

***Controladores Lógicos Programáveis***  
***Básico - Step 5***

**Federação das Indústrias do Estado de Pernambuco**

**Presidente**

Jorge Wicks Côrte Real

**Departamento Regional do SENAI de Pernambuco**

**Diretor Regional**

Antônio Carlos Maranhão de Aguiar

**Diretor Técnico em Exercício**

Uaci Edvaldo Matias

**Diretor Administrativo e Financeiro**

Heinz Dieter Loges

**Ficha Catalográfica**

681.5        SENAI.DR.PE. **Controladores lógicos programáveis básico – step 5**

S474c        Recife, SENAI / DITEC / DET, 2000, 82p.il.

1. ENGENHARIA DE CONTROLE AUTOMÁTICO
  2. CONTROLADORES LÓGICOS
  3. SISTEMAS DE CONTROLE
- I. Título

Direitos autorais de propriedade exclusiva do SENAI. Proibida a reprodução parcial ou total, fora do Sistema, sem a expressa autorização do Departamento Regional de Pernambuco.

SENAI – Departamento Regional de Pernambuco

Rua Frei Cassimiro, 88 – Santo Amaro

50100-260 – Recife – PE

Tel.: (81) 3416-9300

Fax: (81) 3222-3837

## **SUMÁRIO**

Introdução	5
Tipos de Sistemas de Controle	9
Tipos de Comandos	11
Introdução a Portas Lógicas	13
Introdução a Controladores Lógicos Programáveis	21
Programação em Step 5	32
Síntese dos Comandos	73
Bibliografia	83

## INTRODUÇÃO

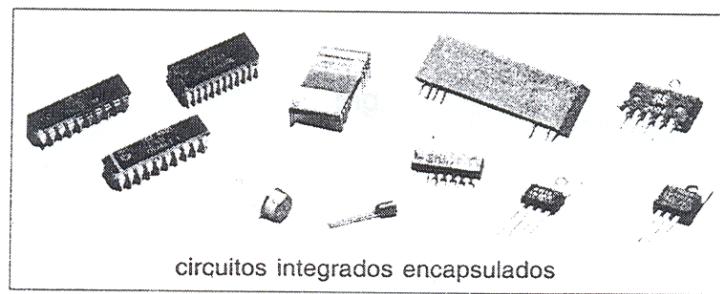
As primeiras iniciativas do homem para mecanizar atividades manuais ocorrem na pré-história, com as invenções da roda, os moinhos movidos por vento ou força animal, e as rodas d'água demonstram a criatividade para poupar esforço.

A automação só ganhou destaque na sociedade quando o sistema de produção agrária e artesanal transformou-se em industrial, a partir do século XVIII, inicialmente na Inglaterra.

No século XX, a tecnologia da automação passou a contar com computadores, servo-mecanismo e controladores programáveis. A origem do computador está relacionada à necessidade de automatizar cálculos, evidenciada inicialmente no uso de ábaco pelos babilônios entre 2000 e 3000 aC.

Em 1946, foi desenvolvido o primeiro computador de grande porte, completamente eletrônico. O Eniac, como foi chamado, ocupava mais de 180m<sup>2</sup>, e pesava 30 toneladas. Funcionava com válvulas e relês que consumiam 150 kW de potência, para realizar cerca de 5.000 cálculos aritméticos por segundo.

A segunda geração ocorreu em 1952, com o aparecimento dos transistores que não precisava se aquecer para funcionar. Com o desenvolvimento tecnológico, foi possível colocar milhares de transistores numa pastilha de silício de 1 cm<sup>2</sup>, o que resultou no circuito integrado (CI).



Em 1975, surgiram os circuitos integrados em escalas muito grande (VLSI), chamados chips, que deram origem aos computadores pessoais, de tamanhos reduzidos e baixo custo de fabricação. Para se ter uma idéia do nível de desenvolvimento desses computadores nos últimos quarenta anos, enquanto o Eniac faz mil cálculos por segundo, um chips atual faz cinqüenta milhões de cálculos no mesmo tempo.

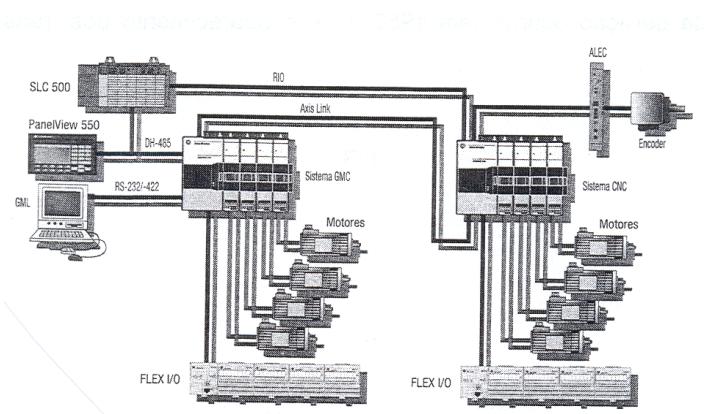
A partir desta época, fabricantes de máquinas-ferramentas começaram a desenvolver projetos particulares, que deram origem ao comando numérico, que implementou uma forma programável de automação com processo controlado por números, letras ou símbolos.

A automação dos diversos elementos de projeto e manufatura tendo como objetivo criar a fábrica do futuro, expandiram vários sistemas como o CAD (Projeto Assistido por Computador), CAM (Manufatura Auxiliada por Computador), CAE (Engenharia Auxiliada por Computador), e o CIM (Manufatura Integrada por Computador).

Cada sistema de automação compõe-se de cinco elementos:

### **Acionamento**

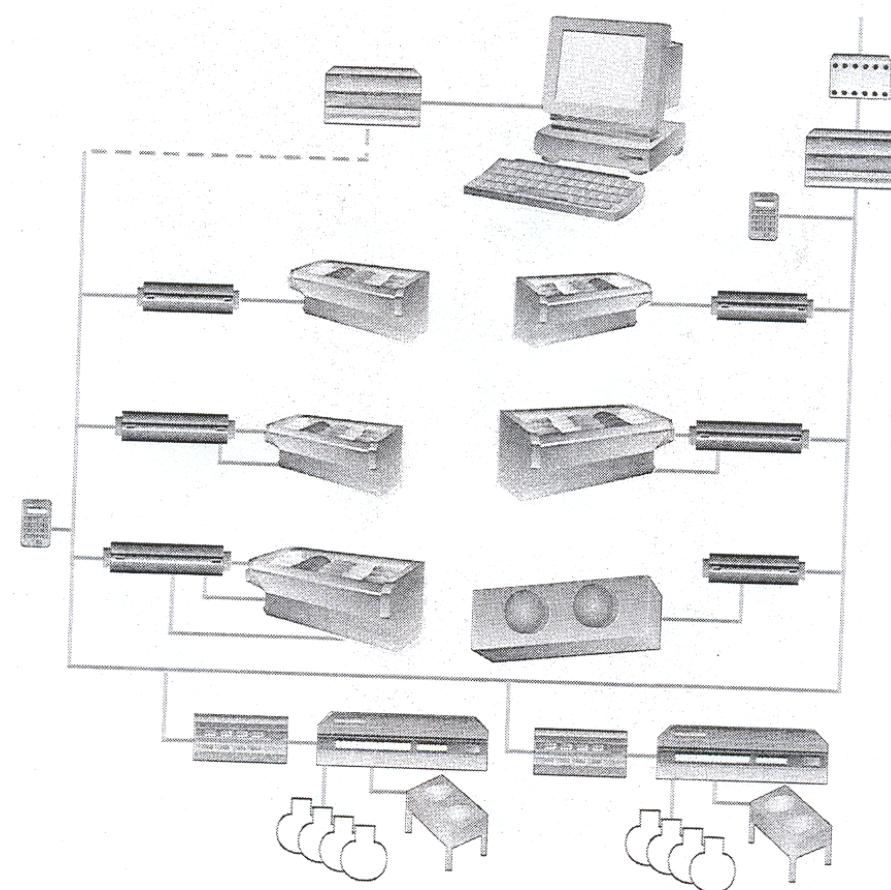
Provê o sistema de energia para atingir determinado objetivo, como acionamento de motores elétricos, atuadores pneumáticos e/ou hidráulicos, etc.



## Sensoramento

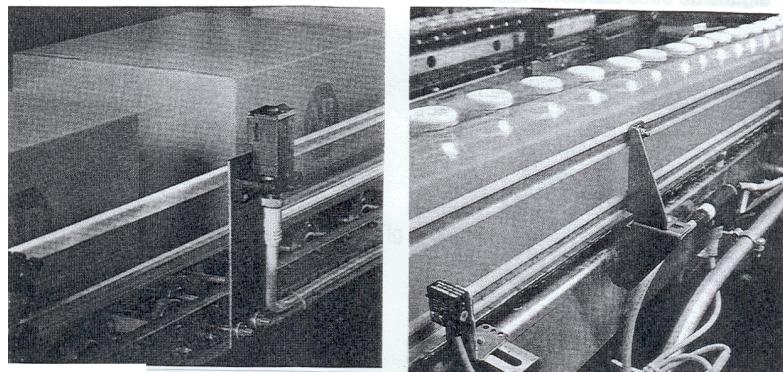
Mede o desempenho do sistema de automação, e as particularidades de alguns de seus componentes.

Exemplo: Os termosparas para medição de temperatura, e os encoders nas medições de velocidade.



## Controle

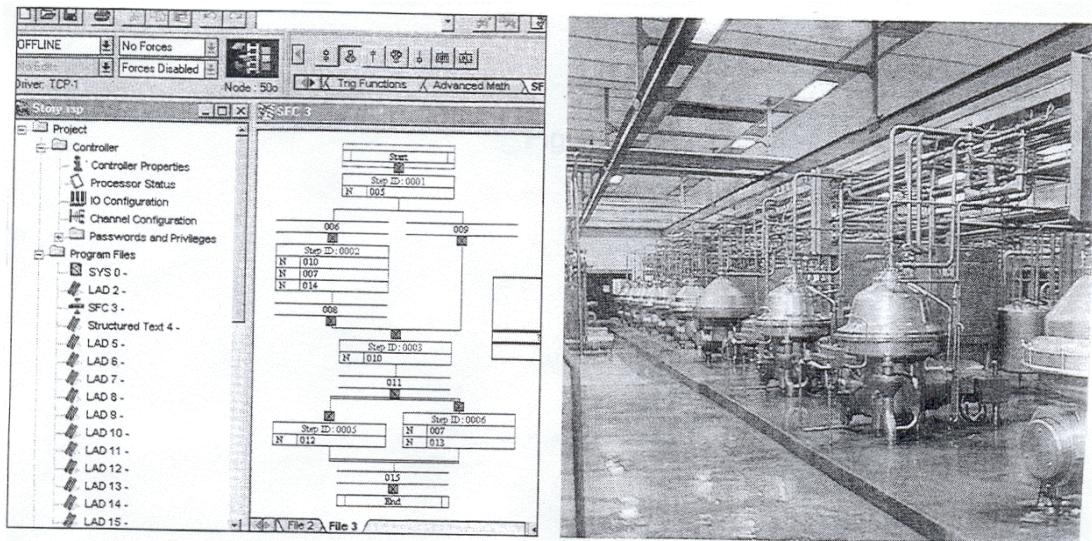
Utiliza a informação de sensores, sondas e outros transdutores, para regular o acionamento do sistema.



## Comparador

É um elemento de decisão que compara os valores medidos com os valores pré-estabelecidos e toma a decisão de quando atuar no sistema.

Exemplos: os termostatos e os softwares.



## Softwares

Contêm as informações de processo e permitem controlar as interações entre os diversos componentes, de acordo com o tipo de sistema de controle.

## TIPOS DE SISTEMAS DE CONTROLE

### SCADA

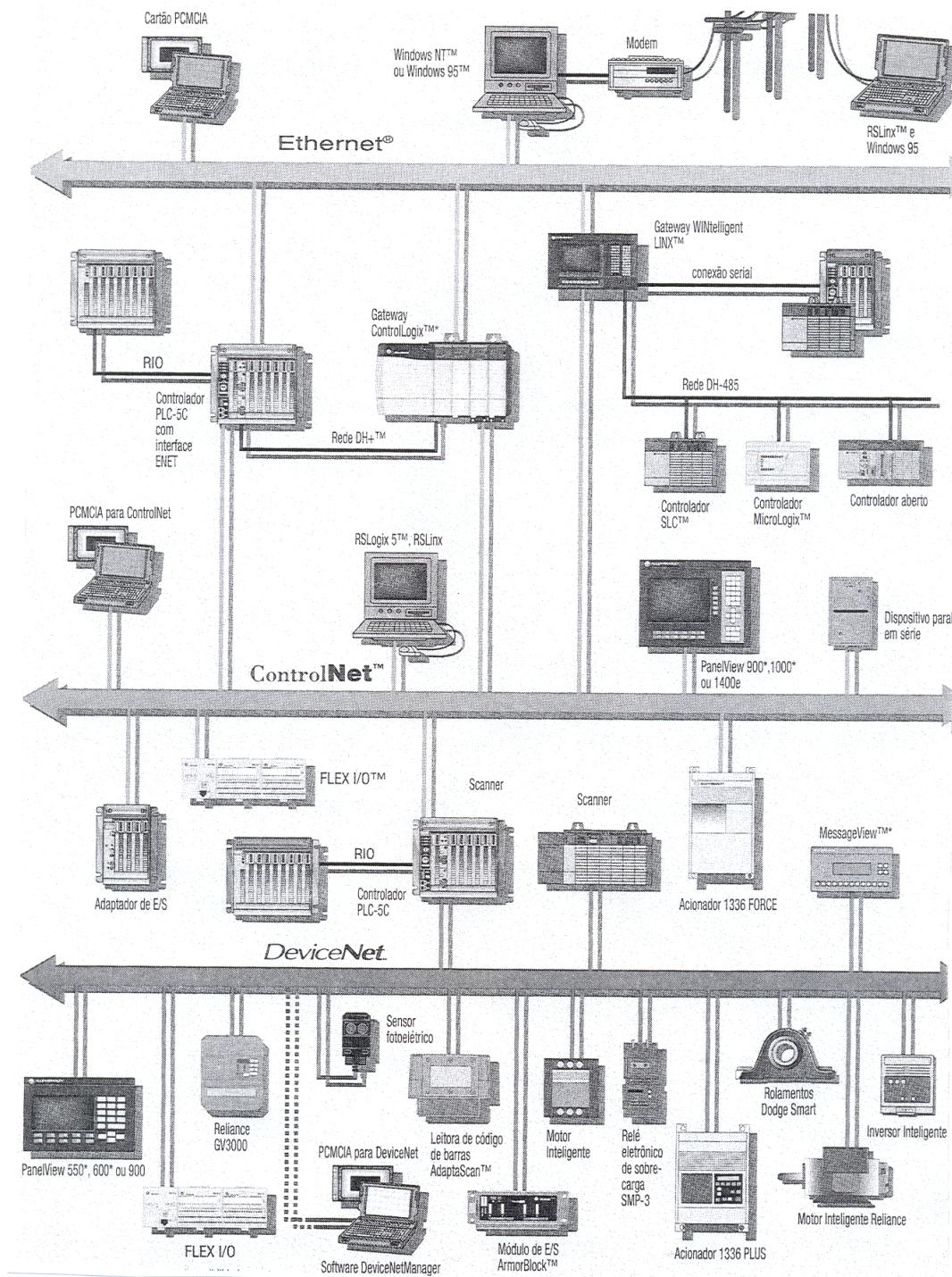
É uma designação internacional adotada para um Sistema de Supervisão, Controle e Aquisição de Dados, local ou remoto, podendo conter centenas de pontas geograficamente distribuídos. O sistema SCADA, foi concebido para aplicações de pequeno e grande porte, tendo as funções de Telemetria, Telecomando, Telesupervisão e Telealarme.

### NDA

O NDA (Network Data Acquisition), é um sistema modular tipo SCADA, projetado com interfaces inteligentes posicionadas entre dispositivos de entrada ou saída e um microcomputador, o qual utiliza um software de supervisão para monitorar e controlar os dispositivos de entrada (sensores, sondas, etc), e os de saídas (válvulas, motores, eletrobombas, etc.)

O sistema NDA tem uma ótima imunidade a ruídos, podendo operar entradas e saídas de sinais digitais ou analógicas, através de uma conexão direta, no caso de rede local, ou através de linha telefônica, rádio, ou mesmo satélite, no caso de uma rede remota.



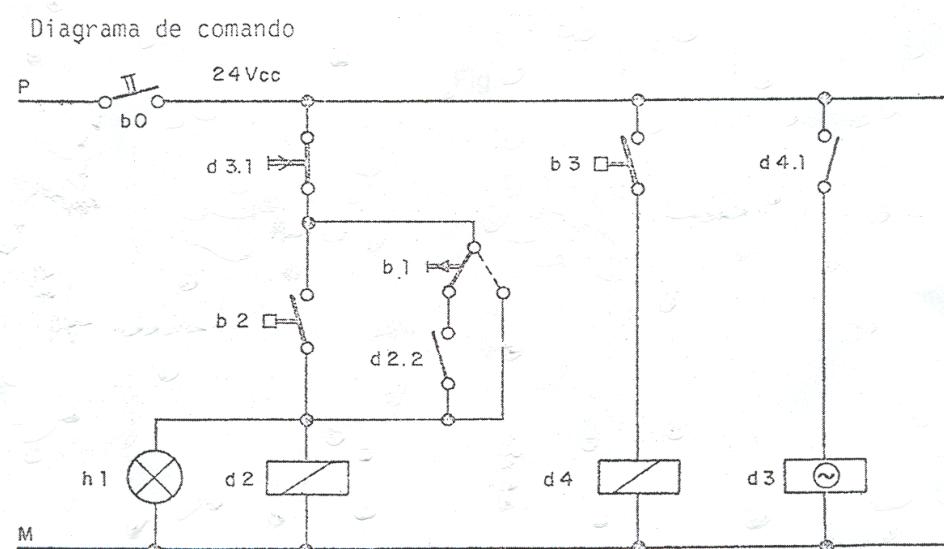


## TIPOS DE COMANDOS

### COMANDOS PROGRAMADOS POR FIAÇÃO

Também conhecidos por comandos convencionais, são aqueles cuja função estabelecida pela escolha de seus elementos de comando e pela interligação entre eles. A localização dos diversos aparelhos no quadro de comando e sua respectiva fiação, dependem neste caso da função a ser executada.

Um comando programado por fiação somente pode ser montado mecânicamente e eletricamente depois que todo o esquema elétrico esteja definido. Qualquer alteração posterior na lógica de funcionamento implica obrigatoriamente na alteração da fiação e possivelmente na quantidade dos aparelhos (contatores, temporizadores, relés, etc) no painel elétrico.

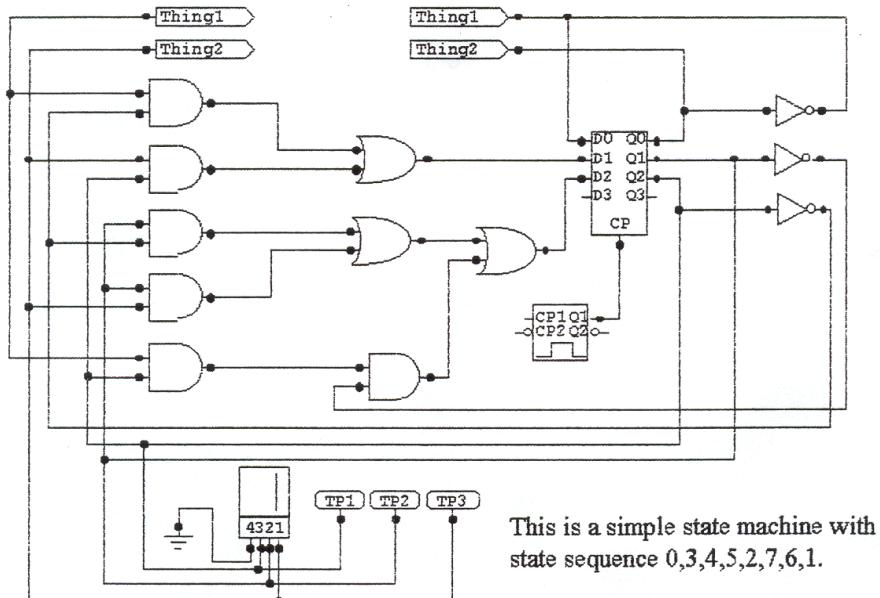


## COMANDOS PROGRAMADOS POR MEMÓRIA

Entre os aparelhos usados para comandos, cujo programa é armazenado em memória, destacam-se os Controladores Programáveis.

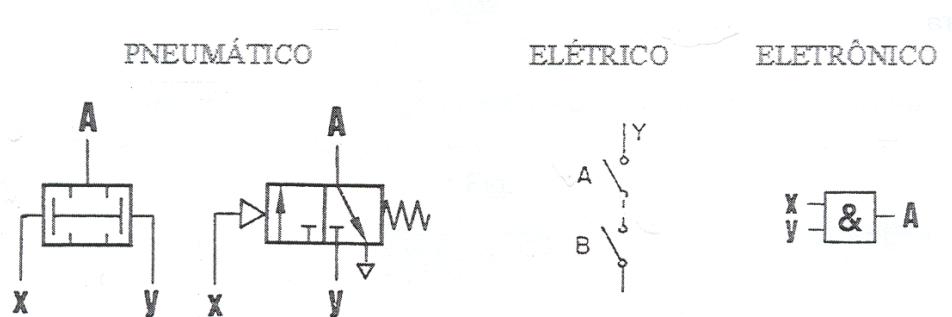
A função dos comandos programados por memória não depende unicamente da configuração mecânica de seus componentes e da respectiva interligação entre eles. Os controladores programáveis são ligados a elementos de comandos como, botoeiras, chaves fim de cursos, sensores, etc., e a elementos operadores como válvulas, contatores, indicadores, etc.

A função deste comando, isto é, a tarefa que ele deve realizar, não é apenas definida pela maneira como os elementos estão interligados, mas sim, pelo programa gravado na memória do controlador.



## INTRODUÇÃO A PORTAS LÓGICAS

Denominamos de portas lógicas todo e qualquer arranjo físico mecânico, elétrico, pneumático etc, capaz de efetuar uma operação lógica.



### Operação Lógica

A relação entre duas ou mais variáveis que representam estados binários, é estabelecida através de operações lógicas que são realizadas conforme os princípios da álgebra de booleana.

### Funções Lógicas

É uma variável binária cujo valor, depende do valor de uma operação na qual se relacionam entre si duas ou mais variáveis binárias.

Exemplo: uma lâmpada só estará acesa (condição verdadeira), se duas condições sejam satisfeitas:

- A – lâmpada esteja boa
- B – o interruptor esteja ligado

**Tabela Verdade**

O estabelecimento de uma tabela verdade é, em geral, o primeiro passo para análise e compreensão de uma função lógica de todas as combinações possíveis, de todas as variáveis da função, incluindo o estado resultante de cada combinação.

Regra:

1. O número de colunas de uma tabela-verdade é igual ao número de variáveis, adicionada a uma coluna de saída relativa aos valores resultantes.

Exemplo:

Lâmpada boa = variável “A”  
Interruptor ligado = variável “B”  
Lâmpada acesa = saída “C”

2. O número de combinações possíveis de  $n$  variáveis é  $2^n$ . O número de combinações determina o número de linhas da tabela verdade.

Exemplo:

Lâmpada boa = variável “A”  
Interruptor ligado = variável “B”  
Lâmpada acesa = saída “C”

Tabela verdade

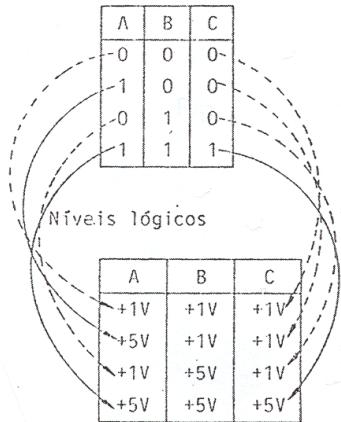
### Situação problema

O sistema de segurança de uma fonte de alimentação será acionado quando a carga aplicada à saída solicitar corrente excessiva da fonte, quando a temperatura do sistema subir a níveis indesejáveis, ou quando a tensão ultrapassar o valor limite permitido.

Elabore a tabela-verdade

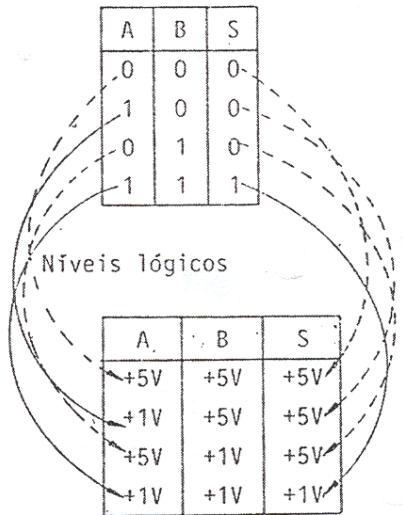
## Lógica Positiva

Temos lógica positiva quando o valor da tensão que representa o estado lógico 1, for positivo em relação ao valor da tensão que representa o estado lógico 0.



## Lógica Negativa

Temos lógica negativa quando o valor da tensão que representa o estado lógico 1, for negativo em relação ao valor da tensão que representa o estado lógico 0.



## Portas Lógica Básicas

NOME	ASA (variante 2)	SÍMBOLO	
	DIN (40700)	ABNT	
Porta "E"			
Porta "OU"			
Porta "NÃO"			
Porta "SIM"			

**Porta E:** Dados de uma porta E

**Porta E:** "Se a entrada A e a entrada B receberem nível 1, a saída apresenta nível 1"

Entradas	Saída	Símbolo adotado no Brasil	Símbolo adotado nos EUA
A	B	S	
0	0	0	
0	1	0	
1	0	0	
1	1	1	

**Porta OU:** Dados de uma porta OU

Porta OU: "Se a entrada A ou a entrada B receberem nível 1, a saída apresenta nível 1"		
Entradas	Saída	
A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

BRASIL                                    EUA

NOME	SÍMBOLO		
	ASA (variante 2)	DIN (40700)	ABNT
Porta "NÃO E"			
Porta "NÃO OU"			
Porta "OU EXCLUSIVO"			
Porta "EQUIVALÊNCIA"			

Função	Representação algébrica	ASA	DIN	ABNT	Equivalente elétrico	Equivalente pneumático
"E"	$Y = A \cdot B$					
"OU"	$Y = A + B$					
"NÃO"	$Y = \bar{A}$					
"SIM"	$Y = A$					
"NÃO E"	$Y = \overline{A \cdot B}$ $Y = \bar{A} + \bar{B}$					

Função	Representação algebrica	AS A	DIN	ABNT	Equivalente elétrico	Equivalente pneumático
"NÃO-OU"	$Y = \overline{A+B}$ $Y = \overline{A} \cdot \overline{B}$					
"OU EXCLUSIVO"	$Y = A \oplus B$ $Y = \overline{A}B + A\overline{B}$					
"IDENTIDADE OU EQUIVALÊNCIA"						

## INTRODUÇÃO A CONTROLADORES LÓGICOS PROGRAMÁVEIS

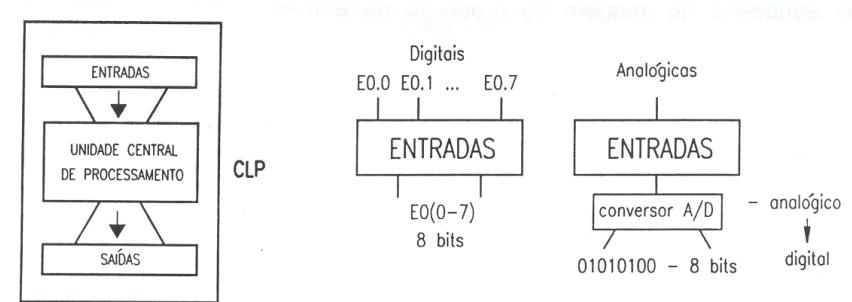
São equipamentos eletrônicos utilizados em sistemas de automação flexível, e permitem desenvolver e alterar com facilidade a lógica para acionamento das saídas em função das entradas, possibilitando inúmeros pontos de entrada de sinal para controlar pontos de saída.

As vantagens dos controladores lógicos programáveis (CLPs), em relação aos sistemas convencionais são:

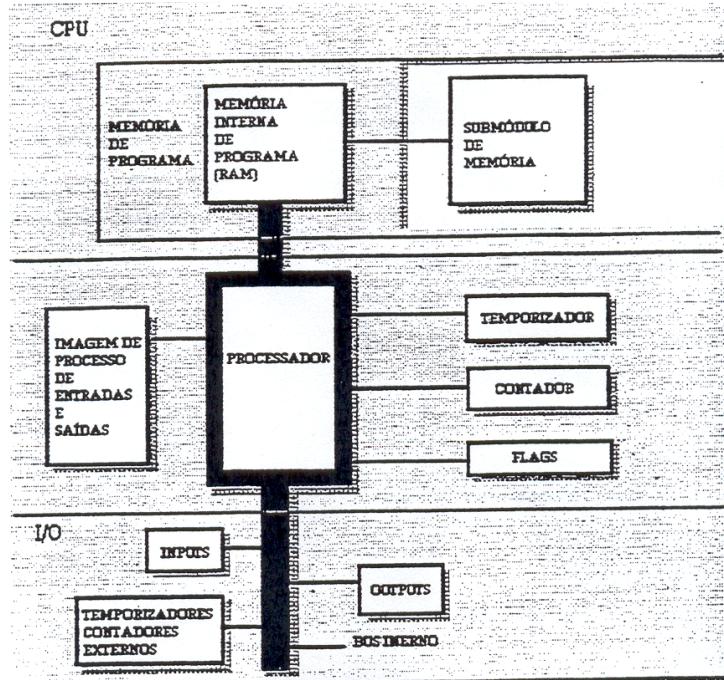
- Ocupam menos espaços;
- Requerem menor potência elétrica;
- Podem ser reutilizados;
- São programáveis;
- Possui maior confiabilidade;
- Oferecem maior flexibilidade;
- Permitem interface de comunicação;
- Maior rapidez.

### Princípio de funcionamento

Podemos dizer que o CLP é um microcomputador aplicado ao controle de um sistema ou de um processo. É composto de módulos de entrada digitais ou analógicas. As entradas digitais são agrupadas em conjuntos de 8 a 16 bits, de forma que a unidade central de processamento (CPU), possa tratar as informações como bytes ou word.



## Arquitetura Básica do CLP



### Memória de Programa

É o local onde está armazenado todo o programa que determina a seqüência da tarefa a ser executada. Esta memória interna é denominada de RAM.

Processador:

No início de cada ciclo, o processador lê o estado de sinal de cada entrada e os armazena na imagem do processo de entrada durante a execução do programa. Enquanto o programa está sendo executado, o processador lê todas as posições da memória, uma após outra, começando do início do programa.

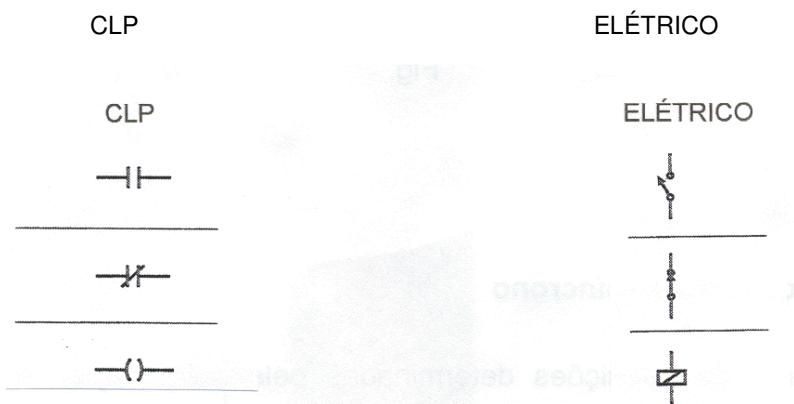
O processador efetua as operações lógicas e aritméticas de acordo com os dados que a imagem do processo das entradas fornecem. Caso uma saída deva ser ativada, seu estado de sinal é armazenado na imagem do processo de saída. Ao final do ciclo, o processador transfere a informação para as saídas físicas. O tempo de ciclo é basicamente 2 ms para 1024 instruções.

## Módulos de Entrada / Saída (I/O)

São classificados em digitais, analógicos, inteligentes (independentes da CPU), de comunicação, contadores e temporizadores.

Como o CLP veio substituir os componentes eletroeletrônicos de acionamento, a linguagem utilizada na sua programação, é similar à linguagem dos diagramas lógicos de acionamento, desenvolvidas por eletrotécnicos, eletricistas, e pessoal da área de controle.

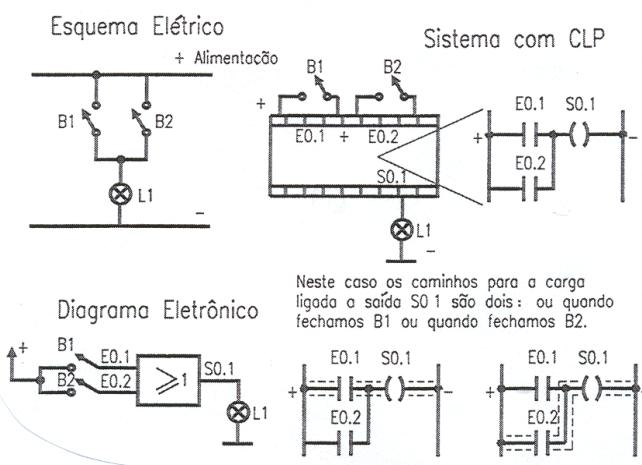
### Comparação / Símbolos de Programação



Nos CLPs, é possível desenvolver lógicas combinatórias, seqüenciais e associadas.

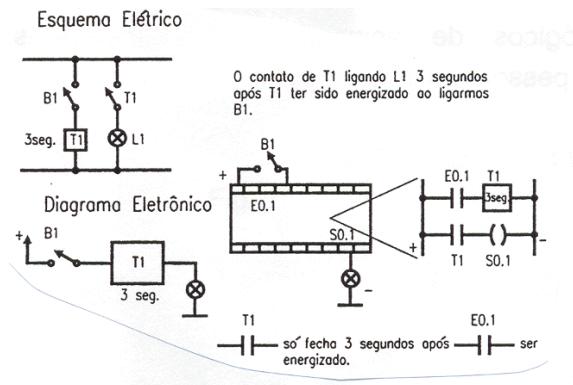
### Lógicas Combinacionais Básicas

Num circuito combinatório, o estado lógico assumido pela saída, num dado instante, depende unicamente do estado das entradas naquele momento, não dependendo dos estados anteriores.



## Lógica Seqüencial Básica

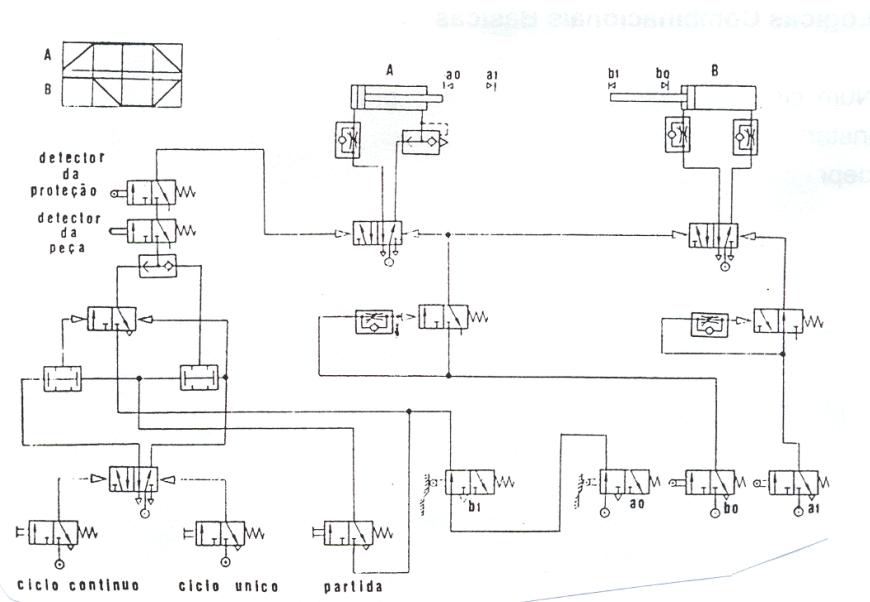
Num circuito seqüencial, o estado lógico assumido pela saída, num dado instante depende das variáveis de entrada naquele momento, dependendo também dos estados anteriores. Os circuitos seqüenciais são classificados em assíncrono e síncrono.



## Sistema Seqüencial Assíncrono

Evolui a partir de transições determinadas pela combinação dos estados internos das variáveis.

Exemplo: Num conjunto pneumático, um cilindro só é acionado quando um segundo cilindro volta para posição inicial.



## Sistema Seqüencial Síncrono

Evolui a partir de transições dos estados internos das variáveis, combinados com pulsos de um gerador de base de tempo (pulso de clock).

Exemplo: numa impressora matricial, o acionamento das agulhas só ocorre após o posicionamento do cabeçote, dependendo ainda da chegada de um pulso clock.

## Estrutura de Programação

Para solucionar tarefas complexas se faz necessário dividi-la em pequenas partes. Estas partes serão denominadas de blocos de programa, que irá executar uma parte da tarefa, que serão gerenciadas por um único bloco de organização. As programações estruturadas dos CLPs, dependem da sua autonomia, família e fabricante.

A linguagem de programação do STEP 5 (SIEMENS), apresenta os seguintes tipos de blocos, para formação de um programa estruturado.

- **Bloco de Organização (OB)**

Organiza os blocos de controle, pode ser considerado o Programa Principal (OB1). São numerados de OBO à OB255;

- **Bloco de Programa (PB)**

São utilizados para a programação das partes da tarefa a ser executada. Podem ser numerados de PB0 a PB255;

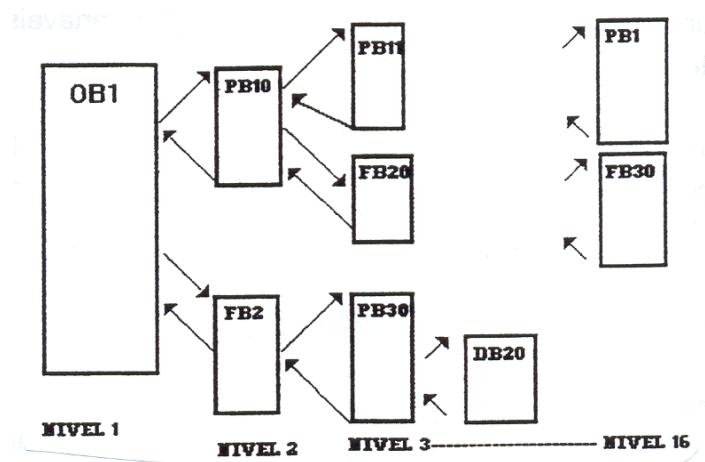
- **Blocos de Funções (FB)**

São utilizados quando uma parte da tarefa exige operações avançadas, tarefas repetitivas no programa, ou suplementares. Podem ser numerados de FB0 a FB255. Existem FBs que são padronizados para determinadas tarefas com PID, posicionamento, conversão numérica, etc.

- **Blocos de Dados (DB)**

São áreas de memória, onde podem ser armazenados dados. Dados estes que poderão ser utilizados pelos FBs, temporizadores, contadores, comparadores, etc.

## Arquitetura do Programa do STEP 5



## MÉTODOS DE REPRESENTAÇÃO

Cada método de representação apresenta suas próprias características. Um bloco de programa criado em STL, não necessariamente poderá ser convertido em CSF ou LAD. Os três métodos de representação não são compatíveis. Entretanto, programas em CSF ou LAD poderão sempre ser convertidos para STL.

### Representação STL (Lista de Instruções)

Assemelha-se a instruções escritas em mônicos, e apresenta o seguinte formato.

Exemplo:

**002 : A I 32.0**

Onde:

002 = endereço relativo

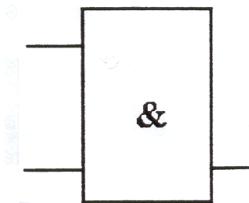
A = operação (comando lógico)

I = identificação do operando (entrada)

32.0 = parâmetro do operando (endereço)

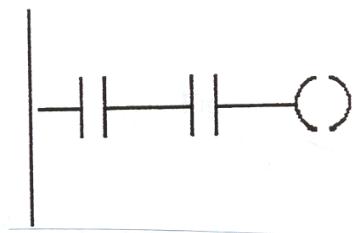
## Representação CSF (Diagrama de Blocos de Funções)

Assemelha-se a blocos de funções digitais.



## Representação LAD (Diagrama de Contatos)

Esta representação é a mesma utilizada em contatos de relês.



A linguagem de programação do STEP 5 apresenta três níveis de operações:

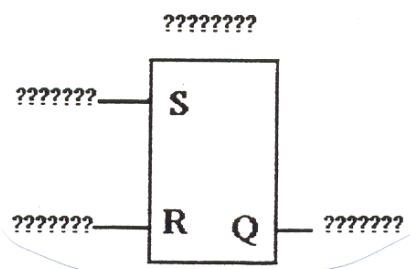
**Básica:** utilizada em todos os blocos de programação, representada em CSF/LAD/STL;

**Suplementar:** utilizada somente em FBs e somente representada em STL;

**Avançada:** utilizada somente em FBs e somente representada em STL.

## Operação de SET/RESET

Representado pelas letras S (SET), onde indica a memorização do estado 1 para flags e saídas, R (RESET), indica que o estado memorizado pela função set esta desativado, ou seja, coloca em zero o estado de sinal das saídas e ou flags.



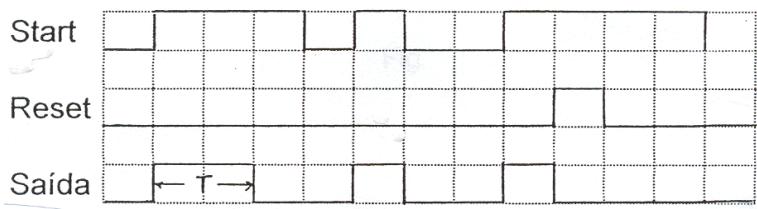
## TEMPORIZADORES

Os CLPs da família SIMATIC 5 apresenta cinco tipos de temporizadores:

### SP (temporizador para pulso)

Com a entrada de start em 1, inicia-se o tempo de contagem do temporizador. A saída da imediatamente vai para 1. Decorrido o tempo programado, a saída vai para 0.

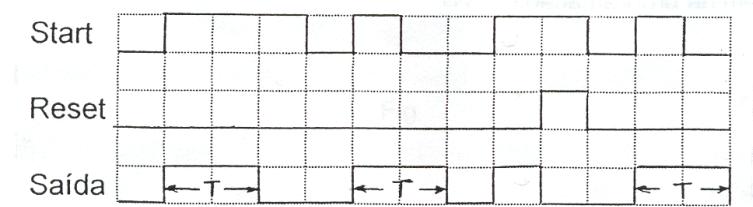
Diagrama de Tempo:



### SE (temporizador para prolongamento de sinal)

Com a entrada e start em 1, a saída fica em 1 durante o período de contagem do temporizador, independente da entrada voltar para zero ou não.

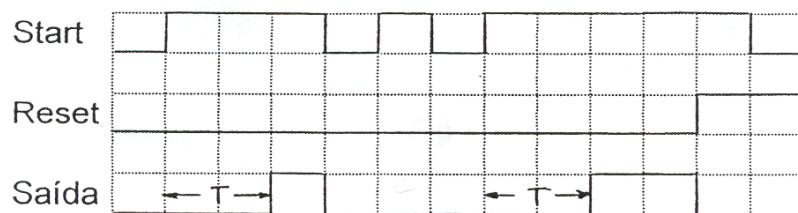
Diagrama de Tempo:



### SD (temporizador para retardo na ligação)

Se a entrada de start permanecer em 1 por um período de tempo superior ao tempo programado, a saída vai para 1 após o período programado, ou seja, a ligação da saída é retardada em relação à entrada.

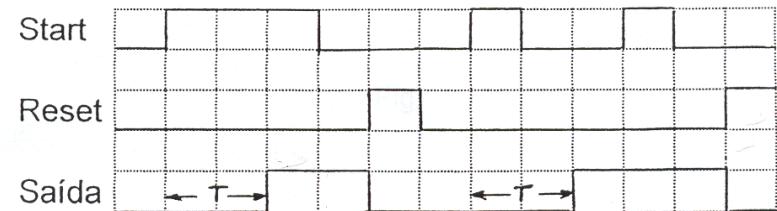
Diagrama de Tempo:



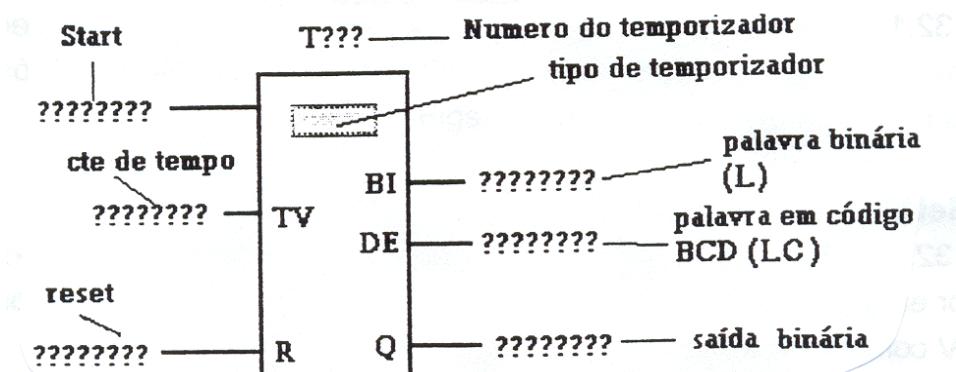
**SS (temporizador para retardo na ligação com retenção)**

Ao ser acionada a entrada de start, o mesmo conta o seu tempo programado e aciona a saída com retardo na ligação. O temporizador permanece com a saída até o sinal de reset.

Diagrama de Tempo:

**SF (temporizador para retardo no desligamento)**

Ao ser acionado a entrada de start, a saída do temporizador vai para 1. O tempo programado começa a contar a partir do momento em que a entrada start é zerada. Ao ser acionada a entrada de reset, a saída vai para zero imediatamente.

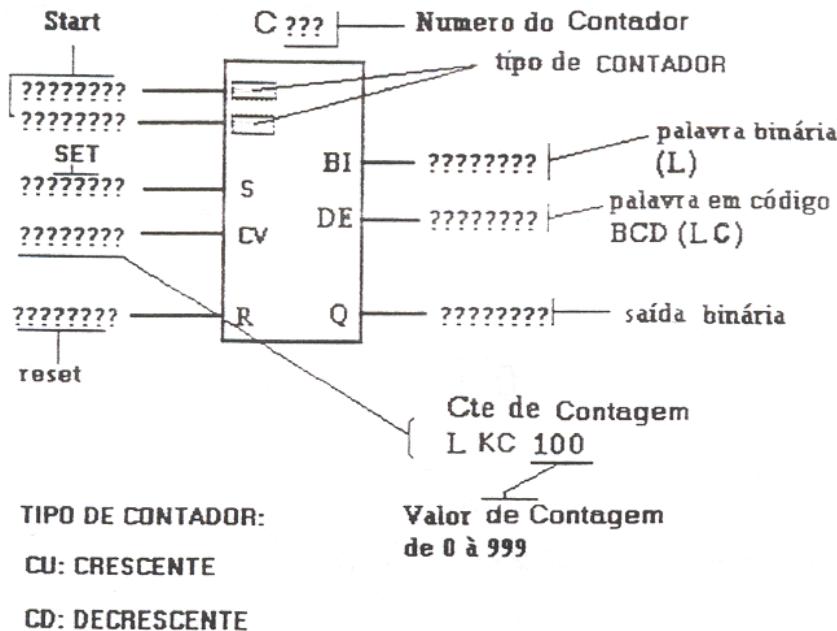


Forma de Programar:

Ex.: KT 100.1 (significa: constante de tempo 100 x 0.1 segundos)

Tabela de Multiplicação:	0 = 0,01 seg.
	1 = 0,1 seg.
	2 = 1 seg.
	3 = 10 seg.

## CONTADORES



### Funções de cada entrada e saída

#### **CU – (Counter UP) Incrementar Valor de Contagem**

Ex.: I 32.0 na Entrada CU. A cada pulso nesta entrada, o contador incrementa 1 no seu valor de contagem. O valor máximo de contagem é 999. Após isso, os pulsos que chegarem serão desprezados.

#### **CD – (Counter Down) Decrementar Valor de Contagem**

Ex.: I 32.1 na Estrada CD. A cada pulso nesta entrada, o contador decrementa 1 no seu valor de contagem. O valor mínimo de contagem é 000. Após isso, os pulsos que chegarem serão desprezados.

#### **S – (Set Counter) Carregar Contador**

Ex.: I 32.2 na Entrada S. Um sinal nesta entrada, carrega o Acumulador 1 com o valor estabelecido em CV (Count Value). O contador assume o valor definido em CV como valor inicial de contagem.

#### **CV – (Counter Value)**

Ex.: KC 100 (Constante de contagem = 100). Ao chegar um sinal na entrada de Set (S), o contador transfere para o Acumulador 1, este valor definido em CV. Essa entrada é normalmente utilizada para efetuar a Contagem em modo decrescente. Ex.: Seta o valor de contagem em 100 e a cada pulso na entrada CD, decrementa 1 no contador.

**BI – (Em Binário)**

A leitura do valor de contagem é feita em binário e armazenada numa posição de memória, que pode ser: FW 10 (FLAG WORD 10) ou QW (OUTPUT WORD 10).

A rotina de transferência do valor em binário é feita como descrito abaixo:

Carrega no Acumulador 1 o valor de contagem em binário.

Transfere o conteúdo do Acumulador 1 para FW ou QW 10. (Posição de memória 10).

**DE – (BCD)**

A leitura do valor de contagem é feita em BCD e armazenada numa posição de memória, que pode ser: FW 10 (FLAG WORD 10) ou QW (OUTPUT WORD 10).

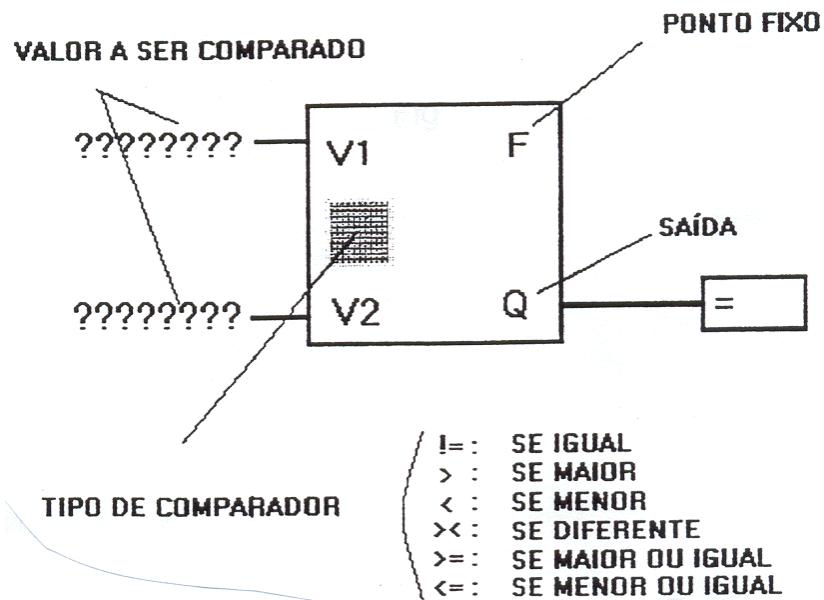
A rotina de transferência do valor em BCD é feita como descrito abaixo:

Carrega no Acumulador 1 o valor de contagem em BCD.

Transfere o conteúdo do Acumulador 1 para FW ou QW 12 (Posição de memória 12).

**COMPARADORES**

As entradas V1 e V2 são comparadas com valores constantes ou valores de memória.



## PROGRAMAÇÃO EM STEP 5

### CRIAÇÃO DO ARQUIVO DE PROJETO

1. Crie um diretório no DOS

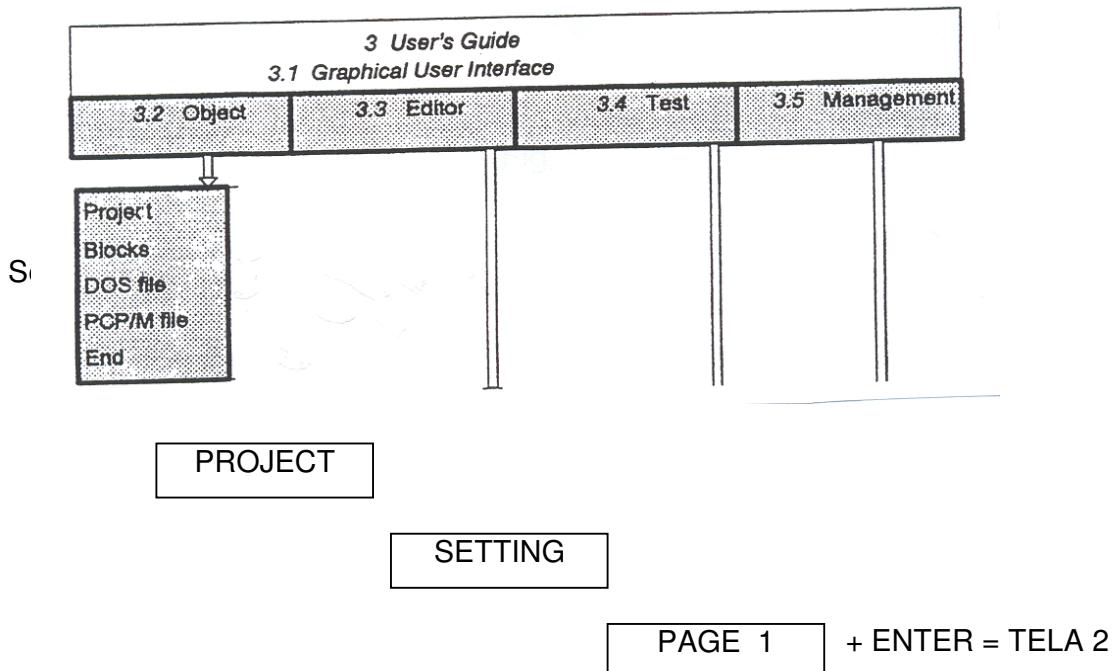
C:\>MD  + ENTER

2. Carregue o STEP5

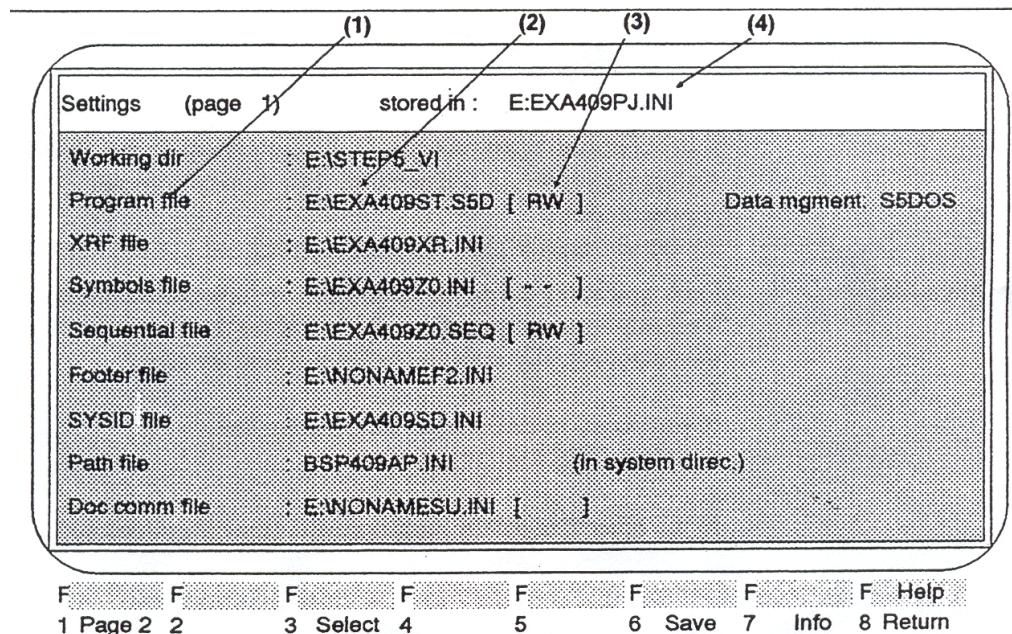
C:\> S5 + ENTER

3. Configurando as páginas 1 e 2.

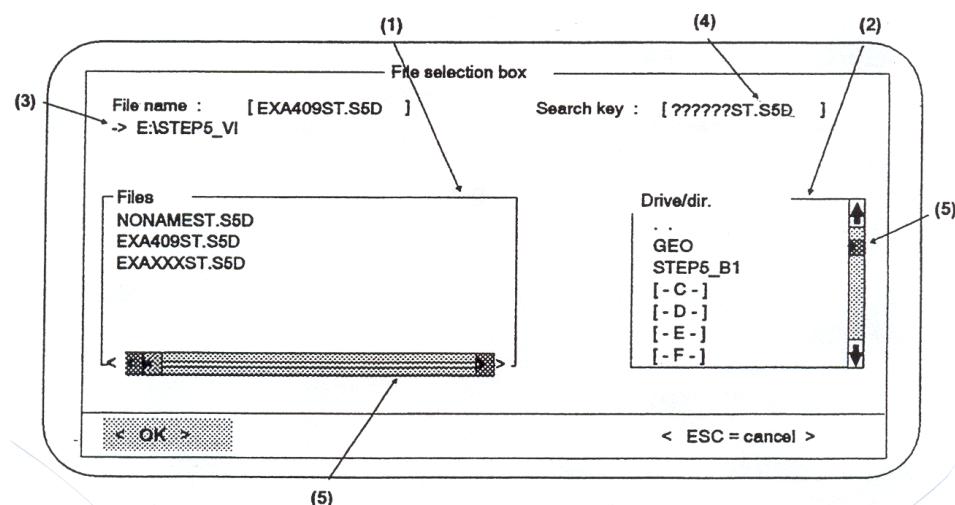
Tela inicial



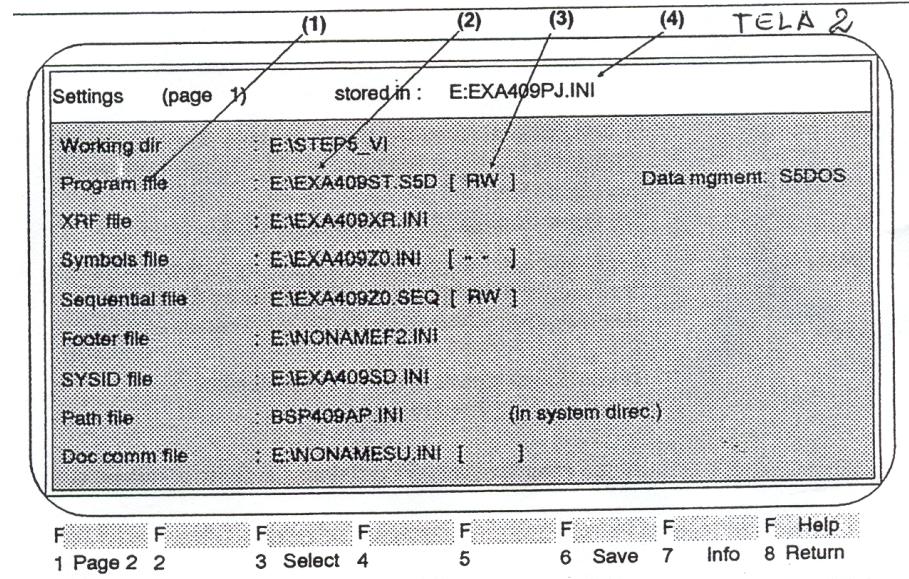
## Página 1



1. Selecione o diretório de trabalho Working Dir – tecla F3-select.  
Surgirá uma nova janela.



Example of a file selection box



Confirmado o Working dir, obedeça aos seguintes procedimentos:

### **ABERTURA DE ARQUIVO**

Coloque o cursor em Program File, +F3, em seguida digite o nome do arquivo (exemplo máquina).

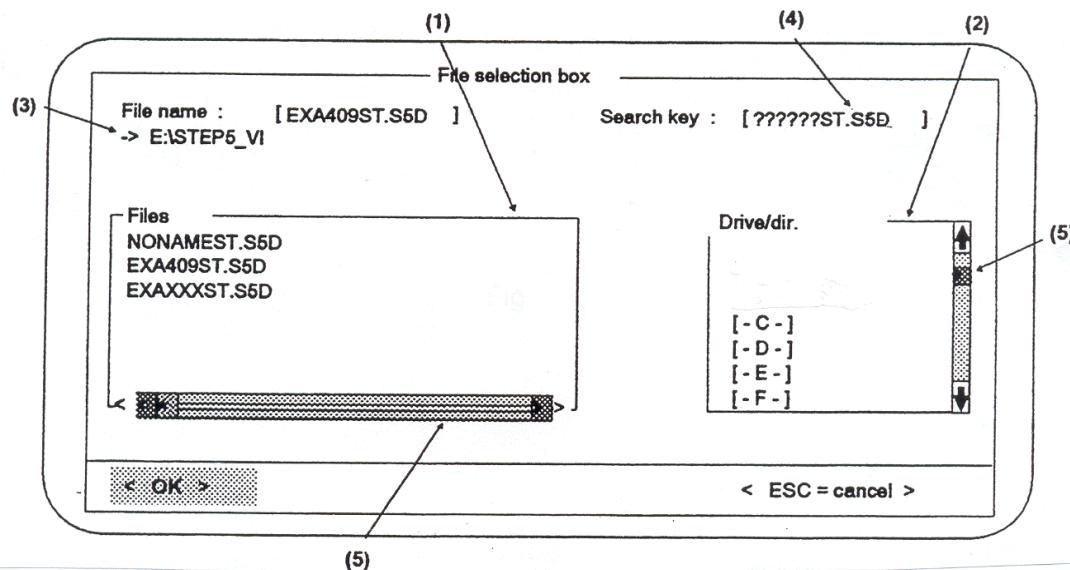
C:\ MÁQUINA @ + ENTER

Selecione Symbol File, +F3, em seguida digite o nome do arquivo (exemplo máquina).

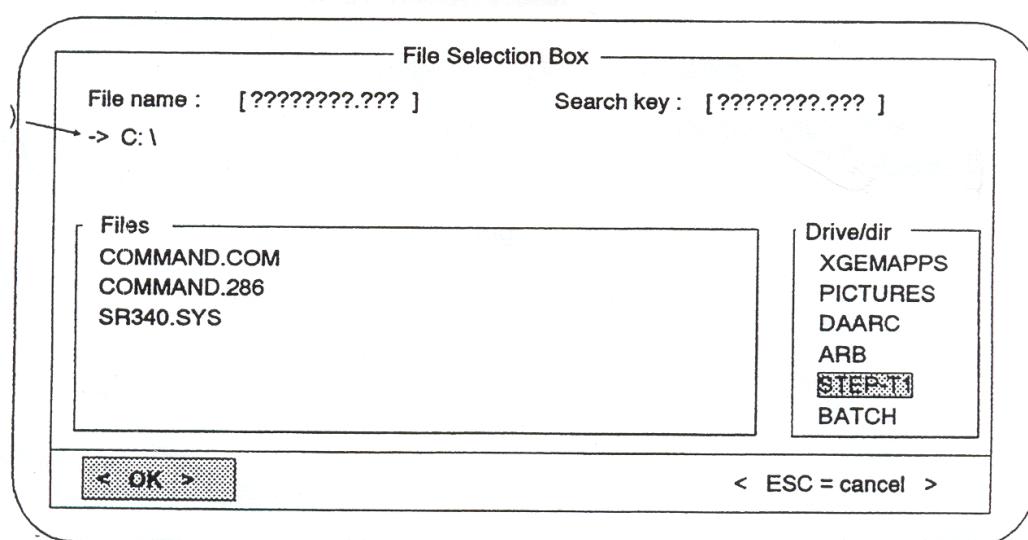
C:\ MÁQUINA @ + ENTER

Digite ESC, e retorne ao menu.

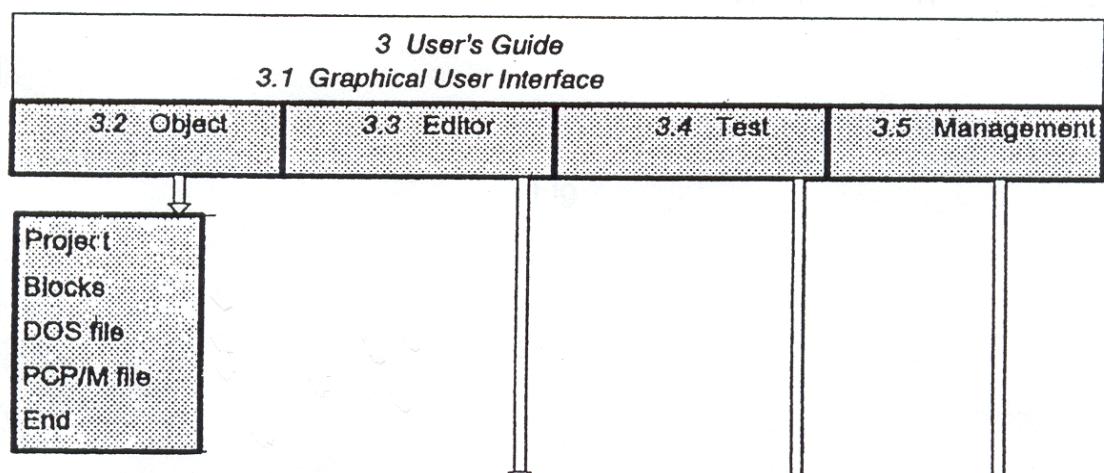
Com a tecla TAB, selecione com o cursor a opção DRIVE/DIR, posicione o cursos em [ - C - ] + ENTER.



Selecione o diretório criado (exemplo STEP – T1), + INSERT.



Retorne a tela 2 F3 + INSERT, e confirme a seleção do diretório em Working dir.



Selecione:

OBJECT			
PROJECT			
SETTING			
PAGE 2	+ ENTER		

Surgirá uma nova tela de módulo de configuração de operação.

<b>Setting ( page 2 )</b>	<b>in C:\ XXXX@ YY.INI</b>	
Mode: offline	[ ----- ]	Representation : LAD
PLC type:		
Interface : AS511		Checksum : no
Path name :		
Path file : NONAMEAP.INI	( in system direc )	
Symbols : No		Symbol length : 8
Display : ----		Comment length : 24
Comments: Yes		
Documentation ( X ) on printer		Char. Set : ASCII
( ) to file		Footer : no
Printer file : NONAMEDR.INI	( in system direc )	
Diagnosis : ---		

Procedimentos:

1. Configuração de operação

Modo: Offline (sem comunicação com o CLP)

Online (comunicação com o CLP)

A comutação de offline para online, e de online para offline, é executada através da tecla:

F 3

2. Representação:

Através da representação podemos escolher o tipo de linguagem a ser utilizada, STL, CSF, ou LAD. A seleção é feita utilizando a tecla de comando:

F 3

3. Symbol: YES (com comentários)

NO (sem comentários)

Tecla de comando:

F 3

4. Display: Abs (mostra o operando)

Sym (mostra o símbolo)

Tecla de comando:

F 3

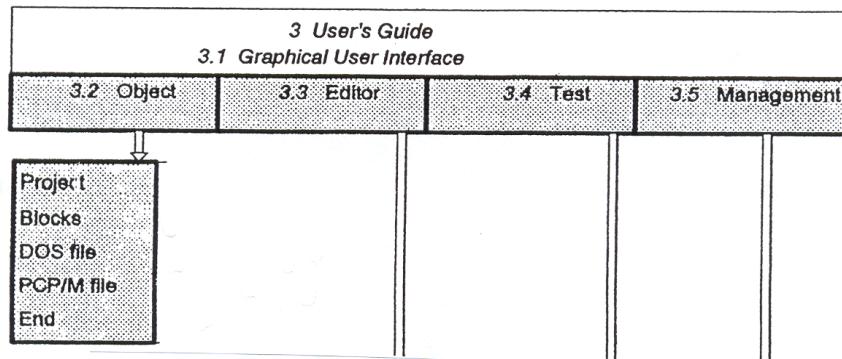
5. Comments: YES (com comentário)

NO (sem comentário)

Tecla de comando:

F 3

Retorne a tela anterior através da tecla INSERT.



Selecione:

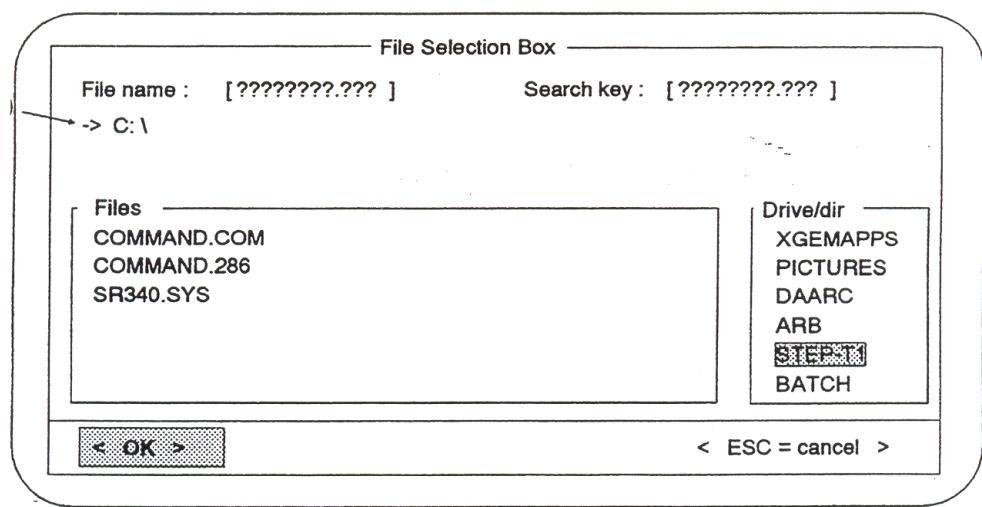
OBJECT

PROJECT

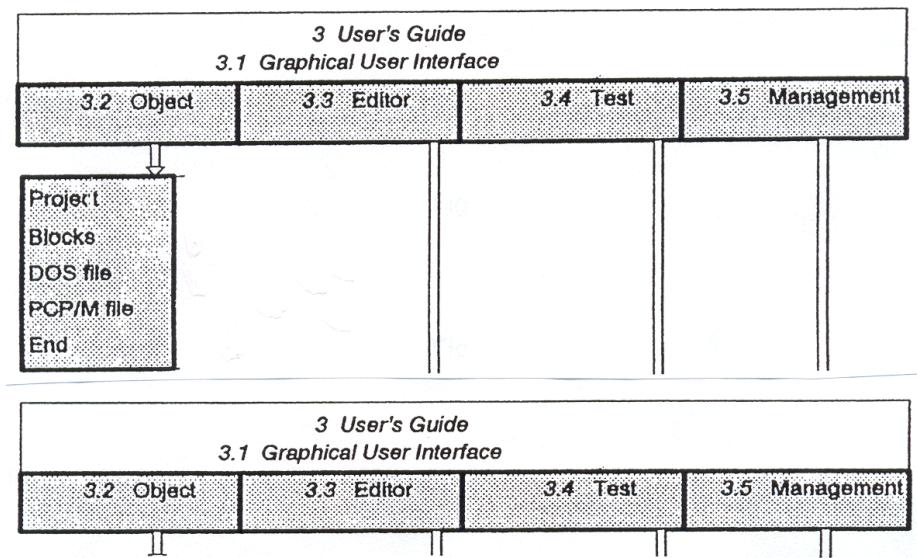
SAVE AS + ENTER

Surgirá uma nova janela:

Em file name: [?????????], digite o nome projeto + INSERT.



Através de ESC ou INSERT retorne a tela anterior.



Selecione:

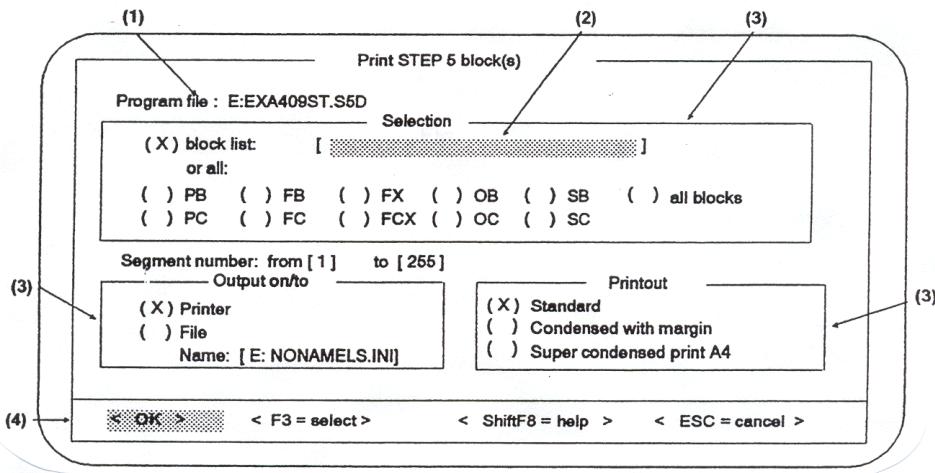
Editor
STEP 5 block

In the program file	Opções: + ENTER F1 Insert
---------------------	---------------------------------

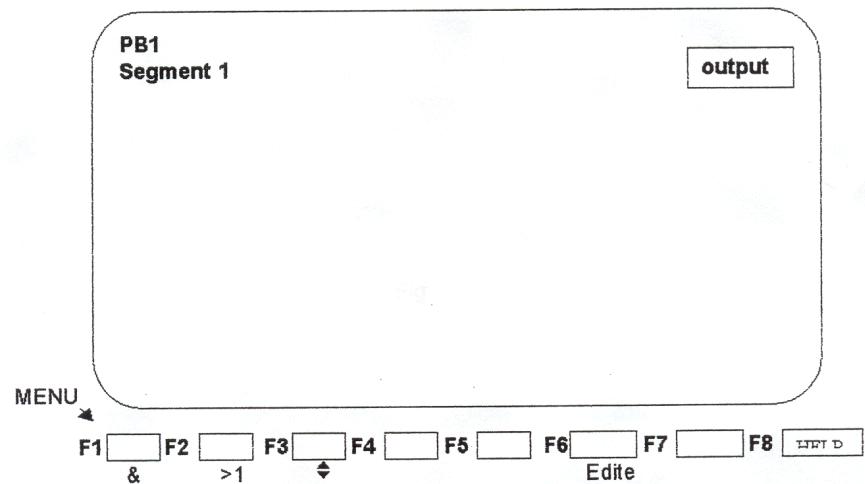
Surgirá uma nova janela:

Em 2, digite o bloco que vai trabalhar, obedecendo a seguinte ordem:

1. Em início de programa, começar digitando o Bloco de Organização (OB) + Insert;
2. Retorne para Editor, refaça a seleção anterior, novamente surgirá à mesma janela, e em 2, determine o Bloco de Programa (PB) a ser desenvolvido + Insert.



Surgirá uma nova tela:



Identificação da Janela:

Informa o PB que se está trabalhando (parte superior);  
 O segmento em que se está trabalhando (ex.: segmento 1);  
 Modo de trabalho, inicialmente output (nesse modo não podemos editar);  
 Menu de trabalho, localizado no final da tela.

Utilização do Menu:

O menu é composto das teclas de funções (F1 ..... F8), as informações contidas dentro dos retângulos quando forem utilizadas, deveram obedecer aos seguintes comandos:

Exemplo:

Solicitação de Help

Acione a tecla Shift (mantenha pressionada), e tecle F8.



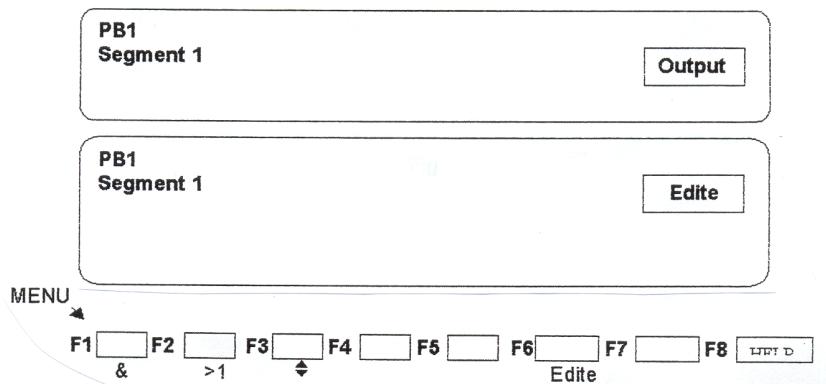
Quando as informações estiverem a baixo do retângulo, basta simplesmente acionar a tecla de função:

Exemplo: Solicitação de uma porta AND:

Basta acionar simplesmente: F1 = &

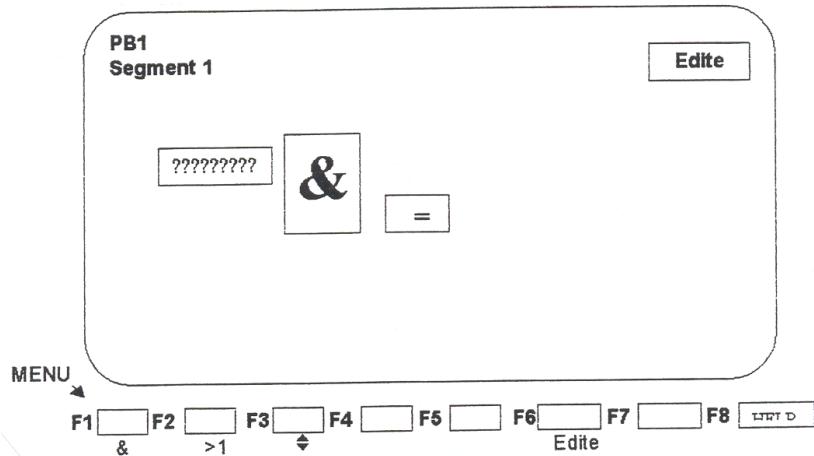
Saída de Output para condição de Edite:

Pressionar simplesmente a tecla F6



Inserir elementos lógicos no Bloco de Programa (BP):

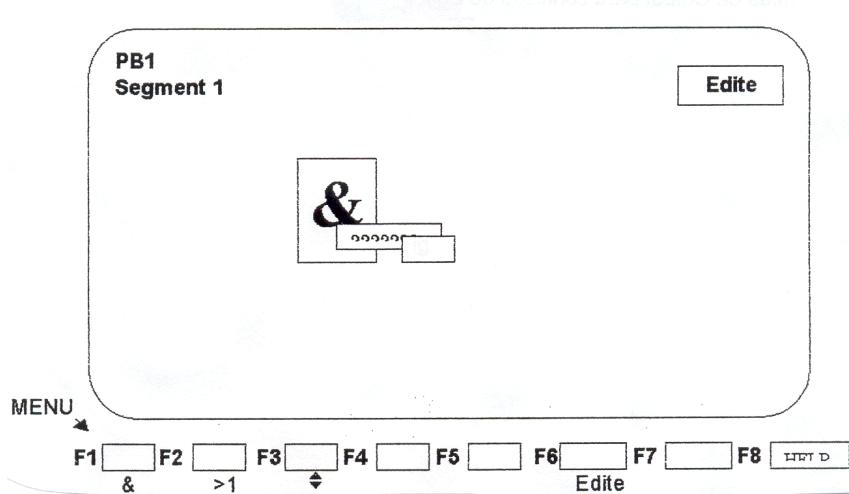
Exemplo: para inserir uma porta And, basta pressionar a tecla F1.



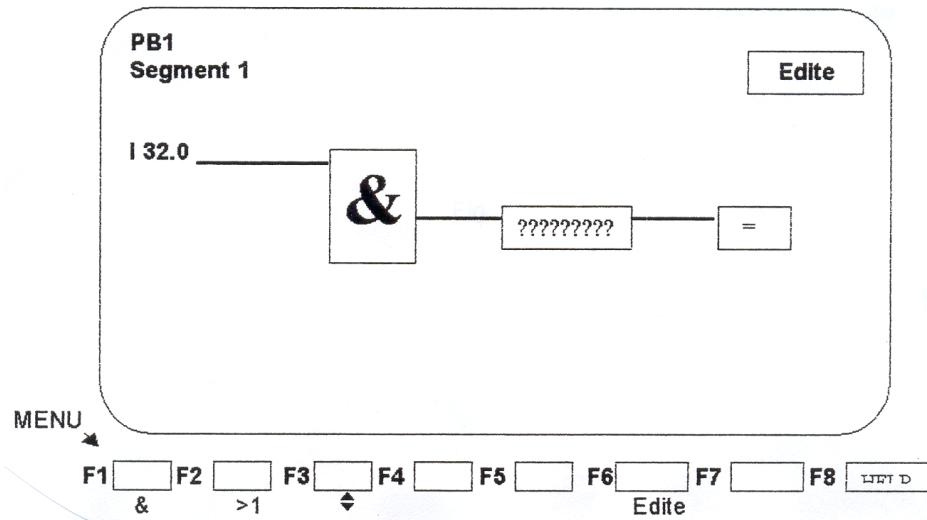
Inserir mais elementos lógicos no mesmo segmento:

Posicione o cursos entre o primeiro elemento e a saída e acione a tecla Home:

a) Posicionamento do cursor;

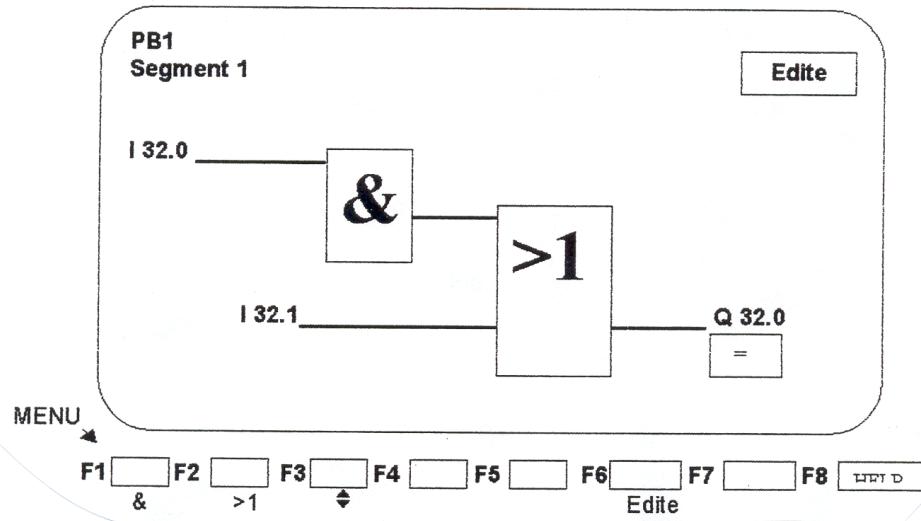


b) Após acionar a tecla Home.

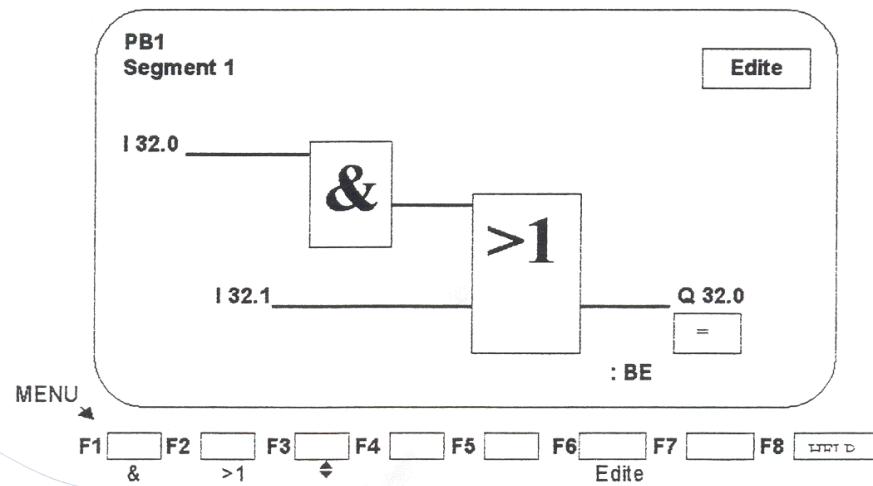


Selecione o novo elemento a ser inserido:

Exemplo: uma porta Or, precione a tecla F2

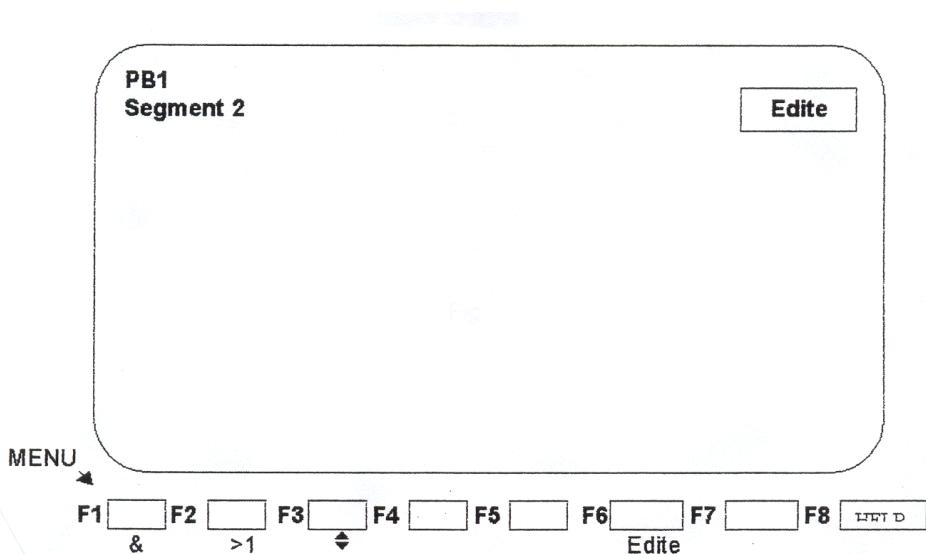


Para salvar o segmento após o seu fechamento, aione a tecla Insert até ser informado o final do bloco ( : BE).

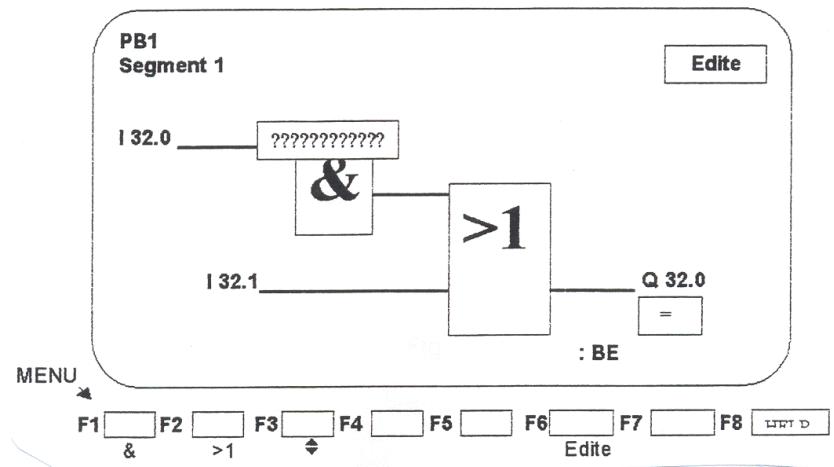


Abertura de novo segmento:

Para inserir um novo segmento precione a tecla F6.



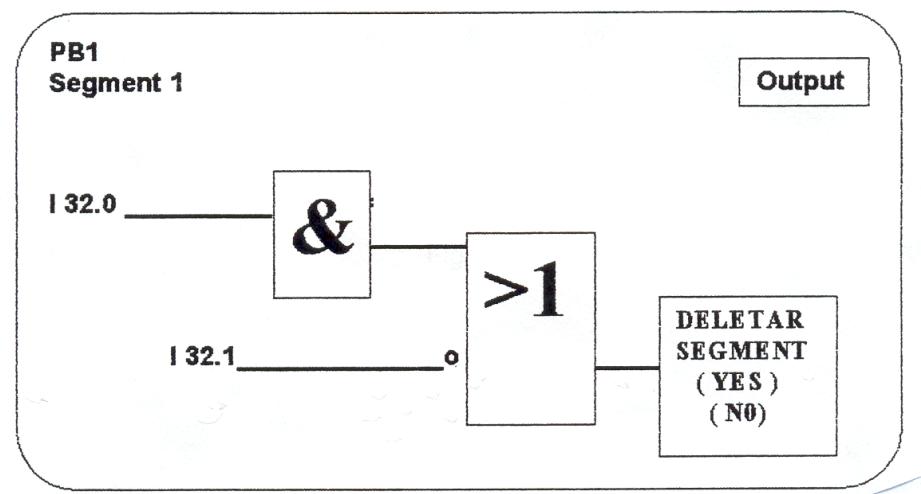
Para alterar um elemento lógico, posicione o cursor em cima da identificação do elemento a ser substituído, selecione o novo elemento, e acione a tecla correspondente.



#### Deletar segmento:

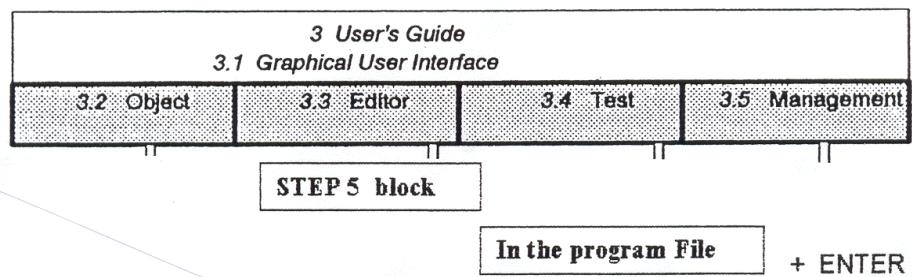
Para deletar um segmento em um bloco de programa utilize os seguintes comandos:

1. Sair de **Edite** para condição de **Output** através da tecla **Esc**;
2. Acione a tecla **F5**;
3. Acione a tecla **Shift + F4**;
4. Confirme o deletar com a tecla **Enter**.



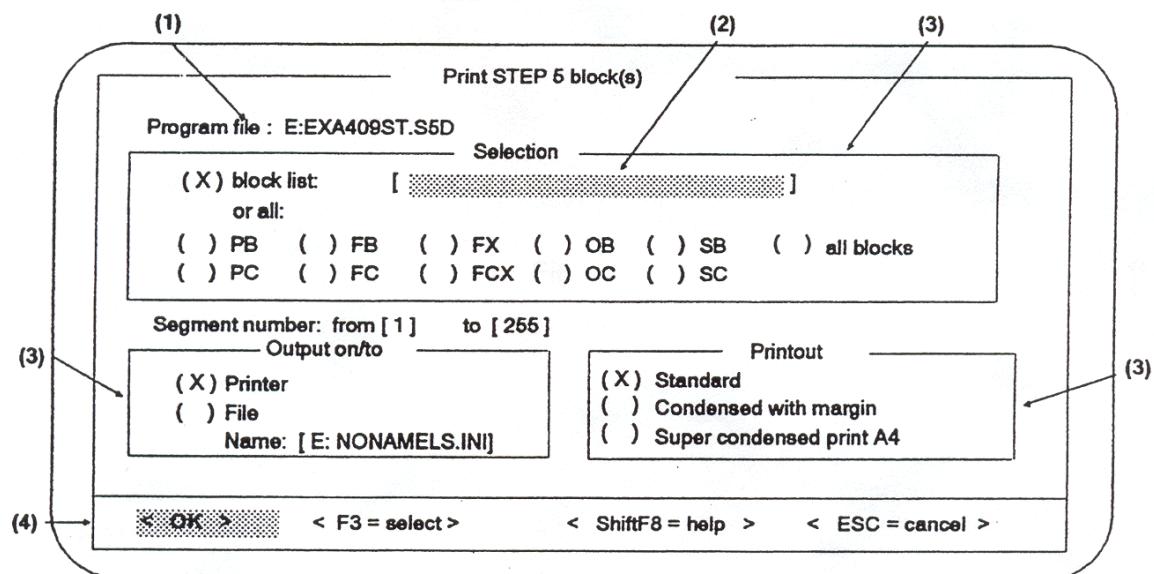
Interligação entre o PB e OB:

No final do Bloco de Programa, retorne a janela abaixo através da tecla **Insert**, e selecione:

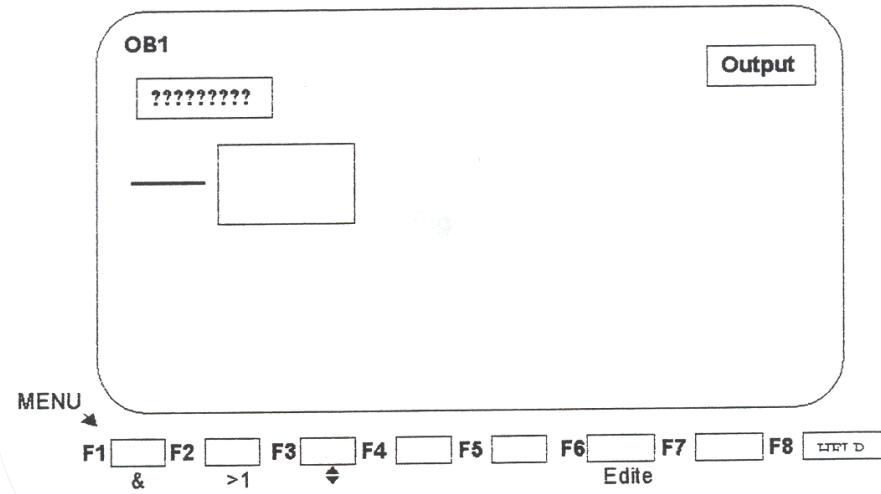


Nova janela:

Digite **OB1** em **2 + Insert**.



Nova janela:



Nova janela:

Procedimentos:

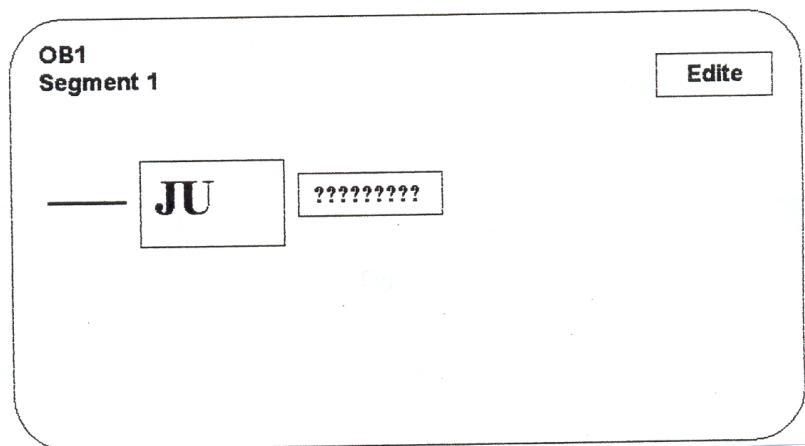
Sair da condição **Output** para **Edite** = tecla **F6**;

Solicitar segmento = tecla **F6**;

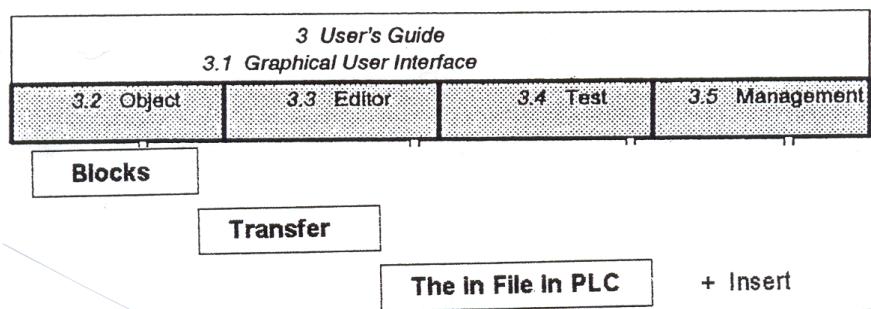
Solicitar o **JU** através de **block** = **Shift + F2**;

Acione a tecla **F4**;

Nova janela, digite o **PB + Insert**.



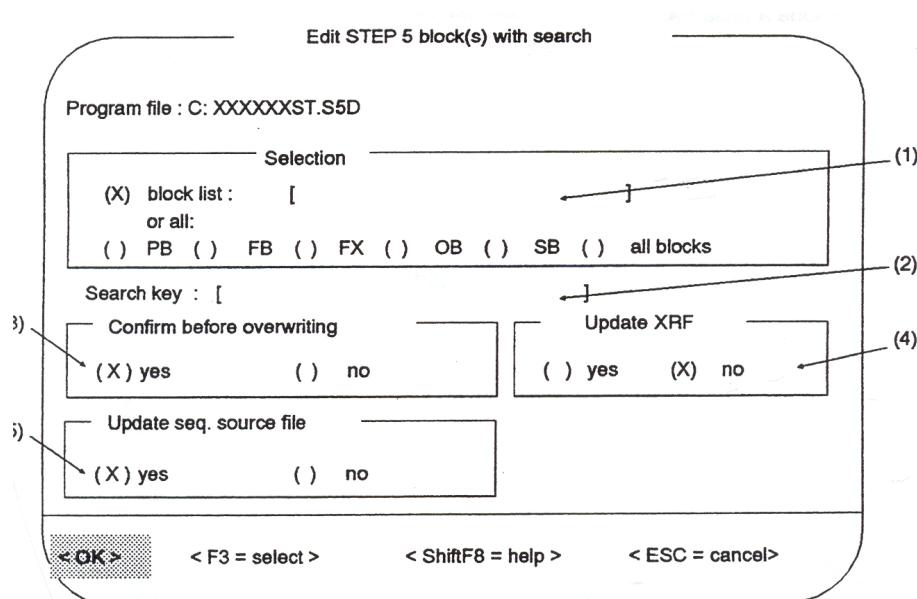
Retorne a tela inicial = **Esc**, selecione:



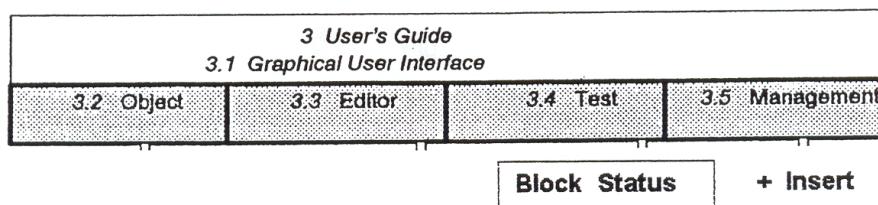
Nova janela:

Procedimentos:

1. Em 1 digite **OB**, e o **PB + Insert**;
2. Aguarde a transferência + **Enter**;
3. Acione a tecla **Esc**.



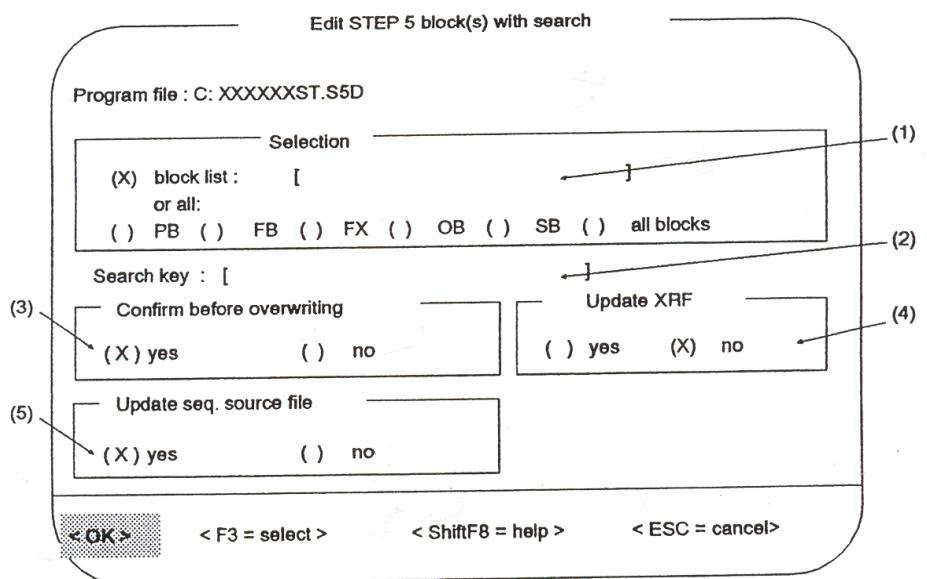
Selecione:



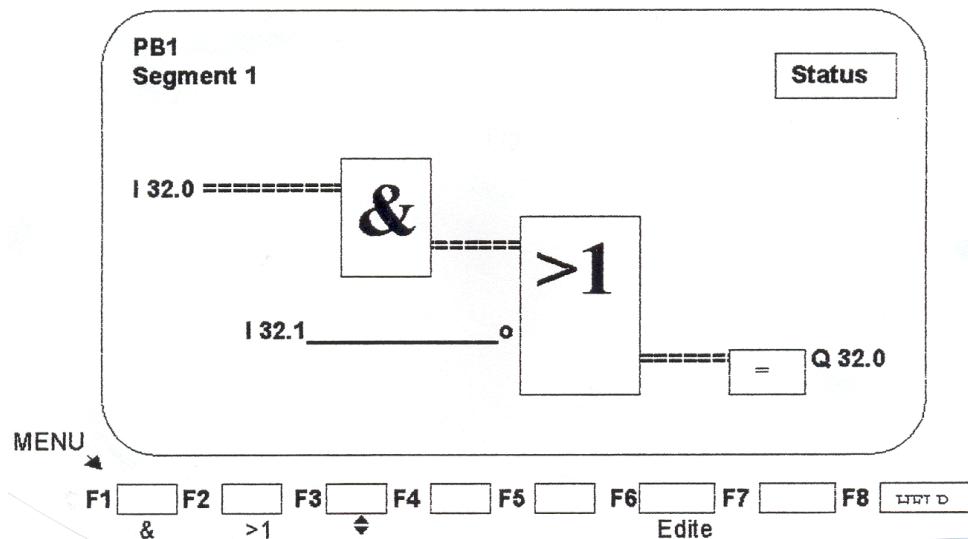
Nova janela:

Procedimentos:

1. Em 1 digite **PB + Insert**;

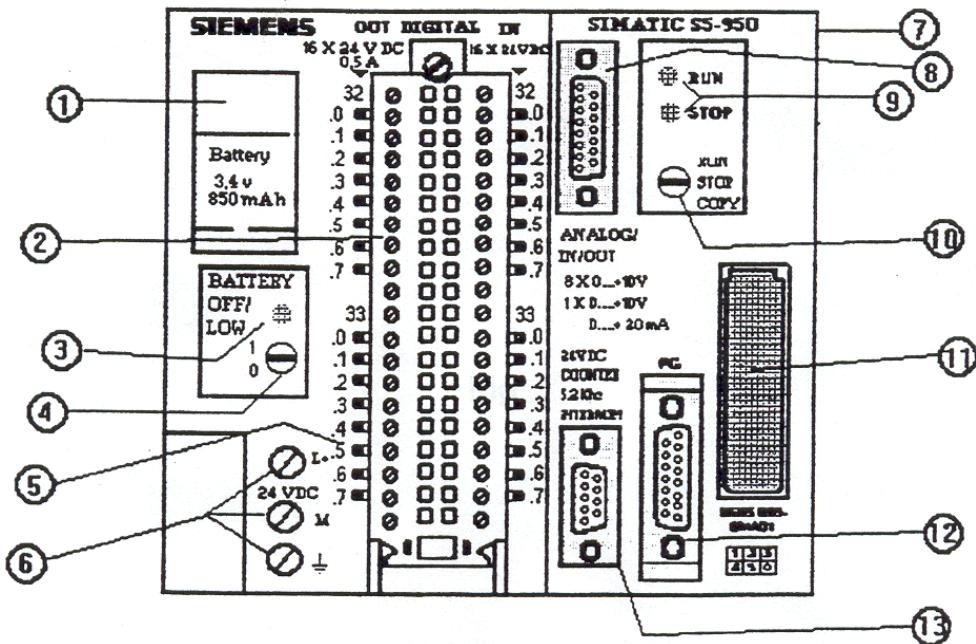


A condição de Status indica todos os sinais de acordo com os acionamentos:



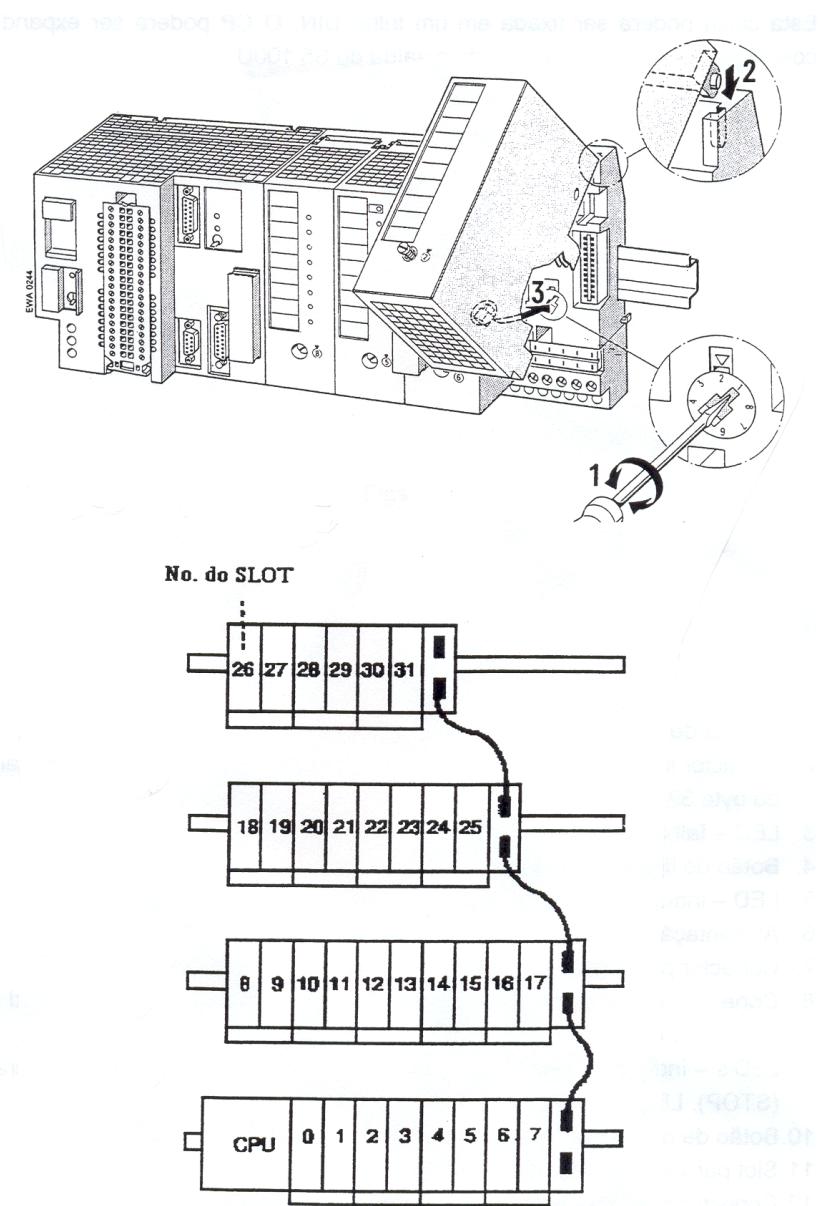
### Características do CP S5U:

O CP S5 95 U consiste principalmente de uma resistente caixa plástica na qual o processador, a fonte de alimentação, as entradas e saídas estão integradas. Esta caixa poderá ser fixada em um trilho DIN. O CP poderá ser expandido com todos os módulos de entrada e saída do S5 100U.



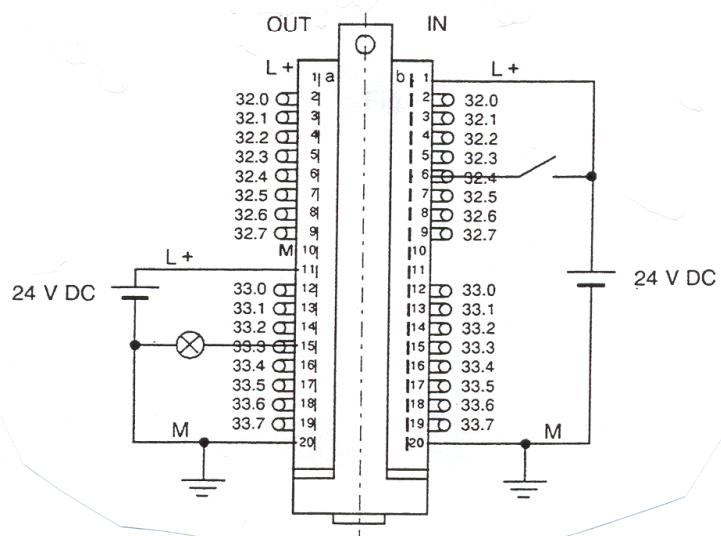
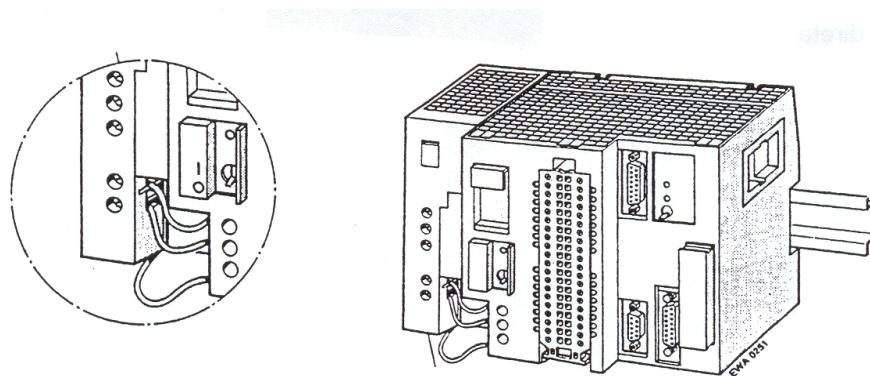
1. Bateria de Back-up.
2. Conector frontal (entradas digitais – do byte 32 ao byte 33; saídas digitais – do byte 32 ao byte 33).
3. LED – falha de bateria.
4. Botão de liga e desliga.
5. LED – indica o estado de sinal das entradas e saídas digitais.
6. Alimentação de 24 Vcc.
7. Conector para ligação dos cabos dos módulos periféricos.
8. Conector de entrada e saídas analógicas (entradas da word 40 a word 50; saída na word 40).
9. LED's – indicam o estado de ciclo (RUN), LED VERDE e estado de parada (STOP), LED VERMELHO.
10. Botão de operação.
11. Slot para submódulos de memória.
12. Conector para ligação da PG, PC, OP ou SINEC L1.
13. Conector para entradas de interrupção / rápidas.

O CLP poderá ser expandido utilizando-se 16 unidades de bus (32 slots), até quatro trilhos. Os slots são numerados em seqüência. A numeração começa no 0, sendo o slot ao lado da CPU.



## ENERGIZAÇÃO DO CLP

A energização do CLP deverá ser feita através de uma fonte de 24Vcc, e deverá apresentar a seguinte configuração:



Obs:

IN = corresponde aos bytes de entrada 32.0, 32.1, 32.2 a 32.7;

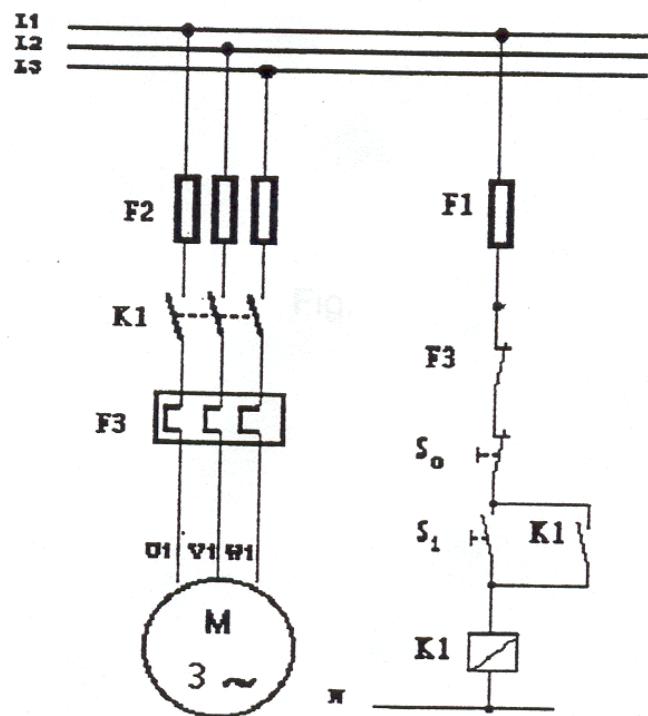
Bytes = corresponde aos oitos bytes, ou seja: 32.0 a 32.7 ou 33.0 a 33.7;

Word = corresponde a some dos bytes, ou seja: 32.0 a 33.7.

## EXERCÍCIOS

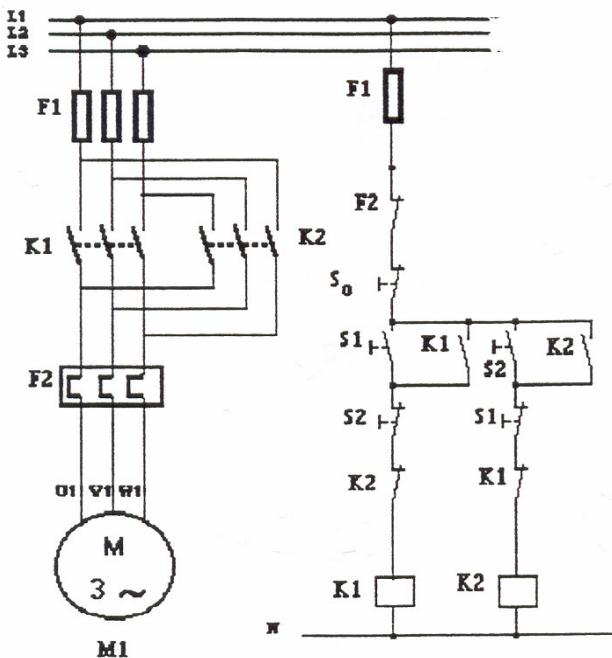
Baseado nos esquemas elétricos, apresentados a seguir, criar os programas em linguagem STEP 5.

1. PB 1 – Acionamento de motor de indução trifásico por chave de partida direta.



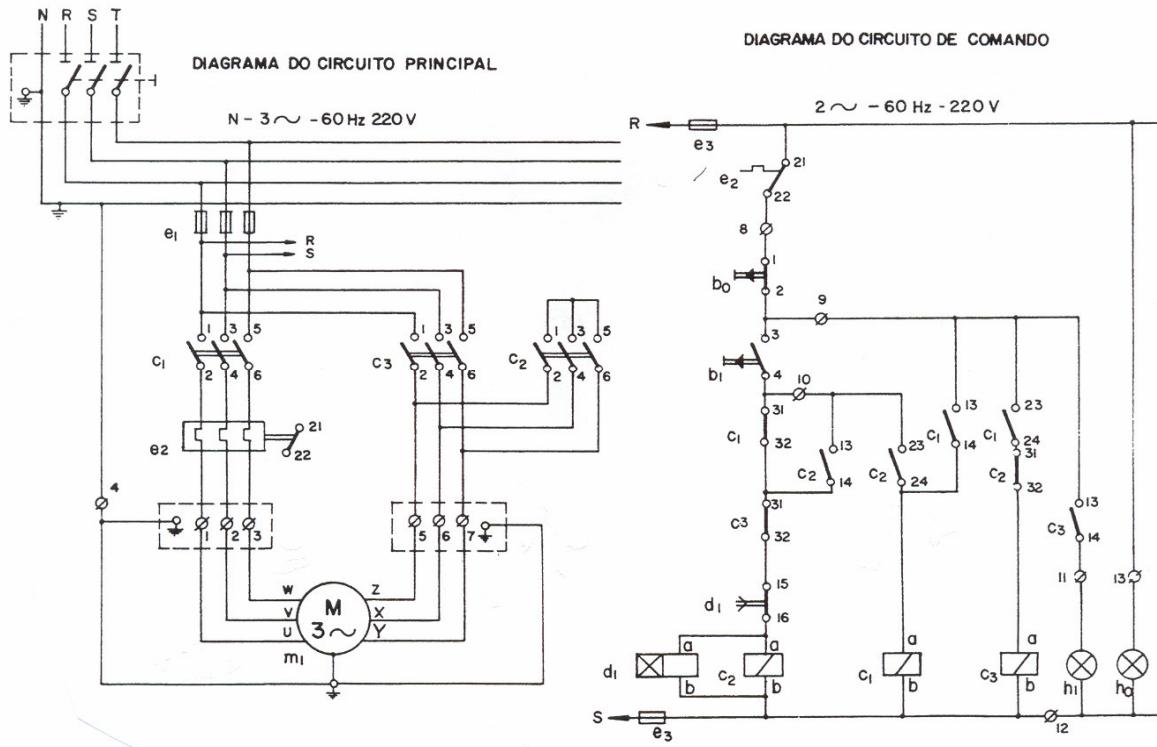
**EXERCÍCIO**

2. PB 2 – Acionamento de motor de indução trifásico por chave de partida direta com reversão.



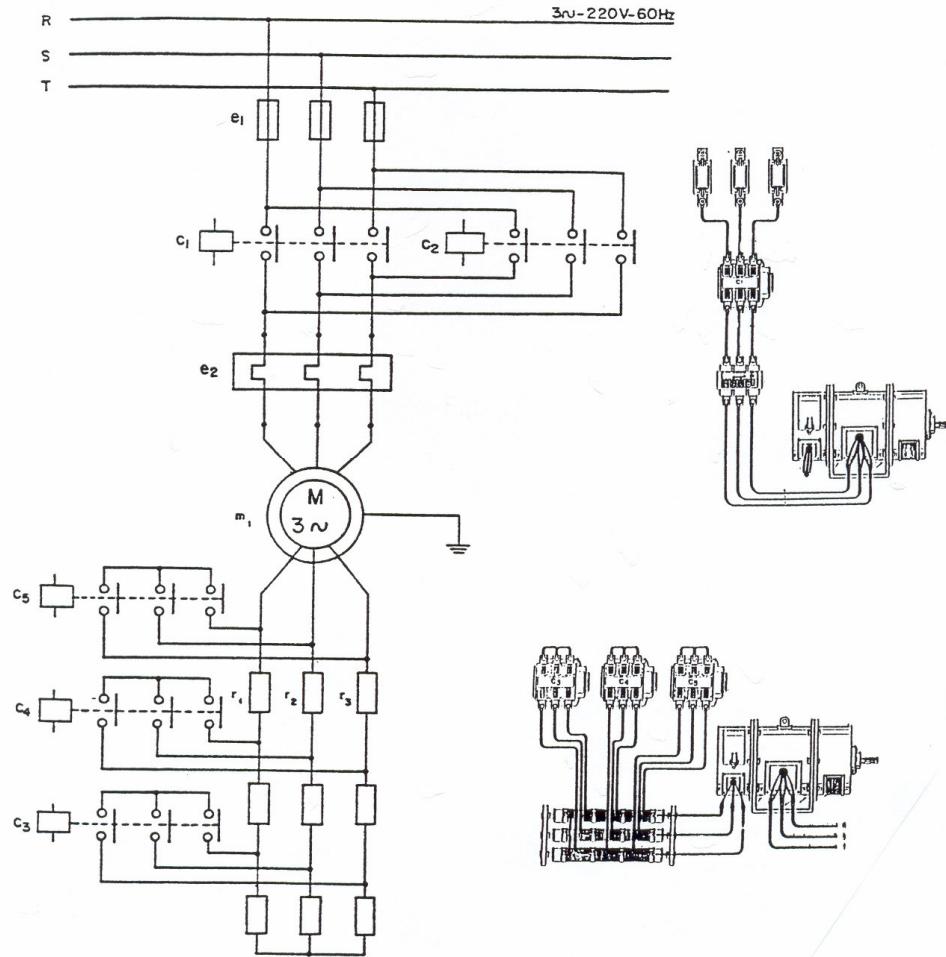
**EXERCÍCIO**

3. PB 3 – Acionamento de motor de indução trifásico por chave estrela triângulo.

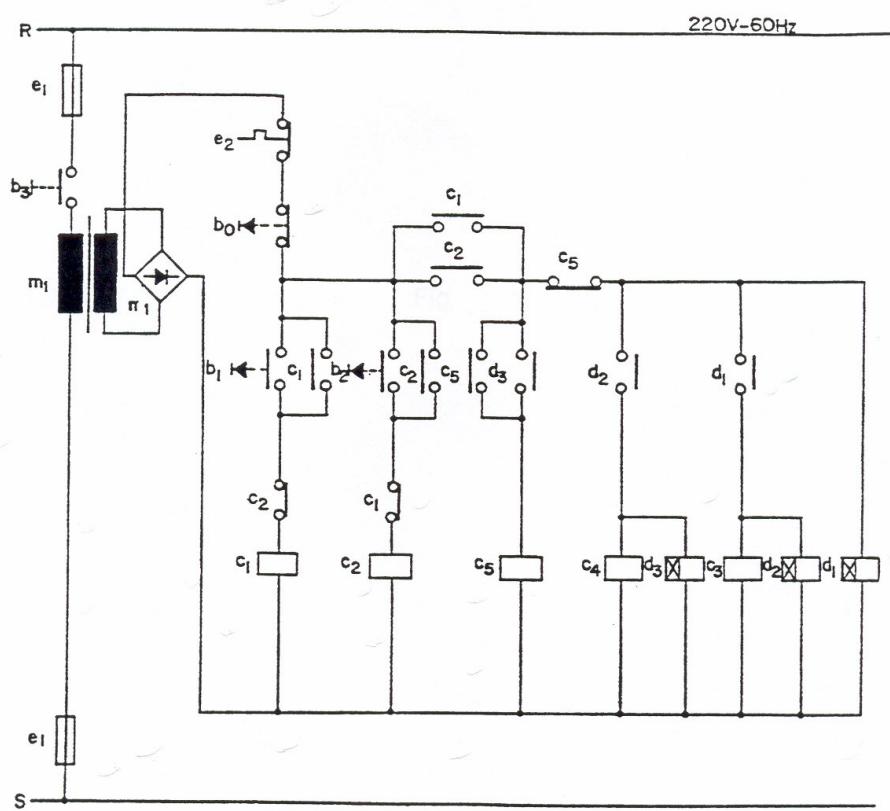


**EXERCÍCIO**

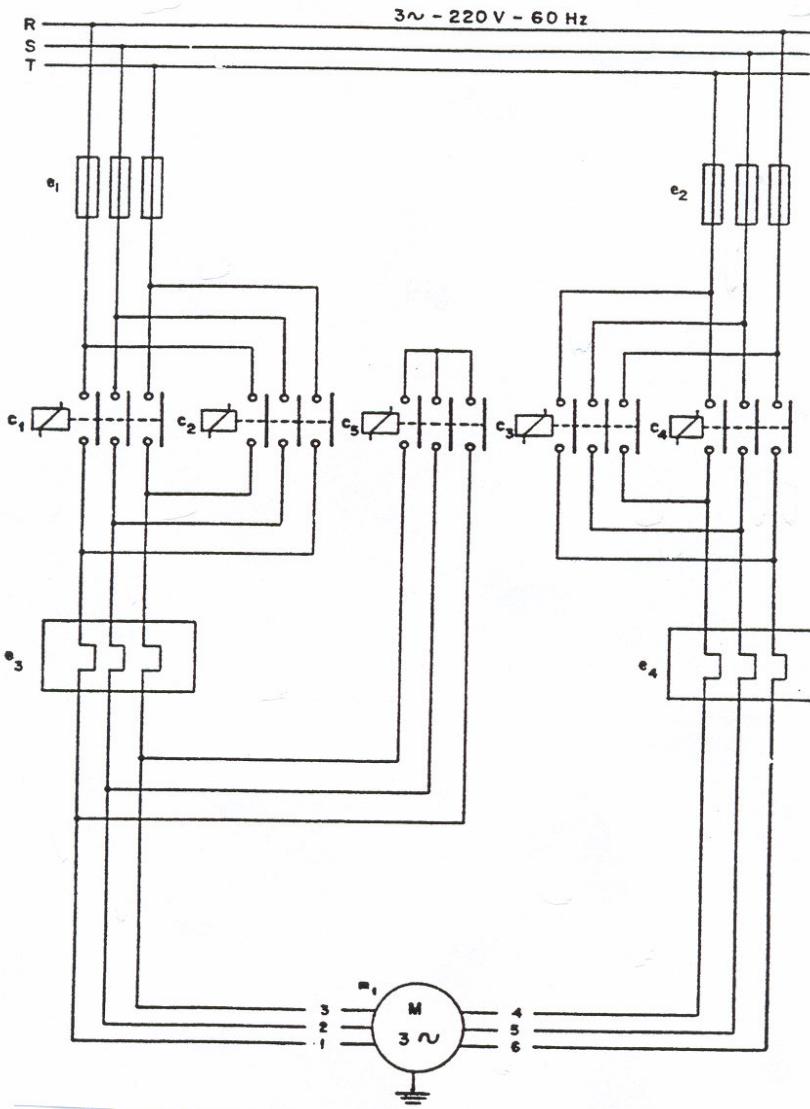
4. PB 4 – Desenvolver, montar e testar, o circuito auxiliar por memória (CLP), para uma chave magnética com reversão para partida por aceleração rotórica.

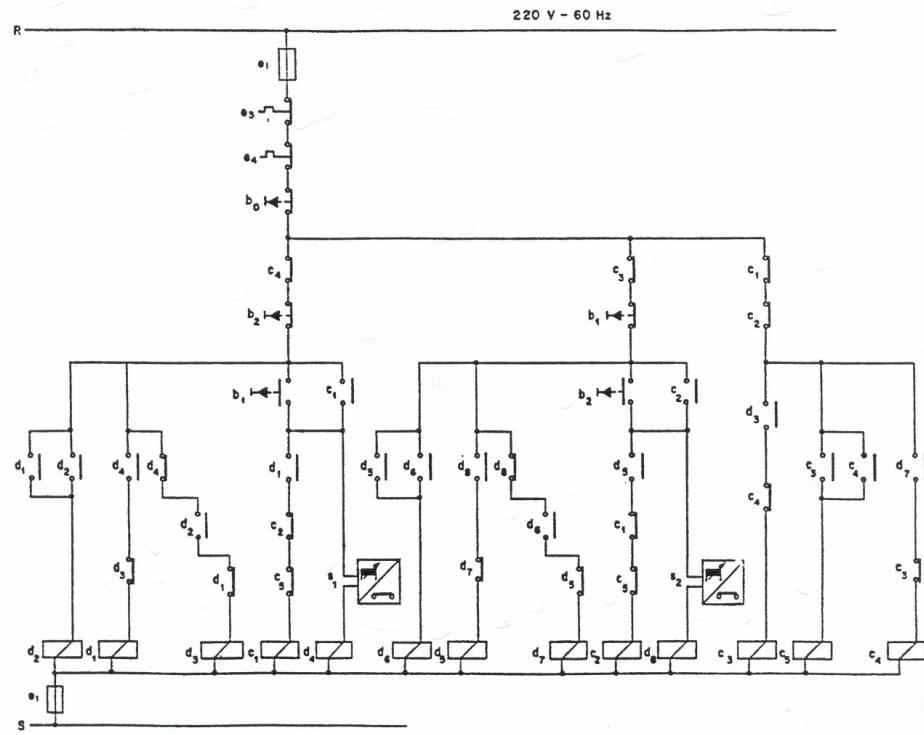


**EXERCÍCIO**



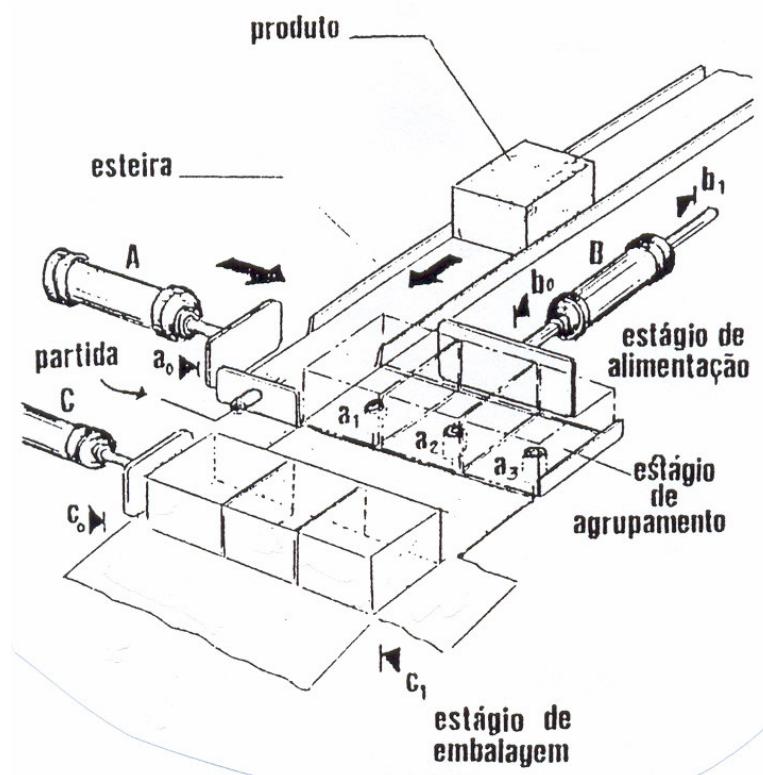
5. PB 5 – Desenvolver, montar e testar, o circuito auxiliar por memória (CLP), para uma chave magnética com reversão para duas velocidades para um motor com bobinado tipo Dahlander.





## 6. PB 6 – Esteira.

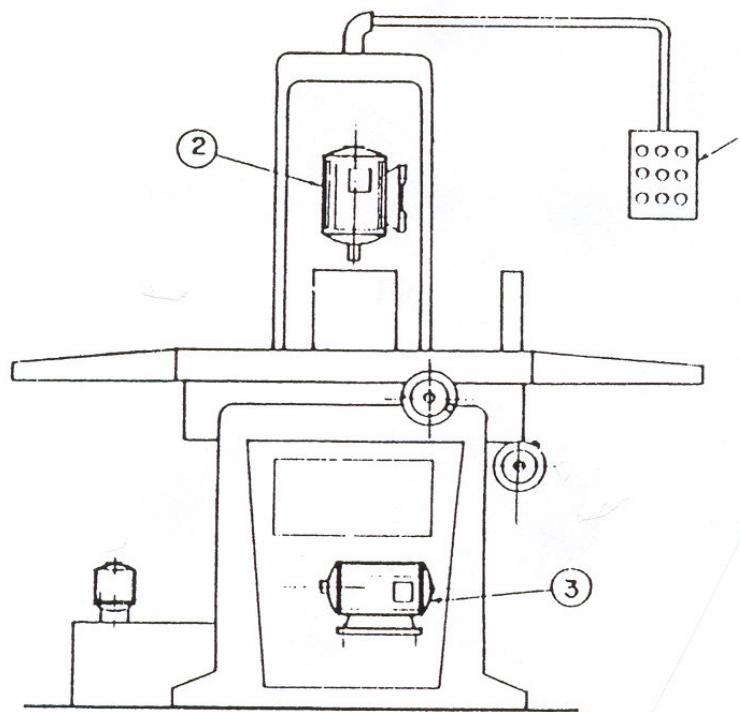
Descrição: Projetar no CLP um circuito eletropneumático para automatizar uma operação de agrupamento de embalagens de pacotes de leite em pó. Os pacotes devem ser retirados de uma esteira transportadora e empurrados um a um até formar lotes de 3 pacotes (cilindro A de dupla ação), sendo em seguida transferidos até o estágio de embalagem (cilindro B de dupla ação). No estágio final de embalagem, os fardos de 45 pacotes deverão ser retirados pelo cilindro C de simples ação com retorno mola.



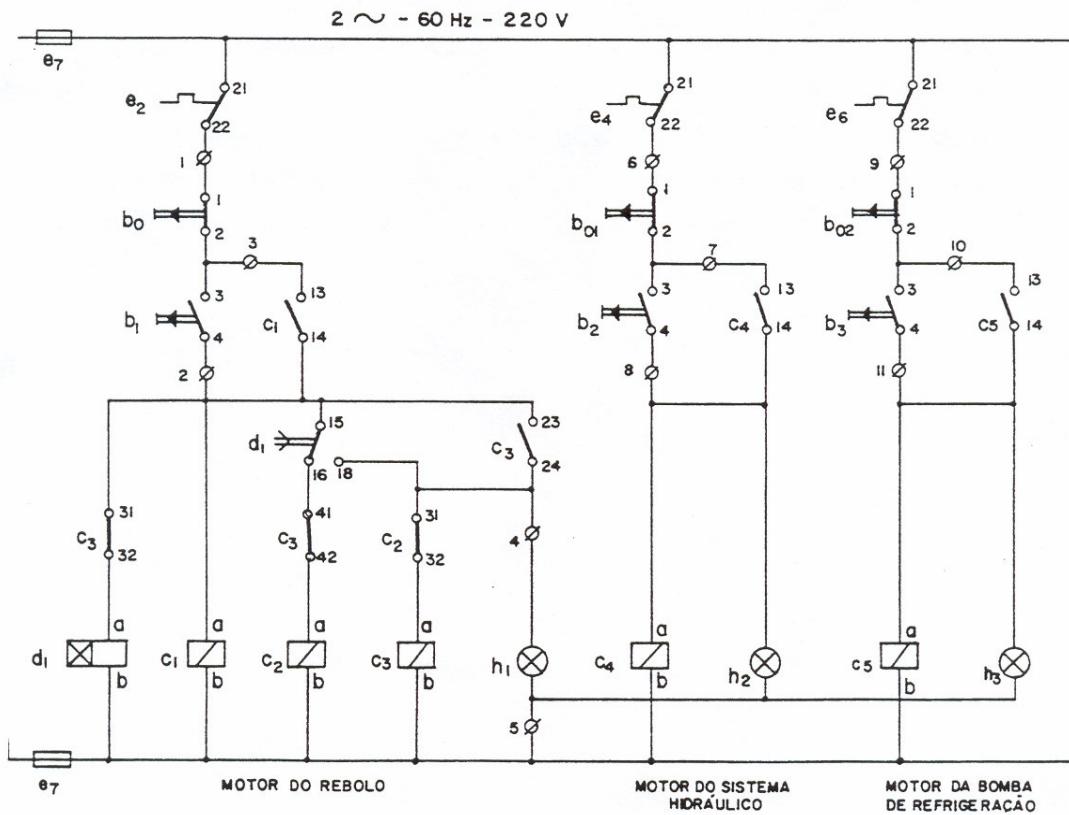
**EXERCÍCIO**

7. PB 7 – Desenvolver, montar e testar, o circuito auxiliar por memória (CLP), para uma RETÍFICA DE SUPERFÍCIE PLANA, dentro das seguintes características de funcionamento:

1. No início de funcionamento deverá ser acionado o motor de rebolo;
2. Após 30 segundos deverá ser acionado automaticamente o motor do sistema hidráulico;
3. Após 15 segundos da entrada do motor do sistema hidráulico, deverá ser acionada automaticamente a eletrobomba do sistema de refrigeração, que deverá fazer o desligamento dos motores dos sistemas de rebolo e hidráulico, permanecendo apenas o sistema de refrigeração durante 60 segundos. Após este tempo a retífica deverá iniciar o seu ciclo de trabalho automaticamente.

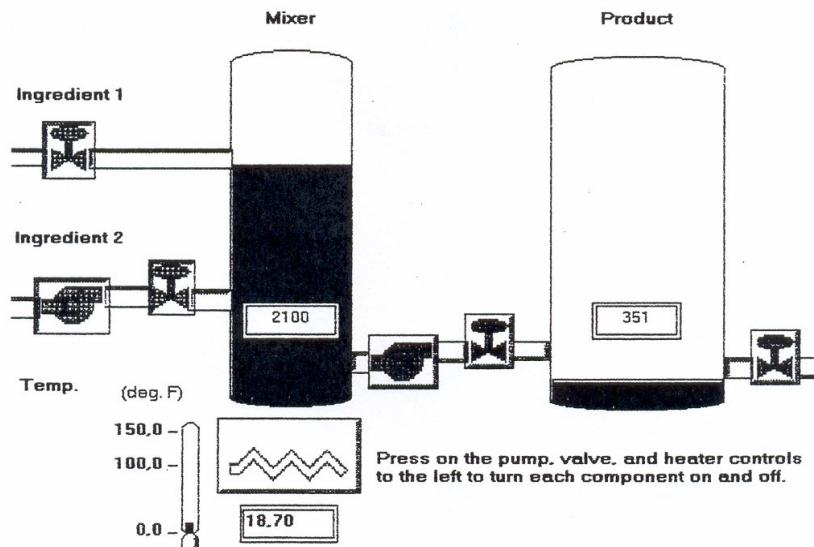


Comando por fiação:



8. PB 8 – Automatizar o processo de vazão de dois tanques industriais, com a seguinte programação:

O tanque número 1 (mixer), deverá conter 5m<sup>3</sup> do ingrediente 2, e completado até 20m<sup>3</sup> pelo ingrediente 1 (15m<sup>3</sup>). Quando alcançar o nível de 20m<sup>3</sup>, deverá ser acionada a eletrobomba 3, transferindo a mistura para o tanque de produtos. A temperatura máxima do sistema deverá ser de 150C.

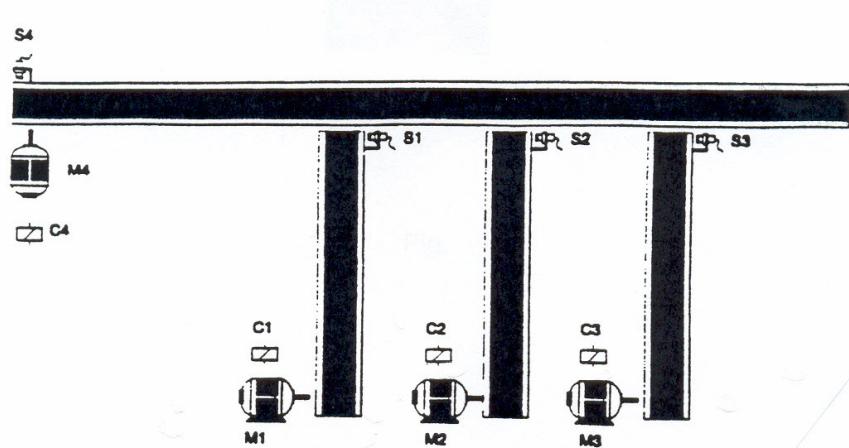


**EXERCÍCIO**

## 9. PB 9 – Esteira.

## Descrição:

A linha de produção abaixo é constituída por quatro esteiras. Inicialmente devem ser ligadas às esteiras menores, e a esteira maior deve ser ligada após a presença de três caixas, proveniente das esteiras menores. No momento do acionamento deverão ser desligados os motores m1, m2, m3, ficando em serviço apenas o motor 4 durante o tempo de 15 segundos. Após decorrido o tempo do motor 4, deverão ser acionados automaticamente os motores m1, m2 e m3.



**EXERCÍCIO**

## SÍNTESE DOS COMANDOS

### PASSOS PARA A CRIAÇÃO E CONFIGURAÇÃO DE UM NOVO ARQUIVO DE PROJETO

A função de um CLP se baseia no controle do Acionamento de motores, Válvulas, e outros equipamentos. Basicamente, ele substitui o circuito de Comandos utilizado para acionar equipamentos elétricos e eletrônicos. O CLP que vamos trabalhar é do fabricante Siemens, modelo S5-95U. Este modelo vem com:

- 16 entradas digitais
- 16 saídas digitais
- Memória RAM de 16 KB
- Bateria de segurança

A Bateria tem função de manter o programa na memória do CLP, no caso de falta de energia elétrica; embora o programa entre no estado STOP e só volte a executar com o restabelecimento da alimentação elétrica. Sem a bateria, o programa seria perdido, pois a memória RAM só mantém seus dados enquanto estiver energizada (Por isso se chama Memória Volátil).

Para funcionar e operar este CLP, devemos utilizar um computador como um Terminal de programação (TP). Nele, poderemos criar os programas que vão interagir com o CLP. Estes programas serão criados por um software específico, chamado STEP 5; um programa criado para este modelo de CLP e que funciona apenas no ambiente DOS 6 ou anterior.

Com ele, criaremos os circuitos de acionamento (Comandos) para os equipamentos desejados.

A grande vantagem se verifica na versatilidade da criação e possibilidade de alteração destes circuitos de comandos. Você faz estas alterações direto no programa, sem ter que mexer com a fiação física instalada. Ou seja, é uma alteração Lógica (do programa).

## Sinal Analógico e Digital

Um sinal Analógico pode assumir qualquer valor em um intervalo de tempo qualquer. Seu gráfico é representado por uma curva.

O sinal Digital tem valores predefinidos (Ex: 5V, 10V, 15V, 20V...) e salta de um valor para outro sem assumir os valores intermediários. Seu gráfico se assemelha a degraus de uma escada.

Dentre os sinais digitais, o mais interessante é o sinal Binário, que só pode assumir dois valores distintos (Ex: 5V e 10V, 0V e 6V, -4V e +4V), com sua representação gráfica se assemelhando a pulsos de larguras variáveis e proporcionais ao tempo. No caso do nosso CLP, teremos dois sinais binários, que são 0V e 24V.

## Criação de um novo diretório (*Pasta*)

Para guardar o nosso projeto, devemos primeiro criar uma nova *Pasta* ou *Diretório* na raiz do disco C:.

Assim, após ligarmos o computador, no ambiente DOS vamos para a raiz de C:, e com o prompt em C: fazemos:

C:\> MD “Nome da pasta” e pressiona o ENTER

Após pressionar o Enter, o prompt volta a C:\>. Se quiser ver se o seu diretório foi realmente criado, digite DIR e pressione ENTER, e aparece a lista de arquivos e pastas do diretório atual (No caso, da raiz).

O programa Step 5 está gravado na pasta STEP5. Para executar o programa, mude para esta pasta fazendo CD STEP5 e pressione ENTER. Depois, digite o comando S5 que aciona o arquivo executável S5.EXE e pressione ENTER. Com isso, aparece a tela do Step5 com as listas de opções na parte superior.

## Configurações de um novo projeto

Selecione o seguinte caminho:

Object → Project → Settings → Page 1  
Page2

Em Page 1, vamos definir como diretório de trabalho a pasta que você criou. Para isso, selecione a opção Working Dir:, e pressione a tecla F3 – *Select*, onde fará surgir uma nova janela. Tecle então F3 – *Change Dir*, e na nova janela que aparece use a tecla TAB e selecione a opção Drive/Dir.

Usando as setas do mouse selecione [ -- C -- ] e tecle ENTER. Isso fará exibir os diretórios dentro da raiz da unidade C:. Selecione na lista o seu diretório criado acima. Com ele selecionado, pressione o ENTER novamente e confirme com a tecla INSERT.

Com isso, retorna a primeira tela aberta, e confirme novamente com INSERT, a opção selecionada.

Com o cursor em Program File, tecle em F3 – Select, e digite o nome do novo arquivo C:"Nome"ST.S5D, e confirme com ENTER.

Na opção Symbol File, vamos definir o nome dos arquivos de símbolos dos operandos. Selecione e tecle F3 e digite o nome do arquivo C:"Nome"Z0.INI e confirme com ENTER. O arquivo Sequential File é automaticamente definido, e é relativo a documentação do programa ( C:"Nome"Z0.SEQ ).

As demais opções da Page 1 são relativas a rodapé de impressão, informações dos sistemas de documentação, Eprom e outros.

Em seguida, selecione a Page 2 teclando em F1 – Page 2.

Em Page 2 vamos configurar primeiro o modo de operação. Esta configuração pode ser feita selecionando a opção Mode: e teclando F3 – *Select*. Os modos de operação podem ser:

**Offline:** onde o CLP não pode receber e enviar informações para o computador (Desativa a Comunicação).

**Online:** onde o CLP pode receber e enviar informações para o computador (Ativa a Comunicação). Neste caso, escolha online para poder transferir informações para o PLC via micro.

A representação do programa pode ser definida selecionando a opção Representaion: e teclar em F3. Oferece os seguintes modos:

STL – Lista de Instruções

CSF - Blocos de funções

LAD – Representação por circuitos de relés.

Se quiser que na visualização dos blocos o documento exiba os símbolos e comentários dos operandos, devemos ativar a opção Symbol: com a tecla F3. Para isto, temos a opção Yes que ativa e No para desativar o arquivo de símbolos.

Na opção Display:, em Abs mostra o próprio operando, em Sym mostra o símbolo. Usa F3 para alterar.

Em Comments:, possibilita ver títulos e comentários. Usa o F3 para alterar entre Yes ou No.

As demais opções referem-se aos arquivos de impressora, destino de impressão, etc. agora vamos voltar à tela inicial com a tecla INSERT para confirmar.

Vamos salvar e nomear nosso projeto, pelo caminho abaixo:

Object → Project → Save As

E apertando ENTER, surgirá uma nova janela. No campo File name: aparece [??????PJ.INI], as interrogações serão substituídas pelo nome do arquivo do projeto. Este projeto conterá todas as configurações feitas anteriormente. Com o cursor em cima do nome, tecle INSERT, então o arquivo estará salvo.

Para carregar um arquivo já existente, não precisa fazer todas as configurações. Basta após carregar o Step 5 com o S5, selecionar:

Object → Project → Load

E surge uma nova janela; com TAB selecione Drive/Dir, e com as setas selecione [ -- C -- ] e tecle ENTER. aparece a lista do conteúdo de C:

Nela, selecione o seu arquivo de projeto criado anteriormente e tecle INSERT para confirmar. E o seu projeto é carregado com as mesmas configurações.

## Criando um novo ou abrindo um Bloco de Programa

Para criarmos um novo bloco de programação dentro de um arquivo de projeto, devemos usar o menu Editor. Através deste menu, podemos criar qualquer tipo de bloco do Step 5. Os principais blocos são:

### **OB1 – Bloco de Organização**

É um bloco cíclico, utilizado para registrar e chamar os outros blocos. Se você criar um novo bloco, este deve ser inserido no OB1 através de um Jump, caso contrário o CLP não executará esse programa.

Podemos comparar o OB1 a um porteiro, que tem uma lista de “convidados” e só permite a entrada desses convidados.

### **PB – Bloco de Programação**

É neste bloco que criamos as programações básicas, envolvendo portas lógicas, flip-flops, temporizadores, contadores e comparadores.

Podemos criar, em um mesmo arquivo, do PB0 ao PB255 (ou seja, 256 PB's), sendo cada um bloco independente, com programações distintas.

Lembre-se ainda que, para cada bloco criado, devemos registrá-lo no OB1 para que possa ser executado

### **FB – Bloco de Função (Avançado)**

Permite criar operações que usam funções avançadas. Este bloco será visto no curso de CLP avançado

### **DB – Bloco de Dados (Avançado)**

Não contém programas, e sim dados que serão utilizados por outros programas, como os PB's e FB's. Também será visto no CLP avançado

Para criar ou abrir um destes blocos, selecione o seguinte caminho:

Editor → Step 5 Blocks → In the Program File  
In the PLC

Estas duas últimas opções, permitem você decidir se seu programa será criado no disco C: do computador, ou direto na memória do CLP.

Lembre-se de que, ao criar direto no CLP, o programa não estará salvo no disco. Se você quiser salvar no disco, use a opção “Program File”. Vamos usar esta última e criar no disco C:. aparece um quadro:

Em Block, digite o nome do bloco a ser criado (Ex. PB3 ou outro desejado). Clique em OK e aparece a tela azul com o nome do bloco na parte superior. Para criar o programa, use a folha que contém a lista dos principais comandos do Step5 para lhe auxiliar.

Após criar o programa, pressione a tecla Insert várias vezes para salvar, até voltar à tela inicial.

Cuidado! Somente estará salvo, após retornar à tela inicial.

Agora que já criamos o PB3, vamos registrar em OB1 para que ele possa ser executado. Para isso, usaremos o mesmo procedimento anterior:

Editor → Step 5 Blocks → In the Program File  
In the PLC

Também escolha “Program File”. Em Block, digite o nome do bloco a ser criado no caso, o OB1. Usando a folha da Lista dos comandos, procure o que se refere à criação de um Jump:

No modo Edit → Shift + F2 ( Blocks ) → F4 ( JU )

Digite então PB3, que é o bloco que será chamado e pressione insert até salvar e voltar à tela inicial.

### **Transferindo os programas para a memória do CLP**

Como criamos os blocos no “Program File” ( Disco C: ), para serem executados eles devem ser transferidos antes para o CLP. Para isto, vamos usar o caminho abaixo:

Object → Blocks → Transfer → File - PLC

No quadro que aparece, digite em Blocks o nome do(s) bloco(s) a serem transferidos. No caso, digite PB3,OB1. Pressione insert para iniciar a transferência. Quando aparecer a mensagem “2 blocks transferred” pressione Enter e depois Esc. Com isso, os blocos já estão na memória do CLP, pronto para funcionar.

OBS: Verifique se o CLP está em modo RUN ( Led Verde ). Caso esteja em STOP ( Led Vermelho ), você pode ativar para RUN através de:

Test → PLC Control → Start PLC ( Vai para o modo RUN )  
Stop PLC ( Vai para o modo STOP )

Após entrar no modo Run, seu programa começa a funcionar automaticamente. Se quiser visualizar graficamente o programa:

### **Para visualizar a execução do programa na memória do CLP**

Selecione o seguinte caminho:

Test → Block Status

No quadro que aparece, digite o nome do PB que você deseja visualizar (no caso, o PB3), e pressione o insert. Na tela, aparecem os elementos ativados em verde e desativados em pontilhado.

Nota: Para o programa ser executado, você não é obrigado a ativar a visualização em tela. Esta visualização, é apenas um recurso auxiliar para o controle e detecção de falhas.

### **Para visualizar a relação de programas na memória do CLP**

Se você quiser visualizar uma lista com todos os programas que estão na memória do CLP, pode usar este caminho:

Object → Blocks → Directory → In the Program File → Insert  
In the PLC

Com estas opções, você pode optar entre ver uma lista do conteúdo da memória do CLP com a opção “In the CLP” ou ver o conteúdo do disco C: com a opção “In the Program File”.

Caso deseje excluir um ou mais blocos da memória ou do disco:

Object → Blocks → Delete → In the Program File → Insert  
In the PLC

No quadro que aparece, digita o nome do bloco a ser deletado em Block e pressiona a tecla insert.

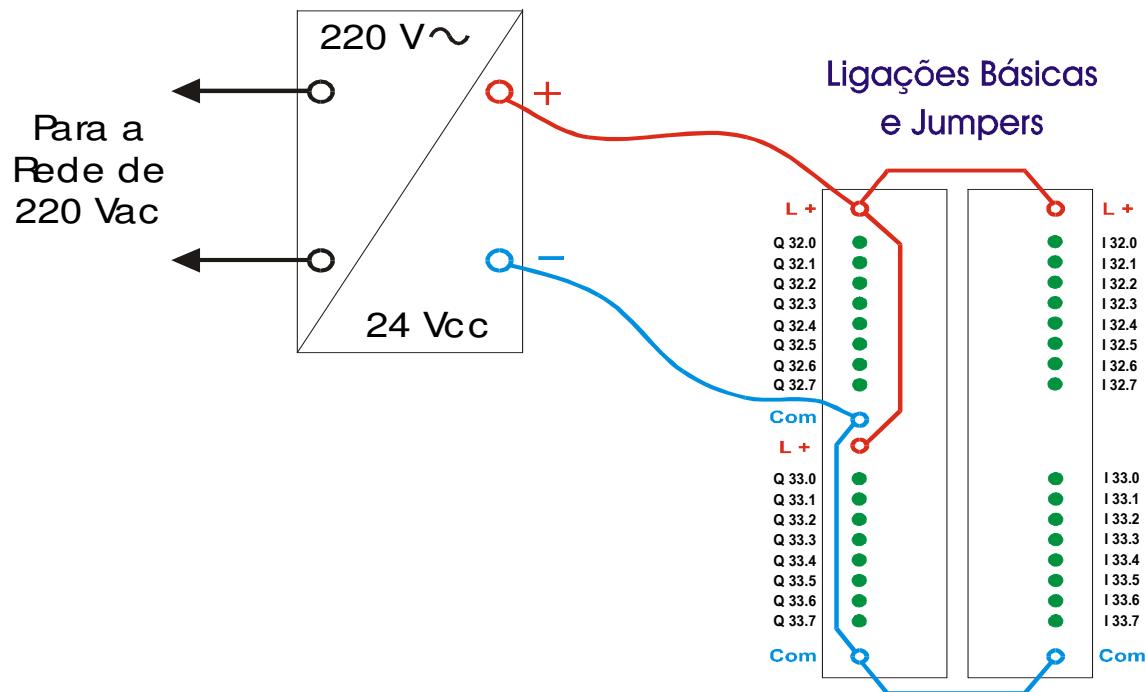
Semelhante ao caso anterior, exclui ou do disco C: (Program File) ou da memória do CLP (In the PLC).

Nota: Quando excluímos um programa da memória do CLP, ele ainda continua no disco C:, caso tenha sido salvo anteriormente. Quando temos mais de um programa na memória do CLP, eles devem usar entradas e saídas diferentes, caso contrário um vai interferir no funcionamento do outro. Por isso, é bom deixar na memória somente os programas que vamos usar no momento.

## LISTA DE COMANDOS DO STEP 5

1. Para entrar no Modo de EDIÇÃO  
No modo OUTPUT ⇒ Tecle F6
2. Portas E e OU  
No Modo EDIT ⇒ F1 ( & ) e F2 ( >=1 )
3. R, S, Temporizador, Contador, Conector (#)  
No modo EDIT ⇒ F5(Binary Op)
4. Comparador  
No Modo EDIT ⇒ Shift + F5 ( Compare )
5. Acrescentar um Novo SEGMENTO  
No modo EDIT ⇒ F6 ( Comp Seg )
6. Operações com SEGMENTOS  
No modo OUTPUT ou STATUS ⇒ F5 (Seg Fct):  
⇒ SHIFT + F4 ( Deletar Segmento ) ⇒ OK  
⇒ F5 ( Inserir Segmento ) ⇒ F1 ( New ) ⇒ OK  
⇒ F6 ( Append ) ⇒ F1 ( New ) ⇒ OK
7. Acrescentar mais 1 Entrada  
No Modo EDIT ⇒ Com a Tarja no final da porta ⇒ F3
8. Para Transformar em INVERSOR  
No Modo EDIT ⇒ Deixa a Tarja na Entrada ⇒ F4
9. Para colocar o JUMP ( JU )  
No Modo EDIT ⇒ Shift + F2 ( Blocks ) ⇒ F4
10. Para Aumentar o Segmento antes da Entrada  
No Modo EDIT ⇒ Seleciona a Entrada ⇒ HOME
11. Para alternar entre CSF, STL e LAD  
No modo OUTPUT ou STATUS ⇒ Tecle Shift + F5
12. Para alternar entre Segmentos de um mesmo programa  
No Modo OUTPUT ⇒ Pelo Teclado ⇒ PgUp/PgDown

Fonte de Alimentação  
no painel do CLP:



## BIBLIOGRAFIA

REDE GLOBO. Automação Industrial. Rio de Janeiro, 2000. (Telecurso 2000)

Controle, Instrumentação e Automação de Sistemas. Revista Oficina. São Paulo, vol. 62, pág. 31 a 36, março/2000.

Controle, Instrumentação e Automação de Sistemas. Revista Oficina. São Paulo, vol. 64, pág. 29 a 32, maio/2000.

Controle, Instrumentação e Automação de Sistemas. Revista Oficina. São Paulo, vol. 65, pág. 27 a 34, junho/2000.

Controle, Instrumentação e Automação de Sistemas. Revista Oficina. São Paulo, vol. 66, pág. 34 a 38, julho/2000.

Controle, Instrumentação e Automação de Sistemas. Revista Oficina. São Paulo, vol. 67, pág. 23 a 31, agosto/2000.

SIEMENS. Simatic S5 – Documentation. 1993. Suplement.

**Elaboração**

Edelson Costa Bivar

**Digitação**

Patrícia de Souza Leão Batista

**Diagramação**

Anna Daniella C. Teixeira

**Editoração**

Divisão de Educação e Tecnologia – DET