PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO CIENTÍFICA E TECNOLÓGICA
UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARCY RIBEIRO

CENTRO CCT
LABORATÓRIO LCMAT

Relatório do período: 06/2017 a 05/2018

RELATÓRIO ANUAL

Título do Projeto: Desenvolvimento e controle de dispositivos para Internet das Coisas

Título do Plano de Trabalho: Estudo e desenvolvimento de um protótipo para Internet das Coisas com base na plataforma Arduino

Nome da Bolsista: Isabela Correia Pereira
Curso e N° Matrícula: Ciência da Computação / 00115110215
Orientador: Fermín Alfredo Tang Montané

Fonte financiadora da Bolsa: PIBIC / CNPq

1.- Etapas propostas no plano de trabalho

O plano de trabalho proposto compreende as seguintes etapas:

a) Revisão bibliográfica sobre Internet das Coisas, envolvendo a filosofia, as diversas áreas de aplicação, as plataformas de desenvolvimento.
b) Estudo da Plataforma Arduino Uno e afins. Com ênfase nas portas digitais e analógicas e nas possíveis fontes de alimentação. Diagramação de projetos usando ferramentas visuais tais como Fritzing e VBB.
c) Escolha de um projeto específico de desenvolvimento de dispositivo controlado por microcontrolador Arduino. Descrição do projeto escolhido, objetivo, alternativas de desenvolvimento e detalhamento das componentes necessárias.
d) Projeto do protótipo eletrônico com base no Arduino Uno. Elaboração do diagrama de circuito usando ferramenta visual. Prototipação usando breadboard. Definição das funcionalidades e programação do dispositivo. Teste funcional.
e) Projeto da estrutura física de suporte para o protótipo eletrônico. Transferência do circuito para uma placa definitiva. Teste funcional.
f) Desenvolvimento de um aplicativo de controle do dispositivo através da internet. Escolha da linguagem de programação mais adequada. Definição das funcionalidade e programação do aplicativo. Teste funcional.
g) Realização de experimentos de avaliação e desempenho do dispositivo. Análise sobre possíveis aprimoramentos ou reformulações das etapas d) a f).
h) Elaboração de relatório técnico.

Todas as etapas propostas no plano de trabalho foram realizadas de maneira integral com exceção do item f) que foi abordado a partir de um projeto secundário. Foram desenvolvidos dois dispositivos baseados no micro-controlador Arduino. Um jogo de memória baseado em cores usado na primeira fase da pesquisa para treinamento e um protótipo de braço robótico para simular aplicações industriais. O controle do braço robô através da internet ficou inconcluso, uma vez que o tempo disponível resultou muito curto e seria necessário pesquisar mais sobre o assunto. No entanto, trabalhou-se em um projeto secundário, no projeto de controle de lâmpadas através da web, que foi de fato implementado.

2.- Introdução

O número de pessoas conectadas à internet tem crescido rapidamente nos últimos anos e com elas também uma diversidade de serviços prestados através da rede. Dentre os principais serviços devemos destacar o surgimento do comércio eletrônico, a mídia eletrônica, principalmente os serviços de notícias, a produção de conteúdo áudio visual, o streaming de vídeos e músicas, entre outras.

A Internet das Coisas é considerada a terceira revolução ligada à internet, após o surgimento dos dispositivos móveis e a disseminação da rede a nível global. Trata-se da ideia de que diversos dispositivos, muitos deles de uso cotidiano ou não, podem também se conectar à internet e produzir conteúdo e/ou prestar serviços. Além disso, tais dispositivos podem se comunicar entre si. Como exemplo de dispositivos pode-se destacar o uso de sensores de temperatura, humidade, que coletem informações de forma automática.

A Internet das Coisas pode ser definida como uma infraestrutura de rede global, dinâmica e com capacidades de autoconfiguração, onde as “coisas” são dispositivos eletrônicos que possuem uma identidade e capacidade de interagir e se comunicar entre si através da rede. Estes dispositivos percebem o seu ambiente, “mundo real/físico” através de sensores e são capazes de produzir dados sobre esse ambiente. Os dados alimentam a rede e ativam objetos virtuais que executam processos e serviços. Tais processos podem envolver a intervenção humana ou não. Quando solicitadas as “coisas” são capazes de reagir de maneira autônoma através de atuadores produzindo mudanças no seu ambiente. Espera-se que as “coisas” se tornem participantes ativas nas mais diversas atividades humanas.

3.- Objetivos

O presente trabalho tem como objetivo principal desenvolver a capacidade de construir um dispositivo micro-controlado com base na plataforma Arduino, que possa acessar ou ser acessado através da internet. Para isso se deverá realizar um estudo aprofundado da filosofia da Internet das Coisas. Fazer um levantamento de projetos com base nesta filosofia escolhendo-se um que se adeque aos recursos disponíveis pelo projeto do orientador. Trabalhar todas as etapas de construção de um dispositivo micro-controlado, desde o desenvolvimento do circuito eletrônico, construção da estrutura física de suporte e programação do micro-controlador e do aplicativo via web. Como objetivos secundários, o trabalho visa incentivar aos alunos do curso da Ciência da Computação para trabalhar na área de micro-controladores e desenvolvimento de aplicativos Web. Também consolidar a área de micro-controladores junto ao referido curso.

4.- Metodologia

A metodologia utilizada no projeto gira em torno da plataforma Arduino e a construção de dispositivos eletrônicos aplicando conceitos de eletrônica básica assim como princípios de programação em uma linguagem baseada em C/C++ específica para a plataforma Arduino.

A fim de cumprir com os objetivos propostos, a pesquisa começou com consultas em livros e artigos disponibilizados pelo orientador e seguiu com execução de práticas simples, para posteriormente realizar outras mais elaboradas como a construção de Jogo de Memória Genius, e a construção de um braço robótico controlado por dois módulos joysticks. Além disso, também foi desenvolvido junto a uma outra bolsista, um sistema de automação residencial que consiste em ligar e desligar duas lâmpadas através de uma página web. Todas as práticas são detalhadas na seção de resultados.

Nesta seção apresenta-se uma breve introdução à plataforma Arduino e as principais componentes eletrônicas utilizadas em projetos simples. Componentes mais sofisticadas serão apresentados na próxima seção.

4.1.- Plataforma Arduino

De acordo com [1], A Internet das Coisas representa a próxima evolução da Internet, dando um grande salto na capacidade de coletar, analisar e distribuir dados que nós podemos transformar em informações, conhecimento e, por fim, sabedoria. Nesse contexto, ela tem como objetivo conectar itens à uma rede mundial de computadores.

Neste contexto, o Arduino é uma plataforma que apresenta diversas vantagem para a construção de dispositivos para a Internet das Coisas. De acordo com [2] Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis foi o grupo de pesquisadores que criou o Arduino, em 2005, com o intuito de elaborar um dispositivo de fácil acesso e programação, barato e funcional, seguindo os conceitos do hardware livre, ou seja, um dispositivo que qualquer pessoa poderia personalizar partindo de um único hardware básico. A Figura 1, ilustra uma placa Arduino Uno.

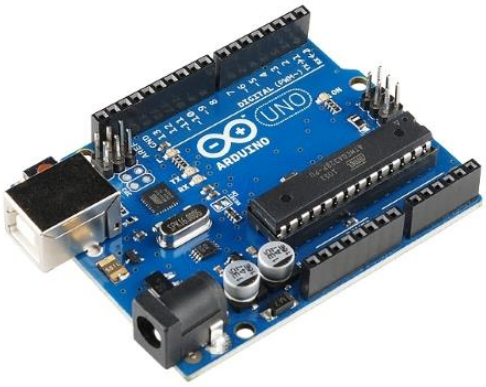

Figura 1 – Placa Arduino Uno – Fonte [1]

A placa é composta por um microcontrolador Atmel de 16 Mhz, com 14 portas digitais e 6 analógicas, pode ser alimentada com tensões entre 7 e 12V, mas somente é capaz de fornecer tensões de até 5V e corrente máxima de 40mA. circuitos de

entrada e saída e pode ser conectada à um computador e programada via IDE (*Integrated Development Environment* ou Ambiente de Desenvolvimento Integrado) utilizando uma linguagem baseada em C/C++, sem a necessidade de equipamentos extras além de um cabo USB.

A maior vantagem do Arduino sobre outras plataformas de desenvolvimento de microcontroladores é a facilidade de sua utilização; e inclusive pessoas que não são da área técnica podem, rapidamente, aprender o básico e criar seus próprios projetos em um intervalo de tempo relativamente curto.

4.2.- Componentes Básicos

Nesta seção são apresentados componentes básicos utilizados na construção de dispositivos desenvolvidos para a plataforma Arduino. O conhecimento das caraterísticas dos componentes permite que eles possam ser utilizados de maneira adequada, evitando possíveis danos a outros dispositivos.

Protoboard

A protoboard é uma placa de ensaio (matriz de contato) que serve como base para a construção de circuitos eletrônicos e teste de projetos. Ela permite que os componentes sejam retirados facilmente e reutilizados depois em uma outra montagem. Seus orifícios são interligados internamente através de contatos, o que elimina a necessidade do uso de solda. A Figura 2, ilustra uma protoboard.

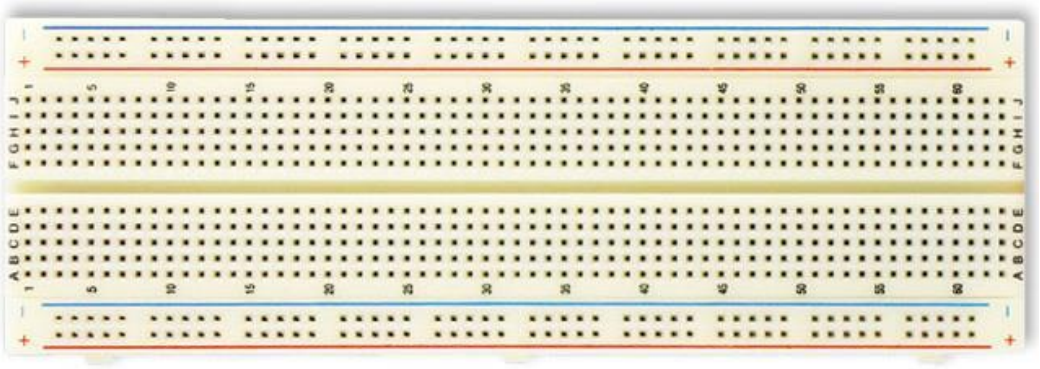

Figura 2 – Protoboard – Fonte[2]

Jumper

Os jumpers são elementos condutores utilizados para conectar dois pontos em um circuito eletrônico. Quando um jumper está conectado à pelo menos dois pinos, ele está fechado. Sem conexão ou apenas com uma conexão, o circuito está aberto.

Figura 3 – Jumper - Fonte [8] - A Autora

Resistores

Um resistor é um componente eletrônico que é usado para limitar a voltagem em um determinado circuito de maneira a garantir níveis adequados de corrente para cada componente sem danificá-los. Cada resistor tem um valor diferente e é escolhido de acordo com a tensão do circuito que queremos diminuir. O grau de resistência deste componente é medido em ohms e codificado utilizando um código de cores. A figura 4 mostra um resistor de 330 ohms.

Figura 4 – Resistor 330 ohms - Fonte [4]

Já a Figura 5, mostra a tabela com o código de cores utilizado para rotular os resistores. Para determinar o valor de um resistor, basta observar as cores das faixas sobre ele e consultar a tabela, segundo as seguintes orientações:

- 1ª Faixa: mostra o primeiro algarismo do valor da resistência.
- 2ª Faixa: mostra o segundo algarismo da resistência.
- 3ª Faixa: mostra quantos zeros devem ser adicionados à resistência.

Cor	1ª Faixa	2ª Faixa	N° de zeros/multiplicador	Tolerância
Preto	0	0	0	
Marrom	1	1	1	
Vermelho	2	2	2	
Laranja	3	3	3	
Amarelo	4	4	4	
Verde	5	5	5	
Azul	6	6	6	
Violeta	7	7	7	
Cinza	8	8	8	
Branco	9	9	9	
Dourado			x0,1	
Prata			x0,01	
Sem cor				± 20%

Figura 5 – Tabela de resistores - Fonte [5]

Por exemplo, o resistor de 330 ohm mostrado na Figura 4, tem as cores laranja, preto e marrão na 1ª, 2ª e 3ª faixas.

Por outro lado, dada a voltagem de uma certa componente, para descobrir qual resistor usar, precisamos utilizar a seguinte fórmula:

$$R = (V - v_C)/ L$$

Onde:

R: resistência (em ohms) do resistor adequado para o componente pesquisado;
V: tensão (em volts) da fonte de energia que alimenta o componente;
V_C : tensão (em volts) suportada pelo componente;
L: Corrente do componente (em amperes).

LEDs

O LED (*Light Emitting Diode* ou Diodo Emissor de Luz) é um condutor de energia elétrica que quando energizado emite luz a olho nu. A figura 6, ilustra LEDs de diferentes cores.

Figura 6 – Diversos LEDs - Fonte[6]

Pushbutton

Um botão é um interruptor momentâneo que, ao ser pressionado, conecta dois pontos do circuito. A figura 7, mostra um exemplo de botão de pressão.

Figura 7 – Botão - Fonte[7]

Todos estes componentes serão utilizado no projeto do Jogo de Memória Genius, onde existem botões associados aos LEDS, de maneira que quando um botão é pressionado o seu LED correspondente é aceso, já que o circuito é fechado e envia-se um fluxo de corrente através dele. Já quando o interruptor está livre, o circuito encontra-se aberto o que faz com que os LEDs permaneçam apagados.

5.- Resultados e Discussão

Como resultado desta pesquisa além dos estudos referentes à plataforma Arduino e Internet das Coisas foram desenvolvidos três dispositivos micro-controlados, todos eles escolhidos em consenso com o orientador, de acordo com a disponibilidade de componentes eletrônicos e viabilidade financeira. O primeiro dispositivo foi proposto como parte final do treinamento sobre a plataforma Arduino e consistiu em um Jogo de Memória baseado em uma sequência de luzes, semelhante ao jogo conhecido comercialmente como Genius. O segundo dispositivo que foi o objeto central da pesquisa, consistiu em desenvolver um braço robô que seria controlado através de um aplicativo e/ou página web, utilizando algum tipo de conexão com a internet. O braço robô foi construído em uma versão controlada através de um par de módulos tipo Joystick. No entanto, devido ao pouco tempo hábil para implementar o controle do braço robô através da internet, trabalhou-se em um terceiro projeto, criado por outra aluna de IC, que consistia em um dispositivo micro-controlado para ligar/desligar lâmpadas de iluminação residencial. Esta última parte foi realizada em colaboração com a aluna Gabriela Peixoto e consistiu no controle das lâmpadas através de um web site. Os experimentos realizados neste projeto secundário, mostraram a viabilidade do controle de lâmpadas através da internet, mas evidenciou alguns problemas como falta de memória no micro-controlador e a necessidade de adquirir maior conhecimentos sobre outras alternativas de conexão com a internet, assim como as componentes eletrônicas para essa finalidade.

Nesta seção serão apresentados os detalhes sobre cada um dos três projetos, contemplando: i) componentes utilizadas, ii) diagrama do dispositivo e iii) Código em linguagem C para o micro-controlador e iv) Protótipo do dispositivo.

5.1.- Projeto do Jogo de Memória Genius

O Genius era um brinquedo muito popular na década de 90 e consistia em estimular a memorização de cores e sons. O jogo desenvolvido com Arduino contempla apenas o aspecto das cores, sendo composto por quatro *leds* que piscam gerando uma sequência aleatória de cores. O jogador precisa memorizar e repetir a sequência usando para isso um conjunto de botões associados a cada *led.* O processo é repetido incrementando-se o nível de dificuldade (número de leds na sequência a ser memorizada) e termina quando o jogador errar a sequência ou conseguir reproduzir corretamente a maior sequência programada no jogo. Este projeto permitiu desenvolver a habilidade prática na montagem de circuitos assim como adquirir conhecimentos sobre a programação em linguagem C para Arduino.

Componentes utilizados

Foram utilizados os seguintes componentes, descritos na seção anterior:

- 1 Arduino Uno com cabo USB;
- 1 Protoboard 800 pontos e Jumpers de conexão;
- 4 Resistores de 330 ohm;
- 4 *LEDs* coloridos;
- Botões *PushButton*.

Diagrama de conexão

A figura 8 ilustra o diagrama de conexão entre a placa Arduino e as componentes que fazem parte do dispositivo do Jogo de memória. Basicamente, temos 4 pinos digitais da placa Arduino (parte superior direita) sendo usados como OUTPUT(saída) para alimentar cada um dos 4 leds coloridos. Os 4 resistores são usados para garantir a voltagem certa em cada led, já que a voltagem de saída padrão (5V) supera a voltagem suportada pelo leds. Outros 4 pinos digitais (parte superior esquerda) são usados como INPUT(entrada) para registrar a pulsação de cada um dos 4 botões. Na parte inferior da placa, encontram-se as linhas de alimentação de energia e o terra (vermelha e preta).

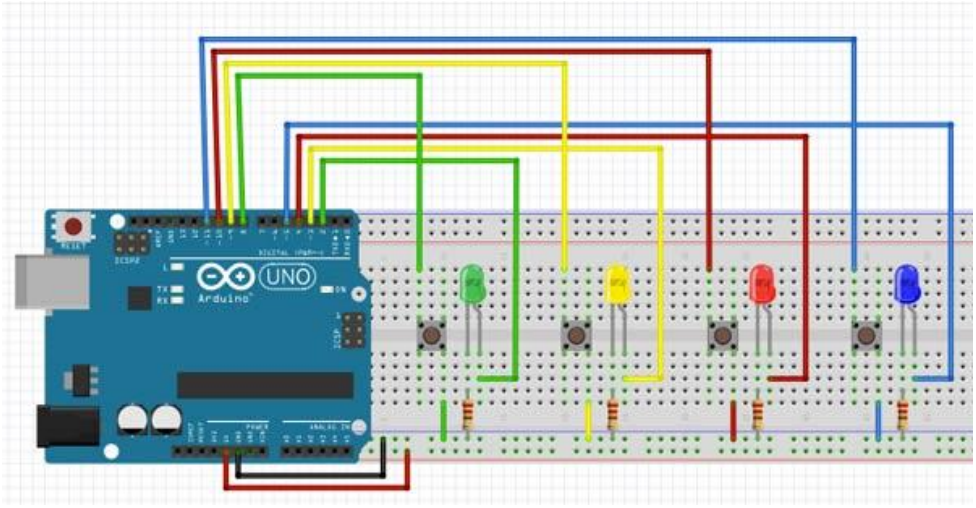

Figura 8 – Diagrama Jogo de Memória Genius - Fonte: A Autora

Código de controle para Arduino

O código de controle do dispositivo é mostrado a seguir, com alguns comentários para facilitar a sua compreensão. Na parte de definições, associam-se os pinos de OUTPUT aos leds e os pinos de INPUT aos botões. Define-se o tamanho máximo da sequência como 10. Define-se um conjunto de quatro estados possíveis para o jogo. Define-se um vetor para armazenar a sequência gerada. Define-se o número de rodada assim como o número de cores respondidas.

```
#define GREEN 2
#define YELLOW 3
#define RED 4
#define BLUE 5
#define GREEN_BUTTON 11
#define YELLOW_BUTTON 10
#define RED_BUTTON 9
#define BLUE_BUTTON 8
#define INDEFINIDO -1
#define SIZE 10
enum Estados {PRONTO_PROX_RODADA,USUARIO_RESPONDENDO,
FIM_DE_JOGO_SUCESSO,FIM_DE_JOGO_FALHA};
int sequenciaLuzes[SIZE];
int rodada = 0;
int leds_respondidos = 0;
```

Na parte de inicialização (Setup), inicializam-se: o monitor serial, os pinos de OUTPUT e INPUT e inicia-se o jogo, que consiste em gerar e armazenar uma sequência aleatória de 10 números entre 2 e 5.

```
void setup(){
  Serial.begin(9600);
  iniciaPortas(); iniciaJogo();
}
void iniciaPortas(){
  pinMode(GREEN, OUTPUT); pinMode(YELLOW, OUTPUT);
  pinMode(RED, OUTPUT);      pinMode(BLUE, OUTPUT);
  pinMode(GREEN_BUTTON, INPUT_PULLUP);
   pinMode(YELLOW_BUTTON, INPUT_PULLUP);
  pinMode(RED_BUTTON, INPUT_PULLUP);
  pinMode(BLUE_BUTTON, INPUT_PULLUP);
}
void iniciaJogo(){
  int jogo = analogRead(0);   randomSeed(jogo);
  for(int i=0; i<SIZE; i++){
     sequenciaLuzes[i] = sorteiaCor();
  }
}
int sorteiaCor(){
  return random(GREEN, BLUE + 1); //2-5
}
```

Na parte de repetição (Loop), verifica-se a ocorrência de quatro casos possíveis para o usuário, para isso usa-se a função **estadoAtual()**. Cada caso requer um tratamento específico por uma função. Temos os seguintes casos: i) completou a rodada atual, **preparaNovaRodada()**, ii) ainda está respondendo a rodada atual, **processaRespostaUsuario();** iii) concluiu a sequência completa sem errar, **jogoFinalizadoSucesso()** e iv) errou antes de concluir a sequência completa **jogoFinalizadoFalha()**. Cada uma destas funções será apresentada a seguir.

```
void loop(){
    switch (estadoAtual()){
    case PRONTO_PROX_RODADA:
        Serial.println("Pronto para proxima rodada");  preparaNovaRodada();
        break;
    case USUARIO_RESPONDENDO:
        Serial.println("usuário respondendo");       processaRespostaUsuario();
        break;
    case FIM_DE_JOGO_SUCESSO:
        Serial.println("Jogo finalizado com sucesso"); jogoFinalizadoSucesso();
        break;
    case FIM_DE_JOGO_FALHA:
        Serial.println("Jogo finalizado com falha");      jogoFinalizadoFalha();
        break;
   }
   delay(500);
}
```

Na função estadoAtual(), se a rodada atual é menor ou igual que o número de cores na sequência, verifica-se se todas as cores de essa rodada já foram respondidas

corretamente. Se isso for verdade prepara-se a próxima rodada. Caso contrário, processa-se a próxima resposta do usuário. Já se a rodada atual for maior que o número de cores na sequência em uma unidade, considera-se que todas as cores foram acertadas e o jogo termina. Enquanto que, se a rodada atual for maior em duas unidades que o número de cores significa que foi identificado um erro na cor da última jogada.

```
int estadoAtual(){
  if (rodada <= SIZE){
     if (leds_respondidos == rodada) return PRONTO_PROX_RODADA;
     else return USUARIO_RESPONDENDO;
  }
 else if (rodada == SIZE + 1) return FIM_DE_JOGO_SUCESSO;
       else return FIM_DE_JOGO_FALHA;
}
```

A função **preparaNovaRodada()** incrementa em um o número de cores a ser reproduzido e zera o contador respostas corretas do jogador. A função **processaRespostaUsuario()** verifica o estado dos botões, caso algum tenha sido pressionado, acende ele por um segundo e verifica se a cor associada é igual a da sequência nessa rodada. Em caso de acerto incrementa o contador de cores acertadas, caso contraria, define a condição de erro: rodada = SIZE + 2, que fara terminar o programa.

```
void preparaNovaRodada(){
  rodada++;
  leds_respondidos = 0;
  if (rodada <= SIZE) tocaLedsRodada();
}

void processaRespostaUsuario(){
   int resposta = checaRespostaJogador();
   if (resposta == INDEFINIDO)  return;
   if (resposta == sequenciaLuzes[leds_respondidos]){
      leds_respondidos++;
   }
   else{
      Serial.println("resposta errada");  rodada = SIZE + 2;
   }
 }

int checaRespostaJogador(){
    if (digitalRead(GREEN_BUTTON) == LOW)    return piscaLed(GREEN);
    if (digitalRead(YELLOW_BUTTON) == LOW)  return piscaLed(YELLOW);
    if (digitalRead(RED_BUTTON) == LOW)         return piscaLed(RED);
    if (digitalRead(BLUE_BUTTON) == LOW)       return piscaLed(BLUE);
    return INDEFINIDO;
```

As funções **tocaLedsRodada()** e **piscaLed()** acendem leds durante um segundo, um de cada vez. Enquanto, a primeira função acendem uma sequência de *leds*, a segunda acende um *led* individual.

```
int void tocaLedsRodada(){
   for (int i=0; i<rodada; i++) piscaLed(sequenciaLuzes[i]);
}

int piscaLed(int portaLed){
  digitalWrite(portaLed,HIGH);   delay(1000);
  digitalWrite(portaLed,LOW);    delay(500);
  return portaLed;
}
```

Finalmente, as funções **jogoFinalizadoSucesso()** e **jogoFinalizadoFalha()** emitem sinais luminosos para indicar o fim do jogo com sucesso e com erro. No primeiro caso, as quatro cores são acendidas em sequência. No segundo, são acendidas ao mesmo tempo.

```
void jogoFinalizadoSucesso(){
  piscaLed(GREEN);   piscaLed(YELLOW);
  piscaLed(RED);       piscaLed(BLUE);
  delay(500);
}

void jogoFinalizadoFalha() {
   digitalWrite(GREEN,HIGH);    digitalWrite(YELLOW,HIGH);
   digitalWrite(RED,HIGH);       digitalWrite(BLUE,HIGH);
   delay(1000);
   digitalWrite(GREEN, LOW);    digitalWrite(YELLOW, LOW);
   digitalWrite(RED, LOW);       digitalWrite(BLUE, LOW);
   delay(500);
 }
```

Protótipo do Dispositivo

A Figura 9 ilustra o protótipo do Jogo de Memória Genius finalizado.

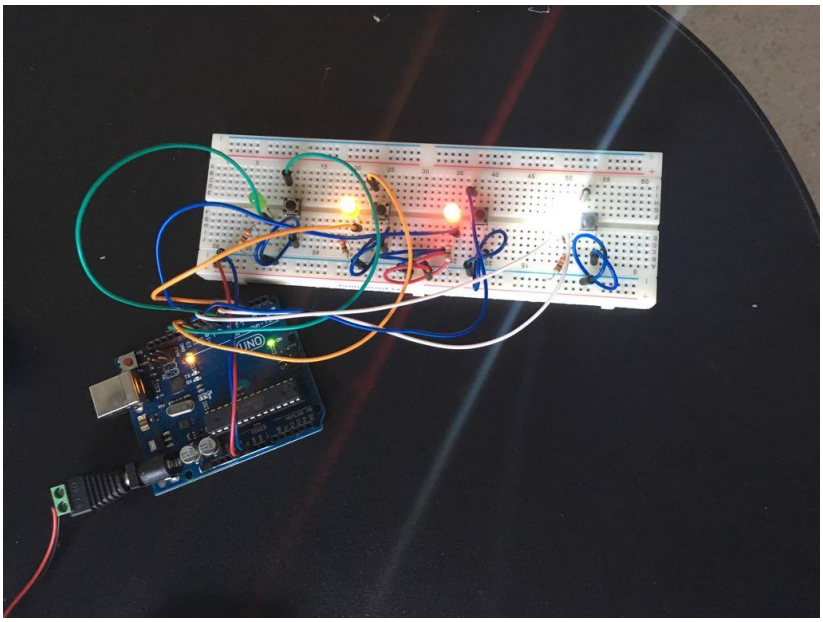

Figura 9 – Protótipo do Jogo de Memória Genius - Fonte: A Autora

5.2.- Projeto do Braço Robótico

O projeto do braço robótico foi escolhido em consenso com o orientador como projeto central da pesquisa, visando construir um pequeno braço articulado para simular aplicações industriais. Este projeto, seria inicialmente controlado através de módulos *Joystick* e posteriormente seria controlado através da internet. A estrutura escolhida para o braço foi em material mdf. Sendo que quatro servomotores forneceriam a movimentação do braço, incluindo uma pequena garra para pegar objetos leves. A implementação do braço robô demandou muito mais tempo que o projeto anterior, devido as etapas de montagem e escolha de componentes adequadas. No entanto, a programação do código ficou simplificada devido ao uso de uma biblioteca específica.

Devido ao pouco tempo disponível após o primeiro protótipo ficar pronto, não foi possível implementar o controle através da internet.

Componentes utilizados

A descrição das componentes é importante para compreender as suas capacidades e limitações. A resistência do material usando na estrutura do braço assim como, a força dos micro servo motores devem ser consideradas. Além disso, é importante conhecer as especificação elétricas das componentes: tensão suportada e corrente requerida por cada componente. Foram utilizados os seguintes componentes, que serão descritos a seguir:

- 1 Braço Robótico em mdf para Arduino
- 4 MicroServo motores Tower Pro 9g
- 1 Placa Arduino Rev3 Genérica e 1 conjunto de *Jumpers*
- 1 Placa Sensor Shield ver 5.0 para Arduino
- 2 Módulos *Joysticks*
- 1 Fonte de Alimentação 5V e 1 Bateria 9V

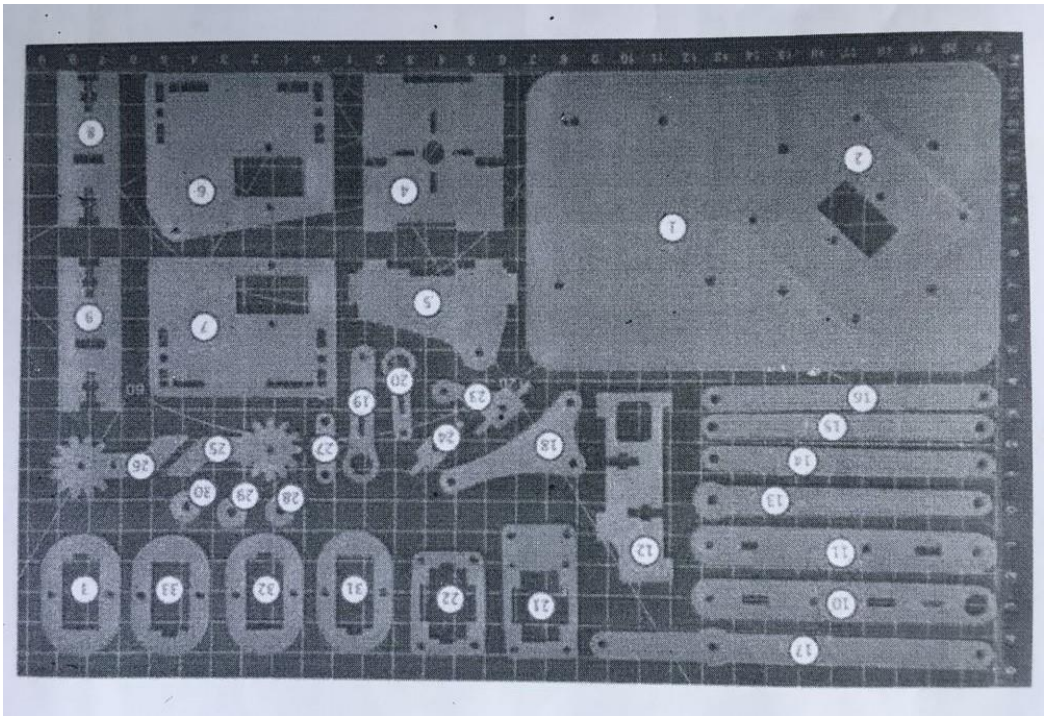

Figura 10 - Peças do Braço Robótico - Fonte: Manual de Montagem

i) Braço Robótico em mdf para Arduino

Escolheu-se uma estrutura pré-fabricada em material mdf devido à sua praticidade e maior resistência. O kit adquirido é formado por 35 peças estruturais, conforme ilustrado na figura [10].

Além disso, foram utilizados um conjunto de 48 parafusos de diâmetro M3 e comprimentos variáveis entre 6 e 20mm. Vale observar que a tarefa de montagem da estrutura foi bastante demorada e demandou a assistência de duas pessoas.

ii) Micro Servo Motor Tower Pro 9g

Os micro servo motores podem ser definidos como motores de precisão com rotação limitada em ângulos fixos. Eles são ideais para controlar a movimentação de peças articuladas. No projeto do braço robótico, são usados 4 servomotores para garantir os seguintes movimentos: i) garra do braço; ii) giro do braço; iii) movimento para frente/atrás e iv) movimento para cima/abaixo. O modelo adotado, possui ângulo de rotação de 180 graus e um cabo de três pinos destinado a alimentação e controle. O eixo do servo pode ser acoplado a três tipos de hélices de formatos diferentes (meia hélice, hélice inteira, hélice forma de cruz). A figura 11 ilustra o Micro Servo motor utilizado.

Figura 11.- Micro Servo Motor TowerPro - Fonte [9]

As especificações do servo motor com base em [1] são as seguintes:

- Voltagem de operação: 3,0 a 7,2V
- Ângulo de rotação: 180 graus
- Torque: 1,2 kg.cm (4,8V) e 1,6 kg.cm (6,0V)
- Tipo de Engrenagem: Nylon
- Tamanho do cabo: 245 mm
- Dimensões: 32x30x12 mm
- Peso 9g

Detalhes sobre a pinagem do servo motor são dadas em [2]. O fio laranja é um fio de controle que deve receber um sinal PWM (*Pulse Wide Modulated*), pulso de amplitude modulada. Já o fio vermelho corresponde à alimentação, +VCC, enquanto o fio marrão corresponde ao terra, GND. A Instalação dos quatro servo motores exigiu a desmontagem parcial da estrutura do braço revelando a necessidade de um melhor planejamento das atividades para evitar trabalho redundante. Junto a instalação dos servo motores foi realizada a instalação da placa Arduino na base do braço robô.

iii) Placa Sensor Shield ver 5.0

Devido a que o número de motores que seriam conectados diretamente a placa Arduino, representava um risco potencial para a mesma, ao exigir uma quantidade de corrente que poderia ultrapassar a sua capacidade, preferimos fazer a alimentação dos motores e do Arduino em separado (alimentação externa). Uma boa alternativa para isso, é a placa Sensor Shield v 5.0 que é uma opção de baixo custo, caracterizando-se por ser um expansor de portas digitais e analógicas dispostas em barramento do tipo alimentação-sinal de 3 pinos. Um detalhe importante para realizar a alimentação externa é remover o *jumper* de configuração acima do conector externo. Além disso, a placa é capaz de operar na faixa de 5V o que garante compatibilidade com o Arduino. A figura 12, mostra o detalhe das conexões da placa sensor shield v5.0.

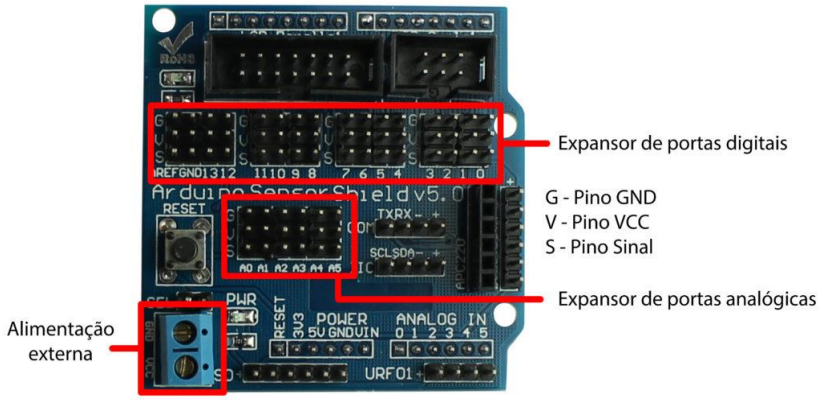

Figura 12.- Placa Sensor Shield ver 5.0 - Fonte [10]

Assim, para evitar qualquer risco desnecessário utilizou-se ainda uma fonte AC/DC de 5V e 1A.

Diagrama de conexão

A figura 13 ilustra o diagrama de conexão entre a Sensor Shield v5.0 e as componentes que fazem parte do braço robô.

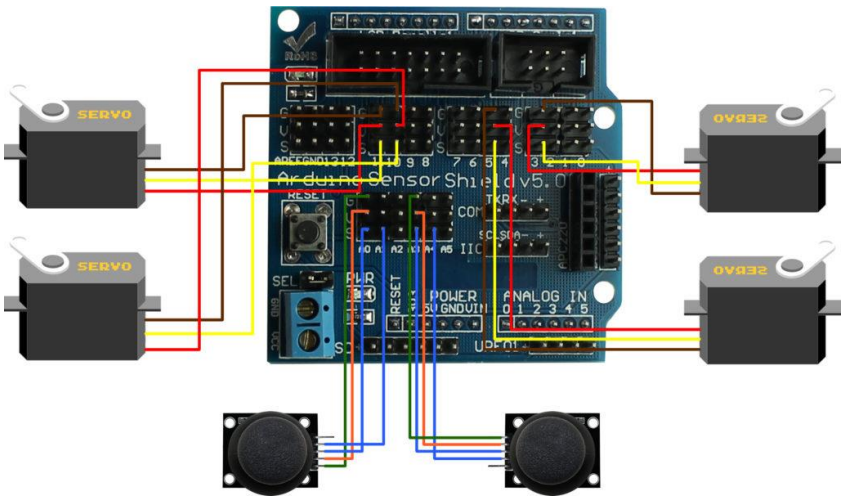

Figura 13 - Diagrama de conexão - Fonte [10]

Os quatro servos foram conectados nas portas digitais 3, 5 10 e 11, segundo a pinagem mostrada na Figura 12. Já os dois *Joysticks* foram conectados nas portas analógicas, ocupando 2 pinos de sinal cada um. Vale lembrar que a placa Sensor Shield encontra-se encaixada sobre a Arduino Uno, mas recebe e fornece alimentação externa a todos os componentes.

Código de controle para Arduino

O código para o controle do braço basicamente, recebe informações a partir de ambos os joysticks e repassa os comandos de movimento para cada um dos quatro servomotores, produzindo o movimento do braço. O código foi desenvolvido com base em código disponível no site da UsinaInfo. Foi utilizada a mesma biblioteca sugerida pelo site, no entanto foi necessário corrigir dois erros presentes nesse código que impediam a correta movimentação do braço na direção para frente/atrás.

Na parte de definições, criam-se quatro objetos VarSpeedServo, definidos na biblioteca VarSpeedServo.h para controlar a movimentação dos quatro servomotores. Define-se os pinos associados aos dois eixos de cada *Joystick* e definem-se também variáveis para armazenar o valor lido pela movimentação de ambos eixos dos *Joysticks.*

```
#include <VarSpeedServo.h> // Inclui a Biblioteca VarSpeedServo.h
VarSpeedServo servo_sobe; //Cria objeto para controlar o servo sobe
VarSpeedServo servo_frente; //Cria objeto para controlar o servo frente
VarSpeedServo servo_garra; //Cria objeto para controlar o servo garra
VarSpeedServo servo_corpo; //Cria objeto para controlar o servo corpo

int pino_x = A0; //Inicializa o pino analógico ao eixo X do joystick1
int pino_y = A1; //Inicializa o pino analógico ao eixo Y do joystick1
int pino_z = A3; //Inicializa o pino analógico ao eixo Z do joystick2
int pino_w = A4; //Inicializa o pino analógico ao eixo W do joystick2
int val_x;       //Armazena o valor lido pelo eixo X do joystick 1
int val_y;       //Armazena o valor lido pelo eixo Y do joystick 1
int val_z;       //Armazena o valor lido pelo eixo Z do joystick 2
int val_w;       //Armazena o valor lido pelo eixo W do joystick 2
```

Na parte de Inicialização (*Setup*), associa-se cada servo conectado a um pino digital a um objeto criado na parte de definições e especificação o ângulo de rotação máximo de 180 graus para todos eles.

```
void setup() {
  servo_frente.attach(5, 1, 180);  //Define que o servo está conectado a porta 5 do Arduino
  servo_sobe.attach(6, 1, 180);    //Define que o servo está conectado a porta 6 do Arduino
  servo_garra.attach(10, 1, 180); //Define que o servo está conectado a porta 10 do Arduino
  servo_corpo.attach(11, 1, 180); //Define que o servo está conectado a porta 11 do Arduino
}
```

Na parte de repetição (*Loop*), realiza-se a leitura dos valores registrados pelos quatro eixos de ambos *joysticks*, e se faz a conversão do valor na escala de 1 a 180 graus. Finalmente, realiza-se a movimentação do braço mediante o método **slowmove()**.

```
void loop() {
  val_x = analogRead(pino_x); //Recebe o valor lido pelo eixo X do joystick
  val_x = map(val_x, 0, 1023, 180, 1); //Converte o valor lido para um valor em graus (1 a 180º)
  servo_sobe.slowmove(val_x, 60); //Movimenta o servo até a posição definida pelo eixo X
  val_y = analogRead(pino_y); //Recebe o valor lido pelo eixo Y do joystick
  val_y = map(val_y, 0, 1023, 1, 180); //Converte o valor lido para um valor em graus (1 a 180º)
  servo_frente.slowmove(val_y, 60); //Movimenta o servo até a posição definida pelo eixo Y

  val_z = analogRead(pino_z); //Recebe o valor lido pelo eixo Z do joystick
  val_z = map(val_z, 0, 1023, 180, 1); //Converte o valor lido para um valor em graus (1 a 180º)
  servo_garra.slowmove(val_z, 60); //Movimenta o servo até a posição definida pelo eixo Z
  val_w = analogRead(pino_w); //Recebe o valor lido pelo eixo W do joystick
  val_w = map(val_w, 0, 1023, 1, 180); //Converte o valor lido para um valor em graus (1 a 180º)
  servo_corpo.slowmove(val_w, 60); //Movimenta o servo até a posição definida pelo eixo W
  delay(30);
}
```

Protótipo do Dispositivo

A Figura 14 ilustra o protótipo do braço robótico construído e segurando um pequeno objeto pela garra. Observam-se também os detalhe do suporte utilizado para fixar os dois controles *Joystick* e a alimentação da placa Arduino através de uma pilha 9V, enquanto a placa *Sensor Shield* é alimentada usando uma fonte de alimentação AC/DC de 5V.

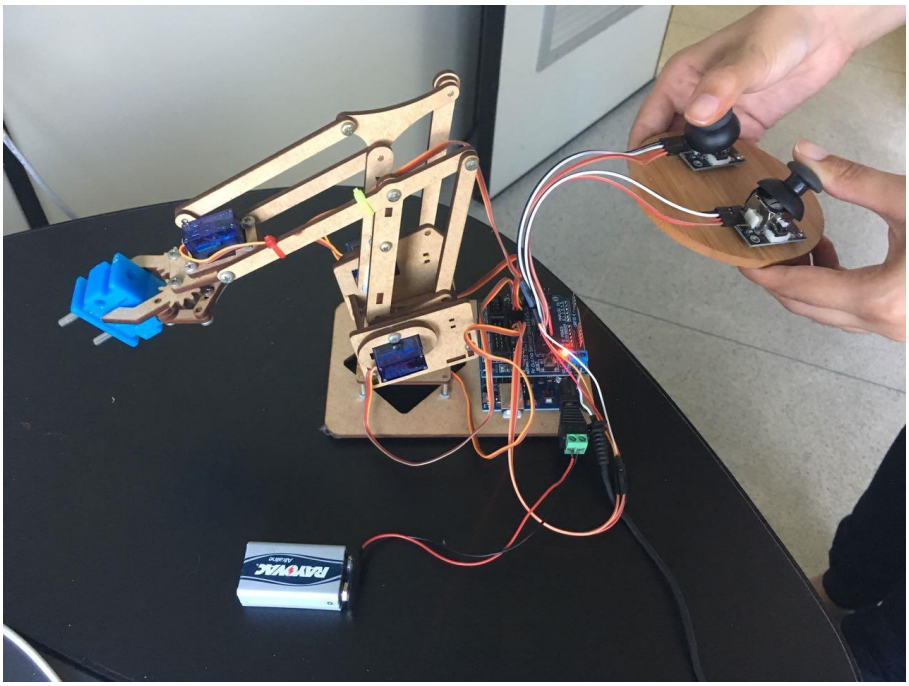

Figura 14 – Braço Robô - Fonte: A Autora

5.3.- Automação Residencial: Controle de lâmpadas pela internet

A participação neste terceiro projeto surge como um projeto secundário que teve como objetivo ganhar experiencia na conexão de dispositivos através da placa *Ethernet Shield*. Trabalhou-se em colaboração com a aluna Gabriela Peixoto tentando estender um dispositivo de controle de lâmpadas de iluminação doméstica construído pela referida aluna pela incorporação do um controle através da internet. O projeto utiliza duas lâmpadas que são controladas através de um módulo relê duplo. Nele é incorporada uma placa Ethernet Shield para o acesso à internet. O trabalho centrou-se em testes de conectividade e na criação de uma pagina *web* com dois botões para acender as lâmpadas. Os testes não foram totalmente satisfatórios, embora fosse comprovada a conexão a internet, a página proposta apresentou problemas por falta de memória.

Componentes Utilizados

O projeto de controle de lâmpadas compreendeu os seguintes componentes:

- 1 Arduino Uno com cabo USB
- 1 Protoboard de 800 pontos e Jumpers variados
- 1 Módulo Relé Arduino (2 relês)
- 2 Lâmpadas fluorescentes, 2 Bocais, 1 Plugue Tomada, Fio elétrico (3mts)
- 1 placa Ethernet Shield W5100

i) *Ethernet Shield* W5100

A placa Ethernet Shield W5100 permite que a placa Arduino se conecte à internet. É baseado no chip da WIZnet, W5100 que fornece acesso à rede (IP) nos protocolos TCP ou UDP e é facilmente utilizado usando a biblioteca Ethernet Library e SD Library. Nessa versão, o shield possui um slot para cartão micro-SD, que pode ser usado para guardar e armazenar arquivos de um servidor na rede.

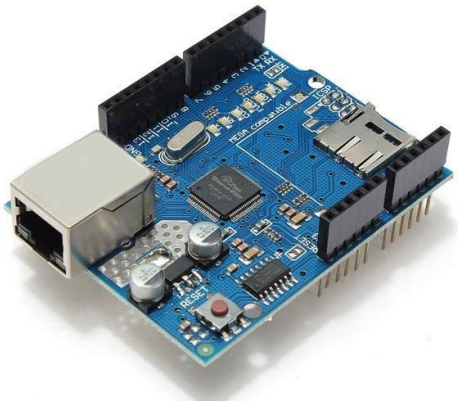

Figura 15 – Ethernet Shield W5100 – Fonte [11]

Diagrama de conexão

A Figura 16, mostra o diagrama de conexão das lâmpadas ao módulo relê e a *Ethernet Shield*. O módulo relé funciona basicamente como um interruptor que pode ser acionado pelo Arduino. Para isso, utilizam-se as portas digitais 3 e 4, equanto a alimentação do módulo relê é feita pelo pino 5V.

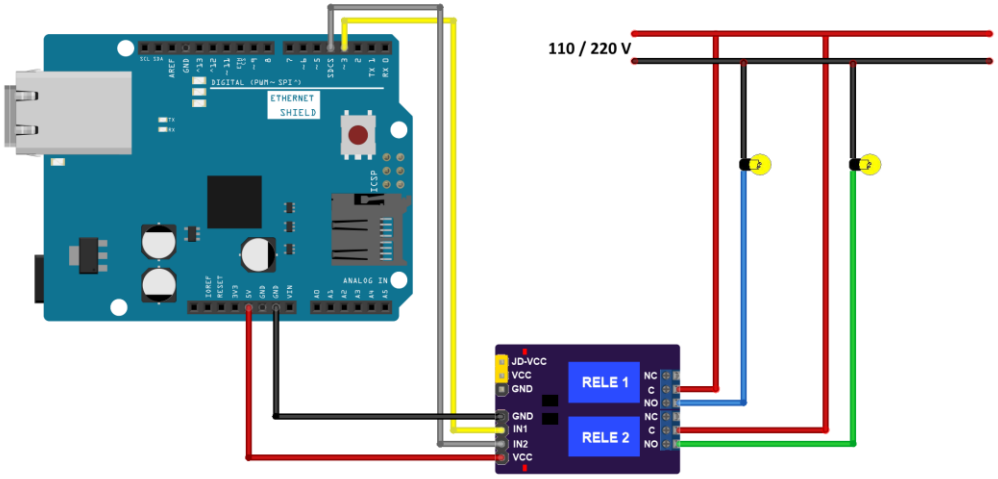

Figura 16 – Diagrama de conexão com Ethernet Shield – Fonte[10]

Código de Controle

A primeira dificuldade encontrada foi descobrir o número de IP usado pelo *Ethernet Shield* na nossa rede. Foi utilizado um código auxiliar para descobrir o IP usado pela shield, e que foi o utilizado para acessar a página de controle carregada no Arduino.

Após descobrir o IP da placa, utilizamos como referência um código do site FilipeFlop, que carrega uma página web feita em html e css na placa arduíno, e que utiliza que funções em *javascript* para controlar a lâmpada. Os testes realizados com este código permitiram acesso via browser, a uma página web semelhante à mostrada na Figura 17. A página permite acender e apagar as lâmpadas mediante um par de botões LIGA e DESLIGA.

Figura 17 – Página Web UENF – Fonte: A Autora

No entanto, após desenvolver um código próprio, para uma página *web* personalizada, incluindo os logos, da UENF, nomes dos autores e certa estilização,

exatamente como mostrado na Figura 17, o Arduino acusou instabilidade durante a execução devido à falta de memória e impossibilitando a acesso a página web criada. Foram feitos novos testes, utilizado um cartão SD, porém o problema persistiu e o acesso só passou a ser possível sem utilizar o css na página. As funções javascript continuaram funcionando, porém toda a estilização teve que ser removida. Assim, embora foi provada a viabilidade da aplicação, na prática a página perdeu seu atrativo visual e de fato a sua utilidade. O código produzido para página foi o seguinte:

Código em C, com a nova página em html e css

```
#include <SPI.h>
#include <Ethernet.h>
String readString;

int pino_rele1 = 3;
int pino_rele2 = 4;
boolean ligado = true;
boolean ligado_2 = true;

//Informacoes de endereco IP, gateway, mascara de rede
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
//byte ip[] = { 192, 168, 1, 103};
//byte gateway[] = { 192, 168, 1, 1 };
//byte subnet[] = { 255, 255, 255, 0 };

EthernetServer server(80);

void setup()
{
  Serial.begin(9600);
  pinMode(pino_rele1, OUTPUT);
  pinMode(pino_rele2, OUTPUT);

  //Inicializa Ethernet Shield
  //Ethernet.begin(mac, ip, gateway, subnet);
  Ethernet.begin(mac);
  Serial.println("Ethernet Configured!");
  server.begin();
  Serial.print("Server Started.\nLocal IP: ");
  Serial.println(Ethernet.localIP());

  Serial.println("FILIPEFLOP - Automacao Residencial");

  //Desliga os dois reles
  digitalWrite(pino_rele1, HIGH);
  digitalWrite(pino_rele2, HIGH);
}

void loop()
{
  EthernetClient client = server.available();
  if (client) {
    while (client.connected())
    {
      if (client.available())
      {
        char c = client.read();
        if (readString.length() < 100) {
```

```
        readString += c;
      }
      if (c == 'n')
      {
        //Controle do rele1
        Serial.println(readString);
        //Liga o Rele 1
        if (readString.indexOf("?ligar") > 0)
        {
          digitalWrite(pino_rele1, LOW);
          Serial.println("Rele 1 Ligado");
          ligado = false;
        }
        else
        {
          //Desliga o Rele 1
          if (readString.indexOf("?desligar") > 0)
          {
            digitalWrite(pino_rele1, HIGH);
            Serial.println("Rele 1 Desligado");
            Serial.println(readString.indexOf("?desligar"));
            ligado = true;
          }
        }

        //Controle do rele2
        Serial.println(readString);
        //Liga o Rele 2
        if (readString.indexOf("?2_ligar") > 0)
        {
          digitalWrite(pino_rele2, LOW);
          Serial.println("Rele 2 Ligado");
          Serial.println(readString.indexOf("?2_ligar"));
          ligado_2 = false;
        }
        else
        {
          //Desliga o Rele 2
          if (readString.indexOf("?2_desligar") > 0)
          {
            digitalWrite(pino_rele2, HIGH);
            Serial.println("Rele 2 Desligado");
            ligado_2 = true;
          }
        }
        readString = "";

        client.println("HTTP/1.1 200 OK");
        client.println("Content-Type: text/html");
        client.println();
        client.println("<html>");
        client.println("<head>");
        client.println("<title>Projeto Ethernet - UENF</title>");
        //client.println("<meta name='viewport' content='width=720, initial-scale=0.5' />");
        client.println("<link rel='stylesheet' type='text/css' href='style.css' />");
        client.println("<script                                                    type='text/javascript'
src='https://img.filipeflop.com/files/download/automacao/automacao_residencial.js'></script>");
        client.println("</head>");
        client.println("<body>");
        client.println("<header>");
```

```
        client.println("<img id='uenf' src='https://i.imgur.com/TXAHHSw.png'>");
        client.println("<img id='arduino' src='https://i.imgur.com/pyoPHnm.png'>");
        client.println("</header>");*/
        client.println("<section>");
        client.println("<h1>Automação Residencial com Arduino</h1>");
        client.println("<div id='funcionalidade'>");
        client.println("<div id='div1'><h2>Rele 1</h2></div>");
        client.println("<div id='botao'></div>");
        client.print("<div id='rele'></div><div id='estado' style='visibility: hidden;'>");
        client.print(ligado);
        client.println("</div>");
        client.println("<div id='div2'><h2>Rele 2</h2></div>");
        client.println("<div id='botao_2'></div>");
        client.print("<div id='rele_2'></div><div id='estado_2' style='visibility: hidden;'>");
        client.print(ligado_2);
        client.println("</div>");
        client.println("<script>AlteraRele1()</script>");
        client.println("<script>AlteraRele2()</script>");
        client.println("</div>");
        client.println("</section>");
        client.println("<footer>");
        client.println("<div id='alunas'>");
        client.println("<h2>Projeto de Iniciação Científica desenvolvido pelas <br>alunas Isabela Correia e
Gabriela Peixoto </h2>");
        client.println("<h2>Orientador: Fermín Alfredo Tang Montané</h2>");
        client.println("</div>");
        client.println("<div id='curso'>");
        client.println("<h2>Ciência da Computação</h2>");
        client.println("<h2>LCMAT - CCT</h2>  ");
        client.println("</div>");
        client.println("</footer>");*/
        client.println("</body>");
        client.println("</head>");

        delay(1);
        client.stop();
      }
    }
  }
 }
}
```

6.- Conclusões

Ao começar a pesquisa, o orientador disponibilizou livros sobre Arduino e Internet das Coisas, além de compartilhar com a orientada artigos e sites sobre esses temas. Esta primeira etapa visou reunir material de referência para estudo para assim poder dar os primeiros passos com a plataforma Arduino. Terminada esta etapa, iniciou-se o estudo da plataforma Arduino através de leituras e práticas, com a execução de pequenos projetos disponíveis nos livros de texto e nos sites disponibilizados. Para concluir a etapa de treinamento, escolheu-se como prática final a implementação do um Jogo de Memória semelhante ao Jogo Genius utilizando uma sequência de *leds* coloridos. O protótipo do jogo foi construído de maneira satisfatória.

Na segunda etapa do plano de trabalho, o projeto escolhido para ser desenvolvido foi um braço robótico que poderia simular operações na indústria, inicialmente ele foi controlado mediante um par de *joysticks*, mas a expectativa era que esse controle passasse a ser feito mediante um aplicativo através da internet.

A etapas do plano de trabalho referentes à construção do dispositivo, foram em sua grande maioria cumpridas de maneira satisfatória. O projeto da estrutura física embora baseado em uma estrutura pré-fabricada, demandou várias sessões de montagem, devido ao grande número de peças envolvidas. Vale observar que após a montagem inicial, foram necessárias desmontagens parciais para a instalação das componentes eletrônicas e fiação. Esta experiência mostrou a importância de um planejamento cuidadoso e detalhado na montagem do dispositivo físico para evitar a repetição de tarefas já realizadas. O projeto eletrônico do braço foi planejado usando diagramas do software *Fritzing*. Já o programa de controle foi codificado na linguagem C, para Arduino.

Entre os principais desafios encontrados na construção do braço robótico devem-se destacar dois: i) o problema de alimentação de energia para os quatro microservo motores. Isso foi resolvido utilizando uma placa *Sensor Shield* e uma fonte de alimentação de 5V. ii) o ajuste da movimentação do braço para atender o resultado esperado, demandou vários testes de precisão.

Finalmente, não foi possível implementar o controle do braço através de um aplicativo mediante a internet O tempo para fazer essa mudança não foi o suficiente, pois era necessário um estudo mais a fundo sobre redes, servidores web e algumas linguagens de programação. Optou-se então, por trabalhar a questão da conectividade em um contexto mais simples, o projeto de controle de lâmpadas através da web, que foi de fato implementado. Visto isso, podemos concluir que foi construído um protótipo eletrônico com base em Arduino e também desenvolvido um WebApp para controle desse protótipo.

7.- Referências

[1] CISCO. A Internet das Coisas - Como a próxima evolução da Internet está mudando tudo

[2] MCROBERTS, M. Arduíno Básico. Primeira edição. Novatec

[3] http://www.eletronicadidatica.com.br/protoboard.html

[4] https://www.oficinadanet.com.br/post/14953-aula-4-corrente-tensao-resistor-e-diodo-emissor-de-luz

[5] https://www.mundodaeletrica.com.br/codigo-de-cores-de-resistores/

[6] https://www.mundodaeletrica.com.br/o-que-e-um-led/

[7] https://playground.arduino.cc/Portugues/LearningButton

[8] https://pt.wikipedia.org/wiki/Jumper

[9] www.alura.com.br

[10] https://www.usinainfo.com.br/mini-bracos-roboticos/braco-robotico-em-mdf-para-arduino-manual-de-montagem-3597.html

[11] https://www.filipeflop.com/blog/automacao-residencial-com-arduino-acenda-lampadas-pela-internet/

8.- Perspectivas de continuidade ou desdobramento do trabalho

Na continuação da pesquisa, conforme descrito no plano de trabalho proposto na solicitação de renovação para o segundo ano, propõe-se dar continuidade ao projeto de braço robótico construído no primeiro ano, abordando-se os aspectos de conectividade e controle do braço através da internet. Como mencionado anteriormente, esta etapa não pode ser realizada durante o primeiro ano devido vários fatores, entre eles principalmente, o fato de que o estudo sobre redes e formas de conexão com a internet através de dispositivos micro-controlados revelou precisar mais tempo de estudo por parte da bolsista, sendo que este tópico não tinha sido contemplado no plano original. Outro fator importante foi a demora na aquisição das componentes de para conexão a internet: placas Ethernet Shield, Esp8266-01, e Wifi-Shield. Sendo que as duas primeiras foram de fato adquiridas enquanto a terceira ainda está em processo de aquisição. Assim, o plano de trabalho para o segundo ano propõe dedicar um tempo ao estudo destes tópicos, a familiarização com estas componentes e também ao estudo de linguagens de programação necessárias para o desenvolvimento de aplicações web.

Além disso, propõe-se também trabalhar em um segundo protótipo de dispositivo para Internet das coisas. A escolha do segundo protótipo a ser construído será decidido durante a execução da pesquisa. Da mesma forma que no plano anterior, se deverá trabalhar todas as etapas de construção do dispositivo micro-controlado, desde o desenvolvimento do circuito eletrônico, construção da estrutura física de suporte e programação do micro-controlador e do aplicativo via web.

9.- Participação em congressos e trabalhos publicados ou submetidos e outras atividades acadêmicas e de pesquisa

Os dois protótipos desenvolvidos nesta pesquisa, o Jogo Genius e o Braço Robótico controlado por um par de *Joysticks*, foram apresentados durante a VII Semana de Computação e Tecnologia da Informação da UENF, que aconteceu de 06 a 11 de Novembro de 2017, como parte de um minicurso ministrado pela autora desta pesquisa. O Certificado deste evento encontra-se no **Anexo I**.

10.- Data e assinatura do bolsista

29 de Abril de 2018, 

11.- Data e assinatura do orientador

29 de Abril de 2018, 

Semana Acadêmica 2017

UENF
Universidade Estadual do Norte Fluminense Darcy Ribeiro

CERTIFICADO

Certificamos que Isabela Correia Pereira ministrou o Minicurso: **Arduino: do zero ao jogo da Semana Acadêmica de Computação e Tecnologia da Informação**, realizada na **Universidade Estadual do Norte Fluminense Darcy Ribeiro – UENF**, no período de 06 a 11 de novembro de 2017, com carga horária de 04 horas.

Campos dos Goytacazes, 11 de novembro de 2017.

Registrado sob o nº 21311 no livro 01 folha 80.

Isabela Correia Pereira
Presidente da Comissão Organizadora

Marina Satika Suzuki
Pró-Reitora de Graduação

ProGraD

ANEXO I