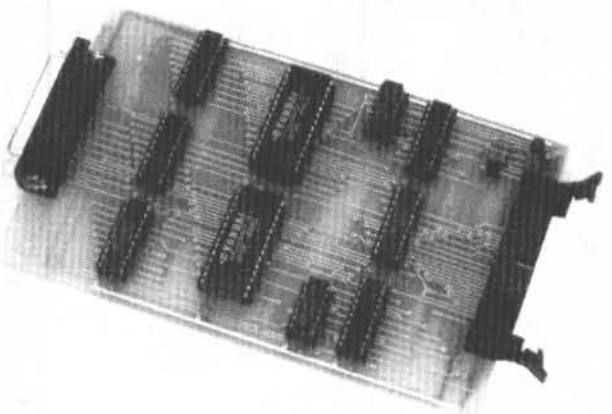
Symulator pamięci EPROM jest narzędziem bezcennym dla projektowania systemów mikroprocesorowych. Opisany układ jest symulatorem pamięci stałej ROM, PROM lub EPROM o pojemności do 64kB.

Symulator EPROMów

kit AVT-214



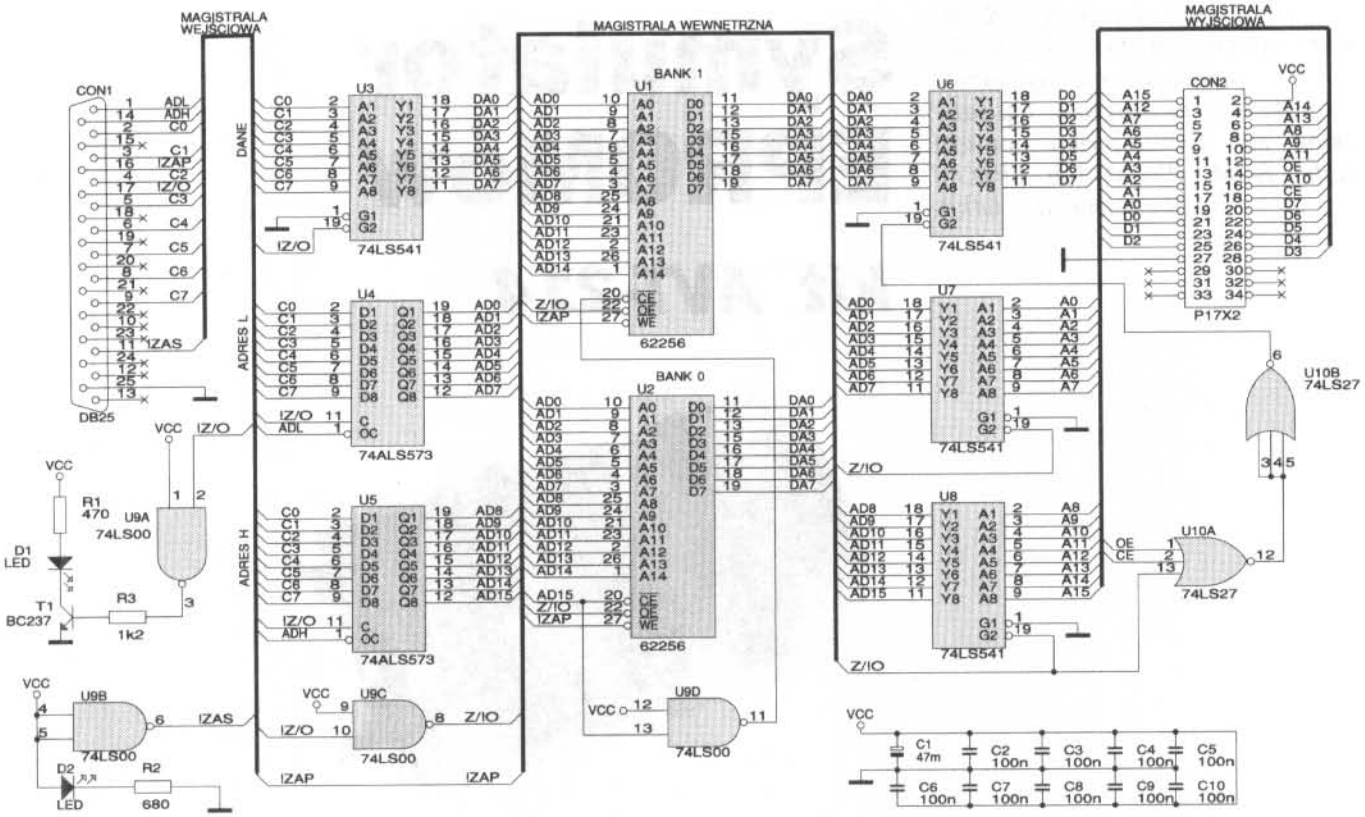
Informacje ogólne

W dalszej części tekstu będzie używana tylko nazwa EPROM, gdyż ten rodzaj pamięci jest obecnie najczęściej wykorzystywany w konstrukcjach. Największa pamięć EPROM jaką jest w stanie zastąpić symulator ma oznaczenie numeryczne 27512. W chwili obecnej nie jest to fascynująca wielkość, lecz układ powstał z przeznaczeniem dla małych systemów mikroprocesorowych, dla których pamięci tej wielkości są zupełnie wystarczające. Poza tym, w projekcie zwracano uwagę na minimalizację kosztu urządzenia, a ponieważ statyczne pamięci są stosunkowo drogie nie miało sensu sztuczne zwiększanie pojemności bez perspektyw jej wykorzystania. Całość jest zbudowana z popularnych układów, łatwo dostępnych na rynku. Charakterystyczna jest znikoma liczba elementów dyskretnych. Niecierpliwi Czytelnicy, którzy zaczynają czytanie artykułu od oglądania schematu, już wiedzą, że występują trzy rezystory, dwie diody świecące oraz jeden tranzystor, nie licząc kondensatorów odsprzęgających, bez których układ działa zupełnie poprawnie, a ich zastosowanie jest sprawą „dobrego stylu" i przestrzegania ogólnie przyjętych zasad. Istniejące oprogramowanie umożliwia symulację pamięci o mniejszych pojemnościach. Konieczne jest wówczas użycie innej końcówki emulacyjnej, ponieważ takie pamięci są produkowane w innych obudowach i mają inaczej rozmieszczone wyprowadzenia. Sposób wykonania połączeń do wszystkich rodzajów końcówek jest podany w części opisującej konstrukcję. Symulator współpracuje z dowolnym komputerem klasy PC posiadającym port równoległy drukarki.

Budowa układu

Jak już wcześniej wspomniano symulator komunikuje się z komputerem przez port drukarki. Ponieważ na tym porcie nie występuje pin zasilania, wydaje się koniecznym wykonanie zasilacza. Wszystkie układy scalone powinny być zasilane napięciem +5V, ponieważ pracują w standardzie TTL jak większość układów mikroprocesorowych. Korzystając z tej cechy zdecydowano się na pewien wybieg polegający na pobieraniu zasilania przez nóżkę, która docelowo ma zasilać symulowaną pamięć. Rozwiązanie to jest wygodne, lecz posiada jedną poważną wadę, o której nie wolno zapominać; symulator pobiera około 250mA prądu, podczas gdy przeciętny EPROM o tej pojemności tylko do 50mA. Przy takim rozwiązaniu oczywistą rzeczą jest, że najpierw należy włączyć uruchamiany układ, a następnie załadować program do symulatora. Po tych czynnościach warto podać reset do procesora. Zapewni to wykonywanie programu od początku. Podane tu czynności są oczywiste, lecz nauczeni doświadczeniem wiemy, iż łatwo jest zapomnieć o nich skupiając się na uruchamianiu w takiej sytuacji zupełnie nie działającego symulatora. O zasilaniu informuje dioda świecąca D2. Jej prąd ogranicza R2. Dodatkowo do plusa zasilania są podłączone wejścia bramki U9B, na której wyjściu podczas zasilania utrzymuje się stan niski. Ta informacja jest odczytywana przez program sterujący w porcie stanu drukarki na bicie 7. Dzięki temu prostemu rozwiązaniu najbardziej roztargnionym konstruktorom o włączeniu zasilania przypomni program ładujący.

Kolejne adresy komórek pamięci oraz ich zawartości są podawane przez piny 2 - 9 w złączu (rejestr danych dla drukarki) jednocześnie na układy U3, U4, U5. Układy U4 i U5 (74LS573) to ośmiobitowe zatrzaski z wyjściami trzystanowymi pamiętające adres programowanej komórki, a U3 (74LS541) jest jednokierun-



Rys. 1. Schemat elektryczny symulatora

kową ośmiobitową bramą transmisyjną również z wyjściem trzystanowym. Zadaniem tego układu jest podanie wartości bajtu programowanej komórki. Wyjścia tych układów łączą się bezpośrednio z pamięciami oznaczonymi tu symbolami U1, U2. Ponieważ brak jest na rynku pamięci statycznych o pojemności 64kB, konieczność zmusiła autora do zastosowania dwóch układów po 32kB każdy, o oznaczeniach numerycznych 62256 co w bardzo niewielkim stopniu skomplikowało całą konstrukcję. W U2 pamiętane są bajty o młodszych adresach, a w U1 bajty starsze. Stąd mają one dodatkowe nazwy BANK 0 i BANK 1. Od strony końcówki emulacyjnej pracują trzy jednakowe układy U6, U7, U8, identyczne jak U3. Układ U6 spełnia rolę bufora wyjściowego w bramie danych EPROM-u. U7 i U8 buforują magistralę adresową uruchamianego systemu. Te trzy układy można by w zasadzie pominąć, lecz nie jest to wskazane z uwagi na ewentualne błędy (zwarcia) w oprogramowywanym urządzeniu. W przypadku usterki spalony zostanie bufor, którego cena jest niewspółmiernie mała do ceny pamięci, a dokładniej dwóch pamięci, bo wszystkie linie z końcówki emulacyjnej (bez A15) są jednocześnie podłączone do U1 i U2.

Ponieważ w porcie drukarki mamy do dyspozycji tylko jedną bramę ośmiobitową, a na informację przekazywaną do symulatora składają się aż 24 bity (16 bitów adresu i 8 bitów danych), należy w jakiś sposób ustalić jakie dane są wystawione na bramę portu. Do tego celu wykorzystany jest ośmiobitowy rejestr sterujący drukarką, z którego tylko cztery młodsze bity są dostępne w złączu. W tym miejscu warto zapoznać się z **tabelą 1** opisującą port drukarki od strony rejestrów z przestrzeni adresowej PC i pinów w złączu.

Symulator posiada dwa tryby pracy: 0 zapis, 1 odczyt.

Tryb ustalany jest przez program, sterując poprzez odpowiednie usta-

Tab. 1.

NR PORTU	ADRES	NR BITU	NR PINU	FUNKCJA
REJESTR DANYCH WYJŚCIOWY				
LPT 1	378H	0	2	DANE WYJŚCIOWE
LPT 2	278H	1	3	
LPT HERC	3BCH	2	4	
		3	5	
		4	6	
		5	7	
		6	8	
		7	9	
REJESTR STEROWANIA WYJŚCIOWY				
LPT 1	37AH	0	1	STROB
LPT 2	27AH	1	14	WYSUW PAPIERU
LPT HERC	3BEH	2	16	INICJACJA
		3	17	WYBÓR WEJŚCIA
REJESTR STANU DRUKARKI WEJŚCIOWY				
LPT 1	379H	3	15	BŁĄD
LPT 2	279H	4	13	WYBÓR
LPT HERC	3BDH	5	12	KONIEC PAPIERU
		6	10	POTWIERDZENIE
		7	11	ZAJĘTOŚĆ

wienie linii !Z/O pin 17 złącza. Bit rejestru sterującego odpowiadający temu pinowi można odczytać z tabeli 1.

Tryb 0 - zapis

W momencie zapisu linia !Z/O przyjmuje stan niski uaktywniając bufory trzystanowe w układach U3, U4, U5. Linia ta jest negowana przez bramkę U9C przechodząc w linię Z/!O, która steruje buforami trzystanowymi układów U6, U7, U8 - w tej sytuacji je blokuje. Takie rozwiązanie eliminuje możliwość jednoczesnego uaktywnienia buforów od strony komputera i końcówki emulacyjnej. Dodatkowo linia Z/!O steruje pamięciami przez wejście OE, a !Z/O przez bramkę U9A otwiera tranzystor T1 co daje informacje o ładowaniu danych w postaci świecenia diody D1. R1, R3 ograniczają odpowiednie prądy.

Etapem pierwszym zapisu danych jest podanie szesnastobitowego adresu komórki. Wykonywane jest to dwuetapowo. Najpierw ustawiamy w bramie portu młodszy bajt adresu. Ten bajt należy zapamiętać w układzie U4, a odbywa się to przez podanie impulsu dodatniego na linię ADL. W stanie wysokim są zapisywane przerzutniki D znajdujące się w U4. Następnie w bramę ustawiany jest starszy bajt adresu i pod wpływem dodatniego impulsu - tym razem na linii ADH - zapamiętywany w układzie U5. W tej chwili na wejścia adresowe pamięci jest podawany stabilny szesnastobitowy adres. Oczywiście, z uwagi na pojemność 32kB pamięci mają tylko po piętnaście bitów adresowych. Najstarszy bit adresu AD15 jest negowany bramką U9D i przez wejście CE uaktywnia do pracy odpowiedni bank pamięci. Etap drugi polega na wystawieniu bajtu przewidzianego do zapisu pod wcześniej ustawiony adres. Dane te są buforowane układem U3 i podawane na bramy obu banków. Niczemu to nie przeszkadza, bo jak wcześniej stwierdziliśmy, zawsze aktywny jest tylko jeden bank. Ponieważ na pamięci podany jest już adres z U4 i U5 oraz dane z portu przez U3, możemy przejść do etapu trzeciego polegającego jedynie na wygenerowaniu na linii !ZAP krótkiego impulsu ujemnego. Impuls ten pojawia się na wejściach WE obu banków, lecz tylko w jednym z nich powoduje zapis informacji. Na tym kończy się cykl zapisu jednej komórki. Aby zapisać całą przestrzeń pamięci symulatora należy powtórzyć taki cykl 65536 razy.

Tryb 1 - odczyt

W tym trybie linia !Z/O jest w stanie wysokim natomiast Z/!O w niskim. Skutkiem tego jest zablokowanie buforów w układach U3, U4, U5, a uaktywnienie układów U7, U8 oraz stworzenie możliwości odblokowania U6. Przez cały czas trwania trybu odczytu adresy wystawiane na magistrali adresowej układu mikroprocesorowego są przekazywane na wejścia adresowe banków. Wybór aktywnego banku odbywa się na tej samej zasadzie jak przy zapisie. Pomocna w tym jest bramka U9D. Skutkiem podania adresu na wyjściu jednego z banków pojawi się bajt wcześniej zapisanych danych. Nie jest to jednak jeszcze dostateczny powód, aby na wyjściu symulatora pojawiły się dane. EPROM podczas standardowej pracy (pomijam tu programowanie) jest sterowany dodatkowo dwoma sygnałami OE i CE. Z sygnałem CE spotkaliśmy się przy omawianiu przełączania banków. Spełnia on tą samą rolę w EPROM-ie co w każdej innej pamięci. Najprościej mówiąc, w stanie niskim zezwala na pracę. Aby odczytywać informacje z pamięci EPROM konieczne jest również wysterowanie stanem niskim wejścia OE. Chcąc zachować pełną zgodność symulatora z pojedynczą kostką pamięci wzięto pod uwagę i te niuanse. Rozwiązanie tego zagadnienia okazało się niezwykle proste. Zamiast uaktywniać bufor wyjściowy symulatora U6 bezpośrednio z linii Z/!O, zastosowano sumę logiczną sygnałów OE, CE i Z/!O na bramce U10A. Podczas spełnienia wszystkich warunków, to jest:

1. symulator pracuje w trybie 1,
2. wejście CE = L,
3. wejście OE = L,

na wyjściu bramki uzyskujemy stan wysoki, a ponieważ U6 jest uaktywniany poziomem niskim, zastosowano bramkę U10B jako negator.

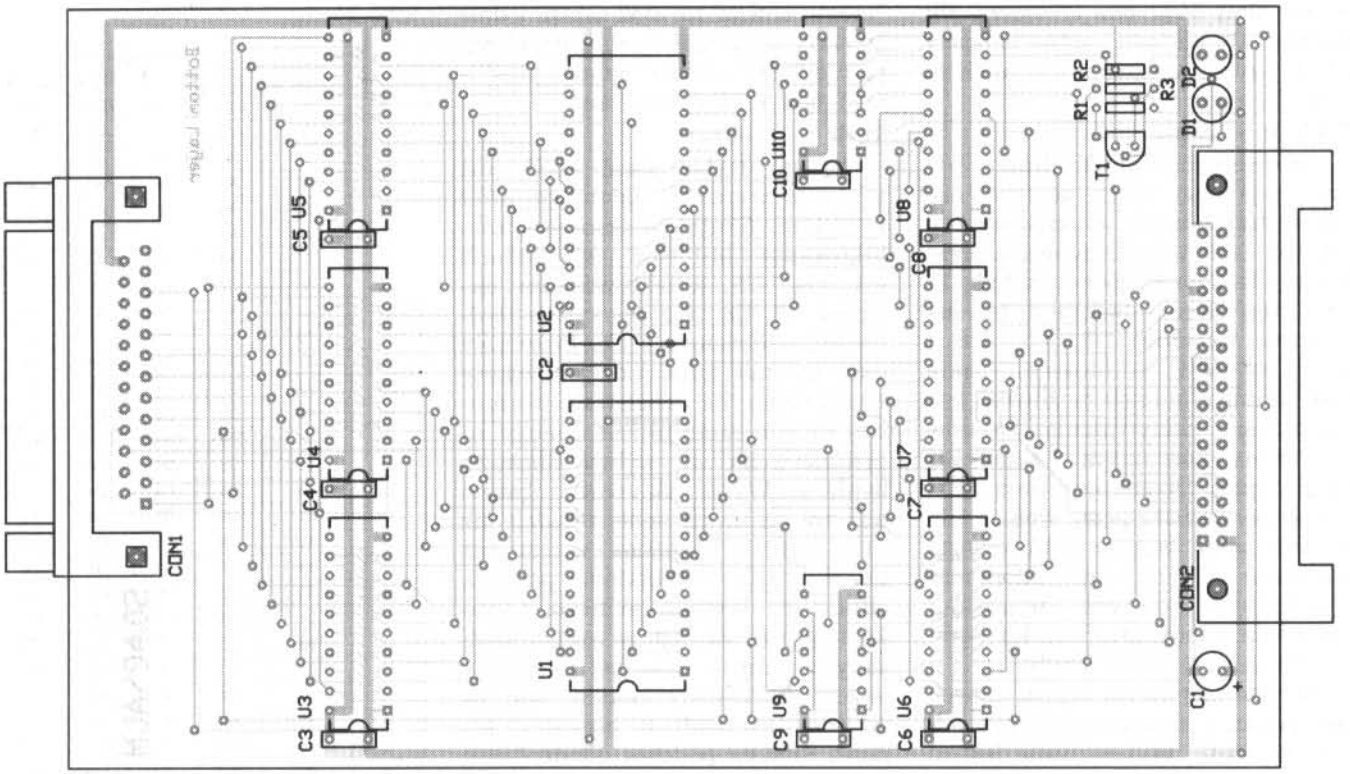
Montaż

Symulator jest zmontowany na płytce drukowanej dwustronnej o wymiarach standardowej eurokarty (**rys. 2**). Ponieważ złącza umieszczono po przeciwnych stronach płytki, nie powinno być kłopotów z umieszczeniem symulatora w obudowie. Tym bardziej, że są dostępne obudowy o takich gabarytach. Zastosowana minimalna ilość elementów dyskretnych upraszcza i przyśpiesza w znaczącym stopniu montaż. Przy każdym układzie scalonym umieszczono kondensator odsprzęgający według ogólnie przyjętych zasad. C1 to elektrolit lub tantal do eliminacji drobnych zakłóceń przychodzących przez zasilanie z uruchamianego układu. Od strony komputera zastosowano złącze DB-25RS identyczne jak w porcie drukarki. Ta cecha daje możliwość stosowania do transmisji danych standardowych kabli drukarkowych. Oczywiście, złącza muszą być identyczne z obu stron. Układ testowano nawet na kablu o długości 5m i nie stwierdzono błędów w transmisji. Aby zwiększyć elastyczność zastosowań przyrządu autor zdecydował się na użycie złącza typu DHR34 od strony wyprowadzeń symulowanego EPROM-u. W gnieździe mamy do dyspozycji 28 aktywnych pinów. Tyle nóżek ma EPROM 27512 i są one ustawione w taki sposób aby połączyć wtyczkę od gniazda z końcówką emulacyjną taśmą 28 żyłową bez zbędnych przeplotów. Kolejne piny w złączu są opisane na schemacie natomiast w **tabeli 2** zamieszczono informacje

Tab. 2.

Nr pinu		Rodzaj pamięci			
Gniazdo	Końcówka	2764	27128	27256	27512
1	1	GND	GND	GND	A15
3	2	A12	A12	A12	A12
5	3	A7	A7	A7	A7
7	4	A6	A6	A6	A6
9	5	A5	A5	A5	A5
11	6	A4	A4	A4	A4
13	7	A3	A3	A3	A3
15	8	A2	A2	A2	A2
17	9	A1	A1	A1	A1
19	10	A0	A0	A0	A0
21	11	D0	D0	D0	D0
23	12	D1	D1	D1	D1
25	13	D2	D2	D2	D2
27	14	GND	GND	GND	GND
28	15	D3	D3	D3	D3
26	16	D4	D4	D4	D4
24	17	D5	D5	D5	D5
22	18	D6	D6	D6	D6
20	19	D7	D7	D7	D7
18	20	CE	CE	CE	CE
16	21	A10	A10	A10	A10
14	22	OE	OE	OE	OE
12	23	A11	A11	A11	A11
10	24	A9	A9	A9	A9
8	25	A8	A8	A8	A8
6	26	GND	A13	A13	A13
4	27	GND	GND	A14	A14
2	28	Vcc	Vcc	Vcc	Vcc

A - adres, D - dane, Vcc - U zasilania, GND - masa



Rys. 3. Rozmieszczenie elementów na dwustronnej płytce drukowanej

```
void zapis (unsigned int Adres, unsigned char Dana) {
  outportb (0x3BC,Adres>>8);
  outportb (0x3BE,2);
  outportb (0x3BE,0);
  outportb (0x3BC,Adres);
  outportb (0x3BE,1);
  outportb (0x3BE,0);
  outportb (0x3BC,Dana);
  outportb (0x3BE,4);
  outportb (0x3BE,0);
}
```

Rys. 3. Procedura zapisująca jedną komórkę

jak wykonać połączenia emulacyjne końcówek dla różnych wielkości pojemności pamięci.

Dla mniejszych pamięci powstaje konieczność krosowania przewodów, co daje się zauważyć analizując tabelę 2. Podczas wykonywania połączeń może być pomocny rysunek gniazda - przodu z opisem pinów, co ilustruje **tabela 3**.

Program

Program sterujący został napisany w języku C i skompilowany ogólnie znanym kompilatorem Borland C++ 2.0. Jest on dostępny w wersji źródłowej i skompilowanej w ofercie kitu AVT-214. Zamieszczenie listingu całego programu w tym miejscu mija się z celem, natomiast dla amatorów programowania, chcących spróbować swoich sił, pomocny będzie listing procedury ładującej (**rys. 3**). Niezbędne są również informacje podane w tabeli 1. Znając te dane można napisać program w dowolnym języku, stosując również QBASIC dostarczany z systemem operacyjnym DOS. Warto pętlę ładującą napisać w asemblerze, aby uzyskać minimalny czas zapisu całej przestrzeni pamięci. W dołączonym programie pętla jest napisana w C i zapis 64kB danych jest odczuwalny.

Marek Jabłoński

Tab. 3.

1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33
2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
NR PINU W GNIEŹDZIE																

WYKAZ ELEMENTÓW

Rezystory
R1: 470Ω
R2: 680Ω
R3: 1,2kΩ
Kondensatory
C1: 47µF
C2...C10: 100nF
Półprzewodniki
U1, U2: 62256
U3, U6, U7, U8: 74LS451
U4, U5: 74ALS573
U9: 74LS00
U10: 74LS27
T1: BC237
D1, D2: dowolne LED-y
Różne
CON1: DB-25RS
CON2: DHR34