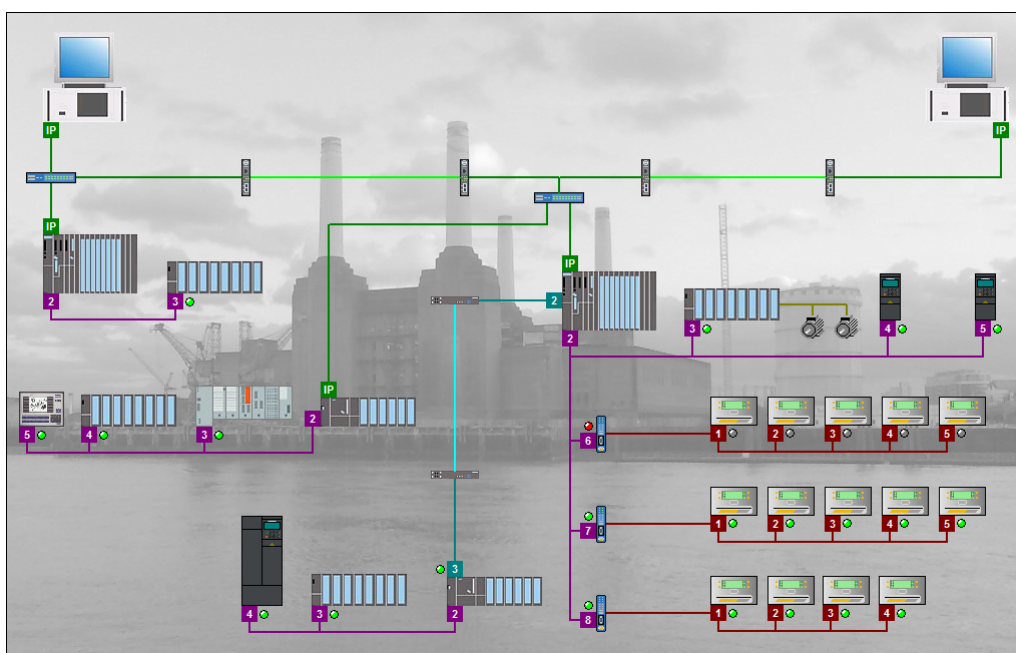




Universidade Federal de Lavras
Departamento de Automática
Engenharia de Controle e Automação

GAT141 - INFORMÁTICA INDUSTRIAL

APOSTILA (EM DESENVOLVIMENTO)



Prof. Dimitri Campos Viana

(versão 2025-10-09 – 200 mm)

Sumário

1	ORIGENS E EVOLUÇÃO DA INFORMÁTICA INDUSTRIAL	5
1.1	INTRODUÇÃO	6
1.2	UM BREVE LEVANTAMENTO HISTÓRICO	7
1.3	SISTEMAS DE AUTOMAÇÃO BASEADOS EM RELÉS ELETROMECAÑICOS	18
1.4	ROTEIRO PARA AULA PRÁTICA	27
1.5	TAREFAS PARA A PRÓXIMA SEMANA	29
1.6	EXERCÍCIOS	32
1.7	REFERÊNCIAS	37
2	CONCEITOS FUNDAMENTAIS SOBRE CLPs	39
2.1	INTRODUÇÃO	40
2.2	PRINCIPAIS CARACTERÍSTICAS DE HARDWARE	42
2.3	ACESSO À MEMÓRIA	42
2.4	ROTEIRO PARA AULA PRÁTICA	43
2.5	TAREFAS PARA A PRÓXIMA SEMANA	47
2.6	EXERCÍCIOS	49
2.7	REFERÊNCIAS	52
3	PROGRAMAÇÃO EM DIAGRAMA LADDER: INSTRUÇÕES BINÁRIAS, DECIMAIS E MISTAS	53
3.1	INTRODUÇÃO	54
3.2	INSTRUÇÕES BINÁRIAS	57
3.3	INSTRUÇÕES DECIMAIS	57
3.4	INSTRUÇÕES MISTAS	57
3.5	ROTEIRO PARA AULA PRÁTICA	58

3.6	TAREFAS PARA A PRÓXIMA SEMANA	65
3.7	EXERCÍCIOS	69
3.8	REFERÊNCIAS	74
4	PROGRAMAÇÃO EM DIAGRAMA LADDER: SUB-ROTINAS E FUNÇÕES	75
4.1	INTRODUÇÃO	75
4.2	SUB-ROTINAS	75
4.3	FUNÇÕES	75
4.4	ROTEIRO PARA AULA PRÁTICA	76
4.5	TAREFAS PARA A PRÓXIMA SEMANA	82
4.6	EXERCÍCIOS	86
4.7	REFERÊNCIAS	89
	ENCERRAMENTO DA PRIMEIRA ETAPA	90
	RESPOSTAS DOS EXERCÍCIOS	91
	ROTEIRO PARA APRESENTAÇÃO DO TRABALHO PRÁTICO	92
5	PROGRAMAÇÃO EM DIAGRAMA LADDER: CONTADORES E TEMPORIZADORES	93
5.1	INTRODUÇÃO	94
5.2	CONTADORES	94
5.3	TEMPORIZADORES	94
5.4	ROTEIRO PARA AULA PRÁTICA	95
5.5	TAREFAS PARA A PRÓXIMA SEMANA	97
5.6	EXERCÍCIOS	99
5.7	REFERÊNCIAS	102
6	DESENVOLVIMENTO DE SSCs: TAGS, TELAS E ADMINISTRAÇÃO DE USUÁRIOS	103
6.1	INTRODUÇÃO	104
6.2	GERENCIAMENTO DE TAGS	104
6.3	DESENVOLVIMENTO DE TELAS	104
6.4	ADMINISTRAÇÃO DE USUÁRIOS	104
6.5	ROTEIRO PARA AULA PRÁTICA	105
6.6	TAREFAS PARA A PRÓXIMA SEMANA	112

6.7	EXERCÍCIOS	114
6.8	REFERÊNCIAS	115
7	DESENVOLVIMENTO DE SSCs: SCRIPTS, ALARMES E CURVAS DE TENDÊNCIA	116
7.1	INTRODUÇÃO	117
7.2	USO DE SCRIPTS	117
7.3	REGISTRO E APRESENTAÇÃO DE ALARMES	117
7.4	CURVAS DE TENDÊNCIA	117
7.5	ROTEIRO PARA AULA PRÁTICA	118
7.6	TAREFAS PARA A PRÓXIMA SEMANA	125
7.7	EXERCÍCIOS	127
7.8	REFERÊNCIAS	132
8	CONCEITOS FUNDAMENTAIS SOBRE OPC	133
8.1	INTRODUÇÃO	134
8.2	MOTIVAÇÕES PARA O SURGIMENTO	134
8.3	PRINCÍPIO DE FUNCIONAMENTO	134
8.4	TECNOLOGIAS OPC	134
8.5	CONSIDERAÇÕES FINAIS	134
8.6	ROTEIRO PARA AULA PRÁTICA	135
8.7	TAREFAS PARA A PRÓXIMA SEMANA	141
8.8	EXERCÍCIOS	146
8.9	REFERÊNCIAS	149
	ENCERRAMENTO DA SEGUNDA ETAPA	150
	RESPOSTAS DOS EXERCÍCIOS	151
	ROTEIRO PARA APRESENTAÇÃO DO TRABALHO PRÁTICO	152

Capítulo 1

ORIGENS E EVOLUÇÃO DA INFORMÁTICA INDUSTRIAL

*Dimitri Campos Viana, Fernanda Costa e Silva e
Fernando Castro Alves de Souza*

1.1 INTRODUÇÃO

O termo “Informática Industrial” é utilizado para designar a área da engenharia que trata, entre outros aspectos, da configuração e programação dos Controladores Lógico-Programáveis (CLPs) e do desenvolvimento dos Sistemas de Supervisão e Controle (SSCs). Estes dois assuntos serão tratados ao longo dos próximos capítulos e, com o objetivo de fornecer um embasamento histórico para tal, as seções a seguir abordam as origens e a evolução dessas tecnologias.

Atualmente, pode-se dizer que os CLPs são tão importantes para os processos industriais como os computadores convencionais são para as pessoas. Aliás, uma boa forma de explicar o que é um CLP é comparando-o com um PC tradicional: ambos são dispositivos microprocessados, possuem memória retentiva (disco rígido ou de estado sólido), memória volátil e, muito provavelmente, capacidade de comunicação em rede. Por outro lado, as maiores diferenças estão relacionadas à interação com o mundo real: enquanto os PCs possuem periféricos como teclado, *mouse* e monitor, permitindo, por exemplo, a elaboração de um texto; os CLPs possuem terminais elétricos de entrada e saída, permitindo, por exemplo, a recepção de sinais elétricos de medidores de temperatura, pressão e nível e também o envio de sinais elétricos para motores, válvulas e outros tipos de atuador.

Assim como os PCs convencionais, que evoluíram a partir de dispositivos grandes e caros para computadores de mesa e *notebooks* e, posteriormente, se diversificaram para *smartphones*, *tablets* e outras formas, os CLPs também evoluíram a partir de projetos que surgiram na década de 1960. Atualmente, é possível encontrar uma grande variedade de modelos desse dispositivo, capazes de atender processos industriais de pequeno, médio e grande porte.

Por sua vez, os SSCs podem ser entendidos, basicamente, como um conjunto de interfaces de operação para processos industriais, ou seja, eles proporcionam meios para que os seres humanos possam aplicar suas decisões e coletar informações sobre esses processos. Sob este ponto de vista, o interruptor usado para acender e apagar a lâmpada de uma sala, assim como o velocímetro de um carro também podem ser considerados como interfaces de operação. Feita esta observação, não é difícil imaginar o motivo da falta de registros históricos relacionados ao uso de elementos como botoeiras, chaves seletoras, lâmpadas, sirenes e indicadores de grandezas físicas (pressão, vazão e temperatura) nos processos industriais: simplesmente porque, de forma natural, esses elementos sempre fizeram parte de seus sistemas de automação, desde o início.

Por outro lado, um marco notório, foi a virtualização dos mencionados elementos. Em meados da década de 1980, cresceu a aplicação dos PCs com arquitetura convencional nos sistemas de automação industrial, pois estes começaram a ser utilizados para ler e escrever valores na memória dos CLPs, por meio de comunicações estabelecidas via rede de dados. Dessa forma, botões virtuais começaram a substituir botoeiras elétricas para ações como ligar um motor ou abrir uma válvula e, ao mesmo tempo, apresentações de valores numéricos em tela começaram a substituir os indicadores físicos de pressão, vazão, temperatura etc. Atualmente, os SSCs industriais continuam baseados em PCs, mas por meio da arquitetura cliente/servidor, podem estender seus recursos de interface a outros PCs e a dispositivos móveis, via redes locais ou *web*.

1.2 UM BREVE LEVANTAMENTO HISTÓRICO

REVOLUÇÃO INDUSTRIAL

Iniciada na Inglaterra, por volta de 1760, a Revolução Industrial apresentou como uma de suas principais características a mecanização dos processos produtivos, que até aquele momento eram baseados em atividades puramente manuais (Goeking, 2010). Essa modernização ocorreu por meio de máquinas capazes de substituir operários que realizavam tarefas repetitivas, principalmente na indústria têxtil. No entanto, essas máquinas possuíam uma vida útil relativamente curta pois, além dos desgastes mecânicos, eram projetadas de forma muito específica para uma determinada tarefa. Assim, a grosso modo, pequenas mudanças em um processo produtivo exigiam a construção de novas máquinas (Franchi; Camargo, 2008, p. 21).

Além disso, mesmo com a mecanização, sob o ponto de vista atual, não se pode considerar que os processos daquela época se tornaram completamente automáticos, pois, salvo possíveis exceções, a grande maioria das decisões ainda eram humanas.

Essa realidade começou a mudar em 1788, quando James Watt projetou seu primeiro regulador de pressão para motores a vapor (na verdade, James Watt nunca se declarou como inventor desse sistema, pois sua proposta se baseou no governador centrífugo de Christiaan Huygens, projetado no século XVII, para controlar a distância entre pedras de moagem em moinhos de vento) (Willems; Polderman, 2013, p. viii). Devido a esta tecnologia, os motores a vapor de James Watt se tornaram um padrão nas indústrias, pois dispensavam a intervenção humana para manter sua velocidade constante, mesmo nas situações em que a carga ou a alimentação de vapor se alteravam. O princípio de funcionamento pode ser observado na Figura 1.1, sendo que, quanto maior é a velocidade angular do governador, mais elevadas se posicionam as esferas que, por meio de hastes e outras partes móveis, reduzem a abertura da válvula que controla a entrada de vapor no motor.

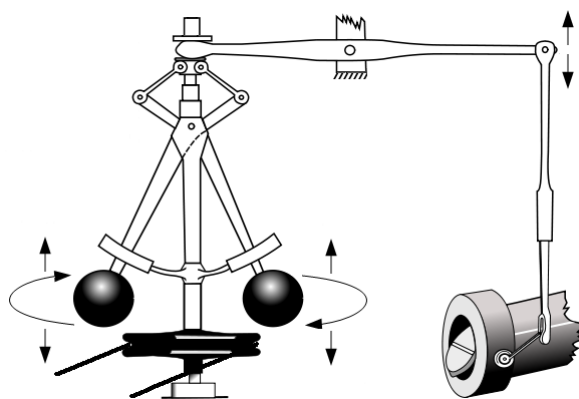


Figura 1.1: Representação esquemática do governador centrífugo de James Watt.

Outro avanço notório, que contribuiu significativamente para o desenvolvimento dos sistemas de automação, ocorreu em 1801, com o tear proposto pelo francês Joseph Marie Jacquard. Por meio de cartões perfurados e intercambiáveis, a mesma máquina permitia a produção de tecidos com diferentes padrões de entrelaçamento (Leite, 2006, p. 3). Na Figura 1.2 pode ser observado um exemplar desse tear, construído em 1910 por B. W. Archer.



Figura 1.2: Tear de Jacquard em exibição no Museu de Ciência e Industria de Manchester.

Várias outras contribuições científicas e tecnológicas ocorreram durante a Revolução Industrial. No entanto, para evitar que este tópico se torne muito extenso, ressalta-se apenas mais uma, que contribuiu diretamente para a Informática Industrial como a conhecemos hoje: a criação dos relés eletromecânicos. Segundo Rehag e Sartori (2009, p. 7), o relé eletromecânico foi inventado por Joseph Henry, em 1836 e, devido à sua relevância em relação ao assunto tratado neste capítulo, o funcionamento e a maneira com a qual foram (e ainda são) utilizados nos sistemas de automação industrial serão descritos de forma mais detalhada, na Seção 1.3.

Não há um consenso entre os historiadores sobre o ano em que a Revolução Industrial chegou ao seu fim. Assume-se como este marco a recessão econômica ocorrida entre o final da década de 1830 e o início da década de 1840, quando a adoção das inovações tecnológicas que marcaram o período, como as máquinas têxteis, diminuíram significativamente devido à saturação de seus mercados.

A SEGUNDA REVOLUÇÃO INDUSTRIAL

A recessão econômica que marcou o fim da primeira revolução industrial perdurou até aproximadamente 1870. Porém, neste intervalo de depressão do mercado, também ocorreram desenvolvimentos importantes, com destaque para as áreas de geração, transmissão e distribuição de energia elétrica (Engelman, 2015). Tais avanços foram a base para um novo período de desenvolvimento científico e tecnológico, que alguns autores classificam como a Segunda Revolução Industrial.

Apoiada nos mencionados desenvolvimentos tecnológicos, a eletrificação dos centros urbanos ganhou sustentação entre 1879 e 1881 (Chirnside, 1979), quando Sir Joseph Swan, inventor das primeiras lâmpadas incandescentes de uso prático, começou a produzi-las em escala suficiente para iluminar vias urbanas e prédios públicos. Este cenário rapidamente se expandiu para residências e indústrias, representando uma enorme contribuição para as várias inovações que já estavam ocorrendo nos processos de produção de aço, borracha, agentes químicos, automóveis, entre outros.

Na década de 1890, os motores elétricos, cujos princípios de funcionamento haviam sido formalizados por Coulomb há mais de um século, após passarem por aperfeiçoamentos propostos por cientistas como Faraday, Jedlik, Sturgeon, Pacinotti, Sprague e Tesla, atingiram um estágio de desenvolvimento tal que puderam ser vantajosamente utilizados em trens urbanos e máquinas industriais (Jefimenko, 2011; Guarnieri, 2018). Estes foram os componentes que faltavam para formar o embrião dos sistemas de automação que dominaram a indústrias por quase um século. Por volta de 1900, em conjunto com elementos de interface como chaves seletoras e botoeiras, os relés eletromecânicos, que já existiam há mais de 50 anos, começaram a ser interligados entre si, em larga escala (Hayden; Assante; Conway, 2014), formando lógicas que permitiam acionar e desligar os motores elétricos sem a necessidade de intervenção humana (na Seção 1.3, este assunto será abordado de forma mais detalhada).

Além disso, este período também ficou marcado pelo surgimento das técnicas modernas de gerenciamento de negócios e, principalmente, pelo modelo de produção baseado em linha de montagem, proposto por Henry Ford. Um dos exemplos que demonstra o quão revolucionário foi este método consiste no processo de fabricação do automóvel denominado “Model T”, que, em 1910, era comercializado por \$780 e, seis anos mais tarde, após adoção do processo de produção em massa, passou a ser comercializado por \$360 (Greenberg; Watts; Bucki, 2008, p. 68).

Considera-se na literatura que este período de desenvolvimento foi encerrado em 1914, com o início da Primeira Guerra Mundial e suas posteriores consequências, como a desestabilização econômica dos principais países europeus e a falta de mão-de-obra nas indústrias, decorrente da morte de milhões de militares e civis.

REVOLUÇÃO DIGITAL

Bem menos estabelecida na literatura do que as duas primeiras, a revolução digital (também conhecida como “Terceira Revolução Industrial”), está longe de ser um consenso, especialmente em relação ao seu período. A existência deste conceito está baseada na substituição da tecnologia analógica pela tecnologia digital, como a que ocorreu durante as duas últimas décadas do século passado, quando os discos de vinil e as fitas cassete deram lugar aos discos ópticos. No entanto, nos processos produtivos industriais, devido à diversidade de tecnologias coexistentes, esta transição não foi (ou não está sendo) tão clara, causando a mencionada falta de consenso. Mesmo assim, não é difícil compreender as mudanças que desencadearam essa revolução, sendo que os eventos descritos a seguir foram escolhidos para mostrar a naturalidade deste processo.

Avanços preliminares: salas de controle, pneumática e modelagem de circuitos

Apesar das catástrofes decorrentes da primeira guerra mundial, como a pandemia do vírus influenza (popularmente conhecida como gripe espanhola), que dizimou aproximadamente 4% da população mundial, e do desvio de foco do desenvolvimento científico e tecnológico causado pela corrida armamentista, os sistemas de automação industrial não pararam de evoluir.

À medida em que os sistemas baseados em relés tornavam-se mais elaborados, a interface para operá-los precisou ser, no mínimo, melhor organizada. Em vez de se usar elementos elétricos como chaves de seleção e mostradores espalhados pela planta, surgiram as primeiras salas de controle, nas quais tais elementos eram reunidos em painéis sinópticos, que colaboravam para o entendimento do processo ao representá-lo de uma forma mais intuitiva. Um exemplo desses antigos painéis pode ser observado na Figura 1.3, que mostra parte da sala de controle da Usina Elétrica de Kelenföld, inaugurada em 1914, na Hungria (Mechanical Engineers, 1995).



Figura 1.3: Trecho do painel sinóptico da sala de controle da Usina Elétrica de Kelenföld, inaugurada em 1914, na Hungria.

Em relação ao controle de grandezas físicas, uma grande contribuição foi dada por Nicolas Minorsky, em 1922. Seu objetivo era projetar um sistema de piloto automático para os navios da marinha americana e, ao observar os procedimentos adotados por um timoneiro, descreveu matematicamente suas ações, formalizando uma lei de controle baseada em três termos, conhecida hoje como controle Proporcional, Integral e Derivativo (PID) (Bennett, 1996). Em 1930, Clesson E. Mason, que trabalhava na empresa Foxboro, inventou o “Stabilog”, um controlador pneumático que implementava as ações de controle proporcional e integral (PI), sendo confiável o bastante para ser utilizado em processos industriais. Após a comprovação de sua eficiência, outros modelos surgiram no mercado, inclusive incorporando o termo derivativo, como é o caso do dispositivo que pode ser observado na Figura 1.4. O fluxo de sinais entre sensor, controlador e atuador era estabelecido por meio da variação de pressão em linhas de ar comprimido, o que acabou contribuindo para a consolidação do padrão 3.15 psi (antecessor do padrão 4.20 mA).

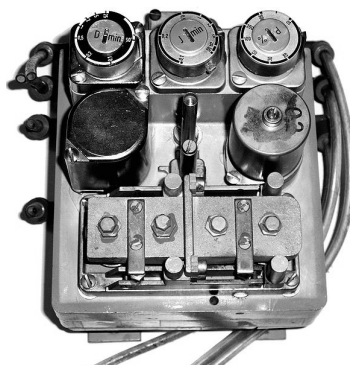


Figura 1.4: Antigo controlador pneumático industrial, capaz de formar um sinal de saída baseado nos termos proporcional, integral e derivativo.

Em 1937, Claude Shannon escreveu sua dissertação de mestrado cujo título poderia ser traduzido para algo como “Uma Análise Simbólica dos Relés e Circuitos Chaveados”, sendo que os principais conceitos também foram formalizados em seu consagrado artigo, publicado em 1938, com o mesmo título (Shannon, 1938). Neste trabalho, Shannon demonstrou que muitos circuitos formados por relés eletromecânicos podiam ser simplificados, basicamente, da seguinte maneira: os circuitos originais podem ser descritos por meio de equações booleanas; tais equações podem ser submetidas aos conhecidos métodos de simplificação; e os resultados desse processo são o guia de montagem dos novos circuitos. Posteriormente, Shannon provou que qualquer problema resolvido com álgebra booleana também poderia ser resolvido com circuitos baseados em relés.

Além da relevância para sua época, o trabalho de Shannon estabeleceu a base que permitiu, após algumas décadas, transformar os diagramas de interligação de relés, originalmente desenvolvidos em papel e conhecidos como *Diagrama Ladder*, em uma das linguagens de programação para CLPs mais populares da atualidade, de mesmo nome.

Mais alguns avanços: eletrônica analógica e dispositivos microprocessados

Por volta de 1950, dispositivos baseados em eletrônica analógica começaram a se tornar financeiramente atrativos e confiáveis o suficiente para serem utilizados nos sistemas de automação industrial. Assim, medidores e controladores de grandezas físicas baseados em princípios pneumáticos começaram a ser substituídos por dispositivos equivalentes, que usavam esta tecnologia. Consequentemente, o padrão 3..15 psi (para transmissão de sinais pneumáticos) começou a ser substituído pelo padrão 4..20 mA (para transmissão de sinais eletrônicos).

A partir da invenção do transistor, surgiram computadores digitais bem mais avançados do que os da geração anterior (Goeking, 2010). Nas décadas de 1950 e 1960, organizações militares, governamentais e grandes empresas privadas passaram a dispor de sistemas computacionais mais elaborados, baseados nesta nova tecnologia. No entanto, estes computadores ainda eram inapropriados para os sistemas de automação industrial, por serem grandes, caros e sensíveis ao ambiente da maioria dos processos produtivos (que, entre outras características, podem apresentar temperatura, umidade, distúrbios magnéticos, partículas em suspensão e vibração em níveis elevados).

Além disso, naquela época, crescia a insatisfação em relação aos sistemas de automação baseados em relés eletromecânicos. Alguns problemas eram o espaço físico que os painéis ocupavam, a dificuldade de manutenção (não era fácil descobrir um relé defeituoso entre muitos outros) e a necessidade de se manter uma documentação atualizada pois, para evitar o agravamento da dificuldade de manutenção, qualquer mudança nos circuitos precisava ser bem documentada. No entanto, talvez o problema mais grave fosse a falta de flexibilidade, uma vez que pequenas mudanças em um processo produtivo podiam implicar em mudanças significativas no circuito utilizado para automatizá-lo, consumindo tempo e recursos financeiros.

Dessa forma, como os computadores daquela época não eram a solução para substituir os painéis de relés, restou aos interessados especificar um equipamento que não existia. Em 1968, Bill Stone, que trabalhava na divisão de transmissões automáticas da *General Motors*, apresentou formalmente essa demanda no *Westinghouse Machine Tool Forum*. Entre as características requeridas, além da eliminação dos mencionados problemas a um custo competitivo, estavam:

- Capacidade para abranger 90 % das máquinas da planta;
- Programável por meio de técnicas já difundidas (*Diagrama Ladder*);
- Baseado em componentes modulares, facilitando manutenções e expansões;
- Confiabilidade apropriada para ambientes industriais.

Contudo, antes de tomar conhecimento desses requisitos, Richard E. Morley, sócio-fundador da *Bedford Associates*, já havia planejado e construído o protótipo de um dispositivo capaz de atendê-los (pelo menos em parte). Após fecharem o contrato de fornecimento com a *General Motors*, a equipe de desenvolvimento da *Bedford Associates* percebeu que este equipamento era tão promissor que seus integrantes resolveram formar uma empresa dedicada a ele: a Modicon, cujo nome abreviava o termo *Modular Digital Controller*. O primeiro modelo fornecido foi o 084, que pode ser observado na Figura 1.5, rodeado por Morley e seus sócios. Deve-se observar que, ao publicar os requisitos para este dispositivo, a *General Motors* recebeu protótipos de outros fabricantes; no entanto, como o Modicon 084 havia sido previamente projetado e também foi o vencedor da concorrência, muitos o consideram como o primeiro Controlador Lógico-Programável (CLP) da história (GICSP; Assante; Conway, 2014).



Figura 1.5: O CLP Modicon 084 e os principais membros da equipe envolvida, da esquerda para a direita: Dick Morley, Tom Boissevain, George Schwenk e Jonas Landau.

Não demorou muito para a Modicon atualizar e implementar melhorias em relação ao seu primeiro CLP. Em 1975, já haviam sido lançados os modelos 184, 284 e 384. Este último modelo, o Modicon 384, merece destaque em um ponto muito importante: foi o primeiro a trabalhar com valores decimais, permitindo, por exemplo, a implementação de algoritmos de controle PID. Até então, os CLPs haviam sido projetados para substituir os painéis de relés, mas a partir deste ponto tornou-se viável substituir também os dispositivos eletrônicos dedicados ao controle de grandezas em malha fechada, que por sua vez já vinham substituindo os controladores pneumáticos (Silveira; Santos, 1998, p. 17–19).

Em paralelo ao desenvolvimento dos CLPs, os painéis sinópticos que, à princípio, proviam interface de operação aos processos industriais exclusivamente por meio de componentes eletromecânicos, também evoluíram. Em um primeiro momento, ocorreu a adoção de dispositivos eletrônicos, como *displays* de 7 segmentos e registradores analógicos de cartas de controle, usados para acompanhar a evolução de grandezas como nível, temperatura e pressão ao longo do tempo, registrando-as em papel por meio de uma agulha móvel (similar aos antigos aparelhos de eletrocardiografia). Posteriormente, um grande avanço ocorreu, quando muitos desses elementos de interface (botões, chaves seletoras, lâmpadas, indicadores e registradores de grandezas físicas) passaram a ser virtualizados em ambiente computacional. Isso começou a ocorrer mesmo antes do surgimento dos CLPs, pois as salas de controle eram ambientes mais limpos e controlados e assim, via de regra, podiam abrigar computadores tradicionais. Dessa forma, os primeiros Sistemas de Supervisão e Controle (SSCs) eram formados por esses computadores, que se comunicavam por meio de protocolos fechados e específicos com dispositivos conhecidos como Unidade Terminal Remota (UTR).

As UTRs possuem entradas e saídas para sinais elétricos discretos e contínuos, sendo capazes de coletar informações de elementos como disjuntores, termostatos, medidores de pressão e de acionar elementos como válvulas e motores. Porém, não podiam ser programadas para fazer tais acionamentos por meio de lógicas automáticas, como nos CLPs. Pode-se dizer, informalmente, que as UTRs eram apenas os olhos e as mãos dos antigos SSCs, sendo que todas as decisões ficavam a cargo de operadores humanos, que observavam dados e as aplicavam por meio dos computadores instalados nas salas de controle.

A revolução propriamente dita

Foi com o aprimoramento e a difusão das redes de dados industriais, respectivamente ocorridos nas décadas de 1970 e 1980, que houve o rearranjo ilustrado na Figura 1.6, no qual os SSCs passaram a se comunicar com os CLPs e as UTRs passaram a ser escravos desses últimos, expandindo seus pontos de entradas e saídas elétricas. Assim, estava estabelecida a base para a revolução digital dentro das indústrias!

Muitos fatos relevantes que ocorreram posteriormente também estão associados às redes de dados industriais (também conhecidas como redes de campo ou pelo termo, usado no idioma inglês, *fieldbus*). À medida em que estas se tornaram mais rápidas e mais confiáveis, houve um aumento significativo da disponibilidade dos medidores e atuadores ditos inteligentes, ou seja, com capacidade de se comunicar com os CLPs via rede. A substituição do padrão analógico 4..20 mA por protocolos de redes digitais é comparável à substituição das fitas magnéticas pelos discos ópticos. A rejeição aos ruídos eletromagnéticos constantemente presentes no ambiente industrial, atingida por meio de algoritmos de verificação e reenvio de dados (e da possível imunidade, com a adoção de cabos de fibra óptica), pode ser apontada como uma das maiores vantagens.

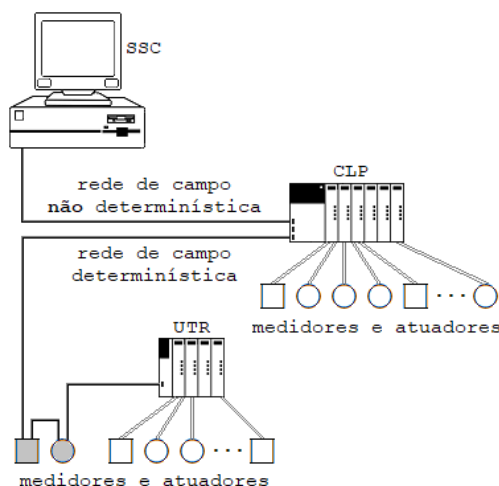


Figura 1.6: Diagrama esquemático de um típico sistema de automação industrial da década de 1980: a maioria dos medidores e atuadores interligados às entradas e saídas dos CLPs por meio de sinais elétricos discretos e do padrão 4..20 mA (e a minoria interligada via rede de campo).

Além disso, como pode ser observado na Figura 1.7, ao se colocar os instrumentos de campo em um mesmo barramento, há uma significativa redução na quantidade de cabos elétricos. Por fim, não se pode deixar de mencionar uma última vantagem: a quantidade de dados trocados entre os CLPs e esses dispositivos não estava mais limitada a um único sinal. Por exemplo, era bastante comum que um CLP estabelecesse a velocidade desejada para um motor de indução enviando um sinal de 4..20 mA para um inversor de frequência. Ao se utilizar um protocolo de rede de campo para fazer isso, o CLP também pode enviar comandos como liga, desliga e reconhecimento de falhas, assim como pode receber valores como a velocidade e a corrente elétrica atual do motor.

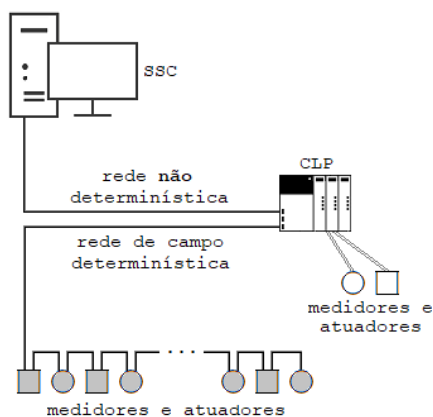


Figura 1.7: Diagrama esquemático de um típico sistema de automação industrial da década de 2010: a maioria dos medidores e atuadores são interligados aos CLPs via rede de dados (e a minoria por meio de sinais elétricos discretos e do padrão 4..20 mA).

Deve-se observar, em conclusão, que a partir do início deste século, há uma forte tendência que indica a eliminação do padrão analógico 4..20 mA nos sistemas de automação. No entanto, isso pode demorar décadas para acontecer, principalmente porque equipamentos industriais são projetados para operar sem falhas durante longos períodos de uso e assim, quando algum dispositivo apresenta defeito, muitas vezes prefere-se substituí-lo por outro similar, em vez de se modernizar o sistema.

Ainda no contexto da revolução digital, é interessante considerar como a Inteligência Artificial (IA) começou a ser aplicada e evoluiu dentro das indústrias. A princípio, as redes neurais artificiais e a lógica *fuzzy* ganharam espaço em tarefas relacionadas à estimativa e ao controle de grandezas físicas, mostrando-se ótimas alternativas à modelagem baseada em equações diferenciais, especialmente para os casos de dinâmicas complexas envolvendo parâmetros como pH, turbidez e viscosidade. Além disso, a nível de supervisão, ferramentas de IA passaram a ser usadas para prever eventos críticos nos mais diversos segmentos, como engaiolamento de carga em altos-fornos, sobrecarga em geradores de energia, transbordamento de barragens e outros. Com base em dados do passado, um sistema de segurança pode prever, em tempo hábil, que tais situações estão prestes a acontecer novamente e, dessa forma, disparar alarmes, sugerir ações aos operadores da planta ou mesmo intervir automaticamente.

Mesmo não sendo o foco deste texto, é importante notar que foi também durante a revolução digital que surgiram os sistemas de gerenciamento MES (do termo em inglês, *Manufacturing Execution System*) e ERP (*Enterprise Resource Planning*). Conforme pode-se observar na Figura 1.8, consensualmente, esses sistemas ocupam as posições mais elevadas da “pirâmide da automação industrial”. Os dois sistemas são complementares e trabalham em conjunto, sendo que o MES é uma ferramenta especializada, com foco em acompanhar e otimizar o desempenho da produção (em um horizonte temporal de horas ou turnos), enquanto que um sistema ERP é geralmente mais abrangente, envolvendo diversos setores de uma empresa, como financeiro, vendas, compras e recursos humanos (trabalhando com análises semanais, mensais e anuais).

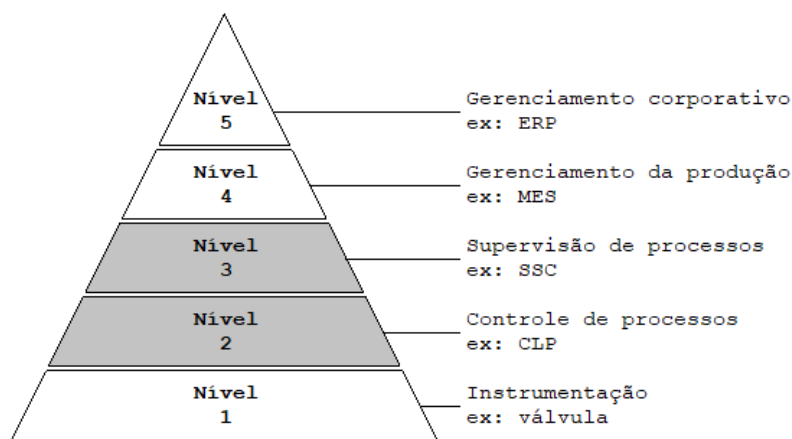


Figura 1.8: Pirâmide da automação industrial, como é conhecida, e seus cinco níveis, da base ao topo: instrumentação (sensores e atuadores), controle de processos, supervisão de processos, gerenciamento da produção, gerenciamento corporativo. Os níveis 2 e 3 foram destacados em cinza porque são o foco deste livro.

Assim como aconteceu nas camadas de controle e supervisão, os sistemas que compõem os níveis superiores da pirâmide também passaram a se beneficiar da IA. Isso foi especialmente importante, pois as ferramentas MES e ERP geralmente trabalham com um grande volume de dados a serem processados, sendo que o uso de IA pode ajudar na identificação de padrões, na indicação de ordens de produção, na gestão da cadeia de suprimentos e em vários outros pontos. Uma vez estabelecido este cenário, juntamente com a facilidade de integração entre sistemas industriais e comerciais alcançada com a maturidade da *internet*, estava pavimentado o caminho para a quarta revolução industrial.

INDÚSTRIA 4.0

O termo “Indústria 4.0”, usado em referência a uma quarta revolução industrial, tem sua origem nos documentos produzidos por um grupo de trabalho formado na Alemanha, com o objetivo de definir os investimentos do país em pesquisas e desenvolvimentos relacionados a processos de manufatura. Em 2013, este grupo de trabalho publicou seu relatório final e, desde então, nota-se um grande interesse dos profissionais envolvidos com processos industriais em compreender por completo as implicações deste conceito em curto, médio e longo prazo.

O principal fundamento deste conceito pode ser exposto fazendo-se uma analogia entre a vida cotidiana e o ambiente industrial. Com a revolução digital, entre outros exemplos, discos de vinil e fitas magnéticas foram substituídas por discos ópticos, assim como também ocorreu a substituição da tecnologia analógica na telefonia. Enquanto isso, nas indústrias, sistemas pneumáticos, eletromecânicos e baseados em eletrônica analógica foram substituídos por sistemas digitais, baseados em CLPs, SSCs e redes de campo. No entanto, note que os dispositivos relacionados a estas menções, se não trabalhavam de forma individual, se comunicavam apenas ponto-a-ponto ou em redes locais. Após a popularização da *internet*, uma nova revolução ocorreu no cotidiano das pessoas, permitindo, entre outras coisas, o acesso rápido à informação, o comércio eletrônico e o amplo uso dos *smartphones*. Porém, muito além dos aspectos pessoais, a possibilidade de comunicação irrestrita entre dispositivos e sistemas industriais também está influenciando drasticamente este setor, sendo que o impacto econômico dessa transformação é enorme, proporcionando uma maior eficácia operacional, bem como o desenvolvimento de novos modelos de negócios, serviços e produtos.

Desde que entrou em cena nas indústrias, esta era de amplas interações tem gerado novas tecnologias que estão permitindo a criação de fábricas inteligentes, conectando máquinas, sistemas e pessoas aos processos produtivos, através da integração do mundo físico e virtual. Figuram entre os benefícios a redução de custos, utilização eficiente de recursos, redução de erros, tomada de decisão otimizada e atendimento a requisitos personalizados. Em outras palavras, a quarta revolução industrial é marcada pela conectividade entre os diferentes equipamentos das linhas de manufatura, além da integração destes com setores externos.

Destaca-se um pequeno exemplo dos benefícios que a integração entre tais elementos pode gerar na indústria automobilística. Supondo que ocorra uma parada de 36 horas na linha de produção dos amortecedores usados na fabricação de um determinado modelo de automóvel, mesmo que esta linha seja terceirizada, o sistema gerencial da montadora, ao receber tal informação, pode recalcular suas estimativas e disponibilizá-las ao sistema do setor de vendas, tudo isso de forma automática. Assim, ao procurar uma concessionária, o cliente final pode ser informado com precisão sobre o prazo de entrega do veículo em que está interessado.

Sistemas mais elaborados do que o descrito no exemplo anterior também são previstos, uma vez que as informações extraídas dos processos podem ser usadas para reconfigurá-los, e assim, se a linha de amortecedores parar por algumas horas, a linha de montagem de motores pode ser automaticamente destinada a atender outros modelos, visto que não se deseja ter componentes em falta ou em excesso na etapa final da montagem. Para que sistemas como este se tornem tecnicamente e economicamente viáveis, Rüßmann *et al.* (2015) definem nove pilares, sendo eles:

- **Robôs autônomos**, capazes de interagir com outras máquinas e com os seres humanos, atuando de maneira mais flexível e colaborativa;
- **Simulação virtual**, permitindo que os processos e produtos sejam testados e passem por ensaios durante a fase de concepção, reduzindo custos com falhas e o tempo de projeto;
- **Integração de sistemas**, abrangendo toda a cadeia produtiva, por meio da análise de dados e tomada de decisão;
- **Internet das coisas**, permitindo a conectividade entre os diversos dispositivos e flexibilizando o acesso e o controle em todo o processo produtivo;
- **Segurança cibernética**, garantindo a confiabilidade da comunicação entre máquinas e sistemas gerenciais, visto que invasões representam uma ameaça ao sigilo de dados estratégicos, integridade física das máquinas e até mesmo à vida dos operários.
- **Cloud computing**, possibilitando que o acesso a banco de dados, bem como a integração de aplicações, suportes e controles possam ser realizados de qualquer localidade, eliminando também os custos com servidores locais;
- **Manufatura aditiva**, permitindo a produção através de impressoras 3D;
- **Realidade aumentada**, permitindo que o usuário atue dentro dos sistemas ciber-físicos, sendo guiados por tutores virtuais, que podem indicar passo a passo todas as ações necessárias para um reparo ou uma nova parametrização do processo, por exemplo.
- **Big data & analytics**, aumentando a eficiência dos processos e melhorando a qualidade dos produtos por meio da análise, em tempo real, de dados coletados ao longo de toda cadeia produtiva. De forma mais específica, visando a identificação de tendências e falhas.

No entanto, como o foco desta obra são os CLPs e SSCs, é importante ressaltar que a quarta revolução industrial não prevê a substituição desses elementos. Pelo contrário, como eles fazem parte do ambiente digital já estabelecido nos mais diversos processos produtivos, a tendência é que continuem desempenhando as importantes funções para as quais foram concebidos e incorporando outras, passando a trabalhar de forma conjunta e integrada com os novos sistemas que estabelecem os pilares da Indústria 4.0.

1.3 SISTEMAS DE AUTOMAÇÃO BASEADOS EM RELÉS ELETROMECHANICOS

PRINCÍPIO DE FUNCIONAMENTO DOS RELÉS

Conforme mencionado na Seção 1.2, alguns autores atribuem a invenção do relé eletromecânico à Joseph Henry, em 1836 (Rehg; Sartori, 2009, p. 7). Desde então, apesar de algumas diversificações, os relés podem ser entendidos como uma combinação de dois elementos fundamentais: bobina eletromagnética e contatos interruptores. A Figura 1.9 ilustra o dispositivo em sua proposta original, na qual são usados um contato Normal-Aberto (NA) e um contato Normal-Fechado (NF).

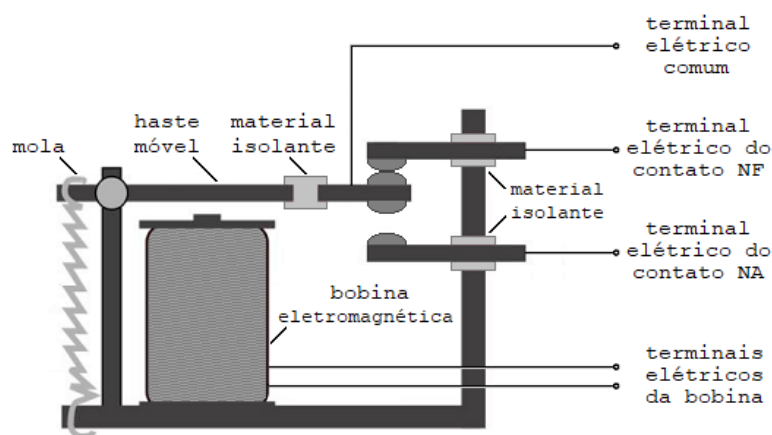


Figura 1.9: Diagrama esquemático do dispositivo proposto por Joseph Henry em 1836: contatos interruptores sofrem alteração da condição de continuidade elétrica que possuem entre si, de acordo com a energização da bobina eletromagnética.

Quando não há corrente elétrica circulando através da bobina, entende-se que o dispositivo está em repouso e, nesse caso, os terminais elétricos “NF” e “Comum” apresentam continuidade elétrica entre si, enquanto os terminais “NA” e “Comum” não apresentam. Quando a bobina é energizada, o campo magnético induzido por suas espiras é forte o suficiente para alterar a posição da haste na qual o terminal “Comum” está interligado. Assim, a situação entre os terminais elétricos se inverte, ou seja, perde-se a continuidade entre os terminais “NF” e “Comum” e obtém-se a continuidade entre os terminais “NA” e “Comum”. Quando a alimentação da bobina é interrompida, o campo magnético se extingue e uma mola faz com que a haste do terminal comum retorne à posição original, restabelecendo a situação de repouso.

Dessa forma, pode-se entender os relés como interruptores, similares aos utilizados nos sistemas de iluminação domésticos porém, em vez das manobras manuais, usa-se a bobina eletromagnética para comandar a passagem (ou a interrupção) da corrente elétrica entre os terminais dos contatos do dispositivo. Uma das grandes vantagens dessa estratégia é a independência das características da corrente elétrica utilizada para energizar a bobina em relação à corrente que pode fluir pelos contatos.

Por exemplo, existem relés em que a bobina deve ser acionada por uma tensão contínua de 24 V, sendo que seus contatos podem ser usados para acionar cargas projetadas para receber uma tensão alternada de 220 V, conforme ilustrado na Figura 1.10. Esse “desacoplamento” de características elétricas é muito importante e permite, entre outras coisas, que um CLP, que produz baixas tensões em seus terminais de saída, faça o acionamento de um motor de indução de 10 kW. Essa é a famosa separação entre circuito de comando e circuito de potência. Deve-se observar que situações análogas também são possíveis, ou seja, existem relés em que a bobina deve ser acionada por uma tensão alternada de 220 V, sendo que seus contatos podem ser usados para acionar cargas projetadas para receber uma tensão contínua de 24 V, por exemplo.

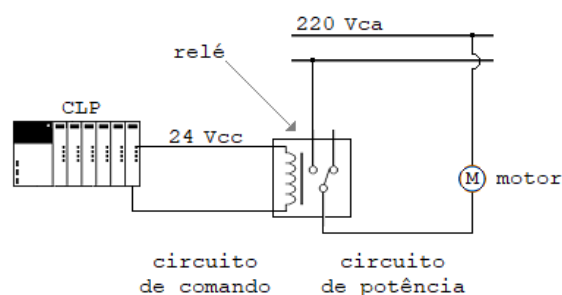


Figura 1.10: Exemplo do desacoplamento de características elétricas proporcionado por um relé eletromecânico, ao permitir que um CLP que produz 24 V_{CC} em seus terminais de saída acione um motor elétrico de 220 V_{CA}.

Além da diversificação das bobinas, que podem ser projetadas para receber correntes alternadas ou contínuas, os relés são atualmente fabricados com diferentes configurações relacionadas aos contatos interruptores, que podem ser, basicamente, de uma ou duas posições e de polo simples, duplo, triplo, etc. O termo “uma posição” é usado para designar estruturas formadas apenas por contatos NA ou apenas por contatos NF, enquanto o termo “duas posições” é usado para designar estruturas formadas por contatos NA e NF, interligados a um terminal comum. Por sua vez, conforme pode ser observado na Tabela 1.1, a quantidade de polos está associada à repetição dessas estruturas.

Tabela 1.1: Algumas das possíveis variações relacionadas aos contatos interruptores dos relés: estruturas com uma ou duas posições, que podem ser únicas ou encontradas repetidas vezes em um mesmo dispositivo.

	Uma posição (somente NA)	Uma posição (somente NF)	Uma posição (NA e NF)	Duas posições (NA e NF)
Polo simples			—	
Polo duplo				









Dessa forma, considerando-se a diversidade de características elétricas das bobinas e a vasta possibilidade de arranjos dos contatos interruptores, pode-se observar uma enorme variedade de modelos de relés eletromecânicos no mercado. Além disso, uma variação “mais radical” desses dispositivos também existe: são os relés de estado sólido que, por se basearem em componentes eletrônicos, não possuem peças móveis. A principal funcionalidade é a mesma, ou seja, permitir o desacoplamento das características elétricas entre o elemento de controle e a carga a ser acionada. Por estarem livres de desgastes mecânicos, os relés de estado sólido podem ser submetidos a chaveamentos de alta frequência o que, sem dúvida, consiste em uma grande vantagem. No entanto, como esta seção trata dos antigos sistemas de automação baseados em *relés eletromecânicos*, estes últimos continuarão em foco nos tópicos a seguir.

CIRCUITOS PARA COMANDOS MANUAIS

Elementos elétricos de interface como botoeiras, chaves seletoras, potenciômetros e lâmpadas sempre estiveram presentes nas indústrias, visto que representam uma forma simples e eficiente para que os operadores de um processo interajam com as máquinas e equipamentos que o compõe. No entanto, se um motor de indução de 380 V fosse manobrado por meio de uma chave manual que interrompe ou deixa passar a corrente necessária para sua alimentação, isso deixaria seus operadores expostos a esta elevada tensão elétrica. De forma a evitar situações perigosas como esta, os relés eletromecânicos podem ser utilizados para desacoplar as características elétricas dos elementos de interface em relação às cargas que estes acionam, ou seja, permitindo a divisão dos sistemas de acionamento em duas partes: circuitos para comandos manuais e circuito de potência.

Como alguns circuitos para comandos manuais podem envolver vários elementos de interface e vários relés, incluindo intertravamentos, é interessante que eles sejam cuidadosamente projetados, sendo que os documentos gerados nesta fase também são muito úteis para posteriores tarefas de manutenção. A Tabela 1.2 traz alguns dos símbolos mais utilizados em tais projetos.

Tabela 1.2: Alguns dos principais elementos elétricos utilizados em circuitos para comandos manuais e seus respectivos símbolos.

Elemento	Símbolo	Elemento	Símbolo
Botoeira NA		Botoeira NF	
Chave seletora de duas posições		Chave seletora de três posições	
Bobina de um relé		Lâmpada	
Contato NA de um relé		Contato NF de um relé	

Na Figura 1.11 pode ser observado um sistema para o acionamento de dois motores, composto por um circuito de comando manual e um circuito de potência. Analisando-os, é possível notar que, se o sistema estiver em repouso, um toque na botoeira *BT-101* fará com que a bobina do relé *CR-101* seja energizada. Em seguida, quando o operador deixa de pressioná-la, esta situação permanece, pois um dos contatos NA do relé *CR-101* está em paralelo à botoeira *BT-101*. Esta técnica, usada para manter um elemento energizado mesmo após o encerramento da condição que o acionou, é popularmente conhecida como “selo”. O mesmo arranjo foi utilizado com a bobina do relé *CR-102*, porém deve-se observar que a botoeira *BT-102* possui dois contatos (um NF e um NA) e, quando pressionada, ao mesmo tempo que corta a corrente que alimenta o relé *CR-101*, energiza a bobina do relé *CR-102*. Assim, para que se tenha as bobinas dos dois relés energizadas ao mesmo tempo, é necessário pressionar *BT-102* e posteriormente *BT-101*. Já a botoeira *BT-DSG*, quando pressionada, corta a corrente que alimenta as bobinas dos dois relés, desligando o sistema.

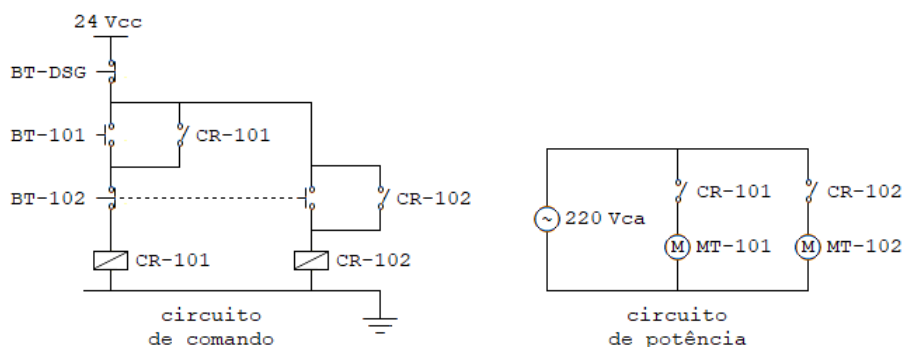


Figura 1.11: Sistema para acionamento de dois motores por meio de três botoeiras não retentivas: circuito para comandos manuais (à esquerda) e circuito de potência (à direita).

Por sua vez, a análise do circuito de potência que consta neste exemplo é bastante simples: quando os contatos NA dos relés *CR-101* e *CR-102* são fechados, os motores *MT-101* e *MT-102* são respectivamente acionados. Isto acontece quando as bobinas desses relés são energizadas no circuito para comandos manuais, conforme análise anterior.

CIRCUITOS PARA LÓGICAS AUTOMÁTICAS

Ao se combinar elementos elétricos de interface, relés eletromecânicos e elementos como chaves fim-de-curso, chaves de nível, pressostatos, termostatos etc, é possível elaborar circuitos elétricos capazes de intertravar e acionar atuadores automaticamente. Aliás, neste ponto, é interessante observar que, até o início da década de 1970, os sistemas de automação dos mais variados segmentos industriais se baseavam nesta técnica. Neste caso, os relés eletromecânicos também são utilizados para dividir os sistemas em duas partes: circuitos para lógicas automáticas e circuitos de potência.

Como os circuitos para lógicas automáticas podem se tornar extremamente complexos, o uso de uma simbologia eficiente é essencial, tanto para a concepção como para a manutenção. Neste sentido, a adoção da representação vertical dos barramentos positivo e negativo desses circuitos contribuiu de forma significativa para a preservação da clareza em diagramas muito extensos. Assim, alguns símbolos foram adaptados para esta mudança, ao passo que outros foram especialmente criados, como se vê nos exemplos fornecidos na Tabela 1.3.

Tabela 1.3: Alguns dos principais elementos utilizados em circuitos para lógicas automáticas e seus respectivos símbolos.

Elemento	Símbolo	Elemento	Símbolo
Botoeira NA		Botoeira NF	
Bobina de um relé		Chave seletora de duas posições	
Contato NA de um relé		Contato NF de um relé	
Chave de nível NA		Chave de nível NF	
Pressostato NA		Pressostato NF	
Termostato NA		Termostato NF	

A Figura 1.12 mostra uma adaptação do circuito do exemplo anterior (Figura 1.11) para este novo padrão, sendo que, além disso, as botoeiras *BT-101* e *BT-102* foram respectivamente substituídas pelos pressostatos *PS-101* e *PS-102*. Note que, com tais substituições, o funcionamento dos motores *MT-101* e *MT-102* passa a depender das pressões monitoradas e não mais, exclusivamente, de decisões humanas. Observando o circuito lógico, também é possível notar a proposta intuitiva deste tipo de diagrama: as cargas, neste caso as bobinas dos relés *CR-101* e *CR-102*, ficam próximas e conectadas ao barramento vertical direito; e os elementos que as precedem são arranjados de forma a impor condições lógicas para que a corrente possa fluir (ou não) até elas, a partir do barramento vertical esquerdo.

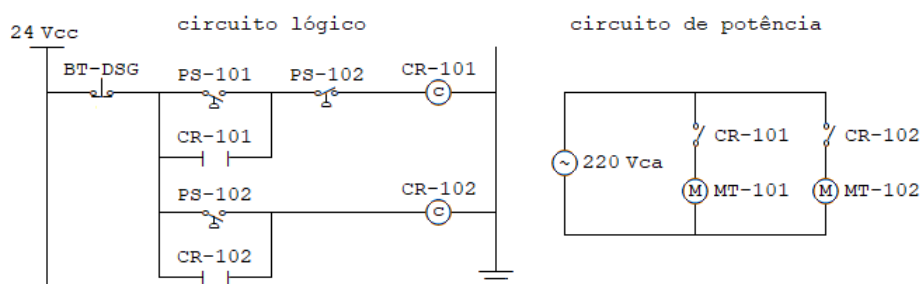


Figura 1.12: Sistema para acionamento de dois motores por meio de uma botoeira não retentiva e dois pressostatos: circuito lógico (à esquerda) e circuito de potência (à direita).

Um diagrama com várias linhas entre esses barramentos, uma embaixo da outra, ao ser observado de uma certa distância (e com um pouco de boa vontade), acaba se parecendo com o desenho de uma escada com vários degraus. Por isso, essa forma de representar a interligação de elementos elétricos ficou conhecida como *Diagrama Ladder*, uma vez que esta última palavra significa “escada”, no idioma inglês.

EXEMPLO PRÁTICO

O exemplo a seguir demonstra como os relés eletromecânicos podem ser utilizados para a obtenção de um sistema de controle *on/off*. Para que este objetivo seja alcançado, considere, inicialmente, o diagrama esquemático no qual o panorama geral do sistema é representado (Figura 1.13). Em complemento, considere as características de cada item da lista de material que pode ser observada na Tabela 1.4 e, por fim, as seguintes informações:

- A vazão de entrada do tanque pode ser considerada constante;
- Quando a válvula está aberta, a vazão de saída é maior do que a vazão de entrada.

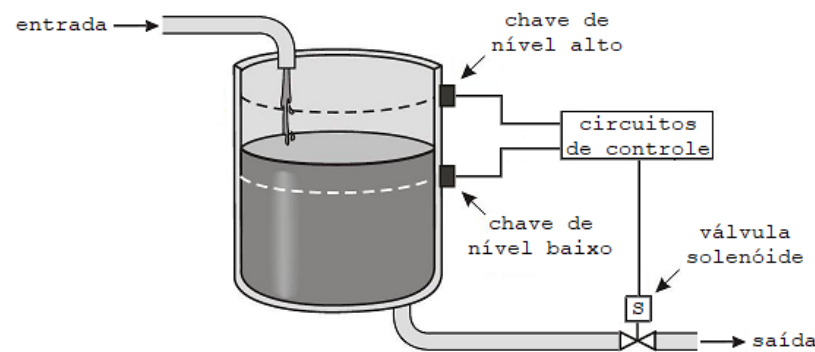


Figura 1.13: Tanque de água drenado por meio de uma válvula solenóide, com vazão de entrada constante e chaves para detecção das condições de nível alto e nível baixo.

Tabela 1.4: Lista do material disponível para elaboração do sistema de controle *on/off*, incluindo as principais características de cada item.

Descrição	Alimentação	Observação	Símbolo
Chave de nível alto LSH-101	—	NF (conduz na ausência de líquido)	
Chave de nível baixo LSL-101	—	NA (não conduz na ausência de líquido)	
Válvula Solenóide VS-101	220 Vca	NF (precisa ser energizada para abrir)	
Relé eletromecânico CR-101	24 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-102	24 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-103	24 Vcc	Possui um contato NA e um contato NF	

Após a análise do material disponível, o próximo passo consiste em estabelecer as condições em que o atuador deverá ser ativado e desativado. Como, neste caso, a monitoração da grandeza controlada é feita por *duas* chaves de nível, é possível resumir as possíveis combinações e o comportamento desejado para o atuador por meio de uma tabela-verdade de quatro linhas (Tabela 1.5). Note que é conveniente aproveitar a mesma tabela para registrar se, em cada situação, as chaves estão permitindo ou impedindo a passagem de corrente elétrica e se o atuador deverá ser energizado ou não. Ao se fazer isso, estamos incorporando na tabela informações que, de uma maneira mais amigável, indicam se tais elementos são do tipo NA ou NF, facilitando a elaboração do circuito lógico que pode ser observado na Figura 1.14.

Tabela 1.5: Resumo das combinações de estado das chaves de nível e ações a serem tomadas em cada situação.

Presença de líquido embaixo	Presença de líquido em cima	Ação
Sim (conduz corrente)	Sim (corta corrente)	Abrir (energizar)
Sim (conduz corrente)	Não (conduz corrente)	Manter último estado
Não (corta corrente)	Sim (corta corrente)	Considerar impossível
Não (corta corrente)	Não (conduz corrente)	Fechar (não energizar)

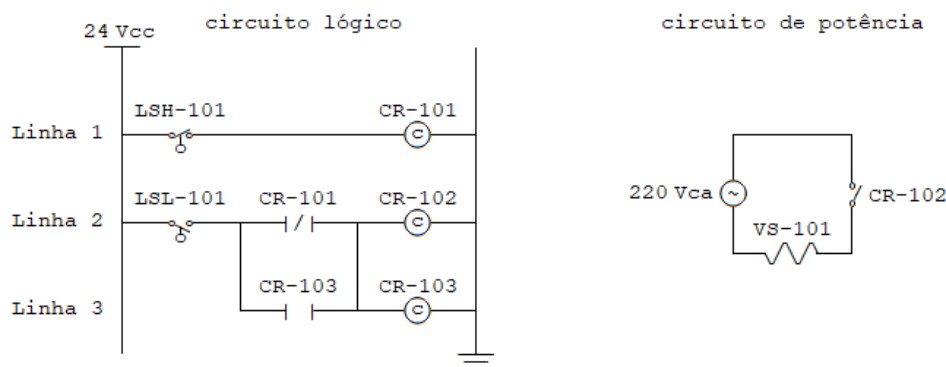


Figura 1.14: Sistema para controle *on/off* do nível de água em um tanque: circuito lógico (à esquerda) e circuito de potência (à direita).

Para entender como os circuitos que aparecem na Figura 1.14 foram elaborados, primeiramente deve-se perceber que a válvula *VS-101* deve ser acionada por meio de um circuito de potência que também incorpora uma fonte de 220 Vca e um contato interruptor para controlar a circulação da corrente. Para este papel, o contato NA do relé *CR-102* foi arbitrariamente escolhido. Assim, sempre que a bobina do respectivo relé for energizada, a válvula *VS-101* também será, causando sua abertura e a consequente diminuição do nível de água no tanque.

O próximo passo consiste em identificar quando a bobina do relé *CR-102* (e por consequência a válvula *VS-101*) deve ser energizada. Com ajuda da Tabela 1.5, percebe-se que isto deve ser feito na situação em que a chave *LSL-101* está conduzindo e a chave *LSH-101* está impedindo a passagem de corrente. No entanto, isso traz o seguinte problema: como a energização do atuador deve ser feita na situação em que tais condições acontecem simultaneamente, as duas chaves precisariam ser colocadas em série com a bobina do relé *CR-102*; logo, no momento em que uma delas está impedindo a passagem de corrente, a energização não aconteceria.

Justamente para resolver este problema, a Linha 1 foi introduzida no circuito lógico que aparece na Figura 1.14. Note que, colocando a chave *LSH-101* como única condição para alimentação da bobina do relé *CR-101*, a seguinte situação aparece: quando a chave está aberta, o contato NF do relé está fechado; e vice-versa. Sendo assim, na Linha 2 do circuito lógico, a chave *LSL-101* e o contato NF do relé *CR-101* foram colocados em série com a bobina do relé *CR-102*, cumprindo a primeira condição que aparece na Tabela 1.5, ou seja, quando *LSL-101* está conduzindo e *LSH-101* está impedindo a passagem de corrente, a bobina do relé *CR-102* é energizada.

Note que, tomando-se a Linha 2 do circuito lógico como referência, verifica-se que a quarta condição da Tabela 1.5 também foi atendida, uma vez que quando a chave *LSL-101* está impedindo a passagem de corrente, é impossível energizar o atuador. Como a terceira condição da tabela é considerada impossível, resta entender como a segunda condição foi atendida, sendo que “manter o último estado” significa que, se atuador está ativo no momento em que esta condição acontece, então ele deve permanecer a ativo; e, se está inativo, deve permanecer como tal.

A parte que do circuito lógico (Figura 1.14) que trata a questão descrita no parágrafo anterior é justamente a Linha 3 e, para perceber como isso acontece, imagine um cenário em que ela não existe: se o tanque estiver cheio, então, como visto, a válvula de escoamento irá abrir e o nível de água irá diminuir; quando a chave *LSH-101* perder o contato com a água, ela irá fechar, energizando a bobina do relé *CR-101* e fazendo com que seu contato NF abra, o que acarretaria em um desligamento precoce da bobina do relé *CR-102* e consequentemente do atuador. Dessa forma, a técnica conhecida como “selo” precisa ser usada para que, uma vez iniciado o fluxo de corrente para a bobina do relé *CR-102*, este seja mantido mesmo quando o contato NF de *CR-101* não estiver conduzindo. Isso poderia ser feito apenas colocando-se um contato NA do relé *CR-102* em paralelo com o contato NF do *CR-101*, o que simplificaria bastante a Linha 3 do circuito lógico.

No entanto, observando o circuito de potência, pode-se notar que o contato NA do relé *CR-102* está sendo usado para controlar o fluxo de corrente entre a fonte de alimentação e válvula. Além disso, voltando à Tabela 1.4, pode-se perceber que este contato NA é o único que o relé *CR-102* possui. Sendo assim, torna-se necessário utilizar o *CR-103* como relé auxiliar, bastando colocar sua bobina em paralelo com a do *CR-102* (as duas passam a ser ativadas e desativadas ao mesmo tempo) para, finalmente, usar o contato NA do *CR-103* como “selo” do contato NF do *CR-101*, como se vê na Linha 3 do circuito em questão.

CONSIDERAÇÕES FINAIS

Se o exemplo anterior foi seu primeiro contato com a elaboração de sistemas de automação baseados em relés, é provável que você tenha despendido mais tempo para compreendê-lo do que tinha planejado, terminando com o sentimento de que a tarefa não foi trivial.

Esse sentimento é perfeitamente normal porque, além de ter sido um primeiro contato, o que foi realizado não deixa de ser notável: um controlador capaz de manter o nível de água em um tanque dentro de uma faixa de trabalho, usando apenas três relés, sem nenhum elemento microprocessado. Lembrando também que, ao se substituir as chaves de nível por outros tipos de detectores, o mesmo raciocínio pode ser utilizado para a manipulação de atuadores que sejam capazes de controlar temperatura, pressão, fluxo e outras grandezas.

Vale destacar também que, até a década de 1970, não só o controle *on/off* de grandezas físicas, mas diversos outros tipos de automação industrial baseadas em eventos discretos eram realizadas por meio de sistemas com a mesma fundamentação do exemplo anterior, ou seja, circuitos lógicos formados por detectores de grandezas físicas, contatos e bobinas de relés e outros elementos acionando atuadores por meio de circuitos de potência. Essa estratégia era adotada para o intertravamento e o acionamento de motores, válvulas, resistores, cilindros pneumáticos e similares na grande maioria dos processos de extração, beneficiamento e manufatura do planeta, deste a mineração até a fabricação dos automóveis, por exemplo.

No entanto, como visto no início desse capítulo, os painéis de relés possuíam muitos problemas (que se tornavam mais significativos à medida em que os sistemas ficavam mais complexos), como: a dificuldade para localização de componentes danificados, a ocupação de grandes espaços físicos e a inflexibilidade que impedia mudanças rápidas na linha de produção. Vimos também que os CLPs vieram para substituir as lógicas baseadas em relés e permitir, justamente, que tais problemas fossem superados.

Por fim, vale lembrar: não termine essa leitura cometendo o erro de pensar que os CLPs foram projetados para eliminar todos os relés utilizados nos sistemas de automação. Eles foram projetados para eliminar apenas aqueles usados para a realização de lógicas automáticas. No caso do exemplo anterior, a substituição do circuito lógico baseado em relés por um CLP levaria ao cenário que pode ser observado na Figura 1.15. Note que a estratégia de separação entre a tomada de decisão e a energização dos atuadores permanece da mesma forma, ou seja, as saídas discretas dos CLPs são muitas vezes conectadas às bobinas dos relés, cujos contatos continuam permitindo ou impedindo o fluxo de corrente nos circuitos de potência.

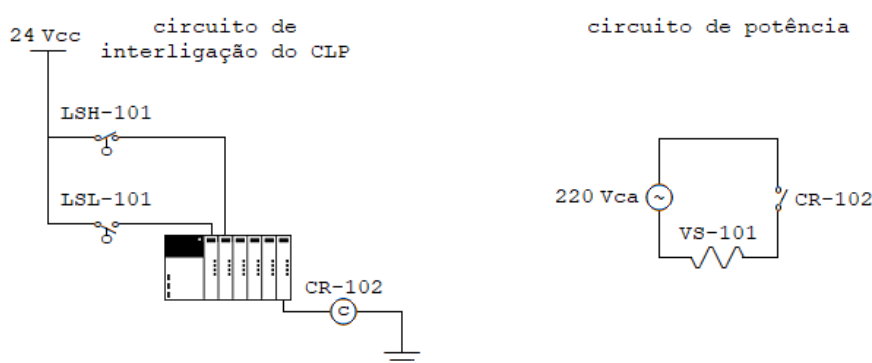


Figura 1.15: Modernização do sistema para controle *on/off* do nível de água no tanque do exemplo anterior: circuito de interligação do CLP aos elementos físicos da planta (à esquerda) e circuito de potência (à direita).

1.4 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Projetar e montar um circuito baseado em relés eletromecânicos para fazer o controle *on/off* do nível de água em um tanque.

MATERIAL NECESSÁRIO

- Módulo com relés eletromecânicos;
- Módulo para simulação de processo produtivo;
- Cabos elétricos apropriados (10 vermelhos e 4 pretos).



CONSIDERAÇÕES E TAREFAS PRELIMINARES

O diagrama esquemático simplificado do sistema de controle *on/off* a ser desenvolvido pode ser observado na Figura 1.16. Considere que, quando a bomba de água está ligada, a vazão de entrada é maior do que a vazão de saída que, por sua vez, possui um escoamento contínuo. Em seguida, faça a análise do material disponível para o projeto, prestando a devida atenção às características de cada item listado na Tabela 1.6.

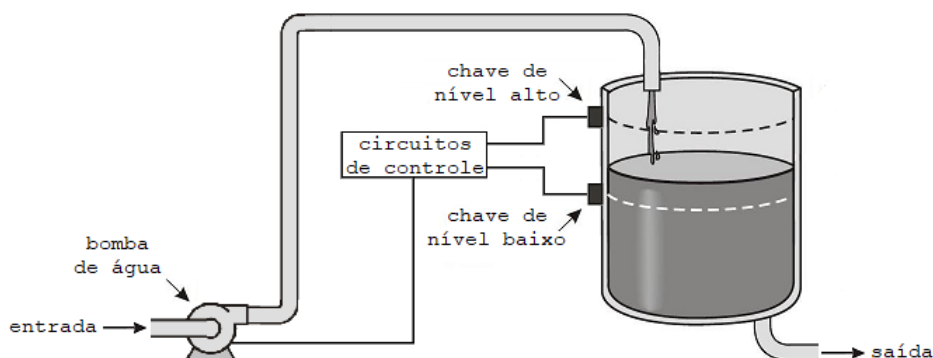








Figura 1.16: Tanque de água alimentado por uma bomba elétrica, com escoamento contínuo e chaves para detecção das condições de nível alto e nível baixo.

De posse dessas informações, utilize uma **folha de rascunho** para elaborar um sistema de controle capaz de manter o nível de água dentro da faixa desejada, cumprindo:

- Resuma as possíveis combinações das chaves de nível e o comportamento desejado para o atuador por meio de uma tabela-verdade; aproveite a mesma tabela para registrar se, em cada situação, as chaves estão permitindo ou impedindo a passagem de corrente;
- Apresente um *Diagrama Ladder* que represente o circuito lógico a ser utilizado no sistema em questão; apresente também o respectivo circuito de potência, que efetivamente acionará a bomba de água.

Tabela 1.6: Lista do material disponível para elaboração do sistema de controle *on/off*, incluindo as principais características de cada item.

Descrição	Alimentação	Observação	Símbolo
Chave de nível alto LSH-101	—	NF (conduz na ausência de líquido)	
Chave de nível baixo LSL-101	—	NA (não conduz na ausência de líquido)	
Bomba de Água BA-101	12 Vcc	—	
Relé eletromecânico CR-101	5 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-102	5 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-103	5 Vcc	Possui um contato NA e um contato NF	

EXPERIMENTO

Com o objetivo de verificar se a solução proposta para o problema está correta, a lógica projetada pelos estudantes será implementada de forma expositiva pelo professor. Para isso, utilizaremos uma planta didática, na qual as duas chaves de nível e a bomba de água estão instaladas; além de uma placa de controle, que contém os três relés previstos na lista de material.

1.5 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

Promover o primeiro contato dos estudantes com as ferramentas de *software* a serem utilizadas ao longo do curso e, similarmente, com o enunciado do trabalho prático a ser desenvolvido. Além da preparação de uma estrutura de diretórios para futura organização dos arquivos relacionados a este último.

MATERIAL NECESSÁRIO

- PC, com acesso à internet (para as atividades de pesquisa);
- PC, com as plataformas STEP 7 e WinCC instaladas (para as demais atividades).

Observação: os horários disponíveis para uso do laboratório e as instruções para fazer *login* nas estações de trabalho devem estar afixadas no local.

FERRAMENTAS SIEMENS

TAREFA 1 – O STEP 7 é a principal ferramenta fornecida pela Siemens para configuração e programação de seus CLPs e será vastamente utilizada ao longo das próximas semanas. Sendo assim, é importante que seus usuários saibam “onde estão pisando”. Para tal, faça uma breve pesquisa, respondendo às seguintes perguntas:

- a) Em qual ano ocorreu o lançamento da primeira versão? E, além disso, o fabricante possuía uma ferramenta predecessora (que foi substituída)? Se sim, qual era?
- b) Qual é a versão atual da ferramenta e, ao longo dos anos, quais foram as versões que representaram os maiores “saltos” em seu desenvolvimento? Produza um histórico resumido.
- c) Qual é a versão desta ferramenta que encontra-se instalada nas estações de trabalho do laboratório e, conseqüentemente, será utilizada ao longo do curso? Uma vez presente no laboratório, faça a verificação.

TAREFA 2 – Por sua vez, a Siemens oferece o software WinCC como principal ferramenta para o desenvolvimento de SSCs e esta será igualmente importante para os estudantes ao longo do curso. Desta forma, refaça, para esta ferramenta, a mesma pesquisa indicada na tarefa anterior.

TAREFA 3 – Atualmente, outra importante ferramenta para o desenvolvimento de sistemas de automação com produtos Siemens é o TIA Portal (Totally Integrated Automation Portal). Como você deve ter percebido durante a elaboração das respostas para as tarefas anteriores, ela também está instalada nas estações de trabalho do laboratório e, portanto, é importante que o seu propósito seja esclarecido. Pesquise e relate qual é a relação dessa ferramenta com as plataformas abordadas nas tarefas anteriores (e com outros produtos da Siemens).

PREPARAÇÃO PARA O TRABALHO PRÁTICO

TAREFA 1 – Grande parte das atividades previstas para as próximas semanas estão relacionadas a um trabalho prático e deverão ser cumpridas dentro de um laboratório preparado para isso, sendo que algumas delas devem ser desenvolvidas durante as aulas práticas do curso e outras em caráter extraclasse. Por exemplo, a aula prática da próxima semana começará com a configuração do *hardware* do CLP a ser utilizado, sendo que nesse momento é importante que você já esteja ciente dos objetivos e das principais características da planta a ser automatizada. Para tal, acesse o documento “Especificação Técnica do Sistema de Automação” por meio do *link* ao lado e em seguida:



- Faça a leitura da **Introdução** (Seção 1);
- Estude com atenção o **Diagrama de Tubulação e Instrumentação** (Seção 2);
- Passe para a próxima tarefa (as demais seções serão exploradas futuramente).

TAREFA 2 – O sistema de automação a ser desenvolvido durante o curso é composto por diversos aplicativos: um com as configurações e trechos de programação do CLP, um com as informações que compõem o SSC e alguns outros, relacionados à tecnologia OPC. Além disso, em diversos pontos do presente material, há indicações para a realização de *backups* relacionados ao projeto. Portanto, ao final do curso, a quantidade de arquivos gerados é considerável e, para que vários problemas decorrentes deste fato sejam evitados, faz-se necessário mantê-los bem organizados. Sendo assim, uma vez presente no laboratório, escolha a mesma estação de trabalho utilizada em sua primeira aula prática (para que não haja conflitos futuros) e prepare uma estrutura de diretórios similar à que pode ser observada na Figura 1.17, sabendo que:

- Provavelmente as estações de trabalho do laboratório já foram preparadas para que a estrutura de diretórios indicada na figura seja criada a partir de uma determinada pasta já existente. Caso esta informação não esteja afixada no local, informe-se com o técnico responsável antes de proceder com esta tarefa.
- O trabalho prático previsto neste curso foi concebido para que seja desenvolvido por uma dupla de estudantes, sendo que sua complexidade foi planejada para tal. Caso tenha a intenção de desenvolvê-lo individualmente ou em grupos maiores, informe-se com o docente responsável antes de proceder com esta tarefa.

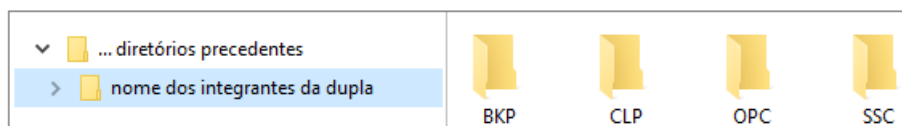


Figura 1.17: Estrutura de diretórios a ser criada em uma das estações de trabalho do laboratório, com o objetivo de armazenar os futuros arquivos do trabalho prático: pastas “BKP”, “CLP”, “OPC” e “SSC” dentro de uma pasta com o nome dos integrantes da dupla.

FINALIZANDO

TAREFA 1 – Quando encerrar suas atividades no laboratório, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Certifique-se de que todos os módulos didáticos da sua bancada de trabalho também estejam desligados e, ao sair, deixe o local limpo e organizado.

1.6 EXERCÍCIOS

QUESTÃO 1 – Cite três semelhanças e três diferenças entre os CLPs e os PCs convencionais.

QUESTÃO 2 – Iniciada por volta de 1760, a Primeira Revolução Industrial apresentou a mecanização dos processos produtivos como uma de suas principais características, substituindo operários que realizavam tarefas repetitivas por máquinas. Neste cenário, a principal força motriz das indústrias era o vapor, gerado em caldeiras que aqueciam a água por meio da queima de carvão ou madeira.

Já na Segunda Revolução Industrial, iniciada na década de 1870, a principal fonte de energia das máquinas foi a eletricidade. Sendo assim, pesquise: como ocorria a geração naquela época? Era por meio de usinas hidrelétricas, termoeletricas, ou outra tecnologia? A geração era centralizada em grandes usinas que atendiam vários processos produtivos por meio de um sistema de distribuição ou cada processo tinha sua própria fonte?

QUESTÃO 3 – Como se sabe, historicamente, uma das primeiras formas utilizadas para a transmissão de informações entre sensores, controladores e atuadores em uma planta industrial foi o padrão 3..15 psi (no qual a proporção da pressão do ar comprimido em uma linha pneumática pode indicar, por exemplo, a temperatura no interior de uma caldeira ou o percentual de abertura de uma válvula em determinado momento).

Atualmente, a transmissão de informações entre esses elementos é feita via redes de dados, ou seja, sensores, controladores e atuadores industriais microprocessados se comunicam por meio de tecnologias similares às utilizadas pelos PCs convencionais e dispositivos móveis que estão presentes em nosso cotidiano.

Portanto, pode-se dizer que o padrão 3..15 psi e a comunicação em rede são dois extremos da tecnologia de transmissão de dados em sistemas de automação industrial, pois o primeiro é muito antigo e a última representa a tecnologia atual. No entanto, entre esses extremos há uma tecnologia muito importante, pois representou uma evolução muito significativa em relação ao padrão 3..15 psi; tanto que ainda não deixou de existir, convivendo até hoje com as redes de dados em um grande número de sistemas de automação em todo planeta. Sendo assim, responda:

- a) Qual é essa tecnologia intermediária?
- b) Quais são os principais fatores que a mantém viva até hoje?

QUESTÃO 4 – Como se sabe, após o lançamento do modelo 084 (considerado por muitas pessoas como o primeiro CLP da história), o sucesso foi tão grande que a Modicon (empresa fabricante) não demorou a fazer aprimoramentos. Em 1975, já haviam sido lançados os modelos 184, 284 e 384. No entanto, este último modelo (o Modicon 384) trouxe uma melhoria que merece destaque. Faça um pequeno texto indicando qual foi essa melhoria e explicando seus benefícios.

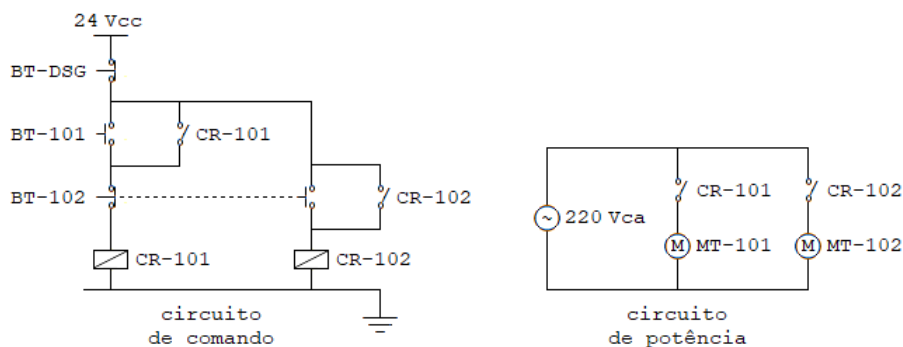
QUESTÃO 5 – A Revolução Digital (ou a Terceira Revolução Industrial) não possui períodos de início e fim tão bem estabelecidos na literatura quanto as duas revoluções anteriores. Isso porque, antes que os elementos microprocessados se tornassem os principais componentes dos sistemas de automação atuais, vários avanços preliminares foram necessários no campo da eletrônica industrial, o que levou vários anos para ocorrer. Porém, muitos autores consideram que a consolidação de uma determinada tecnologia foi o que realmente desencadeou a revolução em questão. Sendo assim, responda:

- Qual é essa tecnologia e em qual época foi consolidada nas indústrias?
- Quais foram seus principais benefícios na época? Explique pelo menos três.

QUESTÃO 6 – No quadro a seguir pode-se observar uma analogia entre períodos históricos relacionados às revoluções industriais e as formas de se registrar imagens de pessoas, lugares ou objetos. É interessante notar que cada avanço na área do registro de imagens foi realmente uma revolução, mas que a transição entre a penúltima e última coluna ocorreu de forma relativamente muito rápida, ou seja, a vida das câmeras digitais foi muito curta. No entanto, ao se pensar em termos de componentes eletrônicos, percebe-se que a tecnologia usada nas câmeras digitais foi apenas transferida para os *smartphones* (com alguns aprimoramentos, claro) e que esta última revolução veio realmente com a possibilidade de se armazenar as fotos na nuvem e também compartilhá-las com quem se deseja. Sendo assim, fazendo a analogia reversa, elabore um pequeno texto explicando qual é a relação entre o cenário anterior e a Quarta Revolução Industrial.






Manufatura manual	Primeira Revolução Industrial	Segunda Revolução Industrial	Revolução Digital	Indústria 4.0
				

QUESTÃO 7 – De acordo com os circuitos que podem ser observados a seguir e sabendo que as três botoeiras (*BT-101*, *BT-102* e *BT-DSG*) são não-retentivas (a primeira é do tipo NA, a segunda possui um contato NF e um NA e a última é do tipo NF), responda:



- Se os motores *MT-101* e *MT-102* estiverem desligados, o que acontecerá se a botoeira *BT-101* for pressionada e liberada em seguida? Justifique.
- Se o motor *MT-101* estiver ligado, o que acontecerá se a botoeira *BT-102* for pressionada e liberada em seguida? Justifique.
- Se algum dos motores estiver ligado, o que acontecerá se as três botoeiras forem pressionadas ao mesmo tempo? Justifique.
- Considerando que os circuitos não podem ser alterados, é possível que os motores *MT-101* e *MT-102* funcionem ao mesmo tempo? Justifique.








QUESTÃO 8 – Considerando a lista de material a seguir e sabendo que, uma vez pressionada a botoeira *BT-LIG*, a lâmpada *LP-HLP* deverá acender e permanecer acesa até que a botoeira *BT-DSG* seja pressionada, apresente um *Diagrama Ladder* que represente o circuito lógico a ser utilizado no sistema em questão. Dica: antes de começar, é importante que você perceba qual é o motivo que torna dispensável o uso de um circuito de potência.

Descrição	Alimentação	Observação	Símbolo
Botoeira BT-DSG	—	NA (conduz quando pressionada)	
Botoeira BT-LIG	—	NA (conduz quando pressionada)	
Lâmpada LP-HLP	24 Vcc	—	
Relé eletromecânico CR-101	24 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-102	24 Vcc	Possui um contato NA e um contato NF	

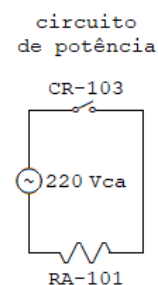
QUESTÃO 9 – Suponha que uma chocadeira para ovos de avestruz deverá ser automatizada. Por meio de um resistor de aquecimento, o sistema de controle *on/off* a ser desenvolvido deverá manter a temperatura interna dentro da faixa determinada por dois termostatos (um determinada o limite máximo e o outro o limite mínimo). Considere que, quando ligado, o resistor é capaz de elevar a temperatura para além do limite alto; e que, quando desligado, as perdas para o meio ambiente são capazes de reduzi-la para além do limite baixo. Sendo assim, após uma análise cuidadosa da lista de material disponível para o projeto em questão, cumpra as seguintes solicitações:

- Observe as possíveis combinações das situações dependentes da temperatura e o comportamento desejado para o atuador na tabela-verdade a seguir. Reproduza a tabela, aproveitando para registrar se, em cada momento, os termostatos estão permitindo ou impedindo a passagem de corrente;

- b) Sabendo que o circuito de potência a ser utilizado no sistema em questão pode ser observado na figura a seguir, apresente um *Diagrama Ladder* correspondente ao circuito lógico, sendo que você não é obrigado(a) a usar todos os componentes que aparecem na lista de material.

Descrição	Alimentação	Observação	Símbolo
Termostato do limite alto TSH-101	—	Possui um contato NA	
Termostato do limite baixo TSL-101	—	Possui um contato NA	
Resistor de Aquecimento RA-101	220 Vca	—	
Relé eletromecânico CR-101	24 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-102	24 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-103	24 Vcc	Possui um contato NA e um contato NF	
Relé eletromecânico CR-104	24 Vcc	Possui um contato NA e um contato NF	

A temperatura está acima do limite baixo?	A temperatura está acima do limite alto?	Ação
Sim	Sim	Desligar aquecimento
Sim	Não	Manter último estado
Não	Sim	Considerar impossível
Não	Não	Ligar aquecimento



QUESTÃO 10 – Com o objetivo de modernizar a chocadeira cujo sistema de automação foi projetado na Questão 9, vamos substituir o circuito lógico baseado em relés eletromecânicos por um CLP com entradas e saídas digitais capazes de receber e produzir sinais de 24 Vcc. Para isso, considerando que o circuito de potência não deverá ser alterado, apresente o diagrama de interligação entre este dispositivo os elementos da lista de material que você julgar necessários (consulte o quadro anterior).

QUESTÃO 11 – Na Questão 10, um CLP foi utilizado para eliminar a maioria dos relés eletromecânicos utilizados na Questão 9. Faça um pequeno texto explicando claramente o que impede a eliminação de todos. Relacione este motivo com o fato de que, mesmo tendo surgido no final do Século XIX, os relés continuam presentes nos sistemas de automação concebidos após a Quarta Revolução Industrial.

QUESTÃO 12 – **ENADE 2014**, QUESTÃO 30 – Um engenheiro de controle e automação necessita especificar um transdutor de temperatura para um projeto de controle automático de um forno. É requisito de projeto que o controlador lógico programável (CLP) esteja localizado na sala de comando, a qual está a uma distância de 300 metros do forno.

Nesse contexto, considerando-se que o engenheiro escolheu um transdutor com transmissão no padrão 4 mA a 20 mA, avalie as seguintes asserções e a relação proposta entre elas.

- I. A transmissão de sinal de transdutores distantes do CLP é mais segura no padrão 4 mA a 20 mA, que nos padrões 0 V a 5 V ou 0 mA a 20 mA.

PORQUE

- II. No padrão 4 mA a 20 mA é possível identificar perda de sinal nos cabos de transmissão, como, por exemplo, em problemas de rompimento de cabo ou desconexão com o CLP.

A respeito dessas asserções, assinale a opção correta.

- (A) As asserções I e II são verdadeiras, e a II é uma justificativa correta da I.
- (B) As asserções I e II são verdadeiras, mas a II não é uma justificativa correta da I.
- (C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- (D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- (E) As asserções I e II são proposições falsas.

1.7 REFERÊNCIAS

- BENNETT, S. A brief history of automatic control. **IEEE Control Systems Magazine**, v. 16, n. 3, p. 17–25, 1996.
- BRASIL, Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira. **ENADE 2014**. Engenharia de Controle e Automação. 2014. https://download.inep.gov.br/educacao_superior/enade/provas/2014/15_engenharia_controle_automacao.pdf. [Online; accessed 17-agosto-2025].
- CHIRNSIDE, R.C. Sir Joseph Swan and the invention of the electric lamp. **Electronics and Power**, v. 25, n. 2, p. 96, 1979.
- ENGELMAN, R. **The Second Industrial Revolution, 1870–1914**. 2015. <http://ushistoryscene.com/article/second-industrial-revolution/>. [Online; accessed 25-setembro-2018].
- FRANCHI, C.M.; CAMARGO, V.L.A. de. **Controladores Lógicos Programáveis: Sistemas Discretos**. Érica, 2008. 352 p. ISBN 9788536501994.
- GICSP, E. H.; ASSANTE, M.; CONWAY, T. **An Abbreviated History of Automation & Industrial Controls Systems and Cybersecurity**. 2014.
- GOEKING, W. **Da máquina a vapor aos *softwares* de automação**. 2010. <https://www.osetoelettrico.com.br/da-maquina-a-vapor-aos-softwares-de-automacao/>. [Online; accessed 16-outubro-2018].
- GREENBERG, B.; WATTS, L.S.; BUCKI, C. **Social History of the United States**. ABC-CLIO, 2008. v. 1930. ISBN 9781598841282.
- GUARNIERI, M. Revolving and Evolving – Early DC Machines. **IEEE Industrial Electronics Magazine**, v. 12, n. 3, p. 38–43, 2018.
- HAYDEN, E.; ASSANTE, M.; CONWAY, T. An abbreviated history of automation & industrial controls systems and cybersecurity. **SANS analyst white papers**, 2014.
- JEFIMENKO, O. **Electrostatic Motors: Their History, Types, and Principles of Operation**. Integrity Research Institute, 2011. 159 p. ISBN 9781935023470.
- LEITE, M. **Técnicas de Programação: uma abordagem moderna**. Brasport, 2006. 436 p. ISBN 9788574522295.
- MECHANICAL ENGINEERS, American Society of. **ASME Technical Papers**. ASME, 1995.
- REHG, J.A.; SARTORI, G.J. **Programmable Logic Controllers**. 2. ed.: Pearson Prentice Hall, 2009. 600 p. ISBN 9780135048818.

- RÜßMANN, M. *et al.* **Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries**. Edição: Boston Consulting Group. 2015.
- SHANNON, C.E. A symbolic analysis of relay and switching circuits. **Electrical Engineering**, v. 57, n. 12, p. 713–723, 1938.
- SILVEIRA, P.R. da; SANTOS, W.E. dos. **Automação e Controle Distreto**. Érica, 1998. 229 p. ISBN 9788571945913.
- WILLEMS, J.C.; POLDERMAN, J. **Introduction to Mathematical Systems Theory: A Behavioral Approach**. Springer, 2013. 424 p. ISBN 9781475729535.

Capítulo 2

CONCEITOS FUNDAMENTAIS SOBRE CLPs

2.1 INTRODUÇÃO

Para se fazer um bom uso de um CLP, antes de programá-lo, é necessária uma boa compreensão de suas características, tanto as específicas (quantidade de memória disponível, capacidade de processamento, limites para a expansão de entradas e saídas etc) como as gerais (modularidade de *hardware*, segmentação da memória, ciclo de funcionamento etc). Enquanto as primeiras precisam ser verificadas nos manuais de cada equipamento, as últimas devem ser objeto de estudo dos profissionais envolvidos com essa tecnologia e serão abordadas ao longo deste capítulo.

Conforme mencionado no capítulo anterior, se alguém lhe perguntar o que é um CLP, você pode começar a explicação por meio de uma comparação entre esses dispositivos e os PCs convencionais, dizendo que os membros dos dois grupos são microprocessados, possuem memórias retentivas e voláteis e, certamente, capacidade de trocar dados via rede. Em contraponto, as diferenças começam na robustez do *hardware*: os CLPs são fabricados para operar continuamente, durante anos, em ambientes não climatizados e na presença de outras adversidades, como partículas em suspensão, vibrações moderadas e ruídos eletromagnéticos; já os PCs, para que possam ser utilizados em um sistema de automação industrial, precisam ser instalados em ambientes nos quais esses problemas sejam controlados.

Outra diferença fundamental entre os dois tipos de equipamento está relacionada às formas com as quais eles interagem com outros elementos do mundo real: enquanto os PCs possuem periféricos como teclado, *mouse* e monitor, para interface com os seres humanos; os CLPs possuem terminais elétricos de entrada e de saída para, entre outras coisas, interligação de instrumentos de medição e atuadores, sendo que seus respectivos sinais passam a ser monitorados e controlados por meio de valores em formato digital, armazenados em determinadas posições de memória do CLP.

Por exemplo, no sistema de controle *on/off* apresentado no capítulo anterior, o circuito lógico formado por relés eletromecânicos (Figura 2.1A) pode ser substituído por um CLP devidamente programado (Figura 2.1B), sem que haja necessidade de se fazer alterações no circuito de potência (Figura 2.1C). Neste caso, é interessante notar que as chaves de nível e o relé utilizado para acionar este último circuito precisam ser mantidos no sistema, pois eles são os meios pelos quais o CLP recebe informações e interfere no mundo real. No entanto, os dois outros relés usados no primeiro circuito foram suprimidos no sistema com CLP, uma vez que os papéis desempenhados por eles podem ser substituídos por recursos de programação.

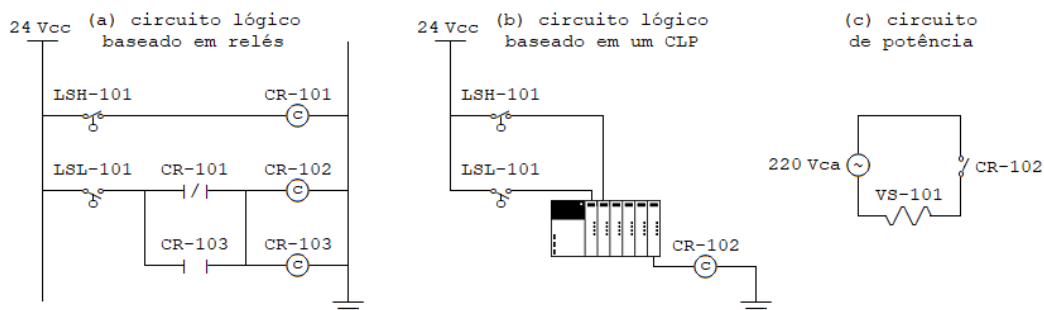


Figura 2.1: Diagramas elétricos para o sistema de controle *on/off* apresentado como exemplo prático no capítulo anterior: circuito lógico baseado em relés (a); circuito lógico baseado em um CLP (b); circuito de potência, que pode atender a qualquer um dos anteriores (c).

Dessa forma, ao se transpor e extrapolar as vantagens decorrentes do uso do CLP neste exemplo para os complexos sistemas de automação que controlam processos industriais de grande porte, onde um controlador do tamanho de uma caixa de sapatos pode conter um programa que substitui centenas ou mesmo milhares de relés eletromecânicos e um enorme volume de cabos elétricos, dois benefícios são imediatamente percebidos: **significativa economia de material elétrico e redução do espaço físico ocupado pelo sistema de automação.**

Para os sistemas de automação que se baseiam em CLPs, as tarefas de elaborar e manter atualizados os projetos elétricos que especificam, entre outras coisas, as interligações entre elementos periféricos e suas entradas e saídas (como na Figura 2.1B) são uma necessidade. No entanto, toda a parte lógica do sistema é implementada por meio de um ou mais programas, que podem ser consultados ou mesmo impressos, sempre que necessário. Além disso, as diversas instruções que compõem esses programas podem ser incluídas, alteradas e excluídas de forma muito mais simples do que as correspondentes tarefas em um painel composto por elementos elétricos. Inclusive, em um ambiente de programação, os códigos podem ser desenvolvidos e testados antes de serem descarregados no CLP. Tais fatos agregam pelo menos mais duas vantagens a esse tipo de sistema: **documentação simplificada e redução do tempo despendido em desenvolvimentos e alterações.**

Existem ainda várias outras vantagens decorrentes da substituição de circuitos lógicos baseados em relés por CLPs. Entre elas, aquelas proporcionadas por algumas linguagens de programação, que por esse motivo, serão discutidas no próximo capítulo. No entanto, neste momento, ainda há de se destacar uma qualidade muito importante desses dispositivos: **a versatilidade**, na medida em que um mesmo modelo de CLP pode ser utilizado em pontos distintos de um processo produtivo ou mesmo em diferentes segmentos industriais, como extrativista, siderurgico, petroquímico, alimentício, em estações de tratamento de água e esgoto ou na geração, transmissão e distribuição de energia.

2.2 PRINCIPAIS CARACTERÍSTICAS DE HARDWARE

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Controladores Lógicos Programáveis	1	1.8 à 1.11

2.3 ACESSO À MEMÓRIA

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Controladores Lógicos Programáveis	5	5.3.5 à 5.3.8
	4	4.2.1 à 4.2.6

2.4 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Praticar tarefas fundamentais disponíveis em uma plataforma de programação de CLPs, como: criação de projeto, configuração de *hardware*, alteração do estado da CPU, realização de *download*, declaração de variáveis, operações de leitura e escrita na memória, verificação do tempo de ciclo e *backup* de projeto.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (STEP 7);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.



EXPERIMENTO 1

Como participante deste curso, este será seu primeiro contato com o CLP a ser utilizado ao longo das próximas semanas. As tarefas solicitadas a seguir são fundamentais e serão necessárias em diversas outras oportunidades. Por isso, realize-as com bastante atenção e lembre-se de retornar a este ponto em caso de dúvidas futuras.

TAREFA 1 – Abra a plataforma de programação de CLPs adotada neste curso. Crie um projeto com o nome “III15S” e salve-o na pasta “CLP”, que deve estar dentro da estrutura de diretórios criada durante as tarefas extra-classe do capítulo anterior.



TAREFA 2 – Adicione a CPU do CLP que você usará ao longo do semestre ao projeto que você acabou de criar, especificando o nome do dispositivo, a família, o modelo e a versão do *hardware*, de acordo com as informações disponíveis em sua bancada.



TAREFA 3 – Configure o *hardware* do seu CLP, definindo o endereço IP da porta de comunicação e realizando ações adicionais, conforme indicado no procedimento que pode ser acessado por meio o *link* ao lado.



TAREFA 4 – Faça o *download* das configurações de *hardware* realizadas na tarefa anterior.



TAREFA 5 – Faça alterações no estado da CPU (transições $STOP \rightleftharpoons RUN$), observando as respectivas sinalizações físicas (troca de cor de um dos *LEDs*). Ao final, deixe a CPU ativada (modo *RUN*) e verifique seu tempo de ciclo.



EXPERIMENTO 2

Outra tarefa fundamental relacionada à programação de CLPs é a declaração de variáveis (ou a criação de *tags*, se preferir). Em seu programa, você até pode usar endereços de memória de forma absoluta (por exemplo: *I0.0*, *IW128*, *Q5.4*, *MD500*), mas isso não é uma boa prática pois, a medida em que seu projeto vai se tornando mais elaborado, a falta de nomes adequados e comentários vai lhe deixando mais perdido. Dessa forma, antes de usar qualquer endereço de memória, o ideal é fazer o cadastro de uma variável, atribuindo a este endereço, pelo menos: um nome único, um comentário e a definição do tipo de dado que ele armazenará. As tarefas a seguir tratam desse assunto, ou seja, mostram como se faz o cadastro de variáveis utilizando a plataforma de programação adotada neste curso.

TAREFA 1 – Encontre, dentro da sua plataforma de programação, o módulo com o qual se faz o cadastro de variáveis. Neste primeiro momento, com o objetivo de organizar seu projeto, crie todos os *grupos de variáveis* indicados no quadro a seguir (os números fazem parte dos nomes).

Como?



Nome do Grupo de variáveis

- 01 – Entradas Digitais
- 02 – Saídas Digitais
- 03 – Entradas Analógicas
- 04 – Saídas Analógicas
- 05 – Comandos
- 06 – Estados
- 07 – Alarmes
- 08 – Auxiliares Digitais
- 09 – Auxiliares Analógicos
- 10 – Grandezas Analógicas
- 11 – Parâmetros

TAREFA 2 – Faça o cadastro de todos os *tags* relacionados às conexões elétricas do CLP, ou seja, preencha os quatro primeiros *grupos de variáveis* (entradas digitais, saídas digitais, entradas analógicas e saídas analógicas) de acordo com as informações contidas no **Projeto Elétrico** do sistema (veja a Seção 3 do documento “Especificação Técnica do Sistema de Automação”, que pode ser acessado por meio do *link* ao lado).

Como?



Link



TAREFA 3 – Faça um novo *download* do projeto, que agora contém as variáveis cadastradas na tarefa anterior.

Como?



EXPERIMENTO 3

Por fim, mas não menos importante do que os primeiros experimentos, vamos explorar a funcionalidade que nos permite monitorar valores na memória do CLP (por meio de operações de leitura) e também alterá-los (por meio de operações de escrita). Para tal, proceda conforme solicitado nas tarefas a seguir.

TAREFA 1 – Encontre, dentro da sua plataforma de programação, o módulo que torna possível **a monitoração e a alteração de valores na memória do CLP**. Em seguida, crie uma tabela para acessar todos os *tags* relacionados às conexões elétricas do CLP, identificando-a com o nome “**Teste de I/O**”.



TAREFA 2 – Preencha a tabela criada na última tarefa com todos os *tags* relacionados às conexões elétricas do CLP. **Dica:** você pode voltar aos *grupos de variáveis* construídos no experimento anterior, copiar as colunas que contêm os nomes dos *tags* e colar tais informações na tabela em questão.



TAREFA 3 – Ative o recurso de monitoração cíclica das posições de memória cadastradas na tabela e, por meio das botoeiras presentes no módulo de interfaces elétricas conectado ao seu CLP, altere o nível lógico das 9 primeiras entradas digitais.



TAREFA 4 – Altere (mais de uma vez, se quiser) o nível lógico das saídas digitais associadas ao **Mixer** e observe o efeito desta ação em sua planta didática (termine este procedimento deixando o atuador desligado).



TAREFA 5 – Ainda com o recurso de monitoração cíclica das posições de memória ativado, escreva o valor 15000 no endereço associado à saída analógica que envia a referência de velocidade para a **Bomba de água**. Após o acionamento do atuador, você deve:



- observar as alterações do valor decimal no endereço associado à entrada analógica que recebe o sinal do medidor de nível (*LT-101*), correspondendo ao aumento da quantidade de água no tanque;
- observar as alterações dos valores binários nos endereços associados às entradas digitais que recebem os sinais da chave de nível baixo (*LSL-101*) e da chave de nível de alto (*LSH-101*), nos respectivos momentos em que a água atingir tais detectores;
- quando o tanque estiver cheio (com água acima da chave de nível alto), desligue a **Bomba de água**, escrevendo o valor 0 na posição de memória associada à sua referência de velocidade.

TAREFA 6 – Ligue e desligue o **Aquecedor** alterando o nível lógico da respectiva saída digital e observe: a entrada digital associada ao retroaviso que indica se ele está ligado ou não; a entrada analógica que indica a temperatura da água do tanque. Termine este procedimento deixando o atuador desligado.



TAREFA 7 – Ative a saída digital apropriada para abrir a **Válvula de dreno direto** (VD101), esgotando a água do tanque de sua planta didática (ao final, feche o atuador).



FINALIZANDO

Nos experimentos anteriores você deve ter criado um projeto em sua plataforma de desenvolvimento, configurado o *hardware* do CLP de sua bancada, declarado variáveis e manipulado seus valores em tempo real, além de ter feito dois ou mais procedimentos de *download* e outras atividades. Se os conceitos envolvidos foram bem assimilados e não houver dúvidas em relação ao que foi desenvolvido, realize as tarefas de encerramento a seguir (caso contrário, antes de terminar, faça os esclarecimentos necessários com o docente).

TAREFA 1 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.

Como?



TAREFA 2 – Feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

2.5 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

Cadastrar as variáveis pertencentes aos grupos de sinais utilizados para que o CLP possa, entre outras coisas, receber e fornecer dados ao SSC. Além disso, fazer a importação das variáveis auxiliares relacionadas ao primeiro, completando a estrutura de dados a ser utilizada durante sua programação.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (STEP 7);

RETOMANDO

Durante a última aula prática, entre outras coisas, você deve ter declarado em seu projeto as variáveis relacionadas às entradas e saídas elétricas do CLP. No entanto, estas representam apenas uma parte da estrutura de dados do sistema a ser desenvolvido, que permitirá ao CLP receber e fornecer dados ao SSC por meio de variáveis distribuídas em cinco grupos: Comandos, Estados, Alarmes, Grandezas analógicas e Parâmetros. Além do cadastro dessas variáveis, as tarefas a seguir também preveem a importação de outras, pertencentes aos grupos Auxiliares Digitais e Auxiliares Analógicos.

TAREFA 1 – Abra a plataforma de programação de CLPs adotada neste curso e, em seguida, localize e abra seu projeto (espera-se que nele estejam incorporadas todas as configurações realizadas na última aula prática).





TAREFA 2 – Faça o cadastro de todos os *tags* pertencentes aos cinco *grupos de variáveis* projetados para que, entre outras coisas, o CLP possa receber e fornecer dados ao SSC, sendo eles: Comandos, Estados, Alarmes, Grandezas analógicas e Parâmetros. Para isso, consulte o **Mapa de Memória do CLP** (veja a Seção 4 do documento “Especificação Técnica do Sistema de Automação”, que pode ser acessado por meio do *link* ao lado).



TAREFA 3 – De acordo com o que está definido ao longo do **Descritivo Funcional** (veja a Seção 5 da “Especificação Técnica do Sistema de Automação”, cujo *link* está na tarefa anterior), faça a importação dos *tags* pertencentes aos dois *grupos de variáveis* indicados na Tabela 2.1, sendo eles: Auxiliares Digitais e Auxiliares Analógicos.



Tabela 2.1: Grupos auxiliares cujos *tags* estão previstos no **Descritivo Funcional** e, portanto, também precisam ser incluídos no sistema.

Grupo	Quantidade de <i>tags</i>	Link
Auxiliares Digitais	40	
Auxiliares Analógicos	13	

FINALIZANDO

Ao longo das últimas tarefas você deve ter cadastrado e importado as variáveis que completam a estrutura de dados do projeto a ser desenvolvido durante o curso. Se não houver dúvidas em relação ao que foi realizado, cumpra as tarefas a seguir (caso contrário, antes de terminar, anote suas dúvidas e procure esclarecê-las com o docente assim que possível).

TAREFA 1 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.

Como?



TAREFA 2 – Quando encerrar suas atividades, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

2.6 EXERCÍCIOS

QUESTÃO 1 – Sabendo que a grande maioria dos CLPs se baseia no conceito de *hardware* modular, cite e explique a finalidade de cinco tipos de módulo.

QUESTÃO 2 – Explique sucintamente cada etapa do ciclo de funcionamento de um CLP.

QUESTÃO 3 – As características de *hardware* dos CLPs estão bastante relacionadas aos seus fabricantes. Ou seja, cada fabricante fornece um ou mais modelos de CLPs com características de *hardware* particulares. Sendo assim, é normal que para cada “família” de CLPs exista uma ferramenta de programação. Porém, existem pontos em comum. Por exemplo, o sistema operacional Windows é utilizado como plataforma para a maioria dessas ferramentas de programação. Dessa forma, o desenvolvimento de programas para CLPs pode ser realizado por meio dos recursos intuitivos desse sistema operacional. Além de facilitar essa tarefa, as diversas ferramentas de programação disponíveis no mercado possuem outras *funcionalidades* em comum, cite 3 delas.

QUESTÃO 4 – Na maioria dos casos, os componentes de *hardware* de um CLP precisam ser declarados de forma apropriada para que a CPU e o programador possam fazer uso dos mesmos. Por exemplo, os cartões de entradas digitais precisam ser declarados e associados a posições específicas da memória dos CLPs, estabelecendo relações do tipo: a primeira entrada do cartão 1 corresponde a posição de memória I0.0, a oitava entrada do cartão 5 corresponde a posição de memória I4.7 e assim por diante. Sendo assim, para alguns modelos de CLP, a inclusão de novos componentes de *hardware* pode causar alguns transtornos caso o CLP esteja em operação (controlando um processo). Explique por que isso pode ocorrer.

QUESTÃO 5 – Com relação ao CLP S7-1200, complete a Tabela de endereçamento 2.2.

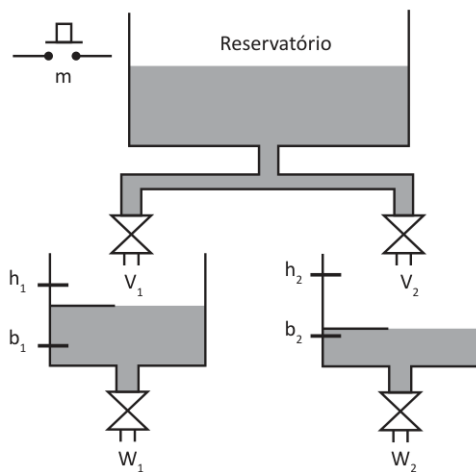
Tabela 2.2: Exemplo de quadro para preenchimento dos endereços solicitados no problema em questão.

Endereço	Comentário
	Décima quarta entrada digital (supondo que a primeira é I0.0)
	Vigésima nona saída digital (supondo que a primeira é Q0.0)
	Qualquer entrada analógica
	Próxima entrada analógica após a do item anterior
	Qualquer variável do tipo real na área de memória do usuário
	Próxima variável do tipo real após a do item anterior

QUESTÃO 6 – Considere um processo de produção automatizado com dois Controladores Lógico-Programáveis. O CLP01 controla os equipamentos EQP01 à EQP04 e o CLP02 controla os equipamentos EQP05 à EQP08. Suponha que, inicialmente, estes CLPs eram independentes um do outro, pois todos os equipamentos do processo funcionavam separadamente. Posteriormente, descobriu-se que o processo teria um rendimento maior caso o EQP04 funcionasse apenas quando o EQP05 estivesse ligado. Ou seja, descobriu-se que o sinal “EQP05 ligado” deveria ser enviado do CLP02 para o CLP01. Descreva em um parágrafo uma possível solução para tal situação, considerando as seguintes restrições:

- Os CLPs são de fabricantes diferentes, assim não podem se comunicar por meio de rede. Além disso, o envio deste único sinal *não* justifica financeiramente a instalação de módulos de comunicação ou *gateways* que viabilizem a troca de dados via rede.
- O processo em questão *não* dispõe de um Sistema de Supervisão e Controle, apenas pequenos painéis de operação, independentes para cada CLP;
- *Não* é viável redistribuir os equipamentos entre os CLPs. Ou seja, o CLP01 deve continuar controlando os equipamentos EQP01 à EQP04 e o CLP02 deve continuar controlando os equipamentos EQP05 à EQP08;
- Existem entradas e saídas digitais disponíveis em ambos os CLPs. No entanto, a distância entre o CLP01 e o EQP05 torna a ligação elétrica entre os mesmos inviável.

QUESTÃO 7 – **ENADE 2019**, QUESTÃO 20 – O gestor de uma pequena indústria alimentícia pretende automatizar o sistema de distribuição de água, apresentado na figura, para aumentar a margem de lucro estimado em R\$ 2.000,00 por mês.



Legenda:

- m: botão de comando;
- h_1 , b_1 e h_2 , b_2 : sensores de nível;
- V_1 e V_2 : válvulas do tipo abre/fecha;
- W_1 e W_2 : válvulas com controle proporcional.

A tabela a seguir apresenta quatro orçamentos para implantação do sistema.

		Proposta da empresa A	Proposta da empresa B	Proposta da empresa C	Proposta da empresa D
Características do CLP	I/O Digitais	8 Entradas/ 4 Saídas	8 Entradas/ 4 Saídas	8 Entradas/ 8 Saídas	4 Entradas/ 4 Saídas
	I/O Analógicas	2 Entradas/ 1 Saídas	4 Entradas/ 2 Saídas	2 Entradas/ 2 Saídas	2 Entradas/ 2 Saídas
Custo do CLP		R\$ 4.000,00	R\$ 4.000,00	R\$ 4.500,00	R\$ 2.000,00
Custo dos sensores de nível		R\$ 1.000,00	R\$ 2.000,00	R\$ 1.000,00	R\$ 1.000,00
Custo de V1 e V2		R\$ 2.000,00	R\$ 4.000,00	R\$ 1.500,00	R\$ 2.000,00
Custo de W1 e W2		R\$ 3.000,00	R\$ 5.000,00	R\$ 3.000,00	R\$ 3.000,00
Custo da instalação e mão de obra		R\$ 8.000,00	R\$ 15.000,00	R\$ 12.000,00	R\$ 8.000,00
Custo total		R\$ 18.000,00	R\$ 30.000,00	R\$ 22.000,00	R\$ 16.000,00

Com base nas informações técnicas e financeiras apresentadas na tabela e considerando que todo o lucro obtido mensalmente devido à implantação do sistema automatizado será utilizado para pagá-lo em um prazo de 12 meses, avalie as afirmações a seguir.

- I. A proposta da Empresa A atende aos critérios técnicos e financeiros para implantação do sistema.
- II. A proposta da Empresa B atende aos critérios técnicos, mas não aos financeiros para implantação do sistema.
- III. A proposta da Empresa C atende aos critérios técnicos e financeiros para implantação do sistema.
- IV. A proposta da Empresa D não atende aos critérios técnicos e financeiros para implantação do sistema.

É correto o que se afirma em:

- (A) I, apenas.
- (B) IV, apenas.
- (C) I e II, apenas.
- (D) II e III, apenas.
- (E) III e IV, apenas.

2.7 REFERÊNCIAS

BRASIL, Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira. **ENADE 2019**. Engenharia de Controle e Automação. 2019. https://download.inep.gov.br/educacao_superior/enade/provas/2019/ENGENHARIA_CONTROLE_AUTOMACAO.pdf. [Online; accessed 17-agosto-2025].

FRANCHI, C.M.; CAMARGO, V.L.A. de. **Controladores Lógicos Programáveis: Sistemas Discretos**. Érica, 2008. 352 p. ISBN 9788536501994.

Capítulo 3

PROGRAMAÇÃO EM DIAGRAMA LADDER: INSTRUÇÕES BINÁRIAS, DECIMAIS E MISTAS

3.1 INTRODUÇÃO

Conforme apresentado no Tópico 1, os Controladores Lógico-Programáveis (CLPs) surgiram no final da década de 1960, projetados para substituir, com vantagens, os sistemas de automação baseados em relés eletromecânicos. Como se sabe, grande parte desses sistemas eram projetados e documentados, para fins de manutenção, por meio de Diagramas Ladder. Um diagrama deste tipo pode ser entendido como um “mapa” para guiar as ligações elétricas entre a fonte de alimentação do circuito, os elementos dos relés (contatos e bobinas) e os demais dispositivos do sistemas de automação, como botoeiras, chaves seletoras, pressostatos, termostatos, válvulas, lâmpadas e sirenes.

Dessa forma, na época em que os CLPs surgiram, os Diagramas Ladder eram vastamente difundidos entre os profissionais que trabalhavam com sistemas de automação e, portanto, em algum momento durante o processo de desenvolvimento dos CLPs, alguém se fez uma pergunta que posteriormente se mostrou muito relevante para a história da Informática Industrial: por que não permitir que os CLPs sejam programados por meio de uma linguagem gráfica, que possibilite a representação virtual dos Diagramas Ladder que todos estão habituados a desenvolver e analisar em papel?

Assim surgiu a linguagem de programação que será apresentada neste e nos próximos capítulos. Por ser uma linguagem que, na prática, nasceu junto com os CLPs, tornou-se muito difundida. Somando-se a este fato as vantagens descritas a seguir, essa linguagem se firmou como uma das mais importantes formas de programação relacionadas ao universo industrial, sendo hoje uma das cinco linguagens regulamentadas pela norma internacional IEC 61131-3, adotada pela maioria dos fabricantes desse tipo de dispositivo (John; Tiegkamp, 1995, p. 12).

Uma vez que esta linguagem de programação foi criada com base nos antigos diagramas que documentavam os sistemas de automação baseados em relés eletromecânicos, pode-se dizer que suas principais instruções são representações virtuais de bobinas e contatos do tipo Normal-Aberto (NA) e Normal-Fechado (NF). Por outro lado, elementos como botoeiras, chaves de nível, lâmpadas e válvulas não aparecem nos diagramas virtualizados, ou seja, não são instruções previstas na linguagem de programação. Como visto no capítulo anterior, elementos como esses podem ser conectados às entradas e saídas elétricas do CLP, sendo que seus respectivos sinais elétricos passam a ser monitorados e controlados por meio de valores em formato digital, armazenados em determinadas posições de memória deste dispositivo.

Um exemplo pode ser observado na Figura 3.1, no qual compara-se duas implementações de uma lógica para acender, manter acesa e apagar uma lâmpada a partir de duas botoeiras não retentivas: a primeira por meio de dois relés eletromecânicos (A) e a segunda por meio de um CLP (B) e seu respectivo trecho de programação (C). Neste ponto, o leitor não deve se esforçar para entender a correspondência entre o funcionamento dos dois sistemas, afinal as instruções usadas no trecho de programação ainda não foram abordadas. Em vez disso, neste momento, é interessante notar que os relés *CR-301* e *CR-302* não foram usados no sistema com CLP (sem prejuízo para este último) e que os traços verticais e horizontais que aparecem no trecho de programa desenvolvido com a linguagem Diagrama Ladder podem ser entendidos como cabos elétricos virtuais.

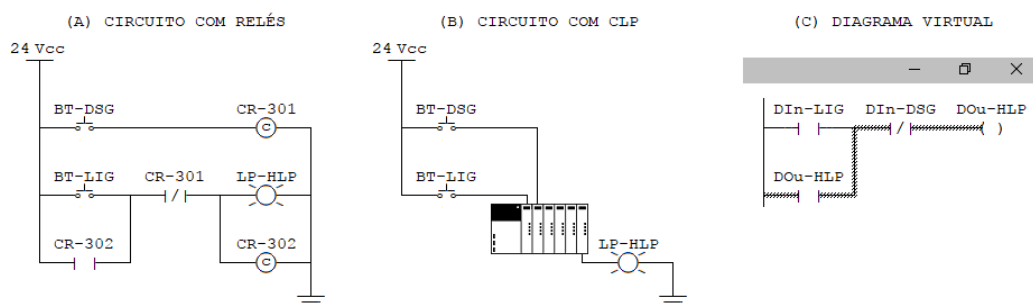


Figura 3.1: Duas implementações de um sistema de sinalização luminosa: a primeira, com base em relés eletromecânicos (A); a segunda, com base em um CLP (B) e seu respectivo trecho de programação, desenvolvido com a linguagem Diagrama Ladder (C).

Dessa forma, conforme discutido na introdução do capítulo anterior, ao se virtualizar elementos elétricos por meio de recursos de programação, a medida em que a complexidade de um sistema de automação aumenta, mais significativas tornam-se as vantagens dos CLPs, ao proporcionarem: **economia de material elétrico, redução do espaço físico ocupado, documentação simplificada do projeto, redução do tempo despendido em desenvolvimentos e alterações** e **versatilidade** suficiente para que sejam aplicados nos mais diversos segmentos industriais. Além disso, ao se considerar as linguagens com as quais os CLPs podem ser programados, outras vantagens aparecem, como **disponibilidade de uma vasta gama de instruções** e, em algumas delas, **possibilidade de monitoração online dos diagramas virtuais**. Ambas são muito importantes e, portanto, merecem os esclarecimentos a seguir.

Conforme já mencionado, pode-se dizer que as principais instruções da linguagem de programação Diagrama Ladder são as representações virtuais de bobinas e contatos NA e NF. No entanto, por ser uma linguagem baseada em um sistema computacional que, sob muitos aspectos, é idêntico a um PC convencional, os programadores podem contar com uma grande diversidade de instruções, incluindo operadores para detecção de borda, para cálculos algébricos e trigonométricos, para comparação de valores decimais, temporizações e muitas outras (as principais serão apresentadas nas próximas seções).

Em complemento, a maioria das plataformas de programação de CLPs vai muito além de prover recursos para a elaboração de Diagramas Ladder virtuais. Entre outros recursos, se o PC em que a plataforma está instalada estiver na mesma rede de dados em que se encontra o CLP, todas as instruções utilizadas em um diagrama previamente descarregado neste dispositivo podem ser monitoradas, bem como os valores que se encontram em suas variáveis. Como exemplo, note os traços hachurados no diagrama que aparece na Figura 3.1C: eles representam uma corrente elétrica virtual, deixando claro para o usuário que a bobina virtual está energizada devido ao fato de que as condições associadas às variáveis *DOu-HLP* e *DIn-DSG* estão satisfeitas, sendo que o mesmo não acontece com a condição que verifica a variável *DIn-LIG*. A importância deste recurso aumenta junto com a complexidade dos diagramas pois, mesmo que um programa esteja totalmente depurado, é comum que muitas de suas condições dependam de elementos físicos, como chaves fim-de-curso, termostatos, medidores de pressão, contatos auxiliares de disjuntores e outros, sendo que rompimentos de cabos ou defeitos em tais elementos podem facilmente levar à parada de um processo de produção. Dessa forma, esse tipo de recurso se torna essencial para as fábricas modernas, a medida em que contribui para reduzir o tempo despendido pelas equipes de manutenção.

Antes do estudo específico das principais instruções que compõem a linguagem Diagrama Ladder, uma última observação: muitas pessoas, ao analisarem trechos de programas como o que pode ser observado na Figura 3.1C, mesmo após estarem em contato com os diagramas virtuais durante várias semanas, continuam atreladas ao comportamento físico dos relés eletromecânicos. Em outras palavras, é comum observar pessoas fazendo análises do tipo: “se a entrada associada à variável *DIn-LIG* receber um sinal elétrico, então o primeiro contato do diagrama irá fechar”; ou, “se a variável *DIn-DSG* estiver em nível lógico alto, o contato associado a ela irá abrir”. Esta não é a forma correta de se analisar um programa desenvolvido com esta linguagem e as pessoas que insistem neste tipo de análise invariavelmente terão problemas ao se depararem com diagramas mais complexos, que envolvem outros tipos de instrução. A postura correta é estar ciente de que o diagrama que se desenvolve ou se analisa é um programa computacional. Dessa forma, nenhum elemento abre e nenhum elemento fecha mas, em vez disso, o que existe são instruções que realizam operações de leitura e/ou escrita nos endereços de memória do CLP, como será mostrado nas próximas seções.

3.2 INSTRUÇÕES BINÁRIAS

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Controladores Lógicos Programáveis	5	5.0.0 à 5.3.4
		5.4.1 à 5.5.2

3.3 INSTRUÇÕES DECIMAIS

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Programmable Logic Controllers	6	6.4 à 6.6

3.4 INSTRUÇÕES MISTAS

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Programmable Logic Controllers	7	7.5

3.5 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Desenvolver e testar três diferentes versões de um pequeno trecho de programação com a linguagem *Diagrama Ladder*, permitindo que, com cada um deles, o CLP da sua bancada seja utilizado para realizar o controle *On/Off* do nível de água no tanque do módulo didático.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo para simulação de processo produtivo;

CONSIDERAÇÕES E TAREFAS PRELIMINARES

Diferentemente do que acontecerá em todas as próximas semanas, o foco das tarefas a serem cumpridas não está no desenvolvimento do trabalho prático cujo início se deu no tópico anterior. A versão mais recente do seu projeto (criado com a plataforma STEP 7) deverá ser utilizada como apoio, mas o principal objetivo nesse momento é o aprendizado das instruções fundamentais da linguagem *Diagrama Ladder*. Sendo assim:

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.



TAREFA 2 – Cadastre as variáveis provisórias “Tpc03_LigBmb”, “Tpc03_NivAux” e “Tpc03_VelAux”, sendo que: a primeira permitirá que a bomba de água, conectada à saída analógica de seu CLP, seja acionada de forma binária (liga ou desliga) durante o Experimento 1; enquanto as últimas serão usadas como auxiliares para operações aritméticas durante o Experimento 3.



EXPERIMENTO 1

Suponha que o sistema de controle On/Off baseado em relés eletromecânicos desenvolvido no primeiro tópico do curso tenha que ser atualizado e que, dessa forma, um CLP será utilizado para acionar a bomba de água de acordo com as informações que recebe das duas chaves de nível. Um bom procedimento para cumprir este objetivo pode ser resumido por meio das tarefas a seguir.

TAREFA 1 – Faça uma análise cuidadosa de todos os elementos conectados às entradas e saídas do seu CLP. Em geral, verifique se os detectores de grandezas físicas, botoeiras e válvulas são do tipo NA ou NF. No caso em questão (lembre-se de que, neste momento, estamos implementando apenas o controle On/Off do nível de água no tanque), basta lembrar que a chave de nível alto (LSH-101) é do tipo NF e que a chave de nível baixo (LSL-101) é do tipo NA.

TAREFA 2 – Em geral, o cadastro das variáveis relacionadas às entradas e saídas do CLP seria a meta desta segunda tarefa, deixando-se claro que as características de instrumentação levantadas na tarefa anterior devem ser incorporadas nos comentários de cada uma delas. No entanto, como tais declarações foram previamente realizadas, apenas analise a Tabela 3.1, na qual as variáveis relacionadas ao problema foram reunidas (repare que os comentários relacionados às chaves de nível são sugestivos, dizendo se elas são do tipo NA ou NF).

Tabela 3.1: Variáveis relacionadas às entradas e saídas do CLP utilizadas na primeira versão do Controlador On/Off do nível de água no tanque.

Nome	Tipo de dado	Endereço	Comentário
DIn_LSH101_AusLiq	Bool	I1.2	Ausência de líquido em cima (chave NF)
DIn_LSL101_PrsLiq	Bool	I1.3	Presença de líquido embaixo (chave NA)
Tpc03_LigBmb	Bool	Q64.6	Tópico 3 – Liga bomba

TAREFA 3 – Uma vez cumpridas as etapas anteriores, o diagrama *Ladder* a ser utilizado para programar o CLP já pode ser desenvolvido, com base apenas nas variáveis envolvidas no problema (incluindo as informações contidas em seus comentários) e no resumo do funcionamento do sistema (que pode ser encontrado na Tabela 3.2). Assim, para começar, use uma **folha de rascunho** para projetar o diagrama desejado, com uma restrição didática: use apenas uma bobina simples para acionar a bomba.

Tabela 3.2: Resumo das combinações de estado das chaves de nível e ações a serem tomadas em cada situação.

Presença de líquido embaixo	Presença de líquido em cima	Ação
Sim	Sim	Desligar bomba
Sim	Não	Manter último estado
Não	Sim	Considerar impossível
Não	Não	Ligar bomba

TAREFA 4 – Considerando que o projeto do seu diagrama *Ladder* está *ok*, encontre, dentro da sua plataforma de programação, o módulo que permite a criação de programas. Em seguida, abra o bloco principal (executado ciclicamente pela CPU) e insira as instruções necessárias (para isso, use o método “arrastar e soltar”).



TAREFA 5 – Compile e faça o *download* completo de seu projeto, ou seja, incluindo as configurações de *hardware*, variáveis cadastradas e o diagrama *Ladder* solicitado na tarefa anterior.



TAREFA 6 – Retornando ao módulo utilizado para desenvolver programas, ative o recurso que permite monitorar seu diagrama *Ladder* e observe o funcionamento do sistema por alguns instantes: caso esteja tudo *ok*, siga para o próximo experimento; caso contrário, tente encontrar o problema ou peça ajuda ao docente.



EXPERIMENTO 2

Deixando de lado as variáveis e instruções binárias, o objetivo deste experimento é a implementação de uma segunda versão do Controlador On/Off de nível, agora usando uma entrada e uma saída analógica do CLP. O foco continua no aprendizado das instruções fundamentais da linguagem *Diagrama Ladder*, porém esta é uma versão mais próxima daquela que será desenvolvida durante as tarefas diretamente relacionadas ao trabalho prático, pois nos dois casos a leitura do nível de água e o acionamento da bomba devem ocorrer por meio das mesmas posições de memória do CLP. De posse dessas informações, cumpra as tarefas indicadas a seguir.

TAREFA 1 – Inicialmente, neste caso, além de levantar informações sobre os instrumentos conectados às entradas e saídas analógicas do CLP, é essencial compreender as características básicas dessas últimas. Felizmente, para o controle On/Off em questão, basta que os seguintes fundamentos fiquem bem estabelecidos:

- Medidor de nível (LT101) – Calibrado para medir no intervalo entre 0 e 200 mm, produzindo tensões correspondentes entre 0 e 10 V;
- Entrada analógica (Conversor A/D) – Configurada para receber tensões entre 0 e 10 V, fornecendo valores decimais correspondentes entre 0 e 27648;
- Saída analógica (Conversor D/A) – Configurada para receber valores decimais entre 0 e 27648, produzindo tensões correspondentes entre 0 e 10 V;
- Bomba de água (BA-101) – Preparada para receber tensões entre 0 e 10 V, fornecendo velocidades correspondentes entre 0 e 100 %.

TAREFA 2 – Após o entendimento da parte instrumental relacionada ao problema, o cadastro das variáveis relacionadas às entradas e saídas do CLP seria agora um passo natural para resolvê-lo. Porém, como o projeto relacionado ao trabalho prático está sendo utilizado como apoio para este experimento, isso já está pronto. Assim, observe a Tabela 3.3 para recapitular as informações utilizadas nas declarações.

Tabela 3.3: Variáveis relacionadas às entradas e saídas do CLP utilizadas na segunda versão do Controlador On/Off do nível de água no tanque.

Nome	Tipo de dado	Endereço	Comentário
AIn_LT101_NivRaw	UInt	IW64	Nível de água no tanque (valor bruto)
AOu_BA101_VelRaw	UInt	QW64	Velocidade da bomba (valor bruto)

TAREFA 3 – Sabendo que as informações fornecidas nas duas tarefas anteriores permitem estabelecer que:

- Se o tanque estiver seco (0 mm), o valor da variável “AIn_LT101_NivRaw” será 0; a medida que o nível de água aumenta, o valor da variável também o faz, proporcionalmente, até que os respectivos limites de 200 mm e 27648 sejam atingidos;

- Se o valor da variável “AOu_BA101_VelRaw” for 0, a bomba estará parada; a medida que o valor da variável aumenta, a velocidade da bomba também o faz, proporcionalmente, até que os respectivos limites de 27648 e 100 % sejam atingidos;

e considerando que a Tabela 3.4 resume os requisitos de funcionamento do controlador a ser implementado, reproduza a Tabela 3.5 em uma **folha de rascunho**, usando algumas regras de 3 para substituir os caracteres ### pelos valores com os quais as entradas e saídas analógicas do CLP trabalham.

Tabela 3.4: Requisitos para acionamento da bomba de água de acordo com o nível aferido no tanque.

Nível do tanque [mm]	Velocidade da bomba [%]
$LT101 > 120$	BA101 = 0
$80 \leq LT101 \leq 120$	Manter último valor
$LT101 < 80$	BA101 = 75

Tabela 3.5: Conversão dos valores de nível e velocidade definidos na Tabela 3.4 para seus correspondentes, a serem lidos em IW64 e escritos em QW64, respectivamente.

Nível do tanque [bruto]	Velocidade da bomba [bruto]
$IW64 > ###$	QW64 = ###
$### \leq IW64 \leq ###$	Manter último valor
$IW64 < ###$	QW64 = ###

TAREFA 4 – Use uma **folha de rascunho** para projetar o diagrama *Ladder* a ser implementado no CLP, de forma que os resultados das comparações entre a variável “AIn_LT101_NivRaw” e as constantes determinadas na tarefa anterior sejam utilizados para determinar qual valor deve ser transferido para a variável “AOu_BA101_VelRaw”.

TAREFA 5 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Utilize o módulo que permite a criação de programas para abrir o bloco no qual a versão anterior do controlador On/Off foi implementada e apague todas as instruções que encontrar (se desejar, tire uma foto para estudos posteriores). Em seguida, implemente a nova versão.

Como?



TAREFA 6 – Faça o *download* das modificações feitas em seu projeto, ou seja, envie o diagrama *Ladder* solicitado na tarefa anterior para o CLP.

Como?



TAREFA 7 – Volte ao módulo usado para desenvolver programas, ative o recurso que permite monitorar seu diagrama *Ladder* e observe o funcionamento do sistema por alguns instantes: caso esteja tudo *ok*, siga para o próximo experimento; caso contrário, tente encontrar o problema ou peça ajuda ao docente.

Como?



EXPERIMENTO 3

Supondo que os limites usados no experimento anterior para determinar a partida e o desligamento da bomba fossem alterados, o que você faria? Novas regras de 3 para atualizar a Tabela 3.5? E se os valores mudassem todo dia? Além dessa necessidade de cálculos feitos “a mão” por parte do programador, a abordagem utilizada no Experimento 2 ainda apresenta outro problema: programas feitos assim são pouco legíveis. Por exemplo, se daqui a uma semana você verificar seu programa, verá que a bomba é acionada toda vez que a variável “AIn_LT101_NivRaw” é menor que a constante 11059, mas o que esse número significa? Você teria que usar uma regra de 3, reversa, para descobrir que essa constante equivale a 80 mm de água no tanque. Felizmente, existe um método simples para resolver esse problema (**por enquanto, é para estudar, não para fazer**):

- Deixe que o CLP faça a regra de 3, dividindo a variável “AIn_LT101_NivRaw” por 27648 e multiplicando o resultado por 200;
- Armazene o resultado final das operações anteriores em uma outra variável, que conterá o valor do nível em milímetros;
- Faça duas linhas em seu diagrama *Ladder*, comparando o valor dessa nova variável com os limites desejados (neste caso, 80 e 120 mm).

Observação: similarmente, os mesmos argumentos se aplicam à velocidade da bomba, ou seja, quando se desejava que sua velocidade fosse 75 %, atribuíam-se a constante 20736 à variável “AOu_BA101_VelRaw”, o que também tornava o código usado no experimento anterior menos amigável. Não seria mais conveniente transferir o valor 75 para uma variável de apoio e deixar o CLP fazer a regra de 3 a partir dela?

Sendo assim, para implementar o controlador On/Off utilizando esse novo conceito, realize as tarefas indicadas seguir (**agora é para fazer**).

TAREFA 1 – O medidor de nível, a bomba de água e as entradas e saídas do CLP são as mesmas utilizadas no experimento anterior, tornando desnecessário o estudo de suas características neste momento. Além disso, neste ponto, todas as variáveis envolvidas no problema já estão declaradas em seu projeto. Analisando a Tabela 3.6, observe que as quatro primeiras fazem parte do trabalho prático iniciado no tópico anterior e que as duas últimas foram cadastradas durante as tarefas preliminares desta aula (o propósito de cada uma será exposto mais a frente).

Tabela 3.6: Variáveis utilizadas na terceira versão do Controlador On/Off do nível de água no tanque: entrada e saída analógica; nível e velocidade em unidade de engenharia; auxiliares para operações aritméticas.

Nome	Tipo de dado	Endereço	Comentário
AIn_LT101_NivRaw	UInt	IW64	Nível de água no tanque (valor bruto)
AOu_BA101_VelRaw	UInt	QW64	Velocidade da bomba (valor bruto)
Ana_LT101_NivEng	Real	MD704	Nível de água no tanque (milímetros)
Ana_BA101_VelEng	Real	MD712	Velocidade da bomba (percentual)
Tpc03_NivAux	Real	MD12	Tópico 3 – Nível auxiliar
Tpc03_VelAux	Real	MD16	Tópico 3 – Velocidade auxiliar

TAREFA 2 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Utilize o módulo que permite a criação de programas para abrir o bloco no qual a versão anterior do controlador On/Off foi implementada e apague todas as instruções que encontrar (se desejar, tire uma foto para estudos posteriores). Em seguida, implemente a nova versão, estruturando-a da seguinte forma:



Linha 1 – Conversão de tipo de dado para o nível do tanque: o valor inteiro e sem sinal em *AIn_LT101_NivRaw* deve ser convertido para um número real e armazenado na variável auxiliar *Tpc03_NivAux*;

Linha 2 – Implementação da regra de 3 para conversão do nível do tanque: o valor adimensional, mas agora em formato real, que foi armazenado em *Tpc03_NivAux* deve sofrer, primeiramente, uma divisão pela constante 27648.0 e este resultado parcial deve ser novamente armazenado em *Tpc03_NivAux*. Em seguida, multiplique-o por 200.0 e armazene o resultado final em *Ana_LT101_NivEng*;

Linha 3 – Comparação para desligamento da bomba: se o valor, agora em milímetros, armazenado em *Ana_LT101_NivEng* for maior que 120.0, transfira o valor 0.0 para a variável *Ana_BA101_VelEng*, que representa o percentual de velocidade desejado para bomba;

Linha 4 – Comparação para ativação da bomba: se o valor, agora em milímetros, armazenado em *Ana_LT101_NivEng* for menor que 80.0, transfira o valor 75.0 para a variável *Ana_BA101_VelEng*, que representa o percentual de velocidade desejado para bomba;

Linha 5 – Implementação da regra de 3 para conversão de velocidade: o percentual de velocidade em *Ana_BA101_VelEng*, deve sofrer, primeiramente, uma divisão por 100.0 e este resultado parcial deve ser armazenado na variável auxiliar *Tpc03_VelAux*. Em seguida, multiplique-o pela constante 27648.0 e armazene o resultado final novamente em *Tpc03_VelAux*;

Linha 6 – Conversão de tipo de dado para a velocidade: o valor real em *Tpc03_VelAux* deve ser convertido para um número inteiro sem sinal e armazenado em *AOu_BA101_VelRaw*.

TAREFA 3 – Faça o *download* das modificações feitas em seu projeto, ou seja, envie o diagrama *Ladder* solicitado na tarefa anterior para o CLP.



TAREFA 4 – Volte ao módulo usado para desenvolver programas, ative o recurso que permite monitorar seu diagrama *Ladder* e observe o funcionamento do sistema por alguns instantes: caso esteja tudo *ok*, siga para a parte final da aula; caso contrário, tente encontrar o problema ou peça ajuda ao docente.



FINALIZANDO

TAREFA 1 – Reflita: o diagrama *Ladder* elaborado durante o Experimento 3 é, de certa forma, mais complexo do que o elaborado durante o Experimento 2, inclusive porque utiliza mais posições de memória e mais instruções de programação. No entanto, o último é inegavelmente mais inteligível do que o anterior, cabendo a questão: qual dos dois é melhor? Ou, em outras palavras, você prefere desempenho ou clareza do código?

A resposta depende de vários fatores, envolvendo exigências da planta e os recursos disponíveis no *hardware* utilizado para automatizá-la. No entanto, devido ao poder de processamento e a disponibilidade de memória dos CLPs atuais, pode-se deixar uma dica: em situações como as de hoje, é muito provável que as diferenças de desempenho e consumo de memória sejam mínimas, sendo recomendável a preferência pela clareza do código, que foi significativamente elevada com o método utilizado no último experimento. Lembrando que, quanto mais complexa for a planta, mais benefício a clareza pode trazer, podendo economizar várias horas da equipe de manutenção e evitar várias horas de mal funcionamento de um processo de produção industrial.

TAREFA 2 – Se sua plataforma de desenvolvimento estiver *online* com seu CLP, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.



TAREFA 3 – Se estiver tudo *ok* e não houver dúvidas em relação ao que foi desenvolvido, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

3.6 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

Calibrar o medidor de nível de água no tanque (LT101) por meio de um ensaio que abrange todas as etapas do sistema de medição dessa grandeza, de forma que o CLP passe a fornecer valores de nível muito próximos à realidade, após alguns ajustes em sua programação.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo para simulação de processo produtivo.

RETOMANDO

Na última prática de laboratório, entre outras coisas, foi desenvolvido um trecho de programa dedicado à medição do nível de água no tanque do módulo didático, sendo que a conversão dos valores adimensionais fornecidos pela entrada analógica do CLP para seus correspondentes em milímetros se baseou na seguinte **relação teórica**:

- se o nível de água no tanque for igual a 0 mm, o valor digitalizado será 0; a medida que o primeiro aumenta, o último também o faz, proporcionalmente, até que os respectivos limites de 200 mm e 27648 sejam atingidos.

Tal relação foi obtida a partir das características da entrada analógica e do instrumento de medição utilizados, bem como das configurações realizadas nestes elementos. No entanto, nesse momento, provavelmente há uma discrepância significativa entre o nível de água que pode ser observado fisicamente no tanque e o valor fornecido pelo CLP, sendo que as causas mais prováveis são:

- simplicidade da parte eletrônica do medidor embarcado no módulo didático que, entre outras diferenças em relação aos que são utilizados em processos industriais, não faz a compensação do desvio que a temperatura ambiente causa na medição;
- o medidor utilizado não fornece em sua saída um sinal entre 0 e 10 V, esperado pela entrada analógica do CLP. Para que haja compatibilidade são usados conversores eletrônicos intermediários, que acabam acrescentando desvios na medição;
- imperfeições na construção ou falta de manutenção do módulo didático, levando à situações em que o medidor possa estar mais próximo ou mais afastado do fundo do tanque do que o valor adotado durante sua configuração.

Dessa forma, nessa e em qualquer outra situação em que for possível, é recomendável substituir a **relação teórica** que rege a conversão linear por uma **relação prática**, na qual os possíveis desvios descritos anteriormente passam a ser naturalmente considerados pelo sistema, deixando de causar erros na medição. Para fazer isso, cumpra as tarefas a seguir.

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.

Como?



TAREFA 2 – Utilize o módulo que permite a criação de programas para abrir o bloco principal (executado ciclicamente pela CPU) e apague todas as instruções programadas durante a última aula prática (se desejar, tire uma foto para estudos posteriores).

Como?



TAREFA 3 – Cadastre as variáveis provisórias a serem utilizadas como auxiliares para operações aritméticas durante o experimento a seguir, sendo elas: “Tpc03_CLinX”, “Tpc03_CLinA”, “Tpc03_CLinB”, “Tpc03_CLin1” e “Tpc03_CLin2”.

Como?



TAREFA 4 – Compile e faça *download* do seu projeto, que agora contém as variáveis cadastradas na tarefa anterior.

Como?



TAREFA 5 – Reproduza a Tabela 3.7 em uma **folha de rascunho**. Para preencher suas lacunas, primeiramente, encontre o módulo que torna possível **a monitoração e a alteração de valores na memória do CLP** e abra a tabela “**Teste de I/O**”.

Como?



Tabela 3.7: Apoio para anotação dos valores levantados durante o ensaio para calibração do medidor de nível.

Parâmetro	Valor
X1	
X2	
Y1	70.0
Y2	130.0

TAREFA 6 – Para encontrar o valor do parâmetro X1, faça manobras com a bomba de água ou com as válvulas solenóides (escrevendo nas posições de memória associadas às saídas elétricas que as comandam) de modo a estabilizar o nível da água no ponto referente a 70 mm da escala fixada no tanque do módulo didático. Quando isso ocorrer, o valor procurado poderá ser observado na variável “AIn_LT101_NivRaw” (que também está relacionada na tabela “**Teste de I/O**”). Anote-o.

Como?



TAREFA 7 – Repita o que foi feito na tarefa anterior para, agora, estabilizar o nível da água no ponto referente a 130 mm da escala fixada no tanque e, assim, anotar o valor de X2.

IMPLEMENTAÇÃO

A principal diferença entre o que foi realizado ao final da última aula prática e o objetivo atual é a seguinte: anteriormente, os valores de nível (em milímetros) podiam ser obtidos por meio de uma regra de 3, usada para resolver o caso particular de um sistema linear em que a reta passa pela origem do plano cartesiano (Figura 3.2A); agora, esses valores devem ser obtidos por meio de um sistema com duas incógnitas e duas equações, usado para descrever qualquer relação linear (Figura 3.2B).

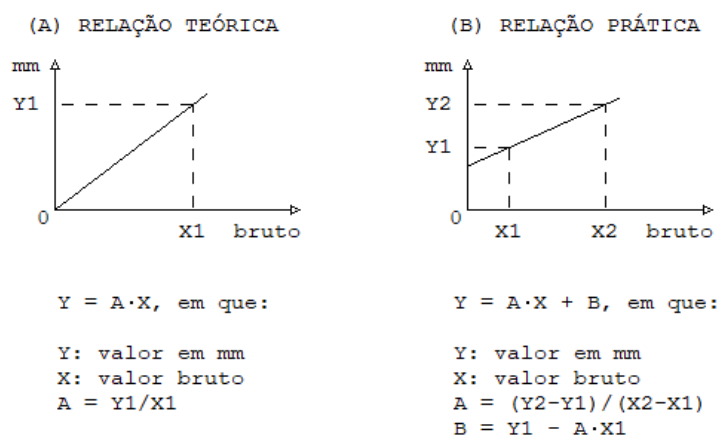


Figura 3.2: Relações para conversão dos valores adimensionais fornecidos pela entrada analógica para valores em milímetros: **relação teórica** baseada nas características do sistema de medição (A) e **relação prática** obtida a partir do ensaio recém-realizado (B).

De posse dessas informações, cumpra as tarefas a seguir para implementar e testar o sistema linear que deverá converter os valores adimensionais fornecidos pelo sistema de medição do nível de água em seus correspondentes em milímetros. Note que, para isso, serão utilizados os parâmetros levantados no ensaio experimental que você acabou de realizar.

TAREFA 1 – Utilize o módulo que permite a criação de programas para abrir o bloco principal (executado ciclicamente pela CPU) e implemente o diagrama referente ao conversor linear, estruturando-o da seguinte forma:

Linha 1 – Conversão de tipo de dado para o nível da água: converta o valor inteiro e sem sinal armazenado em *AIN_LT101_NivRaw* para um número real e armazene-o em *Tpc03_CLinX*;

Linha 2 – Cálculo de A (coeficiente angular): substituindo *X1*, *X2*, *Y1* e *Y2* pelos valores que estão na Tabela 3.7, faça a subtração entre *Y2* e *Y1* e armazene o resultado em *Tpc03_CLin1*. Em seguida, faça a subtração entre *X2* e *X1* e armazene o resultado em *Tpc03_CLin2*. Por fim, faça a divisão entre *Tpc03_CLin1* e *Tpc03_CLin2* e armazene o resultado em *Tpc03_CLinA*;

Linha 3 – Cálculo de B (coeficiente linear): substituindo *X1* e *Y1* pelos valores que estão na Tabela 3.7, multiplique *Tpc03_CLinA* por *X1* e armazene o resultado em *Tpc03_CLin1*. Em seguida, faça a subtração entre *Y1* e *Tpc03_CLin1* e armazene o resultado em *Tpc03_CLinB*;

Linha 4 – Cálculo de Y (valor convertido): multiplique *Tpc03_CLinA* por *Tpc03_CLinX* e armazene o resultado em *Tpc03_CLin1*. Em seguida, faça a soma entre *Tpc03_CLin1* e *Tpc03_CLinB* e armazene o resultado em *Ana_LT101_NivEng*.

TAREFA 2 – Salve seu projeto, compile e faça *download* das modificações realizadas. Volte ao bloco principal (executado ciclicamente pela CPU), ative o modo de monitoração e verifique se a discrepância entre o nível de água que pode ser observado fisicamente no tanque e o valor fornecido pelo CLP foi reduzida (tenha em mente que, com o uso de um medidor cuja eletrônica é simplificada, erros de até 5 mm são esperados).

TAREFA 3 – Para completar a validação do procedimento de calibração, volte à tabela “**Teste de I/O**” e manobre os atuadores pertinentes, causando variações no nível de água e verificando a coerência dos valores medidos em outros pontos da escala. Caso esteja tudo *ok*, siga em frente; caso contrário, tente encontrar o problema e, se não conseguir, peça ajuda ao docente.

FINALIZANDO

TAREFA 1 – Reflita: o diagrama *ladder* que você acabou de elaborar pode ser considerado simples e funcional. No entanto, se vários medidores de grandezas físicas fossem tratados da mesma maneira em seu CLP, o projeto poderia acabar ficando confuso. O que fazer, então?

Resposta: felizmente, este problema pode ser resolvido pela organização e encapsulamento de código em sub-rotinas e funções (que são justamente os assuntos explorados no próximo tópico do curso).

TAREFA 2 – Se sua plataforma de desenvolvimento estiver *online* com seu CLP, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.

Como?



TAREFA 3 – Quando encerrar suas atividades, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

3.7 EXERCÍCIOS

QUESTÃO 1 – Analisando a lógica apresentada na Figura 3.3, responda: se as entradas digitais “Liga” e “Desliga” estiverem em nível lógico 1 ao mesmo tempo, em algum momento durante a execução do programa, o terminal elétrico da saída digital associada ao símbolo “Motor 1” será energizado? Justifique sua resposta.

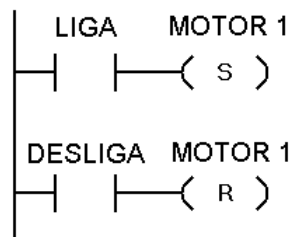


Figura 3.3: Diagrama Ladder a ser analisado.

QUESTÃO 2 – Existem 4 erros de endereçamento no pequeno trecho de Ladder mostrado na Figura 3.4, desenvolvido para o CLP S7-1200. Identifique-os e posteriormente, para cada um desses erros, faça uma ou duas frases explicando a incoerência existente. Dica: você não precisa saber o propósito dessa sequência de operações para identificar os erros.

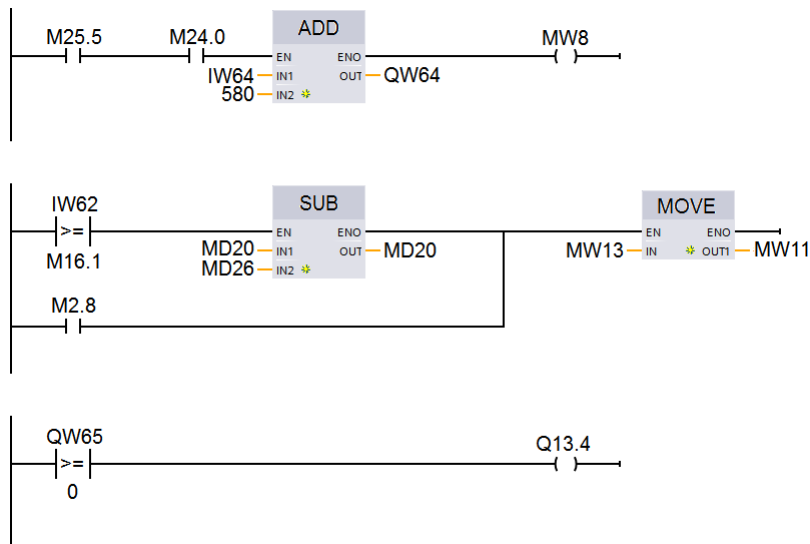


Figura 3.4: Diagrama Ladder contendo os 4 erros de endereçamento a serem identificados.

QUESTÃO 3 – Considere o projeto de interligação elétrica e o trecho do Diagrama Ladder apresentados na Figura 3.5.

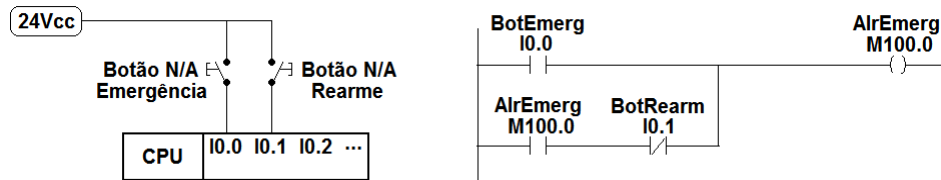


Figura 3.5: Projeto de interligação elétrica e trecho do Diagrama Ladder a serem analisados.

Supondo que a variável *AlrEmerg* (M100.0) venha a ser utilizada em vários outros pontos do programa, para bloquear o funcionamento dos equipamentos da planta caso o botão de emergência seja pressionado, pode-se apontar o seguinte problema: se o fio que conecta o botão de emergência à entrada I0.0 se romper, nada acontecerá no momento do rompimento. Quando alguém precisar usar o botão de emergência, ele não funcionará, podendo gerar graves acidentes. Com o objetivo de sanar esse problema, proponha:

- Uma alteração no projeto de interligação elétrica (refaça o diagrama elétrico).
- A adequação necessária na linha de programação mostrada acima, de forma que o sistema de emergência passe a funcionar com a interligação elétrica proposta por você no item anterior (refaça o diagrama ladder).

QUESTÃO 4 – ENADE 2008 – No processo de tratamento de efluentes de uma unidade industrial, parte do programa aplicativo, representando a lógica de controle, está mostrado na Figura 3.6. Os dispositivos de entrada (*E0* até *E4*) são, fisicamente, chaves que estão conectadas a um módulo de entrada do Controlador Lógico-Programável (CLP). As chaves (*E1*, *E2* e *E3*) são normalmente abertas e as chaves (*E0* e *E4*), normalmente fechadas. Ao módulo de saída do CLP está conectada a bobina do contato auxiliar que liga a bomba elétrica dosadora.

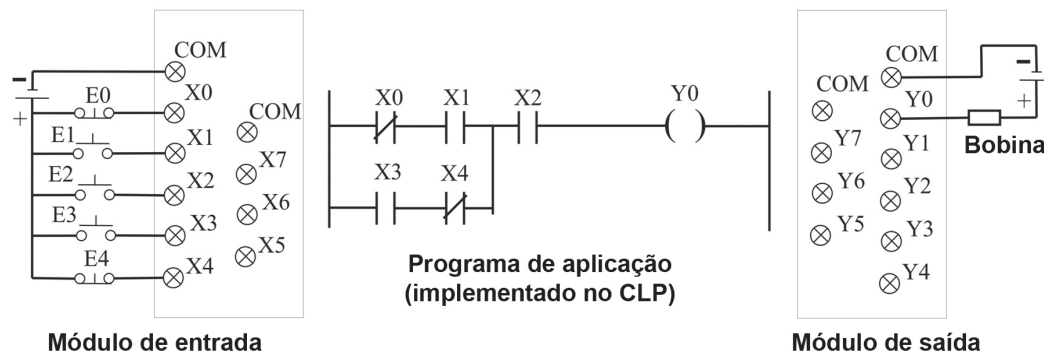


Figura 3.6: Interligação elétrica das botoeiras e *software* aplicativo a serem analisados.

Quais chaves devem ser acionadas para que a bomba dosadora seja ligada?

- E1* e *E2*
- E2* e *E3*
- E3* e *E4*
- E0*, *E1* e *E2*
- E1*, *E2* e *E3*

QUESTÃO 5 – Apresente um trecho de programa em Diagrama Ladder que realize as seguintes tarefas:

- Selecionar três modos de operação a partir de um único *push-button*. Toda vez que for pressionado, o *push-button* deverá causar uma única transição, da seguinte forma: Automático → Manual → Manutenção → Automático → ...
- Indicar o modo de operação selecionado por meio de um único *led*, da seguinte forma:
 - Automático ↔ *led* aceso;
 - Manual ↔ *led* apagado;
 - Manutenção ↔ *led* piscando.

As variáveis mostradas na Tabela 3.8 devem, obrigatoriamente, fazer parte do seu programa. Outras variáveis também podem ser utilizadas, porém, para que elas sejam consideradas válidas durante a correção, você deve “declará-las” em uma tabela similar a esta.

Tabela 3.8: Lista de variáveis que devem, obrigatoriamente, fazer parte programa solicitado (outras variáveis também podem ser utilizadas mas, para serem consideradas válidas, devem ser declaradas em uma tabela similar a esta, a ser apresentada junto ao Diagrama Ladder).

Símbolo	Endereço	Tipo	Comentário
BotSel	I0.0	Bool	Botão para selecionar o modo de operação
LedInd	Q0.0	Bool	Led para indicar o modo de operação selecionado
AutSel	M0.0	Bool	Modo automático selecionado
MaaSel	M0.1	Bool	Modo manual selecionado
MtcSel	M0.2	Bool	Modo manutenção selecionado
Pul0d5	M0.3	Bool	Pulso de 0.5 segundos a cada 0.5 segundos

Considere também que os pulsos em M0.3 já estão sendo gerados em outro trecho do programa, ou seja, você pode usar esta variável sem se preocupar em apresentar as instruções que escrevem neste endereço.

QUESTÃO 6 – Um removedor circular realiza ciclos de 360° para transportar peças de um determinado processo. Em modo manutenção, o acionamento do motor desse equipamento (M) depende exclusivamente de uma botoeira não-retentiva (L) e de um sensor de posição básica (S). Faça um trecho de programa em Ladder para que o equipamento se comporte da seguinte maneira:

- Inicialmente o equipamento encontra-se na posição básica (sensor S atuado);
- O motor (M) deve ser acionado apenas enquanto a botoeira (L) estiver pressionada;
- Porém, ao completar o ciclo (sensor S atuado novamente), o motor deve ser desligado e só poderá voltar a funcionar caso a botoeira seja liberada e pressionada novamente.

QUESTÃO 7 – Supondo que todas as variáveis envolvidas na lógica a seguir começam em nível lógico 0 e que a variável *Clk* se comporta como mostrado na Figura 3.7, desenhe as formas de onda das variáveis *V*, *W*, *X*, *Y* e *Z*, considerando os estados em que elas se encontram ao final de cada ciclo do CLP.

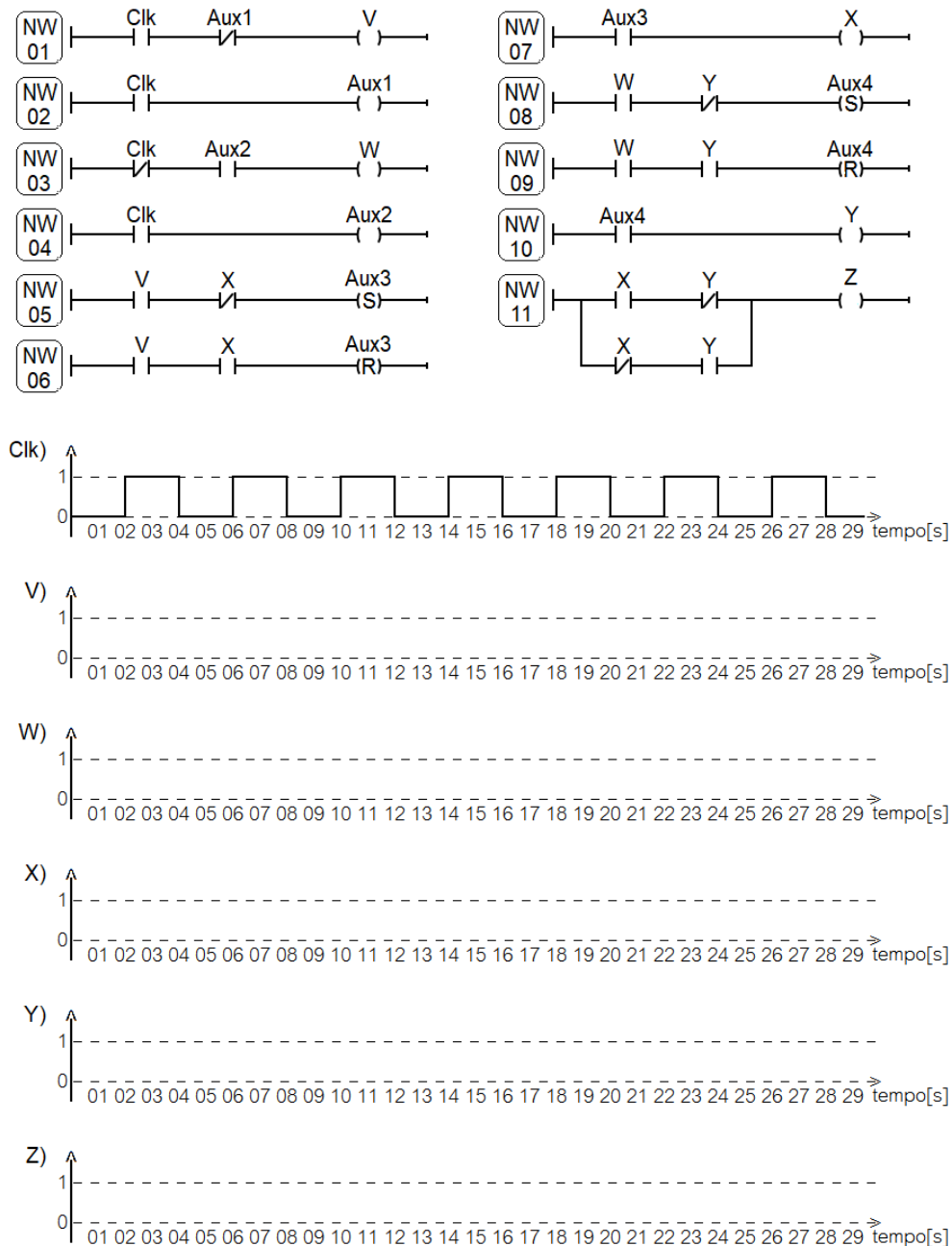


Figura 3.7: Comportamento da variável *Clk* e suporte para desenho das formas de onda solicitadas.

QUESTÃO 8 – O diagrama esquemático *simplificado* do sistema de automação a ser programado pode ser observado na Figura 3.8. Considere também as características de instrumentação apresentadas no quadro logo em seguida e, posteriormente, realize as tarefas solicitadas.

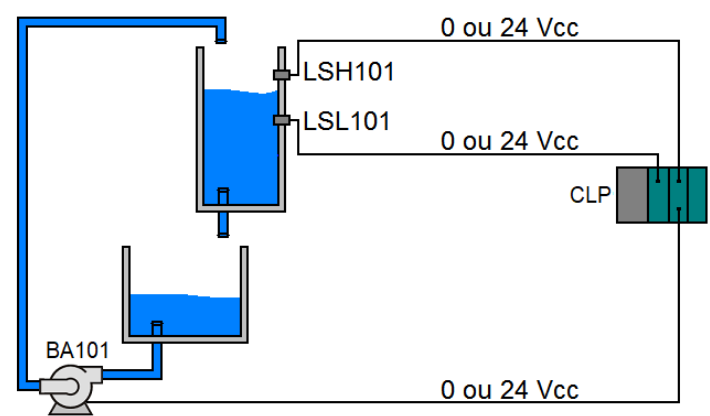


Figura 3.8: Tanque de água alimentado por uma bomba elétrica, com vazão de saída constante e chaves de nível para detecção das condições alto e baixo.

<ul style="list-style-type: none">Chaves de nível (alto e baixo):<ul style="list-style-type: none">Presença de líquido = 0 VccAusência de líquido = 24 Vcc	<ul style="list-style-type: none">Entradas digitais do CLP:<ul style="list-style-type: none">24 Vcc = 10 Vcc = 0
---	---

a) Controle o nível usando as chaves de nível LSL101 e LSH101 e uma bobina simples para acionar a bomba BA101, de acordo com as seguintes regras:

Presença líquido embaixo	Presença líquido em cima	Ação
Não	Não	Ligar
Não	Sim	Considerar impossível
Sim	Não	Manter último estado
Sim	Sim	Desligar

b) Com o objetivo de aperfeiçoar o sistema de controle On/Off desenvolvido no item a), inclua um botão para bloqueá-lo e habilitá-lo. Ou seja, se o sistema estiver bloqueado, um toque no botão deverá habilitá-lo (permitindo que a bomba funcione quando necessário). Por outro lado, se o sistema estiver habilitado, um toque no botão deverá bloqueá-lo (impedindo o funcionamento da bomba em qualquer situação). Supondo que este botão tenha sido conectado eletricamente à entrada digital I0.2 do CLP, apresente um Diagrama Ladder completo para esse novo sistema.

3.8 REFERÊNCIAS

FRANCHI, C.M.; CAMARGO, V.L.A. de. **Controladores Lógicos Programáveis: Sistemas Discretos**. Érica, 2008. 352 p. ISBN 9788536501994.

JOHN, K.H.; TIEGELKAMP, M. **IEC 61131-3: Programming Industrial Automation Systems**: Concepts and Programming Languages, Requirements for Programming Systems, Aids to Decision-making Tools. Springer, 1995. 381 p. ISBN 9783662078471.

REHG, J.A.; SARTORI, G.J. **Programmable Logic Controllers**. 2. ed.: Pearson Prentice Hall, 2009. 600 p. ISBN 9780135048818.

Capítulo 4

PROGRAMAÇÃO EM DIAGRAMA LADDER: SUB-ROTINAS E FUNÇÕES

4.1 INTRODUÇÃO

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação).

4.2 SUB-ROTINAS

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Programmable Logic Controllers	8	8.3.3

4.3 FUNÇÕES

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Programmable Logic Controllers	8	8.3.5

4.4 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Estruturar o projeto referente ao trabalho prático a partir da criação de todos os blocos de programação que abrigarão as sub-rotinas e funções a serem utilizadas ao longo de seu desenvolvimento; além de programar a função ESCLIN (Escalonamento linear) e a sub-rotina LT101 (Nível de água no tanque).

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.

CONSIDERAÇÕES E TAREFAS PRELIMINARES

Ao cumprir as tarefas extraclasse indicadas no tópico anterior, você deve ter criado um trecho de diagrama *ladder* similar ao que pode ser observado na Figura 4.1.

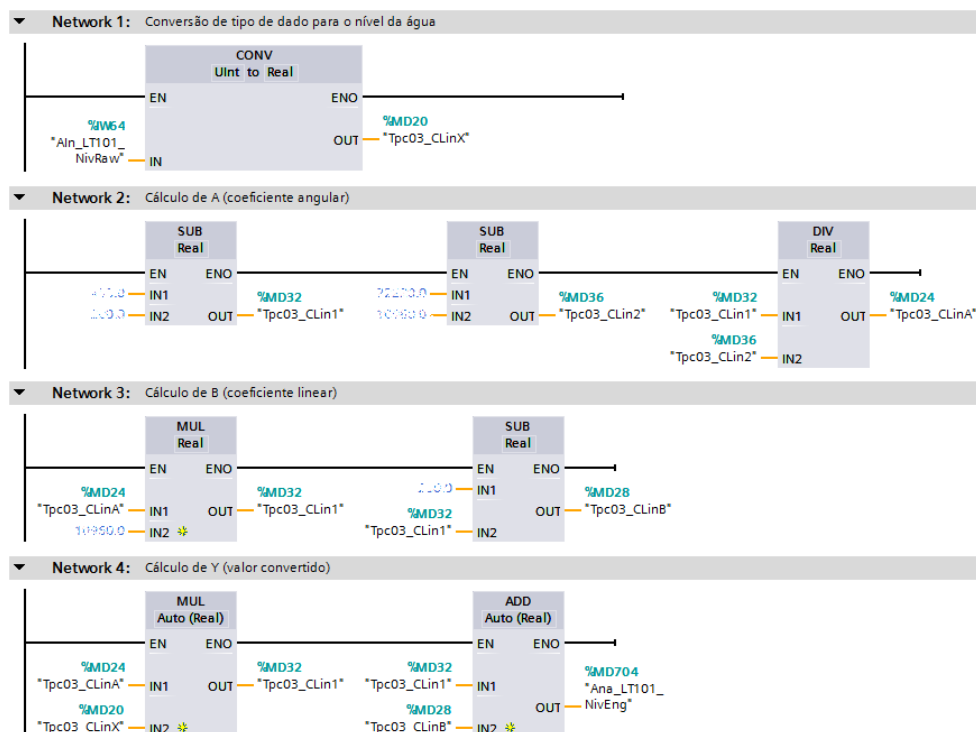


Figura 4.1: Trecho de diagrama *ladder* responsável pela conversão do valor fornecido pela entrada analógica para milímetros (específico para o caso das últimas tarefas extraclasse).

Conforme discutido naquele momento, o trecho de diagrama em questão permite que o programador possa trabalhar com valores em unidade de engenharia, o que facilita desenvolvimentos e manutenções posteriores. No entanto, se as quatro linhas de código usadas para alcançar esse objetivo precisarem ser repetidas, para o caso em que várias conversões similares sejam necessárias, o projeto poderá se tornar confuso. Para resolver esse problema, uma boa opção é generalizar e encapsular as *três* últimas linhas que aparecem na Figura 4.1 em uma função, como a que pode ser observada na Figura 4.2.

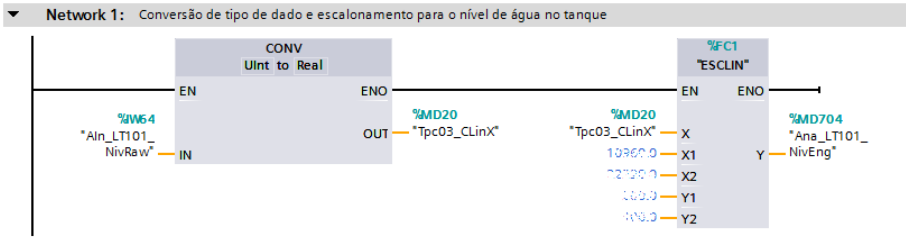


Figura 4.2: Solução alternativa para o problema tratado na Figura 4.1, por meio de uma função genérica, configurada para tal.

TAREFA 1 – O passo inicial para se implementar a função ESCLIN é definir sua estrutura de dados, ou seja, quais serão seus “pinos” de entrada e de saída e quais serão suas variáveis temporárias. Sendo assim, reproduza a Tabela 4.1 em uma folha de rascunho, completando suas lacunas. É importante ter em mente que essas serão as variáveis locais de uma função **genérica**, que poderá ser útil não só para conversões de nível, mas também para conversões de temperatura, pressão, velocidade etc, ou seja, **não** se deve escolher nomes ou fazer comentários para tais variáveis que sejam específicos para um determinado caso.

Tabela 4.1: Variáveis internas da função ESCLIN.

Categoria	Nome	Tipo de dado	Comentário
Input	X	Real	Escala X - Valor a ser convertido
Input			
Input			
Input			Escala Y - Ponto correspondente 1
Input			
Output	Y		
Temp			Coefficiente angular
Temp			Coefficiente linear
Temp	r01Aux		
Temp			Temporário real 02

TAREFA 2 – Use uma folha de rascunho para projetar o diagrama *ladder* a ser encapsulado na função ESCLIN. Mais uma vez, deve-se ressaltar que o código de uma função deve ser genérico. Por exemplo, no sistema que está sendo projetado, a função ESCLIN servirá tanto para escalonar o nível como a temperatura da água no tanque. *Dica:* não se preocupe com isso ainda, apenas lembre-se de **não** usar constantes específicas ou variáveis globais dentro de sua função.

TAREFA 3 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.

Como?



TAREFA 4 – Crie o bloco de programação referente à função ESCLIN e faça a implementação do que foi planejado nas tarefas anteriores. Primeiramente, declare as variáveis locais no cabeçalho da função. Em seguida, transcreva o diagrama *ladder*.

Como?



TAREFA 5 – Volte ao bloco principal (executado ciclicamente pela CPU) e substitua as *três* últimas linhas do diagrama *ladder* atual pela função ESCLIN. Como parâmetros de entrada e saída, use as constantes levantas na atividade extraclasses do tópico anterior (Tabela 3.7) e as variáveis que aparecem na Figura 4.2.

TAREFA 6 – Salve seu projeto, compile e faça *download* das modificações realizadas. Ative o modo de monitoração e verifique a coerência dos valores fornecidos pela função ESCLIN (para causar variações o nível de água, use a tabela “**Teste de I/O**” para manobrar os atuadores pertinentes). Se estiver tudo *ok*, siga em frente; caso contrário, tente encontrar o problema e, se não conseguir, peça ajuda ao docente.

ESTRUTURAÇÃO EM SUB-ROTINAS E FUNÇÕES

Note que muitas atividades realizadas nos últimos dias foram essencialmente didáticas (tanto que vários diagramas foram apagados e reconstruídos). No entanto, agora que você adquiriu os conhecimentos necessários, o desenvolvimento do sistema de automação proposto no início do curso será retomado. No primeiro momento, outras funções (além da ESCLIN) serão incorporadas ao projeto. Em seguida, os blocos de programação referentes às sub-rotinas dos medidores, atuadores e outros elementos serão criados e preparados para serem executados ciclicamente a partir do bloco principal. Por fim, duas sub-rotinas especiais serão configuradas: a primeira atrelada à interrupções cíclicas e a segunda para facilitar a inicialização de variáveis. Para isso, cumpra as tarefas a seguir.

TAREFA 1 – Por meio dos *links* fornecidos na Tabela 4.2, faça a importação das funções EASYPI (Controlador PI genérico), HORMTR (Horímetro regressivo) e MEMRST (Limpeza de áreas de memória), a serem utilizadas em momento oportuno.

Como?



TAREFA 2 – Crie todos os blocos de programação referentes às sub-rotinas indicadas na Tabela 4.3 (atenção à escolha da linguagem, que deve ser diferente para duas delas). Os respectivos códigos internos serão desenvolvidos ao longo do curso.

Como?



TAREFA 3 – Crie as sub-rotinas especiais LC102 (Controlador PI de nível) e *Startup* (Tarefas iniciais), de acordo com as informações disponíveis na Tabela 4.4.

Como?



Tabela 4.2: Outras funções a serem incorporadas ao projeto (para utilização futura), em complemento à função ESCLIN (previamente desenvolvida).



Nome	Tipo	Linguagem	Título	Link
EASYPI	FC	SCL	Controlador PI genérico	
HORMTR	FC	SCL	Horímetro regressivo	
MEMRST	FC	SCL	Limpeza de áreas de memória	

Tabela 4.3: Sub-rotinas a serem desenvolvidas pelos estudantes ao longo do curso e chamadas de forma incondicional dentro do bloco *Main*.

Nome	Tipo	Linguagem	Título
GERAL	FC	LAD	Elementos gerais do sistema
LT101	FC	LAD	Nível de água no tanque
TT101	FC	LAD	Temperatura da água
MX101	FC	LAD	Mixer
BA101	FC	LAD	Bomba de água
AQ101	FC	LAD	Aquecedor
VT101	FC	LAD	Ventilador
VD101	FC	LAD	Válvula de dreno direto
VD102	FC	LAD	Válvula de dreno refrigerado
LC101	FC	FBD	Controlador On/Off de nível
TC101	FC	FBD	Controlador On/Off de temperatura
SQBAT	FC	LAD	Sequenciador para bateladas
SQAUX	FC	LAD	Lógicas auxiliares do sequenciador

Tabela 4.4: Sub-rotinas especiais, cuja execução está associada a interrupções geradas pelo *hardware* do CLP.

Nome	Tipo	Linguagem	Título	Observação
LC102	OB	LAD	Controlador PI de nível	Sub-rotina executada por meio de interrupções cíclicas (a cada 100 ms).
Startup	OB	LAD	Tarefas iniciais	Sub-rotina executada apenas no primeiro ciclo do CLP (quando há uma transição <i>Stop</i> → <i>Run</i>).

PRIMEIROS PASSOS APÓS A ESTRUTURAÇÃO

Nesse ponto, todos os blocos de programação previstos para o CLP já foram criados em seu projeto. No entanto, o conteúdo (código interno) da maioria deles ainda precisa ser desenvolvido ou organizado. Isso será feito aos poucos, ao longo do curso, começando com o bloco principal (executado ciclicamente pela CPU) e com a sub-rotina relacionada à medição de nível (LT101), conforme orientações fornecidas por meio das tarefas a seguir.

TAREFA 1 – Começando a organizar o que já está pronto, transfira o conteúdo atual do bloco *Main* para o bloco LT101. Volte ao primeiro e inclua chamadas incondicionais para cada uma das sub-rotinas indicadas na Tabela 4.3 (faça uma chamada por linha e atribua títulos pertinentes a cada uma). Ao final, seu projeto deverá ter uma estrutura similar à que pode ser observada na Figura 4.3.

Como?

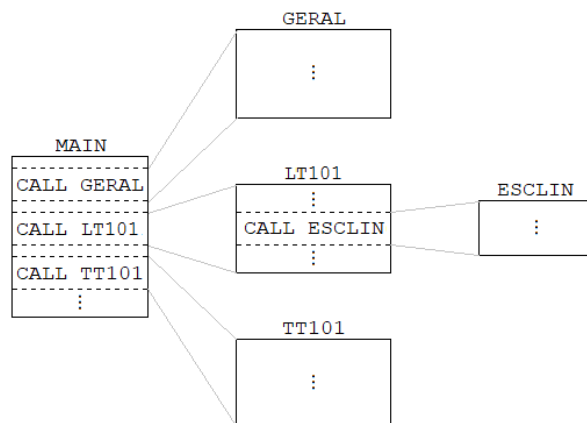


Figura 4.3: Diagrama esquemático que resume parte da sequência de chamadas usada para organizar o projeto (até o momento).

TAREFA 2 – Ainda dentro do bloco *Main*, acrescente ao final uma chamada condicional para a função MEMRST, fazendo com que ela seja executada uma vez a cada segundo. Em seguida, preencha seus “pinos” de entrada, de forma que todas as variáveis do grupo “Comandos” sejam levadas a nível lógico 0 sempre que a execução ocorrer (a justificativa para isso será discutida no Capítulo 6).

Como?



TAREFA 3 – Abra o bloco referente à sub-rotina LT101 e adicione as Linhas 2, 3, 4 e 5. Para fazer a programação necessária, consulte as descrições fornecidas a seguir. Observe que, se você cumpriu todas as tarefas até o momento, a **Linha 1** já estava presente (e não será necessário alterá-la).

Linha 2 – Nível muito alto (alarme): a variável associada a este alarme deverá estar em nível lógico 1 se e somente se o valor da grandeza monitorada for **maior** do que o limite determinado por seu respectivo parâmetro.

Linha 3 – Nível alto (alarme): similar à anterior.

Linha 4 – Nível baixo (alarme): a variável associada a este alarme deverá estar em nível lógico 1 se e somente se o valor da grandeza monitorada for **menor** do que o limite determinado por seu respectivo parâmetro.

Linha 5 – Nível muito baixo (alarme): similar à anterior.

TAREFA 4 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações propostas ao longo das tarefas anteriores.

TAREFA 5 – Abra o bloco LT101 e ative o recurso que permite monitorar seu diagrama. Note que os parâmetros que determinam o limite de disparo de cada alarme não foram inicializados, ou seja, as variáveis “Par_LT101_LimHH”, “Par_LT101_LimH”, “Par_LT101_LimL” e “Par_LT101_LimLL” possuem o valor zero. Para que esses alarmes funcionem corretamente, atribua os valores 160, 140, 60 e 40 à essas variáveis, respectivamente.



TAREFA 6 – Ainda no bloco LT101 e com o recurso de monitoração ativado, verifique se a geração das condições de alarme estão coerentes (se julgar necessário causar variações no nível de água, volte à tabela “**Teste de I/O**” e manobre os atuadores pertinentes). Caso esteja tudo *ok*, siga em frente; caso contrário, tente encontrar o problema ou peça ajuda ao docente.

FINALIZANDO

No início desse roteiro, a função ESCLIN foi desenvolvida e depurada. Ao longo do projeto, ela será utilizada em três diferentes sub-rotinas, para converter valores de nível, temperatura e velocidade. Isso expõe a relevância deste tipo de recurso, que aumenta a medida em que um projeto cresce e se torna mais complexo. Além disso, ao cumprir as demais tarefas do roteiro, você deve ter estruturado seu projeto com todos os blocos a serem programados ao longo do curso, deixando prontos o bloco principal (executado ciclicamente pela CPU) e a sub-rotina LT101. Assim, se você assimilou bem os conceitos envolvidos e não houver dúvidas em relação ao que foi desenvolvido, realize as costumeiras tarefas de encerramento (caso contrário, antes de terminar, faça os esclarecimentos necessários com o docente).

TAREFA 1 – Se sua plataforma de desenvolvimento estiver *online* com seu CLP, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.



TAREFA 2 – Feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

4.5 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

Atribuir valores iniciais às variáveis do grupo “Parâmetros” e, a partir das orientações fornecidas no **Descritivo Funcional** do sistema de automação, desenvolver as sub-rotinas “GERAL” (Elementos gerais do sistema) e “TT101” (Temperatura da água).

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.

RETOMANDO

Durante a última aula prática, a sub-rotina LT101 (Nível de água no tanque) foi desenvolvida. Lembre que, para testá-la, provavelmente você fez operações de escrita na memória do CLP, atribuindo os valores 160, 140, 60 e 40 às variáveis que determinam os limites de disparo dos alarmes de nível muito alto, alto, baixo e muito baixo. No entanto, um sistema de automação geralmente possui diversos parâmetros, sendo que a atribuição manual de valores a cada um deles pode se tornar um grande transtorno, especialmente se essa necessidade for recorrente. Um bom método para remediar esse problema é usar a interrupção gerada no primeiro ciclo da CPU para executar uma sub-rotina e, em seu interior, atribuir valores iniciais a cada parâmetro. É importante que tais atribuições aconteçam apenas nesse momento inicial pois, durante a operação da planta, os usuários precisam ter a oportunidade de alterar esses parâmetros por meio do Sistema de Supervisão e Controle (se as atribuições acontecessem a cada ciclo do CLP, as alterações seriam sobrescritas).

TAREFA 1 – Com o objetivo de preparar o bloco *Startup* para que as variáveis do grupo “Parâmetros” recebam valores iniciais durante o primeiro ciclo do CLP, abra-o e crie 11 linhas com os títulos indicados a seguir.

Como?



- | | |
|-------------------------------|-------------------------------------|
| • Nível de água no tanque | • Temperatura da água |
| • Mixer | • Bomba de água |
| • Aquecedor | • Ventilador |
| • Válvula de dreno direto | • Válvula de dreno refrigerado |
| • Controlador On/Off de nível | • Controlador On/Off de temperatura |
| • Controlador PI de nível | |

TAREFA 2 – Faça uma análise do **Descritivo Funcional** (veja a Seção 5 da “Especificação Técnica do Sistema de Automação”, que pode ser acessada por meio do *link* ao lado). Perceba que, para a maioria dos elementos do sistema, há um quadro com os valores iniciais sugeridos para seus parâmetros. Assim, tomando como exemplo o “Nível de água no tanque”, volte ao bloco *Startup* e faça as quatro operações de atribuição necessárias na linha apropriada. Em seguida, faça o mesmo para os outros 10 elementos (ao final, confira se os 24 parâmetros do sistema foram configurados para receber valores iniciais).

Como?



Link



TAREFA 3 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

TAREFA 4 – Abra a tabela na qual as variáveis do grupo “Parâmetros” foram declaradas. Ative o recurso que permite monitorá-las e verifique se as inicializações realmente ocorreram.

Como?



Observação: caso todas variáveis possuam o valor zero, é provável que o processamento do seu CLP não tenha sido interrompido e reiniciado durante a última operação de *download* e, conseqüentemente, o bloco *Startup* não foi executado. Se necessário, como você está em um ambiente seguro, não há problema em resolver essa situação forçando sua CPU a fazer transições *run* → *stop* → *run* (em processos produtivos reais, há de se tomar bastante cuidado antes de parar o ciclo de uma CPU).

DESENVOLVIMENTO DA SUB-ROTINA “GERAL”

A sub-rotina GERAL (Elementos gerais do sistema), entre outras coisas, deverá gerenciar sinais como condição de emergência ativada, rearme de defeitos e *life bit*, que são pertinentes a dois ou mais atuadores ou a outros dispositivos do sistema de automação. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado aos elementos gerais do sistema (veja a Subseção 5.1 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:

Link



- Leia a descrição geral da sub-rotina;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (GERAL) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Informe-se sobre as particularidades da sub-rotina.

TAREFA 2 – Ao cumprir a tarefa anterior, entre outras coisas, você desenvolveu uma sub-rotina a partir de um **Descritivo Funcional** cujo formato se aproxima do usado em documentos profissionais de mesmo propósito. Como esta foi a primeira entre várias outras a serem programadas dessa forma, é provável que você esteja inseguro em relação ao que foi realizado. Se for o caso, compare seu diagrama com o gabarito fornecido por meio do *link* ao lado.



TAREFA 3 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

TAREFA 4 – Retorne ao módulo utilizado para desenvolver programas, abra a sub-rotina GERAL e ative o recurso que permite monitorar seu diagrama. Considerando as observações a seguir, verifique a coerência dos sinais programados.

Observação 1: após conferir se os sinais de emergência e de rearme estão funcionando de forma independente, verifique se não há uma falha na interação dos respectivos botões físicos: o que acontece se eles forem atuados ao mesmo tempo? Por segurança, a solicitação de emergência deve prevalecer sobre a de rearme;

Observação 2: por sua vez, note que o sinal “Pulso de 1 scan a cada 1.0 segundo” não pode ser visualmente verificado ao se monitorar o diagrama. Isso acontece porque a duração do pulso é muito curta, provavelmente menor do que o tempo de atualização da tela da sua estação de trabalho. No entanto, como este sinal é usado como condição para se fazer alterações na variável “Ana_GERAL_SinAtv”, note que os valores dessa última devem mudar a cada 1 segundo.

DESENVOLVIMENTO DA SUB-ROTINA “TT101”

A sub-rotina TT101 (Temperatura da água) tem o propósito de organizar todos os sinais relacionados à medição desta grandeza. Entre outras coisas, seu código interno deverá fazer uma chamada para a função ESCLIN, que converterá o valor fornecido pela entrada analógica para seu correspondente em graus Celsius. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado ao medidor TT101 (veja a Subseção 5.3 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:



- Leia a descrição geral do medidor;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (TT101) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Informe-se sobre as particularidades do medidor.

Observação: caso você esteja inseguro em relação ao diagrama que acabou de implementar, você pode compará-lo com o que foi previamente desenvolvido para a sub-rotina LT101 (com exceção das variáveis e constantes utilizadas, os diagramas devem ser idênticos).

TAREFA 2 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

TAREFA 3 – Volte à sub-rotina TT101 e ative o recurso que permite monitorar seu diagrama. Verifique se conversão dos valores de temperatura e a geração das condições de alarme estão coerentes.

Observações: para causar variações na temperatura da água, faça manobras com o aquecedor (como nenhum trecho de programação foi desenvolvido para esse atuador, você ainda pode usar a tabela “**Teste de I/O**” para escrever livremente na posição de memória associada à saída digital que o comanda). Porém, um intertravamento elétrico embutido no módulo que simula o processo produtivo impede o funcionamento do aquecedor caso ele não esteja submerso e assim talvez seja necessário encher o tanque (com manobras da bomba de água e das válvulas solenóides).

FINALIZANDO

Ao longo das últimas tarefas você deve ter configurado o bloco *Startup* de forma que todos os parâmetros do seu sistema de automação recebam valores iniciais no primeiro ciclo do CLP, além de ter implementado e depurado as sub-rotinas GERAL e TT101. Se não houver dúvidas em relação ao que foi desenvolvido, realize as costumeiras tarefas de encerramento (caso contrário, antes de terminar, anote suas dúvidas e procure esclarecê-las com o docente assim que possível).

TAREFA 1 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.

Como?



TAREFA 2 – Quando encerrar suas atividades, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

4.6 EXERCÍCIOS

QUESTÃO 1 – Suponha que você seja o responsável pela programação de um CLP que deverá realizar dois escalonamentos distintos um do outro, conforme descrito a seguir.

- Escalonar o valor em *IW64* (0 a 27648) para a variável real *MD0* (-20.0 a 80.0);
- Escalonar a variável real *MD4* (0 a 70.0) para a saída analógica *QW64* (0 a 27648).

Apresente uma solução para realizar esses dois escalonamentos utilizando uma mesma função (com código encapsulado), conforme os itens solicitado a seguir.

- a) Reproduza a Tabela 4.5, preenchendo-a com as variáveis globais que você usará em seu projeto.

Tabela 4.5: Variáveis globais a serem declaradas no projeto em questão.

Símbolo	Endereço	Tipo	Comentário

- b) Reproduza a Tabela 4.6, preenchendo-a com as variáveis a serem declaradas no cabeçalho da função que você usará para resolver o problema.

Tabela 4.6: Variáveis a serem declaradas no cabeçalho da função em questão.

Símbolo	In, Out ou Temp	Tipo	Comentário

- c) Apresente o Diagrama Ladder a ser utilizado como código interno da função.
- d) Apresente o Diagrama Ladder a ser utilizado no bloco MAIN.

QUESTÃO 2 – Suponha que n *displays* de 7 segmentos serão conectados às saídas digitais de um CLP, conforme mostrado na Figura 4.4. O funcionamento do primeiro *display* pode ser resumido por meio da tabela ao lado da figura.

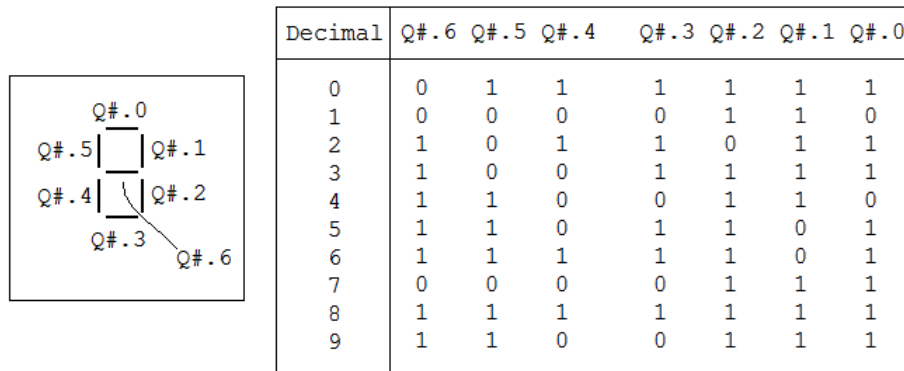


Figura 4.4: Display de 7 segmentos conectado às saídas digitais abrangidas pelo *byte* QB#.

Dessa forma, pode ser interessante criar uma função, com os parâmetros de entrada e saída mostrados na Figura 4.5, para tratar um ou mais *displays* de forma genérica.

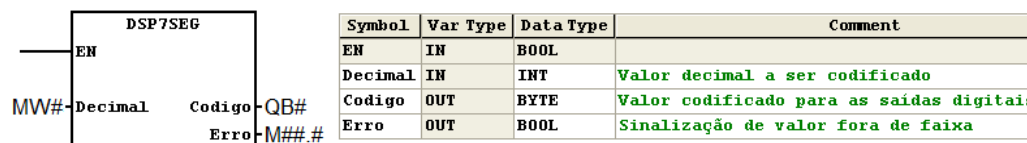


Figura 4.5: Aspecto externo da função a ser criada (lado esquerdo) e as variáveis declaradas em seu cabeçalho (lado direito).

Sabendo que esta função deve apenas codificar valores de 0 a 9, sinalizando valores fora de faixa por meio do *flag* “Erro”, apresente um código em Diagrama Ladder que poderia ser utilizado dentro do bloco “DSP7SEG”.

Dica: no seu Diagrama Ladder, use apenas as variáveis declaradas na tabela anterior: “Decimal”, “Codigo” e “Erro”. Se você sentir a necessidade de incluir outras variáveis, sintase livre, mas provavelmente estará indo por um caminho errado.

QUESTÃO 3 – Faça uma função (com código encapsulado, que poderá ser chamada várias vezes em um mesmo programa) para registrar quais foram o maior e o menor valor assumidos por uma variável do tipo *floating-point* (número real). Considere que os valores de máximo e mínimo fornecidos pela função podem ser incoerentes até que a primeira solicitação de “reset” seja realizada. Neste momento, tanto o valor máximo como o valor mínimo devem assumir o valor atual da variável que está sendo monitorada e, daí por diante, passam a registrar os verdadeiros máximos e mínimos. Sendo assim:

- Reproduza a Tabela 4.7, mostrando a declaração de variáveis a ser realizada no cabeçalho da função.
- O código interno da função, em Diagrama Ladder.

Tabela 4.7: Variáveis a serem declaradas no cabeçalho da função em questão.

Símbolo	In, Out, InOut ou Temp	Tipo de Dado	Comentário

QUESTÃO 4 – Explique porque uma função que implementa o algoritmo PID não deve ser chamada por uma sub-rotina cuja execução esteja vinculada ao ciclo do CLP, mas sim por uma sub-rotina vinculada a uma interrupção cíclica. Em sua explicação, não use termos como “o algoritmo PID precisa de cuidados especiais” (isso é muito genérico). Descreva precisamente o motivo e, se precisar, use ilustrações para isso.

4.7 REFERÊNCIAS

REHG, J.A.; SARTORI, G.J. **Programmable Logic Controllers**. 2. ed.: Pearson Prentice Hall, 2009. 600 p. ISBN 9780135048818.

ENCERRAMENTO DA PRIMEIRA ETAPA

As primeiras avaliações do curso devem ocorrer após a finalização do tópico anterior. Dessa forma, as respostas dos exercícios desenvolvidos para contribuir com sua preparação para a avaliação teórica foram reunidos na próxima seção. Além disso, na seção seguinte, há um pequeno roteiro a ser cumprido em laboratório, com os preparativos para que o trabalho prático possa ser devidamente avaliado.

RESPOSTAS DOS EXERCÍCIOS

Respostas dos exercícios	Link
Capítulo 1	▼
Capítulo 2	▼
Capítulo 3	▼
Capítulo 4	▼

ROTEIRO PARA APRESENTAÇÃO DO TRABALHO PRÁTICO

OBJETIVO

Realizar os preparativos para que as funcionalidades do trabalho prático previstas até o momento possam ser devidamente avaliadas.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.

TAREFAS

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.

TAREFA 2 – Prepare seu projeto para ser avaliado, conferindo e testando as funcionalidades previstas até o momento. Consulte o docente sobre o limite de tempo para esta tarefa e, até o horário estipulado, sinta-se livre para fazer os últimos ajustes no projeto e quantos procedimentos de *download* forem necessários para testá-lo.

TAREFA 3 – Se sua plataforma de desenvolvimento estiver *online* com seu CLP, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto mas, desta vez, atribuindo os nomes dos integrantes da dupla ao arquivo compactado. Posteriormente, consulte o docente sobre como proceder com este arquivo, pois isto será essencial caso uma revisão da avaliação venha a ser solicitada.

Como?



TAREFA 4 – Antes de deixar o laboratório, ative novamente o modo *online* de sua plataforma de desenvolvimento e assegure-se de que o projeto em seu PC seja exatamente o mesmo que está sendo executado pelo CLP de sua bancada. Ao sair, deixe o projeto aberto e o local limpo e organizado.

Como?



Capítulo 5

PROGRAMAÇÃO EM DIAGRAMA LADDER: CONTADORES E TEMPORIZADORES

5.1 INTRODUÇÃO

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação).

5.2 CONTADORES

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Controladores Lógicos Programáveis	8	8.1

5.3 TEMPORIZADORES

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
Controladores Lógicos Programáveis	8	8.2

5.4 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Desenvolver a sub-rotina que faz o gerenciamento do mixer (MX101), de acordo com as orientações fornecidas no **Descritivo Funcional** do sistema de automação.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (STEP 7);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.



DESENVOLVIMENTO DA SUB-ROTINA “MX101”

A sub-rotina MX101 (Mixer), entre outras coisas, deverá permitir que este atuador seja acionado nos sentidos de rotação normal e reverso, tanto por decisões humanas como de forma automática. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.

TAREFA 2 – Encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado ao atuador MX101 (veja a Subseção 5.4 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:



- Leia a descrição geral da sub-rotina;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (MX101) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Trate as particularidades do atuador, ou seja, implemente a lógica de funcionamento automático conforme o detalhamento fornecido.

TAREFA 3 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

TAREFA 4 – Para iniciar o teste da sub-rotina MX101, ative o recurso que permite monitorar seu diagrama e verifique a coerência do estado “Mixer bloqueado”. Para isso, ative e desative os alarmes de emergência e de nível baixo (veja as observações a seguir). Finalize essa tarefa deixando os dois alarmes em nível lógico 0 e, conseqüentemente, o mixer desbloqueado.

Observações: use os botões físicos de emergência e de rearme para alterar o nível lógico do primeiro alarme. Para o segundo, cause variações no nível de água do tanque usando a bomba e as válvulas solenóides (como nenhum trecho de programação foi desenvolvido para esses atuadores, você ainda pode escrever livremente nas posições de memória associadas às saídas elétricas que os comandam, usando a tabela “Teste de I/O”).

TAREFA 5 – Teste os comandos para acionamento do mixer via interface remota (botões virtuais que estarão presentes no SSC). Para começar, configure o modo de operação do atuador para que este passe a aceitar esses comandos. Em seguida, simule a ação dos botões virtuais levando as variáveis “Cmd_MX101_LigNor”, “Cmd_MX101_Deslig” e “Cmd_MX101_LigRev” a nível lógico 1, sempre que desejar ligar o mixer no sentido normal, desligá-lo ou ligá-lo no sentido reverso.

Como?



Observação: devido à lógica de “limpeza de comandos”, configurada durante as atividades do Capítulo 4, nenhuma dessas três variáveis permanecerá em nível lógico 1 por mais de 1 segundo. Isso acontece por uma questão de segurança, a ser discutida no próximo capítulo.

TAREFA 6 – Supondo que as condições que levam ao bloqueio do atuador ainda estejam inativas, para testar sua lógica automática, proceda da seguinte maneira: configure seu modo de operação para que este entre em regime de trabalho automático; satisfaça a condição de ativação da lógica, simulando a necessidade de agitação da água devido a uma suposta ativação do aquecedor; verifique se o comportamento de partidas e paradas especificado no descritivo funcional está correto.

Como?



Observações: note que, nesse momento, o nível lógico da variável “Stt_AQ101_Ligado” pode ser alterado livremente. Isso acontece pois nenhuma instrução está escrevendo nesta posição de memória, uma vez que a sub-rotina AQ101 ainda não foi programada. Além disso, como essa variável não pertence ao grupo “Comandos”, seu nível lógico não sofrerá restaurações periódicas, ou seja, para interromper o teste da lógica automática em questão, deve-se levar a variável a 0 por meio de uma operação análoga à que foi feita para ativá-la.

FINALIZANDO

Ao longo das últimas tarefas você deve ter implementado e depurado a sub-rotina MX101. Se os conceitos envolvidos foram bem assimilados e não houver dúvidas em relação ao que foi desenvolvido, realize as tarefas de encerramento a seguir (caso contrário, antes de terminar, faça os esclarecimentos necessários com o docente).

TAREFA 1 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.

Como?



TAREFA 2 – Feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

5.5 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

Desenvolver a sub-rotina que faz o gerenciamento da bomba de água (BA101), de acordo com as orientações fornecidas no **Descritivo Funcional** do sistema de automação.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (STEP 7);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.



DESENVOLVIMENTO DA SUB-ROTINA “BA101”

A sub-rotina BA101 (Bomba de água), entre outras coisas, deverá permitir que este atuador seja acionado por decisões humanas ou por diferentes métodos automáticos. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.

TAREFA 2 – Encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado ao atuador BA101 (veja a Subseção 5.5 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:



- Leia a descrição geral da sub-rotina;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (BA101) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Informe-se sobre as particularidades do atuador.

TAREFA 3 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

TAREFA 4 – Para iniciar o teste da sub-rotina BA101, ative o recurso que permite monitorar seu diagrama e verifique a coerência da primeira condição que pode levar o estado “Bomba bloqueada” à nível lógico alto (a segunda condição será verificada durante a próxima tarefa). Finalize deixando a bomba desbloqueada.



TAREFA 5 – Teste os trechos que foram preparados para permitir a operação da bomba via interface remota (botões virtuais e campo para escolha da velocidade, que estarão presentes no SSC). Para começar, determine a velocidade com a qual ela deve operar nesta situação. Em seguida, ajuste o modo de operação e simule a ação do botão “liga”, deixando o atuador ligado até ter certeza de que a condição de nível muito alto bloqueará o funcionamento (caso perceba que algo está errado, aperte a botoeira de emergência). Quando o teste terminar, desative o atuador simulando a ação do botão “desliga”.



Observação: antes de passar para a próxima tarefa, é necessário esvaziar o tanque. Para isso, como nenhum trecho de programação foi ainda desenvolvido para as válvulas solenóides, use a tabela “**Teste de I/O**” para escrever nas posições de memória associadas às saídas elétricas que as comandam. Termine deixando-as fechadas.

TAREFA 6 – Aproveitando que os modos de operação “controlador On/Off” e “sequenciador de batelada” compartilham a mesma referência de velocidade usada no modo “interface remota”, faça o teste dos trechos de programação referentes aos dois primeiros. Para cada um deles, proceda da seguinte maneira: configure o modo de operação apropriado e satisfaça a condição de ativação da lógica, simulando a solicitação de bombeamento a ser programada em outras sub-rotinas.



TAREFA 7 – Por sua vez, o modo de operação “controlador PI” faz com que a velocidade da bomba passe a ser igual ao sinal de controle calculado pelo referido algoritmo. Como este último ainda não foi implementado, teste a bomba de água da seguinte forma: configure o modo de operação apropriado e simule um valor para o sinal de controle.



Observação: uma vez que o modo “controlador PI” esteja selecionado, caso não haja condição de bloqueio, a bomba estará sempre ligada, pois se não houver necessidade de bombeamento, espera-se que o controlador calcule um sinal de controle muito próximo de zero.

FINALIZANDO

Ao longo das últimas tarefas você deve ter implementado e depurado a sub-rotina BA101. Se não houver dúvidas em relação ao que foi desenvolvido, realize as costumeiras tarefas de encerramento (caso contrário, antes de terminar, anote suas dúvidas e procure esclarecê-las com o docente assim que possível).

TAREFA 1 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.



TAREFA 2 – Quando encerrar suas atividades, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

5.6 EXERCÍCIOS

QUESTÃO 1 – Utilizando apenas um temporizador, elabore um programa em Ladder para acionar uma lâmpada sinalizadora que deverá ficar 2 segundos acesa e 2 segundos apagada (ciclicamente).

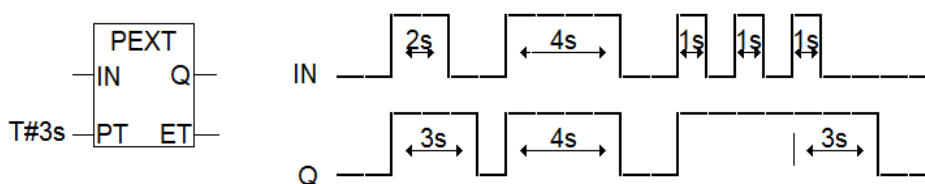
QUESTÃO 2 – Um misturador de tintas utiliza dois motores (M1 e M2) para homogeneizar a mistura, os quais devem ser ligados alternadamente em um intervalo de 30 segundos. Para iniciar o funcionamento do sistema, existe uma botoeira não-retentiva “L” e, para interromper o funcionamento, existe uma botoeira também não-retentiva “D”. Projete a lógica em Ladder para controlar esse sistema.

QUESTÃO 3 – Utilizando um ou mais contadores, elabore um programa em Ladder para acionar uma Lâmpada sinalizadora sempre que o número de pulsos recebidos em uma entrada digital for múltiplo de 5. Assim, no recebimento do quinto pulso a lâmpada acende, sendo desligada no sexto; acende no décimo, apaga no décimo primeiro e assim por diante.

QUESTÃO 4 – Elabore um programa em Ladder para acionar a lâmpada L1 quando a quantidade de pulsos dada na botoeira não-retentiva B1 for igual a 3 em um tempo menor ou igual a 10 segundos. Se o tempo for maior que 10 segundos, deve-se zerar o contador automaticamente. Uma vez ligada, a lâmpada só será desligada por meio da botoeira não-retentiva B0.

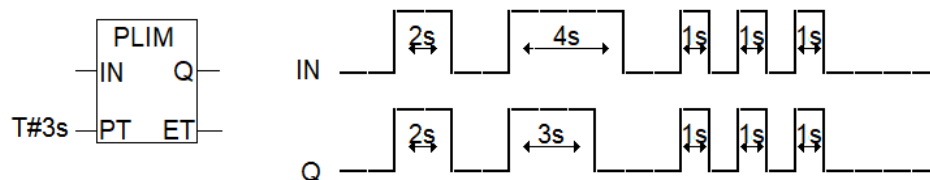
QUESTÃO 5 – Além dos temporizadores TON e TOFF, presentes em quase todos os modelos de CLP, outros tipos de temporizador também existem. Entre eles, estão:

- Temporizador extensor de pulsos (PEXT) – Ao receber nível lógico 1 em sua entrada, transfere este sinal imediatamente para sua saída, porém no mínimo pelo tempo pré-estabelecido (PT).

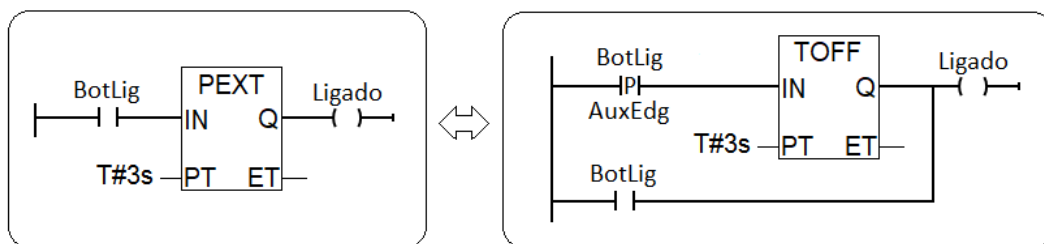


Observação: este temporizador possui algumas similaridades com o TOFF, mas, na verdade, as diferenças existem e são relevantes: como se sabe, o TOFF *sempre* estende o sinal de sua entrada durante o tempo pré-estabelecido. No entanto, o PEXT apenas garante que o sinal de saída não irá durar menos do que o tempo pré-estabelecido, estendendo a entrada apenas quando for necessário.

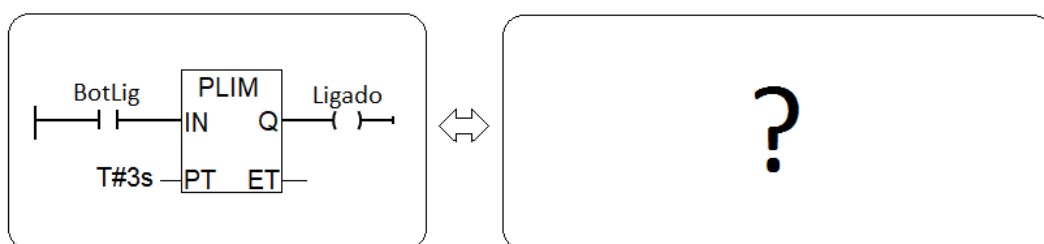
- Temporizador limitador de pulsos (PLIM) – Ao receber nível lógico 1 em sua entrada, transfere este sinal imediatamente para sua saída, porém no máximo pelo tempo pré-estabelecido (PT).



Apesar de serem úteis em algumas situações, nem sempre estes dois tipos temporizador (PEXT e PLIM) estão disponíveis para o programador (no S7-1200, por exemplo, eles não estão). Possivelmente, tal constatação se deve aos seguintes fatos: é muito simples obter o comportamento do PEXT a partir do TOFF, assim como é muito simples obter o comportamento do PLIM a partir do TON. Por exemplo, para reproduzir com o comportamento de um PEXT a partir de um TOFF basta fazer:



Sendo assim, mostre como reproduzir o comportamento de um PLIM a partir de um TON. Considere que a variável auxiliar “AuxTmr” está disponível para isso.



QUESTÃO 6 – Supondo que todas as variáveis envolvidas no Diagrama Ladder que aparece na Figura 5.1 começam em nível lógico 0, desenhe as formas de onda das variáveis W, X, Y e Z, considerando os estados em que elas se encontram ao final de cada ciclo do CLP.

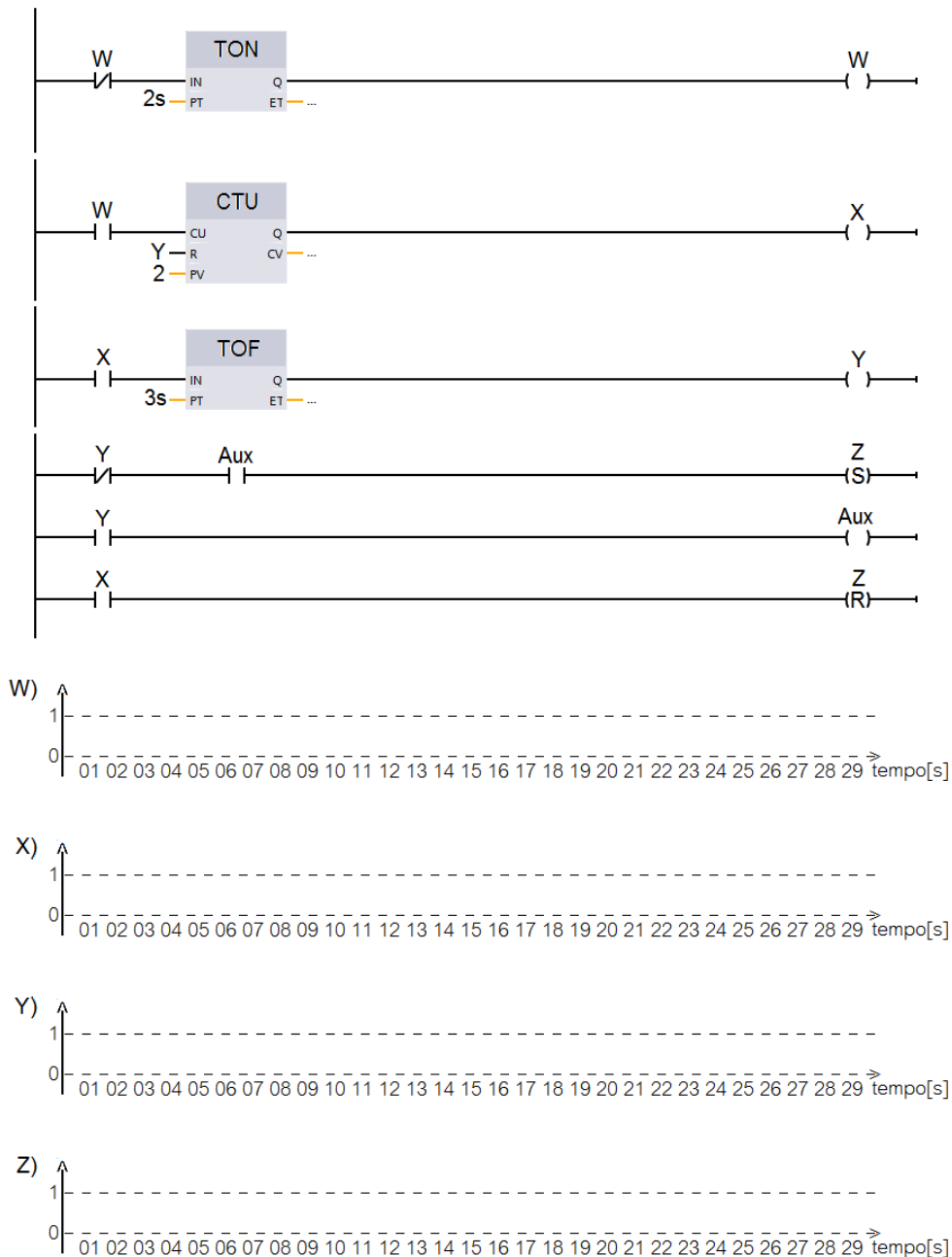


Figura 5.1: Diagrama Ladder a ser considerado e suporte para desenho das formas de onda solicitadas.

5.7 REFERÊNCIAS

FRANCHI, C.M.; CAMARGO, V.L.A. de. **Controladores Lógicos Programáveis:**
Sistemas Discretos. Érica, 2008. 352 p. ISBN 9788536501994.

Capítulo 6

DESENVOLVIMENTO DE SSCs: TAGS, TELAS E ADMINISTRAÇÃO DE USUÁRIOS

6.1 INTRODUÇÃO

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação).

6.2 GERENCIAMENTO DE TAGS

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção	Subseção
SIMATIC WinCC V7.3: Working with WinCC	2	2.1	—
		2.2	—
		2.4	2.4.1.1 a 2.4.1.4 2.4.1.8 a 2.4.1.9
		2.5	2.5.1.1 a 2.5.1.4
			2.5.2.1 a 2.5.2.4
			2.5.2.6
			2.5.3.1 a 2.5.3.5

6.3 DESENVOLVIMENTO DE TELAS

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção	Subseção
SIMATIC WinCC V7.3: Working with WinCC	4	4.1	—
		4.3	4.3.1 a 4.3.4
		4.5	4.5.1 a 4.5.3
		4.8	4.8.1 a 4.8.6
		4.7	4.7.1 a 4.7.8
		4.6	4.6.1 a 4.6.4

6.4 ADMINISTRAÇÃO DE USUÁRIOS

Enquanto não temos um texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção	Subseção
SIMATIC WinCC V7.3: Working with WinCC	14	14.1	—
		14.4	14.4.1 a 14.4.2
		14.5	14.5.1 a 14.5.4
		14.8	—
		14.9	—

6.5 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Praticar tarefas fundamentais disponíveis em uma plataforma de desenvolvimento de SSCs, como: criação de projeto, configuração do modo de exibição, cadastro de *tags*, desenvolvimento de telas (incluindo animações e botões virtuais) e gerenciamento de usuários, senhas e permissões.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com as plataformas STEP 7 e WinCC instaladas);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.

CONSIDERAÇÕES E TAREFAS PRELIMINARES

Como participante deste curso, este será seu primeiro contato com a plataforma de desenvolvimento de SSCs a ser utilizada ao longo das demais semanas. As tarefas solicitadas a seguir são fundamentais e, algumas delas, serão necessárias em diversas outras oportunidades. Por isso, realize-as com bastante atenção e lembre-se de retornar a este ponto em caso de dúvidas.

TAREFA 1 – Como é provável que sua bancada de trabalho seja compartilhada com outros estudantes, antes de mais nada, para garantir que o CLP esteja corretamente configurado, utilize a plataforma de programação deste dispositivo para compilar e fazer o *download* da versão mais recente do seu projeto. Além disso, certifique-se de que o módulo didático que faz a simulação do processo produtivo em sua bancada esteja ligado.

TAREFA 2 – Abra a plataforma de desenvolvimento de SSCs adotada neste curso (provavelmente o projeto do último usuário desta plataforma será aberto também; se for este o caso, feche-o).



TAREFA 3 – Crie um projeto e salve-o na pasta que você usará ao longo do semestre.



TAREFA 4 – Antes de começar o desenvolvimento do sistema, utilize o *link* ao lado para fazer o *download* de um pacote de figuras que serão úteis mais a frente, para representar elementos como a bomba de água, o mixer, as válvulas de dreno, entre outros e facilitar a construção de um sistema similar aos que são usados nas indústrias. Em seguida, descompacte este pacote dentro da estrutura de diretórios que armazena os arquivos do seu projeto.



TAREFA 5 – Configure os recursos que serão utilizados pelo sistema a ser desenvolvido, como execução de *scripts* em segundo plano e armazenamento de dados para formação de históricos, entre outros. Além disso, configure as características do ambiente de operação, como exibição em modo “tela cheia”, por exemplo.



EXPERIMENTO 1

Uma tarefa fundamental relacionada ao desenvolvimento de SSCs é o cadastro dos *tags*, que podem ser internos (usados para armazenar dados relevantes apenas para o SSC) ou de processo (usados para troca de dados com CLPs e outros dispositivos). Ambos os tipos podem ser criados, alterados e excluídos em qualquer etapa do projeto. No entanto, as boas práticas de desenvolvimento propõem que, ao se iniciar a construção de um SSC, os dados a serem trocados com o(s) CLP(s) estejam bem definidos e documentados. O projeto que você vem desenvolvendo ao longo deste curso cumpre esta recomendação e, sendo assim, cumpra as tarefas a seguir para realizar o cadastro dos *tags* a serem utilizados nos próximos experimentos e ao longo das próximas semanas.

TAREFA 1 – Encontre, na sua plataforma de desenvolvimento, o módulo de gerenciamento de *tags* e, neste primeiro momento, crie todos os *tags* internos indicados na Tabela 6.1.



Tabela 6.1: *Tags* Internos, a serem utilizados ao longo das próximas semanas.

Nome	Tipo de dado	Grupo
Aux_DVC01_Inativo	Binary Tag	Auxiliares
Aux_DVC01_LstAtv	Floating-point number 32-bit	Auxiliares
Aux_DVC01_TmeOut	Floating-point number 32-bit	Auxiliares
Aux_DVC02_Inativo	Binary Tag	Auxiliares
Aux_DVC02_LstAtv	Floating-point number 32-bit	Auxiliares
Aux_DVC02_TmeOut	Floating-point number 32-bit	Auxiliares
Aux_TYP01_Aberto	Text tag 8-bit character set	Auxiliares
Aux_TYP01_Abre	Text tag 8-bit character set	Auxiliares
Aux_TYP01_Fecha	Text tag 8-bit character set	Auxiliares
Aux_TYP01_Ilk01Tag	Text tag 8-bit character set	Auxiliares
Aux_TYP01_Ilk01Txt	Text tag 8-bit character set	Auxiliares
Aux_TYP01_Interlock	Text tag 8-bit character set	Auxiliares
Aux_TYP01_ModOpr	Text tag 8-bit character set	Auxiliares

TAREFA 2 – Ainda no módulo de gerenciamento de *tags*, adicione o *driver* de comunicação apropriado para o CLP da sua bancada de trabalho e crie uma conexão específica para possibilitar a troca de dados com ele.



TAREFA 3 – Dentro da conexão configurada na tarefa anterior, crie cinco grupos de *tags*, nomeando-os da seguinte maneira: Comandos, Estados, Alarmes, Analógicas e Parâmetros.



TAREFA 4 – De acordo com as informações contidas no **Mapa de Memória do CLP** (veja a Seção 4 do documento que pode ser acessado por meio do *link* ao lado), faça o cadastro dos *tags* pertencentes aos cinco grupos criados na tarefa anterior. Para isso, utilize as planilhas de apoio fornecidas na Tabela 6.2.






Como?



Link



Tabela 6.2: Grupos cujos *tags* estão previstos no **Mapa de Memória do CLP** e, entre outras coisas, permitem que o SSC realize operações de leitura e escrita em sua memória.

Grupo	Quantidade de <i>tags</i>	Link
Comandos	19	
Estados	18	
Alarmes	12	
Analógicas	8	
Parâmetros	24	

EXPERIMENTO 2

Sem dúvida, um dos principais objetivos de um SSC é a apresentação inteligível de informações para aos operadores da planta. No caso em que essas informações são os estados de um atuador (ligado, bloqueado, defeito etc.), é comum que suas indicações sejam feitas por meio de um padrão cores, ou seja, o objeto que representa o atuador troca de cor para indicar em qual estado este se encontra. Esse tipo de animação pode ser feita de diferentes maneiras, sendo que as técnicas mais comuns são as baseadas na sobreposição de objetos, no uso de objetos capazes de embutir várias figuras e no uso de objetos com *pixels* transparentes. As tarefas indicadas a seguir exploram essas três técnicas, além da criação de botões virtuais que permitem a verificação das animações implementadas.

TAREFA 1 – Antes de se alocar objetos em uma determinada tela do SSC, é preciso que esta última tenha sido previamente criada no sistema. Dessa forma, encontre o módulo apropriado em sua plataforma de desenvolvimento e crie uma tela para dar suporte aos seus primeiros experimentos, denominado-a “Rascunho”. Em seguida, configure-a para ser a tela inicial do seu SSC e termine esta tarefa deixando-a aberta.

Como?



TAREFA 2 – Utilizando a técnica de sobreposição de objetos, faça uma representação preliminar da bomba de água (BA101), associando seus estados (ligada e bloqueada) a uma determinada cor (verde e amarelo). Além disso, quando nenhum desses estados está ativo, considera-se que a bomba está desligada, sendo que sua representação deve receber a cor azul.



TAREFA 3 – Para comparação, faça outra representação da bomba de água (BA101), com a mesma associação entre estados e cores feita na tarefa anterior, mas agora utilizando a técnica de animação baseada em um único objeto, capaz de embutir várias figuras e apresentá-las conforme sua configuração.



TAREFA 4 – Explorando a última técnica de animação prevista neste experimento, faça uma terceira representação da bomba de água (BA101), sobrepondo um objeto com *pixels* transparentes a um objeto que efetivamente mudará de cor, de forma que o padrão de indicação de estados adotado nas duas tarefas anteriores seja preservado.



TAREFA 5 – Para viabilizar o teste dos objetos que representam a bomba de água (BA101), acrescente quatro botões virtuais à sua tela, sendo eles: “Emergência”, “Rearme”, “Liga” e “Desliga”. Em ambiente de execução, quando o usuário clicar nesses botões, deve-se levar as respectivas posições de memória do CLP a nível lógico alto (isso é feito por meio dos *tags* cadastrados no SSC).



TAREFA 6 – Salve sua tela, ative o ambiente de execução do seu SSC e verifique a coerência das animações: quando a emergência estiver ativa, a bomba ficará bloqueada (cor amarela); caso contrário, poderá ser ligada (cor verde) e desligada (cor azul) normalmente. *Obs:* se algo der errado, lembre-se de que o comportamento esperado *também* depende da programação previamente realizada no CLP.



EXPERIMENTO 3

Se você reparar bem, as implementações realizadas no experimento anterior estão relacionadas com operações binárias na memória do CLP: para animar os objetos que representam a bomba de água, ocorre a leitura cíclica dos estados “bloqueada” e “ligada”; ao passo que, para ativar a condição de emergência, rearmá-la, ligar e desligar a bomba, são feitas operações de escrita, por meio dos respectivos botões virtuais. De forma similar, é possível utilizar objetos no SSC para apresentar e alterar valores decimais na memória do CLP. Para ilustrar isso, as tarefas a seguir mostram como a indicação do nível de água no tanque pode ser feita e como os parâmetros relacionados à bomba de água podem ser alterados.

TAREFA 1 – Com o objetivo de indicar numericamente o nível de água no tanque, insira um campo para apresentação e entrada de valores na tela “Rascunho”. Associe o *tag* “Ana_LT101_NivEng” a este objeto e faça ajustes em sua aparência, como a quantidade de casas decimais a serem exibidas, cor de fundo, cor e tamanho da fonte. Como não faz sentido que o usuário do sistema escreva um valor na variável em questão, bloqueie essa opção.



TAREFA 2 – Utilize um “bargraph” para apresentar graficamente o nível de água no tanque. Além de associá-lo ao *tag* apropriado, configure-o para mostrar valores entre 0 e 200 mm.



TAREFA 3 – Utilize um campo para apresentação e entrada de valores para permitir que o usuário do sistema altere a velocidade da bomba. Associe o *tag* “Par_BA101_RefVel” a este objeto e ajuste sua aparência (quantidade de casas decimais, cores, tamanho da fonte etc).



TAREFA 4 – Com o objetivo de permitir que o usuário selecione o modo de operação da bomba de água, utilize uma lista de textos para corresponder códigos numéricos (12, 21, 22 e 29) às suas descrições (INTERFACE REMOTA, CONTROLE ON/OFF, CONTROLE PI e SEQUENCIADOR). Após fazer a associação do *tag* apropriado ao objeto, configure sua aparência.



TAREFA 5 – Salve sua tela, ative o ambiente de execução do seu SSC e verifique a coerência das operações de leitura e escrita implementadas nas quatro tarefas anteriores: nível de água no tanque (valor numérico e indicação gráfica), velocidade e modo de operação da bomba (selecione “INTERFACE REMOTA” para que os botões “Liga” e “Desliga” sejam considerados pelo CLP).

EXPERIMENTO 4

Nos últimos dois experimentos, entre outras coisas, foram implementados recursos de interface para permitir que os usuários do sistema apliquem suas decisões ao processo (botões virtuais e campos para alteração de parâmetros). Como se pode imaginar, em um processo real, é interessante restringir tais ações às pessoas capacitadas para realizá-las, sendo que isso pode ser feito considerando-se diferentes níveis de expertise. Por exemplo, autoriza-se um usuário a manobrar atuadores, mas não a alterar os ganhos de um controlador PID. Basicamente, este gerenciamento é realizado em três etapas (previstas nas tarefas a seguir): cadastro de usuários, senhas e permissões; inclusão de objetos em tela, para que os usuários possam fazer *login* e *logout*; proteção dos objetos de interesse, impedindo que pessoas não autorizadas possam utilizá-los.

TAREFA 1 – Encontre, na sua plataforma de desenvolvimento, o módulo de gerenciamento de usuários, senhas e permissões. Inclua um usuário com seu nome (dois usuários, se o trabalho prático estiver sendo feito em dupla) no grupo de administradores do sistema (as senhas podem ser escolhidas livremente). Em seguida, crie o grupo “Operadores” e adicione a esse último os usuários “Fulana” (senha 123456) e “Beltrano” (senha 123456). Usando a Tabela 6.3 como referência, atribua as devidas permissões aos usuários recém-criados. Ao terminar, feche o módulo em questão (não há necessidade de salvar as configurações).



Tabela 6.3: Usuários recém-criados e suas respectivas permissões.

	Você	Seu/Sua colega	Fulana	Beltrano
User Administration	×	×		
Value input	×	×	×	
Process controlling	×	×	×	×
Project Manager	×	×		

TAREFA 2 – Volte à tela “Rascunho” e acrescente dois objetos: um botão virtual para chamar a janela (já existente no sistema) que permite aos usuários fazerem *login* e *logout*; um campo para apresentação e entrada de valores, para indicar qual usuário está atualmente utilizando o sistema. Para finalizar essa tarefa, configure a aparência de ambos os objetos, deixando-os parecidos com seus pares.



TAREFA 3 – Ainda na tela “Rascunho”, proteja os objetos de interesse, de forma que apenas usuários que possuam as autorizações adequadas (veja a Tabela 6.4) sejam capazes de interferir no processo. Note que os objetos que apenas apresentam informações não entram na lista de objetos protegidos.



Tabela 6.4: Ações previstas na tela “Rascunho” e seus respectivos níveis de autorização.

Ação	Autorização requerida
Ativar condição de emergência	nenhuma
Rearmar o sistema	Process controlling
Ligar a bomba de água	Process controlling
Desligar a bomba de água	Process controlling
Selecionar o modo de operação	Process controlling
Alterar velocidade da bomba de água	Value input

TAREFA 4 – Salve sua tela, ative o ambiente de execução do seu SSC e, antes de fazer *login*, verifique se os objetos que foram protegidos estão realmente inacessíveis. Em seguida, teste o acesso de cada usuário cadastrado aos objetos em questão (se as configurações foram realizadas corretamente, a única restrição imposta até o momento é que “Beltrano” não pode alterar a velocidade da bomba).

FINALIZANDO

Ao longo dos últimos experimentos, utilizando sua plataforma de desenvolvimento de SSCs, você criou um projeto, configurou uma conexão com o CLP de sua bancada e cadastrou todos os *tags* relacionados ao trabalho prático que vem sendo desenvolvido. Além disso, uma tela de rascunho foi utilizada para dar suporte a diversos objetos implementados com o objetivo permitir uma primeira interação com o processo produtivo que vem sendo automatizado nas últimas semanas. Por fim, alguns desses objetos foram protegidos, de forma que apenas os usuários autorizados possam fazer alterações no sistema.

Dessa forma, se você assimilou bem esses conceitos e não houver dúvidas em relação ao que foi desenvolvido, realize as tarefas de encerramento a seguir (caso contrário, antes de terminar, faça os esclarecimentos necessários com o docente).

TAREFA 1 – Com o objetivo de facilitar estudos posteriores, volte à tela “Rascunho” e utilize objetos do tipo adequado para identificar os principais elementos implementados durante os experimentos anteriores.

Como?



TAREFA 2 – Se o ambiente “*runtime*” do seu SSC estiver em execução, faça a desativação. Feche os módulos de configuração (gerenciamento de *tags*, edição de telas etc) que porventura estejam abertos e, em seguida, faça *backup* do seu projeto.

Como?



TAREFA 3 – Feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

6.6 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

Desenvolver as sub-rotinas que fazem o gerenciamento do aquecedor (AQ101) e do ventilador (VT101), de acordo com as orientações fornecidas no **Descritivo Funcional** do sistema.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.

DESENVOLVIMENTO DA SUB-ROTINA “AQ101”

A sub-rotina AQ101 (Aquecedor), entre outras coisas, deverá permitir que este atuador seja acionado por decisões humanas (tanto por interface local como por interface remota) ou por métodos automáticos. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.

TAREFA 2 – Encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado ao atuador AQ101 (veja a Subseção 5.6 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:



- Leia a descrição geral da sub-rotina;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (AQ101) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Trate as particularidades do atuador, associando as devidas variáveis ao bloco da função “Horímetro”, conforme detalhamento fornecido.

TAREFA 3 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

DESENVOLVIMENTO DA SUB-ROTINA “VT101”

A sub-rotina VT101 (Ventilador), assim como no caso do aquecedor, deverá permitir que o atuador em questão seja acionado por decisões humanas (tanto por interface local como por interface remota) ou de forma automática. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Considerando que a versão mais recente do seu projeto está aberta, encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado ao atuador VT101 (veja a Subseção 5.7 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:



- Leia a descrição geral da sub-rotina;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (VT101) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Trate as particularidades do atuador, associando as devidas variáveis ao bloco da função “Horímetro”, conforme detalhamento fornecido.

TAREFA 2 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

FINALIZANDO

Ao cumprir as tarefas anteriores você deve ter implementado as sub-rotinas AQ101 e VT101. Se não houver dúvidas em relação ao que foi desenvolvido, realize as costumeiras tarefas de encerramento (caso contrário, antes de terminar, anote suas dúvidas e procure esclarecê-las com o docente assim que possível).

TAREFA 1 – Se sua plataforma de desenvolvimento estiver *online* com seu CLP, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.



TAREFA 2 – Quando encerrar suas atividades, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

6.7 EXERCÍCIOS

QUESTÃO 1 – Anteriormente à popularização do processamento digital, as *Interfaces de Operação* dos Sistemas de Automação baseavam-se em elementos como botoeiras, chaves seletoras, lâmpadas e indicadores mecânicos de grandezas como pressão, corrente etc. Atualmente, em diversas situações, esses elementos podem ser substituídos por suas representações virtuais em um Sistema de Supervisão e Controle. Sendo assim, cite e explique três vantagens e duas desvantagens da referida modernização dos Sistemas de Automação.

QUESTÃO 2 – Além do *WinCC* (desenvolvido pela *Siemens* e utilizado em nossas aulas práticas), existem diversas outras plataformas para configuração e desenvolvimento de Sistemas de Supervisão e Controle disponíveis no mercado. Sendo assim, pede-se:

- a) Cite, pelo menos, mais *três* plataformas de desenvolvimento disponíveis no mercado e seus respectivos desenvolvedores.
- b) Cite, pelo menos, *três* fatores *determinantes* para escolha da plataforma a ser adquirida durante a concepção de um Sistema de Automação.

QUESTÃO 3 – Leia a seguinte afirmação sobre Sistemas de Supervisão e Controle: em geral, possuem dois ambientes (dois modos básicos de utilização). Comente a afirmação dizendo quais são esses dois ambientes, caracterizando e fornecendo informações sobre cada um.

QUESTÃO 4 – No contexto dos Sistemas de Supervisão e Controle, explique qual é a diferença entre *tags* internos e *tags* de comunicação? Cite *dois* exemplos de utilização de cada tipo.

QUESTÃO 5 – Quais os principais tipos e para que servem os objetos que geralmente compõem uma tela sinóptica de um sistema SCADA? Cite e explique a função de pelo menos 4 tipos de objeto.

6.8 REFERÊNCIAS

AG, Siemens (ed.). **SIMATIC WinCC V7.3: Working with WinCC**. System Manual. Germany, Nürnberg, 2014. Disponível em: <https://drive.google.com/open?id=1s16MduseMvqDQ78r89iQ0Qo--dew4SI0>.

Capítulo 7

DESENVOLVIMENTO DE SSCs: SCRIPTS, ALARMES E CURVAS DE TENDÊNCIA

7.1 INTRODUÇÃO

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação).

7.2 USO DE SCRIPTS

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
WinCC V7.0 SP3: ANSI-C for Creating Functions and Actions	1	—
	12	—
	13	—

7.3 REGISTRO E APRESENTAÇÃO DE ALARMES

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
WinCC V7.0 SP3: Setting up a Message System	3	3.1 e 3.2
	4	4.1 e 4.2
		4.5.1 à 4.5.6
		4.7
	1	—
WinCC V7.0 SP3: Display of Messages during Runtime	2	2.1 e 2.5
	3	3.1

7.4 CURVAS DE TENDÊNCIA

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Livro	Capítulo	Seção
WinCC V7.0 SP3: Archiving Process Values	2	—
	3	3.1
		3.3.1 e 3.3.2
		3.4.1 à 3.4.6
		3.5 e 3.6
	4	4.1 à 4.3
		4.4.1 à 4.4.4
		4.5 e 4.6
	2	2.1 e 2.2
WinCC V7.0 SP3: Process Value Output as a Function of Another Tag	3	3.1 e 3.6
	4	4.1

7.5 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Aperfeiçoar a interface de operação desenvolvida durante a última aula prática, incorporando recursos avançados ao SSC, como: monitoração de atividade do CLP por meio de *script*, cadastro e apresentação de mensagens de alarme, amostragem de valores decimais e apresentação dos mesmos por meio de curvas de tendência (incluindo a formação de históricos).

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com as plataformas STEP 7 e WinCC instaladas);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.

CONSIDERAÇÕES E TAREFAS PRELIMINARES

No primeiro experimento dessa aula, o módulo de *scripts* globais de sua plataforma de desenvolvimento começará a ser explorado, para a criação de funções do usuário. No entanto, para que as vantagens do encapsulamento de código em um SSC fiquem claras, primeiramente, vamos realizar algumas animações na tela “Rascunho”, sem o uso desse recurso. Para isso, cumpra as tarefas a seguir.

TAREFA 1 – Como é provável que sua bancada de trabalho seja compartilhada com outros estudantes, antes de mais nada, para garantir que o CLP esteja corretamente configurado, utilize a plataforma de programação deste dispositivo para compilar e fazer o *download* da versão mais recente do seu projeto. Além disso, certifique-se de que o módulo didático que faz a simulação do processo produtivo em sua bancada esteja ligado.

TAREFA 2 – Abra sua plataforma de desenvolvimento de SSCs e note que, provavelmente, um projeto foi aberto automaticamente. Se sua bancada for compartilhada com outros estudantes, é possível que esse projeto não seja o seu e que, primeiramente, você tenha que fecha-lo. Em seguida, localize e abra a versão mais recente do seu projeto.

TAREFA 3 – Abra a tela “Rascunho” e selecione o objeto usado para apresentar numericamente o nível de água no tanque. Faça um *script* para tornar dinâmica a cor do texto a ser apresentado, sendo que: na situação de normalidade, o valor do nível deve ser exibido na cor verde; a cor amarela deve ser usada quando o nível excede os limites alto ou baixo; a cor vermelha deve ser atribuída às situações em que o nível estiver muito alto ou muito baixo.



TAREFA 4 – Dupliche o objeto utilizado na tarefa anterior e faça as adaptações necessárias para obter uma indicação numérica da temperatura da água (não se esqueça de adaptar o *script* que faz a animação da cor do texto).

TAREFA 5 – Salve a tela “Rascunho”, ative o ambiente de operação do seu SSC e verifique se os valores de nível e temperatura estão sendo apresentados com a cor condizente às suas respectivas situações. Se necessário, faça operações com a bomba de água para alterar a situação do nível (para a temperatura, não são necessárias ações similares com o aquecedor).

TAREFA 6 – Supondo que seja necessário substituir “amarelo” (192,192,0) por “laranja” (192,96,48) nas situações em que uma grandeza analógica excede seus limites alto ou baixo, volte ao ambiente de engenharia do seu SSC e altere os objetos que indicam os valores de nível e temperatura. Salve as alterações e verifique os resultados no ambiente de operação.

TAREFA 7 – Reflita: e se a alteração de cores solicitada na tarefa anterior precisasse ser feita para dezenas de indicações de grandezas analógicas espalhadas em um sistema com várias telas? Entre vários outros casos, esse mostra que o encapsulamento de código em funções genéricas pode ser muito útil ao se desenvolver um SSC (veja o próximo experimento).

EXPERIMENTO 1

Nesse experimento, o módulo de *scripts* globais de sua plataforma de desenvolvimento começará a ser explorado. A lógica para alteração da cor do texto dos objetos manipulados nas tarefas anteriores será generalizada e encapsulada em uma função a ser criada por você. Dessa forma, se for necessário alterar novamente alguma cor relacionada às indicações numéricas de grandezas analógicas, a mudança poderá ser realizada em um único lugar, mesmo se o sistema contiver múltiplas indicações. Para isso, cumpra as tarefas a seguir.

TAREFA 1 – Abra o módulo de *scripts* globais e, usando a linguagem *Visual Basic Script* (VBS), crie uma função genérica para monitorar grandezas analógicas. Suas entradas devem receber os quatro alarmes relacionados ao valor monitorado (muito alto, alto, baixo e muito baixo), enquanto sua saída deve retornar a cor a ser aplicada ao objeto que fará a indicação numérica, sendo que: na situação de normalidade, a cor deve ser verde; a cor laranja deve ser retornada quando os alarmes alto ou baixo estiverem ativados; e a cor vermelha nos casos relacionados aos alarmes muito alto ou muito baixo. Ao final, salve a função.

Como?



TAREFA 2 – Volte à tela “Rascunho” e modifique o *script* associado ao objeto que faz a indicação do nível de água no tanque, de forma que a função desenvolvida na tarefa anterior passe a ser utilizada para determinar a cor dos valores numéricos a serem apresentados.

Como?



TAREFA 3 – Similarmente ao que foi feito na tarefa anterior, modifique o objeto que faz a indicação de temperatura. Ative o ambiente de operação e verifique se os valores de nível e temperatura da água estão sendo apresentados com a cor condizente às suas respectivas situações. Faça operações com a bomba de água para alterar a situação do nível (para a temperatura, você pode ligar e desligar o aquecedor usando as botoeiras físicas de sua bancada, desde que ele não esteja bloqueado pela falta de água no tanque).

EXPERIMENTO 2

Ainda em relação ao módulo de *scripts* globais, a criação de sub-rotinas cuja execução acontece em segundo plano será explorada nesse experimento. Mais especificamente, o objetivo das tarefas a seguir consiste na construção de uma funcionalidade para monitorar o sinal de atividade gerado pelo CLP, sendo que, em caso de congelamento, a variável interna “Aux_DVC01_Inativo” será levada a nível lógico alto.

TAREFA 1 – Volte ao módulo de *scripts* globais e, usando a linguagem *Visual Basic Script* (VBS), crie uma sub-rotina para monitorar a comunicação entre o SSC e o CLP de sua bancada por meio da verificação de atividade deste último. O algoritmo a ser implementado pode ser observado na Figura 7.1 e sua execução deve acontecer em segundo plano, ciclicamente, a cada 1 segundo. Ao terminar, salve a sub-rotina com o nome “DVC01_Watchdog”.

Como?

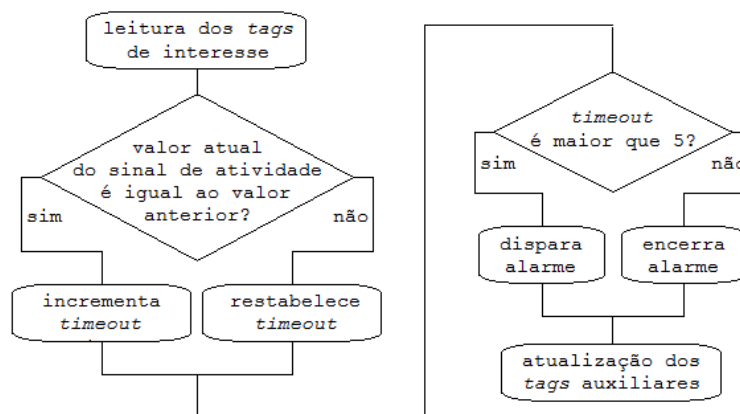


Figura 7.1: Algoritmo para verificação de atividade do CLP e eventual indicação de falhas na comunicação entre esse dispositivo e o SSC.

TAREFA 2 – Volte à tela “Rascunho” e acrescente um objeto para indicações textuais, com a mensagem “COMUNICAÇÃO INATIVA COM O CLP”. Ajuste as propriedades de aparência (posição, tamanho, cor de fundo, alinhamento do texto etc) e vincule sua visibilidade ao *tag* “Aux_DVC01_Inativo”. Salve as alterações e ative o ambiente de operação.

Como?



TAREFA 3 – Para testar, usando sua plataforma de programação de CLPs, faça com que o ciclo de funcionamento da CPU seja interrompido. Realize algumas transições ($RUN \rightleftharpoons STOP$) e verifique se o comportamento da mensagem configurada na tarefa anterior está coerente (ao terminar, deixe seu CLP em operação normal).

EXPERIMENTO 3

Nesse experimento, os recursos da sua plataforma de desenvolvimento relacionados ao cadastro e apresentação de mensagens de alarme serão explorados. O objetivo é permitir que o usuário final gerencie as mensagens que estiverem presentes no sistema em determinado momento e também consulte o histórico dos acontecimentos relacionados a elas. Para isso, cumpra as tarefas a seguir, sendo que é interessante notar que elas podem ser organizadas em dois grupos: primeiramente, faz-se as configurações básicas e o cadastro das mensagens em um módulo específico para isso; em seguida, as tarefas passam a ser executadas no módulo de desenvolvimento de telas, com os esforços voltados para a apresentação dessas mensagens.

TAREFA 1 – Abra o módulo destinado ao cadastro de alarmes e configure quais informações irão compor cada mensagem, sendo elas: DATA, no formato “dd/MM/yyyy”; HORÁRIO, no formato “HH:mm:ss”; e descrição textual da MENSAGEM, com quantidade adequada de caracteres.

Como?



TAREFA 2 – Configure o esquema de cores das mensagens, conforme especificações que podem ser observadas na Tabela 7.1.

Como?



Tabela 7.1: Esquema de cores para as mensagens de alarme do sistema.

Status da mensagem	Cor do texto	Cor do fundo
Presente	vermelho (255,0,0)	preto (0,0,0)
Encerrada	azul (0,128,255)	preto (0,0,0)
Reconhecida	verde (0,255,0)	preto (0,0,0)

TAREFA 3 – Cadastre as mensagens de alarme do sistema, de acordo com as relações entre *tag* de disparo e descrição textual especificadas na Tabela 7.2.

Como?



Tabela 7.2: Especificação das mensagens de alarme a serem cadastradas no SSC, com seus respectivos *tags* de disparo e descrições textuais.

Número da mensagem	Tag de disparo	Descrição textual
1	Aux_DVC01_Inativo	CLP inativo
2	Aux_DVC02_Inativo	Termo-Higrômetro inativo
3	Alr_GERAL_EmgAtv	Condição de emergência ativada
4	Alr_LT101_ExdHH	Nível muito alto
5	Alr_LT101_ExdH	Nível alto
6	Alr_TT101_ExdHH	Temperatura muito alta
7	Alr_TT101_ExdH	Temperatura alta
8	Alr_TT101_ExdL	Temperatura baixa
9	Alr_TT101_ExdLL	Temperatura muito baixa
10	Alr_AQ101_HorFim	Aquecedor com horas esgotadas
11	Alr_AQ101_RetFal	Falha no retroaviso do aquecedor
12	Alr_VT101_HorFim	Ventilador com horas esgotadas

TAREFA 4 – Abra a tela “Rascunho” e acrescente um quadro para apresentação de mensagens de alarme. Ajuste as propriedades de posição e tamanho do objeto e configure-o para mostrar apenas as colunas DATA, HORÁRIO e MENSAGEM. Além disso, reduza as funções disponíveis na barra de ferramentas, deixando apenas: botões para alterar entre mensagens momentâneas e mensagens em histórico; botão para reconhecimento de mensagens; e botão para configuração de filtros de mensagens. Ao final, salve as alterações.



TAREFA 5 – Ative o ambiente de operação do SSC e faça testes disparando, reconhecendo e encerrando algumas mensagens (não necessariamente nessa ordem). Algumas sugestões para fazer isso são:

- Por meio da plataforma de programação de CLPs, interrompa e reinicie o ciclo de funcionamento da CPU;
- Ative e rearme a condição de emergência;
- Por meio das botoeiras físicas de sua bancada, faça manobras com o aquecedor (e veja os alarmes de temperatura);
- Por meio das botões virtuais do seu SSC, faça manobras com a bomba de água (e veja os alarmes de nível).

TAREFA 6 – Para finalizar o experimento, use a barra de ferramentas do quadro de alarmes para: alterar entre a visualização de mensagens momentâneas e em histórico; filtrar mensagens de acordo com o horário em que ocorreram; filtrar mensagens de acordo com o elemento da planta envolvido no problema.

EXPERIMENTO 4

Os recursos relacionados à amostragem de variáveis decimais e a posterior apresentação de curvas de tendência serão explorados nesse experimento. O objetivo é permitir que se possa acompanhar graficamente as principais variáveis do sistema, por meio de valores recentes e em histórico. Assim como no experimento anterior, é interessante notar que as tarefas a seguir podem ser organizadas em dois grupos: em um módulo específico da plataforma, configura-se a amostragem das variáveis de interesse; posteriormente, usa-se o módulo de desenvolvimento de telas para que os valores armazenados possam ser apresentados ao usuário final.

TAREFA 1 – Abra o módulo destinado ao cadastro de variáveis a serem amostradas e crie as seguintes pastas organizadoras: “Levels”, “Others”, “Params” e “Temper”.



TAREFA 2 – Adicione as variáveis indicadas na Tabela 7.3 às respectivas pastas organizadoras, configurando o SSC com as características de amostragem de cada uma.



Tabela 7.3: Especificação das variáveis a serem amostradas pelo SSC, com suas respectivas características de aquisição e armazenamento.

Pasta	Tag a ser amostrado	Tipo de aquisição	Ciclo de aquisição	Fator de armazen.	Ciclo de armazen.
Levels	Ana_LT101_NivEng	Cyclical, cont.	1s	1	1s
Others	Ana_BA101_VelEng	Cyclical, cont.	1s	1	1s
Others	Ana_LC102_SinCtr	Cyclical, cont.	1s	1	1s
Params	Par_LC101_LimDsg	Cyclical, cont.	1s	2	1s
Params	Par_LC101_LimLig	Cyclical, cont.	1s	2	1s
Params	Par_LC102_SetPnt	Cyclical, cont.	1s	2	1s
Params	Par_LT101_LimHH	Cyclical, cont.	1s	2	1s
Params	Par_LT101_LimH	Cyclical, cont.	1s	2	1s
Params	Par_TC101_LimDsg	Cyclical, cont.	1s	2	1s
Params	Par_TC101_LimLig	Cyclical, cont.	1s	2	1s
Params	Par_TT101_LimHH	Cyclical, cont.	1s	2	1s
Params	Par_TT101_LimH	Cyclical, cont.	1s	2	1s
Params	Par_TT101_LimL	Cyclical, cont.	1s	2	1s
Params	Par_TT101_LimLL	Cyclical, cont.	1s	2	1s
Temper	Ana_TT101_TmpEng	Cyclical, cont.	1s	5	1s

TAREFA 3 – Abra a tela “Rascunho” e acrescente um quadro para apresentação de curvas de tendência. Ajuste as propriedades de posição e tamanho do objeto e configure-o para mostrar um intervalo de 5 minutos no eixo do TEMPO e valores em dois eixos de ordenadas: MILÍMETROS (com escala de 0 a 200 mm) e PERCENTUAL (com escala de 0 a 100).

Como?



TAREFA 4 – Ainda em relação ao quadro para apresentação de curvas de tendência, configure as curvas especificadas na Tabela 7.4. Além disso, reduza as funções disponíveis na barra de ferramentas, deixando apenas: botão para alterar entre acessos a valores momentâneos ou em histórico; e botão para configuração de eixos e curvas. Ao final, salve as alterações.

Como?



Tabela 7.4: Especificação das curvas de tendência a serem configuradas.

Nome da curva	Eixos vinculados	Caminho para o tag em arquivo	Cor
NivEng	TEMPO E MILÍMETROS	Levels\Ana_LT101_NivEng	Preto
LimHH	TEMPO E MILÍMETROS	Params\Par_LT101_LimHH	Vermelho
LimH	TEMPO E MILÍMETROS	Params\Par_LT101_LimH	Laranja
VelEng	TEMPO E PERCENTUAL	Others\Ana_BA101_VelEng	Magenta

TAREFA 5 – Ative o ambiente de operação do SSC para testar o que foi desenvolvido, manobrando a bomba e acompanhando os valores nível e velocidade. Para finalizar, use a barra de ferramentas do quadro de curvas de tendência para: alterar entre a visualização de valores recentes e em histórico; configurar novos eixos de ordenadas (CELSIUS, por exemplo); configurar novas curvas (valor atual e limites de temperatura, por exemplo).

FINALIZANDO

Ao longo dos últimos experimentos, você usou o módulo de *scripts* globais de sua plataforma de desenvolvimento para criar uma função com código encapsulado e uma sub-rotina cuja execução se dá em segundo plano. Além disso, foram configurados os módulos relacionados ao registro de mensagens de alarme e à amostragem de variáveis decimais, bem como os respectivos objetos na tela “Rascunho”, destinados a apresentar tais informações aos usuários. Sendo assim, se você assimilou bem esses conceitos e não houver dúvidas em relação ao que foi desenvolvido, realize as tarefas de encerramento a seguir (caso contrário, antes de terminar, faça os esclarecimentos necessários com o docente).

TAREFA 1 – Se o ambiente “*runtime*” do seu SSC estiver em execução, faça a desativação. Feche os módulos de configuração (gerenciamento de *tags*, edição de telas etc) que porventura estejam abertos e, em seguida, faça *backup* do seu projeto.



TAREFA 2 – Feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

7.6 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

Desenvolver as sub-rotinas que fazem o gerenciamento da válvula de dreno direto (VD101) e da válvula de dreno refrigerado (VD102), de acordo com as orientações fornecidas no **Descritivo Funcional** do sistema.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.

DESENVOLVIMENTO DA SUB-ROTINA “VD101”

A sub-rotina VD101 (Válvula de dreno direto), entre outras coisas, deverá permitir que este atuador seja acionado por decisões humanas (tanto por interface local como por interface remota) ou por métodos automáticos. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto.

TAREFA 2 – Encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado ao atuador VD101 (veja a Subseção 5.8 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:



- Leia a descrição geral da sub-rotina;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (VD101) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Informe-se sobre as particularidades do atuador.

TAREFA 3 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

DESENVOLVIMENTO DA SUB-ROTINA “VD102”

A sub-rotina VD102 (Válvula de dreno refrigerado), entre outras coisas, deverá permitir que este atuador seja acionado por decisões humanas (tanto por interface local como por interface remota) ou por métodos automáticos. Para desenvolvê-la, cumpra as tarefas a seguir.

TAREFA 1 – Considerando que a versão mais recente do seu projeto está aberta, encontre, no **Descritivo Funcional** do sistema de automação, o trecho dedicado ao atuador VD102 (veja a Subseção 5.9 do documento que pode ser acessado por meio do *link* ao lado) e, em seguida:



- Leia a descrição geral da sub-rotina;
- Faça uma análise cuidadosa da relação de variáveis auxiliares, procurando entender o propósito de cada uma;
- Abra o bloco de programação apropriado (VD102) e implemente o diagrama *ladder* de acordo com as orientações fornecidas;
- Informe-se sobre as particularidades do atuador.

TAREFA 2 – Caso sua plataforma de desenvolvimento esteja em modo *online*, faça a desconexão. Salve seu projeto, compile e faça o *download* das modificações realizadas.

FINALIZANDO

Nos dois experimentos anteriores você deve ter implementado as sub-rotinas VD101 e VD102. Se não houver dúvidas em relação ao que foi desenvolvido, realize as costumeiras tarefas de encerramento (caso contrário, antes de terminar, anote suas dúvidas e procure esclarecê-las com o docente assim que possível).

TAREFA 1 – Se sua plataforma de desenvolvimento estiver *online* com seu CLP, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto.



TAREFA 2 – Quando encerrar suas atividades, feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

7.7 EXERCÍCIOS

QUESTÃO 1 – Para cada uma das *funcionalidades* dos Sistemas de Supervisão e Controle mencionadas abaixo, faça uma descrição conforme o padrão apresentado a seguir.

- *Gerenciamento de variáveis* – Após a configuração dos *drivers* de comunicação, devem ser definidas as variáveis a serem utilizadas, sendo essencial conhecer o tipo de dado e o endereço de cada uma nos CLPs. A quantidade de variáveis que poderão ser utilizadas em um Sistema de Supervisão normalmente está vinculada à versão da licença de *software* adquirida e à capacidade de processamento do *driver* de comunicação.
- a) Construção de telas com animações gráficas.
- b) Registro e apresentação de mensagens de alarme.
- c) Registro e apresentação de gráficos de tendência.
- d) Desenvolvimento de subrotinas e funções com o módulo de *scripts* globais.

QUESTÃO 2 – Atualmente, o *Visual Basic Script* (VBS) é a linguagem de programação adotada pela maioria das plataformas de sistemas SCADA para permitir a criação de funcionalidades não previstas originalmente nessas plataformas, facilitando também o tratamento de situações mais complexas, como a animação de objetos em que vários sinais devem ser considerados. Por exemplo, a animação de cor de uma válvula motorizada, para a qual deve-se considerar os sinais: aberta, fechada, abrindo, fechando e defeito. Cite *dois* outros exemplos em que a utilização de *scripts* de programação pode ser útil.

QUESTÃO 3 – ENADE 2008 – Em uma Interface Homem-Máquina (IHM) os gráficos de tendência mostram, através de sua imagem gráfica, como determinadas variáveis de processo mudam ao longo do tempo. Os dados mostrados podem ser obtidos em tempo real, sincronizados com o tempo de varredura do Controlador Lógico Programável (CLP), ou podem advir de um histórico arquivado.

Com base nessas informações, considere as afirmações a seguir.

- I – Os dados mostrados na IHM podem ser utilizados na análise de tendência do processo.
- II – As variáveis de processo podem ser arquivadas para garantir a conformidade com leis federais ou outras regulamentações.
- III – Por meio da IHM o operador terá condições de monitorar a eficiência da produção.
- IV – Por meio da IHM o operador terá condições de alterar o tempo de varredura do CLP.

São corretas APENAS as afirmações:

- (a) I e IV
- (b) II e IV
- (c) I, II e III
- (d) II, III e IV
- (e) I, II, III e IV

QUESTÃO 4 – A apresentação de mensagens textuais de alarme aos operadores consiste em um recurso muito utilizado nos Sistemas de Supervisão e Controle para que falhas e situações indesejadas do processo sejam anunciadas com clareza. É bastante comum que uma mensagem de alarme seja composta das seguintes informações, exceto:

- (a) Dados estatísticos do problema que está ocorrendo.
- (b) Nome do usuário que estava operando o sistema no momento de ocorrência.
- (c) Descrição da falha ou situação indesejada ocorrida.
- (d) Data e hora de ocorrência.

QUESTÃO 5 – Considere a tela de um Sistema de Supervisão e Controle que pode ser observada na Figura 7.2. Nela, aparecem os campos de entrada e saída de dados que permitem ao operador definir os pesos dos materiais a serem carregados em um alto-forno por meio de correias transportadoras (dessa forma, é possível criar “receitas” para o ferro gusa a ser produzido). No que tange ao Sistema Supervisório, o desenvolvimento dessa funcionalidade, a princípio seria simples, bastando cumprir os seguintes passos:

- Para cada um dos valores de peso a ser definido, criar *tags* de comunicação com o CLP em questão e especificar corretamente o tipo de dado e os respectivos endereços de memória em que os valores devem ser escritos;
- Na tela desejada, inserir os *I/O Fields* e associá-los aos respectivos *tags*, além de criar textos descritivos para completar a tabela e orientar o operador sobre o que significa cada valor.

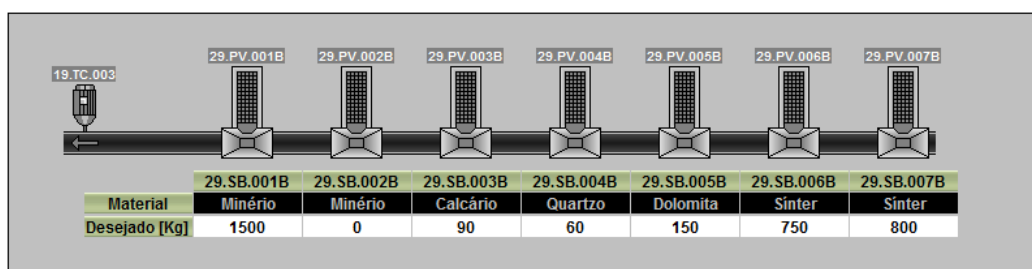


Figura 7.2: Campos de entrada e saída de dados que permitem ao operador definir os pesos dos materiais a serem carregados em um alto-forno.

No entanto, há um problema de sincronismo a ser resolvido: quando o operador digita um valor em um dos *I/O Fields* que está associado a um *tag* de comunicação, ao pressionar a tecla “Enter”, este valor é transferido à posição de memória correspondente do CLP de forma praticamente imediata (não importa se a comunicação é direta ou por OPC). Assim, o operador vai digitando os valores de uma nova receita e eles vão sendo transferidos um a um para o CLP. Essa é uma situação indesejada, pois quando o CLP inicia a preparação de uma receita, ele assume ao mesmo tempo todos os valores que estão nas posições de memória em que o Supervisório escreve, definindo esses valores como meta para o sistema de pesagem de

materiais. Assim, caso o operador esteja digitando os valores no Supervisório no momento em que uma nova receita é assumida pelo CLP, é bastante provável que os valores da receita antiga se misturem com os valores da receita que está em processo de digitação.

Sendo assim, apresente um texto descritivo com indicações claras do que deve ser feito para solucionar este problema, sabendo que:

- O CLP não possui posições de memória para receber comandos como “assumir receita” e não é interessante modificar sua programação para tal (aliás, o programa do CLP não deve ser modificado de forma alguma);
- Não é possível alterar o comportamento dos objetos do tipo *I/O Field* que os faz transferir o valor digitado para o *tag* associado a eles assim que a tecla “Enter” é pressionada.

QUESTÃO 6 – Suponha que, em um Sistema de Automação já existente, foi necessário fazer a inclusão da medição de temperatura do mancal de um determinado motor. Os objetivos são apresentar, no Sistema de Supervisão e Controle, o valor atual da nova temperatura e também uma mensagem de alarme, caso a mesma exceda seu limite máximo. Após a conexão do transmissor do instrumento à uma entrada analógica do CLP, o procedimento adotado para cumprir os objetivos foi o seguinte:

No CLP:

1. Mover o valor recebido da entrada analógica (0 à 65535) para a posição de memória MW50;
2. Escalonar o valor de MW50 para a faixa entre 0 e 200 [°C] e mover o resultado para MW60;
3. Comparar o valor de MW60 com 120 [°C]:
 - Se for menor, atribuir 0 à posição de memória M70.0;
 - Se for maior, atribuir 1 à posição de memória M70.0;
4. Fim.

No WinCC:

1. No gerenciador de tags (*Tag Management*):
 - (a) Criar o tag “TempMancal_ValorAtual” (tipo inteiro) com o endereço MW60;
 - (b) Criar o tag “TempMancal_ValorAlto” (tipo binário) com o endereço M70.0;
2. No editor de telas (*Graphics Designer*):
 - (a) Inserir um campo de apresentação de valores na tela apropriada;
 - (b) Associar esse campo ao tag “TempMancal_ValorAtual”;
3. Na ferramenta de cadastro de alarmes (*Alarm Logging*):
 - (a) Inserir a mensagem com a descrição “Temperatura Alta no Mancal”;
 - (b) Associar essa mensagem ao tag “TempMancal_ValorAlto”;
4. Fim.

Agora, suponha que os mesmos objetivos devam ser cumpridos, mas com a exigência de que a modificação no CLP seja mínima. Sendo assim, seguindo o mesmo padrão de descrição usado acima, você deverá completar o procedimento iniciado a seguir:

No CLP:

1. Mover o valor recebido da entrada analógica (0 à 65535) para a posição de memória MW50;
2. Fim.

No WinCC:

1. No gerenciador de tags (*Tag Management*):
 - (a) Criar o tag “TempMancal_ValorBruto” (tipo inteiro) com o endereço MW50;
 - (b) ...

QUESTÃO 7 – A plataforma de desenvolvimento de Sistemas Supervisórios *WinCC* (usada nas aulas práticas da disciplina), assim como várias outras plataformas, oferece diferentes possibilidades para se atingir um mesmo objetivo. Como exemplo, considere a seguinte situação:

- A posição de memória MD16 do CLP-01 armazena um valor de temperatura que pode variar entre $-20,0^{\circ}C$ e $120,0^{\circ}C$ (note a casa decimal);
- Este valor de temperatura deve ser mostrado aos operadores em uma tela do Sistema de Supervisão e Controle. Porém, neste último, os valores são apresentados em outra unidade, em que:
 1. O valor $-20,0^{\circ}C$ (no CLP-01) corresponde ao valor $-4,0^{\circ}F$ (na IHM);
 2. O valor $120,0^{\circ}C$ (no CLP-01) corresponde ao valor $248,0^{\circ}F$ (na IHM);
 3. A conversão é linear;
 4. Note que o valor a ser apresentado também possui uma casa decimal.

Sendo que, uma das possíveis soluções para esta situação é descrita a seguir.

- No módulo *Tag Management*, considerando que o *driver* “S7 Protocol Suite” já foi inserido e que, tanto o canal “TCP/IP” como a conexão com o CLP-01 já foram devidamente configuradas, faça o seguinte procedimento:
 - Dentro da conexão com CLP-01, criar o *tag* denominado “CLP01_TEMP01”;
 - Configurar esse *tag* como “Floating-point number”;
 - Associar este *tag* ao endereço MD16.
- No módulo *Graphics Designer*, considerando que a tela na qual a temperatura deve ser apresentada já foi criada, faça o seguinte procedimento:
 - Dentro da tela, criar um objeto do tipo *IO/Filed*;
 - Associar um “C-Script” à propriedade “Output value” deste *IO/Filed*;

- Na janela de *scripts*, escrever o seguinte código:

```
float TempCel;  
float TempFah;  
  
TempCel = GetTagFloat("CLP01_TEMP01");  
TempFah = (1.8 * TempCel) + 32.0;  
return TempFah;
```

- Ainda na janela de *scripts*, configurar o *tag* “CLP01_TEMP01” como *trigger* para execução do código.

Esta é uma forma de solucionar o problema, mas existem, pelo menos, dois outros caminhos. Sendo assim, utilize o mesmo padrão de descrição de procedimentos fornecido acima para atender às seguintes questões:

a) Descreva um procedimento para solucionar o mesmo problema utilizando também apenas os módulos *Tag Management* e *Graphics Designer*. Soluções em que se propõe apenas a mudança da linguagem C para VBS ou para *Dynamic Dialog* não serão aceitas. Dica: na verdade, nenhum *script* precisa ser implementado para que se obtenha essa solução.

b) Descreva um procedimento para solucionar o mesmo problema, agora utilizando os módulos *Tag Management*, *Global Script* e *Graphics Designer*. Dica: no módulo *Global Script*, use apenas uma *Action* (subrotina que roda em *background*) para executar um *script* muito similar ao apresentado acima.

7.8 REFERÊNCIAS

- AG, Siemens (ed.). **WinCC V7.0 SP3: ANSI-C for Creating Functions and Actions**. Printout of the Online Help. Germany, Nürnberg, 2011. Disponível em: <https://drive.google.com/open?id=1IoBT9YtDVkd8zM0ZzuT1i7UEgWdf90KS>.
- AG, Siemens (ed.). **WinCC V7.0 SP3: Archiving Process Values**. Printout of the Online Help. Germany, Nürnberg, 2011. Disponível em: https://drive.google.com/open?id=1lDi2zXoLnadNvN_VtoCRxxmp-utIkn0B.
- AG, Siemens (ed.). **WinCC V7.0 SP3: Display of Messages during Runtime**. Printout of the Online Help. Germany, Nürnberg, 2011. Disponível em: https://drive.google.com/open?id=1H0h4pHaFDuz5sDU1UBbTm_iddkg_BIV0.
- AG, Siemens (ed.). **WinCC V7.0 SP3: Process Value Output as a Function of Another Tag**. Printout of the Online Help. Germany, Nürnberg, 2011. Disponível em: <https://drive.google.com/open?id=15Yo19LdEW1SIMBKlvUVGydIM09uuHscW>.
- AG, Siemens (ed.). **WinCC V7.0 SP3: Setting up a Message System**. Printout of the Online Help. Germany, Nürnberg, 2011. Disponível em: https://drive.google.com/open?id=1Sae6A1JRhzWKYJwbe0AvzYRSh_1KA-xE.

Capítulo 8

CONCEITOS FUNDAMENTAIS SOBRE OPC

8.1 INTRODUÇÃO

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação).

8.2 MOTIVAÇÕES PARA O SURGIMENTO

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Material	Capítulo	Seção
“A Evolução das Tecnologias OPC como Subsídio para as Fábricas Inteligentes”	—	—

8.3 PRINCÍPIO DE FUNCIONAMENTO

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Material	Capítulo	Seção
OPC: The Ins and Outs to What It's About	—	—

8.4 TECNOLOGIAS OPC

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação). Além disso, você pode estudar de acordo com as seguintes indicações de leitura:

Material	Capítulo	Seção
“Desenvolvimento de Servidores OPC DA, OPC UA e Wrap-pers para aplicação em Automação”	2	2.1 a 2.5
	3	3.1 a 3.4

8.5 CONSIDERAÇÕES FINAIS

Enquanto não temos texto para esta seção, você pode estudar pelos *slides* apresentados em aula (e anotações que você deve ter feito durante a apresentação).

8.6 ROTEIRO PARA AULA PRÁTICA

OBJETIVO

Explorar a tecnologia OPC por meio de cenários de comunicação a serem estabelecidos entre novos elementos e os já existentes no projeto que vem sendo desenvolvido ao longo do semestre.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com as ferramentas STEP 7, KEPServerEX, WinCC, OpenOffice e Scilab);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo.



CONSIDERAÇÕES E TAREFAS PRELIMINARES

Como participante deste curso, essas serão suas primeiras atividades práticas relacionadas à tecnologia OPC. No entanto, antes de configurar o servidor e os clientes, prepare sua bancada de trabalho e organize seus arquivos conforme solicitado nas tarefas a seguir.

TAREFA 1 – Como é provável que sua bancada de trabalho seja compartilhada com outros estudantes, antes de mais nada, para garantir que o CLP esteja corretamente configurado, utilize a plataforma de programação deste dispositivo para compilar e fazer o *download* da versão mais recente do seu projeto. Além disso, certifique-se de que o módulo didático que faz a simulação do processo produtivo em sua bancada esteja ligado.

TAREFA 2 – Abra a plataforma de comunicação KEPServerEX, crie um projeto e salve-o em local apropriado, dentro da estrutura de diretórios que você utiliza para armazenar seus arquivos.



CONFIGURAÇÃO DO SERVIDOR OPC

Nesta seção, um servidor OPC será configurado para se comunicar com dois dispositivos (por meio de seus respectivos protocolos nativos): o seu CLP (via S7-Protocol) e um termohigrômetro (via Modbus TCP) a ser introduzido no sistema de automação. Observação: nas próximas seções, outros elementos serão configurados como clientes OPC, de forma que possam ler e escrever dados na memória dos mencionados dispositivos.

TAREFA 1 – Adicione um canal de comunicação do tipo “Siemens TCP/IP Ethernet” ao seu servidor e configure-o com o nome “KEPasS7Server” (deixe os demais atributos com valores *default*).



TAREFA 2 – Associe seu CLP ao canal de comunicação criado na tarefa anterior, especificando o nome, a família e o endereço IP, conforme informações disponíveis em sua bancada (deixe os demais atributos com valores *default*).



TAREFA 3 – Adicione os *tags* de interesse à conexão recém-criada, configurando-os de acordo com o que se observa na Tabela 8.1.



Tabela 8.1: *Tags* a serem declarados no servidor OPC, para escrita e leitura de valores na memória do CLP.

Nome	Tipo de dado	Endereço	Comentário
BmbLig	Bool	M104.0	Liga bomba
BmbDsg	Bool	M104.1	Desliga bomba
BmbFun	Bool	M204.0	Bomba funcionando
NivNow	Float	MD704	Nível de água no tanque (milímetros)
BmbVel	Float	MD836	Referência de velocidade (percentual)

TAREFA 4 – Adicione outro canal de comunicação ao seu servidor, agora do tipo “Modbus TCP/IP Ethernet”. Configure-o com o nome “KEPasMBServer” e mantenha as demais propriedades com valores *default*.



TAREFA 5 – Associe o termo-higrômetro ao canal recém-criado. Configure o nome do dispositivo e o seu endereço, de acordo com as informações disponíveis em sua bancada (deixe os demais atributos com valores *default*).



TAREFA 6 – Selecione a conexão criada na tarefa anterior e cadastre os *tags* cujas características podem ser observadas na Tabela 8.2.



Tabela 8.2: *Tags* a serem declarados no servidor OPC, para leitura de valores na memória do termo-higrômetro.

Nome	Tipo de dado	Endereço	Comentário
SinAtv	Word	30001	Sinalização de atividade do TH101
TmpEng	Word	30002	Temperatura ambiente (Celsius)
UmdEng	Word	30003	Umidade relativa do ar (percentual)

TAREFA 7 – Salve seu trabalho e faça um teste para verificar se a comunicação entre o servidor OPC e os demais dispositivos está funcionando. Para isso, abra a ferramenta auxiliar “Quick Client” e confira se todas as variáveis cadastradas possuem o status “Good”. Se sim, feche o “Quick Client” e o configurador do KEPServerEX e siga para o próximo experimento. Se não, reconfigure os itens que apresentaram problema.



CONFIGURANDO SEU SSC PARA SER UM CLIENTE OPC

Na seção anterior, entre outras coisas, o servidor OPC instalado no PC da sua bancada foi configurado para se comunicar com um termo-higrômetro, por meio do protocolo Modbus TCP. Agora, o objetivo é configurar o seu SSC como um cliente OPC, de forma que ele possa se conectar ao servidor e, dessa forma, ler os valores de umidade relativa do ar e temperatura ambiente fornecidos por esse medidor. Para isso, cumpra as tarefas a seguir.

TAREFA 1 – Abra sua plataforma de desenvolvimento de SSCs e note que, provavelmente, um projeto foi aberto automaticamente. Se sua bancada for compartilhada com outros estudantes, é possível que esse projeto não seja o seu e que, primeiramente, você tenha que fechá-lo. Em seguida, localize e abra a versão mais recente do seu projeto.

TAREFA 2 – Abra o módulo de gerenciamento de *tags* e adicione o *driver* de comunicação necessário para tornar o seu SSC um cliente OPC. Em seguida, a partir do *driver* que acabou de ser instalado, procure o servidor que foi configurado na seção anterior e faça a importação dos três *tags* relacionados ao termo-higrômetro: “Ana_TH101_SinAtv”, “Ana_TH101_UmdEng” e “Ana_TH101_TmpEng”.

Como?



TAREFA 3 – Abra a tela “Rascunho”, duplique os objetos relacionados à apresentação numérica da temperatura da água e faça as adaptações necessárias para obter uma indicação numérica da temperatura ambiente (não se esqueça de excluir o *script* que faz a animação da cor do texto, uma vez que não há alarmes relacionados a esta última grandeza). Em seguida, repita esse procedimento, de forma a obter uma indicação da umidade relativa do ar. Ao final, salve sua tela.

Como?



TAREFA 4 – Com o objetivo de monitorar o sinal de atividade gerado pelo termo-higrômetro, abra o módulo de *scripts* globais e duplique a sub-rotina “DVC01_Watchdog”, renomeando a nova instância para “DVC02_Watchdog”. Em seguida, faça a adequação dos *tags* usados no código, substituindo “Ana_GERAL_SinAtv” por “Ana_TH101_SinAtv” e o termo “DVC01” por “DVC02” nos demais. Ao final, salve seu *script*.

Como?



TAREFA 5 – Abra o módulo destinado ao cadastro de variáveis a serem amostradas e adicione as variáveis indicadas na Tabela 8.3 às respectivas pastas organizadoras, configurando o SSC com as características de amostragem de cada uma.

Tabela 8.3: Especificação das variáveis a serem acrescentadas no sistema de amostragem.

Pasta	Tag a ser amostrado	Tipo de aquisição	Ciclo de aquisição	Fator de armazen.	Ciclo de armazen.
Others	Ana_TH101_UmdEng	Cyclical, cont.	1s	5	1s
Temper	Ana_TH101_TmpEng	Cyclical, cont.	1s	5	1s

TAREFA 6 – Ative o ambiente de operação do SSC e verifique se as indicações numéricas da temperatura ambiente e umidade relativa estão coerentes. Em seguida, use a barra de ferramentas do quadro de curvas de tendência para configurar a apresentação dessas grandezas (se necessário, crie os eixos de ordenadas “CELSIUS” e “PERCENTUAL”).



TAREFA 7 – Interrompa a execução do servidor OPC e confira se a mensagem “Termo-Higrômetro inativo” será apresentada no quadro de alarmes do SSC. Observação: após alguns segundos, o servidor será reativado automaticamente (por requisição do cliente) e a mensagem em questão desaparecerá assim que for reconhecida pelo usuário.




CONFIGURANDO O MS-EXCEL COMO CLIENTE OPC

No início deste roteiro, o servidor OPC instalado em seu PC foi configurado para se comunicar com o CLP de sua bancada, por meio do protocolo nativo deste último (S7-Protocol). Agora, o objetivo é configurar o *Microsoft Excel* como um cliente OPC, de forma que ele possa se conectar ao servidor e, dessa forma, ler e escrever valores na memória do CLP. Para isso, cumpra as tarefas a seguir.

TAREFA 1 – Abra o Microsoft Excel. Inicie a instalação do suplemento de interesse seguindo o caminho: *Ferramentas* → *Suplementos* → *Procurar*. Na janela de navegação, clique no botão “browse” e indique o arquivo *D:\Rede\Downloads\OPCExcelAddin.xla*. Após selecionar o arquivo, marque o *checkbox* do suplemento e clique “Ok”.

TAREFA 2 – Autorize a execução de macros por meio de: *Ferramentas* → *Macro* → *Segurança*. Selecione “Baixo” e clique “Ok”. Para utilizar as ferramentas disponíveis nas barras “Formulários” e “Visual basic”, ative a apresentação das mesmas seguindo: *Exibir* → *Barra de Ferramentas*. Feche o Excel e abra-o novamente (as configurações realizadas permanecerão para usos futuros).

TAREFA 3 – Na barra de ferramentas “Visual Basic”, clique no ícone  (Editor de Visual Basic). Após a abertura do editor, crie um novo módulo seguindo: *Inserir* → *Módulo*. Digite as 3 macros a seguir, **substituindo os caracteres “##” pelo número do CLP de sua bancada**. Após a digitação, salve o trabalho realizado no editor de Visual Basic.


```
Sub Liga()
Call Excel.Application.<<obs: faltou espaço no papel, mas o Run vem logo aqui>>
Run("OPCExcelAddin.XLA!OPCWrite", "KEPasS7Server.LII-CLP-##.BmbLig", 1, "")
End Sub
```

```
Sub Desliga()
Call Excel.Application.<<obs: faltou espaço no papel, mas o Run vem logo aqui>>
Run("OPCExcelAddin.XLA!OPCWrite", "KEPasS7Server.LII-CLP-##.BmbDsg", 1, "")
End Sub
```

```

Sub Escreve()
Call Excel.Application.<<obs: faltou espaço no papel, mas o Run vem logo aqui>>
Run("OPCExcelAddin.XLA!OPCWrite", "KEPasS7Server.LII-CLP-##.BmbVel", Cells(6,4), "")
End Sub

```

TAREFA 4 – Na barra de ferramentas “Formulários”, clique no ícone  (Botão) e utilize o cursor para posicionar o objeto na planilha. A janela “Atribuir macro” será aberta automaticamente. Utilize-a para associar uma macro ao botão. Repita esta tarefa de modo que o usuário possa fazer uso das 3 macros programadas anteriormente.

TAREFA 5 – Para exibir informações ao usuário, configure as 6 células indicadas a seguir de acordo com seus respectivos conteúdos, **substituindo os caracteres “##” pelo número do CLP de sua bancada**.

F2	ESTADO
F3	=SE(OPC("KEPasS7Server.LII-CLP-##.BmbFun")=VERDADEIRO;"On";"Off")
B5	NÍVEL ATUAL
B6	=OPC("KEPasS7Server.LII-CLP-##.NivNow")
D5	VELOCIDADE DESEJADA
D6	25

TAREFA 6 – Altere o tamanho e a posição dos botões, as cores e o alinhamento dos textos das células, organizando a interface de operação, de forma que o resultado final fique parecido com o que está sugerido na Figura 8.1.

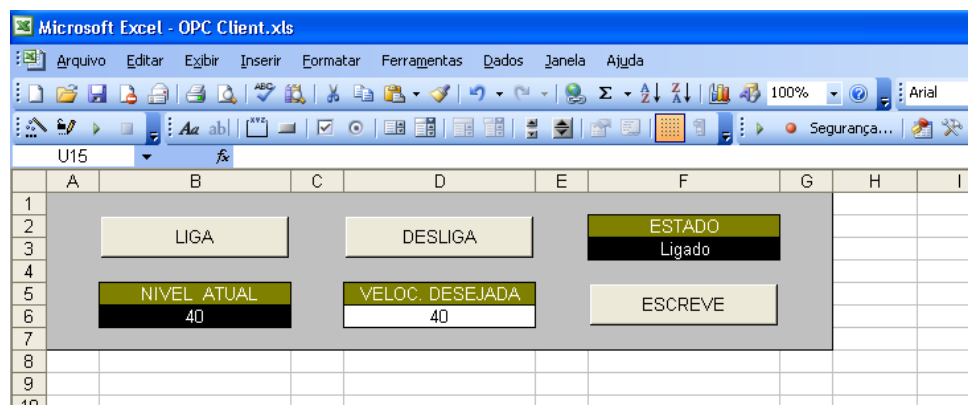


Figura 8.1: Aspecto desejado para interface de operação a ser desenvolvida.

TAREFA 7 – Faça alguns testes e verifique se todos os elementos de interface estão funcionando. Se estiver tudo ok, salve a planilha em sua pasta de trabalho e siga para as tarefas de encerramento da aula; caso contrário, procure resolver o(s) problema(s) e, se não conseguir, peça ajuda ao docente.

FINALIZANDO

TAREFA 1 – Se o ambiente “*runtime*” do seu SSC estiver em execução, faça a desativação. Feche os módulos de configuração (gerenciamento de *tags*, edição de telas etc) que porventura estejam abertos e, em seguida, faça *backup* do seu projeto.

Como?



TAREFA 2 – Feche todas as ferramentas que estiverem abertas em seu PC e desligue-o. Desligue também todos os módulos didáticos da sua bancada de trabalho e, ao sair, deixe o local limpo e organizado.

8.7 TAREFAS PARA A PRÓXIMA SEMANA

OBJETIVO

A partir de um pacote de telas preconcebidas, a ser incorporado no aplicativo produzido durante as últimas aulas práticas, iniciar o desenvolvimento de um SSC com características semelhantes aos utilizados na indústria.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (STEP 7 e WinCC);
- Módulo para simulação de processo produtivo.



CONSIDERAÇÕES E TAREFAS PRELIMINARES

O desenvolvimento profissional de um SSC raramente parte da estaca zero. Isso porque alguns recursos como navegação entre telas, abertura de janelas *popup*, apresentação de mensagens de alarme e curvas de tendência são comuns à grande maioria das aplicações. Sendo assim, o que os desenvolvedores fazem é manter um pacote de telas com esses recursos básicos e, a partir dele, construir as telas específicas para um determinado processo industrial. Uma vez que você já cadastrou em seu sistema, entre outras coisas, os *tags* de comunicação com o CLP, as mensagens de alarme e as variáveis cujos valores devem ser armazenados em banco de dados, cumpra as tarefas a seguir para incorporar ao mesmo as telas que permitem aos usuários uma interação amigável com esses recursos.

TAREFA 1 – Como é provável que sua bancada de trabalho seja compartilhada com outros estudantes, antes de mais nada, para garantir que o CLP esteja corretamente configurado, utilize a plataforma de programação deste dispositivo para compilar e fazer o *download* da versão mais recente do seu projeto.

TAREFA 2 – Antes de abrir sua plataforma de desenvolvimento de SSCs, utilize o *link* ao lado para fazer o *download* do mencionado pacote de telas. Em seguida, descompacte-o dentro da estrutura de diretórios referente à versão mais recente do seu projeto.



TAREFA 3 – Abra a versão mais recente do seu projeto em sua plataforma de desenvolvimento de SSCs, certifique-se de que os arquivos manipulados na tarefa anterior foram realmente incorporados ao sistema e, em caso afirmativo, aproveite para configurar a tela “Principal” como sendo a primeira a ser aberta após a ativação do ambiente de operação (tela inicial do sistema).



TAREFA 4 – Ative o ambiente de operação, explore os recursos incorporados ao sistema e, para dar início ao desenvolvimento da tela “Processo”, siga para o próximo tópico.



DESENVOLVIMENTO DA INTERFACE DE OPERAÇÃO PARA A BOMBA DE ÁGUA

TAREFA 1 – Abra a tela “Biblioteca” e copie o objeto adequado para fazer a indicação do nome e do modo de operação de um atuador. Cole uma instância desse objeto na tela “Processo” e faça as seguintes configurações:



- Atribuição de valores estáticos às propriedades:
 - Coordenadas X e Y: livre
 - Texto: BA-101
 - *Tooltip*: Bomba de água
- Adequação do *script* associado à cor de fundo:
 - Leitura do *tag* “Par_BA101_ModOpr”

TAREFA 2 – Abra a tela “Biblioteca” e copie o objeto adequado para a representação gráfica de um atuador e seus estados. Cole uma instância desse objeto na tela “Processo” e faça as seguintes configurações:



- Atribuição de valores estáticos às propriedades:
 - Coordenadas X e Y: próximas ao objeto que indica o nome do atuador (BA-101)
 - Tamanho: igual ao das figuras que serão embutidas (“bmb01a_@@@.png”)
 - Figura associada ao índice 3: nenhuma (exclua a figura pré-configurada)
 - Figura associada ao índice 2: bmb01a_amr.png
 - Figura associada ao índice 1: bmb01a_vrd.png
 - Figura associada ao índice 0: bmb01a_azl.png
- Adequação do *script* associado ao índice que controla a figura a ser apresentada:
 - Exclusão dos trechos relacionados ao estado “defeito”
 - Leitura dos *tags* “Stt_BA101_Interlock” e “Stt_BA101_Ligada”
- Adequação do *script* associado ao evento gerado pelo clique do *mouse*:
 - Nome da tela a ser aberta como janela: wBA101.Pdl
 - Título da janela: BOMBA DE ÁGUA

TAREFA 3 – Abra a tela “Biblioteca” e copie o objeto adequado para a indicação numérica de grandezas analógicas. Cole uma instância desse objeto na tela “Processo” e faça as seguintes configurações:



- Atribuição de valores estáticos às propriedades:
 - Coordenadas X e Y: próximas ao objeto que representa o atuador (BA-101)
 - Título: VELOCIDADE
 - Unidade: [%]
 - *Tooltip*: Velocidade da bomba
 - Formato de saída: 999,9
- Associação direta de *tags* às propriedades:
 - Valor de saída: Ana_BA101_VelEng (atualização a cada 1s)
- Adequação do *script* associado à cor dos valores apresentados:
 - Exclusão do *script* (não será usado nesse caso)
- Adequação do *script* associado ao evento gerado pelo clique do *mouse*:
 - Exclusão do *script* (não será usado nesse caso)

TAREFA 4 – Salve sua tela, volte ao ambiente de operação (alt+tab) e faça uma navegação rápida entre telas (“Processo” \rightleftharpoons “Início”, por exemplo). Isso é suficiente para que sua tela de processo seja atualizada com as modificações mais recentes, permitindo que você verifique:

- Se, após um clique na representação da BA-101, a janela de controle é aberta;
- Se os modos de operação da BA-101 estão sendo corretamente indicados na tela “Processo” (magenta, para interface remota; e ciano, para os demais);
- Se os estados da BA-101 estão sendo corretamente apresentados na tela “Processo” (azul, para desligada; verde, para ligada; e amarelo, para bloqueada);
- Se a velocidade da BA-101 está sendo corretamente apresentada na tela “Processo”.

DESENVOLVIMENTO DA INTERFACE PARA MONITORAÇÃO DO NÍVEL DE ÁGUA NO TANQUE

TAREFA 1 – Abra a tela “Biblioteca” e copie o objeto adequado para a indicação gráfica de grandezas analógicas. Cole uma instância desse objeto na tela “Processo” e faça as seguintes configurações:



- Atribuição de valores estáticos às propriedades:
 - Coordenadas X e Y: posicione o objeto no centro da tela
- Adequação do *script* associado ao preenchimento da barra:
 - Leitura do *tag* “Ana_LT101_NivEng”
 - Definição do valor máximo: 200

TAREFA 2 – Inclua, na tela “Processo”, um objeto capaz de embutir figuras externas e faça as seguintes configurações:

Como?



- Atribuição de valores estáticos às propriedades:
 - Coordenadas X e Y: deixe para a próxima tarefa
 - Tamanho: igual ao da figura que será embutida (“tnq01.png”)
 - Figura associada ao objeto: tnq01.png
 - Cor transparente da figura: laranja (255,128,64)
 - Cor transparente da figura ativada: sim
 - Esquema global de cores: não

TAREFA 3 – Sobreponha o objeto criado na Tarefa 2 ao objeto criado na Tarefa 1, de forma que o retângulo transparente fique centralizado em cima do *bargraph*.

TAREFA 4 – Abra a tela “Biblioteca” e copie o objeto adequado para a indicação numérica de grandezas analógicas. Cole uma instância desse objeto na tela “Processo” e faça as seguintes configurações:

Como?



- Atribuição de valores estáticos às propriedades:
 - Coordenadas X e Y: à direita do objeto que representa o tanque
 - Título: LT-101
 - Unidade: [mm]
 - *Tooltip*: Nível de água no tanque
 - Formato de saída: 999
- Associação direta de *tags* às propriedades:
 - Valor de saída: Ana_LT101_NivEng (atualização a cada 1s)
- Adequação do *script* associado à cor dos valores apresentados:
 - Leitura dos *tags* “Alr_LT101_ExdHH”, “Alr_LT101_ExdH” e demais alarmes
- Adequação do *script* associado ao evento gerado pelo clique do *mouse*:
 - Nome da tela a ser aberta como janela: wLT101.Pdl
 - Título da janela: NÍVEL DE ÁGUA NO TANQUE

TAREFA 5 – Salve sua tela, volte ao ambiente de operação (alt+tab) e faça uma navegação rápida entre telas (“Processo” \rightleftharpoons “Início”, por exemplo). Isso é suficiente para que sua tela de processo seja atualizada com as modificações mais recentes, permitindo que você verifique:

- Se, o nível de água está sendo coerentemente indicado por meio do objeto que representa graficamente o tanque (faça manobras com a bomba de água, se necessário);
- Se, após um clique no indicador numérico LT-101, a janela de parâmetros é aberta;

- Se, o nível de água está sendo corretamente indicado de forma numérica, incluindo a animação de cores que aparece no valor apresentado (vermelho, para valores que excedem os limites muito alto ou muito baixo; laranja, para valores altos ou baixos; e verde, para valores dentro da faixa de normalidade).

8.8 EXERCÍCIOS

QUESTÃO 1 – Com relação à tecnologia OPC, responda:

- a) Um servidor e um cliente OPC podem ser executados paralelamente pelo mesmo microcomputador? Justifique.
- b) Um servidor OPC pode se comunicar diretamente com um CLP utilizando o protocolo nativo desse CLP? Justifique.
- c) Um sistema de automação de um determinado processo industrial possui dois CLPs de fabricantes diferentes. Existem também dois servidores OPC, um para cada CLP. Nesse caso, é possível utilizar um Sistema de Supervisão de um terceiro fabricante, em um único microcomputador, como interface para todo o processo? Justifique.
- d) Um sistema de automação de um determinado processo industrial possui um único CLP. Existe também um servidor OPC para esse CLP. Nesse caso, é possível usar dois Sistemas de Supervisão de um segundo e de um terceiro fabricante, em microcomputadores diferentes, como pontos de interface para o processo? Justifique.
- e) Um sistema de automação de um determinado processo industrial possui dois CLPs de fabricantes diferentes. Pretende-se utilizar também um Sistema de Supervisão de um terceiro fabricante, em um único microcomputador, como interface para todo o processo. Nesse caso, é possível utilizar um único servidor OPC para viabilizar a comunicação desse sistema de automação? Justifique.

QUESTÃO 2 – Comente as afirmações sobre a tecnologia OPC apresentadas a seguir, dizendo se estão certas ou erradas e elaborando sua resposta da seguinte maneira:

- Se uma determinada afirmação estiver correta, dê um exemplo de utilização do cenário descrito por ela (use descrições e diagramas em seus exemplos).
 - Se uma determinada afirmação estiver errada, faça uma justificativa clara do motivo.
- a) Se um Sistema de Supervisão tem a capacidade de ser cliente-OPC, ele pode se comunicar com qualquer CLP do mercado.
 - b) Geralmente, um servidor-OPC é executado pela CPU do CLP para o qual ele foi desenvolvido.
 - c) Se um Sistema de Supervisão tem a capacidade de ser cliente-OPC, é possível que ele troque dados com vários CLPs, de diferentes modelos, praticamente ao mesmo tempo.
 - d) Geralmente, um servidor-OPC comunica-se com um único CLP, mas podem existir servidores comunicando-se com uma quantidade maior de CLPs, de diferentes modelos, praticamente ao mesmo tempo.

QUESTÃO 3 – A Figura 8.2 descreve possíveis cenários de comunicação entre seres humanos. Esse cenário pode ser visto como uma analogia em relação a comunicação entre dispositivos de um sistema de automação. Dessa forma, faça um novo diagrama para esclarecer essa relação, substituindo pessoas por CLPs, Estações de Sistemas Supervisão ou Softwares.

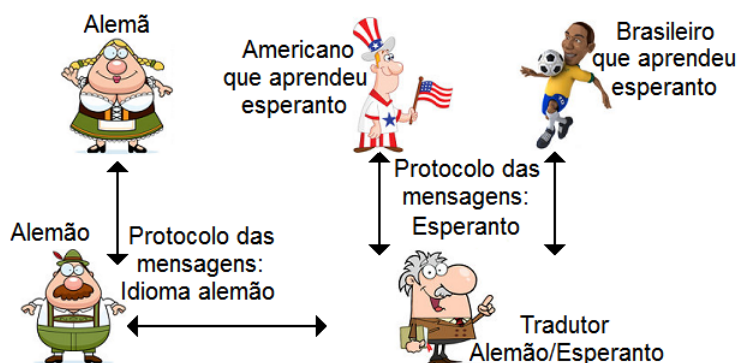


Figura 8.2: Comunicação entre seres humanos, vista em possível analogia com a comunicação entre elementos de um sistema de automação industrial.

Importante: note os dois pontos em que o texto “Protocolo das mensagens:” aparece. Você deve mantê-los em seu novo diagrama, porém indicando protocolos como TCP/IP, Padrão OPC, Ethernet, S7-Functions, Profibus, Device/Net, etc (escolha os tipos corretos ou faça uma descrição, caso você não lembre o nome).

QUESTÃO 4 – Suponha que você trabalha em uma empresa integradora ¹ de sistemas de automação industrial. Seu colega, do departamento comercial, se aproxima e pergunta:

“Estou preparando uma proposta comercial para a venda de um sistema com base na Plataforma de Desenvolvimento de Sistemas Supervisórios “X” e no CLP modelo “Y”. O Sistema Supervisório irá rodar em um *PC* com placa de rede convencional e o CLP possui uma porta integrada para comunicação TCP/IP (*RJ45*) em sua CPU. Preciso incluir um *OPC Server* no orçamento desta proposta?”

Analizando tais afirmações, você deve responder:

- (a) Sim
(b) Não
(c) Talvez

Escreva uma justificativa para sua resposta, mostrando com clareza que você conhece os principais conceitos da tecnologia OPC e que, devido a isso, marcou a alternativa acima com consciência (independentemente de qual tenha sido ela).

¹ Basicamente, empresas integradoras de sistemas de automação industrial desempenham o seguinte papel no mercado: após fecharem um contrato de fornecimento com um determinado cliente, adquirem *software* e *hardware* de fabricantes como a Siemens, Rockwell e GE; desenvolvem o(s) programa(s) do(s) CLP(s) e o sistema supervisão, geralmente em sua sede; e fazem o comissionamento do sistema, ou seja, após concluir o desenvolvimento, vão ao processo industrial do cliente que os contratou e colocam o sistema para rodar.

QUESTÃO 5 – Considere o cenário ilustrado na Figura 8.3 e também as afirmações feitas a seguir.

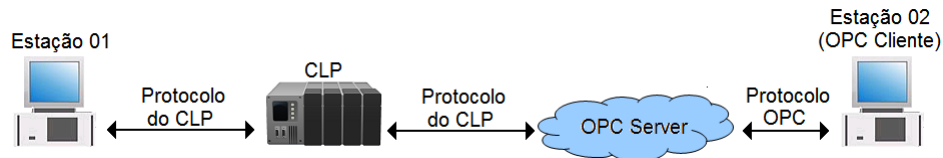


Figura 8.3: Exemplo de comunicação entre um CLP e dois Sistemas de Supervisão e Controle.

- No CLP, existe uma subrotina para o acionamento de uma bomba de água similar à que você programou no 1º trabalho prático da disciplina;
- No momento, esta bomba encontra-se desligada, mas está em modo manual e nenhum defeito ou intertravamento impedem o seu possível acionamento;
- A Supervisão e o Controle desta bomba foram implementados nas duas Estações que aparecem na figura.

Se esta bomba receber o sinal de comando “Liga” de qualquer uma das Estações, ela será acionada, mas a questão é: a animação feita na outra Estação (mesmo tendo sido implementada corretamente) irá mostrar que a bomba está funcionando?

- a) Sim, independentemente de qual estação enviou o comando, a animação funcionará nas duas Estações.
- b) Talvez, se o comando partir da Estação que está se comunicando diretamente, a animação funcionará nas duas Estações. Mas, se o comando for enviado da Estação que está se comunicando via OPC, a animação funcionará apenas nesta última.
- c) Talvez, se o comando partir da Estação que está se comunicando via OPC, a animação funcionará nas duas Estações. Mas, se o comando for enviado da Estação que está se comunicando diretamente, a animação funcionará apenas nesta última.
- d) Não, independentemente de qual estação enviou o comando, a animação funcionará apenas nela mesma.

Escreva uma justificativa para sua resposta.

8.9 REFERÊNCIAS

COELHO, M.T. *et al.* A Evolução das Tecnologias OPC como Subsídio para as Fábricas Inteligentes. *In: XIV CEEL – Conferência de Estudos em Eng. Elétrica.* Uberlândia, MG, 2016.

GONÇALVES, R.N. **Desenvolvimento de Servidores OPC DA, OPC UA e Wrappers para aplicação em Automação.** Dissertação (Mestrado em Engenharia Elétrica) – Universidade Federal de Itajubá, Itajubá, MG, 2012. Disponível em: https://drive.google.com/open?id=1oWR00TDbWpreSt3VtIBAC41EAGnLz_8H.

KOMINEK, D. **OPC: The Ins and Outs to What It's About:** The Every Man's Guide to OPC. Alberta, Canada, 2009. https://www.automation.com/pdf_articles/Guide_to OPC.pdf. [Online; accessed 31-outubro-2018].

ENCERRAMENTO DA SEGUNDA ETAPA

As avaliações intermediárias do curso devem ocorrer após a finalização do tópico anterior. Dessa forma, as respostas dos exercícios desenvolvidos para contribuir com sua preparação para a avaliação teórica foram reunidos na próxima seção. Além disso, na seção seguinte, há um pequeno roteiro a ser cumprido em laboratório, com os preparativos para que o trabalho prático possa ser devidamente avaliado.

RESPOSTAS DOS EXERCÍCIOS

Respostas dos exercícios	Link
Capítulo 5	▼
Capítulo 6	▼
Capítulo 7	▼
Capítulo 8	▼

ROTEIRO PARA APRESENTAÇÃO DO TRABALHO PRÁTICO

OBJETIVO

Realizar os preparativos para que as funcionalidades do trabalho prático previstas até o momento possam ser devidamente avaliadas.

MATERIAL NECESSÁRIO

- Bancada de trabalho com CLP e PC (com plataforma de programação instalada);
- Módulo com elementos elétricos de interface;
- Módulo para simulação de processo produtivo;

TAREFAS

TAREFA 1 – Abra sua plataforma de programação de CLPs e, em seguida, localize e abra a versão mais recente do seu projeto. Antes de prosseguir, faça o mesmo para o seu SSC.

TAREFA 2 – Faça o procedimento de *download* para o CLP de sua bancada e ative o ambiente de operação do seu SSC. Em seguida, prepare seu sistema para ser avaliado, conferindo e testando as funcionalidades previstas até o momento. Consulte o docente sobre o limite de tempo para esta tarefa e, até o horário estipulado, sinta-se livre para fazer os últimos ajustes nos projetos (CLP e SSC).

TAREFA 3 – Se sua plataforma de desenvolvimento estiver *online* com seu CLP, faça a desconexão. Em seguida, salve e faça *backup* do seu projeto (atribua os nomes dos integrantes da dupla ao arquivo compactado).



TAREFA 4 – Se o ambiente “*runtime*” do seu SSC estiver em execução, faça a desativação. Feche os módulos de configuração (gerenciamento de *tags*, edição de telas etc) que porventura estejam abertos e faça *backup* do seu projeto (atribua os nomes dos integrantes da dupla ao arquivo compactado).



TAREFA 5 – Consulte o docente sobre como proceder com os arquivos gerados nas tarefas anteriores (eles serão essenciais caso uma revisão da avaliação venha a ser solicitada). Antes de deixar o laboratório, ative novamente o ambiente de operação do seu SSC e certifique-se de que a comunicação com o CLP da sua bancada tenha sido estabelecida. Ao sair, deixe os projetos abertos e o local limpo e organizado.