Universitatea Politehnică Timișoara

Proiect Arduino

Programare orientate pe obiecte

Ceas & Alarmă digitală

Student:

Paducel Dan Alexandru , an2 , grupa 3

Timișoara 2025

Cerințe inițiale :

Utilizand un Arduino si un LCD 2x16 sa se implementeze un ceas si alarma digitala. La pornire ceasul porneste de la 00:00:00 (hh:mm:ss) si se incrementeaza cu o secunda. Folosind butonul select se selecteaza: modul normal care afiseaza ceasul pe randul 1 si alarma pe randul 2, modul de setare a ceasului si modul de setare al alarmei.

Butoanele:

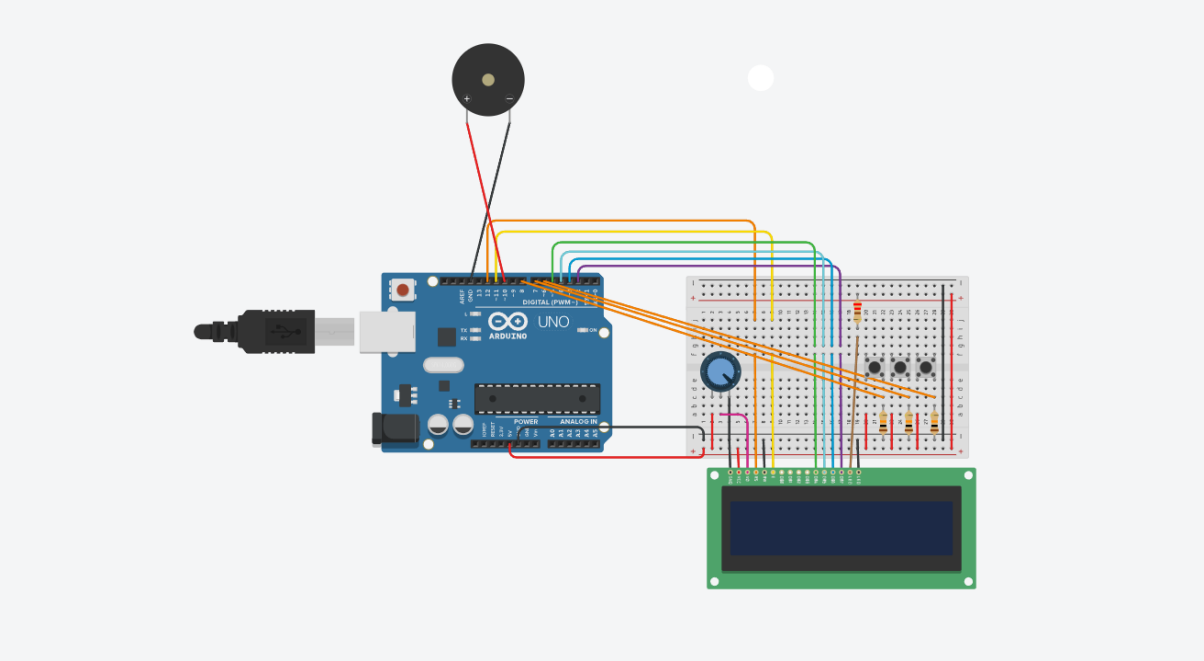
set: permite trecerea in modul de setare/afisare date

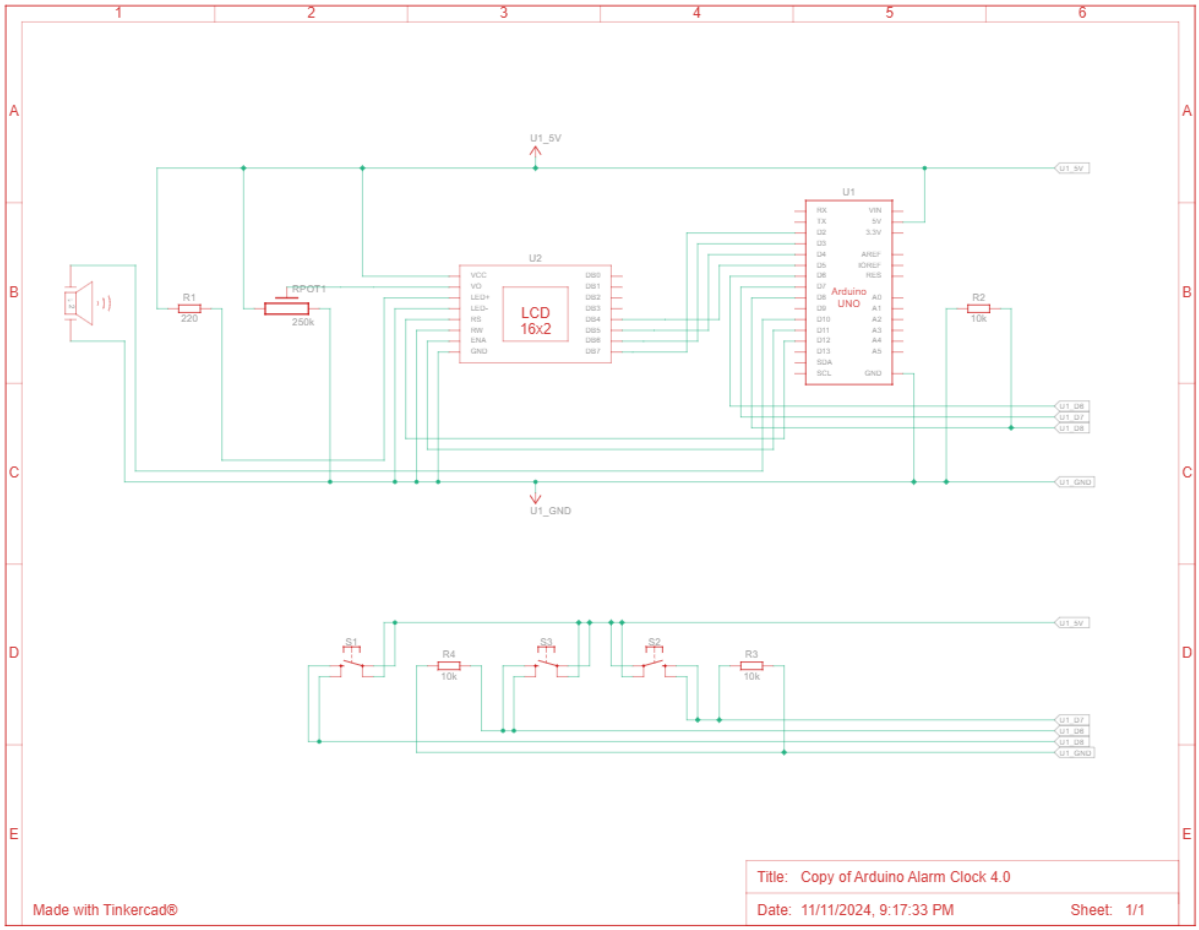
left/right: muta cursorul care indica valoare care v-a fi setata iar butoanele

up/down: seteaza valoarea corespunzatoare selectiei.

Pentru ceas setam ora si minutul iar pentru alarma ora, minutul si starea de active sau inactiv.

Schema hardware a sistemului





Aceasta schemă hardware arată un circuit cu un Arduino UNO (U1), un afișaj LCD 16x2 (U2), un difuzor și două butoane (S1 și S2). Arduino UNO controlează afișajul LCD prin intermediul mai multor pini de date și control. Difuzorul este conectat la un pin digital al Arduino-ului, iar butoanele sunt conectate la alți pini digitali pentru a permite interacțiunea utilizatorului.

Diagrama UML

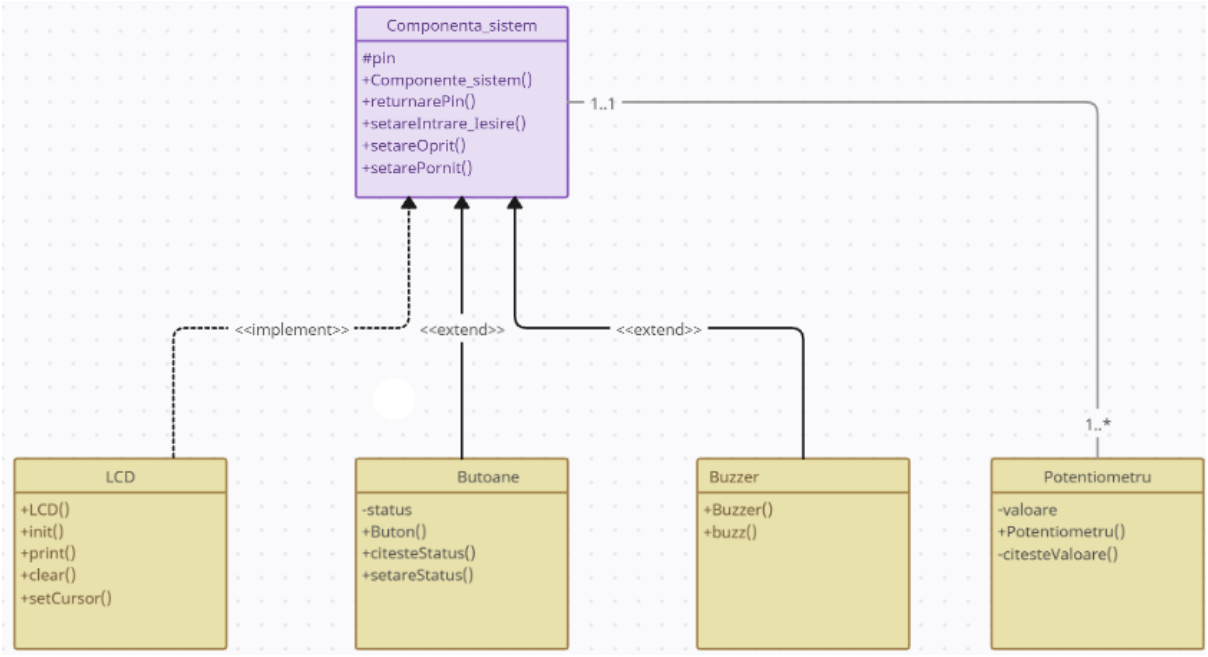


Diagrama UML prezentată reprezintă un sistem pentru un ceas de alarmă pe Arduino. Clasa principală este ComponenteSistem, care servește drept bază pentru clasele derivate: SenzorVibrații,

Difuzor, Led și Buton. Fiecare clasă derivată moștenește funcționalitățile de bază și adaugă metode specifice. SenzorVibrații gestionează citirea vibrațiilor, Difuzor emite sunete, Led controlează un indicator luminos, iar Buton gestionează intrările de la utilizator. Aceasta structură modulară facilitează gestionarea componentelor hardware într-un mod organizat și extensibil.

Testarea aplicației

Din momentul în care codul incepe să fie utilizat,ceasul alarmă este pornit și afișează ora 00:00:00. Ora poate fi setată prin intermedia codului la secvența (int h=0,int m=0,int s=0). Dacă nu dorim să setam ora din cod atunci la fie care data de pornire a ceasului el va începe la ora 0. Ajustarea alarmei este realizată prin cod la secvența (alarmset) unde in cod este ajustată deja la 10s. După trecerea celor 10s, buzzerul emite un sunet care arată ca alarma sună.

Concluzii

- În urma acestui proiect mi-am aprodundat conoștințele teoretice a limbajului C++.
- Am învațat să utilizez cunoștințele din C++ în domeniul practic al microcontrorelor.
- A existat o dificultate în gasirea și folosirea unui simulator, mai ales la partea montajului.
- Difucultăți au existat și pe partea de cod. De multe ori pâna la rezultatul final, codul avea errori care conduceau la pornirea greșită a ceasului.

Codul sursă al programului

```
/*
 PENTRU PORNIREA ALARMEI SE TINE APASAT PE BUTOANE
 PRIMUL BUTON ESTE IN ORE
 AL DOILEA BUTON ESTE IN MINUTE
 AL TREILEA BUTON ESTE IN SECUNDE

 AJUSTAREA SE FACE MANUAL PRIN CODUL DE MAI JOS
 WITH h = #
 WITH m = #
```

WITH s = #

*/

```
#include <LiquidCrystal.h>

int h = 0, m = 0, s = 0;
int alarmh = 0, alarmm = 0, alarms = 0;
int alarmseth = 0, alarmsetm = 0, alarmsets =
10;
bool alarmTriggered = false;
int inPin = 8, inPin2 = 7, inPin3 = 6;
int val, val2, val3;
bool prevVal = LOW, prevVal2 = LOW, prevVal3 =
LOW;
unsigned long previousMillis = 0;
unsigned long alarmStartMillis = 0;
unsigned long debounceMillis1 = 0,
debounceMillis2 = 0, debounceMillis3 = 0;
const long interval = 1000;       // Interval for one-
second increments
const long debounceDelay = 200;   // Debounce
time for button presses
```

```
const long alarmDuration = 5000;  // Alarm
duration in milliseconds
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
  pinMode(10, OUTPUT);
  pinMode(inPin, INPUT);
  pinMode(inPin2, INPUT);
  pinMode(inPin3, INPUT);
}

void loop() {
  unsigned long currentMillis = millis();

  // Update time every second
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    incrementTime();
    checkAlarm();
  }

  // Check button inputs for setting the alarm
  handleButtons();
```

```
  // Update display
  displayTimeAndAlarm();
}

void incrementTime() {
  s++;
  if (s == 60) {
    s = 0;
    m++;
    if (m == 60) {
      m = 0;
      h++;
      if (h == 24) h = 0;
    }
  }
}

void checkAlarm() {
  // Check if current time matches alarm time
and trigger alarm if not already triggered
  if (alarmseth == h && alarmsetm == m &&
alarmsets == s && !alarmTriggered) {
    alarmTriggered = true;
```

```
    alarmStartMillis = millis(); // Start alarm
duration timer
    tone(10, 2000); // Start alarm tone
  }

  // Turn off alarm after a certain duration
  if (alarmTriggered && (millis() - alarmStartMillis
>= alarmDuration)) {
    noTone(10); // Stop the alarm tone
    alarmTriggered = false; // Reset alarm trigger
  }
}

void handleButtons() {
  unsigned long currentMillis = millis();

  // Check first button (for hours)
  val = digitalRead(inPin);
  if (val == HIGH && prevVal == LOW &&
(currentMillis - debounceMillis1 >=
debounceDelay)) {
    alarmseth = (alarmseth < 23) ? alarmseth + 1 :
0;
    debounceMillis1 = currentMillis;
```

```
  }
  prevVal = val;

  // Check second button (for minutes)
  val2 = digitalRead(inPin2);
  if (val2 == HIGH && prevVal2 == LOW &&
(currentMillis - debounceMillis2 >=
debounceDelay)) {
    alarmsetm = (alarmsetm < 59) ? alarmsetm +
1 : 0;
    debounceMillis2 = currentMillis;
  }
  prevVal2 = val2;

  // Check third button (for seconds)
  val3 = digitalRead(inPin3);
  if (val3 == HIGH && prevVal3 == LOW &&
(currentMillis - debounceMillis3 >=
debounceDelay)) {
    alarmsets = (alarmsets < 59) ? alarmsets + 1 :
0;
    debounceMillis3 = currentMillis;
  }
  prevVal3 = val3;
```

```
}

void displayTimeAndAlarm() {
  // Display Alarm
  lcd.setCursor(0, 0);
  lcd.print("Alarm:");
  lcd.setCursor(7, 0);
  lcd.print((alarmseth < 10 ? "0" : "") +
String(alarmseth));
  lcd.print(":");
  lcd.print((alarmsetm < 10 ? "0" : "") +
String(alarmsetm));
  lcd.print(":");
  lcd.print((alarmsets < 10 ? "0" : "") +
String(alarmsets));

  // Display Time
  lcd.setCursor(0, 1);
  lcd.print("Time: ");
  lcd.setCursor(6, 1);
  lcd.print((h < 10 ? "0" : "") + String(h));
  lcd.print(":");
  lcd.print((m < 10 ? "0" : "") + String(m));
  lcd.print(":");
```

```
  lcd.print((s < 10 ? "0" : "") + String(s));
}
```