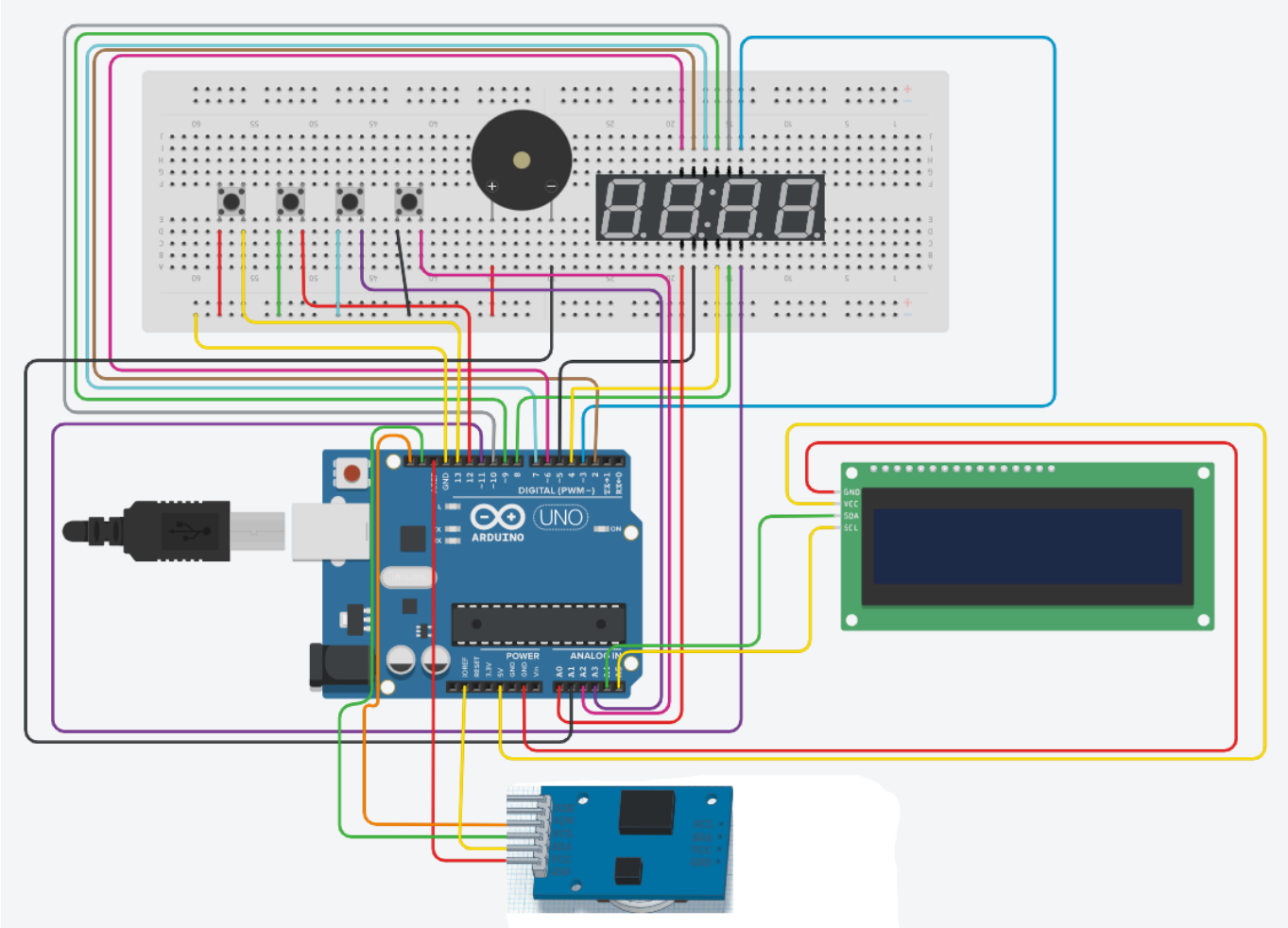
Real Time Clock și Timer cu Alarmă pentru bucătărie

Autor: Maria-Alexandra Barbu

Introducere

Acest proiect realizează un **ceas** în timp real care are și funcție de **timer** cu alarmă- un obiect util în bucătărie, pentru a măsura timpul de gătit și a fi mereu la curent cu trecerea timpului. Cât timp nu este apăsat butonul de Turn_On/Reset pentru timer, pe LCD se afișează ora, minutul, secunda, ziua, luna și anul în timp real. Funcția de timer se pornește prin apăsarea unui buton. Timer-ul este construit cu ajutorul unui afișor de tip **7-segmente** cu 4 cifre, deci poate măsura trecerea a maxim 9999 de secunde, adică aproape 3 ore (suficient pentru a-și îndeplini funcționalitatea de kitchen timer). Acesta pooate fi ajustat să numere oricâte secunde se dorește cu ajutorul celor două butoane “plus” și “minus” (în modul default, el numără 60 de secunde). Odată ce timpul s-a scurs, buzzer-ul va cânta o melodie de alarmă, iar pe LCD se va afișa un mesaj cu animații menit să anunțe utilizatorul că poate opri aragazul. Timerul poate fi resetat de oricâte ori prin apăsarea unui buton. Mai jos poate fi văzută schema circuitului realizată în **Tinkercad**.



Descriere generală

În următoarea imagine se poate observa **schema bloc** a proiectului. La plăcuța Arduino am conectat direct un Breadboard, un LCD și un modul ceas de timp real cu baterie (RTC). Acest modul, odată setat la prima utilizare, poate menține ora corectă la următoarele utilizări, chiar dacă plăcuța Arduino este deconectată de la PC. Atât modulul RTC, cât și LCD-ul, folosesc protocolul **I2C**, ceea ce face conexiunea la plăcuță mult mai ușoară și aduce avantajul economisirii de pini. La Breadboard am conectat 4 butoane, un buzzer piezoelectric și un afișor 7-segmente cu 4 cifre care afișează numărătoarea inversă atunci când funcția de timer este activată. Acest display are 12 pini (unul dintre pini este nefolosit, pinii A-G sunt pentru fiecare dintre cele 7 segmente ale unei cifre, iar pinii D1-D4 au rolul de a indica starea fiecăreia dintre cele 4 cifre- dacă cifra respectivă afișează ceva, pinul corespunzător va fi pe LOW, altfel va fi pe HIGH).

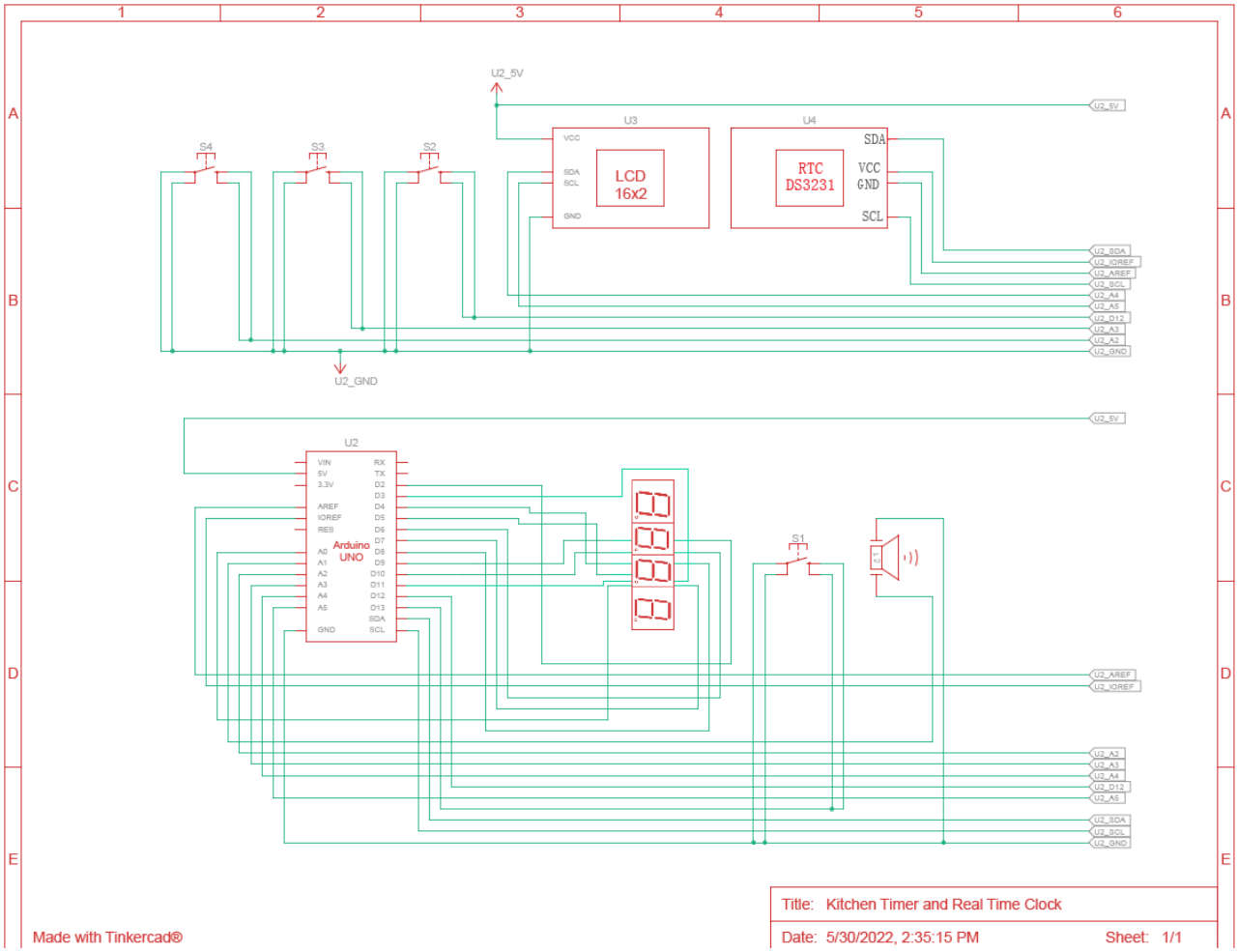
Hardware Design

Piese necesare pentru **design-ul hardware**:

- Placă de dezvoltare Arduino Uno

- Breadboard
- LCD 16×2 (cu I2C)
- Modul RTC DS3231
- 4-digit 7-segment Display
- Buzzer
- 4 push buttons
- 22 fire tată-tată pentru conectarea afișorului de timer, a buzzerului și a butoanelor
- 8 fire mamă-tată pentru conectarea modulului RTC și a LCD-ului

Mai jos am reprezentat în **Eagle** schema electrică a proiectului.



Software Design

Ca și mediu de dezvoltare am folosit **Arduino IDE**. Principalele funcții implementate sunt:

- void **setup()** –> configurează pinii utilizați de 4-digit 7-segment Display și de Buzzer în modul OUTPUT; configurează pinii utilizați de cele 4 butoane în modul INPUT_PULLUP; inițializează modulul RTC și LCD-ul prin apelarea funcției "**begin()**"
- void **lightNumber(int numberToDisplay)** –> primește ca argument o cifră și, pentru a o putea afișa, scrie valori de HIGH sau LOW pe cei 7 pini corespunzători celor 7-segmente ale display-ului de timer;
- void **SwitchDigit(int digit)** –> primește ca argument un index de la 0 la 3 și scrie valori HIGH sau

LOW pe cei 4 pini D1-D4 astfel încât să se afișeze cifra de la indexul primit;
- struct digits **todigits(int n)** –> funcție care primește un număr de maxim 4 cifre și va returna într-o structură toate cifrele care trebuie afișate pe 7-segment display;
- void **PrintNumber(int n, int time)** –> primește ca argument un număr de maxim 4 cifre și un timp exprimat în milisecunde și afișează numărul pe 4-digit 7-segment display atât de mult timp cât indică al doilea argument;
- void **displayTime()** –> afișează ora, minutul, secunda, ziua, luna și anul în timp real pe LCD
- bool **countdown(int n, int del)** –> afișează pe 4-digit 7-segment display numărătoarea descrescătoare de la n la 0; după afișarea lui 0, Buzzerul cântă o melodie de la Nokia care semnalează faptul că timpul de gătire s-a încheiat; după ce Buzzer-ul termină de cântat, LCD-ul afișează mesajul “Timpul s-a scurs!”, înconjurat de niște animații care se repetă de 3 ori; în urma derulării animațiilor, pe LCD reapare ora și se reactivează funcția de ceas;
- void **reset()** –> implementează funcționalitatea celor 4 butoane: butonul 1 este butonul care pornește numărătoarea inversă- cât timp nu este apăsat, se pot apăsa butoanele 3 și 4 (PLUS și MINUS) pentru a ajusta numărul de secunde ce se doresc a fi măsurate;
- void **loop()** –> implementează flow-ul programului apelând funcțiile de mai sus;

Biblioteci folosite:

- **LiquidCrystal_I2C**–> bibliotecă ce conține funcțiile de lucru cu LCD-ul
- **DS3232RTC**–> bibliotecă ce conține funcțiile de lucru cu modulul RTC

Cod Sursă

```
#include <math.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// librarie necasara pentru afisarea la LCD
#include <DS3232RTC.h>
// librarie necesara pentru functionarea modulului RTC

DS3232RTC myRTC;
LiquidCrystal_I2C lcd(0x27, 16, 2);
// 2 linii si 16 caractere pe fiecare linie

#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_E4  330
#define NOTE_FS4 370
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_B4  494
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_E5  659

int D1 = 6;
int D2 = 9;
int D3 = 10;
```

```
int D4 = 11;
int buzzer = A1;
int pinA = 2;
int pinB = 3;
int pinC = 4;
int pinD = 5;
int pinE = A0;
int pinF = 7;
int pinG = 8;
int button1=13; // Start numaratoare inversa
int button2=12; // Reset
int button3=A3; // Minus
int button4=A2; // Plus
int initial_time = 60;
int tempo = 165;
byte char00[] = {B00001, B00011, B00111, B01111, B11111, B11111, B11111,
B11111};
byte char01[] = {B11111, B11111, B11111, B11111, B11111, B11111, B11111,
B10101};
byte char02[] = {B10000, B11000, B11100, B11110, B11111, B11111, B11111,
B11111};
byte char10[] = {B11111, B11111, B11111, B11111, B01111, B00111, B00011,
B00001};
byte char11[] = {B11111, B10101, B11011, B11111, B11111, B11111, B11111,
B11111};
byte char12[] = {B11111, B11111, B11111, B11111, B11110, B11100, B11000,
B10000};
byte animation[] = {B11111, B10001, B10001, B10001, B10001, B11111, B11111,
B11111};
// nota si durata ei- patrimi, optimi sau doimi
int melody[] = {NOTE_E5, 8, NOTE_D5, 8, NOTE_FS4, 4, NOTE_GS4, 4, NOTE_CS5,
8, NOTE_B4, 8,
  NOTE_D4, 4, NOTE_E4, 4, NOTE_B4, 8, NOTE_A4, 8, NOTE_CS4, 4, NOTE_E4, 4,
NOTE_A4, 2};
int notes = sizeof(melody) / sizeof(melody[0]) / 2;
// numarul total de note muzicale
int wholenote = (60000 * 4) / tempo;
// durata unei note muzicale
int divider = 0, noteDuration = 0;

struct digits {
    int digit[4];
};

void setup() {
  Serial.begin(9600);
  pinMode(pinA, OUTPUT);
  pinMode(pinB, OUTPUT);
  pinMode(pinC, OUTPUT);
  pinMode(pinD, OUTPUT);
  pinMode(pinE, OUTPUT);
```

```
  pinMode(pinF, OUTPUT);
  pinMode(pinG, OUTPUT);
  pinMode(D1, OUTPUT);
  pinMode(D2, OUTPUT);
  pinMode(D3, OUTPUT);
  pinMode(D4, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(button1,INPUT_PULLUP);
  pinMode(button2,INPUT_PULLUP);
  pinMode(button3,INPUT_PULLUP);
  pinMode(button4,INPUT_PULLUP);
  lcd.begin();
  lcd.backlight();
  lcd.home();
  myRTC.begin();
}

void lightNumber(int numberToDisplay) {
// functie de afisare a unei cifre la 7-segment display
  switch (numberToDisplay){
  case 0:
    digitalWrite(pinA, HIGH);
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, HIGH);
    digitalWrite(pinE, HIGH);
    digitalWrite(pinF, HIGH);
    digitalWrite(pinG, LOW);
    break;

  case 1:
    digitalWrite(pinA, LOW);
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, LOW);
    digitalWrite(pinE, LOW);
    digitalWrite(pinF, LOW);
    digitalWrite(pinG, LOW);
    break;

  case 2:
    digitalWrite(pinA, HIGH);
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, LOW);
    digitalWrite(pinD, HIGH);
    digitalWrite(pinE, HIGH);
    digitalWrite(pinF, LOW);
    digitalWrite(pinG, HIGH);
    break;

  case 3:
```

```
    digitalWrite(pinA, HIGH);
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, HIGH);
    digitalWrite(pinE, LOW);
    digitalWrite(pinF, LOW);
    digitalWrite(pinG, HIGH);
    break;

  case 4:
    digitalWrite(pinA, LOW);
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, LOW);
    digitalWrite(pinE, LOW);
    digitalWrite(pinF, HIGH);
    digitalWrite(pinG, HIGH);
    break;

  case 5:
    digitalWrite(pinA, HIGH);
    digitalWrite(pinB, LOW);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, HIGH);
    digitalWrite(pinE, LOW);
    digitalWrite(pinF, HIGH);
    digitalWrite(pinG, HIGH);
    break;

  case 6:
    digitalWrite(pinA, HIGH);
    digitalWrite(pinB, LOW);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, HIGH);
    digitalWrite(pinE, HIGH);
    digitalWrite(pinF, HIGH);
    digitalWrite(pinG, HIGH);
    break;

  case 7:
    digitalWrite(pinA, HIGH);
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, LOW);
    digitalWrite(pinE, LOW);
    digitalWrite(pinF, LOW);
    digitalWrite(pinG, LOW);
    break;

  case 8:
    digitalWrite(pinA, HIGH);
```

```
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, HIGH);
    digitalWrite(pinE, HIGH);
    digitalWrite(pinF, HIGH);
    digitalWrite(pinG, HIGH);
    break;

  case 9:
    digitalWrite(pinA, HIGH);
    digitalWrite(pinB, HIGH);
    digitalWrite(pinC, HIGH);
    digitalWrite(pinD, HIGH);
    digitalWrite(pinE, LOW);
    digitalWrite(pinF, HIGH);
    digitalWrite(pinG, HIGH);
    break;

  case 10:
    digitalWrite(pinA, LOW);
    digitalWrite(pinB, LOW);
    digitalWrite(pinC, LOW);
    digitalWrite(pinD, LOW);
    digitalWrite(pinE, LOW);
    digitalWrite(pinF, LOW);
    digitalWrite(pinG, LOW);
    break;
  }

}

void SwitchDigit(int digit) {
// functie ce alege care dintre cele 4 cifre ale 7-segment display-ului va
fi afisata
  for (int i = 0; i < 4; i++) {
    if (i == digit) {
      if (i == 0) {
        digitalWrite(D1, LOW);
        digitalWrite(D2, HIGH);
        digitalWrite(D3, HIGH);
        digitalWrite(D4, HIGH);
      }
      else if (i == 1) {
        digitalWrite(D1, HIGH);
        digitalWrite(D2, LOW);
        digitalWrite(D3, HIGH);
        digitalWrite(D4, HIGH);
      }
      else if (i == 2) {
        digitalWrite(D1, HIGH);
        digitalWrite(D2, HIGH);
```

```
        digitalWrite(D3, LOW);
        digitalWrite(D4, HIGH);
      }
      else if (i == 3) {
        digitalWrite(D1, HIGH);
        digitalWrite(D2, HIGH);
        digitalWrite(D3, HIGH);
        digitalWrite(D4, LOW);
      }
    }
  }
}

struct digits todigits(int n){
// functie care primeste un numar de maxim 4 cifre si va returna
// intr-o structura toate cifrele care trebuie afisate pe 7-segment display
  struct digits mystruct;
  int zeros = 0;
  int d;
  for (int i = 0; i < 4; i++) {
    d = n / pow(10, 3 - i);
    zeros += d;
    n = n - d * pow(10, 3 - i);
    if ((zeros != 0) || (i==3)) {
      mystruct.digit[i] = d;
      // se va afisa cifra "d" pe pozitia "i"
    } else {
      mystruct.digit[i] = 10;
      // nu se va afisa nicio cifra pe pozitia "i"
    }
  }

return mystruct;
}

void PrintNumber(int n, int time) {
  struct digits mystruct;
  mystruct = todigits(n);

  for (int i = 0; i <= time / 20; i++) {
    if (digitalRead(button2) == LOW) {
    // daca este apasat butonul de reset in timpul numararii
      return;
    }
    for (int j = 0; j < 4; j++) {
      // afiseaza pe rand fiecare cifra cu delay atat de mic
      // incat pare ca sunt afisate simultan
      SwitchDigit(j);
      lightNumber(mystruct.digit[j]);
      delay(5);
    }
```

```
  }
}

void displayTime() {
// functie care afiseaza ora, ziua, anul in timp real pe LCD
  char buf[40];
  char buf1[40];
  time_t t = myRTC.get();
  sprintf(buf, "%.2d:%.2d:%.2d",hour(t), minute(t), second(t));
  sprintf(buf1, "%.2d %s %d", day(t),  monthShortStr(month(t)), year(t));
  lcd.setCursor(4, 0);
  lcd.print(buf);
  lcd.setCursor(3, 1);
  lcd.print(buf1);
}

bool countdown(int n, int del){
  for (int q = n; q > 0; q--){
    PrintNumber(q, del);
    if (digitalRead(button2) == LOW) {
      // daca este apasat butonul de reset in timpul numararii
      return false;
    }
  }
  PrintNumber(0, 0);

  for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2) {
    // durata fiecarei note
    divider = melody[thisNote + 1];
    noteDuration = (wholenote) / divider;
    // buzzerul canta doar 90% din durata notei, restul fiind un delay
    tone(buzzer, melody[thisNote], noteDuration * 0.9);
    delay(noteDuration);
    // opreste buzzerul inainte de urmatoarea nota
    noTone(buzzer);
  }

  // dupa ce buzzerul a terminat de cantat, LCD-ul afiseaza animatiile
  // menite sa anunte scurgerea timpului
  lcd.clear();
  for (int i = 0; i < 3; i++) {
    lcd.createChar(0, char00);
    lcd.setCursor(0, 0);
    lcd.write(0);
    lcd.createChar(1, char01);
    lcd.setCursor(1, 0);
    lcd.write(1);
    lcd.createChar(2, char02);
    lcd.setCursor(2, 0);
    lcd.write(2);
    lcd.createChar(3, char10);
```

```
    lcd.setCursor(0, 1);
    lcd.write(3);
    lcd.createChar(4, char11);
    lcd.setCursor(1, 1);
    lcd.write(4);
    lcd.createChar(5, char12);
    lcd.setCursor(2, 1);
    lcd.write(5);
    lcd.createChar(0, char00);
    lcd.setCursor(13, 0);
    lcd.write(0);
    lcd.createChar(1, char01);
    lcd.setCursor(14, 0);
    lcd.write(1);
    lcd.createChar(2, char02);
    lcd.setCursor(15, 0);
    lcd.write(2);
    lcd.createChar(3, char10);
    lcd.setCursor(13, 1);
    lcd.write(3);
    lcd.createChar(4, char11);
    lcd.setCursor(14, 1);
    lcd.write(4);
    lcd.createChar(5, char12);
    lcd.setCursor(15, 1);
    lcd.write(5);
    delay(1000);
    lcd.clear();
    lcd.createChar(0, char00);
    lcd.setCursor(0, 0);
    lcd.write(0);
    lcd.createChar(1, char01);
    lcd.setCursor(1, 0);
    lcd.write(1);
    lcd.createChar(2, char02);
    lcd.setCursor(2, 0);
    lcd.write(2);
    lcd.createChar(3, char10);
    lcd.setCursor(0, 1);
    lcd.write(3);
    lcd.createChar(4, animation);
    lcd.setCursor(1, 1);
    lcd.write(4);
    lcd.createChar(5, char12);
    lcd.setCursor(2, 1);
    lcd.write(5);
    lcd.setCursor(5, 0);
    lcd.print("Timpul");
    lcd.setCursor(3, 1);
    lcd.print("s-a scurs!");
    lcd.createChar(0, char00);
```

```
    lcd.setCursor(13, 0);
    lcd.write(0);
    lcd.createChar(1, char01);
    lcd.setCursor(14, 0);
    lcd.write(1);
    lcd.createChar(2, char02);
    lcd.setCursor(15, 0);
    lcd.write(2);
    lcd.createChar(3, char10);
    lcd.setCursor(13, 1);
    lcd.write(3);
    lcd.createChar(4, animation);
    lcd.setCursor(14, 1);
    lcd.write(4);
    lcd.createChar(5, char12);
    lcd.setCursor(15, 1);
    lcd.write(5);
    delay(1000);
    if (i != 2)
      lcd.clear();
  }

  delay(500);
  lcd.clear();
  displayTime();
  delay(2000);
  lcd.clear();

return true;
}

void reset() {
  int m, zeros, d, pressed3 = 0, pressed4 = 0;
  m = initial_time;
  struct digits mystruct;
  mystruct = todigits(initial_time);
  displayTime();
  delay(3000);
  lcd.clear();

  while (digitalRead(button1) == HIGH) {
    // daca nu este apasat butonul de start
    for (int j = 0; j < 4; j++) {
      SwitchDigit(j);
      lightNumber(mystruct.digit[j]);
      delay(2);
    }
    if (digitalRead(button3) == LOW) {
      // daca am apasat butonul 3
      if (pressed3 == 0 || pressed3 > 30) {
        if (initial_time > 0) {
```

```
          initial_time -= 1 ;
        }
        mystruct = todigits(initial_time);
      }
      pressed3 += 1;
    }
    else if (digitalRead(button4) == LOW) {
      // daca am apasat butonul 4
      if (pressed4 == 0 || pressed4 > 30) {
        if (initial_time < 9999) {
          initial_time += 1 ;
        }
        mystruct = todigits(initial_time);
      }
      pressed4 += 1;
    }
    if (digitalRead(button3) == HIGH) {
      // daca butonul 3 nu este apasat
      pressed3 = 0;
    }
    if (digitalRead(button4) == HIGH) {
      // daca butonul 4 nu este apasat
      pressed4 = 0;
    }
  }
}

void loop(){
  while (digitalRead(button2) == 1){
    displayTime();
    delay(200);
    lcd.clear();
  };

  reset();
  while (countdown(initial_time, 962) == false) {
    reset();
  }

  while (digitalRead(button2) == 1){
    displayTime();
    delay(200);
    lcd.clear();
  };
}
```

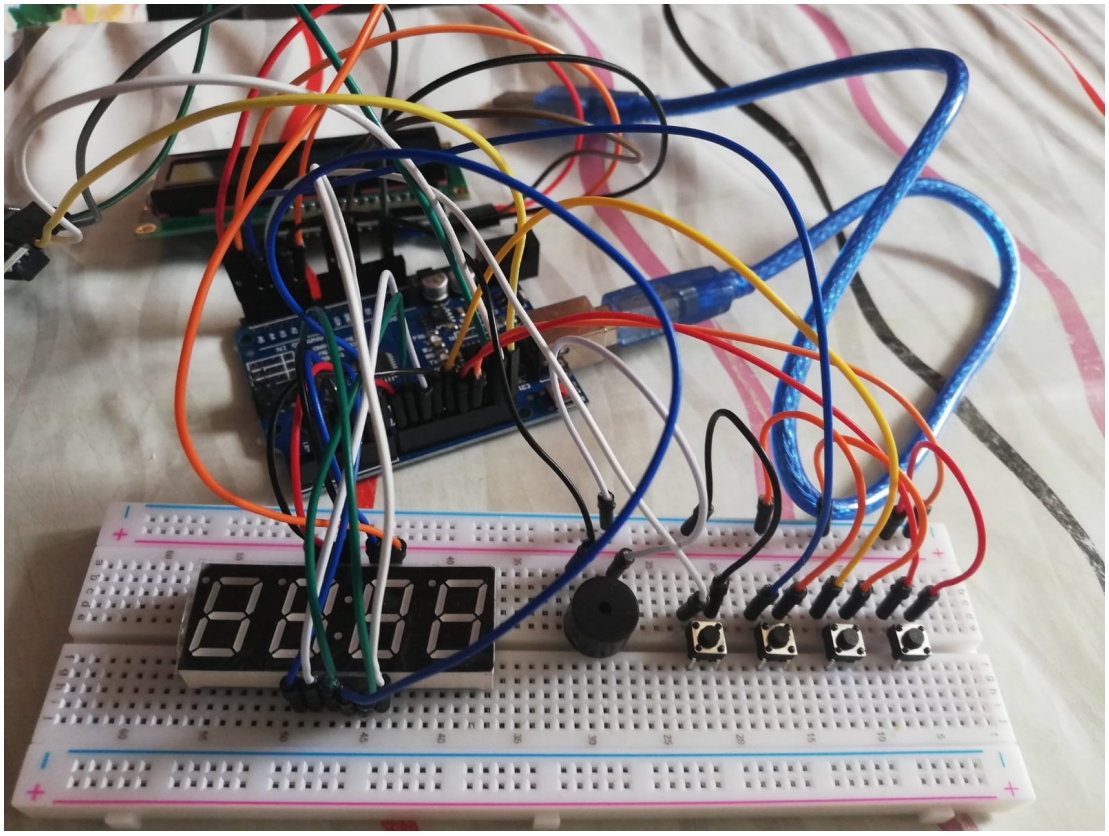
Rezultate Obţinute

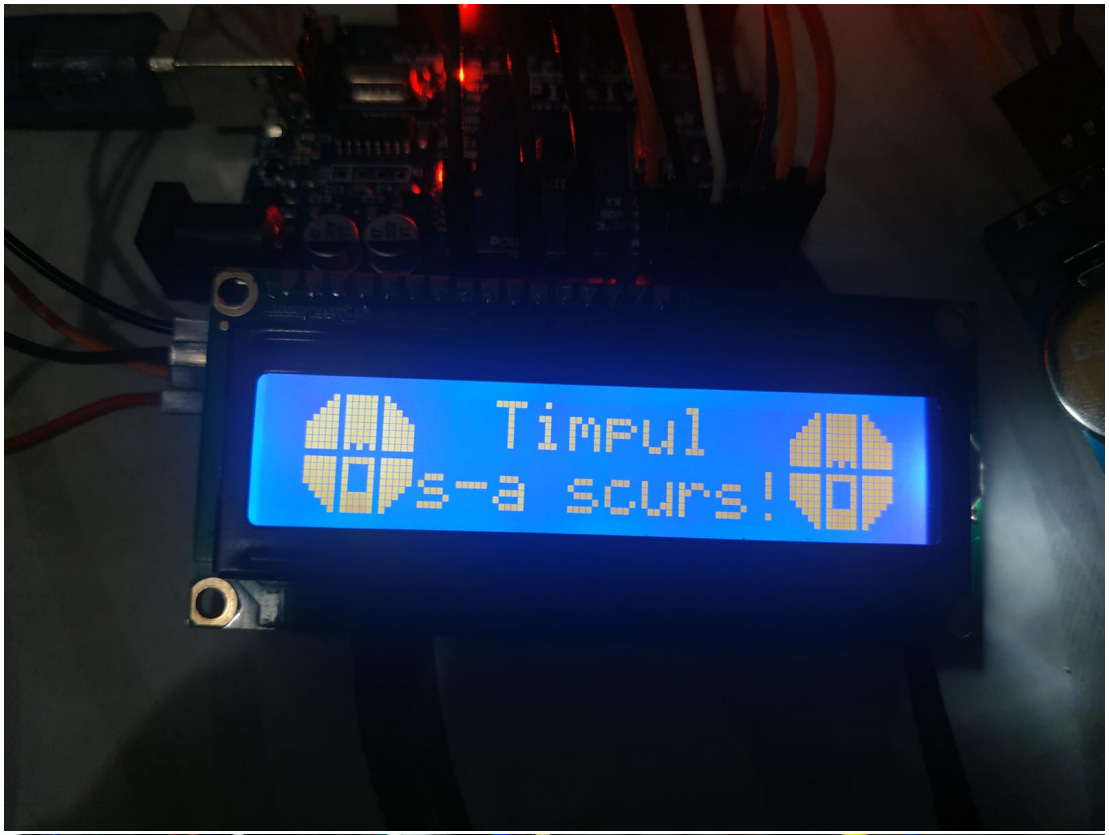
În urma implementării hardware și software, am obținut un **ceas** extrem de accurate care poate fi folosit în bucătărie mai ales datorită funcționalității sale de **timer**- adică ceasul meu poate, în plus, număra oricâte secunde până la aproape 3 ore și poate declanșa o alarmă la finalizarea timpului, alarmă care semnalizează bucătarului că timpul de coacere în cuptor s-a încheiat. Am încercat să ofer timerului o interfață cât mai friendly prin desenele animate de pe LCD. Funcțiile de ceas și timer nu sunt disponibile simultan, dar, în schimb, timerul poate fi resetat să măsoare orice număr de secunde la alegere, iar ceasul va menține mereu ora exactă, indiferent dacă deconectez plăcuța Arduino de la calculator pentru o vreme.

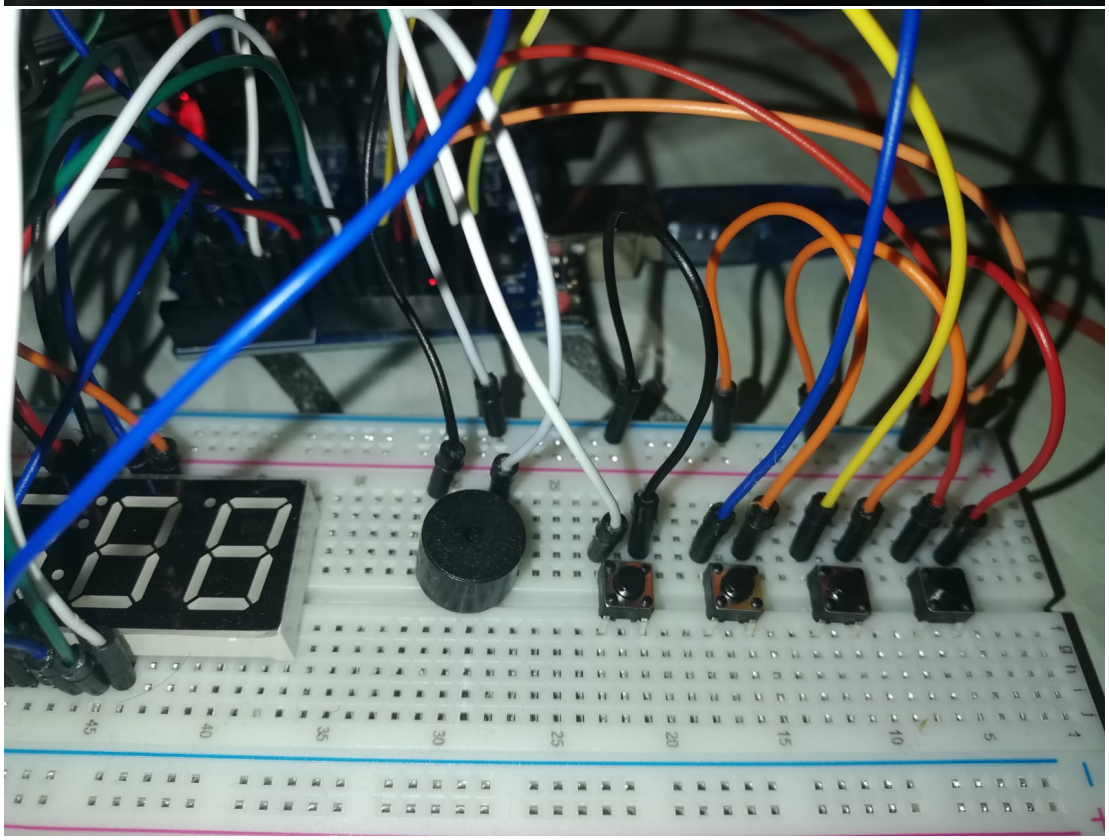
La **prima utilizare** a modulului RTC, acesta trebuie setat cu datele exacte referitoare la oră, minut, secundă, zi, lună și an. El trebuie setat folosind instrucțiunile:

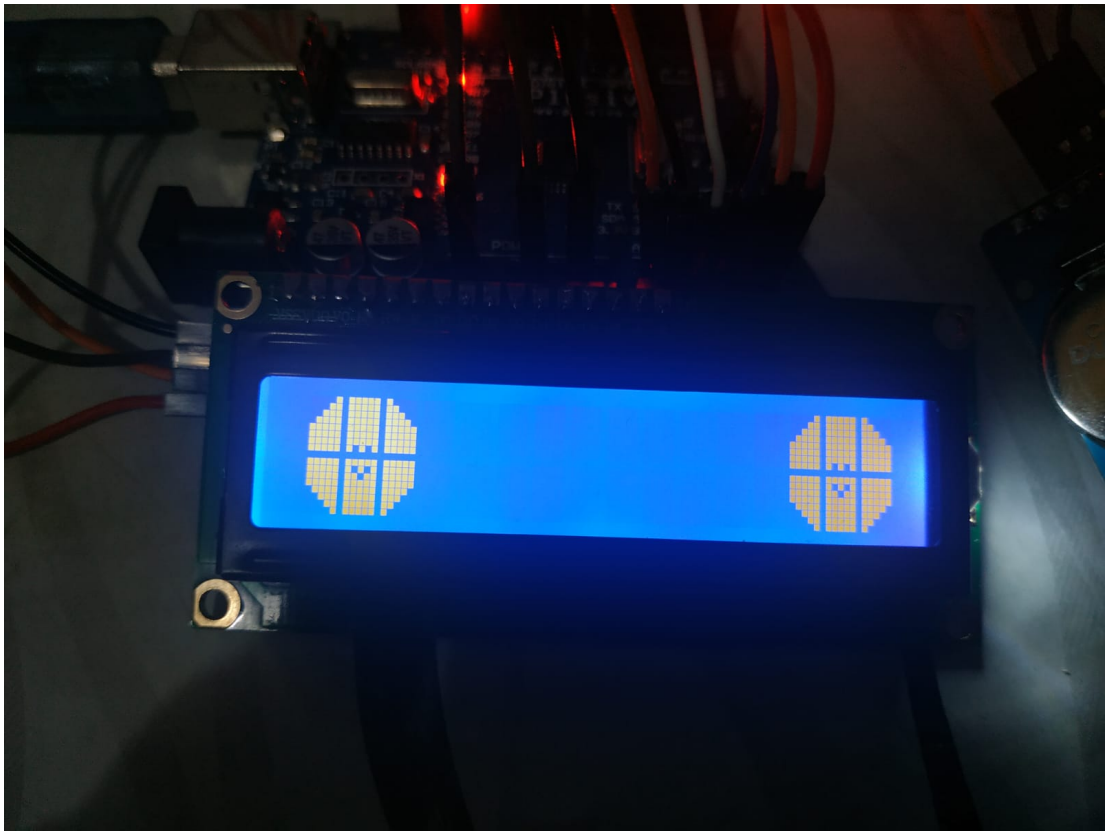
```
setTime(oră, minut, secundă, zi, lună, an);
myRTC.set(now());
```

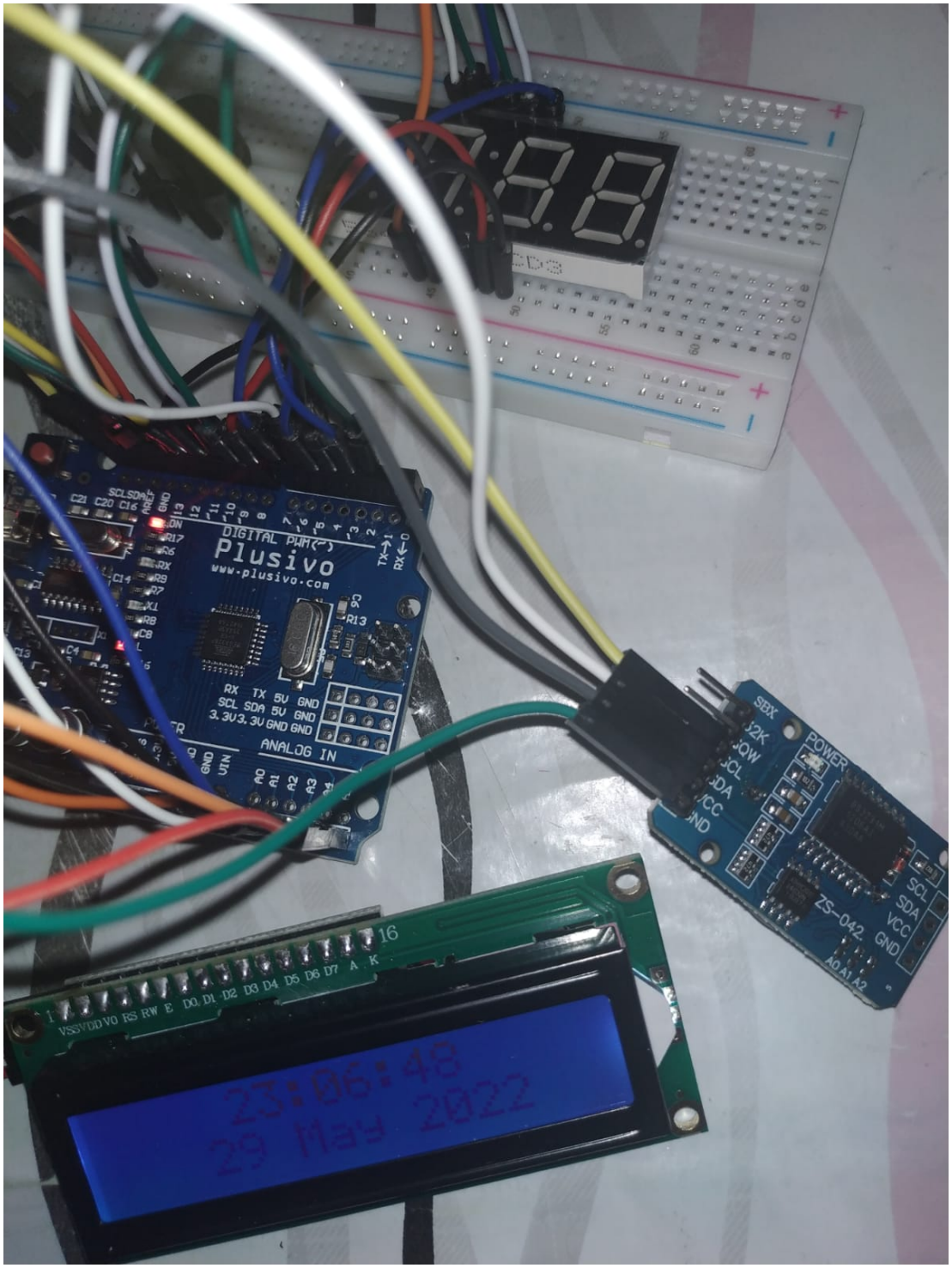
Mai jos se pot observa câteva fotografii cu proiectul rezultat:

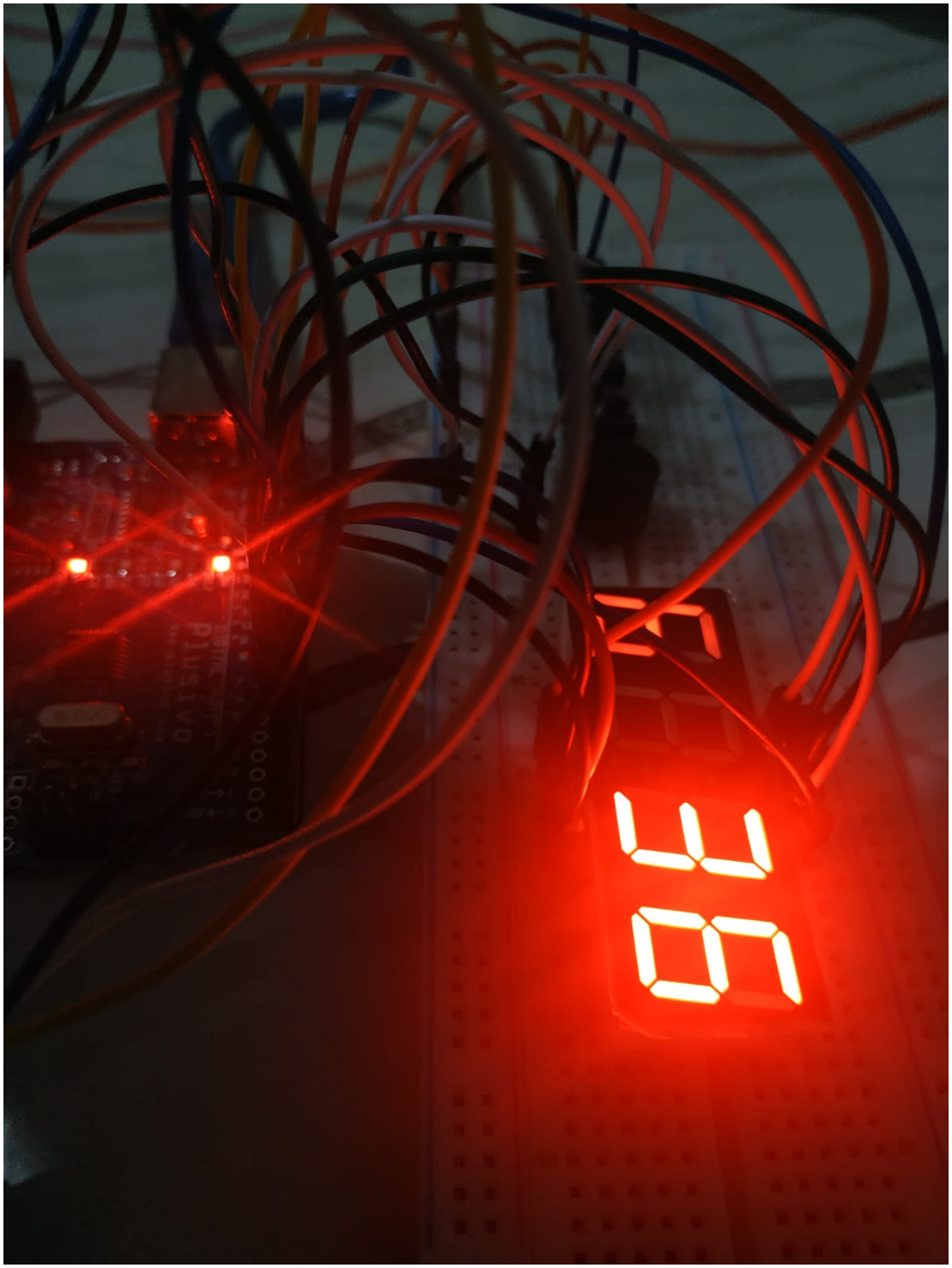


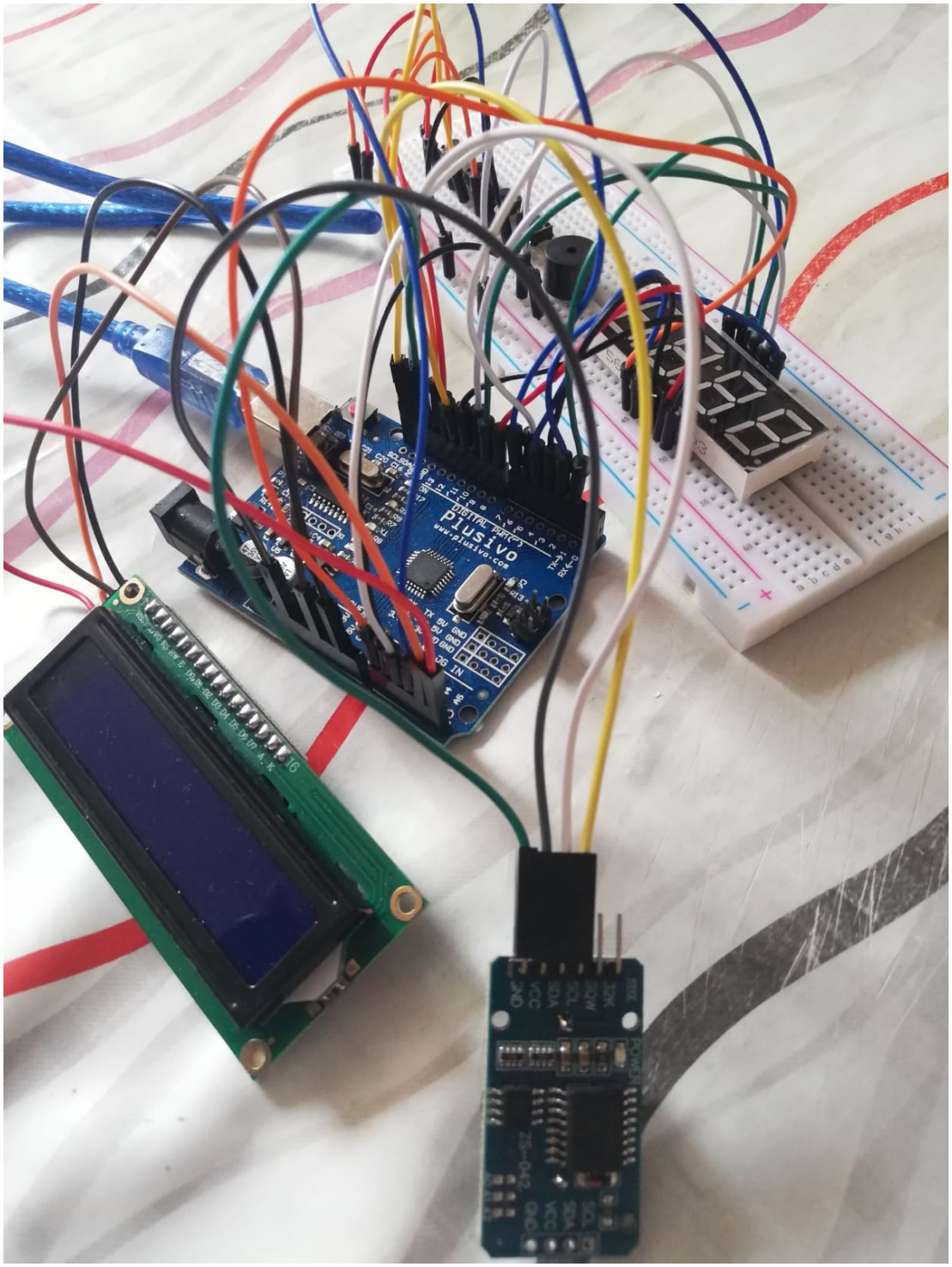

Timpul
s-a scurs!






Plusivo
www.plusivo.com
DIGITAL PWM(~)
ANALOG IN
RX TX 5V GND
SCL SDA 5V GND
3.3V 3.3V GND GND
POWER
SCL
SDA
VCC
GND
ZS-042
A0 A1 A2
16
VSS VDD VO RS RW E D0 D1 D2 D3 D4 D5 D6 D7 A K
23:06:48
29 May 2022





Concluzii

Acest proiect a fost prima mea experiență cu adevărat interesantă cu Arduino. Am învățat să lucrez cu piese și librării cu care nu am mai interacționat vreodată. De exemplu, am învățat să folosesc un 4-digit 7-segment display, un modul RTC sau să fac un buzzer să cânte melodii. Am înțeles că se pot construi obiecte chiar utile cu piese puține și ieftine. Pe viitor aș dori să îmbunătățesc proiectul făcând funcțiile de ceas și timer să fie disponibile simultan și adăugând un senzor de temperatură pentru a afișa și temperatura la LCD.

Download

Arhiva ce conține codul sursă: projecct.zip

PDF-ul ce conține pagina curentă: real_time_clock_si_timer_cu_alarma_pentru_bucatarie.pdf

Bibliografie/Resurse

- Lucru cu modulul RTC (Real Time Clock): https://www.youtube.com/watch?v=E6wkvTG2Ofs&t=91s
- Lucru cu LCD-ul cu I2C: https://www.youtube.com/watch?v=EAeuxjtkumM
- Displaying custom characters on LCD: https://www.youtube.com/watch?v=bjblQqfjdWM&t=304s, https://maxpromer.github.io/LCD-Character-Creator/
- Lucrul cu 4-digit 7-segment display- cum funcționează cei 12 pini: https://www.youtube.com/watch?v=iZI1GjCvIiw&t=490s
- Folosire bibliotecă pentru modulul RTC: https://github.com/JChristensen/DS3232RTC
- Folosire bibliotecă LiquidCrystal_I2C: https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library
- Note muzicale necesare pentru melodia de la Nokia cântată de Buzzer: https://github.com/robsoncouto/arduino-songs/blob/master/nokia/nokia.ino
- Exemplu de timer de la care m-am inspirat: https://www.youtube.com/watch?v=nLfRUNcb0ZQ

Export to PDF