Sleeping Monitor

Introducere

Un Monitor de Somn este exact ceea ce sugerează și numele lui. Este un dispozitiv care urmărește experiența ta de somn și mediul în care dormi, folosind o serie de factori precum temperatura, umiditatea și nivelul de zgomot din timpul somnului(indiferent că este vorba de foșnete, sunete aleatorii etc.). Toate datele sunt înregistrate pe un SD card și ulterior sunt procesate pentru a genera o diagramă a experienței tale de somn. Scopul acestui dispozitiv este să se asigure că, prin monitorizarea zgomotului, persoana are un somn profund, calm și lipsit de stres.

Descriere generală

Proiectul implică utilizarea microcontroller-ului Arduino Uno și a senzorilor (deși schema bloc nu include un SD card sau un senzor de zgomot, deoarece Tinkercad nu dispune de ele). Pe placa de conexiuni avem, de asemenea, 3 butoane, fiecare cu funcționalitatea sa. Un buton pentru începerea monitorizării, unul pentru oprirea monitorizării și unul pentru Opțiuni. De asemenea, avem și 3 LED-uri care vor funcționa ca indicatori vizuali pentru a denota nivelul zgomotului detectat. Pe LCD va fi afișat timpul monitorizării, iar la apăsarea butonului de opțiuni diferite statistici .

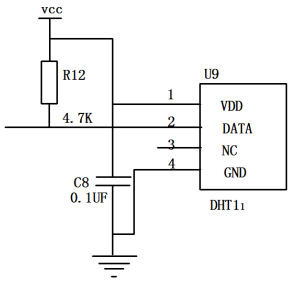
Cele 3 LED-uri(care trebuie conectate fiecare la un pin diferit, nu ca pe schema de mai sus) se vor aprinde în funcție de nivelul de zgomot perceput(în funcție de un anumit threshold). Ideea codului este un FSM(Finite State Machine). La bază este un meniu de opțiuni care ne lasă să alegem ce dorim să monitorizăm(temperatura, umiditatea, fie ambele deodată).

Hardware Design

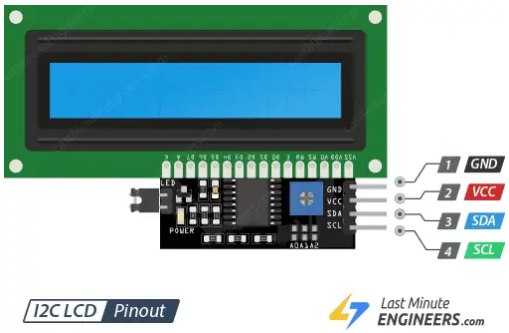
NOTE: Buzzer-ul si modulul 2 digit sunt funcționale, dar trebuie solder-uite.

Listă de piese

1. Senzor de temperatură DHT11



1. LED-uri
2. butoane
3. microcontroller Arduino Uno
4. buzzer
5. rezistențe
6. Ecran LCD I2C



1. Modul senzor sunet cu microfon KY-038
2. Modul 2 digit 74HC595

Software Design

Codul a fost dezvoltat in Arduino IDE. În cadrul acestuia, m-am folosit de 2 biblioteci thirdy-party.

1. LiquidCrystal_I2C.h (pentru gestionarea ecranului LCD)
2. DHT11.h (pentru gestionarea senzorului de temperatură și umiditate)
3. ShiftRegister74HC595.h (pentru gestionarea modului 2 led digits) NOTE! Acest modul a avut probleme, unele functii a trebuit sa le rescriu cand am importat manual zip-ul cu libraria si in continuare, gruparea de led-uri din stanga, nu merge intotdeauna as intended.

De asemenea, deoarece citirea temperaturii și a umidității este destul de costisitoare d.p.d.v. al ciclilor de ceas, vom face această acțiune la un interval specific de timp(o dată la 10 secunde).

```
unsigned char buttonState = digitalRead(SETTINGS_BUTTON);

  switch(STATE){
    case IDLE: {
      unsigned long currMillis = millis();
```

```
      // Verificam daca au trecut 10 secunde
      if (currMillis - prevMillis >= interval) {
        prevMillis = currMillis;
        dht.readTemperatureHumidity(t, h);
      }

      // Pressed
      if(buttonState == 0){
        STATE = SETTINGS_STATE;
        lcd.clear();
        delay(250);
      }

      switch(selectedSetting){
        // Temp
        case 0: {
          temperature_led_handle(t);
          displaySetting("Temp", t);
          break;
        }
        // Humidity
        case 1: {
          TURNALLOFF();
          displaySetting("Humidity", h);
          break;
        // Both
        }case 2: {
          TURNALLOFF();
          displaySetting("Tmp", t, "Hum", h);
          break;
        }
        default: break;
      }

      break;
    }
    case SETTINGS_STATE: {
      lcd.setCursor(0, 0);
      lcd.print("Settings Menu");
      lcd.setCursor(0, 1);

      unsigned char nextButtonState = digitalRead(NEXT_BUTTON);
      unsigned char backButtonState = digitalRead(BACK_BUTTON);

      if(!nextButtonState){
        idx = (idx + 1) % SETTINGS_ARR_SIZE;
        currentOption = &settingsArr[idx];
        delay(250);
        lcd.clear();
        Serial.println("apasat next");
      }else if(!backButtonState){
```

```
        idx = (idx <= 0) ? SETTINGS_ARR_SIZE - 1 : (idx - 1) % 3;
        currentOption = &settingsArr[idx];
        delay(250);
        lcd.clear();
        Serial.println("apasat next");
      }

      if(buttonState == 0){
        selectedSetting = idx;
        lcd.setCursor(0, 0);
        lcd.clear();
        lcd.print("Monitoring...");
        STATE = IDLE;
        // Putin delay, deoarece va iesi din acest state si va intra inapoi
instant tot in SETTINGS_STATE
        delay(250);
      }

      lcd.print((char*)*currentOption);

      break;
    }
    default: {
      break;
    }
  }
```

NEW! MULTIPLE LANGUAGES!! Limba poate fi aleasa din meniul de setari.

```
const unsigned char settingsArrEnglish[][17] = {
  "<- Disp Temp ->", "<- Disp Humid ->", "<- Disp Both ->", "<- Language ->",
"<- Zen ->"
};
const unsigned char settingsArrRomanian[][17] = {
  "<- Afis Temp ->", "<- Afis Umid ->", "<- Afis Ambele ->", "<- Limba ->",
"<- Zen ->"
};
```

Concluzii

M-am distrat creand un FSM pentru a controla meniul principal.

Download

Codul sursa poate fi descarcat de aici sleepingmonitor.zip

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

https://www.diodes.com/assets/Datasheets/74HC595.pdf (foarte enervant)

Export to PDF