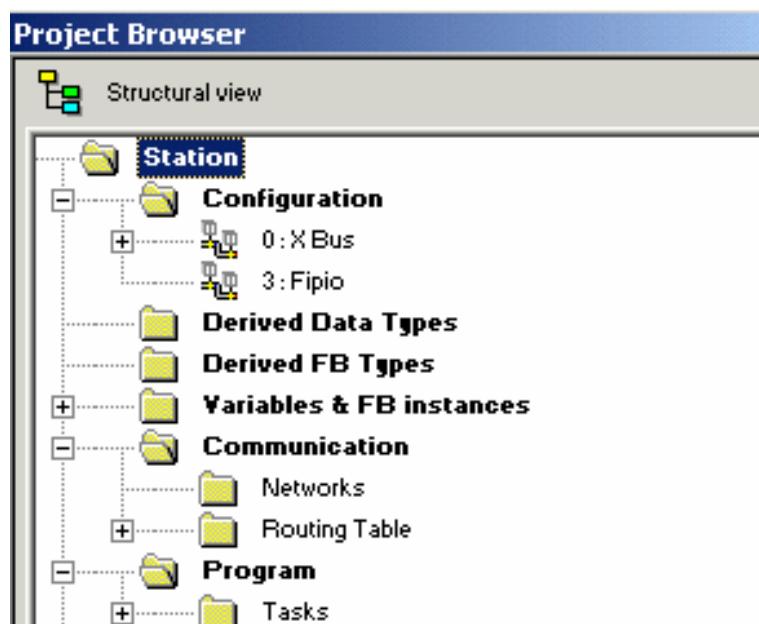
O primeiro passo para criar uma aplicação é escolher a plataforma; Modicon M340, Premium ou Quantum (não intercambiável) e do processador.

O acesso é feito através de File =>New na barra de ferramentas.



## Editor de Configuração

O Editor de Configuração é acessível a partir do “Project Browser” com vista estrutural.



O Editor de configuração é usado para:

Configurar os racks do barramento local:

Endereço 0 para os CLPs Premium e Modicon M340.

Endereço 1 para os CLPs Quantum.

Configurar os racks remotos (X Bus ou remote I/O)

Configurar o barramento de campo.

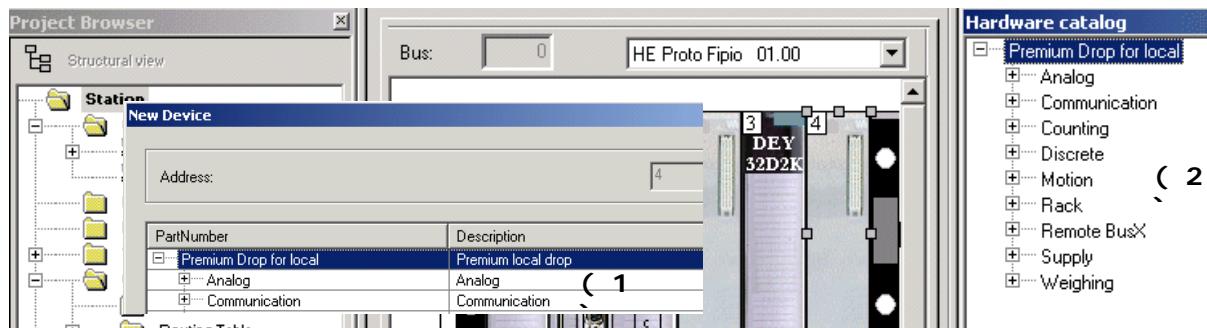
Link de hardware de comunicação com a rede lógica (Ethernet, Modbus +, Fipway)

## Configuração do Rack

O rack é configurado nas seguintes etapas:

1. Escolher o Rack
2. Definir a fonte de alimentação (posição esquerda)
3. Troque o processador se necessário
4. Defina os módulos:

Com duplo clique na posição em vazia do rack e adicione o dispositivo (1) ou arraste e cole do catálogo de hardware (2).

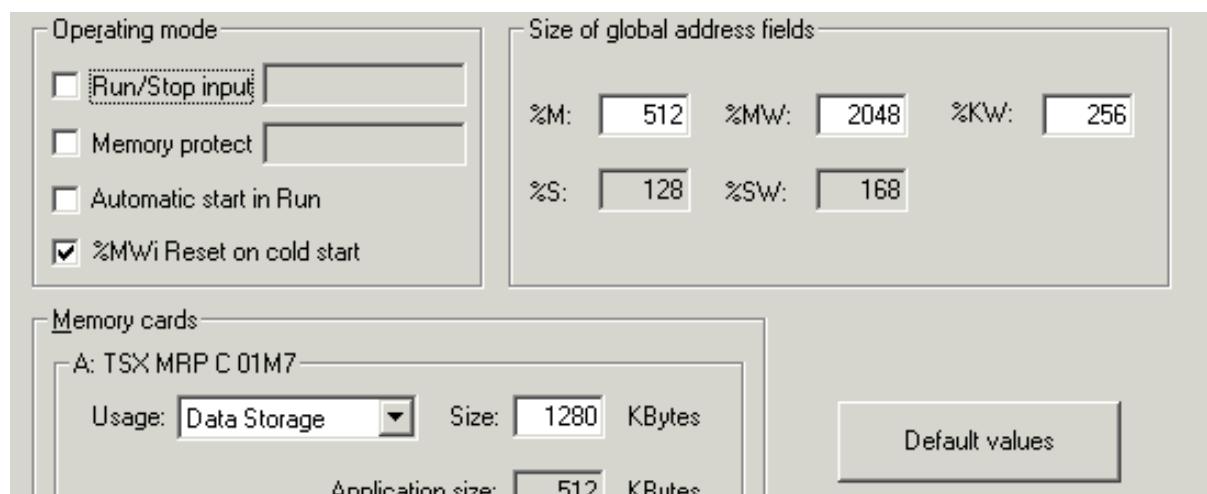


## Configuração do Processador Premium

O **Processador Premium** é configurado nas seguintes etapas

1. A caixa de diálogo para configuração do Processador é disponibilizada com dois cliques sobre o mesmo no rack.
2. Selecione o modo de operação: Entrada para Run/Stop, proteção de memória, ...
3. Defina o cartão de Memória
4. Defina os objetos globais da aplicação: número de bits e words.

Caixa de diálogo para configuração do Processador



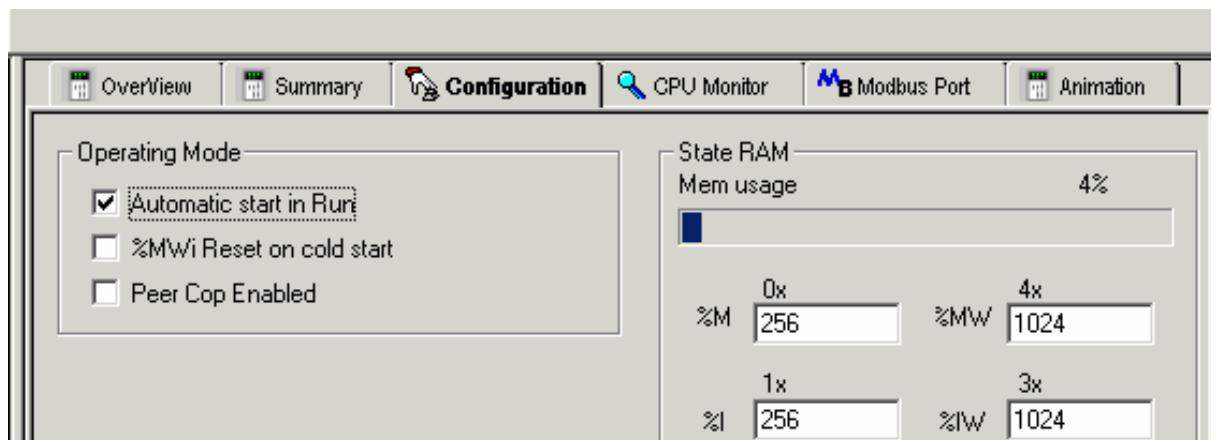
## Configuração do Processador Quantum

O **Processador Quantum** é configurado nas seguintes etapas:

1. A caixa de diálogo para configuração do Processador é disponibilizada com dois cliques sobre o mesmo no rack.
2. Selecionar o modo de operação: Automatic Start in Run, Peer Cop
3. Definir os objetos globais da aplicação. Número de bits e words.

## Configuração de Hardware

Caixa de diálogo para configuração do Processador.

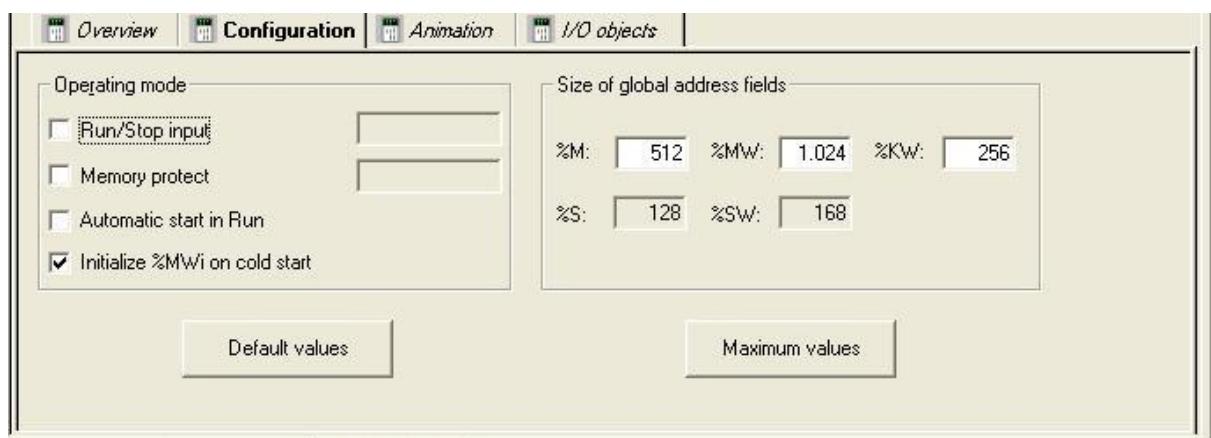


## Configuração do Processador Modicon M340

O Processador Modicon M340 é configurado nas seguintes etapas:

1. A caixa de diálogo para configuração do Processador é disponibilizada com dois cliques sobre o mesmo no rack.
2. Selecione o modo de operação: Entrada para Run/Stop, proteção de memória, ...
3. Defina os objetos globais da aplicação: número de bits e words.

Caixa de diálogo para configuração do Processador.



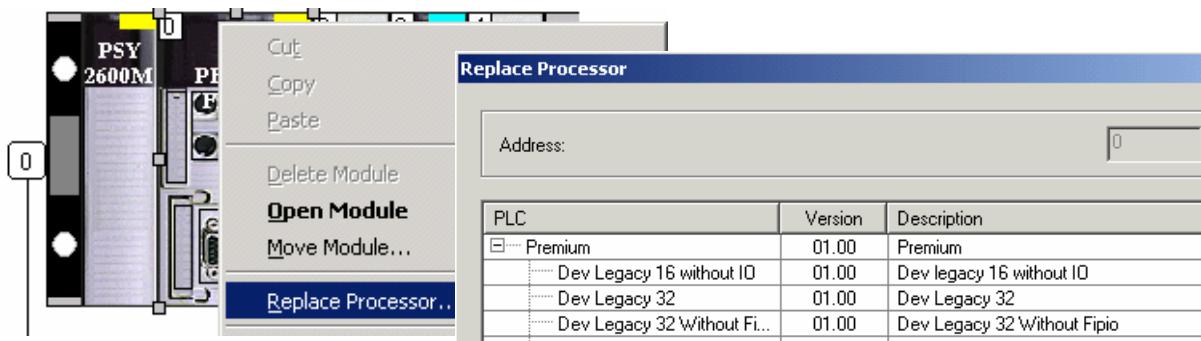
## Substituição do Processador

Para substituir o processador:

- Clicar com botão direito no processador
- Somente os processadores de mesma família são listados
- A substituição do processador é possível no modo off-line.



Somente o processador Quantum pode ser colocado em quer posição no rack.



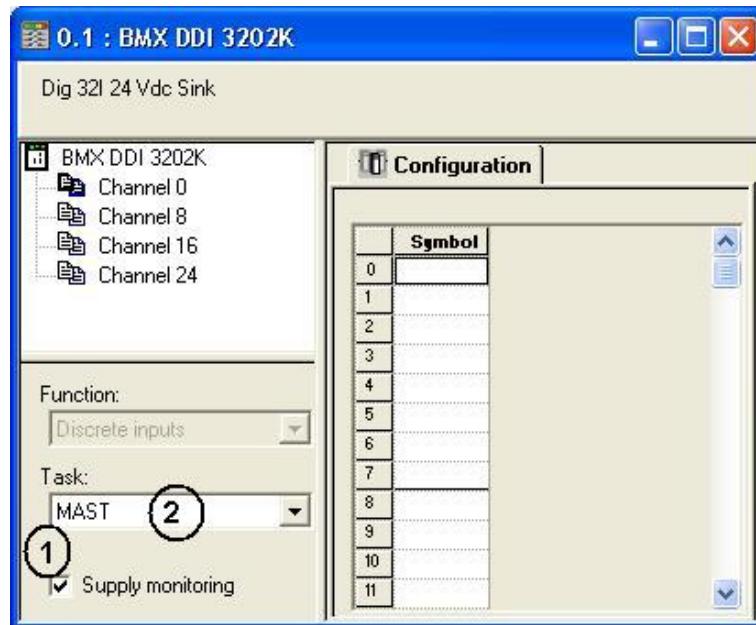
## Configuração de Módulos

A caixa de diálogo para configuração dos módulos é disponibilizada com dois cliques sobre o mesmo no rack em vista expandida ou sobre o código do mesmo em “configuration” no “Project Browser” em vista estruturada.

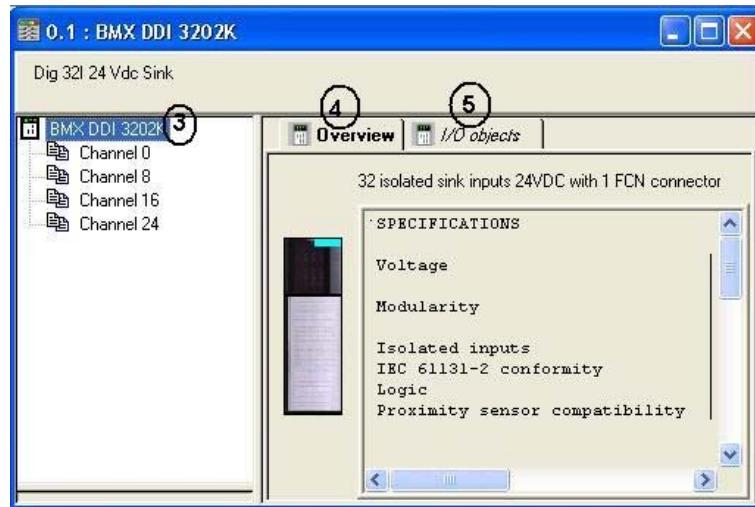
De acordo com o módulo selecionado haverá as configurações necessárias dos parâmetros e dos canais do mesmo.

- Configurações necessárias de módulos de entrada digitais.**

Ao acessar a configuração do módulo de entrada digital a partir do project browser ou vista expandida do rack, é visualizada a caixa de diálogo para configuração do mesmo, onde deve permanecer a caixa “Supply monitoring” (1) (Supervisão de alimentação) e tipo de tarefa (2) “Mast” (tarefa Mestre, a ser visto no Cap. 10 - Estrutura da Aplicação), a coluna “Symbol” é atualizada a partir das declarações de variáveis a ser visto Cap. 9 – Variáveis.

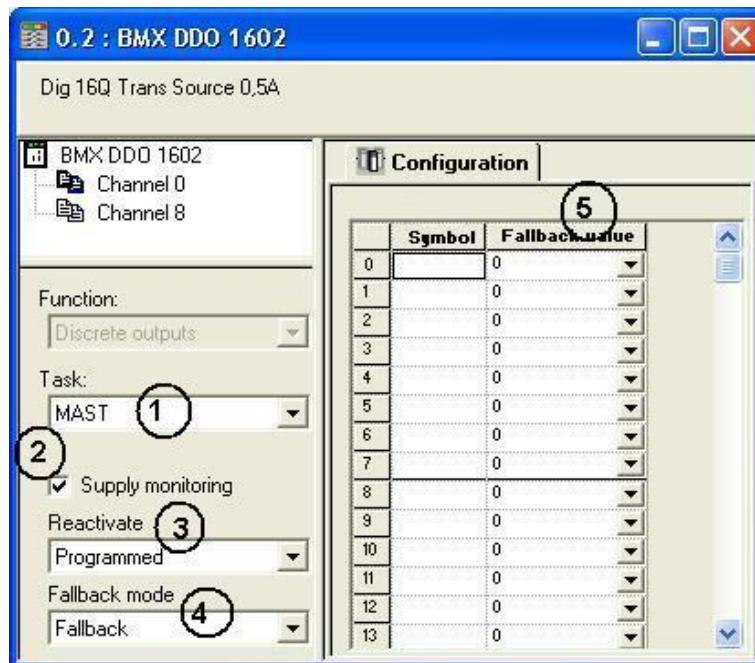


Ao clicar no código do tipo do módulo (3), são visualizadas as abas “Overview” (4) que apresenta informações gerais do módulo e “I/O Objects” que significa configuração de objetos de Entradas/saídas e que será visto no Cap.16 Tipo de dados derivados – DDT.

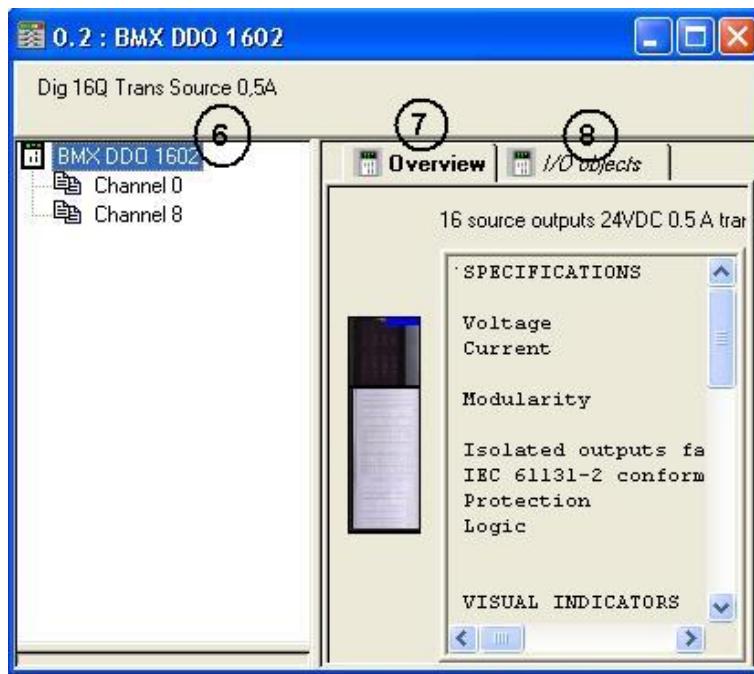


- Configurações necessárias de módulos de saídas digitais.**

Ao acessar a configuração do módulo de saída digital a partir do project browser ou vista expandida do rack, é visualizada a caixa de diálogo para configuração do mesmo, onde deve permanecer o tipo de tarefa (1) “Mast” (tarefa Mestre, a ser visto no Cap. 10 - Estrutura da Aplicação), a caixa “Supply monitoring” (2) (Supervisão de alimentação), selecionar reativação após curto circuito ou sobre carga com desmagnetização eletromagnética rápida dos circuitos para “programed” (3), o retorno de falha “Fallback mode” permanecer em Fallback (4), em “Fallback value” (5) permanecer em “0”, a coluna “Symbol” é atualizada a partir das declarações de variáveis a ser visto Cap. 9 – Variáveis.

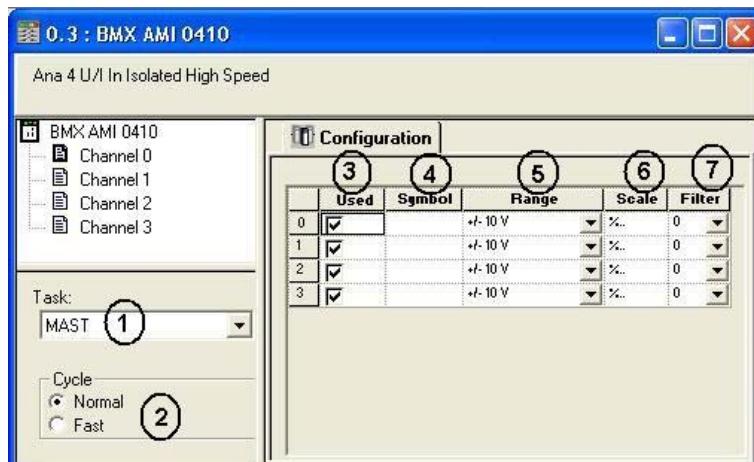


Ao clicar no código do tipo do módulo (6), são visualizadas as abas “Overview” (7) que apresenta informações gerais do módulo e “I/O Objects” (8) que significa configuração de objetos de Entradas/saídas e que será visto no Cap.16 Tipo de dados derivados – DDT.

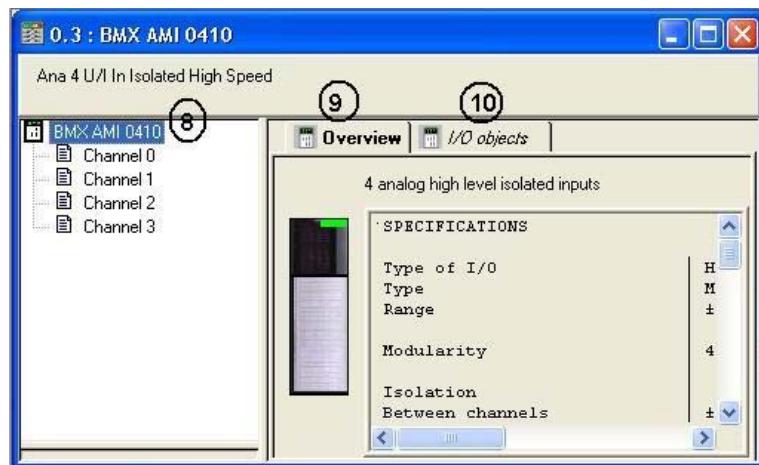


- **Configurações necessárias de módulos de entradas analógicas.**

Ao acessar a configuração do módulo de entradas analógicas a partir do “Project browser” ou vista expandida do rack, é visualizada a caixa de diálogo para configuração do mesmo, onde deve permanecer a caixa o tipo de tarefa (1) “Mast” (tarefa Mestre, o ciclo de processamento “Cycle” (2) deve permanecer em “Normal” ambos a serem vistos no Cap. 10 - Estrutura da Aplicação, permanecer com as caixas “Used” (3) marcadas, indicando que o módulo será utilizado na aplicação, a coluna “Symbol” (4) é atualizada a partir das declarações de variáveis a ser visto Cap. 9 – Variáveis, selecionar a faixa de tensão ou de corrente de trabalho do mesmo em “Range” (5), a configuração de escala “scale” (6) deve ser selecionada de acordo com a conveniência e faixa “Range” (5) selecionada, o parâmetro “Filter” (7) deve ser selecionado de acordo com a conveniência com 7 possibilidades que alteram o tempo de resposta e freqüência de corte do canal, tal filtro é classificado de acordo com o valor selecionado como filtro Baixo, Médio, alto ou nenhum no caso de seleção zero.

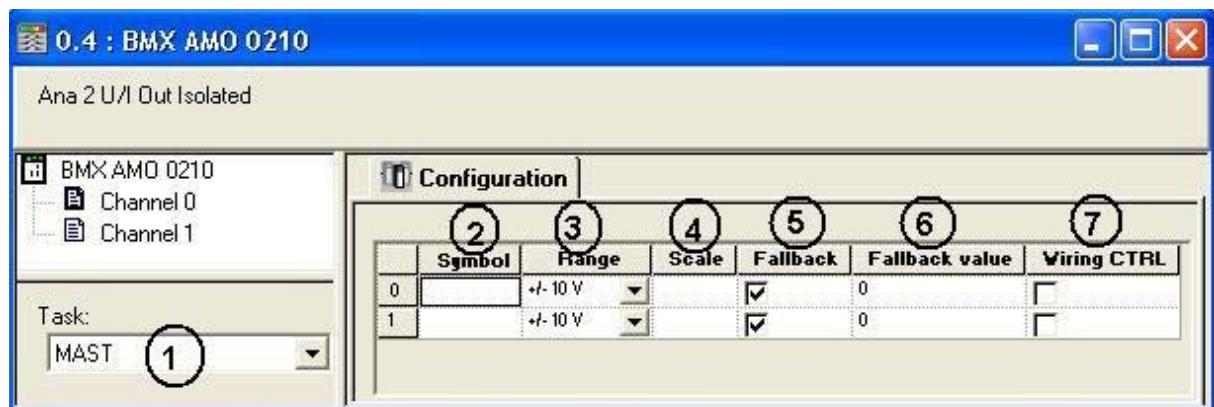


Ao clicar no código do tipo do módulo (8), são visualizadas as abas “Overview” (9) que apresenta informações gerais do módulo e “I/O Objects” (10) que significa configuração de objetos de Entradas/saídas e que será visto no Cap.16 Tipo de dados derivados – DDT.

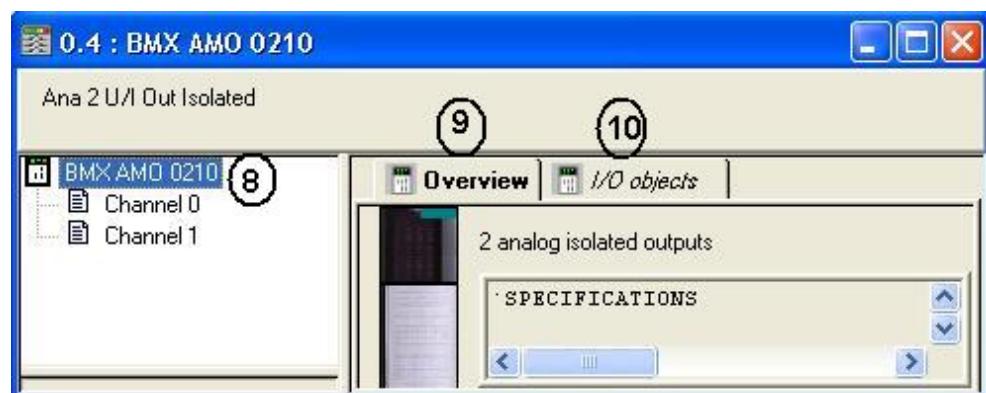


- **Configurações necessárias de módulos de saídas analógicas.**

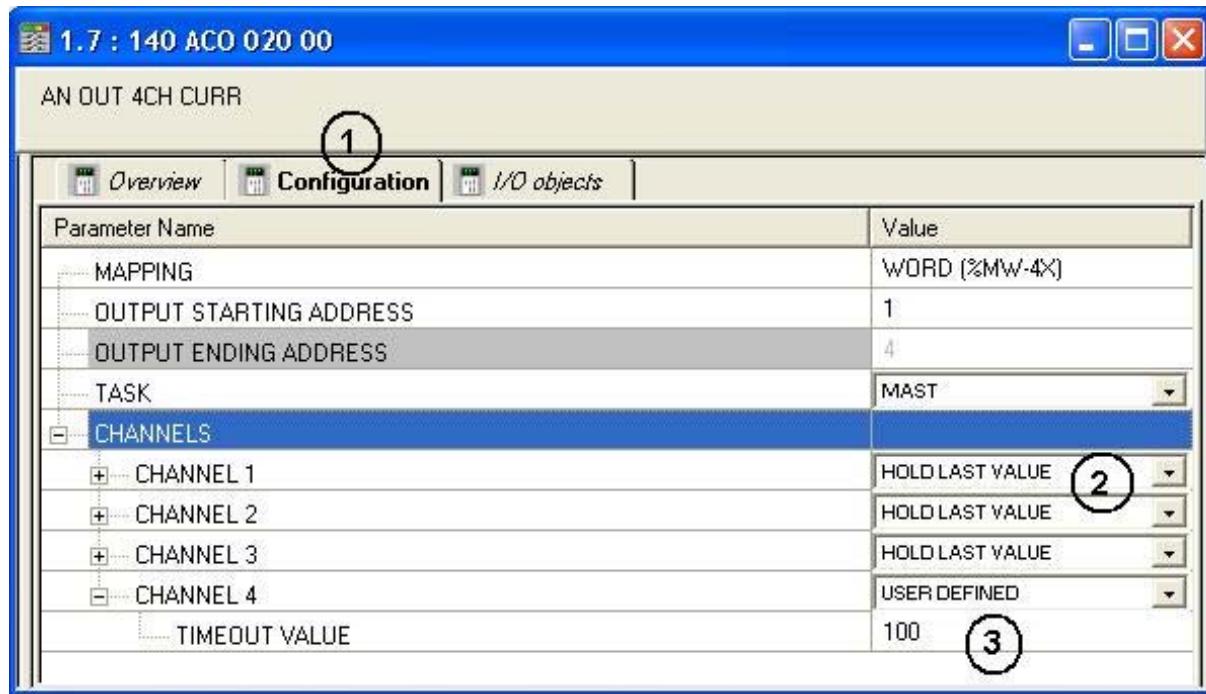
Ao acessar a configuração do módulo de saídas analógicas a partir do “Project browser” ou vista expandida do rack, é visualizada a caixa de diálogo para configuração do mesmo, onde deve permanecer a caixa o tipo de tarefa (1) “Mast” (tarefa Mestre a ser visto no Cap. 10 - Estrutura da Aplicação), a coluna “Symbol” (2) é atualizada a partir das declarações de variáveis a ser visto Cap. 9 – Variáveis, selecionar a faixa de tensão ou de corrente de trabalho do mesmo em “Range” (3), a configuração de escala “scale” (4) deve ser selecionada de acordo com a conveniência e faixa “Range” (3) selecionada, a caixa “Fallback” (5) que se marcada, o valor de “Fallback value” (6) será assumido no retorno da falha, as caixas “Wiring CTRL” (7) controle de fiação, devem permanecer desmarcadas.



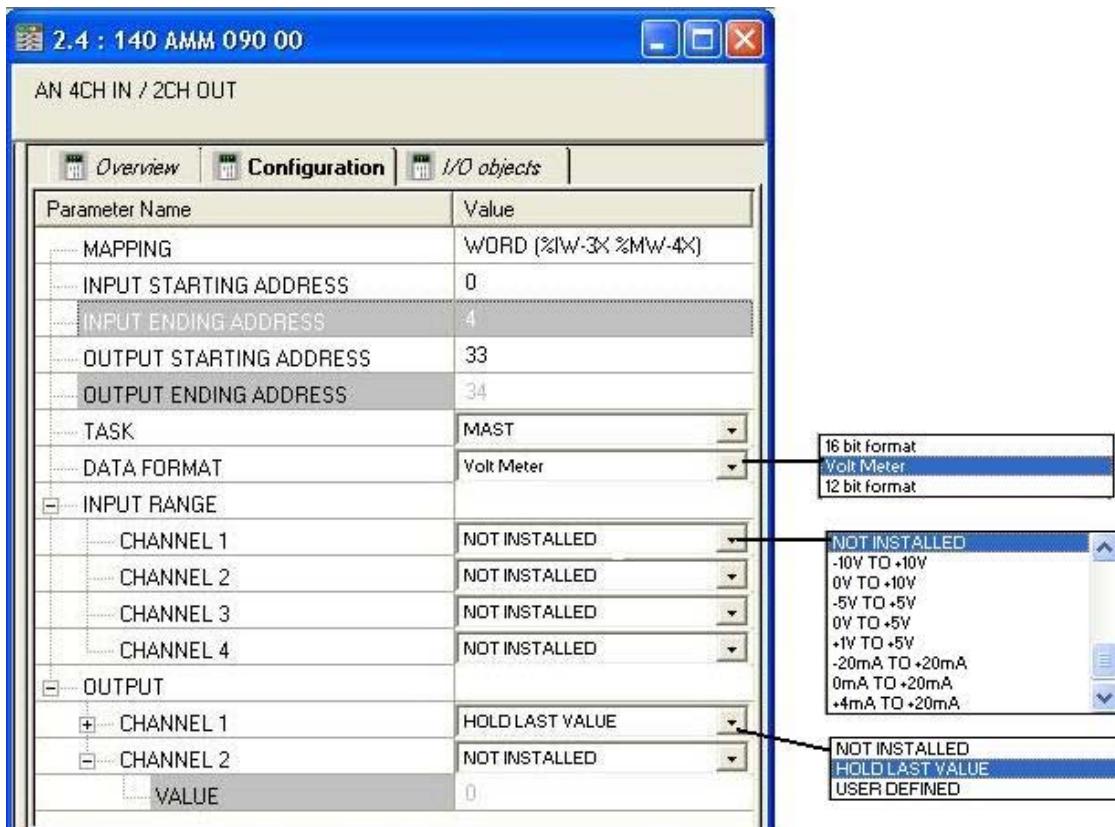
Ao clicar no código do tipo do módulo (8), são visualizadas as abas “Overview” (9) que apresenta informações gerais do módulo e “I/O Objects” (10) que significa configuração de objetos de Entradas/saídas e que será visto no Cap.16 Tipo de dados derivados – DDT.



Particularmente no CLP Quantum a configuração de “Fallback” é feita na aba “Configuration” (1) onde deve-se selecionar o tipo de retorno “Fallback” para cada canal (2) podendo ser “Disable”, Hold Last Value ou “User Defined”, quando selecionado “User Defined, deve-se especificar “Timeout Value” em (3) o qual é especificado com constante de 0 a 65535.



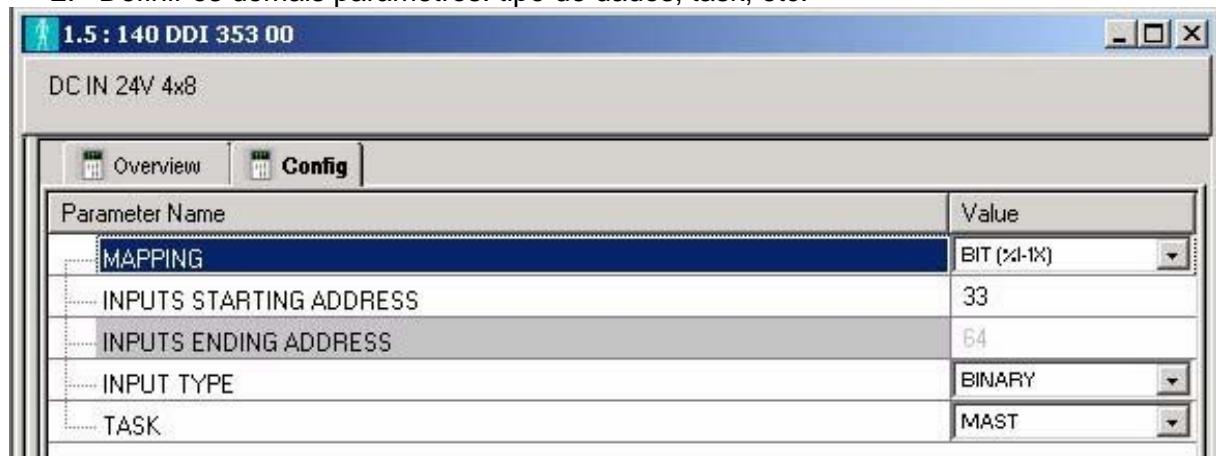
Dependendo do tipo de módulo, por exemplo, os bi-direcionais e multi-escalas com seleção de link de corrente ou tensão, como o módulo AMM 090 00 são necessárias ainda os ajustes de faixa de entrada, o valor da saída no caso de falha e o formato do dado, como visto na figura seguinte.



## Endereçamento do CLP Quantum

Particularmente para o processador Quantum deve-se:

1. Definir o endereço usado pelo módulo: Mapeamento por bit ou por Word e o primeiro endereço do mesmo, por ex. 33, o endereço final será o endereço inicial 33 + (Nº de canais -1); portanto 64.
2. Definir os demais parâmetros: tipo de dados, task, etc.



No padrão IEC, o objeto é definido por:

**Um símbolo:**

%

**Um tipo:**

- M : para variáveis internas
- K : para constantes
- S : para variáveis de sistema
- N : para variáveis de rede
- I : para variáveis de entrada
- Q : para variáveis de saída

**Um formato :**

- X : booleana (Implícita no Unity, não necessita escrever-la)
- B : byte (não usada no Unity)
- W : word (16 bits)
- D : double word (32 bits)
- F: Número real (Flutuante)

## Endereçamento Físico de Entrada / Saída

%	I / Q	X / W / D	r	.	m	.	c	.	d	.	j
Símbolo	Tipo: I=Entrada Q=Saída	Formato X=Booleana W=Word D= Dupla word	Rack		Modulo		Canal		rank		Bit

## Endereço topológico de dados alocados Quantum

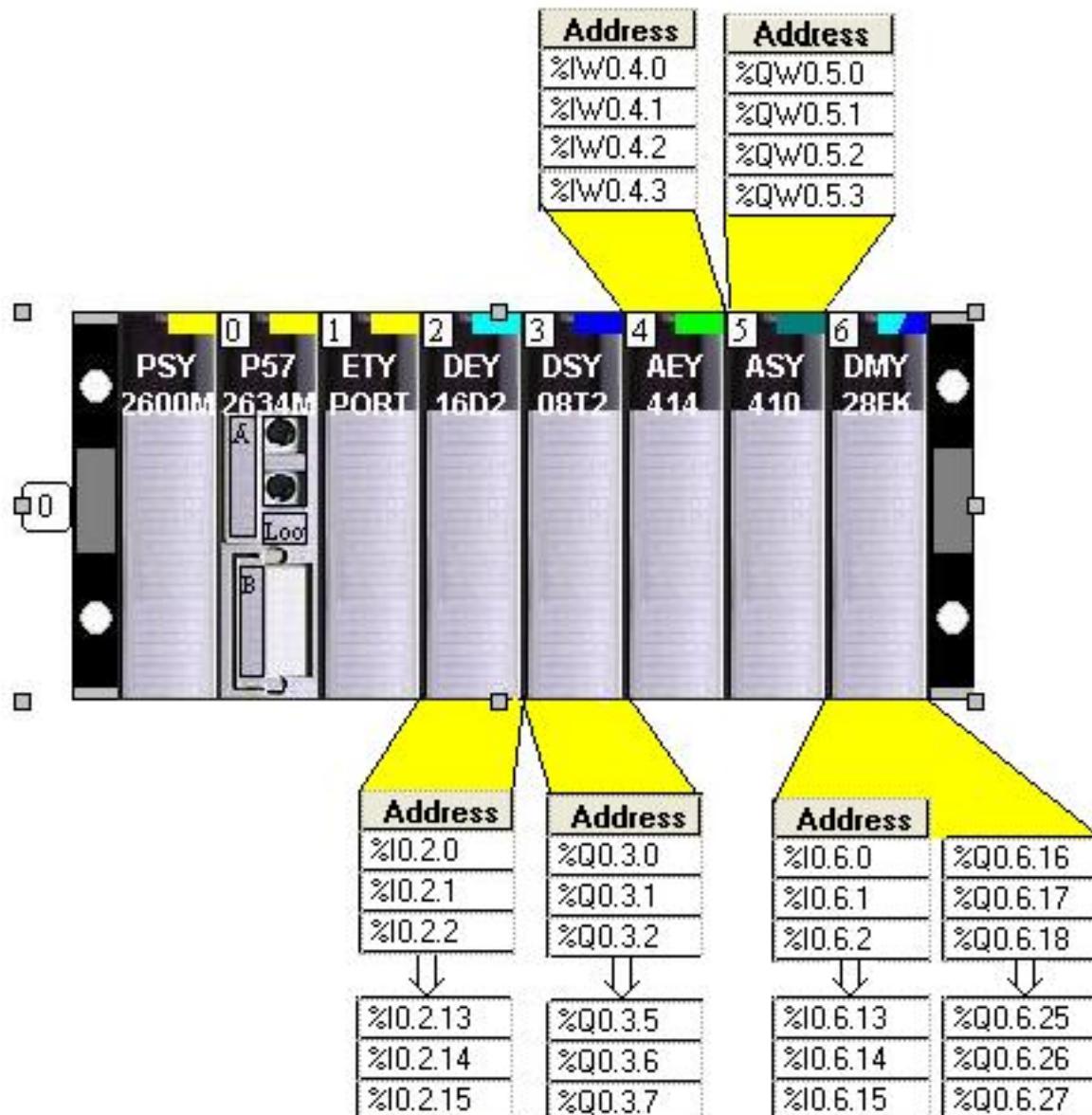
Endereço	Endereço Topológico	Dado usado
0xxxxx	%Qr.m.c.d / %Mi	Bits do Módulo e bits internos
1xxxxx	%Ir.m.c.d / %li	Bits do Módulo de entrada
3xxxxx	%IWr.m.c.d / %IWi	Words de entrada de módulos de Entrada/saída
4xxxxx	%QWr.m.c.d / %MWi	Words de saída de módulos de Entrada/saída e words internas

## *Exemplos de endereçamento.*

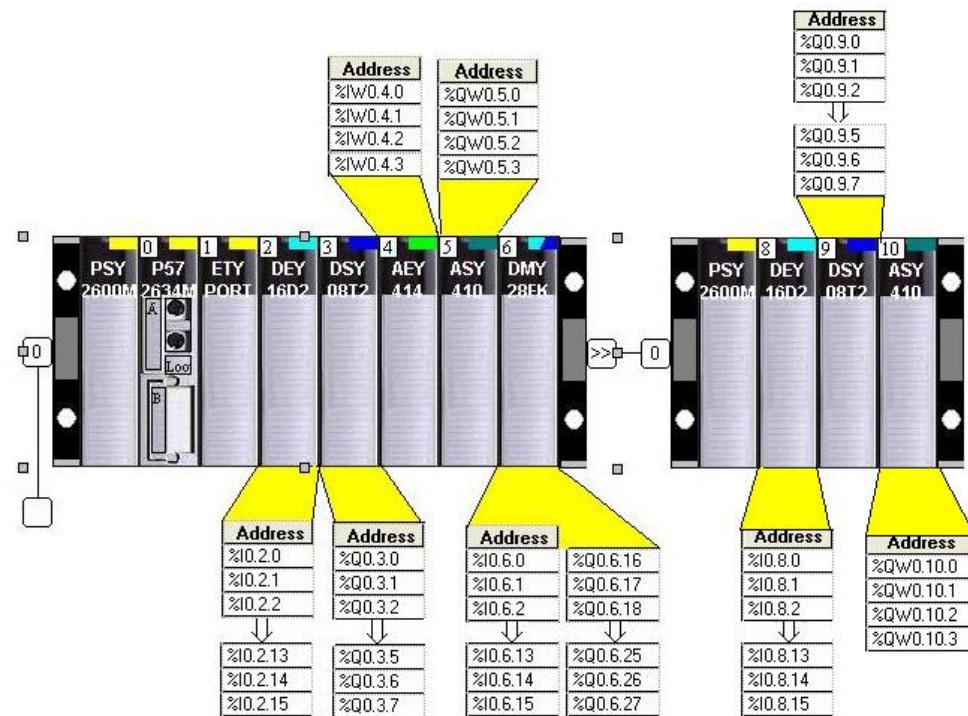
Serão apresentados os endereços gerados a partir de configuração de hardware dos CLPS Premium, Quantum e M340.

- **Endereçamento de estação CLP Premium.**

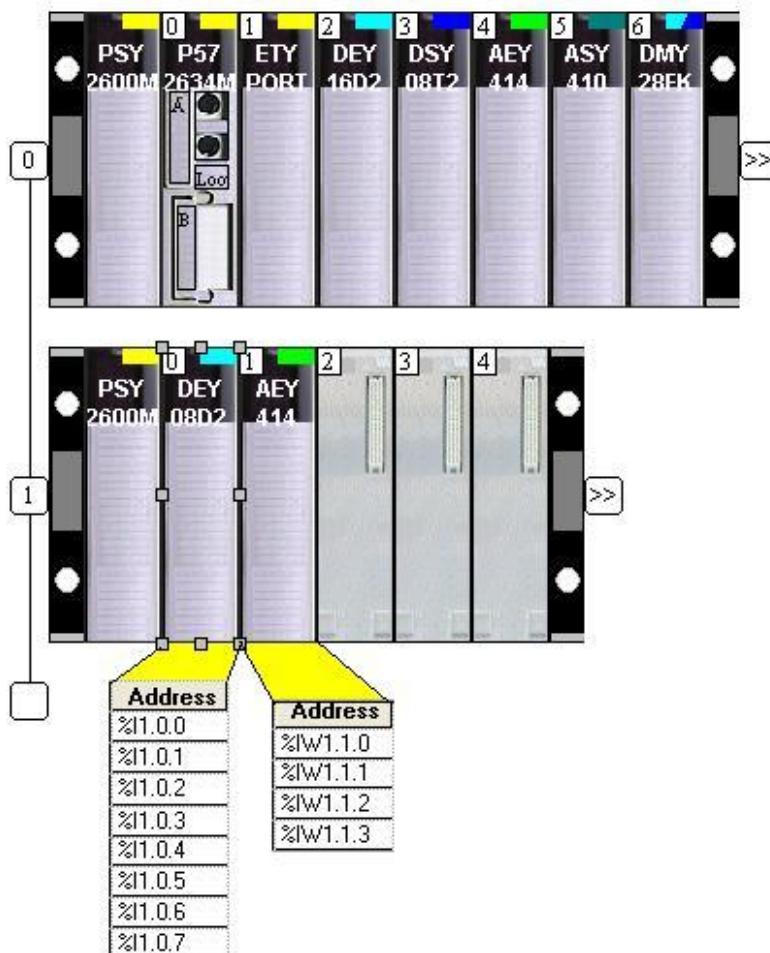
- Estação com rack local:**



- Estação com expansão de rack.**



c) Estação com expansão de rack.

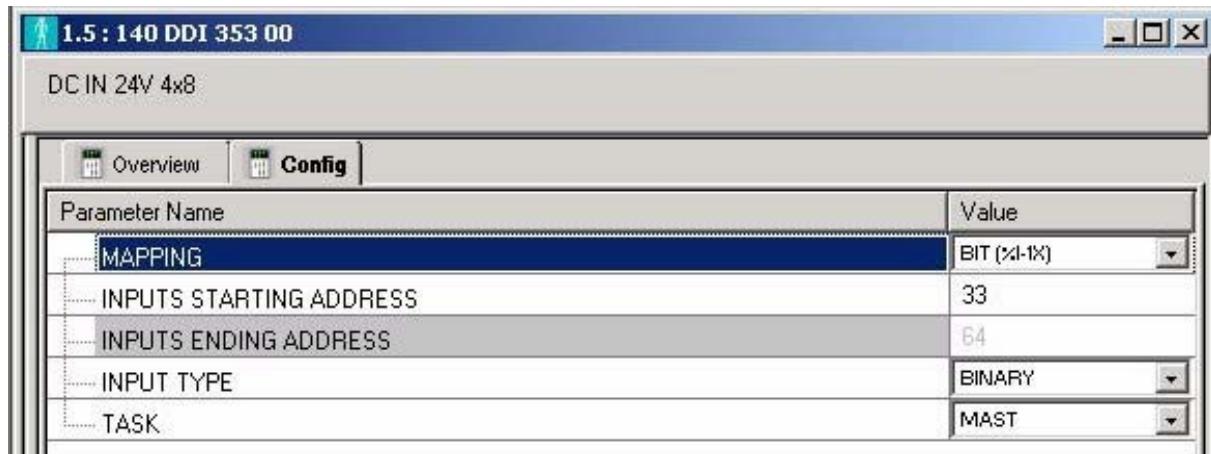


- Endereçamento de estação CLP Quantum.

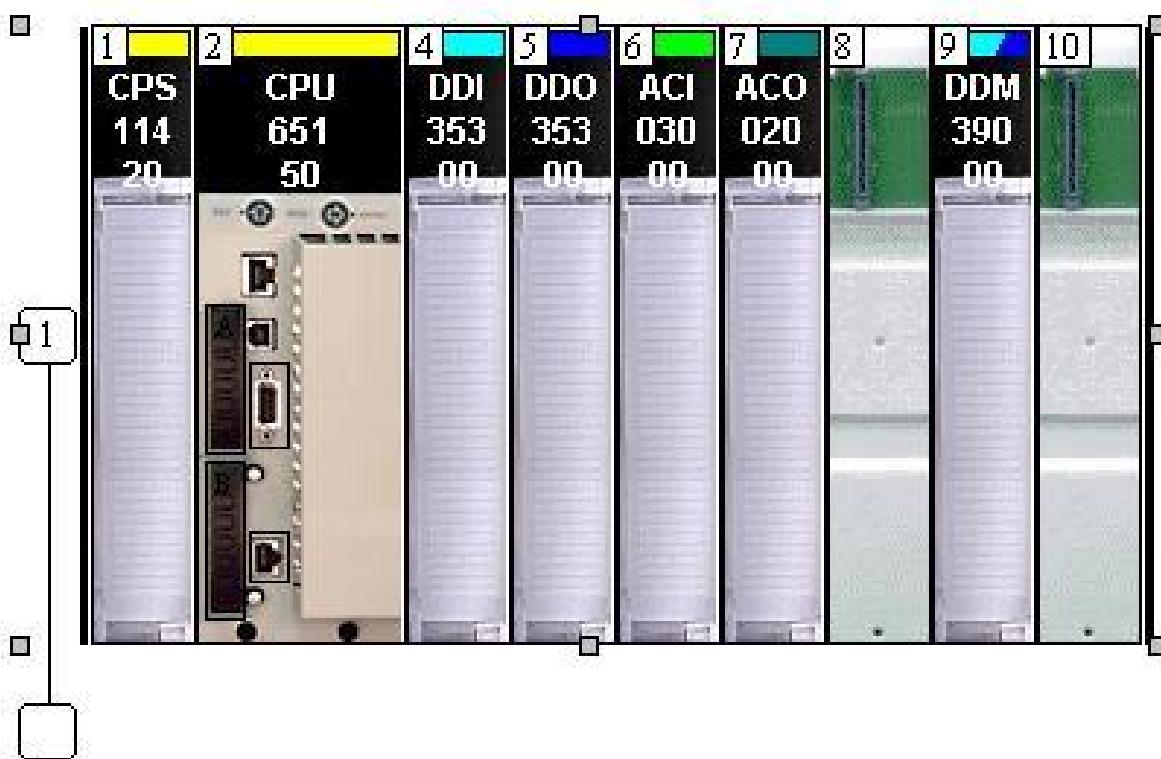
#### a) Estação com rack Local

Especificamente para o CLP Quantum é necessário especificar o sistema de mapeamento, como visto anteriormente, sendo:

- Definir o endereço usado pelo módulo: Mapeamento por bit ou por Word e o primeiro endereço do mesmo, por ex. **33**, o endereço final será o endereço inicial 33 + (Nº de canais -1); portanto **64**.
- Definir os demais parâmetros: tipo de dados, task, etc.



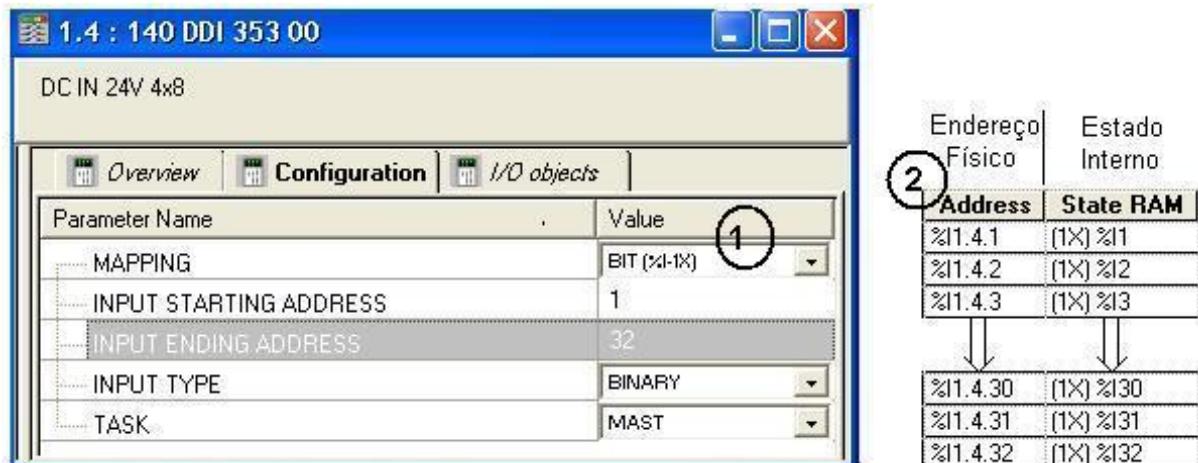
Considerando a configuração da estação Quantum a seguir:



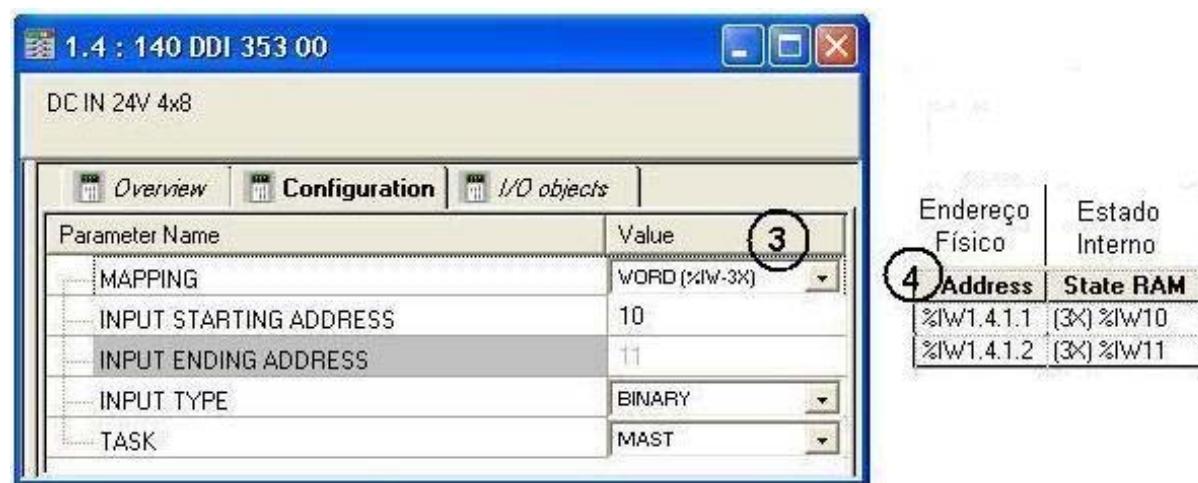
#### Endereços do módulo de entradas digitais DDI 353 00:

Ao definir o tipo de mapeamento por BIT (1), as faixas de endereços (2) serão de %I1.4.1 a %I1.4.32 para os endereços físicos e %I1 a %I32 para os estados internos.

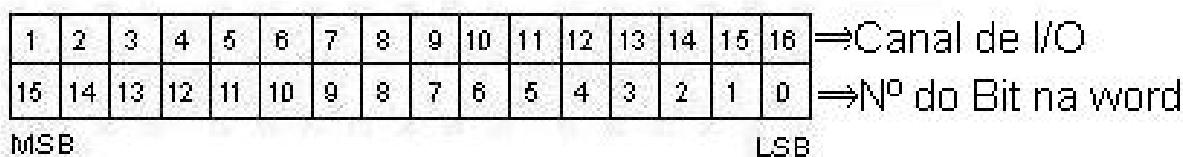
Os endereços podem ser utilizados de maneira mesclada, ou seja, estado interno RAM (%I2) e/ou endereço físico (%I1.4.1), no caso de utilizar %I1.4.1 com %I1, ao acionar uma delas a outra é acionada juntamente, por se tratar de mesma posição de memória.



Ao definir o tipo de mapeamento por Word **(3)**, as faixas de endereços **(4)** serão; %IW1.4.1.1 e %IW1.4.1.2 para os enderecos físicos e %IW10 e %IW11 para os estados internos.



Ao utilizar o endereçamento por word, deve-se indicar o bit desta word que está sendo utilizado, obedecendo a seguinte regra:

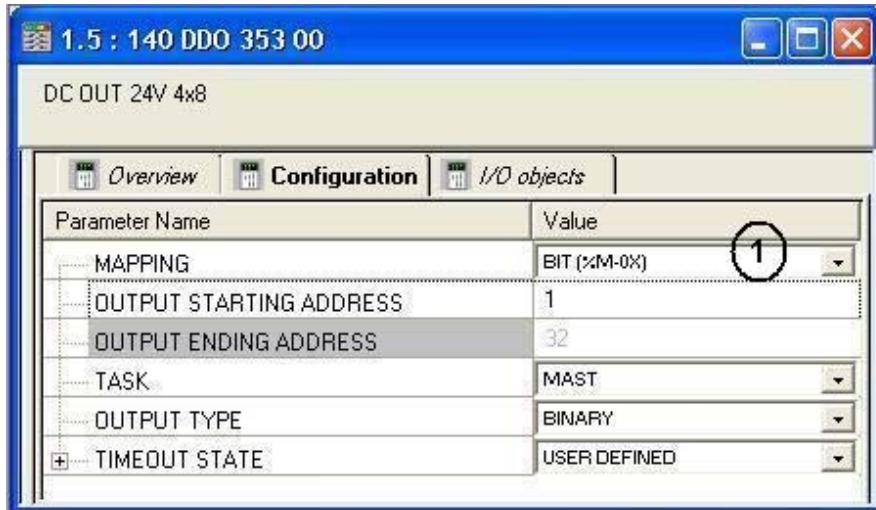


Por exemplo, acionamento do canal 1 do módulo => %IW1.4.1.1.15 ou %IW10.15

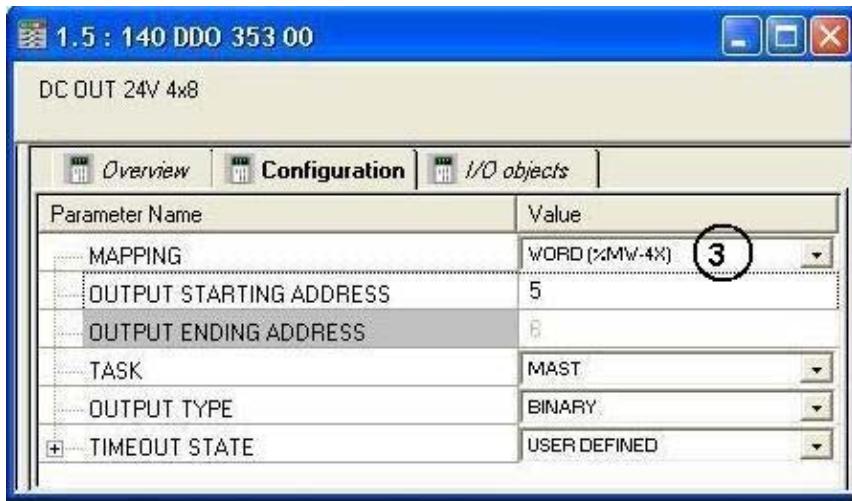
**Endereços do módulo de saídas digitais DD0 353 00:**

Ao definir o tipo de mapeamento por **BIT (1)**, as faixas de endereços **(2)** serão de %Q1.5.1 a %Q1.5.32 para os endereços físicos e %M1 a %M32 para os estados internos.

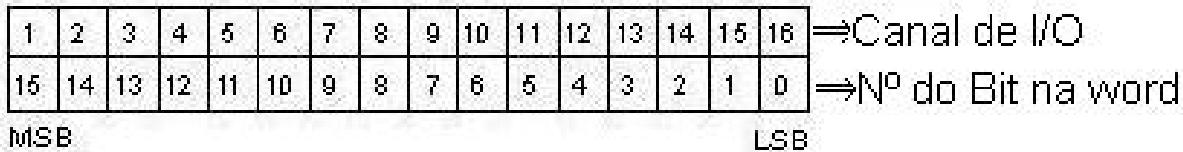
Os endereços podem ser utilizados de maneira mesclada, ou seja, estado interno RAM (%M2) e/ou endereço físico (%Q1.5.1), no caso de utilizar %Q1.5.1 com %M1, ao acionar uma delas a outra é acionada juntamente, por se tratar de mesma posição de memória.



Ao definir o tipo de mapeamento por Word (3), as faixas de endereços (4) serão; %QW1.5.1.1 e %QW1.5.1.2 para os endereços físicos e %MW5 e %MW6 para os estados internos.



Ao utilizar o endereçamento por Word, deve-se indicar o bit desta Word que está sendo utilizado, obedecendo a seguinte regra:



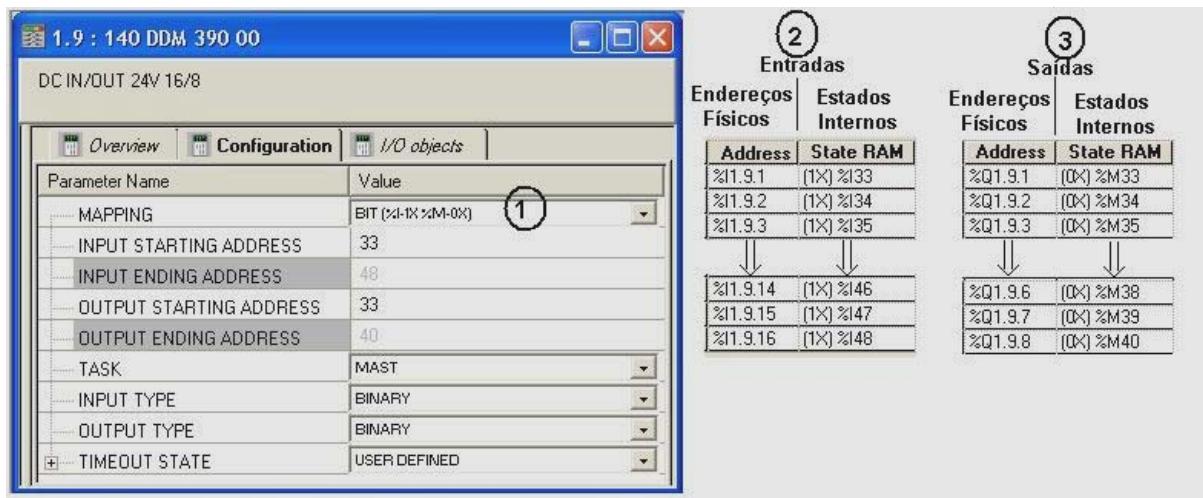
Por exemplo, acionamento do canal 1 do módulo => %QW1.5.1.1.15 ou %MW5.15

#### Endereços do módulo combinado entradas/saídas digitais DDM 390 00:

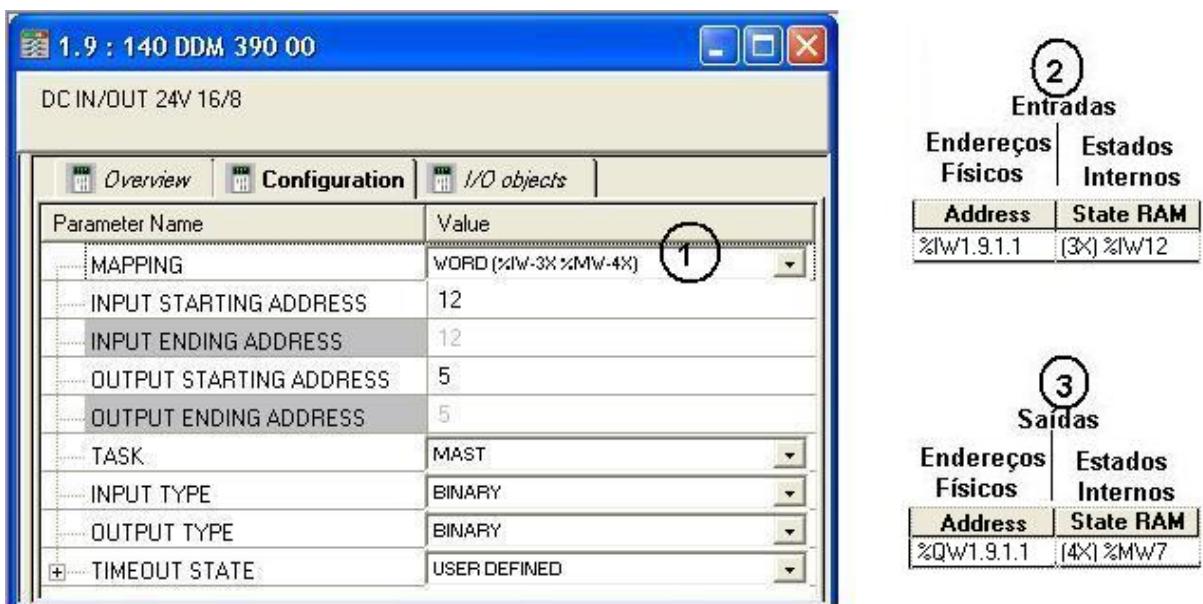
Ao definir o tipo de mapeamento por BIT (1), as faixas de endereços (2) para as entradas serão de %I1.9.1 a %I1.9.16 para os endereços físicos e %I33 a %I48 para os estados internos o qual acompanha a seqüência de endereçamento do módulo de entrada digital anterior instalado no rack na posição 4. As faixas de endereços (3) para as saídas serão de %Q1.9.1 a %Q1.9.8 para os endereços físicos e %M33 a %M40 para os estados internos os quais acompanham a seqüência de endereçamento do módulo de saída digital anterior instalado no rack na posição 5.

## Configuração de Hardware

Os endereços podem ser utilizados de maneira mesclada, ou seja, estado interno RAM e/ou endereço físico.



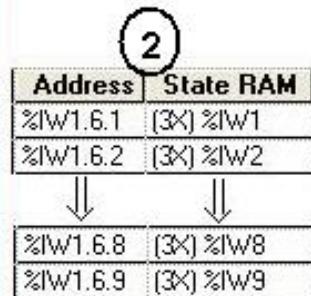
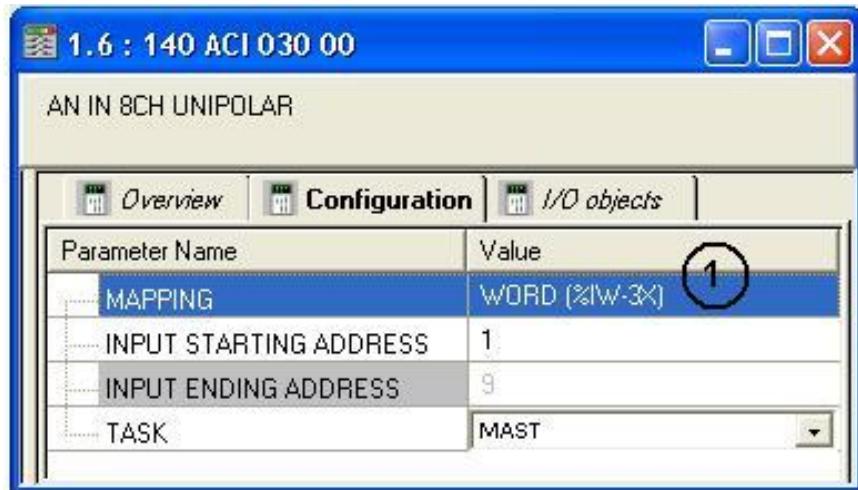
Ao definir o tipo de mapeamento por WORD (1), as faixas de endereços (2) para as entradas serão de %IW1.9.1.1 para os endereços físicos e %IW1.9.1.8 para os estados internos o qual acompanha a seqüência de endereçamento do módulo de entrada digital anterior instalado no rack na posição 4. As faixas de endereços (3) para as saídas serão de %QW1.9.1.1 para os endereços físicos e %MW7 para os estados internos os quais acompanham a seqüência de endereçamento do módulo de saída digital anterior instalado no rack na posição 5.



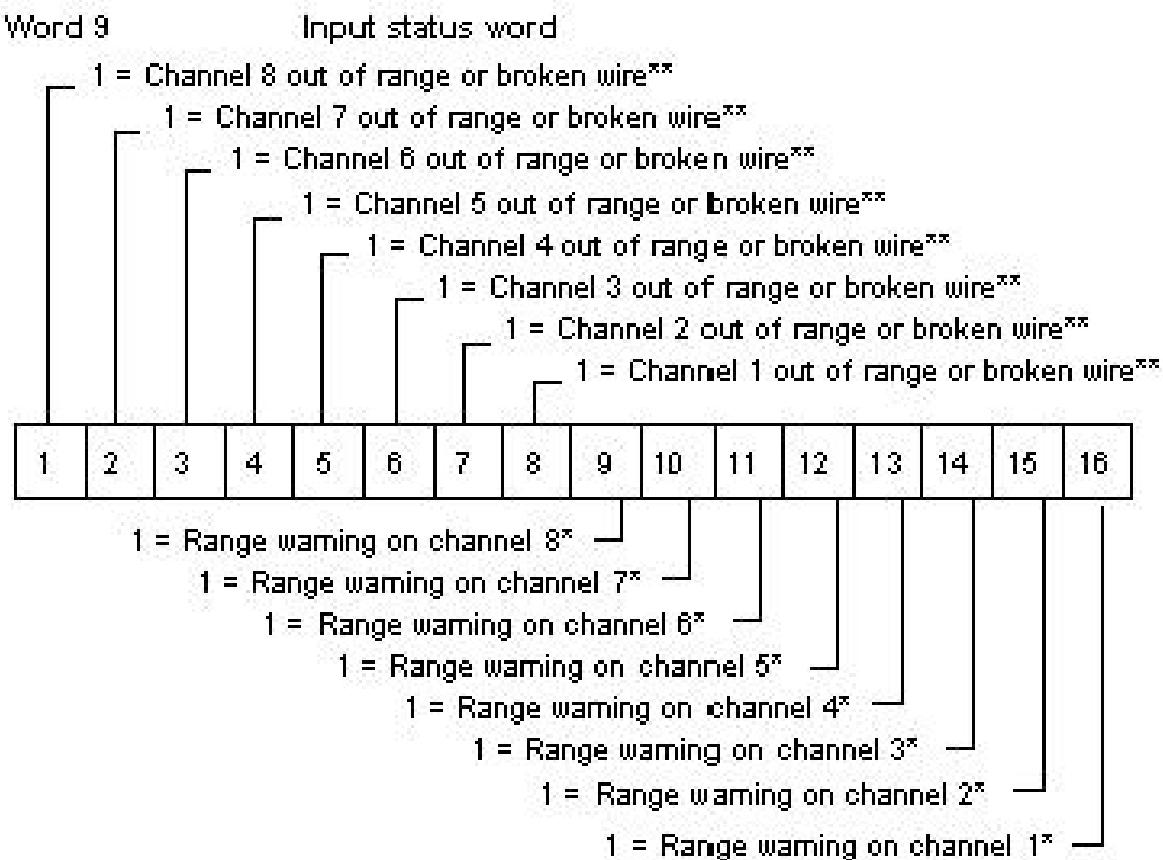
### Endereços do módulo de entradas analógicas ACI 030 00:

O mapeamento da entrada analógica é somente por Word (1) sendo a faixa de endereçamento %IW1.6.1 a %IW1.6.8 para os endereços físicos e %IW1 a %IW8 para os estados internos (2).

Os módulos analógicos possuem um canal dedicado ao estado "Status" dos mesmos, que tem a função de sinalizar valores fora de faixa ou quebra de fios dos canais sendo %IW1.6.9 para os endereços físicos e %IW9 para os estados internos.

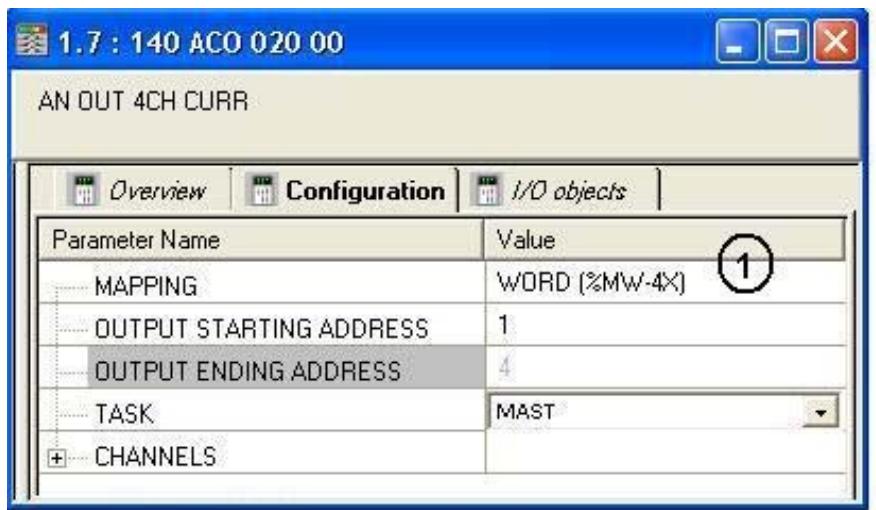


Função dos bits do canal de estado das entradas analógicas:



#### Endereços do módulo de saídas analógicas ACO 020 00:

O mapeamento da saída analógica é somente por Word (1) sendo a faixa de endereçamento %QW1.7.1 a %QW1.7.4 para os endereços físicos, e %MW1 a %MW4 para os estados internos (2).

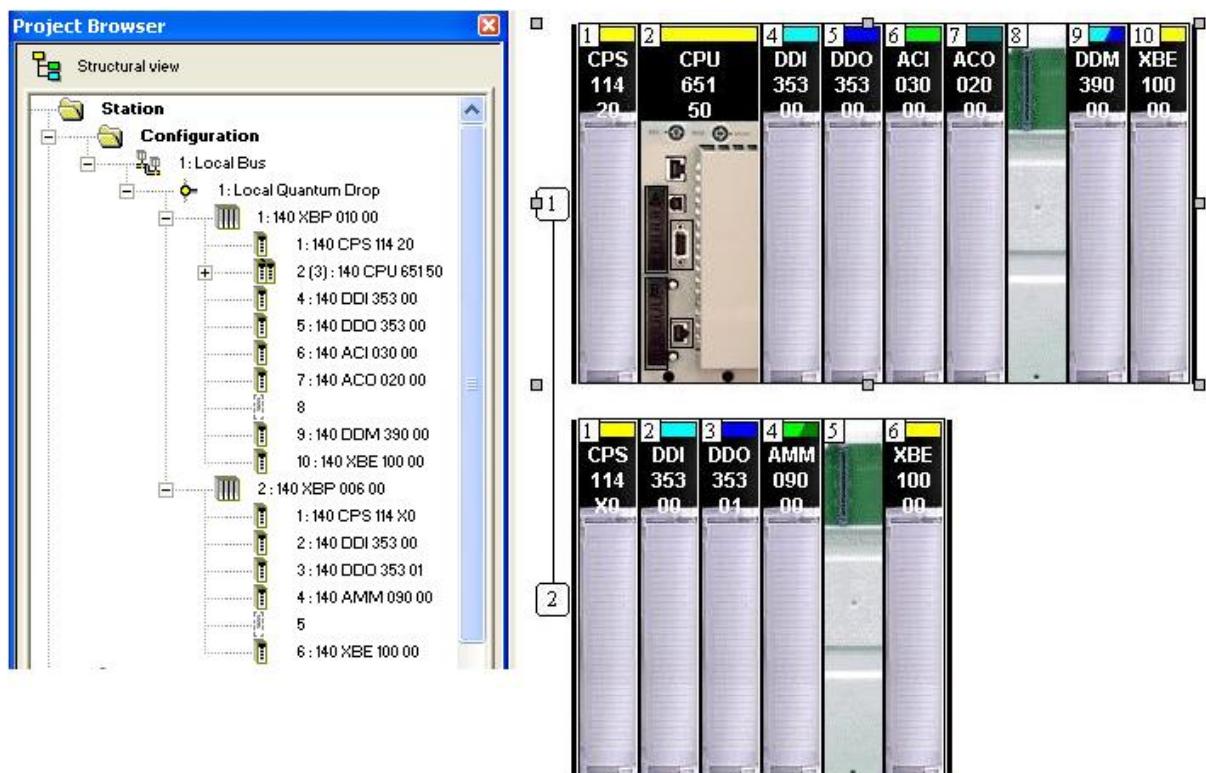


### a) Estação com expansão de rack.

A expansão de um rack para um segundo rack com CLP Quantum é possível com a instalação de módulos Expansores 140 XBE 100 00 um em cada rack, como visto no capítulo 3 em “3.4 Expansão de Bastidor - Módulo Expansor (XBE)”, o qual não necessita de endereçamento

Será considerada a configuração anterior, acrescida da expansão de rack, onde serão analisados os endereços do segundo rack.

### Vista estruturada do Project browser e dos racks.



### Endereçamento do módulo de entradas digitais DDI 353 00:

**Endereçamento  
por BIT**

Address	State RAM
%I2.2.1	(1X) %I49
%I2.2.2	(1X) %I50
%I2.2.3	(1X) %I51
↓	↓
%I2.2.30	(1X) %I78
%I2.2.31	(1X) %I79
%I2.2.32	(1X) %I80

**Endereçamento  
por Word**

Address	State RAM
%IW2.2.1.1	(3X) %IW15
%IW2.2.1.2	(3X) %IW16

Endereçamento do módulo de saídas digitais DDO 353 01:

**Endereçamento  
por BIT**

Address	State RAM
%Q2.3.1	(0X) %M33
%Q2.3.2	(0X) %M34
%Q2.3.3	(0X) %M35
↓	↓
%Q2.3.30	(0X) %M62
%Q2.3.31	(0X) %M63
%Q2.3.32	(0X) %M64

**Endereçamento  
por WORD**

Address	State RAM
%QW2.3.1.1	(4X) %MW7
%QW2.3.1.2	(4X) %MW8

Endereçamento do módulo combinado de Entradas e Saídas analógicas AMM 090:

**Endereçamento por Word**
**Entradas**

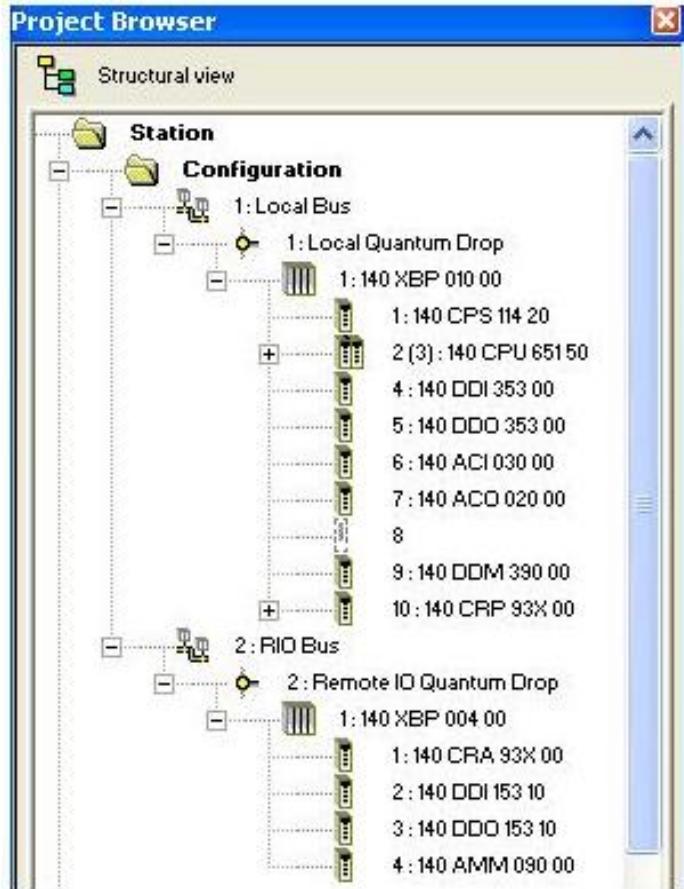
Address	State RAM
%IW2.4.1	(3X) %IW0
%IW2.4.2	(3X) %IW1
%IW2.4.3	(3X) %IW2
%IW2.4.4	(3X) %IW3
%IW2.4.5	(3X) %IW4

**Saídas**

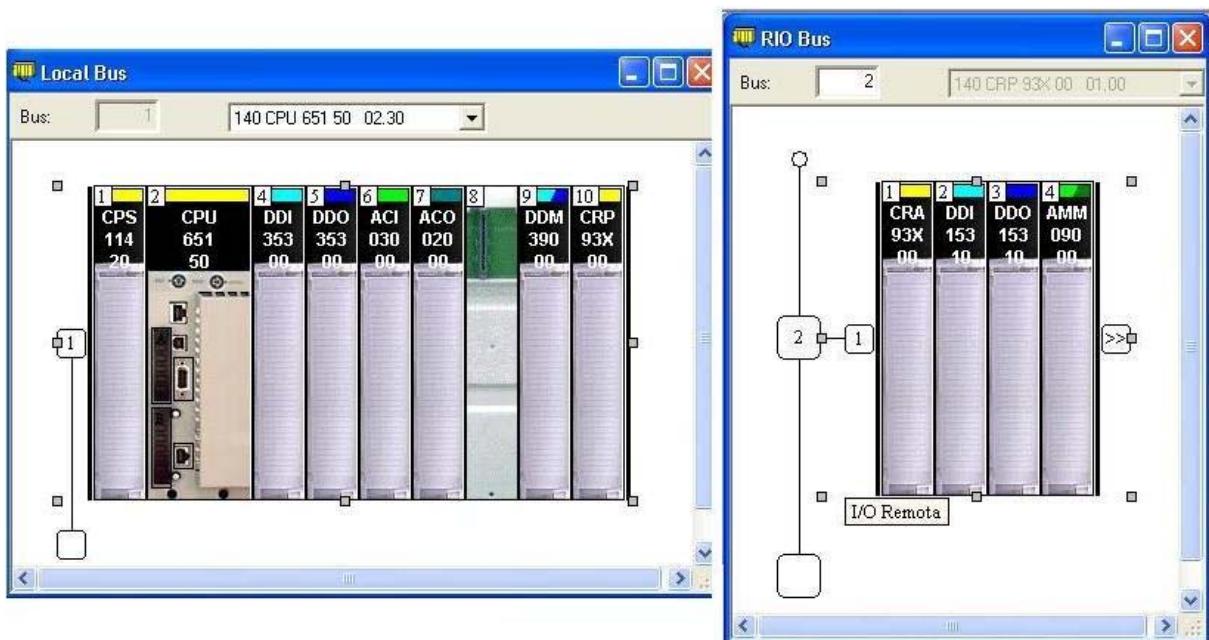
Address	State RAM
%QW2.4.1	(4X) %MW33
%QW2.4.2	(4X) %MW34

b) Estação com estação remota RIO.

Vista estruturada do Project Browser



### Estrutura de Hardware Rack Local e Rack Remoto.



Endereçamento do módulo de entradas digitais DDI 153 10:

## Endereçamento por BIT

Address	State RAM
%I\2\1.2.1	(1X) %I1
%I\2\1.2.2	(1X) %I2
%I\2\1.2.3	(1X) %I3
%I\2\1.2.30	(1X) %I30
%I\2\1.2.31	(1X) %I31
%I\2\1.2.32	(1X) %I32

## Endereçamento por Word

Address	State RAM
%W\2\1.2.1.1	(3X) %W10
%W\2\1.2.1.2	(3X) %W11

Endereçamento do módulo de saídas digitais DDO153 10:

## Endereçamento por Bit

Address	State RAM
%Q\2\1.3.1	(0X) %M33
%Q\2\1.3.2	(0X) %M34
%Q\2\1.3.3	(0X) %M35
%Q\2\1.3.30	(0X) %M62
%Q\2\1.3.31	(0X) %M63
%Q\2\1.3.32	(0X) %M64

## Endereçamento por Word

Address	State RAM
%QW\2\1.3.1.1	(4X) %MW9
%QW\2\1.3.1.2	(4X) %MW10

Endereçamento do módulo combinado de entradas / saídas analógicas AMM 090 00:

## Endereçamento por Word

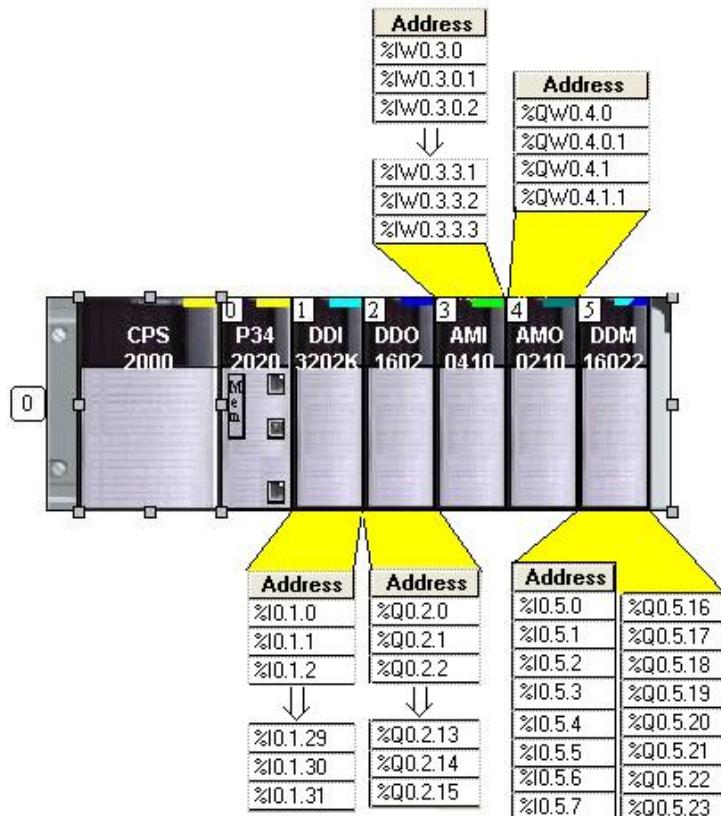
### Entradas

Address	State RAM
%IW\2\1.4.1	(3X) %IW15
%IW\2\1.4.2	(3X) %IW16
%IW\2\1.4.3	(3X) %IW17
%IW\2\1.4.4	(3X) %IW18
%IW\2\1.4.5	(3X) %IW19

### Saídas

Address	State RAM
%QW\2\1.4.1	(4X) %MW7
%QW\2\1.4.2	(4X) %MW8

- Endereçamento de estação CLP M 340.



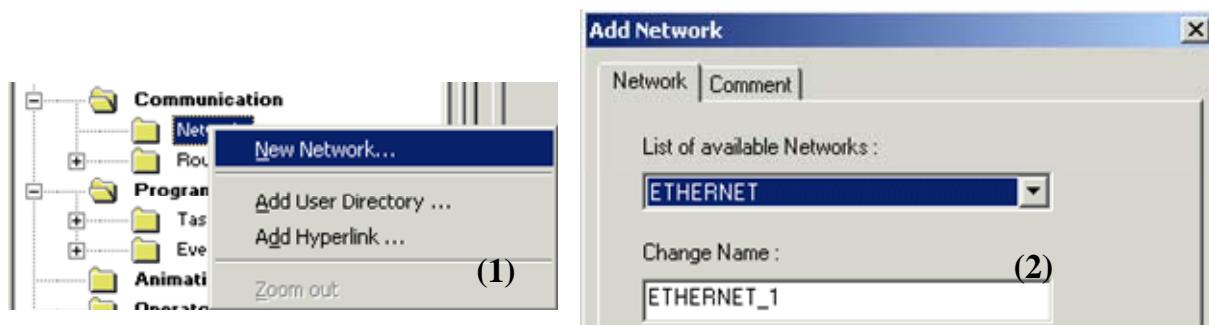
## Configuração da Rede.

Para Configurar uma rede (Ethernet, Modbus+ ou Fipway) deve-se seguir as seguintes etapas:

1. Criar a rede lógica;
2. Configurar a rede lógica;
3. Definir o módulo de comunicação ou o cartão PCMCIA;
4. Associar o modulo ou o cartão PCMCIA com a rede lógica.

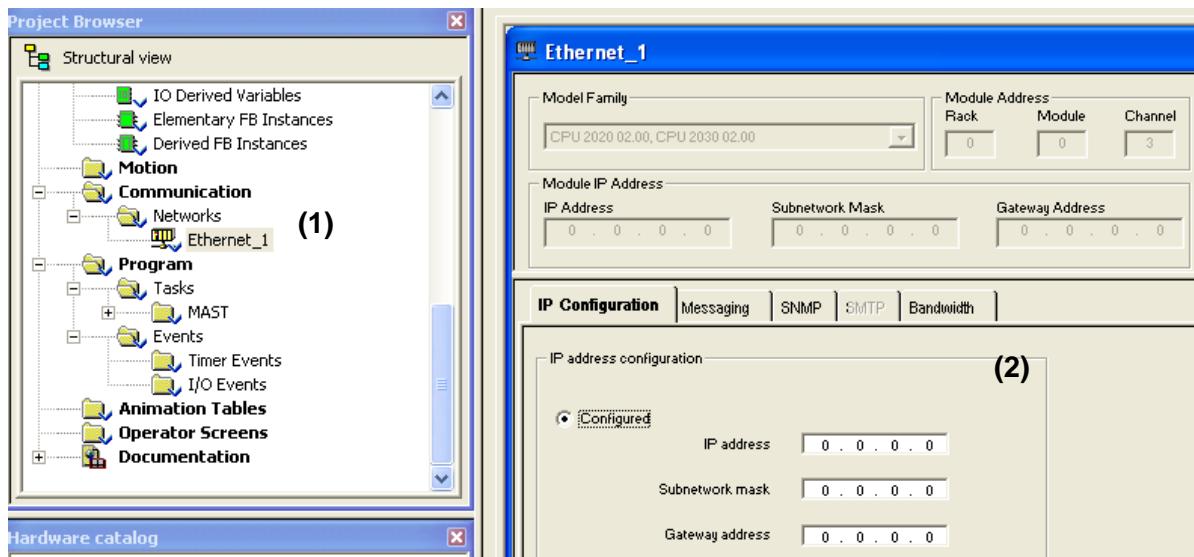
### 1. Criação da Rede Lógica;

- Adicionar uma nova rede (clicar com botão direito do mouse na pasta Networks no browser da aplicação (1)
- Escolher o tipo de rede a ser criada (Ethernet, Modbus+, Fipway) definindo o nome se necessário (2)
- Se necessário entrar com comentários.



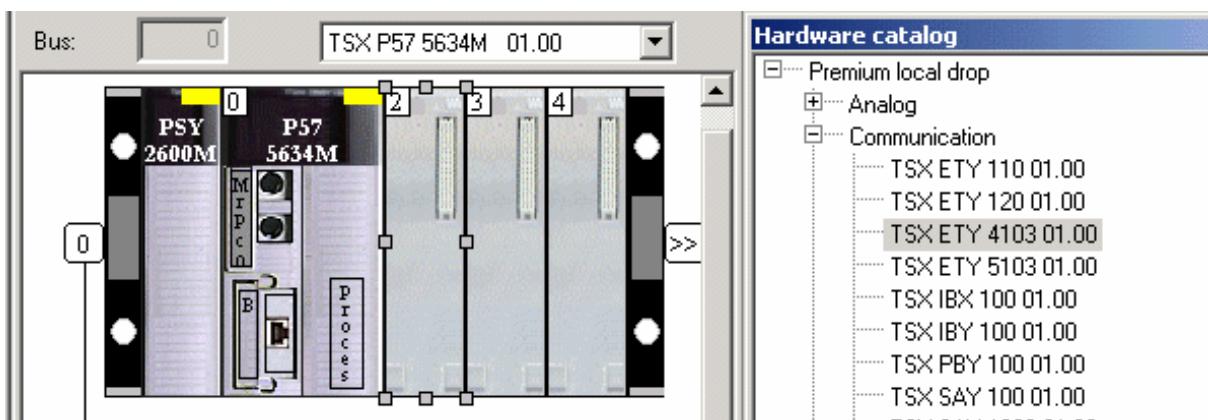
### 2. Configuração da rede lógica.

- Ativar a rede lógica a configurar (1)
- Configurar a rede Lógica: Endereço IP, global data, I/O scanning,... (2)



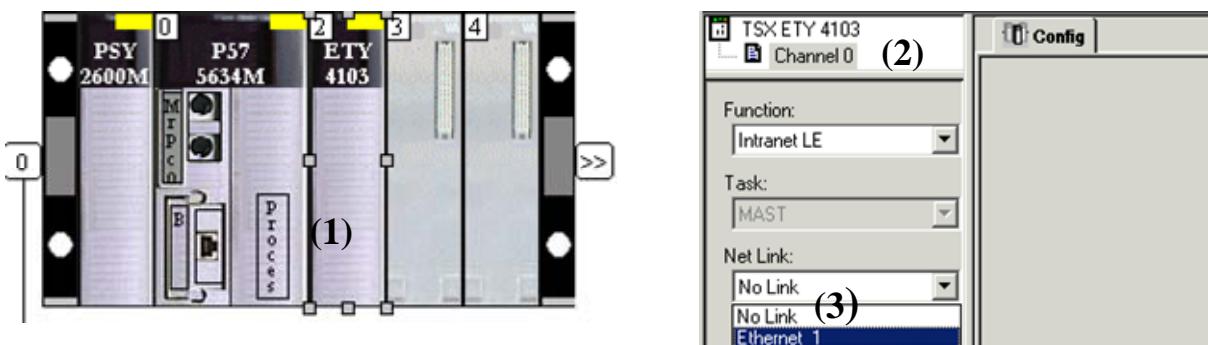
### 3. Definição do modulo de comunicação/cartão PCMCIA

- Definir o módulo de comunicação (arraste e cole do catálogo de hardware), ou definir o cartão PCMCIA (duplo clique na posição do cartão e adicionar o sub-módulo)



### 4. Associação do módulo / cartão com a rede lógica.

- Abrir o módulo de comunicação (1), clicando duas vezes sobre o mesmo.
- Selecionar o canal (2)
- Associar o módulo a rede Lógica (3), criada anteriormente.



# CAPÍTULO 7

## Variáveis

---



# Variáveis

## Introdução

Uma Variável pode ser do tipo de dado BOOL, WORD, DWORD etc. Seu conteúdo pode ser modificado durante a execução do programa.

## Tipos de Variáveis

**Variáveis Elementares:** São variáveis associadas a um único elemento, por exemplo, Entrada Digitais, analógicas etc.

**Variável Derivada:** é uma variável pertencente a um conjunto de elementos do mesmo tipo de dado (ARRAY) ou de vários tipos de dados (Structure).

**Variável Alocada:** É uma variável mapeada em um módulo de I/O ou associada a uma memória .

Por exemplo, a pressão da água é associada a uma memória %MW102, pressão da água é uma variável alocada.

**Variável Não Alocada:** É uma variável que não está mapeada ao módulo de I/O ou associada a uma memória.

**Variável Pública:** É uma variável com algum bloco de função. Estas variáveis transferem valores para blocos de função, são usadas para “setar” parâmetros do bloco de função.

**Variável Privada:** é uma variável usada por bloco de função, estas não são acessíveis pelo programa de aplicação.

**I/ODDT:** É a abreviação de Input/Output Derived Data Type. O termo I/ODDT designa uma estrutura de tipo de dado representando um canal do CLP, cada modulo específico processa seus próprios I/ODDT.

**Constantes:** Pode ser do tipo INT, DINT ou REAL. E esta pode ser usada em endereçamento direto (%KW, %KD ou %KF). As constantes não podem ser modificadas pelo programa de aplicação.

Cada variável deve ser declarada no editor de variáveis antes de serem usadas:

Através de duplo clique na área de Variables & FB Instances na árvore do project browser em vista estruturada ou durante a programação com duplo clique em uma entrada/saída indicada.

Um tipo de dado “data type” precisa ser indicado para cada variável  
O software Unity Pro fornece tipos de dados elementares e derivados

## Tipos e Faixa de dados comuns

- **BOOL/EBOOL:** Variáveis Booleanas “BOOL” precisam ser FALSA (0) ou VERDADEIRA (1). As variáveis EBOOL permitem a função “force” e a detecção de borda.
- **WORD :** Representa uma string de 16 bits, significa que o comprimento do dado é de 16 bits.
- **INT :** Representa um valor inteiro. A faixa de valores são -32768 até 32767
- **UINT :** Representa um valor inteiro não sinalizado (+ ou -). A faixa de valores são 0 até 65535.

- **REAL** : Representa um valor com ponto flutuante. A faixa de valores são -3.40e+38 até 3.40e+38

	Sintaxe	Formato	Exemplo
Bit	%M<i> or %MX<i>	1 bit (EBOOL)	%M1
Word	%MW<i>	16 bits (INT)	%MW10
Bit extraído de uma word	%MW<i>.<j>	1 bit (BOOL)	%MW15.5
Double word	%MD<i>	32 bits (DINT)	%MD8
Real (ponto flutuante)	%MF<i>	32 bits (REAL)	%MF15



A sintaxe dos tipos de dados: "Double word" e "Real" não estão disponíveis no M340. Neste caso, devem ser alocados estes tipos de dados em um tipo de dado inteiro %MW.

#### Legenda:

<i>: representa o número da instância (começa em 0 para o Premium e 1 para o Quantum)

#### Variáveis booleanas e tipos de dados permitidos:

Variável	Tipo
Bit interno	EBOOL
Bit de sistema	BOOL
Bit extraído de uma Word	BOOL
<b>%I (Entradas)</b>	
Bit de erro do módulo	BOOL
Bit de erro do canal	BOOL
Bit de entrada	EBOOL
<b>%Q (Saídas)</b>	
Bit de saída	EBOOL

#### Variáveis de sistema:

%S >> são variáveis internas do CLP

## Regras para Entrada de valores Literais

- Valores Literais são usados para atribuir valores a pinos, ou para atribuir constantes a variáveis, e não podem ser alterados pelo programa.
- **Pode-se entra com valores tais como:**
  - Base 2 (binário) 2#1111111111111111
  - base 8 (octal) 8#177777
  - base 10 (decimal) 65535 (no 10# needed)
  - base 16 (hexadecimal) 16#FFFF



Todos os valores acima são os mesmos, eles foram escritos com bases diferentes.

## Regras para Nomeação de Variáveis

Regras a serem obedecidas para nomear variáveis:

- Máximo de 32 caracteres, podendo iniciar com número.

### Não alocadas

- Nome “tag” sem um endereço de hardware
- Variáveis não alocadas não podem ser cíclicas. Se for necessária variável cíclica no projeto, usar variáveis locadas.

### Locadas

- Nome “tag” com endereço de hardware

### Constantes

- Variáveis com proteção de escrita
- Usar um valor fixo para atribuir a uma variável

## Endereçamento Direto de variáveis

Cada endereço direto tem uma referência que indica sua posição na seqüência e se for um endereço de entrada (somente leitura) ou um endereço de saída (leitura/escrita).

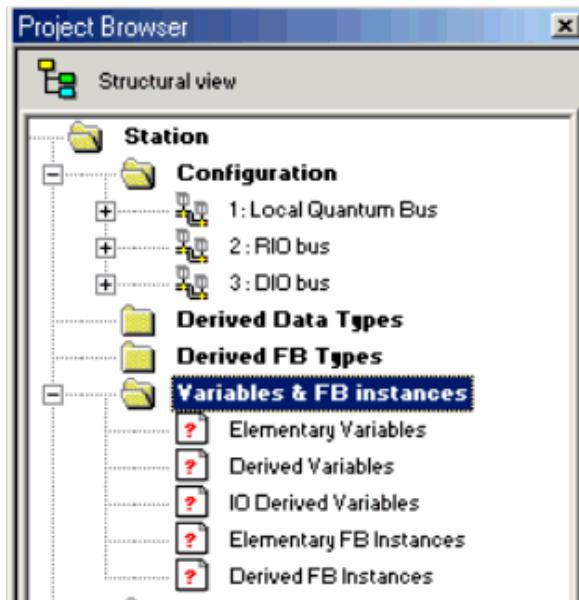
- Área 0x / %Qx = bit de saída (digital) - exemplo 1: 000001 saída digital 1, exemplo 2: %Q00001 é o bit de saída 1
- Área 1x / %Ix = bit de entrada digital: exemplo 1: 100017 é a entrada digital 17 - exemplo 2: %I00017 é o bit de entrada 17
- Área 3x / %IWx = registros de entrada, - exemplo 1: 300300 é o registro de entrada 300  
- exemplo2: %IW000300 é a Word de entrada 300
- Área %QWx = registro de saída - exemplo1: 400029 é o registro de saída 29.

## Acesso as variáveis

Com duplo clique na pasta Variables &FB instances, aos mostrados os sub-grupos das variáveis elementares e derivadas.

Diretório das variáveis:

O diretório das Variáveis e instancias FB da vista estruturada do projeto, permite o acesso as variáveis (EDT, DDT,, IODDT) e aos blocos de função (EFB, DFB).



A figura seguinte mostra um exemplo de arvore de diretório das Variáveis & instancias FB.

## Edição de Variável

Ao dar duplo clique na pasta Variables &FB instances, juntamente com os sub-grupos das variáveis elementares e derivadas é mostrada ainda o editor de dados “data editor” a seguir. Qualquer uma das colunas é editada com dois cliques na mesma.

The screenshot shows the Data Editor window with the title 'Data Editor'. It has tabs for 'Variables', 'DDT Types', 'Function Blocks', and 'DFB Types'. A filter bar at the top includes fields for 'Name', checkboxes for 'EDT', 'DDT', and 'IODDT', and a 'Filter' button. The main area is titled 'Identificador' and contains a table of variables:

	Type	Address	Value	Comment
heating_duration_int	INT		-2000	Variável não alocada
heating_monitoring	SFCSTEP...			
delay	TIME			
min	TIME		t#1s	
max	TIME		t#5s	
heating_step_max_INT	INT		5000	
heating_step_min_INT	INT		1000	
Ind	INT			
Index	INT			
Ls_close	BOOL		1	
Ls_open	BOOL		0	
Machine_running	EBOOL	%Q0.30		Output for Mach...
Material_A	INT			Value material A

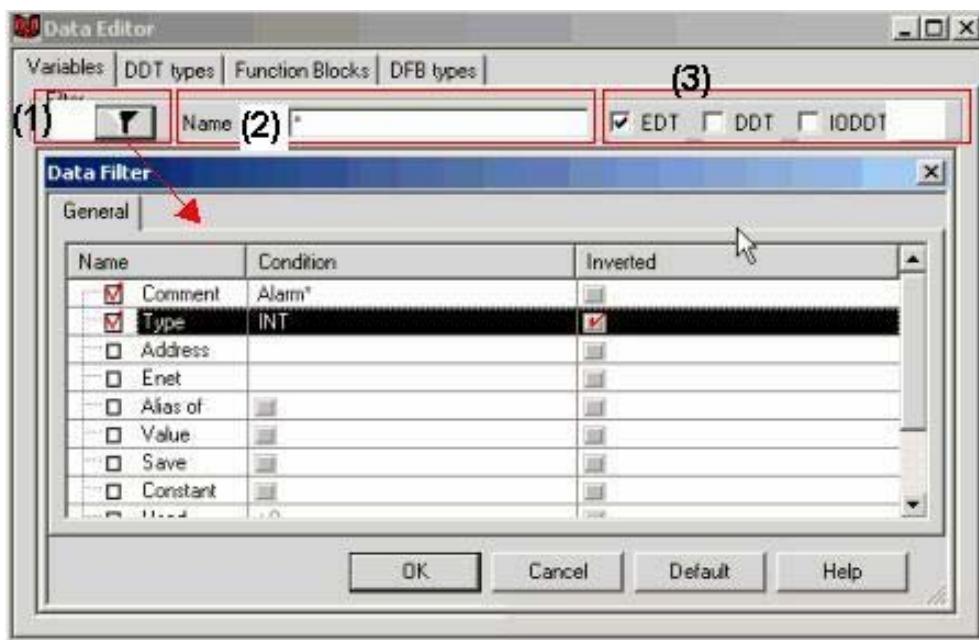
A yellow box on the left says 'Duplo click na célula para entrar em edição'. Three callout boxes point to specific rows: one points to the first row with the text 'Variável não alocada'; another points to the 'heating\_step\_max\_INT' row with the text 'Tipo do Dado'; and a third points to the 'Machine\_running' row with the text 'Var. Alocada'.

## Filtro de variáveis

O filtro permite visualização versátil de uma parte do banco de dados das variáveis. Com duplo clique no botão abaixo de “filter” (1) pode ser selecionado vários parâmetros na coluna “name” tais como; tipo, Comentários etc. e uma condição precisa ser dada na coluna “Condiction”, a condição pode ser invertida por uma caixa de checagem na coluna “Inverted”. No exemplo mostrado, o filtro mostrará todas as variáveis que contenham o comentário alarm no inicio do documentário e sejam do tipo é diferente de INT (Inteiro). O nome de uma variável pode ser diretamente filtrado (2), para não filtrar o nome deve-se especificar “asterístico” (\*), onde serão mostradas todas as variáveis independentes do nome. O filtro direto no tipo geral da variável é também possível pela marcação das caixas de checagens EDT (Tipo de dados elementares - Elementary Data Types), DDT (Tipo de dados

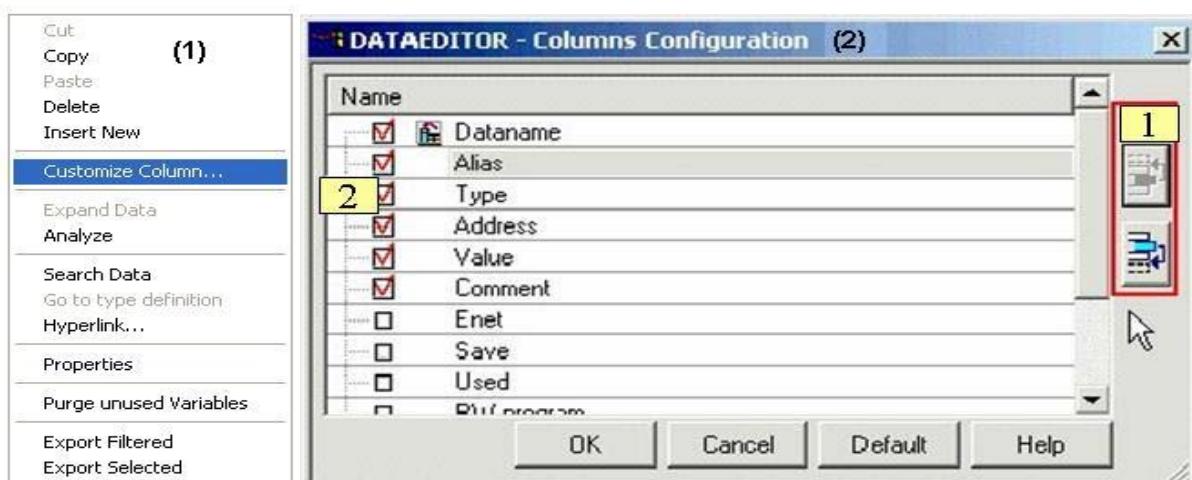
## Variáveis

Derivados – Derived Data Types) ou IODDT (Tipo de dados derivados de Entradas/Saídas-Input/Output Derived Data Type) (3).



### Configuração das colunas a serem visualizadas.

Com um clique com botão direito sobre o “data editor” abre a lista com diversos recursos de configuração do mesmo, inclusive “Customize Column...” (1) que ao ser selecionado permite o ajuste das colunas desejadas e sua posição no “data editor” (2).

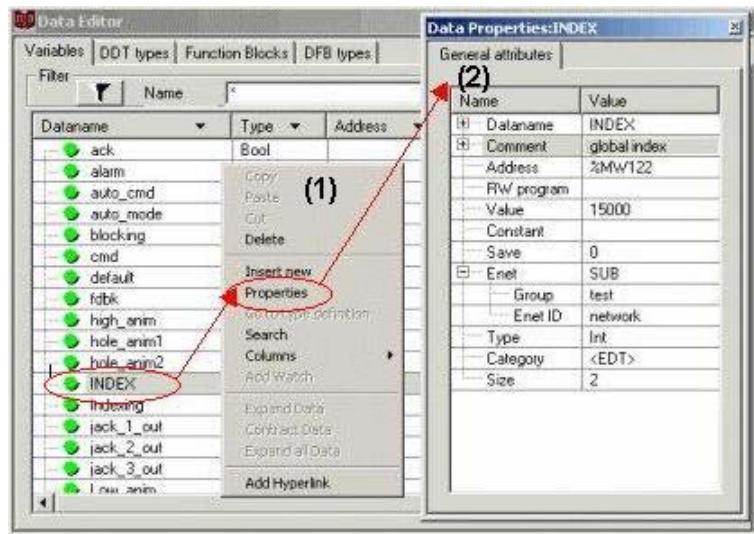


Na caixa de diálogo anterior “DATAEDITOR – Columns Configuration (2) temos em:

- 1** As colunas podem ser movidas usando os botões da direita
- 2** Uma marca em frente a coluna , a mesma será mostrada

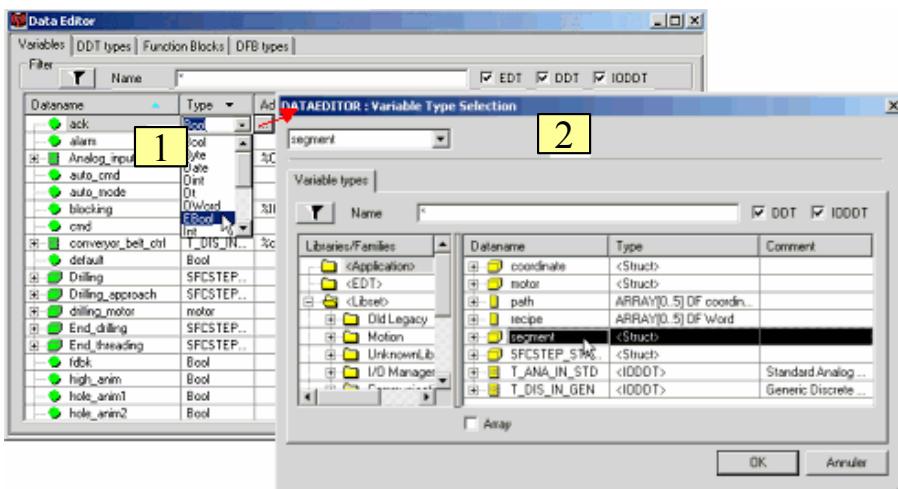
## Propriedades dos Dados

Mesmo se as Colunas não forem visualizadas, todos os parâmetros de uma variável podem ser mostrados na janela do editor de propriedades “Data Properties:<nome da variável>”. Com um clique com botão direito sobre o nome da variável no “data editor” abre a lista com diversos recursos de configuração da mesma, inclusive “Properties...” (1) que ao ser selecionado permite a visualização de todos os parâmetros da variável (2).



## Editor do tipo de dado – Data Type Editor

O tipo de dado pode ser selecionado a partir de uma lista previamente dos tipos usados [1] ou diretamente da seleção do tipo de dado clicando no botão [...] [2].



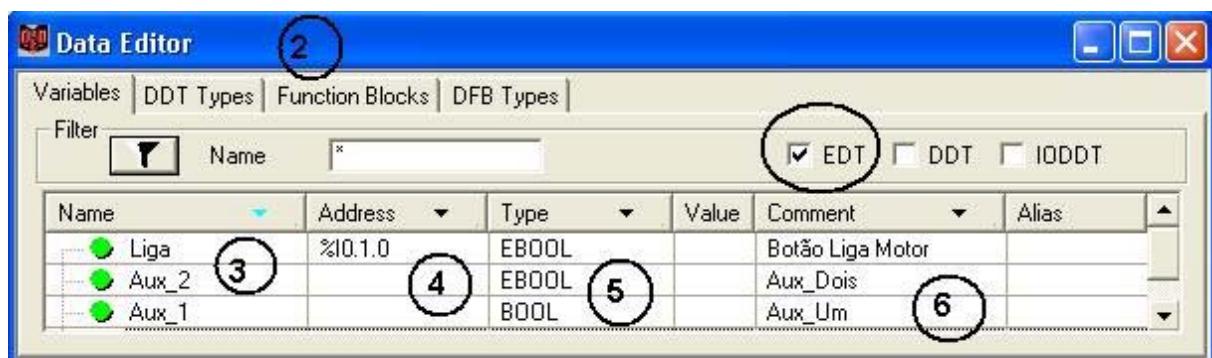
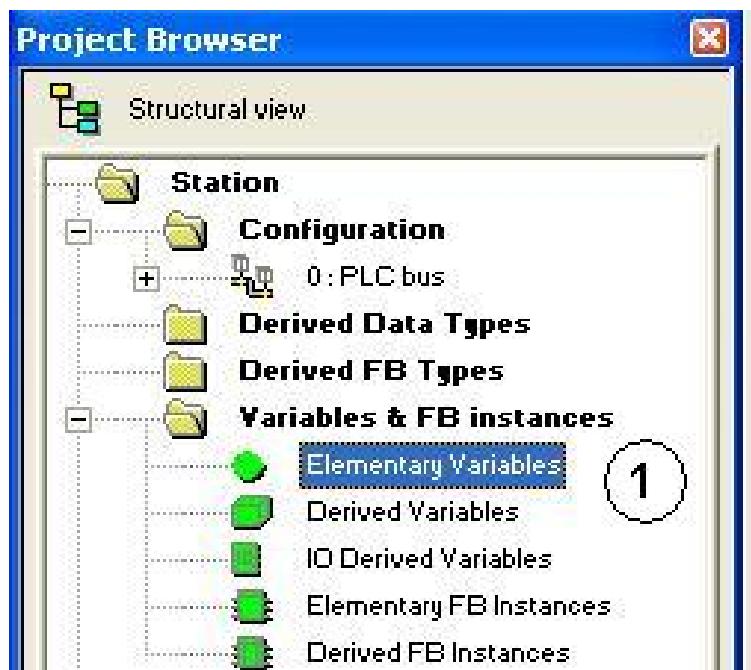
## Criando Variáveis Elementares

As variáveis elementares podem ser criadas de três maneiras:

- A partir da pasta “Elementary Variables”,
- A partir da pasta “Configurations”,
- Durante a edição do programa.

### *Criando variáveis elementares a partir da pasta “Elementary Variables”*

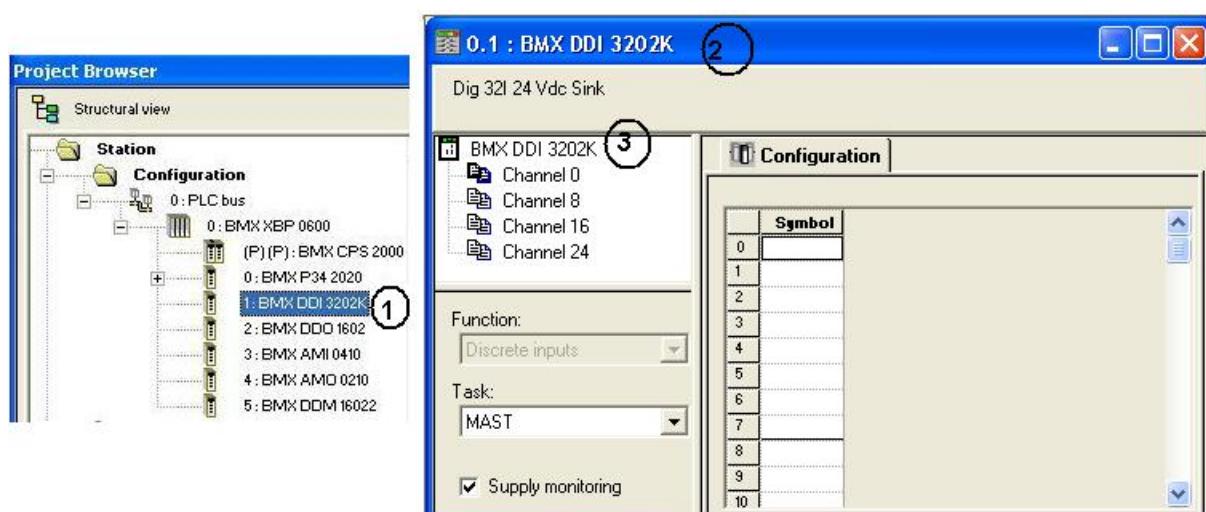
Com dois cliques em “Elementary Variables” (1). É visualizado o editor de dados (2) com a opção EDT(Tipo de dado Elementar) marcada, no editor deve ser especificado o nome da variável (3), selecionar endereço (4), tipo de dado sendo EBOOL para variáveis alocadas e indiferente para variáveis não alocadas (5) e comentário (6).



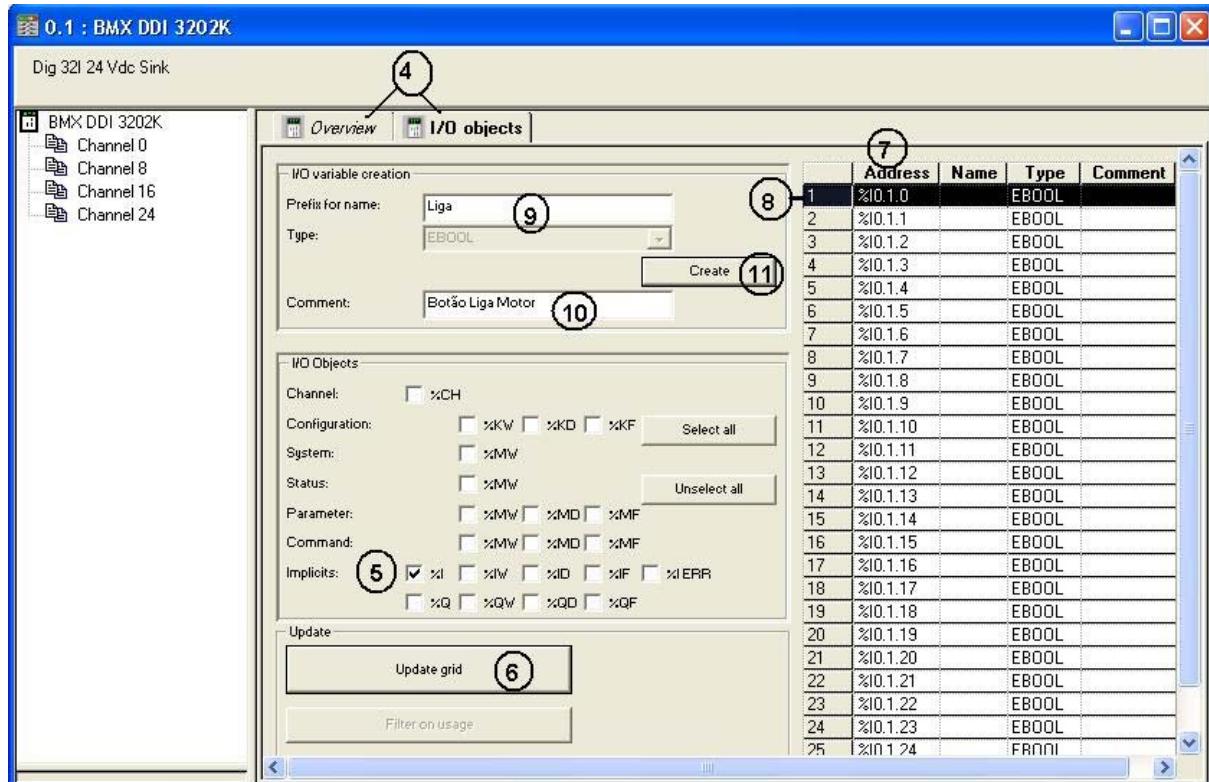
### Criando variáveis elementares a partir da pasta “Configurations”

A outra maneira de declarar variável elementar é através da Edição de Objetos de I/O a partir do “Project browser” em vista estrutura:

Na pasta “Configurations” do “Project browser”, dar dois cliques no módulo módulo desejado (1) sendo aberta a caixa de configuração do módulo (2).

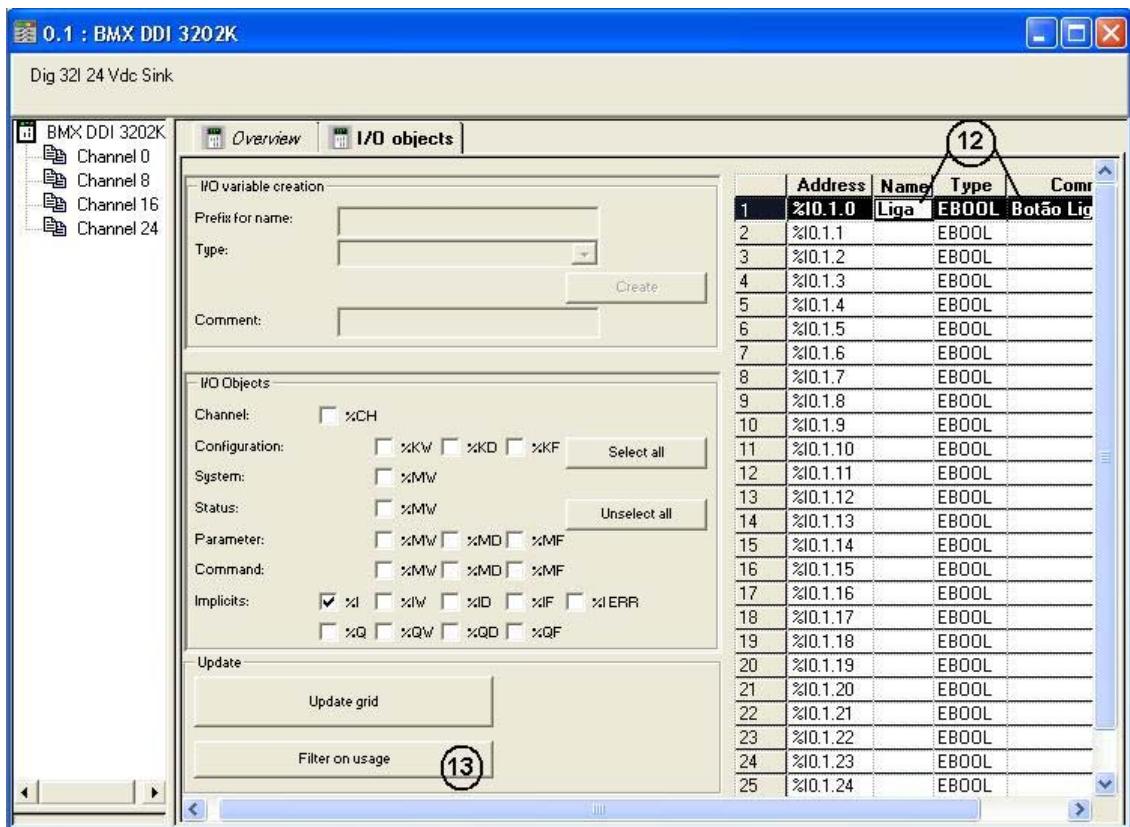


Ao clicar no modelo do módulo (3), são visualizadas na figura seguinte, as abas “Overview” e “I/O Objects” (4) clicando na aba “I/O Objects” deve-se especificar os objetos desejados (5) e acionar o botão “update grid” (6) sendo mostrado agora toda a faixa de endereços (7) referentes ao objeto selecionado, ao selecionar um endereço (8) é possível informar o nome (9) e comentário (10) do objeto criado finalizando, deve-se clicar em “Create” (11).

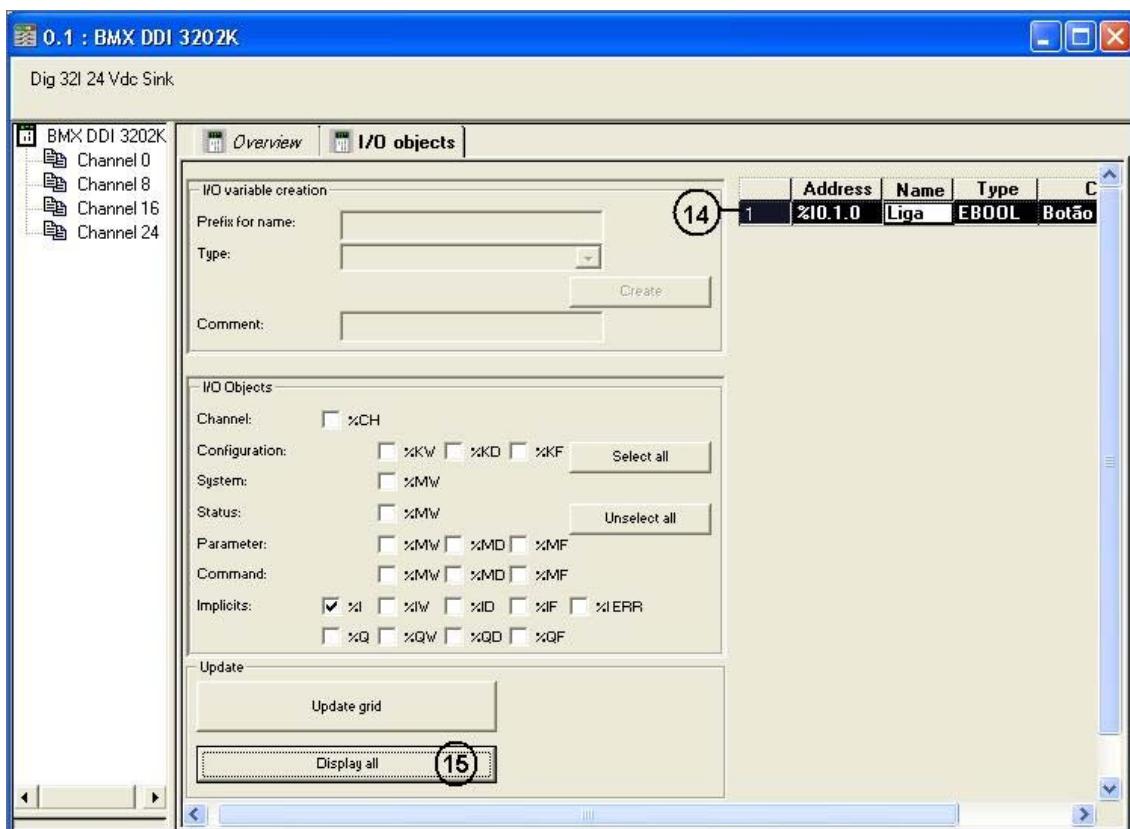


Ao clicar em “Create” (11) são atualizadas as colunas de configuração dos objetos (12). É possível agora filtrar a visualização dos objetos ao clicar no botão “Filter on usage” (13).

## Variáveis

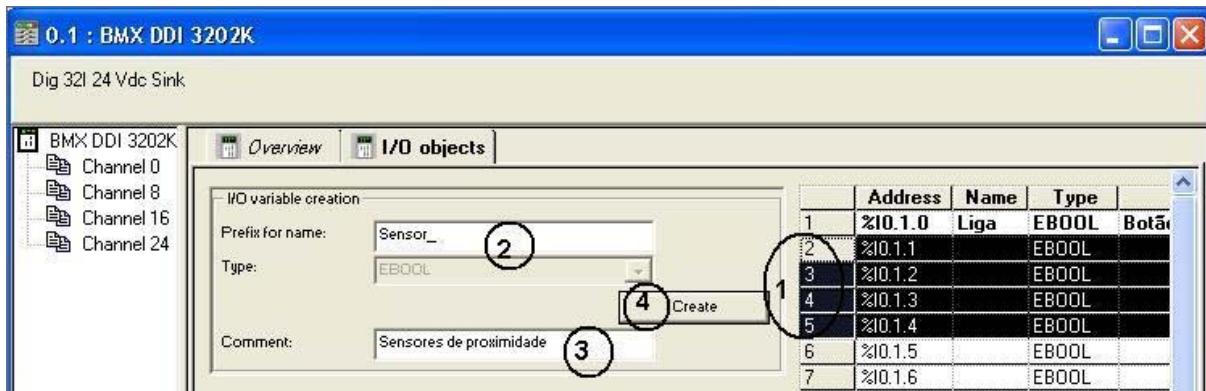


Ao clicar no botão “Filter on usage” (13), são visualizados apenas os objetos utilizados no projeto (14). Para a visualização de todos os objetos deve-se clicar no botão “Display All” (15).

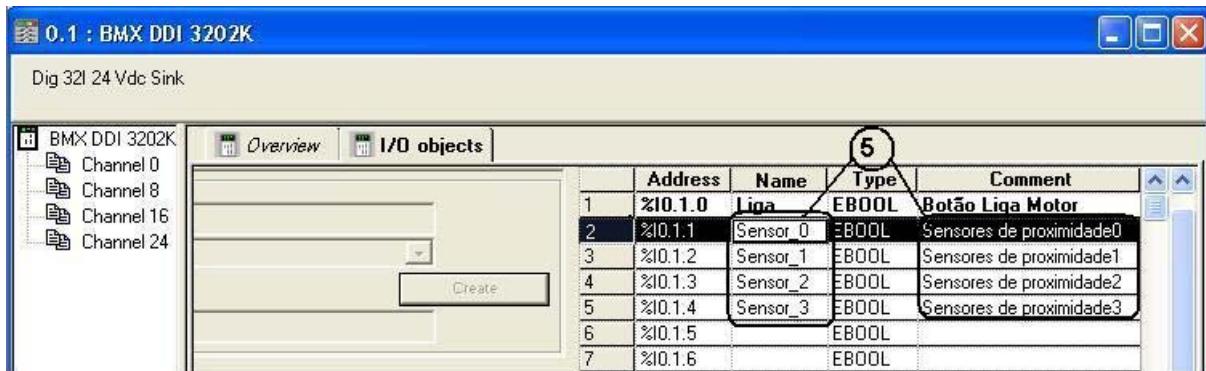


O mesmo processo pode ser realizado quando se deseja criar um grupo de objetos que se relacionem, por exemplo, pela aplicação em uma determinada etapa do processo, para isto

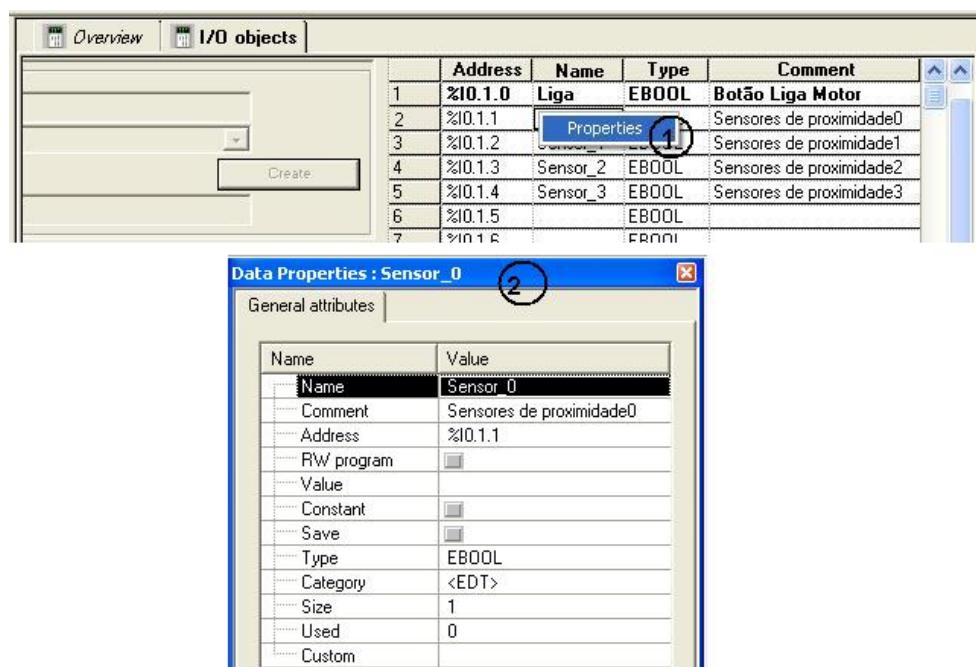
deve-se selecionar a quantidade de objetos desejados (1) e especificar um nome, agora prefixo (2) e comentário (3) para os mesmos e clicar em “create” (4).



Ao clicar em “Create” (4), as informações dos objetos são atualizadas (5).



Para fazer alterações em um determinado objeto deve-se clicar com o botão direito sobre o mesmo e clicar em “Properties” (1), será visualizado então a caixa de propriedades do mesmo (2), fazer a alteração desejada e validar acionando Enter.

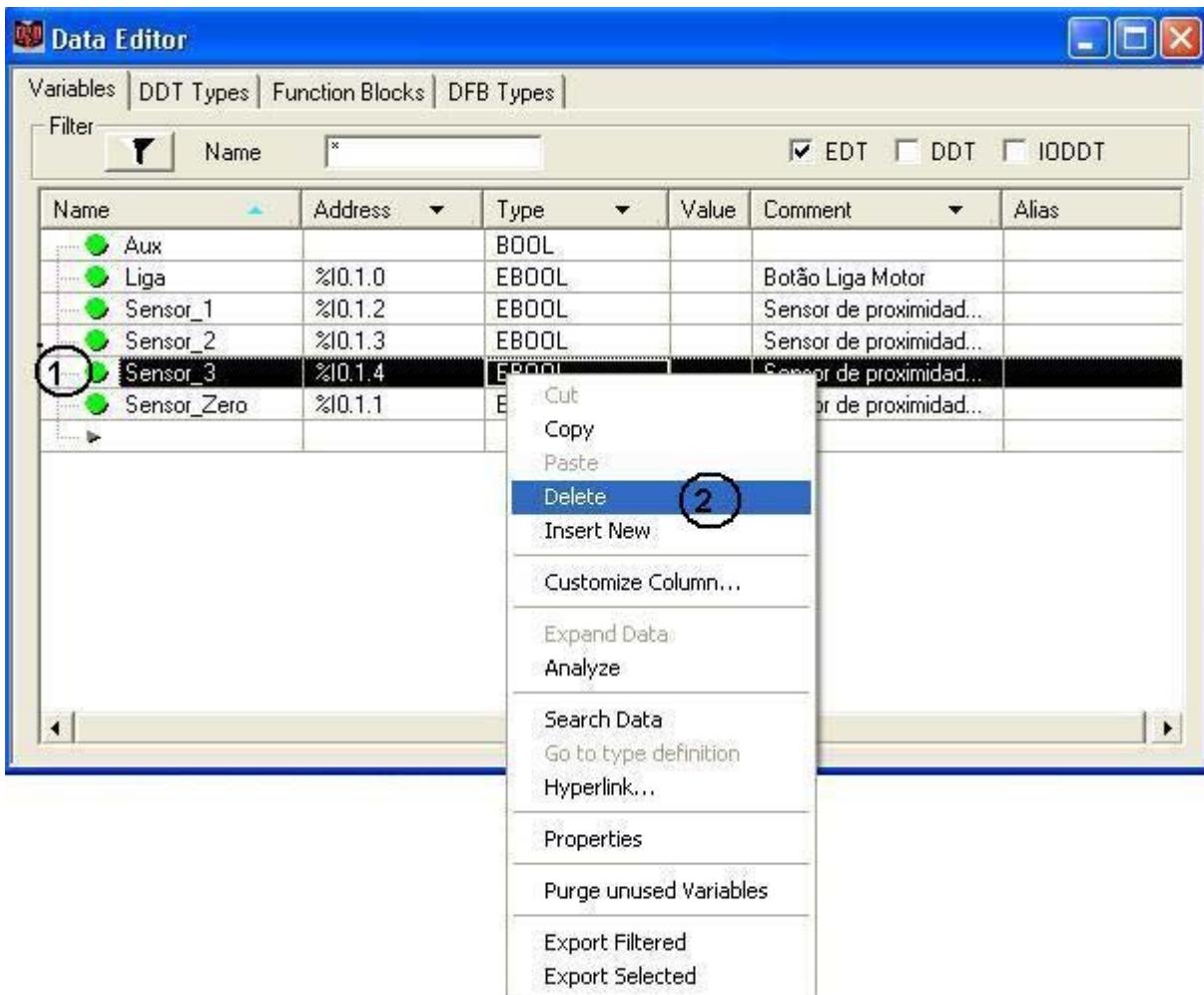


A cada alteração, a informação do mesmo será atualizada (3).

The screenshot shows a table titled "I/O objects" with columns: Address, Name, Type, and Comment. The table contains the following data:

	Address	Name	Type	Comment
1	%I0.1.0	Liga	EBOOL	Botão Liga Motor
2	%I0.1.1	Sensor_Zero	EBOOL	Sensor de proximidade Zero
3	%I0.1.2	Sensor_1	EBOOL	Sensor de proximidade 1
4	%I0.1.3	Sensor_2	EBOOL	Sensor de proximidade 2
5	%I0.1.4	Sensor_3	EBOOL	Sensor de proximidade 3
6	%I0.1.5		EBOOL	
7	%I0.1.6		EBOOL	

Para apagar um objeto, deve-se, a partir do “Project browser” acessar a pasta “Elementary Variables” selecionar o objeto **(1)** e clicar com o botão direito sobre o mesmo e selecionar “Delete” **(2)**.



Ao selecionar “Delete” **(2)**, o objeto é apagado da lista do “Data Editor”.

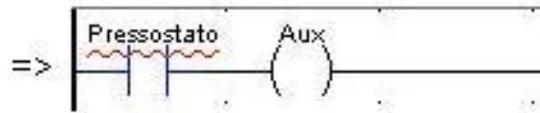
**Data Editor**

Variables   DDT Types   Function Blocks   DFB Types						
Filter	Name	*	<input checked="" type="checkbox"/> EDT	<input type="checkbox"/> DDT	<input type="checkbox"/> IODDT	
Name	Address	Type	Value	Comment	Alias	
Aux		BOOL				
Liga	%I0.1.0	EBOOL		Botão Liga Motor		
Sensor_1	%I0.1.2	EBOOL		Sensor de proximidad...		
Sensor_2	%I0.1.3	EBOOL		Sensor de proximidad...		
Sensor_Zero	%I0.1.1	EBOOL		Sensor de proximidad...		

*Criando variáveis elementares durante a edição do programa.*

Ao ser colocado no programa uma variável não declarada “desconhecida”, o editor a indica sublinhada com traço vermelho, como pode ser visto a seguir.

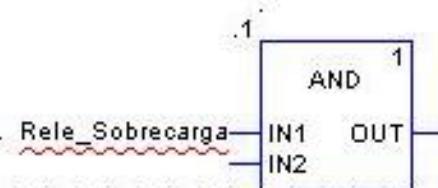
Linguagem Ladder



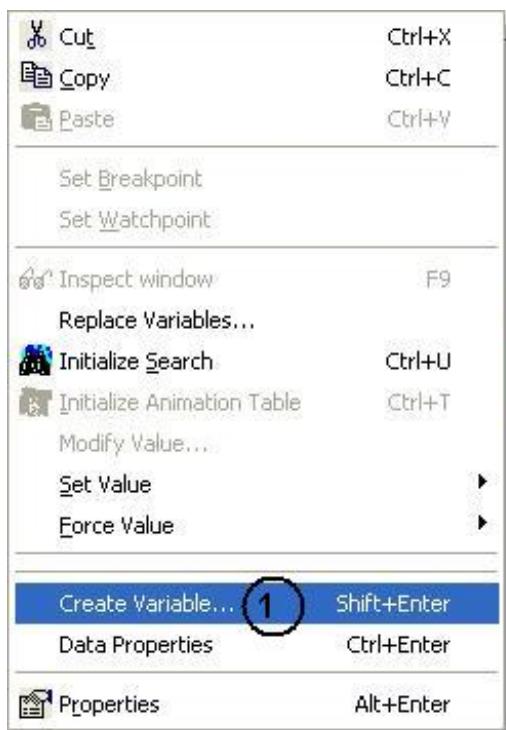
Linguagem de Texto Estruturado => Pressostato2:= Aux;

Linguagem de Lista de Instrução => LD Exaustor  
ST Motor

Linguagem de Diagrama de Blocos Funcionais => Rele\_Sobrecarga



Ao clicar sobre uma variável não declarada, abre a caixa de diálogo de ações na mesma, selecionar “Create variable” (1) e será visualizada a caixa de criação de variáveis “Create variable?” (2).

**Variáveis**


Na caixa de diálogo ‘Create variable?’ deve-se selecionar o tipo (1), especificar o endereço (2) para variável alocada assim como o comentário (3) se for o caso, para confirmar a criação clicar em (4), no caso de ignorar o aviso clicar em (5) e para esconder o endereço e comentário, clicar em (6).





# CAPÍTULO 8

## Gerenciamento da Biblioteca de Funções

---



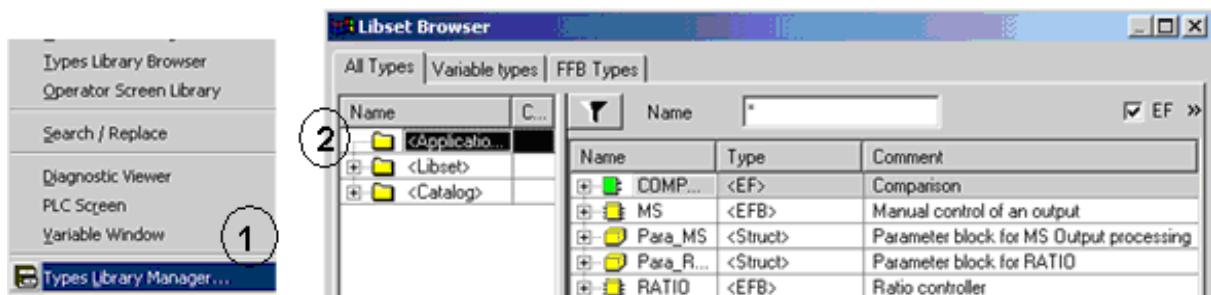
# Gerenciamento da Biblioteca de Funções

## Visão Geral

- Contém todos os objetos disponíveis para desenvolver um projeto de automação.
  - EFs (Funções elementares - Elementary Functions)
  - EFBs (Blocos de Função Elementares - Elementary Functions Blocks)
  - DFBs ( Bloco de Função derivada - Derived Function Blocks)
  - DDTs (Tipo de dados Derivado - Derived Data types)
- Fornece um conjunto de funcionalidades permitindo modificar o conteúdo da biblioteca.
- Permite a transferência entre bibliotecas e projetos.

## Gerenciamento do tipo de biblioteca

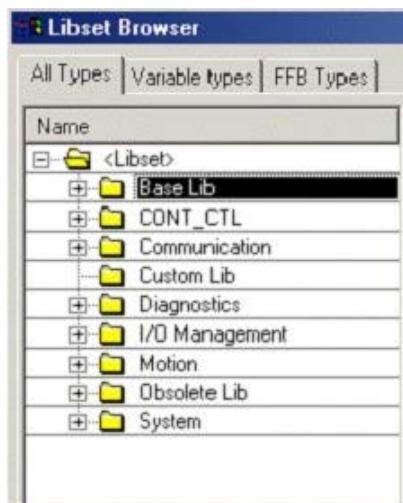
O Gerenciamento do tipo de biblioteca, “Types Library Manager” é acessível a partir da barra de ferramenta “Tools” (1) a biblioteca é organizada em três tipos; (2) “Application” (Funções utilizadas na aplicação), “Libset” (Biblioteca global) a “Catalog” (funções relativas a certos hardwares particulares).



O gerenciamento da biblioteca permite:

- Modificar o conteúdo da biblioteca Global **Libset** => adiciona uma nova biblioteca / família, apagar biblioteca / família....
- Criar uma biblioteca Local **Application** => transferir da biblioteca global.
- Verificar a versão e a coerência dos objetos (global e biblioteca local)
- Apagar um tipo de biblioteca não utilizada.

## Bibliotecas Fornecidas na biblioteca global “Libset”

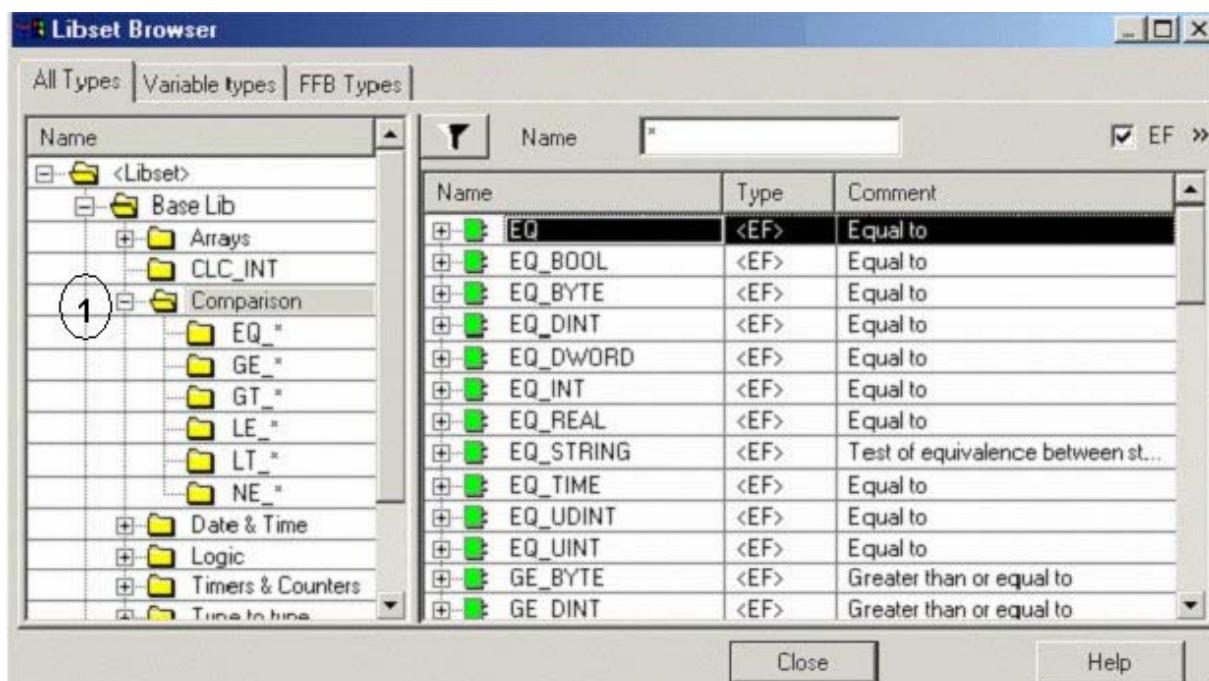


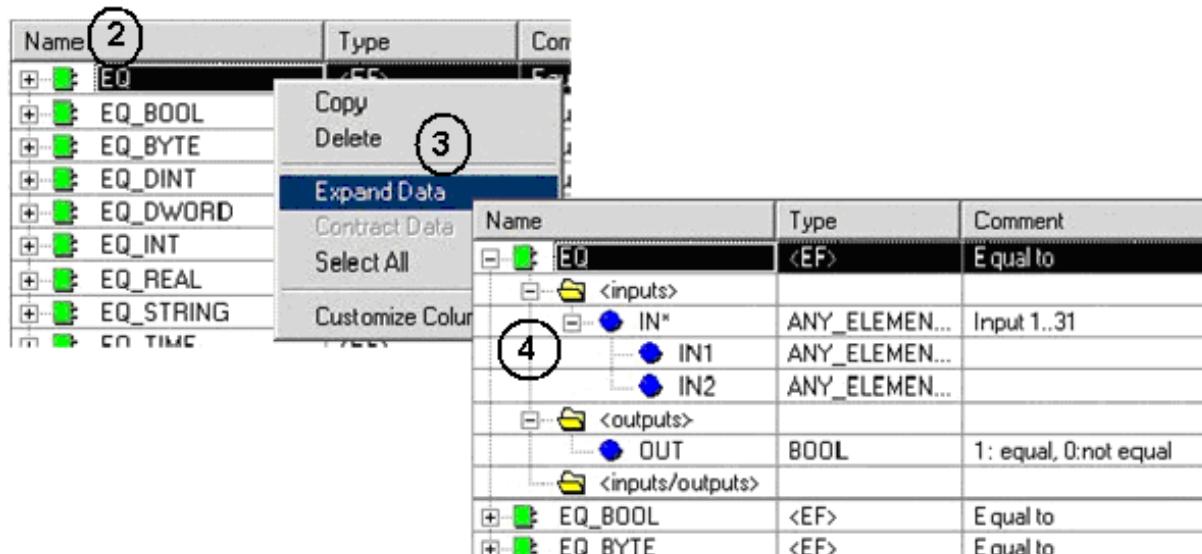
- **BASE LIB** : com os grupos Arrays, CLC\_INT, Comparison, Date & Time, Logic, Mathematics, Statistical, Strings, Timers & Counters, Type to Type
- **CONT\_CTL** : com os grupos Conditioning, Controller, Mathematics, Measurement, Output Processing, Setpoint Management
- **COMMUNICATION** : com grupos estendidos contendo 28 FFBs de comunicação
- **DIAGNOSTICS** : com um grupo de diagnósticos contendo 15 FFBs de Diagnóstico.
- **I/O MANAGEMENT**: com os grupos Analog I/O Configuration, Analog I/O Scaling, Explicit Exchange, Immediate I/O, Interbus\_S, Quantum I/O Configuration
- **MOTION** : com os grupos de controle de eixos Axis Control, CAM Control, MMF Start
- **OBSOLETE LIB** : com os grupos CLC, CLC\_PRO, Extensions/Compatibility
- **SYSTEM** : com os grupos Events, SFCManagement, SysClock

## Expansões do Bowser Libset

O Browser libset possui vários níveis, podendo chegar até ao detalhamento e edição das entradas em uma determinada função.

A expansão pode ser feita clicando diretamente na caixa **[+]** (1) ou selecionando o objeto desejado (2) e com botão direito selecionar “Expand Data” (3) resultando em um maior detalhamento do mesmo (4).



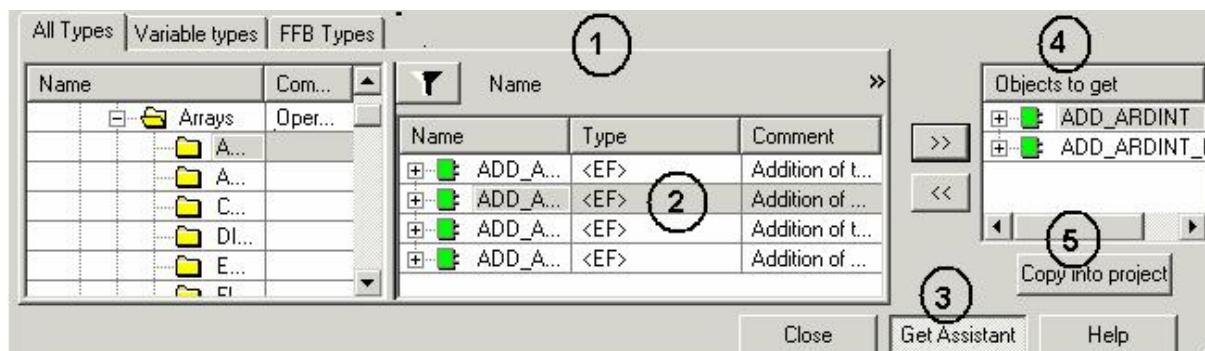


## Transferência para uma biblioteca local

Abrir o gerenciador do tipo de biblioteca “Types library manager” para transferir um objeto da biblioteca global “Libset” para a biblioteca local “Application”, isto é fortemente indicado para selecionar os objetos a serem utilizados na aplicação, resultando em melhoria da performance no desenvolvimento do projeto e na organização do mesmo.

Procedimento:

Acessar o “Types library manager” (1) selecionar o objeto desejado (2) e acionar o botão “Get Assistant” (3), com isto a parte “Objects to Get” (4) é aberta, acionar o botão >> para adicionar o objeto a lista de transferência ou << para retirar o objeto da lista de transferência. Para copiar o objeto no projeto, portanto biblioteca local “Application” deve-se acionar o botão “Copy into project” (5).

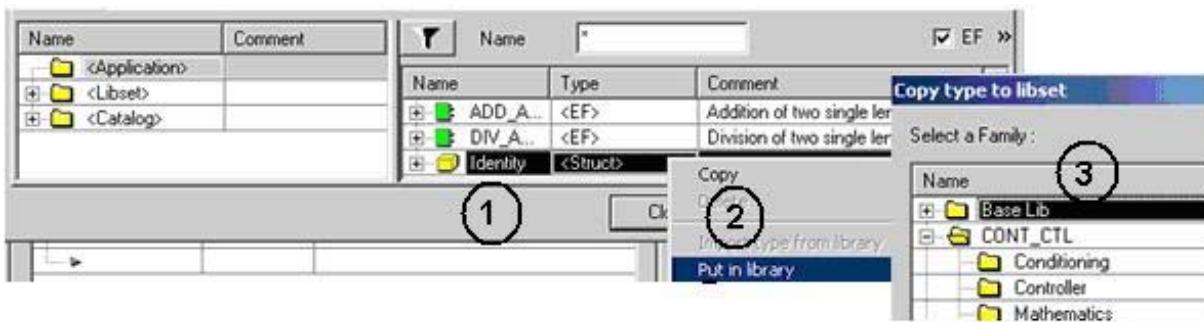


## Transferência para a biblioteca global

Abrir o gerenciador do tipo de biblioteca “Types library manager” para transferir um objeto da biblioteca local “Application” para a biblioteca global “Libset”, isto é fortemente indicado para selecionar os objetos utilizados em uma determinada aplicação, disponibilizando-os para uso posterior em outras aplicações, resultando em melhoria da performance no desenvolvimento de projetos e na organização dos mesmos.

Procedimento:

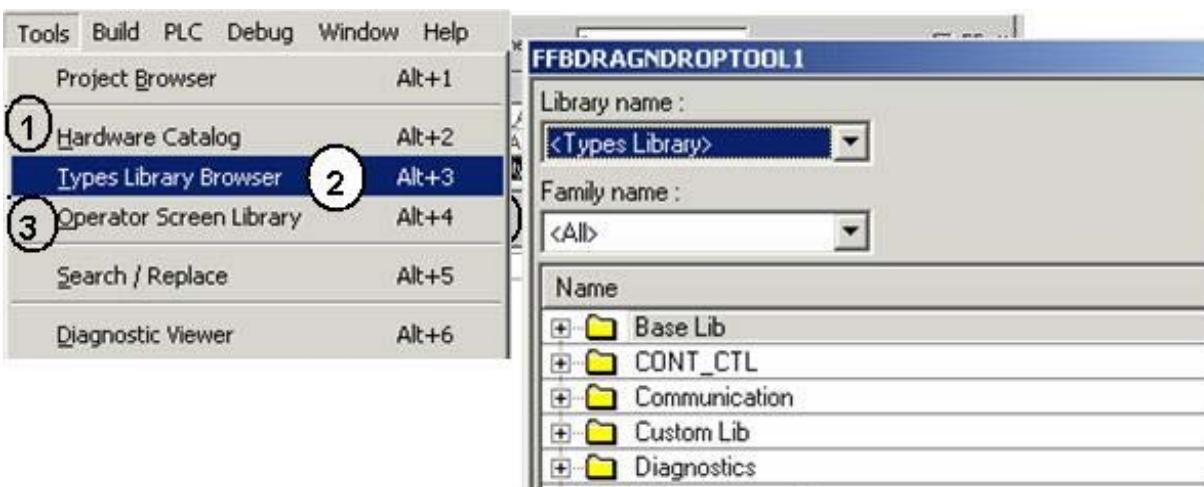
1. Selecionar o objeto a ser transferido (por ex. estrutura Identy)
2. Colocá-lo na biblioteca (clicar com botão direito no objeto)
3. Selecionar a biblioteca destino / família a transferir ( botão OK)



## Acesso a biblioteca

Clicar no menu “Tools” para acessar catálogos “Hardware Catalog” (1), bibliotecas “Types Library Browser” (2), ou tela do operador “Operator screen library” (3).

- Hardware Catalog: Lista de equipamentos de Hardware
- Types library Browser: lista dos tipos DDT, EF, EFB e DFB
- Operator screen library: lista de objetos para definir a tela do operador “operator screens”

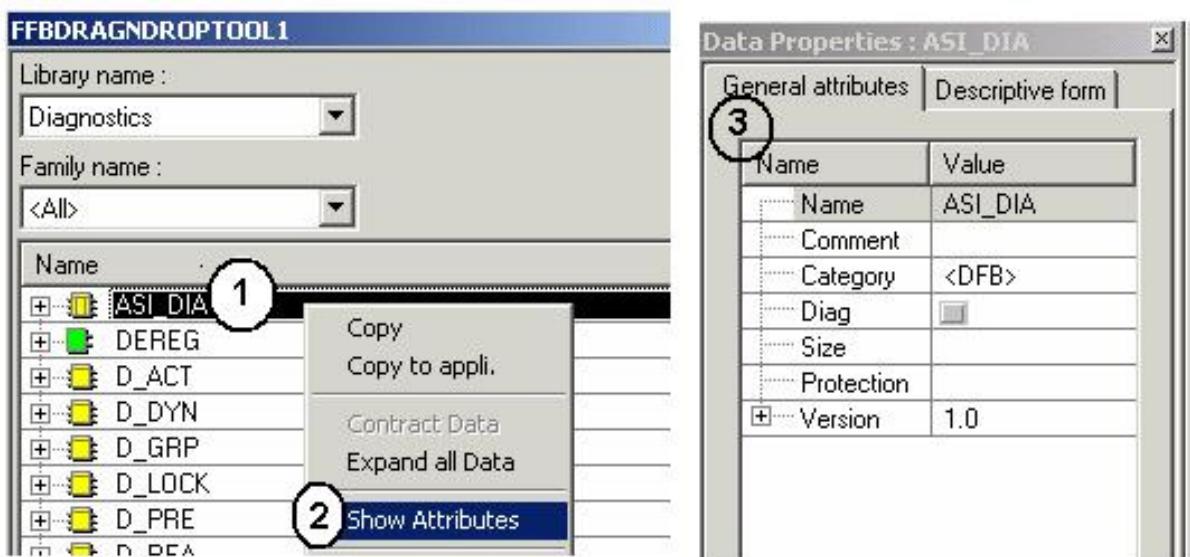


## Tipos de Browser de biblioteca

Através do browser de biblioteca é possível:

- Listar dos tipos de variáveis predefinidas (Array, EF, EFB, DFB)
- Arrastar e solta um tipo de variável para aplicação
- Mostrar os atributos “Show attributes” para visualização das propriedades do tipo de variável.

Para utilizar o browser de biblioteca, ao selecionar o objeto (1) clicar com botão direito e selecionar “Show Attributes” (2), será mostrada a caixa de diálogo “Data Properties” (3).

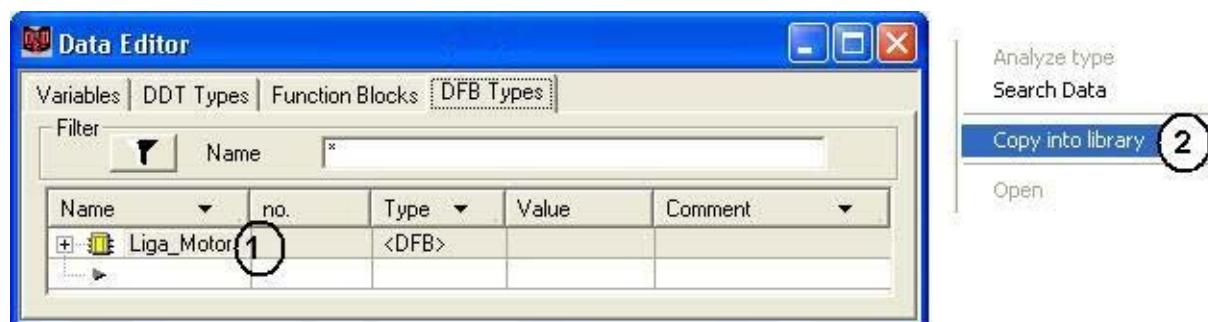


## Armazenar um “DDT” ou um “DFB” em uma biblioteca

Ao trabalhar com um tipo de dado derivado “DDT” ou bloco de função derivado “DFB”, tem-se informações e parâmetros específicos dos mesmos, tornando esta operação muito útil na aplicação e organização dos mesmos.

Procedimento:

Nas abas DDT ou DFB, selecionar o objeto a ser armazenado (1);  
Clicar com botão direito sobre o mesmo e selecionar Copy into library (2).



Na caixa “Copy type to library”, selecionar em Custom Lib (3) uma família (4), clicar no botão OK para copiar o objeto na biblioteca desejada.



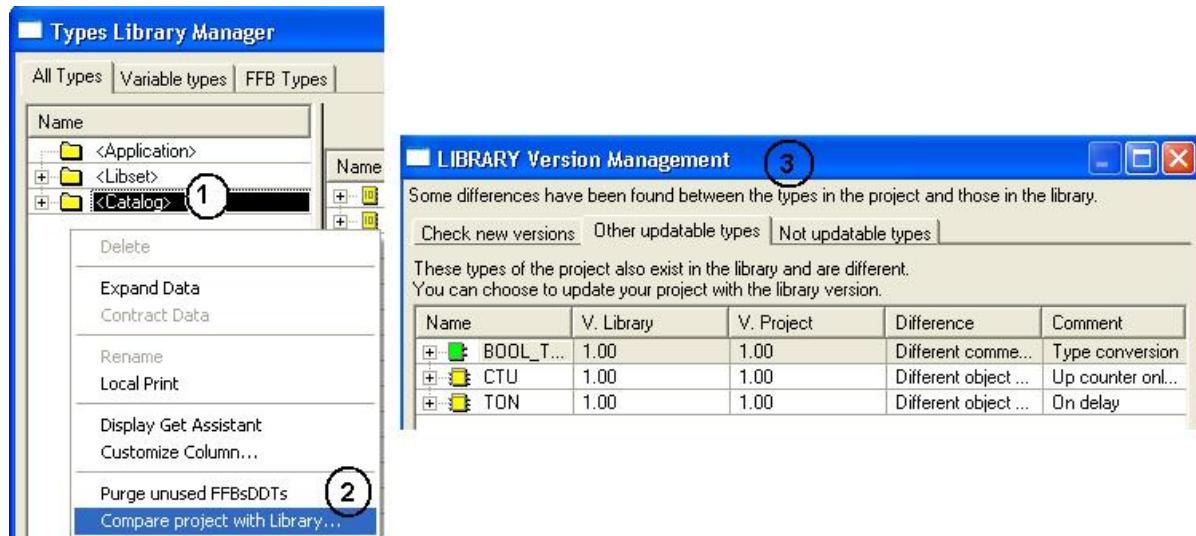
## Gerenciamento de Versão

Tem a função de verificar a coerência de versão dos objetos nas bibliotecas global e local.

- Verifica a versão dos objetos usados no projeto (na biblioteca local)
- Atualiza um ou todos os objetos do projeto.

Verificação da versão:

Em Types library manager, selecionar a biblioteca (1), clicar com o botão direito e selecionar “Compare project with Library..” (2), será mostrada a caixa de diálogo “LIBRARY Version Management (3), selecionar a aba e ação desejada e finalizar acionando **OK**.



# CAPÍTULO 9

## Linguagens de Programação



# Linguagem de Programação Ladder

## *Introdução*

A estrutura de um Diagrama Ladder corresponde a uma linha lógica de um diagrama de contato a rele.

O Trilho de alimentação é localizado no lado esquerdo do Ladder, correspondendo ao condutor fase no diagrama de contato a rele, já o trilho do lado direito do Ladder corresponde ao condutor neutro no diagrama de contato a rele.

A Programação em Ladder é regulamentada pela norma IEC 6 1131-3

Um grupo de objetos os quais são conectados entre si e não possuem conexão com outros elementos são chamados de “**Networks**” ou “**Rung**”.

A linguagem de programação em Ladder é orientada a célula, somente um objeto pode ser colocado em cada célula.

Uma seção Ladder consiste de uma simples janela paginada, esta página tem uma grade que é dividida em seções em linhas e colunas.

Uma largura de 11 a 63 colunas e 17 a 2000 linhas podem ser definidas na seção LD.

Limitações quanto ao número de seções:

Tarefa: MAST, FAST, AUXi	Sem limitação
Eventos EVT	1
Blocos DFBs	1
Sub-rotinas: SR	1

## **Objetos do Diagrama Ladder:**

Os objetos na linguagem de programação Ladder ajudam a dividir uma seção em um número de:

- Contatos;
- Bobinas;
- EFs e EFBs (Elementary Functions e Elementary Function Blocks);
- Elementos de controle;
- Operação e Blocos de comparação que representam uma extensão da IEC 6 1131-3.
- Estes objetos podem ser conectados entre si por links ou por parâmetros (somente FFBs)
- Comentários podem ser colocados na lógica da seção com objetos de textos.

## *Seqüência de execução em uma Networks/Rung*

A **Seqüência de Processamento** de um objeto individual em uma seção Ladder é determinada pelo fluxo de dados dentro da seção. Networks conectadas no trilho esquerdo são processadas do topo para a base. Networks que são independentes uma da outra dentro de uma seção são processadas em ordem de posicionamento (do topo para a base).

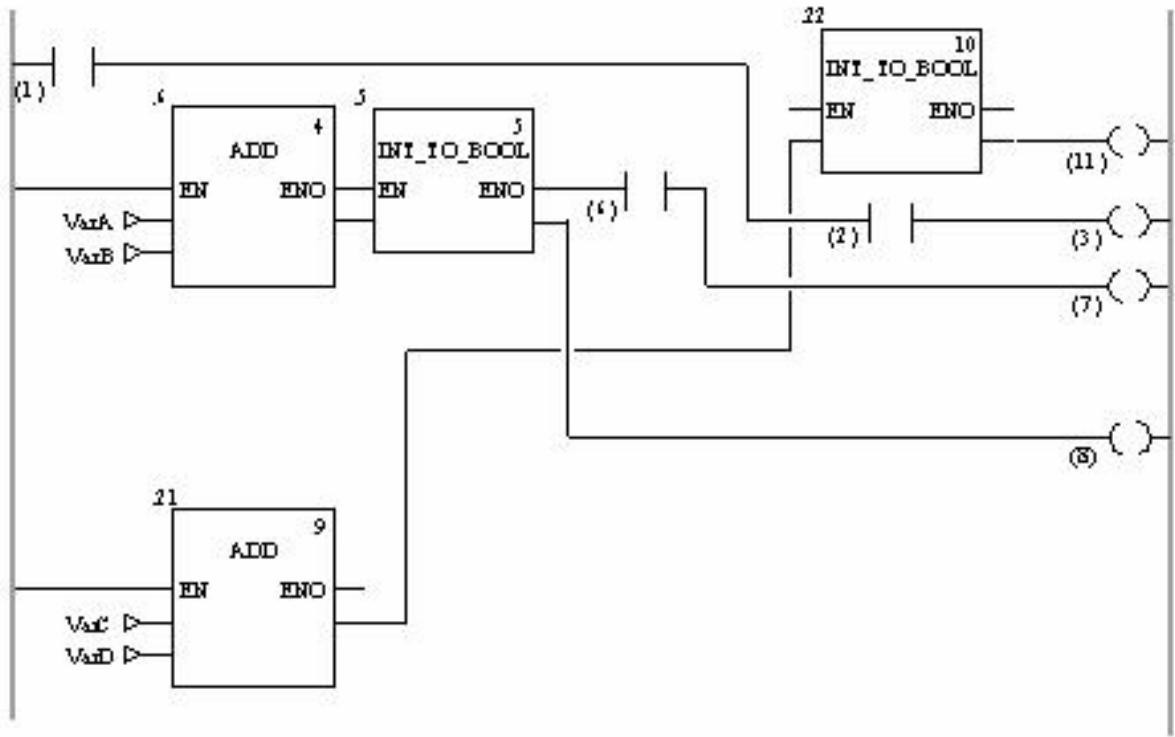
## **Regras:**

A execução de uma seção é baseada em networks com objetos conectados acima e abaixo. A seqüência de execução de networks que são conectadas somente pelo trilho de alimentação esquerda, é determinada pela seqüência gráfica (do topo para a base) no quais estes são conectados com o trilho esquerdo.

Isto não se aplica se a seqüência é influenciada por elementos de controle. Processamento de uma network é terminado completamente antes do processamento iniciar uma outra network.

O processamento de uma network é somente terminada se todas as saídas nesta network tenham sido processadas. Isto também se aplica se a network contém um ou mais elementos de controle.

A seqüência de processamento do diagrama ladder seguinte é indicada entre parêntese e no canto superior direito dos blocos FFBs.



### Fluxo de sinal em uma network (Rung)

#### Regras:

O fluxo de sinal para conexões booleanas é:

- da esquerda para a direita com conexões booleanas horizontais e
- do topo para a base com conexões booleanas vertical.

O fluxo de sinal com conexões FFBs é da saída do FFB para a entrada, não importando a direção que são feitos.

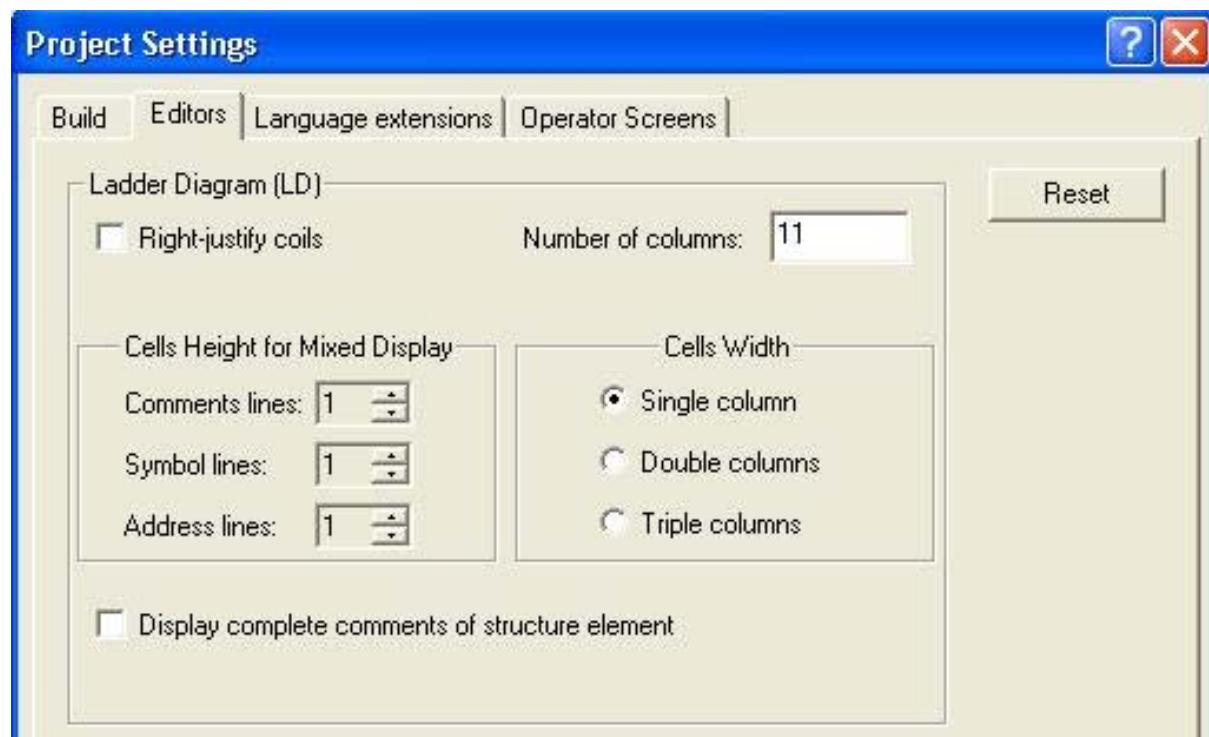
Um FFB é somente processado se todos os elementos (saídas FFBs, etc.) aos quais estão conectados são processados.

A seqüência de execução de FFBs que são conectados com várias saídas de mesmo FFB funciona do topo para a base.

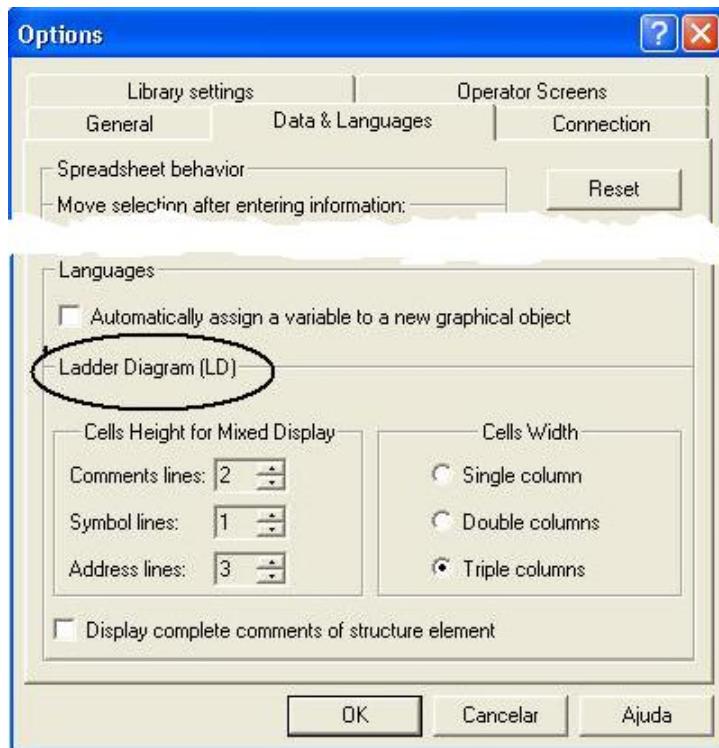
A seqüência de execução de objetos não é influenciada pela localização do mesmo na network.

### Configuração do editor Ladder

A configuração do editor em ladder é acessível através da barra de ferramenta no menu “Tools” selecionar “project settings” e na aba “Editors” configurar a posição da bobina marcando a caixa “Right-justify coils”, “Numbers of Columns”, número de células para visualização mista em “Cells Height for mixed Display”, largura das células em “Cells Width” e modo de visualização de comentário em “Display complete comments of structure element”.

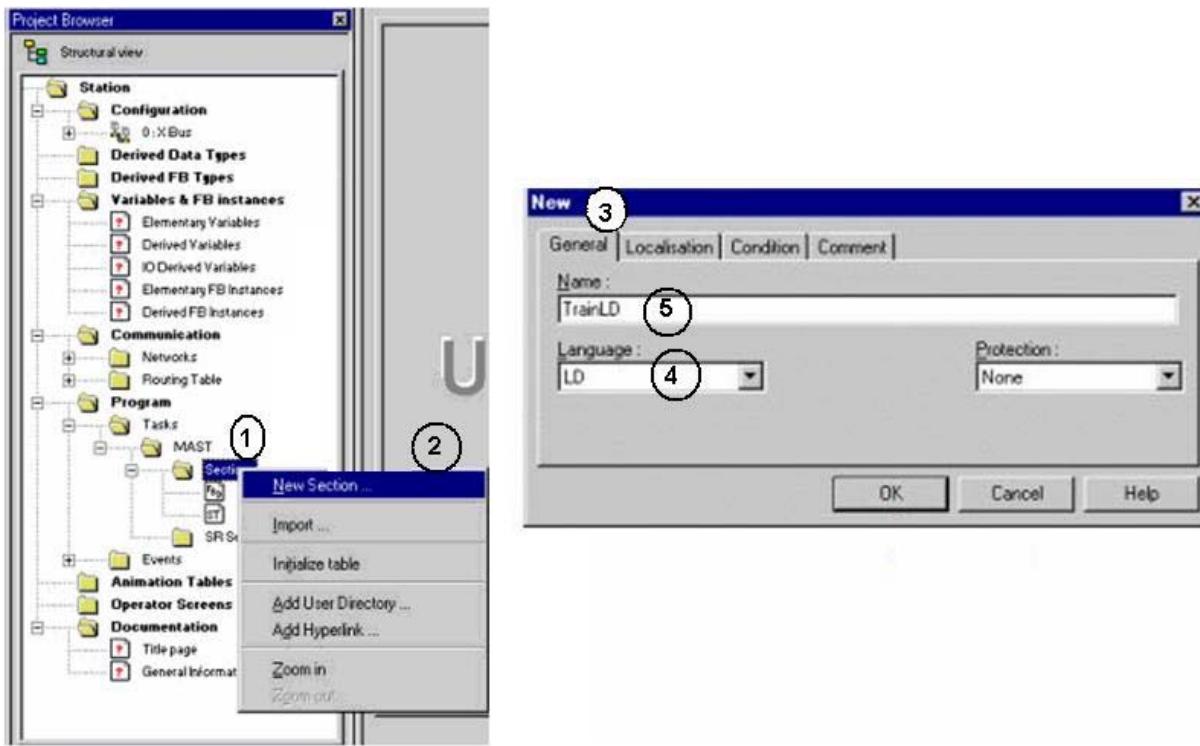


Os mesmos ajustes **para futuros projetos** são feitos através da barra de ferramenta no menu “Tools” selecionar “options” e na aba “Data & languages” fazer os ajustes em “Ladder Diagram [LD]”.

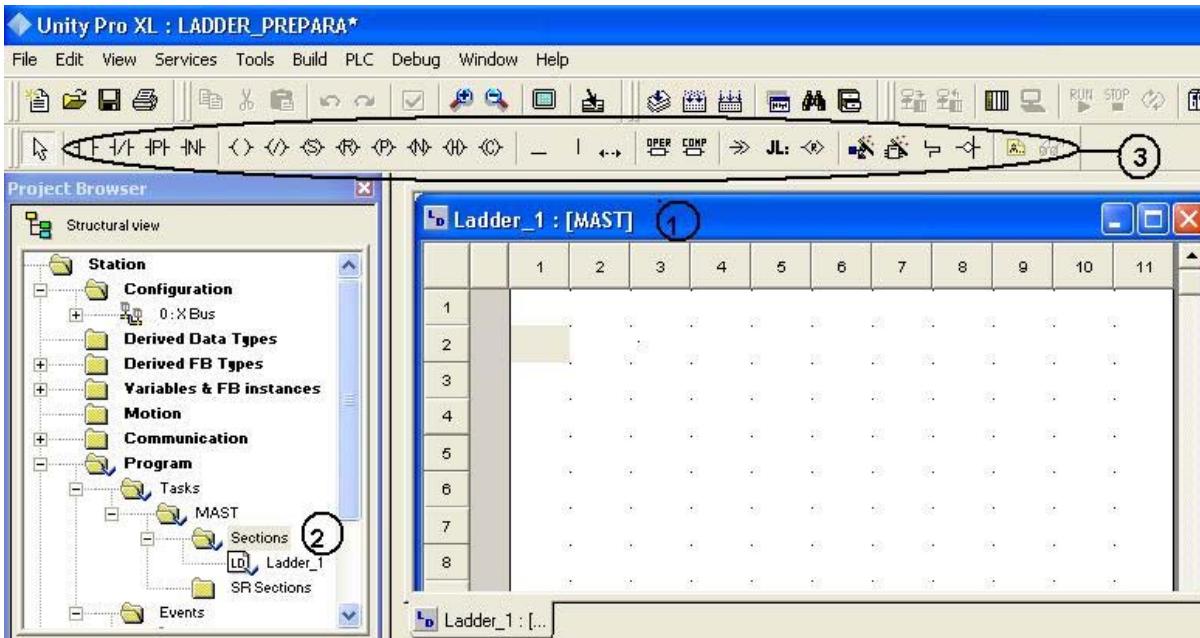


### *Criação de uma seção de Diagrama Ladder*

Uma seção Ladder é criada a partir do “project browser”, com a pasta “Sections” (1) selecionada, clicar com botão direito sobre a mesma e selecionar “New Section” (2), será aberta a caixa de diálogo “New” (3) onde deve ser selecionada a linguagem Ladder “LD”(4) e especificado um nome (5) sem espaço e obedecendo as regras de nomes da IEC, finalizando clicar no botão “OK”.

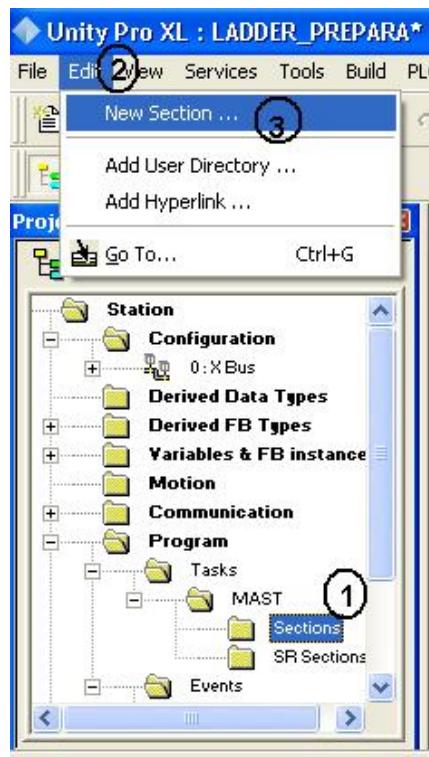


Ao Clicar em “OK” é aberta a área de edição do diagrama Ladder (1) com 11 colunas e 100 linhas, no “project browser” é mostrada a pasta da nova seção (2) e na barra de ferramenta são disponibilizados os objetos para edição ladder (3).



O número de linhas é ajustado diretamente no editor ladder, para inserir uma linha deve-se selecionar o número da linha (lado esquerdo do editor) onde se deseja fazer a inserção, e com o botão direito selecionar “insert rows”, para apagar uma linha o processo é o mesmo, basta selecionar “Delete”.

Para inserir mais de uma linha, deve-se marcar um número de uma linha, e com a tecla “Shift” acionada clicar no número de linhas que se deseja inserir, com isto serão marcadas todas as linhas entre os dois números. Com o botão direito selecionar “Insert rows”, acrescentando assim o número de linhas desejadas, utilizando o mesmo procedimento é possível apagar várias linhas simultaneamente.



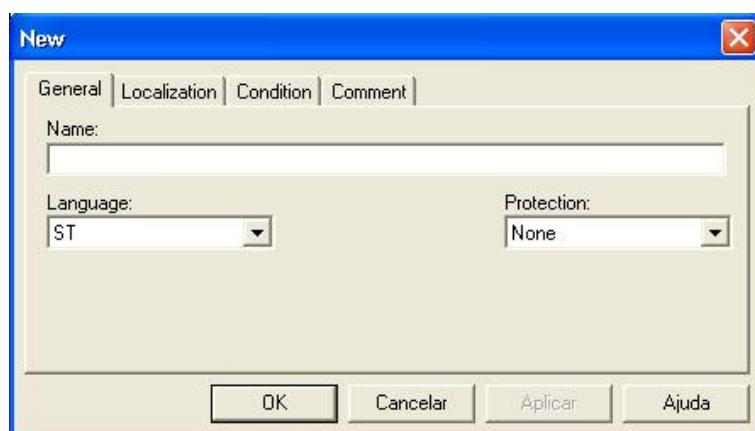
Uma seção também pode ser criada a partir da barra de ferramenta; com a pasta “Sections” (1) selecionada, acessar o menu “Edit” (2) e clicar com botão esquerdo em “New section” (3).

- **Configuração da seção**

A caixa de diálogo “New” que é visualizada ao criar uma seção independente da linguagem a ser utilizada possui além da aba “General” comentada anteriormente, as abas “Location”, “Condition” e “Comments”.

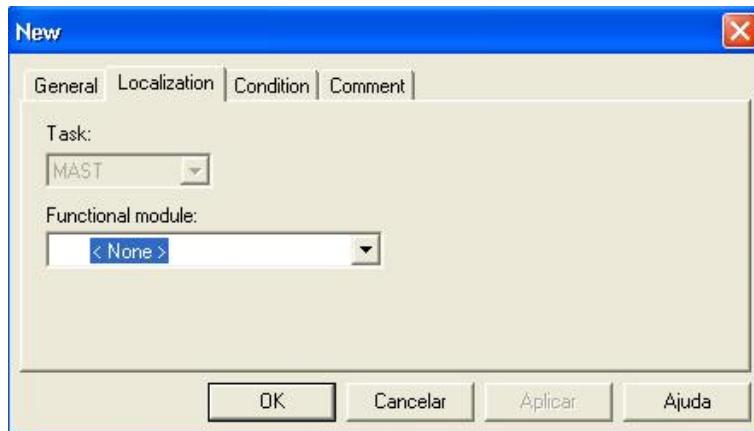
#### **Aba General:**

Na aba geral deve-se especificar o nome da seção, a linguagem que será utilizada e o tipo de proteção, que pode ser sem proteção “None”, somente leitura “Read only” ou proteção de leitura e escrita “No read & Write”.



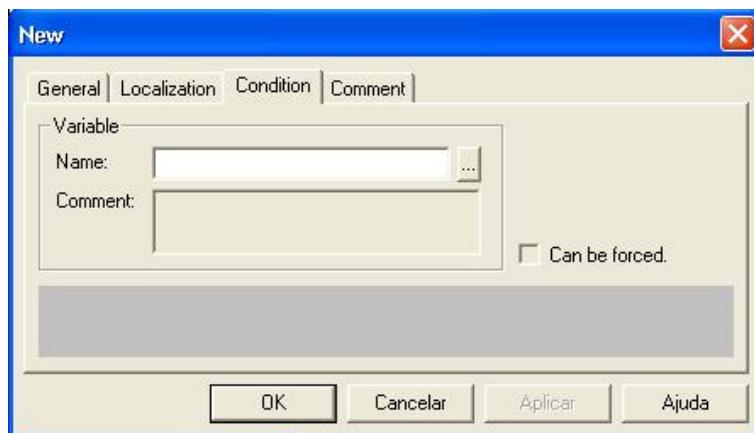
#### **Aba Location.**

Na localização deve ser especificado o módulo funcional ao qual a seção pertence. O tipo de tarefa foi selecionado anteriormente.



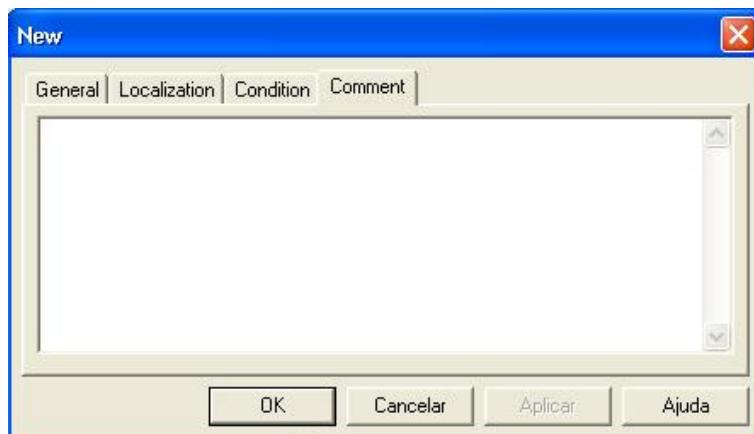
### Aba Condition.

Na aba condition é especificada uma variável que habilita ou não a execução da seção.



### Aba Comments.

Na aba comentário são digitadas informações gerais da seção



### Edição do Ladder

Ao criar a seção é mostrada a área de edição ladder e na barra de ferramentas, são visualizados os objetos utilizados nesta linguagem.



Descrição dos objetos Ladder:

	Contato normalmente aberto
	Contato normalmente fechado
	Contato detector de borda positiva/subida
	Detector de borda negativa/descida
	Bobina (ativa em um)
	Bobina invertida (ativa em zero)
	Bobina de "Set"
	Bobina de "Reset"
	Bobina ativa durante um "Scan" na borda de subida (Ligaçāo)
	Bobina ativa durante um "Scan" na borda de descida (Desligamento)
	Bobina de Halt/espera
	Bobina de chamada de Sub-rotina
	Conexão horizontal de uma célula
	Conexão vertical
	Conexão horizontal entre objetos não vizinhos
	Bloco "Operate"
	Bloco comparador
	Chamada de salto
	Rótulo/Label de um salto
	Retorno de sub-rotina
	Seleção de FFBs
	Assistente de entradas FFBs
	Conexão em desnível
	Inversor de pino
	Objeto de texto/comentário

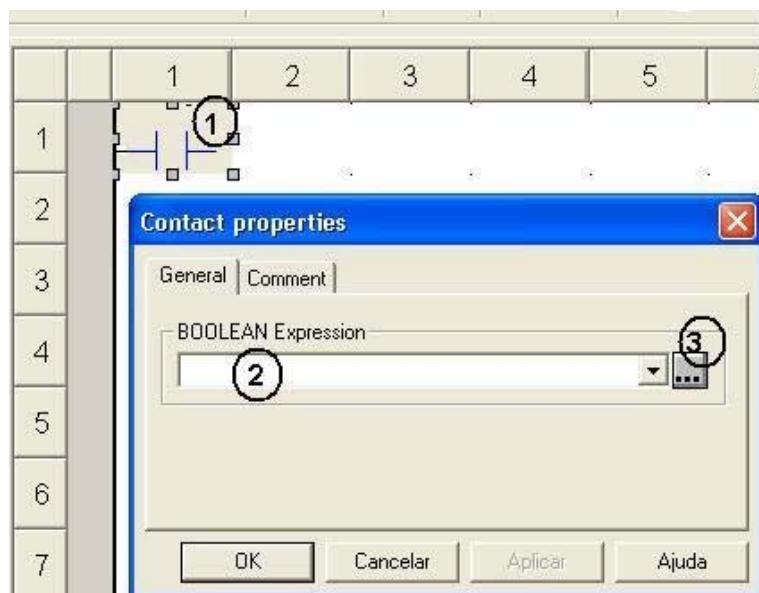
- **Edição de Diagrama Ladder com Contatos e Bobinas**

A edição do diagrama Ladder pode ser feita de duas maneiras, através da barra de ferramenta no menu “Edit”, selecionado “New”, serão mostrados os objetos disponíveis para que seja selecionado e colado na área de edição.

	Normally open contact	F3
	Normally closed contact	Shift+F3
	Positive transition-sensing contact	Ctrl+F3
	Negative transition-sensing contact	Ctrl+Shift+F3
	Coil	F5
	Negated coil	Shift+F5
	Set coil	Alt+F5
	Reset coil	Shift+Alt+F5
	Positive transition-sensing coil	
	Negative transition-sensing coil	
	Halt coil	
	Call coil	F4
	Boolean Connection	F7
	Vertical Connection	Shift+F7
	Boolean Link	Alt+F6
	Link	F6
	Operate Block	Alt+F7
	Compare Block	Ctrl+F7
	Jump	
	Jump Label	
	Return	
	Pin negation	
	Comment	F8
	Inspect Window	F9

Após colocação do objeto, acionar a tecla “**Esc**” ou o ícone “Select mode”, com isto o “prompt” do mouse será liberado.

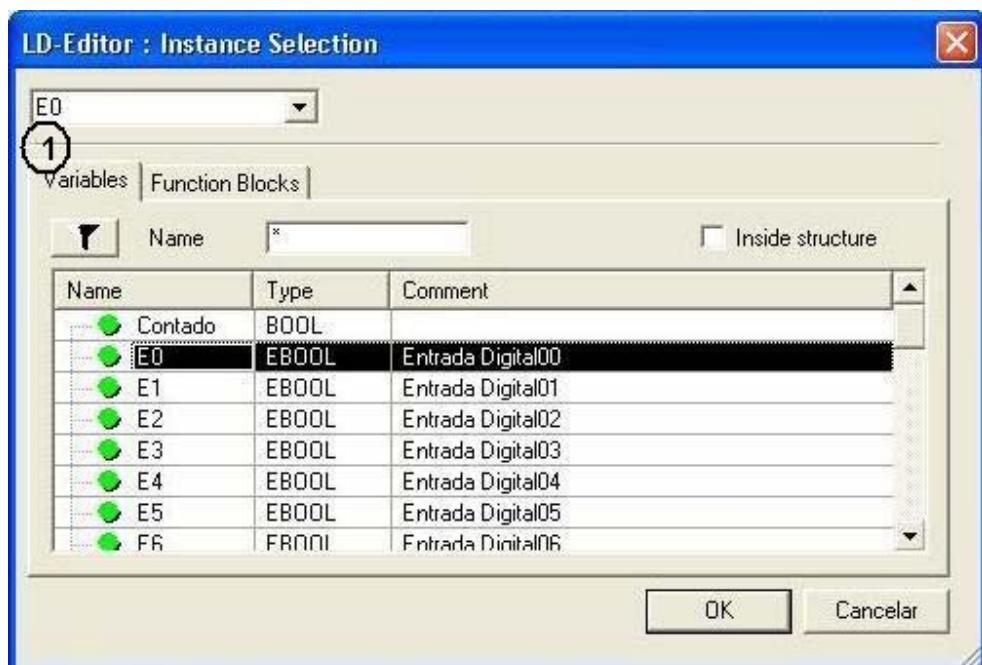
Com dois cliques sobre o objeto (1), será aberta a caixa de diálogo para especificação do mesmo, o qual poderia ter sido declarado ou não anteriormente como visto no “**Cap. 9 Variáveis**”, no campo “**BOOLEAN Expression**” deve ser especificada a variável que será associada ao objeto (2) e acionar “OK”.



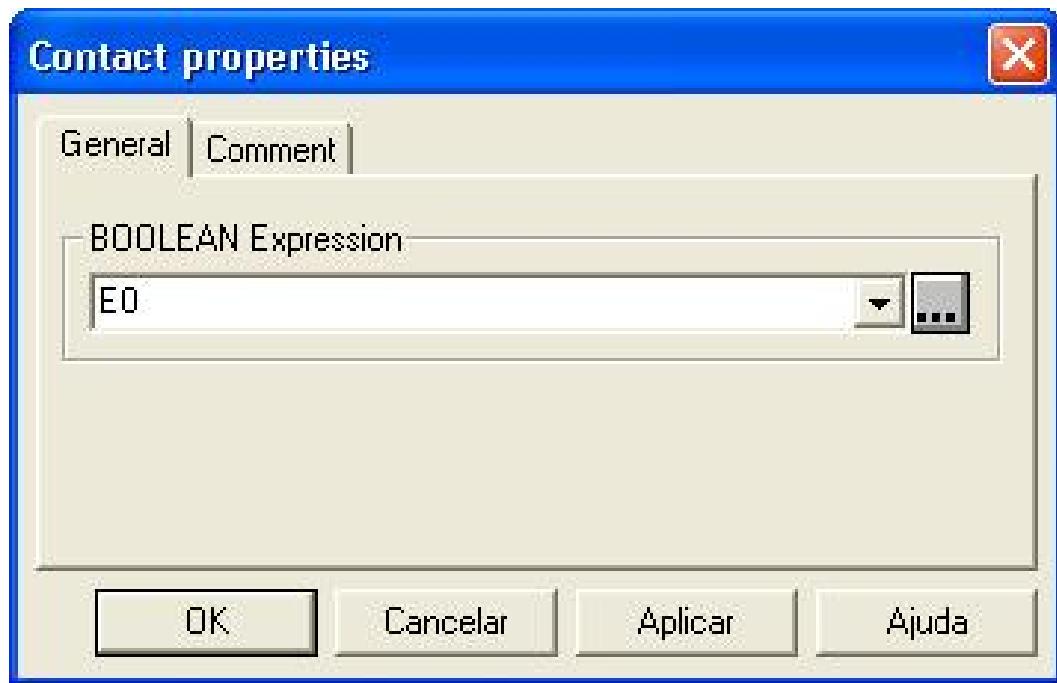
Ao acionar o botão **OK** com uma variável não declarada no campo “**BOOLEAN Expression**”, a variável é indicada no objeto sublinhado em vermelho (1) e será mostrada juntamente a caixa de diálogo para criação da variável (2) como visto no “**Cap. 9 - Variáveis**”.



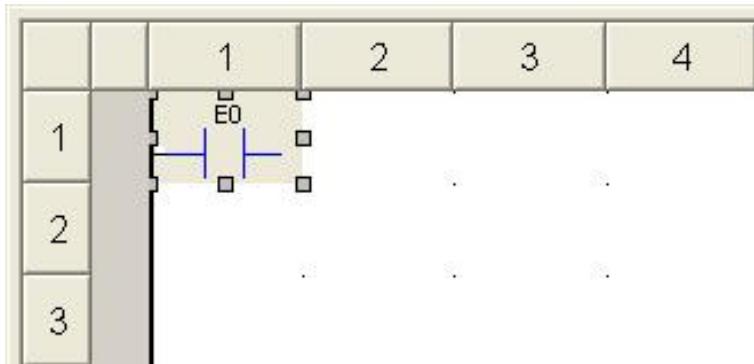
A variável também pode ser associada ao objeto através do acionamento do botão [...] (3), que ao ser acionado é aberta a caixa de diálogo “**LD – Editor: Instance selection**”, selecionando a aba “**Variables**” (1) serão listadas as variáveis declaradas anteriormente, selecionar a desejada e acionar **OK**,



Ao acionar o botão **OK** será mostrada novamente a caixa de especificação do objeto, agora com a variável selecionada, clicar em **OK**, finalizando a edição do referido objeto.

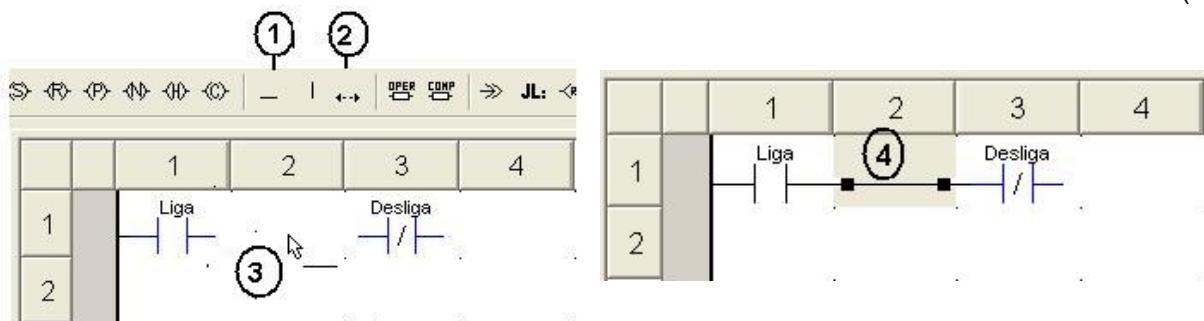


Objeto totalmente editado



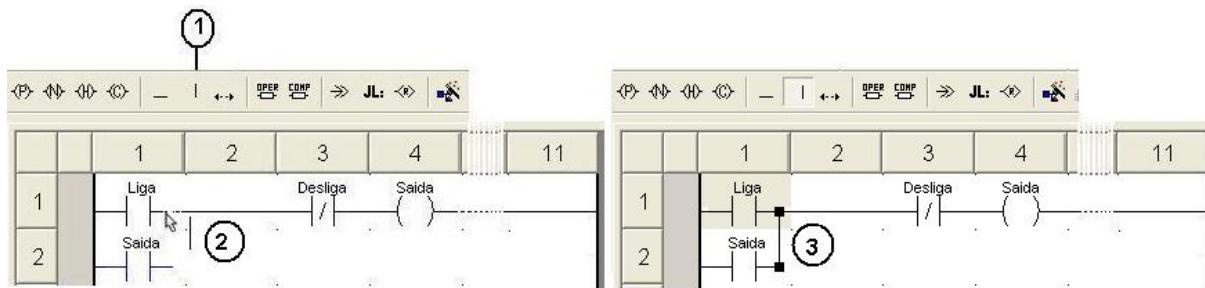
#### Conexão horizontal entre objetos.

A conexão entre os objetos “Liga” e “Desliga” é executada selecionando os ícones (1) ou (2) e posicionar o “prompt” (3) entre os objetos e açãoando o botão esquerdo, a conexão será concluída (4).

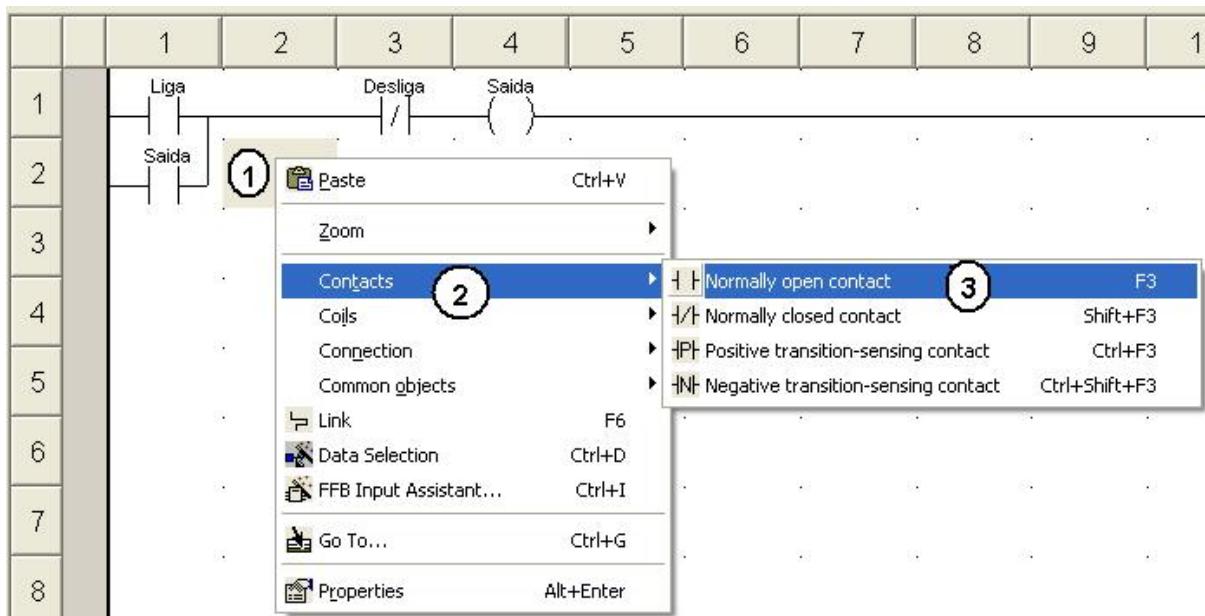


#### Conexão vertical entre objetos.

A conexão vertical entre os objetos “Liga” e “Saída” é executada selecionando o ícone (1) e posicionar o “prompt” no elemento que dará origem a conexão (2) e açãoando o botão esquerdo, concluindo assim a conexão (3).



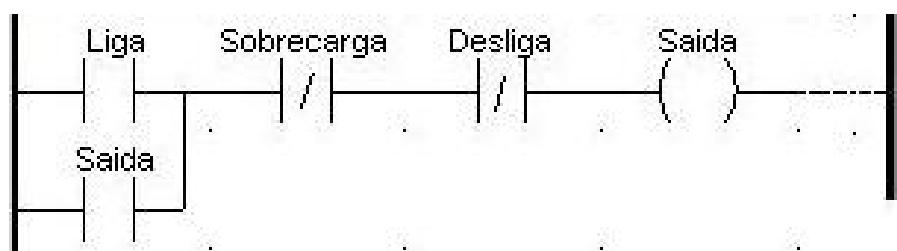
Pode-se também editar o ladder clicando com o botão direito na área de edição (1) e selecionar a família de objetos (2), selecionando o objeto desejado (3).



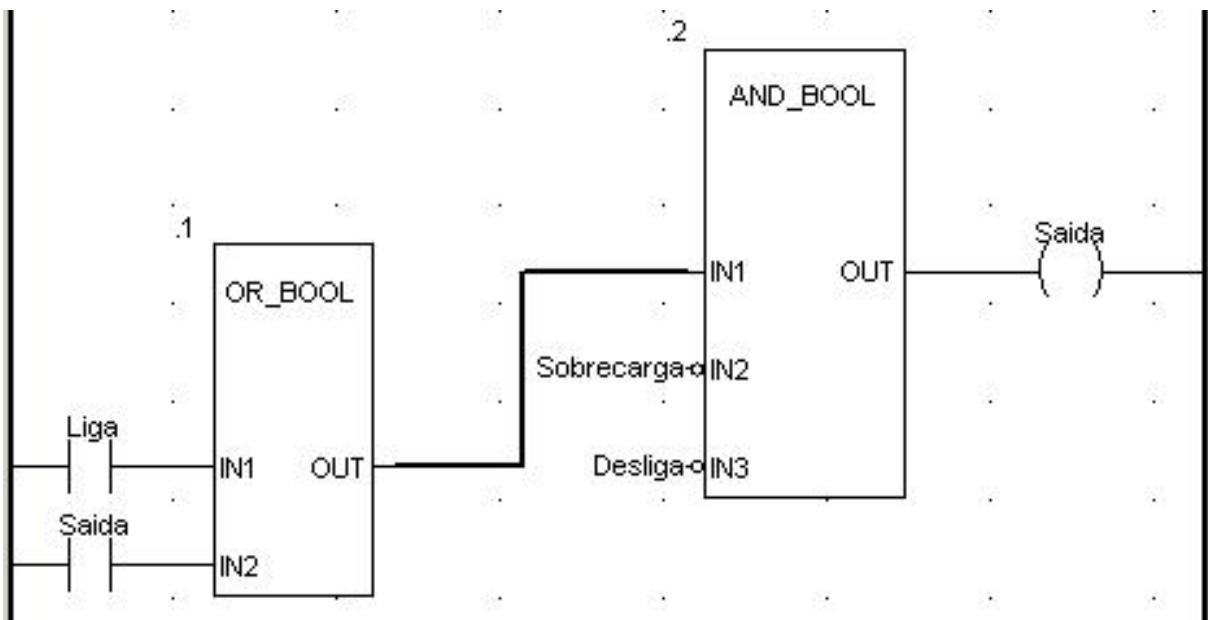
A conexão vertical serve como uma lógica **OU**. Desta forma é permitido conectar até 32 entradas (Contatos) e 64 saídas (Bobinas, links)

Um diagrama ladder pode ser construído exclusivamente com contatos e bobinas, como visto até então, o Unity Pro possui o recurso de FFBs que é um termo coletivo quando se refere a EF (elementary function), EFB (elementary function block) e DFB (derived function block).

#### **Exemplo de diagrama ladder exclusivamente com contatos e bobina.**

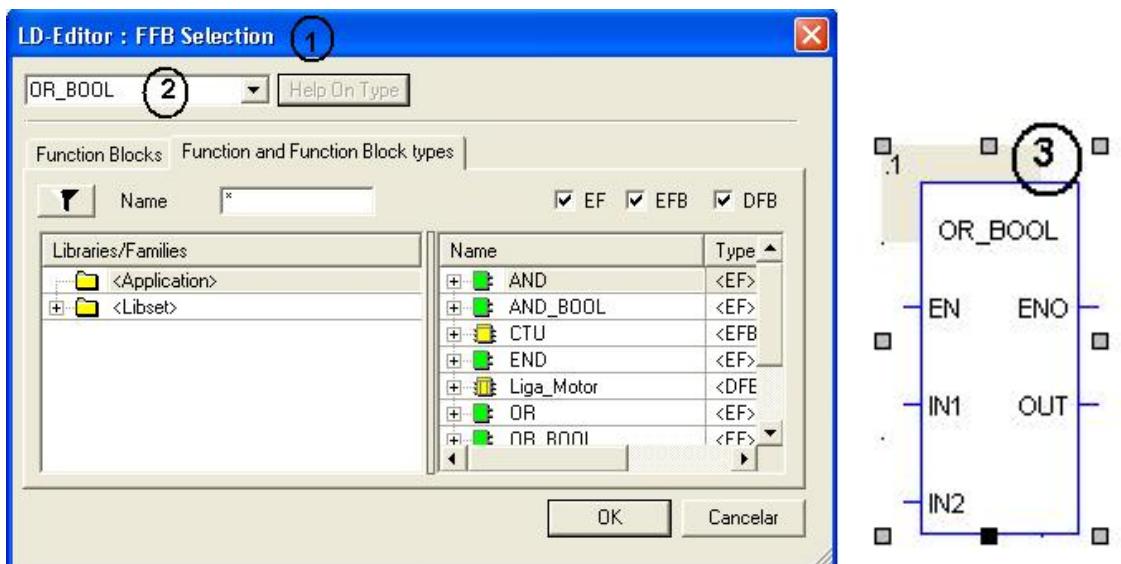


**Mesmo diagrama ladder do exemplo anterior, agora com blocos de função FFBs.**



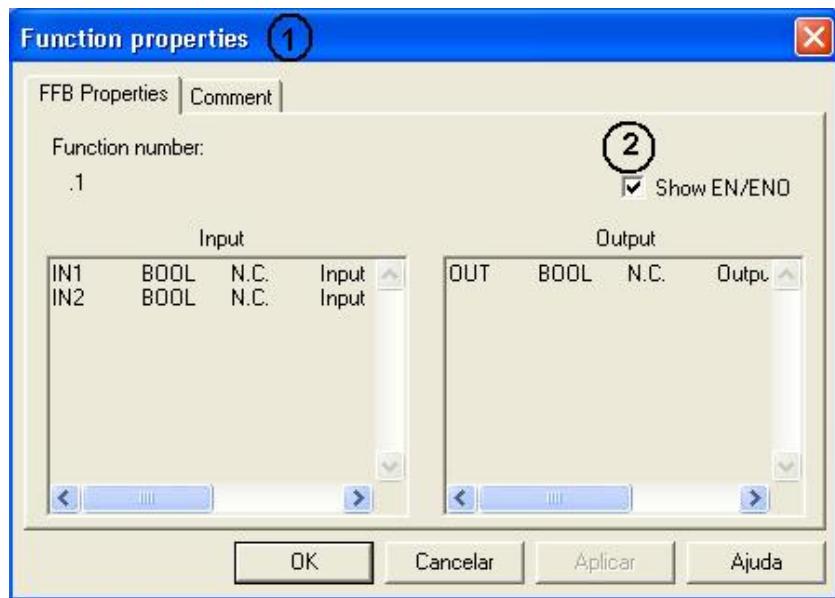
- Edição de Diagrama Ladder com FFBs

O acesso aos blocos FFBs é possível a partir da barra de ferramentas clicando no ícone “Data Selection”, que ao ser acionado é aberta a caixa de diálogo “LD-Editor: FFB Selection” (1) na qual deve ser digitado o bloco desejado no campo (2) e clicar no botão OK, com isto, posicionar o prompt do mouse na posição desejada na área de edição e clicar no botão esquerdo do mouse, colocando o bloco na área de edição (3).

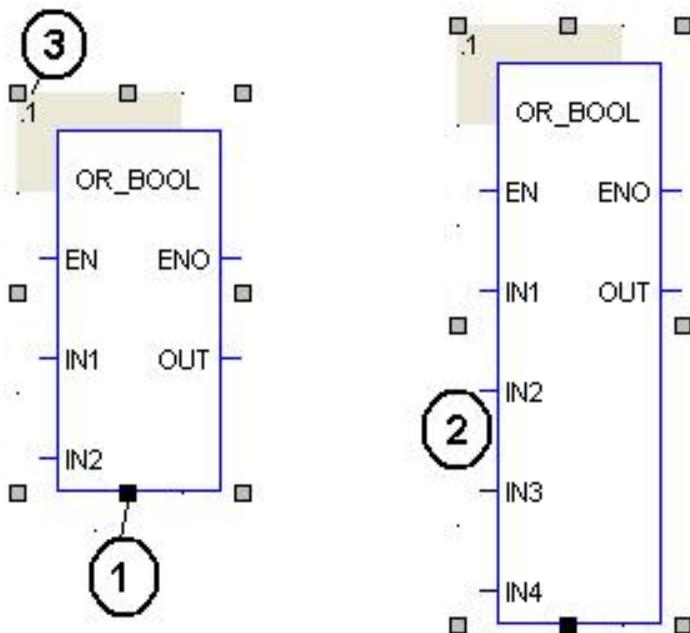


O bloco FFB apresenta três entradas; EN (Habilitação do bloco), IN1 (entrada 1), IN2 \*(entrada2) e duas saídas ENO (indicação de que o bloco está habilitado e OUT (resultado da lógica do bloco).

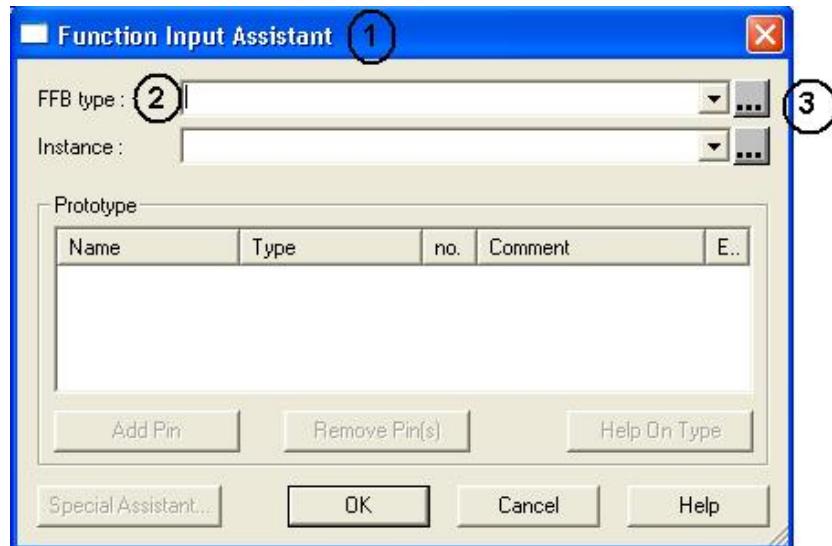
Os pinos de habilitação EN e ENO podem ser suprimidos, para isto deve-se dar dois cliques sobre o bloco que será aberto caixa de diálogo “Function Properties” (1) onde na aba “FFB Properties”, deve ser desmarcado o campo (2) “Show EN/ENO”, na aba “Comment” pode ser digitado comentário sobre o bloco e para finalizar clicar no botão **OK**.



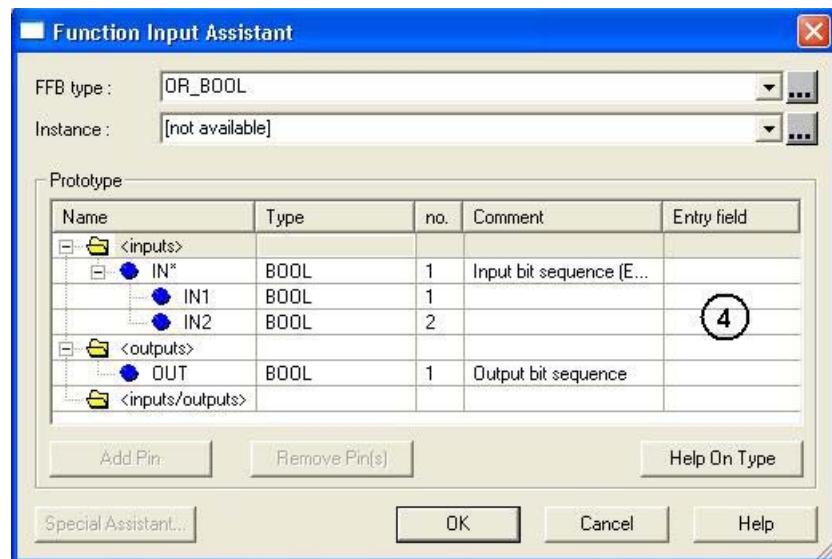
Blocos FFBs de função lógica básica, quando são inseridos, apresentam duas entradas, que podem ser aumentadas, para tal, deve-se clicar sobre o ponto (1) e arrastar para baixo até completar o número desejado (2) sendo no máximo é 32. Na seqüência em que os FFBs são colocados no programa, é gerado automaticamente seu número seqüencial, indicado no canto superior esquerdo (3).



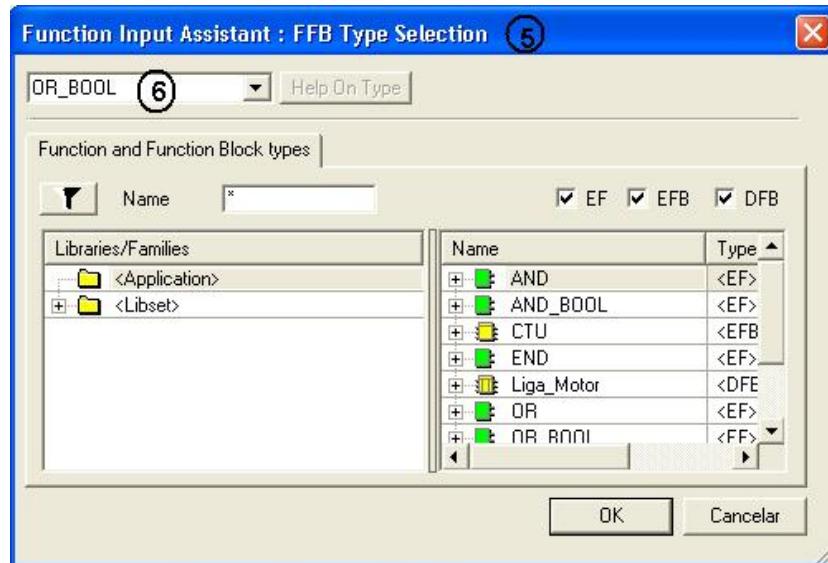
O acesso aos FFBs pode também ser feito através do ícone “FFP Input assistant” na barra de ferramentas, que quando acionado é aberta a caixa de diálogo “ Function Input assistant” (1), nesta dever especificado o tipo do bloco desejado em “FFB Type” (2) ou clicar no botão [...] (3).



Ao clicar em “FFB Type” (2), o mesmo é localizado na biblioteca e apresentado na mesma, onde pode-se entrar com o nome das variáveis nos diversos pinos do mesmo (4).

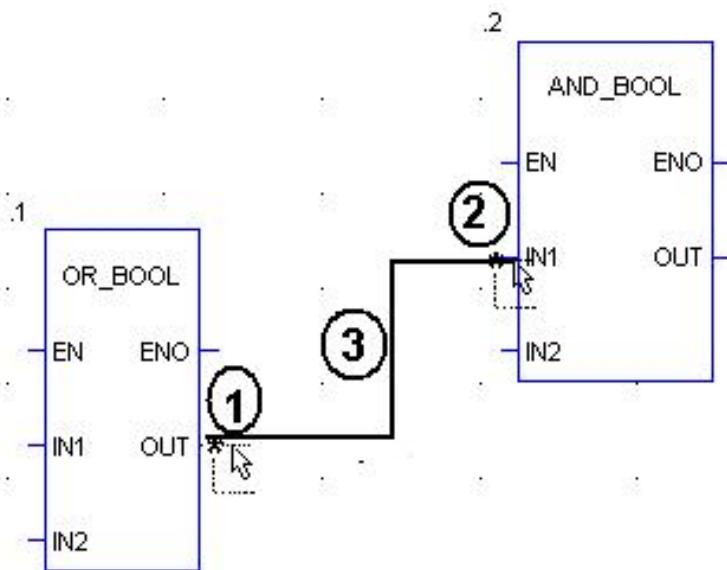


Ao clicar no botão [...] (3), é aberta a caixa de diálogo “Function input assistant: FFB Assistant” (5) onde deve ser digitado o tipo do bloco desejado (6) e clicar em OK, com isto a caixa de diálogo “Function input assistant” é completada com as informações do mesmo como explicado anteriormente.



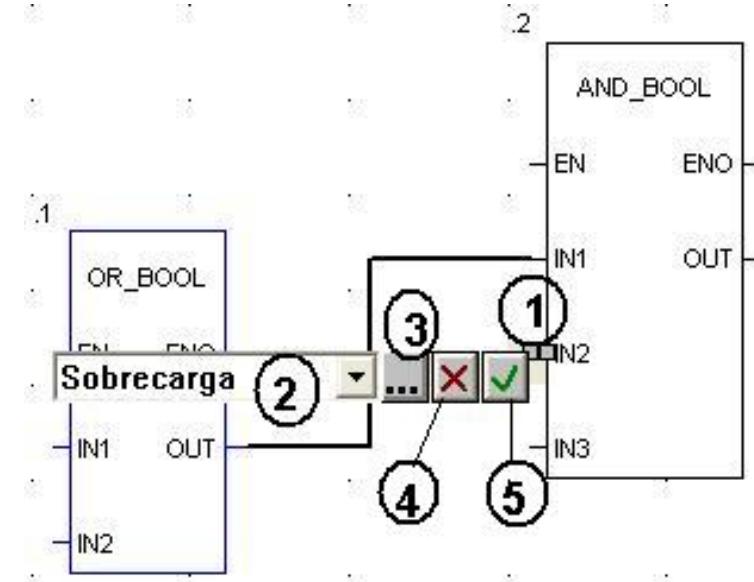
### Conexão entre FFBs.

A conexão é feita clicando no ícone “Link” e ao deslocar o prompt do mouse sobre a área de edição o mesmo mostra , indicando que a conexão naquele é possível ou , indicando conexão não permitida. Quando o “prompt” indica conexão possível deve-se clicar com botão esquerdo (1), iniciando assim a conexão e posicionar o mouse no término da conexão (2) e clicar com o botão esquerdo, finalizando assim a conexão (3).

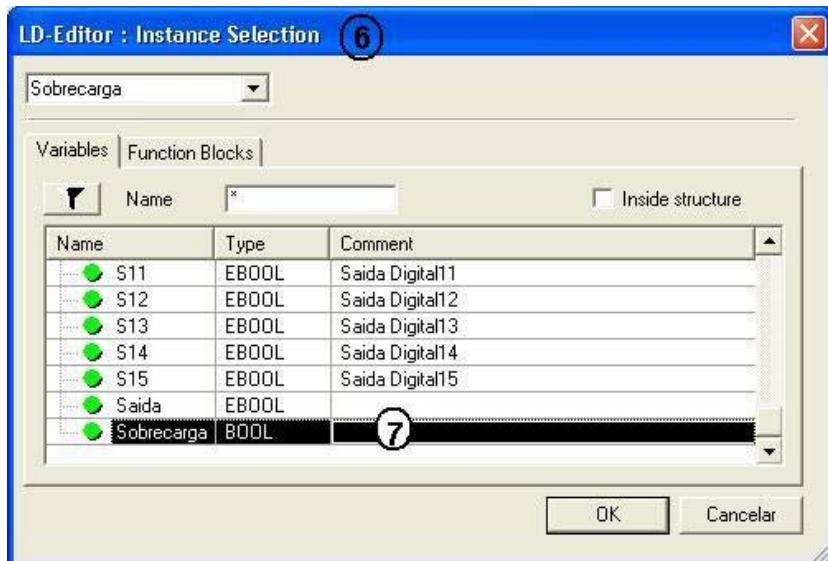


### Especificação de variáveis no bloco FFB.

Dar dois Cliques com botão esquerdo na entrada a ser colocada a variável (1), com isto será aberto a área de especificação da variável (2), ou clicar no botão [...] (3).



Ao clicar no botão [...] (3), será aberta a caixa de diálogo “LD Editor : Instance Selection” (6), que selecionando a aba “variables” é possível selecionar a variável desejada (7), para finalizar clicar no botão **OK**, o bloco será atualizado com a variável especificada na entrada.



Ao clicar no botão (4), abandona-se a especificação da variável.

Ao clicar no botão (5), efetiva-se a especificação que no caso de variável nova, será mostrada a caixa de diálogo “Create variable?”, neste caso, realizar o procedimento de criação de variáveis visto no “**Cap.9 Variáveis**”.

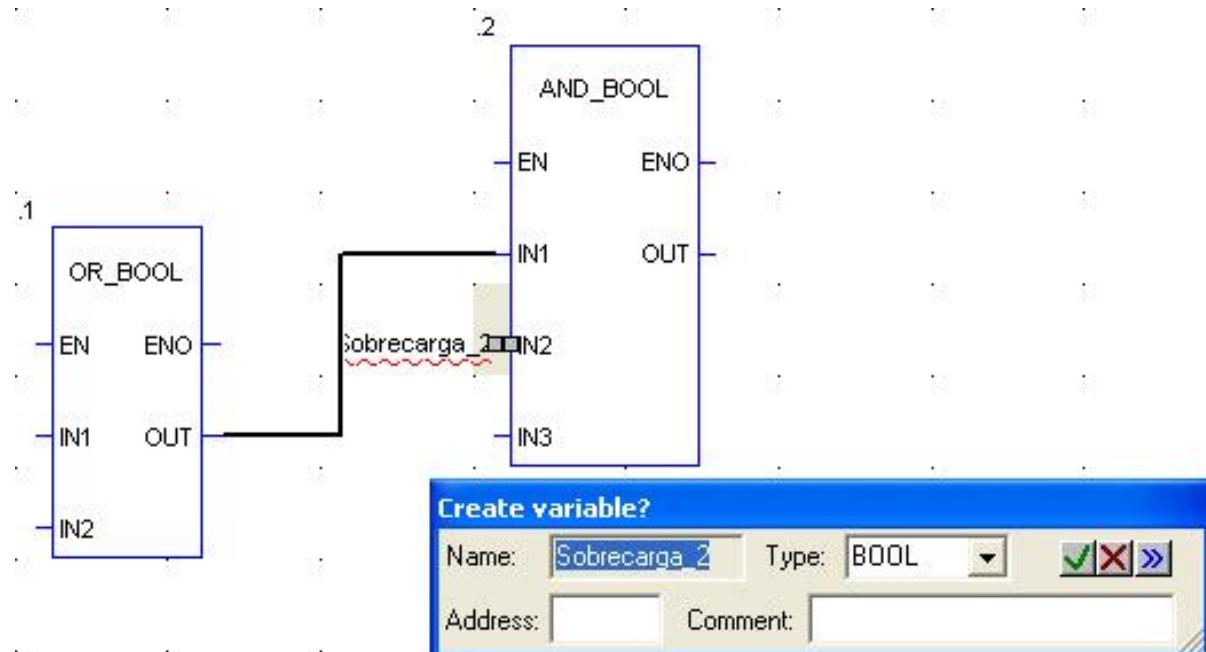
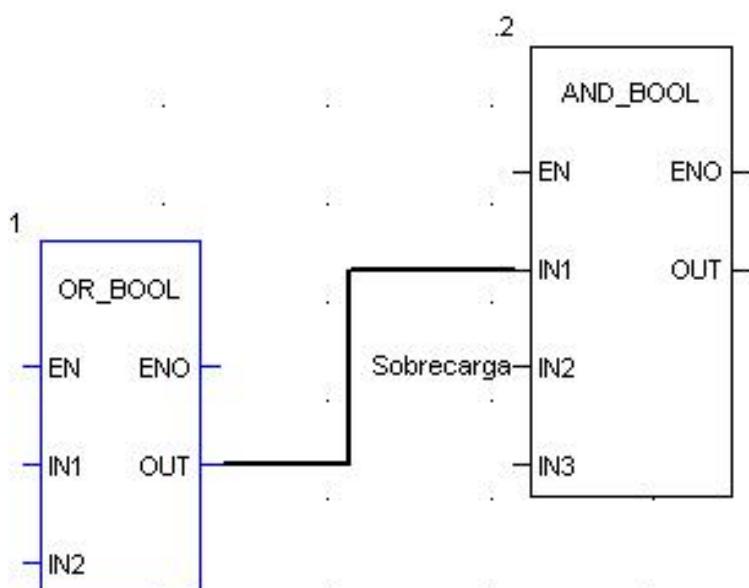


Diagrama com a especificação da variável concluída.



#### Inversão da função lógica de uma variável ( contato normalmente fechado).

Após a especificação da variável, selecionar na barra de ferramenta o ícone “Toggle Pin-negation”, com isto o “prompt” do mouse mostrará , o qual deve ser posicionado no pino da entrada que se deseja inverter (1) e acionar o botão esquerdo do mouse, fazendo assim a inversão.

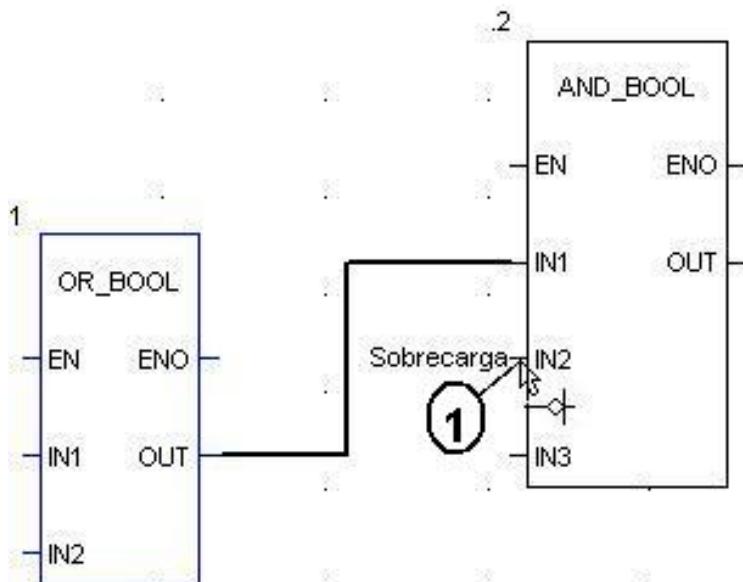
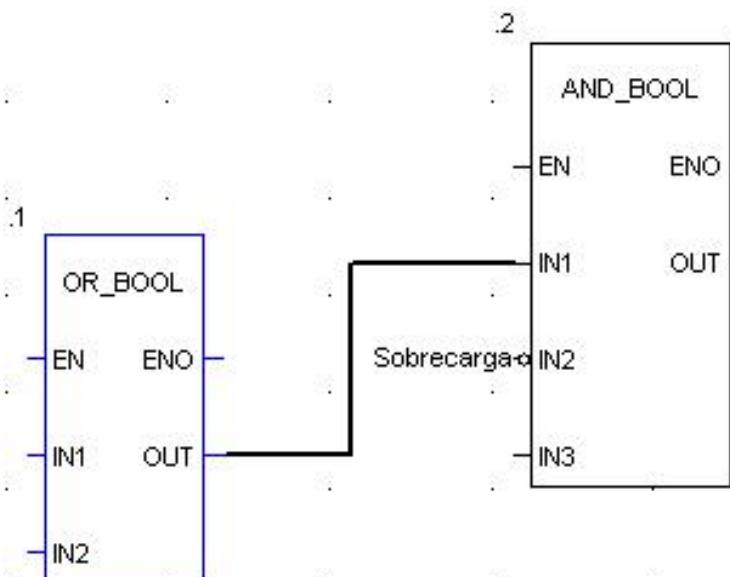
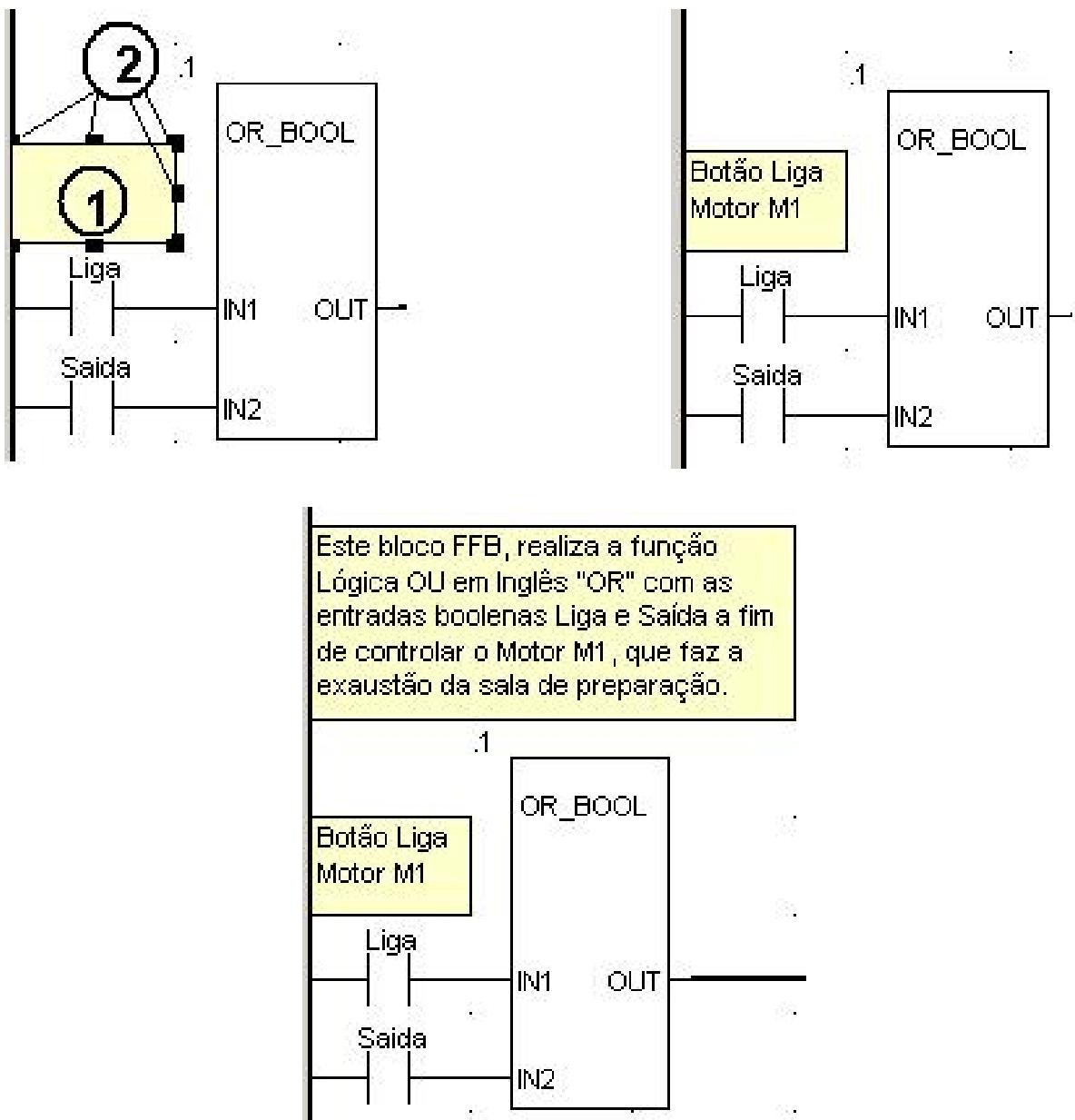


Diagrama com a inversão da função lógica da variável concluída.



#### Colocação de comentário (Texto) no diagrama Ladder.

A colocação de comentários no diagrama Ladder é feita através do ícone “Comment”, que ao ser clicado o “prompt” do mouse mostra . Ao clicar na área de edição do Ladder, é aberta a área de digitação (1) do comentário, a qual pode ser selecionada e arrastada dentro da área de edição do diagrama Ladder, para edição de textos que ocupem mais de uma célula, é possível expandir a área de digitação clicando em arrastando nos pontos (2).



- **Elementos de Controle em Ladder.**

Elementos de controle são usados executar saltos em uma seção ladder e retornar de uma sub-rotina SRx ou blocos DFBs para retornar para o programa principal.

Os elementos de controle estão disponíveis na barra de ferramentas e são:

⇒ **Salto “Jump”**, quando o estado do elemento conectado a esquerda é “1”, ocorre o salto para um rótulo “Label” na mesma seção.

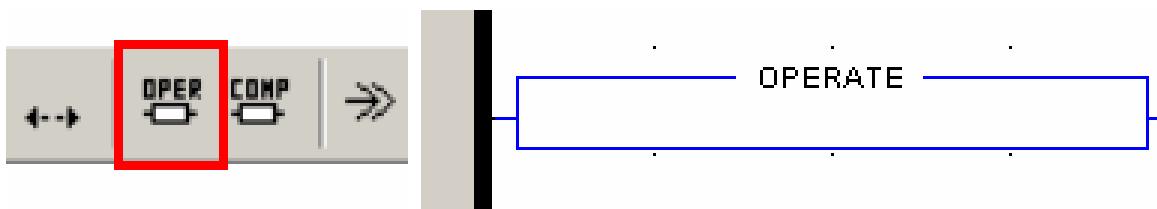
**JL:** **Rótulo do salto “Jump Label”**, O Label é indicado como um texto terminado por ponto e vírgula “;”, este texto é limitado a 32 caracteres e precisa ser único na seção. O texto precisa obedecer a convenção de nomeação. Labels podem ser colocados somente na primeira célula do trilho de alimentação.

⇒ **Retorno “Return”**, Toda sub-rotina e todo DFB é abandonada após a mesma ser processada, por exemplo, retorno para o programa principal que gerou a chamada

- **Bloco Operate.**

O Bloco Operate executa instruções e Expressões na linguagem ST (Texto Estruturado). Este bloco é somente disponível na linguagem Ladder.

O Bloco Operate é acessado a partir da barra de ferramenta no ícone identificado por “Operate block”.



Todas as instruções ST são permitidas, exceto as instruções de controle:

RETURN, IF, FOR  
JUMP, CASE, etc.)

Para o bloco operate, o estado da conexão esquerda é passada para a conexão direita (independente do resultado da instrução ST), ou seja, a ativação do elemento da esquerda, habilita a ativação do elemento da direita.

Bloco operate é colocado em qualquer célula livre e requer 1 linha e 4 colunas, a colocação de um bloco operate em uma célula ocupada, é retornada uma mensagem de erro.

Uma colocação do bloco operate, cria automaticamente conexões com objetos vizinhos na esquerda se estes são tipos de dados booleanos e não há células vagas entre os mesmos.

Um Bloco Operate pode conter até 4096 caracteres. Se todos os caracteres não puderem ser visualizados, o início da seqüência de caracteres será seguido por pontos (...).

### Ativação do Bloco Operate

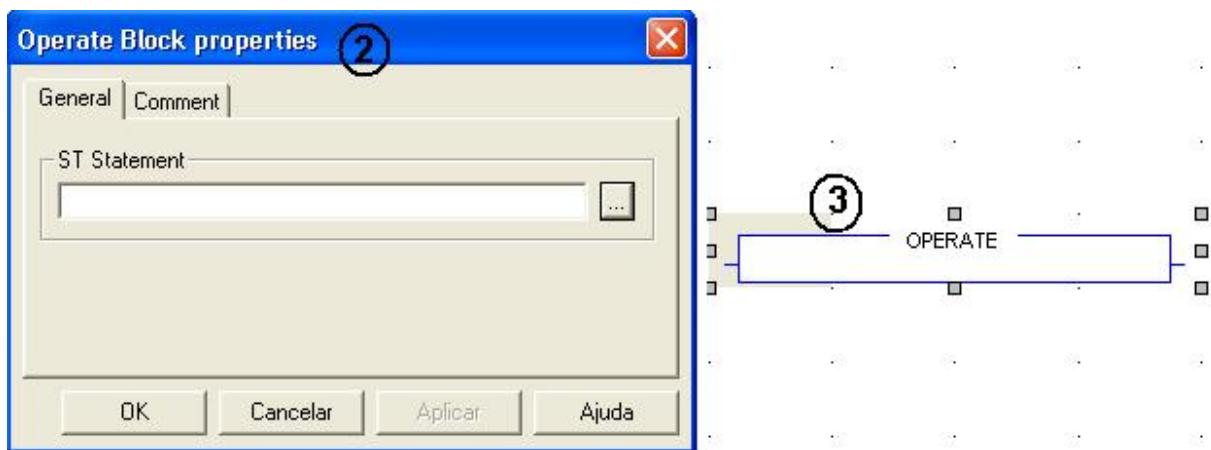
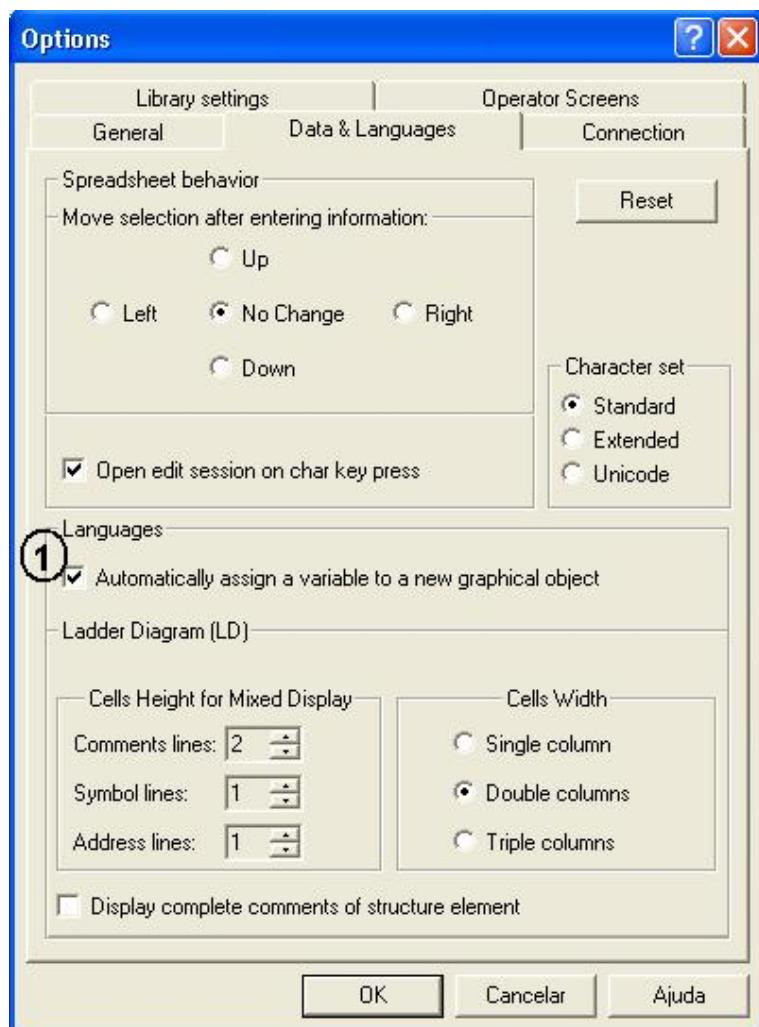
O Bloco Operate pode ser ativado das seguintes maneiras:

- Através dos comandos **Edit => New => Operate Block**;
- Através de clique no botão direito selecionar => **Common Objects => Operate Block**;
- Através da combinação de teclados **ALT + F7** ou
- Através do ícone “Operate block” na barra de ferramenta.

Para qualquer opção acima, o “prompt” do mouse mostra



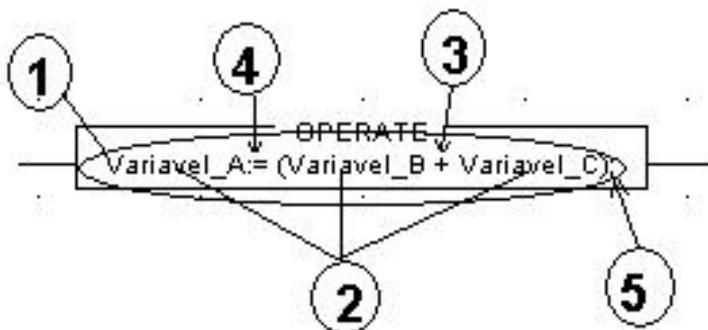
**NOTA** Na aba **Tools** na barra de ferramentas, na caixa “**Options**”, na aba “**Data and Languages**”, ao marcar a caixa “**Automatically assign a variable to a new graphical object**” (1), a caixa de diálogo para atribuição de variável para o elemento gráfico “**Operate block Properties**” (2) é aberta automaticamente quando se coloca o bloco operate na área de edição (3).



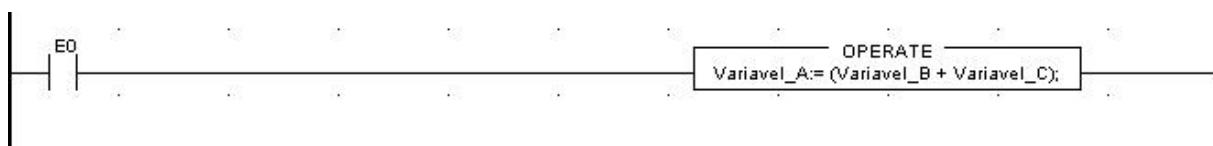
### Funcionamento do Bloco Operate:

Com o Bloco operate podemos realizar transferência de dados entre variáveis, instruções aritméticas e expressões.

Uma instrução ST (1) é composta por operandos (2) que são os elementos da instrução, operadores (3) que representa a operação a ser executada e fazendo parte da sintaxe da instrução “:=” (4). O ponto e vírgula “;” (5) no final da expressão é colocada automaticamente pelo sistema.



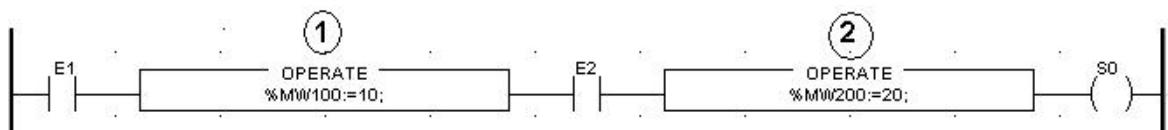
Quando a entrada E0 for verdadeira, o resultado da soma das variáveis "Variável\_B" e "Variável\_C" será armazenado na variável "Variável\_A"



Exemplo de mais de um operate na mesma linha:

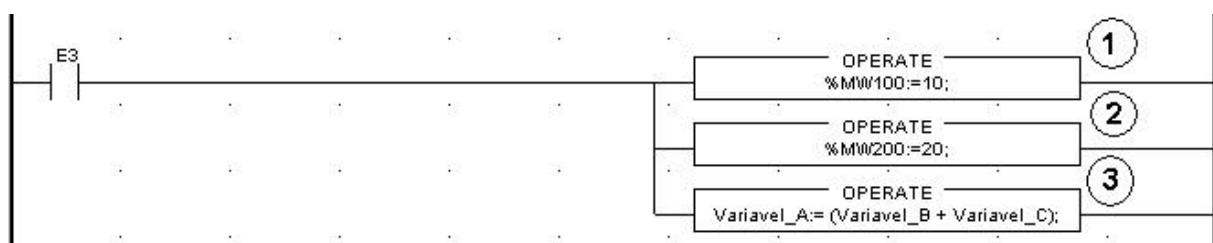
Quando a entrada E1 for verdadeira (ativa), o bloco operate (1) será executado, armazenando o valor 10 na posição de memória %MW100, não interferindo no operate (2) e na saída S0.

Quando as entradas E1 e E2 forem verdadeiras (ativas), os blocos operates (1) e (2) serão executados, escrevendo os valores 10 e 20 nas posições de memória %MW100 e %MW200 respectivamente e a saída S0 será verdadeira (ativada).



Exemplo de mais de um operate conectados verticalmente:

Quando a entrada E3 for verdadeira (ativa), os blocos (1), (2) e (3) serão executados.

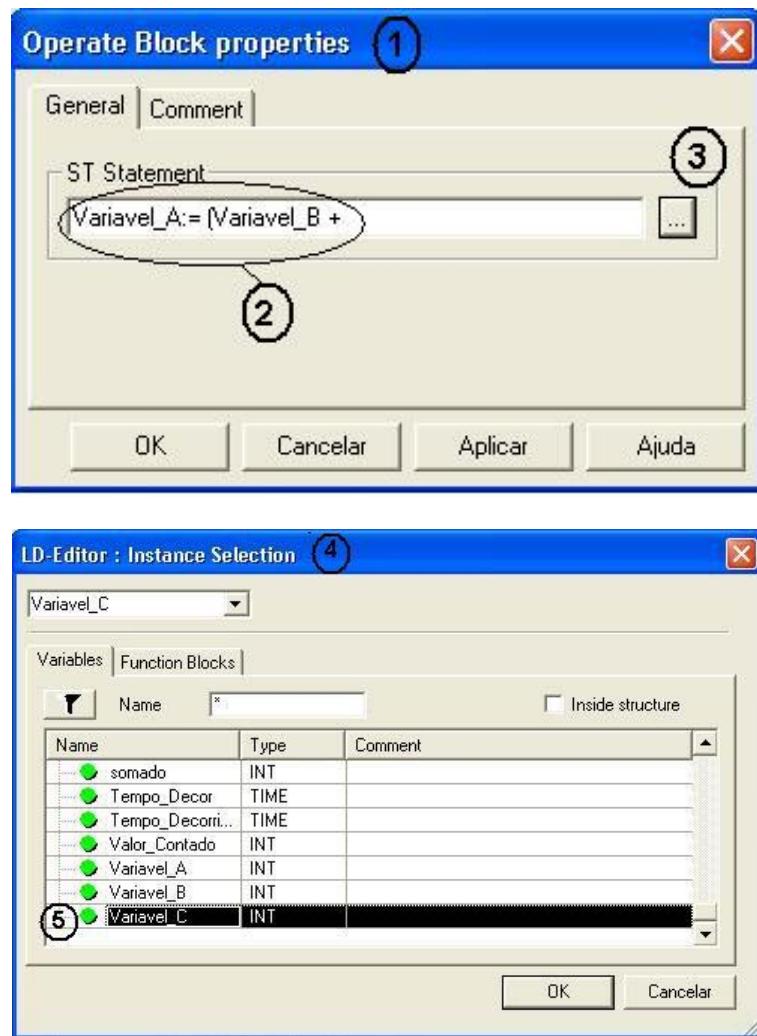


#### Colocação do Bloco Operate:

Ativar a colocação do bloco operate comentado anteriormente, posicionar no local desejado e clicar com botão esquerdo do mouse, fixando o bloco na célula, ou após a ativação da colocação do bloco operate, utilizar as teclas de navegação e posicionar o campo "cinza" na célula desejada e acionar a combinação de teclas **Ctrl + Enter**.

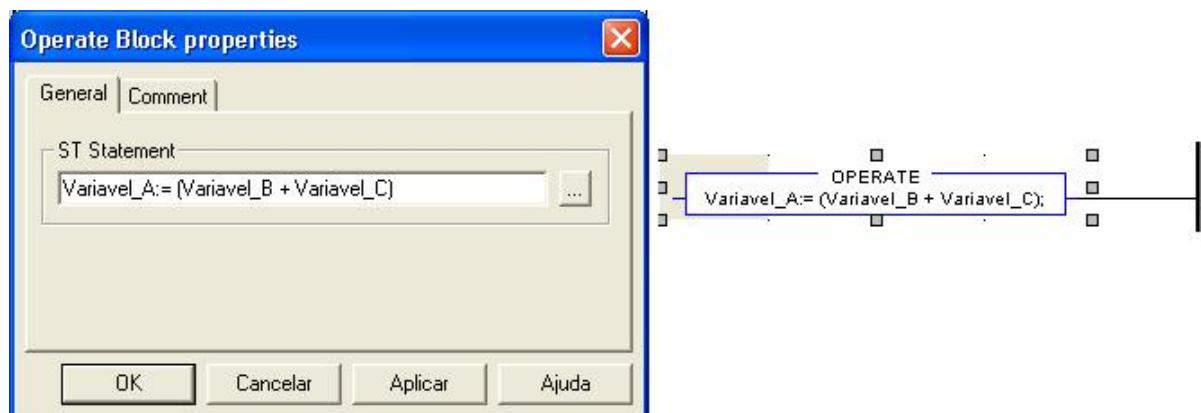
#### Definido a operação:

Com a caixa de diálogo "Automatically assign a variable to a new graphical object", desmarcada, é necessário dar dois cliques sobre o bloco operate para ser aberta a caixa de diálogo "Operate Block properties", (1) digitar a expressão (2), acionando o botão [...] (3) abre a caixa de diálogo "LD – Editor : Instance Selection" (4) onde é possível selecionar a variável desejada (5).



A utilização de uma variável já declarada no projeto, o bloco será atualizado, caso contrário a mesma deve ser declarada, como visto no “**Cap. 9 Variáveis**”.

### Resultado:



- **Bloco Compare.**

O Bloco Compare realiza comparação ( $<$ ,  $>$ ,  $\leq$ ,  $\geq$ ,  $=$ ,  $\neq$ ) entre dados de mesmo tipo, sendo seu resultado um dado booleano.

O Bloco compare é acessado através do ícone “**COMP.**” na barra de ferramentas.



Blocos Compare ocupam 1 linha e duas colunas e podem ser colocados em qualquer célula livre, exceto no trilho de alimentação direito.

A colocação do bloco compare cria automaticamente a conexão com os objetos vizinhos na esquerda e direita, caso sejam do tipo de dado BOOL e não exista célula livre entre os mesmos.

Um Bloco Compare pode conter até 4096 caracteres. Se todos os caracteres não puderem ser visualizados, o início da seqüência de caracteres será seguida por pontos (...).

### Ativação do Bloco Compare.

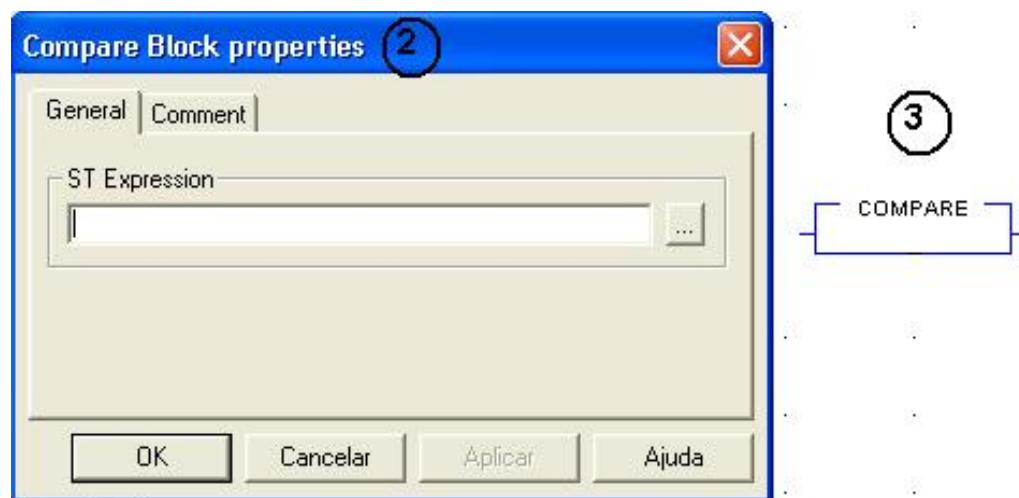
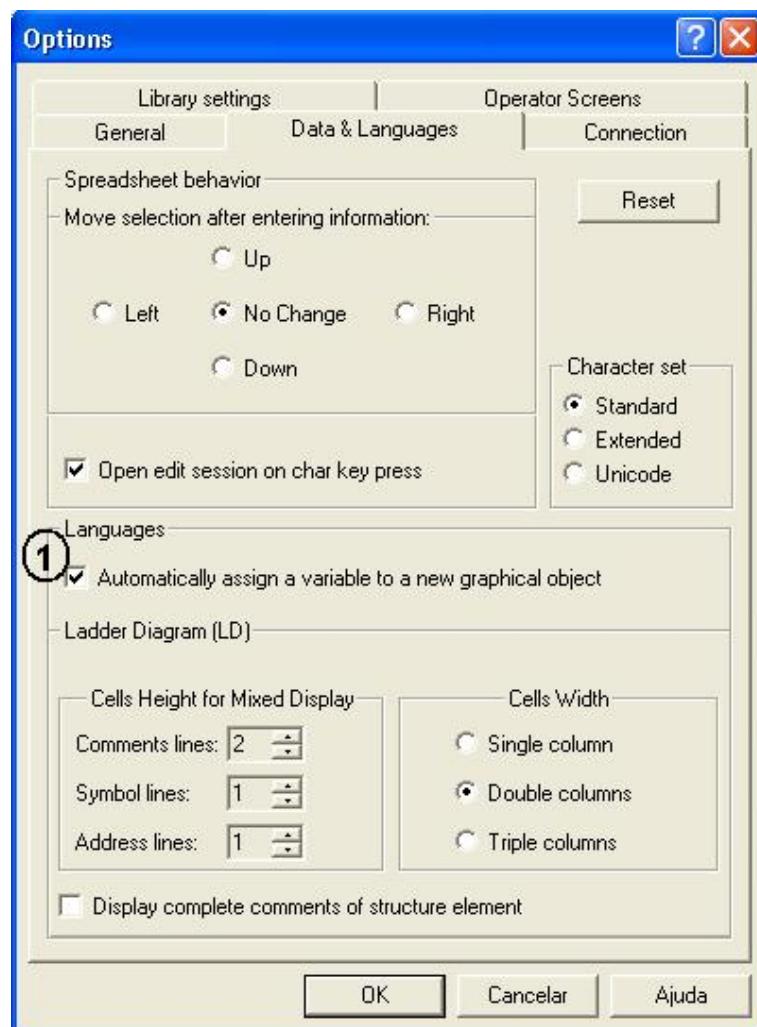
O Bloco Compare pode ser ativado das seguintes maneiras:

- Através dos comandos **Edit => New => Compare Block**;
- Através de clique no botão direito selecionar => **Common Objects => Compare Block**;
- Através da combinação de teclados **Ctrl + F7** ou
- Através do ícone “Compare block” na barra de ferramenta.

Para qualquer opção acima, o “prompt” do mouse mostra .



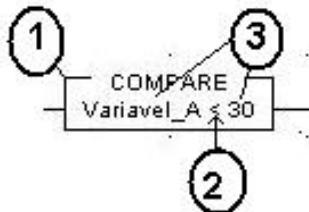
Na aba **Tools** na barra de ferramentas, na caixa “**Options**”, na aba “**Data and Languages**”, ao marcar a caixa “**Automatically assign a variable to a new graphical object**” (1), a caixa de diálogo para atribuição de variável para o elemento gráfico “**Compare block Properties**” (2) é aberta automaticamente quando se coloca o bloco compare na área de edição (3).



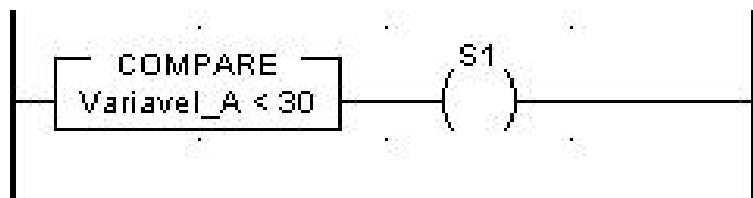
### Funcionamento do Bloco Compare:

O Bloco Compare realiza comparação ( $<$ ,  $>$ ,  $\leq$ ,  $\geq$ ,  $=$ ,  $\neq$ ) entre dados de mesmo tipo, sendo seu resultado um dado booleano.

O Bloco Compare (1) é composto por operandos (2) que são os elementos da instrução podendo variáveis ou constantes, operadores (3) que representa a operação a ser executada.



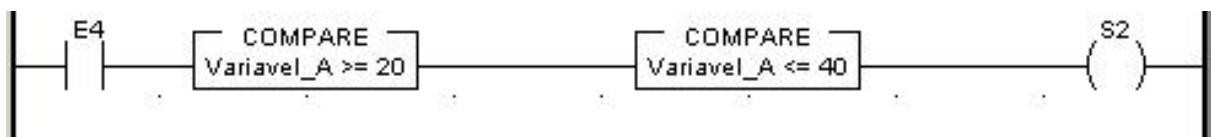
Quando o valor da variável “Variável\_A” for menor que 30, a saída S1 será ativada.



Por ser o resultado do bloco comparador um dado do tipo BOOL, é possível fazer diversas associações lógicas com o mesmo.

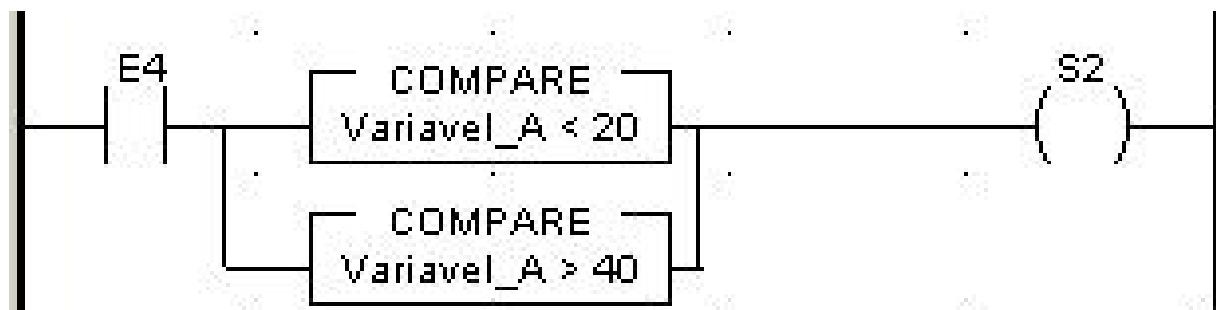
Detecção de valores dentro de uma faixa:

Estando a entrada E4 verdadeira, e o valor da variável “Variável\_A” for maior ou igual 20 e menor ou igual a 40, a saída S2 será ligada.



Detecção de valores fora de uma faixa:

Estando a entrada E4 verdadeira, e o valor da variável “Variável\_A” for menor que 20 ou maior que 40, a saída S2 será ligada.

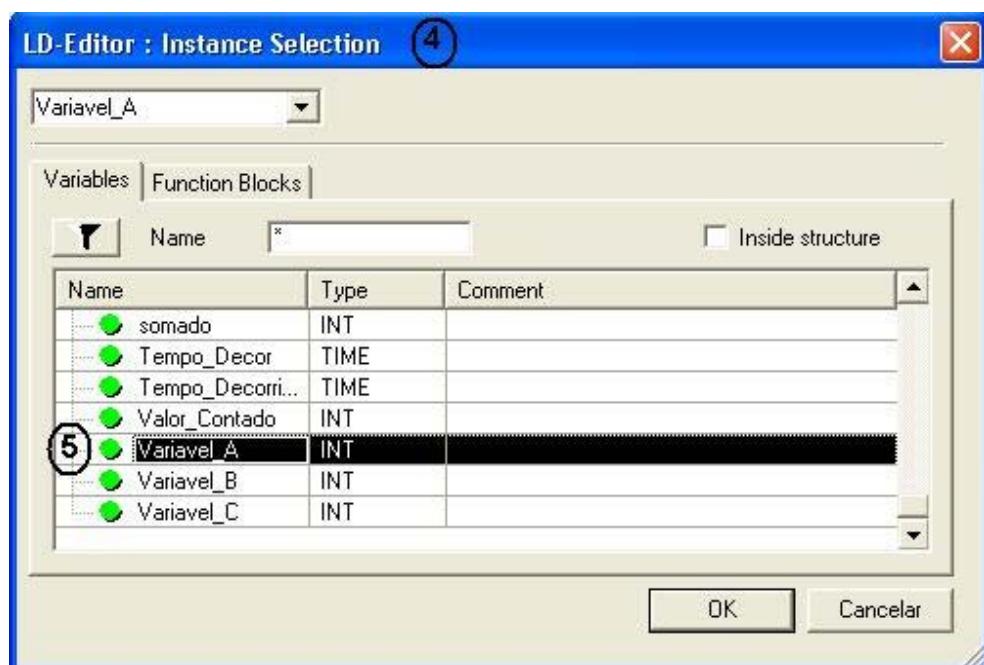
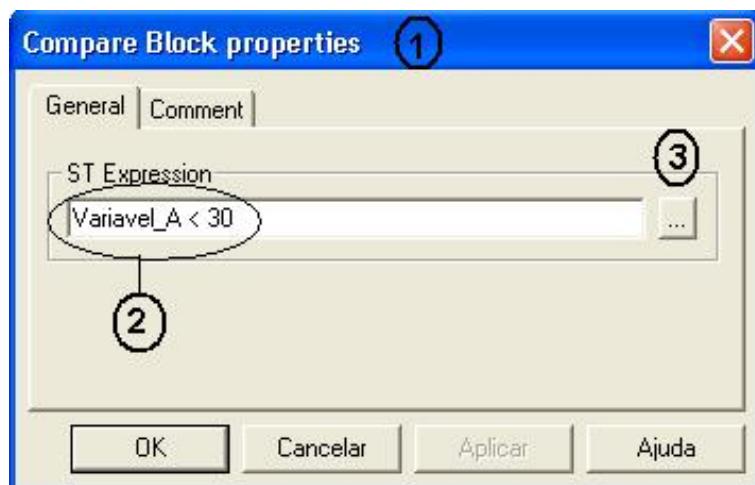


#### **Colocação do Bloco Compare:**

Ativar a colocação do bloco compare comentado anteriormente, posicionar no local desejado e clicar com botão esquerdo do mouse, fixando o bloco na célula, ou após a ativação da colocação do bloco compare, utilizar as teclas de navegação e posicionar o campo “cinza” na célula desejada e acionar a combinação de teclas **Ctrl + Enter**.

#### **Definido a operação:**

Com a caixa de diálogo “Automatically assign a variable to a new graphical object”, desmarcada, é necessário dar dois cliques sobre o bloco operate para ser aberta a caixa de diálogo “Compare Block properties”, (1) digitar a expressão (2), acionando o botão [...] (3) abre a caixa de diálogo “LD – Editor: Instance Selection” (4) onde é possível selecionar a variável desejada (5).



A utilização de uma variável já declarada no projeto, o bloco será atualizado, caso contrário a mesma deve ser declarada, como visto no “**Cap. 9 Variáveis**”.

#### Resultado:



- **Detectores de Borda.**

Diferentes implementações de objetos Ladder requerem o uso de variáveis internas “StateRam” (registros %M/%I). Sob esta condição de vários acessos de escritas para os registros %M/%I são possíveis em um “scan” um comportamento diferente “online” pode ser conseguido.



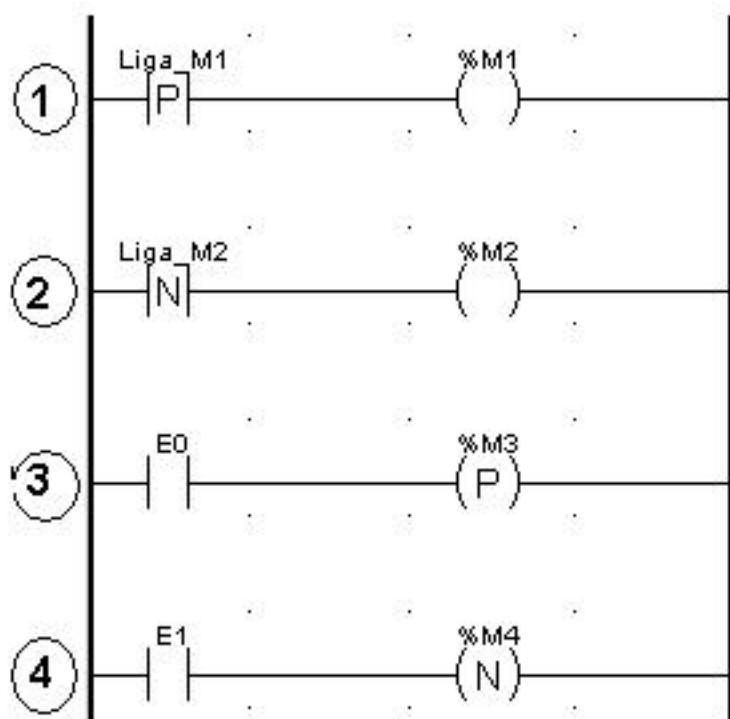
No Unity Pro, os objetos de detecção de borda são possíveis apenas para variáveis do tipo EBOOL.

A detecção de borda pode ser realizada tanto em contatos como em bobinas, que são:

- Detector da borda de subida (no acionamento)
- Detector da borda de descida (no desacionamento)
- Bobina ativa na borda de subida durante um “scan”
- Bobina ativa na borda de descida durante um “scan”

#### Exemplos de aplicação:

1. O estado interno %M1 será ligado durante um “scan” no acionamento da entrada “Liga\_M1”;
2. O estado interno %M2 será ligado durante um “scan” no desacionamento da entrada “Liga\_M2”;
3. O estado interno %M3 detecta a transição positiva da entrada E0, permanecendo ligada durante um “scan”.
4. O estado interno %M4 detecta a transição negativa da entrada E1, permanecendo acionada durante um “scan”.



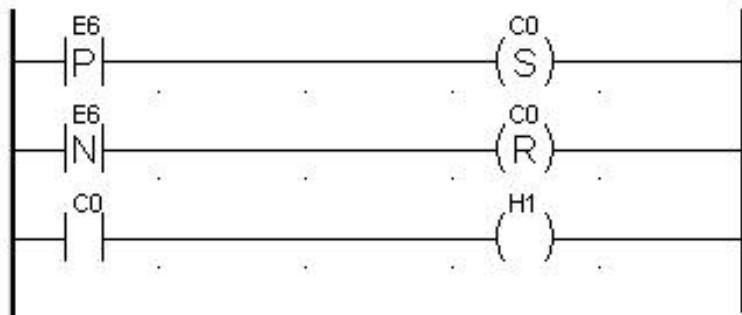
- **Instruções Set e Reset.**

As instruções Set e Reset são úteis para os casos onde se deseja memorizar o estado de um objeto conectado a uma variável dinâmica, dependendo da aplicação estas reduzem a complexidade da lógica.

O acesso as instruções Set e Reset é realizado através da barra de ferramenta nos ícones “Set coil” e “Reset coil”.

### **Exemplo de aplicação:**

Ao acionar a entrada E6, a borda de subida da mesma aciona a bobina “Set-S” de C0, no desacionamento da entrada E6, a borda de descida da mesma aciona a bobina “Reset-R” de C0, desligando-a.

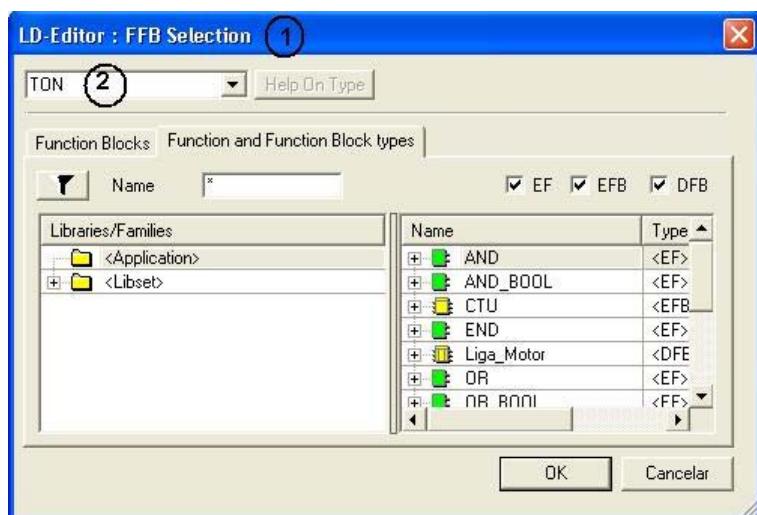


- Edição de Bloco de Função EFB – Temporizadores.**

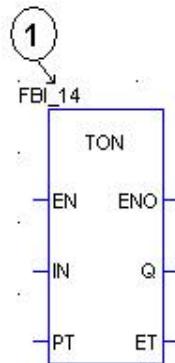
O Unity Pro possui três tipos de temporizadores, TON (Temporização na ligação), TOF (temporização no desligamento) e TP (temporização de pulso).

- Edição do Temporizador TON.**

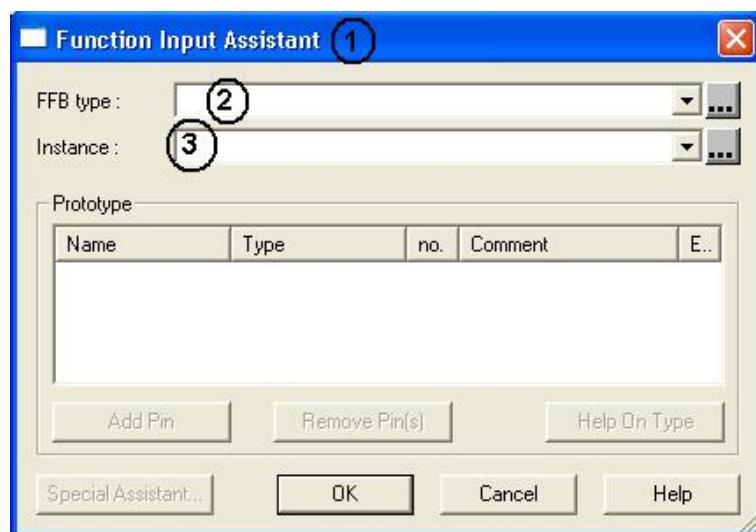
O acesso aos temporizadores pode ser feito através da barra de ferramentas no ícone “Data Selection” que ao ser acionado, é aberta a caixa de diálogo “LD – Editor :FFB Selection” (1) no campo (2) deve ser especificado o tipo de temporizador desejado e clicar o botão OK.



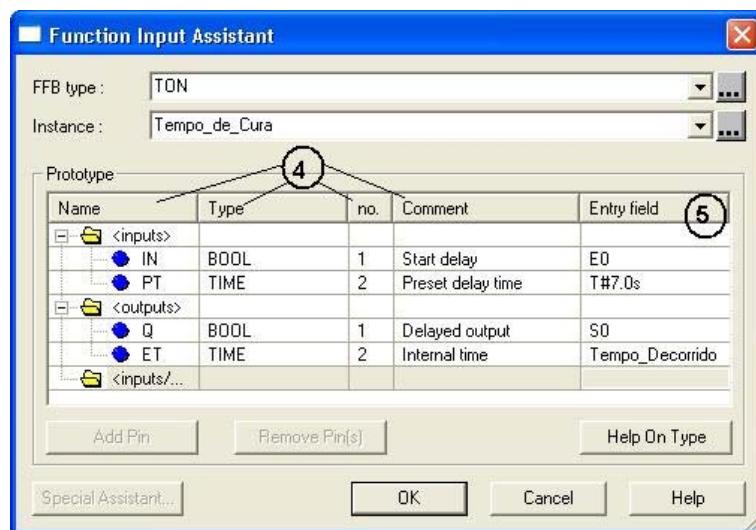
Ao clicar o botão OK, o “prompt” do mouse indica que deve ser posicionado no local desejado clicando no botão esquerdo do mouse o bloco é visualizado com o número seqüencial do mesmo no projeto (1).



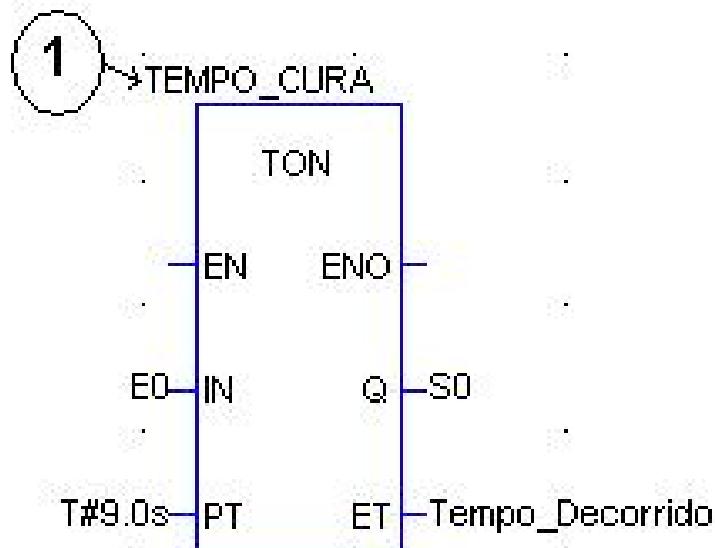
O acesso do temporizador pode ser feito também através do ícone “FFB Input assistant” (1), onde dever especificado o tipo de temporizador desejado no campo FFB Type (2) e nome no campo instance (3) é indicado o número seqüencial do bloco no projeto, o qual pode ser substituído por um nome.



Ao especificar o tipo do temporizador no campo FFB Type (2), o temporizador é localizado na biblioteca e suas informações são disponibilizadas (4). Na coluna “Entry field” (5) podem ser especificados os parâmetros e variáveis relacionadas ao mesmo.

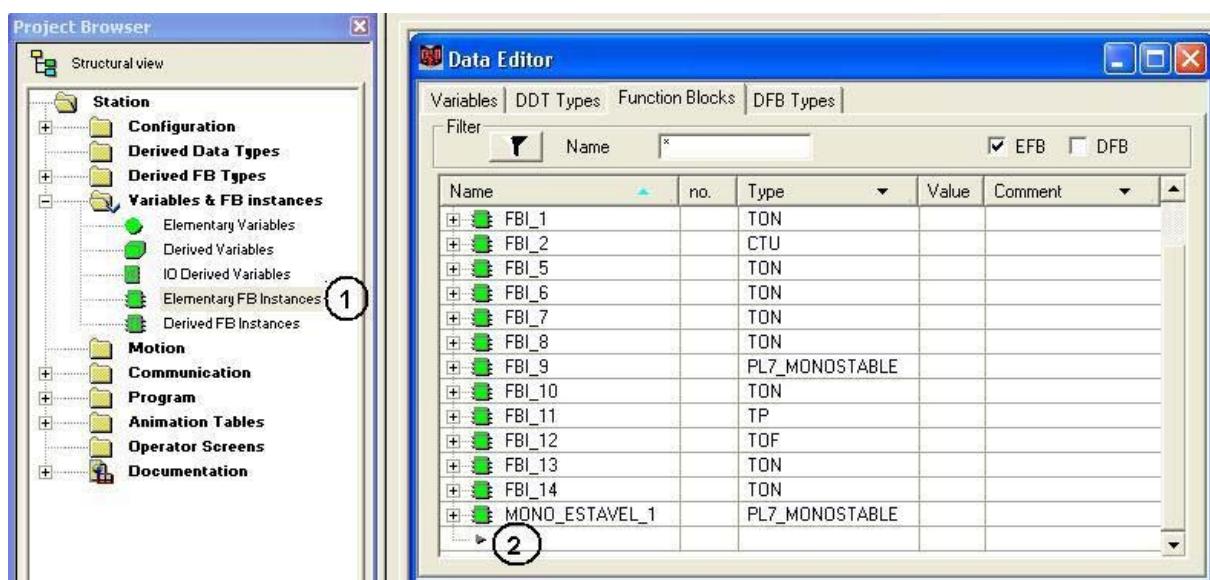


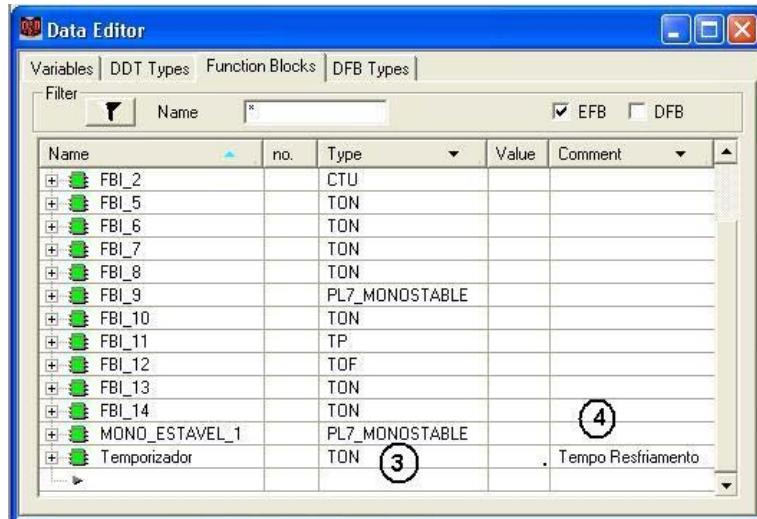
Ao clicar o botão OK, o “prompt” do mouse indica que deve ser posicionado no local desejado clicando no botão esquerdo do mouse o bloco é visualizado com o número sequencial do mesmo no projeto ou o nome desejado (1).



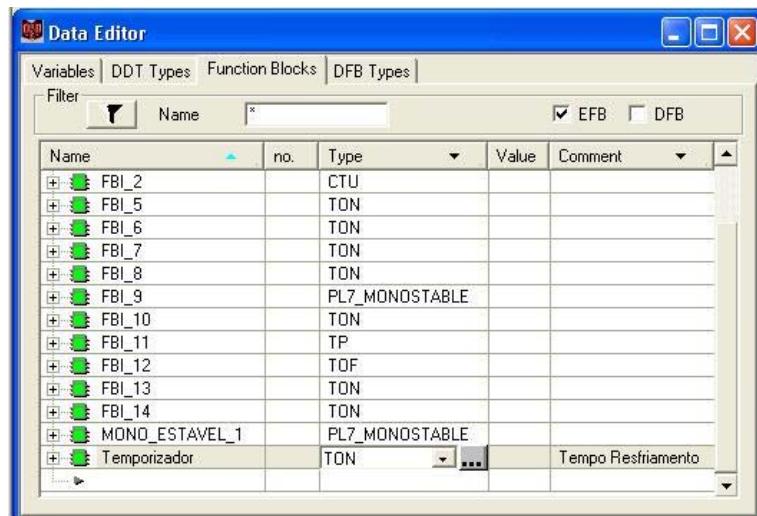
Os ícones “Data selection” e “FFB Input Assistant” podem ser acessados através da barra de ferramenta na aba “Edit” e clicando com botão direito na área de edição ladder.

O bloco temporizador pode ser editado através da declaração do mesmo na pasta “Elementary FB instances” (1), que ao ser selecionado abre a caixa de diálogo “Data Editor”, onde na última célula da coluna “Name” (2) deve ser especificado o nome do temporizador que ao acionar a tecla “Enter” o campo da coluna “Type” é especificado tipo TON (3), na coluna Comments pode ser digitado um comentário referente ao temporizador (4).

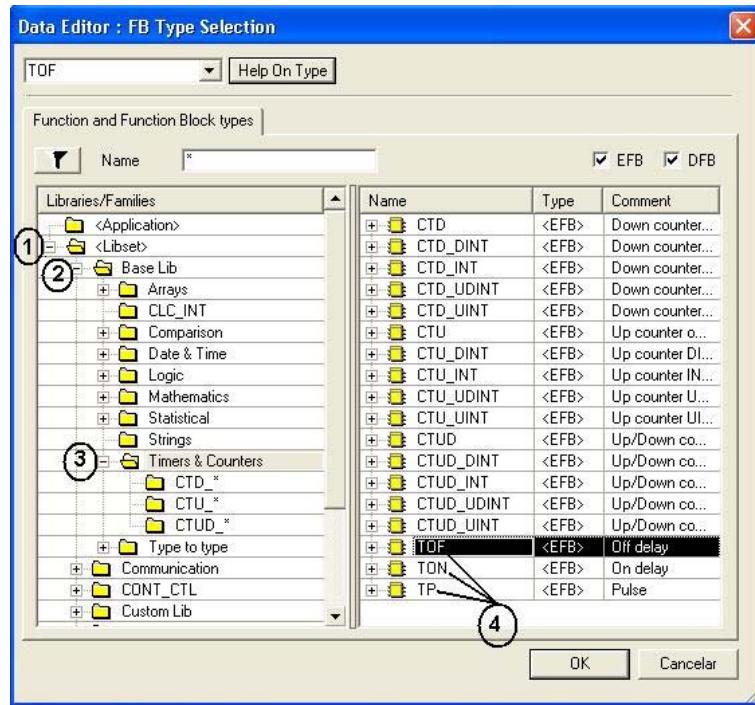




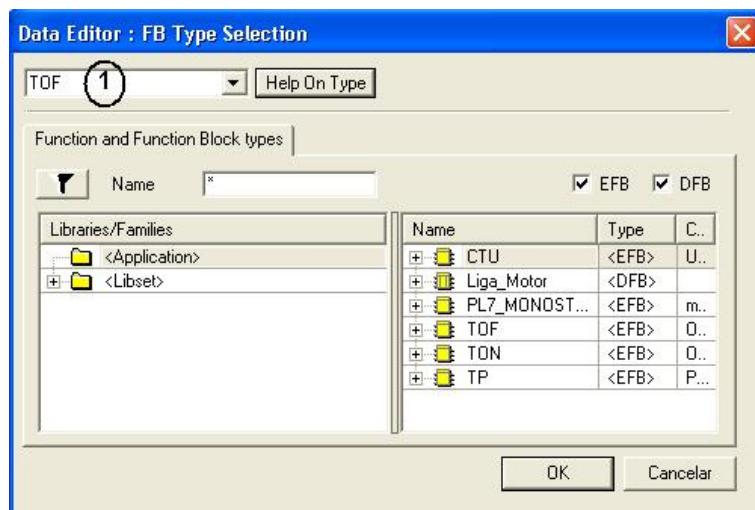
O tipo do Temporizador “TON” pode ser alterado clicando duas vezes na célula do tipo disponibilizando o botão [...].



Ao clicar no botão [...], abre a caixa de diálogo “Data Editor: FB Type Selection”, deve-se expandir as pastas “Libset” (1), “Base Lib” (2) e “Timers & Counters” (3) onde são disponibilizados todos os tipos de contadores e temporizadores (4), escolher o novo tipo de temporizador e acionar o botão OK.



O Tipo de temporizador pode também ser alterado sem a abertura das pasta das bibliotecas, para isto, deve-se especificar o novo tipo no campo **(1)** e acionar o botão **OK**.



O Tipo de temporizador e nome podem também serem alterados através do “Data Editor” selecionando o temporizador na aba “Function blocks” **(1)** e clicar com o botão direito do mouse e selecionar “Properties” **(2)**, a caixa “Data Properties” **(3)** é aberta permitindo as alterações desejadas.

(3)

Name	Value
Name	TEMPO_DE_CURA
Comment	
RW program	<input checked="" type="checkbox"/>
Type	TON
Category	<EFB>
Size	20
Diag	
Used	0

Variables | DDT Types | Function Blocks | DFB Types

Filter Name

1

Cut  
Copy  
Paste  
Delete  
Insert New  
Customize Column...  
Expand Data  
Analyze  
Analyze type  
Search Data  
Go to type definition  
Hyperlink...  
Properties 2  
Purge unused FB Instances  
Export Filtered  
Export Selected

2

Name	Type
FBI_5	
FBI_6	
FBI_7	
FBI_8	
FBI_9	
FBI_10	
FBI_11	
FBI_12	
FBI_13	
FBI_14	
FBI_15	
MONO_ESTAVEL_1	
T	
TE	
Tempo_1	
TEMPO_CURA	
Tempo_cura2	
TEMPO_DE_CURA	TON
TEMPORIZADOR	TON

Função dos pinos do bloco temporizador TON:

Entradas:

EN => Entrada boolena que quando acionada habilita o funcionamento do temporizador  
IN => Entrada booleana que quando acionada inicia a temporização desejada e quando desabilitada reseta o temporizador

PT => Entrada do valor de tempo de pré-seleção “Preset” com tipo de dado “TIME”, tempo que depois de decorrido a saída Q será acionada.

Saídas:

ENO => Saída de indicação de bloco habilitado

Q => Saída booleana acionada no disparo do temporizador

ET => Saída do valor do tempo decorrido, com tipo de dado “TIME”.

Funcionamento do Temporizador TON:

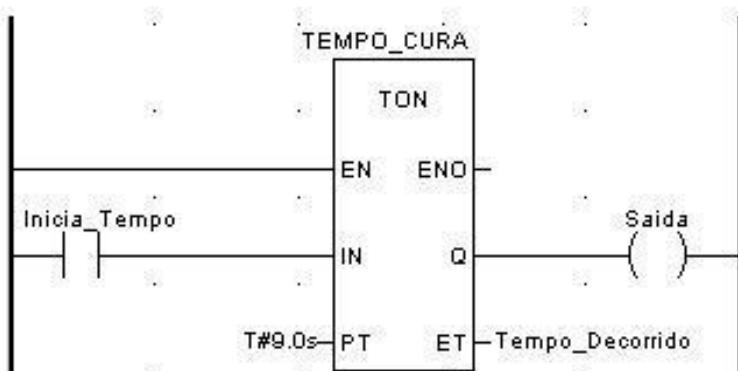
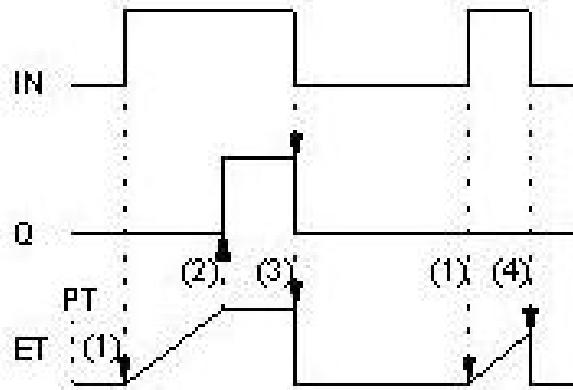


Diagrama de tempo:

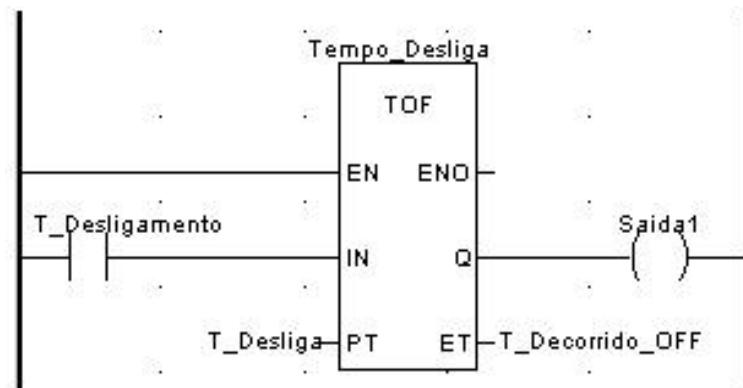


- (1) se a entrada “Inicia\_Tempo” for 1, o tempo interno (ET) “Tempo\_Decorrido” inicia;
  - (2) se o tempo interno “Tempo\_Decorrido” atinge o valor de PT, a saída Q “Saída” torna-se 1;
  - (3) se a entrada “Inicia\_Tempo” torna-se “0”, a saída Q “Saída” vai para “0” e o tempo interno é parado/resetado;
  - (4) se a entrada “Inicia\_Tempo” tornar-se “0”, antes do tempo interno atingir o valor de PT, o tempo interno para/reseta sem levar a saída Q “Saída” para “1”.

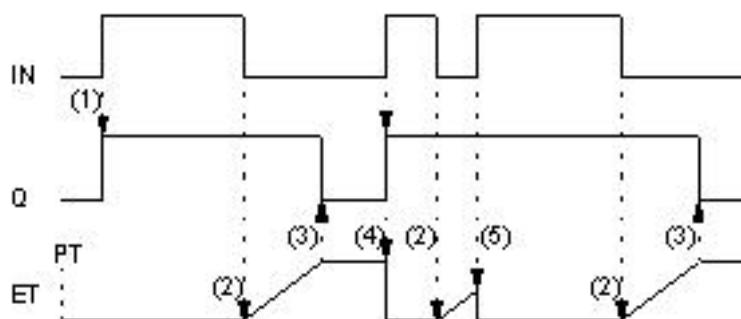
- **Edição do Temporizador TOF.**

Toda a edição e configuração do temporizador TOF são idênticas ao temporizado TON, diferenciando-se apenas no funcionamento.

#### Funcionamento do Temporizador TOF:



## Diagrama de tempo:



- (1) se a entrada “T\_Desligamento” for “1”, a saída “Q” vai para “1”;  
 (2) se a entrada “T\_Desligamento” for “0”, o tempo interno (ET) “T\_Decorrido\_OFF” é iniciado;

- (3) se o tempo interno (ET) "T\_Decorrido\_OFF" atinge o valor de (PT) "T\_Desliga", a saída "Q" "Saida1" é levada para "1";
- (4) se a entrada "T\_Desligamento" vai para "1", a saída Q vai para "1", e o tempo interno é parado/resetado;
- (5) se a entrada "T\_Desligamento" for 1 antes do tempo interno ter alcançado o valor de PT "T\_Desliga", o tempo interno (ET) "T\_Decorrido\_OFF" é parado sem colocar de volta a saída "Q" "Saida1" para "0".

- **Edição do Temporizador TP.**

Toda a edição e configuração do temporizador TP são idênticas ao temporizado TON, diferenciando-se apenas no funcionamento.

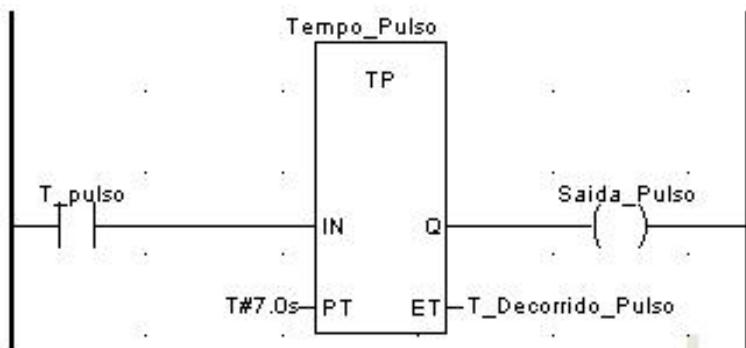
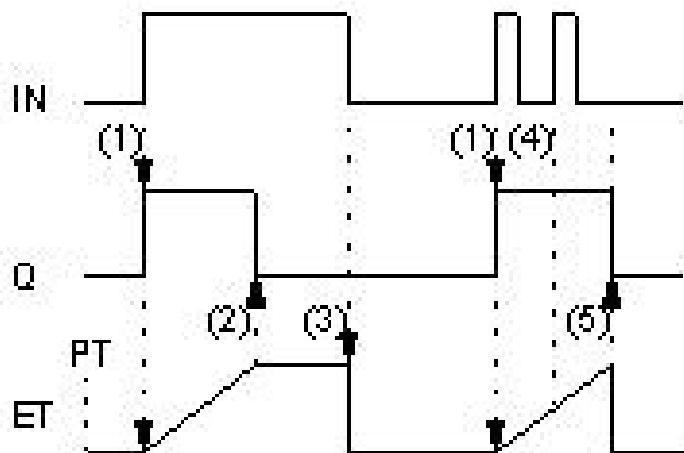


Diagrama de tempo:



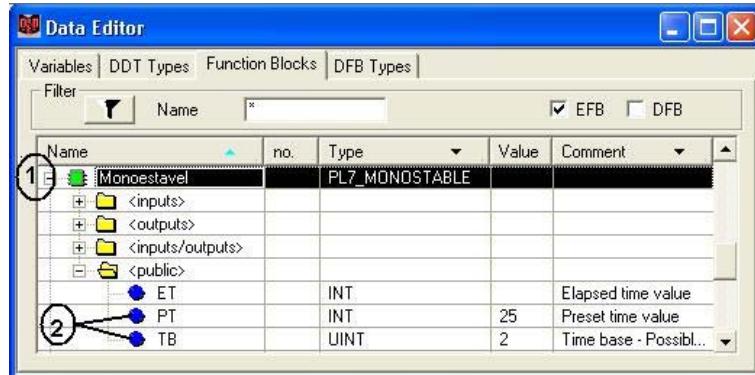
- (1) se a entrada "IN" "T\_Pulso" for "1", a saída "Q" "Saida\_Pulso" torna-se "1" e tempo interno (ET) "T\_Decorrido\_Pulso" inicia;
- (2) se o tempo interno (ET) alcança o valor (PT) "T#7.0s", a saída "Q" "Saida\_Pulso" torna-se "0" (independente da entrada IN);
- (3) se a entrada "IN" "T\_Pulso" vai para "0", o tempo interno (ET) "T\_Decorrido\_Pulso" é zerado;
- (4) se o tempo interno (ET) ainda não alcançou o valor de PT, o tempo interno (ET) "T\_Decorrido\_Pulso" não é afetado por um novo pulso na entrada "IN" "T\_Pulso".

- **Edição de Bloco de Função EFB – Monoestável.**

O bloco monoestável no Unity Pro é denominado de "PL7\_MONOSTABLE", é um bloco que permite a geração de pulso com duração precisa.

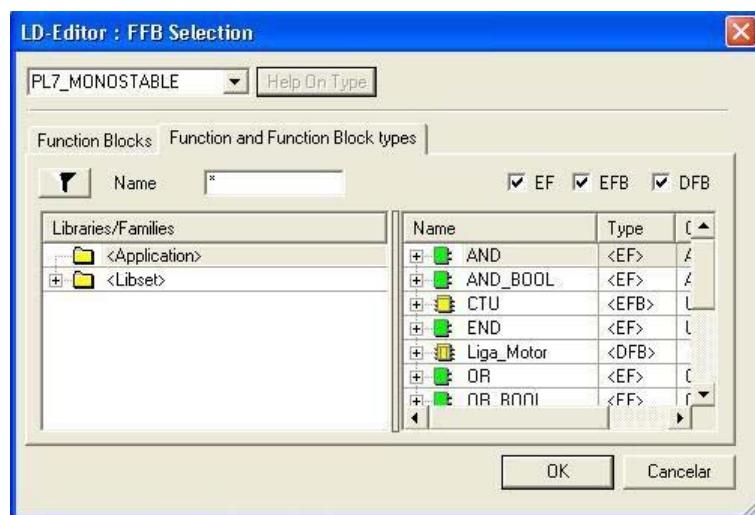
O acesso ao monoestável é feito da mesma maneira que os temporizadores, devendo ser declarado na pasta "Elementary FB instance".

A configuração é feita através do “Data Editor” na aba “Function block” selecionar o monoestável (1) e na pasta “Public” (2) especificar os parâmetros “PT” Tempo de preset e “TB” base de tempo do valor especificado em “PT”



A colocação do Monoestável no ladder é feita através da barra de ferramentas nos ícones:

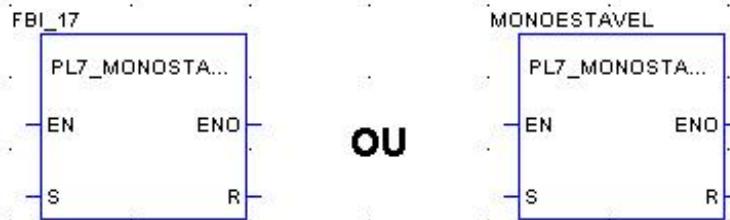
“Data Selection”;



“FFB Input assistant”.



Clicando com o mouse na área de edição o bloco é colocado na mesma:



Esta instrução está disponível apenas no CLP Premium.

### Função dos pinos do Monoestável:

#### Entradas:

EN => Entrada booleana de habilitação do bloco

S => Entrada booleana de inicialização da temporização na borda de subida, estado interno associado "Mn\_i.S"

#### Saídas:

ENO => Saída booleana de indicação de habilitação do bloco

R => Saída booleana que é ativa "1" quando o tempo interno (ET) for diferente do tempo de preset (PT) ou diferente de "0". estado interno associado "Mn\_i.R"

#### Parâmetros:

ET => Dado tipo INT, valor corrente do monoestável que decresce o valor de PT até "0".

Este valor pode ser lido e testado mas não pode ser escrito pelo programa. registro interno associado "Mn\_i.ET"

PT => Dado tipo INT, valor na faixa de 0 e 9999, é conhecido como preset do monoestável.

Pode ser escrito, lido e testado pelo programa. O valor do pulso gerado pelo monoestável é igual ao valor de PT x TB. registro interno associado "Mn\_i.PT"

TB => Dado tipo UINT, tempo base do ciclo de programa, pode ser:

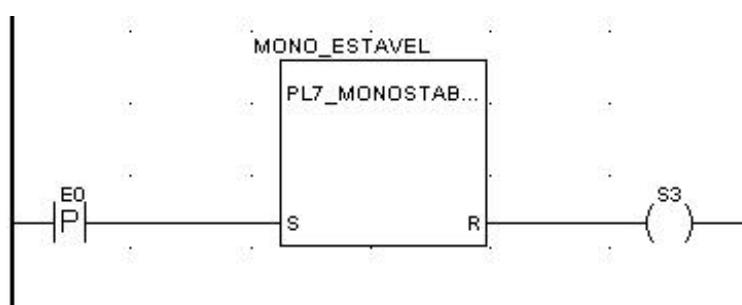
TB = 8: 1 mn (valor default),

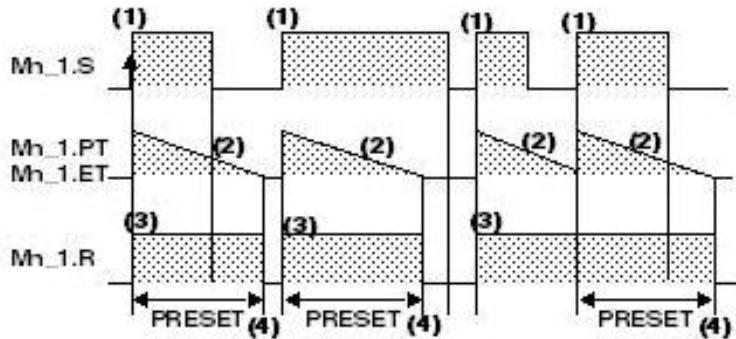
TB = 4: 1 s

TB = 2: 100 ms,

TB = 1: 10 ms.

#### Funcionamento:





- (1) Quando a borda de subida é detectada na entrada S do monoestável, o tempo interno ET assume o valor de PT.
- (2) O valor do tempo interno (ET) decresce para “0” na taxa de uma unidade para cada pulso do tempo base TB.
- (3) A saída R (Running) vai para “1” sempre que o valor de ET é diferente de “0”.
- (4) Quando o valor de ET é igual a “0”, a saída R retorna para “0”.

- **Edição de Bloco de Função EFB – Contador.**

Os contadores default disponíveis no Unity Pro utilizam tipo de dados INT podendo ser crescente, decrescente.

Contadores Decrescente:

CTD => funciona apenas com tipo de dados INT de 16 bits, faixa de -32768 a 32767;

CTD\_INT=> funciona com tipo de dados INT de 16 bits, faixa de -32768 a 32767;

CTD\_DINT =>funciona com tipo de dados Duplo INT de 32 bits, faixa de -2147483648 a 2147483647;

CTD\_UINT =>funciona com tipo de dados INT sem sinal de 16 bits, faixa de 0 a 65535

CTD\_UDINT =>funciona com tipo de dados Duplo INT sem sinal de 32 bits, faixa de 0 a 4294967295.

Contadores Crescente:

CTU => funciona apenas com tipo de dados INT (16 bits);

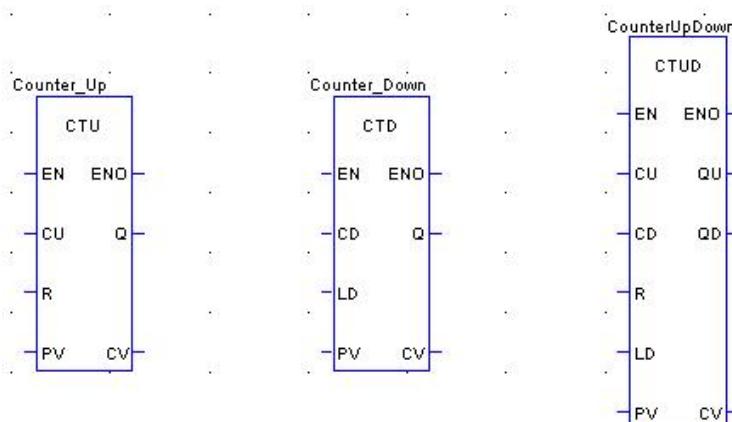
CTU\_INT=>funciona com tipo de dados INT (16 bits);

CTU\_DINT =>funciona com tipo de dados Duplo INT de 32 bits,

CTU\_UINT =>funciona com tipo de dados INT sem sinal (16 bits)

CTU\_UDINT =>funciona com tipo de dados Duplo INT sem sinal (32 bits).

O acesso aos blocos contadores é feito da mesma maneira que o bloco temporizador e bloco monoestável, devendo ser declarado na pasta “Elementary FB instance”.



## Função dos pinos dos blocos contadores.

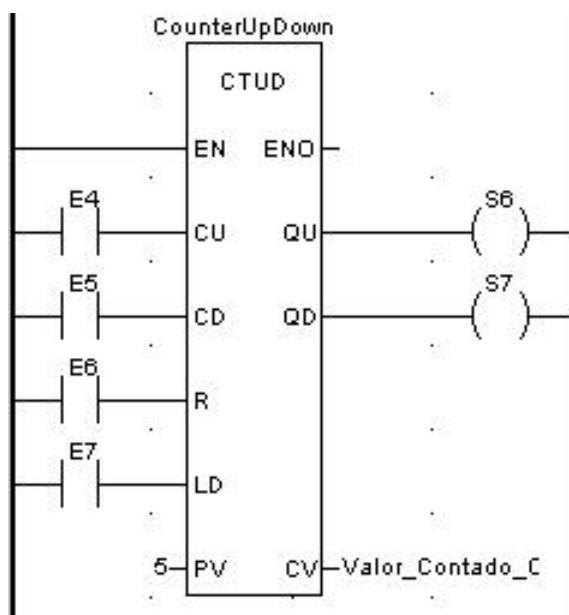
### Entradas:

EN => Entrada booleana que quando em “1” habilita o bloco;  
 CU => Entrada booleana que na borda de subida, o bloco incrementa o valor contado (CV)  
 CD => Entrada booleana que na borda de subida, o bloco decresce o valor contado (CV)  
 R => Entrada booleana que na borda de subida, o valor contado (CV) é zerado.  
 LD => Entrada booleana que na borda de subida, carrega o valor de preset (PV) no valor contado (CV)  
 PV => Registro de entrada do valor de preset a ser contado.

### Saídas:

ENO => Saída booleana de indicação de bloco habilitado.  
 Q => Saída booleana, que é ativa quando o valor contado (CV) é igual ao valor de preset (PV) para o contador CTU, para o contador CTD quando CV for “0”.  
 QU => Saída booleana, que é ativa quando o valor contado (CV) é igual ao valor de preset (PV) na contagem crescente.  
 QD => Saída booleana, que é ativa quando o valor contado (CV) é igual a “0” na contagem decrescente.  
 CV => Registro de saída que contém o valor corrente da contagem.

### Exemplo de aplicação do contador Crescente/Decrescente



- **Conversores de tipos de dados.**

Em determinadas aplicações torna-se necessário o processamento entre tipos de dados diferentes, processá-los diretamente é impossível, através da utilização dos diversos tipos de conversores, esta tarefa torna-se possível na maioria dos casos.

### **Tipos de conversores do Unity Pro:**

ASCII\_TO\_STRING: Conversão de um dado no formato ASCII para formato STRING.  
 ASCII\_TO\_STRING\_INV: Conversão de dado no formato ASCII para formato STRING invertida.  
 BCD\_TO\_INT: Conversão de um valor em BCD em Inteiro.  
 BIT\_TO\_BYTE: Conversão de BIT em BYTE.  
 BIT\_TO\_WORD: Conversão de BIT em WORD.  
 BOOL\_TO\_\*\*\*: Conversão de BOOL em dado tipo BYTE, WORD, DWORD, INT, DINT, UINT, UDINT, REAL ou TIME.  
 BYTE\_AS\_WORD: Conversão de BYTE em WORD.  
 BYTE\_TO\_BIT: Conversão de BYTE em BIT.

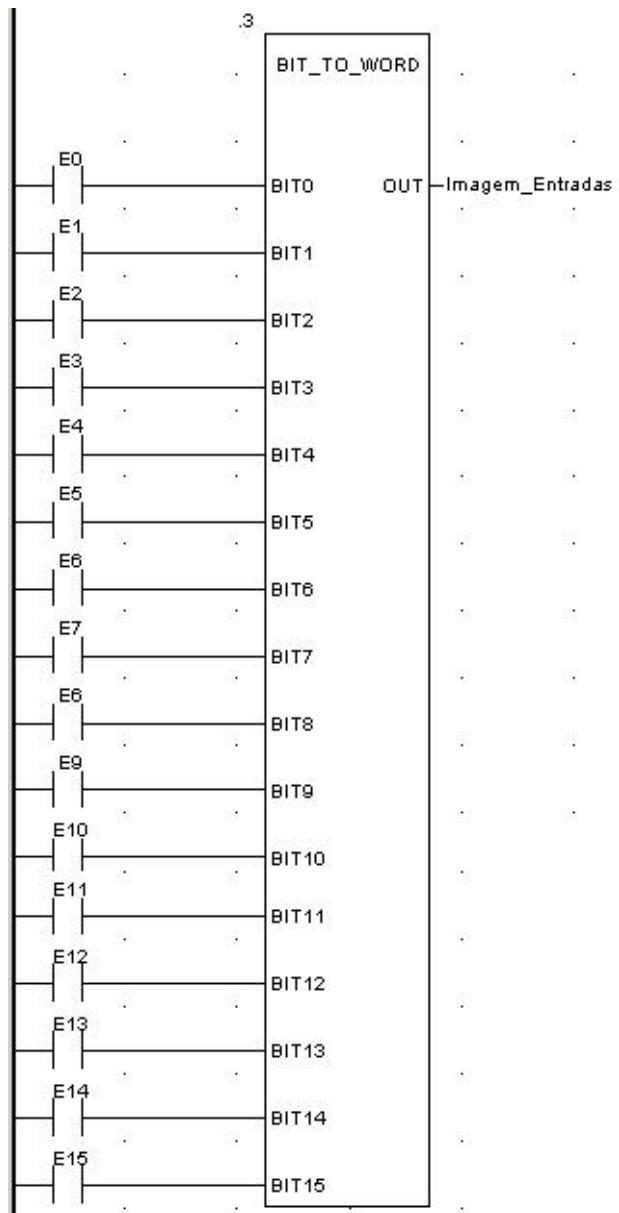
**BYTE\_TO\_\*\*\*:** Conversão de BYTE em dado tipo BOOL, WORD, DWORD, INT, DINT, UINT, UDINT, REAL or TIME.  
**DATE\_TO\_STRING:** Conversão de uma variável no formato DATE em uma STRING.  
**DBCD\_TO\_\*\*\*:** Conversão de Duplo inteiro em BCD para Binário  
**DEG\_TO\_RAD :** Conversão de graus em radianos  
**DINT\_AS\_WORD:** Conversão de Duplo Inteiro em WORD  
**DINT\_TO\_\*\*\*:** Conversão de Duplo Inteiro em dado tipo BOOL, BYTE, WORD, DWORD, INT, UINT UDINT, REAL ou TIME.  
**DINT\_TO\_DBCD:** Conversão de Duplo Inteiro em Duplo BCD  
**DT\_TO\_STRING:** Conversão de um dado tipo DT (Data and Time) em uma STRING.  
**DWORD\_TO\_\*\*\*:** Conversão de um dado tipo Dupla Word em dado tipo BOOL, BYTE, WORD, INT, DINT, UINT, UDINT, REAL ou TIME.  
**GRAY\_TO\_INT:** Conversão de um dado Inteiro no código Gray em um dado tipo inteiro codificado em binário.  
**INT\_AS\_DINT:** Concatenação de dois dados inteiros para formar um duplo inteiro.  
**INT\_TO\_\*\*\*:** Conversão de Inteiro em dado tipo BOOL, BYTE, WORD, DWORD, DINT, UINT, UDINT, REAL ou TIME.  
**INT\_TO\_BCD:** Conversão de Inteiro em BCD  
**INT\_TO\_DBCD:** Conversão de Inteiro em Duplo BCD.  
**RAD\_TO\_DEG:** Conversão de radianos para graus.  
**REAL\_AS\_WORD:** Conversão de REAL para WORD  
**REAL\_TO\_\*\*\*:** Conversão de REAL para um dado tipo BOOL, BYTE, WORD , DWORD, INT, DINT, UINT, UDINT ou TIME.  
**REAL\_TRUNC\_\*\*\*:** Conversão de REAL para dado tipo INT, DINT, UINT ou UDINT com arredondamento.  
**STRING\_TO\_ASCII:** Conversão de uma STRING em ASCII  
**STRING\_TO\_ASCII\_INV:** Conversão de uma STRING em ASCII com inversão  
**STRING\_TO\_\*\*\* :** Conversão de uma STRING em dado tipo INT, DINT ou REAL.  
**TIME\_AS\_WORD:** Conversão de um dado tipo TIME em dois valores de saída em WORD.  
**TIME\_TO\_\*\*\*:** Conversão de um dado tipo TIME em um dado tipo BOOL, BYTE, WORD, DWORD, INT, DINT, UINT, UDINT ou REAL.  
**TIME\_TO\_STRING:** Conversão de um dado tipo TIME em uma STRING  
**TOD\_TO\_STRING:** Conversão de uma variável em TOD (Time of Day) em uma STRING.  
**UDINT\_AS\_WORD:** Conversão de um dado duplo inteiro sem sinal em uma WORD  
**UDINT\_TO\_\*\*\*:** Conversão de um dado duplo inteiro em um dado tipo BOOL, BYTE, WORD, DWORD, INT, DINT, UINT, REAL ou TIME.  
**UINT\_TO\_\*\*\*:** Conversão de um dado tipo Inteiro sem sinal em um dado tipo BOOL, BYTE, WORD, DWORD, INT, DINT, UDINT, REAL ou TIME.  
**WORD\_AS\_BYTEx:** Conversão de uma WORD em um BYTE  
**WORD\_AS\_DINT:** Conversão de uma WORD em um dado tipo Duplo Inteiro  
**WORD\_AS\_REAL:** Conversão de uma WORD em um dado em REAL  
**WORD\_AS\_TIME:** Conversão de uma WORD em um dado tipo TIME  
**WORD\_AS\_UDINT:** Conversão de uma WORD em um dado Duplo Inteiro sem sinal  
**WORD\_TO\_BIT:** Conversão de uma WORD em BIT  
**WORD\_TO\_\*\*\*:** Conversão de uma WORD em um dado tipo BOOL, BYTE, DWORD, INT, DINT, UINT, UDINT, REAL ou TIME.  
**\*\*\*\_TO\_STRING:** Converte uma variável com dado tipo INT, DINT ou REAL em uma STRING.

O acesso aos conversores de dados é feito através da barra de ferramentas nos ícones “Data Selection” e “FFB Input Assistant”, que quando acionado é aberta a caixa de diálogo para especificação do bloco desejado.

#### **Exemplo de Utilização de Conversão de Dados:**

- Imagem das entradas de um módulo digital:

A variável “Imagen\_Estradas” contém o valor em Hexadecimal de acordo com o bit (Entrada) acionada.



#### b) Aplicação de conversores INT\_TO\_REAL e REAL\_TO\_INT

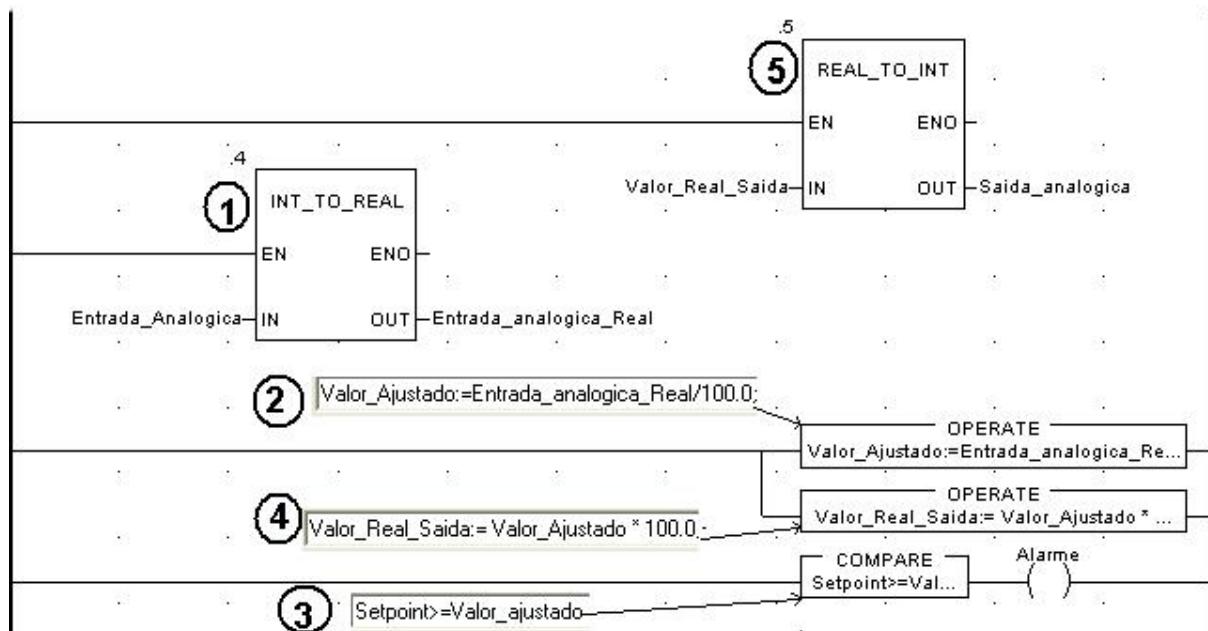
Os módulos analógicos funcionam com tipo de dado INT (Inteiro, sem vírgula), isto pode não ser ideal quando se deseja precisão nos valores tratados, usa-se então conversores associados com blocos operate para atingir a precisão desejada, como mostra o exemplo seguinte.

#### Funcionamento:

O bloco conversor **(1)** faz com que o valor da entrada analógica passe a ter uma casa após a vírgula na variável “Entrada\_analogica\_Real” que devido o operate **(2)** que divide o valor por 100.0, a variável “Valor\_Ajustado” passa a ter este valor com a vírgula deslocada duas casas permitindo que o comparador **(3)** faça a comparação com um valor com duas casa após a vírgula a fim de gerar o sinal de alarme.

Após o processamento através do operate **(4)**, o valor ajustado é multiplicado por 100.0 para que a variável “Valor\_Real\_Saida” tenha o mesmo valor da variável

“Entrada\_analogica\_Real” permitindo que o conversor (5) converta o valor processado para inteiro a fim de ser fornecido ao módulo de saída.



- Instruções Aritméticas

O Unity Pro disponibiliza várias instruções aritméticas as quais podem ser realizadas utilizando o bloco operate como visto anteriormente ou através de Blocos de Funções Aritméticas específicos.

#### Blocos de Funções Aritméticas:

ABS: Valor absoluto de uma entrada e o atribui a uma saída, ou seja, calcula o módulo da entrada

ACOS: Arco coseno

ADD: Adição

ADD\_TIME: Adição com o tipo de dado TIME

ASIN: Arco seno

ATAN: Arco tangente

COS: Coseno

DEC: Decremento de uma variável

DIV: Divisão

DIVMOD: Divisão de uma entrada dividendo por uma entrada divisor e o resultado é atribuído a uma saída quociente, bem como o resto

EXP: Exponencial Natural

EXPT\_REAL\_\*\*\*: Exponencial de um valor em REAL por outro valor em REAL

INC: Incremento de uma variável

LN: Logaritmo Natural

LOG : Logaritmo base 10

MOD: Módulo, calcula o resto de uma divisão

MOVE: Atribuição/ transferência

MUL: Multiplicação

NEG: Negação

SIGN: Detecção de sinal negativo

SIN: Seno

SUB: Subtração

SUB\_TIME: Subtração de dados tipo TIME

SQRT\_\*\*\* : Raiz Quadrada

TAN: Tangente

COMP\_DB: Comparação de duas entradas em REAL

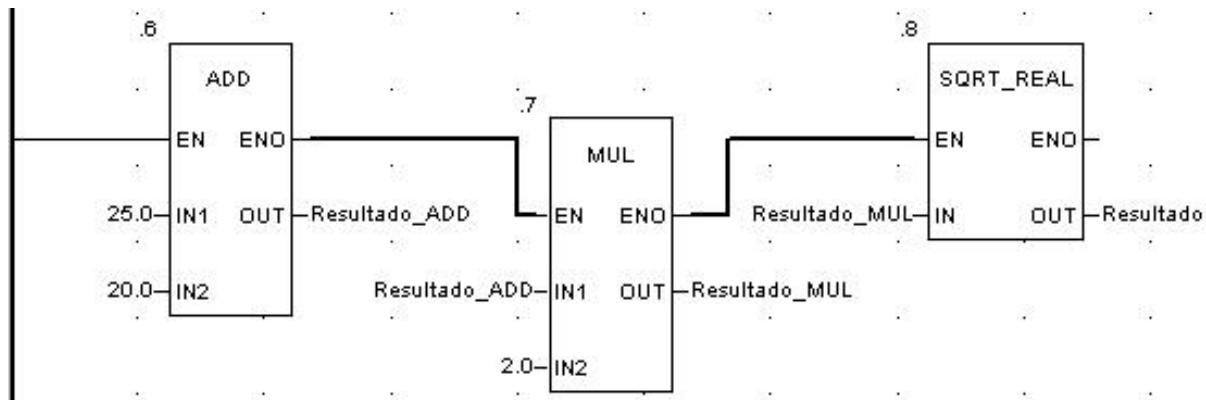
K\_SQRT: Determina o peso de uma Raiz Quadrada

MULDIV\_W: Determina o peso de multiplicação/divisão de três números.

SUM\_W: Determina o peso de uma soma de três valores.

O acesso aos Blocos de Funções Aritméticas, é feito através da barra de ferramentas nos ícones “Data Selection” e “FFB Input Assistant”, que quando acionado é aberta a caixa de diálogo para especificação do bloco desejado.

### Exemplo de Utilização de Blocos de Função Aritméticas:



### Endereçamento indexado por Word e por Bit.

#### Endereçamento indexado.

O endereço é dito indexado quando o mesmo é formado partir de mais de um elemento, seja posição de memória ou de um endereço.

Exemplos:

1 – Indexação por endereço de memória.

%MW20 [%MW15] => Considerando que o conteúdo de %MW15 seja 5, o endereço resultante será %MW25

%MW30:9 => Refere-se a 9 posições de memória a partir de %MW30, ou seja, de %MW30 a %MW38.

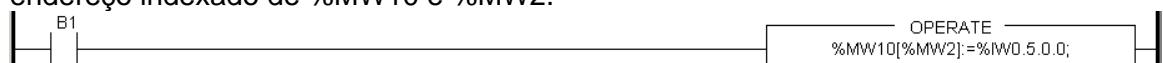
2 – Indexação por endereço físico.

%I0.1.0:8 => Refere-se a 8 entradas a partir de %I0.1.0, ou seja, de %I0.1.0 a %I0.1.7.

%Q0.2.0:16 => Refere-se a 16 entradas a partir de %Q0.2.0, ou seja, de %Q0.2.0 a %Q0.15

#### Exemplos de aplicação.

Com o acionamento de B1, o valor da entrada analógica %IW0.5.0.0 será escrita no endereço indexado de %MW10 e %MW2.



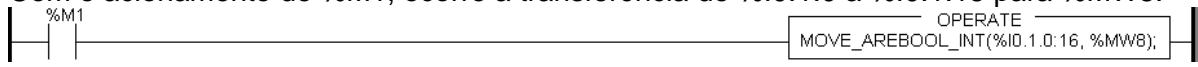
Com o acionamento de B2, ocorre a transferência Indexada de %MW20 e %MW12 para o endereço indexado por %MW4 e %MW6



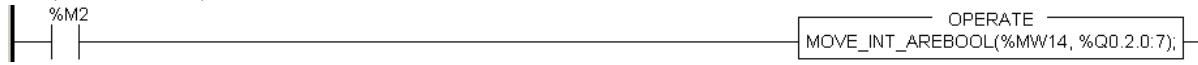
Com o acionamento de %M0, ocorre a transferência Indexada de %I0.1.0 a %I0.1.7 para o endereço indexado por %MW6 e %MW14



Com o acionamento de %M1, ocorre a transferência de %I0.1.0 a %I0.1.15 para %MW8.



Com o acionamento de %M2, ocorre a transferência do conteúdo de %MW14 para as saídas %Q0.2.0 a %Q02.6.



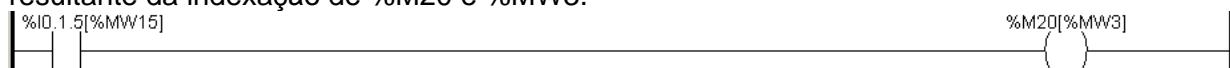
O bit 4 de %MW6 ativa Aux\_2.



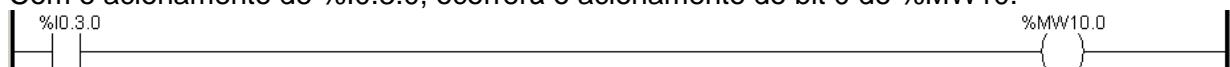
O valor 342 será escrito em 4 posições de memória a partir de %MW10, ou seja, nas posições de memória %MW10 a %MW13.



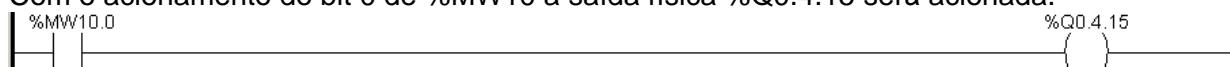
O endereço resultante da indexação de %I0.1.5 e %MW15 quando acionado, acionará o bit resultante da indexação de %M20 e %MW3.



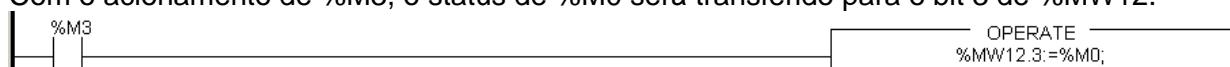
Com o acionamento de %I0.3.0, ocorrerá o acionamento do bit 0 de %MW10.



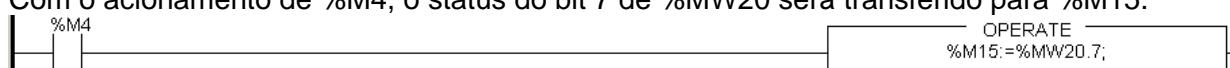
Com o acionamento do bit 0 de %MW10 a saída física %Q0.4.15 será acionada.



Com o acionamento de %M3, o status de %M0 será transferido para o bit 3 de %MW12.



Com o acionamento de %M4, o status do bit 7 de %MW20 será transferido para %M15.



# Diagrama de Blocos Funcionais – FBD

## *Introdução*

Diagrama de Blocos Funcionais ‘FBDs’ consistem de um editor de gráfico livre que tem fluxo de dado orientado sendo indicada para aplicações de controle discreto ou contínuo.

As seções FBDs com uma grade de fundo com 36 Colunas e 24 linhas.

A linguagem de programação FBD não é orientada por célula, embora os objetos sejam alinhados com as coordenadas da grade.

A linguagem consiste do re-uso de blocos de função derivados “DBF” e elementares “EFB”, elementos de programas, Macros e Estruturas de dados.

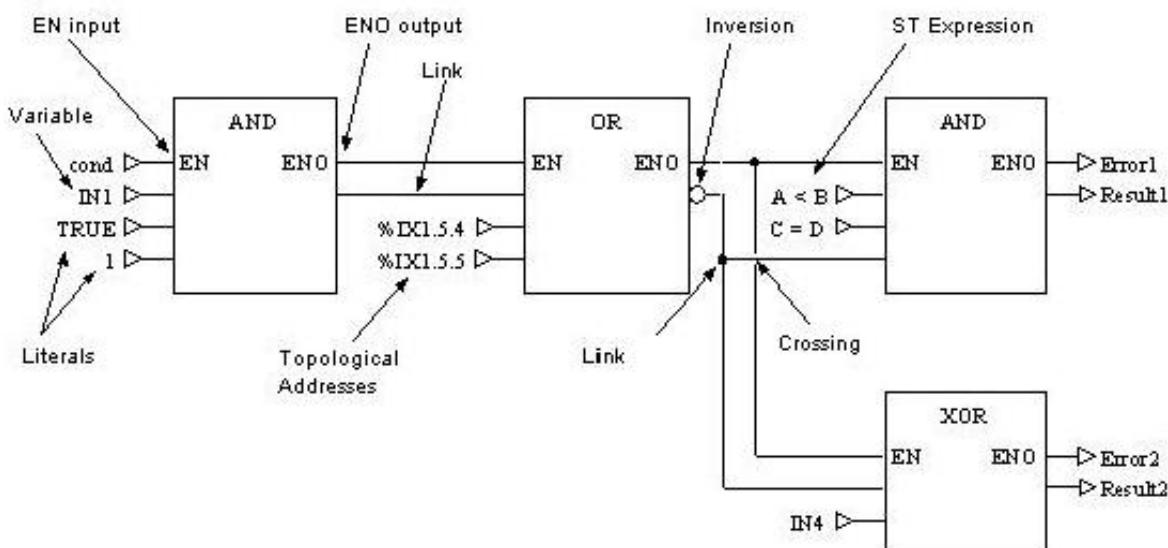
Os objetos da linguagem de programação FBD (Function Block Diagram) ajudam na divisão de uma seção em um número de:

- EFs e EFBs (Funções Elementares e Blocos de Funções Elementares)
- DFBs (Blocos de Função Derivados)
- Procedimentos e
- Elementos de controle.

Estes objetos combinados pelo nome de FFBs podem ser “linkados” entre si por “Links” ou por parâmetros.

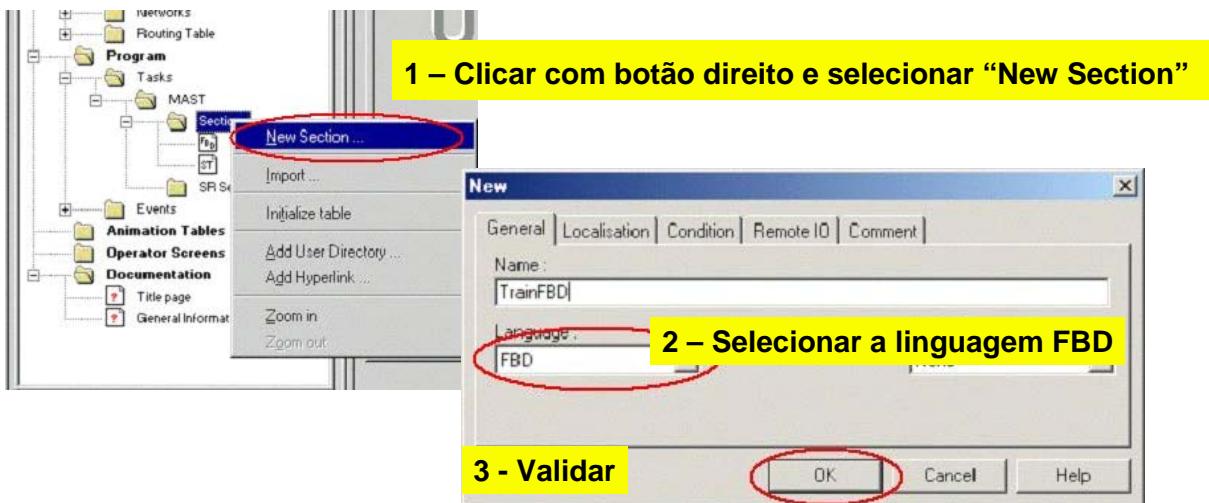
Pode-se ainda editar comentários na lógica da seção através do objeto de texto.

Representação de uma seção FBD.



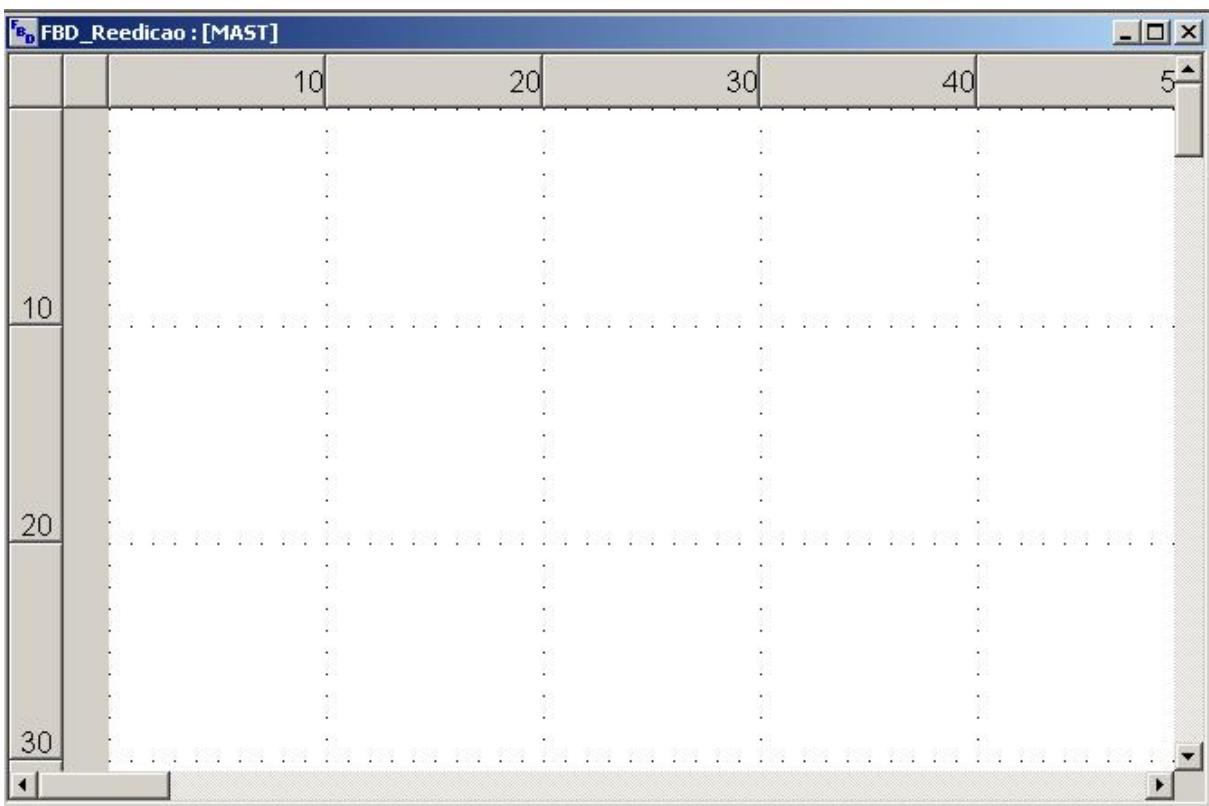
## *Criando uma Seção FBD*

A partir do Project Browser, expandir a pasta de tarefa “MAST” e na pasta “Section” clicar com o botão direito e selecionar “New Section” (1), selecionar o tipo de linguagem “FBD” (2) e validar acionando OK (3).



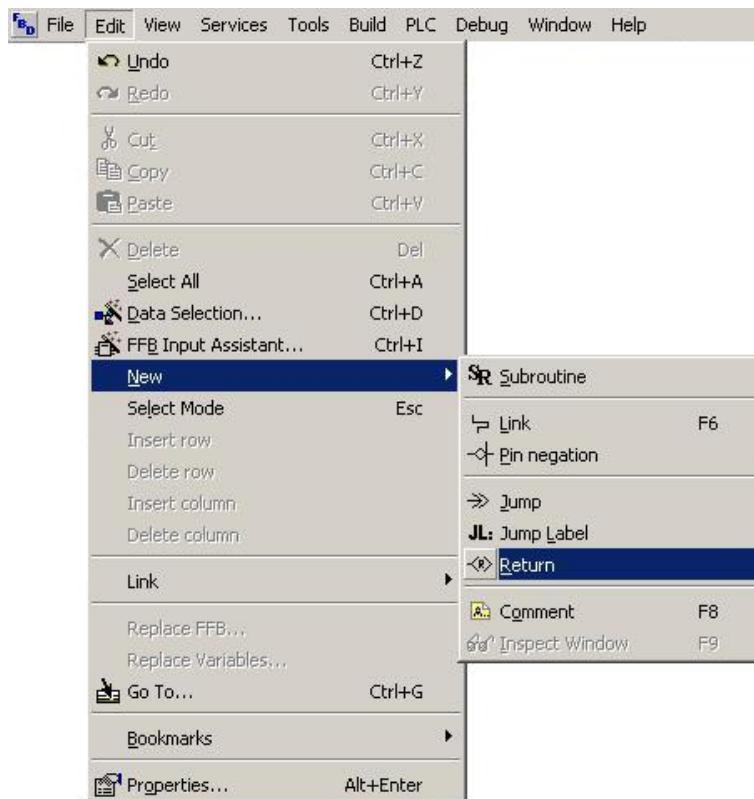
As demais configurações obedecem aos mesmos procedimentos vistos na linguagem Ladder em “**12.1.4.1 Configuração da seção**”.

Ao validar a nova seção clicando em OK (3), é aberta a área de edição FBD com 36 colunas e 24 linhas.

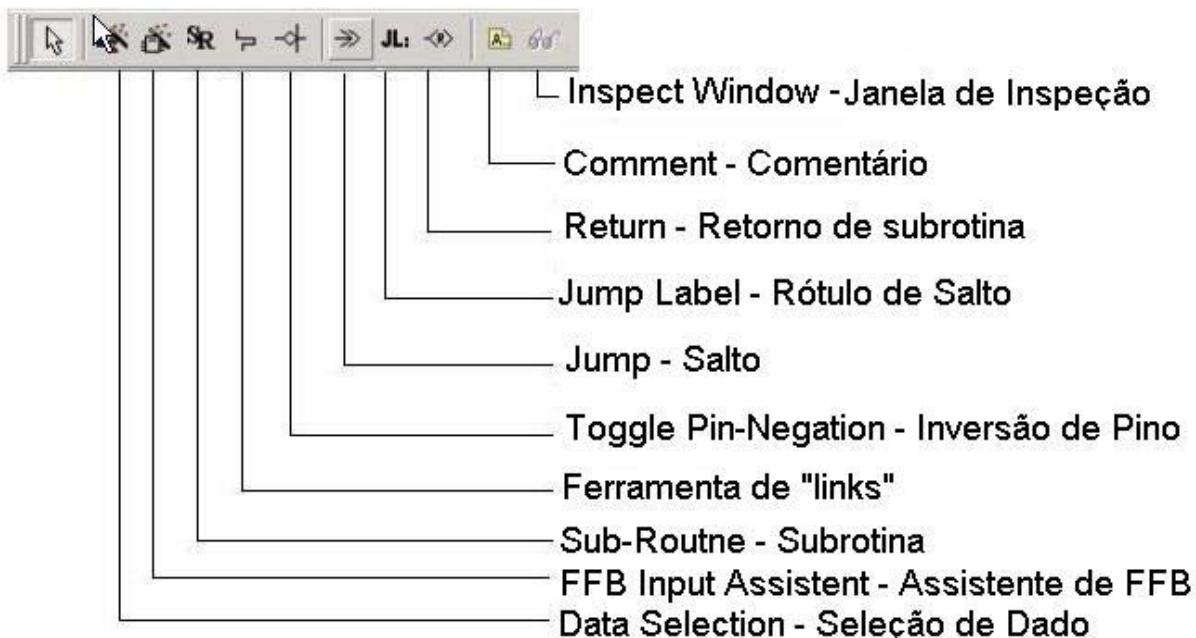


Existem três métodos para acessar os objetos de programação em FBD:

**1 =>** Na barra de ferramenta, acessar a régua “Edit” onde é possível selecionar “Data Selection..”, “FFB Input Assistant” ou “New”, que ser selecionado será disponibilizada a régua seguinte com os objetos para Subrotina, Link, Inversão de pino, Salto, rótulo de salto, retorno e comentário, os quais devem ser selecionados e ao posicionar o prompt na área de edição clicar no botão esquerdo do mouse.



2 => Na barra de ferramentas de edição de Objetos FBD.



3 => Clicando com o botão direito na área de edição, onde será disponibilizada a caixa com os objetos a serem selecionados.



### Regras de programação

Quanto às variáveis:

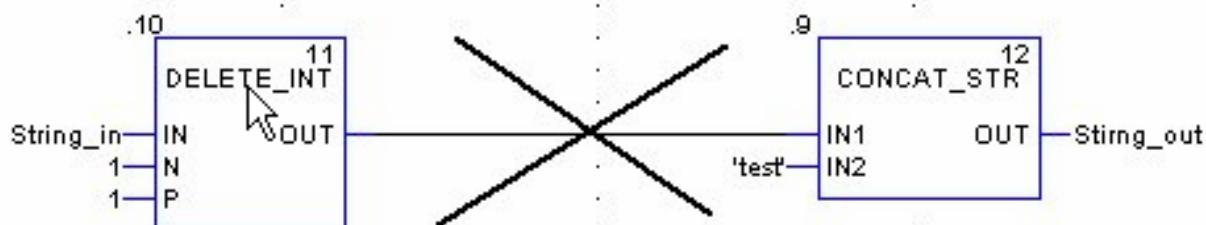
- Entrada diretamente, com o “data selector” ou pelo editor de dados.

Funções elementares, Blocos de Função Elementares, ou Bloco de Função Derivado:

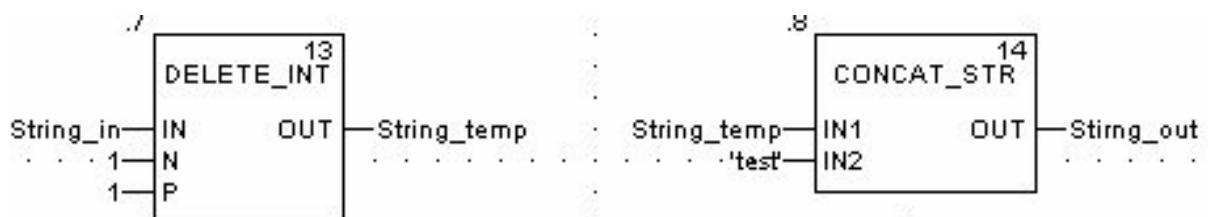
- Com o seletor de dado “data Selector” ou pelo “libset browser”.
- É possível ter o assistente de FFB “FFB input assistant”

“Lincando” com tipos de “string”:

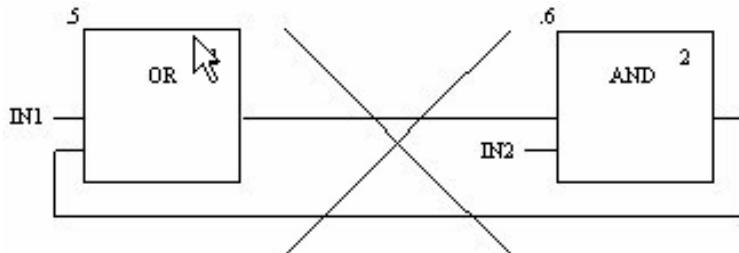
- A conexão de uma saída de FFB com a entrada de outro, não é permitido.



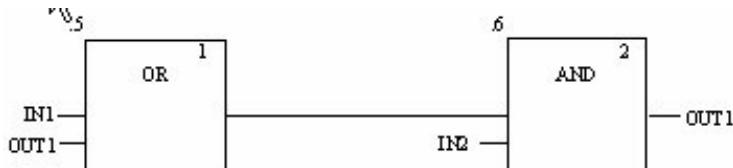
- Solucionar usando variáveis intermediárias



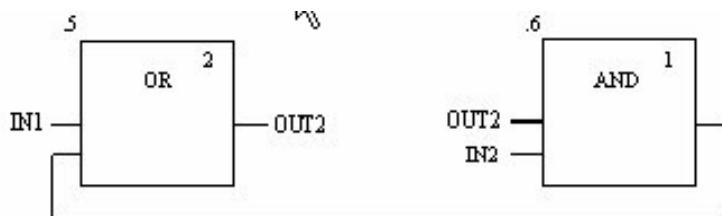
Não é permitido “Loops” via links:



Solucionar usando parâmetros, por exemplo, “OUT1”



Solucionar usando parâmetros, por exemplo, “OUT2”



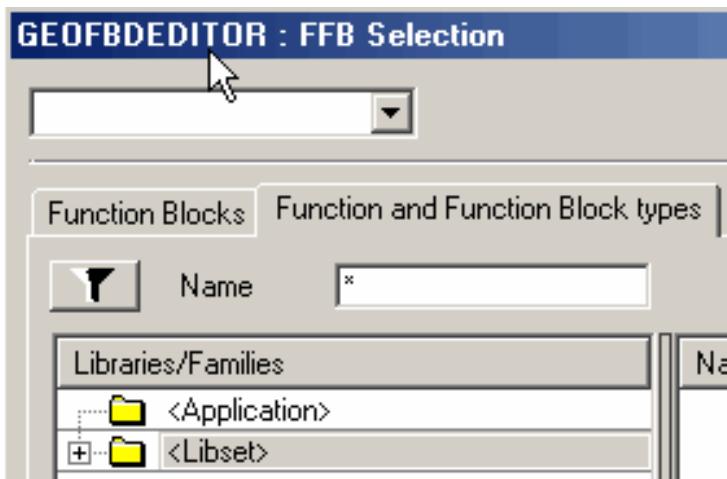
### *Inserção de Bloco de Função a partir da barra de ferramentas*

Para inserir um Bloco de Função na seção FBD a partir da barra de ferramentas, seguir os seguintes passos:

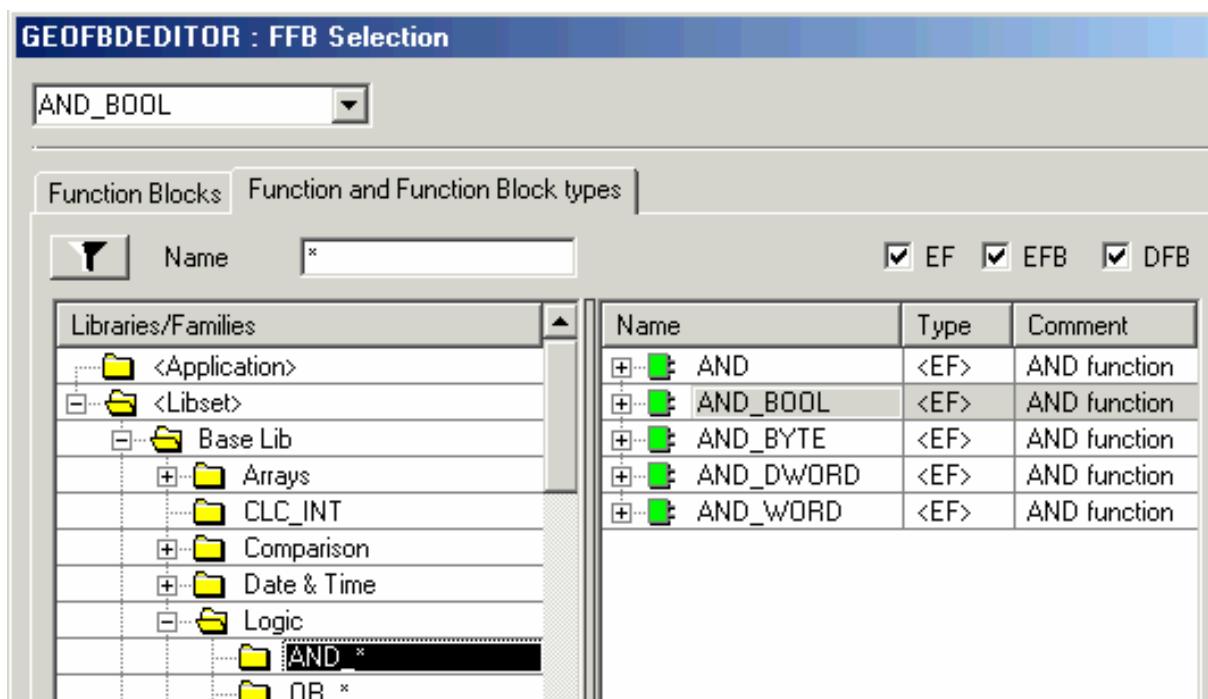
1 – Clicar no ícone FFB na barra de ferramentas,



2 – Selecionar a aba “Function and Function Blocks”,



3 – Em Libset, acessar: Base Library, Logic e selecionar o bloco desejado e clicar OK,

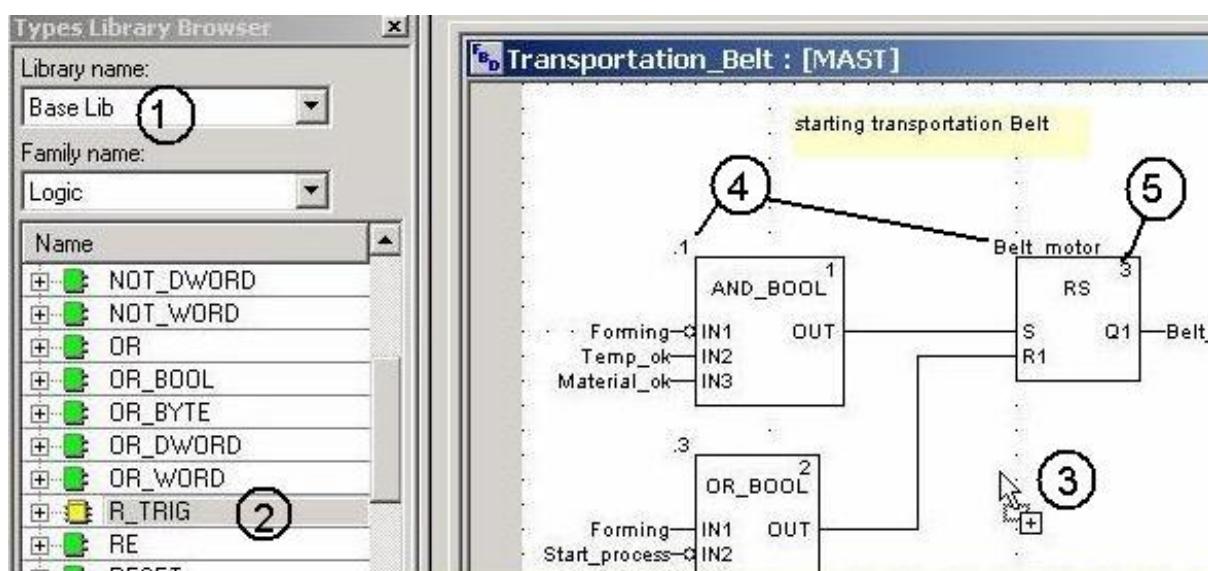


4 – Posicionar o prompt na posição desejada da área de edição,

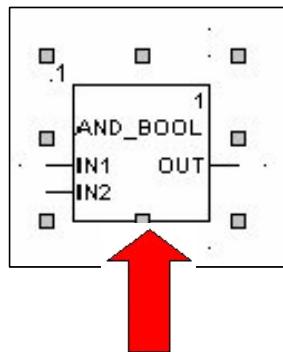
5 – Selecionar as variáveis para os pinos (Entradas/Saídas).

### *Inserção de FFBs a partir do “Types Library Browser”*

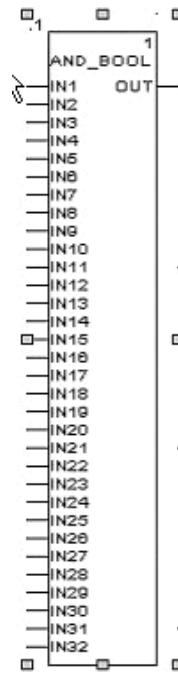
Para inserir um Bloco de Função diretamente através de cópia e cola a partir da biblioteca, em “Types Library Browser” (1) (disponível em “Tool”) deve-se selecionar a família, selecionar o bloco de função (2) e colar na área de edição (3). O bloco traz a indicação do número e seqüência de inserção do mesmo na lógica ou nome da instância no canto superior esquerdo (4) e a seqüência de execução no canto superior direito (5).



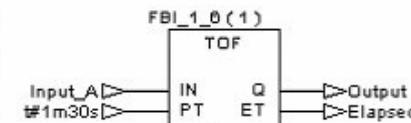
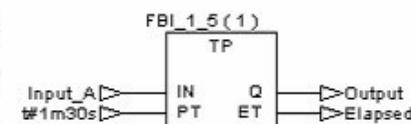
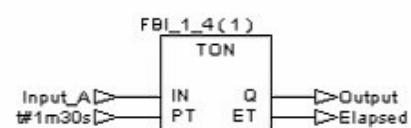
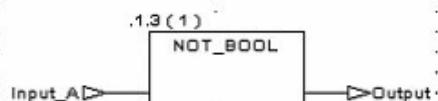
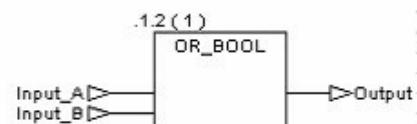
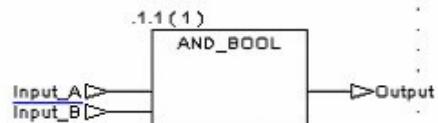
A quantidade de pinos de um bloco pode ser aumentada (até 32) da mesma maneira que no diagrama Ladder, ou seja, clicando no ponto na parte de baixo do bloco e arrastando até completar a quantidade de pinos desejados.



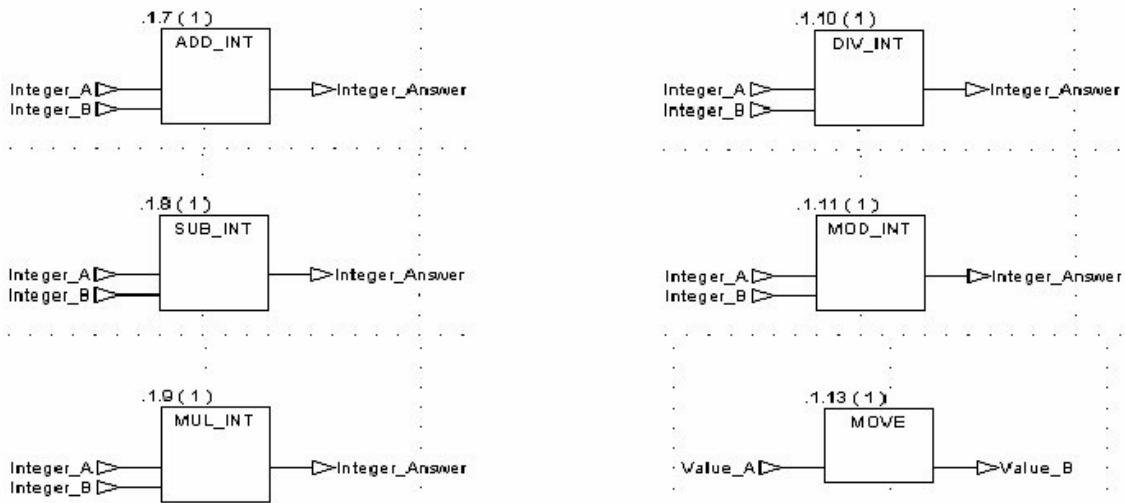
até 32 pinos!



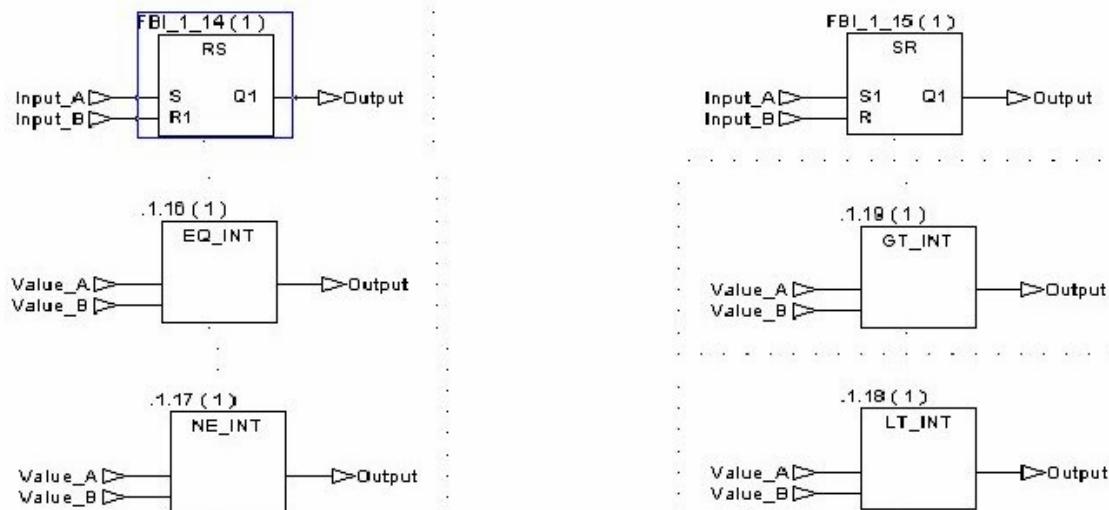
### Blocos Lógicos Básicos e Temporizadores:



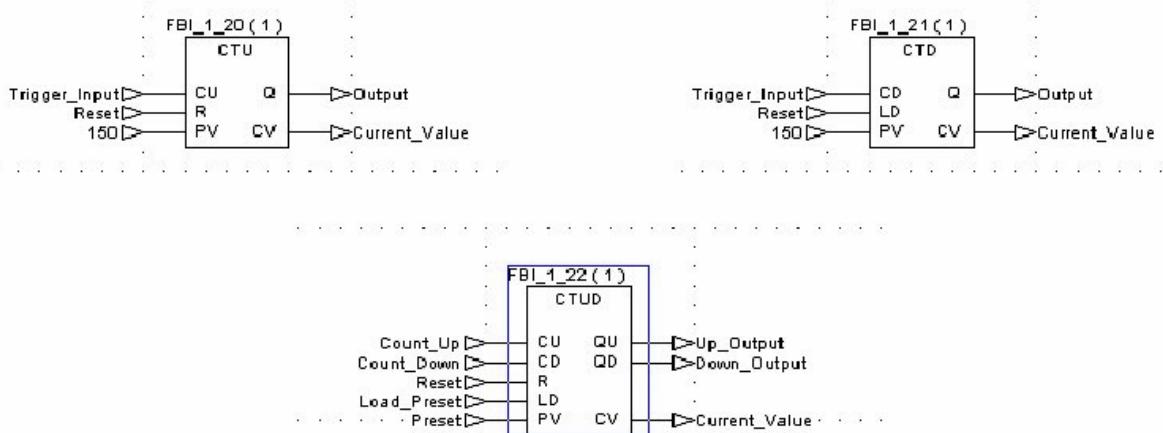
### Blocos de Funções Matemáticas básicas:



### Blocos básicos Set/Reset e Comparadores:



### Bloco de Função Contadores:



Observar que os contadores IEC não param a contagem quando alcançam sua contagem final ou o valor de preset, pois continuam acumulando a contagem.

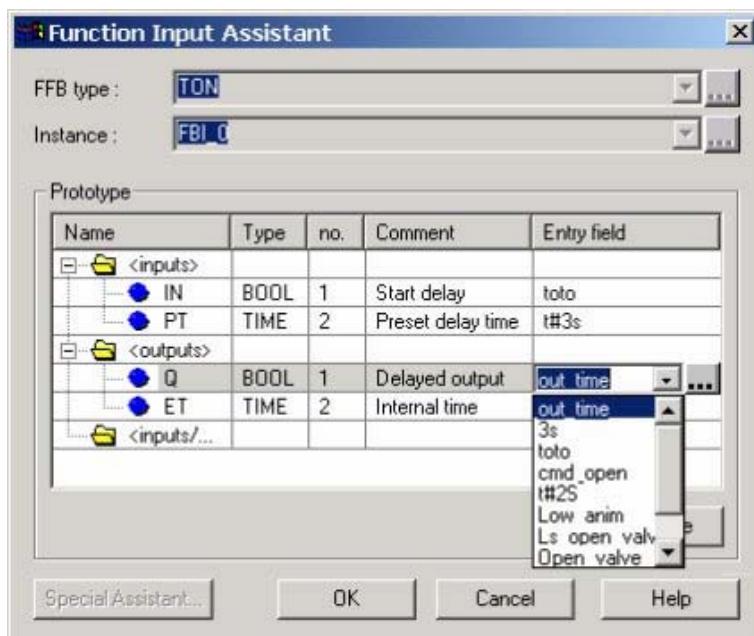
## Inserção de Bloco de Função a partir do “FFB input assistant”

A inserção de Bloco de Função a partir do “FFB input assistant” é acessada a partir do ícone “FFB Input assistant” na barra de ferramentas.



O FFB deve ser selecionado por tipo ou por instâncias, sendo que as instâncias são criadas automaticamente se a seleção do FFB for por tipo.

Na caixa de diálogo, “Function Input Assistant” no campo “Entry”, selecionar a variável para cada entrada e saída, para validar acionar o botão OK.



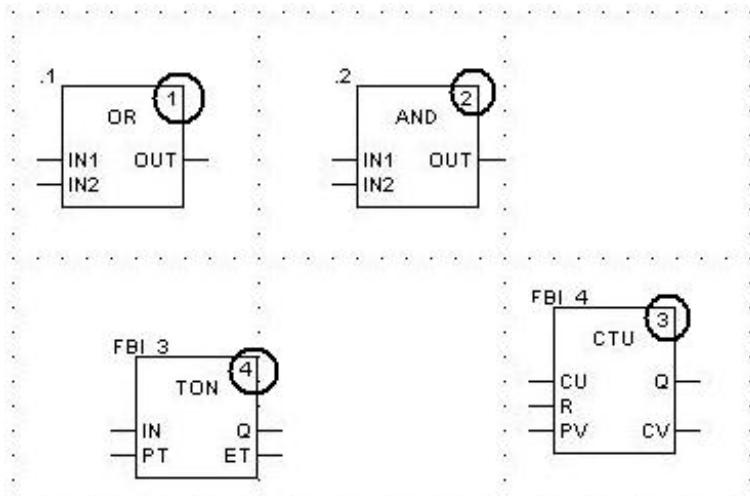
Após acionar o botão OK, posicionar o prompt na posição desejada da área de edição e clicar no botão direito do mouse, finalizando a edição.

## Seqüência de execução de objetos em FBD

A seqüência de execução de objetos em uma seção FBD é da esquerda para a direita e de cima para baixo, sendo que a seqüência é determinada pela posição do FFB na seção FBD.

Se os FFBs forem “lincados” graficamente, a seqüência é determinada pelo fluxo de sinal.

A seqüência de execução é indicada pelo número de execução atribuído automaticamente ao bloco (número no canto superior direito da moldura do FFB).



### *Seqüência de execução em uma rede lógica “Networks”*

A Seqüência de execução em uma rede lógica “Networks”, que é semelhante ao “Rung” na programação Ladder obedece as seguintes regras:

- A execução de uma seção é completada por uma “network” baseada nos links dos FFBs de cima para baixo.
- Links não podem ser usados para criar “loops”, pois a seqüência de execução não é muito clara, os “loops” devem ser criados usando parâmetros atuais (variáveis intermediárias)
- A seqüência de execução de “networks” que não são “lincadas” é determinada pela seqüência gráfica (do topo direito para a base esquerda).
- O processamento de uma “Network” é terminada completamente antes de iniciar outra “network” para a qual as saídas são usadas na “network” anterior.
- Nenhum elemento de uma “network” é considerado enquanto o estado de todas as entradas destes elementos não for calculado.
- O processamento de uma “network” é somente terminado se todas as saídas desta “network” forem processadas.

#### **Fluxo de sinal dentro de uma “Network”.**

A seqüência de execução dentro de uma “network” obedece as seguintes regras:

Um FFB é somente processado se todos os elementos (saídas FFB, etc.) com as quais suas entradas estão “lincadas” sejam processadas.

A seqüência de execução de FFBs que são “lincadas” com várias saídas do mesmo FFB funciona de cima para baixo.

A seqüência de FFBs não é influenciada pela localização do mesmo na “Network”, isto não se aplica se mais de um FFB são “lincados” para a mesma saída do FFB chamado. Neste caso, a seqüência de execução é determinada pela seqüência gráfica (de cima para baixo)

#### **Prioridades**

Prioridades na definição do fluxo de sinal dentro de uma seção:

Prioridade	Regra	Descrição
1	Link	Links possuem a mais alta prioridade na definição do sinal
2	Definição pelo usuário	Acesso do usuário para a seqüência de execução
3	“Network” por “Network”	Processamento em uma “network” é completamente terminada antes do processamento iniciar em outra “network”.
4	Seqüência de saída	FFBs que são “lincados” para saídas de mesma chamada FFB são processados de cima para baixo.

5	Rung por Rung	Mais baixa prioridade. (Somente aplicada se nenhuma das outras regras for aplicadas)
---	---------------	--

### Mudança da seqüência de execução.

A ordem de execução de uma “Network” e a ordem de execução de objetos dentro de uma “Network” é definida por regras. Em alguns casos, a ordem de execução sugerida pelo sistema pode ser alterada.

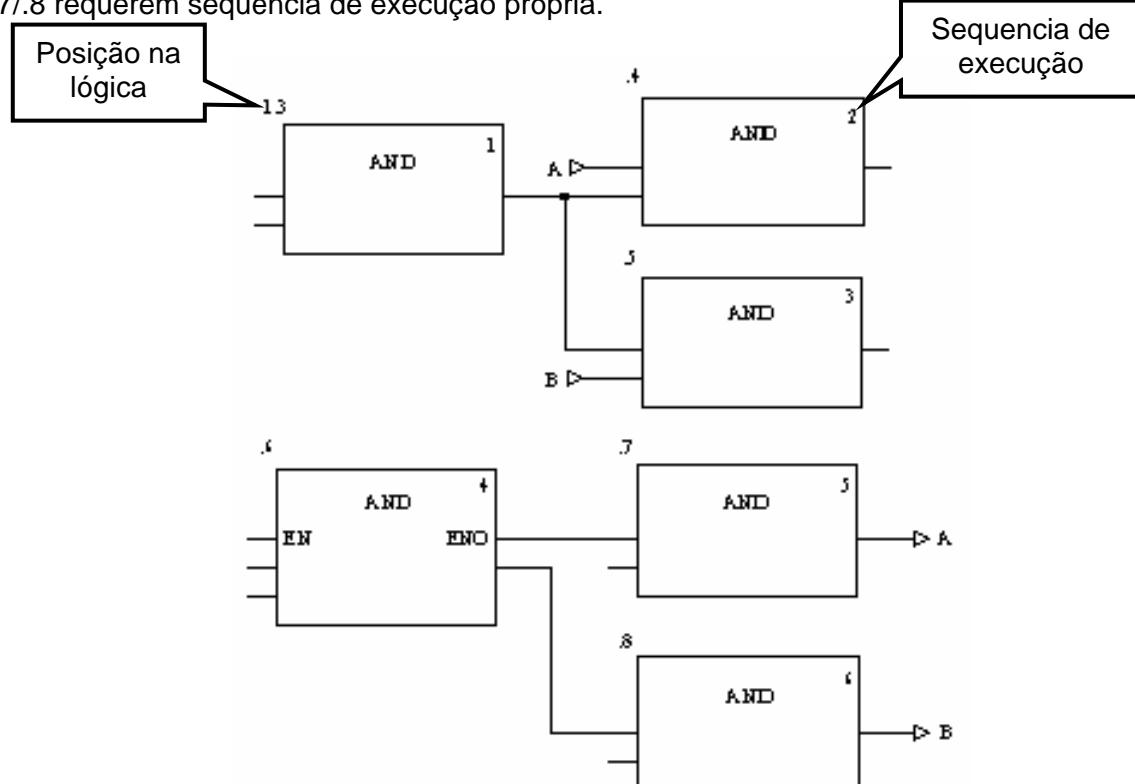
#### Passos para definição/alteração da seqüência de execução:

- Utilizar links ao invés de parâmetros atuais (Variáveis intermediárias)
- Alterar a posição da “Network”
- Definir explicitamente a seqüência de execução
- Alterar a posição dos FFBs.

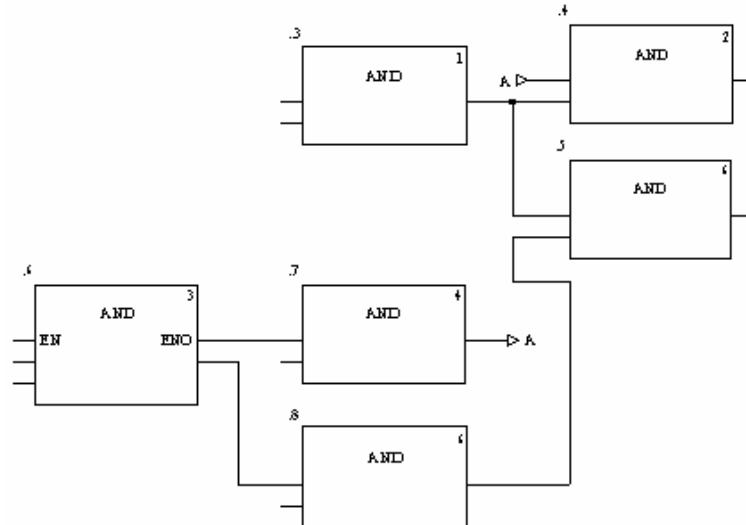
#### Procedimento para definição/alteração da seqüência de execução de objetos dentro de uma “Network”

#### Situação Original:

A figura mostra duas “networks” para as quais a seqüência de execução são definidas somente por sua posição dentro da seção, sem levar em consideração que os blocos .4/.5 e .7/.8 requerem seqüência de execução própria.

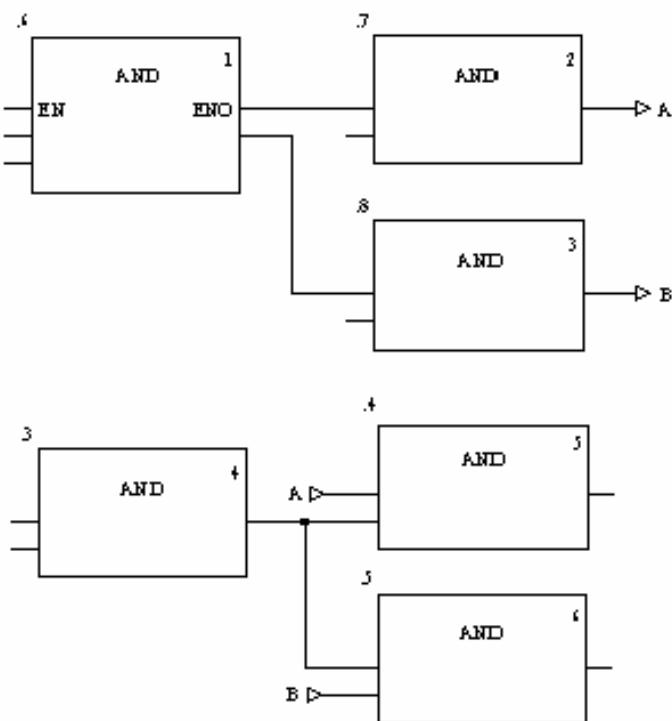


Solução utilizando link ao invés de parâmetro atual (variáveis) as duas “networks” funcionam na seqüência própria.



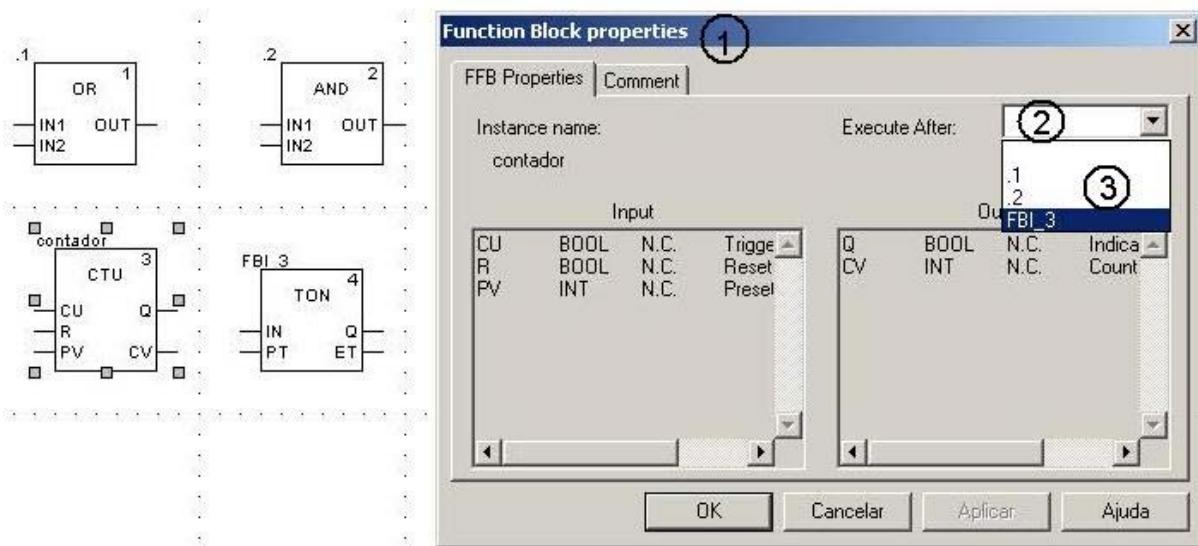
### Reposicionando a “Network”

Pelo uso de um link ao invés de variáveis as duas “networks” funcionam na seqüência própria.

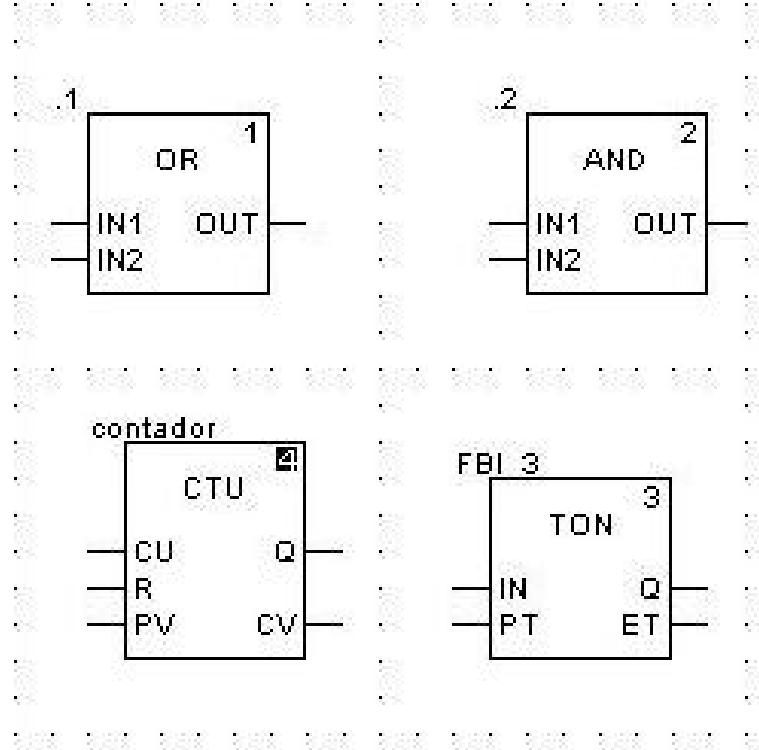


### Definição explícita da seqüência de execução

A Definição explícita da seqüência de execução é feita a partir da propriedade do FFB, que deve ser acessado através de clique com botão direito do mouse sobre o mesmo e selecionar “Properties”, que será aberta a caixa de diálogo “Function block properties” (1), no campo “Execute after” (2) selecionar qual bloco será executado antes do mesmo (3), para validar acionar o botão OK.



Observe que após validar a alteração para execução após o bloco “FBI\_3”, a indicação de seqüência de execução do bloco é alterada assim como a do bloco seguinte.



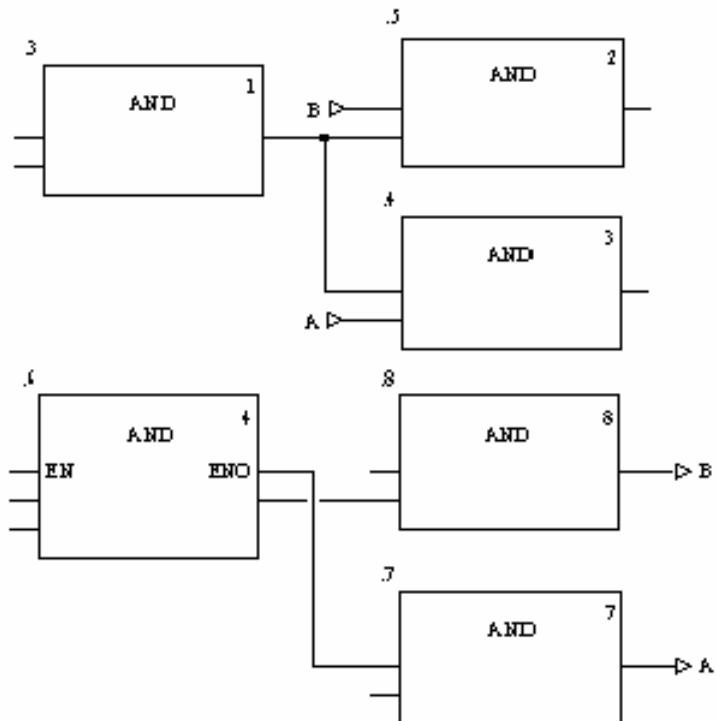
### Alterando a posição do FFB

A posição do FFB somente influencia na seqüência de execução se mais de um FFB está “lincado” para a mesma saída da chamada de FFB.

Na primeira “network”, os blocos .4 e .5 são chaveados. Neste caso (origem comum para as entradas de ambos os blocos) a seqüência de execução

In the first network, block positions .4 and .5 are switched. In this case (common origins for both block inputs) the execution sequence of both blocks is normally keyed (processed from top to bottom).

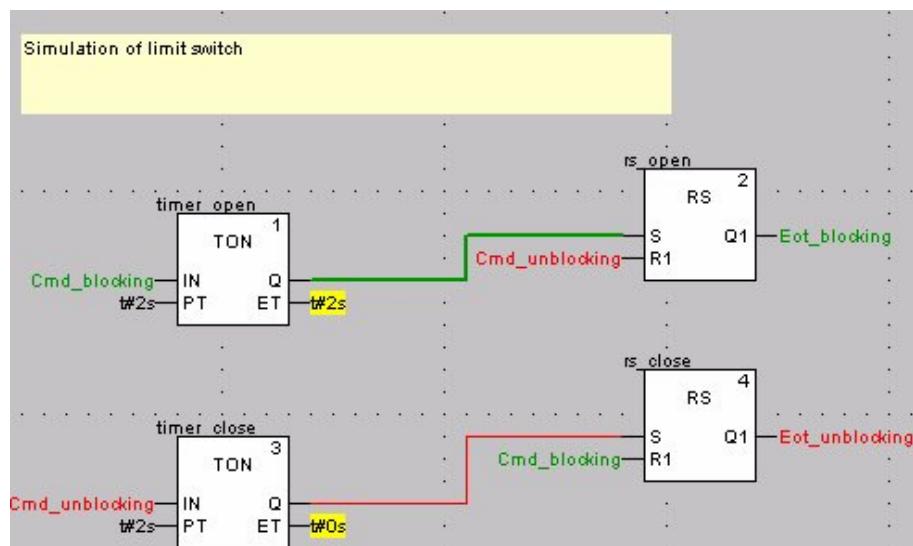
Na segunda “network”, a posição dos blocos .7 e .8 são chaveadas. Neste caso (origem diferente para as entradas dos blocos) a seqüência de execução dos blocos não é chaveada (processada na seqüência que as saídas dos blocos chamam a entrada).



## Depuração - Debugging

A depuração no Unity Pro obedece a uma seqüência de cores que são ativas no modo online animando a seção FBD:

- Variáveis Booleanas verdadeiras: Verde Green
- Variáveis Booleanas Falsas: Vermelho Red
- Outros tipos de variáveis: Amarelo Yellow
- Link animado no caso de tipos booleanos (vermelho ou verde).
- Valores numéricos e string são visualizados diretamente.





# **CAPÍTULO 10**

## **Teste de Aplicação**

---

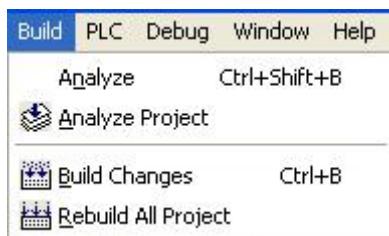


# Teste da Aplicação

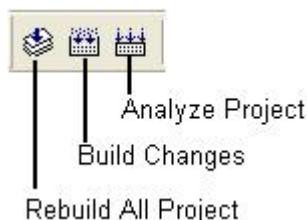
## Análise e Compilação do projeto.

O Unity Pro possui o recurso de teste e verificação do funcionamento do projeto antes mesmo de ser instalada no campo, isto reduz o tempo de “start-up” e ajustes do projeto na colocação do processo em funcionamento.

O processo de teste inicia-se com a análise e compilação do projeto, o acesso a estes recursos é feito através da barra de ferramentas na aba “Build”, que ao ser selecionado disponibiliza as opções de análise e de compilação “Build”.



O teste pode ser acessado também diretamente na barra de ferramenta nos ícones:



**Análise - Analyse:** verifica que na seção corrente existe um programa válido.

Erros de programas tais como: símbolos indefinidos, elementos desconectados etc. serão mostrados na janela “Output Window” da tela.



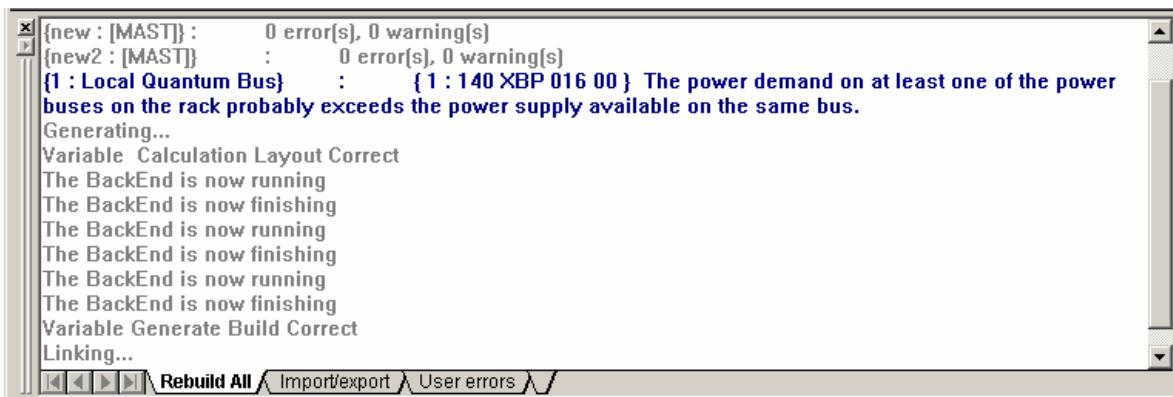
**Análise do Projeto - Analyze project :** varre as entradas do projeto para analisar erros.

**Compilação do Projeto - Build Project**

Cria o arquivo que pode ser baixado para o CLP ou simulador do CLP.

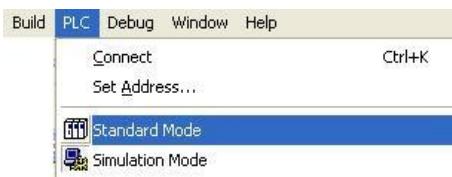
**Recompilação de todo o projeto - Rebuild all Project**

Leva em consideração todas as alterações feitas em um projeto existente.

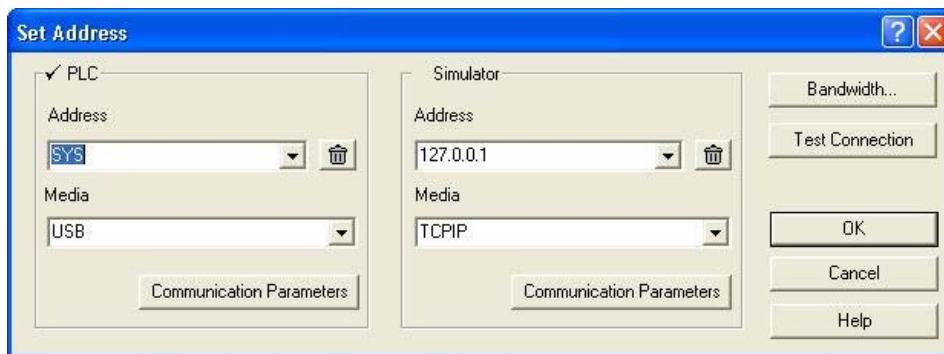


## Conectando ao CLP

Na barra de ferramentas através da aba PLC é possível acessar as opções de conexão com o **CLP** “Standard mode” ou com o **Simulador** “Simulation Mode” e ajuste do endereço.



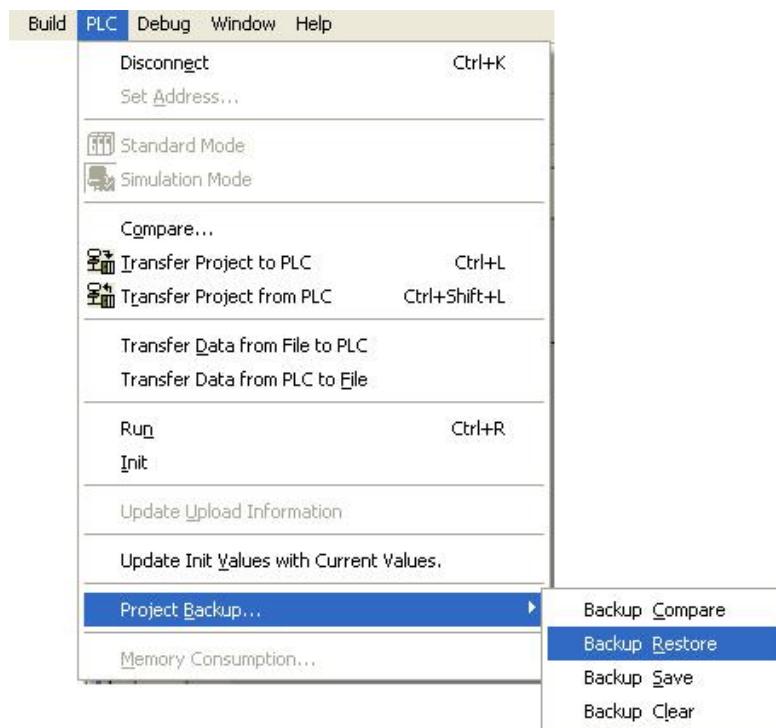
Deve-se selecionar “Standard Mode” e definir o endereço do CLP, clicando na opção “Set Address..” será aberta a caixa de diálogo “Set Address”.



A aba de “Communications Parameters” permite o acesso direto ao “Driver Manager” utilizando o ícone “Driver Settings”, a pós os ajustes acionar “Connect” na aba PLC, colocando o CLP “online”.

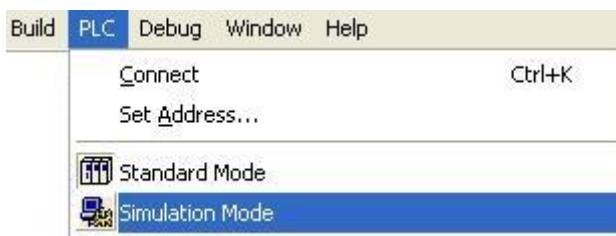
No modo conectado “online” pode se fazer as seguintes operações:

- Comparação entre o projeto no PC e o projeto no CLP,
- Transferência de programa para o CLP ou do CLP,
- Funções de backup do projeto tais como Backup, Comparação, Salvar, restaurar ou limpar,
- Comando de Run/Stop do CLP ou inicializar,
- Tela de diálogo para verificação de memória utilizada.

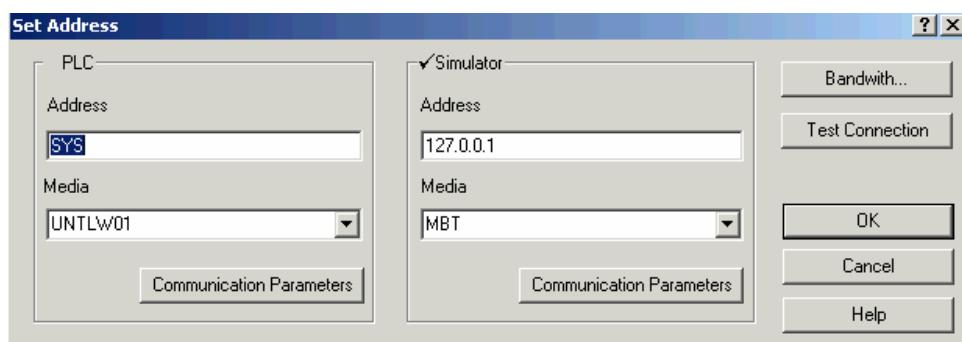


## Conectando ao Simulador.

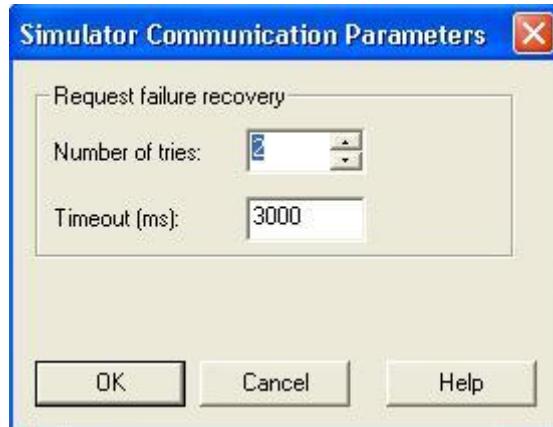
Na barra de ferramentas através da aba PLC acessar a opção de conexão com o Simulador “Simulation Mode”.



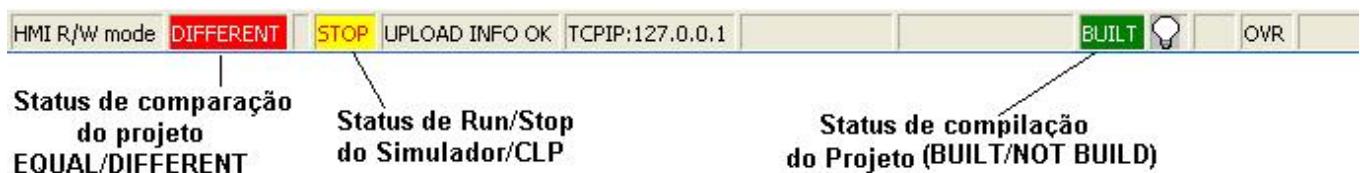
Deve-se selecionar “Standard Mode” e definir o endereço do CLP, clicando na opção “Set Address..” será aberta a caixa de diálogo “Set Address”.



A aba de “Communications Parameters” permite o acesso a configuração da comunicação com o simulador através da caixa de diálogo “Simulator Communication Parameters”, para finalizar clicar o botão **OK** e **OK**.



=>**Obs.** Para qualquer modo de conexão CLP ou Simulador, no rodapé da tela mostra informações da conexão, como pode ser visto na figura seguinte.



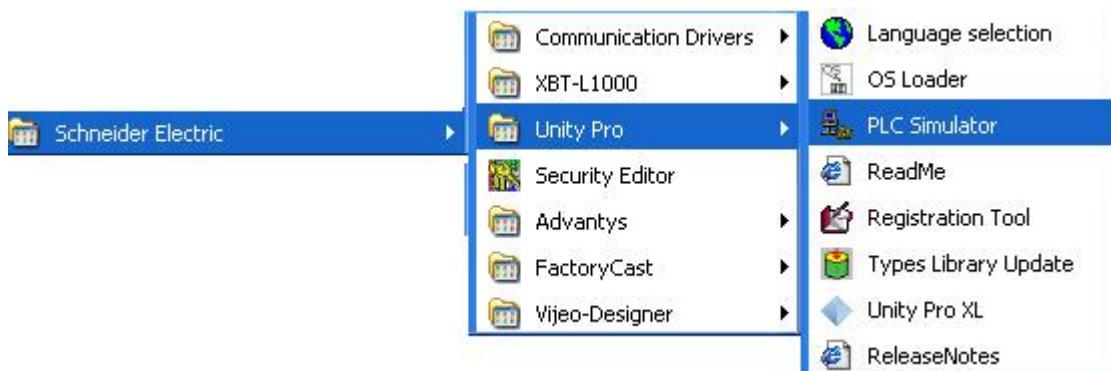
### Simulador Unity.

O Simulador Unity pode ser usado para:

- Simular a operação de um CLP Quantum Premium ou M340
- Procura de gravação de erros no programa

### Iniciando o simulador:

O simulador é acessado através da aba Programas => Schneider Electric => Unity Pro => PLC Simulator.



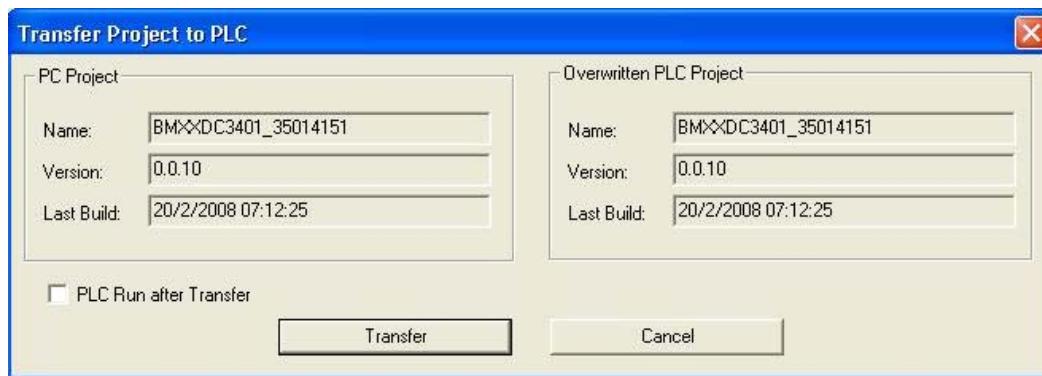
=>**Obs.** O simulador pode ser iniciado com o Unity pro funcionando.

### Carregando uma aplicação no simulador:

Na barra de ferramenta do Unity Pro, selecionar na “Connect” na aba PLC, pode-se acessar

ainda pelo ícone da barra de ferramentas, a transferência do projeto é feito pelo ícone

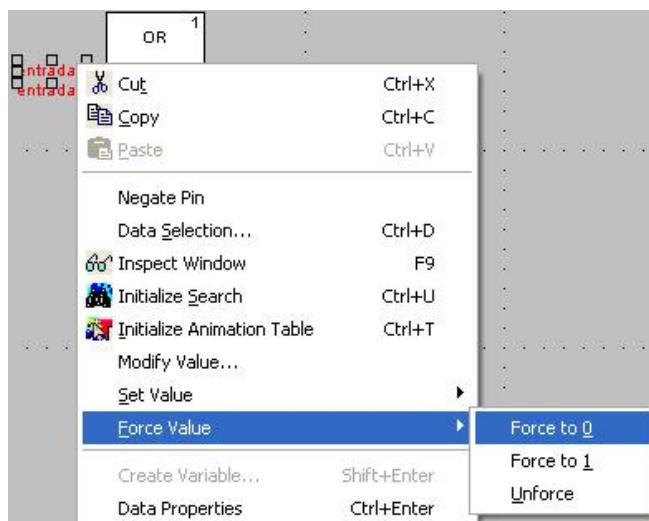
que ao ser acionado abre a caixa de diálogo “Transfer Project to PLC”, caso o usuário deseje que ao concluir a transferência o CLP entre no modo “Run”, deve-se marcar o campo “PLC” Run after Transfer e clicar no botão “transfer” para realizar a transferência.



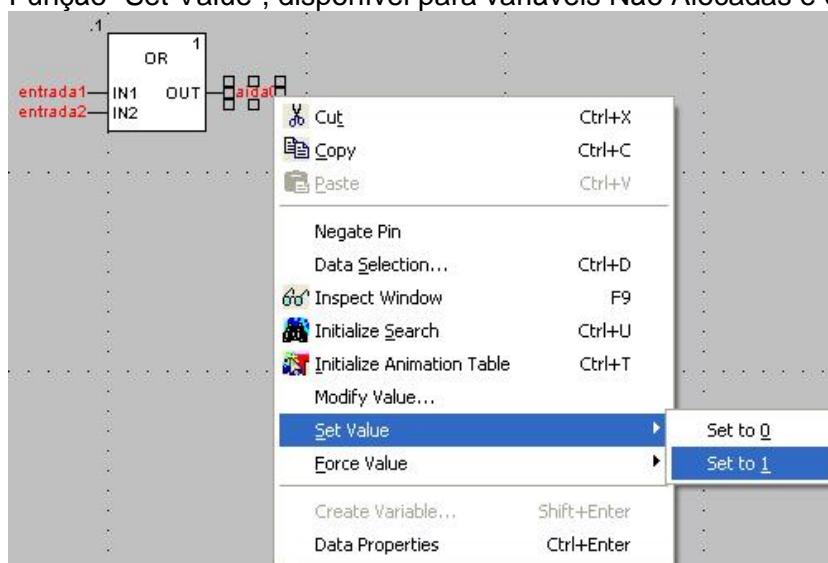
### *Simulação do funcionamento do projeto*

Concluída a transferência, acionar o ícone “Animation” na barra de ferramentas para que seja possível utilizar as funções “Force” e de modificação do valor de uma variável. A animação pode ser acessada também a partir da barra de ferramentas através da aba “Services” e selecionar “Animation”. Com a animação ativa clicar com o botão direito do mouse sobre um determinado objeto ou variável e selecionar a ação desejada.

Função “Force”, disponível para variáveis Alocadas e do Tipo Ebool:



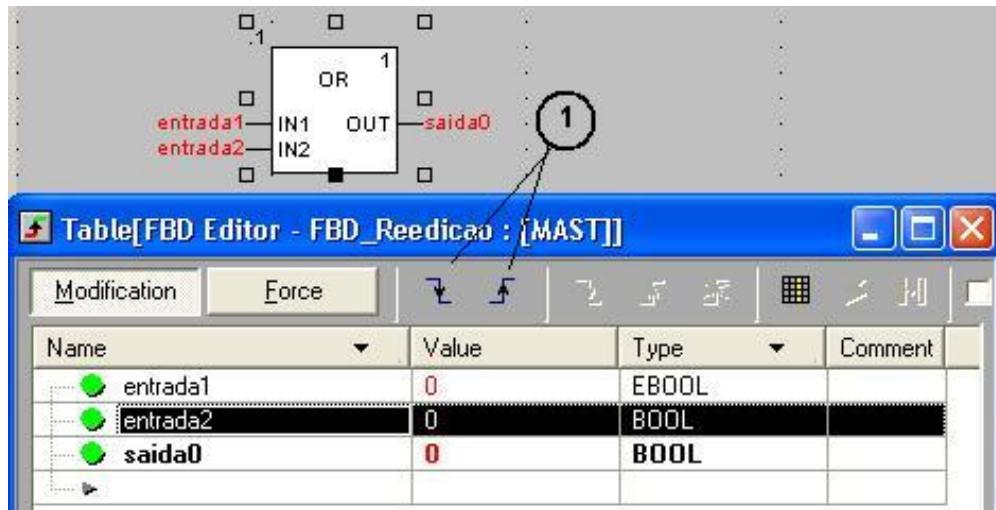
Função “Set Value”, disponível para variáveis Não Alocadas e do tipo Bool.



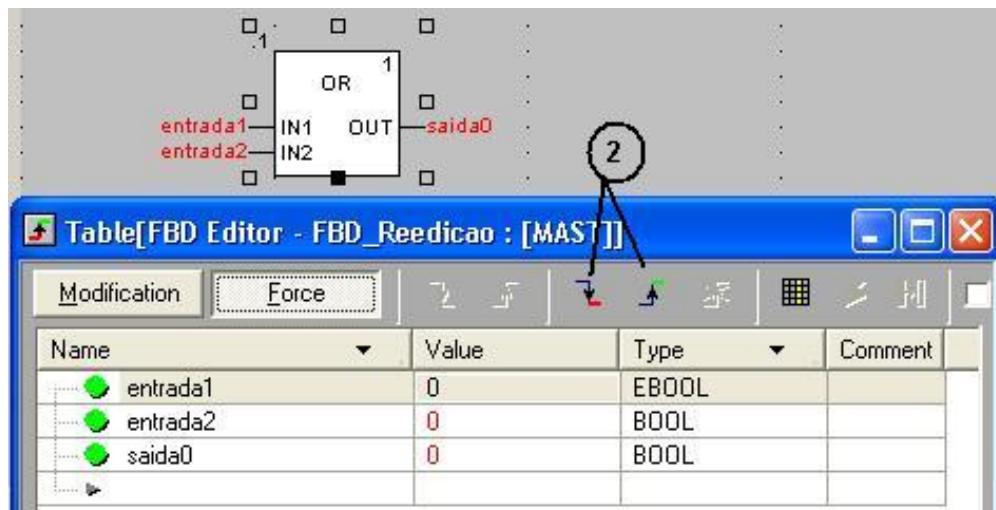
É possível ainda testar o projeto utilizando o recurso de tabela animada do Unity Pro, a qual é acessada através da barra de ferramentas através da aba “Services” e selecionar “Initialize Animation Table” ou clicar no botão direito do mouse e selecionar “Iniatialize Animmation Tables”

Com o projeto no modo Run, selecionar um objeto ou uma variável e acessar a tabela animada, que ao ser selecionado “Initialize Animation Table” é aberta a caixa de diálogo com as opções de Force ou Modificação de valores.

Para variável tipo BOOL é possível modificar valores para Zero ou para Um (1)



Para variável tipo BOOL é possível Forçar valores para Zero ou para Um (2)



### *Simulação de entradas digitais e analógicas.*

A simulação de entradas digitais e analógicas é possível com a utilização dos seguintes blocos:

- WRITE\_INPUT\_DINT: Escreve entradas do tipo DINT
- WRITE\_INPUT\_EBOOL: Escreve entradas do tipo EBOOL
- WRITE\_INPUT\_INT: Escreve entradas do tipo INT
- WRITE\_INPUT\_REAL: Escreve entradas do tipo REAL

## Teste de Aplicação

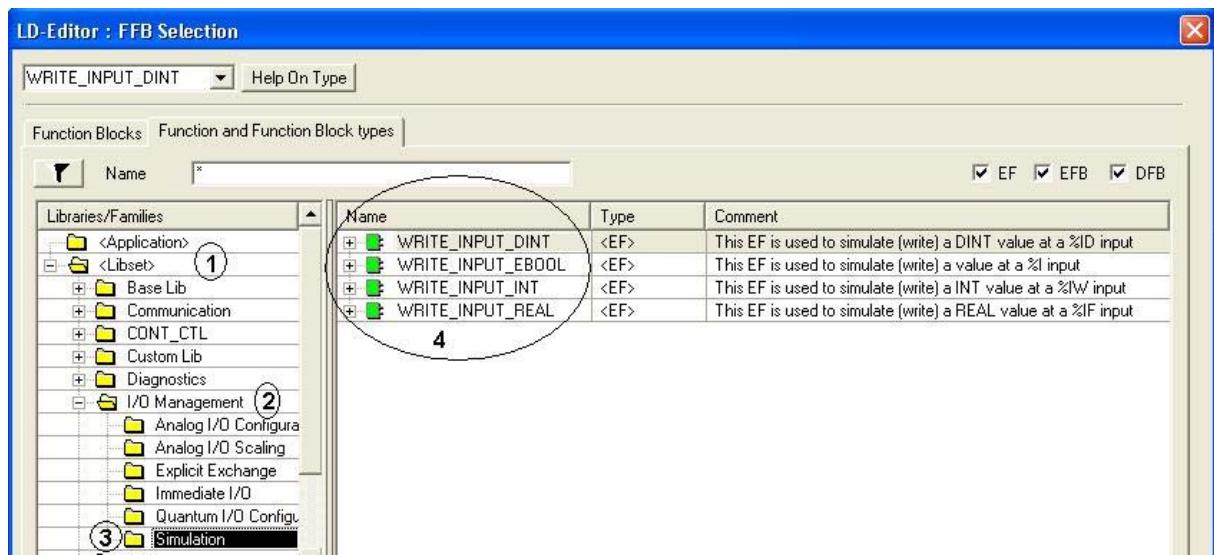
`WRITE_INPUT_INT` é usada para similar (escrever) um valor em uma entrada `%IW`. A entrada é escrita diretamente quando `WRITE_INPUT_INT` é evocada.

O bloco de função pode ser usado com um CLP real ou com simulador de CLP.



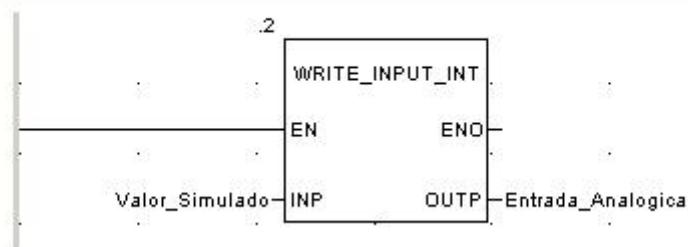
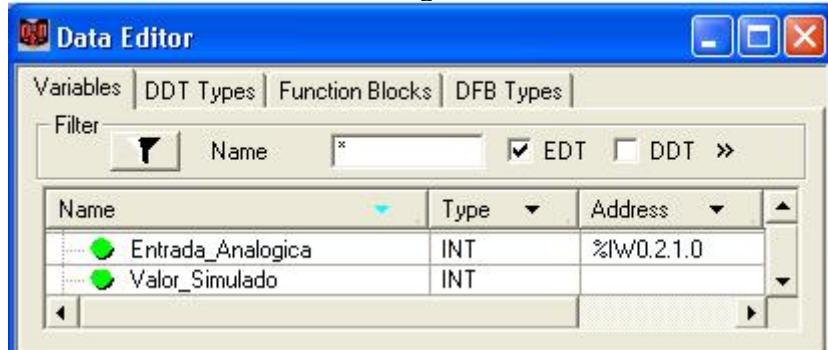
O bloco de função permite acesso a somente uma variável por vez.

O Acesso a estes blocos é feito a partir da biblioteca Liset (1), a biblioteca IOManagement (2), biblioteca Simulation (3), que expandindo a mesma tem-se acesso aos blocos de escrita nas entradas digitais e analógicas (4).



### Exemplo de aplicação.

Escrita de um valor simulado na entrada analógica `%IW0.2.1.0`.



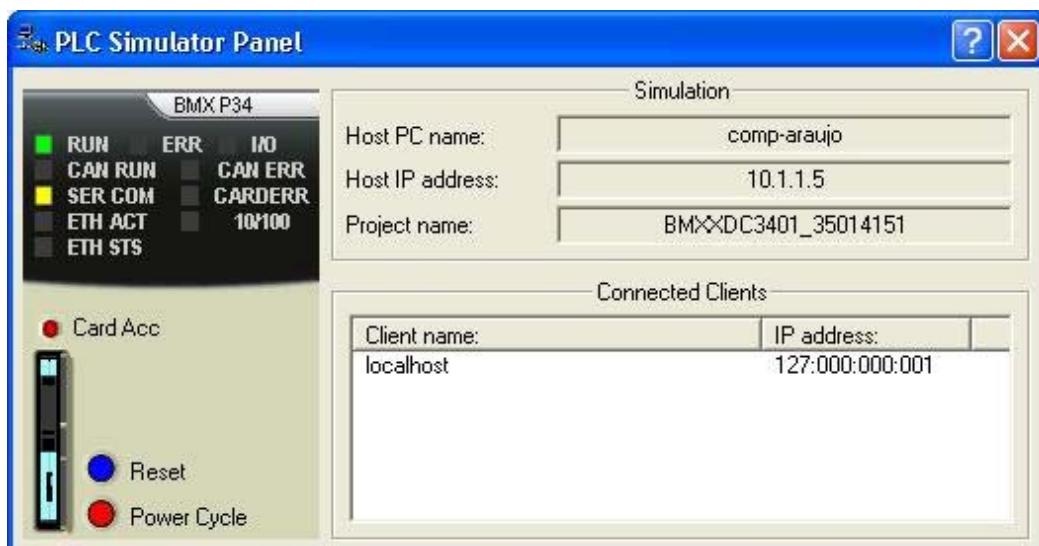
### Painel de Controle do Simulador.

O Painel de Controle de Simulador é acessado através de clique no botão direito do mouse no ícone do simulador na roda pé da tela do PC (1) e selecionar “Simulator Panel..” (2) e será visualizado o painel de controle do simulador.

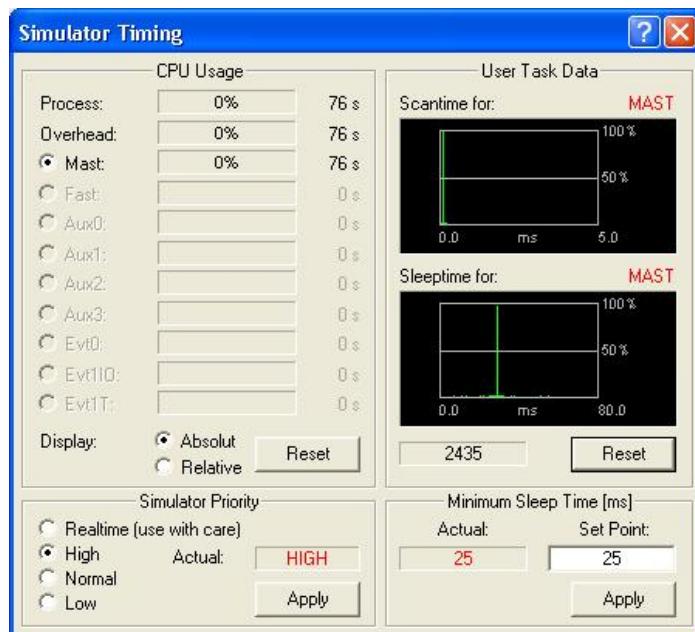


Quando o projeto é transferido para o simulador o painel do simulador mostra uma imagem da configuração do CLP. Esta imagem mostra:

- O status do CLP, por exemplo, Run, Err, I/O (Error) etc..
- O nome do PC Host e seu endereço IP (IP gerado pelo sistema para o Simulador)
- O nome do projeto que está funcionando no simulador
- Os nomes e endereços IP dos clientes que estão conectados.



O tempo de simulação também é possível verificar, para isto ao clicar com o botão direito do mouse sobre o ícone do simulador no roda pé da tela do PC, selecionar “Timing” e será aberta a caixa de diálogo “Simulator Timing”, onde se tem informações do processo de simulação.





# **CAPÍTULO 11**

## **Funções de Diagnósticos**

---



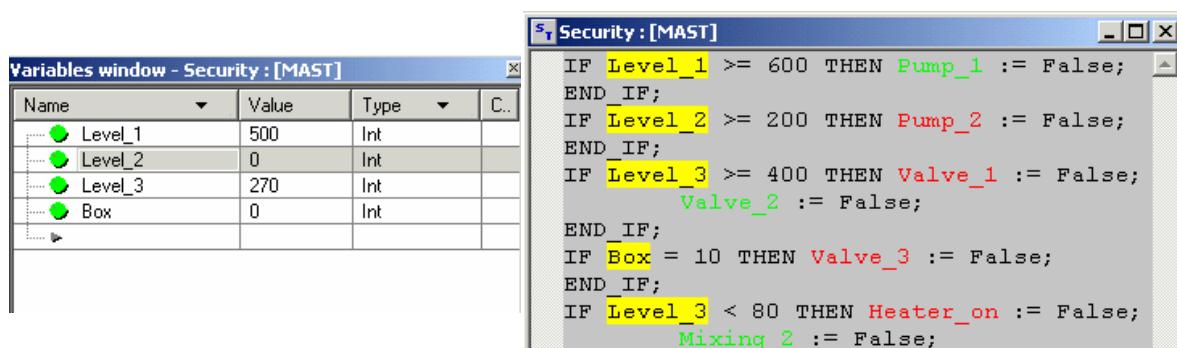
# Funções de Diagnóstico

## Introdução

Para reduzir o tempo de depuração e comissionamento o Unity Pro fornece os recursos de:  
 Animação dinâmica do Programa;  
 Inserção de ponto de parada “breakpoint” do programa;  
 Execução do programa passo a passo;  
 Fornecer entradas e saídas usando tabela animada ou por integrando tela de operador.

## Animação Dinâmica

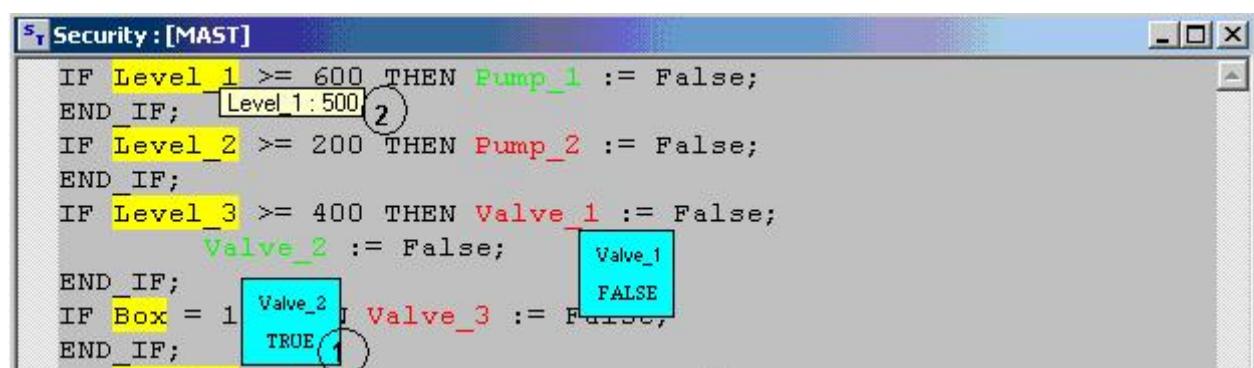
Quando no modo online, as variáveis analógicas e booleanas são animadas diretamente no editor de programa, obedecendo as cores verde, vermelho ou amarelo.



## Caixas de Display

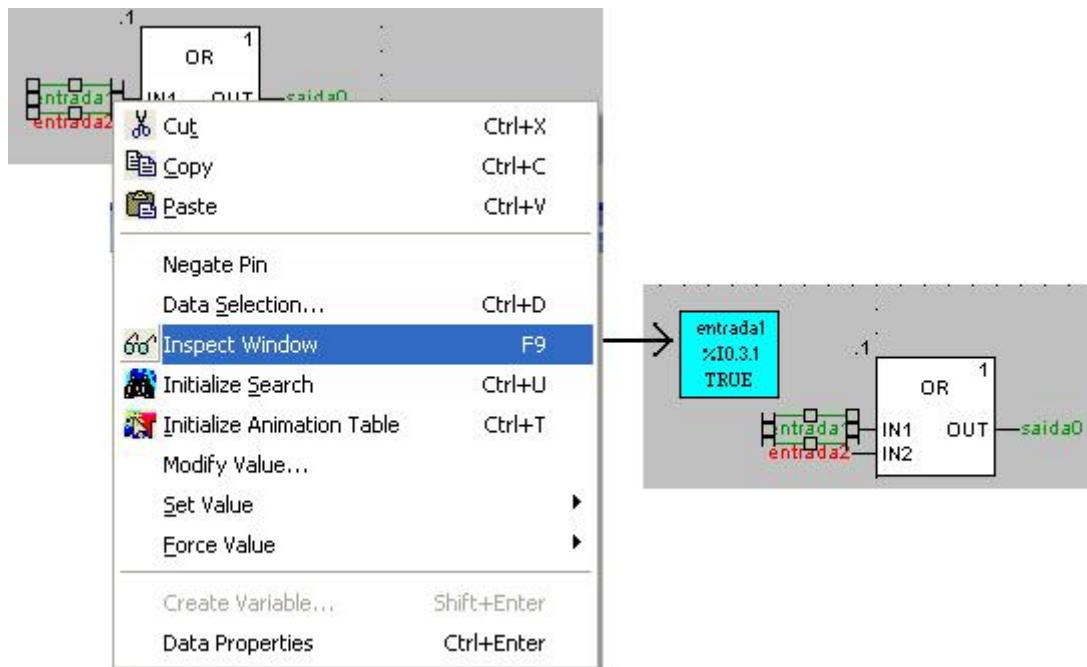
**Inspect window (1):** anexa a uma variável a visualização do valor da mesma, a cor da janela depende do valor comparado com valores mínimo e máximo (amarela, Azul ou violeta)

**Tool tip (2):** ao passar com o prompt sobre uma variável analógica, é visualizado seu valor, ou variável booleana é visualizado o nome da mesma.



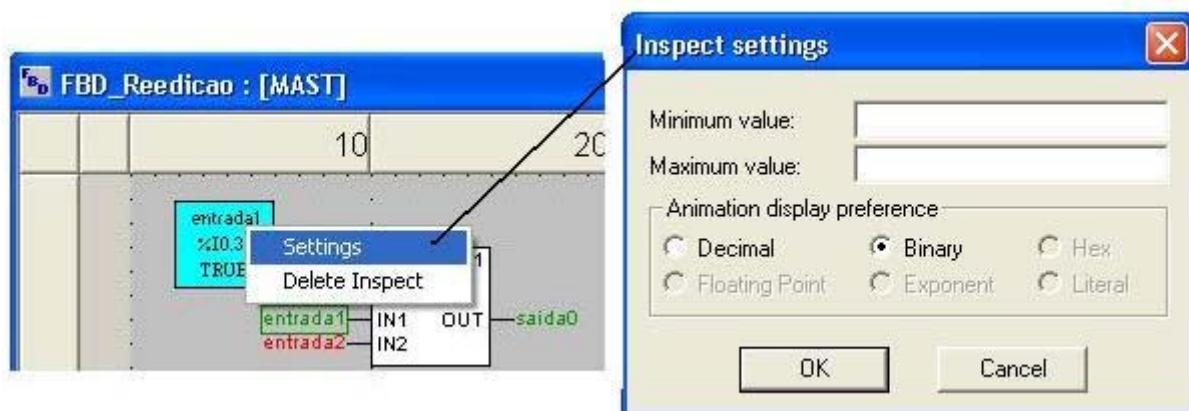
## Acesso ao inspect window

O acesso ao inspect window é feito através da seleção da variável desejada e clicar com botão direito do mouse sobre a mesma e selecionar “Inspect Window”, será anexada na área de edição a janela “inspect window” com as informações da variável



### Configuração do “Inspect Window”.

Ao clicar com o botão esquerdo sobre o “Inspect window” é possível selecionar a configuração dos valores mínimos e máximos para mudança de cor do mesmo ou apagá-lo.

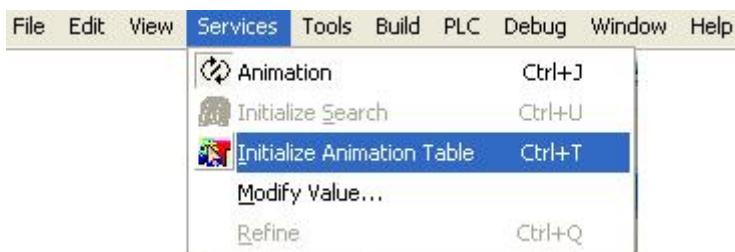


### Tabela Animada

Como visto no “Cap. 14 Teste de Aplicação” na parte de simulação do funcionamento do projeto, é possível forçar e alterar valores de uma variável.

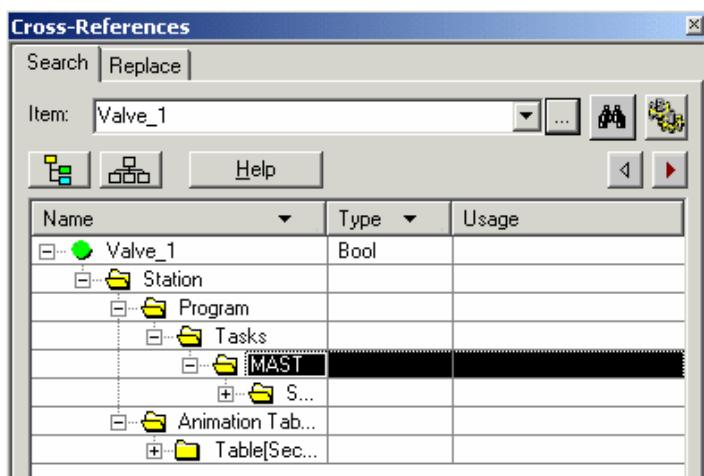
Name	Value	Type	Comment
Level_1	500	Int	
Pump_1	1	Bool	
Level_2	0	Int	
Pump_2	0	Bool	
Level_3	270	Int	
Valve_1	0	Bool	
Valve_2	1	Bool	
Box	0	Int	
Valve_3	0	Bool	

A partir da aba “Services” é possível iniciar a tabela animada pela seleção de “Initialize Animation Table” a qual apresenta todas as variáveis utilizadas na seção.



### **Referência cruzada - Cross Reference**

A ferramenta de referência cruzada permite visualizar o uso de uma variável em qualquer ponto do programa onde a mesma é utilizada.



### **Tela de Depuração do CLP - PLC debug screen**

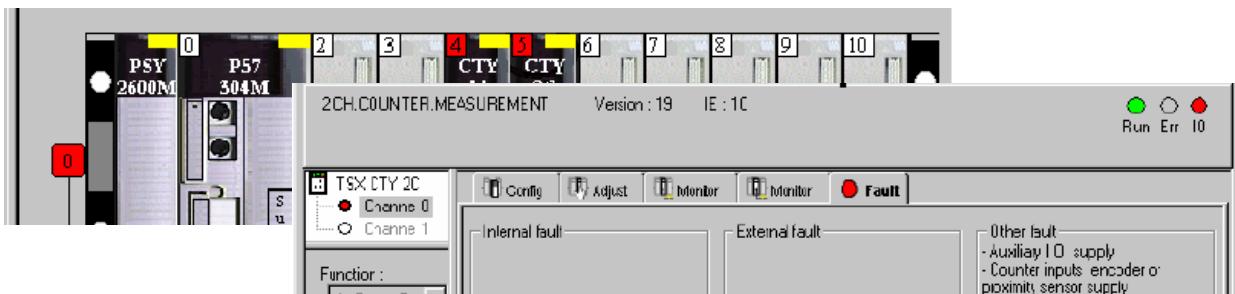


Mostra o status e comandos de execução da tarefa do programa, ajuste do relógio em tempo real assim como informações no projeto e processador do CLP.

### **Módulo de reportagem de falha.**

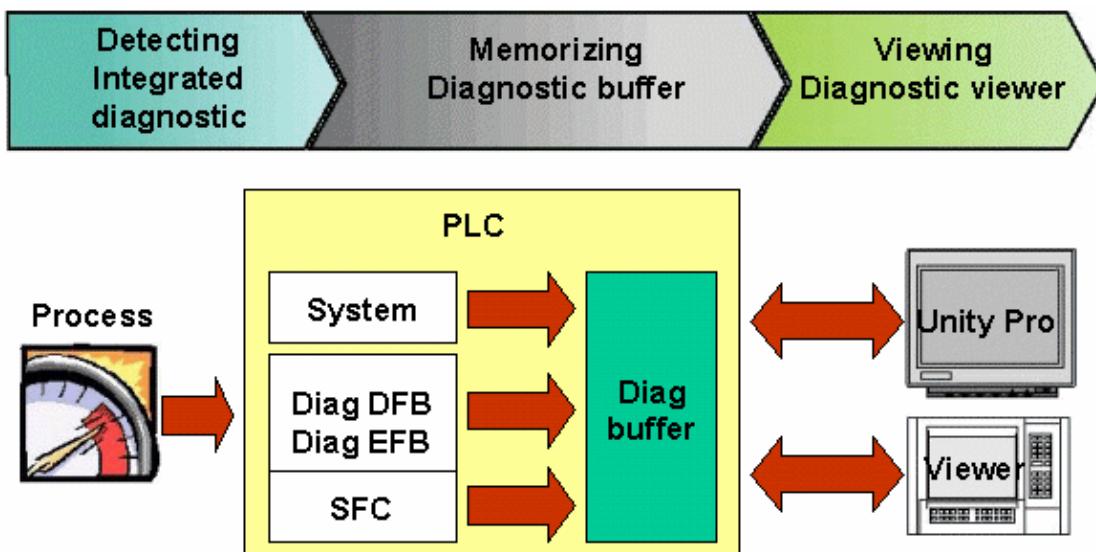
Uma falha não mascarada é reportada em:

- No módulo através da vista centralizada;
- Na tela de configuração do rack (marca vermelha);
- Em todas as telas do módulo ou canal em falha (marca vermelha e words de status)
- Com o objeto de uma linguagem dedicada ( bit de erro e word de status).



## Diagnóstico

**Arquitetura do Diagnóstico.**

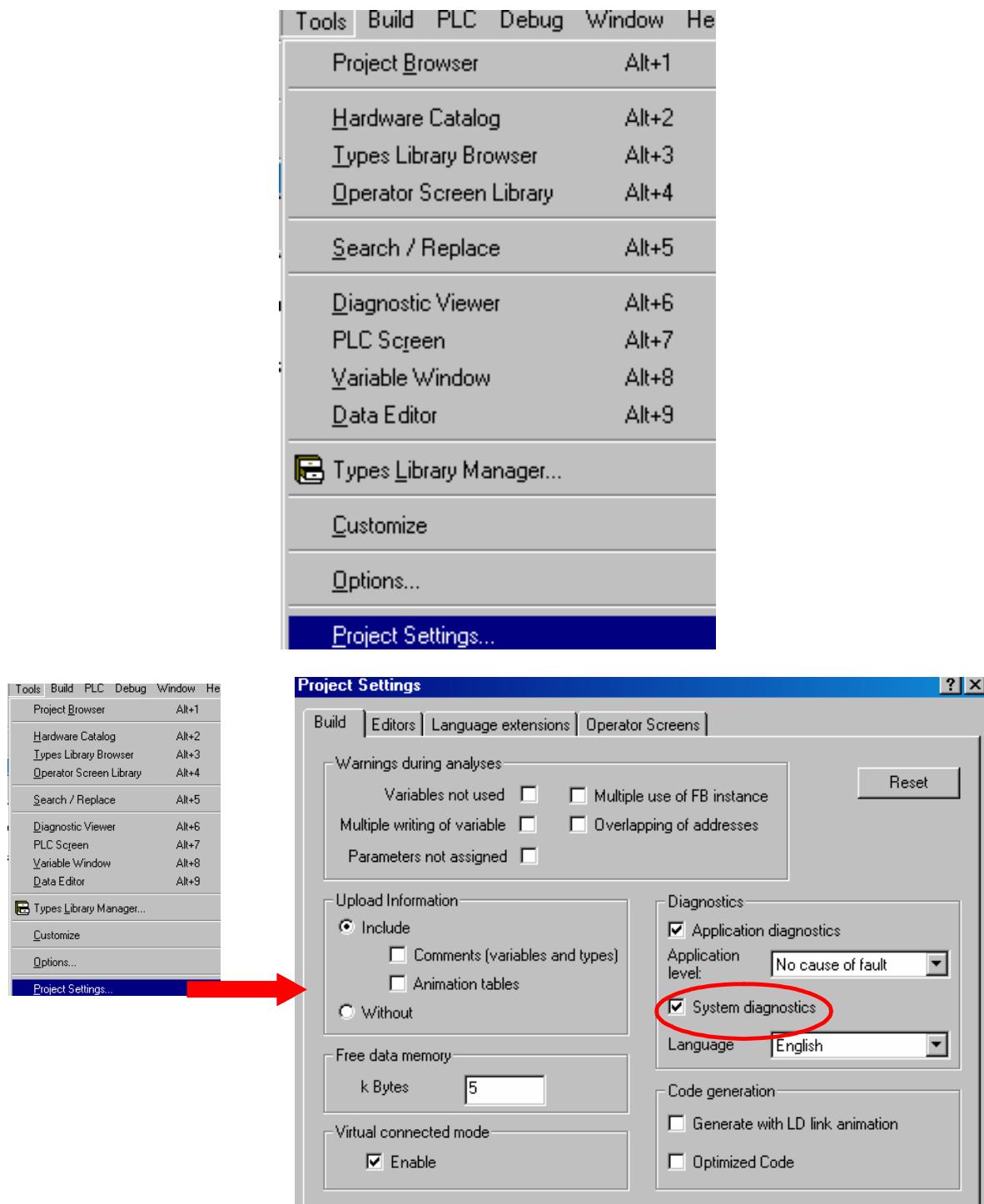


O sistema de diagnóstico funciona automaticamente (sem necessidade de programação), todos os erros do sistema são reconhecidos e salvos no CLP.

O sistema de diagnóstico utiliza os alarmes default dos bits de sistema e words (por exemplo, %S68 que indica falha na bateria de backup do processador Premium)

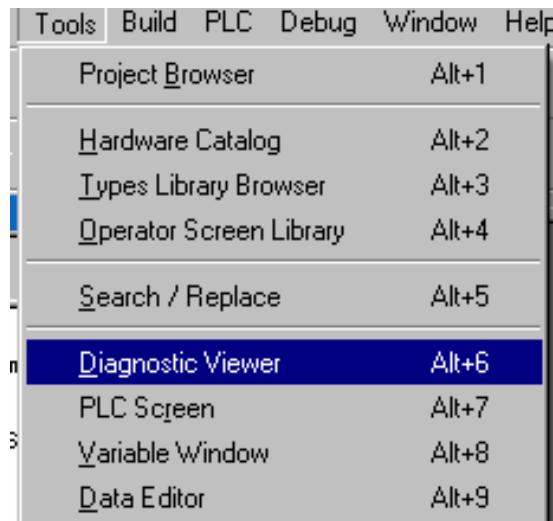
**Configuração do Diagnostic Viewer.**

Usando a aba **Tools** na barra de ferramentas selecionar **Project Settings**, marcar **Application diagnostic** e **System diagnostics** para visualizar mensagens de erro no visualizador de diagnóstico.



Na caixa Language, definir o idioma da mensagem a ser visualizada, na caixa Application level seleciona-se ou não a visualização da causa da falha.

O visualizador de diagnóstico pode ser acessado da barra de rolagem **Tools** ou pelo uso de tecla de atalho **Alt+6**.



Acknowledgment : 11	Message	Fault	Symbol	Area	Appearance Date : 14	Disappearance Date : 1
	Backup battery fault	System ... %S68	0	1/24/2006 8:41:11 AM		
	Backup battery fault	System ... %S68	0	1/23/2006 2:32:37 PM	1/23/2006 2:54:16 PM	
	Arithmetic error	System ... %S18 (MAST)	0	1/20/2006 1:47:32 PM		
	FB Alarm ALARM_DFB_10	0	1/20/2006 11:45:15 AM			
	FB Alarm ALARM_DFB_9	0	1/20/2006 11:45:15 AM			
	FB Alarm ALARM_DFB_8	0	1/20/2006 11:45:15 AM			
	FB Alarm ALARM_DFB_7	0	1/20/2006 11:45:15 AM			
	FB Alarm ALARM_DFB_6	0	1/20/2006 11:45:15 AM			

Para cada um dos alarmes, a lista de mensagem de erros mostra as seguintes informações:

**Acknowledgement** – O ícone e o texto indicam o status da mensagem: Não reconhecida “not acknowledged”, reconhecida “acknowledged”, apagada ou não reconhecida.

**Error message** – Esta mensagem de fato é o texto que foi editado no comentário da instância do bloco de função que gerou o erro.

**Fault** – Tipo de diagnóstico FB ou bit de sistema que detectou o erro.

**Symbol** – símbolo associado a uma falha.

**Area** - Zona de falha do CLP.

**Appearance** – Data e hora da aparição da falha.

**Disappearance** - Data e hora da desaparição da falha.

**Acknowledge** - Data e hora do reconhecimento da falha.

## Bits e Words do Sistema para Diagnóstico

### Bits de Sistema

Os CLPs Modicon M340, Premium, Atrium e Quantum usam bits de sistema %Si os quais indicam o estado do CLP, ou podem ser utilizados para verificar como os mesmos operam.

Estes bits podem ser testados no programa de usuário para detectar um determinado funcionamento requerendo um conjunto de procedimento para processamento.

Alguns destes bits precisam ser “resetados” para o seu valor inicial ou estado normal pelo programa.

### Bits de Sistema utilizados para diagnóstico

Objeto de sistema:	Descrição do Alarme
%S10	Erro de Entrada/Saída
%S11	Disparo de Watchdog!
%S15	Falha de string de caracteres
%S18	Overflow ou erro aritmético
%S19	Estouro do período de uma task
%S20	Estouro de índice
%S39	Saturação no processamento de evento

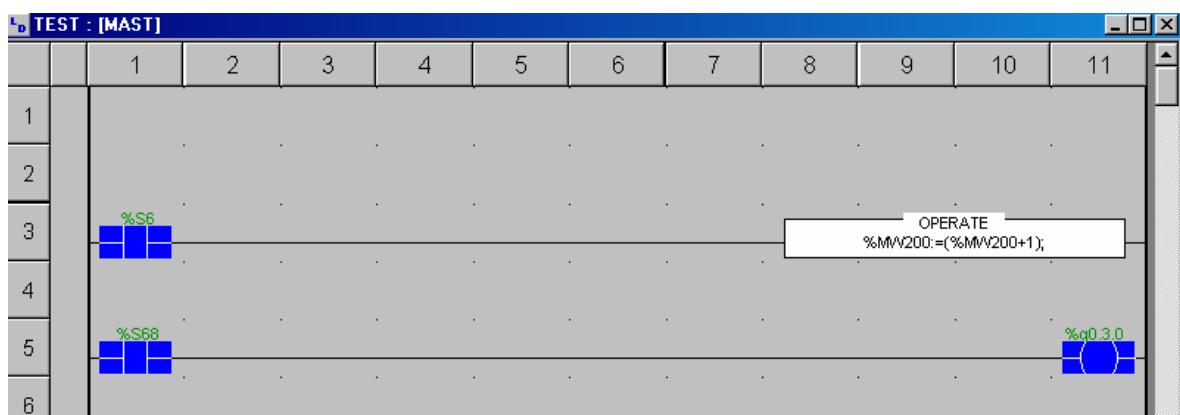
%S51	Perda do tempo no relógio de tempo real.
%S67	Estado da bateria do cartão de memória PCMCIA.
%S68	Estado da bateria do processador.
%S76	Buffer de diagnóstico configurado
%S77	Buffer de diagnóstico cheio.
%S118	Falha geral de Fipio I/O.
%S119	Falha geral no rack de I/O.

### Words de Sistema utilizadas para diagnóstico

%SW0	Período de scan de Master task
%SW1	Período de Fast task
%SW2	Período de auxiliary task scanning 0.
%SW3	Período de auxiliary task scanning 1.
%SW4	Período de auxiliary task scanning 2.
%SW5	Período de auxiliary task scanning 3.
%SW11	Duração de Watchdog
%SW17	Status de erro de operações com ponto flutuante.
%SW76	Função de diagnóstico: save
%SW77	Função de diagnóstico: de-registration
%SW78	Função de diagnóstico: number of errors
%SW125	Tipo de falha de bloqueio
%SW146	Função de visualização árbito do barramento Fipio.
%SW153	Lista de falhas gerenciadas dos canais de Fipio.
%SW154	Lista de falhas gerenciadas dos canais de Fipio.

Bits e words de sistema podem ser utilizados através da Aplicação para executar uma lógica ou verificar as condições do sistema.

No exemplo a seguir, o bit de sistema %S6 executa o bloco operate e %S68 monitora a situação da carga da bateria do CLP.



O estado dos bits e words de sistema podem ser verificados na tabela animada.

Name	Ad...	Value	Comment	Type
SYSTEM_BIT_6	%S6	1	Time base 1 s	BOOL
SYSTEM_BIT_9	%S9	0	Outputs set to the fallback position on all buses	BOOL
SYSTEM_BIT_10	%S10	1	Input/output fault	BOOL
SYSTEM_BIT_13	%S13	0	First cycle after switching to RUN	BOOL
SYSTEM_BIT_19	%S19	0	Task period overrun (periodical scanning)	BOOL
SYSTEM_BIT_30_TO_35	%S30	1	Activation/deactivation of the tasks	BOOL
SYSTEM_BIT_68	%S68	0	State of processor battery	BOOL
SYSTEM_WORD_8	%SW8	2#0000_0000_0000_0000	Inhibits the input acquisition phase of each task:	INT
SYSTEM_WORD_9	%SW9	2#0000_0000_0000_0000	Inhibits the output updating phase of each task	INT
SYSTEM_WORD_49	%SW49	16#0003	Day of the week	INT
SYSTEM_WORD_50	%SW50	16#0200	Seconds (16#SS00)	INT
SYSTEM_WORD_51	%SW51	16#0934	Hours and Minutes (16#HHMM)	INT
SYSTEM_WORD_52	%SW52	16#0308	Month and Day (16#MMDD)	INT
SYSTEM_WORD_53	%SW53	16#2006	Year (16#YYYY)	INT
SYSTEM_WORD_54_TO_58	%SW54	0	Contain date and time of the last power failure or ...	INT

## Bits e Words de sistema

Dependendo do controlador utilizado (Modicon M340, Atrium, Premium ou Quantum) estes podem possuir os seguintes Bits de Sistema.

%S0: Partida a frio

%S1: Retorno a quente

%S4: clock com tempo base de 10 ms

%S5: clock com tempo base de 100 ms

%S6: clock com tempo base de 1 s

%S7: clock com tempo base de 1 min

%S9: Coloca as saídas para a posição de fallback em todo o barramento.

%S10: Falha de Input/output

%S11: Estouro de watchdog

%S12:PLC em RUN

%S13: primeiro ciclo após a colocação em RUN.

%S15: Falha de string de caractere.

%S17: rotacionar ou deslocar saída

%S18: Overflow ou erro aritmético

%S19 :Estouro do período de uma task

%S20: Estouro de índice

%S21: Ciclo da primeira tarefa.

%S30: Ativação/desativação da tarefa master

%S31: Ativação/desativação da tarefa fast

%S32 a %S35: Ativação/desativação da tarefa auxiliary tasks 0-3

%S38:habilitação/desabilitação de eventos

%S39: Saturação no processamento de evento

%S40: Falha de input/output no Rack 0

%S41: Falha de input/output no Rack 1

%S42: Falha de input/output no Rack 2

%S43: Falha de input/output no Rack 3

%S44: Falha de input/output no Rack 4

%S45: Falha de input/output no Rack 5

%S46: Falha de input/output no Rack 6

%S47: Falha de input/output no Rack 7

%S50:Atualização da data e Hora via words %SW50 to %SW53

%S51: Perda do tempo no relógio de tempo real.

%S59:Incremento da data e hora via word %SW59

%S60: Comando de chaveamento voluntário

%S62: Proteção de memória desligada (OFF)

%S65: Desabilitação do cartão (Mirano)

%S66: Backup da aplicação

%S67: Estado da bateria do cartão de memória PCMCIA.  
%S68: Estado da bateria do processador.  
%S75:Estado da bateria do cartão de memória de armazenamento  
%SW76:Função de diagnóstico: save  
%SW77:Função de diagnóstico: de-registration  
%SW78:Função de diagnóstico: number of errors  
%S80:Zera contadores de mensagem  
%S90: Atualiza words comuns  
%S91:Trava requisição assíncrona  
%S92: Modo de medida da função de comunicação  
%S94:Salvando valores de ajustes  
%S95:Restaurando valores iniciais  
%S96: backup do programa OK  
%S97: salvamento de %MW OK  
%S100:Protocolo no terminal port  
%S118:Falha geral de Fipio I/O.  
%S119:Falha geral no rack de I/O.  
%S120: Falha no barramento DIO (CPU)  
%S121:Falha no barramento DIO (NAME No. 1)  
%S122:Falha no barramento DIO (NAME No. 2)  
Dependendo do controlador utilizado (Modicon M340, Atrium, Premium ou Quantum) estes podem possuir as seguintes Words de Sistema.  
%SW0:Período de scan de Master task  
%SW1:Período de Fast task  
%SW2:Período de auxiliary task scanning 0.  
%SW3:Período de auxiliary task scanning 1.  
%SW4:Período de auxiliary task scanning 2.  
%SW5:Período de auxiliary task scanning 3.  
%SW6 e %SW7:Endereço IP Address  
%SW8:Aquisição da monitoração de entrada de tarefa  
%SW9:Monitoração da atualização da tarefa de saída  
%SW10:Primeiro ciclo após cold start  
%SW11:Duração de Watchdog  
%SW12:modo do processador de application  
%SW13:modo do processador Intel.  
%SW14:Versão comercial do processador do CLP  
%SW15:Versão do processador do CLP  
%SW16:Número da versão de Firmware  
%SW17:Status de erro de operações com ponto flutuante.  
%SW18 e %SW19:contador de tempo absoluto  
%SW20 e %SW21: contador de tempo absoluto  
%SW23: valor da chave rotativa  
%SW26:número de requisições processadas  
%SW27,%SW28 e %SW29: Tempo de overhead do sistema  
%SW30: tempo de execução de tarefa master  
%SW31: tempo máximo de execução de tarefa master  
%SW32: tempo mínimo de execução de tarefa master  
%SW33:Tempo de execução de tarefa fast  
%SW34: Tempo de máximo de execução de tarefa fast  
%SW35: Tempo mínimo de execução de tarefa fast  
%SW36: tempo de execução de auxiliary task 0  
%SW39 tempo de execução de auxiliary task 1  
%SW42 tempo de execução de auxiliary task 2  
%SW45: tempo de execução de auxiliary task 3  
%SW37: tempo máximo de execução de auxiliary task 0  
%SW40 : tempo máximo de execução de auxiliary task 1  
%SW43 : tempo máximo de execução de auxiliary task 2  
%SW46: tempo máximo de execução de auxiliary task 3

%SW38 a %SW47: Tempo mínimo de execução de auxiliary tasks  
%SW48:número de eventos  
%SW49 a %SW53:funções de relógio de tempo real  
%SW54 a %SW58: Função de relógio de tempo real na última parada.  
%SW59: Ajuste da data corrente  
%SW70: Função de relógio de tempo real  
%SW71:posição da chave no painel frontal Quantum  
%SW75:Contador de tempo de evento  
%SW76:Função de diagnóstico: save  
%SW77:Função de diagnóstico: de-registration  
%SW78:Função de diagnóstico: number of errors  
%SW80 a %SW84: Gerenciamento de mensagem  
%SW85 Premium: Gerenciamento de Telegrama  
%SW86 Modicon M340: Gerenciamento de Telegrama  
%SW87: gerenciamento do fluxo de comunicação  
%SW88 e %SW89: Premium: gerenciamento do fluxo de comunicação  
%SW90: número máximo de requisições processadas por ciclo de tarefa master  
%SW91-92: Taxa de mensagem de blocos de função  
%SW93: arquivo de sistema do cartão de memória (Mirano)  
%SW94 e %SW95: Assinatura de modificação de aplicação.  
%SW96:Comando e diagnóstico para salvar e restaurar.  
%SW97: Status do cartão  
%SW99: Gerenciamento de comunicação de redundância  
%SW108:número de bits de modulo de I/O forçados  
%SW109:número de canais analógicos forçados  
%SW116:erro de Fipio I/O  
%SW122: Tipo de Restart  
%SW124: tipo de falha de sistema  
%SW125:Tipo de falha de bloqueio  
%SW126 e %SW127:endereço da instrução de falha de bloqueio.



Alem das Words de sistema citadas anteriormente, os controladores Atrium, Premium, Quantum e Modicon M340 possuem words específicas para os mesmos.

## CAPÍTULO 12

### Tipo de Dados Derivados - DDT

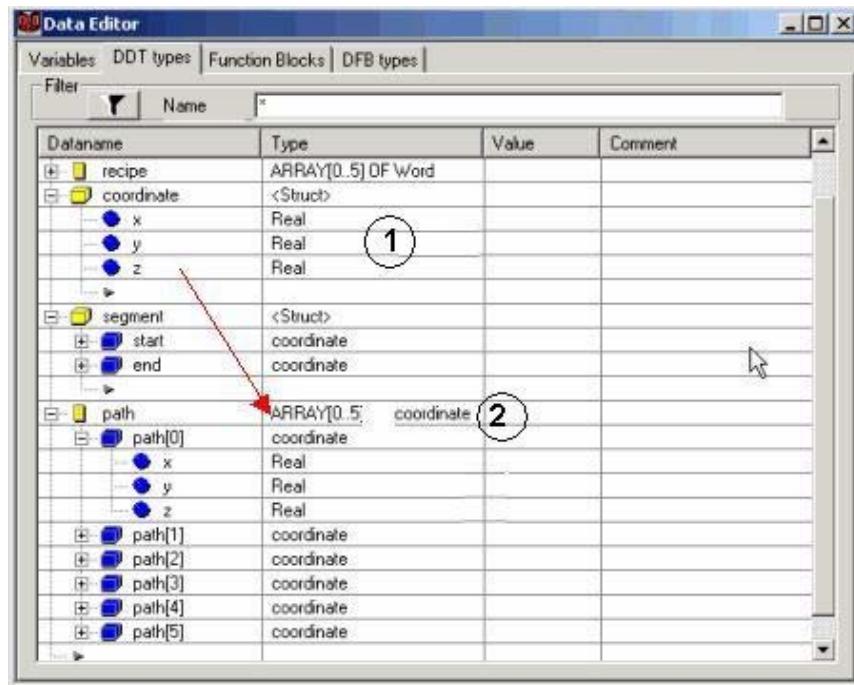


# Tipo de Dados Derivados - DDT

## Tipo de Dados Derivados - Derived Data Type- DDT

Um tipo de dado derivado é uma definição de uma estrutura de qualquer tipo predefinido (**EDT, DDT**).

Estes podem ser aninhados (de até 8 níveis), incluindo arrays (de até 6 dimensões) de tipos de dados elementares **(1)** ou tipo de dados derivados **(2)**.

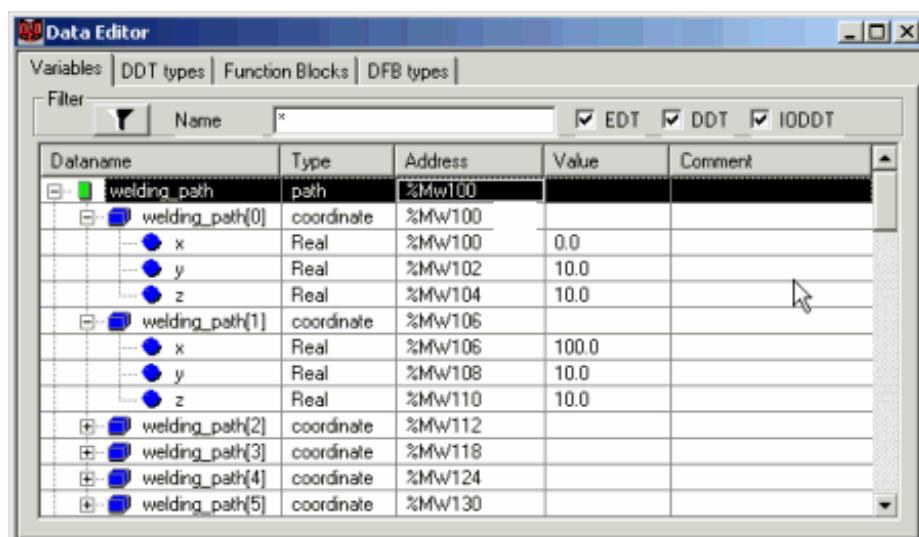


## Variáveis Tipos DDT.

Para usar uma instância de estrutura, deve-se:

- => Definir a estrutura
- => Aninhar as estruturas

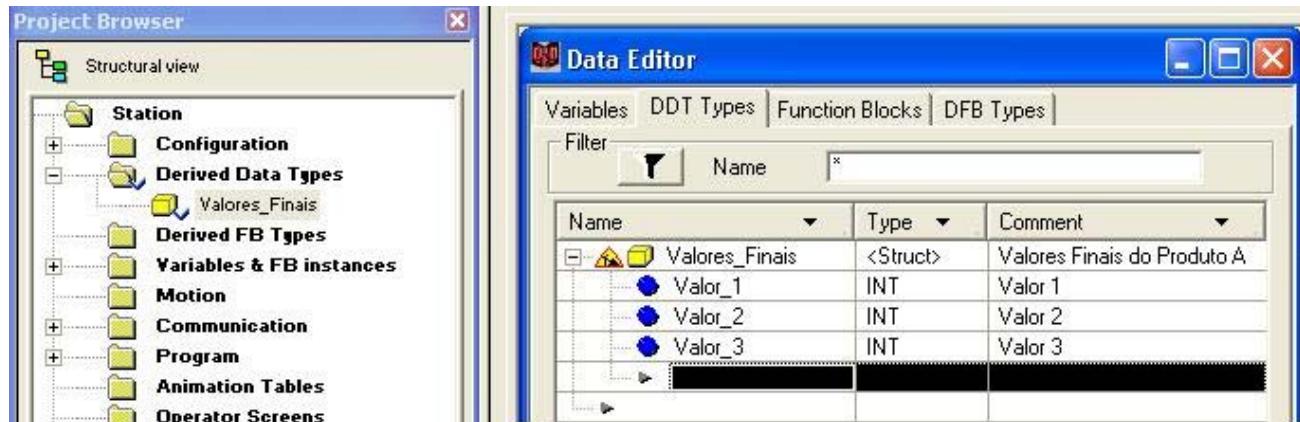
Os DDTs podem ser usados como um tipo de dado elementar para definir uma instância de uma variável. Estes podem ser mapeados na memória física se for necessário.



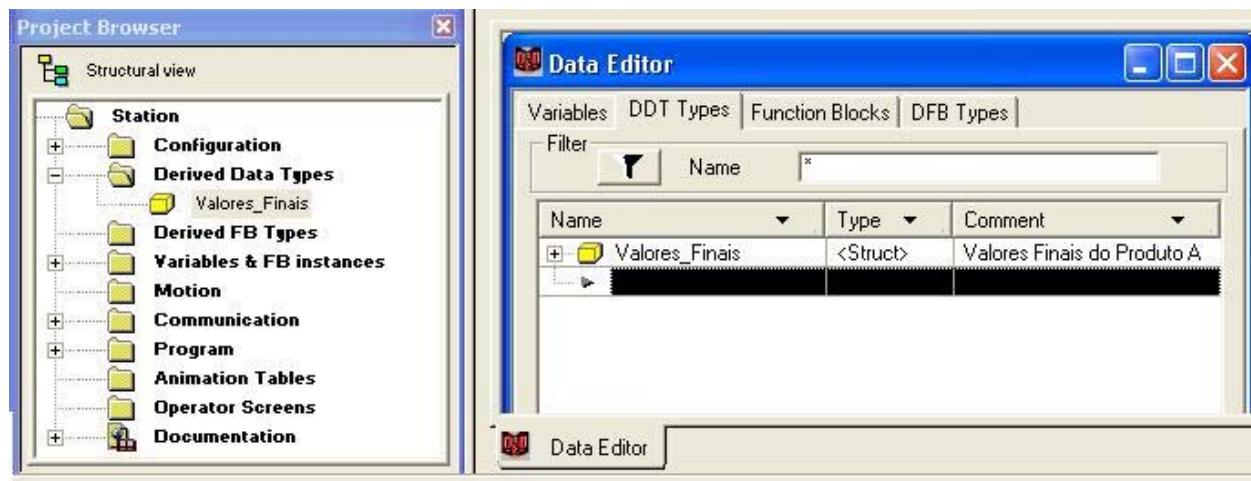
## Edição de Variáveis Tipo DDT

### Definição da Estrutura

A definição da estrutura é feita no Project Browser na pasta “Derived Data Types” que ao ser selecionada deve-se clicar com o botão direito e selecionar “Open”, e será aberto o editor de dados “Data Editor” onde na coluna “Name” é definido o nome da variável derivada e seus elementos e na coluna “Type” é definida o tipo de estrutura que pode ser “Array” ou “Struct”.

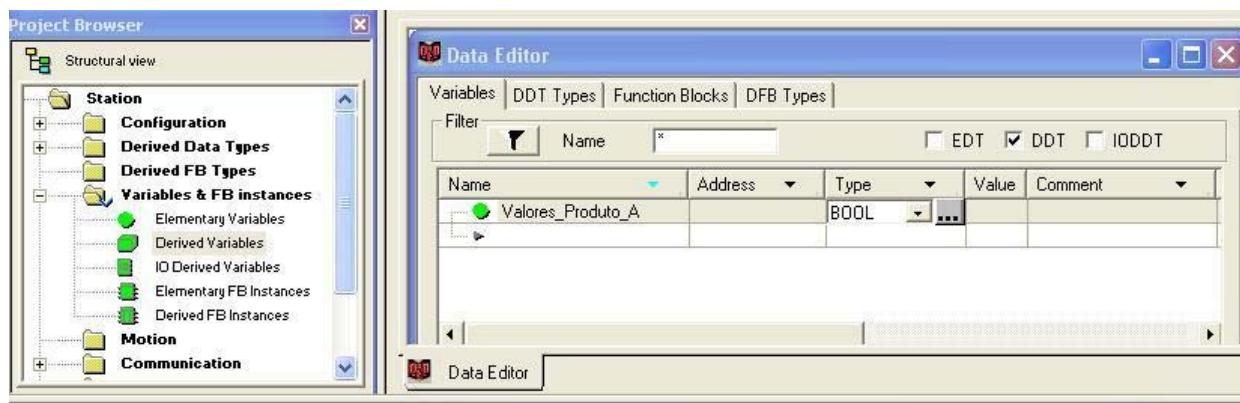


Após a definição da estrutura deve-se fazer a compilação “Build”, validando assim a estrutura criada.

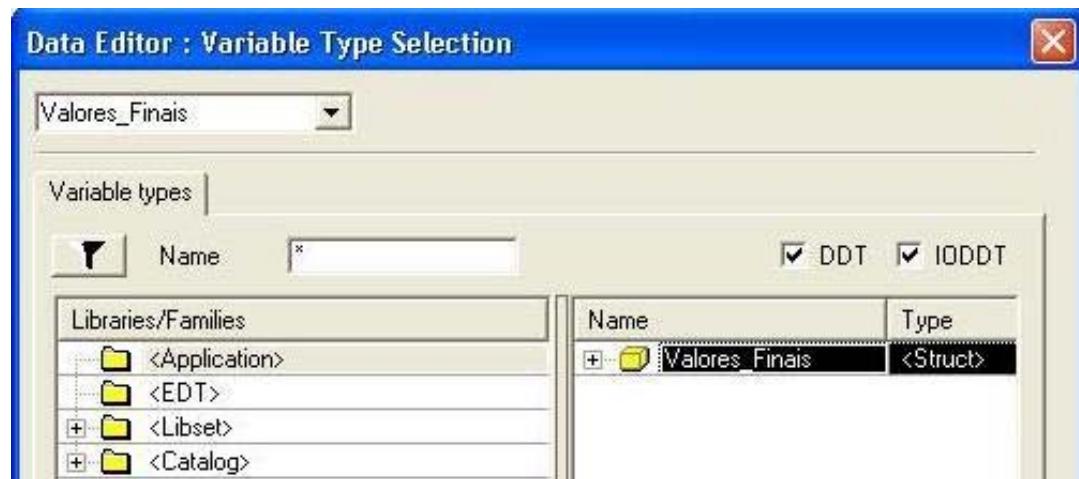


### Aplicação das Variáveis Tipo DDT

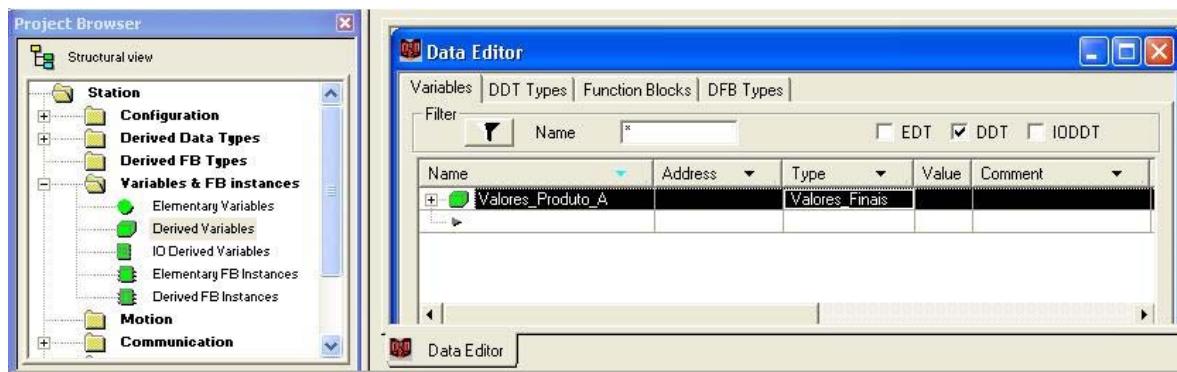
A partir do project browser, selecionar a pasta “Derived Variables” e clicar em “Open”, será aberto o editor de dados “Data Editor” onde na coluna “Name” deve-se especificar a variável e na coluna “Type” clicar no botão [...] para selecionar a estrutura criada anteriormente.



Ao clicar no botão [...], é aberto o editor de Texto “Data Editor: Variable Type Selection”, onde na coluna “Name” deve ser selecionada a estrutura criada anteriormente.

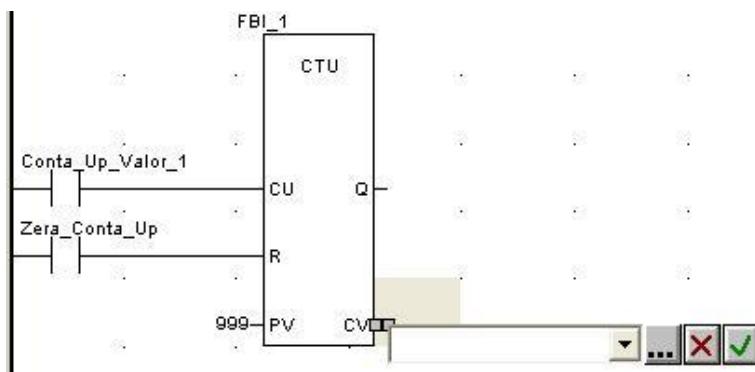


Ao clicar no botão OK no editor de Texto “Data Editor: Variable Type Selection”, a variável é criada, para finalizar deve se feita a compilação “Build” validando assim a criação da variável derivada.

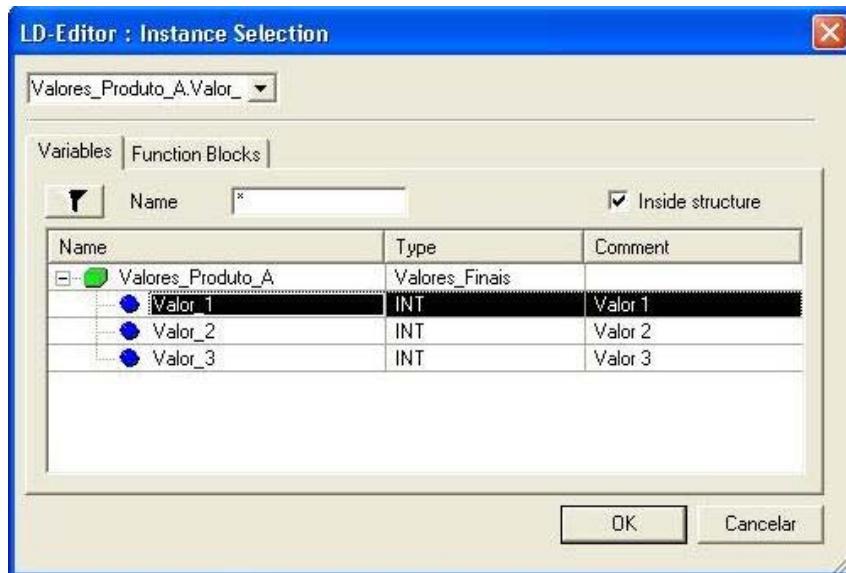


### Aplicação.

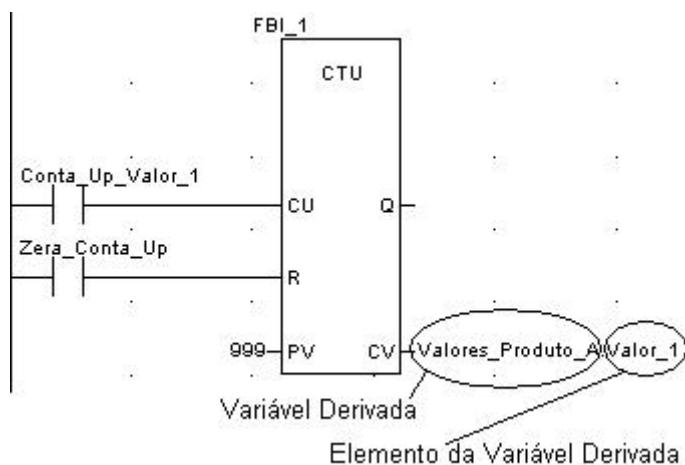
O contador “FBI\_1” computará o “Valor\_1” da variável estruturada “Valores\_Produto\_A”.



Na definição da variável deve-se acionar o botão [...], onde será aberto o editor “LD-Editor: Instance Selection” na qual deve ser marcada a caixa “Inside structure” para visualização das variáveis estruturadas criadas anteriormente, selecionar a variável desejada e acionar o botão OK.



Ao acionar o botão **OK**, a variável é adicionada na lógica.



## Tipo de Dados Derivados de Entradas/Saídas - I/DDT:

O tipo de dado específico I/DDT permite o mapeamento completo de uma estrutura de I/O (1) de um canal diretamente de um determinado módulo (2) com um nome de dado (3).

Automaticamente a estrutura é mapeada para os bits de I/O e words (4).

Um apelido “Alias” (5) pode ser dado para nomear uma estrutura de variável particular.

O I/DDT permite programar sem um endereço físico, isto simplifica a criação de módulos padrões, o mapeamento pode ser feito após a exportação.

Na figura são mostradas as variáveis derivadas referente a uma entrada analógica.

## Tipo de Dados Derivados - DDT

Variables | DDT Types | Function Blocks | DFB Types |

Filter  Name  EDT  DDT  IODDT

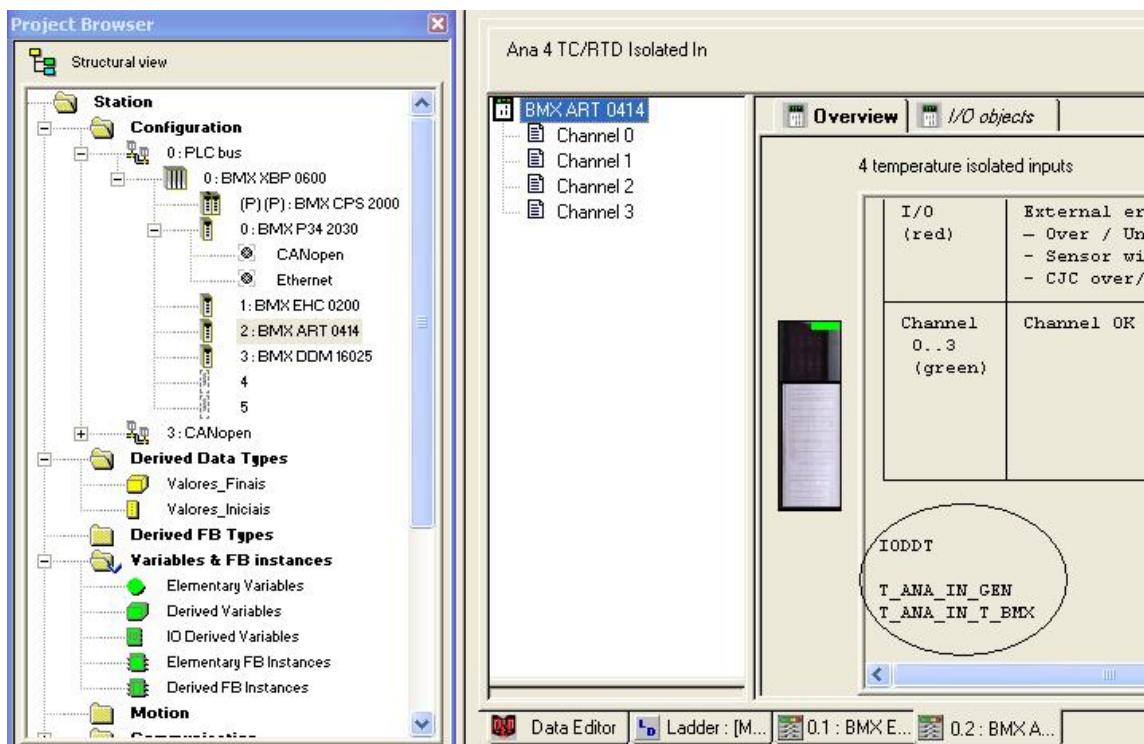
Name	Alias	Type	Address	Value	Comment
Analog_input10		T_ANA_I...	%CH0.4.10		
(1) temp_feeding_box		INT	%IW0.4.12.0	(2)	
(3) Analog_input12		T_ANA_I...	%CH0.4.12		
(3) CH_ERROR	(5) temp_feeding_box	BOOL	%IO.4.12.ERR		Channel error
(3) EXCH_STS		INT	%MW0.4.12.0	(4)	Analog input value
(3) STS_IN_PR...		INT	%MW0.4.12.0.0		Exchange status
(3) CMD_IN_PR...		BOOL	%MW0.4.12.0.1		Status parameter...
(3) ADJ_IN_PR...		BOOL	%MW0.4.12.0.2		Command param...
(3) EXCH_RPT		INT	%MW0.4.12.1		Adjust parameter...
(3) STS_ERR		BOOL	%MW0.4.12.1.0		Error while readin...
(3) CMD_ERR		BOOL	%MW0.4.12.1.1		Error while sendi...

## Edição de Tipo de Dados Derivados de Entradas/Saídas - I/ODDT

O Tipo de dado I/ODDT esta associado diretamente a um módulo de I/O, por exemplo, no módulo analógico “BMX ART 0414”, os I/ODDTs associados são:

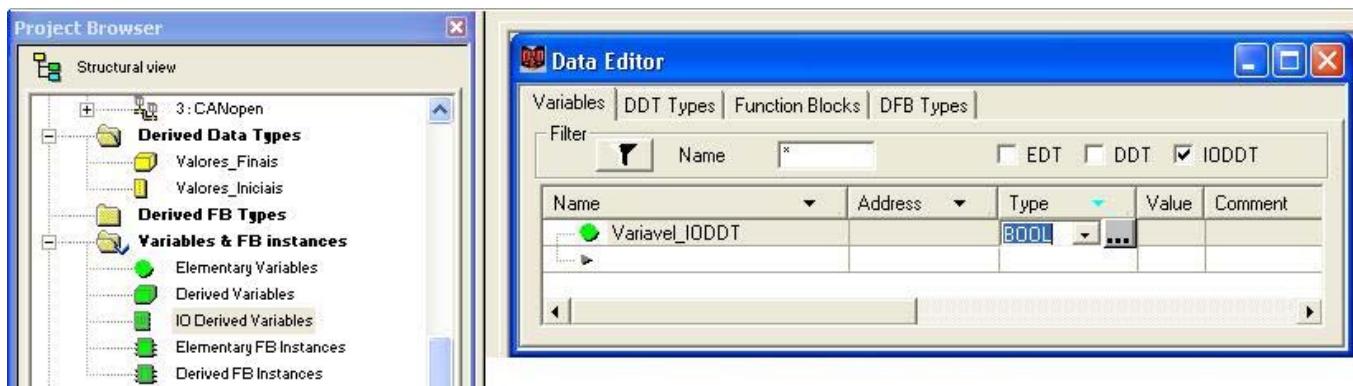
T\_ANA\_IN\_GEN => aplicável a todos os módulos de entrada analógicos, que contem o valor do canal e um bit de erro do mesmo.

T\_ANA\_IN\_T\_BMX => Aplicável apenas ao modulo de entrada analógico BMX AMI 0410, que contem o valor do canal e um bit de erro do mesmo.

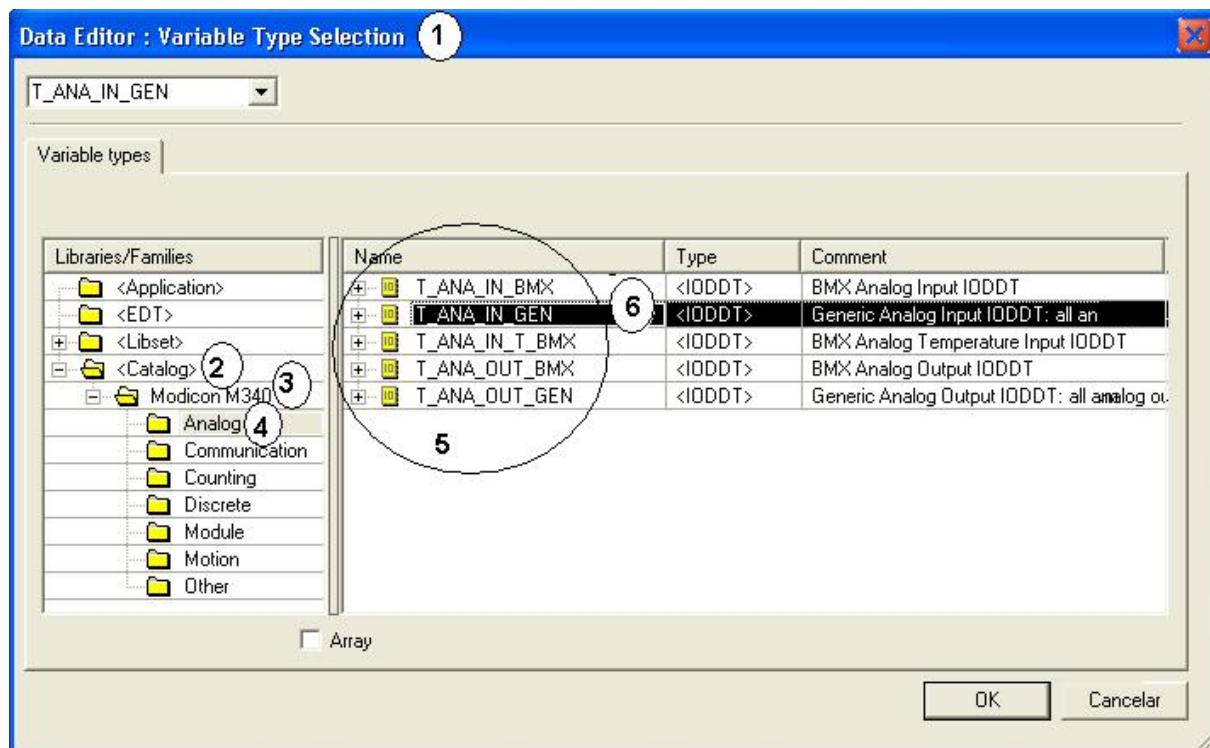


## Criação I/ODDT.

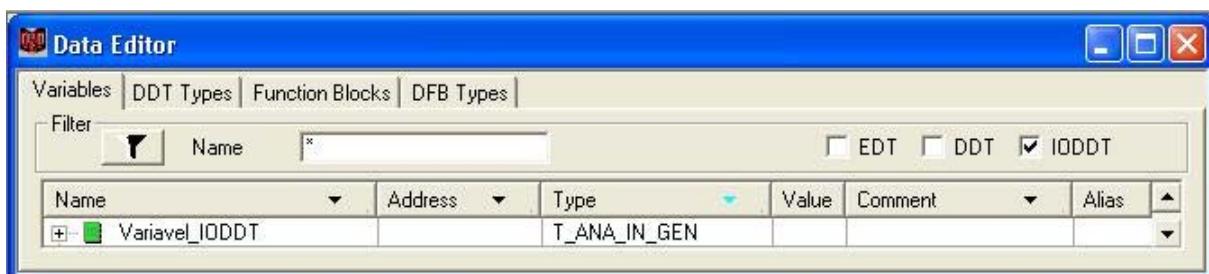
A partir do Project Browser, selecionar a pasta “IO Derived Variables”, clicar com o botão direito e selecionar “Open” e será aberto o editor de dados “Data Editor” onde na coluna “Name” deve-se especificar o nome do IODDT e na coluna “Type” clicar no botão [...].



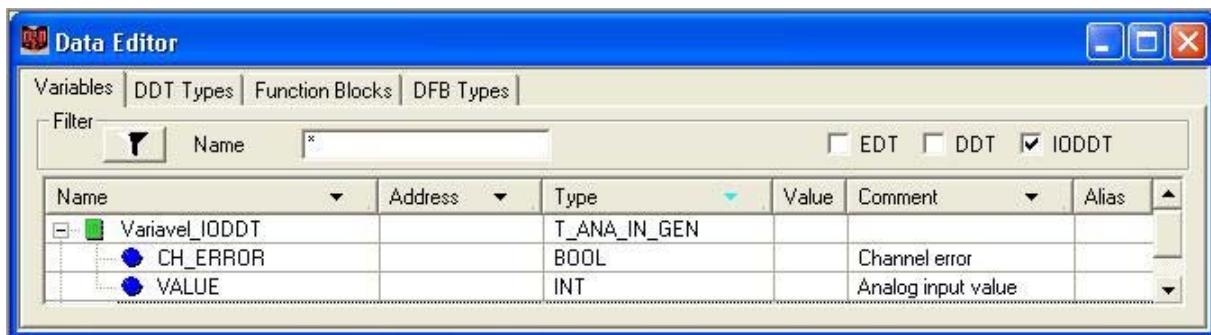
Ao clicar no botão [...] e será aberto o editor de dados “Data Editor: Variable Type Selection” (1) expandir as pastas “Catalog” (2) e “Modicon M340” (depende do CLP utilizado) (3) e nesta pasta selecionar módulo desejado, por exemplo, “Analog” (4) e serão visualizados todos os IODDTs (5) referente aos módulos analógicos do CLP em uso, selecionar o IODDT desejado, por exemplo “T\_ANA\_IN\_GEN” (6), finalizando clicar no botão **OK**.



Ao clicar no botão OK, o IODDT criado será visualizado no “Data Editor”.



IODDT “Variável\_IODDT” expandido.



### Endereçamento do IODDT.

Um IODDT permite que todas as informações (bits e registros) relacionadas a um canal sejam manuseadas pelo usuário. Esta variável é definida no editor de dados do Unity Pro pela seleção do IODDT apropriado ao módulo assim como o tipo de dado e especificando o endereço topológico do módulo obedecendo a seguinte sintaxe:

%CH[\b.e]\r.m.c

Onde:

b = bus

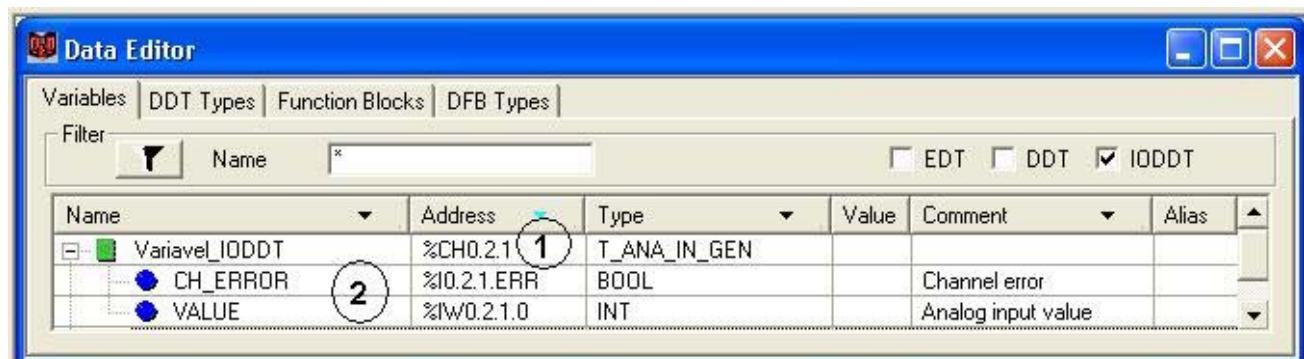
e = equipamento (drop)

r = rack

m = slot do módulo

c = canal

Por exemplo, endereçamento de módulo analógico de temperatura no slot 2, canal 1.

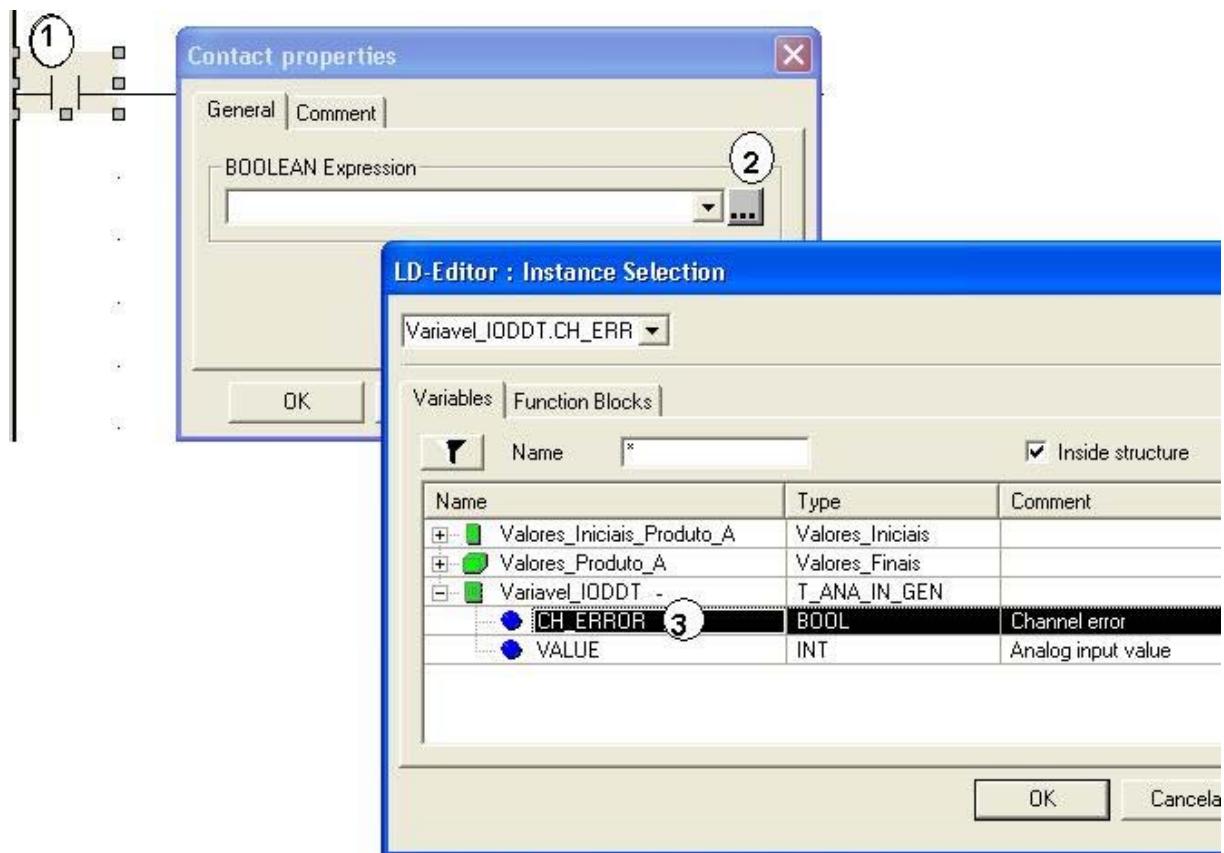


O endereço %CH0.2.1 (1) é colocado pelo usuário, os demais endereços (2) são gerados automaticamente.

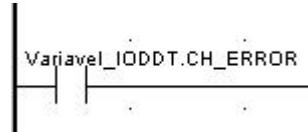
### Aplicação de IODDT

Monitorização de erro no canal analógico de entrada.

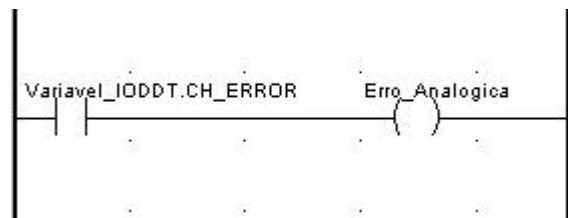
A utilização da variável IODDT é feita com dois cliques sobre o objeto (1) disponibilizando a caixa de propriedades do mesmo, na qual pode-se especificar diretamente o IODDT no campo “BOOLEAN Expression” ou clicar no botão [...] (2), disponibilizando o editor “LD-Editor: Instance Selection” onde deve-se expandir a variável desejada e selecionar o elemento da mesma (3), acionar o botão OK e finalizar a operação.



Objeto com o IODDT especificado.



Lógica completa.



# **CAPÍTULO 13**

## **Bloco de Função Derivado**

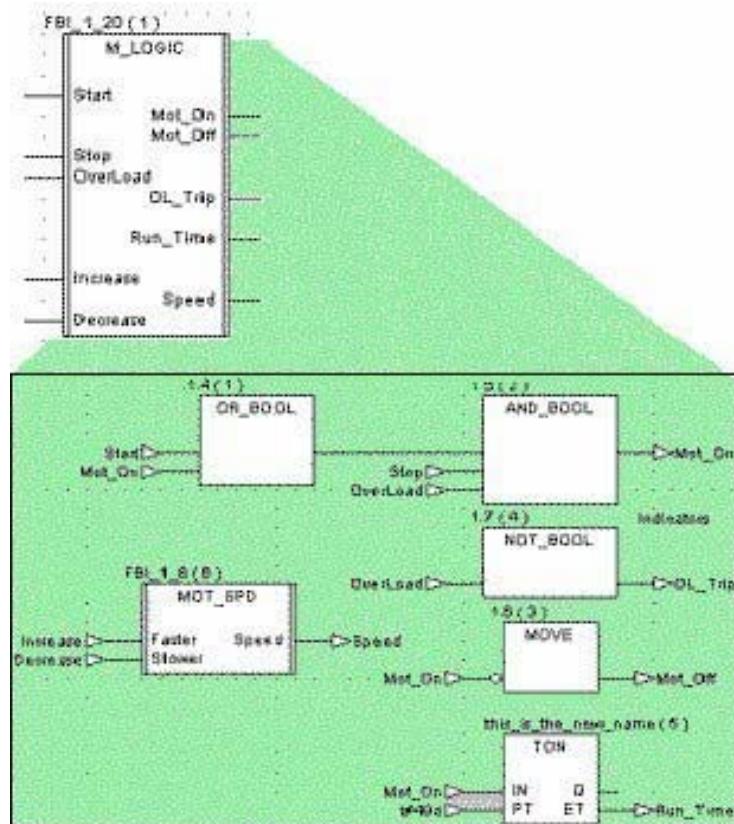
---



# Bloco de Função Derivado

## Introdução

Blocos de Função Derivado “Derived Function Block - DFB” utilizam uma lógica encapsulada para atender uma necessidade específica da aplicação para os casos onde a mesma é utilizada várias vezes na aplicação alterando apenas as variáveis em questão.



O programa que desenvolve do Bloco DFB utiliza um modelo chamado “DFB Type”. O usuário final cria uma imagem deste bloco chamado “Instance”, entrando com os parâmetros para cada uso na aplicação. Os DFBs podem ser Exportados/Importados entre projetistas e usuários sendo carregada uma única vez na CPU e podem ser protegidos. Os DFBs são locais a uma aplicação ou disponibilizados na biblioteca global.

## Vantagens do uso de DFBs.

- Aumentam a adaptabilidade do programa, promovendo na linguagem Ladder, uma visão gráfica funcional do bloco tornado fácil a programação e a depuração.
- Torna fácil a depuração, as variáveis do CLP são manuseadas pelo bloco de função DFB são idênticas na interface.
- Reduz o volume de código gerado, pois o código correspondente ao DFB é carregado somente uma vez, e podem ser chamados várias vezes no mesmo programa.

## Características do DFB.

### Estrutura do DFB.

O bloco DFB é composto por:

Nome: máximo 32 caracteres

Parâmetros: Entradas, saídas e entradas/saídas.

Variáveis internas do tipo:

Pública: acessível pela aplicação

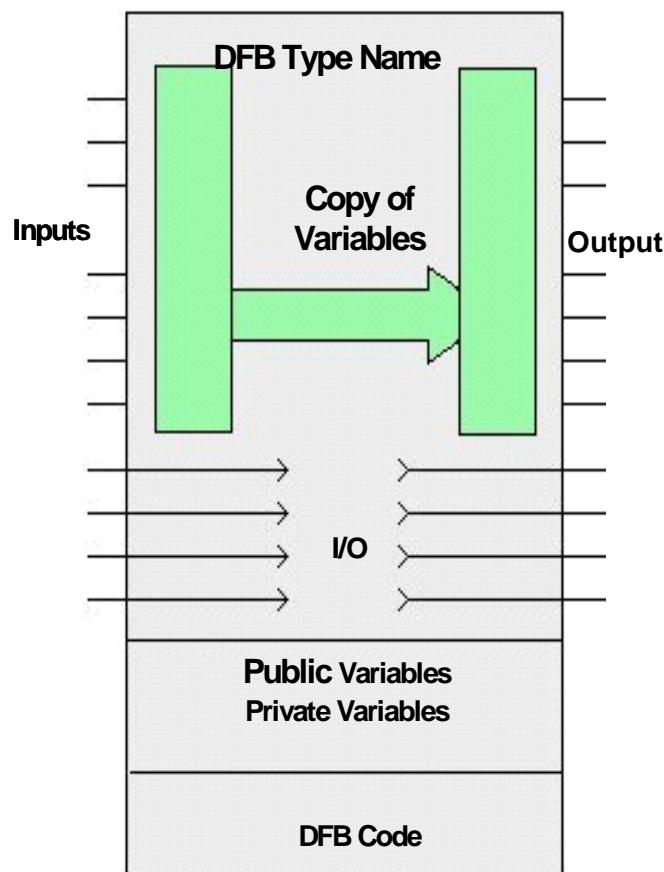
Privadas: não acessível pela aplicação

Código das seções: Escrito em **LD**, **IL**, **ST** ou **FBD**

Compatível com IEC para seção simples

Não compatível com IEC para múltiplas seções

Comentário: até 1024 caracteres sem formatação dos mesmos.

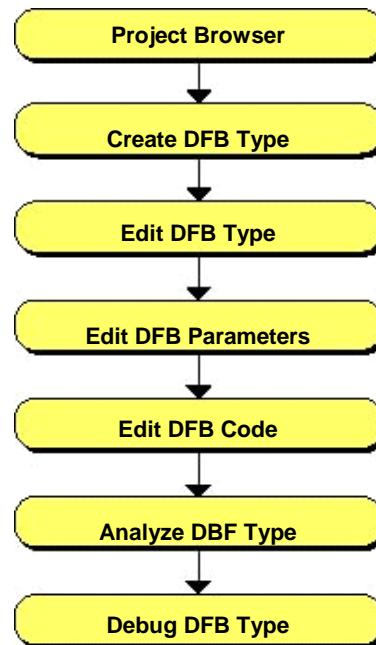


### Princípio de criação do DFB.

Abrir uma aplicação Unity Pro,

- Criar um novo DFB,
- Editar as propriedades do novo DFB o qual está sendo criado,
- Entrar com os parâmetros do DFB,
- Entrar com o código do DFB em uma ou mais seções,
- Analisar “analyze” o DFB validando-o,
- Depurar “debug” o DFB,
- Exportar o DFB, caso seja necessário para outra aplicação,
- Usar o DFB.

Fluxo de criação do DFB:



### Propriedades.

As propriedades do DFB são acessadas através da aba “DFB Types” no editor de dados, selecionar o DFB e clicar com botão direito e selecionar “properties”, será aberta a caixa com as informações do DFB tais como:

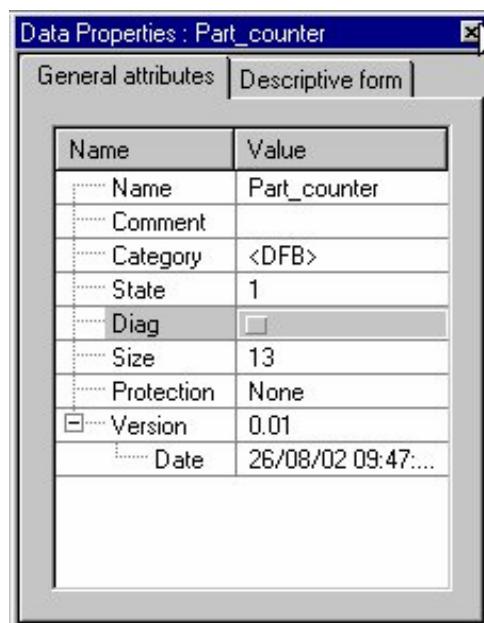
**Name,**

**Comments,**

**Protection**, para escolher o nível de proteção e senha (questionado automaticamente após a seleção),

**Diag**, para definir se o DFB será utilizado no diagnóstico do usuário.

Quando o DFB é protegido nos modos “Read only” ou “No Read & Write”, é possível a exportação do DFB, mas o arquivo.XDB” é codificado.



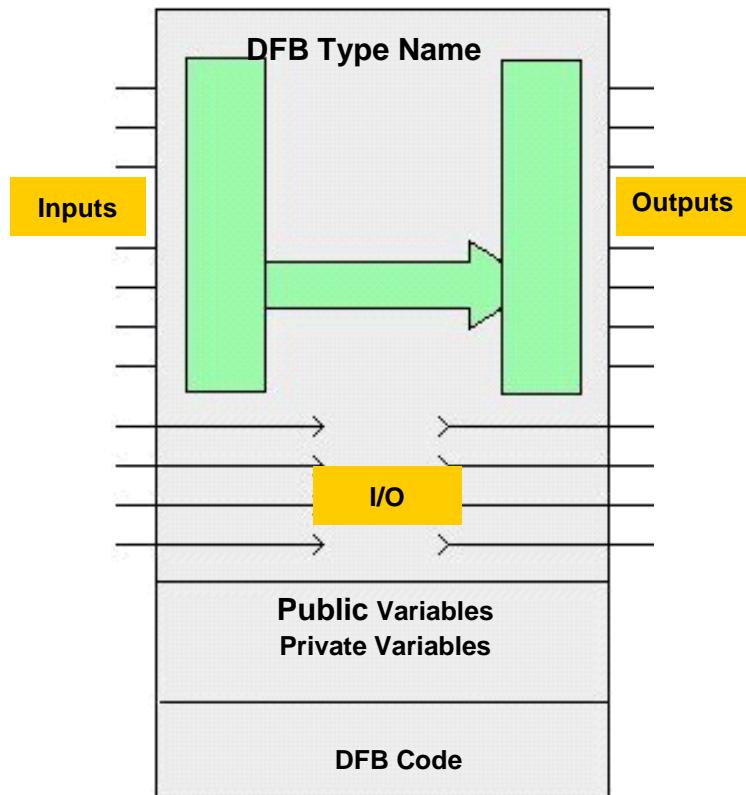
### I/O interface

Características da interface de I/O:

**Inputs:** Máx. 32

**Outputs:** Máx. 32 (tipo e nome no máx. 32 caracteres)

**Inputs/Outputs:** Máx. 32 (tipo e nome no máx. 32 caracteres)



Para cada parâmetro do DFB, as seguintes famílias de objetos podem ser utilizadas:

Object families	EDT	DDT (1)	IODDT	ANY_...(2)	ANY_ARRAY	FB
Inputs	Yes	Yes	No	Yes (4)	Yes	No
Inputs/outputs	Yes (3)	Yes	Yes	Yes (4)	Yes	No
Outputs	Yes	Yes	No	Yes (4)(5)	No	No

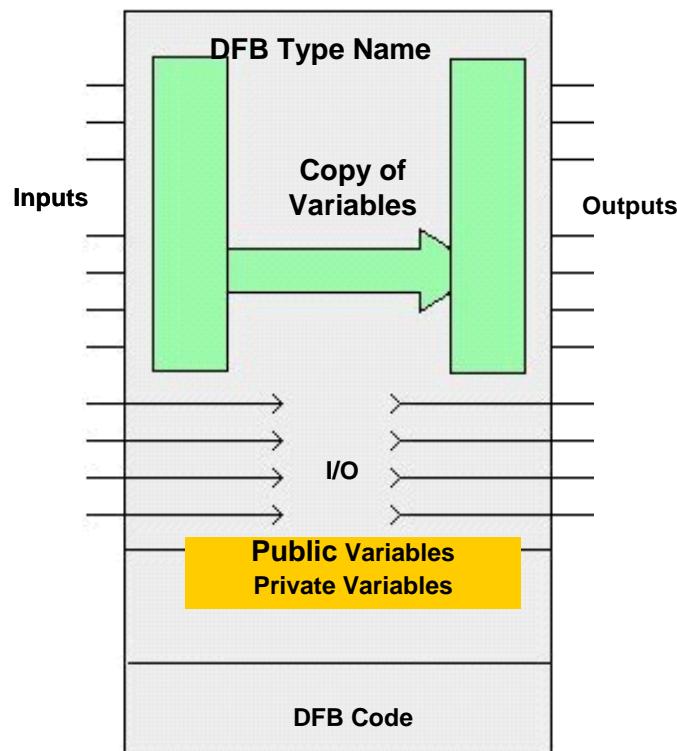
- (1) Família de dados Derived, exceto input/output derived data types (IODDT)
- (2) Todas as famílias de dados genéricos, exceto a família ANY\_ARRAY,
- (3) Exceto para tipo EBOOL- variáveis do tipo estática, com CLPs Quantum,
- (4) Exceto para variáveis do tipo BOOL e EBOOL,
- (5) Estes parâmetros não podem ser usados por chamadas de blocos de função.

## Variables

**Private**, são variáveis internas do DFB usadas somente para programar o DFB( por exemplo cálculos e lógicas intermediárias), o nome pode ter no máximo 8 caracteres.

**Public** são variáveis internas do DFB e pode ser acessada pela aplicação ou pelo usuário, o nome pode ter no máx. 8 caracteres.

Os valores da variável pública, modificando pelo programa ou por ajustes, podem ser salvos no lugar dos valores iniciais, pela colocação de %S94 em “1”.



Para cada parâmetro do DFB, as seguintes famílias de objetos podem ser utilizadas:

Object families	EDT	DDT (1)	IODDT	ANY_...(2)	ANY_ARRAY	FB
Public variables	Yes	Yes	No	No	No	No
Private variables	Yes	Yes	No	No	No	Yes

- (1) Família de dados Derivados, exceto input/output derived data types (IODDT)  
(2) Todas as famílias de dados genéricos, exceto a família ANY\_ARRAY,

### Código do DFB

O código é estruturado em seções, compatível com IEC para seções simples e não compatível para várias seções.

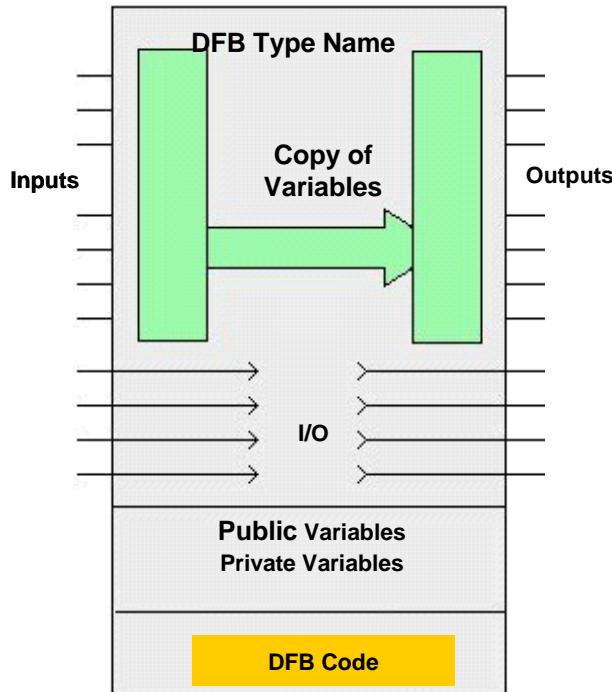
### Seções.

As seções são escritas em IL, LD e FBD, com nome simbólico de até 32 caracteres e proteção atribuída de “no protection”, “write protection” e “read/write protection”.

### Comentário

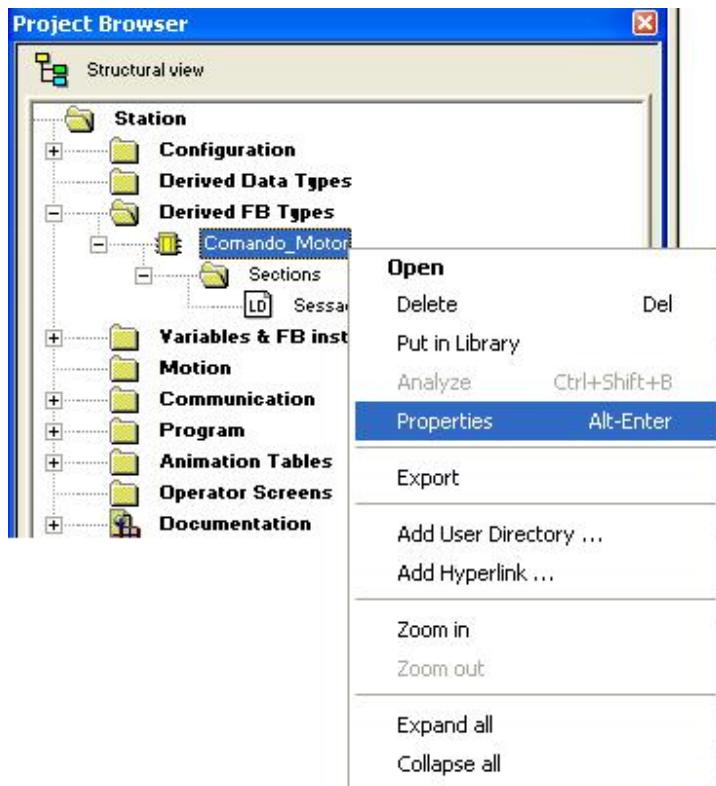
O comentário pode ter no máximo 256 caracteres.

Utilizar somente os parâmetros definidos para o bloco de função ou bits e words de sistema.

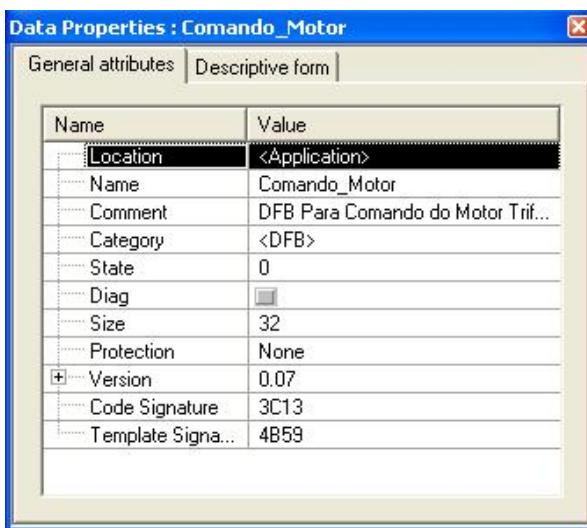


## Proteção do DFB

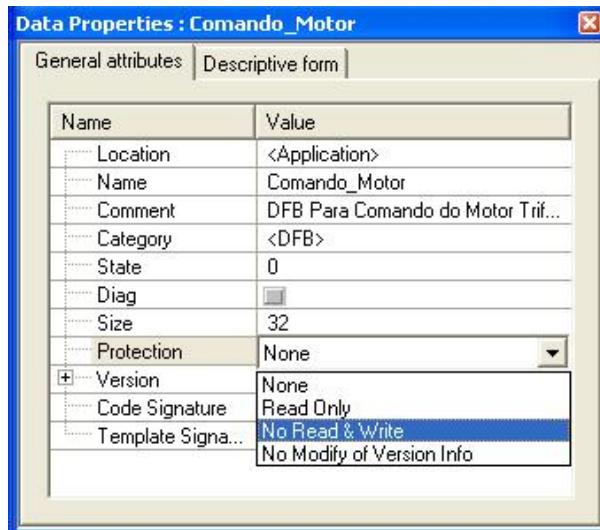
A proteção do DFB é ajustada na aba **Properties** acessada a partir da pasta do mesmo no Project Browser.



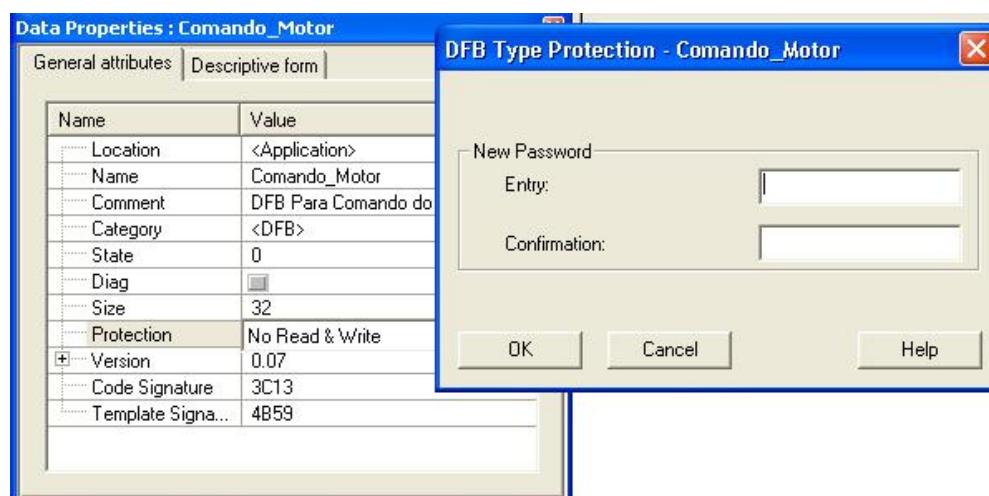
Ao selecionar **Properties**, é aberta a caixa de diálogo **Data Properties**: **Comando\_Motor** com as propriedades do mesmo.



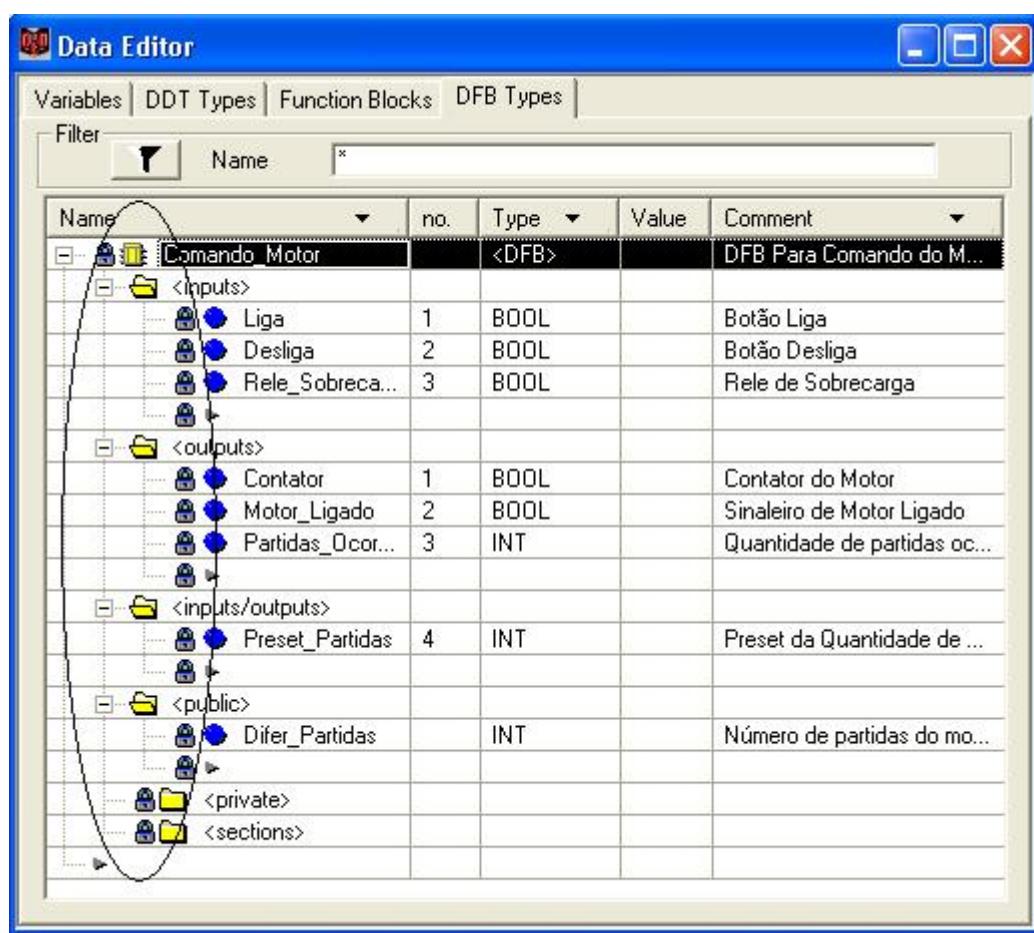
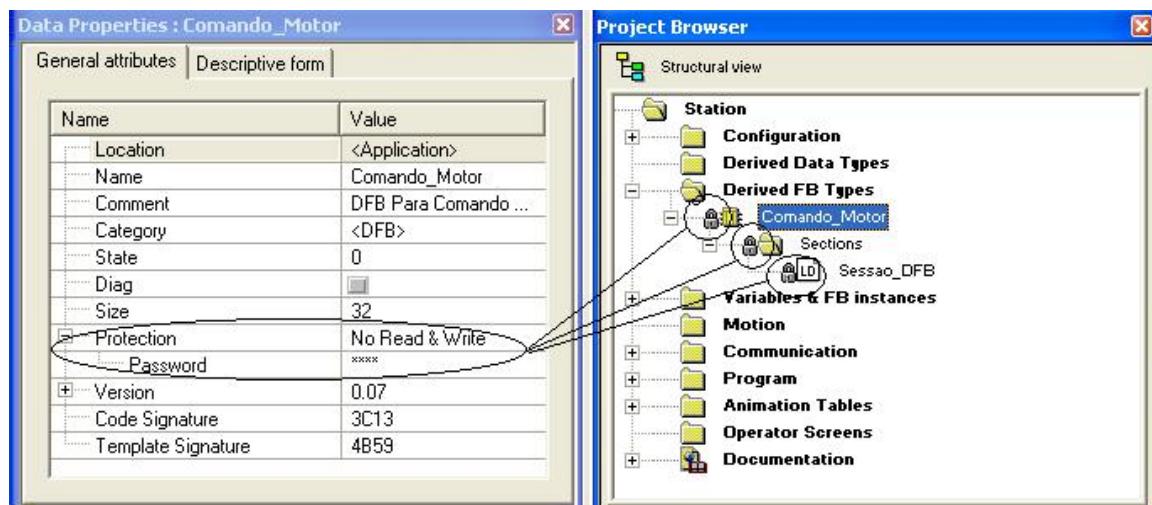
Ao clicar duas vezes na célula a frente de **Protection**, é aberta a barra de rolagem com as opções de proteção.



Ao selecionar o nível de proteção desejada, é aberta a caixa **DFB Type Protection - Comando\_Motor** para configuração da senha.



Ao concluir a configuração da proteção, o **Project Browser**, a caixa **Data Properties** e o editor de dados **Data Editor** indicam a existência da proteção.

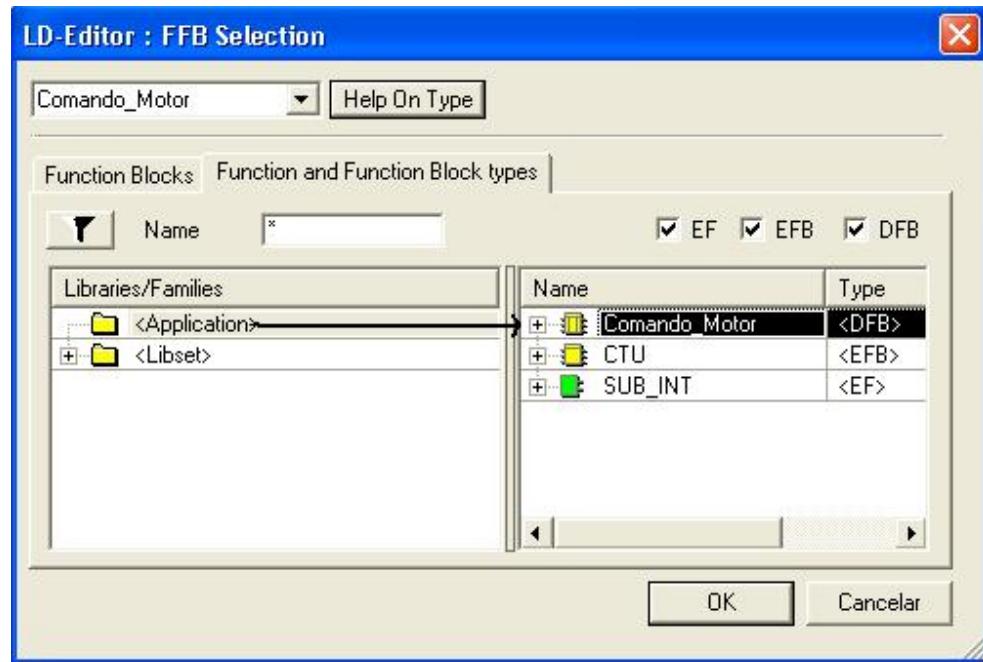


Para o nível de proteção “**No Read & Write**”, não é permitido à uma pessoa não autorizada verificar a lógica do mesmo, apresentando a seguinte mensagem.

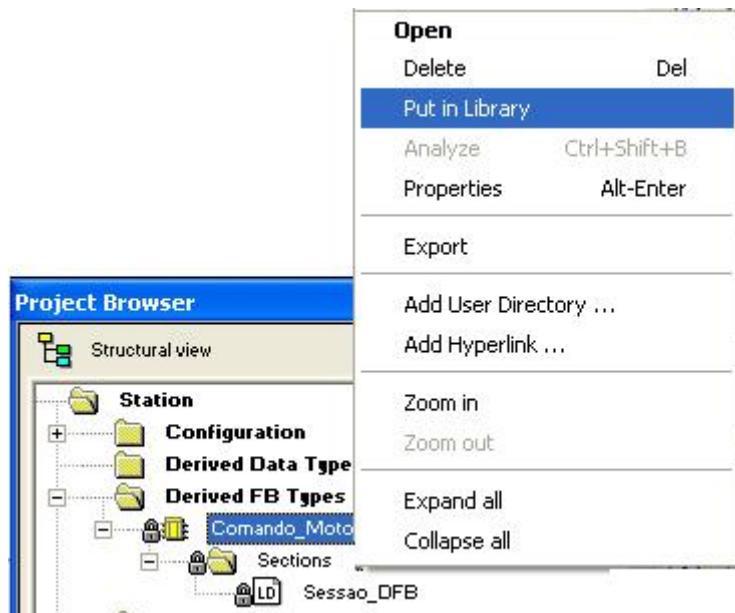


## Armazenamento do DFB

Ao criar um DFB, o mesmo é salvo na biblioteca local (Biblioteca da aplicação), embora seja possível disponibilizá-lo de maneira global na biblioteca **Libset**, disponibilizando-o para outras aplicações.



Para disponibilizar um DFB de maneira global na biblioteca **Libset**, deve-se a partir da pasta do mesmo no **Project Browser**, clicar com o botão direito e selecionar **Put in Library**.



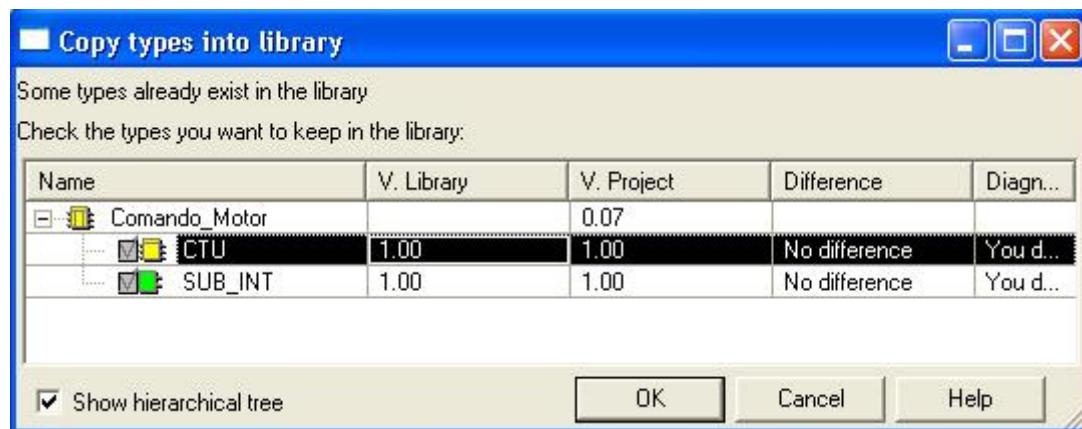
Ao selecionar **Put in Library**, é aberta a caixa **Copy type to library**, onde deve-se expandir a pasta **Custom Lib**.



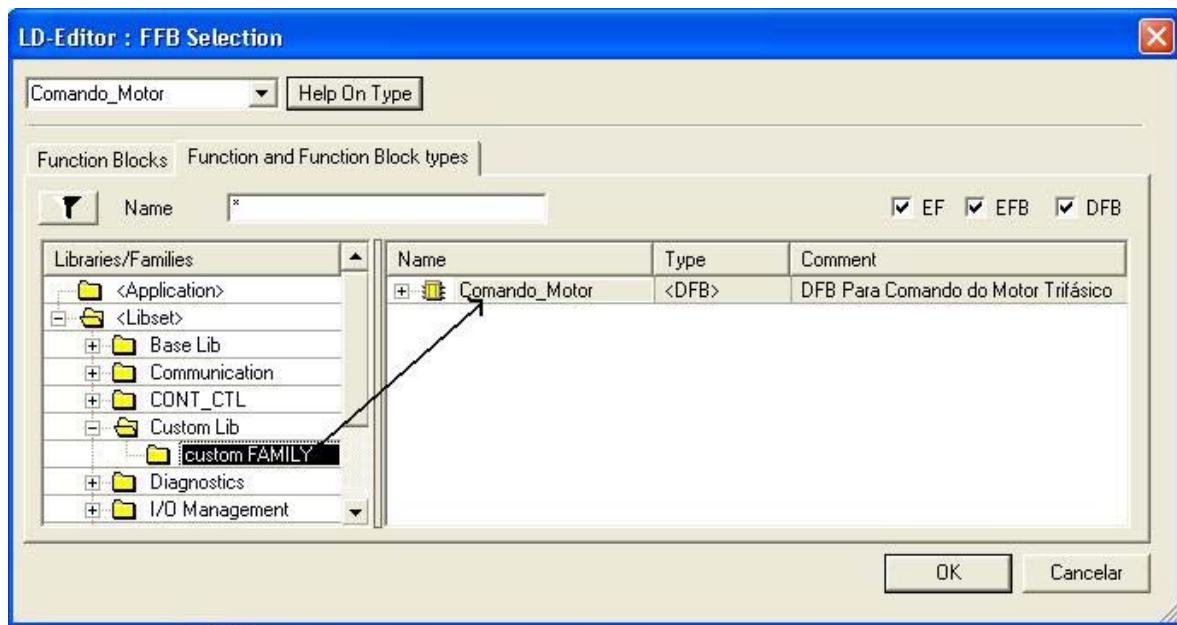
Com a pasta **Custom Lib** expandida, selecionar **custom FAMILY**.



Caso exista algum elemento semelhante na biblioteca, o sistema questiona sobre a manutenção dos mesmos na biblioteca.



Ao acionar o botão **OK**, o DFB é armazenado na biblioteca global **Libset**.



## Aplicação De DFB Para Controle De Partida Direta De Motor De Indução Trifásico.

O Bloco DFB conterá:

Entradas:

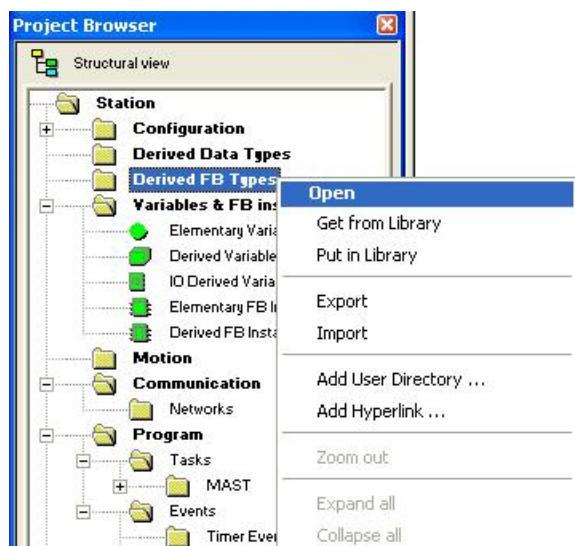
- Para o botão liga,
- Para o botão desliga,
- Para o relé de sobrecarga.

Saídas:

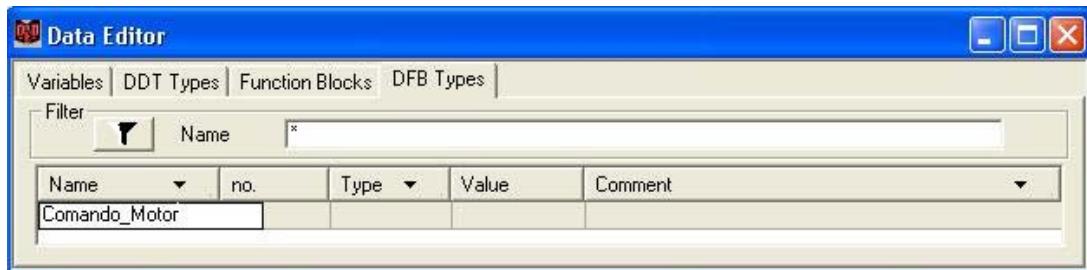
- Para o contator do motor,
- Para o sinaleiro de motor ligado,
- Indicação da quantidade de partidas,
- Interface do preset de partidas,
- Variável Pública com a diferença entre a quantidade de partidas ocorridas e o Valor de preset de partidas.

### Criação do DFB.

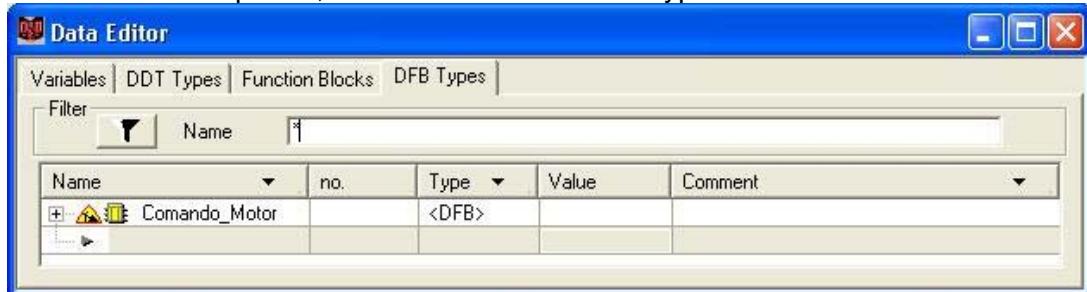
O primeiro passo é criar o “**DFB Type**” através do project browser na pasta **Derived FB Types**, clicando com botão direito sobre a mesma e selecionar **Open**.



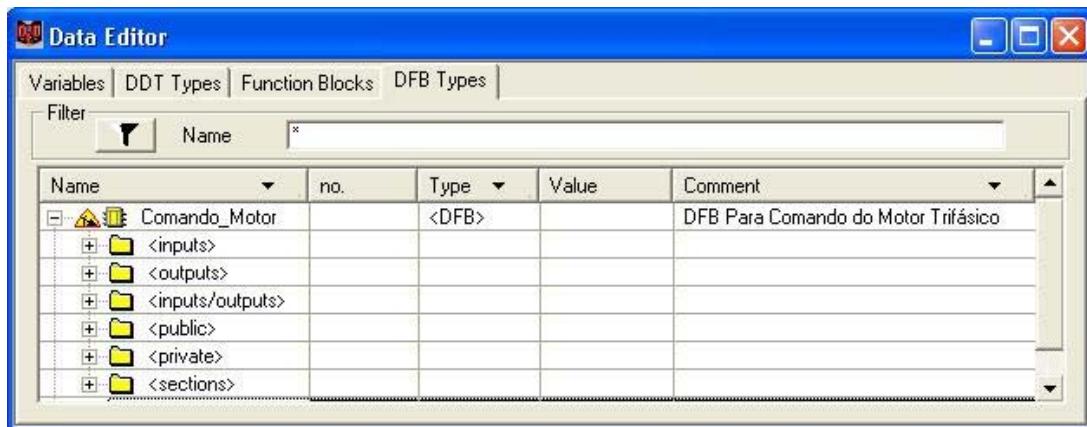
Ao selecionar **Open**, é aberto o editor de dados **Data Editor**, onde na coluna “Nome” deve-se especificar o nome do DFB, e acionar o botão esquerdo.



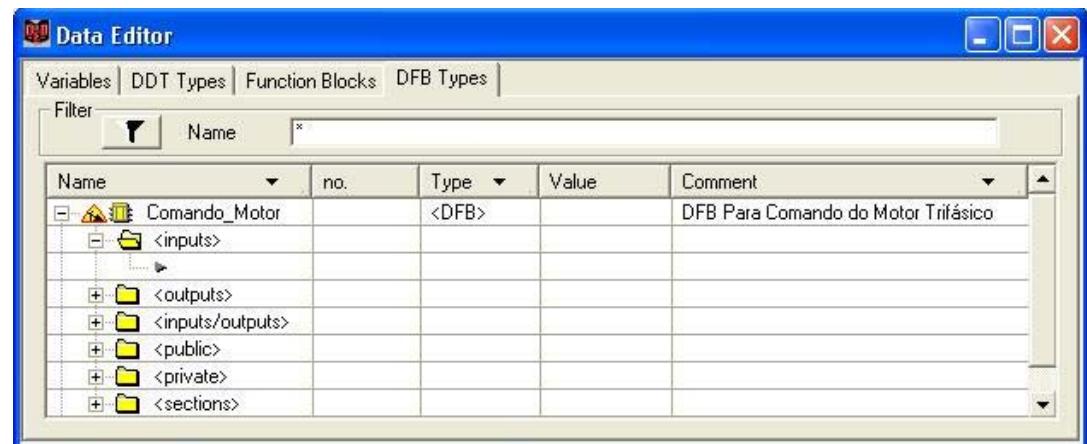
Ao açãoar o botão esquerdo, é visualizado na coluna Type <DFB>.



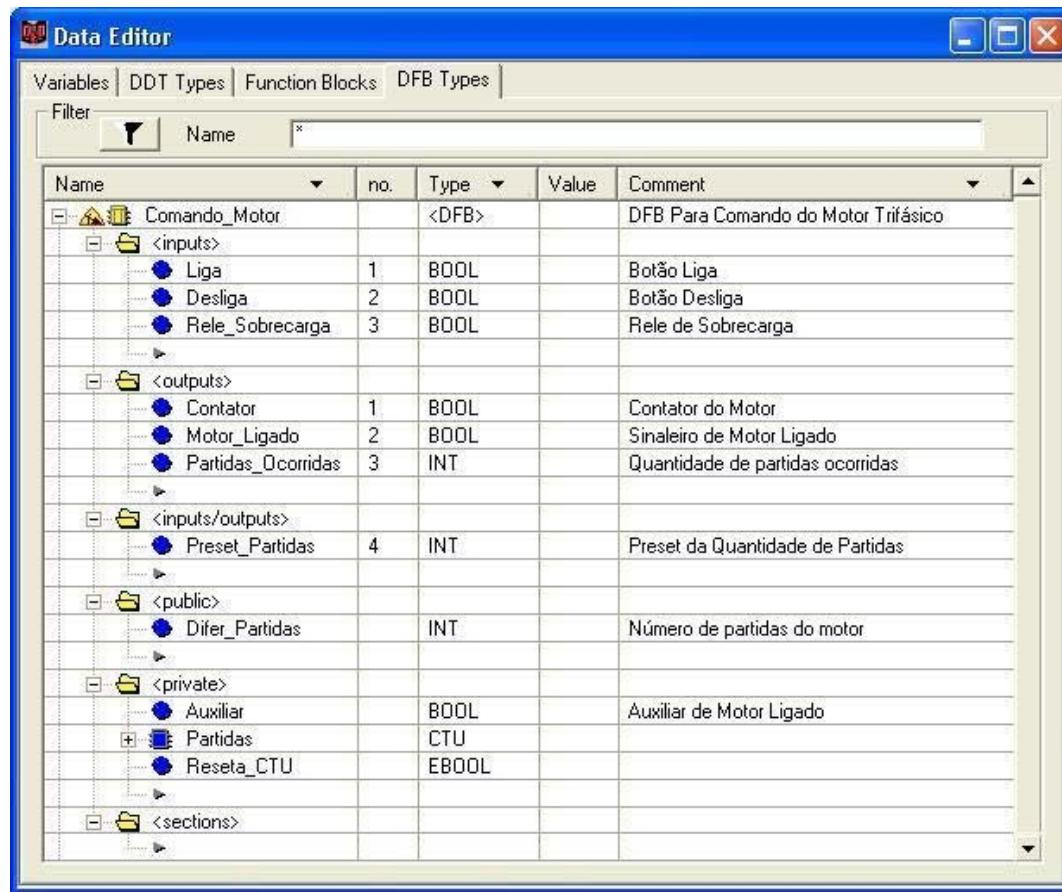
Expandindo a pasta com o nome do DFB, na coluna “Name” são visualizadas as pastas com parâmetros do DFB.



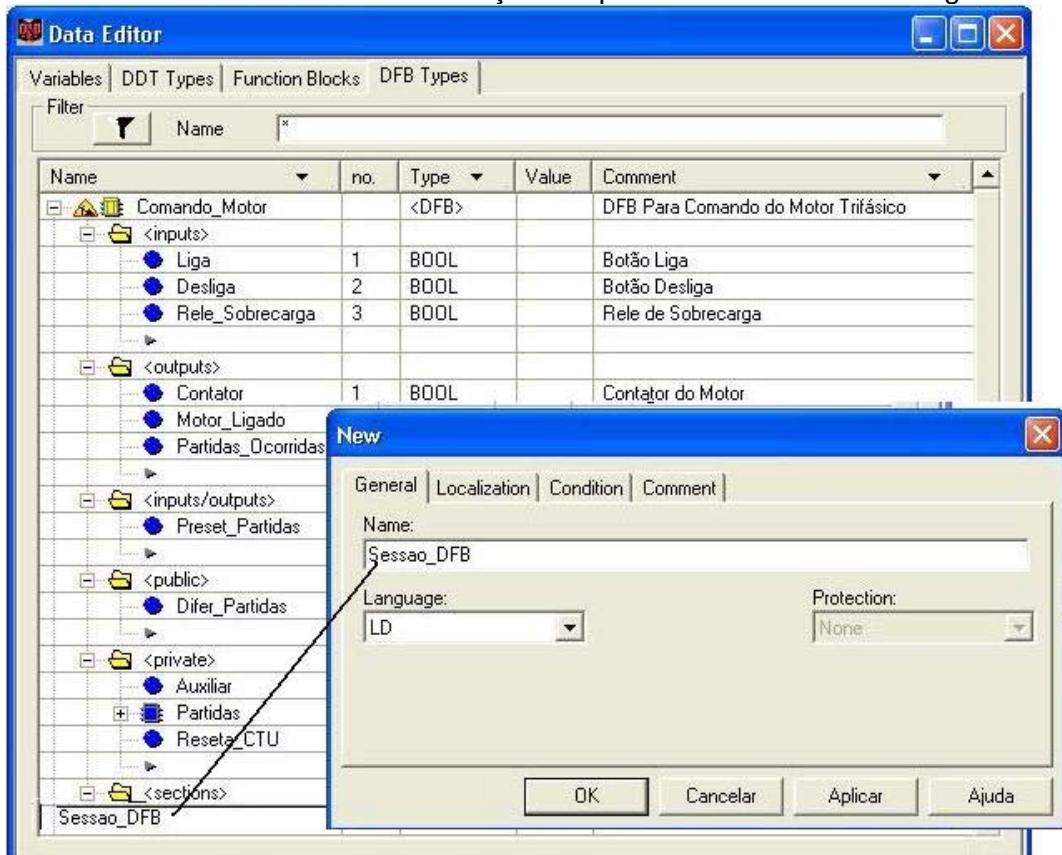
Deve-se expandir cada uma das pastas de parâmetros e especificar abaixo da mesma, as variáveis de acordo com a necessidade da aplicação.



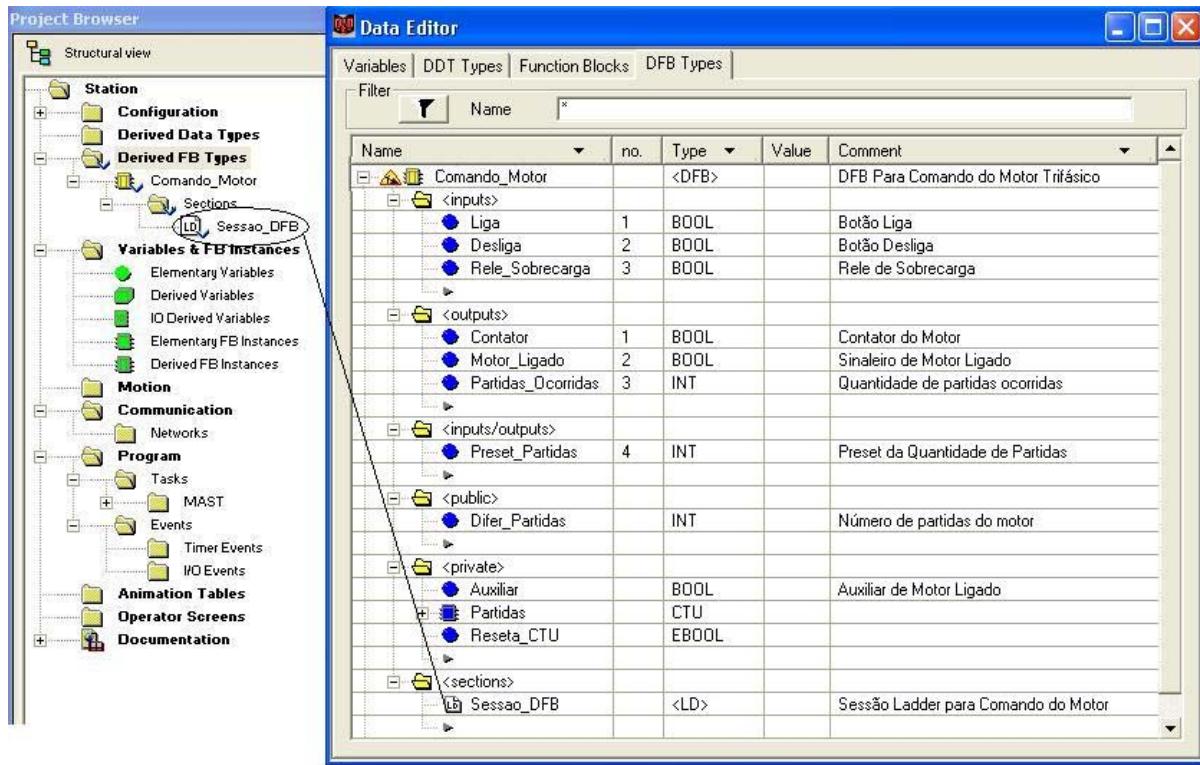
Declaração das variáveis.



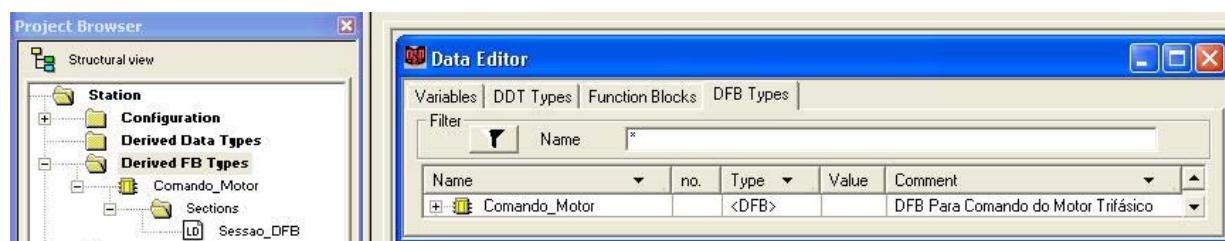
Na pasta <sections> deve ser definida a seção na qual será desenvolvida a lógica do DFB.



Ao acionar o botão **OK** na caixa de diálogo de criação da seção, é criada na pasta **Derived FB Types** a seção destinada ao **DFB Type**.

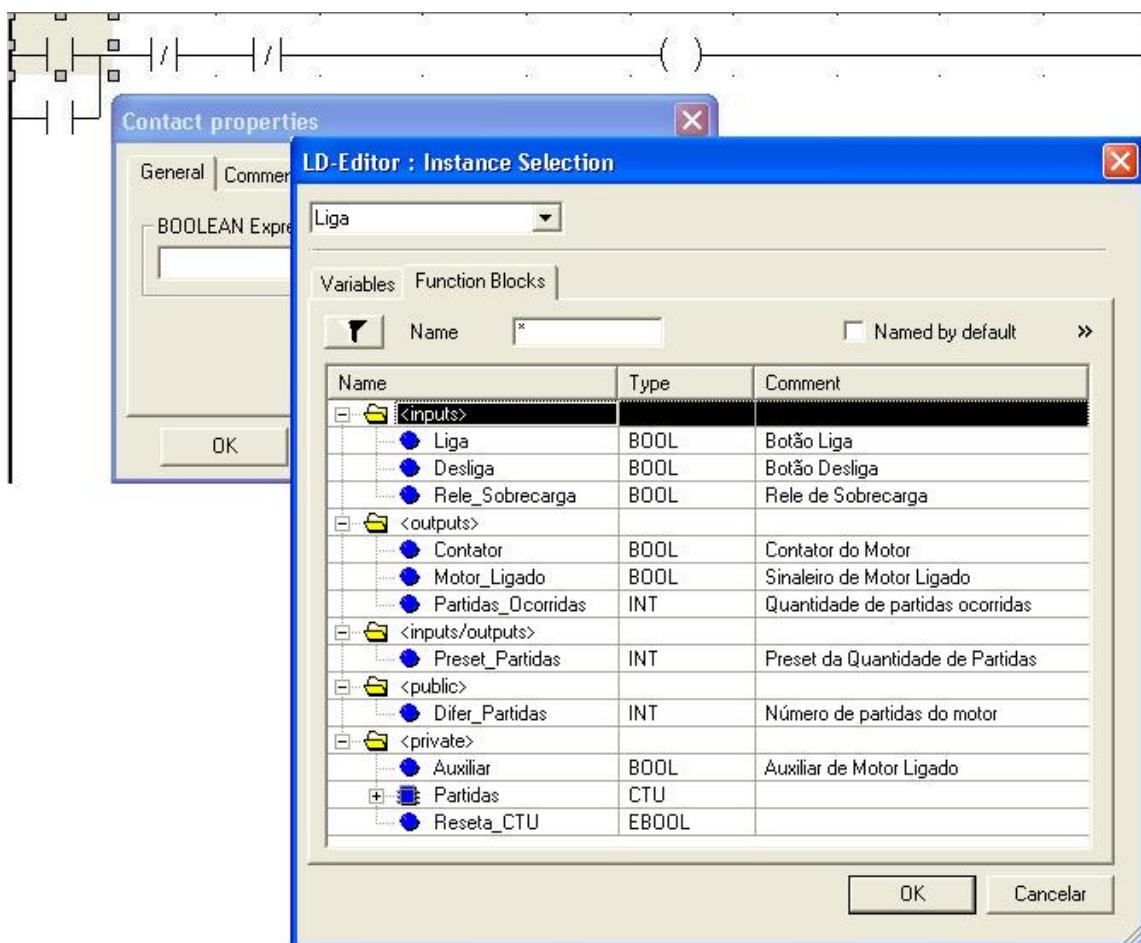


Para concluir a criação do **DFB Types** é necessário fazer a compilação **Build**, validando assim o mesmo.



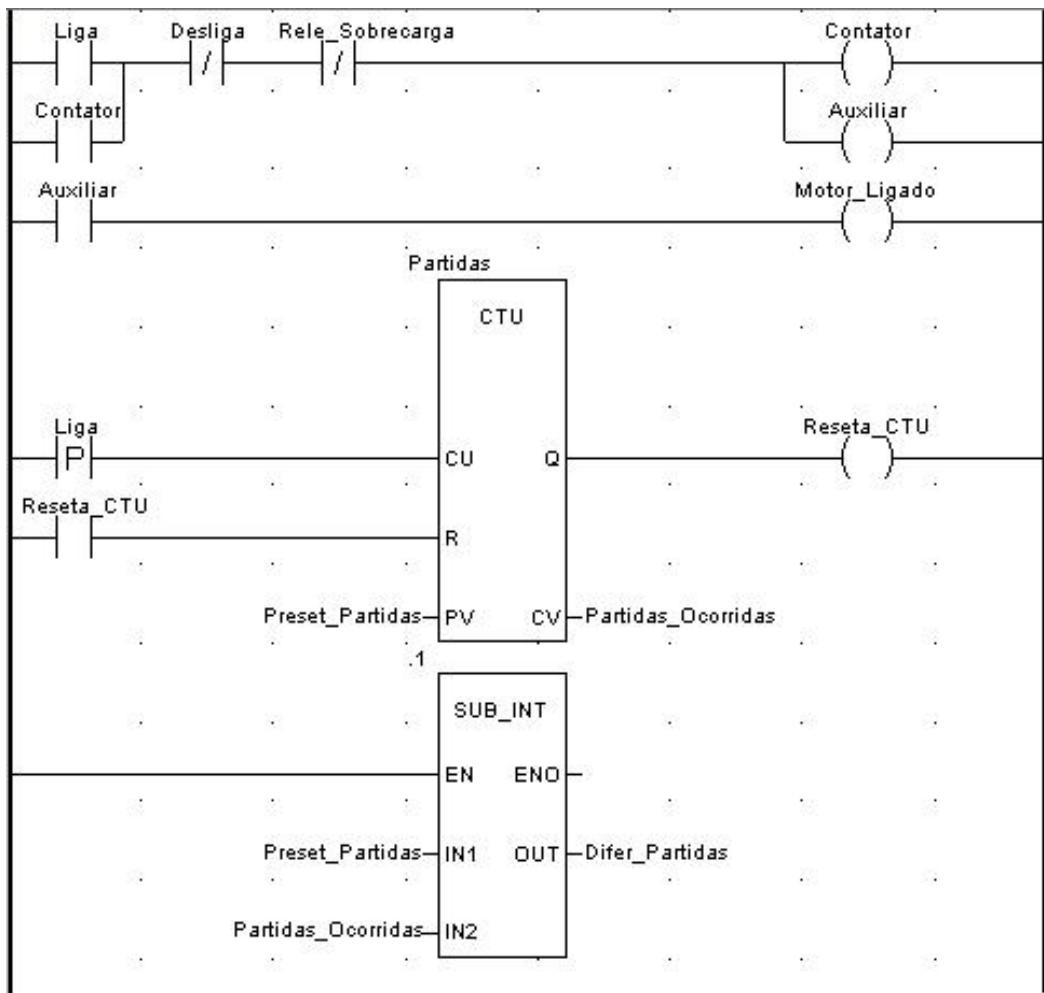
## Edição da lógica do DFB

Ao definir a variável do objeto, é aberto editor de dado **LD-Editor: Instance selection** com as variáveis criadas especificamente para o DFB.



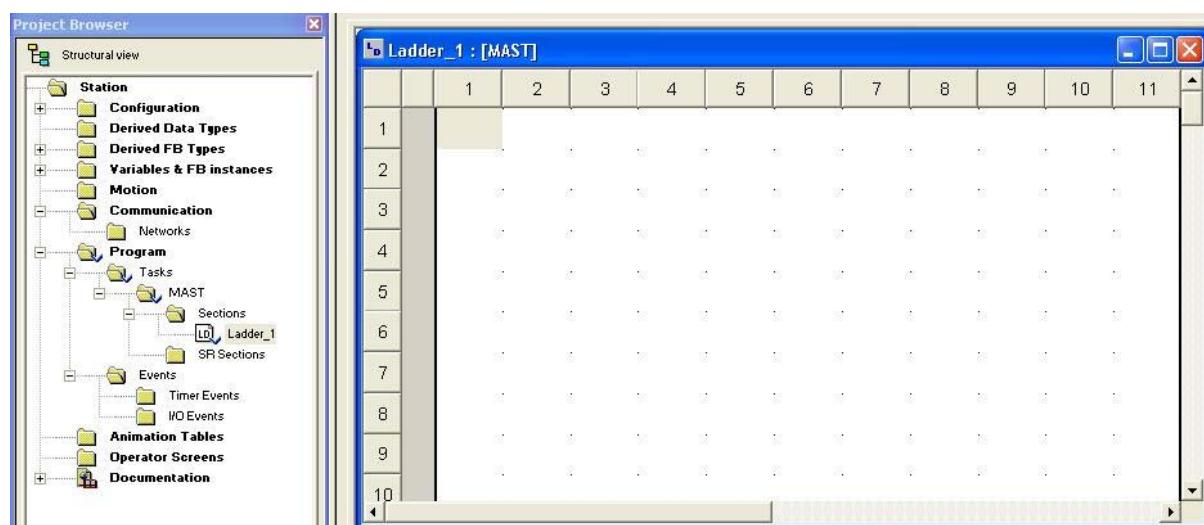
Durante a edição da lógica do DFB, as variáveis do mesmo são atualizadas automaticamente.

### Solução da Lógica do DFB.



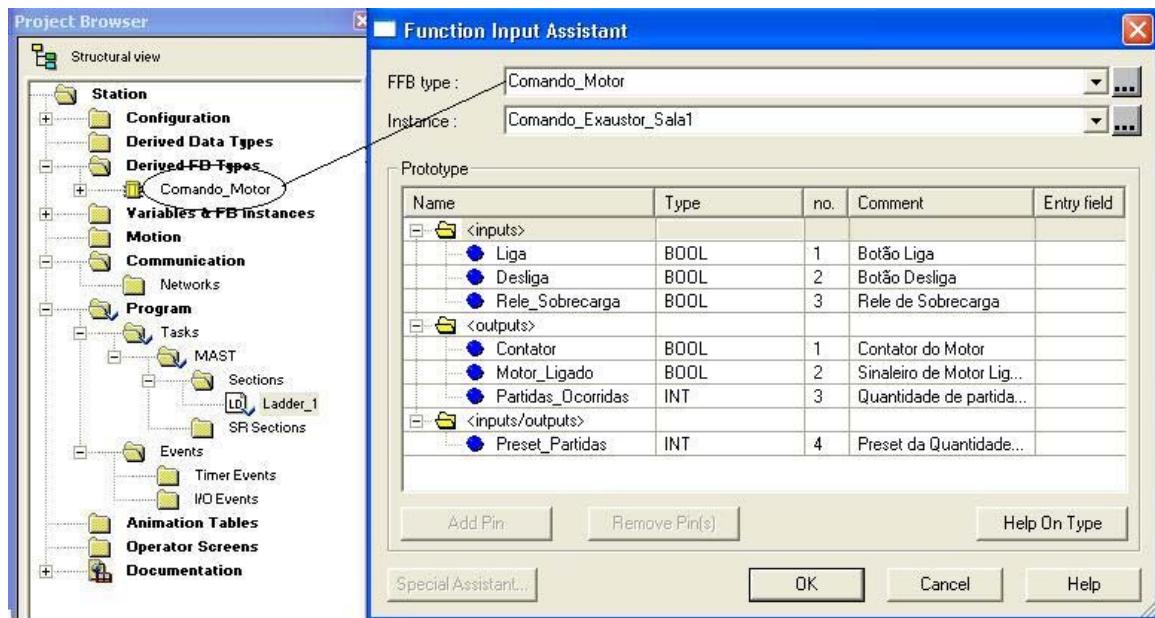
### Utilização do DFB.

Com a seção da lógica da aplicação aberta, clicar no ícone “**FFB Input Assistant**” na barra de ferramenta.



Ao clicar no ícone “**FFB Input Assistant**” é aberta a caixa de diálogo “**Function Input Assistant**”, onde no campo “**FFB type**” deve ser especificado o DFB desejado, o qual será localizado pelo sistema na **biblioteca local da aplicação**, devendo agora ser definido o nome da instância no campo “**Instance**” e clicar no botão **OK**.

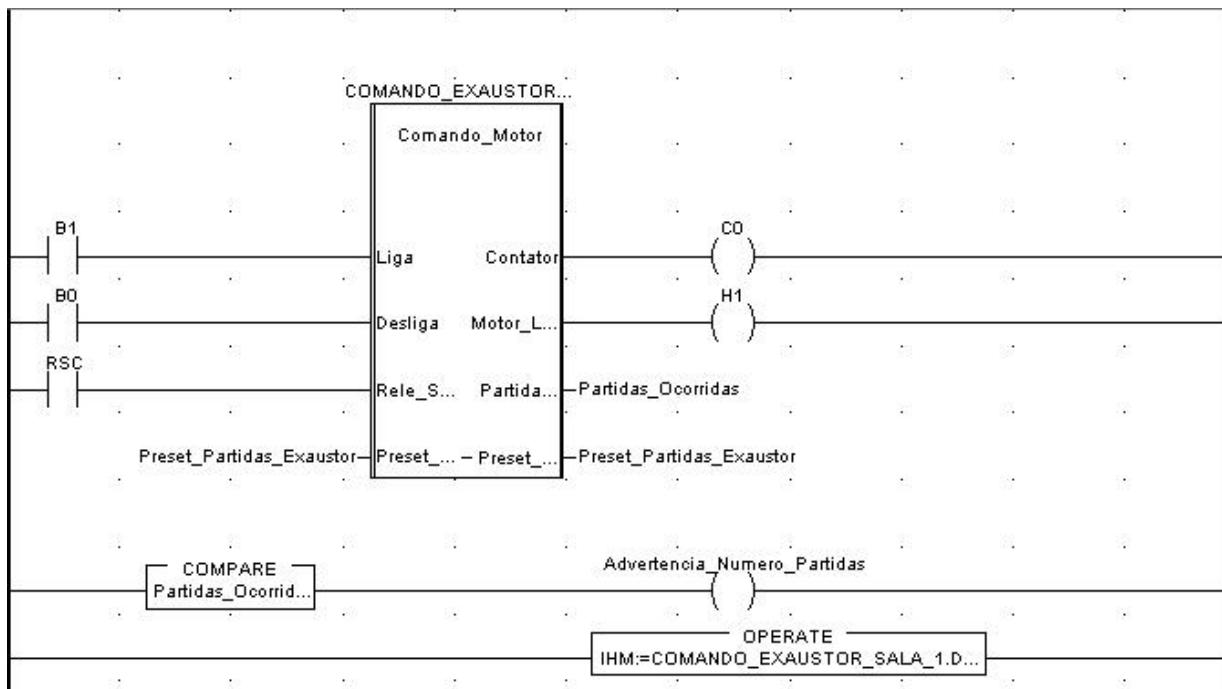
## Bloco de Função Derivado



Ao clicar no botão **OK**, o objeto deve ser colocado na área de edição clicando no botão direito do mouse.

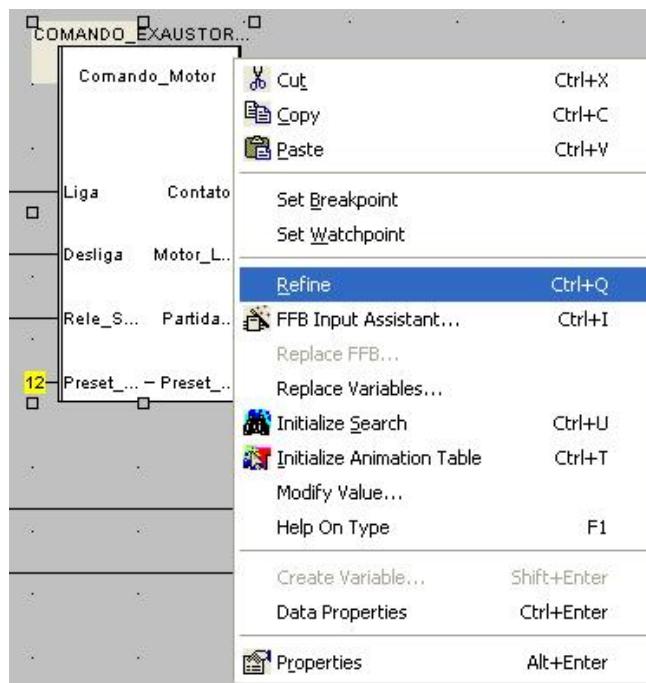


Com o DFB colocado na área de edição, realizar a lógica para atender os requisitos da aplicação.

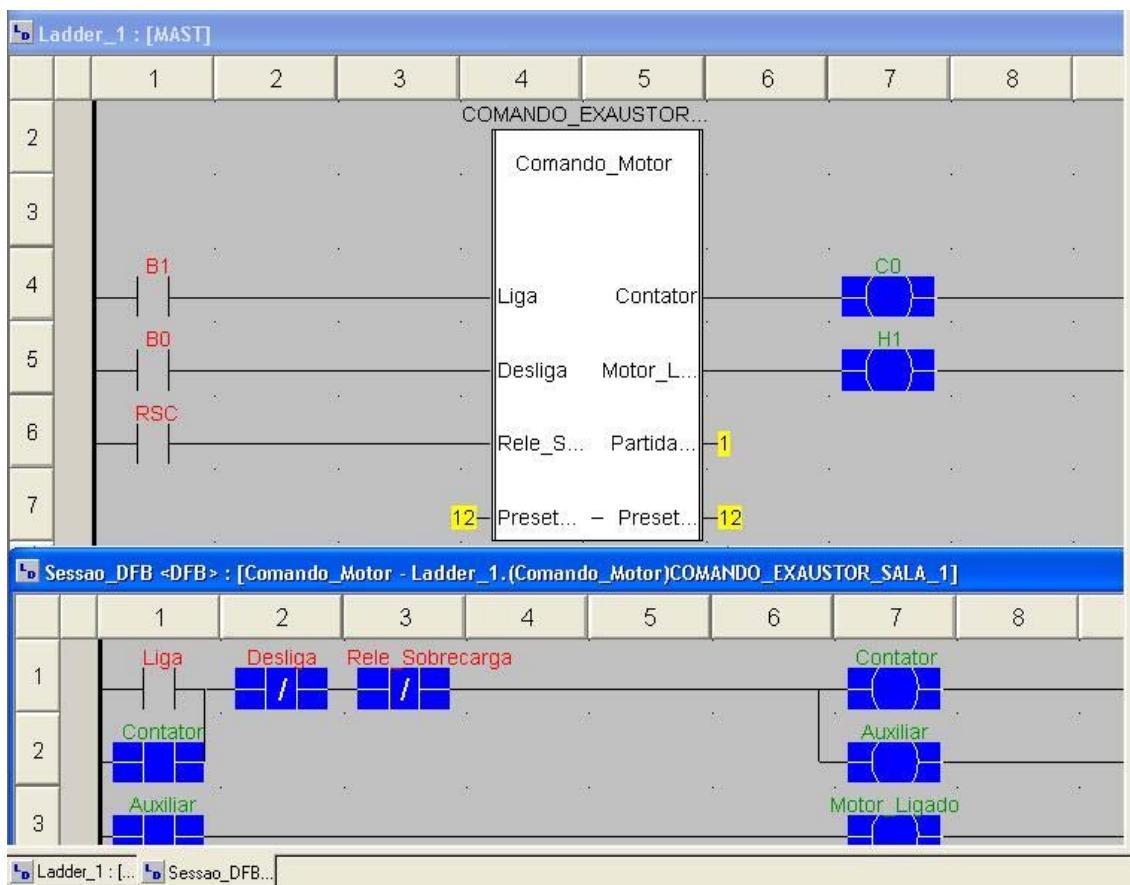


## Depuração do DFB

A verificação do funcionamento e depuração da aplicação é feita da mesma maneira vista anteriormente no **Capítulo Teste da Aplicação**, sendo que ao clicar com o botão direito sobre o DFB e selecionar “refine” é possível verificar simultaneamente o funcionamento da lógica do DFB assim como da lógica da aplicação.



Visualização simultânea das lógicas da aplicação e do DFB.





# **CAPÍTULO 14**

## **Telas de Operação – Operator Screens**

---



# Telas de Operação – Operator Screens

## Características da Tela de Operação

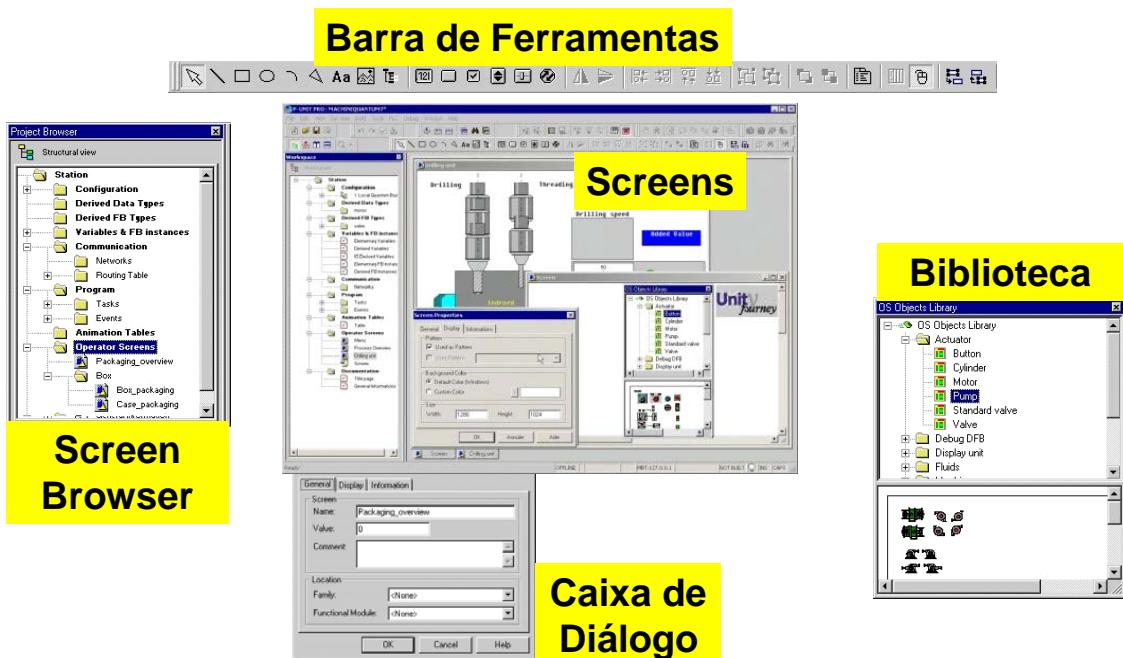
Ferramenta de operação customizada atendendo ao primeiro e segundo nível de funções de diagnóstico sendo para controle (mostrando em tempo real o estado de uma máquina/processo) e monitoração (Telas de Run-time customizadas para atender a necessidade do operador).

Recursos que facilitam a manutenção e diagnóstico através de procura interativa no caso de falha no programa via sinalização em uma tela de Run-time.

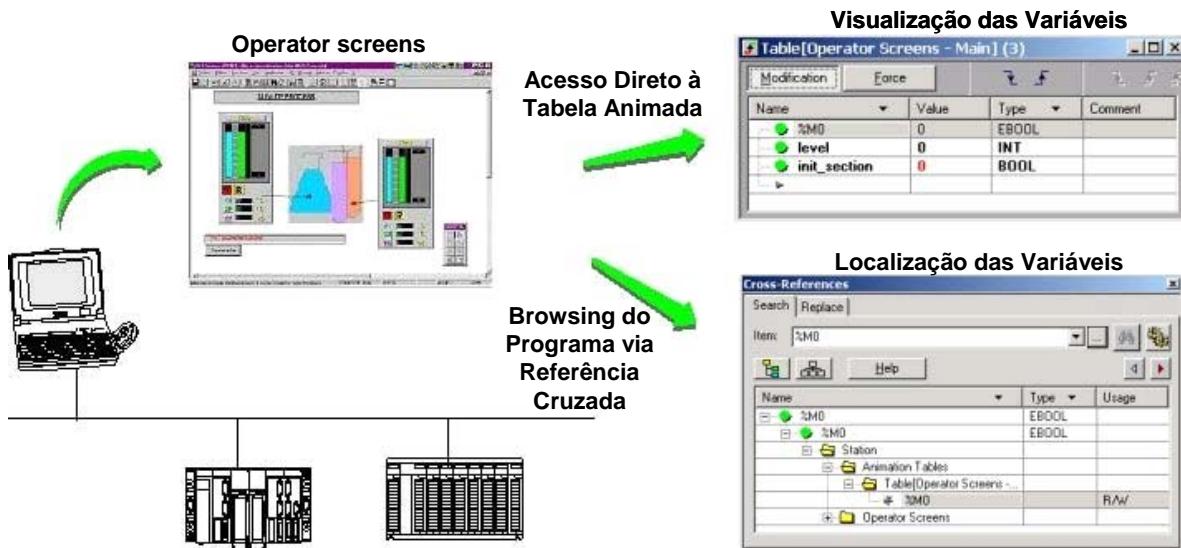
As telas de Run-time estão no terminal, não no CLP.

### Características Gerais

Devido a interatividade entre Barra de ferramentas, browser, biblioteca e caixas de diálogos, as telas de operação “**Operator Screens**”, facilitam a criação de telas e de animações.



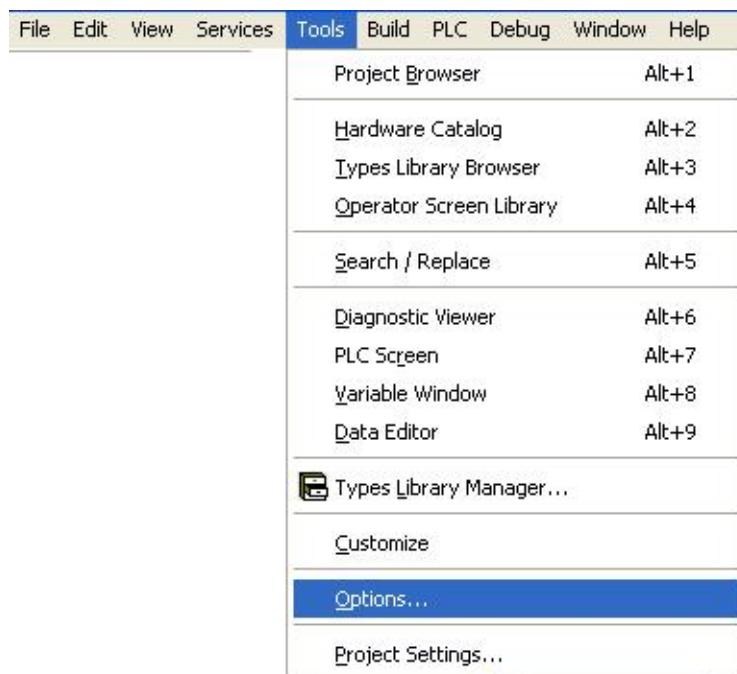
## Funções de telas integradas



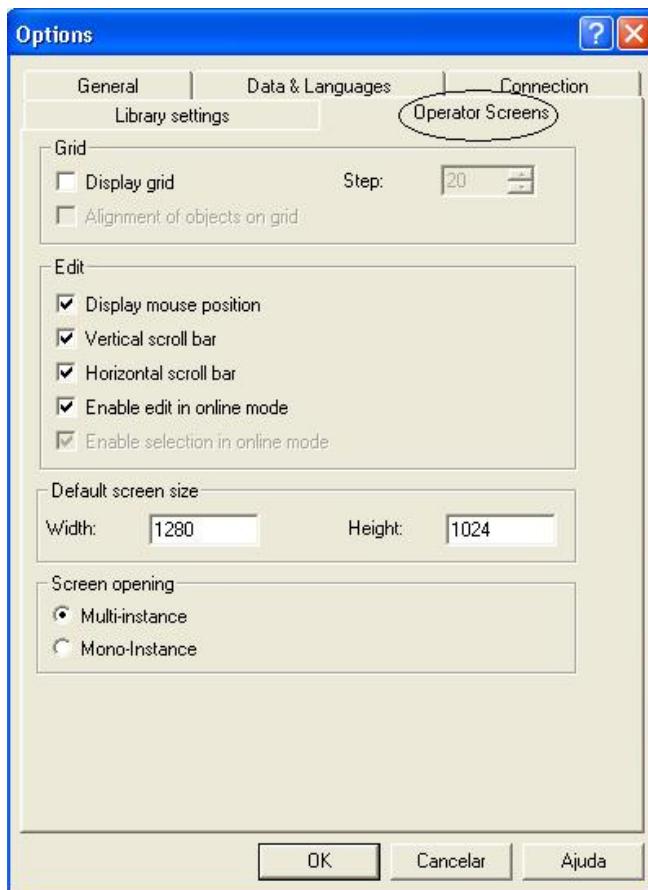
## Criação de Aplicação de Tela de Operação.

### Configuração das Opções “Options” da Tela de Operação.

Com a aplicação aberta, configurar as opções “Options” da tela de operação. A partir da barra de ferramentas **Tools**, selecionar **Options**.



Ao selecionar “Options”, é aberta a caixa de diálogo “Options” a aba “Operator Screens” é utilizada para configurar grade, funções de edição e edição no modo online. Estas opções são salvas no Windows e são comuns a todas as aplicações.



**Display grid:** mostra a grade de acordo com o passo ajustado

**Step:** ajusta o espaço entre as linhas horizontais e verticais da grade

**Alignment of objects on grid:** quando esta caixa é selecionada, a criação, movimentação e modificação dos objetos são automaticamente posicionados nos pontos da grade.

**Display mouse position:** Quando esta caixa é selecionada, a posição do mouse é mostrada no barra de status.

**Vertical scroll bar:** a barra de rolagem é mostrada a vertical

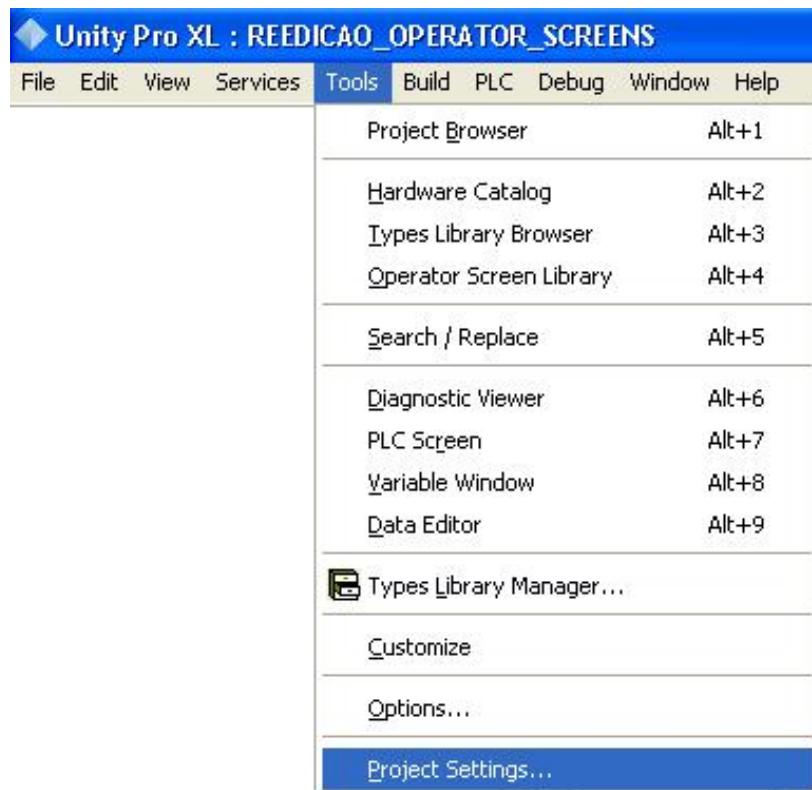
**Horizontal scroll bar:** a barra de rolagem é mostrada a horizontal.

**Edit in online mode:** possibilita a criação, modificação e apagamento de uma tela no modo online.

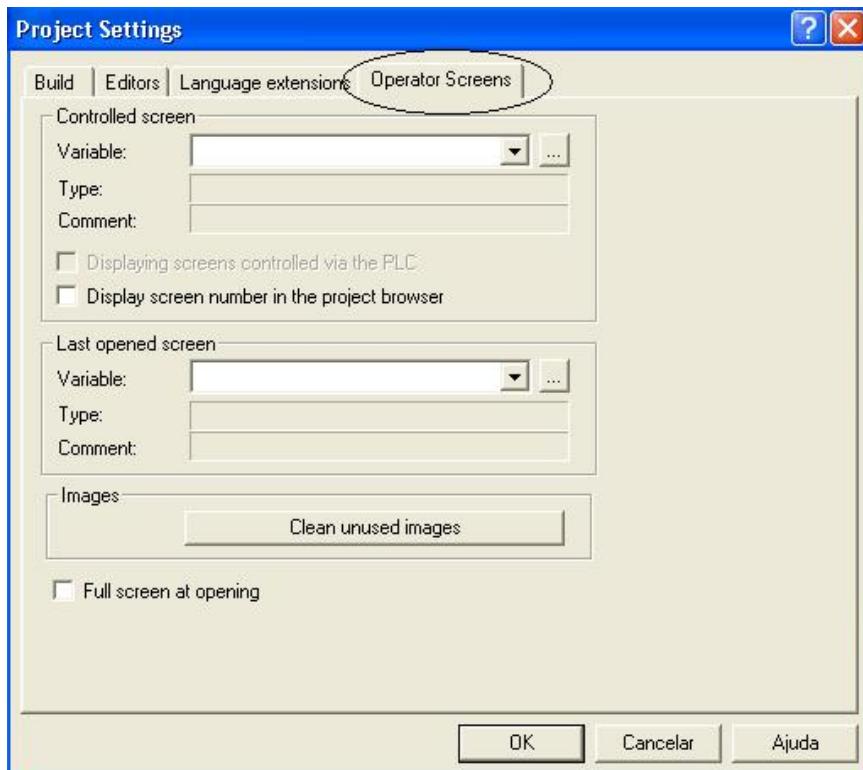
**Selection in online mode:** possibilita a seleção de um objeto no modo online.

**Default screen size:** define valores default para todas as telas novas, não afetando as telas existentes.

**Configuração dos ajustes do projeto “Project settings” da Tela de Operação.**  
Com a aplicação aberta, acessar “Project Settings” via barra de ferramentas **Tools**.



Ao selecionar Project Settings, é aberta a caixa de diálogo “Project Settings” e na aba “**Operator Screens**”, configura-se o controle de telas via programa da aplicação, detectar a tela mostrada e gerenciar toda a tela aberta. Estes ajustes são salvos na aplicação corrente.



**Variable** (Controlled screen): a variável (bit, byte, word ou dupla word), a qual permite que uma determinada tela seja mostrada no modo online diretamente pelo CLP.

**Type:** Tipo da variável.

**Comment:** Comentário da variável.

**Display screens controlled via the PLC;** Quando esta caixa é selecionada no mood online, o CLP pode controlar a amostragem da tela pela modificação da variável de controle.

**Display screen number in project browser:** Quando esta caixa é selecionada, o número da tela é mostrada à esquerda do nome da tela no application browser e na barra de título da tela (se a tela estiver aberta).

**Last opened screen:** A variável que contém o número da última tela mostrada( pelo usuário ou pelo CLP (através da variável de controle).

**Type:** Tipo da variável prévia.

**Full screen:** Quando esta caixa é selecionada, o editor de tela de operação opera no modo de tela cheia (a barra de menu é escondida). Para retornar ao modo de janela, acionar a tecla Escape.

**Images:** O botão permite que o usuário limpe a área de trabalho de todas as imagens não utilizadas.

### Criação de Telas.

A partir do Project Browser, selecionar a pasta **Operator Screens** e com o botão direito selecionar **New screen**.

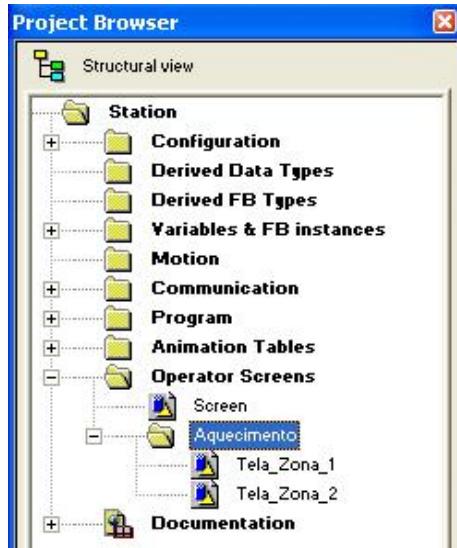


Tela criada:



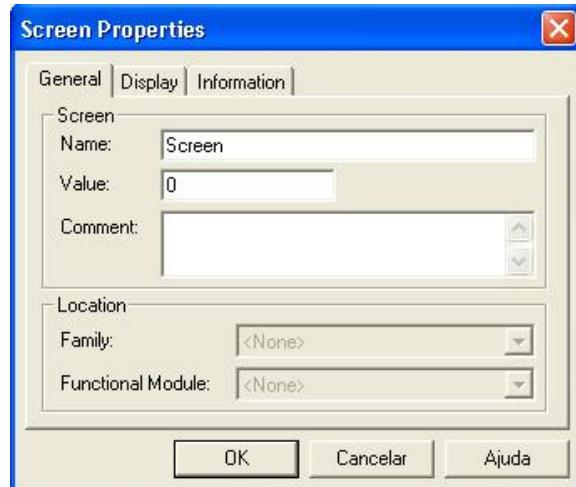
Em uma aplicação, podem ser necessárias várias telas, e estas aplicadas a uma determinada parte do processo, existindo assim uma relação entre as mesmas, a esta relação da-se o nome de família “Family”, a qual é criada a partir do Project Browser, selecionar a pasta **Operator Screens** e com o botão direito selecionar **New Family**.

Telas criadas em uma família “Aquecimento”.



Ao selecionar **New Screen**, a caixa de diálogo “**Screen Properties**” é aberta contendo as abas:

**General:** Modificação de nome, valores, comentário e localização.



Name: Nome da tela pode ser modificado e pode ser composto de no máximo 255 caracteres.

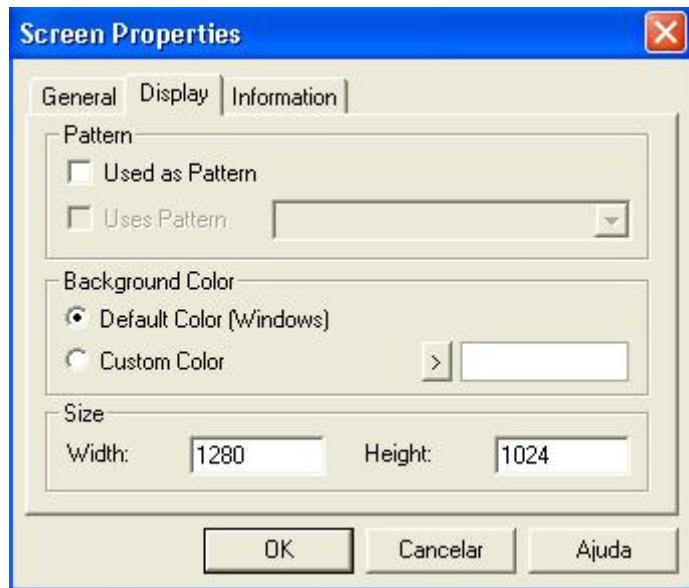
Value: Identificador de tela, pode ser modificado e usado quando se deseja associar um browser ou um botão a tela ou se desejar permitir o controle da tela pelo CLP no caso de gerenciamento do controle da tela pelo operador. Um valor é obrigatório. Por default o valor é 1 na primeira tela criada e é incrementado a cada tela criada.

Comment: Comentário da tela.

Family: Família da tela <none> por default. Para associar uma tela a uma família, selecionar a família na barra de rolagem no campo Family.

Functional Module: Módulo funcional da tela <none> por default. Para anexar uma tela a um módulo funcional, selecionar o módulo funcional na barra de rolagem no campo Functional Module.

**Display:** Modificação de recursos da tela.



**Used as Pattern:** quando esta caixa é selecionada, a tela é usada como um modelo. Neste caso, pode ser usada por qualquer tela (exceto por um modelo). Se um objeto animado é criado na tela, esta não pode ser usada como modelo, se desejar, deve-se suprimir todas as animações da tela.

**Uses Pattern:** Quando seta caixa é selecionada, a tela usa a tela modelo mostrada na barra de rolagem no campo de Uses Pattern. O modelo é mostrado como um fundo estático na tela, os objetos do modelo não podem ser selecionados na tela.

**Background color:** definição da cor de fundo da tela.

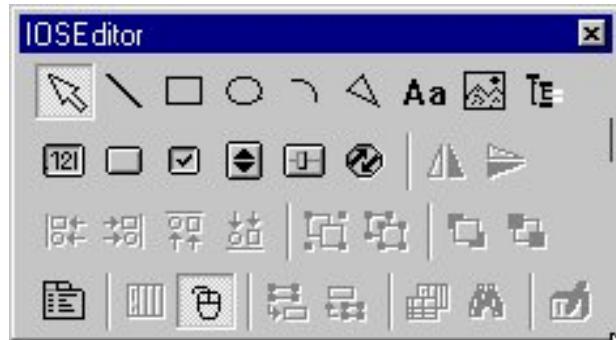
**Size:** Este é o tamanho da tela, que pode ser modificado e é por default é fixo em 1280 por 1024 pixels.

**Information:** Estas informações não podem ser modificadas e indica data de criação da tela, data da última modificação, número de objetos contidos na tela e número de variáveis contidas na tela.



## Criação de Objetos Gráficos.

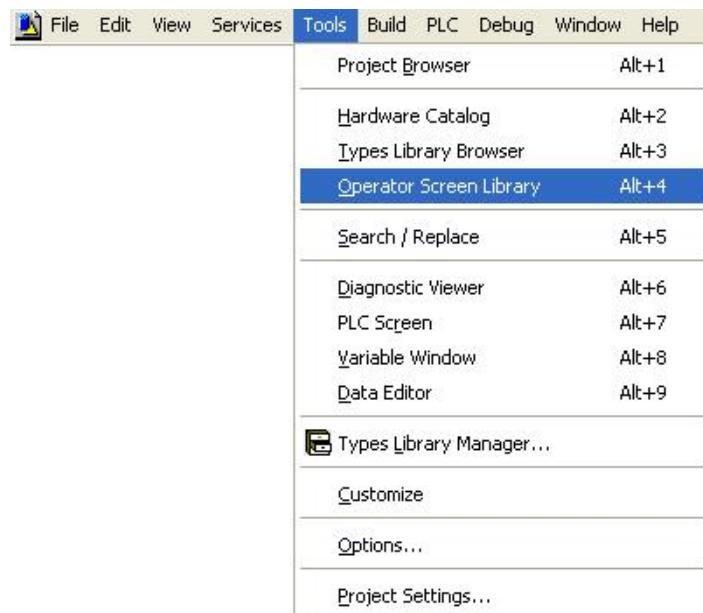
A barra de ferramentas disponibiliza vários objetos da biblioteca para a edição de telas.



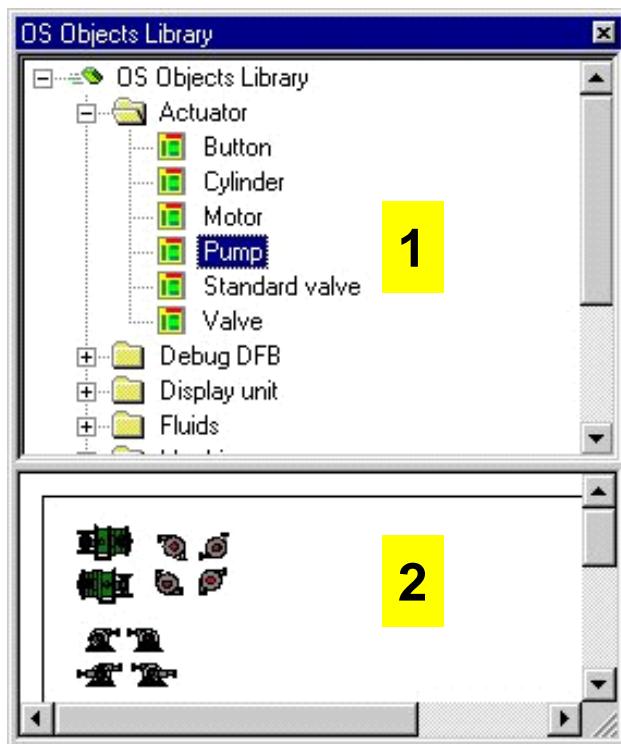
## Inserir Objetos Gráficos Na Tela.

### Inserindo objetos a partir da biblioteca.

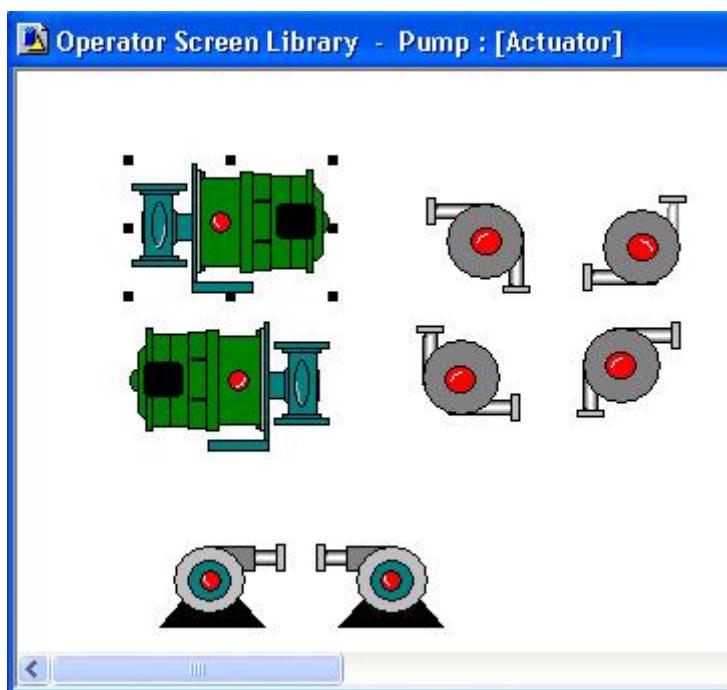
Através da barra de ferramentas, acessar **Tools** e selecionar **Operator screen library**.



Ao selecionar **Operator screen library**, é mostrado a janela com a mesma, onde deve ser selecionada a família do objeto desejado (1) abaixo da família são mostrados os objetos da mesma (2).

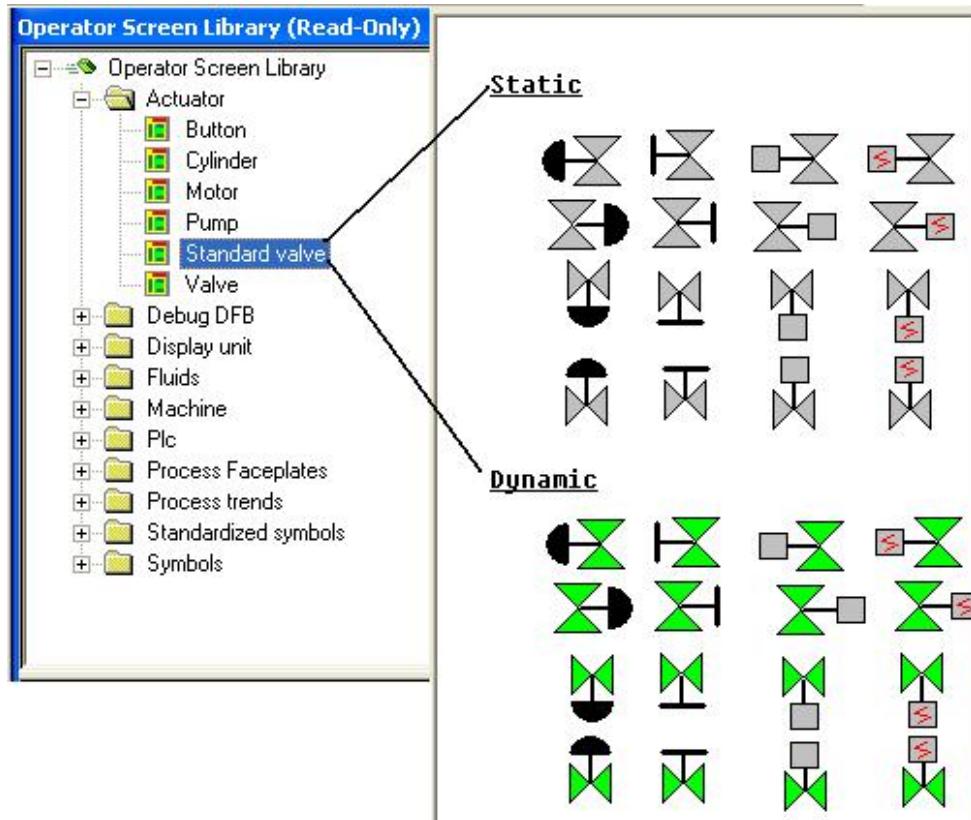


Dar dois cliques sobre a família e selecionar o objeto desejado, copiar e colar o mesmo na tela.



Os objetos da biblioteca podem ser **estáticos**, não possuem variável associada ao mesmo não permitindo animação, ou **dinâmicos** que possuem variáveis associadas aos mesmos permitindo animação,

Por exemplo, a família **Standard Valves** possui os dois tipos.



### Animação dos objetos da biblioteca

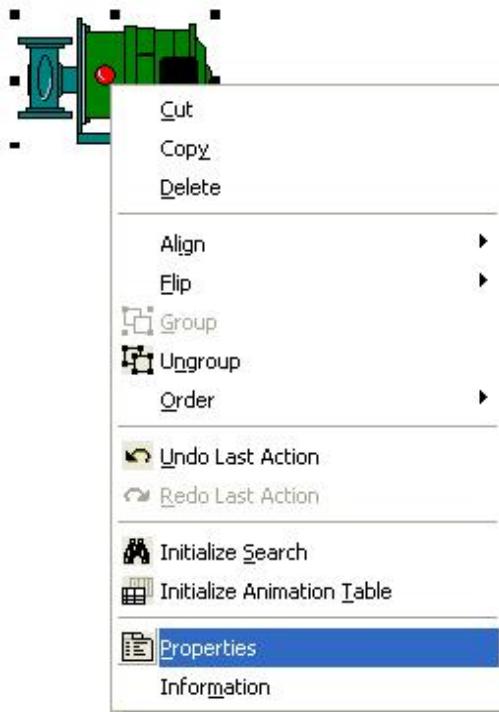
A clicar duas vezes sobre o objeto colocado na tela, será aberta a caixa de diálogo de propriedades do mesmo, com as abas:

**Animation:** onde deve ser especificada a variável que controlará a visualização da mesma (aparecer/desaparecer).



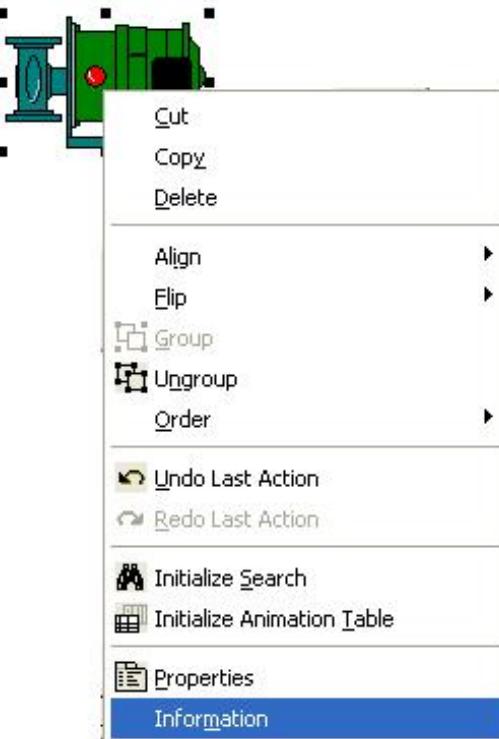
**Animation type:** onde é definido o tipo de animação

A caixa de propriedades pode ser acessada, através de clique com o botão direito do mouse sobre o mesmo, será mostarda a lista com várias funções entre elas **Properties**.

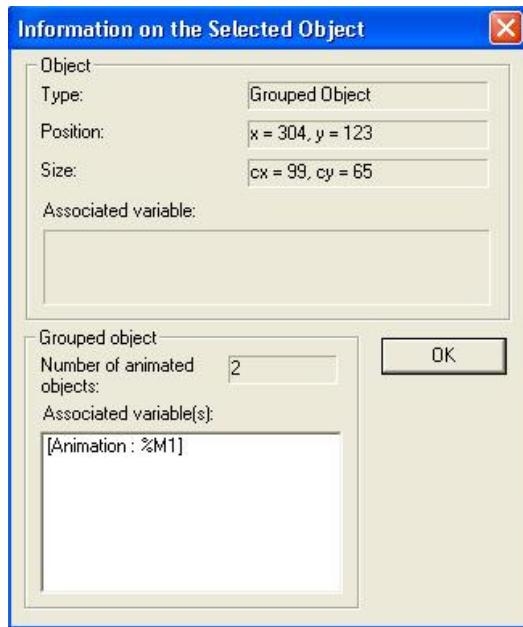


### Informações de um objeto da biblioteca

Clicando com o botão direito do mouse sobre o objeto da tela, será aberta uma lista com várias funções, selecionar **Information**.



Ao ser selecionado **Information**, é aberta a caixa de texto **Information on the Selected Object**, contendo informações gerais do mesmo, assim como número de objetos animados (no caso 2) e variáveis associadas (no caso %M1).

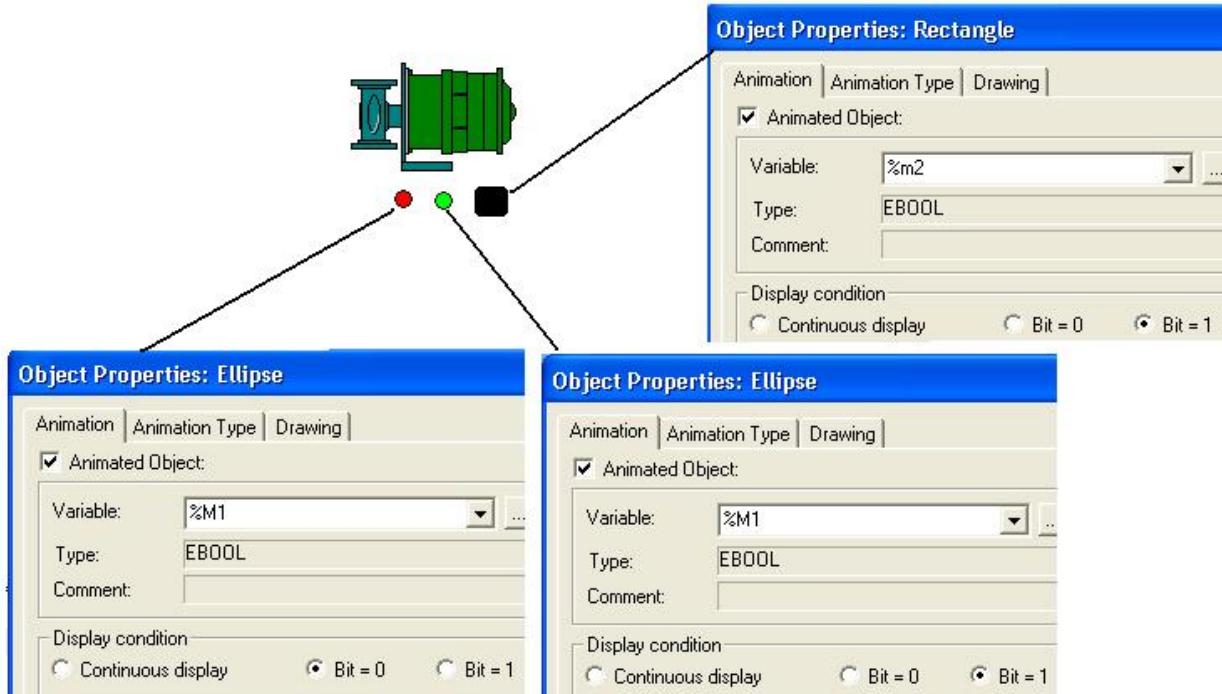


### *Desagrupando objetos da biblioteca.*

Os objetos da biblioteca são objetos compostos por vários elementos agrupados, os quais ao serem desagrupados, tornam possível a animação de cada um destes elementos com as respectivas variáveis, para tornar um objeto único, deve-se refazer o agrupamento após a configuração dos mesmos.

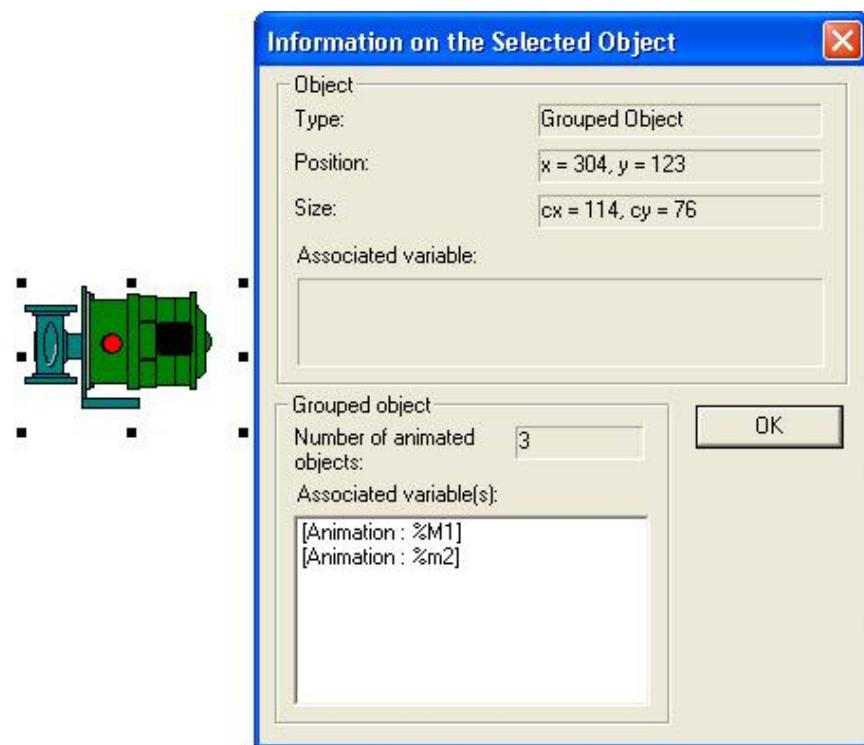
Para desagrupar um objeto, clicar com botão direito sobre o mesmo e selecionar “**Ungroup**”, separar os elementos e fazer a configuração de animação individualmente dos mesmos, após as configurações deve-se re-organizar os elementos e reagrupá-los.

Objeto desagrupado:



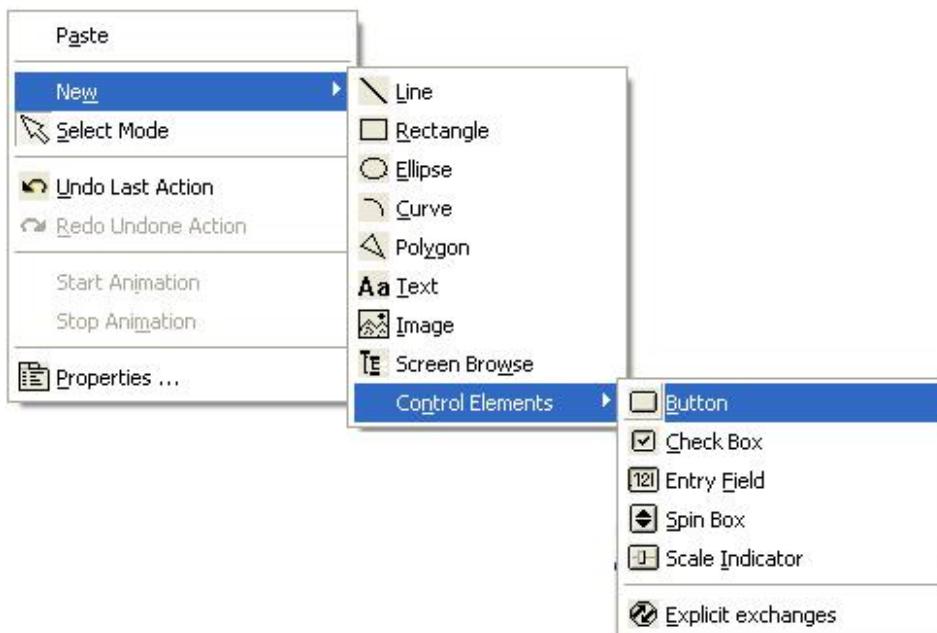
Para reagrupar o objeto deve-se clicar o botão esquerdo do mouse na área de edição e arrastar sobre o objeto, selecionando-os e clicar com o botão direito do mouse selecionar “**Group**”.

Objeto re-agrupado, com as novas informações (variável %m2).



### *Inserindo objetos a partir da área de edição.*

Um objeto pode também ser criado clicando com o botão direito do mouse na área de edição da tela e selecionar New onde são disponibilizados vários objetos para animação, selecionando “Control Elements” são disponibilizados pos objetos de controle, objetos que interagem no CLP.



### *Inserir animações*

#### **Tipos de animação.**

Qualquer objeto gráfico pode ser animado, com os seguintes tipos de animação (para visualização):

- Aparecer/desaparecer (dependendo da condição de visualização)
- Piscar (dependendo da condição de visualização)
- Gráfico de barras (horizontal, vertical, multi-direcional)

- Diagramas de tendências.

### Outros tipos de animações predefinidas (para visualização e ação)

Botão de browser de tela (para mudança de tela)  
 Campo de entrada de dado (para entrar em um registro)  
 Botão (para enviar um bit ou valor de registro)  
 Contador (para setar/resetar um bit ou para incrementar ou decrementar um registro)  
 Cursor (para enviar um valor a um registro)  
 Troca explícita (para enviar uma função como READ\_STS, WRITE\_PARM, ...)

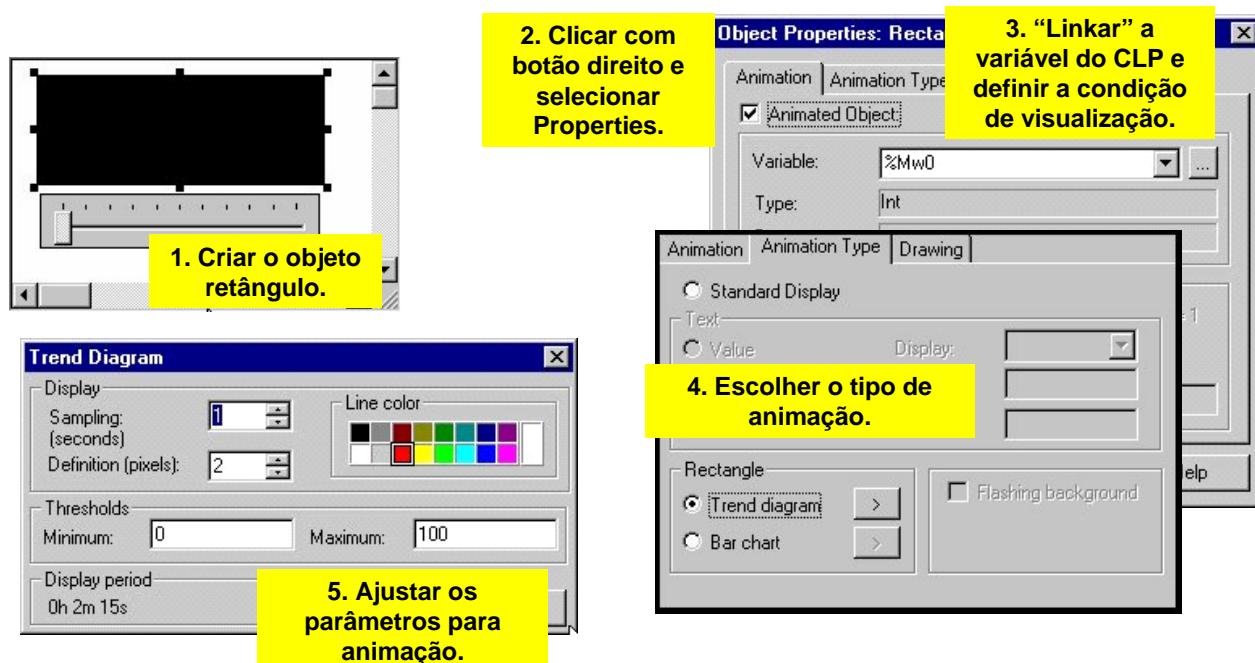
### Diagramas de Tendência.

Diagramas de tendência são usados para representar a evolução de uma variável graficamente.

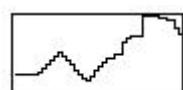
Características dos diagramas de tendência:

- A curva representando a tendência evolui da direita para esquerda.
- Cada diagrama de tendência tem sua própria definição assim como sua própria freqüência de amostragem.
- É possível sobrepor vários diagramas de tendência.
- O número de diagrama de tendência deve ser limitado a ponto de não afetar o desempenho da amostragem.

### Criação do diagrama de tendência.



### Diagrama de Tendência



Em um diagrama de tendência é possível visualizar mais de um gráfico, para isto devem-se criar retângulos na quantidade desejada de gráficos, configurá-los como sem fundo “**Pattern: None**” com suas respectivas variáveis e sobrepor os mesmos.

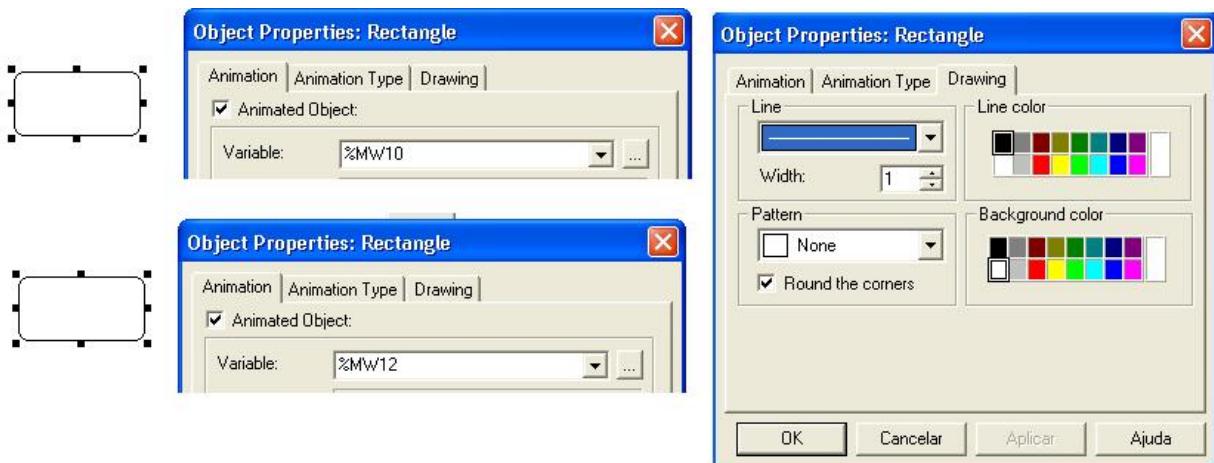
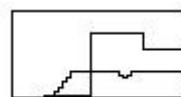
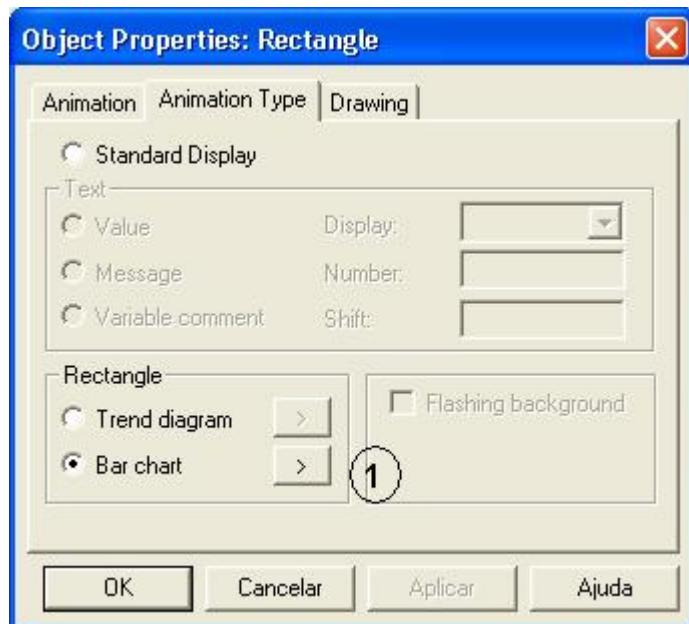


Diagrama de tendência completo.

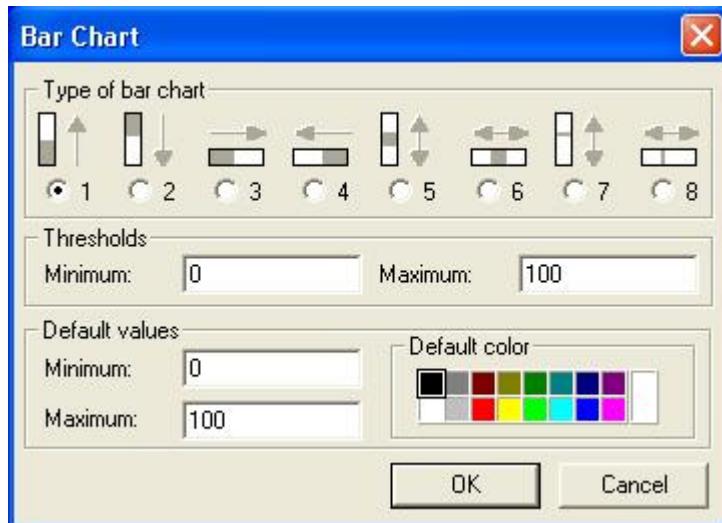


- **Criação de gráfico de barra.**

A criação do gráfico de barra é semelhante a criação diagrama de tendência, para tal deve-se selecionar **Bar chart** na aba **animation type** do retângulo, e para definir os parâmetros do gráfico de barra, acionar o botão [>] (1).



Ao acionar o botão (1), é visualizada a caixa de diálogo “Bar chart” para configuração do mesmo.



Na aba **Animation** do retângulo, é definida a variável que escreverá os valores no gráfico de barra, e na condição de visualização “**Display condition**” são definidos os valores mínimos e máximos permitidos para visualização.

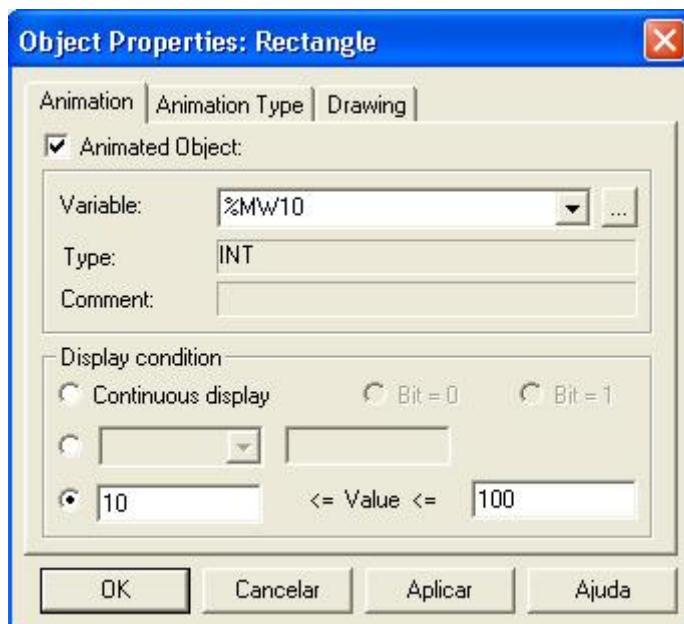


Gráfico de barra criado.

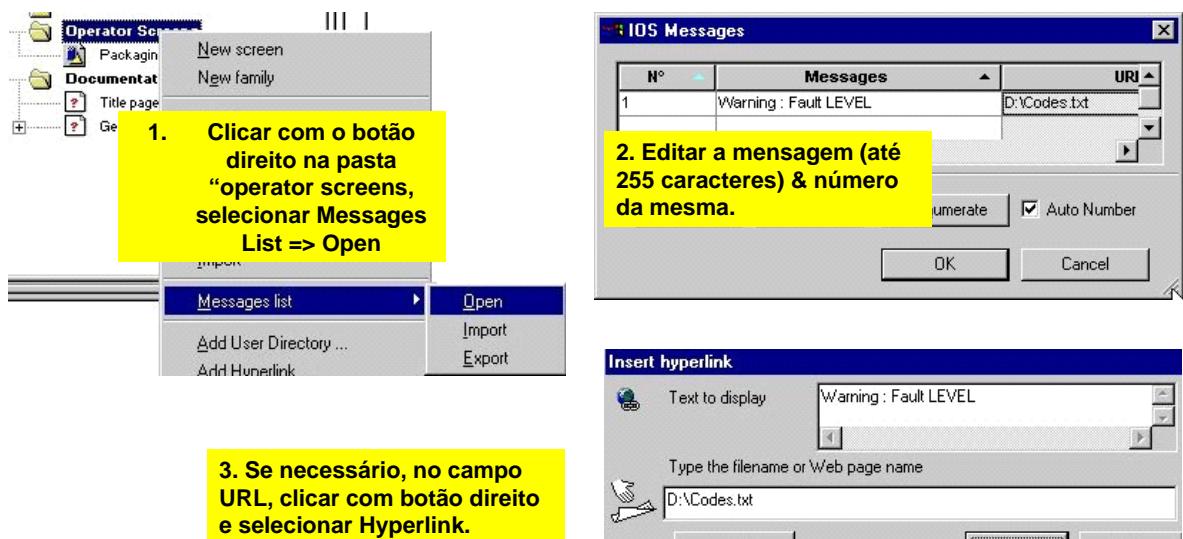


- **Criação de Mensagens.**

Uma mensagem pode ser animada quando está “linkada” com um objeto de texto, ou mostra um rótulo na ativação de um evento.

A criação de mensagem é feita a partir da pasta **Operator Screen** no **project browser**, seguindo as etapas:

## Recursos de Utilidades do Unity



### Chamada de uma mensagem.

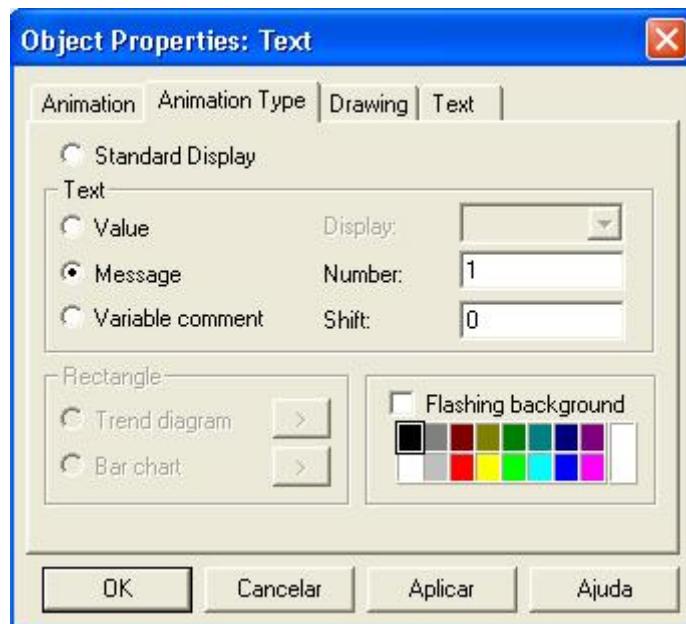
A chamada da mensagem é feita através da ativação de um evento ou anexada a um objeto texto.

Para chamar uma mensagem através de um objeto texto, deve-se:

=>na aba **Animation** definir a variável que controlará a mesma.



=> na aba **Animation Type** marcar **Message** e definir o número da mesma em **Number**.

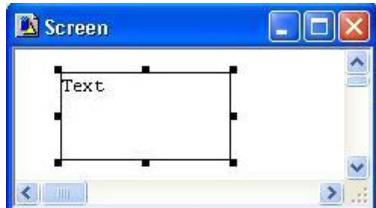


## Funcionamento



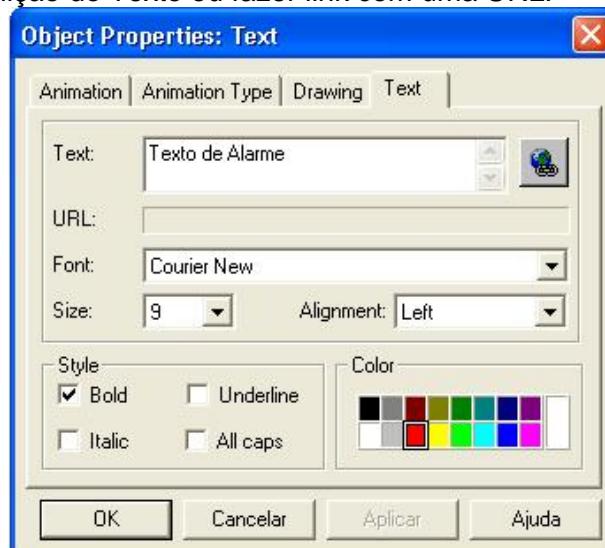
- Inserção de um objeto de texto na tela.**

Clicar com o botão direito do mouse na área de edição, selecionar **New** e então selecionar o objeto **text**, clicar e arrastar o prompt do mouse na área de edição, no tamanho desejado.

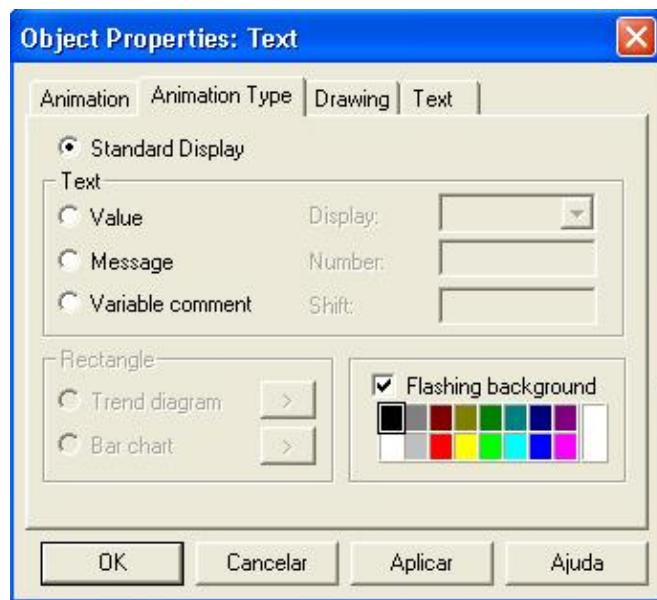


Com o objeto selecionado, com dois cliques sobre o mesmo, será aberta a caixa de diálogo “**Object Properties: Text**”, com as abas:

- Text:** Para definição do Texto ou fazer link com uma URL.



- **Drawing:** Definição do desenho
- **Animation Type:** Definição do tipo de animação, podendo mostrar, o texto editado na aba Text, o valor da variável associada, uma mensagem ou o comentário da variável associada.



- **Animation:** Definição das condições de visualização.

Ao clicar o botão **OK**, o texto é fixado na área de edição da tela:



- **Inserção de uma imagem na tela**

Clicar com o botão direito do mouse na área de edição, selecionar **New** e então selecionar o objeto **Image**, será a caixa de mensagem “**Load na Image**”, onde se deve especificar o tipo e caminho da imagem e açãoar o botão “Abrir”.



Ao acionar o botão Abrir, a imagem é adicionada à área de edição da tela.



- Inserção de um objeto Screen Browser na tela.**

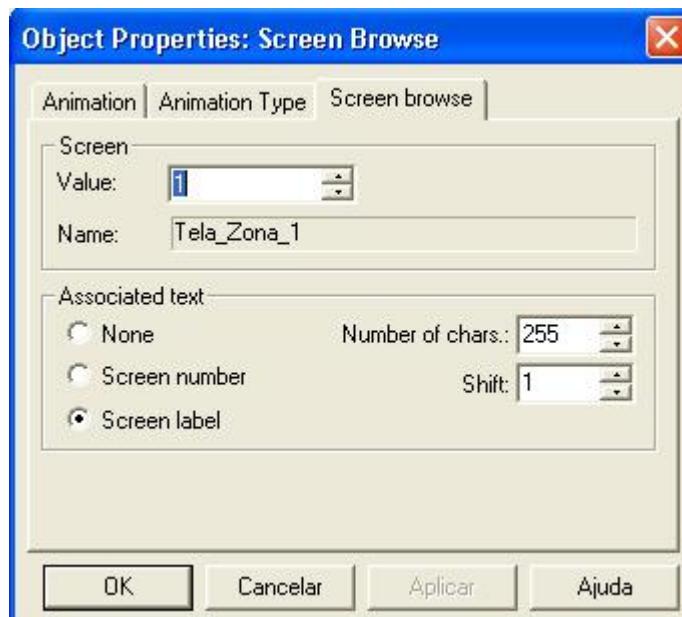
O objeto **screen browser** permite ao ser acionado, a troca de página visualizada.

Clicar com o botão direito do mouse na área de edição, selecionar **New** e então selecionar o objeto **screen browser**, clicar e arrastar o prompt do mouse na área de edição, no tamanho desejado.



O objeto **browser screen** ao ser colocado na tela apresenta por default, o nome da próxima tela que poderá ser visualizada no acionamento do mesmo. Com dois cliques sobre o mesmo, é visualizada a caixa de mensagem “**Object properties: Screen Browser**”, com as seguintes abas:

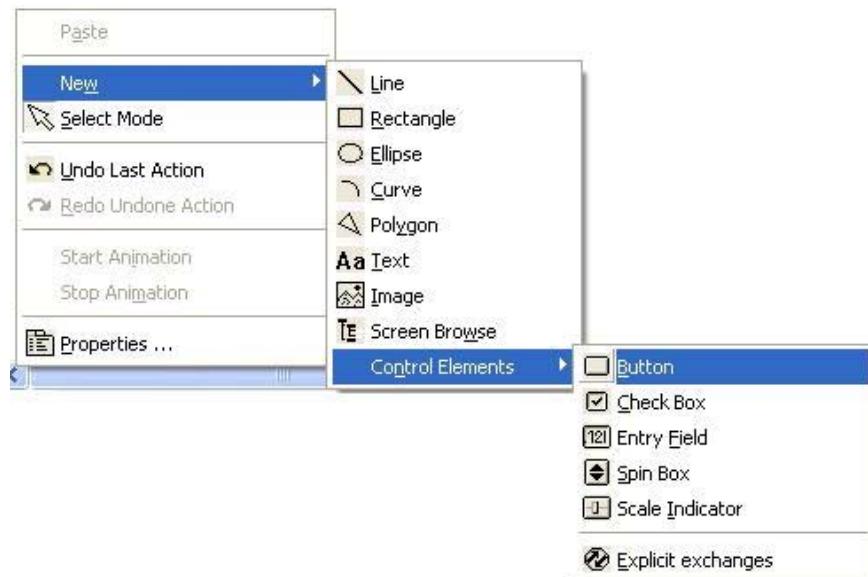
- Screen browser:** onde se define no value, o número da tela a ser visualizada quando acionado, assim como o texto associado.
- Animation type:** define o tipo de animação.
- Animation:** define as condições e variável de controle da visualização.



- **Inserindo Elementos de Controle**

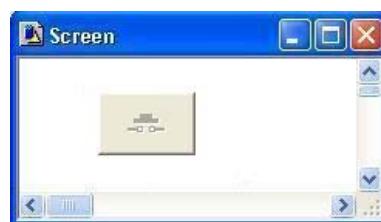
Estes elementos têm a função de interagir com o CLP, seja escrevendo em um registro, “setando” um bit ou gerando comando de troca de mensagens.

Para acessar a estes objetos, clicar com o botão direito do mouse na área de edição, selecionar **New** e então selecionar o a família **Control Elements** para então selecionar o objeto desejado.



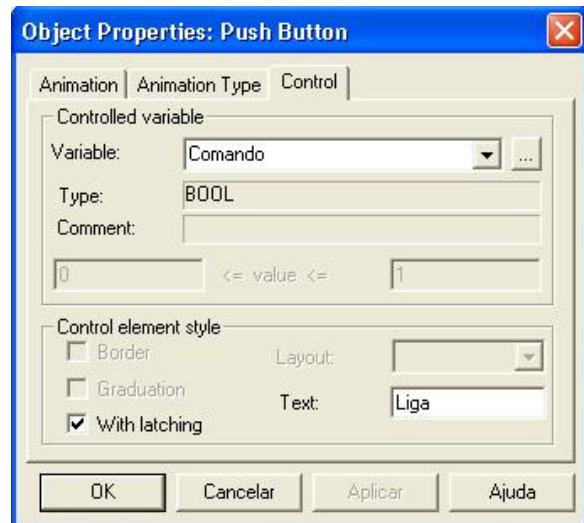
- **Inserindo um Botão.**

Ao selecionar o objeto **Button**, clicar e arrastar o prompt do mouse na área de edição, no tamanho desejado, será visualizado o objeto.



Com dois cliques sobre o mesmo, é visualizada a caixa de mensagem “**Object properties: Push Button**”, com as seguintes abas:

- **Control:** onde se define a variável que será acionada pelo acionamento do botão e o estilo do elemento de controle definindo memorização ou não do acionamento e texto associado ao mesmo.
- **Animation type:** define o tipo de animação.
- **Animation:** define as condições e variável de controle da visualização.



Botão criado.



# CAPÍTULO 15

## Exercícios

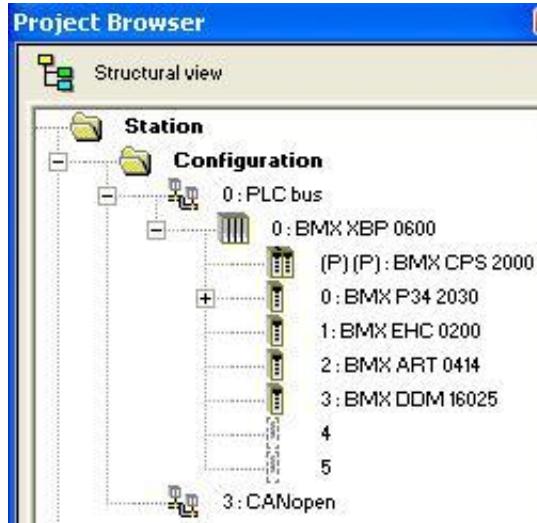
---



## Exercício 1: Criação de projeto padrão

Elaborar um projeto considerando o hardware da maleta didática disponível, fazendo os devidos ajustes do projeto, assim como a declaração das variáveis elementares.

### Hardware da Maleta M340



Código/Modelo	Slot	Descrição de Hardware.
BMX XBP 0600	----	Rack de 6 posições
BMX CPS 2000	----	Fonte de alimentação de Corrente Alternada
BMX P34 2030	0	CPU M340 com porta CANopen e Ethernet TCP/IP integrada
BMX EHC 0200	1	Módulo contador com dois canais
BMX ART 0414	2	Módulo de entrada analógico de temperatura com quatro canais
BMX DDM16025	3	Módulo misto digital de oito entradas e 8 saídas
CANopen	----	Drop de conexão com CANopen da CPU

### Endereçamento de I/Os.

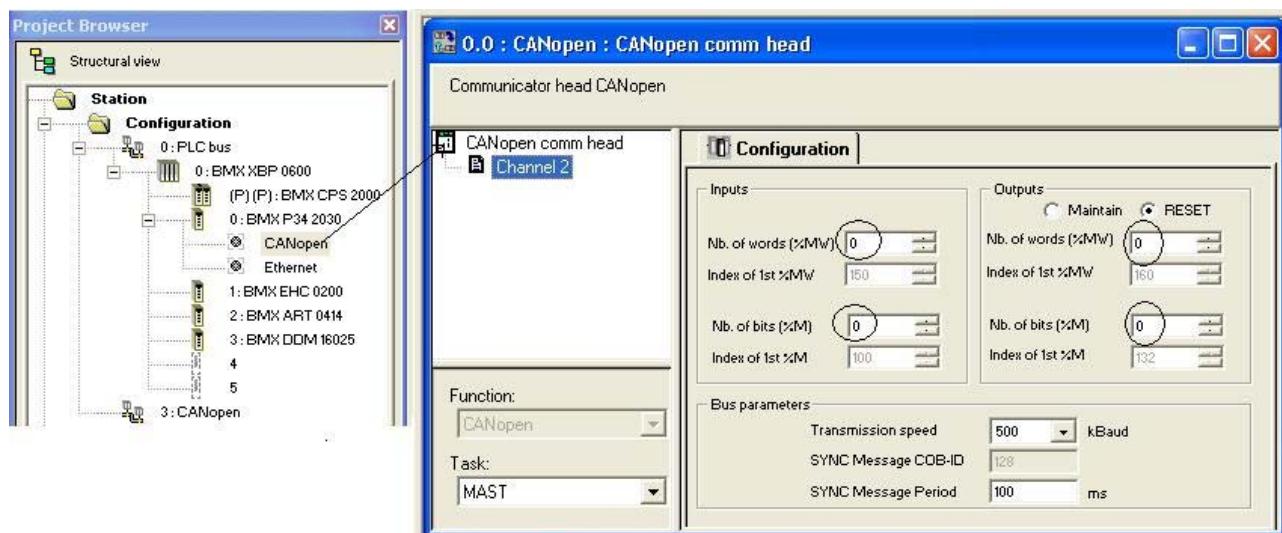
Dispositivo	Tipo de I/O	Endereço	Símbolo
Botão de Pulso Branco	Entrada digital	%I0.3.0	B0
Botão de Pulso Branco	Entrada digital	%I0.3.1	B1
Chave Comutadora	Entrada digital	%I0.3.2	B2
Sinal de Vermelho	Saída digital	%Q0.3.16	L1
Sinal de Amarelo	Saída digital	%Q0.3.17	L2
Sinal de Verde	Saída digital	%Q0.3.18	L3
Potenciômetro	Entrada analógica	%IW0.2.1	E_Ana_1



No módulo contador BMX EHC 0200, é necessário configurar no campo “Function” o canal “Counter 0” em “Free large Counter mode”.



A porta CANopen integrada a CPU reserva por default Bits e Words para seu funcionamento, como a mesma não é utilizada neste curso é necessário colocar “0” Bits e Words reservadas para a mesma, como mostra a figura a seguir:



## Exercício 2: Sistema de Partida Direta de Motor de Indução Trifásico

A partir do hardware e variáveis importadas do exercício1, elaborar um programa em Linguagem Ladder -LD para efetuar a partida direta de motor de Indução trifásico que contenha os seguintes elementos:

- Botão liga
- Botão desliga
- Rele de Sobre carga
- Contator
- Sinalizador de motor ligado
- Sinalizador de ocorrência de sobre carga

## Exercício 3: Sistema de Partida Direta com Reversão de Motor de Indução

Elaborar um programa em Linguagem Ladder -LD para efetuar a partida direta com Reversão de motor de Indução trifásico que contenha os seguintes elementos:

- Botão liga Direita,

Botão Liga – Esquerda,  
Botão desliga,  
Rele de Sobrecarga,  
Contatores,  
Sinalizadores do sentido de giro do motor,  
Sinalizador de ocorrência de sobre carga.

## Exercício 4: Controle do Sistema de Partida Direta com Reversão de Motor de Indução

A aplicação para o sistema de partida utilizado no exercício 3 será utilizado em um misturador de alta viscosidade, devendo possuir agora um sistema que limite o número de partidas em qualquer sentido dentro de um determinado tempo, assim como a indicação de ocorrência da igualdade do número de partidas dos sentidos.

Novas partidas após excedido o número de partida dentro de um determinado tempo, será possível após o acionamento de uma chave “reset” em poder do supervisor do setor.

## Exercício 5: Controle Básico de Entrada Analógica

Desenvolver uma aplicação em linguagem Ladder para um forno com zona de aquecimento simples que possui uma IHM a qual mostra a diferença entre o valor de Setpoint fornecida pela mesma e o valor de feedback fornecido pelo sensor de temperatura conectado a entrada analógica do CLP. Uma saída de alarme deve ser acionada sempre que o valor de feedback for maior que o valor de setpoint.



O Valor de Setpoint e da diferença entre o mesmo e o valor de Feedback, devem ser do tipo REAL.

## Exercício 6: Controle Básico de Entrada e Saída Analógicas

Desenvolver uma aplicação em linguagem Ladder para uma incubadora, que utiliza um sistema de controle de aquecimento de precisão, o qual fornece o valor de temperatura com décimos e centésimos. O valor de erro entre o valor de setpoint fornecido por uma entrada analógica e o valor de feedback fornecido pelo sensor deve ser enviado a uma saída analógica que controla o driver de aquecimento e a uma IHM.

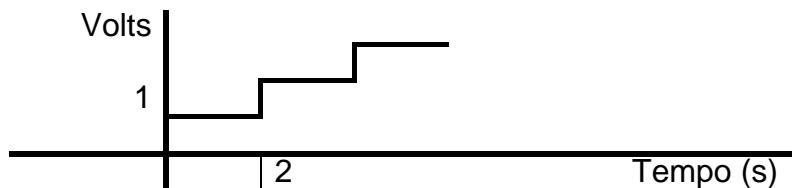
Duas saídas digitais devem sinalizar o status do erro positivo/negativo.



Acrescentar ao hardware até aqui utilizado, o módulo BMX AMO 0210.

## Exercício 7: Rampa de aceleração

Construir uma rampa de aceleração, para acionamento de um motor, mostrada no gráfico abaixo:



A cada 2 s o CLP deve adicionar à sua saída analógica 1 volt, até alcançar 10 volts. Existe um inversor de freqüência utilizando a saída analógica de 0 a 10 V do CP para o acionamento do motor.

Nova partida será possível após o acionamento de uma chave “Re-iniciar” a qual zera a saída analógica.

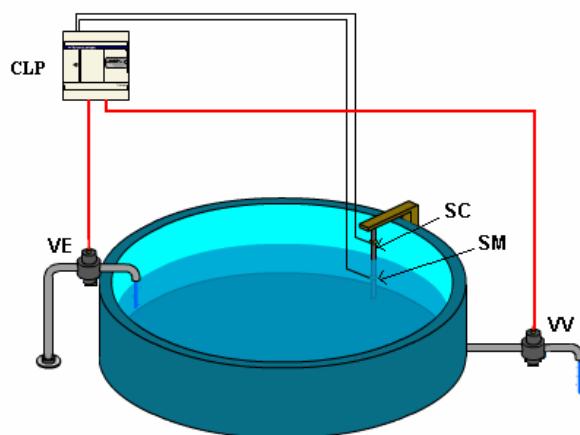
## Exercício 8: Utilização de Bits e Words de sistema

Utilizar um bit de sistema para gerar pulsos com período de 100 ms os quais serão fornecidos a um contador que será zerado no primeiro ciclo após a colocação do CLP em RUN.

## Exercício 9: Controle de Nível em Diagrama de Blocos Funcionais – FBD

Elaborar um programa em Linguagem de Blocos Funcionais – FBD para executar um processo de controle de nível que funcione da seguinte maneira:

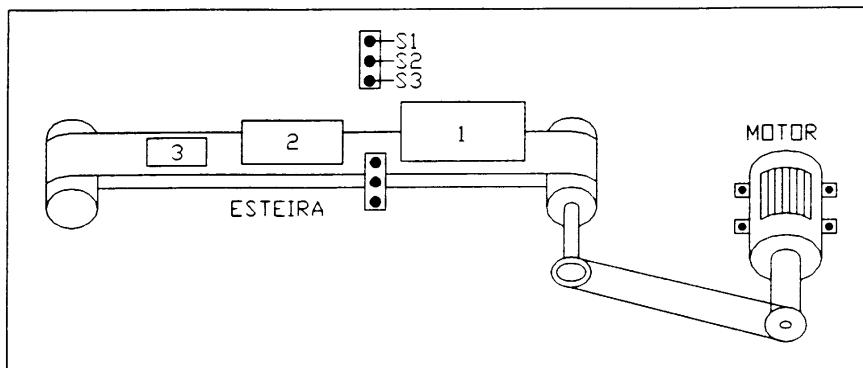
- Este processo deve ser contínuo, bastando pulsar um botão de comando LIGA (B1) o desligamento ocorre ao pulsar o botão de comando DESLIGA (B0);
- No tanque que terá o nível controlado possui dois sensores, SENSOR CHEIO (SC) e SENSOR MÉDIO (SM) uma VÁLVULAS DE ENCHIMENTO (VE) e outra de VAZÃO (VV);
- Quando o nível do tanque estiver abaixo do sensor SM, deve-se acionar a válvula VE impedindo o acionamento da válvula VV;
- Quando o nível do tanque for detectado pelo sensor SC, deve-se acionar a válvula VV impedindo o acionamento da válvula VE;
- Estando o nível detectado por SM e não detectado por SC as válvulas VV e VE devem ser mantidas acionadas a fim de manter o nível dentro dos limites.



## Exercício 10: Contagem dos produtos de uma linha de produção

Elaborar um programa em Linguagem de Blocos Funcionais – FBD para uma linha de produção de 3 produtos. O primeiro produto foi embalado em uma caixa de 10 cm de altura, o segundo com 8 cm de altura e o terceiro com 5 cm de altura. Deve-se contar a quantidade de cada produto que passa na esteira pelos sensores infravermelhos. Cada sensor possui a mesma altura dos produtos, 1, 2 e 3 respectivamente SI-1, SI-2 e SI-3. Programe três contadores utilizando uma lógica com os sensores, que contem os produtos.

A cada inicio de produção os contadores devem ser zerados através do acionamento de um botão “reset”



## Exercício 11: Utilização de Variáveis DDT em Controle de Turno

Criar uma instância denominada “Controle\_de\_Turno” (Tipo Struct) com as seguintes variáveis derivadas Nome\_do\_responsavel (tipo de dado String), Hora\_de\_Inicio (Tipo de dados INT), Hora\_de\_Encerramento(Tipo de dados INT), Processo\_Ativo (Tipo de dados BOOL) e Quantidade\_produzida (Tipo de dados INT).

## Exercício 12: Utilização de I/ODDT na supervisão de Entrada e de Saída Digital

O sistema de segurança de delimitação do campo de atuação de um robô é composto por uma entrada digital, e uma saída digital para indicação de invasão da referida área. Por ser um sistema crítico, desenvolver uma aplicação que utilizando I/ODT detecte a ocorrência de falha na entrada ou na saída digital, tal falha deve acionar uma saída digital com freqüência de 1HZ.

## Exercício 13: Aplicação de Bloco de Função Derivado – DFB no Controle de Sobrecarga

Substituir a lógica do exercício 2 por um Bloco de Função Derivado – DFB, que além dos comandos citados no exercício 2 possua um sinalizador amarelo de ocorrência de sobrecarga que pisque a uma freqüência de 1HZ. E que após um determinado número de acionamentos do sinaleiro amarelo, seja acionado de maneira permanente um sinaleiro vermelho, o sinaleiro amarelo, permanecendo assim até o re-arme do rele de sobrecarga.



A entrada do número de acionamentos para acionamento do sinaleiro vermelho deve ser externa ao bloco.

## Exercício 14: Utilização de Telas de Operação para Controle de Nível

Desenvolver uma Tela de Operador para a aplicação desenvolvida no exercício 9 “Controle de Nível em Diagrama de Blocos Funcionais – FBD, com um botão desliga no terminal e textos indicadores de estado de sensores, válvulas e processo, criar também uma família de objetos “Sensores de Nível” com os objetos sensores de nível SC e SM.

## Exercício 15: Sistema de Análise de pH

Programe em linguagem Ladder um sistema de análise do pH da água de uma Estação de Tratamento de Água - ETA, para o controle da dosagem do alcalinizante. Considere um sinal de uma entrada analógica, proveniente de um medidor de PH que tem sua faixa de 0 a 20mA para variação do PH de 0,00 a 14,00. Programe também uma tela de operação para somente visualização da variação do pH na escala de 0,00 a 14,00 e na escala de 0 a 20mA com as seguintes mensagens para o operador:

**Verificar Alcalinizante - pH MUITO BAIXO** => valores de pH 0.0 a 2.0.

**pH Baixo** => valores de pH 2.1 a 4.0

**pH Normal** => valores de pH 4.1 a 7.9

**pH Alto**, => valores de pH 8.0 a 11.9

**Dosagem parada – pH MUITO ALTO**, => valores de pH 12.0 a 14.0.

## Exercício 16: Aplicação das funções de Exportação/Importação

O sistema de controle de nível desenvolvido no exercício 14 será acrescido de uma bomba de recalque com sistema de partida direta e bloco DFB utilizados no exercício 13, utilizar os recursos de Exportação/Importação das sessões e partes dos exercícios citados a fim de desenvolver a aplicação.

## Exercício 17: Utilização de Recursos de Utilidades do Unity

Adicionar à aplicação do exercício 14, diretório de usuário com hyperlinks contendo informações técnicas dos sensores e das válvulas. Deverá ser gerado ainda o arquivo de impressão do mesmo.

## **CAPÍTULO 16**

### **Glossário**

---



# Glossário

1. IEC - International Electrotechnical Committee.
2. ST – Structured Text - Texto Estruturado: Linguagem Textual regulamentada pela IEC
3. IL – Instruction List - Lista de Instruções: Linguagem Textual regulamentada pela IEC
4. LD - Diagrama Ladder: Linguagem Gráfica regulamentada pela IEC
5. FBD - Function Block Diagram Diagrama Blocos Funcionais: Linguagem Gráfica regulamentada pela IEC
6. SFC - Sequential Function Chart - Seqüenciamento Gráfico de Funções: Linguagem Gráfica regulamentada pela IEC
7. Tasks - Tarefas: Uma Task é um mecanismo de escalonamento muito útil para sistemas de tempo real, que executa Programas ou Blocos funcionais periodicamente ou em resposta a um evento (mudança de estado de alguma variável booleana), permitindo a execução de programas em diferentes taxas.
8. Data Types - Tipos de dados: Como os CLP's atuais são utilizados em diversas aplicações, a norma define a capacidade de se utilizarem diversos tipos de dados. É suportado o uso de literais para todos os tipos de dados, os quais devem iniciar com uma identificação do mesmo.
9. ANY - Tipos de dados genéricos: Os tipos de dados genéricos utilizam o prefixo ANY, podendo ser utilizados em funções ou blocos funcionais.
10. Variável Global: São acessadas por todos os elementos contidos no software que as declara, incluindo os elementos aninhados. A norma exige a declaração de variáveis dentro de diferentes elementos de software, tais como Programas e Blocos Funcionais, podendo utilizar nomes com significado abrangente (simbólicos) e serem de diferentes tipos de dados, podendo ainda ser de alocação dinâmica e associadas a posições de memória (representação direta).
11. Variável Local: O escopo das variáveis é local ao elemento de software que as declara, permitindo acesso dentro do próprio elemento que pode ser uma Configuração, Recurso, Programa, Bloco Funcional ou Função.
12. Project Browser: permite visualizar todo o conteúdo de um projeto do Unity Pro separado por pastas. Visualizadas de duas diferentes maneiras; Vista Estruturada ou Vista Funcional.
13. Fallback: valor assumido por um canal na volta de uma falha.
14. State RAM: Estado interno (sem endereçamento) da memória RAM.
15. RIO BUS- Remote Input/Output BUS: Rack remoto composto por Módulos de I/O e módulo de comunicação sem Processador.
16. Local Bus: Rack responsável pelo processamento da aplicação, contendo Processador, módulos de I/O e de comunicação.
17. I/ODDT: É a abreviação de Input/Output Derived Data Type. O termo I/ODDT designa uma estrutura de tipo de dado representando um canal do CLP, cada módulo específico processa seus próprios I/ODDT
18. BOOL/EBOOL: Variáveis Booleanas “BOOL” precisam ser FALSA (0) ou VERDADEIRA (1). As variáveis EBOOL permitem a função “force” e a detecção de borda.
19. WORD: Representa uma string de 16 bits, significa que o comprimento do dado é de 16 bits.
20. INT: Representa um valor inteiro. A faixa de valores são -32768 até 32767
21. UINT: Representa um valor inteiro não sinalizado (+ ou -). A faixa de

	valores são 0 até 65535.
22.	REAL: Representa um valor com ponto flutuante. A faixa de valores são -3.40e+38 até 3.40e+38
23.	Variables &FB instances: Variáveis e Instâncias de Blocos de Função.
24.	EDT - Elementary Data Types: Tipo de dados elementares
25.	DDT - Derived Data Types: Tipo de dados Derivados
26.	MAST Task: Tarefa principal Composta por Sub-rotinas “SR” e Sessões programadas em LD, FBD, IL, ST ou SFC, podendo ter execução cíclica ou periódica (0..255 ms, 0ms = operação cíclica, sendo controlada por watch dog e bits/words de sistema)
27.	FAST Task: Tarefa rápida Composta por Sub-rotinas “SR” e sessões programadas em LD, FBD, IL, ST com execução periódica (1..255 ms) sendo controlada por watch dog e bits/words de sistema.
28.	AUX Task: Tarefas Auxiliares: Este tipo de tarefa é disponível nas CPUs Premium TSX P575** e Quantum 140 CPU 6*****, podendo ter até 4 tarefas (AUX0 a AUX3). São compostas por sessões e sub-rotinas “SR” programadas em LD, FBD, IL e ST com execução periódica de 10ms a 2.55s.
29.	Events - EVT e TIMER: Tarefas de Eventos ou Temporizada: São tarefas que são executadas quando ocorre um determinado evento.
30.	Multitarefa: Quando a aplicação possui algumas tarefas.
31.	Seções: É uma parte da lógica da aplicação. As sessões podem ser elaboradas em qualquer uma das linguagens referenciadas na IEC 6 1131-3 que são LD, IL, ST, FBD ou SFC, e cada tarefa pode ter um número ilimitado de sessões.
32.	Sub-Rotinas: Uma parte lógica da aplicação e pode ser elaborada nas linguagens LD, IL, ST e FBD, e são chamadas a partir de uma sessão ou sub-rotina.
33.	Build Project: => Cria um arquivo que pode ser baixado para o CLP ou para o simulador do CLP
34.	EFs - Elementary Functions: Funções elementares
35.	EFBs -Elementary Functions Blocks: Blocos de Função Elementares
36.	FFBs - Coletivo do termo EF (elementary function), EFB (elementary function block) and DFB (derived function block).
37.	EBOOL: Tipo de dado “booleano” que permite a detecção de borda “Edge”
38.	DFBs - Derived Function Blocks: Blocos de Função derivado utilizam uma lógica encapsulada para atender uma necessidade específica da aplicação para os casos onde a mesma é utilizada várias vezes na aplicação alterando apenas as variáveis em questão
39.	DDTs - Derived Data types: Tipo de dados Derivados
40.	Libset: Biblioteca global: Biblioteca disponível a todos os projetos criados com a ferramenta;
41.	Custom Lib: Biblioteca do usuário possui objetos selecionados pelo usuário.
42.	Application Library Biblioteca Local: Biblioteca disponível apenas ao projeto em questão.
43.	Rung: Um grupo de objetos os quais são conectados entre si e não

	possui conexão com outros elementos, denominação utilizada principalmente em Diagrama Ladder.
44.	Networks: Um grupo de objetos os quais são conectados entre si e não possuem conexão com outros elementos. Denominação utilizada principalmente em Diagrama de Blocos.
45.	Scan: Processo compreendido entre a leitura das entradas, processamento da aplicação e atualização das saídas.
46.	Standard Mode: Modo de conexão com o Hardware do CLP
47.	Bits e Words do Sistema: Bits e Words utilizadas pelo sistema que dependendo do tipo pode ser lida e/ou escrita pelo usuário.
48.	Array: Forma de organização de dados de mesmo tipo, também conhecido como "Arranjo"
49.	Struct: Forma de organização de dados de diferentes tipos, também conhecido como "Estrutura"
50.	Operator Screens: Ferramenta de operação customizada atendendo ao primeiro e segundo nível de funções de diagnóstico sendo para controle (mostrando em tempo real o estado de uma máquina/processo) e monitoração (Telas de Run-time customizadas para atender a necessidade do operador).
51.	Hyperlink :O recurso de hyperlink permite “linkar” o projeto com documentações externas tais como folhas de informações em todos os diretórios e sub-diretórios,
52.	User Directory :Diretório de usuário utilizado para agrupar os diversos links do projeto.