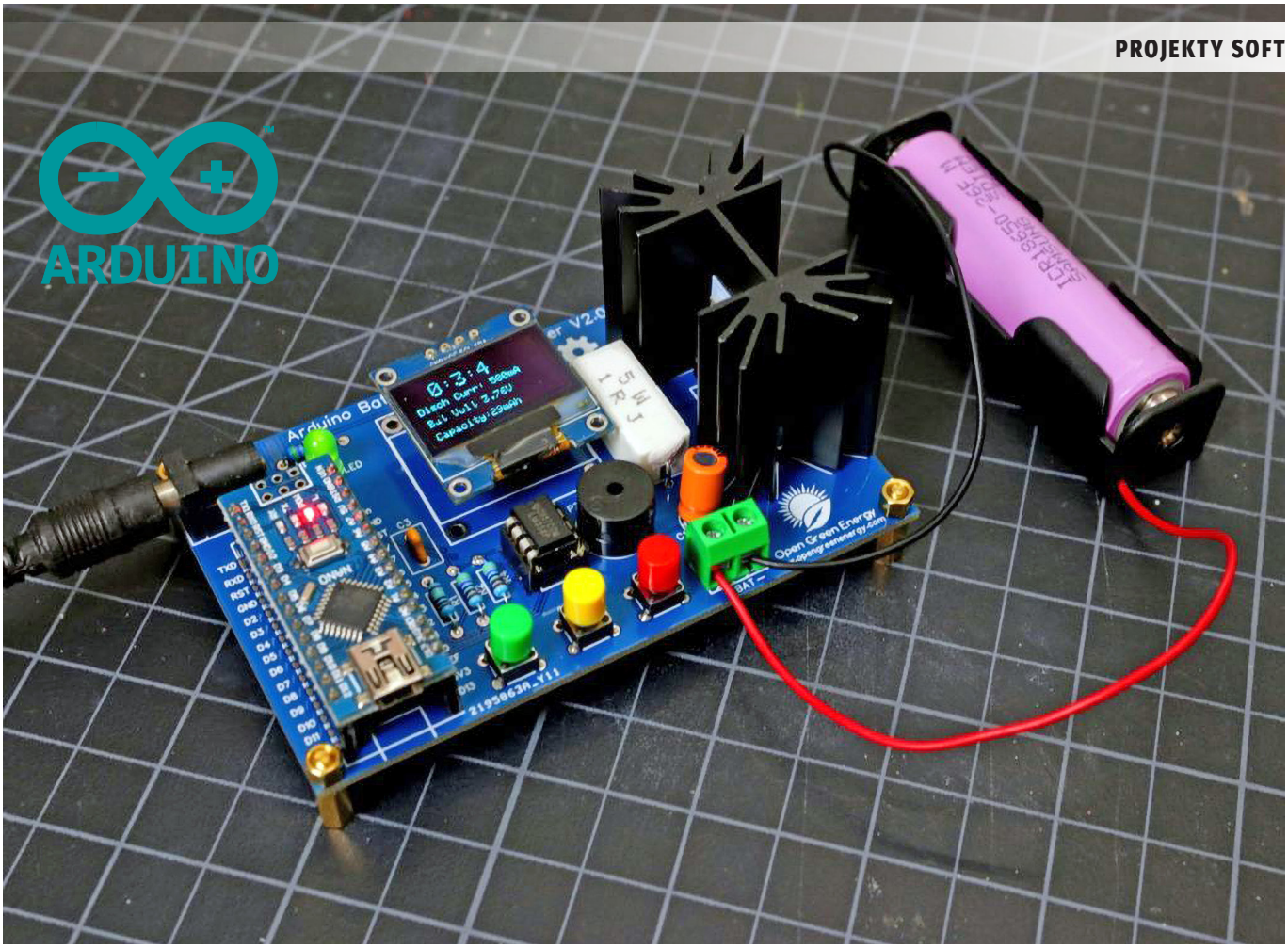


Miernik pojemności akumulatorków litowych i NiMH

Fałszowanie ogniw – szczególnie akumulatorków litowo-jonowych i niklowo-wodorkowych (NiMH) jest smutnym faktem. Wiele firm reklamuje ogniwa czy powerbanki, podając ogromne wartości pojemności, które znacznie przekraczają rzeczywiste parametry tych komponentów. Dzięki prezentowanemu urządzeniu z łatwością można zmierzyć realną pojemność ogniw i wybrać te o odpowiednich parametrach.

Trudno jest zweryfikować rzeczywistą wydajność baterii czy akumulatorków, szczególnie gdy kupuje się ich dużo. Podobnie trudno jest określić pojemność dostępną z odzyskiwanych ogniw w formacie 18650 (np. z laptopów). Urządzenie do pomiaru rzeczywistej pojemności akumulatorów pozwoliłoby na odpowiednią selekcję ogniw do zasilania dowolnego urządzenia.

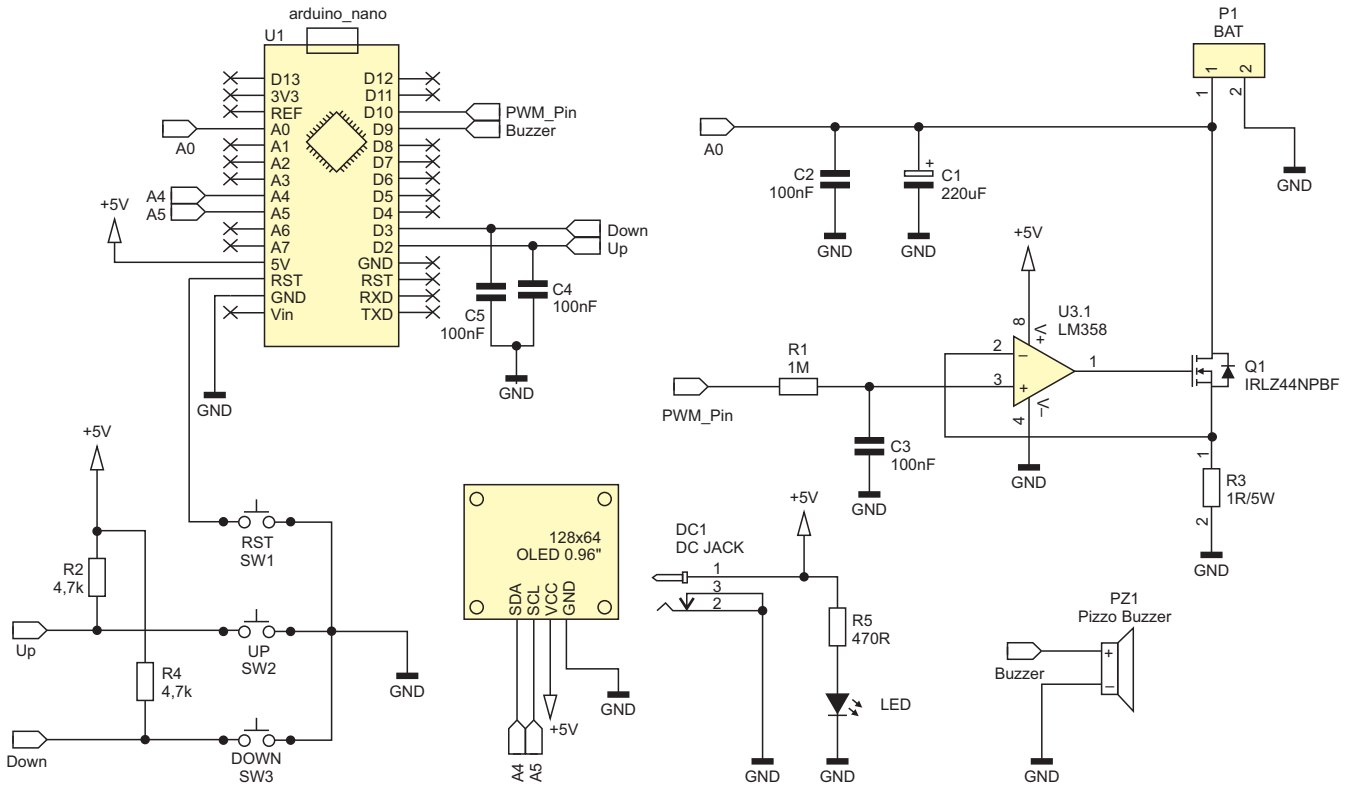
Prezentowany poniżej układ jest drugą generacją miernika pojemności ogniw, jaką stworzył autor projektu, występujący na portalu Instructables pod nickiem Open Green Energy. Zbudowany wcześniej, w 2016 roku, system rozładowywał ogniwo przez stały opornik, a moduł Arduino mierzył prąd i napięcie w funkcji czasu – finalnie, po rozładowaniu ogniwa do predefiniowanego poziomu, pojemność wyznaczana była jako iloczyn prądu rozładowywania i czasu, jaki zajęło rozładowanie ogniwa do zadanego poziomu. Jednak podczas pomiaru, wraz ze spadkiem napięcia akumulatora, prąd również maleje, przez co obliczenia pojemności są złożone i z tego powodu końcowo niedokładne.

Aby zaradzić temu problemowi, autor stworzył wersję określoną jako V2.0, która została zaprojektowana w taki sposób, aby prąd pozostawał stały przez cały proces rozładowywania. Udało się to osiągnąć dzięki zastosowaniu aktywnego obciążenia do rozładowania akumulatora. Prąd rozładowywania ogniwa może być stały w całym czasie trwania pomiaru. Główne cechy testera pojemności V2.0 to:

- możliwość pomiaru pojemności ogniw NiMH, NiCd, Li-Ion, Li-Poli oraz $LiFePo_4$ w różnych rozmiarach – AA, AAA, 18650 i innych. Jedyny warunek to napięcie ogniwa poniżej 5 V,
- konfigurowalny prąd rozładowywania ogniwa,
- interfejs użytkownika na bazie czytelnego wyświetlacza OLED,
- możliwość zastosowania układu jako programowalnego obciążenia elektronicznego.

Do budowy tego systemu potrzebne będą m.in.:

- płytka drukowana (projekt autora udostępniony jest w Internecie),
- moduł Arduino Nano,



Rysunek 1. Schemat testera do akumulatorków

- wzmacniacz operacyjny LM358,
- moduł wyświetlacza OLED o przekątnej 0,96”.

Budowa

Cały schemat został pokazany na **rysunku 1**, można w nim wydzielić następujące sekcje:

1. zasilania,
2. obciążenia stałoprądowego,
3. pomiaru napięcia akumulatora,
4. interfejsu użytkownika,
5. sygnalizacji – brzęczyk alarmowy.

Obwód zasilania

Obwód zasilający składa się z gniazda dla zasilacza (napięcie wejściowe w zakresie 7...9 V, i dwóch kondensatorów filtrujących C1 i C2. Napięcie wejściowe (Vin) jest podłączone do pinu Vin Arduino. Układ korzysta z wbudowanego w Arduino stabilizatora napięcia, aby obniżyć napięcie do 5 V, które zasila cały układ.

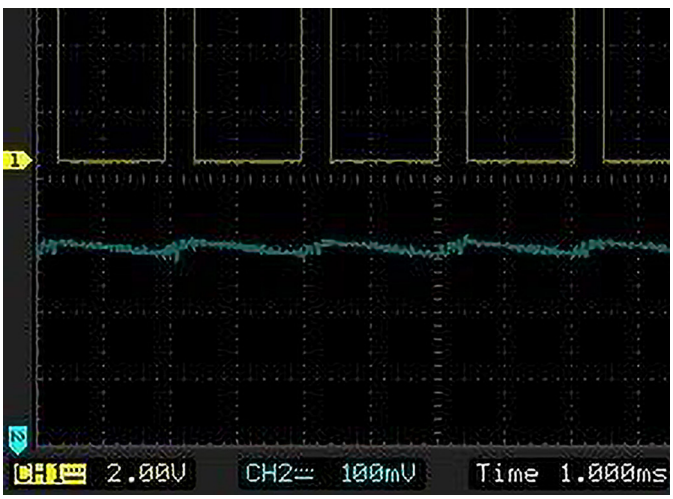
Obwód stałego prądu obciążenia

Podstawowym elementem tego obwodu jest układ scalony LM358, który zawiera w sobie dwa wzmacniacze operacyjne. Sygnał PWM z cyfrowego pinu D10 Arduino jest doprowadzony do filtra dolnoprzepustowego złożonego z elementów R1 i C3 i podawany na wejście nieodwracające op-ampa skonfigurowanego, wraz z tranzystorem polowym Q1, jako źródło prądowe.

Wzmacniacz operacyjny U3 zasilany jest stabilizowanym napięciem 5 V, filtrowanym przez kondensator, który musi być umieszczony blisko tego elementu. Układ U3, wraz z R3 i Q1, formuje aktywne obciążenie DC, które rozładowuje akumulator. Prąd, płynący przez rezystor R3, jest sterowany poprzez modulację szerokości impulsu (PWM) podawanego przez Arduino na Q1.

Zasada działania źródła prądowego, zbudowanego w tym systemie, jest bardzo prosta. Wzmacniacz operacyjny U3 porównuje napięcie na pinie 2 (wejście odwracające wzmacniacza) i pinie 3 (wejście nieodwracające). Sam op-amp skonfigurowany jest jako bufor o jednostkowym wzmocnieniu. Na wejście nieodwracające podawany jest filtrowany przebieg PWM, co powoduje, że na wyjściu wzmacniacza pojawia się jakieś napięcie, które otwiera bramkę MOSFET-a. Gdy włącza się MOSFET, prąd przepływa przez R3, wytwarza pewien spadek napięcia na tym oporniku, który zapewnia ujemne sprzężenie zwrotne dla wzmacniacza operacyjnego. Układ kontroluje tranzystor polowy w taki sposób, aby napięcia na wejściach odwracającym i nieodwracającym było takie samo (za pomocą sterowania bramką MOSFET-a). Tak więc prąd płynący przez rezystor obciążenia będzie proporcjonalny do napięcia na wejściu nieodwracającym op-ampa.

Sterujące działaniem źródła prądowego napięcie na wejściu nieodwracającym pochodzi z filtra dolnoprzepustowego RC, na który podawany jest sygnał prostokątny o zmiennym wypełnieniu, generowany przez Arduino. Na **rysunku 2** zaprezentowano sygnał PWM z Arduino (kanał 1, kolor żółty) i sygnał za filtrem (kanał 2, kolor zielony). W ten sposób z wyjścia PWM i filtra RC zestawiony jest prosty przetwornik DAC. Wartości elementów filtra RC można dobrać, posługując się oscyloskopem i analizując sygnał wyjściowy z filtra dla różnych wartości elementów i/lub częstotliwości sygnału PWM.



Rysunek 2. Oscylogram napięcia na wejściu filtra – przebieg PWM (żółta linia) i na wyjściu filtra (zielona linia)

Listing 1. Kod programu miernika pojemności akumulatorków

```
#include <JC_Button.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // Szerokość ekranu OLED (w pikselach)
#define SCREEN_HEIGHT 64 // Wysokość ekranu OLED (w pikselach)

// Deklaracja podłączenia pinów do sterownika SSD1306 przez I2C
// Pin resetu dla modułu OLED (-1, jeżeli współdzieli on reset z Arduino)
#define OLED_RESET     4
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

const float Low_BAT_level = 3.0;
// Ustawienia prądu (w mA) dla zainstalowanego opornika R7
const int Current [] = {0,110,210,300,390,490,580,680,770,870,960,1000};
const byte PWM_Pin = 10;
const byte Buzzer = 9;
const int BAT_Pin = A0;
int PWM_Value = 0;
unsigned long Capacity = 0;
int ADC_Value = 0;
// Napięcie zasilania 5 V (zmierzone multimetrem na module Arduino)
float Vcc = 4.96 ;
float BAT_Voltage = 0;
float sample =0;
byte Hour = 0, Minute = 0, Second = 0;
bool calc = false, Done = false;
Button UP_Button(2, 25, false, true);
Button Down_Button(3, 25, false, true);


void setup () {
    // Inicjalizacja portu szeregowego i wyjść/wejść Arduino
    Serial.begin(9600);
    pinMode(PWM_Pin, OUTPUT);
    pinMode(Buzzer, OUTPUT);
    analogWrite(PWM_Pin, PWM_Value);
    UP_Button.begin();
    Down_Button.begin();
    // Inicjalizacja wyświetlacza OLED
    display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
    display.clearDisplay();
    display.setTextColor(WHITE);
    display.setTextSize(1);
    display.setCursor(12,25);
    display.print(„Open Green Energy”);
    display.display();
    delay(3000);
    display.clearDisplay();
    display.setTextSize(2);
    display.setCursor(2,15);
    display.print(„Ustaw prąd:”);
    display.setCursor(2,40);
    display.print(„Góra/Dół:”);
    display.print(„0”);
    display.display();
}

void loop() {
    // Odczytywanie przycisków i zmiana parametrów
    UP_Button.read();
    Down_Button.read();
    if (UP_Button.wasReleased() && PWM_Value < 55 && calc == false) {
        PWM_Value = PWM_Value + 5;
        analogWrite(PWM_Pin,PWM_Value);
        display.clearDisplay();
        display.setCursor(2,25);
        display.print(„Prąd:”);
        display.print(String(Current[PWM_Value / 5])+” mA”);
        display.display();
    }
    if (Down_Button.wasReleased() && PWM_Value > 1 && calc == false) {
        PWM_Value = PWM_Value - 5;
        analogWrite(PWM_Pin,PWM_Value);
        display.clearDisplay();
        display.setCursor(2,25);
        display.print(„Prąd:”);
        display.print(String(Current[PWM_Value / 5])+” mA”);
        display.display();
    }

    if (UP_Button.pressedFor (1000) && calc == false) {
        digitalWrite(Buzzer, HIGH);
        delay(100);
        digitalWrite(Buzzer, LOW);
        display.clearDisplay();
        timerInterrupt();
    }
}

void timerInterrupt(){
    calc = true;
    while (Done == false) {
        Second ++;
        if (Second == 60) {
            Second = 0;
            Minute ++;
        }
        if (Minute == 60) {
            Minute = 0;
            Hour ++;
        }
        for(int i=0;i< 100;i++) {
            // Pomiar napięcia na ogniwie
            sample = sample + analogRead(BAT_Pin);
            delay(2);
        }
        sample=sample / 100;
        BAT_Voltage = sample * Vcc / 1024.0;
```

Obwód pomiaru napięcia akumulatora

Napięcie akumulatora jest mierzone przez pin A0 (wejście przetwornika analogowo-cyfrowego w Arduino). Dwa kondensatory C1 i C2 służą do filtrowania wszelkich zakłóceń obecnych w napięciu badanego ogniwa, które pochodzą z tętnień prądu z obwodu aktywnego obciążenia. Zakłócenia te mogłyby obniżyć precyzję pomiaru przetwornika ADC w Arduino.

Interfejs użytkownika

Obwód interfejsu użytkownika składa się z dwóch przycisków i modułu wyświetlacza OLED o przekątnej 0,96”, sterowanego przez interfejs I²C. Dwa przyciski opisane są jako UP i DOWN i służą do zwiększania lub zmniejszania szerokości impulsu PWM, co pozwala na sterowanie prądem rozładowywania akumulatora. Oporniki R2 i R4 podciągają w górę obie linie podłączone do przycisków. Trzeci przycisk (RST) służy do resetowania modułu Arduino.

Napięcie akumulatora, prąd rozładowania i wyliczona pojemność są prezentowane na wyświetlaczu OLED. Jego rozdzielczość wynosi 128×64 pikseli a komunikacja z Arduino odbywa się poprzez magistralę I²C, dzięki czemu do przesyłania do niego danych wystarczą tylko dwie linie sygnałowe – pin SCL (A5) oraz SDA (A4). Pozostałe dwa piny płytki z wyświetlaczem to zasilanie (+5 V i GND).

Alarm

Buzzer zasilany 5 V dołączony jest do pinu cyfrowego D9 modułu Arduino. Wyjście to steruje alarmem, który sygnalizuje rozpoczęcie i zakończenie pomiaru pojemności ogniwa. W prototypowej konstrukcji autor umieścił buzzer na kablach, gdyż na płytce uniwersalnej, na której montował system, nie znalazło się już miejsce na ten element, ale na docelowej płytce PCB jest już miejsce na jego zamocowanie. Sposób umieszczenia elementu w systemie nie ma wpływu na jego funkcjonalność.

Software

Podczas pomiaru pojemności ogniwa układ stabilizuje prąd na zadanym poziomie i rozładowuje ogniwo do uzyskania zadanego napięcia, zależnego od rodzaju akumulatora (np. 3,2 V dla ogniwa litowo-jonowego). Pojemność baterii (w mAh) wyliczana jest jako ustawiony prąd (w mA) pomnożony przez czas (w godzinach), jaki zajęło rozładowanie ogniwa do napięcia progowego. Dlatego też stabilizacja prądu w czasie rozładowywania ogniwa jest tak istotna – jeśli przez cały czas T prąd utrzymywano na stałym poziomie I, to pojemność ogniwa jest łatwa do obliczenia – wynosi T×I. Prąd rozładowywania ogniwa może być regulowany poprzez zmianę wypełnienia sygnału PWM, sterującego źródłem prądowym.

Zanim zaczniemy zajmować się samym szkicem Arduino, sterującym urządzeniem, musimy pobrać i zainstalować dwie biblioteki:

- JC_Button, która służy do obsługi przycisków (znaleźć można ją na GitHubie) ***https://github.com/JChristensen/JC_Button***.
- Adafruit_SSD1306 do obsługi wyświetlacza OLED ze sterownikiem SSD1306 (również na GitHubie) ***https://github.com/JChristensen/JC_Button***.

Kod programu dla Arduino IDE zawarto na **listingu 1**. Przed skompilowaniem w kodzie uzupełnić należy dwie wartości kalibracyjne:

- Wartości tablic prądu dla różnych ustawień PWM dla zastosowanego opornika mocy. Prąd ten mierzymy, umieszczając multimetr w szeregu z ogniwem, już po zmontowaniu naszego urządzenia. Przyciskami zmieniamy wypełnienie sygnału sterującego i mierzymy płynący dla poszczególnych progów prąd ogniwa. Poszczególne wartości prądu umieszczamy w tablicy *Current*.
- Poziom napięcia VCC w naszym układzie. Mierzymy multimetrem napięcie 5 V na pinie VCC Arduino, a fizyczną wartość wpisujemy w zmiennej *Vcc*.

Po dokonaniu powyższej kalibracji kod programu jest gotowy do kompilacji. Możemy jeszcze zmienić wartość zmiennej *Low_BAT_Level* – jest to próg napięcia, do jakiego rozładowywane będzie ogniwo. Jest on zależny od rodzaju ogniwa. Dobrze jest ustawić ten próg lekko powyżej najniższego możliwego napięcia dla danej chemii ogniw. Dalej znajdują się minimalne napięcia ogniw o różnej chemii dla rozładowywania ogniw prądem 1 C.

Listing 1. Kod programu miernika pojemności akumulatorków - cd.

```
        display.clearDisplay();
        display.setTextSize(2);
        display.setCursor(20,5);
        display.print(String(Hour) + „:” + String(Minute) + „:” + String(Second));
        display.setTextSize(1);
        display.setCursor(0,25);
        display.print(„Prąd rozł.: „);
        display.print(String(Current[PWM_Value / 5])+” mA”);
        display.setCursor(2,40);
        display.print(„Nap. ogniwa:” + String(BAT_Voltage)+” V” );
        Capacity =  (Hour * 3600) + (Minute * 60) + Second;
        Capacity = (Capacity * Current[PWM_Value / 5]) / 3600;
        display.setCursor(2, 55);
        display.print(„Pojemność:” + String(Capacity) + „ mAh”);
        display.display();

        if (BAT_Voltage < Low_BAT_level) {
            Capacity =  (Hour * 3600) + (Minute * 60) + Second;
            Capacity = (Capacity * Current[PWM_Value / 5]) / 3600;
            display.clearDisplay();
            display.setTextSize(2);
            display.setCursor(2,15);
            display.print(„Pojemność:”);
            display.setCursor(2,40);
            display.print(String(Capacity) + „ mAh”);
            display.display();
            Done = true;
            PWM_Value = 0;
            analogWrite(PWM_Pin, PWM_Value);
            digitalWrite(Buzzer, HIGH);
            delay(100);
            digitalWrite(Buzzer, LOW);
            delay(100);
            digitalWrite(Buzzer, HIGH);
            delay(100);
            digitalWrite(Buzzer, LOW);
            delay(100);
        }
        delay(1000);
    }
}
```

1. tlenek litowo-kobaltowy: napięcie odcięcia = 2,5 V,
2. tlenek litowo-manganowy: napięcie odcięcia = 2,5 V,
3. fosforan litu i żelaza: napięcie odcięcia = 2,5 V,
4. tytanian litu: napięcie odcięcia = 1,8 V,
5. tlenek litowo-manganowo-kobaltowy: napięcie odcięcia = 2,5 V,
6. tlenek glinowo-litowo-kobaltowo-glinowy: napięcie odcięcia = 3,0 V.

Nikodem Czechowski

Źródło: ***http://bit.ly/35C0Qtq***