Manual de Programação Ladder MasterTool Extended Edition MT8000

Rev. G 08/2010
Cód. Doc.: MP399102

altus

Conforme o Código de Defesa do Consumidor vigente no Brasil, informamos, a seguir, aos clientes que utilizam nossos produtos aspectos relacionados com a segurança de pessoas e instalações.

Os equipamentos de automação industrial fabricados pela Altus são robustos e confiáveis devido ao rígido controle de qualidade a que são submetidos. No entanto, equipamentos eletrônicos de controle industrial (controladores programáveis, comandos numéricos, etc.) podem causar danos às máquinas ou processos por eles controlados em caso de defeito em suas partes e peças ou de erros de programação ou instalação, podendo inclusive colocar em risco vidas humanas.

O usuário deve analisar as possíveis consequências destes defeitos e providenciar instalações adicionais externas de segurança que, em caso de necessidade, sirvam para preservar a segurança do sistema, principalmente nos casos da instalação inicial e de testes.

Os equipamentos fabricados pela Altus não trazem riscos ambientais diretos, não emitindo nenhum tipo de poluente durante sua utilização. No entanto, no que se refere ao descarte dos equipamentos, é importante salientar que quaisquer componentes eletrônicos incorporados em produtos contêm materiais nocivos à natureza quando descartados de forma inadequada. Recomenda-se, portanto, que quando da inutilização deste tipo de produto, o mesmo seja encaminhado para usinas de reciclagem que deem o devido tratamento para os resíduos.

É imprescindível a leitura completa dos manuais e/ou características técnicas do produto antes da instalação ou utilização do mesmo.

Os exemplos e figuras deste documento são apresentados apenas para fins ilustrativos. Devido às possíveis atualizações e melhorias que os produtos possam incorrer, a Altus não assume a responsabilidade pelo uso destes exemplos e figuras em aplicações reais. Os mesmos devem ser utilizados apenas para auxiliar na familiarização e treinamento do usuário com os produtos e suas características.

A Altus garante os seus equipamentos conforme descrito nas Condições Gerais de Fornecimento, anexada às propostas comerciais.

A Altus garante que seus equipamentos funcionam de acordo com as descrições contidas explicitamente em seus manuais e/ou características técnicas, não garantindo a satisfação de algum tipo particular de aplicação dos equipamentos.

A Altus desconsiderará qualquer outra garantia, direta ou implícita, principalmente quando se tratar de fornecimento de terceiros.

Os pedidos de informações adicionais sobre o fornecimento e/ou características dos equipamentos e serviços Altus devem ser feitos por escrito. A Altus não se responsabiliza por informações fornecidas sobre seus equipamentos sem registro formal.

DIREITOS AUTORAIS

Sumário

1. Introdução

O Software MasterTool Extended Edition

O software MasterTool Extended Edition MT8000, ou simplesmente MToolXE, é a ferramenta de configuração e programação de equipamentos Altus (Séries Grano, Série PX, Ponto e AL-2004), incluindo CPs e remotas. Esta ferramenta permite também o monitoramento de processos, configuração de módulos e geração de relatórios. É executável nos sistemas operacionais Windows® 2000, Windows® XP e Windows® 7, (todos 32bits)., tendo disponíveis versões em inglês e português do software e manuais. Este software é uma versão aprimorada do MasterTool Programming, contendo inúmeros recursos melhorados e extras, tornando esta ferramenta de programação muito versátil.

O MasterTool Extended Edition permite o desenvolvimento de aplicações. A edição do programa aplicativo utiliza o conceito de programação simbólica (tags ou nicknames), possibilitando a documentação do projeto durante a edição dos módulos. O conceito de projeto, que estabelece uma relação entre vários arquivos formando um ambiente de trabalho, facilita o trabalho, reduzindo de forma significativa o tempo de desenvolvimento, além de impedir o usuário de cometer erros de configurações mais comuns, através da opção de verificação de projeto.

MasterTool ProPonto MT6000

Os CPs da Série Ponto foram incluídos na versão 3.00 do MasterTool e o software MasterTool ProPonto MT6000 é necessário para sua programação. Este software, referenciado neste documento como ProPonto, encontra-se no mesmo CD-ROM do MasterTool XE, em um subdiretório com o mesmo nome, e deve ser instalado a partir de lá.

ATENÇÃO: O ProPonto é necessário apenas para a configuração de barramentos de CPs da Série Ponto. Maiores informações sobre o ProPonto podem ser obtidas no Manual de Utilização do ProPonto, que pode ser encontrado no formato PDF no diretório ProPonto\Manual do CD-ROM.

Lite, Professional e Advanced

O software MasterTool Extended Edition MT8000 possui três versões de distribuição, cada uma com um perfil otimizado de acordo com a necessidade. São elas:

- **Lite**: software programador específico para pequenas aplicações
- **Professional**: software programador contendo as ferramentas necessárias para todas as linhas de CPs da Altus
- **Advanced**: software programador com ferramentas para aplicações de grande porte

Cada uma destas versões possui características, finalidades e funcionalidades específicas para cada propósito. O detalhamento sobre as diferenças entre as versões pode ser visto no Manual de Utilização do MasterTool Extended Edition.

Documentos Relacionados a este Manual

Para obter informações adicionais sobre o MasterTool Extended Edition podem ser consultados outros documentos (manuais e características técnicas) além deste. Estes documentos encontram-se disponíveis em sua última revisão em www.altus.com.br.

Aconselha-se os seguintes documentos como fonte de informação adicional:

- Características Técnicas MT8000
- Manual de Utilização do MasterTool Extended Edition
- Manual de Programação ST do MasterTool Extended Edition
- Manual de Utilização do MasterTool ProPonto - MT6000

Inspeção Visual

Antes de proceder à instalação, é recomendável fazer uma inspeção visual cuidadosa do material, verificando se não há danos causados pelo transporte. Verifique se todos os CD-ROMs estão em perfeito estado. Em caso de defeitos, informe a companhia transportadora e o representante ou distribuidor Altus mais próximo.

É importante registrar o número de série de cada equipamento recebido bem como as revisões de software, caso existentes. Essas informações serão necessárias caso se necessite contatar o Suporte Técnico da Altus.

Suporte Técnico

Para entrar em contato com o Suporte Técnico da Altus em São Leopoldo, RS, ligue para +55-51-3589-9500. Para conhecer os centros de Suporte Técnico da Altus existentes em outras localidades, consulte nosso site (www.altus.com.br) ou envie um e-mail para altus@altus.com.br.

Se o software já estiver instalado, tenha em mãos as seguintes informações ao solicitar assistência:

- A versão do software MasterTool Extended Edition
- A versão da chave de software utilizada no MasterTool Extended Edition
- A revisão do equipamento e a versão do software executivo, constantes na etiqueta afixada na lateral do produto, quando a questão refere-se com comunicação com dispositivos
- O conteúdo do programa aplicativo (módulos)
- A versão do sistema operacional Windows (juntamente com seu Service Pack) do computador em que o software está sendo executado

Mensagens de Advertência Utilizadas neste Manual

Neste manual, as mensagens de advertência apresentarão os seguintes formatos e significados:

PERIGO: Indica situações potenciais que, se não observadas, causam danos à integridade física e saúde, patrimônio, meio ambiente e perda da produção.

CUIDADO:
Indica detalhes de configuração, aplicação e instalação que *devem* ser seguidos para evitar condições que possam levar a falha do sistema e suas conseqüências relacionadas.

ATENÇÃO:
Indica detalhes importantes de configuração, aplicação ou instalação para obtenção da máxima performance operacional do sistema.

2. A Linguagem de Programação

Os controladores programáveis surgiram para substituir painéis de controle a relés. Neste contexto, uma linguagem de programação que mais se aproximasse da experiência de técnicos e engenheiros seria a solução mais adequada para desenvolvimento de programas aplicativos de CPs.

Em vista disso, as instruções disponíveis para a construção do programa aplicativo no MasterTool XE são programadas em linguagem de relés e blocos, muito semelhante à linguagem de contatos elétricos e bobinas, utilizadas na descrição dos painéis de controle a relé.

A principal vantagem da utilização deste tipo de linguagem é seu rápido aprendizado, pois se assemelha muito com os esquemas elétricos convencionais.

O acompanhamento e verificação de funcionamento de um programa aplicativo é similar ao de um esquema elétrico, com a vantagem de visualizar o estado dos contatos e bobinas na janela do MasterTool XE.

Este capítulo descreve a linguagem de Relés e Blocos ALTUS detalhando os elementos da linguagem, a estruturação modular de um programa aplicativo e a função de cada módulo.

Ao final da leitura deste capítulo será possível estruturar um programa aplicativo bem como realizar a configuração de CPs.

Elementos de Programação

Um programa aplicativo é composto por quatro elementos básicos:

- Módulos
- Lógicas
- Instruções
- Operandos

Um programa aplicativo é composto por diversos módulos, permitindo uma melhor estruturação das rotinas de acordo com as suas funções. Os módulos são programados em linguagem de relés, seguindo a tendência mundial de normatização nesta área.

Um módulo de programa aplicativo é dividido em lógicas de programação. O formato de uma lógica de programa aplicativo utilizado nos CPs das séries AL-2000, PONTO, PX e GRANO permite até oito elementos em série e até quatro caminhos em paralelo.

As instruções são utilizadas para executar determinadas tarefas por meio de leituras e/ou alterações do valor dos operandos.

Os operandos identificam diversos tipos de variáveis e constantes utilizadas na elaboração de um programa aplicativo, podendo ter seu valor modificado de acordo com a programação realizada. Como exemplo de variáveis pode-se citar pontos de E/S e memórias contadoras.

Cada elemento componente do programa aplicativo é explicado em detalhes nas seções seguintes.

Lógicas

Chama-se lógica a matriz de programação formada por 32 células (elementos da matriz) dispostas em quatro linhas (0 a 3) e 8 colunas (0 a 7). Em cada uma das células podem ser colocadas instruções, podendo-se programar até 32 instruções em uma mesma lógica.

Cada lógica presente no programa simula um pequeno trecho de um diagrama de relés real. A figura a seguir mostra o formato de uma lógica do programa aplicativo.

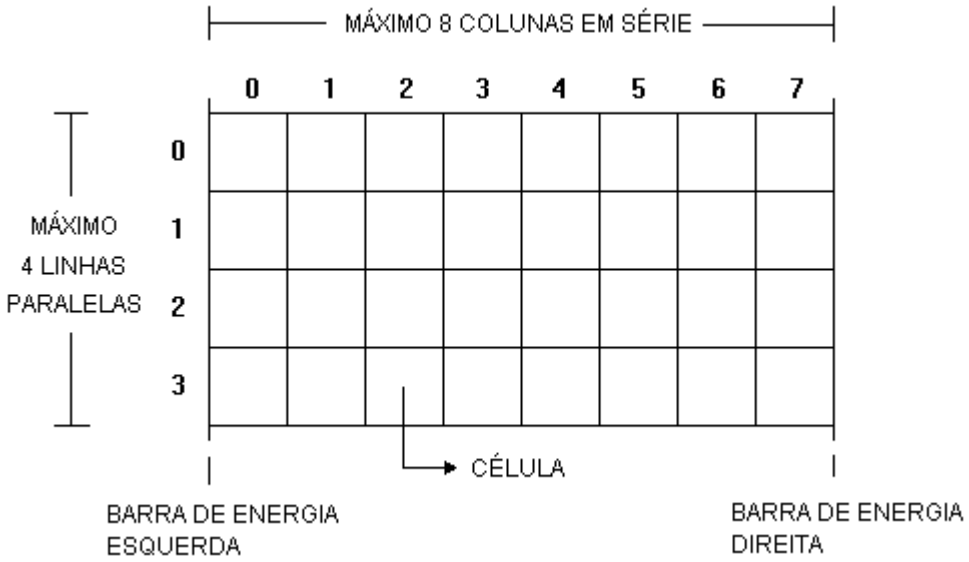


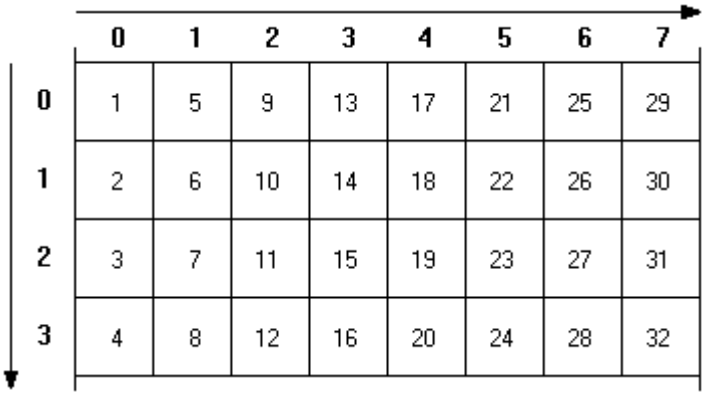
Figura 2-1. Formato de uma lógica

As duas linhas laterais da lógica representam barras de energia entre as quais são colocadas as instruções a serem executadas.

Estão disponíveis para a programação instruções simbólicas tipicamente encontradas em diagramas, tais como contatos, bobinas, ligações e instruções representadas em caixas, como temporizadores, contadores e aritméticas.

A lógica deve ser programada de forma que bobinas e entradas das instruções de caixas sejam "energizadas" a partir do fechamento de um fluxo de "corrente" da esquerda para a direita entre as duas barras, através de contatos ou das saídas das caixas interligados. Entretanto, o fluxo de "corrente elétrica" simulado em uma lógica flui somente no sentido da barra de energia esquerda para a direita, diferentemente dos esquemas elétricos reais. O conceito utilizado simplifica o projeto lógico de relés, uma vez que não é necessário a preocupação com caminhos de fuga de corrente.

O processamento das instruções de uma lógica é realizado em colunas, desde a coluna 0 até a 7. Uma coluna é processada na ordem seqüencial de suas linhas, desde a linha 0 até a linha 3. A figura a seguir mostra a ordem de processamento das células da lógica. O número existente dentro de cada célula indica a sua ordem de processamento.



	0	1	2	3	4	5	6	7
0	1	5	9	13	17	21	25	29
1	2	6	10	14	18	22	26	30
2	3	7	11	15	19	23	27	31
3	4	8	12	16	20	24	28	32

Figura 2-2. Ordem de processamento das células na lógica

Operandos

Operandos são elementos utilizados pelas instruções do MasterTool XE na elaboração de um programa aplicativo. Os operandos podem ser valores constantes, definidos no momento da

programação, ou variáveis, identificadas através de um endereço ou de um tag, com valores possíveis de serem alterados durante a execução do programa aplicativo.

Identificação de um Operando pelo Endereço

A identificação e utilização de um operando pelo seu endereço é caracterizada pelo caractere % como primeiro caractere do nome. O restante do nome utilizado deve seguir às regras de formatação de endereço de operandos.

O formato de cada operando pode ser visto na seção **Identificação dos Operandos Simples** e nas subseqüentes, neste mesmo capítulo.

Identificação de um Operando pelo Tag

A identificação e utilização de um operando pelo seu tag é caracterizada pela utilização de um nome, com até 25 caracteres (alfanuméricos), que pode ser atribuído a qualquer operando, exceto constantes. Este nome passa a representar o operando nos processos de programação, monitoração, depuração e documentação de um programa aplicativo.

ATENÇÃO: O MasterTool XE não permite a utilização de tags para operandos do tipo constante (%KM, %KD, %KF e %KI).

Ex.:

Atribui-se o tag **CONT1** ao operando **%M0000**.

Sempre que o operando **%M0000** necessite ser utilizado na edição do programa aplicativo, pode-se utilizar o seu tag **CONT1**.

ATENÇÃO: A escolha do nome do tag para o operando deve refletir ao máximo a função que o conteúdo do operando executa no programa aplicativo. Ex.: **TANQUE1**, armazena o volume do tanque 1.

ATENÇÃO: A identificação de um operando pelo seu endereço poderá ser feita sempre, uma vez que todo operando possui um endereço. A identificação de um operando pelo seu tag, somente poderá ser feita após a atribuição do tag a um operando.

Para maiores informações sobre criação e atribuição de tags para operandos, ver o Manual de Utilização do MasterTool XE.

ATENÇÃO: Os operandos também podem ser visualizados através de seu wire-info associado. No entanto, um operando não pode ser forçado ou monitorado digitando-se o wire-info ao invés do tag ou endereço.

Operandos Utilizados no MasterTool XE

São mostrados na tabela abaixo os operandos disponíveis no MasterTool XE:

Tipo	Operando
%E	Relés de Entrada
%S	Relés de Saída
%R	Endereço no Barramento
%A	Relés Auxiliares
%M	Memórias
%D	Decimais
%F	Reais
%I	Inteiros
%KM	Constantes Memórias
%KD	Constantes Decimais
%KF	Constantes Reais
%KI	Constantes Inteiros
%TM	Tabelas Memórias
%TD	Tabelas Decimais
%TF	Tabelas Reais
%TI	Tabelas Inteiros

Tabela 2-1. Operandos utilizados no MasterTool XE

Os operandos dividem-se em 3 grupos:

- Operandos simples
- Operandos constante
- Operandos tabela

Identificação dos Operandos Simples

Os operandos simples são utilizados como variáveis de armazenamento de valores no programa aplicativo. Conforme a instrução que os utilizam, eles podem ser referenciados na sua totalidade ou em uma subdivisão (uma parte do operando). As subdivisões de operandos podem ser **palavra**, **octeto**, **nibble** ou **ponto**.

O formato geral de um operando simples pode ser visto na figura abaixo:

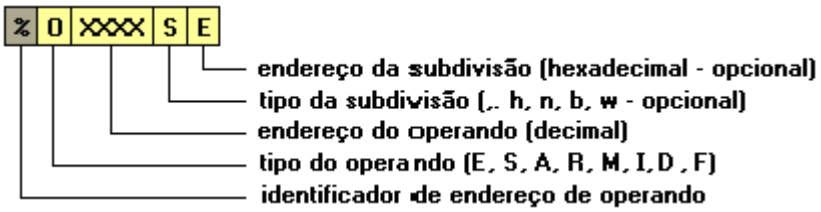


Figura 2-3. Formato de um operando simples

Tipo do operando:

- %E - entrada
- %S - saída
- %A - auxiliar
- %M - memória
- %I - inteiro
- %D - decimal
- %F - real

Tipo da subdivisão:

- . - ponto da palavra baixa (1 ponto)
- h - ponto da palavra alta (1 ponto) vide operandos decimais e reais
- n - nibble (4 pontos)
- b - octeto (8 pontos)
- w - palavra (16 pontos)

Exemplos de endereços:

- %E0002.3 - ponto 3 do operando de entrada 2
- %S0004.7 - ponto 7 do operando de saída 4
- %A0039n1 - nibble 1 do operando auxiliar 39
- %A0045 - octeto auxiliar 45
- %I0234 - operando inteiro 234
- %M0205 - operando memória 205
- %M0205b0 - octeto 0 da memória 205
- %D0029 - operando decimal 29
- %D0034w1 - palavra 1 do operando decimal 34
- %F0001 - operando real 1

Identificação de Operandos Constante

Os operandos constante são utilizados para a definição de valores fixos durante a edição do programa aplicativo.

O formato geral de um operando constante pode ser visto na figura a seguir:

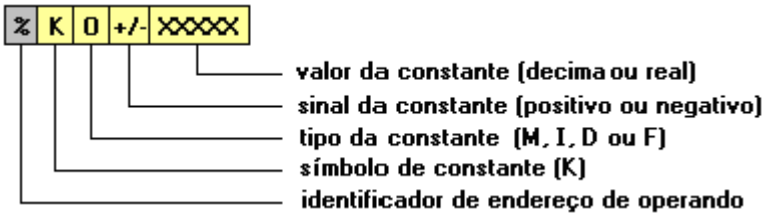


Figura 2-4. Formato de um operando constante

Tipo da constante:

- %M - memória
- %I - inteiro
- %D - decimal
- %F - real

Exemplos:

- %KM+05172 - constante memória positiva
- %KI-1 - constante inteira negativa
- %KD-0974231 - constante decimal negativa

- %KF+0153.78 - constante real positiva

Identificação dos Operandos Tabela

Tabelas de operandos são conjuntos de operandos simples, constituindo arranjos unidimensionais. São utilizados índices para determinar a posição da tabela que se deseja ler ou alterar. São possíveis tabelas de operandos memória, inteiro, decimal ou real.

O formato geral de um operando tabela pode ser visto na figura abaixo:

Erro! Não é possível criar objetos a partir de códigos de campo de edição.

Figura 2-5. Formato de um operando tabela

Tipo da tabela:

- %TM - memória
- %TI - inteiro
- %TD - decimal
- %TF - real

Exemplos:

- %TM0026 - tabela memória 26
- %TI0020 - tabela inteiro 20
- %TD0015 - tabela decimal 15
- %TF0069 - tabela real 69

Operandos %E - Relés de Entrada

São operandos usados para referenciar pontos de módulos digitais de entrada. Sua quantidade é determinada pelo número de módulos de E/S que estão dispostos nos bastidores que compõem o sistema.

Os operandos %E são normalmente utilizados em instruções binárias (contatos, bobinas) e de movimentação. Ocupam um byte de memória (8 bits), armazenando os valores dos pontos diretamente em cada bit. Os valores dos operandos são armazenados na memória interna do microprocessador, não utilizando o espaço disponível ao programa aplicativo.

Os formatos dos operandos %E podem ser vistos na figura abaixo:

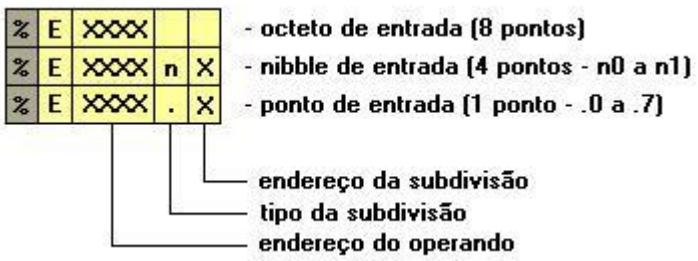


Figura 2-6. Formatos dos operandos %E

Exemplos:

- %E0018.6 - ponto 6 do octeto de entrada 18
- %E0021n0 - nibble 0 do octeto de entrada 21

- %E0025 - octeto de entrada 25

Limites:

- Mínimo: 0
- Máximo : 255

Operandos %S - Relés de Saída

São operandos usados para referenciar pontos de módulos digitais de saída. Sua quantidade é determinada pelo número de módulos de E/S que estão dispostos nos bastidores que compõem o sistema.

Os operandos %S são utilizados em instruções binárias (contatos, bobinas) e de movimentação. Ocupam um byte de memória (8 bits), armazenando os valores dos pontos diretamente em cada bit. Os valores dos operandos são armazenados na memória interna do microprocessador, não utilizando o espaço disponível ao programa aplicativo.

Os formatos dos operandos %S podem ser vistos na figura abaixo:

Figura 2-7. Formatos dos operandos %S

Exemplos:

- %S0011.2 - ponto 2 do octeto de saída 11
- %S0010n1 - nibble 1 do octeto de saída 10
- %S0015 - octeto de saída 15

Limites:

- Mínimo: 0
- Máximo: 255

Operandos %A - Relés Auxiliares

Os relés auxiliares são operandos usados para armazenamento e manipulação de valores binários intermediários no processamento do programa aplicativo. Sua quantidade nos controladores é fixa (ver seção **Declaração de Operandos** neste mesmo capítulo).

Operandos %A são utilizados em instruções binárias (contatos, bobinas) e de movimentação. Ocupam um byte de memória (8 bits), armazenando valores diretamente em cada bit. Os valores dos operandos são armazenados na memória interna do microprocessador, não utilizando o espaço disponível ao programa aplicativo.

Os formatos dos Operandos %A podem ser vistos na figura abaixo:

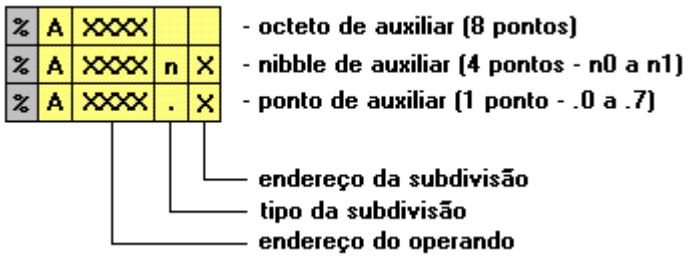


Figura 2-8. Formatos dos operandos %A

Exemplos:

- %A0032.7 - ponto 7 do auxiliar de saída 32
- %A0087n1 - nibble 1 do auxiliar de saída 87
- %A0024 - octeto auxiliar 24

Limites:

- Mínimo: 0
- Máximo: 255

Operandos %R - Endereços no Barramento

São operandos usados para referenciar pontos ou octetos no barramento de módulos de entrada e saída do controlador. Estes operandos representam apenas endereços do barramento, não armazenando valores nem ocupando espaço de memória. São utilizados em algumas instruções ou funções que realizam acessos a módulos.

Os formatos dos operandos %R podem ser vistos na figura abaixo:

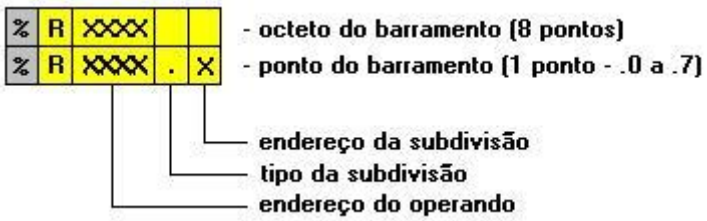


Figura 2-9. Formatos dos operandos %R

Nos barramentos 0 e 1 do CP AL-2004, cada posição no barramento corresponde a 8 octetos de operandos %R. Desta forma, na posição 0 tem-se os operandos %R0256 a %R0257; na posição 1, %R0258 a %R0259 e assim por diante.

Para se obter o primeiro operando de uma posição do barramento, basta calcular:

Endereço do Octeto = Posição no Barramento * 8

Nos barramentos de 2 a 9 do AL-2004, cada posição no barramento corresponde a 2 octetos de operandos %R. Desta forma, na posição 0 tem-se os operandos %R0000 e %R0001; na posição 1, %R0002 e %R0003, e assim por diante.

Para se obter o primeiro operando de determinada posição do barramento, basta calcular:

Endereço do Octeto = Posição no Barramento X 2

Os endereços para cada posição do barramento são automaticamente mostrados na janela de declaração do barramento na coluna Endereços.

Exemplos:

- %R0026 - octeto 26 do barramento
- %R0015.7 - ponto 7 do octeto 15 do barramento

Limites:

- Mínimo: 0
- Máximo: 32767

Operandos %M - Memórias

Os operandos %M são usados para processamento numérico, armazenando valores em precisão simples, com sinal.

Os formatos dos operandos %M podem ser vistos na figura abaixo:

Figura 2-10. Formatos dos operandos %M

A quantidade de operandos memória é configurável na declaração do módulo de configuração,, sendo o limite máximo dependente do modelo de CP em uso (ver seção **Declaração de Operandos** neste mesmo capítulo).

Os operandos %M são utilizados em instruções de movimentação, de comparação, aritméticas, de contagem, de temporização e de conversão. Podem ser utilizados em contatos, da mesma forma que os operandos %E, %S e %A. Estes operandos ocupam dois bytes de memória (16 bits), armazenando o valor no formato complemento de dois (2'), conforme a figura abaixo:

Figura 2-11. Formato do operando memória

Exemplos:

- %M0032 - memória 32
- %M0072n1 - nibble 1 da memória 72
- %M0084.F - ponto 15 da memória 84

Limites:

- Mínimo: -32768
- Máximo: 32767

Operandos %D - Decimais

Os operandos %D são usados para processamento numérico, armazenando valores em formato BCD com até 7 dígitos e sinal.

Os formatos dos operandos %D podem ser vistos na figura a seguir:

Figura 2-12. Formatos dos operandos %D

A quantidade de operandos decimal é configurável na declaração do módulo de configuração, sendo o limite máximo dependente do modelo de CP em uso (ver seção **Declaração de Operandos** neste mesmo capítulo).

Os operandos %D são utilizados em instruções de movimentação, de comparação, aritméticas, e de conversão. Podem ser utilizados em contatos, da mesma forma que os operandos %E, %S e %A. Estes operandos ocupam quatro bytes de memória (32 bits), armazenando o valor no formato BCD (cada dígito ocupa 4 bits), com sinal, conforme a figura a seguir:

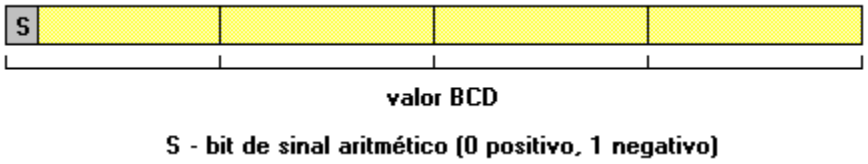


Figura 2-13. Formato do operando decimal

Exemplos:

- %D0041 - decimal 41
- %D0023b2 - octeto 2 do decimal 23
- %D0059n6 - nibble 6 da memória 59
- %D0172hA - ponto 10 da palavra 1 da memória 172

Limites:

- Mínimo: -9999999
- Máximo: 9999999

Operandos %F - Reais

Os formatos dos operandos %F podem ser vistos na figura a seguir:

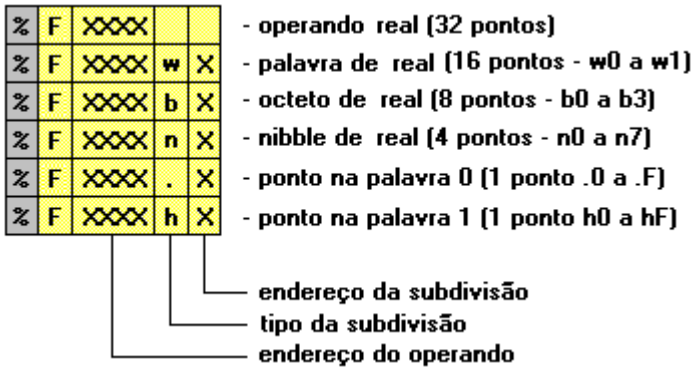


Figura 2-14. Formatos dos operandos %F

A quantidade de operandos real é configurável na declaração do módulo de configuração, sendo o limite máximo dependente do modelo de CP em uso (ver seção **Declaração de Operandos** neste mesmo capítulo).

Os operandos %F são usados para processamento numérico, armazenando valores em 32 bits com ponto flutuante, precisão simples e sinal, conforme a norma IEEE 754. Estes operandos ocupam quatro bytes de memória (32 bits), armazenando o valor conforme a figura a seguir:

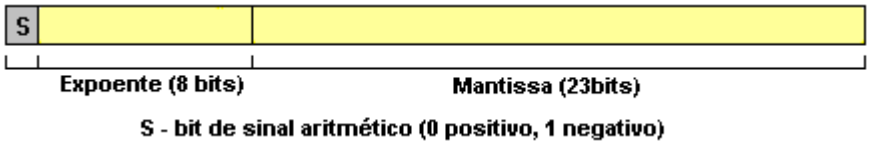


Figura 2-15. Formato do operando real

O valor de um operando real (%F) é obtido pela seguinte expressão:

$$\text{Valor} = \text{Sinal} \times 1,\text{Mantissa} \times 2^{(\text{Expoente} - 127)}$$

Sendo assim, a faixa de valores armazenáveis vai de -3,4028234663852886E+38 a 3,4028234663852886E+38.

Valores cujo módulo é maior que zero e menor que 1,1754943508222875E-38 são tratados como zero pelos CPs por serem muito pequenos. Os CPs não tratam os seguintes casos especiais previstos na norma: números não normalizados, infinito e NANs (not a number).

Exemplo:

- %F0032 - real 32

Limites:

- Mínimo: -3.4028235e+38
- Máximo: 3.4028235e+38

Operandos %I - Inteiros

Os operandos %I são usados para processamento numérico, armazenando valores em precisão simples, com sinal. A diferença básica entre este tipo de operando e o operando Memória %M, é que o operando Inteiro %I possui 32 bits.

Os formatos dos operandos %I podem ser vistos na figura a seguir:

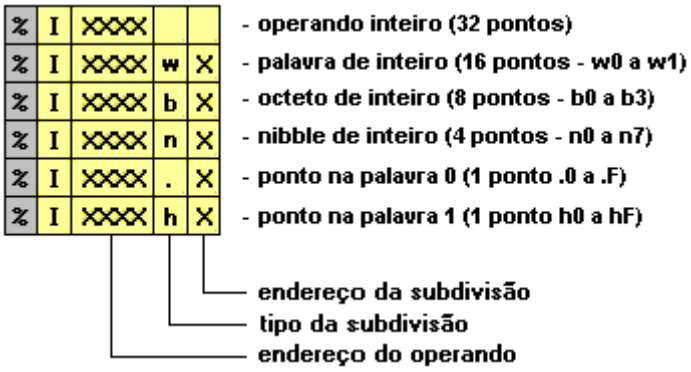


Figura 2-16. Formatos dos operandos %I

A quantidade de operandos inteiro é configurável na declaração do módulo de configuração, sendo o limite máximo dependente do modelo de CP em uso (ver seção **Declaração de Operandos** neste mesmo capítulo).

Os operandos %I são utilizados em instruções de movimentação, de comparação, aritméticas, e de conversão. Estes operandos ocupam quatro bytes de memória (32 bits), com sinal, conforme a figura a seguir:

Figura 2-17. Formato do operando inteiro

Exemplos:

- %I0041 - inteiro 41
- %I0023b2 - octeto 2 do inteiro 23
- %I0059n6 - nibble 6 do inteiro 59
- %I0172hA - ponto 10 da palavra 1 do inteiro 172

Limites:

- Mínimo: -2147483648
- Máximo: 2147483647

Operandos %KM, %KD, %KF e %KI - Constantes

São operandos usados para a definição de valores fixos na elaboração do programa aplicativo. Existem tipos de constante, %KM, %KI, %KD e %KF, cada um seguindo um formato diferente de representação de valores, sendo idênticos aos operandos %M, %I, %D e %F, respectivamente.

O formato dos operandos constantes pode ser visto na figura a seguir:

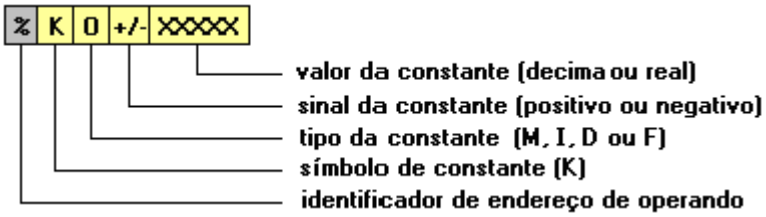


Figura 2-18. Formato dos operandos constantes

Estes operandos são utilizados em instruções de movimentação, de comparação, aritméticas, de contagem e de temporização.

Exemplos:

- %KM+00241 - constante memória + 241
- %KI+2000000000 - constante inteiro 2 bilhões ou 2 x 10^9
- %KD-0019372 - constante decimal - 19.372
- %KF+0125.78 - constante real + 125.78
- %KF+3.1415E23 - constante real 3.1415 x 10^{23}

Operandos %TM, %TD, %TF e %TI - Tabelas

Tabelas de operandos são conjuntos de operandos simples, constituindo arranjos unidimensionais com a finalidade de armazenar valores numéricos. Cada tabela possui uma quantidade de posições configuráveis, onde cada posição pode conter exatamente os mesmos valores de um operando %M, %I, %D ou %F, se a tabela for do tipo %TM, %TI, %TD ou %TF, respectivamente.

O formato dos operandos tabelas pode ser visto na figura a seguir:

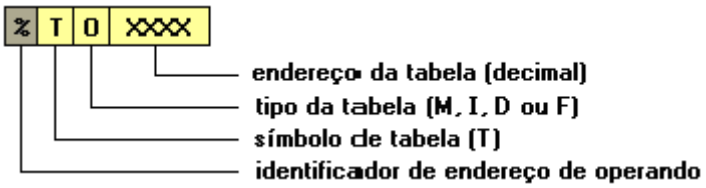


Figura 2-19. Formato dos operandos tabelas

A quantidade de tabelas e o número de posições de cada uma são configuráveis na declaração do módulo de configuração. Podem ser definidas até 255 tabelas totais e, no máximo, 255 posições em cada tabela, respeitando-se o limite da memória de operandos do CP.

As tabelas são utilizadas em instruções de movimentação.

Limites:

- Mínimo: 0
- Máximo: 254

Acesso Indireto

Esta forma de acesso é usada em conjunto com um operando memória %M para referenciar indiretamente outros operandos do sistema.

O sinal *, colocado na frente de um tipo de operando, indica que este é referenciado pelo endereço contido na memória especificada à esquerda do sinal.

O formato do acesso indireto pode ser visto na figura a seguir:

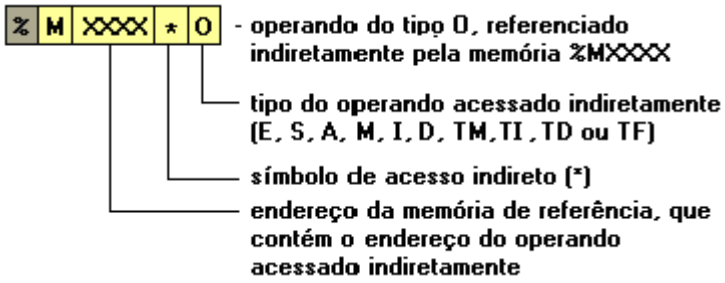


Figura 2-20. Formato de um acesso indireto

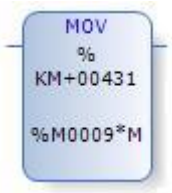
No MasterTool XE, o acesso indireto às tabelas é representado sem o asterisco.

O acesso indireto é utilizado em instruções de movimentação, de comparação, de contagem e de temporização.

Exemplos:

- %M0043*E - octeto de entrada referenciada indiretamente pela memória 43
- %M1824*A - octeto auxiliar referenciado indiretamente pela memória 1824
- %M0371TD - tabela de decimais referenciada indiretamente pela memória 371
- %M0009*M - operando memória referenciado indiretamente pela memória 9
- %F0045*M - operando real referenciado indiretamente pela memória 45

Exemplo:



Esta instrução movimenta o valor +431 para o operando memória cujo endereço é o valor correntemente armazenado em %M0009. Se %M0009 contiver o valor 32, então o valor +431 será armazenado em %M0032. Se %M0009 contiver o valor 12 então o valor constante será armazenado em %M0012.

ATENÇÃO: É responsabilidade do programa aplicativo que o valor contido na memória de referência (%M0009, no exemplo) represente endereços válidos, não contendo valores negativos ou acima dos endereços existentes para o tipo de operando referenciado indiretamente. As instruções não realizam os acessos indiretos inválidos, normalmente possuindo um sinal de saída para a indicação do erro.

Se no programa do exemplo anterior houvessem sido declarados 256 operandos %M, o valor de %M0009 deveria estar entre 0 e 255 para que a instrução fosse corretamente executada. Caso o valor não estivesse nesta faixa, o acesso não seria realizado.

Declaração de Operandos

Os operandos %E, %S e %A ocupam áreas de memórias próprias, permanentemente reservadas no microprocessador do CP. A quantidade destes operandos nos controladores, portanto, é constante.

Os operandos %R não ocupam espaço em memória, sendo apenas endereços para o acesso aos barramentos.

Por representarem valores fixos, os operandos constante (%KM, %KI, %KD e %KF) também não ocupam espaço em memória, sendo armazenados no próprio programa aplicativo na etapa de programação. Não há limites no número de operandos constante utilizados no programa.

Pode-se declarar a quantidade de operandos %M, %I, %D, %F, %TM, %TI, %TD e %TF, ocupando estes uma área de memória RAM própria do CP em uso. A tabela a seguir mostra a capacidade máxima de memória para o armazenamento destes operandos em cada controlador. Os operandos %E, %S e %A não ocupam esta área.

A declaração dos operandos é realizada através da janela de edição do módulo de configuração do MasterTool XE, sendo armazenada no módulo configuração. A quantidade de operandos declarada deve se adequar à capacidade máxima de memória disponível. Ver itens **Configurando Operandos Simples, Configurando Operandos Tabelas e Configurando Operandos Retentivos** no Manual de Utilização do MasterTool XE.

ATENÇÃO:
Deve-se declarar uma quantidade mínima de operandos memória (%M) que comporte os bytes de diagnóstico utilizados nos módulos do barramento.

A reserva dos operandos %M, %I, %D e %F é realizada em blocos de 256 bytes. No caso de operandos memória, esta quantidade corresponde a 128 operandos. Em operandos decimais e reais, correspondem a 64 operandos.

Os operandos %TM, %TI, %TD e %TF são declarados informando-se o número de tabelas necessário para cada tipo e o número de posições que cada tabela contém. É possível a definição de até 255 tabelas totais e até 255 posições para uma tabela, respeitando-se o limite da memória RAM de operandos.

A tabela a seguir, mostra o espaço de memória ocupado por cada tipo de operando e onde os seus valores são armazenados.

Operando	Ocupação de Memória	Localização
%E – entrada	1 byte	Microprocessador
%S – saída	1 byte	Microprocessador
%A – auxiliar	1 byte	Microprocessador
%KM – constante M	-	-
%KI – constante I	-	-
%KD – constante D	-	-
%KF – constante F	-	-
%M – memória	2 bytes	RAM de operandos
%I – inteiro	4 bytes	RAM de operandos
%D – decimal	4 bytes	RAM de operandos
%F – real	4 bytes	RAM de operandos
%TM – tabela M	2 bytes por posição	RAM de operandos
%TI – tabela I	4 bytes por posição	RAM de operandos
%TD – tabela D	4 bytes por posição	RAM de operandos
%TF – tabela F	4 bytes por posição	RAM de operandos

Tabela 2-2. Ocupação de memória e localização dos operandos

Operandos Retentivos

Operandos retentivos são operandos que têm seus valores preservados quando o CP é desenergizado (desligado). Os operandos não retentivos têm o seu valor zerado no momento em que o controlador programável é ligado.

Todos os operandos tabela são sempre retentivos. É possível configurar a quantidade de operandos %M (memória), %I (inteiro), %D (decimal), %F (real), %S (saída) e %A (auxiliar) retentivos.

Os operandos de entrada (%E) também são sempre retentivos. Quando estes operandos estão associados a módulos do barramento local, garante-se que terão seu valor atualizado antes mesmo do primeiro ciclo de execução. Já quando associados a módulos do barramento remoto (PROFIBUS, MODBUS, etc.) deve-se considerar um tempo de latência até a atualização dos mesmos.

Os operandos retentivos são configurados a partir dos últimos endereços até os iniciais, obedecendo a mesma regra dos operandos simples. Ou seja, a reserva é realizada em blocos de 256 bytes para operandos numéricos. A declaração dos operandos %S e %A é realizada de octeto em octeto.

Por exemplo, se existem 512 operandos %M declarados (%M0000 a %M0511) e deseja-se que 128 desses operandos sejam retentivos, serão considerados retentivos os operandos %M0384 ao %M0511.

Instruções

Os CPs ALTUS utilizam a linguagem de relés e blocos para a elaboração do programa aplicativo, cuja principal vantagem, além de sua representação gráfica, é ser similar a diagramas de relés convencionais.

A programação desta linguagem, realizada através do MasterTool XE, utiliza um conjunto de poderosas instruções.

As instruções do MasterTool XE podem ser divididas em 7 grupos:

RELÉS contendo as instruções:

- **RNA** contato normalmente aberto
- **RNF** contato normalmente fechado
- **BOB** bobina simples
- **BBL** bobina liga
- **BBD** bobina desliga
- **SLT** bobina de salto
- **PLS** relé de pulso
- **RM** relé mestre
- **FRM** fim de relé mestre

MOVIMENTADORES contendo as instruções:

- **MOV** movimentação de operandos simples
- **MOP** movimentação de partes de operandos
- **MOB** movimentação de blocos de operandos
- **MOT** movimentação de tabelas de operandos
- **MES** movimentação de entradas ou saídas
- **CES** conversão de entradas ou saídas
- **AES** atualização de entradas ou saídas
- **CAB** carrega bloco de constantes

ARITMÉTICOS contendo as instruções:

- **SOM** soma
- **SUB** subtração
- **MUL** multiplicação
- **DIV** divisão
- **AND** função "e" binário entre operandos
- **OR** função "ou" binário entre operandos
- **XOR** função "ou exclusivo" binário entre operandos
- **CAR** carrega operando
- **=** igual
- **<** menor
- **>** maior

CONTADORES contendo as instruções:

- **CON** contador simples
- **COB** contador bidirecional
- **TEE** temporizador na energização
- **TED** temporizador na desenergização

CONVERSORES contendo as instruções:

- **B/D** conversão binário - decimal
- **D/B** conversão decimal - binário
- **A/D** conversão analógico - digital
- **D/A** conversão digital - analógico

GERAIS contendo as instruções:

- **LDI** liga ou desliga pontos indexados
- **TEI** teste de estado de pontos indexados
- **SEQ** seqüenciador
- **CHP** chama módulo procedimento
- **CHF** chama módulo função
- **ECR** escrita de operandos em outro CP
- **LTR** leitura de operandos de outro CP
- **LAI** libera atualização de imagem de operandos
- **ECH** escrita de operandos em outro CP
- **LTH** leitura de operandos em outro CP
- **LAH** libera atualização de imagem de operandos.

LIGAÇÕES contendo as instruções:

- **LGH** ligação horizontal
- **LGV** ligação vertical
- **NEG** ligação negada

Restrições Quanto ao Posicionamento das Instruções

Existem regras a serem respeitadas quanto ao posicionamento das instruções nas 8 colunas da lógica. Pode-se dividir as instruções em três categorias:

Instruções que podem ser editadas somente na coluna 7:

- **BOB** bobina simples
- **BBL** bobina liga
- **BBD** bobina desliga
- **SLT** bobina de salto
- **RM** relé mestre
- **FRM** fim de relé mestre

Instruções que podem ser editadas nas colunas 0 a 6:

- **RNA** relé normalmente aberto
- **RNF** relé normalmente fechado
- **PLS** relé de pulso
- **LGH** ligação horizontal
- **LGV** ligação vertical
- **NEG** ligação negada
- **DIV** divisão
- **MOB** movimentação de blocos de operandos
- **>** maior

- < menor
- = igual
- **SEQ** seqüenciador
- **CHF** chama módulo função
- **ECR** escrita de operandos em outro CP
- **LTR** leitura de operandos de outro CP

Instruções que podem ser editadas em todas as colunas:

- **MOV** movimentação de operandos simples
- **MOP** movimentação de partes de operandos
- **MOT** movimentação de tabelas de operandos
- **MES** movimentação de entradas ou saídas
- **CES** conversão de entradas ou saídas
- **AES** atualização de entradas ou saídas
- **CAB** carrega bloco de constantes
- **SOM** soma
- **SUB** subtração
- **MUL** multiplicação
- **AND** função 'e' binário entre operandos
- **OR** função 'ou' binário entre operandos
- **XOR** função 'ou exclusivo' binário entre operandos
- **CON** contador simples
- **COB** contador bidirecional
- **TEE** temporizador na energização
- **TED** temporizador na desenergização
- **B/D** conversão binário - decimal
- **D/B** conversão decimal - binário
- **CAR** carrega operando
- **LDI** liga ou desliga pontos indexados
- **TEI** teste de estado de pontos indexados
- **CHP** chama módulo procedimento
- **LAI** libera atualização de imagem
- **A/D** conversão analógico - digital
- **D/A** conversão digital - analógico

Projeto de Programação

Estruturação de um Projeto de Programação

Funcionalmente, um projeto de programação pode ser visto como uma coleção de módulos utilizados para realizar uma tarefa especifica e é também conhecido como programa aplicativo. Isto permite uma visão hierárquica do projeto com a criação de sub-rotinas e funções.

Os módulos são chamados para a execução pelo software executivo (sistema operacional do CP) ou por outros módulos através de instruções apropriadas. Quando armazenado em disco, o projeto de programação corresponde a um conjunto de arquivos, sendo que cada arquivo contém um módulo, denominados como mostra a figura a seguir:

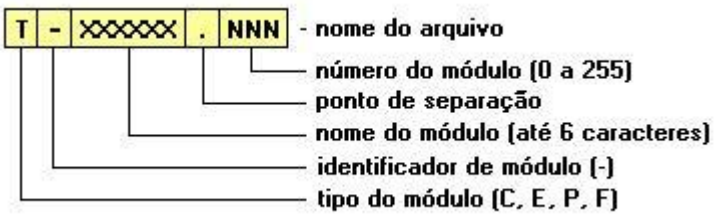


Figura 2-21. Formato do nome dos módulos em arquivo

Em alguns locais deste manual e na seção **Ajuda**, os módulos de programa são referenciados somente pelo seu tipo e número quando não for relevante o nome utilizado no mesmo.

Exemplo: **E018**

ATENÇÃO: O nome do arquivo correspondente a um módulo de programa não deve ser alterado através de outro aplicativo do Windows. Para alterar o nome de um arquivo, deve-se ler e salvar o mesmo com o nome desejado através do MasterTool XE.

Se o nome do arquivo for modificado através de outro aplicativo do Windows, poderá ser atribuído um nome inválido, fazendo com que o mesmo não possa mais ser lido pelo MasterTool XE ou carregado no CP.

Existem 4 tipos de módulos que podem fazer parte de um projeto de programação:

- **Módulo C** (Configuração): existe um módulo de configuração por projeto, contendo os parâmetros de configuração do CP (C000).
- **Módulo C Estendido** (Configuração): este módulo de configuração existe quando o usuário utiliza em seu projeto uma determinada característica do CP que necessita de um módulo de configuração estendido. Para maiores informações consultar o manual de utilização do MasterTool XE (C003 a C009).
- **Módulo E** (Execução): podem existir até 4 módulos de execução por projeto. Os mesmos são chamados somente pelo sistema operacional do CP (E000, E001, E018 e E020).
- **Módulo P** (Procedimento): podem existir até 200 módulos procedimento por projeto. Eles contêm trechos de programa aplicativo, sendo chamados por instruções colocadas em módulos de execução, procedimento ou função. Após serem executados, o processamento retorna para a instrução seguinte à de chamada. Os módulos P funcionam como sub-rotinas, não permitindo a passagem de parâmetros para o módulo chamado (P000 a P199).
- **Módulo F** (Função): podem existir até 229 módulos função por projeto. Eles contêm trechos de programa aplicativo escritos de forma genérica, permitindo a passagem de parâmetros para o módulo chamado, de forma a poderem ser reaproveitados em vários programas aplicativos diferentes. São semelhantes a instruções, podendo ser chamados por módulos de execução, procedimento ou função. (F000 a F228).

Módulo C - Configuração

O módulo C contém os parâmetros de configuração do CP. Sua criação é pré-requisito para a edição dos demais módulos do projeto de programação no MasterTool XE. A definição dos parâmetros contidos no mesmo é realizada através da janela de edição de módulo C. Para maiores detalhes sobre como configurar um módulo C, ver o Manual de Utilização do MasterTool XE.

Há somente um módulo C por projeto de programação, tendo como número 000.

Conteúdo de um módulo C:

- **Declaração do Barramento de módulos de E/S:** especifica a configuração dos módulos de E/S a serem utilizados no controlador programável, indicando a distribuição dos mesmos e módulos especiais no barramento do CP. A declaração dos módulos define, desta forma, o número de pontos e os endereços de E/S a serem utilizados no programa aplicativo. A mesma é realizada

através da janela de edição do módulo C. Para maiores informações sobre como configurar o barramento, ver Manual de Utilização do MasterTool XE.

- **Declaração de Operandos:** especifica o número de operandos simples e tabelas de operandos que serão utilizados no projeto de programação, dentro de cada tipo disponível. Permite também a definição da retentividade dos operandos, ou seja, quais operandos devem manter seu conteúdo com a falta de energia do sistema.
    - **Declaração de Operandos Simples**: permite a definição da quantidade de operandos Memória (%M), Inteiro (%I), Decimal (%D) e Real (%F). É realizada através da janela de edição de módulo C. Para maiores informações sobre como declarar operandos simples, ver Manual de Utilização do MasterTool XE.
    - **Declaração de Operandos Tabela**: permite a definição do número de tabelas de operandos Memória (%TM), de operandos Inteiro (%TI), de operandos Decimal (%TD), de operandos Real (%TF) e do número de posições de cada uma. Uma tabela representa um conjunto de operandos, sendo a sua definição realizada através da janela de edição de módulo C. Para maiores informações sobre como configurar operandos tabela, ver Manual de Utilização do MasterTool XE.
    - **Declaração de Operandos Retentivos:** especifica o número de operandos simples que são retentivos, dentro dos operandos já declarados. Operandos retentivos são aqueles que continuam com o seu conteúdo inalterado com a falta de energia do CP, sendo os não retentivos zerados com a reinicialização do sistema. Os operandos tabela são todos retentivos. A declaração é realizada através da janela de edição de módulo C. Para maiores informações sobre como configurar operandos retentivos, ver Manual de Utilização do MasterTool XE.
- **Declaração dos Parâmetros Gerais do CP:** são parâmetros genéricos necessários para o funcionamento do controlador programável, tais como o tipo de CP na qual o programa aplicativo será carregado, o período de chamada dos módulos acionados por interrupção e o tempo máximo de ciclo de varredura . Estes parâmetros são declarados através da janela de edição de módulo C. Para maiores informações sobre como configurar os parâmetros gerais, ver Manual de Utilização do MasterTool XE.
- **Declaração dos Parâmetros da Rede ALNET I:** especifica os diversos parâmetros necessários ao funcionamento da comunicação em rede ALNET I. Estes parâmetros são configurados na janela de edição de módulo C. Para maiores informações sobre como configurar parâmetros da rede ALNET I, ver Manual de Utilização do MasterTool XE.
- **Declaração dos Parâmetros da Rede ALNET II:** especifica os diversos parâmetros necessários ao funcionamento da comunicação em rede ALNET II, para os controladores programáveis que permitem o seu uso. Estes parâmetros são configurados na janela de edição de módulo C. Para maiores informações sobre como configurar parâmetros da rede ALNET II, ver Manual de Utilização do MasterTool XE
- **Declaração dos Parâmetros da Rede Ethernet:** especifica os diversos parâmetros necessários ao funcionamento da comunicação em rede Ethernet, para os controladores programáveis que permitem o seu uso. Estes parâmetros são configurados na janela de edição de módulo C. Para maiores informações sobre como configurar parâmetros da rede Ethernet, ver Manual de Utilização do MasterTool XE
- **Declaração dos Parâmetros da Rede de Sincronismo:** especifica os diversos parâmetros necessários ao funcionamento da comunicação com rede de sincronismo, para os controladores programáveis que permitem o seu uso. Estes parâmetros são configurados na janela de edição de módulo C. Para maiores informações sobre como configurar parâmetros da rede de sincronismo, ver Manual de Utilização do MasterTool XE

Módulo C Estendido – Configuração

Estes módulos contém configurações de determinadas características das CPs. Estes módulos são totalmente gerenciados pelo usuário, isto é, deve ser criado e apagado conforme necessidade do usuário. Isto se deve ao fato de que a quantidade deste tipo de módulo varia de acordo com cada aplicação, podendo não ter nenhum a ter até 7 módulos (C003 a C009).

Para maiores informações consultar Manual de Utilização do MasterTool XE.

Módulo E - Execução

Os módulos E contêm trechos do programa aplicativo, sendo chamados para a execução pelo software executivo. Existem diversos módulos E, diferenciando-se entre si pelo modo como são chamados à execução, conforme o seu número.

Tipos de módulos E:

- **E000 - Módulo de Inicialização:** é executado uma única vez, ao se energizar o CP ou na passagem de modo programação para execução com o MasterTool XE, antes da execução cíclica do módulo E001.
- **E001 - Módulo Seqüencial de Programa Aplicativo:** contém o trecho principal do programa aplicativo, sendo executado ciclicamente.
- **E018 - Módulo Acionado por Interrupção de Tempo:** o trecho de programa aplicativo colocado neste módulo é chamado para a execução em intervalos de tempo periódicos. Define-se o período de chamada do mesmo nos parâmetros gerais do módulo C, podendo ser escolhido entre 50 ms, 25 ms, 10 ms, 5 ms, 3,125 ms, 2,5 ms, 1,25 ms e 0,625 ms. Ao ser transcorrido o tempo programado, a execução seqüencial do programa aplicativo é interrompida e o módulo E018 é executado. Após o seu final, o sistema retorna a execução para o ponto do processamento seqüencial onde o módulo E0001 havia sido interrompido. O tempo continua a ser contado durante a chamada do módulo E018, devendo a sua execução ser o mais breve possível para não haver o aumento excessivo no tempo de ciclo do módulo E001.

ATENÇÃO:
O tempo de execução do módulo E018 não pode ser maior ou igual ao período de chamada. Caso isto aconteça, o CP entra em modo erro sendo exibida a mensagem **Reentrada no módulo E018**, na janela **Informações** (comando **Comunicação, Estado, Informações**).

ATENÇÃO:
O Módulo Acionado por Interrupção de Tempo (E018) poderá ser executado pela primeira vez depois de executar a E000.

- **E020 - Módulo Acionado pela Entrada de Interrupção:** Quando ocorrer uma transição de subida no sinal presente nesta entrada, a execução seqüencial do programa aplicativo é interrompida e o módulo E020 é executado. Após o seu final, o sistema retorna a execução para o ponto do processamento seqüencial onde o módulo E0001 havia sido interrompido. Se a entrada for acionada com muita freqüência, o tempo de execução do módulo E020 deve ser o mais breve possível, para não haver o aumento excessivo no tempo de ciclo do módulo E001.

ATENÇÃO:
O tempo de execução do módulo E020 não pode ser maior ou igual ao período de chamada. Caso isto aconteça, o CP entra em modo erro sendo exibida a mensagem **Reentrada no módulo E020**, na janela **Informações** (comando **Comunicação, Estado, Informações**).

ATENÇÃO:
O Módulo Acionado pela Entrada de Interrupção (E020) poderá ser executado pela primeira vez depois de executar a E000.

Módulo P - Procedimento

Os módulos P contêm trechos de programas aplicativos chamados a partir de módulos E, P ou F através da instrução CHP (Chama Procedimento).

Este tipo de módulo não possui passagem de parâmetros, sendo similar ao conceito de sub-rotina.

O número máximo de módulos deste tipo é 200 (P000 a P199).

O módulo P é útil para conter trechos de programas aplicativos que devem ser repetidos várias vezes no programa principal, sendo assim programados uma só vez e chamados quando necessário, economizando memória de programa.

Podem ser usados também para uma melhor estruturação do programa principal, dividindo-o em segmentos de acordo com a sua função e declarando-os em diversos módulos P. Neste caso, o módulo de execução contínua E001 somente chama os módulos P na seqüência desejada.

Exemplos:

- P-MECAN.000 - realiza o intertravamento mecânico da máquina
- P-TEMPER.001 - realiza o controle de temperaturas
- P-VIDEO.002 - realiza o interfaceamento homem-máquina
- P-IMPRES.003 - gerência a impressão de relatórios

Módulo F - Função

Os módulos F contêm trechos de programas aplicativos chamados a partir de módulos E, P ou F, através da instrução CHF (Chama Função).

Na chamada dos módulos F é possível a passagem de valores como parâmetros para o módulo chamado. Estes módulos são usualmente escritos de forma genérica para serem aproveitados por vários programas aplicativos, em linguagem de relés ou de máquina, sendo semelhantes às instruções da linguagem de relés. Os valores dos parâmetros são enviados e devolvidos através de listas de operandos existentes na instrução de chamada e no módulo F.

Na edição de um instrução CHF, devem ser definidas 2 listas de operandos que são utilizadas para:

- Enviar parâmetros para execução do módulo função (Entrada)
- Receber os valores retornados pelo módulo função (Saída)

Na edição do módulo função, também devem ser definidas 2 listas de operandos, utilizando o comando **Edição, Editar, Parâmetros** que são utilizados para:

- Receber parâmetros da instrução CHF (Entrada)
- Enviar valores de retorno para a instrução CHF (Saída)

A passagem de parâmetros é realizada através da cópia dos valores dos operandos declarados (passagem de parâmetros por valor). A figura, a seguir, apresenta o fluxo de dados entre a instrução CHF e o módulo função:

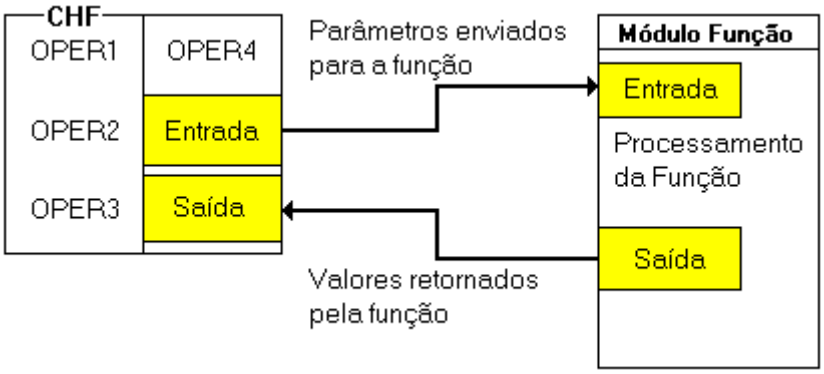


Figura 2-22. Passagem de parâmetros para o módulo F

Maiores informações a respeito da passagem de parâmetros podem ser encontradas na descrição da instrução CHF neste mesmo manual.

Exemplos:

- F-LINEAR.002 - executa a linearização de valores lidos de um sensor
- F-PID.033 - realiza cálculos para implementação de laço PID de controle

Estados de Operação do CP

Existem cinco estados ou modos de operação do CP: inicialização, execução, programação, ciclado e erro. O estado em que o controlador programável se encontra é indicado nos LEDs do painel frontal do CP, podendo também ser consultado pelo MasterTool XE, através da caixa de diálogo **Estado** (opções **Comunicação, Estado,** a partir do menu principal). Para obter informações específicas sobre os modos de operação, consultar o manual de utilização do controlador utilizado.
Estado Inicialização: o CP inicializa as diversas estruturas de dados de uso do programa executivo e realiza consistências no projeto de programação presente na memória. Este estado ocorre após a energização do controlador, passando após alguns segundos para o estado execução. Caso não exista programa aplicativo na memória, o CP passa para modo erro.

Enquanto o CP está inicializando, pode-se acionar o comando **Comunicação, Estado,** Programação, ou atalho equivalente na barra de ferramentas, fazendo com que o CP passe diretamente para o estado de programação, ao invés de executar o programa aplicativo. Este procedimento é útil para a reinicialização de CPs com programas contendo erros graves de programação.

Por exemplo, um módulo com um laço infinito de execução, programado com uma instrução de salto para uma lógica anterior, provoca o acionamento do circuito de cão-de-guarda do CP sempre que for ligada, após o estado de inicialização. Executando-se o procedimento anterior logo após a energização, o CP passa para o estado programação após inicializar, permitindo o apagamento ou a substituição do programa.

Estado Execução: normalmente o controlador programável se encontra neste estado, varrendo continuamente os pontos de entrada e atualizando os pontos de saída de acordo com a lógica programada. Este estado indica que o CP está executando corretamente um programa aplicativo.

Estado Programação: o programa aplicativo não é executado, não havendo a leitura dos pontos de entrada, sendo as saídas desativadas e a memória do CP é compactada. O CP permanece inoperante, esperando comandos do MasterTool XE. Este modo normalmente é utilizado para a carga dos módulos do projeto de programação pelo MasterTool XE, através do canal serial. Ao passar para estado execução ou ciclado a partir do estado programação, os operandos são zerados.

Estado Ciclado: quando em modo ciclado, o controlador programável não executa ciclicamente o módulo E001, permanecendo à espera de comandos do MasterTool XE. Cada comando **executa ciclo** acionado no MasterTool XE (opções **Comunicação, Estado, Executa Ciclo** a partir do menu

principal ou atalho equivalente) dispara uma única varredura do programa aplicativo (módulo E001), permanecendo o CP à espera de um novo comando após a execução da mesma. Quando o CP passa para modo ciclado, a contagem de tempo nos temporizadores pára, sendo os mesmos incrementados de uma unidade de tempo a cada duas varreduras executadas. As chamadas para o módulo de interrupção de tempo E018 não são realizadas neste modo. O módulo E020, acionado pela entrada de interrupção externa, continua sendo chamado neste modo.

Estado de Erro: indica que houve alguma anomalia no CP durante o processamento do projeto de programação. O tipo de erro ocorrido pode ser consultado através da caixa de diálogo (opções **Comunicação, Estado, Informações** a partir do menu principal), enquanto o CP estiver neste estado. A saída do estado de erro somente é possível passando-se o controlador programável para modo programação.

Em condições normais, o controlador programável pode estar nos modos execução, programação e ciclado, sendo esses modos acionados através de comandos do MasterTool XE (opções **Execução, Programação e Ciclado** da caixa de diálogo **Estado,** ou seus atalhos equivalentes na **Barra de Ferramentas**). Na ocorrência de alguma situação de funcionamento errôneo nestes modos, o CP passa para estado de erro. A recuperação do modo erro somente é possível passando-se o controlador programável para modo programação. A figura, a seguir, apresenta as possibilidades de troca de estados:

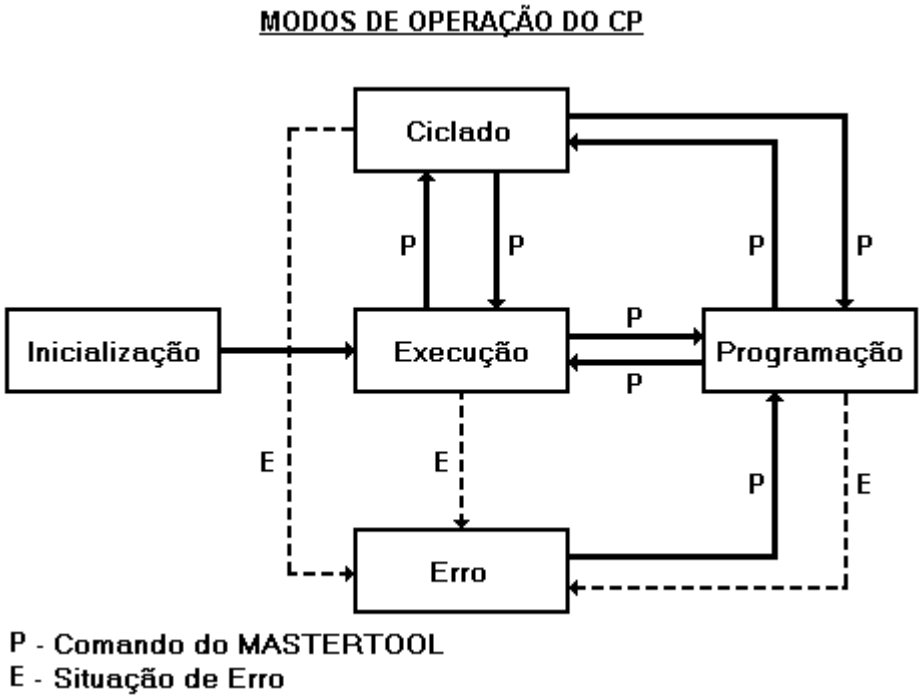


Figura 2-23. Estados de operação do CP

Nos modos execução, programação e ciclado é possível carregar e ler módulos do projeto de programação pelo canal serial do controlador programável, bem como monitorar e forçar quaisquer operandos utilizados. Essas operações não são possíveis caso o CP esteja em modo erro.

Os operandos que não são retentivos são zerados na passagem de modo programação para execução ou programação para ciclado, permanecendo os demais inalterados.

Execução do Projeto de Programação

Quando o CP é energizado ou após a passagem para modo execução, as inicializações do sistema são realizadas de acordo com o conteúdo do módulo C, sendo logo após executado o módulo E000 uma única vez. O controlador programável passa então para o processamento cíclico do módulo E001, atualizando as entradas e saídas e chamando o módulo E018, quando existente, a cada período de

tempo de interrupção programado. A figura abaixo mostra esquematicamente a execução do programa aplicativo.

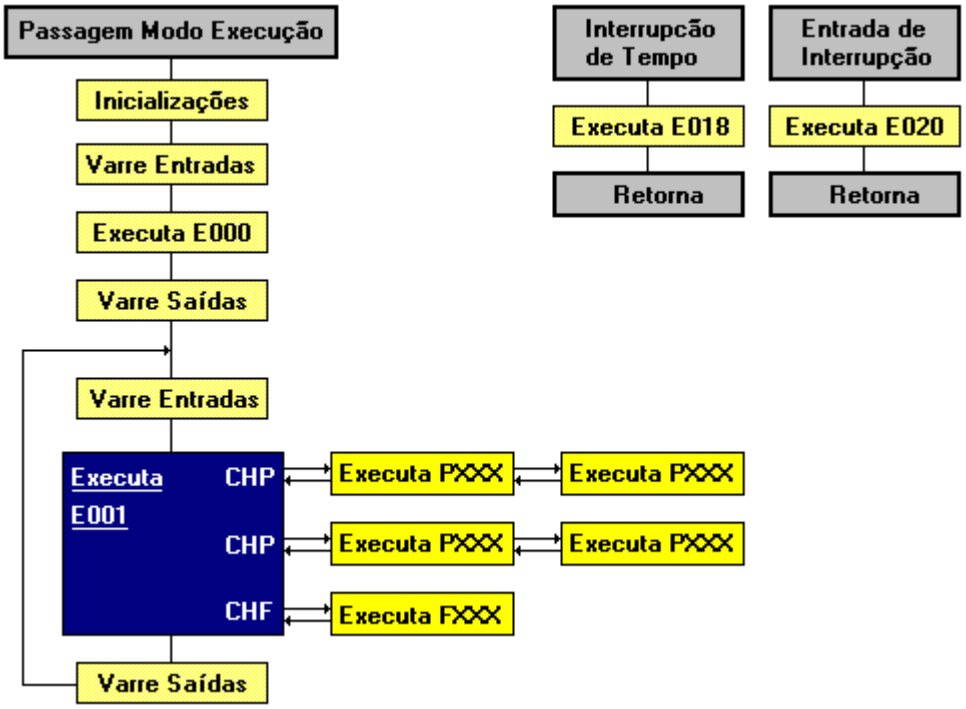


Figura 2-24. Execução do projeto de programação

ATENÇÃO: Os módulos E020 e E018 poderão ser executados pela primeira vez depois de executar o E000.

Elaboração de Projetos de Programação

Considerações Gerais

Um projeto de programação é composto ao menos por um módulo C (configuração) e um módulo E001 (execução). A condição mínima para a execução de um projeto de programação é a presença destes dois módulos no CP.

O primeiro passo para a edição de um projeto de programação no MasterTool XE é a criação ou a leitura de um projeto. O módulo de configuração do projeto é criado automaticamente na criação de um novo projeto, uma vez que neste módulo estão contidas as declarações dos módulos de entrada e saída e operandos utilizados em todo o projeto. Cada módulo que contenha trechos de programa aplicativo (E, P ou F) necessita que o módulo C esteja presente no MasterTool XE para que possa ser editado.

Após a criação ou leitura de um projeto, pode-se editar o mesmo adicionando módulos já existentes, criando módulos novos para o projeto ou excluindo módulos que já façam parte do projeto.

O MasterTool XE permite que vários módulos sejam carregados e permaneçam simultaneamente em sua memória.

Considerações sobre Operandos

Os diversos módulos que compõem um projeto de programação devem preferencialmente ter sido programados utilizando-se o mesmo módulo C. Caso um módulo já programado necessite ser usado em outro projeto de programação, os operandos utilizados pelo módulo devem obrigatoriamente estar declarados no módulo C do novo projeto.

Os operandos disponíveis no controlador programável são de uso comum a todos os módulos do projeto de programação presentes no CP (operandos globais). Em conseqüência deste fato, dois módulos quaisquer podem estar inadvertidamente acessando o mesmo operando, ocorrendo erros no funcionamento de ambos.

Ao elaborar um projeto de programação, deve-se reservar operandos em número suficiente para o mesmo, preferencialmente separando-os em grupos, cada grupo utilizado somente por um módulo. Os operandos utilizados nos módulos F programados em linguagem de relés e blocos também podem ser acessados por quaisquer outros módulos de programa presentes no CP, mesmo os operandos utilizados na passagem de parâmetros. Para garantir o seu caráter genérico, cada módulo F deve utilizar um grupo de operandos diferente dos demais utilizados no programa aplicativo.

Utilização dos Módulos P e F

Dentro de um módulo do projeto de programação podem ser colocadas as instruções de chamada de outros módulos. As instruções CHP e CHF chamam, respectivamente, módulos de procedimento e função. Elas realizam o gerenciamento das chamadas dos módulos, verificando a existência ou não dos mesmos no diretório do controlador programável, baseadas em seus tipos e números.

No CP AL-2004 existem 32 níveis de chamada. Ou seja, podem ser executadas até 32 chamadas consecutivas de módulos sem ser finalizada a execução de nenhum. Deve-se considerar que o módulo E018 (se existir) e os módulos por ele chamados também ocupam níveis de chamada.

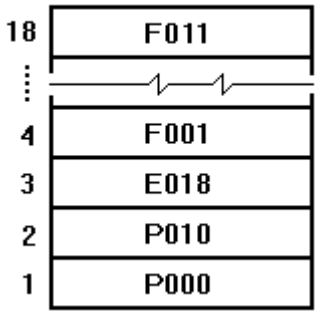


Figura 2-25. Número máximo de níveis de chamada de módulos

Quando o número máximo de chamadas acumuladas sem retorno for ultrapassado, o sistema não mais as realizará, prosseguindo com a execução normal do programa aplicativo. Nos casos de ocorrência de chamadas para módulos inexistentes ou o excesso do número de chamadas totais, são mostradas mensagens de advertência na janela Informações (opções Comunicação, Estado, Informações a partir do menu principal), pois estas situações poderão causar erros no processamento conforme a lógica programada.

É possível a chamada de um módulo por ele mesmo (programação por recursividade) tomando-se os cuidados necessários, ou seja, deve ser previsto no trecho de programa aplicativo com recursividade um momento em que não há mais chamadas para o mesmo módulo. Embora seja possível, o uso de tal procedimento não é aconselhável em controladores programáveis, devido ao grande tempo de processamento que um pequeno trecho de programa aplicativo pode necessitar para ser executado e à facilidade da ocorrência de laços infinitos de execução (loops).

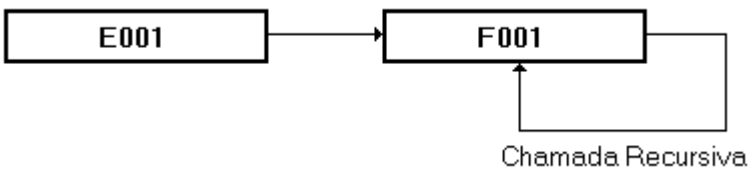


Figura 2-26. Chamada recursiva de módulos

Deve-se evitar a programação indevida de laços de chamada de módulos sem fim ("dead locks"). Caso um módulo do projeto de programação chame outro e este também realize uma chamada para o primeiro, se as instruções de chamada nos dois módulos não forem desabilitadas ambos permanecerão chamando-se mutuamente até a passagem do controlador programável para modo erro, por excesso de tempo de execução do programa aplicativo.

A mesma situação pode ocorrer com chamadas encadeadas entre diversos módulos, quando um módulo chamado volte a chamar algum módulo inicial da cadeia. Por exemplo, se o módulo P005 chamar o P002, este chamar P007 e este chamar novamente o P005, o processamento poderá permanecer neste laço se nenhuma instrução de chamada for desabilitada.

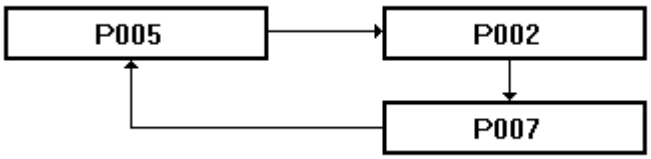


Figura 2-27. Laço de chamada de módulos

Utilização do Módulo E018

O módulo E018 deve ser utilizado quando for necessário um processamento rápido para alguns pontos de entrada e saída do controlador programável, como por exemplo, no sensoriamento de fins-de-curso em sistemas de posicionamento rápido. Para este caso deve ser empregada a instrução de atualização de pontos de E/S (AES), realizando dentro do módulo E018 um processamento semelhante a um laço completo de execução do programa principal. As entradas são lidas, o trecho de programa aplicativo desejado é executado e as saídas são atualizadas.

Da mesma forma, este módulo torna-se útil quando se deseja uma resposta dos acionamentos de saída após um tempo fixo dos estímulos das entradas, independente do tempo de varredura do programa principal, que pode ser variável. Essa característica é importante também em sistemas de controle de posição.

Outra aplicação para o módulo E018 é a geração de bases de tempo menores que 100 ms para o programa principal. Podem ser gerados temporizadores com precisão de 50 ms, 10 ms ou menos, se necessário, através do uso de instruções contadoras dentro do módulo de interrupção de tempo.

Este módulo é útil quando se deseja um controle preciso de tempos no processamento do CP.

Utilização do Módulo E020

Com o seu acionamento, o módulo E020 é chamado para a execução, realizando o processamento necessário e a atualização de pontos de saída através da instrução AES.

O módulo E020 também pode ser utilizado no acionamento de dispositivos ou procedimentos de segurança, dispositivos de frenagem ou outras aplicações que necessitem rapidez de atuação.

Caso o módulo E020 esteja presente no CP, este é chamado a cada acionamento da entrada. Se o programa aplicativo chamar o módulo F-CONT.005, este realiza a leitura e escrita do valor de contagem, incrementado a cada acionamento da entrada. Se desejado, pode-se utilizar esta entrada com ambas as funções, com o módulo F-CONT.005 contando o número de vezes que o módulo E020 foi acionado.

Cuidados no Utilização do Módulo E018

Alguns cuidados especiais são necessários na programação do módulo E018. Como o mesmo é chamado de modo síncrono a cada período fixo de tempo, interrompendo o processamento do módulo E001, o seu tempo de execução deve ser o mais breve possível para não aumentar excessivamente o tempo de ciclo total do programa aplicativo.

Se o intervalo entre as chamadas do módulo E018 for programado para 25 ms, por exemplo, e o seu tempo de execução for 20 ms, restarão somente 5 ms para a execução do programa principal antes

que o E018 seja chamado novamente. Esta situação aumenta de forma considerável o tempo de ciclo do módulo E001.

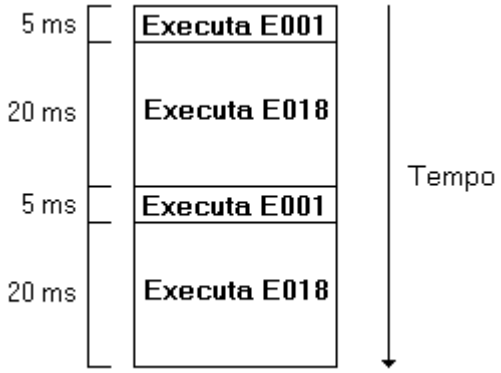


Figura 2-28. Cuidados na utilização do módulo E018

Caso a execução do módulo E018 demore mais do que o intervalo de tempo programado para suas chamadas, o CP passa para o estado de erro, sendo exibida a mensagem "Reentrada no módulo E018" na janela **Informações** (opções **Comunicação, Estado, Informações** a partir do menu principal). Nesta situação, deve-se aumentar o período de chamada utilizado ou diminuir o tempo de execução do módulo E018 para que o projeto de programação possa ser executado corretamente.

As instruções permanecem com o mesmo comportamento quando executadas dentro do módulo E018, exceto em relação a algumas características particulares. É o caso das instruções de temporização, que podem passar a contar menos ou mais tempo que o realmente transcorrido se utilizadas no E018, ou seja, as instruções TEE e TED não devem ser utilizadas no mesmo. O relé de pulso (PLS) aciona a sua saída durante uma execução do módulo E018, zerando a mesma na próxima chamada. As instruções CHP e CHF podem ser usadas da mesma forma como no programa principal, devendo os módulos acionados pelas mesmas obedecerem às mesmas regras de programação válidas para o módulo E018 propriamente dito. O número máximo de níveis de chamada de módulos utilizados dentro do módulo E018 deve ser acrescentado ao máximo nível empregado em E001, devendo esta soma ser menor que o limite do sistema (18 níveis).

Cuidados na Programação do Módulo E020

Alguns cuidados especiais são necessários na programação do módulo E020. O seu tempo de processamento deve ser breve, principalmente se a entrada de interrupção for acionada de forma freqüente, para não aumentar de forma excessiva o tempo de ciclo total do programa aplicativo.

Se a entrada de interrupção for acionada de forma periódica a cada 30 ms, por exemplo, e o tempo de execução do E020 for 25 ms, só restarão 5 ms para a execução do programa principal antes que o módulo seja chamado novamente. Esta situação aumenta de forma considerável o tempo de ciclo do módulo E001.

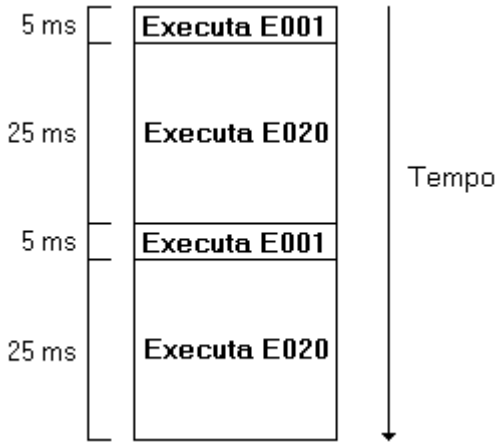


Figura 2-29. Cuidados na utilização do módulo E020

Caso o módulo E020 esteja sendo executado e ocorra novo acionamento na entrada de interrupção do CP, este acionamento é desconsiderado, continuando normalmente a execução do módulo. Esta situação não provoca a mudança para modo erro, permanecendo o CP em execução normal. Portanto, o CP ignora acionamentos da entrada rápida de interrupção que ocorram em tempos menores que o tempo de execução do E020.

As instruções continuam a ter o mesmo comportamento quando executadas dentro do módulo E020, exceto em relação a algumas características particulares.

A chamada do módulo depende do processo que está sendo controlado, não ocorrendo de forma periódica. Esta característica inviabiliza o uso dos temporizadores no E020, ou seja, as instruções TEE e TED não devem ser utilizadas no mesmo. O relé de pulso (PLS) aciona a sua saída durante uma execução do módulo E020, zerando a mesma na próxima chamada. As instruções CHP e CHF podem ser usadas da mesma forma como no programa principal, devendo os módulos acionados pelas mesmas obedecerem às mesmas regras de programação válidas para o módulo E020 propriamente dito. O número máximo de níveis de chamada de módulos utilizados dentro do módulo E020 deve ser acrescentado ao máximo nível empregado em E001 e E018, devendo esta soma ser menor que o limite do sistema (18 níveis).

Utilização dos Operandos na Programação dos Módulos E018 e E020

Outro cuidado necessário diz respeito ao compartilhamento de dados entre os módulos E018 ou E020 e os demais presentes no controlador programável. As interrupções podem ocorrer em qualquer ponto do programa principal de execução cíclica (módulo E001 ou módulos P ou F chamados pelo mesmo), inclusive durante o processamento das suas instruções. Como os operandos são todos de uso comum a qualquer módulo do projeto de programação, deve-se tomar o cuidado para não utilizar inadvertidamente nos módulos E018 ou E020 algum operando que seja utilizado em outro módulo do projeto de programação, pois este uso, conforme o caso, pode ocasionar o funcionamento incorreto. Quando o módulo E018 e E020 são utilizados simultaneamente, ambos devem empregar operandos exclusivos.

Para possibilitar o compartilhamento de dados entre os módulos E018, E020 e outro módulo qualquer de execução cíclica devem ser utilizadas as instruções MOV (movimentação de operandos simples) e MOB (movimentação de blocos de operandos), para criar uma imagem dos operandos que contêm os dados a serem compartilhados. Estas instruções devem ser utilizadas nos módulos pertencentes ao ciclo normal de execução e não nos módulos E018 ou E020.

ATENÇÃO: Por exemplo, se for necessário que o módulo E018 utilize o valor contido em uma memória usada no programa principal, deve-se passar o valor desta memória para outra através da instrução MOV, devendo o módulo E018 utilizar somente esta última. A instrução MOV deve estar no programa principal, e não no módulo E018.

ATENÇÃO:
O fluxo contrário de dados também exige a criação de operandos imagem. Se o módulo E020 manipula uma tabela e o programa principal precisa utilizar os valores desta tabela, esses valores devem ser copiados para uma segunda tabela de uso exclusivo do programa principal, através de uma instrução MOB. A instrução MOB deve estar no programa principal, e não no módulo E020.

ATENÇÃO:
Uma situação semelhante ocorre para as instruções bobinas. Se algum ponto de um operando é modificado no programa principal por uma bobina, não é permitida a alteração de qualquer ponto pertencente a todo octeto do mesmo operando nos módulos E018 ou E020. Esta restrição não existe quando os octetos utilizados pertencem à faixa %S0000 a %S0015.

Entretanto, é possível que pontos de um operando sejam alterados nos módulos E018 ou E020 por uma bobina e sejam somente testados por outro módulo com instruções contatos, por exemplo. A situação contrária também é permitida, ou seja, os pontos de um operando modificados no programa principal por bobinas podem ser testados nos módulos E018 ou E020 por contatos.

Outro cuidado a ser tomado diz respeito à atualização das entradas e saídas dentro dos módulos E018 ou E020.

Preferencialmente devem ser atualizadas dentro destes módulos somente as entradas utilizadas no seu processamento, utilizando-se a instrução AES. Como o programa aplicativo de execução cíclica pode ser interrompido em qualquer local por estes módulos, se neles forem atualizadas as imagens das entradas do programa principal, estas poderão assumir valores diversos em pontos diferentes do programa aplicativo durante o mesmo ciclo de execução. Este fato pode provocar erros se um operando de entrada for utilizado em vários lugares do programa principal, pois normalmente é suposto que seu valor permaneça inalterado na mesma varredura.

Devido a este fato, é aconselhável o uso de octetos de entrada exclusivos para os módulos E018 ou E020, se for necessária a sua atualização dentro do mesmo, não sendo estes octetos utilizados no programa principal.

Caso seja necessária a atualização de entradas utilizadas simultaneamente nas interrupções e no processamento cíclico, o valor das mesmas pode ser copiado para operandos auxiliares ou memórias no início do programa principal, sendo estes operandos utilizados no restante do mesmo. Pode-se também não atualizar as imagens das entradas nos módulos E018 ou E020 com a instrução AES, mas somente ler diretamente os valores dos módulos de E/S para operandos memória através da instrução MES, e utilizar estas memórias em contatos para realizar o processamento nos módulos de interrupção.

A atualização de octetos de saída nos módulos E018 ou E020 (através da instrução AES) é possível, desde que os pontos pertencentes a estes octetos sejam acionados por bobinas somente dentro destes módulos.

Nos módulos E018 e E020, não se deve escrever valores com a instrução MES em módulos de saída declarados no barramento através do MasterTool XE, pois a varredura das saídas também realiza atualização de valores nos mesmos.

Quando um módulo E018 ou E020 está sendo executado e a compactação for disparada, os mesmos poderão ser transferidos para outra posição na memória pela rotina de compactação. Durante esta transferência a sua chamada será desabilitada, podendo algumas interrupções ocorrerem sem que os módulos E018 ou E020 sejam processados. Deve-se atentar para este efeito da compactação sobre a execução do módulo acionado por interrupção. Durante a compactação dos demais módulos, todavia, os módulos E018 ou E020 continuarão sendo executados.

Utilização Simultânea dos Módulos E018 e E020

Não existem prioridades de execução quanto às interrupções dos dois módulos. Ou seja, se estiver sendo executado o módulo E020 e ocorrer a próxima interrupção de tempo, o processamento do E020 é interrompido, é executado o módulo E018, retornando após para a execução interrompida do E020.

Da mesma forma, se estiver sendo executado o módulo E018 e for acionada a entrada de interrupção, o processamento do E018 é interrompido, é executado o módulo E020, retornando após para a execução interrompida do E018.

Alguns cuidados devem ser observados na utilização simultânea dos dois módulos. Deve-se somar o tempo de execução do módulo E018 ao E020, devendo este total ser menor do que o período de chamada programado para o E018, evitando-se o erro de reentrada de execução no mesmo.

Deve-se também somar o número máximo de níveis de chamada dos módulos E001, E018 e E020, devendo o resultado ser menor ou igual ao número máximo permitido (32 níveis).

Os dois módulos devem utilizar operandos exclusivos, obedecendo às regras da seção **Utilização dos Operandos na Programação dos Módulos E018 e E020,** neste mesmo capítulo.

Depuração de Projetos de Programação

Várias facilidades estão previstas no controlador programável para auxiliar a depuração dos projetos de programação, sendo descritas a seguir.

Informações do Estado do CP

Diversas informações sobre o estado do controlador podem ser obtidas com o acionamento das opções **Comunicação, Informações** no MasterTool XE.

- **Modelo da UCP** - indica o tipo do controlador com o qual o MasterTool XE está comunicando.
- **Versão do Executivo** - mostra o número da versão do programa executivo que o CP contém.
- **Modo de Operação** - indica o modo de operação atual do CP: execução, programação, ciclado ou erro.
- **Mensagem de Erro/Advertência** - se o CP estiver em modo erro, é apresentada uma mensagem indicando a causa do erro ocorrido. Caso o CP esteja em qualquer outro modo, a mensagem indica a existência de problemas que não causam a mudança para modo erro (por exemplo, a bateria do CP descarregada). Ver Manual de Utilização do MasterTool XE.
- **Saídas** - indica se as saídas estão habilitadas ou desabilitadas.
- **Relés Forçados** - indica se existe algum ponto de entrada ou saída forçado.
- **Troca de Módulos com CP Energizado** - indica a possibilidade de troca de módulos com CP energizado.
- **Compactando RAM** - indica se o CP está compactando a memória RAM do programa aplicativo.
- **Copiando Módulo** - indica se algum módulo está sendo carregado no CP, transferido da RAM para a FLASH EPROM ou da FLASH EPROM para a RAM, ou se o CP está apagando a memória FLASH.
- **Nível de Proteção** - mostra o nível de proteção atual do CP.
- **Tempos de Ciclo** - mostra os valores instantâneo, médio, máximo e mínimo do tempo de ciclo do programa aplicativo. Ver seção **Tempos de Ciclo de Execução do Programa** neste mesmo capítulo.
- **Período de Acionamento de E018** - mostra o período de chamada do módulo acionado por interrupção de tempo E018, se estiver presente no CP.

As janelas de estado do CP (opções **Comunicação, Estado, Informações**), diretório de módulos (opções **Comunicação, Módulos**) e monitoração (opções **Comunicação, Monitorar Operandos ou**

Monitorar Bloco de Operandos ou Monitorar Tabelas) fornecem diversas informações úteis para a verificação do bom funcionamento do controlador. Estas informações podem ser obtidas à distância, caso o CP esteja ligado em rede. Sempre que o MasterTool XE for conectado a algum CP, aconselha-se a consulta dessas informações como o primeiro procedimento a ser tomado.

Monitoração

Através do MasterTool XE é possível monitorar os valores de um ou mais operandos do CP em qualquer modo de operação, exceto em modo erro.

Os valores dos operandos contidos em uma lógica de programa aplicativo também podem ser visualizados diretamente sobre a mesma facilitando a verificação de seu funcionamento.

Para maiores informações sobre como realizar a monitoração, ver itens **Monitorando Operandos Simples**, **Monitorando Operandos Tabelas** e Monitorando Programas no Manual de Utilização do MasterTool XE.

ATENÇÃO: A monitoração de operandos no CP ocorre no final do ciclo de execução do programa aplicativo. Devido a este fato, situações incoerentes podem ser visualizadas na monitoração de lógicas, se os valores dos operandos forem modificados nas lógicas posteriores à monitorada.

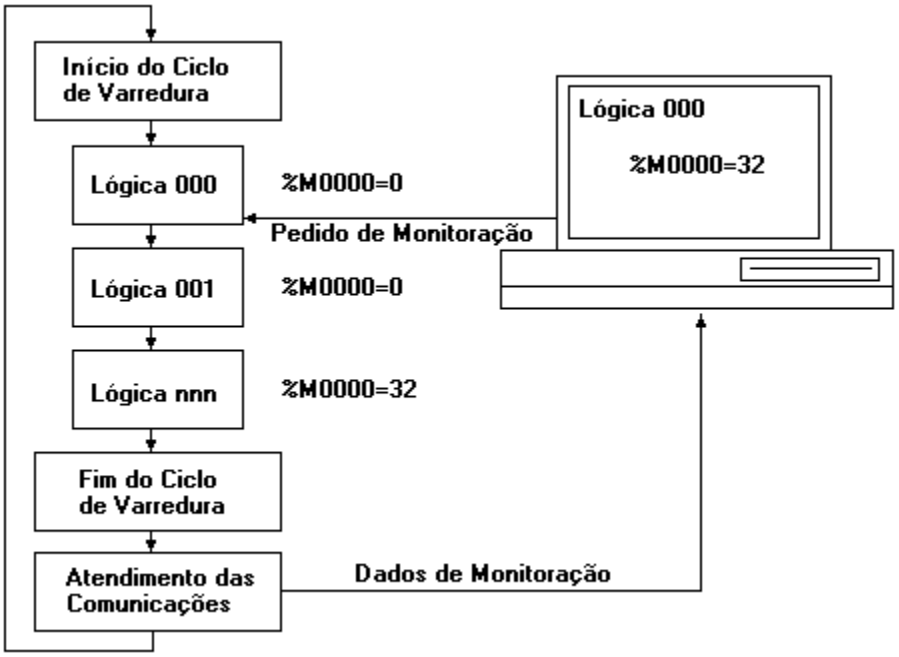


Figura 2-30. Situação incoerente na monitoração de lógicas

Forçamento

Os valores dos operandos também podem ser forçados com o MasterTool XE, ou seja, pode-se modificar o conteúdo de qualquer operando do projeto de programação. O forçamento de operandos é permitido em qualquer modo de operação, exceto em modo erro.

Os operandos %A, %M, %I, %D, %F, %TM, %TI, %TD e %TF têm o seu valor alterado somente por uma varredura, logo após o envio do comando ao CP. Para que o valor forçado permaneça no operando, não pode haver no programa nenhuma instrução que o modifique.

O forçamento dos operandos %E e %S é realizado de forma permanente no controlador. Após o envio do comando ao CP, o valor é forçado em todas as varreduras do programa aplicativo, até que o operando seja liberado. O LED FC no painel do CP permanece ligado se houver algum operando %E ou %S forçado.

Os valores forçados em operandos %E sobrepõem os obtidos na leitura dos módulos de entrada, antes do início de cada ciclo de execução do programa aplicativo. O programa é executado com o valor forçado, como se o ponto de entrada correspondente estivesse com este valor, podendo ser visualizado na monitoração.

Por exemplo, caso o operando %E0002.5 esteja forçado com o valor 1, o programa aplicativo será executado com este valor para este operando, não importando o estado do ponto no módulo de entrada correspondente. A monitoração de %E0002.5 mostra sempre o valor 1.

Os valores forçados nos operandos %S são diretamente enviados para os módulos de saída, independente do valor obtido após a execução do programa aplicativo. A monitoração mostra o valor forçado, que corresponde ao valor assumido pelo ponto correspondente ao operando no módulo de saída.

Por exemplo, caso o operando %S0024.3 esteja forçado com o valor 0, o ponto respectivo no módulo de saída permanecerá desligado, não importando o estado da bobina que contenha o operando no programa aplicativo. A monitoração de %S0024.3 mostra sempre o valor 0.

ATENÇÃO:
Podem ser visualizadas situações incoerentes na monitoração de lógicas com operandos %S forçados. Isto ocorre porque o valor monitorado pode ser diferente do valor realmente obtido pelo programa aplicativo.

ATENÇÃO:
Todos os forçamentos de operandos %E e %S são removidos com a desenergização do CP. O forçamento destes operandos deve ser utilizado de forma temporária, somente para auxiliar a depuração do projeto de programação. Não devem ser deixados operandos %E ou %S forçados em caráter permanente, pois serão liberados com a desenergização e posterior energização do controlador.

Os operandos %E e %S deixam de ser forçados pelo CP através do comando de liberação de forçamento. A liberação consiste em anular o forçamento anteriormente determinado. Os operandos %E voltam a ter seus valores atualizados de acordo com os módulos de entrada, enquanto que os módulos de saída recebem os valores obtidos no processamento do programa aplicativo.

ATENÇÃO:
A operação de forçamento não atua sobre operandos %E ou %S atualizados com a instrução AES. Esta instrução executa a leitura para operandos %E ou a escrita de operandos %S no momento em que é executada, não considerando os efeitos de forçamento sobre os mesmos. Por este motivo, recomenda-se que não sejam forçados os operandos atualizados pela instrução AES que estejam ativas no programa.

Desabilitação das Saídas

Para a segurança na posta-em-marcha quando se utiliza o programa aplicativo diretamente na máquina, os acionamentos das saídas do controlador programável podem ser desabilitados através do comando desabilita. O programa aplicativo continua a ser executado no CP, com a varredura das entradas e cálculo dos valores das saídas, porém com todos os pontos de saída mantidos desacionados. Os operandos %S podem ser monitorados e conferidos com os valores esperados para os mesmos.

ATENÇÃO:
Se o CP for desenergizado, a desabilitação dos pontos de saída é removida. Ou seja, quando o CP for novamente energizado, o estado dos operandos da memória será normalmente transferido, ao final de cada varredura, para os pontos de saída. A desabilitação deve ser utilizada de forma temporária, somente para auxiliar a depuração do projeto de programação .

Modificações no Programa

A carga de módulos durante a execução do projeto de programação (carga "on line") possibilita sucessivas modificações e envios do módulo em depuração para o controlador programável. Deste modo não é necessária a reinicialização do programa aplicativo de controle nem a troca do estado do controlador programável a cada alteração efetuada em um módulo.

ATENÇÃO:
Após qualquer modificação realizada no módulo C do projeto de programação, o mesmo deve ser enviado para o CP, para que as alterações tenham efeito.

ATENÇÃO:
Se a declaração de operandos simples ou tabelas for modificada, aconselha-se reinicializar o CP, passando para modo programação, carregando o módulo C e retornando para modo execução. Podem ocorrer erros no funcionamento alterando-se a configuração de operandos e enviando o módulo C com o controlador em modo execução.

Após um certo número de cargas sucessivas em modo execução, entretanto, pode se tornar necessária a compactação da memória RAM de programa pelos motivos explicados na seção **Gerenciamento de Módulos do Projeto de Programação** , neste capítulo. Este tipo de carga somente é possível se houver memória livre suficiente no CP para armazenamento do módulo a ser enviado.

Ao acabar a depuração de um módulo de programa, sugere-se a transferência do mesmo para a memória FLASH EPROM ou a sua gravação no cartucho de EPROM, liberando o espaço disponível na memória RAM de programa.

Modo Ciclado

A execução do projeto de programação em modo ciclado torna-se útil na verificação do funcionamento de intertravamentos rápidos no programa aplicativo. As demais facilidades de depuração continuam atuando da mesma forma como no modo execução (monitoração, forçamento, carga e outras operações com módulos).

Em modo ciclado, os valores dos operandos permanecem constantes entre os ciclos, exceto os pontos de entrada (%E) que continuam sendo continuamente atualizados, mostrando seus valores reais.

Gerenciamento de Módulos do Projeto de Programação

Os módulos que compõem o programa aplicativo são independentes entre si, não necessitando de ligação ("link") através de programas auxiliares. A carga dos módulos no controlador programável pelo canal serial pode ser realizada em qualquer ordem, permitindo que somente o módulo alterado seja carregado no CP, em caso de manutenção de projetos de programação.

ATENÇÃO:
Para o CP do sistema, somente o tipo do módulo e seu número são relevantes, sendo o seu nome desconsiderado. Se dois módulos com tipo e número iguais mas com nomes diferentes forem carregados no CP, somente o último carregado será considerado.

O controlador programável organiza um diretório interno onde são armazenadas diversas informações a respeito dos módulos contidos no mesmo, podendo ser consultado pelo MasterTool XE através do comando diretório de módulos (opções **Comunicação, Módulos** a partir do menu principal). Quando este comando for acionado, uma caixa de diálogo é aberta, mostrando na sua parte

superior, dois quadros chamados Módulos em RAM e Módulos em EPROM com a lista dos nomes e a ocupação de memória de cada módulo presente no CP.

No quadro Memória Ocupada é informado o número total de módulos e o espaço de memória total ocupado pelos mesmos (soma de todas as ocupações individuais), além do espaço total ocupado em RAM ou EPROM.

No quadro Memória Livre estão expostas as quantidades de memória RAM e EPROM disponível para a carga de novos módulos, em cada banco de memória existente no controlador programável.

ATENÇÃO: Somente os módulos presentes no diretório são considerados válidos para a execução no CP.

ATENÇÃO: Um módulo de programa presente no diretório do CP pode estar somente em um tipo de memória, RAM ou EPROM, nunca simultaneamente em ambas. Os módulos carregados pelo canal serial são sempre armazenados na memória RAM de programa aplicativo.

Compactação

A memória de programa do controlador programável está dividida em um ou mais bancos, dependendo do modelo de CP utilizado.

À medida que os módulos que compõem o projeto de programação são enviados para o CP através do canal serial, os mesmos ocupam o primeiro banco de memória RAM, desde o seu início até o seu final. Quando o espaço restante no primeiro banco não for suficiente para carregar o próximo módulo, este será carregado no banco seguinte, se este existir.

A cada carga de novo módulo no controlador programável, o software executivo testa se há espaço suficiente para o mesmo desde o primeiro até o último banco disponível. A carga de um novo módulo somente é possível se houver memória livre à disposição para o seu armazenamento.

Dentro de um banco de memória RAM, a carga de um módulo é realizada sempre a partir da primeira posição após o último módulo presente. Se um módulo no início do banco for removido, os módulos que estão após o mesmo devem ser transferidos para ocupar o seu espaço de memória, para que este espaço esteja disponível no final do banco para a carga de outros módulos. Este procedimento denomina-se compactação da memória RAM do programa aplicativo.

Exemplo:

Suponha-se que o primeiro banco de memória do controlador programável esteja inicialmente com os seguintes módulos:

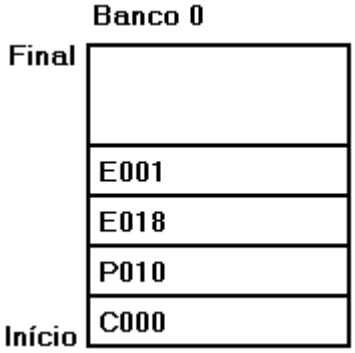


Figura 2-31. Compactação de memória RAM

Se o módulo P010 for removido do CP o banco 0 passará a ter a seguinte organização:

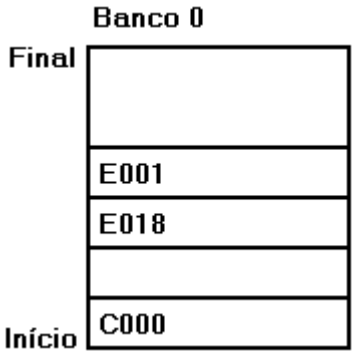


Figura 2-32. Compactação da memória RAM - 2

O espaço anteriormente ocupado por P010 não é aproveitável pelo controlador programável, pois a carga de um novo módulo somente é possível após o último, o módulo E001. Após realizar a compactação de memória do CP, o banco 0 passa para a seguinte configuração:

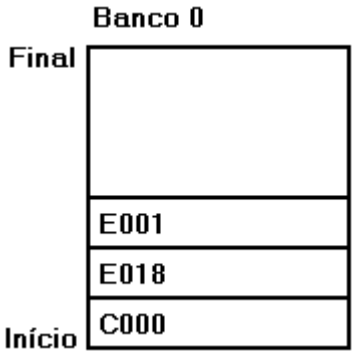


Figura 2-33. Compactação da memória RAM - 3

Os módulos E018 e E001 são transferidos para o espaço anteriormente ocupado pelo módulo P010, tornando este espaço disponível ao final da memória do banco para a carga de outro módulo.

Se o controlador programável estiver em modo programação ou ciclado, os bancos de memória RAM de programa são mantidos automaticamente compactados pelo programa executivo. Em modo execução, todavia, deve-se disparar a compactação manualmente, através do MasterTool XE (opções **Comunicação**, **Módulos**, **Compactar RAM** desde o menu principal). Este procedimento é comum quando são realizadas diversas cargas de módulos em modo execução (cargas "on line"), tipicamente quando um módulo está sendo depurado, necessitando de sucessivas alterações e transmissões para o CP.

ATENÇÃO: Dependendo da localização dos módulos na memória, o procedimento de compactação pode aumentar em muito o tempo de alguns ciclos do programa aplicativo, quando realizado em modo execução. Deve-se estar consciente dos efeitos deste aumento de tempo de processamento. Aconselha-se que a compactação não seja disparada caso a máquina sob controle esteja em operação ou com seus acionamentos principais ativos.

Devido a este mecanismo de gerenciamento de módulos no controlador programável, é possível que a soma da memória disponível nos bancos do CP com o valor ocupado pelos módulos seja menor que a memória de programa total do CP, se esta estiver em modo execução. Este fato significa que a memória de programa não está compactada. Após a compactação, entretanto, a soma dos valores ocupados com a memória livre deve ser igual à memória total.

ATENÇÃO:
No MasterTool XE não existe a Compactação de FLASH. O método para se "compactar" a FLASH é carregar os módulos para a RAM, limpar a FLASH e recarregar os módulos para a FLASH.

Transferência de módulos de RAM para FLASH:

Após serem carregados na memória RAM de programa, através do canal serial do CP, os módulos do projeto de programação podem ser transferidos para a FLASH EPROM. Este comando somente é utilizável nos CPs que possuam memória FLASH.

Pode-se transferir um único módulo ou um conjunto de módulos, mesmo com o CP executando o programa. A transferência em modo execução é realizada parcialmente em cada varredura, podendo demorar vários segundos até ser completada, principalmente se houver alto tempo de ciclo de execução. No final da transferência, o módulo em RAM é automaticamente apagado e as informações do diretório modificadas.

O gerenciamento da carga do módulo na FLASH EPROM é idêntico ao da memória RAM, mostrada na seção anterior **Compactação**. Ou seja, o módulo da RAM é gravado no primeiro banco de FLASH que possua espaço livre suficiente para o conter, após o último módulo do banco. A pesquisa do espaço livre ocorre na ordem seqüencial dos bancos (0, 1, 2 e 3).

Transferência de módulos de FLASH para RAM:

Os módulos presentes na memória FLASH EPROM ou no cartucho de EPROM também podem ser transferidos para a memória RAM de programa.

Pode-se transferir um único módulo ou um conjunto de módulos, mesmo com o CP executando o programa. A transferência em modo execução é realizada parcialmente em cada varredura, podendo demorar vários segundos até ser completada, principalmente se houver alto tempo de ciclo de execução. No final da transferência, as informações do diretório são modificadas.

O gerenciamento da carga do módulo na FLASH EPROM é idêntico ao da memória RAM, mostrada na seção anterior **Compactação**.

Apagamento de módulos em FLASH:

O comando de apagamento pode ser utilizado para módulos armazenados na memória FLASH do CP. Como o apagamento de FLASHs somente é possível para todo o seu conteúdo, este comando apenas retira as informações do diretório de módulos, não realizando um apagamento real da memória.

O mesmo ocorre se um módulo gravado em FLASH for substituído por um novo módulo de mesmo tipo e número carregado pelo canal serial. O novo módulo é armazenado em RAM, permanecendo o antigo em EPROM, sendo mostrado no diretório apenas o novo em RAM.

Apagamento da memória FLASH:

Com o apagamento total da memória EPROM, todos os módulos são removidos, ficando todo o seu espaço disponível para a gravação de novos módulos.

Para apagar o cartucho de EPROM deve ser utilizado um dispositivo apagador apropriado, após a remoção do cartucho do CP.

Para apagar a memória FLASH EPROM, utiliza-se as opções **Comunicação**, **Módulos**, **Apaga FLASH** estando o CP em modo programação. O apagamento pode demorar vários segundos, dependendo da capacidade da memória FLASH utilizada no CP.

Tempos de Ciclo de Execução do Programa

O tempo máximo possível para a execução de um ciclo completo do programa aplicativo no controlador programável é configurável de 100 ms a 800 ms. Ou seja, a execução completa de uma varredura do módulo E001 não pode se estender por mais do que o valor configurado, incluindo as chamadas para módulos P e F e acionamentos do módulo de interrupção de tempo E018. O software executivo realiza uma verificação contínua no tempo de ciclo, passando automaticamente para estado de erro caso este limite seja ultrapassado.

Pode-se verificar os tempos de execução do programa aplicativo através da janela de informações do CP (opções **Comunicação**, **Informações** a partir do menu principal), sendo informados diversos tempos de ciclos de execução, especificados a seguir:

- **Tempo de ciclo instantâneo:** indica o tempo de ciclo da última varredura executada pelo CP antes de enviar as informações do seu estado para o MasterTool XE. Este item é útil em modo ciclado, quando mostra o tempo de execução do último ciclo disparado no controlador programável.
- **Tempo de ciclo médio:** indica a média de tempos de execução das últimas 256 varreduras realizadas pelo CP. Em modo execução este parâmetro fornece uma idéia geral do tempo de processamento do programa aplicativo, ao contrário do tempo de ciclo instantâneo, que pode estar mostrando um valor atípico de uma varredura isoladamente. Como este tempo é calculado somente a cada 256 varreduras, por vezes o seu valor necessita de alguns segundos para ser atualizado, principalmente em caso de aumento brusco no tempo de execução (inclusão de novos módulos no controlador programável, por exemplo).
- **Tempo de ciclo máximo:** indica o maior tempo entre todos os ciclos realizados desde a passagem do CP para modo execução ou ciclado.
- **Tempo de ciclo mínimo:** indica o menor tempo entre todos os ciclos realizados desde a passagem do CP para modo execução ou ciclado.

Os tempos de ciclo são indicados em milisegundos (ms), sendo as contagens inicializadas na passagem de modo programação para execução ou programação para ciclado.

O atendimento da comunicação serial com o MasterTool XE aumenta o tempo de ciclo do programa aplicativo no CP, podendo, em alguns casos, ultrapassar o tempo de ciclo máximo selecionado. Caso o tempo de execução limite for ultrapassado somente devido aos comandos da comunicação serial (monitoração, forçamento e demais), o CP não entrará em estado de erro. É possível, portanto, a indicação de um tempo de ciclo máximo maior que o selecionado sem que o controlador programável tenha entrado em modo erro.

O procedimento de compactação da memória de programa do controlador programável também segue a regra anterior. Em alguns casos, a rotina de compactação necessita copiar um módulo muito extenso na memória entre dois ciclos do programa aplicativo, aumentando exageradamente o tempo de execução de uma varredura. Nesta situação o CP não entrará em estado de erro.

Deve-se tomar cuidados especiais quando os tempos de ciclo de execução se aproximam do tempo máximo selecionado. O simples fato de o programa aplicativo estar executando corretamente com as condições mais comuns dos pontos de entrada não garante que seu tempo de varredura, em condições reais de funcionamento da máquina, permaneça dentro do valor limite.

ATENÇÃO:
Cada projeto de programação deve ser examinado cuidadosamente em busca de situações que irão provocar os maiores tempos de execução. Essas situações devem ser simuladas e os tempos medidos, verificando se não são excessivos. Este procedimento deve ser realizado mesmo em projeto de programação com tempos de ciclo bem abaixo do limite, para assegurar o seu bom funcionamento.

É possível que em algumas varreduras isoladas o tempo de ciclo exceda o tempo máximo selecionado sem que o CP passe para modo erro, caso estas varreduras esporádicas não causem atrasos nos temporizadores do sistema.

ATENÇÃO:
Se o CP indicar o tempo de ciclo máximo maior que o selecionado sem que tenha havido uma compactação de memória, mesmo que continue normalmente em modo execução, deve-se examinar o programa para diminuir o seu tempo de ciclo nas situações que causem maiores tempos.

ATENÇÃO:
Existem alguns procedimentos típicos para diminuir o tempo de execução de programas aplicativos muito extensos. Um bom gerenciamento na chamada de módulos pode diminuir sensivelmente o tempo de ciclo total, sendo realizadas chamadas de poucos módulos do programa aplicativo em cada varredura, não permitindo que todos sejam disparados em um mesmo ciclo. O uso de instruções de salto dentro dos módulos, diminui o tempo de execução dos mesmos, pois um trecho de programa aplicativo saltado é desconsiderado pelo software executivo. As instruções relé mestre e fim de relé mestre (RM e FRM) não possuem esta propriedade, pois o segmento de programa aplicativo delimitado pelas mesmas continua a ser executado mesmo quando a bobina RM o desabilita.

ATENÇÃO:
Deve-se realizar inicializações de valores em operandos ou tabelas dentro do módulo E000, idealizado especialmente para este propósito. A execução do módulo E000, por não ser cíclica, pode demorar mais que o tempo máximo, sendo este tempo desconsiderado na contagem do tempo da primeira varredura do módulo E001. Pelo modo como é executado, torna-se sem sentido a programação de temporizadores (TEE, TED) no módulo E000.

Níveis de Proteção do CP

Os CPs da série AL-2000 possuem um mecanismo de proteção do projeto de programação e dos operandos, permitindo o bloqueio da carga de módulos de programa, forçamentos de valores ou mesmo leituras de módulos e monitoração por operadores não autorizados.

Estas características são interessantes em processos críticos, para evitar modificações acidentais nos dados ou no programa de controle ou na necessidade de sigilo dos mesmos.

O bloqueio de operações é realizado através de níveis de proteção, que podem ser definidos apenas por operadores que conheçam uma senha preestabelecida. O controlador pode operar em quatro diferentes níveis de proteção:

- **Nível 0** - Sem proteção
- **Nível 1** - Permitido monitoração, escrita, forçamento de operandos e leitura de módulos de programa
- **Nível 2** - Permitido monitoração, escrita, forçamento de operandos
- **Nível 3** - Permitido apenas leitura das informações do CP

A troca do nível de proteção é realizada com as opções **Comunicação**, **Estado**, **Proteção** no MasterTool XE, devendo-se digitar a senha de acesso correta para efetivá-la. O nível de proteção do CP pode ser consultado com o MasterTool XE através das opções **Comunicação, Estado, Informações**.

A utilização de níveis de proteção diferentes de zero permite que somente pessoas autorizadas, que conheçam a senha, modifiquem o programa ou os dados do CP. Operadores não autorizados, mesmo dispondo do MasterTool XE, ficam impedidos de realizar alterações inadvertidas.

A senha de acesso pode ter de um a oito caracteres alfanuméricos. É definida ou trocada com as opções **Comunicação**, **Estado**, **Senha**, devendo-se digitar a senha anterior e a nova senha duas vezes, para ser confirmada a mudança.

O CP é fornecido sem senha. Não é necessário digitar qualquer valor no campo senha anterior para a definição da primeira senha.

ATENÇÃO:
A senha deve ser escrita e guardada em lugar seguro. Em caso de perda da senha programada no CP, entrar em contato com a ALTUS.

A proteção do CP atua não somente para as operações realizadas com o MasterTool XE, mas também para os comandos recebidos pelas redes ALNET I e ALNET II, com as mesmas características definidas para cada nível.

Intertravamento de Comandos no CP

Na série AL-2000 pode-se utilizar as redes de comunicação ALNET I e ALNET II em conjunto. Quando interligado desta forma, é possível a recepção simultânea de dois comandos cuja execução concorrente seja indesejável, devido às suas características. Por exemplo, o CP pode receber um comando de transferência de módulo da EPROM para a RAM pela ALNET II enquanto o mesmo módulo está sendo carregado na ALNET I.

Situações semelhantes ocorrem com os comandos de transferência de módulos de programa da memória EPROM para RAM, de RAM para FLASH ou de apagamento da memória FLASH. A execução destes comandos pode estender-se por vários segundos, durante os quais o CP pode receber outros comandos que entrem em conflito com a operação em curso. Por exemplo, o CP pode receber um comando para apagar a memória FLASH enquanto um módulo estava sendo transferido para a mesma memória.

Para resolver as possíveis situações de conflito, há um mecanismo de intertravamento para a execução de alguns comandos disponíveis no CP. Estes comandos não podem ser executados caso o CP esteja realizando uma operação específica. Existem dois sinais internos, carregando módulo (CM) e compactando RAM (CR), que são usados para este propósito. As tabelas abaixo mostram os comandos que utilizam o intertravamento e o acionamento dos sinais.

O estado dos sinais carregando módulo e compactando RAM pode ser verificado na janela de informações do CP, opções **Comunicação, Informações** no MasterTool XE. Enquanto qualquer um dos sinais estiver acionado, o LED FC do painel do CP permanece aceso.

Operação realizada no CP	Comando bloqueado (ALNET I, ALNETII)	Sinal Ligado
Carga de Módulos	Carga de módulos Transferência de EPROM para RAM Transferência de RAM para FLASH Pedido de carga de módulos Reabilitação de módulos em EPROM Apagamento de FLASH EPROM Compactação	CM
Transferência de EPROM para RAM	Carga de módulos Transferência de EPROM para RAM Transferência de RAM para FLASH Pedido de carga de módulos Reabilitação de módulos em EPROM Apagamento de FLASH EPROM Compactação	CM
Transferência de RAM para FLASH	Carga de módulos Transferência de EPROM para RAM Transferência de RAM para FLASH Pedido de carga de módulos Reabilitação de módulos em EPROM Apagamento de FLASH EPROM Compactação	CM
Apagamento de FLASH EPROM	Carga de módulos Transferência de EPROM para RAM Transferência de RAM para FLASH Pedido de carga de módulos Reabilitação de módulos em EPROM Apagamento de FLASH EPROM Compactação	CM
Legenda: CM - Carregando Módulo		

Tabela 2-3. Intertravamento de comandos no CP (carregando módulo)

Operação realizada no CP	Comando bloqueado (ALNET I, ALNETII)	Sinal Ligado
Carga de Módulos	Carga de módulos Transferência de EPROM para RAM Transferência de RAM para FLASH Pedido de carga de módulos Reabilitação de módulos em EPROM Remoção de módulos Compactação	CR
Legenda: CR - Compactando RAM		

Tabela 2-4. Intertravamento de comandos no CP (Compactando RAM)

Por exemplo, enquanto um módulo está sendo carregado no CP pela rede ALNET I ou ALNET II, os comandos de carga de módulos, transferência de EPROM para RAM, transferência de RAM para FLASH, pedido de carga de módulos, reabilitação de módulos em EPROM, apagamento de FLASH EPROM e compactação não podem ser executados, caso sejam recebidos através da outra rede. Se forem recebidos pelo CP, uma resposta indicando a impossibilidade de sua execução é transmitida para o solicitante.

3. Referência das Instruções

Este capítulo apresenta a lista de instruções integrantes da Linguagem de Diagramas e Relés ALTUS, descrevendo o formato, o uso, a sintaxe e fornecendo exemplos de cada instrução.

Lista das Instruções

Os CPs ALTUS utilizam a linguagem de relés e blocos para a elaboração do programa aplicativo, cuja principal vantagem, além de sua representação gráfica, é ser similar a diagramas de relés convencionais.

A programação desta linguagem, realizada através do MasterTool XE, utiliza um conjunto de poderosas instruções apresentadas nas seções seguintes.

As instruções do MasterTool XE podem ser divididas em 7 grupos:

- **RELÉS**
- **MOVIMENTADORES**
- **ARITMÉTICOS**
- **CONTADORES**
- **CONVERSÕES**
- **GERAIS**
- **LIGAÇÕES**

Convenções Utilizadas

Foram utilizadas diversas convenções para a apresentação dos grupos e instruções tornando melhor a visualização e reconhecimento dos itens descritos, visando com isto um aprendizado mais simples e uma fonte de consulta direta aos tópicos desejados.

Apresentação dos Grupos

A descrição de cada grupo segue o seguinte roteiro.

1. O grupo é descrito com um título contendo o nome do grupo.
2. Logo após o título, é realizada uma breve descrição das características comuns às instruções do grupo.
3. Finalizando a apresentação do grupo, é exibida uma tabela contendo na primeira coluna o nome da instrução, na segunda coluna a descrição do nome da instrução e na terceira coluna a seqüência de teclas para realizar a inserção da instrução diretamente pelo teclado.

Instruções do Grupo Relés

As instruções do grupo Relés são utilizadas para o processamento lógico dos diagramas de relés. Através das mesmas pode-se manipular os valores dos pontos digitais de entrada (%E) e saída (%S), bem como pontos de operandos auxiliares (%A), memória (%M) e decimal (%D).

São usadas também para desvio do fluxo e controle do processamento do programa aplicativo.

Nome	Descrição do Nome	Seqüência de Edição
RNA	Contato normalmente aberto	Alt, I, R, A
RNF	Contato normalmente fechado	Alt, I, R, F
BOB	Bobina simples	Alt, I, R, B
SLT	Bobina de salto	Alt, I, R, S
BBL	Bobina liga	Alt, I, R, L
BBD	Bobina desliga	Alt, I, R, D
PLS	Relé de pulso	Alt, I, R, P
FRM	Fim de relé mestre	Alt, I, R, M
RM	Relé mestre	Alt, I, R, R

Tabela 3-1. Instruções do grupo relés

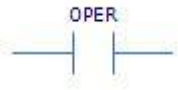
Apresentação das Instruções

A descrição de cada instrução é feita da seguinte maneira:

1. A instrução é descrita com um título contendo o nome da instrução e a descrição do nome. Uma figura apresentando como a instrução é visualizada no diagrama de relés contendo seus operandos, entradas e saídas. Abaixo da figura, é exibida uma breve descrição do significado de cada operando.
2. O item **Descrição** contém informações descrevendo o funcionamento da instrução conforme as entradas habilitadas e os tipos de operando utilizados. Neste item também são descritas as saídas que serão acionadas após a execução da instrução.
3. O item **Sintaxe** descreve as combinações de operandos que podem ser utilizados na instrução. Este item somente esta presente nas instruções que possuam operandos.
4. O item **Exemplo** fornece um exemplo de utilização da instrução descrevendo seu comportamento. Este item somente esta presente nas instruções que requeiram um detalhamento maior de seu funcionamento.
5. Podem existir outros itens descrevendo uma característica específica da instrução caso haja necessidade.

Contatos

RNA - Contato Normalmente Aberto



RNF - Contato Normalmente Fechado

Descrição:

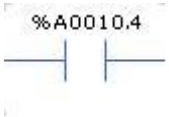
Estas instruções refletem, logicamente, o comportamento real de um contato elétrico de um relé no programa aplicativo.

O contato normalmente aberto fecha conforme o estado do seu operando associado. Caso o ponto do operando esteja no estado lógico **1** ou **0**, o contato normalmente aberto está fechado ou aberto, respectivamente.

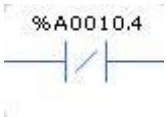
O contato normalmente fechado possui comportamento oposto ao normalmente aberto. Caso o ponto do operando associado esteja no estado lógico **1** ou **0**, o contato normalmente fechado está aberto ou fechado, respectivamente.

Quando um contato está fechado, a instrução transmite o estado lógico da sua entrada para a sua saída. Se estiver aberto, o valor da entrada não é colocado na saída.

Exemplo:



No caso acima, de um contato normalmente aberto, o contato somente será fechado caso o estado do operando %A0010.4 for igual a **1**. Caso contrário ele permanecerá aberto.



Já neste caso, onde o contato é normalmente fechado, o contato estará aberto caso o estado do operando %A0010.4 for igual a **1**. Do contrário ele permanecerá fechado.

Sintaxe:

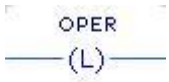
OPER1
%EXXXX.X
%SXXXX.X
%AXXXX.X
%MXXXX.X
%DXXXX.X
%DXXXXhX

Tabela 3-2. Sintaxe das instruções RNA e RNF

Bobinas

BOB - Bobina Simples

BBL - Bobina Liga



BBD - Bobina Desliga

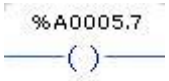
Descrição:

As instruções bobina modificam o estado lógico do operando na memória imagem do controlador programável, conforme o estado da linha de acionamento das mesmas.

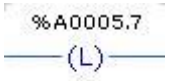
A bobina simples liga ou desliga o ponto do operando conforme a linha de acionamento, enquanto que as bobinas do tipo liga e do tipo desliga somente ligam ou desligam os operandos quando a linha está energizada ("set"/"reset").

Estas instruções somente podem ser posicionadas na coluna 7 da lógica.

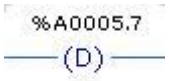
Exemplo:



No caso acima, enquanto a bobina estiver energizada o operando %A0005.7 terá valor lógico **1**, caso contrário o seu valor será **0**.



Neste caso, no momento em que a bobina for energizada, o operando %A0005.7 terá valor lógico **1.** Ele permanecerá com este estado mesmo que a bobina liga esteja desenergizada.



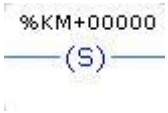
Nesta bobina, no momento em que a bobina for energizada, o operando %A0005.7 terá valor lógico **0.** Ele permanecerá com este estado mesmo que a bobina desliga esteja desenergizada.

Sintaxe:

OPER1
%SXXXX.X
%AXXXX.X
%MXXXX.X
%DXXXX.X
%DXXXXhX

Tabela 3-3. Sintaxe das instruções BOB, BBL e BBD

SLT - Bobina de Salto



Descrição:

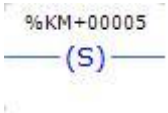
A instrução bobina de salto serve para controlar a seqüência de execução de um programa aplicativo, sendo usada para desviar o processamento do mesmo para uma lógica determinada.

Seu operando é uma constante que determina o número de lógicas a serem saltadas a partir da energização da bobina. A determinação da lógica destino é realizada pela soma da constante que acompanha a instrução com o número da lógica onde a mesma se encontra.

Quando a linha de acionamento da bobina de salto estiver desenergizada, o salto não ocorre, e a instrução seguinte àquela em que esta bobina está declarada é executada.

Exemplo:

Supondo que a instrução a seguir esteja na lógica 1, a execução do programa aplicativo é desviada para a lógica 6 se a linha de acionamento estiver energizada. Se a bobina não for energizada, o processamento continua normalmente, ou seja, a próxima lógica a ser executada seria a 2.



Pode ser utilizada nesta instrução uma constante %KM com valor zero ou mesmo com valor negativo. Se programada com o valor zero, a lógica destino é a mesma que contém a bobina de salto, quando esta é energizada. Ou seja, o processamento é desviado para o início da própria lógica da bobina. Se o valor programado é negativo, o processamento é desviado para uma lógica anterior à lógica que contém a bobina de salto.

ATENÇÃO: O uso de constante zero ou negativa corresponde a um uso não convencional da instrução. Caso deseje-se utilizá-la, deve-se tomar os cuidados necessários para evitar a entrada em laço infinito de execução "loop" ou o aumento excessivo do tempo de ciclo do programa aplicativo. Recomenda-se, contudo, a utilização da bobina de salto somente com constantes positivas maiores que zero.

O controle da execução nestas situações deve ser realizado através de um intertravamento que desligue o salto para a lógica anterior, após um certo número de laços executados no trecho de retorno.

ATENÇÃO: Caso a lógica destino ultrapasse a última lógica do programa aplicativo, o CP salta para o final do programa e continua seu ciclo normal.

ATENÇÃO: Caso a lógica destino de um salto de retorno seja menor do que a primeira lógica do programa aplicativo, a execução é reiniciada a partir da lógica 0.

ATENÇÃO: Caso se utilize um salto com operando negativo após uma instrução RLM, a barra estará desligada para as instruções executadas.

Sintaxe:

OPER1
%KM +XXXXX %KM-XXXXX

Tabela 3-4. Sintaxe da instrução SLT

PLS - Relé de Pulso

Descrição:

A instrução relé de pulso gera um pulso de uma varredura em sua saída, ou seja, permanece energizada durante uma varredura do programa aplicativo quando o estado da sua entrada passar de desenergizado para energizado.

O relé auxiliar declarado serve como memorizador, evitando limitações quanto ao número de instruções de pulso presentes no programa aplicativo.

ATENÇÃO: O valor do relé auxiliar não deve ser modificado em nenhum outro ponto do programa aplicativo.

Exemplo:

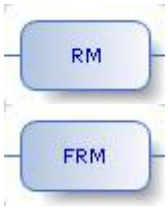
Neste caso quando a entrada deste relé é acionada, a saída permanece energizada durante toda a varredura do programa aplicativo. No operando %A0000.0 é armazenado o estado deste relé.

Sintaxe:

OPER1
%AXXXX.X

Tabela 3-5. Sintaxe da instrução PLS

RM, FRM - Relé Mestre, Fim de Relé Mestre



Descrição:

As instruções relé mestre e fim de relé mestre são utilizadas para delimitar trechos de programas aplicativos, energizando ou não a barra lógica de alimentação nos mesmos, conforme o estado da sua linha de acionamento.

Estas instruções não necessitam de operandos, podendo ser posicionadas somente na coluna 7 da lógica.

Quando a entrada da instrução RM estiver desenergizada, a barra lógica de alimentação é desenergizada desde a lógica seguinte até a lógica que contém a instrução FRM.

Como estas instruções atuam sempre na lógica seguinte a que estão contidas é aconselhável o seu posicionamento sempre como últimas instruções da lógica em que estiverem presentes. Assim sendo, o trecho de programa aplicativo delimitado visualmente pelas instruções no diagrama corresponde exatamente ao controlado pelas mesmas, evitando assim má interpretação de seu funcionamento.

A instrução FRM atua sempre, mesmo que sua entrada esteja desenergizada.

ATENÇÃO: As instruções CON, COB, TEE e TED contém saídas energizadas mesmo sem o acionamento das suas entradas. Estas saídas permanecem energizadas mesmo dentro de um trecho sob comando de um relé mestre desenergizado, podendo causar acionamentos indesejáveis.

Exemplo:

Para o exemplo abaixo, quando o contato normalmente aberto , associado ao operando %A0000.0 estiver em 0 (contato aberto) a entrada da instrução RM estará desenergizada desabilitando a barra de alimentação das lógicas seguintes (Lógica:001) até a que contiver a instrução FRM (Lógica:002).

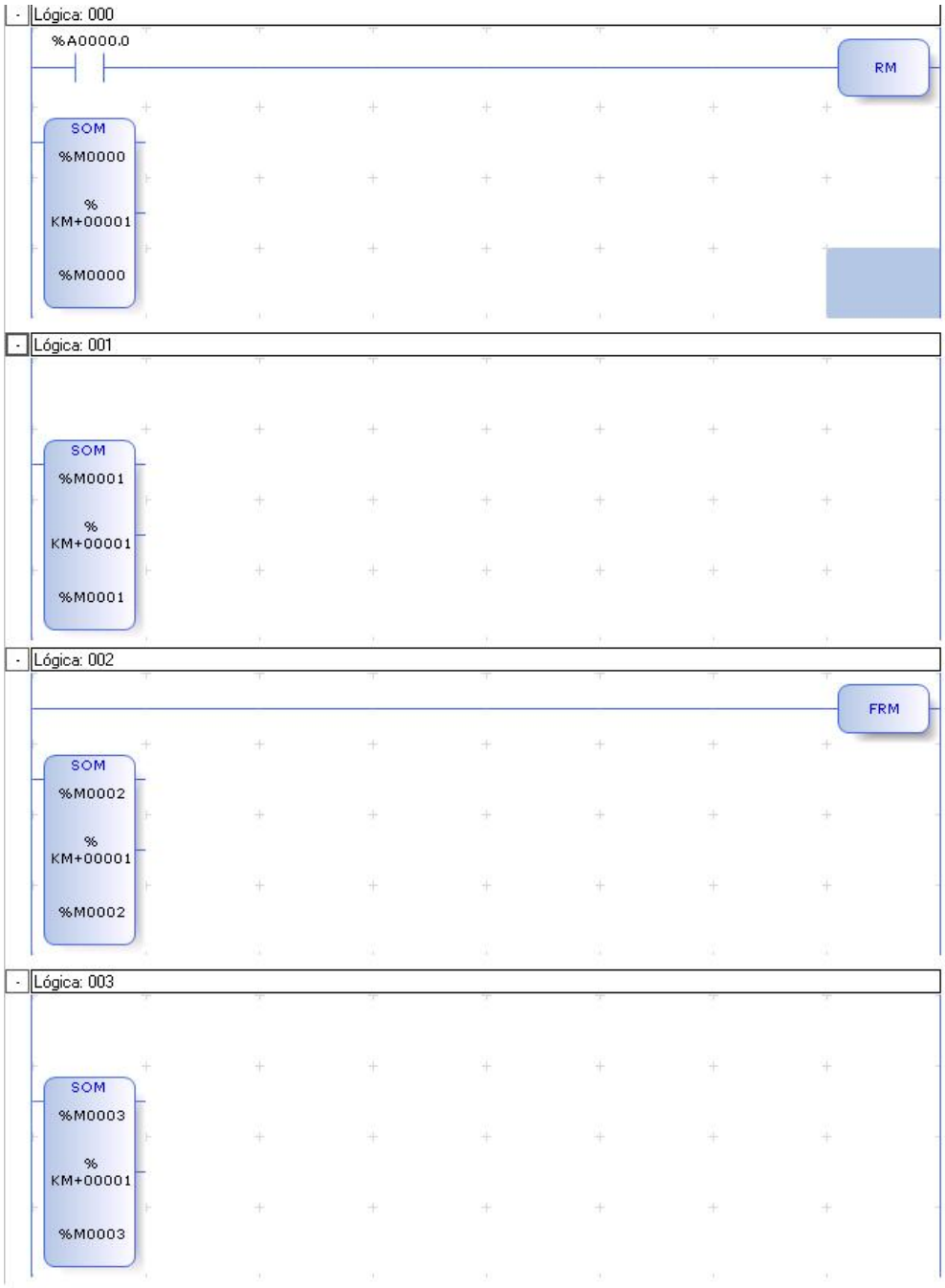


Figura 3-1. Exemplo de utilização da instrução FRM

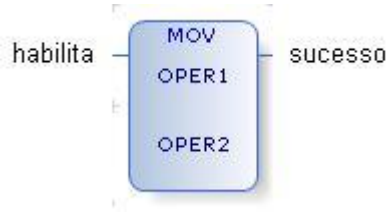
Instruções do Grupo Movimentadores

Estas instruções são utilizadas para a manipulação e transferência de valores numéricos entre constantes, operandos simples ou tabelas de operandos.

Nome	Descrição do Nome	Seqüência de Edição
MOV	Movimentação de operandos simples	Alt, I, M, V
MOP	Movimentação de partes de operandos	Alt, I, M, P
MOB	Movimentação de blocos de operandos	Alt, I, M, B
MOT	Movimentação de tabelas de operandos	Alt, I, M, T
MES	Movimentação de entradas ou saídas	Alt, I, M, E
AES	Atualização de entradas ou saídas	Alt, I, M, A
CES	Conversão de entradas ou saídas	Alt, I, M, S
CAB	Carrega bloco	Alt, I, M, C

Tabela 3-6. Instruções do grupo movimentadores

MOV - Movimentação de Operandos Simples



OPER1 - operando origem
OPER2 - operando destino

Descrição:

Esta instrução move o conteúdo de operandos simples, quando a entrada **habilita** é acionada.

O operando que ocupa a primeira célula da instrução (OPER1) é o operando origem, cujo valor é movimentado para o operando destino, especificado na segunda célula (OPER2).

Se o formato do operando destino for menor que o de origem, os octetos mais significativos do valor origem são desprezados. Se o formato do destino for maior, seus octetos mais significativos são zerados. Se a movimentação for realizada, a saída **sucesso** é acionada.

Se os índices indiretos excederem os limites de operandos declarados no módulo de configuração, a movimentação não é efetuada e a saída **sucesso** não é ligada.

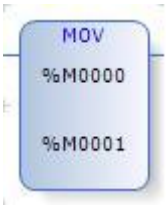
Não é permitida a movimentação de subdivisões de operandos. Para isto, deve ser usada a instrução MOP.

Quando o operando destino da instrução é um inteiro (%M) e pelo menos um dos demais operandos da instrução é um real (%F) o resultado armazenado será truncado, ou seja, armazena-se no operando M apenas a parte inteira do resultado da operação, desprezando-se a parte fracionária.

ATENÇÃO: Se OPER1 for de maior precisão que OPER2 então é preservada a precisão de OPER2.

Exemplo:

Na instrução abaixo, o valor do operando %M0000 é movimentado para o operando %M0001 assim que a entrada habilita for energizada. Ou seja, suponhamos que inicialmente o operando %M0000 possua o valor 5 e %M0001 possua o valor 3. Após a energização da entrada habilita, os valores de %M0000 e %M0001 serão 5 e 5 respectivamente.



Sintaxe:

OPER1	OPER2
%E	
%S	%M
%A	%E
%M	%S
%D	%A
%M*E	%D
%M*S	%M*E
%M*A	%M*A
%M*M	%M*M
%M*D	%M*D
%KM	%M*S
%KD	

OPER1	OPER2
%M	
%F	%M
%I	%F
%M*M	%I
%M*F	%M*M
%M*I	%M*F
%KM	%M*I
%KF	
%KI	

Tabela 3-7. Sintaxe da instrução MOV

MOP - Movimentação de Partes (Subdivisões) de Operandos

OPER1 - operando origem
OPER2 - operando destino

Descrição:

Esta instrução move conteúdos de partes de operandos simples (palavras, octetos, "nibbles", pontos) quando a entrada habilita é energizada. Não é realizada a conversão entre tipos de operandos, apenas a movimentação dos valores.

O operando que ocupa a primeira célula da instrução (OPER1) é o operando origem, cujo valor é movimentado para o operando destino especificado na segunda célula (OPER2). O tipo de subdivisão usado no primeiro operando deve ser o mesmo do segundo.

ATENÇÃO: Se a movimentação é realizada de uma constante para um operando, é considerada sempre a subdivisão menos significativa da constante igual à declarada no operando destino. Devido a esta característica, sugere-se que sempre seja declarado na constante origem o valor real a ser movimentado, para maior clareza do programa.

Exemplo:

O operando destino da instrução está declarado com subdivisão de nibble. Portanto, o nibble menos significativo da constante origem (com valor igual a 1101 em binário, 13 em decimal) será movido para o nibble 2 da memória M0061.

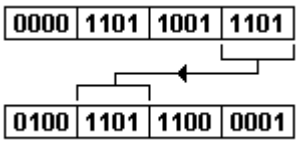


Figura 3-2. Exemplo da instrução MOP

Os demais bits que compõem a constante são desprezados, ou seja, o resultado da movimentação seria idêntico utilizando-se uma constante %KM00013. O exemplo apresentado utiliza um valor excedente ao da movimentação para melhor ilustrar o funcionamento da MOP. Para melhor interpretação do programa deve-se utilizar o valor %KM00013.

Sintaxe:

OPER1	OPER2
%EXXXX.X	
%SXXXX.X	%EXXXX.X
%AXXXX.X	%SXXXX.X
%MXXXX.X	%AXXXX.X
%DXXXX.X	%MXXXX.X
%DXXXXhX	%DXXXX.X
%FXXXX.X	%DXXXXhX
%FXXXXhX	%FXXXX.X
%IXXXX.X	%FXXXXhX
%IXXXXhX	%IXXXX.X
%KMXXXXX	%IXXXXhX
%KDXXXXX	

OPER1	OPER2
%MXXXXbX	
%DXXXXbX	
%FXXXXbX	
%IXXXXbX	%MXXXXbX
%EXXXX	%DXXXXbX
%SXXXX	%FXXXXbX
%AXXXX	%IXXXXbX
%KMXXXXX	
%KDXXXXX	

OPER1	OPER2
%EXXXXnX	
%SXXXXnX	%EXXXXnX
%AXXXXnX	%SXXXXnX
%MXXXXnX	%AXXXXnX
%DXXXXnX	%MXXXXnX
%FXXXXnX	%DXXXXnX
%IXXXXnX	%FXXXXnX
%KMXXXXX	%IXXXXnX
%KDXXXXX	

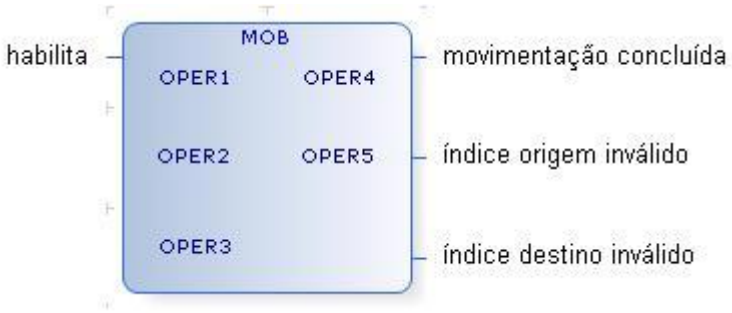
OPER1	OPER2
%MXXXXbX	
%DXXXXbX	%EXXXX
%FXXXXbX	%SXXXX
%IXXXXbX	%AXXXX

OPER1	OPER2
%DXXXXwX	
%FXXXXwX	%DXXXXwX
%IXXXXwX	%FXXXXwX
%MXXXX	%IXXXXwX
%KMXXXXX	
%KDXXXXX	

OPER1	OPER2
%DXXXXwX	
%FXXXXwX	%MXXXX
%IXXXXwX	

Tabela 3-8. Sintaxe da instrução MOP

MOB - Movimentação de Blocos de Operandos



OPER1 - primeiro operando do bloco origem
OPER2 - número de transferências a realizar
OPER3 - operando de controle
OPER4 - primeiro operando do bloco destino
OPER5 - número de transferências por varredura

Descrição:

Esta instrução realiza a cópia dos valores de um bloco de operandos origem para um bloco destino.

Especifica-se o primeiro operando do bloco origem em OPER1 e o primeiro operando do bloco destino em OPER4. O número total de transferências a serem realizadas é declarado no parâmetro OPER2, devendo também ser especificados o número de transferências por varredura (OPER5) e uma memória acumuladora para a contagem do número de transferências (OPER3).

Se o bloco origem ou destino for uma tabela, a transferência tem início na sua primeira posição.

Se o formato do operando destino for menor que o de origem, os octetos mais significativos do valor origem são desprezados. Caso contrário, os octetos mais significativos do destino são zerados.

O número de transferências por varredura é limitado em 255 operandos. Na medida do possível deve-se evitar um número elevado de transferências na mesma varredura, para diminuir o tempo de execução do programa.

Em cada instrução MOB é utilizada uma memória como operando de controle (OPER3), que deve estar zerada antes da primeira execução.

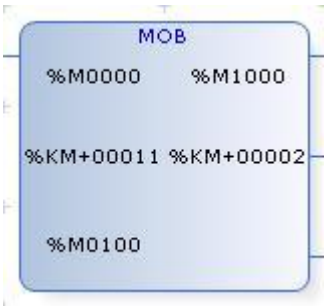
ATENÇÃO: O operando de controle não deve ter seu conteúdo alterado em nenhuma parte do programa aplicativo, sob pena de prejudicar a execução correta da instrução. Cada ocorrência desta instrução no programa deve possuir um operando de controle exclusivo, diferente dos demais. Este operando não pode ser retentivo.

Quando ligadas, as saídas da segunda e terceira células indicam, respectivamente,endereço superior ao número máximo declarado para o operando ou tabela utilizada, não sendo realizada nenhuma movimentação. Caso o valor do segundo operando seja negativo, a saída **índice origem inválido** é acionada.

A saída da primeira célula é acionada na varredura em que a movimentação for completada.

ATENÇÃO: A entrada **habilita** deve permanecer ativa até que a movimentação esteja concluída. Como esta instrução é executada em múltiplos ciclos de execução, não deve ser saltada enquanto não estiver terminada a movimentação.

Exemplo:



No exemplo acima se deseja movimentar os valores do bloco de operandos de %M0000 até %M0010, pois o OPER1 indica o inicio do bloco origem para movimentação e o OPER2 indica o seu tamanho. Em OPER4 temos o operando inicial da faixa destino, ou seja, o primeiro operando do bloco destino ao qual se deseja movimentar os valores é %M1000. OPER5 representa o número de operandos que deve ser movimentado a cada varredura do CP, que neste caso será 2 (%KM+00002). Em OPER3 (%M0100) é armazenado qual foi a última movimentação feita pela instrução.

Sintaxe:

OPER1	OPER2	OPER3	OPER4	OPER5
%E %S %A %M %I %D %TM %TI %TD	%KM	%M	%E %S %A %M %I %D %TM %TI %TD	%KM

OPER1	OPER2	OPER3	OPER4	OPER5
%F %TF	%KM	%M	%F %TF	%KM

Tabela 3-9. Sintaxe da instrução MOB

Tabela Verdade (Válida somente para a PO3x47):

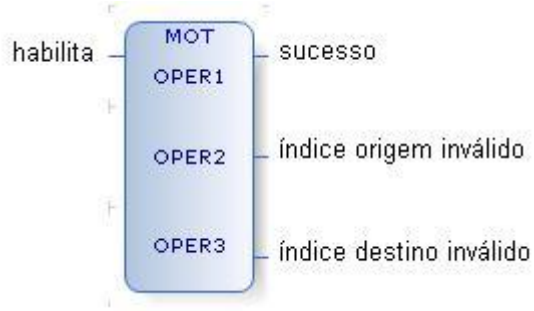
Tabela Verdade - instrução MOB								
	Entradas				Saídas			
Situação	Habilita	Oper1 Inval	Oper3 Inval	Oper4 Inval	Oper4	Concl.	Org Inv	Dest Inv
Instrução não habilitada	0	x	x	x	inalterado	0	0	0
Operando 1 Inválido	1	1	x	x	inalterado	0	1	0
Operando 3 Inválido [1]	1	0	1	x	inalterado	0	1	0
Operando 4 Inválido	1	0	0	1	inalterado	0	0	1
Cópia parcial [2]	1	0	0	0	Oper1	0	0	0
Operação completada	1	0	0	0	Oper1	1	0	0

Tabela 3-10. Tabela verdade da instrução MOB

Legendas:

- Inval = inválido
- Org.Inv = origem inválido
- Dest.Inv = destino inválido
- 1 = Esta indicação está disponível somente nas UCP's PO3x47
- 2 = Operação ainda não concluída, ocorre quando a transferência está em andamento

MOT - Movimentação de Tabelas



OPER1 - tabela origem ou operando origem
OPER2 - índice da tabela
OPER3 - operando destino ou tabela destino

Descrição:

Esta instrução permite duas operações: transferir o valor de uma posição de tabela para um operando simples ou de um operando simples para uma posição de tabela.

O operando que ocupa a primeira célula da instrução (OPER1) é o operando origem, cujo valor é movimentado para o operando destino especificado na terceira célula (OPER3). OPER2 contém a posição da tabela declarada em OPER1 ou OPER3.

Leitura de conteúdo de tabela:

Permite ler o conteúdo de uma posição de tabela e carregá-lo em um operando memória ou decimal.

A instrução é programada da seguinte forma:

- OPER1 - especifica o endereço da tabela a ser lida
- OPER2 - especifica a posição (%KM) a ser lida ou a memória (%M) que contém esta posição
- OPER3 - especifica para onde o conteúdo da posição de tabela deve ser transferido

Se o primeiro operando referenciar indiretamente uma tabela não especificada, ou se o valor do segundo operando for negativo ou maior que a última posição definida para a tabela, a transferência não é realizada e a saída **índice origem inválido** é acionada. Se o terceiro operando referenciar indiretamente um operando não declarado, a transferência não é realizada e a saída **índice destino inválido** é acionada. Quando ambos os operando de origem e destino forem inválidos, a saída **índice origem inválido** é acionada.

A saída **sucesso** é energizada quando a movimentação foi realizada com sucesso.

Escrita de valores em tabela:

Permite escrever um valor constante ou o conteúdo de um operando memória ou decimal em uma posição de tabela.

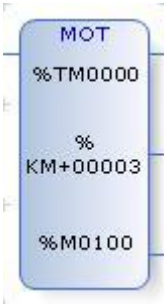
A instrução é programada da seguinte forma:

- OPER1 - especifica o operando origem
- OPER2 - especifica a posição (%KM) a ser escrita na tabela ou a memória (%M) que contém esta posição
- OPER3 - especifica o endereço da tabela para onde é transferido o conteúdo

Se o primeiro operando referenciar indiretamente um operando não declarado, a transferência do conteúdo não é realizada e a saída **índice origem inválido** é acionada. Se o valor do segundo operando for negativo ou maior que a última posição definida para a tabela, ou se o terceiro operando referenciar indiretamente uma tabela não especificada, a transferência do conteúdo não é realizada e a saída **índice destino inválido** é acionada.

Esta instrução simplifica a programação de uma série de algoritmos envolvendo decodificações, seqüenciamentos, geração de curvas, armazenamento e comparação de valores, entre outros.

Exemplo:



Neste exemplo o valor da posição 3 (%KM00003) da tabela %TM0000 será transferido para a memória %M0100.

Sintaxe:

Leitura:

OPER1	OPER2	OPER3
%TM %M*TM	%KM %M	%M %M*M

OPER1	OPER2	OPER3
%TD %M*TD	%KM %M	%D %M*D

OPER1	OPER2	OPER3
%TF %M*TF	%KM %M	%F %M*F

OPER1	OPER2	OPER3
%TI %M*TI	%KM %M	%I %M*I

Escrita:

OPER1	OPER2	OPER3
%KM %M %M*M	%KM %M	%TM %M*TM

OPER1	OPER2	OPER3
%KD %D %M*D	%KM %M	%TD %M*TD

OPER1	OPER2	OPER3
%KF %F %M*F	%KM %M	%TF %M*TF

OPER1	OPER2	OPER3
%KI %I %M*I	%KM %M	%TI %M*TI

Tabela 3-11. Sintaxe da instrução MOT

Tabela Verdade (Válida somente para a PO3x47)::

Tabela Verdade - instrução MOT: Leitura de conteúdo de tabela								
Entradas					Saídas			
Situação	Habilita	Oper1 Inval	Oper2 <0 ou > tam	Oper3 Inval	Oper3	Suces	Org Inv	Dest Inv
Instrução não habilitada	0	x	x	x	inalterado	0	0	0
Operando 1 Inválido	1	1	x	x	inalterado	0	1	0
Operando 2 negativo ou fora da tabela	1	0	1	x	inalterado	0	1	0
Operando 3 Inválido	1	0	0	1	inalterado	0	0	1
Operação normal	1	0	0	0	valor lido	1	0	0

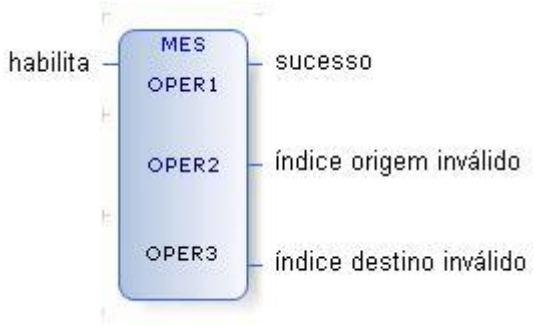
Tabela Verdade - instrução MOT: Escrita de valores em tabela								
Entradas					Saídas			
Situação	Habilita	Oper1 Inval	Oper2 <0 ou > tam	Oper3 Inval	Oper3	Suces	Org Inv	Dest Inv
Instrução não habilitada	0	x	x	x	inalterado	0	0	0
Operando 1 Inválido	1	1	x	x	inalterado	0	1	0
Operando 2 negativo ou fora da tabela	1	0	1	x	inalterado	0	0	1
Operando 3 Inválido	1	0	0	1	inalterado	0	0	1
Operação normal	1	0	0	0	valor lido	1	0	0

Tabela 3-12. Tabela verdade da instrução MOT

Legendas:

- Inval = inválido
- Org.Inv = origem inválido
- Dest.Inv = destino inválido
- Suces = sucesso.

MES - Movimentação de Entradas/Saídas



OPER1 - primeiro operando origem
OPER2 - número de octetos a transferir
OPER3 - primeiro operando destino

Descrição:

Esta instrução é utilizada para a transferência de dados diretamente entre operandos memória e octetos do barramento de módulos de entrada e saída. É possível realizar leituras de valores dos octetos do barramento ou escritas no mesmo, conforme os operandos programados na instrução.

O operando que ocupa a primeira célula da instrução (OPER1) é o operando origem, cujo conteúdo será movimentado para o operando destino especificado na terceira célula (OPER3). OPER2 define o número de octetos a serem transferidos a partir do primeiro origem e destino especificados.

ATENÇÃO: O número de octetos a serem transferidos está limitado em 255.

Caso seja programada uma constante na primeira célula (escrita de valor no barramento), o seu valor é movido para todos os octetos do barramento especificados pelos operandos da segunda e terceira células.

Sempre que a entrada **habilita** está energizada, uma das saídas da instrução é energizada, conforme as regras a seguir.

A saída **índice origem inválido** é energizada em 3 situações:

- O número de transferências especificado em OPER2 for negativo, zero, maior do que o número máximo de octetos no barramento do CP utilizado (leitura do barramento) ou que o limite de memórias configurado (escrita no barramento)
- A primeira posição lida for maior do que o número máximo de octetos no barramento do CP utilizado (%M*R programado em OPER1)
- O primeiro endereço de memória a ser escrito for negativo ou maior do que o último endereço de memória configurado (%M*M programado em OPER1)

A saída **índice destino inválido** é energizada quando:

- O número de transferências especificado em OPER2 for maior do que o limite de memórias configurado (leitura do barramento) ou que o número máximo de octetos no barramento do CP utilizado (escrita no barramento)
- A primeira posição escrita for maior do que o número máximo de octetos no barramento do CP utilizado (%M*R programado em OPER3)
- O primeiro endereço de memória a ser lido for negativo ou maior do que o último endereço de memória configurado (%M*M programado em OPER3)

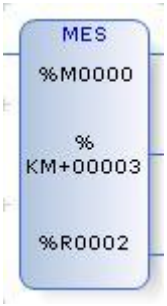
A saída **sucesso** é energizada quando as saídas **índice origem inválido** e **índice destino inválido** não forem energizadas.

Esta instrução é utilizada somente para acessos especiais ao barramento. Para o seu uso, deve-se saber exatamente que módulo de E/S está colocado na posição física do barramento lida ou escrita pela MES e como acessá-lo. Como os módulos de entrada e saída fornecidos pela ALTUS possuem instruções específicas para o seu acesso, a instrução MES não é necessária na maioria dos programas aplicativos.

Não é possível escrever valores em octetos de módulos digitais de entrada ou ler valores de octetos de módulos digitais de saída com a instrução **MES**.

ATENÇÃO: Esta instrução só é válida para os CPs da série AL.

Exemplo:



Neste caso, os valores faixa de memória %M0000 até %M0002 (o operando %KM+00003 determina o tamanho da faixa) serão movimentados para os registradores de barramento %R0002 até %R0004.

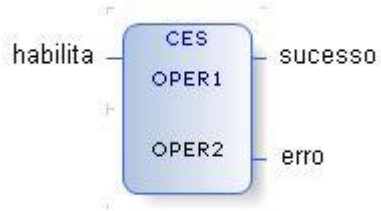
Sintaxe:

OPER1	OPER2	OPER3
%R %M*R	%KM %M	%M %M*M

OPER1	OPER2	OPER3
%KM %M %M*M	%KM %M	%R %M*R

Tabela 3-13. Sintaxe da instrução MES

CES - Conversão de Entradas/Saídas



OPER1 - operando– origem
OPER2 - operando destino

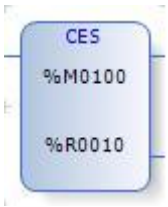
Descrição:

Esta instrução é utilizada para a transferência de dados diretamente entre operandos memória e octetos no barramento, convertendo os valores de binário para BCD, em caso de escrita no barramento, ou BCD para binário, em caso de leitura.

Caso se deseje converter octetos do barramento para uma memória, deve-se programar em OPER1 o octeto inicial e em OPER2 a memória a receber os valores convertidos. A instrução concatena o valor do octeto especificado com o octeto seguinte, converte do formato BCD para binário e armazena o valor convertido na memória destino.

Caso se deseje converter valores de uma memória ou constante memória para o barramento, deve-se especificar em OPER1 o valor a ser convertido em OPER2 o octeto inicial a receber os valores. A instrução converte o valor para o formato BCD e escreve o mesmo no octeto especificado e no seguinte. Se o valor movimentado para o barramento possuir mais do que 4 dígitos, os dígitos mais significativos excedentes serão descartados.

Exemplo:



Movimentar o conteúdo de %M0100 para %R0010:

- Valor de %M0100=21947, equivalente a 101010110111011 no formato binário
- Valor de %M0100=21947, convertido para 0010 0001 1001 0100 0111 no formato BCD
- Valor movido para %R0010=47 no formato BCD, equivalente a 0100 0111 escrito no octeto
- Valor movido para %R0011=19 no formato BCD, equivalente a 0001 1001 escrito no octeto

A instrução é executada sempre que a entrada **habilita** é energizada. A saída sucesso é energizada caso a instrução tenha sido executada corretamente.

A saída erro é energizada quando se faz um acesso inválido a algum operando referenciado indiretamente por uma memória.

Sintaxe:

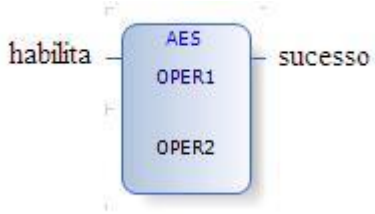
OPER1	OPER2
%R	%M
%M*R	%M*M

OPER1	OPER2
%KM	
%M	%R
%M*M	%M*R

Tabela 3-14. Sintaxe da instrução CES

ATENÇÃO: Esta instrução só é válida para os CPs da série AL.

AES - Atualiza Entradas/Saídas



OPER1 - primeiro octeto de operandos a atualizar
OPER2 - número de octetos a atualizar

Descrição:

Esta instrução executa uma atualização imediata na memória imagem para os operandos especificados. Sua atuação é idêntica à varredura dos pontos de E/S realizada pelo programa executivo ao final de cada varredura, porém com o número de operandos limitados.

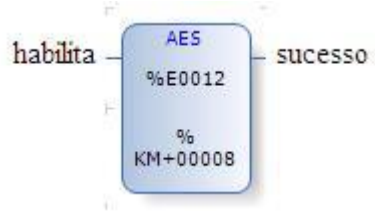
O primeiro operando (OPER1) contém o primeiro octeto de operandos a ser atualizado, enquanto que o segundo operando (OPER2) indica o número total de octetos a atualizar. Os operandos %E (entrada) são lidos do barramento para a memória imagem e os operandos %S (saída) são escritos da memória imagem para o barramento quando a instrução é executada.

Se o número de operandos a atualizar ultrapassar o número de operandos declarados, são atualizados somente os possíveis dentro do tipo declarado.

Se nenhum octeto for atualizado pela instrução, a saída **sucesso** é desenergizada.

A instrução AES deve ser usada somente em processamentos especiais, nos quais um tempo de resposta muito rápido ou constante é exigido do CP. Em programas aplicativos relativamente pequenos, com baixo tempo de varredura e tarefas de controle comuns, a mesma não necessita ser utilizada.

Exemplo:



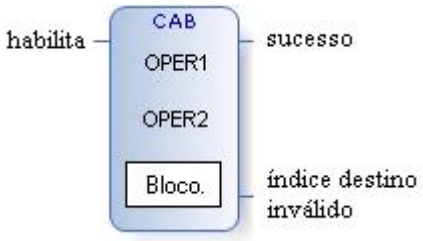
Caso a configuração do CP seja 16 octetos de entrada (%E0000 a %E0015) e 8 octetos de saída (%S0016 a %S0023), a instrução apresentada atualizaria apenas 4 octetos (%E0012 a %E0015). Nenhum octeto de saída é atualizado.

Sintaxe:

OPER1	OPER2
%E	%KM
%S	%M

Tabela 3-15. Sintaxe da instrução AES

CAB - Carrega Bloco



OPER1 - operando inicial ou tabela a ser carregada
OPER2 - quantidade de operandos ou posições de tabela

Descrição:

Esta instrução permite a carga de até 255 valores constantes em um bloco de operandos ou em tabelas.

O operando inicial ou tabela a ser carregada é especificado no primeiro parâmetro (OPER1), a quantidade de operandos ou posições da tabela a serem carregados no segundo operando (OPER2).

O valor do segundo operando deve ser positivo, menor ou igual a %KM+255.

Estes valores são declarados selecionando o botão **Bloco**, sendo aberta uma janela de edição no MasterTool. As constantes editadas no bloco variam conforme o tipo de operando:

- São %KM se o tipo do primeiro operando for %E, %S, %A, %M, %TM
- São %KD se o primeiro operando for %D ou %TD
- São %KF se o primeiro operando for %F ou %TF
- São %KI se o primeiro operando for %I ou %TI

Caso o primeiro operando seja um octeto (%E, %S ou %A), somente serão movimentados os valores dos octetos menos significativos de cada constante declarada.

Quando o botão **Bloco** é selecionado é exibida a janela edição dos valores conforme é mostrado na figura a seguir:

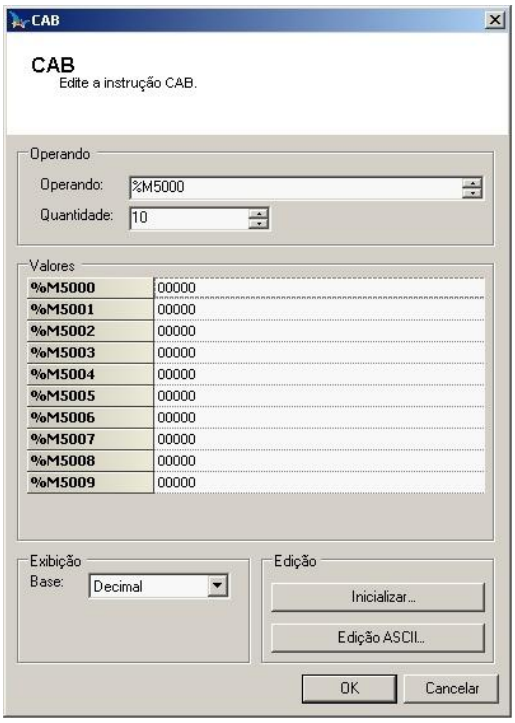


Figura 3-3. Janela de edição da CAB

Também é possível realizar a declaração dos valores da tabela em ASCII. Ao pressionar o botão **Edição ASCII** que a abre a janela **CAB – Edição em ASCII** onde é possível digitar um texto que será carregado na tabela com os valores ASCII relativos a cada caractere. A edição dos valores em ASCII está limitada aos operandos %TM e ao tamanho máximo de 128 posições de tabela.

Figura 3-4. Janela de edição da CAB em ASCII

Os valores podem ser inicializados com um único valor através da janela **CAB – Inicializar Bloco**, acessada através do botão **Inicializar.** Nesta janela é possível editar o valor e as posições para onde o valor será movido.

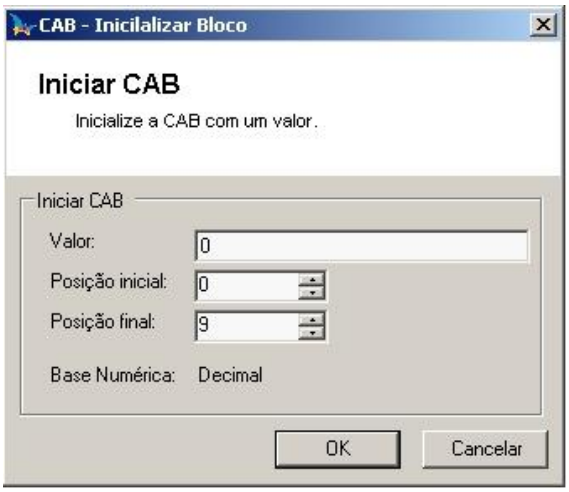


Figura 3-5. Janela de inicialização da CAB

A saída **índice destino inválido** é acionada quando algum operando não puder ser acessado ou uma posição de tabela não existir. A saída **sucesso** é acionada sempre que a instrução for executada corretamente. Se a saída **índice destino inválido** foi acionada, nenhuma carga de constantes ocorreu.

A carga dos valores constantes é inteiramente realizada em uma só varredura do programa aplicativo, podendo ocasionar um tempo de ciclo excessivo quando o mesmo for extenso. Na maior parte dos programas aplicativos, a instrução CAB pode ser executada somente na inicialização do mesmo (carga de tabelas cujos conteúdos serão somente lidos) ou em alguns momentos especiais, não precisando ser chamada em todas as varreduras. Nestes casos, recomenda-se a sua programação no módulo de programa aplicativo de inicialização (E000) ou que seja acionada apenas nos momentos de carga necessários.

Exemplo:

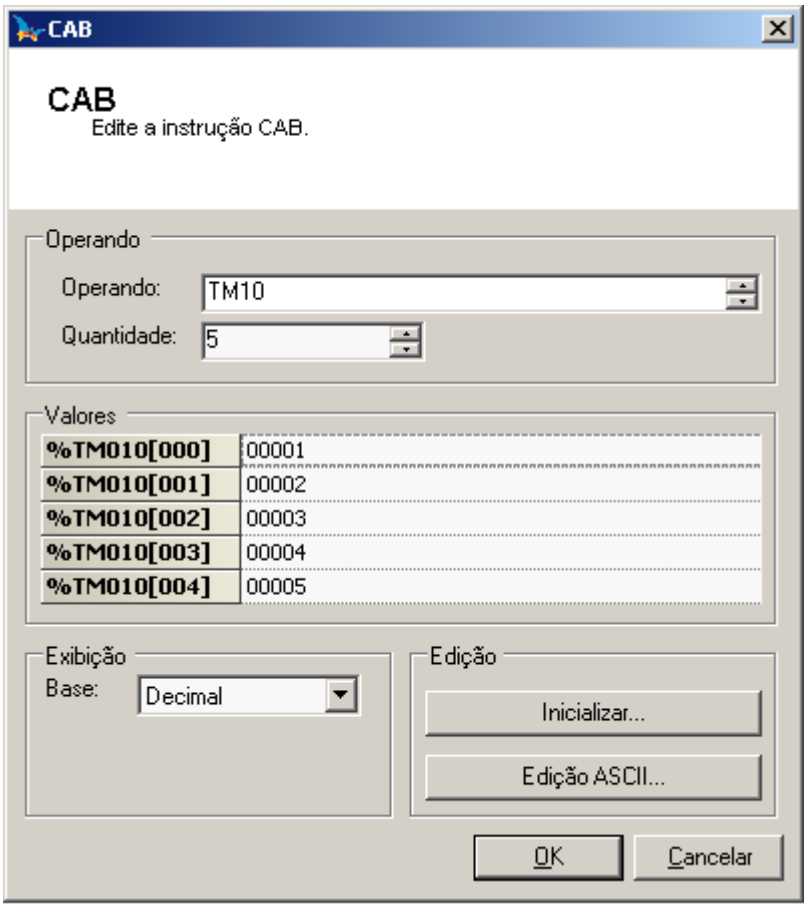


Figura 3-6. Exemplo de configuração da instrução CAB

Este exemplo mostra como configura a instrução CAB para copiar os valores 1, 2, 3, 4 e 5 para as primeiras 5 posições do operando %TM0010.

Sintaxe:

OPER1	OPER2	OPER3
%E %S %A %M %TM %M*E %M*S %M*A %M*M %M*TM	%KM	Tabela de valores memória

OPER1	OPER2	OPER3
%D %TD %M*D %M*TD	%KM	Tabela de valores decimais

OPER1	OPER2	OPER3
%F %TF %M*F %M*TF	%KM	Tabela de valores reais

OPER1	OPER2	OPER3
%I %TI %M*I %M*TI	%KM	Tabela de valores inteiro

Tabela 3-16. Sintaxe da instrução CAB

Instruções do Grupo Aritméticas

As instruções aritméticas modificam os valores dos operandos numéricos, permitindo a realização de cálculos aritméticos e lógicos entre os mesmos. Permitem também comparações entre valores de operandos.

Nome	Descrição do Nome	Seqüência de Edição
SOM	Adição	Alt, I, A, S
SUB	Subtração	Alt, I, A, B
MUL	Multiplicação	Alt, I, A, M
DIV	Divisão	Alt, I, A, D
AND	Função **E** binário entre operandos	Alt, I, A, A
OR	Função **OU** binário entre operandos	Alt, I, A, O
XOR	Função **OU EXCLUSIVO** entre operandos	Alt, I, A, X
CAR	Carrega operando	Alt, I, A, C
IGUAL	Igual	Alt, I, A, I
MENOR	Menor	Alt, I, A, N
MAIOR	Maior	Alt, I, A, R

Tabela 3-17. Instruções do grupo aritiméticas

SOM - Adição

OPER1 - primeira–parcela
OPER2 - segunda–parcela
OPER3 - resultado

Descrição:

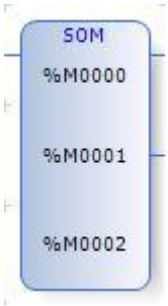
Esta instrução realiza a soma aritmética de operandos. Quando a entrada **habilita** é energizada, os valores dos operandos especificados nas duas primeiras células são somados e o resultado armazenado no operando da terceira célula.

Se o resultado da operação for maior ou menor do que o armazenável, a saída **estouro** é energizada e o máximo ou mínimo valor armazenável é atribuído a variável total como resultado.

Se a entrada **habilita** não está energizada, todas as saídas são desenergizadas e o valor de OPER3 não é alterado.

Quando o operando destino da instrução é um inteiro (%M) e pelo menos um dos demais operandos da instrução é um real (%F) o resultado armazenado será truncado, ou seja, armazena-se no operando M apenas a parte inteira do resultado da operação, desprezando-se a parte fracionária.

Exemplo:



Suponhamos que o operando %M0000 possua o valor 100, e o operando %M0001 possua o valor 34. Após a entrada habilita energizada, o valor no operando %M0002 será de 134 e a saída copia da entrada também será energizada.

No caso em que o valor do operando %M0000 seja 30000 e o de %M0001 seja 15000, por exemplo, a saída **estouro** será energizada, pois ultrapassou o limite de armazenamento dos operandos %M.

Sintaxe:

OPER1	OPER2	OPER3
%KD %D	%KD %D	 %D

OPER1	OPER2	OPER3
%KF	%KF	
%F	%F	%F
%KM	%KM	%M
%M	%M	%I

OPER1	OPER2	OPER3
%KM	%KM	
%M	%M	%M
%KI	%KI	%I
%I	%I	

Tabela 3-18. Sintaxe da instrução SOM

SUB - Subtração

OPER1 - minuendo
OPER2 - subtraendo
OPER3 - resultado

Descrição:

Esta instrução realiza a subtração aritmética entre operandos. Quando **habilita** é energizada, o valor do operando da segunda célula é subtraído do valor do operando da primeira célula. O resultado é armazenado na memória especificada na terceira célula.

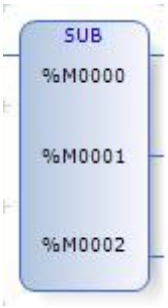
As linhas de saída **resultado > 0**, **resultado = 0** e **resultado < 0** podem ser usadas para comparações e são acionadas de acordo com o resultado da subtração.

Se a entrada **habilita** não está energizada, todas as saídas são desenergizadas e OPER3 permanece inalterado.

Se o resultado da operação excede o maior ou menor valor armazenável no operando, o respectivo valor limite é considerado como resultado.

Quando o operando destino da instrução é um inteiro (%M) e pelo menos um dos demais operandos da instrução é um real (%F) o resultado armazenado será truncado, ou seja, armazena-se no operando M apenas a parte inteira do resultado da operação, desprezando-se a parte fracionária.

Exemplo:



Suponhamos que o operando %M0000 possua o valor 100, e o operando %M0001 possua o valor 34. Após a entrada habilita energizada, o valor no operando %M0002 será de 66 e a saída resultado > 0 (maior que zero) será energizada.

No caso em que o valor do operando %M0000 seja 1000 e o de %M0001 seja 1500, por exemplo, a saída resultado < 0 (menor que zero) será energizada.

E finalmente se o valor de %M0000 for 10 e o de %M0001 também for 10, a saída resultado =0 (igual a zero) será habilitada.

Sintaxe:

OPER1	OPER2	OPER3
%KD %D	%KD %D	 %D

OPER1	OPER2	OPER3
%KF	%KF	
%F	%F	%F
%KM	%KM	%M
%M	%M	%I

OPER1	OPER2	OPER3
%KM	%KM	%M
%M	%M	%I
%KI	%KI	
%I	%I	

Tabela 3-19. Sintaxe da instrução SUB

MUL - Multiplicação

OPER1 - multiplicando
OPER2 - multiplicador
OPER3 - produto

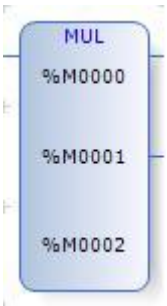
Descrição:

Esta instrução realiza a multiplicação aritmética de operandos. Quando a entrada **habilita** está energizada, ocorre a multiplicação do conteúdo do operando especificado na primeira célula pelo especificado na segunda.

O resultado é armazenado na memória especificada na terceira célula. Caso este exceda o valor máximo armazenável em uma memória, o resultado final é este valor e a saída **estouro** é energizada. Se a entrada **habilita** é desenergizada, nenhuma saída é ligada e OPER3 permanecerá inalterado.

Quando o operando destino da instrução é um inteiro (%M) e pelo menos um dos demais operandos da instrução é um real (%F) o resultado armazenado será truncado, ou seja, armazena-se no operando M apenas a parte inteira do resultado da operação, desprezando-se a parte fracionária.

Exemplo:



No caso acima, suponhamos que o valor de %M0000 seja 10, e o valor de %M0001 seja 6, após a entrada habilita ser energizada, o valor em %M0002 será de 60, e a saída cópia da entrada será habilitada. Entretanto, se o valor de %M0000 for 10000 e o valor de %M0001 for de 5, a saída estouro será energizada.

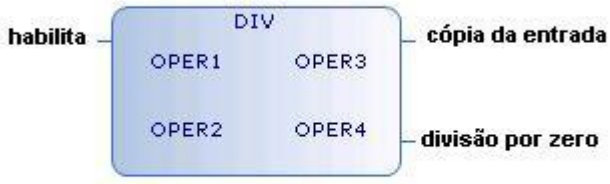
Sintaxe:

OPER1	OPER2	OPER3
%KF	%KF	
%F	%F	%F
%KM	%KM	%M
%M	%M	%I

OPER1	OPER2	OPER3
%KM	%KM	
%M	%M	
%KI	%KI	%M
%I	%I	%I

Tabela 3-20. Sintaxe da instrução MUL

DIV - Divisão



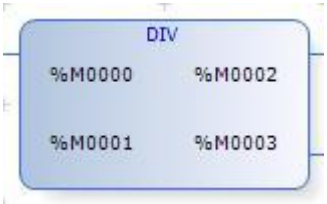
OPER1 - dividendo
OPER2 - divisor
OPER3 - quociente
OPER4 - resto

Descrição:

Esta instrução realiza a divisão aritmética de operandos. Quando a entrada **habilita** está energizada, ocorre a divisão do valor do operando da primeira célula pelo da segunda, sendo o resultado armazenado na memória especificada na terceira célula e o resto da operação colocado no quarto operando. Os operandos da primeira e segunda células podem ser do tipo memória ou constante.

Se o valor do segundo operando for zero, a saída **divisão por zero** é acionada e em OPER3 é colocado o valor máximo ou mínimo armazenável no operando, conforme o sinal de OPER1. Neste caso, em OPER4 (resto) será armazenado zero. As saídas da instrução somente são energizadas se a entrada **habilita** estiver acionada. Se não estiver acionada, OPER3 e OPER4 permanecerão inalterados.

Exemplo:



Suponhamos que na instrução acima o valor de %M0000 seja 11, e o valor de %M0001 seja 10. Após a energização da entrada habilita, o resultado da divisão aparecera no operando %M0002 com o valor 1, e o no operando %M0003 teremos o valor 1, que e o resto da divisão. Se tudo ocorrer normalmente a saída cópia da entrada também será habilitada, caso ocorra divisão por zero, a saída com este mesmo nome será habilitada.

Sintaxe:

OPER1	OPER2	OPER3	OPER4
%KM %M %KI %I	%KM %M %KI %I	%I	%I

OPER1	OPER2	OPER3	OPER4
%KF %F %KM %M	%KF %F %KM %M	%M	%M

OPER1	OPER2	OPER3	OPER4
%KF %F %KM %M	%KF %F %KM %M	%F	%M(NU) %F(NU) %I(NU)

OPER1	OPER2	OPER3	OPER4
%KF %F %KM %M	%KF %F %KM %M	%I	%I

OPER1	OPER2	OPER3	OPER4
%KM %M %KI %I	%KM %M %KI %I	%M	%M

Tabela 3-21. Sintaxe da instrução DIV

NU= Operando não utilizado pela instrução, podendo ser informado qualquer endereço de operando declarado no CP.

AND - E Binário entre Operandos

OPER1 - primeiro operando
OPER2 - segundo operando
OPER3 - resultado

Descrição:

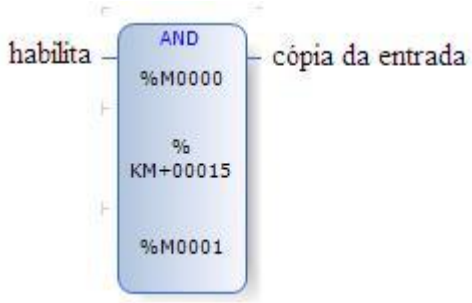
Esta instrução realiza a operação "e" binário entre os dois primeiros operandos, armazenando o resultado no terceiro.

A operação é realizada ponto a ponto entre os operandos. A tabela, a seguir, mostra as combinações da operação "e" ponto a ponto possíveis.

Ponto OPER1	Ponto OPER2	Ponto OPER3 (resultado)
0	0	0
0	1	0
1	0	0
1	1	1

Tabela 3-22. Operações ponto a ponto (AND)

Exemplo:



Neste exemplo deseja-se preservar o valor do nibble menos significativo de %M0000, zerando o resto do operando. Se %M0000 contém 215 (11010111 binário), o resultado do "e" binário com 15 (00001111 binário) é 7 (00000111 binário).

Decimal	Binário
215	00000000 11010111 (conteúdo de %M0000)
AND 15	AND 00000000 00001111 (valor de %KM+00015)
7	00000000 00000111 (resultado em %M0001)

Tabela 3-23. Exemplo de uma operação AND

Portanto, o valor 7 decimal é armazenado em %M0001.

Sintaxe:

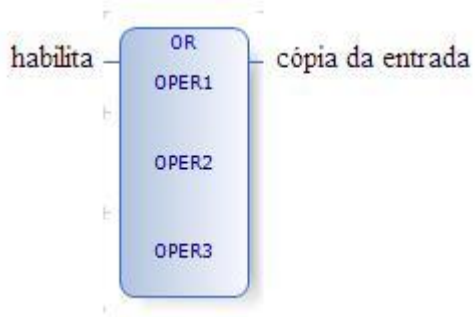
OPER1	OPER2	OPER3
%KM %M	%KM %M	%M

OPER1	OPER2	OPER3
%KD	%KD	
%D	%D	%D

OPER1	OPER2	OPER3
%KI	%KI	
%I	%I	%I

Tabela 3-24. Sintaxe da instrução AND

OR - Ou Binário entre Operandos



OPER1 - primeiro operando
OPER2 - segundo operando
OPER3 - resultado

Descrição:

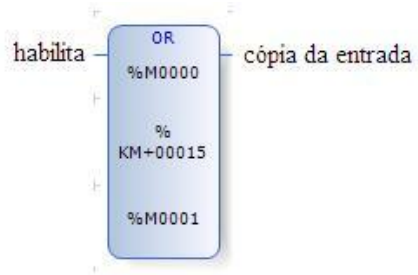
Esta instrução realiza a operação "ou" binário entre os valores dos dois primeiros operandos, armazenando o resultado no terceiro.

A operação é realizada ponto a ponto entre os operandos. A tabela, a seguir, mostra as combinações da operação "ou" ponto a ponto possíveis.

Ponto OPER1	Ponto OPER2	Ponto OPER3 (resultado)
0	0	0
0	1	1
1	0	1
1	1	1

Tabela 3-25. Operações ponto a ponto (OR)

Exemplo:



Neste exemplo deseja-se forçar o nibble menos significativo de %M0000 para 1, preservando-se o valor nos outros nibbles. Se %M0000 contém 28277 (0110111001110101 binário) o resultado é 28287 (0110111001111111 binário).

Decimal	Binário
28277	01101110 01110101 (conteúdo de %M0000)
OR 15	OR 00000000 00001111 (valor de %KM+00015)
28287	01101110 01111111 (resultado em %M0001)

Tabela 3-26. Exemplo de uma operação OR

Sintaxe:

OPER1	OPER2	OPER3
%KM %M	%KM %M	 %M

OPER1	OPER2	OPER3
%KD	%KD	
%D	%D	%D

OPER1	OPER2	OPER3
%KI	%KI	
%I	%I	%I

Tabela 3-27. Sintaxe da instrução OR

XOR - Ou Exclusivo entre Operandos

OPER1 - primeiro operando
OPER2 - segundo operando
OPER3 - resultado

Descrição:

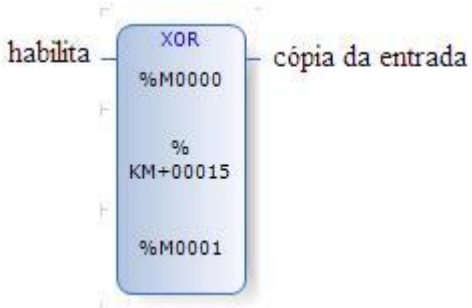
Esta instrução realiza a operação "ou exclusivo" binário entre os dois primeiros operandos, armazenando o resultado no terceiro.

A operação é realizada ponto a ponto entre os operandos. A tabela, a seguir, mostra as combinações da operação "ou exclusivo" ponto a ponto possíveis.

Ponto OPER1	Ponto OPER2	Ponto OPER3 (resultado)
0	0	0
0	1	1
1	0	1
1	1	0

Tabela 3-28. Operações ponto a ponto (XOR)

Exemplo:



Neste exemplo deseja-se inverter os pontos contidos no nibble menos significativo de %M0000, preservando o resto do operando. Se %M0000 contém 1612 (0000011001001100 binário), o resultado é 1603 (0000011001000011 binário)

Decimal	Binário
1612	00000110 01001100 (conteúdo de %M0000)
XOR 15	XOR 00000000 00001111 (valor de %KM+00015)
1603	00000110 01000011 (resultado em %M0001)

Tabela 3-29. Exemplo de uma operação XOR

Portanto, o valor 1603 decimal é armazenado em M001.

Sintaxe:

OPER1	OPER2	OPER3
%KM %M	%KM %M	 %M

OPER1	OPER2	OPER3
%KD	%KD	
%D	%D	%D

OPER1	OPER2	OPER3
%KI	%KI	
%I	%I	%I

Tabela 3-30. Sintaxe da instrução XOR

CAR - Carrega Operando

OPER1 - operando a ser carregado

Descrição:

A instrução carrega operando realiza a carga do valor do operando especificado em registrador especial interno ao CP, para subseqüente uso das instruções de comparação (maior, menor, igual). O operando permanece carregado até a próxima instrução de carga, podendo ser utilizado por várias lógicas, inclusive em ciclos de varredura subseqüentes.

A saída **sucesso** é acionada se a carga for realizada. Se algum acesso indireto a operando não for possível (índice inválido), a saída **sucesso** não é acionada.

ATENÇÃO: Não pode ser feita a comparação entre operandos decimais e operandos flutuante.

ATENÇÃO: Ver considerações e exemplos apresentados na seção seguinte, **Instruções de Comparação de Operandos**.

Exemplo:

Ver exemplos de uso desta instrução em Instruções de Comparação de Operandos - Igual, Maior e Menor.

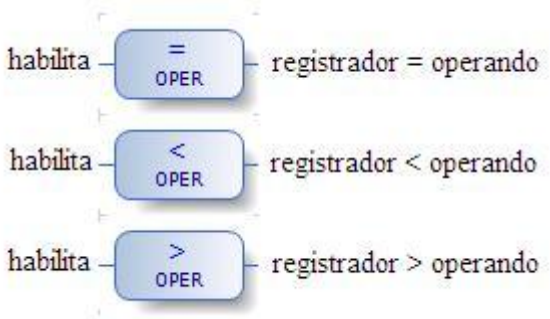
Sintaxe:

OPER1
%E
%S
%A
%M
%D
%F
%I
%KM
%KD
%KF
%KI
%M*E
%M*S
%M*A
%M*M
%M*D
%M*F
%M*I

Tabela 3-31. Sintaxe da instrução CAR

ATENÇÃO:
- Para se no caso, for habilitado uma instrução CAR e logo após desta, uma outra instrução CAR com operando indexado inválido, a instrução CAR permanece com o valor anteriormente carregado, mesmo se uma nova instrução CAR for acionada com operando inválido.
- As instruções de comparação serão executadas após a instrução CAR ter sido executada, caso contrário as instruções de comparação não retornarão verdadeiro.
- Se colocar instruções de comparação no módulo E000 e uma CAR no módulo E001, as instruções de comparação não serão executadas após o POWER-ON, mas sim após trocar o modo para programação/execução.

Instruções de Comparação de Operandos - Igual, Maior e Menor



OPER - operando a ser comparado

Descrição:

As instruções maior, menor e igual realizam comparações do operando especificado com o valor previamente carregado no registrador interno com a instrução CAR (Carrega Operando), fornecendo o resultado da comparação em suas saídas. Caso algum acesso indireto seja inválido, a saída é desacionada.

Por exemplo, a instrução maior energiza a sua saída se o valor do operando presente na última instrução CAR ativa for maior que o valor do seu operando. As instruções igual e menor operam de forma idêntica, mudando apenas o tipo de comparação realizada.

Se os operandos a serem comparados são do mesmo tipo, são comparados conforme o seu formato de armazenamento (considerando o seu sinal). Se não são do mesmo tipo, são comparados ponto a ponto (como valores binários sem sinal). Se algum dos operandos diferentes for do tipo real, o operando de menor precisão é convertido para real e após é efetuada a mesma comparação ponto a ponto.

ATENÇÃO: Sugere-se que sempre sejam comparados operandos de tipos iguais, para evitar má interpretação nos resultados quando os operandos possuírem valores negativos. Ver o exemplo a seguir.

ATENÇÃO: Não pode ser feita a comparação entre operandos decimais e operandos flutuante.

Exemplo:

Figura 3-7. Exemplo das instruções de comparação

Como os tipos dos operandos são diferentes (%M e %D), a comparação é realizada ponto a ponto, sem considerar os sinais aritméticos. Devido a este fato, se %M0012–possuir valor -45 e %D0010

possuir valor +21, o operando %A0003.2 será energizado, como se o valor de %M0012 fosse maior que %D0010, o que não ocorre na realidade.

%M0012	=-45					1111	1111	1101	0011
%D0000	=+21	0000	0000	0000	0000	0000	0000	0010	0001

Para considerar os sinais na comparação do exemplo, deve-se converter o valor do operando memória para um decimal, utilizando este último na instrução CAR, como mostrado na lógica a seguir:

O valor 1111 1111 1101 0011 (%M0012) é maior que 10 0001 (%D0010) na comparação ponto a ponto, mesmo representando um valor negativo.

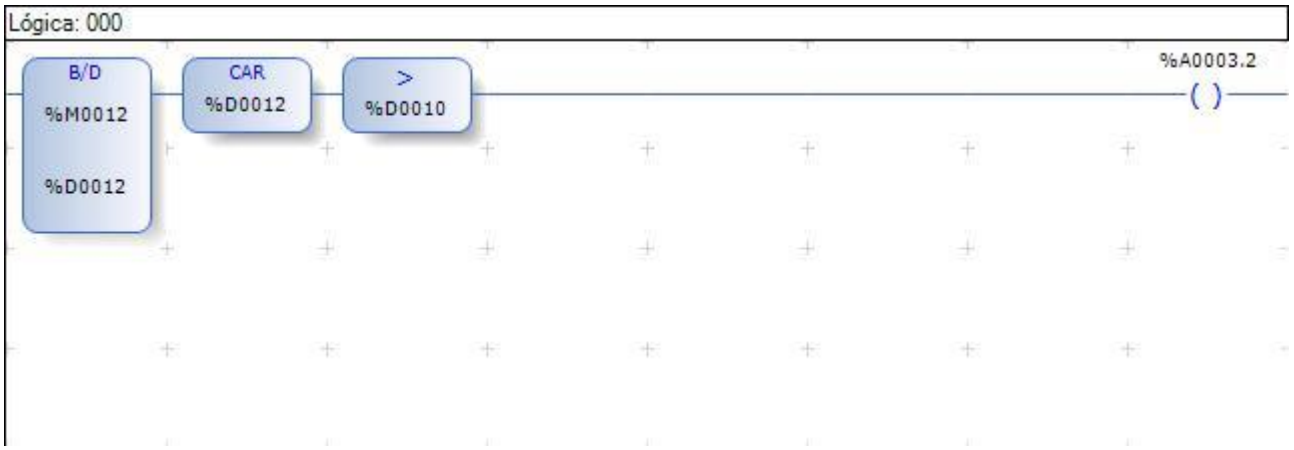


Figura 3-8. Exemplo das instruções de comparação

ATENÇÃO: Devido à ordem de processamento das instruções na lógica, deve-se cuidar o posicionamento das instruções de comparação para evitar erros na interpretação no seu funcionamento. Ver seção **Lógicas** no capítulo 2 e o exemplo a seguir.

Exemplo:

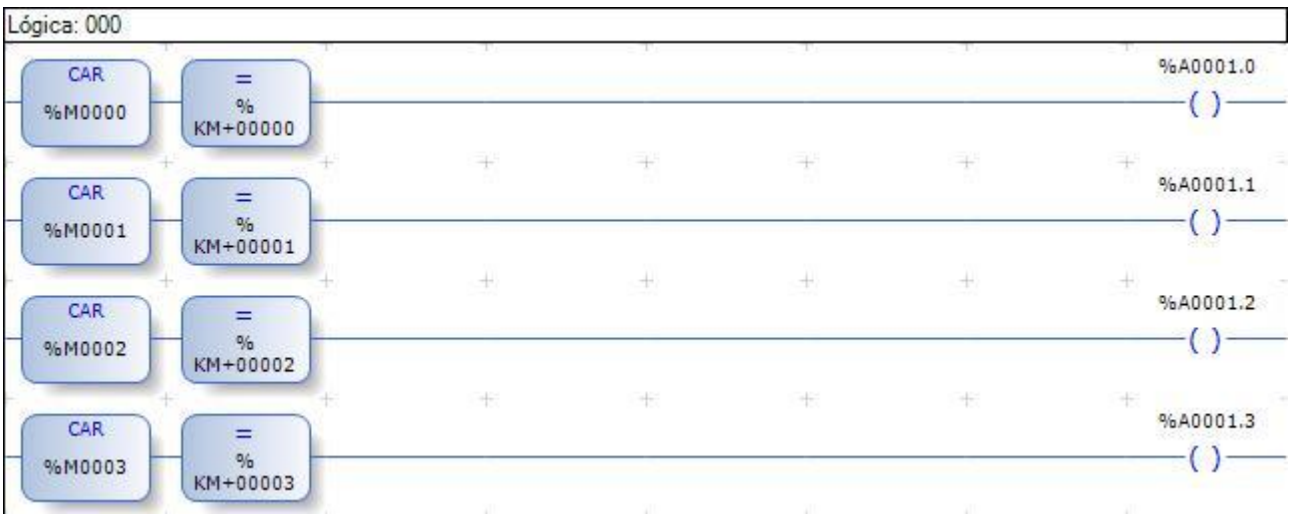


Figura 3-9. Utilização incorreta da instrução CAR

Na lógica apresentada, deseja-se comparar os valores dos operandos %M0000, %M0001, %M0002 e %M0003 com as constantes %KM00000, %KM00001, %KM00002 e %KM00003, respectivamente.

Entretanto, o funcionamento ocorre de forma diversa do que o aspecto visual sugere. Como o processamento da lógica ocorre em colunas, no final da execução da coluna 0 estará carregado o valor de %M0003 para as comparações na coluna 1. Na realidade, somente o valor do operando %M0003 será comparado com as constantes presentes na coluna 1.

Para o funcionamento desejado, a lógica deve ser programada da seguinte forma:

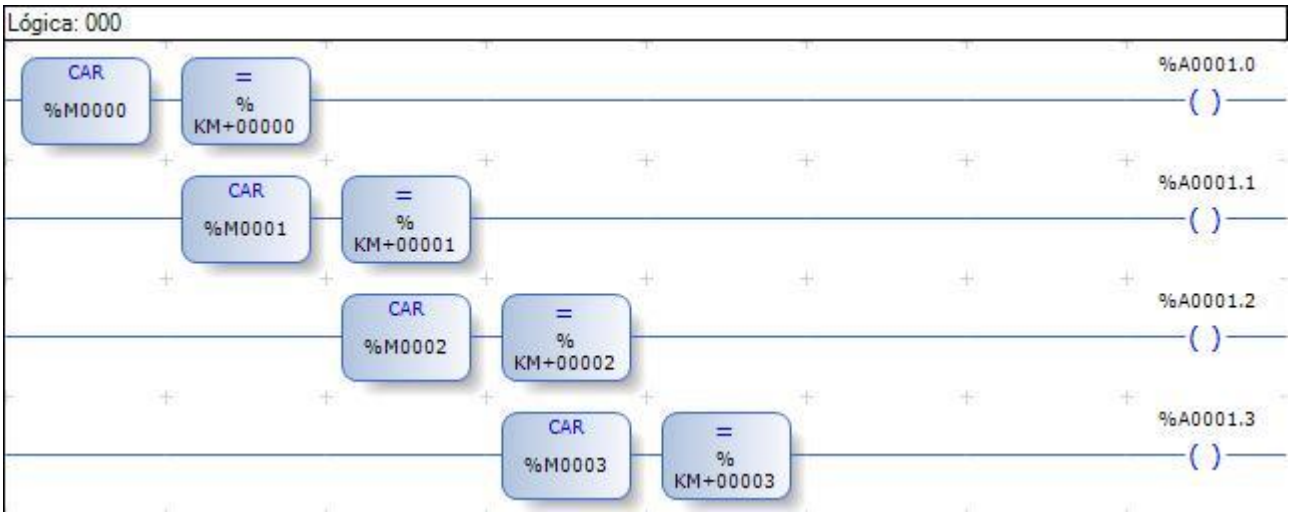


Figura 3-10. Utilização correta da instrução CAR

ATENÇÃO: Para evitar interpretações errôneas no funcionamento das instruções de comparação, sugere-se o uso de apenas uma instrução CAR por coluna da lógica.

Sintaxe:

OPER1
%E
%S
%A
%M
%D
%F
%I
%KM
%KD
%KF
%KI
%M*E
%M*S
%M*A
%M*M
%M*D
%M*F
%M*I

Tabela 3-32. Sintaxe das instruções Maior, Igual e Menor

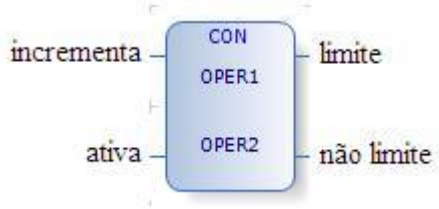
Instruções do Grupo Contadores

As instruções contadoras são utilizadas para realizar contagens de eventos ou de tempo no programa aplicativo.

Nome	Descrição do Nome	Seqüência de Edição
CON	Contador simples	Alt, I, C, N
COB	Contador bidirecional	Alt, I, C, B
TEE	Temporizador na energização	Alt, I, C, T
TED	Temporizador na desenergização	Alt, I, C, D

Tabela 3-33. Instruções do campo contadores

CON - Contador Simples



OPER1 - contador
OPER2 - limite de contagem

Descrição:

Esta instrução realiza contagens simples, com o incremento de uma unidade em cada acionamento.

A instrução contador simples possui dois operandos. O primeiro, sempre do tipo %M, especifica a memória que contabiliza os eventos. O segundo estabelece o valor limite de contagem para energização da saída da célula superior e pode ser do tipo %KM, %M ou operando %M referenciado indiretamente.

Se a entrada **ativa** está desenergizada, a memória em OPER1 é zerada, a saída **não limite** é energizada e a saída **limite** é desenergizada.

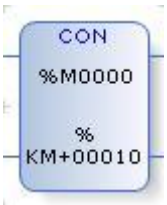
Quando a entrada **ativa** está energizada, cada transição de ligação na entrada **incrementa** aumenta o valor do operando contador (OPER1) de uma unidade.

Se o valor do primeiro operando igualar-se ao do segundo operando, a saída **limite** é energizada. A variável contadora não é incrementada com novas transições na entrada **incrementa**, permanecendo com o valor limite. Se for menor, a saída **limite** é desenergizada. O estado lógico da saída **não limite** é exatamente o oposto da saída **limite**, mesmo estando a instrução desativada.

Em caso de acesso indireto inválido para o segundo operando da instrução, a saída **não limite** é energizada.

ATENÇÃO: Com a entrada **ativa** desativada, a saída **não limite** permanece sempre energizada, mesmo quando a instrução estiver em um trecho comandado pela instrução RM (relé mestre). Deve-se ter cuidado para não realizar acionamentos indesejáveis na lógica devido a este fato.

Exemplo:



Neste caso, quanto um pulso é gerando na entrada incrementa, o valor do contador dentro da memória %M0000 aumenta de uma unidade. Para que isto ocorra a entrada ativa deve estar sempre energizada. Enquanto a contagem não atinge o limite estipulado por %KM+00010, a saída não limite permanece habilitada. Quando a contagem chega ao seu limite, a saída não limite é desabilitada e a saída limite passa a ficar energizada. Para que o contador seja reiniciado é necessário desenergizar a entrada ativa e energizar novamente para uma nova contagem.

Sintaxe:

OPER1	OPER2
%M	%KM %M %M*M

Tabela 3-34. Sintaxe da instrução CON

Tabela Verdade (Válida somente para a PO3x47):

Tabela Verdade - instrução CON							
Entradas					Saídas		
Situação	Inc	Ativa	Oper1 < 0	Oper1 >= Oper2	Operando	Lim	NL
Oper1 inválido	x	x	x	x	Inalterado	0	1
Entrada não ativada	x	0	x	x	0	0	1
Oper2. Inválido ou negativo	x	1	0	x	Inalterado	0	1
Oper2. Inválido ou negativo	x	1	1	x	0	0	1
Sem transição	nT	1	0	0	Oper1	0	1
Sem transição, limite superior	nT	1	0	1	Oper2	1	0
Transição	T	1	0	0	Oper1 + 1	0	1
Transição, limite superior	T	1	0	1	Oper2	1	0

Tabela 3-35. Tabela verdade da instrução CON

Legendas:

- T = transição
- nT = não transição;
- x = não influi
- Inval = inválido
- Lim = limite
- NL = não limite
- Inc = incrementa

Quando algum operando é inválido ou o operando 2 é negativo, não zera saída.

COB - Contador Bidirecional

OPER1 - contador
OPER2 - passo de contagem
OPER3 - limite de contagem

Descrição:

Esta instrução realiza contagens com o valor de incremento ou decremento definido por um operando. A instrução contador bidirecional permite contagens em ambos os sentidos, isto é, incrementa ou decrementa o conteúdo de um operando do tipo memória.

O primeiro operando contém a memória acumuladora do valor contado, enquanto que o segundo especifica o valor do incremento ou decremento desejado. O terceiro operando contém o valor limite da contagem.

A contagem ocorre sempre que a entrada **ativa** está energizada e as entradas **incrementa** ou **decrementa** sofrerem uma transição de desligadas para ligadas. Se ambas as entradas sofrem a transição no mesmo ciclo de varredura do programa, não há incremento nem decremento no valor da memória declarada em OPER1.

Caso o valor do incremento seja negativo, a entrada **incrementa** provoca decrementos e a entrada **decremento** provoca incrementos no valor da contagem.

Se o valor do primeiro operando tornar-se maior ou igual ao do terceiro operando, a saída **limite superior** é energizada, não havendo incremento.

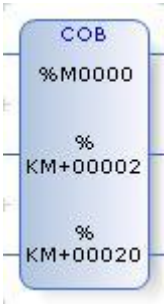
Se o valor do primeiro operando tornar-se igual ou inferior a zero, a saída **limite inferior** é acionada, sendo armazenado zero no primeiro operando.

Se o valor do primeiro operando está entre zero e o limite, a saída **não limite** é acionada. Se a entrada **ativa** não está energizada, a saída **limite inferior** é energizada e o primeiro operando é zerado.

Em caso de acesso indireto inválido para qualquer um dos operandos da instrução, a saída **limite inferior** é energizada.

ATENÇÃO: Com a entrada **ativa** desativada, a saída **limite inferior** permanece sempre energizada, mesmo quando a instrução estiver em um trecho comandado pela instrução RM (relé mestre). Deve-se ter cuidado para não realizar acionamentos indesejáveis na lógica devido a este fato.

Exemplo:



No caso acima o valor da contagem fica armazenado em %M0000. Para habilitar o contador é necessário que a entra ativa esteja energizada. Para incrementar a contagem, basta dar um pulso na entrada incrementa, e para decrementar, faça o mesmo na entrada decrementa. A contagem neste exemplo será de dois em dois, devido ao valor do operando %KM+00002, e o limite é de 20 (%KM00020). Enquanto o contador não atinge os limites, a saída não limite permanece habilitada. Caso o limite superior ou inferior for atingido, suas respectivas saídas são energizadas.

Sintaxe:

OPER1	OPER2	OPER3
	%M	%M
%M	%M*M	%M*M
%M*M	%KM	%KM

Tabela 3-36. Sintaxe da instrução COB

Tabela Verdade (Válida somente para a PO3x47):

Tabela Verdade - instrução COB										
Entradas							Saídas			
Situação	Inc	Dec	Ativa	Oper3 < 0	Oper1 < 0	Oper1 >= Oper3	Operando	Lim+	NL	Lim-
Oper 1. Inválido	x	x	x	x	x	x	inalterado	0	0	1
Entrada 2 não ativada	x	x	0	x	x	x	0	0	0	1
Oper 2 ou Oper 3 inválido	x	x	1	x	1	x	0	0	0	1
Oper 2 ou Oper 3 inválido	x	x	1	x	0	x	inalterado	0	0	1
Oper 3 negativo	x	x	1	1	1	x	0	0	0	1
Oper 3 negativo	x	x	1	1	0	X	Inalterado	0	0	1
Limite inferior	x	x	1	0	0	0	0	0	0	1
Sem transição, limite superior	nT ou T	nT ou T	1	0	0	1	Oper3	1	0	0
Sem transição	nT ou T	nT ou T	1	0	0	0	Oper1	0	1	0
Transição positiva	T	nT	1	0	0	0	Oper1 + Oper2	0	1	0
Transição positiva, limite inferior	T	nT	1	0	1	0	Oper2	0	1	0
Transição positiva, limite superior	T	nT	1	0	0	1	Oper3	1	0	0
Transição negativa	nT	T	1	0	0	0	Oper1 – Oper2	0	1	0
Transição negativa, limite inferior	NT	T	1	0	1	0	Oper2	0	1	0
Transição negativa, limite superior	nT	T	1	0	0	1	Oper3-Oper2	0	1	0

Tabela 3-37. Tabela verdade da instrução COB

Legendas:

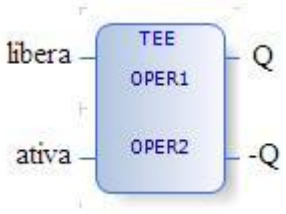
- T = transição
- nT = não transição;

- x = não influi
- Inval = inválido
- Lim = limite
- NL = não limite
- Inc = incrementa

Quando ambas as entradas transicionam, não há incremento ou decremento.

Quando algum operando é inválido ou operando 3 é negativo, não altera a saída.

TEE - Temporizador na Energização



OPER1 - acumulador de tempo
OPER2 - limite de tempo (décimos de segundos)

Descrição:

Esta instrução realiza contagens de tempo com a energização das suas entradas de acionamento.

A instrução TEE possui dois operandos. O primeiro (OPER1) especifica a memória acumuladora da contagem de tempo. O segundo operando (OPER2) indica o tempo máximo a ser acumulado. A contagem de tempo é realizada em décimos de segundos, ou seja, cada unidade incrementada em OPER1 corresponde a 0,1 segundo.

Enquanto as entradas **libera** e **ativa** estiverem simultaneamente energizadas, o operando OPER1 é incrementado a cada décimo de segundo. Quando OPER1 for maior ou igual a OPER2, a saída **Q** é energizada e **-Q** desenergizada, permanecendo OPER1 com o mesmo valor de OPER2.

Desacionando-se a entrada **libera**, há a interrupção na contagem do tempo, permanecendo OPER1 com o mesmo valor. Desacionando-se a entrada **ativa**, o valor em OPER1 é zerado.

Se OPER2 for negativo ou o acesso indireto for inválido, OPER1 é zerado e a saída **-Q** é energizada.

O estado lógico da saída **Q** é exatamente o oposto da saída **-Q**, mesmo estando a instrução desativada.

ATENÇÃO: Com a entrada **ativa** desativada, a saída **-Q** permanece sempre energizada, mesmo quando a instrução estiver em um trecho comandado pela instrução RM (relé mestre). Deve-se ter cuidado para não realizar acionamentos indesejáveis na lógica devido a este fato.

Diagrama de Tempos:

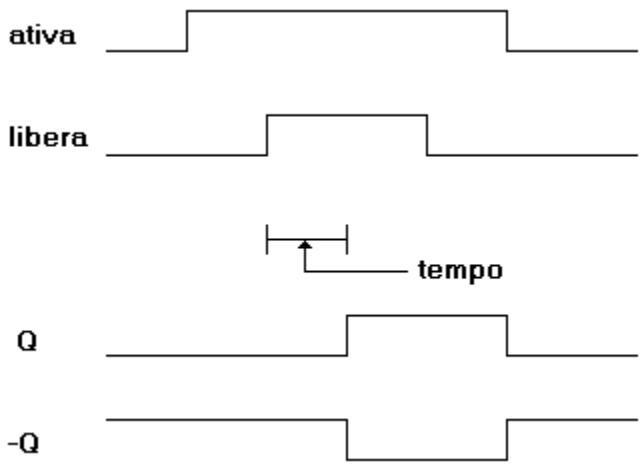


Figura 3-11. Diagrama de tempos da instrução TEE

Exemplo:

No exemplo acima, suponhamos que a entrada ativa esteja ligada, quanto a entrada libera for ativada a saída Q será energizada após 10 (%KM+00100) segundos. Na saída –Q temos o valor da saída Q invertida. No operando %M0000 fica armazenado o valor da contagem de tempo.

Sintaxe:

OPER1	OPER2
%M	%M %M*M %KM

Tabela 3-38. Sintaxe da instrução TEE

Tabela Verdade (Válida somente para a PO3x47):

Tabela Verdade - instrução TEE							
Entradas					Saídas		
Situação	Libera	Ativa	Oper1 < 0	Oper1 >= Oper2	Operando	Q	- Q
Oper1 inválido	x	x	x	x	Inalterado	0	1
Entrada não ativada	x	0	x	x	0	0	1
Oper2. Inválido ou negativo	x	1	0	x	Inalterado	0	1
Oper2. Inválido ou negativo	x	1	1	x	0	0	1
Tempo = T	1	1	x	0	Oper1+T	0	1
Tempo = T, limite superior	1	1	x	1	Oper2	1	0
Tempo < T	1	1	x	0	Oper1	0	1
Tempo < T, limite superior	1	1	x	1	Oper2	1	0
Não liberada	0	1	x	0	Oper1	0	1
Não liberada, limite superior	0	1	x	1	Oper2	1	0

Tabela 3-39. Tabela verdade da instrução TEE

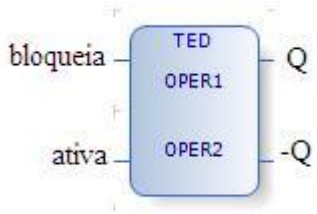
Legendas:

- T = tempo da varredura em unid 0,1s
- x = não influi

Quando Oper1 for negativo ele é tratado como zero.

ATENÇÃO:
Esta instrução tem alguns pontos que necessitam de atenção:
- A instrução TEE não está habilitada para funcionamento em módulos de interrupção externa E020
- Não execute esta instrução em um ciclo
- Executar esta instrução mais de uma vez no mesmo ciclo
- Esta instrução não pode ser saltada

TED - Temporizador na Desenergização



OPER1 - acumulador de tempo
OPER2 - limite de tempo (décimos de segundo)

Descrição:

Esta instrução realiza contagens de tempo com a desenergização da sua entrada de acionamento.

A instrução TED possui dois operandos. O primeiro (OPER1) especifica a memória acumuladora da contagem de tempo. O segundo operando (OPER2) indica o tempo máximo a ser acumulado. A contagem de tempo é realizada em décimos de segundos, ou seja, cada unidade incrementada em OPER1 corresponde a 0,1 segundo.

Enquanto a entrada **ativa** estiver energizada e a entrada **bloqueia** desenergizada, o operando OPER1 é incrementado a cada décimo de segundo. Quando OPER1 for maior ou igual a OPER2, a saída **Q** é desenergizada e - **Q** energizada, permanecendo OPER1 com o mesmo valor de OPER2.

A saída **Q** fica energizada sempre que a entrada **ativa** estiver energizada e OPER1 for menor do que OPER2. Acionando-se a entrada **bloqueia**, há a interrupção na contagem do tempo, enquanto que desacionando a entrada **ativa**, o tempo do acumulador é zerado e a saída **Q** é desacionada.

Se OPER2 for negativo ou o acesso indireto for inválido, OPER1 é zerado e a saída **Q** é energizada. O estado lógico da saída - **Q** é exatamente o oposto da saída **Q**, mesmo estando a instrução desativada.

ATENÇÃO: Com a entrada **ativa** desativada, a saída - **Q** permanece sempre energizada, mesmo quando a instrução estiver em um trecho comandado pela instrução RM (relé mestre). Deve-se ter cuidado para não realizar acionamentos indesejáveis na lógica devido a este fato.

Diagrama de Tempos:

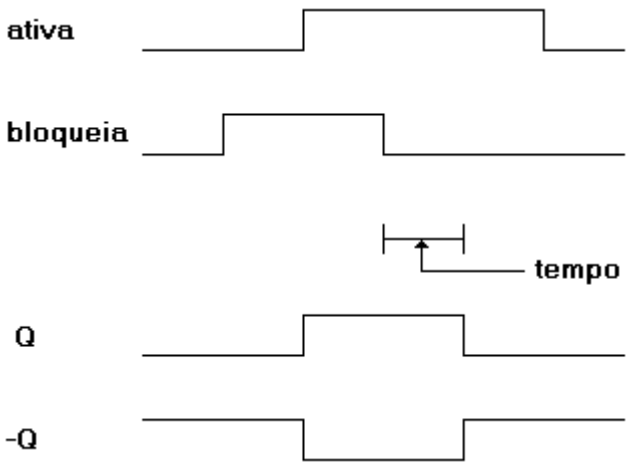
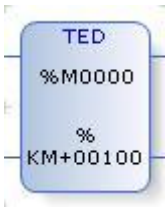


Figura 3-12. Diagrama de tempos da instrução TED

Exemplo:



No exemplo acima, suponhamos que a entrada ativa esteja ligada, quando a entrada bloqueia for desenergizada a saída Q será desenergizada após 10 (%KM+00100) segundos. Na saída –Q temos o valor da saída Q negada. No operando %M0000 fica armazenado o valor da contagem de tempo.

Sintaxe:

OPER1	OPER2
%M	%M %M*M %KM

Tabela 3-40. Sintaxe da instrução TED

Tabela Verdade (Válida somente para a PO3x47):

Tabela Verdade - instrução TED							
Entradas					Saídas		
Situação	Bloqueia	Ativa	Oper1 < 0	Oper1 >= Oper2	Operando	Q	- Q
Oper1 inválido	x	x	x	x	Inalterado	0	1
Entrada não ativada	x	0	x	x	0	0	1
Oper2. Inválido ou negativo	x	1	0	x	Inalterado	1	0
Oper2. Inválido ou negativo	x	1	1	x	0	1	0
Tempo = T	0	1	x	0	Oper1+T	1	0
Tempo = T, limite superior	0	1	x	1	Oper2	0	1
Tempo < T	0	1	x	0	Oper1	1	0
Tempo < T, limite superior	0	1	x	1	Oper2	0	1
Bloqueada	1	1	x	0	Oper1	1	0
Bloqueada, limite superior	1	1	x	1	Oper2	0	1

Tabela 3-41. Tabela verdade da instrução TED

Legendas:

- T = tempo da varredura em unid 0,1s
- x = não influi

Quando Oper1 for negativo ele é tratado como zero.

ATENÇÃO:
Esta instrução tem alguns pontos que necessitam de atenção:
- A instrução TED não está habilitada para funcionamento em módulos de interrupção externa E020
- Não execute esta instrução em um ciclo
- Executar esta instrução mais de uma vez no mesmo ciclo
- Esta instrução não pode ser saltada

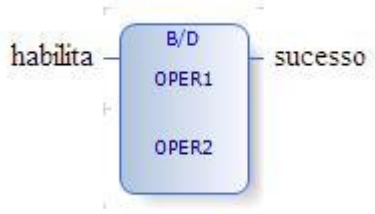
Instruções do Grupo Conversores

Esse grupo possui instruções que permitem a conversão entre os formatos de armazenamento dos valores utilizados nos operandos do programa aplicativo e acessos a módulos analógicos no barramento de entrada e saída.

Nome	Descrição do Nome	Seqüência de Edição
BIN/DEC	Conversão binário-decimal	Alt, I, V, B
DEC/BIN	Conversão decimal - binário	Alt, I, V, D
ANA/DIG	Conversão analógico – decimal	Alt, I, V, A
DIG/ANA	Conversão digital – analógico	Alt, I, V, G

Tabela 3-42. Instruções do grupo conversores

B/D - Conversão Binário-Decimal



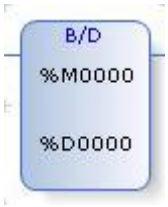
OPER1 - origem
OPER2 - destino

Descrição:

Esta instrução converte valores armazenados em formato binário, contidos em operandos memória (%M), para o formato decimal (BCD), armazenando-os em operandos decimais (%D).

O valor binário contido no primeiro operando (OPER1) é convertido para valor decimal e armazenado no segundo operando (OPER2). A saída **sucesso** é acionada se a conversão for corretamente realizada. Se algum acesso indireto inválido a operando ocorrer, a saída **sucesso** não é energizada.

Exemplo:



Suponhamos que no operando %M0000 temos o valor 10 (em binário 0000 0000 0000 1010). Após a energização da entrada habilita, o valor do operando %D0000 passará a ser 10 (em binário considerando apenas a parte baixa 0000 0000 0001 0000), ou seja, o valor de %M0000 foi convertido para o formato BCD.

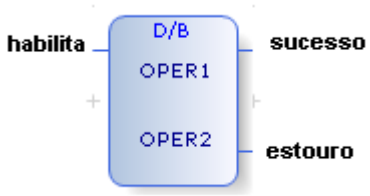
ATENÇÃO: Caso utilizada a instrução MOV para este tipo de operação, poderão haver discrepâncias nos valores dos operandos %D.

Sintaxe:

OPER1	OPER2
%M	%D
%M*M	%M*D

Tabela 3-43. Sintaxe da instrução B/D

D/B - Conversão Decimal-Binário



OPER1 - origem
OPER2 - destino

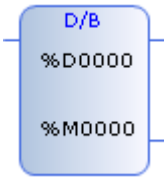
Descrição:

Esta instrução converte valores armazenados em formato decimal, contidos em operandos decimais (%D), para o formato binário, armazenando-os em operandos memórias (%M).

O valor decimal contido no primeiro operando (OPER1) é convertido para valor binário e armazenado no segundo operando (OPER2). A saída **sucesso** é acionada se a conversão for corretamente realizada. Se algum acesso indireto inválido a operando ocorrer, a saída **sucesso** não é energizada.

Caso o valor convertido resultar em um valor maior do que os máximos armazenáveis em operandos %M, a saída **sucesso** não é energizada, sendo armazenado o valor limite no operando destino. Neste caso, a saída **estouro** é energizada.

Exemplo:



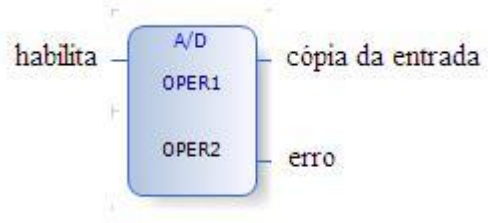
Suponhamos que no operando %D0000 temos o valor 23 (em binário considerando apenas a parte baixa 0000 0000 0010 0011). Após a energização da entrada habilita, o valor do operando %M0000 passará a ser 23 (em binário 0000 0000 0001 0111), ou seja, o valor de %D0000 em DCD foi convertido o formato do operando %M0000.

Sintaxe:

OPER1	OPER2
%D	%M
%M*D	%M*M

Tabela 3-44. Sintaxe da instrução D/B

A/D - Conversão Analógico-Digital



OPER1 - endereço do módulo no barramento / número de canais a converter
OPER2 - primeiro operando a receber o valor convertido

Descrição:

Esta instrução converte os valores lidos de um módulo de entrada analógica para valores numéricos armazenados em operandos.

É possível efetuar a leitura de 1 ou 8 canais alterando-se apenas a especificação do primeiro operando, o qual indica o endereço no barramento ocupado pelo módulo A/D. Este módulo deve estar especificado na declaração do barramento, realizada no MasterTool XE. O endereço a ser programado em OPER1 pode ser obtido diretamente no MasterTool XE. Os valores convertidos são colocados em operandos do tipo memória, definidos em OPER2.

A conversão é realizada apenas se a entrada **habilita** estiver energizada.

Caso OPER1 seja especificado com subdivisão do tipo ponto (%RXXXX.X), a conversão é realizada somente para o canal do módulo relativo ao ponto. Os pontos .0 a .7 do operando correspondem aos canais 0 a 7 do módulo, respectivamente. Neste formato, o tempo de execução da instrução é significativamente menor do que a conversão dos 8 canais, sendo adequada, por exemplo, para uso em módulos de programa E018, acionados por interrupção de tempo.

Se OPER1 for especificado como %RXXXX (conversão de 8 canais), os valores convertidos são colocados na memória declarada em OPER2 e nas 7 memórias subseqüentes.

Se OPER1 for especificado como %RXXXX.X (conversão de 1 canal), o valor já convertido é colocado na memória declarada em OPER2.

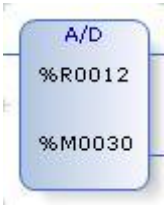
Os módulos disponíveis para realizar a conversão A/D são mostrados a seguir. Os valores convertidos pela instrução pertencem a uma faixa relacionada com a característica de cada módulo:

- AL-1103 (10 bits) : valores de 0000 a 1023
- AL-1116 (12 bits) : valores de 0000 a 4095
- AL-1119 e QK1119 (12 bits): valores de 0000 a 4095
- AL-1139: valores de 0000 a 3999, com indicação de overflow (4000 a 4095)

A saída de erro da instrução é ativada em alguma das seguintes situações:

- Módulo declarado no barramento é inválido para a instrução (não é um dos módulos relacionados anteriormente)
- Tentativa de acesso a operandos não declarados
- Erro de conversão (exceto AL-1103)

Exemplo:



Suponhamos que no registro de operando %R0012 é declarada uma entrada analógica. Estes dados serão todos transferidos para o operando %M0000.

Sintaxe:

OPER1	OPER2
%RXXXX %RXXXX.X	%M

Tabela 3-45. Sintaxe da instrução A/D

D/A - Conversão Digital-Analógico

OPER1 - primeiro operando com os valores a serem convertidos
OPER2 - endereço do módulo no barramento / número de canais a converter

Descrição:

Esta instrução converte os valores numéricos de memórias para sinais analógicos. Os valores são convertidos através dos cartões de saída analógica AL-1203, AL-1214 ou AL-1222 sendo possível a conversão de 1 ou 4 canais utilizando-se apenas uma instrução D/A.

O primeiro operando especifica a primeira memória com o valor a converter.

O segundo operando indica o endereço do módulo D/A no barramento de módulos. O módulo deve estar especificado na declaração do barramento, realizada no MasterTool XE. O endereço a ser programado em OPER2 pode ser obtido diretamente através do MasterTool XE.

A conversão é efetuada apenas se a entrada **habilita** estiver energizada.

Caso OPER2 seja especificado com subdivisão do tipo ponto (%RXXXX.X), a conversão é realizada do operando declarado em OPER1 para o canal do módulo correspondente ao ponto. Os pontos .0 a .3 do operando correspondem aos canais 0 a 3 do módulo, respectivamente.

Se OPER2 for especificado como %RXXXX (conversão de 4 canais), os valores a serem convertidos são obtidos da memória declarada em OPER1 e das 3 subseqüentes.

Os módulos disponíveis para realizar a conversão D/A são mostrados a seguir. Os valores convertidos pela instrução pertencem a uma faixa relacionada com a característica de cada módulo:

Saída em Tensão:

Módulo	Resolução	Normalização	Faixa
AL-1203	10 bits	Não utilizada	0000 a 1000
AL-1214	10 bits	Não utilizada	0000 a 1000
AL-1222	12 bits	Desligada	0000 a 4000
AL-1222	12 bits	Ligada	-2000 a +2000

Tabela 3-46. Instrução D/A - Saída em tensão

Saída em Corrente:

Módulo	Resolução	Normalização	Faixa
AL-1203	10 bits	Não utilizada	0000 a 1000
AL-1214	10 bits	Não utilizada	0000 a 1000
AL-1222	11 bits	Desligada	0000 a 4000

Tabela 3-47. Instrução D/A - Saída em corrente

Os valores convertidos do AL-1222 depende ainda da entrada **normaliza**, que converte valores simétricos quando energizada. Isto torna-se útil quando se necessita trabalhar com valores negativos, por exemplo na faixa de tensão de ±10V.

Não existe normalização para os módulos AL-1203 e AL-1214, apenas para o AL-1222. Entretanto, a normalização somente é possível para operação em modo tensão.

ATENÇÃO:
Em modo corrente a entrada **normaliza** não deve ser energizada.

Os AL-1222 podem trabalhar com as 4 saídas em modo tensão ou modo corrente, ou os dois modos simultaneamente. A seleção do modo de operação é feita pelo usuário através da programação do endereçamento do módulo em OPER2:

- Se %RXXXX for par, converte corrente
- Se %RXXXX for ímpar, converte tensão

Exemplo:

Caso o módulo seja colocado no endereço %R0024 do barramento, se a instrução for programada com %R0024, os AL-1222 irão operar em modo corrente. Se for programada com %R0025, irá operar em modo tensão.

ATENÇÃO:
A instrução não pode ser saltada durante a execução do programa aplicativo sob pena dos valores amostrados estarem incorretos.

A saída de erro da instrução é ativada em alguma das seguintes situações:

- Módulo declarado no barramento é inválido para a instrução (não é um dos módulos relacionados anteriormente)
- Tentativa de acesso a operandos não declarados

Sintaxe:

OPER1	OPER2
%M	%RXXXX RXXXX.X

Tabela 3-48. Sintaxe da instrução D/A

Instruções do Grupo Geral

As instruções do grupo gerais permitem teste e acionamentos de pontos indiretamente, implementações de máquinas de estado, chamadas de procedimentos e funções e escrita e leitura de operandos na rede ALNET II.

Nome	Descrição do Nome	Seqüência de Edição
LDI	Liga ou desliga pontos indexados	Alt, I, G, L
TEI	Teste de estado de pontos indexados	Alt, I, G, T
SEQ	Seqüenciador	Alt, I, G, S
CHP	Chama modulo procedimento	Alt, I, G, P
CHF	Chama modulo função	Alt, I, G, F
ECR	Escrita de operandos em outro CP	Alt, I, G, E
LTR	Leitura de operandos de outro CP	Alt, I, G, T, T
LAI	Libera atualização de imagens	Alt, I, G, I
ECH	Escrita de Operandos em Outro CP para Ethertnet	Alt, I, G, E
LTH	Leitura de Operandos em Outro CP para Ethertnet	Alt, I, G,

Tabela 3-49. Instruções do grupo geral

LDI - Liga/Desliga Indexado

OPER1 - endereço do ponto a ser ligado ou desligado
OPER2 - endereço limite inferior
OPER3 - endereço limite superior

Descrição:

Esta instrução é utilizada para ligar ou desligar pontos indexados por uma memória, delimitados por operandos de limite inferior e superior.

O primeiro operando especifica a memória cujo conteúdo referencia o operando auxiliar, entrada ou saída a ser ligado ou desligado. Deve ser declarado como operando de acesso indireto a operando %E ou %A (%MXXXX*E ou %MXXXX*A). Mesmo quando a instrução for utilizada para ligar ou desligar pontos de saída (%S), a representação deste operando será como acesso indireto a entrada (%MXXXX*E).

O segundo operando especifica o endereço do primeiro relé de saída ou auxiliar válido na instrução. Deve ser especificado com subdivisão de ponto (%RXXXX.X, %SXXXX.X ou %AXXXX.X).

O terceiro operando especifica o endereço do último relé de saída ou auxiliar válido na instrução. Deve ser especificado com subdivisão de ponto (%EXXXX.X, %SXXXX.X ou %AXXXX.X).

Se as entradas **liga** ou **desliga** forem acionadas, o ponto especificado pelo valor contido no operando memória (OPER1) é ligado ou desligado se estiver dentro da área de endereços limitada por OPER2 e OPER3. Por exemplo, se estes operandos correspondem a %S0003.3 e %S0004.5, respectivamente, esta instrução só atua para os elementos de %S0003.3 a %S0003.7 e de %S0004.0 a %S0004.5.

Se o relé ou auxiliar apontado pela memória índice estiver fora dos limites definidos pelos parâmetros da segunda e terceira células, a saída **índice superior inválido** ou **índice inferior inválido** é ligada. A saída da primeira célula é acionada se qualquer uma das entradas **liga** ou **desliga** é energizada e o acesso for corretamente realizado.

Caso as entradas permaneçam desacionadas, todas as saídas da instrução permanecem desenergizadas.

Se ambas as entradas forem energizadas simultaneamente, nenhuma operação é realizada, e todas as saídas da instrução são desenergizadas.

Em OPER1 deve ser carregado um valor que especifique o ponto desejado para ligar ou desligar, de acordo com a seguinte fórmula:

VALOR OPER1 = (OCTETO * 8) + PONTO

Exemplo:

Por exemplo, se S0010.5 é o ponto que se deseja ligar indiretamente, então:

- OCTETO = 10
- PONTO = 5
- VALOR OPER1 = (10 * 8) + 5 = 85

O valor a ser carregado em OPER1 é 85.

ATENÇÃO:
Esta instrução permite ligar ou desligar indiretamente pontos de operandos %E, sobrepondo o valor da varredura dos módulos de entrada após a sua execução.

Sintaxe:

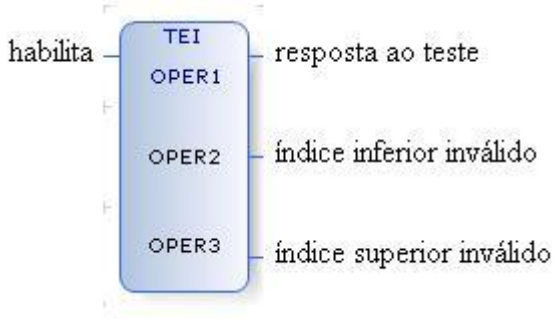
OPER1	OPER2	OPER3
%M*E	%EXXXX.X	%EXXXX.X

OPER1	OPER2	OPER3
%M*S	%SXXXX.X	%SXXXX.X

OPER1	OPER2	OPER3
%M*A	%AXXXX.X	%AXXXX.X

Tabela 3-50. Sintaxe da instrução LDI

TEI - Teste de Estado Indexado



OPER1 - endereço do ponto a ser testado
OPER2 - endereço limite inferior
OPER3 - endereço limite superior

Descrição:

Esta instrução é utilizada para testar o estado de pontos indexados por uma memória, delimitados por operandos de limite inferior e superior.

O primeiro operando especifica a memória cujo conteúdo referencia o operando auxiliar ou relé de saída a ser testado. Deve ser declarado como operando de acesso indireto a operando %E ou %A (%MXXXX*E ou %MXXXX*A). Mesmo quando a instrução for utilizada para testar pontos de saída (%S), a representação deste operando será como acesso indireto a entrada (%MXXXX*E).

O segundo operando especifica o endereço do primeiro relé de saída ou auxiliar válido na instrução. Deve ser especificado com subdivisão de ponto (%EXXXX.X, %SXXXX.X ou %AXXXX.X).

O terceiro operando especifica o endereço do último relé de saída ou auxiliar válido na instrução. Deve ser especificado com subdivisão de ponto (%EXXXX.X, %SXXXX.X ou %AXXXX.X).

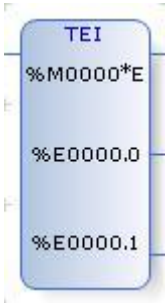
Se a entrada **habilita** estiver energizada, o estado do relé ou auxiliar especificado pelo valor contido na memória índice (OPER1) é examinado. Conforme esteja em 1 ou 0, a saída **resposta** é ligada ou não.

O ponto indexado pela memória é testado se estiver dentro da área de endereços limitada por OPER2 e OPER3. Por exemplo, se estes operandos correspondem a %S0003.3 e %S0004.5, respectivamente, esta instrução só atua para os elementos de %S0003.3 a %S0003.7 e de %S0004.0 a %S0004.5.

Se o relé ou auxiliar apontado pela memória índice estiver fora dos limites definidos pelos parâmetros da segunda e terceira células, a saída **índice superior inválido** ou **índice inferior inválido** é ligada e a saída da primeira célula desligada. Esta verificação é feita somente no momento em que a entrada **habilita** é energizada.

O cálculo do valor a ser armazenado no primeiro operando, para referência do ponto desejado, é o mesmo especificado na instrução **LDI**.

Exemplo:



Suponhamos que o operando %M0000 possua o valor 1, assim será feito o teste se o operando %E0000.1 está ativo ou não. Se %M0000 possuir o valor 0, o operando testado será o %E0000.0.

Sintaxe:

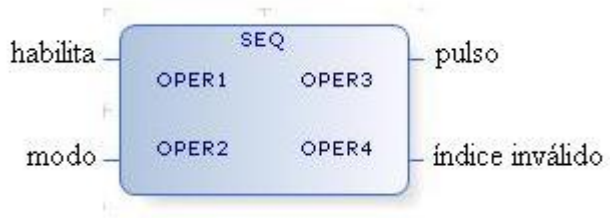
OPER1	OPER2	OPER3
%M*E	%EXXXX.X	%EXXXX.X

OPER1	OPER2	OPER3
%M*S	%SXXXX.X	%SXXXX.X

OPER1	OPER2	OPER3
%M*A	%AXXXX.X	%AXXXX.X

Tabela 3-51. Sintaxe da instrução TEI

SEQ - Seqüenciador



OPER1 - tabela de condições ou primeira tabela de estados
OPER2 - índice da(s) tabela(s) (estado atual)
OPER3 - operando base da primeira série de condições
OPER4 - operando base da segunda série de condições

Descrição:

Esta instrução permite a programação de seqüenciamentos complexos com condições de evolução específicas para cada estado. Sua forma de programação é semelhante a "máquinas de estado".

A instrução pode ser executada em dois modos: o modo 1000 e o modo 3000. Quando a entrada **modo** está desenergizada, a instrução é executada no modo 1000, e quando ela está energizada, a instrução é executada no modo 3000. No modo 3000 seqüenciamentos mais complexos podem ser programados.

Modo 1000:

Neste modo ocorre uma seqüência de evolução fixa dos estados. A evolução sempre ocorre do estado atual para o seguinte, e do último para o primeiro.

O primeiro operando especifica uma tabela onde cada posição contém o endereço de um ponto de operando auxiliar que é testado como condição de evolução para o próximo estado.

O segundo operando especifica uma memória que armazena o estado atual e serve de índice para a tabela especificada no primeiro operando.

O terceiro operando é irrelevante, porém deve ser especificado um operando do tipo memória ou auxiliar nesta célula, pois o MasterTool XE realiza a consistência conforme o modo 3000.

O quarto operando é irrelevante, porém deve ser especificado um operando do tipo memória ou auxiliar nesta célula, pois o MasterTool XE realiza a consistência conforme o modo 3000.

Quando a entrada **habilita** está desenergizada, as saídas **pulso** e **índice inválido** ficam desenergizadas, independente de qualquer outra condição. Quando a entrada **habilita** estiver energizada, a saída de **pulso** fica normalmente energizada, e a saída de **índice inválido** fica normalmente desenergizada.

Além disso, quando a entrada **habilita** está energizada, a posição da tabela (OPER1) indexada pelo estado atual (OPER2) é acessada e o ponto de operando auxiliar referenciado nesta posição da tabela é examinado. Se este ponto estiver energizado, o conteúdo de OPER2 é incrementado (ou zerado, se estiver apontando para a última posição da tabela OPER1) e na saída **pulso** ocorre um pulso de desenergização com duração de um ciclo de programa. Se o ponto examinado estiver desenergizado nada ocorre e o valor da memória em OPER2 permanece inalterado.

A saída **índice inválido** é ativada se a memória OPER2 (estado atual) contiver um valor que indexa uma posição não existente na tabela especificada em OPER1. Isto pode ocorrer modificando-se a memória OPER2 em um ponto do programa aplicativo fora da instrução SEQ (na inicialização de OPER2, por exemplo). Deve-se ter o cuidado de definir e inicializar a tabela especificada em OPER1 com valores legais.

Na tabela especificada em OPER1 devem ser carregados valores em formato decimal que especifiquem pontos de operandos auxiliares que devem ser testados como condições de evolução. O cálculo destes valores é especificado pela equação:

VALOR = (endereço do operando * 8) + endereço da subdivisão

Exemplo:

Se %A0030.2 é o ponto que se deseja usar como condição de evolução a partir do estado 4, então:

- Endereço do operando = 30
- Endereço da subdivisão = 2
- VALOR = (30 * 8) + 2 = 242

O valor a ser carregado na posição 4 da tabela OPER1 deve ser 242 para que o ponto %A0030.2 causa a evolução para o próximo estado, que é o estado 5 (ou o estado 0, se a tabela tiver 5 posições).

Modo 3000:

Neste modo é possível definir a seqüência de evolução e escolher um entre dois caminhos a partir do estado atual. Portanto, 2 graus de liberdade a mais são oferecidos em relação ao modo 1000, permitindo-se implementar máquinas de estado bem mais complexas.

Existe, entretanto, menos liberdade para escolher as condições de evolução em relação ao modo 1000, além de ser necessário o uso de mais memória (tabelas) no modo 3000.

O primeiro operando especifica a primeira de duas tabelas subseqüentes que são utilizadas pela instrução. As duas tabelas devem ter o mesmo tamanho. Cada posição da primeira tabela contém o próximo estado caso a condição associada ao operando 3 esteja energizada. Cada posição da segunda tabela contém o próximo estado caso a condição associada ao operando 4 esteja energizada.

O segundo operando especifica uma memória que indica qual o estado atual e serve de índice para as tabelas especificadas no primeiro operando.

O terceiro operando especifica um operando que serve de base para determinar a condição de evolução a partir do estado OPER2 para o estado indexado por OPER2 na primeira tabela.

O quarto operando especifica um operando que serve de base para determinar a condição de evolução a partir do estado OPER2 para o estado indexado por OPER2 na segunda tabela.

Quando a entrada **habilita** está desenergizada, as saídas **pulso** e **índice inválido** ficam desenergizadas, independente de qualquer outra condição. Quando a entrada **habilita** está energizada, a saída de **pulso** fica normalmente energizada, e a saída de **índice inválido** fica normalmente desenergizada.

Além disso, quando a entrada **habilita** está energizada, a instrução busca o valor da memória OPER2 (estado atual) e testa a respectiva condição de evolução com base em OPER3. Se esta condição estiver energizada, o operando OPER2 é carregado com um novo estado, indexado pelo próprio operando OPER2 na primeira tabela especificada por OPER1. Caso a condição de evolução associada a OPER2 e com base em OPER3 estiver desenergizada, testa-se a condição de evolução associada a OPER2 e com base em OPER4. Se esta última condição estiver energizada, o operando OPER2 é carregado com um novo estado, indexado pelo próprio operando OPER2 na segunda tabela especificada por OPER1. Se pelo menos uma das 2 condições acima estiver energizada, uma transição de estado ocorrerá, e um pulso de desenergização com duração de um ciclo de programa aplicativo ocorrerá na saída **pulso** da instrução. Se nenhuma das 2 condições estiver energizada, nada acontece, e o valor da memória OPER2 (estado atual) permanece inalterado, bem como a saída **pulso** continua energizada.

A saída **índice inválido** é ativada se a memória OPER2 contiver um valor que indexa uma posição não existente nas tabelas especificadas em OPER1. Isto pode ocorrer modificando-se a memória OPER2 em um ponto do programa aplicativo fora da instrução SEQ (na inicialização de OPER2, por exemplo) ou dentro da própria instrução SEQ, caso algumas das posições das tabelas especificadas em OPER1 contenham valores inválidos para serem o próximo estado. Deve-se ter o cuidado de definir as 2 tabelas especificadas por OPER1 com o mesmo tamanho, e deve-se inicializá-las com valores legais (exemplo: se as tabelas tiverem 10 posições, somente valores entre 0 e 9 devem ser carregados em posições desta tabela, pois somente estes podem ser estados legais).

As condições de evolução associadas ao estado atual (OPER2) são determinadas com base em OPER3 (próximo estado é carregado a partir da primeira tabela) ou com base em OPER4 (próximo estado é carregado a partir da segunda tabela). Sabendo-se que os operandos OPER3 e OPER4 são do tipo memória (16 bits) ou do tipo auxiliar (8 bits), suponha-se o seguinte:

- ESTADO = conteúdo do operando OPER2 (estado atual)
- END3 = endereço de OPER3
- END4 = endereço de OPER4
- END1 = endereço do ponto a ser testado, com base em OPER3
- SUB1 = subdivisão do ponto a ser testado, com base em OPER3
- END2 = endereço do ponto a ser testado, com base em OPER4
- SUB2 = subdivisão do ponto a ser testado, com base em OPER4

Os pontos testados como condição de evolução associada a cada tabela serão:

- M<END1>.<SUB1> ou A<END1>.<SUB1> (primeira tabela)
- M<END2>.<SUB2> ou A<END2>.<SUB2> (segunda tabela)

onde:

- END1 = END3 + ESTADO / 16 (se operando %M)
- END1 = END3 + ESTADO / 8 (se operando %A)
- SUB1 = RESTO (ESTADO / 16) (se operando %M)
- SUB1 = RESTO (ESTADO / 8) (se operando %A)
- END2 = END4 + ESTADO / 16 (se operando %M)
- END2 = END4 + ESTADO / 8 (se operando %A)
- SUB2 = RESTO (ESTADO / 16) (se operando %M)
- SUB2 = RESTO (ESTADO / 8) (se operando %A)

Exemplo:

Sejam:

- OPER1 = %TM000
- OPER2 = %M0010
- OPER3 = %M0100
- OPER4 = %A0020

Onde:

%TM0000	Posição	Valor
	000	00001
	001	00002
	002	00004
	003	00001
	004	00000

%TM0001	Posição	Valor
	000	00001
	001	00003
	002	00001
	003	00004
	004	00000

%M0010 = 00001

%M0100	XXXXX
%M0101	XXXXX
%M0102	XXXXX
%M...	...

%A0020	XXXXX
%A0021	XXXXX
%A0022	XXXXX
%A...	...

Tabela 3-52. Operandos e valores utilizados no exemplo

Então as condições de evolução a partir do estado 1 serão:

Para a primeira tabela:

- 100 + 1/16 = 100
- Resto(1/16) = 1
- Ponto a ser testado = %M0100.1

Para a segunda tabela:

- 20 + 1/8 = 20
- Resto(1/8) = 1
- Ponto a ser testado = %A0020.1

Baseado nas condições de %M0100.1 e %A0020.1 teremos, a partir de uma das tabelas, o novo estado do operando %M0010:

%M0100.1	%A0020.1	%M0010	Observação
0	0	00001	Não há mudança de estado
0	1	00003	Mudança de estado conforme %TM001
1	0	00002	Mudança de estado conforme %TM000
1	1	00002	Mudança de estado conforme %TM000 (OPER3 tem prioridade sobre OPER4)

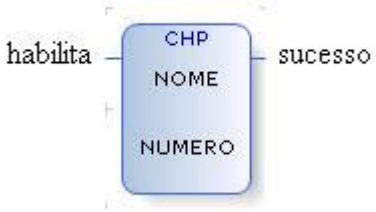
Tabela 3-53. Estados encontrados

Sintaxe:

OPER1	OPER2	OPER3	OPER4
%TM	%M	%M	%M
%M*TM	%M*M	%A	%A

Tabela 3-54. Sintaxe da instrução SEQ

CHP - Chama Módulo Procedimento



OPER1 - nome do módulo a chamar
OPER2 - número do módulo a chamar

Descrição:

Esta instrução realiza o desvio do processamento do módulo corrente para o módulo Procedimento especificado, se o mesmo estiver presente no CP. Ao final da execução do módulo chamado, o processamento retorna para a instrução seguinte à CHP. Não há passagem de parâmetros para o módulo chamado.

A instrução ao ser inserida, ou editada, abrirá uma lista dos módulos procedimentos contidos no projeto, como ilustra a figura a seguir:

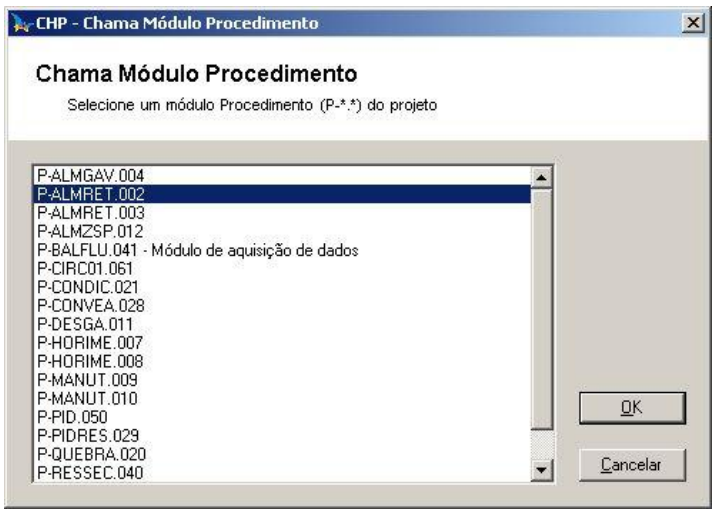


Figura 3-13. Lista de seleção do módulo procedimento chamado pela CHP

Para selecionar um módulo, basta efetuar um duplo clique em cima do módulo desejado ou então selecionar na lista e clicar no botão OK.

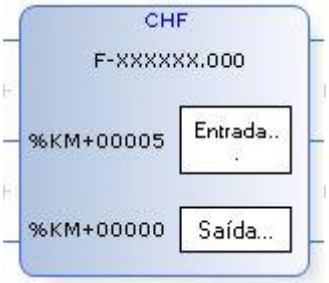
Caso o módulo chamado não exista, a saída **sucesso** é desenergizada e a execução continua normalmente após a instrução.

ATENÇÃO: O nome do módulo não é considerado pelo CP para a chamada, mas apenas o seu número. Caso exista o módulo P com o mesmo número do módulo chamado, porém com nome diferente, este módulo mesmo assim é executado. Porém o MasterTool Extended Edition vai acusar erro na verificação de projeto, impossibilitando de enviar para o CP.

Exemplo:

Ao ser energizada a entrada da instrução CHP, será chamado o Módulo procedimento P-Forno.003.

CHF - Chama Módulo Função



F-XXXXXX.000 - nome do módulo função a chamar
OPER1 - número de parâmetros a enviar
OPER2 - número de parâmetros a retornar
Entrada... - lista dos parâmetros a enviar
Saída... - lista dos parâmetros a retornar

Descrição:

A instrução CHF chama um módulo função que realiza o desvio do processamento do módulo corrente para o módulo especificado na mesma, se este estiver presente no CP. Ao final da execução do módulo chamado, o processamento retorna para a instrução seguinte à CHF. A instrução CHF possui a seguinte tela para edição:

Figura 3-14. Tela de preenchimento da instrução CHF

Deve ser selecionado um módulo função, presente no projeto corrente ou um Módulo Função fornecido pela Altus e instalado juntamente com o MasterTool Extended Edition. Esta seleção, que deve ser feita no momento da inserção ou edição da instrução CHF, clicando no botão que aparece o nome do módulo função que será chamado. Assim, será aberto a seguinte janela:

Figura 3-15. Lista de seleção do módulo função chamado pela CHF

Para selecionar um módulo função, basta efetuar um duplo clique em cima do módulo desejado ou então selecionar na lista e clicar no botão OK. Caso tenha sido selecionado um módulo função instalado no MToolXE, mas que não pertence ao projeto, será perguntado se deseja inserir o módulo no projeto corrente. Deve ser respondido Sim para esta pergunta, pois do contrário o módulo não será selecionado.

Durante a execução da instrução no CP, caso o módulo chamado não exista na memória do CP, a saída **sucesso** é desenergizada e a execução continua normalmente após a instrução.

ATENÇÃO: O nome do módulo não é considerado pelo CP para a chamada, mas apenas o seu número. Caso exista o módulo F com o mesmo número do módulo chamado, porém com nome diferente, este módulo mesmo assim é executado. Porém o MasterTool Extended Edition vai acusar erro na verificação de projeto, impossibilitando de enviar para o CP.

Diferentemente da instrução CHP, a CHF permite a passagem de parâmetros para o módulo função chamado. Isto pode ser feito de duas maneiras: um módulo função distribuído pela Altus, ou um módulo função feito pelo usuário.

Se for um módulo distribuído pela Altus e que tenha uma forma padronizada de preenchimento, não será necessário definir a quantidade de parâmetros de entrada e saída. Neste caso, ao clicar nos botões **Entrada...** ou **Saída...** será aberta uma tela correspondente ao módulo chamado, para edição dos parâmetros, como mostra o exemplo a seguir:

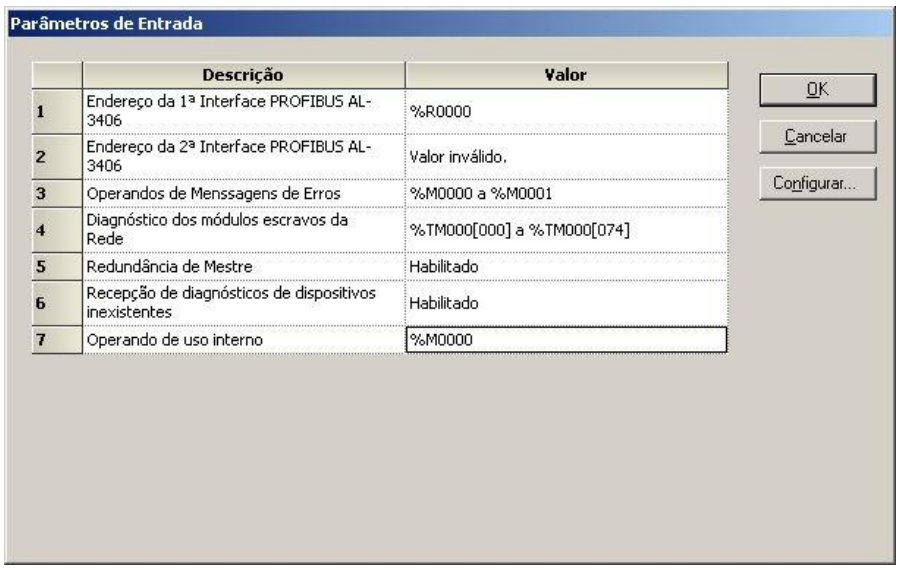


Figura 3-16. Edição de parâmetros de um módulo função padronizado

Neste modo, na edição dos parâmetros é informado o que é permitido e por que não é, através de uma interface amigável.

Além disso, alguns dos módulos fornecidos pela Altus também podem possuir mais de um modo de configuração, onde cada modo possui uma quantidade diferente de parâmetros de entrada. Ao selecionar o novo modo o usuário deverá refazer a configuração do mesmo. Caso o novo módulo tenha menos operandos que o anterior, os operandos a mais serão descartados. Caso o novo módulo tenha mais operandos que o anterior, os operando a mais serão configurados com o operando padrão para o módulo. Será gerado um erro de verificação se um dos parâmetros for de tipo diferente no novo modo.

Alguns módulos função possuem uma configuração personalizada que é acessada através do botão **Configuração Especial...**. Os módulos que habilitam esta função são apresentados nas próximas seções.

Agora, no caso do módulo Função selecionado ter sido criado pelo usuário, ou não haver uma chamada padrão para o módulo, será parametrizado de maneira diferente. Primeiramente deve definir a quantidade de parâmetros de entrada e saída e posteriormente clicar no botão **Entrada...** ou **Saída...** conforme a necessidade. Neste caso será aberto a seguinte janela para a edição dos parâmetros:

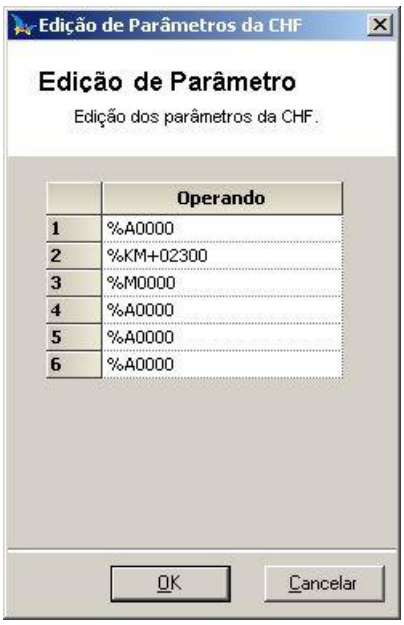


Figura 3-17. Edição de parâmetros de um módulo função não padronizado

ATENÇÃO: O MasterTool XE não realiza nenhuma consistência em relação aos operandos programados como parâmetros, tanto na instrução CHF como no módulo Função.

A lista de operandos a serem enviados para o módulo F deve conter o mesmo número de operandos com o mesmo tipo dos declarados como parâmetros de entrada do módulo, para que a cópia dos seus valores seja realizada corretamente. A cópia dos operandos é realizada na mesma ordem em que os mesmos estão dispostos nas listas. Caso uma das duas listas possua menos operandos em relação à outra, os valores dos operandos excedentes não serão copiados. Se os operandos possuírem tipos diferentes, a cópia dos valores será realizada com as mesmas regras usadas na instrução MOV (movimentação simples de operandos). Este princípio é válido também para as listas de parâmetros de retorno do módulo Função.

A passagem de parâmetros é realizada com a cópia dos valores dos operandos declarados (passagem de parâmetros por valor), embora esses operandos ainda continuem de uso global, utilizáveis por qualquer módulo presente no CP. Os módulos F podem ser programados de forma genérica, para serem reutilizados em diversos programas aplicativos como novas instruções. É aconselhável que os mesmos empreguem operandos próprios, não usados por nenhum outro módulo presente no programa aplicativo, evitando alterações inadvertidas em operandos utilizados em outros módulos.

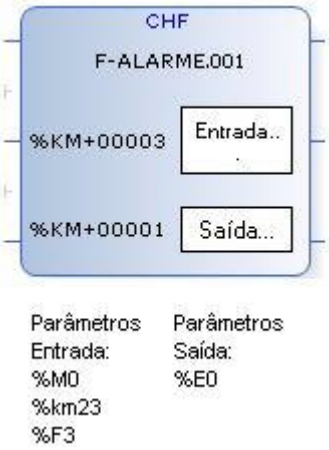
É possível a passagem de operandos simples e constantes para o módulo Função. Não é permitida a passagem de operandos com subdivisão para o módulo Função, tais como %M004.2, %A0021n1, etc. Devem ser usados somente operandos simples para esta passagem. A quantidade máxima de parâmetros de entrada, ou parâmetros de saída é igual a 10.

Será permitido utilizar os seguintes operandos como parâmetros:

Operandos Permitidos como Parâmetros	
%TM	%M*M
%TD	%M*D
%TF	%M*I
%TI	%M*F
%M	%M*A
%D	%M*TM
%F	%M*TD
%I	%M*TI
%E	%M*TF
%S	%KM
%A	%KD
%R	%KF
	%KI

Tabela 3-55. Operandos permitidos como parâmetros

Exemplo:



A CHF chamará o Módulo Função F-Alarme.001, possuindo 3 parâmetros de entrada (%M0000, %KM0023 e %F0003) e 1 parâmetro de saída (%E0000).

CHF - Chama Módulo Função – Configuração Especial F-PID16.056

A instrução CHF possui uma interface de configuração especializada para o módulo F-PID16.056. Esta interface, também chamada de skin PID, permite o ajuste e configuração do laço PID. A janela de configuração contêm três abas:

- Ajuste e Gráfico (Gráfico Tendência): monitorar as variáveis e configurar os parâmetros básicos do PID
- Configurações: permite configurar itens avançados do PID
- Operandos: permite configurar os operandos utilizados pelo módulo função F-PID16.056

ATENÇÃO: Para o correto entendimento do skin PID é necessário ler a documentação do módulo F-PID16.056.

Aba: Ajuste e Gráfico

A aba Ajuste e Gráfico (Gráfico de Tendência) é utilizada para parametrizar o PID permitindo inclusive monitorar as principais variáveis em um gráfico de tempo real.

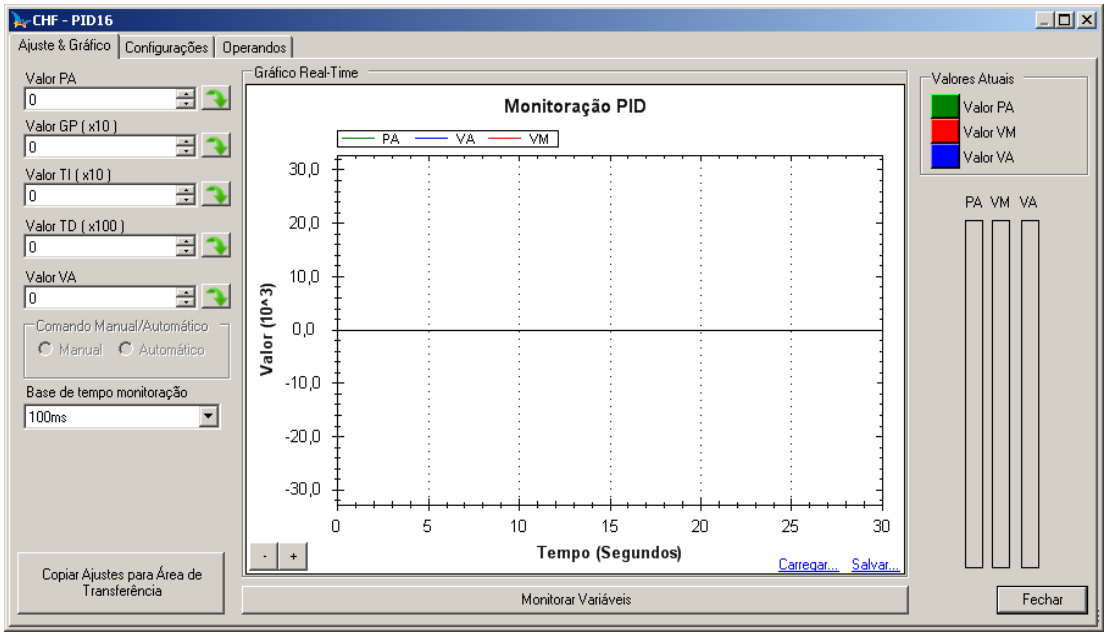


Figura 3-18. Janela skin PID para módulo F-PID16.056, aba Ajuste e Gráfico

Na área à esquerda desta tela é possível configurar as seguintes variáveis:

- PA – ponto de ajuste
- GP – ganho proporcional
- TI – termo integral
- TD – termo derivativo
- VA – variável de atuação
- Modo Automático ou Manual
- Modo Direto ou Reverso

Para configuração dos valores de PA, GP, TI, TD e VA o usuário deve digitar o novo valor dentro da caixa de texto. Quando isto acontecer, será parada a monitoração da respectiva variável e a mesma ficará na cor vermelho, aguardando que o usuário confirme o envio do valor para o CP através do botão que fica à direta da caixa de texto. Se o usuário fechar a janela, os valores editados não são enviados, são descartados.

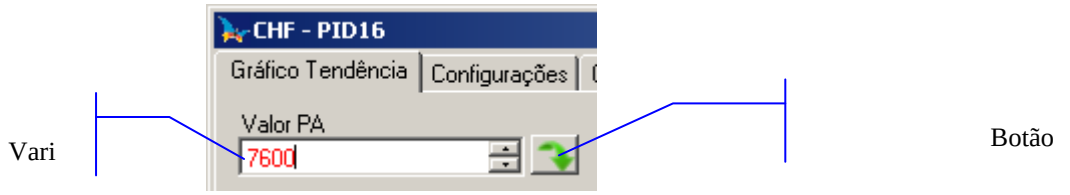


Figura 3-19. Edição de uma variável do PID

Os valores atualizados nesta janela são enviados para o CP apenas. Se o laço PID possuir instruções de ladder configurando as variáveis, os mesmos não são atualizados automaticamente devendo o usuário configurar esta instrução manualmente. Para facilitar esta ação o botão **Copiar valores de inicialização** copia os valores para utilização da CAB de iniciação do operando tabela de controle.

Para configurar os modos Automático, Manual, Direto e Reverso basta clicar que o comando será enviado para o CP.

ATENÇÃO:
Os modos Automático/Manual e Direto/Reverso só estão habilitados para o módulo F-PID16.056 na versão 1.10 ou superior; e a CHF estiver configurada para 7 parâmetros. Para maiores detalhes sobre a versão 1.10 e o sétimo parâmetro ver a documentação do módulo F-PID16.056 no help dos módulos função.

ATENÇÃO:
A configuração dos valores não altera o programa ladder do usuário. Se o usuário estiver configurado os valores através de instruções ladder, estas instruções não serão alteradas.

Na área central é possível visualizar os valores de PA, VM e VA no gráfico de tendência tempo real. Para iniciar a monitoração é necessário pressionar o botão **Monitorar Variáveis**. No mesmo botão é possível parar a monitoração. O gráfico pode ser manipulado utilizando o mouse e o teclado, conforme é apresentado na tabela a seguir:

Comando	Ação
Botão esquerdo	Zoom na área selecionada
Botão esquerdo + Tecla SHIFT ou Botão do meio	Movimenta o gráfico horizontalmente ou verticalmente
Botão direto	Menu de contexto, com as opções: • Un-Zoom: desfaz o último zoom • Undo All Zoom Pan: retorna o gráfico aos valores iniciais de zoom e eixos.

Tabela 3-56. Comandos do mouse e teclado para gráfico de tendência

Também é possível salvar os valores monitorados nos últimos 5 minutos, pressionando o botão **Salvar.** Para carregar os valores é necessário pressionar o botão **Carregar** e depois selecionar o arquivo**.**

Na área à direta é possível monitorar os valores de PA, VM e VA**.** Como também acompanhar estes mesmos valores em escala percentual através dos gráficos de barra vertical. A escala dos gráficos de barra é de 0 a 30000.

Aba: Configurações

A aba Configurações é utilizada para configurar os parâmetros avançados do PID.

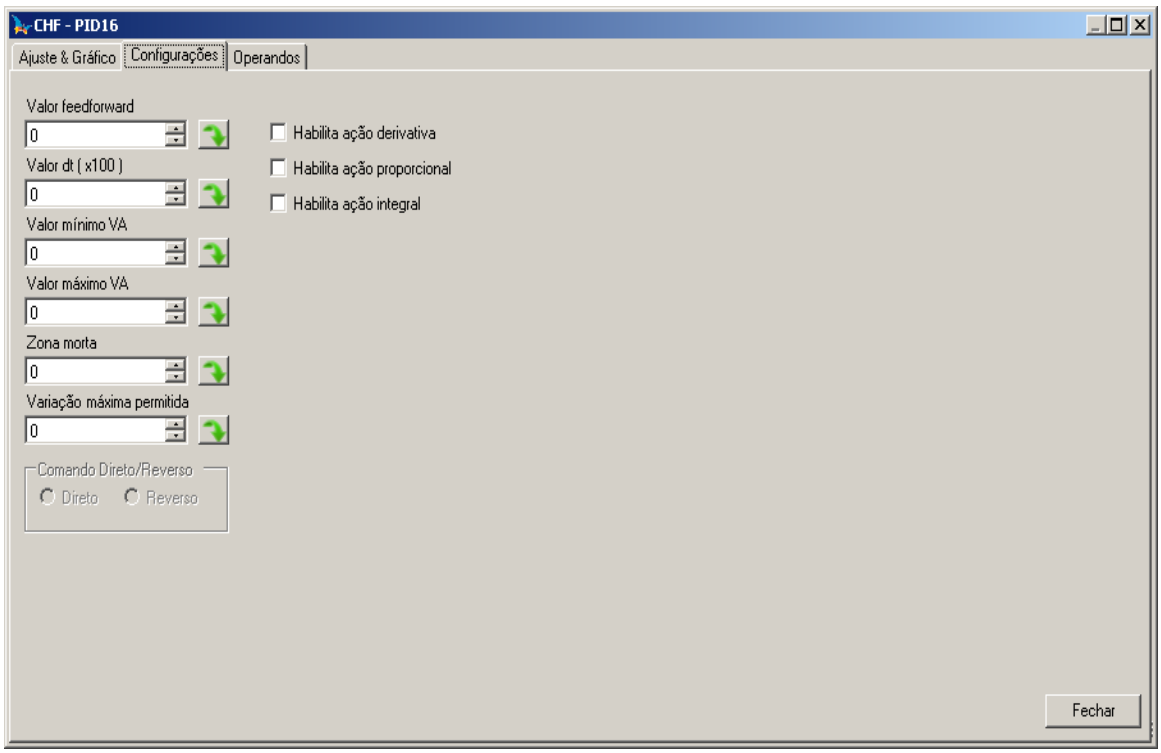


Figura 3-20. Janela módulo F-PID16.056, aba Configurações

Os valores que podem ser configurados são:

- Valor feedforward/bias
- Valor dt
- Valor mínimo VA
- Valor Máximo VA
- Zona morta
- Variação máxima da entrada
- Habilita ação derivativa
- Habilita ação proporcional
- Habilita ação integral
- Comando direto e reverso

Para configuração dos valores o usuário deve digitar o novo valor dentro da caixa de texto. Quando isto acontecer, será parada a monitoração da respectiva variável e a mesma ficará na cor vermelho, aguardando que o usuário confirme o envio do valor para o CP através do botão que fica à direta da caixa de texto. Se o usuário fechar a janela, os valores editados, mas não enviados são descartados. Os comandos de habilitação de ação são enviados imediatamente.

Aba: Operandos

Na aba Operandos é possível configurar os operandos utilizados pelo módulo F-PID16.056. As alterações nesta janela alteram os parâmetros carregados no módulo F-PID16.056. Esta janela edita os seguintes parâmetros de entrada da CHF para o módulo F-PID16.056. A configuração dos operando só é habilitada quando a monitoração está parada.

Parâmetro de Entrada CHF	Nome	Tipo de operando aceito
1	Operando de controle	%TM
2	Operando VM	%M
3	Operando PA	%M
4	Operando VA	%M
5	Operando feedvorward/bias	%M
6	Octeto de controle	%A
7	Operando de controle	%M

Tabela 3-57. Relação dos operandos com a configuração da CHF

A configuração do operando de controle no sétimo parâmetro só é possível a partir da versão 1.10 do módulo função F-PID16.056. Com o sétimo parâmetro habilitado é possível configurar de dentro do skin PID os parâmetros de manual e automático e direto e reverso. Caso o usuário não esteja utilizado esta versão estará disponível o link **Clique aqui para atualizar o módulo para a versão mais recente e habilitar o operando de controle.** Este link só está disponível a partir da versão 3.05 dos Módulos de Função (ver menu **Ajuda/Sobre** para consultar a versão dos Módulos Função).

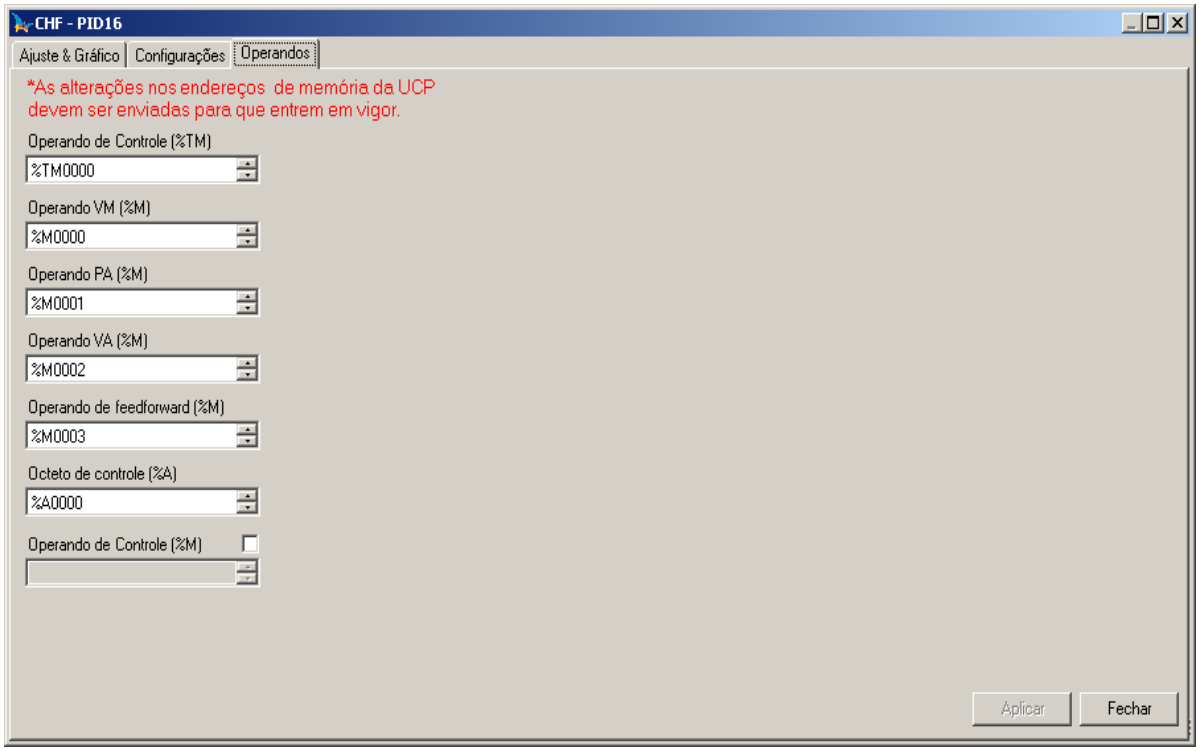


Figura 3-21. Janela módulo F-PID16.056, aba Operandos

Se o módulo função utilizado for da versão 1.10 ou superior, o usuário poderá selecionar se deseja utilizar o sétimo parâmetro da função.

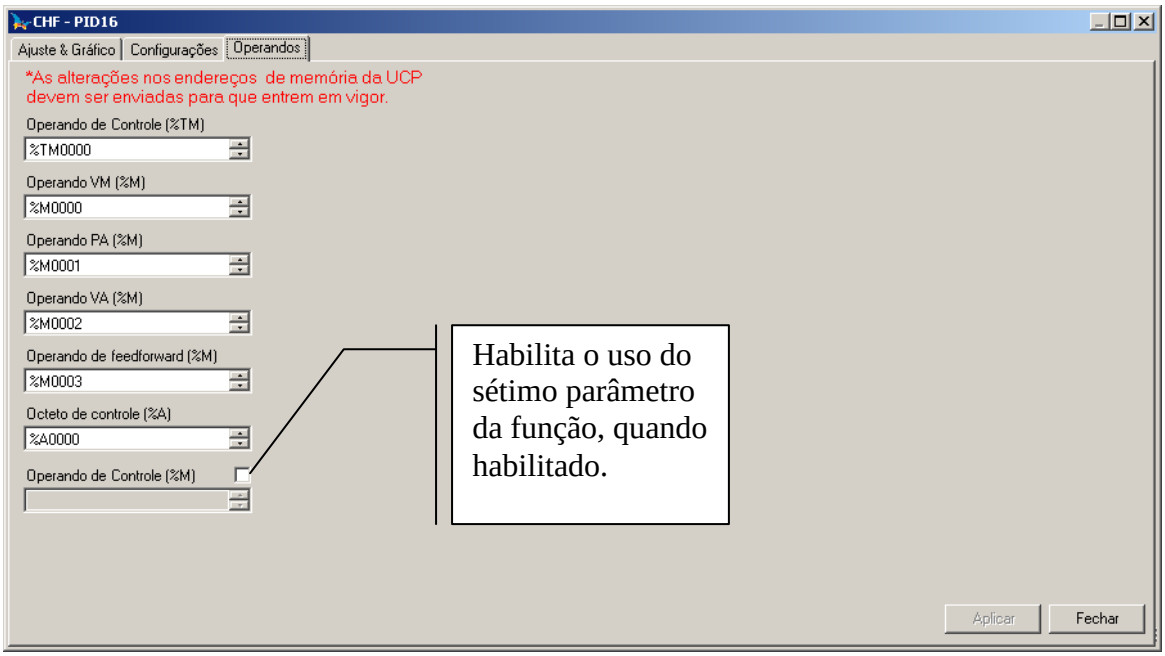


Figura 3-22. Janela módulo F-PID16.056, aba operandos, a partir da versão 1.10

ATENÇÃO: Qualquer alteração nesta janela só será válida após o envio do módulo ladder que contém a instrução CHF para o CP.

A janela de configuração deve ser utilizada para a configuração *online* do PID. Caso ocorra um erro de comunicação será exibido um sinal exclamação, em cima da variável com este problema como é mostrado na figura a seguir:

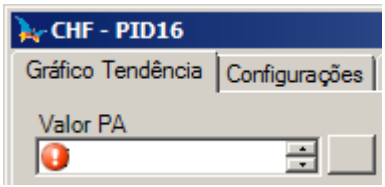


Figura 3-23. Variável com problema de comunicação

CHF - Chama Módulo Função – Especial AL-2752

A instrução CHF também possui uma configuração especial para o módulo F-2005.016, quando este está configurado para usar Função 100 Laços PID AL-2752. O AL-2752 é um progama executado pelo AL-2005 que calcula até 100 laços PID. O skin PID para AL-2752 é semelhante ao skin PID para o módulo função F-PID16.056, com poucas diferenças.

ATENÇÃO: Para o correto entendimento do modo especial da CHF é necessário ler o manual de utilização do AL-2752 Função 100 laços PID.

Para habilitar o uso do skin PID para o AL-2752 a instrução CHF deve estar corretamente configurada para o uso com o AL-2752. Ou seja:

- Deve haver uma única instrução CAB configurando a tabela de configuração do AL-2752 (segundo parâmetro de entrada da instrução CHF)
- Na instrução CAB, a posição 2 deve estar com o valor 2752 (código para o AL-2752)
- Na instrução CAB, a posição 4, que indica o número de laços PID deve estar com um valor entre 1 e 100

Ao abrir a janela de skin PID o usuário deverá antes selecionar qual laço ele pretende configurar.

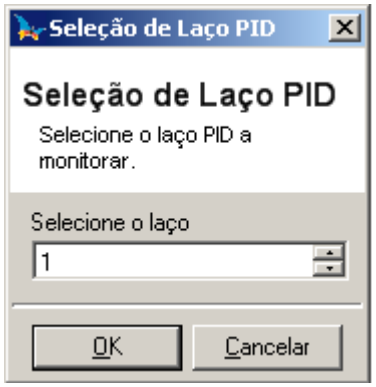


Figura 3-24. Janela de seleção do laço PID

Aba: Ajuste e Gráfico

A aba Ajuste e Grágico(gráfico de tTendência) possui as mesmas funcionalidades descritas no skin PID para o módulo F-PID16.056. Contudo não possui botão para configurar a instrução CAB de configuração do laço PID, já que isto é um recurso exclusivo do módulo deste módulo F.

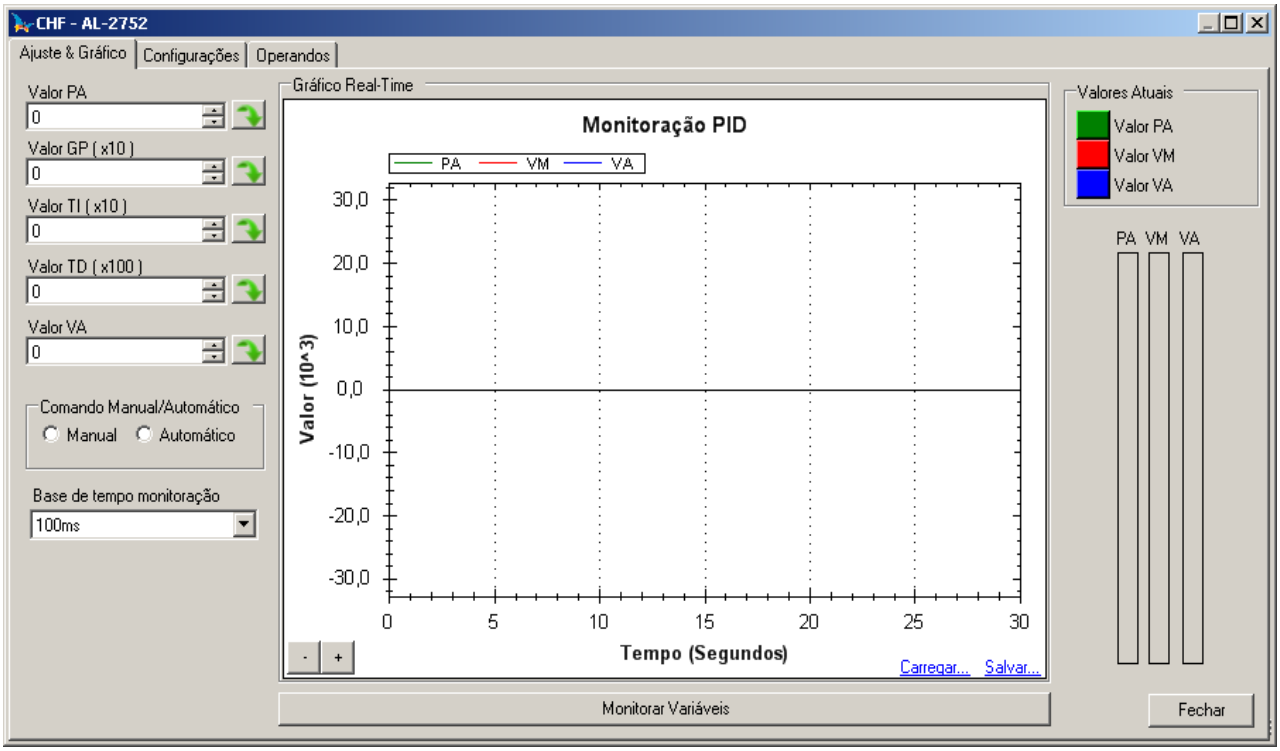


Figura 3-25. Janela skin PID para o AL-2752 , aba Ajuste e Gráfico

Aba: Configurações

A aba Configurações é utilizada para configurar os parâmetros avançados do PID.

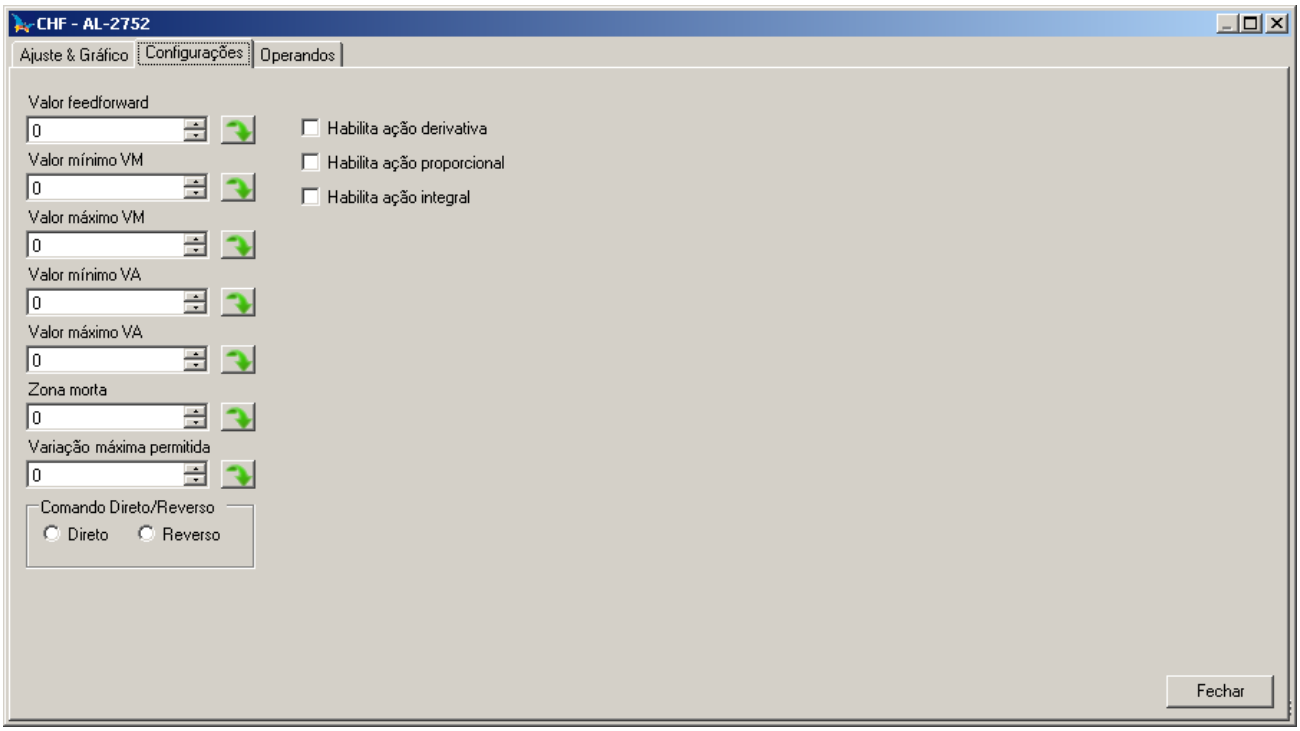


Figura 3-26. Janela para o AL-2752, aba Configurações

Os valores que podem ser configurados são:

- Valor feedforward/bias
- Valor mínimo VM
- Valor Máximo VM

- Valor mínimo VA
- Valor máximo VA
- Zona morta
- Variação máxima permitida
- Habilita ação derivativa
- Habilita ação proporcional
- Habilita ação integral
- Comando direto e reverso

Para configuração dos valores o usuário deve digitar o novo valor dentro da caixa de texto. Quando isto acontecer, será parada a monitoração da respectiva variável e a mesma ficará na cor vermelho, aguardando que o usuário confirme o envio do valor para o CP através do botão que fica à direta da caixa de texto. Se o usuário fechar a janela, os valores editados não são enviados, são descartados. Os comandos de habilitação de ação são enviados imediatamente.

Aba: Operandos

Na janela skin PID para o AL-2752 não é possível configurar os operandos do laço PID, apenas visualizar quais operandos são utilizados para as variáveis do laço.

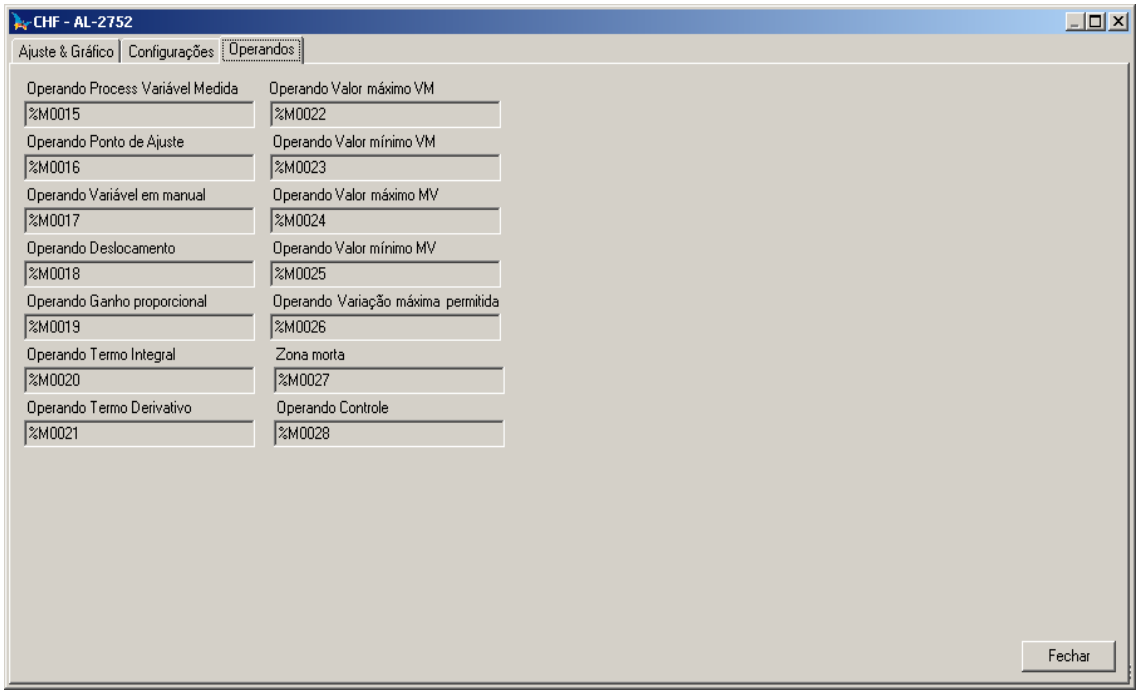


Figura 3-27. Janela para o AL-2752, aba Operandos

ECR - Escrita de Operandos em Outro CP

OPER1 - endereço de nodo do controlador remoto
OPER2 - endereço da sub-rede do controlador remoto
OPER3 - operando de controle da instrução

Descrição:

Esta instrução realiza a escrita de valores de operandos do controlador onde está sendo executada em operandos presentes em outros CPs, através da rede ALNET II de comunicação. Para o seu uso, portanto, é imprescindível que o controlador que a execute esteja conectado a outros CPs pela ALNET II.

Através da ECR podem ser transferidos valores de operandos individuais ou de conjuntos de operandos, sendo possível a programação de até seis comunicações diferentes em uma mesma instrução. A ECR pode ser programada para ser prioritária, enviando uma comunicação urgente, processada pelos "bridges" e pelo CP destino antes das comunicações comuns. A ECR prioritária permite somente uma comunicação, sendo útil para a sinalização de alarmes ou situações de emergência entre CPs.

Para programar a instrução, deve-se declarar na primeira e segunda células (OPER1 e OPER2) os endereços de nodo e sub-rede do controlador programável destino que irá receber os valores escritos. Estes operandos são programados como constantes do tipo memória (%KM) e possuem o mesmo significado dos endereços configurados nas opções **Comunicação, Endereço, Sub-rede e Comunicação, Endereço, Nó**.

A tabela a seguir apresenta os valores possíveis para endereços de nó e sub-rede.

Sub-rede	Nó	Tipo de Comunicação
001 a 063	001 a 031	ALNET II com um nó
064	xxx	Endereço IP, onde xxx especifica outro endereço IP dentro da mesma subrede.
100	001 a 015	ALNET II com grupo de multicast em todas as sub-redes
101 a 163	001 a 015	ALNET II com grupo de multicast em uma sub-rede específica
200	xxx	ALNET II em broadcast para todas as sub-redes
201 a 263	xxx	ALNET II em broadcast para uma sub-rede específica

Tabela 3-58. Endereços de nó e sub-rede

O endereço de sub-rede entre os valores 001 e 063 indica que a comunicação é realizada utilizando a rede ALNET II e que é destinada a um único nó indicado na opção Nó.

O endereço de sub-rede 100 indica que a comunicação é realizada utilizando a rede ALNET II e destinada a todos os nós de todas as sub-redes que pertençam ao grupo de multicast especificado na opção Nó (multicast global).

O endereço de sub-rede entre 101 e 163 indica que a comunicação é realizada utilizando a rede ALNET II e que é destinada a todos os nós da sub-rede indicada na opção Sub-rede menos 100 que pertençam ao grupo de multicast especificado na opção Nó (multicast local).

O endereço de sub-rede 200 indica que a comunicação é realizada utilizando a rede ALNET II e é destinado a todos os nós de todas as sub-redes (broadcast global). O valor contido na opção Nó não é relevante nesta opção.

O endereço de sub-rede entre 201 e 263 indica que a comunicação é realizada utilizando a rede ALNET II e destinada a todos os nós da sub-rede indicada na opção Sub-rede menos 200 (broadcast local). O valor contido na opção Nó não é relevante nesta opção.

Na terceira célula (OPER3) deve ser declarado um operando decimal (%D) para ser utilizado pela própria instrução no controle do seu processamento.

ATENÇÃO:
O operando %D programado em OPER3 não pode ter o seu valor modificado em nenhum outro ponto do programa aplicativo para o correto funcionamento da ECR. Conseqüentemente, cada nova instrução ECR, LTR, CEH ou LTH inserida no programa aplicativo deve utilizar um operando %D diferente das demais. Este operando não pode ser retentivo.

Para realizar a edição dos parâmetros da ECR deve-se clicar no botão **CP...** A fazer isto, será aberto uma janela de edição das mensagens da instrução, como mostrado na próxima figura:

Figura 3-28. Caixa de diálogo dos parâmetros ECR

Nesta janela, há uma tabela onde cada linha representa uma das seis comunicações possíveis entre CP Local e CP Remoto. Nesta tabela há as seguintes colunas:

- **CP Local:** Faixas de operandos do CP local que serão enviados ao CP Remoto através da instrução;
- **CP Remoto:** Faixas de operandos do CP Remoto que será escrito os valores vindos das comunicações da instrução.

Quando uma linha estiver vazia, significa que não será efetuado uma comunicação, logo, na figura anterior teríamos apenas duas comunicações sendo feitas pela instrução.

O item **Mensagem Prioritária** permite a edição de uma ECR prioritária quando estiver selecionado. Neste caso, será permitido apenas uma comunicação, ficando na tabela apenas a primeira linha.

ATENÇÃO:
Na verificação de Projeto do MasterTool Extended Edition, os operandos especificados para o CP local são consistidos pelo MasterTool XE de acordo com as declarações constantes no módulo C presente no mesmo, por pertencerem ao programa aplicativo que está sendo editado. Porém, os operandos declarados para o CP remoto não sofrem consistências quanto a tipo e endereços, por pertencerem a um programa aplicativo de outro controlador programável.

Para editar ou inserir uma comunicação na instrução, basta efetuar um duplo clique sobre a linha correspondente, ou então selecionar a linha e clicar no botão Configuração... Independente da forma utilizada será aberto a seguinte janela de edição das mensagens:

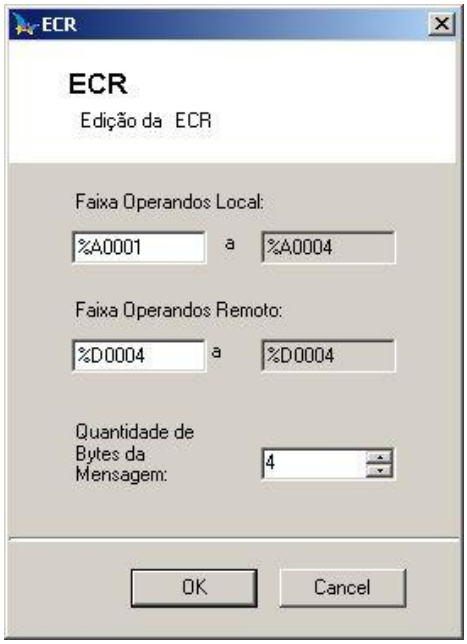


Figura 3-29. Edição de mensagem do CP local para CP remoto

Nesta janela, deverá ser informado o operando inicial do CP Local e o operando inicial do CP Remoto. Deve ser informado também a quantidade de bytes utilizados na comunicação, tendo como limite superior 220 bytes. O valor desta quantidade de bytes será múltiplo da maior quantidade de bytes em um único operando do CP Local ou CP Remoto (Ver operandos neste mesmo manual). Na figura anterior o Número Múltiplo é 4 porque o operando decimal (%D) possui quatro bytes, enquanto que o operando auxiliar (%A) possui apenas 1.

A seguir estão relacionados os tipos de operandos possíveis de serem programados para o CP local e remoto, com a disposição correta dos mesmos nas colunas de edição e os seus respectivos significados.

Operandos Permitidos como Parâmetros
%M
%D
%F
%E
%S
%A
%TM[x]
%TD[x]
%TF[x]
%TI[x]

Tabela 3-59. Operandos permitidos como parâmetros

Ao ser acionada a entrada **habilita**, é disparada a comunicação da primeira escrita presente na ECR, sendo energizada a saída **ocupado** da mesma. No momento em que esta comunicação for completada, a instrução dispara a próxima escrita, independentemente do estado da entrada de habilitação, repetindo este procedimento para as demais comunicações existentes nesta instrução. Ao final da última escrita, a saída **ocupado** da ECR é desenergizada, com o disparo de um pulso com duração de uma varredura na saída **erro** caso não tenha sido possível realizar alguma comunicação.

Nos seis primeiros nibbles do operando D programado em OPER3 são colocados os estados das seis comunicações da instrução. Os últimos dois nibbles são utilizados para o controle do seu processamento.

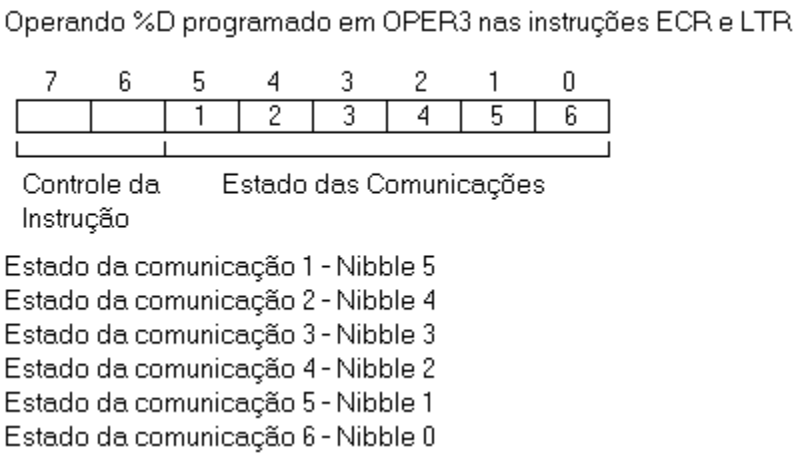


Figura 3-30. Operando de controle da instrução ECR e LTR

O estado da comunicação armazenado em cada nibble é codificado da seguinte forma:

- 0 - comunicação com sucesso
- 1 - operando não definido
- 2 - endereço do controlador local igual ao remoto (comunicação para o próprio CP)
- 3 - bloco de operando inválido
- 4 - tipo de operando inválido
- 5 - time-out de transmissão de pacote
- 6 - não há espaço na fila de espera de transmissão
- 7 - falta de buffer de transmissão
- 8 - time-out de requisição
- 9 - erro de hardware
- 10 - CP remoto protegido

Em resumo, ao ser disparada a execução de uma instrução ECR todas as comunicações existentes na mesma são realizadas, mesmo que a sua entrada de habilitação seja desenergizada. Quando todas as escritas forem completadas, a próxima instrução ECR ou LTR encontrada no programa aplicativo com a entrada **habilita** energizada torna-se ativa, começando a processar as suas comunicações.

ATENÇÃO: O programa aplicativo não pode realizar saltos sobre a instrução ECR ativa ou deixar de executar o módulo que a contém, para assegurar o seu correto processamento.

Em um programa aplicativo sendo executado no CP, em um dado momento, somente uma instrução de acesso à rede ALNET II (ECR ou LTR) é considerada ativa, mesmo que existam várias instruções com entrada **habilita** acionadas. A saída **ocupado** determina qual a instrução ativa, podendo ser utilizada para sincronizar as comunicações com o programa aplicativo. Para evitar sobrecargas no tráfego de informações na rede, aconselha-se disparar as instruções ECR periodicamente, evitando mantê-las permanentemente habilitadas no programa aplicativo, se possível. Um procedimento recomendado é desligar a entrada **habilita** logo que a saída **ocupado** for energizada, evitando um novo disparo da instrução após seu término.

A ECR prioritária não segue a ordem de processamento das ECRs não prioritárias, sendo processada e transmitindo seus dados o mais rápido possível, ao ser habilitada. Por esse motivo uma ECR prioritária não deve ficar permanentemente habilitada, devendo ser disparada somente em situações de alarme ou periodicamente. Caso contrário, pode impedir que as demais ECRs do programa realizem as suas comunicações ou causar o esgotamento de buffers de recepção do CP destino.

Se a instrução for programada especificando-se o endereço de nodo igual ao endereço do próprio controlador que a executa (escrita de valores de si próprio), a saída **erro** é energizada.

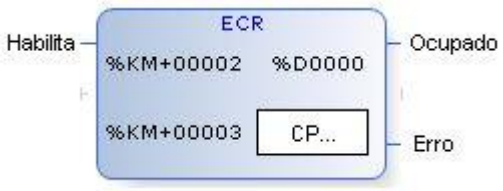
Caso nenhum operando tenha sido definido em OPER4, as saídas **erro** e **ocupado** ficam desenergizadas.

Sintaxe:

OPER1	OPER2	OPER3	OPER4
%KM	%KM	%D	COMUNICAÇÕES

Tabela 3-60. Sintaxe da instrução ECR

Exemplo:



Conteúdo das mensagens da ECR não prioritária:

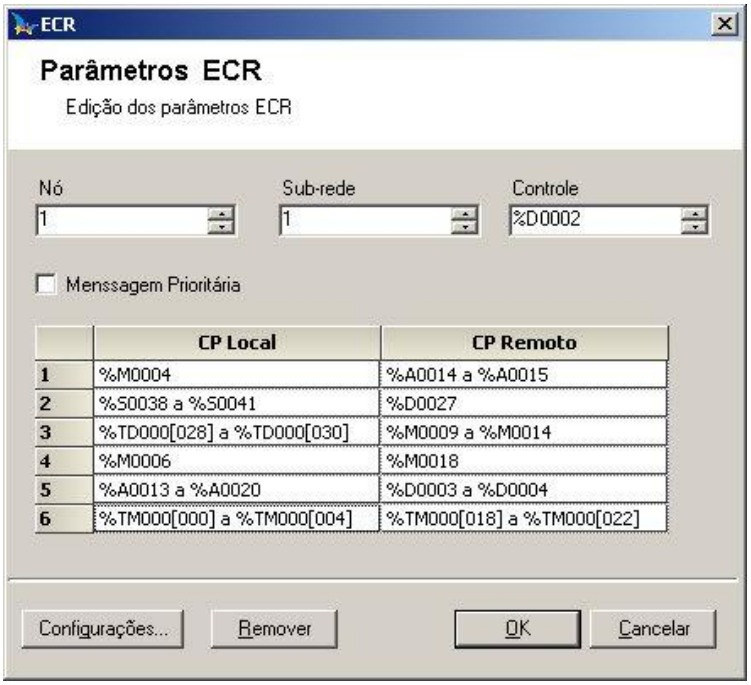


Figura 3-31. Configuração dos parâmetros ECR com mensagem não prioritária

Esta instrução realiza escritas no controlador programável com o endereço de nó igual a 2 na sub-rede 1. São definidas seis comunicações para a mesma, transferindo dados de diversos tipos entre os CPs. A comunicação 0 envia o conteúdo de um operando memória no CP local para dois operandos auxiliares no CP remoto, sendo transferidos 2 octetos. As comunicações 1, 2, 3, 4 e 5 transferem, respectivamente, 4, 12, 2, 8 e 10 octetos entre os controladores.

Conteúdo das mensagens da ECR prioritária:

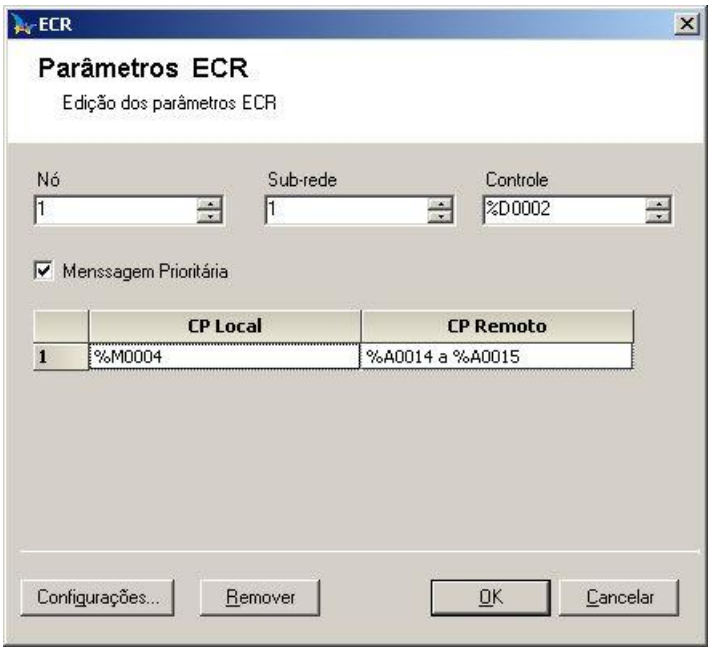


Figura 3-32. Configuração dos parâmetros ECR com mensagem prioritária

Esta instrução realiza escritas no controlador programável com o endereço de nó igual a 2 na sub-rede 1. É definida uma comunicação prioritária para a mesma. A comunicação P/1 envia o conteúdo de um operando memória no CP local para dois operandos auxiliares no CP remoto, sendo transferidos 2 octetos.

ATENÇÃO: Esta instrução somente pode ser utilizada nas UCPs AL-2004.

LTR - Leitura de Operandos de Outro CP

OPER1 - endereço de nodo do controlador remoto
OPER2 - endereço da sub-rede do controlador remoto
OPER3 - operando de controle da instrução

Descrição:

Esta instrução realiza a leitura de valores de operandos presentes em outros controladores programáveis para operandos do controlador programável onde está sendo executada, através da rede ALNET II de comunicação. Para o seu uso, portanto, é imprescindível que o CP que a execute esteja conectado a outros CPs pela ALNET II.

Através da LTR podem ser lidos valores de operandos individuais ou de conjuntos de operandos, sendo possível a programação de até 6 comunicações de leitura diferentes em uma mesma instrução.

A programação da instrução LTR é idêntica à ECR, observando as mesmas restrições. Na LTR, a transferência dos valores ocorre dos operandos declarados no CP remoto para o CP local, sendo esta a única diferença entre ambas.

ATENÇÃO: A instrução LTR difere da ECR quanto a possibilidade de mensagens prioritárias, ou seja, não é possível editar uma LTR prioritária.

Sintaxe:

OPER1	OPER2	OPER3	OPER4
%KM	%KM	%D	COMUNICAÇÕES

Tabela 3-61. Sintaxe da instrução LTR

Exemplo:

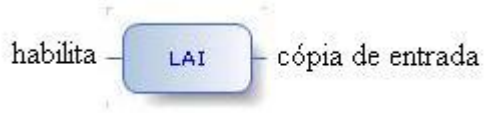
Conteúdo das mensagens da LTR:

Figura 3-33. Configuração dos parâmetros LTR

Esta instrução realiza leituras no controlador programável com o endereço de nó igual a 2 na sub-rede 1. São definidas seis comunicações para a mesma, transferindo dados de diversos tipos entre os CPs. A comunicação 0 lê o conteúdo de dois operandos auxiliares no CP remoto para um operando memória no CP local, sendo transferidos 2 octetos. As comunicações 1, 2, 3, 4 e 5 transferem, respectivamente, 4, 12, 2, 8 e 10 octetos entre os controladores programáveis.

ATENÇÃO: Esta instrução somente pode ser utilizada nas AL-2004.

LAI - Libera Atualização de Imagens dos Operandos



Descrição:

A instrução libera atualização da imagem de operandos realiza o processamento das comunicações pendentes da rede ALNET II para o CP local.

Ao retornar para o processamento do software executivo, ao final de cada varredura, o CP processa as requisições de leitura e outros serviços que tenham sido solicitados para o mesmo por outros CPs presentes na rede, durante a execução do programa aplicativo.

O controlador programável possui uma área de memória reservada para o armazenamento de até 32 comunicações recebidas durante o laço de execução do programa aplicativo, enquanto o software executivo não as processa. Caso o programa aplicativo possua tempo de execução relativamente alto e o controlador programável receba muitas requisições de serviços da rede, pode ocorrer a situação do CP não conseguir atendê-las, chegando ao limite de 32 comunicações pendentes à espera de processamento. Neste caso, o CP devolve uma resposta ao requisitante indicando a impossibilidade de atender a sua comunicação.

A instrução LAI executa o processamento de recepções e transmissões pendentes no CP, diminuindo a possibilidade de ocorrência da situação descrita anteriormente e reduzindo o tempo de atendimento às requisições. É recomendado o seu uso em programas aplicativos com alto tempo de ciclo, devendo ser inserida em pontos intermediários dos módulos, dividindo-os em trechos com aproximadamente 20 ms de tempo de execução.

ATENÇÃO: Os valores dos operandos do programa aplicativo podem ser modificados após a execução de uma LAI, pois outro equipamento ligado à rede pode estar solicitando escritas nos mesmos. Deve-se considerar a influência deste fato ao se inserir esta instrução no programa aplicativo.

ATENÇÃO: Esta instrução somente pode ser utilizada nas UCPs AL-2004.

ECH - Escrita de Operandos em Outro CP pela Ethernet

IP - endereço IP do controlador remoto
OPER1 - operando de controle da instrução

Descrição:

Esta instrução realiza a escrita de valores de operandos do controlador onde está sendo executada em operandos presentes em outros CPs, através da rede Ethernet de comunicação. Para o seu uso, portanto, é imprescindível que o controlador que a execute esteja conectado a outros CPs pela Ethernet.

Através da ECH podem ser transferidos valores de operandos individuais ou de conjuntos de operandos, sendo possível a programação de até 6 comunicações diferentes em uma mesma instrução.

Para programar a instrução, deve-se declarar o endereço IP do controlador programável destino que irá receber os valores escritos. Já no campo OPER1 deve ser declarado um operando decimal (%D) para ser utilizado pela própria instrução no controle do seu processamento.

ATENÇÃO: O operando %D programado em OPER1 não pode ter o seu valor modificado em nenhum outro ponto do programa aplicativo para o correto funcionamento da ECH. Conseqüentemente, cada nova instrução ECH, LTH, ECR ou LTR inserida no programa aplicativo deve utilizar um operando %D diferente das demais. Este operando não pode ser retentivo.

A instrução ECH possui o mesmo comportamento e as mesmas sintaxes que a instrução ECR, no que diz respeito ao operando de controle e a edição das mensagens. A diferença básica fica por conta da rede onde a instrução atua, neste caso é via Ethernet. Sendo assim, para maiores detalhes que como preencher e utilizar os operandos desta instrução, ver a instrução ECR.

LTH - Leitura de Operandos de Outro CP pela Ethernet

IP - endereço IP do controlador remoto
OPER1 - operando de controle da instrução

Descrição:

Esta instrução realiza a leitura de valores de operandos presentes em outros controladores programáveis para operandos do controlador programável onde está sendo executada, através da rede Ethernet de comunicação. Para o seu uso, portanto, é imprescindível que o CP que a execute esteja conectado a outros CPs pela Ethernet.

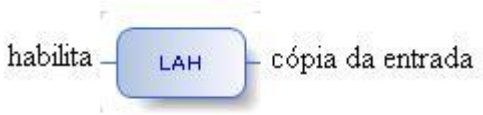
Para programar a instrução, deve-se declarar o endereço IP do controlador programável destino que irá receber os valores escritos. Já no campo OPER1 deve ser declarado um operando decimal (%D) para ser utilizado pela própria instrução no controle do seu processamento.

ATENÇÃO: O operando %D programado em OPER1 não pode ter o seu valor modificado em nenhum outro ponto do programa aplicativo para o correto funcionamento da LTH. Conseqüentemente, cada nova instrução ECH, LTH, ECR ou LTR inserida no programa aplicativo deve utilizar um operando %D diferente das demais. Este operando não pode ser retentivo.

Através da LTH podem ser lidos valores de operandos individuais ou de conjuntos de operandos, sendo possível a programação de até 6 comunicações de leitura diferentes em uma mesma instrução.

A programação da instrução LTH é idêntica à LTR, observando as mesmas restrições. A única diferença fica por conta de que esta instrução é utilizada em redes Ethernet.

LAH - Libera Atualização de Imagens dos Operandos pela Ethernet



Descrição:

A instrução libera atualização da imagem de operandos realiza o processamento das comunicações pendentes da rede Ethernet para o CP local.

Ao retornar para o processamento do software executivo, ao final de cada varredura, o CP processa as requisições de leitura e outros serviços que tenham sido solicitados para o mesmo por outros CPs presentes na rede, durante a execução do programa aplicativo.

O controlador programável possui uma área de memória reservada para o armazenamento de até 32 comunicações recebidas durante o laço de execução do programa aplicativo, enquanto o software executivo não as processa. Caso o programa aplicativo possua tempo de execução relativamente alto e o controlador programável receba muitas requisições de serviços da rede, pode ocorrer a situação do CP não conseguir atendê-las, chegando ao limite de 32 comunicações pendentes à espera de processamento. Neste caso, o CP devolve uma resposta ao requisitante indicando a impossibilidade de atender a sua comunicação.

A instrução LAH executa o processamento de recepções e transmissões pendentes no CP, diminuindo a possibilidade de ocorrência da situação descrita anteriormente e reduzindo o tempo de atendimento às requisições. É recomendado o seu uso em programas aplicativos com alto tempo de ciclo, devendo ser inserida em pontos intermediários dos módulos, dividindo-os em trechos com aproximadamente 20 ms de tempo de execução.

ATENÇÃO: Os valores dos operandos do programa aplicativo podem ser modificados após a execução de uma LAH, pois outro equipamento ligado à rede pode estar solicitando escritas nos mesmos. Deve-se considerar a influência deste fato ao se inserir esta instrução no programa aplicativo.

Instruções do Grupo Ligações

As instruções do grupo ligações permitem a construção de caminhos em série e em paralelo bem como a inversão do sinal.

Nome	Descrição do Nome	Seqüência de Edição
LGH	Ligação horizontal	Alt, I, L, H
NEG	Ligação negada	Alt, I, L, N
LGV	Ligação vertical	Alt, I, L, V

Tabela 3-62. Instruções do grupo ligações

LGH - Ligação Horizontal

NEG - Ligação Negada

LGV - Ligação Vertical

Descrição:

As ligações são elementos auxiliares na construção dos diagramas de relés, para interligar as demais instruções.

A ligação negada inverte na sua saída o estado lógico da sua entrada.

4. Glossário

Algoritmo	Seqüência finita de instruções bem definidas, objetivando à resolução de problemas.
Barramento	Conjunto de sinais elétricos agrupados logicamente com a função de transferir informação e controle entre diferentes elementos de um subsistema.
Bit	Unidade básica de informação, podendo estar no estado 0 ou 1.
BT	Sigla para teste de bateria em inglês (battery test).
Byte	Unidade de informação composta por oito bits.
Ciclo de varredura	Uma execução completa do programa aplicativo de um controlador programável.
Circuito de cão de guarda	Circuito eletrônico destinado a verificar a integridade do funcionamento de um equipamento.
Código comercial	Código do produto, formado pelas letras PO, seguidas por quatro números.
Controlador programável	Também chamado de CP. Equipamento que realiza controle sob o comando de um programa aplicativo. É composto de uma UCP, uma fonte de alimentação e uma estrutura de E/S.
CP	Veja controlador programável.
Database	Banco de dados.
Default	Valor predefinido para uma variável, utilizado em caso de não haver definição.
Diagnóstico	Procedimento utilizado para detectar e isolar falhas. É também o conjunto de dados usados para tal determinação, que serve para a análise e correção de problemas.
Download	Carga de programa ou configuração no CP.
E/S	Veja entrada/saída.
E2PROM	Memória não-volátil, que pode ser apagada eletricamente.
Encoder	Transdutor para medidas de posição.
Endereço de módulo	Endereço pelo qual o CP realiza acessos a um determinado módulo de E/S.
Entrada/saída	Também chamado de E/S. Dispositivos de E/S de dados de um sistema. No caso de CPs, correspondem tipicamente a módulos digitais ou analógicos de entrada ou saída que monitoram ou acionam o dispositivo controlado.
EPROM	Significa Erasable Programmable Read Only Memory. É uma memória somente de leitura, apagável e programável. Não perde seu conteúdo quando desenergizada.
ER	Sigla usada para indicar erro nos LEDs.
ESD	Sigla para descarga devida a eletricidade estática em inglês (electrostatic discharge).
Estação de supervisão	Equipamento ligado a uma rede de CPs ou instrumentação com a finalidade de monitorar ou controlar variáveis de um processo.
FLASH EPROM	Memória não-volátil, que pode ser apagada eletricamente.
Hardware	Equipamentos físicos usados em processamento de dados onde normalmente são executados programas (software).
IEC 61131	Norma genérica para operação e utilização de CPs. Antiga IEC 1131.
IEC Pub. 144 (1963)	Norma para proteção contra acessos incidentais e vedação contra água, pó ou outros objetos estranhos ao equipamento.
IEC-536-1976	Norma para proteção contra choque elétrico.
IEC-801-4	Norma para testes de imunidade a interferências por trem de pulsos.
IEEE C37.90.1 (SWC)	SWC significa Surge Withstand Capability. Esta norma trata da proteção do equipamento contra ruídos tipo onda oscilatória.
Interface	Dispositivo que adapta elétrica e/ou logicamente a transferência de sinais entre dois equipamentos.
Interrupção	Evento com atendimento prioritário que temporariamente suspende a execução de um programa e desvia para uma rotina de atendimento específica
ISOL.	Sigla usada para indicar isolado ou isolamento.
kbytes	Unidade representativa de quantidade de memória. Representa 1024 bytes.
LED	Sigla para light emitting diode. É um tipo de diodo semicondutor que emite luz quando estimulado por eletricidade. Utilizado como indicador luminoso.
Linguagem Assembly	Linguagem de programação do microprocessador, também conhecida como linguagem de máquina.
Linguagem de programação	Um conjunto de regras e convenções utilizado para a elaboração de um programa.
Linguagem de relés e blocos Altus	Conjunto de instruções e operandos que permitem a edição de um programa aplicativo para ser utilizado em um CP.
Lógica	Matriz gráfica onde são inseridas as instruções de linguagem de um diagrama de relés que compõe um programa aplicativo. Um conjunto de lógicas ordenadas seqüencialmente constitui um módulo de programa.
Menu	Conjunto de opções disponíveis e exibidas por um programa no vídeo e que podem ser selecionadas pelo

	usuário a fim de ativar ou executar uma determinada tarefa.
Módulo (referindo-se a hardware)	Elemento básico de um sistema completo que possui funções bem definidas. Normalmente é ligado ao sistema por conectores, podendo ser facilmente substituído.
Módulo (referindo-se a software)	Parte de um programa aplicativo capaz de realizar uma função específica. Pode ser executado independentemente ou em conjunto com outros módulos, trocando informações através da passagem de parâmetros.
Módulo C	Veja módulo de configuração.
Módulo de configuração	Também chamado de módulo C. É um módulo único em um programa de CP que contém diversos parâmetros necessários ao funcionamento do controlador, tais como a quantidade de operandos e a disposição dos módulos de E/S no barramento.
Módulo de E/S	Módulo pertencente ao subsistema de entradas e saídas.
Módulo E	Veja módulo execução.
Módulo execução	Módulo que contém o programa aplicativo, podendo ser de três tipos: E000, E001 e E018. O módulo E000 é executado uma única vez, na energização do CP ou na passagem de programação para execução. O módulo E001 contém o trecho principal do programa que é executado ciclicamente, enquanto que o módulo E018 é acionado por interrupção de tempo.
Módulo F	Veja módulo função.
Módulo função	Módulo de um programa de CP que é chamado a partir do módulo principal (módulo E) ou a partir de outro módulo função ou procedimento, com passagem de parâmetros e retorno de valores. Atua como uma sub-rotina.
Módulo P	Veja módulo procedimento.
Módulo procedimento	Módulo de um programa de CP que é chamado a partir do módulo principal (módulo E) ou a partir de outro módulo procedimento ou função, sem a passagem de parâmetros.
Nibble	Unidade de informação composta por quatro bits.
Octeto	Conjunto de oito bits numerados de 0 a 7.
Operandos	Elementos sobre os quais as instruções atuam. Podem representar constantes, variáveis ou um conjunto de variáveis.
PA	Ver pontes de ajuste.
PROFIBUS PA	Significa protocolo PROFIBUS Process Automation.
PC	Sigla para programmable controller. É a abreviatura de controlador programável em inglês.
Ponte de ajuste	Chave de seleção de endereços ou configuração composta por pinos presentes na placa do circuito e um pequeno conector removível, utilizado para a seleção.
Posta em marcha	Procedimento de depuração final do sistema de controle, quando os programas de todas as estações remotas e UCPs são executados em conjunto, após terem sido desenvolvidos e verificados individualmente.
Programa aplicativo	É o programa carregado em um CP, que determina o funcionamento de uma máquina ou processo.
Programa executivo	Sistema operacional de um controlador programável. Controla as funções básicas do controlador e a execução de programas aplicativos.
RAM	Sigla para random access memory. É a memória onde todos os endereços podem ser acessados diretamente de forma aleatória e com a mesma velocidade. É volátil, ou seja, seu conteúdo é perdido quando o equipamento é desenergizado, a menos que se possua uma bateria para a retenção dos valores.
Ripple	Ondulação presente em tensão de alimentação contínua.
RX	Sigla usada para indicar recepção serial.
Sistema redundante	Sistema que contém elementos de reserva ou duplicados para executar determinada tarefa, que podem tolerar determinados tipos de falha sem que execução da tarefa seja comprometida.
Software	Programas de computador, procedimentos e regras relacionadas à operação de um sistema de processamento de dados.
Soquete	Dispositivo no qual se encaixam circuitos integrados ou outros componentes, facilitando a substituição dos mesmos e simplificando a manutenção.
Subsistema de E/S	Conjunto de módulos de E/S digitais ou analógicos e interfaces de um controlador programável.
Tag	Nome associado a um operando ou a uma lógica que permite uma identificação resumida de seu conteúdo.
Toggle	Elemento que possui dois estados estáveis, trocados alternadamente a cada ativação.
Troca a quente	Procedimento de substituição de módulos de um sistema sem a necessidade de desenergização do mesmo. Normalmente utilizado em trocas de módulos de E/S.
TX	Sigla usada para indicar transmissão serial.
UCP	Sigla para unidade central de processamento. Controla o fluxo de informações, interpreta e executa as instruções do programa e monitora os dispositivos do sistema.
UCP ativa	Em um sistema redundante, a UCP ativa realiza o controle do sistema, lendo os valores dos pontos de entrada, executando o programa aplicativo e acionando os valores das saídas.
UCP inoperante	É a UCP que não está no estado ativo (controlando o sistema) nem no estado reserva (supervisionando a UCP ativa). Não pode assumir o controle do sistema.
UCP redundante	Corresponde à outra UCP do sistema, como, por exemplo, a UCP2 em relação à UCP1 e vice-versa.

UCP reserva Em um sistema redundante, é a UCP que supervisiona a UCP ativa, não realizando o controle do sistema, mas estando pronta para assumir o controle em caso de falha na UCP ativa.

Upload Leitura do programa ou configuração do CP.

Varistor Dispositivo de proteção contra surto de tensão.

WD Sigla para cão de guarda em inglês (watchdog). Veja circuito de cão de guarda.

Word Unidade de informação composta por 16 bits.