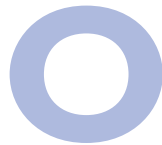


Capítulo 8

Controladores

lógicos

programáveis



controlador lógico programável é um dispositivo de estado que permite, por meio de um programa armazenado e de atuações (saídas) e leituras (entradas), o controle de máquinas ou processos. A NEMA (National Electrical Manufacturers Association), em 1978, definiu o CLP – controlador lógico programável – como um aparelho eletrônico digital capaz de realizar o controle de uma máquina ou processo, por meio de funções lógicas, de sequenciamento, de temporização, operações aritméticas. Esse controle é obtido com o emprego de um programa armazenado em uma memória interna contendo instruções para execuções de funções específicas que interagem com o meio externo através de dispositivos de entrada e saída (I/O) digitais ou analógicos.

O controlador lógico programável permite a automatização de processos industriais, de sequenciamento, intertravamento, controle de processos, produção por lote etc.

Praticamente não existem ramos ou aplicações industriais em que não possam ser aplicados os CLPs:

- máquinas industriais (operatrizes, injetoras, extrusoras, têxteis, calçados);
- equipamentos industriais para processos (mineração, siderurgia, petroquímica, química, alimentação, papel e celulose etc.);
- controle de sistemas embarcados em diversas aplicações aeroespaciais;
- equipamentos para controle de energia (demanda, fator de carga);
- aquisição de dados de supervisão em: fábricas, prédios inteligentes etc.;
- controle de processos com realização de controle PID, sinalização e intertravamento;
- bancadas de teste automático de componentes industriais.

Os controladores lógicos programáveis têm basicamente as seguintes características:

- *hardware* e/ou dispositivo de controle, facilmente programável e reprogramável, provocando o mínimo de interrupção da produção;
- *hardware* que ocupa espaço reduzido e baixo consumo de energia;
- capacidade de operação em ambiente industrial (trabalha em temperaturas na faixa de 0 a 60° C e em umidade relativa de 5 a 95%);
- *hardware* de controle que permite a expansão em módulos, conforme a necessidade;
- sinalizadores de estado e módulos *plug-in* de fácil substituição e manutenção;

- possibilidade de monitoração do estado de operação do processo ou do sistema através de sistemas supervisórios de controle e aquisição de dados (SCADA);
- capacidade de alimentar cargas que consomem até 2 amperes de corrente de forma contínua ou chaveada;
- compatibilidade com diferentes sinais de entrada e saída;
- conexão com outros CLPs por meio de redes de comunicação;
- possibilidade de integração com redes de chão de fábrica;
- possibilidade de expansão da capacidade de memória;
- possibilidade de programação de até cinco linguagens diferentes de um mesmo programa;
- custo competitivo em relação aos sistemas de controle convencionais;
- possibilidade de expansão da capacidade de memória;
- conexão com outros CLPs através de rede de comunicação;
- possibilidade de integração com redes de chão de fábrica;
- programação em pelo menos uma linguagem com possibilidade de programação em até cinco linguagens distintas em um mesmo programa.

8.1 Estruturação de um CLP

Um CLP é constituído por módulos de entrada e de saída. As funções disponíveis podem ser programadas em uma memória interna por uma linguagem de programação que possui um padrão internacional chamado IEC 61131-3, uma fonte de alimentação e uma CPU (unidade central de processamento).

Entre os componentes integrantes dessa estrutura temos o microprocessador, a memória, as entradas e saídas, o terminal de programação e a fonte de alimentação (figura 8.1).

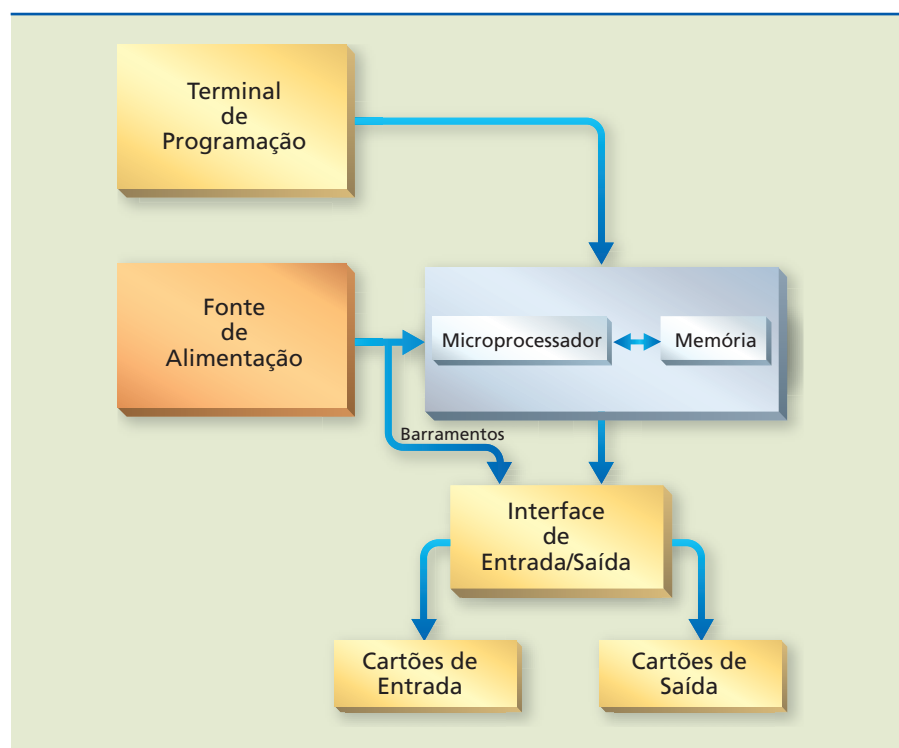


Figura 8.1

Diagrama de blocos com os componentes básicos de um CLP.

8.1.1 Microprocessador

É o responsável pelo processamento do programa, que coleta os dados da entrada e efetua o processo segundo o programa do usuário, armazenando na memória e enviando os dados para a saída como resposta ao processamento. Pode ser de 8, 16 ou 32 bits. Em CLPs de maior porte, adiciona-se um coprocessador para aumentar a capacidade de processamento em cálculos complexos com aritmética de ponto flutuante, uma memória RAM (*random access memory*) e uma memória Flash EPROM (*electrically-erasable programmable read-only memory*) ou E2PROM (para cópia do programa em memória não volátil).

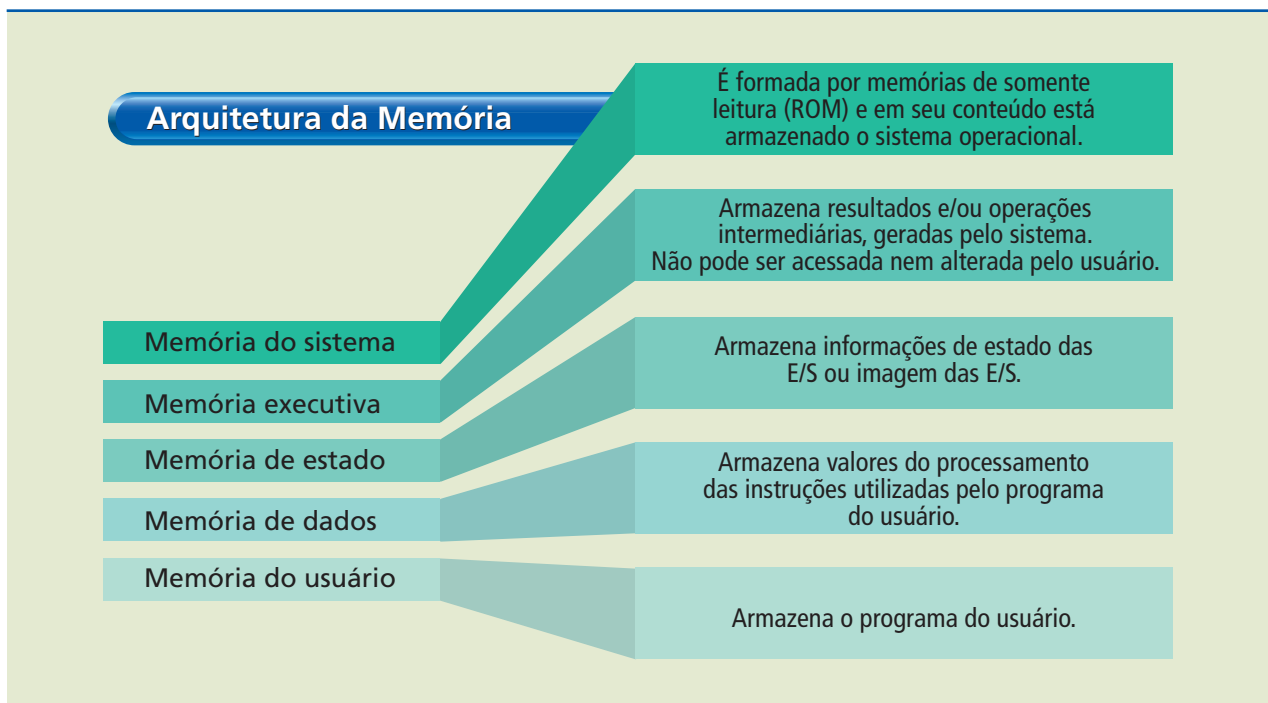
8.1.2 Memória

As memórias podem ser divididas em dois grupos, conforme a função:

- memória de dados: serve para armazenar temporariamente os estados de E/S, marcadores de *presets* de temporizadores/contadores e valores digitais para que a CPU possa processá-los. A cada ciclo de varredura a memória de dados é atualizada. Geralmente é uma memória do tipo RAM e é conhecida também como memória de rascunho;
- memória de usuário: armazena as instruções do *software* aplicativo e do usuário – programas que controlam a máquina ou a operação do processo, os quais são continuamente executados pela CPU.

Figura 8.2
Arquitetura da
memória de um CLP.

A capacidade de memória de um CLP é definida em função do número de palavras de memória previstas para o sistema. Ver um esquema da arquitetura da memória de um CLP na figura 8.2.



8.1.3 Terminal de programação

O terminal de programação é um periférico que permite a interação homem-máquina, permitindo ao usuário a introdução de programa (*software*) e a configuração do sistema. Pode ser um terminal exclusivo de um certo fabricante de CLP, ou seja, um equipamento dedicado; ou um *software* capaz de transformar um computador pessoal em um terminal do CLP por ocasião da conexão entre ambos (ver figura 8.3).

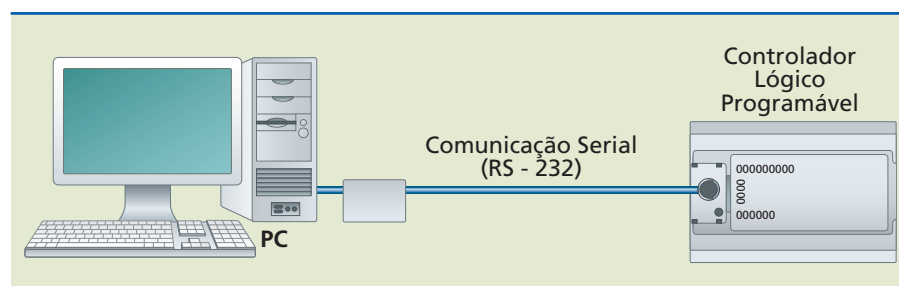


Figura 8.3

Terminal não dedicado
(Manual MicroLogix
– Allen-Bradley).

Nesse dispositivo de entrada, é realizada a codificação das informações do usuário, numa linguagem que deverá ser entendida pelo processador do CLP.

As funções que poderão ser realizadas de acordo com o tipo de terminal de programação são:

- elaboração do programa;
- introdução de novas instruções;
- modificação de instruções já existentes;
- monitoração do programa do usuário;
- verificação do estado de funcionamento do *hardware* do CLP;
- análise do conteúdo dos endereços de memória;
- atuação de saídas independente da lógica (forçando);
- cópia do programa do usuário em disco ou impressora.

8.1.4 Fonte de alimentação

A alimentação de energia do CLP usa uma fonte chaveada e uma única tensão de saída de 24 V. Esse valor é utilizado com a finalidade de alimentar os módulos de entrada e saída de dados e a CPU, ao mesmo tempo. Outra característica importante é que normalmente os componentes eletrônicos das máquinas industriais funcionam com essa tensão por ser bem menos suscetível a ruídos.

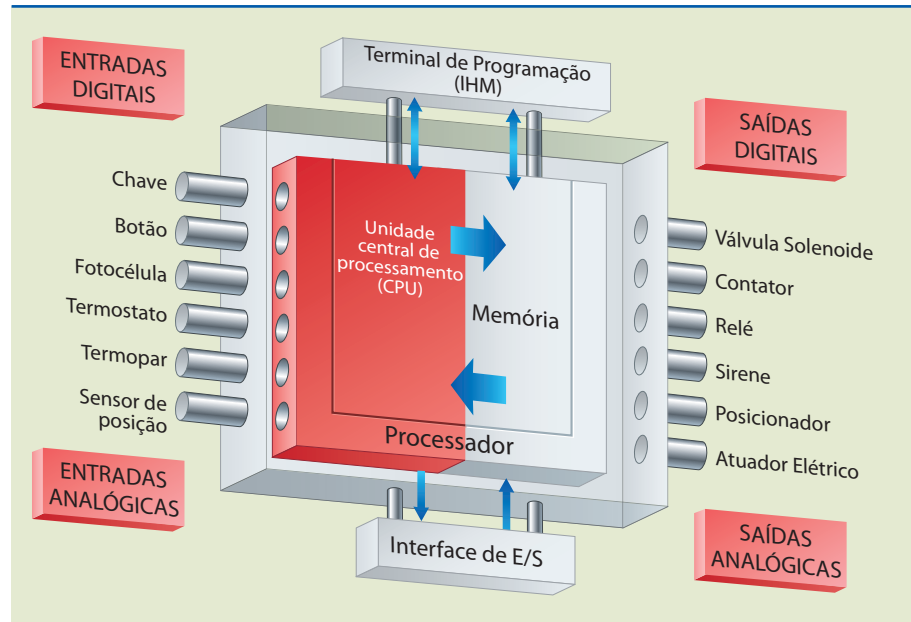
8.1.5 Componentes de entradas e saídas

São os circuitos responsáveis pela interação entre o homem e a máquina; pelos quais o homem se comunica com a máquina e pelos quais a máquina externa seus resultados ou controles. Nesses dispositivos, tanto o usuário quanto a máquina podem trocar informações. Os dispositivos de entrada são os responsáveis pela aquisição de dados de variáveis do processo, e os dispositivos de

saída são os responsáveis pelo acionamento de dispositivos físicos como relés, sinalizadores etc.

O acesso a essa interface pode ocorrer por bornes, blocos de bornes ou cabos e conectores. As entradas e saídas de um CLP podem ser divididas em duas categorias: as analógicas e as digitais. Na figura 8.4 são ilustrados os dois modelos de interfaces I/O.

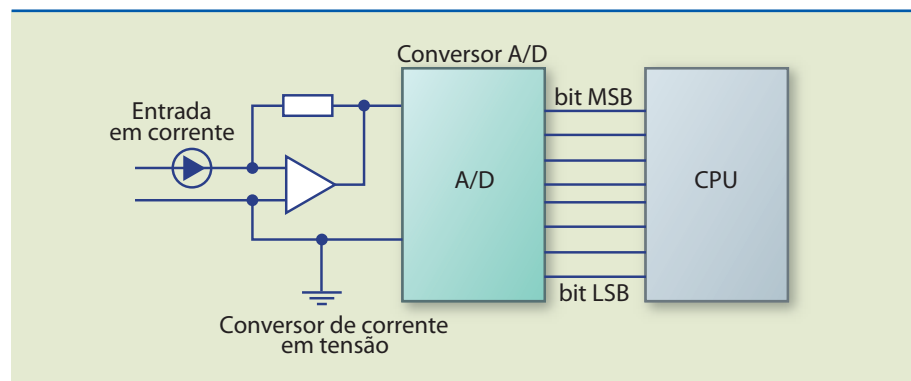
Figura 8.4
Arquitetura de um CLP com interfaces de entrada e saída.



Na entrada, o CLP permite trabalhar com as tensões usuais de comando (24 V em corrente contínua ou 110/220 V em corrente alternada) e as transforma em tensões de nível lógico aceitos pela CPU.

Para que a CPU opere com grandezas analógicas, é necessário que essas entradas sejam convertidas em digitais. O processo utiliza os conversores A/D (analógico para digital), e, assim, grandezas analógicas, como, por exemplo, temperatura, umidade relativa, pressão, entre outras, são convertidas em sequências numéricas binárias interpretadas pela CPU. Na figura 8.5 pode ser visto um esquema de uma interface para entrada analógica.

Figura 8.5
Interface para entrada analógica.



O módulo de saída comuta as tensões de controle fornecidas, necessárias para acionar vários dispositivos conectados. O isolamento é feito por meio de opto-acopladores ou transformadores (isolamento galvânico) – ver figura 8.6.

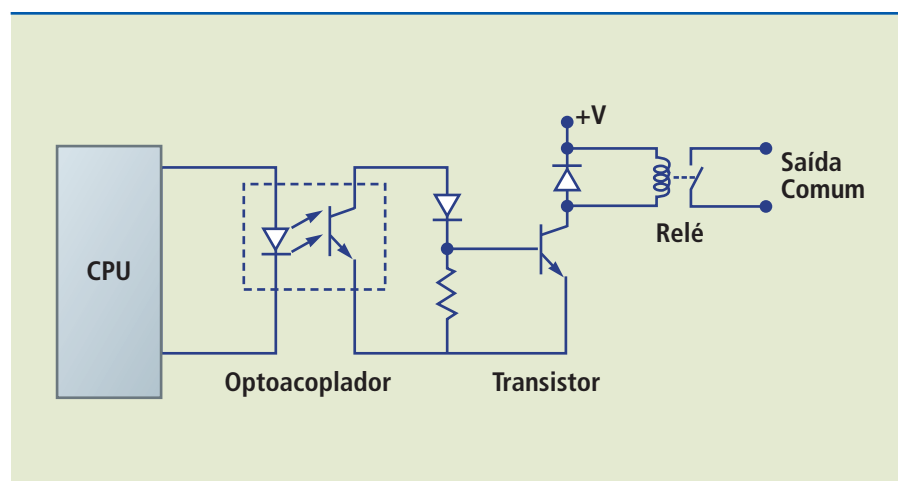


Figura 8.6

Interface para saída de sinal via contato de relé.

As entradas e saídas são organizadas por tipos e funções, em grupos de 2, 4, 8, 16 e até 32. Assim, são agrupadas normalmente como potências de dois por causa do processamento digital (binário) por interface (cartão eletrônico) de E/S. Os cartões são em geral do tipo de encaixe e, configuráveis, de forma a possibilitar uma combinação adequada de pontos de E/S, digitais e analógicas.

A quantidade máxima de E/S, disponíveis no mercado de CLPs, pode variar de 16 a 8 192 pontos, normalmente, o que caracteriza a existência de pequenos, médios e grandes CLPs.

8.2 Interface homem-máquina

O surgimento de novas necessidades do usuário, como a verificação dos processos ou a mudança de certos parâmetros no programa, sem que necessariamente o usuário precise se conectar a um computador para a realização dessa tarefa, fez com que os fabricantes de CLPs desenvolvessem um novo dispositivo que interagisse com o usuário, conhecido como interface homem-máquina ou IHM. Ela é responsável pela comunicação do usuário com o sistema para atuar em variáveis do processo (como temperatura, pressão etc.), sem que interfira no programa ou que entenda perfeitamente como ele funciona.

Existem no mercado duas versões de IHM: a alfa-numérica e a gráfica. Em uma interface alfa-numérica, a IHM é ligada ao CLP por sua porta de comunicação serial. Além dos parâmetros normais, quando se está programando uma IHM, indicam-se qual será a marca e o modelo do CLP com o qual vai se comunicar. Nas interfaces gráficas, o usuário pode, por meio de um programa específico, desenhar comandos em forma de botões, bem como lâmpadas para avisos ou alarmes, escolhendo cores, formatos, tamanhos, e definindo, também, endereços do CLP para cada elemento. Veja uma IHM na figura 8.7.

Figura 8.7
IHM



DAVID J. GREEN - TECHNOLOGY / ALAMY / OTHER IMAGES

8.2.1 Interface para comunicação em rede

Permite a comunicação do CLP com outros CLPs e com um PC. É colocada no lugar de um dos módulos de E/S ou em uma parte específica da CPU.

O tipo de interface e o cabo utilizado vão definir o padrão físico e o protocolo de rede. Ex.: MPI ou PPI (*point to point*), MODEBUS, FIELDBUS, PROFIBUS.

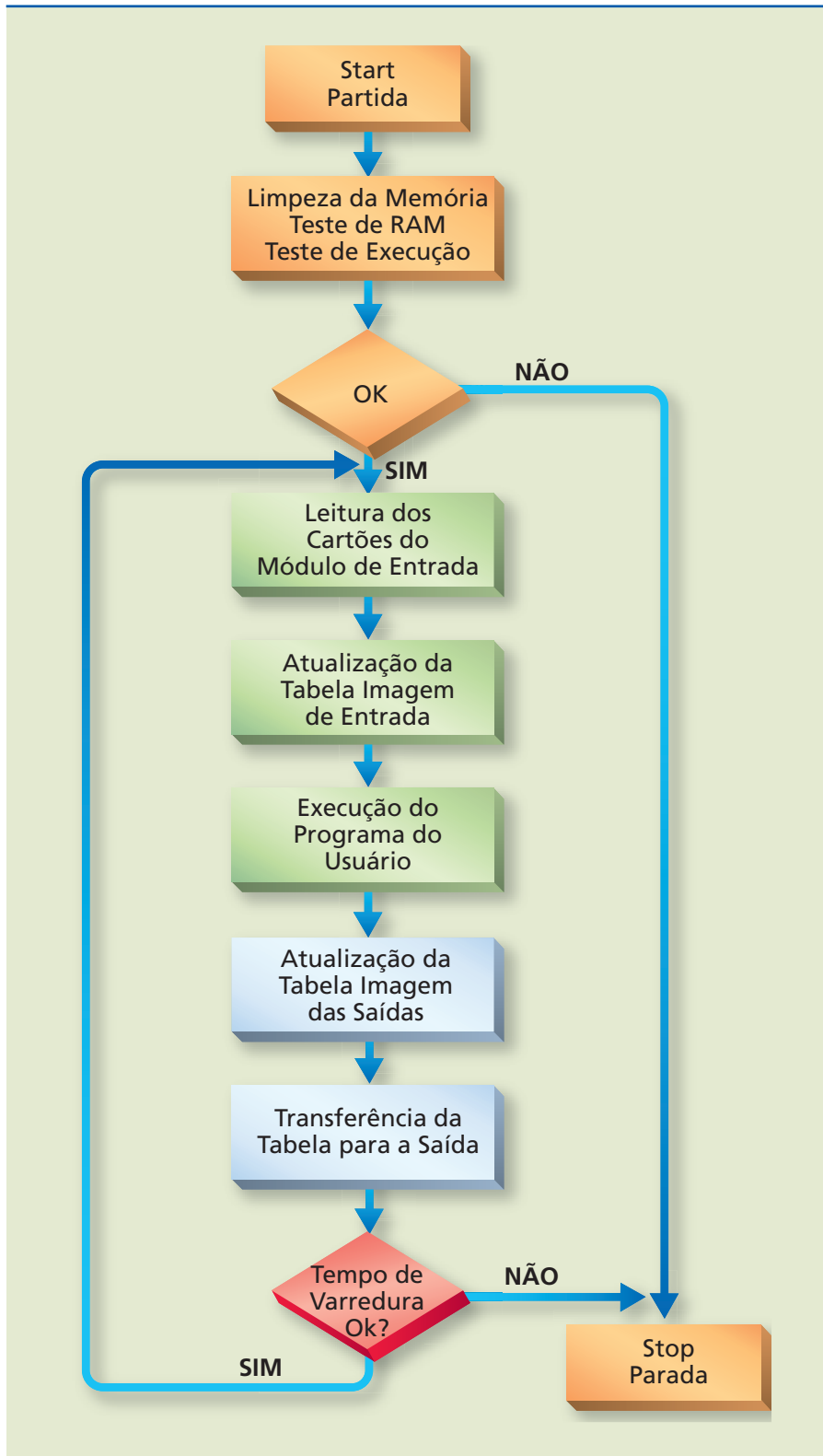
A comunicação serial usada com maior frequência é feita por simples cabos de par trançado. Os padrões mais comumente utilizados são o RS-232, *loop* de corrente 20 mA, o RS-422, RS-485 e portas USB em alguns casos.

8.2.2 Princípio de funcionamento de um CLP

O funcionamento de um controlador lógico programável baseia-se em um sistema de microcomputador em que há uma estrutura de *software* e *hardware* que, através das entradas e saídas de dados, realiza ciclos de varredura, adquirindo, executando e operando, como mostra a figura 8.8.

A CPU do controlador programável possui dois estados de operação: programação e execução, e pode assumir também o estado de erro, quando ocorrem falhas de operação ou de execução do programa.

No estado programação, o CLP não executa o programa, ou seja, não assume lógica de controle, apenas fica preparado para ser configurado ou receber novos programas, ou ainda, modificações de novos programas. A esse tipo de programação chamamos *off-line* (fora de linha).

**Figura 8.8**

Fluxograma simplificado de funcionamento de um CLP.

No estado execução, o CLP assume a função de execução do programa. Em alguns controladores, algumas modificações podem ser feitas no programa durante este estado que é chamado de *on-line* (em linha).

8.2.3 Recursos dos *softwares*

A capacidade de trabalho de um CLP é determinada pelo número de pontos de E/S, pela quantidade de passos de programação, e também pelos recursos de *software* disponíveis, ou seja, pelas funções que podem ser executadas. Os CLPs mais simples possuem pelo menos as seguintes funções básicas de *software*:

- Lógica E, OU e XOR.
- *SET* e *RESET*.
- Temporização e contagem.
- Cálculos com aritmética básica (+, -, ×, %).
- Parênteses (para associação de lógicas).
- Comparação de valores.
- Registrador de deslocamento.
- Salto.

À medida que os CLPs aumentam sua capacidade de processamento, são também desenvolvidas novas funções de *software* mais avançadas, como:

- Cálculos com ponto flutuante.
- Cálculos integrais e trigonométricos.
- Malha de controle PID.
- Posicionamento.
- Contagem rápida.
- Leitura de sinais analógicos.
- Linearização de sinais analógicos.
- Lógica *fuzzy*.

8.3 Linguagem de programação

A linguagem de programação é padronizada de acordo com a norma IEC 61131-3, que teve sua segunda edição publicada no ano de 2003. Visa atender tanto os conhecimentos do uso do relé, no qual os sistemas são automatizados quando se faz uso deles, como os conhecimentos da era digital, em que os sistemas são automatizados com os CLPs. No primeiro caso, fazemos a adequação da representação da linguagem pelos diagramas de contatos; no segundo, a representação pelos diagramas lógicos da tecnologia digital, ou ainda, a representação matemática.

Entre as principais vantagens da norma, podemos destacar a facilidade que o usuário tem em utilizar módulos e estruturar a programação em elementos funcionais, bem como definir a linguagem em que vai programar determinada parte do projeto, além de estar usando um ambiente de programação mundialmente conhecido, adaptável aos inúmeros fabricantes.

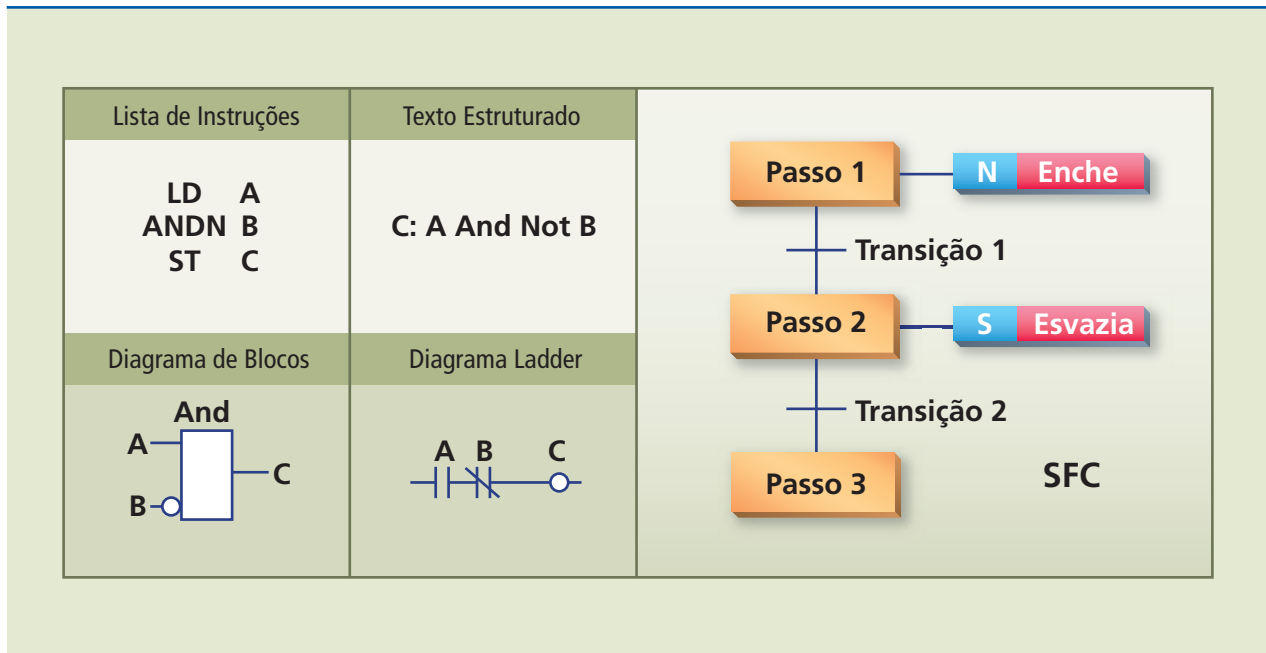
De acordo com a norma IEC 61131-3 e considerando a forma de representação, as linguagens de programação podem ser classificadas em dois grupos: gráficas e textuais.



Nas linguagens de programação gráficas, temos a *Sequential Function Chart* (SFC), o *Diagrama Ladder* (Ladder Diagram – LD) e Blocos de Função (*Function Block Diagram* – FBD). Nas linguagens de programação textuais temos a Lista de Instruções (*Instruction List* – IL) e Texto Estruturado (*Structured Text* – ST). Uma representação das cinco linguagens de programação é mostrada na figura 8.9.

Figura 8.9

Representação das cinco linguagens de programação.



A SFC estrutura a organização interna do programa, decompõe um problema de controle em partes facilmente gerenciáveis, mantendo a visão global da solução do problema. De outra forma, são passos ligados a blocos de ação e transições, em que cada passo representa um estado do sistema sob controle e a transição é associada à condição. Se verdadeira, desativa o passo anterior e ativa o seguinte.

A Lista de Instruções (IL) é uma linguagem textual, semelhante à linguagem *Assembler*, e, da mesma forma, seu programa é escrito com instruções abreviadas, chamadas mnemônicos. Alguns fabricantes disponibilizam a Lista de Instruções como uma segunda opção em um pacote que traz também outra linguagem de programação.

O Diagrama de Blocos (FBD), também chamado de Diagrama de Blocos de Funções, é amplamente utilizado na indústria de processos e de manufatura, expressando o comportamento das funções, programas e blocos de funções como um conjunto de gráficos interconectados. Assemelha-se à representação de um sistema de fluxo de sinais entre os elementos e expõe os blocos.

O Texto Estruturado (ST) é uma linguagem muito poderosa e vem substituir todas as linguagens declarativas, como Linguagem de Instruções, BASIC estruturado, pois tem suas raízes na linguagem Pascal e na linguagem C. Essa

linguagem pode ser usada na definição de blocos de funções complexas, com aplicação em qualquer outra linguagem e no detalhamento de ações e ainda em transições de um programa SFC.

A linguagem de programação Diagrama Ladder (LD) é uma linguagem gráfica, que tem como base os diagramas elétricos e que representa contatos e bobinas interconectados, destacando o fluxo de energização entre os elementos. É a linguagem mais difundida por ser semelhante aos esquemas elétricos usados para o comando convencional e pela facilidade de visualização pelo programador.

8.3.1 Programação em Ladder







O Diagrama Ladder utiliza lógica de relé, com contatos e bobinas, e é por esse motivo que pessoas que já têm conhecimento de circuitos elétricos, assimilam essa linguagem com muita facilidade. Esses diagramas já eram utilizados para documentar antigos blocos de relés, antes da existência dos CLPs. Ainda hoje se mantém como a mais usada, estando presente praticamente em todos os CLPs disponíveis no mercado.

Os vários circuitos são dispostos horizontalmente, com a bobina na extremidade direita, e alimentados por duas barras verticais laterais, formando uma figura que lembra uma escada. Por esse formato, recebe o nome de Ladder, que significa “escada”, em inglês. A cada linha horizontal está associada uma sentença lógica, sendo os contatos as entradas das sentenças, as bobinas, as saídas das sentenças, e a associação dos contatos, a lógica das sentenças.

Os símbolos básicos são indicados na figura 8.10.

Figura 8.10

Símbolos básicos do Ladder.

TIPO	SÍMBOLO	DIAGRAMA ELÉTRICO
CONTATO ABERTO		
CONTATO FECHADO		
SAÍDA		

Os operandos (nome genérico dos contatos e bobinas) no Ladder são identificados com um endereço da memória à qual se associa na lógica do CLP. Esse endereço de memória aparece no Ladder com um rótulo (*label*), nome simbólico para facilitar a programação, arbitrariamente escolhido pelo fabricante do CLP.

O estado de cada operando é representado em um *bit* correspondente na memória imagem (denominada palavra de *status*): este *bit* assume nível 1 (um) se o operando estiver acionado, e 0 (zero) quando não está acionado.

Se a bobina com endereço de saída estiver acionada, um par de terminais no módulo de saída apresentará condição de condução elétrica. Os contatos endereçados como entradas se acionam, enquanto seu respectivo par de terminais no módulo de entrada é acionado: fecham-se, se forem NA, e se abrem se forem NF.

Os contatos que irão energizar bobinas deverão ser de mesmo tipo do contato externo que aciona seu ponto no módulo de entrada e os que tiverem a função de desacionar as bobinas devem ser do tipo oposto do contato externo que os aciona. Veja quadro 8.1.

	Se a chave externa for	O contato no Ladder deve ser
Para ligar	NA	NA
	NF	NF
Para desligar	NA	NF
	NF	NA

Quadro 8.1

Relação entre a chave externa e o contato Ladder para ligar e desligar:

A lógica indica que a chave externa pode ser de qualquer tipo, desde que no diagrama Ladder se utilize o tipo de contato conveniente. Contudo, por segurança, não se deve utilizar chave externa NF para operação de ligar, nem NA para operação de desligar.

Na figura 8.11 temos um circuito exemplo básico para acionamento em Ladder:

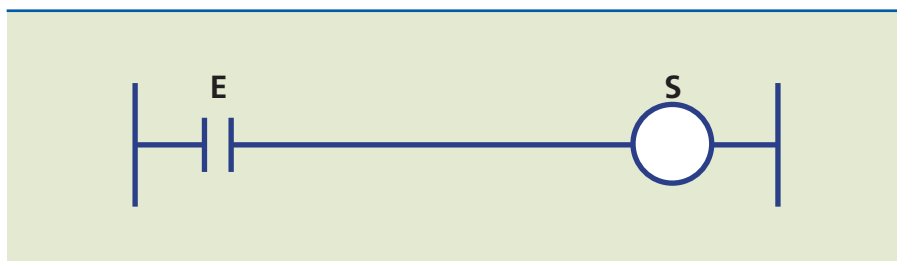


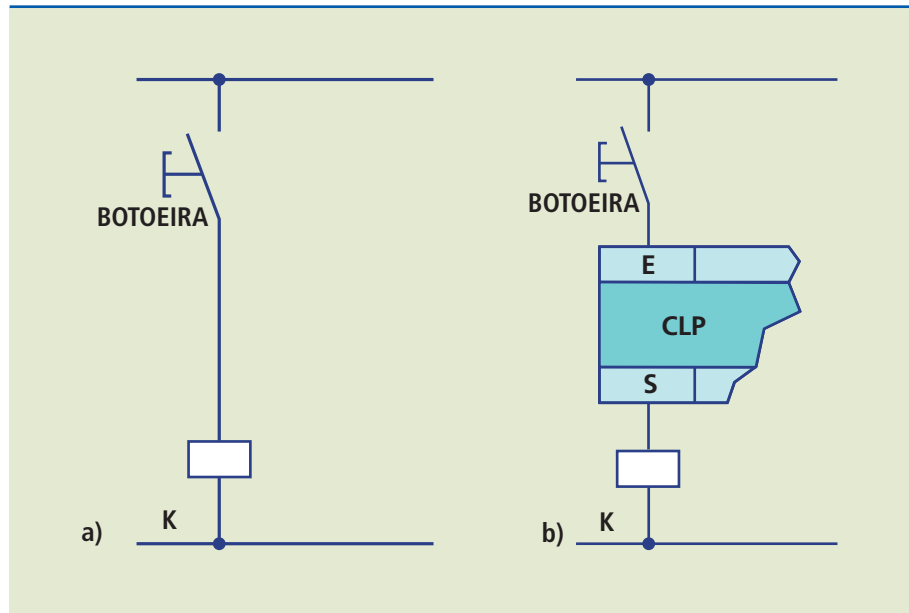
Figura 8.11

Circuito para análise de um acionamento em Ladder.

Ao analisarmos os módulos de entrada e saída do CLP, vemos que, quando o dispositivo (botoeira, sensor, fim de curso) ligado à entrada E fechar, este acionará o contato E, que estabelecerá uma continuidade de forma a acionar a bobina S (saída). Assim, o dispositivo ligado à saída S será acionado. Os termos E e S têm aqui fins didáticos para indicar entradas e saídas; cada fabricante utiliza uma nomenclatura diferente. Na figura 8.12, vemos a montagem física do Diagrama Ladder indicado na figura 8.11, em duas versões, com o circuito elétrico e com a ligação de entrada e saída ao CLP.

Figura 8.12

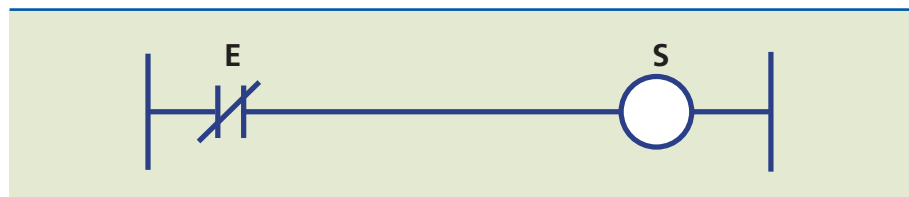
Circuito básico de acionamento: a) diagrama elétrico de comando; b) mesmo circuito com a ligação de entrada e saída ao CLP.



O contato normalmente fechado NF é um contato de negação (também chamado inversor), como pode ser visto no exemplo abaixo, similar ao programa anterior substituindo o contato NA por um NF (ver indicação na figura 8.13).

Figura 8.13

Acionamento de uma saída por um contato inversor NF.

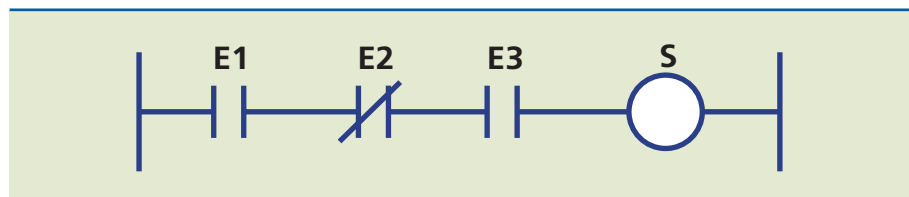


Na análise dos módulos de entrada e saída, vemos que, quando o dispositivo ligado à entrada E abrir, desacionará o contato E, o qual, por ser NF, estabelecerá uma continuidade elétrica de forma a acionar a bobina S. Portanto, o dispositivo ligado à saída S será acionado.

No diagrama Ladder, os contatos são associados para criar as lógicas E e OU com a saída. Quando em série, executam a lógica E, pois a bobina só é energizada quando todos os contatos estiverem fechados.

Figura 8.14

Associação de contatos em série em Ladder.



A saída S será acionada quando:

E1 estiver acionada e E2 estiver não acionada e E3 estiver acionada.

A lógica **OU** é conseguida com a associação paralela de contatos NA acionando a saída, desde que pelo menos uma das entradas esteja fechada.

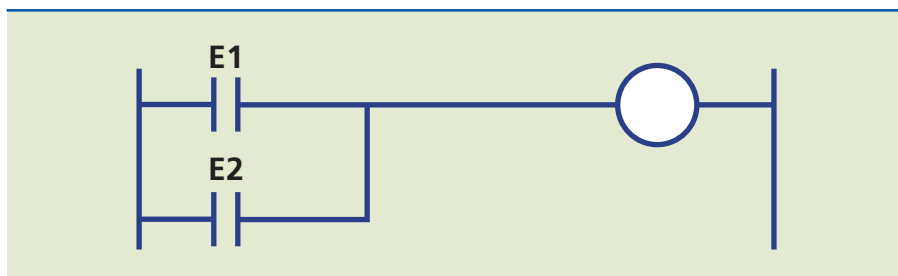


Figura 8.15

Associação de contatos em paralelo em Ladder.

Muitas vezes algumas resoluções obrigam a utilizar as associações mistas (figura 8.16), com soluções mais complexas, como a do exemplo a seguir.

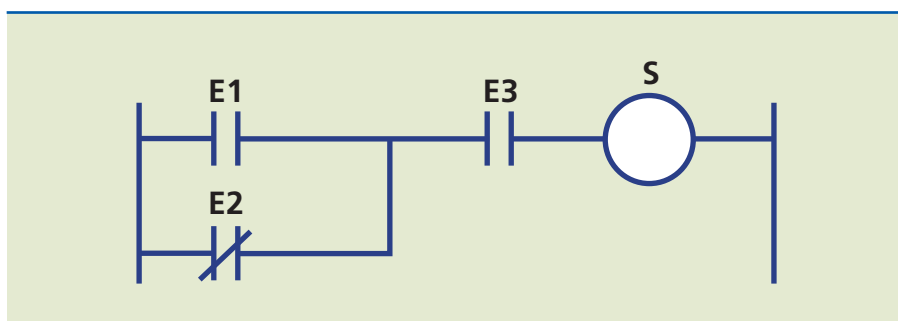


Figura 8.16

Associação mista de contatos em Ladder.

Em várias circunstâncias na programação em Ladder fazemos o uso da técnica do selo (figura 8.17), utilizando um contato da saída, que pode ser um relé interno do CLP ou uma memória auxiliar. Com isso, conseguimos manter a saída energizada usando uma técnica simples, mesmo que a entrada E seja um botão pulsador.

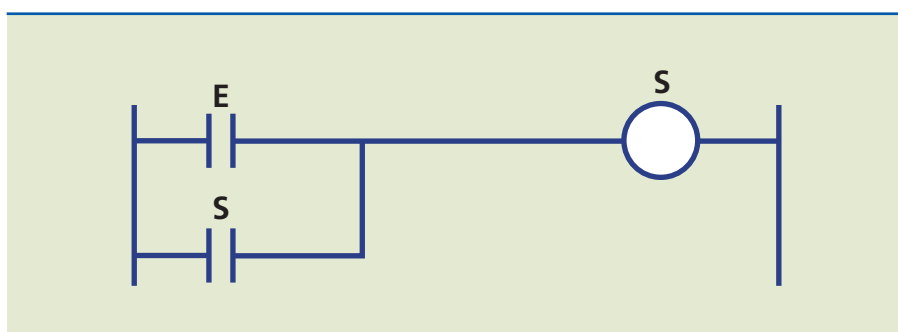


Figura 8.17

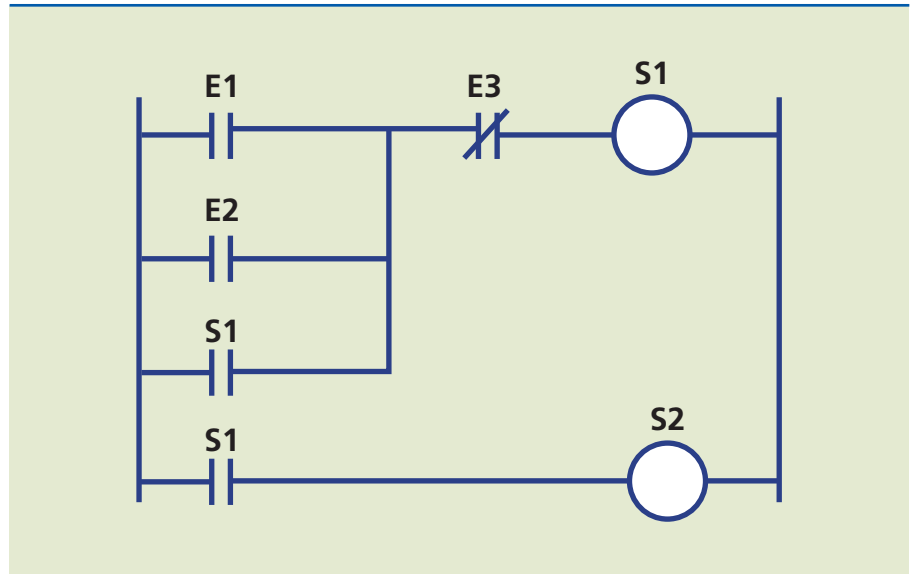
Técnica do selo utilizando a programação Ladder.

Exemplo

Elaborar um programa em Ladder que permita o acionamento de um motor quando acionarmos um botão pulsador E1 ou E2 e desligá-lo ao acionarmos um botão pulsador E3.

Solução:

Figura 8.18
Diagrama Ladder
para solução do
exemplo proposto.

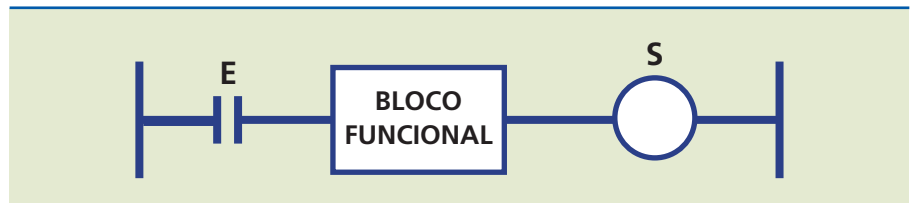


Ao acionarmos E1 ou E2, S1 é ligada e um primeiro contato S1 cria o selo ao fechar, mantendo a bobina S1 acionada, mesmo quando desacionamos E1 e E2. O segundo contato S1 também fecha, mantendo a bobina S2 ligada. Então, o motor entra em rotação, desligando somente quando acionamos o botão pulsador E3.

8.3.2 Algumas instruções básicas

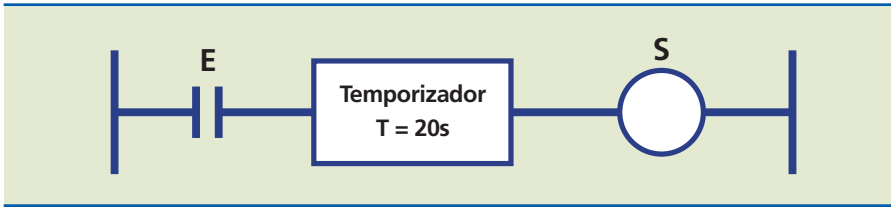
Por causa da grande quantidade de fabricantes de CLP, os blocos funcionais normalmente apresentam-se de diferentes formas, mas o princípio de funcionamento é o mesmo. Esses blocos têm a função de auxiliar ou completar o controle do equipamento, usando na lógica Ladder, instruções como contagem, temporização, instruções *SET*, *RESET*, soma, divisão, subtração, multiplicação, PID, conversão BCD/Decimal etc. Ver exemplo de indicação de bloco funcional no diagrama Ladder (figura 8.19).

Figura 8.19
Introdução de um
bloco funcional em um
diagrama Ladder:



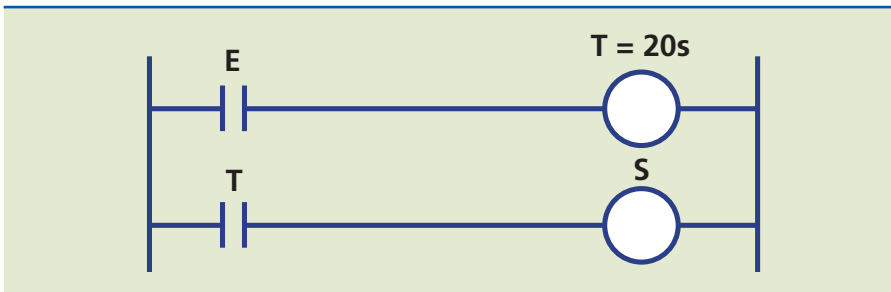
Temporização

O temporizador compara o intervalo de tempo transcorrido desde sua habilitação até este se igualar ao tempo preestabelecido. Quando completar a temporização, a CPU eleva a nível 1 um *bit* próprio na memória de dados e aciona a saída a ela associada.

**Figura 8.20**

Bloco funcional de temporização.

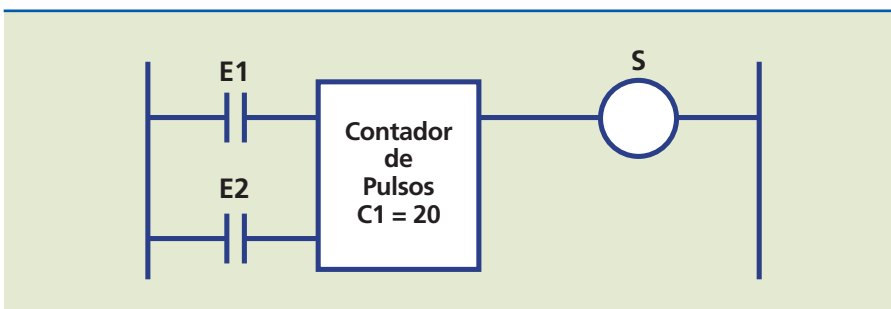
Na figura 8.20 há um exemplo de aplicação. Nesse caso, quando a entrada E é acionada, habilita-se o temporizador que após 20 segundos aciona a saída S. Quando E for desativada, o temporizador será desabilitado e desacionará a saída S. Normalmente, os temporizadores voltam para o valor inicial quando o sinal de condição é removido. Alguns são retentivos; significa que, apesar de o sinal da condição ter sido desativado, o temporizador armazena o valor alcançado e recomeçará a partir daquele valor, quando o sinal for novamente estabelecido. Para zerá-lo é necessário o uso da instrução RST. Em alguns CLPs, essa instrução apresenta duas entradas, uma de habilitação da contagem e outra para zera-mento ou *reset* da saída. Uma forma alternativa de programação para alguns CLPs é mostrada no diagrama Ladder (figura 8.21).

**Figura 8.21**

Outra forma de apresentação de temporização.

Contagem

Essa instrução é bloco funcional especial; no diagrama Ladder, ela aparece no meio de uma linha de programação e, usualmente, o contador após computar o número de eventos, deposita essa contagem em byte reservado. Quando a contagem for igual ao valor prefixado, essa instrução energizará um bit de contagem completa o qual é utilizado para energizar ou desenergizar um dispositivo. Os contadores são encontrados em diferentes formas e os mais comuns são: o contador *up*, o contador *down*, o contador bidirecional, o contador de duas fases A e B, o contador de 16 *bits*, o contador de 32 *bits* etc. Veja a linha de programação com um bloco funcional de contagem na figura 8.22.

**Figura 8.22**

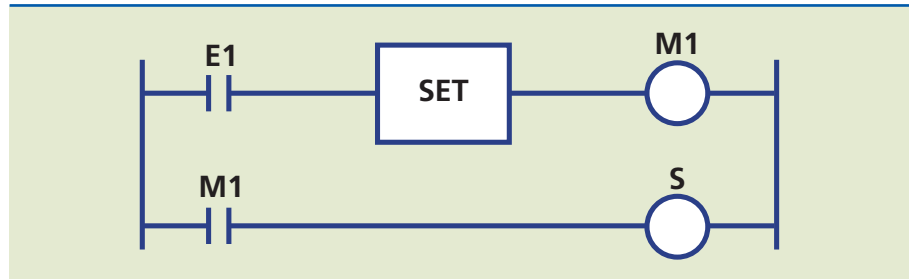
Linha de programação com bloco funcional de contagem.

Todos os contadores são retentivos e essa característica é essencial para sua operação, visto que eles contam o número de vezes que vão de OFF (desligado) para ON (ligado), através dos pulsos recebidos do contato que os controlam, para poder operar as saídas. Os contadores devem ser zerados após o evento, para desacionar as saídas que estão até então mantidas acionadas.

Instrução *SET*

A instrução *SET* é uma saída especial vinculada ativa; é ativada como resultado de um conjunto verdadeiro de condições. Ao ser acionada, todos os contatos de saída a ela associados (relés, memórias, saídas) permanecem acionados, como uma contadora. Ver exemplo na figura 8.23.

Figura 8.23
Instrução *SET*.

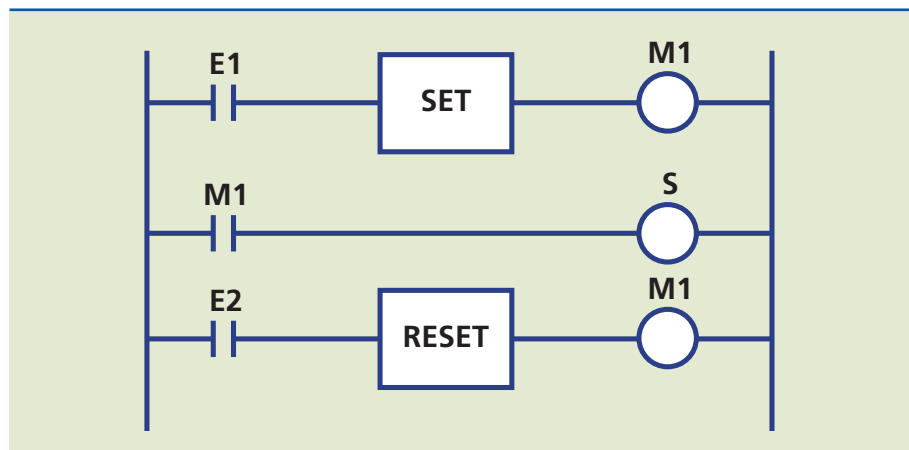


O exemplo da figura 8.23 mostra que a saída *S* não é ativada até que a entrada *E1* tenha sido ativada. O sinal de *E1* ativará a memória *M1*. Esse contato controla diretamente, mesmo que a entrada *E1* tenha sido desativada, *M1* permanece ativada e, conseqüentemente, *S* permanece ativada.

Instrução *RESET*

A instrução *RESET* é uma instrução especial vinculada ativa (desativa); é ativada como resultado de um conjunto verdadeiro de condições. Ao ser acionada a instrução *Reset*, todos os contatos da saída a ela associados (relés, memórias, saídas, contadores, temporizadores, *flags*) serão desacionados, como se fosse uma contadora. Ver figura 8.24, que contém as funções *SET* e *RESET*.

Figura 8.24
Linhas de programação
com as instruções
SET e *RESET*.



O esquema da figura 8.24 indica que a saída S é ativada pela instrução *SET*, ao ser acionado E1. Para desativar M1 e, conseqüentemente, S, basta que E2 seja acionado, introduzindo a instrução *RESET*.

8.4 Exercício para fixação de conceito

Vamos elaborar um exercício utilizando uma programação em Ladder, no qual uma lâmpada é ativada pelo botão pulsador E1, demorando 2 segundos para acender. A lâmpada permanece acesa durante 3 segundos e se desliga automaticamente, ficando assim por 2 segundos. Após esse período, acende novamente por mais 3 segundos, repetindo o ciclo inicial. O ciclo da lâmpada se repete automaticamente, a menos que seja encerrado manualmente ou que a lâmpada já tenha se desligado por 7 vezes. Quando a lâmpada desligar-se por 7 vezes, a entrada E2 deverá ser utilizada para resetar o contador.

Este exercício foi elaborado para programação em um controlador lógico programável que obedece aos parâmetros definidos neste capítulo. Em um CLP específico, teremos outras disposições de instruções, blocos lógicos, saídas etc., que variam de um fabricante para outro. Observar ainda que nesta resolução foram utilizadas praticamente todas as instruções apresentadas neste capítulo.

Solução:

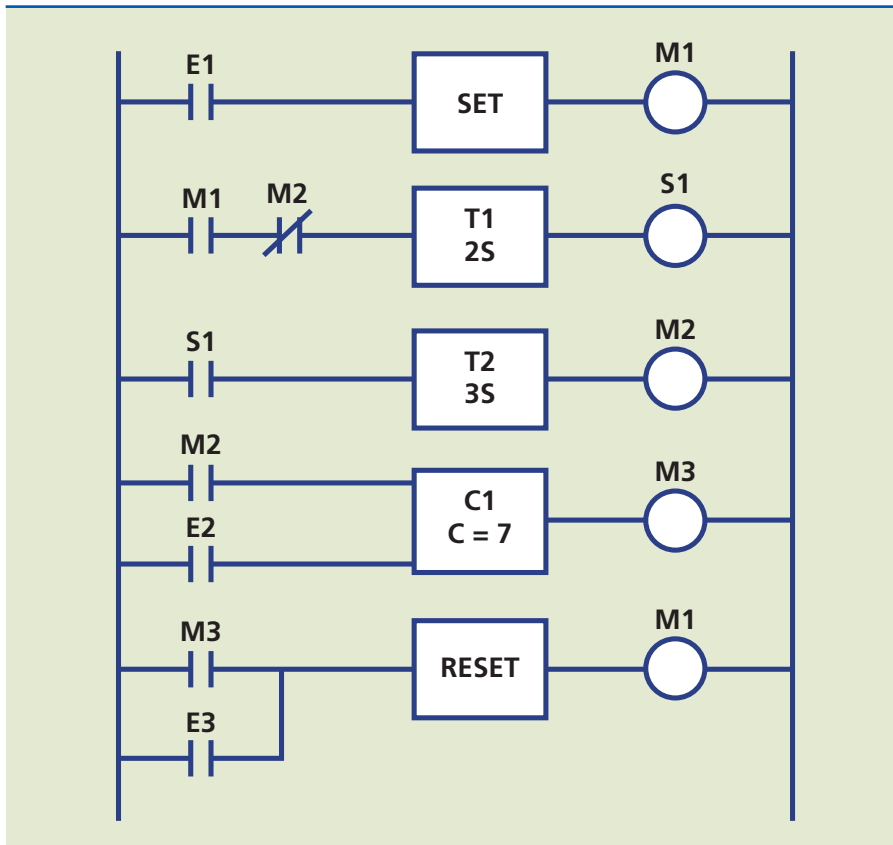


Figura 8.25

O exemplo (figura 8.25) mostra que, ao ser acionado o botão pulsador E1, a memória M1 fica ativada pela instrução *SET* e o contato M1 fecha, energizando o temporizador com retardo no acionamento T1. Após 2 segundos, a lâmpada S1 acende e aciona o fechamento de seu contato S1, que energiza o temporizador com retardo no desligamento T2. Ao terminar a contagem de tempo de 3 segundos, a memória M2 é acionada e ativa os dois contatos M2. A lâmpada S1 apaga ao abrir o contato M2 normalmente fechado, que desativa T1; o contato M2 normalmente aberto emite um pulso para o contador C1, que faz a primeira contagem de desligamento da lâmpada. O ciclo se repete por 7 vezes, quando então C1 ativa a memória M3, cujo contato aciona a instrução *RESET*, que desaciona M1, apagando definitivamente a lâmpada. Caso se deseje interromper o ciclo de funcionamento, basta acionar E3; o contador C1 é zerado com o botão pulsador E2.