UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO DE ESPECIALIZAÇÃO EM AUTOMAÇÃO INDUSTRIAL

ALVARO CEZAR PARIETTI FILHO

APLICAÇÃO DE CONCEITOS DA NORMA IEC 61131-3 NA OTIMIZAÇÃO DE BLOCOS DE FUNÇÕES NA PROGRAMAÇÃO DE CLP

MONOGRAFIA DE ESPECIALIZAÇÃO

CURITIBA
2016

ALVARO CEZAR PARIETTI FILHO

APLICAÇÃO DE CONCEITOS DA NORMA IEC 61131-3 NA OTIMIZAÇÃO DE BLOCOS DE FUNÇÕES NA PROGRAMAÇÃO DE CLP

Monografia de Especialização, apresentado ao Curso de Especialização em Automação Industrial, do Departamento Acadêmico de Eletrônica, da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Especialista.

Orientador: Prof. Dr. Guilherme Alceu Schneider

CURITIBA
2016

RESUMO

PARIETTI FILHO, Alvaro C. **Aplicação de conceitos da norma IEC 61131-3 na otimização de blocos de funções na programação de CLP.** 2016. 54 f. Monografia (Curso de Especialização em Automação Industrial). Departamento Acadêmico de Eletrônica, Universidade Tecnológica Federal do Paraná. Curitiba, 2016.

A automatização de linhas de produção em indústrias é, em geral, realizada por meio de controladores lógicos programáveis (CLPs). As principais fabricantes deste tipo de dispositivo permitem que os programadores agrupem determinados trechos da aplicação em blocos de funções (FBD), para que seja possível reutilizá-los em outros projetos. Contudo, quando aspectos da automação sofrem mudanças nas indústrias e os blocos não acompanham esta evolução, esforços valiosos são aplicados em tarefas repetitivas, que poderiam ser evitadas através da atualização dos blocos. Deste modo, este trabalho visa a otimização de blocos de funções de programas de CLP aplicados em uma indústria de processamento de café, para reduzir o tempo de automação no desenvolvimento de sistemas, sem perda de aspectos de segurança. A base teórica do estudo foi a norma IEC 61131-3, com uma revisão dos seus principais conceitos relacionados a blocos de funções. Foram tomados para análise 2 blocos principais de dispositivos e apresentados os resultados de otimização.

Palavras-chave: IEC 61131-3. Blocos de funções. Padronização. Otimização.

ABSTRACT

PARIETTI FILHO, Alvaro C. **Application of concepts of IEC 61131-3 in function blocks to optimize PLC programming**. 2016. 54 f. Monografia (Curso de Especialização em Automação Industrial). Departamento Acadêmico de Eletrônica, Universidade Tecnológica Federal do Paraná. Curitiba, 2016.

The automatization of production lines in industries is, in general, made through programmable logic controllers (PLCs). The main manufacturers of this kind of devices allows the grouping of some code sections in function block diagrams (FBD), to allow the reuse in others projects. However, when aspects of automation undergo changes in the industries and the blocks do not follow this trend, valuable efforts are applied on repetitive tasks that could be avoided by updating blocks. Thus, this work aims to optimize some function block diagrams used by a coffee processing industry, to reduce the time applied in the development of systems, with no loss of safety aspects. The theoretical basis was the IEC 61131-3 standard, with a review of the main concepts related to function block diagrams. Were taken for analysis 2 main blocks of devices and the results of the optimization were presented.

Keywords: IEC 61131-3. Function blocks . Standardization. Optimitazion.

LISTA DE FIGURAS

Figura 1: principais torradores produzidos pela Probat Leogap. À esquerda, torrador Turbo 4000; à direita , torrador Basic 2000. 14
Figura 2: principais moinhos produzidos pela Probat Leogap. À esquerda, moinho MRM 500; à direita, moinho MRAL 2500. 14
Figura 3: esquema simplificado de indústria de café (fora de escala).....15
Figura 4: exemplo de tela de supervisório da ala de café torrado.....16
Figura 5: controlador CompactLogix 1769-L33ER e IHM PanelView Plus 1000.....17
Figura 6: fluxograma do processo.....18
Figura 7: ilustração do sistema.....19
Figura 8: fluxograma para criação de motores nos programas em estudo 21
Figura 9: representação simplificada do bloco de um motor simples.....22
Figura 10: representação simplificada do bloco de um motor reversível.....23
Figura 11: representação simplificada do bloco de alarme de disjuntor.....25
Figura 12: representação simplificada do bloco de filtragem de sinais de sensores.26
Figura 13: representação simplificada do bloco de alarmes de sensores 27
Figura 14: tela de controle de motores do sistema em estudo.....28
Figura 15: representação simplificada do bloco de registro pneumático.....29
Figura 16: representação simplificada do bloco de alarmes de registros pneumáticos 31
Figura 17: tela de controle de registros pneumáticos do sistema em estudo.....32
Figura 18: bloco para criação de motores nos novos programas.....34
Figura 19: nova representação simplificada do bloco de motores.....35
Figura 20: nova tela para controle de motores.....38
Figura 21: tela auxiliar para configurações extras do novo bloco de motor.....39
Figura 22: representação simplificada do bloco principal de registros pneumáticos.40
Figura 23: nova tela de controle de registros pneumáticos 42
Figura 24: tela auxiliar para configurações extras do registro pneumático.....44
Figura 25: bloco principal de um motor simples 49
Figura 26: bloco de alarme de disjuntor 49
Figura 27: bloco de alarme por sensor fim de curso 49
Figura 28: bloco de filtro de sensores 50
Figura 29: bloco principal de registros pneumáticos 50
Figura 30: bloco de alarmes de registro pneumático.....50
Figura 31: novo bloco principal de motores.....51
Figura 32: novo bloco principal de registros pneumáticos.....52

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

CLP	Controlador Lógico Programável
FBD	*Function Block Diagrams*
IEC	*International Eletrotechnical Comission*
IHM	Interface Homem Máquina
PID	Proporcional-Integral-Derivativo
POU	*Program Organization Unit*
NA	Normalmente aberto
RP	Registro pneumático

SUMÁRIO

1 INTRODUÇÃO 7

1.1 PROBLEMA 8
1.2 OBJETIVOS 8
1.2.1 Objetivo Geral 8
1.2.2 Objetivos Específicos 8
1.3 JUSTIFICATIVA 9
1.4 ESTRUTURA DO TRABALHO 9

2 NORMA IEC 61131-3 11

3 OTIMIZAÇÃO DE BLOCOS DE FUNÇÕES 13

3.1 PRINCIPAIS COMPONENTES DE UMA INDÚSTRIA DE TORREFAÇÃO DE CAFÉ 13
3.2 HARDWARE E *SOFTWARE* 16
3.3 SISTEMA DE ESTUDO 17
3.4 BLOCOS DE FUNÇÕES ANALISADOS 19
3.4.1 Blocos para motores 20
3.4.2 Blocos para registros pneumáticos 29
3.5 BLOCOS DE FUNÇÕES PROPOSTOS 33

4 RESULTADOS 34

4.1 BLOCOS PARA MOTORES 34
4.2 BLOCOS PARA REGISTROS PNEUMÁTICOS 40

5 CONSIDERAÇÕES FINAIS 45

REFERÊNCIAS 47

APÊNDICE A – Blocos de motores e registros pneumáticos anteriormente utilizados em sistemas Probat Leogap 49

APÊNDICE B – Novos blocos de motores e registros pneumáticos utilizados pela Probat Leogap 51

1 INTRODUÇÃO

O desenvolvimento de programas de CLP que sejam simples, confiáveis e reaproveitáveis, torna os sistemas menos propensos a apresentar problemas de lógica, mais rápidos de serem programados, com tempo de *startup* reduzido, confiáveis e flexíveis (PLCOPEN, 2004). Do ponto de vista dos usuários, os equipamentos que dispõem de telas de IHM com informações relevantes, claras e objetivas, tendem a tornar as atividades de manutenção e operação mais rápidas, precisas e intuitivas. Observando estas necessidades e a falta de padronização, um grupo composto de representantes das principais fabricantes de controladores para automação industrial propôs a norma IEC 61131-3 (IEC, 2003).

Embora a primeira edição tenha sido publicada em 1993, a IEC 61131-3 ainda tem sua redação e facilidades desconhecidas por parte dos engenheiros e programadores da automação industrial. De forma simplificada (mais detalhes são apresentados no capítulo 2), a norma sugere a padronização de boas práticas para o desenvolvimento de programas de CLPs, visando qualidade e economia de tempo durante as fases de desenvolvimento e testes.

Em literaturas, encontramos bons exemplos de aplicação da norma. Faustino (2005) apresenta, em sua dissertação, um caso de adequação do programa de navios-varredores da Marinha do Brasil, utilizando a norma IEC 61131-3. Hamilton (1993) recomenda a utilização da norma como ferramenta de auxílio ao engenheiro de testes, por produzir uma “curva de aprendizagem reduzida, uma metodologia consistente de programação e alta confiabilidade”. A abordagem da norma não se limita ao ambiente industrial, sendo aplicada também como ferramenta educacional, como pode ser visto no trabalho de Verwer (1999).

Diante deste panorama de aceitação da IEC 61131-3, o tema desta monografia surge da necessidade de atualização dos FBDs utilizados em uma indústria, onde os avanços dos requisitos de automação ocasionaram um aumento do tempo de desenvolvimento e testes de *softwares*. Os blocos propostos foram testados em linhas de transporte de produtos, em uma indústria processadora de grãos de café.

1.1 PROBLEMA

O tempo de desenvolvimento de um *software* de CLP muitas vezes influencia diretamente os objetivos de uma empresa. Se feito às pressas, pode levar o programador a realizar simulações sem o nível de detalhamento necessário. Isso pode levar a um gasto de tempo além do previsto durante as etapas de comissionamento e *startup*, além do fornecimento de uma documentação incompleta (FENTON, 2014).

Em se tratando de operação, as IHMs podem ser grandes vilãs quando as informações são de difícil acesso ou quando as telas não informam com clareza as falhas que ocorrem. Estas situações tornam as manutenções mais demoradas e impossibilitam uma tomada de decisão rápida por parte do operador (MYERS *et al.*, 2015).

1.2 OBJETIVOS

Nesta seção são apresentados os objetivos geral e específicos do trabalho, relativos ao problema apresentado no tópico anterior.

1.2.1 Objetivo Geral

Aplicar conceitos da norma IEC 61131-3 no desenvolvimento de sistemas de transporte de café, com ênfase na redução do tempo de programação, manutenção e operação das IHMs.

1.2.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Estudar a norma IEC 61131-3;
- Revisar os procedimentos de programação de CLPs dos sistemas já existentes;
- Revisar os recursos disponíveis já existentes para os operadores nas IHMs;

- Alterar, se aplicável e com base na norma estudada, os procedimentos de programação de CLPs;
- Otimizar os recursos disponíveis para os operadores nas IHMs, se necessário;
- Implementar e simular as alterações em um projeto;
- Validar as implementações em campo.

1.3 JUSTIFICATIVA

Este trabalho apresenta um estudo de melhoria aplicado diretamente em chão de fábrica. Inicialmente são considerados os blocos de CLP já utilizados por uma fabricante de máquinas que, embora funcionais, necessitavam de manutenção. Uma vez que a IEC 61131-3 aborda a importância da padronização dos programas, os conceitos dessa norma foram utilizados como base para esta monografia, de forma a enriquecer a escassa base de dados referente aos aspectos práticos do assunto.

Os blocos de CLP otimizados possuem funcionalidades específicas. Estas funcionalidades contribuem para que engenheiros de automação e programadores encontrem ideias e soluções para as etapas de programação, comissionamento, *startup*, operação e manutenção dos sistemas automatizados.

1.4 ESTRUTURA DO TRABALHO

O trabalho terá a estrutura abaixo apresentada.

Capítulo 1 - Introdução: apresentação do tema, do problema, dos objetivos da pesquisa, a justificativa e a estrutura geral do trabalho;
Capítulo 2 – Norma IEC 61131-3: serão abordados aspectos históricos e de aplicação da norma cujos conceitos foram utilizados neste trabalho;
Capítulo 3 – Otimização de blocos de funções: será abordado o desenvolvimento do trabalho, englobando os materiais e métodos utilizados e os locais de desenvolvimento e aplicação do programa otimizado;

Capítulo 4 – Resultados: serão apresentados os resultados do desenvolvimento e da aplicação do sistema otimizado, inclusive com questionários respondidos por programadores e técnicos de manutenção.

Capítulo 5 – Considerações finais: será avaliada a viabilidade da proposta de solução do problema, através da análise dos resultados e das informações repassadas pelos envolvidos, tais como programadores e técnicos de manutenção.

2 NORMA IEC 61131-3

Este capítulo apresenta conceitos e definições a respeito da norma IEC 61131-3. Serão abordados aspectos referentes à motivação da criação da norma, casos de sucesso que utilizaram a norma como base e, também, as vantagens e as desvantagens.

A norma IEC 61131 foi redigida pelo grupo de trabalho SC65B WG7 da IEC, que é constituído por representantes de diferentes fabricantes de dispositivos para automação (JOHN et al., 2001). A IEC 61131-3 é parte da IEC 61131, e sua criação foi motivada pelo aumento da capacidade de armazenamento e processamento dos controladores, cujas aplicações passaram a utilizar muitas linhas de código, de forma desestruturada. Esta característica dificultava a manutenção e a reutilização do *software.*

A IEC 61131 traz requisitos de *hardware* e *software* para sistemas que envolvam CLPs (SEIXAS FILHO, 1999) e é dividida em cinco partes:

- Parte 1: IEC 61131-1 Informações gerais;
- Parte 2: IEC 61131-2 Requisitos de *hardware;*
- Parte 3: IEC 61131-3 Linguagens de programação;
- Parte 4: IEC 61131-4 Guia de orientação ao usuário;
- Parte 5: IEC 61131-5 Comunicação.

Este trabalho abordará a parte 3 (IEC 61131-3), que apresenta um guia (não são regras) para a programação de controladores. Sendo o grupo de desenvolvimento da norma composto pelos principais fabricantes, as diretrizes são aceitas pela grande maioria dos desenvolvedores de *hardware* e *software* para CLPs, proporcionando padronização e sinergia. Estas características levam a uma redução dos custos de treinamento de programadores (JOHN et al., 2001).

Apesar da parte 3 da norma apresentar novidades relacionadas a tipos de dados, variáveis, configurações, recursos, tarefas e linguagens de programação, a principal vantagem que será destacada neste trabalho e é evidenciada por Bottura Filho (2000), está na capacidade de criar elementos reutilizáveis e personalizados

pelo usuário, os chamados POUs, divididos em três partes: função, bloco de função e programa.

Algumas funções padrões são definidas, tais como operações de soma, multiplicação, seno e cosseno. Para cada entrada da função, existe uma única saída; já os blocos de funções (FBD) possuem memória e executam rotinas de controle. Eles podem ser considerados como “caixas pretas”, uma vez que sua arquitetura interna não importa, mas sim suas entradas e saídas (PLCOPEN, 2016). Para uma mesma entrada em um FBD, saídas diferentes podem ser obtidas. O PID é um exemplo de bloco de função; finalmente, os programas são compostos por redes de funções e blocos de funções, que trocam informações entre si e acessam as entradas e saídas do controlador, permitindo o acesso às demais POUs (BRAGA, 2005; JOHN et al, 2001).

Em sua tese, Faustino (2005) explora as vantagens da otimização de FBDs, ao modernizar o sistema de varredura de navios da Marinha do Brasil. Ele aponta que a estrutura e a consistência dos novos blocos facilitaram os procedimentos de teste, validação e manutenção da aplicação, com redução de homens-hora. Opinião semelhante é compartilhada por Hamilton (1993), que ressalta a possibilidade de reutilização dos blocos personalizados, visto que uma vez testado e aprovado, basta apenas inserir o bloco na aplicação, dispensando novos testes.

A principal desvantagem da IEC 61131-3 está na portabilidade de *software* entre os diferentes fabricantes de CLPs, pois nenhum formato de dados para exportação foi definido. Na prática, significa que a migração de programas para um controlador de um fabricante diferente necessita de manipulação (ALLEN-BRADLEY, 2016).

3 OTIMIZAÇÃO DE BLOCOS DE FUNÇÕES

Este capítulo apresenta o sistema onde o trabalho foi aplicado, bem como detalha os FBD estudados. A seção 3.1 traz uma visão geral dos principais componentes de uma indústria de café, com explicações sucintas. A seção 3.2 apresenta o *hardware* e *software* utilizados para o desenvolvimento do trabalho. Na seção 3.3 é explicado o sistema em que o trabalho foi desenvolvido e, por fim, a seção 3.4 detalha os blocos de funções analisados.

Os blocos de funções alvos do estudo deste trabalho fazem parte dos padrões de automação do sistema de transporte de produtos da Probat Leogap. Trata-se de uma indústria metalúrgica focada em soluções automatizadas para o ramo alimentício, com destaque para as indústrias de café. Os sistemas de automação desenvolvidos vão desde o controle de equipamentos até o controle total de plantas de torrefação de café.

3.1 PRINCIPAIS COMPONENTES DE UMA INDÚSTRIA DE TORREFAÇÃO DE CAFÉ

É ampla a gama de equipamentos que podem ser encontrados em uma indústria de café, dependendo do volume da produção e do tipo de produto. Os equipamentos mais comuns são os torradores (Figura 1) e os moinhos (Figura 2), produzidos em série. Como não há alterações significativas de *software*, o custo de programação de equipamentos pode ser considerado como zero, dispensando a aplicação dos blocos utilizados neste trabalho.

Figura 1: principais torradores produzidos pela Probat Leogap. À esquerda, torrador Turbo 4000; à direita , torrador Basic 2000.
Fonte: Probat Leogap (2016a).

Figura 2: principais moinhos produzidos pela Probat Leogap. À esquerda, moinho MRM 500; à direita, moinho MRAL 2500.
Fonte: Probat Leogap (2016b).

Grandes indústrias de torrefação de café são divididas em alas. A Figura 3 apresenta um esquema simplificado da planta de uma torrefação de café, separada e numerada de acordo com a ala.

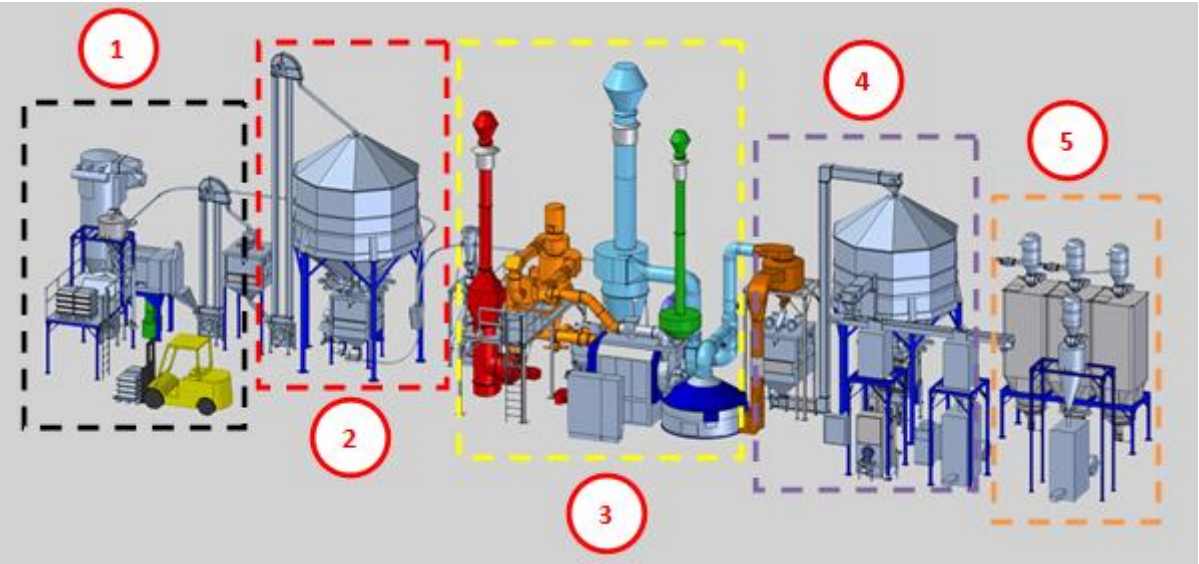

Figura 3: esquema simplificado de indústria de café (fora de escala)
Fonte: Probat Leogap (2016c).

1) Recebimento: é a entrada de produto na indústria. As formas mais comuns de recebimento de café são por caminhões, cuja carga é pesada em balanças automáticas ou por sacas (*big bags*), cujo peso é previamente determinado (entrada manual de peso);

2) Silos de café cru: após o recebimento, o café é armazenado nos silos de café cru. O sistema de automação registra dados importantes do produto, como tipo de café e peso;

3) Torrefação: nesta ala encontram-se os torradores, que recebem o café cru e produzem o café torrado, de acordo com a receita;

4) Silos de café torrado: após o processo de torra, o café é enviado para armazenagem nos silos de café torrado. Assim como no café cru, o sistema registra dados importantes do produto. Um exemplo de tela do supervisório de café torrado é apresentado na Figura 4. O trecho marcado com o número 1 ilustra a saída do torrador. O trecho 2 representa o transporte do produto até uma balança, enquanto o trecho 3 mostra o caminho a ser percorrido pelo café até atingir os silos de café torrado.

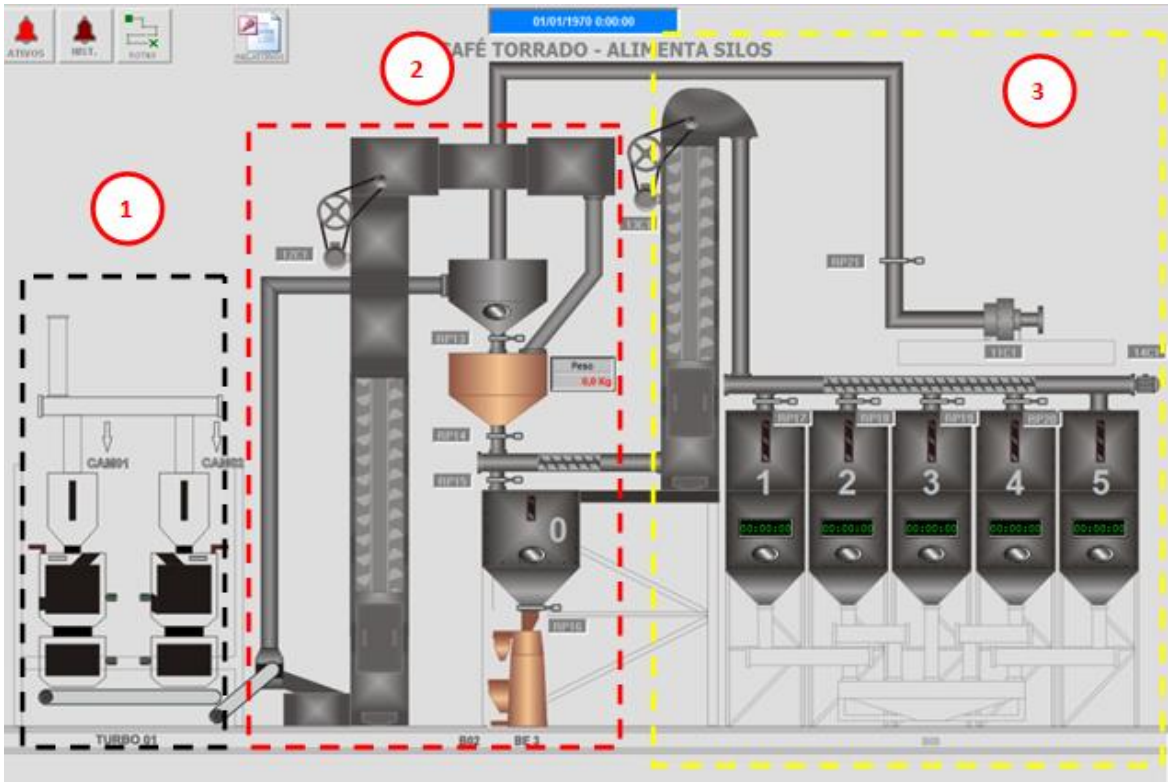

Figura 4: exemplo de tela de supervisório da ala de café torrado
Fonte: autoria própria

5) Moagem: nesta ala encontram-se os moinhos, responsáveis pela produção do café moído. Este tipo de produto é o responsável pela maioria das vendas das torrefações de café. Algumas indústrias têm a opção de não moer, e fornecem o produto torrado e em grãos.

Para que o conjunto de alas opere corretamente, o programa de controle de cada setor deve ser confiável. Os blocos que serão apresentados neste trabalho são aplicados nos programas dos sistemas que transferem os produtos entre as alas de torrefação, café torrado e café moído. Diferente da automação de equipamentos, esses *softwares* necessitam de constantes alterações, conforme a necessidade individual dos clientes. Quanto menor o tempo gasto em tarefas repetitivas, maior o tempo disponível para melhorias e menores os custos homem-hora.

3.2 HARDWARE E *SOFTWARE*

Os estudos e testes dos FBD serão implementados em um controlador Allen-Bradley Compactlogix 1769-L33ER, desenvolvido no ambiente Studio 5000, que atende aos requisitos da norma IEC 61131-3 (ALLEN-BRADLEY, 2016). Para as

telas, será utilizada a IHM PanelView Plus 1000, também da americana Allen-Bradley, em ambiente de desenvolvimento FactoryTalk View Studio. A Figura 5 apresenta o controlador e a IHM. A justificativa para a escolha do *hardware* é que este foi o produto ofertado na proposta de venda do sistema e é amplamente utilizado pela Probat Leogap.

Figura 5: controlador CompactLogix 1769-L33ER e IHM PanelView Plus 1000
Fonte: Allen-Bradley (2016).

3.3 SISTEMA DE ESTUDO

Os novos blocos serão aplicados em um sistema que envolve o transporte, armazenagem, moagem e empacotamento de café. A Figura 6 apresenta um fluxograma simplificado do processo e a Figura 7 ilustra o sistema montado.

O fluxograma mostra que há dois caminhos possíveis para o café atingir a empacotadeira: pelo torrador Turbo 4000 ou pelo torrador Probatone 60. Se o operador deseja empacotar o café torrado pelo Turbo, o produto será classificado por peneiras e armazenado, onde aguardará a instrução de envio para a empacotadeira. Quando é solicitado produto do Probatone, não é necessária a classificação pelas peneiras de grãos, uma vez que a matéria-prima é rigorosamente selecionada (tipo especial). Na sequência, existem duas opções: empacotar o café em grãos ou empacotar o café moído. Caso seja escolhida a opção de empacotar o café moído, uma peneira seleciona o produto conforme as especificações, antes de abastecer a empacotadeira.

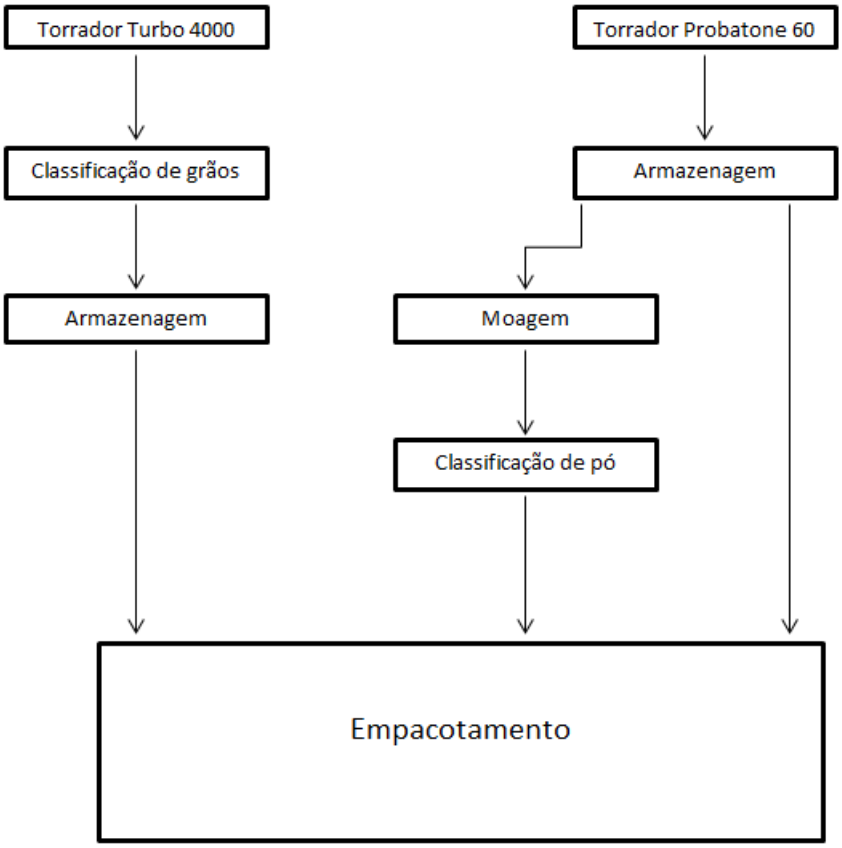

Figura 6: fluxograma do processo
Fonte: autoria própria

O sistema da Figura 7 é uma representação que contém os atuadores, representados com círculos preenchidos em verde, e elementos sensores. Os círculos de cor vermelha representam sensores de segurança (por exemplo, detecção de porta aberta), enquanto os preenchidos na cor amarela indicam sensores que fornecem informações para o sistema (níveis máximo e mínimo, pressostatos, etc).

Uma parte do processo é realizada pelo torrador Turbo 4000, que envia seu produto para uma peneira classificadora de grãos (trecho 1). Os grãos aceitos são transportados para silos de armazenagem com sistema de controle de injeção e alívio de gases. A extração dos silos pode ser realizada através de solicitação da empacotadeira ou manualmente (trecho 2).

A outra parte do processo, destacada com o número 3, ocorre com o envio do produto do torrador Probatone 60 para um silo de armazenagem (também com controle de gases). A extração pode ser realizada diretamente pela empacotadeira

ou por intermédio de um moinho de rolos MRM 500, que envia para uma peneira de pó e, por fim, para a empacotadeira.

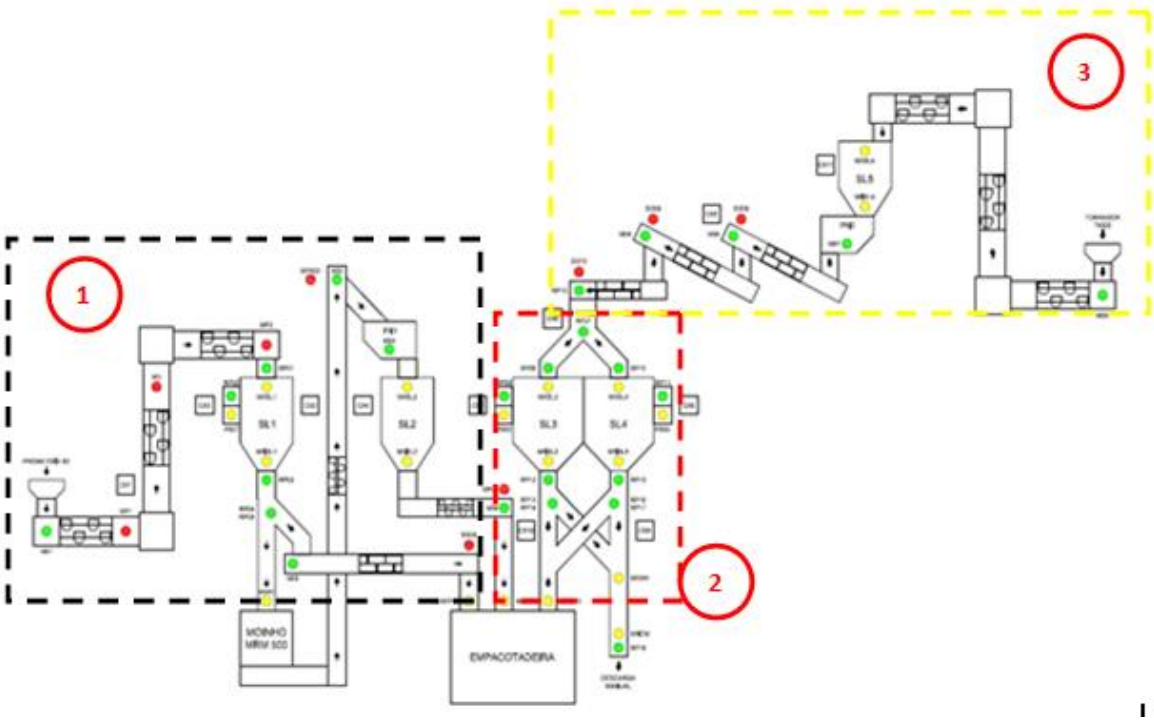

Figura 7: ilustração do sistema
Fonte: autoria própria

3.4 BLOCOS DE FUNÇÕES ANALISADOS

Desde as primeiras implementações pela Probat Leogap de blocos de funções personalizados em CLPs, não foram realizadas melhorias significativas ou qualquer estudo das estruturas.

Devido à elevada quantidade de FBDs utilizados pela companhia, foram selecionados para a apresentação deste trabalho apenas os blocos de controle de motores e pistões pneumáticos (para este trabalho, será usado o termo registro pneumático, ou apenas RP).

É importante ressaltar que a maioria dos dispositivos (motores, RPs, solenoides, etc.) que são inseridos em um programa Probat Leogap são criados com um tipo próprio de dado (desenvolvido pela equipe de engenharia de automação). Tais dados possuem todos os recursos necessários para uma programação e operação eficientes, com parte das informações disponíveis nas IHMs, nas telas de

controle dos dispositivos. Um motor, por exemplo, deve ser criado utilizando-se o tipo de dado MOTOR[x], onde *x* é o índice do dispositivo, para sequenciamento (MOTOR[1], MOTOR[2], etc.). Dessa forma, tem-se uma matriz de motores. As características de cada tipo de dado serão descritas nos subtópicos seguintes.

A ideia de matriz é aplicada também em alarmes. Porém, como são utilizados *triggers* digitais (até mesmo em variáveis de processo que são analógicas), não é necessária a criação de um bloco puramente para alarmes. O que é feito é a criação de uma matriz de booleanos, em que cada elemento corresponde a um *bit* de alarme.

É relevante para o entendimento deste trabalho conhecer a forma como são criados os blocos e os parâmetros requeridos. Os detalhes internos, as formas de utilização e funcionamento de cada bloco não são relevantes para este trabalho e não serão abordados.

Os blocos estudados neste capitulo poderão ser consultados no Apêndice A.

3.4.1 Blocos para motores

Para a inserção de motores no programa, é seguido um fluxograma semelhante ao apresentado na Figura 8. O primeiro passo é a criação, no programa, do bloco principal do motor (seja acionamento simples ou reversível). Na sequência, são criados os blocos de proteções, que incluem na aplicação os dispositivos como disjuntores, fins de curso, sensores de giro, etc. Finalmente, são criadas *tags* auxiliares, que determinam condições específicas de cada dispositivo. Por exemplo, uma situação de bloqueio de acionamento.

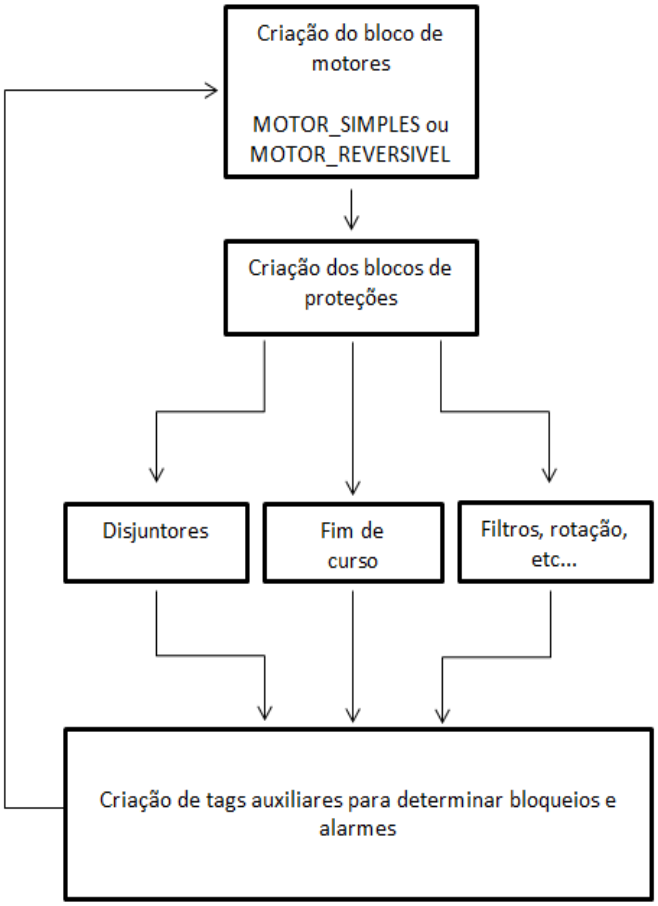

Figura 8: fluxograma para criação de motores nos programas em estudo

A Figura 9 apresenta o bloco principal de um motor sem inversão de sentido de rotação (chamado MOTOR_SIMPLES).

Caso o cliente decida, com a linha já em operação, alterar o tipo de acionamento do motor de simples para reversivel, o bloco principal também deve ser substituído de MOTOR_SIMPLES para MOTOR_REVERSIVEL. Este é um ponto importante, pois, apesar de mudanças sutis, será necessária a reprogramação do bloco principal. A Figura 10 mostra o bloco MOTOR_REVERSIVEL, para acionamento de um motor com inversão do sentido de rotação

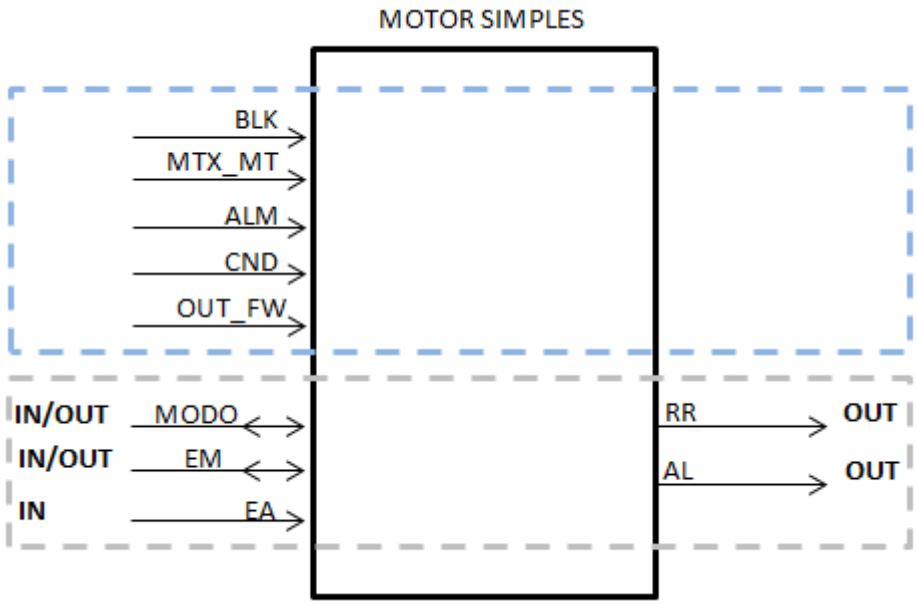

Figura 9: representação simplificada do bloco de um motor simples
Fonte: autoria própria

Onde,

BLK : *Tag* que referencia o bloco;

MTX_MT : Posição na matriz de motores;

ALM : *Bit* de alarme;

CND : *Bit* de condições de partida;

OUT_FW : *Tag* que contém o endereço da saída;

MODO : Modo de operação, se em manual ou automático;

EM : Acionamento do motor em manual;

EA : Acionamento do motor em automático;

RR : Confirmação de motor rodando;

AL : Motor em alarme;

IN/OUT : Parâmetro de entrada e saída (CLP lê e escreve no bloco);

IN : Parâmetro de entrada (CLP escreve no bloco);

OUT : Parâmetro de saída (CLP lê do bloco);

– – – – : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa;

– – – – : Parâmetros pertencentes ao tipo de dado MOTOR, com acesso via IHM.

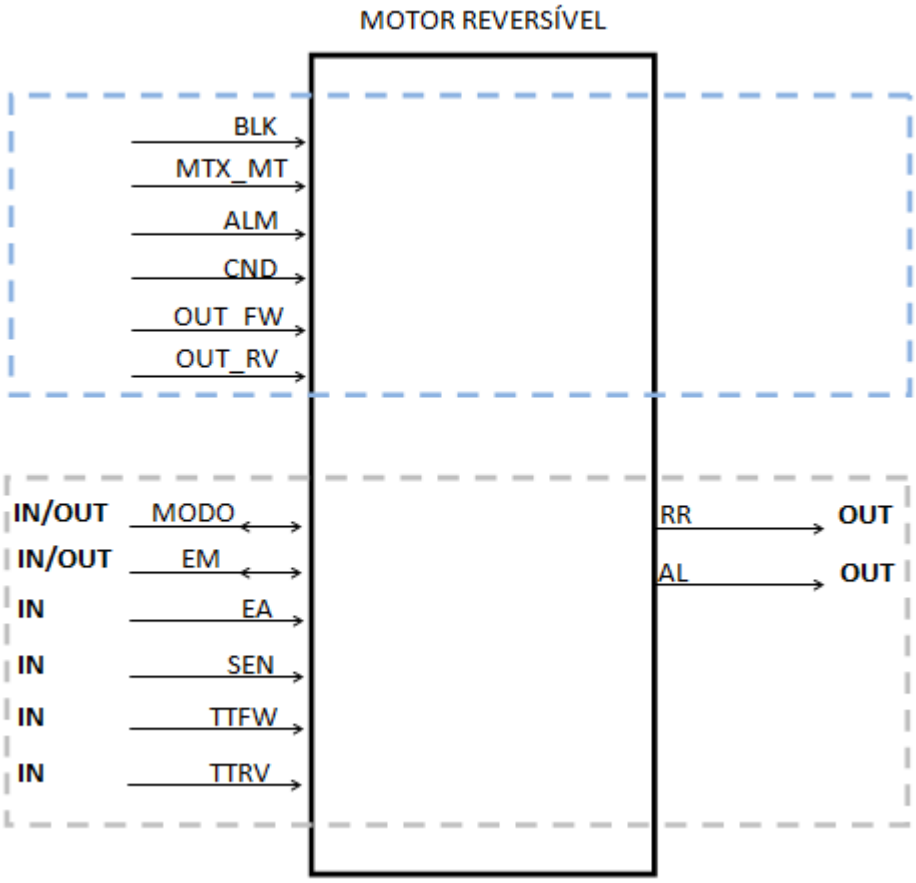

Figura 10: representação simplificada do bloco de um motor reversível
Fonte: autoria própria

Onde,

BLK : *Tag* que referencia o bloco;

MTX_MT : Posição na matriz de motores;

ALM : *Bit* de alarme;

CND : *Bit* de condições de partida;

OUT_FW : *Tag* que contém o endereço da saída;

OUT_RV : *Tag* que contém o endereço de inversão de fases da saída;

MODO : Modo de operação, se em manual ou automático;

EM : Acionamento do motor em manual;

EA : Acionamento do motor em automático;

SEN : Sentido de rotação do motor

TTFW : Tempo para acionar a saída direta do motor;

TTRV : Tempo para acionar a saída reversa do motor;

RR : Confirmação de motor rodando;

AL : Motor em alarme;

IN/OUT : Parâmetro de entrada e saída (CLP lê e escreve no bloco);

IN : Parâmetro de entrada (CLP escreve no bloco);

OUT : Parâmetro de saída (CLP lê do bloco);

– – – – : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa;

– – – – : Parâmetros pertencentes ao tipo de dado MOTOR, com acesso via IHM.

Os *bits* ALM e CND são ativados por bobinas simples, através de linhas em rotinas do programa. Por exemplo, o contato NA de um alarme de disjuntor pode ser inserido em série com a bobina de ALM, sinalizando que existe um alarme ativo para o motor. Outra situação possível é que um motor só pode ser iniciado quando a

máquina estiver ligada. Sendo assim, o *bit* NA que informa que a máquina está ligada pode ser inserido em série com a bobina de CND.

Todo motor é protegido por um disjuntor, mesmo os com partida eletrônica. Disjuntores industriais podem disponibilizar contatos auxiliares para informar aos controladores se estes estão ou não armados. Esta informação é processada pelo bloco de alarmes de disjuntores, resumido na Figura 11.

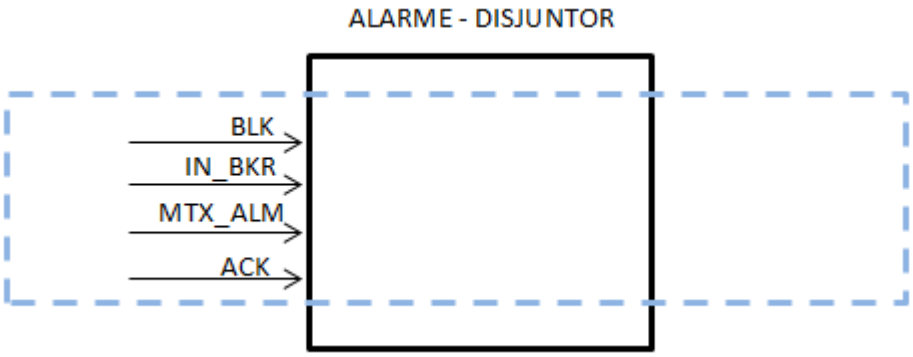

Figura 11: representação simplificada do bloco de alarme de disjuntor
Fonte: autoria própria

Onde,

BLK : *Tag* que referencia o bloco;

IN_BKR : Endereço da entrada do CLP para o sinal de disjuntor;

MTX_ALM : Posição do alarme de disjuntor na matriz de alarmes;

ACK : *Bit* de reconhecimento de alarme (*acknowledgment*);

- - - - : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa.

Outro bloco de proteção é o filtro de sensores. Tem como função filtrar os sinais de sensores que protegem o acionamento do motor (sensor fim de curso, por exemplo), através da produção de saídas com atraso. Sua forma simplificada é ilustrada na Figura 12.

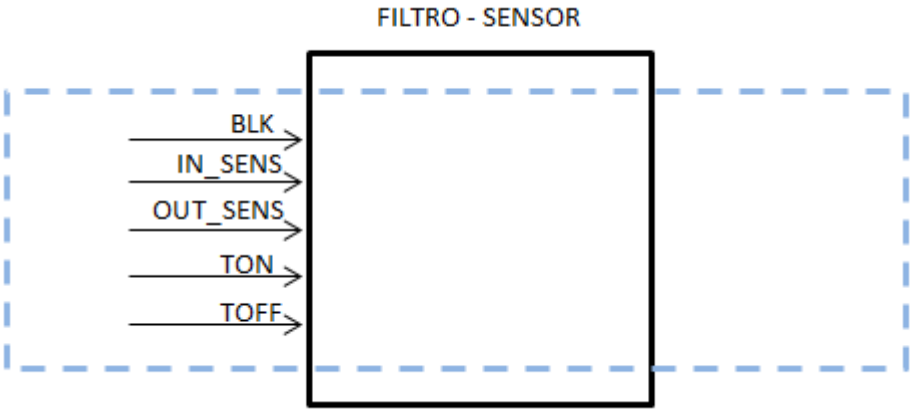

Figura 12: representação simplificada do bloco de filtragem de sinais de sensores
Fonte: autoria própria

Onde,

BLK : *Tag* que referencia o bloco;

IN_SENS : Endereço da entrada do CLP para o sinal do sensor;

OUT_SENS : *Bit* com saída atrasada com relação à IN_SENS;

TON : Tempo de atraso para borda de subida do sinal do sensor;

TOFF : Tempo de atraso para borda de descida do sinal do sensor;

- - - - : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa.

O último bloco de proteção de acionamento que será apresentado neste trabalho será para alarmes de sensores (fim de curso, por exemplo), conforme ilustrado na Figura 13.

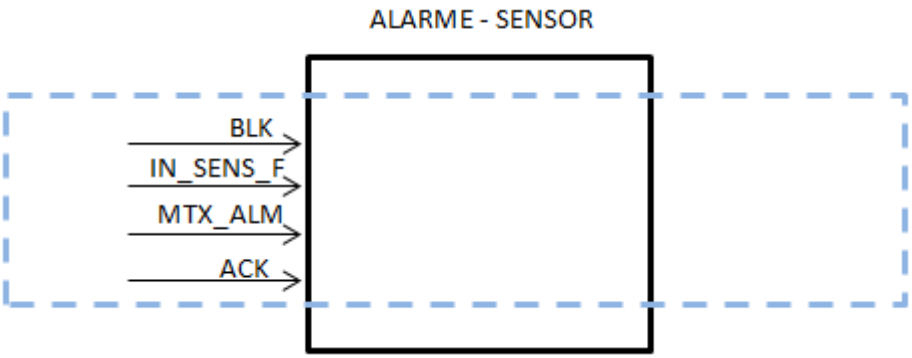

Figura 13: representação simplificada do bloco de alarmes de sensores
Fonte: autoria própria

Onde,

BLK : *Tag* que referencia o bloco;

IN_SENS_F : Bit com atraso do sinal de entrada do sensor (produzido pela saída OUT_SENS, do bloco de filtros);

MTX_ALM : Posição do alarme na matriz de alarmes;

ACK : Bit de reconhecimento de alarme (*acknowledgment*);

\- - - - : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa.

Alguns sensores e atuadores não possuem FBD, de forma que a lógica de funcionamento deve ser construída manualmente. Um exemplo é a lógica de verificação de rotação de um motor, que depende dos pulsos de um sensor indutivo (chamado, na aplicação, de sensor de giro). A rotina possui 6 linhas de código e 8 *tags*.

Uma vez definidos os *bits* de condições de partida/funcionamento e alarmes, o bloco de motor está configurado no controlador.

No total, para configurar um motor, são utilizadas aproximadamente 14 *tags* e 6 linhas em rotina de programa; caso o motor tenha sensor de giro, o total de *tags* sobe para 20 e de linhas, para 14. Para motores com inversão de sentido de rotação, acrescenta-se ao total: 1 *tag* para a saída a reversa, 1 *tag* indicando o sentido desejado de rotação, 1 *tag* com o tempo de reversão para o sentido direto e, por fim, 1 *tag* com o tempo de reversão para o sentido reverso.

A tela da IHM para controle de motores é apresentada na Figura 14.

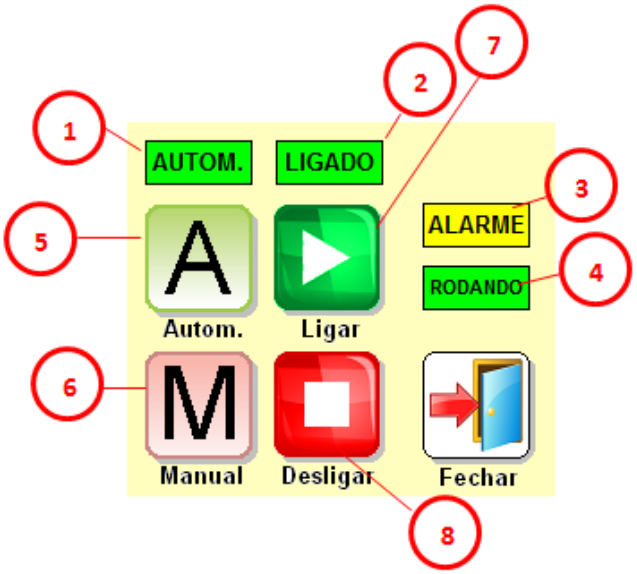

Figura 14: tela de controle de motores do sistema em estudo
Fonte: IHM Probat Leogap

Onde,

1 : Status do modo de operação: automático ou manual;

2 : Status do comando do motor: ligado ou desligado;

3 : Status do motor: ok ou em alarme;

4 : Status do motor: parado ou rodando;

5 : Comando: motor em automático;

6 : Comando: motor em manual;

7 : Comando: ligar motor em manual;

8 : Comando: desligar o motor em manual.

3.4.2 Blocos para registros pneumáticos

Para inclusão de registros pneumáticos, o procedimento requer menores esforços, pois são utilizados apenas 2 blocos. A Figura 15 é uma representação simplificada do bloco principal. Tal como nos blocos de motores, os detalhes internos dos blocos não serão abordados, apenas a forma de inclusão do dispositivo e os parâmetros necessários.

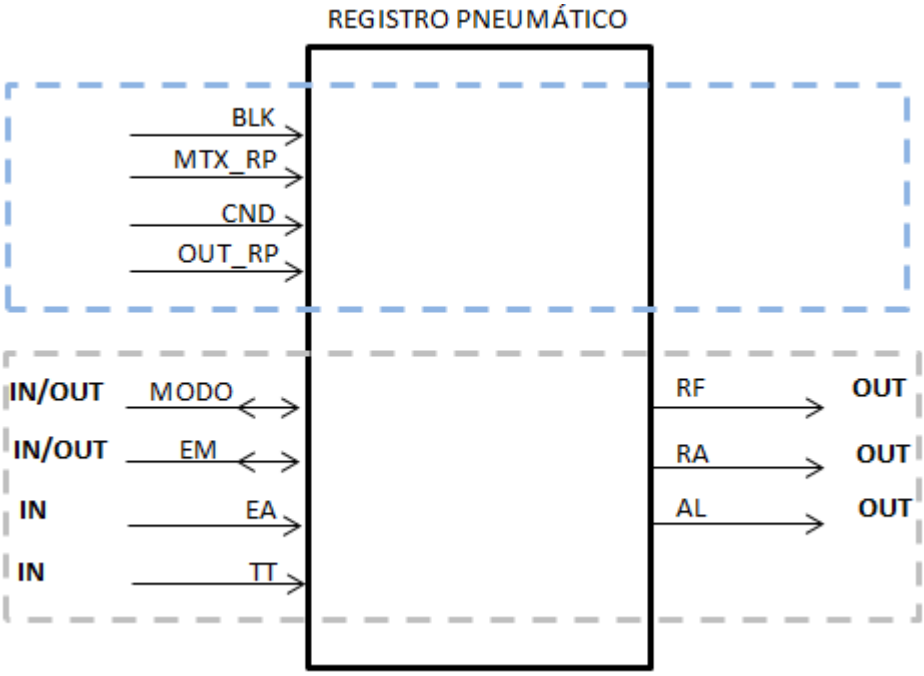

Figura 15: representação simplificada do bloco de registro pneumático
Fonte: autoria própria

Onde,

BLK : *Tag* que referencia o bloco;

MTX_RP : Posição na matriz de registros pneumáticos;

CND : *Bit* de condições de acionamento;

OUT_RP : Tag que contém o endereço da saída;

MODO : Modo de operação, se em manual ou automático;

EM : Acionamento do motor em manual;

EA : Acionamento do motor em automático;

TT : Tempo de transição do êmbolo magnético interno do pistão (entre as posições aberto e fechado), para evitar a ocorrência de alarmes falsos;

RF : Confirmação de registro pneumático na posição fechada;

RA : Confirmação de registro pneumático na posição aberta;

AL : Registro pneumático em alarme;

IN/OUT : Parâmetro de entrada e saída (CLP lê e escreve no bloco);

IN : Parâmetro de entrada (CLP escreve no bloco);

OUT : Parâmetro de saída (CLP lê do bloco);

– – – – : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa;

– – – – : Parâmetros pertencentes ao tipo de dado REGISTRO_PNEUMATICO, com acesso via IHM.

O *bit* CND, assim como no bloco de MOTOR, é de bobina simples, configurado em linha de rotina do programa. Uma condição possível para um registro pneumático, é que este seja acionado somente quando um determinado motor estiver em funcionamento. Assim, o *bit* NA que indica o funcionamento do motor deve ser inserido em série com a bobina de CND.

Para configuração de alarmes de registros pneumáticos, é utilizado um bloco específico, representado na Figura 16.

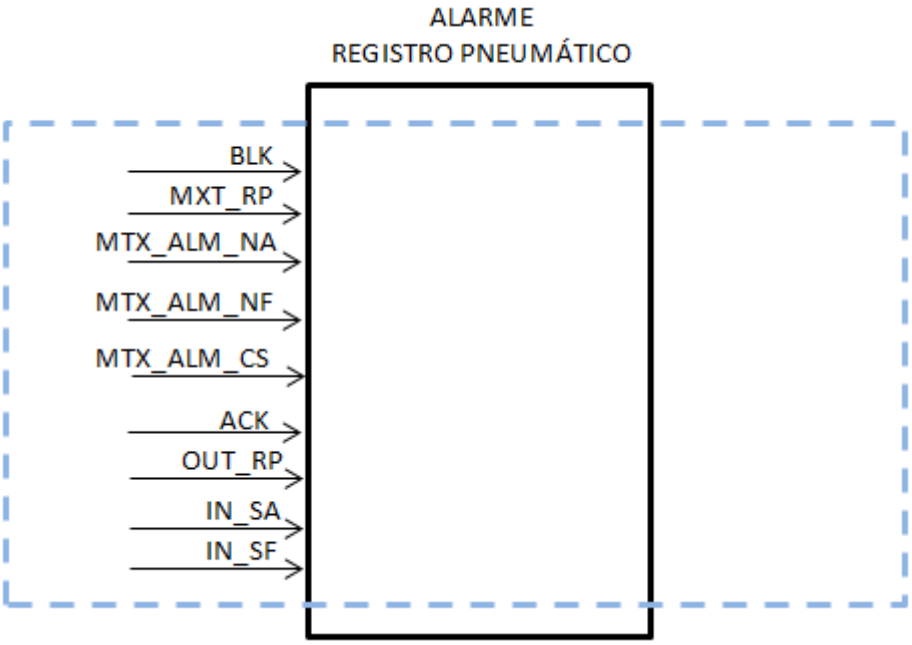

Figura 16: representação simplificada do bloco de alarmes de registros pneumáticos
Fonte: autoria própria

Onde,

BLK : *Tag* que referencia o bloco;

MTX_RP : Posição na matriz de registros pneumáticos;

MTX_ALM_NA : Posição do alarme de falha de abertura, na matriz de alarmes;

MTX_ALM_NF : Posição do alarme de falha de fechamento, na matriz de alarmes;

MTX_ALM_CS : Posição do alarme de conflito de sensores, na matriz de alarmes;

ACK : *Bit* de reconhecimento de alarme (*acknowledgment*);

OUT_RP : *Tag* que contém o endereço da saída;

IN_SA : Endereço de entrada do CLP que recebe o sinal do sensor de registro pneumático aberto;

IN_SF : Endereço de entrada do CLP que recebe o sinal do sensor de registro pneumático fechado;

– – – – : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa;

Concluídas as configurações de condições de acionamento e alarmes, o registro pneumático está pronto para uso. A tela de IHM para controle individual de registros pneumáticos é apresentada na Figura 17

Para a inserção de um dispositivo do tipo registro pneumático foram utilizadas 9 *tags* e 4 linhas de rotina de programa.

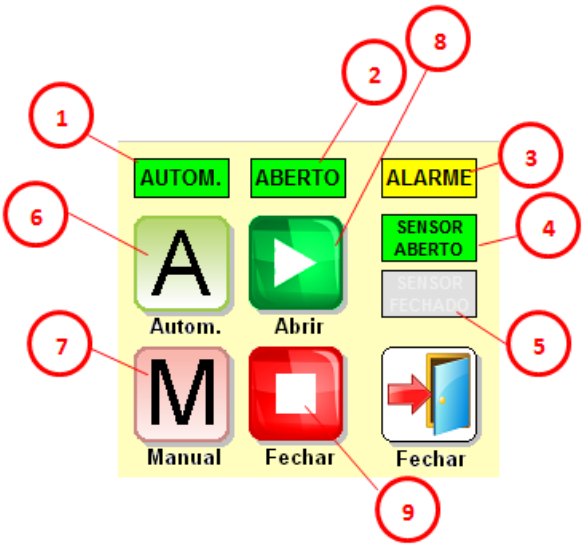

Figura 17: tela de controle de registros pneumáticos do sistema em estudo
Fonte: IHM Probat Leogap

Onde,

1 : Status do modo de operação: automático ou manual;

2 : Status do registro pneumático: aberto ou fechado;

3 : Status do registro pneumático: ok ou em alarme;

4 : Status do registro pneumático aberto;

5 : Status do registro pneumático fechado;

6 : Comando: registro pneumático em automático;

7 : Comando: registro pneumático em manual;

8 : Comando: acionar registro pneumático em manual;

9 : Comando: desligar o acionamento do registro pneumático em manual.

3.5 BLOCOS DE FUNÇÕES PROPOSTOS

Após a análise dos blocos e das telas de IHM apresentadas no item 3.4 deste trabalho, algumas propostas de otimização são apresentadas:

- Eliminação do bloco dedicado ao alarme de disjuntores, incorporando-o ao bloco principal de motores (disjuntores estão sempre presentes em motores);
- Eliminação do bloco dedicado ao alarme de sensores de proteção de acionamentos de motores, incorporando-o ao bloco principal de motores (caso o projeto do acionamento do motor não contemple sensores de proteção, bastará desativá-lo no bloco);
- Incorporação de rotina de filtro para atraso de sinal de sensores ao bloco principal de motores;
- Incorporação de lógica para alarme de sensor de rotação ao bloco principal de motores;
- Eliminação do bloco de alarmes de registros pneumáticos, incorporando-o ao bloco principal de registros pneumáticos;
- Criação de rotina de *by-pass* de proteções, para fins de testes e manutenções. Acessível via IHM;
- Diagnóstico de alarmes na tela da IHM;
- Configuração de temporizadores dos dispositivos na tela da IHM;
- Criação de rotina de testes para os RPs. Acessível via IHM.

4 RESULTADOS

Este capítulo mostra os resultados obtidos com a otimização dos blocos de funções existentes. As seções 4.1 e 4.2 apresentam os resultados obtidos na criação de motores e registros pneumáticos, respectivamente.

Os blocos propostos neste capítulo poderão ser consultados no Apêndice B.

4.1 BLOCOS PARA MOTORES

Anteriormente, a configuração de motores deveria seguir o fluxograma apresentado na Figura 8. Nesta nova proposta, todas as funcionalidades da antiga configuração foram agrupadas em apenas um bloco, o qual está representado na Figura 18.

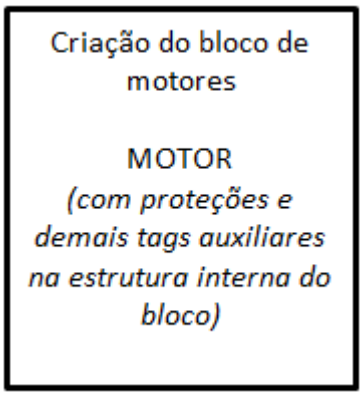

Figura 18: bloco para criação de motores nos novos programas
Fonte: autoria própria

A Figura 19 mostra o novo bloco para motores.

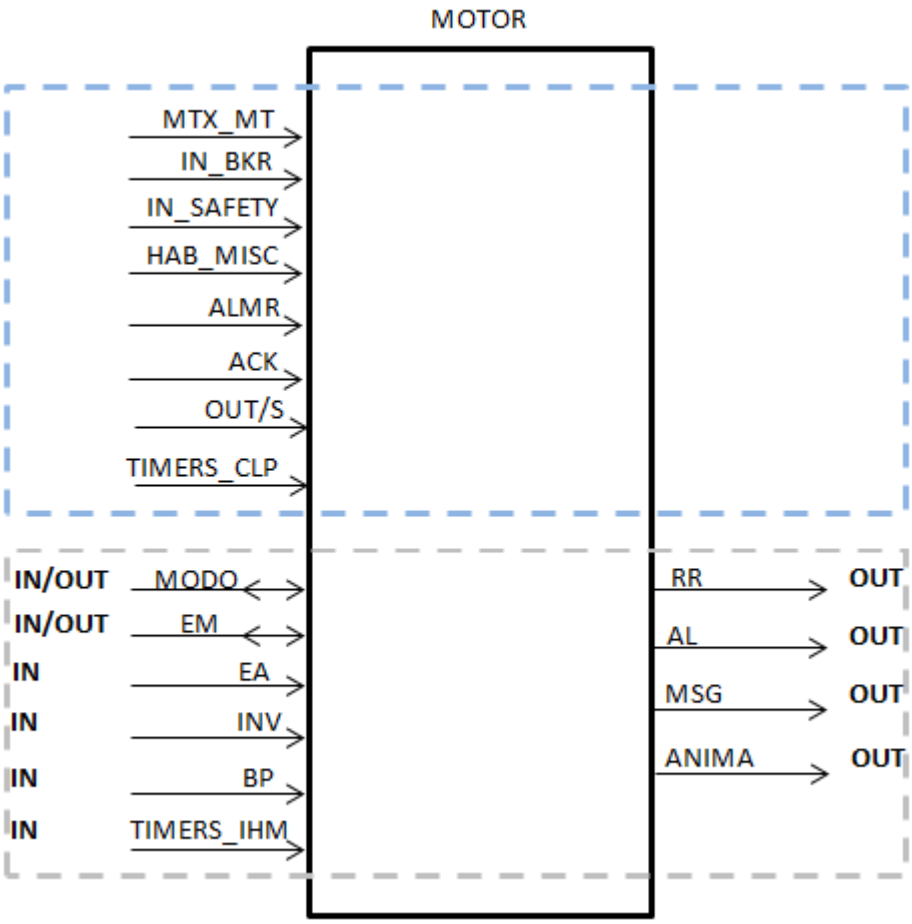

Figura 19: nova representação simplificada do bloco de motores
Fonte: autoria própria

Onde,

MTX_MT : Posição na matriz de motores;

IN_BKR : Endereço da entrada do CLP para o sinal de disjuntor;

IN_SAFETY : Endereços de entrada dos dispositivos de segurança;

HAB_MISC : *Bits* para habilitar alarmes, telas de IHM, reversão, etc.

ALMR : Posições dos alarmes do bloco (segurança, acionamento, giro, etc.), na matriz de alarmes;

ACK : *Bit* de reconhecimento de alarme (*acknowledgment*);

OUT/S : Endereços de saída do CLP para acionamento direto e reverso (caso necessário);

TIMERS_CLP : Configuração de temporizadores fixos (ex: tempo de pausa para inversão do sentido de rotação de um motor);

MODO : Modo de operação, se em manual ou automático;

EM : Acionamento do motor em manual;

EA : Acionamento do motor em automático;

INV : *Bit* para determinar o sentido de rotação do motor;

BP : *Bit* para ativar o by-pass das proteções do motor;

TIMERS_IHM : Configuração de temporizadores variáveis (ex: tempo de partida, parada, filtros de sensores, etc.);

RR : Confirmação de motor rodando;

AL : Motor em alarme;

MSG : Mensagens de status do motor (parado, partindo, etc.) e mensagens de alarme (disjuntor desarmado, bloqueio de segurança, etc.);

ANIMA : *Bits* diversos de animações da IHM (cores, formas, etc.);

IN/OUT : Parâmetro de entrada e saída (CLP lê e escreve no bloco);

IN : Parâmetro de entrada (CLP escreve no bloco);

OUT : Parâmetro de saída (CLP lê do bloco);

– – – – : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa;

– – – – : Parâmetros pertencentes ao tipo de dado MOTOR, com acesso via IHM.

Anteriormente, foi mostrada a necessidade de criação de 4 blocos para gerenciar os acionamentos de um motor. A primeira etapa (antes mesmo de inserir qualquer bloco no programa) era definir o tipo de acionamento, se normal ou reversível. Na nova configuração, o mesmo bloco serve para qualquer motor, com ou sem inversão de rotação. Para o procedimento de reversão, o único requisito necessário é a ativação do *bit* do parâmetro de reversão e declaração do endereço de saída que corresponde ao outro sentido de giro, mantendo idêntica toda a estrutura anterior.

Para os dispositivos de segurança, assim como no procedimento de motores reversíveis, foram criados *bits* de ativação no bloco. Além dos disjuntores, os seguintes sensores de proteção podem ser ativados no bloco: fins de curso (até 2 por motor) e sensor de giro.

Uma grande novidade foi a implementação de parâmetros básicos acessíveis pela IHM. Ou seja, dispensam o uso de um computador e a presença do programador. São acessíveis: configuração de tempo de partida/parada e *by-pass* das proteções, para testes do motor. Um técnico eletricista, com permissões de *login*, pode configurar estes parâmetros na IHM com um conhecimento básico de CLPs.

Com relação ao número de blocos, a configuração de motores simples estudada exigia a inclusão de, pelo menos, 2 blocos (principal e alarme de disjuntores). Caso fosse necessária a inclusão de dispositivos de proteção, (um sensor fim de curso, por exemplo), seriam necessários outros 2 blocos adicionais (filtro e alarme do sensor). A nova configuração utiliza apenas 1 bloco.

Analisando o número de *tags* para uma configuração simples de motor, há uma redução de 14 para 5 *tag*s. Ao adicionar um sensor de giro ao motor, o número sobe para 7 *tags* (dispensando as linhas adicionais citadas anteriormente). Incluindo proteção por fim de curso, são totalizadas 9 *tags*.

Quanto ao número de linhas utilizadas, o novo bloco de motores é configurado em apenas 1, enquanto a configuração estudada no capítulo 3 utilizava de 6 a 14 linhas, dependendo da configuração.

A nova tela de IHM para controle dos motores é apresentada na Figura 20.

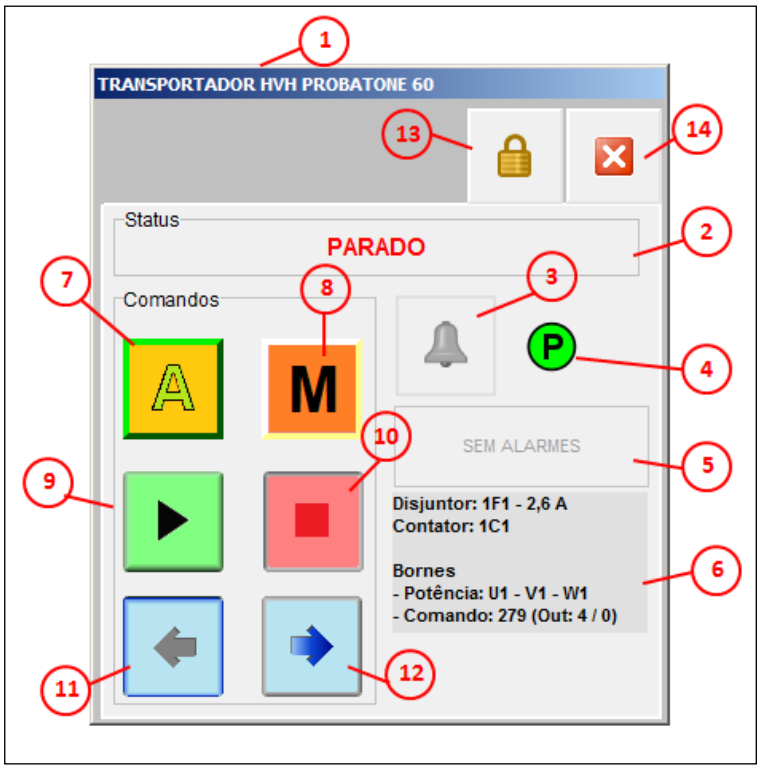

Figura 20: nova tela para controle de motores
Fonte: autoria própria

Onde,

1 : Nome do motor;

2 : Status do motor;

3 : Sinalizador de alarme;

4 : Status de permissões;

5 : Indicador de alarme ativo;

6 : Lista de conexões;

7 : Comando: motor em automático;

8 : Comando: motor em manual;

9 : Comando: ligar motor em manual;

10 : Comando: desligar motor em manual;

11 : Comando: acionamento reverso do motor em manual (disponível para motores reversíveis);

12 : Comando: acionamento direto do motor em manual (disponível para motores reversíveis);

13 : *Login* (informações de *login* fornecidas apenas para pessoal autorizado);

14 : Fecha a tela

Uma característica importante da nova tela é que, para os usuários que possuem privilégios de acesso, é possível configurar os parâmetros de tempo de parada, partida e reversão do motor, bem como ativar o *by-pass* das proteções (Figura 21).

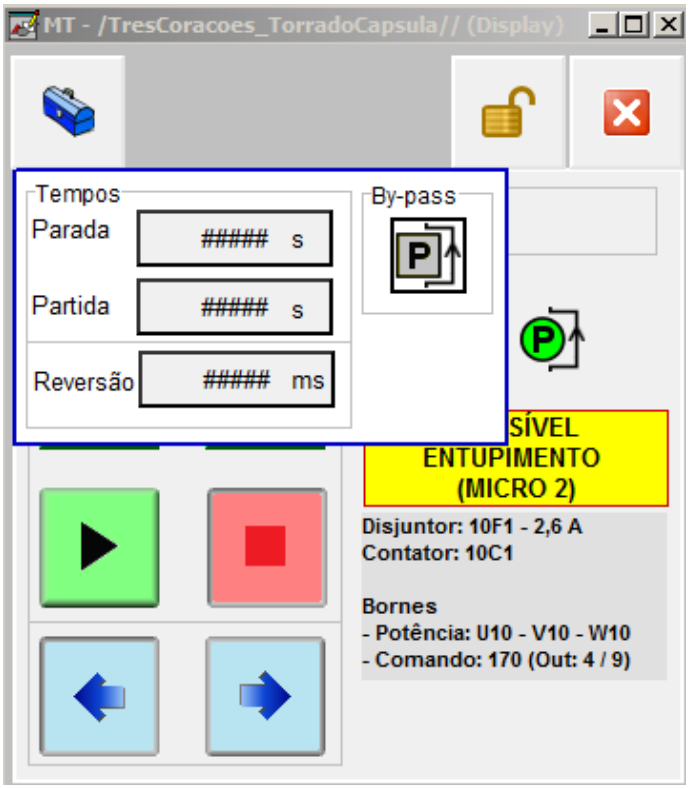

Figura 21: tela auxiliar para configurações extras do novo bloco de motor
Fonte: autoria própria

4.2 BLOCOS PARA REGISTROS PNEUMÁTICOS

Assim como ocorreu no item 4.1, um único bloco deverá ser inserido. A Figura 22 representa a nova configuração.

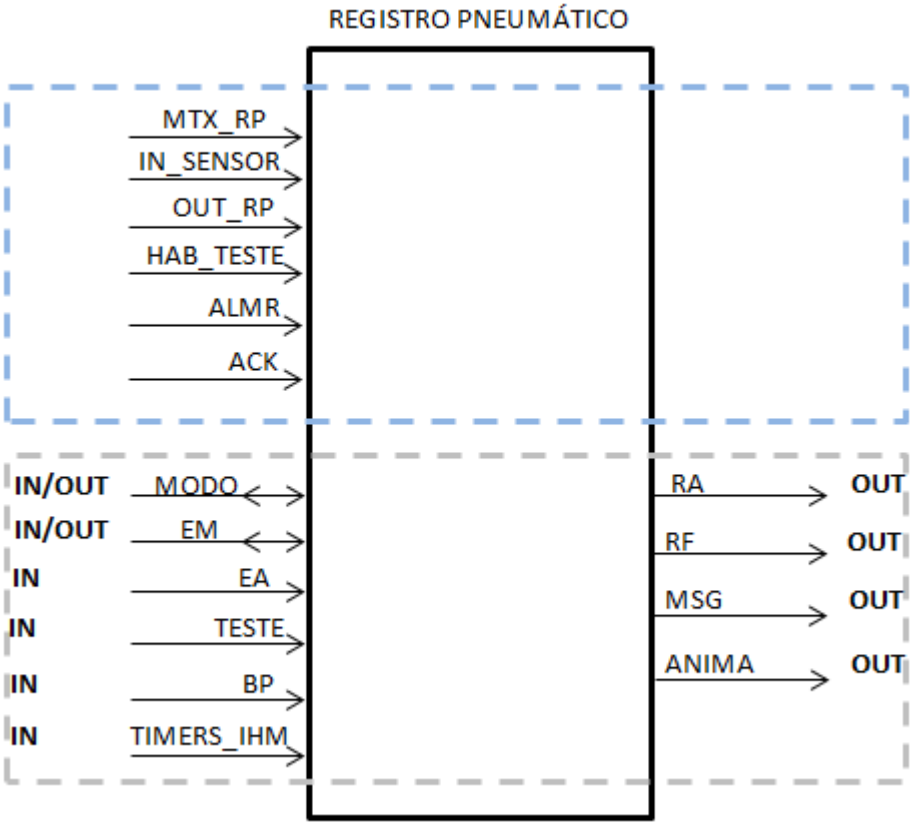

Figura 22: representação simplificada do bloco principal de registros pneumáticos
Fonte: autoria própria

Onde,

MTX_RP : Posição na matriz de registros pneumáticos;

IN_SENSOR : Endereços das entradas do CLP para os sinais de posição do pistão;

OUT_RP : Endereço da saída do CLP para acionamento do pistão;

HAB_TESTE : *Bit* para habilitar rotina de testes no RP.

ALMR : Posições dos alarmes do bloco (segurança, acionamento, giro,

etc.), na matriz de alarmes;

ACK : *Bit* de reconhecimento de alarme (*acknowledgment*);

MODO : Modo de operação, se em manual ou automático;

EM : Acionamento do RP em manual;

EA : Acionamento do RP em automático;

TESTE : *Bit* para iniciar o teste de abertura e fechamento do RP;

TIMERS_IHM : Configuração de temporizadores variáveis (ex: tempo de transição);

RF: Confirmação de registro pneumático fechado;

RA : Confirmação de registro pneumático aberto;

AL : RP em alarme;

MSG : Mensagens de status do RP e mensagens de alarme;

ANIMA : *Bit*s diversos de animações da IHM (cores, formas, etc.);

IN/OUT : Parâmetro de entrada e saída (CLP lê e escreve no bloco);

IN : Parâmetro de entrada (CLP escreve no bloco);

OUT : Parâmetro de saída (CLP lê do bloco);

– – – – : Parâmetros de entrada do bloco, declarados pelo programador no desenvolvimento do programa;

– – – – : Parâmetros pertencentes ao tipo de dado REGISTRO, com acesso via IHM.

Nesta nova configuração, os alarmes e acionamentos estão no mesmo bloco, diferente da configuração estudada, que exigia dois blocos: o principal e o de alarmes. Além dessa novidade, o bloco possui um sinal que habilita a rotina de testes do registro, acessível via IHM (desde que o usuário possua o privilégio de

acesso). A configuração anterior exigia que pelo menos dois técnicos estivessem disponíveis para testes de registros: um para ligar e desligar as bobinas na IHM e outro para ajustar os acionamentos e sensores localmente. Com a rotina de testes, o técnico eletricista habilita a temporização dos comandos liga/desliga do registro, de forma a poder realizar ajustes de sensores e acionamentos sem a necessidade de outro técnico.

A implementação do novo bloco propicia a redução de 9 para 4 *tags*. Com relação às linhas de rotinas utilizadas, a nova configuração utiliza apenas 1 linha, frente às 4 linhas utilizadas na configuração anterior. O número de blocos utilizados também é reduzido: antes eram utilizados 2 blocos, no bloco proposto é necessário apenas 1.

A nova tela de controle de registros pneumáticos é apresentada na Figura 23.

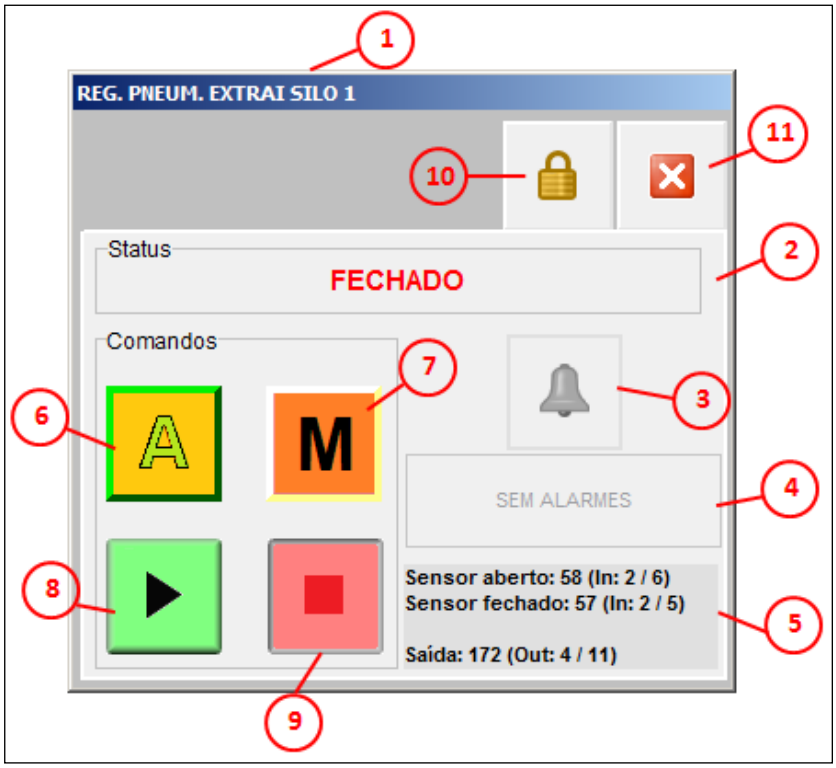

Figura 23: nova tela de controle de registros pneumáticos
Fonte: autoria própria

Onde,

1 : Nome do registro pneumático;

2 : Status do registro pneumático;

3 : Sinalizador de alarme;

4 : Indicador de alarme ativo;

5 : Lista de conexões;

6 : Comando: registro pneumático em automático;

7 : Comando: registro pneumático em manual;

8 : Comando: liga registro pneumático em manual;

9 : Comando: desliga registro pneumático em manual;

10 : *Login* (informações de *login* fornecidas apenas para pessoal autorizado);

11 : Fecha a tela.

Assim como na tela de motores, os usuários que possuem privilégios de acesso podem configurar os parâmetros de tempo de transição entre as posições aberto e fechado do registro pneumático. Também é possível habilitar a rotina de testes e configurar os tempos de registro aberto e fechado, como pode ser observado na Figura 24.

Figura 24: tela auxiliar para configurações extras do registro pneumático
Fonte: autoria própria

Diante da redução do tempo de desenvolvimento dos sistemas, devido à redução do número de *tags* e do número de blocos, foram criados outros 3 FBD para o sistema apresentado (totalizando 5 novos blocos).

Utilizando como referência os 2 blocos apresentados no capítulo 4, itens 4.1.1 e 4.1.2, foi desenvolvido um questionário para programadores (desenvolvedores do *software*) e técnicos de manutenção (usuários do *software*). O resultado pode ser consultado no Apêndice C.

5 CONSIDERAÇÕES FINAIS

O tempo de desenvolvimento e manutenção de um *software* causa impacto direto nos custos de automação de um projeto. Sendo assim, a revisão das ferramentas existentes é imprescindível. Neste trabalho, buscou-se a otimização dos blocos de funções utilizados pela empresa Probat Leogap, os quais ficaram por quase 10 anos sem estudos de melhorias. Os blocos de funções são conceitos da norma IEC 61131-3, que visa a reutilização de trechos de programas (já testados e aprovados) para facilitar a programação e manutenção.

Foram analisadas as etapas para a criação de dois dispositivos no projeto: motores e registros pneumáticos.

Para o primeiro caso, foram eliminados os blocos desnecessários, agrupando, em apenas um, todas as funções essenciais. Foram agregadas funções, em conjunto com a IHM, que aceleram a manutenção e o desenvolvimento dos blocos, de modo a reduzir o tempo de parada de manutenção e o tempo de programação.

O segundo caso, para a criação de registros pneumáticos no projeto, seguiu-se a mesma linha de raciocínio utilizada para a criação de motores. Foram removidos os blocos desnecessários e utilizadas as principais funções de cada um para agrupá-las em um bloco, de modo a tornar o desenvolvimento do *software* mais ágil e facilitar a manutenção do sistema.

Os testes realizados em campo demonstraram que os objetivos propostos foram atingidos. As ferramentas desenvolvidas são uma “via de mão de única”, ou seja, sem abertura para a utilização dos FBD antigos, pois os novos blocos possuem as funcionalidades dos antigos, aliado à exigência de poucos conhecimentos de programação, como sugere a IEC 61131-3 (basta apenas declarar sinais de entrada e saída).

Relacionado ao pessoal de campo, a habilitação de rotinas de teste dispensa a utilização de ferramentas físicas que comprometam a segurança dos operadores e dos equipamentos. Isto permite que o responsável técnico pelo *startup* do sistema trabalhe com mais confiança e tranquilidade. Para o setor de engenharia de automação, as mudanças significam mais agilidade em desenvolvimento de

programas (redução de *tags*, linhas e blocos de funções), em testes, comissionamentos de painéis e em *startups*.

Para trabalhos futuros relacionados ao tema desta monografia, algumas sugestões são:

- Desenvolvimento de blocos que acionem motores controlados por conversores de frequência utilizando protocolo Ethernet, com todas as informações relevantes disponibilizadas na IHM;
- Atualização dos blocos de dispositivos de modo a gerar informações a nível gerencial, tais como: quantidade de acionamentos, tempo de parada por ocorrência de falha, totalizador de desarmes do disjuntor, etc.
- Atualização de IHMs conforme a ISA 101;

REFERÊNCIAS

ALLEN-BRADLEY. **Logix5000 Controllers IEC 61131-3 Compliance**. Istambul, Turquia: Rockwell Automation, 2016a. Disponível em: <http://literature.rockwellautomation.com/idc/groups/literature/documents/pm/1756-pm018_-en-p.pdf>. Acesso em: 14 set. 2016a.

ALLEN-BRADLEY. **Products.** 2016b. Disponível em: <http://www.ab.com/>. Acesso em 14 set. 2016.

BOTTURA FILHO, João A. **Benefícios da norma IEC 61131-3 aplicada a CLP's.** Revista Intech 2010. Rio de Janeiro: ISA, 2010.

BRAGA, Carmela M. P. B. **Norma IEC 1131-3: Padronização em Programação de Controle Industrial.** Minas Gerais: UFMG, 2005. Disponível em: <http://www.cpdee.ufmg.br/~carmela/NORMA%20IEC%201131.doc>. Acesso em: 13 set. 2016.

FAUSTINO, Marcos R. **Norma IEC 61131-3: aspectos históricos, técnicos e um exemplo de aplicação**. 2005. 123 f. Dissertação (Mestrado), Departamento de Engenharia de Energia e Automação Elétricas, Escola Politécnica da Universidade de São Paulo. São Paulo, 2005. Disponível em: < www.teses.usp.br/teses/disponiveis/3/3143/tde-02122005-215523/publico/tesefaustino.pdf>. Acesso em: 10 abr. 2016.

FENTON, Daniel. **PLC programming: 5 mistakes to avoid**. 2014. Disponível em: <https://www.controleng.com/single-article/plc-programming-5-mistakes-to-avoid/30b0e46fb63a7ffbb54284cf22f90d4d.html>. Acesso em: 12 abr. 2016.

HAMILTON, A. **The effect of IEC 1131-3 on the testing process for real time software applications**. Londres, Reino Unido: The Institution of Electrical Engineers (IEE), 1993. Disponível em: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=274594>. Acesso em: 10 abr. 2016.

IEC. **International Standard IEC 61131-3: Programming languages.** 2 ed. Geneva: IEC, 2003. Disponível em: <d1.amobbs.com/bbs_upload782111/files_31/ourdev_569653.pdf> Acesso em: 08 nov. 2016.

JOHN, Karl-Heinz; TIEGELKAMP, Michael. **IEC 61131-3: Programming Industrial Automation Systems**. Berlim, Alemanha: Springer, 2001.

MYERS, Leila; WILKINS, Maurice J. **ISA-101: toward a more effective HMI strategy.** Portal Automation.com, 2015. Disponível em: < http://www.automation.com/automation-news/article/isa-101-toward-a-more-effective-hmi-strategy>. Acesso em: 01 abr. 2016.

PLCOPEN. **IEC 61131-3: a norma para programação**. Zaltbommel, Países Baixos: 2004. Disponível em: <www.plcopen.org/pages/pc2_training/introductions_in_spanish_and_portugese/downloads/intro_iec_march04_portuguese.doc>. Acesso em: 07 abr. 2016.

PLCOPEN. **How to recognize an IEC 61131-3 Programming System in 8 easy steps. 2016.** Disponível em: <http://www.plcopen.org/pages/tc1_standards/how_to_recognize_iec_61131-3/>. Acesso em: 10 set. 2016.

PROBAT LEOGAP. **Indústria: torradores**. Curitiba, 2016a. Disponível em: <http://probatleogap.com.br/industria/equipamentos/categoria/torradores/>. Acesso em: 12 set. 2016.

PROBAT LEOGAP. **Indústria: moinhos**. Curitiba, 2016b. Disponível em: <http://probatleogap.com.br/industria/equipamentos/categoria/moinhos/>. Acesso em: 12 set. 2016.

PROBAT LEOGAP. **Indústria: plantas**. Curitiba, 2016c. Disponível em: <http://probatleogap.com.br/industria/plantas/>. Acesso em: 12 set. 2016.

SEIXAS FILHO, Constantino. **Aula IEC 61131-3.** Centro de Pesquisa e Desenvolvimento em Engenharia Elétrica – CPDEE. Universidade Federal de Minas Gerais (UFMG). Belo Horizonte, 1999. Disponível em: < http://www.cpdee.ufmg.br/~seixas/PaginaII/Download/DownloadFiles/Aula%20IEC%2061131-3.pdf>. Acesso em: 13 set. 2016.

VERWER, Andy. **The impact of IEC (6)1131-3 on the teaching of control engineering**. Londres, Reino Unido: The Institution of Electrical Engineers (IEEE) Colloquium, 1999. Disponível em: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=771825>. Acesso em: 10 abr. 2016.

APÊNDICE A – Blocos de motores e registros pneumáticos anteriormente utilizados em sistemas Probat Leogap

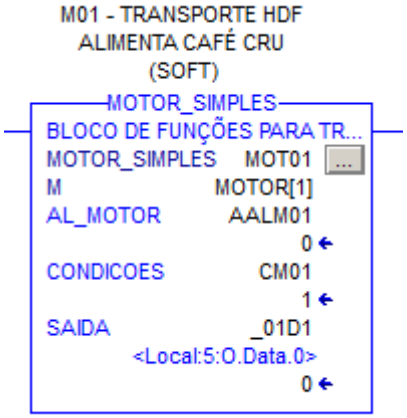

Figura 25: bloco principal de um motor simples

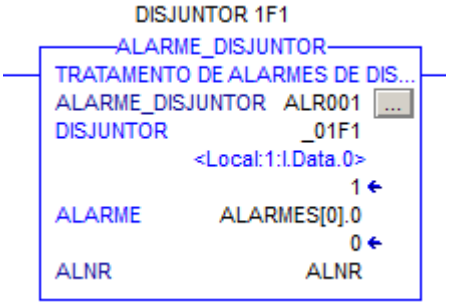

Figura 26: bloco de alarme de disjuntor

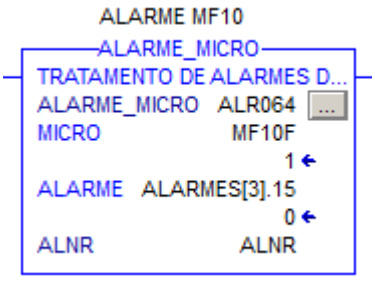

Figura 27: bloco de alarme por sensor fim de curso

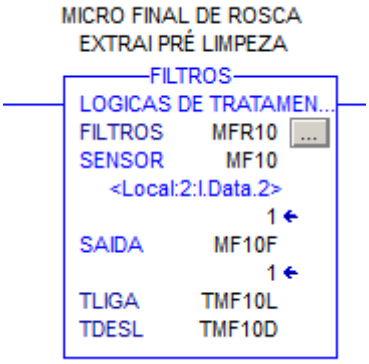

Figura 28: bloco de filtro de sensores

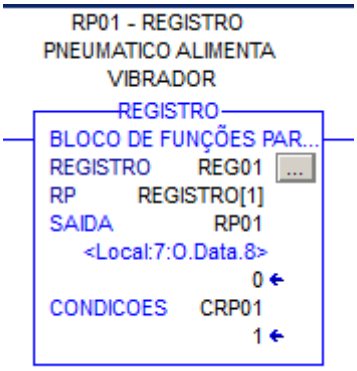

Figura 29: bloco principal de registros pneumáticos

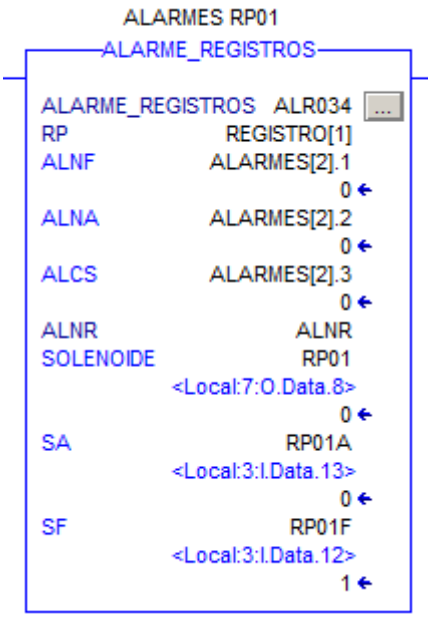

Figura 30: bloco de alarmes de registro pneumático

APÊNDICE B – Novos blocos de motores e registros pneumáticos utilizados pela Probat Leogap

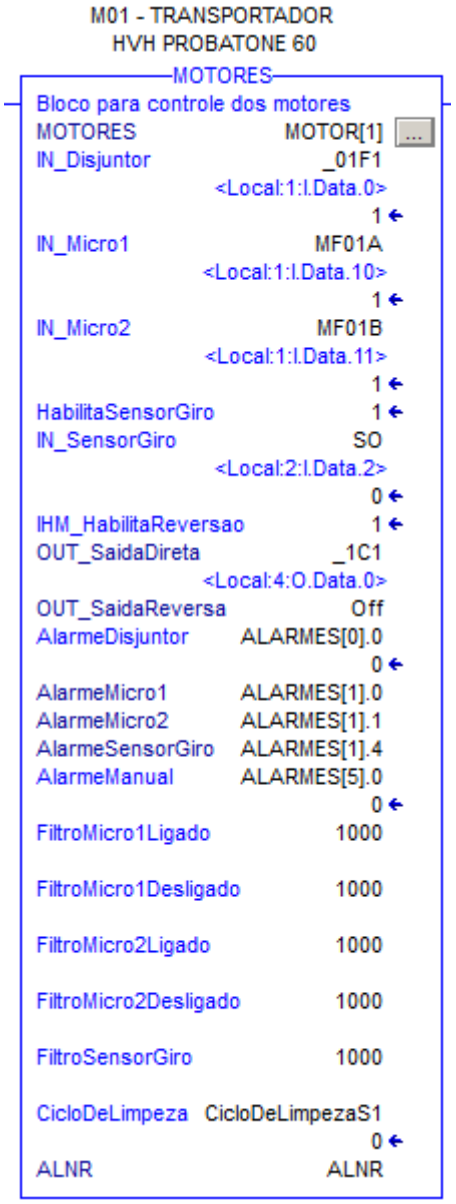

Figura 31: novo bloco principal de motores

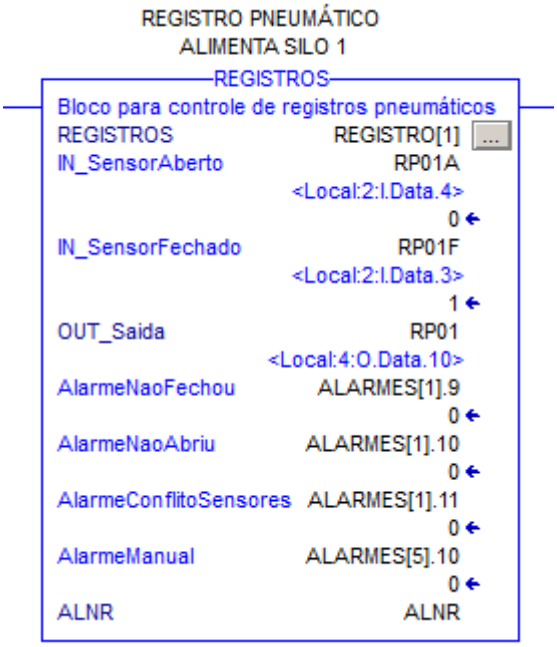

Figura 32: novo bloco principal de registros pneumáticos