Obsługa modułu AVTduino Motor w Arduino

W artykule opiszemy programową obsługę modułu AVTduino Motor dla systemu Arduino UNO, którego konstrukcję opisano w bieżącym numerze „Elektroniki Praktycznej” w rubryce „miniprojekty”. Umożliwia on sterowanie czterema silnikami prądu stałego oraz ma elementy, które pozwalają na budowę nieskomplikowanego interfejsu użytkownika.

Moduł ma wbudowane dwie diody LED, cztery przyciski oraz dwa potencjometry, których można użyć do sygnalizowania statusu sterownika Arduino oraz do sterowania prędkością obrotową silnika z wykorzystaniem sygnału PWM. Moduł sterujący silnikami DC będzie podstawowym układem przeznaczonym do budowy pojazdów czy robotów.

Na **rysunku 1** pokazano sposób dołączenia do modułu silników zasilanych prądem stałym, które dodatkowo powinny być zabezpieczone diodami. Układ sterujący pracą silników umożliwia ich załączanie, wyłączanie, zmianę kierunku obrotów oraz zmiany prędkości poprzez sterowanie sygnałem PWM jednego z wejść driverów sterujących danym silnikiem. Przykład obsługi jest przeznaczony dla użytkowników systemu Arduino, którzy na jego podstawie będą mogli zrozumieć ideę sterowania silnikami i łatwo zaadaptować go do własnych potrzeb.

Na **listingu 1** zamieszczono przykładowy program testowy dla modułu AVTduino Motor. Program ten umożliwia sterowanie 4 silnikami. Dwa silniki będą obracały się w umownym kierunku „w lewo”, natomiast dwa „w prawo”. Naciśnięcie przycisku S1 umożliwia włączenie lub wyłączenie silników M1 i M2, a S3 silników M3 i M4. Dioda LED1 sygnalizuje załączenie silników M1 i M2, natomiast LED2 silników M3 i M4. Przycisk S2 umożliwia zmianę kierunku obrotów silników M1 i M2 na przeciwny, a przycisk S4 silników M3 i M4. Potencjometr Pot1 służy do regulowania prędkości obrotowej silników M1 i M2, a potencjometr Pot2 silników M3 i M4.

W programie w pierwszej kolejności definiowane są stałe, w których zdefiniowano numery portów. Jako kolejne definiowane są zmienne (flagi), które sygnalizują stan silników. W procedurze *setup()* są konfigurowane linie sterujące silnikami oraz diodami LED jako wyjściowe, a linie, do których dołączono przyciski, jako wejściowe. W procedurze głównej programu *loop()* za pomocą komendy *mot1 = analogRead(A0)* jest odczytywana wartość analogowa z potencjometru Pot1. Liczba to zostaje zapamiętana w zmiennej *Mot1*. Podobnie, w dalszej części programu, jest odczytywana nastawa potencjometru Pot2 i odpowiadająca jej liczba jest zapisywana do zmiennej *Mot2*. W instrukcjach:

```
  if (digitalRead(SW1) == LOW) {
//Sprawdzenie czy naciśniety
przycisk SW1
    flaga1_on=!flaga1_on;      //
odwrócenie stanu flagi flaga1_on
wskazującej załączenie silnika M1
i M2
    while(digitalRead(SW1) ==
LOW);    //oczekiwanie na puszc-
zenie przycisku SW1
  }
```

jest realizowana obsługa przycisku S1, którego naciśnięcie zmienia na przeciwny stan zmiennej *flaga1_on*. Od tej flagi zależy załączenie silników M1 i M2. Komendy:

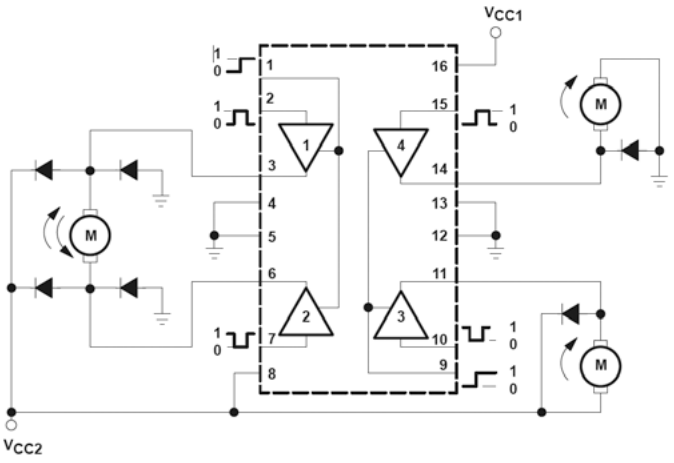
```
if (digitalRead(SW2) == LOW) {
//Sprawdzenie czy naciśnięty
przycisk SW2
    flaga1_obr=!flaga1_obr;      //
odwrócenie stanu flagi1_obr
wskazującej kierunek obrotów
    while(digitalRead(SW2) ==
LOW);    //oczekiwanie na puszc-
zenie przycisku SW2
  }
```

dotyczą obsługi przycisku S2, którego naciśnięcie zmienia stan flagi *flaga1_obr*, od której zależy kierunek obrotów silnika (fizycznie kierunek obrotów silnika jest uzależniony od polaryzacji dwóch linii zasilających dany silnik). W komendach:

```
if (flaga1_on == 1) {         //
Sprawdzenie czy ustawiona jest
flaga1_on, jeśli tak to
    digitalWrite(Motor_1_S,
HIGH);  //załączenie silnika M1
    digitalWrite(Motor_2_S,
HIGH);  //załączenie silnika M2
    digitalWrite(LED1, LOW);  //
załączenie diody Led 1
  }
  else   //w przeciwnym razie
  {
    digitalWrite(Motor_1_S, LOW);
//wyłączenie silnika M1
    digitalWrite(Motor_2_S, LOW);
//wyłączenie silnika M2
    digitalWrite(LED1, HIGH);  //
wyłączenie Led1
  }
```

jest sprawdzany stan flagi *flaga1_on*. Jeśli jest ustawiona, załączane są silniki M1 i M2. Załączana jest również dioda Led1 (wyzerowanie wyjścia sterującego Led1). Przeciwny stan flagi *flaga1_on* spowoduje ustawienie na liniach załączających silniki M1 i M2 poziomu niskiego, co spowoduje ich wyłączenie. Instrukcje

```
  if (flaga1_obr == 1) {
//Sprawdzenie czy ustawiona fla-
ga1_obr, jeśli tak to
```



Rysunek 1. Dołączenie silników prądu stałego do układu L297

Listing 1. Program przykładowy demonstrujący działanie AVTduino Motor

```
/*
Przykład obsługi komponentów, jakie zawiera moduł obsługi silników DC dla Arduino UNO.
Program zawiera przykład konfiguracji i obsługi:
 - sterowanie 4 silnikami
 - regulacja predkości silnikow
 - zmiana kierunku obrotu
Przycisk S1 i S3 umożliwiaja właczenie silników, przyciski S2, S4 zmianę kierunku obrotów, natomiast potencjometry
umożliwiają regulację prędkości.
Moduł steruje 4 silnikami w dwóch grupach, w których każdy obraca się w przeciwnym kierunku.
Dioda Led1 wskazuje włączenie silnika M1, M2, a Led2 silnika M3 i M4
 */

const int Motor_1_A =  12;                //przypisanie aliasow linia portów
const int Motor_1_B =  10;                //przypisanie aliasow linia portów
const int Motor_2_A =  11;                //przypisanie aliasow linia portów
const int Motor_2_B =  9;                 //przypisanie aliasow linia portów
const int Motor_3_A =  3;                 //przypisanie aliasow linia portów
const int Motor_3_B =  5;                 //przypisanie aliasow linia portów
const int Motor_4_A =  6;                 //przypisanie aliasow linia portów
const int Motor_4_B =  4;                 //przypisanie aliasow linia portów
const int Motor_1_S =  13;                //przypisanie aliasow linia portów
const int Motor_2_S =  8;                 //przypisanie aliasow linia portów
const int Motor_3_S =  2;                 //przypisanie aliasow linia portów
const int Motor_4_S =  7;                 //przypisanie aliasow linia portów
const int LED1 =  A2;                     //przypisanie aliasow linia portów
const int LED2 =  A3;                     //przypisanie aliasow linia portów
const int SW1 =  1;                       //przypisanie aliasow linia portów
const int SW2 =  0;                       //przypisanie aliasow linia portów
const int SW3 =  A4;                      //przypisanie aliasow linia portów
const int SW4 =  A5;                      //przypisanie aliasow linia portów
const int Pot1 =  A0;                     //przypisanie aliasow linia portów
const int Pot2 =  A1;                     //przypisanie aliasow linia portów

byte flaga1_on = 0;                       //zmienna flaga
byte flaga1_obr = 0;                      //zmienna flaga
byte flaga2_on = 0;                       //zmienna flaga
byte flaga2_obr = 0;                      //zmienna flaga
int mot1;                                 //zmienna wartości z potencjometru Pot1
int mot2;                                 //zmienna wartości z potencjometru Pot2

void setup()                              //procedura konfiguracyjna
{
  pinMode(Motor_1_A, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_1_A, LOW);
  pinMode(Motor_1_B, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_1_B, LOW);
  pinMode(Motor_2_A, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_2_A, LOW);
  pinMode(Motor_2_B, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_2_B, LOW);
  pinMode(Motor_3_A, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_3_A, LOW);
  pinMode(Motor_3_B, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_3_B, LOW);
  pinMode(Motor_4_A, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_4_A, LOW);
  pinMode(Motor_4_B, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_4_B, LOW);
  pinMode(Motor_1_S, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_1_S, LOW);
  pinMode(Motor_2_B, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_2_S, LOW);
  pinMode(Motor_3_S, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_3_S, LOW);
  pinMode(Motor_4_B, OUTPUT);             //konfiguracja linii
  digitalWrite(Motor_4_S, LOW);
  pinMode(LED1, OUTPUT);                  //konfiguracja Led1
  digitalWrite(LED1, HIGH);
  pinMode(LED2, OUTPUT);                  //konfiguracja Led2
  digitalWrite(LED2, HIGH);
  pinMode(SW1, INPUT);                    //konfiguracja linii do których dołączono przyciski jako wejścia
  pinMode(SW2, INPUT);
  pinMode(SW3, INPUT);                    //konfiguracja linii do których dołączono przyciski jako wejścia
  pinMode(SW4, INPUT);
  digitalWrite(SW1, HIGH);                //dołączenie do linii do których dołączono przyciski rezystorów
                                          //podciągajacych co wymusi na nich domyslnie stan wysoki
  digitalWrite(SW2, HIGH);
  digitalWrite(SW3, HIGH);                //dołączenie do linii do których dołączono przyciski rezystorów
                                          //podciągajacych co wymusi na nich domyślnie stan wysoki
  digitalWrite(SW4, HIGH);

  analogReference(DEFAULT);               //konfiguracja przetwornika A/C
}

void loop()                               //petla główna programu
{
  mot1 = analogRead(A0);                  //odczyt wartości analogowej z Pot1
  delay(10);                              //opóźnienie 10 ms
  mot2 = analogRead(A1);                  //odczyt wartości analogowej z Pot2
  delay(10);                              //opóźnienie 10 ms

  if (digitalRead(SW1) == LOW) {          //Sprawdzenie czy naciśniety przycisk SW1
    flaga1_on=!flaga1_on;                 //odwrocenie stanu flagi flaga1_on wskazujacej załączenie silnika M1 i M2
    while(digitalRead(SW1) == LOW);       //oczekiwanie na puszczenie przycisku SW1
  }
  if (digitalRead(SW2) == LOW) {          //Sprawdzenie czy naciśnięty przycisk SW2
    flaga1_obr=!flaga1_obr;               //odwrócenie stanu flagi1_obr wskazujacej kierunek obrotów
    while(digitalRead(SW2) == LOW);       //oczekiwanie na puszczenie przycisku SW2
  }
  if (flaga1_on == 1) {                   //Sprawdzenie czy ustawiona jest flaga1_on, jesli tak to
    digitalWrite(Motor_1_S, HIGH);        //załączenie silnika M1
    digitalWrite(Motor_2_S, HIGH);        //załączenie silnika M2
```

Listing 1. c.d.

```
    digitalWrite(LED1, LOW);  //zalaczenie diody Led 1
  }
  else   //w przeciwnym razie
  {
    digitalWrite(Motor_1_S, LOW);  //wylaczenie silnika M1
    digitalWrite(Motor_2_S, LOW);  //wylaczenie silnika M2
    digitalWrite(LED1, HIGH);  //wylaczenie Led1
  }
  if (flaga1_obr == 1) {         //Sprawdzenie czy ustawiona flaga1_obr, jesli tak to
    analogWrite(Motor_1_A, (255-(mot1/4)));  //zapisanie wartosci PWM odczytanej z pot1 do M1
    digitalWrite(Motor_1_B, HIGH);  //ustawienie drugiej linii M1
    analogWrite(Motor_2_A, (255-(mot1/4)));  //zapisanie wartosci PWM z Pot1 do M2
    digitalWrite(Motor_2_B, HIGH);  //ustawienie drugiej linii M2
  }
  else   //w przeciwnym razie
  {
    analogWrite(Motor_1_A, mot1/4);  //zapisanie wartosci PWM odczytanej z pot1 do M1
    digitalWrite(Motor_1_B, LOW);    //zerowanie drugiej linii M1
    analogWrite(Motor_2_A, mot1/4);  //zapisanie wartosci PWM odczytanej z pot1 do M2
    digitalWrite(Motor_2_B, LOW);   //zerowanie drugiej linii M2
  }

  if (digitalRead(SW3) == LOW) {         //Sprawdzenie czy nacisniety przycisk SW3
    flaga2_on=!flaga2_on;      //odwrocenie stanu flagi flaga2_on wskazujacej zalaczenie silnika M3 i M4
    while(digitalRead(SW3) == LOW);    //oczekiwanie na puszczenie przycisku SW3
  }
  if (digitalRead(SW4) == LOW) {         //Sprawdzenie czy nacisniety przycisk SW4
    flaga2_obr=!flaga2_obr;       //odwrocenie stanu flagi2_obr wskazujacej kierunek obrotow
    while(digitalRead(SW4) == LOW);    //oczekiwanie na puszczenie przycisku SW4
  }
  if (flaga2_on == 1) {         //Sprawdzenie czy ustawiona jest flaga2_on, jesli tak to
    digitalWrite(Motor_3_S, HIGH); //zalaczenie silnika M3
    digitalWrite(Motor_4_S, HIGH);  //zalaczenie silnika M4
    digitalWrite(LED2, LOW);  //zalaczenie diody Led 2
  }
  else  //w przeciwnym razie
  {
    digitalWrite(Motor_3_S, LOW);  //wylaczenie silnika M1
    digitalWrite(Motor_4_S, LOW);  //wylaczenie silnika M1
    digitalWrite(LED2, HIGH);  //wylaczenie Led1
  }
  if (flaga2_obr == 1) {         //Sprawdzenie czy ustawiona flaga2_obr, jesli tak to
    analogWrite(Motor_3_A, (255-(mot2/4)));  //zapisanie wartosci PWM odczytanej z pot2 do M3
    digitalWrite(Motor_3_B, HIGH);  //ustawienie drugiej linii M3
    analogWrite(Motor_4_A, (255-(mot2/4)));  //zapisanie wartosci PWM odczytanej z pot2 do M4
    digitalWrite(Motor_4_B, HIGH);  //ustawienie drugiej linii M4
  }
  else  //w przeciwnym razie
  {
    analogWrite(Motor_3_A, mot2/4);  //zapisanie wartosci PWM odczytanej z pot1 do M3
    digitalWrite(Motor_3_B, LOW);  //zerowanie drugiej linii M3
    analogWrite(Motor_4_A, mot2/4);  //zapisanie wartosci PWM odczytanej z pot1 do M4
    digitalWrite(Motor_4_B, LOW);  //zerowanie drugiej linii M4
  }
}
```

```
    analogWrite(Motor_1_A, (255-(mot1/4)));  //zapisanie wartości PWM odczytanej z pot1 do M1
    digitalWrite(Motor_1_B, HIGH);  //ustawienie drugiej linii M1
    analogWrite(Motor_2_A, (255-(mot1/4)));  //zapisanie wartości PWM z Pot1 do M2
    digitalWrite(Motor_2_B, HIGH);  //ustawienie drugiej linii M2
  }
  else   //w przeciwnym razie
  {
    analogWrite(Motor_1_A, mot1/4);  //zapisanie wartości PWM odczytanej z pot1 do M1
    digitalWrite(Motor_1_B, LOW);  //zerowanie drugiej linii M1
```

```
    analogWrite(Motor_2_A, mot1/4);  //zapisanie wartości PWM odczytanej z pot1 do M2
    digitalWrite(Motor_2_B, LOW);  //zerowanie drugiej linii M2
  }
```

są wykonywane w zależności od stanu flagi *flaga1_obr* odpowiedzialnej za kierunek obrotów silników. Jeśli flaga jest ustawiona, wykonywane są instrukcje ustawiające jedną linię silników M1 i M2, natomiast na drugą podawany jest sygnał PWM o wypełnieniu od 0 do 100% zależnym od ustawienia potencjometru Pot1. Dzięki temu jest możliwa regulacja potencjometrem prędkości obrotowej silników M1 i M2. Wartości sygnału PWM są ograniczane do zakresu od 0 do 255. Jeśli flaga *flaga1_obr* będzie wyzerowana, wykonywane są instrukcje po klauzuli *else*. Powodują one odwrócenie kierunku obrotu silników poprzez podanie na drugie linie sterujące poziomu niskiego. Pozostałe instrukcje w programie z listingu 1 dotyczą silników M3 i M4, do których sterowania użyto przycisków S3, S4 oraz potencjometru Pot2. Ich działanie jest identyczne, jak w przypadku obsługi silników M1 i M2.

Podsumowanie

Działanie przykładowego programu dla modułu AVTDUINO MOTOR pokazuje możliwości i prostotę sterowania takimi elementami, jak silniki DC. Moduł sterujący AVTduino Motor można zastosować do budowy napędu robotów, zabawek czy inteligentnych pojazdów. Elementy przykładowego programu obsługi można wykorzystać we własnych programach po ewentualnym dostosowaniu ich do projektowanego urządzenia.

Marcin Wiązania
marcin.wiazania@ep.com.pl