UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACULDADE DE ENGENHARIAS, ARQUITETURA E URBANISMO E GEOGRAFIA

GREGORY GELATI

PROJETO DE UM SISTEMA DE AUTOMAÇÃO DE UM PROCESSO INDUSTRIAL COM INTEGRAÇÃO AO USUÁRIO VIA INTERFACE HOMEM MÁQUINA

CAMPO GRANDE, MS

2020

GREGORY GELATI

PROJETO DE UM SISTEMA DE AUTOMAÇÃO DE UM PROCESSO INDUSTRIAL COM INTEGRAÇÃO AO USUÁRIO VIA INTERFACE HOMEM MÁQUINA

Trabalho de conclusão de curso apresentado como exigência para obtenção do grau de Bacharelado em Engenharia Elétrica da Universidade Federal de Mato Grosso do Sul – UFMS

Orientador: Prof. Dr. Valmir Machado Pereira

CAMPO GRANDE, MS

2020

GREGORY GELATI

PROJETO DE UM SISTEMA DE AUTOMAÇÃO DE UM PROCESSO INDUSTRIAL COM INTEGRAÇÃO AO USUÁRIO VIA INTERFACE HOMEM MÁQUINA

Trabalho de conclusão de curso apresentado à Universidade Federal de Mato Grosso do Sul na Faculdade de Engenharias, Arquitetura e Geografia, para obtenção da graduação em Engenharia Elétrica.

Banca examinadora:

Prof. Dr. Valmir Machado Pereira – Orientador

Prof. Dr. Cristiano Quevedo Andrea

Prof. Dr. Paulo Irineu Koltermann

DECLARAÇÃO DE AUTORIA E RESPONSABILIDADE

Eu, Gregory Gelati, residente e domiciliado na cidade de Campo Grande, Estado de Mato Grosso do Sul, portador do RG de nº 2.220.234 SEJUSP/MS e CPF nº 008.278.521-00, declaro que o Trabalho de Conclusão de Curso (TCC) apresentado, com o título "PROJETO DE UM SISTEMA DE AUTOMAÇÃO DE UM PROCESSO INDUSTRIAL COM INTEGRAÇÃO AO USUÁRIO VIA INTERFACE HOMEM MÁQUINA" é de minha autoria e assumo a total responsabilidade pelo seu conteúdo e pela originalidade do texto. Declaro que identifiquei e referenciei todas as fontes e informações gerais que foram utilizadas para construção do presente texto. Declaro também que este texto não foi publicado, em parte, na íntegra ou conteúdo similar em outros meios de comunicação, tendo sido enviado com exclusividade para a conclusão do curso de graduação em Engenharia Elétrica da Universidade Federal de Mato Grosso do Sul (UFMS).

Campo Grande, 27 de Julho de 2020.

Gregory Gelati.

Gregory Gelati

RA 2011.2103.280-0

Curso de Engenharia Elétrica – UFMS

AGRADECIMENTOS

Primeiramente eu agradeço a Deus por me conceder sabedoria, discernimento e paciência para executar este trabalho

Sou grato também aos meus pais, que sempre investiram na minha educação e me apoiaram com palavras e atos de incentivo e coragem.

Agradeço a minha esposa Carliani que esteve comigo desde o início desta jornada, obrigada pelo seu amor, paciência e motivação.

Agradeço também ao meu Prof. Dr. Valmir por te me apresentado e motivado à área de automação industrial em suas aulas de PAI, bem como a confiança em mim depositada ao dar acesso ao laboratório de automação e pelo pronto suporte sempre que necessário.

Agradeço também ao engenheiro eletricista e grande amigo José Henrique Medeiros que dedicou por várias vezes suas horas livres para me apoiar na execução deste trabalho.

Por fim, sou grato a todos que contribuíram de alguma forma na realização deste trabalho.

RESUMO

O presente trabalho surgiu com a necessidade de abordar a automação de processos na indústria, vista sua já difundida utilização, devido a vasta necessidade dos mercados competitivos, seja na otimização de tempo ou de custos. Este trabalho teve como objetivo executar toda programação necessária para controle lógico e integração de uma planta de controle industrial. Para que isso fosse feito de maneira mais incisiva foi realizado um estudo sobre as principais linguagens de programação utilizadas, seguido de um detalhamento dos principais componentes da linguagem mais popular. Estas etapas facilitam a compreensão das etapas de programação e nos resultados obtidos. Para a implementação prática foi utilizado um CLP S7-1200 aliado à uma IHM KTP700, ambos da Siemens. Para modelagem da lógica da planta foi utilizado o *software* TIA Portal V13. Já para simulação do processo industrial foram utilizados dois reservatórios de 20 litros cada, válvula solenoide, bomba d'água e sensores ultrassônicos e de temperatura.

Palavras Chaves: Automação Industrial, CLP, IHM, Linguagem Ladder, TIA Portal V13.

ABSTRACT

This project came up with the need to approach process automation in the industry, given its widespread use, due to the vast need of competitive markets, whether in the optimization of time or costs. This work aimed to execute all necessary programming for logical control and integration of an industrial control system. To that this could be done in a more incisive way, a study was carried out on the main programming languages used, followed by a detailing of the main components of the most popular language. These steps further the understanding of the programming steps and the results obtained. For the practical implementation, a PLC (Programmable Logic Controller) S7-1200 was used together with a KTP700 HMI (Human Machine Interface), both from Siemens. To model the logic of the plant, the software TIA Portal V13 was used. For the simulation of the industrial process, two tanks of 20 liters each, solenoid valve, water pump and ultrasonic and temperature sensors were used.

Keywords: Industrial Automation, PLC, HMI, Ladder Diagram, TIA Portal V13.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

V	Volts
VCC	Voltagem em corrente contínua
Mbits	Megabits
s	Segundo
mm	Milímetros
MB	Megabyte
Bit	Binary Digit
ºC	Grau Celsius
L	Litros
A	Ampere
m	Metros

LISTA DE ABREVIATURAS

PGM	Pulse General Module
CLP	Controlador Lógico Programável
IHM	Interface Homem Máquina
TIA	Totally Integrated Automation
PLC	Programmable Logic Controller
CPU	Central Processing Unit
I/O	Input/Output
LCD	Liquid Crystal Display
IL	Instruction List
ST	Structured Text
SFC	Sequencial Function Chart
FBD	Function Block Diagram
NA	Normalmente aberto
NF	Normalmente fechado
PID	Proporcional integral derivativo
OB	Organization Blocks
FC	Functions
FB	Functions Blocks
DB	Data Blocks
SM	Signal Module
PWM	Pulse-width modulation
LED	Light Emitting Diode

SUMÁRIO

1 INTRODUÇÃO

Criada para facilitar e otimizar diversas áreas da atividade humana a automação pode ser observada nas residências em sistemas integrados de alarme e PGM's (do inglês, *Pulse General Module*), em escritórios e prédios comerciais pode ser utilizada na eficiência energética controlando iluminação natural e artificial através persianas e lâmpadas e na indústria na automação de diversos tipos de processos.

Dentre os usos da automação é possível notar que eles permeiam o conforto, agilidade, praticidade, segurança, dentre outros. Já na automação industrial alguns fins devem ser observados:

- Aumento de produtividade;
- Diminuição de custos;
- Aumento de qualidade;
- Aumento de segurança;
- Otimização de tempo.

Dados estes pontos e considerando a globalização dos mercados e bens de consumo, torna-se evidente o aumento de competitividade nos mercados, o que acarreta diretamente na necessidade de diminuição de custos de produção, gerando uma crescente procura por sistemas automatizados na indústria.

Este contexto de competitividade, aquecimento do mercado e consequente procura, cada vez mais necessária por automação na indústria, é o que motiva este trabalho.

O presente trabalho aborda alguns conceitos sobre automação industrial, seguido por uma simulação de um processo industrial com componentes comumente utilizados na indústria, tais como sensores de temperatura, sensores ultrassônicos, válvulas e bombas. Para a simulação de tal processo foi utilizado um Controlador Lógico Programável (CLP) *Simatic*S7-1200 CPU 1214C DC/DC/DC integrado com uma Interface Homem Máquina (IHM) Siemens KTP700.

Além de utilizar equipamentos já difundidos na indústria como o CLP e a IHM, anteriormente citados, também foi utilizado uma plataforma de prototipagem e estudos amplamente conhecida (NodeMcu). Essa plataforma é responsável pela integração e implementação de um modelo de sensoriamento atípico que visa driblar as limitações de entradas analógicas do *hardware* utilizado.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Desenvolver um processo de automação utilizando sensores, simular processos úteis à implantação na indústria, além de implementar sistema de integração via IHM de forma que o usuário consiga controlar todo o processo de forma intuitiva.

1.1.2 Objetivo Específico

- Desenvolver um sistema de automação industrial utilizando o *software* TIA Portal V13;
- Implementar o sistema de automação à um simulador de uma planta industrial que contém dois tanques;
- Analisar o desempenho da automação e da integração do processo industrial.

1.2 ORGANIZAÇÃO DO TRABALHO

Este trabalho está estruturado da seguinte maneira:

- No Capítulo 1 são apresentados a introdução e os objetivos;
- No Capítulo 2 é feita uma fundamentação teórica apresentando conceitos básicos tais como o que é um CLP e uma IHM bem como seus componentes e sua programação;
- No Capítulo 3 são apresentados os materiais e métodos, abordando então o *hardware* e o *software* utilizados no projeto;
- No Capítulo 4 é apresentada a modelagem do sistema bem como sua programação em linguagem Ladder;
- No Capítulo 5 são feitas as considerações finais e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados alguns conceitos importantes para o projeto bem como as linguagens de programação mais comuns utilizadas na automação industrial, sua lógica de funcionamento, seus componentes básicos e estrutura de programação.

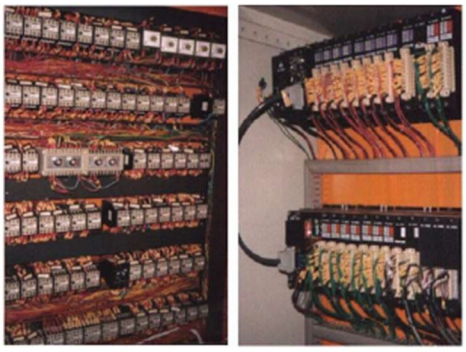
2.1 AUTOMAÇÃO

No final do século XIX e início do século XX, com o advento da 2ª Revolução Industrial, houve um aumento cada vez maior da demanda e competição entre fabricantes, então passou a se fazer necessário uma otimização de tempo e diminuição de custo de produção para que os fabricantes se mantivessem competitivos.

Nesta época Henry Ford criou um conceito que veio a chamar de Linhas de Montagem, que revolucionou a Indústria até os dias de hoje. Ford implementou um sistema de produção em série, onde a linha de montagem estava em movimento e operários especializados em determinada área, aliados com máquinas, faziam o mesmo processo de forma específica e repetitiva. A linha de produção como um todo mostrava um processo sequencial onde para um produto ser finalizado ele passava por diversos operários que realizavam uma tarefa, estritamente, porém com domínio e agilidade.

Para otimizar e comandar as máquinas da linha de montagem, Ford utilizava de relés na automação. No entanto a programação das máquinas era extremamente complexa, as salas de comando ocupavam muito espaço e gastavam muita energia e talvez seu maior empecilho fosse sua estaticidade, não era possível alterar de maneira rápida, simples ou não dispendiosa um comando ou processo.

Figura 1 - Comparativo entre um quadro de relés e um quadro com CLP’s.



Fonte:[1]

Era clara a necessidade de modernização desse tipo de sistema, uma simples mudança em um processo poderia acarretar grandes mudanças de topologia, ligação ou de centenas de componentes no sistema, o que tornava mudanças nos processos fabris inviáveis, além disso os relés eram mecânicos e, portanto, susceptíveis ao desgaste e possuíam um alto gasto de energia.

2.2 CLP

Dada necessidade de encontrar uma alternativa para os sistemas de controle a relés, aliado aos sistemas computadorizados que surgiam na época, foram desenvolvidos os primeiros PLC's (*Programmable Logic Controller*) ou CLP's, que utilizavam a mesma lógica dos sistemas de controle a relés porém em um dispositivo eletrônico que possibilita substituir dispositivos de entrada e saída facilmente viabiliza mudanças de operação e com um preço competitivo comparado aos sistemas a relés.

O CLP teve seu surgimento no final da década de 60, criado pelo engenheiro Richard Morley, com intuito de substituir os sistemas controlados a relés e acabou por revolucionar os sistemas de comando e automação industriais.

Pode ser definido como um dispositivo de estado sólido – um computador industrial, capaz de armazenar instruções para implementação de funções de controle (sequência lógica, temporização e contagem por exemplo), além de realizar operações lógicas aritméticas, manipulação de dados e comunicação em rede, sendo utilizado no controle de Sistemas Automatizados [2].

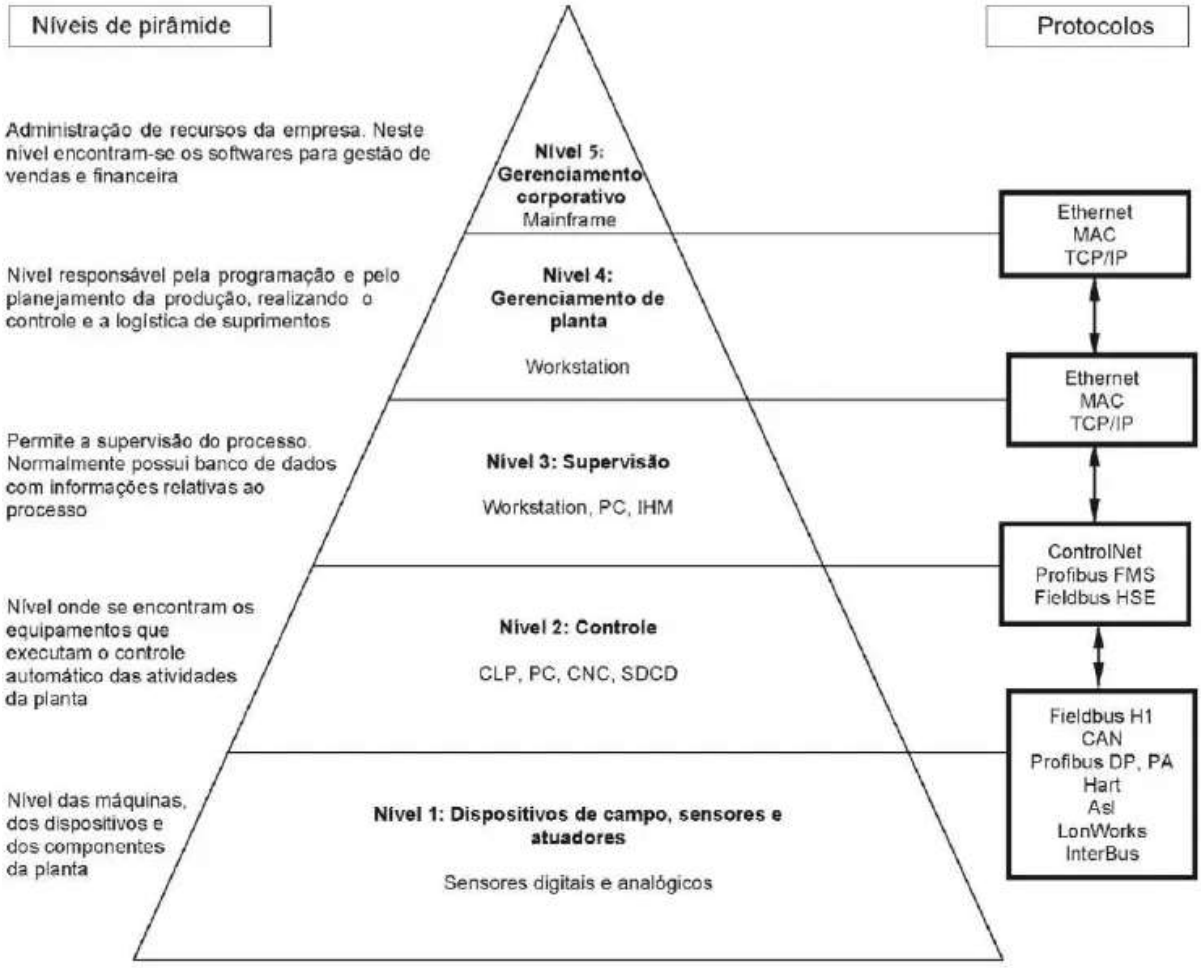
Quando comparados aos seus antecessores é notável a grande evolução que o CLP proporcionou, dentre as principais vantagens pode-se elencar:

- Maior confiabilidade que relés, eliminando o desgaste mecânico e chances de erro em ligações;
- Simples diagnóstico de erros e reprogramação sem necessidade de mudar topologia do sistema;
- Menor consumo de energia, e emissão de ruídos além de ser muito mais compacto;
- Apresenta interface de integração para controle programável e mais intuitiva para os mais distintos grupos de usuários, sendo especializados ou não;
- Comunicação entre outros dispositivos ou CLP's via rede.

Além de trazer mais confiabilidade no sistema de controle e mais segurança ao ambiente fabril, o CLP também proporcionou avanços na supervisão e controle dos processos bem como

no gerenciamento geral de uma indústria. É ilustrada na Figura 2 a pirâmide da automação onde é possível verificar a importância do CLP na automação.

Figura 2 - Pirâmide da Automação.



Fonte: [3]

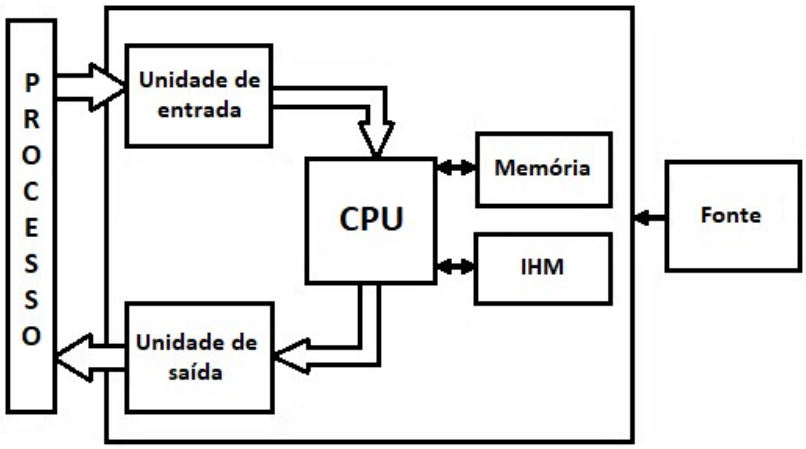
O CLP se encontra no nível 2 da Pirâmide da Automação, ele é o equipamento responsável por realizar a leitura e controle dos dispositivos contidos no nível 1, envia as informações para o nível de supervisão, nível 3, e viabiliza o gerenciamento da planta.

É evidente o avanço tecnológico conquistado pelos CLP's. Com sua utilização cada vez mais difundida na indústria foram criados diversos tipos de CLP's os quais podem ser modulares ou compactos, porém independente disso seguem uma arquitetura básica e um fluxograma de operação padronizado.

2.2.1 Constituição de um CLP

Um CLP é composto de uma CPU (do inglês, *Central Processing Unit*) ou Unidade Central de Processamento, unidades de gerenciamento de entradas e saídas, memória, fonte de alimentação e pode conter uma IHM.

Figura 3 – Arquitetura básica de um CLP.



Fonte: Adaptado de [4]

CPU: Segundo [3], a CPU é "responsável pela execução do programa do usuário, atualização da memória de dados e memória-imagem das entradas e saídas". Ela é o cérebro do CLP, nela são encontrados o processador e as memórias de dados (usadas para armazenar dados temporários para processamento do CLP e são atualizados/apagados a cada varredura do sistema) e as memórias do usuário (onde são encontradas as instruções de operação do programa) [5].

Unidade de Entrada e Saída: Também chamados de módulos I/O (do inglês, Input/Output) são responsáveis pelo gerenciamento das entradas e saídas discretas (contato seco ou binário) ou analógicas.

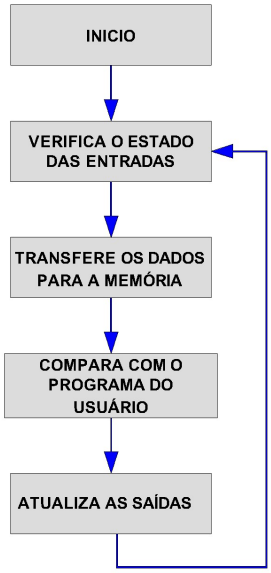
Fonte de Alimentação: Também denominado circuito de alimentação é responsável pela alimentação da CPU e seus componentes bem como das unidades de entrada e saída. Costumam ser de até 24 V por serem menos suscetíveis a ruídos.

IHM: é responsável por fazer a integração do usuário com o programa, possibilitando controle e monitoramento do processo.

2.2.2 Fluxograma de Operação de um CLP

O CLP funciona de maneira sequencial e cíclica e seu fluxograma de operação pode ser observado na Figura 4. É válido ressaltar que enquanto um processo está em operação os outros estão inativos.

Figura 4 - Fluxograma de operação de um CLP.



Fonte: Elaborado pelo autor

Início: Verifica o funcionamento da CPU, memórias, circuitos auxiliares, estados das chaves, existência de um programa de usuário, emite aviso de erro em caso de falha. Desativa todas as saídas.

Verifica o estado das entradas: Lê cada uma das entradas, verificando se houve acionamento. O processo é chamado de ciclo de varredura.

Compara com o programa do usuário: Através das instruções do usuário sobre qual ação tomar em caso de acionamento das entradas o CLP atualiza a memória imagem das saídas.

Atualiza as saídas: As saídas são acionadas ou desativadas conforme a determinação da CPU. Um novo ciclo é iniciado. [6]

2.3 IHM

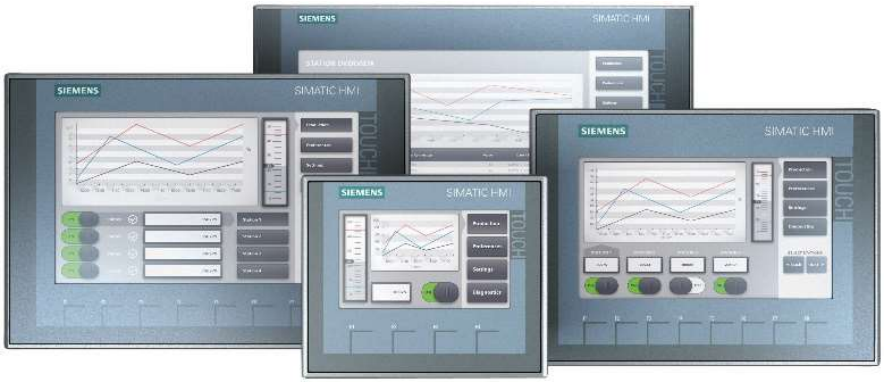
A IHM (Figura 5) é um periférico utilizado junto ao CLP, podendo também ser incorporado ao mesmo, que permite o usuário interagir com a CLP, sendo monitorando ou

fazendo ajustes no processo. Pode ser de diversos tipos, alfa numérica com painel LCD (*Liquid Crystal Display*) e até mesmo uma tela *touchscreen* (sensível ao toque).

O princípio de funcionamento envolve desenvolver previamente, no programa de controle contido no CLP, integrações visuais, alertas ou informações vinculadas com determinadas ações do CLP, possibilitando o monitoramento da planta. Também é possível, caso a IHM seja *touchscreen* ou possua botões auxiliares, desenvolver funções para estes que possibilitem ao usuário alterar parâmetros do programa.

A IHM é responsável por traduzir os sinais vindos do sistema de controle em sinais gráficos para compreensão do usuário. Geralmente estão próximos aos processos controlados, já que seus principais intuitos são de monitorar e configurar parâmetros, o que justifica a necessidade de sua construção robusta.

Figura 5 - Exemplos de IHM.



Fonte: [7]

2.4 LINGUAGENS DE PROGRAMAÇÃO CLP

Como já citado em 2.2.2 o CLP funciona de maneira sequencial e cíclica, analogamente o programa do CLP é um conjunto de expressões booleanas que funciona da mesma maneira. Esta programação pode ser feita em várias linguagens de programação distintas.

Elas consistem em duas linguagens textuais, Lista de Intruções (*InstructionList* – IL) e Texto Estruturado (*Structured Text* – ST), e duas linguagens gráficas, Linguagem Ladder (*Ladder Diagram* – LD) e Diagrama de Blocos de Funções (*Function Block Diagram* – FBD). Os elementos do Diagrama de Funções Sequenciais (*Sequencial Function Chart* – SFC) são definidos para estruturar a organização interna dos programas de CLP e blocos de função. [8]

A linguagem Ladder é a mais comumente utilizada, mais difundida na indústria, isso se deve à sua maneira de programação intimamente ligada à sua origem, os controladores via relés.

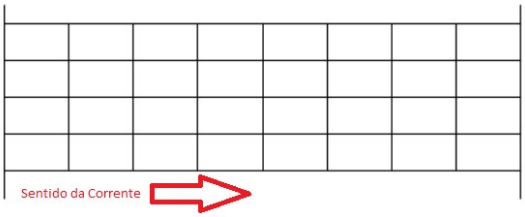
Devido à sua grande aderência à indústria a linguagem Ladder foi a escolhida para programação dessa simulação.

2.4.1 Ladder

A linguagem Ladder foi a primeira que surgiu na programação dos Controladores Lógico Programáveis, pois sua funcionalidade procurava imitar os antigos diagramas elétricos, utilizados pelos técnicos e engenheiros da época. O objetivo era o de evitar uma quebra de paradigmas muito grande, permitindo assim a melhor aceitação do produto no mercado. [2]

Devido a sua funcionalidade semelhante aos diagramas elétricos, seu princípio de funcionamento, de modo geral, considera contatos normalmente abertos (NA) e normalmente fechados (NF) e para melhor entendimento é válido imaginar uma corrente passando entre as linhas no sentido esquerda-direita, sempre nesse sentido, como pode ser visto na Figura 6.

Figura 6 - Simulação de uma malha de um diagrama Ladder.

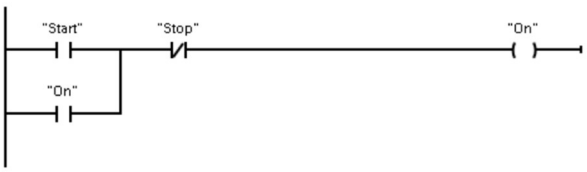


Fonte: Elaborado pelo autor

Por funcionar no sentido esquerda-direita as saídas costumam estar alocadas à direita, pois normalmente são o fim da instrução enquanto as entradas e as condicionais ficam à esquerda.

Na ocorrência de um sistema mais complexo, ou com mais de uma condicional, linhas paralelas podem acionar o mesmo contato, como pode ser observado na Figura 7, em um exemplo de um botão de acionamento com sistema de retenção para que ele permaneça ligado até que o botão desliga seja acionado.

Figura 7 - Exemplo de instrução em Ladder.



Fonte: [9]

A linguagem Ladder, além de proporcionar contatos NA e NF e bobinas, oferece as mais variadas funções como temporizadores, contadores, cálculos matemáticos, dentre outros.

2.4.2 Componentes Básicos

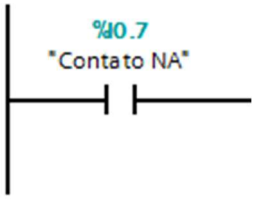
A linguagem Ladder, por ser muito difundida na indústria, evolui bastante, em seu início procurava imitar os controladores à relés, hoje já possui componentes mais complexos, desde contadores, e temporizadores até blocos de controle PID (Proporcional Integral Derivativo). Os componentes mais comuns e mais utilizados na programação da simulação executada são apresentados a seguir:

2.4.2.1 Contatos

Os contatos representam variáveis digitais binárias, tendo nível lógico alto ou baixo (1 ou 0) e podem ser divididos em dois tipos: contatos normalmente abertos (NA) e normalmente fechados (NF).

Os contatos NA (Figura 8), como o próprio nome já diz, possuem em seu estado inativo a representação de um circuito aberto, já quando são acionados fecham o circuito. De forma prática quando a variável associada ao contato possui valor 0 (zero) ou falso o circuito permanece aberto, já quando a variável associada possui valor 1 (um) ou verdadeiro o contato é acionado e muda seu estado fechando o circuito. Na Figura 8 é possível visualizar a representação em Ladder no TIA Portal v13 da Siemens de um contato NA.

Figura 8 - Representação de um contato NA no TIA Portal v13.

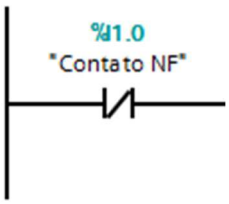


Fonte: Elaborado pelo autor

De maneira análoga os contatos NF (Figura 9), como o próprio nome já diz, possuem em seu estado inativo a representação de um circuito fechado, já quando são acionados abrem

o circuito. De forma prática quando a variável associada ao contato possui valor 0 (zero) ou falso o circuito permanece fechado, já quando a variável associada possui valor 1 (um) ou verdadeiro o contato é acionado e muda seu estado abrindo o circuito. Na Figura 9 é possível visualizar a representação em Ladder no TIA Portal v13 da Siemens de um contato NF.

Figura 9- Representação de um contato NF no TIA Portal v13.



Fonte: Elaborado pelo autor

Além dos contatos simples NA e NF também existem os contatos que escaneiam a mudança de estado da variável associada. Os contatos tipo P (*Scan Positive Signal Edge and Operand*) ou detector de borda de subida escaneiam a mudança de estado de 0 (nível baixo) para 1 (nível alto). Já os contatos do tipo N (*Scan Negative Signal Edge and Operand*) ou detector de borda de descida funcionam de maneira oposta, escaneiam a mudança de estado de 1 (nível alto) para 0 (nível baixo).

2.4.2.2 Bobinas

As bobinas são componentes fundamentais na linguagem Ladder, estão sempre associadas ao fim de uma instrução, são usadas para armazenar um estado de uma memória ou de uma saída.

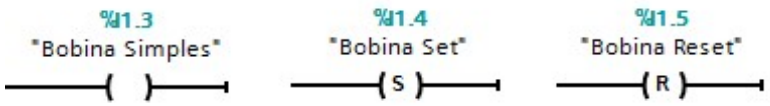
Existem alguns tipos diferentes de bobina para que se encaixam em diferentes propósitos, são eles:

- Bobina simples: caso ativa torna o valor da variável associada verdadeiro (ou nível lógico alto), caso inativa torna o valor da variável associada falso (nível lógico baixo);
- Bobina *set*: semelhante à bobina simples, porém retentiva. Caso ativa torna o valor da variável associada verdadeiro (ou nível lógico alto), porém quando inativa não muda o estado da variável associada.

- Bobina *reset*: oposta à bobina *set*. Caso ativa torna o valor da variável associada falso (ou nível lógico baixo), porém quando inativa não muda o estado da variável associada.

Na Figura 10 é possível visualizar a representação em Ladder no TIA Portal v13 da Siemens das bobinas simples, *set* e *reset*.

Figura 10 - Representação de bobina simples, bobina *set* e bobina *reset* no TIA Portal v13.

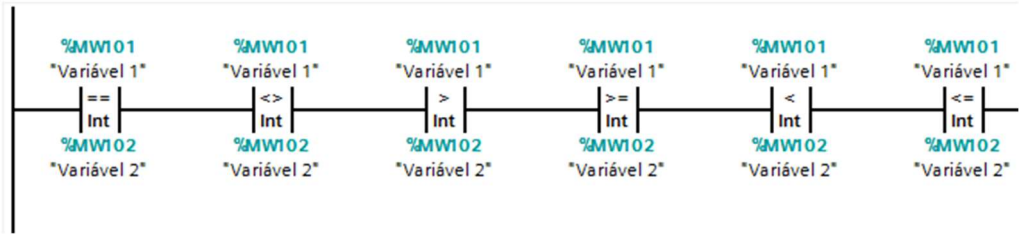


Fonte: Elaborado pelo autor

2.4.2.3 Comparadores

Os comparadores são um grupo de componentes muito importantes em toda área de programação, com eles é possível criar condicionais simples de controle. Têm função de comparar dois valores usando condições de: igual (==), diferente (<>), maior (>), maior/igual (>=), menor (<) ou menor/igual (<=). Caso a condição seja atendida o bloco muda estado para verdadeiro (nível lógico alto). Na Figura 11 é possível visualizar a representação em Ladder no TIA Portal v13 da Siemens dos comparadores.

Figura 11 - Representação dos comparadores: igual (==), diferente (<>), maior (>), maior/igual (>=), menor (<) ou menor/igual (<=), respectivamente, no TIA Portal v13.



Fonte: Elaborado pelo autor

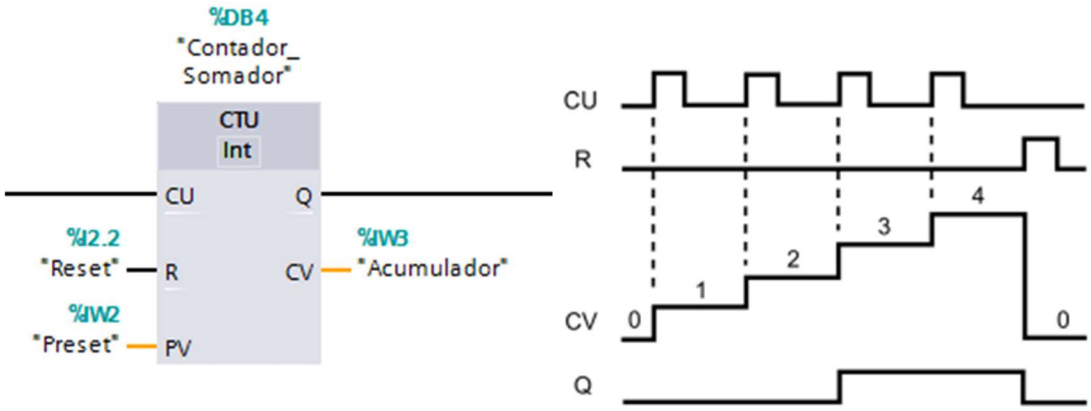
2.4.2.4 Contadores

Os contadores são utilizados para contar eventos internos e externos ao processo, são responsáveis por contar o número de gatilhos associados e armazenar em uma memória

reservada. Além do sinal de gatilho para contagem os contadores possuem *Reset* (para zerar a contagem), uma memória onde a contagem é armazenada (no TIA v13 corresponde ao CV), um *Set Point* ou valor associado para acionamento da saída (no TIA V13 corresponde ao PV) e uma saída (corresponde à Q).

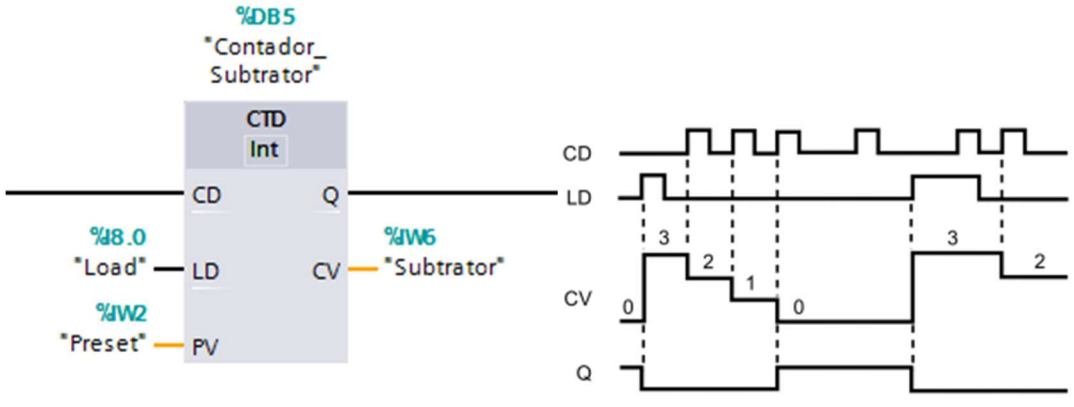
Existem diversos tipos de contadores, os mais comuns são os contadores acumuladores ou somadores (CTU), representados na Figura 12, responsáveis por somar à memória associada a cada gatilho de entrada, contadores subtratores (CTD), representados na Figura 13, responsáveis por subtrair a cada gatilho e por fim contadores somadores/subtratores (CTUD), representados na Figura 14, que é um contador misto. As representações de contadores correspondem aos blocos usados em linguagem Ladder no TIA Portal v13 da Siemens, também foi inserido um exemplo de operação correspondente a cada contador.

Figura 12 - Contador somador e simulação de operação.



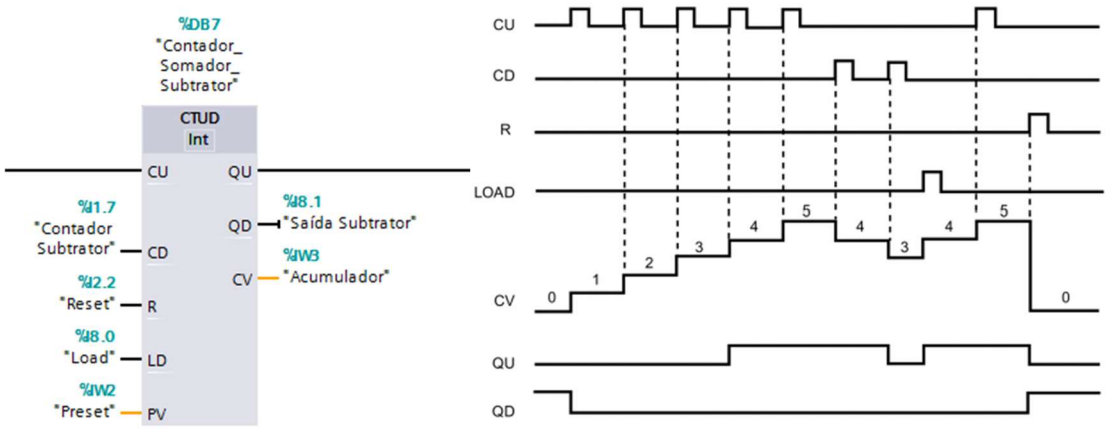
Fonte: Adaptado de [9]

Figura 13 - Contador subtrator e simulação de operação.



Fonte: Adaptado de [9]

Figura 14 - Contador somador/subtrator e simulação de operação.



Fonte: Adaptado de [9]

2.4.2.5 Temporizadores

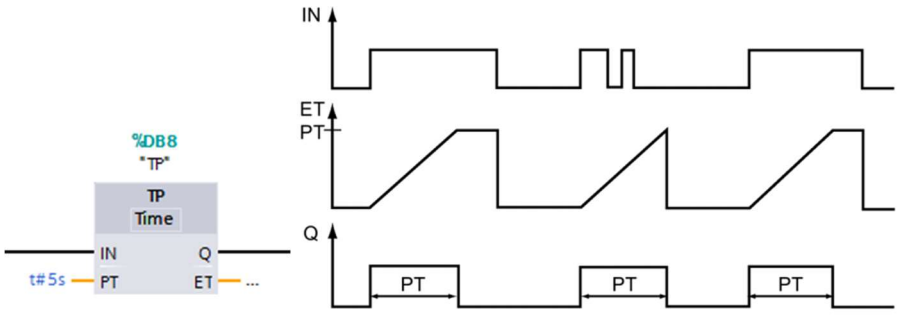
Os temporizadores são blocos muito utilizados na automação, são circuitos que geram um pulso de saída de duração limitada ou com atraso a depender da sequência de pulsos gerada em sua entrada.

Existem alguns tipos de temporizadores, dentre eles têm-se:

- Temporizadores de pulso (TP no Siemens TIA v13), que gera um pulso de saída limitado no tempo definido;
- Temporizadores de retardo ou *delay* na energização (TON no Siemens TIA v13) que gera um set à saída, porém com atraso de acionamento;
- Temporizadores de retardo ou *delay* no desligamento (TOFF no Siemens TIA v13) que gera um *reset* à saída, porém com atraso de acionamento.

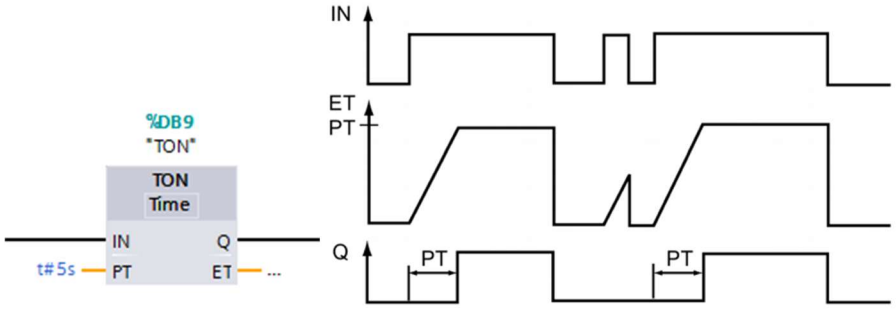
É possível visualizar uma representação em Ladder no TIA Portal v13 da Siemens dos temporizadores TP (Figura 15), TON (Figura 16) e TOFF (Figura 17) bem como exemplos de operação.

Figura 15 - Temporizador de pulso TP e simulação de operação.



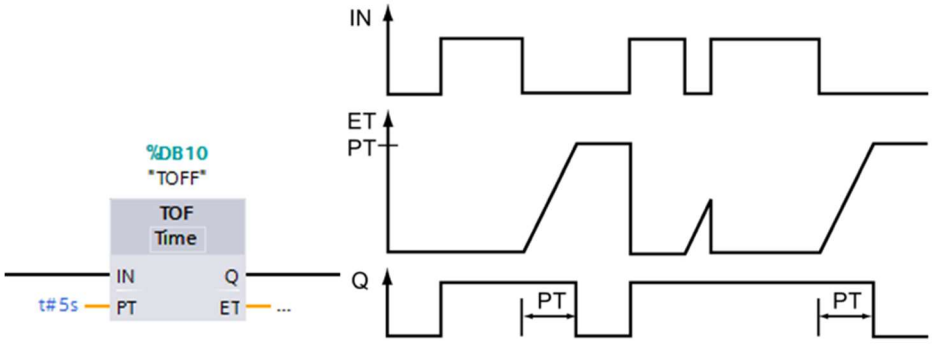
Fonte: Adaptado de [9]

Figura 16 - Temporizador de retardo TON e simulação de operação.



Fonte: Adaptado de [9]

Figura 17 - Temporizador de retardo TOFF e simulação de operação.



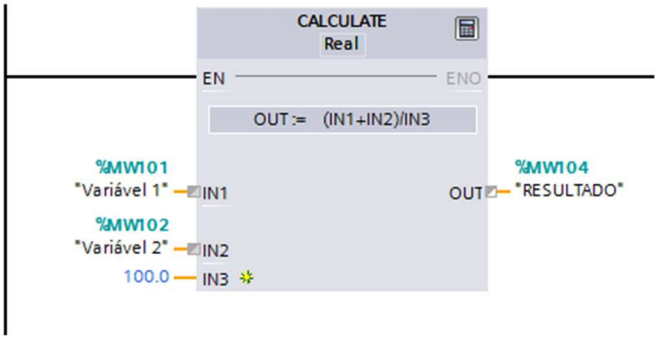
Fonte: Adaptado de [9]

2.4.2.6 Operações Matemáticas

Blocos de funções matemáticas auxiliam o sistema de automação a ser mais independente, dando poder de cálculo e decisão ao sistema. Existem vários tipos de blocos de cálculo, o bloco de adição (ADD), de subtração (SUB), de multiplicação (MUL) e de divisão

(DIV). Para contas mais complexas ou com mais de uma operação existe o bloco calcular (CALCULATE), nele é possível adicionar inúmeras entradas, sendo valores associados a variáveis ou não, e descrever o cálculo desejado. Na Figura 18 é possível visualizar a representação em Ladder no TIA Portal v13 da Siemens do bloco calcular (CALCULATE).

Figura 18 - Representação do bloco calcular no TIA Portal v13.



Fonte: Adaptado de [9]

2.5 ESTRUTURA DE UM PROGRAMA

Dada tamanha versatilidade e possibilidades que um CLP proporciona, em alguns casos sua programação pode vir a ficar extensa, com muitas variáveis, algumas vezes repetitivas, o que dificulta a programação ou compreensão do programa.

Para facilitar a programação e compreensão do programa e até mesmo tornar a sua estrutura mais eficiente, o TIA Portal v13 permite estruturar o programa em diferentes tipos de blocos, com várias funções distintas. É possível, por exemplo, alterar a sequência de execução de uma tarefa, condicionar a execução de um bloco dentro da execução cíclica do CLP ou até mesmo para sistemas de segurança onde uma execução longa é interrompida para acionar uma ação de emergência.

O CLP suporta os seguintes tipos de blocos para estruturar o programa:

- *Organization Blocks* (OBs);
- *Functions* (FCs);
- *Functions Blocks* (FBs).

2.5.1.1 OB

Os *Organization Blocks* (OBs) ou blocos de organização correspondem a um evento específico no CLP e definem a estrutura do programa, formam a interface entre o CPU e o programa de aplicativo. Têm comportamento predefinido e eventos de início, mas também podem ter início personalizado.

OBs executam tarefas específicas, funções como tarefas de inicialização, manipulação de interrupções e erros ou execução de código de programa específico em intervalos de tempo específicos.

2.5.1.2 DB

Data Blocks (DBs) ou bloco de dados armazenam informações que podem ser acessadas e utilizadas pelos blocos do programa. O valor inicial armazenado em um DB pode ser modificado durante a execução do programa e podem ser acessados no mesmo ciclo de varredura ou outros ciclos.

Os blocos de um programa, de modo geral podem ser chamados diversas vezes, porém cada vez que é chamado é utilizado um DB único.

2.5.1.3 FC

Uma *Function* (FC) ou função é um bloco de código sem memória que normalmente executa uma operação específica em um conjunto de valores de entrada. Geralmente são utilizadas em operações com reutilização (como para cálculos matemáticos).

Por não terem memória todos os parâmetros de entrada e saída devem ser conectados, além de não possuírem um bloco DB associado. Para armazenar dados de uma FC deve-se usar uma memória ou uma DB.

2.5.1.4 FB

Um *Function Block* (FB) ou bloco de funções é uma sub-rotina que é executada quando chamada de outro bloco de código (OB, FB ou FC).

O bloco de chamada passa parâmetros para o FB e identifica um bloco de dados específico (DB) que armazena os dados para a chamada ou instância específica desse FB. A

alteração do banco de dados da instância permite que um FB genérico controle a operação de um conjunto de dispositivos. Por exemplo, um FB pode controlar várias bombas ou válvulas, com diferentes DBs de instância que contêm os parâmetros operacionais específicos para cada bomba ou válvula [9].

Um FB possui memória variável e armazena a entrada, saída e memórias estáticas em um DB o que permite utilizar um FB genérico para controlar vários dispositivos, pois um mesmo FB com DBs diferentes não afetam qualquer outro DB.

3 *HARDWARE* E *SOFTWARE* UTILIZADOS

Neste capítulo serão apresentadas as características dos principais elementos de *hardware* e *software* utilizados na automação do processo, destacando-se entre eles: CLP, IHM, NodeMcu, sensores de temperatura e ultrassônicos e os atuadores da planta

3.1 HARDWARE

A seguir serão apresentados os principais equipamentos utilizados neste trabalho.

3.1.1 CLP

O CLP utilizado neste trabalho foi o *Simatic* S7-1200 da Siemens (Figura 19), o qual é compatível com diversos tipos de CPU, sendo o 1214C DC/DC/DC o modelo utilizado.

Figura 19 - CLP Siemens S7-1200.

Fonte: [9]

Este CLP é um modelo compacto, porém modular, aliado à CPU 1214C DC/DC/DC possui as seguintes características:

- Quatorze (14) entradas e dez (10) saídas digitais 24 VCC, na placa (*on board*);
- Duas entradas analógicas 0-10 VCC, na placa (*on board*);
- Expansão de até 8 módulos SM (*Signal Module*), podendo ser analógicos ou digitais;
- Interface de comunicação PROFINET via RJ-45 categoria *Fast Ethernet* (10/100 Mbit/s);
- Integração com até 4 IHM's;
- Expansão para até um (1) módulo de sinal, comunicação e bateria;

- *Slot* para cartão de memória.

É visto a versatilidade do CLP empregado bem como sua interface de comunicação, sua PROFINET, por exemplo, permite soluções tecnológicas integradas em um projeto flexível.

3.1.2 IHM

Para a o controle e integração do sistema foi utilizada uma IHM de última geração, uma IHM Siemens KTP700 (Figura 20), com tela de 7 polegadas de alta resolução, colorida e sensível ao toque. Além disso possui oito (8) botões de função disponíveis. Mais informações importantes sobre a IHM são listadas a seguir:

- Tela de 154,1 x 85,9 mm (7 polegadas), *touchscreen* (tela sensível ao toque) de 800 x 480 *pixels* com mais de 65.500 cores;
- 8 botões de função disponíveis;
- Uma porta USB 2.0;
- Interface de comunicação PROFINET via RJ-45 categoria *Fast Ethernet* (10/100 Mbit/s);
- Memória de dados de 256 MB e memória de programa de 512 MB.

Figura 20 - IHM Siemens KTP700.

Fonte: [10]

O equipamento possui algumas restrições de operação como utilizar apenas em ambientes internos e secos. Abaixo são listadas algumas destas condições:

- Temperatura com tela em formato paisagem: se em vertical 0-50 ºC, se em inclinação de máxima até 35º 0-40 ºC;
- Temperatura com tela em formato retrato: se em vertical 0-40 ºC, se em inclinação de máxima até 35º 0-35 ºC;

- Humidade relativa: 10-90% não podendo haver condensação na parte posterior do dispositivo.

3.1.3 Módulo *Ethernet*

A integração entre o CLP, a IHM e o computador, onde foi elaborado o programa, foi feita através de um módulo *ethernet*, o *Scalance* XB005 (Figura 21), o qual possui capacidade de transmissão *Fast Ethernet* (10/100 Mbit/s).

Figura 21 - Módulo *Ethernet Scalance* XB005.

Fonte: [11]

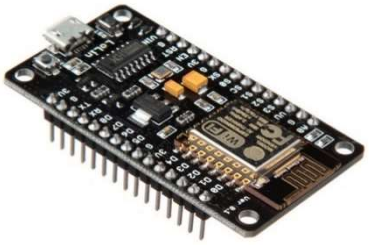
3.1.4 NodeMcu

Visando aprimorar a sensibilidade do sensoriamento de nível uma placa NodeMcu ESP-8266 (Figura 22) foi inserida ao sistema para a utilização de sensores ultrassônicos. O NodeMcu é uma placa de prototipagem e desenvolvimento que utiliza de plataforma de código aberto, dentre as suas principais características pode-se citar:

- 11 pinos de entrada e saída digital com modulação PWM (*Pulse-width modulation*) (exceto pino D0);
- 1 pino de entrada ou saída analógica com resolução de 10 *bits* (*Binary Digit*);
- Alimentação é feita via conector micro USB;

- Pinos operam nível lógico de 3,3 V;
- Compatível com Arduino IDE. Programação é feita por um programa de código aberto.

Figura 22–Placa NodeMcu Esp-8266.



Fonte: [12]

3.1.5 Sensores

3.1.5.1 Sensor de Temperatura

Na aferição de temperatura dos líquidos contidos no tanque foi utilizado o sensor Ds18b20 (Figura 23), do tipo sonda, submersível. Algumas das suas principais características podem ser elencadas a seguir:

- Tensão de alimentação: 3 V a 5,5 V;
- À prova d'água, possui ponta encapsulada por material em aço inoxidável;
- Possui comunicação através de um único fio (1-*wire*);
- Faixa de medição: -55 ºC a 125 ºC;
- Precisão de +-0,5 ºC;

Figura 23 - Sensor de temperatura tipo sonda.

Fonte: [13]

3.1.5.2 Ultrassônico

Para maior precisão no aferimento de nível do sistema foi utilizado um sensor ultrassônico HC-SR04 (Figura 24) integrado com um NodeMcu, o qual faz a lógica de leitura e conversão. Este módulo funciona como uma espécie de sonar, contendo um emissor e um receptor ultrassônico. Ao acionar seu pino *Trigger* ele irá emitir uma onda sonora que, ao encontrar um objeto, irá rebater e ao retornar será lida pelo receptor ultrassônico e acionará seu pino *Echo*.

Figura 24 - Sensor ultrassônico HC-SR04.

Fonte: [14]

Abaixo são listadas suas principais características:

- Tensão de operação: 5 VDC;
- Corrente de operação: 15 mA;
- Ângulo de detecção: ±15º;
- Alcance: 2 cm a 4 m;
- Margem de erro: ±3 mm.

3.1.6 Acionadores

3.1.6.1 Bomba

Para simulação de um atuador de nível foi utilizada uma mini bomba de diafragma e pulverização RS385 (Figura 25). Esta bomba possui tamanho compacto e baixo peso, geralmente é utilizada em projetos de desenvolvimento. Abaixo seguem algumas de suas características:

- Vazão máxima de 120 L/h;

- Tensão de alimentação 12 VCC e corrente de trabalho: 0,5 a 0,7 A (Amperes);
- Elevação máxima: 3 m;
- Altura de aspiração máxima: 2 m.

Figura 25 - Mini bomba RS385.

Fonte: [15]

3.1.6.2 Válvula e Resistor

Para que fosse possível um estudo na variação de nível e temperatura foi utilizado o módulo Simulador de Processos Industriais da De Lorenzo (Figura 26), o qual possibilitou a implantação, além dos dois tanques de 20 litros interligados, de uma válvula solenoide e um resistor de aquecimento.

Figura 26 - Planta De Lorenzo com reservatórios e válvula solenoide.

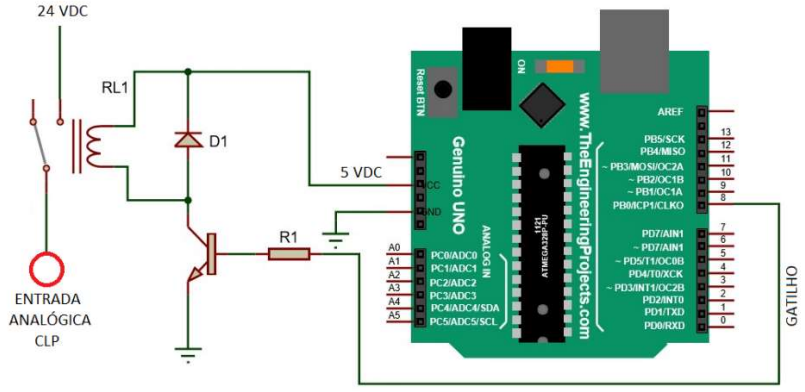
Fonte: Adaptado de [16]

3.1.7 Placa de Relés e Integração

Considerando que o NodeMcu possui saídas digitais 0-5 VCC e o CLP possui entradas digitais 0-24 VCC foi necessário confeccionar uma placa de relés para cada reservatório, com objetivo de proporcionar o correto acionamento das entradas digitais do CLP. Desta forma o NodeMcu é responsável pelo gatilho dos relés e os mesmos, alimentados ligados a um circuito 24 VCC, estão interligados às entradas digitais do CLP, possibilitando assim o correto acionamento.

O circuito de acionamento dos relés pode ser visto na Figura 27.

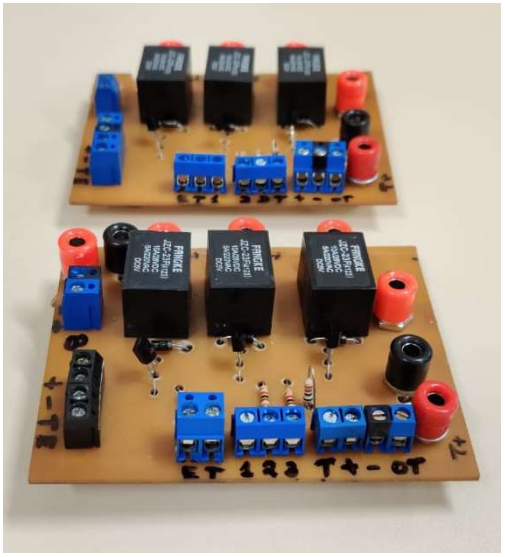
Figura 27 - Circuito de acionamento de relés via NodeMcu.



Fonte: Elaborado pelo autor

A placa também foi utilizada com o intuito de efetuar melhores conexões entre os periféricos, o NodeMcu e o CLP. Foram inseridos terminais para conexão dos sensores de temperatura, de nível, válvula e bomba, bem como para os terminais do NodeMcu. Na Figura 28 pode ser observada a placa confeccionada.

Figura 28 - Placa de integração e acionamento.



Fonte: Elaborado pelo autor

3.2 SOFTWARE

Para desenvolvimento e execução da programação foram utilizados dois principais *softwares*, sendo o TIA Portal V13 responsável pelo controle do CLP e da IHM e o Arduino IDE pela implementação dos sensores.

3.2.1 TIA Portal V13

O TIA Portal V13 é um *software* licenciado pela Siemens utilizado na área de automação, possui interface completa para desenvolvimento, monitoramento, execução ou simulação de processos.

Possui programação versátil, a qual pode ser feita em Ladder, SCL ou FBL. Com TIA Portal é possível implementar toda integração necessária para o desenvolvimento de uma solução de automação, como a programação de lógicas do CLP, telas de integração IHM e *upload* de novas instruções.

3.2.2 Arduino IDE

O Arduino IDE (do inglês, *Integrated Development Environment*) é um aplicativo multiplataforma utilizado para escrever, compilar e fazer upload de programas para placas

Arduino ou placas de desenvolvimento de terceiros compatíveis (a exemplo a placa NodeMcu). Possui programação em C e C++ com algumas funções e estrutura especiais. Por ter grande aderência por parte de desenvolvedores possui extensa biblioteca de aplicações disponível, neste trabalho foi utilizado deste artifício para facilitar e agilizar sua programação de leitura de sensores.

4 SIMULAÇÃO DE PROCESSO INDUSTRIAL

No capítulo será apresentada a planta de controle simulada bem como a lógica do controle almejado e a respectiva programação implementada para tal. Ao final do capítulo uma breve discussão sobre os resultados obtidos.

4.1 APRESENTAÇÃO DO PROCESSO

A planta simula um processo de controle fabril cuja especificação e finalidade não é objetivo deste trabalho. O sistema é constituído por dois reservatórios de 20 L (litros) cada e possui alguns elementos integrados que permitem sua automação, os quais podem ser divididos em elementos de comando e elementos atuadores.

Os elementos de controle do sistema são: um CLP e um NodeMcu os quais são utilizam de dois sensores de temperatura e dois ultrassônicos localizados um em cada reservatório. Já os atuadores do sistema são: uma válvula solenoide, uma resistência para aumento de temperatura e uma bomba de água.

A automação desta simulação prevê três pontos principais:

- Monitorar o sistema de forma rápida e intuitiva;
- Controle por predefinição: o usuário insere os valores desejados de temperatura e nível do tanque inferior (chamado de tanque quente) e o sistema faz a leitura dos níveis atuais e atua de maneira a manter ou alcançar estes valores;
- Controle manual: com objetivo de tornar o sistema mais flexível há necessidade de aliar ao sistema um controle manual dos acionadores, de modo a possibilitar ao operador ligar ou desligar cada um destes de maneira isolada e descorrelacionada, não envolvendo as lógicas de controle da planta.

Um diagrama de blocos das ligações entre o CLP, NodeMcu e sensores é apresentado no Apêndice A deste trabalho.

4.2 PROGRAMAÇÃO

Na implementação do processo apresentado anteriormente foi necessária a execução de um programa para o ESP-8266 realizar as leituras de sensores e um programa com as logicas de controle para o CLP. Concluída a programação foi possível implementar a integração entre

ambos (a qual é apresentada em um diagrama de blocos no Anexo B). Esta programação foi feita via Arduino IDE e TIA Portal V13, respectivamente, a seguir estas serão apresentadas.

4.2.1 NodeMcu

Como já observado anteriormente o CLP utilizado apresenta limitação de ter, de forma integrada, apenas duas entradas analógicas, as quais foram utilizadas para os sensores de temperatura. O NodeMcu foi utilizado para implementar os sensores de nível, de modo a obter maior sensibilidade e integrá-los com as entradas digitais do CLP, além de realizar a leitura dos sensores de temperatura, a qual será enviada para as portas analógicas do CLP. Para estes fins foram utilizadas bibliotecas previamente implementadas pela comunidade para a utilização desses tipos de sensores.

A programação do NodeMcu não é o objetivo deste trabalho, seu *sketch* é apresentado no Apêndice B deste trabalho, a seguir uma breve explicação de sua lógica é apresentada.

Para realizar a leitura dos sensores de temperatura foi utilizada a biblioteca *Dallas Temperature*, responsável por ler o sensor e entregar o valor lido em ºC. Essa informação será posteriormente mapeada de forma a converter a leitura do sensor de 0-100ºC para 0-5VDC, de modo a tornar compatível com as saídas digitais do NodeMcu. Este sinal é enviado às portas analógicas do CLP, o qual é responsável por realizar novo mapeamento para conversão de Volts para graus *Celsius* para então realizar a lógica de controle.

Para aferir o nível dos tanques foi utilizado um sensor ultrassônico em cada tanque, este sensor realiza a aferição de distância até o líquido ou fundo do tanque. Esta aferição é feita pelo programa do NodeMcu, considerando o tempo passado entre o disparo do *Trigger* do sensor ultrassônico e a resposta obtida pelo *Echo* bem como a velocidade do som, como pode ser visto na equação a seguir.

$$Distancia = \frac{(Tempo\ Echo - Tempo\ Trigger) * Velocidade\ do\ som}{2}$$

Como as duas entradas analógicas do CLP já estavam em uso para os sensores de temperatura a saída adotada para uma leitura de nível mais sensível foi utilizar de seis portas digitais três para cada tanque, efetuando uma mudança bit a bit de modo a graduar seus níveis a cada 2 litros. Para tal implementação foi utilizada uma verdade a qual pode ser vista na Tabela 1.

Tabela 1 - Entradas digitais vinculadas aos Bit's de leitura de nível.

BIT_A	BIT_B	BIT_C	NÍVEL (L)
0	0	0	0
0	0	1	2
0	1	0	4
0	1	1	6
1	0	0	8
1	0	1	10
1	1	0	12
1	1	1	14

Fonte: Elaborado pelo autor

4.2.2 CLP

A programação do CLP consistiu em implementar os controles lógicos e a integração destes com a IHM para que o usuário possa interagir com a planta. A programação foi dividida em grupos de OB's de modo a organizar cada etapa de controle do programa, cada uma das etapas de programação será abordada a seguir.

4.2.2.1 Telas IHM

Para melhor compreensão da programação implementada no CLP serão apresentadas as telas de integração IHM responsáveis pelos comandos. Foram criadas cinco telas de navegação para o usuário, sendo:

- Tela inicial (HOME);
- Tela de configuração de temperatura (TEMP);
- Tela de configuração de nível (NÍVEL);
- Tela de comando manual (MANUAL);
- Tela de emergência.

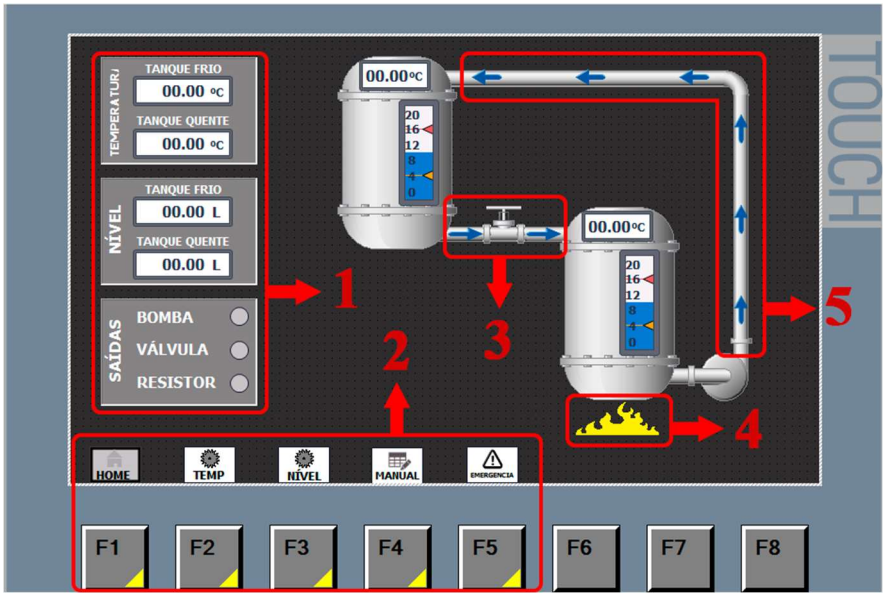
As transições entre as telas de usuário são feitas através dos botões de função F1 à F5 da IHM.

Uma representação da tela inicial (HOME) pode ser observada Figura 29, ela é utilizada apenas para fins de visualização de estado em tempo real e monitoramento da planta, não foram inseridos comandos.

Com intuito de facilitar e agilizar uma inspeção visual do processo, foi inserido nessa tela uma representação da planta através de dois tanques, os quais também possuem indicativo de nível e temperatura. Os demais grupos componentes observados nesta tela estes foram numerados e indicados na Figura 29 e cada um deles é descrito a seguir:

1. Conjunto de leitura de entradas e saídas. Neste conjunto é possível monitorar a temperatura e o nível de cada um dos tanques, bem como quais acionadores estão ativos (bomba, válvula e resistor) através de uma representação de um LED (*Light Emitting Diode*) indicativo para cada um destes;
2. Botões de seleção de tela da IHM;
3. Setas representando fluxo do tanque superior (tanque frio) ao tanque inferior (tanque quente), normalmente invisíveis, são apenas mostradas ao usuário caso o acionador de saída da válvula seja ativo. Têm objetivo de facilitar a inspeção visual da planta;
4. Representação de aquecimento aplicado ao tanque inferior (tanque quente) normalmente invisíveis é mostrado ao usuário caso o acionador de saída do resistor seja ativo, tem objetivo de facilitar a inspeção visual da planta;
5. Setas representando fluxo do tanque inferior (tanque quente) ao tanque superior (tanque frio), normalmente invisíveis, são apenas mostradas ao usuário caso o acionador de saída da bomba seja ativo. Têm objetivo de facilitar a inspeção visual da planta.

Figura 29 - Tela de monitoramento (HOME).

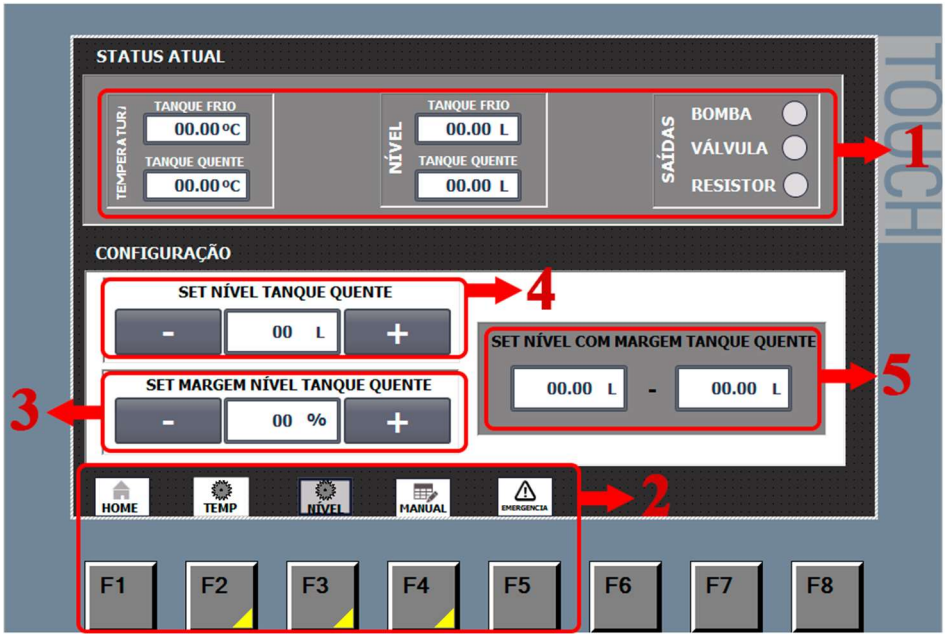


Fonte: Elaborado pelo autor

Uma extensão simplificada da tela de monitoramento foi implementada em cada tela de controle para que o operador, quando for alterar os valores de referência, possa visualizar os níveis atuais e como estas alterações estão influenciando o comportamento da planta.

Pode ser observado na Figura 30 a tela de configuração de nível (NÍVEL).

Figura 30 - Tela de configuração de nível (NÍVEL).



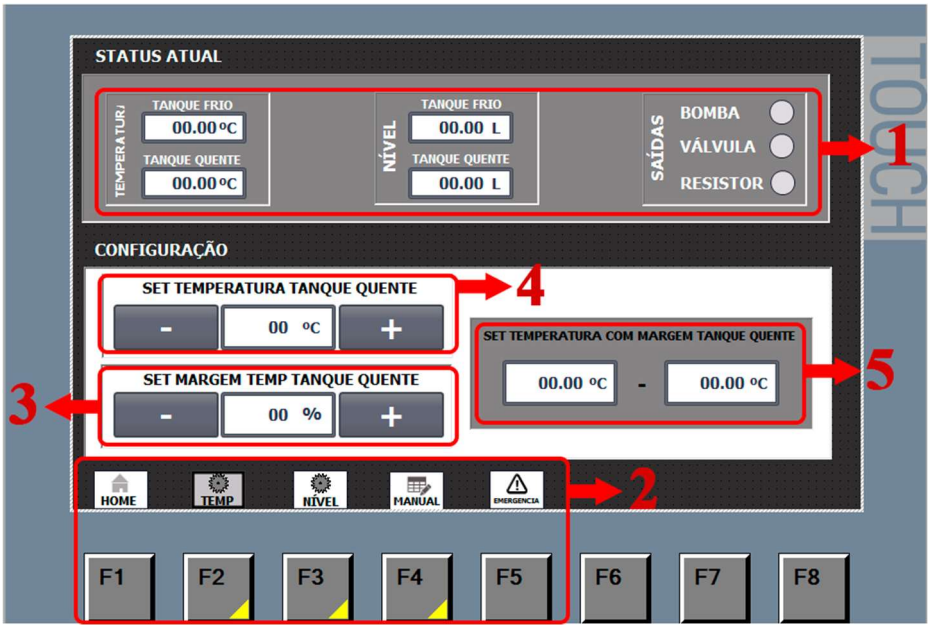
Fonte: Elaborado pelo autor

Esta tela tem como objetivo principal controlar o nível desejado no tanque inferior (tanque quente) de modo que a planta atue de modo a atingir este valor desejado. Os grupos componentes observados nesta tela foram numerados e indicados na Figura 30 e cada um deles é descrito a seguir:

1. Conjunto de leitura de entradas e saídas. Neste conjunto é possível monitorar a temperatura e o nível de cada um dos tanques, bem como quais acionadores estão ativos (bomba, válvula e resistor) através de uma representação de um LED indicativo para cada um destes;
2. Botões de seleção de tela da IHM;
3. Controle de margem de tolerância de nível aplicada no controle, pode ser acrescida ou decrescida manualmente com os botões auxiliares laterais ou inserindo o valor desejado via teclado ao clicar no bloco central, o qual representa a margem desejada;
4. Controle de nível desejado no tanque inferior (tanque quente), pode ser acrescido ou decrescido manualmente com os botões auxiliares laterais ou inserindo o valor desejado via teclado ao clicar no bloco central, o qual representa o nível desejado;
5. Representa os valores máximo e mínimo de nível considerando o valor set desejado e a margem de nível inserida, tem como objetivo mostrar ao usuário quais valores serão utilizados pelo controle da planta.

O mesmo pode ser observado na tela de configuração de temperatura (TEMP) vista na Figura 31, com diferencial de controlar os valores desejados de temperatura ao invés de nível, ademais não há mudanças, ambas as telas possuem interface semelhante para tornar a interface mais amigável ao usuário. Para diferenciar a tela atual das demais existe, próximo aos botões de função da IHM, um indicador de tela que muda de cor conforme a tela atual.

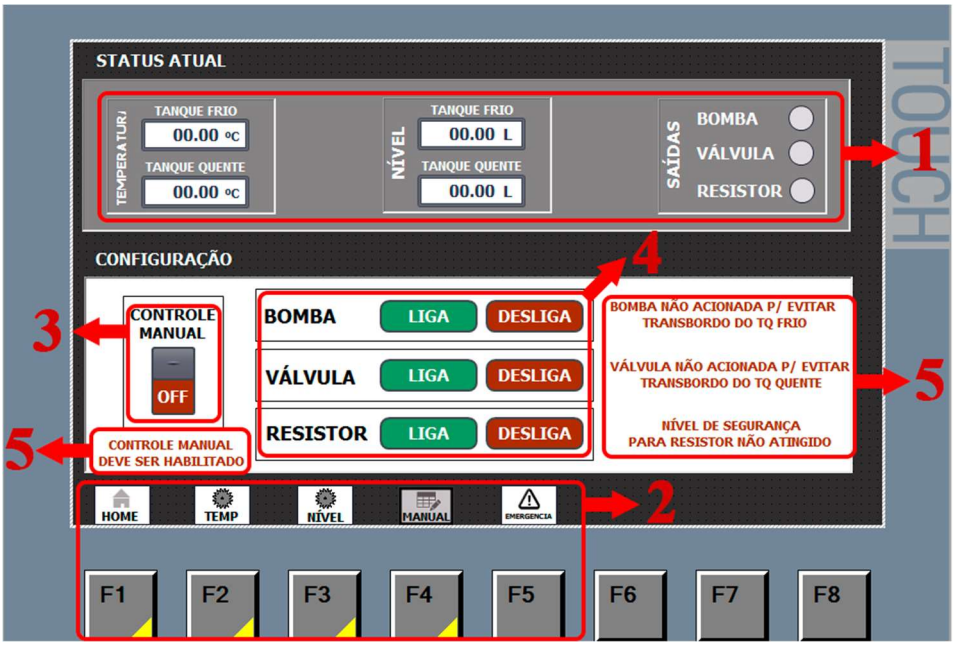
Figura 31 - Tela de configuração de temperatura (TEMP).



Fonte: Elaborado pelo autor

Com objetivo de tornar a planta mais flexível foi implementado um comando manual dos acionadores (Figura 32), de modo a desabilitar os controles automáticos da planta, com exceção dos controles de segurança, e dar ao usuário livre controle dos acionadores.

Figura 32 - Tela de comando manual (MANUAL)



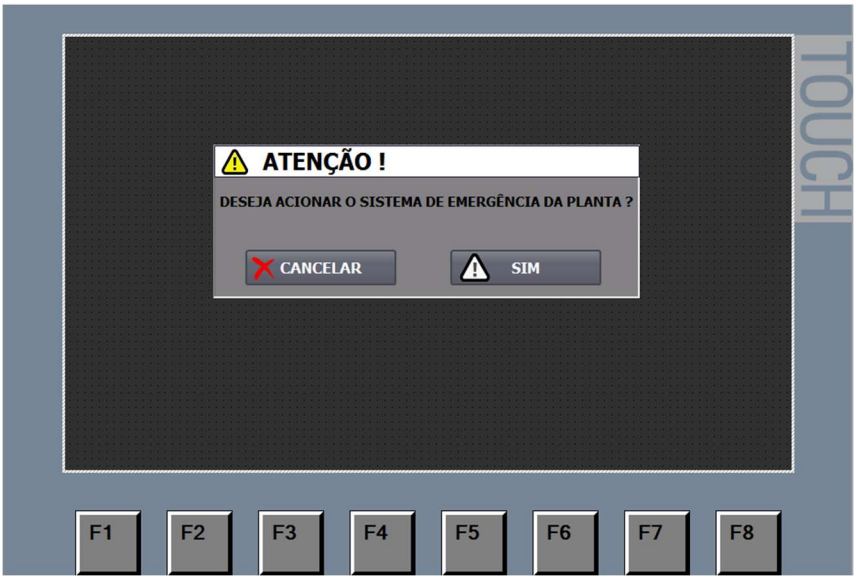
Fonte: Elaborado pelo autor

Os grupos componentes observados nesta tela de comando manual (MANUAL) foram numerados e indicados na Figura 31e Figura 32, e cada um deles é descrito a seguir:

1. Conjunto de leitura de entradas e saídas. Neste conjunto é possível monitorar a temperatura e o nível de cada um dos tanques, bem como quais acionadores estão ativos (bomba, válvula e resistor) através de uma representação de um LED indicativo para cada um destes;
2. Botões de seleção de tela da IHM;
3. Seletor para ativar ou desativar o comando manual;
4. Comandos de ligar ou desligar os acionadores da planta (bomba, válvula e resistor);
5. Mensagens de alerta normalmente invisíveis aparecem ao usuário apenas se a sua memória vinculada seja acionada, utilizadas para alertar o estado do controle manual (ligado ou desligado) e para alertar sobre os níveis de segurança dos acionadores.

Além das telas de controle também foi utilizada uma tela de emergência (Figura 33), e uma tela auxiliar apresentada na Figura 34. A primeira tela é acionada quando o usuário pressiona o botão de função emergência e um alerta é exibido, nele há opção de acionar o sistema de emergência ou cancelar o comando.

Figura 33 - Tela de emergência.

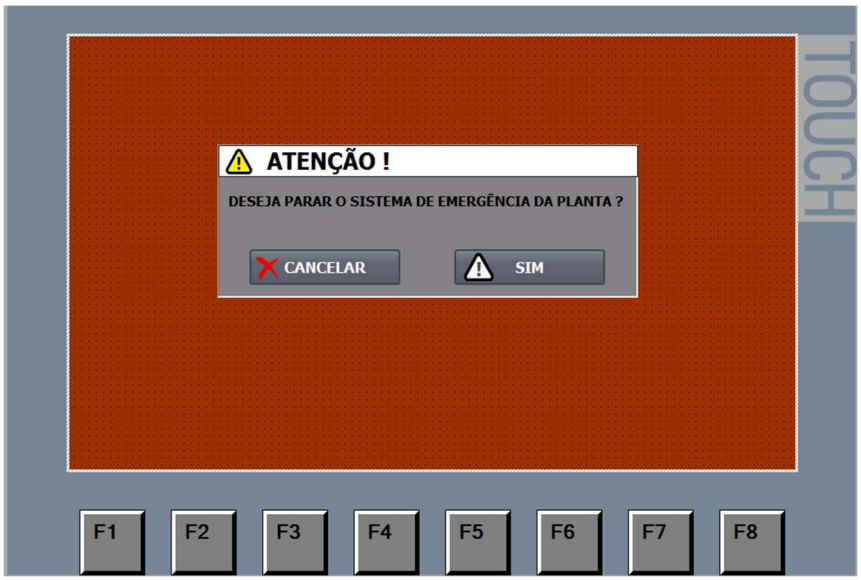


Fonte: Elaborado pelo autor

Caso o usuário opte por cancelar o acionamento, a planta retorna à tela inicial e retorna aos comandos anteriores, caso contrário uma sub-tela (Figura 34) é acionada com intuito de

deixar claro ao usuário que o estado de emergência está ativo. Este estado desabilita qualquer um dos controles que estiver em andamento e habilita o modo de emergência. Para desabilitar o sistema de emergência basta pressionar “SIM” no alerta da sub-tela de emergência.

Figura 34 - Sub-tela de emergência.



Fonte: Elaborado pelo autor

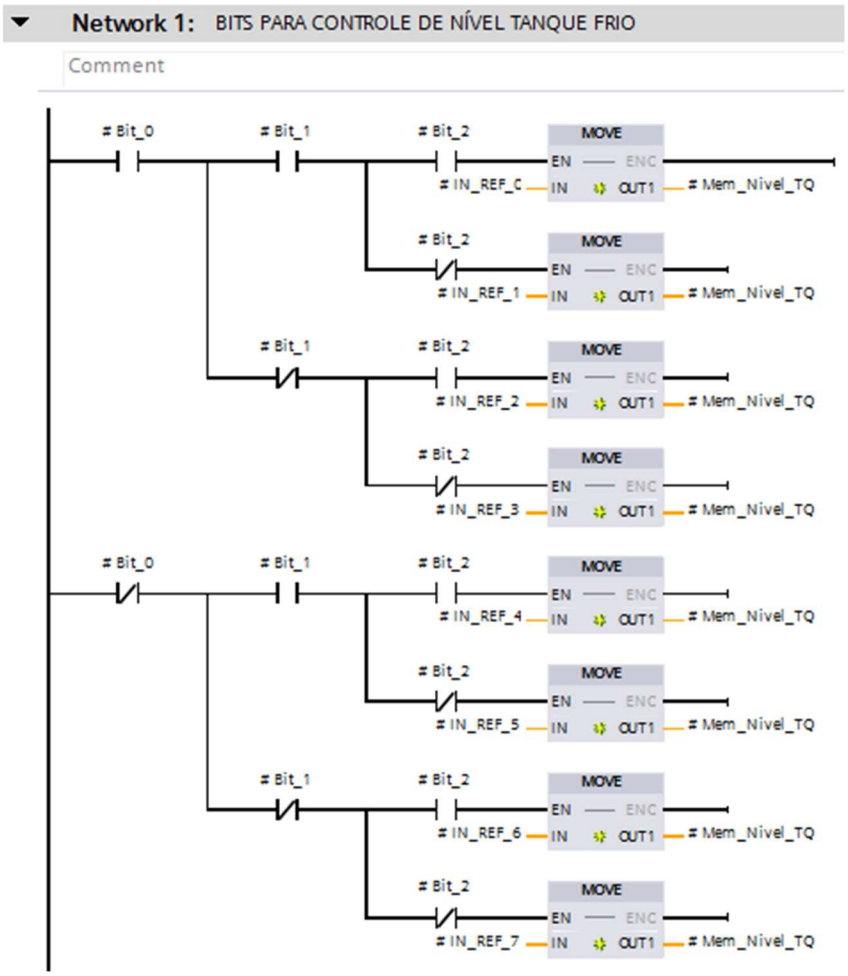
4.2.2.2 FB’s de Ferramentas

Com objetivo de organizar o programa e tornar a programação mais ágil foram implementados alguns FB’s para lógicas extensas ou utilizadas mais de uma vez.

FB_SET_NIVEL – Este FB é responsável por realizar a leitura das entradas digitais associadas aos níveis dos tanques superior e inferior e mover seu respectivo valor à variável Real associada. A Tabela 1 relaciona as entradas digitais à leitura de nível.

A programação deste FB pode ser vista na Figura 35 seguida de uma breve explicação de sua lógica de funcionamento.

Figura 35 - FB_SET_NIVEL.



Fonte: Elaborado pelo autor

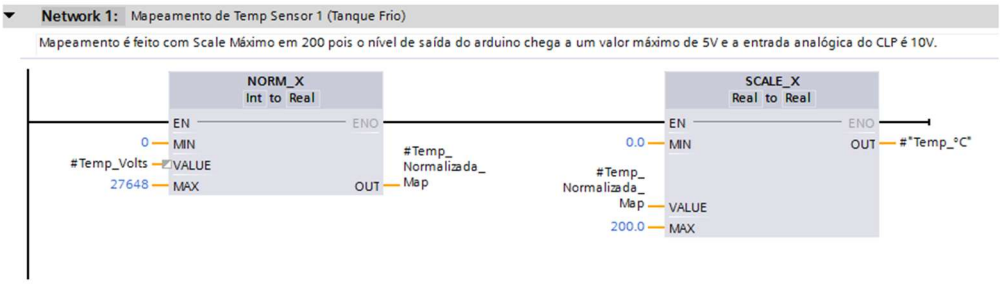
Com base nos valores de referência adotados na Tabela 1 foram implementadas três bobinas NA (correspondem aos três bits vindos do NodeMcu, responsáveis pelo controle de nível) e valores de nível correspondente à cada uma das combinações.

A lógica funciona de maneira a satisfazer apenas uma das oito possíveis condições, então, através da função MOVE, o valor de referência inserido pelo usuário em "#IN_REF_X" é levado à memória vinculada ao respectivo sensor de nível.

Em "Bit_0" à "Bit_2" são inseridas as três variáveis de entrada relacionadas ao controle de nível do respectivo tanque, a seguir "IN_REF_0" à "IN_REF_7" são inseridos os valores respectivos à cada uma das graduações de nível. O campo "Mem_Nivel_TQ" corresponde à memória associada ao nível atual do tanque.

FB_MAP_TEMP – Este FB é responsável por converter os valores oriundos das entradas analógicas de tensão em ºC.

Figura 36 - FB_MAP_TEMP.



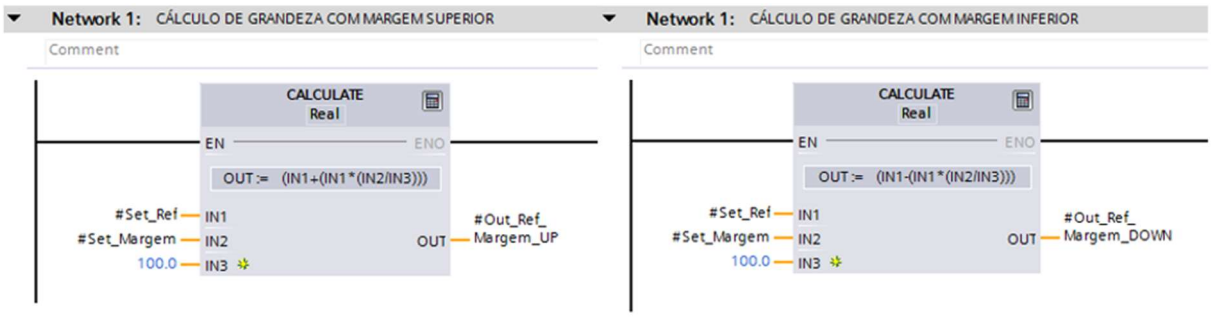
Fonte: Elaborado pelo autor

Conforme pôde ser visto na Figura 36, primeiramente o valor é convertido de Inteiro para Real, limitando os valores de 0 à 27648 (valores mínimo e máximo da entrada analógica do CLP), posteriormente foi aplicado um SCALE proporcionalizar os valores de entrada. Como a entrada analógica do CLP varia de 0 V à 10 V e o NodeMcu apenas de 0 V à 5 V o valor máximo do SCALE foi atribuído como 200.

Em "Temp_Volts" é inserida a variável analógica de entrada correspondente ao sensor de temperatura, já em "Temp_ºC" a memória associada à temperatura atual do respectivo tanque.

FB_SET_MARGEM_UP e **FB_SET_MARGEM_DOWN** (Figura37) – Estes FB's são responsáveis por ler os valores de referência de set de grandeza (nível e temperatura) e margem de tolerância inseridos pelo usuário e calcular a margem superior (em FB_SET_MARGEM_UP) e inferior (em FB_SET_MARGEM_DOWN).

Figura37 - FB_SET_MARGEM_UP e FB_SET_MARGEM_DOWN.



Fonte: Elaborado pelo autor

Em "Set_Ref" é inserida a grandeza (temperatura ou litros) desejada pelo usuário, já em "Set_Margem" é inserida a margem de tolerância permitida ao sistema, válido ressaltar que caso a margem seja muito baixa a planta pode permanecer em constante atuação, caso margem

seja muito alta pode perder a sensibilidade do controle. Em “Out_Ref_Margem_UP” e “Out_Ref_Margem_DOWN” são inseridas as memórias associadas aos níveis com margem superior e inferior, respectivamente.

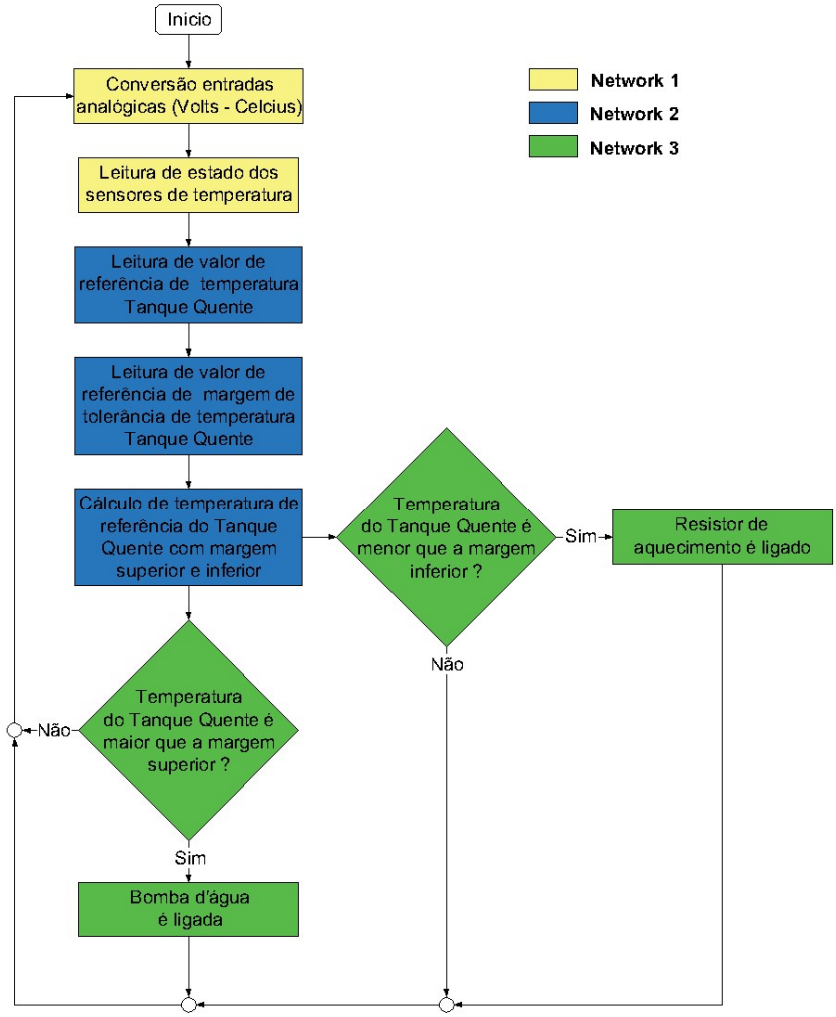
4.2.2.3 OB de Controle de Temperatura

O controle de temperatura teve sua lógica dividida em três *Network’s.*

- Leitura de sensores de temperatura;
- Cálculo de set de temperatura;
- Logica de controle de acionadores.

O fluxograma apresentado na Figura 38 mostra a lógica de controle utilizada no OB de temperatura.

Figura 38 - Fluxograma de controle de temperatura.



Fonte: Elaborado pelo autor

A Tabela 2 mostra o nome, tipo, endereço lógico bem como breve descrição da utilização das variáveis criadas para este OB.

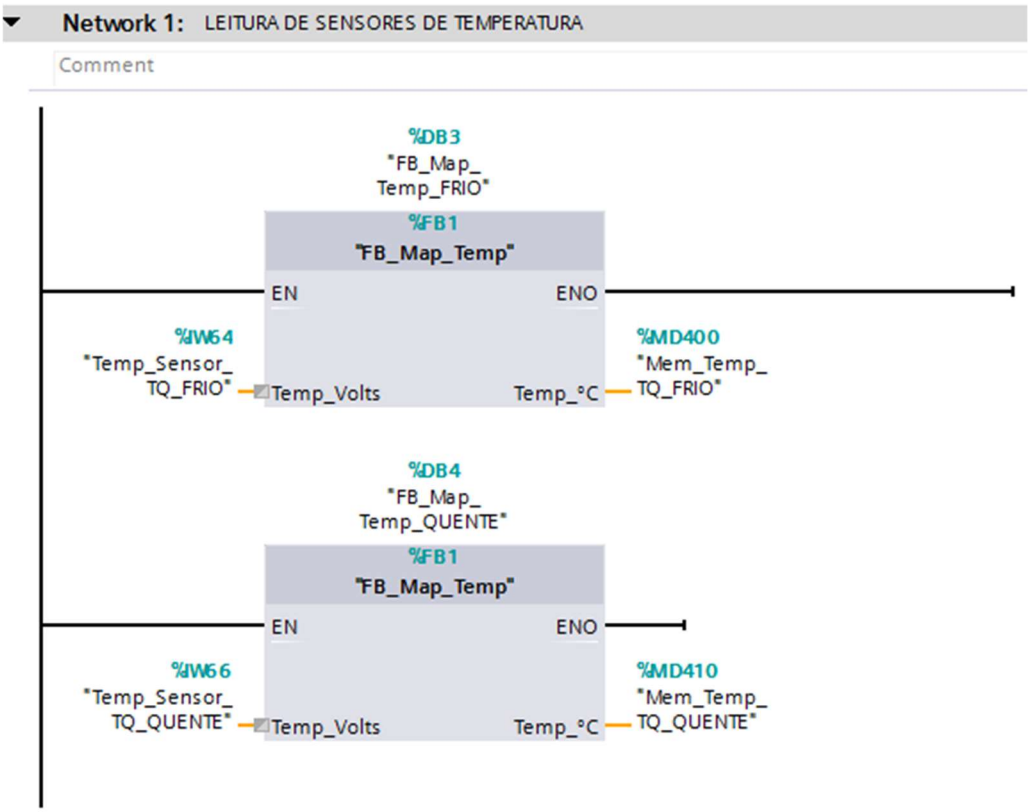
Tabela 2– Variáveis utilizadas no OB de controle de temperatura.

VARIÁVEL	TIPO	ENDEREÇO	DESCRIÇÃO
Mem_Temp_TQ_FRIO	Real	%MD400	Temperatura Sensor TQ Frio Convertida para ºC
Mem_Temp_TQ_QUENTE	Real	%MD410	Temperatura Sensor TQ Quente Convertida para ºC
Out_Resistor_Temp	Bool	%M410.0	Memória de Acionamento de Resistor Vindo da Lógica de Temperatura
Out_Valvula_Temp	Bool	%M400.0	Memória de Acionamento de Válvula Vindo da Lógica de Temperatura
Set_Margem_Temperatura	Real	%MD430	Referência de Margem de Tolerância para Temperatura
Set_Ref_Temp	Int	%MW420	Referência de Temperatura Desejada
Temp_Sensor_TQ_FRIO	UInt	%IW64	Entrada do Sensor de Temperatura do Tanque Frio Analógica 0 do Painel
Temp_Sensor_TQ_QUENTE	UInt	%IW66	Entrada do Sensor de Temperatura do Tanque Quente Analógica 1 do Painel
Temperatura_Set_Com_Margem_DOWN	Real	%MD450	Temperatura com Margem Inferior
Temperatura_Set_Com_Margem_UP	Real	%MD440	Temperatura com Margem Superior

Fonte: Elaborado pelo autor

No primeiro *Network* (Figura 39), responsável por realizar a leitura dos sensores de temperatura, foi utilizado o FB_MAP_TEMP duas vezes, de maneira a realizar a leitura dos sensores dos tanques de modo independente.

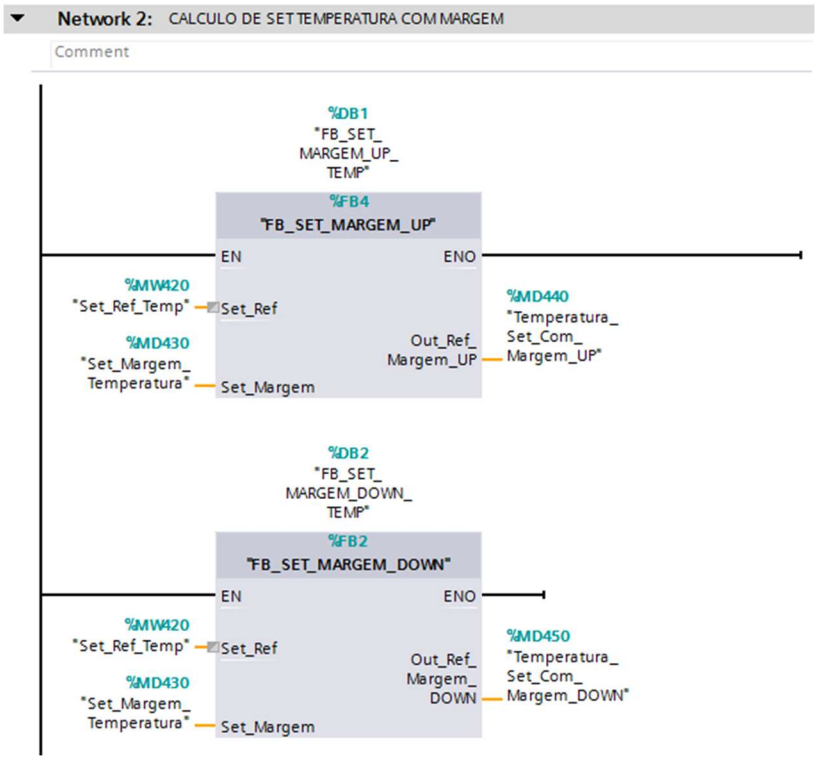
Figura 39 - *Network* 1 do OB de controle de temperatura.



Fonte: Elaborado pelo autor

No segundo *Network* (Figura 40), é realizada a leitura dos valores de temperatura inseridos pelo usuário bem como a margem de tolerância, junto a isto são calculados os valores de temperatura com margem superior e inferior. Esta implementação é feita utilizando FB_SET_MARGEM_UP e FB_SET_MARGEM_DOWN.

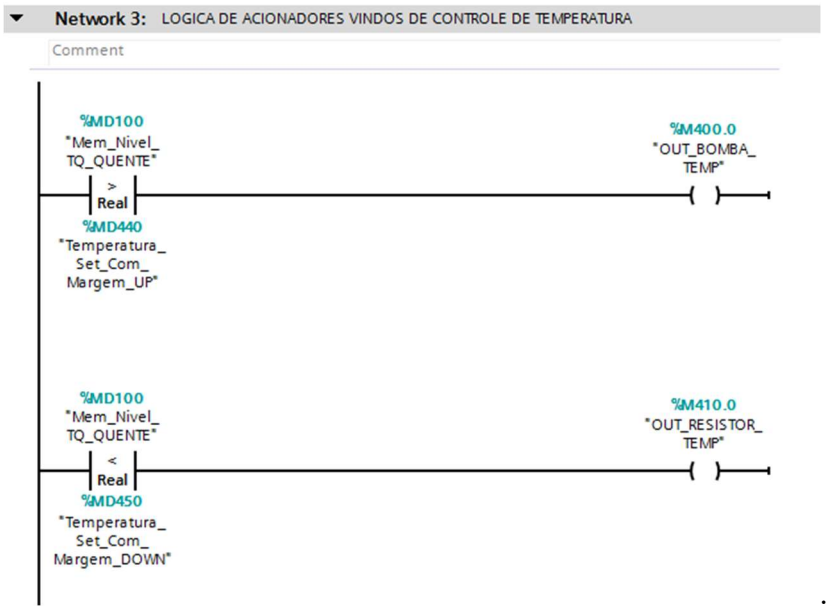
Figura 40 - *Network* 2 do OB de controle de temperatura.



Fonte: Elaborado pelo autor

O terceiro e último *Network* de temperatura (Figura 41) realiza a lógica de controle dos acionadores.

Figura 41 – *Network* 3 do OB de controle de temperatura.



Fonte: Elaborado pelo autor

Para o controle automático de temperatura são consideradas duas atuações.

- Caso o valor lido pelo sensor de tanque quente seja superior ao valor inserido pelo usuário adicionado à margem de tolerância (temperatura do líquido do tanque quente está mais alta do que a desejada) a bomba é acionada, de modo a liberar líquido contido no tanque superior (tanque frio) ocasionando um arrefecimento do líquido contido no tanque inferior (tanque quente);
- Caso o valor lido pelo sensor de tanque quente seja inferior ao valor inserido pelo usuário subtraído a margem de tolerância (temperatura do líquido do tanque quente está mais baixa do que a desejada) o resistor é acionado, de modo a aquecer líquido contido no tanque inferior (tanque quente).

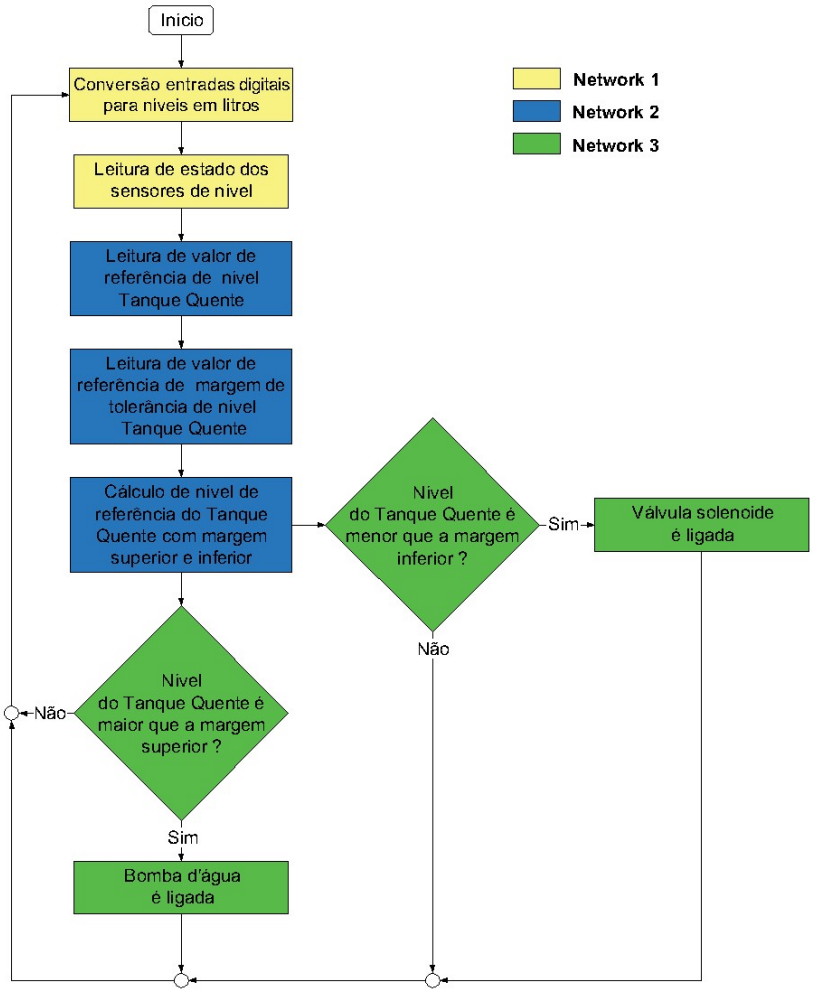
4.2.2.4 OB de Controle de Nível

O controle de nível teve sua lógica dividida em três *Network's.*

- Leitura de sensores de nível;
- Cálculo de set de nível;
- Logica de controle de acionadores.

O fluxograma apresentado na Figura 42 mostra a lógica de controle utilizada no OB de temperatura.

Figura 42 - Fluxograma de controle de nível.



Fonte: Elaborado pelo autor

A Tabela 3 mostra o nome, tipo, endereço lógico bem como breve descrição da utilização das variáveis criadas para este OB.

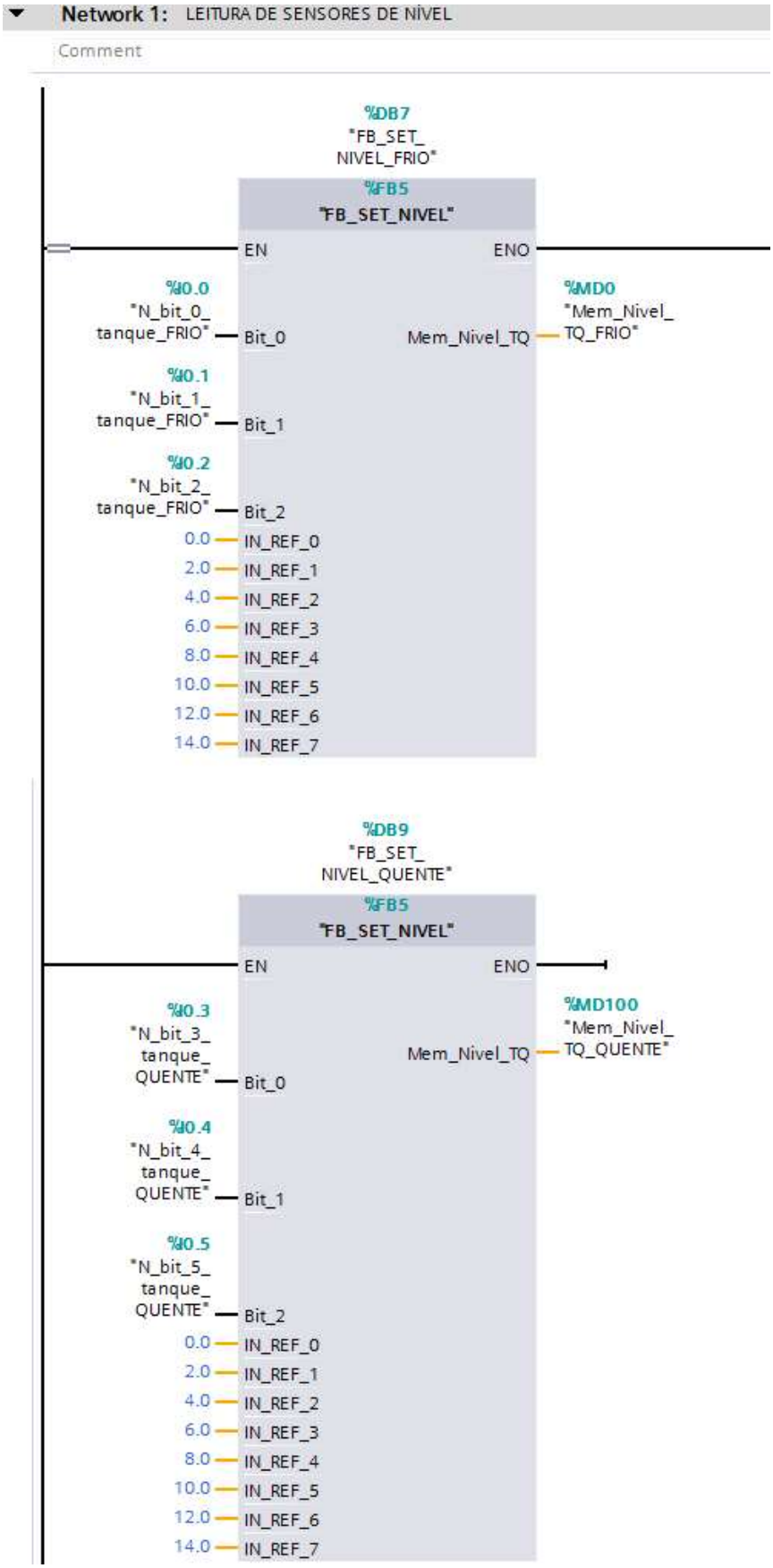
Tabela 3 - Variáveis utilizadas no OB de controle de nível.

VARIÁVEL	TIPO	ENDEREÇO	DESCRIÇÃO
N_bit_0_tanque_FRIO	Bool	%I0.0	Bit_0 de Entrada para Sensoriamento de Nível do Tanque Frio
N_bit_1_tanque_FRIO	Bool	%I0.1	Bit_1 de Entrada para Sensoriamento de Nível do Tanque Frio
N_bit_2_tanque_FRIO	Bool	%I0.2	Bit_2 de Entrada para Sensoriamento de Nível do Tanque Frio
N_bit_3_tanque_QUENTE	Bool	%I0.3	Bit_3 de Entrada para Sensoriamento de Nível do Tanque Quente
N_bit_4_tanque_QUENTE	Bool	%I0.4	Bit_4 de Entrada para Sensoriamento de Nível do Tanque Quente
N_bit_5_tanque_QUENTE	Bool	%I0.5	Bit_5 de Entrada para Sensoriamento de Nível do Tanque Quente
Mem_Nivel_TQ_FRIO	Real	%MD0	Memória que Armazena Valor do Nível (em Litros) do TQ Frio
Mem_Nivel_TQ_QUENTE	Real	%MD100	Memória que Armazena Valor do Nível (em Litros) do TQ Quente
Set_Margem_Nivel	Real	%MD110	Referência de Nível Desejado
Set_Ref_Nivel	Real	%MD140	Referência de Margem de Tolerância para Nível
Out_Bomba_Nivel	Bool	%M100.0	Memória de Acionamento de Bomba Vindo da Lógica de Nível
Out_Valvula_Nivel	Bool	%M110.0	Memória de Acionamento de Válvula Vindo da Lógica de Nível
Nivel_Com_Margem_UP	Real	%MD120	Nível com Margem Superior
Nivel_Com_Margem_DOWN	Real	%MD130	Nível com Margem Inferior

Fonte: Elaborado pelo autor

Considerando a limitação de apenas duas entradas analógicas no CLP e aliado à um almejado controle de nível refinado foi implementado sensoriamento de nível através de três entradas digitais (para cada sensor) possibilitando assim, através de uma tabela verdade, uma variação de oito níveis. Sua implantação foi realizada no *Network* 1 (Figura 43) do OB de nível e para programação deste sensoriamento foi utilizado o FB_SET_NIVEL, já mencionado anteriormente.

Figura 43- *Network* 1 do OB de controle de nível.

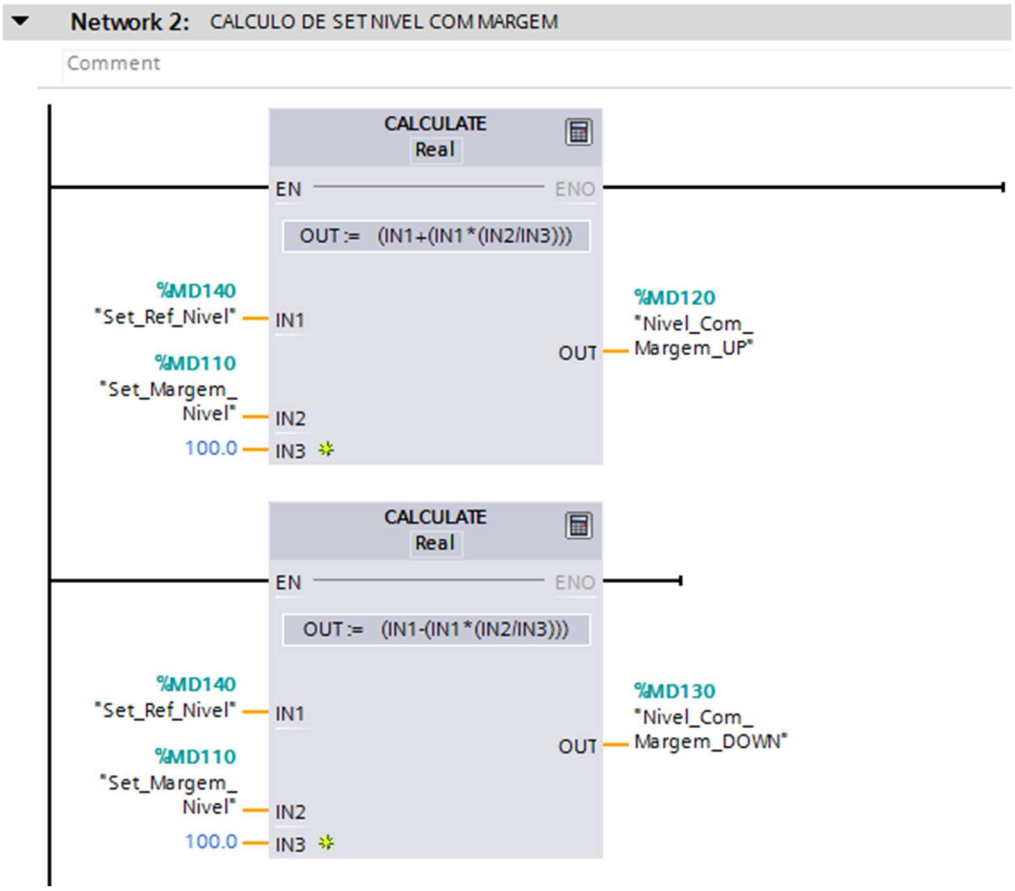


Fonte: Elaborado pelo autor

Os valores correspondentes a cada uma das graduações de nível estão representados na Tabela 1.

No segundo *Network* de nível (Figura 44), é realizada a leitura dos valores de nível inseridos pelo usuário bem como a margem de tolerância, junto a isto são calculados os valores de nível com margem superior e inferior. Esta implementação é feita utilizando FB_SET_MARGEM_UP e FB_SET_MARGEM_DOWN.

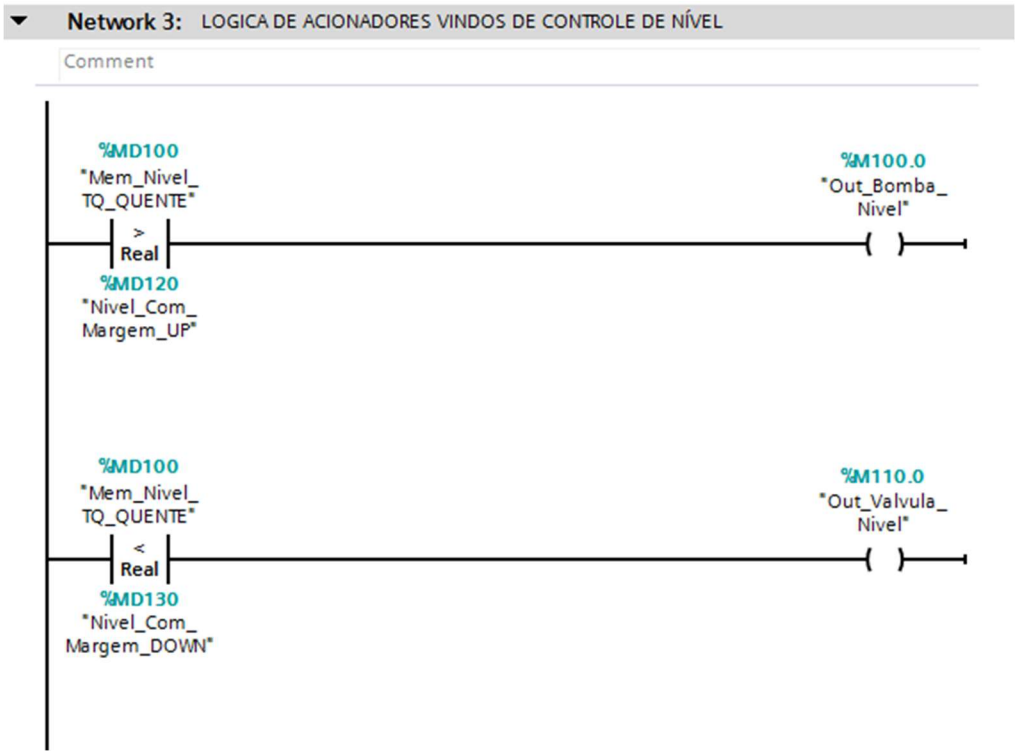
Figura 44 – *Network* 2 do OB de controle de nível.



Fonte: Elaborado pelo autor

O terceiro e último *Network* de nível (Figura 45) realiza a lógica de controle dos acionadores.

Figura 45 - *Network* 3 do OB de controle de nível.



Fonte: Elaborado pelo autor

Para o controle automático de temperatura são consideradas duas atuações.

- Caso o valor lido pelo sensor de nível do tanque quente seja superior ao valor inserido pelo usuário adicionado à margem de tolerância (nível do tanque quente está mais alto do que o desejado) a bomba é acionada, de modo a extrair parte do líquido contido no tanque inferior (tanque quente) diminuindo seu nível e aumentando do tanque superior (tanque frio);
- Caso o valor lido pelo sensor de nível do tanque quente seja inferior ao valor inserido pelo usuário subtraído a margem de tolerância (nível do tanque quente está mais baixo do que o desejado) a válvula é acionada, de modo a liberar líquido contido no tanque superior (tanque frio) aumentando o nível do tanque inferior (tanque quente).

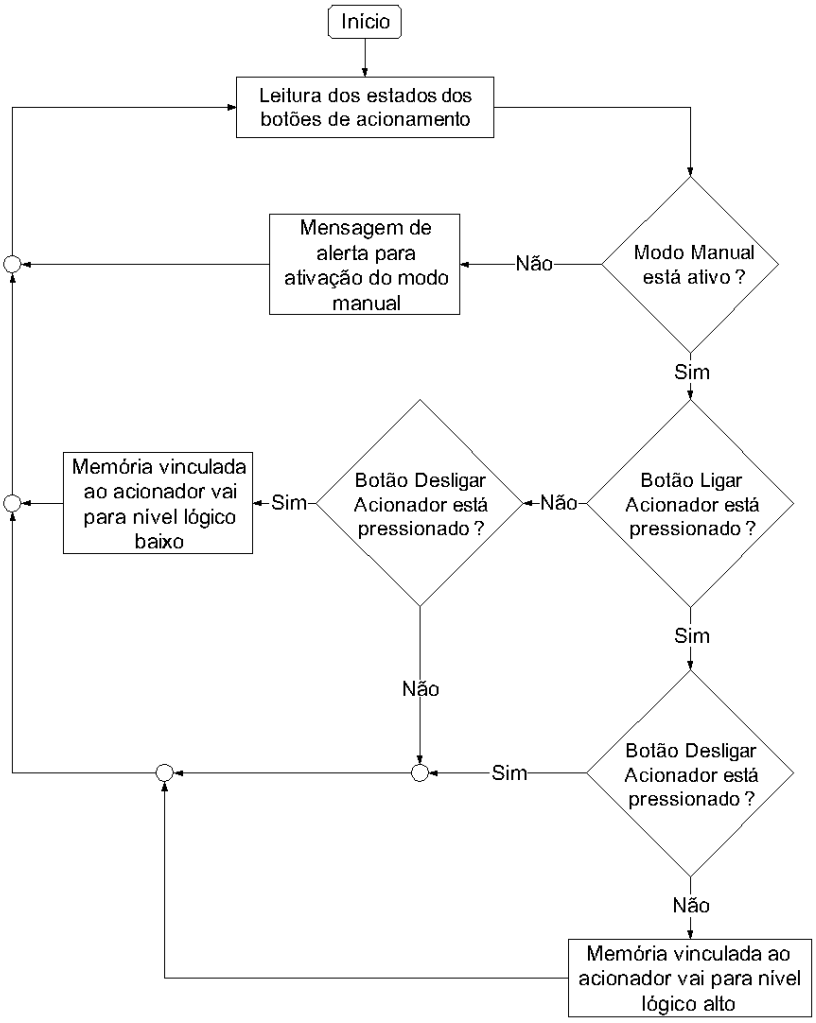
4.2.2.5 OB de Controle Manual

O OB de controle manual é o mais simples de toda modelagem, apenas associando botões inseridos na IHM com memórias de acionadores. A mesma lógica de acionamento foi

aplicada para o comando da bomba, do resistor e da válvula, como pode ser visto na Figura 47 e Figura 48.

O fluxograma apresentado na Figura 46 mostra a lógica de controle utilizada no OB de temperatura.

Figura 46 - Fluxograma de controle manual.



Fonte: Elaborado pelo autor

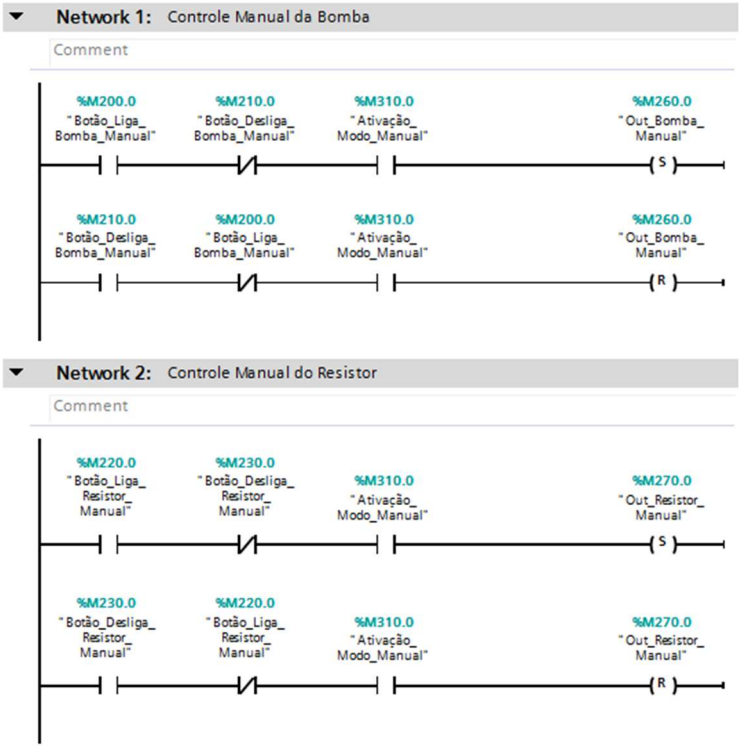
São apresentados na Tabela 4 o nome, tipo, endereço lógico bem como breve descrição da utilização das variáveis criadas para este OB.

Tabela 4 - Variáveis utilizadas no OB de controle de manual.

VARIÁVEL	TIPO	ENDEREÇO	DESCRIÇÃO
Botão_Liga_Bomba_Manual	Bool	%M200.0	Botão do Usuário para Ligar Bomba
Botão_Desliga_Bomba_Manual	Bool	%M210.0	Botão do Usuário para Desligar Bomba
Botão_Liga_Resistor_Manual	Bool	%M220.0	Botão do Usuário para Ligar Resistor
Botão_Desliga_Resistor_Manual	Bool	%M230.0	Botão do Usuário para Desligar Resistor
Botão_Liga_Valvula_Manual	Bool	%M240.0	Botão do Usuário para Ligar Válvula
Botão_Desliga_Valvula_Manual	Bool	%M250.0	Botão do Usuário para Desligar Válvula
Out_Bomba_Manual	Bool	%M260.0	Memória de Acionamento de Bomba Vindo do Controle Manual
Out_Resistor_Manual	Bool	%M270.0	Memória de Acionamento de Resistor Vindo do Controle Manual
Out_Valvula_Manual	Bool	%M280.0	Memória de Acionamento de Bomba Vindo do Controle Manual
Ativação_Modo_Manual	Bool	%M300.0	Memória Vinculada a Mensagem de Alerta de Ativação do Modo Manual
Alerta_Modo_Manual	Bool	%M310.0	Memória Vinculada ao Estado do Seletor de Modo Manual ON/OFF

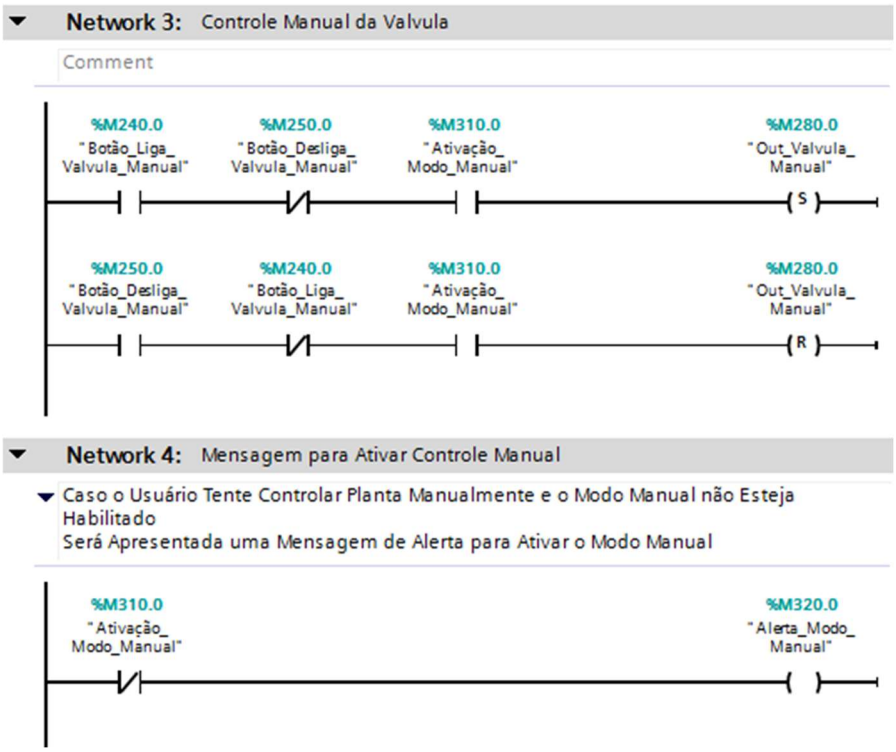
Fonte: Elaborado pelo autor

Figura 47 – *Network* 1 e 2 do OB de controle manual.



Fonte: Elaborado pelo autor

Figura 48 – *Network* 3 e 4 do OB de controle manual.



Fonte: Elaborado pelo autor

Para cada acionador foi vinculada uma saída *set* e outra *reset*, ambos acionamentos só são possíveis se o seletor de modo manual esteja ativo. A saída *set* é acionada caso seja pressionado o botão liga do acionamento do respectivo acionador e o botão desliga esteja desabilitado, já a saída *reset* é acionada caso seja pressionado o botão desliga do acionamento do respectivo acionador e o botão liga esteja desabilitado.

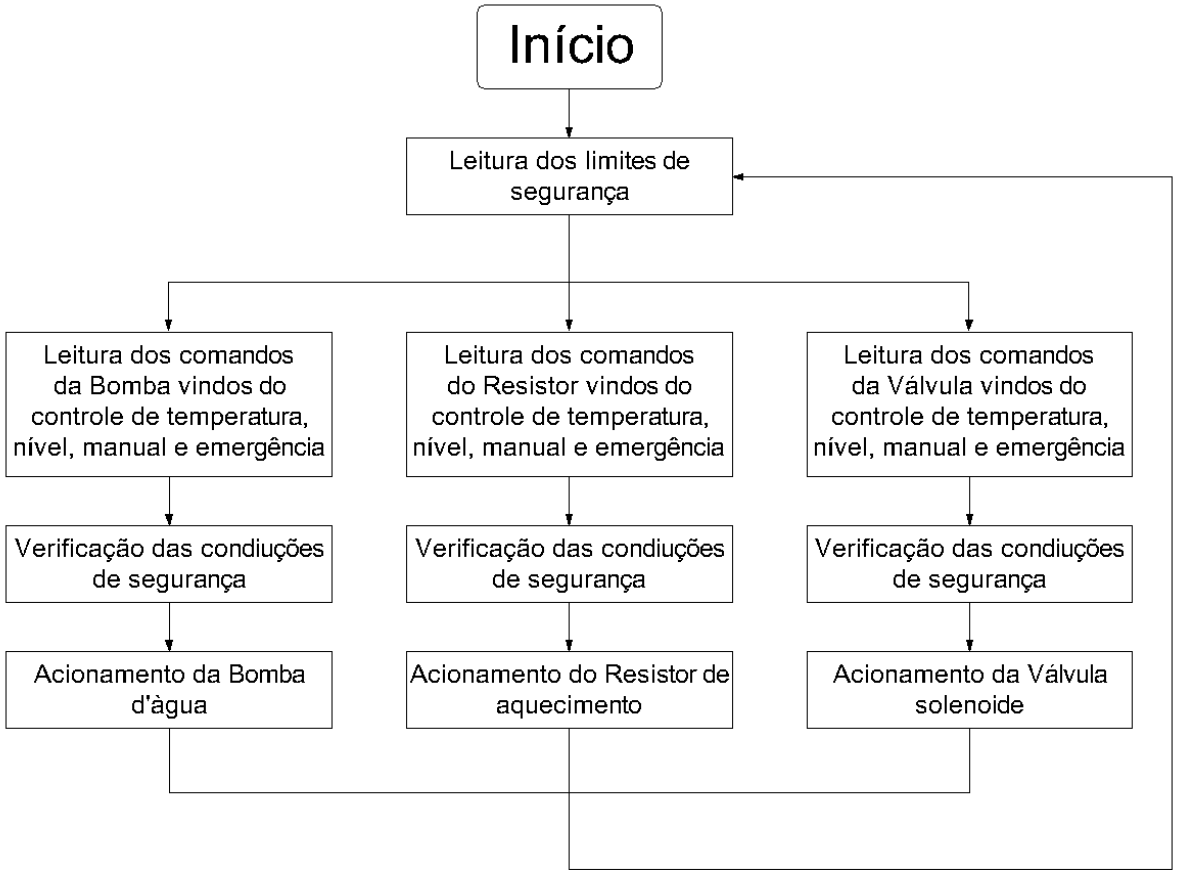
O último *Network* de controle manual é apenas para alertar o usuário para ativar o controle manual (caso ele ainda não esteja ativo).

4.2.2.6 OB de Controle de Geral

Este OB é responsável por realizar a leitura dos acionamentos de nível, temperatura e manual, verificar as condições de segurança para então acionar as saídas de controle.

O fluxograma apresentado na Figura 49 mostra a lógica de controle utilizada no OB de temperatura.

Figura 49 - Fluxograma de controle geral.



Fonte: Elaborado pelo autor

A Tabela 5 mostra o nome, tipo, endereço lógico bem como breve descrição da utilização das variáveis criadas para este OB.

Tabela 5 - Variáveis utilizadas no OB de controle de geral.

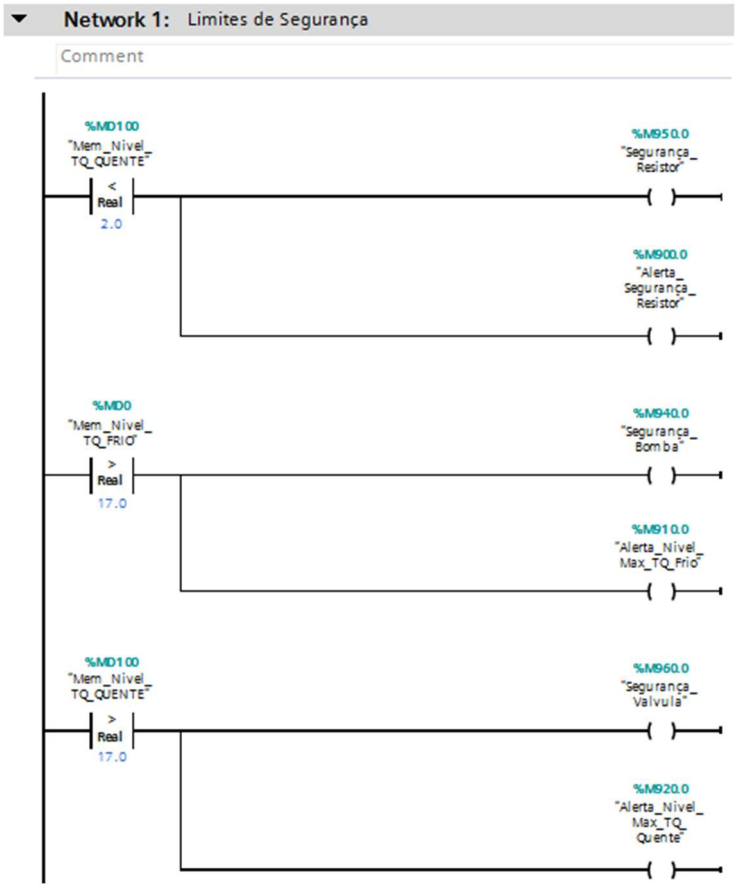
VARIÁVEL	TIPO	ENDEREÇO	DESCRIÇÃO
Alerta_Segurança_Resistor	Bool	%M900.0	Memória Vinculada à Mensagem de Alerta de Segurança do Resistor
Alerta_Nivel_Max_TQ_Frio	Bool	%M910.0	Memória Vinculada à Mensagem de Alerta de Segurança da Bomba (Transbordo do TQ Frio)
Alerta_Nivel_Max_TQ_Quente	Bool	%M920.0	Memória Vinculada à Mensagem de Alerta de Segurança da Válvula (Transbordo TQ Quente)
Botão_de_Emergencia	Bool	%M930.0	Botão do Usuário para Acionar o Modo de Emergência
Nivel_0_TQ_Quente	Bool	%I0.6	Entrada de Sensor de Boia Inferior do TQ Quente p/ Segurança do Resistor
Out_Bomba_Geral	Bool	%Q0.1	Saída de Controle de Acionamento da Bomba
Out_Resistor_Geral	Bool	%Q0.2	Saída de Controle de Acionamento do Resistor
Out_Valvula_Geral	Bool	%Q0.3	Saída de Controle de Acionamento da Válvula
Segurança_Bomba	Bool	%M940.0	Memória Responsável por Garantir a Segurança da Bomba
Segurança_Resistor	Bool	%M950.0	Memória Responsável por Garantir a Segurança do Resistor
Segurança_Valvula	Bool	%M960.0	Memória Responsável por Garantir a Segurança da Válvula

Fonte: Elaborado pelo autor

O controle geral da planta foi dividido em quatro *Network's*, sendo eles:

1. Limites de segurança - No primeiro *Network* foram levantados os limites de segurança da planta (Figura 50), que foram elencados como:
    - Segurança Resistor: Estabelece nível mínimo do tanque inferior (tanque quente) para acionamento do resistor, aliado a ele é acionada a mensagem de alerta de segurança do resistor;
    - Segurança Bomba: Impede acionamento da bomba caso o tanque superior (tanque frio) atinja seu nível máximo, com intuito de impedir transbordo do tanque e o acionamento da bomba sem líquido no tanque inferior (tanque quente), aliado a este comando tem-se a mensagem de alerta de segurança de nível máximo do tanque frio;
    - Segurança Válvula: Impede o acionamento da válvula caso o nível máximo do tanque inferior (tanque quente) seja atingido, visa evitar transbordo deste tanque, aliado a este comando tem-se a mensagem de alerta de segurança de nível máximo do tanque quente.

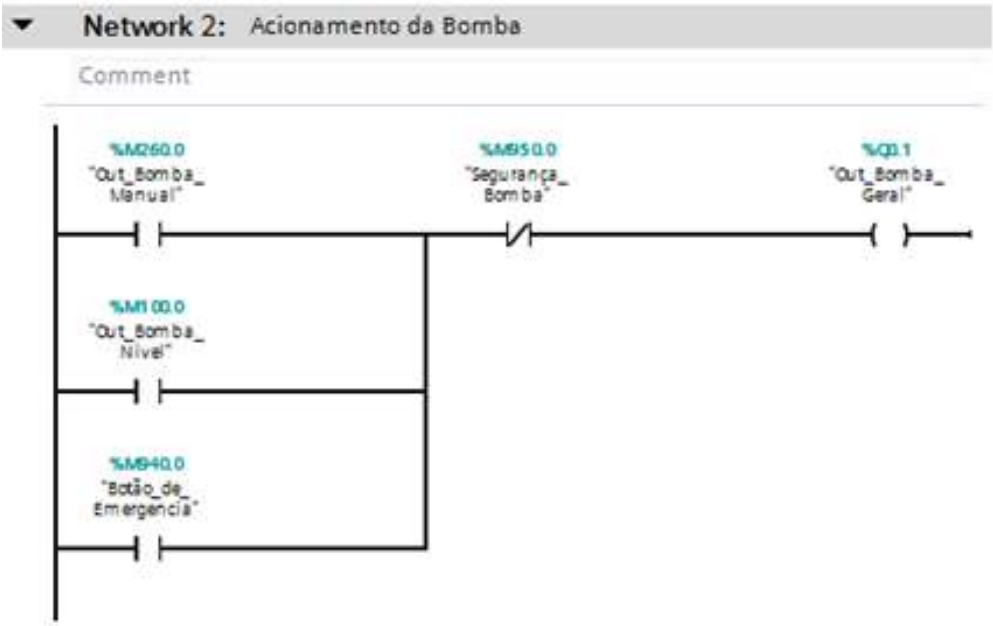
Figura 50 – *Network* 1 do OB de controle geral.



Fonte: Elaborado pelo autor

2. Acionamento da bomba (Figura 51) - Responsável por fazer varredura das memórias de acionamento de bomba vindos das lógicas de nível e manual e integrar com os limites de segurança já inseridos. Sua lógica de acionamento pode ser dividida em:
   - Acionamento via controle manual – caso usuário ligue manualmente a bomba via IHM;
   - Acionamento via controle de nível - caso a condição de acionamento de bomba vindo do controle de nível seja atendida (nível do tanque quente está mais alto do que o desejado);
   - Acionamento via botão de emergência - com intuito de circular o líquido para promover um rápido resfriamento.

Figura 51 – *Network* 2 do OB de controle geral.

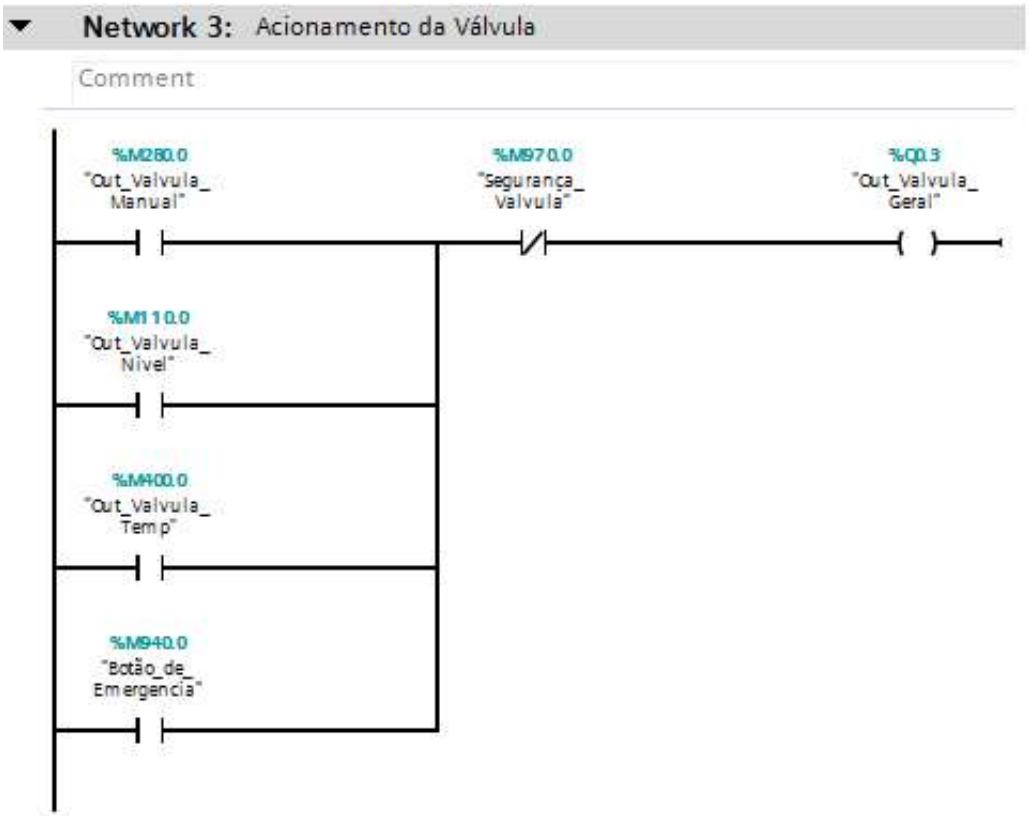


Fonte: Elaborado pelo autor

Para qualquer uma das lógicas de controle acionar a saída vinculada ao acionamento da bomba é necessário que o nível de segurança dela seja atingido.

3. Acionamento da válvula Figura 52 - Responsável por fazer varredura das memórias de acionamento de válvula vindos das lógicas de nível, temperatura e manual e integrar com os limites de segurança já inseridos. Sua lógica de acionamento pode ser dividida em:
    - Acionamento via controle manual – caso usuário ligue manualmente a válvula via IHM, desde que o modo manual esteja habilitado;
    - Acionamento via controle de nível - caso a condição de acionamento da válvula vindo do controle de nível seja atendida (nível do tanque quente está mais baixo do que o desejado);
    - Acionamento via controle de temperatura - caso a condição de acionamento da válvula vindo do controle de temperatura seja atendida (temperatura do líquido do tanque quente está mais alta do que a desejada);
    - Acionamento via botão de emergência - com intuito de circular o líquido para promover um rápido resfriamento.

Figura 52 – *Network* 3 do OB de controle geral.

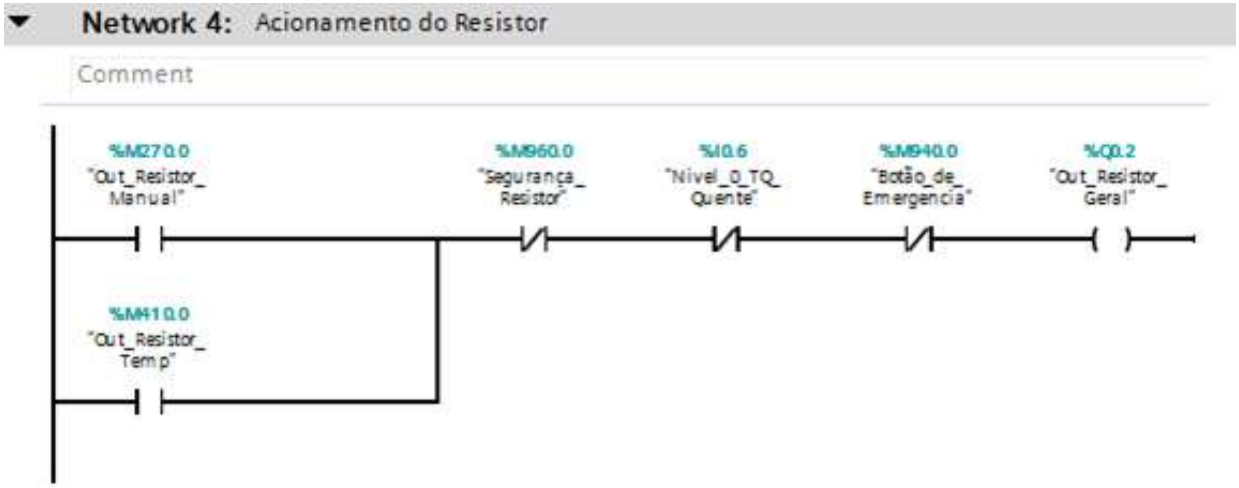


Fonte: Elaborado pelo autor

Para qualquer uma das lógicas de controle acionar a saída vinculada ao acionamento da válvula é necessário que o nível de segurança dela seja atingido.

4. Acionamento do resistor (Figura 53) - Responsável por fazer varredura das memórias de acionamento do resistor vindos das lógicas de temperatura e manual e integrar com os limites de segurança já inseridos. Sua lógica de acionamento pode ser dividida em:
    - Acionamento via controle manual – caso usuário ligue manualmente o resistor via IHM;
    - Acionamento via controle de temperatura - caso a condição de acionamento do resistor vindo do controle de temperatura seja atendida (temperatura do líquido do tanque quente está mais baixa do que a desejada).

Figura 53 – *Network 4* do OB de controle geral.



Fonte: Elaborado pelo autor

É válido notar que para qualquer uma das lógicas de controle acionar a saída vinculada ao acionamento do resistor é necessário que o nível de segurança deste seja atingido e o botão de emergência não pode estar acionado. Uma segurança extra foi inserida, um sensor de nível tipo boia na parte inferior do tanque quente deve estar acionado, de forma a atingir uma redundância na segurança do resistor, evitando assim a queima deste.

4.3 DISCUSSÃO

Concluída a explicação de toda programação executada nesse trabalho surge a oportunidade para discussão das metodologias empregadas e desafios encontrados.

Logo nos primeiros testes envolvendo a planta de sensores da De Lorenzo pôde-se notar um problema, provavelmente oriundo do circuito de alimentação, que levou à instabilidade ou não funcionamento dos sensores nela contidos, originando então a necessidade de utilização de sensores externos à planta.

Buscando uma solução viável economicamente e que atendesse aos padrões, senão de resistência ao ambiente fabril, ao menos em usabilidade, chegou-se a um grupo de componentes (sensor ultrassônico, sensor de temperatura submersível e bomba) compatível com placas Arduíno.

Paralelamente foi executado toda lógica de controle do CLP sem utilizar os sensores. O programa foi feito e testado por meio de potenciômetros (simulando as entradas analógicas de temperatura) e chaves (simulando os gatilhos de relés para sensoriamento de nível).

Dada a obtenção dos sensores externos iniciou-se, então, a busca de implementação de leitura e integração destes componentes através de um Arduíno Uno e uma placa de integração desenvolvida, porém ao término dessa etapa verificou-se a necessidade de maior número de portas digitais do tipo PWM para aumentar a gradação dos sensores de nível.

Esta necessidade levou à divisão da placa de integração em duas, uma para cada tanque e seus grupos de sensores. Para tal integração deixou de ser usada uma placa Arduino Uno para utilizar duas placas NodeMcu ESP-8266. Feito este último ajuste foi possível realizar os testes de comando lógico do CLP com sensores integrados.

Apesar de não ter sua execução implementada em ambiente fabril, este trabalho propôs se aproximar o máximo possível de uma solução encontrada no campo industrial. Pode-se dizer que atingiu o objetivo, pois toda modelagem e programação executada pode ser feita de maneira similar em uma planta industrial.

5 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Já não é preciso comprovar a necessidade da automação na indústria, pois é observado em toda gama de produção, desde os controles mais simples, como uma esteira de linha de produção, até os mais complexos, como solda de componentes de uma placa de circuito.

Este trabalho teve como objetivo a modelagem de uma planta industrial que visava o controle de temperatura e nível em malha fechada de um tanque, onde o usuário definia apenas nos valores desejados em ºC e litros, respectivamente.

Para execução desta modelagem foram utilizados CLP S7-1200 aliado à uma IHM KTP700, ambos da Siemens, de sensores de nível e de temperatura. A planta tinha como objetivo dar ao usuário o controle de nível e de temperatura ou controle manual dos acionadores da planta (bomba, resistor e válvula solenoide).

Alguns desafios foram encontrados no decorrer deste trabalho, como a impossibilidade de utilização da planta de sensores disponível. Para suprir tal impossibilidade foram aliados sensores externos, implementados através de placas NodeMcu ESP 8266 que realizaram a leitura de nível (através de sensores ultrassônicos) e de temperatura.

Os desafios encontrados ao longo da execução deste trabalho apenas evidenciam a necessidade de conhecimento amplo do profissional da área de automação, além de demonstrarem de maneira prática a versatilidade do CLP. Fica clara a possibilidade infinita de integração que ele pode proporcionar, o que justifica seu uso difundido na indústria.

A modelagem contida neste trabalho visou aproximar-se ao máximo dos padrões utilizados na indústria, porém devido a algumas limitações não foram utilizados apenas equipamentos industriais e mesmo assim houve boa resposta e integração.

Dada conclusão deste trabalho é possível elencar alguns novos estudos a serem executados, como a implementação de controle PID no controle de nível e temperatura, implementação de relatórios e gráficos do comportamento da planta e gerenciamento de usuários com acesso a configuração de níveis da planta.

REFERÊNCIAS

[1] PEREIRA, D. A. R. **Projeto de um sistema de automação industrial para uma indústria de produtos saneantes.** 85 p. Monografia (Graduação em Engenharia Automotivada); UnB, Brasília, DF, 2015

[2] STURARO, A. Z. **Automação de bombas dosadoras com auxílio de controlador lógico programável.** 58 p. Trabalho de conclusão de curso (Graduação em Engenharia Mecânica) – USF, Campinas, SP, 2009.

[3] DE MORAES, C. C.; CASTRUCCI, P. L. **Engenharia de Automação Industrial:** Hardware e Software, Redes de Petri, Gestão da Automação. Rio de Janeiro: LTC, 2010.

[4] ARAÚJO, D. Arquitetura básica do CLP. **Arquitetura básica do CLP**, Engmecatonico.blogspot.com, v. 03, p. Arquitetura, 25 maio 2011. Disponível em: https://engmecatonico.blogspot.com/2011/05/arquitetura-basica-do-clp.html. Acesso em: 8 jun. 2020.

[5] MARTINS, G. **Princípios da Automação Industrial**. 2012. Apostila (Engenharia Elétrica) - Universidade Federal de Santa Maria, [*S. l.*], 2012. Disponível em: https://pt.slideshare.net/sayonarasilva125/apostila-032012-clp. Acesso em: 2 mar. 2020.

[6] SILVA, M. E. **Controladores Lógicos Programáveis** - Ladder. EEP Escola de Engenharia de Piracicaba. 2007.

[7] SIMATIC, Siemens. Painéis Básicos de IHM SIMATIC. *In*: **Operação econômica e monitoramento na 2ª Geração**. [*S. l.*], 29 jul. 2019. Disponível em: https://new.siemens.com/br/pt/produtos/automacao/simatic-hmi/paineis/paineis-basicos.html. Acesso em: 17 mar. 2020.

[8] IEC - INTERNATIONAL ELECTROTECHNICAL COMMISSION. **Programmable controllers:** Part 3: Programming languages. 2. ed. Geneva, Switzerland, 2003.

[9] SIEMENS. **S7-1200 Programmable controller**: System Manual. Disponível em: https://cache.industry.siemens.com/dl/files/593/109741593/att_895681/v1/s71200_system_manual_en-US_en-US.pdf#page=83&zoom=100,73,125. Acesso em: 28 nov. 2019

[10] SIEMENS. **SIMATIC HMI:** HMI devices Basic Panels 2nd Generation. Disponível em:https://support.industry.siemens.com/cs/attachments/90114350/hmi_basic_panels_2nd_gen_operating_instructions_enUS_en-US.pdf?download=true. Acesso em: 28 nov. 2019

[11] SIEMENS. **6GK5005-0BA00-1AB2**. Disponível em: https://mall.industry.siemens.com/mall/en/ww/Catalog/Product/6GK5005-0BA00-1AB2. Acesso em: 28 nov. 2019

[12] ELETRONIC Shop. [*S. l.*], 7 jul. 2020. Disponível em: https://blogmasterwalkershop.com.br/embarcados/nodemcu/nodemcu-uma-plataforma-com-caracteristicas-singulares-para-o-seu-projeto-iot/. Acesso em: 28 nov. 2019.

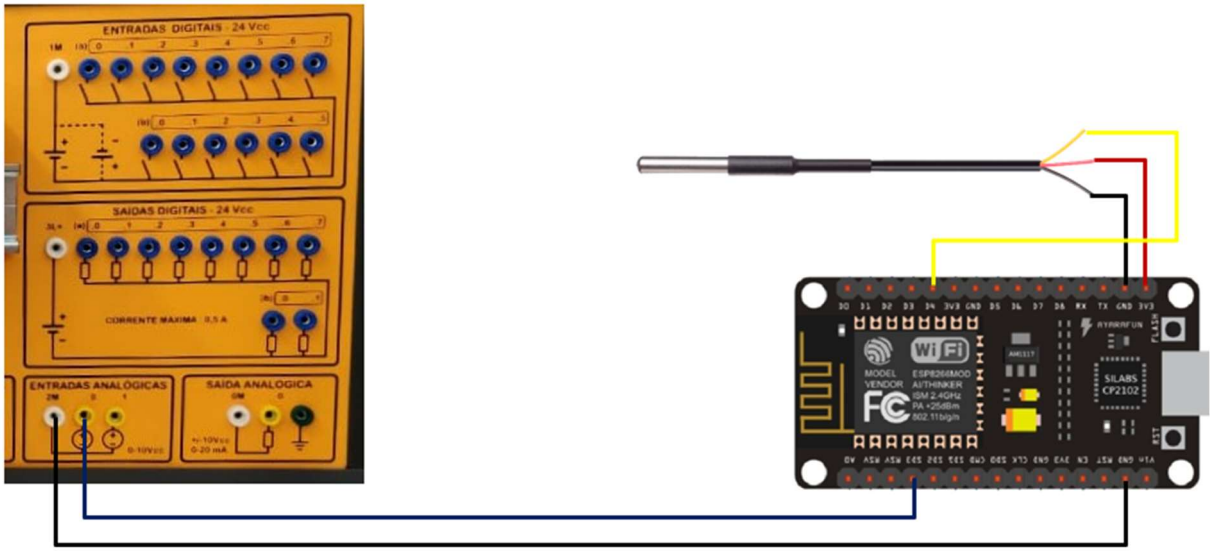
[13] LIVRE, Mercado. **Sensor De Temperatura Ds18b20 À Prova D`água Para Arduino**. Disponível em: https://produto.mercadolivre.com.br/MLB-820021391-sensor-de-temperatura-ds18b20-prova-dagua-para-arduino-_JM?quantity=1. Acesso em: 15 abr. 2020.

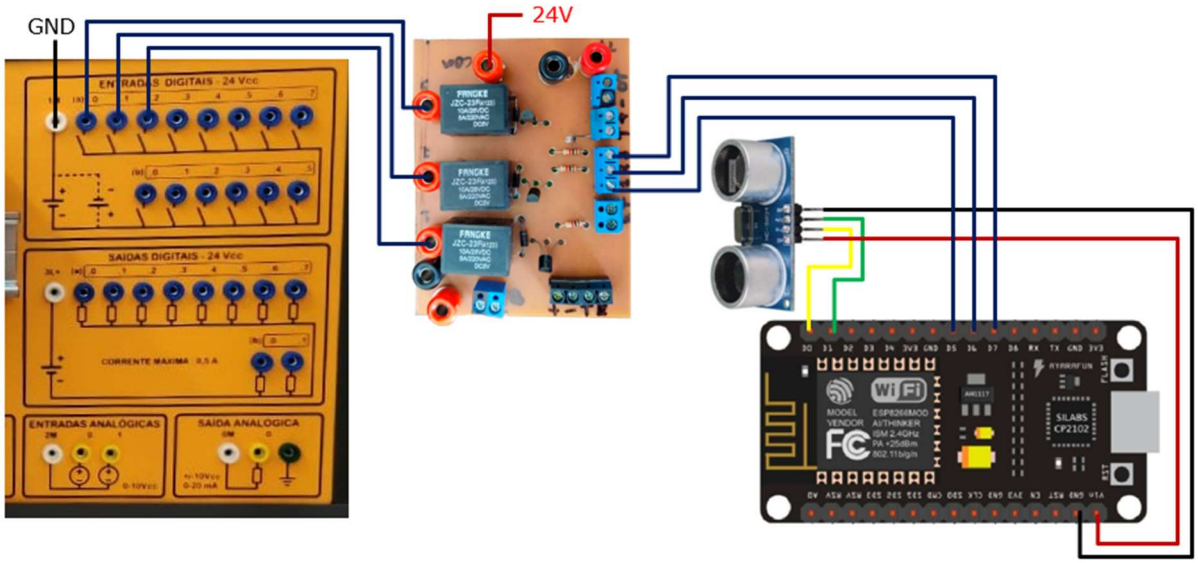
[14] CYTRON TECHNOLOGIES. **ProductUser's Manual:** HCSR04 Ultrasonic Sensor. Disponível em: http://web.eece.maine.edu/~zhu/book/lab/HC-SR04%20User%20Manual.pdf. Acesso em: 15 abr. 2020.

[15] ELETRODEX. **Mini Bomba de água p/ Arduino com motor RS-385 12V** Disponível em: https://www.eletrodex.com.br/mini-bomba-de-agua-p-arduino-com-motor-rs-385-12v.html. Acesso em: 15 abr. 2020.

[16] DE LORENZO do Brasil. **Guia do Usuário:** Sistema didático para estudo de controlador lógico programável.

APÊNDICE A – DIAGRAMA DE BLOCOS DE LIGAÇÃO DE SENSORES, PLACA NODEMCU E CLP





APÊNDICE B – PROGRAMAÇÃO DESENVOLVIDA NO ARDUINO IDE PARA PLACA NODEMCU ESP 8266

```
//TEMPERATURA
#include <OneWire.h>
#include <DallasTemperature.h>
//DISTANCIA
int trigPin1=16; //D0
int echoPin1=5; //D1
OneWire pino(2); //D4
DallasTemperature barramento(&pino);
//DeviceAddress Probe02 = {0x28, 0xEA, 0x8D, 0x79, 0x97, 0x19, 0x03, 0x0A};
DeviceAddress Probe01 = {0x28, 0x17, 0x1F, 0x79, 0x97, 0x18, 0x03, 0xC2};
void setup(void)
{
  Serial.begin(9600);
   //TEMPERATURA
     barramento.begin();
  //DISTANCIA
      pinMode(trigPin1, OUTPUT);
      pinMode(echoPin1, INPUT);
     // pinMode(trigPin2, OUTPUT);
     // pinMode(echoPin2, INPUT);
      pinMode(14, OUTPUT);  //D5
      pinMode(12, OUTPUT);   //D6
      pinMode(13, OUTPUT);   //D7
}
void loop()
{
  //TEMPERATURA
  barramento.requestTemperatures();
 float temperatura = barramento.getTempC(Probe01);
 float regra1;
  Serial.print("sensor1:   ");
  Serial.println(temperatura);
  regra1 = map(temperatura,0,100,0,255);
  //Serial.print("Temperatura 1: ");
  //Serial.println(regra1);
  analogWrite(10,regra1);   //SD3
  //Serial.println(" ");
//DISTANCIA
 long duration1, distance1;
  digitalWrite(trigPin1, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin1, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin1, LOW);
  duration1 = pulseIn(echoPin1, HIGH);
  distance1 = (duration1/2) / 29.1;
  float distancia_total_tq1 = 31;
  float nivel1 = 20 - (20*distance1 / distancia_total_tq1) ;
  if(nivel1 <=0) {nivel1=0;}
  Serial.print("distancia: ");
  Serial.println(distance1);
           if(nivel1<=0){digitalWrite(14,LOW);digitalWrite(12,LOW);digitalWrite(13,LOW);}
           else if(0<nivel1 && nivel1<=2){digitalWrite(14,LOW);digitalWrite(12,LOW);digitalWrite(13,HIGH);}
           else if(2<nivel1 && nivel1<=4){digitalWrite(14,LOW);digitalWrite(12,HIGH);digitalWrite(13,LOW);}
           else if(4<nivel1 && nivel1<=6 ){digitalWrite(14,LOW);digitalWrite(12,HIGH);digitalWrite(13,HIGH);}
           else if(6<nivel1 && nivel1<=8){digitalWrite(14,HIGH);digitalWrite(12,LOW);digitalWrite(13,LOW);}
           else if(8<nivel1 && nivel1<=10 ){digitalWrite(14,HIGH);digitalWrite(12,LOW);digitalWrite(13,HIGH);}
           else if(10<nivel1 && nivel1<=12){digitalWrite(14,HIGH);digitalWrite(12,HIGH);digitalWrite(13,LOW);}
           else{digitalWrite(14,HIGH);digitalWrite(12,HIGH);digitalWrite(13,HIGH);}
  delay(1000);
}
```