Smart Desk System - Bolontoc Daniel

Daniel Bolontoc Website

Student: *Daniel BOLONTOC*
Grupă: **335CB**
An universitar: **2022-2023**

Introducere

Sistemul inteligent pentru birou bazat pe Arduino este o soluție avansată ce monitorizează și afișează informații vitale într-o manieră eficientă și practică. Echipat cu senzori de temperatură și umiditate, acest sistem poate măsura și raporta în timp real aceste date în biroul dumneavoastră.

Cu ajutorul funcționalității sale inteligente, sistemul este capabil să ofere o imagine precisă și actualizată a condițiilor ambientale. Astfel, puteți ști întotdeauna temperatura și umiditatea exactă din birou, ceea ce poate contribui la confortul și productivitatea dumneavoastră.

Pe lângă aceasta, sistemul inteligent pentru birou este dotat și cu un afișaj integrat, care vă permite să vizualizați, în timp real, valorile temperaturii și umidității, într-un format ușor de înțeles. De asemenea, acesta afișează și ora actuală, astfel încât să puteți monitoriza în același loc informații esențiale.

Folosind tehnologia Arduino, sistemul este flexibil și poate fi personalizat în funcție de necesitățile dumneavoastră specifice. Aceasta înseamnă că puteți adăuga funcționalități suplimentare sau conectivitate cu alte dispozitive, pentru a optimiza performanța și utilitatea sistemului.

Cu un sistem inteligent pentru birou ce masoară temperatura, umiditatea și afișează ora facut cu Arduino, beneficiați de un instrument util și inteligent pentru a vă menține un mediu de lucru confortabil și productiv.

Descriere generală

Sistemul inteligent pentru birou este o soluție tehnologică avansată care utilizează diverse senzori și componente electronice pentru a monitoriza și gestiona diferite aspecte ale mediului de lucru dintr-un birou. Aceste sisteme sunt proiectate pentru a crea un mediu de lucru optim și eficient, îmbunătățind confortul, productivitatea și bunăstarea angajaților.

- Măsurarea temperaturii: Sistemul este echipat cu un senzor de temperatură precis, care monitorizează temperatura din birou. Această funcționalitate vă permite să cunoașteți întotdeauna temperatura ambientală în timp real.
- Monitorizarea umidității: Un senzor de umiditate integrat în sistem permite măsurarea nivelului de

umiditate din birou. Aceasta vă ajută să aveți un control asupra nivelului de umiditate și să ajustați condițiile ambientale pentru a asigura confortul optim.

- Afișarea orei: Sistemul include un afișaj digital care vă arată ora curentă. Această caracteristică utilă vă permite să vă gestionați timpul eficient și să respectați termenele limită.
- Conectivitate Arduino: Sistemul este construit pe platforma Arduino, oferind flexibilitate și opțiuni de personalizare extinse. Puteți adăuga sau modifica funcționalități suplimentare în funcție de cerințele și preferințele dumneavoastră specifice.
- Ușor de utilizat: Interfața sistemului este intuitivă și ușor de utilizat. Puteți accesa și interpreta rapid informațiile referitoare la temperatură, umiditate și oră într-un mod simplu și convenabil.
- Monitorizare în timp real: Sistemul actualizează constant datele și le afișează în timp real. Acest lucru vă permite să monitorizați și să reacționați rapid la schimbările condițiilor de mediu în birou.
- Contribuție la confort și productivitate: Prin furnizarea informațiilor exacte despre temperatură și umiditate, sistemul vă ajută să creați un mediu de lucru confortabil și sănătos. Aceasta poate contribui la îmbunătățirea productivității și a bunăstării angajaților.
- Utilitate versatilă: Sistemul de monitorizare a temperaturii, umidității și orei poate fi folosit într-o gamă largă de medii de lucru, cum ar fi birouri, săli de ședințe sau spații comerciale.

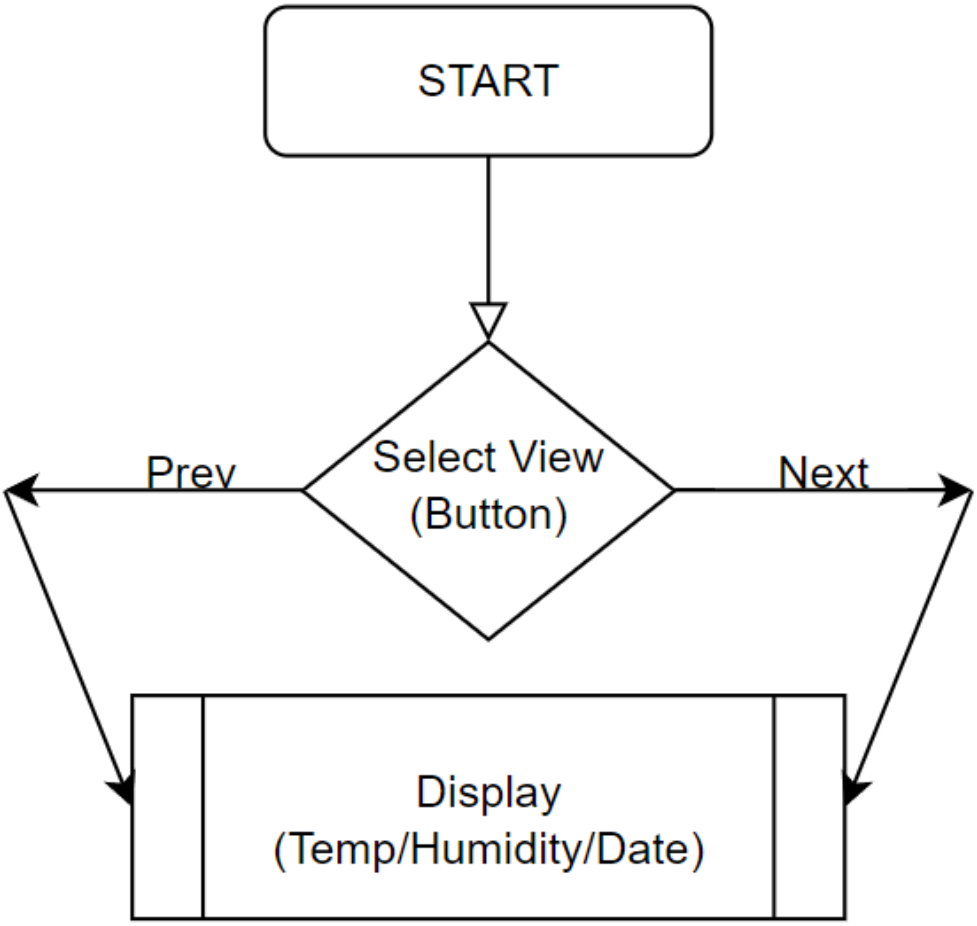
Cu aceste funcționalități avansate și caracteristici versatile, sistemul de monitorizare a temperaturii, umidității și orei aduce un nivel suplimentar de control și confort în biroul dumneavoastră.

Pentru cine poate fi util acest proiect?
Sistemul inteligent pentru birou este util atât pentru angajați, oferindu-le un mediu de lucru confortabil și eficient, cât și pentru companie, optimizând productivitatea și reducând costurile asociate cu gestionarea mediului de lucru.

Organigrama si Mod de functionare

Funcționalitatea sistemului:

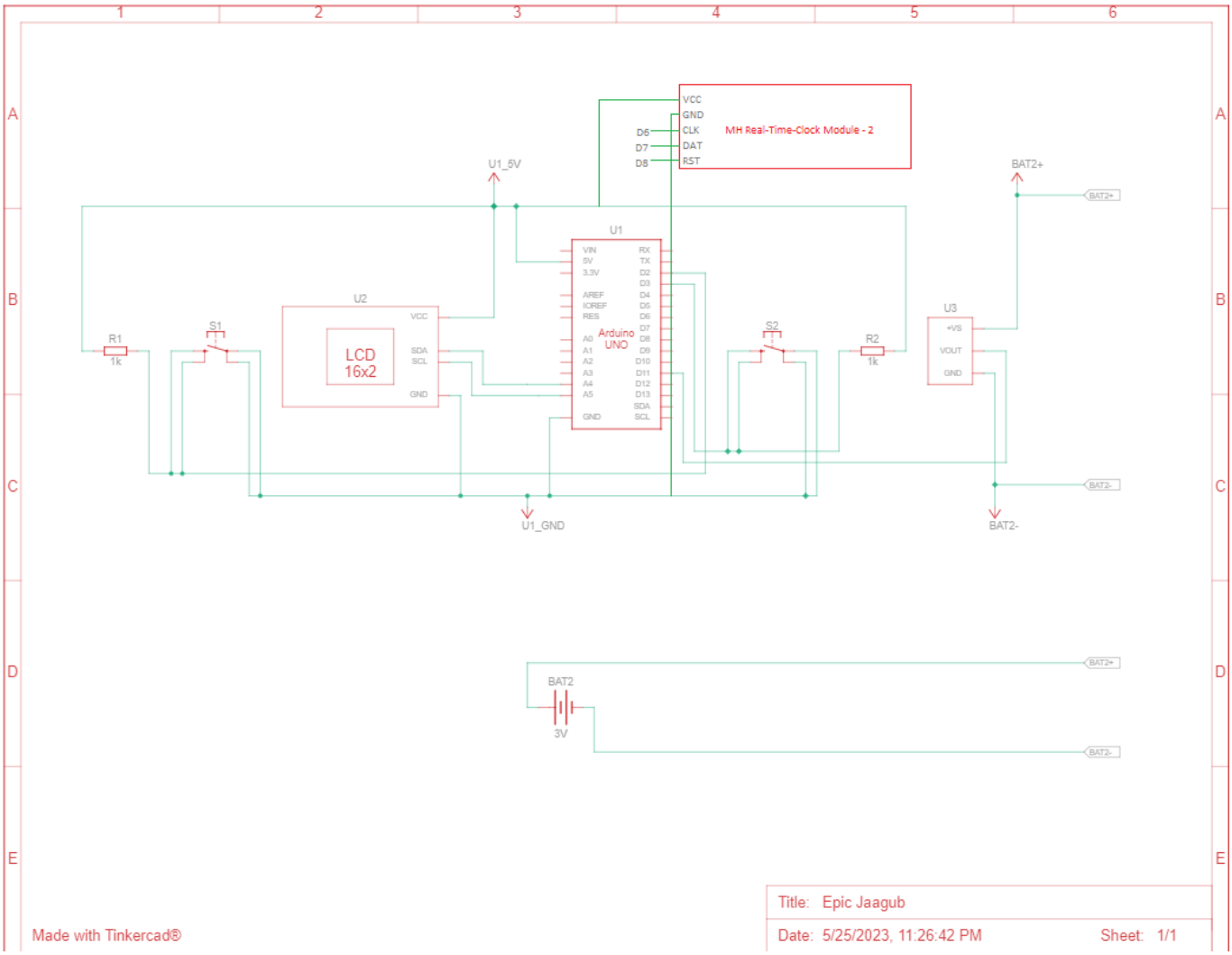
- Afișarea datelor: Valorile de temperatură și umiditate, precum și data și ora curente, sunt afișate pe display. De exemplu, displayul poate arăta “Temperatura: 25°C, Umiditate: 60%, Ora: 14:10:00.
- Navigarea prin informații: Utilizatorul poate folosi butonul “Next” pentru a trece la următoarea informație disponibilă pe display. De exemplu, următoarea informație poate fi “Temperatura maximă înregistrată: 30°C” sau “Umiditate minimă înregistrată: 40%”. Apăsarea butonului “Prev” permite revenirea la informația anterioară.
- Actualizarea datelor: Sistemul actualizează în mod continuu datele de temperatură și umiditate, afișând cele mai recente informații pe display. Aceasta poate fi utilă, de exemplu, pentru a monitoriza schimbările de temperatură și umiditate în timp real.



Componente

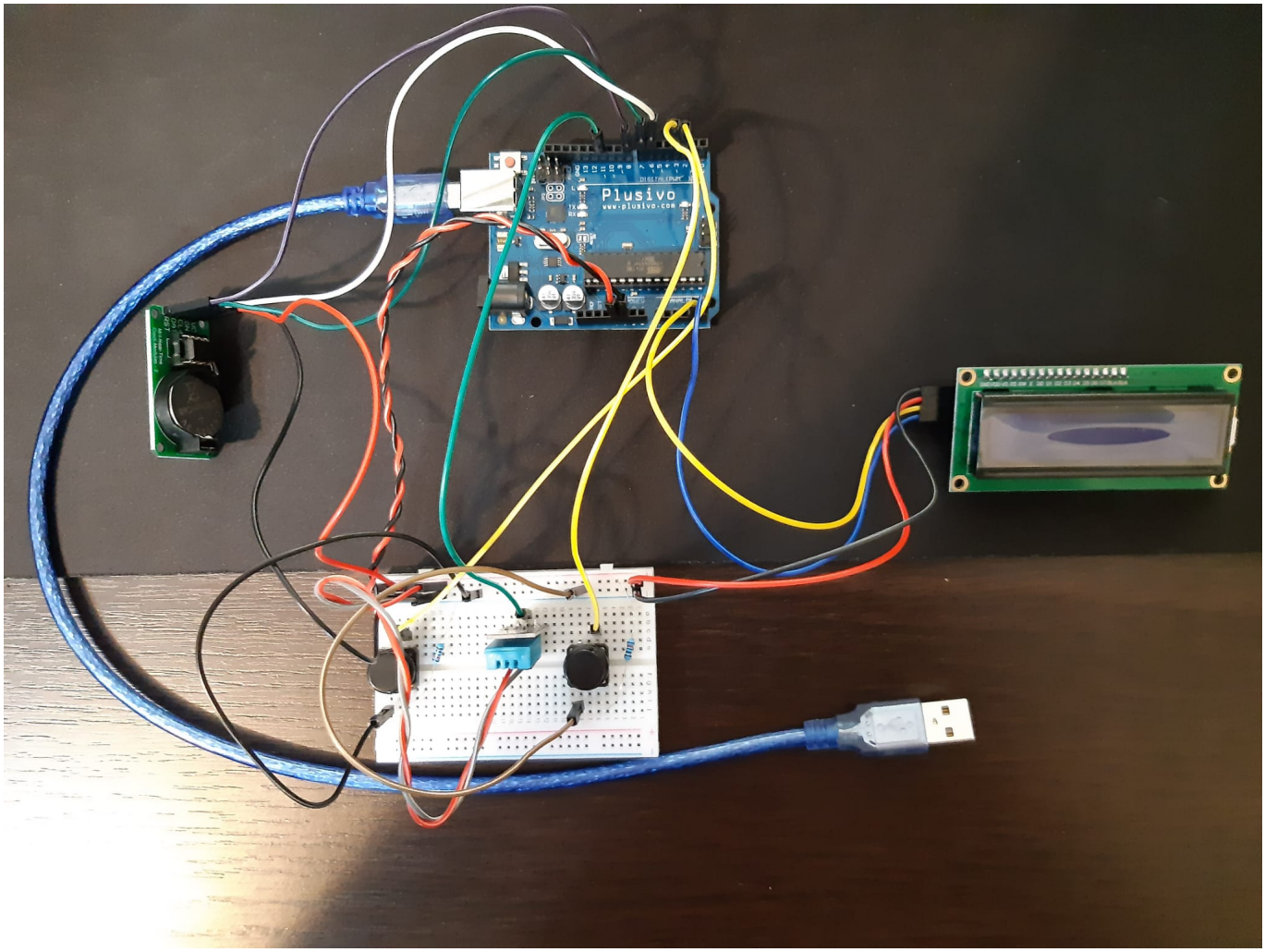
- Arduino UNO,
- LCD Display + (Modul I2C) + Potentiometru integrat,
- Senzor Temperatura si Umiditate: DHT11,
- Ceas in timp real: MH Real-Time-Clock Module - 2,
- Butane (PUSH),
- Rezistente 10K,
- Fire,
- Baterie: CR2025
- Breadboard.

Schema electrica a circuitului



Click pe imagine daca nu se vede clar!

Circuit fizic implementat



Software Design

Source Code - GitHub

Mediu de dezvoltare: Arduino IDE

- Aceste linii de cod includ librăriile necesare pentru a utiliza funcțiile pentru afișajul LCD I2C, senzorul DHT, comunicația prin I2C și modulul de ceas RTC (Real-Time Clock):

```
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
#include <Wire.h>
#include <virtuabotixRTC.h>
```

- Aici se realizează inițializarea obiectelor pentru comunicarea cu afișajul LCD, senzorul DHT și modulul RTC. Parametrii 0x27, 16 și 2 pentru obiectul lcd indică adresa I2C a afișajului LCD, numărul de coloane și numărul de linii al afișajului. Parametrii 11 și DHT11 pentru obiectul dht indică pinul digital la care este conectat senzorul DHT și tipul acestuia (DHT11). Parametrii 6, 7 și 8 pentru obiectul myRTC indică pinii digitali la care este conectat modulul RTC:

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
DHT dht(11, DHT11);
```

```
virtuabotixRTC myRTC(6, 7, 8);
```

- Aici este definit un nou simbol personalizat pentru afișajul LCD. Este un simbol de grad utilizat pentru a afișa temperatura în grade Celsius:

```
byte degree_symbol[8] = {
  0b00111,
  0b00101,
  0b00111,
  0b00000,
  0b00000,
  0b00000,
  0b00000,
  0b00000
};
```

- Aceste linii de cod declară și inițializează variabilele pentru pini și modul de afișare. buttonPin și buttonPin2 indică pinii digitali la care sunt conectate butoanele. displayMode este o variabilă care stochează modul curent de afișare (0 pentru temperatură, 1 pentru umiditate și 2 pentru timp):

```
const int buttonPin = 2;
const int buttonPin2 = 3;
int displayMode = 0;
```

- Aceste linii de cod declară și inițializează variabilele pentru starea butoanelor și debounce (evitarea fluctuațiilor induse de zgomot). buttonState și lastButtonState memorează starea actuală și ultima starea a butonului principal. buttonState2 și lastButtonState2 memorează starea actuală și ultima starea a butonului secundar. lastDebounceTime este folosit pentru a memora momentul ultimei modificări a stării butonului.

```
int buttonState = HIGH;
int lastButtonState = HIGH;
int buttonState2 = HIGH;
int lastButtonState2 = HIGH;
unsigned long lastDebounceTime = 0;
```

- Aceasta este funcția setup() care se execută o singură dată la pornirea sistemului. În această funcție, se inițializează afișajul LCD, se activează iluminarea de fundal, se creează simbolul personalizat de grad, se setează cursorul afișajului la poziția (0, 0) și se afișează un mesaj de început. De asemenea, se configurează pinii butoanelor ca intrări cu rezistențe pull-up interne și se inițializează senzorul DHT:

```
void setup() {
  lcd.init();
  lcd.backlight();
  lcd.createChar(1, degree_symbol);
  lcd.setCursor(0, 0);
  lcd.print(" Smart Desk Sys");

  pinMode(buttonPin, INPUT_PULLUP);
  pinMode(buttonPin2, INPUT_PULLUP);
```

```
  dht.begin();
}
```

- Aceasta este funcția loop() care rulează într-o buclă continuă. În această buclă, se citesc stările butoanelor și se compară cu stările anterioare pentru a detecta apăsările butoanelor. În funcție de starea butoanelor, se modifică modul de afișare.

Dacă starea butonului principal (buttonPin) se modifică, se actualizează buttonState și, dacă noul starea este LOW (butonul a fost apăsat), displayMode este incrementat cu 1 folosind operatorul modulo 3 pentru a trece la următorul mod de afișare (temperatură, umiditate, timp).

Dacă starea butonului secundar (buttonPin2) se modifică, se actualizează buttonState2 și, dacă noul starea este LOW (butonul a fost apăsat), displayMode este incrementat cu 2 și apoi este aplicat operatorul modulo 3 pentru a trece la modul de afișare anterior (rotație în direcție inversă).

În funcție de valoarea displayMode, se afișează informațiile corespunzătoare pe afișajul LCD. Dacă displayMode este 0, se citește temperatura cu ajutorul senzorului DHT și se afișează pe afișaj împreună cu simbolul personalizat de grad. Dacă displayMode este 1, se citește umiditatea și se afișează pe afișaj. Dacă displayMode este 2, se actualizează timpul folosind modulul RTC și se afișează pe afișaj.

La sfârșitul buclei, se actualizează lastButtonState pentru a memora starea butonului principal pentru următoarea iterație a buclei:

```
void loop() {
  int reading = digitalRead(buttonPin);
  int reading2 = digitalRead(buttonPin2);

  if (reading != buttonState) {
    buttonState = reading;
    if (buttonState == LOW) {
      displayMode = (displayMode + 1) % 3;  // Toggle display mode between
temperature, humidity, and time
    }
  }

  if (reading2 != buttonState2) {
    buttonState2 = reading2;
    if (buttonState2 == LOW) {
      displayMode = (displayMode +2) % 3;  // Toggle display mode in the
opposite direction
    }
  }

  if (displayMode == 0) {
    float temperature = dht.readTemperature();
    lcd.setCursor(0, 1);
    lcd.print("Temp: ");
    lcd.print(temperature);
    lcd.write(1);
    lcd.print("C");
    lcd.print("    ");
```

```
  } else if (displayMode == 1) {
    float humidity = dht.readHumidity();
    lcd.setCursor(0, 1);
    lcd.print("Humidity: ");
    lcd.print(humidity);
    lcd.print("%");
  } else if (displayMode == 2) {
    myRTC.updateTime();
    lcd.setCursor(0, 1);
    lcd.print("Time: ");
    if (myRTC.hours < 10) {
      lcd.print("0");
    }
    lcd.print(myRTC.hours);
    lcd.print(":");
    if (myRTC.minutes < 10) {
      lcd.print("0");
    }
    lcd.print(myRTC.minutes);
    lcd.print(":");
    if (myRTC.seconds < 10) {
      lcd.print("0");
    }
    lcd.print(myRTC.seconds);
    lcd.print("   ");
  }
  lastButtonState = reading;
}
```

Concluzii

- Efectiv schema fizica e plina de fire.
- Display-ul LCD trebuie setat manual cu “surubelnita” de grosime nanometrica.
- Codul in Arduino desi are sens si probabil functioneaza intr-o simulare pe calculator, cand il incarci pe placuta sufletul...

Sunt AS pe lipit componente cu pistolul!

Download:

Export to PDF