Kurs Arduino (5)

Obsługa modułu manipulatora

W EP 6/2011 wśród miniprojektów opisaliśmy moduł joysticka dla AVTduino. Może on posłużyć do budowy manipulatora lub aparatury do zdalnego sterowania modelem czy jakimś urządzeniem. W tym artykule zaprezentujemy sposób, w jaki można wykonać program odczytujący pozycję potencjometrów joysticka, stany przycisków oraz sterujący 3 serwomechanizmami modelarskimi.

Przykładowy program obsługujący moduł zamieszczono na **listingu 1**. Jak wspomniano, program umożliwia sterowanie 3 serwomechanizmami za pomocą joysticka i przycisków. Serwomechanizm nr 1 porusza się przy pochylaniu joysticka w prawo/lewo, serwomechanizm nr 2 – góra/dół, natomiast przyciski SW1 i SW2 umożliwiają sterowanie serwomechanizmem nr 3.

Po naciśnięciu przycisku joysticka następuje zamiana sposobu regulacji położenia serwomechanizmów nr 1 i nr 2. Teraz pochylanie joysticka na boki powoduje działanie serwomechanizmu nr 2, natomiast do przodu i do tyłu – serwomechanizmu nr 1.

Naciśnięcie przycisku SW1 ustawia serwomechanizm nr 3 w pozycji 0° i powoduje zaświecenie się diody LED1. Naciśnięcie przycisku SW2 ustawia serwomechanizm nr 3 w pozycji 180 stopni i jest zaświecana dioda LED2.

W środowisku Arduino do obsługi serwomechanizmów modelarskich jest przeznaczona biblioteka o nazwie *servo.h*. Ma ona możliwość jednoczesnej obsługi do 12 serwomechanizmów, co umożliwia budowę nawet skomplikowanych układów mechanicznych służących np. do poruszania nogami robota kroczącego. Komendy służące do sterowania serwomechanizmami zamieszczono w **tabeli 1**.

Zazwyczaj serwomechanizmy modelarskie mają wbudowane układy elektroniczne, które precyzyjnie ustalają położenie układu mechanicznego z własnym silniczkiem napędowym. Zwykle mogą one być ustawiane pod kątami z zakresu 0...180°. Do regulacji położenia służy wejściowy przebieg PWM.

W przykładowym programie (listingu 1) w pierwszej kolejności są deklarowane 3 zmienne do obsługi serwomechanizmów tj.:

Servo servo1;
Servo servo2;
Servo servo3;

Za pomocą komendy *attach()* w procedurze konfiguracyjnej *setup()* do tych zmiennych zostały przypisane piny mikrokontrolera, do których dołączono serwomechanizmy (wyprowadzenia nr 8, 9 i 10):

servo1.attach(8);
servo2.attach(9);
servo3.attach(10);

W tej procedurze również są konfigurowane parametry portów I/O mikrokontrolera, do których dołączono przyciski oraz diody LED. Porty z dołączonymi przyciskami mają załączone rezystory zasilające ustalające domyślny poziom wysoki na liniach I/O pracujących jako wejściowe. Za pomocą komendy *servo3.write(0)* serwomechanizm nr 3 jest ustawiany w wychyleniu 0°. W pętli głównej programu za pomocą komend:

temp1 = analogRead(A0);
temp2 = analogRead(A1);

jest odczytywana wartość analogowa położenia joysticka składającego się z dwóch potencjometrów, osobnego dla kierunku prawo/lewo i góra/dół. Wartości położenia joysticka

Dodatkowe materiały na CD/FTP:
ftp://ep.com.pl, user: 12040, pass: 15735862
• poprzednie części kursu

Tabela 1. Komedy do obsługi serwomechanizmów modelarskich za pomocą Arduino

Komenda	Opis
attach(pin)	Konfiguruje wyprowadzenie, do którego dołączono serwomechanizm.
write(kąt)	Ustala wychylenie serwomechanizmu o kąt podany jako argument wywołania funkcji.
writeMicroseconds(czas)	Umożliwia sterowanie wychyleniem serwomechanizmu za pomocą czasu jako argumentu (czas µs).
read()	Odczytuje spodziewane położenie serwomechanizmu (zwracany jest kąt).
attached()	Umożliwia sprawdzenie, czy zmienna do obsługi serwomechanizmu jest przypisana do pinu mikrokontrolera.
detach()	Odłącza zmienną do obsługi serwomechanizmu od przypisanego pinu mikrokontrolera.

są zapisywane do zmiennych *temp1* i *temp2*. Następnie za pomocą komend:

temp1 = map(temp1, 0, 1023, 0, 179);

temp2 = map(temp2, 0, 1023, 0, 179);

wartości odczytane z joysticka z zakresu od 0 do 1023 są przeliczane na położenia kątowe z zakresu 0...179°, które będą ustalać wychylenia serwomechanizmów. W dalszej części programu, jeśli zmienna *flaga* jest równa 0, wykonywane są instrukcje:

servo1.write(temp1);

servo2.write(temp2);.

Jeśli zmienna *flaga* będzie różna od 0, do serwomechanizmu 1 zostanie wpisana wartość z *temp2*, a do serwa 2 z *temp1*. W ten sposób jest realizowana funkcjonalność zmiany sposobu działania joysticka, o której pisano wcześniej.

Wartościami podawanymi jako argumenty funkcji ustalających położenia serwomechanizmów są wartości liczbowe kątów ich położenia. Instrukcje:

```
if (digitalRead(SWJ) == LOW)
{
 flaga=!flaga;
 while(digitalRead(SWJ) == LOW
}
```

obsługują przycisk joysticka i powodują po każdym jego naciśnięciu zmianę stanu zmiennej *flaga* na odwrotny. Ostatnia instrukcja powoduje oczekiwanie na puszczenie przycisku joysticka.

Kolejne instrukcje dotyczą obsługi przycisku SW1. Jego naciśnięcie powoduje zapalenie diody LED1, ustawienie serwomechanizmu nr 3 w pozycji 0° i zgaszenie LED2. Natomiast przyciśnięcie przycisku SW2 spowoduje wykonanie instrukcji zaświecających LED2, ustawiających położenie serwomechanizmu nr 3 na kąt 180° i zgaszenie LED1. Więcej informacji o komendach dostępnych w bibliotece *servo.h* można znaleźć w kodach źródłowych samych bibliotek.

Podsumowanie

Działanie przykładowego programu dla modułu AVTduino JOY pokazuje możliwości i prostotę obsługi takich elementów, jak joystick czy serwomechanizm. Za jego pomocą można wykonać sterujące robotami czy kamerami z możliwością obracania. Elementy przykładowego programu obsługi elementów modułu JOY niewątpliwie będzie można wykorzystać we własnych projektach.

Marcin Wiązania
marcin.wiazania@ep.com.pl

Listing 1. Przykładowy program obsługi modułu AVTduino JOY

```
/*  Przykład obsługi komponentów, jakie zawiera moduł sterowania serw
joystickiem. Program zawiera przykład konfigurowania i obsługi:
 - joysticka sterującego dwoma serwami,
 - przycisk SW1 zapala LED1 oraz ustawia serwo 3 w pozycji 0°,
 - przycisk SW2 zapala LED2 oraz ustawia serwo 3 w pozycji 180°.
 Przycisk joysticka zamienia działanie serwa 1 i 2   */
#include <Servo.h> //biblioteka obsługi serw

Servo servo1;  // objekt serwo 1
Servo servo2;  // objekt serwo 2
Servo servo3;  // objekt serwo 3

const int Led1 = 6;  //przypisanie aliasów do pinów portów
const int Led2 = 5;
const int SW1 = 3;
const int SW2 = 4;
const int SWJ = 2;
int temp1;  // zmienne pomocnicze
int temp2;
byte flaga = 0;  //zmienna flaga

void setup() //pętla konfiguracyjna programu
{
  servo1.attach(8);  // konfigurowanie pinu serwa 1
  servo2.attach(9);  // konfigurowanie pinu dla serwa 2
  servo3.attach(10);  // konfigurowanie pinu dla serwa 3
  pinMode(SW1, INPUT);  //konfigurowanie linii, do których są dołączone przyciski jako wejścia
  pinMode(SW2, INPUT);
  pinMode(SWJ, INPUT);
  digitalWrite(SW1, HIGH);  //dołączenie do linii, do których są dołączone przyciski rezystorów podciągających co
wymusi na nich domyślnie stan wysoki
  digitalWrite(SW2, HIGH);
  digitalWrite(SWJ, HIGH);
  pinMode(Led1, OUTPUT);  //konfigurowanie linii, do których są dołączone diody jako wyjścia
  pinMode(Led2, OUTPUT);
  digitalWrite(Led1, LOW);  //domyślne ustawienie stanu diod LED
  digitalWrite(Led2, HIGH);
  servo3.write(0);  //ustawienie serwa 3 w pozycji 0 stopni
}

void loop()   //petla glowna programu
{
  temp1 = analogRead(A0);  //odczyt napięcia z joysticka lewo/prawo
  temp2 = analogRead(A1);  //odczyt napięcia z joysticka góra/dół
  temp1 = map(temp1, 0, 1023, 0, 179);  //zamiana napięcia na kąt 0 do 180°
  temp2 = map(temp2, 0, 1023, 0, 179);  //zamiana napięcia na kąt 0 do 180°
  if (flaga==0)  //jeśli zmienna flaga równa 0 to
  {
    servo1.write(temp1);  //ustawienie serwa 1 wartością w temp1
    servo2.write(temp2);  //ustawienie serwa 2 wartością w temp2
  }
  else  //w przeciwnym razie
  {
    servo1.write(temp2);  //ustawienie serwa 1 wartością w temp2
    servo2.write(temp1);  //ustawienie serwa 2 wartością w temp1
  }

  delay(15); //opóźnienie 15 ms

 if (digitalRead(SWJ) == LOW) {  //sprawdzenie, czy naciśniety przycisk SWJ
    flaga=!flaga;  //odwrócenie stanu flagi flaga
    while(digitalRead(SWJ) == LOW);  //oczekiwanie na zwolnienie przycisku S1
  }

  if (digitalRead(SW1) == LOW) {  //sprawdzenie, czy naciśniety przycisk S1
    digitalWrite(Led1, LOW);  //zaświecenie LED1
    digitalWrite(Led2, HIGH);  //zgaszenie LED2
    servo3.write(0);  //ustawienie pozycji serwa 3
    while(digitalRead(SW1) == LOW);  //oczekiwanie na zwolnienie przycisku S1
  }
  if (digitalRead(SW2) == LOW) {  //sprawdzenie, czy naciśnięty przycisk S2
    digitalWrite(Led1, HIGH);  //zgaszenie LED1
    digitalWrite(Led2, LOW);  //zaświecenie LED2
    servo3.write(180);  // ustawienie pozycji serwa 3
    while(digitalRead(SW2) == LOW);  //oczekiwanie na zwolnienie przycisku S1
  }
}
```