Apostila do Curso
PTC2619 / PTC3418
Laboratório de Automação

Escola Politécnica da USP
PTC - Depto. de Engenharia de Telecomunicações e Controle
LAC - Laboratório de Automação e Controle

Edição 2018

Sumário

1 Introdução aos Controladores Lógicos Programáveis

1.1 Objetivo

Esta experiência tem por objetivo a familiarização com Controladores Lógicos Programáveis (CLPs) em aplicações práticas.

1.2 Introdução teórica

Os CLPs são dispositivos digitais capazes de armazenar instruções para implementação de **funções de controle**, tais como sequências lógicas, temporizações e contagens, bem como realizar **operações lógicas e aritméticas, manipulações de dados e comunicações em rede**.

São largamente utilizados na indústria e 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 Figura 1.1 ilustra a estrutura de um CLP.

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 1.2 temos uma representação de lógica de controle através da linguagem ladder:

O diagrama da Figura 1.2 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 –

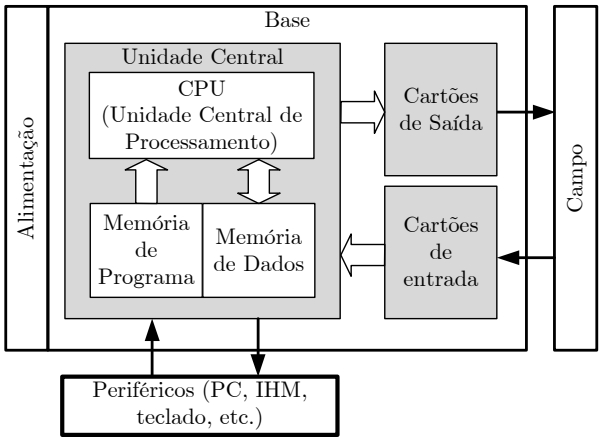

Figura 1.1: Diagrama de um CLP.

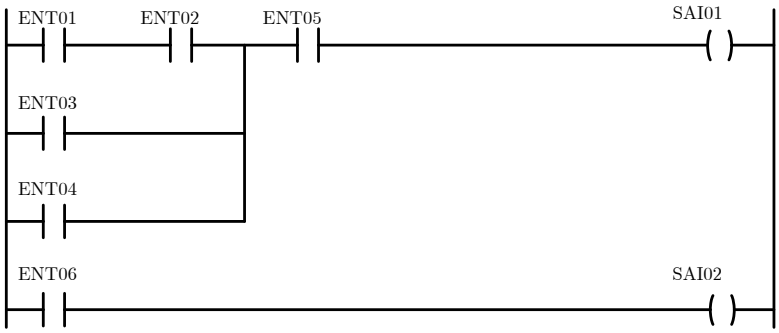

Figura 1.2: Exemplo de diagrama Ladder.

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.

Para uma introdução mais completa sobre CLPs, o leitor está convidado a consultar [Prudente 2013] e [Georgini 2006].

1.3 O CLP do laboratório

Os CLPs do laboratório são do fabricante Allen-Bradley, modelo CompactLogix 1769-L30ER. Eles podem ser programados, configurados e monitorados, via interface de comunicação USB, diretamente dos PCs, utilizando-se o programa RSLogix 5000.

A Figura 1.3 ilustra o CLP do laboratório.

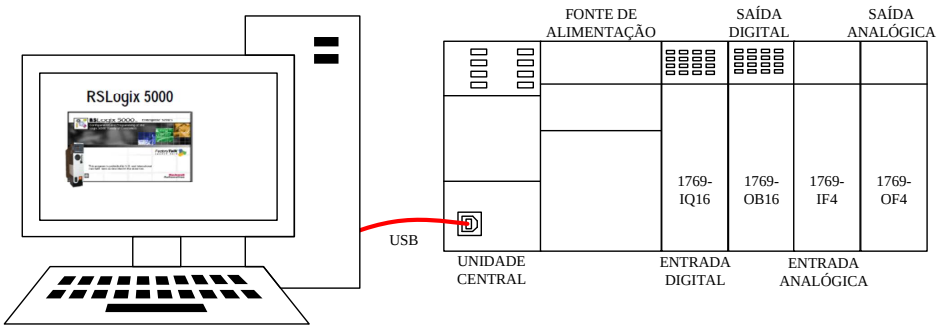

Figura 1.3: CLP do laboratório.

1.3.1 Como iniciar uma seção de programação

A seguir são apresentados os passos para criação de um novo projeto e elaboração de um programa básico em linguagem Ladder.

1. Conecte o CLP ao PC via cabo USB.
2. Abra o programa RSLogix 5000. Isso pode demorar alguns segundos;
3. Criação de um novo projeto.

   Clique em `New Project` e escolha o modelo `1769-L30ER`, como mostrado na Figura 1.4. Atribua um nome ao projeto.

   Observações:

   - Ao nomear projetos, *tags*, rotinas, módulos de E/S, etc, deve-se usar apenas letras, números e *underline* ("_"), onde o primeiro caractere não pode ser um número;
   - Não há distinção entre letras maiúsculas e minúsculas.

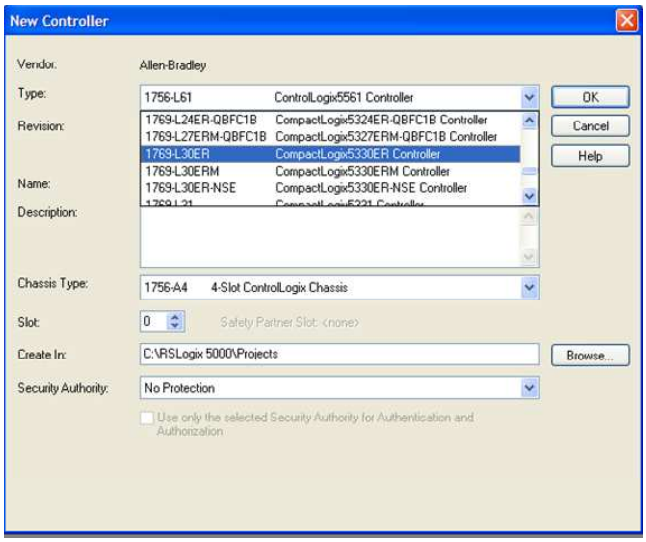

Figura 1.4: Criação do projeto e escolha do CLP.

4. Selecione a porta de comunicação com o CLP a ser empregada, que no nosso caso é USB. Clique no botão Who Active, conforme apresentado na Figura 1.5. Selecione `Set Project Path`. Isso é necessário para posteriormente fazer o download do programa no CLP. Feche a janela.

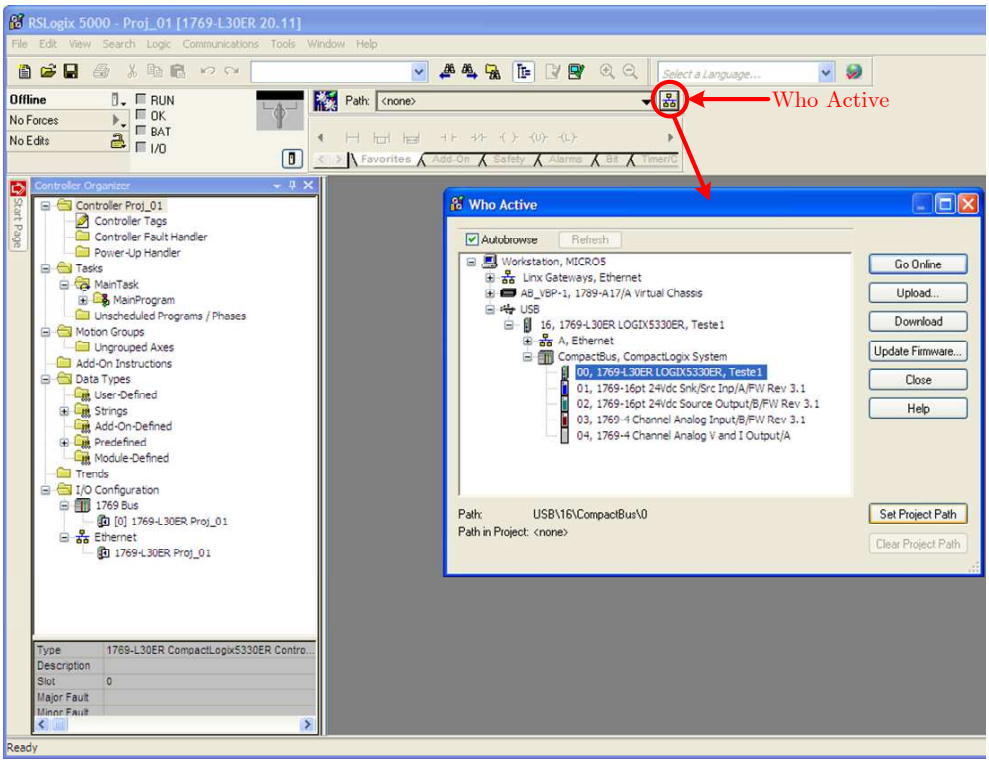

Figura 1.5: Definição da porta de comunicação USB.

5. Adicionar cartões ao CLP.

   Clique com o botão da direita em `I/O Configuration -> 1769 Bus`, e selecione `New Module...`, conforme a Figura 1.6.

   No chassi do CLP há quatro *slots* para cartões de I/O. A seguinte montagem está disponível no laboratório:

Código	Descrição Simplificada	Slot
1769-IQ16	16 Entradas Digitais, 24Vdc	1
1769-OB16	16 Saídas Digitais, 24Vdc	2
1769-IF4	4 Entradas Analógicas	3
1769-OF4	4 Saídas Analógicas	4

   Observações:

   - Os módulos devem ser adicionados **respeitando a ordem** que se encontram no chassi do CLP, da esquerda para a direita;
   - **Não inverter a posição** de um cartão no slot do CLP.

   Na tela `Select Module Type`, deve-se adicionar cada um dos quatro cartões separadamente em seus respectivos *slots*. Depois de encontrado o módulo, selecione `Create`. Como exemplo, para selecionar o módulo de entrada digital na posição 1, faça como apresentado na Figura 1.7.

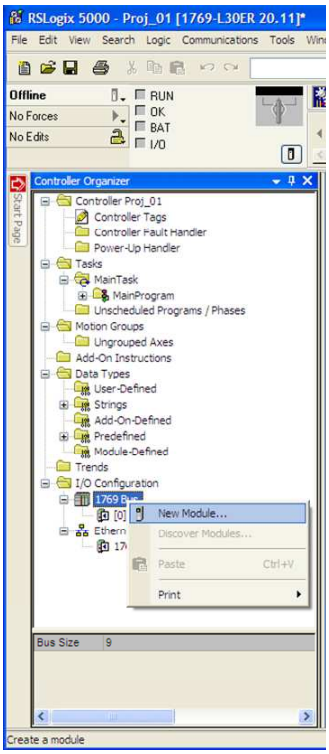

Figura 1.6: Configuração dos cartões do CLP.

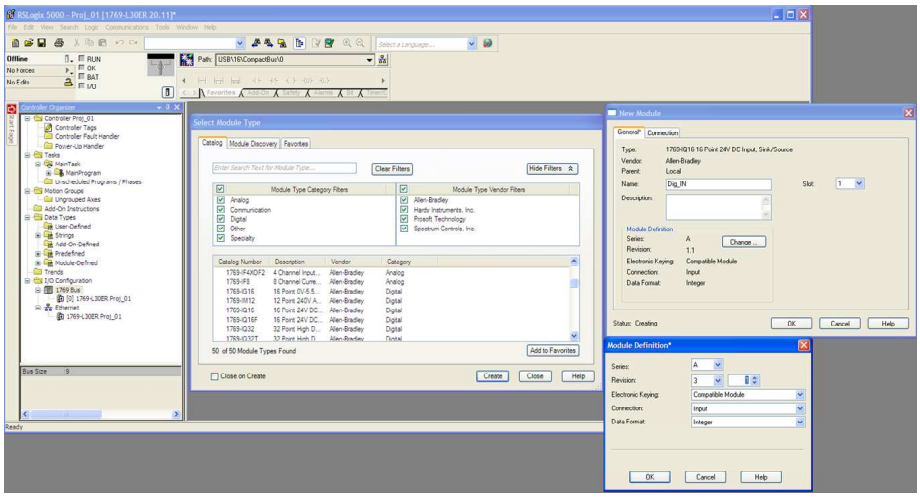

Figura 1.7: Configuração dos cartão de entrada digital.

- Na tela `New Module`, coloque um `Alias` para o cartão. Ex: `Dig_IN`
- O cartão de Entrada Digital está no *Slot* 1
- Clique em `Module Definition -> Change Module` e escolha a revisão mais recente, mas deixe-a na opção `Compatible Module`.

Como o endereçamento foi feito criando-se um *alias*, quando for necessário utilizar elementos já existentes, basta fazer um duplo clique no símbolo "?" do elemento e digitar o *alias*.

Repita as operações para os três cartões restantes.

Pronto, o CLP está configurado e o primeiro programa pode ser efetuado!

6. Para criar uma nova rotina de programa em Ladder, selecione a opção `Tasks -> MainTask -> MainRoutine`. Note que abrirá uma tela para iniciar o programa. No menu superior aparacerá uma série de contatos e componentes para compor o programa.

   No CLP CompactLogix um endereço segue o formato `Local:Slot:Tipo.Membro.Bit`, conforme o quadro abaixo:

`Local`	Indica que a localização do módulo de E/S está no mesmo rack do CLP
`Slot`	Número do Slot de E/S no rack
`Tipo`	`I`: entrada (input) `O`: saída (output) `C`: configuração
`Membro`	Para um módulo de E/S discretas, um membro do tipo Data armazena os valores dos bits de E/S
`Bit`	Ponto específico de um módulo de E/S discretas

1.3.2 Meu primeiro programa em Ladder

A seguir são apresentados os passos para a construção de um programa bem simples, que consiste em acender uma lâmpada (ligar um contato de saída) quando um botão (contato de entrada normalmente aberto — NA) é acionado.

a) **Inserção de um contato NA.** Adicione um contato de entrada clicando em seu símbolo na barra de ferramentas. O contato será endereçado criando-se um "*alias*". Clique com o botão direito do mouse sobre o símbolo "?" do contato e selecione `New Tag`. A tela da Figura 1.8 será aberta.

   Escolha as seguintes configurações:

   - `Name`: Escolha um nome para a tag do botão (Ex: BT_1);
   - `Type`: Alias;

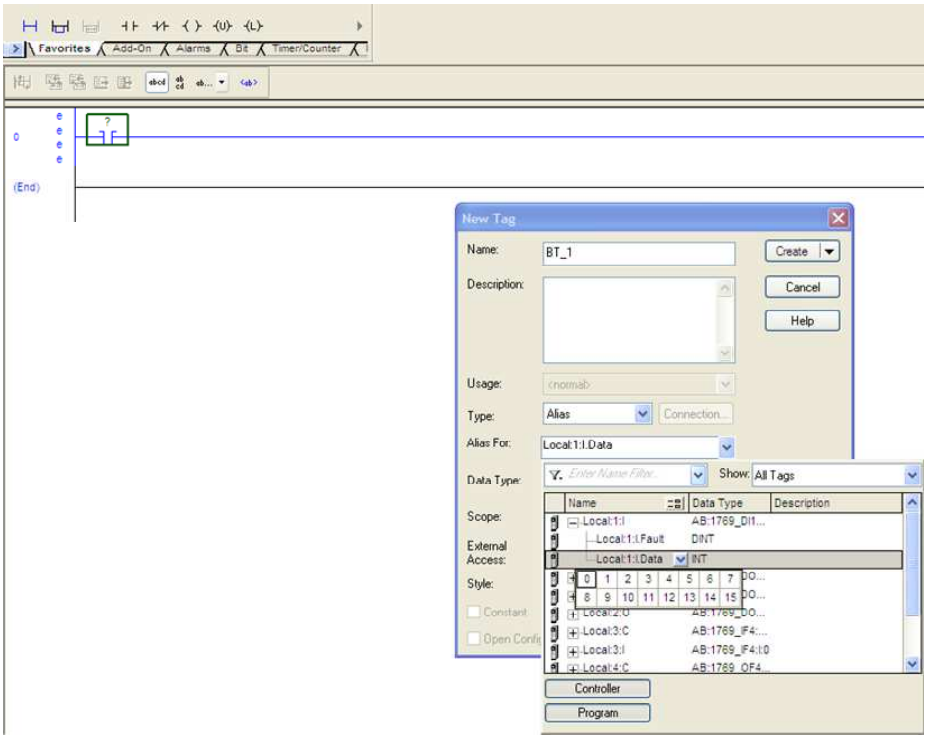

Figura 1.8: Configuração e endereçamento do contato de entrada para o botão.

- `Alias For`: selecione `Local:1:I.Data`. Clique na flecha para baixo e escolha a entrada digital referente ao botão (0 a 7). Por exemplo, se a entrada 0 for selecionada, deve aparecer o endereço `Local:1:I.Data.0`;
- `Data Type`: BOOL.

b) **Inserção de um contato de saída.** Adicione um contato de saída clicando em seu símbolo na barra de ferramentas. Clique com o botão direito do mouse sobre o símbolo "?" do contato e selecione `New Tag`. A tela da Figura 1.9 será aberta.

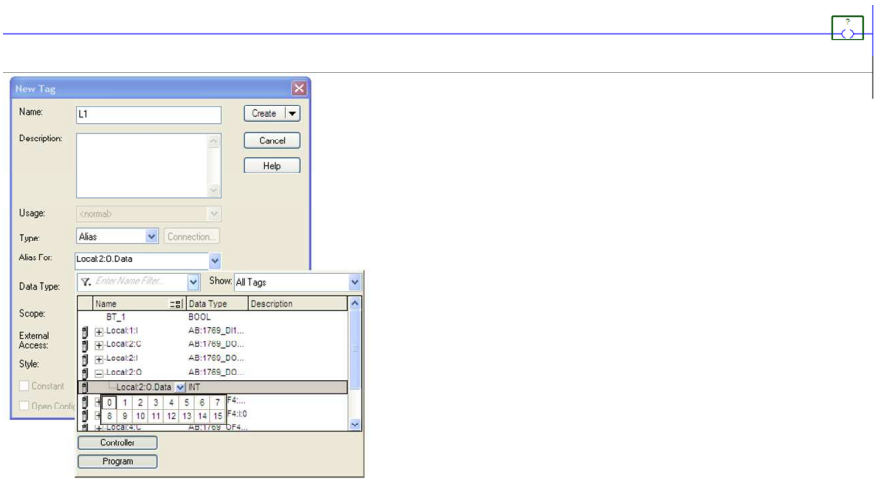

Figura 1.9: Configuração e endereçamento do contato de saída para a lâmpada.

Escolha as seguintes configurações:

- `Name`: Escolha um nome para a tag do botão (Ex: L_1);
- `Type`: Alias;
- `Alias For`: selecione `Local:2:O.Data`. Clique na flecha para baixo e escolha a saída digital referente à Lâmpada (0 a 7). Por exemplo, se a saída 0 for selecionada, deve aparecer o endereço `Local:2:O.Data.0`;
- `Data Type`: BOOL.

Se tudo estiver certo, as indicações de erro no lado esquerdo das linhas do programa desaparecerão.

c) Salve o programa. Em seguida, clique na flecha ↓ ao lado de `Offline` no menu superior e selecione a opção `Download`. Verifique as mensagens.

d) Faça as ligações físicas no CLP, conforme apresentado na Figura 1.10. Selecione a opção `Run Mode` para executar o programa. Para modificar o código, vá novamente para o modo `Offline`.

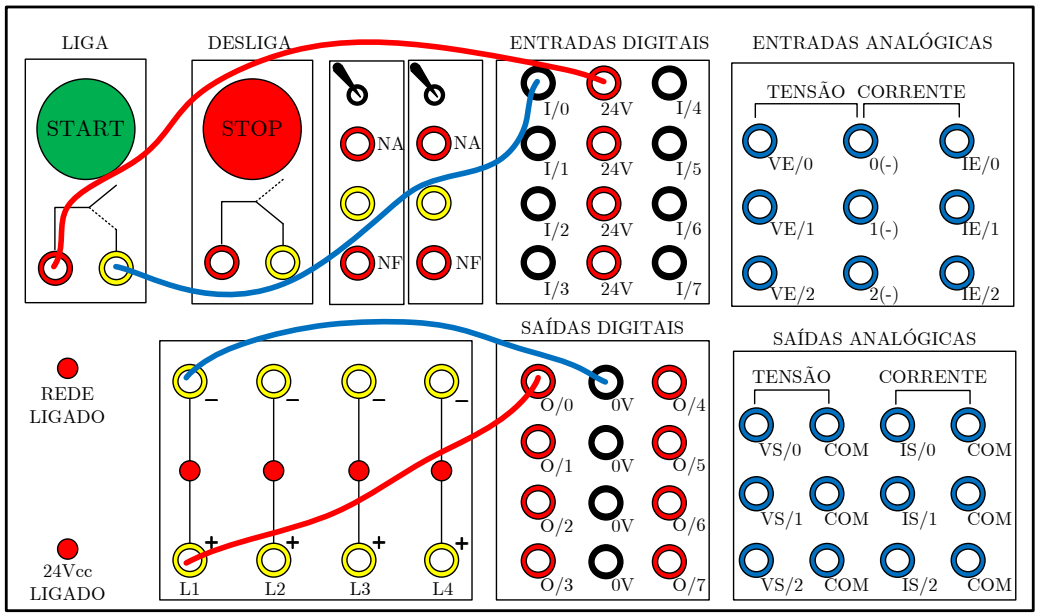

Figura 1.10: Configuração e endereçamento do contato de saída para a lâmpada.

1.3.3 Modos de funcionamento do CLP

A seguir é apresentada uma descrição sucinta dos modos de operação do CLP.

- Modo RUN: modo em que o programa carregado é atualizado e as saídas são atualizadas. Não permite edição do programa;
- Modo PROG: modo de criação e edição de códigos *offline*, ou seja, as saídas estão desabilitadas;
- Modo REM: modo remoto, com as seguintes opções:

- REMOTE RUN: modo onde o programa carregado é executado e as saídas são atualizadas, mas permite a edição *online*;
- REMOTE PROG: permite a edição online do programa, mas as saídas estão desabilitadas.
- REMOTE TEST: semelhante ao REMOTE RUN, mas com saídas desabilitadas.

O CLP a ser utilizado no laboratório está no modo REM. Como sugestão, para criar/editar o programa, deixe-o em modo Offline.

1.3.4 Funções básicas do CLP

A Tabela 1.1 apresenta as instruções de bit (contatos e bobinas) no editor Ladder do RSLogix 5000:

Tabela 1.1: Instruções de bit.

Contatos	$--\|\ \ \|--$	normalmente aberto
	$--\|\,/\,\|--$	normalmente fechado
Bobinas	$--(\ \ )--$	bobina simples
	$--(\,\mathrm{L}\,)--$	bobina tipo *latched*
	$--(\,\mathrm{U}\,)--$	bobina tipo *unlatched*

As bobinas L (liga um bit) e U (desliga um bit) são retentivas, ou seja, manterão seu estado mesmo que as condições de entrada da linha se tornem falsas.

O comando $|-|$ insere uma nova linha de comando no diagrama Ladder.

O comando $|\frac{}{|-|}|$ insere um ramo paralelo (*branch*) no diagrama Ladder.

No Ladder do RSLogix há uma instrução denominada ONS (*one shot*), com símbolo $--[\mathrm{ONS}]--$. De forma simplificada, essa expressão gera um pulso de duração de um período de *scan* quando a linha é energizada. Para gerar outro pulso, a linha precisa ser desenergizada e energizada novamente.

1.3.4.1 Algumas Instruções de Temporização do RSLogix

A seguir serão descritas três instruções de temporização presentes no CLP: TON e TOF e RTO. As duas primeiras instruções são não retentivas, ou seja, o acumulador de contagem de tempo é ressetado quando o contador não está habilitado. Já a terceira, trata-se de um temporizador retentivo, que mantém o valor do contador quando desabilitado.

- Instrução TON - Temporizador na Energização

A Figura 1.11 ilustra o diagrama Ladder da instrução

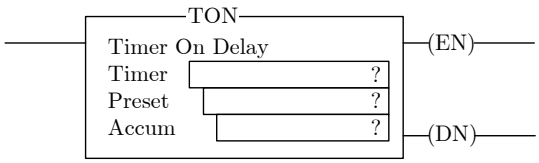

Figura 1.11: Instrução TON.

É utilizada para ligar ou desligar uma saída após um tempo especificado no valor Preset (PRE). A contagem de tempo se inicia quando a condição de linha é verdadeira. O Acumulador (ACC) é incrementado a cada ciclo de scan enquanto a linha permanece verdadeira, até que o valor de PRE seja alcançado.

O ACC é ressetado quando a condição da linha se torna falsa, independentemente do valor de PRE ter sido atingido. A base de tempo é em milissegundos (ms).

A tabela 1.2 apresenta a descrição dos bits de estado.

Tabela 1.2: Bits de estado de TON

Bit	Descrição
Timer Done (DN)	É setado quando o valor do ACC é igual ao PRE Permanece setado até que as condições da linha sejam falsas
Timer Timing (TT)	É setado se as condições da linha são verdadeiras e ACC < PRE Permanece setado até que as condições da linha sejam falsas ou DN seja setado
Timer Enable (EN)	É setado se as condições da linha são verdadeiras Permanece setado até que as condições da linha sejam falsas

Tais bits podem ser utilizados como *tags* de contatos no programa Ladder. Por exemplo, se um timer for denominado `T1`, pode-se atribuir a *tag* `T1.DN` a um contato.

Ao inserir um bloco TON, deve-se atribuir uma tag ao temporizador, como ilustrado na Figura 1.12

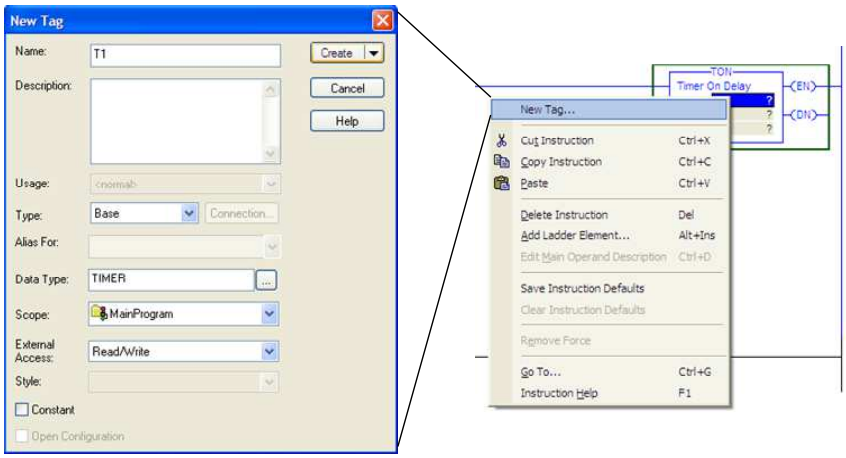

Figura 1.12: Atribuição de tag ao módulo TON.

Em seguida, atribua um valor para o Preset e, se for necessário, ao Acumulador, lembrando que a base de tempo é em milissegundos. Procedimentos similares devem ser executados para os demais temporizadores e contadores.

- Instrução TOF - Temporizador na Desenergização

A Figura 1.13 ilustra o diagrama Ladder da instrução.

Também utilizada para ligar ou desligar uma saída após um tempo especificado no valor Preset (PRE). A contagem de tempo se inicia quando a condição de linha passa de verdadeira para falsa. O Acumulador (ACC) é incrementado a cada ciclo de scan enquanto a linha permanece na condição falsa,

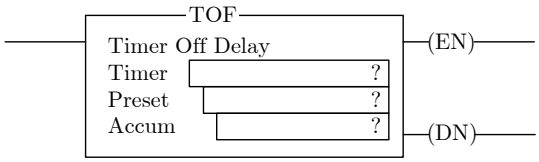

Figura 1.13: Instrução TOF.

até que o valor de PRE seja alcançado. O ACC é ressetado quando a condição da linha se torna verdadeira, independentemente do valor de PRE ter sido atingido.

A tabela 1.3 apresenta a descrição dos bits de estado.

Tabela 1.3: Bits de estado de TOF

Bit	**Descrição**
Timer Done	É ressetado quando o valor do ACC é igual ao PRE
(DN)	Permanece ressetado até que as condições da linha sejam verdadeiras
Timer Timing	É setado se as condições da linha são falsas e ACC $<$ PRE
(TT)	Permanece setado até que as condições da linha sejam verdadeiras ou DN seja ressetado
Timer Enable	É ressetado se as condições da linha são falsas
(EN)	Permanece ressetado até que as condições da linha sejam verdadeiras

Tais bits também podem ser utilizados como *tags* de contatos no programa Ladder.

- Instrução RTO - Temporizador Retentivo

A Figura 1.14 ilustra o diagrama Ladder da instrução

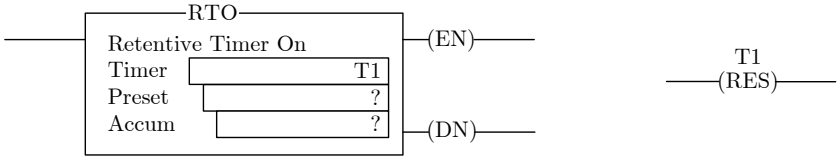

Figura 1.14: Instrução RTO.

Também utilizada para ligar ou desligar uma saída após um tempo especificado no valor Preset (PRE). A diferença é que o valor do acumulador é mantido quando a condição da linha se torna falsa, quando o modo de programação passa de RUN para PROGRAM, quando a alimentação é perdida ou uma falha ocorrer. A memória acumulada é, portanto, não volátil. Ao voltar para o modo RUN, a contagem é restabelecida a partir do valor armazenado.

Para ressetar os bits de estado e o valor do ACC, é necessário programar uma instrução de reset (RES) com o endereço do temporizador em uma outra linha do código Ladder.

A descrição dos bits de estado é igual ao do TON apresentada na Tabela 1.2. Tais bits também podem ser utilizados como *tags* de contatos no programa Ladder.

1.3.4.2 Algumas Instruções de Contagem do RSLogix

A seguir serão apresentadas duas instruções de contagem utilizadas na programação do CLP: CTU e CTD. A primeira se trata de um contador crescente, enquanto a segunda representa um contador decrescente.

- `Instrução CTU - Contador Crescente`

A Figura 1.15 ilustra o diagrama Ladder da instrução.

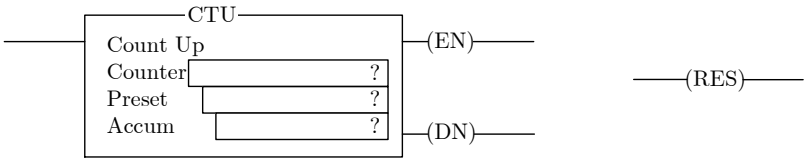

Figura 1.15: Instrução CTU.

Em tal instrução, o valor do acumulador é incrementado quando ocorrem transições falso $\rightarrow$ verdadeiro da linha, que podem ser devidas a eventos internos de programação ou dispositivos externos, como botões, sensores de presença, etc. O valor do acumulador, assim como dos bits de estado, são retidos quando a linha torna-se falsa. Portanto, para ressetar o contador, deve-se programar uma instrução de reset (RES) com o mesmo endereço do contador em outra linha.

A tabela 1.4 apresenta a descrição dos bits de estado. Tais bits podem ser utilizados como *tags* de contatos no programa Ladder.

Tabela 1.4: Bits de estado de CTU

Bit	**Descrição**
Overflow (OV)	É setado quando o valor do ACC $> +2.147.483.647$ Permanece setado até que RES seja executado ou seja decrementado utlizando CTD
Done (DN)	É setado se ACC $\geq$ PRE Permanece setado até que ACC $<$ PRE
Count Up Enable (CU)	É setado se as condições da linha são verdadeiras Permanece setado até que as condições da linha sejam falsas ou RES seja executado

- `Instrução CTD - Contador Decrescente`

A Figura 1.16 ilustra o diagrama Ladder da instrução.

Em tal instrução, o valor do acumulador é decrementado quando ocorrem transições falso $\rightarrow$ verdadeiro da linha, que também podem ser devidas a eventos internos de programação ou dispositivos externos, como botões, sensores de presença, etc. O valor do acumulador, assim como dos bits de estado, são retidos quando a linha torna-se falsa. Portanto, para ressetar o contador, deve-se programar uma instrução de reset (RES) com o mesmo endereço do contador em outra linha.

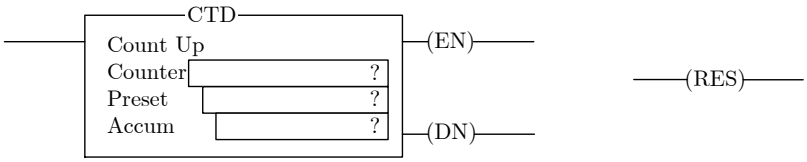

Figura 1.16: Instrução CTD.

A tabela 1.5 apresenta a descrição dos bits de estado. Tais bits também podem ser utilizados como *tags* de contatos no programa Ladder.

Tabela 1.5: Bits de estado de CTD

Bit	**Descrição**
Underflow (UN)	É setado quando o valor do ACC $< -2.147.483.648$ Permanece setado até que RES seja executado ou seja incrementado utilizando CTU
Done (DN)	É setado se ACC $\geq$ PRE Permanece setado até que ACC $<$ PRE
Count Up Enable (CU)	É setado se as condições da linha são verdadeiras Permanece setado até que as condições da linha sejam falsas ou RES seja executado

1.3.4.3 Algumas Instruções de Comparação do RSLogix

- `Instrução EQU – Equal to`

A Figura 1.17 ilustra o diagrama Ladder da instrução. A linha é energizada se a condição de *source A* for igual à *source B*.

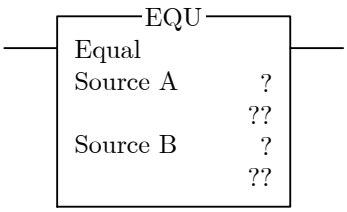

Figura 1.17: Instrução EQU.

Há também as instruções GEQ (*Greater Than or Equal To*), GRT (*Greater Than*), LEQ (*Less Than or Equal To*), LES (*Less Than*), com a mesma sintaxe da instrução EQU. Há ainda a instrução CMP (*Compare*) que verifica se a condição de uma dada expressão é verdadeira para energizar a linha. Todas as fuções de comparação descritas anteriormente podem ser implementadas com a CMP.

Para verificar a gama completa de instruções do RSLogix5000, favor consultar o manual de referência *Logix5000 Controllers General Instructions Reference Manual*.

1.4 Problemas com Solução

Problema 1: 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.

Solução: Inicialmente, as seguintes ligações apresentadas na Figura 1.18 são efetuadas.

Passo 1: Identifique as entradas e saídas do sistema.

As seguintes entradas são identificadas:

- $A \rightarrow$ estado da chave CH1;
- $B \rightarrow$ estado da chave CH2;

Saída:

- L estado da lâmpada;

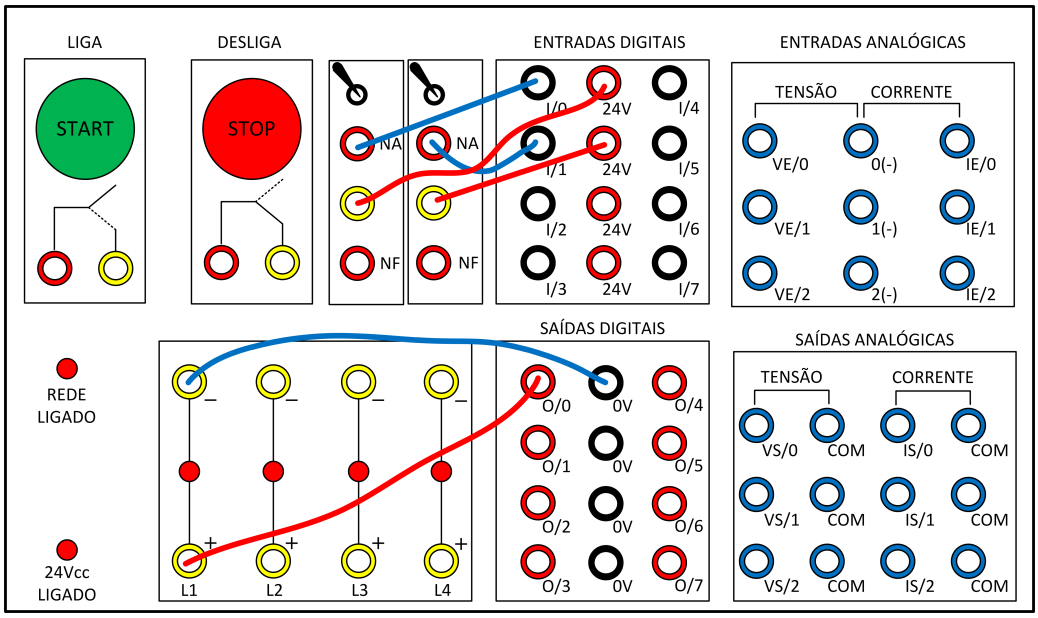

Figura 1.18: Ligações do exemplo.

Passo 2: Construção da Tabela Verdade. A tabela 1.6 é elaborada.

Tabela 1.6: Tabela Verdade do problema 1 com solução.

Linha	A	B	L
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

Passo 3: Obtenção de soma canônica (ou produto canônico). A soma canônica é obtida pela soma dos mintermos da tabela verdade. Um mintermo pode ser definido como o termo produto que resulta em exatamente "1" em uma dada linha da tabela verdade. A soma canônica é a expressão lógica que representa a tabela verdade Assim:

$$L = \sum_{A,B}(1,2) = (\bar{A}B) + (A\bar{B})$$

Passo 4: Simplificação da expressão. Neste caso, a soma canônica é a soma mínima, ou seja, não é possível simplificar mais a expressão resultante.

Passo 5: Elaboração da lógica em linguagem Ladder. As seguintes atribuições são feitas, de acordo com o painel de ligações do laboratório:

- $A \rightarrow$ chave CH1 (NA);
- $B \rightarrow$ chave CH2 (NA);

- $L \rightarrow$ lâmpada L;

O diagrama Ladder resultante é apresentado na Figura 1.19

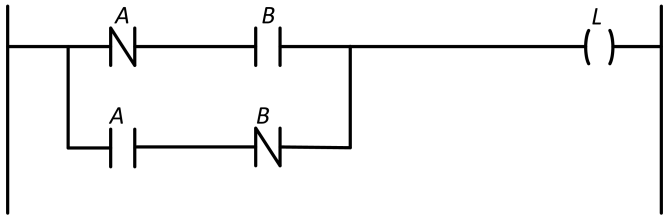

Figura 1.19: Diagrama Ladder da solução do problema 1.

A Figura 1.20 apresenta a tela da rotina programada no RSLogix 5000.

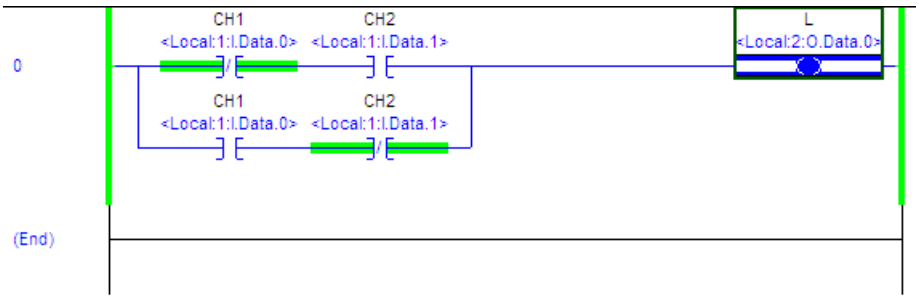

Figura 1.20: Diagrama Ladder da solução do problema 1 no RSLogix 5000.

Problema 2: A Figura 1.21 mostra o cruzamento de uma rodovia com uma via de acesso.

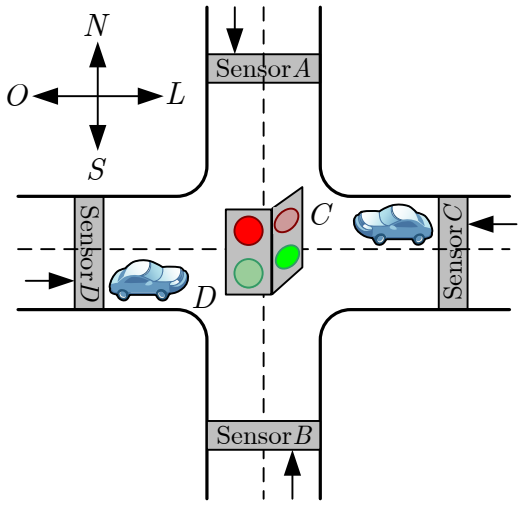

Figura 1.21: Ilustração do problema 2 com solução.

Sensores detectores de veículos são colocados ao longo das pistas C e D da rodovia e das pistas A e B da via de acesso. A saída desse tipo de sensor está em BAIXO quando não existe nenhum carro presente e está em ALTO quando um veículo está presente. Um sinal de trânsito colocado no cruzamento deve funcionar de acordo com a seguinte lógica:

- O sinal da direção leste-oeste LO deve estar verde quando as pistas C **E** D estiverem ocupadas.
- O sinal da direção LO deve estar verde quando C **OU** D estiverem ocupadas, e **NÃO** houver carros nas pistas A **E** B.
- O sinal da direção norte-sul NS deve estar verde quando ambas as pistas A **E** B estiverem ocupadas, e **NÃO** houver carros nas pistas C **E** D.
- O sinal da direção NS deve estar verde quando A **OU** B estiverem ocupadas e enquanto ambas as pistas C **E** D estiverem vazias.
- O sinal da direção LO deve estar verde quando **NÃO** houver nenhum veículo presente.

Utilizando as saídas dos sensores A, B, C e D como entradas, desenvolva uma lógica em Ladder que controle esse sinal de trânsito e implemente no CLP do laboratório. Devem existir duas saídas, NS e LO, que devem ir para ALTO quando o sinal correspondente estiver verde.

Solução: Inicialmente, as seguintes ligações apresentadas na Figura 1.26 são efetuadas.

Note que o problema é de natureza combinatória. Assim, o seguinte procedimento pode ser utilizado:

Passo 1: Identifique as entradas e saídas do sistema.

As seguintes entradas são identificadas:

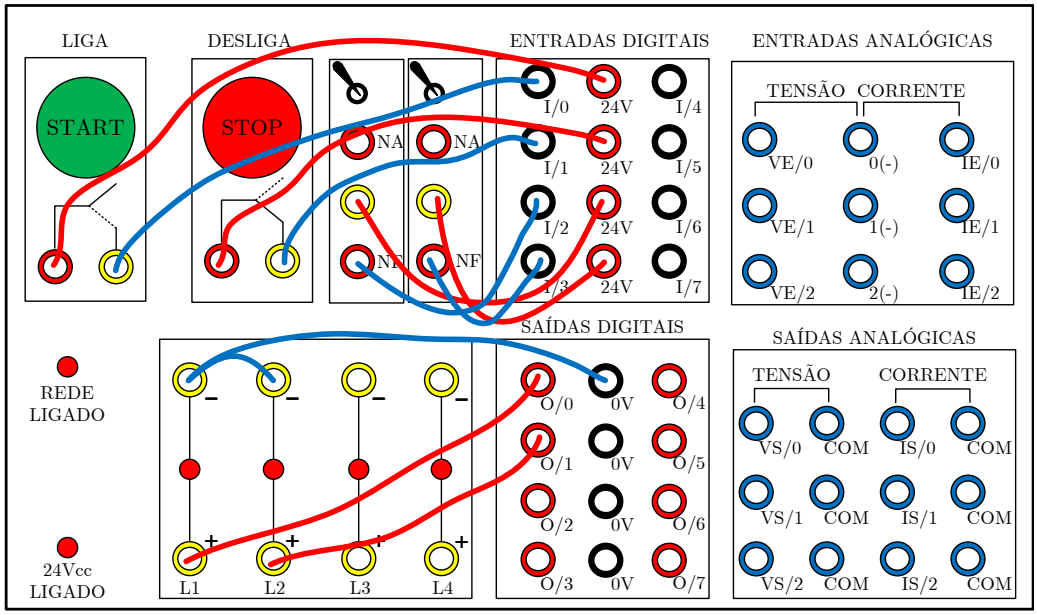

Figura 1.22: Ligações do exemplo.

- $A \rightarrow$ sensor da pista A;
- $B \rightarrow$ sensor da pista B;
- $C \rightarrow$ sensor da pista C;
- $D \rightarrow$ sensor da pista D;

Saídas:

- $NS \rightarrow$ Sentido Norte$\leftrightarrow$Sul habilitado;
- $LO \rightarrow$ Sentido Leste$\leftrightarrow$Oeste habilitado;

Passo 2: Construção da tabela verdade. De acordo com o enunciado, a Tabela 1.7 é construída.

Note que as saídas LO e NS são complementares, ou seja, basta resolver o problema para uma delas. Nesse caso, será escolhida a saída NS.

Passo 3: Obtenção de soma canônica (ou produto canônico). A soma canônica é obtida pela soma dos mintermos da tabela verdade. Um mintermo pode ser definido como o termo produto que resulta em exatamente "1" em uma dada linha da tabela verdade. A soma canônica é a expressão lógica que representa a tabela verdade. Assim:

$$NS = \sum_{A,B,C,D} (4, 8, 12, 13, 14) = (\bar{A}B\bar{C}\bar{D}) + (A\bar{B}\bar{C}\bar{D}) + (AB\bar{C}\bar{D}) + (AB\bar{C}D) + (ABC\bar{D})$$

Tabela 1.7: Tabela Verdade do problema 2 com solução.

Linha	A	B	C	D	LO	NS
0	0	0	0	0	1	0
1	0	0	0	1	1	0
2	0	0	1	0	1	0
3	0	0	1	1	1	0
4	0	1	0	0	0	1
5	0	1	0	1	1	0
6	0	1	1	0	1	0
7	0	1	1	1	1	0
8	1	0	0	0	0	1
9	1	0	0	1	1	0
10	1	0	1	0	1	0
11	1	0	1	1	1	0
12	1	1	0	0	0	1
13	1	1	0	1	0	1
14	1	1	1	0	0	1
15	1	1	1	1	1	0

Passo 4: Simplificação da expressão. Pode-se utilizar os axiomas da álgebra de Boole ou um método gráfico, como o mapa da Karnaugh. A segunda opção é escolhida[1]. O mapa de Karnaught da Figura 1.23 é gerado.

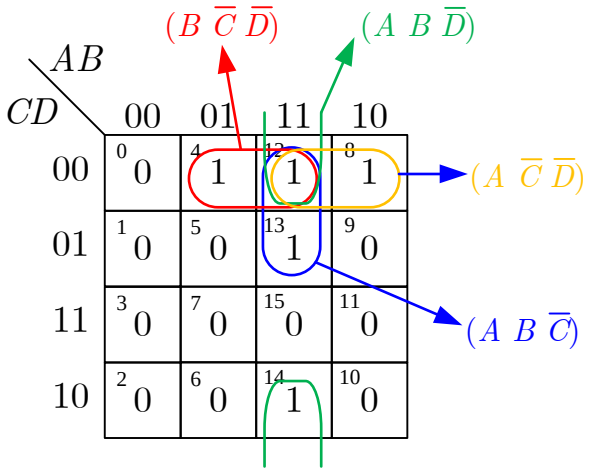

Figura 1.23: Mapa de Karnaught da solução do problema 2.

Logo, a expressão simplificada é dada por:

$$NS = (B\bar{C}\bar{D}) + (AB\bar{D}) + (A\bar{C}\bar{D}) + (AB\bar{C})$$

Passo 5: Elaboração da lógica em linguagem Ladder. As seguintes atribuições são feitas, de acordo com o painel de ligações do laboratório:

- $A \rightarrow$ botão Start (NA);
- $B \rightarrow$ botão Stop (NF - trocar B por $\bar{B}$ e vice-versa);
- $C \rightarrow$ chave CH1;
- $D \rightarrow$ chave CH2;
- $NS \rightarrow$ lâmpada L1;
- $LO \rightarrow$ lâmpada L2;

O diagrama Ladder resultante é apresentado na Figura 1.24.

A Figura 1.25 apresenta a tela da rotina programada no RSLogix 5000.

[1]Esta opção pode ficar complicada para elevado número de variáveis.

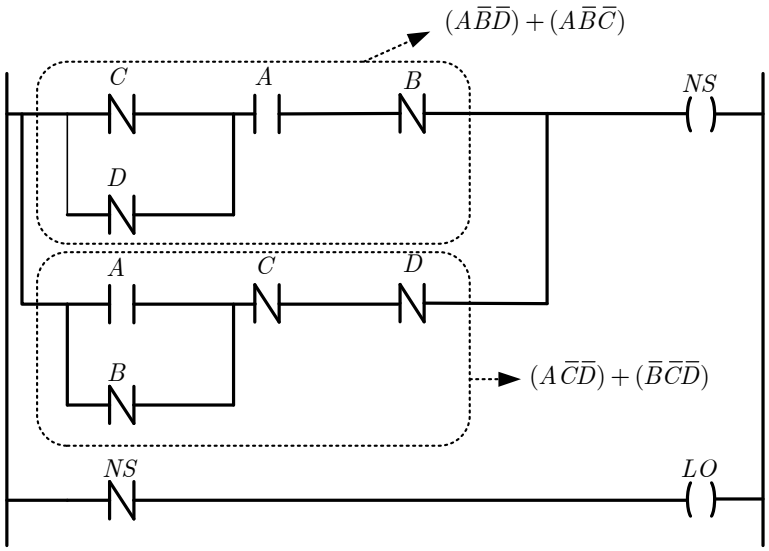

Figura 1.24: Diagrama Ladder da solução do problema 2.

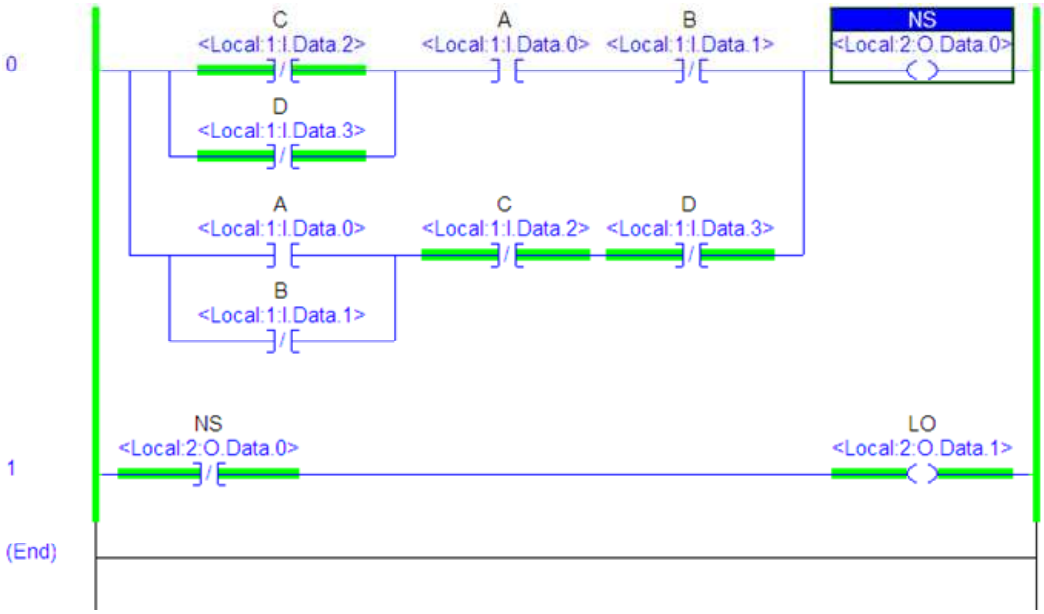

Figura 1.25: Diagrama Ladder da solução do problema 2 no RSLogix 5000.

Problema 3: Desenvolva uma aplicação em linguagem Ladder com as seguintes especificações:

- Quando BT1 (NA) é pressionado, as lâmpadas L1 e L2 ligam de forma alternada e sequenciada a cada 2 segundos.
- Quando BT2 (NF) é pressionado, as lâmpadas apagam e o ciclo é iniciado somente se BT1 for novamente pressionado.

`Solução`: Inicialmente, as seguintes ligações apresentadas na Figura 1.26 são efetuadas.

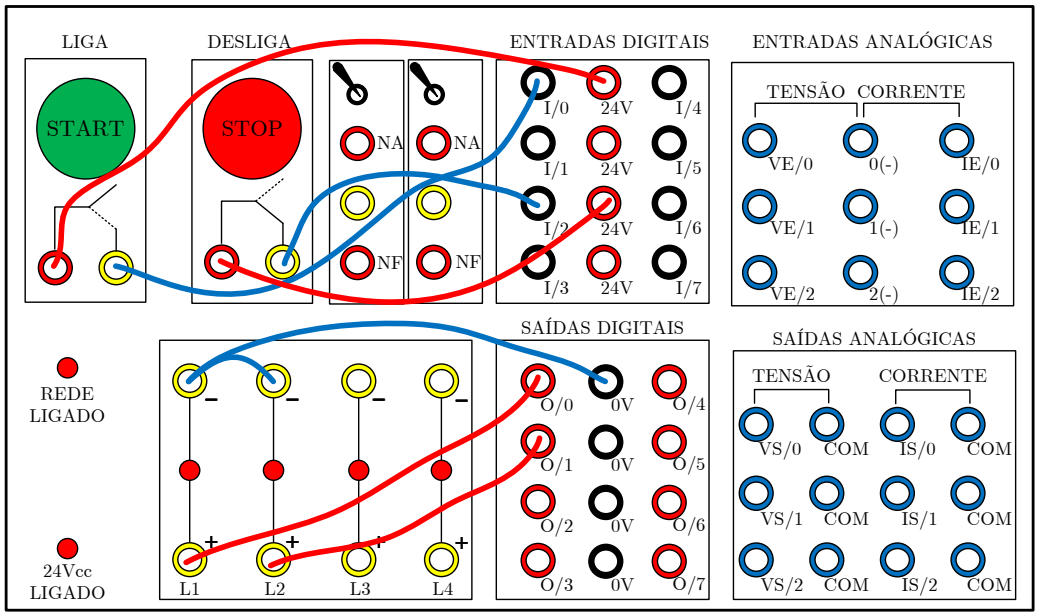

Figura 1.26: Ligações do exemplo.

O problema pode ser resolvido utilizando-se apenas um temporizador. No entanto, será considerado aqui o uso de dois temporizadores do tipo TON (Timer1 e Timer2), um para cada lâmpada, com período de contagem de 2 segundos em cada.

Nota-se que o problema é de natureza sequencial. Assim, o seguinte procedimento pode ser utilizado:

Passo 1: Identifique as entradas e saídas do sistema, os estados do sistema, e as transições de estado

As seguintes entradas e saídas são identificadas:

- Entradas:
  - $BT1 \rightarrow$ botão de habilitação (NA);
  - $BT2 \rightarrow$ botão de desabilitação (NF);
- Saídas:
  - $L1 \rightarrow$ lâmpada 1;
  - $L2 \rightarrow$ lâmpada 2;

Os seguintes estados são identificados.

- S0: L1 e L2 desligadas;
- S1: L1 ligada e L2 desligada;
- S2: L2 ligada e L1 desligada;

Transições:

- T1: S0 $\rightarrow$ S1 ($\mathrm{BT1} = 1$);
- T2: S1 $\rightarrow$ S2 ($\mathrm{Timer1.DN} = 1$);
- T3: S2 $\rightarrow$ S1 ($\mathrm{Timer2.DN} = 1$);

Passo 2: Crie um mapa de transição de estados, onde as transições ocorrem devido à mudança de nível lógico de variáveis booleanas, e também provocam mudanças de estado em variáveis booleanas. O mapa criado é apresentado na Figura 1.27.

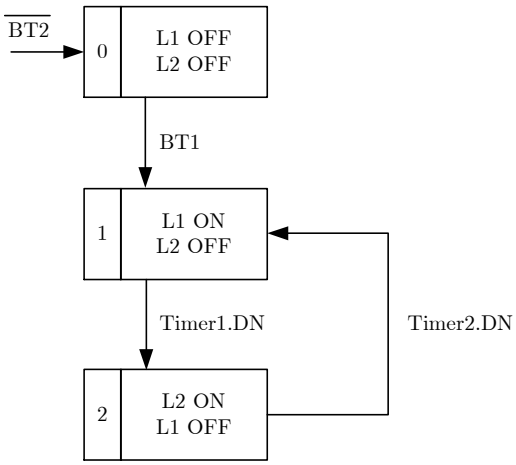

Figura 1.27: Mapa de transição de estados da solução do problema 3.

Note que se o botão B2 (NF) for pressionado, o sistema retorna ao estado inicial sem depender de qualquer outra condição.

Passo 3: Elaboração da lógica em linguagem Ladder a partir do mapa de transições. As seguintes atribuições são feitas, de acordo com o painel de ligações do laboratório:

- $BT1 \rightarrow$ botão Start (NA);
- $BT2 \rightarrow$ botão Stop (NF);
- $L1 \rightarrow$ lâmpada L1;
- $L2 \rightarrow$ lâmpada L2;

A passagem direta de mapa de transições de estado para Ladder ocorre quase que de forma direta, utilizando-se bobinas do tipo *latched* e *unlatched* para os estados, e efetuando o processo em duas etapas: **transições de estado** e **ações nos estados**.

A lógica desenvolvida está apresentada na Figura 1.28.

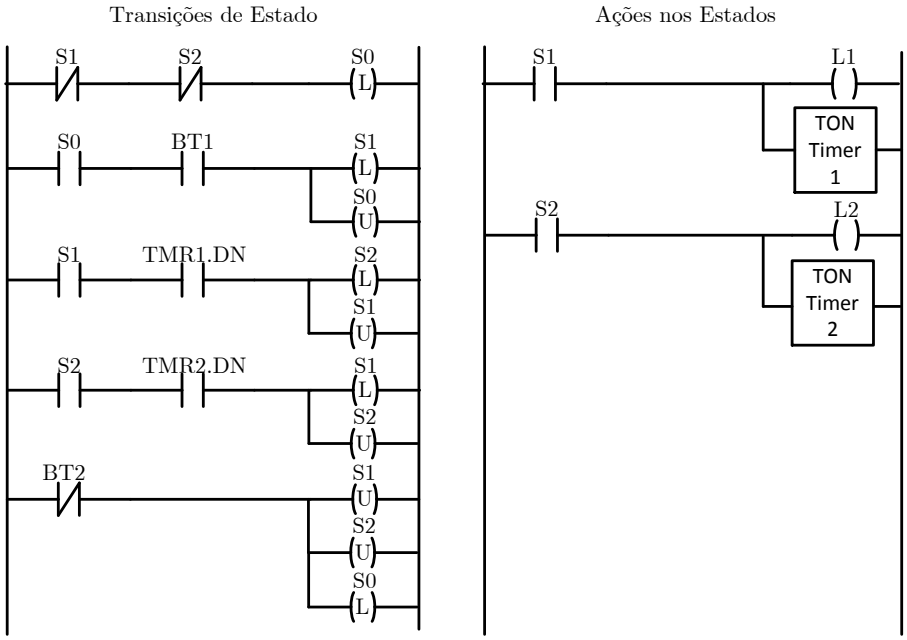

Figura 1.28: Lógica em Ladder da solução do problema 3.

A Figura 1.29 apresenta a tela da rotina programada no RSLogix 5000.

Outra forma de solucionar o problema utilizando contatos de saídas convencionais é apresentada na Figura 1.30, utilizando o conceito de selo. Tal solução não é imediata, principalmente para problemas com maiores dimensões.

A Figura 1.31 apresenta a tela da rotina programada no RSLogix 5000.

Tente resolver o problema utilizando apenas um temporizador!

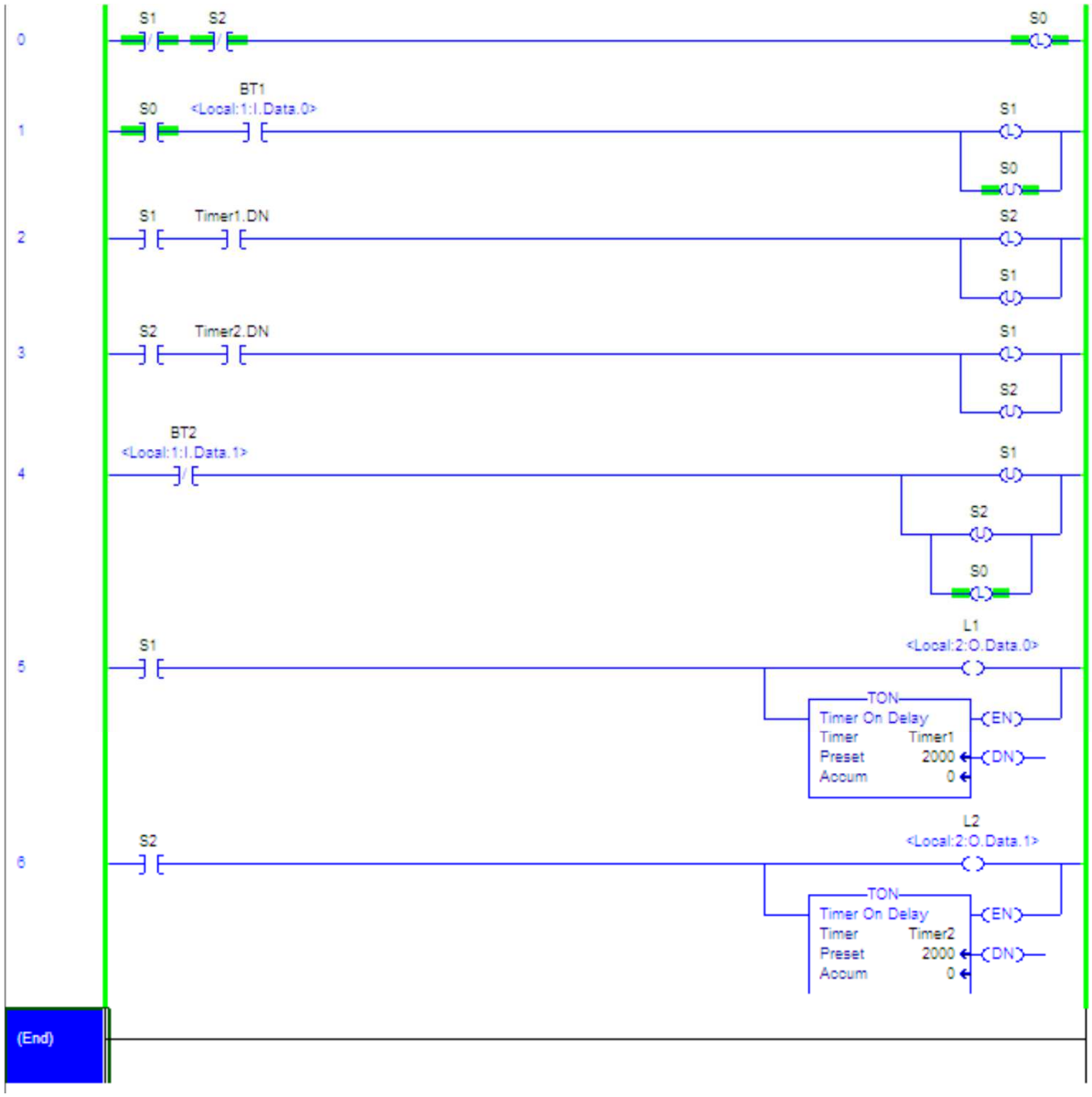

Figura 1.29: Lógica em Ladder da solução do problema 3.

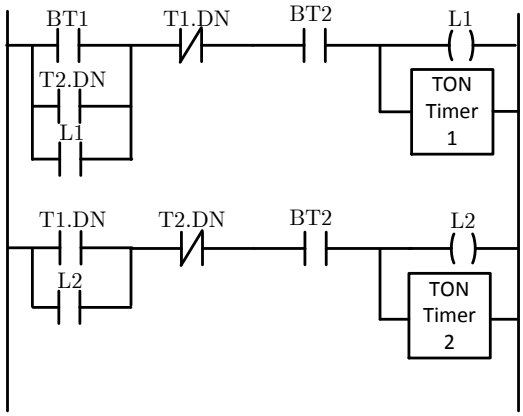

Figura 1.30: Lógica em Ladder da solução do problema 3 com contatos de saída convencionais.

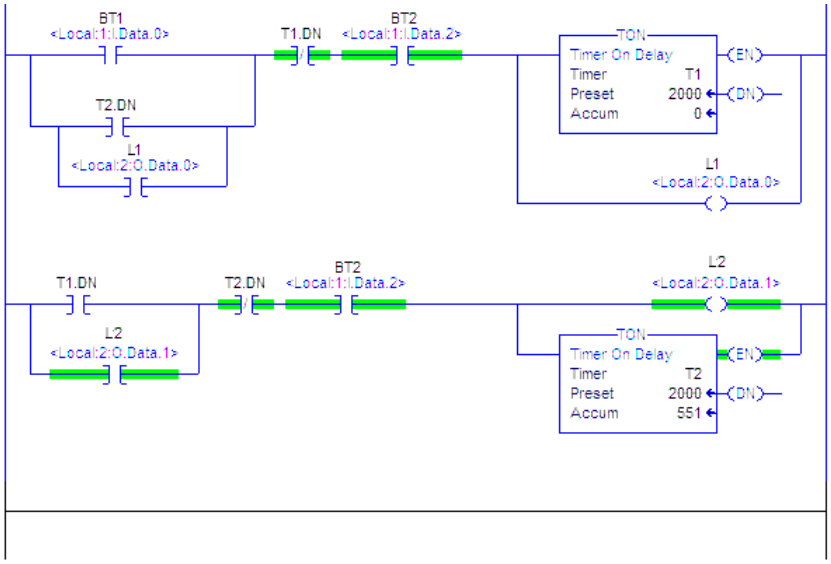

Figura 1.31: Lógica em Ladder da solução do problema 3 com contatos de saídas convencionais no RSLogix 5000.

1.5 Atividades

1.5.1 Exercícios simples

a) 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** (ou seja, que opera com dois sinais de atuação distintos), que avança mediante o acionamento de dois botões ($S1$ e $S2$) e retorna caso qualquer um dos botões seja desativado. Elabore um programa em linguagem Ladder que resolva este problema e teste no CLP.

b) Elaborar um programa Ladder para controlar dois relés (R1 e R2) de tal maneira que R1 pode atuar de forma independente e R2 só pode atuar se R1 estiver ligado, mas pode continuar ligado após o desligamento de R1. Os relés são ligados pelas botoeiras L1 e L2, e são desligados pelas botoeiras D1 e D2.

1.5.2 Tanque industrial

a) Um tanque industrial possui uma eletroválvula $V1$ que permite a entrada de líquido e uma $V2$ que permite sua 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 1.32 e as convenções do funcionamento do sistema estão apresentadas na Tabela 1.8. Quando o alarme A for acionado, deve-se fechar V1 e V2, mesmo com B pressionado.

Tabela 1.8: Tabela de Funcionamento do 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 $V2$

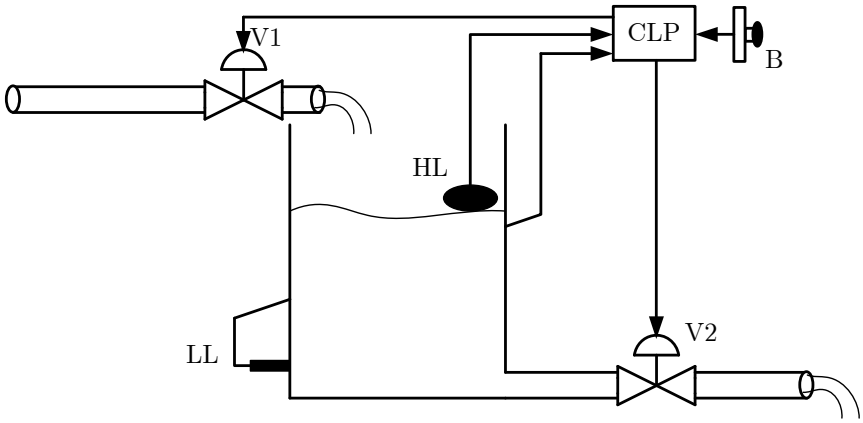

Figura 1.32: Esquema do tanque industrial.

b) Elaborar um programa Ladder de controle para um reservatório da Figura 1.33 composto de uma válvula de entrada V1, duas bombas (acionadas por M1 e M2), um alarme AL e quatro sensores de nível (A, B, C, D). As condições de funcionamento são as seguintes: se o nível for 'A', então fecha-se a válvula V1. Se o nível for inferior a 'B' então abre-se a válvula V1. Acima de 'B', M1 e M2 bombeiam. Abaixo de 'B', somente M1 bombeia. Abaixo de 'C' soa o alarme AL e a lampada L1. Em 'D', nenhuma das bombas deverá funcionar.

Elabore um diagrama Ladder simplificado para encher ou esvaziar o tanque industrial por meio de suas eletroválvulas.

1.5.3 Exercício usando contadores

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.

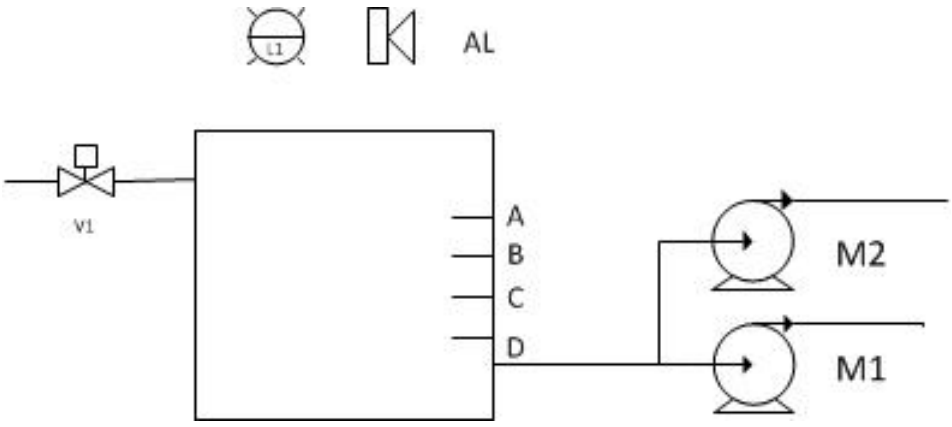

Figura 1.33: Diagrama do controle de nível no tanque.

b) Elabore um programa em Ladder para contar o número de caixas que passa pelo sensor S1 em uma esteira. Após 4 caixas, a lâmpada L1 deve ser acionada para que o fardo de caixas seja fechado. Neste ponto, o processo é interrompido para que o fardo seja transportado. O processo é reiniciado quanto o botão BT1 for pressionado. Quando o total de caixas que passa pelo sensor for igual a 20, o processo para e a lâmpada L2 é acionada para que o lote seja fechado. Tudo reinicia quando BT1 for pressionado. Se BT1 for pressionado acidentalmente no meio do processo, nada acontece. O botão BT2 é de emergência: quando pressionado, tudo para.

1.5.4 Exercícios usando temporizadores

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 ou 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 Tabela 1.9.

Tabela 1.9: Tabela de funcionamento do sistema de alarme.

Sinal	**Significado**
$B = 0$	Comando habilitar alarme
$B = 1$	Comando desabilitar 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 1.34, sendo que cada semáforo está posicionado numa das ruas. A sequência de acionamento de cada fase (amarelo, vermelho e verde) dos semáforos é mostrada na Tabela 1.10.

d) Resolva o item c) utilizando SFC.

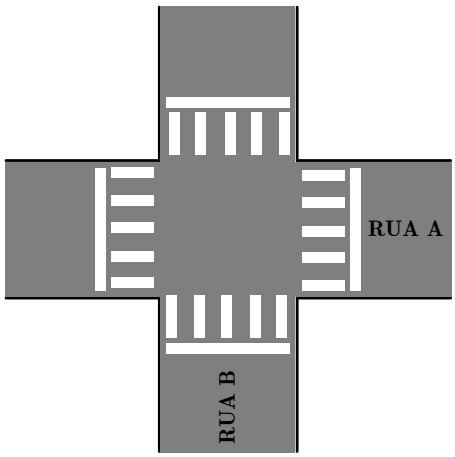

Figura 1.34: Cruzamento de ruas.

Tabela 1.10: Tabela de funcionamento do sistema de semáforos.

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.

1.5.5 Exercício 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 1.35. 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.

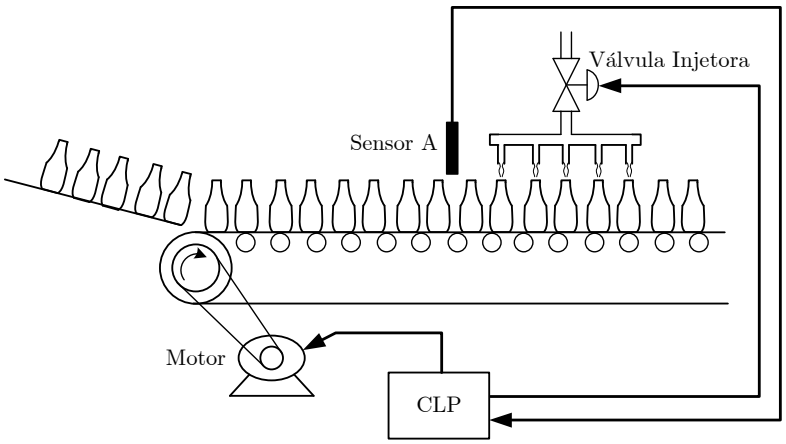

Figura 1.35: Engarrafadora.

b) Resolv ao item a) utilizando SFC.

1.6 Relatório

Um relatório desta experiência deverá ser entregue.

1.7 Problemas e dúvidas frequentes

a) **A comunicação USB não foi reconhecida?**

Espere o computador ligar para conectar o cabo USB

b) Há um erro no endereçamento dos pinos de entradas e/ou saída.

Por exemplo, para endereçar o pino 0 do módulo de entrada, verifique se o endereço está `Local:1:I.Data.0` ao invés de `Local:1:I.Data`.

Créditos

Esta experiência foi desenvolvida e/ou atualizada pelos seguintes professores:

- Ricardo Paulino Marques
- Bruno Augusto Angélico
- Fábio de Oliveira Fialho

Apêndice — Programação SFC

A linguagem SFC (*Sequential Function Chart*) tem mais a ver com uma técnica de programação do que com uma linguagem propriamente dita. É usada para modelar lógicas de controle baseadas na sequência temporal de eventos de processo [Prudente 2013]. Surgiu em 1977 na França, onde foi chamada de GRAFCET. Tem estreita relação com a rede Petri.

Um ciclo industrial é formado por passos e transições. Cada passo possui um determinado número de operações. A passagem de um passo para outro denomina-se transição. Uma transição ocorre quando certas condições são satisfeitas. A Figura 1.36 apresenta um exemplo de passos com transição.

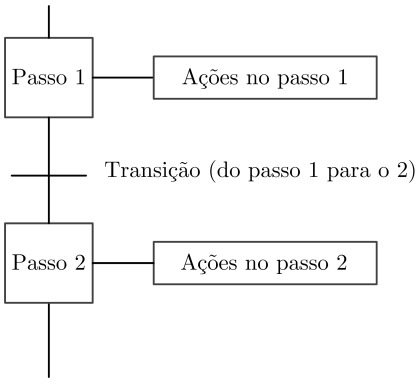

Figura 1.36: Esquema de um fragmento de código SFC.

Na Figura 1.37 são descritos alguns símbolos e gráficos, segundo a norma IEC 60848 (*GRAFCET specification language for sequential function charts*).

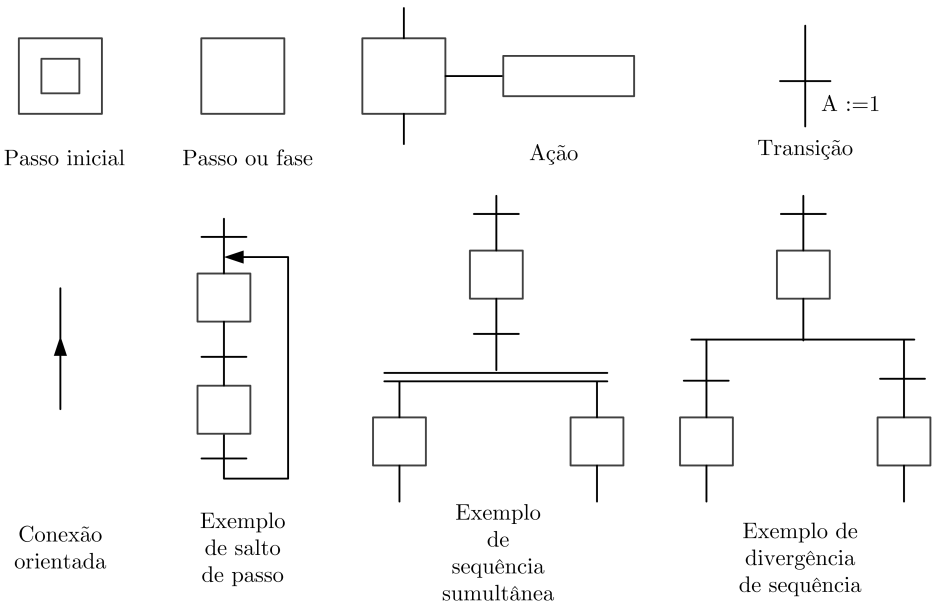

Figura 1.37: Alguns símbolos SFC.

Para elaboração de um programa em SFC, algumas regras devem ser respeitadas [Prudente 2013]:

- deve-se ter pelo menos um passo inicial;

- deve-se ter alternância entre passo e transição;
- a superação de uma transição representa a finalização das ações do passo anterior e a ativação das ações do(s) passo(s) seguinte(s).

O exemplo a seguir ilustra um programa SFC no RSLogix 5000 [Allen-Bradley 2016].

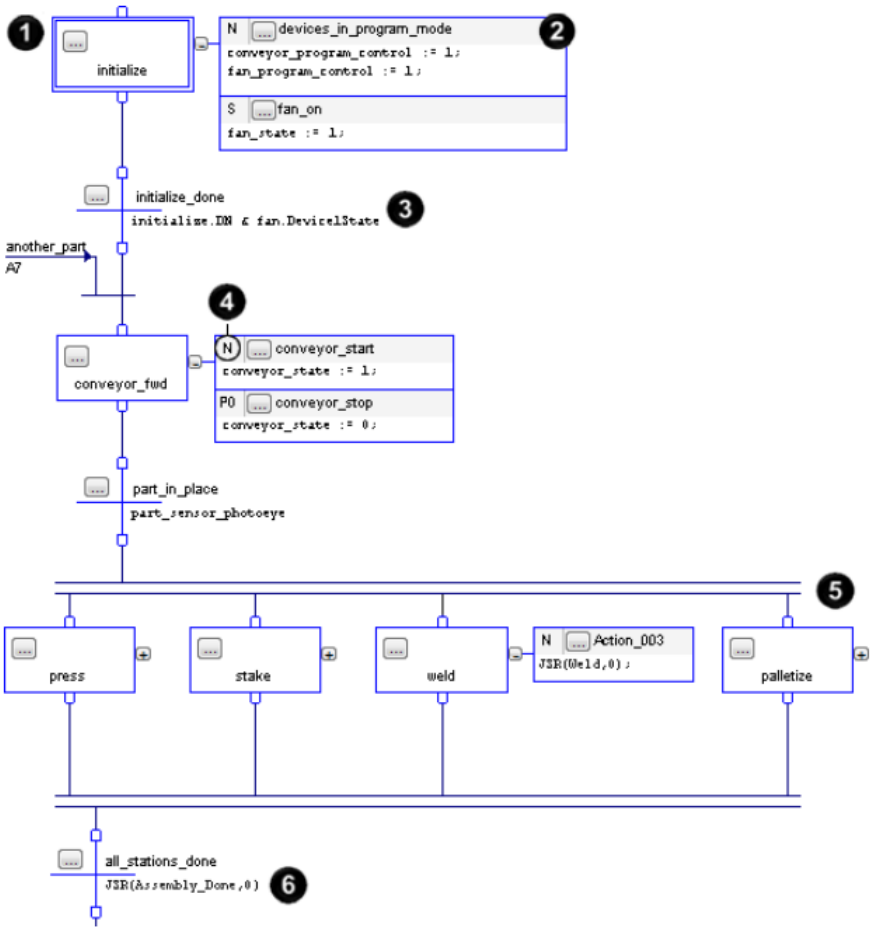

Figura 1.38: Exemplo de programa SFC. Fonte: [Allen-Bradley 2016].

Os itens da Figura 1.38 são apresentados a seguir:

1. O **passo** representa um estado e contém as ações a serem executadas em um dado instante;
2. Uma **ação** é uma das funções executadas por um passo;
3. Uma **transição** é uma condição VERDADEIRA ou FALSA que determina a passagem de um passo a outro;
4. Um **qualificador** determina quando a ação começa e termina;
5. Um **ramo simultâneo** executa mais de um passo de uma vez;
6. A **instruçao JSR** chama uma sub-rotina;

Criação de Programa em SFC

Ao iniciar uma nova rotina no RSLogix 5000, automaticamente o programa escolhe a linguagem Ladder. Para criar uma rotina em SFC, a *main routine* em Ladder deve ser apagada e, em seguida, uma rotina SFC deve ser criada, conforme as Figuras 1.39 e 1.40.

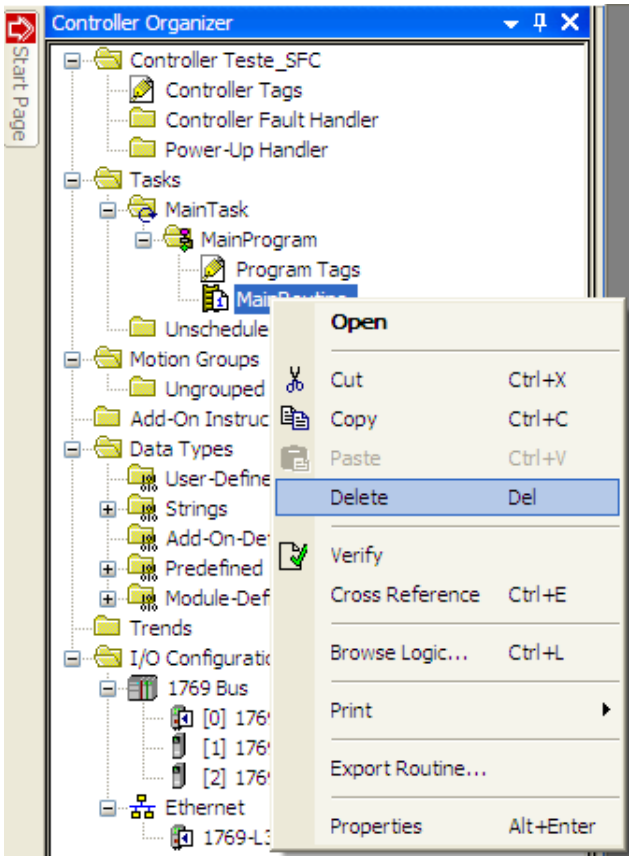

Figura 1.39: Apagar rotina Ladder.

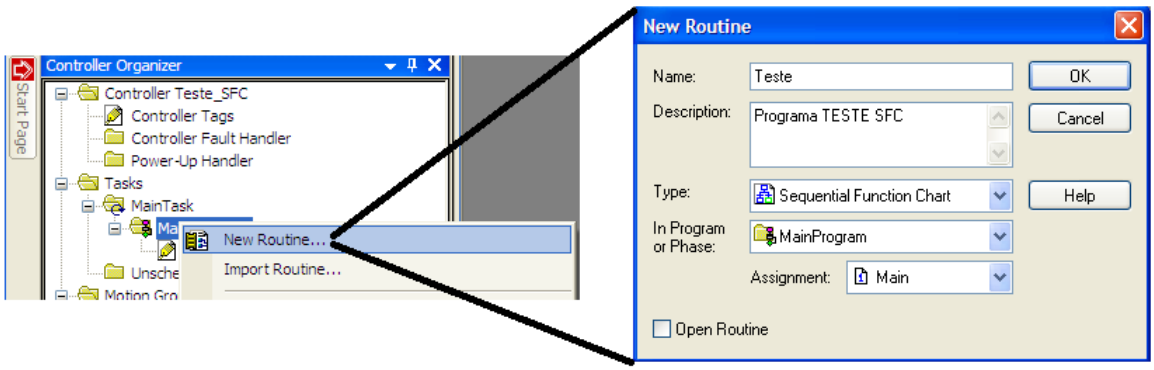

Figura 1.40: Criar rotina SFC.

O *template* inicial do programa SFC é apresentado na Figura 1.41.

Para acrescentar passos e transições, acesse a barra de menu superior do RSLogix 5000. Para acrescentar ações a um passo, clique com o botão direito no passo e selecione *Add Action*, ou clique no campo

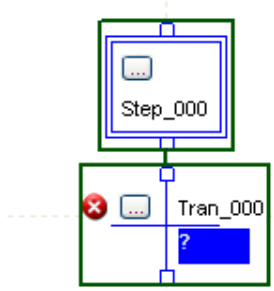

Figura 1.41: Template inicial da rotina em SFC.

correspondente no menu superior.

Para programar uma ação, o RSLogix 5000 só aceita a linguagem de Texto Estruturado (ST). Por exemplo, `L1:=0; L2:=1;`. Para escrever o código, clique no símbolo "?", conforme a Figura 1.42.

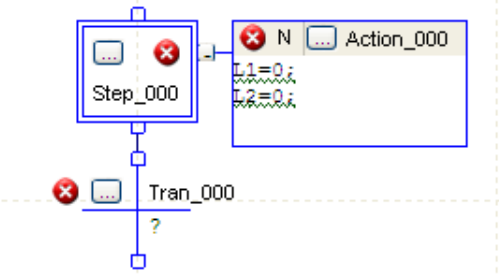

Figura 1.42: Inserção de código ST na ação.

Para definir uma *tag*, basta clicar com o botão direito do mouse sobre a variável e em *New Tag*, como ilustrado na Figura 1.43. As *tags* também podem ser definidas antes da edição do programa pela opção *Edit Tags* na janela principal do RSLogix 5000.

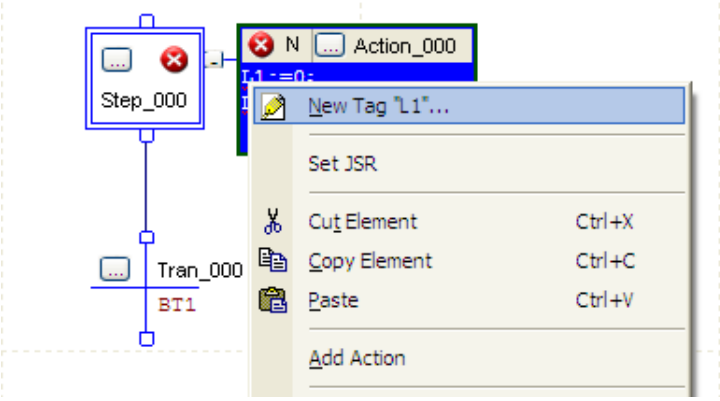

Figura 1.43: Criando *tags*.

Para definir o qualificador a ser utilizado na ação, clique no símbolo "...". Os três principais qualificadores são:

- N: executa continuamente enquanto o passo estiver ativo;

- P1: executa uma única vez quando o passo é ativado;
- P0: executa uma única vez quando o passo é desativado.

Para definir a condição de una transição, clique no símbolo "?" e adicione o código ST.

A conexão entre passos e transições é feita posicionando-se o mouse sobre o ponto de entrada ou saída do elemento até o aparecimento de um círculo verde. Clique sobre o mesmo e então arraste-o até o ponto de conexão desejado.

Um passo possui algumas variáveis internas que podem ser utilizadas no código, por exemplo, como temporizadores ou contadores. As principais são:

- `Step_001.X` : indica se o passo 001 está ativo.
- `Step_001.Count`: indica quantas vezes o passo 001 foi ativado.
- `Step_001.T`: determina por quanto tempo o passo está ativo, em milisegundos. Tem valor indefinido quando o passo não estiver ativado.

Problemas com Solução

Problema 4: Refaça o **Problema 3**, mas agora em SFC:

- Quando BT1 (NA) é pressionado, as lâmpadas L1 e L2 ligam de forma alternada e sequenciada a cada 2 segundos.
- Quando BT2 (NF) é pressionado, as lâmpadas apagam e o ciclo é iniciado somente se BT1 for novamente pressionado.

Solução: O código final é apresentado na Figura 1.44.

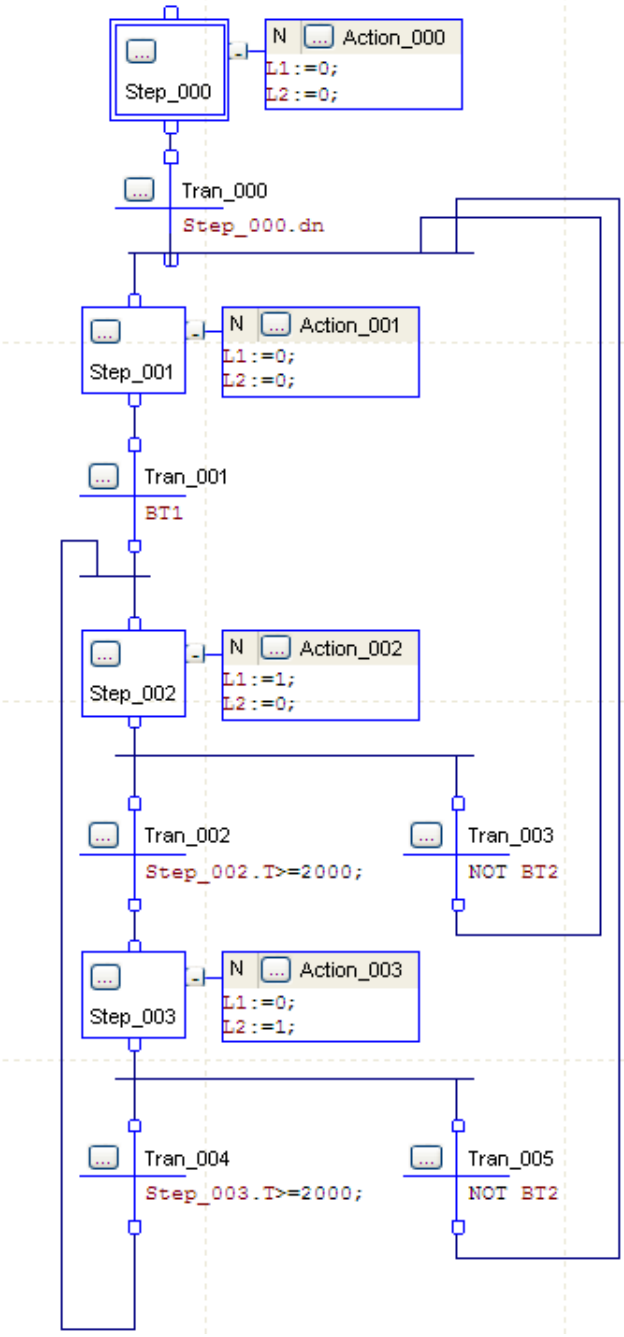

Figura 1.44: Solução do Problema 4.

Outra forma de solução é apresentada na Figura 1.45.

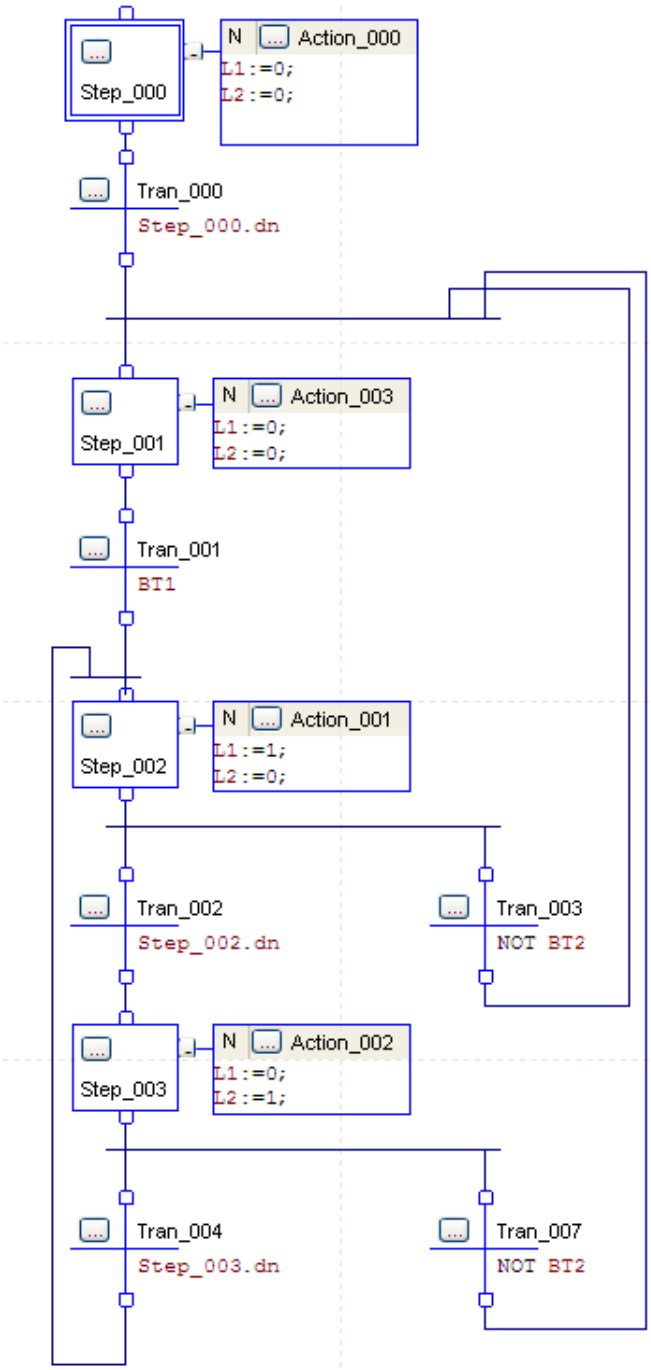

Figura 1.45: Solução alternativa do Problema 4.

Nesta solução alternativa, para definir o tempo de 2 segundos em cada estado, utilizou-se o valor de *preset* do temporizador do passo. Ele é configurado clicando-se no símbolo "..." do passo, conforme a Figura 1.46. Note que a transição é efetuada quando a condição `Step_002.dn` é satisfeita.

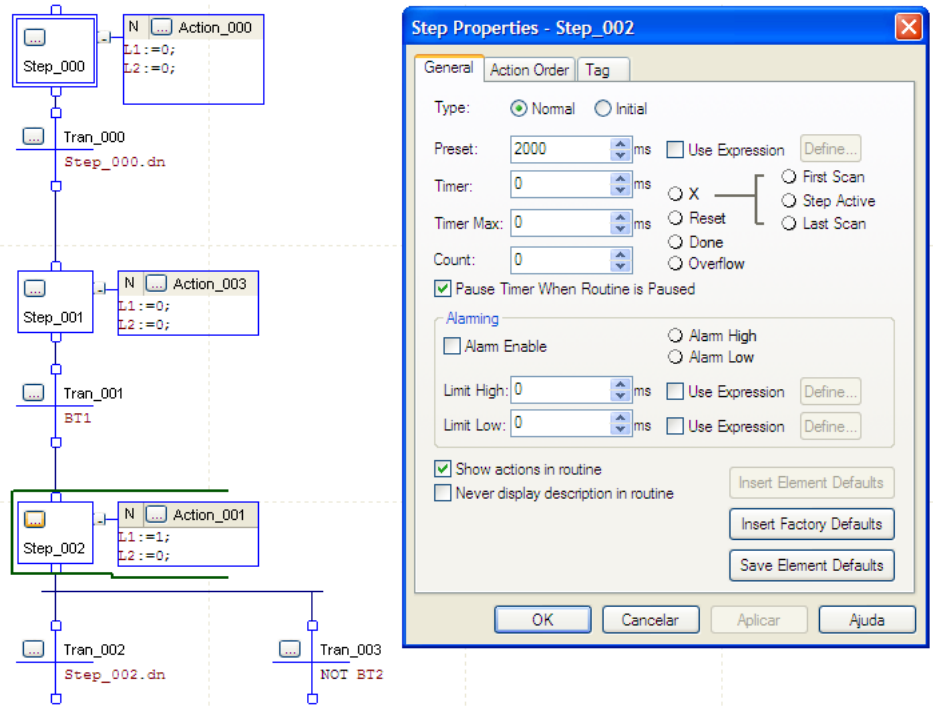

Figura 1.46: Preset do timer para a solução alternativa do Problema 4.

Problema 5: Ao pressionar um botão NA (BT1), uma lâmpada (L1) deve piscar 5 vezes com um período de 1 segundo. Após isso, o ciclo pode ser iniciado ao clicar o botão novamente.

Solução: O código final é apresentado na Figura 1.47.

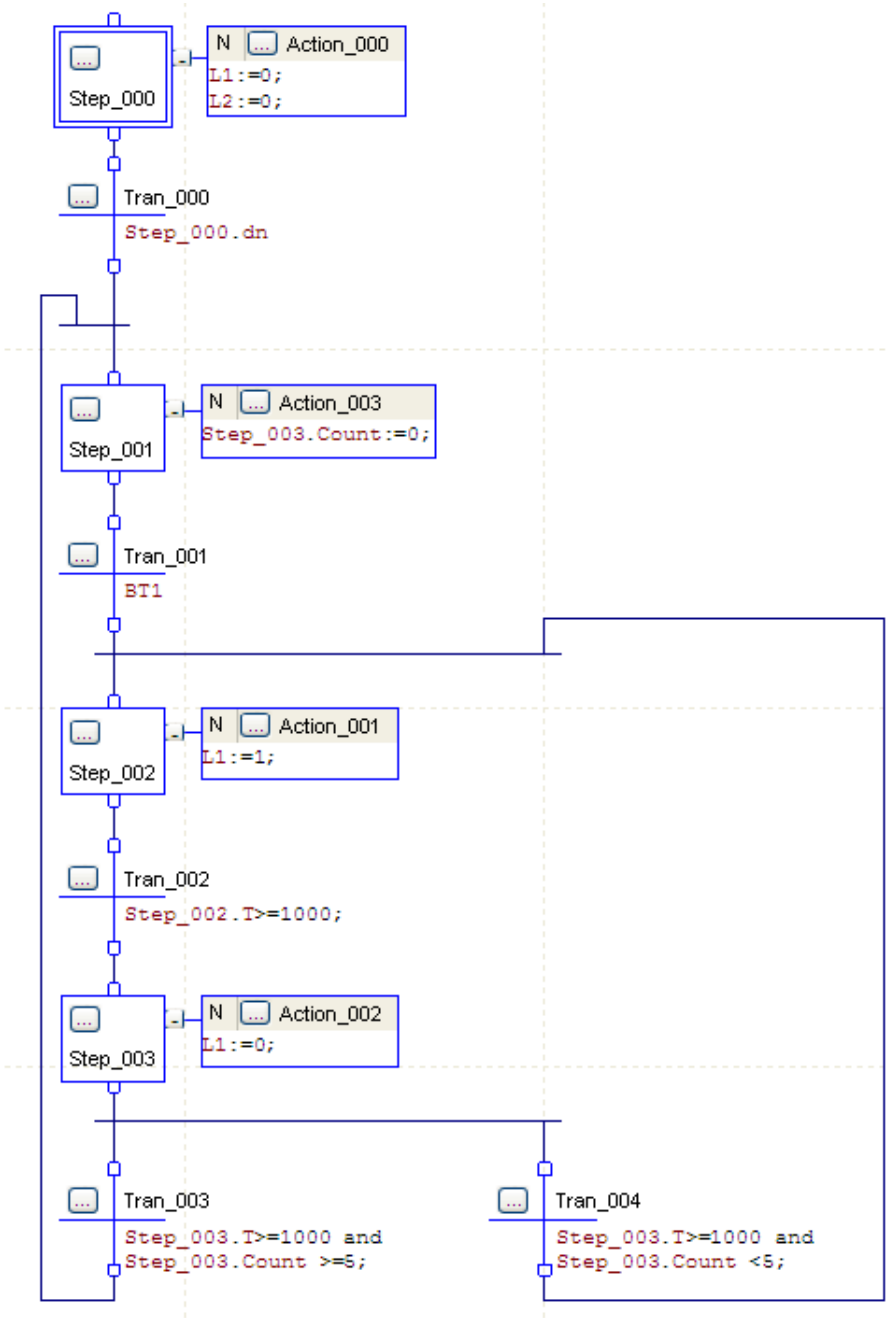

Figura 1.47: Solução do Problema 5.

Uma solução alternativa do Problema 5 é apresentada na Figura 1.48. Note que ao `Step_002` foram atribuídas 2 ações com qualificadores diferentes. Na segunda ação, utilizou-se o qualificador P1 para incrementar um contador. Nesta solução, utilizou-se o *preset* dos *timers* para temporização, como na solução alternativa do Problema 4.

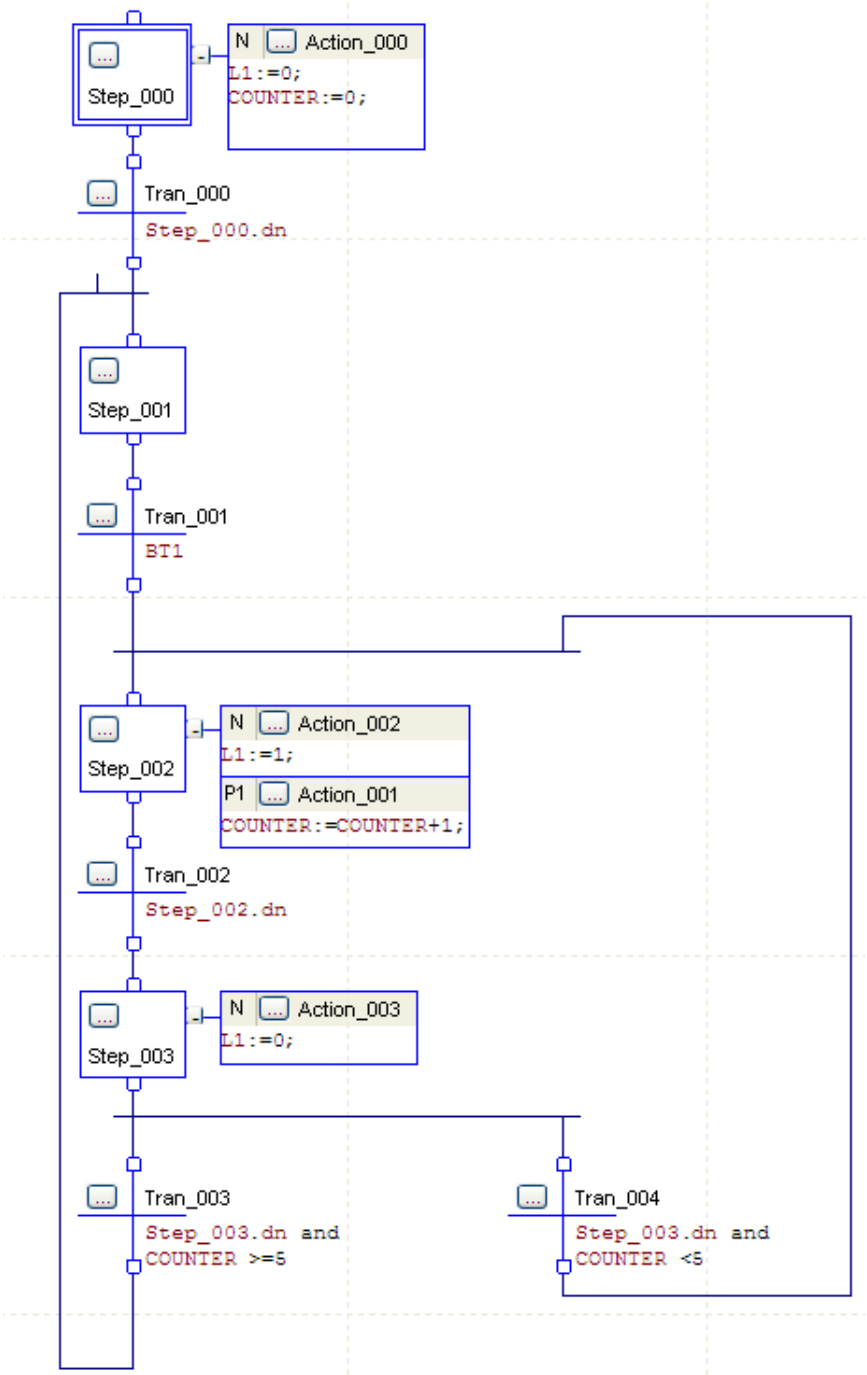

Figura 1.48: Solução alternativa do Problema 5.

Créditos

Esta experiência foi desenvolvida e/ou atualizada pelos seguintes professores:

- Bruno Augusto Angélico

2 Dispositivos Pneumáticos

2.1 Objetivo

O laboratório dispõe de diversos dispositivos pneumáticos e outros que podem ser utilizados em conjunto com os CLPs. O objetivo desta aula é implementar algumas aplicações utilizando atuadores pneumáticos, sensores de fim de curso, válvulas solenóides e o CLP.

2.2 Cuidados com a Segurança

- **Antes de pressurizar o sistema verifique se as mangueiras estão bem presas aos conectores. Mangueiras pressurizadas soltas podem chicotear e causar acidentes graves.**
- **Não mexa nas conexões das mangueiras com o sistema pressurizado.**
- **Mantenha suas mãos e objetos a uma distância segura do curso de acionamento dos pistões.**

2.3 Introdução teórica

A seguir serão descritos alguns sensores componentes básicos de pneumáticas presentes no laboratório.

- **Unidade de Conservação**: tem a finalidade de purificar o ar comprimido, ajustar uma pressão constante do ar e, em alguns casos, acrescentar uma fina neblina de óleo ao ar comprimido, para fins de lubrificação. Aumenta consideravelmente a segurança de funcionamento dos equipamentos pneumáticos. A Figura 2.1 ilustra uma unidade de conservação sem lubrificador, juntamente com seu diagrama.

A unidade de conservação disponível no laboratório possui as seguintes características:

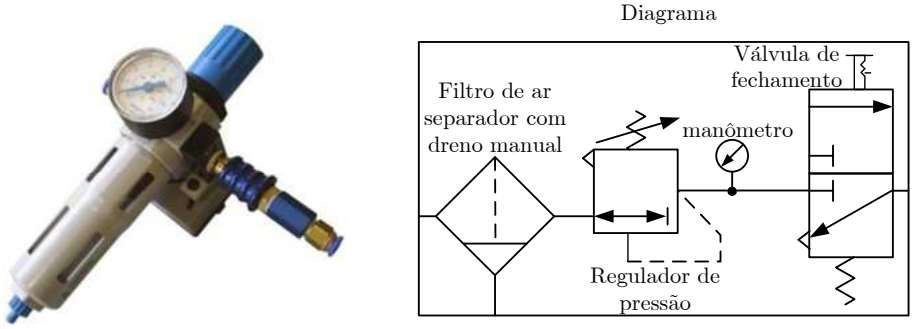

Figura 2.1: Unidade de conservação.

- conjunto de filtro, regulador de pressão, manômetro e válvula de fechamento;
- dreno manual;
- pressão de operação: de 0 a 12 bar;
- vazão nominal: 750 lpm;
- escala métrica: de 0 a 16 bar; escala inglesa: de 0 a 220 psi;
- válvula deslizante de acionamento manual biestável.

• **Bloco Distribuidor**: funciona como um demultiplexador de ar comprimido. A entrada de ar comprimido é redistribuída em várias saídas. A Figura 2.8 ilustra um bloco distribuidor e seu diagrama.

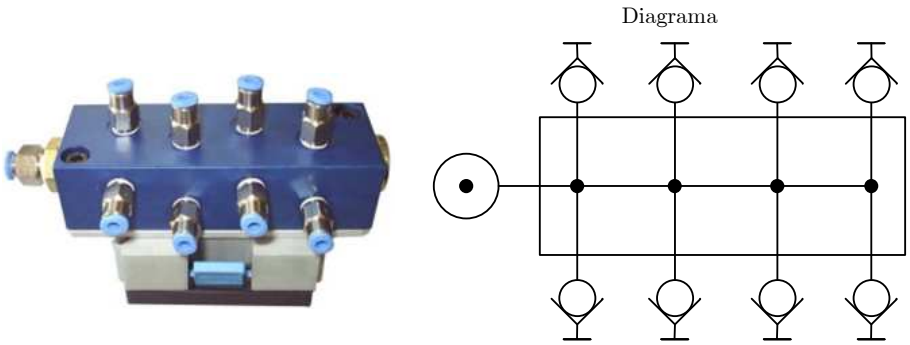

Figura 2.2: Bloco distribuidor.

O bloco distribuidor disponível no laboratório possui uma entrada e oito saídas de ar comprimido.

• **Cilindro de Simples Ação**: A Figura 2.3 apresenta um exemplo de cilindro de simples ação com avanço pneumático e retorno por mola.

A pressão máxima de trabalho do modelo presente no laboratório é de 10 bar.

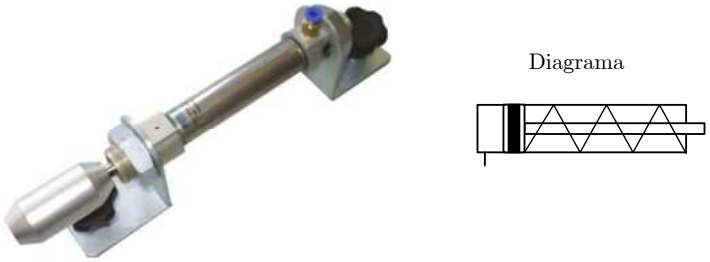

Figura 2.3: Cilindro de simples ação.

- **Eletroválvula direcional de 3/2 vias NF**: Válvula acionada por servocomando elétrico e piloto. É importante frisar que o acionamento por servocomando é indireto, ou seja, não é o solenóide quem aciona diretamente o carretel da válvula; ele epenas abre uma passagem para o ar comprimido (piloto) que aciona o carretel e muda a posição da válvula. Possui 3 vias de trabalho e 2 posições. O contato é normalmente fechado. A Figura 2.4 apresenta um exemplo desta válvula e seu diagrama.

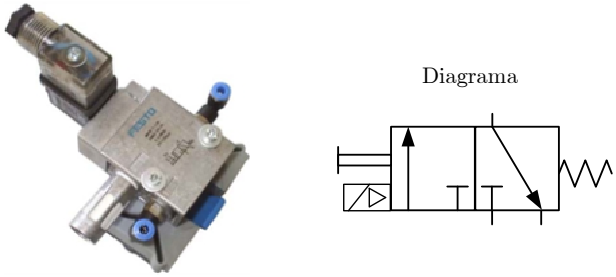

Figura 2.4: Eletroválvula direcional de 3/2 vias NF.

Algumas características adicionais:

- acionamento por servocomando, simples piloto e solenóide de 24 Vcc.
- retorno por mola;
- possibilidade de acionamento manual de emergência;
- pressão de operação: de 1,5 a 8 bar;
- vazão nominal: 500 lpm.

- **Cilindro de Dupla Ação**: A Figura 2.5 apresenta um exemplo de cilindro de dupla ação com avanço e retorno pneumáticos.

A pressão máxima de trabalho do modelo presente no laboratório é de 10 bar. Além disso, o modelo possui êmbolo magnético para detecção por sensores sem contato físico,

- **Eletroválvula Direcional de 5/2 vias com mola de reposição**: Válvula acionada por servocomando elétrico e piloto. Possui 5 vias de trabalho e 2 posições. A Figura 2.6 apresenta um exemplo desta válvula e seu diagrama.

Figura 2.5: Cilindro de dupla ação.

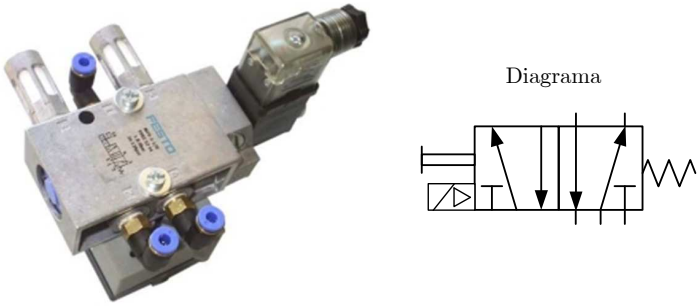

Figura 2.6: Eletroválvula direcional de 5/2 vias com retorno por mola.

As características adicionais são as mesmas da eletroválvula anterior.

- **Eletroválvula Direcional de 5/2 do tipo memória**: Válvula acionada por servocomando elétrico e piloto. Possui 5 vias de trabalho e 2 posições. Também chamada de válvula de impulso. Para sua comutação, basta emitir um pulso de pilotagem, não sendo necessário mantê-lo após a mudança de posição. A Figura 2.7 apresenta um exemplo desta válvula e seu diagrama.

Figura 2.7: Eletroválvula direcional de 5/2 vias com duplo acionamento e memória.

- **Chave de Fim de Curso**: chave de contato mecânico utilizada como sensor de fim de curso por contato físico. A Figura 2.8 ilustra uma chave de fim de curso e seu diagrama.

Algumas características da chave de fim de curso disponível no laboratório:

- reposicionado por mola;
- corrente: 5 A.

- **Sensor Capacitivo**: Os sensores de proximidade capacitivos registram a presença de qualquer tipo de material. A distância de detecção varia de 0 a 20 mm, dependendo da massa do material a ser

Figura 2.8: Chave fim de curso.

detectado e das características determinadas pelo fabricante. A Figura 2.9 apresenta um sensor capacitivo e seu diagrama.

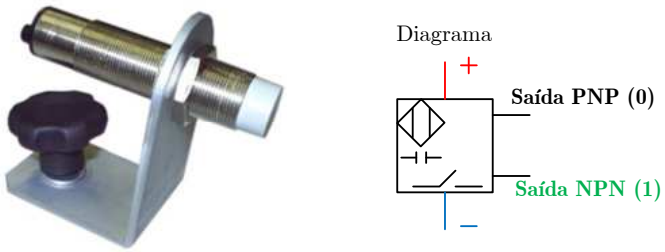

Figura 2.9: Sensor de proximidade capacitivo.

O sensor capacitivo presente no laboratório possui as seguintes características:

- distância de sensoriamento: 20 mm;
- tensão de alimentação: 10 a 30 Vcc;
- frequência máxima: 100 Hz;
- sinal de saída: 24 Vcc PNP;
- positivo: **vermelho**, negativo: **azul**, saída PNP: **preto**.

• **Sensor Magnético**: Os sensores de proximidade magnéticos detectam apenas a presença de materiais metálicos e magnéticos. São utilizados com maior frequência em máquinas e equipamentos pneumáticos e são montados diretamente sobre as camisas dos cilindros dotados de êmbolos magnéticos. A Figura 2.10 apresenta um sensor magnético e seu diagrama.

Algumas características do sensor disponível no laboratório:

- sensoriamento de êmbolos magnéticos de cilindros, sem contato físico;
- tensão de comutação: de 12 a 27 Vcc;

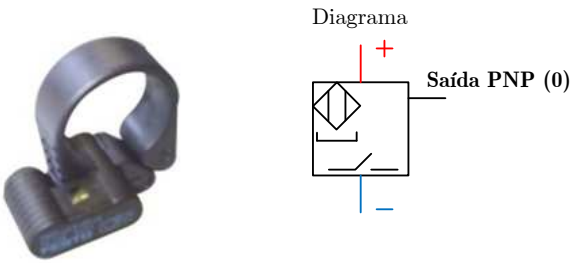

Figura 2.10: Sensor de proximidade magnético.

- freqüência máxima: 800 Hz;
- sinal de saída: 12 a 27 Vcc PNP;
- positivo: **vermelho**, negativo: **azul**, saída PNP: **preto**.

• **Sensor Indutivo**: Os sensores de proximidade indutivos são capazes de detectar apenas materiais metálicos. A Figura 2.11 apresenta um sensor indutivo e seu diagrama.

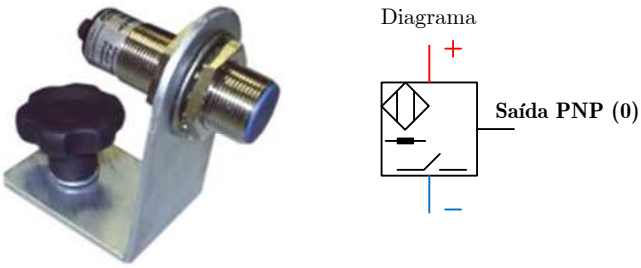

Figura 2.11: Sensor de proximidade indutivo.

O sensor indutivo presente no laboratório possui as seguintes características:

- distância de sensoriamento: 5 mm;
- tensão de alimentação: 10 a 30 Vcc;
- frequência máxima: 800 Hz;
- sinal de saída: 24 Vcc PNP;
- positivo: **vermelho**, negativo: **azul**, saída PNP: **preto**.

2.4 Atividades práticas

2.4.1 Familiarização com os equipamentos pneumáticos do laboratório

Teste os equipamentos na seguinte sequência:

- **Unidade de conservação.**

  Leia atentamente a descrição de funcionamento do equipamento. Em seguida acione a unidade abrindo o registro da tubulação de distribuição de ar comprimido do laboratório e verifique que a pressão está calibrada entre 4 e 5 bar;

- **Bloco distribuidor.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Cilindro de simples ação.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Eletroválvula direcional de 3/2 vias NF.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Cilindro de dupla ação.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Eletroválvula direcional de 5/2 vias com retorno por mola.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Eletroválvula direcional de 5/2 do tipo memória.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Chave de fim de curso.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Sensor de proximidade capacitivo.**

  Leia atentamente a descrição de funcionamento do equipamento.

- **Sensor de proximidade magnético.**

  Leia atentamente a descrição de funcionamento do equipamento.

2.4.2 Cancela em linha ferroviária

A cancela possui dois sensores de presença espaçados de 100m, conforme a Figura 2.12. O acionamento da cancela é feito através de um pistão pneumático. Considere que os trens podem vir de ambos os sentidos e que podem (ou não) ter mais de 100m de comprimento. Considere também as seguintes características funcionais do sistema.

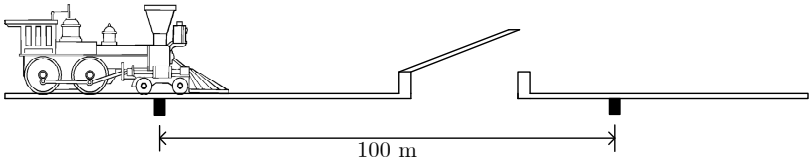

Figura 2.12: Cancela em linha ferroviária.

i) Caso haja problemas com o pistão e a cancela não seja acionada (um sensor de fim de curso verifica essa situação), um alarme luminoso deve ser acionado.

ii) Caso demore mais de 30s para um trem passar pelo segundo sensor após ter passado pelo primeiro, o mesmo alarme deve ser acionado.

Implemente um programa em Ladder para o sistema e teste no CLP em duas etapas.

a) Inicialmente assuma que os trens podem vir apenas de um sentido.

b) Altere o programa para considerar os dois sentidos de deslocamento dos trens.

2.4.3 Porta de um vagão do metrô

A porta de um vagão do metrô é mostrada na Figura 2.13. Dois pistões pneumáticos (Pistões A e B) são empregados para abrir e fechar a porta automaticamente. A presença de pessoas no curso das portas é verificada por sensores instalados em cada lado da porta (Sensores A e B).

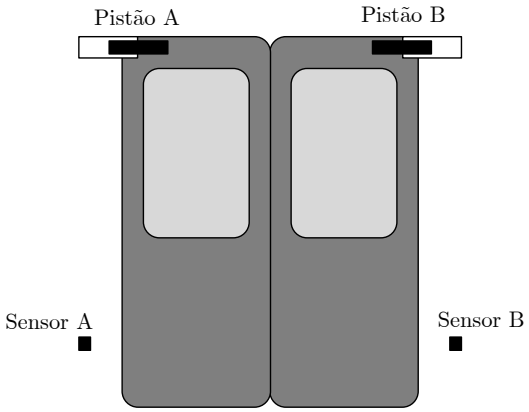

Figura 2.13: Porta de vagão do metrô.

Os requisitos funcionais do sistema de controle das portas são esquematizados a seguir.

i) A porta é aberta automaticamente quando o trem chega à estação, ativando um sensor de contato sobre a linha férrea.

ii) A porta permanece aberta por 30s, independente de haver ou não pessoas entrando e saindo pela porta. Considera-se que haja a presença de pessoas na porta se qualquer um dos Sensores A ou B for ativado.

iii) Após os 30s iniciais, a porta fecha caso não haja pessoas, ou permanece aberta por mais 10s. Se nesse período ninguém passar pela porta, ela fecha, e caso contrário permanece aberta por mais 10s e assim por diante.

iv) Caso a porta tenha permanecido aberta por três ciclos de 10s, totalizando 60s aberta, um alarme luminoso deve ser acionado para que o pessoal de segurança do metrô verifique a situação, por exemplo desobstruindo a porta ou limitando a entrada de passageiros. Note que mesmo com o acionamento do alarme, o algoritmo do item anterior deve continuar sua execução.

Utilize o CLP do laboratório para implementar o sistema em duas etapas.

a) Conforme acima.

b) Incluindo sensores nos pistões de cada porta para verificar se quando a porta é acionada ela efetivamente fecha. Em caso de falha o comando de abrir deve ser enviado à porta e um alarme luminoso deve ser ativado. Inclua um botão de ”Acknowledge”de falha das portas.

2.4.4 Sistema de eclusas do Canal do Panamá

O canal do Panamá (2.14) possui um sistema de eclusas para possibilitar a navegação sobre o istmo do Panamá, entre os Oceanos Atlântico e Pacífico.

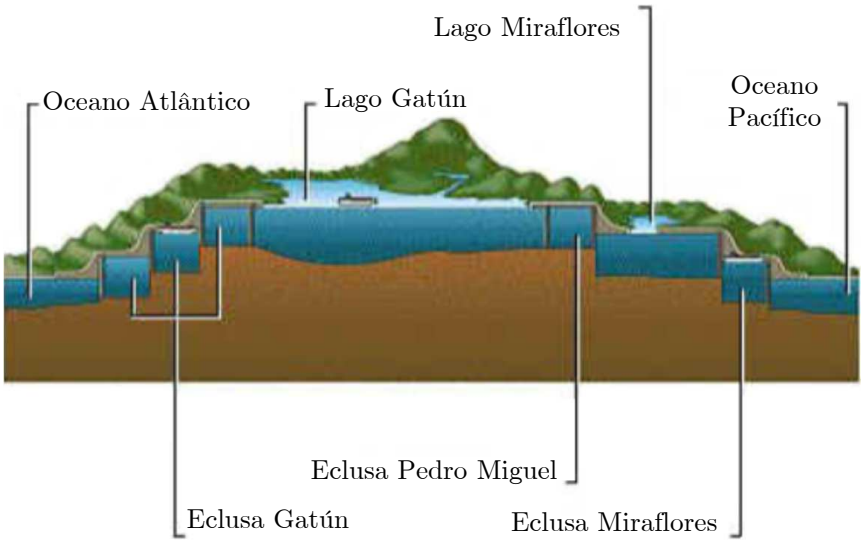

Figura 2.14: Sistema de eclusas do Canal do Panamá.

A operação simplificada de um sistema de automação da operação de uma eclusa é apresentada a seguir.

i) Quando um barco chega à entrada de uma eclusa, esta deve ser aberta caso não esteja ocupada ou em processo de enchimento ou esvaziamento. Considere que haja sensores que detectam a chegada de um barco assim como a presença de um barco na eclusa.

ii) Quando o barco entra a eclusa deve ser fechada.

iii) A eclusa deve encher até atingir o nível de saída. Considere que um sensor de nível avise quando este limite é atingido.

iv) A saída da eclusa deve ser aberta para o barco sair.

v) Após a saída do barco a saída deve ser fechada e a eclusa esvaziada até o nível de entrada, quando o ciclo recomeça.

Utilize esta descrição para obter os requisitos funcionais de um programa em Ladder que resolva o problema.

Considere que as entradas e saídas das eclusas são acionadas através de pistões pneumáticos e que há sensores que detectam a presença de embarcações na entrada, na saída e dentro da eclusa.

Elabore um programa em Ladder para implementação deste sistema em duas etapas.

a) Conforme acima.

b) Considerando que barcos podem chegar de ambos os sentidos. Neste caso, um sistema de intertravamento de chegada de barcos deve ser adicionalmente implementado.

2.5 Relatório

Um relatório desta experiência deverá ser entregue.

2.6 Problemas e dúvidas frequentes

Ainda não elencados problemas e dúvidas frequentes desta experiência.

Créditos

Esta experiência foi desenvolvida e/ou atualizada pelos seguintes professores:

- Ricardo Paulino Marques
- Bruno Augusto Angélico

3 Controle nebuloso

3.1 Objetivo

Esta experiência tem por objetivo a familiarização com a técnica de Controle Nebuloso (Fuzzy Control, em inglês). Para isso será construído um controlador de posição para o servomecanismo do laboratório.

3.2 Introdução teórica

Os fundamentos da teoria de conjuntos nebulosos foram apresentados pela primeira vez em 1965 pelo Prof. L.A. Zadeh no artigo [Zadeh 1965]. Essa teoria permite que informação imprecisa, qualitativa, possa ser expressa e manipulada através de um formalismo matemático. Como seu nome sugere, ela apresenta uma generalização do conceito tradicional de conjunto [Tong 1977].

O nosso interesse aqui é apresentar algumas noções básicas do assunto com a perspectiva de utilização em sistemas de controle. Contudo, são inúmeras as áreas atualmente em que tal teoria tem sido aplicada, dentre as quais podem ser citadas: reconhecimento de imagem, reconhecimentos de voz, sistemas especialistas (incluindo-se aqui sistemas de apoio à decisão, diagnóstico médico, etc...), aplicações na área de negócios, etc... [Terano, Asai e Sugeno 1994].

3.2.1 Conjuntos Fuzzy

Um sistema cujas proposições são verdadeira ou falsa, mas não ambas, usa apenas dois valores lógicos, que representam apenas uma aproximação da razão humana. Os seguintes exemplos não constituem conjuntos no sentido matemático usual:

- o conjunto de homens altos;
- o conjunto de mulheres bonitas;
- o conjunto de carros caros;

- o conjunto de pessoas obesas;
- o conjunto de idades infantis;
- o conjunto de temperaturas quentes.

Para definir, por exemplo, o conjunto de homens altos poderia se definir que uma altura x é alta se $x \geq 176$ cm. Esta é uma aproximação abrupta para o significado "alto".

Usando uma função com um grau de pertinência pode-se fazer a transição de "não alto" para "alto" de uma forma gradual, conforme é mostrado na Figura 3.1 .

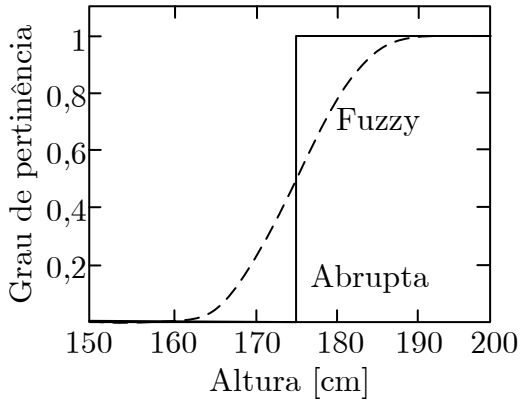

Figura 3.1: Duas definições para o conjunto de homens "altos": conjunto abrupto e conjunto fuzzy.

Definição: dada uma coleção de objetos U, um conjunto fuzzy A é o conjunto do par ordenado

$$A = \{[\, x, \mu(x)\,] \mid x \in U\} \tag{3.1}$$

sendo que $\mu(x)$ é chamada de função de pertinência do conjunto de todos os objetos x de U.

Para cada x há uma grau de pertinência $\mu(x)$ no intervalo fechado $[0, 1]$.

O conjunto fuzzy sugere uma região limitada, ao invés da fronteira abrupta dos conjuntos clássicos.

Alguns exemplos:

- O conjunto de temperaturas altas, o conjunto de ventos fortes ou o conjunto de dias ensolarados são conjuntos fuzzy em relatórios de previsão de tempo.
- O conjunto de pessoas jovens. Um bebê de 1 ano seria claramente membro do conjunto, enquanto que um idoso de 100 anos não seria membro do conjunto. Uma pessoa de 30 anos poderia ser membro do conjunto, mas com grau de pertinência de $0,5$.
- O conjunto de pessoas adultas. Nas eleições do Brasil é permitido votar as pessoas com idade igual ou superior a 16 anos. Segundo esta definição o conjunto de pessoas adultas é um conjunto abrupto.

3.2.2 Funções de pertinência

A função de pertinência $\mu(x)$, de um conjunto fuzzy contínuo A, expressa o quanto um elemento x pertence ao conjunto A, conforme ilustra a Figura 3.2.

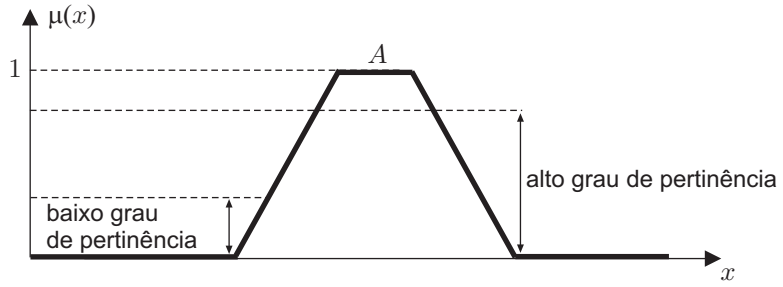

Figura 3.2: Graus de pertinência de um conjunto fuzzy A.

A Figura 3.3 ilustra quatro possíveis funções de pertinência.

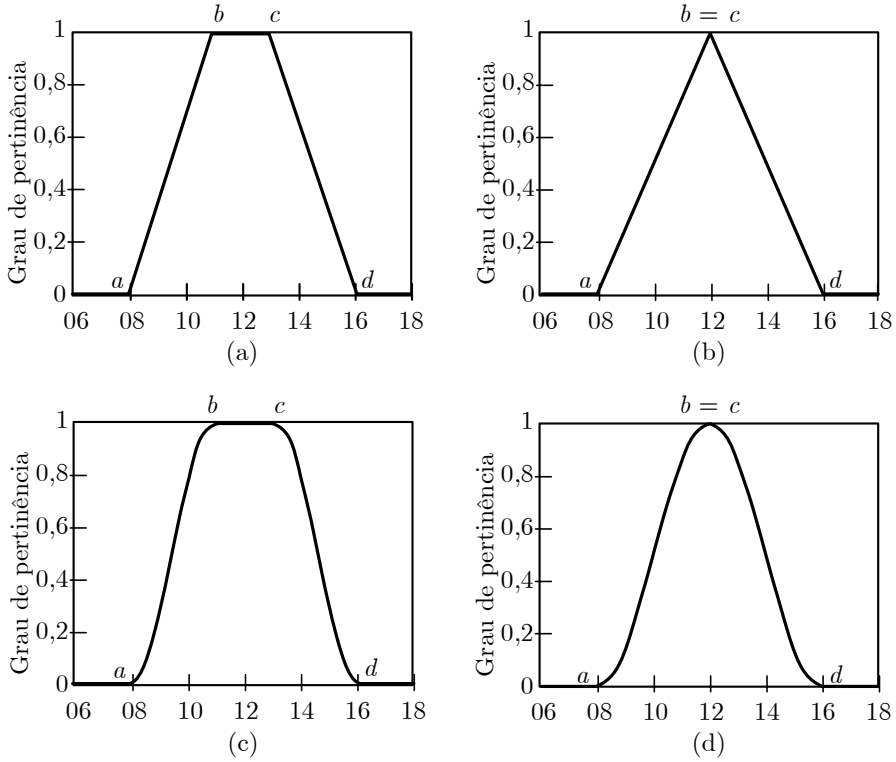

Figura 3.3: Funções de pertinência representando as horas por volta do meio-dia. (a) trapezoidal (b) tringular (c) trapezoidal suave (d) triangular suave.

A função de pertinência trapezoidal é uma função contínua, *linear por trechos*, com quatro parâmetros a, b, c, d, dada por

$$\mu(x) = \begin{cases} 0 & , \ x < a \\ \frac{x-a}{b-a} & , \ a \leq x < b \\ 1 & , \ b \leq x < c \\ \frac{d-x}{d-c} & , \ c \leq x < d \\ 0 & , \ d \leq x \end{cases} \tag{3.2}$$

A função de pertinência triangular é resultante da função trapezoidal com $b = c$, ou seja

$$\mu(x) = \begin{cases} 0 & , \ x < a \\ \frac{x-a}{b-a} & , \ a \leq x < b \\ \frac{d-x}{d-b} & , \ b \leq x < d \\ 0 & , \ d \leq x \end{cases} \tag{3.3}$$

Funções de pertinência suaves podem ser obtidas *trocando-se a parte linear pela função cosseno.* A função de pertinência trapezoidal suave é dada por

$$\mu(x) = \begin{cases} 0 & , \ x < a \\ \frac{1}{2} + \frac{1}{2}\cos\left(\frac{x-b}{b-a}\pi\right) & , \ a \leq x < b \\ 1 & , \ b \leq x < c \\ \frac{1}{2} + \frac{1}{2}\cos\left(\frac{x-c}{d-c}\pi\right) & , \ c \leq x < d \\ 0 & , \ d \leq x \end{cases} \tag{3.4}$$

Fazendo $b = c$ na função (3.4), a função de pertinência triangular suave resulta como

$$\mu(x) = \begin{cases} 0 & , \ x < a \\ \frac{1}{2} + \frac{1}{2}\cos\left(\frac{x-b}{b-a}\pi\right) & , \ a \leq x < b \\ \frac{1}{2} + \frac{1}{2}\cos\left(\frac{x-b}{d-b}\pi\right) & , \ b \leq x < d \\ 0 & , \ d \leq x \end{cases} \tag{3.5}$$

Outras funções de pertinência podem ser utilizadas como a gaussiana, sigmoide, etc.

3.2.3 Operações com conjuntos fuzzy

Dois conjuntos fuzzy A e B são iguais se eles têm a mesma função de pertinência para todo x

$$A = B \ \equiv \ \mu_A(x) = \mu_B(x) \ . \tag{3.6}$$

Um conjunto fuzzy A está contido num conjunto fuzzy B, se a função de pertinência de A é menor ou igual a de B, ou seja

$$A \subseteq B \ \equiv \ \mu_A(x) \leq \mu_B(x) \ . \tag{3.7}$$

A união clássica dos conjuntos X e Y, simbolizado por $X \cup Y$, é o conjunto de todos os objetos que são membros de X ou Y, ou ambos, isto é,

$$X \cup Y \equiv \{x \mid x \in X \text{ ou } x \in Y\} . \tag{3.8}$$

Por exemplo,

$$\{1, 2, 3\} \cup \{1, 3, 4\} = \{1, 2, 3, 4\} .$$

A intersecção clássica dos conjuntos X e Y, simbolizado por $X \cap Y$, é o conjunto de todos os objetos que são membros de ambos X e Y, isto é,

$$X \cap Y \equiv \{x \mid x \in X \text{ e } x \in Y\} . \tag{3.9}$$

Por exemplo,

$$\{1, 2, 3\} \cap \{1, 3, 4\} = \{1, 3\} .$$

O complemento clássico de um conjunto X, simbolizado por $\overline{X}$, é o conjunto dos membros x que não pertencem a X, isto é,

$$\overline{X} \equiv \{x \mid x \notin X\} . \tag{3.10}$$

Os diagramas com operações clássicas de conjuntos são apresentados na Figura 3.4.

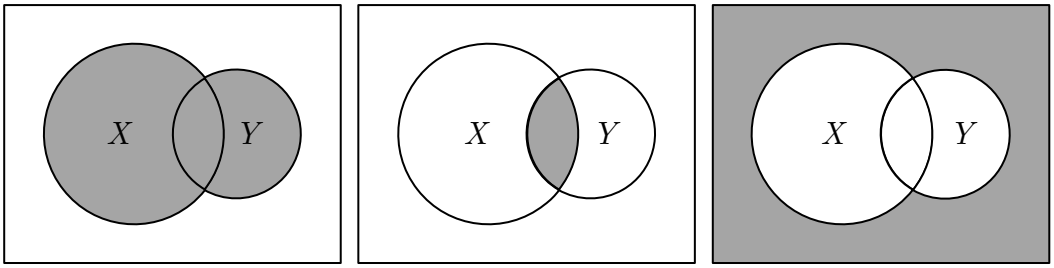

Figura 3.4: Operações clássicas de conjuntos: (a) união $X \cup Y$ (b) intersecção $X \cap Y$ (c) complemento $\overline{X \cup Y}$.

No caso de conjuntos fuzzy, deve-se considerar as funções de pertinência.

A união de dois conjuntos fuzzy A e B é

$$A \cup B \equiv \{[\, x, \mu_{A \cup B}(x)\,] \mid \mu_{A \cup B}(x) = \max[\, \mu_A(x), \mu_B(x)\,]\, \} . \tag{3.11}$$

A intersecção de dois conjuntos fuzzy A e B é

$$A \cap B \equiv \{[\, x, \mu_{A \cap B}(x)\,] \mid \mu_{A \cap B}(x) = \min[\, \mu_A(x), \mu_B(x)\,]\, \} . \tag{3.12}$$

O complemento de um conjunto fuzzy A é

$$\overline{A} \equiv \{[\, x, \mu_{\overline{A}}(x)\,] \mid \mu_{\overline{A}}(x) = 1 - \mu_A(x)\}\ . \tag{3.13}$$

Os diagramas com operações de conjuntos fuzzy são apresentados na Figura 3.5.

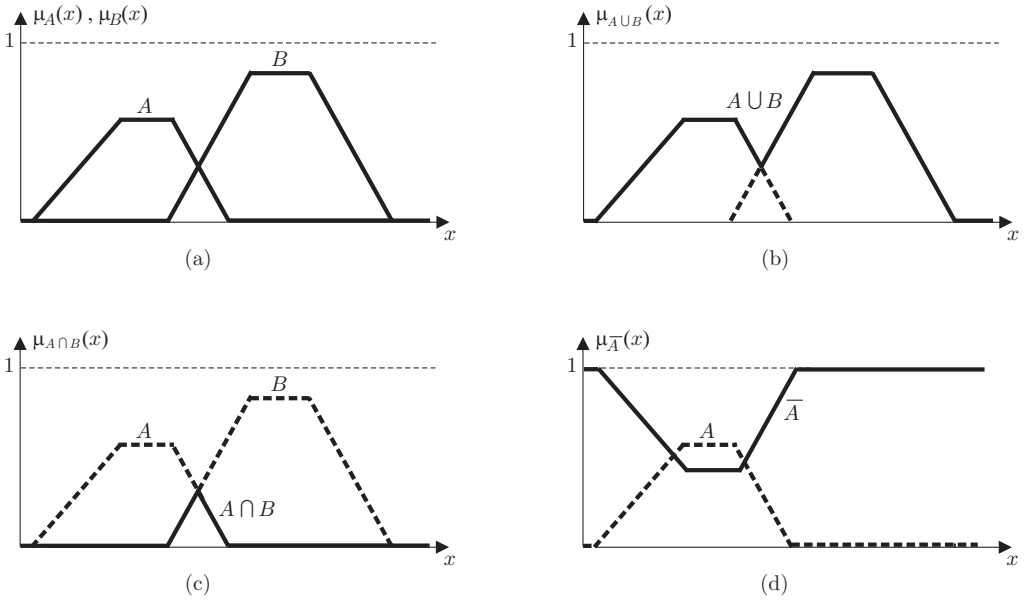

Figura 3.5: Operações com conjuntos fuzzy: (a) $\mu_A(x)$, $\mu_B(x)$ (b) $\mu_{A\cup B}(x)$ (c) $\mu_{A\cap B}(x)$ (d) $\mu_{\overline{A}}(x)$.

Considere o seguinte exemplo:

Uma família de quatro pessoas deseja comprar uma casa. O conjunto de casas disponíveis no mercado possui o seguinte número de quartos

$$U = \{1\ ;\ 2\ ;\ 3\ ;\ 4\ ;\ 5\ ;\ 6\ ;\ 7\ ;\ 8\ ;\ 9\ ;\ 10\}\ .$$

O conjunto fuzzy C representa o Conforto de cada casa, com grau de pertinência

$$\mu_C = \{0,2\ ;\ 0,5\ ;\ 0,8\ ;\ 1\ ;\ 0,7\ ;\ 0,3\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0\}\ .$$

O conjunto fuzzy G representa o quão Grande é cada casa, com grau de pertinência

$$\mu_G = \{0\ ;\ 0\ ;\ 0,2\ ;\ 0,4\ ;\ 0,6\ ;\ 0,8\ ;\ 1\ ;\ 1\ ;\ 1\ ;\ 1\}\ .$$

A intersecção dos conjuntos Conforto e Grande é

$$C \cap G = \min(\mu_C, \mu_G) = \{0\ ;\ 0\ ;\ 0,2\ ;\ 0,4\ ;\ 0,6\ ;\ 0,3\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0\}\ .$$

Logo, a melhor solução é uma casa com cinco quartos que tem um grau de pertinência igual a $0,6$.

A união dos conjuntos Conforto e Grande é

$$C \cup G = \max(\mu_C, \mu_G) = \{0,2\ ;\ 0,5\ ;\ 0,8\ ;\ 1\ ;\ 0,7\ ;\ 0,8\ ;\ 1\ ;\ 1\ ;\ 1\ ;\ 1\}\ .$$

No caso da união, uma casa com quatro ou de sete a dez quartos é completamente adequada, pois o grau de pertinência vale 1.

Se os filhos estiverem para se casar e forem mudar da casa em alguns meses, então é necessário comprar uma casa com Conforto e não Grande, com conjunto

$$C \cap \overline{G} = \min(\mu_C, 1 - \mu_G) = \{0,2\ ;\ 0,5\ ;\ 0,8\ ;\ 0,6\ ;\ 0,4\ ;\ 0,2\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0\} \ ,$$

sendo que o melhor resultado é uma casa com três quartos, com grau de pertinência igual a $0,8$.

3.2.4 Regra de Mamdani

Frequentemente usada em controle fuzzy.

Sejam A e B dois conjuntos fuzzy definidos em x e y, respectivamente. Então

$$\mu(x, y) = \min[\mu_A(x), \mu_B(y)] \ . \tag{3.14}$$

3.2.5 Variáveis linguísticas

Uma variável linguística possui palavras ou sentenças como valores.

Suponha, por exemplo, que **idade** seja uma variável linguística de um conjunto fuzzy U, definido como

$$U(\text{idade}) = \{\text{jovem}, \text{muito jovem}, \text{não muito jovem}, \text{mais ou menos velho}, \text{velho}\} \ . \tag{3.15}$$

Cada termo é definido num intervalo de, por exemplo, inteiros de 0 a 100 anos.

Um termo pode ter o seu significado modificado. Por exemplo, na sentença “muito próximo de zero”, a palavra “muito” modifica o termo “próximo de zero”. Outros exemplos são: “pequeno”, “mais ou menos”, “possivelmente”, etc.

O efeito de “muito” é o de intensificar o valor da função de pertinência, ou seja,

$$\text{muito } A \equiv \{x, \mu_{\text{muito } A}(x) \mid \mu_{\text{muito } A}(x) = \mu_A^2(x), x \in X\} \ . \tag{3.16}$$

O efeito de “mais ou menos” é o oposto

$$\text{mais ou menos } A \equiv \{x, \mu_{\text{mais ou menos } A}(x) \mid \mu_{\text{mais ou menos } A}(x) = \sqrt{\mu_A(x)}, x \in X\} \ . \tag{3.17}$$

As funções de pertinência de “muito jovem” e “não muito jovem” são oriundas de “jovem” e a função de pertinência de “mais ou menos velho” é oriunda de “velho”, conforme é ilustrado na Figura 3.6.

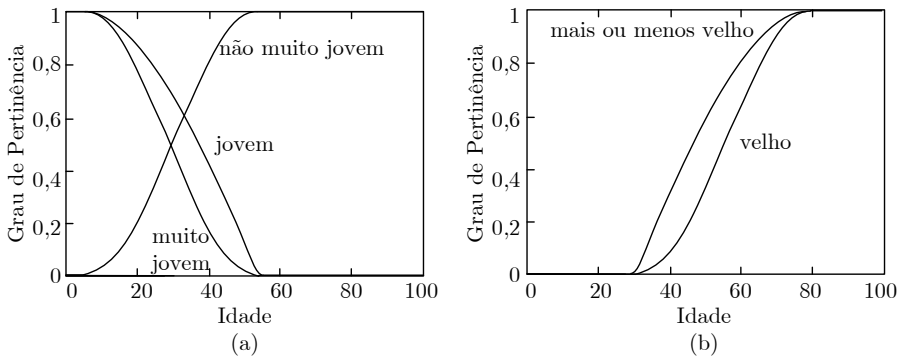

Figura 3.6: Funções de pertinência modificadas.

Uma família de funções de pertinência pode ser gerada por μ_A^k ou $\mu_A^{1/k}$ de acordo com os termos modificadores de significado.

Considere como exemplo o conjuntos de idades

$$U = \left[\begin{array}{ccccc} 0 & 20 & 40 & 60 & 80 \end{array} \right] \tag{3.18}$$

com funções de pertinência

$$\mu_{\text{jovem}} = \left[\begin{array}{ccccc} 1 & 0,6 & 0,1 & 0 & 0 \end{array} \right] \quad \text{e} \quad \mu_{\text{velho}} = \left[\begin{array}{ccccc} 0 & 0 & 0,1 & 0,6 & 1 \end{array} \right] . \tag{3.19}$$

A função de pertinência do conjunto muito jovem é

$$\mu_{\text{muito jovem}} = \mu_{\text{jovem}}^2 = \left[\begin{array}{ccccc} 1 & 0,36 & 0,01 & 0 & 0 \end{array} \right] . \tag{3.20}$$

A função de pertinência do conjunto mais ou menos velho é

$$\mu_{\text{mais ou menos velho}} = \sqrt{\mu_{\text{velho}}} = \left[\begin{array}{ccccc} 0 & 0 & 0,32 & 0,77 & 1 \end{array} \right] . \tag{3.21}$$

3.3 Controle Fuzzy

Controle fuzzy significa controlar com regras. O diagrama de blocos de um sistema de controle típico com realimentação é apresentado na Figura 3.7.

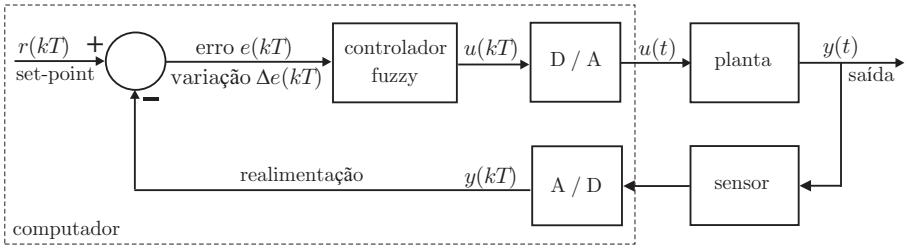

Figura 3.7: Diagrama de blocos de um sistema de controle em malha fechada.

Um controle fuzzy pode possuir regras empíricas, baseadas principalmente em plantas controladas por operadores. As regras são do tipo `se-então` com a premissa do lado do `se` e a conclusão do lado do `então`. Exemplo:

Se o erro é `N` **e** a variação do erro é `N` **então** o controle é `GN`.

Se o erro é `N` **e** a variação do erro é `Z` **então** o controle é `MN`.

Os termos linguísticos `N (Negativo)` ou `Z (Zero)` são premissas e `GN (Grande Negativo)` ou `MN (Médio Negativo)` são conclusões.

Um computador ou equipamento digital pode executar as regras e calcular o sinal de controle $u[kT]$ a partir de medidas do erro e da variação do erro, ou seja,

$$\text{erro atual} = \text{set-point} - \text{saída atual da planta: } e[kT] = r[kT] - y[kT] \ , \tag{3.22}$$

$$\text{variação do erro} = \text{erro atual} - \text{erro no passo anterior: } \Delta e[kT] = e[kT] - e[(k-1)T] \ . \tag{3.23}$$

3.3.1 Tipos de Controladores Fuzzy

Um controlador fuzzy pode se assemelhar a um controlador **PID**. A seguir são apresentados alguns tipos de controladores com base em [Jantzen 2007].

- **Fuzzy P**

Um controlador proporcional amplifica o sinal de erro e possui uma função do tipo

$$u(t) = K_p \, e(t) \ . \tag{3.24}$$

Um controlador fuzzy proporcional tem regras do tipo

`Se erro` $e(kT)$ `= A então sinal de controle` $u(kT)$ `= B.`

sendo A e B conjuntos fuzzy com funções de pertinência linguísticas atribuídas às variáveis $e(kT)$ e $u(kT)$.

O diagrama de blocos é apresentado na Figura 3.8.

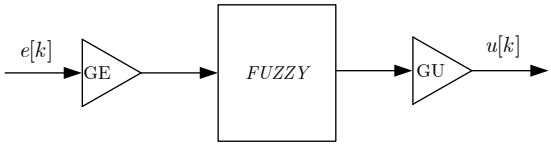

Figura 3.8: Digrama de blocos do controlador fuzzy P.

- **Fuzzy PD**

Um controlador proporcional+derivativo tem a função de melhorar a estabilidade relativa do sistema e possui uma função do tipo

$$u(t) = K_p\, e(t) + K_d \frac{de(t)}{dt}\ . \tag{3.25}$$

Um controlador fuzzy proporcional+derivativo tem regras do tipo

`Se erro` $e(kT)$ `= A e variação do erro` $\Delta e(kT)$ `= B então sinal de controle` $u(kT)$ `= C.`

sendo A, B e C conjuntos fuzzy com funções de pertinência linguísticas atribuídas às variáveis $e(kT)$, $\Delta e(kT)$ e $u(kT)$.

O diagrama de blocos é apresentado na Figura 3.9.

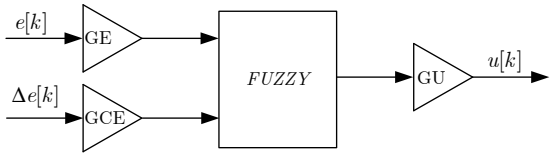

Figura 3.9: Digrama de blocos do controlador fuzzy PD.

- **Fuzzy Incremental (equivalente PI)**

Um controlador proporcional+integral tem a função de eliminar o erro estacionário entre a referência e a saída do processo e possui uma função do tipo

$$u(t) = K_p\, e(t) + K_i \int e(t)dt \quad \text{ou} \tag{3.26}$$

$$\frac{du(t)}{dt} = K_p\, \frac{de(t)}{dt} + K_i\, e(t)\ . \tag{3.27}$$

Um controlador fuzzy proporcional+integral tem regras do tipo

`Se erro` $e(kT)$ `= A e variação do erro` $\Delta e(kT)$ `= B então variação do controle` $\Delta u(kT)$ `= C.`

Note que o resultado da conclusão da regra é a variação do sinal de controle $\Delta u(kT)$ e não $u(kT)$ como nos controladores fuzzy P ou PD. Trata-se de um controlador incremental, onde a saída do controlador é dada por um incremento, Δu, positivo ou negativo na ação de controle. A Figura 3.10 apresenta o diagrama de blocos deste controlador.

- **Fuzzy PD+I**

Trata-se de um controlador fuzzy PD, onde o erro e a variação do erro são as entradas do bloco fuzzy. Uma ação acumulativa (integral) do erro é adicionada à ação de controle, conforme ilustrado na Figura 3.11.

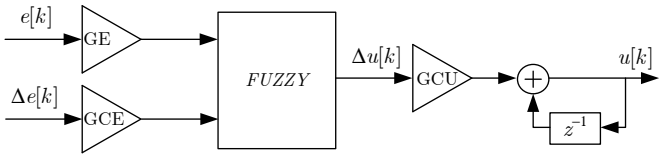

Figura 3.10: Digrama de blocos do controlador fuzzy incremental (PI).

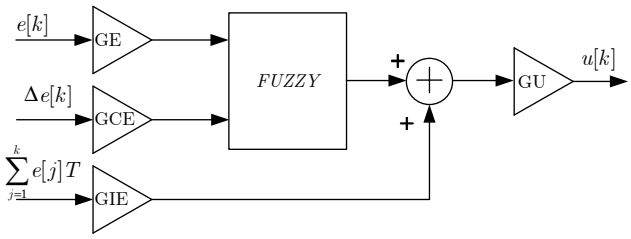

Figura 3.11: Digrama de blocos do controlador fuzzy PD+I.

3.3.2 Estrutura interna de um controlador fuzzy

Um controlador fuzzy pode ser dividido em quatro blocos, conforme a Figura 3.12.

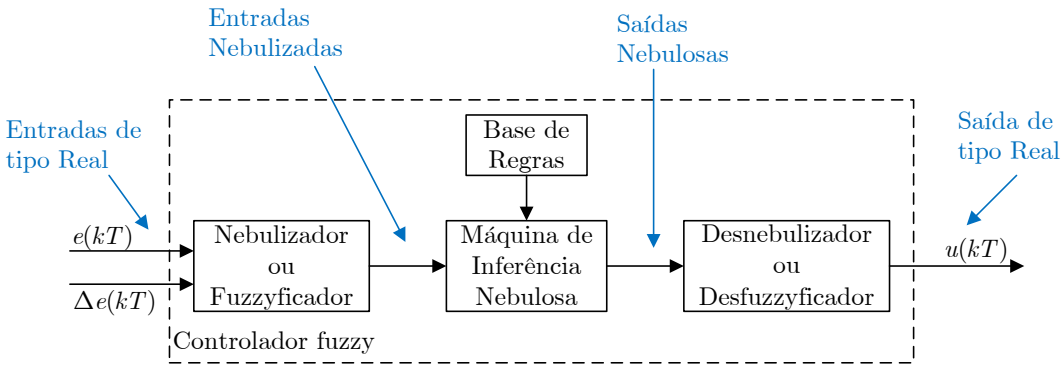

Figura 3.12: Estrutura de um controlador fuzzy.

O **Nebulizador** ou **fuzzyficador** tem por funções:

- converter esses valores numéricos em valores linguísticos associados aos conjuntos nebulosos.
- mudar as escalas das variáveis de entrada do controlador, normalizando-as de maneira a pertencerem aos universos de discurso dos conjuntos nebulosos correspondentes;
- discretizar os valores numéricos das variáveis de entrada;

A **Base de regras** é uma base de conhecimento do processo e dos objetivos de controle, traduzida num conjunto de regras linguísticas de controle.

A **Máquina de inferência nebulosa** realiza a lógica de tomada de decisões, baseando-se na **Base de regras** e nos valores linguísticos das variáveis medidas e controladas.

O **Desnebulizador** ou **desfuzzificador** é responsável por converter os valores nebulizados para variáveis numéricas.

A seguir será detalhado cada um desses blocos.

3.3.2.1 Base de regras

Como anteriormente dito, um controlador fuzzy se apoia em regras. As regras do controlador são baseadas na linguagem natural e podem ser entendidas por operadores ou por equipes de manutenção. Alguns critérios para a elaboração das regras são apresentados a seguir.

- *Experiência de funcionários e conhecimento de engenheiros*: elaborar um questionário detalhado a ser preenchido por funcionários ou engenheiros com o objetivo de extrair regras de controle fuzzy.
- *Baseado nas ações de controle do operador*: observando as ações de controle do operador pode-se encontrar relações do tipo entrada-saída que podem ser utilizadas na elaboração das regras fuzzy.
- *Examinando manuais* também podem ser obtidas informações úteis para a elaboração das regras.
- *Baseado no modelo da planta*: uma regra linguística pode ser vista como o inverso da planta. Assim, é possível obter regras fuzzy invertendo o modelo da planta. Este método é restrito a sistemas de baixa ordem.

Num sistema de controle, o sinal do erro é usualmente numérico. Num servomecanismo posicionador, como será o caso desta prática, o ângulo de saída pode ser uma tensão variando, por exemplo, entre $-4,5$V a $+4,5$V. Vamos imaginar que a experiência de processo tenha levado à quantificação do erro e da variação do erro da seguinte forma:

```
GP = Grande Positivo;

PP = Pequeno Positivo;

ZE = Zero;

PN = Pequeno Negativo;

GN = Grande Negativo.
```

E do esforço de controle:

```
GP = Grande Positivo;

MP = Médio Positivo;

ZE = Zero;

MN = Médio Negativo;
```

```
GN = Grande Negativo.
```

Essa mesma experiência de processo levou também à definição das seguintes regras básicas apresentadas na Tabela 3.1.

Tabela 3.1: Regras básicas de controle.

Erro	Variação do Erro	Sinal de Controle
GN	GN	GN
GN	PN	GN
GN	ZE	GN
GN	PP	MN
GN	GP	MN
PN	GN	MN
PN	PN	MN
PN	ZE	ZE
PN	PP	ZE
PN	GP	MP
ZE	GN	MN
ZE	PN	MN
ZE	ZE	ZE
ZE	PP	MP
ZE	GP	MP
PP	GN	MN
PP	PN	ZE
PP	ZE	ZE
PP	PP	MP
PP	GP	MP
GP	GN	MP
GP	PN	MP
GP	ZE	GP
GP	PP	GP
GP	GP	GP

Uma forma mais compacta é apresentar as regras na forma matricial da Tabela 3.2.

Nota importante: para manter a simplicidade desta seção, não foram definidas as regras para Erro e Variação do Erro em MN e MP. Obviamente, para a implementação da tabela de regras é necessário que

Tabela 3.2: Forma compacta de visualização das regras básicas de controle.

		Erro				
Variação do Erro		GN	PN	ZE	PP	GP
	GN	GN	MN	MN	MN	MP
	PN	GN	MN	MN	ZE	MP
	ZE	GN	ZE	ZE	ZE	GP
	PP	MN	ZE	MP	MP	GP
	GP	MN	MP	MP	MP	GP

todos os casos tenham sido previstos.

A definição da Tabela 3.2 é o ponto de partida para a definição do conteúdo de todos os outros blocos do controlador fuzzy da Figura 3.12.

3.3.2.2 Nebulizador ou fuzzyficador

Considere por exemplo um sistema que tem como entrada a temperatura, que assume diversos valores fuzzy e que tem como saída o fluxo de calor, que também assume valores fuzzy. Este sistema pode ser representado pelo modelo linguístico:

se temperatura é `alta` **então** o fluxo de calor é `positivo`.

se temperatura é `ideal` **então** o fluxo de calor é `nulo`.

se temperatura é `baixa` **então** o fluxo de calor é `negativo`.

As funções de pertinência para a temperatura podem ser escolhidas como, por exemplo, na Figura 3.13. Foram escolhidas funções trapezoidais com centros em `alta, ideal` e `baixa.` As larguras e os centros dos trapézios são parâmetros que podem ser ajustados com base na experiência do operador. Outros tipos de funções poderiam ser escolhidas, como, por exemplo, gaussianas ou triangulares.

Note que a temperatura de 25^o é `ideal` com grau de pertinência $\mu = 1$ e ao mesmo tempo é `alta` com grau de pertinência $\mu = 0,6$ e é `baixa` com grau de pertinência $\mu = 0,6$.

Retomando o exemplo da Tabela 3.2, para cada um dos conjuntos nebulosos `GP, PP, ZE, PN, GN` deve-se associar funções de pertinência μ. A forma da função é arbitrária. Por simplicidade, as mais utilizadas são as funções trapezoidais e triangulares.

Na Figura 3.14 são ilustradas funções triangulares para cada um dos conjuntos.

Note que o erro $e(kT) = 3,375$ pertence ao conjunto `PP` e `GP` com o mesmo grau de pertinência $\mu = 0,5$. Por extensão, ele também é `GN, PN` e `ZE` com grau de pertinência 0.

Assim, o **Nebulizador** ou **fuzzyficador**, como eram suas atribuições, muda as escalas das variáveis de entrada do controlador, normalizando-as de maneira a pertencerem aos universos de discurso dos conjuntos nebulosos correspondentes, e converte esses valores numéricos em valores linguísticos associados

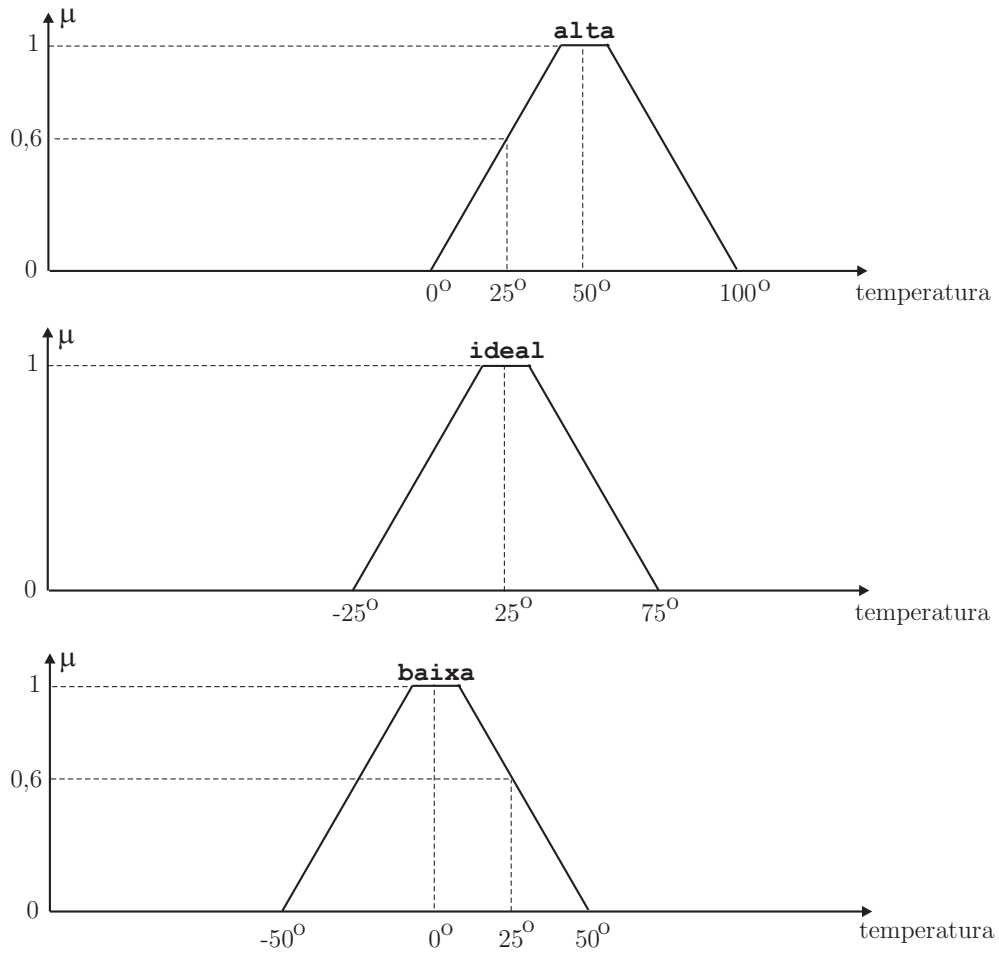

Figura 3.13: Funções de pertinência.

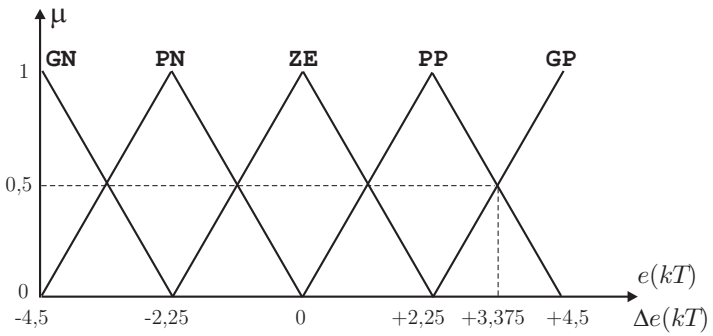

Figura 3.14: Funções de pertinência ou de associação.

aos conjuntos nebulosos.

3.3.2.3 Máquina de inferência nebulosa

Como citado anteriormente, a **Máquina de inferência nebulosa** realiza a lógica de tomada de decisões, baseando-se na **Base de regras** e nos valores linguísticos das variáveis medidas e controladas.

Um controlador fuzzy pode assumir a forma de um sistema aditivo como o da Figura 3.15.

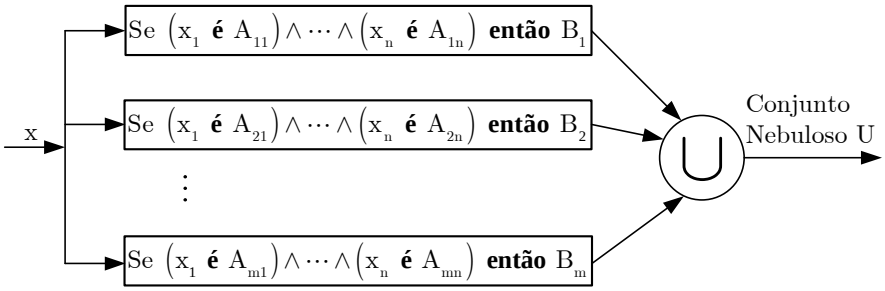

Figura 3.15: Sistema nebuloso aditivo.

Suponha que quatro regras tenham sido formuladas por um operador experiente para o controle de um servomecanismo posicionador:

- **regra 1: se** $e(kT)$ é `ZE` **e** $\Delta e(kT)$ é `PP` **então** $u(kT)$ deve ser `PP`;

  ou

- **regra 2: se** $e(kT)$ é `ZE` **e** $\Delta e(kT)$ é `ZE` **então** $u(kT)$ deve ser `ZE`;

  ou

- **regra 3: se** $e(kT)$ é `PN` **e** $\Delta e(kT)$ é `PN` **então** $u(kT)$ deve ser `PN`;

  ou

- **regra 4: se** $e(kT)$ é `GP` **e** $\Delta e(kT)$ é `ZE` **então** $u(kT)$ deve ser `GP`.

Note que cada regra apresenta um conectivo **e,** caracterizando assim uma operação de intersecção entre conjuntos nebulosos. Além disso, entre as regras há um conectivo **ou,** indicando operações de união.

Suponha que num certo instante o servomecanismo posicionador apresente um erro $e(kT) = -0,75$V e uma variação de erro $\Delta e(kT) = +1,2$V. Nessa situação, os graus de pertinência são ilustrados na Figura 3.16.

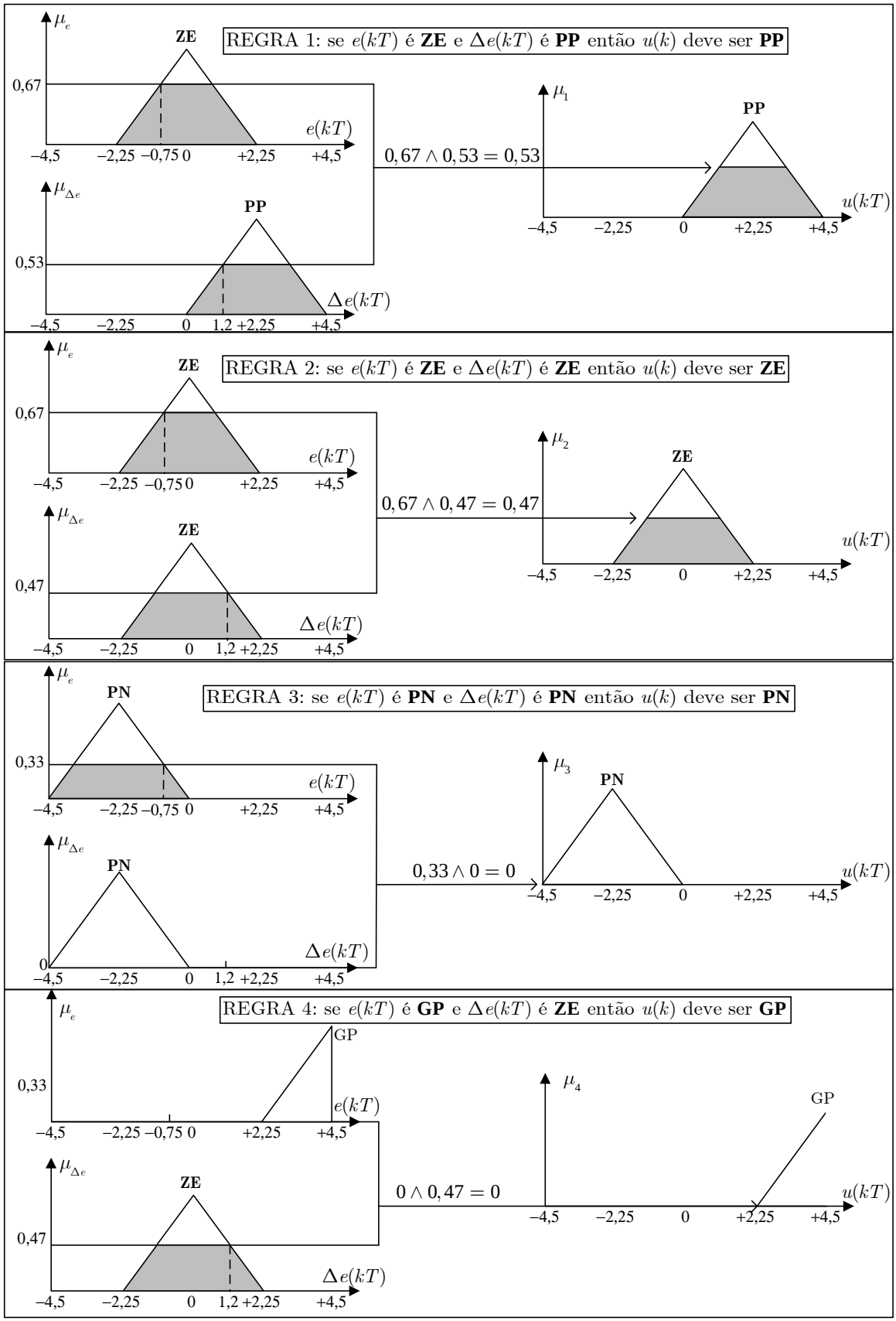

Figura 3.16: Graus de pertinência para $e(kT) = -0,75$V e $\Delta e(kT) = +1,2$V.

Dado o valor do erro $e(kT)$ e da variação do erro $\Delta e(kT)$, em cada instante kT, o valor da função de pertinência de cada regra pode ser calculado por meio da equação (3.3) para funções triangulares:

regra 1

$$\mu_e = \frac{x-a}{b-a} = \frac{x-(-2,25)}{0-(-2,25)} = \frac{e(kT)+2,25}{2,25} = \frac{-0,75+2,25}{2,25} \cong 0,67\ . \tag{3.28}$$

$$\mu_{\Delta e} = \frac{x-a}{b-a} = \frac{x-0}{2,25-0} = \frac{\Delta e(kT)}{2,25} = \frac{1,2}{2,25} \cong 0,53\ . \tag{3.29}$$

regra 2

$$\mu_e = \frac{x-a}{b-a} = \frac{x-(-2,25)}{0-(-2,25)} = \frac{e(kT)+2,25}{2,25} = \frac{-0,75+2,25}{2,25} \cong 0,67\ . \tag{3.30}$$

$$\mu_{\Delta e} = \frac{d-x}{d-b} = \frac{2,25-x}{2,25-0} = \frac{2,25-\Delta e(kT)}{2,25} = \frac{2,25-1,2}{2,25} \cong 0,47\ . \tag{3.31}$$

regra 3

$$\mu_e = \frac{d-x}{d-b} = \frac{0-x}{0-(-2,25)} = \frac{-e(kT)}{2,25} = \frac{-(-0,75)}{2,25} \cong 0,33\ . \tag{3.32}$$

$$\mu_{\Delta e} = 0\ . \tag{3.33}$$

regra 4

$$\mu_e = 0 \tag{3.34}$$

$$\mu_{\Delta e} = \frac{d-x}{d-b} = \frac{2,25-x}{2,25-0} = \frac{2,25-\Delta e(kT)}{2,25} = \frac{2,25-1,2}{2,25} \cong 0,47\ . \tag{3.35}$$

O grau de pertinência μ_i $(i = 1, 2, 3, 4)$ de cada conjunto U_i $(i = 1, 2, 3, 4)$ é obtido por meio da regra de Mamdani:

- **regra 1:** $\mu_1 = \min[\mu_e, \mu_{\Delta e}] = \min[0,67; 0,53] = 0,53.$
- **regra 2:** $\mu_2 = \min[\mu_e, \mu_{\Delta e}] = \min[0,67; 0,47] = 0,47.$
- **regra 3:** $\mu_3 = \min[\mu_e, \mu_{\Delta e}] = \min[0,33; 0] = 0.$
- **regra 4:** $\mu_4 = \min[\mu_e, \mu_{\Delta e}] = \min[0; 0,47] = 0.$

O conjunto fuzzy resultante é mostrado na Figura 3.17, que é obtido por meio da união dos conjuntos U_i $(i = 1, 2, 3, 4)$ de cada regra.

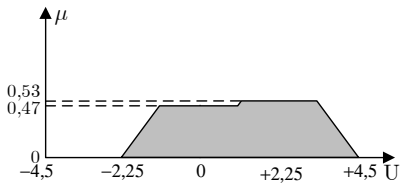

Figura 3.17: Conjunto fuzzy resultante.

3.3.2.4 Desnebulizador ou desfuzzificador

O **Desnebulizador** ou **desfuzzificador** é responsável por converter os valores nebulizados para variáveis numéricas. As regras nebulosas produzem como saída o conjunto nebuloso U da Figura 3.17. Para obter o sinal de saída do controlador $u(kT)$, que vai ser aplicado na entrada da planta no instante kT, é necessário converter os graus de pertinência numa variável numérica. Uma das formas (não a única) de se efetuar a desfuzzificação consiste no método das médias ponderadas, que consiste na seguinte expressão

$$u(kT) = \frac{\sum b_i \, . \mu_i}{\sum \mu_i} \, , \tag{3.36}$$

sendo b_i a abscissa central com o maior grau de pertinência em cada regra i.

No exemplo do servomecanismo posicionador obtém-se

$$u(kT) = \frac{0,53.(+2,25) + 0,47.(0) + 0.(-2,25) + 0.(+4,5)}{0,53 + 0,47 + 0 + 0} \approx 1,192 \, . \tag{3.37}$$

Uma outra forma de desfuzificação denomina-se método do centro de gravidade (CoG), que consiste em determinar a coordenada x do baricentro do conjunto fuzzy de saída. Considerando A_i como a área abaixo da i-ésima função de pertinência, tem-se a seguinte expressão

$$u(kT) = \frac{\sum b_i \, . A_i}{\sum A_i} \, , \tag{3.38}$$

No caso de funções de pertinência triangulares, a área A_i é simplesmente a área de um trapézio, como ilustrado na Figura 3.18.

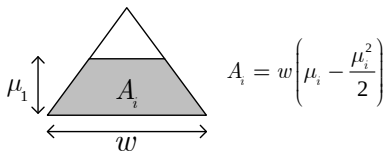

Figura 3.18: Área abaixo de μ_i.

3.3.3 Algumas considerações relevantes

Ao projetar um controlador fuzzy, o engenheiro se depara sempre com dois problemas subjetivos que dependem da preferência de escolha do projetista:

1. Qual formato deve ser utilizado para o conjunto?
2. Quantos conjuntos devem ser utilizados?

Algumas regras que podem orientar nas escolhas são:

- os conjuntos devem possuir uma quantidade razoável de sobreposições. Caso contrário, o controlador pode retornar uma saída indefinida;
- se há uma lacuna na vizinhança de dois conjuntos, então o sinal de controle fica indefinido nesta região;
- a quantidade de conjuntos depende da largura dos conjuntos e da faixa de funcionamento do sistema;
- um conjunto deve ser suficientemente largo para que seja diferenciado o ruído na medida do sinal.

3.4 Especificações de projeto

Efetuar o controle digital de posição do servomecanismo, utilizando controle nebuloso PD. Fixe o período de amostragem em $T_s = 1/20$ s.

3.5 Atividades de preparação da experiência

A seguir são descritas as atividades de preparação para a experiência 3, ou seja, a serem desenvolvidas **preferencialmente** em casa.

3.5.1 Projeto de controlador nebuloso

Escreva uma rotina em MATLAB de controle nebuloso PD para o servomecanismo do laboratório. Não é permitido utilizar a toolbox de controle nebuloso para a execução da atividade. Considere como saída a tensão do potenciômetro que mede a posição do eixo do servomecanismo. Para as simulações, utilize o SIMULINK e considere que a planta tenha o modelo da Figura 3.19.

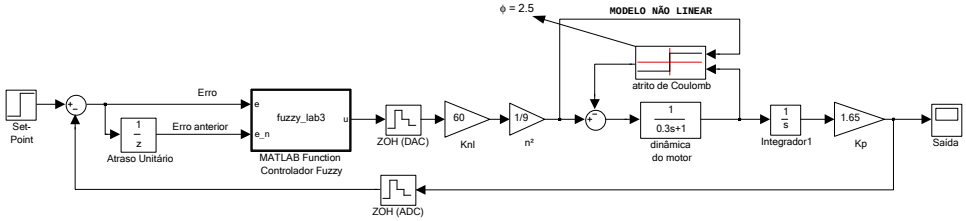

Figura 3.19: Modelo de simulação no SIMULINK.

Utilize o template da Experiência 4 do Laboratório de Controle (digite `>> exp4_template` no prompt do Matlab). Salve o arquivo com um nome adequado em sua pasta de trabalho. Modifique o arquivo mantendo apenas o modelo do sistema (apague o bloco do servo mecamisno), conforme apresentado na Figura 3.19.

O controlador fuzzy será implementado em uma estrutuda denominada MATLAB function, conforme Figura 3.20.

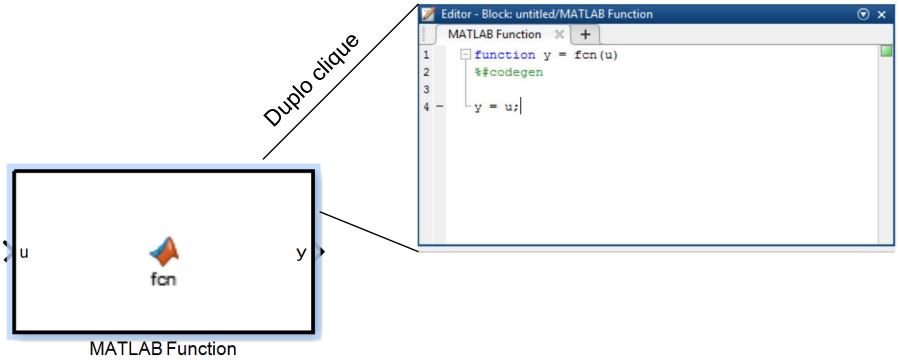

Figura 3.20: Bloco MATLAB function.

Ao clicar duplamente sobre bloco, abre-se o arquivo para edição da função, exatamente como uma função qualquer do MATLAB, permitindo definir mais de uma entrada e mais de uma saída, caso necessário. Note que logo após a declaração da função, aparece a diretiva `%#codegen`. Deve-se trocar tal diretiva por `%#eml`, para permitir que outras funções sejam chamadas dentro da MATLAB funtion.

Uma nova variável incluída na chamada de uma função dentro do bloco MATLAB function será inicialmente configurada como do tipo input. Por outro lado, algumas vezes, o usuário desejará passar constantes para a função a partir, por exemplo, de variáveis contidas no espaço de trabalho do MATLAB. Para que isso seja possível, alguns passos de configuração são necessários:

- Defina a variável que deseja passar para a função na sua chamada;
- Abra o editor "Edit Data/Ports" conforme a Figura 3.21:

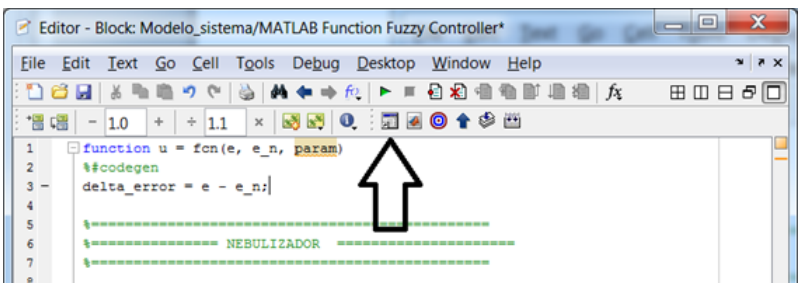

Figura 3.21: Seleção do editor "Edit Data/Ports".

- Selecione Parameter em Scope e elimine a seleção Tunable conforme a Figura 3.22:

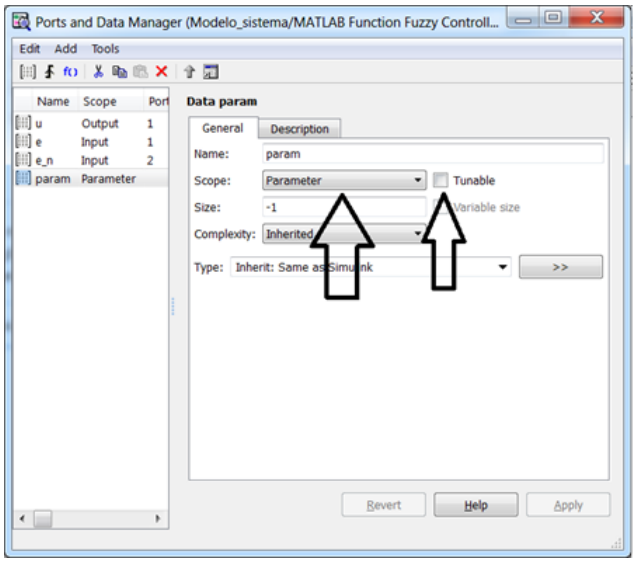

Figura 3.22: Seleção do tipo associado a uma variável da função contida no bloco MATLAB function.

Pronto! A variável param agora é, na realidade, uma constante cujo conteúdo está definido no espaço de trabalho do MATLAB.

SUGESTÕES:

- Inspire-se da Figura 3.14 para criar as funções de pertinência do Nebulizador. O uso de função triangular não é obrigatório.
- Utilize o conjunto de regras de controle da Tabela 3.3 para a implementação da Máquina de Inferência Nebulosa.

Tabela 3.3: Regras de controle para o servomecanismo

<table>
<tr><td></td><td colspan="8">Erro</td></tr>
<tr><td rowspan="8">Variação do Erro</td><td></td><td>GN</td><td>MN</td><td>PN</td><td>ZE</td><td>PP</td><td>MP</td><td>GP</td></tr>
<tr><td>GN</td><td colspan="2" rowspan="5">GN</td><td colspan="2" rowspan="2">MN</td><td>PN</td><td colspan="2">PP</td></tr>
<tr><td>MN</td><td>PN</td><td>ZE</td><td colspan="2">MP</td></tr>
<tr><td>PN</td><td rowspan="2">MN</td><td>ZE</td><td rowspan="2">MP</td><td colspan="2" rowspan="4">GP</td></tr>
<tr><td>ZE</td><td>PP</td></tr>
<tr><td>PP</td><td colspan="2">MN</td><td>ZE</td><td>PP</td></tr>
<tr><td>MP</td><td colspan="2" rowspan="2">PN</td><td rowspan="2">PP</td><td colspan="2" rowspan="2">MP</td></tr>
<tr><td>GP</td></tr>
</table>

3.6 Atividades práticas

A seguir são descritas as atividades práticas da experiência 3, ou seja, a serem desenvolvidas no laboratório.

Explore o conteúdo copiado de `c:/labaut` e leia o manual "Acesso ao Módulo Lynx AC1160-VA" para familiarização com as funções Matlab de configuração e uso do módulo de aquisição de dados Lynx.

Analise o código do arquivo `prog4.m` e utilize-o como base para elaboração das atividades propostas.

NÃO SE ESQUEÇA: Mantenha cópias de segurança de seus arquivos!

3.6.1 Implementação da planta analógica

Ligue o servomecanismo conforme a Figura 3.23.

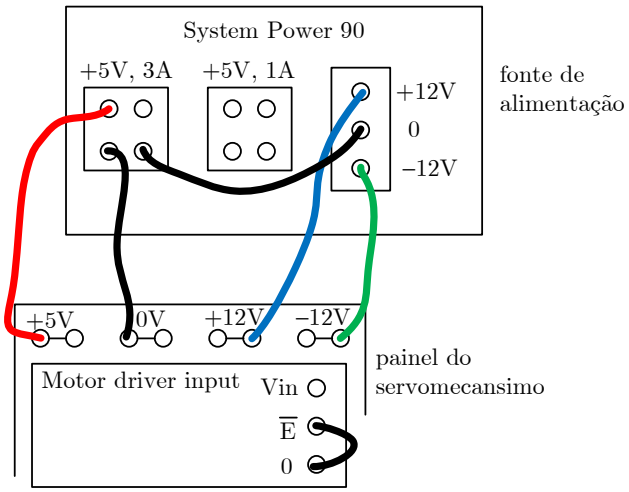

Figura 3.23: Ligação da fonte de alimentação ao servomecanismo.

3.6.2 Conexão do sistema de aquisição de sinais

O esquema de conexão abaixo deve ser utilizado para a realização desta prática. A saída de tensão do processo é o potenciômetro que mede a posição do eixo do servomecanismo.

Algumas observações importantes:

- As entradas e saídas do módulo de aquisição são do pondo de vista do controlador digital (PC), ou seja, o que entra no PC deve ser conectado na(s) entrada(s) INP0 ou/e INP1, e a(s) ação(ões) de controle "sai/saem" do PC pela(s) conexão(ões) de saída OUT0 e OUT1.
- **IMPORTANTE: nunca inverta entradas com saídas.**

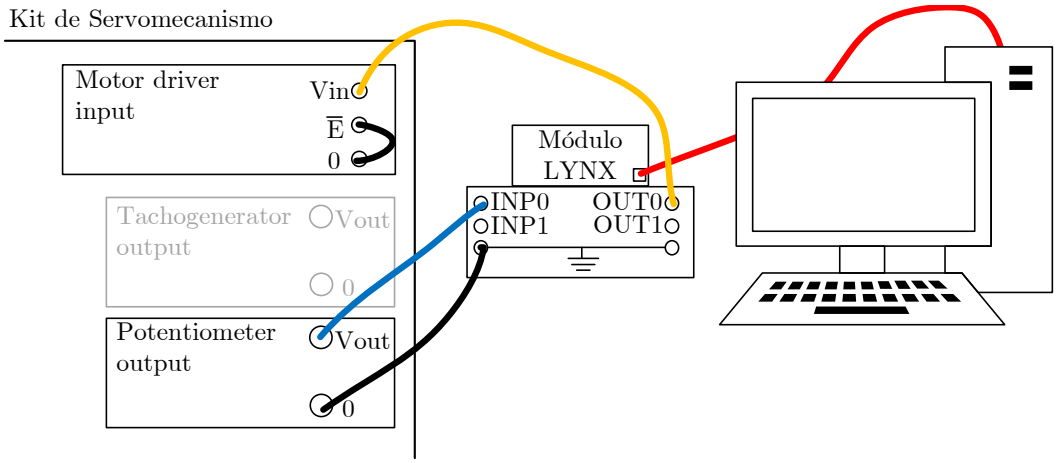

Figura 3.24: Conexão do sistema de aquisição de sinais para a experiência 3.

- **IMPORTANTE: jamais ligue os terminais OUT0 ou OUT1 diretamente a um terminal de terra ou aos terminais GND.**
- **IMPORTANTE: jamais ligue os terminais OUT0 ou OUT1 a uma fonte de tensão (como um gerador de funções ou a rede elétrica).**
- **IMPORTANTE: jamais ligue os terminais INP0 ou INP1 a fontes de tensão acima da faixa de ± 10V (como, por exemplo, à rede elétrica).**

3.6.3 Implementação do controlador nebuloso

a) Implemente no MATLAB a rotina do controlador fuzzy projetada no item 3.5.1 e registre a resposta ao degrau do sistema.

b) Faça experimentos, por exemplo, variando a frequência de amostragem e as regras da Tabela 3.3.

3.7 Relatório

Um relatório desta experiência deverá ser entregue. Para tanto, utilize o modelo de relatórios disponível em http://disciplinas.stoa.usp.br/.

3.8 Problemas e dúvidas frequentes

a) **Qual frequência de amostragem devo usar?**

A que você achar melhor. Note que uma frequência de amostragem muito alta fará com que o sinal "variação do erro" fique sempre muito pequeno ou nulo, fazendo com que apenas as linhas PN, ZE e PP (ou num caso extremo apenas a linha ZE) da Tabela 3.3 sejam empregadas.

Créditos

Esta experiência foi desenvolvida e/ou atualizada pelos seguintes professores:

- Anselmo Bittar
- José Jaime da Cruz
- Ricardo Paulino Marques
- Bruno Augusto Angélico
- Fábio de Oliveira Fialho

4 Projeto de Controlador PI Digital

4.1 Objetivo

A finalidade desta experiência é projetar controladores PI digitais para a malha de velocidade do servomecanismo do laboratório. Inicialmente, o controle será projetado em tempo contínuo e depois discretizado. Em seguida, o projeto será feito diretamente no plano-z.

4.2 Introdução teórica

A função de transferência de velocidade do servomecanismo é dada por:

$$G(s) = \frac{KK_t}{Ts+1}, \tag{4.1}$$

sendo K a relação entre a velocidade angular no eixo do tacogerador e a tensão da armadura, K_t o ganho do tacogerador e T a constante de tempo.

4.2.1 Projeto Contínuo e Discretização

Seja um controlador PI dado pela seguinte função de transferência:

$$C(s) = K_p\left(1 + \frac{1}{T_i s}\right), \tag{4.2}$$

onde K_p e T_i são, respectivamente, o ganho proporcional e o tempo de *reset* do integrador.

Uma vez que o controlador tenha sido devidamente projetado no plano-s, o mesmo pode ser discretizado utilizando o método de *Tustin*, dado por:

$$s = \frac{2}{T_s}\frac{(z-1)}{(z+1)} \tag{4.3}$$

A ação de controle pode ser dividida nas parcelas proporcional e integrativa, respectivamente dadas por

$$u_P(t) = K_P\, e(t), \tag{4.4}$$

e,

$$u_I(t) = \frac{K_P}{T_I} \int_0^t e(\tau)\, d\tau, \tag{4.5}$$

A discretização da parte proporcional é dada por $u_P[n] = K_P\, e[n]$. Ao considerar a aproximação *Tustin* para a parcela integrativa, tem-se que

$$u_I[n] = u_I[n-1] + \frac{K_P\, T_s}{2T_I}(e[n] + e[n-1]), \tag{4.6}$$

Portanto, o sinal de controle é dado por:

$$u[n] = u_P[n] + u_I[n] \tag{4.7}$$

Outra forma de PI digital é denominada PI incremental. Tal forma é obtida tomando-se a diferença do PI anteriormente definido no instante n e no instante $(n-1)$, ou seja;

$$\Delta u[n] = u[n] - u[n-1] = \Delta u_P[n] + \Delta u_I[n] \tag{4.8}$$

Assim,

$$u[n] = u[n-1] + \Delta u_P[n] + \Delta u_I[n] \tag{4.9}$$

Para o termo incremental proporcional, tem-se:

$$\Delta u_P[n] = u_P[n] - u_P[n-1] = K_P(e[n] - e[n-1]), \tag{4.10}$$

Para a parcela integrativa, chega-se em:

$$\Delta u_I[n] = \frac{K_P\, T_s}{2T_I}(e[n] + e[n-1]), \tag{4.11}$$

A desvantagem da discretização direta é que o atraso do segurador de ordem zero (ZOH) não é considerado no projeto.

4.2.2 Projeto Discreto

Para o projeto no plano-z, o equivalente discreto da planta é obtido pelo método ZOH, ou seja, o atraso devido ao segurador de ordem zero já aparece naturalmente. O equivalente discreto por ZOH é apresentado na Figura 4.1.

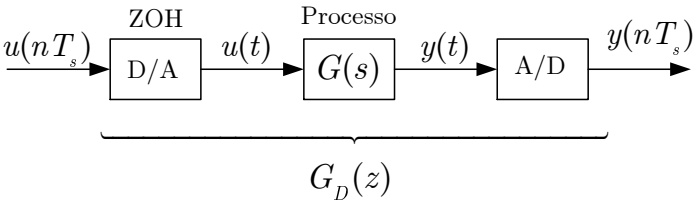

Figura 4.1: Equivalência do subsistema ZOH + processo + A/D.

O equivalente discreto de $G(s)$ com ZOH pode ser obtido como:

$$G_D(z) = \mathcal{Z}\left\{\frac{1-e^{-T_s s}}{s}G(s)\right\} = \mathcal{Z}\left\{\frac{G(s)}{s}\right\} - \mathcal{Z}\left\{e^{-T_s s}\frac{G(s)}{s}\right\} \tag{4.12}$$

Como $e^{-T_s s}$ é exatamente um atraso de um período de amostragem, verifica-se que

$$\mathcal{Z}\left\{e^{-T_s s}\frac{G(s)}{s}\right\} = z^{-1}\mathcal{Z}\left\{\frac{G(s)}{s}\right\}. \tag{4.13}$$

Portanto,

$$G_D(z) = \left(1-z^{-1}\right)\mathcal{Z}\left\{\frac{G(s)}{s}\right\} \tag{4.14}$$

O comando em MATLAB para transformar uma função de transferência contínua $G(s)$ considerando o efeito do ZOH em uma discreta $G_D(z)$, com período de amostragem T_s, é: `G_D = c2d(G,T_s,'zoh')`.

O equivalente discreto de (4.1) com ZOH na entrada é dado por:

$$G_D(z) = \frac{KK_t(1-e^{-T_s/T})}{z-e^{-T_s/T}} \tag{4.15}$$

A função de transferência discreta do PI pode ser dada por:

$$C_D(z) = K_c\frac{z-a}{z-1} \tag{4.16}$$

Ao escolher a para cancelar o polo em $z = e^{-T_s/T}$, tem-se em malha aberta

$$G_{D_{MA}}(z) = \frac{K_cKK_t(1-e^{-T_s/T})}{z-1} \tag{4.17}$$

Em malha fechada:

$$G_{D_{MF}}(z) = \frac{K_cKK_t(1-e^{-T_s/T})}{z-(1-K_cKK_t(1-e^{-T_s/T}))} \tag{4.18}$$

Se a constante de tempo desejada em malha for igual a τ, então deve-se resolver

$$e^{-T_s/\tau} = 1 - K_cKK_t(1-e^{-T_s/T}) \tag{4.19}$$

a fim de encontrar o ganho K_c do controlador. Note que a equação de diferenças que implementa o controle em (4.16) é dado por:

$$u[n] = u[n-1] + K_c\left(e[n] - ae[n-1]\right), \tag{4.20}$$

que é naturalmente uma forma incremental.

4.2.3 Sistema Anti-windup

Quando a implementação do controlador é digital, uma forma simples de *anti windup* consiste em "congelar" a ação de controle quando há saturação, conforme o código abaixo

```
% PID incremental com anti-windup
e[n] = r[n]-y[n];
Du_P = K_P*(e[n]-e[n-1]);
Du_I=(K_P*T_s/(2*T_I))*(e[n] + e[n-1]);
u[n] = u[n-1] + Du_P + Du_I;
% PID incremental com anti-windup (v.2)
if (u[n] ≥ sat)
u[n] = sat;
elseif (u[n] ≤ - sat)
u[n] = -sat;
end
```

4.3 Atividades Práticas

Efetue as ligações do servomecanismo com a fonte de alimentação e com o sistema de aquisição.

Analise o código do arquivo `prog4.m` e utilize-o como base para elaboração das atividades propostas.

a) Assuma $K_t = 0,017$. Aplique um degrau de 3V na entrada da planta. Identifique T e K.

b) Projete um controlador PI en tempo contínuo, tal que a resposta em malha fechada seja um sistema de primeira ordem com constante de tempo $\tau = 0,8T$.

c) Implemente o controlador PI da Equação (4.7), discretizando o controlador contínuo do item b) com período de amostragem $T_s = 1/20$ s. Rode o experimento por 10 s. Como entrada, considere o degrau de amplitude 3V aplicado em 1 segundo.

d) Implemente o controlador PI incremental da Equação (4.9) que foi projetado no item b). Considere o período de amostragem $T_s = 1/20$ segundos. Rode o experimento por 10 s. Como entrada, considere o degrau de amplitude 3V aplicado em 1 s.

e) Implemente o controlador PI projetado no domínio-z apresentado na Equação (4.20). Considere o período de amostragem $T_s = 1/20$ s. Rode o experimento por 10 segundos. Como entrada, considere o degrau de amplitude 3V aplicado em 1 s.

d) Considere uma das formas incrementais previamente projetadas. Rode o experimento por 30 segundos. Como entrada, considere o degrau de amplitude 3,5V aplicado em 1 s. Em $t = 5$ s, abaixe o freio e mantenha abaixado até $t = 10$ s. Verifiquei efeito do *windup*. Aplique a técnica *anti-windup*, execute o experimento novamente e compare como o resultado sem *anti-windup*.

NÃO SE ESQUEÇA: Mantenha cópias de segurança de seus arquivos!

Créditos

Esta experiência foi desenvolvida e/ou atualizada pelos seguintes professores:

- Bruno Augusto Angélico
- Ricardo Paulino Marques

5

Projeto de controladores digitais por alocação de polos utilizando variáveis de estado

5.1 Objetivo

O objetivo desta experiência é, utilizando o enfoque de espaço de estados, projetar e implementar um controlador digital para a malha de posição do servomecanismo (veja [Franklin, Powell e Workman 2006]). A realimentação de estados será feita, primeiramente, medindo-se diretamente os estados, e, em seguida, via estimador de estados.

5.2 Introdução teórica

5.2.1 Modelagem por espaço de estados

A modelagem por espaço de estados possui as seguintes vantagens:

- Equações de estado fornecem um modelo matemático de grande generalidade;
- A notação matricial compacta facilita muito manipulações complexas;
- Pode descrever sistemas lineares e também **não-lineares**;
- Pode descrever sistemas invariantes no tempo e também **variantes no tempo**;
- Adequada para sistemas de múltiplas entradas e múltiplas saídas (MIMO);
- Há técnicas computacionais para solução das equações;
- Possibilita o projeto de controladores usando alocação de polos e outras técnicas mais avançadas;

Considere um sistema SISO linear e invariante no tempo. O mesmo pode ser representado da seguinte forma

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u \\ y &= \mathbf{C}\mathbf{x} + \mathbf{D}u \end{aligned} \tag{5.1}$$

onde,

$\dot{\mathbf{x}} =$ vetor de estado (vetor n);

$u =$ sinal de controle (escalar);

$y =$ sinal de saída (escalar);

$\mathbf{A} =$ matriz $n \times n$ denominada matriz de estados;

$\mathbf{B} =$ matriz $n \times 1$ denominada matriz de entrada;

$\mathbf{C} =$ matriz $1 \times n$ denominada matriz de saída;

$\mathbf{D} =$ matriz 1×1 denominada matriz de transmissão direta.

Os autovalores da matriz $\mathbf{A}$ representam os polos do sistema.

5.2.2 Controlabilidade e Observabilidade

Os conceitos de controlabilidade e observabilidade foram introduzidos por Kalman e possuem um papel importante no projeto de sistemas de controle no espaço de estados.

- **Controlabilidade**: um sistema será dito controlável no instante t_0 se existir uma entrada (vetor de controle) capaz de transferir o sistema de qualquer estado inicial $\mathbf{x}(t_0)$ para qualquer outro estado, em um intervalo de tempo finito.

Considere o sistema descrito pela seguinte equação de estado

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u, \tag{5.2}$$

Tal sistema será dito de estado controlável em $t = t_0$ se houver um sinal de controle que transfira o sistema de um estado inicial par qualquer estado final, em um intervalo de tempo finito $t_0 \leq t \leq t_1$. Se todo estado for controlável, então o sistema será considerado **completamente controlável**. Para o sistema ser completamente controlável, a matriz

$$\mathcal{C} = \left[\begin{array}{c|c|c|c} \mathbf{B} & \mathbf{AB} & \cdots & \mathbf{A}^{n-1}\mathbf{B} \end{array} \right], \tag{5.3}$$

deve possuir posto (*rank*) n, ou seja, se as colunas de $\mathcal{C}$ forem linearmente independentes. Tal matriz é denominada **matriz de controlabilidade**.

- **Observabilidade**: Um sistema será dito observável no instante t_0 se, com o sistema no estado $\mathbf{x}(t_0)$, for possível determinar esse estado a partir da observação da saída durante um intervalo de tempo finito.

O sistema sem excitação é suficiente para determinar a condição de observabilidade. Considere o sistema descrito por

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} \tag{5.4}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} \tag{5.5}$$

O sistema será dito completamente observável se todo estado $\mathbf{x}(t_0)$ puder ser determinado pela observação de $\mathbf{y}(t)$ durante um intervalo de tempo finito, $t_0 \leq t \leq t_1$. Pode-se mostrar que o posto da matriz $nm \times n$

$$\mathcal{O} = \left[\begin{array}{c} \mathbf{C} \\ \hline \mathbf{CA} \\ \hline \vdots \\ \hline \mathbf{CA}^{n-1} \end{array}\right], \tag{5.6}$$

deve ser n, ou seja, $\mathrm{rank}(\mathcal{O}) = n$. Tal matriz denomina-se **matriz de observabilidade**.

5.2.3 Obtenção do Equivalente Discreto

O equivalente discreto do sistema descrito em (5.1) é dado por

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{\Phi}\mathbf{x}[k] + \mathbf{\Gamma}u[k] \\ y &= \mathbf{C}\mathbf{x}[k] + \mathbf{D}u[k] \end{aligned} \tag{5.7}$$

onde

$$\mathbf{\Phi} = e^{\mathbf{A}T} \tag{5.8}$$

$$\mathbf{\Gamma} = \int_0^\infty e^{\mathbf{A}\eta}\, d\eta\, \mathbf{B} \tag{5.9}$$

O comando `c2d` do MATLAB efetua tal conversão.

É importante observar que os conceitos de controlabilidade, observabilidade para sistemas discretos são idênticos ao caso contínuo.

5.2.4 Controle por Realimentação de Estados

Considere como exemplo o sistema a controlar representado no espaço de estados por:

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{\Phi}\mathbf{x}[k] + \mathbf{\Gamma}u[k] \\ y &= \mathbf{C}\mathbf{x}[k] \end{aligned} \tag{5.10}$$

Ao invés de realimentar a saída y, que tal utilizar a retroação das variáveis de estado?

Se cada uma das variáveis de estado for empregada no controle através de um ganho k_c^i, haverá n ganhos k_c^i, representados pelo vetor $\mathbf{K}_c$ que podem ser ajustados para produzir os valores desejados dos polos de malha fechada.

Com a realimentação de estados (e considerando $r[k] = 0$), tem-se que:

$$\mathbf{x}[k+1] = \mathbf{\Phi}\mathbf{x}[k] - \mathbf{\Gamma}\mathbf{K}_c\mathbf{x}[k] \tag{5.11}$$

Ao aplicar a transformada Z, tem-se

$$\left(z\mathbf{I} - \mathbf{\Phi} + \mathbf{\Gamma}\mathbf{K}_c\right) X(z) = 0 \tag{5.12}$$

A Figura 5.1 apresenta a representação em espaço de estados bem como o sistema com realimentação de estados.

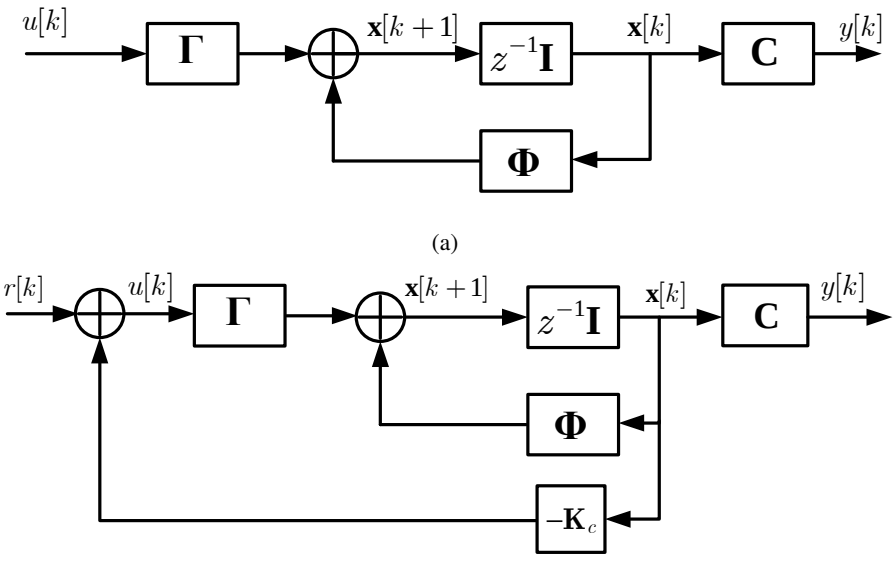

Figura 5.1: (a) Representação no espaço de estados do sistema a ser controlado; (b) realimentação de estados.

A equação característica do sistema descrito em (5.11) é dada por

$$\det\left(z\mathbf{I} - \mathbf{\Phi} + \mathbf{\Gamma}\mathbf{K}_c\right) = 0 \tag{5.13}$$

Se a posição desejada dos polos de malha fechada é conhecida, tal que

$$p_c(z) = (z - p_1) \cdot (z - p_2) \cdots (z - p_n) = 0, \tag{5.14}$$

então, o vetor $\mathbf{K_c}$ pode ser obtido como

$$\det\left(z\mathbf{I} - \mathbf{\Phi} + \mathbf{\Gamma}\mathbf{K}_c\right) = (z - p_1) \cdot (z - p_2) \cdots (z - p_n) \tag{5.15}$$

Se o sistema dinâmico $\mathbf{x}[k+1] = \mathbf{\Phi}\mathbf{x}[k] + \mathbf{\Gamma}u[k]$ é completamente controlável, então existe $\mathbf{K}_c = \begin{bmatrix} k_c^1 & k_c^2 & \cdots & k_c^n \end{bmatrix}$, tal que $\det\left(z\mathbf{I} - \mathbf{\Phi} + \mathbf{\Gamma}\mathbf{K}_c\right) = p_c(z)$ para qualquer polinômio $p_c(z)$ de grau n especificado.

Uma forma de se encontrar $\mathbf{K}_c$ é utilizando a fórmula de Ackermann. No MATLAB há a função `acker.m`, com a seguinte sintaxe `K_c = acker(Φ,Γ,p_c)`, sendo `Φ`, `Γ` as matrizes de estado e de saída, e `p_c` o vetor com a posição desejada dos polos. O comando `place`, com a mesma sintaxe, também pode ser utilizado.

5.2.5 Projeto de Observadores

O controlador necessita de acesso às variáveis de estado para realizar a realimentação com os ganhos ajustados. O acesso a tais variáveis pode ser feito através de sensores. Entretanto, em alguns casos, a medição de estados é impraticável por motivos como custo do sensor, inviabilidade física ou mesmo pelo fato de um dado estado não ser explicitamente disponível para medição.

Há, basicamente, duas formas de se estimar o estado $\mathbf{x}[k]$:

- estimativa atual, onde a estimativa $\hat{\mathbf{x}}[k]$ é obtida com base em medições de $y[k]$ passadas e presente;
- estimativa de predição, onde a estimativa $\hat{\mathbf{x}}[k]$ é obtida com base em medições passadas da saída até $y[k-1]$.

Na impossibilidade de realimentar os estados reais, a ideia é fazer $u = -\mathbf{K}_c\hat{\mathbf{x}}$.

Considere novamente a seguinte representação

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{\Phi}\mathbf{x}[k] + \mathbf{\Gamma}u[k] \\ y &= \mathbf{C}\mathbf{x}[k] \end{aligned} \tag{5.16}$$

Apesar de se conhecer $\mathbf{\Phi}$, $\mathbf{\Gamma}$ e $\mathbf{C}$, algumas das variáveis de estado podem ser inacessíveis. Assim, as estimativas de estado são obtidas através de um modelo do sistema, tal como representado na Figura 5.2

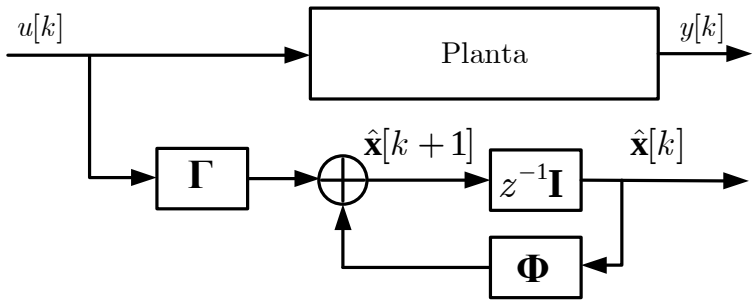

Figura 5.2: Diagrama do estimador em malha aberta.

Se as condições iniciais do modelo e da planta são as mesmas, o comportamento da variável real e o da estimada são os mesmos. Se isso não for verdade, haverá um erro de estimação.

Para corrigir o erro de estimação, pode-se utilizar uma realimentação deste erro, como mostrado na Figura 5.3.

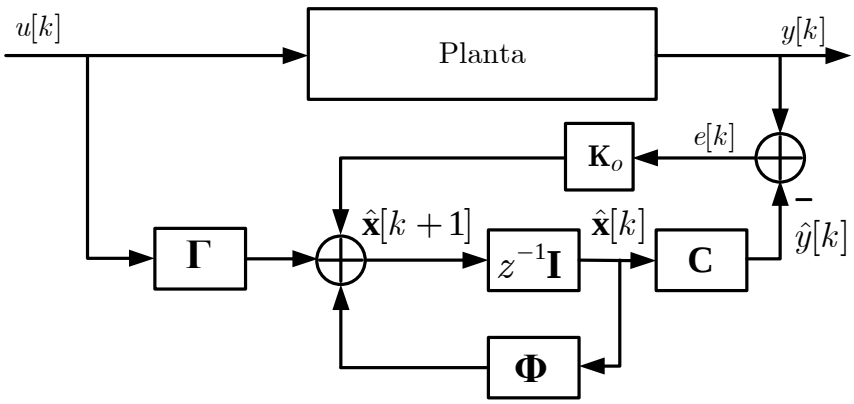

Figura 5.3: Diagrama do estimador em malha fechada.

A equação do estimador é então dada por:

$$\begin{aligned}\hat{\mathbf{x}}[k+1] &= \mathbf{\Phi}\hat{\mathbf{x}}[k] + \mathbf{\Gamma}u[k] + \mathbf{K}_o(y[k] - \hat{y}[k]) \\ &= \mathbf{\Phi}\hat{\mathbf{x}}[k] + \mathbf{\Gamma}u[k] + \mathbf{K}_o\mathbf{C}(\mathbf{x}[k] - \hat{\mathbf{x}}[k])\end{aligned} \tag{5.17}$$

onde $\mathbf{K}_o = \begin{bmatrix} k_o^1 & k_o^2 & \cdots & k_o^n \end{bmatrix}^\top$ é o vetor que representa os ganhos do estimador. Ao definir $\tilde{\mathbf{x}}[k] = \mathbf{x}[k] - \hat{\mathbf{x}}[k]$, tem-se

$$\begin{aligned}\tilde{\mathbf{x}}[k+1] &= \mathbf{x}[k+1] - \hat{\mathbf{x}}[k+1] = \mathbf{\Phi}(\mathbf{x}[k] - \hat{\mathbf{x}}[k]) - \mathbf{K}_o\mathbf{C}\tilde{x}[k] \\ &= (\mathbf{\Phi} - \mathbf{K}_o\mathbf{C})\tilde{x}[k]\end{aligned} \tag{5.18}$$

Se o sistema for completamente observável, pode-se dar a $\tilde{\mathbf{x}}[k]$ o desempenho desejado, ou seja, existe $\mathbf{K}_o$, tal que $\det(z\mathbf{I} - \mathbf{\Phi} + \mathbf{K}_o\mathbf{C}) = p_o(z)$ para qualquer polinômio $p_o(z)$ de grau n especificado.

Em geral, deseja-se que $\tilde{\mathbf{x}}[k]$ convirja rapidamente para zero. Note que o erro de estimação depende dos autovalores de $(\mathbf{\Phi} - \mathbf{K}_o\mathbf{C})$. Portanto, se estes autovalores são estáveis, o erro tende a zero assintoticamente. Por este motivo, tal estimador é denominado **Estimador Assintótico de Estados**.

Note que o projeto do observador consiste em resolver o problema dual do controlador, ou seja, resolver o problema de alocação de polos para o sistema

$$\begin{aligned} \mathbf{z}[k+1] &= \mathbf{\Phi}^\top \mathbf{z}[k] + \mathbf{C}^\top v[k] \\ q &= \mathbf{\Gamma}\mathbf{z}[k], \end{aligned} \tag{5.19}$$

pois os autovalores de $(\mathbf{\Phi}^\top - \mathbf{C}^\top\mathbf{K}_o^\top)$ são os mesmos de $(\mathbf{\Phi} - \mathbf{K}_o\mathbf{C})$. Assim como no problema de alocação de polos, pode-se utilizar a fórmula de Ackermann para encontrar o ganho do estimador. No MATLAB, tem-se a seguinte sintaxe `K_o = acker(Φ',C',p_o)'`, sendo `Φ`, `Γ` as matrizes de estado e de saída, e `p_o` o vetor com a posição dos polos desejada. O comando `place`, com a mesma sintaxe, também pode ser utilizado. Note que os polos do observador devem ser escolhidos de forma a serem mais rápidos que os polos do sistema em malha fechada.

5.3 Inserção de Integrador

Até agora consideramos o problema de controle regulatório, ou seja $r = 0$. Em muitos casos, o objetivo do controle, além de estabilizar o sistema, é seguir uma dada referência.

Para o rastreamento da referência ocorrer adequadamente, muitas vezes é necessária a inserção de integradores na malha de controle. Uma forma de inserir um integrador consiste em introduzir um novo vetor de estados que integre o erro entre o vetor de saída $\mathbf{y}$ e o vetor de comando de entrada $\mathbf{r}$, ambos com m elementos, como apresentado no diagrama da Figura 5.4.

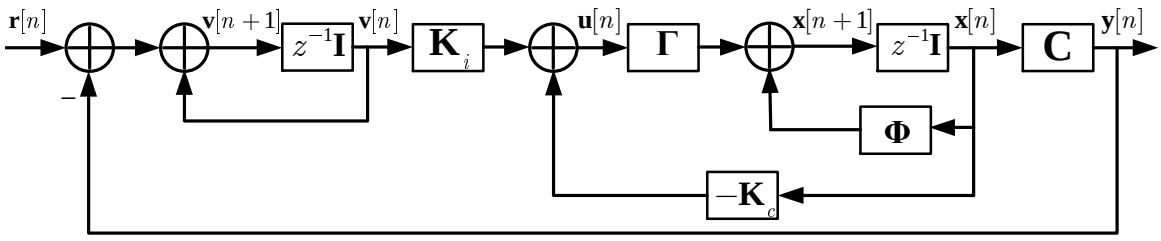

Figura 5.4: Servossistema com realimentação de estados e controle integral.

A equação de estados do integrador inserido é dada por

$$\mathbf{v}[n+1] = \mathbf{v}[n] + \mathbf{r}[n] - \mathbf{y}[n] \Rightarrow \mathbf{v}[n+1] = \mathbf{v}[n] + \mathbf{r}[n] - \mathbf{C}\mathbf{x}[n] \tag{5.20}$$

A equação de estados do sistema em malha fechada é dada por:

$$\mathbf{x}[n+1] = (\mathbf{\Phi} - \mathbf{\Gamma}\mathbf{K})\,\mathbf{x}[n] + \mathbf{\Gamma}\mathbf{K}_i\mathbf{v}[n] \tag{5.21}$$

Tem-se, portanto, a seguinte equação para o sistema aumentado:

$$\begin{bmatrix} \mathbf{x}[n+1] \\ \mathbf{v}[n+1] \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi} - \mathbf{\Gamma K} & \mathbf{\Gamma K}_i \\ -\mathbf{C} & \mathbf{I}_{m\times m} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}[n] \\ \mathbf{v}[n] \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{I}_{m\times m} \end{bmatrix} \cdot \mathbf{r}[n], \tag{5.22}$$

ou seja,

$$\begin{bmatrix} \mathbf{x}[n+1] \\ \mathbf{v}[n+1] \end{bmatrix} = \left(\underbrace{\begin{bmatrix} \mathbf{\Phi} & \mathbf{0}_{k\times m} \\ -\mathbf{C} & \mathbf{I}_{m\times m} \end{bmatrix}}_{\hat{\mathbf{\Phi}}} - \underbrace{\begin{bmatrix} \mathbf{\Gamma} \\ \mathbf{0}_{m\times m} \end{bmatrix}}_{\hat{\mathbf{\Gamma}}} \cdot \underbrace{\begin{bmatrix} \mathbf{K} & -\mathbf{K}_i \end{bmatrix}}_{\hat{\mathbf{K}}} \right) \cdot \begin{bmatrix} \mathbf{x}[n] \\ \mathbf{v}[n] \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{k\times m} \\ \mathbf{I}_{m\times m} \end{bmatrix} \cdot \mathbf{r}[n] \tag{5.23}$$

Assim, basta determinar o ganho $\hat{\mathbf{K}}$ que aloca os polos para o sistema aumentado da Equação (5.23).

5.4 Atividades Prévias

A planta a controlar é descrita pela seguinte função de transferência:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{KK_p n^2}{s(Ts+1)} \tag{5.24}$$

Considere $Kp = 1,7$, $n = 1/3$ e os valores de K e T identificados na Experiência 5.

1. Encontre uma representação em espaço de estados para o servomecanismo, considerando como variáveis de estado a a posição angular no eixo do potenciômetro ($x_1 = \theta_p$) e a velocidade no eixo do potenciômetro ($x_2 = \omega_p$). Verifique se o sistema é controlável e observável.

2. Obtenha o equivalente discreto para $T_s = 1/20$ s.

3. Projete um controlador por realimentação de estados com integrador, alocando os polos em malha fechada do sistema com a seguinte especificação: $\zeta = 0,707$ e $\omega_n = 10$ rad/s. Escolha o polo do integrador em $s = -10$.

4. Projete um observador de estados para a planta não aumentada , considerando a seguinte especificação para os polos do observador: $\zeta = 0,707$ e $\omega_n = 30$ rad/s.

5.5 Atividades Práticas

Efetue as ligações do servomecanismo com a fonte de alimentação e com o sistema de aquisição.

Analise o código do arquivo `prog4.m` e utilize-o como base para elaboração das atividades propostas.

a) Implemente o controle por realimentação de estados e integrador considerando velocidade e posição medidas. Note que os estados são θ_p e ω_p. Assuma como referência para θ_p uma onda quadrada com amplitude de $-\pi/2$ a $+\pi/2$, com frequência igual a $0,1$ Hz.

b) Implemente o controle por realimentação de estados e integrador considerando os estados obtidos a partir do observador projetado. Assuma como referência para θ_p uma onda quadrada com amplitude de $-\pi/2$ a $+\pi/2$, com frequência igual a $0,1$ Hz.

NÃO SE ESQUEÇA: Mantenha cópias de segurança de seus arquivos!

Créditos

Esta experiência foi desenvolvida e/ou atualizada pelos seguintes professores:

- Bruno Augusto Angélico
- Ricardo Paulino Marques

Referências Bibliográficas

[Allen-Bradley 2009]ALLEN-BRADLEY. *Compact I/O Analog Output Module*. USA: Rockwell Automation Publication, 2009.

[Allen-Bradley 2016]ALLEN-BRADLEY. *Programming Manual - Logix5000 Controllers Sequential Function Charts*. USA: Rockwell Automation Publication, 2016.

[Franklin, Powell e Workman 2006]FRANKLIN, G. F.; POWELL, J. D.; WORKMAN, M. L. *Digital Control of Dynamic Systems*. 3rd ed. (reprint). ed. [S.l.]: Ellis-Kagle Press, 2006.

[Georgini 2006]GEORGINI, M. *Automação aplicada: descrição e implementação de sistemas seqüenciais com PLCs*. 2a. ed. São Paulo, Brasil: Ed. Érica, 2006.

[Jantzen 2007]JANTZEN, J. *Foundations of fuzzy control*. West Sussex, England: John Wiley & Sons, 2007.

[Prudente 2013]PRUDENTE, F. *Automação Industrial PLC: Teoria e Aplicações*. 2a. ed. Rio de Janeiro, Brasil: LTC, 2013.

[Terano, Asai e Sugeno 1994]TERANO, T.; ASAI, K.; SUGENO, M. (Ed.). *Applied Fuzzy Systems*. San Diego, CA, USA: Academic Press Professional, Inc., 1994. ISBN 0-12-685242-1.

[Tong 1977]TONG, R. A control engineering review of fuzzy systems. *Automatica*, v. 13, n. 6, p. 559 – 569, 1977.

[Zadeh 1965]ZADEH, L. A. Fuzzy sets. *Information Control*, v. 8, p. 338–353, 1965.