DOCUMENTATIE

PROIECTARE CU MICROPROCESOARE

-Dispenser de hrana pentru animale-

Student: ARBA ELEONORA

Grupa: 30239

CUPRINS

1. Introducere
2. Solutie propusa si implementare
3. Testare si validare
4. Concluzii

CAPITOLUL I -Introducere

Proiectul pe care l-am ales isi propune sa dezvolte un robotel care sa fie utilizat pentru a hrani pisici la o anumita ora. Astfel incat utilizatorul prin introducerea unei ore in formatul cerut garanteaza ca pisica va primi un garamaj de mancare pedefinit la ora stabilita. Am ales acest proiect pentru a avea parte de ceva care este de folos si datorita costurilor ridicate ale unui astfel de dispozitiv din comert.

CAPITOLUL II - Solutie propusa si implementare

Solutia propusa de mine este utilizarea unui keypad pentru introducerea datelor dorite, folosirea unui RTC(Real Time Clock) DS1307 care sa ma ajute sa obtin ora curenta, un servomotor MG995 pentru a realiza permiterea curegerii macarii din recipientului creea de catre mine, un LCD pentru a afisa ora curenta si ora introdusa la care se asteapta sa se faca hranirea si desigur o placuta de dezvolatre Arduino in cazul meu Arduino Uno.

Implemenatrea a fost gandita in urmatorul fel: de la keypad datele vor fi introduce ora(ex:12) urmata de “\*”(care face trecerea in algoritm de la scrierea in variabila de ora in cea de minute) minutele(ex:34) urmat de “#” care face concatenarea ore si a minutelor in format HH:MM deci vom avea in final ora formatata 12:34. In cazul introducerii minutelor sau a orei gresit inainte de a se apasa”#” pentru formatarea stringului se poate folosi “C” pentru clear care va sterge tot din variabilele de minut si ora si ne va permite sa reintroducem datele dorite.

Componenta RTC(Real Time Clock) este folosita pentru a obtine ora curenta care de asemnea se formateaza intr-un string care va fi comparat cu stringul generat prin introducerea de la keypad, atunci cand cele doua sunt egale se realizeaza deschiderea servomotorului atasat modelului construit de mine care determina curgerea hranei.

Lcd-ul este utilizat pentru a afisa ora curenta si ora introdusa de la keypad pentru a sti cand se va face hranirea.

CAPITOLUL III- Testare si Validare

In momentul implementarii am intalnit mai multe probleme cea mai mare fiind faptul ac RTC-ul nu se sincronizeaza perfect de fiecare data cu ora curenta ceea ce creeaza idea ca de fapt nu ar functioa alte componente, dar LCD-ul ajuta in solutionarea acestei probleme in cazul in care RTC-ul nu se sincronizeaza bine putem calcula ora la care vrem sa se afca hranirea in concordanta cu ora primita de la RTC.

CAPITOLUL IV- COMPONENTE

1. Keypad (Matricea de tastare): Folosit pentru introducerea manuală a orei și minutei de către utilizator. Oferă interfața de intrare pentru setarea programului de hrănire.
2. RTC (Real Time Clock - Ceas în timp real DS1307): Utilizat pentru a obține ora curentă și a asigura sincronizarea corectă a sistemului. Realizează funcția de ceas în timp real necesară pentru programarea hrănirii la ore precise.
3. Servo Motor (MG995): Motorul servo controlează deschiderea unui recipient pentru hrană. Este activat la ora programată pentru a elibera hrana.
4. LCD (Display cu cristale lichide I2C): Afișează ora curentă și ora programată pentru hrănire. Oferă informații vizuale utilizatorului.
5. Arduino Uno: Placa de dezvoltare care controlează toate componentele. Găzduiește codul sursă și asigură interacțiunea între toate modulele.

CAPITOLUL V- Concluzii

Proiectul meu a fost realizat in mare masura pe cat mi-as fi dorit, pe viitor as vrea sa adaug o functionalitate pentru a alege cantitatea de mancare distribuita.

Consider ca acest dispozitiv este util, iar deoarece costurile de realizare sunt mai mici decat ale unui dispozitiv de acelasi gen din comert il face cu atat mai atractiv si avantajos.

CAPITOLUL VI- CODUL

#include <Wire.h>

#include <RTClib.h>

#include <Keypad.h>

#include <Servo.h>

RTC\_DS1307 rtc;

int oraCurenta = 0;

int minutCurent = 0;

int ok = 0;

const byte ROWS = 4;

const byte COLS = 4;

char hexaKeys[ROWS][COLS] = {

{'D', 'C', 'B', 'A'},

{'#', '9', '6', '3'},

{'0', '8', '5', '2'},

{'\*', '7', '4', '1'}

};

byte rowPins[ROWS] = {2, 3, 4, 5};

byte colPins[COLS] = {6, 7, 8, 9};

Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);

int hourValue = 0;

int minuteValue = 0;

int nrcifre = 0;

String formattedTime = "";

String oraSiMinut = "";

bool enteringHour = true;

Servo myservo;

int servoMoving = 1;

void setup() {

Serial.begin(9600);

myservo.attach(11);

delay(500);

myservo.write(120); // Set servo angle to 90 degrees

if (!rtc.begin()) {

Serial.println("Couldn't find RTC");

while (1);

}

if (!rtc.isrunning()) {

// Serial.println("RTC is NOT running!");

rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_)));

}

}

void loop() {

//rtc

DateTime now = rtc.now();

oraCurenta = now.hour();

minutCurent = now.minute();

oraSiMinut = String(oraCurenta) + ':' + (minutCurent < 10 ? "0" : "") + String(minutCurent);

//keypad

char customKey = customKeypad.getKey();

if (customKey) {

if (customKey >= '0' && customKey <= '9') {

if (enteringHour) {

if (nrcifre < 2) {

hourValue = (hourValue \* 10 + (customKey - '0'));

if (hourValue > 23) {

hourValue = 23;

}

nrcifre++;

} else {

enteringHour = false;

nrcifre = 0;

}

} else {

if (nrcifre < 2) {

minuteValue = (minuteValue \* 10 + (customKey - '0'));

if (minuteValue > 59) {

minuteValue = 59;

}

nrcifre++;

}

}

} else if (customKey == '#') {

formattedTime = String(hourValue) + ":" + (minuteValue < 10 ? "0" : "") + String(minuteValue);

hourValue = 0;

minuteValue = 0;

enteringHour = true;

nrcifre = 0;

} else if (customKey == 'C') {

hourValue = 0;

minuteValue = 0;

enteringHour = true;

nrcifre = 0;

} else if (customKey == '\*') {

enteringHour = !enteringHour;

nrcifre = 0;

}

}

verificare();

}

bool servoMoved = false;

void verificare() {

if (!oraSiMinut.equals(formattedTime)) {

// Serial.println("nu");

//Serial.print("Ora formată: ");

Serial.println(formattedTime);

Serial.println(oraSiMinut);

servoMoved = false;

} else if (oraSiMinut.equals(formattedTime) && servoMoved == false) {

myservo.write(90);

servoMoved = true;

}

}