UNIVERSIDADE CATÓLICA DE PERNAMBUCO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
COORDENAÇÃO DO CURSO DE FÍSICA

JOÃO VICTOR OLIVEIRA
PEDRO CERQUEIRA
RENATO LIRA
RODRIGO CAVALCANTI

DOG FEEDER

Recife

26/11/2018

JOÃO VICTOR OLIVEIRA
PEDRO CERQUEIRA
RENATO LIRA
RODRIGO CAVALCANTI

DOG FEEDER

Trabalho apresentado à disciplina Eletrônica básica, como parte dos requisitos necessários para obtenção de nota do 2º GQ.

Prof. Antônio Cruz.

Recife
26/11/2018

SUMÁRIO

1.0 - INTRODUÇÃO

Trabalho tem a ideia de facilitar o dia-a-dia de donos de pet em geral, automatizando a tarefa de colocar comida. Tendo também o foco no problema de deixar o animal sozinho em casa e ter a segurança que ele está alimentado.

1.1 – O Projeto

O Dog Feeder é um alimentador automático para animais domésticos, que traz a praticidade na hora de colocar a comida do seu animal. Funciona a partir de um aplicativo que você informa a hora em que você quer que o pote de ração seja enchido, uma ou duas vezes ao dia e a porção que você deseja.

1.1.1 – Justificativa

A escolha do projeto foi em busca da automatização de alguma tarefa presente no cotidiano da grande parte da população, pois hoje em dia é difícil ver uma casa que não possui um animal de estimação. Com isso, a tarefa que as vezes é até esquecida ou deixada pra lá, fica bastante simples, já que basta programar a hora e porção apenas uma vez. Sem falar de quando você vai precisar passar o dia fora ou viajar e precisa pedir para alguém colocar a ração do seu animal.

1.1.2 – Expectativas

Fazer um projeto útil, com fácil usabilidade e com interação até mesmo com o pet, esperamos que o animal aprenda que está na hora de comer.

2.0 - OBJETIVOS

O principal objetivo do projeto é otimizar uma tarefa diária que muitas vezes é até esquecida. Com isso implementamos uma interação entre usuário e máquina, via bluetooth, a partir de um aplicativo no celular, para controle de um motor (na liberação da comida) na qual só precisa ser programado, informando a hora, uma única vez ou utilizando uma função de liberação imediata de comida.

3.0 - MATERIAIS E MÉTODOS

3.1- Materiais

A. Módulo bluetooth HC-05 para comunicação com o app de celular;
B. Servo Motor (9g) com função de controlar saída da ração;
C. Clock PCF8563 utilizado para obter o horário;
D. Tela LCD 16x2, tela de interação com usuário, exibição da hora atual, exibição dos horários programados e exibição da função que ocorre no momento em questão.

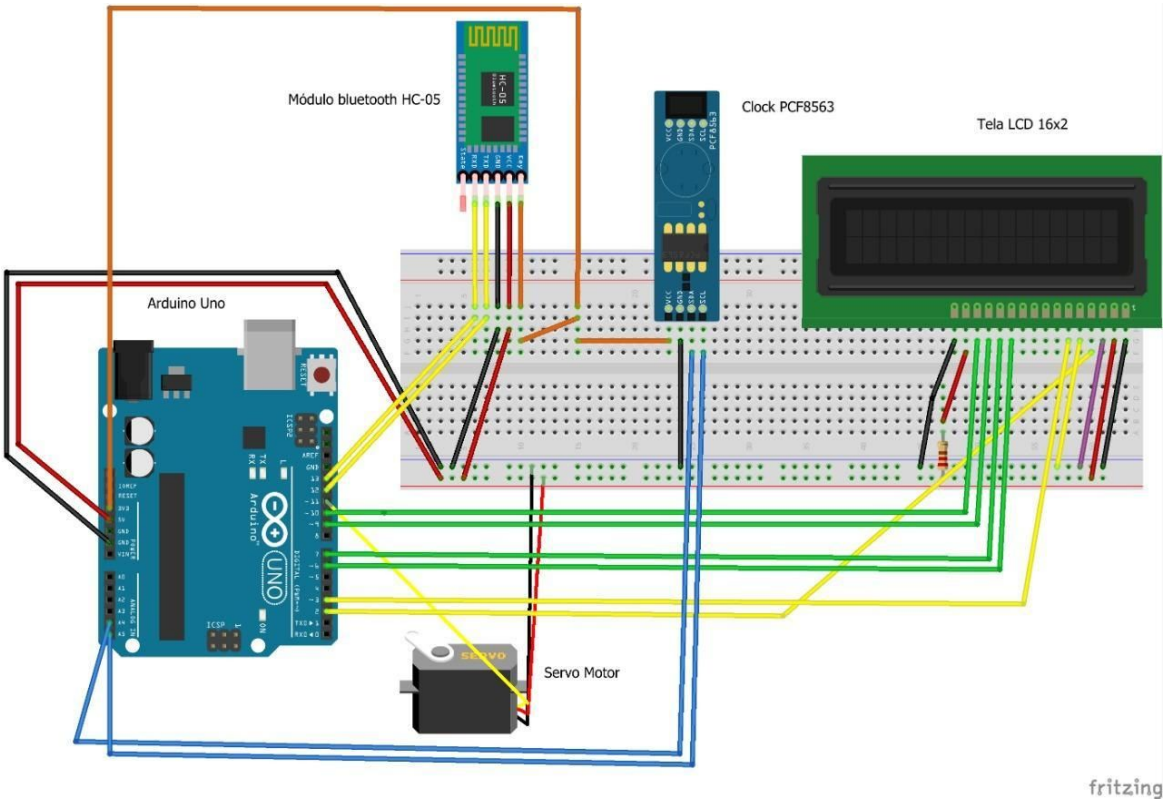
3.2- Metodologia

Projeto foi montado utilizando um Arduino Uno e duas protoboards. Uma protoboard com conexões de servo motor, clock PCF8563 e o módulo HC-05 e outra protoboard exclusiva para ligação da tela LCD, nessa última a ligação foi feita a partir de jumpers fêmeas podendo assim elevar a tela e conseguir fixar em uma posição de maior altura.

Foi necessário para montagem do projeto uma caixa em mdf, construída com 40cm de comprimento, 60cm de altura e 25cm de largura e internamente foi instalada uma prateleira para servir de suporte ao arduino e todo o resto do circuito, a prateleira possui 22cm de comprimento e 15cm de largura, a caixa também possui uma abertura para encaixe do visor LCD (7,5cmX2,5cm) e duas entrada para passagem de um cano (que servirá para passagem da ração) com diâmetro de 4cm.

O cano apresenta uma abertura e, contando com o posicionamento exato do servo motor, o projeto é capaz de controlar a queda da ração.

4.0 - MONTAGEM DO PROJETO



4.1 - Módulo bluetooth HC-05

Utilizado na função de escravo, ligação de 5 pinos para seu perfeito funcionamento: RX e TX (fazem a comunicação, envio ou recebimento de dados) ligados nas portas 13 e 12, GND ligado ao GND do Arduino, VCC ligado aos 5v do Arduino e KEY que funciona como um pulso, ligado aos 3,3v do Arduino.

4.2 - Módulo de Clock PCF8563

Ligação necessária apenas com 4 pinos do módulo: SCL (serial clock, temporização entre dispositivos) e SDA (serial data, transfere os dados) esses dois seguem o protocolo de barramento I2C, ligados nas portas A5 e A4 do Arduino, respectivamente, essas portas são de padrão do

Arduino Uno para ligação do SCL e SDA. Para uso do módulo também é necessário a ligação do pino VCC aos 3,3v e o GND ao GND do Arduino.

4.3 - Servo Motor

Para seu uso é necessária ligação de seus 3 pinos, VCC ligado aos 5v, GND ligado ao GND e uma ligação com porta digital, no caso, ligado a porta digital 11.

4.4 - Visor LCD 16x2

De seus 16 pinos foram usados somente 12, desse modo a passagem dos dados ao visor ficou de 4 bits por vez ao invés dos 8 bits. Começando da direita para a esquerda, os dois primeiros pinos servem de VCC e GND ligados respectivamente no VCC e GND do arduino, o terceiro pino é referente ao contraste das letras na tela, ligada ao GND pois assim fica maior contraste possível, o pino 4 indica um pino de controle (leitura) e foi feita ligação na porta digital 2, pino 5 serve para escrita e tem função de reconhecer o que foi escrito na tela, como não precisamos dele, então foi conectado no GND, o pino 6 conectado a porta 3 serve para indicar a habilitação do visor ao Arduino. Os pinos de 11 até 14 são de recebimento de bits e estão ligados nas portas digitais 6, 7, 9, 10, por fim, os pinos 15 e 16 são para ligar o background, funciona como led e são ligados a 5v e GND.

5.0 - O CÓDIGO DO PROJETO

```
#include <SoftwareSerial.h>
#include <Servo.h>
#include "refeicao.h"
#include "PCF8563.h"
#include "Wire.h"
#include <LiquidCrystal.h>

SoftwareSerial bluetooth(12, 13); //TX e RX (HC-05 se coincidem)

Servo servoMotor;
int pinServo = 11;

char entrada[9];//ex.: 1;15:30;7
bool novaEntrada; //validador para parar repetições
int i, comer;
bool salvo;
Refeicao ref1, ref2; //objetos que utilizam classe importada Refeicao

PCF8563 relogio; //objeto que utiliza classe importada PCF8563
LiquidCrystal lcd(2,3,6,7,9,10); //lcd apenas pra escrita com 4bits por vez

int bip = 4;
int exibeRef; //contador para exibir horários salvos

void setup() {
  Serial.begin(9600); //inicia console em 9600
  bluetooth.begin(9600); //inicia serial emulada em 9600
  servoMotor.attach(pinServo);
  servoMotor.write(45); //porta fechada
  Wire.begin(); //inicialização necessária para uso do módulo PCF8563
  i = 0; //serve para controlar posição do array
  lcd.begin(16,2); //tamanho da tela (2 linhas)
  pinMode(bip, OUTPUT);
  exibeRef = 0; //contagem para exibir horários salvos a cada 1min
}
void loop() {
  while (bluetooth.available()) { //dados são recebidos byte a byte (cada caracter por vez)
    delay(3); //parada para garantir que todos os caracteres são recebidos
    entrada[i] = bluetooth.read();
    novaEntrada = true; //indicador para validar a análise da string recebida
    i++;
  }
  relogio.atualizarHora();//necessária constante leitura no módulo para atualizar objeto
  lcd.setCursor(0, 0); //posiciona cursor no começo da tela
  lcd.print(receberHorarioFormatado(relogio.getHora(), relogio.getMinuto()));
  lcd.print(":");
  lcd.print(relogio.getSegundo());
```

```
  i = 0; //saindo do loop a próxima entrada vai sobrescrever a anterior, posição do array volta ao início
  if(novaEntrada){
    novaEntrada = false;
    salvo = false;
    if(entrada[0] == 49){//if = 1
       salvo = ref1.setRefeicao(entrada); //método retorna bool, assim valida se entrada foi aceita ou não
      if(salvo){
       lcd.setCursor(0,1);
       lcd.print("Ref1 "+ref1.getHorario()+"->"+ref1.getPorcao());
      }
    }
    else if(entrada[0] == 50){//if = 2
       salvo = ref2.setRefeicao(entrada); //método retorna bool, assim valida se entrada foi aceita ou não
      if(salvo){
       lcd.setCursor(0,1);
       lcd.print("Ref2 "+ref2.getHorario()+"->"+ref2.getPorcao());
      }
    }
    else if(comidaAgora(entrada)){ //if = comida;7 -> comando;porção
      lcd.setCursor(0,1);
      lcd.print("Liberar comida!");
      liberarComida(((int)entrada[7] - 48));
      tocarAlarme();
    }
    else if(!comidaAgora(entrada)){ //casos de entradas inválidas
      lcd.setCursor(0,1);
      lcd.print("Erro");
    }
  }
  comer = checarHorarioRefeicao(); //recebe número inteiro referente a refeição
  if(comer == 1){ //refeição salva 1
    lcd.setCursor(0,1);
    lcd.print("Hora da ref1");
    liberarComida(ref1.getPorcao());
    tocarAlarme();
  }
  else if(comer == 2){ //refeição salva 2
    lcd.setCursor(0,1);
    lcd.print("Hora da ref2");
    liberarComida(ref2.getPorcao());
    tocarAlarme();
  }
  if(comer == 1 || comer == 2){
       delay(60000); //1 minuto de espera, garante que caso esteja na hora de uma refeição, isso só ocorra 1 vez (liberar comida dura alguns segundos)
  }
```

```
  exibeRef++; //contador para mostrar horários salvos a cada 1min
  if(exibeRef == 60){
    lcd.setCursor(0,1);
    lcd.print("Ref1 "+ref1.getHorario()+"->"+ref1.getPorcao());
    delay(3000);
    lcd.setCursor(0,1);
    lcd.print("Ref2 "+ref2.getHorario()+"->"+ref2.getPorcao());
    delay(3000);
    exibeRef = 0;
  }
  delay(1000); //garantir que loop demora no mínimo 1 segundo
  lcd.clear();
}
bool comidaAgora(char *entrada){
  char validacao[7] = {'c','o','m','i','d','a',';'}; //Ex.: comida;7
  for(int i = 0; i<7; i++){
    if(validacao[i] != entrada[i]){
      return false;
    }
  }
  if((entrada[7] >= 48 && entrada[7] <= 57)){
    return true;
  }
  else{
    return false;
  }
}
void liberarComida(int porcao){
  int unidadePorcao = 250; //1 porção (30g)
  servoMotor.write(100); //porta aberta
  delay(unidadePorcao*porcao);
  servoMotor.write(45); //porta fechada
}
String receberHorarioFormatado(int hora, int minuto){
  String horario;

  if(hora < 10){
    horario = '0';
    horario += (char)hora%10;
  }
  else{
    horario = (char)hora/10;
    horario += (char)hora%10;
  }
  horario += ':';
  if(minuto < 10){
    horario += '0';
    horario += (char)minuto%10;
```

```
  }
  else{
    horario += (char)minuto/10;
    horario += (char)minuto%10;
  }
  return horario;
}
int checarHorarioRefeicao(){
   String horario = receberHorarioFormatado(relogio.getHora(), relogio.getMinuto()); //Ex.: 09:02
(só precisamos validar hora e minuto)
  if(horario.equals(ref1.getHorario())){
    return 1;
  }
  else if(horario.equals(ref2.getHorario())){
    return 2;
  }
  else{
    return 0;
  }
}
void tocarAlarme(){
  digitalWrite(bip, HIGH);
  delay(5000);
  digitalWrite(bip, LOW);
}
```

----------classe criada para controle do módulo PCF8563(.cpp)----------

```
#include "PCF8563.h"
#include "Arduino.h"
#include "Wire.h"
#define PCF8563address 0x51

byte PCF8563::bcdToDec(byte valor){
  return ((valor / 16) * 10 + valor % 16);
}
byte PCF8563::decToBcd(byte valor){
  return (valor / 10 * 16 + valor % 10);
}
void PCF8563::setHora(int segundo, int minuto, int hora, int diaSemana, int diaMes, int mes, int
ano){
  //módulo clock durante transmissão para a contagem (precisa ser algo rápido)
  Wire.beginTransmission(PCF8563address);//inicia transmissão com o clock
  Wire.write(0x02);
  Wire.write(decToBcd(segundo));
  Wire.write(decToBcd(minuto));
  Wire.write(decToBcd(hora));
  Wire.write(decToBcd(diaSemana));
  Wire.write(decToBcd(diaMes));
```

```
  Wire.write(decToBcd(mes));
  Wire.write(decToBcd(ano));
  Wire.endTransmission();//finaliza transmissão
}
void PCF8563::atualizarHora(){
  Wire.beginTransmission(PCF8563address);
  Wire.write(0x02);
  Wire.endTransmission();//devolve inteiro que identifica se houve erro
  Wire.requestFrom(PCF8563address, 7);

  this->segundo =   bcdToDec(Wire.read() & B01111111); // remove VL error bit
  this->minuto =    bcdToDec(Wire.read() & B01111111); // remove unwanted bits from MSB
  this->hora =      bcdToDec(Wire.read() & B00111111);
  this->diaMes =    bcdToDec(Wire.read() & B00111111);
  this->diaSemana = bcdToDec(Wire.read() & B00000111);
  this->mes =       bcdToDec(Wire.read() & B00011111);  // remove century bit, 1999 is over
  this->ano =       bcdToDec(Wire.read());
}
int PCF8563::getSegundo(){
  return this->segundo;
}
int PCF8563::getMinuto(){
  return this->minuto;
}
int PCF8563::getHora(){
  return this->hora;
}
int PCF8563::getDiaSemana(){
  return this->diaSemana;
}
int PCF8563::getDiaMes(){
  return this->diaMes;
}
int PCF8563::getMes(){
  return this->mes;
}
int PCF8563::getAno(){
  return this->ano;
}
```

----------classe criada como representação de uma refeição (.cpp)----------

```
#include "refeicao.h"
#include "Arduino.h"


//houve ajustes nas posições do array com a troca do uso da classe
bool Refeicao::validaEntrada(char *entrada){
  if(!(entrada[0] == 49 || entrada[0] == 50)){  //posição 0 só pode ser 1 ou 2 (Refeições)
    return false;
  }

   if(!((entrada[2] >= 48 && entrada[2] <= 50) && (entrada[3] >= 48 && entrada[3] <= 57))){
//posição 0 só pode ser 0, 1 ou 2 e posição 1 pode ser de 0 até 9 (Hora)
    return false;
  }

//   if(!(entrada[2] == 50) && (entrada[3] >= 48 && entrada[3] <= 51)){//caso a posição 2 seja 2,
então só pode ser 20h, 21h, 22h ou 23h (Hora)
//    return false;
//  }

   if(!((entrada[5] >= 48 && entrada[5] <= 53) && (entrada[6] >= 48 && entrada[6] <= 57))){
//posição 5 só pode ser de 0 até 5 e posição 6 pode ser de 0 até 9 (Minuto)
    return false;
  }

  if(!(entrada[8] >= 48 && entrada[8] <= 57)){  //posição 8 pode ser de 0 até 9 (Porções)
    return false;
  }

  return true;
}

bool Refeicao::setRefeicao(char *entrada){
          //exemplo.: 1;15:00;7
          if(Refeicao::validaEntrada(entrada)){
          this->hora[0] = entrada[2];
     this->hora[1] = entrada[3];
     this->hora[2] = ':';
     this->hora[3] = entrada[5];
     this->hora[4] = entrada[6];
     this->porcao = ((int)entrada[8] - 48); //conversão da tabela ASCII para inteiro
     return true;
          }

          else{
```

```
        return false;
        }
}
String Refeicao::getHorario(){
  String horario;
  for(int i = 0; i<5; i++){
    horario += this->hora[i];
  }
  return horario;
}
int Refeicao::getPorcao(){
  return porcao;
}
```

6.0 - CONCLUSÃO

Com a realização do projeto conseguimos atingir nossos objetivos e ainda melhorar a ideia inicial. Conseguimos obter mais aprofundamento na parte prática de eletrônica e aprender sobre a programação do Arduino, bem como conhecer algumas de suas bibliotecas, logo, abriu ainda mais o leque de áreas que podemos seguir ao terminar o curso, seja na parte da montagem ou de programação.

Acreditamos também que existem detalhes a melhorar, seja na montagem da caixa, em que não contamos com um reservatório para se acumular uma maior quantidade de comida, como também poderíamos ter incluído uma fonte alternativa de energia, assim o circuito funcionaria mesmo sem um computador por perto.

7.0 - REFERÊNCIAS

- www.arduino.cc
- Fóruns sobre eletrônica
- www.youtube.com/brincandocomideias
- Notas do professor
- Outros projetos semelhantes a ideia do nosso
- labdegaragem.com