UNIVERSIDADE DO ESTADO DE MINAS GERAIS

Unidade Ituiutaba

Curso de Bacharelado em Engenharia de Computação

COFRE DE SEGURANÇA PARA UM BANCO

THIAGO PIZARRO CARVALHO GOUVEIA

Ituiutaba – MG

2014

THIAGO PIZARRO CARVALHO GOUVEIA

COFRE DE SEGURANÇA PARA UM BANCO

Monografia de Conclusão de Curso apresentada à Coordenação do Curso de Engenharia da Computação, da Universidade do Estado de Minas Gerais, como requisito para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: Prof. Me. Walteno Martins Parreira Jr.

Ituiutaba
2014

THIAGO PIZARRO CARVALHO GOUVEIA

COFRE DE SEGURANÇA PARA UM BANCO

Monografia de Conclusão de Curso apresentada à Coordenação do Curso de Engenharia da Computação, da Universidade do Estado de Minas Gerais, como requisito para obtenção do grau de Bacharel em Engenharia da Computação.

Ituiutaba, 27 de Novembro de 2014

Banca Examinadora

_____

Prof(a). Msc. Anderson de Melo Valadão

_____

Prof(a). Esp. Flávio Eurípedes de Oliveira

_____

Prof. Me. Walteno Martins Parreira Jr (Orientador).

Dedico este trabalho a minha família, aos meus colegas de curso, e em especial aos meus professores, meu orientador, meus pais, meu irmão e aos amigos de verdade, pois todos sempre me apoiaram.

AGRADECIMENTOS

A Deus por ter me dado saúde e força para superar as dificuldades.

Agradeço aos meus pais Eder Franco Gouveia e Beatriz Pizarro de Carvalho Gouveia que sempre me apoiaram em minhas escolhas, me incentivando e ajudando. Ao meu irmão que sempre me deu bons conselhos e sempre me estendeu a mão.

Sou eternamente grato aos familiares que me apoiaram. Aos professores por compartilharem seus conhecimentos comigo. A minha banca examinadora.

Ao meu orientador Prof. Me. Walteno Martins Parreira Jr, pelo suporte no pouco tempo que lhe coube, pelas suas correções e incentivos.

Ao Saulo de Moraes Garcia Júnior por me fornecer a ideia deste trabalho de conclusão de curso.

E a todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

"Que os vossos esforços desafiem as impossibilidades, lembrai-vos de que as grandes coisas do homem foram conquistadas do que parecia impossível."
(Charles Chaplin)

Resumo

No projeto será utilizado o microcontrolador Arduino, motor de passo, display LCD, keypad e outros componentes. O objetivo é desenvolver um protótipo de cofre que receberá um código, que através do número inserido saberá a posição em que a gaveta do cliente está, através de rotações do motor, esta gaveta será parada na posição de acesso ao cliente, onde o cliente poderá inserir ou retirar objetos ou pertences desta.

Palavras-chave: Trabalho de Conclusão, Automação e Segurança, Arduino.

Abstract

Project in the Arduino microcontroller, stepper motor, LCD display, keypad and other components will be used. The goal is to develop a prototype vault that received a code, which means the number inserted will know the position in the drawer of this client through engine speeds, this drawer will stop at the customer access location where the client can inserting or removing objects or possessions thereof.

Password: Project Completion, Safety and Security, Arduino.

SUMÁRIO

1 - INTRODUÇÃO

Hoje em dia um dos mercados que está evoluindo é o do ramo da segurança, com isso o projeto será desenvolvido com base neste tema. Um corpo circular em formato de um disco onde haverá abertura em apenas uma área. Dentro deste estará a parte robótica responsável pelos movimentos de todo o Disco.

Pretende-se com este projeto construir um modelo de cofre de segurança para clientes de bancos, este cofre terá três gavetas que poderão ser usadas pelos proprietários para guardar seus pertences. O cofre funciona da seguinte forma: Este cofre contém um motor de passo na base que é responsável por rotacionar as gavetas, possui um Teclado (*Keypad)* responsável por enviar comandos ao cofre, contém um Display LCD (*Liquid Crystal Display*) que é encarregado de mostrar em sua tela a posição da gaveta que o cofre irá parar e caso a senha digitada seja incorreta este irá mostrar uma mensagem de "SENHA INCORRETA" .

Será utilizado o microcontrolador Arduino, que atualmente esta sendo aplicado em muitos projetos multidisciplinares por sua facilidade de utilização e ser acessível às propostas de desenvolvimento de atividades acadêmicas. Com o Arduino é possível desenvolver códigos para controlar os movimentos necessários do motor de passo.

A primeira etapa a ser realizada é a montagem do motor de passo junto ao Arduino e testar o funcionamento destes já acoplados, com isso funcionando o próximo passo é fazer a comunicação entre o Arduino, o teclado numérico e o display LCD.

Em seguida após estabelecer esta comunicação, será necessária a programação sendo esta em uma linguagem específica para o Arduino, para executar varias funções sendo que estas seguirão lógicas de programação, pois será necessário que o Arduino consiga interpretar a senha inserida no teclado, sendo que com esta senha o motor irá girar horizontalmente e parar na gaveta correspondente à senha, onde o objeto será colocado.

Em seguida, para finalizar a parte prática, será realizado o teste para verificar a funcionalidade e ver se os objetos foram exatamente para as gavetas escolhidas. Por fim, será confeccionada a monografia, que terá o seu desenvolvimento efetivamente realizado ao longo de todo o ano de 2014.

1.1 - Objetivos

1.1.1 Objetivo Geral

Desenvolver protótipo de um cofre automatizado, este receberá uma senha que irá ser fornecida pelo cliente do banco através de um *keypad*, caso essa senha seja valida o motor de passo irá rotacionar na posição horizontal até a respectiva gaveta, parando em uma abertura que o cliente terá acesso a sua gaveta. Com isso o cliente poderá inserir pertences na gaveta ou retirá-los. Quando o cliente terminar de utilizar sua gaveta, ele deverá inserir a senha responsável por voltar à gaveta na sua posição inicial.

1.1.2 Objetivos Específicos

- Ter uma maior segurança e passar isso para o cliente usuário do produto final, para que este veja que a melhor forma de segurança para seus pertences é esta oferecida pelo banco fornecedor deste serviço.
- Acoplar teclado numérico com display LCD para que o cliente veja que o protótipo está indo para a sua gaveta.
- Pesquisar melhorias para que o protótipo seja de baixo custo e tenha uma boa funcionalidade.
- Desenvolver os códigos para controlar o motor.
- Realizar testes para verificar a funcionalidade do motor.

1.2 - Justificativas

Com o crescimento do mercado de automação e segurança, decidiu-se montar o Trabalho de conclusão de curso de engenharia de computação em relação a estes ramos que estão relacionados com os objetivos do curso de atuação, preferencialmente na área de automação e assim com a possibilidade de alcançar parte significativa da população.

Esta proposta abrange diversos conhecimentos disponibilizados ao longo do curso e deste modo, o projeto dará oportunidade de integrá-los para a obtenção dos resultados propostos. E a aplicação destes conhecimentos em um projeto interdisciplinar contribuirá para complementar minha formação e os resultados alcançados servirão de experiência profissional.

Este projeto foi escolhido em função das oportunidades de utilização de vários recursos de automação que serão utilizados para o seu desenvolvimento.

1.3 - Cronograma do Projeto

O quadro 1 mostra o cronograma seguido para a conclusão do Projeto.

Tarefa/Mês	1	2	3	4	5	6	7	8	9	10	11	12
Pesquisa Bibliográfica	X	X	X	X	X	X	X	X	X	X		
Aquisição ou confecção de peças para montar a estrutura no protótipo	X	X	X	X	X	X	X	X	X			
Escrita dos capítulos 2 e 3		X	X	X								
Programação			X	X	X	X	X	X	X			
Desenvolvimento do Circuito				X	X	X	X	X				
Escrita e entrega do Artigo			X	X								
Apresentação do Artigo					X							
Montagem estrutural do protótipo			X	X	X	X	X	X	X			
Período de testes					X	X	X	X	X			
Escrita do capítulo 4						X	X	X				
Escrita dos demais capítulos								X	X	X		
Defesa											X	

Quadro 1 - Cronograma das atividades. Fonte: do Autor

2 - FUNDAMENTAÇÃO TEORICA

2.1 - Robótica

Segundo Carrara citando Groover (1988), foi Karel Capek, novelista e escritor de uma peça teatral na antiga Tchecoslováquia, quem usou o termo robô pela primeira vez, em 1920. A palavra “robota”, que significa serviço compulsório ou atividade forçada que deu origem a palavra “robot” em inglês e que traduzido para o português como “robô”.

> Diversos filmes de ficção cientifica mostraram robôs produzidos com o comportamento e a forma humana, levando muitos jovens a pesquisar e desenvolver robôs para o mundo real. Com o surgimento dos computadores na metade do século, iniciaram-se especulações em termos da capacidade de um robô pensar e agir como um ser humano. No entanto, os robôs foram, neste período, criados especialmente para executarem tarefas difíceis, perigosas e impossíveis para um ser humano. Por outro lado, eles não eram projetados com a capacidade de criar ou executar processos que não lhes foram ensinados ou programados. Assim sendo, foram as indústrias que mais se beneficiaram com o desenvolvimento da robótica, aumentando a produção e eliminando tarefas perigosas, antes executadas por seres humanos. (CARRARA).

2.2 - Arduino

Segundo McRoberts (2011) um Arduino é um pequeno computador que se pode programar para processar entradas e saídas entre o dispositivo e os vários componentes externos que são conectados a ele. O Arduino pode ser denominado de plataforma de computação física ou embarcada, isto é, um sistema que pode interagir com seu ambiente por meio de hardware e software.

> Por exemplo, um uso simples de um Arduino seria para acender uma luz por certo intervalo de tempo, digamos, 30 segundos, depois que um botão fosse pressionado. Nesse exemplo, o Arduino teria uma lâmpada e um botão conectados a ele. O Arduino aguardaria pacientemente até que o botão fosse pressionado; uma vez pressionado o botão, ele acenderia a lâmpada e iniciaria a contagem. Depois de contados 30 segundos, apagaria a lâmpada e aguardaria um novo apertar do botão. Você poderia utilizar essa configuração para controlar uma lâmpada em um closet, por exemplo. Esse conceito

> poderia ser estendido pela conexão de um sensor, como um sensor de movimento PIR (*Passive Infrared*), para acender a lâmpada quando ele fosse disparado. Esses são alguns exemplos simples de como você poderia utilizar um Arduino. (McROBERTS, 2011).

A Figura 1 mostra o Arduino que pode ser utilizado para desenvolver objetos interativos independentes, ou mesmo ser conectado a varias situações, tais como: a um computador, a uma rede ou a Internet, permitindo recuperar e enviar dados do Arduino e atuar sobre eles. Como exemplo, "ele pode enviar um conjunto de dados recebidos de alguns sensores para um site, dados estes que poderão, assim, ser exibidos na forma de um gráfico". (McROBERTS, 2011).

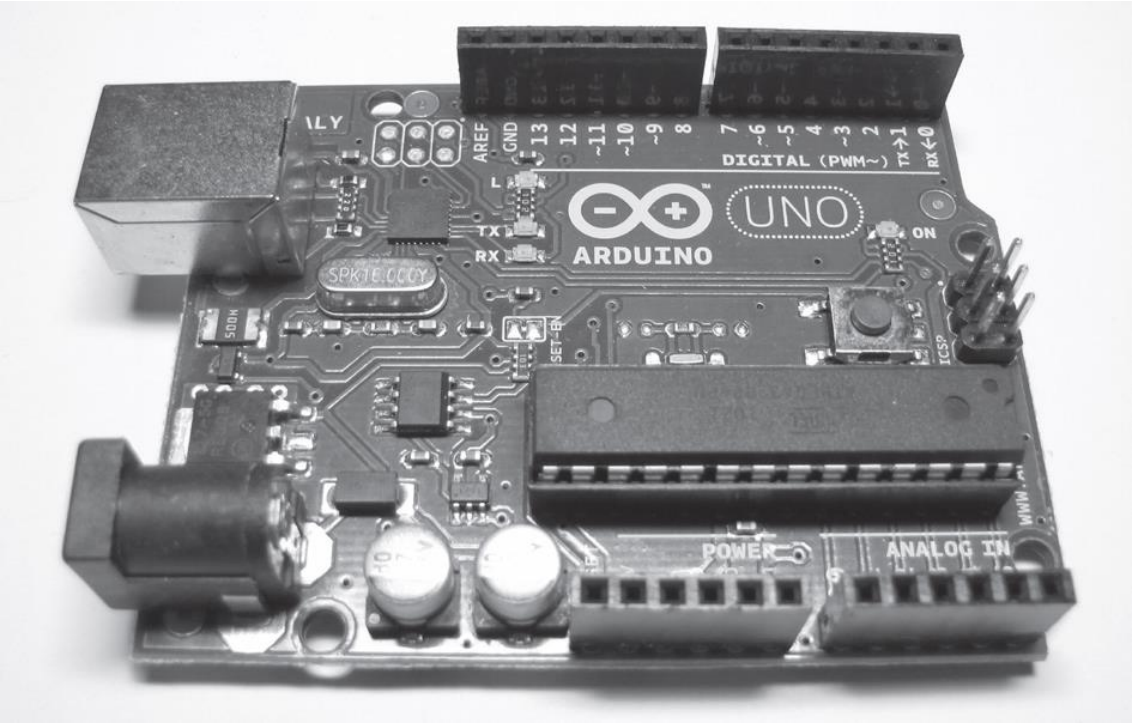

Figura 1 - Arduino UNO. Fonte: McRoberts (2011)

> A placa do Arduino é composta de um microprocessador Atmel AVR, um cristal ou oscilador (relógio simples que envia pulsos de tempo em uma frequência especificada, para permitir sua operação na velocidade correta) e um regulador linear de 5 volts. Dependendo do tipo de Arduino que você utiliza, ele também pode ter uma saída USB, que permite conectá-lo a um Computador ou Mac para *upload* (enviar dados) ou recuperação dos dados. A placa expõe os pinos de entrada/saída do microcontrolador, para que você possa conectá-los a outros circuitos ou sensores. (McROBERTS, 2011).

> Os microcontroladores AVR foram desenvolvidos na Noruega em 1995 e são produzidos pela ATMEL. Apresentam ótima eficiência de processamento e núcleo compacto (poucos milhares de portas lógicas), possuem uma estrutura RISC avançada, com mais de uma centena de instruções e uma arquitetura voltada à programação C, a qual permite produzir códigos compactos. Por causa de sua arquitetura, o desempenho do seu núcleo de 8 bits é equivalente ao desenvolvido por microcontroladores de 16bits (OKI; MONTOVANI, 2013).

Segundo Oki e Montovani (2013), algumas características dos microcontroladores AVR são:

- Executam poderosas instruções em um simples ciclo de clock e operam com tensões entre 1,8 e 5,5 V, com velocidades de até 20 MHz. Estão disponíveis em diversos encapsulamentos (de 8 até 64 pinos);
- Alta integração e grande número de periféricos com efetiva compatibilidade entre toda a família AVR ;
- Possuem vários modos para redução do consumo de energia;
- Possuem 32 registradores de propósito geral, memória de acesso loadstore e a maioria das instruções é de 16bits;

2.3 - Arduino Mega

De acordo com o site laboratório de garagem a definição de Arduino Mega é a seguinte: é uma placa baseada no microcontrolador ATmega2560. Este contém 54 portas digitais de entrada/saída pinos (dos quais 14 podem ser usados como saídas de modulação por largura de pulso), 16 entradas analógicas, existe nele um oscilador de cristal16 MHz, uma conexão USB, 4 portas seriais de hardware, um *plug* de energia, conexão ICSP (In Circuit Serial Programmer), e um botão de reset. (LABORATÓRIO).

> Este é o novo Arduino Mega 2560 (Figura 2). Além de todos os recursos da placa anterior, o Mega 2560 usa um ATmega16U2 em vez de o chip FTDI (*Future Technology Devices International*). Este permite taxas de transferência mais rápidas, tem a capacidade de ter a placa aparecendo como um teclado, mouse, joystick, etc. Ele também tem o dobro de memória. (LABORATÓRIO).

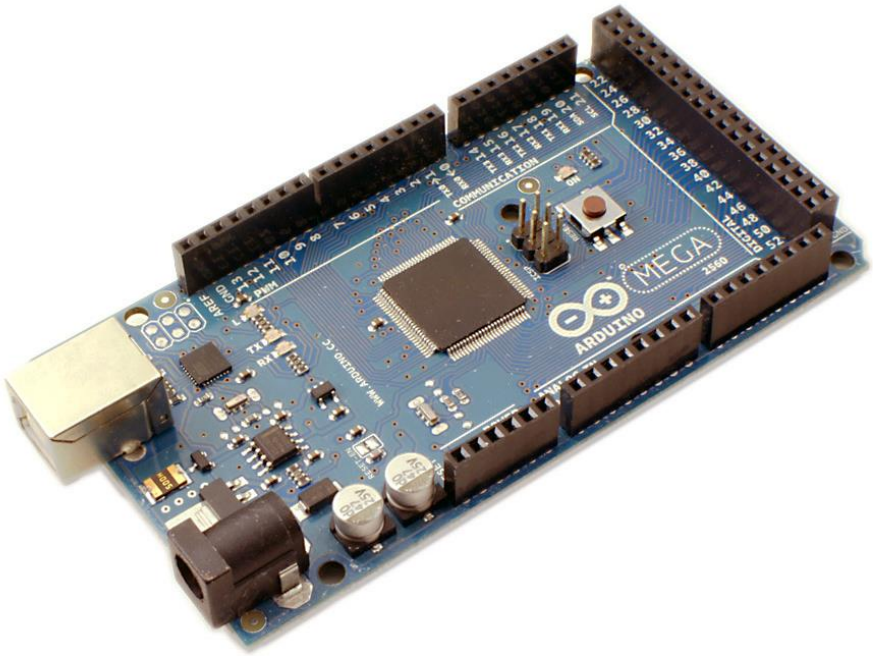

Figura 2 - Arduino Mega. Fonte: Liquidware (2014).

2.4 - Motor de passo

De acordo com Felipe Gonçalves Brites e Vinicius Puga da Almeida Santos Motores de Passo são dispositivos eletromecânicos que convertem pulsos elétricos em movimentos mecânicos que geram pequenas variações angulares. Os passos ocorrem quando o eixo de um motor de passo é rotacionado em pequenos incrementos angulares, somente com a aplicação de pulsos elétricos em determinada sequencia nos terminais pode se notar os passos do motor de passo. (BRITES; SANTOS, 2008).

> A rotação de tais motores é diretamente relacionada aos impulsos elétricos que são recebidos, bem como a sequência a qual tais pulsos são aplicados reflete diretamente na direção a qual o motor gira. A velocidade que o rotor gira é dada pela frequência de pulsos recebidos e o tamanho do ângulo rotacionado é diretamente relacionado com o número de pulsos aplicados. (BRITES; SANTOS, 2008).

2.4.1 Ponte H

> É um circuito eletrônico que permite que um motor rode tanto para um sentido quanto para o outro. Estes circuitos são geralmente utilizados em robótica e estão disponíveis em circuitos prontos ou podem ser construídos por componentes. O nome ponte H é dado pela forma que assume o circuito quando montado. O circuito é construído com quatro "chaves" (S1-S4) que são acionadas de forma alternada (S1 e S4 ou S2 e S3). Para cada configuração das chaves o motor gira em um sentido. As chaves S1 e S2 assim como as chaves S3 e S4 não podem ser ligadas ao mesmo tempo pois podem gerar um curto circuito. Para construção da ponte H pode ser utilizado qualquer tipo de componente que simule uma chave liga-desliga como transistores, relés, mosfets. Para que o circuito fique protegido, é aconselhável que sejam configuradas portas lógicas com componentes 7408 e 7406 a fim de que nunca ocorram as situações de curto circuito descritas acima. Outro melhoramento que pode ser feito à ponte, seria a colocação de diodos entre as "chaves", pois quando a corrente não tem onde circular, no caso de o motor parar, ela volta para a fonte de alimentação economizando assim o gasto de energia de uma bateria por exemplo. (BRITES; SANTOS, 2008).

A figura 3 mostra como é o circuito de uma ponte H.

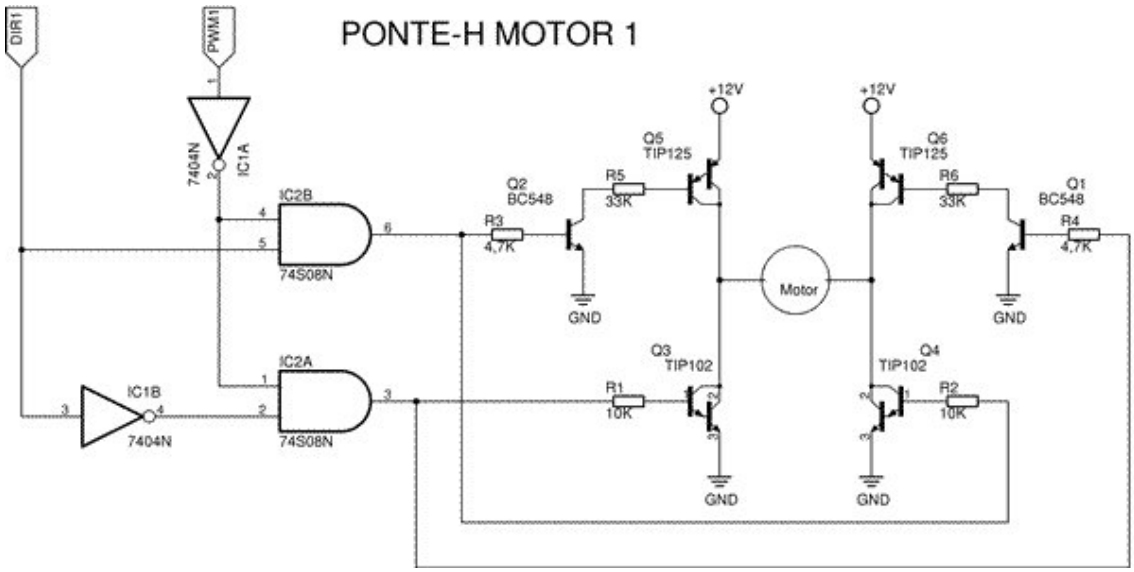

Figura 3 - Ponte H. Fonte: Brites; Santos (2008, p.9).

2.4.2 Tipos de Motores Existentes

2.4.2.1 Quanto a sua estrutura:

Quanto a sua estrutura, eles podem ser de Relutância variável, de Imã permanente e Híbrido. A diferença entre eles é que os motores de relutância variável não contem campo magnético permanente, os motores de imã permanente tem melhor torque que os de relutância variável, pois seus polos magnetizados tem uma maior intensidade de fluxo magnético e os motores de passos híbridos são uma misturo entre a mecânica sofisticada do projeto do motor de relutância variável junto com a potência do imã permanente, fazendo com que este seja mais preciso nos passos que os citados anteriores.

a) Relutância Variável

O motor de relutância variável consiste de um rotor de ferro, com um estator com enrolamentos e com múltiplos dentes. Com a energização dos enrolamentos do estator através de uma corrente DC seus pólos ficam magnetizados. Os passos neste tipo de motor ocorrem quando os dentes do estator são atraídos para os polos do estator energizado, devido à força de campo magnético gerada, para que o sistema tenha o circuito com menor relutância. (BRITES; SANTOS, 2008).

A figura 4 apresenta o esquema de um motor de relutância variável.

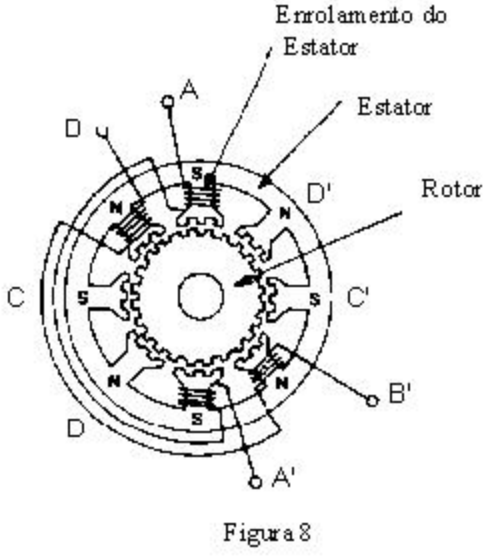

Figura 8

Figura 4 - Motor de relutância variável. Fonte: Brites; Santos (2008).

b) Imã Permanente

A figura 5 mostra um motor de imã permanente.

> Motores de imã permanente tem baixo custo e baixa resolução, com passos típicos de 7, 5° a 15° (48 - 24 passos/revolução). O rotor é construído com imãs permanentes e não possui dentes. Os pólos magnetizados do rotor provém uma maior intensidade de fluxo magnético e por isto o motor de ımã permanente exibe uma melhor característica de torque, quando comparado ao de relutância variável. (BRITES; SANTOS, 2008).

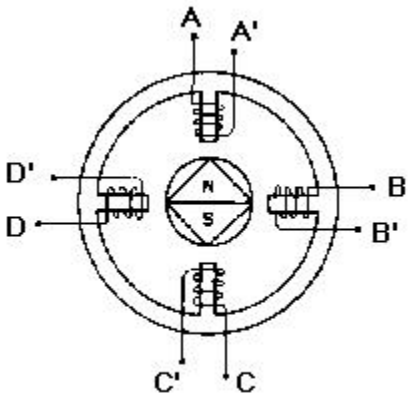

Figura 7

Figura 5 - Motor de ımã permanente. Fonte: Brites; Santos (2008).

c) Híbrido

Este provém melhor desempenho com respeito à resolução de seus passos, velocidade e torque, fazendo com que seu custo seja maior que o do motor de imã permanente. O motor de passo híbrido (Figura 6) é uma combinação das melhores características dos motores de imã permanente e motor de relutância variável. O rotor contém um imã permanente ao redor do seu eixo sendo multidentado como no motor de relutância variável. (BRITES; SANTOS, 2008).

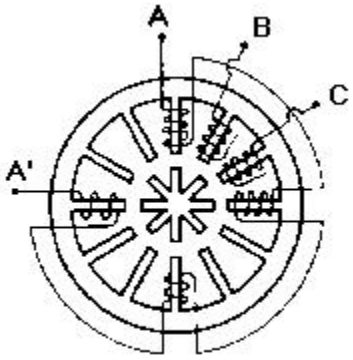

Figura 6

Figura 6 - Motor Híbrido. Fonte: Brites; Santos (2008).

2.4.2.2 Quanto a sua forma de operação

Quanto a sua forma de operação, eles podem ser Unipolares e Bipolares. A diferença é que o motor de passo bipolar contém apenas um enrolamento por fase e o motor de passo bipolar tem dois enrolamentos por fase.

a) Motores Unipolares

Um motor é considerado unipolar por conter dois enrolamentos por fase, sendo uma para cada sentido da corrente. Considerando este arranjo, o pólo magnético pode ser invertido sem comutar o sentido da corrente, logo o circuito da comutação pode ser feito de forma muito simples, como por exemplo usando um único transistor para cada enrolamento. "Tipicamente, dado uma fase, um terminal de cada enrolamento é feito como terra: dando três ligações por fase e seis ligações para um motor bifásico típico. Frequentemente, estas terras comuns bifásicas são juntadas internamente, assim o motor tem somente cinco ligações". (BRITES; SANTOS, 2008).

A Figura 7 contem ilustrações de como é um motor unipolar.

> A resistência entre o fio comum e o fio de excitação da bobina é sempre metade do que entre os fios de excitação da bobina. Isto é, devido ao fato de que há realmente duas vezes o comprimento da bobina entre as extremidades e somente meio comprimento do centro (o fio comum) à extremidade. Os motores de passo unipolares com seis ou oito fios podem ser conduzidos usando excitadores bipolares deixando as terras comuns da fase desconectadas, e conduzindo os

dois enrolamentos de cada fase junto. É igualmente possível usar um excitador bipolar para conduzir somente um enrolamento de cada fase, deixando a metade dos enrolamentos não utilizada. (BRITES; SANTOS, 2008).

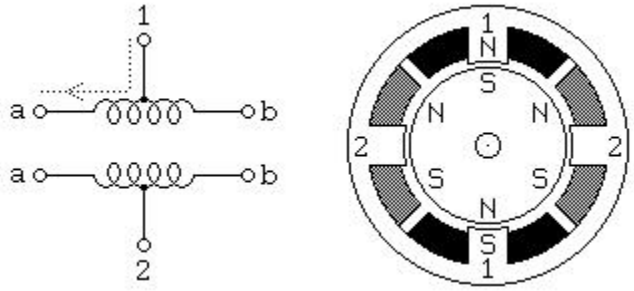

Figura 7 - Motor Unipolar. Fonte: Brites; Santos (2008).

b) Motores Bipolares

Motores são considerados bipolares (Figura 8) por conterem um único enrolamento por fase. Para que ocorra a inversão de um pólo magnético é necessária à inversão da corrente do enrolamento, assim o circuito de condução se torna mais complicado, precisando de um arranjo de ponte H. Existem duas ligações por fase, onde estas não estão em comum.

Os motores bipolares tem um único enrolamento por fase. A corrente em um enrolamento precisa ser invertida a fim de inverter um pólo magnético, assim o circuito de condução é um pouco mais complicado, usando um arranjo de ponte H. Há duas ligações por fase, nenhuma está em comum. "Os efeitos de estática da fricção que usam uma ponte são observadas em determinadas topologias de movimentação. Como os enrolamentos são melhor utilizados, são mais poderosos do que um motor unipolar do mesmo peso". (BRITES; SANTOS, 2008).

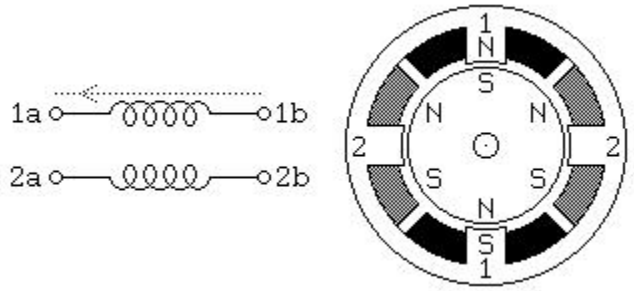

Figura 8 - Motor Bipolar. Fonte: Brites; Santos (2008).

2.5 - Linguagem De Programação

Uma linguagem de programação é um método definido para expressar instruções para um computador, isto é um conjunto de regras semânticas e sintáticas usadas para construir um programa de computador. "Uma linguagem permite que um programador especifique precisamente sobre quais dados um computador vai atuar, como estes dados serão armazenados ou transmitidos e quais ações devem ser tomadas sob várias circunstâncias". (DIGITALDEV).

A linguagem utilizada para a criação do código do cofre de segurança é específica do Arduino, está é baseada nas linguagens C e C++.

2.5.1 Tipos de Linguagens

2.5.1.1 Linguagens Compiladas

"São as linguagens que passam por um processo de tradução (compilação), sendo transformadas para um segundo código (código de maquina) compreensível ao processador, onde o programa responsável por essa tradução é chamado de compilador". (DIGITALDEV).

2.5.1.2 Linguagens interpretadas

> São linguagens em que o código fonte da mesma é executado por outro programa de computador chamado interpretador, que em seguida é executado pelo sistema operacional ou processador. Mesmo que um código em uma linguagem passe pelo processo de compilação, a linguagem pode ser considerada interpretada, se o programa

> resultante não for executado diretamente pelo sistema operacional ou processador. (DIGITALDEV).

2.5.1.3 Linguagens de alto nível

Este tipo de linguagem tem um nível de abstração relativamente alto, mais próximo à linguagem humana e mais longe do código de máquina. Desse modo, as linguagens de nível elevado não estão diretamente relacionadas à arquitetura do computador. "O programador de uma linguagem de alto nível não precisa conhecer características do processador, como instruções e registradores. Essas características são abstraídas na linguagem de alto nível". (DIGITALDEV).

2.5.1.4 Linguagens de baixo nível

Segundo o site DigitalDev este tipo de linguagem de programação compreende-se as características da arquitetura do computador. Onde utilizam-se somente instruções do processador, mas para isso ocorrer é necessário conhecer os registradores do computador. "Nesse sentido, as linguagens de baixo nível estão diretamente relacionadas com a arquitetura do computador. Um exemplo é a linguagem Assembly, que trabalha diretamente com os registradores do processador, manipulando dados". (DIGITALDEV).

2.6 - Display LCD

Os módulos LCD são interfaces de saída muito útil em sistemas microprocessados, onde informações enviadas pelo Arduino são exibidas pelo Display de LCD.

O quadro 2 a seguir descreve cada pino do módulo ou do display para conexão deste a outras placas:

Pino	Função	Descrição
1	Alimentação	Terra ou GND
2	Alimentação	VCC ou +5V
3	V0	**Tensão para ajuste de contraste**
4	RS Seleção:	1 - Dado, 0 - Instrução
5	R/W Seleção:	1 - Leitura, 0 - Escrita
6	E Chip select	1 ou (1 → 0) - Habilita, 0 - Desabilitado
7	B0 LSB	Barramento de Dados
8	B1	
9	B2	
10	B3	
11	B4	
12	B5	
13	B6	
14	B7 MSB	
15	A (qdo existir)	Anodo p/ *LED backlight*
16	K (qdo existir)	Catodo p/ *LED backlight*

Quadro 2 - Descrição dos pinos de um LCD. Fonte: Barbacena; Fleury (1996, p.3).

O LCD requer do microcontrolador mais 3 linhas de controle: a linha de habilitação *Enable*, a linha de escrita e leitura (R/W) e a linha de seleção do registrador (RS). (MICROCHIP).

A linha E (pino 6): permite a ativação do display e a utilização das linhas escrita e leitura (R/W) e a linha de seleção do registrador (RS). Quando a linha de habilitar (E) está a nível baixo, então o *display* LCD fica inibido e ignora os sinais de escrita e leitura e os de deleção do registrador. "Quando (E) está a nível alto, o LCD verifica os estados das duas linhas de controle e reage de acordo com estes". (MICROCHIP).

A linha de escrita e leitura (pino 5): "determina o sentido dos dados entre o microcontrolador e o LCD. Quando está a nível baixo, os dados estão a ser escritos no LCD. Quando está a nível alto, os dados estão a ser lidos do LCD". (MICROCHIP).

A linha RS (pino 4): com a ajuda da linha desta linha, o LCD irá interpretar o tipo de dados das linhas de dados. "Quando está a nível baixo, está a ser escrita uma instrução no LCD. Quando está a nível alto é um caractere que está a ser escrito no LCD". (MICROCHIP).

Segundo Microchip o estado lógico nas linhas de controle, são os seguintes:

E: - 0 Acesso ao LCD inibido

- 1 Acesso ao LCD habilitado

R/W: - 0 Escrever dados no LCD

- 1 Ler dados do LCD

RS: - 0 Instrução

- 1 Caractere

Segundo Microchip a escrita dos dados no *display* de LCD é feita em várias etapas, estas são:

Colocar o bit R/W a nível baixo.

Colocar o bit RS a nível lógico 0 (instrução) ou a nível lógico 1 (caractere).

Colocar o dado na linha de dados (se for uma operação de escrita).

Colocar a linha E a nível alto.

Colocar a linha E a nível baixo.

2.7 - Fonte de Alimentação

É um circuito indispensável para o funcionamento do hardware. Para que o sistema esteja sempre em funcionamento. A fonte escolhida para a alimentação do projeto foi a de um carregador de celular da samsung, esta tem entrada bivolt, fornecendo de saida uma tensão de 5 volts e uma corrente de 0,7 amperes.

2.8 - Keypad 4x4

É um teclado alfanumérico (Figura 9) que será responsável pela comunicação entre o usuário e o microcontrolador. O teclado escolhido é um teclado matricial de 16 teclas, onde existem teclas de 0 a 9, letras de A ate D e duas teclas especiais "*" e "#".

Este terá a funcionalidade de receber a senha inserida pelo cliente do banco, onde o Arduino terá a função de verificar se a senha fornecida por este é valida, caso seja valida o cofre de segurança irá se movimentar ate a posição da senha fornecida através de movimento circular horizontal.

Figura 9 - Keypad 4x4. Fonte: Minikits (2014).

3 - TRABALHOS RELACIONADOS

Podemos citar como trabalhos relacionados aqueles que necessitam da utilização de microcontrolador e motores sendo estes de passo ou servos, com o objetivo de realizar uma determinada tarefa. Como exemplo, podemos citar: empilhadeiras e braços mecânicos.

3.1 - Empilhadeira

É considerado um equipamento que é utilizado na movimentação de diversos tipos de mercadorias. Há diferentes tipos e modelos, tais como: hidráulicas, manuais, semi-elétricas, retráteis, elétricas e tracionarias. (TUDO, 2013).

3.1.1 - Empilhadeira Elétrica

De acordo com o site Tudo Sobre Empilhadeiras estas são fabricadas para serem operadas em locais fechados, como por exemplo: galpões, depósitos, armazéns, etc. Estas são de tamanho pequeno na maioria, porque são utilizadas para realizar trabalhos em corredores, "geralmente possuem uma torre de elevação aumentando a capacidade de armazenagem e estocagem em prateleiras". (TUDO, 2013).

> Como o próprio nome diz, são movidas a eletricidade, sua principal fonte de energia são baterias tracionarias, normalmente de 48 volts e trabalham silenciosamente, fator de muita importância em qualquer local produtivo, diminuindo grande parte de barulhos operacionais. Possuem grande grau de rotação possibilitando manobras em seu próprio eixo. (TUDO, 2013).

As empilhadeiras elétricas (Figura 10) são recomendadas para o uso em lugares pequenos, pois este modelo foi projetado para que os operadores possam trabalhar em locais com pouco espaço e com uma melhor precisão. (TUDO, 2013).

Figura 10 - Empilhadeira Elétrica. Fonte: JBS Empilhadeiras (2014).

3.2 - Braço Mecânico

São protótipos que são elaborados principalmente para a substituição de trabalhos humanos, pois estes braços executam tarefas com mais rapidez, evitam acidentes e fazem o serviço com mais eficiência.

3.2.1 Funcionamento

De acordo com Rodrigo Bernardo Moreira o braço mecânico tem a função de executar movimentos como, por exemplo: conseguir transferir um determinado objeto de um lugar para outro sendo este instruído pelo controlador.

> Na extremidade do braço existe um atuador usado pelo robô na execução de suas tarefas. Todo braço de robô é composto de uma série de vínculos e juntas, onde a junta conecta dois vínculos permitindo o movimento relativo entre eles, [...]. Todo robô possui uma base fixa e o primeiro vínculo está preso a esta base. A mobilidade dos robôs depende do número de vínculos e articulações que o mesmo possui. (MOREIRA).

A figura 11 mostra um exemplo ilustrado de uma junta conectada a dois vínculos que permite a movimentação deste conforme apresentado por Rodrigo Bernardo Moreira.

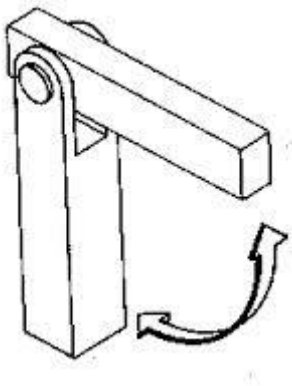

Figura 11 - Junta entre dois vínculos. Fonte: Tibério (2014).

4 - DESENVOLVIMENTO

Nesta parte serão apresentadas todas as fases do desenvolvimento do cofre de segurança, códigos e também serão apresentadas as peças e onde elas foram adquiridas.

4.1 - Análises de motores

Para as escolhas dos motores ideais para o bom funcionamento do Cofre de segurança, teve que analisar cada motor de acordo com seu torque, tamanho e rotação. Com isso tudo se pode encontrar o motor ideal. O primeiro motor a ser analisado, foi o servo motor 9g *Tower Pro Sg90* conforme a Figura 12.

Figura 12 - TowerPro SG90 9G. Fonte: Mercado Livre (2014a).

Este servo motor possui torque de 1 kg/cm, trabalha em 5V e pesa apenas 9g, seu custo é em torno de R$15,00 no mercado brasileiro. Ele foi adquirido através da empresa Laboratório de Garagem (www.labdegaragem.org), ao chegar foi testado, mas foi descartado, pois a sua rotação só atinge no máximo 180 graus e no projeto é necessária uma rotação que chegue a 360 graus, pois o modelo desenvolvido para o cofre exige uma rotação superior a 180 graus..

4.1.1 Motor de passo

Este é mais forte que um servo motor e tem uma rotação livre, onde pode girar no sentido horário e anti-horário quantas vezes for programado. Estes são maiores e com um torque superior.

Através de consultas foi escolhido o Motor de Passo Nema 17 4kgf 17hs5425 unipolar (Figura 13). Este consegue dar 200 passos em uma volta, dando uma simetria na rotação; seu torque é de 4Kg força, uma tensão de 3,1V e contem 4 fios de saída. Seu custo é de R$70,00 e foi adquirido através de um anuncio no site da empresa Mercado Livre (www.mercadolivre.com.br).

Figura 13 - Motor de passo. Fonte: Mercado Livre (2014b).

A Figura 14 apresenta as dimensões do Motor adquirido:

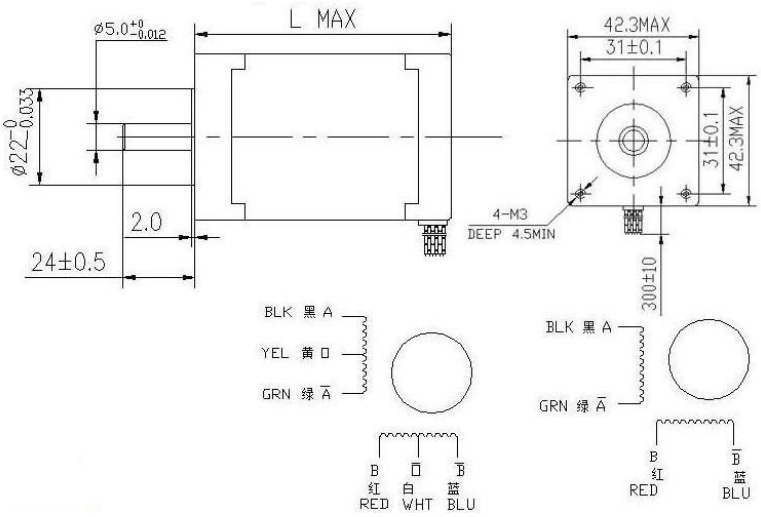

Figura 14 - Dimensão: Motor de passo. Fonte: Mercado Livre (2014b).

Com o motor em mãos foi realizado a montagem com Arduino e o *EasyDriver* e verificou-se o funcionamento, o que permitiu a definição pela compra do motor, já que este estava adequado com a necessidade.

Esquema apropriado para o circuito ficou conforme a figura 15.

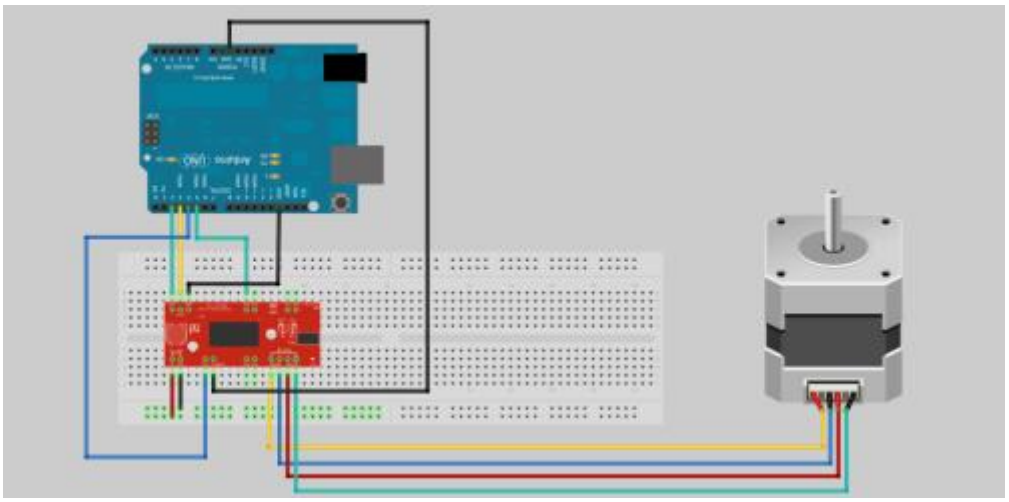

Figura 15 - Circuito de controle de um Motor de passo. Fonte: Laboratório de Garagem (2014).

De acordo com as características especificadas anteriormente, foi encontrado o motor que atende as necessidades do projeto definido para o Trabalho de Conclusão de Curso e pode-se dizer que estas necessidades são:

- Rotação superior ou igual a 360 graus;
- Torque necessário para movimentar a estrutura proposta.

Na Figura 16 é apresentada a montagem do circuito para atender o projeto proposto.

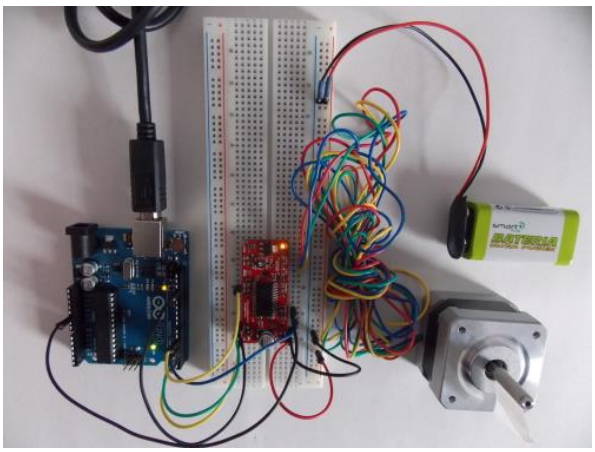

Figura 16 - Montagem do circuito. Fonte: Laboratório de Garagem (2014).

4.2 - EasyDriver

> É um controlador para motores de passo fácil de usar, compatível com qualquer sistema que possa dar saída de um pulso digital de 0 a 5V. Necessita de uma alimentação entre 7 e 30V para alimentar o motor e consegue alimentar motores de passo de qualquer voltagem. Ele possui um regulador de voltagem embarcado para a interface digital que pode ser ajustado para 5 ou 3,3V. Para funcionar, deve-se conectar um motor de passo de 4 cabos e um microcontrolador e será possível ter um controle preciso sobre o motor. O *EasyDriver* pode controlar motores bipolares e motores conectados como bipolares, ou seja, motores com conexões de 4,6 e 8 cabos (MULTILOGICA, 2014).

A figura 17 mostra a placa de circuito do *EasyDriver*.

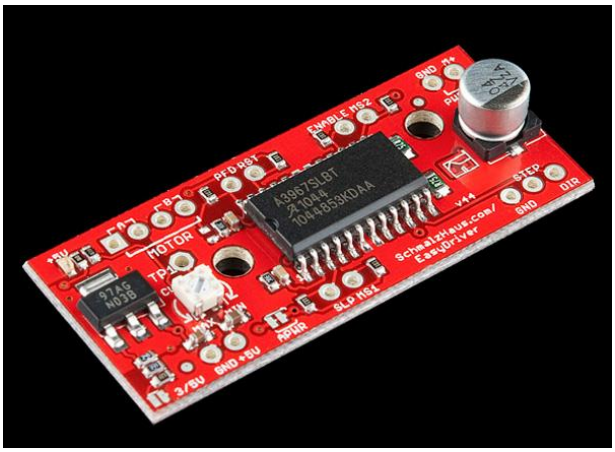

Figura 17 - *EasyDriver*. Fonte: Multilogica (2014).

> Esta é a mais recente versão do *EasyDriver* V4 co-desenvolvida com Brian Schmalz. Ela fornece muito mais flexibilidade e controle sobre motores de passo quando comparada às versões anteriores comercializadas. Os pinos de seleção de micropassos (MS1 e MS2) estão disponíveis permitindo ajustes de micro resolução. Os pinos de *sleep* e *enable* também estão disponíveis para outros controles (MULTILOGICA, 2014).

Este foi adquirido a partir da empresa Instituto Digital encontrada no site de pesquisa Mercado Livre. Foram comprados dois *EasyDriver*, sendo que foi usado somente um no projeto do cofre de segurança, o segundo só foi adquirido com o intuito

de que caso o primeiro estragasse, não seria necessário a aquisição de um outro produto.

4.3 - Formas de aquisição dos componentes eletrônicos

Todos os componentes foram comprados no mercado nacional. As ferramentas necessárias para o desenvolvimento e testes do circuito como: multímetro, solda, ferro de solda, sugador de solda, *protoboard*, cabos *jumpers*, bateria de 9V, carregador de celular, Arduino, motor de passo, *Display* LCD, *EasyDriver* , potenciômetro, entre outras peças.

No Quadro 3 estão os custos das peças adquiridas para o projeto.

Itens	Quantidade	Preço(R$)
Arduino Mega	01	89,90
Display LCD	01	25,00
Keypad	01	10,95
Protoboard 850 pontos	01	25,90
Jumper	80	31,80
Potenciômetro	01	4,00
EasyDriver	01	30,00
Bateria 9V	01	14,00
Motor de Passo	01	70,00
Ferro de solda	01	22,00
Multímetro	01	38,00
Total	90	361,55

Quadro 3 – Custo das peças. Fonte: do Autor (2014)

4.4 - Montagem do Motor de passo junto ao Arduino

Na sequência estão apresentadas as etapas desenvolvidas através de imagens, código e os componentes utilizados para montagem do motor de passo ao Arduino. O programa permitirá, através de uma programação, girar o motor de acordo com o ângulo programado.

Para montar o circuito, foi necessária a utilização dos seguintes itens:

- 1x Motor de Passo Nema 17 4kgf 17hs5425 (Figura 18);
- *Jumpers* para a ligação do circuito (Figura 19);
- 1x Arduino Mega;
- 1x *Protoboard*;
- 1x Bateria de celular, para a alimentação do Motor de Passo (Figura 20);
- 1x *EasyDriver* para controlar os passos do Motor (Figura 21);
- *Software* IDE Arduino (Figura 22).

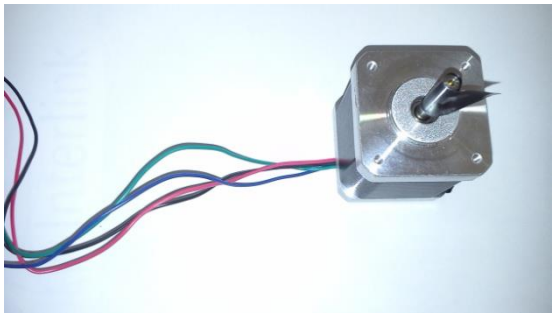

Figura 18 - Motor de Passo. Fonte: do autor.

A Figura 19 mostra os Jumpers necessários para efetuar as ligações entre os componentes do circuito.

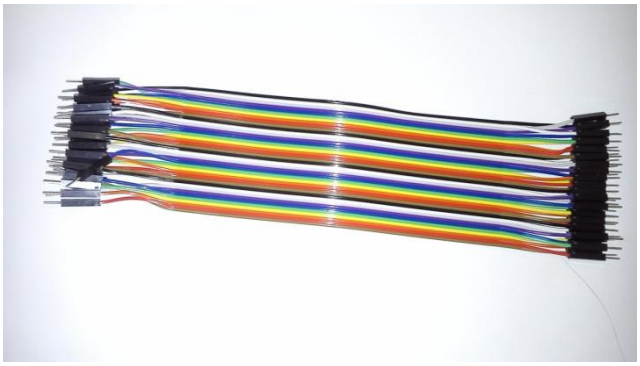

Figura 19 - *Jumpers*. Fonte: do autor.

A Figura 20 mostra o carregador de celular utilizado para fornecer a alimentação de energia que é necessária para que o *EasyDriver* funcione.

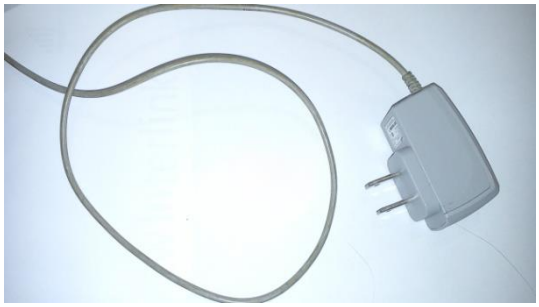

Figura 20 - Carregador de celular. Fonte: do autor.

Figura 21 - *EasyDriver*. Fonte: do autor.

A Figura 22 apresenta a IDE do Arduino e é o local em que é possível desenvolver a comunicação e a programação dele.

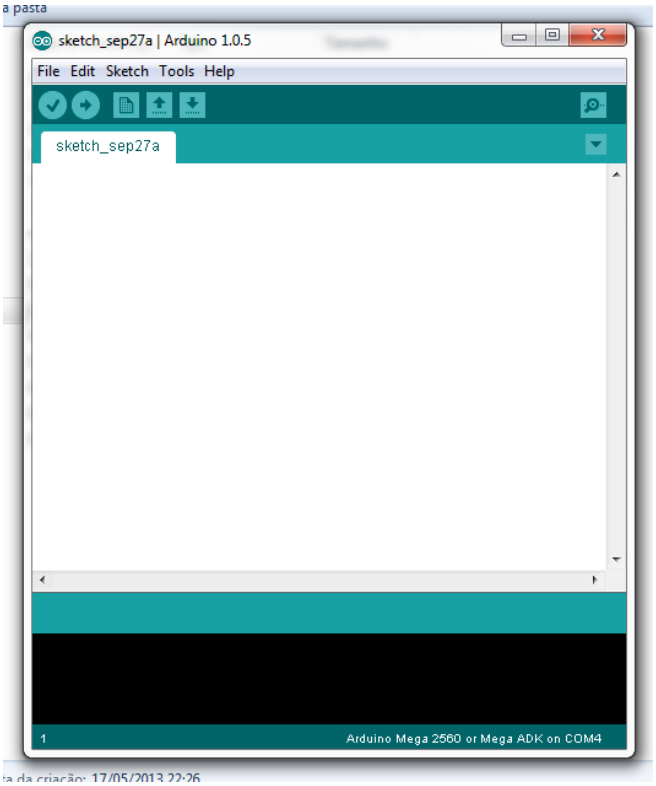

Figura 22 - IDE Arduino. Fonte: do autor.

Primeiramente foi necessária a utilização de um multímetro, para identificar os pares dos fios de saída do Motor de Passo. Em seguida foi inserido o *EasyDriver* na *protoboard* e foi feita a comunicação entre eles através de cabos *jumpers*, logo após foi feita a comunicação deste circuito com o Arduino. Por ultimo foi possibilitada a alimentação do *EasyDriver* com a utilização de um carregador de celular, e fazendo a comunicação do Arduino a IDE através de um cabo *usb* e foi desenvolvido o código necessário para o controle (Figura 23).

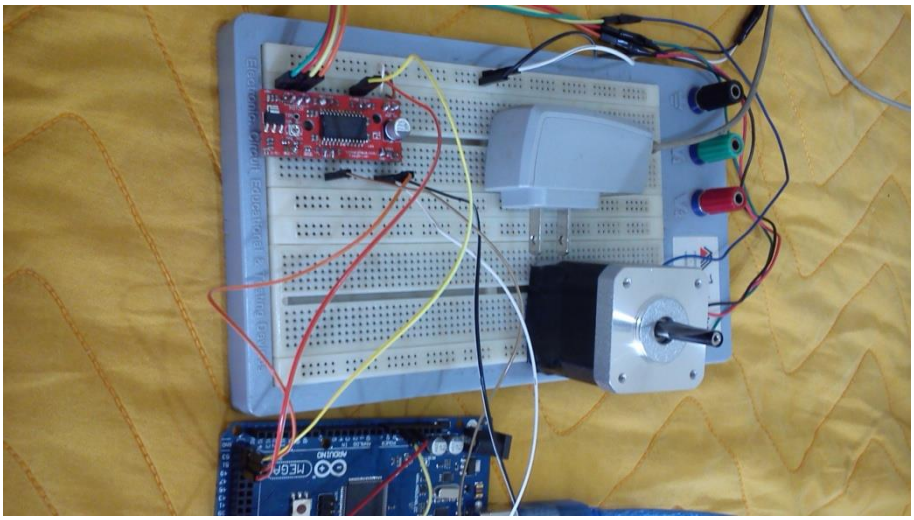

Figura 23 - Circuito de comunicação Arduino, Motor de Passo e *EasyDriver*. Fonte: do autor.

4.4.1 Código

No Quadro 4 é apresentado o código comentado do teste realizado com o circuito identificado na Figura 23.

```
#define step_pin 13    // Define o pino 13 como pino dos passos
#define dir_pin 53    // Define o pino 53 como pino de direção
#define MS1 12    // Define o pino 12 como "MS1"
#define MS2 51      // Define o pino 51 como "MS2"
int direcao;    // Para determinar o sentido do motor
int passos = 200;
    // Número de passos que você deseja executar (para passos completos, 200 = 1 volta)
void setup() {
  pinMode(MS1, OUTPUT);   // Configura "MS1" como saída
  pinMode(MS2, OUTPUT);     // Configura "MS2" como saída
  pinMode(dir_pin, OUTPUT);    // Configura "dir_pin" como saída
  pinMode(step_pin, OUTPUT);    // Configura "step_pin" como saída
  digitalWrite(MS1, LOW);      // Configura divisão de passos do motor (ver acima)
  digitalWrite(MS2, LOW);    // Configura divisão de passos do motor (ver acima)
  digitalWrite(dir_pin, HIGH);
   // Sentido (HIGH = anti-horário / LOW = horário) - Também pode ser alterado
}
void loop() {
while(passos>=0) {    // Enquanto o valor de passos for maior ou igual a zero
      digitalWrite(step_pin, HIGH);
   // Envia nível lógico alto para o pino de passos do motor
      delay(5);    // Aguarda 5ms para o próximo passo
      digitalWrite(step_pin, LOW);
   // Envia nível lógico baixo para o pino de passos do motor
      delay(5);    // Aguarda 5ms para o próximo passo
      passos--;   // Decrementa a variável "passos"

  }
}
```

Quadro 4 – Código-fonte que controla os motores. Fonte: do Autor (2014)

4.5 - Comunicação entre *Display* LCD ao Arduino

Neste tópico serão apresentadas imagens, código-fonte e os componentes utilizados para montagem do *Display* LCD ao Arduino. Após a programação será inserida uma frase na tela do D*i*splay

Para montar o circuito foi necessária a utilização dos seguintes itens:

- Arduino Mega;
- Display LCD;
- Potenciômetro 10 k;
- *Jumpers* para a comunicação.

A Figura 24 apresenta o circuito necessário para ligar o display com o Arduino.

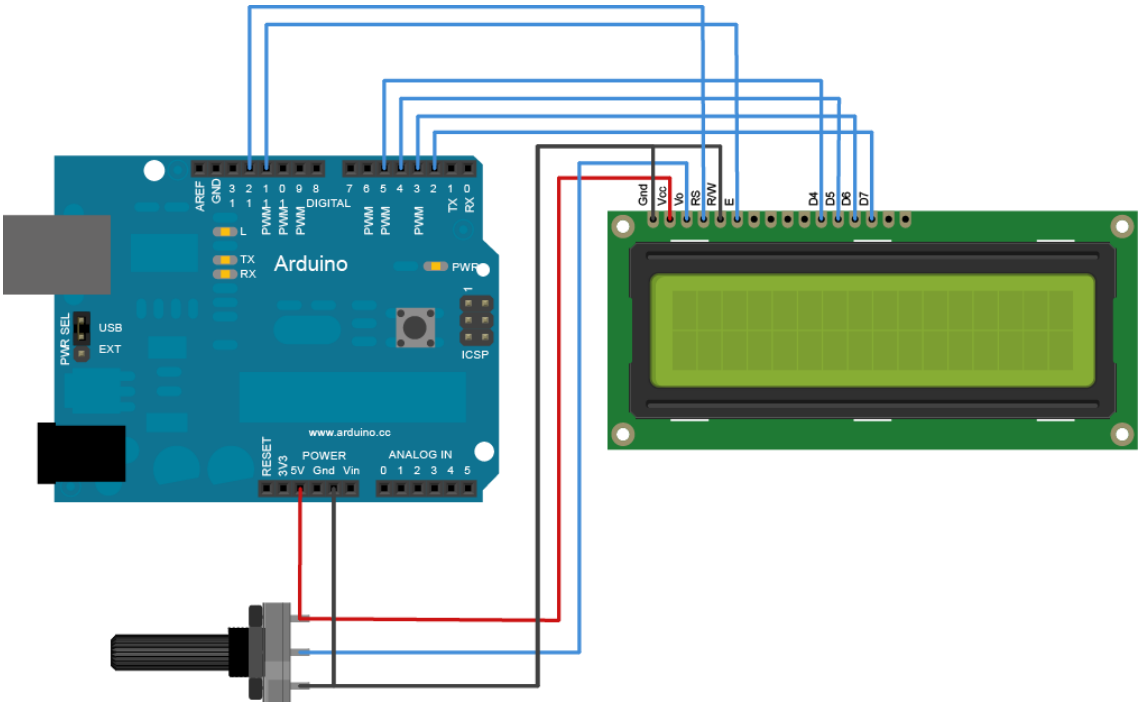

Figura 24 - Circuito de comunicação Arduino ao *Display* LCD. Fonte: Arduino.cc.

Na Figura 25 é mostrada a montagem do circuito usando um protoboard para realizar os testes necessários para a continuidade do projeto.

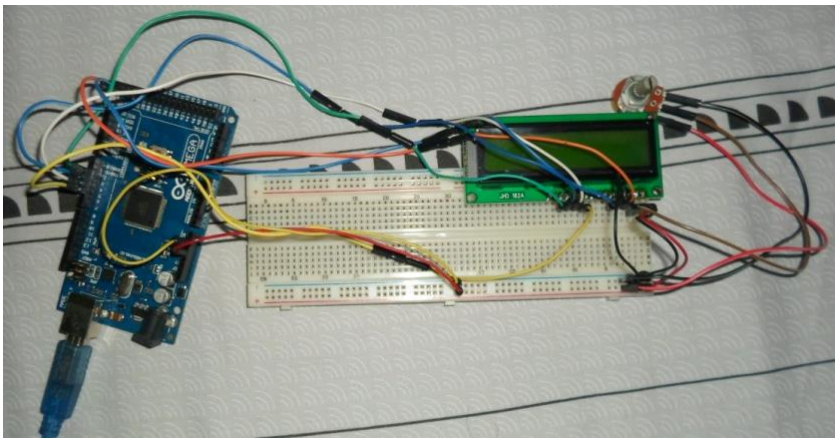

Figura 25 - Circuito de comunicação Arduino ao *Display* LCD. Fonte: do Autor.

4.5.1 Código comunicação LCD ao Arduino

No Quadro 5 está descrito o código-fonte necessário para inicializar e controlar o display a partir do Arduino.

```
#include <LiquidCrystal.h>

// Inicializa a biblioteca do Display LCD
LiquidCrystal lcd(22, 24, 5, 4, 3, 2);

void setup() {
  // Define o numero de linhas e colunas:
  lcd.begin(16, 2);
  // Escreve a mensagem no Display LCD.
  lcd.print("Bem Vindo!");
}

void loop() {
  // Definir o cursor para a coluna 0, linha 1
  // (Nota: a linha 1 é a segunda linha, uma vez que a contagem inicia em 0):
  lcd.setCursor(0, 1);
  // imprimir o número de segundos desde a redefinição:
  lcd.print(millis()/1000);
}
```

Quadro 5 – Código-fonte de controle do display. Fonte: do Autor (2014)

4.6 - Protótipo

A estrutura apresentada nas figura 26, 27 e 28 foram desenvolvidas, com a utilização dos seguintes materiais:

- Tesoura;
- Régua;
- Canetinha preta para marcar cortes;
- Estilete;
- Isopor;
- Cola quente;
- 1 vasilha de plástico.

A figura 26 representa uma visão superior de como estão distribuídas as gavetas neste protótipo, onde existem somente três gavetas de clientes e uma posição inicial que é onde estará a abertura de acesso ao cliente. Esta estrutura está encaixada no motor de passo para permitir a sua movimentação.

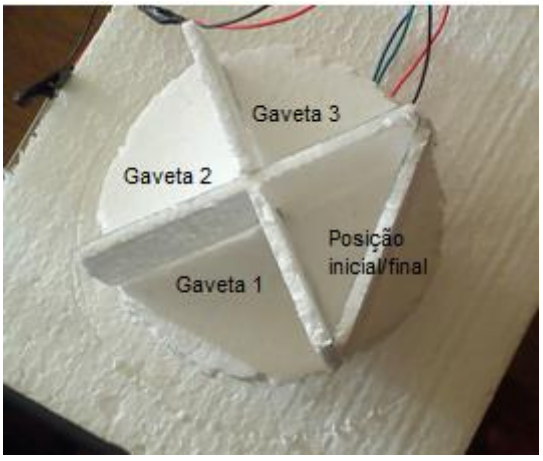

Figura 26 - Estrutura do protótipo com distribuição das gavetas. Fonte: do Autor.

A seguir está apresentada na Figura 27, a estrutura que será acoplada com a estrutura representada na Figura 26, formando o corpo do protótipo final (Figura 28).

Pode-se observar na Figura 27 que a direita da estrutura existe uma abertura, e esta é o local onde o cliente irá retirar sua gaveta.

4.6.1 Código de Inicialização das Variáveis

O Quadro 6 apresenta os valores configurados nos pinos do *EasyDrive*r para a inicialização do motor de passo e contém também o código de inicialização do *display* LCD, onde ao iniciar o funcionamento do projeto irá apresentar a frase "SEJA BEM VINDO" e logo após a mensagem "Digite sua senha".

```
void setup() {
  lcd.begin(16,2);
  lcd.setCursor(0,0);
  lcd.print("SEJA BEM VINDO");
  delay(3000);
  Serial.begin(9600);
  Serial.println("Digite sua senha");
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Digite sua senha:");
  pinMode(MS1, OUTPUT);   // Configura "MS1" como saída
  pinMode(MS2, OUTPUT);     // Configura "MS2" como saída
  pinMode(dir_pin, OUTPUT);    // Configura "dir_pin" como saída
  pinMode(step_pin, OUTPUT);    // Configura "step_pin" como saída
  digitalWrite(MS1, LOW);      // Configura divisão de passos do motor (ver acima)
  digitalWrite(MS2, LOW);    // Configura divisão de passos do motor (ver acima)
  digitalWrite(dir_pin, HIGH);
// Sentido (HIGH = anti-horário / LOW = horário) - Também pode ser alterado
 }
```

Quadro 6 - Código-fonte de Inicialização das Variáveis. Fonte: do Autor.

4.6.2 Código de Acesso a Gaveta

O Quadro 7 mostra o código que será executado caso a senha digitada pelo cliente seja 345 e que a gaveta 0 (posição inicial/final) esteja na posição inicial do protótipo. Se a função2 for verdadeira será apresentada a frase "Gaveta na posição 2", esta função existe para que não seja possível executar o sistema utilizando a mesma senha sem que digite a senha ao contrario para voltar a posição inicial.

A função "resposta(2)" é para mostrar o deslocamento do sistema até a gaveta do cliente, pode se ver isso no código do quadro 8. Após isso é determinado a quantidades de passos que será executado para que se chegue a gaveta 2, no caso do exemplo, são necessários 125 para chegar a gaveta do cliente 2. Já a função *while* é usada para que o motor de passo execute os passos. Todas as duas demais gavetas contém código semelhante a este, só modificando os valores.

```
void resposta(int gaveta)
{
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Indo para Gaveta");
  lcd.setCursor(0,2);
  lcd.print(gaveta);
  Serial.print("Indo para Gaveta: ");
  Serial.println(gaveta);
}
```

Quadro 7 - Código da função resposta. Fonte: do Autor.

O Quadro 8 apresenta o código que controla o funcionamento do motor.

```
else if(valorlido == "345" && gaveta0 == true){
    if(gaveta2 == true)
    {
      Serial.println("Gaveta na posição 2");
    }
    else{
    resposta(2);
    passos = 125;
    while(passos) {                    // Enquanto o valor de passos for maior ou igual a zero
      digitalWrite(step_pin, HIGH);    // Envia nível lógico alto para o pino de passos do motor
      delay(5);                        // Aguarda 5ms para o próximo passo
      digitalWrite(step_pin, LOW);     // Envia nível lógico baixo para o pino de passos do motor
      delay(5);                        // Aguarda 5ms para o próximo passo
      passos--;
    }
```

Quadro 8 - Código principal para o funcionamento do Motor. Fonte: do Autor.

Figura 27 - Parte superior da estrutura. Fonte: do Autor.

A Figura 28 mostra a estrutura final do protótipo que foi montada e posteriormente testada com o código disponível no anexo. Foi testado é sua funcionalidade atendeu a proposta deste trabalho de conclusão.

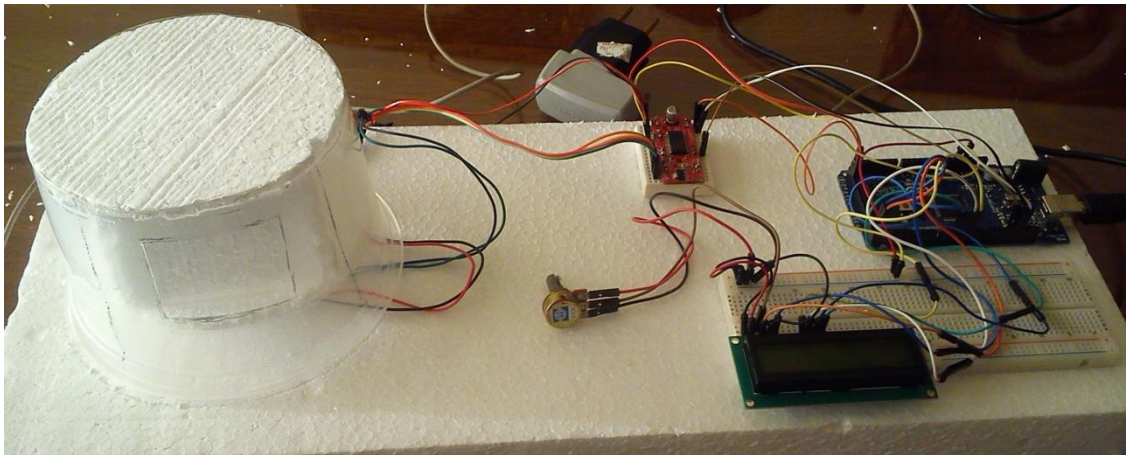

Figura 28 - Protótipo Final montado. Fonte: do Autor.

4.6.3 Exemplificando o funcionamento.

Inicialmente o cliente ao chegar no banco e se dirigi as instalações do cofre de segurança. No painel, ele deverá digitar sua senha e o sistema reconhecerá a gaveta onde estão os seus pertences ou a que lhe foi destinada para sua utilização. O sistema

então fará a movimentação das gavetas e vai disponibilizar o acesso a sua gaveta pré-determinada. E assim ele terá acesso ao conteúdo ou poderá armazenar os seus pertences para ficarem guardados. Para completar a operação, o usuário digita novamente a sua senha e o sistema fará a movimentação voltando à posição inicial. No quadro 9 é possível observar o código de validação de senha do sistema

Considerando que o sistema trabalha somente com o acesso autorizado por senha, a vulnerabilidade existente ocorre se alguém conseguir digitar uma senha válida. Se for digitada uma senha inválida, será apresentada uma mensagem de senha incorreta.

```
   if((valorlido != "123" && valorlido !="") || (valorlido !="345" && valorlido !="") ||
(valorlido !="678" && valorlido !=""))
   {
    Serial.println("Senha incorreta");
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Senha Incorreta!");
    delay(2000);
    Serial.println("Digite sua senha:");
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Digite sua senha:");
   }
```

Quadro 9 - Código-fonte relativo validação da senha. Fonte: do Autor.

4.6.4 Código comunicação LCD, Motor de passo e Arduino

No Anexo esta apresentada todo o código necessário para o funcionamento do protótipo. Este código foi desenvolvido especificamente para o protótipo nas condições necessárias para permitir os testes que contribuem para a exemplificação da proposta apresentada para o Trabalho de Conclusão de Curso. Para generalizar o trabalho, algumas alterações no código se fazem necessárias, tais como o armazenamento das senhas em um Banco de Dados, modificar os testes internos para identificar o usuário de forma a permitir um número maior de gavetas, entre outras alterações.

5 - CONCLUSÕES

O desenvolvimento do projeto proposto para o Trabalho de Conclusão de Curso foi efetuado com algumas alterações que foram identificadas ao longo da execução das atividades, considerando que as mesmas ocorreram por dificuldades detectadas posteriormente a apresentação da proposta.

Algumas dificuldades podem ser enumeradas, tais como, a dificuldade de construir o protótipo proposto com material rígido por não encontrar na cidade os equipamentos necessários para cortar chapas metálicas (ou de alumínio) com precisão e tendo então que se deslocar, gerando gastos financeiros e de tempo. Com a utilização de materiais alternativos que suprem as características imaginadas e mais algumas alterações no modelo inicial proposto, foi possível construir o protótipo.

Outra dificuldade foi com relação ao Teclado alfanumérico que estragou durante os testes e com o tempo exíguo para finalizar os testes, fazendo com que fosse viável a utilização do teclado de um computador, pois até a chegada de um novo, iria atrapalhar na conclusão deste ao prazo estabelecido.

E também ocorreram situações imprevisíveis, pois ao ligar o motor de passo as vezes ele apresenta movimentos involuntários, fazendo com que as gavetas parem em posição diferente do previsto, um pouco além da posição correta.

Com a execução dos testes foi possível avaliar que a proposta é viável e pode ser implementada em uma escala maior, considerando então que o trabalho atingiu o propósito, que é demonstrar a utilização da automação em ações de segurança.

5.1 - Trabalhos Futuros

Dá para construir diversas funcionalidades com a utilização do código de controle do motor de passo, como: Esteira rolante; Busca e movimentação de materiais dentro de uma empresa; entre outros.

Para aperfeiçoar o projeto, pode-se com este código-fonte, continuar o desenvolvimento de um cofre que seja maior e que tenha mais funcionalidades,

utilizando movimentos horizontais e verticais com a utilização de mais motores de passo. Para evitar que as gavetas passem do ponto de parada, em um trabalho futuro, pode-se utilizar sensores, onde ao passar pelo sensor o motor de passo pare.

REFERÊNCIAS:

BARBACENA, Ilton L.; FLEURY, Claudio Afonso. **Display LCD**. 1996.

BRITES, Felipe G.; SANTOS, Vinicius P. A. **Motor de passo**. Niterói: Petele - UFF, 2008.

CARRARA, Valdemir. **Apostila de robótica**. Mogi das Cruzes: Universidade Braz Cubas, 200?.

DIGITALDEV. **Linguagens para que servem?** Disponível em <http://www.digitaldev.com.br/linguagens/>, Acesso em 20 fev. 2014.

JBS. **Empilhadeiras Elétrica em SP**. Disponível em <http://www.jbsempilhadeiras.com.br/empilhadeiras-eletricas-em-sp.html>, Acesso em 21 mar. 2014.

LABORATÓRIO de Garagem. **Arduino Mega 2560 REV3**. Disponível em <http://www.labdegaragem.org/loja/29-Arduino/Arduino-mega-2560.html >, Acesso em 20 fev. 2014.

LIQUIDWARE. **Arduino mega 2560**. Disponível em <http://www.liquidware.com/shop/show/AMEGA2560/Arduino+Mega+2560>, Acesso em 20 fev. 2014.

McROBERTS, Michael. **Arduino básico**. Tradução de Rafael Zanolli. São Paulo: Novatec, 2011.

MERCADO Livre. **Servo Motor 9g Tower Pro Sg90 Micro Servo Arduino Pic Id880**. 2014a. Disponível em < http://produto.mercadolivre.com.br/MLB-577483427-servo-motor-9g-tower-pro-sg90-micro-servo-Arduino-pic-id880-_JM>, acesso em 22 mar. 2014.

MERCADO Livre. **Motor De Passo Nema 17 4kgf 17hs5425 Impressora 3d Reprap**. 2014b. Disponível em < http://produto.mercadolivre.com.br/MLB-540362090-motor-de-passo-nema-17-4kgf-17hs5425-impressora-3d-reprap-_JM?redirectedFromSearch=true>, acesso em 22 mar. 2014.

MICROCHIP. **Introdução aos microcontroladores PIC.** Disponível em <http://www.trajanocamargo.com.br/wp-content/uploads/2012/05/Apostila_Microcontrolado_PIC_16F84.pdf>, Acesso em 20 fev. 2014.

MINIKITS. **16 Key 4x4 Numeric Keypad**. Disponível em <http://www.minikits.com.au/Keypad3>, Acesso em 20 fev. 2014.

MOREIRA, Rodrigo Bernardo. **O Braço Mecânico**. Disponível em <http://www.ebah.com.br/content/ABAAABS20AE/apostila-braco-mecanico>, Acesso em 21 mar. 2014.

MULTILOGICA Shop. **EasyDriver para motor de passo**. 2014. Disponível em <http://multilogica-shop.com/easydriver-para-motor-de-passo>, acesso em 30 mar 2014.

OKI, Nobuo; MONTOVANI, Suely C. A. **TEEE I** - Projeto de Robôs Móveis. Ilha Solteira: Faculdade de Engenharia de Ilha Solteira. 2013. Disponível em <http://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/microcontrolador_atmel_1-1.pdf >, acesso 30 mar. 2014 .

TIBÉRIO, Erico. **Braço Mecânico e a Realização de Trabalho**. 18 jun. 2012. Disponível em < http://embuscadaidentidade.blogspot.com.br/>, Acesso em 21 mar. 2014.

TUDO sobre Empilhadeira. **Você sabe como funciona uma empilhadeira?** 4 fev. 2013. Disponível em <http://www.amelhorempilhadeira.com.br/empilhadeiras/>, Acesso em 21 mar. 2014.

ANEXO

```
#define step_pin 13      // Define o pino 13 como pino dos passos
#define dir_pin 53       // Define o pino 53 como pino de direção
#define MS1 12           // Define o pino 12 como "MS1"
#define MS2 51          // Define o pino 51 como "MS2"
#include <LiquidCrystal.h>
int direcao;                // Para determinar o sentido do motor
int passos;
LiquidCrystal lcd(22, 24, 5, 4, 3, 2);
// Número de passos que você deseja executar (para passos completos, 200 = 1 volta)
String valorlido;

boolean gaveta1 = false;
boolean gaveta2 = false;
boolean gaveta3 = false;

boolean gaveta0 = true;

void setup() {
  lcd.begin(16,2);
  lcd.setCursor(0,0);
  lcd.print("SEJA BEM VINDO");
  delay(3000);
  Serial.begin(9600);
  Serial.println("Digite sua senha");
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Digite sua senha:");
  pinMode(MS1, OUTPUT);          // Configura "MS1" como saída
  pinMode(MS2, OUTPUT);          // Configura "MS2" como saída
  pinMode(dir_pin, OUTPUT);      // Configura "dir_pin" como saída
  pinMode(step_pin, OUTPUT);     // Configura "step_pin" como saída
  digitalWrite(MS1, LOW);          // Configura divisão de passos do motor (ver acima)
  digitalWrite(MS2, LOW);          // Configura divisão de passos do motor (ver acima)
  digitalWrite(dir_pin, HIGH);
// Sentido (HIGH = anti-horário / LOW = horário) - Também pode ser alterado

}

void resposta(int gaveta)
{
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("Indo para Gaveta");
 lcd.setCursor(0,2);
 lcd.print(gaveta);
```

```
  Serial.print("Indo para Gaveta: ");
  Serial.println(gaveta);
}

void loop() {
//-----------------------------------------------------

  valorlido = Serial.readString();

  if(valorlido == "123" && gaveta0 == true){
    if(gaveta1 == true)
    {
      Serial.println("Gaveta na posição 1");
    }
    else{
      resposta(1);
      passos = 50;
      while(passos) {                    // Enquanto o valor de passos for maior ou igual a zero
        digitalWrite(step_pin, HIGH);    // Envia nível lógico alto para o pino de passos do motor
        delay(5);                        // Aguarda 5ms para o próximo passo
        digitalWrite(step_pin, LOW);     // Envia nível lógico baixo para o pino de passos do motor
        delay(5);                        // Aguarda 5ms para o próximo passo
        passos--;
      }
      Serial.println("Digite 321 para voltar");
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Digite 321 para");
      lcd.setCursor(0,2);
      lcd.print("voltar:");
      gaveta1 = true;
      gaveta0 = false;
    }
  }

  else if(valorlido == "321"){
    if(gaveta1 == true)
    {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Indo p/ posição");
      lcd.setCursor(0,2);
      lcd.print("inicial");
      Serial.println("Indo para posicao inicial!");
      digitalWrite(dir_pin, LOW);
      passos = 50;
      while(passos) {                    // Enquanto o valor de passos for maior ou igual a zero
        digitalWrite(step_pin, HIGH);    // Envia nível lógico alto para o pino de passos do motor
```

```
      delay(5);                          // Aguarda 5ms para o próximo passo
      digitalWrite(step_pin, LOW);    // Envia nível lógico baixo para o pino de passos do motor
      delay(5);                          // Aguarda 5ms para o próximo passo
      passos--;
     }
     Serial.println("Digite sua senha:");
     digitalWrite(dir_pin, HIGH);
     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("Digite sua senha:");
     gaveta1 = false;
     gaveta0 = true;
    }
    else
    {
     Serial.println("Gaveta na posição 0");
    }
  }
//------------------------------------------------------------
  else if(valorlido == "345" && gaveta0 == true){
    if(gaveta2 == true)
    {
     Serial.println("Gaveta na posição 2");
    }
    else{
    resposta(2);
    passos = 125;
    while(passos) {                      // Enquanto o valor de passos for maior ou igual a zero
     digitalWrite(step_pin, HIGH);     // Envia nível lógico alto para o pino de passos do motor
     delay(5);                          // Aguarda 5ms para o próximo passo
     digitalWrite(step_pin, LOW);     // Envia nível lógico baixo para o pino de passos do motor
     delay(5);                          // Aguarda 5ms para o próximo passo
     passos--;
    }
    Serial.println("Digite 543 para voltar");
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Digite 543 para");
    lcd.setCursor(0,2);
    lcd.print("voltar:");
    gaveta2 = true;
    gaveta0 = false;
  }

  }

  else if(valorlido == "543"){
    if(gaveta2 == true)
```

```
    {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Indo p/ posição");
    lcd.setCursor(0,2);
    lcd.print("inicial");
    Serial.println("Indo para posicao inicial!");
    digitalWrite(dir_pin, LOW);
    passos = 125;
    while(passos) {                       // Enquanto o valor de passos for maior ou igual a zero
      digitalWrite(step_pin, HIGH);       // Envia nível lógico alto para o pino de passos do motor
      delay(5);                           // Aguarda 5ms para o próximo passo
      digitalWrite(step_pin, LOW);        // Envia nível lógico baixo para o pino de passos do motor
      delay(5);                           // Aguarda 5ms para o próximo passo
      passos--;
    }
    Serial.println("Digite sua senha:");
    digitalWrite(dir_pin, HIGH);
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Digite sua senha:");
      gaveta2 = false;
      gaveta0 = true;
    }
    else
    {
      Serial.println("Gaveta na posição 0");
    }
 }
 //---------------------------------------------------------------
   else if(valorlido == "678" && gaveta0 == true){
    if(gaveta3 == true)
    {
      Serial.println("Gaveta na posição 3");
    }
    else{
    resposta(3);
    passos = 175;
    while(passos) {                       // Enquanto o valor de passos for maior ou igual a zero
      digitalWrite(step_pin, HIGH);       // Envia nível lógico alto para o pino de passos do motor
      delay(5);                           // Aguarda 5ms para o próximo passo
      digitalWrite(step_pin, LOW);        // Envia nível lógico baixo para o pino de passos do motor
      delay(5);                           // Aguarda 5ms para o próximo passo
      passos--;
    }
    Serial.println("Digite 876 para voltar");
    lcd.clear();
    lcd.setCursor(0,0);
```

```
   lcd.print("Digite 876 para");
   lcd.setCursor(0,2);
   lcd.print("voltar:");
   gaveta3 = true;
   gaveta0 = false;
  }
 }

 else if(valorlido == "876"){
   if(gaveta3 == true)
   {
   lcd.clear();
   lcd.setCursor(0,0);
   lcd.print("Indo p/ posição");
   lcd.setCursor(0,2);
   lcd.print("inicial");
   Serial.println("Indo para posicao inicial!");
   digitalWrite(dir_pin, LOW);
   passos = 175;
   while(passos) {                        // Enquanto o valor de passos for maior ou igual a zero
     digitalWrite(step_pin, HIGH);       // Envia nível lógico alto para o pino de passos do motor
     delay(5);                            // Aguarda 5ms para o próximo passo
     digitalWrite(step_pin, LOW);       // Envia nível lógico baixo para o pino de passos do motor
     delay(5);                           // Aguarda 5ms para o próximo passo
     passos--;
   }
   Serial.println("digite sua senha:");
   digitalWrite(dir_pin, HIGH);
   lcd.clear();
   lcd.setCursor(0,0);
   lcd.print("Digite sua senha:");
    gaveta3 = false;
     gaveta0 = true;
   }
   else
   {
     Serial.println("Gaveta na posição 0");
   }
 }

//--------------------------------------------------
  else if(gaveta0 == true)
  {
    if((valorlido != "123" && valorlido !="") || (valorlido !="345" && valorlido !="") ||
(valorlido !="678" && valorlido !=""))
    {
      Serial.println("Senha incorreta");
      lcd.clear();
```

```
      lcd.setCursor(0,0);
      lcd.print("Senha Incorreta!");
      delay(2000);
      Serial.println("Digite sua senha:");
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Digite sua senha:");
    }
  }
}
```