CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

CAMPUS IV - ARAXÁ

ENGENHARIA DE AUTOMAÇÃO INDUSTRIAL

ACREUCIMAR APARECIDO LEITE

**FÁBRICA DE SABÃO AUTOMATIZADA UTILIZANDO MICROCONTROLADOR**

ARAXÁ- MG

2014

ACREUCIMAR APARECIDO LEITE

**FÁBRICA DE SABÃO AUTOMATIZADA UTILIZANDO MICROCONTROLADOR**

Trabalho de Conclusão do Curso apresentado ao Centro Federal de Educação Tecnológica de Minas Gerais, unidade descentralizada de Araxá, como requisito parcial à obtenção do título de Engenheiro de Automação Industrial.

Orientador: Prof. Dr. Kleber Lopes Fontoura

Araxá- MG

2014

**TERMO DE APROVAÇÃO**

ACREUCIMAR APARECIDO LEITE

**FÁBRICA DE SABÃO AUTOMATIZADA UTILIZANDO MICROCONTROLADOR**

Trabalho de Conclusão do Curso aprovado como requisito parcial para obtenção do Diploma de Engenheiro de Automação Industrial no curso de Engenharia de Automação Industrial, Centro Federal de Educação Tecnológica de Minas Gerais- unidade descentralizada de Araxá, pela seguinte banca examinadora:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Dr. Kleber Lopes Fontoura

Orientador

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Vinicius dos Reis Alves Ferreira

Examinador

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Dr. Carlos Alberto D. Ramos

Examinador

Araxá- MG

2014

**DEDICATÓRIA**

Dedico a minha esposa Lenize e as minhas filhas Gabriella e Giovanna, pelo amor, carinho e imenso apoio.

**AGRADECIMENTOS**

Agradeço primeiramente a Deus por tudo que tem-me proporcionado.

Agradeço profundamente a minha esposa e filhas que souberam inúmeras vezes aceitar a minha ausência, e sempre me apoiaram e incentivaram, com muito amor e carinho.

Agradeço, também, a todos os meus familiares, colegas de curso e de trabalho os quais de alguma forma, contribuíram para esta conquista.

E por último, mas não menos importante, agradeço a todos os professores e servidores do CEFET, sem os quais, sem dúvida nenhuma, eu não estaria aqui hoje.

[Eu não creio que exista algo mais emocionante para o coração humano do que a emoção sentida pelo inventor quando ele vê alguma criação da mente se tornando algo de sucesso. Essas emoções fazem o homem esquecer comida, sono, amigos, amor, tudo.](http://kdfrases.com/frase/163077)”  
*Nikola Tesla*

**RESUMO**

O presente trabalho trata, em princípio, de descrever teoricamente alguns conceitos e abordagens vinculados ao monitoramento e controle do processo de fabricação de sabão caseiro. Posteriormente detalha-se um pouco a utilização do Arduíno, plataforma de prototipagem eletrônica de hardware livre que é projetada com um microcontrolador Atmel para controle do processo. Aborda-se também o desenvolvimento de uma planta piloto para representar o processo automatizado de fabricação de sabão caseiro utilizando-se óleo de fritura como matéria prima, visando melhoria e otimização do processo. Este trabalho contempla a pesquisa para a construção mecânica, elétrica, instrumentação e automação de uma fábrica de sabão caseiro, destacando-se a utilização de um microcontrolador Arduino, com o qual será possível comandar e monitorar o funcionamento de toda a fabricação. Para tal serão utilizados dois métodos neste projeto: a pesquisa bibliográfica e a pesquisa prática. Na pesquisa bibliográfica serão levantadas informações a respeito da organização e execução das etapas do processo de implementação da fábrica, bem como o estabelecimento do tipo de microcontrolador, linguagem computacional e software a ser utilizado, materiais necessários e o impacto ecológico que a reutilização do óleo de frituras traz ao meio ambiente, impactando na sustentabilidade do planeta. A parte prática será constituída do desenvolvimento e construção da planta piloto, bem como testes de operação da fábrica. Nesse trabalho também se destacam os resultados obtidos, onde são apresentadas as conclusões sobre a implementação, segurança do operador e otimização dessa fábrica, além das sugestões dos autores, desenvolvidas perante as dificuldades e limitações encontradas no decorrer do processo de desenvolvimento do projeto.

**Palavras-chave:** Fábrica de sabão caseiro. Automação. Controle. Arduino. Óleo de fritura, sustentabilidade.

**ABSTRACT**

At first, this work describes theoretically some concepts and approaches about monitoring and control of a homemade soap fabrication process. After, it details the use of Arduino, a free electronics prototyping platform hardware, projected with an Atmel microcontroller, used to process control. This work also describes the development of a pilot plant to represent the homemade soap automated fabrication process, using frying oil as raw material. This document brings the technical research in mechanical, electrical, instrumentation and automation areas, necessary to the construction of an homemade soap factory, highlighting the use of Arduino microcontroller, which will make possible to control e monitoring the fabrication operation. Will be used two research methods in this project: literature and practice. In the first one, will be related information about organization and execution of each step of factory construction. As well as the choice of controller type, programming language and the software to be used. Will be also related the list of necessary materials and the ecological impact caused by the reutilization of frying oil, that should help on Earth sustainability. The development and construction of pilot plant and operational tests will be part of practice. In this work, the results will be highlighted, as well as the conclusions about implementation, operator security and factory optimization, beyond authors suggestions, reached by difficulties and limitations found in the project development.

**Keywords:** homemade soap factory, automation, control, arduino, frying oil, sustainability.

**SUMÁRIO**

[1. INTRODUÇÃO 1](#_Toc403158130)

[2. PROCESSO DE FABRICAÇÃO DE SABÃO 5](#_Toc403158131)

[2.1 História do sabão caseiro 5](#_Toc403158132)

[2.2 A química do sabão 6](#_Toc403158133)

[2.3 Reutilização do óleo de fritura 8](#_Toc403158134)

[2.4 Processo artesanal de fabricação de sabão 12](#_Toc403158135)

[2.5 Importância da automação no controle de processos 12](#_Toc403158136)

[2.6 Metodologia utilizada para fabricação do sabão 13](#_Toc403158137)

[2.7 Segurança do operador para fabricação do sabão 15](#_Toc403158138)

[2.8 Arquitetura da fábrica de sabão 16](#_Toc403158139)

[3. AUTOMATIZAÇÃO DA FÁBRICA DE SABÃO 19](#_Toc403158140)

[3.1 Instrumentação 19](#_Toc403158141)

[3.2 Malha de controle de nível 19](#_Toc403158142)

[3.3 Malha de controle de temperatura 21](#_Toc403158143)

[3.4 Lógica de controle da fábrica de sabão 24](#_Toc403158144)

[3.5 Diagrama eletroeletrônico da fábrica de sabão 28](#_Toc403158145)

[3.6 IHM – Interface Homem Máquina, a estação de operação 28](#_Toc403158146)

[4. DESENVOLVIMENTO DA IHM 31](#_Toc403158147)

[4.1 Criação das TAGS e Configurações de Endereços de Comunicação 34](#_Toc403158148)

[4.2 Criação das telas de operação 39](#_Toc403158149)

[4.3 Descrição operacional das telas 42](#_Toc403158150)

[5. CONFIGURAÇÃO DO ARDUINO COM O SISTEMA 45](#_Toc403158151)

[5.1 Arduino Mega 2560 45](#_Toc403158152)

[5.2 Biblioteca *Modbus Slave* 48](#_Toc403158153)

[5.3 Biblioteca *OneWire* 49](#_Toc403158154)

[6. MONTAGEM 51](#_Toc403158155)

[6.1 Módulo de 8 relés de 12 volts 51](#_Toc403158156)

[7. RESULTADOS E CONCLUSÃO 55](#_Toc403158157)

[7.1 CONCLUSÃO 56](#_Toc403158158)

[APÊNDICE A – PROGRAMAÇÃO DO ARDUINO 60](#_Toc403158159)

[APÊNDICE B – DIAGRAMA DA FÁBRICA DE SABÃO 84](#_Toc403158160)

[APÊNDICE C – PROJETO DO PAINEL 86](#_Toc403158161)

[APÊNDICE D – FLUXOGRAMA DA FÁBRICA DE SABÃO 87](#_Toc403158162)

**LISTA DE FIGURAS**

[Figura 1 - Arranjo estrutural da reação de saponificação 7](#_Toc403158163)

[Figura 2 – Poluição provocada pelo óleo na água 9](#_Toc403158164)

[Figura 3 – Poluição provocada pelo óleo no meio ambiente 11](#_Toc403158165)

[Figura 4 – Arquitetura da fábrica de sabão 17](#_Toc403158166)

[Figura 5 - Sensor óptico de nível 20](#_Toc403158167)

[Figura 6 – Diagrama de malha do controle de nível 21](#_Toc403158168)

[Figura 7 - Sensor de Temperatura DS18B20 23](#_Toc403158169)

[Figura 8 – Diagrama de malha do controle de temperatura 24](#_Toc403158170)

[Figura 9 - Simbologia e típicos de comando liga/desliga 25](#_Toc403158171)

[Figura 10 - Lógica da sequência em automático 26](#_Toc403158172)

[Figura 11 - Lógica do compressor, agitador e bomba 26](#_Toc403158173)

[Figura 12 - Lógica das válvulas pneumáticas 27](#_Toc403158174)

[Figura 13 - Lógica do aquecedor 27](#_Toc403158175)

[Figura 14 - Tela inicial do TatSoft, inserindo novo projeto 31](file:///C:\Documents%20and%20Settings\ulecreuc\Desktop\Acreucimar\CEFET\Relatorio_TCC\TCC_Fabrica_de_Sabao_Automatizada%20(15_09_2014)_corrigido.docx#_Toc403158176)

[Figura 15 - Tela criação do novo projeto 32](#_Toc403158177)

[Figura 16 - Tela ajuste do canal de comunicação 33](#_Toc403158178)

[Figura 17 - Tela ajuste da porta serial 33](#_Toc403158179)

[Figura 18 - Tela criação do Nó de comunicação 34](#_Toc403158180)

[Figura 19 - Tela de Criação de TAGS 38](#_Toc403158181)

[Figura 20 - Tela de Criação de TAGS / Devices 39](#_Toc403158182)

[Figura 21 - Componentes para criação de telas na IHM 40](#_Toc403158183)

[Figura 22 - Configuração dos componentes dinâmicos 41](#_Toc403158184)

[Figura 23 - Tela de entrada de Scripts da IHM 42](#_Toc403158185)

[Figura 24 - Tela principal de operação da fábrica 43](#_Toc403158186)

[Figura 25 - Tela da botoeira de acionamento dos equipamentos 44](#_Toc403158187)

[Figura 26 - Tela de entrada de Set Points 44](#_Toc403158188)

[Figura 27 - Arduino UNO 46](#_Toc403158189)

[Figura 28 - Ambiente de desenvolvimento integrado do Arduino 48](#_Toc403158190)

[Figura 29 - Serial monitor com indicação da temperatura 50](#_Toc403158191)

[Figura 30 - Módulo de relés de 8 canais 52](#_Toc403158192)

[Figura 31 - Circuito de um canal do módulo de relés 53](#_Toc403158193)

**LISTA DE TABELAS**

[Tabela 1 - Detalhamento da palavra 40001 do mapa de memória 35](#_Toc403158194)

[Tabela 2 - Detalhamento da palavra 40003 do mapa de memória 35](#_Toc403158195)

[Tabela 3 - Detalhamento da palavra 40004 do mapa de memória 36](#_Toc403158196)

[Tabela 4 - Detalhamento da palavra 40005 do mapa de memória 36](#_Toc403158197)

[Tabela 5 - Detalhamento da palavra 40015 do mapa de memória 37](#_Toc403158198)

[Tabela 6 - Dados do Arduino Mega 2560 47](#_Toc403158199)

# INTRODUÇÃO

A fabricação de sabão a partir de óleo usado é um processo muito utilizado para reciclar, evitando assim o seu descarte na natureza.

A política dos 3R’s (Redução, Reutilização e Reciclagem) ganha um papel muito importante dentro das providências que estão ao alcance da população em geral para ajudar a minimizar os impactos descritos a médio prazo. A redução, reutilização e reciclagem se tornam palavras chave para retomar aos poucos a pouca sustentabilidade dos países e alavancar o que pode ser uma mudança, recuperação e reeducação ambiental mundial (MAIMON, 1996).

O processo de fabricação de sabão é muito simples, por isso, geralmente é realizado de forma caseira, ou seja, completamente manual. Essa operação manual, apesar de simples, traz inúmeros inconvenientes, principalmente no que diz respeito à eficiência de produtividade e qualidade do produto, na qual não há um padrão de repetibilidade em suas características. Com isso, a substituição das operações que dependem exclusivamente da atuação humana por um controle de processo automatizado torna-se essencial. “O controle de processos destina-se basicamente a: manter os processos em seus pontos operacionais mais eficientes e econômicos; prevenir condições instáveis no processo que podem por em risco pessoas e/ou equipamentos” (BEGA, DELMÉE *et al.,* 2006, p. 434). E, para que essa automação apresente um funcionamento de forma eficiente e segura, vários outros componentes são necessários, podendo ser eles mecânicos, elétricos, eletrônicos e recursos de *hardware* e *software*.

Desde as lógicas discretas de intertravamento até às técnicas de controle avançado e multivariável, o controle automático de processos desempenha um papel fundamental na indústria. Ele pode atuar de forma a aumentar a produtividade e eficiência do processo, promover uma maior segurança operacional, reduzir o impacto ambiental da produção, proteger e aumentar a vida útil dos equipamentos, reduzir a variabilidade do processo produtivo e/ou aumentar a qualidade dos produtos (CARVALHO, 2010).

Em vista da importância de cada um desses componentes, e com o objetivo de demonstrar de forma prática como estes se relacionam, será realizada a automatização de um processo de fabricação de sabão caseiro de forma que se possa monitorar e controlar o processo durante toda a fabricação. Além de exigir toda uma sistemática para o desenvolvimento do projeto, o presente trabalho é de caráter eminentemente prático, integrando e interligando os conteúdos de várias disciplinas da grade curricular do curso de engenharia de automação industrial de forma a atender a uma necessidade do mundo real.

Este trabalho contempla a pesquisa para a construção mecânica, elétrica, instrumentação e automação de uma fábrica de sabão caseiro com todas as funcionalidades através de um microcomputador a fim de conseguir comandar e monitorar o funcionamento de toda a fabricação. Para tal serão utilizados dois métodos neste projeto: a pesquisa bibliográfica e a pesquisa prática. Na pesquisa bibliográfica serão levantadas informações a respeito da organização e execução das etapas do processo de implementação da fábrica, bem como o estabelecimento do tipo de microcontrolador, linguagem computacional e software a ser utilizado, materiais necessários e o impacto ecológico que a reutilização do óleo de frituras traz ao meio ambiente. A pesquisa prática englobará várias ações que resultarão em informações práticas para a montagem final da fábrica. Serão analisadas as formas de recolhimento do óleo e como seu armazenamento deve ser feito até o processamento. Nessa fase, testes de programação serão feitos no software escolhido de modo a se construir a menor rotina para a execução do controlador sem perda de eficiência. A partir dos testes feitos podemos estabelecer como as montagens devem ser executadas e o projeto finalizado.

Desse modo, o objetivo geral deste trabalho é montar um modelo de fábrica de sabão automática que utiliza o óleo de fritura como matéria prima, visando melhoria e otimização do processo.

Para alcançar este resultado, desenvolveremos um supervisório em um *notebook*, o qual terá como finalidade comandar e monitorar os valores lidos pelos instrumentos de campo e determinar os valores do *Set-Point* de funcionamento das variáveis controladas. Também foi montado um circuito de força e controle, que receberá e enviará dados para o supervisório através de uma rede de comunicação com o *notebook.*

Como procedimentos metodológicos, em princípio realizamos pesquisas teóricas através de *sites* na internet, artigos e livros, focando inicialmente no processo de fabricação de sabão caseiro utilizando óleo de fritura usado, buscando compreender suas principais vantagens e contribuições para a preservação do meio ambiente. Posteriormente levantamos os componentes necessários para montagem de uma fábrica de sabão caseiro, sendo eles, *hardwares* e *softwares*. Adquiridos esses componentes, iniciamos a montagem de toda parte física do sistema, assim como a programação do controlador Arduino Mega 2560 e o desenvolvimento do supervisório FactoryStudio e, por fim, efetuamos os ensaios necessários para que o sistema produzi-se um sabão caseiro com as qualidades desejadas.

Nosso trabalho se apresenta em sete capítulos. No primeiro é descrita uma introdução e os objetivos a serem atingidos. No segundo capítulo apresentaremos os pressupostos teóricos necessários ao desenvolvimento da pesquisa, a saber: apresentação de uma introdução sobre processo de fabricação de sabão, sua história, química, reutilização do óleo de fritura, processo artesanal de fabricação de sabão. Além disso, abordaremos sobre a importância da automação no controle de processos, metodologia utilizada para fabricação do sabão, segurança do operador para fabricação do sabão e a arquitetura da fábrica de sabão protocolo. No terceiro capítulo focaremos no desenvolvimento da automatização da fábrica de sabão, desenvolvendo e analisando detalhadamente as malhas de controles envolvidas, com seus respectivos instrumentos. No quarto capítulo, focaremos no desenvolvimento da IHM utilizando o FactoryStudio e efetuaremos as configurações para que possa haver a comunicação do *notebook* com os controladores Arduino Mega 2560. No quinto capítulo, será apresentado o Arduino Mega 2560. Serão mostradas as configurações a serem realizadas e as bibliotecas utilizadas na programação do Arduino, assim como o *software* utilizado. No sexto capítulo, será apresentado as etapas necessárias a montagem do painel de força e controle. Mostrando as ligações entre Arduino Mega 2560, dispositivos de entradas e saídas e fontes de alimentações. No sexto capítulo serão descritos os resultados e melhorias realizadas durante a montagem da fabrica de sabão e ajustes realizados para atingir o funcionamento adequado para produzir um sabão de qualidade. E, por fim, no sétimo capítulo, será apresentada a conclusão referente ao trabalho.

# PROCESSO DE FABRICAÇÃO DE SABÃO

A forma mais simples da fabricação de sabão consiste basicamente em misturar alguns reagentes. Os quais provocam uma reação entre si resultando produção de sabões. Apesar da simplicidade, é necessário conhecer todos os princípios envolvidos. Por isso, será detalhado desde a sua descoberta, suas características químicas, seus aspectos ambientais até a importância de se buscarem métodos mais eficientes nos processos de fabricação, utilizando recursos de tecnologia.

# 2.1 História do sabão caseiro

Não é possível afirmar, com certeza, o local e a data do seu surgimento, pois cada historiador possui uma teoria diferente a respeito do surgimento do sabão e sua utilização por parte dos seres humanos. O historiador Plínio (ZANIN *et al.,* 2001) relata sobre o sabão em meados do ano 70 a.C.. Entretanto, Alberici e Pontes (2004) apontam para a existência de um produto muito parecido com o sabão, encontrado em locais relacionados com a antiga Babilônia, e datados de aproximadamente 2800 a.C., mostrando um processo de fabricação muito semelhante ao atual, em que a gordura era fervida junto com cinzas.

De acordo com os autores, o nome saponificação tem origem relacionada ao monte Sapo, onde eram realizados sacrifícios de animais. Em períodos de chuva, a água levava uma mistura de sebo e cinzas, desses sacrifícios, formando uma borra (sabão). Essa borra que caía nas margens do rio Tibre era utilizada pelas mulheres na lavagem das roupas, pois as deixava muito mais limpas. Este foi o primeiro relato da utilização do sabão para limpeza (ALBERICI; PONTES, 2004).

A descoberta realizada pelas mulheres daquela região, apesar de empírica, possui uma explicação científica para a eficiência do sabão na remoção de certos tipos de sujidade.

A água por si só não consegue remover certos tipos de sujeira, como, por exemplo, restos de óleo. Isso acontece porque as moléculas de água são *polares* e as de óleo, *apolares*. O sabão exerce um papel importantíssimo na limpeza porque consegue, por assim dizer, *jogar nos dois times*, no das substâncias polares e no das apolares. Sendo o sabão um sal de metal alcalino (mais frequentemente de sódio) de ácido carboxílico, ao se dissolver na água sofre um processo de dissociação semelhante àquele de qualquer sal solúvel, fornecendo o cátion do metal e o ânion carboxilato. (OLIVEIRA, 2005).

Em 1791, tem-se o registro da primeira patente do processo de fabricação do sabão. Já de acordo com Wandas e colaboradores (2004), foram encontrados em cilindros de barros datados 2008 a.C. uma mistura de gordura animal fervida com cinzas de madeira. Os estudos mostram que essa mistura era utilizada para tratamento de feridas e doenças da pele, assim como cosmético para penteados artísticos. Não se tem relato da utilização dessa mistura para limpeza humana naquele período.

# 2.2 A química do sabão

Para que a fabricação de sabão ocorra, é necessário entender as reações químicas que envolvem o processo. A reação química que ocorre na formação do sabão é chamada de saponificação, que é a reação de ácidos graxos com óleos. A saponificação é feita à quente. Os ácidos graxos normalmente usados são o oléico, o esteárico e o palmítico, encontrados sob a forma de ésteres, oleatos, estearatos e palmitatos, nas substâncias gordurosas. Uma vez que óleos e gorduras são ésteres, eles sofrem reação de hidrólise ácida ou básica. A hidrólise ácida produzirá simplesmente o glicerol e os ácidos graxos constituintes. Já a hidrólise básica produzirá o glicerol e os sais desses ácidos graxos.

Pois bem, esses sais são o que chamamos de sabão. Assim, aquecendo gordura em presença de uma base, realizamos uma reação química que produz sabão, um sal de ácido graxo, isto é, um sal de ácido carboxílico de cadeia longa. Essa reação, a hidrólise básica de um triéster de ácidos graxos e glicerol, é chamada de saponificação. A glicerina é um subproduto da fabricação do sabão que é utilizado em produtos de higiene como o sabonete. Na FIG.1 se pode visualizar o arranjo estrutural dessa reação orgânica. (PERUZZO e CANTO, 2002).

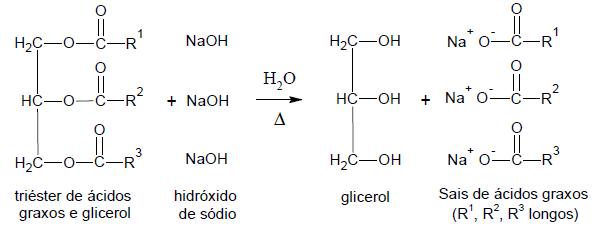


Figura 1 - Arranjo estrutural da reação de saponificação

Fonte: Peruzzo e Canto, 2002

Assim, aquecendo gordura em presença de uma base, realizamos uma reação química que produz sabão. Essa reação, a hidrólise básica de um triéster de ácidos graxos e glicerol, é chamada de saponificação. Essa reação geral de produção do sabão pode também ser conseguida utilizando-se matérias primas de diversas origens. O triglicerídeo, que é o tipo de gordura mais abundante na natureza, é usado como matéria prima na fabricação do sabão, e pode ser proveniente do sebo de origem animal, dos óleos vegetais ou da mistura de ambos de acordo com Mcmurry (2006). Dessa forma, resumidamente, o processo de fabricação de sabão pode ser representado conforme a seguir:

A glicerina (ou glicerol) é um subproduto da fabricação do sabão. Por esse motivo, toda fábrica de sabão também pode vender glicerina. Ela é adicionada aos cremes de beleza e sabonetes, pois é um bom umectante, isto é, mantém a umidade da pele. Já os sais de ácido graxos são o que chamamos de sabão (PERUZZO e CANTO, 2002).

Óleo ou gordura +base 🡺 glicerol +sabão

A saponificação baseia-se na adição de uma base forte, neste caso a soda, ao sistema contendo os triglicerídeos, nesse caso, o óleo de fritura. Assim, se pudermos determinar a quantidade de base necessária para saponificar todo o conteúdo lipídico de uma amostra (o que pode ser feito através da simples titulação com um ácido), teremos o chamado Índice de Saponificação (I.S). Esse índice é definido como a massa de base necessária para saponificar 1g de óleo, e é muito útil na caracterização do óleo ou gordura.

Desse processo temos a descrição de vários reagentes, já que de acordo com a aplicação e ajustes do índice de saponificação é que determinamos a proporção de reagentes, temperatura e o tempo de reação para uma melhor conversão, pois o óleo *in natura* tem uma conversão natural de 60%. Um aumento na eficiência da reação é conseguido quando a mistura de soda e água é colocada para aquecer e agitar, conseguindo na maioria das vezes atingirmos uma conversão próxima a 100%. Mas quando os ácidos graxos são combinados, e utilizamos diversos tipos, temos de atuar na temperatura e relação de óleo x volume total, e óleo x soda.

# 2.3 Reutilização do óleo de fritura

Com a evolução e desenvolvimento dos seres humanos, torna-se necessário cada dia mais se preocupar com as questões ambientais, com a sustentabilidade do planeta, assim como a possibilidade de escassez dos recursos naturais. Por ser o óleo um dos itens mais consumidos na refeição do brasileiro, e por ele ser associado às questões referenciadas anteriormente, daremos uma ênfase no assunto pois, segundo Nogueira e Beber (2009), além da utilização em frituras, ele está intrinsecamente presente na composição de leguminosas, carnes e frutas e pode ser parte integrante na fabricação de pães e massas. Duas características presentes no óleo vegetal são importantes, tanto para o manuseio e consumo como também para o meio ambiente, que são a saturação e a insolubilidade em meio aquoso, respectivamente.

Os óleos são substâncias insolúveis em água (hidrofóbicas), pois são formados predominantemente por ésteres de triacilgliceróis (produtos resultantes da esterificação entre o glicerol e ácidos graxos), porém são solúveis em solventes orgânicos. Em função desta imiscibilidade e por possuir densidade menor que a água (ser mais leve), quando lançados em mananciais, emerge para a superfície, conforme FIG. 2.



Figura 2 – Poluição provocada pelo óleo na água

**Fonte:** Revista Eletrônica Lubes em foco, matéria / Rerrefino: um enfoque ecológico.

<<http://www.lubes.com.br/edicoes/edicao02/index.html> > Visitado em: 8 de set. 2013/ AGO/SET 2007/ ANO 1, Nº 2

Na superfície formam películas oleosas as quais, além de diminuir e/ou acabar com a tensão superficial da água, segundo Sampaio (2003) e Almeida (2002), dificultam a entrada de luz e oxigenação da água, comprometendo a base da cadeia alimentar aquática, os fitoplanctons, e resultam, consequentemente, na mortandade de peixes e de todas as formas de vida no local afetado. Também contribui para a formação de bancos de lamas nos rios, contaminando, assim, águas que, por vezes, são usadas inadequadamente para o consumo humano.

De acordo com Alberici e Pontes (2004) e Almeida (2002), quando descartado na rede coletora de esgotamento sanitário, o óleo se acumula nas canalizações formando uma crosta, contribuindo para a sua obstrução. Para a retirada do óleo ou desentupimento das tubulações, que deve ser realizada periodicamente, são utilizados produtos químicos tóxicos como a soda cáustica. Tais medidas aumentam o custo do tratamento de esgoto em até 45% segundo Biodiselbr (2007), além de contribuir para a ocorrência de enchentes e proliferação de doenças, caso a desobstrução da tubulação não ocorra em tempo hábil.

Quando lançado diretamente no solo, segundo Nogueira e Beber (2009), o óleo ocupa os espaços que, naturalmente, seriam ocupados pela água e pelo ar, ou seja, provoca a impermeabilização do solo. Como conseqüência, a fauna e a flora deste local ficam impedidas de absorver os nutrientes e acabam morrendo. Dessa forma, as sementes não conseguem germinar e o solo fica impróprio para o cultivo. A reparação deste solo, tornando-o fértil e apto a novos cultivos torna-se caro e difícil. Outro aspecto que precisa ser considerado é que, ao ser lançado no solo, o óleo entra em contato direto com a água que percola e escoa superficialmente e, dependendo das características físicas químicas do solo e outras características como relevo e regime climático, pode migrar, atingindo mananciais hídricos e, até mesmo, o lençol freático.

Outra forma de descarte adotado por muitas pessoas é colocar o óleo no lixo comum que irá parar em aterros sanitários. Como o esquema típico de um aterro é a compactação do lixo, e se o aterro não possuir um sistema que impeça a infiltração do óleo nos taludes, como por exemplo, uma geomembrana de polietileno de alta densidade que promove a cobertura de uma área no solo impedindo o vazamento ou infiltração de efluentes, o óleo que é descartado no lixo comum terá novamente como destino final as infiltrações e contaminação do lençol freático, além de diminuir a vida útil dos aterros sanitários.

Na FIG. 3 é possível visualizar a poluição provocada pelo óleo quando este é descartado na pia, no solo, ou no lixo comum.

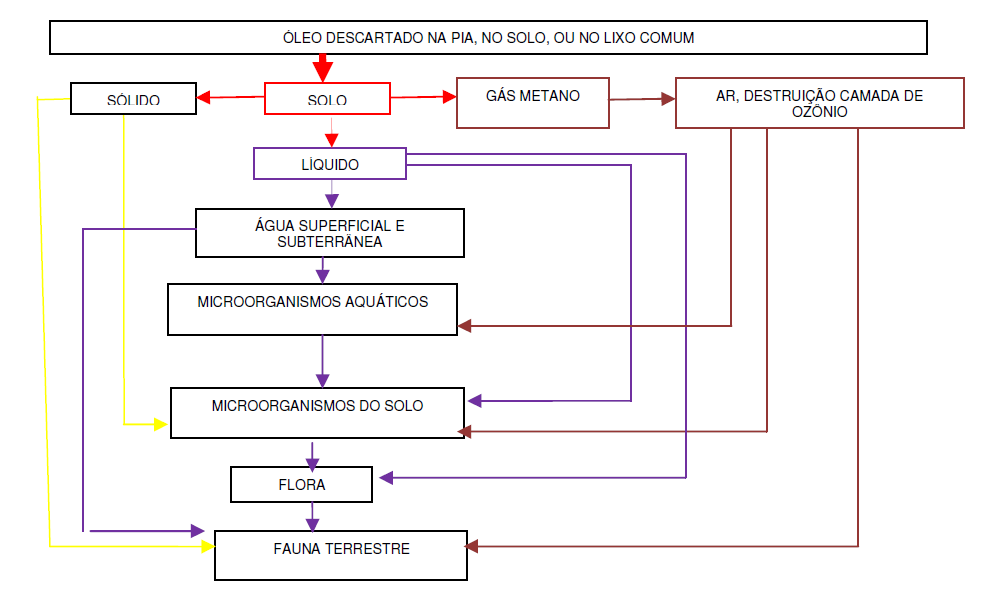


Figura 3 – Poluição provocada pelo óleo no meio ambiente

**Fonte:** Revista Eletrônica em Gestão, Educação e Tecnologia Ambiental, v(5), n°5, p. 813 - 824, 2012. (gráfico p 818) - RECICLAGEM DE ÓLEO COMESTÍVEL E FABRICAÇÃO DE SABÃO COMO INSTRUMENTOS DE EDUCAÇÃO AMBIENTAL, Loreni Beatriz Arnold Wildner, Clayton Hillig

Como visto anteriormente, além dos problemas ambientais, o descarte incorreto do óleo pode acarretar em grandes prejuízos econômicos e causar sérios problemas para a saúde da população. Apesar de toda a problemática relacionada ao descarte do óleo utilizado para fritura, não existe um modelo ideal para destinação desse óleo. Segundo Alberici e Pontes (2004), apenas 1,5 % dos resíduos urbanos produzidos, são reciclados. Dentre esses resíduos está o óleo, que é o que mais polui o meio ambiente, devido as suas características químicas.

Para o resíduo de óleo já existem vários projetos de fabricação de sabão ecológico com resultados bastante satisfatórios. Com isso, a reciclagem desse produto usado em frituras agrega aspectos importantes, principalmente educacionais, culturais, sanitários, ambientais, econômicos, sociais, políticos e institucionais, contribuindo para a formação de cidadãos mais conscientes.

Segundo (VALLE, *et al.*, 2009, p. 7), o aproveitamento de óleo usado em cozinhas como proposta de reciclagem de materiais descartáveis, visando à redução do impacto ambiental e na saúde humana, é uma forma de educar o consumidor sobre consumo consciente, para prover um meio ambiente melhor, sendo a produção de sabão a partir do resíduo de óleo comestível doméstico viável.

# 2.4 Processo artesanal de fabricação de sabão

Devido às quantidades envolvidas, principalmente em função de uma das matérias primas terem origem de descarte, é muito comum realizar a fabricação de sabão de forma totalmente manual. Esse procedimento artesanal afeta diretamente a qualidade do produto final, uma vez que depende da atuação e atenção do operador. Isso sem mencionar as possibilidades de acidentes que podem provocar perdas de matérias primas ou danos pessoais. Daí a importância de buscarmos meios simples e de baixo custo para implementar uma completa automação nesse processo de fabricação de sabão.

# 2.5 Importância da automação no controle de processos

Com a globalização e a forte concorrência que se encontra na atualidade, é necessário buscar cada dia mais a perfeição no que se diz respeito à controle de processo, buscando obter produtos de alta qualidade, com melhores condições de rendimento e diminuindo a interferência humana no processo. E, consequentemente, isso aumentaria a segurança para todos os envolvidos no processo de fabricação.

Segundo Ogata (1967), controlar um processo significa atuar sobre ele ou sobre as condições a que o processo está sujeito, de modo a atingir algum objetivo. Por exemplo, podemos achar necessário ou desejável manter o processo sempre próximo de um determinado estado estacionário, mesmo que efeitos externos tentem desviá-lo dessa condição. Esse estado estacionário pode ter sido escolhido para atender melhor aos requisitos de qualidade e segurança do processo.

Devido à real disponibilidade de recursos tecnológicos da atualidade para os mais diversos fins, justifica-se a utilização da automação em controle de processo, uma vez que esses recursos são desenvolvidos para transformar tarefas manuais repetitivas em processos automáticos de forma a maximizar a produção, minimizar custos e proporcionar o alcance da qualidade, além de eliminar possíveis riscos envolvidos na produção, pois são realizadas por equipamentos projetados para alcançar total eficiência e maior velocidade. Portanto a automação possui um papel vital para a garantia de eficiência operacional do processo.

# 2.6 Metodologia utilizada para fabricação do sabão

O processo para fabricar o sabão caseiro, pode ser dividido em 08 etapas, as quais são:

Etapa 1: Aquecer a água até atingir temperatura de aproximadamente 70ºC;

Etapa 2: Diluir a soda cáustica, que é adquirida no estado sólido;

Etapa 3: Dosar a água pré-aquecida no batedor;

Etapa 4: Dosar a soda pré-diluída no batedor;

Etapa 5: Dosar o óleo usado de fritura no batedor;

Etapa 6: Agitar e aquecer até atingir a temperatura ideal para saponificação;

Etapa 7: Vazar a solução de sabão em uma forma;

Etapa 8: Aguardar a cura do sabão.

Essas etapas atualmente são realizadas pelas donas de casa de forma completamente manual, desde a diluição da soda até a cura do sabão.

O projeto propõe realizar todas as etapas do processo de forma completamente automática, sendo necessário informar apenas a concentração do sabão, ou seja, digitar na IHM (Interface Homem Máquina) as porcentagens de água, óleo, soda, álcool, e o valor da temperatura da água de acordo com a acidez com que se deseja produzir o sabão.

Após digitar as porcentagens de água, óleo, soda e álcool, o programa ficará aguardando a permissão para iniciar uma nova batelada. A permissão será concedida através de um *click* no botão de iniciar partida da IHM da fábrica de sabão.

Com a permissão de iniciar habilitada, inicia-se o processo de fabricação propriamente dito. A IHM envia um sinal para o microcontrolador Arduino que, por sua vez, aciona uma saída que irá atuar sobre um relé e ligar a bomba responsável em transferir a água com seus 70ºC para o reator, até atingir o nível desejado. Obtendo o nível desejado de água a 70ºC, o programa do Arduino desabilita a saída que liga a bomba de água e habilita a saída que aciona o relé da solenoide, e esta inicia a adição de soda diluída. Na fase seguinte, o Arduino desabilita esta saída e habilita a saída que liga a transferência de óleo. Este, por sua vez, quando atinge o nível, desliga o relé da solenoide do óleo e liga a solenoide do álcool. Concluído a dosagem do álcool, o Arduino desliga a referida solenoide e liga o agitador do tanque reator, que permanecerá ligado por um tempo ajustado até alcançar a saponificação adequada. Finalizando o tempo de agitação, o agitador será desligado e uma sirene será acionada indicando que está na hora de vazar o sabão para forma.

Para realizar o processo de acordo com essas perspectivas, serão utilizados conhecimentos adquiridos no curso de Engenharia de Automação Industrial para unir a tecnologia industrial à domótica, de forma a atingir o objetivo esperado, com o menor custo possível.

A fábrica de sabão disponibilizará opção para executar o processo de forma semi-automática, ou seja, o operador terá liberdade para dosar os reagentes da forma que julgar necessário, sendo preciso apenas passar o processo de automático para manual. Esta seleção será efetuada através da IHM, assim como os comandos de liga e desliga da bomba de água, da solenoide da soda, da solenoide de óleo, da solenoide do álcool, do aquecedor e do agitador.

Para o funcionamento, tanto em automático como em manual, a fábrica será composta basicamente por:

01 sistema supervisório;

01 microcontrolador Arduino;

01 tanque reator;

01 agitador;

01 aquecedor;

04 reservatórios, sendo um de água a 70ºC, um de soda diluída, um de álcool e o último, de óleo usado em fritura .

# 2.7 Segurança do operador para fabricação do sabão

A preocupação em atender às legislações e normas de segurança deve ser levada em consideração em todo e qualquer projeto que pretende atender o mercado atual, uma vez que a segurança de todos os envolvidos é um dos requisitos básicos exigidos e de direito dos consumidores e da sociedade.

Com uma fábrica de sabão caseiro não podia ser diferente, pois a soda cáustica, que é um de seus reagentes básicos, oferece vários riscos à saúde e ao meio ambiente. De acordo com a ficha de informações de produto químico, a soda pode causar queimaduras severas nos olhos, pele e tecidos que entrem em contato com o produto ou provocar irritações do trato respiratório e edema pulmonar se inalada, não se restringindo somente a essas consequências, pois trata-se de um produto altamente corrosivo (SUPERQUÍMICA, 2011).

Outro aspecto relacionado com a segurança deste projeto é a temperatura, seja esta proveniente do aquecimento intencional dos reagentes ou da liberação de calor devido a reações químicas ocorridas durante o processo.

Neste tipo de operação em que as condições operacionais oferecem risco às pessoas a utilização de automação através de controle de processos é essencial (BEGA, DELMÉE, *et al.*, 2006).

Um dos pontos fortes e de destaque deste trabalho é a capacidade de controlar os riscos existentes no processo de fabricação de sabão caseiro, através da eliminação da interferência direta do operador no processo e, assim, diminuir consideravelmente os riscos existentes, de forma a assegurar a integridade física do operador durante a operação de fabrico.

# 2.8 Arquitetura da fábrica de sabão

Para a automatização proposta da fábrica de sabão, será necessária a integração de diferentes equipamentos e instrumentos formando um completo e robusto sistema de controle.

Os principais equipamentos, componentes e instrumentos desse sistema são:

Equipamentos mecânicos, sendo tanques, reator, hélice e tubos;

Equipamentos elétricos, sendo motores, relés e cabos;

Equipamentos pneumáticos, sendo válvulas, compressor e mangueiras;

Equipamentos eletrônicos, sendo o módulo microprocessado do Arduíno Mega 2560;

Instrumentos, sendo o sensor de nível e o sensor de temperatura;

Equipamentos de *hardware*, sendo o microcomputador;

Recursos de *software*, sendo o programa do Arduíno e o aplicativo da estação de operação.

Conforme pode ser observado na arquitetura apresentada na FIG. 4, os vários sensores medem continuamente as variáveis de processo e levam os valores até as entradas do microprocessador. Assim que esses sinais são coletados e processados conforme o programa que está rodando dentro do Arduíno, são tomadas ações em que o controlador envia sinais elétricos de comando para que os atuadores realizem as devidas correções no processo.

Todo esse processo - ler entradas, processar e atualizar saídas - é realizado em milésimos de segundo numa rotina infinita. Esta rotina é utilizada no controle do processo já descrito, inclusive disponibilizando todos os dados *on line*, ou seja, sinalizando o estado de tudo que está ocorrendo no processo produtivo de modo a facilitar o monitoramento e controle do processo.

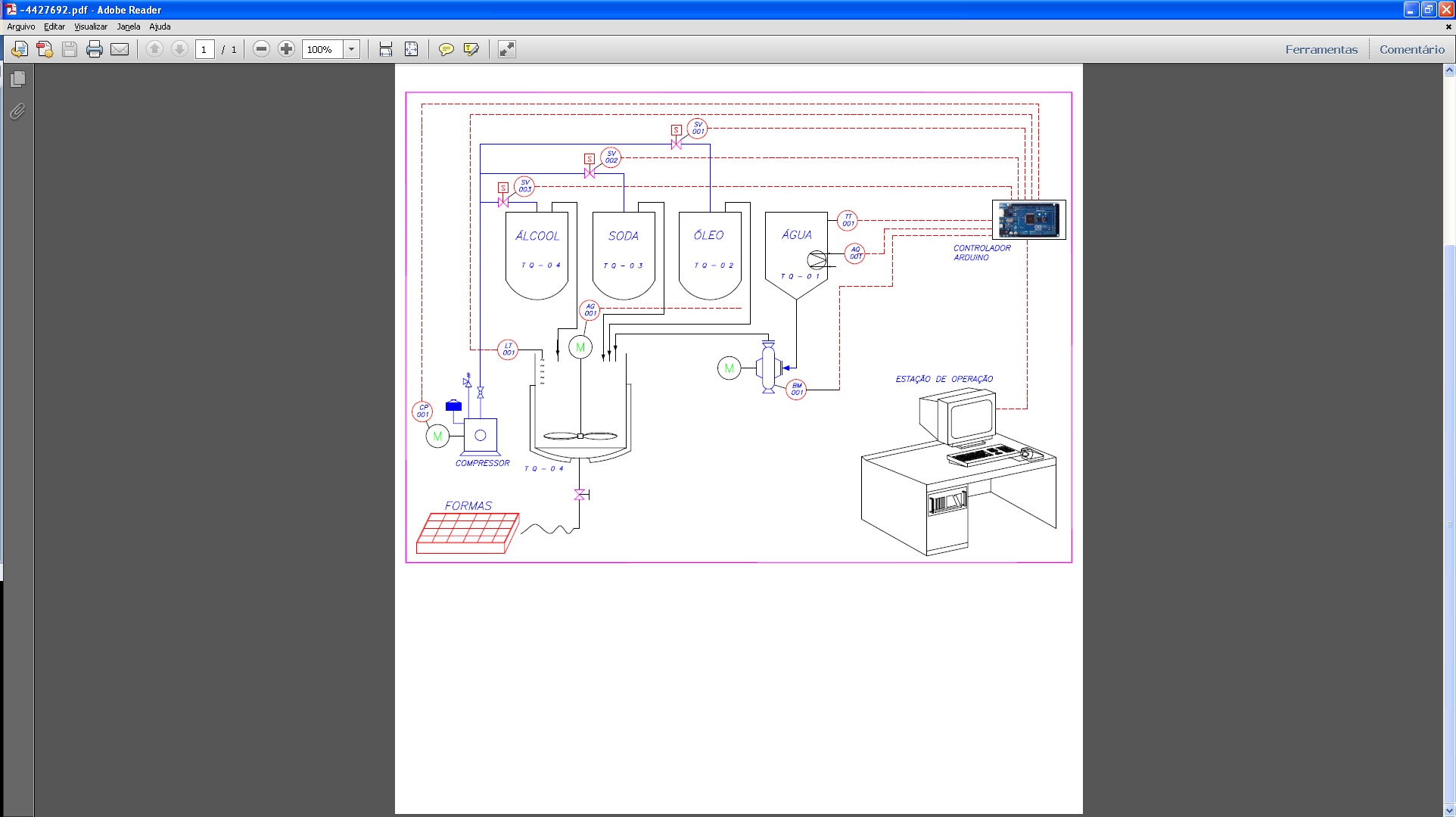


Figura 4 – Arquitetura da fábrica de sabão

**Fonte:** Autoria própria

É importante salientar que o presente trabalho não entrará nos detalhes da seleção e especificação de todos os dispositivos utilizados no processo de automação desta fábrica, pois é um assunto extensivo que envolve inclusive conhecimentos de características de materiais. Dessa forma, baseado em experiência de outros trabalhos industriais utilizamos equipamentos já comprovados para suportarem tais condições de processo.

# AUTOMATIZAÇÃO DA FÁBRICA DE SABÃO

Para um melhor entendimento de todo o sistema automatizado da fábrica de sabão é necessário analisar detalhadamente as malhas de controle envolvidas, com seus respectivos instrumentos. Essas são poucas e de fácil entendimento.

# 3.1 Instrumentação

Instrumentos são equipamentos necessários para estabelecer e manter padrões de qualidades que identificam um produto a ser produzido. São usados para monitorar e controlar as variáveis em um processo ou sistema, com o objetivo de proporcionar um processo produtivo capaz de fabricar um determinado produto dentro das especificações necessárias para manter um padrão de qualidade referente à forma, à medida de acabamento, assim como a outras especificações do produto.

Para atingir o resultado esperado de um instrumento, é necessário que este esteja integrado a um sistema, em muitas das vezes, fazendo parte de uma malha de controle. Em nosso trabalho, os instrumentos estão interligados de forma a compor basicamente duas malhas de controle, as quais são descritas a seguir:

# 3.2 Malha de controle de nível

Essa malha é composta dos seguintes instrumentos:

**3.2.1 – Transmissor de nível a laser:** responsável pela medição e transmissão do nível dos reagentes dentro do reator. Seu sinal de saída é um sinal elétrico em corrente contínua que varia linearmente com a altura do líquido dentro do reator. O sensor óptico de nível utilizado detecta, continuamente, de modo óptico, o nível no tanque reator, gerando sinais de saída conforme a parametrização realizada no mesmo, ou seja, de acordo com os ajustes realizados, o sensor óptico de nível detecta o nível e gera um sinal de 4 a 20 mA em sua saída. Esse sinal é convertido para um sinal de 1 a 5 *volts* que, por sua vez, é ligado na entrada analógica do controlador Arduino. A escolha por este modelo de sensor para medir o nível, ocorreu devido ao fato de o processo possuir insumos corrosivos como a soda, o que poderia causar danos em sensores que entrassem em contato com a solução durante a reação. O sensor óptico de nível produzido pela IFM consegue medir uma faixa de 20 a 1000 cm, sendo indicado pelo fabricante para medição de nível de material granulados, material a granel e líquidos turvos. Como o nível a ser medido está em uma faixa bem inferior ao recomendado pelo fabricante, que é de uma distância máxima de 10 m entre o visor do sensor e o fundo do tanque, foi necessário configurá-lo para operar com a distância mínima D (20cm) recomendada para ficar entre o visor e o material a ser medido, de acordo com a FIG. 05:

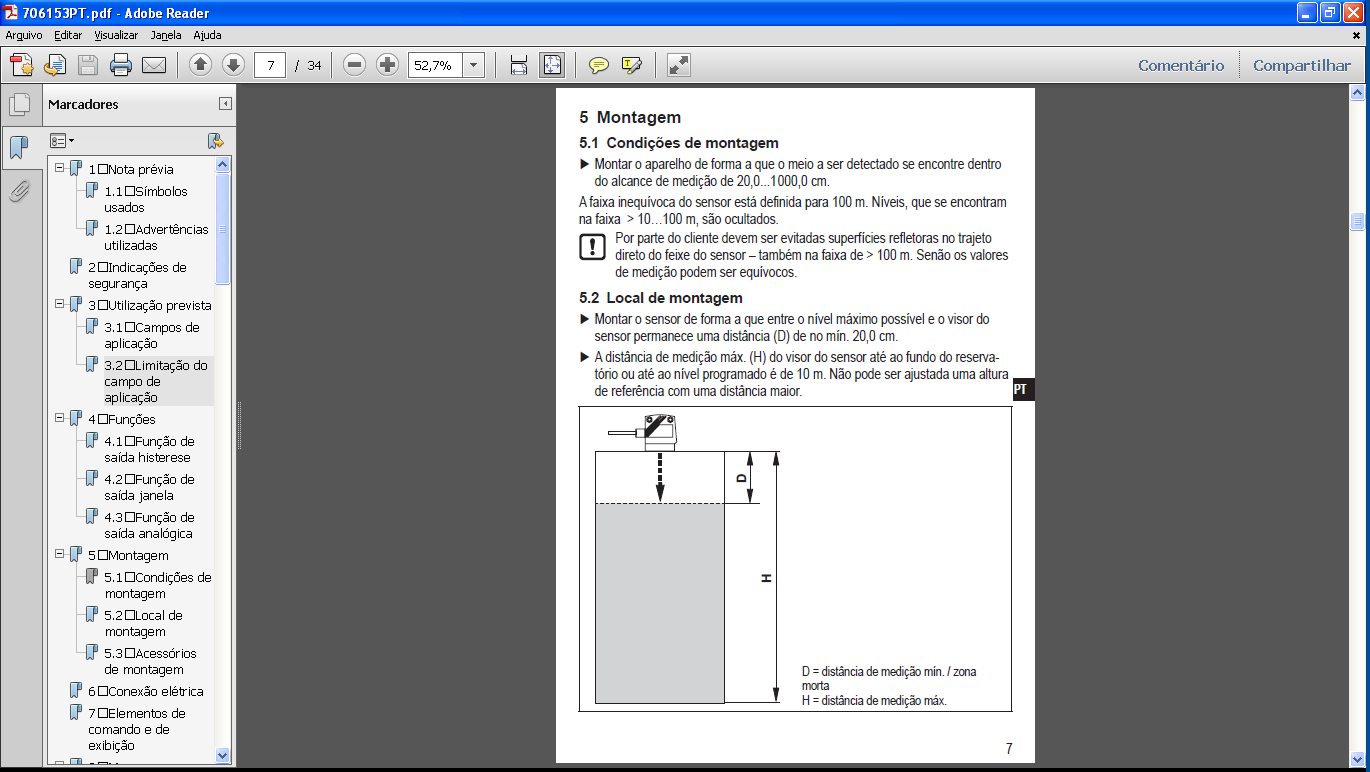


Figura 5 - Sensor óptico de nível

**Fonte:** Manual do fabricante

**3.2.2 – Válvulas solenoides tipo *ON-OFF*:** responsável pela pressurização dos tanques a fim de provocar a retirada dos reagentes levando-os até o tanque reator, conforme as quantidades necessárias na reação. O sinal necessário para alterar o estado de *OFF* para *ON* é um nível de tensão em 12 Vcc. Ou seja, quando enviar o comando para abrir uma válvula solenoide deverá existir uma tensão contínua, em sua bobina, no valor de 12 Vcc.

**3.2.3 – Controlador:** responsável por processar o sinal do transmissor de nível e atuar nas válvulas solenoides conforme lógica programada. O controlador é um microprocessador Arduíno Mega 2560. Seus sinais de entrada e saída são em níveis de tensão contínua de 5 Vcc, logo é necessário utilizar vários recursos para adequar esses sinais aos outros dispositivos externos. Para o transmissor de nível, é utilizado um resistor *shunt* em sua entrada, de forma a obter um sinal de tensão de 1 a 5 Vcc, proporcional ao nível. Já para a saída que aciona as válvulas é utilizado um *drive* com relé.

A malha de controle de nível é apresentada na FIG. 6:

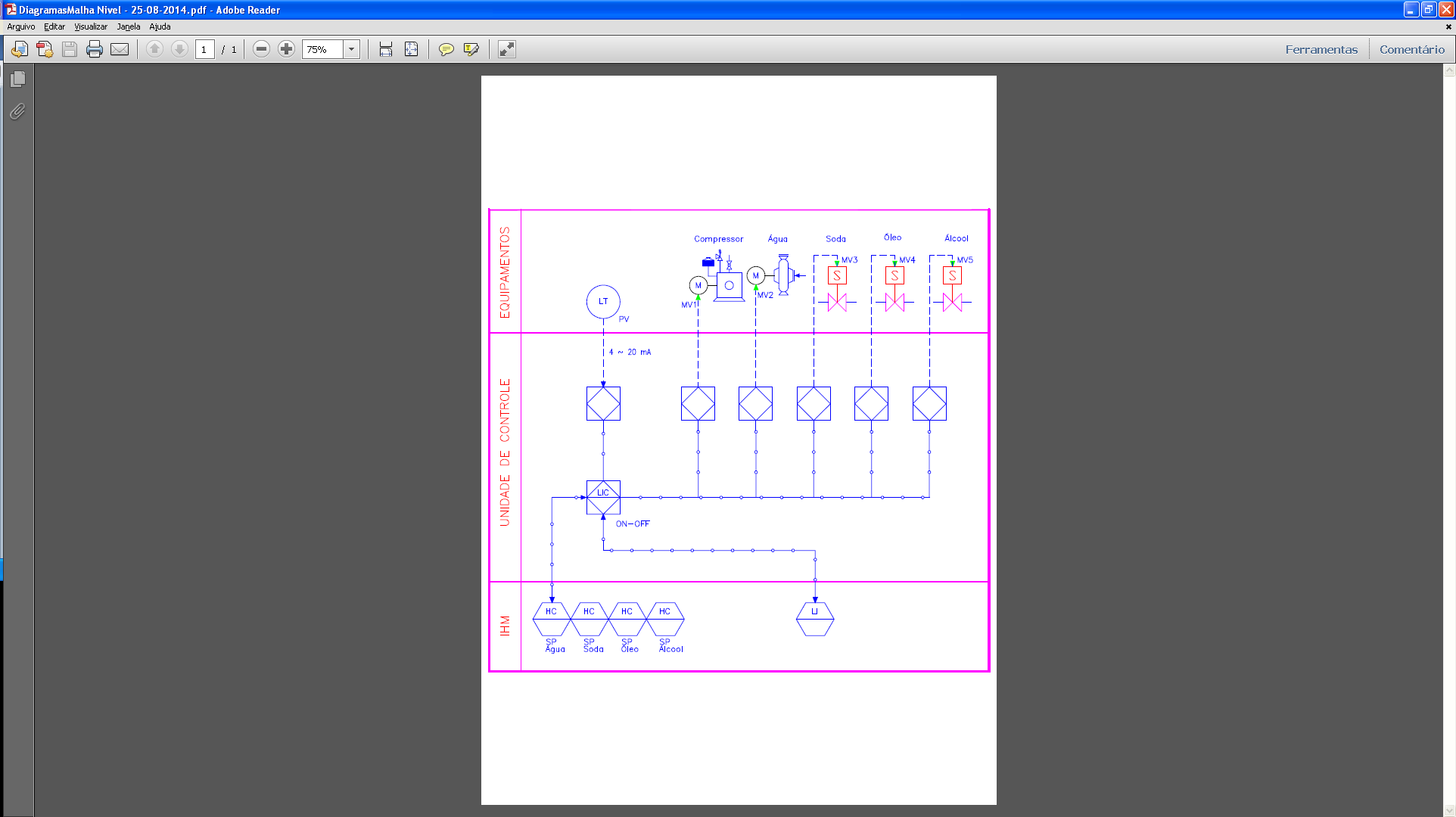


Figura 6 – Diagrama de malha do controle de nível

**Fonte:** Autoria própria

# 3.3 Malha de controle de temperatura

É composta dos seguintes instrumentos:

**3.3.1 – Sensor de temperatura:** Neste projeto foi utilizado o sensor de temperatura DS18B20 que fornece medições de temperatura de 9 a 12 *bits* de resolução configurável. A medição de temperatura é realizada por meio do envio ou recebimento de informações ao longo de uma interface de um-fio (*OneWire*), de modo que apenas um fio precisa ser ligado do Arduino para o DS18B20. O sensor pode ser ligado, utilizando uma derivação da alimentação, direto da linha de dados, dispensando a necessidade de utilizar uma fonte externa de energia (“*parasite Power*”).

Devido às características apresentadas por este sensor e pela facilidade de implementação do mesmo em lógicas de monitoramento e controle de processo, o mesmo tornou-se a melhor opção para medição de temperatura no desenvolvimento deste trabalho.

Características:

Utilizável com 3.0V a 5.5V alimentação /dados;

Precisão de ±0.5ºC na faixa de -10 a +85ºC;

Gama de temperatura utilizável: -5,5 a 125ºC (-6.7 a 257ºF);

Resolução selecionável de 9 a 12 *bits*;

Usa interface *OneWire* que requer apenas um pino digital para comunicação;

ID único de 64 – *bits* gravados em cada sensor;

Vários sensores podem compartilhar um pino;

Sistema de alarme programável para uma temperatura limite;

Tempo de consulta inferior a 750ms;

Tubo de aço inoxidável de 6mm X 35mm;

Diâmetro do cabo de 4 mm;

Comprimento 90cm.

Significado dos três fios da interface:

Fio vermelho – VCC;

Fio preto – GND;

Fio amarelo – DATA.



Figura 7 - Sensor de Temperatura DS18B20

**Fonte:** Manual do fabricante

**3.3.2 – Resistência de aquecimento:**Responsável por fornecer a energia térmica à água a ser aquecida, é o elemento final de controle. Devido a sua potência ser de valor elevado, é necessário um contator de potência que será acionado por meio de um dos relés de saída do Arduíno de forma a fornecer uma tensão alternada para esta resistência de aquecimento de 110 Vac.

**3.3.3 – Controlador de temperatura:** Responsável por processar o sinal do transmissor de temperatura e atuar na resistência de aquecimento da água conforme lógica programada. O nosso controlador é um microprocessador Arduíno Mega 2560. Seus sinais de entradas e saídas são em níveis de tensão contínua de 5 Vcc, logo é necessário utilizar de vários recursos para adequar os sinais externos diferentes desse nível de tensão. Para o nosso transmissor de temperatura, foi utilizado o recurso *OneWire*, que dispensa a utilização de qualquer acessório para leitura da temperatura. Já para a saída que aciona a resistência é utilizado um *drive* com relé.

A malha de controle de temperatura é apresentada na FIG. 8:

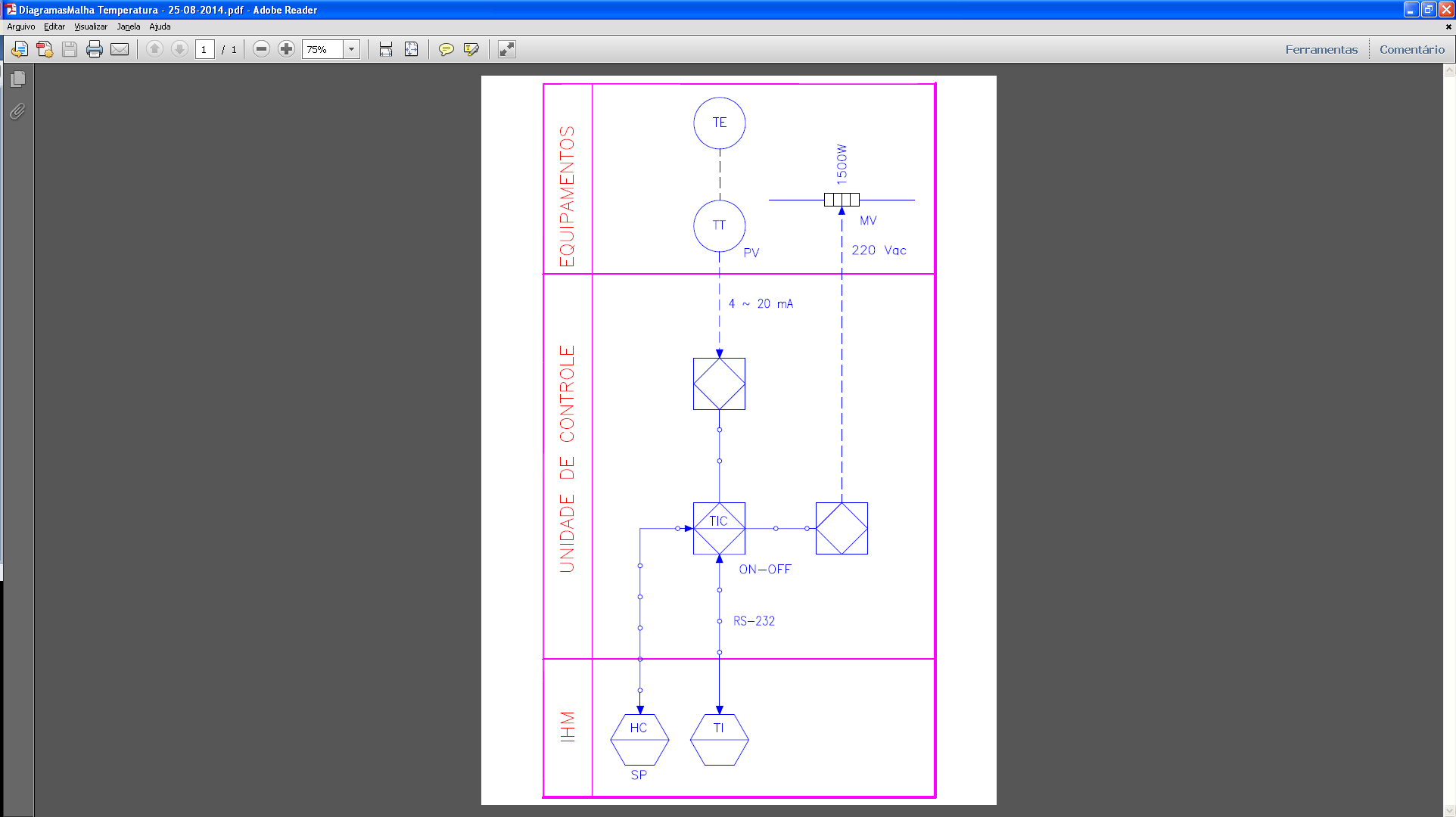


Figura 8 – Diagrama de malha do controle de temperatura

**Fonte:** Autoria própria

# 3.4 Lógica de controle da fábrica de sabão

Para que a fábrica opere de forma automática, uma lógica das sequências operacionais é necessária ser programada na memória do Arduíno. Essa lógica é executada em um *loop* infinito executando sempre as seguintes tarefas: realiza a atualização dos dados de todas as entradas, sejam estes oriundos das entradas analógicas, discretas ou dados originados da porta de comunicação; processa esses dados segundo a lógica programada e então escreve os resultados em suas saídas. Todo esse procedimento é executado continuamente em alta velocidade chegando a poucos milissegundos.

Essa lógica que é o centro inteligente de toda a fábrica de sabão, foi implementada e gravada com o *software* específico do Arduíno Mega 2560, sendo que a mesma deverá ser codificada conforme a sintaxe exigida.

Nas FIG. 9, 10, 11, 12 e 13 são apresentadas as lógicas utilizadas para a fábrica de sabão.

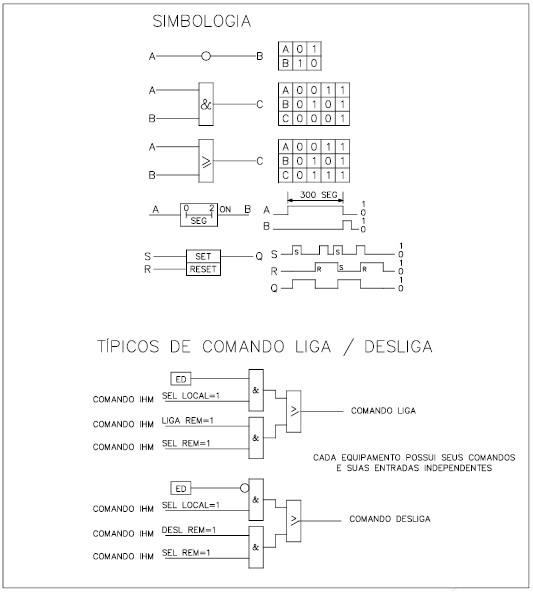


Figura 9 - Simbologia e típicos de comando liga/desliga

**Fonte:** Autoria própria

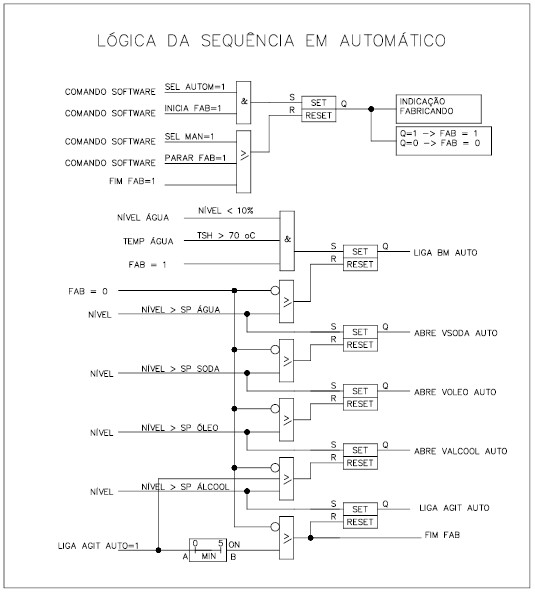


Figura 10 - Lógica da sequência em automático

**Fonte:** Autoria própria

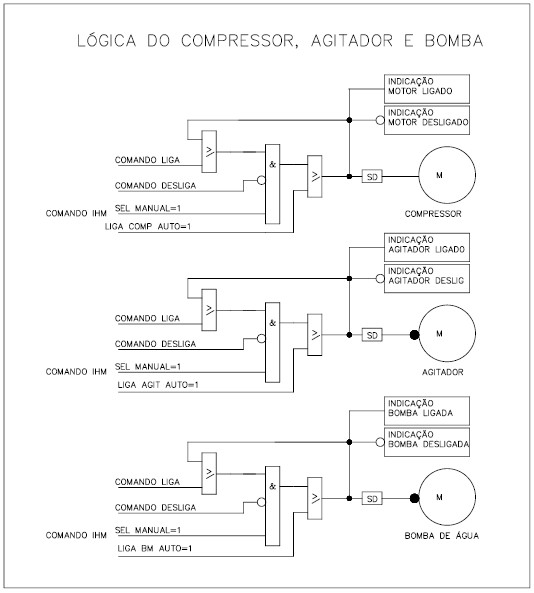


Figura 11 - Lógica do compressor, agitador e bomba

**Fonte:** Autoria própria

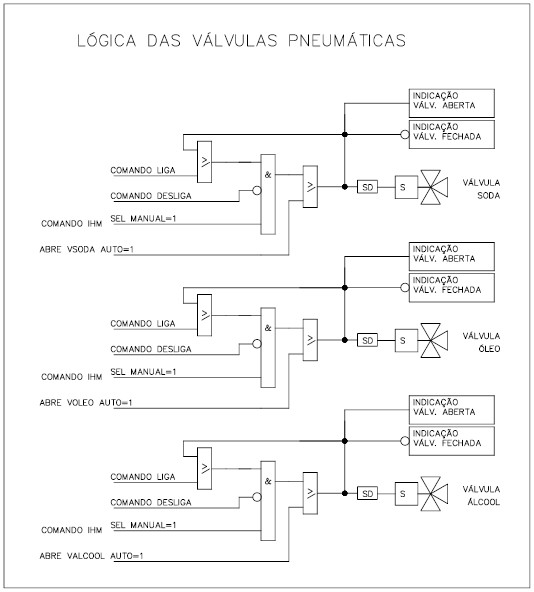


Figura 12 - Lógica das válvulas pneumáticas

**Fonte:** Autoria própria

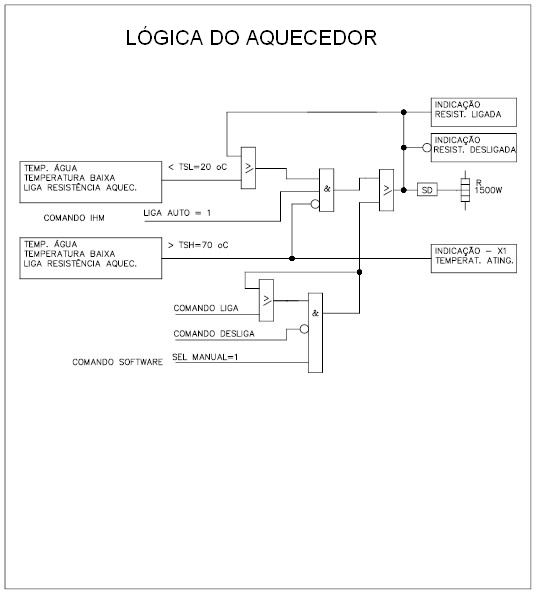


Figura 13 - Lógica do aquecedor

**Fonte:** Autoria própria

Conforme pode ser observado nas figuras anteriores das lógicas, para o compressor existe a opção de ser ligado e desligado pela IHM, e o mesmo deverá sempre ser ligado anteriormente ao início do processo de fabricação do sabão. Isso é necessário para garantir a disponibilidade do ar comprimido quando as válvulas solenoides forem acionadas.

# 3.5 Diagrama eletroeletrônico da fábrica de sabão

Para que a fábrica funcione da forma desejada, é necessária a interligação de diversos dispositivos elétricos e eletrônicos e de instrumentos de forma apropriada permitindo que o sistema consiga enxergar as mudanças no mundo real e possuir meios da atuar de maneira a manter dentro de limites os valores de processo a fim de atingir a melhor qualidade do produto final.

No APÊNDICE B apresentamos o digrama completo da fábrica de sabão.

# 3.6 IHM – Interface Homem Máquina, a estação de operação

Todo o conteúdo apresentado até o momento, apenas é necessário do ponto de vista de construção eletromecânica da fábrica de sabão. Para um operador, essas informações durante a operação normal são irrelevantes. Para ele, o importante é que o sistema ofereça recursos para operar a fábrica de forma totalmente intuitiva com o mínimo de esforço. Assim, tornou-se necessário desenvolver uma interface de operação em que o operador possa visualizar em um único local todas as informações necessárias para monitoramento e controle do processo de fabricação.

Esta IHM ou estação de operação foi desenvolvida com a utilização de uma linguagem de alto nível baseada em todos os recursos do *Windows*. Poderia ser utilizada qualquer linguagem com estas características, porém neste projeto foi adotado o software *FactoryStudio*, que é desenvolvido pela empresa *TATSOFT*, para aplicações industriais. *FactoryStudio* é desenvolvido de acordo com características de *software* para desenvolvimento de Sistemas SCADA que vêm do inglês *Supervisory Control And Data Acquisition*, ou seja, Controle Supervisório de Aquisição de Dados.

Com a evolução da tecnologia e a utilização de recursos computacionais, os Sistemas SCADAs ou apenas Supervisórios de Controle passaram a ser projetados de maneira a disponibilizar o maior número possível de informações ao operador e possibilitando ao mesmo navegar virtualmente por toda a planta, obtendo, assim, informações do processo produtivo em tempo real. Geralmente os Sistemas SCADAs funcionam em conjunto com Controladores Lógicos Programáveis (CLP) que são responsáveis por receber os sinais oriundos de dispositivos e sensores de campo que são enviados para o SCADA que realiza operações de leitura e gravação de valores. Essas informações são tratadas e apresentadas de diversas formas para o operador, tais como gráficos de visualizações, gráficos de tendências dentre outros, buscando sempre uma forma amigável e intuitiva para apresentar tais informações. De posse de todos esses dados o operador pode monitorar e controlar com precisão todo o processo produtivo da respectiva planta, mesmo que os equipamentos de chão de fábrica estejam distribuídos em áreas distintas da planta.

Neste trabalho está sendo utilizando o *software FactoryStudio* que é uma plataforma de *state-of-the-art* e estrutura para criação de sistemas de interface gráfica em tempo real, projetado para fornecer completo controle criativo sobre processos de engenharia e possui uma versão *Express* para pequenos projetos e avaliação do *software*. Com a utilização deste *software* é possível desenvolver aplicativos para os mais variados usos nas indústrias mineradoras, químicas, automobilísticas, alimentícias, petroquímicas, mineroquímicas, nos sistemas de geração e distribuição de energia. Geralmente os sistemas SCADAs oferecem recursos incorporados como *drivers* de comunicação com equipamentos, sistemas de registros contínuos e uma interface gráfica do usuário, conhecida como IHM. Na Interface Homem-Máquina ou simplesmente IHM são disponibilizados recursos gráficos como botoeiras, PIDs, indicações das variáveis do processo tais como temperaturas, pressões, vazões, gráficos de tendências em tempo real. Todos esses recursos juntos mostram o comportamento do processo produtivo da fábrica. Estes, por sua vez, são utilizados para realizar as principais funções de uma IHM, que são:

a) Comando de acionamento remoto;

b) Ajustes de *Set-Points*;

c) Detecção de alarmes;

d) Relatórios de eventos gerados.

Encontramos disponíveis no mercado softwares para desenvolvimento de IHMs dos mais variados fabricantes - Schneider Electric, Rockwell Automation, Elutions, Siemens, SCADABR, TatSoft etc.. - A decisão em utilizar o *software FactoryStudio*, desenvolvido pela TatSoft, foi motivado pelo fato de ser o único *free* com todos os recursos disponíveis e sendo sua única restrinção o número de TAGs menor que 75. Como este projeto não atingiu este número, o *FactoryStudio* foi considerado o ideal para o perfeito desenvolvimento deste trabalho.

# DESENVOLVIMENTO DA IHM

Neste tópico serão apresentadas todas as etapas seguidas para configurar os *drivers* de comunicação para a troca de dados entre a IHM e o Arduino Mega 2560, bem como a sequência de desenvolvimento das telas de operação na IHM. Essas configurações, desenvolvimento de telas e ajustes foram realizados seguindo as instruções do fabricante, que é disponibilizada para *download* juntamente com o *software Express* no site do fabricante (www.tatsoft.com).

Primeiro passo – Após a instalação do *software*, executa-se o *FactoryStudio*. Acessa o icone *New Project* para abrir a tela de configuração do projeto.

Figura 14 - Tela inicial do TatSoft, inserindo novo projeto

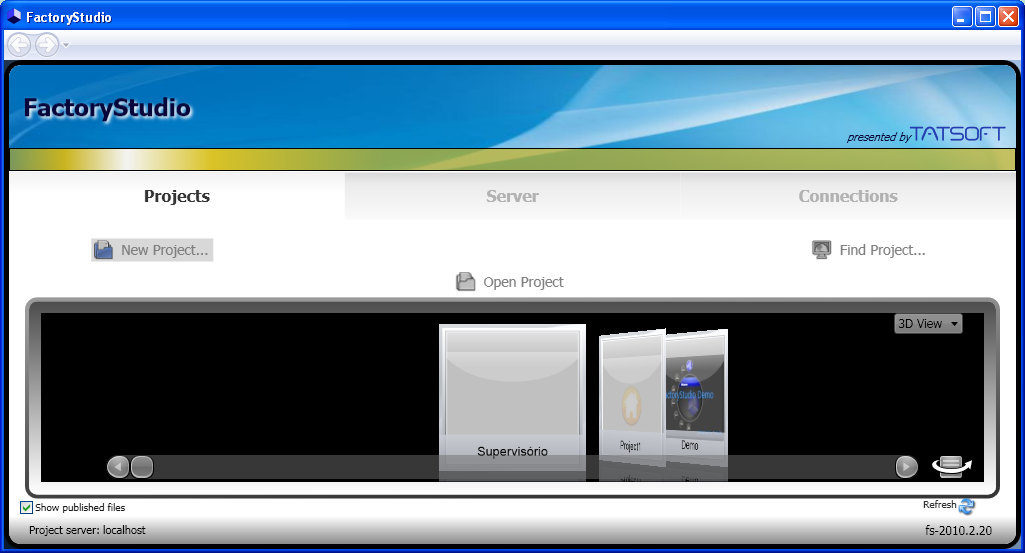


Figura 9 - Tela inicial do TatSoft, inserindo novo projeto

**Fonte:** Autoria própria

Segundo passo – Realiza-se a configuração principal do projeto com nome, descrição, endereço onde serão gravados os arquivos do projeto, família do produto, resolução do *desktop*, código etc..

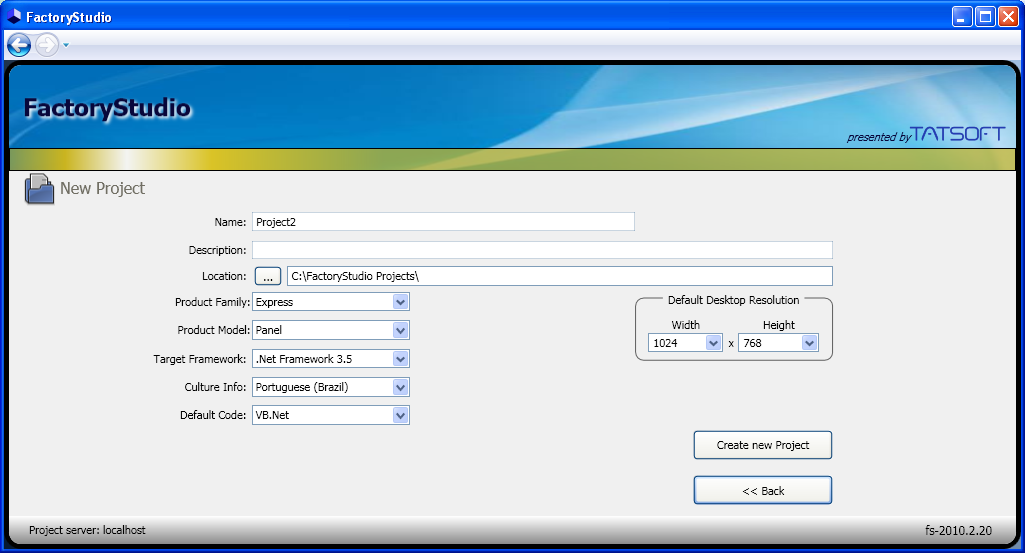


Figura 15 - Tela criação do novo projeto

**Fonte:** Autoria própria

Terceiro passo – terminada a parte de criação do novo projeto, deverá abrir o mesmo para ajustar o *driver* de comunicação a ser utilizado com o controlador Arduino e seus respectivos endereços de acordo com a FIG.11. Esses ajustes são realizados acessando o ícone *Edit*, posteriormente *Devices*, em seguida *Create new* e ajustando de acordo com a comunicação disponíveis no controlador. No caso específico deste projeto, adotamos a comunicação serial, nativa do controlador com o USB, com o protocolo de comunicação *Modbus*. Por isso será dado o nome ao canal de “*modbus*”.

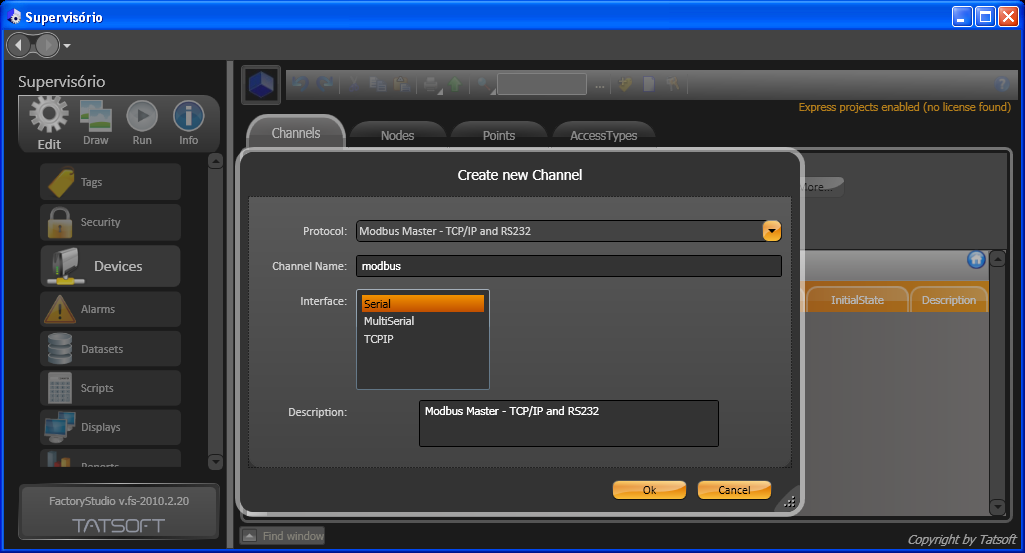


Figura 16 - Tela ajuste do canal de comunicação

**Fonte:** Autoria própria

Quarto passo – Depois de configurado o *driver* de comunicação deverá ajustar os parâmetros da porta serial responsável pela comunicação entre IHM e Arduino. Esses ajustes deverão ser iguais aos valores ajustados na porta de comunicação do Arduino.



Figura 17 - Tela ajuste da porta serial

**Fonte:** Autoria própria

Quinto passo – Agora que temos o *driver* ajustado, deverá ser criado o “Nó”, endereço para comunicação com o controlador que, neste caso, é a placa do Arduino. Para isto deverá acessar no *menu* lateral esquerdo no ícone *Devices*, em seguida a aba *Nodes* no menu superior e preencher os campos *Name, Channel* em nosso caso *modbus*, *Primarystation*, *Description*.

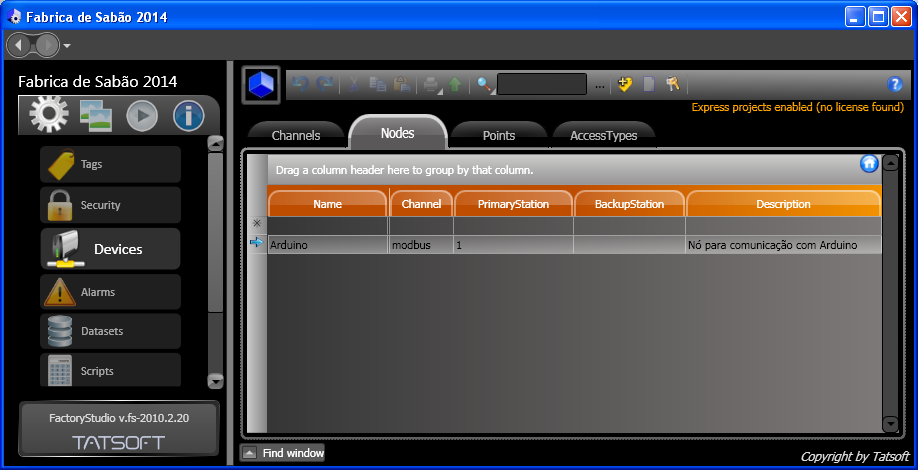


Figura 18 - Tela criação do Nó de comunicação

**Fonte:** Autoria própria

Através das etapas demonstradas, configuramos, ajustamos e criamos os recursos necessários à comunicação entre o controlador e a IHM, agora iniciaremos o desenvolvimento das telas da IHM necessários para realizar os comandos de acionamento e monitoramento das respectivas variáveis do processo na fábrica tais como a indicação de nível, temperatura, tempo de reação e status dos equipamentos.

# 4.1 Criação das TAGS e Configurações de Endereços de Comunicação

Iniciamos realizando levantamento de todos os TAGs necessários para desenvolver uma IHM com as características de comandos desejados. De posse destes dados, foi criado um mapa de memória que descreve os respectivos endereços utilizados pelo *Modbus* na troca de dados. Esses endereços, ou TAGS, podem contar diversos formatos de dados sendo alguns do tipo inteiros, ou agrupamentos de *bits* formando palavras de 16 *bits*. Esta fase é de grande importância, pois cada *bit* nestas palavras, ou *words*, podem informar ou determinar uma condição na lógica do controlador. A seguir são apresentadas as palavras utilizadas com seus respectivos significados dos dados contidos.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Word 40001 - Comandos1 | | | | | |
| **Bit** | **Função** | **Tag Aux** | **Equipamento** | **True** | **False** |
| 0 | Seleciona Aut/Man | SelecAuto | Geral | Auto | Manual |
| 1 | Emergência | Emergenc | Geral | Atuada | Desatuada |
| 2 | Rearme | RearmGer | Geral | Atuada | Desatuada |
| 3 | Liga Supervisório | LigaEqp | Geral | Liga | - |
| 4 | Desliga Supervisório | DeslEqp | Geral | Desliga | - |
| 5 | Seleciona Remoto | SelRemEqp | Geral | Remota | - |
| 6 | Seleciona Local | SelLogEqp | Geral | Local | - |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |
| 10 |  |  |  |  |  |
| 11 |  |  |  |  |  |
| 12 |  |  |  |  |  |
| 13 |  |  |  |  |  |
| 14 |  |  |  |  |  |
| 15 |  |  |  |  |  |

Tabela 1 - Detalhamento da palavra 40001 do mapa de memória

**Fonte:** Autoria própria

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Word 40003 - Status1 | | | | | |
| **Bit** | **Função** | **TAG** | **Equipamento** | **True** | **False** |
| 0 | Selecionado Aut/Man | SelecAuto | Válvula de água | Em Auto | Em Man |
| 1 | Emergencia Atuada | EmergAtuada | Geral | Atuada | - |
| 2 | Selecionado Remoto | SelecRemVAlcool | Válv de álcool | Em Rem | Em Local |
| 3 | Selecionado Local | SelecLocVAlcool | Válv de álcool | Em Local |  |
| 4 | ligado / Desligado | LigaVAlcool | Válv de álcool | Ligada | Desligada |
| 5 | Selecionado Remoto | SelecRemVSoda | Válvula de soda | Em Rem | - |
| 6 | Selecionado Local | SelecLocVSoda | Válvula de soda | Em Local | - |
| 7 | ligado / Desligado | LigaVSoda | Válvula de soda | Ligada | Desligada |
| 8 | Selecionado Remoto | SelecRemVOleo | Válvula de óleo | Em Rem | - |
| 9 | Selecionado Local | SelecLocVOleo | Válvula de óleo | Em Local | - |
| 10 | ligado / Desligado | LigaVOleo | Válvula de óleo | Ligada | Desligada |
| 11 | Selecionado Remoto | SelecRemBomba | Bomba | Em Rem | - |
| 12 | Selecionado Local | SelecLocBomba | Bomba | Em local | - |
| 13 | ligado / Desligado | LigaBomba | Bomba | Ligada | Desligada |
| 14 | Selecionado Remoto | SelecRemAgit | Agitador | Em Rem | - |
| 15 | Selecionado Local | SelecLocAgit | Agitador | Em Local | - |

Tabela 2 - Detalhamento da palavra 40003 do mapa de memória

**Fonte:** Autoria própria

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Word 40004 - Status2 | | | | | |
| **Bit** | **Função** | **TAG** | **Equipamento** | **True** | **False** |
| 0 | Ligado / Desligado | LigaAgit | Agitador | Ligada | Desligada |
| 1 | Selecionado Remoto | SelecRemAquec | Aquecedor | Em Rem | - |
| 2 | Selecionado Local | SelecRemAquec | Aquecedor | Em Local | - |
| 3 | Ligado / Desligado | LigaAquec | Aquecedor | Ligado | Desligado |
| 4 | Selecionado Remoto | SelecRemComp | Compressor | Em Rem | - |
| 5 | Selecionado Local | SelecLocComp | Compressor | Em Local | - |
| 6 | Ligado / Desligado | LigaComp | Compressor | Ligada | Desligada |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |
| 10 |  |  |  |  |  |
| 11 |  |  |  |  |  |
| 12 |  |  |  |  |  |
| 13 |  |  |  |  |  |
| 14 |  |  |  |  |  |
| 15 | Pulso a cada segundo | ON\_OF\_1S | Geral |  |  |

Tabela 3 - Detalhamento da palavra 40004 do mapa de memória

**Fonte:** Autoria própria

|  |  |
| --- | --- |
| Word 40005 – Seleção dos equipamentos | |
| Valor | Significado |
| 1 | Seleciona válvula de álcool |
| 2 | Seleciona válvula de soda |
| 3 | Seleciona válvula de óleo |
| 4 | Seleciona bomba de água |
| 5 | Seleciona agitador |
| 6 | Seleciona aquecedor |
| 7 | Seleciona compressor |
| 8 | Inicia a fabricação |
| 9 | Finaliza a fabricação |

Tabela 4 - Detalhamento da palavra 40005 do mapa de memória

**Fonte:** Autoria própria

|  |  |
| --- | --- |
| Word 40015 - Status da fabricação | |
| Valor | Significado |
| 0 | Parado |
| 1 | Dosando água |
| 2 | Dosando soda |
| 3 | Dosando óleo |
| 4 | Dosando álcool |
| 5 | Agitando |
| 6 | Finalizado a fabricação |

Tabela 5 - Detalhamento da palavra 40015 do mapa de memória

**Fonte:** Autoria própria

Essa técnica de empacotar os *bits* dentro de palavras simplifica e economiza o canal de comunicação do *modbus*, por isto foi utilizado estas *words* contendo em seus *bits* várias informações, todas binárias, ou seja, apenas dois estados.

Todas as outras palavras possuem seus dados do tipo inteiro, não sendo necessário detalhamento de seus *bits*.

Essas palavras são as posições de memória que o *modbus* utiliza para trocar os dados entre o Arduino e a IHM, portanto é necessário realizar a configuração desta área de memória na estrutura de dados do *modbus*. Dessa forma, esses endereços também foram programados no programa do Arduino. Veja a seguir que, com poucas posições de memória e conforme a utilização da técnica de empacotamento de *bits*, foi possível trocar todos os dados necessários para o monitoramento e controle da fábrica de sabão. As posições dos registros dos dados para o *modbus* usado neste programa e programados no Arduino são:

* 40001 - Comandos1 - Vem do *FactoryStudio*;
* 40002 - Comandos2 - Vem do *FactoryStudio*;
* 40003 - Status1 - Vai para o *FactoryStudio*;
* 40004 - Status2 - Vai para o *FactoryStudio*;
* 40005 - SelecEqp - Vem do *FactoryStudio*;
* 40006 - Contagem de tempo – Vai para *FactoryStudio*;
* 40007 - Nível do tanque reator – Vai para *FactoryStudio*;
* 40008 - Temperatura do aquecedor de água – Vai para *FactoryStudio*;
* 40009 - SP Temperatura da água – Vem do *FactoryStudio*;
* 40010 - SP Nível água – Vem do *FactoryStudio*;
* 40011 - SP Nível soda – Vem do *FactoryStudio*;
* 40012 - SP Nível óleo – Vem do *FactoryStudio*;
* 40013 - SP Nível álcool – Vem do *FactoryStudio*;
* 40014 - SP tempo agitação – Vem do *FactoryStudio*;
* 40015 - Status da reação – Vai para *FactoryStudio*;
* 40016 - Status do tempo de agitação – Vai para *FactoryStudio*;

Definidos os TAGs a serem utilizados, foi criado os mesmos no *software* *FactoryStudio* para prosseguir com o desenvolvimento da Interface Homem Máquina. Para criação desses TAGs, foram realizadas as seguintes etapas:

Clica-se no ícone *Edit* na barra de ferramentas, no *menu* superior esquerdo do *FactoryStudio*, em seguida clica-se no ícone Tags na barra de ferramentas vertical do lado esquerdo do programa.

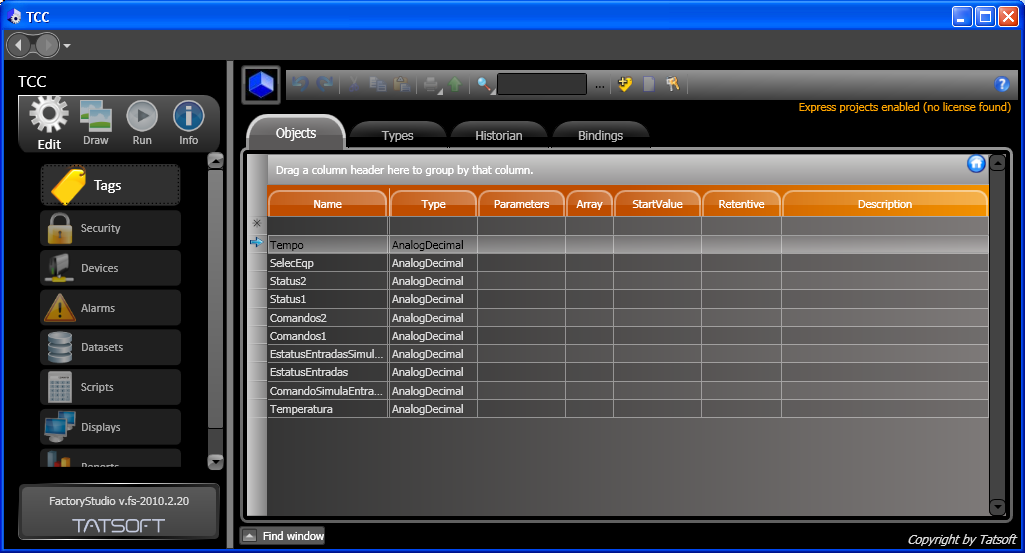


Figura 19 - Tela de Criação de TAGS

**Fonte:** Autoria própria

Com a janela de criação de TAGs aberta, dá-se um duplo *click* sobre a linha abaixo da aba *Name* e insere-se o nome do TAGs. Em seguida efetua-se um duplo *click* na linha abaixo do *Type* e define-se o tipo do TAG. Após essa etapa, deve-se dar um *enter* para salvar a configuração. Essa sequência deverá ser repetida sucessivamente para a criação de todos os demais TAGs envolvidos nesta aplicação.

Observe que até a etapa apresentada, os TAGs foram definidos, porém não possuem endereços de memórias para referenciar o local no controlador Arduino. Esta referência será efetuada acessando o ícone *Devices* em seguida a aba *Points*, ou seja, traduzindo “pontos”, que liga o TAG a um endereço de memória no Arduino Mega 2560. Note que na FIG.15, definimos as características necessárias para o TAG ser de - leitura, escrita ou leitura e escrita - e definimos também o Nó de comunicação, o endereço de memória de referência, o tipo de dado, o tipo de acesso e a escala.



Figura 20 - Tela de Criação de TAGS / Devices

**Fonte:** Autoria própria

# 4.2 Criação das telas de operação

Para a operação da fábrica, é necessário o desenvolvimento de telas com elementos gráficos de forma a representar o processo com todas as informações do atual estado e fornecer todos os recursos para atuar nos dispositivos de controle. Dessa forma, algumas telas foram criadas, algumas específicas para comandos ou entradas de dados e outra geral com todo o processo.

Neste projeto, para a operação, foram criadas três telas:

* Tela geral do processo completo;
* Tela de botoeira;
* Tela de *Set Points*.

O desenvolvimento dessas telas é realizado totalmente dentro do *software*, e o mesmo possui várias ferramentas e componentes para a edição e configuração de seus comportamentos e dinâmicas.



Figura 21 - Componentes para criação de telas na IHM

**Fonte:** Autoria própria

Após desenvolver a tela colocando todos os componentes e desenhos necessários, basta clicar com o botão direito do *mouse* e selecionar o item *“dinamics”* e então proceder com as seleções e ajustes conforme FIG.17:

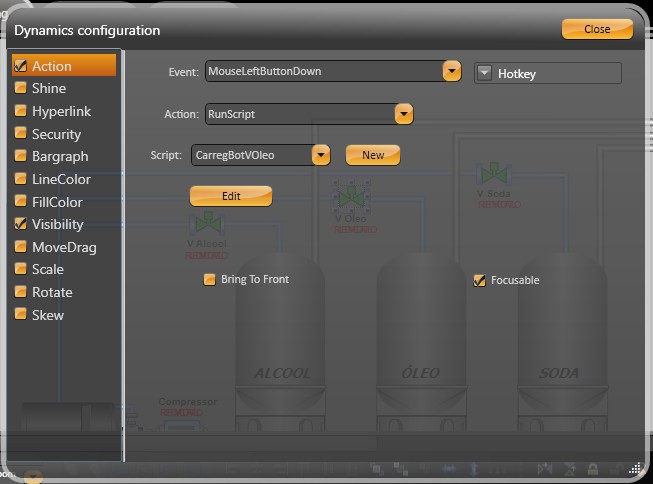


Figura 22 - Configuração dos componentes dinâmicos

**Fonte:** Autoria própria

Todo o desenvolvimento é realizado de forma bastante intuitiva sendo necessário apenas relacionar os componentes visuais com os devidos TAGs e ajustar as suas propriedades ou eventos de forma a atender às necessidades. Em poucos pontos onde se faz necessário executar um número maior de operações em conjunto, pode-se utilizar recursos de programação em *scripts* podendo ser em C# ou *VB.Net*. Para conseguir carregar a tela da botoeira de acordo com o equipamento que se deseja operar, foi necessário escrever um valor no TAG de forma que o programa do Arduino conseguisse identificar para qual equipamento a botoeira iria comandar e ao mesmo tempo carregar a tela da botoeira correspondente, foi usado esse recurso de *script* conforme pode ser observado na FIG.18:

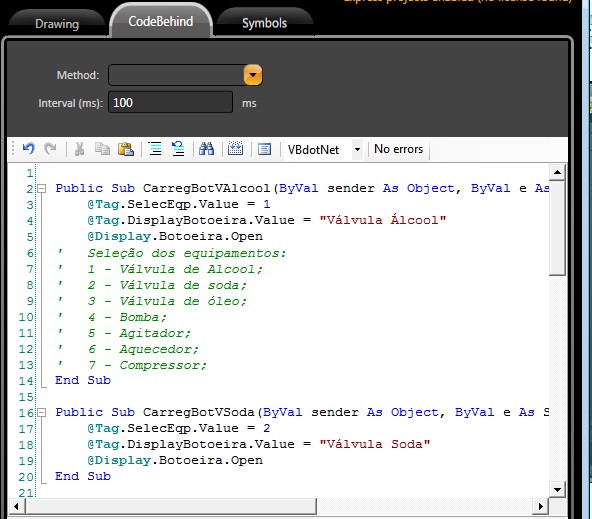


Figura 23 - Tela de entrada de Scripts da IHM

**Fonte:** Autoria própria

# 4.3 Descrição operacional das telas

Na FIG.19 é apresentada a tela principal, onde o operador pode monitorar as variáveis da fábrica em tempo real ou controlar a mesma selecionando um determinado equipamento para realizar o comando desejado. Clicando sobre o equipamento, a janela da botoeira é aberta com os botões para os respectivos comandos. Ainda na tela principal temos, na parte inferior, um *menu* com opções para acionar o botão de emergência, botão de rearme, passar o modo de operação para automático ou manual, iniciar ou finalizar o processo de fabricação. Além de todos esses comandos, é possível alterar a concentração do sabão clicando sobre o botão *set points*, que abre a janela de ajustes do nível de água, de soda, de óleo, de álcool, tempo de agitação, de reação e temperatura da água. Podemos ainda visualizar o estado atual do processo de fabricação no *menu* inferior da tela principal.

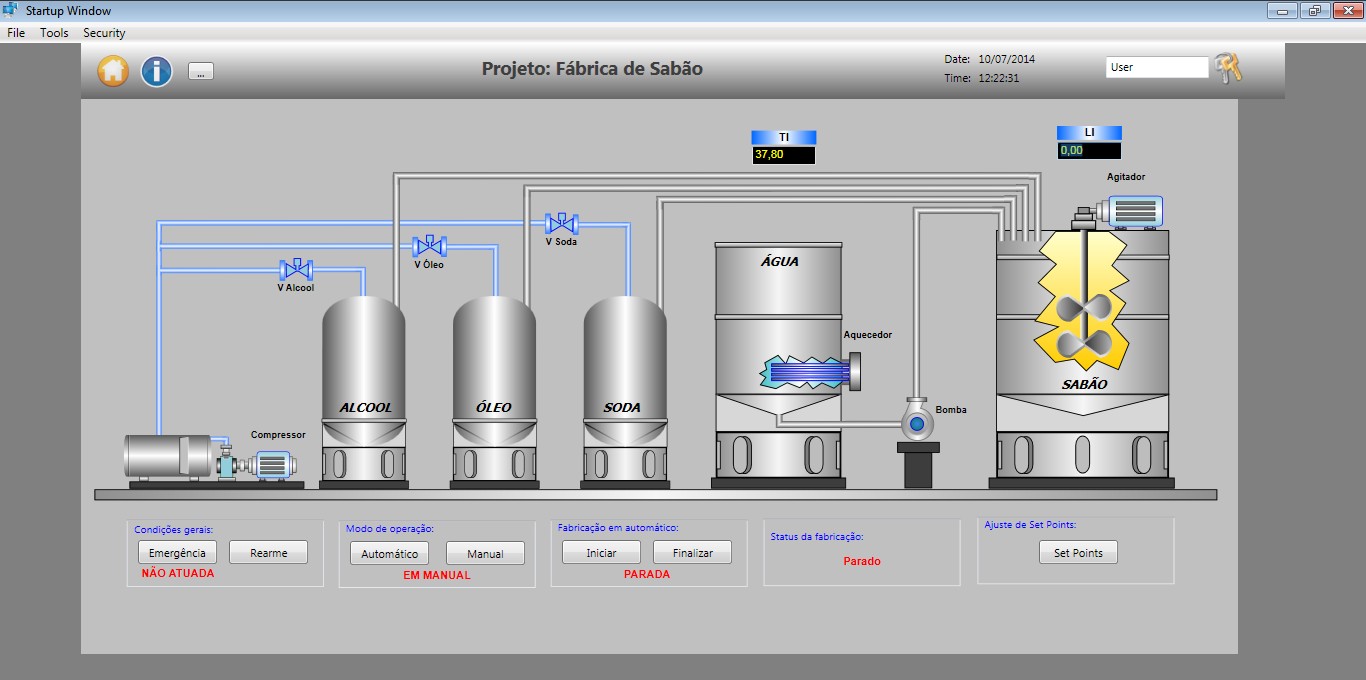


Figura 24 - Tela principal de operação da fábrica

**Fonte:** Autoria própria

Como visto em etapas anteriores, cada TAG do supervisório é relacionado a um endereço de memória do controlador Arduino, o que possibilita a indicação do estado do respectivo equipamento. Nessa tela de operação, a cor azul no equipamento mostra que está liberado para operação, o verde representa que o equipamento está funcionando. Se o equipamento não estiver operando em automático, ou seja, se estiver manual, temos as opções de operar em local ou remoto, sendo a operação local sinalizada pela palavra local ao lado do equipamento, que passam a receber os comandos de acionamento através de chaves conectadas às entradas digitais do controlador Arduino Mega 2560. Para o modo remoto os comandos para o acionamento dos equipamentos são realizados por meio das botoeiras da IHM.

Na FIG.20 é apresentada a tela da botoeira. Através desta tela é possível comandar o equipamento selecionado para ligar, desligar, passar para local, para remoto ou rearmar o mesmo. Depois de realizados os comandos desejados, dever-se á clicar no botão cancela para fechar a botoeira.

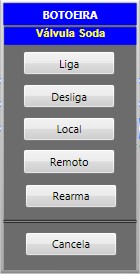


Figura 25 - Tela da botoeira de acionamento dos equipamentos

**Fonte:** Autoria própria

Na FIG.21 é apresentada a tela de ajuste dos *Set Points*. É nesta tela que entramos com os valores dos pontos de ajustes para definir a quantidade de cada reagente, o valor da temperatura da água e o tempo que irá agitar até conseguir total reação. Esses valores são ajustados antes de se iniciar a operação e sua alteração somente será necessária caso ocorrer alteração na formulação do sabão.

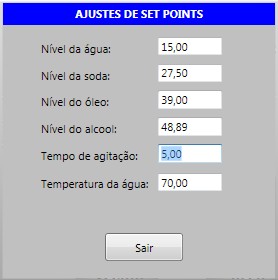


Figura 26 - Tela de entrada de Set Points

**Fonte:** Autoria própria

# CONFIGURAÇÃO DO ARDUINO COM O SISTEMA

Para a comunicação entre controladores e supervisórios industriais, um protocolo bastante utilizado é o *Modbus* devido a sua simplicidade, robustez e recursos disponíveis e principalmente por ser um protocolo aberto. Dessa forma é fácil conseguir grande quantidade de informação e dispositivos utilizando esse protocolo. Diante dessas características, a comunicação serial com o protocolo modbus será utilizada para realizar a comunicação entre o supervisório e o controlador Arduino Mega 2560. O protocolo será apenas o responsável por integrar controlador e IHM. A IHM será interface onde o operador irá visualizar todas as informações e atuará para comandar os equipamentos. Essas informações são enviadas para o controlador, que é o responsável por processar todas as informações da fábrica e retornar uma resposta de acordo com os resultados de suas lógicas. Em sistemas industriais geralmente utilizam-se CLPs para processar as informações do chão de fábrica, isso devido à confiabilidade e robustez desse tipo de controlador. Porém ele é um tipo de controlador que possui um valor bastante elevado para sua aquisição, o que o torna inviável para aplicações como a desse projeto. Diante disso optamos por utilizar o Arduino Mega 2560, que é uma placa de circuito constituída basicamente de um processador, entradas e saídas analógicas e digitais, com opção para comunicação serial.

# 5.1 Arduino Mega 2560

O Arduino Mega 2560 é uma placa única composta por um microcontrolador ATmega 2560, 54 pinos de entradas/saídas digitais configuráveis, 16 entradas analógicas, 4 UARTS (portas seriais de *hardware*), um oscilador de cristal de 16 MHz, uma conexão USB, uma entrada de alimentação, uma conexão ICSP e um botão de *reset*.

O *hardware* e o *software* do Arduino são ambos de fonte aberta, o que significa que

o código, os esquemas, o projeto etc. podem ser utilizados livremente por qualquer pessoa e com qualquer propósito. Dessa forma, há muitas placas-clone e outras placas com base no Arduino disponíveis para compra, ou que podem ser criadas a partir de um diagrama. De fato, nada impede que se comprem os componentes apropriados e se crie seu próprio Arduino em uma matriz de pontos ou em uma *PCB* (*Printed Circuit Board*, placa de circuito impresso) feita em casa. A única ressalva que a equipe do Arduino impõe e que você não utilize a palavra “Arduino”. Esse nome é reservado à placa oficial. Daí a existência de nomes para as placas-clone como Freeduino, Roboduino etc.( MCROBERTS, 2011). Na FIG.22 é apresentado a placa do Arduino que está sendo utilizada neste projeto.

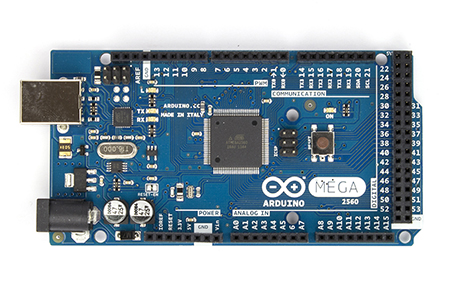


Figura 27 - Arduino UNO

Fonte: www.arduino.cc

Para a correta operação desse Arduino, é necessário conhecer as suas principais características que são apresentadas a seguir:

|  |  |
| --- | --- |
| Características do Arduino Mega2560  Microcontrolador | ATmega2560 |
| Voltagem de alimentação | 5V |
| Voltagem de entrada recomendada | 7-12V |
| Voltagem de entrada limites | 6-20V |
| Pinos digitais I/O | 54(dos quais 14 podem ser saídas PWM) |
| Pinos de entradas analógicas | 16 |
| Corrente contínua por pino I/O | 40ma |
| Corrente contínua para o pino 3.3V | 50ma |
| Memória *Flash* | 256 KB *of which* 4 KB *used by bootloader* |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Velocidade de *Clock* | 16 MHz |

Tabela 6 - Dados do Arduino Mega 2560

**Fonte:** Adaptado de www.arduino.cc

Para se realizar a programação da aplicação do usuário no Arduino Mega 2560 deve-se possuir um *software* capaz de escrever uma linguagem de programação que o Arduino compreenda. Neste projeto foi utilizado o IDE do Arduino que vem do inglês *Integrated Development Environment*, ou Ambiente de Desenvolvimento Integrado do Arduino. Esse *software* é livre e seu *download* pode ser feito no site [www.arduino.cc](http://www.arduino.cc). O programa do IDE não necessita ser instalado, basta apenas executá-lo e a partir daí, desenvolver o seu código fonte, de acordo com suas necessidades, ou seja, criar um programa de computador, que nada mais é do que um conjunto de instruções passo a passo dos quais é compilado no IDE e realizado *upload* para o Arduino. O IDE traz ainda uma série de exemplos de códigos fontes para diversas aplicações. Além dos exemplos, existem várias bibliotecas que podem ser úteis para facilitar o desenvolvimento dos projetos. Caso seja necessário, podem-se baixar bibliotecas desenvolvidas por outros usuários e importar para o IDE. Na FIG.23 é apresentada a janela que se abre ao executar o IDE. Nessa IDE existem todos os recursos necessários para se desenvolver o programa, compilar e realizar o *upload* na memória do Arduino. A seguir serão detalhadas as bibliotecas que foram incorporadas nesta aplicação. No Apêndice A, apresentamos o código fonte do nosso programa com todas as bibliotecas utilizadas.

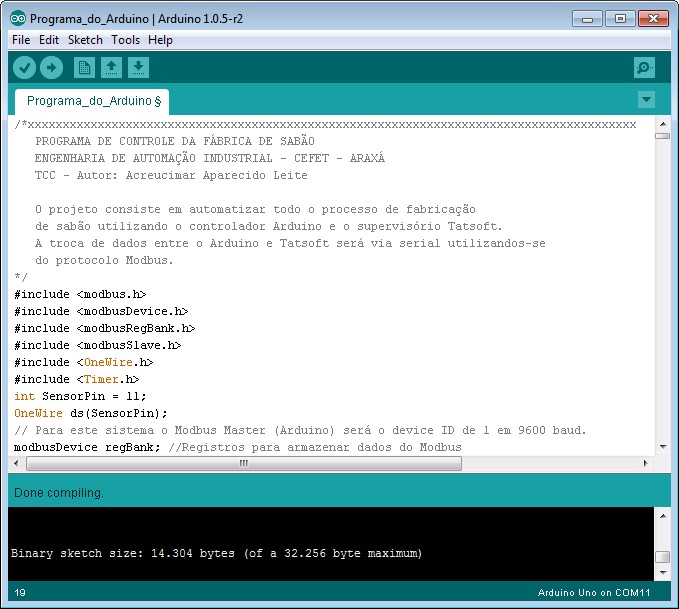
****

Figura 28 - Ambiente de desenvolvimento integrado do Arduino

**Fonte:** Autoria própria

# 5.2 Biblioteca *Modbus Slave*

Neste trabalho, o Arduino é o responsável pelo tratamento das informações recebidas do campo através de sensores instalados em suas entradas. Essas informações são processadas e devem ser enviadas ao supervisório que, por sua vez,deve enviar informações para o Arduino. Conforme já citado anteriormente, para realizar essa troca de informações, é utilizado o protocolo de comunicação *modbus*, devido ao fato de essa placa de Arduino já possuir incorporados canais de comunicação serial. Mas, para que essa comunicação ocorra é necessário informar ao Arduino através de programação todos os passos para entendimento do envio e recebimento de dados com a utilização desse protocolo. Para agilizar a programação, foi realizado o *download* da biblioteca *modbus slave*, que foi desenvolvida por Juan Pablo Zometa, estudante PhD do instituto de Engenharia de Automação e assistente de pesquisa na Universidade de Magdeburga, Alemanha. Essa biblioteca é a responsável por realizar funções baseadas no protocolo *Modbus* tornando, assim, o Arduino um escravo e o supervisório, o mestre.

# 5.3 Biblioteca *OneWire*

Para realizar a leitura da temperatura, utilizamos a biblioteca *OneWire*, baixada através do *link* disponível no site: www.labdegaragem.com. Após baixar o arquivo, extraímos a biblioteca *OneWire* para dentro da pasta “*libraries*” localizada dentro da pasta Arduino. Excutamos o IDE Arduino e adaptamos o código para nosso projeto. O programa foi adaptado de forma a obtermos a temperatura em graus celsos. Um sensor de temperatura modelo DS18B20 foi conectado a uma entrada digital por utilizar o protocolo digital de apenas um fio. Realizado o *upload* do código para o Arduino, o mesmo verifica se o sensor está corretamente ligado. Verificada a conexão do sensor, o Arduino inicia a aquisição de dados através de um endereço para leitura. Concluída a leitura, o Arduino trata essas informações e as envia, através da rede *modbus* para o supervisório. Antes de realizar o *upload* é necessário configurar o Arduino e sua respectiva porta de comunicação selecionando na IDE o modelo da placa Arduino utilizada. Em nosso caso, o Arduino Mega 2560 é a porta serial a ser utilizada. Para realizar o teste de comunicação do sensor de temperatura com o Arduino, abrimos o Serial Monitor e configuramos sua comunicação, tornando possível a visualização da indicação de temperatura conforme mostra a FIG.24:

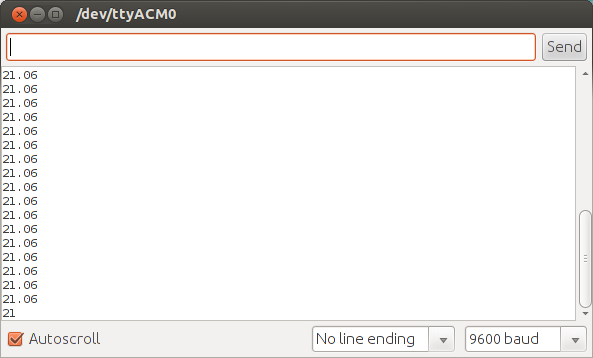


Figura 29 - Serial monitor com indicação da temperatura

**Fonte:** Autoria própria

Com a utilização dessas duas bibliotecas e bibliotecas nativas do próprio IDE foi desenvolvido todo o código fonte necessário para monitorar e controlar a fábrica de sabão. Todo o código fonte da aplicação é apresentado no apêndice 01.

# MONTAGEM

A montagem eletroeletrônica foi realizada em oficina própria, sendo confeccionado um painel de força e comando composto pela placa Arduino Mega 2560, módulo de 8 relés de 12 *volts*, contatores, fonte de 12 *volts*, retificador de onda completa, régua de bornes e acessórios como fios, canaletas etc.. Para a comunicação com o Arduino, foi utilizado um cabo USB para conectar a placa do Arduino com o *notebook*, não sendo necessários outros acessórios para esta conexão. Para realizar a montagem eletroeletrônica, além dos itens descritos anteriormente, foi necessário pesquisar as características do módulo de relés, do qual será feita uma breve descrição a seguir:

# 6.1 Módulo de 8 relés de 12 volts

Um relé é basicamente uma chave eletromecânica e quando aplicada uma tensão em sua bobina circula uma baixa corrente que faz com que os contatos desta chave se desloquem, fechando ou abrindo um determinado circuito elétrico. No caso desse trabalho, o controlador não possui potência para acionar as válvulas solenoides, o aquecedor e o motor da bomba, portanto foi utilizado como interface um módulo de 8 relés de 5 volts apresentado na FIG.25. Quando o controlador enviar um comando de acionamento, uma tensão de 5 volts é aplicada no respectivo relé, fazendo com que a chave eletromecânica seja acionada fechando o circuito e ligando o dispositivo. O módulo de relé é facilmente integrado ao controlador através das saídas digitais, permitindo acionar cargas maiores como motores, solenóides, resistência de aquecimento etc.. Cada saída do módulo de relé possui um *led* de indicação do seu estado, o que facilita a visualização do seu funcionamento e o diagnóstico de falhas de acionamento.

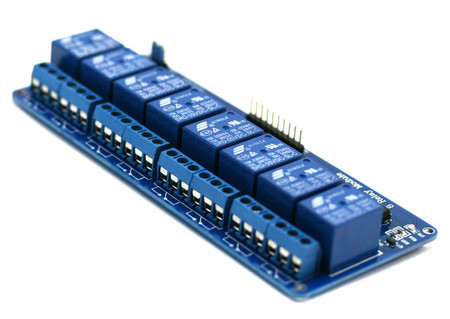


Figura 30 - Módulo de relés de 8 canais

**Fonte:** <http://www.filipeflop.com/pd-6b851-modulo-rele-5v-8-canais.html>

O módulo de relé utilizado possui as seguintes características:

* Tensão de operação: 5VDC (VCC e GND);
* Tensão de sinal: TTL 5VDC(IN1 e IN2);
* Corrente típica de operação: 15 – 20 mA;
* Cada relé possui três terminais proporcionando 1 contato NF, 1 NA e o comum;
* Contato do relé permite tensão de até 30 Vdc a 10A ou 250 Vac a 10ª;
* Tempo de resposta 5 – 10 ms;
* Indicador LED de funcionamento;
* Proteção por optoacopladores contra surtos de até 5 mil volts;
* Conectores parafusáveis para garantir uma conexão segura das cargas;
* 4 furos de 3mm para fixação nas extremidades da placa;
* Dimensões 135 mm X 52 mm X20 mm.

De acordo com Zheng Yi Ming (2014) o modo tradicional de acionamento de relés opera um pino digital do microcontrolador como saída, selecionando nível lógico alto ou baixo para acionamento do relé. Esse método usualmente requer que a tensão de saída em nível lógico alto seja a mesma utilizada no sinal de controle dos relés. O sistema implementado alterna o pino digital entre estado de saída com nível lógico baixo e estado de entrada, em alta impedância. O estado de alta impedância simula o nível lógico alto ao cortar a corrente do circuito, com a vantagem de passar a ser independente da tensão.



Figura 31 - Circuito de um canal do módulo de relés

Fonte: Manual Fabricante

De posse de todas as características dimensionais dos componentes do painel, foi elaborado uma configuração (*layout*) da distribuição dos componentes no “espelho” do painel. Fixados os respectivos componentes de acordo com o *layout* elaborado. Após realização da fixação dos mesmos e dos acessórios, foram iniciadas as interligações entre controlador, modulo de relé, fonte de alimentação, régua de borne e fusíveis. Concluída essa etapa, foi elaborado o *layout* da porta do painel com as respectivas chaves de comandos para ligar e desligar os equipamentos em modo Local de operação. Prosseguido as etapas de montagem, foram fixadas as respectivas chaves e as bases dos fusíveis de entrada. Concluída a etapa de fixação, foram realizadas as interligações das chaves com o controlador Arduino Mega 2560. No APÊNDICE C apresentamos fotos/projeto do painel.

Para elaborar a montagem eletroeletrônica deste trabalho, foram levados em consideração os conceitos de circuito de manobra, dividindo em circuito de comando e circuito de força ou potência, de maneira a proporcionar em primeiro lugar a segurança do usuário e em segundo lugar, a disponibilidade de recursos para a completa automação do sistema elétrico. Em nosso projeto o circuito de comando elétrico, restringe-se ao painel de comando, já o circuito de potência ou força abrange desde a fonte até o acionamento da força motora, que em nosso caso é o motor da bomba de água, o motor do agitador, o compressor, o aquecedor e as bobinas das solenoides. De posse desses conceitos e finalizada a montagem do quadro de força e comando, iniciamos a interligação deste circuito de força com a carga instalada no campo, finalizando a montagem dos circuitos responsáveis por acionar as cargas de acordo com os respectivos comandos, sejam em modo local ou remoto.

# RESULTADOS E CONCLUSÃO

Finalizado a montagem eletroeletrônica de todo o sistema, atingimos pleno objetivo no que se refere à comunicação do controlador Arduino Mega 2560, com o supervisório *FactoryStudio* instalado no *notebook*. Tanto os sinais dos sensores analógicos, que são responsáveis pelas leituras do nível do reator e da temperatura do tanque de água, assim como as entradas digitais responsáveis pelos sinais de comando liga/desliga e emergência dos equipamentos, foram transmitidos com exatidão do controlador para o supervisório instalado no *notebook*. Concluído os testes de transmissão de dados do controlador Arduino Mega 2560 para o supervisório, iniciamos os testes de transmissão de dados do supervisório para o controlador, que apresentou um excelente resultado, pois todos os dados enviados do supervisório foram recebidos com exatidão no controlador. Realizado os testes de comunicação, inserimos no supervisório um valor de *Set-Point* de 70ºC, e aguardamos a temperatura da água atingir o valor desejado, de modo a verificar o funcionamento da malha de controle de temperatura, que apresentou um resultado extremamente satisfatório no que diz respeito à malha de controle de temperatura. Posteriormente, ajustamos os valores de *Set-Point* da água, soda, óleo e álcool, e aguardamos as dosagens de cada reagente de forma a observar o funcionamento da malha de controle de nível. Com o êxito no funcionamento das malhas de controle, passamos todos os equipamentos para modo de operação remoto, ajustamos os *Set-Points* de acordo com os valores desejados para o processo e iniciamos os testes de operação 100% automática, onde aguardamos a temperatura da água atingir o valor ajustada, para posteriormente iniciar a dosagem dos reagentes no modo automático. Todas as etapas do processo foram concluídas de forma satisfatória, uma vez que atingido a temperatura a resistência de aquecimento foi desligado e as dosagens realizadas na quantidade e na sequencia programada, ligando em seguida e também de forma automática o agitador, responsável pela mistura da batelada, finalizando o processo com o desligamento do agitador e apresentando um alarme visual e sonoro no supervisório de batelada concluída.

# 7.1 CONCLUSÃO

Podemos concluir que o presente trabalho apresentou resultados satisfatórios, pois atingimos todos os objetivos propostos. A parte referente à comunicação entre o controlador Arduino Mega 2560 e o supervisório instalado no notebook, ocorreu sem apresentar problemas de atraso na transmissão ou na recepção de dados, assim como erros na leitura ou escrita no controlador ou supervisório que pudessem leva-los a um possível travamento. As malhas de controles de temperatura e nível responderam de forma satisfatória, sendo o controle compatível com controladores comercializados para as respectivas finalidades, não observamos nenhuma variação de controle que pudesse prejudicar o desempenho das respectivas malhas. O comando tanto do supervisório como local, funcionou perfeitamente, dando liberdade para o operador atuar no processo remotamente de forma manual ou automática, ou mesmo intervir no processo localmente atuando no comando através das chaves locais instaladas no painel de comando. Na primeira batelada realizado o produto final não apresentou as características esperadas, mas com pequenos ajustes nos valores dos *Set-Points*, encontramos sem grandes dificuldades os valores necessários para produzir um sabão com as características desejadas de viscosidade e acidez.

Os equipamentos e componentes que compõem o sistema automatizado são de suma importância para o bom funcionamento e operacionalização da fabrica de sabão, pois adicionam recursos que em conjunto com desenvolvimento de lógicas em controladores como Arduino Mega 2560 permite o controle remoto.

Consolidando os conhecimentos adquiridos sobre o funcionamento da fábrica de sabão, os conceitos e a implementação da automação para acionamento, controle e monitoração, possibilitaram o desenvolvimento e pesquisas do nosso projeto. Abordando a automação que vem de encontro com a maioria das necessidades de organizações. A cada dia o controle de qualidade e o aumento na produção fazem com que os empresários busquem soluções tecnológicas avançadas capazes de viabilizar e modificar a economia de uma região. Considerando a aplicabilidade da fábrica de sabão, milhares de litros de óleo usados deixarão de ser descartados na natureza, e passarão a gerar renda, tornando claro, a redução de riscos de impactos e acidentes ambientais.

Podemos concluir ainda que este sistema pode ser adaptado para as mais diversas aplicações, proporcionando maior eficiência de controle e desempenho do respectivo processo, obtendo também ganho de espaço físico, pois trata-se de um sistema extremamente compacto se comparado com sistemas convencionais. De modo geral concluímos que o presente trabalho apresentou resultados válidos no que se refere à qualidade do produto final, segurança do usuário e sustentabilidade, uma vez que implantado um sistema como este, podemos evitar que muito óleo utilizado venha ser descartado de forma imprópria, degradando o meio ambiente e comprometendo as gerações futuras.

1. **REFERÊNCIAS BIBLIOGRÁFICAS**

ALBERCI, R.M. & PONTES, F.F.F.. ***Reciclagem de Óleo Comestível Usado Através da Fabricação de Sabão.***Engenharia Ambiental, Espírito Santo do Pinhal, 2004.

ALMEIDA, C. M. ***Biocombustível: uma análise econômica para a região metropolitana de Salvador. 2002. 69f****.* Monografia (Trabalho de Conclusão da Especialização em Ciências Econômicas) – Universidade Estadual de Santa Cruz, Ilhéus, Bahia, 2002.

BEGA, E. A. et al. ***Instrumentação Industrial****.* 2. ed. Rio de Janeiro: Interciência, 2006.

BIODIESEL. **Reciclagem de óleo de cozinha**. Disponível em: <www.biodieselbr.com>. Acessado em março de 2013. BRASIL. Resolução nº 275, de 25 de abril de 2001 do Conselho Nacional do Meio Ambiente (CONAMA).

CARVALHO, F. B. D. ***Estimação de ganhos de Ganhos Financeiros em Projetos de Automação e Controle*** *– Uma Proposta Metodológica e Estudos de Caso.* UFMG: Belo Horizonte, 2010.

MAIMON, D. ***Passaporte verde: Gestão Ambiental e Competitividade****.* Rio de Janeiro: Qualimark, 1996.

MCMURRY, *John.* ***Química Orgânica***, V.3. São Paulo. Pioneira Thomson Learning, 2006.

NOGUEIRA, G. R.; BEBER, J. ***Proposta de metodologia para o gerenciamento de óleo vegetal residual oriundo de frituras.* 2009**. Tese de Mestrado em Bioenergia – Universidade Estadual do Centro-Oeste do Paraná, Irati. 2009.

OGATA, K. ***Modern Control Engineering, Englewood Cliffs****, New Jersy: Prentice-Hall*, *Inc.,* 1967.

OLIVEIRA, A.M.C. ***A Química no Ensino Médio e a Contextualização: A fabricação do sabão como tema gerador de ensino aprendizagem. 2005. 119 f****. Dissertação (Mestrado em Ciências da Natureza, Matemática e suas Tecnologias) –* Departamento de Química, Universidade Federal do Rio Grande do Norte, Natal, 2005.

PERUZZO, T. M.; CANTO, E. L. D. ***Química na Abordagem do Cotidiano****.* 1a. ed. São Paulo: MODERNA, 2002.

SAMPAIO, L. A. G.***Reaproveitamento de óleos e gorduras residuais de frituras: tratamento, matéria-prima para produção de biodiesel. 2003. 59f*.** Dissertação (Mestrado em Desenvolvimento Regional e Meio Ambiente) – Universidade Estadual de Santa Cruz, Ilhéus, 2003.

SUPERQUÍMICA, C. E. T. ***Ficha de Informações de Segurança de Produto Químico*** - FISPQ. Canoas/RS: Superquímica. 2011

VALLE, et al. ***Família e Economia Doméstica: Reflexões, Perpectivas e Desafios*** - XX Congresso Brasileiro de Economia Doméstica, Fortaleza, 2009. Disponível em: http://www.xxcbed.ufc.br/arqs/gt8/gt8\_03.pdf. Acesso em: 05 set. 2014.

WANDAS, C.M.; SCCARTON, l.; MACAHDO, R. B. ***Análise dos custos do sabão caseiro x industrializado.*** *Projeto da Universidade Regional Noroeste do Estado do Rio Grande do Sul.* Departamento de Física, Estatística e Matemática – DeFEM. 2004.

ZANIN, W.S.M. et al. ***Desenvolvimento de Sabão Base Transparente.*** PARANÁ: RGSA, 2001.

ZHENG, Y. M.***Painel eletrônico de comando com interface sensível a toque para embarcações marítimas.*** Projeto de Graduação apresentado ao Curso de Engenharia Eletrônica e de Computação– UFRJ/Escola Politécnica, Rio de Janeiro, 2014.

**APÊNDICE A – PROGRAMAÇÃO DO ARDUINO**

/\*xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

PROGRAMA DE CONTROLE DA FÁBRICA DE SABÃO

ENGENHARIA DE AUTOMAÇÃO INDUSTRIAL - CEFET - ARAXÁ

TCC - Autor: Acreucimar Aparecido Leite

O projeto consiste em automatizar todo o processo de fabricação

de sabão utilizando o controlador Arduino e o supervisório Tatsoft.

A troca de dados entre o Arduino e Tatsoft será via serial utilizandos-se

do protocolo Modbus.

\*/

#include <modbus.h>

#include <modbusDevice.h>

#include <modbusRegBank.h>

#include <modbusSlave.h>

#include <OneWire.h>

#include <Timer.h>

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx Medição da temperatura xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

int SensorPin = 23; //Entrada do sensor de temperatura.

OneWire ds(SensorPin);

/\*

Para este sistema o Modbus Master (Arduino) será o device ID de 1 em 9600 baud.

\*/

modbusDevice regBank; //Registros para armazenar dados do Modbus

modbusSlave slave; //Cria o ModbusSlave

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx Declaração de Variávei xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Entradas Digitais xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

const int BL\_Compressor = 37; //Chave 01

const int BL\_Agitador = 39; //Chave 02

const int BL\_Agua = 41; //Chave 03

const int BL\_Soda = 43; //Chave 04

const int BL\_Oleo = 45; //Chave 05

const int BL\_Alcooll = 47; //Chave 06

const int BL\_Aquecedor = 49; //Chave 07

const int BL\_Manual = 51; //Chave 08

const int BL\_Emergencia = 35; // Chave de Emergencia

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxx Entradas analógicas xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

int PinEntAnalog = 15; // selecione o pino de entrada analógica

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Saidas Digitais xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

const int Saida\_Liga\_Compressor = 34; //Relé 01

const int Saida\_Liga\_Soda = 36; //Relé 02

const int Saida\_Liga\_Oleo = 38; //Relé 03

const int Saida\_Liga\_Alcool = 40; //Relé 04

const int Saida\_Liga\_Agitador = 42; //Relé 05

//const int Saida\_Liga\_Reserva = 44; //Relé 06

const int Saida\_Liga\_Agua = 46; //Relé 07

const int Saida\_Liga\_Aquecedor = 48; //Relé 08

const int pin = 13; //Saida com led para indicar pulso segundo

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Variáveis auxiliares xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

boolean LigEqp = false;

// Liga equipamentos

boolean DeslEqp = false;

// Desliga equipamentos

boolean SelRemEqp = false;

// Seleciona remoto equipamentos

boolean SelLocEqp = false;

// Seleciona local equipamentos

boolean InicFabric = false;

// Inicia / Finaliza a fabricação

boolean SelecVAlcool = false;

// Selecionada válvula de álcool

boolean SelecVSoda = false;

// Selecionada válvula de soda

boolean SelecVOleo = false;

// Selecionada válvula de oleo

boolean SelecBomba = false;

// Selecionada a bomba

boolean SelecAgit = false;

// Selecionado o agitador

boolean SelecAquec = false;

// Selecionado o aquecedor

boolean SelecComp = false;

// Selecionado o compressor

boolean SelecAuto = false;

//Seleciona para automático

boolean Emergenc = false;

//Acionada a chave de emergência do Tatsoft

boolean RearmGer = false;

//Acionada o reame geral

boolean SelecRemVAlcool= false;

// Seleciona em Remoto a Valvula de alcool

boolean SelecLocVAlcool= false;

// Seleciona em Local a Valvula de alcool

boolean SelecRemVSoda= false;

// Seleciona em Remoto a Valvula de soda

boolean SelecLocVSoda= false;

// Seleciona em Local a Valvula de soda

boolean SelecRemVOleo= false;

// Seleciona em Remoto a Valvula de oleo

boolean SelecLocVOleo= false;

// Seleciona em Local a Valvula de oleo

boolean SelecRemBomba= false;

// Seleciona em Remoto a bomba

boolean SelecLocBomba= false;

// Seleciona em Remoto a bomba

boolean SelecRemAgit= false;

// Seleciona em Remoto o agitador

boolean SelecLocAgit= false;

// Seleciona em Remoto o agitador

boolean SelecRemAquec= false;

// Seleciona em Remoto o aquecedor

boolean SelecLocAquec= false;

// Seleciona em Remoto o aquecedor

boolean SelecRemComp= false;

// Seleciona em Remoto o compressor

boolean SelecLocComp= false;

// Seleciona em Remoto o compressor

boolean LigaAlcoolAuto = false;

// Liga a Valv alcool em automático

boolean LigaSodaAuto = false;

// Liga a Valv Soda em automático

boolean LigaOleoAuto = false;

// Liga a Valv óleo em automático

boolean LigaBombaAuto = false;

// Liga a bomba em automático

boolean LigaAgitAuto = false;

// Liga o agitador em automático

boolean LigaAquecAuto = false;

// Liga o aquecedor em automático

boolean LigaCompAuto = false;

// Liga o compressor em automático

boolean EmergAtuada = false;

// Emergência local acionada

unsigned int SaidaAlcool=LOW;

//Saída para Valv alcool

unsigned int SaidaSoda=LOW;

//Saída para Valv soda

unsigned int SaidaOleo=LOW;

//Saída para Valv óleo

unsigned int SaidaBomba=LOW;

//Saída para a bomba

unsigned int SaidaAgit=LOW;

//Saída para o agitador

unsigned int SaidaAquec=LOW;

//Saída para o aquecedor

unsigned int SaidaComp=LOW;

//Saída para o compressor

const unsigned long PERIOD1 = 1000;

//Ajusta período de 1000 ms para timer

unsigned int Segundos = 0;

//Contagem de segundo

unsigned int TempAgit = 0;

//Tempo do agitador ligado

boolean Pulso = false;

//Pulso de segundo em segundo

Timer t; //Temporizador de segundos

//long previousMillis = 0;

// previousMillis irá armazenar o ultimo número da função millis()

//long interval = 1000;

// intervalo em que o motor irá esperar

float temp = 0;

int temp1=0;

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Words auxiliares xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

unsigned int EstadoEntradas=0;

//Agrupa estado das entradas para o Tatsoft

//unsigned int EntSimuladas=0;

//Recebe simulação do Tatsoft

unsigned int Status1=0;

//Status gerais para o Tatsoft

unsigned int Status2=0;

//Status gerais para o Tatsoft

unsigned int StatusFabric=0;

//Status da fabricação para o Tatsoft

unsigned int ComandoSimulaEntradas=0;

unsigned int Comandos1=0;

//Comandos do Tatsoft

unsigned int Comandos2=0;

//Comandos do Tatsoft

unsigned int SelecEqp=0;

//Seleção de equipamentos pelo Tatsoft

unsigned int NivelTQ=0;

// Deverá ser desativado a leitura do Modbus qdo finalizado a programação

unsigned int TempAquec=0;

// Deverá ser desativado a leitura do Modbus qdo finalizado a programação

unsigned int SPTemperat=0;

//SP da temperatura da água

unsigned int SPNivAgua=0;

//SP do nível da água

unsigned int SPNivSoda=0;

//SP de nível de soda

unsigned int SPNivOleo=0;

//SP de nível de óleo

unsigned int SPNivAlcool=0;

//SP de nível de alcool

unsigned int SPTempAgit=0;

//SP de tempo de agitação

int ValEntradaAnalog = 0;

// Variável para receber o valor da entrada analógica

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx Realiza as configurações xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

void setup()

{

//Configura os pinos de entradas

pinMode(BL\_Manual, INPUT\_PULLUP);

pinMode(BL\_Compressor, INPUT\_PULLUP);

pinMode(BL\_Agitador, INPUT\_PULLUP);

pinMode(BL\_Agua, INPUT\_PULLUP);

pinMode(BL\_Soda, INPUT\_PULLUP);

pinMode(BL\_Oleo, INPUT\_PULLUP);

pinMode(BL\_Alcooll, INPUT\_PULLUP);

pinMode(BL\_Aquecedor, INPUT\_PULLUP);

pinMode(BL\_Emergencia, INPUT\_PULLUP);

//Configura os pinos de saídas

pinMode(Saida\_Liga\_Compressor, OUTPUT);

pinMode(Saida\_Liga\_Soda, OUTPUT);

pinMode(Saida\_Liga\_Oleo, OUTPUT);

pinMode(Saida\_Liga\_Alcool, OUTPUT);

pinMode(Saida\_Liga\_Agitador, OUTPUT);

pinMode(Saida\_Liga\_Aquecedor, OUTPUT);

pinMode(Saida\_Liga\_Agua, OUTPUT);

pinMode(pin, OUTPUT); // Coloca a saída 13 (com led) piscando a cada segundo

t.every(PERIOD1, Conta\_1s); // A cada 1000 ms

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Configuração do Modbus xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//Foi utilizado o mesmo modelo apresentado na biblioteca do Modbus

regBank.setId(1); //Assign the modbus device ID.

/\*

modbus registers follow the following format

00001-09999 Digital Outputs, A master device can read and write to these registers

10001-19999 Digital Inputs, A master device can only read the values from these registers

30001-39999 Analog Inputs, A master device can only read the values from these registers

40001-49999 Analog Outputs, A master device can read and write to these registers

Analog values are 16 bit unsigned words stored with a range of 0-32767

Digital values are stored as bytes, a zero value is OFF and any nonzer value is ON

It is best to configure registers of like type into contiguous blocks. this

allows for more efficient register lookup and and reduces the number of messages

required by the master to retrieve the data

\*/

//Add Digital Output registers 00001-00016 to the register bank

regBank.add(1);

regBank.add(2);

regBank.add(3);

regBank.add(4);

regBank.add(5);

regBank.add(6);

regBank.add(7);

regBank.add(8);

regBank.add(9);

regBank.add(10);

regBank.add(11);

regBank.add(12);

regBank.add(13);

regBank.add(14);

regBank.add(15);

regBank.add(16);

//Add Digital Input registers 10001-10008 to the register bank

regBank.add(10001);

regBank.add(10002);

regBank.add(10003);

regBank.add(10004);

regBank.add(10005);

regBank.add(10006);

regBank.add(10007);

regBank.add(10008);

//Add Analog Input registers 30001-10010 to the register bank

regBank.add(30001);

regBank.add(30002);

regBank.add(30003);

regBank.add(30004);

// regBank.add(30005);

// regBank.add(30006);

// regBank.add(30007);

// regBank.add(30008);

// regBank.add(30009);

// regBank.add(30010);

//Add Analog Output registers 40001-40020 to the register bank

regBank.add(40001);

regBank.add(40002);

regBank.add(40003);

regBank.add(40004);

regBank.add(40005);

regBank.add(40006);

regBank.add(40007);

regBank.add(40008);

regBank.add(40009);

regBank.add(40010);

regBank.add(40011);

regBank.add(40012);

regBank.add(40013);

regBank.add(40014);

regBank.add(40015);

regBank.add(40016);

regBank.add(40017);

regBank.add(40018);

regBank.add(40019);

regBank.add(40020);

regBank.add(40021);

regBank.add(40022);

regBank.add(40023);

regBank.add(40024);

regBank.add(40025);

/\*

Assign the modbus device object to the protocol handler

This is where the protocol handler will look to read and write

register data. Currently, a modbus slave protocol handler may

only have one device assigned to it.

\*/

slave.\_device = &regBank;

slave.setBaud(9600);

// Initialize the serial port for coms at 9600 baud

}

void loop(){

/\*

Registros utilizados para troca de dados entre o Arduino e o Tatsofte:

40001 - Comandos1 - Vem do Tatsoft

40002 - Comandos2 - Vem do Tatsoft

40003 - Status1 - Vai para o Tatsoft

40004 - Status2 - Vai para o Tatsoft

40005 - SelecEqp - Vem do Tatsoft

40006 - Contagem de tempo (60seg)

40007 - Nivel do tanque reator

40008 - Temperatura do aquecedor de água

40009 - SP Temperatura da água

40010 - SP Nivel água

40011 - SP Nivel soda

40012 - SP Nivel óleo

40013 - SP Nivel álcool

40014 - SP tempo agitação

40015 - Status da reação

40016 - Status do tempo de agitação

Valor e segnificado nesta word:

0 = Parado

1 = Dosando água

2 = Dosando Soda

3 = Dosando óleo

4 = Dosando álcool

5 = Agitando

6 = Finalizando reação

7 = Finalizado fabricação

40020 - Recebe comandos simulando entradas

40021 - Retorna status das entradas simuladas

40022 - Retorna status das entradas

\*/

ComandoSimulaEntradas=regBank.get(40020);

//Recebe entradas a serem simuladas

Comandos1=regBank.get(40001);

//Recebe os comandos do Tatsoft

Comandos2=regBank.get(40002);

//Recebe os comandos do Tatsoft

SelecEqp=regBank.get(40005);

//Recebe a seleção de equipamentos do Tatsoft

SPTemperat=regBank.get(40009);

//Recebe o SP da temperatura do Tatsoft

SPNivAgua=regBank.get(40010);

//Recebe o SP da água do Tatsfot

SPNivSoda=regBank.get(40011);

//Recebe o SP da soda do Tatsoft

SPNivOleo=regBank.get(40017);

//Recebe o SP do óleo do Tatsoft

SPNivAlcool=regBank.get(40013);

//Recebe o SP do álcool do Tatsoft

SPTempAgit=regBank.get(40014);

//Recebe o SP do tempo do agitador do Tatsoft

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

// xxxxxxxxxxxxxxxxxxxxxxxxxxxx Início da leitura da Temperatura xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

temp = getTemp();

// Realiza a leitura da temperatura

temp1=temp \* 100;

TempAquec = temp1;

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

// xxxxxxxxxxxxxxxxxxxxxxxxxxxx Fim da leitrua da temperatura xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

// xxxxxxxxxxxxxxxxxxxxxxxxxxx Início da leitura do nível do tanque xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

ValEntradaAnalog = analogRead(PinEntAnalog);

// Realiza a leitura do nível

NivelTQ = ValEntradaAnalog \* 10;

// xxxxxxxxxxxxxxxxxxxxxxxxx Fim da leitura do nível do tanque xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx Realiza a leitura das entradas digitais xxxxxxxxxxxxxxxxxxxxxxxxx

if (digitalRead(BL\_Manual) == HIGH){

bitSet(EstadoEntradas, 0);

// Seta bit 0

}

else{

bitClear(EstadoEntradas, 0);

}

if (digitalRead(BL\_Compressor) == HIGH){

bitSet(EstadoEntradas, 1);

// Seta bit 1

}

else{

bitClear(EstadoEntradas, 1);

}

if (digitalRead(BL\_Agitador) == HIGH){

bitSet(EstadoEntradas, 2);

// Seta bit 2

}

else{

bitClear(EstadoEntradas, 2);

}

if (digitalRead(BL\_Agua) == HIGH){

bitSet(EstadoEntradas, 3);

// Seta bit 3

}

else{

bitClear(EstadoEntradas, 3);

}

if (digitalRead(BL\_Soda) == HIGH){

bitSet(EstadoEntradas, 4);

// Seta bit 4

}

else{

bitClear(EstadoEntradas, 4);

}

if (digitalRead(BL\_Oleo) == HIGH){

bitSet(EstadoEntradas, 5);

// Seta bit 5

}

else{

bitClear(EstadoEntradas, 5);

}

if (digitalRead(BL\_Alcooll) == HIGH){

bitSet(EstadoEntradas, 6);

// Seta bit 6

}

else{

bitClear(EstadoEntradas, 6);

}

if (digitalRead(BL\_Aquecedor) == HIGH){

bitSet(EstadoEntradas, 7);

// Seta bit 7

}

else{

bitClear(EstadoEntradas, 7);

}

if (digitalRead(BL\_Emergencia) == HIGH){

bitSet(EstadoEntradas, 8);

// Seta bit 8

}

else{

bitClear(EstadoEntradas, 8);

}

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

// xxxxxxxxxxxxxxxxxxxxxxxxxx Início de recebimento dos comandos dos Tatsoft xxxxxxxxxxxxxxxxxxxxx

// Recebe os comandos de operação do supervisório e decompoe em bits

if (bitRead(Comandos1, 0)==1){

//Bit 0 vem o comando de seleciona Auto ou Man

SelecAuto = true;

// Selecionado para Automático

bitSet(Status1, 0);

//Retorna bit indicando seleção de automático

}

else{

SelecAuto = false;

bitClear(Status1, 0);

//Retorna bit indicando seleção de manual

InicFabric = false;

// finaliza a fabricação

StatusFabric = 0;

TempAgit = 0;

}

if (bitRead(Comandos1, 1)==1){

//Bit 1 vem o comando de emergêcia atuada

Emergenc = !Emergenc;

// A cada clique do mouse, inverste o estado da chave de emergência

}

if (Emergenc){

// Emergência atuada paraliza toda fabricação em Auto

InicFabric = false;

// finaliza a fabricação

StatusFabric = 0;

TempAgit = 0;

}

if (bitRead(Comandos1, 2)==1){

//Bit 2 vem o comando de rearme geral atuada

RearmGer = true;

}

else{

RearmGer = false;

}

if (bitRead(Comandos1, 3)==1){

//Bit 3 vem o comando de liga equipamentos

LigEqp = true;

}

else{

LigEqp = false;

}

if (bitRead(Comandos1, 4)==1){

//Bit 4 vem o comando de desliga equipamentos

DeslEqp = true;

}

else{

DeslEqp = false;

}

if (bitRead(Comandos1, 5)==1){

//Bit 5 vem o comando de seleciona remoto equipamentos

SelRemEqp = true;

}

else{

SelRemEqp = false;

}

if (bitRead(Comandos1, 6)==1){

//Bit 6 vem o comando de seleciona local equipamentos

SelLocEqp = true;

}

else{

SelLocEqp = false;

}

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx Tipo de seleção dos equipamentos xxxxxxxxxxxxxxxxxxxxxxxxxxxx

/\* Seleção dos equipamentos, os valores na word significa:

1 - Válvula de alcool;

2 - Válvula de soda;

3 - Válvula de óleo;

4 - Bomba de água;

5 - Agitador;

6 - Aquecedor;

7 - Compressor;

8 - Inicia fabricação

9 - Finaliza fabricação

\*/

if (SelecEqp == 1){

SelecVAlcool = true;

// Seleciona a válvula de alcool

}

else{

SelecVAlcool = false;

}

if (SelecEqp == 2){

SelecVSoda = true;

// Seleciona a válvula de soda

}

else{

SelecVSoda = false;

}

if (SelecEqp == 3){

SelecVOleo = true;

// Seleciona a válvula de óleo

}

else{

SelecVOleo = false;

}

if (SelecEqp == 4){

SelecBomba = true;

// Seleciona a bomba

}

else{

SelecBomba = false;

}

if (SelecEqp == 5){

SelecAgit = true;

// Seleciona o agitador

}

else{

SelecAgit = false;

}

if (SelecEqp == 6){

SelecAquec = true;

// Seleciona o aquecedor

}

else{

SelecAquec = false;

}

if (SelecEqp == 7){

SelecComp = true;

// Seleciona o compressor

}

else{

SelecComp = false;

}

if (SelecAuto && (SelecEqp == 8) && (InicFabric == false)){

// Iniciar fabricação em auto

InicFabric = true; // Inicia a fabricação

StatusFabric = 10;

}

if ((SelecEqp == 9) && (InicFabric == true)){

// Finaliza a fabricação em auto

InicFabric = false; // finaliza a fabricação

StatusFabric = 0;

}

if ((SelecEqp == 8) || (SelecEqp == 9)){

// Evita comando ficar agarrado para iniciar ou finalizar

SelecEqp = 0;

regBank.set(40005, SelecEqp);

}

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx Seleção de Remoto ou Local xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

if ((SelecEqp == 1) && SelRemEqp){

// Seleciona a Válvula de alcool em remoto

SelecRemVAlcool = true;

SelecLocVAlcool = false;

bitSet(Status1, 2);

//Retorna bit indicando Válvula em remoto

bitClear(Status1, 3);

//Retorna bit indicando Válvula não local

}

if (((SelecEqp == 1) && SelLocEqp) || !digitalRead(BL\_Manual)){

// Seleciona a Válvula de alcool em Local

SelecRemVAlcool = false;

SelecLocVAlcool = true;

bitSet(Status1, 3);

//Retorna bit indicando Válvula em local

bitClear(Status1, 2);

//Retorna bit indicando Válvula não remoto

}

if ((SelecEqp == 2) && SelRemEqp){

// Seleciona a Válvula de soda em remoto

SelecRemVSoda = true;

SelecLocVSoda = false;

bitSet(Status1, 5); //Retorna bit indicando Válvula em remoto

bitClear(Status1, 6);

}

if (((SelecEqp == 2) && SelLocEqp) || !digitalRead(BL\_Manual)){

// Seleciona a Válvula de soda em Local

SelecRemVSoda = false;

SelecLocVSoda = true;

bitSet(Status1, 6);

bitClear(Status1, 5);

}

if ((SelecEqp == 3) && SelRemEqp){

// Seleciona a Válvula de óleo em remoto

SelecRemVOleo = true;

SelecLocVOleo = false;

bitSet(Status1, 8); //Retorna bit indicando Válvula em remoto

bitClear(Status1, 9);

}

if (((SelecEqp == 3) && SelLocEqp) || !digitalRead(BL\_Manual)){

// Seleciona a Válvula de óleo em Local

SelecRemVOleo = false;

SelecLocVOleo = true;

bitSet(Status1, 9);

bitClear(Status1, 8);

}

if ((SelecEqp == 4) && SelRemEqp){

// Seleciona bomba em remoto

SelecRemBomba = true;

SelecLocBomba = false;

bitSet(Status1, 11);

//Retorna bit indicando bomba em remoto

bitClear(Status1, 12);

}

if (((SelecEqp == 4) && SelLocEqp) || !digitalRead(BL\_Manual)){

// Seleciona a bomba em Local

SelecRemBomba = false;

SelecLocBomba = true;

bitSet(Status1, 12);

bitClear(Status1, 11);

}

if ((SelecEqp == 5) && SelRemEqp){

// Seleciona o agitador em remoto

SelecRemAgit = true;

SelecLocAgit = false;

bitSet(Status1, 14);

//Retorna bit indicando agitador em remoto

bitClear(Status1, 15);

}

if (((SelecEqp == 5) && SelLocEqp) || !digitalRead(BL\_Manual)){

// Seleciona o agitador em Local

SelecRemAgit = false;

SelecLocAgit = true;

bitSet(Status1, 15);

bitClear(Status1, 14);

}

if ((SelecEqp == 6) && SelRemEqp){

// Seleciona o aquecedor em remoto

SelecRemAquec = true;

SelecLocAquec = false;

bitSet(Status2, 1); //Retorna bit indicando Aquecedor em remoto

bitClear(Status2, 2);

}

if (((SelecEqp == 6) && SelLocEqp) || !digitalRead(BL\_Manual)){

// Seleciona o aquecedor em Local

SelecRemAquec = false;

SelecLocAquec = true;

bitSet(Status2, 2);

bitClear(Status2, 1);

}

if ((SelecEqp == 7) && SelRemEqp){

// Seleciona o compressor em remoto

SelecRemComp = true;

SelecLocComp = false;

bitSet(Status2, 4); //Retorna bit indicando Compressor em remoto

bitClear(Status2, 5);

}

if (((SelecEqp == 7) && SelLocEqp) || !digitalRead(BL\_Manual)){

// Seleciona o compressor em Local

SelecRemComp = false;

SelecLocComp = true;

bitSet(Status2, 5);

bitClear(Status2, 4);

}

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxx Retira todos os equipamentos de Remoto ou Local quando está em Automático xxxxxxxxxxxxx

if (SelecAuto){ //Retira de remoto ou local

SelecRemVAlcool = false;

SelecLocVAlcool = false;

bitClear(Status1, 2);

bitClear(Status1, 3);

SelecRemVSoda = false;

SelecLocVSoda = false;

bitClear(Status1, 5);

bitClear(Status1, 6);

SelecRemVOleo = false;

SelecLocVOleo = false;

bitClear(Status1, 8);

bitClear(Status1, 9);

SelecRemBomba = false;

SelecLocBomba = false;

bitClear(Status1, 11);

bitClear(Status1, 12);

SelecRemAgit = false;

SelecLocAgit = false;

bitClear(Status1, 14);

bitClear(Status1, 15);

SelecRemAquec = false;

SelecLocAquec = false;

bitClear(Status2, 1);

bitClear(Status2, 2);

SelecRemComp = false;

SelecLocComp = false;

bitClear(Status2, 4);

bitClear(Status2, 5);

}

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxx Início da lógica para controle de temperatura em Auto xxxxxxxxxxxxxxxxx

if (SelecAuto){

if (TempAquec > (SPTemperat + 100)){

LigaAquecAuto = false;

}

if (TempAquec < (SPTemperat -100)){

LigaAquecAuto = true;

}

}

//xxxxxxxxxxxxxxxxxxxxxxxxx Fim da lógica para controle de temperatura em Auto xxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxx Atuação das emergências - Local ou Remota xxxxxxxxxxxxxxxxxxxxxxxxx

if (digitalRead(BL\_Emergencia) || Emergenc){

EmergAtuada = true;

}

if (RearmGer) {

EmergAtuada = false;

}

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxx Início da lógica para fabricação em Automático xxxxxxxxxxxxxxxxxxxxxxx

if (SelecAuto && (StatusFabric == 10)){

// Liga a bomba de água em automática até SP de nível de água

if (TempAquec > SPTemperat){

if (NivelTQ < SPNivAgua){

LigaBombaAuto = true;

StatusFabric = 1;

}

}

}

if (SelecAuto && (StatusFabric == 1)){

if (NivelTQ > SPNivAgua){

LigaBombaAuto = false;

StatusFabric = 2;

}

}

if (SelecAuto && (StatusFabric == 2)){

// Liga a válvula de soda em automática até SP de nível de soda

if (NivelTQ < SPNivSoda){

LigaSodaAuto = true;

}

else{

LigaSodaAuto = false;

StatusFabric = 3;

}

}

if (SelecAuto && (StatusFabric == 3)){

// Liga a válvula de óleo em automática até SP de nível de óleo

if (NivelTQ < SPNivOleo){

LigaOleoAuto = true;

}

else{

LigaOleoAuto = false;

StatusFabric = 4;

}

}

if (SelecAuto && (StatusFabric == 4)){

// Liga a válvula de álcool em automática até SP de nível de álcool

if (NivelTQ < SPNivAlcool){

LigaAlcoolAuto = true;

}

else{

LigaAlcoolAuto = false;

StatusFabric = 5;

}

}

if (SelecAuto && (StatusFabric == 5)){

// Liga o agitador em automático até SP de Tempo de agitação

if (TempAgit < SPTempAgit){

LigaAgitAuto = true;

}

else{

LigaAgitAuto = false;

StatusFabric = 6;

}

}

if (StatusFabric < 5){

// Garante não realizar contagem em etapa de fabricação diferente da agitação

TempAgit = 0;

}

if (SelecAuto){ // O compressor sempre ficará ligado em Auto.

LigaCompAuto = true;

}

//xxxxxxxxxxxxxxxxxxxxxxxx Fim da lógica para fabricação em Automático xxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxx Início da Lógica de comando da válvula de alcool xxxxxxxxxxxxxxxxxxxxxxx

if (!SelecAuto && SelecRemVAlcool && SelecVAlcool && LigEqp){

// Em Manual, em Remoto e Liga Remoto

SaidaAlcool = HIGH;

}

if (!SelecAuto && SelecRemVAlcool && SelecVAlcool && DeslEqp){

// Em Manual, em Remoto e Desliga Remoto

SaidaAlcool = LOW;

}

if (!SelecAuto && SelecLocVAlcool){

// Em Manual, em Remoto e Liga Local

SaidaAlcool = digitalRead(BL\_Alcooll);

}

if ((!SelecAuto || !InicFabric) && LigaAlcoolAuto){

/ /Caso passar para Man ou finalizar e saída estiver ligada, deve desligar

LigaAlcoolAuto = false;

SaidaAlcool = LOW;

}

if (SelecAuto){

// Em Automático, liga em Automático

if (LigaAlcoolAuto){

SaidaAlcool = HIGH;

}

else{

SaidaAlcool = LOW;

}

}

if (EmergAtuada){

SaidaAlcool = LOW;

// Com emergência atuada, desliga todas as saídas

}

if (SaidaAlcool == HIGH){

bitSet(Status1, 4);

//Retorna bit indicando Válvula de alcool aberta

}

else{

bitClear(Status1, 4);

//Retorna bit indicando Valvula de alcool fechada

}

//xxxxxxxxxxxxxxxxxxxxxxxxx Fim da Lógica de comando da válvula de alcool xxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxx Início da Lógica de comando da válvula de Soda xxxxxxxxxxxxxxxxxxxxx

if (!SelecAuto && SelecRemVSoda && SelecVSoda && LigEqp){

// Em Manual, em Remoto e Liga Remoto

SaidaSoda = HIGH;

}

if (!SelecAuto && SelecRemVSoda && SelecVSoda && DeslEqp){

// Em Manual, em Remoto e Desliga Remoto

SaidaSoda = LOW;

}

if (!SelecAuto && SelecLocVSoda){

// Em Manual, em Remoto e Liga Local

SaidaSoda = digitalRead(BL\_Soda);

}

if ((!SelecAuto || !InicFabric) && LigaSodaAuto){

// Caso passar para Man ou finalizar e a saída estiver ligada, deve desligar

LigaSodaAuto = false;

SaidaSoda = LOW;

}

if (SelecAuto){ // Em Automático, liga em Automático

if (LigaSodaAuto){

SaidaSoda = HIGH;

}

else{

SaidaSoda = LOW;

}

}

if (EmergAtuada){

SaidaSoda = LOW;

// Com emergência atuada, desliga todas as saídas

}

if (SaidaSoda == HIGH){

bitSet(Status1, 7);

//Retorna bit indicando Válvula de soda aberta

}

else{

bitClear(Status1, 7); /

/Retorna bit indicando Valvula de soda fechada

}

//xxxxxxxxxxxxxxxxxxxxxxxxx Fim da Lógica de comando da válvula de Soda xxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxx Início da Lógica de comando da válvula de óleo xxxxxxxxxxxxxxxxxxxxxxxx

if (!SelecAuto && SelecRemVOleo && SelecVOleo && LigEqp){

// Em Manual, em Remoto e Liga Remoto

SaidaOleo = HIGH;

}

if (!SelecAuto && SelecRemVOleo && SelecVOleo && DeslEqp){

// Em Manual, em Remoto e Desliga Remoto

SaidaOleo = LOW;

}

if (!SelecAuto && SelecLocVOleo){

// Em Manual, em Remoto e Liga Local

SaidaOleo = digitalRead(BL\_Oleo);

}

if ((!SelecAuto || !InicFabric) && LigaOleoAuto){

// Caso passar para Man ou finalizar e saída estar ligada, deve desligar

LigaOleoAuto = false;

SaidaOleo = LOW;

}

if (SelecAuto){

// Em Automático, liga em Automático

if (LigaOleoAuto){

SaidaOleo = HIGH;

}

else{

SaidaOleo = LOW;

}

}

if (EmergAtuada){

SaidaOleo = LOW;

// Com emergência atuada, desliga todas as saídas

}

if (SaidaOleo == HIGH){

bitSet(Status1, 10);

//Retorna bit indicando Válvula de óleo aberta

}

else{

bitClear(Status1, 10);

//Retorna bit indicando Valvula de óleo fechada

}

// digitalWrite(S1, SaidaOleo);

// Aciona a saída para ligar a válvula de Óleo

//xxxxxxxxxxxxxxxxxxxxxxxx Fim da Lógica de comando da válvula de óleo xxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxx Início da Lógica de comando da Bomba xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

if (!SelecAuto && SelecRemBomba && SelecBomba && LigEqp){

// Em Manual, em Remoto e Liga Remoto

SaidaBomba = HIGH;

}

if (!SelecAuto && SelecRemBomba && SelecBomba && DeslEqp){

// Em Manual, em Remoto e Desliga Remoto

SaidaBomba = LOW;

}

if (!SelecAuto && SelecLocBomba){

// Em Manual, em Remoto e Liga Local

SaidaBomba = digitalRead(BL\_Agua);

}

if ((!SelecAuto || !InicFabric) && LigaBombaAuto){

// Caso passar para Man ou finalizar e saída estar ligada, deve desligar

LigaBombaAuto = false;

SaidaBomba = LOW;

}

if (SelecAuto){

// Em Automático, liga em Automático

if (LigaBombaAuto){

SaidaBomba = HIGH;

}

else{

SaidaBomba = LOW;

}

}

if (EmergAtuada){

SaidaBomba = LOW;

// Com emergência atuada, desliga todas as saídas

}

if (SaidaBomba == HIGH){

bitSet(Status1, 13);

//Retorna bit indicando bomba ligada

}

else{

bitClear(Status1, 13);

//Retorna bit indicando bomba desligada

}

// digitalWrite(S1, SaidaOleo);

// Aciona a saída para ligar a bomba

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Fim da Lógica de comando da bomba xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Início da Lógica de comando do agitador xxxxxxxxxxxxxxxxxxxxxxxxxx

if (!SelecAuto && SelecRemAgit && SelecAgit && LigEqp){

// Em Manual, em Remoto e Liga Remoto

SaidaAgit = HIGH;

}

if (!SelecAuto && SelecRemAgit && SelecAgit && DeslEqp){

// Em Manual, em Remoto e Desliga Remoto

SaidaAgit = LOW;

}

if (!SelecAuto && SelecLocAgit){

// Em Manual, em Remoto e Liga Local

SaidaAgit = digitalRead(BL\_Agitador);

}

if ((!SelecAuto || !InicFabric) && LigaAgitAuto){

// Caso passar para Man ou finalizar e saída estiver ligada, deve desligar

LigaAgitAuto = false;

SaidaAgit = LOW;

}

if (SelecAuto){

// Em Automático, liga em Automático

if (LigaAgitAuto){

SaidaAgit = HIGH;

}

else{

SaidaAgit = LOW;

}

}

if (EmergAtuada){

SaidaAgit = LOW;

// Com emergência atuada, desliga todas as saídas

}

if (SaidaAgit == HIGH){

bitSet(Status2, 0);

//Retorna bit indicando agitador ligado

}

else{

bitClear(Status2, 0);

//Retorna bit indicando agitador desligado

}

// digitalWrite(S1, SaidaAgit);

// Aciona a saída para ligar o agitador

//xxxxxxxxxxxxxxxxxxxxxxxxxx Fim da Lógica de comando do Agitador xxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxx Início da Lógica de comando do aquecedor xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

if (!SelecAuto && SelecRemAquec && SelecAquec && LigEqp){

// Em Manual, em Remoto e Liga Remoto

SaidaAquec = HIGH;

}

if (!SelecAuto && SelecRemAquec && SelecAquec && DeslEqp){

// Em Manual, em Remoto e Desliga Remoto

SaidaAquec = LOW;

}

if (!SelecAuto && SelecLocAquec){

// Em Manual, em Remoto e Liga Local

SaidaAquec = digitalRead(BL\_Aquecedor);

}

if ((!SelecAuto || !InicFabric) && LigaAquecAuto){

// Caso passar para Man ou finalizar e saída estar ligada, deve desligar

LigaAquecAuto = false;

SaidaAquec = LOW;

}

if (SelecAuto){

// Em Automático, liga em Automático

if (LigaAquecAuto){

SaidaAquec = HIGH;

}

else{

SaidaAquec = LOW;

}

}

if (EmergAtuada){

SaidaAquec = LOW;

// Com emergência atuada, desliga todas as saídas

}

if (SaidaAquec == HIGH){

bitSet(Status2, 3);

//Retorna bit indicando aquecedor ligado

}

else{

bitClear(Status2, 3);

//Retorna bit indicando aquecedor desligado

}

// digitalWrite(S1, SaidaAgit);

// Aciona a saída para ligar o aquecedor

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Fim da Lógica de comando do aquecedor xxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxx Início da Lógica de comando do compressor xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

if (!SelecAuto && SelecRemComp && SelecComp && LigEqp){

// Em Manual, em Remoto e Liga Remoto

SaidaComp = HIGH;

}

if (!SelecAuto && SelecRemComp && SelecComp && DeslEqp){

// Em Manual, em Remoto e Desliga Remoto

SaidaComp = LOW;

}

if (!SelecAuto && SelecLocComp){

// Em Manual, em Remoto e Liga Local

SaidaComp = digitalRead(BL\_Compressor);

}

if ((!SelecAuto || !InicFabric) && LigaCompAuto){

// Caso passar para Man ou finalizar e saída estar ligada, deve desligar

LigaCompAuto = false;

SaidaComp = LOW;

}

if (SelecAuto){

// Em Automático, liga em Automático

if (LigaCompAuto){

SaidaComp = HIGH;

}

else{

SaidaComp = LOW;

}

}

if (EmergAtuada){

SaidaComp = LOW;

// Com emergência atuada, desliga todas as saídas

}

if (SaidaComp == HIGH){

bitSet(Status2, 6);

//Retorna bit indicando compressor ligado

}

else{

bitClear(Status2, 6);

//Retorna bit indicando compressor desligado

}

// digitalWrite(S1, SaidaAgit);

// Aciona a saída para ligar o compressor

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Fim da Lógica de comando do compressor xxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxx Escreve todas as saída físicas xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

digitalWrite(Saida\_Liga\_Agua, !SaidaBomba);

digitalWrite(Saida\_Liga\_Oleo, !SaidaOleo);

digitalWrite(Saida\_Liga\_Soda, !SaidaSoda);

digitalWrite(Saida\_Liga\_Alcool, !SaidaAlcool);

digitalWrite(Saida\_Liga\_Agitador, !SaidaAgit);

digitalWrite(Saida\_Liga\_Compressor, !SaidaComp);

digitalWrite(Saida\_Liga\_Aquecedor, !SaidaAquec);

//xxxxxxxxxxxxxxxxxxxxxxxxxxx Fim escreve todas saídas físicas xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

if (EmergAtuada) {

bitSet(Status1, 1);

//Retorna bit indicando emergência atuada

}

else{

bitClear(Status1, 1);

//Retorna bit indicando emergência atuada

}

bitClear(Comandos1, 1);

//Reset o comando de emergência

bitClear(Comandos1, 2);

//Reset o comando de rearme

bitClear(Comandos1, 3);

//Reset o comando de Liga

bitClear(Comandos1, 4);

//Reset o comando de desliga

bitClear(Comandos1, 5);

//Reset o comando de seleciona remoto

bitClear(Comandos1, 6);

//Reset o comando de seleciona Local

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxx Envia os dados para o Modbus xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

regBank.set(40001, Comandos1);

regBank.set(40002, Comandos2);

regBank.set(40003, Status1);

regBank.set(40004, Status2);

//regBank.set(40006, Segundos);

regBank.set(40007, NivelTQ);

regBank.set(40008, TempAquec);

//regBank.set(40009, SPTemperat);

//regBank.set(40010, SPNivAgua);

//regBank.set(40011, SPNivSoda);

//regBank.set(40013, SPNivAlcool);

//regBank.set(40014, SPTempAgit);

regBank.set(40015, StatusFabric);

regBank.set(40016, TempAgit);

regBank.set(40017, SPNivOleo);

//regBank.set(40021, EntSimuladas);

regBank.set(40022, EstadoEntradas);

slave.run();

// Chama o evento do Modbus Slave

digitalWrite(pin, LOW);

//Apaga led da saída 13

t.update();

// Chama o evento do Timer para atualizar

delay(300);

// Gera atraso de 200 ms

}

void Conta\_1s(){

// função disparada a cada 1000 ms do evento de Timer

Segundos = Segundos + 1;

if (LigaAgitAuto){

TempAgit = TempAgit + 1;

}

digitalWrite(pin, HIGH);

Pulso = !Pulso;

// Pulso usado para gerar movimentos no Tatsoft

if ((Pulso) && (SaidaAgit == HIGH)){

bitSet(Status2, 15);

}

else{

bitClear(Status2, 15);

}

if (Segundos > 60){

Segundos = 0;

}

}

float getTemp(){

byte data[12];

byte addr[8];

if ( !ds.search(addr)) {

//no more sensors on chain, reset search

ds.reset\_search();

return -1000;

}

if ( OneWire::crc8( addr, 7) != addr[7]) {

return -1000;

}

if ( addr[0] != 0x10 && addr[0] != 0x28) {

return -1000;

}

ds.reset();

ds.select(addr);

ds.write(0x44,1);

byte present = ds.reset();

ds.select(addr);

ds.write(0xBE);

for (int i = 0; i < 9; i++) {

data[i] = ds.read();

}

ds.reset\_search();

byte MSB = data[1];

byte LSB = data[0];

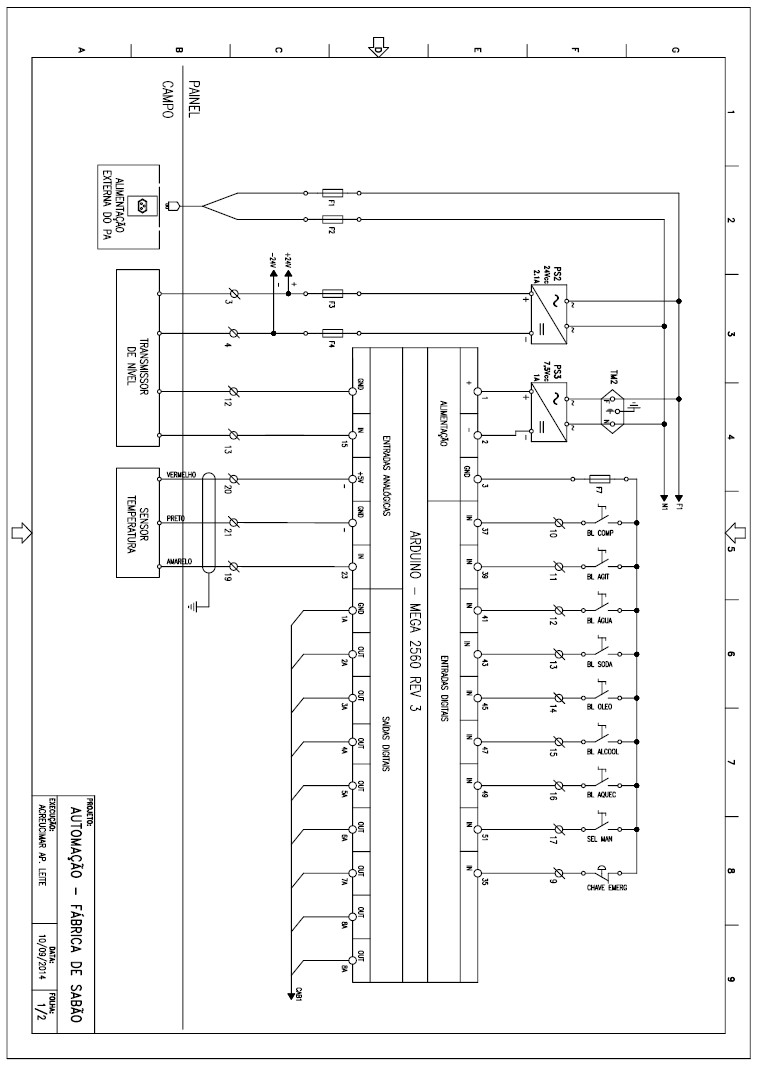
float TRead = ((MSB<<8) | LSB);

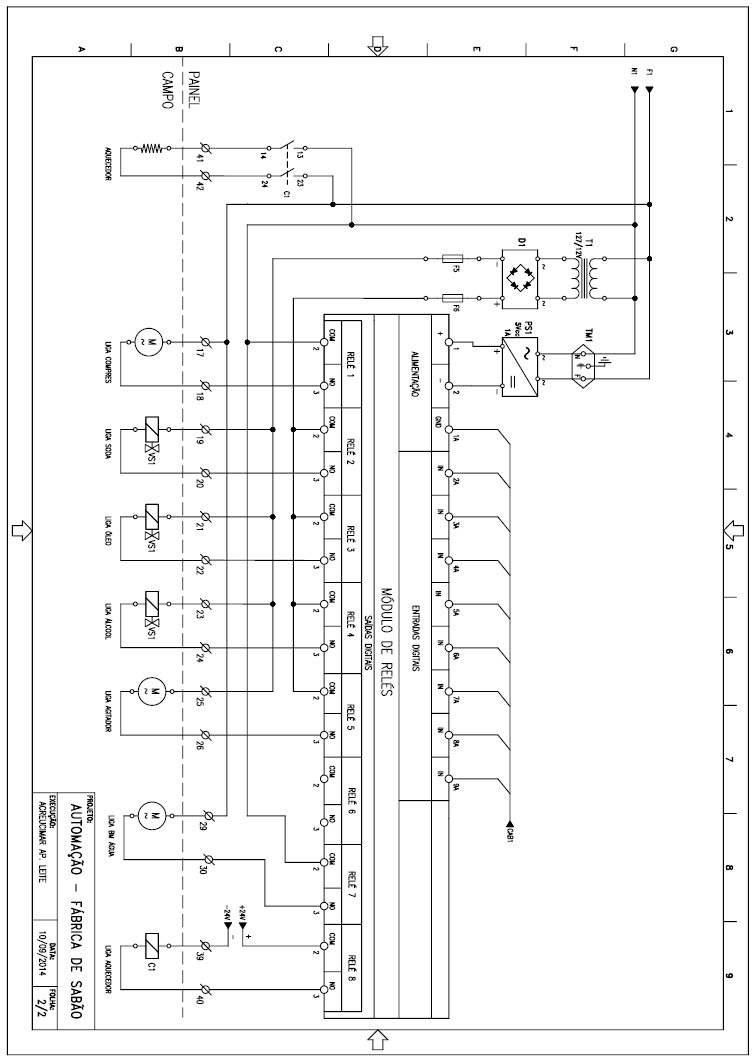
float Temperature = TRead / 16;

return Temperature;

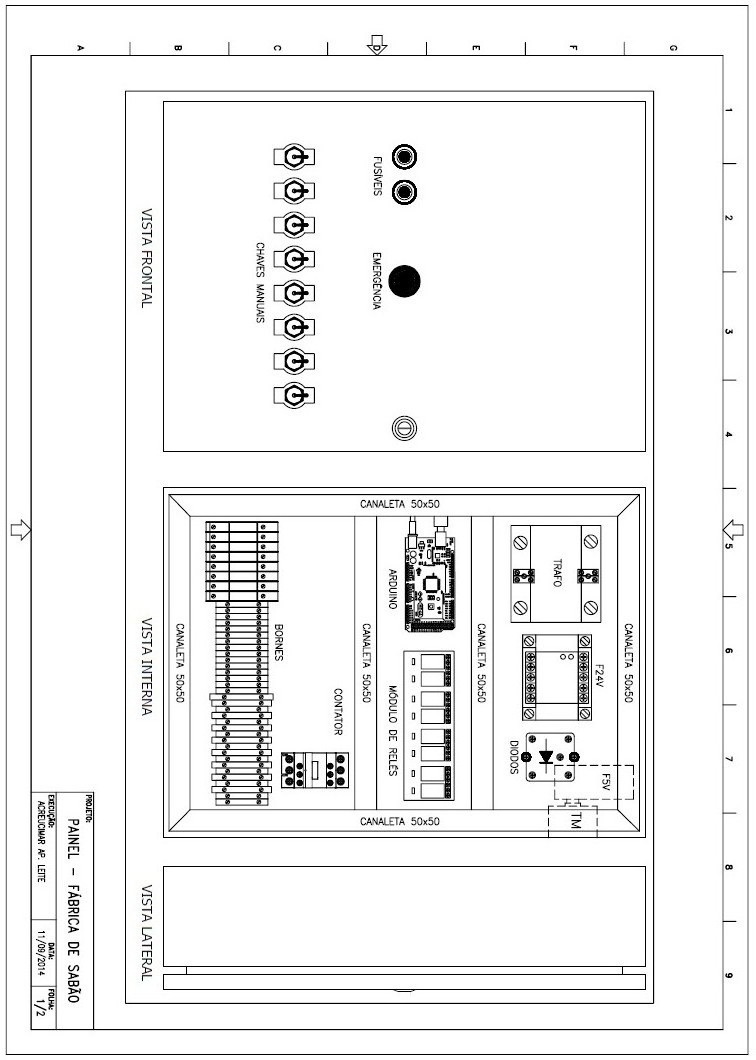
}

**APÊNDICE B – DIAGRAMA DA FÁBRICA DE SABÃO**

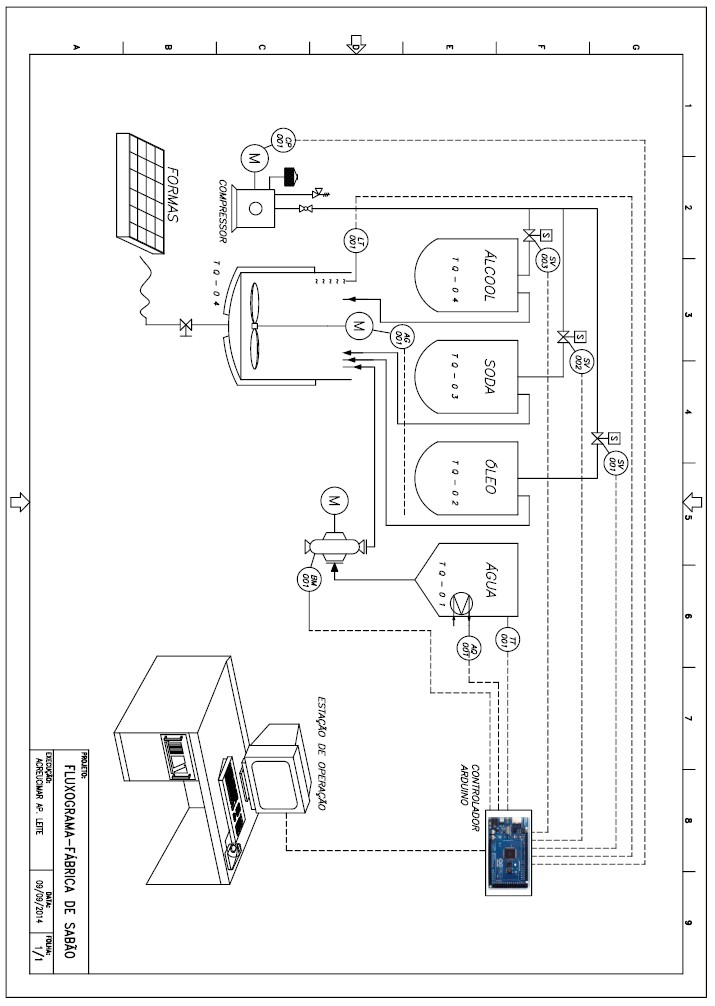
****

****

**APÊNDICE C – PROJETO DO PAINEL**

****

**APÊNDICE D – FLUXOGRAMA DA FÁBRICA DE SABÃO**

****