LabAuto — Laboratório de Automação Inteligente

Automação Industrial

Fundamentos, CLPs, Controle PID e SCADA

CDSA/UFCG · Sumé–PB · 2025 · Engenharia de Produção e Biossistemas

Sumário

Capítulo 1 — Fundamentos de Automação

1.1 O que é automação industrial?

Automação industrial é o uso de sistemas de controle — como controladores lógicos programáveis (CLPs), computadores industriais e robôs — para operar equipamentos e processos com mínima intervenção humana. O objetivo principal é aumentar a produtividade, melhorar a qualidade dos produtos, reduzir custos operacionais e garantir a segurança dos trabalhadores.

Breve Histórico — As Revoluções Industriais

1ª Revolução Industrial (séc. XVIII): Introdução da máquina a vapor e mecanização da produção têxtil na Inglaterra. A energia hidráulica e o vapor substituíram o trabalho manual e animal, marcando o início da produção fabril.

2ª Revolução Industrial (séc. XIX–XX): Uso da eletricidade e do petróleo como fontes de energia. Surgiram as linhas de montagem (Henry Ford), a produção em massa e os primeiros dispositivos de controle elétrico, como relés e contatores. O telégrafo e o telefone possibilitaram comunicação a distância.

3ª Revolução Industrial (déc. 1960–2000): Revolução digital. Surgimento dos computadores, CLPs (o primeiro CLP, Modicon 084, data de 1969), robôs industriais e redes de comunicação. A automação passou a integrar lógica programável, permitindo flexibilidade nos processos produtivos.

4ª Revolução Industrial (Indústria 4.0): Integração de IoT (Internet das Coisas), computação em nuvem, inteligência artificial, big data e sistemas ciberfísicos. Fábricas inteligentes com tomada de decisão autônoma e comunicação máquina-a-máquina (M2M) em tempo real.

Benefícios da Automação Industrial

- Aumento de produtividade e throughput
- Melhoria da qualidade e repetibilidade do produto
- Redução de custos operacionais a longo prazo
- Maior segurança para os operadores (remoção de tarefas perigosas)
- Rastreabilidade e coleta de dados do processo
- Flexibilidade na produção (reprogramação de CLPs e robôs)

1.2 Hierarquia de automação — Pirâmide ISA-95

A norma ISA-95 (ANSI/ISA-95) estabelece um modelo hierárquico para a integração de sistemas de automação com sistemas corporativos. A pirâmide de automação organiza os níveis de decisão e controle em camadas:

Nível	Nome	Função	Exemplos
5	Corporativo	Planejamento estratégico e financeiro	ERP (SAP, TOTVS)
4	Gestão da Planta	Programação da produção e logística	MES, APS
3	Supervisão	Monitoramento e controle supervisório	SCADA, IHM
2	Controle	Controle de processos em tempo real	CLP, DCS, PID
1	Dispositivos Inteligentes	Sensores e atuadores com comunicação	Transmissores HART, drives
0	Campo	Processo físico	Válvulas, motores, sensores simples

Tabela 1 — Níveis da pirâmide de automação ISA-95.

A integração entre níveis é fundamental para a Indústria 4.0. A convergência de TI (Tecnologia da Informação) e TA (Tecnologia de Automação) permite que dados do chão de fábrica alimentem sistemas corporativos em tempo real, possibilitando decisões mais rápidas e precisas.

1.3 Tipos de controle: malha aberta vs. malha fechada

Controle em Malha Aberta: O sinal de controle é enviado ao atuador sem realimentação da saída. O controlador não sabe se o objetivo foi atingido. Exemplo: uma torradeira com temporizador — ela aquece por um tempo fixo, sem medir a cor da torrada.

Controle em Malha Fechada (Feedback): A variável de saída é medida por um sensor e comparada com o valor desejado (setpoint). O erro resultante alimenta o controlador, que ajusta a saída do atuador para reduzir o erro. Este é o princípio da realimentação negativa, base de quase todos os sistemas de controle industrial.

Diagrama de blocos — Malha Fechada

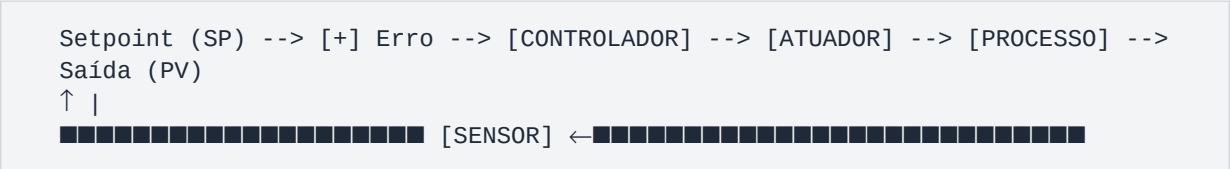

Figura 1 — Diagrama de blocos de um sistema de controle em malha fechada.

No diagrama acima, SP (setpoint) é o valor desejado, PV (process variable) é o valor medido, e o erro é a diferença e(t) = SP − PV. O controlador calcula a ação de controle para minimizar esse erro ao longo do tempo.

1.4 Variáveis de processo

As principais variáveis controladas em processos industriais são:

Variável	Unidade SI	Sensores Típicos	Aplicações
Temperatura	°C ou K	Termopar, PT100, NTC	Fornos, estufas, reatores
Pressão	Pa, bar, psi	Piezelétrico, Bourdon	Caldeiras, tubulações, HVAC
Nível	m, mm, %	Ultrassônico, boia, capacitivo	Tanques, reservatórios
Vazão	m^3/h, L/min	Eletromagnético, Coriolis, vórtice	Tubulações, dosagem
Posição	mm, graus	Encoder, LVDT, potenciômetro	Válvulas, robôs, eixos CNC

Tabela 2 — Variáveis de processo, sensores e aplicações típicas.

A seleção correta do sensor é crítica: fatores como faixa de medição, precisão, tempo de resposta, resistência ao ambiente (temperatura, vibração, corrosão) e protocolo de comunicação (4–20 mA, HART, Modbus) devem ser considerados.

Capítulo 2 — Controladores Lógicos Programáveis (CLP)

2.1 Arquitetura do CLP

O Controlador Lógico Programável (CLP) é um computador industrial projetado para operar em ambientes severos (poeira, vibração, temperaturas extremas) e executar lógica de controle em tempo real. Sua arquitetura básica compreende os seguintes componentes:

CPU (Unidade Central de Processamento): Executa o programa do usuário, gerencia a comunicação e coordena os módulos de E/S. Processadores modernos operam com ciclos de varredura na faixa de 1 a 10 ms para programas típicos.

Módulos de Entrada: Recebem sinais do campo. Podem ser digitais (24 V DC, contatos secos) ou analógicos (0–10 V, 4–20 mA). Cada módulo possui isolação galvânica para proteger a CPU.

Módulos de Saída: Enviam sinais para atuadores. Saídas digitais (relé, transistor) acionam contatores, válvulas solenoides e lâmpadas. Saídas analógicas (0–10 V, 4–20 mA) controlam drives de frequência, válvulas proporcionais e posicionadores.

Rack (Base) e Barramento: Estrutura física que aloja a CPU e os módulos. O barramento interno (backplane) transfere dados entre os módulos e a CPU em alta velocidade.

Fonte de Alimentação: Converte a tensão da rede (110/220 V AC) para 24 V DC ou 5 V DC, alimentando a CPU e os módulos. Fontes redundantes são usadas em aplicações críticas.

Interfaces de Comunicação: Ethernet/IP, PROFINET, Modbus TCP/RTU, RS-485. Permitem integração com IHMs, sistemas SCADA e outros CLPs.

2.2 Ciclo de varredura (Scan Cycle)

O CLP opera em um ciclo contínuo e determinístico chamado ciclo de varredura (scan cycle), que se repete indefinidamente enquanto o CLP está em modo RUN. O ciclo consiste em três etapas principais:

1. Leitura das Entradas (Input Scan): O CLP lê o estado de todos os módulos de entrada e armazena os valores na memória de imagem de entrada. Durante a execução do programa, os valores das entradas permanecem congelados (não são atualizados até o próximo ciclo).

2. Execução do Programa (Program Execution): A CPU executa o programa do usuário linha a linha (rung a rung em Ladder), da esquerda para a direita e de cima para baixo. Os resultados são escritos na memória de imagem de saída.

3. Atualização das Saídas (Output Scan): Os valores da memória de imagem de saída são transferidos para os módulos de saída físicos, acionando os atuadores.

Adicionalmente, o CLP executa tarefas de manutenção (housekeeping): diagnósticos internos, comunicação com IHMs e verificação de integridade da memória. O tempo total do ciclo é

chamado de scan time, tipicamente entre 1 e 50 ms, dependendo do tamanho do programa e da velocidade da CPU.

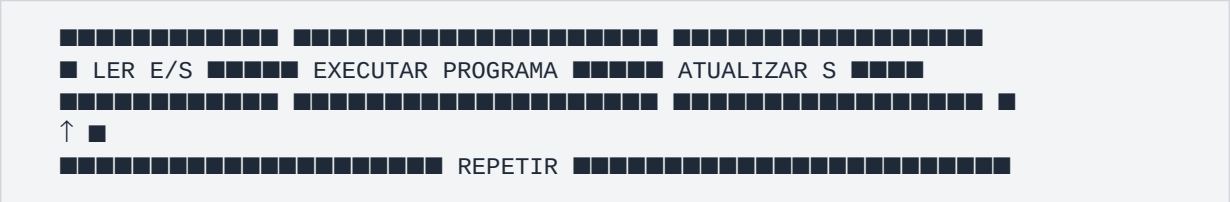

Figura 2 — Ciclo de varredura do CLP.

2.3 Linguagens de programação IEC 61131-3

A norma IEC 61131-3 padroniza cinco linguagens de programação para CLPs, permitindo que programadores escolham a linguagem mais adequada para cada tipo de aplicação:

Linguagem	Sigla	Tipo	Descrição
Ladder Diagram	LD	Gráfica	Baseada em diagramas de relés elétricos. Mais utilizada mundialmente. Intuitiva para eletricistas e técnicos.
Function Block Diagram	FBD	Gráfica	Baseada em blocos funcionais interconectados. Ideal para controle de processos contínuos (PID, filtros).
Instruction List	IL	Textual	Semelhante a assembly. Baixo nível, eficiente em memória. Em desuso na indústria moderna.
Structured Text	ST	Textual	Similar a Pascal/C. Ideal para cálculos matemáticos complexos, manipulação de strings e algoritmos.
Sequential Function Chart	SFC	Gráfica	Descreve processos sequenciais com etapas e transições. Ideal para bateladas, partida/parada de máquinas.

Tabela 3 — Linguagens de programação IEC 61131-3.

Na prática industrial brasileira, Ladder (LD) é a linguagem predominante em aplicações de automação discreta (acionamentos, intertravamentos), enquanto Structured Text (ST) ganha popularidade em aplicações que envolvem cálculos matemáticos e processamento de dados.

2.4 Exemplos de programação Ladder

A linguagem Ladder representa a lógica de controle como um circuito elétrico. Os elementos básicos são:

- Contato Normalmente Aberto (NA) —[ ]—: Conduz quando a variável associada é TRUE (1).
- Contato Normalmente Fechado (NF) —[/]—: Conduz quando a variável associada é FALSE (0).
- Bobina —( )—: É energizada quando a lógica à esquerda resulta em TRUE.

- Bobina SET (S) e RESET (R): SET ativa a saída e a mantém, RESET desativa.

Exemplo 1: Partida direta de motor com selo

```
   | I0.0 I0.1 Q0.0 |
   |--[ ]--+--[/]-----( )---| I0.0 = Botão Liga (NA)
   | | | I0.1 = Botão Desliga (NF)
   | Q0.0 | | Q0.0 = Contator do Motor
   |--[ ]--+ |
   | |
```

Figura 3 — Ladder: partida direta com selo (auto-retenção).

Neste circuito, ao pressionar I0.0 (botão liga), a bobina Q0.0 é energizada. O contato Q0.0 em paralelo com I0.0 faz o selo: mesmo soltando o botão, o motor permanece ligado. Pressionar I0.1 (botão desliga, NF) interrompe a corrente e desliga o motor.

Temporizadores e Contadores

Temporizadores e contadores são blocos fundamentais em programas Ladder:

Bloco	Função	Parâmetros
TON	Temporização na energização (atraso para ligar)	IN (entrada), PT (preset time), Q (saída), ET (tempo decorrido)
TOF	Temporização na desenergização (atraso para desligar)	IN, PT, Q, ET
TP	Pulso de duração fixa	IN, PT, Q, ET
CTU	Contador crescente (Count Up)	CU (clock), R (reset), PV (preset), Q (saída), CV (valor atual)
CTD	Contador decrescente (Count Down)	CD (clock), LD (load), PV, Q, CV

Tabela 4 — Temporizadores e contadores em CLP.

Exemplo 2: Motor com temporização para desligar após 10 s

```
   | I0.0 TON |
   |--[ ]---[IN Q]--( Q0.0)| TON configurado:
   | [PT=10s ] | PT = T#10s
   | [ET ] | Ao ativar I0.0, Q0.0 liga
   | | após 10 segundos.
```

Figura 4 — Ladder com temporizador TON.

2.5 Fabricantes e modelos

Os principais fabricantes de CLPs e seus modelos mais utilizados no mercado brasileiro são:

Fabricante	Modelos	Software	Comunicação

Siemens	S7-1200, S7-1500, S7-300 (legado)	TIA Portal	PROFINET, PROFIBUS, Modbus TCP
Allen-Bradley (Rockwell)	CompactLogix 5380, Micro820/850	Studio 5000, CCW	EtherNet/IP, DeviceNet
Schneider Electric	M221, M241, M340, M580	SoMachine, EcoStruxure	Modbus TCP/RTU, CANopen
WEG	PLC300, TPW-04	WPS (WEG Programming Suite)	Modbus RTU, Ethernet
Altus	Nexto Xpress, Duo, DUO340	MasterTool IEC XE	PROFINET, Modbus, EtherCAT

Tabela 5 — Principais fabricantes de CLPs e seus modelos.

Para aplicações didáticas e prototipação, existem simuladores de CLP gratuitos como o OpenPLC (open-source, compatível com IEC 61131-3), o CODESYS (ambiente de desenvolvimento com simulador integrado) e o PLCfiddle (simulador online de Ladder). Esses recursos são valiosos no contexto acadêmico para praticar programação sem necessidade de hardware físico.

Capítulo 3 — Controle PID

3.1 O controlador PID

O controlador PID (Proporcional-Integral-Derivativo) é o algoritmo de controle mais utilizado na indústria, presente em mais de 90% das malhas de controle industriais. Ele calcula a ação de controle u(t) com base no erro e(t) entre o setpoint (SP) e a variável de processo (PV):

O PID combina três ações de controle complementares:

Ação Proporcional (P): Produz uma saída proporcional ao erro atual. Quanto maior o erro, maior a correção. O ganho proporcional Kp determina a intensidade da resposta. Uma ação P pura sempre deixa um erro residual em regime permanente (offset).

Ação Integral (I): Acumula o erro ao longo do tempo. Elimina o offset em regime permanente, pois enquanto houver erro (mesmo pequeno), a integral continua crescendo, forçando a saída a se ajustar. O parâmetro Ki (ou o tempo integral Ti = Kp/Ki) define a velocidade de eliminação do offset.

Ação Derivativa (D): Responde à taxa de variação do erro (ou da PV). Tem efeito antecipatório: se o erro está diminuindo rapidamente, a ação D reduz a saída para evitar overshoot. O parâmetro Kd (ou o tempo derivativo Td = Kd/Kp) controla essa antecipação.

3.2 Equação do PID

Forma contínua (domínio do tempo):

```
u(t) = Kp · e(t) + Ki · ∫e(τ)dτ + Kd · de(t)/dt

Onde:
e(t) = SP − PV(t) (erro)
Kp = ganho proporcional
Ki = ganho integral (Ki = Kp / Ti)
Kd = ganho derivativo (Kd = Kp · Td)
```

Forma discreta (implementação digital):

Em CLPs e microcontroladores, o PID é implementado na forma discreta com período de amostragem Ts:

```
u[k] = Kp · e[k] + Ki · Ts · Σ e[j] + Kd · (e[k] − e[k−1]) / Ts

Forma de velocidade (incremental):
Δu[k] = Kp · (e[k] − e[k−1])
+ Ki · Ts · e[k]
+ Kd · (e[k] − 2·e[k−1] + e[k−2]) / Ts

u[k] = u[k−1] + Δu[k]
```

A forma incremental (de velocidade) é preferida na prática porque facilita a implementação de anti-windup e a transferência bumpless entre modos manual e automático.

3.3 Efeitos de Kp, Ki e Kd na resposta do sistema

Parâmetro	Aumento causa	Efeito no tempo de subida	Overshoot	Erro em regime (offset)	Estabilidade
Kp	Maior ganho	Diminui	Aumenta	Reduz (não elimina)	Pode degradar
Ki	Acúmulo mais rápido	Diminui	Aumenta	Elimina	Pode degradar
Kd	Maior amortecimento	Pouco efeito	Diminui	Não afeta	Melhora (em geral)

Tabela 6 — Efeitos dos parâmetros PID na resposta do sistema.

Na prática, a sintonia do PID é um compromisso: Kp alto dá resposta rápida mas pode causar oscilação; Ki alto elimina offset mas pode gerar overshoot e oscilação; Kd alto suaviza a resposta mas amplifica ruído de medição.

3.4 Métodos de sintonia

Método de Ziegler-Nichols — Malha Aberta (Curva de Reação)

Aplica-se um degrau na saída do controlador (em manual) e registra-se a resposta da PV. Da curva de reação, extraem-se: o ganho do processo Kpu, o atraso de transporte L e a constante de tempo T. Os parâmetros são calculados pela tabela:

Controlador	Kp	Ti	Td
P	T / (Kpu · L)	∞	0
PI	0,9 · T / (Kpu · L)	L / 0,3	0
PID	1,2 · T / (Kpu · L)	2 · L	0,5 · L

Tabela 7 — Ziegler-Nichols, método da curva de reação (malha aberta).

Método de Ziegler-Nichols — Malha Fechada (Oscilação Sustentada)

Com controle apenas proporcional (Ki = 0, Kd = 0), aumenta-se Kp gradualmente até o sistema entrar em oscilação sustentada (amplitude constante). O ganho nesse ponto é o ganho crítico Ku e o período de oscilação é Pu.

Controlador	Kp	Ti	Td
P	0,5 · Ku	∞	0

PI	0,45 · Ku	Pu / 1,2	0
PID	0,6 · Ku	Pu / 2	Pu / 8

Tabela 8 — Ziegler-Nichols, método da oscilação sustentada (malha fechada).

Outros métodos:

- IMC (Internal Model Control): Baseado em um modelo interno do processo. Permite ajustar a agressividade via um parâmetro de filtro λ.
- Cohen-Coon: Variante do método de curva de reação, mais adequado para processos com atraso de transporte dominante (L/T > 0,3).
- Tentativa e erro: Começa com Kp baixo, aumenta até resposta adequada; adiciona Ki para eliminar offset; adiciona Kd se necessário para reduzir overshoot.
- Autotuning: Muitos CLPs modernos possuem função de autossintonia, que aplica perturbações controladas e calcula os parâmetros automaticamente.

3.5 Problemas práticos

Windup Integral:

Quando o atuador satura (atinge 0% ou 100%), o erro persiste e o termo integral continua acumulando, atingindo valores enormes. Quando o erro finalmente muda de sinal, demora muito para a integral 'esvaziar', causando overshoot excessivo. Soluções: clamping (limitar o termo integral quando a saída satura), back-calculation (reduzir a integral baseando-se na diferença entre saída desejada e saída real do atuador), ou a forma incremental do PID.

Ruído Derivativo:

A ação derivativa amplifica ruído de alta frequência presente no sinal de medição. Soluções: aplicar um filtro passa-baixa no termo derivativo (Td·s / (1 + Td·s/N), com N entre 3 e 20); derivar a PV em vez do erro (evita picos no degrau do setpoint); ou simplesmente usar controlador PI quando o ruído é excessivo.

Saturação do Atuador:

Válvulas, inversores e aquecedores têm limites físicos (0–100%). O controlador deve ter limites de saída (output clamping) e lógica anti-windup para operar corretamente quando o atuador satura.

3.6 Exemplo prático: controle de temperatura de estufa agrícola

Considere uma estufa agrícola onde a temperatura interna deve ser mantida em 25 °C para o cultivo de mudas. O sistema possui:

- Sensor: PT100 (faixa 0–50 °C, saída 4–20 mA)
- Atuador: Aquecedor elétrico 2 kW + ventilador, controlados por SSR (relé de estado sólido)
- Controlador: CLP com bloco PID, saída PWM (período 2 s)

- Setpoint: 25 °C

Procedimento de sintonia (tentativa e erro):

1. Configurar o PID como P puro com Kp = 2,0. Observar a resposta: a temperatura estabiliza em 23 °C (offset de 2 °C).
2. Adicionar ação integral: Ki = 0,1 (Ti = 20 s). O offset é eliminado, mas há overshoot de 3 °C.
3. Adicionar ação derivativa: Kd = 0,5 (Td = 0,25 s). O overshoot reduz para 0,5 °C.
4. Parâmetros finais: Kp = 2,0; Ki = 0,1; Kd = 0,5 — tempo de acomodação ~120 s.

Em Structured Text (IEC 61131-3), o bloco PID seria configurado assim:

```
PROGRAM ControleTempEstufa
VAR
PV_Temp : REAL; (* Temperatura medida - PT100 *)
SP_Temp : REAL := 25.0; (* Setpoint *)
Out_PWM : REAL; (* Saída 0-100% *)
PID_Inst : PID; (* Instância do bloco PID *)
END_VAR

PID_Inst(
AUTO := TRUE,
PV := PV_Temp,
SP := SP_Temp,
KP := 2.0,
TI := T#20s,
TD := T#250ms,
CYCLE := T#100ms,
LIMIT_H := 100.0,
LIMIT_L := 0.0,
MN := Out_PWM
);
END_PROGRAM
```

Código 1 — Controle PID de estufa em Structured Text.

Capítulo 4 — SCADA e IHM

4.1 O que é SCADA?

SCADA (Supervisory Control and Data Acquisition) é um sistema de software e hardware que permite monitorar e controlar processos industriais remotamente. Enquanto o CLP executa o controle em tempo real no nível de campo, o SCADA opera no nível de supervisão (Nível 3 da pirâmide ISA-95), fornecendo ao operador uma visão panorâmica de todo o processo.

Componentes de um sistema SCADA:

- RTU/PLC (Unidades Remotas): Coletam dados de sensores e executam comandos de controle. Localizados próximos ao processo.
- Rede de Comunicação: Conecta as RTUs ao servidor SCADA. Pode usar Ethernet industrial, fibra óptica, rádio, celular (4G/5G) ou satélite.
- Servidor SCADA: Computador central que agrega dados de todas as RTUs, executa scripts, gera alarmes, armazena histórico e disponibiliza dados para IHMs.
- Estações de Trabalho / IHM: Interfaces gráficas onde operadores visualizam o processo, reconhecem alarmes, alteram setpoints e comandam atuadores.
- Historiador (Historian): Banco de dados otimizado para séries temporais. Armazena milhões de pontos por dia para análise de tendências, relatórios e auditoria.

4.2 Protocolos de comunicação industrial

Protocolo	Meio Físico	Velocidade	Aplicação Típica
Modbus RTU	RS-485 (serial)	9600–115200 bps	CLPs legados, instrumentação, inversores de frequência
Modbus TCP	Ethernet	10/100 Mbps	Integração SCADA-CLP, comunicação entre CLPs
PROFIBUS DP	RS-485 (par trançado)	Até 12 Mbps	Automação de fábrica (Siemens), drives, E/S remotas
PROFINET	Ethernet industrial	100 Mbps – 1 Gbps	Automação Siemens, robôs, acionamentos
EtherNet/IP	Ethernet industrial	100 Mbps – 1 Gbps	Automação Rockwell (Allen-Bradley)
OPC-UA	Ethernet (TCP/IP)	Variável	Integração entre sistemas heterogêneos, IoT, Indústria 4.0

Tabela 9 — Protocolos de comunicação industrial.

OPC-UA (Open Platform Communications — Unified Architecture) merece destaque por ser independente de fabricante, seguro (criptografia e autenticação), e capaz de modelar dados semanticamente. É considerado o protocolo-chave para a Indústria 4.0, permitindo interoperabilidade entre equipamentos de diferentes fornecedores.

4.3 Desenvolvimento de telas IHM — Boas Práticas

O design de telas IHM (Interface Homem-Máquina) impacta diretamente a segurança e a eficiência operacional. A norma ISA-101 (Human Machine Interfaces for Process Automation Systems) e as diretrizes da ASM (Abnormal Situation Management) recomendam:

Hierarquia de telas:

- Nível 1 — Overview: Visão geral de toda a planta ou área. KPIs, estado dos equipamentos, alarmes ativos. O operador deve entender a situação em 2 s.
- Nível 2 — Área/Unidade: Detalhes de uma seção do processo. Valores de PV, SP, saída do controlador, tendências de 1–4 h.
- Nível 3 — Detalhe: Faceplates de malhas de controle individuais, configuração de parâmetros, diagnósticos de equipamentos.

Uso de cores (high-performance HMI):

- Fundo: cinza neutro (não preto, não branco) — reduz fadiga visual
- Equipamentos: cinza médio em estado normal
- Alarme crítico: vermelho (uso exclusivo para segurança)
- Alarme alto: amarelo/âmbar
- Indicação de execução: verde ou azul (em destaque apenas quando necessário)
- Animações: mínimas; piscar só para alarmes não reconhecidos

Tendências (Trends):

Gráficos de tendência em tempo real são essenciais para que o operador identifique desvios antes que se tornem alarmes. Recomenda-se exibir PV, SP e saída do controlador na mesma tela para malhas críticas. Escalas devem ser fixas (não auto-escala) para facilitar a percepção de variações.

4.4 Exemplos de software SCADA

Software	Fabricante	Licença	Características
Ignition	Inductive Automation	Comercial	Baseado em Java, licenciamento ilimitado por servidor, módulos MQTT e Edge
Wonderware (AVEVA)	AVEVA	Comercial	Líder de mercado, InTouch HMI, Historian, ambiente integrado
WinCC	Siemens	Comercial	Integrado ao TIA Portal, ideal para ecossistema Siemens
FactoryTalk View	Rockwell	Comercial	Integrado ao Studio 5000, ecossistema Allen-Bradley
ScadaBR	Comunidade	Open-source	Baseado no Mango M2M, suporta Modbus, OPC, HTTP. Ideal para ensino e pequenas aplicações
OpenSCADA	Comunidade	Open-source	Linux, modular, suporta diversos protocolos

Tabela 10 — Softwares SCADA populares.

4.5 Segurança em sistemas SCADA

A segurança cibernética de sistemas SCADA tornou-se uma preocupação crítica após incidentes como o Stuxnet (2010) e o ataque à estação de tratamento de água na Flórida (2021). Diferentemente da TI tradicional, sistemas de automação priorizam disponibilidade sobre confidencialidade.

Princípios fundamentais:

- Defesa em profundidade: Múltiplas camadas de proteção — firewalls, DMZ industrial, segmentação de rede, autenticação.
- Segmentação de rede: Separar rede corporativa (TI) da rede industrial (TA) usando firewalls e zonas DMZ. Norma IEC 62443.
- Princípio do mínimo privilégio: Cada usuário e sistema tem apenas os acessos estritamente necessários.
- Monitoramento e detecção: IDS (Intrusion Detection System) adaptados para protocolos industriais (Modbus, PROFINET).
- Patches e atualizações: Devem ser testados em ambiente de homologação antes de serem aplicados em produção, devido ao risco de interrupção.
- Backup e recuperação: Backup regular de programas de CLP, configuração SCADA e banco de dados do historiador. Plano de recuperação de desastres (DRP).

Capítulo 5 — Aplicações no Agronegócio e Semiárido

O semiárido brasileiro, que abrange grande parte do Nordeste e norte de Minas Gerais, apresenta desafios específicos para a produção agropecuária: escassez hídrica, altas temperaturas, variabilidade climática e limitações de infraestrutura. A automação oferece soluções eficientes para otimizar o uso de recursos e aumentar a produtividade nesse contexto.

5.1 Automação de irrigação

A irrigação é responsável por cerca de 70% do consumo de água no Brasil. No semiárido, a eficiência hídrica é ainda mais crítica. Sistemas automatizados de irrigação permitem aplicar água na quantidade certa, no momento certo, reduzindo desperdício e melhorando a produtividade.

Componentes típicos de um sistema automatizado:

- Sensores de umidade do solo: Capacitivos (ex: Decagon 10HS), tensiômetros, TDR. Medem o teor de água no solo em diferentes profundidades.
- Estação meteorológica: Temperatura, umidade relativa, radiação solar, vento, pluviômetro. Permite calcular a evapotranspiração (ETo) pelo método Penman-Monteith.
- Válvulas solenoides: Controlam a abertura/fechamento de setores de irrigação. Operação típica em 24 V AC ou 12 V DC (para sistemas solares).
- CLP ou microcontrolador: Executa a lógica de controle. Para sistemas simples, um Arduino/ESP32 pode ser suficiente; para sistemas maiores, CLPs compactos (Schneider M221, WEG TPW-04) são indicados.
- Comunicação: LoRa, Zigbee ou GSM/4G para transmissão de dados do campo para a central de monitoramento.

O algoritmo de controle pode ser baseado em limites fixos de umidade (on/off) ou em modelos de balanço hídrico que consideram a ETo, a precipitação e as características do solo para calcular a lâmina de irrigação necessária.

5.2 Monitoramento de aviários

A avicultura é uma atividade econômica relevante no semiárido. O estresse térmico é o principal fator limitante da produção, podendo causar redução no consumo de ração, queda na postura e, em casos extremos, mortalidade do plantel.

Caso do LabAuto — Monitoramento de aviário em Sumé-PB:

O LabAuto desenvolveu um sistema de monitoramento ambiental para aviários na região de Sumé–PB, utilizando sensores de temperatura e umidade (DHT22/SHT31), sensor de CO_2 (MH-Z19B) e luminosidade (BH1750). Os dados são coletados por um microcontrolador ESP32, transmitidos via Wi-Fi para um servidor local com banco de dados InfluxDB e visualizados em dashboards Grafana.

Variáveis monitoradas e faixas ideais:

Variável	Faixa Ideal	Ação Corretiva
Temperatura	20–26 °C (frangos adultos)	Ventiladores, nebulizadores, cortinas
Umidade relativa	50–70%	Ventilação, nebulização
CO_2	< 3000 ppm	Aumento da ventilação
Amônia (NH_3)	< 20 ppm	Troca de cama, ventilação
Luminosidade	5–20 lux (programa de luz)	Dimmers automáticos

Tabela 11 — Variáveis de conforto térmico em aviários.

5.3 Sistemas de bombeamento solar

O bombeamento solar é uma alternativa viável para regiões sem acesso à rede elétrica ou com custo elevado de energia. Painéis fotovoltaicos alimentam uma bomba d'água submersível (ou de superfície) através de um inversor de frequência (VFD) ou controlador dedicado.

Arquitetura típica:

- Arranjo fotovoltaico (módulos em série/paralelo conforme a tensão/corrente do inversor)
- Inversor/VFD solar com MPPT (Maximum Power Point Tracking)
- Bomba submersível ou centrífuga
- Sensores: nível do reservatório (boia ou ultrassônico), pressão, vazão
- Proteções: funcionamento a seco, sobrecorrente, subtensão

O controlador do sistema deve implementar: partida suave (rampa de frequência), proteção contra funcionamento a seco (desliga a bomba se o nível do poço cair), controle de nível do reservatório (desliga ao atingir nível máximo, liga ao atingir nível mínimo — histerese para evitar liga-desliga excessivo).

5.4 Perspectivas: Agricultura 4.0 no Nordeste brasileiro

A Agricultura 4.0 representa a convergência de tecnologias digitais com a produção agropecuária: IoT, drones, imagens de satélite, inteligência artificial e big data. No contexto do Nordeste e do semiárido, as perspectivas incluem:

• Agricultura de precisão com baixo custo: Sensores IoT de baixo custo (ESP32 + LoRa) para monitoramento distribuído de grandes áreas de cultivo.

• Irrigação inteligente baseada em IA: Modelos de machine learning para predição de demanda hídrica com base em dados meteorológicos e de solo.

• Drones para mapeamento e pulverização: Identificação de pragas, estresse hídrico e falhas de plantio por imagens multiespectrais.

• Energia solar integrada: Agrivoltaicos — painéis solares sobre plantios, gerando energia e sombreamento parcial benéfico para certas culturas.

• Conectividade rural: Redes LoRaWAN e satélite (Starlink) para superar a falta de conectividade em áreas remotas do semiárido.

• Capacitação local: Universidades como a UFCG (campus Sumé) e laboratórios como o LabAuto desempenham papel fundamental na formação de profissionais e no desenvolvimento de soluções adaptadas à realidade regional.

Exercícios Resolvidos

Exercício 1 — Lógica Ladder: Partida Sequencial

Enunciado: Projete um programa Ladder para a partida sequencial de dois motores. O Motor 2 (Q0.1) só deve ligar 5 segundos após o Motor 1 (Q0.0) ser ligado. Um botão de emergência (I0.2, NF) desliga ambos os motores imediatamente.

Solução:

```
Rung 1 — Motor 1 com selo:
| I0.0 I0.1 I0.2 Q0.0 |
|--[ ]--+--[/]----[/]----( )---|
| Q0.0 | |
|--[ ]--+ |

Rung 2 — Temporizador de 5 s:
| Q0.0 TON_1 |
|--[ ]---[IN Q]--- |
| [PT=5s ] |

Rung 3 — Motor 2:
| TON_1.Q I0.1 I0.2 Q0.1 |
|--[ ]--+--[/]---[/]----( )---|
| Q0.1 | |
|--[ ]--+ |
```

O TON_1 começa a contar quando Q0.0 (Motor 1) liga. Após 5 s, TON_1.Q fica TRUE e permite a partida do Motor 2 (Q0.1). O botão de emergência I0.2 (NF) está em série com ambas as bobinas, garantindo parada imediata.

Exercício 2 — Cálculo de Parâmetros PID (Ziegler-Nichols)

Enunciado: Um processo de controle de temperatura apresentou, pelo método da oscilação sustentada, ganho crítico Ku = 4,0 e período de oscilação Pu = 6 s. Calcule os parâmetros Kp, Ti e Td para um controlador PID usando Ziegler-Nichols.

Solução:

Usando a tabela de Ziegler-Nichols (malha fechada) para PID:

```
Kp = 0,6 · Ku = 0,6 × 4,0 = 2,4
Ti = Pu / 2 = 6 / 2 = 3,0 s
Td = Pu / 8 = 6 / 8 = 0,75 s

Convertendo para Ki e Kd:
Ki = Kp / Ti = 2,4 / 3,0 = 0,8
Kd = Kp · Td = 2,4 × 0,75 = 1,8

Parâmetros finais: Kp = 2,4 ; Ki = 0,8 ; Kd = 1,8
```

Nota: Estes valores são um ponto de partida. Ziegler-Nichols tende a gerar respostas com overshoot de ~25%. Em aplicações sensíveis, ajustes finos (redução de Ki, aumento de Kd) são

necessários.

Exercício 3 — Contador de Peças em Ladder

Enunciado: Um sensor fotoelétrico (I0.3) detecta peças em uma esteira. Quando 10 peças forem contadas, a saída Q0.2 (alarme de caixa cheia) deve ser ativada. Um botão de reset (I0.4) zera o contador.

Solução:

```
Rung 1 — Contador crescente:
| I0.3 CTU_1 |
|--[ ]---[CU Q]---( Q0.2 )|
| I0.4 [R ] |
|--[ ]---[ ] |
| [PV = 10 ] |

Quando CV (valor atual) atingir PV (10),
a saída Q do CTU fica TRUE → Q0.2 liga.
Pressionar I0.4 reseta CV para 0.
```

Exercício 4 — Configuração Modbus RTU

Enunciado: Um sistema SCADA precisa ler a temperatura de um transmissor que usa Modbus RTU. O transmissor tem endereço 1, e a temperatura está no registrador Holding Register 40001 (endereço 0 na função 03). A faixa de medição é 0–100 °C, representada por 0–10000 (inteiro). Se a leitura retornar 3250, qual é a temperatura?

Solução:

```
Configuração Modbus RTU:
Endereço do escravo: 1
Função: 03 (Read Holding Registers)
Endereço inicial: 0 (= 40001)
Quantidade: 1 registrador

Conversão:
Leitura bruta = 3250
Faixa bruta: 0 a 10000
Faixa engenharia: 0 a 100 °C

Temperatura = (3250 / 10000) × 100 = 32,50 °C
```

Exercício 5 — Controlador PI e Erro em Regime

Enunciado: Um controlador proporcional com Kp = 5,0 controla a pressão de um tanque. O setpoint é 10 bar e a pressão estabiliza em 9,2 bar. (a) Qual é o offset? (b) Se adicionarmos ação integral com Ki = 0,5, o que acontece com o offset em regime permanente?

Solução:

```
(a) Offset = SP − PV = 10,0 − 9,2 = 0,8 bar

O controlador P puro gera saída u = Kp · e = 5,0 × 0,8 = 4,0
Esta saída de 4,0 é o necessário para manter PV = 9,2 bar.
Para PV = 10,0 bar, o erro seria 0, e a saída seria 0,
o que não sustentaria a pressão. Logo, o offset é inevitável.

(b) Com ação integral (Ki = 0,5), o termo integral acumula
o erro ao longo do tempo:
u_i(t) = Ki · ∫e(τ)dτ

Enquanto e > 0 (PV < SP), a integral cresce, aumentando
a saída do controlador, que por sua vez aumenta a PV.
O processo só entra em equilíbrio quando e = 0 (PV = SP).

Resultado: o offset é ELIMINADO. A pressão converge
para exatamente 10,0 bar em regime permanente.
```

Roteiro de Laboratório — Lab 01

Programação Básica de CLP em Ladder usando o OpenPLC Editor

Objetivo:

Familiarizar o aluno com o ambiente de programação de CLPs utilizando o OpenPLC Editor (software livre, IEC 61131-3). Ao final, o aluno será capaz de criar um programa Ladder com contatos NA/NF, bobinas, temporizadores (TON) e contadores (CTU), e testá-lo no simulador integrado.

Materiais Necessários:

- Computador com OpenPLC Editor instalado (download gratuito em openplcproject.com)
- Alternativamente: CODESYS (versão gratuita) ou PLCfiddle (online)

Procedimento:

Parte 1 — Instalação e Configuração (10 min)

1. Baixe o OpenPLC Editor em https://openplcproject.com/plcopen-editor/
2. Instale seguindo as instruções do sistema operacional (Windows/Linux/macOS)
3. Abra o OpenPLC Editor e crie um novo projeto: File → New → Nome: 'Lab01'
4. Selecione a linguagem Ladder (LD) para o programa principal (POU)

Parte 2 — Programa 1: Partida Direta com Selo (20 min)

1. Declare as variáveis:
— BTN_LIGA: BOOL, endereço %IX0.0 (entrada digital 0)
— BTN_DESLIGA: BOOL, endereço %IX0.1 (entrada digital 1)
— MOTOR: BOOL, endereço %QX0.0 (saída digital 0)
2. No editor Ladder, crie o circuito de partida com selo conforme o Exemplo 1 do Capítulo 2
3. Compile (Build → Build) e verifique que não há erros
4. Execute a simulação e teste: ative BTN_LIGA → MOTOR liga; desative BTN_LIGA → MOTOR permanece ligado (selo); ative BTN_DESLIGA → MOTOR desliga

Parte 3 — Programa 2: Temporização e Contagem (30 min)

1. Crie um novo programa (POU) chamado 'Contador_Pecas'
2. Declare as variáveis: SENSOR (%IX0.2), RESET (%IX0.3), ALARME (%QX0.1), contador CTU_1 (tipo CTU)
3. Implemente a lógica do Exercício 3: contar 10 pulsos do sensor e ativar alarme
4. Adicione um temporizador TON de 3 s entre o alarme e uma segunda saída (SIRENE, %QX0.2): a sirene só deve tocar 3 s após o alarme
5. Compile, simule e teste todos os cenários

Parte 4 — Desafio Extra (opcional, 20 min)

Implemente um semáforo de 2 fases (verde/vermelho) com temporização: Verde = 10 s, Amarelo = 3 s, Vermelho = 10 s. Use SFC (Sequential Function Chart) se seu simulador suportar, ou implemente com Ladder usando comparadores e um timer cíclico.

Questões para o Relatório:

1. Explique o conceito de 'selo' (auto-retenção) em Ladder. Por que é necessário?
2. Qual a diferença entre os temporizadores TON, TOF e TP?
3. O que acontece se o scan time do CLP for maior que a duração de um pulso do sensor? Como resolver esse problema?
4. Cite duas vantagens do OpenPLC em relação a softwares proprietários para uso acadêmico.
5. Descreva uma aplicação real na região de Sumé–PB onde o programa desenvolvido neste laboratório poderia ser utilizado.

Referências

BOLTON, W. Mecatrônica: uma abordagem multidisciplinar. 4. ed. Porto Alegre: Bookman, 2010. 664 p.

CAPELLI, A. Automação industrial: controle do movimento e processos contínuos. 3. ed. São Paulo: Érica, 2013. 288 p.

FRANCHI, C. M.; CAMARGO, V. L. A. Controladores lógicos programáveis: sistemas discretos. 2. ed. São Paulo: Érica, 2014. 352 p.

GROOVER, M. P. Automação industrial e sistemas de manufatura. 3. ed. São Paulo: Pearson, 2011. 581 p.

MORAES, C. C.; CASTRUCCI, P. L. Engenharia de automação industrial. 2. ed. Rio de Janeiro: LTC, 2007. 358 p.

NISE, N. S. Engenharia de sistemas de controle. 7. ed. Rio de Janeiro: LTC, 2017. 772 p.

OGATA, K. Engenharia de controle moderno. 5. ed. São Paulo: Pearson, 2011. 824 p.

PRUDENTE, F. Automação industrial: PLC — teoria e aplicações. 2. ed. Rio de Janeiro: LTC, 2013. 316 p.

THOMAZINI, D.; ALBUQUERQUE, P. U. B. Sensores industriais: fundamentos e aplicações. 9. ed. São Paulo: Érica, 2020. 272 p.

IEC 61131-3. Programmable controllers — Part 3: Programming languages. International Electrotechnical Commission, 2013.