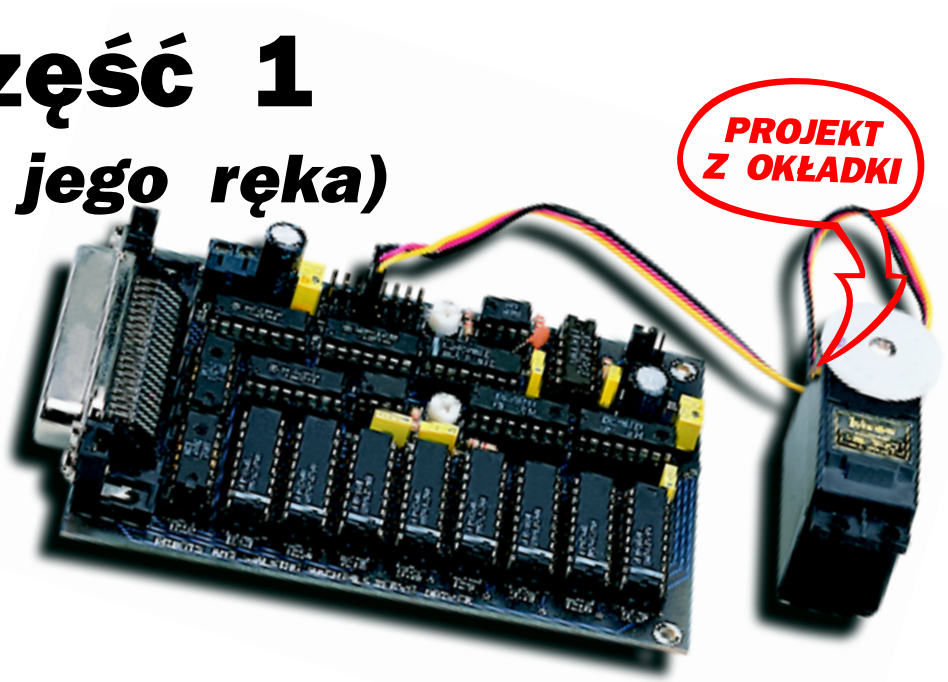
Robot, część 1

(a właściwie jego ręka)

AVT-821

Urządzenia, które w tym miesiącu zdobi naszą okładkę jest niestandardowe i niebanalne. Jak zapewnia autor opracowania, jego robot może służyć nie tylko do nalewania piwa...



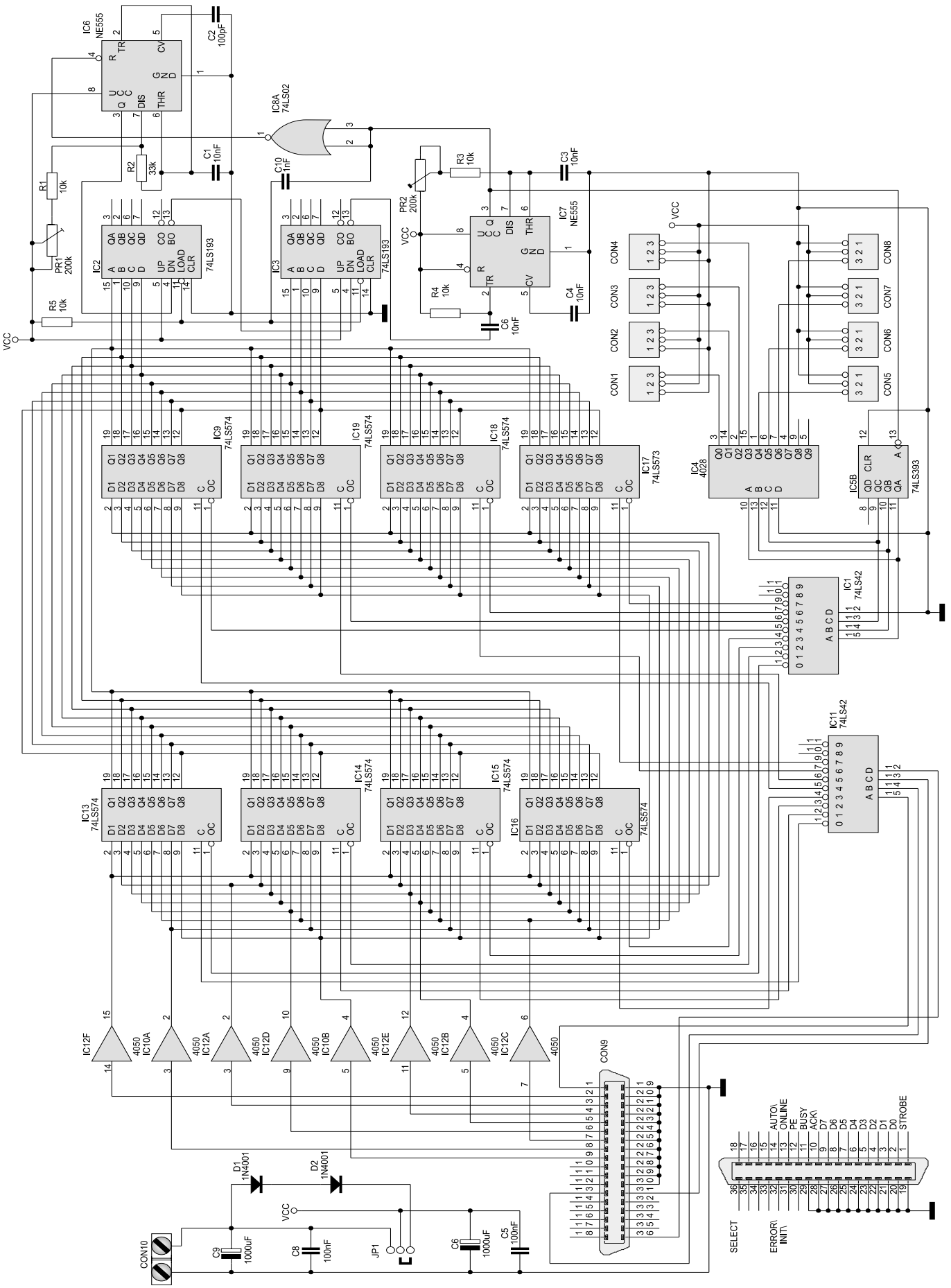
Układ, który za chwilę wykonamy, trzeba będzie wypróbować praktycznie na działającym modelu. I tu pozwolę sobie zaproponować Wam coś zupełnie nowego, coś, czym nie zajmowało się jeszcze żadne pismo dla elektroników wychodzące w Polsce. Coś z „Gwiezdnych wojen", coś bardzo atrakcyjnego dla miłośników science fiction. Z pewnością pamiętacie znakomity film „Saturn 3", w którym główną rolę zagrał właściwie nie Kirk Douglas, ale znakomicie skonstruowany i przystosowany do potrzeb filmu robot. Być może wielu z Was oglądało bardzo sympatyczny film „Short Circuit", w którym także głównym bohaterem był perfekcyjnie zbudowany, uroczy robocik.

Z tym robotem to trochę przesadziłem: tak naprawdę, to zbudujemy tylko jego najważniejszą część: rękę o kilku stopniach swobody zaopatrzoną w „łapę" mogącą chwytać i przytrzymywać z dużą siłą najróżniejsze przedmioty. Do wykonania ręki robota użyjemy oczywiście serwomechanizmów modelarskich. No właśnie, sądzę, że wielu Czytelników z przerażeniem myśli o czekających ich pracach mechanicznych, o przycinaniu i wyginaniu blaszanych kształtek i o innych tego typu okropieństwach. Mylicie się, moi Drodzy! Do wykonania ręki robota będziecie potrzebować w zasadzie tylko lutownicy, garstki śrubek M3 z nakrętkami i śrubokręta! Wszystkie potrzebne elementy mechaniczne otrzymacie gotowe, w postaci dość nietypowego kitu zawierającego płytki laminatu zastosowane do budowy jako główne elementy ręki robota. Tak, to nie pomyłka w druku, z laminatu epoksydowo - szklanego możemy wykonać kształtki, które posłużą nam do budowy dość skomplikowanej maszyny! Laminat posiada wystarczającą odporność mechaniczną, a wykonane z niego elementy możemy łatwo łączyć ze sobą za pomocą lutowania. Przygotowałem zestaw gotowych płytek, z którego będziecie mogli wykonać ramię robota o czterech stopniach swobody ruchu, co wydaje się być zupełnie wystarczające w większości konstrukcji amatorskich.

A teraz przysłowiowy „kubeł zimnej wody": porywamy się na coś, na czym niejeden zespół konstruktorski połamał sobie zęby. Będziemy naśladować naturę lub Siłę Wyższą, której istoty nie znamy. Nie sądzę, aby przy obecnym stanie techniki było możliwe skonstruowanie czegoś, co chociaż w przybliżeniu mogłoby naśladować precyzję i wielofunkcyjność ręki człowieka. Jednak do wykonywania prostych czynności nasze urządzenie będzie się nadawać bardzo dobrze, a ponadto jeszcze raz powtarzam: chodzi tu przede wszystkim o naprawdę wyśmienitą zabawę.

Opis działania

Schemat elektryczny komputerowego sterownika ośmiu serwomechanizmów pokazano na **rys. 1**. Zanim jednak przejdziemy do jego analizy, uściślijmy jakie zadania ma wykonywać zbudowany układ.



Rys. 1. Schemat elektryczny "ręki" robota.

Podstawową funkcją urządzenia jest wysyłanie na wejścia ośmiu serwomechanizmów impulsów dodatnich o czasie trwania od ok. 0,5 do ok. 2,5ms. Czas ten powinien być regulowany w podanym zakresie za pomocą poleceń wydawanych z klawiatury komputera. Układ, a właściwie współpracujące z nim oprogramowanie, musi umożliwiać zapamiętanie w dowolnym momencie czasu trwania wszystkich impulsów aktualnie wysyłanych do serw i zapisania informacji o tym na dysku.

Jak łatwo zauważyć przyglądając się schematowi, układ można podzielić na dwie części: jedna z nich, przeznaczona do współpracy z komputerem, oddzielona jest od części wykonawczej blokiem pamięci zbudowanej z ośmiu buforów typu 74LS574. Takie rozwiązanie pozwoliło na odciążenie komputera i pozostawienie mu czasu na wykonywanie operacji nie związanych bezpośrednio ze sterowaniem serwami. Zajmijmy się najpierw blokiem przeznaczony do współpracy z komputerem.

Wysyłanie do układu informacji o wymaganych aktualnych położeniach serwomechanizmów odbywa się magistralą danych interfejsu CENTRONICS, natomiast rejestr dwukierunkowy tego interfejsu ma za zadanie sterować zapisem danych do właściwych komórek pamięci. Informacja o aktualnym położeniu każdego z serwomechanizmów przechowywana jest w ośmiu buforach IC9, IC13..IC19. W momencie, kiedy komputer otrzyma sygnał (podany z klawiatury lub w inny sposób), że zawartość jednego z buforów ma zostać zmieniona, muszą zostać wykonane następujące czynności:

1. Na magistralę danych interfejsu CENTRONICS musi zostać wysłana właściwa liczba, zawierająca się w zakresie od 0 do 255.

2. Na wyjściu SELECT rejestru dwukierunkowego utrzymywany jest zwykle stan wysoki i w związku z tym na wszystkich wejściach zegarowych buforów 74LS574 wymuszony jest stan wysoki.

W momencie zapisu danych komputer wysyła do tego rejestru liczbę będącą adresem odpowiedniego bufora, a następnie ponownie ustawia na wyjściu SELECT stan wysoki. Efektem wykonania tych czynności będzie powstanie na wejściu zegarowym właściwego rejestru krótkiego impulsu ujemnego, który spowoduje przepisanie danych z interfejsu CENTRONICS do tego bufora.

Są to w zasadzie wszystkie czynności, jakie musi wykonywać komputer podczas obsługi naszego sterownika.

Już na samym początku projektowania tego układu napotkałem na poważny problem, który na szczęście udało mi się w prosty sposób rozwiązać. Jak wiemy z lektury poprzednich artykułów opisujących układy przeznaczone do sterowania serwomechanizmami, na wejście sterujące serwa muszą być dostarczane impulsy prostokątne o czasie trwania od 1 do 2 ms. Wiemy także, że są to wartości przyjęte w standardzie stosowanym w radiomodelarstwie i powodują zmianę kąta ustawienia wału napędowego serwa o 90^O lub 60^O (w zależności od typu serwomechanizmu). Pamiętamy także, że z łatwością możemy „zmusić“ standardowy serwomechanizm do obrotu o kąt znacznie większy od typowego, i że jego wartość może nawet przekroczyć 180^O. Takie serwo musimy sterować impulsami o czasie trwania od ok. 0,5 do ok. 3 ms. W opisywanym obecnie układzie informacja o aktualnym położeniu każdego serwa przekazywana jest z komputera cyfrowo, w postaci słowa ośmiobitowego. Załóżmy więc, że impulsowi o szerokości 2 ms odpowiada wartość 255. A zatem zakładając, że stosujemy proste, liniowe przetwarzanie cyfrowo - analogowe, aby uzyskać impuls o czasie trwania 1 ms powinniśmy na magistralę danych wysłać liczbę równą 127, uzyskując w ten sposób raster równy 128 kroków, co w wielu wypadkach może okazać się zbyt małą wartością, uniemożliwiającą precyzyjne pozycjonowanie serwomechanizmów. W przypadku sterowania impulsami o mniejszej i większej długości od przyjętej w aparaturach RC, sytuacja byłaby nieco lepsza, ale i tak liczba z zakresu 0..58 byłaby „marnowana“, ponieważ wartości te reprezentowałyby czas impulsu krótszy od najkrótszego stosowanego do sterowania serwem.

Rozwiązanie tego problemu okazało się stosunkowo proste. W przyjętym przeze mnie rozwiązaniu impuls wyjściowy składa się jakby z dwóch części: jednej o czasie trwania 1 ms (lub 0,5 ms przy niestandardowym sterowaniu serwa) i drugiej, zmienianej w zakresie 0..1ms (0..2ms), której czas trwania określony jest słowem ośmiobitowym wysyłanym przez komputer na magistralę danych. A zatem, w każdym przypadku raster wynosić będzie 255 kroków, co umożliwia bardzo precyzyjne pozycjonowanie wału napędowego serwomechanizmu. Część układu odpowiedzialna za bezpośrednie sterowanie serwami pracuje w pętli, tak że dość trudno znaleźć właściwy moment do rozpoczęcia analizy układu. Rozpocznijmy więc może od momentu, w którym układ IC7 - NE555 wygenerował jeden z impulsów „wyrównujących“ (z przedziału od ok. 0,5 do 1 ms) o czasie trwania określonym rezystancją R3+PR2 i pojemnością C3. Opadające zbocze impulsu wygenerowanego przez IC7 powoduje wystąpienie następujących zjawisk:

1. Zmianę stanu licznika binarnego IC5B o 1. Załóżmy, że na wyjściu tego licznika mamy stan 0000. Wyjścia licznika IC7 dołączone są do wejść dwóch dekoderów BCD - 1 z 10, IC11 - 74LS42 i IC4 - 4028. A zatem, na wyjściu „0“ dekodera IC11 pojawia się w tym momencie stan niski, co powoduje, że na wyjściach bufora IC13, będących do tej pory w stanie wysokiej impedancji, pojawiają się dane wpisane do niego uprzednio przez komputer. Dane te przekazywane są na wejścia ustawiające rewersyjnych liczników binarnych IC2 i IC3 - 40193.

Na wyjściu 3 złącza CON1, do którego dołączony jest pierwszy serwomechanizm, pojawia się stan wysoki przekazywany z wyjścia Q0 dekodera IC4, co dla układu elektronicznego wbudowanego w serwo jest sygnałem o rozpoczęciu generacji impulsu sterującego.

2. Opadające zbocze sygnału na wyjściu Q układu IC7 powoduje wygenerowanie za pośrednictwem kondensatora C10 krótkiego impulsu ujemnego na wejściach ładowania (LOAD) liczników IC2 i IC3. Od tego momentu w rejestrach tych liczników znajduje się ośmiobitowa liczba, przekazana tam z aktualnie aktywnego bufora IC13.

Dane techniczne standardowego serwomechanizmu firmy HITEC typu HS300:

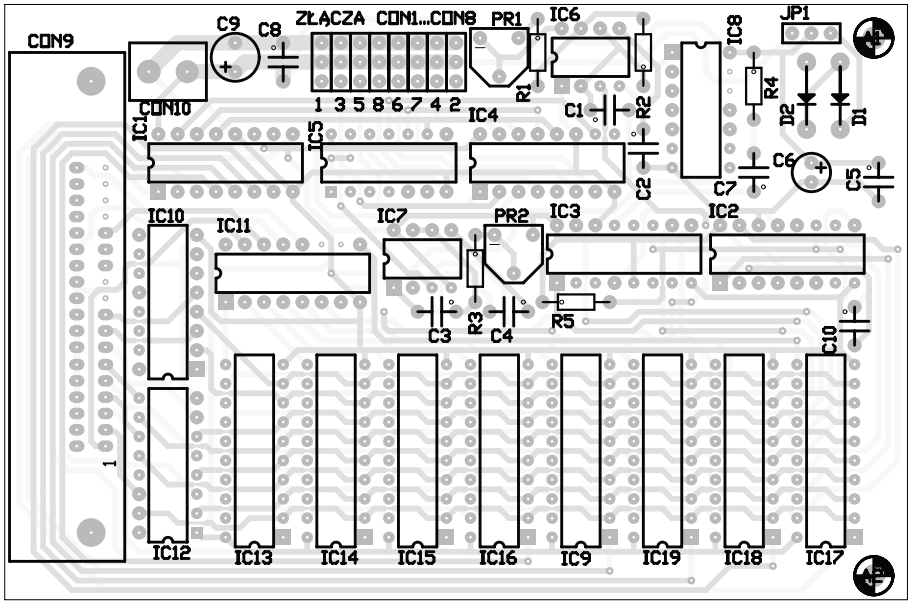
Napięcie zasilania: typowe, 4,8..6VDC;
Kąt obrotu przy sterowaniu typowymi impulsami (1..2ms): 60^O;
Kąt obrotu przy sterowaniu impulsami 0,5..3ms: 190^O.

3. Kolejnym następstwem pojawienia się stanu niskiego na wyjściu układu IC7 (multiwibratora monostabilnego) jest zezwolenie na pracę multiwibratora astabilnego zbudowanego na drugim układzie NE555 - IC6. Podczas trwania impulsu „wyrównującego“ generowanego przez IC7, układ ten był zablokowany stanem niskim panującym na jego wejściu zerującym R.

A zatem podsumujmy: po zaistnieniu wszystkich opisanych zdarzeń, liczniki IC2 i IC3 rozpoczynają zliczanie w dół, od liczby określającej położenie wału napędowego pierwszego serwomechanizmu. Zliczanie trwa aż do momentu osiągnięcia przez te liczniki stanu 00000000, kiedy to na wyjściu przeniesienia BO licznika IC3 pojawi się stan niski. Spowoduje to powstanie krótkiego impulsu ujemnego na wejściu wyzwalającym multiwibratora monostabilnego IC7 i rozpoczęcie przez ten układ generacji impulsu „wyrównującego“. Zwróćmy uwagę, że na wyjściu 3 CON1 panuje jeszcze stan wysoki i czas trwania impulsu generowanego przez IC7 zostanie dodany do czasu zliczania określonego liczbą podaną na wejścia liczników IC2 i IC3.

Zakończenie generacji impulsu przez IC7 spowoduje zmianę stanu licznika IC5B o 1. Na wyjściu 3 CON1 pojawi się stan niski, co dla elektronicznego układu wbudowanego w serwo jest sygnałem o końcu impulsu sterującego. Otwarty zostanie kolejny bufor - IC14, jego zawartość zostanie przepisana do rejestrów liczników IC2 i IC3 i opisane wyżej zjawiska będą się powtarzać aż do wyłączenia zasilania.

Warto zauważyć, że część układu sterująca bezpośrednio serwomechanizmami może pracować niezależnie od komputera. Po wpisaniu przez komputer właściwych danych do rejestrów, układ pracuje samodzielnie, utrzymując wały napędowe serwomechanizmów w zadanej pozycji. Komputer interweniuje dopiero w momencie, kiedy zajdzie konieczność ponownego pozycjonowania serwomechanizmów. Daje do duże możliwości konstruktorom i programistom, umożliwiając wykorzystane „wolnego czasu“ komputera do obsługi innych urządzeń, np. czujników dołączonych do rejestru wejściowego interfejsu CENTRONICS.



Rys. 2. Rozmieszczenie elementów na płytce drukowanej.

Należy powiedzieć jeszcze parę słów na temat generatorów zbudowanych na IC6 i IC7. W zależności od przyjętego trybu pracy układ IC7 powinien generować impuls o długości dokładnie 1 ms (w przypadku stosowania standardu przyjętego w aparaturach RC) lub impuls o długości ustalanej najczęściej doświadczalnie i wynoszącej ok. 0,5 ms. (w przypadku, jeżeli zdecydujemy się wykorzystywać maksymalny, możliwy do osiągnięcia kąt obrotu wałów serwomechanizmów). Częstotliwość pracy generatora zbudowanego na układzie IC6 także ściśle zależy od przyjętego trybu pracy. W przypadku korzystania z standardowego kąta obrotu wałów serw powinna wynosić dokładnie 255kHz, natomiast jeżeli będzie nam zależało na zwiększeniu tego kąta do 180° lub więcej stopni, należy ją zwiększyć do około 600..650kHz.

Na zakończenie warto powiedzieć jeszcze parę słów na temat zasilania naszego sterownika serwomechanizmów. Układ, podobnie jak każdy wykorzystujący technologię TTL, wymaga zasilania napięciem 5VDC, najlepiej stabilizowanym. Natomiast serwomechanizmy modelarskie mogą poprawnie pracować w zakresie napięć od 4,8 do 6VDC. Z pozoru sprawa jest więc prosta: należy cały układ włącznie z serwami zasilić napięciem 5V i po kłopocie. Niestety, nie jest to aż tak proste, a powodem powstawania problemów jest duży prąd pobierany przez serwomechanizmy. Jedno serwo może, pokonując duże obciążenie, pobrać z zasilacza prąd o wartości do 1A. Ponieważ zasady projektowania zasilaczy do układów elektronicznych nakazują wykonać zasilacz pracujący poprawnie nawet w najbardziej krytycznej sytuacji, musielibyśmy zastosować źródło prądu o napięciu 5V i obciążalności prądowej minimum 8,5A. Jest to zadanie oczywiście wykonalne, ale dość kłopotliwe w praktycznej realizacji. Dlatego też sugerowałbym zastosowanie innego rozwiązania. Jeżeli nie będzie zachodziła konieczność pracy układu przez dłuższy czas bez jakichkolwiek przerw, to można pomyśleć o zasileniu go z okresowo doładowywanego akumulatora o pojemności rzędu 10Ah, najlepiej kwasowego, tzw. niewylewnego. Rozwiązuje to problem wydajności prądowej źródła zasilania, natomiast pojawia się inny kłopot: akumulator dostarcza napięcie 6,2V, którym nie można bezpośrednio zasilać układów TTL. Dlatego też zastosowałem dwie diody D1 i D2, których zadaniem jest redukcja napięcia wejściowego podanego na złącze CON10 o 1,2V. Jeżeli będziemy korzystali z zasilacza o napięciu 5V, to jumper JP1 musi być ustawiony w pozycji przeciwnej do pokazanej na schemacie. Jeżeli do zasilania układu wykorzystamy akumulator, to jumper JP1 musi dołączyć zasilanie części cyfrowej układu do diody D2.

Montaż i uruchomienie

Na **rys. 2** pokazano rozmieszczenie elementów na dwustronnej płytce drukowanej, której widoki mozaiki ścieżek znajdują się na wkładce wewnątrz numeru.

Zmontowany ze sprawdzonych elementów układ nie wymaga uruchamiania, ale jedynie prostej regulacji polegającej na ustawieniu czasu trwania impulsu generowanego przez IC7 i częstotliwości pracy generatora z IC6. Jeżeli zdecydujemy się na stosowanie standardów używanych w układach radiowego sterowania proporcjonalnego, to sprawa jest prosta: ustawiamy długość impulsu IC7 na 1 ms, a częstotliwość na wyjściu IC6 na 255kHz. Jednak nasz układ przeznaczony jest przede wszystkim do sterowania robotami i nie sądzę, aby kąt obrotu wałów serwomechanizmów wynoszący 60° mógł okazać się wystarczający np. przy budowie ręki robota. Dlatego też przypuszczam, że większość z Was zdecyduje się na sterowanie serwami w pełnym zakresie ich kąta obrotu i będzie zmuszonych do dokonania regulacji układu metodą doświadczalną, na szczęście trywialnie prostą. Kolejność postępowania będzie następująca:

1. Do wszystkich rejestrów pamięci układu zapisujemy liczbę 0. Możemy to uczynić za pomocą prostego programu napisanego w języku BASIC, którego listing zamieszczam poniżej. Program ten jest jednocześnie przykładem, jakie procedury utworzone w tym języku potrzebne są do napisania prostego programu obsługującego zbudowany przez nas układ.

```
OUT &H378, 0
GOSUB ZA
GOSUB ZB
GOSUB ZC
GOSUB ZD
GOSUB ZE
GOSUB ZF
GOSUB ZG
GOSUB ZH

ZA:
OUT &H37A, 11
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
RETURN
ZB:
OUT &H37A, 10
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
RETURN
ZC:
OUT &H37A, 9
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
RETURN
ZD:
OUT &H37A, 8
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
RETURN
ZE:
OUT &H37A, 15
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
ZF:
OUT &H37A, 14
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
RETURN
ZG:
OUT &H37A, 13
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
RETURN
ZH:
OUT &H37A, 12
FOR R = 1|TO 5: NEXT R
OUT &H37A, 3
RETURN
```

2. Do któregokolwiek z wyjść CON1..CON8 dołączamy standardowy serwomechanizm modelarski i obserwujemy jego zachowanie podczas obracania potencjometrem montażowym PR2. Wał napędowy serwa będzie się poruszał wraz ze zmianą położenia PR2, ale z pewnością natrafimy na moment, w którym dalszy obrót wału nie będzie możliwy, a serwomechanizm zacznie wydawać z siebie ciche brzęczenie. Oznacza to, że długość impulsu generowanego przez IC7 przekroczyła minimalną wartość. Następnie obracamy powoli oskę PR2 w kierunku przeciwnym niż poprzednio, aż do uzyskania minimalnego poruszenia wału napędowego. W takim położeniu pozostawiamy PR2 wiedząc, że osiągnęliśmy najmniejszy kąt ustawienia wałów napędowych serw.

3. Kolejną czynnością będzie ustawienie częstotliwości impulsów generowanych przez IC6, od której zależy maksymalny kąt obrotu wałów napędowych serwomechanizmów. Do wszystkich rejestrów pamięci układu wpisujemy teraz liczbę 255. Za pomocą potencjometru montażowego PR1 ustawiamy, w sposób analogiczny do opisanego wyżej, maksymalne wychylenie wału napędowego serwa dołączonego do jednego z wyjść CON1..CON8.

Po wykonaniu powyższych czynności możemy uznać nasz układ za sprawny i rozpocząć jego eksploatację. Jak już wspominałem, na początek proponuję Wam świetną zabawę: wykonanie ręki robota, do której zbudowania będziecie potrzebować jedynie śrubokręta i garstki śrubek M3 z nakrętkami (zakładam, że lutownicę każdy z Was już posiada)!

Pozwoliłem sobie przygotować dla Was bardzo nietypowy kit, składający się z równie nietypowych płytek drukowanych, będących w rzeczywistości mechanicznymi elementami konstrukcyjnymi. Aby zbudować robota wystarczy jedynie zlutować ze sobą odpowiednie kształtki i skręcić całość śrubkami. Zawarte w kicie płytki pozwolą na budowę ręki robota o czterech stopniach swobody oraz „łapy“ mogącej chwytać i przenosić przedmioty o wadze do 500g. Jak już wspomniałem, lepsze rezultaty można osiągnąć stosując serwomechanizmy modelarskie o zwiększonym momencie obrotowym, stosowane np. przy budowie wyczynowych modeli samolotów akrobacyjnych. Niestety, takie serwa są bardzo kosztowne i ich zastosowanie wiąże się z kilkakrotnym zwiększeniem kosztów budowy robota.

Zbgniew Raabe, AVT
zbigniew.raabe@ep.com.pl

Serwomechanizmy modelarskie już w ofercie handlowej AVT!

WYKAZ ELEMENTÓW

Rezystory
PR1, PR2: potencjometr regulacyjny miniaturowy 200kΩ
R1, R3, R5: 10kΩ
R2: 33kΩ
R4: 4,7kΩ

Kondensatory
C1, C3, C4, C6: 10nF
C2: 100pF
C8, C5: 100nF
C9, C6: 1000µF/16V
C10: 1nF

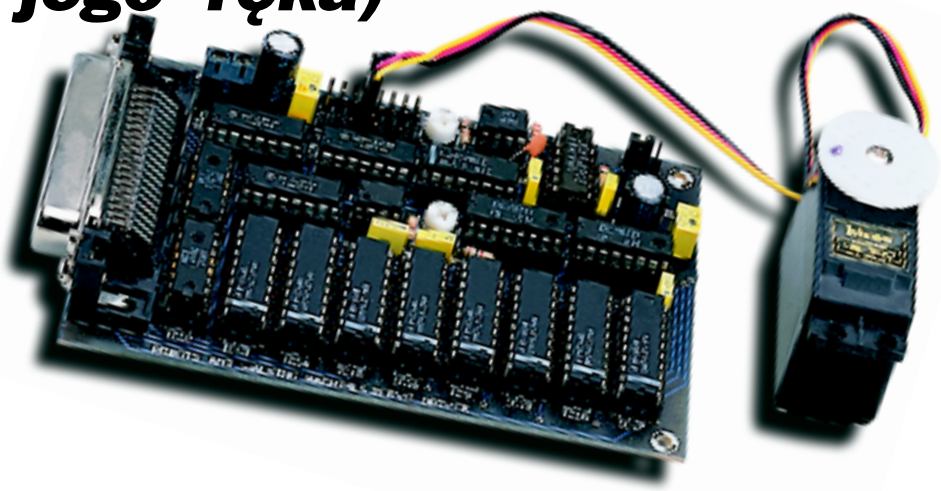
Półprzewodniki
D1, D2: 1N4001 lub odpowiednik
IC1, IC11: 74LS42
IC2, IC3: 74LS193
IC4: 4028
IC5: 74LS393
IC6, IC7: NE555
IC8: 74LS02
IC9, IC13, IC14, IC15, IC16, IC17, IC18, IC19: 74LS574
IC10, IC12: 4050

Różne
CON1..CON8: 3 x goldpin
CON9: kątowe złącze Centronics-36
CON10: ARK2
JP1: jumper + 3 goldpin

Robot, część 2

(a właściwie jego ręka)

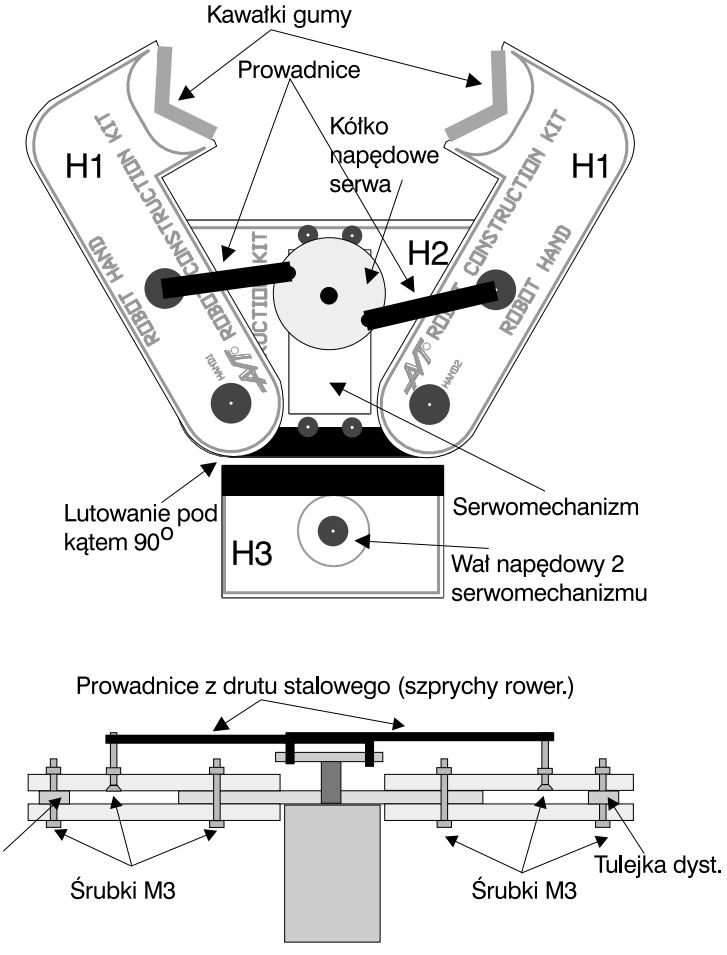
AVT-821



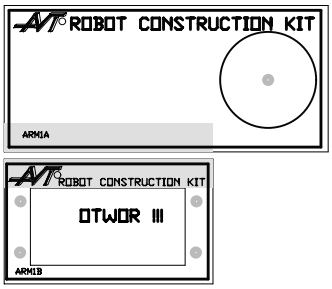
Po szczegółowym zapoznaniu się z budową i zasadami działania części elektronicznej robota najwyższy czas przejść do omówienia programu sterującego jego pracą oraz problemów związanych z wykonaniem elementów mechanicznych konstrukcji.

Zajmijmy się teraz najtrudniejszą sprawą związaną z budową ramienia robota - mechaniką. Na szczęście, dzięki zastosowaniu prefabrykowanych kształtek wykonanych z laminatu epoksydowo - szklanego prace mechaniczne zostały maksymalnie uproszczone. Prototyp ramienia robota został przeze mnie wykonany w ciągu niecałych dwóch godzin i mam nadzieję, że i Wam nie zajmie on więcej czasu. Praktycznie wszystkie materiały potrzebne do mechanicznego wykonania ramienia otrzymacie w kicie, a samodzielnie będziecie musieli skompletować tylko następujące elementy:

- Śrubki M3 o długości ok. 3 cm z łebkami walcowymi i stożkowymi.
- Nakrętki M3.
- Podkładki o średnicy wewnętrznej 3mm.
- Odcinki drutu stalowego o średnicy ok. 1mm. Mogą to być kawałki szprych rowerowych, łat-



Rys. 3. Schemat budowy "ręki" robota.

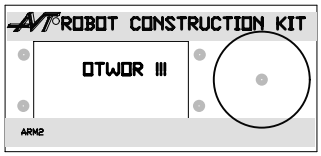


Rys. 4a. Płytki drukowane konstrukcji mechanicznej ramienia.

wych do kupienia w każdym sklepie z częściami do rowerów.
- Klej, np. Super Glue lub podobny.
- Kawałki gumy lub podobnego tworzywa.

Na trudności możecie napotkać jedynie podczas budowy nieco skomplikowanej ręki robota, budowa elementów ramienia i podstawy sprowadzająca się do zlutowania ze sobą dwóch kawałków laminatu nie powinna nastręczyć nikomu żadnych problemów. Na **rys. 3** pokazano szczegóły wykonania ręki, a właściwie szczypiec, które umożliwią naszemu robotowi chwytanie i przenoszenie przedmiotów o nawet sporych rozmiarach i wadze.

Do budowy ręki potrzebować będziemy zestawu kształtek pokazanych na **rys. 4** (w dużym pomniejszeniu). Każdy z dwóch palców ręki robota składa się z dwóch kawałków laminatu w kształcie litery „L“, pomiędzy którymi umieszczona zostanie płytka główna ręki z zamocowanym do niej serwomechanizmem.



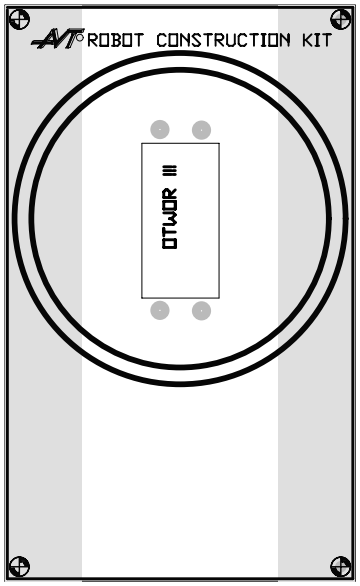
Rys. 4b. Płytki drukowane konstrukcji mechanicznej ramienia.

Palce należy przykręcić do płytki głównej za pomocą dwóch śrubek M3, tak aby poruszać się z minimalnym oporem. Aby zabezpieczyć konstrukcję przed rozkręceniem się, każda śrubka powinna zostać zaopatrzona w kontrnakrętkę i podkładki.

Następną czynnością będzie zamocowanie serwomechanizmu do płytki głównej i wykonanie mimośrodowych prowadnic umożliwiających robotowi poruszanie palcami. Prowadnice wykonamy z drutu stalowego wygiętego tak, aby jeden koniec prowadnic trafiał w otworki na obwodzie kółka napędowego serwomechanizmu a drugi, uformowany w „oczko“ o średnicy wewnętrznej 3mm dał się nałożyć na śrubki M3 przykręcone do palców ręki.

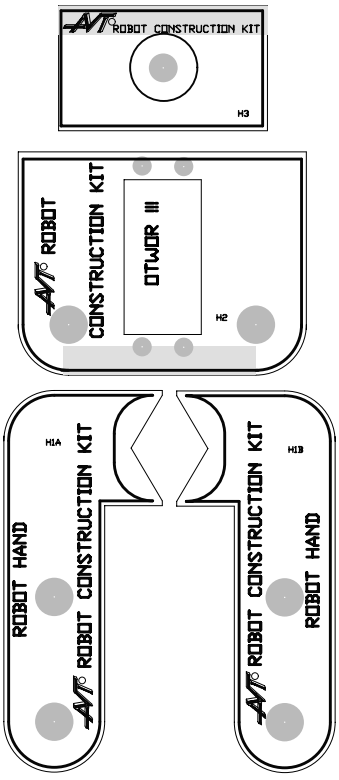
Ostatnią czynnością podczas montażu dłoni robota będzie przylutowanie kształtki oznaczonej jako H3 do płytki głównej, dokładnie pod kątem prostym. Do kształtki H3 zamocujemy następnie serwo będące obrotowym przegubem ręki.

Montaż elementów ramienia nie powinien przysporzyć nikomu problemów. Serwomechanizmy możemy łączyć zarówno w jednej



Rys. 4c. Płytki drukowane konstrukcji mechanicznej ramienia.

płaszczyźnie jak i pod kątem prostym. Do łączenia serw w jednej płaszczyźnie służy kształtka oznaczona jako ARM2, natomiast do połączenia kątowego wykorzystamy zlutowane ze sobą kształtki oznaczone jako ARM1A i ARM1B. Teoretycznie możemy połączyć ze sobą dowolną ilość serwomechanizmów, uzyskując w ten sposób wiele stopni swobody poruszeń



Rys. 4d. Płytki drukowane konstrukcji mechanicznej ramienia.

ramienia robota. Jednak ze względu na wymaganą stabilność konstrukcji nie polecam konstruowania ramion o ilości serw większej od 3..4.

Ostatnim elementem, który wchodzi w skład kitu jest podstawa (BASE), do której należy przymocować serwomechanizm obracający całą konstrukcję ramienia wokół osi pionowej. Dotarliśmy wreszcie do bardzo ważnej sprawy, jaką jest stabilne zamocowanie całej konstrukcji. Na okładce poprzedniego numeru Elektroniki Praktycznej mogliśmy zobaczyć robota zamocowanego na ruchomej platformie z napędem gąsienicowym. Jest to rozwiązanie bardzo widowiskowe, ale uniemożliwiające podnoszenie cięższych przedmiotów i nie zapewniające pełnej powtarzalności ruchów robota. Dlatego też lepiej zamocować podstawę ramienia do czegoś bardziej solidnego i cięższego. Do laboratoryjnych testów robota wykonana została solidna, metalowa podstawa z zamocowanymi w jej wnętrzu akumulatorami.

Pewnie wielu Czytelników zadaje sobie pytanie, do czego właściwie można wykorzystać zbudowane wielkim nakładem pracy ramię robota? Przede wszystkim do eksperymentów i zabawy. Robot taki jest przecież niewyobrażalnie atrakcyjną zabawką dla dzieci, a gdy dzieci pójdą spać to dla... dorosłych. Nie obiecujmy jednak sobie zbyt wiele: precyzja poruszeń tak zbudowanego ramienia nie jest zbyt wielka, ale do prostych czynności m. może okazać się wystarczająca. Prototyp robota z łatwością radził sobie z nalewaniem piwa (oczywiście bezalkoholowego), potrafił poodnosić i przemieszczać przedmioty, a nawet... opiekać kiełbaski na grillu!

SPIS ELEMENTÓW WCHODZĄCYCH W SKŁAD ZESTAWU "MECHANICZNEGO"

Elementy do budowy dłoni robota

płytka H1A	2 szt.
płytka H1B	2 szt.
płytka H2	1 szt.
płytka H3	1szt.

Elementy do budowy ramienia

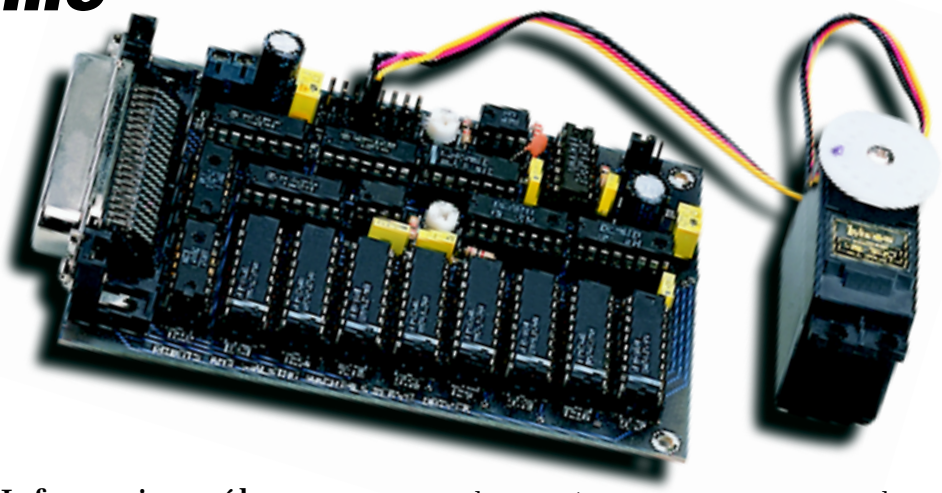
płytka ARM1A	3 szt.
płytka ARM1B	3 szt.
płytka ARM2	3 szt.
Element podstawy BASE	1szt.

Zbgniew Raabe, AVT
zbigniew.raabe@ep.com.pl

Serwomechanizmy modelarskie już w ofercie handlowej AVT!

Robot, część 3

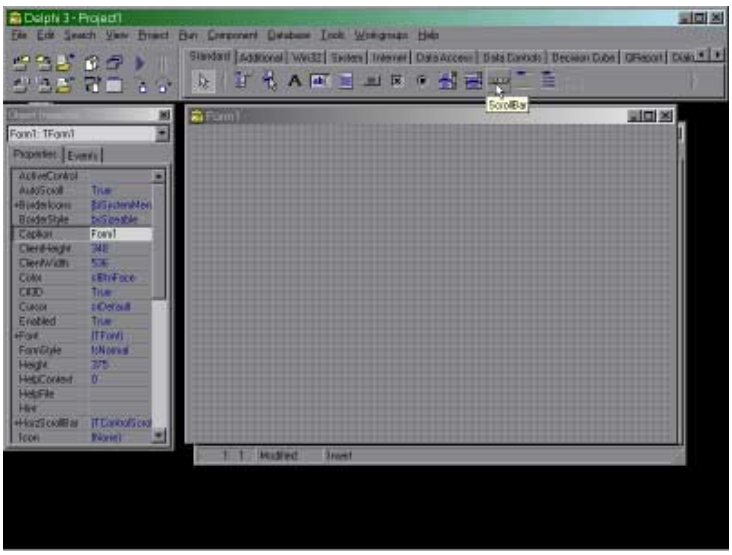
Oprogramowanie AVT-821



W trzeciej, ostatniej części artykułu przedstawiamy opis oprogramowania sterującego „ręką" robota. Opis procedur przygotowanych przez autora ułatwi Czytelnikom przygotowanie oprogramowania spełniającego ich indywidualne wymagania.

Informacje ogólne

Program *Raabot* jest dostarczany z kitem AVT-821. Służy do edytowania, zapisywania i odtwarzania kolejnych położeń serwomechanizmów stanowiących elementy wykonawcze robota. Program przeznaczony jest do pracy w systemie operacyjnym Windows 95 lub 98. Nie będzie działał prawidłowo z wcześniejszymi wersjami Windows i z Windows NT. Został napisany za pomocą Delphi 3, z wykorzystaniem funkcji API32. Niemożliwia jest więc jego współpraca ze starszymi, 16-bitowymi wersjami Windows. Nie pomoże nawet zainstalowanie Win32 do Windows wersji 3.1 i 3.11. Ponieważ program odwołuje się bezpośrednio do portów za pomocą wstawek asemblerowych, co jest niedozwolonym rozwiązaniem w systemie Windows NT, może działać niepoprawnie również w tym systemie.



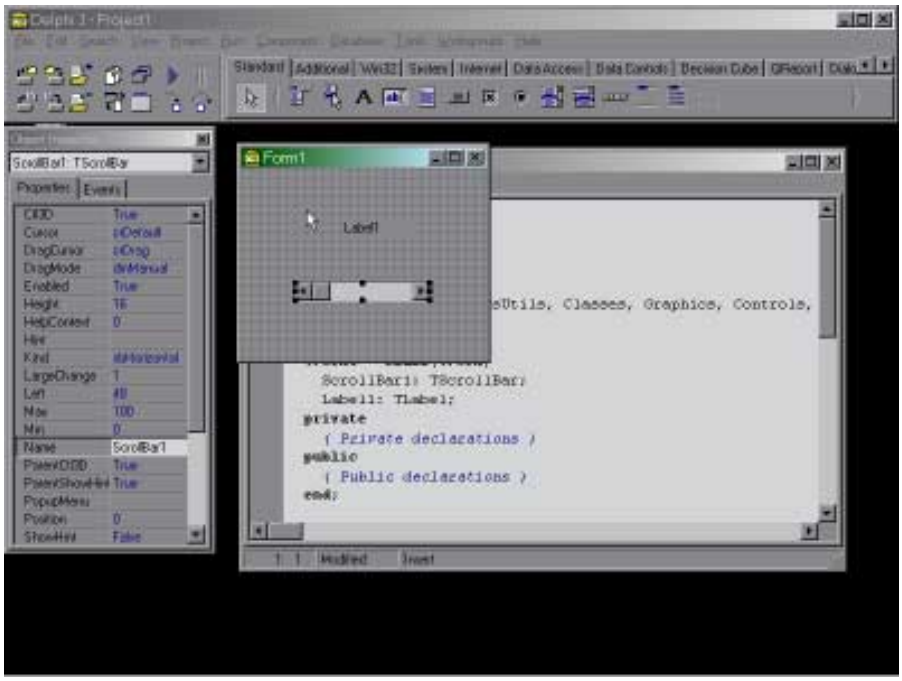
Rys. 5. Otwarte okno nowej aplikacji w Delphi.

Porty, adresy

Program *Raabot* steruje układem z kitu AVT-821 za pośrednictwem łącza równoległego. W programie przewidziana jest współpraca ze standardowymi portami LPT1 (adres 378h) i LPT2 (adres 278h). Zmiany portu można dokonać wybierając odpowiednią pozycję w menu Konfiguracja. W programie nie przewidziano współpracy z portami ustawionymi niestandardowo.

Instalowanie i odinstalowywanie

Aby zainstalować to oprogramowanie należy uruchomić program *setup.exe* z dyskietki instalacyjnej. Program instalacyjny skopiuje pliki oprogramowania na dysk twardy i utworzy grupę programu *Raabot* w Menu Start. Program zapisuje w systemie dane konfiguracyjne w pliku rejestrów. Dlatego, aby program odinstalować całkowicie, należy skorzystać z polecenia *Dodaj/Usuń Programy* z Panelu Sterowania lub z programu *Usuń Raabota*, znajdującego się w grupie programu *Raabot*. Nie zaleca się ręcznego usuwania programu. Nie stanowi to żadnego niebezpieczeństwa dla systemu, a nawet nie zaśmieca go w istot-



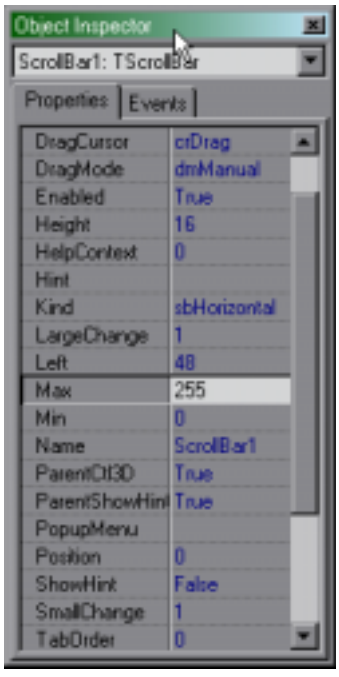
Rys. 6. Sposób lokalizowania suwaka.

nym stopniu. Jest to po prostu nieeleganckie.

Programy instalacyjny i odinstalowywujący są całkowicie zgodne ze standardami obowiązującymi w systemach Windows 95 i 98.

Uwagi dla użytkowników

Program służy do ustawiania, modyfikacji, zapisywania i odtwarzania kolejnych położeń każdego z ośmiu serwomechanizmów oddzielnie. Przewidziana jest możliwość zapisania do 10000 kolejnych położeń każdego z serwomechanizmów. Położenie każdego z suwaków odpowiada aktualnemu położeniu odpowiedniego serwomechanizmu. Aby zapamiętać bieżące położenia serwomechanizmów i przejść do kolejnej pozycji naciśnij *Następny*. Przyciski: *Poprzedni*, *Pierwszy* i *Ostatni* umożliwiają zmianę aktualnej pozycji. Przyciski *Wstaw* i *Usuń* umożliwiają dodawanie i usuwanie pozycji.



Rys. 7. Okno definiujące parametry suwaka.

Pola znajdujące się po lewej stronie suwaków sterujących serwami pozwalają na wpisanie własnej nazwy danego serwomechanizmu. W konkretnym modelu pozwala to nadawać nazwy poszczególnym serwom, np. szczypce czy obrót przedramienia. Umożliwia to łatwą identyfikację aktualnie edytowanego stopnia swobody Raabota. Pola znajdujące się po prawej stronie suwaków wskazują aktualne położenie serwa w przedziale od 0 do 255.

Dla każdej pozycji można indywidualnie ustawić prędkość poruszania się serwomechanizmów za pomocą wskaźników wyboru (radio button), znajdujących się po prawej stronie w grupie Prędkość. Przewidziano trzy różne prędkości.

Włączenie odtwarzania całej sekwencji położeń można dokonać klikając na przełącznik *On/Off*. Przewidziano również odtwarzanie całej sekwencji w pętli.

Szybka edycja położeń serwomechanizmów w kolejnych pozycjach możliwa jest przy wykorzystaniu klawisza TAB i kursorów.

Całą utworzoną sekwencję ruchów można zapisać w pliku w sposób standardowy dla systemu Windows 95/98.

Jak działa program

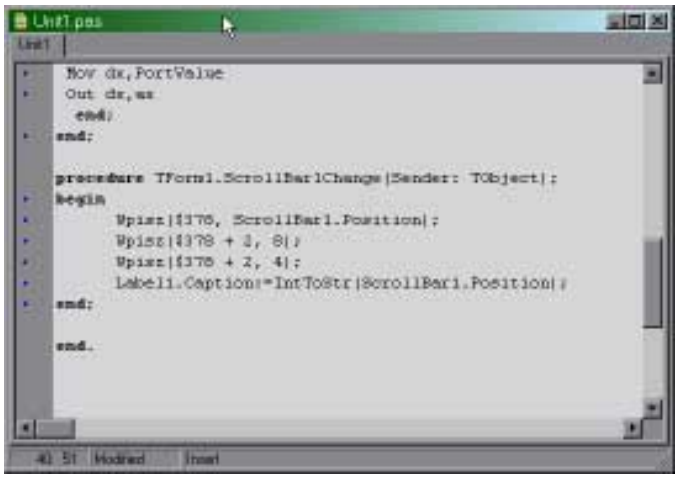
Delphi jest wyjątkowo wdzięcznym narzędziem programistycznym. Stanowi bardzo inteligentne i intuicyjne połączenie starego, powszechnie lubianego Pascala, środowiska graficznego i obiektowych technik programowania. Programowanie w nim jest bardzo proste, nawet jeżeli ktoś od Delphi zaczyna programowanie pod Windows.

Serce aplikacji Raabot stanowi króciutka, standardowa procedurka zapisująca do portu o podanym adresie wartość podaną jako parametr:

```
procedure Wpisz(PortValue, DataValue:
word);
begin
  DataValue := (DataValue * 256) + Da-
taValue;
  asm
        Mov ax,DataValue
        Mov dx,PortValue
        Out dx,ax
  end;
end;
```

To co znajduje się pomiędzy słowami kluczowymi asm i end, to właśnie wspomniana wcześniej wstawka asemblerowa zapisująca do portu o adresie *PortValue* wartość *DataValue*. Jednak, aby poinformować urządzenie AVT-821 o położeniu kolejnego serwa, należy do portu zapisać cały ciąg informacji.

```
function DoPortu(P: Pointer): integer;
var
  i: integer;
begin
  repeat
    for i := 1 to 8 do
    // dla każdego serwa
    begin
      Wpisz(nrportu, a[i]);
      // podaj położenie
      Wpisz(nrportu + 2, 7 + i);
     // wraz z informacją o którym
     // serwie mowa
```



Rys. 8. Okno edytora tekstu z programem obsługi suwaka.

```
      Wpisz(nrportu + 2, 4);
     // koniec przekazu :-)
    end;
  until blad=true;
end;
```

W macierzy a[8] są zapisane aktualne położenia wszystkich ośmiu serw.

I to wszystko!

Tym, którzy chcieliby się przekonać, jak proste jest dziś programowanie pod Windows proponuję następujący eksperyment: spróbujcie napisać własny program do sterowania Raabotem w 10 minut! Najpierw powinniście rozejrzeć się za darmową wersją Delphi. Na przykład Delphi 2 było wielokrotnie publikowane na krążkach dołączanych do czasopism o tematyce "komputerowej".

Po otwarciu nowego projektu aplikacji (Menu File| New Application) powinniśmy zobaczyć na ekranie komputera sytuację jak na **rys. 5**.

Okno zatytułowane Form1, to nasza aplikacja. Powinniśmy dodać do niego jakieś elementy. Na przykład suwak (ScrollBar) i opis tekstowy (Label). W tym celu powinniśmy na pasku narzędzi znaleźć potrzebne nam elementy i dodać je do naszego projektu poprzez dwukrotne kliknięcie. Elementy te możemy na naszym oknie dowolnie przesuwać za pomocą myszy. Ustawmy je mniej więcej tak, jak to widać na **rys. 6**.

W oknie *Object Inspector* musimy teraz zmienić jedną z właściwości naszego paska. Właściwość *Max* odpowiedzialną za to, jaką maksymalną wartość zwraca pasek, zmieniamy z domyślnej wartości 100 na 255 (**rys. 7**).

Trzecie okno, jakie mamy do dyspozycji, to edytor kodu źródłowego. Zawiera on w tej chwili jedynie generowany automatycznie kod, który musimy teraz uzupełnić o napisane przez nas procedury.

Pod słowem kluczowym *implementation* wpiszmy naszą procedurkę Wpisz(*PortValue, DataValue: word*) podaną wyżej.

Potem kliknijmy dwukrotnie na pasku *ScrollBar1*. Pojawi się kod źródłowy tworzonej aplikacji z gotowym szkieletem procedury, która będzie wywoływana za każdym razem, gdy przesuniesz pasek *ScrollBar1*, czyli procedurę *TForm1.ScrollBar1Change(Sender: TObject)*. Pomiędzy słowa kluczowe *begin* i *end* wpiszmy:

```
Wpisz($378, ScrollBar1.Position);
Wpisz($378 + 2, 8);
Wpisz($378 + 2, 4);
Label1.Caption:=IntToStr
   (ScrollBar1.Position);
```

Całość, wraz z kodem generowanym automatycznie, powinna wyglądać następująco:

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils,
Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls;

type
  TForm1 = class(TForm)
    ScrollBar1: TScrollBar;
    Label1: TLabel;
    procedure ScrollBar1Change(Sen-
der: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure Wpisz(PortValue, DataValue:
word);
begin
  DataValue := (DataValue * 256) +
DataValue;
  asm
 Mov ax,DataValue
 Mov dx,PortValue
 Out dx,ax
  end;
end;

procedure TForm1.ScrollBar1Chan-
ge(Sender: TObject);
begin
 Wpisz($378, ScrollBar1.Position);
      Wpisz($378 + 2, 8);
      Wpisz($378 + 2, 4);
      Label1.Caption:=IntToStr(Scrol-
lBar1.Position);
end;
end.
```

I to wszystko! Pozostało nam nacisnąć F9, czyli skompilować projekt i cieszyć się z własnego programu, który na ruch suwaczka reaguje ruchem pierwszego serwa naszego Raabota. To naprawdę proste. Zachęcam do eksperymentowania. Jeżeli jesteście zainteresowani tą tematyką napiszcie do nas.

Być może przygotujemy publikację wyjaśniającą sposoby zapisywania i odczytywania portów z poziomu Delphi.

Adam Dębowski, AVT

Rys. 9. Okno działającego programu.