FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

DEPARTAMENTUL AUTOMATICĂ

SINTEZA

Proiectului de laborator cu titlul:

Dozator automat de medicamente

Autor: Ștefania-Liliana Mutu

Grupa 30123

I. Cerințele temei:

Cerința propusă a fost predată sub următoarea formă: „Sunt o persoană ocupată și trebuie să fac un tratament la ore fixe de 3 ori pe zi. Am nevoie de un dozator de medicamente”. Prin urmare, proiectul realizat reprezintă un dozator automat de medicamente.

II. Soluții alese:

În ceea ce privește implementarea aplicației, principiile sunt simple și ușor de gestionat. Aplicația este compusă dintr-o parte de testare, pentru simplificarea prezentării, și una propriu-zisă care implementează funcționalitatea cerută a proiectului, și anume dispensarea medicației la ore fixe de 3 ori pe zi.

Dozatorul distribuie pastilele o dată la 6 ore, începând cu ora 08:00, continuând cu 14:00 și în cele din urmă cu 20:00, pentru a crea un program de administrare a medicamentelor cât mai sănătos și ușor de respectat. În ceea ce privește funcționalitatea sa în funcția de testare, acesta distribuie pastilele o dată pe minut pentru a demonstra modul de operare rapid.

Sistemul verifică periodic ora curentă comparând-o cu un set de ore predefinite, mai exact 08, 14 și 20, iar în momentul în care cele două ore coincide, algoritmul de dozare este activat, iar un indicator este setat pentru a marca orele la care medicamentele au fost eliberate. Dozarea constă în eliberarea pastilelor și anunțarea momentului de administrare prin textul „Luați medicația!”, printr-un semnal sonor, dar și prin aprinderea unui LED de culoare roșie care indică faptul că medicamentul nu a fost luat. După momentul dozării, algoritmul revine la etapa comparării orelor, așteptând următoarea oră de dozare. La ora 00:00, algoritmul resetează indicatorii aferenți orelor de dozare, pentru a-și relua funcția în următoarea zi.

III. Rezultate obținute:

Proiectul a fost implementat utilizând o serie de componente electronice conectate la o placă de dezvoltare compatibilă cu Arduino Uno, printr-o placă de testare. Placa de dezvoltare va fi alimentată cu curent de la laptop pentru a-i fi demonstrată funcționalitatea.

Componentele utilizate, funcția acestora și felul în care sunt conectate la placa de dezvoltare:

→ Servomotorul pozițional SG90 rotește fundul detașabil al recipientului în care medicamentele urmează să fie depozitate atunci când este ora dozării, astfel încât medicația să fie eliberată.

- Firul maro -> GND
- Firul roșu -> VCC
- Firul galben -> pinul digital 3, care are semnal PWM

→ Ecranul LCD 16*2, având fiecare pin conectat la un modul I2C pentru o mai ușoară conectare la placa de dezvoltare, afișează pe prima linie timpul current sub formatul hh:mm, iar pe a doua linie un cronometru, care arată timpul rămas pâna la următoarea dozare sub același format. În momentul dozării acesta va afișa mesajul „Luați medicația!”, text care va dispărea doar după apăsarea butonului.

- GND -> GND
- VCC -> VCC
- SDA -> SDA
- SCL -> SCL

→ Modulul RTC DS3231 ține evidența timpului și a datei curente, având o baterie incorporată pentru a nu rămâne în urmă atunci când întregul circuit nu este alimentat cu curent.

- GND -> GND
- VCC -> VCC
- SDA -> pinul analogic A4
- SCL -> pinul analogic A5

→ Modulul cu buzzer activ emite semnale sonore pentru a anunța utilizatorul cu privier la momentul dozării.

- GND -> GND
- VCC -> VCC
- I/O -> pinul digital 10

→ 2 LED-uri (roșu, verde) anunță vizual utilizatorul cu privire la starea administrării medicației.

- Anodul (+) LED-ului roșu -> rezistență -> pinul digital 9
- Catodul (-) LED-ului roșu -> GND
- Anodul (+) LED-ului verde -> rezistență -> pinul digital 8
- Catodul (-) LED-ului verde -> GND

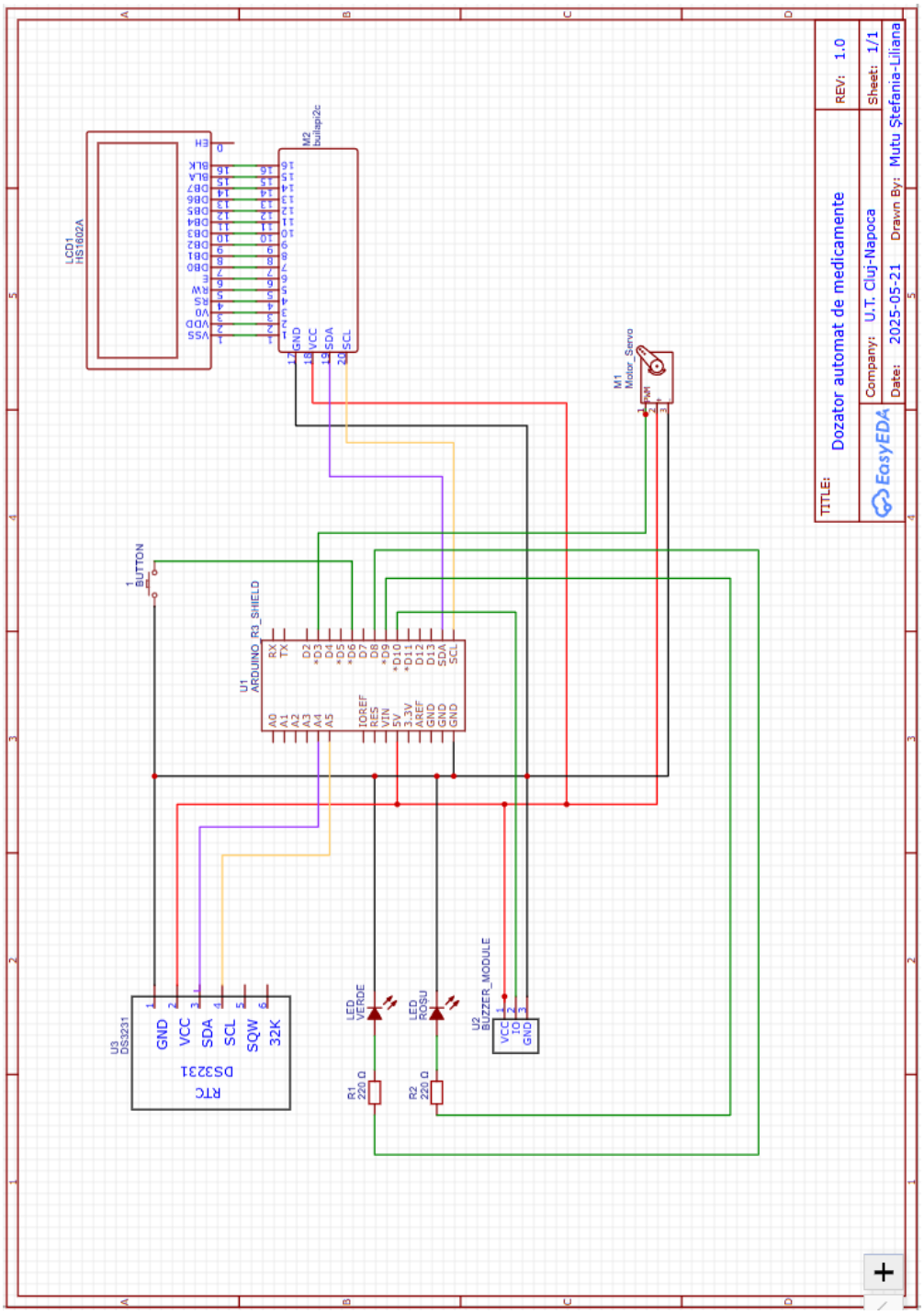
→ Butonul indică că medicația a fost administrată. O dată apăsat va opri LED-ul roșu și îl va porni pe cel verde, iar textul de pe LCD, „Luați medicația!”, va dispărea și va reapărea cronometrul până la următoarea dozare.

- Primul pin -> GND
- Pinul opus pe diagonală față de primul -> pinul digital 6

→ Fire de legătură (jumper).

→ 2 rezistențe 220 kΩ.

Schema electrică:



IV. Testări și modificări:

Dozatorul automat de medicamente funcționează simplu. Modul de utilizare al acestuia este următorul:

1. Conectați dispozitivul la o sursă de curent;
2. Adaugați medicația în recipientul din partea de sus;
3. Așteptați momentul dozării;
4. După momentul dozării, după ce au încetat cele trei semnale auditive, a apărut textul „Luați medicația!” pe ecranul din partea de sus și LED-ul roșu s-a aprins, puteți colecta medicația din recipientul de jos;
5. După administrarea acesteia apăsați pe butonul din partea de sus, mesajul „Luați medicația!” va dispărea de pe ecran, LED-ul roșu se va opri, iar cel verde va porni pentru a nu uita dacă medicația a fost luată sau nu;
6. Repetați procesul de la pasul 2.

De asemenea, dozatorul are implementată o funcție de testare pentru a-i putea fi demonstrată funcționalitatea într-un mod rapid și ușor. Modul de utilizare este același, singurul lucru care se schimbă este intervalul orar al dozării. Medicația va fi dozată o dată la un minut, în locul de 6 ore.

Anexe:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <RTClib.h>
#include <Servo.h>

// Pentru functia de testare
// #define MODE_TEST

LiquidCrystal_I2C lcd(0x27, 16, 2);
RTC_DS3231 rtc;
Servo servo;

const int servoPin = 3;
const int resetButtonPin = 6;
const int redLedPin = 9;
const int greenLedPin = 8;
const int buzzerPin = 10;

DateTime lastDoseTime;
bool doseDispensed = false;
bool buzzerDone = false;

#ifdef MODE_TEST
const long testIntervalSec = 60;
#endif

int doseHours[] = {8, 14, 20};

void setup() {
  Serial.begin(9600);
```

```
  lcd.init();
  lcd.backlight();

  if (!rtc.begin()) {
    lcd.print("RTC ERROR");
    while (1);
  }

  if (rtc.lostPower()) {
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
  }

  servo.attach(servoPin);
  servo.write(0);

  pinMode(resetButtonPin, INPUT_PULLUP);
  pinMode(redLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);

  digitalWrite(greenLedPin, HIGH);
  digitalWrite(redLedPin, LOW);
  digitalWrite(buzzerPin, HIGH);

#ifdef MODE_TEST
  lastDoseTime = rtc.now();
#endif
}

void loop() {
  DateTime now = rtc.now();
```

```
  lcd.setCursor(0, 0);
  lcd.print("Ora: ");
  printHourMinute(now.hour(), now.minute());
  lcd.print("      ");

  bool timeToDispense = false;

#ifdef MODE_TEST
  long elapsed = now.unixtime() - lastDoseTime.unixtime();
  long remaining = testIntervalSec - elapsed;

  lcd.setCursor(0, 1);
  if (remaining > 0) {
    int h = remaining / 3600;
    int m = (remaining % 3600) / 60;
    int s = remaining % 60;
    lcd.print("Urm: ");
    printTime(h, m, s);
    lcd.print(" ");
  } else {
    lcd.setCursor(0, 1);
    lcd.print("Luati medicatia! ");
    timeToDispense = true;
  }

#else
  DateTime nextDose = getNextDoseTime(now);
  long remaining = nextDose.unixtime() - now.unixtime();

  if (remaining <= 0 && !doseDispensed) {
    lcd.setCursor(0, 1);
    lcd.print("Luati medicatia! ");
```

```
    timeToDispense = true;
  } else {
    int h = remaining / 3600;
    int m = (remaining % 3600) / 60;

    lcd.setCursor(0, 1);
    lcd.print("Urm: ");
    if (h < 10) lcd.print("0");
    lcd.print(h);
    lcd.print(":");
    if (m < 10) lcd.print("0");
    lcd.print(m);
    lcd.print("      ");
  }
#endif

  if (timeToDispense && !doseDispensed) {
    dispensePills();
    doseDispensed = true;
    buzzerDone = false;
  }

  if (digitalRead(resetButtonPin) == LOW) {
    delay(100);
    if (digitalRead(resetButtonPin) == LOW) {
#ifdef MODE_TEST
      lastDoseTime = now;
      buzzerDone = false;
#endif
      digitalWrite(greenLedPin, HIGH);
      digitalWrite(redLedPin, LOW);
      digitalWrite(buzzerPin, HIGH);
```

```
      doseDispensed = false;
      lcd.clear();
    }
  }

  delay(200);
}

void dispensePills() {
  for (int pos = 0; pos <= 180; pos++) {
    servo.write(pos);
    delay(10);
  }
  delay(1000);
  for (int pos = 180; pos >= 0; pos--) {
    servo.write(pos);
    delay(10);
  }

  digitalWrite(redLedPin, HIGH);
  digitalWrite(greenLedPin, LOW);

  if (!buzzerDone) {
    for (int i = 0; i < 3; i++) {
      digitalWrite(buzzerPin, LOW);
      delay(1000);
      digitalWrite(buzzerPin, HIGH);
      delay(1000);
    }
    buzzerDone = true;
  }
}
```

```
DateTime getNextDoseTime(DateTime now) {
  for (int i = 0; i < 3; i++) {
    if (now.hour() < doseHours[i] || (now.hour() == doseHours[i] && now.minute()
== 0)) {
      return DateTime(now.year(), now.month(), now.day(), doseHours[i], 0, 0);
    }
  }
  DateTime nextDay = now + TimeSpan(1, 0, 0, 0);
  return DateTime(nextDay.year(), nextDay.month(), nextDay.day(), doseHours[0],
0, 0);
}

void printHourMinute(int h, int m) {
  if (h < 10) lcd.print("0");
  lcd.print(h);
  lcd.print(":");
  if (m < 10) lcd.print("0");
  lcd.print(m);
}

void printTime(int h, int m, int s) {
  if (h < 10) lcd.print("0");
  lcd.print(h);
  lcd.print(":");
  if (m < 10) lcd.print("0");
  lcd.print(m);
  lcd.print(":");
  if (s < 10) lcd.print("0");
  lcd.print(s);
}
```