DOCUMENTATIE

Proiectare cu microprocesoare

Saca Victor-Valentin

Universitatea Tehnica din Cluj-Napoca

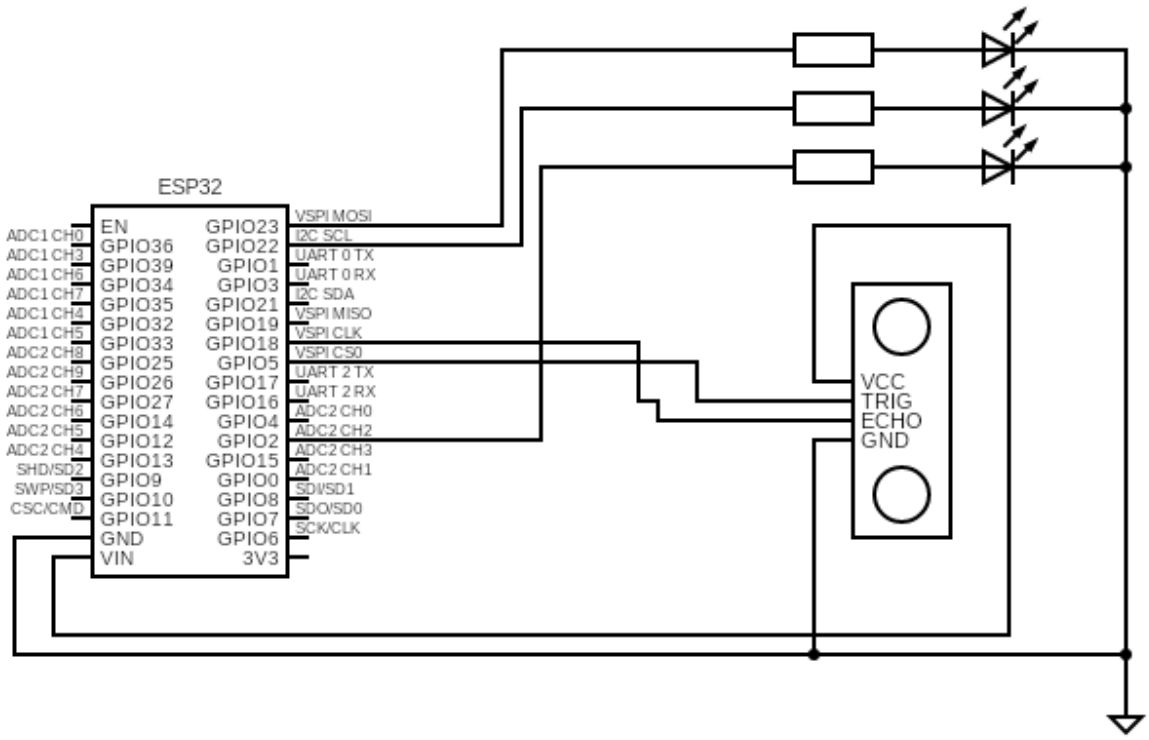
Dispozitiv de proximitate

Proiectul ales este un dispozitiv de proximitate. Acestuia, i-am integrat o conexiue la baza de date, care ii citeste datele senzorului. Senzorul este conectat la o placa ESP32 DEV kit 1, care prin WiFi, transmite datele catre baza de date dezvoltata in firebase.

Senzorul este compus din transmitator si receptor. Transmitatorul prin ultrasunete, calculeaza timpul in care sunetul a venit inapoi la receptor, si baza timpului obtinut, se obtine distanta la care se afla obiectul.

Folosind un token, ESP32 reuseste sa se conecteze la baza de date (configurat sa se conecteze anonim), unde se va afisa distanta unui obiect fata de senzor.

Dispozitivul poate fi folosit pentru parking-ul masinilor, pentru a atentiona soferul cat de aproape se afla de un obstacol. De asemenea se poate folosi in spitale unde dispozitivul poate fi folosit pentru a detecta prezenta sau miscarile unui pacient.



Schema bloc

Am folosit ca alimentare pinul Vin al placii ESP32, si am legat pinii Trig si Echo la GPIO-uri. De asemenea, am legat 3 leduri care sa alerte cat de apropiat este obiectul.

Cod C pentru mediul de dezvoltare arduino

```
#include <WiFi.h>
#include <Firebase_ESP_Client.h>
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

// WiFi credentials
#define WIFI_SSID "rue"
#define WIFI_PASSWORD "8428670103045"

// Firebase credentials
const char dburl[] = "https://esp32-5230c-default-rtdb.europe-
west1.firebasedatabase.app";
const char dbapi[] = "AIzaSyBgQDsfEHfkRQ8VFwpNDerczQwLOzavIrY";

// Firebase objects
FirebaseData firebaseData;
FirebaseConfig firebaseConfig;
FirebaseAuth auth;
// Fingerprint sensor setup
#define RX_PIN 17
#define TX_PIN 16

const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701

long duration;
float distanceCm;
float distanceInch;

bool signedUp = false;

#define GREENLED1 19
#define GREENLED2 29
#define REDLED 30

#define FIRSTLIMIT 100
#define SECONDLIMIT 50
#define LASTIMIT 10

void setup() {
  Serial.begin(115200);
   Serial.printf("Starting ESP32...");
  // WiFi connection
```

```
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to Wi-Fi...");
  }
  Serial.println("Connected with ip");
  Serial.println(WiFi.localIP());

  // Firebase configuration
  firebaseConfig.database_url = dburl;
  firebaseConfig.api_key = dbapi;
  if(Firebase.signUp(&firebaseConfig, &auth, "", "")){
    Serial.println("Signed in");
    signedUp = true;
  }else Serial.printf("Sign up error");

  firebaseConfig.token_status_callback = tokenStatusCallback;
  Firebase.begin(&firebaseConfig, &auth);
  Firebase.reconnectWiFi(true);


  // proximity sensor setup
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT);

  pinMode(GREENLED1, OUTPUT);
  pinMode(GREENLED2, OUTPUT);
  pinMode(REDLED, OUTPUT);
}

void loop() {
  // Example: Enroll a fingerprint or verify

  if(Firebase.ready() && signedUp){
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in microseconds
    duration = pulseIn(echoPin, HIGH);

  // Calculate the distance
```

```
    distanceCm = duration * SOUND_SPEED/2;

  // Convert to inches
    distanceInch = distanceCm * CM_TO_INCH;

  // Prints the distance in the Serial Monitor
    Serial.print("Distance (cm): ");
    Serial.println(distanceCm);
    Serial.print("Distance (inch): ");
    Serial.println(distanceInch);

    if(distanceCm < FIRSTLIMIT)
      digitalWrite(GREENLED1, HIGH);


    if(distanceCm < SECONDLIMIT)
      digitalWrite(GREENLED2, HIGH);


    if (distanceCm < LASTIMIT)
      digitalWrite(REDLED, HIGH);


    String path = "/sensors/proximity";  // Replace with your desired
database path
    if (Firebase.RTDB.setFloat(&firebaseData, path + "/distanceCm",
distanceCm)) {
      Serial.println("Distance in cm sent to Firebase.");
    } else {
      Serial.print("Failed to send distance in cm. Reason: ");
      Serial.println(firebaseData.errorReason());
    }

    if (Firebase.RTDB.setFloat(&firebaseData, path + "/distanceInch",
distanceInch)) {
      Serial.println("Distance in inch sent to Firebase.");
    } else {
      Serial.print("Failed to send distance in inch. Reason: ");
      Serial.println(firebaseData.errorReason());
    }
  }
  delay(2000);
}
```

In prima parte se afla declaratiile de macro cu link-ul URL al bazei de date, API-ul bazei de date si pini.

Urmeaza conectarea la WiFi pentru conectarea si inregistrarea la baza de date real-time. Pe langa asta, se configureaza pinii care sunt conectati la seznor.

Daca inregistrarea a reusit, urmeaza sa se citeasca de pe senzor cu functia PulseIn() care asteapta ca pinul echo sa treaca de la LOW la HIGH, si masoara timpul pana cand trece din nou la LOW. Astfel, se trimite lungimea de unda inapoi pentru a calcula distanta obiectului in functie de timp.

Se seteaza situatiile in care led-urile sa lumineze, iar apoi se trimit datele calculate (distanta in cm) la baza de date, unde se actualizeaza valorile in timp real.

REFERINTE:

Schema circuitului: https://www.circuit-diagram.org/editor/

Token-uri pentru conectivitate al bd: https://forum.arduino.cc/t/esp32-https-request-with-api-token/1194885

ESP32 with HC-SR04:

https://randomnerdtutorials.com/esp32-hc-sr04-ultrasonic-arduino/

baza de date firebase:

https://console.firebase.google.com/project/esp32-5230c/overview