LAB4
Introdução aos Controladores Lógicos Programáveis

4.1 Introdução

Os Controladores Lógicos Programáveis (CLPs) são dispositivos digitais, muito utilizados na indústria, capazes de armazenar instruções para implementação de funções de controle (tais quais seqüência lógica, temporização e contagem), bem como realizar operações lógicas e aritméticas, manipulação de dados e comunicação em rede, sendo utilizados no controle de sistemas automatizados (Georgini, 2006). Seus principais componentes são a unidade central de processamento (CPU), os módulos de I/O (ou módulos de entrada/ saída), a fonte de alimentação e a base.

A CPU do CLP compreende o microprocessador, o sistema de memória (ROM e RAM) e os circuitos auxiliares de controle. Os módulos de I/O são dispositivos através dos quais podemos conectar sensores, atuadores ou outros equipamentos à CPU do CLP; assim, a CPU pode ler sinais de entrada, ou enviar sinais para a saída do CLP através dos módulos de I/O. Esses módulos podem ser discretos ou analógicos. A fonte de alimentação é responsável pela tensão de alimentação fornecida à CPU e aos módulos de I/O. A base do CLP proporciona conexão mecânica e elétrica entre a CPU, os módulos de I/O e a fonte. Ela contém o barramento de comunicação entre eles, em que estão presentes os sinais de dados, endereço, controle e tensão de alimentação (Georgini, 2006).

A programação de um CLP pode ser feita através de uma variedade de linguagens. Uma das mais populares é a linguagem Ladder, tendo recebido este nome devido à sua semelhança com uma escada (ladder), na qual duas barras verticais paralelas são interligadas pela lógica de controle, formando os degraus (rungs) da escada (Georgini, 2006). Na Figura 4.1 temos uma representação de lógica de controle através da linguagem ladder:

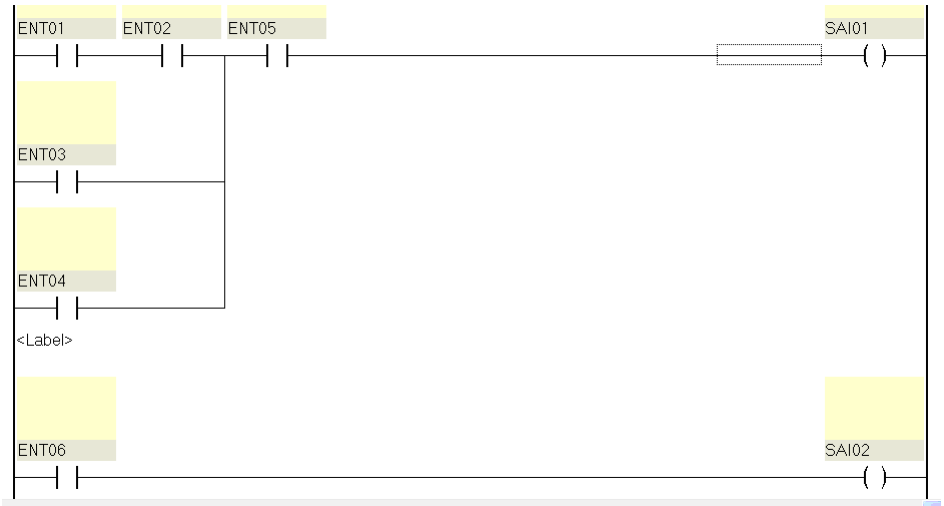

Figura 4.1: Exemplo de diagrama Ladder

O diagrama da Figura 4.1 apresenta uma lógica de controle com dois rungs: o primeiro é formado por 3 linhas (primeira linha – ENT01, ENT02, ENT05 e SAI01; segunda linha – ENT03; terceira linha – ENT04) e o segundo é formado por uma linha (ENT06 e SAI02). Este diagrama é formado por contatos e bobinas. As

entradas são os contatos, e as saídas são as bobinas. Cada elemento da lógica de controle representa uma instrução da linguagem Ladder, sendo alocado em um endereço específico e consumindo uma determinada quantidade de memória disponível para armazenamento do programa de aplicação (Georgini, 2006). Os contatos utilizados nas entradas do diagrama da Figura 4.1 são chamados contatos normalmente abertos (NA). Nesses contatos, há passagem de corrente se o contato for fechado. Assim, se o contato for uma botoeira, o contato será fechado se o botão for pressionado, permitindo a passagem de corrente; enquanto o botão não estiver apertado, o contato estará aberto, e não haverá passagem de corrente. Já nos contatos normalmente fechados (NF) ocorre o contrário: há passagem de corrente se o contato for aberto; enquanto o contato estiver fechado, não há passagem de corrente.

contato normalmente aberto (NA) contato normalmente fechado (NF)

Figura 4.2: Tipos de contatos

O fluxo de corrente elétrica fictícia num diagrama Ladder sempre ocorre da extremidade esquerda para a extremidade direita. Assim, no diagrama da Figura 4.1, se os contatos das entradas 1, 2 e 5 forem fechados simultaneamente, haverá passagem de corrente fictícia pela linha 1 do rung 1. Então, a bobina SAI01 será energizada, acionando a saída 1. Se forem fechados apenas os contatos ENT01 e ENT05, mantendo ENT02 aberto, não haverá a passagem de corrente fictícia pela linha 1 do rung 1, e então a saída 1 não será acionada. Outros modos de acionar a saída 1 seriam fechar os contatos ENT03 e ENT05 simultaneamente, ou ainda fechar os contatos ENT04 e ENT05 simultaneamente. A seqüência de leitura do programa em linguagem Ladder é do rung superior para o rung inferior; dessa forma, no exemplo da figura, se fecharmos os contatos ENT03, ENT05 e ENT06, primeiro será acionada a saída SAI01, e depois a saída SAI02.

4.2 Elementos de linguagem Ladder

4.2.1 Lógica básica

Uma instrução lógica ‘E’ é implementada do seguinte modo na linguagem Ladder.

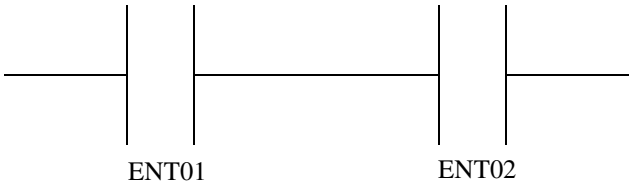

Figura 4.3: ‘E’ lógico em Ladder

Assim, se fecharmos os contatos ENT01 ‘E’ ENT02, haverá passagem de corrente pelo rung. Já a instrução lógica ‘OU’ é implementada como dois contatos em paralelo (Figura 4.4)..

Deste modo, se fecharmos o contato ENT01 ‘OU’ o contato ENT02, haverá passagem de corrente pelo rung.

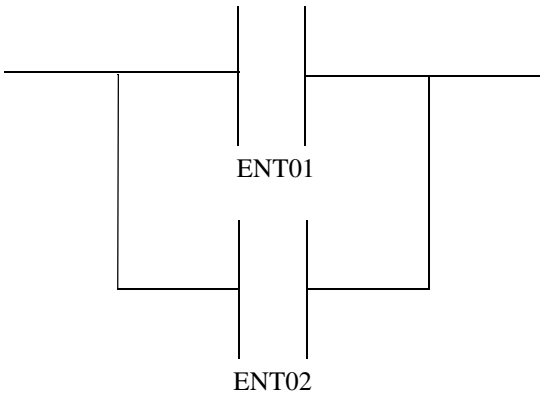

Figura 4.4: 'OU' lógico em Ladder

4.2.2 Temporizadores

Os temporizadores, como o nome sugere, são utilizados para temporização de condições no diagrama Ladder. O temporizador é identificado pela letra T seguida dos dígitos correspondentes ao seu endereço: T0, T1, etc. Há um bit de status relacionado ao temporizador, o qual é ativado quando o valor atual do temporizador for maior ou igual ao valor de preset (valor pré-configurado). O incremento de tempo depende do temporizador utilizado. O temporizador abaixo apresenta uma entrada de controle (enable) que, ao ser acionada (rung = 1), habilita o início da temporização, e ao ser desligada (rung = 0), reinicia o temporizador, mantendo-o nesta condição até novo acionamento da entrada enable (Georgini, 2006).

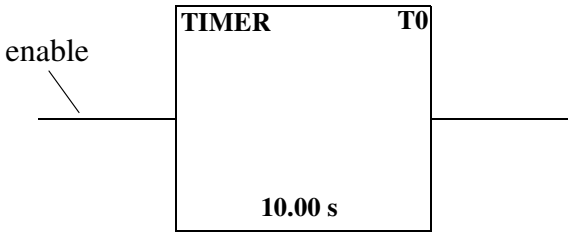

Figura 4.5: Temporizador

Na lógica de controle da Figura 4.6, quando os contatos BOT01 e BOT02 são fechados e permanecem fechados simultaneamente, é iniciada a temporização de T1, que tem um valor de preset fixo em 10 segundos (este temporizador apresenta incremento de tempo de 0.01 s). Ao ser atingido o valor de preset, o bit de status de T1 é acionado, ativando a saída SAI01. O bit de status permanece ativado até que o temporizador seja desativado (ou seja, quando um dos contatos BOT01 ou BOT02 for aberto). Esse bit pode ser associado a um contato normalmente aberto, conforme podemos observar na figura.

Também é possível ter acesso ao valor atual do temporizador durante a execução do diagrama Ladder utilizando-se os bits de valor atual. Por exemplo, TW0 armazena o valor atual de T0, TW1 armazena o valor atual de T1, e assim por diante, sendo que esses dados podem ser utilizados na lógica de controle.

A Figura 4.7 ilustra a utilização conjunta de temporizadores e comparadores. Assim, ao ser fechado o contato BOT01, é iniciada a temporização de T1. Quando o valor atual do temporizador T1 for maior ou igual a 1s, a saída SAI01 é acionada. Quando o valor atual de T1 for maior ou igual a 2s, a saída SAI02 é acionada, e quando esse valor atingir 3s, a saída SAI03 é acionada. O temporizador segue sua ação até atingir 10s, quando seu valor atual é zerado até que ele seja habilitado novamente através do acionamento de BOT01.

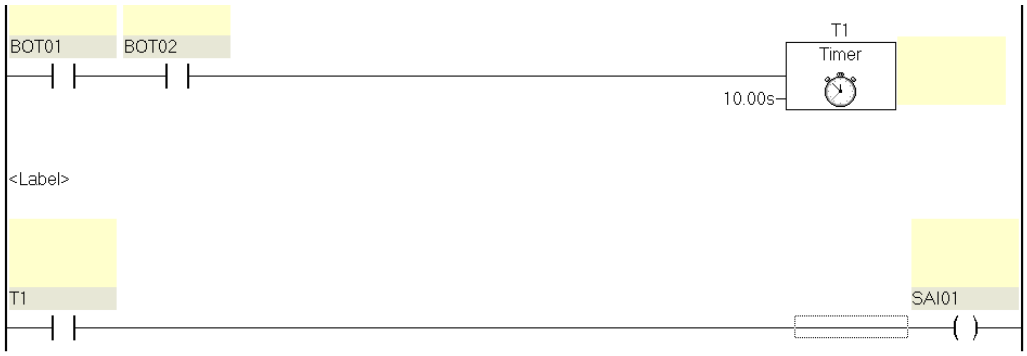

Figura 4.6: Lógica com temporizador

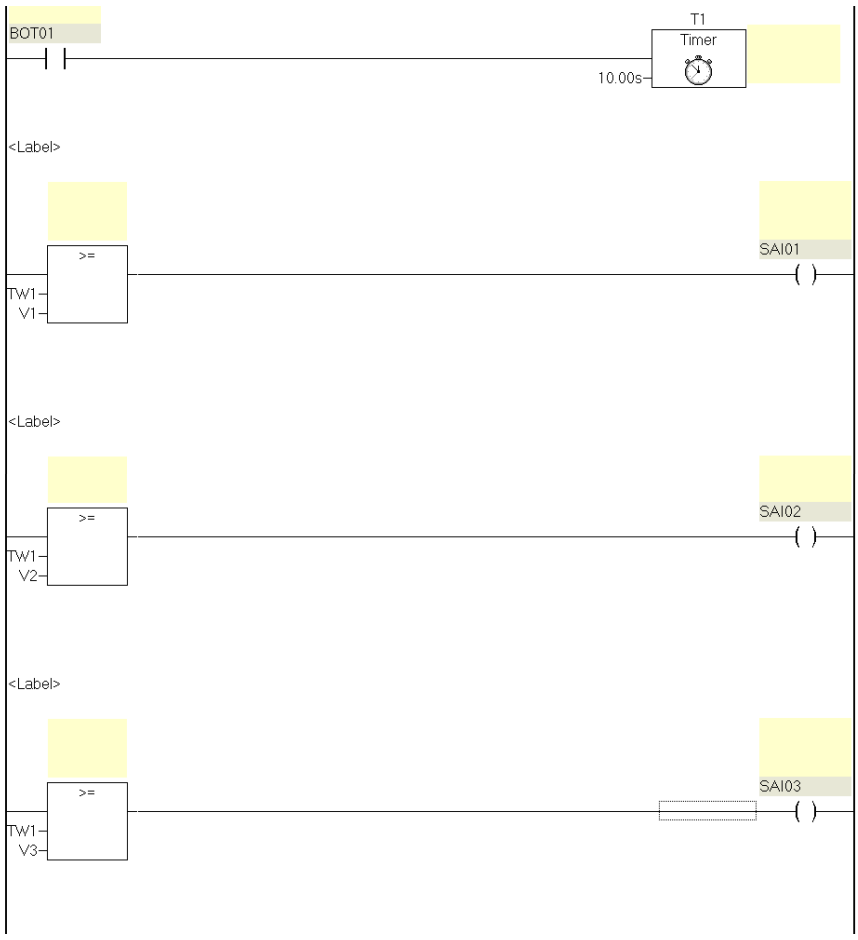

Figura 4.7: Lógica com temporizador e comparadores

4.2.3 Contadores

Os contadores são utilizados para contagem de condições ou eventos controlados pelo programa de aplicação. O contador é identificado pela letra C seguida do número do contador (C0, C1, C2, etc.). Há um bit de status relacionado ao contador, que é ativado quando o valor atual deste atingir o valor de preset (valor pré-configurado) (Georgini, 2006).

@ ERRRO!!!!! @Na lógica de controle da Figura 4.8, a cada transição de 0 para 1 (ou seja, off on) da entrada BOT01, o valor atual de C1 é incrementado em uma unidade, apresentando valor de preset de 20. Quando o valor de preset é atingido, o bit de status de C1 é ativado, acionando a saída SAI01. Esse bit de sta-

tus permanece ativado durante um ciclo do programa; no loop seguinte, o valor do bit de status de C1 é zerado automaticamente.

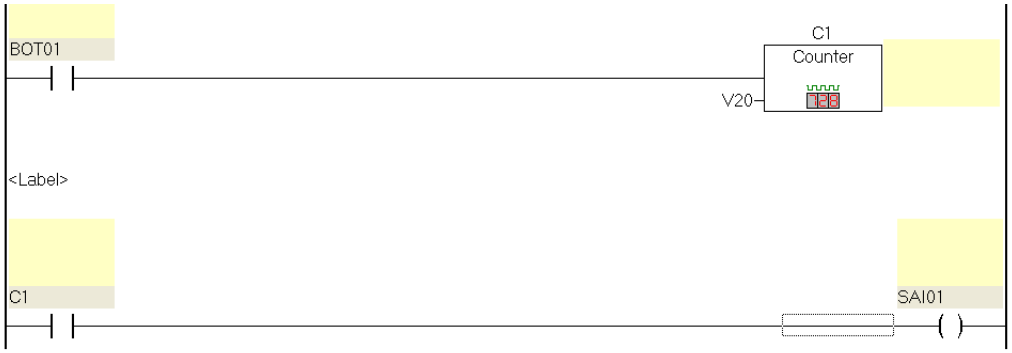

Figura 4.8: Lógica com contador

Além de ativar o bit de status quando atinge o valor de preset, o contador também oferece acesso ao seu valor atual durante a execução do programa de aplicação. Assim, a variável CW0 armazena o valor atual de C0, CW1 armazena o valor atual de C1, e assim por diante. Essas variáveis podem ser utilizadas na construção da lógica de controle.

A Figura 4.9 apresenta uma seqüência lógica envolvendo um contador e comparadores.

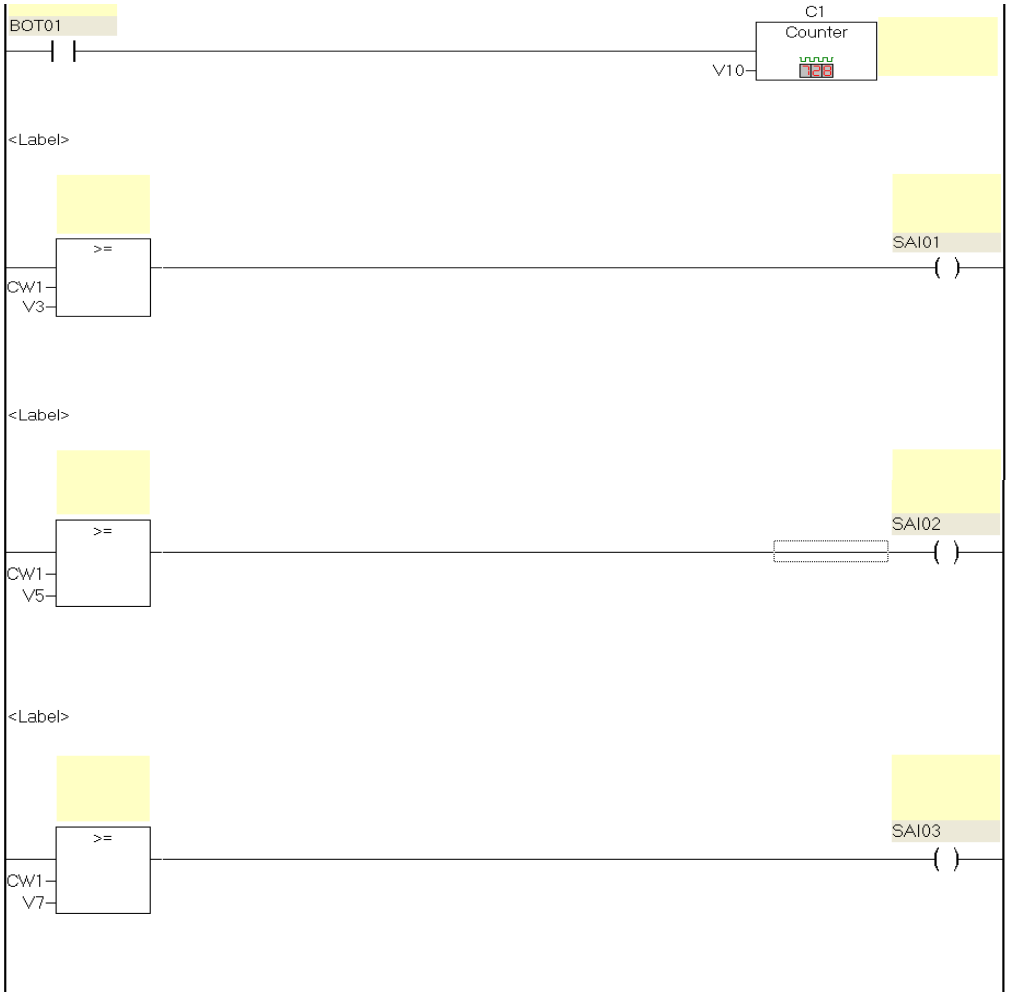

Figura 4.9: Lógica com contador e comparadores

Na lógica de controle da Figura 4.9, a cada transição de 0 para 1 do contato BOT01, o valor atual de C1 é incrementado em 1 unidade. Ao atingir a contagem de 3, a saída SAI01 é acionada; ao atingir a contagem de 5, a saída SAI02 é acionada, e ao atingir a contagem de 7, a saída SAI03 é acionada.

4.2.4 Funções SET e RESET

Ao ser executada (ou seja, quando o rung ligado à bobina set for igual a 1), a função set aciona o operando controlado, mantendo-o acionado mesmo que o rung não permaneça acionado (rung = 0). O operando acionado pela função set pode ser desligado pela função reset; esta função o manterá desligado mesmo que o rung não permaneça acionado. Se as instruções set e reset forem executadas durante o mesmo ciclo sobre o mesmo operando, terá controle final sobre o operando aquela que for executada por último (Georgini, 2006).

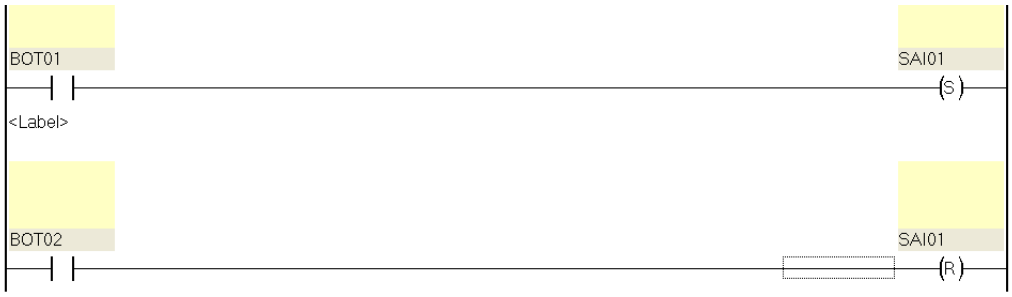

Figura 4.10: Lógica com SET e RESET

Na lógica de controle da Figura 4.10, ao ser pressionado o botão BOT01, a saída SAI01 será acionada, permanecendo nesta condição mesmo que BOT01 não continue sendo apertado. Ao ser pressionado BOT02, a saída SAI01 é desligada, permanecendo assim mesmo que BOT02 não continue sendo apertado. Se os botões BOT01 e BOT02 forem pressionados simultaneamente, a saída SAI01 é desligada, já que a instrução RESET será executada por último.

4.2.5 Como iniciar uma seção de programação

Primeiramente, deve-se iniciar o programa FST 4.10 da FESTO clicando-se no seu ícone, presente na área de trabalho do Windows. Uma vez iniciado o programa, clicar no item *'New'* do menu *'Project'*. Aparecerá uma tela pedindo um nome para o programa que será construído. Deve-se escolher um nome e clicar em *'ok'*. Aparecerá outra janela, *'Project Settings'*, perguntando o tipo de controlador a ser utilizado. Deve-se escolher a opção '*FEC Compact'* (veja a Figura 4.11):.

No item *'Project Settings'* , é possível adicionar comentários ao programa. Para começarmos a construir o diagrama ladder, devemos clicar no item *'New'* do menu *'Program'*. Aparecerá uma janela perguntando o tipo de programa a ser elaborado. São duas possibilidades: diagrama ladder e lista de instruções. Trataremos aqui apenas da primeira opção. Seleciona-se então a opção diagrama ladder. Outra janela de opções se abre, onde é possível definir a versão e o número do programa, bem como adicionar comentários a ele. Clicando-se em 'ok' , aparecerá um diagrama ladder em branco:

Ao se escolher o item *'Shortcuts'* do menu *'View'* aparece na tela uma barra de ferramentas. Esta barra tem as principais funções a serem utilizadas na construção de um diagrama ladder.

A opção *'Help'* do programa oferece a documentação completa, tanto do CLP como do software de programação.

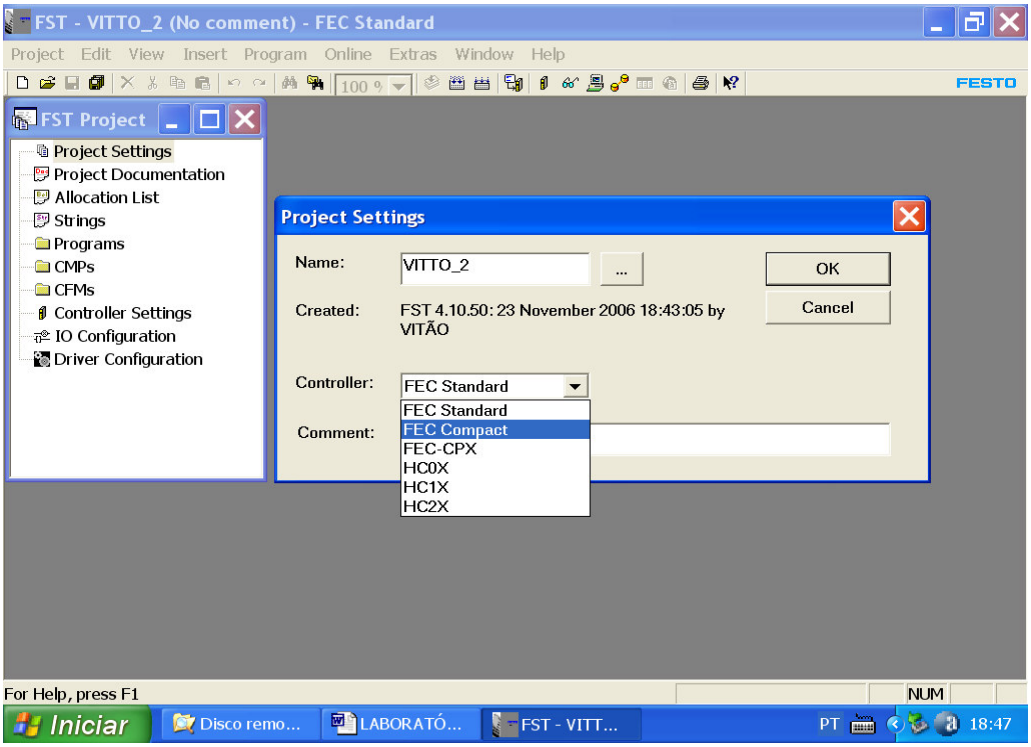

Figura 4.11: O software de programação do CLP FESTO

4.3 Exercícios

4.3.1 Exercícios simples

a) Deseja-se acender e apagar uma lâmpada através de um botão. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) Um dispositivo de uma indústria metalúrgica tem como função a fixação de peças em um molde. Esta fixação é feita por um atuador linear de dupla ação que avança mediante o acionamento de dois botões (S1 e S2) e retorna caso os botões sejam desacionados. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

c) Verificar através de botões e de uma lâmpada a tabela da verdade da função ou-exclusivo. Elabore um programa em linguagem Ladder e teste no CLP.

4.3.2 Tanque industrial

a) Elabore um diagrama Ladder simplificado para encher ou esvaziar um tanque industrial por meio de duas eletroválvulas. A eletroválvula V1 permite a entrada de líquido e a V2 permite o escoamento de saída. Quando o líquido atinge o nível máximo do tanque, um sensor A envia um sinal para o circuito lógico. Abaixo do nível máximo o sensor A não envia sinal algum. Há ainda um botão B, que deve encher o tanque quando for acionado e esvaziar em caso contrário. O esquema do tanque está apresentado na Figura 4.12 e as convenções do funcionamento do sistema estão apresentados na Tabela 4.13.

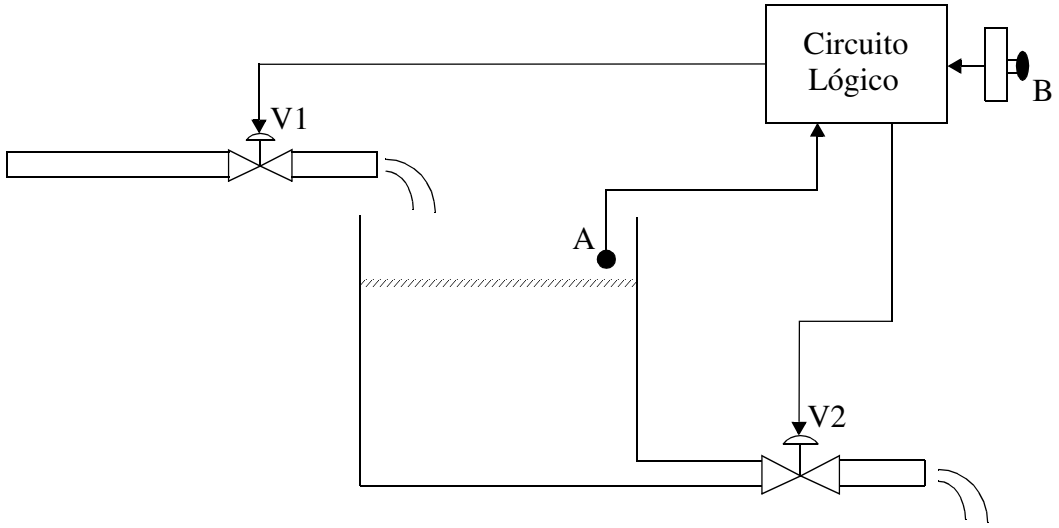

Figura 4.12: Esquema do tanque industrial

Tabela 4.13: Convenções para o tanque industrial

Sinal	Significado
A=1	Tanque cheio
A=0	Tanque não cheio
B=1	Comando encher
B=0	Comando esvaziar
V1=1	Comando fechar V1
V1=0	Comando abrir V1
V2=1	Comando fechar V2
V2=0	Comando abrir V2

4.3.3 Usando contadores

Há várias modalidades de contagem que podem ser utilizadas no CLP.

a) Deseja-se acender uma lâmpada após um botão ser acionado cinco vezes. Outro botão apaga a lâmpada (se ela estiver acesa) e reinicia a contagem. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

4.3.4 Exercícios com temporizadores

O CLP da FESTO também possui diferentes tipos de temporizadores. Consulte a documentação do programa para escolher o mais adequado a cada tipo de problema.

a) Deseja-se acender uma lâmpada de alarme durante 10s, quando um botão de emergência é acionado. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) O alarme A de uma casa é ativado por um sensor de movimento M e por um sensor de abertura de janelas J. O sensor M ativa o alarme quando detecta a presença de pessoas. O sensor J ativa o

alarme quando a janela é aberta. Há ainda um botão B para ligar ou desligar o alarme. Supondo que o alarme deve ser acionado por 10s e depois desligar automaticamente, elabore um diagrama ladder simplificado para resolver este problema. As convenções estão indicadas na Tabela4.14.

Tabela 4.14: Convenções para o sistema de alarme

Sinal	Significado
B=0	Comando desligar alarme
B=1	Comando ligar alarme
M=0	Ausência de pessoas
M=1	Presença de pessoas
J=0	Janela aberta
J=1	Janela fechada

c) Um sistema de dois semáforos controla o tráfego de um cruzamento de duas ruas (rua A e rua B), conforme a Figura 4.15,, sendo que cada semáforo está posicionado numa das ruas. A seqüência de

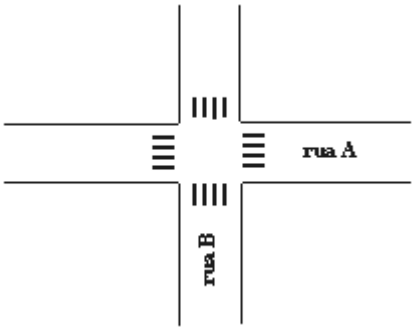

Figura 4.15: Cruzamento de ruas

acionamento de cada fase (amarelo, vermelho e verde) dos semáforos é mostrada na Tabela 4.16.

Tabela 4.16: Temporização

Fase	Tempo (s)	Semáforo A	Semáforo B
1	10	Verde	Vermelho
2	3	Amarelo	Vermelho
3	2	Vermelho	Vermelho
4	10	Vermelho	Verde
5	3	Vermelho	Amarelo
6	2	Vermelho	Vermelho

Implemente o semáforo em um programa em linguagem Ladder e teste no CLP.

d) Seja um sistema de semáforos similar ao anterior, controlando o tráfego no cruzamento de duas vias, conforme a Figura 4.17. Há um semáforo na rua A e um na rua B (as setas indicam o sentido do tráfego em cada rua). A diferença é que agora há quatro semáforos adicionais (S1, S2, S3 e S4), sendo dois deles dotados de botão para travessia de pedestres (S3 e S4). Cada um está posicionado

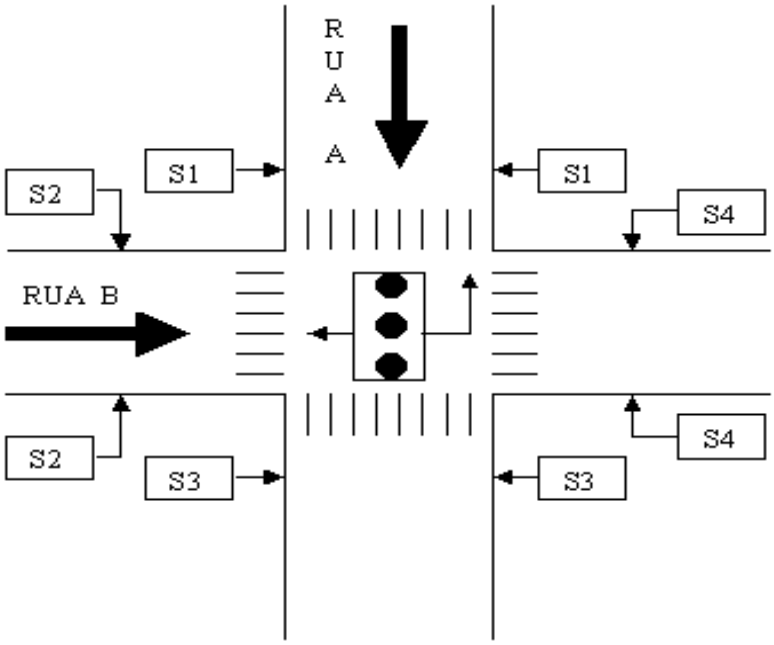

Figura 4.17: Cruzamento de ruas com travessia de pedestres

de um dos lados do cruzamento entre as ruas A e B. Ao acionamento do botão de S3 ou S4, os semáforos das ruas A e B devem passar de verde para amarelo, e deste para vermelho, a menos que já esteja(m) fechado(s) ou em processo de fechamento. Os estados correspondentes às fases dos semáforos são descritos na Tabela 4.18.

Tabela 4.18: Temporização

Fase	Tempo (s)	Semaf. A	S1	S3	Semaf. B	S2	S4
1	15	Verde	Vermelho	Vermelho	Vermelho	Verde	Vermelho
2	3	Amarelo	Vermelho	Vermelho	Vermelho	Verde	Vermelho
3	2	Vermelho	Vermelho	Vermelho	Vermelho	Vermelho*	Vermelho
4	15	Vermelho	Verde	Vermelho	Verde	Vermelho	Vermelho
5	3	Vermelho	Verde	Vermelho	Amarelo	Vermelho	Vermelho
6	2	Vermelho	Vermelho*	Vermelho	Vermelho	Vermelho	Vermelho
7	10	Vermelho	Verde	Verde	Vermelho	Verde	Verde
8	5	Vermelho	Vermelho*	Vermelho*	Vermelho	Vermelho*	Vermelho*

Nesta tabela, *Vermelho** indica vermelho piscante. As fases 7 e 8 só poderão ocorrer se durante as fases 1 a 6 houver o acionamento de pelo menos um botão para travessia de pedestres. Se isso ocorrer, o semáforo (A ou B) que estiver verde deve passar para amarelo (e ficar nessa fase durante 3 segundos), a menos que já esteja no amarelo. Após isso, passa-se para a fase 7 e, posteriormente, para a fase 8. Terminada a fase 8, volta-se à fase 1. Se nenhum botão for pressionado, a seqüência de fases é 1-2-3-4-5-6-1-2-etc. Implemente este semáforo em um programa em linguagem Ladder e teste no CLP.

4.3.5 Usando contadores e temporizadores

a) Deseja-se engarrafar bebidas de modo automático utilizando-se um CLP. As garrafas movimentam-se através de uma esteira rolante acionada por um motor elétrico, o qual é ligado e desligado pelo CLP, conforme a Figura 4.19. Quando cinco garrafas passarem por um sensor de presença (Sensor A), o motor deve ser desligado e um conjunto de cinco bicos injetores de cerveja deve ser acionado por 10 segundos (para encher as garrafas); após esses 10 segundos, o motor da esteira deve voltar a movimentá-la, até que outras cinco garrafas vazias passem pelo Sensor A; quando isso ocorrer, o processo se repetirá.

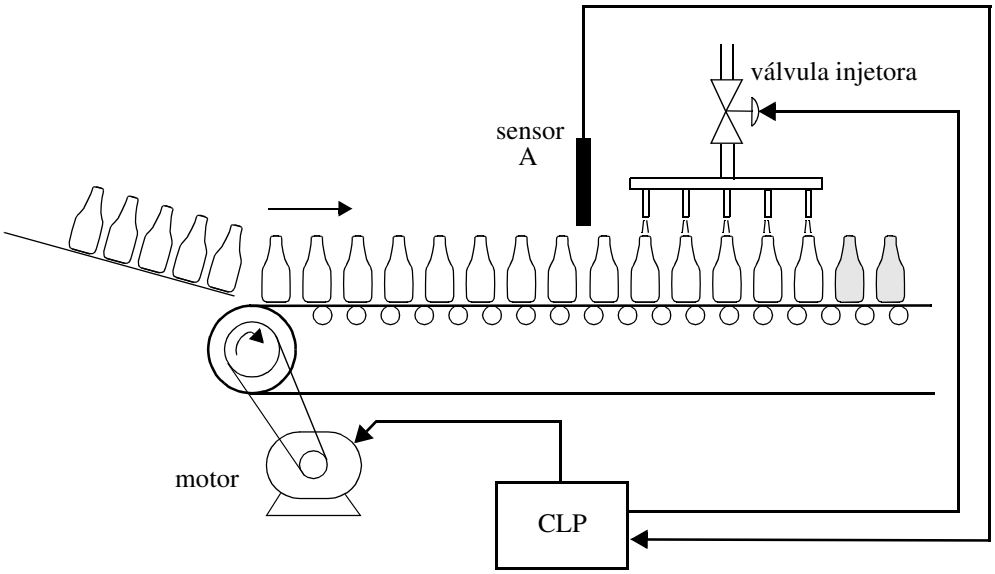

Figura 4.19: Engarrafadora

RPM/2009

LAB4
Introdução aos Controladores Lógicos Programáveis

4.1 Introdução

Os Controladores Lógicos Programáveis (CLPs) são dispositivos digitais, muito utilizados na indústria, capazes de armazenar instruções para implementação de funções de controle (tais quais seqüência lógica, temporização e contagem), bem como realizar operações lógicas e aritméticas, manipulação de dados e comunicação em rede, sendo utilizados no controle de sistemas automatizados (Georgini, 2006). Seus principais componentes são a unidade central de processamento (CPU), os módulos de I/O (ou módulos de entrada/ saída), a fonte de alimentação e a base.

A CPU do CLP compreende o microprocessador, o sistema de memória (ROM e RAM) e os circuitos auxiliares de controle. Os módulos de I/O são dispositivos através dos quais podemos conectar sensores, atuadores ou outros equipamentos à CPU do CLP; assim, a CPU pode ler sinais de entrada, ou enviar sinais para a saída do CLP através dos módulos de I/O. Esses módulos podem ser discretos ou analógicos. A fonte de alimentação é responsável pela tensão de alimentação fornecida à CPU e aos módulos de I/O. A base do CLP proporciona conexão mecânica e elétrica entre a CPU, os módulos de I/O e a fonte. Ela contém o barramento de comunicação entre eles, em que estão presentes os sinais de dados, endereço, controle e tensão de alimentação (Georgini, 2006).

A programação de um CLP pode ser feita através de uma variedade de linguagens. Uma das mais populares é a linguagem Ladder, tendo recebido este nome devido à sua semelhança com uma escada (ladder), na qual duas barras verticais paralelas são interligadas pela lógica de controle, formando os degraus (rungs) da escada (Georgini, 2006). Na Figura 4.1 temos uma representação de lógica de controle através da linguagem ladder:

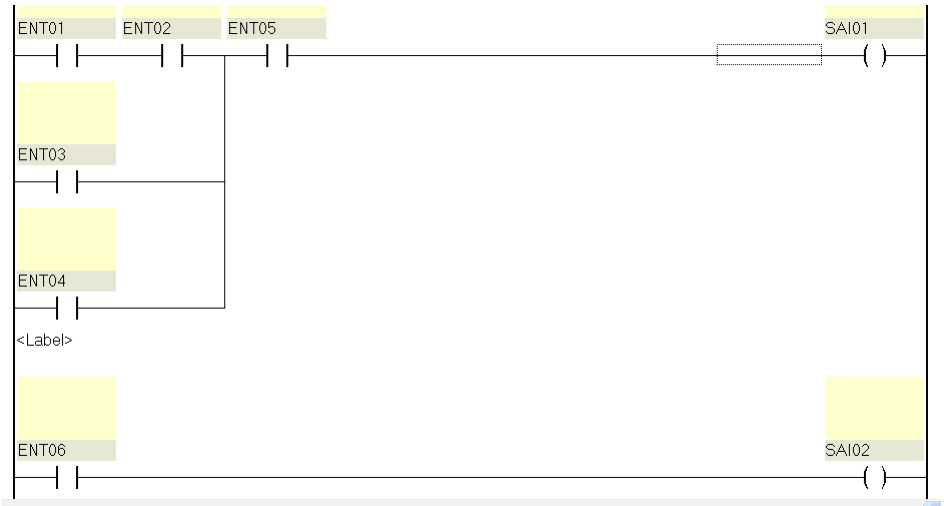

Figura 4.1: Exemplo de diagrama Ladder

O diagrama da Figura 4.1 apresenta uma lógica de controle com dois rungs: o primeiro é formado por 3 linhas (primeira linha – ENT01, ENT02, ENT05 e SAI01; segunda linha – ENT03; terceira linha – ENT04) e o segundo é formado por uma linha (ENT06 e SAI02). Este diagrama é formado por contatos e bobinas. As

entradas são os contatos, e as saídas são as bobinas. Cada elemento da lógica de controle representa uma instrução da linguagem Ladder, sendo alocado em um endereço específico e consumindo uma determinada quantidade de memória disponível para armazenamento do programa de aplicação (Georgini, 2006). Os contatos utilizados nas entradas do diagrama da Figura 4.1 são chamados contatos normalmente abertos (NA). Nesses contatos, há passagem de corrente se o contato for fechado. Assim, se o contato for uma botoeira, o contato será fechado se o botão for pressionado, permitindo a passagem de corrente; enquanto o botão não estiver apertado, o contato estará aberto, e não haverá passagem de corrente. Já nos contatos normalmente fechados (NF) ocorre o contrário: há passagem de corrente se o contato for aberto; enquanto o contato estiver fechado, não há passagem de corrente.

contato normalmente aberto (NA) contato normalmente fechado (NF)

Figura 4.2: Tipos de contatos

O fluxo de corrente elétrica fictícia num diagrama Ladder sempre ocorre da extremidade esquerda para a extremidade direita. Assim, no diagrama da Figura 4.1, se os contatos das entradas 1, 2 e 5 forem fechados simultaneamente, haverá passagem de corrente fictícia pela linha 1 do rung 1. Então, a bobina SAI01 será energizada, acionando a saída 1. Se forem fechados apenas os contatos ENT01 e ENT05, mantendo ENT02 aberto, não haverá a passagem de corrente fictícia pela linha 1 do rung 1, e então a saída 1 não será acionada. Outros modos de acionar a saída 1 seriam fechar os contatos ENT03 e ENT05 simultaneamente, ou ainda fechar os contatos ENT04 e ENT05 simultaneamente. A seqüência de leitura do programa em linguagem Ladder é do rung superior para o rung inferior; dessa forma, no exemplo da figura, se fecharmos os contatos ENT03, ENT05 e ENT06, primeiro será acionada a saída SAI01, e depois a saída SAI02.

4.2 Elementos de linguagem Ladder

4.2.1 Lógica básica

Uma instrução lógica ‘E’ é implementada do seguinte modo na linguagem Ladder.

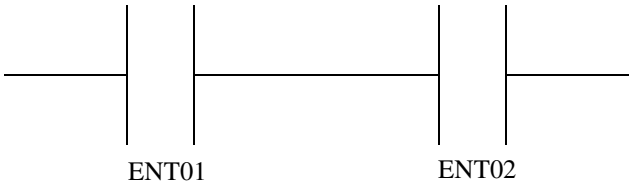

Figura 4.3: ‘E’ lógico em Ladder

Assim, se fecharmos os contatos ENT01 ‘E’ ENT02, haverá passagem de corrente pelo rung. Já a instrução lógica ‘OU’ é implementada como dois contatos em paralelo (Figura 4.4)..

Deste modo, se fecharmos o contato ENT01 ‘OU’ o contato ENT02, haverá passagem de corrente pelo rung.

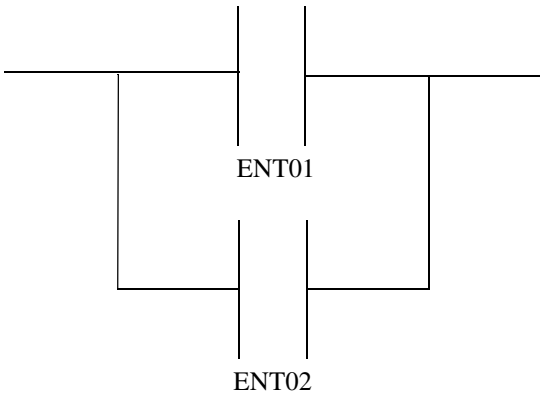

Figura 4.4: 'OU' lógico em Ladder

4.2.2 Temporizadores

Os temporizadores, como o nome sugere, são utilizados para temporização de condições no diagrama Ladder. O temporizador é identificado pela letra T seguida dos dígitos correspondentes ao seu endereço: T0, T1, etc. Há um bit de status relacionado ao temporizador, o qual é ativado quando o valor atual do temporizador for maior ou igual ao valor de preset (valor pré-configurado). O incremento de tempo depende do temporizador utilizado. O temporizador abaixo apresenta uma entrada de controle (enable) que, ao ser acionada (rung = 1), habilita o início da temporização, e ao ser desligada (rung = 0), reinicia o temporizador, mantendo-o nesta condição até novo acionamento da entrada enable (Georgini, 2006).

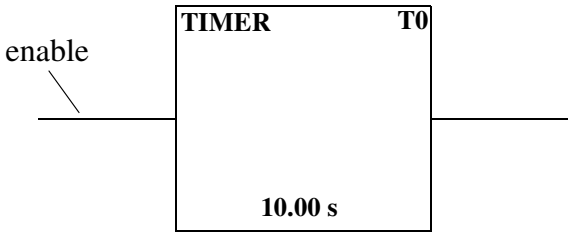

Figura 4.5: Temporizador

Na lógica de controle da Figura 4.6, quando os contatos BOT01 e BOT02 são fechados e permanecem fechados simultaneamente, é iniciada a temporização de T1, que tem um valor de preset fixo em 10 segundos (este temporizador apresenta incremento de tempo de 0.01 s). Ao ser atingido o valor de preset, o bit de status de T1 é acionado, ativando a saída SAI01. O bit de status permanece ativado até que o temporizador seja desativado (ou seja, quando um dos contatos BOT01 ou BOT02 for aberto). Esse bit pode ser associado a um contato normalmente aberto, conforme podemos observar na figura.

Também é possível ter acesso ao valor atual do temporizador durante a execução do diagrama Ladder utilizando-se os bits de valor atual. Por exemplo, TW0 armazena o valor atual de T0, TW1 armazena o valor atual de T1, e assim por diante, sendo que esses dados podem ser utilizados na lógica de controle.

A Figura 4.7 ilustra a utilização conjunta de temporizadores e comparadores. Assim, ao ser fechado o contato BOT01, é iniciada a temporização de T1. Quando o valor atual do temporizador T1 for maior ou igual a 1s, a saída SAI01 é acionada. Quando o valor atual de T1 for maior ou igual a 2s, a saída SAI02 é acionada, e quando esse valor atingir 3s, a saída SAI03 é acionada. O temporizador segue sua ação até atingir 10s, quando seu valor atual é zerado até que ele seja habilitado novamente através do acionamento de BOT01.

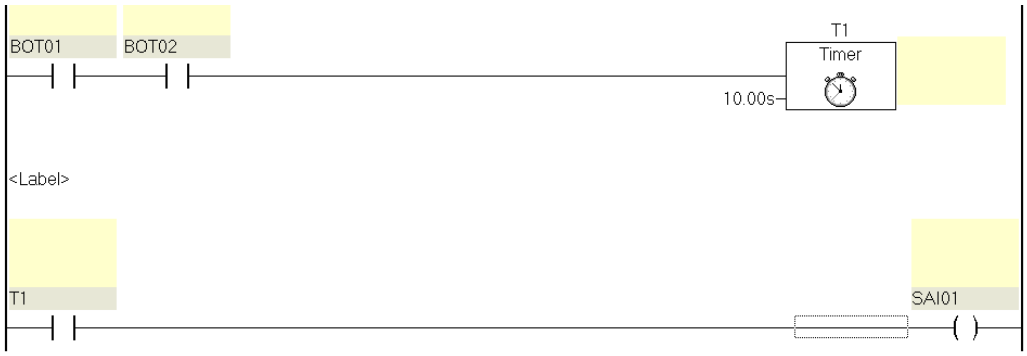

Figura 4.6: Lógica com temporizador

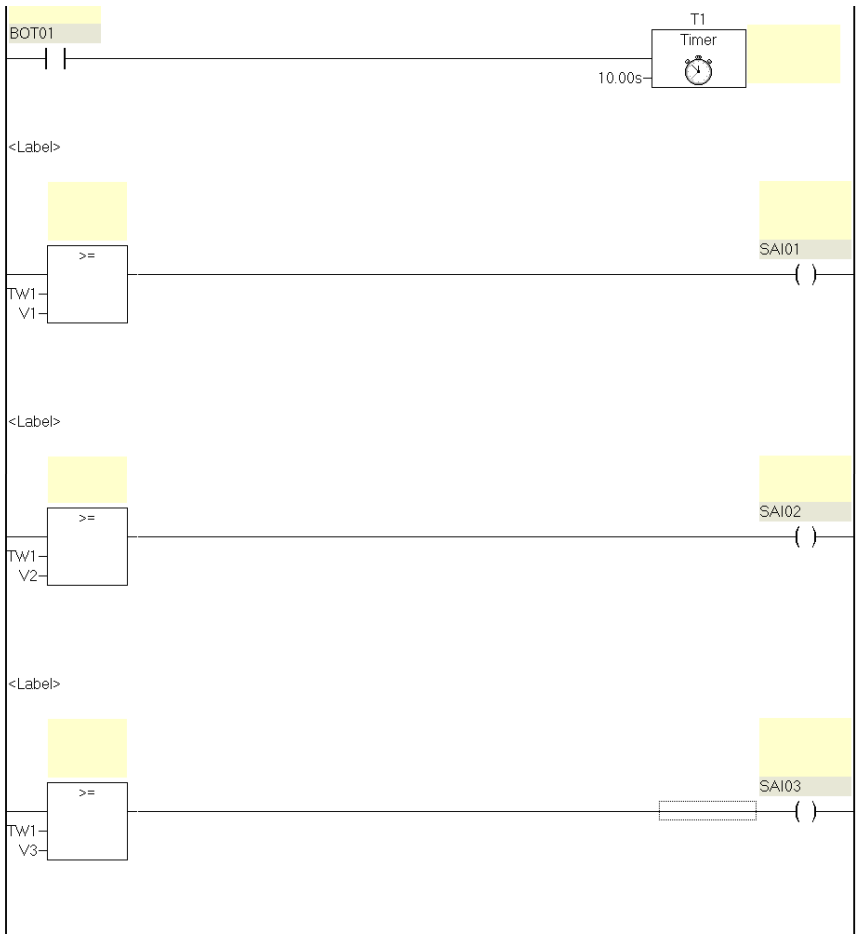

Figura 4.7: Lógica com temporizador e comparadores

4.2.3 Contadores

Os contadores são utilizados para contagem de condições ou eventos controlados pelo programa de aplicação. O contador é identificado pela letra C seguida do número do contador (C0, C1, C2, etc.). Há um bit de status relacionado ao contador, que é ativado quando o valor atual deste atingir o valor de preset (valor pré-configurado) (Georgini, 2006).

@ ERRRO!!!!! @Na lógica de controle da Figura 4.8, a cada transição de 0 para 1 (ou seja, off on) da entrada BOT01, o valor atual de C1 é incrementado em uma unidade, apresentando valor de preset de 20. Quando o valor de preset é atingido, o bit de status de C1 é ativado, acionando a saída SAI01. Esse bit de sta-

tus permanece ativado durante um ciclo do programa; no loop seguinte, o valor do bit de status de C1 é zerado automaticamente.

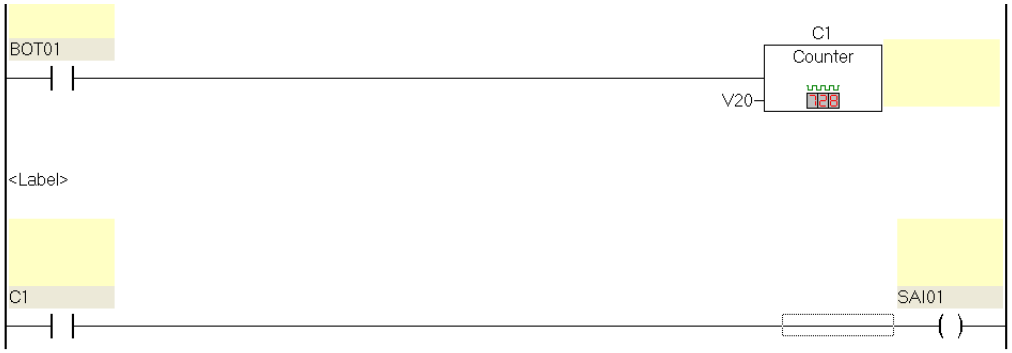

Figura 4.8: Lógica com contador

Além de ativar o bit de status quando atinge o valor de preset, o contador também oferece acesso ao seu valor atual durante a execução do programa de aplicação. Assim, a variável CW0 armazena o valor atual de C0, CW1 armazena o valor atual de C1, e assim por diante. Essas variáveis podem ser utilizadas na construção da lógica de controle.

A Figura 4.9 apresenta uma seqüência lógica envolvendo um contador e comparadores.

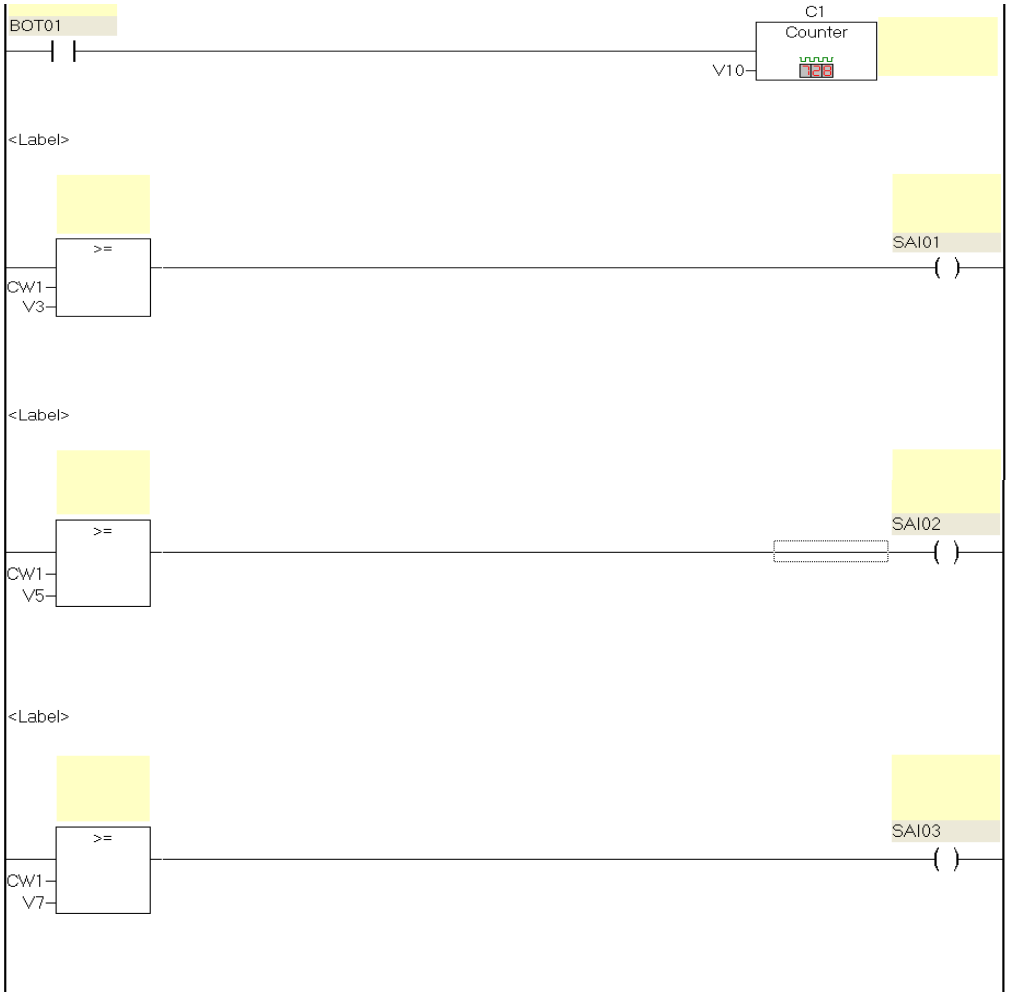

Figura 4.9: Lógica com contador e comparadores

Na lógica de controle da Figura 4.9, a cada transição de 0 para 1 do contato BOT01, o valor atual de C1 é incrementado em 1 unidade. Ao atingir a contagem de 3, a saída SAI01 é acionada; ao atingir a contagem de 5, a saída SAI02 é acionada, e ao atingir a contagem de 7, a saída SAI03 é acionada.

4.2.4 Funções SET e RESET

Ao ser executada (ou seja, quando o rung ligado à bobina set for igual a 1), a função set aciona o operando controlado, mantendo-o acionado mesmo que o rung não permaneça acionado (rung = 0). O operando acionado pela função set pode ser desligado pela função reset; esta função o manterá desligado mesmo que o rung não permaneça acionado. Se as instruções set e reset forem executadas durante o mesmo ciclo sobre o mesmo operando, terá controle final sobre o operando aquela que for executada por último (Georgini, 2006).

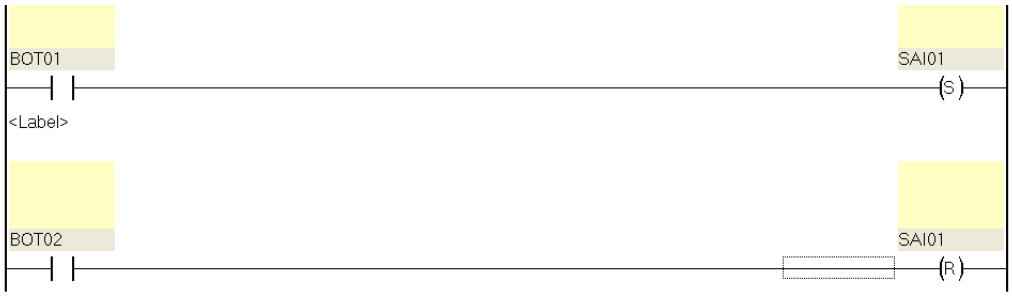

Figura 4.10: Lógica com SET e RESET

Na lógica de controle da Figura 4.10, ao ser pressionado o botão BOT01, a saída SAI01 será acionada, permanecendo nesta condição mesmo que BOT01 não continue sendo apertado. Ao ser pressionado BOT02, a saída SAI01 é desligada, permanecendo assim mesmo que BOT02 não continue sendo apertado. Se os botões BOT01 e BOT02 forem pressionados simultaneamente, a saída SAI01 é desligada, já que a instrução RESET será executada por último.

4.2.5 Como iniciar uma seção de programação

Primeiramente, deve-se iniciar o programa FST 4.10 da FESTO clicando-se no seu ícone, presente na área de trabalho do Windows. Uma vez iniciado o programa, clicar no item *'New'* do menu *'Project'*. Aparecerá uma tela pedindo um nome para o programa que será construído. Deve-se escolher um nome e clicar em *'ok'*. Aparecerá outra janela, *'Project Settings'*, perguntando o tipo de controlador a ser utilizado. Deve-se escolher a opção '*FEC Compact'* (veja a Figura 4.11):.

No item *'Project Settings'* , é possível adicionar comentários ao programa. Para começarmos a construir o diagrama ladder, devemos clicar no item *'New'* do menu *'Program'*. Aparecerá uma janela perguntando o tipo de programa a ser elaborado. São duas possibilidades: diagrama ladder e lista de instruções. Trataremos aqui apenas da primeira opção. Seleciona-se então a opção diagrama ladder. Outra janela de opções se abre, onde é possível definir a versão e o número do programa, bem como adicionar comentários a ele. Clicando-se em 'ok' , aparecerá um diagrama ladder em branco:

Ao se escolher o item *'Shortcuts'* do menu *'View'* aparece na tela uma barra de ferramentas. Esta barra tem as principais funções a serem utilizadas na construção de um diagrama ladder.

A opção *'Help'* do programa oferece a documentação completa, tanto do CLP como do software de programação.

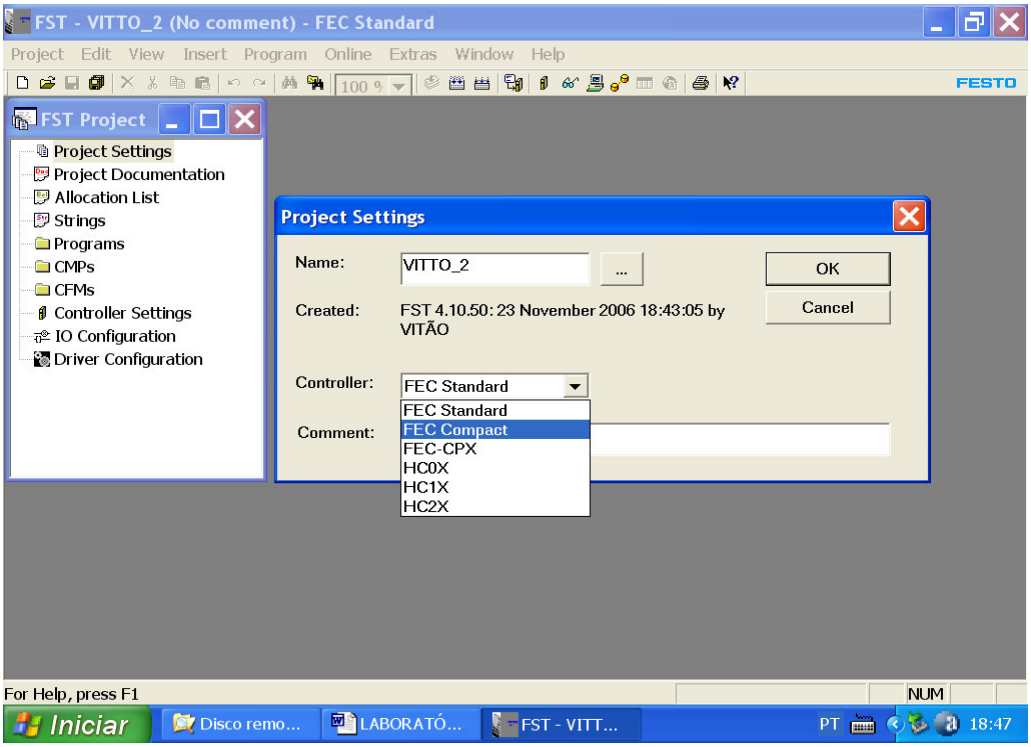

Figura 4.11: O software de programação do CLP FESTO

4.3 Exercícios

4.3.1 Exercícios simples

a) Deseja-se acender e apagar uma lâmpada através de um botão. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) Um dispositivo de uma indústria metalúrgica tem como função a fixação de peças em um molde. Esta fixação é feita por um atuador linear de dupla ação que avança mediante o acionamento de dois botões (S1 e S2) e retorna caso os botões sejam desacionados. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

c) Verificar através de botões e de uma lâmpada a tabela da verdade da função ou-exclusivo. Elabore um programa em linguagem Ladder e teste no CLP.

4.3.2 Tanque industrial

a) Elabore um diagrama Ladder simplificado para encher ou esvaziar um tanque industrial por meio de duas eletroválvulas. A eletroválvula V1 permite a entrada de líquido e a V2 permite o escoamento de saída. Quando o líquido atinge o nível máximo do tanque, um sensor A envia um sinal para o circuito lógico. Abaixo do nível máximo o sensor A não envia sinal algum. Há ainda um botão B, que deve encher o tanque quando for acionado e esvaziar em caso contrário. O esquema do tanque está apresentado na Figura 4.12 e as convenções do funcionamento do sistema estão apresentados na Tabela 4.13.

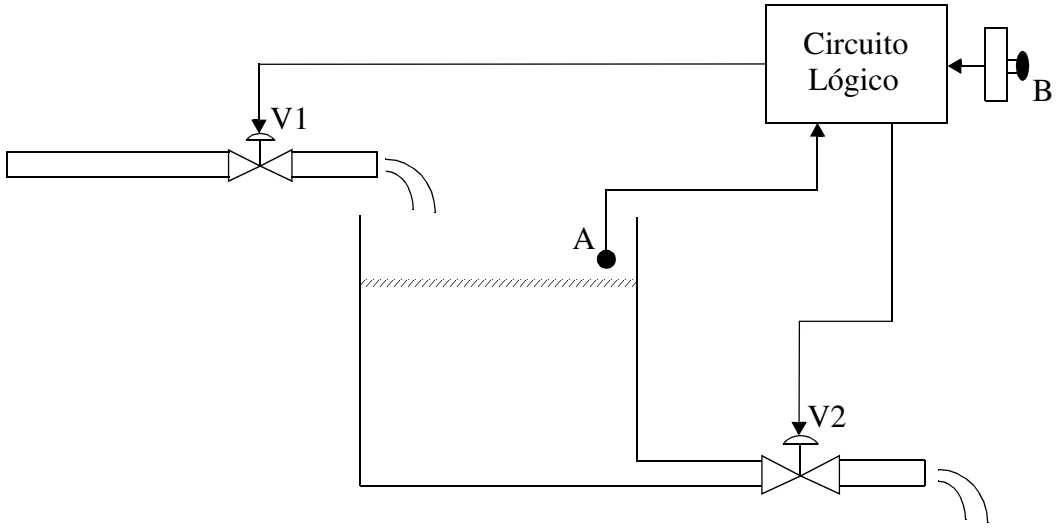

Figura 4.12: Esquema do tanque industrial

Tabela 4.13: Convenções para o tanque industrial

Sinal	Significado
A=1	Tanque cheio
A=0	Tanque não cheio
B=1	Comando encher
B=0	Comando esvaziar
V1=1	Comando fechar V1
V1=0	Comando abrir V1
V2=1	Comando fechar V2
V2=0	Comando abrir V2

4.3.3 Usando contadores

Há várias modalidades de contagem que podem ser utilizadas no CLP.

a) Deseja-se acender uma lâmpada após um botão ser acionado cinco vezes. Outro botão apaga a lâmpada (se ela estiver acesa) e reinicia a contagem. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

4.3.4 Exercícios com temporizadores

O CLP da FESTO também possui diferentes tipos de temporizadores. Consulte a documentação do programa para escolher o mais adequado a cada tipo de problema.

a) Deseja-se acender uma lâmpada de alarme durante 10s, quando um botão de emergência é acionado. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) O alarme A de uma casa é ativado por um sensor de movimento M e por um sensor de abertura de janelas J. O sensor M ativa o alarme quando detecta a presença de pessoas. O sensor J ativa o

alarme quando a janela é aberta. Há ainda um botão B para ligar ou desligar o alarme. Supondo que o alarme deve ser acionado por 10s e depois desligar automaticamente, elabore um diagrama ladder simplificado para resolver este problema. As convenções estão indicadas na Tabela4.14.

Tabela 4.14: Convenções para o sistema de alarme

Sinal	Significado
B=0	Comando desligar alarme
B=1	Comando ligar alarme
M=0	Ausência de pessoas
M=1	Presença de pessoas
J=0	Janela aberta
J=1	Janela fechada

c) Um sistema de dois semáforos controla o tráfego de um cruzamento de duas ruas (rua A e rua B), conforme a Figura 4.15,, sendo que cada semáforo está posicionado numa das ruas. A seqüência de

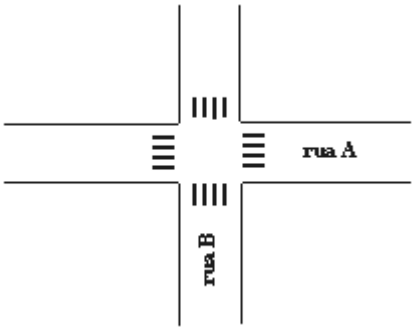

Figura 4.15: Cruzamento de ruas

acionamento de cada fase (amarelo, vermelho e verde) dos semáforos é mostrada na Tabela 4.16.

Tabela 4.16: Temporização

Fase	Tempo (s)	Semáforo A	Semáforo B
1	10	Verde	Vermelho
2	3	Amarelo	Vermelho
3	2	Vermelho	Vermelho
4	10	Vermelho	Verde
5	3	Vermelho	Amarelo
6	2	Vermelho	Vermelho

Implemente o semáforo em um programa em linguagem Ladder e teste no CLP.

d) Seja um sistema de semáforos similar ao anterior, controlando o tráfego no cruzamento de duas vias, conforme a Figura 4.17. Há um semáforo na rua A e um na rua B (as setas indicam o sentido do tráfego em cada rua). A diferença é que agora há quatro semáforos adicionais (S1, S2, S3 e S4), sendo dois deles dotados de botão para travessia de pedestres (S3 e S4). Cada um está posicionado

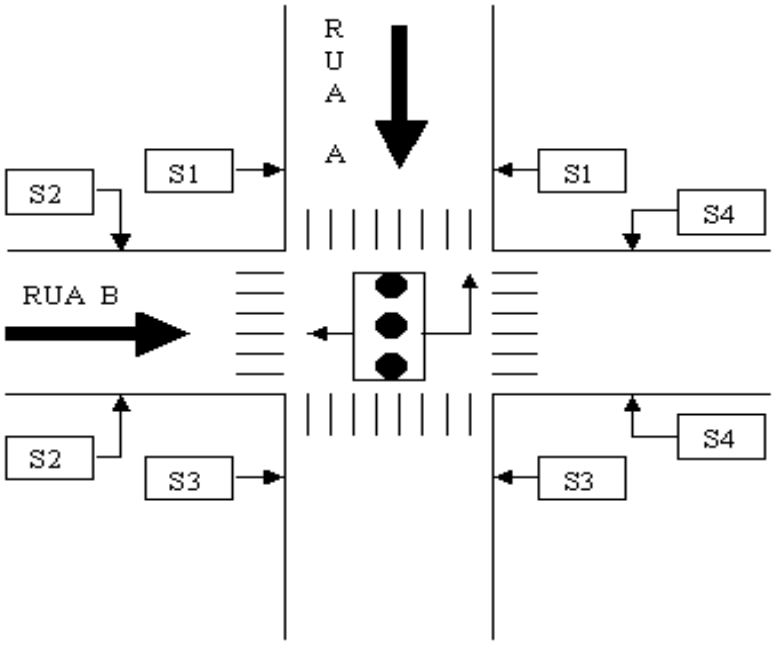

Figura 4.17: Cruzamento de ruas com travessia de pedestres

de um dos lados do cruzamento entre as ruas A e B. Ao acionamento do botão de S3 ou S4, os semáforos das ruas A e B devem passar de verde para amarelo, e deste para vermelho, a menos que já esteja(m) fechado(s) ou em processo de fechamento. Os estados correspondentes às fases dos semáforos são descritos na Tabela 4.18.

Tabela 4.18: Temporização

Fase	Tempo (s)	Semaf. A	S1	S3	Semaf. B	S2	S4
1	15	Verde	Vermelho	Vermelho	Vermelho	Verde	Vermelho
2	3	Amarelo	Vermelho	Vermelho	Vermelho	Verde	Vermelho
3	2	Vermelho	Vermelho	Vermelho	Vermelho	Vermelho*	Vermelho
4	15	Vermelho	Verde	Vermelho	Verde	Vermelho	Vermelho
5	3	Vermelho	Verde	Vermelho	Amarelo	Vermelho	Vermelho
6	2	Vermelho	Vermelho*	Vermelho	Vermelho	Vermelho	Vermelho
7	10	Vermelho	Verde	Verde	Vermelho	Verde	Verde
8	5	Vermelho	Vermelho*	Vermelho*	Vermelho	Vermelho*	Vermelho*

Nesta tabela, *Vermelho** indica vermelho piscante. As fases 7 e 8 só poderão ocorrer se durante as fases 1 a 6 houver o acionamento de pelo menos um botão para travessia de pedestres. Se isso ocorrer, o semáforo (A ou B) que estiver verde deve passar para amarelo (e ficar nessa fase durante 3 segundos), a menos que já esteja no amarelo. Após isso, passa-se para a fase 7 e, posteriormente, para a fase 8. Terminada a fase 8, volta-se à fase 1. Se nenhum botão for pressionado, a seqüência de fases é 1-2-3-4-5-6-1-2-etc. Implemente este semáforo em um programa em linguagem Ladder e teste no CLP.

4.3.5 Usando contadores e temporizadores

a) Deseja-se engarrafar bebidas de modo automático utilizando-se um CLP. As garrafas movimentam-se através de uma esteira rolante acionada por um motor elétrico, o qual é ligado e desligado pelo CLP, conforme a Figura 4.19. Quando cinco garrafas passarem por um sensor de presença (Sensor A), o motor deve ser desligado e um conjunto de cinco bicos injetores de cerveja deve ser acionado por 10 segundos (para encher as garrafas); após esses 10 segundos, o motor da esteira deve voltar a movimentá-la, até que outras cinco garrafas vazias passem pelo Sensor A; quando isso ocorrer, o processo se repetirá.

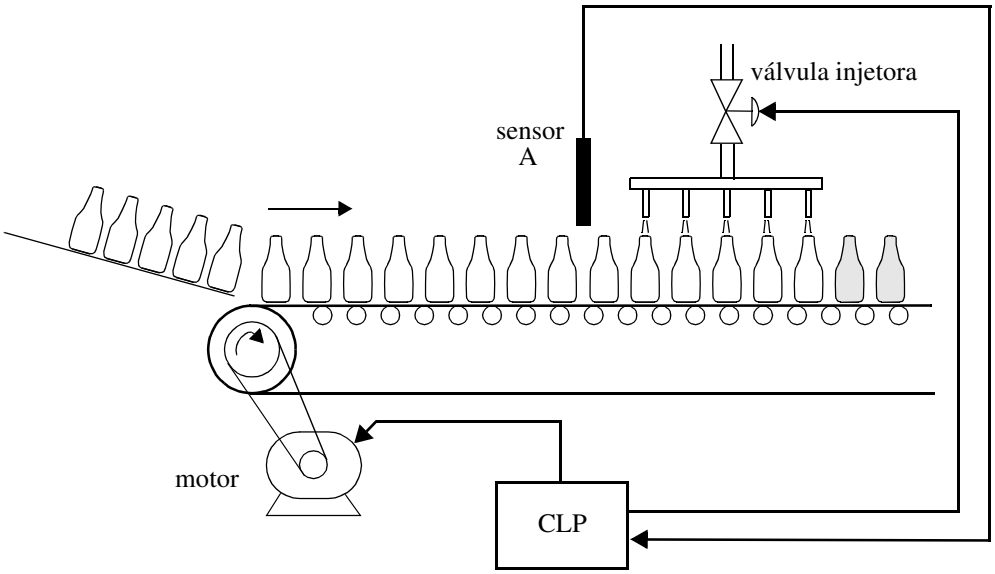

Figura 4.19: Engarrafadora

RPM/2009

LAB4
Introdução aos Controladores Lógicos Programáveis

4.1 Introdução

Os Controladores Lógicos Programáveis (CLPs) são dispositivos digitais, muito utilizados na indústria, capazes de armazenar instruções para implementação de funções de controle (tais quais seqüência lógica, temporização e contagem), bem como realizar operações lógicas e aritméticas, manipulação de dados e comunicação em rede, sendo utilizados no controle de sistemas automatizados (Georgini, 2006). Seus principais componentes são a unidade central de processamento (CPU), os módulos de I/O (ou módulos de entrada/ saída), a fonte de alimentação e a base.

A CPU do CLP compreende o microprocessador, o sistema de memória (ROM e RAM) e os circuitos auxiliares de controle. Os módulos de I/O são dispositivos através dos quais podemos conectar sensores, atuadores ou outros equipamentos à CPU do CLP; assim, a CPU pode ler sinais de entrada, ou enviar sinais para a saída do CLP através dos módulos de I/O. Esses módulos podem ser discretos ou analógicos. A fonte de alimentação é responsável pela tensão de alimentação fornecida à CPU e aos módulos de I/O. A base do CLP proporciona conexão mecânica e elétrica entre a CPU, os módulos de I/O e a fonte. Ela contém o barramento de comunicação entre eles, em que estão presentes os sinais de dados, endereço, controle e tensão de alimentação (Georgini, 2006).

A programação de um CLP pode ser feita através de uma variedade de linguagens. Uma das mais populares é a linguagem Ladder, tendo recebido este nome devido à sua semelhança com uma escada (ladder), na qual duas barras verticais paralelas são interligadas pela lógica de controle, formando os degraus (rungs) da escada (Georgini, 2006). Na Figura 4.1 temos uma representação de lógica de controle através da linguagem ladder:

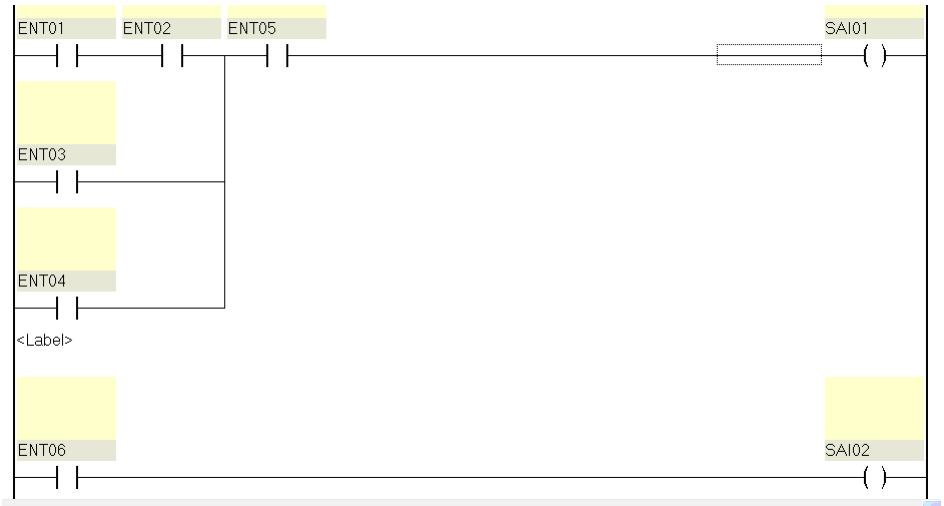

Figura 4.1: Exemplo de diagrama Ladder

O diagrama da Figura 4.1 apresenta uma lógica de controle com dois rungs: o primeiro é formado por 3 linhas (primeira linha – ENT01, ENT02, ENT05 e SAI01; segunda linha – ENT03; terceira linha – ENT04) e o segundo é formado por uma linha (ENT06 e SAI02). Este diagrama é formado por contatos e bobinas. As

entradas são os contatos, e as saídas são as bobinas. Cada elemento da lógica de controle representa uma instrução da linguagem Ladder, sendo alocado em um endereço específico e consumindo uma determinada quantidade de memória disponível para armazenamento do programa de aplicação (Georgini, 2006). Os contatos utilizados nas entradas do diagrama da Figura 4.1 são chamados contatos normalmente abertos (NA). Nesses contatos, há passagem de corrente se o contato for fechado. Assim, se o contato for uma botoeira, o contato será fechado se o botão for pressionado, permitindo a passagem de corrente; enquanto o botão não estiver apertado, o contato estará aberto, e não haverá passagem de corrente. Já nos contatos normalmente fechados (NF) ocorre o contrário: há passagem de corrente se o contato for aberto; enquanto o contato estiver fechado, não há passagem de corrente.

contato normalmente aberto (NA) contato normalmente fechado (NF)

Figura 4.2: Tipos de contatos

O fluxo de corrente elétrica fictícia num diagrama Ladder sempre ocorre da extremidade esquerda para a extremidade direita. Assim, no diagrama da Figura 4.1, se os contatos das entradas 1, 2 e 5 forem fechados simultaneamente, haverá passagem de corrente fictícia pela linha 1 do rung 1. Então, a bobina SAI01 será energizada, acionando a saída 1. Se forem fechados apenas os contatos ENT01 e ENT05, mantendo ENT02 aberto, não haverá a passagem de corrente fictícia pela linha 1 do rung 1, e então a saída 1 não será acionada. Outros modos de acionar a saída 1 seriam fechar os contatos ENT03 e ENT05 simultaneamente, ou ainda fechar os contatos ENT04 e ENT05 simultaneamente. A seqüência de leitura do programa em linguagem Ladder é do rung superior para o rung inferior; dessa forma, no exemplo da figura, se fecharmos os contatos ENT03, ENT05 e ENT06, primeiro será acionada a saída SAI01, e depois a saída SAI02.

4.2 Elementos de linguagem Ladder

4.2.1 Lógica básica

Uma instrução lógica ‘E’ é implementada do seguinte modo na linguagem Ladder.

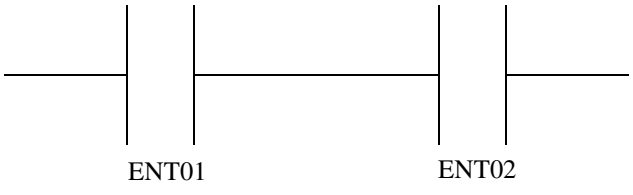

Figura 4.3: ‘E’ lógico em Ladder

Assim, se fecharmos os contatos ENT01 ‘E’ ENT02, haverá passagem de corrente pelo rung. Já a instrução lógica ‘OU’ é implementada como dois contatos em paralelo (Figura 4.4)..

Deste modo, se fecharmos o contato ENT01 ‘OU’ o contato ENT02, haverá passagem de corrente pelo rung.

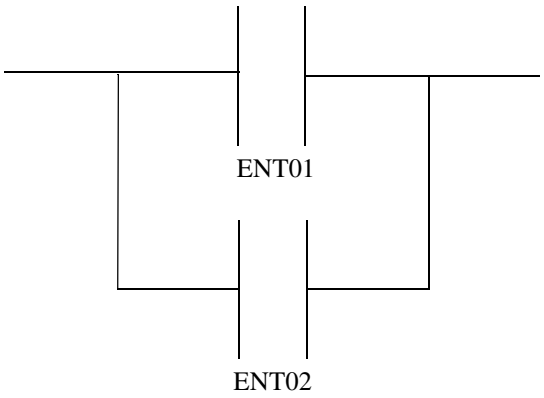

Figura 4.4: 'OU' lógico em Ladder

4.2.2 Temporizadores

Os temporizadores, como o nome sugere, são utilizados para temporização de condições no diagrama Ladder. O temporizador é identificado pela letra T seguida dos dígitos correspondentes ao seu endereço: T0, T1, etc. Há um bit de status relacionado ao temporizador, o qual é ativado quando o valor atual do temporizador for maior ou igual ao valor de preset (valor pré-configurado). O incremento de tempo depende do temporizador utilizado. O temporizador abaixo apresenta uma entrada de controle (enable) que, ao ser acionada (rung = 1), habilita o início da temporização, e ao ser desligada (rung = 0), reinicia o temporizador, mantendo-o nesta condição até novo acionamento da entrada enable (Georgini, 2006).

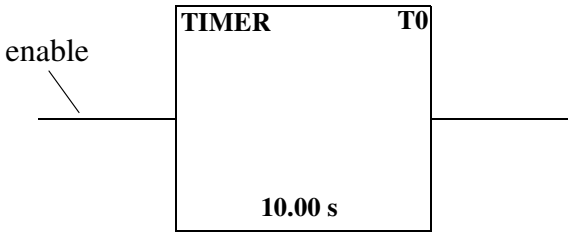

Figura 4.5: Temporizador

Na lógica de controle da Figura 4.6, quando os contatos BOT01 e BOT02 são fechados e permanecem fechados simultaneamente, é iniciada a temporização de T1, que tem um valor de preset fixo em 10 segundos (este temporizador apresenta incremento de tempo de 0.01 s). Ao ser atingido o valor de preset, o bit de status de T1 é acionado, ativando a saída SAI01. O bit de status permanece ativado até que o temporizador seja desativado (ou seja, quando um dos contatos BOT01 ou BOT02 for aberto). Esse bit pode ser associado a um contato normalmente aberto, conforme podemos observar na figura.

Também é possível ter acesso ao valor atual do temporizador durante a execução do diagrama Ladder utilizando-se os bits de valor atual. Por exemplo, TW0 armazena o valor atual de T0, TW1 armazena o valor atual de T1, e assim por diante, sendo que esses dados podem ser utilizados na lógica de controle.

A Figura 4.7 ilustra a utilização conjunta de temporizadores e comparadores. Assim, ao ser fechado o contato BOT01, é iniciada a temporização de T1. Quando o valor atual do temporizador T1 for maior ou igual a 1s, a saída SAI01 é acionada. Quando o valor atual de T1 for maior ou igual a 2s, a saída SAI02 é acionada, e quando esse valor atingir 3s, a saída SAI03 é acionada. O temporizador segue sua ação até atingir 10s, quando seu valor atual é zerado até que ele seja habilitado novamente através do acionamento de BOT01.

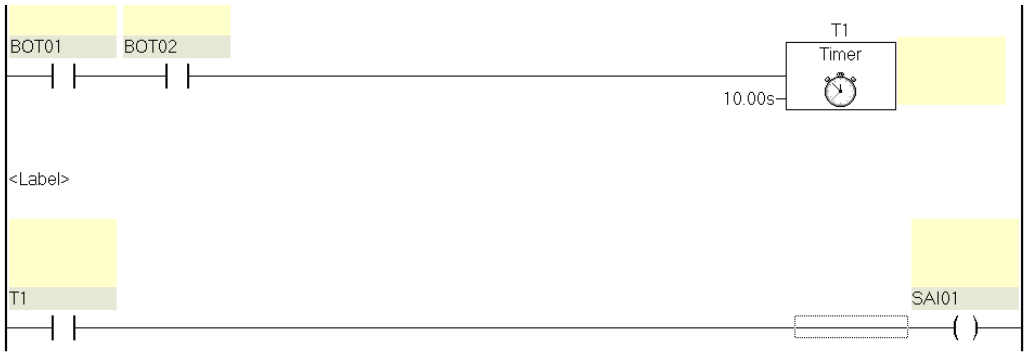

Figura 4.6: Lógica com temporizador

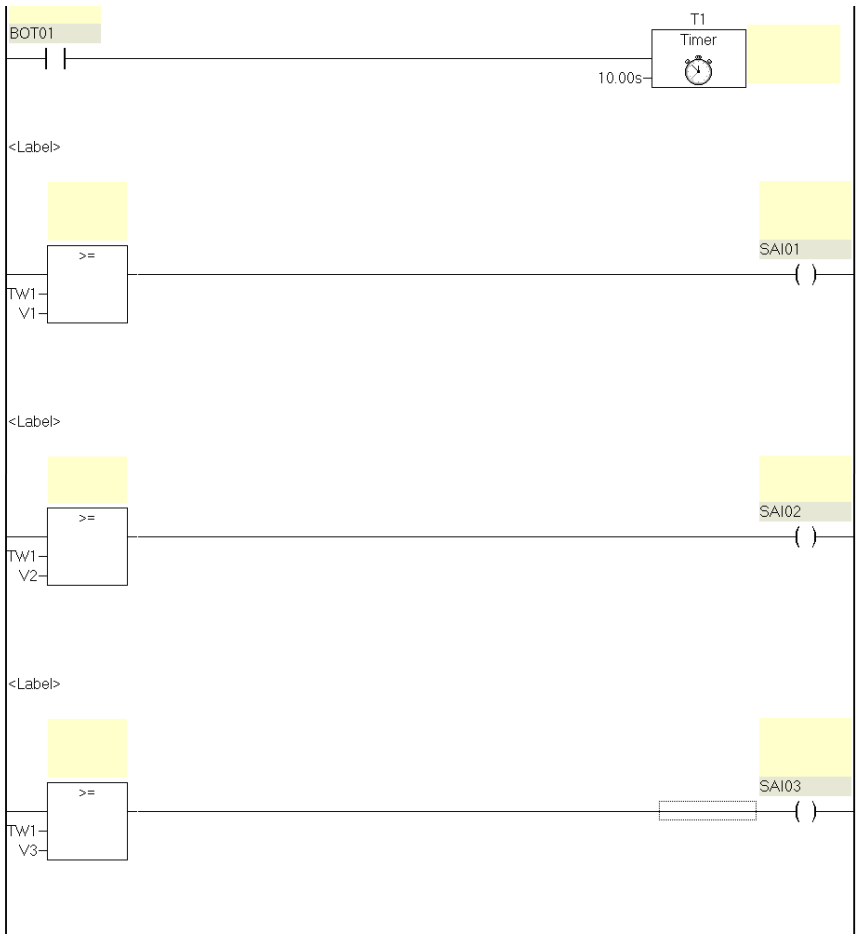

Figura 4.7: Lógica com temporizador e comparadores

4.2.3 Contadores

Os contadores são utilizados para contagem de condições ou eventos controlados pelo programa de aplicação. O contador é identificado pela letra C seguida do número do contador (C0, C1, C2, etc.). Há um bit de status relacionado ao contador, que é ativado quando o valor atual deste atingir o valor de preset (valor pré-configurado) (Georgini, 2006).

@ ERRRO!!!!! @Na lógica de controle da Figura 4.8, a cada transição de 0 para 1 (ou seja, off on) da entrada BOT01, o valor atual de C1 é incrementado em uma unidade, apresentando valor de preset de 20. Quando o valor de preset é atingido, o bit de status de C1 é ativado, acionando a saída SAI01. Esse bit de sta-

tus permanece ativado durante um ciclo do programa; no loop seguinte, o valor do bit de status de C1 é zerado automaticamente.

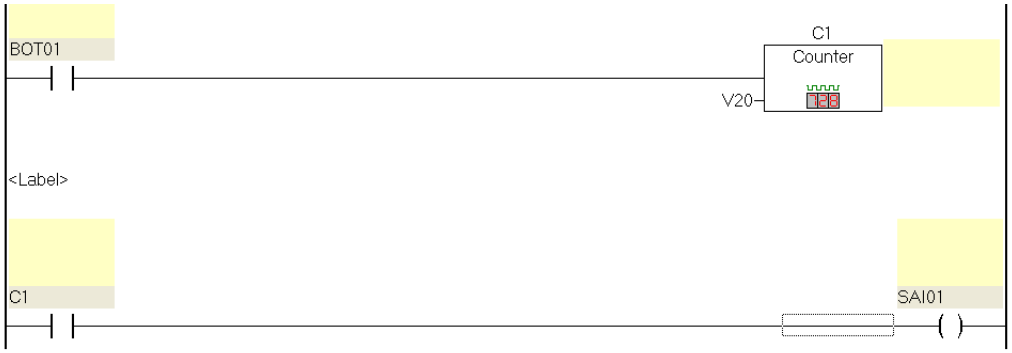

Figura 4.8: Lógica com contador

Além de ativar o bit de status quando atinge o valor de preset, o contador também oferece acesso ao seu valor atual durante a execução do programa de aplicação. Assim, a variável CW0 armazena o valor atual de C0, CW1 armazena o valor atual de C1, e assim por diante. Essas variáveis podem ser utilizadas na construção da lógica de controle.

A Figura 4.9 apresenta uma seqüência lógica envolvendo um contador e comparadores.

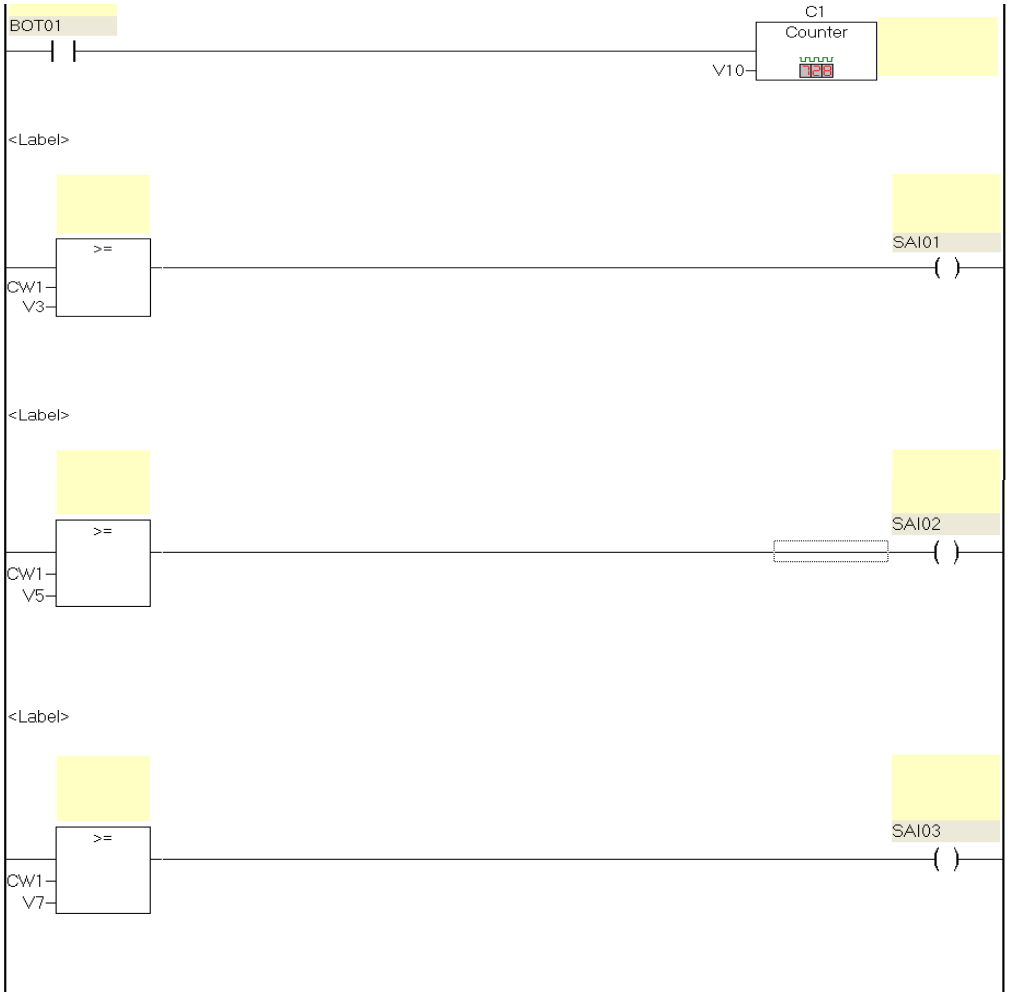

Figura 4.9: Lógica com contador e comparadores

Na lógica de controle da Figura 4.9, a cada transição de 0 para 1 do contato BOT01, o valor atual de C1 é incrementado em 1 unidade. Ao atingir a contagem de 3, a saída SAI01 é acionada; ao atingir a contagem de 5, a saída SAI02 é acionada, e ao atingir a contagem de 7, a saída SAI03 é acionada.

4.2.4 Funções SET e RESET

Ao ser executada (ou seja, quando o rung ligado à bobina set for igual a 1), a função set aciona o operando controlado, mantendo-o acionado mesmo que o rung não permaneça acionado (rung = 0). O operando acionado pela função set pode ser desligado pela função reset; esta função o manterá desligado mesmo que o rung não permaneça acionado. Se as instruções set e reset forem executadas durante o mesmo ciclo sobre o mesmo operando, terá controle final sobre o operando aquela que for executada por último (Georgini, 2006).

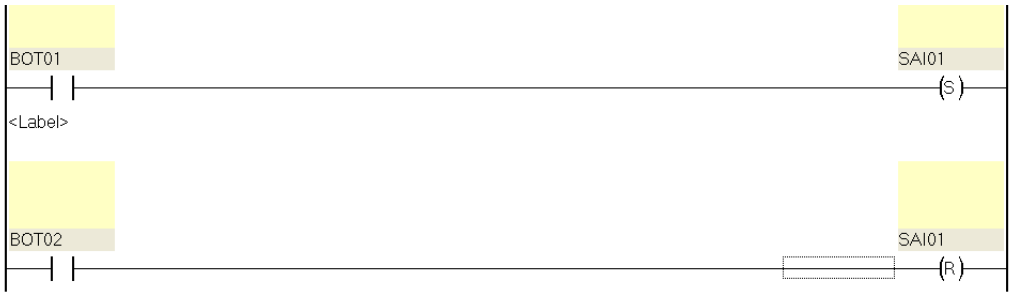

Figura 4.10: Lógica com SET e RESET

Na lógica de controle da Figura 4.10, ao ser pressionado o botão BOT01, a saída SAI01 será acionada, permanecendo nesta condição mesmo que BOT01 não continue sendo apertado. Ao ser pressionado BOT02, a saída SAI01 é desligada, permanecendo assim mesmo que BOT02 não continue sendo apertado. Se os botões BOT01 e BOT02 forem pressionados simultaneamente, a saída SAI01 é desligada, já que a instrução RESET será executada por último.

4.2.5 Como iniciar uma seção de programação

Primeiramente, deve-se iniciar o programa FST 4.10 da FESTO clicando-se no seu ícone, presente na área de trabalho do Windows. Uma vez iniciado o programa, clicar no item *'New'* do menu *'Project'*. Aparecerá uma tela pedindo um nome para o programa que será construído. Deve-se escolher um nome e clicar em *'ok'*. Aparecerá outra janela, *'Project Settings'*, perguntando o tipo de controlador a ser utilizado. Deve-se escolher a opção '*FEC Compact'* (veja a Figura 4.11):.

No item *'Project Settings'* , é possível adicionar comentários ao programa. Para começarmos a construir o diagrama ladder, devemos clicar no item *'New'* do menu *'Program'*. Aparecerá uma janela perguntando o tipo de programa a ser elaborado. São duas possibilidades: diagrama ladder e lista de instruções. Trataremos aqui apenas da primeira opção. Seleciona-se então a opção diagrama ladder. Outra janela de opções se abre, onde é possível definir a versão e o número do programa, bem como adicionar comentários a ele. Clicando-se em 'ok' , aparecerá um diagrama ladder em branco:

Ao se escolher o item *'Shortcuts'* do menu *'View'* aparece na tela uma barra de ferramentas. Esta barra tem as principais funções a serem utilizadas na construção de um diagrama ladder.

A opção *'Help'* do programa oferece a documentação completa, tanto do CLP como do software de programação.

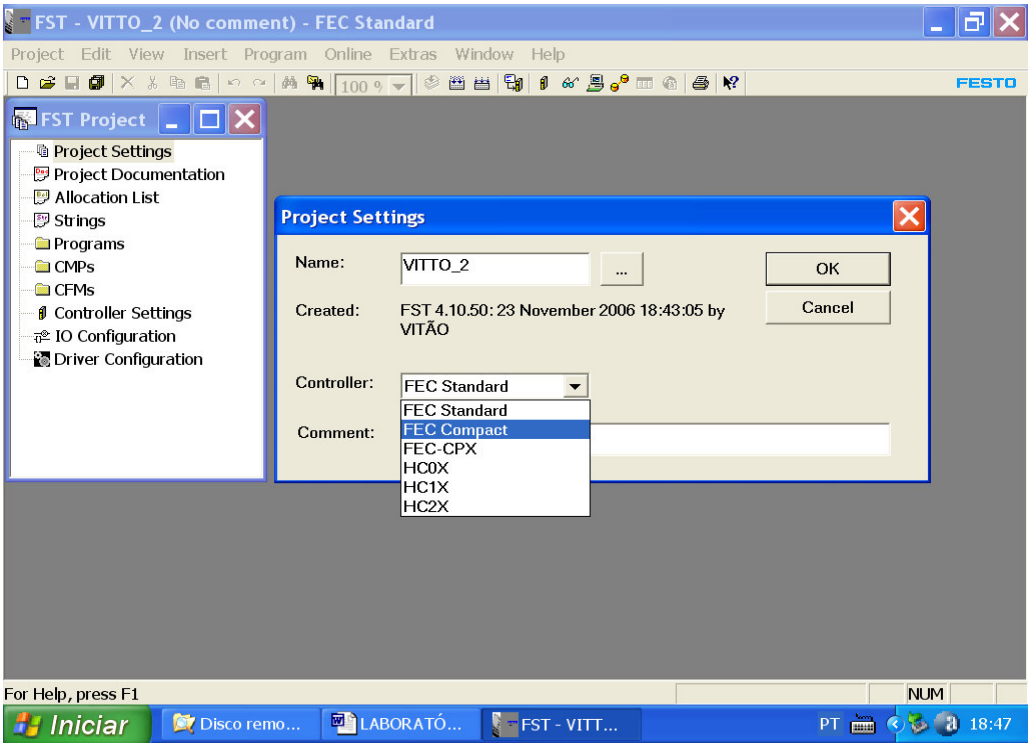

Figura 4.11: O software de programação do CLP FESTO

4.3 Exercícios

4.3.1 Exercícios simples

a) Deseja-se acender e apagar uma lâmpada através de um botão. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) Um dispositivo de uma indústria metalúrgica tem como função a fixação de peças em um molde. Esta fixação é feita por um atuador linear de dupla ação que avança mediante o acionamento de dois botões (S1 e S2) e retorna caso os botões sejam desacionados. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

c) Verificar através de botões e de uma lâmpada a tabela da verdade da função ou-exclusivo. Elabore um programa em linguagem Ladder e teste no CLP.

4.3.2 Tanque industrial

a) Elabore um diagrama Ladder simplificado para encher ou esvaziar um tanque industrial por meio de duas eletroválvulas. A eletroválvula V1 permite a entrada de líquido e a V2 permite o escoamento de saída. Quando o líquido atinge o nível máximo do tanque, um sensor A envia um sinal para o circuito lógico. Abaixo do nível máximo o sensor A não envia sinal algum. Há ainda um botão B, que deve encher o tanque quando for acionado e esvaziar em caso contrário. O esquema do tanque está apresentado na Figura 4.12 e as convenções do funcionamento do sistema estão apresentados na Tabela 4.13.

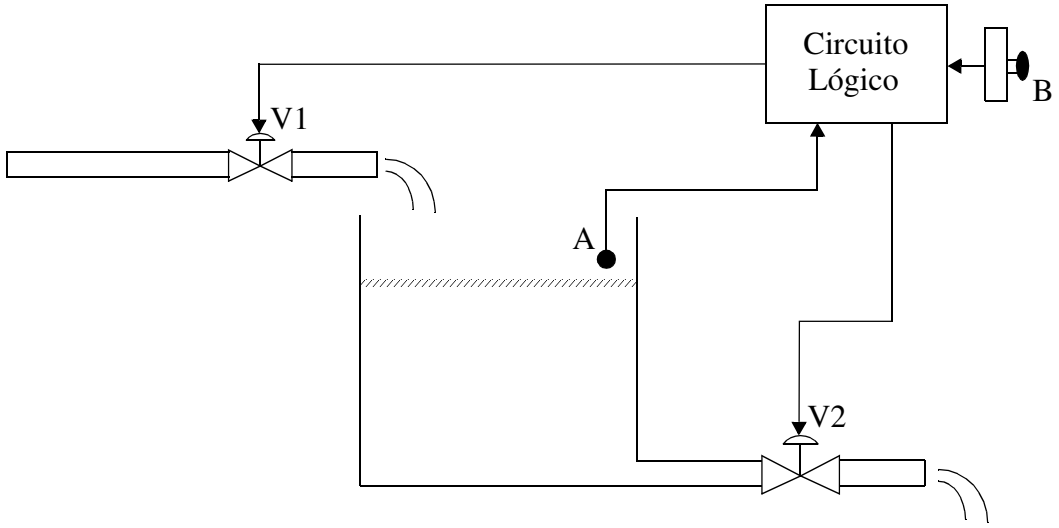

Figura 4.12: Esquema do tanque industrial

Tabela 4.13: Convenções para o tanque industrial

Sinal	Significado
A=1	Tanque cheio
A=0	Tanque não cheio
B=1	Comando encher
B=0	Comando esvaziar
V1=1	Comando fechar V1
V1=0	Comando abrir V1
V2=1	Comando fechar V2
V2=0	Comando abrir V2

4.3.3 Usando contadores

Há várias modalidades de contagem que podem ser utilizadas no CLP.

a) Deseja-se acender uma lâmpada após um botão ser acionado cinco vezes. Outro botão apaga a lâmpada (se ela estiver acesa) e reinicia a contagem. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

4.3.4 Exercícios com temporizadores

O CLP da FESTO também possui diferentes tipos de temporizadores. Consulte a documentação do programa para escolher o mais adequado a cada tipo de problema.

a) Deseja-se acender uma lâmpada de alarme durante 10s, quando um botão de emergência é acionado. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) O alarme A de uma casa é ativado por um sensor de movimento M e por um sensor de abertura de janelas J. O sensor M ativa o alarme quando detecta a presença de pessoas. O sensor J ativa o

alarme quando a janela é aberta. Há ainda um botão B para ligar ou desligar o alarme. Supondo que o alarme deve ser acionado por 10s e depois desligar automaticamente, elabore um diagrama ladder simplificado para resolver este problema. As convenções estão indicadas na Tabela4.14.

Tabela 4.14: Convenções para o sistema de alarme

Sinal	Significado
B=0	Comando desligar alarme
B=1	Comando ligar alarme
M=0	Ausência de pessoas
M=1	Presença de pessoas
J=0	Janela aberta
J=1	Janela fechada

c) Um sistema de dois semáforos controla o tráfego de um cruzamento de duas ruas (rua A e rua B), conforme a Figura 4.15,, sendo que cada semáforo está posicionado numa das ruas. A seqüência de

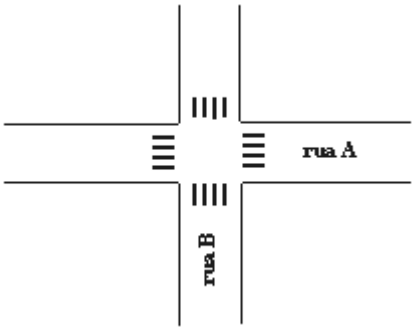

Figura 4.15: Cruzamento de ruas

acionamento de cada fase (amarelo, vermelho e verde) dos semáforos é mostrada na Tabela 4.16.

Tabela 4.16: Temporização

Fase	Tempo (s)	Semáforo A	Semáforo B
1	10	Verde	Vermelho
2	3	Amarelo	Vermelho
3	2	Vermelho	Vermelho
4	10	Vermelho	Verde
5	3	Vermelho	Amarelo
6	2	Vermelho	Vermelho

Implemente o semáforo em um programa em linguagem Ladder e teste no CLP.

d) Seja um sistema de semáforos similar ao anterior, controlando o tráfego no cruzamento de duas vias, conforme a Figura 4.17. Há um semáforo na rua A e um na rua B (as setas indicam o sentido do tráfego em cada rua). A diferença é que agora há quatro semáforos adicionais (S1, S2, S3 e S4), sendo dois deles dotados de botão para travessia de pedestres (S3 e S4). Cada um está posicionado

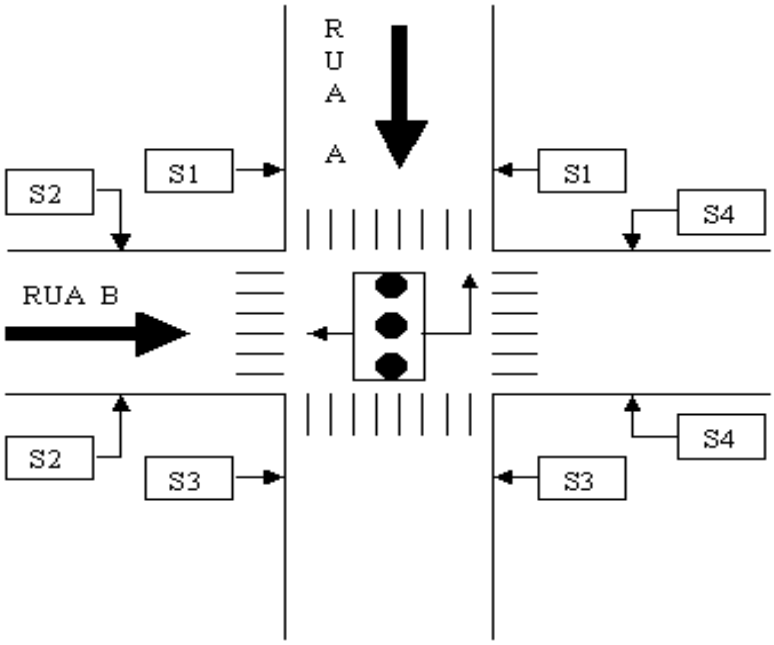

Figura 4.17: Cruzamento de ruas com travessia de pedestres

de um dos lados do cruzamento entre as ruas A e B. Ao acionamento do botão de S3 ou S4, os semáforos das ruas A e B devem passar de verde para amarelo, e deste para vermelho, a menos que já esteja(m) fechado(s) ou em processo de fechamento. Os estados correspondentes às fases dos semáforos são descritos na Tabela 4.18.

Tabela 4.18: Temporização

Fase	Tempo (s)	Semaf. A	S1	S3	Semaf. B	S2	S4
1	15	Verde	Vermelho	Vermelho	Vermelho	Verde	Vermelho
2	3	Amarelo	Vermelho	Vermelho	Vermelho	Verde	Vermelho
3	2	Vermelho	Vermelho	Vermelho	Vermelho	Vermelho*	Vermelho
4	15	Vermelho	Verde	Vermelho	Verde	Vermelho	Vermelho
5	3	Vermelho	Verde	Vermelho	Amarelo	Vermelho	Vermelho
6	2	Vermelho	Vermelho*	Vermelho	Vermelho	Vermelho	Vermelho
7	10	Vermelho	Verde	Verde	Vermelho	Verde	Verde
8	5	Vermelho	Vermelho*	Vermelho*	Vermelho	Vermelho*	Vermelho*

Nesta tabela, *Vermelho** indica vermelho piscante. As fases 7 e 8 só poderão ocorrer se durante as fases 1 a 6 houver o acionamento de pelo menos um botão para travessia de pedestres. Se isso ocorrer, o semáforo (A ou B) que estiver verde deve passar para amarelo (e ficar nessa fase durante 3 segundos), a menos que já esteja no amarelo. Após isso, passa-se para a fase 7 e, posteriormente, para a fase 8. Terminada a fase 8, volta-se à fase 1. Se nenhum botão for pressionado, a seqüência de fases é 1-2-3-4-5-6-1-2-etc. Implemente este semáforo em um programa em linguagem Ladder e teste no CLP.

4.3.5 Usando contadores e temporizadores

a) Deseja-se engarrafar bebidas de modo automático utilizando-se um CLP. As garrafas movimentam-se através de uma esteira rolante acionada por um motor elétrico, o qual é ligado e desligado pelo CLP, conforme a Figura 4.19. Quando cinco garrafas passarem por um sensor de presença (Sensor A), o motor deve ser desligado e um conjunto de cinco bicos injetores de cerveja deve ser acionado por 10 segundos (para encher as garrafas); após esses 10 segundos, o motor da esteira deve voltar a movimentá-la, até que outras cinco garrafas vazias passem pelo Sensor A; quando isso ocorrer, o processo se repetirá.

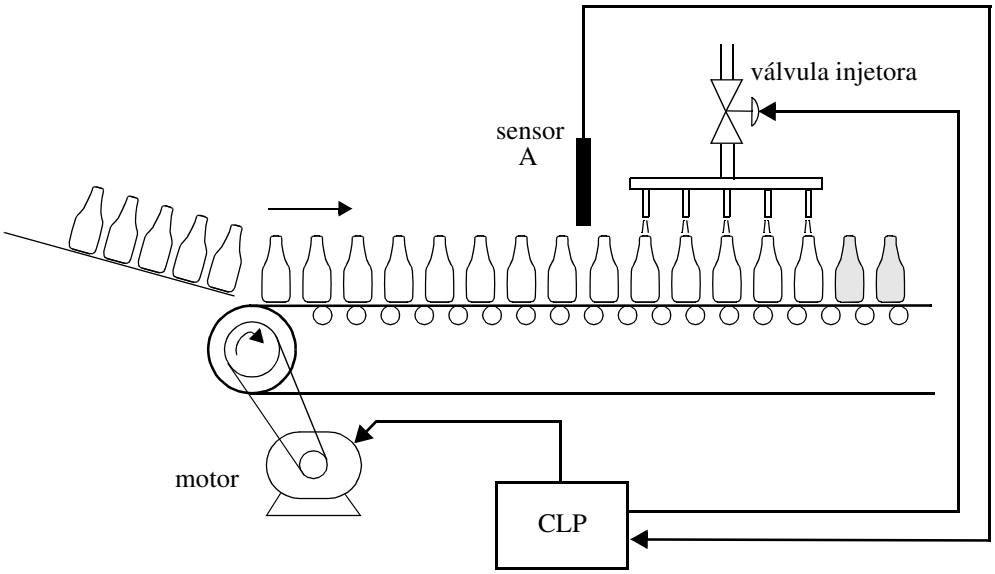

Figura 4.19: Engarrafadora

RPM/2009

LAB4
Introdução aos Controladores Lógicos Programáveis

4.1 Introdução

Os Controladores Lógicos Programáveis (CLPs) são dispositivos digitais, muito utilizados na indústria, capazes de armazenar instruções para implementação de funções de controle (tais quais seqüência lógica, temporização e contagem), bem como realizar operações lógicas e aritméticas, manipulação de dados e comunicação em rede, sendo utilizados no controle de sistemas automatizados (Georgini, 2006). Seus principais componentes são a unidade central de processamento (CPU), os módulos de I/O (ou módulos de entrada/ saída), a fonte de alimentação e a base.

A CPU do CLP compreende o microprocessador, o sistema de memória (ROM e RAM) e os circuitos auxiliares de controle. Os módulos de I/O são dispositivos através dos quais podemos conectar sensores, atuadores ou outros equipamentos à CPU do CLP; assim, a CPU pode ler sinais de entrada, ou enviar sinais para a saída do CLP através dos módulos de I/O. Esses módulos podem ser discretos ou analógicos. A fonte de alimentação é responsável pela tensão de alimentação fornecida à CPU e aos módulos de I/O. A base do CLP proporciona conexão mecânica e elétrica entre a CPU, os módulos de I/O e a fonte. Ela contém o barramento de comunicação entre eles, em que estão presentes os sinais de dados, endereço, controle e tensão de alimentação (Georgini, 2006).

A programação de um CLP pode ser feita através de uma variedade de linguagens. Uma das mais populares é a linguagem Ladder, tendo recebido este nome devido à sua semelhança com uma escada (ladder), na qual duas barras verticais paralelas são interligadas pela lógica de controle, formando os degraus (rungs) da escada (Georgini, 2006). Na Figura 4.1 temos uma representação de lógica de controle através da linguagem ladder:

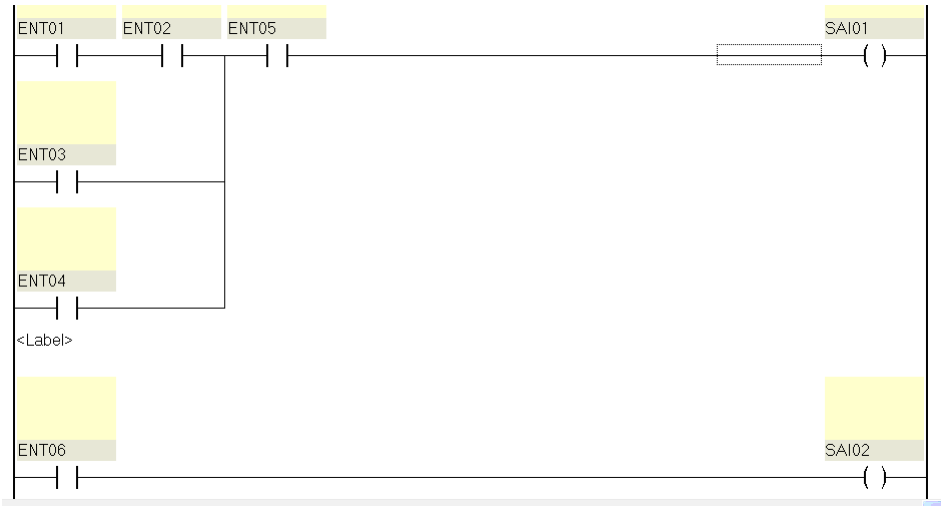

Figura 4.1: Exemplo de diagrama Ladder

O diagrama da Figura 4.1 apresenta uma lógica de controle com dois rungs: o primeiro é formado por 3 linhas (primeira linha – ENT01, ENT02, ENT05 e SAI01; segunda linha – ENT03; terceira linha – ENT04) e o segundo é formado por uma linha (ENT06 e SAI02). Este diagrama é formado por contatos e bobinas. As

entradas são os contatos, e as saídas são as bobinas. Cada elemento da lógica de controle representa uma instrução da linguagem Ladder, sendo alocado em um endereço específico e consumindo uma determinada quantidade de memória disponível para armazenamento do programa de aplicação (Georgini, 2006). Os contatos utilizados nas entradas do diagrama da Figura 4.1 são chamados contatos normalmente abertos (NA). Nesses contatos, há passagem de corrente se o contato for fechado. Assim, se o contato for uma botoeira, o contato será fechado se o botão for pressionado, permitindo a passagem de corrente; enquanto o botão não estiver apertado, o contato estará aberto, e não haverá passagem de corrente. Já nos contatos normalmente fechados (NF) ocorre o contrário: há passagem de corrente se o contato for aberto; enquanto o contato estiver fechado, não há passagem de corrente.

contato normalmente aberto (NA) contato normalmente fechado (NF)

Figura 4.2: Tipos de contatos

O fluxo de corrente elétrica fictícia num diagrama Ladder sempre ocorre da extremidade esquerda para a extremidade direita. Assim, no diagrama da Figura 4.1, se os contatos das entradas 1, 2 e 5 forem fechados simultaneamente, haverá passagem de corrente fictícia pela linha 1 do rung 1. Então, a bobina SAI01 será energizada, acionando a saída 1. Se forem fechados apenas os contatos ENT01 e ENT05, mantendo ENT02 aberto, não haverá a passagem de corrente fictícia pela linha 1 do rung 1, e então a saída 1 não será acionada. Outros modos de acionar a saída 1 seriam fechar os contatos ENT03 e ENT05 simultaneamente, ou ainda fechar os contatos ENT04 e ENT05 simultaneamente. A seqüência de leitura do programa em linguagem Ladder é do rung superior para o rung inferior; dessa forma, no exemplo da figura, se fecharmos os contatos ENT03, ENT05 e ENT06, primeiro será acionada a saída SAI01, e depois a saída SAI02.

4.2 Elementos de linguagem Ladder

4.2.1 Lógica básica

Uma instrução lógica ‘E’ é implementada do seguinte modo na linguagem Ladder.

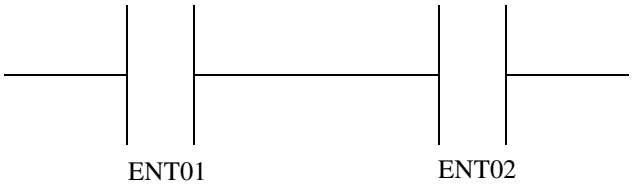

Figura 4.3: ‘E’ lógico em Ladder

Assim, se fecharmos os contatos ENT01 ‘E’ ENT02, haverá passagem de corrente pelo rung. Já a instrução lógica ‘OU’ é implementada como dois contatos em paralelo (Figura 4.4)..

Deste modo, se fecharmos o contato ENT01 ‘OU’ o contato ENT02, haverá passagem de corrente pelo rung.

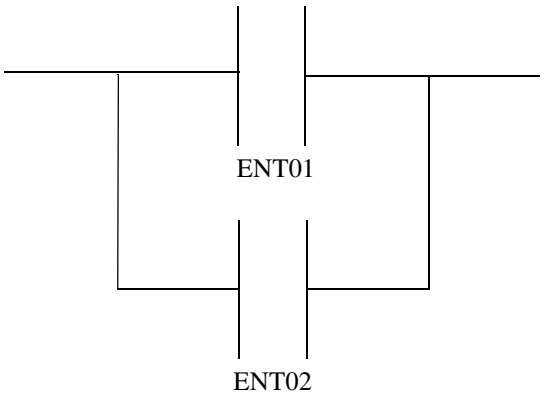

Figura 4.4: 'OU' lógico em Ladder

4.2.2 Temporizadores

Os temporizadores, como o nome sugere, são utilizados para temporização de condições no diagrama Ladder. O temporizador é identificado pela letra T seguida dos dígitos correspondentes ao seu endereço: T0, T1, etc. Há um bit de status relacionado ao temporizador, o qual é ativado quando o valor atual do temporizador for maior ou igual ao valor de preset (valor pré-configurado). O incremento de tempo depende do temporizador utilizado. O temporizador abaixo apresenta uma entrada de controle (enable) que, ao ser acionada (rung = 1), habilita o início da temporização, e ao ser desligada (rung = 0), reinicia o temporizador, mantendo-o nesta condição até novo acionamento da entrada enable (Georgini, 2006).

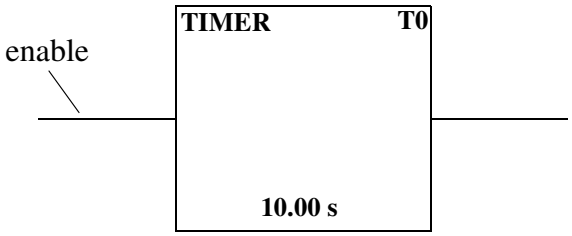

Figura 4.5: Temporizador

Na lógica de controle da Figura 4.6, quando os contatos BOT01 e BOT02 são fechados e permanecem fechados simultaneamente, é iniciada a temporização de T1, que tem um valor de preset fixo em 10 segundos (este temporizador apresenta incremento de tempo de 0.01 s). Ao ser atingido o valor de preset, o bit de status de T1 é acionado, ativando a saída SAI01. O bit de status permanece ativado até que o temporizador seja desativado (ou seja, quando um dos contatos BOT01 ou BOT02 for aberto). Esse bit pode ser associado a um contato normalmente aberto, conforme podemos observar na figura.

Também é possível ter acesso ao valor atual do temporizador durante a execução do diagrama Ladder utilizando-se os bits de valor atual. Por exemplo, TW0 armazena o valor atual de T0, TW1 armazena o valor atual de T1, e assim por diante, sendo que esses dados podem ser utilizados na lógica de controle.

A Figura 4.7 ilustra a utilização conjunta de temporizadores e comparadores. Assim, ao ser fechado o contato BOT01, é iniciada a temporização de T1. Quando o valor atual do temporizador T1 for maior ou igual a 1s, a saída SAI01 é acionada. Quando o valor atual de T1 for maior ou igual a 2s, a saída SAI02 é acionada, e quando esse valor atingir 3s, a saída SAI03 é acionada. O temporizador segue sua ação até atingir 10s, quando seu valor atual é zerado até que ele seja habilitado novamente através do acionamento de BOT01.

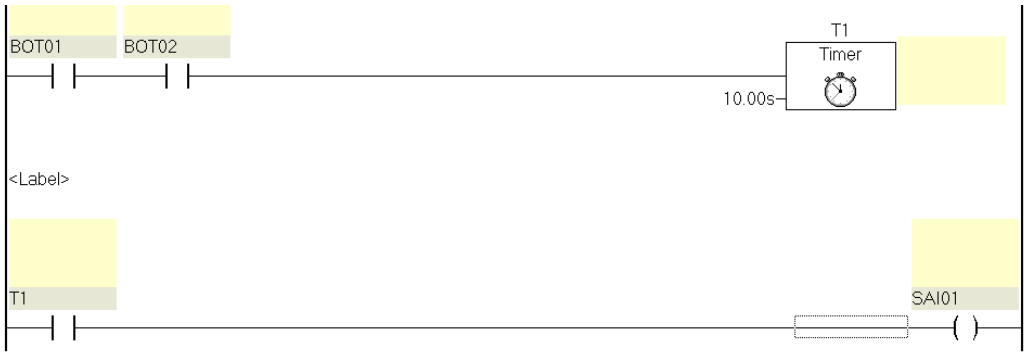

Figura 4.6: Lógica com temporizador

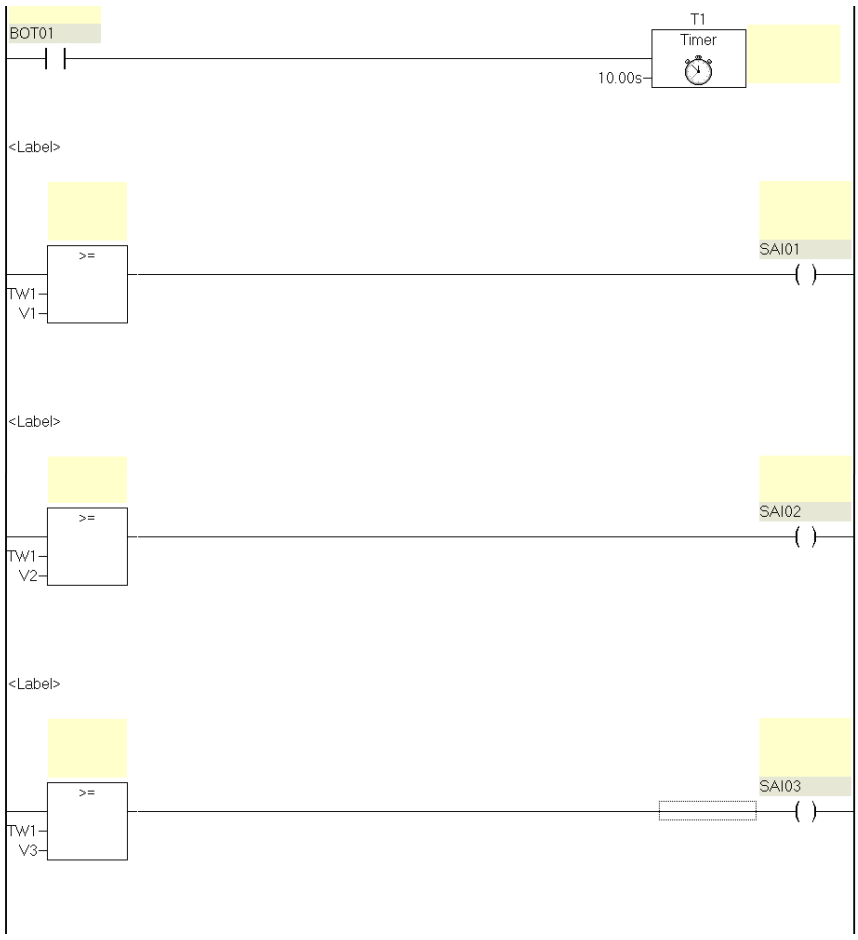

Figura 4.7: Lógica com temporizador e comparadores

4.2.3 Contadores

Os contadores são utilizados para contagem de condições ou eventos controlados pelo programa de aplicação. O contador é identificado pela letra C seguida do número do contador (C0, C1, C2, etc.). Há um bit de status relacionado ao contador, que é ativado quando o valor atual deste atingir o valor de preset (valor pré-configurado) (Georgini, 2006).

@ ERRRO!!!!! @Na lógica de controle da Figura 4.8, a cada transição de 0 para 1 (ou seja, off on) da entrada BOT01, o valor atual de C1 é incrementado em uma unidade, apresentando valor de preset de 20. Quando o valor de preset é atingido, o bit de status de C1 é ativado, acionando a saída SAI01. Esse bit de sta-

tus permanece ativado durante um ciclo do programa; no loop seguinte, o valor do bit de status de C1 é zerado automaticamente.

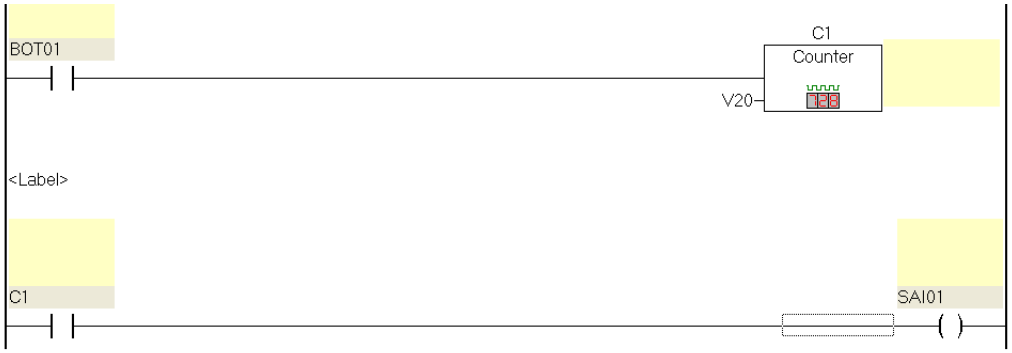

Figura 4.8: Lógica com contador

Além de ativar o bit de status quando atinge o valor de preset, o contador também oferece acesso ao seu valor atual durante a execução do programa de aplicação. Assim, a variável CW0 armazena o valor atual de C0, CW1 armazena o valor atual de C1, e assim por diante. Essas variáveis podem ser utilizadas na construção da lógica de controle.

A Figura 4.9 apresenta uma seqüência lógica envolvendo um contador e comparadores.

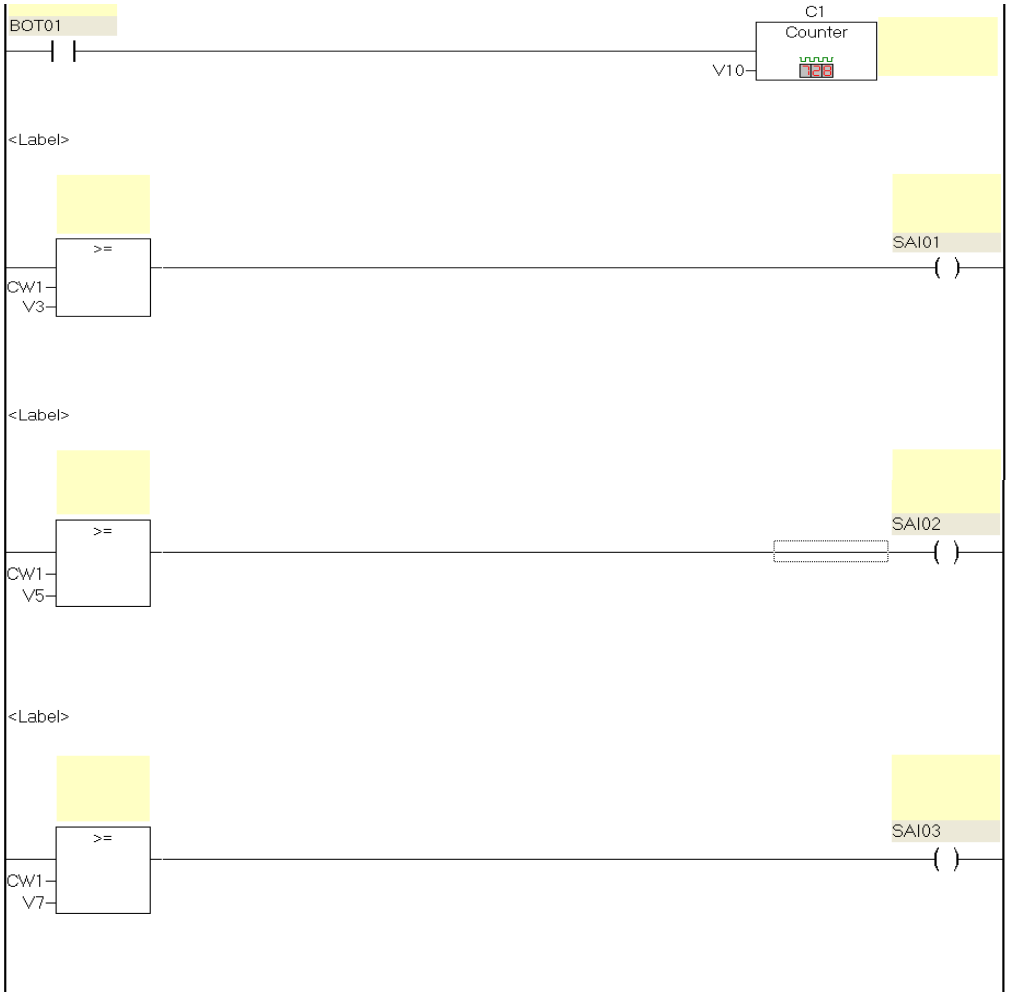

Figura 4.9: Lógica com contador e comparadores

Na lógica de controle da Figura 4.9, a cada transição de 0 para 1 do contato BOT01, o valor atual de C1 é incrementado em 1 unidade. Ao atingir a contagem de 3, a saída SAI01 é acionada; ao atingir a contagem de 5, a saída SAI02 é acionada, e ao atingir a contagem de 7, a saída SAI03 é acionada.

4.2.4 Funções SET e RESET

Ao ser executada (ou seja, quando o rung ligado à bobina set for igual a 1), a função set aciona o operando controlado, mantendo-o acionado mesmo que o rung não permaneça acionado (rung = 0). O operando acionado pela função set pode ser desligado pela função reset; esta função o manterá desligado mesmo que o rung não permaneça acionado. Se as instruções set e reset forem executadas durante o mesmo ciclo sobre o mesmo operando, terá controle final sobre o operando aquela que for executada por último (Georgini, 2006).

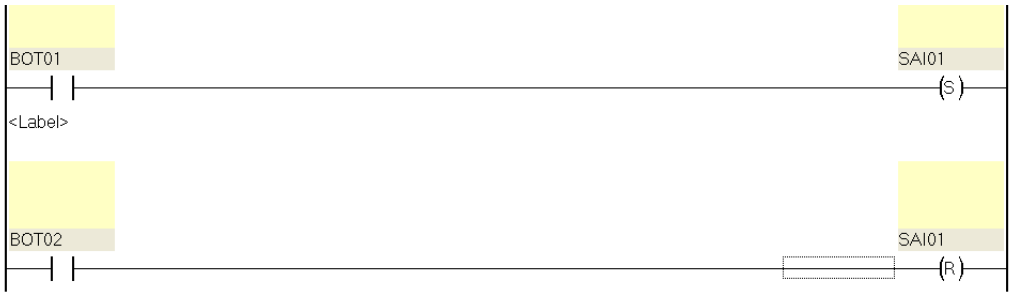

Figura 4.10: Lógica com SET e RESET

Na lógica de controle da Figura 4.10, ao ser pressionado o botão BOT01, a saída SAI01 será acionada, permanecendo nesta condição mesmo que BOT01 não continue sendo apertado. Ao ser pressionado BOT02, a saída SAI01 é desligada, permanecendo assim mesmo que BOT02 não continue sendo apertado. Se os botões BOT01 e BOT02 forem pressionados simultaneamente, a saída SAI01 é desligada, já que a instrução RESET será executada por último.

4.2.5 Como iniciar uma seção de programação

Primeiramente, deve-se iniciar o programa FST 4.10 da FESTO clicando-se no seu ícone, presente na área de trabalho do Windows. Uma vez iniciado o programa, clicar no item *'New'* do menu *'Project'*. Aparecerá uma tela pedindo um nome para o programa que será construído. Deve-se escolher um nome e clicar em *'ok'*. Aparecerá outra janela, *'Project Settings'*, perguntando o tipo de controlador a ser utilizado. Deve-se escolher a opção '*FEC Compact'* (veja a Figura 4.11):.

No item *'Project Settings'* , é possível adicionar comentários ao programa. Para começarmos a construir o diagrama ladder, devemos clicar no item *'New'* do menu *'Program'*. Aparecerá uma janela perguntando o tipo de programa a ser elaborado. São duas possibilidades: diagrama ladder e lista de instruções. Trataremos aqui apenas da primeira opção. Seleciona-se então a opção diagrama ladder. Outra janela de opções se abre, onde é possível definir a versão e o número do programa, bem como adicionar comentários a ele. Clicando-se em 'ok' , aparecerá um diagrama ladder em branco:

Ao se escolher o item *'Shortcuts'* do menu *'View'* aparece na tela uma barra de ferramentas. Esta barra tem as principais funções a serem utilizadas na construção de um diagrama ladder.

A opção *'Help'* do programa oferece a documentação completa, tanto do CLP como do software de programação.

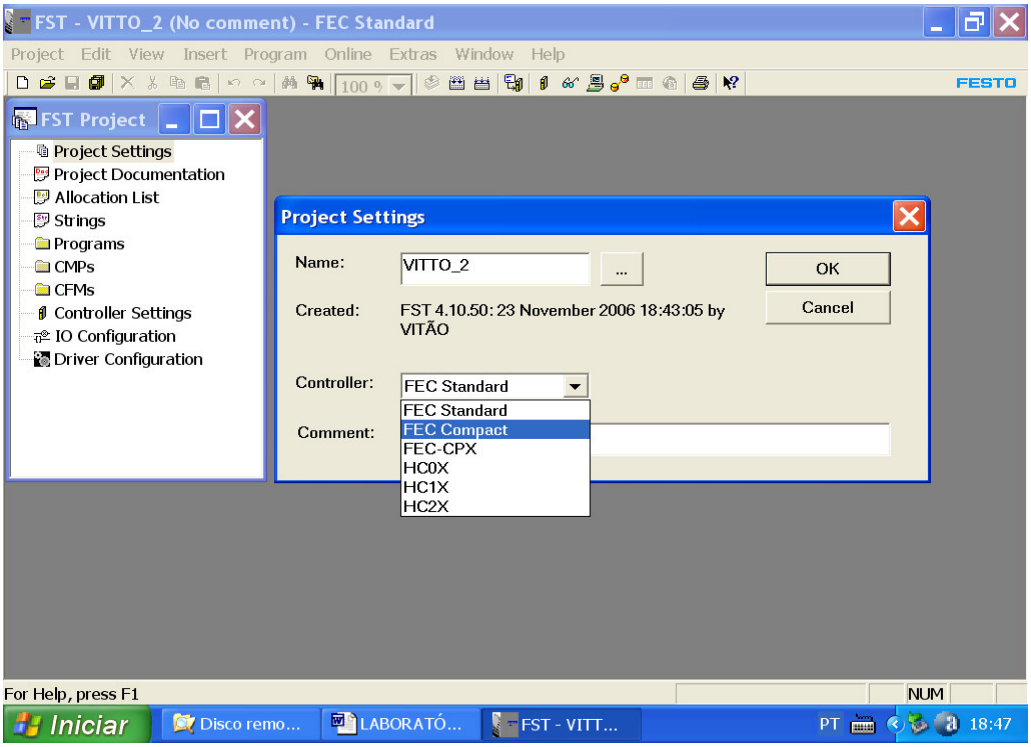

Figura 4.11: O software de programação do CLP FESTO

4.3 Exercícios

4.3.1 Exercícios simples

a) Deseja-se acender e apagar uma lâmpada através de um botão. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) Um dispositivo de uma indústria metalúrgica tem como função a fixação de peças em um molde. Esta fixação é feita por um atuador linear de dupla ação que avança mediante o acionamento de dois botões (S1 e S2) e retorna caso os botões sejam desacionados. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

c) Verificar através de botões e de uma lâmpada a tabela da verdade da função ou-exclusivo. Elabore um programa em linguagem Ladder e teste no CLP.

4.3.2 Tanque industrial

a) Elabore um diagrama Ladder simplificado para encher ou esvaziar um tanque industrial por meio de duas eletroválvulas. A eletroválvula V1 permite a entrada de líquido e a V2 permite o escoamento de saída. Quando o líquido atinge o nível máximo do tanque, um sensor A envia um sinal para o circuito lógico. Abaixo do nível máximo o sensor A não envia sinal algum. Há ainda um botão B, que deve encher o tanque quando for acionado e esvaziar em caso contrário. O esquema do tanque está apresentado na Figura 4.12 e as convenções do funcionamento do sistema estão apresentados na Tabela 4.13.

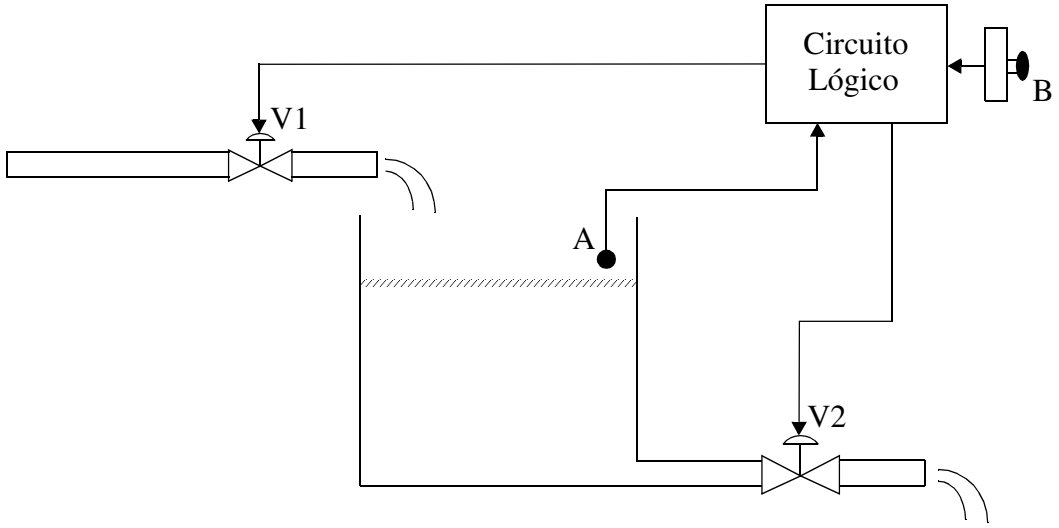

Figura 4.12: Esquema do tanque industrial

Tabela 4.13: Convenções para o tanque industrial

Sinal	Significado
A=1	Tanque cheio
A=0	Tanque não cheio
B=1	Comando encher
B=0	Comando esvaziar
V1=1	Comando fechar V1
V1=0	Comando abrir V1
V2=1	Comando fechar V2
V2=0	Comando abrir V2

4.3.3 Usando contadores

Há várias modalidades de contagem que podem ser utilizadas no CLP.

a) Deseja-se acender uma lâmpada após um botão ser acionado cinco vezes. Outro botão apaga a lâmpada (se ela estiver acesa) e reinicia a contagem. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

4.3.4 Exercícios com temporizadores

O CLP da FESTO também possui diferentes tipos de temporizadores. Consulte a documentação do programa para escolher o mais adequado a cada tipo de problema.

a) Deseja-se acender uma lâmpada de alarme durante 10s, quando um botão de emergência é acionado. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) O alarme A de uma casa é ativado por um sensor de movimento M e por um sensor de abertura de janelas J. O sensor M ativa o alarme quando detecta a presença de pessoas. O sensor J ativa o

alarme quando a janela é aberta. Há ainda um botão B para ligar ou desligar o alarme. Supondo que o alarme deve ser acionado por 10s e depois desligar automaticamente, elabore um diagrama ladder simplificado para resolver este problema. As convenções estão indicadas na Tabela4.14.

Tabela 4.14: Convenções para o sistema de alarme

Sinal	Significado
B=0	Comando desligar alarme
B=1	Comando ligar alarme
M=0	Ausência de pessoas
M=1	Presença de pessoas
J=0	Janela aberta
J=1	Janela fechada

c) Um sistema de dois semáforos controla o tráfego de um cruzamento de duas ruas (rua A e rua B), conforme a Figura 4.15,, sendo que cada semáforo está posicionado numa das ruas. A seqüência de

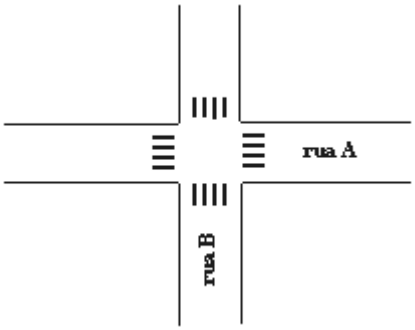

Figura 4.15: Cruzamento de ruas

acionamento de cada fase (amarelo, vermelho e verde) dos semáforos é mostrada na Tabela 4.16.

Tabela 4.16: Temporização

Fase	Tempo (s)	Semáforo A	Semáforo B
1	10	Verde	Vermelho
2	3	Amarelo	Vermelho
3	2	Vermelho	Vermelho
4	10	Vermelho	Verde
5	3	Vermelho	Amarelo
6	2	Vermelho	Vermelho

Implemente o semáforo em um programa em linguagem Ladder e teste no CLP.

d) Seja um sistema de semáforos similar ao anterior, controlando o tráfego no cruzamento de duas vias, conforme a Figura 4.17. Há um semáforo na rua A e um na rua B (as setas indicam o sentido do tráfego em cada rua). A diferença é que agora há quatro semáforos adicionais (S1, S2, S3 e S4), sendo dois deles dotados de botão para travessia de pedestres (S3 e S4). Cada um está posicionado

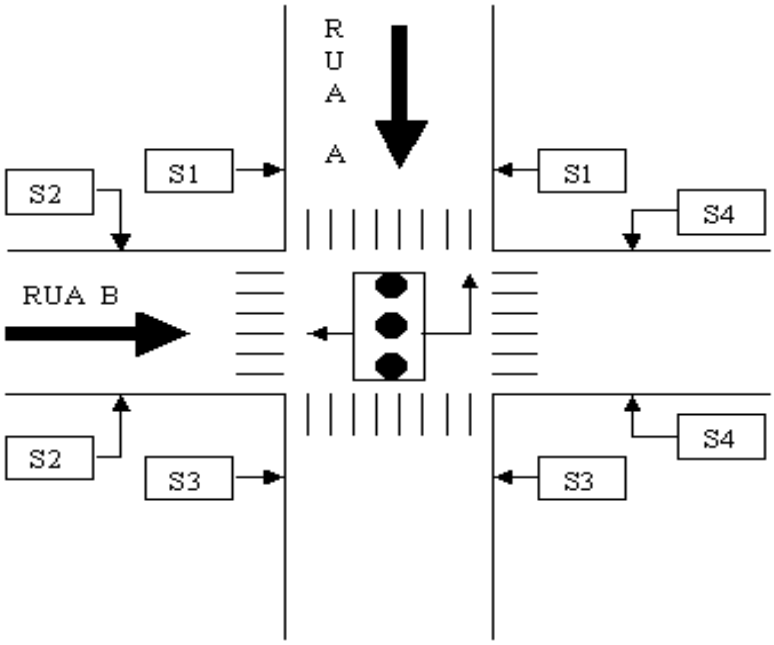

Figura 4.17: Cruzamento de ruas com travessia de pedestres

de um dos lados do cruzamento entre as ruas A e B. Ao acionamento do botão de S3 ou S4, os semáforos das ruas A e B devem passar de verde para amarelo, e deste para vermelho, a menos que já esteja(m) fechado(s) ou em processo de fechamento. Os estados correspondentes às fases dos semáforos são descritos na Tabela 4.18.

Tabela 4.18: Temporização

Fase	Tempo (s)	Semaf. A	S1	S3	Semaf. B	S2	S4
1	15	Verde	Vermelho	Vermelho	Vermelho	Verde	Vermelho
2	3	Amarelo	Vermelho	Vermelho	Vermelho	Verde	Vermelho
3	2	Vermelho	Vermelho	Vermelho	Vermelho	Vermelho*	Vermelho
4	15	Vermelho	Verde	Vermelho	Verde	Vermelho	Vermelho
5	3	Vermelho	Verde	Vermelho	Amarelo	Vermelho	Vermelho
6	2	Vermelho	Vermelho*	Vermelho	Vermelho	Vermelho	Vermelho
7	10	Vermelho	Verde	Verde	Vermelho	Verde	Verde
8	5	Vermelho	Vermelho*	Vermelho*	Vermelho	Vermelho*	Vermelho*

Nesta tabela, *Vermelho** indica vermelho piscante. As fases 7 e 8 só poderão ocorrer se durante as fases 1 a 6 houver o acionamento de pelo menos um botão para travessia de pedestres. Se isso ocorrer, o semáforo (A ou B) que estiver verde deve passar para amarelo (e ficar nessa fase durante 3 segundos), a menos que já esteja no amarelo. Após isso, passa-se para a fase 7 e, posteriormente, para a fase 8. Terminada a fase 8, volta-se à fase 1. Se nenhum botão for pressionado, a seqüência de fases é 1-2-3-4-5-6-1-2-etc. Implemente este semáforo em um programa em linguagem Ladder e teste no CLP.

4.3.5 Usando contadores e temporizadores

a) Deseja-se engarrafar bebidas de modo automático utilizando-se um CLP. As garrafas movimentam-se através de uma esteira rolante acionada por um motor elétrico, o qual é ligado e desligado pelo CLP, conforme a Figura 4.19. Quando cinco garrafas passarem por um sensor de presença (Sensor A), o motor deve ser desligado e um conjunto de cinco bicos injetores de cerveja deve ser acionado por 10 segundos (para encher as garrafas); após esses 10 segundos, o motor da esteira deve voltar a movimentá-la, até que outras cinco garrafas vazias passem pelo Sensor A; quando isso ocorrer, o processo se repetirá.

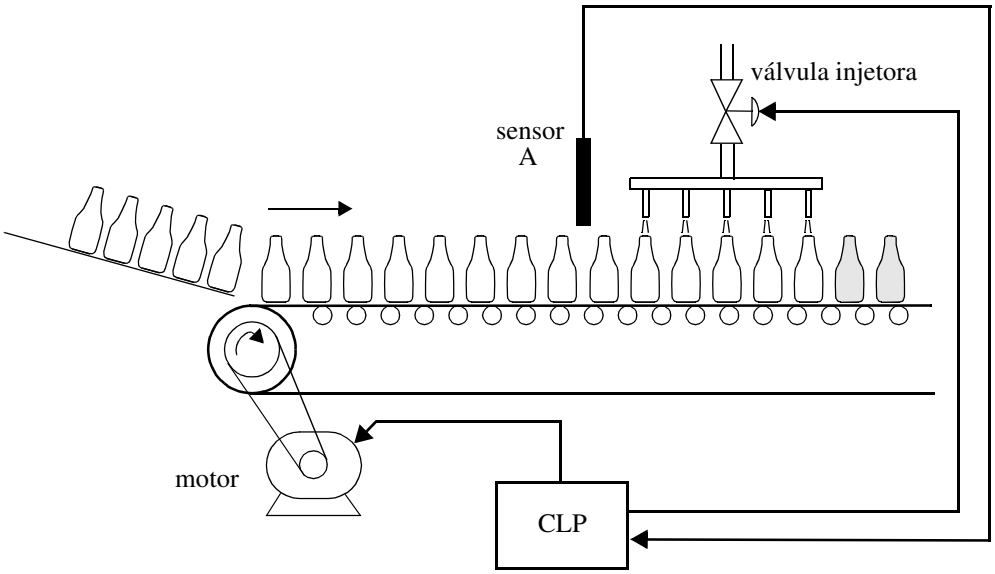

Figura 4.19: Engarrafadora

RPM/2009