Precyzyjny, elektroniczny lokalizator źródła zapylenia

Latem zapylenie powietrza jest zazwyczaj podwyższone, ale wokół nas jest także wiele źródeł niebezpiecznych dla zdrowia pyłów PM2,5. Opisane poniżej proste urządzenie pozwala nie tylko je wykrywać i mierzyć ich zawartość w powietrzu, lecz także lokalizować ich źródła.

Autor niniejszej konstrukcji, pracując wcześniej nad innymi monitorami poziomu tzw. pyłów zawieszonych PM2,5 (o średnicy nie większej niż 2,5 μm), zauważył jedną z wad typowych konstrukcji tego rodzaju – brak możliwości zlokalizowania punktowych źródeł zanieczyszczenia drobnymi cząstkami. Większość analiz wykonywanych na poziomie samorządów, jak i wielkopowierzchniowe analizy satelitarne gromadzą źródła dotyczące rozległych terenów, ale nie lokalizują precyzyjnie źródeł pyłów w naszym otoczeniu.

Pyły PM2,5 są, zdaniem Światowej Organizacji Zdrowia (WHO), jednym z najbardziej szkodliwych dla zdrowia człowieka zanieczyszczeń atmosfery. Jak pokazują raporty WHO, długotrwałe narażenie na obecność pyłów PM2,5 skutkuje redukcją średniej długości życia. Natomiast krótkotrwałe ekspozycje na wysokie stężenie tego rodzaju pyłów może powodować choroby układu oddechowego i krążenia oraz wzrost ryzyka nagłych przypadków wymagających hospitalizacji (astma, ostra reakcja układu oddechowego, osłabienie czynności płuc, itp.). Wynika to z faktu, iż tak drobny pył przenika przez płuca do krwi. Jak szacuje WHO, przeciętny mieszkaniec naszego kraju traci przez to nawet ponad 10 miesięcy życia.

Aby zredukować ilość zanieczyszczeń wokół nas, trzeba poznać ich źródła. Dopiero wtedy można je wyeliminować lub odizolować. Nawet w naszych domach czai się wiele emiterów tego rodzaju pyłów. Opisany poniżej „wąchacz" pozwala wyśledzić ich lokalizację, badając punktowo zawartość pyłów PM2,5 w powietrzu. Umożliwia to zastosowany sensor produkcji Honeywell. Ma on własny wentylator oraz okienka (wejściowe i wyjściowe), co umożliwia pobieranie powietrza do sensora. Autor projektu musiał jedynie opracować układ, umożliwiający selektywne pobieranie próbek powietrza w postaci ręcznego urządzenia, zasilanego z baterii. Dołożenie na wlocie „wąchacza" wydrukowanego w 3D psiego nosa było tylko uzupełnieniem estetyki urządzenia.

Potrzebne materiały

Do skonstruowania tego systemu potrzebnych jest kilka modułów elektronicznych, które można nabyć w sklepach internetowych. Podstawowym i najdroższym elementem jest, wysokiej klasy, sensor pyłów PM2,5. Autor projektu zastosował układ HPMA115S0-TIR firmy Honeywell, zainstalowany w module PMS5003. Jest on laserowym detektorem i licznikiem cząstek pyłów zawieszonych w powietrzu. Działa na zasadzie rozpraszania światła na cząsteczkach pyłu – analizowane powietrze jest przepuszczane przez wiązkę światła laserowego, która odbija się od cząstek pyłu – odbicia te są rejestrowane przez wbudowane fotodetektory i konwertowane na sygnał elektryczny, który pozwala sensorowi na obliczenie koncentracji pyłów w powietrzu.

Sensor zapylenia komunikuje się np. z mikrokontrolerem poprzez interfejs szeregowy UART. Jako kontroler tego systemu autor zastosował moduł z układem ESP32 – D1 MINI. Jest to niedrogi mikrokontroler, wyposażony w wydajny procesor oraz, między innymi, interfejsy bezprzewodowe Wi-Fi oraz Bluetooth.

Płytka z mikrokontrolerem i sensor zasilane są z pojedynczego ogniwa 18650 poprzez moduł zasilacza z przetwornicą typu boost. Moduł ten zapewnia kontrolę całej sekcji zasilania, uniemożliwia niebezpiecznie głębokie rozładowanie ogniwa, a także kontroluje jego ładowanie.

Interfejs użytkownika składa się z dwóch elementów – ekranu OLED oraz spustu. Wyświetlacz ma przekątną 0,96" i rozdzielczość 128×64 piksele, co przy tak małym ekranie jest w zupełności wystarczające do zapewnienia czytelnego obrazu. Spust steruje microswitchem w urządzeniu, który wyzwala pomiar. Jego wartość jest następnie wyświetlana na ekranie OLED. Drugi przycisk, znajdujący się na obudowie, steruje zasilaniem układu.

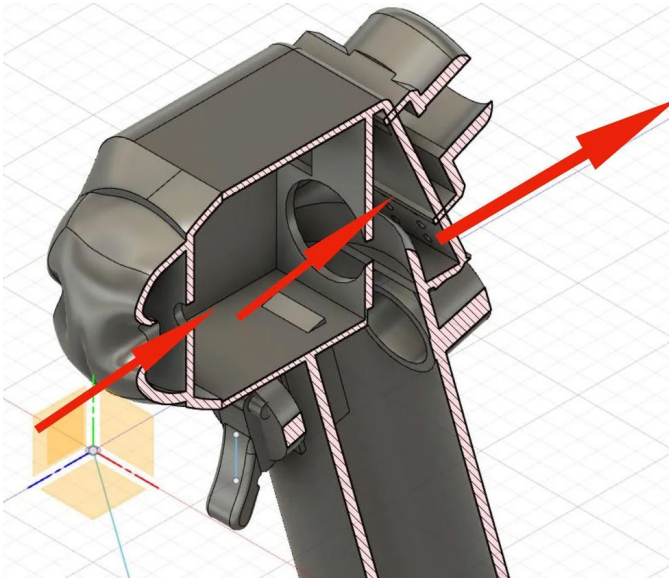
Konstrukcja mechaniczna

Oprócz elementów elektronicznych, do budowy systemu potrzebna jest również obudowa. Ta została wykonana w pełni w technice druku 3D, więc dostęp do tego typu drukarki będzie konieczny. Projekt wymaga wydrukowania pięciu elementów.

Analizator zaprojektowany jest w taki sposób, że zintegrowane w czujniku Honeywella wentylatory przemieszczają powietrze przez urządzenie. Nozdrza na jednym końcu łączą się bezpośrednio z portami wejściowymi czujnika, a wylotowy otwór wentylacyjny przechodzi przez obudowę i przez liczne otwory w tylnej pokrywie. Dzięki

Listing 1. Szkic Arduino sterujący analizatorem zawartości pyłów

```
#include <HardwareSerial.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Fonts/FreeSerifBoldItalic24pt7b.h>
#include <Fonts/FreeSerifBold9pt7b.h>
// Szerokość ekranu OLED w pikselach
#define SCREEN_WIDTH 128
 // Wysokość ekranu OLED w pikselach
#define SCREEN_HEIGHT 64
#include <Adafruit_NeoPixel.h>

#define PIN 18
// Deklaracja pinów do podłączenia kontrolera SSD1306
Adafruit_SSD1306 display
(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
// Deklaracja pinów do podłączenia diod RGB LED
Adafruit_NeoPixel strip = Adafruit_NeoPixel
(12, PIN, NEO_GRB + NEO_KHZ800);

long lastMsg = 0;
char msg[50];
bool HPMAstatus = false;
int zedLevel = 1;
int PM25;
int PM10;
int oldP;
HardwareSerial HPMA115S0(1);
TXD2 17

void setup(){
    Serial.begin(9600, SERIAL_8N1);
    delay(100);
    while (!Serial);
     if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
        Serial.println("SSD1306 allocation failed");
        for(;;);
    }
    delay(2000);
    strip.begin();
    strip.show(); // inicjalizacja LED RGB jako wyłączone
    HPMA115S0.begin(9600, SERIAL_8N1, RXD2, TXD2);
    while (!HPMA115S0);
    start_autosend();
}

void loop(){
    strip.show();
    brighten(zedLevel);
    long now = millis();

    if (now - lastMsg > 1000) {
        lastMsg = now;
        display.clearDisplay();
        // Odbior informacji z sensora pyłów
        HPMAstatus = receive_measurement();
        if (!HPMAstatus) {
            Serial.println
            ("Cannot receive data from HPMA115S0!");
            return;
        }
    snprintf (msg, 16, "%D", PM25);
    snprintf (msg, 16, "%D", PM10);
    if(PM10 != 0){
        Serial.println("PM 2.5:\t" + String(PM25) + " ug/m3");
        Serial.println("PM 10:\t" + String(PM10) + " ug/m3");
        Serial.println("MQTT published HPMA115S0.");
        display.setFont(&FreeSerifBold9pt7b);
        display.clearDisplay();
        display.setTextSize(1);
        display.setTextColor(WHITE);
        display.setCursor(10,13);
        display.print("PPM");
        display.setCursor(90,13);
        display.println("2.5");
        display.setFont(&FreeSerifBoldItalic24pt7b);
        display.setTextSize(1);
        display.setTextColor(WHITE);
        display.setCursor(30,50);
        display.println(String(PM25));
        display.display();
        if(PM25 < 25) zedLevel = 3;
        else if(PM25 > 24 && PM25 < 80) zedLevel = 2;
        else zedLevel = 1;
        Serial.println(zedLevel);
    }
}

bool receive_measurement (void){
    while(HPMA115S0.available() < 32);
    byte HEAD0 = HPMA115S0.read();
     byte HEAD1 = HPMA115S0.read();
    while (HEAD0 != 0x42) {
        if (HEAD1 == 0x42) {
            HEAD0 = HEAD1;
            HEAD1 = HPMA115S0.read();
        } else {
            HEAD0 = HPMA115S0.read();
            HEAD1 = HPMA115S0.read();
        }
    }
    if (HEAD0 == 0x42 && HEAD1 == 0x4D) {
        byte LENH = HPMA115S0.read();
        byte LENL = HPMA115S0.read();
        byte Data0H = HPMA115S0.read();
        byte Data0L = HPMA115S0.read();
        byte Data1H = HPMA115S0.read();
        byte Data1L = HPMA115S0.read();
        byte Data2H = HPMA115S0.read();
        byte Data2L = HPMA115S0.read();
        byte Data3H = HPMA115S0.read();
        byte Data3L = HPMA115S0.read();
        byte Data4H = HPMA115S0.read();
        byte Data4L = HPMA115S0.read();
        byte Data5H = HPMA115S0.read();
        byte Data5L = HPMA115S0.read();
        byte Data6H = HPMA115S0.read();
        byte Data6L = HPMA115S0.read();
        byte Data7H = HPMA115S0.read();
        byte Data7L = HPMA115S0.read();
        byte Data8H = HPMA115S0.read();
        byte Data8L = HPMA115S0.read();
        byte Data9H = HPMA115S0.read();
        byte Data9L = HPMA115S0.read();
        byte Data10H = HPMA115S0.read();
        byte Data10L = HPMA115S0.read();
        byte Data11H = HPMA115S0.read();
        byte Data11L = HPMA115S0.read();
        byte Data12H = HPMA115S0.read();
        byte Data12L = HPMA115S0.read();
        byte CheckSumH = HPMA115S0.read();
        byte CheckSumL = HPMA115S0.read();
        if (((HEAD0 + HEAD1 + LENH + LENL + Data0H + Data0L +
            Data1H + Data1L + Data2H + Data2L + Data3H + Data3L +
            Data4H + Data4L + Data5H + Data5L + Data6H + Data6L +
            Data7H + Data7L + Data8H + Data8L + Data9H + Data9L +
            Data10H + Data10L + Data11H + Data11L + Data12H +
Data12L)
            % 256) != CheckSumL){
            Serial.println("Checksum fail");
            return 0;
        }
        PM25 = (Data1H * 256) + Data1L;
        PM10 = (Data2H * 256) + Data2L;
        return 1;
    }
}

bool start_autosend(void){
    // Start auto send
    byte start_autosend[] = {0x68, 0x01, 0x40, 0x57 };
    HPMA115S0.write(start_autosend, sizeof(start_autosend));
    HPMA115S0.flush();
    delay(500);
    // Czekanie na odpowiedź
    while(HPMA115S0.available() < 2);
    byte read1 = HPMA115S0.read();
    byte read2 = HPMA115S0.read();
    // Sprawdzanie poprawności odpowiedzi
    if ((read1 == 0xA5) && (read2 == 0xA5)){
        // ACK
        return 1;
    } else if ((read1 == 0x96) && (read2 == 0x96)) {
        // NACK
        return 0;
    }
    else return 0;
}

void brighten(int q){
    uint16_t i, j;
    Serial.print("this is q");
    Serial.println(q);
    switch (q) {
        case 1:
        for (j = 5; j < 155; j++) {
            for (i = 0; i < strip.numPixels(); i++) {
                strip.setPixelColor(i, j,  0, 0);
            }
            strip.show();
            delay(255/(j+1));
        }
        for (j = 155; j > 6; j--) {
            for (i = 0; i < strip.numPixels(); i++) {
                strip.setPixelColor(i, j, 0, 0);
            }
            strip.show();
            delay(255/(j+1));
        }
        break;
        case 2:
        for (j = 5; j < 155; j++) {
            for (i = 0; i < strip.numPixels(); i++) {
                strip.setPixelColor(i, 0,  j, 0);
            }
            strip.show();
            delay(255/(j+1));
        }
        for (j = 155; j > 6; j--) {
            for (i = 0; i < strip.numPixels(); i++) {
                strip.setPixelColor(i, 0, j, 0);
            }
            strip.show();
            delay(255/(j+1));
        }
        break;
        case 3:
        for (j = 5; j < 155; j++) {
            for (i = 0; i < strip.numPixels(); i++) {
                strip.setPixelColor(i, 0,    0, j);
            }
            strip.show();
            delay(255/(j+1)); ‘p4h m
        }
        for (j = 155; j > 6; j--) {
            for (i = 0; i < strip.numPixels(); i++) {
                strip.setPixelColor(i, 0, 0, j);
            }
            strip.show();
            delay(255/(j+1));
        }
        break;
    }
}
```



Rysunek 1. Przekrój komory sensora, wraz z zaznaczonymi ścieżkami przepływu powietrza

temu możliwe jest wprowadzanie do sensora powietrza z precyzyjnie określonej lokalizacji. Jest to dokładnie zobrazowane na **rysunku 1**.

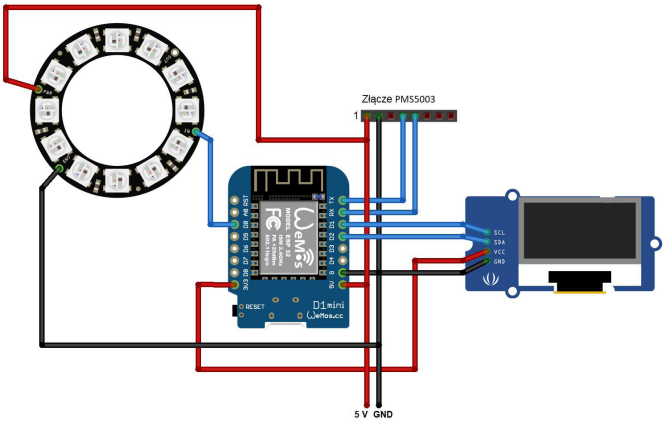
Pokaźny uchwyt „wąchacza" pozwala na ukrycie w nim akumulatora o dużej pojemności i reszty elektroniki. Port ładowania znajduje się w dolnej części obudowy rękojeści. Oświetlenie Neopixel wokół nosa zostało zaprojektowane tak, aby świecić przez obudowę u góry. Konstrukcja górnej części głównej obudowy została wydrukowana z przezroczystego PLA, a następnie – podczas druku – zamieniono materiał na szare PLA, którym wydrukowano resztę obudowy. Podobny zabieg wykonano podczas druku finalnej części obudowy, dzięki czemu możliwe jest dostrzeżenie kolorów diod LED, umieszczonych wewnątrz obudowy, sygnalizujących stan naładowania akumulatora.

Mechanizm spustowy jest złożony z przycisku, który jest wydrukowany jako jeden element. Podczas montażu należy upewnić się, że porusza się on swobodnie i naciska na dźwignię przełącznika krańcowego, zwierającego obwód.

Pliki STL z poszczególnymi elementami, gotowymi do druku 3D, znaleźć można na stronie z projektem (patrz link na końcu artykułu). Autor przygotowywał je do druku ze standardowymi ustawieniami w programie Cura dla drukarki Ender 3. Żadna z części nie była wyposażona w podpory podczas drukowania.

Budowa

Schemat połączeń poszczególnych modułów pokazano na **rysunku 2**. Do modułu z mikrokontrolerem podłączony jest sensor pyłów, przez interfejs UART na pinach X (21) i Y (22) modułu z ESP32. Wyświetlacz komunikuje się z mikrokontrolerem poprzez interfejs I²C na pinach 19 (SDA)



Rysunek 2. Schemat połączeń elektrycznych analizatora pyłów

oraz 20 (SCL). Moduł z mikrokontrolerem, diody LED RGB Neopixel oraz sensor pyłów zasilane są z napięcia 5 V pochodzącego z przetwornicy napięcia. Ekran OLED zasila stabilizator liniowy 3,3 V, który zintegrowany jest w module z mikrokontrolerem. Przetwornica zasilana jest z pojedynczego ogniwa litowo-jonowego w rozmiarze 18650. Całość uzupełnia przycisk zasilania, odcinający połączenie baterii z przetwornicą i w ten sposób całkowicie odcinający zasilanie od systemu.

Oprogramowanie

System oprogramowany jest w pełni w Arduino IDE. Kod programu sterującego „wąchaczem" zamieszczono na **listingu 1**. Jest również dostępny w postaci szkicu Arduino na stronie internetowej projektu (patrz link na końcu artykułu).

Program wykorzystuje port szeregowy do komunikacji z czujnikiem. Pomiar wykonywany jest raz na sekundę, a wyniki są przesyłane do modułu ESP32. Jest to dostateczna częstotliwość, by zapewnić urządzeniu wygodną responsywność np. po naciśnięciu spustu.

Wyniki pomiarów prezentowane są na ekranie OLED z kontrolerem SSD1306 sterowanym przez interfejs I²C. Diody LED Neopixel są kontrolowane przez bibliotekę Adafruit Neopixel w raczej konwencjonalny sposób – kolor diod zależny jest od zmierzonego stężenia pyłów PM2,5. Jeśli poziom jest mniejszy niż 25, diody LED migają na niebiesko, jeśli poziom mieści się pomiędzy 25 a 80 – na zielono, i na czerwono, jeśli poziom jest większy niż 80. Te, wstępnie ustawione, poziomy można zmienić w kodzie programu, dostosowując urządzenie do własnych potrzeb czy krajowych norm.

Po skompilowaniu programu i zaprogramowaniu modułu ESP32 (wystarczy zwykły kabel USB) system jest gotowy do pracy. Wystarczy nacisnąć spust, aby dokonać pomiaru i monitorować poziom pyłów zawieszonych dookoła nas.

Nikodem Czechowski, EP

Źródło: ***https://bit.ly/2OKX7mH***