Temat: Sposoby wykorzystania serwomechanizmu.

Serwomechanizm to silnik, przekładnia oraz dedykowany sterownika zamknięty w jednej obudowie. Napędy te nie są jednak przystosowane do wykonywania pełnego obrotu. Najczęściej serwomechanizmy mogą poruszać zamontowanym ramieniem o kąt **0-180º**.

Jak działa serwomechanizm?

Skąd serwomechanizm wie, w którą pozycje ma się obrócić? Wszystko za sprawą wbudowanego sterownika. To własnie on, na podstawie dostarczonego **sygnału PWM**, steruje silnikiem. Przyjętym standardem jest, że do serw dostarcza się sygnał o **okresie równym 20ms.** Natomiast wypełnienie sygnału interpretowane jest jako pozycja, w którą należy przemieścić ramię serwa

Z każdego serwomechanizmu wyprowadzone są **3 przewody:**

1. Masa (**czarny**, **ciemnobrązowy**)
2. Zasilanie (**czerwony**)
3. Sygnał sterujący (**żółty/pomarańczowy**)

W zależności od producenta kolory przewodów mogą się różnić. Jednak dwa na pewno będą zbliżone do czarnego i czerwonego (zasilanie). Pozostały, trzeci będzie przewodem sygnałowym.

Wejście stabilizatora łączymy z pinem *Vin* Arduino, masę z GND, a do wyjścia podłączamy czerwony przewód serwomechanizmu. Oczywiście konieczne są również kondensatory filtrujące. Pozostałe połączenia powinny być już jasne:

Pora na program, który będzie stopniowo poruszał serwem. Na początek gotowy program, poniżej znajdziesz wyjaśnienie:

```
1  #include <Servo.h> //Biblioteka odpowiedzialna za serwa
2
3  Servo serwomechanizm;  //Tworzymy obiekt, dzięki któremu możemy odwołać się do serwa
4  int pozycja = 0; //Aktualna pozycja serwa 0-180
5  int zmiana = 6; //Co ile ma się zmieniać pozycja serwa?
6
7  void setup()
8  {
9    serwomechanizm.attach(9);  //Serwomechanizm podłączony do pinu 9
10 }
11
12 void loop()
13 {
14   if (pozycja < 180) { //Jeśli pozycja mieści się w zakresie
15     serwomechanizm.write(pozycja); //Wykonaj ruch
```

```
16   } else { //Jeśli nie, to powrót na początek
17     pozycja = 0;
18   }
19
20   pozycja = pozycja + zmiana; //Zwiększenie aktualnej pozycji serwa
21   delay(200); //Opóźnienie dla lepszego efektu
22 }
```

Tym razem musimy dodać nową bibliotekę, która rozszerzy możliwości naszego programu o nasze funkcje. Służy do tego polecenie:

```
1 #include Servo.h
```

W tym wypadku dodaliśmy plik *Servo.h*, który zawiera dodatkowe instrukcje dla serwomechanizów. Dzięki niej nie będziemy musieli samodzielnie kontrolować generowanego sygnału PWM. Wystarczy, że podamy pozycje (kąt) do jakiej ma obrócić się serwomechanizm.

Jeżeli chcemy sterować serwem, to musimy stworzyć dla niego obiekt:

```
1 Servo serwomechanizm;
```

Funkcja **attach(Pin)** - dla obiektu *Servo* - działa podobnie do **pinMode** - argumentem jest pin, do którego podłączony jest element. Od tego momentu na danym wyprowadzeniu (w tym przypadku *9*) będzie generowany sygnał PWM.

Po uruchomieniu programu serwomechanizm powinien płynnie poruszać się z jednej skrajnej pozycji do drugiej, a następnie wracać na początek. Kluczową linijką jest:

```
1 serwomechanizm.write(pozycja);
```

Gdzie jako pozycję musimy wpisać kąt z zakresu 0-180º.

Ćwiczenie:

Wykorzystując własne moduły Arduino proszę wykonać powyższe zadanie. (dodatkowe materiały na stronie:

https://forbot.pl/blog/kurs-arduino-podstawy-programowania-spis-tresci-kursu-id5290