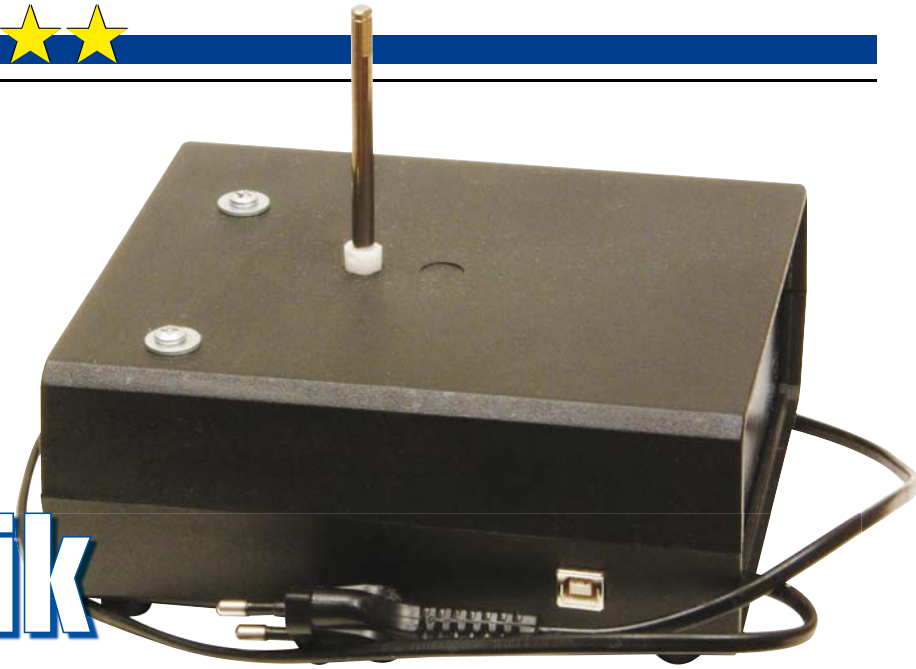
kit 2933 AVT



Sterownik silnika krokowego USB

Do czego to służy?

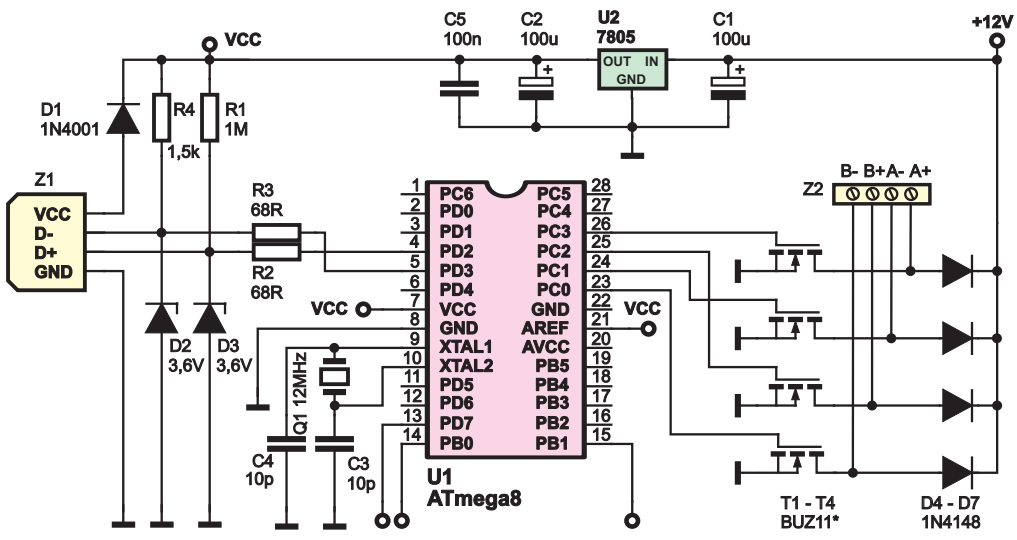
Na łamach EdW wielokrotnie publikowane były projekty różnego rodzaju sterowników silników krokowych. Co wyróżnia spośród nich poniższy projekt? Otóż sterowanie silnikiem odbywa się z poziomu komputera za pomocą portu USB. Wielu Czytelników zapewne pomyślało w tym miejscu, że potrzebne będą specjalne drogie i/lub trudniejsze do zdobycia i wlutowania układy. Nic bardziej mylnego! Cały układ opiera się na znanym i lubianym mikrokontrolerze ATmega8. Sterownik pierwotnie miał pełnić funkcje zdalnego obracania anteną Wi-Fi, jednak może być on łatwo przystosowany do innych zadań, gdzie konieczne jest sterowanie przez komputer. Pomysł sterowania przez USB wziął się z tego, że autor w swym komputerze nie posiada portów LPT i COM. Jeżeli, Drogi Czytelniku, zainteresował Cię ten projekt, to zapraszam do dalszej lektury.

Jak to działa?

Schemat ideowy układu został przedstawiony na **rysunku 1.** Rezystory R2 i R3 stanowią dopasowanie impedancji do linii USB. Napięcie Vcc dostępne poprzez złącze USB wynosi 5V, poziomy logiczne magistrali USB są niższe. Aby odpowiednio dopasować napięcia, użyto diod Zenera 3,6V. Rezystor R4 służy do tego, żeby komputer mógł wykryć, że dołączono urządzenie (jeśli komputer „zobaczy" oporność 1,5kΩ na tej linii to „wie", że musi rozpocząć komunikację z urządzeniem). Układ ten przystosowany jest do sterowania czterofazowych silników krokowych unipolarnych. Ponieważ idea sterowania tych silników była już wielokrotnie omawiana, ciekawych Czytelników odsyłam do serii artykułów, jakie ukazały się w EdW (pierwszy z nich w lipcu 2002). Jako elementy sterujące zostały zastosowane tranzystory T1–T4. Napięcie 5V występujące na pinach mikrokontrolera jest wystarczające, by tranzystory BUZ11 przewodziły, jednak nie będą one mogły przewodzić swego nominalnego prądu. Do podłączenia silników o niewielkim poborze mocy takie rozwiązanie wystarczy, jednak w celu poprawienia parametrów zamiast tranzystorów BUZ11 należałoby wlutować tranzystory sterowane poziomami logicznymi (zwykle oznaczane są one literką L). Diody D4–D7 chronią układ przed przepięciami powstałymi w uzwojeniach silnika. Przejdźmy teraz do „mózgu" układu, jakim jest mikrokontroler ATmega8. Jest on taktowany sygnałem o częstotliwości 12MHz z zewnętrznego rezonatora. Pewnego wyjaśnienia może wymagać sposób podłączenia silnika krokowego do układu. Jeżeli zastosujemy takie połączenie, jak na schemacie, to w najprostszym przypadku obrót silnika sprowadza się do przesuwania rejestru czterobitowego. Oprócz obsługi silnika krokowego do prezentowanego układu może być dołączony serwomechanizm. Aby regulować jego wychylenie, mikrokontroler musi wygenerować dodatni impuls o czasie trwania od 1ms do 2ms z częstością 50Hz. Został do tego wykorzystany Timer1 pracujący w trybie PWM.

Przejdźmy teraz do najciekawszej części, czyli do omówienia komunikacji urządzenia z komputerem. Została ona uzyskana programowo. W projekcie tym wykorzystano darmową bibliotekę opracowaną przez firmę Objective Development. Za jej pomocą obsługa USB sprowadza się do napisania kilku prostych

Rys. 1 Schemat ideowy



funkcji oraz zdefiniowania nazwy, typu i identyfikatora urządzenia oraz podania pinów, do których są podłączone linie USB. Najnowsza wersja biblioteki obsługuje kilka częstości taktowania oraz układy taktowane z wewnętrznego rezonatora RC. Jedyne wymaganie sprzętowe to podłączenie jednej z linii do wejścia przerwania zewnętrznego. Biblioteka ta wykorzystuje przerwania od Timera0, a cały jej kod zajmuje niecałe 2kB pamięci flash i 100B RAM. Aby uświadomić Czytelnikom, jak wielkie wyzwanie stało przed programistami, nadmienię, że na odkodowanie pojedynczego bitu przypada 8 taktów zegara, a całość jest realizowana w trybie rzeczywistym. Co prawda w tym projekcie korzystamy z gotowej biblioteki, jednak warto się zapoznać z podstawowym schematem działania złącza USB. Jeżeli ktoś się przyglądał kiedyś wtykowi USB, to z pewnością wie, że ma on cztery pola połączeniowe, oznaczone Vcc, GND, D+ i D–. Nietrudno się domyślić, do czego służą przewody Vcc i GND, pozostałe dwa to przewody sygnałowe. Aby zwiększyć podatność systemu na zakłócenia, dane są przesyłane w sposób różnicowy. Znaczy to, że gdy na linii D+ jest stan niski, to na D– jest wysoki itd. Takie rozwiązanie ma tę wadę, że niemożliwe jest jednoczesne wysyłanie i odbieranie danych. Z tego powodu kolejność wysyłania danych przez komputer i sposobu odpowiedzi na te dane przez jakiekolwiek urządzenie jest ściśle określona. Z tego powodu, aby dane urządzenie mogło działać poprawnie, niezbędna jest wcześniejsza instalacja sterowników. Dla niektórych typów urządzeń protokół transmisji danych jest uniwersalny i nie wymagają one zewnętrznych sterowników (np. pendrive'y). Aby odróżnić jedne urządzenia od drugich, wprowadzono pojęcie klasy. Każde urządzenie po podłączeniu do komputera „przedstawia się", czyli wysyła między innymi swoją nazwę, numery identyfikacyjne i klasę (wymienione wcześniej pendrive'y to urządzenia klasy magazynującej). Nasze urządzenie ma klasę 0xFF – znaczy to, że wykonawca urządzenia musi dostarczyć swoje własne sterowniki.

Listing 1

```
//--------------------------- Definicje instrukcji wysyłanych do uC

#define INS_TEST 0                      //test połączenia
#define INS_USTAW_WYJ 1 //bezpośrednie ustawienie wyjść
#define INS_ODCZYT_SENS 2       //odczyt z czujników (sensorów)
#define INS_TRYB 3                      // 0 - ciągły obrót; 1 - obrót o określony kąt
#define INS_STEROWANIE 4        // 0 - pełnokrokowy; 1 - półkrokowy; 2 - falowy
#define INS_KIERUNEK 5          //kierunek obrotu
#define INS_OPOZ 6                      //opóźnienie między wykonaniem kroków *100us
#define INS_KROKI 7             //ilość kroków na pełen obrót
#define INS_KAT 8                       //kąt obrotu
#define INS_SERVO 9             //ustawia serwo; wartości od 1500 do 3000
```

Listing 2

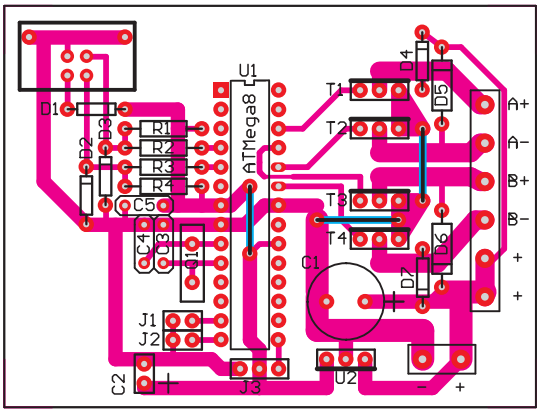
Komputer:

```
wyslij_instrukcje (INS_SERVO, 2000, handle);
```

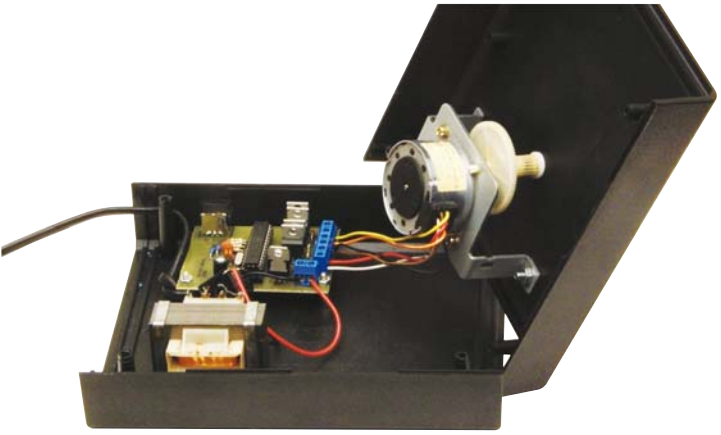
Mikrokontroler:

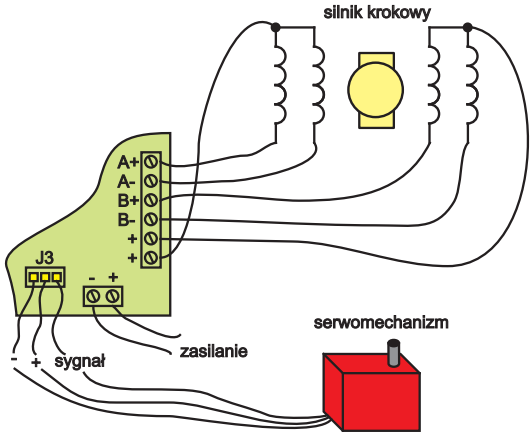
```
uchar   usbFunctionSetup(uchar wej_bufor_usb[8]) //funkcja obsługująca USB
{
…
        else if(wej_bufor_usb[1] == INS_SERVO)
        {
                ...
                OCR1A = wej_bufor_usb[2] | (wej_bufor_usb[3] << 8);
                wyj_bufor_usb[0] = 0;
        }
…
}
```

Program na komputerze wymaga posługiwania się biblioteką libusb (dla środowiska Windows dostępna pod adresem *http://libusb-win32.sourceforge.net/*). Ponieważ sterownik silnika nie wymaga, by przesyłano do niego wielkich ilości danych, to do komunikacji wykorzystujemy tak zwane wiadomości kontrolne. Zanim jednak wyślemy wiadomość do naszego urządzenia, musimy uzyskać wskaźnik do tego urządzenia. Do tego celu wykorzystujemy funkcję *nawiaz_kontakt()*. Po uzyskaniu wskaźnika możemy już wysyłać instrukcje, służy do tego funkcja: *wyslij_instrukcje()*. Pierwszym jej argumentem jest numer instrukcji, drugim jest zmienna wysyłana wraz z instrukcją, a trzeci to wskaźnik do naszego urządzenia. Znaczenie dodatkowej wysyłanej zmiennej zależy od typu instrukcji. Jeżeli chcemy zmienić wartość opóźnienia pomiędzy kolejnymi krokami, to dodatkowa zmienna będzie miała wartość tego opóźnienia. Pełna lista instrukcji znajduje się na **listingu 1**. Po stronie mikrokontrolera obsługa portu USB sprowadza się do napisania jednej funkcji, która interpretuje odebrane dane. Fragment kodu odpowiedzialny za zmianę



Rys. 2 Schemat montażowy





Rys. 3 Schemat połączeń płytki z silnikiem

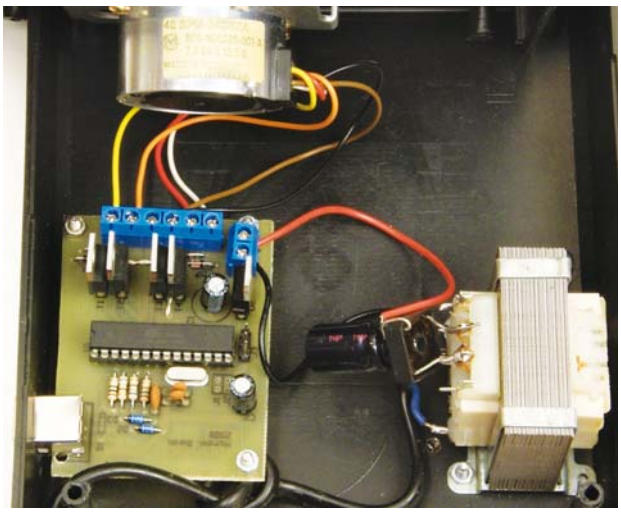
wychylenia serwomechanizmu prezentowany jest na **listingu 2**. Wyjaśnienia wymaga tablica *wej_bufor_usb[]*. Pierwszy jej element to numer instrukcji, następne dwa przechowują wysłaną do ATmegi zmienną, aby odtworzyć jej wartość, należy wykonać prostą operację bitową, tak jak w przykładzie.

Montaż i uruchomienie

Na **rysunku 2** pokazany jest schemat montażowy. Prezentowany układ nie zawiera dużej liczby elementów, więc jego montaż nie powinien stwarzać trudności. Należy najpierw zamontować elementy najmniejsze, a na końcu tranzystory i złącza ARK. Aby umożliwić sterowanie silnikami o różnym napięciu znamionowym, w układzie zastosowano stabilizator napięcia. W takim wypadku można nie montować diody D1. Gdy przewidywany pobór prądu będzie niewielki i napięcie 5V jest wystarczające, to cały układ można zasilać z komputera poprzez złącze USB. Wtedy zamiast stabilizatora należy wlutować zworę między jego skrajne wyprowadzenia. Do montażu mikrokontrolera warto zastosować podstawkę, co umożliwi późniejsze zmiany oprogramowania. Pewne trudności może sprawić podłączenie silnika krokowego – zwłaszcza gdy pochodzi on z demontażu i nie znamy jego wyprowadzeń. Warto za pomocą omomierza znaleźć „środkowe odczepy" uzwojeń silnika i połączyć je z plusem zasilania. Resztę przewodów zwykle trzeba łączyć metodą prób i błędów. Schemat połączenia układu z silnikiem pokazany jest na **rysunku 3**. Układ ten powinien działać od razu po włączeniu, jednak wymaga on zainstalowania sterowników po podłączeniu do komputera.

Instalacja urządzenia jest dosyć prosta, jeżeli ktoś instalował jakiekolwiek urządzenie USB, to nie będzie miał z tym problemów. W systemie Windows XP po włożeniu wtyczki od urządzenia do portu USB komputera powinien wyświetlić się kreator znajdowania nowego sprzętu. Należy polecić, by system nie łączył się z witryną Windows. Następnie należy wybrać opcję, by komputer zainstalował urządzenie z listy lub określonej lokalizacji i w następnym okienku podać lokalizację sterowników. Po kliknięciu na *Dalej* system operacyjny powinien zainstalować urządzenie. Jeśli wykonaliśmy wszystkie te czynności, urządzenie jest gotowe do pracy. Wraz z projektem dołączonych jest kilka prostych programów, umożliwiających dostosowanie urządzenia do swoich potrzeb – można je ściągnąć z Elportalu.

Marcin Bieda
gudilol@gmail.com



Wykaz elementów

Rezystory
R1 ..... 1MΩ
R2,R3 ..... 68Ω
R4 ..... 1,5kΩ

Kondensatory
C1,C2 ..... 100µF/25V
C3,C4 ..... 10pF
C5 ..... 100nF

Półprzewodniki
D1 ..... 1N4007
D2,D3 ..... diody Zenera 3,6V
D4-D7 ..... 1N4148
T1-T4 ..... BUZ11 (patrz opis w tekście)
U1 ..... ATmega8
U2 7805

Inne
Q1 ..... rezonator kwarcowy 12MHz
Z1 ..... złącze USB-B
Z2 ..... złącza ARK

Komplet podzespołów z płytką jest dostępny w sieci handlowej AVT jako kit szkolny AVT-2933.