



**UNIVERSIDADE FEDERAL DO CERÁ**  
**CAMPUS DE SOBRAL**  
**CURSO DE ENGENHARIA ELÉTRICA**

**RODOLFO MARQUES LIMA DE SANTANA**

**SUBSTITUIÇÃO DE CLP POR RASPBERRY PI E ARDUINO EM MÁQUINA  
EMPACOTADORA AUTOMÁTICA**

**SOBRAL**

**2016**

**RODOLFO MARQUES LIMA DE SANTANA**

**AUTOMAÇÃO DE UMA MÁQUINA DE ENVASE DE REJUNTE**

Monografia apresentada ao Curso de Engenharia Elétrica da Universidade Federal do Ceará, como requisito parcial para obtenção do título de Graduado em Engenharia Elétrica. Áreas de Concentração: Automação e Microcontrolador.

Orientador: Prof. Me. Rômulo Nunes de Carvalho Almeida.

**SOBRAL**

**2016**

RODOLFO MARQUES LIMA DE SANTANA

**SUBSTITUIÇÃO DE CLP POR RASPBERRY PI E ARDUINO EM MÁQUINA  
EMPACOTADORA AUTOMÁTICA**

Monografia apresentada ao Curso de Engenharia Elétrica da Universidade Federal do Ceará, como requisito parcial para obtenção do título de Graduado em Engenharia Elétrica. Áreas de Concentração: Automação e Microcontrolador.

Aprovada em: 17/02/2016.

BANCA EXAMINADORA

---

Prof. Me. Rômulo Nunes de Carvalho Almeida (Orientador)  
Universidade Federal do Ceará (UFC)

---

Prof. Esp. Killdary Aguiar de Santana  
Universidade Federal do Ceará (UFC)

---

Wilker Francisco Dias Felipe  
Engenheiro Eletricista

À Deus, minha família, aos meus  
professores.

## **AGRADECIMENTOS**

Agradeço a Deus, primeiramente, pela saúde, disposição e sabedoria.

Agradeço a minha Mãe, meu Pai e familiares.

Agradeço ao meu orientador Professor Rômulo Nunes de Carvalho Almeida e aos componentes da banca examinadora Professor Kildary Aguiar de Santana e o Engenheiro Eletricista Wilker Francisco Dias Felipe.

Agradeço também aos professores André dos Santos Lima, Carlos Elmano de Alencar e Silva, Juan Carlos Oliveira de Medeiros, Marcelo Marques Simões de Souza, Marcus Rogério de Castro, Rômulo Nunes de C. Almeida.

A mente que se abre a uma nova ideia jamais voltará ao seu tamanho original.

(Albert Einstein)



## **RESUMO**

Este trabalho desenvolve o estudo de uma máquina automática de empacotar analisando suas partes constituintes e o funcionamento. É proposto a substituição do CLP utilizado, por um conjunto Raspberry Pi, Arduino e IHM sensível ao toque, de forma a possibilitar maior número de ferramentas ao equipamento e facilitar a operação, alinhado com baixo custo. É mostrando o estudo de algumas tecnologias disponíveis e o desenvolvimento deste trabalho. Para verificação inicial é confeccionado e testado um protótipo, de forma a avaliar a viabilidade da substituição, o desempenho e a aceitação.



## **ABSTRACT**

This paper develops the study of an automatic packaging machine analyzing its constituent parts and operation. It is proposed to replace the PLC used by a set Raspberry Pi, Arduino and touch IHM so as to enable greater number of tools and equipment to facilitate operation, lined with a low cost. It is showing the study of some technologies available and the development of this work. For initial verification is made and tested a prototype in order to assess the feasibility of substitution, the performance and acceptance.

## LISTA DE FIGURAS

Figura 1 - Modelo simplificado do processo. ....	19
Figura 2 - Foto geral da MAE. ....	20
Figura 3 – Processo de transformação da embalagem. ....	21
Figura 4 - Processo do produto na MAE. ....	21
Figura 5 - Procasso de varredura do CLP. ....	22
Figura 6 - Topologia do CLP. ....	23
Figura 7 - Periféricos empregados no microcotrolador. ....	24
Figura 8 - Comparativo entre modelos do Raspberry Pi. ....	25
Figura 9 - a) Embalagem em filme b) Embalagem transformada. ....	27
Figura 10 - Disposição das soldas na embalagem. ....	27
Figura 11 - Diagrama de funcionamento do CLP na MAE. ....	28
Figura 12 - Interface Homem Máquina do CLP. ....	29
Figura 13 - Partes integrantes do tubo formador. ....	30
Figura 14 - Diagrama do puxador. ....	31
Figura 15 - Solda vertical. ....	32
Figura 16 - Diagrama da mordança. ....	33
Figura 17 - Resfriadores. ....	34
Figura 18 - Diagrama de blocos simplificado do funcionamento da MAE. ....	35
Figura 19 – Tabela com características do PIC18F4550. ....	36
Figura 20 - Diagrama de pinos do PIC18F4550 (Microchip). ....	37
Figura 21 - Cubieboard 2. ....	38
Figura 22 - Arduino Mega 2560. ....	38
Figura 23 – Exemplo ilustrativo do protocolo de comunicação I <sup>2</sup> C. ....	39
Figura 24 - Slogam UWP. ....	40
Figura 25 - Topologia atual. ....	42
Figura 26 – Cabeçalho do código. ....	44
Figura 27 - Função setup. ....	45
Figura 28 Função “loop”. ....	46
Figura 29 - Função CicloMaquina. ....	47
Figura 30 - Função Recebimento. ....	48
Figura 31 - Função Requisicao. ....	48
Figura 32 - Tela de parametrização. ....	49

Figura 33 - Tela com divisão de linhas e colunas.....	49
Figura 34 - Definição de linha e colunas. ....	50
Figura 35 – Inicialização da comunicação I2C C#. ....	51
Figura 36 - Escrever no Arduino.....	52
Figura 37 - Sequência de bytes para escrita. ....	52
Figura 38 - Evento PointerExited do parâmetro INICIO CICLO.....	53
Figura 39 – Evento de click no botão -1. ....	53
Figura 40 – Evento click botão CARREGAR. ....	54
Figura 41 - Base para blocos. ....	55
Figura 42 - Bloco para acionamento de cargas com corrente alternada. .	55
Figura 43 - Bloco amplificador operacional. ....	56
Figura 44 - Bloco de entradas digitais. ....	56
Figura 45 - Bloco de saídas digitais.....	57
Figura 46 - Placas de controle montadas. ....	57
Figura 47 – CLP Com IHM incorporada (Unitronics). ....	59
Figura 48 – Máquina automática em teste. ....	61
Figura 49 - Detalhe da IHM. ....	62
Figura 50 - Hardware de acionamento instalado.....	62

**LISTA DE TABELAS**

	Páginas
Tabela 1 - Sequancia de bytes protocolo interno.	47

**LISTA DE ABREVIATURAS E SIGLAS**

CLP - Controlador Lógico Programável

PIC - Controlador Integrado de Periféricos

MAE - Máquina Automática de Empacotar

IEC - "*Internacional Electrotechnical Commission*"

PIC - Controlador Integrado de Periféricos

IoT – "*Internet Of Things*"

UWP - Universal Windows Platform

## Sumário

1. INTRODUÇÃO .....	16
1.1. Introdução uso da embalagem.....	16
1.2. Objetivos .....	17
1.3. Divisão do Trabalho .....	18
2. Fundamentação teórica.....	19
2.1. Escopo geral das máquinas automáticas de empacotar.....	19
2.2. Controlador Lógico Programável CLP .....	21
2.3. Microcontrolador .....	23
2.4. Raspberry Pi .....	24
3. PARTES E FUNCIONAMENTO DA MÁQUINA ESTUDO DE CASO.	26
3.1. Introdução ao funcionamento.....	26
3.1.1. Embalagem.....	26
3.2. Utilização do controlador lógico programável (CLP) .....	28
3.3. Funcionamento do tubo formador .....	29
3.4. Funcionamento do puxador .....	30
3.5. Funcionamento da Solda vertical .....	31
3.6. Funcionamento da mordaza e faca.....	32
3.7. Funcionamento dos resfriamentos .....	34
3.8. Funcionamento geral da máquina.....	34
4. PROJETO DO BLOCO DE CONTROLE .....	36
4.1. Visão Geral .....	36
4.1.1. Microchip PIC 18F4550 .....	36
4.1.2. Cubieboard 2 .....	37
4.1.3. Arduino Mega 2560 .....	38
4.1.4. Raspberry PI 2 B .....	39
4.2. Protocolo I <sup>2</sup> C (Inter-Integrated Circuit) .....	39

4.3. Windows 10 IoT ( <i>Internet Of Things</i> ) .....	39
4.4. Componentes selecionados.....	40
4.5. Topologia Atual .....	42
4.6. Código utilizado no Arduino .....	42
4.7. Código no Visual Studio.....	48
4.7.1. XAML.....	49
4.7.2. C# .....	50
4.8. Projeto eletrônico .....	54
4.9. Comparativo de custo .....	57
4.9.1. CLP atual.....	57
4.9.2. CLP semelhante ao proposto .....	58
4.9.3. Conjunto proposto .....	60
5. RESULTADOS EXPERIMENTAIS .....	61
6. CONCLUSÃO.....	63
7. BIBLIOGRAFIA .....	64
8. Anexos .....	67
8.1. Código Arduino .....	67
8.2. Código Raspberry Pi.....	70
8.2.1. XAML.....	70
8.2.2. C# .....	74

## 1. INTRODUÇÃO

### 1.1. Introdução uso da embalagem

Segundo (Resende, 2014) “*Embalagem para alimentos é o artigo que está em contato direto com alimentos, destinado a contê-los, desde a sua fabricação até a sua entrega ao consumidor, com a finalidade de protegê-los de agente externos, de alterações e de contaminações, assim como de adulterações (RDC n 91/01).*”

A embalagem surgiu com a necessidade do homem de armazenar produtos principalmente, de gênero alimentício. A mais de 10.000 anos, essa necessidade surgiu devido a mudanças comportamentais, onde anteriormente o homem se deslocava até o alimento e o consumia, com aumento das distâncias entre as moradias e as fontes de comida, surgiu assim à necessidade do homem de proteger e transportar o alimento (Cavalcanti, et al., 2006).

No Brasil por volta de 1945, a maioria dos produtos eram vendidos sem embalagem prévia, onde eram pesados e posteriormente embalados em papel ou colocados em sacos de papel. Poucos eram os produtos vendidos já embalados como o cigarro, cerveja, inseticidas, entre outros. Somente depois da 2º Guerra Mundial, houve uma maior utilização de embalagens processadas. Neste período surgiram fábricas de embalagens, principalmente de sacos de papel, para suprir as necessidades de transporte e as demandas dos supermercados. A partir dos anos 60 aumenta a produção de embalagens plásticas. A partir dos anos 70 a indústria vem modificando seus formatos de embalagens para se adequar as novas necessidades de mercado e oferecer mais segurança e praticidade aos consumidores (Cavalcanti, et al., 2006).

Pelo decreto-lei nº 986, de 21 de outubro de 1969, segue parte do artigo 11 referente ao rótulo das embalagens.

*“Art 11. Os rótulos deverão mencionar em caracteres perfeitamente legíveis:*

*I - A qualidade, a natureza e o tipo do alimento, observadas a definição, a descrição e a classificação estabelecida no respectivo padrão de identidade e*



*qualidade ou no rótulo arquivado no órgão competente do Ministério da Saúde, no caso de alimento de fantasia ou artificial, ou de alimento não padronizado;*

*II - Nome e/ou a marca do alimento;*

*III - Nome do fabricante ou produtor;*

*IV - Sede da fábrica ou local de produção;*

*V - Número de registro do alimento no órgão competente do Ministério da Saúde;*

*VI - Indicação do emprego de aditivo intencional, mencionando-o expressamente ou indicando o código de identificação correspondente com a especificação da classe a que pertencer;*

*VII - Número de identificação da partida, lote ou data de fabricação, quando se tratar de alimento perecível;*

*VIII - O peso ou o volume líquido;*

*IX - Outras indicações que venham a ser fixadas em regulamentos.”*

No Brasil, segundo o artigo 8 da Lei n. 9782/99 a Agência Nacional de Vigilância Sanitária ANVISA é responsável por regulamentar embalagens e outros materiais ou equipamento que tenham contato com alimentos, que sejam utilizados durante a elaboração, fracionamento, armazenamento... Sendo dividido em tipos de materiais plástico, celulósico, metálico, vidro, têxtil e elastomérico (ANVISA).

A embalagem pode ser considerada como veículo de venda, divulgação e propaganda, onde 18 mil produtos entram no mercado anualmente, em sua grande maioria não aparecem em propagandas, além de diferenciar e agregar valor ao produto (Barão, 2011).

Em vista desta necessidade do homem de procurar sistemas mais eficientes e novas tecnologias para fabricação, faz-se necessária a utilização destas embalagens com processos a cada dia mais automatizados, com medidas de peso, volume, entre outros de forma obter maior qualidade e higiene com menos desperdícios no processo fabril.

## **1.2. Objetivos**

Este trabalho tem o objetivo geral de estudar o funcionamento de uma máquina de empacotamento automático e verificar a viabilidade da substituição do

CLP, já existente na máquina por um conjunto: tela sensível ao toque, Arduino e Raspberry.

O objetivo específico é substituir a automação existente por outra com menor custo e maior quantidade de recursos.

### **1.3. Divisão do Trabalho**

Este capítulo mostra uma sequência histórica do surgimento e aplicação das embalagens, o que permite compreender a importância da embalagem e sua evolução. Além de definir o assunto a ser desenvolvido durante o trabalho.

O segundo capítulo faz a descrição geral da Máquina Automática de Empacotar MAE, descreve de forma sucinta o que que é um CLP, um microcontrolador e o Raspberry Pi.

O terceiro capítulo Faz uma descrição aprofundada da MAE, mostrando de forma individual suas partes integrantes e explicando seu funcionamento.

O quarto capítulo descreve de forma sucinta os componentes utilizados durante o processo de evolução deste projeto, além de explicar os motivos para a utilização da topologia atual. Também descrever a parte de software desenvolvida.

O quinto capítulo apresenta os resultados experimentais, mostrando o projeto desenvolvido sendo testado. O Sexto capítulo trata das considerações finais e vertentes para trabalhos futuros.

## 2. Fundamentação teórica

### 2.1. Escopo geral das máquinas automáticas de empacotar

A MAE opera de forma automática, sem que haja o contato direto entre o produto a ser embalado e o operador, durante o processo de empacotamento. A MAE possui atuadores e motores, os quais operam de forma sincronizada, onde cada ciclo de movimento dos motores e atuadores são programados para ter o início e término do acionamento no tempo especificado.

De forma simplificada a MAE recebe: o produto a ser embalado, a embalagem em formato de filme, o ar comprimido e eletricidade. Tendo na saída o produto embalado na quantidade correta. Esse processo pode ser visto de forma simplificada na Figura 1.

*Figura 1 - Modelo simplificado do processo.*



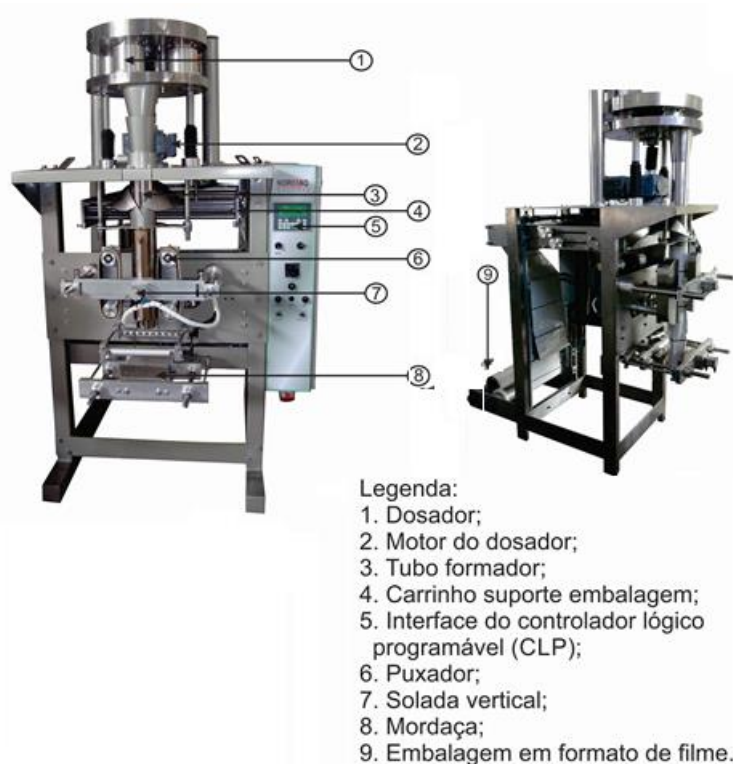
*Fonte (autor)*

O modelo da MAE estudado efetua o empacotamento vertical. Possui a regulação de peso de forma volumétrica, produz uma quantidade de aproximadamente sessenta pacotes por minuto, podendo variar dependendo do produto a ser embalado, sendo bastante indicada para a utilização em grãos, cereais, entre outros.

A MAE opera de forma automática, sem que haja o contato direto entre o produto a ser embalado e o operador, durante o processo de empacotamento. A MAE possui atuadores e motores, os quais operam de forma sincronizada, onde cada ciclo de movimento dos motores e atuadores são programados para ter o início e término do acionamento no tempo especificado.

As partes integrantes da MAE de forma simplificada, onde cada parte é melhor explicada em seu respectivo tópicos (Figura 2).

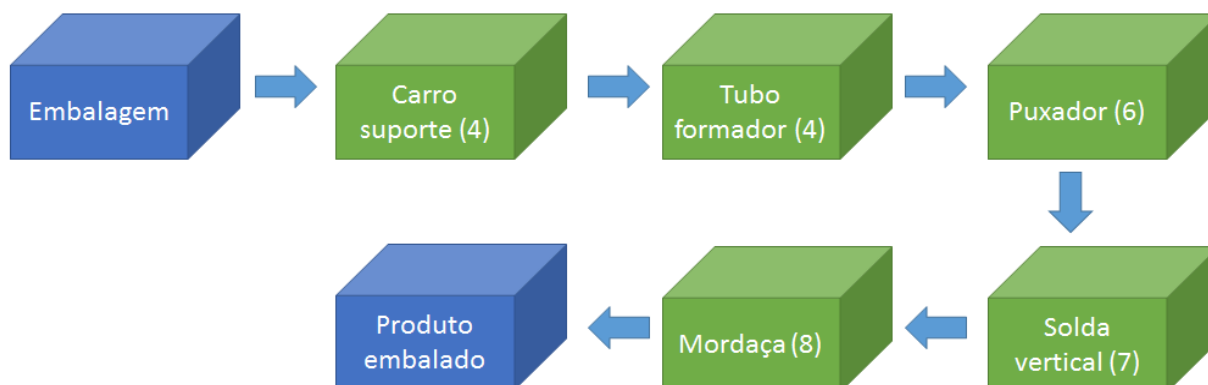
*Figura 2 - Foto geral da MAE.*



*Fonte: Autor.*

O processo da embalagem na MAE é dado da seguinte forma, ela é colocada na parte de trás, conforme mostra o número 9 na Figura 2, posteriormente é desenrolada conforme seu uso, passando pelos seguintes componentes, carro suporte, tubo formador, puxador, solda vertical e mordança. A Figura 3 mostra em formato de blocos o processo da embalagem na MAE.

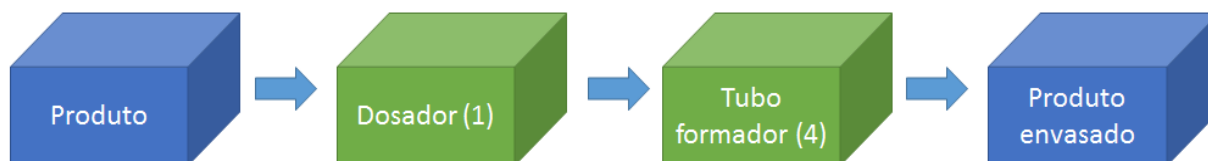
Figura 3 – Processo de transformação da embalagem.



Fonte (autor).

O produto é inserido na MAE no dosador, que de forma volumétrica retira a quantidade especificada do produto, então libera essa quantidade no interior do tubo formador, onde na parte externa do tubo formador deve estar a embalagem pronta para receber o produto. A Figura 4 mostra por blocos o processo do produto durante o envase.

Figura 4 - Processo do produto na MAE.



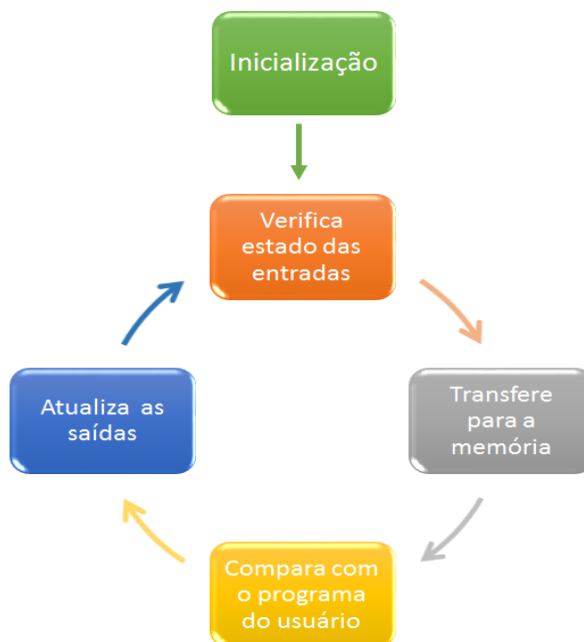
Fonte (autor)

## 2.2. Controlador Lógico Programável CLP

É um computador industrial capaz de armazenar instruções para aplicação de funções, da mesma forma que um computador pessoal o CLP também possui uma unidade central de processamento. O CLP também possui módulos de entradas e saídas, módulos de comunicação, entre outros. Para a programação do CLP são padronizadas linguagens pela Comissão Eletrotécnica Internacional “International Electrotechnical Commission” IEC (Guedes, 2009).

O CLP funciona por intermédio de ciclos de varredura, na primeira vez que é energizado é feita a inicialização, logo após é iniciado o ciclo, onde são verificados o status de todas as entradas, posteriormente esses valores são transferidos para memória, logo após esse mapa de memória é comparado com o programa gravado no CLP, e então são tomadas decisões quanto as saídas e o ciclo volta novamente a se repetir (Junior, 2012). A Figura 5 mostra o processo de Varredura do CLP.

Figura 5 - Processo de varredura do CLP.

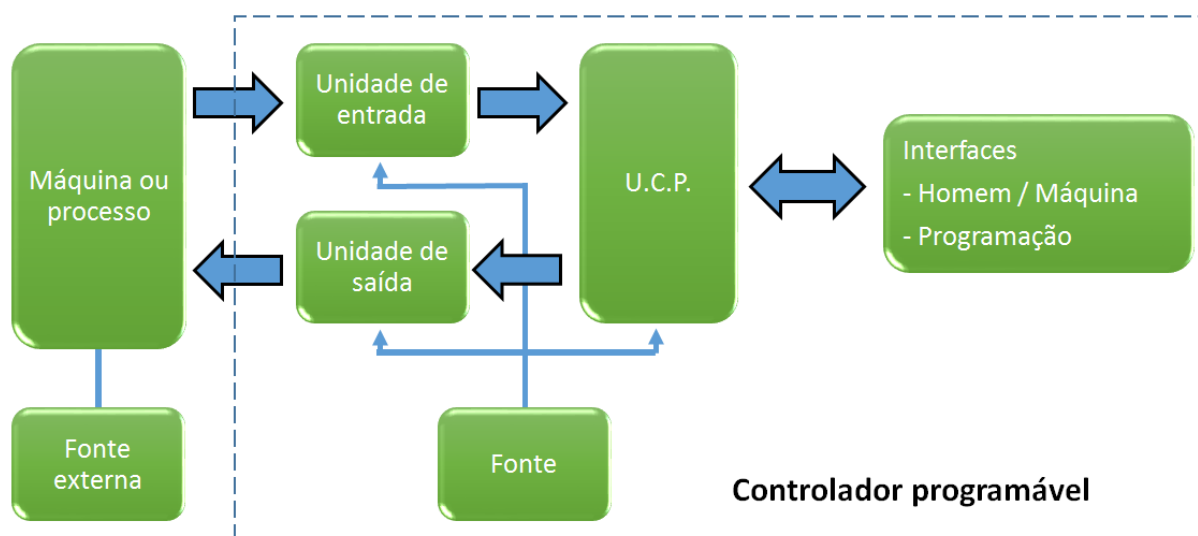


Fonte: Junior 2012 (modificado)

Teve seu advento em 1968 pela divisão da General Motors, onde o objetivo inicial foi relativo a melhorias nos painéis a reles, com redução de custo, menor volume e maior flexibilidade de programação, onde em painéis a relé necessitavam de modificações nas fiações para alteração de lógica. Com aumento da demanda e advento da tecnologia foram sendo atribuídas novas funções, melhor capacidade de processamento diversos tipos de expansão, entre outros. O CLP se difere de outros dispositivos por ter lógica de programação sequencial além de ser projetado para ambiente industrial, podendo suportar níveis de ruído interferências eletromagnéticas, elevadas temperaturas, vibrações mecânicas, entre outros conforme definidos pelos fabricantes (Pereira, et al., 2009).

A Figura 6 mostra de forma geral a topologia interna de um CLP.

Figura 6 - Topologia do CLP.



Fonte: Pereira, et al., 2009 (modificado).

### 2.3. Microcontrolador

Os microcontroladores surgiram com um pedido da BUSICOM a Intel em 1969, para a fabricação de um circuito integrado para calculadoras, porém o engenheiro Marcian Hoff propôs um circuito integrado programável, podendo assim ser utilizado em várias aplicações alterando apenas a programação, com isso surgiu o microprocessador 4004, que operava a 6 khz e 4 bits. Posteriormente com a necessidade de integrar periféricos foi criado o microcontrolador (Nunes, 2013).

É um componente eletrônico que agrega vários dispositivos em um único encapsulamento, podendo ser citados: microprocessador, memória, interfaces de entrada e saída, além de periféricos como: temporizadores, comparadores, geradores de clock, conversores digital/analógico. Possuem volume reduzido, baixo custo e baixo consumo de energia. A programação pode ser realizada por intermédio de várias linguagens de programação como Assembly, C, Pascal, entre outras (Silva, 2009).

A Figura 7 mostra de forma ilustrativa os periféricos que são empregados e um microcontrolador.





Figura 8 - Comparativo entre modelos do Raspberry Pi.



	Model A	Model A+	Model B	Model B+	2 Model B
SoC	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2836
CPU	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	900MHz Quad-core ARM Cortex-A7
GPU	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV
RAM	256Mb	256Mb	512Mb	512Mb	1Gb
USB	1	1	2	4	4
Video	RCA, HDMI	Jack, HDMI	RCA, HDMI	Jack, HDMI	Jack, HDMI
Audio	Jack, HDMI	Jack, HDMI	Jack, HDMI	Jack, HDMI	Jack, HDMI
Boot	SD	MicroSD	SD	MicroSD	MicroSD
Red	-	-	Ethernet 10/100	Ethernet 10/100	Ethernet 10/100
Consumo	300mA / 1,5w / 5v	400mA / 2w / 5v	700mA / 3,5w / 5v	500mA / 2,5w / 5v	800mA / 4w / 5v
Alimentación	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO
Tamaño	85,6 x 53,98 mm	65 x 56 mm	85,6 x 53,98 mm	85 x 56 mm	85 x 56 mm
Precio	25\$	20\$	35\$	35\$	35\$

Fonte: <http://comohacer.eu/comparativa-y-analisis-raspberry-pi-vs-competencia/> (modificado).

### **3. PARTES E FUNCIONAMENTO DA MÁQUINA ESTUDO DE CASO**

#### **3.1. Introdução ao funcionamento**

É estudado uma máquina empacotamento vertical, que possui a regulagem de peso de forma volumétrica, produz uma quantidade de aproximadamente sessenta pacotes por minuto, podendo variar dependendo do produto a ser embalado, sendo bastante indicada para a utilização em grãos, cereais, entre outros.

Para esse trabalho foi escolhido a MAE modelo Rimaq RM10C do fabricante Norimaq Maquinas, devido à liberdade e as facilidades disponíveis por parte do proprietário da empresa, para o estudo da tecnologia e disponibilidade de uso de maquinário e materiais.

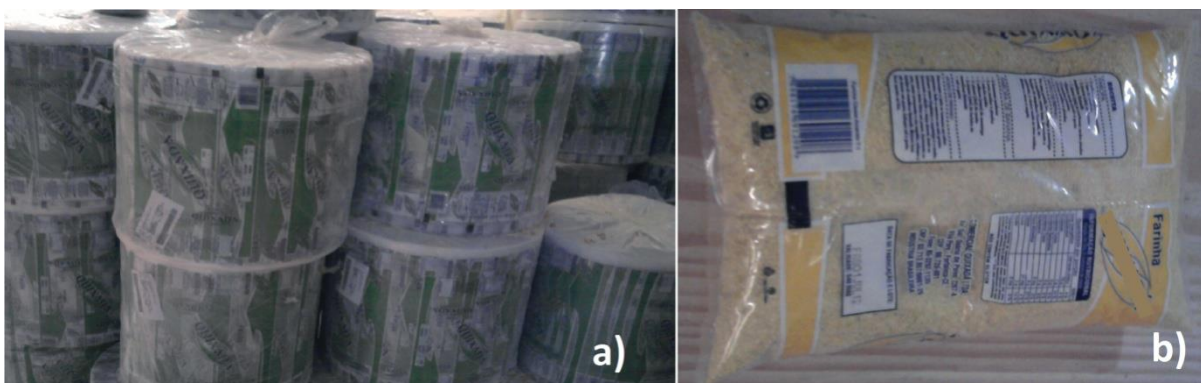
Pelo manual do fabricante, a tensão elétrica utilizada para alimentar a MAE é de 380 V de linha ou 220/380 V trifásico. Em casos específicos de mudanças dessa configuração deve ser informado ao fabricante durante o processo de compra para que sejam feitas as devidas modificações.

Para maiores informações a respeito desse equipamento pode ser obtido pelo site do fabricante (<http://www.norimaq.com.br/>)

##### **3.1.1. Embalagem**

A embalagem oriunda da indústria de embalagens vem em formato de rolo, filme plástico, sendo seu uso destinado a MAE, onde possuem custo reduzido em relação a outros tipos de embalagem que já possam vir com soldas, onde geralmente são utilizadas em processos de empacotamento semiautomático (Figura 9), a) mostra a embalagem que sai da indústria de embalagens, b) mostra o produto já embalado, como é normalmente encontrado.

Figura 9 - a) Embalagem em filme b) Embalagem transformada.



Fonte: Autor.

Porém para essa transformação é necessário que a MAE efetue as soldas no filme, essas soldas são principalmente de dois tipos a solda horizontal e a solda vertical, podendo haver produtos com soldas diversificadas, que por possuírem o mesmo princípio de funcionamento não serão comentadas neste trabalho. O posicionamento da solda vertical e horizontal na embalagem (Figura 10).

Figura 10 - Disposição das soldas na embalagem.



Fonte: Autor.

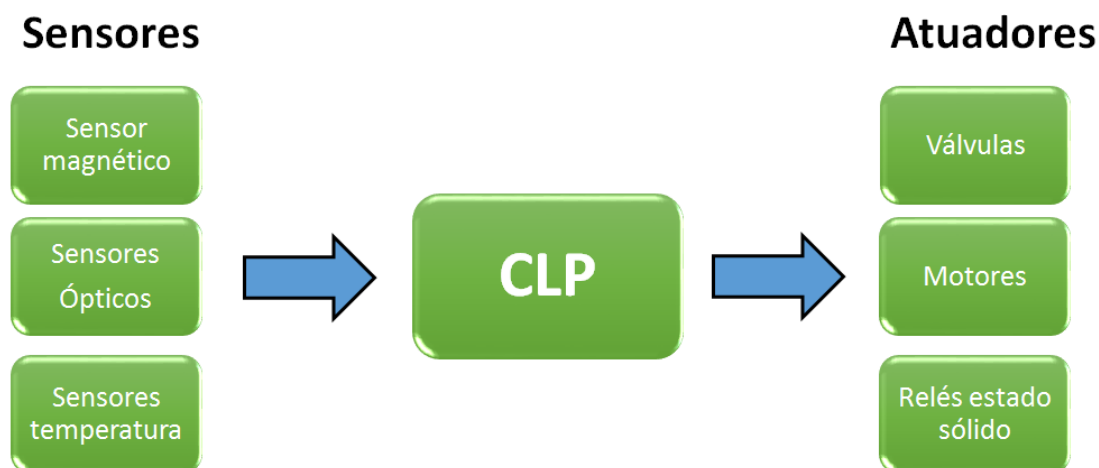
Existem várias técnicas de soldagem, podem ser citadas: selagem dielétrica, soldagem térmica, soldagem por vibração, soldagem por ultrassom, entre outros (LEMOS, 2009).

O processo de soldagem utilizado na MAE ocorre usando o método térmico, onde com o aquecimento de uma resistência elétrica transfere-se temperatura ao filme, que executa a fusão entre as superfícies do filme.

### 3.2. Utilização do controlador lógico programável (CLP)

O CLP é um equipamento bem difundido no meio industrial. Na MAE o CLP é utilizado para receber os sinais dos sensores e conforme sua programação interna tomar as devidas decisões sobre os atuadores. A Figura 11 mostra de forma genérica o função do CLP na MAE.

Figura 11 - Diagrama de funcionamento do CLP na MAE.



Fonte: Autor.

Atualmente está sendo utilizado um CLP da fabricante Schneider Eletric modelo Atos Expert. A interface Homem máquina do CLP utilizado pode ser visualizado na Figura 12.

Figura 12 - Interface Homem Máquina do CLP.



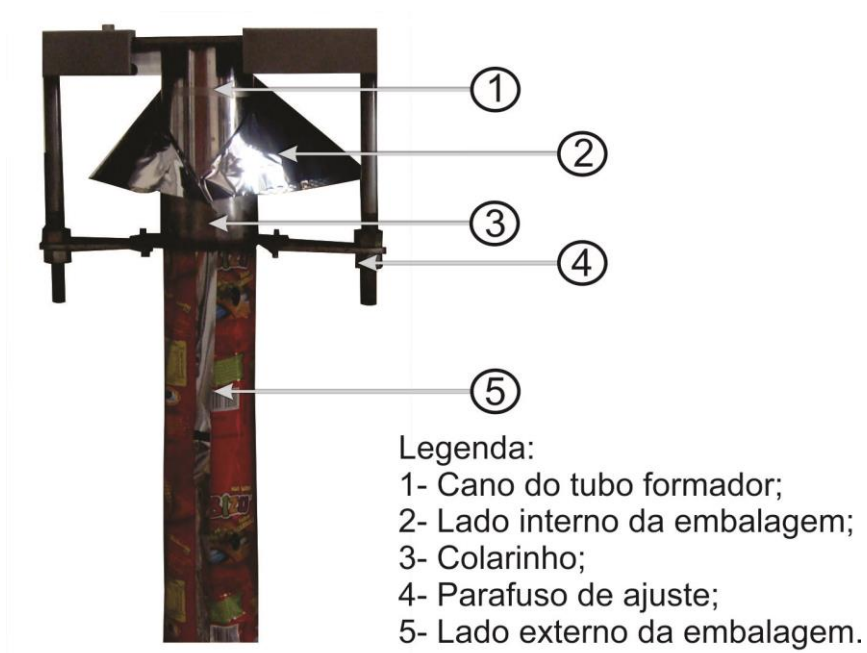
Fonte (autor)

Para maiores informações a respeito desse CLP pode ser verificado no site do fabricante Schineider, conforme segue o link ([http://download.schneider-electric.com/files?p\\_File\\_Id=2465951&p\\_File\\_Name=ma00600\\_0510\\_Atos\\_Expert\\_BF.pdf](http://download.schneider-electric.com/files?p_File_Id=2465951&p_File_Name=ma00600_0510_Atos_Expert_BF.pdf))

### 3.3. Funcionamento do tubo formador

O tubo formador é responsável por dar forma à embalagem, além de servir como apoio para a solda vertical. A Figura 13 mostra as partes integrantes do tudo formador.

Figura 13 - Partes integrantes do tubo formador.



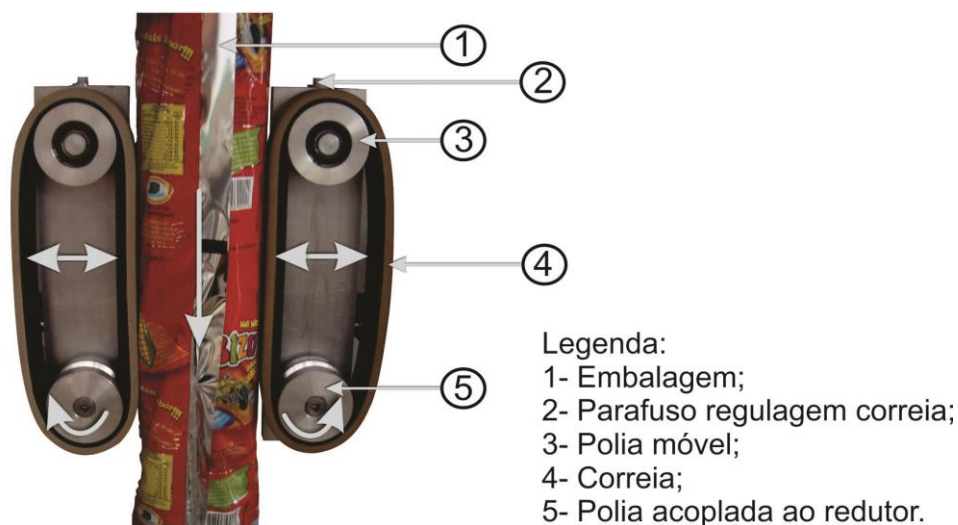
Fonte: Autor.

O tubo formador recebe a embalagem dos rolos, e o colarinho faz com que a embalagem seja envolta no cano do tubo formador, ficando as extremidades do filme sobrepostas onde receberam a solda horizontal. No interior do tubo formador cai o produto proveniente do dosador, para dentro da embalagem.

#### 3.4. Funcionamento do puxador

Após o produto embalado é necessário que a embalagem seja reposicionada para novo empacotamento. O posicionamento da embalagem é feita pelo puxador. A Figura 14 mostra o diagrama do puxador pela vista frontal.

Figura 14 - Diagrama do puxador.



Fonte: Autor.

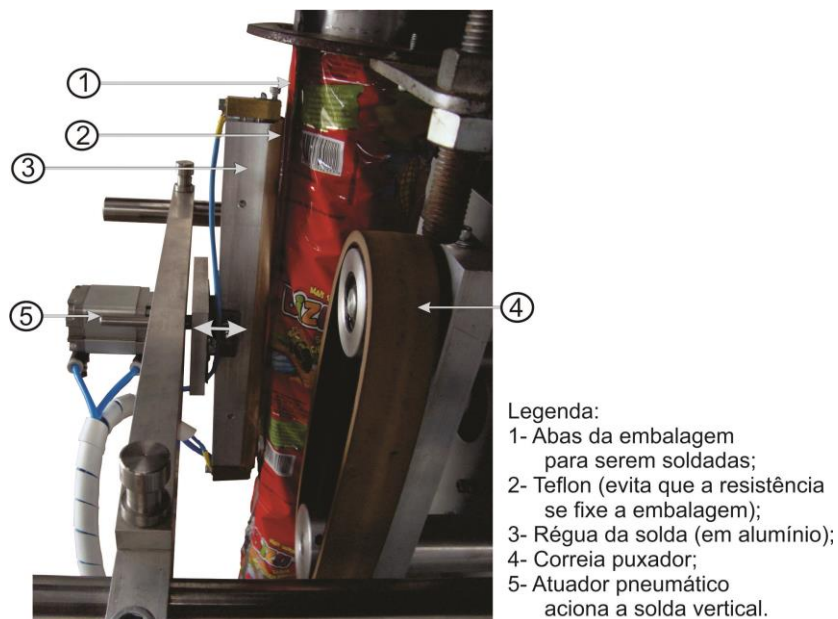
A embalagem é puxada devido ao atrito das correias com a embalagem, onde as correias efetuam o movimento de rotação, que é produzido por um conjunto de motor e redutor interno na MAE para cada correia. As correias do puxador também possuem o movimento horizontal que é utilizado nas manutenções e troca de embalagem, além de garantir o contato das correias com a embalagem.

### 3.5. Funcionamento da Solda vertical

A solda vertical é responsável por soldar as extremidades da embalagem, efetuando assim a solda vertical. A Figura 15 mostra a vista lateral das partes integrantes da solda vertical.



Figura 15 - Solda vertical.



Fonte: Autor.

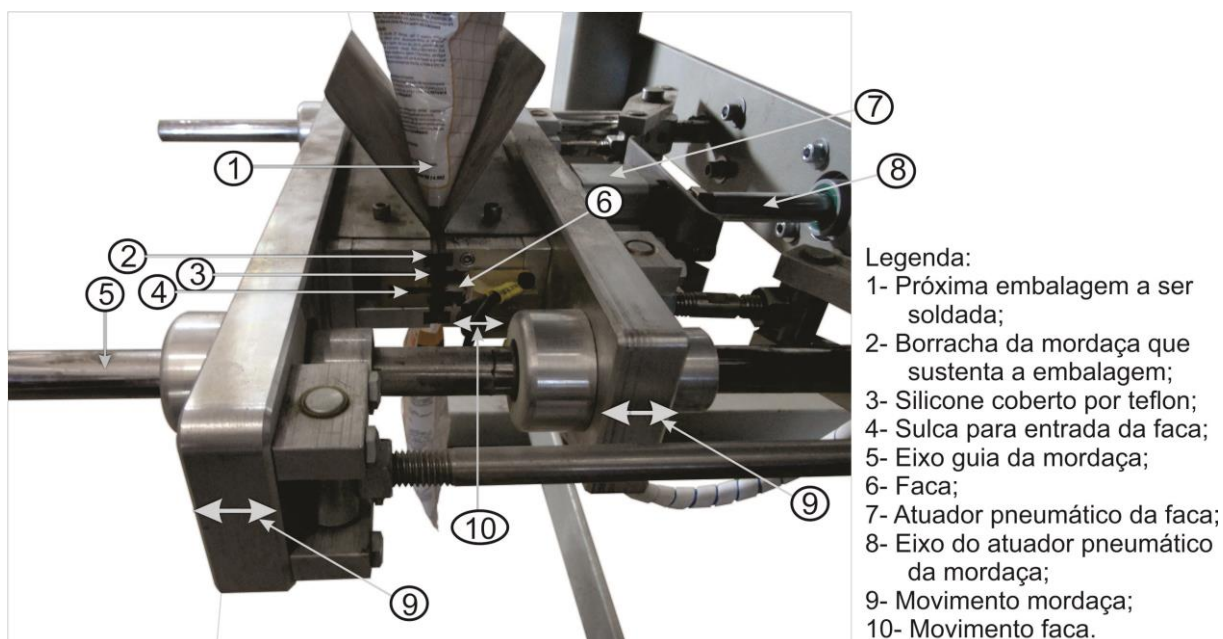
Quando o atuador é acionado (avançado), a régua da solda vertical é movimentada e a temperatura gerada pela resistência faz com que as extremidades da embalagem sejam soldadas. A resistência não pode ser observada na Figura 15, pois está coberta pelo teflon.

### 3.6. Funcionamento da mordaza e faca

O filme de embalagem vem em formato de rolo, como mostrado na Figura 9 a), ou seja, em apenas um rolo estão contidas as embalagens de vários produtos, logo essas várias embalagens precisam ser separadas. A mordaza tem por função servir de apoio, ou seja, manter a embalagem esticada durante o corte e a soldagem horizontal da embalagem (Figura 16).



Figura 16 - Diagrama da mordça.



Fonte: Autor.

Após o fechamento da mordça, o atuador da faca faz com que esta pressione a extremidade da embalagem, cortando-a e soldando-a. Posteriormente, a mordça retorna, descartando assim o produto embalado e aguardando o próximo. Esse movimento é indicado pelas duas setas maiores indicadas pelo número 9.

O atuador da mordça possui grandes dimensões e força, pois caso a embalagem escorregue durante o corte, a faca apenas empurra a embalagem e não a corta.

As resistências das soldas estão situadas na mesma peça em que está a faca. As resistências e o silicone estão cobertos pelo teflon, onde o silicone é utilizado como apoio para a solda.

Em apenas um movimento ocorre o corte da embalagem e duas soldas simultaneamente, a solda superior da embalagem que está abaixo e a solda inferior da embalagem que está acima. O movimento da faca e as soldas são mostrados pela setas menores indicadas pelo número 10 na Figura 17.

### 3.7. Funcionamento dos resfriamentos

Posterior as soldas (vertical e horizontal), são utilizados os resfriamentos, tendo por função resfriar rapidamente a solda, possibilitando que o próximo produto possa ser embalado, acelerando a produção (Figura 17).

Figura 17 - Resfriadores.



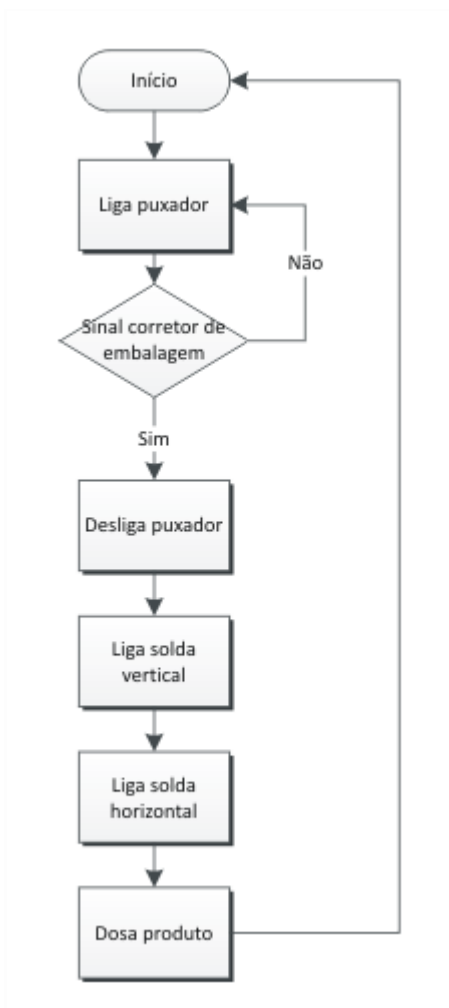
Fonte: Autor.

Para os resfriamentos é utilizado o mesmo ar comprimido proveniente do compressor. Ou seja, os resfriamentos utilizam uma quantidade significativa de ar comprimido.

### 3.8. Funcionamento geral da máquina

A Figura 18 mostra a ordem do funcionamento dos atuadores e motores da MAE, sendo todo o ciclo de funcionamento conforme a programação contida no CLP.

Figura 18 - Diagrama de blocos simplificado do funcionamento da MAE.



Fonte: Autor.

Os parâmetros do CLP são configurados pelo operador conforme as necessidades de operação, dependendo das condições do ambiente e do produto a ser embalado. São exemplos de parâmetros que podem ser modificados pelo operador: tempo das soldas, quantidade de pacotes por segundo, tempo dos atuadores, entre outros.

## 4. PROJETO DO BLOCO DE CONTROLE

### 4.1. Visão Geral

Devido surgimento frequente de novas tecnologias, consumidores cada vez mais exigentes, além da necessidade de melhorias no processo foi visto a necessidade de substituir o CLP existente, por outro com tela de maior tamanho, além de maior número de entradas e saídas, possibilidade de montar um sistema de supervisão, comunicar com outros equipamentos da mesma linha de processo, além de outras necessidades.

Porém um CLP com as configurações necessárias para as necessidades possui um custo elevado, devido a isso foram pesquisadas alternativas de menor valor aquisitivo, mas que atendesse as necessidades citadas, nos tópicos seguintes são citados componentes que foram estudados durante o desenvolvimento desse projeto.

#### 4.1.1. Microchip PIC 18F4550

O PIC 18F4550 é um Controlador Integrado de Periféricos PIC, fabricado pela Microchip, esse microcontrolador possui em resumo as seguintes características:

- Compatível com USB 2.0;
- Três interrupções externas;
- Três timers (T0 ao T3);
- Dois módulos de Captura, Comparação e PWM CCP;
- Conversor analógico digital com resolução de 10 bits.
- Arquitetura Harvard, tecnologia RISC.

A Figura 19 mostra um comparativo entre microcontroladores da mesma família.

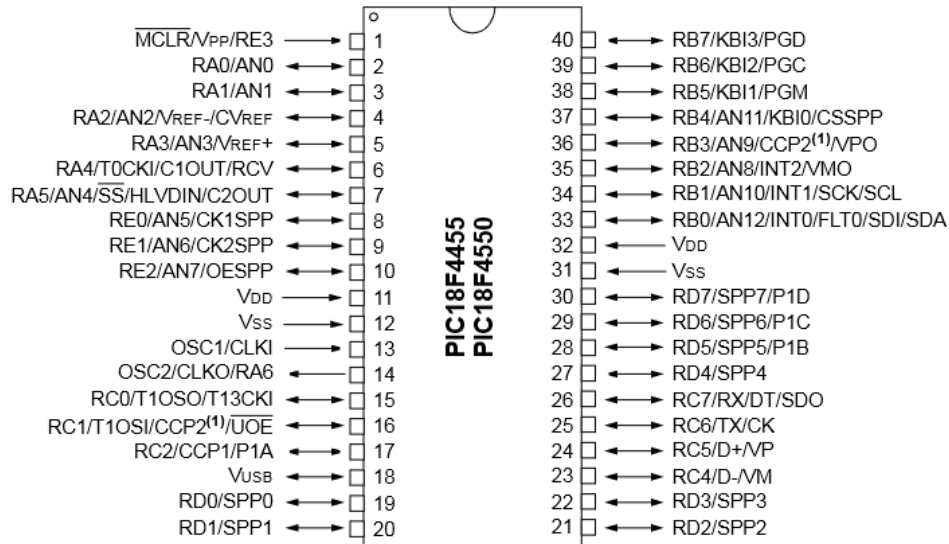
Figura 19 – Tabela com características do PIC18F4550.

Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I <sup>2</sup> C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

Fonte: (Microchip).

A Figura 20 mostra o diagrama de pinos do PIC18F4550 com 40 pinos e encapsulamento PDIP.

Figura 20 - Diagrama de pinos do PIC18F4550 (Microchip).



Fonte: Autor.

#### 4.1.2. Cubieboard 2

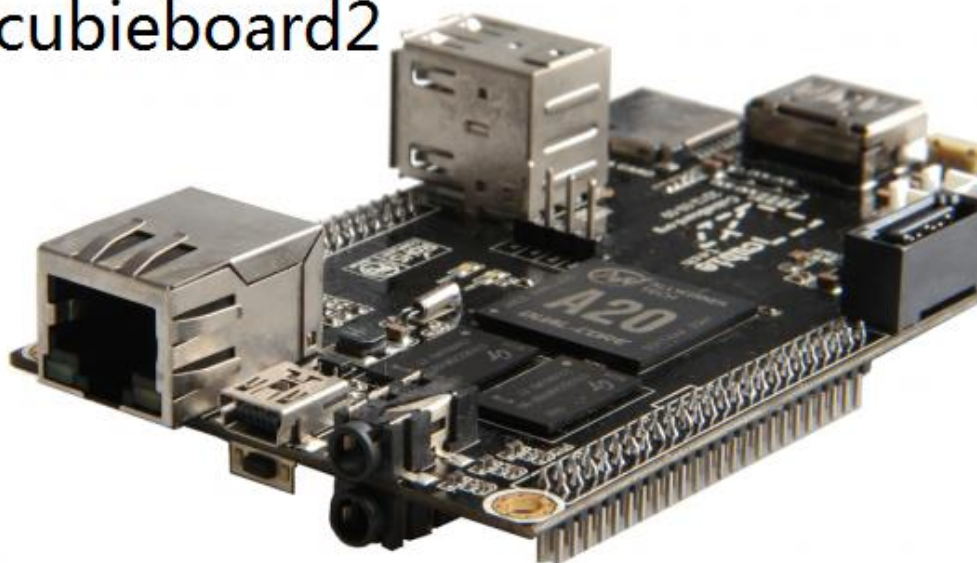
É um microcomputador portátil em uma única placa, segue algumas características de hardware:

- CPU - ARM® Cortex™-A7 Dual-Core
- GPU - ARM® Mali400MP2, Complies with OpenGL ES 2.0/1.1
- Memória - 1GB DDR3 @960M
- Disco rígido - 4GB internal NAND flash, up to 64GB on uSD slot, up to 2T on 2.5 SATA disk
- Rede - 10/100 ethernet, optional wifi
- USB - Two USB 2.0 HOST, one USB 2.0 OTG
- GPIO - 96 extend pin including I2C, SPI, RGB/LVDS, CSI/TS, FM-IN, ADC, CVBS, VGA, SPDIF-OUT, R-TP

A Figura 21 mostra a Cubieboard 2 e seus periféricos.

Figura 21 - Cubieboard 2.

## cubieboard2



Fonte: <http://cubieboard.org/model/>.

### 4.1.3. Arduino Mega 2560

O Arduino Mega 2560 é uma placa com microcontrolador ATmega2560 do fabricante Atmel, segue algumas características:

- Barramento de 8 bits;
- Frequência de clock de 16 MHz;
- 54 entradas/saídas de uso geral, onde 15 podem ser utilizadas para PWM;
- 16 entradas analógicas;
- 4 UARTs (Transmissão/Recepção Universal Assíncrona).

Figura 22 - Arduino Mega 2560.



Fonte: <https://www.arduino.cc/en/Main/ArduinoBoardMega2560>.

#### 4.1.4. Raspberry PI 2 B

É um microcomputador semelhante a Cubieboard 2, segue algumas características de hardware:

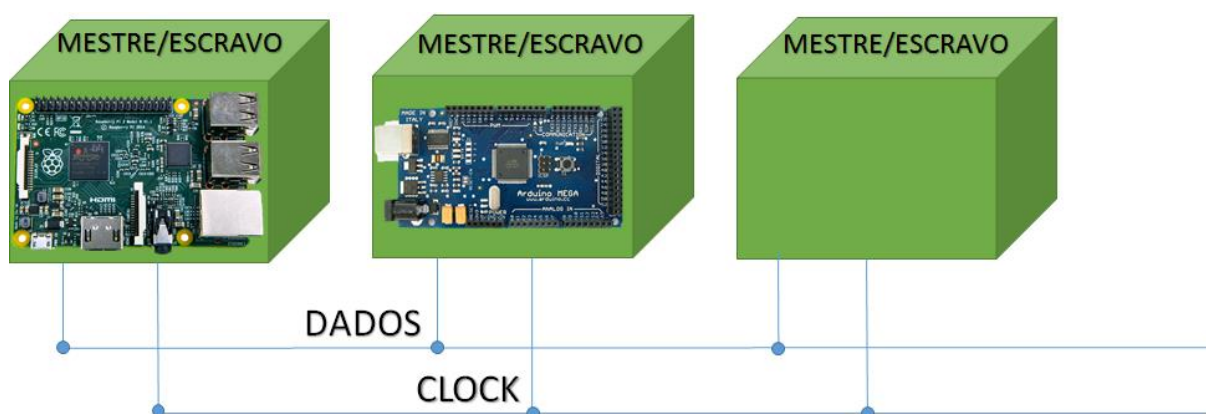
- CPU - 900MHz quad-core ARM Cortex-A7 CPU
- Memória - 1GB RAM
- USB - 4 portas USB
- Rede – Porta Ethernet
- GPIO - 40 GPIO pins

#### 4.2. Protocolo I<sup>2</sup>C (Inter-Integrated Circuit)

Protocolo de comunicação desenvolvido pela Philips, é um tipo de comunicação serial síncrona, que utiliza duas vias de comunicação, admite mais de um mestre no mesmo barramento e possibilita até 128 dispositivos conectados ao mesmo barramento (Junior, 2012).

A Figura 23 mostra de forma simplificada um exemplo de conexão entre vários dispositivos por intermédio do protocolo I<sup>2</sup>C.

Figura 23 – Exemplo ilustrativo do protocolo de comunicação I<sup>2</sup>C.



Fonte: Autor.

#### 4.3. Windows 10 IoT (Internet Of Things)

O Windows 10 IoT Internet das Coisas, é uma proposta da Microsoft para integrar dispositivos, sendo disponibilizada uma versão de sistema operacional para microcomputadores, tendo suporte para: Raspberry Pi 2, Seta DragonBoard 410c e MinnowBoard MAX. A Microsoft, também vem com a proposta do UWP (Universal



Windows Platform), Plataforma Universal Windows, onde o mesmo aplicativo desenvolvido pode ser utilizado em diversos dispositivos (Microsoft).

A Figura 24 mostra o um resumo da proposta da sua plataforma universal desenvolvida pela Microsoft.

*Figura 24 - Slogam UWP.*



*Fonte: <http://ms-iot.github.io/content/en-US/IoTCore.htm>.*

#### **4.4. Componentes selecionados**

A princípio foi cogitado a utilização do PIC 18F4550, por já ter utilizado esse e outros microcontroladores da linha PIC, porém foi observado limitações quanto a quantidade de pinos e processamento, ficando o projeto final bem semelhante ao já existente.

Posteriormente foi migrado para a Cubieboard 2, pois possuía um dos melhores processamentos da época, comparado com os microcomputadores existentes além das 96 entradas e saídas digitais disponíveis, comparada com 26 entradas e saídas digitais do Raspberry Pi B. Porém foram encontrada grandes dificuldades de documentação até mesmo para coisas básicas, como instalar um sistema operacional.

Mesmo com as dificuldades foi desenvolvida uma interface com intermédio da linguagem Python, com auxílio do meu colega David Borges, para acionar as entradas e saídas nativas, foi de consenso em utilizar a Cubieboard 2 apenas para rodar o sistema operacional, pois o programa desenvolvido para tratar as entradas e



saídas poderiam gerar atrasos, devido a isso foi dividido a parte de alto nível e parte de baixo nível.

A Cubieboard 2 sendo responsável pelo alto nível: rodar o sistema operacional, rodar o programa, exibir informações na tela, entre outros. Além de outro dispositivo responsável apenas por verificar as entradas e acionar as saídas.

Baseado nessa nova topologia foi desenvolvida um novo programa onde a Cubieboard se comunicava com um PIC via serial assíncrona. Porém surgiram dificuldades para um protocolo interno de forma a identificar falhas durante a comunicação, mais especificamente na sequência dos bytes, identificar os bytes de endereço e os bytes de comando, caso parte da informação fosse perdida, ou alguma outra anomalia.

Foi feita a aquisição do Raspberry Pi 2, que já possuía grandes melhorias em relação ao modelo B anterior, além da possibilidade de rodar o Windows IoT, que ainda não havia sido lançado. No Raspberry Pi 2 foi utilizado o sistema operacional Raspbian juntamente com o Python. Pouco tempo depois o David Borges, que me auxiliava na programação em Python não pode mais me ajudar.

Depois da aquisição do Raspberry Pi 2 a Microsoft lança o Windows IOT, o qual foi testado e está sendo utilizado até hoje, mesmo sendo uma ferramenta recente é bem amigável. Outra vantagem muito boa para esse processo é que quando o aplicativo é desenvolvido e colocado para rodar no Raspberry Pi 2, o aplicativo fica como se ele fosse o próprio sistema operacional da placa. Comparando com o Raspbian onde seria necessário depois de iniciar o sistema operacional o operador abrir o aplicativo, com o Windows IOT torna a aplicação bem mais específica.

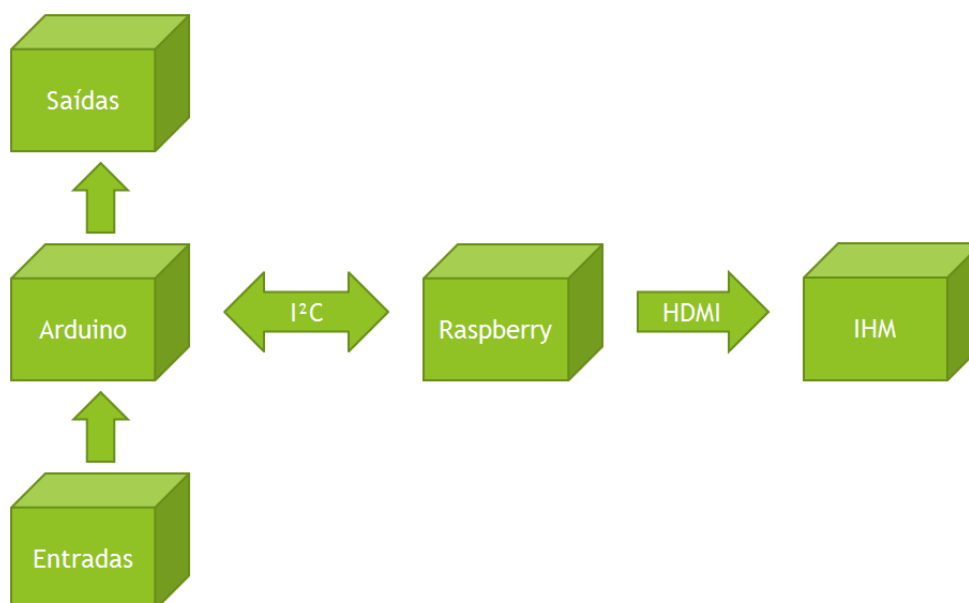
Na parte do baixo nível o PIC foi substituído pelo Arduino Mega 2560, devido a quantidade de entradas e saídas disponíveis, além de vir em uma placa já pronta e com grande referência inclusive em português. Mesmo sendo uma ferramenta que não havia trabalhado, a adaptação decorreu de forma bem rápida e satisfatória.

#### 4.5. Topologia Atual

É utilizado o Windows IoT no Raspberry, juntamente com o Arduino, a programação do Raspberry é feita por intermédio do Visual Studio 2015, onde são utilizadas as linguagens XAML e C#, já o Arduino é programado por seu programa obtido no site oficial.

Devido aos exemplos disponíveis e a possibilidade de conexão de diversos dispositivos no mesmo barramento e o reduzido número de vias, optou-se pelo protocolo I<sup>2</sup>C, sendo o protocolo mais complexo que uma comunicação serial assíncrona ou o protocolo SPI, porém utiliza menos vias que o protocolo SPI e quando comparado com a comunicação serial assíncrona que é ponto a ponto possibilita até 128 dispositivos no mesmo barramento, ou seja oferece um bom custo benefício. A Figura 25 mostra a topologia atual utilizada.

Figura 25 - Topologia atual.



Fonte: Autor.

#### 4.6. Código utilizado no Arduino

A princípio foi definida a função de configuração do Timer 1, essa biblioteca pode ser baixada no site oficial pelo link (<http://playground.arduino.cc/Code/Timer1>), neste site também é disponibilizado a biblioteca para o Timer 3. O Arduino possui 6 Timers o site do Laboratório de Garagem também disponibiliza biblioteca para os outros Timers, além de explicar o funcionamento qual função de cada Timer, pelo link

<http://labdegaragem.com/profiles/blogs/tutorial-executando-fun-es-em-intervalos-de-tempo-fixos-timers>), porém essas bibliotecas não foram testadas.

A biblioteca Wire.h já vem instalada e só é preciso chama-la, essa biblioteca auxilia na comunicação I<sup>2</sup>C, para maiores informações do funcionamento, além de exemplos pode ser acessado o link (<https://www.arduino.cc/en/Reference/Wire>).

Posteriormente são definidas variáveis e constantes que no decorrer do código serão entendidas de forma mais clara.

Salientando que para declaração de vetores, a quantidade especificada indica quantas lacunas o vetor vai possuir, porém os índices do vetor iniciam a contagem por 0 (<https://www.arduino.cc/en/Reference/Array>). Ou seja o VetorEscrita[3], possui 3 elementos, sendo eles VetorEscrita[0], VetorEscrita[1], VetorEscrita[2]. Caso seja chamado o VetorEscrita[3] será tido como resposta lixo da memória.

A Figura 26 mostra o cabeçalho utilizado no código do Arduino

Figura 26 – Cabeçalho do código.

```

1 ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
2 // Implementação de protocolo de comunicação entre arduino e raspberry com I2C//
3 // Utilizando W IOT 05/01/16 //
4 // Rodolfo M. L. de Santana TCC engenharia elétrica //
5 // UFC Universidade Federal do Ceará //
6 // Orientador Rômulo Nunes //
7 ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
8
9 #include <TimerOne.h> // INCLUI BIBLIOTECA TIMER
10 #include <Wire.h> // INCLUI BIBLIOTECA I2C
11
12 int cont=0; // DEFINE VARIÁVEL INCREMENTO TIMER
13 byte leitura; // DEFINE VARIÁVEL RESPOSTA DO MESTRE
14 int VetorEscrita [3]; // DEFINE VETOR QUE RECEBE RECEBE DADOS DO MESTRE
15 byte VetorDados [30]; // DEFINE VETOR PRINCIPAL DE CONFIGURAÇÃO DA MÁQUINA
16
17 // DEFINIÇÃO DO PINOS QUE SERÃO UTILIZADOS
18 const int Mordaca = 4;
19 const int Vertical = 5;
20 const int Horizontal = 6;
21 const int Dosador = 7;
22 const int Puxador = 8;
23 const int Datador = 9;
24 const int Resfriamento = 10;
25 const int Reserva = 11;
26 const int Led = 13;

```

Fonte: Autor.

A função setup é utilizada para configurações dos registradores sendo utilizada apenas uma vez na inicialização do Arduino, a princípio foram definidos os pinos utilizados como saída, posteriormente é configurado o tempo de estouro do Timer 1, para intervalos de 10 ms, ou seja a cada 10 ms ocorre um estouro no Timer que por consequência chama a função CicloMaquina.

Logo após é configurado a velocidade da comunicação serial, sendo esta muito útil identificação de defeitos e análise do funcionamento em tempo real pelo monitor serial da IDE do Arduino. Posteriormente são realizadas as configurações do I<sup>2</sup>C, onde é definido o endereço do dispositivo e uma função para quando o mestre quiser requisitar algo do Arduino e quando quiser escrever no Arduino.

A Figura 27 mostra o trecho referente a função setup.

Figura 27 - Função setup.

```

28 void setup()
29 {
30     // Definição dos pinos
31     pinMode(Mordaca, OUTPUT);
32     pinMode(Vertical, OUTPUT);
33     pinMode(Horizontal, OUTPUT);
34     pinMode(Dosador, OUTPUT);
35     pinMode(Puxador, OUTPUT);
36     pinMode(Datador, OUTPUT);
37     pinMode(Resfriamento, OUTPUT);
38     pinMode(Reserva, OUTPUT);
39     pinMode(Led, OUTPUT);
40
41     // Configuração do timer
42     Timer1.initialize(10000);           // Configura o timer para estouros de 10 ms
43     Timer1.attachInterrupt(CicloMaquina); // Chama função de interrupção do timer
44
45     // Configura a velocidades da comunicação serial
46     Serial.begin(38400);
47
48     // Configura a utilização do I2C
49     Wire.begin(8);                     // Endereço desse dispositivo I2C
50     Wire.onRequest(Requisicao);         // Mestre requisita valor desse escravo
51     Wire.onReceive(Recebimento);      // Mestre faz uma escrita no escravo
52 }

```

Fonte: Autor.

A função “loop” é a função de repetição do Arduino, nela é feito a comparação dos valores incrementados na função “CicloMaquina” com os valores armazenados no “VetorDados” alterando assim o nível lógico das saídas quando necessário. Tomando por exemplo a Mordaca, quando “cont” for igual a “VetorDados[4]” a Mordaca é acionada; quando for igual a “VetorDados[5]” ela é desligada. A Figura 28 mostra a função “loop”.

Figura 28 Função “loop”.

```

63 void loop()
64 {
65     //MORDAÇA
66     if(cont==VetorDados[4]){digitalWrite(Mordaca, HIGH);}// ACIONA MORDAÇA
67     if(cont==VetorDados[5]){digitalWrite(Mordaca, LOW);} // DESLIGA MORDAÇA
68
69     //VERTICAL
70     if(cont==VetorDados[6]){digitalWrite(Vertical, HIGH);}// ACIONA SOLDA VERTICA
71     if(cont==VetorDados[7]){digitalWrite(Vertical, LOW);} // DESLIGA SOLDA VERTICAL
72
73     //HORIZONTAL
74     if(cont==VetorDados[8]){digitalWrite(Horizontal, HIGH);}// ACIONA HORIZONTAL
75     if(cont==VetorDados[9]){digitalWrite(Horizontal, LOW);} // DESLIGA HORIZONTAL
76
77     //DOSADOR
78     if(cont==VetorDados[10]){digitalWrite(Dosador, HIGH);}// ACIONA DOSADOR
79     if(cont==VetorDados[11]){digitalWrite(Dosador, LOW);} // DESLIGA DOSADOR
80
81     //TRACIONADOR
82     if(cont==VetorDados[12]){digitalWrite(Puxador, HIGH);}// ACIONA TRACIONADOR
83     if(cont==VetorDados[13]){digitalWrite(Puxador, LOW);} // DESLIGA TRACIONADOR
84
85     //DATADOR
86     if(cont==VetorDados[14]){digitalWrite(Datador, HIGH);}// ACIONA DATADOR
87     if(cont==VetorDados[15]){digitalWrite(Datador, LOW);} // DESLIGA DATADOR
88
89     //RESFRIAMENTO
90     if(cont==VetorDados[16]){digitalWrite(Resfriamento, HIGH);}// ACIONA RESFRIAMENTO
91     if(cont==VetorDados[17]){digitalWrite(Resfriamento, LOW);} // DESLIGA RESFRIAMENTO
92
93     //RESERVA
94     if(cont==VetorDados[18]){digitalWrite(Reserva, HIGH);}// ACIONA RESERVA
95     if(cont==VetorDados[19]){digitalWrite(Reserva, LOW);} // DESLIGA RESERVA
96
97     //LED
98     if(cont==VetorDados[20]){digitalWrite(Led, HIGH);}// ACIONA LED
99     if(cont==VetorDados[21]){digitalWrite(Led, LOW);} // DESLIGA LED
100 }

```

*Fonte: Autor.*

Em modelos mais antigos de outros fabricantes a MAE possui um encoder acoplado no motor principal, de forma a facilitar a adaptação do operador a função “CicloMáquina” simula um encoder,

A função “CicloMáquina” é chamada a cada estouro do Timer, conforme configurado no “setup” e possui funcionamento bem simples, cada vez que ela é chamada a variável “cont” é incrementada em uma unidade, caso o valor de “cont” seja igual ao valor de “VetorDados[2]”, “cont” recebe o valor de “VetorDados[1]”.

Onde o operador pode variar o início da contagem (“VetorDados[1]”) e termino da contagem (“VetorDados[2]”), porém o tempo de incremento é fixo em 10 ms, ou seja caso seja necessário aumentar o tempo de ciclo total da máquina deve ser aumentado a quantidade de incrementos, o valor é limitado por software em 250 incrementos, ou seja 2,5 segundos máximo de ciclo.

A Figura 29 mostra o trecho referente a função CicloMaquina

*Figura 29 - Função CicloMaquina.*

```

54 void CicloMaquina(void)
55 {
56     cont=cont+1; // Incrementa a contagem
57     if(cont==VetorDados[2])
58     {
59         cont=VetorDados[1];
60     }
61 }

```

*Fonte: Autor.*

Na comunicação I<sup>2</sup>C o protocolo interno o Arduino espera sempre 3 bytes, A Tabela 1 mostra a sequência de bytes esperado no comandos de escrita e leitura.

*Tabela 1 - Sequencia de bytes protocolo interno.*

	Byte 1	Byte 2	Byte 3
<i>Escrita</i>	101	Endereço	Valor
<i>Leitura</i>	108	Endereço	108

Quando ocorre a comunicação I<sup>2</sup>C o buffer armazena os dados recebidos, quando a quantidade de dados é maior que 2, esses dados são tratados, sendo armazenados no vetor “VetorEscrita[]” e posteriormente é verificado se é uma requisição de leitura, ou se é uma escrita. Caso seja escrita o valor recebido é atualizado no “VetorDados[]”, no respectivo endereço indicado, caso seja leitura o endereço é preparado para que o dado requerido seja enviado na próxima requisição.

Os dados recebidos no I<sup>2</sup>C são enviados pela serial, para possibilitar a visualização em tempo real e possibilitar encontrar de forma mais rápida a causa raiz de algum problema, essa forma de diagnóstico é bem útil, principalmente nos no início do desenvolvimento dos códigos. A Figura 30 mostra a função Recebimento.

Figura 30 - Função Recebimento.

```

104 void Recebimento(int howMany)
105 {
106     if(Wire.available() > 2) // se tem mais de 2 bytes no buffer inicia a leitura
107     {
108         int x = 0;
109         while (0 < Wire.available())
110         {
111             VetorEscrita [x] = Wire.read();
112             Serial.print(VetorEscrita[x]);
113             x = x+1;
114         }
115         Serial.println(" ");
116     }
117
118     // PROTOCOLO ESCRITA
119     // 101 (confirmação) - endereço - byte 1.
120     if(VetorEscrita[0] == 101) // letra "e" tabela asc2, ESCRITA
121     {
122         VetorDados[VetorEscrita[1]] = VetorEscrita [2]; // Escreve no vetor dados
123     }
124
125     // PROTOCOLO DE LEITURA
126     // 108 (Confirmação de leitura) - endereço de leitura - 108 (Confirmação de leitura)
127     if(VetorEscrita[0] == 108) // letra "l" tabela asc2, LEITURA
128     {
129         leitura=VetorDados[VetorEscrita[1]];
130     }
131 }

```

Fonte: Autor.

Quando o mestre faz um pedido de requisição pelo I<sup>2</sup>C o Arduino responde com o valor contido no último endereço utilizado na função Recebimento. A Figura 31 mostra a função Requisição.

```

134 // RESPONDE COM O DADO REQUISITADO
135 void Requisicao()
136 {
137     Wire.write(leitura);
138 }

```

Figura 31 - Função Requisicao.

#### 4.7. Código no Visual Studio

O Visual Studio é um programa da Microsoft que permite a utilização de diversas linguagens de programação, para a programação no Raspberry Pi 2 foi utilizado o XAML para o desenvolvimento da interface gráfica e o C# para fazer a interação da interface com o Arduino.



#### 4.7.1. XAML

A tela foi desenvolvida de forma a possibilitar uma visualização geral e a alteração rápida de parâmetros. Onde em para cada dispositivo tem seus respectivos tempo de ligar e desligar, nas duas colunas ao lado, para que ocorra a alteração de algum parâmetro o operador deve clicar ou tocar (ainda não implementado) no local que queira alterar e posteriormente acionar os botões para incrementar ou decrementar (-1, +1, -10, +10). Para confirmar a alteração e enviar para o Arduino, deve ser pressionado o botão carregar. A Figura 32 mostra a tela de parametrização.

Figura 32 - Tela de parametrização.

<b>MÁQUINA AUTOMÁTICA DE EMPACOTAR</b>				
INICIO CICLO	0	MORDAÇA	25	50
FINAL CICLO	100	VERTICAL	30	70
<b>CARREGAR</b>		HORIZONTAL	30	45
-1	+1	DOSADOR	65	80
-10	+10	TRACIONADOR	60	90
<b>LIGA</b>		DATADOR	0	15
<b>DESLIGA</b>		RESFRIAMENTO	80	95

Fonte: Autor.

Para construção da tela foram criadas linhas e colunas, de forma a facilitar o posicionamento dos componentes e evitar possíveis sobreposições. A Figura 33 mostra a interface com as linha e colunas.

Figura 33 - Tela com divisão de linhas e colunas.

<b>MÁQUINA AUTOMÁTICA DE EMPACOTAR</b>				
INICIO CICLO	0	MORDAÇA	25	50
FINAL CICLO	100	VERTICAL	30	70
<b>CARREGAR</b>		HORIZONTAL	30	45
-1	+1	DOSADOR	65	80
-10	+10	TRACIONADOR	60	90
<b>LIGA</b>		DATADOR	0	15
<b>DESLIGA</b>		RESFRIAMENTO	80	95

Fonte: Autor.

A Figura 34 mostra o trecho do código que define as linhas e colunas.

Figura 34 - Definição de linha e colunas.

```
<Grid.RowDefinitions>
  <RowDefinition Height="auto"></RowDefinition>
  <RowDefinition Height="*"></RowDefinition>
  <RowDefinition Height="*"></RowDefinition>
  <RowDefinition Height="*"></RowDefinition>
  <RowDefinition Height="*"></RowDefinition>
  <RowDefinition Height="*"></RowDefinition>
  <RowDefinition Height="*"></RowDefinition>
  <RowDefinition Height="auto"></RowDefinition>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
  <ColumnDefinition Width="*"></ColumnDefinition>
  <ColumnDefinition Width="*"></ColumnDefinition>
  <ColumnDefinition Width="auto"></ColumnDefinition>
  <ColumnDefinition Width="auto"></ColumnDefinition>
  <ColumnDefinition Width="auto"></ColumnDefinition>
</Grid.ColumnDefinitions>
```

Fonte: Autor

Posterior a definição das linhas e colunas as lacunas criadas devem ser preenchidas, para exemplificar esse preenchimento será mostrado o preenchimento de parte da linha sete. A primeira e a segunda colunas são mescladas “`ColumnSpan="2"`” sendo o conteúdo centralizado tanto na horizontal quanto na vertical, o tamanho do botão é 200 por 60, possui a palavra LIGA em seu interior com tamanho da fonte 26.667 a espessura da letra é Bold (similar ao negrito), possui a cor de fundo vermelho e o evento associado para o click sobre o botão é BotaoLiga, segue o trecho do código.

```
<!--Linha 07-->
  <Grid Grid.Row="6" Grid.ColumnSpan="2" HorizontalAlignment="Center"
  VerticalAlignment="Center">
    <Button Content="LIGA" Width="200" Height="60"
    HorizontalAlignment="Right" VerticalAlignment="Bottom" Background="Red"
    FontSize="26.667" FontWeight="Bold" Click="BotaoLiga"></Button>
  </Grid>
```

#### 4.7.2. C#

O C# é utilizado para tratar os eventos criados no XAML, realizar operações necessárias, além de realizar a comunicação com o Arduino.

Para inicializar a utilização da comunicação I2C no C# é necessário informar o endereço do escravo, informar a velocidade de comunicação do

barramento entre outras configurações. Ainda não foi estudado o Raspberry operando como escravo. A Figura 35 mostra o trecho do código referente a inicialização da comunicação.

*Figura 35 – Inicialização da comunicação I2C C#.*

```
public sealed partial class MainPage : Page
{
    private const byte EnderecoEscravo = 0x08; /* ENDEREÇO DE 7-bits DO ESCRAVO NO I2C */
    private int Selecao = 0; /* VARIÁVEL DE IDENTIFICAÇÃO DE CAMPO */
    private I2cDevice ARDUINO; /* DECLARA COMUNICAÇÃO I2C */

    public MainPage()
    {
        this.InitializeComponent();
        Unloaded += MainPage_Unloaded;
        InitI2CAccel(); /* INICIALIZAÇÃO DA COMUNICAÇÃO I2C */
    }

    private async void InitI2CAccel()
    {
        string aqs = I2cDevice.GetDeviceSelector();
        var dis = await DeviceInformation.FindAllAsync(aqs);
        if (dis.Count == 0)
        {
            return;
        }

        var settings = new I2cConnectionSettings(EnderecoEscravo);
        settings.BusSpeed = I2cBusSpeed.FastMode;
        ARDUINO = await I2cDevice.FromIdAsync(dis[0].Id, settings);
        if (ARDUINO == null)
        {
            return;
        }
    }
}
```

*Fonte: Autor.*

A Figura 36 mostra a escrita dos bytes no Arduino associado ao evento de click do botão LIGA, conforme foi definido no XAML e já comentado no tópico 494.7.1.

Figura 36 - Escrever no Arduino.

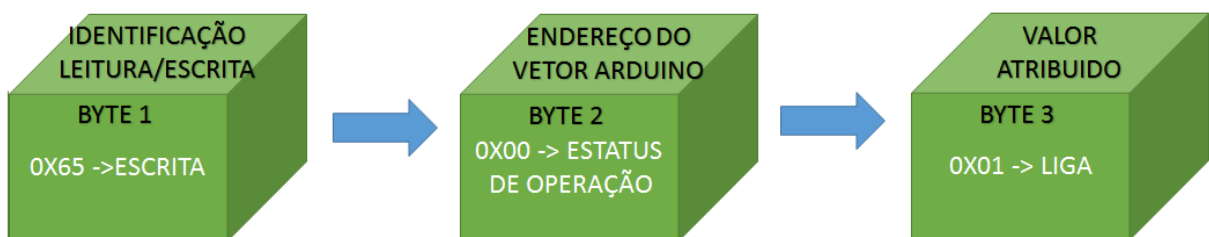
```
private void BotaoDesliga(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x00, 0x01 };

    /* Write the register settings */
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
        //I2CAccel.Write(WriteBuf_PowerControl);
    }
    /* If the write fails display the error and stop running */
    catch (Exception ex)
    {
        return;
    }
}
```

Fonte: Autor.

Para efetuar escrita no Arduino é utilizado o comando `Arduino.Write`, onde os bytes são enviados conforme seu agrupamento nesse trecho citado está sendo enviado os seguintes bytes `0X65`, `0X00` e `0X01`, com essa sequência está sendo alterado o valor da posição 0 do vetor no Arduino para 1. A Figura 37 mostra de forma ilustrativa o procedimento para escrita.

Figura 37 - Sequência de bytes para escrita.



Fonte: Autor.

Para efetuar a alteração de algum parâmetro pelos botões `-1`, `+1`, `-10` e `+10`, primeiro foi necessário identificar qual parâmetro deveria ser alterado, para resolver esse problema foi adicionado o evento `PointerExited` esse evento é semelhante ao evento de `click`, onde quando esse evento é chamado ele altera o valor da variável global `Selecao`, conforme o parâmetro relativo ao evento.

Quando pressionado os botões de incremento, ou decremento é chamado o evento de `click` referente ao botão, nesse evento é analisado o valor contido na variável `Selecao` e então é incrementado ou decrementado o parâmetro que alterou a variável `Selecao`.

A Figura 38 mostra o evento que é chamado quando o parâmetro relativo a INICIO CICLO recebe um click, o valor da variável Selecao é alterado para 1.

*Figura 38 - Evento PointerExited do parâmetro INICIO CICLO.*

```
private void EventoBoxInicial(object sender, Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
{
    Selecao = 1;
}
```

*Fonte: Autor.*

A Figura 39 mostra o evento de click do botão -1, note que é verificado o valor da variável Selecao, para depois receber, decrementar e atualizar no respectivo campo.

*Figura 39 – Evento de click no botão -1.*

```
private void BotaoMenosUm(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    if (Selecao == 1)
    {
        int x = Int32.Parse(BoxInicio.Text); /* CONVERSÃO DE INT32 PARA INT */
        x--; /* DECREMENTO EM UMA UNIDADE */
        if (x < 0) /* LIMITA O VALOR ENTRE 0 E 250 */
        { x = 250; }
        BoxInicio.Text = "" + x; /* ATUALIZA O VALOR NO PARÂMETRO */
    }
}
```

*Fonte: Autor.*

De forma a evitar uma estrutura de repetição periódica, ou mesmo envios desnecessário de informação, toda alteração de parâmetros só é enviada para o Arduino após o click no botão CARREGAR, foi utilizada a mesma variável Selecao para identificar e enviar o parâmetro correto.

A Figura 40 mostra o envio dos 3 bytes referente ao parâmetro início de ciclo com click no botão carrega.

Figura 40 – Evento click botão CARREGAR.

```
private void BotaoCarrega(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    if (Selecao == 1)
    {
        int x = Int32.Parse(BoxInicio.Text);    /* CONVERTE DE INT32 PARA INT */
        byte y = Convert.ToByte(x);            /* CONVERTE EM BYTE E CARREGA EM Y */

        byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x01, y }; /* ORGANIZA OS BYTES */
        try
        {
            ARDUINO.Write(WriteBuf_DataFormat); /* ENVIA OS BYTES PARA O ARDUINO */
        }
        catch (Exception ex)
        {
            return;
        }
    }
}
```

Fonte: Autor.

#### 4.8. Projeto eletrônico

Segundo o item 12.36 da NR-12 – SEGURANÇA NO TRABALHO EM MÁQUINAS E EQUIPAMENTOS com última atualização em, 17/12/2010, segue o trecho:

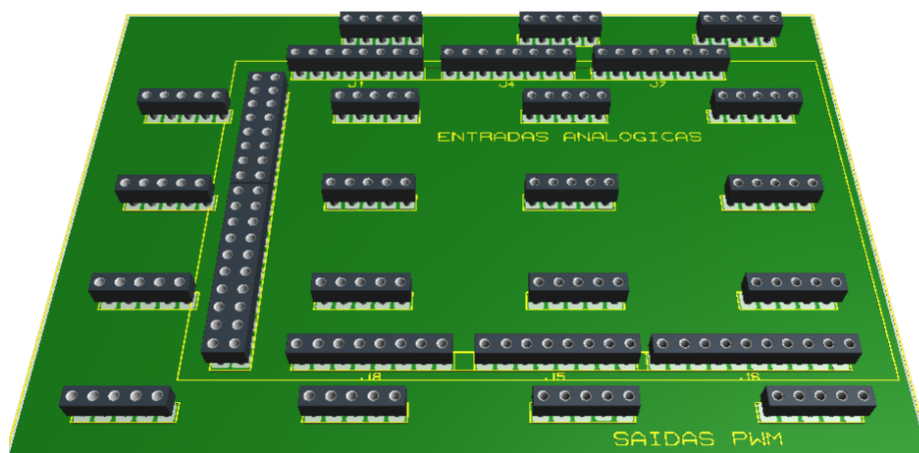
*“12.36. Os componentes de partida, parada, acionamento e outros controles que compõem a interface de operação das máquinas devem:*

*a) operar em extra Baixa tensão de até 25V (vinte e cinco volts) em corrente alternada ou de até 60V (sessenta volts) em corrente contínua;”*

É de uso comum a utilização de 24 V em corrente contínua em equipamentos industriais, ou seja o Arduino opera com tesões da ordem de 5 V, logo não é recomendável o Arduino ter acionamento direto com as cargas e as entradas, sendo necessário interfaces para adequar os níveis de tensão e corrente.

Foi projetado uma placa de circuito impresso, para ser acoplada ao Arduino e receber blocos de dimensões padronizadas. A Figura 41 mostra uma visualização 3d da placa base projetada no software Proteus 8.

Figura 41 - Base para blocos.

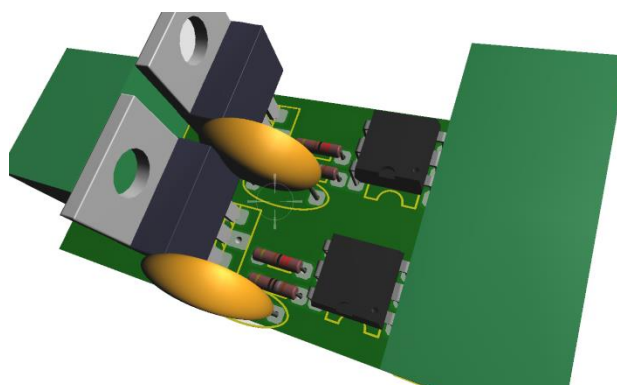


Fonte: Autor.

Os blocos foram desenvolvidos com base em chaves de estado sólido com intuito de obter maior vida útil, sendo utilizado transistores, a entradas e saídas são isoladas por opto-acoplador de forma a evitar a queima dos pinos do Arduino em caso de alguma anomalia.

Para o acionamento de cargas em corrente alternada, podendo ser citado as resistências elétricas que são utilizadas na máquina, foi projetado um bloco com Triacs, porém devido ao nível de tensão utilizado (220 V) e a maior dimensão desse bloco, devido a necessidade de dissipadores. Esse bloco não pode ficar acoplado diretamente a placa base, juntamente com os demais blocos. A Figura 42 mostra o bloco para acionamento de cargas de corrente alternada com dois canais opto-acoplados.

Figura 42 - Bloco para acionamento de cargas com corrente alternada.

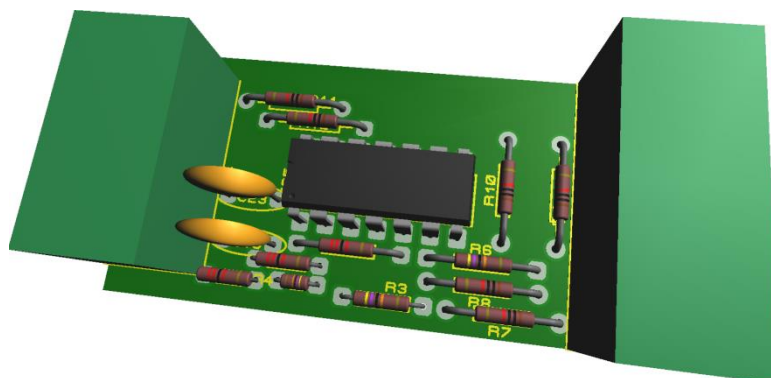


Fonte: Autor.

Para a leitura de entradas analógicas, foi projetado um bloco com amplificador operacional para tornar mensuráveis pelo Arduino pequenos sinais,

podendo ser utilizado principalmente para leitura de termopar e célula de carga. A Figura 43 mostra o bloco desenvolvido com o amplificador operacional.

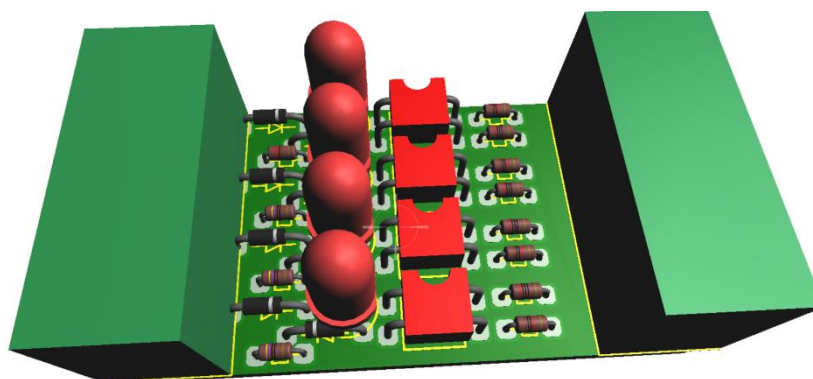
*Figura 43 - Bloco amplificador operacional.*



*Fonte: Autor.*

O bloco de entradas digitais possui suporte para até 4 entradas e foi projetado para operar com tensões na faixa de 24 V, sendo as entradas isoladas por opto-acoplador, além de possuir LEDs que indicam o status de cada entrada de forma individual. A Figura 44 mostra o bloco de entradas digitais.

*Figura 44 - Bloco de entradas digitais.*

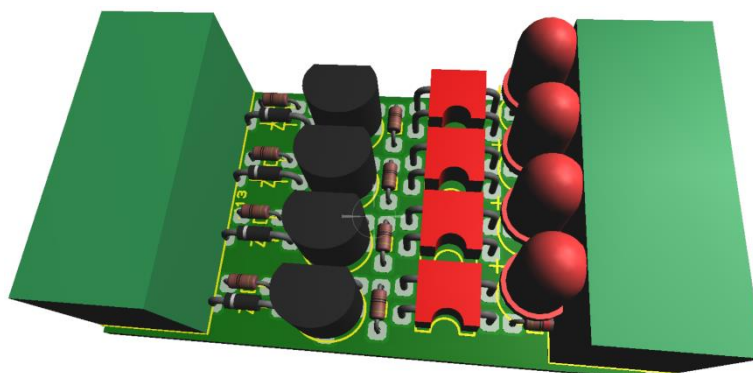


*Fonte: Autor.*

Também foi desenvolvido um bloco para saída digital, este bloco possui 4 canais e foi projetado a para suprir correntes de até 800 mA, cada saída é isolada por opto-acoplador e possui um LED, que indica o status de cada saída. A Figura 45 mostra o bloco de saídas digitais.



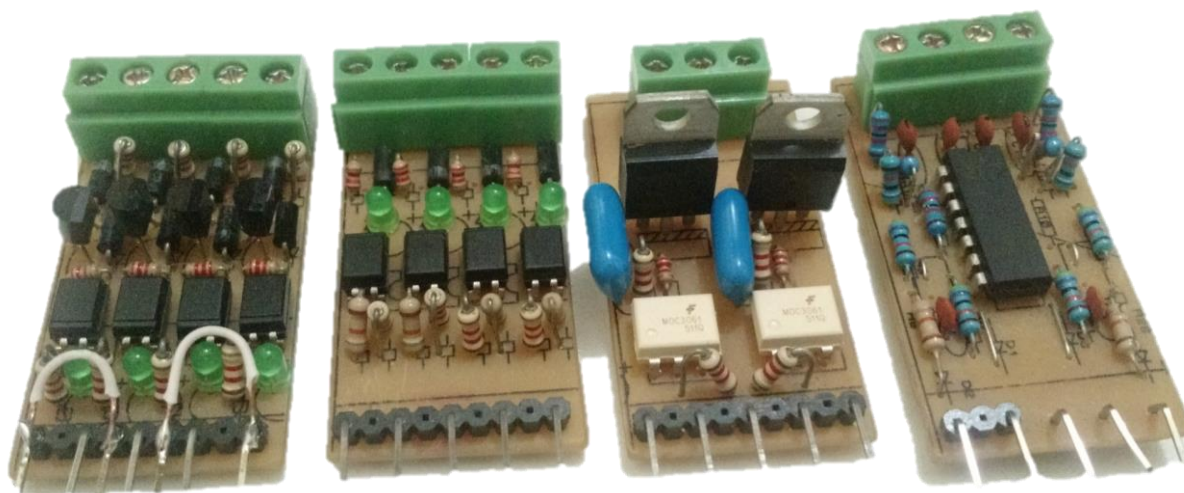
*Figura 45 - Bloco de saídas digitais.*



*Fonte: Autor.*

A Figura 46 - Placas de controle montadas. Figura 46 mostra as placas citadas anteriormente montadas.

*Figura 46 - Placas de controle montadas.*



*Fonte: Autor.*

#### **4.9. Comparativo de custo**

Na comparação de custos serão verificados três tipos de componente dois tipos de CLP e o conjunto proposto neste trabalho, o primeiro CLP é o utilizado na máquina, o segundo possui configuração semelhante ao conjunto proposto.

##### **4.9.1. CLP atual**

Controlador Programável ATOS EXPERT 1752P62

Características gerais:

- Programável através do software ATOS Winsup (Português e gratuito)

- CLP com IHM incorporada 20E 12S “P” com relógio interno, alimentação 90 a 240 VCA e saída aux. 24 VCC;
- Canal de comunicação serial RS232 e RS485;
- Frontal com teclado e display LCD 2 linhas x 20 caracteres;
- 20 Entradas 24 VCC tipo “P”;
- 12 Saídas 24 VCC tipo “P”;
- Memória FLASH para carregamento automático do programa de usuário;
- Programa de usuário armazenado em NVRAM.

#### Características Elétricas:

- Tensão de alimentação: 90 a 240Vca
- Consumo: 6,5VA @ 220Vca
- Tensão da saída auxiliar: 24Vcc
- Máxima corrente: 600mA

#### ENTRADAS DIGITAIS

- Tensão de trabalho: 24Vcc -30 / +40%
- Máxima corrente de entrada: 10mA (por canal)

#### SAÍDAS DIGITAIS:

- Tensão de trabalho: 24Vcc -30 / +40%
- Máxima corrente por saída: 2A
- Máxima corrente para as 12 saídas: 8A

Custo do CLP: 2.500,00, fonte mercado livre, link (<http://produto.mercadolivre.com.br/MLB-744310158-clp-atos-1752p62-clp-ihm- JM>).

#### 4.9.2. CLP semelhante ao proposto

##### CLP com IHM incorporada:

- Display gráfico Monocromático TFT 5.7 touchscreen
- Resolução 320 x 240
- Expansão de até 171 I/Os
- 2 Portas RS232/485

- 1 Porta CANbus
- Comunicação Modbus Master ou Slave
- Utilização de módulos RS485 ou Ethernet
- Bits/Coils: 4096
- Registros: 2048
- Temporizadores: 192
- Contadores: 24
- Até 255 telas configuráveis
- Tamanho do Programa: 1Mb
- Relógio em Tempo Real
- Acesso remoto para operação do equipamento
- Download/upload de programa remotamente
- Exportação de dados
- Criação de protocolos de comunicação
- Programação em Ladder e ambiente Windows
- Certificação CE e UL
- Alimentação 12/24 VDC

Valor do CLP 3.500,00 link compra:

*Figura 47 – CLP Com IHM incorporada (Unitronics).*



Fonte: [http://produto.mercadolivre.com.br/MLB-710489564-clp-com-ihm-incorporada-v530-53-b20b-nova- JM](http://produto.mercadolivre.com.br/MLB-710489564-clp-com-ihm-incorporada-v530-53-b20b-nova-JM).

#### 4.9.3. Conjunto proposto

São apresentados os componentes que possuem maior valor no desenvolvimento do projeto, os valores dos componentes do conjunto proposto são cotados em dólar, devido à dificuldade de encontrar todos eles no mercado nacional.

Componente	Valor (US\$)	Link
<b>Raspberry Pi 2</b>	41,91	<a href="http://www.dx.com/p/raspberry-pi-2-model-b-arm-cortex-a7-quad-core-cpu-900mhz-1gb-ram-support-windows-10-ubuntu-etc-378251#.VtsW7pwrKhc">http://www.dx.com/p/raspberry-pi-2-model-b-arm-cortex-a7-quad-core-cpu-900mhz-1gb-ram-support-windows-10-ubuntu-etc-378251#.VtsW7pwrKhc</a>
<b>Arduino Mega</b>	10,79	<a href="http://www.dx.com/p/mega2560-r3-atmega2560-16au-control-board-w-usb-cable-for-arduino-400944#.VtsYzpwKhc">http://www.dx.com/p/mega2560-r3-atmega2560-16au-control-board-w-usb-cable-for-arduino-400944#.VtsYzpwKhc</a>
<b>Tela Touch 7"</b>	49,99	<a href="http://www.dx.com/p/7-digital-touch-screen-drive-board-hdmi-vga-2av-for-raspberry-pcduino-cubieboard-318884#.VtsfTpwrKhc">http://www.dx.com/p/7-digital-touch-screen-drive-board-hdmi-vga-2av-for-raspberry-pcduino-cubieboard-318884#.VtsfTpwrKhc</a>
<b>Total</b>	106,69	

O dólar no dia 05/06/2016 está cotado em 3,7597, ou seja o valor em real dos principais componentes fica em torno de R\$ 386,08, ou seja, um valor bem abaixo dos outros CLPs mostrados.

## 5. RESULTADOS EXPERIMENTAIS

Para os testes em campo foi realizada a substituição do CLP original da máquina pelo conjunto Arduino e Raspberry, conforme proposto neste trabalho. A Figura 48 mostra a máquina automática de empacotar em teste com o sistema proposto, podendo ser verificado a nova IHM implementada na máquina.

*Figura 48 – Máquina automática em teste.*



*Fonte: Autor.*

A Figura 49 mostra de forma mais próxima a IHM instalada na máquina, onde pode ser verificado, que está conforme ao desenvolvido no tópico 4.7.1.

Figura 49 - Detalhe da IHM.

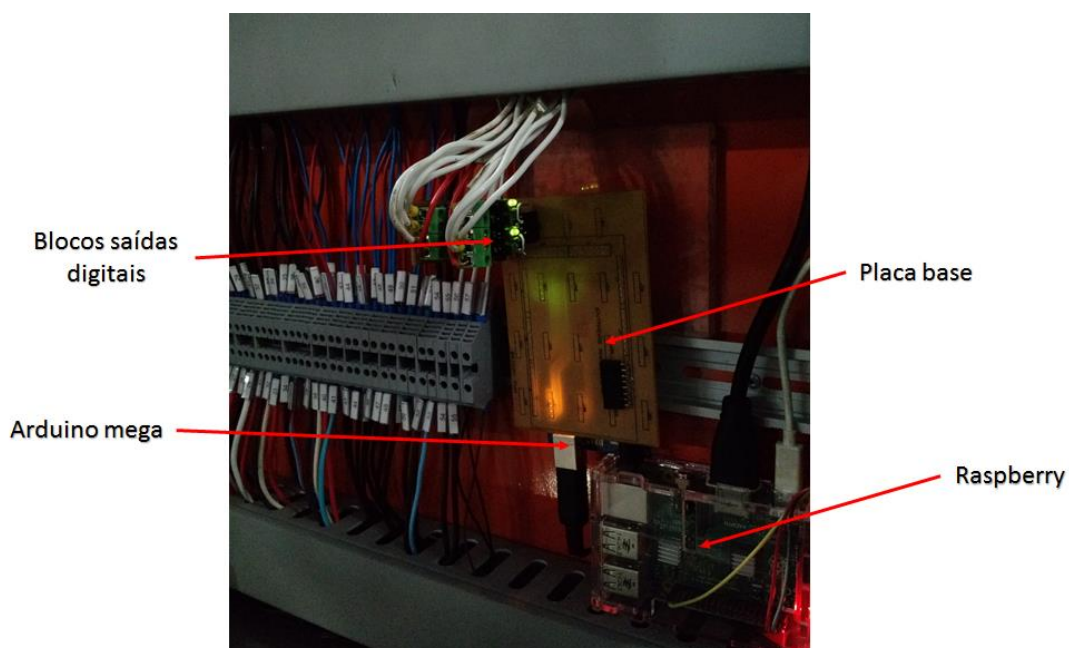


Fonte: Autor

Na parte de hardware de acionamento, foram utilizados dois blocos de saídas digitais, totalizando oito saídas.

A Figura 50 mostra o hardware que foi utilizado, onde o Arduino está por baixo da placa base, nos blocos de saídas é possível observar que duas saídas estão acionadas conforme os LEDs indicativos.

Figura 50 - Hardware de acionamento instalado.



Fonte: Autor.

## 6. CONCLUSÃO

Esse trabalho apresentou uma análise da substituição de um CLP por um conjunto, Raspberry e Arduino em uma máquina automática de empacotar. O trabalho mostra a possibilidade da aplicação, tendo em vista que o apresentou funcionamento conforme esperado e que o conjunto Raspberry e Arduino proporcionaram a execução das tarefas, neste caso o controle da máquina.

Este estudo poderá proporcionar a substituição efetiva do CLP pelo conjunto Arduino e Raspberry, além de possibilitar agregar mais tarefas, principalmente leitura de sinais analógicos, além da possibilidade de uma IHM para várias máquinas, com a utilização do protocolo I<sup>2</sup>C.

Esse projeto pode também facilmente ser migrado para controle de outros tipos de equipamentos além da possibilidade de criação de sistema supervisório com vários dispositivos em rede.

Para trabalhos futuros é visto a necessidade de continuar os testes e verificar itens como: confiabilidade, repetitividade, além de aplicar técnicas de software em tempo real para possibilitar aplicação em maior gama de equipamentos.

Tendo também como trabalho futuro a conclusão do projeto da caixa para o condicionamentos de todo o conjunto Arduino, Raspberry, IHM, blocos de entradas e blocos de saídas, o qual já foi iniciado, também foi feito a aquisição de uma impressora 3D, para impressão da caixa, de forma a criar um produto comercial e agregar valor ao equipamento.

## 7. BIBLIOGRAFIA

**ANVISA.** ANVISA. *Agencia Nacional de vigilancia Sanitária.* [Online] [Citado em: 30 de 01 de 2016.] [http://portal.anvisa.gov.br/wps/content/Anvisa+Portal/Anvisa/Inicio/Alimentos/Assuntos+de+Interesse/Embalagens.](http://portal.anvisa.gov.br/wps/content/Anvisa+Portal/Anvisa/Inicio/Alimentos/Assuntos+de+Interesse/Embalagens)

**Barão, Mariana Zanon. 2011.** *Dossie técnico embalagens para produtos alimentícios.* Paraná : Tecpar, 2011.

**Barros, André Ricardo Gouveia. 2013.** *Implementação de um serviço utilizando Linux embarcados com acesso e gerenciamento através de smartphone.* São Carlos : USP, 2013.

**BARRY, Willians W. 1992.** *Power Eletronics Devices, Drivers, Applications, and Passive Componets.* Glasgow : University of Strathclyde, 1992.

**BOLTON, WILLIAN. 2010.** *Mecatronica Uma abordagem multidisciplinar.* 4°. Porto Alegre : Bookman, 2010. p. 664.

**Cavalcanti, Pedro e Carmo, Chagas. 2006.** *História da embalagem no brasil.* São Paulo : GRIFFO, 2006.

**CETINKUNT, SABRI. 2008.** *Mecatrônica.* Rio de Janeiro : LTC, 2008.

**COLLINS, J A. 2006.** *Projeto Mecânico de Elementos de Máquinas: Uma Perspectiva de Prevenção da Falha.* Rio de Janeiro : LTC, 2006.

**Emadi, Ali. 205.** *Handbook of Automotive Power Eletronics And Motor Drives.* Espanha : LLC, 205.

**FIALHO, ARIVELTO BUSTAMANTE\_.** 2004. *Automação Pneumática Projetos, Dimensionamentos e análises de Circuitos.* 2. São Paulo : Érica Ltda, 2004.

**Guedes, Rodrigo Luiz. 2009.** *Sistema de Controle, Utilizando CLP e Supervisório Para Correção de Fator de Potência e Balanciamento de Fases no Secundário de um Transformador de uma Subestação.* Ouro Preto : Universidade Federal de Ouro Preto, 2009.



**Junior, Helio Taliani. 2012.** *ESTUDO DOS PROTOCOLOS DE COMUNICAÇÃO DAS ARQUITETURAS ELETROELETRÔNICAS AUTOMOTIVAS, COM FOCO NAS SUAS CARACTERÍSTICAS E RESPECTIVAS APLICAÇÕES, VISANDO O DIRECIONAMENTO PARA O USO ADEQUADO E CUSTOMIZADO EM CADA CATEGORIA DE VEÍCULO*. São Caetano do Sul : s.n., 2012.

**Junior, Milton Gontijo Ferreira. 2012.** *Controle de um Inversor de Frequência Via CLP*. Ouro Preto : Universidade Federal de Ouro Preto Escola de Minas, 2012.

**LEMOS, J F. 2009.** *Soldagem por Vibração em Matérias Termoplásticos (Monografia)*. São Paulo : FATEC ZL, 2009.

#### **Microchip.**

<http://ww1.microchip.com/downloads/en/DeviceDoc/39632e.pdf>. [Online] [Citado em: 05 de 01 de 2016.]

**Microsoft.** <http://ms-iot.github.io/content/en-US/IoTCore.htm>. [Online] [Citado em: 05 de 01 de 2016.]

**Noel, Felipe Kern. 2013.** *Medidor de consumo de energia elétrica utilizando rede em malhas sem fio*. Rio de Janeiro : Universidade Federal do Rio de Janeiro Escola Politécnica, 2013.

**Nunes, Felipe Vilela. 2013.** *Desenvolvimento de Sistema de Segurança Utilizando Microcontrolador PIC18F4550*. Ouro Preto : Universidade Federal de Ouro Preto Escola de Minas, 2013.

**Pereira, Felipe Átila Santos, et al. 2009.** *Controle de Eixos de Máquinas Industriais Utilizando CLP e Inversor de Frequência*. Governador Valadares : UNIVALE Universidade Vale do Rio Doce FENG Faculdade de Engenharia, 2009.

**Resende, Denise de Oliveira. 2014.** *Perguntas e Respostas sobre Materiais em contato com alimentos*. Brasília : Anvisa, 2014.

**Santos, Samuel de Souza. 2014.** *Implantação Raspberry Pi e Computação em Nuvem Para Sala de Aula Interagindo com Tecnologia Promethean*. Marília : UNIVEM Centro Universitário Eurípides de Marília, 2014.

**Silva, Davidson Felipe da. 2009.** *Sistema de Comunicação Bluetooth Utilizando Microcontrolador.* Recife : POLI Escola Politécnica de Pernambuco, 2009.

**Vitoratto, José Guilherme. 2014.** *PROTÓTIPO DE AMBIENTE INTELIGENTE BASEADO EM RASPBERRY PI, WEB E MOBILIDADE.* Assis : FEMA Fundação Educacional do Município de Assis, 2014.

## 8. Anexos

### 8.1. Código Arduino

```

////////////////////////////////////
// Implementação de protocolo de comunicação entre arduino e raspberry com I2C//
// Utilizando W IOT 05/01/16 //
// Rodolfo M. L. de Santana TCC engenharia elétrica //
// UFC Universidade Federal do Ceará //
// Orientador Rômulo Nunes //
////////////////////////////////////

#include <TimerOne.h> // INCLUI BIBLIOTECA TIMER
/*#include <Wire.h> // INCLUI BIBLIOTECA I2C */

int cont=0; // DEFINE VARIÁVEL INCREMENTO TIMER
//byte leitura; // DEFINE VARIÁVEL RESPOSTA DO MESTRE
//int VetorEscrita [3]; // DEFINE VETOR QUE RECEBE RECEBE DADOS DO
MESTRE
byte VetorDados [30]; // DEFINE VETOR PRINCIPAL DE CONFIGURAÇÃO DA
MÁQUINA

// DEFINIÇÃO DO PINOS QUE SERÃO UTILIZADOS
const int Mordaca = 31;
const int Vertical = 6;
const int Horizontal = 33;
const int Dosador = 35;
const int Puxador = 37;
const int Datador = 4;
const int Resfriamento = 8;
const int Reserva = 11;
const int Led = 13;

void setup()
{
// Definição dos pinos
pinMode(Mordaca, OUTPUT);
pinMode(Vertical, OUTPUT);
pinMode(Horizontal, OUTPUT);
pinMode(Dosador, OUTPUT);
pinMode(Puxador, OUTPUT);
pinMode(Datador, OUTPUT);
pinMode(Resfriamento, OUTPUT);
pinMode(Reserva, OUTPUT);
pinMode(Led, OUTPUT);

// Configuração do timer
Timer1.initialize(10000); // Configura o timer para estouros de 10 ms
Timer1.attachInterrupt(CicloMaquina); // Chama função de interrupção do timer

```

```

// Configura a velocidades da comunicação serial
Serial.begin(38400);

// Configura a utilização do I2C
//Wire.begin(8);           // Endereço desse dispositivo I2C
//Wire.onRequest(Requisicao); // Mestre requisita valor desse escravo
//Wire.onReceive(Recebimento); // Mestre faz uma escrita no escravo

// MORDAÇA
VetorDados[4] = 2;
VetorDados[5] = 80;

// HORIZONTAL
VetorDados[8] = 80;
VetorDados[9] = 50;

// DOSADOR
VetorDados[10] = 50;
VetorDados[11] = 90;

// TRACIONADOR
VetorDados[12] = 77;
VetorDados[13] = 37;

// LED
VetorDados[20] = 1;
VetorDados[21] = 97;

// CONT
VetorDados[2] = 100;
VetorDados[1] = 0;

}

void loop()
{
  //MORDAÇA
  if(cont==VetorDados[4]){digitalWrite(Mordaca, HIGH);}// ACIONA MORDAÇA
  if(cont==VetorDados[5]){digitalWrite(Mordaca, LOW);} // DESLIGA MORDAÇA

  //VERTICAL
  if(cont==VetorDados[6]){digitalWrite(Vertical, HIGH);}// ACIONA SOLDA VERTICA
  if(cont==VetorDados[7]){digitalWrite(Vertical, LOW);} // DESLIGA SOLDA
  VERTICAL

  //HORIZONTAL

```

```

if(cont==VetorDados[8]){digitalWrite(Horizontal, HIGH);}// ACIONA HORIZONTAL
if(cont==VetorDados[9]){digitalWrite(Horizontal, LOW);} // DESLIGA HORIZONTAL

//DOSADOR
if(cont==VetorDados[10]){digitalWrite(Dosador, HIGH);}// ACIONA DOSADOR
if(cont==VetorDados[11]){digitalWrite(Dosador, LOW);} // DESLIGA DOSADOR

//TRACIONADOR
if(cont==VetorDados[12]){digitalWrite(Puxador, HIGH);}// ACIONA TRACIONADOR
if(cont==VetorDados[13]){digitalWrite(Puxador, LOW);} // DESLIGA
TRACIONADOR

//DATADOR
if(cont==VetorDados[14]){digitalWrite(Datador, HIGH);}// ACIONA DATADOR
if(cont==VetorDados[15]){digitalWrite(Datador, LOW);} // DESLIGA DATADOR

//RESFRIAMENTO
if(cont==VetorDados[16]){digitalWrite(Resfriamento, HIGH);}// ACIONA
RESFRIAMENTO
if(cont==VetorDados[17]){digitalWrite(Resfriamento, LOW);} // DESLIGA
RESFRIAMENTO

//RESERVA
if(cont==VetorDados[18]){digitalWrite(Reserva, HIGH);}// ACIONA RESERVA
if(cont==VetorDados[19]){digitalWrite(Reserva, LOW);} // DESLIGA RESERVA

//LED
if(cont==VetorDados[20]){digitalWrite(Led, HIGH);}// ACIONA LED
if(cont==VetorDados[21]){digitalWrite(Led, LOW);} // DESLIGA LED
}

void CicloMaquina(void)
{
  cont=cont+1; // Incrementa a contagem
  if(cont==VetorDados[2])
  {
    cont=VetorDados[1];
  }
}

// função receber escrita de um mestre
void Recebimento(int howMany)
{
  if(Wire.available() > 2) // se tem mais de 2 bytes no buffer inicia a leitura
  {
    int x = 0;

```

```

while (0 < Wire.available())
{
    VetorEscrita [x] = Wire.read();
    Serial.print(VetorEscrita[x]);
    x = x+1;
}
Serial.println(" ");
}

// PROTOCOLO ESCRITA
// 101 (confirmação) - endereço - byte 1.
if(VetorEscrita[0] == 101) // letra "e" tabela asc2, ESCRITA
{
    VetorDados[VetorEscrita[1]] = VetorEscrita [2]; // Escreve no vetor dados
}

// PROTOCOLO DE LEITURA
// 108 (Confirmação de leitura) - endereço de leitura - 108 (Confirmação de leitura)
if(VetorEscrita[0] == 108) // letra "l" tabela asc2, LEITURA
{
    leitura=VetorDados[VetorEscrita[1]];
}
}

// RESPONDE COM O DADO REQUISITADO
void Requisicao()
{
    Wire.write(leitura);
}

```

## 8.2. Código Raspberry Pi

### 8.2.1. XAML

```

<Page
    x:Class="I2CRaspberry.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:I2CRaspberry"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d">
    <Grid Background="LightGray">
        <Grid>
            <!--Definição de linhas/colunas-->
            <Grid.RowDefinitions>
                <RowDefinition Height="auto"></RowDefinition>
                <RowDefinition Height="*"></RowDefinition>
                <RowDefinition Height="*"></RowDefinition>
                <RowDefinition Height="*"></RowDefinition>
            </Grid.RowDefinitions>
        </Grid>
    </Grid>

```

```

        <RowDefinition Height="*"></RowDefinition>
        <RowDefinition Height="*"></RowDefinition>
        <RowDefinition Height="*"></RowDefinition>
        <RowDefinition Height="auto"></RowDefinition>
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="*"></ColumnDefinition>
        <ColumnDefinition Width="*"></ColumnDefinition>
        <ColumnDefinition Width="auto"></ColumnDefinition>
        <ColumnDefinition Width="auto"></ColumnDefinition>
        <ColumnDefinition Width="auto"></ColumnDefinition>
    </Grid.ColumnDefinitions>

    <!--Preenchimento com Conteúdo-->
    <!--Linha 01-->
    <Grid Grid.Row="0" Grid.ColumnSpan="5" HorizontalAlignment="Center"
VerticalAlignment="Center">
        <TextBlock Text="MÁQUINA AUTOMÁTICA DE EMPACOTAR"
HorizontalAlignment="Center" Margin="20,20,15,15" VerticalAlignment="Center"
FontSize="50" FontWeight="Bold" RequestedTheme="Dark" Foreground="{ThemeResource
AppBarBorderThemeBrush}"/>
    </Grid>

    <!--Linha 02-->
    <Grid Grid.Column="0" Grid.Row="1" >
        <TextBlock Text="INICIO CICLO" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"></TextBlock>
    </Grid>
    <Grid Grid.Column="1" Grid.Row="1">
        <TextBlock x:Name="BoxInicio" Text="0" HorizontalAlignment="Right"
VerticalAlignment="Center" FontSize="40" FontWeight="Bold"
PointerExited="EventoBoxInicial" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}" Margin="0,11,255,10"/>
    </Grid>
    <Grid Grid.Column="2" Grid.Row="1">
        <TextBlock Text="MORDAÇA" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"></TextBlock>
    </Grid>
    <Grid Grid.Column="3" Grid.Row="1">
        <TextBlock x:Name="BoxMordacaI" Margin="20,0,20,0" Text="25"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoBoxMordacaI" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
    </Grid>
    <Grid Grid.Column="4" Grid.Row="1">
        <TextBlock x:Name="BoxMordacaF" Margin="20,0,30,0" Text="50"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoBoxMordacaF" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
    </Grid>
    <!--<Grid Grid.Column="5" Grid.Row="1">
        <CheckBox x:Name="Hmordaca" Checked="Hmordaca_Checked" ></CheckBox>
    </Grid-->

    <!--Linha 03-->
    <Grid Grid.Column="0" Grid.Row="2">
        <TextBlock Text="FINAL CICLO" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"></TextBlock>

```

```

        </Grid>
        <Grid Grid.Column="1" Grid.Row="2">
            <TextBlock x:Name="BoxFinal" Text="100" HorizontalAlignment="Left"
VerticalAlignment="Center" FontSize="40" FontWeight="Bold"
PointerExited="EventoBoxFinal" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>
        <Grid Grid.Column="2" Grid.Row="2">
            <TextBlock Text="VERTICAL" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/></TextBlock>
        </Grid>
        <Grid Grid.Column="3" Grid.Row="2">
            <TextBlock x:Name="BoxVerticalI" Text="30" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="40" FontWeight="Bold"
PointerExited="EventoBoxVerticalI" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>
        <Grid Grid.Column="4" Grid.Row="2">
            <TextBlock x:Name="BoxVerticalF" Margin="20,0,30,0" Text="70"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoBoxVerticalF" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>

        <!--Linha 04-->
        <Grid Grid.Row="3" Grid.ColumnSpan="2" HorizontalAlignment="Center"
VerticalAlignment="Center">
            <Button Content="CARREGAR" Width="200" Height="60" Background="Blue"
FontSize="26.667" FontWeight="Bold" Click="BotaoCarrega" ></Button>
        </Grid>
        <Grid Grid.Column="2" Grid.Row="3">
            <TextBlock Text="HORIZONTAL" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/></TextBlock>
        </Grid>
        <Grid Grid.Column="3" Grid.Row="3">
            <TextBlock x:Name="BoxHorizontalI" Text="30"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoBoxHorizontalI" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>
        <Grid Grid.Column="4" Grid.Row="3">
            <TextBlock x:Name="BoxHorizontalF" Margin="20,0,30,0" Text="45"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoBoxHorizontalF" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>

        <!--Linha 05-->
        <Grid Grid.Row="4" Grid.Column="0" HorizontalAlignment="Center"
VerticalAlignment="Center">
            <Button Content="-1" Width="130" Height="50"
HorizontalAlignment="Center" VerticalAlignment="Bottom" Background="Orange"
FontSize="26.667" FontWeight="Bold" Click="BotaoMenosUm"></Button>
        </Grid>
        <Grid Grid.Row="4" Grid.Column="1" HorizontalAlignment="Center"
VerticalAlignment="Center">
            <Button Content="+1" Width="130" Height="50"
HorizontalAlignment="Center" VerticalAlignment="Bottom" Background="Orange"
FontSize="26.667" FontWeight="Bold" Click="BotaoMaisUm"></Button>

```



```

        </Grid>
        <Grid Grid.Column="2" Grid.Row="4">
            <TextBlock Text="DOSADOR" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"></TextBlock>
        </Grid>
        <Grid Grid.Column="3" Grid.Row="4">
            <TextBlock x:Name="BoxDosadorI" Text="65" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="40" FontWeight="Bold"
PointerExited="EventoBoxDosadorI" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>
        <Grid Grid.Column="4" Grid.Row="4">
            <TextBlock x:Name="BoxDosadorF" Margin="20,0,30,0" Text="80"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoDosadorBoxF" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>

<!--Linha 06-->

        <Grid Grid.Row="5" Grid.Column="0" HorizontalAlignment="Center"
VerticalAlignment="Center">
            <Button Content="-10" Width="150" Height="60"
HorizontalAlignment="Center" VerticalAlignment="Bottom" Background="OrangeRed"
FontSize="35" FontWeight="Bold" Click="BotaoMenosDez"></Button>
        </Grid>
        <Grid Grid.Row="5" Grid.Column="1" HorizontalAlignment="Center"
VerticalAlignment="Center">
            <Button Content="+10" Width="150" Height="60"
HorizontalAlignment="Center" VerticalAlignment="Bottom" Background="OrangeRed"
FontSize="35" FontWeight="Bold" Click="BotaoMaisDez"></Button>
        </Grid>
        <Grid Grid.Column="2" Grid.Row="5">
            <TextBlock Text="TRACIONADOR" Margin="15,0,15,0"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="35"
FontWeight="Bold" Foreground="{ThemeResource AppBarBackgroundThemeBrush}"></TextBlock>
        </Grid>
        <Grid Grid.Column="3" Grid.Row="5">
            <TextBlock x:Name="BoxTracionadorI" Text="60"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoBoxTracionadorBoxI" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>
        <Grid Grid.Column="4" Grid.Row="5">
            <TextBlock x:Name="BoxTracionadorF" Margin="20,0,30,0" Text="90"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoBoxTracionadorBoxF" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
        </Grid>

<!--Linha 07-->
        <Grid Grid.Row="6" Grid.ColumnSpan="2" HorizontalAlignment="Center"
VerticalAlignment="Center">
            <Button Content="LIGA" Width="200" Height="60"
HorizontalAlignment="Right" VerticalAlignment="Bottom" Background="Red"
FontSize="26.667" FontWeight="Bold" Click="BotaoLiga"></Button>
        </Grid>
        <Grid Grid.Column="2" Grid.Row="6">

```

```

                <TextBlock Text="DATADOR" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"></TextBlock>
            </Grid>
            <Grid Grid.Column="3" Grid.Row="6">
                <TextBlock x:Name="BoxDatadorI" Text="0" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="40" FontWeight="Bold"
PointerExited="EventoDatadorBoxI" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
            </Grid>
            <Grid Grid.Column="4" Grid.Row="6">
                <TextBlock x:Name="BoxDatadorF" Margin="20,0,30,0" Text="15"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoDatadorBoxF" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
            </Grid>

            <!--Linha 08-->
            <Grid Grid.Row="7" Grid.ColumnSpan="2" HorizontalAlignment="Center"
VerticalAlignment="Center">
                <Button Content="DESLIGA" Width="200" Height="60" Margin="15,10,15,20"
Background="Green" FontSize="26.667" FontWeight="Bold" Click="BotaoDesliga"></Button>
            </Grid>
            <Grid Grid.Column="2" Grid.Row="7">
                <TextBlock Text="RESFRIAMENTO" HorizontalAlignment="Center"
VerticalAlignment="Center" FontSize="35" FontWeight="Bold" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"></TextBlock>
            </Grid>
            <Grid Grid.Column="3" Grid.Row="7">
                <TextBlock x:Name="BoxResfriamentoI" Text="80"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoResfriamentoBoxI" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
            </Grid>
            <Grid Grid.Column="4" Grid.Row="7">
                <TextBlock x:Name="BoxResfriamentoF" Margin="20,0,30,0" Text="95"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40"
FontWeight="Bold" PointerExited="EventoResfriamentoBoxF" Foreground="{ThemeResource
AppBarBackgroundThemeBrush}"/>
            </Grid>

        </Grid>
    </Grid>
</Page>

```

### 8.2.2. C#

```

using System;
using System.Threading;
using Windows.UI.Xaml.Controls;
using Windows.Devices.Enumeration;
using Windows.Devices.I2c;

namespace I2CRaspberry
{
    public sealed partial class MainPage : Page
    {
        private const byte EnderecoEscravo = 0x08; /* ENDEREÇO DE 7-bitS DO ESCRAVO
NO I2C */
        private int Selecao = 0; /* VARIÁVEL DE IDENTIFICAÇÃO DE CAMPO */
    }
}

```

```

private I2cDevice ARDUINO;          /* DECLARA COMUNICAÇÃO I2C*/

public MainPage()
{
    this.InitializeComponent();
    Unloaded += MainPage_Unloaded;
    // InitI2CAccel();              /* INICIALIZAÇÃO DA COMUNICAÇÃO I2C */
}

private async void InitI2CAccel()
{
    string aqs = I2cDevice.GetDeviceSelector();
    var dis = await DeviceInformation.FindAllAsync(aqs);
    if (dis.Count == 0)
    {
        return;
    }

    var settings = new I2cConnectionSettings(EnderecoEsravo);
    settings.BusSpeed = I2cBusSpeed.FastMode;
    ARDUINO = await I2cDevice.FromIdAsync(dis[0].Id, settings);
    if (ARDUINO == null)
    {
        return;
    }
}

private void MainPage_Unloaded(object sender, object args)
{
    /* Cleanup */
    ARDUINO.Dispose();
}

private void BotaoLiga(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x0E, 0x11 };

    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    /* Em caso de falha na escrita para de rodar */
    catch (Exception ex)
    {
        return;
    }
}

private void BotaoDesliga(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x00, 0x01 };

    /* Write the register settings */
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
        //I2CAccel.Write(WriteBuf_PowerControl);
    }
    /* If the write fails display the error and stop running */
    catch (Exception ex)

```

```

    {
        return;
    }
}

// BOTÃO CARREGA
private void BotaoCarrega(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    if (Selecao == 1)
    {
        int x = Int32.Parse(BoxInicio.Text);
        /* CONVERTE DE INT32 PARA INT */
        byte y = Convert.ToByte(x);
        /* CONVERTE EM BYTE E CARREGA EM Y */

        byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x01, y };
        /* ORGANIZA OS BYTES */
        try
        {
            ARDUINO.Write(WriteBuf_DataFormat);
        }
        /* ENVIA OS BYTES PARA O ARDUINO */
        catch (Exception ex)
        {
            return;
        }
    }

    if (Selecao == 2)
    {
        int x = Int32.Parse(BoxFinal.Text);
        byte y = Convert.ToByte(x);

        byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x02, y };
        try
        {
            ARDUINO.Write(WriteBuf_DataFormat);
        }
        catch (Exception ex)
        {
            return;
        }
    }

    if (Selecao == 3)
    {
        int x = Int32.Parse(BoxMordacaI.Text);
        byte y = Convert.ToByte(x);

        byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x04, y };
        try
        {
            ARDUINO.Write(WriteBuf_DataFormat);
        }
        catch (Exception ex)
        {
            return;
        }
    }
}

```

```
}  
  
if (Selecao == 4)  
{  
    int x = Int32.Parse(BoxMordacaF.Text);  
    byte y = Convert.ToByte(x);  
  
    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x05, y };  
    try  
    {  
        ARDUINO.Write(WriteBuf_DataFormat);  
    }  
    catch (Exception ex)  
    {  
        return;  
    }  
}  
  
if (Selecao == 5)  
{  
    int x = Int32.Parse(BoxVerticalI.Text);  
    byte y = Convert.ToByte(x);  
  
    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x06, y };  
    try  
    {  
        ARDUINO.Write(WriteBuf_DataFormat);  
    }  
    catch (Exception ex)  
    {  
        return;  
    }  
}  
  
if (Selecao == 6)  
{  
    int x = Int32.Parse(BoxVerticalF.Text);  
    byte y = Convert.ToByte(x);  
  
    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x07, y };  
    try  
    {  
        ARDUINO.Write(WriteBuf_DataFormat);  
    }  
    catch (Exception ex)  
    {  
        return;  
    }  
}  
  
if (Selecao == 7)  
{  
    int x = Int32.Parse(BoxHorizontalI.Text);  
    byte y = Convert.ToByte(x);  
  
    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x08, y };  
    try  
    {  
        ARDUINO.Write(WriteBuf_DataFormat);  
    }  
    catch (Exception ex)
```

```

    {
        return;
    }
}

if (Selecao == 8)
{
    int x = Int32.Parse(BoxHorizontalF.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x09, y };
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}

if (Selecao == 9)
{
    int x = Int32.Parse(BoxDosadorI.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x0A, y };
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}

if (Selecao == 10)
{
    int x = Int32.Parse(BoxDosadorF.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x0B, y };
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}

if (Selecao == 11)
{
    int x = Int32.Parse(BoxTracionadorI.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x0C, y };
    try
    {

```

```

        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}

if (Selecao == 12)
{
    int x = Int32.Parse(BoxTracionadorF.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x0D, y };
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}

if (Selecao == 13)
{
    int x = Int32.Parse(BoxDatadorI.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x0E, y };
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}

if (Selecao == 14)
{
    int x = Int32.Parse(BoxDatadorF.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x0F, y };
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}

if (Selecao == 15)
{
    int x = Int32.Parse(BoxResfriamentoI.Text);
    byte y = Convert.ToByte(x);

```

```

byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x10, y };
try
{
    ARDUINO.Write(WriteBuf_DataFormat);
}
catch (Exception ex)
{
    return;
}
}

if (Selecao == 16)
{
    int x = Int32.Parse(BoxResfriamentoF.Text);
    byte y = Convert.ToByte(x);

    byte[] WriteBuf_DataFormat = new byte[] { 0x65, 0x11, y };
    try
    {
        ARDUINO.Write(WriteBuf_DataFormat);
    }
    catch (Exception ex)
    {
        return;
    }
}
}

private void BotaoMenosUm(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    if (Selecao == 1)
    {
        int x = Int32.Parse(BoxInicio.Text);
        /* CONVERSÃO DE INT32 PARA INT */
        x--;
        /* DECREMETO EM UMA UNIDADE */
        if (x < 0)
        /* LIMITA O VALOR ENTRE 0 E 250 */
        { x = 250; }
        BoxInicio.Text = "" + x;
        /* ATUALIZA O VALOR NO PARÂMETRO */
    }

    if (Selecao == 2)
    {
        int x = Int32.Parse(BoxFinal.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxFinal.Text = "" + x;
    }

    if (Selecao == 3)
    {
        int x = Int32.Parse(BoxMordacaI.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxMordacaI.Text = "" + x;
    }
}

```



```
if (Selecao == 4)
{
    int x = Int32.Parse(BoxMordacaF.Text);
    x--;
    if (x < 0)
    { x = 250; }
    BoxMordacaF.Text = "" + x;
}

if (Selecao == 5)
{
    int x = Int32.Parse(BoxVerticalI.Text);
    x--;
    if (x < 0)
    { x = 250; }
    BoxVerticalI.Text = "" + x;
}

if (Selecao == 6)
{
    int x = Int32.Parse(BoxVerticalF.Text);
    x--;
    if (x < 0)
    { x = 250; }
    BoxVerticalF.Text = "" + x;
}

if (Selecao == 7)
{
    int x = Int32.Parse(BoxHorizontalI.Text);
    x--;
    if (x < 0)
    { x = 250; }
    BoxHorizontalI.Text = "" + x;
}

if (Selecao == 8)
{
    int x = Int32.Parse(BoxHorizontalF.Text);
    x--;
    if (x < 0)
    { x = 250; }
    BoxHorizontalF.Text = "" + x;
}

if (Selecao == 9)
{
    int x = Int32.Parse(BoxDosadorI.Text);
    x--;
    if (x < 0)
    { x = 250; }
    BoxDosadorI.Text = "" + x;
}

if (Selecao == 10)
{
    int x = Int32.Parse(BoxDosadorF.Text);
    x--;
    if (x < 0)
    { x = 250; }
    BoxDosadorF.Text = "" + x;
}
```

```

    }

    if (Selecao == 11)
    {
        int x = Int32.Parse(BoxTracionadorI.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxTracionadorI.Text = "" + x;
    }

    if (Selecao == 12)
    {
        int x = Int32.Parse(BoxTracionadorF.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxTracionadorF.Text = "" + x;
    }

    if (Selecao == 13)
    {
        int x = Int32.Parse(BoxDatadorI.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxDatadorI.Text = "" + x;
    }

    if (Selecao == 14)
    {
        int x = Int32.Parse(BoxDatadorF.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxDatadorF.Text = "" + x;
    }

    if (Selecao == 15)
    {
        int x = Int32.Parse(BoxResfriamentoI.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxResfriamentoI.Text = "" + x;
    }

    if (Selecao == 16)
    {
        int x = Int32.Parse(BoxResfriamentoF.Text);
        x--;
        if (x < 0)
        { x = 250; }
        BoxResfriamentoF.Text = "" + x;
    }
}

private void BotaoMaisUm(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    if (Selecao == 1)
    {

```

```
        int x = Int32.Parse(BoxInicio.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxInicio.Text = "" + x;
    }

    if (Selecao == 2)
    {
        int x = Int32.Parse(BoxFinal.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxFinal.Text = "" + x;
    }

    if (Selecao == 3)
    {
        int x = Int32.Parse(BoxMordacaI.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxMordacaI.Text = "" + x;
    }

    if (Selecao == 4)
    {
        int x = Int32.Parse(BoxMordacaF.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxMordacaF.Text = "" + x;
    }

    if (Selecao == 5)
    {
        int x = Int32.Parse(BoxVerticalI.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxVerticalI.Text = "" + x;
    }

    if (Selecao == 6)
    {
        int x = Int32.Parse(BoxVerticalF.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxVerticalF.Text = "" + x;
    }

    if (Selecao == 7)
    {
        int x = Int32.Parse(BoxHorizontalI.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxHorizontalI.Text = "" + x;
    }
}
```

```
if (Selecao == 8)
{
    int x = Int32.Parse(BoxHorizontalF.Text);
    x++;
    if (x > 250)
    { x = 0; }
    BoxHorizontalF.Text = "" + x;
}

if (Selecao == 9)
{
    int x = Int32.Parse(BoxDosadorI.Text);
    x++;
    if (x > 250)
    { x = 0; }
    BoxDosadorI.Text = "" + x;
}

if (Selecao == 10)
{
    int x = Int32.Parse(BoxDosadorF.Text);
    x++;
    if (x > 250)
    { x = 0; }
    BoxDosadorF.Text = "" + x;
}

if (Selecao == 11)
{
    int x = Int32.Parse(BoxTracionadorI.Text);
    x++;
    if (x > 250)
    { x = 0; }
    BoxTracionadorI.Text = "" + x;
}

if (Selecao == 12)
{
    int x = Int32.Parse(BoxTracionadorF.Text);
    x++;
    if (x > 250)
    { x = 0; }
    BoxTracionadorF.Text = "" + x;
}

if (Selecao == 13)
{
    int x = Int32.Parse(BoxDatadorI.Text);
    x++;
    if (x > 250)
    { x = 0; }
    BoxDatadorI.Text = "" + x;
}

if (Selecao == 14)
{
    int x = Int32.Parse(BoxDatadorF.Text);
    x++;
    if (x > 250)
    { x = 0; }
    BoxDatadorF.Text = "" + x;
}
```

```

    }

    if (Selecao == 15)
    {
        int x = Int32.Parse(BoxResfriamentoI.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxResfriamentoI.Text = "" + x;
    }

    if (Selecao == 16)
    {
        int x = Int32.Parse(BoxResfriamentoF.Text);
        x++;
        if (x > 250)
        { x = 0; }
        BoxResfriamentoF.Text = "" + x;
    }
}

private void BotaoMenosDez(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    if (Selecao == 1)
    {
        int x = Int32.Parse(BoxInicio.Text);
        x = x - 10;
        if (x < 0)
        { x = 250; }
        BoxInicio.Text = "" + x;
    }

    if (Selecao == 2)
    {
        int x = Int32.Parse(BoxFinal.Text);
        x = x - 10;
        if (x < 0)
        { x = 250; }
        BoxFinal.Text = "" + x;
    }

    if (Selecao == 3)
    {
        int x = Int32.Parse(BoxMordacaI.Text);
        x = x - 10;
        if (x < 0)
        { x = 250; }
        BoxMordacaI.Text = "" + x;
    }

    if (Selecao == 4)
    {
        int x = Int32.Parse(BoxMordacaF.Text);
        x = x - 10;
        if (x < 0)
        { x = 250; }
        BoxMordacaF.Text = "" + x;
    }

    if (Selecao == 5)
    {

```

```
    int x = Int32.Parse(BoxVerticalI.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxVerticalI.Text = "" + x;
}

if (Selecao == 6)
{
    int x = Int32.Parse(BoxVerticalF.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxVerticalF.Text = "" + x;
}

if (Selecao == 7)
{
    int x = Int32.Parse(BoxHorizontalI.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxHorizontalI.Text = "" + x;
}

if (Selecao == 8)
{
    int x = Int32.Parse(BoxHorizontalF.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxHorizontalF.Text = "" + x;
}

if (Selecao == 9)
{
    int x = Int32.Parse(BoxDosadorI.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxDosadorI.Text = "" + x;
}

if (Selecao == 10)
{
    int x = Int32.Parse(BoxDosadorF.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxDosadorF.Text = "" + x;
}

if (Selecao == 11)
{
    int x = Int32.Parse(BoxTracionadorI.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxTracionadorI.Text = "" + x;
}
```

```

if (Selecao == 12)
{
    int x = Int32.Parse(BoxTracionadorF.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxTracionadorF.Text = "" + x;
}

if (Selecao == 13)
{
    int x = Int32.Parse(BoxDatadorI.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxDatadorI.Text = "" + x;
}

if (Selecao == 14)
{
    int x = Int32.Parse(BoxDatadorF.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxDatadorF.Text = "" + x;
}

if (Selecao == 15)
{
    int x = Int32.Parse(BoxResfriamentoI.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxResfriamentoI.Text = "" + x;
}

if (Selecao == 16)
{
    int x = Int32.Parse(BoxResfriamentoF.Text);
    x = x - 10;
    if (x < 0)
    { x = 250; }
    BoxResfriamentoF.Text = "" + x;
}
}

private void BotaoMaisDez(object sender, Windows.UI.Xaml.RoutedEventArgs e)
{
    if (Selecao == 1)
    {
        int x = Int32.Parse(BoxInicio.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxInicio.Text = "" + x;
    }

    if (Selecao == 2)
    {
        int x = Int32.Parse(BoxFinal.Text);
        x = x + 10;
    }
}

```

```
        if (x > 250)
        { x = 0; }
        BoxFinal.Text = "" + x;
    }

    if (Selecao == 3)
    {
        int x = Int32.Parse(BoxMordacaI.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxMordacaI.Text = "" + x;
    }

    if (Selecao == 4)
    {
        int x = Int32.Parse(BoxMordacaF.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxMordacaF.Text = "" + x;
    }

    if (Selecao == 5)
    {
        int x = Int32.Parse(BoxVerticalI.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxVerticalI.Text = "" + x;
    }

    if (Selecao == 6)
    {
        int x = Int32.Parse(BoxVerticalF.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxVerticalF.Text = "" + x;
    }

    if (Selecao == 7)
    {
        int x = Int32.Parse(BoxHorizontalI.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxHorizontalI.Text = "" + x;
    }

    if (Selecao == 8)
    {
        int x = Int32.Parse(BoxHorizontalF.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxHorizontalF.Text = "" + x;
    }

    if (Selecao == 9)
    {
```



```
        int x = Int32.Parse(BoxDosadorI.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxDosadorI.Text = "" + x;
    }

    if (Selecao == 10)
    {
        int x = Int32.Parse(BoxDosadorF.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxDosadorF.Text = "" + x;
    }

    if (Selecao == 11)
    {
        int x = Int32.Parse(BoxTracionadorI.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxTracionadorI.Text = "" + x;
    }

    if (Selecao == 12)
    {
        int x = Int32.Parse(BoxTracionadorF.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxTracionadorF.Text = "" + x;
    }

    if (Selecao == 13)
    {
        int x = Int32.Parse(BoxDatadorI.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxDatadorI.Text = "" + x;
    }

    if (Selecao == 14)
    {
        int x = Int32.Parse(BoxDatadorF.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxDatadorF.Text = "" + x;
    }

    if (Selecao == 15)
    {
        int x = Int32.Parse(BoxResfriamentoI.Text);
        x = x + 10;
        if (x > 250)
        { x = 0; }
        BoxResfriamentoI.Text = "" + x;
    }
}
```

```

        if (Selecao == 16)
        {
            int x = Int32.Parse(BoxResfriamentoF.Text);
            x = x + 10;
            if (x > 250)
            { x = 0; }
            BoxResfriamentoF.Text = "" + x;
        }
    }

    private void EventoBoxInicial(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 1;
    }

    private void EventoBoxFinal(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 2;
    }

    private void EventoBoxMordacaI(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 3;
    }

    private void EventoBoxMordacaF(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 4;
    }

    private void EventoBoxVerticalI(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 5;
    }

    private void EventoBoxVerticalF(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 6;
    }

    private void EventoBoxHorizontalI(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 7;
    }

    private void EventoBoxHorizontalF(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Selecao = 8;
    }

    private void EventoBoxDosadorI(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {

```

```
        Seleccion = 9;
    }

    private void EventoDosadorBoxF(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Seleccion = 10;
    }

    private void EventoBoxTracionadorBoxI(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Seleccion = 11;
    }

    private void EventoBoxTracionadorBoxF(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Seleccion = 12;
    }

    private void EventoDatadorBoxI(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Seleccion = 13;
    }

    private void EventoDatadorBoxF(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Seleccion = 14;
    }

    private void EventoResfriamentoBoxI(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Seleccion = 15;
    }

    private void EventoResfriamentoBoxF(object sender,
Windows.UI.Xaml.Input.PointerRoutedEventArgs e)
    {
        Seleccion = 16;
    }
}
}
```