

INSTITUTO FEDERAL DE SANTA CATARINA
ENGENHARIA ELÉTRICA

PEDRO HENRIQUE MIRA DE MEDEIROS

**UTILIZAÇÃO DE UM MÉTODO FORMAL SISTEMATIZADO PARA A
PROGRAMAÇÃO DE CLPS EM TEXTO ESTRUTURADO (SCL)**

Joinville

2025

PEDRO HENRIQUE MIRA DE MEDEIROS

**UTILIZAÇÃO DE UM MÉTODO FORMAL SISTEMATIZADO PARA A
PROGRAMAÇÃO DE CLPS EM TEXTO ESTRUTURADO (SCL)**

Trabalho de conclusão de curso
apresentado ao Curso de Engenharia Elétrica
do Instituto Federal de Santa Catarina, Campus
Joinville, como requisito parcial para obtenção
do título de Engenheiro Eletricista.

Orientador: Jeferson Luiz Curzel

Joinville

2025

Medeiros, Pedro.

Utilização de um método formal sistematizado para a programação de CLPs em texto estruturado (SCL) / Pedro Henrique Mira de Medeiros. – Joinville, SC, 2025.

103 p.

Trabalho de Conclusão de Curso (Graduação) - Instituto Federal de Educação Ciência e Tecnologia de Santa Catarina, Curso de Bacharelado em Engenharia Elétrica, Joinville, 2025.

Orientador: Jeferson Luiz Curzel.

1. CLP. 2. Texto estruturado. 3. Método passo a passo. I. Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina. II. Título.

PEDRO HENRIQUE MIRA DE MEDEIROS

UTILIZAÇÃO DE UM MÉTODO FORMAL SISTEMATIZADO PARA A
PROGRAMAÇÃO DE CLPS EM TEXTO ESTRUTURADO (SCL)

Este trabalho foi julgado adequado para obtenção do título em
Engenharia Elétrica, pelo Instituto Federal de Educação, Ciência e Tecnologia
de Santa Catarina, e aprovado na sua forma final pela comissão avaliadora
abaixo indicada.

Joinville, 10 de dezembro de 2025.



Prof. Dr. Jeferson Luiz Curzel, Orientador

Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina



Prof. MSc. Stefano Romeu Zeplin

Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina



Prof. Dr. Michael Klug

Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina

AGRADECIMENTOS

Agradeço primeiramente a Deus, pela saúde, força e sabedoria concedidas ao longo desta jornada acadêmica. Aos meus familiares, pelo apoio incondicional, incentivo e compreensão durante todo o período de desenvolvimento deste trabalho.

Expresso minha sincera gratidão ao meu orientador, Jeferson Luiz Curzel, pela dedicação, orientação técnica, paciência e confiança depositada em mim ao longo deste projeto.

Agradeço também aos professores e colaboradores do curso, que contribuíram direta ou indiretamente para minha formação. Aos colegas e amigos, pela troca de conhecimento, pelas discussões construtivas e pelo apoio mútuo durante a graduação.

Por fim, agradeço ao IFSC pelo suporte e pelos recursos disponibilizados para a realização deste Trabalho de Conclusão de Curso.

RESUMO

Este trabalho propõe a aplicação de um método formal e sistematizado para a programação de Controladores Lógicos Programáveis (CLPs) utilizando a linguagem Texto Estruturado (SCL – Structured Control Language). Tradicionalmente, a programação de CLPs é ensinada com foco na linguagem Ladder, o que pode limitar a adoção de abordagens mais modernas e eficientes. Para enfrentar esse desafio, este estudo aplica um método baseado em cinco etapas formais sequenciais: (1) análise do processo automatizado, (2) modelagem lógica do controle utilizando o método passo a passo, (3) tradução para código estruturado em SCL, (4) validação e testes nas bancadas da festo e (5) desenvolvimento de um material didático sobre a programação em SCL. Essa abordagem visa promover clareza, padronização e reaproveitamento de código, tornando o desenvolvimento de sistemas de automação mais acessível e robusto. O método foi implementado nas bancadas didáticas da Festo utilizando o CLP Siemens S7-1214C e o ambiente TIA Portal. Foram desenvolvidos exemplos práticos para demonstrar a eficácia do procedimento, os quais comprovaram a facilidade de implementação e a melhoria na organização e manutenção do código. O trabalho contribui como um guia para estudantes e técnicos que desejam adotar práticas modernas de programação estruturada com maior confiabilidade.

Palavras-chave: CLP. Texto Estruturado. SCL. Automação Industrial. TIA Portal. Método Passo a Passo. Programação Sistemática.

ABSTRACT

This paper proposes the application of a formal and systematic method for programming Programmable Logic Controllers (PLCs) using Structured Control Language (SCL). Traditionally, PLC programming is taught with a focus on Ladder logic, which may limit the adoption of more modern and efficient approaches. To address this challenge, this study implements a method based on five sequential formal steps: (1) analysis of the automated process, (2) logical modeling of control using the step-by-step method, (3) translation into structured SCL code, (4) validation and testing on Festo workbenches, and (5) development of educational material on SCL programming. This approach aims to promote clarity, standardization, and code reuse, making the development of automation systems more accessible and robust. The method was implemented on a Festo didactic workbench using the Siemens S7-1214C PLC and the TIA Portal environment. Practical examples were developed to demonstrate the effectiveness of the procedure, which confirmed the ease of implementation and improvements in code organization and maintenance. This work serves as a guide for students and technicians who wish to adopt modern structured programming practices with greater reliability.

Keywords: PLC, Structured Control Language (SCL), Industrial Automation, TIA Portal, Step-by-Step Method, Systematic Programming.

SUMÁRIO

1 INTRODUÇÃO.....	16
2 REVISÃO BIBLIOGRÁFICA.....	19
2.1 HARDWARE CLP	19
2.2 CICLO DE SCAN	21
2.3 UNIDADE DE ENTRADA E SAÍDA	23
2.4 LINGUAGEM SCL X LADDER.....	24
2.4.1 CARACTERÍSTICAS DO LADDER.....	25
2.4.2 CARACTERÍSTICAS DO SCL	27
2.4.3 COMPARAÇÃO LADDER X SCL.....	29
2.5 MÉTODO PASSO A PASSO	30
2.6 TEMPORIZADORES.....	34
2.6.1 TEMPORIZADOR TON	34
2.6.2 TEMPORIZADOR TOF	36
2.6.3 TEMPORIZADOR TP	37
2.6.4 TEMPORIZADOR TONR	38
2.7 CONTADORES.....	40
2.7.1 CONTADOR CTU	40
2.7.2 CONTADOR CTD	41

2.7.3 CONTADOR CTUD	42
2.8 CONCLUSÕES DO CAPÍTULO	44
3 CONCEITOS BÁSICOS DO TIA PORTAL	45
3.1 SOFTWARE	45
3.2 ESTRUTURA DE BLOCOS	45
3.3 CRIAÇÃO DO PROJETO E CONFIGURAÇÃO DO CLP	46
3.4 CRIAÇÃO DE BLOCOS OB	50
3.5 DEFINIÇÃO DE IO'S NO TIA PORTAL.....	52
3.6 CONCLUSÕES DO CAPÍTULO	53
4 APLICAÇÃO DO MÉTODO	54
4.1 BANCADA DE DISTRIBUIÇÃO	55
4.1.1 IO'S BANCADA DE DISTRIBUIÇÃO	56
4.1.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA	56
4.1.3 APLICAÇÃO DO MÉTODO PASSO A PASSO	57
4.1.4 IMPLEMENTAÇÃO NO TIA PORTAL	60
4.2 BANCADA DE TESTE	66
4.2.1 IO'S BANCADA DE TESTE.....	67
4.2.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA	67
4.2.3 APLICAÇÃO DO MÉTODO PASSO A PASSO	68
4.2.4 IMPLEMENTAÇÃO NO TIA PORTAL	71
4.3 BANCADA DE PROCESSAMENTO	73

4.3.1 IO'S BANCADA DE PROCESSAMENTO	74
4.3.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA	75
4.3.3 APLICAÇÃO DO MÉTODO PASSO A PASSO	75
4.3.4 IMPLEMENTAÇÃO NO TIA PORTAL	78
4.4 BANCADA DE MANIPULAÇÃO.....	82
4.4.1 IO'S BANCADA DE MANIPULAÇÃO	83
4.4.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA	84
4.4.3 APLICAÇÃO DO MÉTODO PASSO A PASSO	85
4.4.4 IMPLEMENTAÇÃO NO TIA PORTAL	87
4.5 BANCADA DE SEPARAÇÃO	90
4.5.1 IO'S BANCADA DE SEPARAÇÃO.....	91
4.5.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA	92
4.5.3 APLICAÇÃO DO MÉTODO PASSO A PASSO	93
4.5.4 IMPLEMENTAÇÃO NO TIA PORTAL	96
4.6 CONCLUSÕES DO CAPÍTULO	99
5 CONCLUSÃO	100
6 REFERENCIAL TEÓRICO	102

LISTA DE FIGURAS

Figura 1 - Ciclo Scan (BARROS, 2017)	22
Figura 2 - Imagem de Processo de Entrada (PII) e Imagem de Processo de Saída (PIQ)	23
Figura 3 - Entradas Digitais PLC (PRUDENTE, 2011).....	24
<i>Figura 4 – Projeto Exemplo Acionamento de Motor (Autor, 2025)</i>	<i>26</i>
<i>Figura 5 – Exemplo Ladder Acionamento de Motor (Autor, 2025)</i>	<i>27</i>
<i>Figura 6 – Exemplo SCL Acionamento de Motor (Autor, 2025).....</i>	<i>28</i>
<i>Figura 7 - Sequência de Condições (MANFRIN, 2021)</i>	<i>30</i>
Figura 8 - Método Passo a Passo em Ladder (VIEIRA, 2005)	32
Figura 9 - Método Passo a Passo em SCL – Fonte: Autor	33
<i>Figura 10 – Temporizador TON (Autor, 2025).....</i>	<i>34</i>
<i>Figura 11 – Adicionando o TON (Autor, 2025)</i>	<i>35</i>
<i>Figura 12 – Nomeando o TON (Autor, 2025)</i>	<i>35</i>
<i>Figura 13 – Estrutura TON (Autor, 2025).....</i>	<i>36</i>
<i>Figura 14 – Lógica TOF (Autor, 2025)</i>	<i>36</i>
<i>Figura 15 – Estrutura SCL TOF (Autor, 2025)</i>	<i>37</i>
<i>Figura 16 – Lógica TP (Autor, 2025).....</i>	<i>37</i>
<i>Figura 17 – Estrutura SCL TP (Autor, 2025).....</i>	<i>37</i>
<i>Figura 18 – Lógica TONR (Autor, 2025)</i>	<i>38</i>
<i>Figura 19 – Estrutura SCL TONR (Autor, 2025)</i>	<i>39</i>

<i>Figura 20 – Lógica CTU (Autor, 2025)</i>	40
<i>Figura 21 – Localização do CTU no software (Autor, 2025)</i>	41
<i>Figura 22 – Estrutura SCL CTU (Autor, 2025)</i>	41
<i>Figura 23 – Lógica CTD (Autor, 2025)</i>	42
<i>Figura 24 – Estrutura SCL CTD (Autor, 2025)</i>	42
<i>Figura 25 – Lógica CTUD (Autor, 2025)</i>	43
<i>Figura 26 – Estrutura SCL CTUD (Autor, 2025)</i>	43
<i>Figura 27 – Criação de Projeto (Autor, 2025)</i>	46
<i>Figura 28 – Criação de Projeto (Autor, 2025)</i>	46
<i>Figura 29 – Criação de Projeto (Autor, 2025)</i>	47
<i>Figura 30 – Adicionando o CLP (Autor, 2025)</i>	48
<i>Figura 31 – Adicionando o CLP (Autor, 2025)</i>	48
<i>Figura 32 – Adicionando Cartões de Expansão (Autor, 2025)</i>	49
<i>Figura 33 – Configuração CLP (Autor, 2025)</i>	50
<i>Figura 34 – Adicionando Blocos (Autor, 2025)</i>	51
<i>Figura 35 – Adicionando Blocos (Autor, 2025)</i>	51
<i>Figura 36 – Definição IO's Tia Portal (Autor, 2025)</i>	52
<i>Figura 37 – Bancadas Festo (Autor, 2025)</i>	54
<i>Figura 38 – Bancada de Distribuição (Festo, 2025)</i>	55
<i>Figura 39 – IO's na Bancada Física (Autor, 2025)</i>	57
<i>Figura 40 – Passo a Passo Bancada de Distribuição (Autor, 2025)</i>	59

Figura 41 – Definição de IO's CLP (Autor, 2025)	60
Figura 42 – Definição de Memórias dos Passos (Autor, 2025)	61
Figura 43 – Habilita Memórias do Sistema (Autor, 2025).....	61
Figura 44 – Blocos de organização (Autor, 2025).....	62
Figura 45 – Código SCL Passo a Passo Bancada de Distribuição (Autor, 2025).....	63
Figura 46 – Código SCL Passo a Passo Bancada de Distribuição (Autor, 2025).....	64
Figura 47 – Método Passo a Passo Aplicado (Autor, 2025)	65
Figura 48 – Código SCL Acionamentos Bancada de Distribuição (Autor, 2025)	65
Figura 49 – Bancada de Teste (Festo, 2025)	66
Figura 50 – IO's na Bancada Física (Autor, 2025).....	68
Figura 51 – Passo a Passo Bancada de Teste (Autor, 2025)	70
Figura 52 – Código SCL Passo a Passo Bancada de Teste (Autor, 2025).....	71
Figura 53 – Código SCL Passo a Passo Bancada de Teste (Autor, 2025).....	71
Figura 54 – Código SCL Acionamentos Bancada de Distribuição (Autor, 2025)	72
Figura 55 – Bancada de Processamento (Festo, 2025).....	73
Figura 56 – IO's Bancada Processamento (Autor, 2025).....	75
Figura 57 – Método Passo a Passo Bancada de Processamento (Autor, 2025)	77
Figura 58 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)	78
Figura 59 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)	79
Figura 60 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)	79
Figura 61 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)	80

Figura 62 – Código SCL Acionamentos Bancada de Processamento (Autor, 2025)	81
Figura 63 – Código SCL Acionamentos Bancada de Processamento (Autor, 2025)	81
Figura 64 – Bancada de Manipulação (Festo, 2025)	82
Figura 65 – IO's Bancada Manipulação (Autor, 2025)	84
Figura 66 – IO's Bancada Manipulação (Autor, 2025)	84
Figura 67 – Método Passo a Passo Bancada de Manipulação (Autor, 2025)	86
Figura 68 – Código SCL Passo a Passo Bancada de Manipulação (Autor, 2025)	87
Figura 69 – Código SCL Passo a Passo Bancada de Manipulação (Autor, 2025)	88
Figura 70 – Código SCL Acionamentos Bancada de Manipulação (Autor, 2025)	89
<i>Figura 71 – Bancada de Separação (Festo, 2025)</i>	<i>90</i>
Figura 72 – IO's na Bancada Física (Autor, 2025)	92
Figura 73 – IO's na Bancada Física (Autor, 2025)	93
Figura 74 – Passo a passo bancada de separação (Autor, 2025)	95
Figura 75 – Código SCL Passo a Passo (Autor, 2025)	96
Figura 76 – Código SCL Passo a Passo (Autor, 2025)	97
Figura 77 – Código SCL Acionamentos (Autor, 2025)	98

LISTA DE TABELAS

Tabela 1 - IO's Bancada Distribuição (Autor, 2025).....	56
Tabela 2 - IO's Bancada de Teste (Autor, 2025).....	67
Tabela 3 - IO's Bancada Processamento (Autor, 2025)	74
Tabela 4 - IO's Bancada Manipulação (Autor, 2025)	83
Tabela 5 - IO's Bancada Separação (Autor, 2025)	91

1 INTRODUÇÃO

A automação industrial tem se consolidado como uma ferramenta essencial para o aprimoramento da produção e aumento da eficiência em diversos setores industriais. O Controlador Lógico Programável (CLP) é uma das principais tecnologias utilizadas para o controle de processos automatizados. As linguagens de programação aplicadas aos CLPs evoluíram ao longo dos anos, oferecendo diferentes soluções conforme as necessidades dos sistemas industriais. Dentre essas linguagens, o Texto Estruturado (SCL) tem ganhado destaque, principalmente devido à sua capacidade de lidar com lógica complexa e suas vantagens na estruturação do código, o que facilita a manutenção e expansão dos sistemas (PRUDENTE, 2011).

A norma IEC 61131-3, que define as linguagens de programação para CLPs, introduziu o Texto Estruturado como uma das opções mais avançadas, destacando-se em aplicações que exigem maior complexidade algorítmica e clareza (IEC, 2013). Diferente de linguagens gráficas, como o Ladder, o SCL oferece uma abordagem textual que pode ser mais próxima da programação de computadores tradicionais, o que permite uma maior flexibilidade e controle sobre a lógica do processo automatizado. De acordo com Prudente (2011), "o uso da linguagem Texto Estruturado permite maior organização e clareza em processos de automação com lógica complexa", ressaltando a eficiência dessa linguagem em ambientes industriais.

A proposta deste trabalho é apresentar e aplicar um método formal sistematizado para a programação de CLPs utilizando o Texto Estruturado (SCL), com o objetivo de proporcionar uma abordagem mais eficiente e confiável na automação industrial.

O método a ser utilizado neste estudo segue uma sequência lógica de etapas, incluindo a análise do processo a ser automatizado, a modelagem do controle, a codificação em SCL e a validação do sistema, sempre com o foco na redução de erros e facilidade de manutenção do código, conforme sugerido por Petruzella (2010), que afirma: "A linguagem ladder continua sendo a mais utilizada, porém, para aplicações que exigem algoritmos matemáticos ou estruturas condicionais mais sofisticadas, o uso de linguagens textuais se torna essencial para manter a legibilidade e eficiência do programa."

A metodologia proposta visa, assim, não apenas explorar o potencial do SCL, mas também proporcionar uma solução sistemática que seja amplamente aplicável em sistemas industriais de qualquer porte. A formalização da programação com o uso de métodos passo a passo contribui para um controle mais preciso sobre os processos automatizados, conforme as tendências da automação moderna, em que a segurança, a eficiência e a confiabilidade dos sistemas são elementos-chave (JACK, 2011).

Este trabalho se justifica pela crescente necessidade de sistemas industriais mais robustos, adaptáveis e fáceis de manter. A proposta visa atender a essas necessidades, fornecendo uma metodologia clara e prática para a programação de CLPs em SCL, baseada em boas práticas aplicadas à automação industrial.

Diante do contexto apresentado, torna-se evidente a necessidade de uma abordagem metodológica estruturada para a programação de Controladores Lógicos Programáveis utilizando a linguagem Texto Estruturado (SCL). Nesse sentido, com o propósito de atingir o objetivo geral deste trabalho, foram definidos objetivos específicos que orientam o desenvolvimento das etapas metodológicas, desde a análise do processo até a validação prática da solução proposta.

Objetivos específicos:

- Realizar o levantamento das entradas e saídas de cada bancada.
- Modelar o processo por meio da aplicação do método passo a passo.
- Converter o modelo sequencial em um programa estruturado utilizando a linguagem Texto Estruturado (SCL).
- Demonstrar a aplicação prática do método nas bancadas FESTO.
- Produzir um tutorial didático que apresente o método e sua implementação em SCL de forma acessível.

2 REVISÃO BIBLIOGRÁFICA

2.1 HARDWARE CLP

O Controlador Lógico Programável (CLP) é composto por uma série de elementos de hardware responsáveis pela execução de tarefas de controle em sistemas de automação industrial. Conforme explicado por Prudente (2011), o CLP é projetado para suportar ambientes industriais severos, oferecendo robustez, confiabilidade e modularidade. Seu hardware é estruturado de forma a possibilitar uma fácil manutenção, ampliação e integração com os demais componentes do sistema automatizado.

A arquitetura básica de um CLP inclui a unidade central de processamento (CPU), fontes de alimentação, módulos de entrada e saída (E/S), memória, interfaces de comunicação e, em muitos casos, módulos especiais. A CPU é o cérebro do sistema, responsável pelo processamento lógico das instruções, leitura das entradas, execução do programa de controle e atualização das saídas. De acordo com Prudente (2011), “a CPU é composta por um microprocessador, uma memória de trabalho e uma unidade de interface que gerencia a comunicação com os módulos de entrada e saída”.

A memória do CLP pode ser subdividida em três tipos principais: a memória do sistema (ROM), a memória de programa (geralmente EEPROM ou flash) e a memória de dados (RAM). Cada uma dessas memórias tem funções específicas, como o armazenamento permanente do sistema operacional, o armazenamento do programa do usuário e o gerenciamento das variáveis durante a execução do programa. Prudente (2011) destaca a importância da separação entre esses tipos de memória para garantir que falhas elétricas ou desligamentos não comprometam o funcionamento do sistema.

Os módulos de entrada e saída (E/S) são os responsáveis pela comunicação entre o CLP e o ambiente externo. As entradas recebem sinais de sensores, botões, chaves e outros dispositivos de detecção, enquanto as saídas comandam atuadores como motores, válvulas, lâmpadas e relés. Prudente (2011) explica que “os módulos de E/S podem ser digitais ou analógicos,

dependendo do tipo de sinal que devem processar, e podem ser fixos ou expansíveis, conforme a arquitetura do CLP”.

Outro componente relevante é a fonte de alimentação, que converte a energia elétrica da rede em níveis adequados para o funcionamento interno do CLP. Essa fonte precisa ser estável e protegida contra surtos, pois a falha nesse componente compromete todo o sistema. Além disso, os módulos de comunicação possibilitam a integração do CLP com outros dispositivos ou redes industriais, como Modbus, Profibus ou Ethernet/IP, permitindo o controle distribuído e a supervisão remota.

A modularidade dos CLPs modernos é um fator que favorece a manutenção e a escalabilidade dos sistemas. Conforme Prudente (2011), “os CLPs modulares permitem a substituição ou adição de módulos sem a necessidade de alterar o controlador principal, o que reduz custos e facilita futuras expansões do sistema”.

2.2 CICLO DE SCAN

Além dos componentes físicos do CLP, o funcionamento interno da CPU obedece a uma lógica de execução cíclica denominada ciclo de scan. Este ciclo é composto por quatro etapas principais: leitura das entradas, execução do programa, atualização das saídas e operações de comunicação e diagnósticos. Segundo Prudente (2011), “a cada leitura das entradas os dados são armazenados em um registro particular da memória, chamado imagem do processo das entradas”, e algo semelhante ocorre com as saídas, conforme visto na figura 1.

O tempo necessário para completar esse ciclo recebe o nome de tempo de scan, e sua duração afeta diretamente o tempo de resposta do CLP. Um tempo de scan excessivamente longo pode fazer com que o sistema não detecte rapidamente mudanças nas entradas, o que compromete a operação da máquina ou do processo automatizado. Como alerta Prudente (2011), se uma entrada muda de estado rapidamente durante o ciclo e volta ao estado anterior antes da próxima leitura, o CLP pode não perceber essa variação — gerando um erro de operação.

Por isso, a escolha do CLP deve considerar não apenas o número de entradas e saídas ou a capacidade de memória, mas também o desempenho da CPU em relação ao tempo de scan. Ainda segundo o autor, muitos CLPs possuem instruções específicas para verificar o tempo de varredura e contêm mecanismos de segurança como o watchdog timer, que interrompe o funcionamento do sistema caso o tempo de scan ultrapasse um valor limite, geralmente na faixa de 150 ms (PRUDENTE, 2011).

Com isso, evidencia-se a importância de um hardware de alto desempenho, não só para suportar cargas de processamento intensas, mas também para garantir a confiabilidade e a segurança do sistema industrial automatizado.

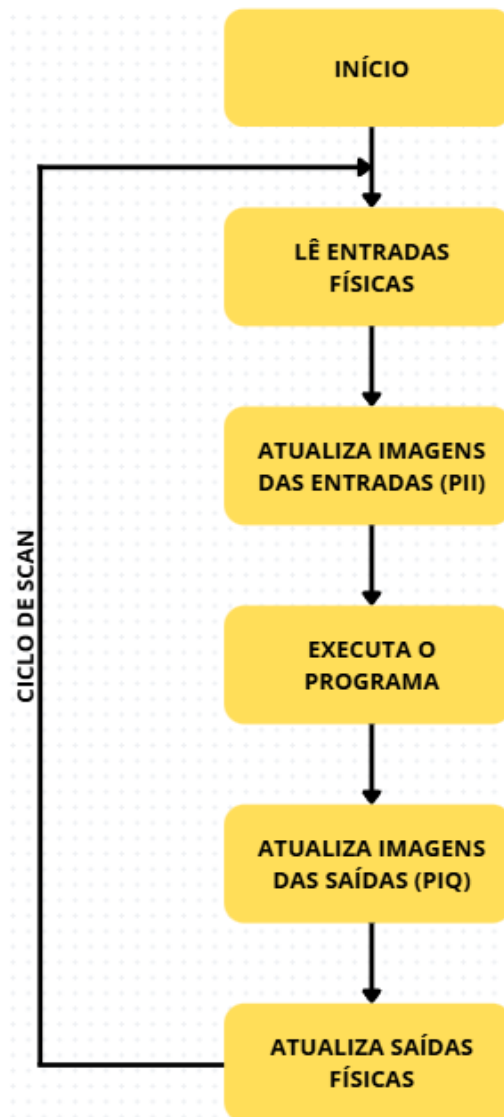


Figura 1 - Ciclo Scan (BARROS, 2017)

Conforme ilustrado na figura acima, que mostra o ciclo de scan, a Figura 2 apresenta a PII (Process Image of Inputs) e a PIQ (Process Image of Outputs), que representam a atualização do mapa de memória das entradas e saídas.

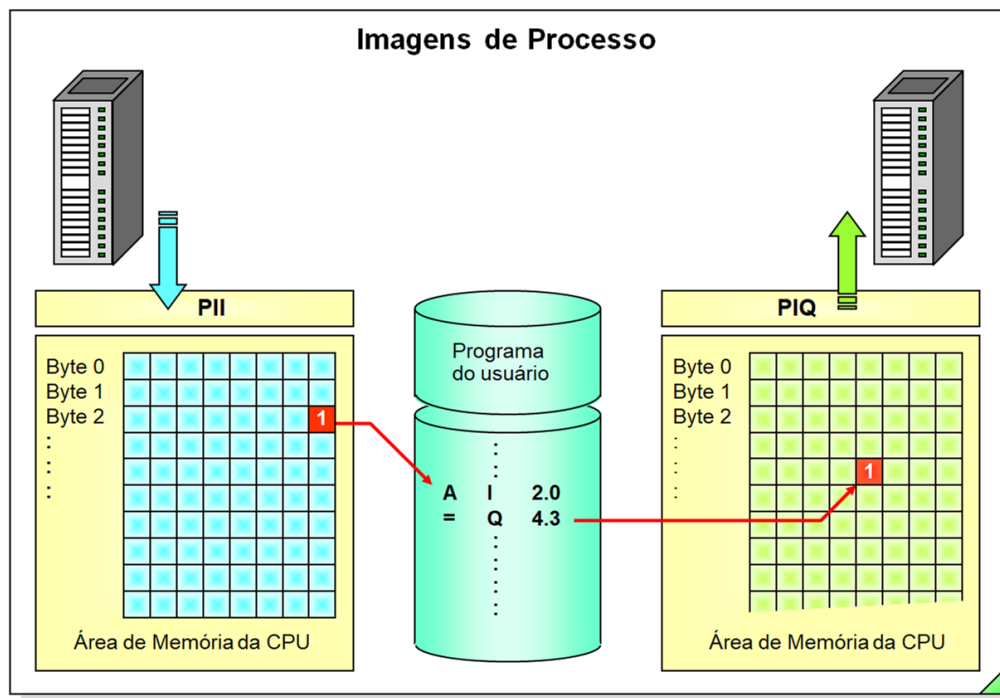


Figura 2 - Imagem de Processo de Entrada (PII) e Imagem de Processo de Saída (PIQ)

(POMARI, 2018)

2.3 UNIDADE DE ENTRADA E SAÍDA

As unidades de entrada e saída (I/O) dos CLPs são componentes essenciais para o diálogo entre o controlador e o ambiente externo. São essas unidades que permitem ao CLP receber sinais provenientes dos sensores e enviar comandos para atuadores. A unidade de entrada atua convertendo os sinais recebidos do campo — como de botoeiras, sensores ou chaves de fim de curso — em sinais compatíveis com a CPU. Por outro lado, a unidade de saída converte os sinais lógicos gerados pelo programa em comandos acionadores para dispositivos físicos, como motores, válvulas, relés e lâmpadas (PRUDENTE, 2011).

Essas interfaces são conhecidas como placas I/O e podem trabalhar com dois tipos principais de sinal: digitais e analógicos. Seguindo a figura 3, os sinais digitais assumem apenas dois estados — “ligado” (1) ou “desligado” (0) — e sua identificação é baseada na presença ou ausência de tensão elétrica, de acordo com a lógica positiva adotada pelo sistema. Nesse contexto, o CLP não interpreta diretamente o estado físico de um contato, como uma chave aberta ou

fechada, mas sim o nível de tensão presente no terminal da entrada (PRUDENTE, 2011).

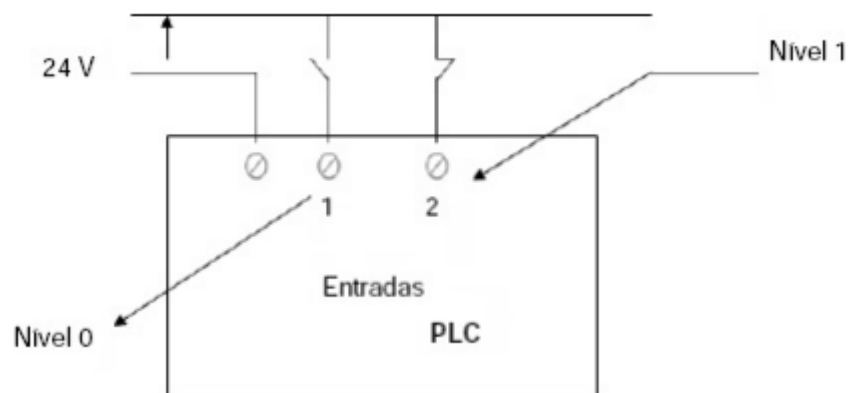


Figura 3 - Entradas Digitais PLC (PRUDENTE, 2011)

Já os sinais analógicos representam valores contínuos dentro de um determinado intervalo (range), como por exemplo temperaturas, pressões ou níveis de tanques. O processamento desses sinais exige maior complexidade e precisão, o que requer uso de memórias de 16 ou 32 bits para garantir fidelidade na leitura e controle. Prudente (2011) destaca que o tratamento desses sinais é mais trabalhoso, especialmente quando há necessidade de precisão elevada.

Dessa forma, compreender a natureza dos sinais de entrada e saída e a forma como eles são tratados internamente no CLP é fundamental para o projeto eficiente de sistemas automatizados, especialmente em ambientes industriais onde precisão e confiabilidade são fatores críticos.

2.4 LINGUAGEM SCL X LADER

A programação de Controladores Lógicos Programáveis (CLPs) utiliza diversas linguagens, com destaque para o Ladder e o SCL, ambas descritas pela norma IEC 61131-3. Cada uma dessas linguagens apresenta características que as tornam mais adequadas para determinados tipos de aplicação. A linguagem Ladder é tradicionalmente utilizada em ambientes industriais devido à sua similaridade com diagramas elétricos e sua facilidade de compreensão para eletricitistas. Por outro lado, a SCL (ou Texto Estruturado) é uma linguagem de

programação textual, mais adequada para operações matemáticas e algoritmos complexos.

2.4.1 CARACTERÍSTICAS DO LADDER

A linguagem Ladder, conforme descrito por Prudente (2019), é uma linguagem gráfica baseada em relés elétricos. Sua estrutura lembra um diagrama de circuitos, com contatos e bobinas, facilitando a interpretação e programação por profissionais da área de eletrônica e elétrica. A programação em Ladder é intuitiva, onde os elementos de controle (como botões, temporizadores e relés) são conectados para definir as condições de operação do sistema.

Prudente (2011) afirma que, devido à sua simplicidade e facilidade de leitura, o Ladder é amplamente utilizado em processos industriais que exigem pouca complexidade e onde a manutenção do sistema envolve principalmente a interpretação de diagramas elétricos.

Na Figura 4, é apresentado um esquema elétrico típico para o acionamento de um motor trifásico. No circuito de comando, observam-se as entradas digitais, compostas por um botão de emergência (E-STOP), um botão de parada e um botão de partida, responsáveis pela lógica de habilitação do sistema. Já no circuito de potência, a saída digital do CLP atua sobre a contatora de acionamento, permitindo ou interrompendo a energização do motor.

Ao lado, é exibido o projeto elétrico correspondente, no qual são destacados o disjuntor de proteção, responsável por salvaguardar o circuito contra curtos-circuitos e sobrecargas, e a contatora, que realiza o chaveamento do motor trifásico. Juntos, esses componentes formam a estrutura básica necessária para o controle seguro e eficiente do acionamento do motor.

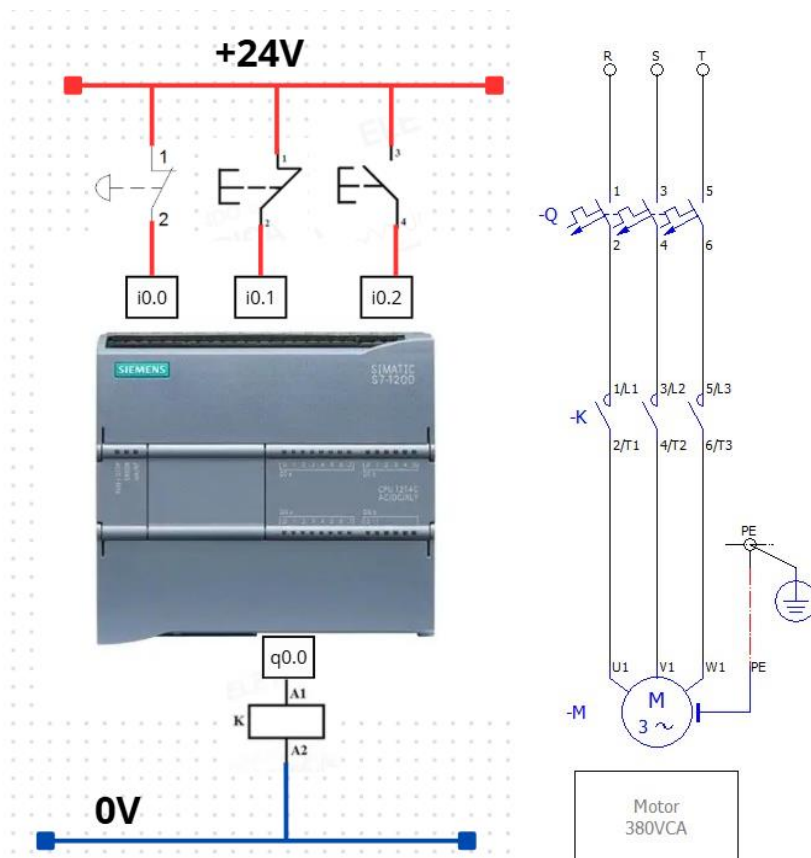


Figura 4 – Projeto Exemplo Acionamento de Motor (Autor, 2025)

Na Figura 5, é apresentada a programação em Ladder correspondente ao sistema de acionamento do motor. Nessa lógica, os botões de emergência e de parada são configurados com lógica invertida, uma vez que ambos possuem contato normalmente fechado (NF). Dessa forma, a condição de funcionamento normal é representada por sinal lógico ativo, e a abertura de qualquer um desses botões interrompe o circuito.

Quando o botão de partida, que possui contato normalmente aberto (NA), é acionado, e desde que as condições de segurança (emergência e parada) estejam satisfeitas, a lógica ativa a saída digital do CLP, energizando a bobina da contatora responsável pelo acionamento do motor.

O reset da bobina ocorre automaticamente sempre que o botão de emergência ou o botão de parada for pressionado, abrindo o circuito e desenergizando a contatora, garantindo a interrupção imediata do acionamento do motor.



Figura 5 – Exemplo Ladder Acionamento de Motor (Autor, 2025)

2.4.2 CARACTERÍSTICAS DO SCL

Por outro lado, a linguagem SCL é uma linguagem textual, estruturada e de alto nível, permitindo a criação de programas mais complexos e eficientes. Conforme Prudente (2011), o SCL permite o uso de estruturas de controle como if-then-else, loops e funções matemáticas avançadas, que são essenciais para o desenvolvimento de algoritmos e controle de processos mais sofisticados. A sintaxe da linguagem SCL é mais próxima de linguagens de programação tradicionais, como Pascal ou C, e permite o desenvolvimento de programas modulares e reutilizáveis.

A principal vantagem do SCL em comparação com o Ladder é sua capacidade de lidar com operações matemáticas mais complexas e maior flexibilidade no design de algoritmos de controle. Prudente (2011) destaca que, enquanto o Ladder é mais intuitivo para simples operações de controle, o SCL é mais indicado para processos que exigem cálculos avançados, manipulação de dados e controle de sistemas dinâmicos.

Na Figura 6, observa-se a programação em SCL referente ao mesmo acionamento de motor apresentado anteriormente, destacando de forma objetiva as diferenças entre a implementação em Ladder e em Texto Estruturado (SCL).

Nessa lógica, o motor será acionado quando todas as condições de operação estiverem satisfeitas: o botão de emergência e o botão de parada (ambos normalmente fechados) estiverem em estado ativo, e o botão de partida for acionado. Caso o botão de emergência ou o botão de parada seja pressionado, o sistema imediatamente desativa a variável de comando, desligando o motor.

O operador lógico “NOT”, presente no código, representa justamente a inversão da condição. Ou seja, quando a variável associada ao botão estiver igual a zero, condição que ocorre ao pressionar o botão de emergência ou o botão de parada, o sistema interpreta essa situação como falha/remoção de segurança, interrompendo o acionamento do motor.

```
1 IF "i_Emergencia" AND "i_Desliga" AND "i_Liga" THEN
2     "q_Motor" := 1;
3 END_IF;
4
5 IF NOT "i_Emergencia" OR NOT "i_Desliga" THEN
6     "q_Motor" := 0;
7 END_IF;
```

Figura 6 – Exemplo SCL Acionamento de Motor (Autor, 2025)

2.4.3 COMPARAÇÃO LADDER X SCL

A comparação entre as duas linguagens pode ser feita sob diversas perspectivas:

- **Facilidade de Uso:** O Ladder é mais fácil de entender e utilizar, especialmente por profissionais que têm um certo conhecimento em elétrica e eletrônica, devido à sua semelhança com esquemas elétricos. Já a programação em SCL exige maior familiaridade com linguagens de programação textuais e estruturas de controle mais complexas (PRUDENTE, 2011).
- **Complexidade dos Algoritmos:** O Ladder é mais limitado em relação à manipulação de dados e execução de cálculos complexos, enquanto o SCL permite a implementação de algoritmos mais sofisticados, incluindo operações matemáticas, lógicas e controle avançado de sistemas (PRUDENTE, 2011).
- **Modularidade e Reutilização de Código:** Em SCL, a criação de funções e blocos de código modulares é muito mais simples e eficiente, enquanto no Ladder a reutilização de código pode ser mais trabalhosa, exigindo duplicação de funções em vários pontos do programa (PRUDENTE, 2011).
- **Aplicações:** O Ladder é ideal para sistemas de controle simples, como controle de dispositivos, acionamento de motores e sistemas de alarme, enquanto o SCL é mais indicado para processos industriais que exigem controle mais rigoroso e calculado, como controle de temperatura, pressão ou qualquer aplicação que envolva cálculos em tempo real (PRUDENTE, 2011).

2.5 MÉTODO PASSO A PASSO

A modelagem de sistemas sequenciais em automação industrial requer métodos que garantam clareza, estrutura lógica e confiabilidade. Nesse contexto, o método passo a passo proposto por Vieira (2005) se destaca como uma abordagem sistematizada que permite a representação e implementação lógica de sistemas de controle discreto, especialmente em CLPs.

O processo de programação de Controladores Lógicos Programáveis (CLPs) exige uma modelagem clara e representativa do comportamento do sistema ou processo em questão. Segundo Manfrin (2021), a criação de um modelo que represente as interações dos elementos do sistema é essencial para a programação eficiente em CLPs. Este modelo deve detalhar os componentes e como eles se relacionam com os eventos que ocorrem ao longo do tempo, um ponto fundamental para facilitar a compreensão do comportamento do sistema e para a construção de soluções programáveis.

As transições são elementos lógicos que conectam diferentes passos ou condições em uma sequência de controle, permitindo que o sistema evolua de uma etapa para a próxima com base em circunstâncias específicas.

Condições são ocorrências ou mudanças de estado que podem alterar o comportamento de um sistema. Eles podem ser internos (gerados dentro do sistema) ou externos (provenientes do ambiente físico ou de sistemas externos).

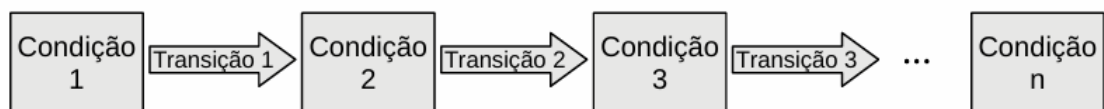


Figura 7 - Sequência de Condições (MANFRIN, 2021)

Esse método é composto por etapas organizadas que facilitam tanto a compreensão quanto a transcrição dos processos industriais em linguagem de programação. As etapas principais incluem: levantamento e descrição textual do processo, definição dos estados (etapas), construção do fluxograma funcional, definição das ações de controle, e finalmente a transcrição para linguagens como o Texto Estruturado (SCL). Segundo Vieira (2005), o objetivo é “fornecer um roteiro claro para a implementação de sistemas que envolvem sequências lógicas complexas”.

Uma das grandes vantagens do método é sua compatibilidade com diversas linguagens normatizadas pela IEC 61131-3, como Ladder, Texto Estruturado e Grafcet. Isso permite sua aplicação em ambientes industriais com diferentes plataformas de CLP, como Siemens, Schneider ou Allen-Bradley. Ao utilizar o método passo a passo como base, o programador é capaz de reduzir erros lógicos, melhorar a manutenção futura do código e garantir um desenvolvimento mais rápido e robusto.

No contexto deste trabalho, o método de Vieira (2005) será empregado como base metodológica formal para a transcrição de processos em Texto Estruturado (SCL), visando a padronização e melhoria da legibilidade dos programas.

A programação de controladores lógicos programáveis (CLPs) para sistemas sequenciais exige uma abordagem estruturada e lógica. O método passo a passo, proposto por Agnelo Denis Vieira, tem se destacado por sistematizar essa implementação de forma clara e eficiente.

Esse método associa a cada passo do SFC (Sequential Function Chart) uma memória específica (M_n) do tipo Set/Reset, o que permite ativar ou desativar estados de maneira lógica e sequencial. Seguindo a lógica da figura 8, o princípio central consiste em garantir que a ativação de um passo da sequência (M_n) habilite o próximo (M_{n+1}) e desative o anterior (M_{n-1}). A transição entre passos ocorre apenas quando a condição de transição (T_n) definida for verdadeira e a memória do passo anterior estiver ativa. Essa lógica garante

previsibilidade e controle sobre o fluxo de execução, conforme mostrado a seguir:

A cada passo é associada uma memória (M_n) do tipo Set/Reset (ativa/desativa);

A ativação da memória (M_n) de um passo habilita a ativação da memória (M_{n+1}) do próximo passo;

A ativação da memória (M_n) desativa a memória (M_{n-1}) do passo anterior;

A ativação da memória (M_n) só ocorre quando a memória do passo anterior (M_{n-1}) estiver ativada e a condição de transição (T_n) for satisfeita.

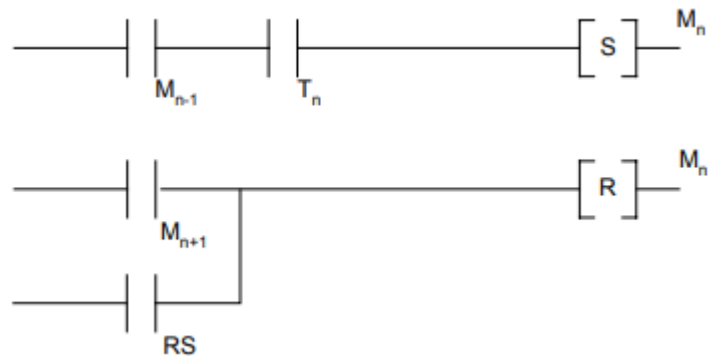


Figura 8 - Método Passo a Passo em Ladder (VIEIRA, 2005)

Conforme pode ser observado na Figura 9, é apresentada a mesma lógica implementada na Figura 8, porém utilizando a linguagem SCL. Dessa forma, quando as variáveis $M(n-1)$ e T_n assumem valor verdadeiro, a memória M_n é acionada. De maneira análoga, quando $M(n+1)$ ou RS assumem valor verdadeiro, a variável booleana M_n é resetada.

```
IF "M(n-1)" AND "Tn" THEN
    "Mn" := 1;
END_IF;

IF "M(n+1)" OR "RS" THEN
    "Mn" := 0;
END_IF;
```

Figura 9 - Método Passo a Passo em SCL – Fonte: Autor

Para ativar a memória M_n , é utilizado a instrução condicional “IF”, sendo descrita da seguinte maneira:

```
IF <condição> THEN

// instruções se condição for verdadeira

ELSIF <outra condição> THEN

// instruções se a outra condição for verdadeira

ELSE

// instruções se nenhuma condição for verdadeira

END_IF;
```

2.6 TEMPORIZADORES

Os temporizadores são funções internas do CLP utilizadas para gerar atrasos ou pulsos temporizados em determinadas condições do processo. Entre os tipos disponíveis, destacam-se:

- TON (Timer On Delay): Temporizador de retardo na energização;
- TOF (Timer Off Delay): Temporizador de retardo na desenergização;
- TP (Pulse Timer): Temporizador de pulso.
- TONR (Timer On Delay Retentivo): Temporizador de retardo na energização com retenção.

2.6.1 TEMPORIZADOR TON

O TON é amplamente utilizado em sistemas industriais. Sua função é atrasar a ativação de uma saída enquanto a entrada de controle permanece energizada. Ele possui quatro principais parâmetros: IN (entrada de controle), PT (tempo pré-ajustado), ET (tempo decorrido) e Q (saída lógica). Quando a entrada IN é acionada, inicia-se a contagem de tempo até o valor definido em PT. Após o término da contagem, a saída Q é ativada. O uso do temporizador TON pode ser representado por meio da figura 10, onde ET representa o tempo de contagem e PT o valor configurado de retardo.

$$Q = \begin{cases} 0, & \text{se } IN = 0 \\ 0, & \text{se } IN = 1 \text{ e } ET < PT \\ 1, & \text{se } IN = 1 \text{ e } ET \geq PT \end{cases}$$

Figura 10 – Temporizador TON (Autor, 2025)

O TON e os outros temporizadores estão localizados na aba de instruções básicas do tia portal, conforme figura 11, podendo ser arrastado ao código para a adição do temporizador

Na Figura 13, é apresentada a estrutura do temporizador TON em SCL no TIA Portal. A variável IN é do tipo booleano, assumindo os valores 0 ou 1. A variável PT é do tipo TIME e pode ser definida diretamente na linha de código, por exemplo, utilizando o formato T#3s. A variável Q também é do tipo booleano, assim como a IN. Já a variável ET, do tipo TIME, indica o tempo decorrido desde que a variável IN assume nível lógico alto, contabilizando até o valor definido em PT.

```

1  ▢ "IEC_Timer_0_DB".TON(IN:=_bool_in_,
2    PT:=_time_in_,
3    Q=>_bool_out_,
4    ET=>_time_out_);
5

```

Figura 13 – Estrutura TON (Autor, 2025)

2.6.2 TEMPORIZADOR TOF

O TOF é um temporizador utilizado para gerar um retardo no momento da desenergização de uma entrada. Seu funcionamento é o oposto do TON: enquanto a entrada permanece ligada, a saída está imediatamente ativa. Somente quando a entrada é desligada é que o temporizador inicia a contagem do tempo programado. Ao final desse tempo, a saída é finalmente desativada, ilustrado na figura 14.

$$Q = \begin{cases} 1, & \text{se } IN = 1 \\ 1, & \text{se } IN = 0 \text{ e } ET < PT \text{ (ainda contando)} \\ 0, & \text{se } IN = 0 \text{ e } ET \geq PT \end{cases}$$

Figura 14 – Lógica TOF (Autor, 2025)

O temporizador TOF está localizado na mesma aba de instruções que o TON, sendo ele estruturado em SCL conforme figura 15.

```

1  "Temporizador".TOF(IN:=_bool_in_,
2      PT:=_time_in_,
3      Q=>_bool_out_,
4      ET=>_time_out_);

```

Figura 15 – Estrutura SCL TOF (Autor, 2025)

2.6.3 TEMPORIZADOR TP

O temporizador do tipo TP (Pulse Timer) é utilizado em CLPs para gerar um pulso de duração fixa. Seu funcionamento baseia-se na detecção de uma borda de subida na entrada IN. Quando essa transição ocorre, o temporizador inicia a contagem do tempo pré-ajustado (PT) e a saída Q é imediatamente ativada.

Durante todo o intervalo em que o tempo decorrido (ET) for inferior a PT, a saída permanece ativa, mesmo que a entrada retorne ao estado lógico baixo. Ao final do tempo programado, o temporizador desativa a saída Q, retornando-a ao estado baixo, preparando-se para um novo ciclo de acionamento. O TP é indicado para aplicações que exigem pulsos controlados e de duração constante e o seu funcionamento pode ser visualizado na figura 16.

$$Q = \begin{cases} 1, & \text{se ocorreu borda de subida de } IN \text{ e } ET < PT \\ 0, & \text{se } ET \geq PT \end{cases}$$

Figura 16 – Lógica TP (Autor, 2025)

Na imagem 17 é possível ver a estrutura em SCL do temporizador TP.

```

1  "Temporizador".TP(IN:=_bool_in_,
2      PT:=_time_in_,
3      Q=>_bool_out_,
4      ET=>_time_out_);

```

Figura 17 – Estrutura SCL TP (Autor, 2025)

2.6.4 TEMPORIZADOR TONR

O temporizador do tipo TONR (Timer On Delay Retentive) é utilizado em CLPs para realizar a temporização com retenção do tempo decorrido, mesmo após a perda do sinal de entrada. Diferentemente do temporizador TON convencional, o TONR não zera automaticamente o valor de ET quando a entrada IN retorna ao estado lógico baixo.

Seu funcionamento ocorre da seguinte forma: quando a entrada IN é ativada, o temporizador inicia a contagem do tempo pré-ajustado (PT). Enquanto $ET < PT$, a saída Q permanece desligada. Ao atingir ou superar o tempo programado ($ET \geq PT$), a saída Q é ativada. Caso a entrada seja desligada antes do término da contagem, o valor de ET é preservado, permanecendo armazenado internamente.

A reinicialização do temporizador somente ocorre mediante a atuação de um comando de reset, que zera o tempo decorrido e retorna o bloco ao estado inicial. Dessa forma, o TONR é adequado para processos que exigem acúmulo de tempo, mesmo diante de interrupções momentâneas do sinal de entrada, como ciclos de operação intermitentes ou condições de segurança que exigem retomada precisa da temporização, o estado da sua saída pode ser visto na figura 18.

$$Q = \begin{cases} 0, & ET < PT \\ 1, & ET \geq PT \end{cases}$$

Figura 18 – Lógica TONR (Autor, 2025)

Na Figura 19, observa-se a estrutura do temporizador TONR em SCL. O TONR possui uma variável adicional em relação aos demais temporizadores, denominada R, que é do tipo booleano e pode assumir os valores 0 ou 1.

```
1  ▢ "Temporizador".TONR(IN:=_bool_in_,  
2    R:=_bool_in_,  
3    PT:=_time_in_,  
4    Q=>_bool_out_,  
5    ET=>_time_out_);
```

Figura 19 – Estrutura SCL TONR (Autor, 2025)

2.7 CONTADORES

Os contadores são funções internas do CLP utilizadas para registrar a quantidade de eventos ocorridos em um processo, como pulsos de sensores, ciclos de operação e acionamentos repetitivos. Entre os tipos disponíveis, destacam-se:

- CTU (Count Up): Contador crescente, incrementa o valor da contagem a cada borda de subida na entrada de pulso.
- CTD (Count Down): Contador decrescente, reduz o valor da contagem a cada borda de subida na entrada de pulso.
 - CTUD (Count Up/Down): Contador bidirecional, capaz de incrementar ou decrementar o valor da contagem conforme o comando recebido nas entradas de subida e descida.

2.7.1 CONTADOR CTU

O CTU (Count Up) é um contador crescente utilizado em CLPs para realizar incrementos sucessivos a cada pulso detectado na entrada de contagem. Sua função é acumular o número de eventos, habilitando uma saída quando o valor acumulado atinge ou ultrapassa o valor pré-ajustado (PV – Preset Value). A lógica do contador é exposta conforme figura 20.

$$Q = \begin{cases} 0, & \text{se } CV < PV \\ 1, & \text{se } CV \geq PV \end{cases}$$
$$CV = \begin{cases} 0, & \text{se } R = 1 \\ CV + 1, & \text{na borda de subida de } CU \\ CV, & \text{caso contrário} \end{cases}$$

Figura 20 – Lógica CTU (Autor, 2025)

O CTU está localizado na aba de “Counter operations”, como observado na figura 21.

Basic instructions		
Name	Description	
▶ Bit logic operations		
▶ Timer operations		
▼ Counter operations		
IEC Counters		
CTU	Count up	
CTD	Count down	
CTUD	Count up and down	

Figura 21 – Localização do CTU no software (Autor, 2025)

Na Figura 22, é apresentada a estrutura do contador CTU em SCL. As variáveis CU, R e Q são do tipo booleano, assumindo os valores 0 ou 1. Já as variáveis PV e CV são do tipo inteiro, representando os valores de contagem.

```

1  "Contador".CTU(CU:=bool_in,
2                      R:=bool_in,
3                      PV:=in,
4                      Q=>bool_out,
5                      CV=>out);

```

Figura 22 – Estrutura SCL CTU (Autor, 2025)

2.7.2 CONTADOR CTD

O CTD (Counter Down) é um contador decrescente utilizado em Controladores Lógicos Programáveis (CLPs) para realizar a contagem regressiva de eventos, pulsos ou ciclos de processo. Esse tipo de contador reduz o valor atual de contagem (CV) a cada borda de subida recebida na entrada de decremento (CD).

O bloco também possui a entrada de carga (LD), que permite restaurar o valor pré-determinado (PV) sempre que necessária a reinicialização da contagem. Quando o valor atual atinge zero, a saída lógica (Q) é desativada,

indicando que a contagem regressiva foi concluída. Sua lógica de funcionamento pode ser visualizada na figura 23.

$$Q = \begin{cases} 1, & \text{se } CV > 0 \\ 0, & \text{se } CV = 0 \end{cases}$$

$$CV = \begin{cases} PV, & \text{se } LD = 1 \\ CV - 1, & \text{se } CD = 1 \\ CV, & \text{caso contrário} \end{cases}$$

Figura 23 – Lógica CTD (Autor, 2025)

Conforme ilustrado na Figura 24, apresenta-se a estrutura do contador CTD em SCL. O CTD possui as mesmas variáveis do CTU, com a diferença de que a variável R é substituída pela variável LD, também do tipo booleano.

```

1  "Contador".CTD(CD:=bool_in,
2      LD:=bool_in,
3      PV:=in,
4      Q=>bool_out,
5      CV=>out);

```

Figura 24 – Estrutura SCL CTD (Autor, 2025)

2.7.3 CONTADOR CTUD

O CTUD (Counter Up/Down) é um contador bidirecional utilizado em Controladores Lógicos Programáveis (CLPs) para executar operações de contagem tanto crescente quanto decrescente. Esse bloco combina as funcionalidades dos contadores CTU e CTD, permitindo incrementar ou decrementar o valor atual (CV) de acordo com os pulsos recebidos.

O contador possui duas entradas de contagem:

- CU (Count Up): incrementa o valor atual (CV) a cada borda de subida.

- CD (Count Down): decrementa o valor atual (CV) a cada borda de subida.

Além disso, dispõe das entradas:

- LD (Load): carrega o valor pré-definido (PV) no contador, permitindo reinicialização imediata.
- RESET: zera o valor atual (CV), retornando o contador ao estado inicial.

$$CV = \begin{cases} 0, & \text{se RESET} = 1 \\ PV, & \text{se LD} = 1 \\ CV + 1, & \text{se CU} = 1 \\ CV - 1, & \text{se CD} = 1 \\ CV, & \text{caso contrário} \end{cases} \quad \begin{aligned} QU &= \begin{cases} 1, & \text{se } CV \geq PV \\ 0, & \text{se } CV < PV \end{cases} \\ QD &= \begin{cases} 1, & \text{se } CV = 0 \\ 0, & \text{se } CV > 0 \end{cases} \end{aligned}$$

Figura 25 – Lógica CTUD (Autor, 2025)

A Figura 26 apresenta a estrutura do contador CTUD em SCL. Diferentemente dos contadores CTU e CTD, o CTUD possui variáveis específicas para incrementar e decrementar a contagem, como, por exemplo, CU para incremento e CD para decremento, mantendo, entretanto, a mesma lógica de funcionamento.

```
1 "Contador".CTUD(CU:=_bool_in_,
2                 CD:=_bool_in_,
3                 R:=_bool_in_,
4                 LD:=_bool_in_,
5                 PV:=_in_,
6                 QU=>_bool_out_,
7                 QD=>_bool_out_,
8                 CV=>_out_);
```

Figura 26 – Estrutura SCL CTUD (Autor, 2025)

2.8 CONCLUSÕES DO CAPÍTULO

Neste capítulo foram apresentados os fundamentos teóricos necessários para compreender o funcionamento dos Controladores Lógicos Programáveis e sua aplicação em sistemas de automação industrial. A análise do hardware do CLP, de sua estrutura de memória, do ciclo de scan e das unidades de entrada e saída permitiu estabelecer uma base sólida para o entendimento de como as informações são processadas e como o controlador interage com os dispositivos de campo.

Além disso, a comparação entre as linguagens de programação Ladder e SCL evidenciou as particularidades e vantagens de cada uma, destacando o potencial do Texto Estruturado para o desenvolvimento de soluções mais estruturadas e modulares. Por fim, a apresentação do método passo a passo forneceu o embasamento conceitual para a modelagem sequencial adotada neste trabalho, demonstrando sua importância como ferramenta formal para a implementação de sistemas automatizados.

Na sequência, foram apresentados os principais temporizadores (TON, TOF, TP e TONR) e contadores (CTU, CTD e CTUD) disponíveis no TIA Portal, juntamente com sua lógica de funcionamento e estrutura de aplicação em Texto Estruturado (SCL).

3 CONCEITOS BÁSICOS DO TIA PORTAL

3.1 SOFTWARE

O TIA Portal (Totally Integrated Automation Portal) é o ambiente de desenvolvimento integrado da Siemens, utilizado para a configuração, programação, simulação e diagnóstico de controladores lógicos programáveis (CLPs) e interfaces homem-máquina (IHMs). A principal característica dessa plataforma é a integração de todas as etapas do projeto de automação em um único software, possibilitando a criação e a gestão de sistemas industriais de forma unificada e eficiente.

No TIA Portal é possível realizar a configuração do hardware, estabelecer a comunicação entre dispositivos via redes industriais (como PROFINET e Modbus TCP), desenvolver programas em diferentes linguagens de programação (Ladder, Bloco de Funções e Texto Estruturado – SCL) e criar telas de supervisão para IHMs. Essa integração facilita o comissionamento, a manutenção e o diagnóstico de falhas em sistemas automatizados.

3.2 ESTRUTURA DE BLOCOS

O projeto de um CLP Siemens no TIA Portal é organizado em blocos de programa, cada um com uma função específica. Os principais tipos são:

- OB (Organizational Blocks): controlam a execução do programa e determinam a sequência lógica das tarefas do CLP;
- FB (Function Blocks): blocos de função com memória interna, utilizados para implementar funções que necessitam armazenar estados;
- FC (Functions): blocos de função sem memória interna, usados para cálculos e operações que não requerem retenção de valores;
- DB (Data Blocks): áreas de memória destinadas ao armazenamento de variáveis e dados de processo.

O OB1 é o bloco principal do programa, sendo executado de forma cíclica durante o funcionamento normal do CLP. Nele são chamadas as funções e blocos responsáveis pelas diversas etapas do processo automatizado. Outros

OBs, como o OB100 e o OB35, são utilizados para funções específicas, como inicialização do sistema e execução periódica de tarefas.

3.3 CRIAÇÃO DO PROJETO E CONFIGURAÇÃO DO CLP

A inserção do controlador lógico programável (CLP) no ambiente de desenvolvimento TIA Portal constitui uma das primeiras etapas na criação de um sistema de automação. Esse procedimento tem como objetivo definir o hardware que será utilizado no projeto, permitindo a configuração de parâmetros de rede, endereçamento de entradas e saídas, bem como a posterior implementação da lógica de controle.

Para iniciar o processo, é necessário criar um novo projeto no TIA Portal, por meio do menu Project → New Project, onde são definidos o nome e o diretório de salvamento, conforme figuras 27 e 28.

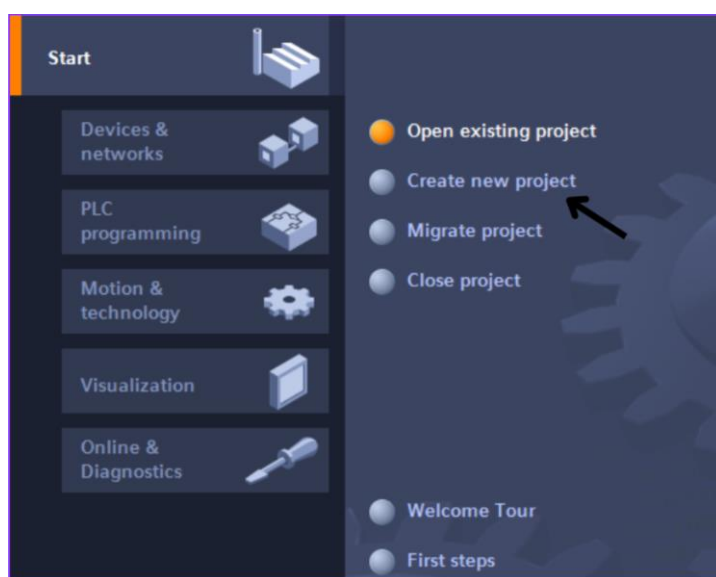


Figura 27 – Criação de Projeto (Autor, 2025)



Figura 28 – Criação de Projeto (Autor, 2025)

Após a criação, o usuário deve acessar o modo de visualização do projeto (Project View) e adicionar um novo dispositivo, conforme figura 29.

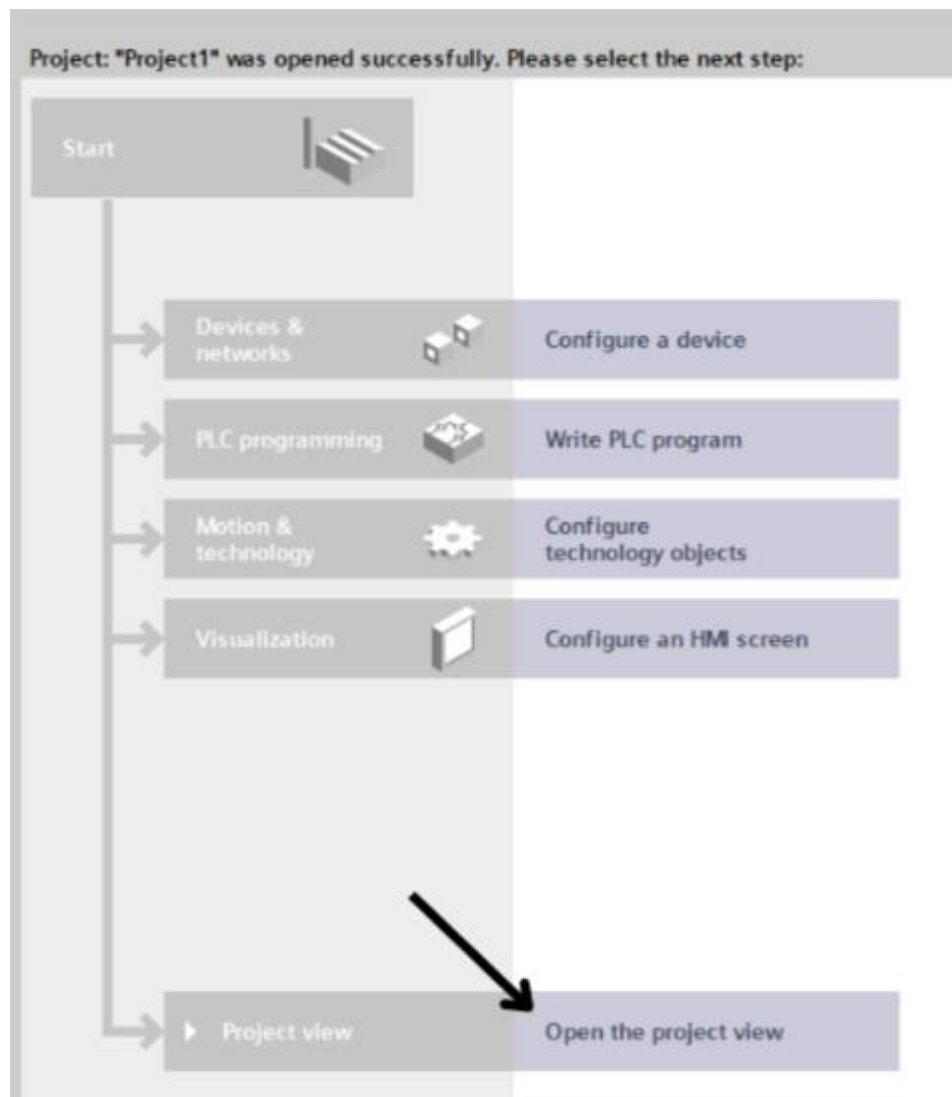


Figura 29 – Criação de Projeto (Autor, 2025)

Essa operação é realizada selecionando “Add new device”, onde são exibidas as opções de controladores disponíveis, como as famílias SIMATIC S7-1200 e S7-1500. Em seguida, é escolhido o modelo específico da CPU a ser utilizada, conforme o hardware físico do sistema, e confirmado o comando para inseri-la no projeto, visto nas figuras 30 e 31.

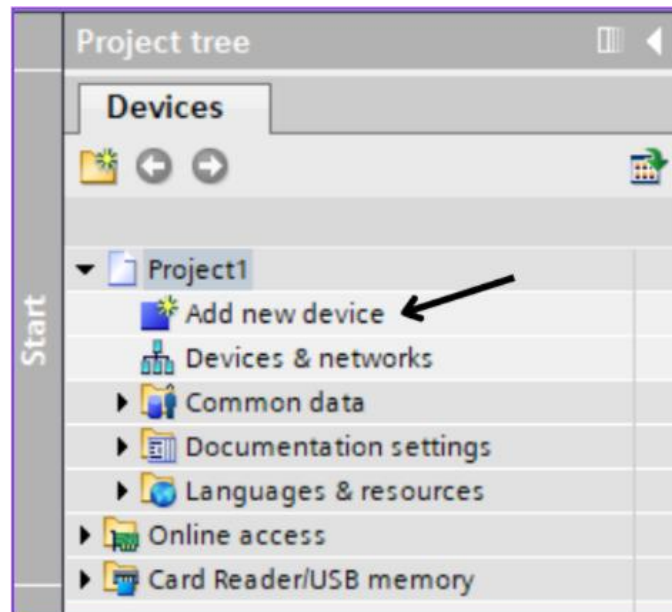


Figura 30 – Adicionando o CLP (Autor, 2025)

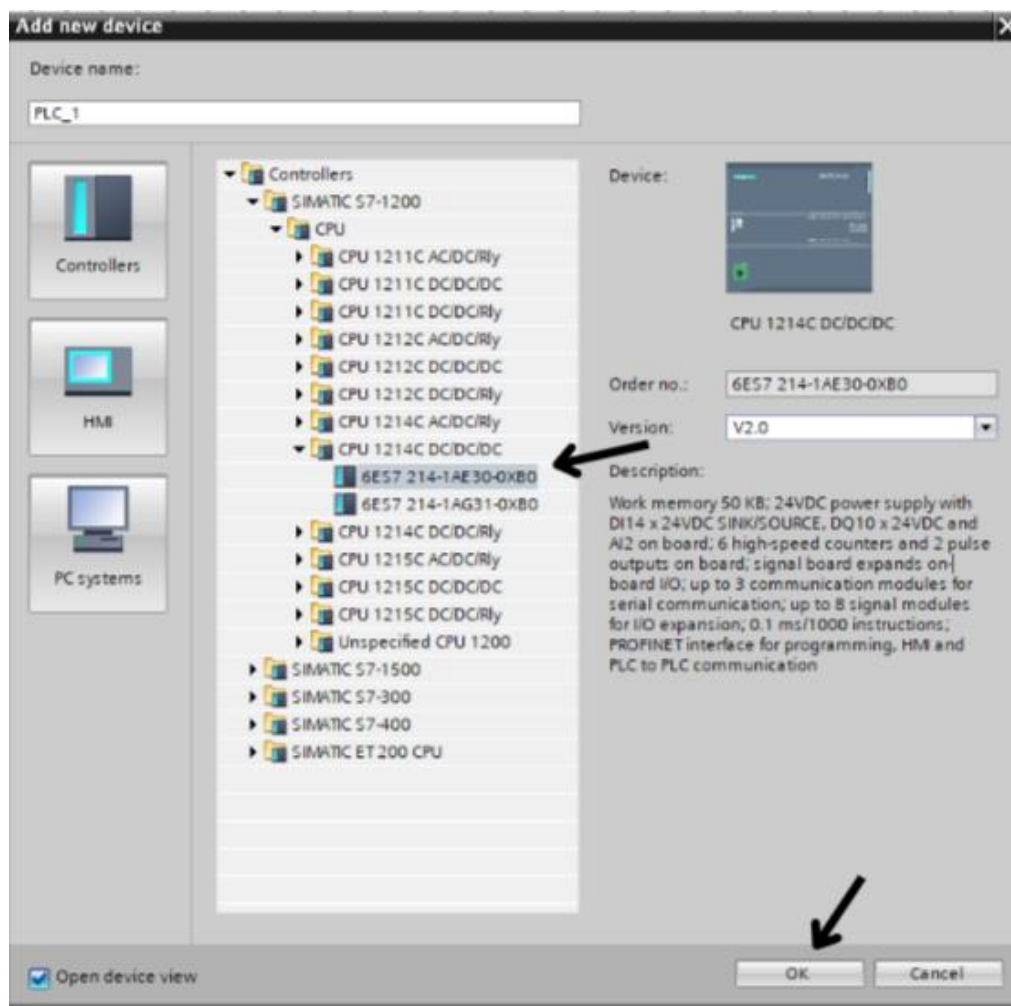


Figura 31 – Adicionando o CLP (Autor, 2025)

Na etapa seguinte, o TIA Portal apresenta a interface gráfica do dispositivo, denominada “Device configuration”, onde é possível adicionar módulos de expansão, como entradas e saídas digitais, módulos analógicos e interfaces de comunicação. Esses módulos são selecionados a partir do catálogo de hardware disponível no ambiente e inseridos na CPU por meio de arraste. Nesse momento, também são definidos os endereços lógicos correspondentes a cada módulo, que serão utilizados posteriormente na programação do CLP, visualizado na figura 32.

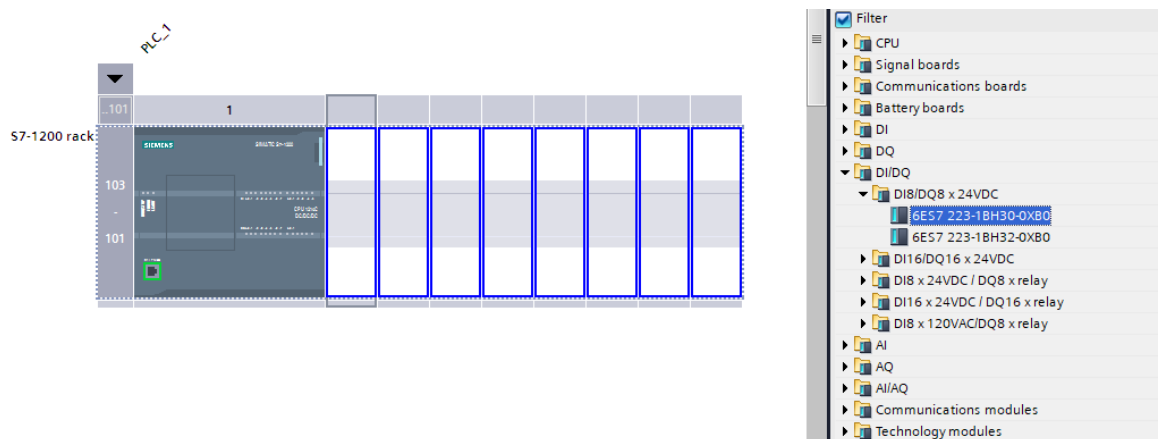


Figura 32 – Adicionando Cartões de Expansão (Autor, 2025)

Na figura 33, realiza-se a configuração da comunicação da CPU, definindo o endereço IP na aba Properties → Ethernet Addresses. Essa configuração é essencial para a integração do CLP com outros dispositivos da rede, como interfaces homem-máquina (IHMs), inversores de frequência ou outros controladores.

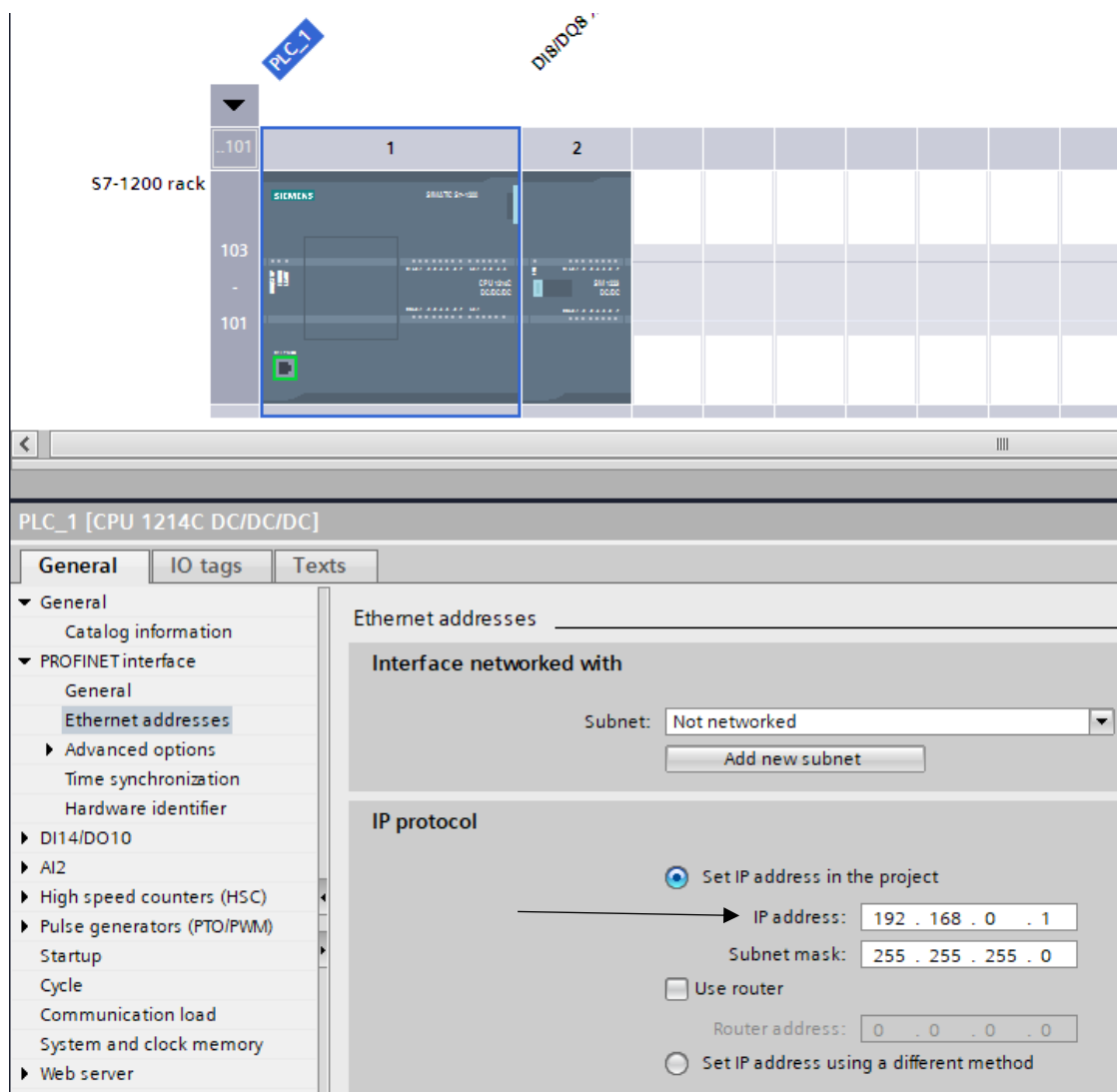


Figura 33 – Configuração CLP (Autor, 2025)

3.4 CRIAÇÃO DE BLOCOS OB

A criação de um bloco OB é feita a partir da área de Program blocks do projeto. O usuário deve clicar com o botão direito do mouse, selecionar “Add new block” e escolher o tipo de OB desejado (por exemplo, OB1 para o programa cíclico principal ou OB35 para tarefas periódicas), selecionando também a linguagem de programação desejada, conforme figuras 34 e 35.

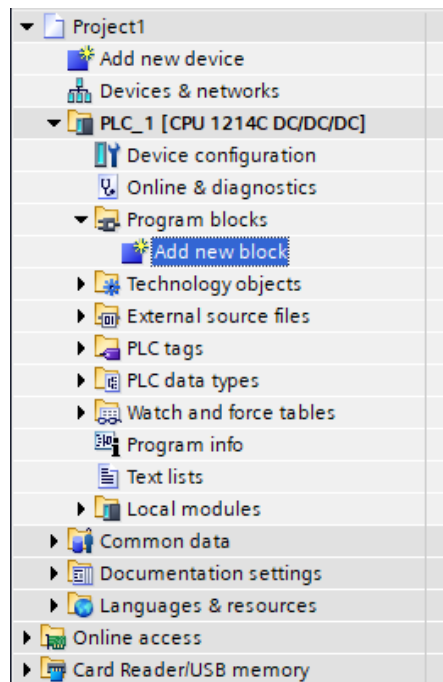


Figura 34 – Adicionando Blocos (Autor, 2025)

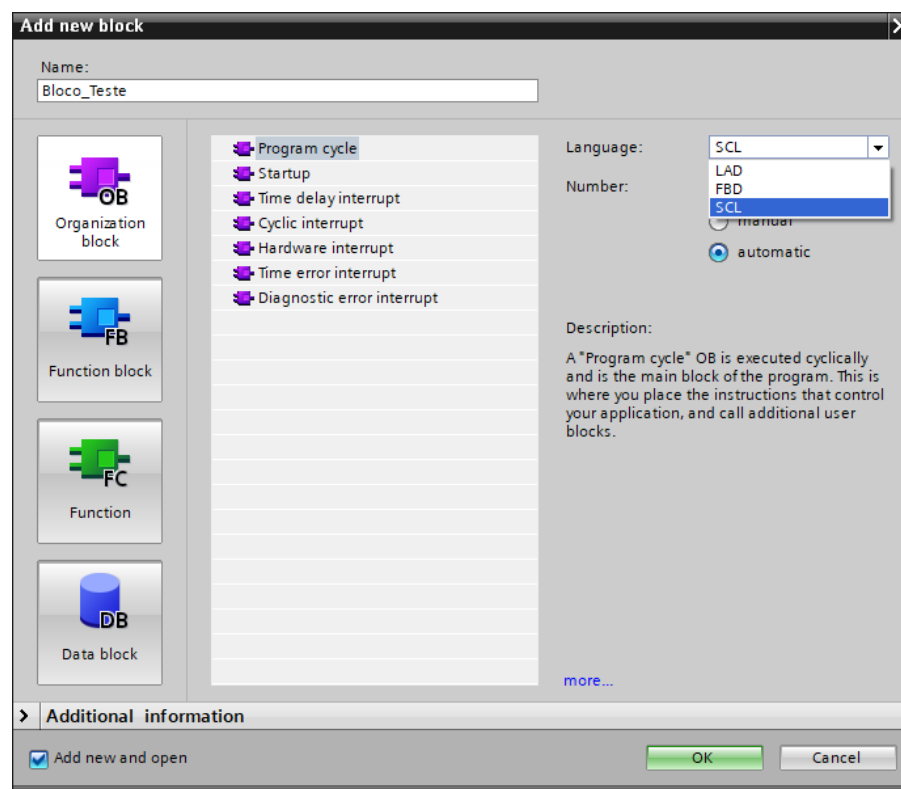
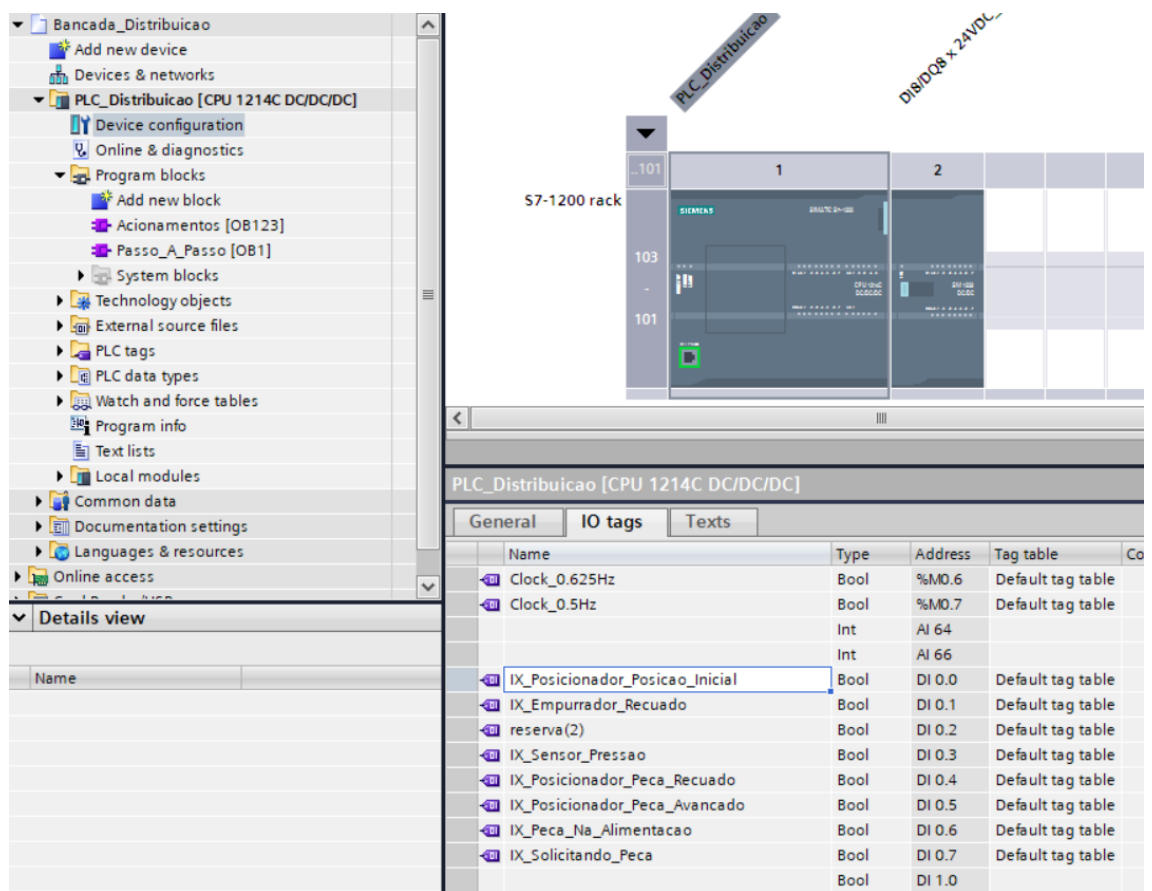


Figura 35 – Adicionando Blocos (Autor, 2025)

3.5 DEFINIÇÃO DE IO'S NO TIA PORTAL

No software da Siemens, é possível nomear as entradas e saídas do CLP conforme a necessidade do projeto. Para isso, deve-se acessar a opção *Device Configuration* e selecionar o CLP. Em seguida, basta abrir a aba de propriedades e escolher a seção *IO Tags*, onde todas as entradas e saídas são listadas, permitindo sua nomeação conforme desejado, conforme ilustrado na Figura 36.



The screenshot displays the Siemens TIA Portal interface. On the left, the 'Project Explorer' shows the hierarchy: 'Bancada_Distribuicao' > 'Devices & networks' > 'PLC_Distribuicao [CPU 1214C DC/DC/DC]' > 'Device configuration'. The 'Details view' is open, showing the 'IO tags' tab for the selected PLC. The table below lists the configured IO tags.

Name	Type	Address	Tag table	Comments
Clock_0.625Hz	Bool	%M0.6	Default tag table	
Clock_0.5Hz	Bool	%M0.7	Default tag table	
	Int	AI 64		
	Int	AI 66		
IX_Posicionador_Posicao_Inicial	Bool	DI 0.0	Default tag table	
IX_Empurrador_Recuado	Bool	DI 0.1	Default tag table	
reserva(2)	Bool	DI 0.2	Default tag table	
IX_Sensor_Pressao	Bool	DI 0.3	Default tag table	
IX_Posicionador_Peca_Recuado	Bool	DI 0.4	Default tag table	
IX_Posicionador_Peca_Avancado	Bool	DI 0.5	Default tag table	
IX_Peca_Na_Alimentacao	Bool	DI 0.6	Default tag table	
IX_Solicitando_Peca	Bool	DI 0.7	Default tag table	
	Bool	DI 1.0		

Figura 36 – Definição IO's Tia Portal (Autor, 2025)

3.6 CONCLUSÕES DO CAPÍTULO

Neste capítulo foram apresentados os principais conceitos referentes ao uso do TIA Portal no desenvolvimento de sistemas de automação industrial. Inicialmente, descreveu-se o software TIA Portal como uma plataforma integrada capaz de unificar configuração, programação, diagnóstico e supervisão de CLPs e IHMs, destacando seus recursos e vantagens na automação. Em seguida, abordou-se a estrutura de blocos utilizada pelos controladores Siemens, OBs, FBs, FCs e DBs, explicando a função de cada um e sua importância na organização do programa.

Também foi detalhado o processo de criação de um novo projeto, incluindo a inserção da CPU, a configuração do hardware, o endereçamento de entradas e saídas e os ajustes de comunicação via Ethernet. Além disso, descreveu-se o procedimento para a criação de blocos OB dentro do ambiente de programação.

Por fim, apresentou-se o processo de nomeação das entradas e saídas do CLP, realizado por meio da aba IO Tags na configuração do dispositivo, recurso que permite padronizar o projeto e facilitar sua interpretação.

Os conhecimentos discutidos ao longo do capítulo servem como suporte direto para a etapa seguinte, na qual o método será aplicado à modelagem e programação do sistema proposto, integrando teoria e prática de forma coerente.

4 APLICAÇÃO DO MÉTODO

Antes da modelagem e implementação do sistema sequencial utilizando o método passo a passo, é fundamental realizar o levantamento completo das entradas e saídas envolvidas no processo de automação. Esse levantamento consiste na identificação dos dispositivos de campo conectados ao Controlador Lógico Programável (CLP), classificando-os conforme suas funções no sistema: sensores e botoeiras (entradas) ou atuadores como motores, válvulas e sinalizadores (saídas).

A análise detalhada das entradas e saídas é essencial para garantir a correta representação lógica do processo automatizado e possibilita a elaboração de um diagrama preciso que reflita o comportamento desejado do sistema. Com base nessas informações, cada etapa da sequência será associada a condições de transição que envolvem as entradas, e a ações que envolvem as saídas, conforme indica o método passo a passo.

Na Figura 37, é possível visualizar a estrutura completa do sistema composto pelas bancadas didáticas da Festo, permitindo uma compreensão integrada dos módulos, dispositivos e interligações que compõem o processo automatizado.

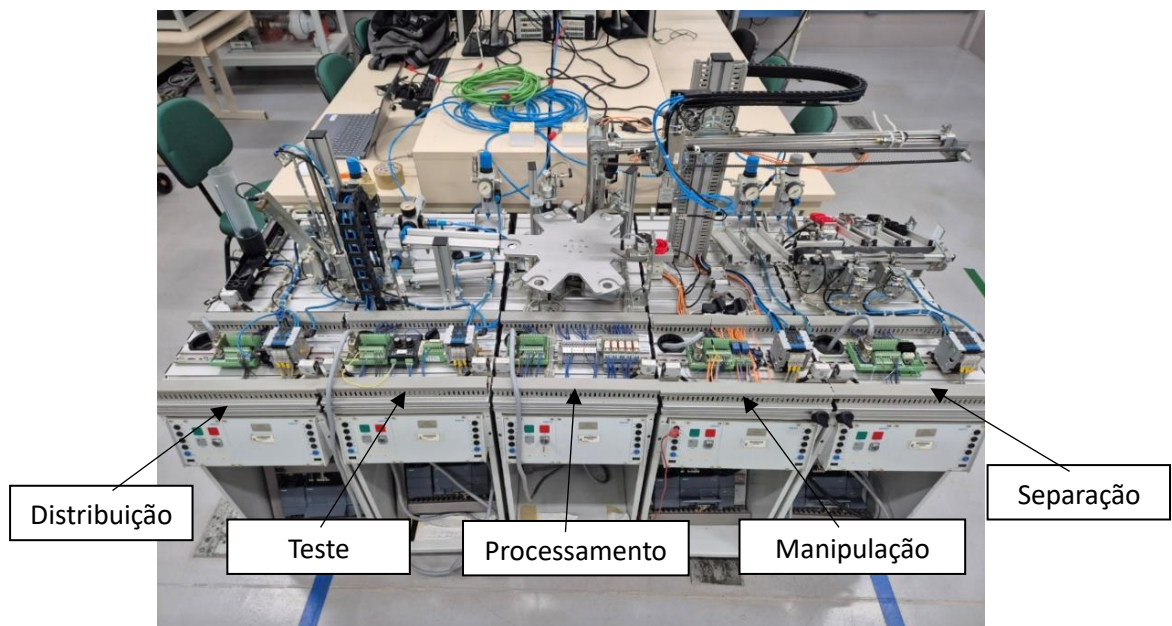


Figura 37 – Bancadas Festo (Autor, 2025)

4.1 BANCADA DE DISTRIBUIÇÃO

Apresenta-se nesta seção a bancada de distribuição. Essa estação tem papel fundamental no processo, sendo responsável por fornecer as peças que darão início à sequência automatizada das demais bancadas do sistema, sendo ela vista na figura 38.

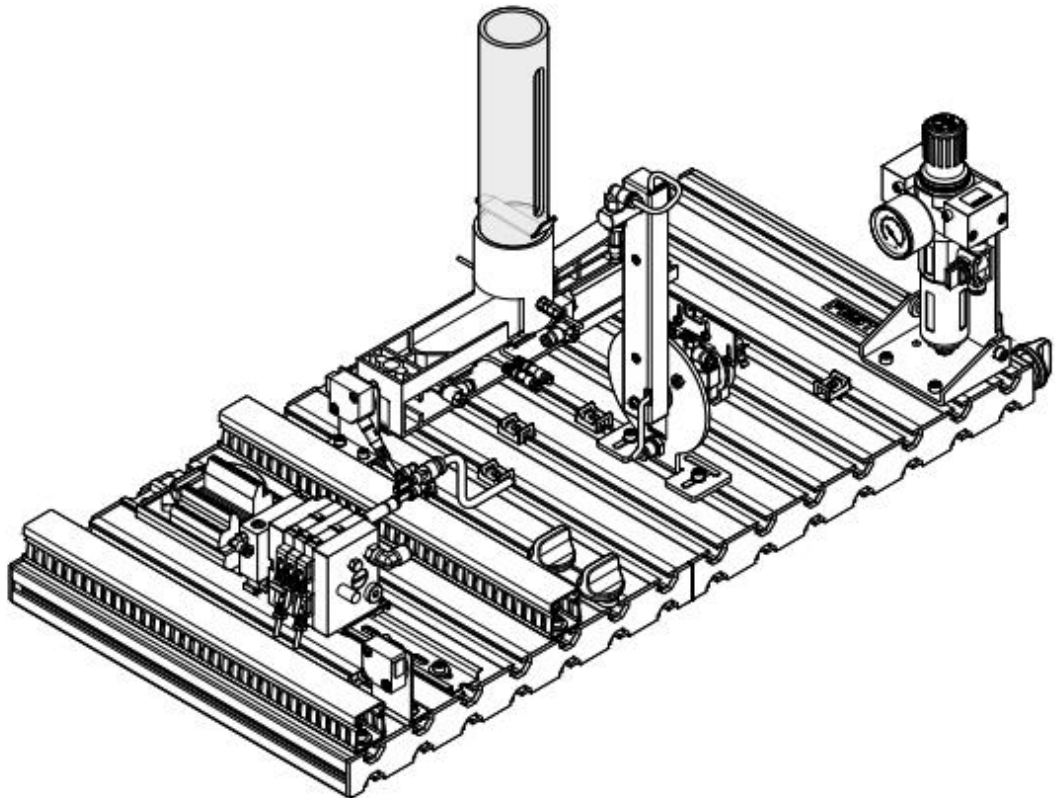


Figura 38 – Bancada de Distribuição (Festo, 2025)

4.1.1 IO'S BANCADA DE DISTRIBUIÇÃO

Na Tabela 1 são apresentadas as entradas e saídas da bancada de distribuição, organizadas de forma a facilitar a identificação dos dispositivos envolvidos no processo.

Tipo	Endereço no CLP	Descrição do Dispositivo	Nome da Tag	Tipo de Sinal
Entrada	I8.0	Botão de partida	IX_Start	Digital
Entrada	I0.0	Posicionador posição inicial	IX_Posicionador_Posicao_Inicial	Digital
Entrada	I0.1	Empurrador recuado	IX_Empurrador_Recuado	Digital
Entrada	I0.4	Posicionador recuado	IX_Posicionador_Peca_Recuado	Digital
Entrada	I0.5	Posicionador avançado	IX_Posicionador_Peca_Avancado	Digital
Entrada	I0.6	Peça na alimentação	IX_Peca_Na_Alimentacao	Digital
Entrada	I0.7	Bancada solicitando peça	IX_Solicitando_Peca	Digital
Saída	Q0.0	Empurrador	QX_Empurrador	Digital
Saída	Q0.1	Ventosa Puxa	QX_Ventosa_Puxa	Digital
Saída	Q0.2	Ventosa Solta	QX_Ventosa_Solta	Digital
Saída	Q0.3	Recua posicionador	QX_Posicionador_Peca_Recua	Digital
Saída	Q0.4	Avança posicionador	QX_Posicionador_Peca_Avanca	Digital

Tabela 1 - IO's Bancada Distribuição (Autor, 2025)

4.1.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA

Na Figura 39 é possível visualizar a localização das entradas e saídas relacionadas à bancada de distribuição, permitindo identificar de forma clara a disposição dos dispositivos utilizados nesta estação.

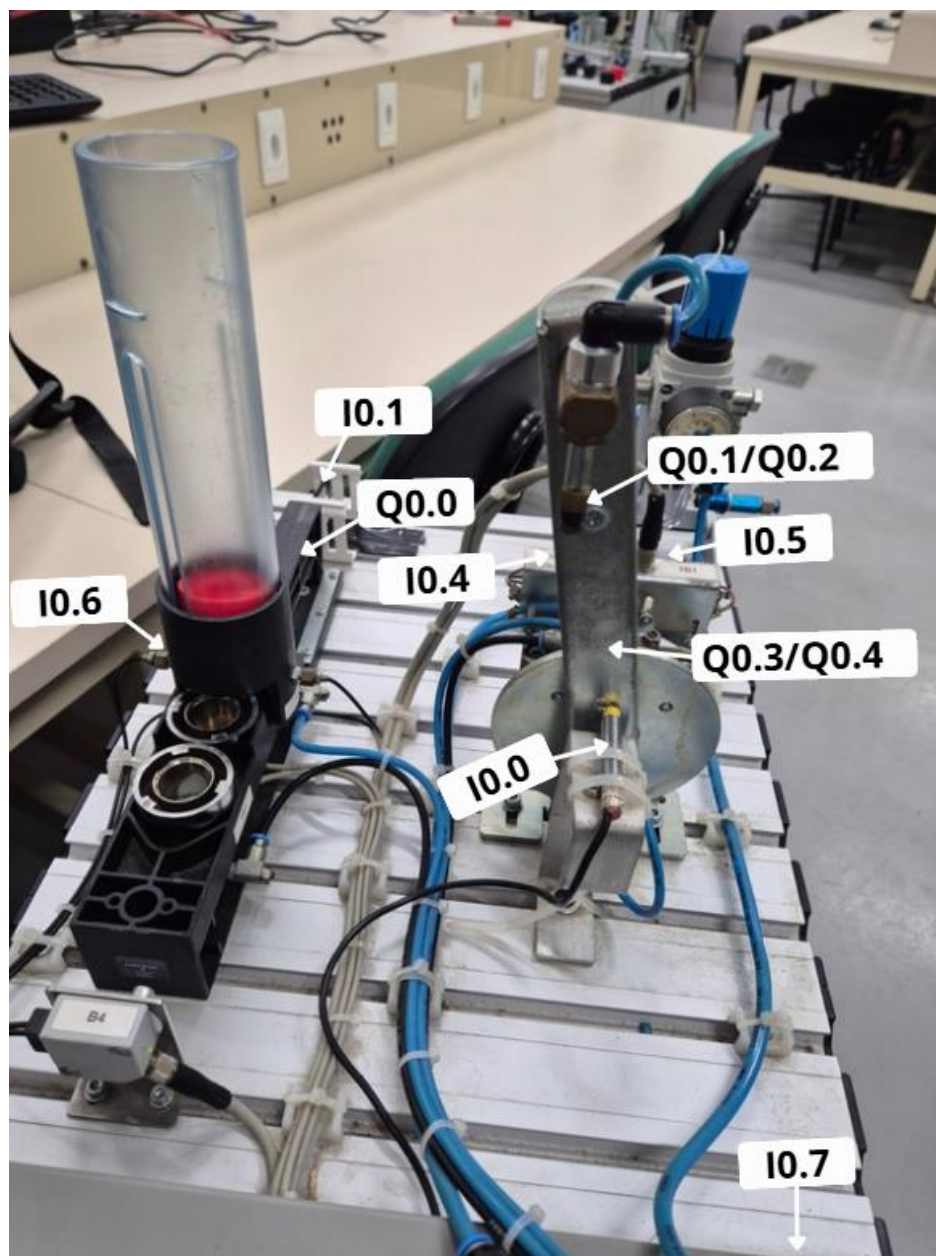


Figura 39 – IO's na Bancada Física (Autor, 2025)

4.1.3 APLICAÇÃO DO MÉTODO PASSO A PASSO

Conforme ilustrado na Figura 40, o funcionamento passo a passo da bancada de distribuição inicia-se de maneira semelhante às demais bancadas do sistema. Inicialmente, o CLP é energizado e entra no passo 0, permanecendo em espera até que o botão de partida (I8.0) seja pressionado, momento em que ocorre a transição para o passo 1.

A transição do passo 1 para o passo 2 ocorre quando há a queda do sinal do sensor que detecta a presença de peça na alimentação da bancada (I0.6), em conjunto com o sinal do sensor de recuo do cilindro empurrador (I0.1). No passo 2, inicia-se a contagem do temporizador de avanço do empurrador, simultaneamente à ativação da saída responsável pelo avanço do cilindro.

Após a conclusão da contagem, o passo 3 é acionado, promovendo o recuo do posicionador. Quando o sensor de posicionador recuado (I0.4) é detectado, ocorre a transição para o passo 4, no qual é acionado um temporizador de espera, juntamente com a saída que recua o posicionador e a saída da ventosa responsável por realizar a pega da peça.

Com o término do tempo e a solicitação de peça pela bancada seguinte (I0.7), o passo 5 é ativado, promovendo o avanço do posicionador. Ao atingir o sensor de posicionador avançado (I0.5), ocorre a transição para o passo 6, momento em que a peça é liberada na próxima bancada, sendo novamente acionado o temporizador de espera para aguardar alguns segundos.

Finalizado o tempo, o passo 7 é ativado, provocando o recuo do posicionador. Quando o sensor de posição inicial (I0.0) é detectado, o sistema avança para o passo 8, no qual ocorre uma nova espera temporizada, retornando em seguida ao passo 1, reiniciando assim o ciclo de operação da bancada.

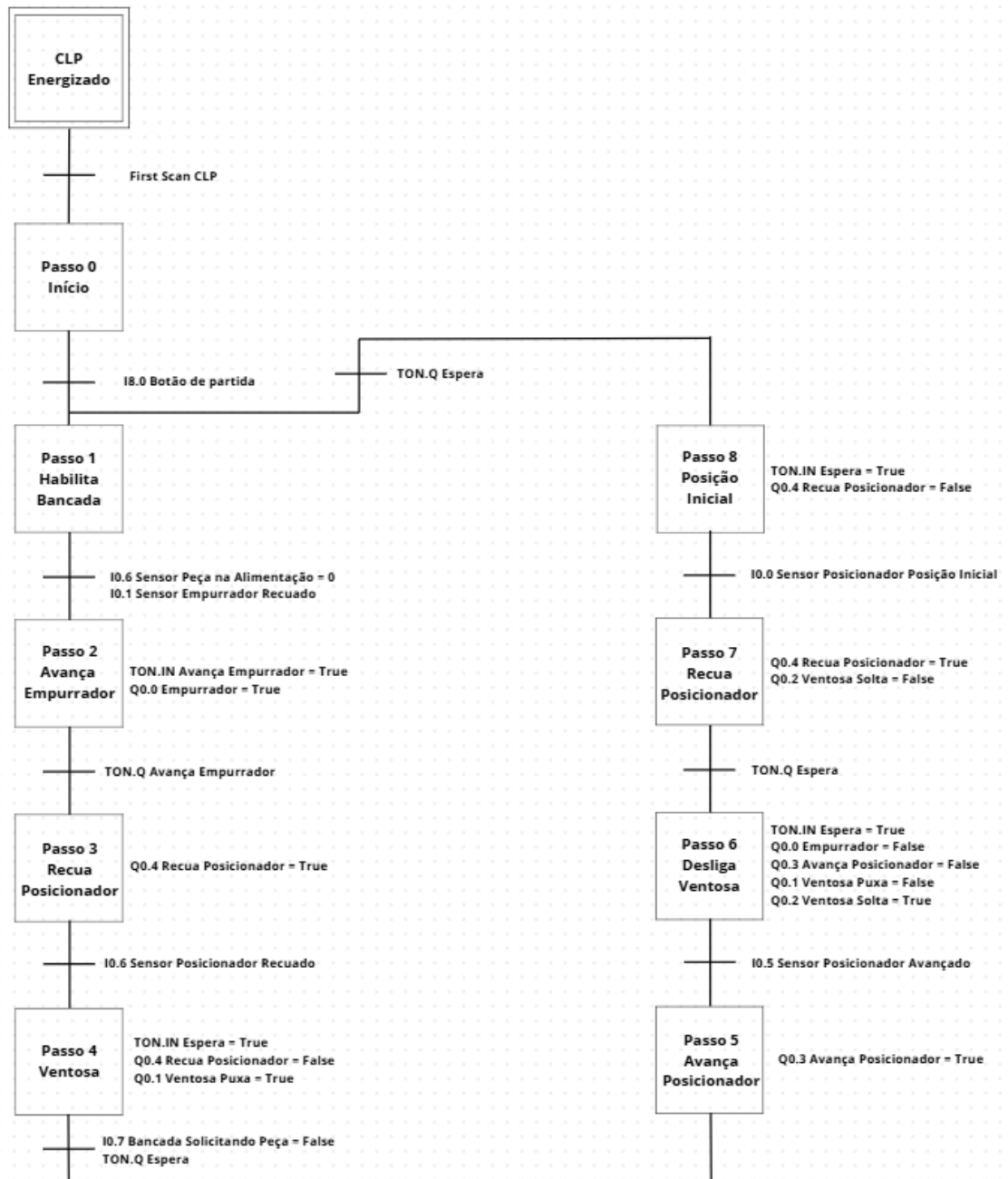
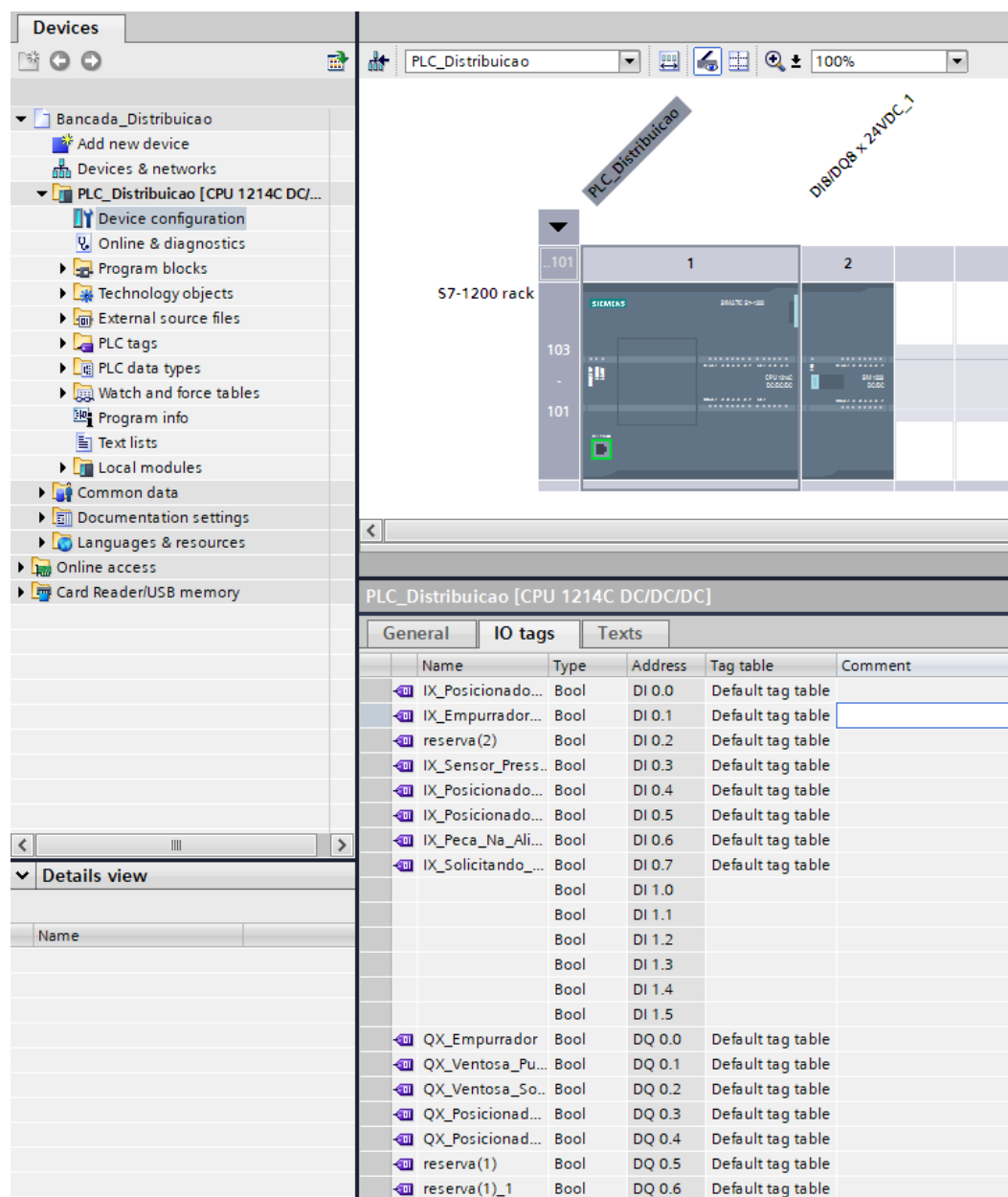


Figura 40 – Passo a Passo Bancada de Distribuição (Autor, 2025)

4.1.4 IMPLEMENTAÇÃO NO TIA PORTAL

As variáveis de entradas e saídas foram definidas na aba IO's tags no hardware do CLP, conforme figura 41.



The screenshot displays the Siemens TIA Portal interface for configuring IO tags. The left pane shows the project tree with 'PLC_Distribuicao [CPU 1214C DC/DC/DC]' selected. The right pane shows a rack diagram with a Siemens CPU 1214C-2 and a DI8/DQ8 x 24VDC module. Below the rack diagram is a table of IO tags.

Name	Type	Address	Tag table	Comment
IX_Posicionado...	Bool	DI 0.0	Default tag table	
IX_Empurrador...	Bool	DI 0.1	Default tag table	
reserva(2)	Bool	DI 0.2	Default tag table	
IX_Sensor_Press...	Bool	DI 0.3	Default tag table	
IX_Posicionado...	Bool	DI 0.4	Default tag table	
IX_Posicionado...	Bool	DI 0.5	Default tag table	
IX_Peca_Na_Ali...	Bool	DI 0.6	Default tag table	
IX_Solicitando...	Bool	DI 0.7	Default tag table	
	Bool	DI 1.0		
	Bool	DI 1.1		
	Bool	DI 1.2		
	Bool	DI 1.3		
	Bool	DI 1.4		
	Bool	DI 1.5		
QX_Empurrador	Bool	DQ 0.0	Default tag table	
QX_Ventosa_Pu...	Bool	DQ 0.1	Default tag table	
QX_Ventosa_So...	Bool	DQ 0.2	Default tag table	
QX_Posicionad...	Bool	DQ 0.3	Default tag table	
QX_Posicionad...	Bool	DQ 0.4	Default tag table	
reserva(1)	Bool	DQ 0.5	Default tag table	
reserva(1)_1	Bool	DQ 0.6	Default tag table	

Figura 41 – Definição de IO's CLP (Autor, 2025)

As memórias de cada passo e seus endereços foram definidas na “Default tag table” em “PLC tags”, conforme pode ser visto na figura 42.

		Name	Data type	Address
▼ Bancada_Distribuicao				
Add new device	13	QX_Posicionador_Peca_Recua	Bool	%Q0.3
Devices & networks	14	reserva(1)	Bool	%Q0.5
▼ PLC_Distribuicao [CPU 1214C DC/...	15	reserva(1)_1	Bool	%Q0.6
Device configuration	16	reserva(1)_2	Bool	%Q0.7
Online & diagnostics	17	reserva(1)_3	Bool	%Q1.0
Program blocks	18	reserva(1)_4	Bool	%Q1.1
Technology objects	19	IX_Start	Bool	%I8.0
External source files	20	IX_Stop	Bool	%I8.1
▼ PLC tags	21	IX_Man_Aut	Bool	%I8.2
Show all tags	22	IX_Reset	Bool	%I8.3
Add new tag table	23	Passo_0	Bool	%M100.0
Default tag table [63]	24	Passo_1	Bool	%M100.1
PLC data types	25	Passo_2	Bool	%M100.2
Watch and force tables	26	Passo_3	Bool	%M100.3
Program info	27	Passo_4	Bool	%M100.4
Text lists	28	Passo_5	Bool	%M100.5
Local modules	29	Passo_6	Bool	%M100.6
Common data	30	Passo_7	Bool	%M100.7
Documentation settings	31	Passo_8	Bool	%M101.0
Languages & resources	32	Passo_9	Bool	%M101.1
Online access	33	Passo_10	Bool	%M101.2

Figura 42 – Definição de Memórias dos Passos (Autor, 2025)

A memória “First Scan”, que é acionada durante o primeiro ciclo do CLP, pode ser habilitada através da configuração do dispositivo. Para isso, deve-se acessar a aba “Device Configuration”, selecionar o CLP e abrir a aba “Properties”. Em seguida, na seção General, é necessário selecionar “System and Clock Memory” e habilitar a opção “Enable the Use of System Memory Byte”, podendo ser visualizado na figura 43.

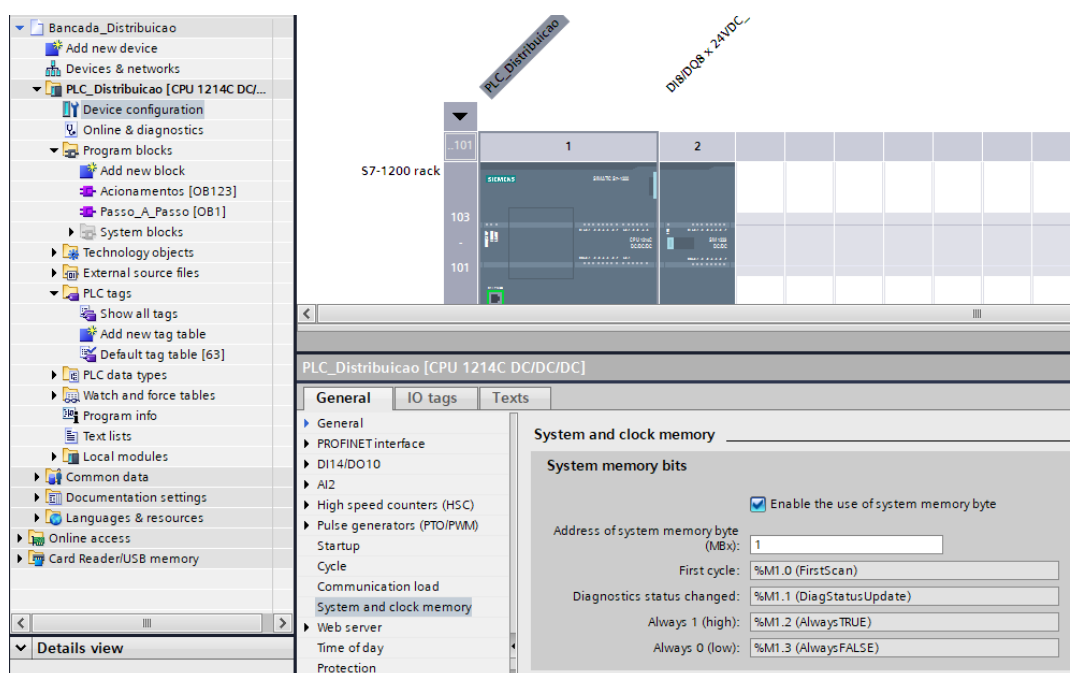


Figura 43 – Habilita Memórias do Sistema (Autor, 2025)

O código foi implementado em dois Blocos de Organização (OBs), conforme ilustrado na Figura 44. Esses blocos correspondem às estruturas onde o código do CLP é armazenado e executado de forma cíclica. As numerações atribuídas aos blocos podem ser definidas pelo programador, não possuindo nenhuma função específica relacionada ao número escolhido.

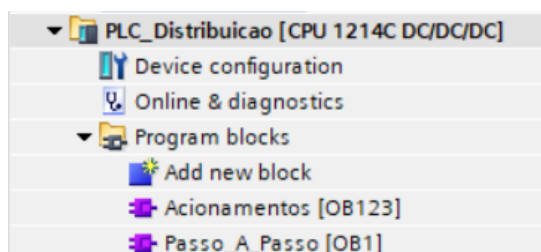


Figura 44 – Blocos de organização (Autor, 2025)

O bloco “Passo_A_Passo” (OB1) é utilizado para realizar a sequência de eventos e a transição entre cada passo, que pode ser visualizado nas figuras 39 e 40. Já no bloco “Acionamentos” (OB123), será feita a atuação de cada saída digital pertinente a cada passo.

Conforme apresentado nas Figuras 45 e 46, encontra-se o código em Texto Estruturado (SCL) que representa o passo a passo da programação da bancada de distribuição.

No Bloco OB1 também foram alocados os temporizadores utilizados nesta bancada, procedimento que foi aplicado de forma semelhante em todas as demais bancadas, conforme pode ser observado nas linhas de código 66 e 69, apresentadas na Figura 46.

```

1 IF "FirstScan" THEN
2     "Passo_0" := 1;
3 END_IF;
4
5 IF "Passo_0" AND "IX_Start" THEN
6     "Passo_0" := 0;
7     "Passo_1" := 1;
8 END_IF;
9
10 IF "Passo_1" AND NOT "IX_Peca_Na_Alimentacao" AND "IX_Empurrador_Recuado" THEN
11     "Passo_2" := 1;
12     "Passo_1" := 0;
13 END_IF;
14
15 IF "Passo_2" THEN
16     #Inicia_Temp_Avanca_Empurrador := 1;
17 IF "Temp_Avanca_Empurrador".Q THEN
18     "Passo_3" := 1;
19     #Inicia_Temp_Avanca_Empurrador := 0;
20     "Passo_2" := 0;
21 END_IF;
22 END_IF;
23
24 IF "Passo_3" AND "IX_Posicionador_Peca_Recuado" THEN
25     "Passo_4" := 1;
26     "Passo_3" := 0;
27 END_IF;
28
29 IF "Passo_4" THEN
30     #Inicia_Temp_Espera := 1;
31 IF "Temp_Espera".Q AND NOT "IX_Solicitando_Peca" THEN
32     "Passo_5" := 1;
33     #Inicia_Temp_Espera := 0;
34     "Passo_4" := 0;
35 END_IF;
36 END_IF;
37
38 IF "Passo_5" AND "IX_Posicionador_Peca_Avancado" THEN
39     "Passo_6" := 1;
40     "Passo_5" := 0;
41 END_IF;
42

```

Figura 45 – Código SCL Passo a Passo Bancada de Distribuição (Autor, 2025)

```

43 IF "Passo_6" THEN
44     #Inicia_Temp_Espera := 1;
45 IF "Temp_Espera".Q THEN
46     "Passo_7" := 1;
47     #Inicia_Temp_Espera := 0;
48     "Passo_6" := 0;
49 END_IF;
50 END_IF;
51
52 IF "Passo_7" AND "IX_Posicionador_Posicao_Inicial" THEN
53     "Passo_8" := 1;
54     "Passo_7" := 0;
55 END_IF;
56
57 IF "Passo_8" THEN
58     #Inicia_Temp_Espera := 1;
59 IF "Temp_Espera".Q THEN
60     "Passo_1" := 1;
61     #Inicia_Temp_Espera := 0;
62     "Passo_8" := 0;
63 END_IF;
64 END_IF;
65
66 "Temp_Avanca_Empurrador".TON(IN:=#Inicia_Temp_Avanca_Empurrador,
67                               PT:=T#3S);
68
69 "Temp_Espera".TON(IN:=#Inicia_Temp_Espera,
70                   PT:=T#2S);
71

```

Figura 46 – Código SCL Passo a Passo Bancada de Distribuição (Autor, 2025)

Como discutido anteriormente na apresentação geral do método passo a passo, é possível observar na figura 47 a aplicação de sua lógica na bancada de distribuição. Nesse exemplo, o passo anterior $M(n-1)$ corresponde ao passo 3, a transição Tn é representada pelo sinal do sensor do posicionador de peça recuado, o passo atual Mn corresponde ao passo 4 e o próximo passo $M(n+1)$ está associado ao passo 5. Essa relação evidencia a aderência entre a teoria do método e sua implementação prática no processo automatizado.

```

IF "M(n-1)" AND "Tn" THEN
    "Mn" := 1;
END_IF;

IF "M(n+1)" OR "RS" THEN
    "Mn" := 0;
END_IF;

IF "Passo_3" AND "IX_Posicionador_Peca_Recuado" THEN
    "Passo_4" := 1;
    "Passo_3" := 0;
END_IF;

IF "Passo_4" THEN
    #Inicia_Temp_Espera := 1;
    IF "Temp_Espera".Q AND NOT "IX_Solicitando_Peca" THEN
        "Passo_5" := 1;
        #Inicia_Temp_Espera := 0;
        "Passo_4" := 0;
    END_IF;
END_IF;

```

Figura 47 – Método Passo a Passo Aplicado (Autor, 2025)

A Figura 48 apresenta o bloco de acionamentos, que é responsável pelo controle das saídas digitais da bancada.

```

1 IF "Passo_0" THEN
2     // Passo de espera
3     ;
4 END_IF;
5
6 IF "Passo_1" THEN
7     ;
8 END_IF;
9
10 IF "Passo_2" THEN
11     "QX_Empurrador" := 1;
12 END_IF;
13
14 IF "Passo_3" THEN
15     "QX_Empurrador" := 0;
16     "QX_Posicionador_Peca_Recua" := 1;
17 END_IF;
18
19 IF "Passo_4" THEN
20     "QX_Posicionador_Peca_Recua" := 0;
21     "QX_Ventosa_Puxa" := 1;
22 END_IF;
23
24 IF "Passo_5" THEN
25     "QX_Posicionador_Peca_Avanca" := 1;
26 END_IF;
27
28 IF "Passo_6" THEN
29     "QX_Posicionador_Peca_Avanca" := 0;
30     "QX_Ventosa_Puxa" := 0;
31     "QX_Ventosa_Solta" := 1;
32 END_IF;
33
34 IF "Passo_7" THEN
35     "QX_Posicionador_Peca_Recua" := 1;
36     "QX_Ventosa_Solta" := 0;
37 END_IF;
38
39 IF "Passo_8" THEN
40     "QX_Posicionador_Peca_Recua" := 0;
41 END_IF;

```

Figura 48 – Código SCL Acionamentos Bancada de Distribuição (Autor, 2025)

4.2 BANCADA DE TESTE

Apresenta-se nesta seção a bancada de teste. Essa estação desempenha um papel fundamental no processo, sendo responsável por executar as verificações necessárias antes do avanço da sequência automatizada das demais bancadas do sistema. Sua estrutura pode ser visualizada na figura 49.

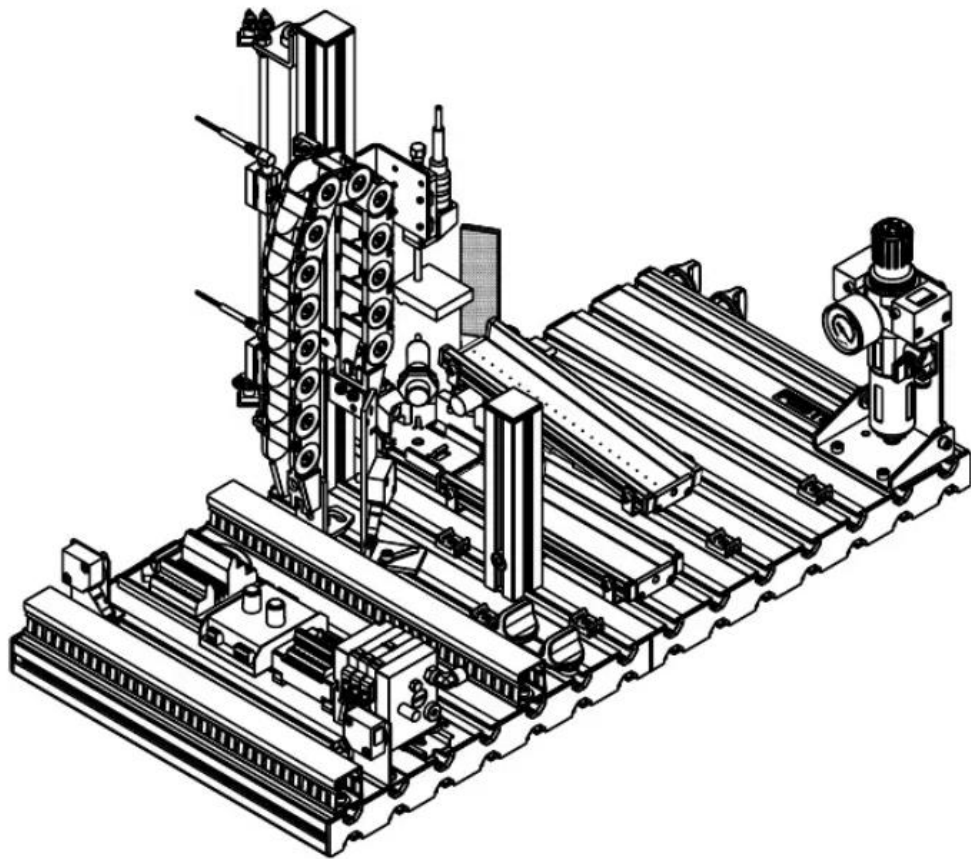


Figura 49 – Bancada de Teste (Festo, 2025)

4.2.1 IO'S BANCADA DE TESTE

Na Tabela 2 são listadas as entradas e saídas da bancada de teste, distribuídas de maneira a facilitar a identificação e o entendimento dos dispositivos que compõem o processo.

Tipo	Endereço no CLP	Descrição do Dispositivo	Nome da Tag	Tipo de Sinal
Entrada	I8.0	Botão de partida	Botao_Start	Digital
Entrada	I0.0	Sensor chegou peça	IX_Sensor_Chegou_Peca	Digital
Entrada	I0.1	Sensor elevador em cima	IX_Sensor_Elevador_Cima	Digital
Entrada	I0.2	Sensor elevador embaixo	IX_Sensor_Elevador_Baixo	Digital
Entrada	I0.3	Sensor presença posicionador	IX_Sensor_Presenca_Posicionador	Digital
Entrada	I0.5	Sensor empurrador avançado	IX_Sensor_Empurrador_Avançado	Digital
Entrada	I0.7	Bancada solicitando peça	IX_Bancada_Solicitando_Peca	Digital
Saída	Q0.2	Empurrador	QX_Empurrador	Digital
Saída	Q0.3	Sopro desliza peça	QX_Sopro_Desliza_Peca	Digital
Saída	Q0.7	Pede peça	QX_Pede_Peca	Digital
Saída	Q1.0	Sobe elevador	QX_Sobe_Elevador	Digital
Saída	Q1.1	Desce elevador	QX_Desce_Elevador	Digital

Tabela 2 - IO's Bancada de Teste (Autor, 2025)

4.2.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA

Na figura 50 é apresentada a disposição das entradas e saídas da bancada de teste, permitindo identificar com precisão a localização dos dispositivos que compõem esta estação.

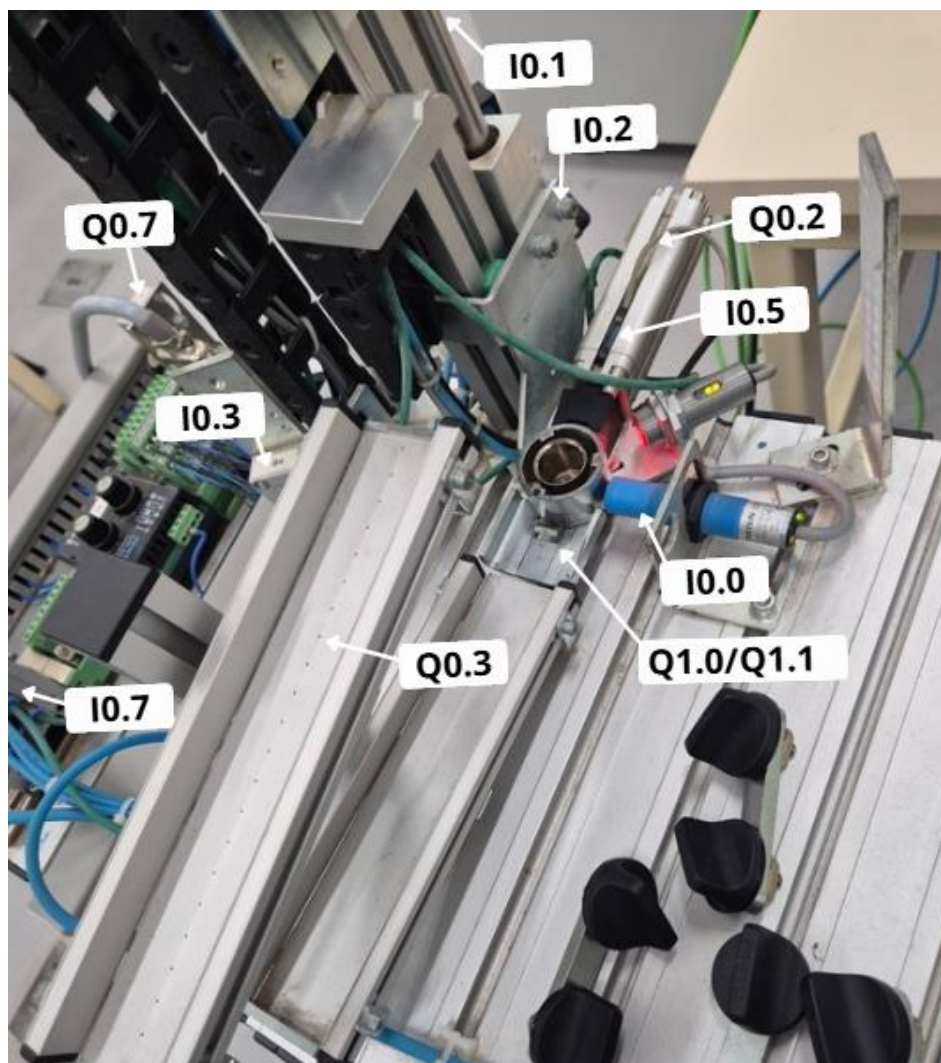


Figura 50 – IO's na Bancada Física (Autor, 2025)

4.2.3 APLICAÇÃO DO MÉTODO PASSO A PASSO

Conforme apresentado na Figura 51, o funcionamento passo a passo da bancada de teste segue uma lógica semelhante à primeira estação do sistema. No início do processo, o CLP é energizado e permanece no passo 0, aguardando o acionamento do botão de partida (I8.0). Quando o operador pressiona o botão, ocorre a transição para o passo 1.

A passagem do passo 1 para o passo 2 acontece mediante três condições simultâneas: a chegada da peça à bancada, detectada pelo sensor de chegada de peça (I0.0), o acionamento do sensor de elevador na posição inferior (I0.2) e a confirmação de que o posicionador da bancada de distribuição não se encontra

presente, por meio do sensor de presença do posicionador (I0.3). Atendidas essas condições, o elevador é acionado para subir.

Ao atingir o sensor de elevador na posição superior (I0.1) e havendo solicitação de peça pela bancada de processamento (I0.7), ocorre a transição para o passo 3, no qual o empurrador é ativado. Com a confirmação do empurrador avançado pelo sensor correspondente (I0.5), o sistema avança para o passo 4, etapa em que o sopro é acionado para deslocar a peça, iniciando também um tempo de espera programado.

Encerrado o tempo de sopro, o processo evolui para o passo 5, responsável por realizar a descida do elevador. Quando o sensor de elevador na posição inferior (I0.2) é novamente detectado, o ciclo é reiniciado.

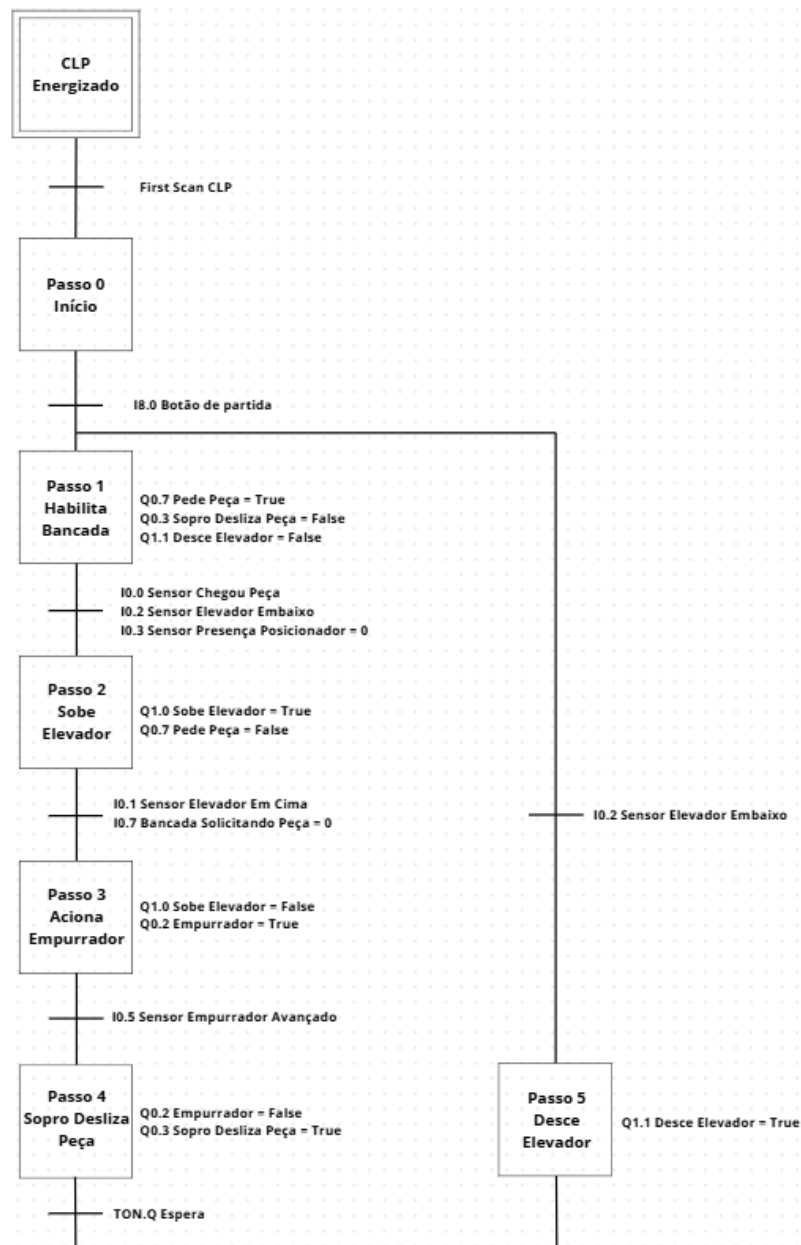


Figura 51 – Passo a Passo Bancada de Teste (Autor, 2025)

4.2.4 IMPLEMENTAÇÃO NO TIA PORTAL

Como pode ser observado nas Figuras 52 e 53, o código em Texto Estruturado (SCL) implementa o método passo a passo utilizado na programação da bancada de teste.

```
1 IF "FirstScan" THEN
2   "Passo_0" := 1;
3 END_IF;
4
5 IF "Passo_0" AND "Botao_Start" THEN
6   "Passo_1" := 1;
7   "Passo_0" := 0;
8 END_IF;
9
10 IF "Passo_1" AND "IX_Sensor_Chegou_Peca" AND "IX_Sensor_Elevador_Baixo" AND NOT "IX_Sensor_Presenca_Posicionador" THEN
11   "Passo_2" := 1;
12   "Passo_1" := 0;
13 END_IF;
14
15 IF "Passo_2" AND "IX_Sensor_Elevador_Cima" AND NOT "IX_Bancada_Solicitando_Peca" THEN
16   "Passo_3" := 1;
17   "Passo_2" := 0;
18 END_IF;
19
20 IF "Passo_3" AND "IX_Sensor_Empurrador_Avancado" THEN
21   "Passo_4" := 1;
22   "Passo_3" := 0;
23 END_IF;
24
25 IF "Passo_4" THEN
26   #Inicia_Temp_Espera := 1;
27 IF "Temp_Espera".Q THEN
28   "Passo_5" := 1;
29   "Passo_4" := 0;
30 END_IF;
31 END_IF;
```

Figura 52 – Código SCL Passo a Passo Bancada de Teste (Autor, 2025)

```
33 IF "Passo_5" AND "IX_Sensor_Elevador_Baixo" THEN
34   #Inicia_Temp_Espera := 0;
35   "Passo_1" := 1;
36   "Passo_5" := 0;
37 END_IF;
38
39 "Temp_Espera".TON(IN:=#Inicia_Temp_Espera,
40   PT:=T#3s);
41
```

Figura 53 – Código SCL Passo a Passo Bancada de Teste (Autor, 2025)

A Figura 54 ilustra o bloco de acionamentos, encarregado do controle das saídas digitais da bancada.

```
1 IF "Passo_1" THEN
2   "QX_Pede_Peca" := 1;
3   "QX_Sopro_Desliza_Peca" := 0;
4   "QX_Desce_Elevador" := 0;
5 END_IF;
6
7 IF "Passo_2" THEN
8   "QX_Sobe_Elevador" := 1;
9   "QX_Pede_Peca" := 0;
10 END_IF;
11
12 IF "Passo_3" THEN
13   "QX_Sobe_Elevador" := 0;
14   "QX_Empurrador" := 1;
15 END_IF;
16
17 IF "Passo_4" THEN
18   "QX_Empurrador" := 0;
19   "QX_Sopro_Desliza_Peca" := 1;
20 END_IF;
21
22 IF "Passo_5" THEN
23   "QX_Desce_Elevador" := 1;
24 END_IF;
```

Figura 54 – Código SCL Acionamentos Bancada de Distribuição (Autor, 2025)

4.3 BANCADA DE PROCESSAMENTO

A Figura 55 mostra o modelo da bancada de processamento e suas características físicas. Essa bancada é responsável pelas operações de inspeção e classificação das peças, integrando sensores e atuadores que permitem identificar características do produto e direcioná-lo adequadamente dentro do sistema automatizado.

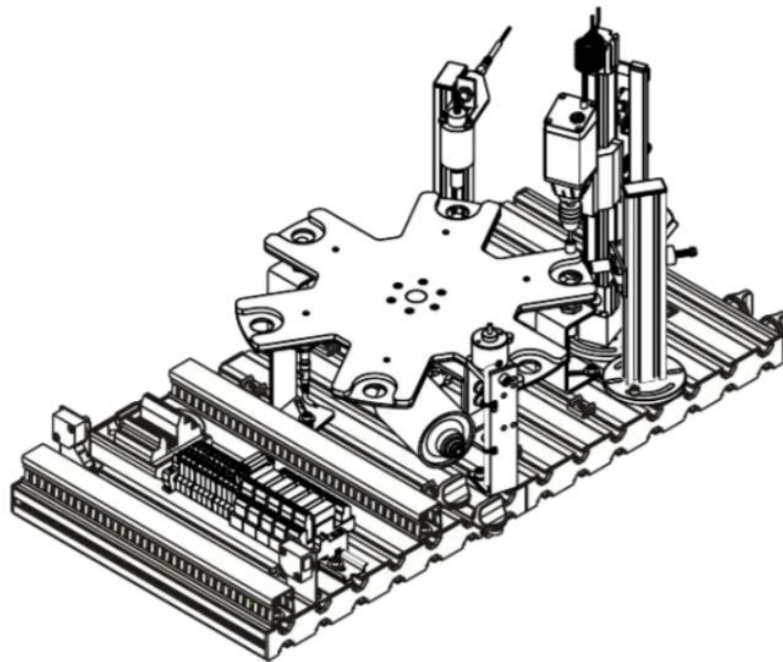


Figura 55 – Bancada de Processamento (Festo, 2025).

4.3.1 IO'S BANCADA DE PROCESSAMENTO

Conforme apresentado na Tabela 3, estão organizadas as entradas e saídas correspondentes à bancada de processamento.

Tipo	Endereço no CLP	Descrição do Dispositivo	Nome da Tag	Tipo de Sinal
Entrada	I8.0	Botão de partida	Botao_Start	Digital
Entrada	I0.0	Sensor peça na entrada	Sensor_Peca_Entrada	Digital
Entrada	I0.1	Sensor peça posição furadeira	Sensor_Peca_Posicao_Furadeira	Digital
Entrada	I0.2	Sensor peça posição teste	Sensor_Peca_Posicao_Testes	Digital
Entrada	I0.5	Sensor rotação completa	Sensor_Rotacao_Completa	Digital
Entrada	I0.6	Cilindro de teste avançado	Sensor_Cil_Testes_Avançado	Digital
Entrada	I0.7	Bancada solicitando peça	Sinal_Bancada_Separacao	Digital
Saída	Q0.0	Liga furadeira	Liga_Furadeira	Digital
Saída	Q0.1	Motor rotação	Motor_Rotacao	Digital
Saída	Q0.2	Desce furadeira	Desce_Furadeira	Digital
Saída	Q0.3	Sobe furadeira	Sobe_Furadeira	Digital
Saída	Q0.4	Cilindro estampador	Cil_Estampador	Digital
Saída	Q0.5	Cilindro teste	Cil_Testes	Digital
Saída	Q0.6	Ejetor de peça	Ejetor_Peca	Digital
Saída	Q0.7	Sinal pede peça	Sinal_Pede_Peca	Digital

Tabela 3 - IO's Bancada Processamento (Autor, 2025)

4.3.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA

A Figura 56 ilustra as entradas e saídas da bancada de processamento, juntamente com a forma como estão distribuídas fisicamente no sistema.

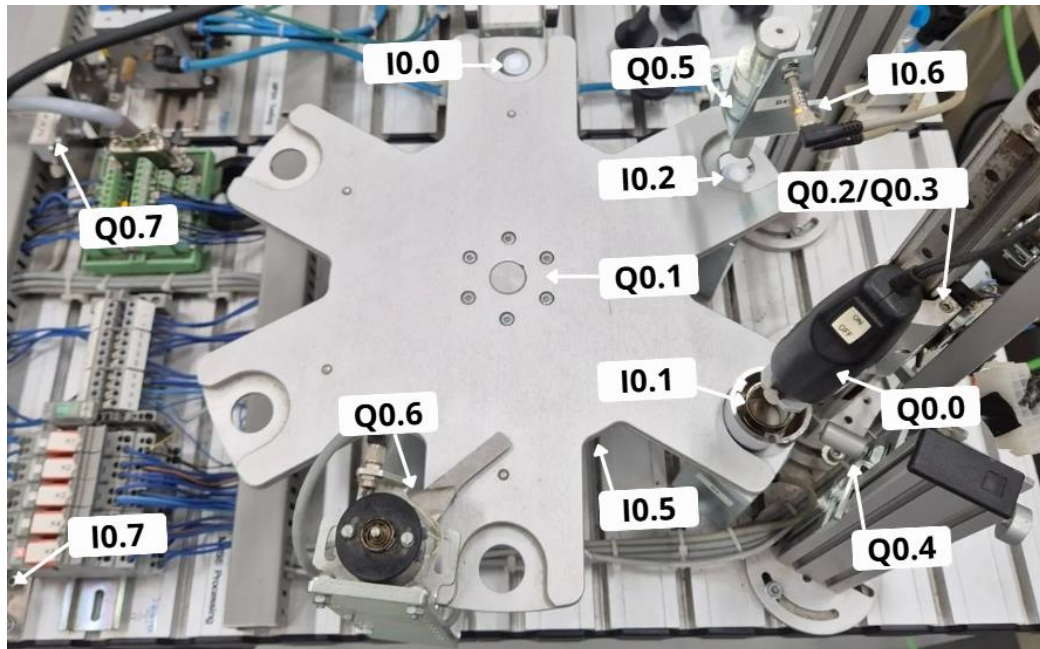


Figura 56 – IO's Bancada Processamento (Autor, 2025)

4.3.3 APLICAÇÃO DO MÉTODO PASSO A PASSO

Observando a Figura 57, é possível visualizar o método passo a passo aplicado à bancada de processamento. O ciclo inicia-se com a energização do CLP, que leva o sistema ao passo 0, permanecendo em espera até o acionamento do botão de partida (I8.0).

No passo 1, a bancada ativa o sinal de pedido de peça. Quando o sensor de chegada de peça é acionado (IO.0), ocorre a transição para o passo 2, no qual a mesa rotativa é acionada. Após o tempo de rotação e com a detecção simultânea do sinal de rotação completa (IO.5) e da presença de peça na posição de teste (IO.2), o sistema avança para o passo 3, onde o cilindro de teste é acionado e mantém-se avançado durante um tempo programado.

Concluído esse tempo e com a confirmação do sinal de cilindro de teste avançado (IO.6), o sistema passa para o passo 4, realizando uma nova rotação

da mesa até que o sensor indique a presença da peça na posição da furadeira (I0.1).

No passo 5, a furadeira é avançada e permanece nessa posição por um tempo determinado. Encerrado esse tempo, ocorre a passagem para o passo 6, momento em que a furadeira é acionada e é realizada a contagem do tempo de furação.

Ao atingir o tempo programado, o sistema evolui para o passo 7, efetuando o recuo da furadeira e aguardando o tempo correspondente à sua retração. Com a confirmação do recuo completo, a mesa é novamente acionada no passo 8, avançando até a detecção do sinal de rotação completa (I0.5), o que, em conjunto com o término do tempo de rotação, possibilita a entrada para o passo 9. Nesse passo, o sistema permanece aguardando o sinal de solicitação de peça da bancada seguinte (I0.7).

Quando a próxima bancada solicita a peça (I0.7), o sistema avança para o passo 10, realizando a ejeção da peça para a etapa subsequente. Após o tempo de acionamento da ejeção, o ciclo retorna ao passo 1, reiniciando o processo.

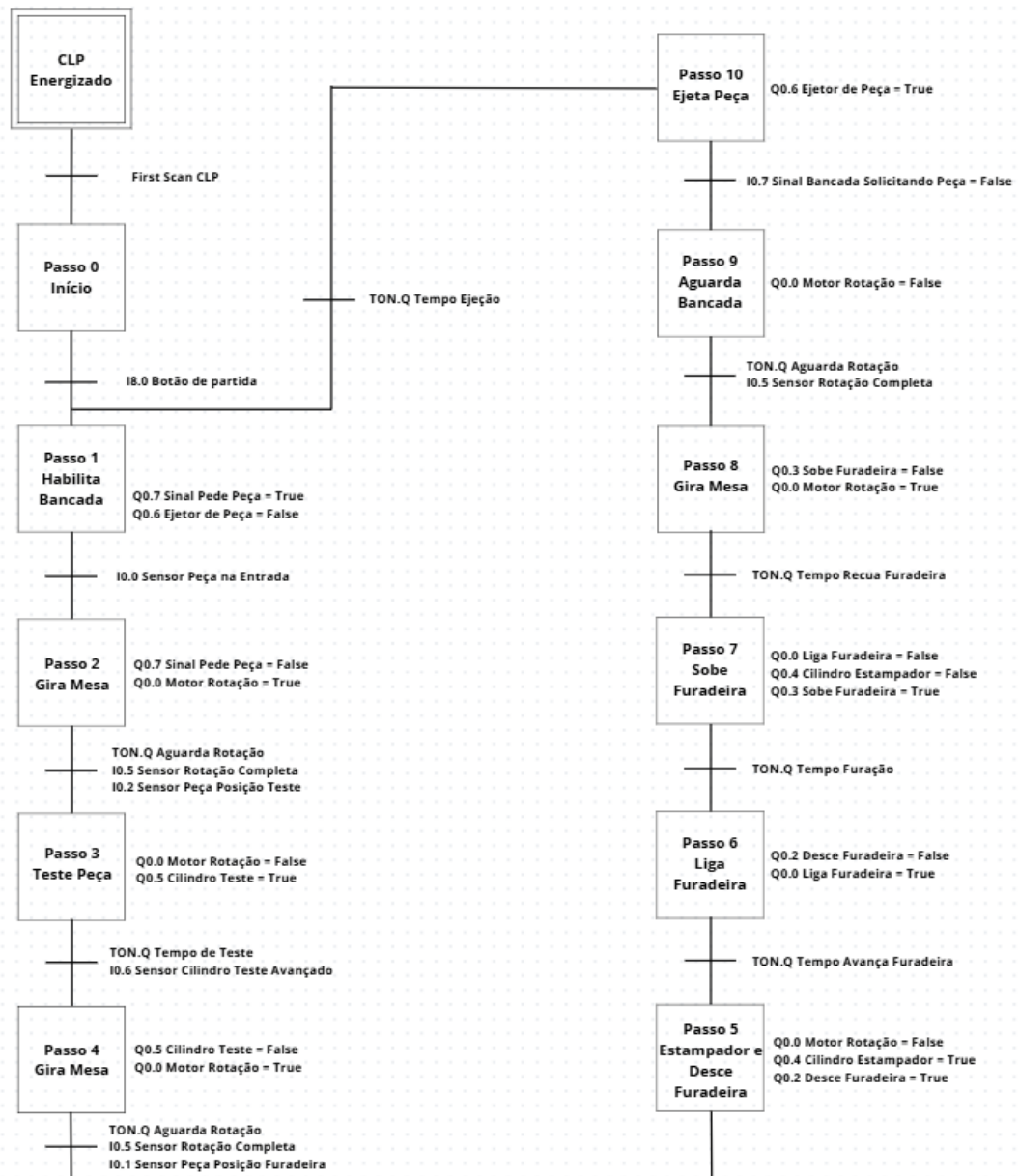


Figura 57 – Método Passo a Passo Bancada de Processamento (Autor, 2025)

4.3.4 IMPLEMENTAÇÃO NO TIA PORTAL

As Figuras 58, 59, 60 e 61 apresentam os trechos de código que implementam logicamente as transições entre os passos do método passo a passo aplicado à bancada de processamento, evidenciando a estrutura de controle adotada no CLP.

```
1 IF "FirstScan" THEN
2   "Passo_0" := 1;
3 END_IF;
4
5 IF "Passo_0" AND "Botao_Start" THEN
6   "Passo_1" := 1;
7   "Passo_0" := 0;
8 END_IF;
9
10 IF "Passo_1" AND "Sensor_Peca_Entrada" THEN
11   "Passo_2" := 1;
12   "Passo_1" := 0;
13 END_IF;
14
15 IF "Passo_2" THEN
16   #Inicia_Temp_Aguarda_Rotacao := 1;
17 IF "Temp_Aguarda_Rotacao".Q AND "Sensor_Rotacao_Completa" AND "Sensor_Peca_Posicao_Testes" THEN
18   "Passo_3" := 1;
19   "Passo_2" := 0;
20 END_IF;
21 END_IF;
22
23 IF "Passo_3" THEN
24   #Inicia_Temp_Aguarda_Rotacao := 0;
25   #Inicia_Temp_Testes := 1;
26 IF "Temp_Testes".Q AND "Sensor_Cil_Testes_Avancado" THEN
27   "Passo_4" := 1;
28   "Passo_3" := 0;
29 END_IF;
30 END_IF;
```

Figura 58 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)

```

32 IF "Passo_4" THEN
33     #Inicia_Temp_Teste := 0;
34     #Inicia_Temp_Aguarda_Rotacao := 1;
35 IF "Temp_Aguarda_Rotacao".Q AND "Sensor_Rotacao_Completa" AND "Sensor_Peca_Posicao_Furadeira" THEN
36     "Passo_5" := 1;
37     "Passo_4" := 0;
38 END_IF;
39 END_IF;
40
41 IF "Passo_5" THEN
42     #Inicia_Temp_Aguarda_Rotacao := 0;
43     #Inicia_Temp_Avanco_Furadeira := 1;
44 IF "Temp_Avanco_Furadeira".Q THEN
45     "Passo_6" := 1;
46     #Inicia_Temp_Avanco_Furadeira := 0;
47     "Passo_5" := 0;
48 END_IF;
49 END_IF;
50
51 IF "Passo_6" THEN
52     #Inicia_Temp_Furadeira := 1;
53 IF "Temp_Furadeira".Q THEN
54     "Passo_7" := 1;
55     #Inicia_Temp_Furadeira := 0;
56     "Passo_6" := 0;
57 END_IF;
58 END_IF;

```

Figura 59 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)

```

60 IF "Passo_7" THEN
61     #Inicia_Temp_Recuo_Furadeira := 1;
62 IF "Temp_Recuo_Furadeira".Q THEN
63     "Passo_8" := 1;
64     #Inicia_Temp_Recuo_Furadeira := 0;
65     "Passo_7" := 0;
66 END_IF;
67 END_IF;
68
69 IF "Passo_8" THEN
70     #Inicia_Temp_Aguarda_Rotacao := 1;
71 IF "Temp_Aguarda_Rotacao".Q AND "Sensor_Rotacao_Completa" THEN
72     "Passo_9" := 1;
73     "Passo_8" := 0;
74 END_IF;
75 END_IF;
76
77 IF "Passo_9" THEN
78     #Inicia_Temp_Aguarda_Rotacao := 0;
79 IF NOT "Sinal_Bancada_Separacao" THEN
80     "Passo_10" := 1;
81     "Passo_9" := 0;
82 END_IF;
83 END_IF;
84
85 IF "Passo_10" THEN
86     #Inicia_Temp_Ejecao := 1;
87 IF "Temp_Ejecao".Q THEN
88     "Passo_1" := 1;
89     #Inicia_Temp_Ejecao := 0;
90     "Passo_10" := 0;
91 END_IF;
92 END_IF;

```

Figura 60 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)

```

94  "Temp_Teste".TON(IN:=#Inicia_Temp_Teste,
95      PT:=T#2s);
96
97  "Temp_Avanco_Furadeira".TON(IN:=#Inicia_Temp_Avanco_Furadeira,
98      PT:=T#1.5s);
99
100 "Temp_Furadeira".TON(IN:=#Inicia_Temp_Furadeira,
101     PT:=T#2s);
102
103 "Temp_Recua_Furadeira".TON(IN:=#Inicia_Temp_Recuo_Furadeira,
104     PT:=T#1s);
105
106 "Temp_Ejecao".TON(IN:=#Inicia_Temp_Ejecao,
107     PT:=T#1s);
108
109 "Temp_Aguarda_Rotacao".TON(IN:=#Inicia_Temp_Aguarda_Rotacao,
110     PT:=T#1s);
111

```

Figura 61 – Código SCL Passo a Passo Bancada de Processamento (Autor, 2025)

As Figuras 62 e 63 apresentam os trechos de código responsáveis pelos acionamentos utilizados no controle da bancada de processamento, evidenciando a lógica aplicada aos atuadores do sistema.

```

1 IF "Passo_1" THEN
2     "Sinal_Pede_Peca" := 1;
3     "Ejetor_Peca" := 0;
4 END_IF;
5
6 IF "Passo_2" THEN
7     "Sinal_Pede_Peca" := 0;
8     "Motor_Rotacao" := 1;
9 END_IF;
10
11 IF "Passo_3" THEN
12     "Motor_Rotacao" := 0;
13     "Cil_Testes" := 1;
14 END_IF;
15
16 IF "Passo_4" THEN
17     "Cil_Testes" := 0;
18     "Motor_Rotacao" := 1;
19 END_IF;
20
21 IF "Passo_5" THEN
22     "Motor_Rotacao" := 0;
23     "Cil_Estampador" := 1;
24     "Desce_Furadeira" := 1;
25 END_IF;
26
27 IF "Passo_6" THEN
28     "Liga_Furadeira" := 1;
29     "Desce_Furadeira" := 0;
30 END_IF;

```

Figura 62 – Código SCL Acionamentos Bancada de Processamento (Autor, 2025)

```

32 IF "Passo_7" THEN
33     "Liga_Furadeira" := 0;
34     "Cil_Estampador" := 0;
35     "Sobe_Furadeira" := 1;
36 END_IF;
37
38 IF "Passo_8" THEN
39     "Sobe_Furadeira" := 0;
40     "Motor_Rotacao" := 1;
41 END_IF;
42
43 IF "Passo_9" THEN
44     "Motor_Rotacao" := 0;
45 END_IF;
46
47 IF "Passo_10" THEN
48     "Motor_Rotacao" := 0;
49     "Ejetor_Peca" := 1;
50 END_IF;

```

Figura 63 – Código SCL Acionamentos Bancada de Processamento (Autor, 2025)

4.4 BANCADA DE MANIPULAÇÃO

A Figura 64 mostra o modelo da bancada de manipulação e suas características físicas. Essa bancada é responsável pelos movimentos de posicionamento e transferência de peças, integrando sensores e atuadores que permitem sua operação dentro do sistema automatizado.

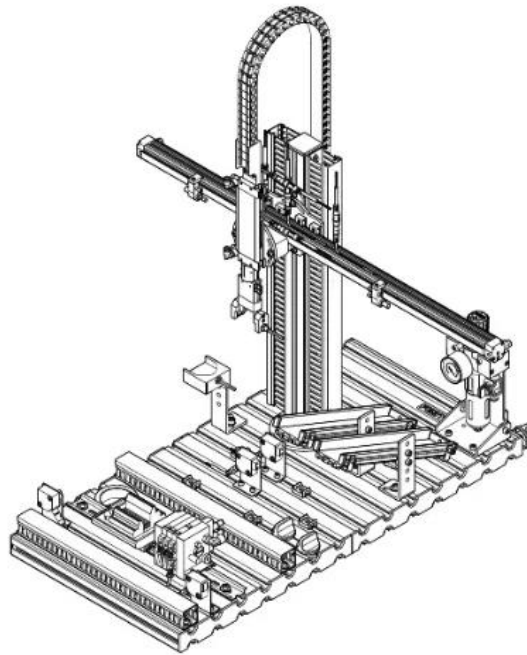


Figura 64 – Bancada de Manipulação (Festo, 2025)

4.4.1 IO'S BANCADA DE MANIPULAÇÃO

Conforme apresentado na Tabela 4, são listadas as entradas e saídas relacionadas à bancada de manipulação.

Tipo	Endereço no CLP	Descrição do Dispositivo	Nome da Tag	Tipo de Sinal
Entrada	I8.0	Botão de partida	Botao_Start	Digital
Entrada	I0.0	Sensor chegou peça	IX_Sensor_Chegou_Peca	Digital
Entrada	I0.1	Fim de curso garra no início	IX_Sensor_Garra_Comeco	Digital
Entrada	I0.2	Fim de curso garra no final	IX_Sensor_Garra_Final	Digital
Entrada	I0.4	Sensor garra embaixo	IX_Sensor_Garra_Embaixo	Digital
Entrada	I0.5	Sensor garra em cima	IX_Sensor_Garra_Em_Cima	Digital
Entrada	I0.6	Sensor peça na garra	IX_Sensor_Peca_Na_Garra	Digital
Entrada	I0.7	Bancada solicitando peça	IX_Bancada_Solicitando_Peca	Digital
Saída	Q0.2	Abaixa/Sobe Garra	QX_Abaixa_Sobe_Garra	Digital
Saída	Q0.3	Fecha/Abre Garra	QX_Fecha_Abre_Garra	Digital
Saída	Q0.4	Recua Garra	QX_Garra_Comeco	Digital
Saída	Q0.5	Avança Garra	QX_Garra_Final	Digital
Saída	Q0.7	Sinal pede peça	QX_Pede_Peca	Digital

Tabela 4 - IO's Bancada Manipulação (Autor, 2025)

4.4.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA

As Figuras 65 e 66 apresentam as entradas e saídas da bancada de manipulação e sua respectiva alocação física.

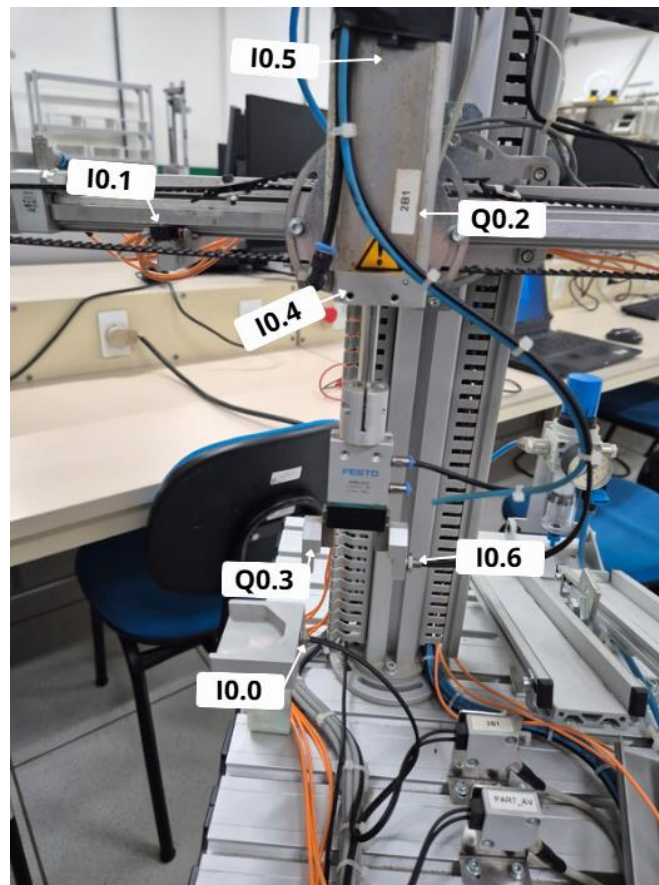


Figura 65 – IO's Bancada Manipulação (Autor, 2025)

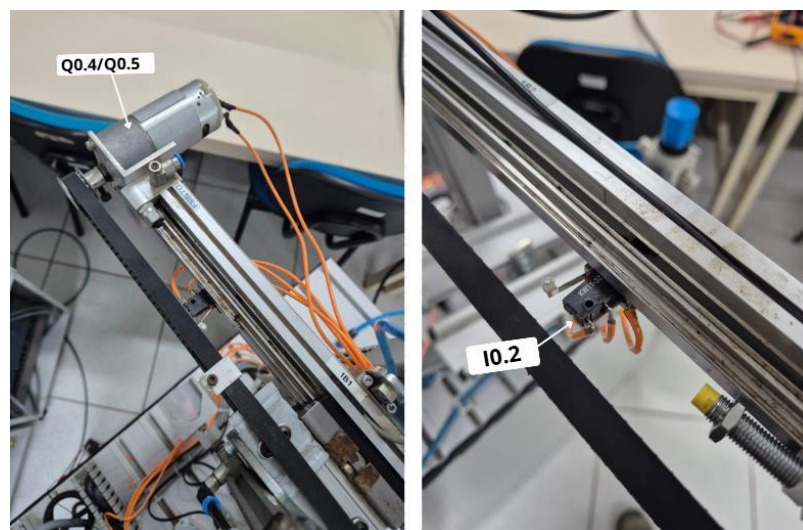


Figura 66 – IO's Bancada Manipulação (Autor, 2025)

4.4.3 APLICAÇÃO DO MÉTODO PASSO A PASSO

Observando a Figura 67, é possível visualizar o método passo a passo aplicado à bancada de manipulação. O ciclo inicia-se com a energização do CLP, passando para o passo 0, no qual o sistema aguarda o acionamento do botão de partida (I8.0).

No passo 1, a saída responsável por recuar a garra é desligada, enquanto é acionada a saída que solicita uma peça à bancada anterior. A transição para o passo 2 ocorre com a detecção da peça pelo sensor de chegada de peça (I0.0), momento em que é acionada a saída que comanda a descida da garra para realizar a coleta.

Com a garra posicionada na parte inferior, confirmado pelo sensor de garra embaixo (I0.4), o sistema avança para o passo 3, realizando o fechamento da garra e a pega da peça. Assim que o sensor de detecção de peça na garra (I0.6) é acionado, o processo segue para o passo 4, no qual a garra é elevada novamente.

Ao atingir a posição superior, indicada pelo sensor de garra em cima (I0.5), o passo 5 é executado, avançando a garra até o final de seu curso. Quando a bancada posterior solicita uma nova peça (I0.7) e o movimento de avanço é concluído, confirmado pelo fim de curso da garra no final (I0.2), o passo 6 é ativado, descendo a garra para depositar a peça.

Com o sensor de garra embaixo (I0.4) acionado, o passo 7 realiza a abertura da garra, liberando a peça. Após a perda da detecção da peça na garra (I0.6), o passo 8 eleva novamente a garra e, no passo 9, a garra retorna à posição inicial, confirmada pelo fim de curso da garra no início (I0.1). Dessa forma, o sistema reinicia o ciclo, voltando ao passo 1, onde aguarda a chegada de uma nova peça para repetir o processo.

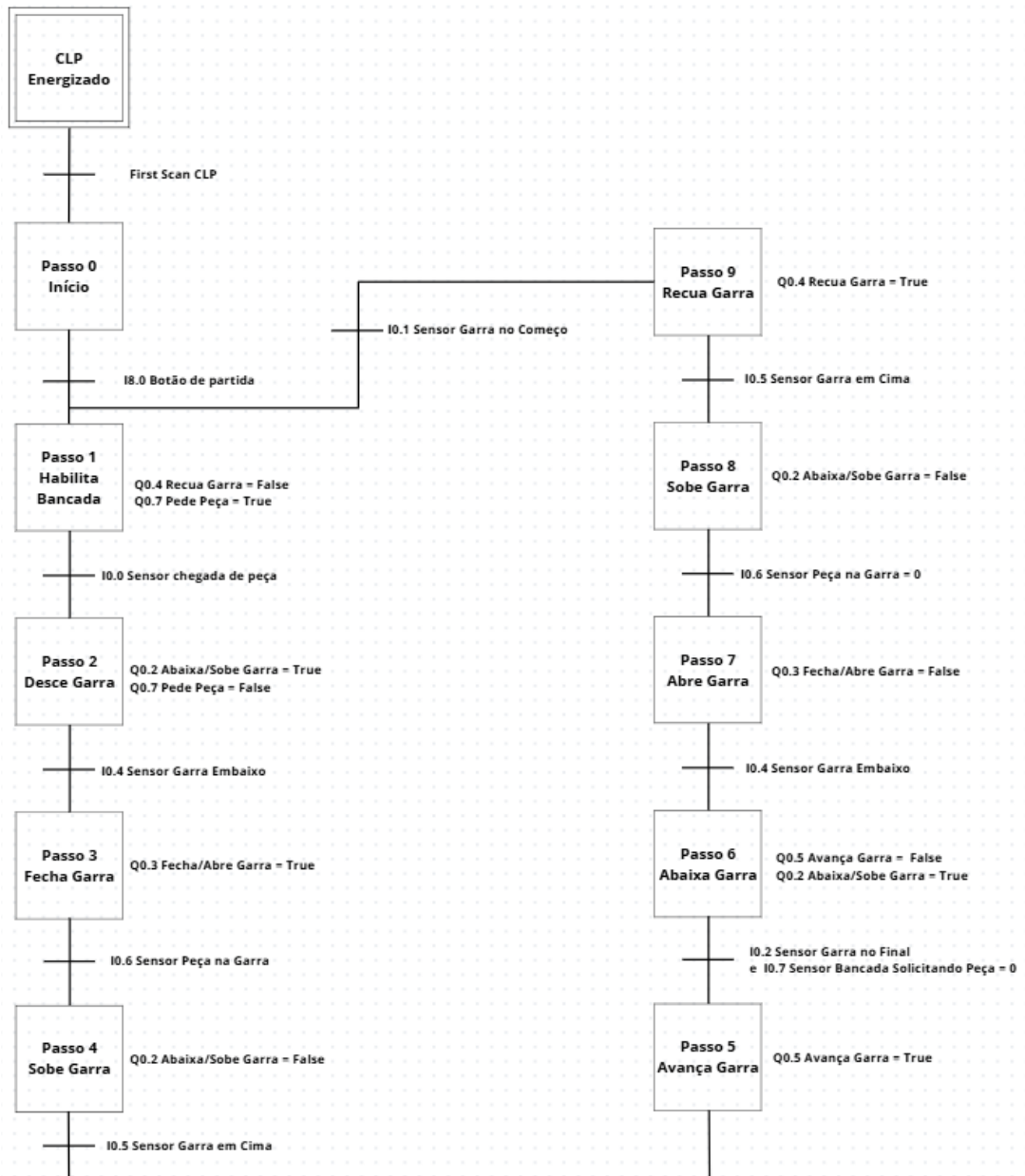


Figura 67 – Método Passo a Passo Bancada de Manipulação (Autor, 2025)

4.4.4 IMPLEMENTAÇÃO NO TIA PORTAL

Nas Figuras 68 e 69, são apresentados os trechos de código responsáveis pela realização das transições entre os passos da bancada de manipulação.

```
1 IF "FirstScan" THEN
2     "Passo_0" := 1;
3 END_IF;
4
5 IF "Passo_0" AND "Botao_Start" THEN
6     "Passo_1" := 1;
7     "Passo_0" := 0;
8 END_IF;
9
10 IF "Passo_1" AND "IX_Sensor_Chegou_Peca" THEN
11     "Passo_2" := 1;
12     "Passo_1" := 0;
13 END_IF;
14
15 IF "Passo_2" THEN
16     IF "IX_Sensor_Garra_Embaixo" THEN
17         "Passo_3" := 1;
18         "Passo_2" := 0;
19     END_IF;
20 END_IF;
21
22 IF "Passo_3" THEN
23     IF "IX_Sensor_Peca_Na_Garra" THEN
24         "Passo_4" := 1;
25         "Passo_3" := 0;
26     END_IF;
27 END_IF;
28
29 IF "Passo_4" THEN
30     IF "IX_Sensor_Garra_Em_Cima" THEN
31         "Passo_5" := 1;
32         "Passo_4" := 0;
33     END_IF;
34 END_IF;
35
36 IF "Passo_5" THEN
37     IF "IX_Sensor_Garra_Final" AND NOT "IX_Bancada_Solicitando_Peca" THEN
38         "Passo_6" := 1;
39         "Passo_5" := 0;
40     END_IF;
41 END_IF;
```

Figura 68 – Código SCL Passo a Passo Bancada de Manipulação (Autor, 2025)

```

43 IF "Passo_6" THEN
44     IF "IX_Sensor_Garra_Embaixo" THEN
45         "Passo_7" := 1;
46         "Passo_6" := 0;
47     END_IF;
48 END_IF;
49
50 IF "Passo_7" THEN
51     IF NOT "IX_Sensor_Peca_Na_Garra" THEN
52         "Passo_8" := 1;
53         "Passo_7" := 0;
54     END_IF;
55 END_IF;
56
57 IF "Passo_8" THEN
58     IF "IX_Sensor_Garra_Em_Cima" THEN
59         "Passo_9" := 1;
60         "Passo_8" := 0;
61     END_IF;
62 END_IF;
63
64 IF "Passo_9" THEN
65     IF "IX_Sensor_Garra_Comeco" THEN
66         "Passo_1" := 1;
67         "Passo_9" := 0;
68     END_IF;
69 END_IF;
70

```

Figura 69 – Código SCL Passo a Passo Bancada de Manipulação (Autor, 2025)

A Figura 70 apresenta o código responsável pelos acionamentos utilizados no controle da bancada.

```
1 IF "Passo_1" THEN
2     "QX_Recua_Garra" := 0;
3     "QX_Pede_Peca" := 1;
4 END_IF;
5
6 IF "Passo_2" THEN
7     "QX_Abaixa_Sobe_Garra" := 1;
8     "QX_Pede_Peca" := 0;
9 END_IF;
10
11 IF "Passo_3" THEN
12     "QX_Fecha_Abre_Garra" := 1;
13 END_IF;
14
15 IF "Passo_4" THEN
16     "QX_Abaixa_Sobe_Garra" := 0;
17 END_IF;
18
19 IF "Passo_5" THEN
20     "QX_Avanca_Garra" := 1;
21 END_IF;
22
23 IF "Passo_6" THEN
24     "QX_Avanca_Garra" := 0;
25     "QX_Abaixa_Sobe_Garra" := 1;
26 END_IF;
27
28 IF "Passo_7" THEN
29     "QX_Fecha_Abre_Garra" := 0;
30 END_IF;
31
32 IF "Passo_8" THEN
33     "QX_Abaixa_Sobe_Garra" := 0;
34 END_IF;
35
36 IF "Passo_9" THEN
37     "QX_Recua_Garra" := 1;
38 END_IF;
39
40
```

Figura 70 – Código SCL Acionamentos Bancada de Manipulação (Autor, 2025)

4.5 BANCADA DE SEPARAÇÃO

Apresenta-se nesta seção a bancada de separação, última etapa do sistema automatizado. Essa estação tem a função de identificar e classificar as peças provenientes das bancadas anteriores, destinando-as conforme suas características e finalizando o processo. Sua estrutura pode ser observada na Figura 71.

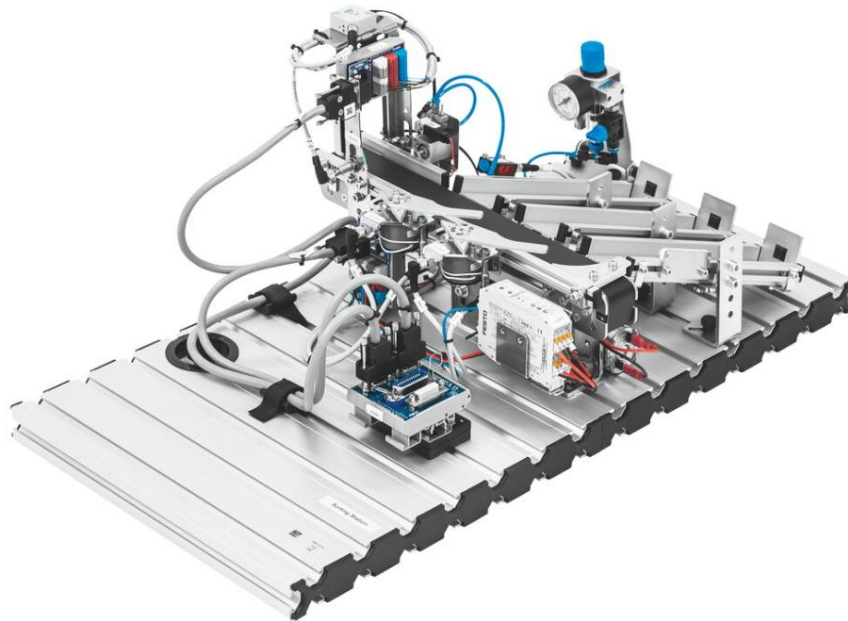


Figura 71 – Bancada de Separação (Festo, 2025)

4.5.1 IO'S BANCADA DE SEPARAÇÃO

Conforme apresentado na Tabela 5, estão organizadas as entradas e saídas referentes à bancada de separação.

Tipo	Endereço no CLP	Descrição do Dispositivo	Nome da Tag	Tipo de Sinal
Entrada	I8.0	Botão de partida	IX_Start	Digital
Entrada	I0.0	Sensor chegada de peça	IX_Peca_Na_Esteira	Digital
Entrada	I0.1	Sensor indutivo identificação de peça metalizada	IX_Peca_Metal	Digital
Entrada	I0.2	Sensor fotoelétrico identificação de cor vermelha	IX_Peca_Vermelha1_Preta0	Digital
Entrada	I0.3	Sensor fotoelétrico peça separada	IX_Sensor_Peca_Separada	Digital
Entrada	I0.4	Sensor cilindro separa peça vermelha recuado	IX_Valv_1_Recuado	Digital
Entrada	I0.5	Sensor cilindro separa peça vermelha avançado	IX_Valv_1_Avançado	Digital
Entrada	I0.6	Sensor cilindro separa peça metálica recuado	IX_Valv_2_Recuado	Digital
Entrada	I0.7	Sensor cilindro separa peça metálica avançado	IX_Valv_2_Avançado	Digital
Saída	Q0.0	Motor elétrico esteira	QX_Esteira	Digital
Saída	Q0.1	Cilindro separa peça vermelha	QX_Cil_Vermelha	Digital
Saída	Q0.2	Cilindro separa peça metalizada	QX_Cil_Metal	Digital
Saída	Q0.3	Cilindro libera peça	QX_Libera_Peca	Digital
Saída	Q0.7	Sinal para bancada anterior	QX_Prox_Estacao	Digital

Tabela 5 - IO's Bancada Separação (Autor, 2025)

4.5.2 LOCALIZAÇÃO DAS ENTRADAS E SAÍDAS NA BANCADA

De acordo com as figuras 72 e 73, são vistos fisicamente onde estão os sensores e atuadores dessa bancada.

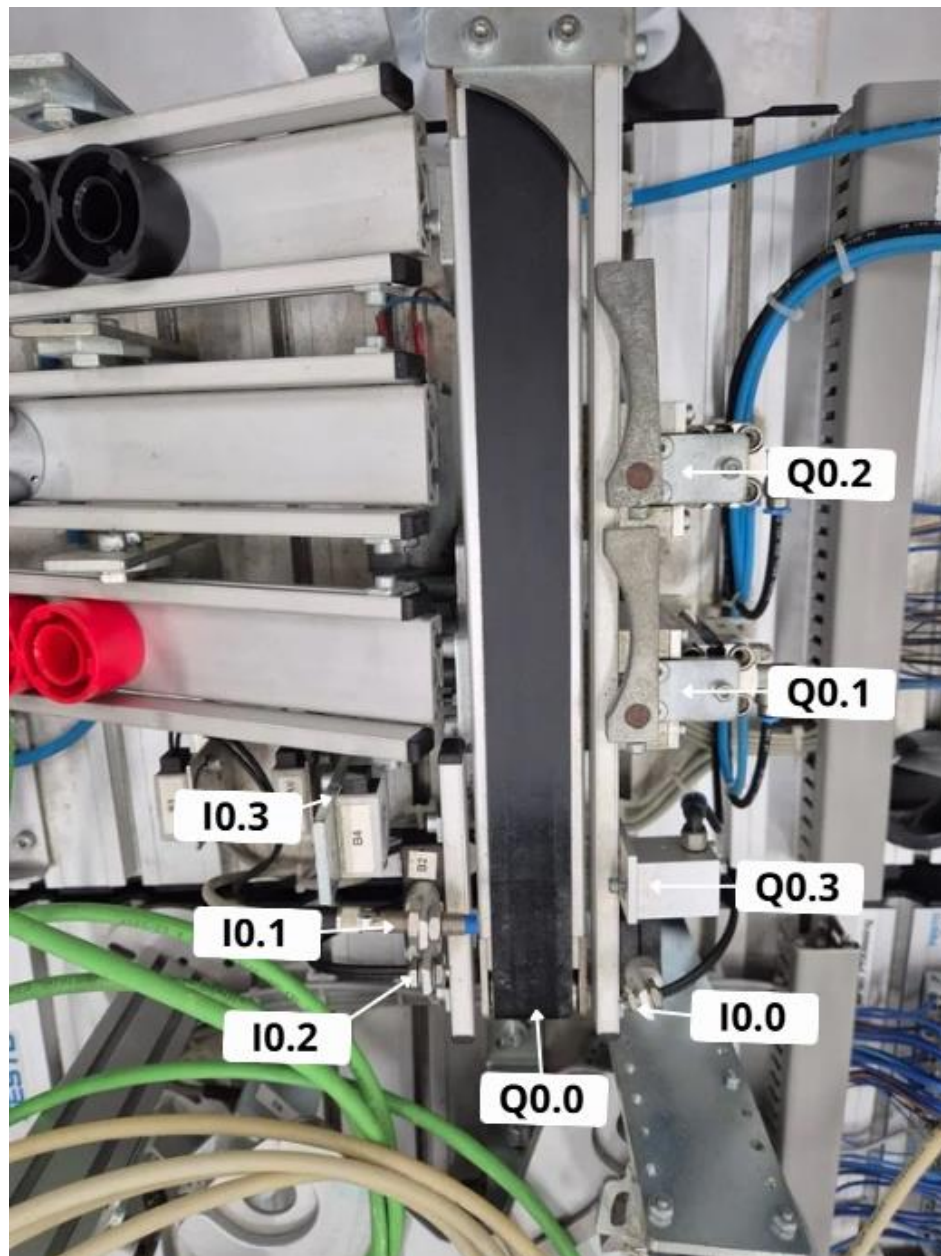


Figura 72 – IO's na Bancada Física (Autor, 2025)

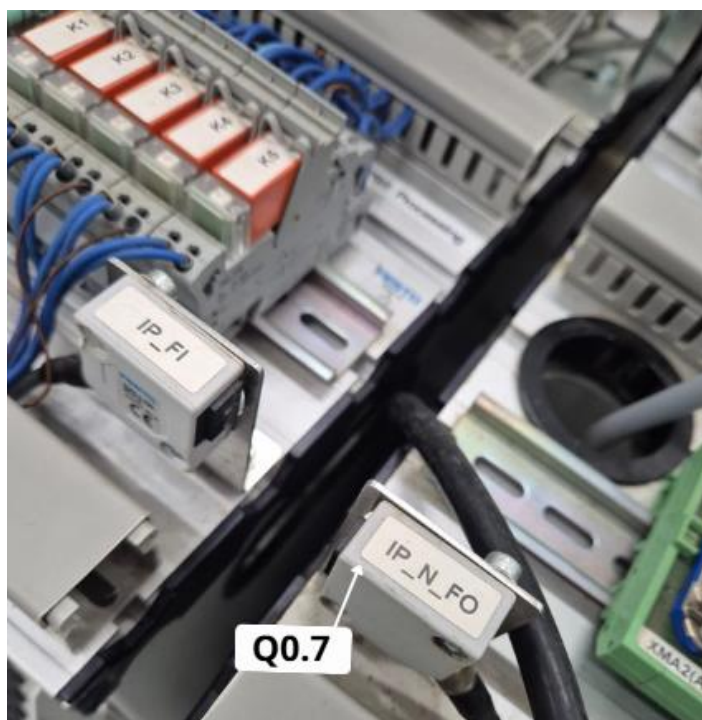


Figura 73 – IO's na Bancada Física (Autor, 2025)

4.5.3 APLICAÇÃO DO MÉTODO PASSO A PASSO

Na Figura 74, foi aplicado o método passo a passo na bancada de separação da Festo. A seguir, é apresentada a sequência de passos definida para esta bancada.

Ao energizar o CLP, a memória que identifica o primeiro ciclo é acionada, levando o sistema ao passo 0. O passo 0 corresponde à etapa inicial do processo, na qual o sistema permanece em estado de espera, aguardando o acionamento do botão de partida (I8.0) localizado na bancada. Ao receber esse sinal, o ciclo de separação de peças é iniciado, ocorrendo a transição para o passo 1. Nessa etapa, é ativada uma saída digital responsável por enviar um sinal à bancada anterior (Q0.7), informando que o sistema está pronto para receber uma nova peça.

Ao detectar o sinal de chegada da peça por meio do sensor de chegada (I0.0), o sistema passa ao passo 2, no qual a esteira transportadora é acionada (Q0.0), dando início à contagem de um temporizador interno. Após o término do

tempo determinado, o sistema realiza a identificação do tipo de peça, de acordo com os sensores acionados.

Caso a peça identificada seja vermelha, o sensor fotoelétrico de identificação de cor vermelha (I0.2) será acionado, conduzindo o sistema ao passo 3. Se a peça for metálica, ela será detectada pelo sensor indutivo de identificação de peça metalizada (I0.1), permitindo a transição para o passo 4. Caso a peça seja preta, nenhum dos sensores de identificação de peça vermelha (I0.2) ou metálica (I0.1) será acionado, o que, por consequência, leva o sistema ao passo 5.

Após a identificação da peça na esteira, o cilindro responsável pela liberação e direcionamento da peça é acionado (Q0.3). Dependendo do tipo de peça identificado, a saída digital correspondente é ativada para comandar o cilindro de separação: o cilindro separa peça vermelha (Q0.1) ou o cilindro separa peça metalizada (Q0.2). No caso específico da peça preta, nenhum cilindro de separação é acionado, permitindo que a peça siga diretamente até sua posição de destino.

Por fim, ao ser acionado o sensor fotoelétrico de peça separada (I0.3), o ciclo é reiniciado automaticamente, retornando ao passo 1, com o reenvio do sinal de “aguardando peça” à bancada anterior (Q0.7).

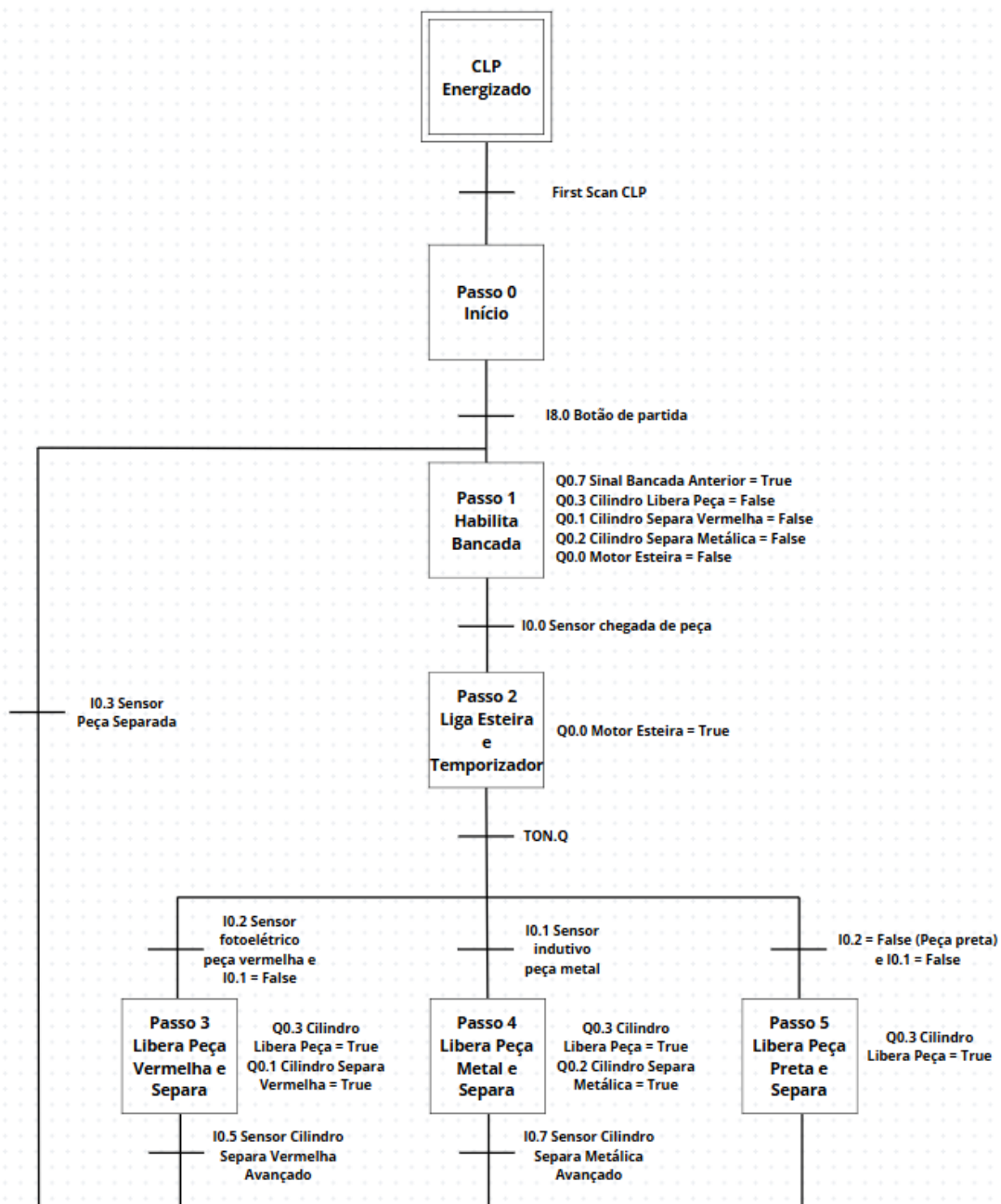


Figura 74 – Passo a passo bancada de separação (Autor, 2025)

4.5.4 IMPLEMENTAÇÃO NO TIA PORTAL

As Figuras 75 e 76 apresentam os trechos de código que implementam as condições de avanço entre os passos do método aplicado à bancada de separação, evidenciando sua lógica de controle sequencial.

```
1 //First scan do clp vai para o passo 0
2 IF "FirstScan" THEN
3   "Passo_0" := 1;
4 END_IF;
5
6 //Está no passo 0 e aguarda o start do botão
7 IF "IX_Start" THEN
8   "Passo_0" := 0;
9   "Passo_1" := 1;
10 END_IF;
11
12 //Passo 1 aguarda sensor chegada de peça na esteira
13 IF "Passo_1" AND "IX_Peca_Na_Esteira" THEN
14   "Passo_2" := 1;
15   "Passo_1" := 0;
16 END_IF;
17
18 //Passo 2 converge para o passo 3, 4 ou 5 de acordo com a peça
19 IF "Passo_2" THEN
20   #Inicia_Temp_Peca_Presa := 1;
21   IF "Temp_Peca_Presa".Q AND "IX_Peca_Vermelhal_Preta0" AND NOT "IX_Peca_Metal" THEN
22     "Passo_3" := 1;
23     #Inicia_Temp_Peca_Presa := 0;
24     "Passo_2" := 0;
25   END_IF;
26
27   IF "Temp_Peca_Presa".Q AND "IX_Peca_Metal" THEN
28     "Passo_4" := 1;
29     #Inicia_Temp_Peca_Presa := 0;
30     "Passo_2" := 0;
31   END_IF;
32
33   IF "Temp_Peca_Presa".Q AND NOT "IX_Peca_Vermelhal_Preta0" AND NOT "IX_Peca_Metal" THEN
34     "Passo_5" := 1;
35     #Inicia_Temp_Peca_Presa := 0;
36     "Passo_2" := 0;
37   END_IF;
38 END_IF;
```

Figura 75 – Código SCL Passo a Passo (Autor, 2025)

```

40 //Se está no passo 3 e a peça é separada, volta ao passo 1
41 IF "Passo_3" AND "IX_Valv_1_Avancado" AND "IX_Sensor_Peca_Separada" THEN
42     "Passo_1" := 1;
43     "Passo_3" := 0;
44 END_IF;
45
46 //Se está no passo 4 e a peça é separada, volta ao passo 1
47 IF "Passo_4" AND "IX_Valv_2_Avancado" AND "IX_Sensor_Peca_Separada" THEN
48     "Passo_1" := 1;
49     "Passo_4" := 0;
50 END_IF;
51
52 //Se está no passo 5 e a peça é separada, volta ao passo 1
53 IF "Passo_5" AND "IX_Sensor_Peca_Separada" THEN
54     "Passo_1" := 1;
55     "Passo_5" := 0;
56 END_IF;
57
58 //Temporizador peça presa
59 "Temp_Peca_Presa".TON(IN:=#Inicia_Temp_Peca_Presa,
60     PT:=T#3S);

```

Figura 76 – Código SCL Passo a Passo (Autor, 2025)

Conforme figura 77, é observado o código que condiz ao acionamento de cada saída, acordante ao método teórico utilizado.

```
1 IF "Passo_0" THEN
2     // Passo de espera
3     ;
4 END_IF;
5
6 IF "Passo_1" THEN
7     "QX_Prox_Estacao" := 1;
8     "QX_Libera_Peca" := 0;
9     "QX_Cil_Vermelha" := 0;
10    "QX_Cil_Metal" := 0;
11    "QX_Esteira" := 0;
12 ELSE
13     "QX_Prox_Estacao" := 0;
14 END_IF;
15
16 IF "Passo_2" THEN
17     "QX_Esteira" := 1;
18 END_IF;
19
20 IF "Passo_3" THEN
21     "QX_Libera_Peca" := 1;
22     "QX_Cil_Vermelha" := 1;
23 END_IF;
24
25 IF "Passo_4" THEN
26     "QX_Libera_Peca" := 1;
27     "QX_Cil_Metal" := 1;
28 END_IF;
29
30 IF "Passo_5" THEN
31     "QX_Libera_Peca" := 1;
32 END_IF;
```

Figura 77 – Código SCL Acionamentos (Autor, 2025)

4.6 CONCLUSÕES DO CAPÍTULO

Neste capítulo foram definidas todas as entradas e saídas correspondentes às bancadas de distribuição, teste, processamento, manipulação e separação. Após a etapa de definição das IO's, cada sinal foi identificado fisicamente em sua respectiva bancada, permitindo confirmar o correto endereçamento dos dispositivos de campo e assegurar a coerência entre o ambiente físico e o projeto lógico.

Com o mapeamento validado, aplicou-se o método passo a passo do Prof. Agnelo, estruturando a sequência operacional de cada bancada. Para cada estação foi realizada a modelagem dos passos, transições, condições de avanço e ações associadas, garantindo uma representação clara, sistemática e livre de ambiguidades.

Em seguida, essa modelagem foi traduzida para o ambiente de programação do TIA Portal, por meio da implementação em Texto Estruturado (SCL). Nessa etapa, cada passo e transição foi convertido em estruturas de código bem definidas. A implementação em SCL permitiu maior clareza lógica, facilidade de manutenção e fidelidade ao método passo a passo.

Esse processo de definição das IO's, verificação física, modelagem pelo método passo a passo e implementação em SCL no TIA Portal, consolidou uma metodologia completa, organizada e reproduzível, garantindo que a lógica de controle desenvolvida represente fielmente o funcionamento real de cada bancada.

5 CONCLUSÃO

A realização deste Trabalho de Conclusão de Curso permitiu o desenvolvimento e a aplicação de um método formal sistematizado para a programação de Controladores Lógicos Programáveis (CLPs) utilizando a linguagem Texto Estruturado (SCL). O estudo integrou práticas consolidadas de automação industrial, resultando em um procedimento estruturado que facilita a modelagem, análise e implementação de sistemas sequenciais.

Ao longo do trabalho, foram abordados os principais recursos do TIA Portal, incluindo configuração de hardware, criação de blocos funcionais, utilização de temporizadores, contadores e estratégias de organização lógica do programa. Esse embasamento teórico foi essencial para a etapa prática, na qual o método passo a passo foi aplicado de forma sistemática às cinco bancadas que compõem o sistema automatizado: distribuição, teste, processamento, manipulação e separação. Cada uma delas teve seus passos, condições de transição, lógicas de controle e interações analisadas e implementadas, demonstrando a eficácia do método e sua aplicabilidade em diferentes configurações de processos industriais.

A implementação em SCL, aliada aos diagramas de etapas, permitiu construir rotinas claras, modulares e coerentes, reforçando a importância de boas práticas na programação de CLPs. O conjunto de procedimentos apresentados consolida este trabalho como um material de referência, com caráter didático e acessível.

Dessa forma, o TCC cumpre não apenas seu objetivo técnico, mas também assume um papel importante como tutorial estruturado, capaz de auxiliar futuros alunos e profissionais que desejem compreender de maneira clara e orientada como aplicar métodos formais à programação de CLPs. Espera-se que este material contribua para reduzir dificuldades encontradas por iniciantes, ampliar a compreensão dos métodos sequenciais e incentivar práticas mais organizadas e seguras no desenvolvimento de sistemas automatizados.

Por fim, este estudo abre caminho para trabalhos futuros, como a integração com sistemas supervisórios SCADA e aplicação de técnicas de Controle Supervisório (TCS). Assim, o trabalho aqui desenvolvido não apenas consolida conhecimentos, mas também estimula novas investigações e avanços na área de automação industrial.

6 REFERENCIAL TEÓRICO

BARROS, Anderson Rocha. CLP a base de Microcontrolador PIC. In: BTSym-17 – Brazilian Technology Symposium, 2017, Campinas. Anais [...]. Campinas: Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, 2017. Disponível em: <https://www.lcv.fee.unicamp.br/images/BTSym-17/Papers/74603.pdf>.

VIEIRA, Agnelo Denis. Sistemas sequenciais: método passo a passo. Apostila. Curitiba: Pontifícia Universidade Católica do Paraná, out. 2005. Documento digital.

PRUDENTE, Francesco. Automação industrial – PLC: teoria e aplicações. Rio de Janeiro: GEN, 2011.

MENDES, Rafael Manfrin. Programação de CLPs: GRAFCET e GEMMA. São Carlos: Editora Ciência Moderna, 2022.

FESTO DIDACTIC. *Distributing Station: manual da estação de distribuição (MPS® Distributing Station 648811)*. Denkendorf: Festo Didactic GmbH & Co. KG, 2006.

FESTO Corporation; Festo Didactic. *Training Guide – Testing Station. Order # 13099406, Revised 07.19.2013*. Hauppauge, NY: Festo Learning Systems, 2013.

FESTO DIDACTIC. *Processing Station: Assembly Instructions (Gb-Processing)*. Denkendorf: Festo Didactic GmbH & Co., novembro 2002.

FESTO DIDACTIC. *Handling Station (MPS Handling Station): manual*. Denkendorf: Festo Didactic GmbH & Co. KG, 2018.

FESTO DIDACTIC. *Sorting Station: manual da estação de separação / triagem (MPS® Sorting Station 8046325)*. Denkendorf: Festo Didactic SE, 2015.

SIEMENS AG. *SIMATIC S7-1200 – System Manual*. Nuremberg: Siemens, 2024.

SIEMENS AG. *SIMATIC STEP 7 Basic / TIA Portal V12 – Getting Started*.
Nuremberg: Siemens, 01/2013.

POMARI, C. Z. Aula 11 – Arquitetura e edição de blocos. 2018. Disponível em:
<<https://czpomari.wordpress.com/2018/05/26/aula-11-arquitetura-e-edicao-de-blocos/>>. Acesso em: 13 dez. 2025.