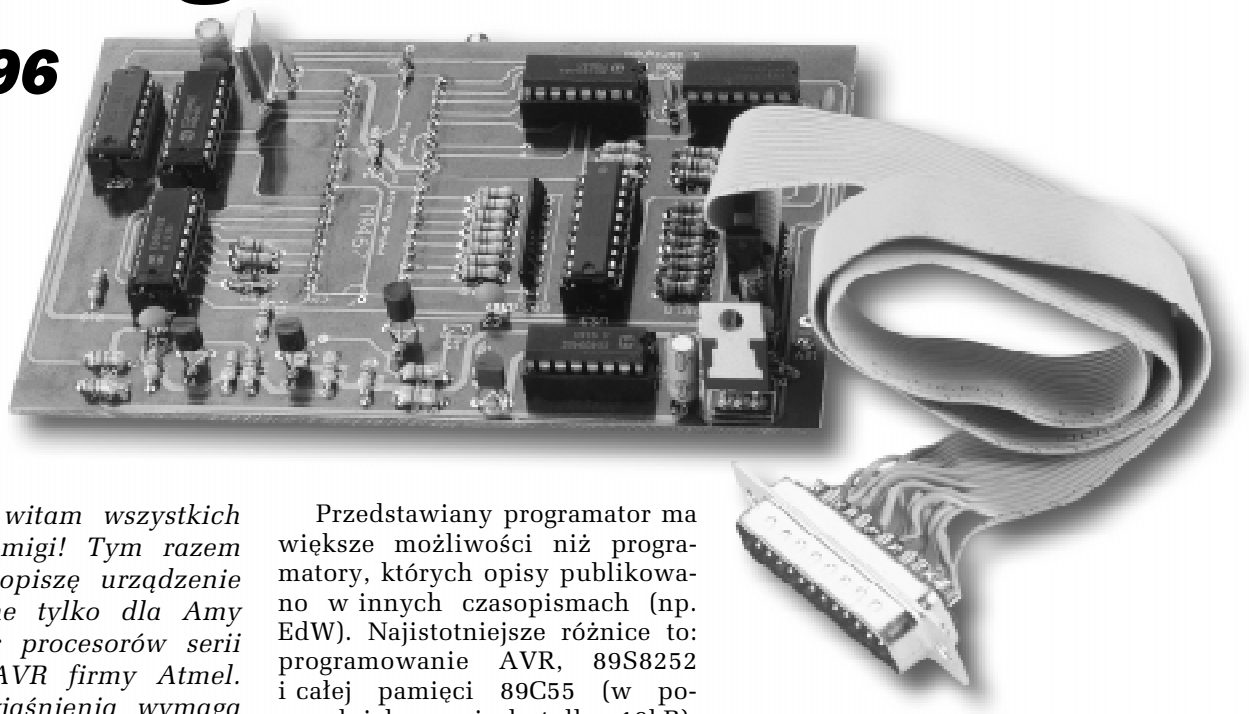
Programator 8051 & AVR do Amigi

AVT-996



Ponownie witam wszystkich miłośników Amigi! Tym razem opiszę urządzenie przeznaczone tylko dla Amy - programator procesorów serii 8051 i AVR firmy Atmel. Wyjaśnienia wymaga umożliwienie programowania procesorów AVR i AT89S8252, które daje się przecież programować w układzie przez SPI, tym bardziej, że czas programowania przez SPI jest taki sam jak w trybie równoległym. Jednak tylko tryb równoległy umożliwia ustawianie bitów RCENT, FSTRT i SPIEN. Najważniejszy jest ten ostatni, ponieważ jego skasowanie uniemożliwia przeprogramowanie układu przez SPI.

Przedstawiany programator ma większe możliwości niż programatory, których opisy publikowano w innych czasopismach (np. EdW). Najistotniejsze różnice to: programowanie AVR, 89S8252 i całej pamięci 89C55 (w poprzednich wersjach tylko 16kB).

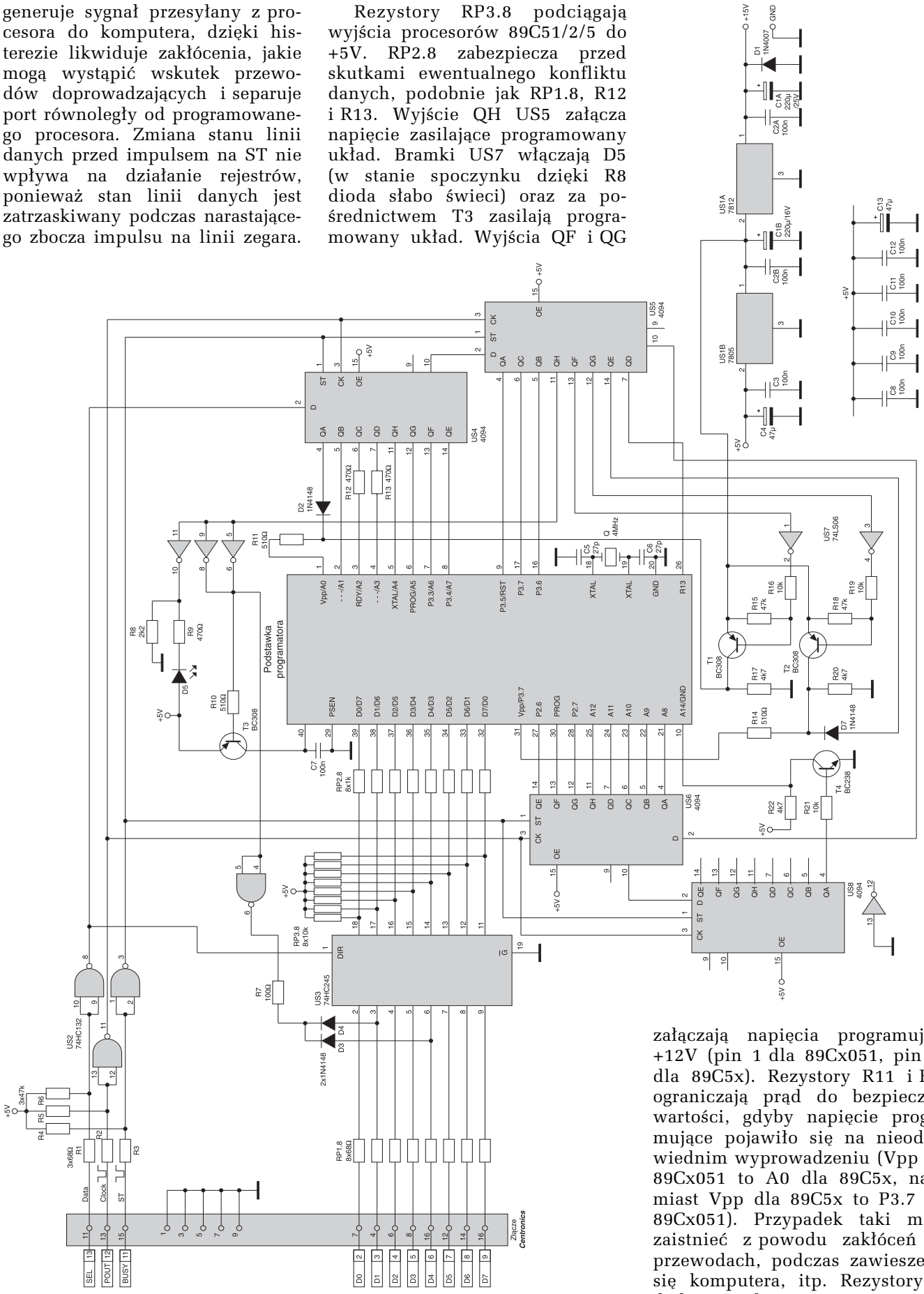
Zasada działania

Schemat elektryczny programatora przedstawiono na **rys. 1**. Urządzenie jest zasilane z zasilacza 15..18V. Napięcie nie musi być stabilizowane. Układ można też zasilić ze stabilizowanego zasilacza 12V/200mA. Wtedy w miejsce US1A należy zamontować zworę. US1B obniża napięcie do +5V do zasilania układów cyfrowych. Oznaczenie elementów z literkami A i B wprowadzono dlatego, że w poprzednich wersjach programatora nie było stabilizatora 7812 i konieczne było zasilanie go z zasilacza stabilizowanego +12V. W stosunku do modelu na fotografii zmieniono też umiejscowienie elementów R14 i D7. Dzięki temu nie jest konieczne podpiłowanie nóżek elementów, aby zamontować podstawkę zatrzaskową. D1 zabezpiecza programator przed skutkami odwrotnej polaryzacją napięcia. Dane z komputera (adres, tryb pracy, napięcia programujące) są wysyłane szeregowo do rejestrów typu 4094. Linią SEL są przesyłane dane, POUT jest przesyłany sygnał zegara, BUSY zatrzaskuje dane w rejestrach. Rejestry 4094 mają tę „przyjemną" cechę, że w swojej strukturze zawierają zatrzask. Dzięki temu stany wyjść nie zmieniają się podczas transmisji tylko po podaniu dodatniego impulsu na wejście ST. Bramki układu US2 poprawiają zbocza sygnałów przechodzących długimi jak na układy cyfrowe przewodami. Po wysłaniu informacji do rejestrów 4094, przed ich przepisaniem impulsem ST, linią SEL portu równoległego można ustawić kierunek transmisji bramy US3. Spełnia ona trzy funkcje:

Programator obsługuje następujące procesory:	
✗ AT89C51	5/12V
✗ AT89C52	5/12V
✗ AT89C55	5/12V
✗ AT89S8252	12V (także pamięć danych)
✗ AT89C1051	12V
✗ AT89C2051	12V
✗ AT89C4051	12V
✗ AT89C8051	12V
✗ AT90S1200	12V
✗ AT90S2313	12V
✗ 87C51	(tylko odczyt)
✗ 87C52	(tylko odczyt)
✗ 87C592	(tylko odczyt)

generuje sygnał przesyłany z procesora do komputera, dzięki histerezie likwiduje zakłócenia, jakie mogą wystąpić wskutek przewodów doprowadzających i separuje port równoległy od programowanego procesora. Zmiana stanu linii danych przed impulsem na ST nie wpływa na działanie rejestrów, ponieważ stan linii danych jest zatrzaskiwany podczas narastającego zbocza impulsu na linii zegara.

Rezystory RP3.8 podciągają wyjścia procesorów 89C51/2/5 do +5V. RP2.8 zabezpiecza przed skutkami ewentualnego konfliktu danych, podobnie jak RP1.8, R12 i R13. Wyjście QH US5 załącza napięcie zasilające programowany układ. Bramki US7 włączają D5 (w stanie spoczynku dzięki R8 dioda słabo świeci) oraz za pośrednictwem T3 zasilają programowany układ. Wyjścia QF i QG



Rys. 1. Schemat elektryczny programatora.

załączają napięcia programujące +12V (pin 1 dla 89Cx051, pin 31 dla 89C5x). Rezystory R11 i R14 ograniczają prąd do bezpiecznej wartości, gdyby napięcie programujące pojawiło się na nieodpowiednim wyprowadzeniu (Vpp dla 89Cx051 to A0 dla 89C5x, natomiast Vpp dla 89C5x to P3.7 dla 89Cx051). Przypadek taki może zaistnieć z powodu zakłóceń na przewodach, podczas zawieszenia się komputera, itp. Rezystory te dodano podczas pisania oprogra-

Charakterystyka programatora:

- ✓ Programator sam rozpoznaje typ procesora i producenta,
- ✓ Zapis, odczyt, weryfikacja pamięci programu ww. procesorów (w kolejnych wersjach programu obsługa pamięci danych procesorów AVR),
- ✓ Ustawianie bitów zabezpieczających i konfiguracyjnych,
- ✓ Dioda sygnalizująca stan pracy programatora,
- ✓ Pełne zabezpieczenie portów CIA Amigi,
- ✓ Procedury programowania napisane w języku maszynowym,
- ✓ Zasilanie programatora: +15V/200mA,
- ✓ Lokalizacja programu,
- ✓ Możliwość otworzenia programu na dowolnym ekranie (także na CGX).

Wymagania programu:

- ✦ OS 2.04+,
- ✦ minimum 1,1MB wolnego miejsca na dysku,
- ✦ dowolny CPU.

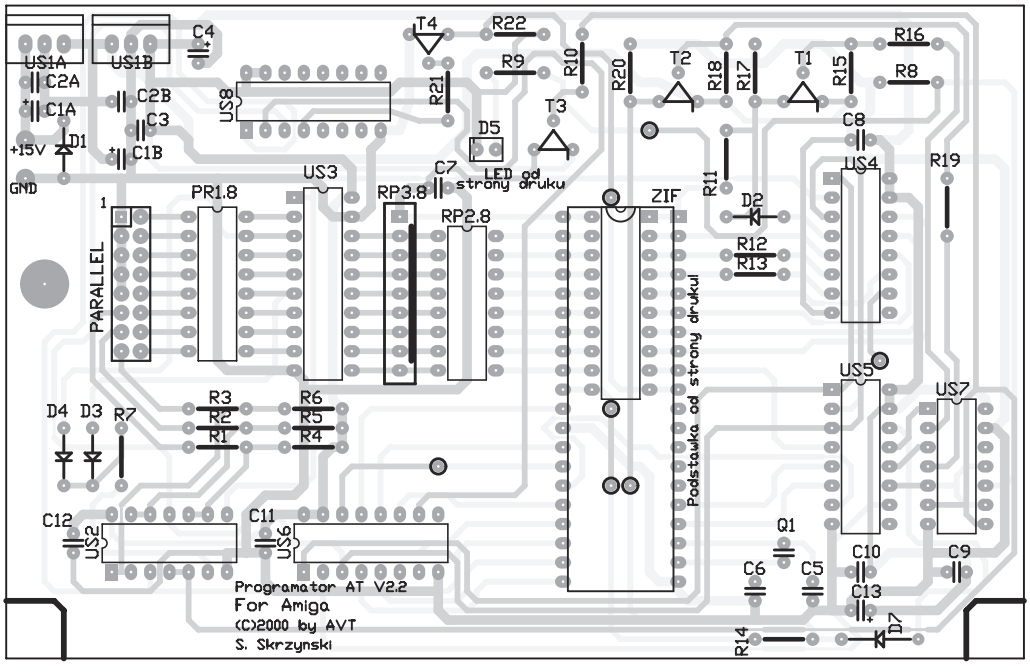
Zalecana konfiguracja komputera:

- ❑ OS 3.0+,
- ❑ CPU 68020/14MHz.

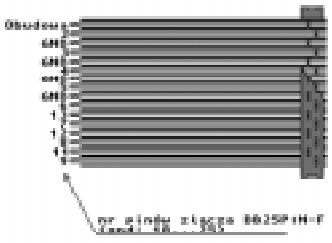
mowania w wersji V2.2. Przyczyną ich zastosowania było uszkodzenie wejścia A0 procesora 89C51 spowodowane błędem w programie. Na tym wejściu pojawiło się napięcie +12V. Wartości rezystorów dobrano tak, aby spadek napięcia 12V na wejściu Vpp nie przekraczał dopuszczalnej wartości, a jednocześnie, gdy wejście nie jest wejściem Vpp, prąd płynący z +12V do wejścia/wyjścia A0 nie przekraczał dopuszczalnej wartości powodującej uszkodzenie układu. O skuteczności zabezpieczenia autor przekonał się podczas pisania oprogramowania w wersji V2.5, gdy kilkakrotnie na wejściu P3.7 89C2051 pojawiło się napięcie +12V. Zgodnie z obliczeniami układ „przeżył". Diody D2 i D7 separują wyjścia rejestrów 4094 od napięcia +12V. Rezystory R17 i R20 wymuszają poziom niski, gdy na piny Vpp nie jest podawany wysoki poziom logiczny ani +12V. Wyjście QA US8 steruje za pośrednictwem T4 linią A14 układu 89C55 lub GND dla 89Cx051. Tranzystor jest konieczny, ponieważ wydajność prądowa układu US8 jest za mała do zasilenia programowanego procesora. Na koniec została do omówienia funkcja spełniana przez jedną z bramek US2. Za pośrednictwem R7, D3 i D4 umożliwia ona wykrycie przez program dołączenia programatora i jego wersji. Dzięki temu zapis 89C55 zostanie ograniczony do 16kB dla wersji hardware v2.0, o czym użytkownik zostanie poinformowany. Gdyby program nie wprowadził tego ograniczenia przy przekroczeniu 16kB, procesor zostałby źle zaprogramowany, a program pracowałby źle. Jeśli układ US2 będzie wykonany w technologii TTL (np. 74LS132), konieczne może okazać się zastosowanie diod D3, D4 wykonanych w technologii Schottky'ego (np. BAT85). Spowodowane jest to maksymalnym napięciem wyjścia w stanie niskim. W układach C-MOS napięcie to wynosi maksymalnie 0,33V. Dodając spadek na diodzie krzemowej 0,7V, razem 0,93V. Napięcie to z pewnością będzie zinterpretowane przez układ CIA jako stan niski. Gdy natomiast US2 jest wykonany w technologii TTL, maksymalne napięcie wyjściowe w stanie niskim wynosi 0,4V, spadek na diodzie krzemowej 0,7V, razem 1,1V. Gwarantowane napięcie wejściowe, jakie będzie zinterpretowane jako poziom niski wynosi 1V. Jak widać kombinacja z US2-TTL i D3, D4 (diody krzemowe) nie gwarantuje rozpoznania przez CIA takiego poziomu jako niski. Gdy zastosujemy diody Schottky'ego, otrzymamy następujące wyniki: napięcie wyjściowe w stanie niskim 0,4V, spadek napięcia na diodzie 0,4V, razem 0,8V. W tym przypadku wszystko jest OK. Podałem tu skrajne przypadki. Natrafienie na taką kombinację spadków napięć jest mało prawdopodobne, ale możliwe, zwłaszcza że trzeba wziąć pod uwagę mały (ale jednak) spadek napięcia na rezystorach RP1.8. Gdyby więc program testujący nie wykrywał programatora, przyczyną może być wspomniana niekorzystna kombinacja spadków napięć.

Montaż i uruchomienie

Schemat montażowy programatora pokazano na **rys. 2**. Montaż rozpoczynamy od elementów najmniejszych do największych półprzewodników i złącz. Dzięki temu, że podstawkę zatrzaskową i diodę LED umieszczono od strony druku nie ma kłopotu z zamknięciem urządzenia w obudowie typu KM60. Płytkę można wykonać jako dwustronną bez metalizacji otworów. Należy jednak pamiętać o wykonaniu siedmiu przelotek oznaczonych na płytce kółkami po stronie opisu. Funkcję pozostałych przelotek spełniają nóżki elementów. W tym



Rys. 2. Rozmieszczenie elementów na płytce drukowanej.



Rys. 3. Sposób wykonania kabla połączeniowego.

wypadku pod układy scalone należy zastosować podstawki precyzyjne, które można lutować także od strony druku. Podstawkę zatrzaskową i złącze IDC można lutować tylko z jednej strony. Do tych elementów ścieżki prowadzone są z jednej strony. Niektórzy nie lutują wyprowadzeń układów, złącz itp., które nie są do niczego podłączone. W tym wypadku nie zalecam takiego postępowania, ponieważ płytka jest dwustronna i to, że z jednej strony nie ma połączeń nie oznacza, że z drugiej też ich nie będzie. W podstawce zatrzaskowej, ze względu na wymaganą wytrzymałość mechaniczną, należy konieczne przylutować wszystkie wyprowadzenia. Przy zakupie podstawki należy zwrócić uwagę, aby można było umieścić w niej układy o szerokości 0,3 cala (20 pin) i 0,6 cala (40 pin). Aby obniżyć koszt programatora, można zastosować dwie podstawki precyzyjne (20 i 40 pin). Wygodniej jest jednak umieścić tam listwy tulipanowe (tzw. goldpiny) ze względu na trudności umieszczenia podstawki 20 pin w podstawce 40 pin. Sposób połączenia złącza DB25 z taśmą przedstawiono na **rys. 3**. Przed umieszczeniem układów w podstawkach włączamy zasilanie i sprawdzamy napięcie +5V. Jeśli wszystko jest dobrze, to po wyłączeniu zasilania umieszczamy układy w podstawkach, podłączamy urządzenie do komputera (przy wyłączonym zasilaniu tak komputera, jak i programatora). Włączamy zasilanie programatora i komputera.

Instalacja i testowanie

Zawartość foldera dyskietki po rozpakowaniu przedstawiono na **rys. 4**. W skład zestawu wchodzą następujące programy:

- „Prog_AT“ - jest to program główny, dzięki któremu można programować procesor kodem wczytanym z dyskietki itp.,
- „TestProg_AT“ - program testujący programator,
- „Rysunki“ - w tym katalogu znajdują się schematy (IffILBM) i rysunki płytek (Platine, IffILBM, AutoTrax) programatora, sterownika węża świetlnego i emulatora zegara DCF,
- „Źródła“ - w katalogu znajdują się kody źródłowe programu „Prog_AT“ i “TestProg_AT“ napisane w AmigaE oraz przykładowe programy dla 8051 (sterownik węża świetlnego i emulator zegara DCF),
- „Przykłady“ - w katalogu zawarte są przykładowe pliki IntelHex,
- „Instaluj na HD“ - program instalujący,
- „Instrukcja.guide“ - instrukcja do programu w formacie AmigaGuide. Tę instrukcję warto przeczytać przed instalacją programu. Opisano tam szczegółowo:
  - zmiany poczynione w kolejnych wersjach software i hardware,
  - sposób wydrukowania rysunków płytek, jeśli wykonujemy je sami,
  - szczegóły dotyczące instalacji i lokalizacji programu,
  - sposób otrzymywania darmowego uaktualnienia oprogramowania,
  - miejsce, z którego można ściągnąć kompilatory 8051, AVR i inne,
  - możliwe błędy i sposób ich usunięcia.

Instrukcja zajmuje 32kB tekstu ASCII, więc jest co czytać, a program jest ciągle rozwijany, warto więc otrzymywać upgrade. Program instalujemy standardowym instalerem, klikając na ikonę „Instaluj na HD“. Po instalacji uruchamiamy program „TestProg_AT“. Po tekście ostrzegawczym pojawi się okno z gadżetami (**rys. 5**). Po kliknięciu na linię gadżetów pojawi się okno z informacją, jakie stany powinny wystąpić na pinach układów US4, US5, US6, US8. Do badania stanów logicznych najlepsza będzie sonda logiczna. Napięcia na wyprowadzeniach 1 i 31 sprawdzamy miernikiem uniwersalnym. Jeśli wszystko jest dobrze, naciskamy gadżet *Dalej*, co zmieni stany na pinach rejestrów 4094. Test ten umożliwia sprawdzenie US2, US4, US5, US6, US8. Po sprawdzeniu wszystkich kombinacji pojawi się okno główne (**rys. 6**).

Gadżet Vpp umożliwia sprawdzenie układu US7 sterującego załączaniem napięcia zasilającego i programującego.

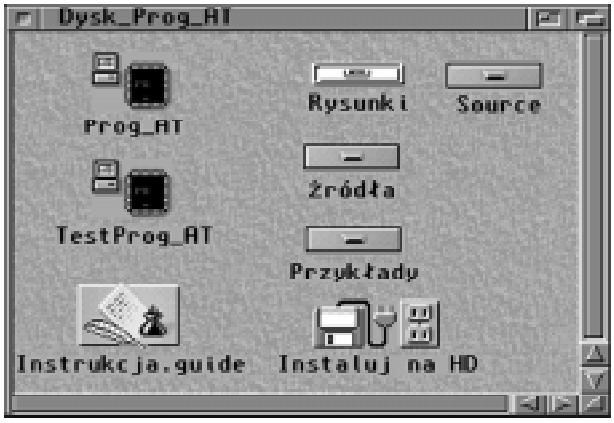
Gadżet Poziomy umożliwia sprawdzenie US3 i rejestrów 4094 oraz wyeliminowanie ewentualnych przerw i zwarć w ich obwodach.

Gadżet Wersja testuje układ wykrywający programator.

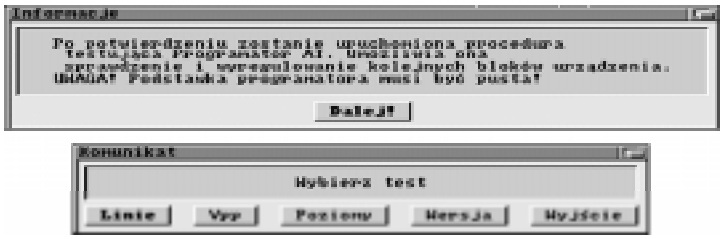
Jeśli test przebiegł prawidłowo, to prawdopodobnie nie popełniliśmy błędu i urządzenie zadziała poprawnie. Wychodzimy więc z programu testującego i uruchamiamy program „Prog_AT“.

Obsługa programu

Po komunikacie powitalnym ukaże się główne okno programu (rys. 6). Program rezerwuje port równoległy, tak więc inne programy nie będą mogły z niego korzystać (zakomunikują błąd: „Nie mogę otworzyć parallel.device“, czy też „Nie mogę otworzyć printer.device“). Jeśli w czasie, gdy uruchomimy program, port będzie zarezerwowany, pojawi się komunikat o niemożliwości jego wykorzystania, a program przerwie swe działanie. Program korzysta także z timerów CIA i bibliotek: *dos.library*, *intuition.library*, *graphic.lib-*



Rys. 4. Zawartość foldera dyskietki po rozpakowaniu.



Rys. 5. Okno z gadżetami.

rary, *icon.library*, *gadtools.library*, *reqtools.library*. Większość z nich znajduje się w ROM-ie. Przeważnie *reqtools.library* i *gadtools.library* są bibliotekami zewnętrznymi, ale stały się one niejako standardem i z pewnością znajdują się na każdym dysku (99% programów wymaga tych bibliotek).

Po uruchomieniu programu otworzy się okno informacyjne, po wyjściu z którego mamy do wyboru opcje:

Otwórz plik - otwiera okno umożliwiające wczytanie plików *bin* lub IntelHex z dysku do bufora programu. Format pliku jest rozpoznawany automatycznie. Akceptowane są także pliki, których dane są niespójne, np.:

dane 1 - $0000-$0F00
dane 2 - $1000-$1100
dane 3 - $2000-$3FFF

Ważne jest, aby bloki nie zachodziły na siebie, a ich adresy były rosnące. W kolejnych wersjach programu ograniczenie to zostanie usunięte. Kolejność występowania bloków nie ma znaczenia. Rekordy danych mogą zawierać od 1 do 255 bajtów. W programach na PC często ww. opcje nie są akceptowane, co przeczy idei plików IntelHex (np. w MCS Basic-emulator dane muszą być zawarte w jednym bloku). Program akceptuje kody końca linii CR, LF i CR+LF (LF+CR). Dzięki takiemu rozwiązaniu możliwe jest odczytanie kodu generowanego przez kompilatory na innych platformach. Budowa pliku IntelHex jest dosyć prosta - składa się z rekordów. Na rekord składają się liczby *hex* zapisane w kodach ASCII według wzoru:

:cc aaaa tt dd.....dd ss

gdzie:

: - znacznik rekordu IntelHex,
cc - liczba danych (dd),
aaaa - 16-bitowy adres ładowania,
tt - typ rekordu:

00 - rekord z danymi
01 - ostatni rekord sygnalizujący koniec pliku (tworzy sekwencję „:00000001FF“)
02 - dane w formacie IntelExtended
03 - adres startowy dla 8086

dd - dane,
ss - suma kontrolna modulo 256 obejmująca cc aaaa tt dd.

Zapisz plik - zapisuje plik w formacie binarnym z bufora na dysk. Jeśli podejmiemy próbę zapisu do istniejącego pliku, to pojawi się komunikat ostrzegawczy z możliwością wyboru (**rys. 7**).

Zapisz procesor - zapisuje procesor kodem z bufora. Jeśli bufor jest pusty, pojawi się komunikat ostrzegawczy, po czym procedura zostanie przerwana. Jeśli w buforze znajduje się program, zostanie odczytana sygnatura procesora i dobrany odpowiedni algorytm. Następnie pamięć procesora zostanie skasowana oraz będzie sprawdzona poprawność tej operacji, po czym procesor zostanie zaprogramowany. Podczas programowania wyświetlana jest informacja, który bajt jaką wartością jest programowany. Jeśli wystąpi błąd, w oknie programu możemy zobaczyć adres, pod którym występuje błąd, jaka wartość bajtu znajduje się w buforze i jaka została zapisana w procesorze. Po zapisaniu całego bufora pojawi się pytanie, czy programować bity blokady, czy nie. Programowane są zawsze bity 1 i 2. Jeśli chcemy zaprogramować tylko bit 1 lub bity 1, 2 i 3, musimy na pytanie czy programować bity blokady odpowiedzieć nie, po czym wybrać odpowiedni gadżet lub skrót klawiszowy ustawiający wybrane bity blokady.

Czytaj procesor - odczytuje całą zawartość procesora do bufora, po czym otwiera okno umożliwiające zapisanie pliku w formacie binarnym. Jeśli procesor jest „czysty“ lub ma ustawione bity blokady, pojawi się komunikat informujący o tym, a procedura zostanie przerwana.

Weryfikuj - porównuje pamięć procesora z zawartością bufora. Jeśli procesor ma ustawione bity zabezpieczające, operacja ta nie powiedzie się.

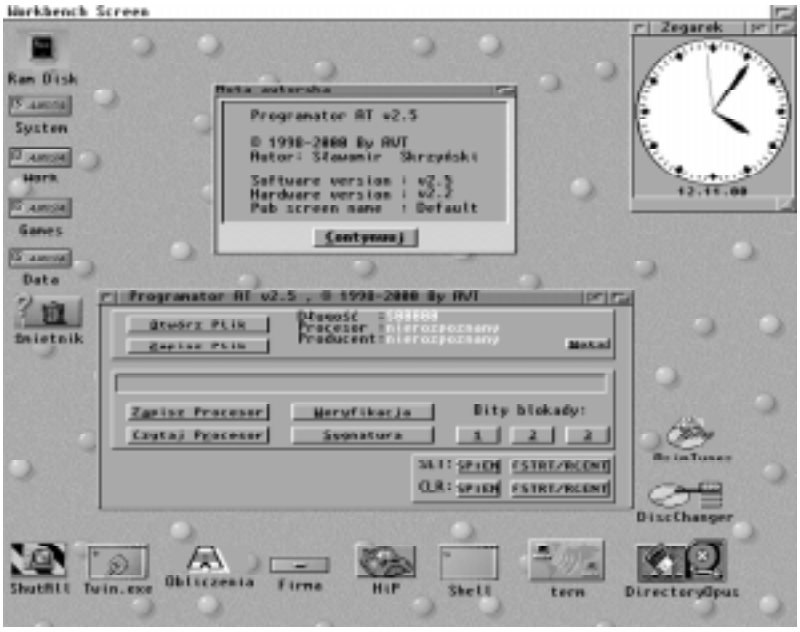
Sygnatura - określa wersję procesora, producenta i napięcie programujące. Odczytywany jest także stan bitów konfiguracyjnych i blokady (dla 8051 tylko bity konfiguracyjne). Operacja ta jest przeprowadzana przed każdą operacją na procesorze (zapis, odczyt, weryfikacja, ustawianie bitów zabezpieczających).

Nota - wyświetla informację o autorze, wersji software, wersji hardware, nazwę ekranu, na którym program otworzył swoje okno (napis „Default“, jeśli program otworzył okno na ekranie domyślnym).

Bity blokady - wybranie odpowiedniego gadżetu. Programuje bi-

Tab. 1.

Dla procesorów AT89C5x i AT89S8252	
Ustawiony bit blokady	**Rodzaj ochrony**
żaden	Brak ochrony
1	Instrukcje MOVC z pamięci zewnętrznej nie mogą pobierać bajtów kodu z pamięci wewnętrznej. Przeprogramowywanie pamięci FLASH zabronione.
1 i 2	Jak wyżej, ponadto zabroniona weryfikacja
1, 2 i 3	Jak wyżej, ponadto zabronione wykonywanie programu z zewnętrznej pamięci programu (zablokowany rozkaz MOVEX)
Dla procesorów AT89Cx051 i AVR	
Ustawiony bit blokady	**Rodzaj ochrony**
żaden	Brak ochrony
1	Programowanie pamięci FLASH zabronione
1 i 2	Jak wyżej, ponadto weryfikacja FLASH zabroniona



Rys. 6. Główne okno programu.

ty blokady, których znaczenie pokazano w **tab. 1**.

Wybranie gadżetu programującego bit 2 programuje także bit 1. Wybranie gadżetu programującego bit 3 programuje także bity 1 i 2. Jeśli w podstawce znajduje się procesor 89Cx051, a wybrany zostanie bit blokady 3, pojawi się komunikat o niemożliwości wykonania takiej operacji (procesory w obudowach DIL-20 mają tylko bity 1 i 2). Po operacji programowania bitów zabezpieczających przeprowadzana jest weryfikacja jej skuteczności. Bity blokady można skasować tylko kasując pamięć programu (i danych) procesora. Czynność ta jest wykonywana automatycznie przed programowaniem procesora.

Bity konfiguracyjne - „kliknięcie“ na wybrany gadżet w wierszu CLR powoduje skasowanie wybranego bitu. „Kliknięcie“ na gadżet w wierszu SET powoduje ustawienie wybranego bitu. Jako ustawiony jest traktowany stan „L“, skasowany - stan „H“. Jeśli wybierzemy ustawienie bitu blokady, który w danym procesorze nie istnieje, program poinformuje nas o niemożliwości wykonania takiej operacji. Bity RCENT i FSTRT są obsługiwane tym samym gadżetem. O tym, który z nich zostanie obsłużony decyduje typ programowanego procesora (**tab. 2**).

Podczas operacji kasowania pamięci (co jest przeprowadzane przed każdym programowaniem) kasowane są również bity konfiguracyjne. Uniemożliwiłoby to programowanie procesorów przez SPI. Programator po udanej operacji programowania procesora ustawia bit SPIEN. Dzięki temu możliwe jest programowanie szeregowe i tylko świadomie można z niego zrezygnować.

Dodatkowe wskazówki

1. Wszystkie funkcje można wywołać, naciskając prawy klawisz Amiga i klawisz z podkreśleniem na gadżecie (np. Otwórz plik PrawaAmiga+o, Zapisz procesor PrawaAmiga+a).

2. Wszystkie operacje (zapis, odczyt, weryfikacja) można przerwać, naciskając lewy przycisk myszy (w kolejnych wersjach programu lewy i prawy równocześnie).

3. Jeśli program testujący nie wykrywa błędu, a procesor nie programuje się poprawnie, to przyczyny mogą być dwie: uszkodzony procesor lub zbyt długie przewody. W modelowym wykonaniu przy przewodach o długości 3,6m programator działał prawidłowo.

4. W *tooltypes* (parametry) ikony można ustawić kilka parametrów:

SCREEN_NAME - ekran, na którym program otworzy swoje okno (domyślnie ekran *default*, czyli przeważnie ekran Workbencha).

DISPLAYID - jeśli w SCREEN_NAME wpiszemy nazwę ekranu który nie istnieje, program otworzy swój ekran. DISPLAYID określi jego typ. Program akceptuje zarówno nazwy słowne (np. *pal:low res*, *dblpal:high res no flicker*), jak i opis liczbowy użyteczny w nietypowych trybach ekranowych). Domyślne ustawienie *pal:high res*.

TEST - ustawienie na YES uruchomi program w trybie testu. Opcja zarezerwowana dla serwisu. Ustawienie domyślne: NO.

DEVICE_TEST - ustawienie na NO pominie odczyt wersji urządzenia podczas uruchamiania. Może to być przydatne podczas testowania urządzenia. Ustawienie domyślne: YES.

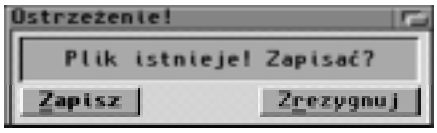
Lokalizacja programu

Program umożliwia jego lokalizację. Aby tego dokonać, systemowym programem ustawiającym preferencje „Locale“ wybieramy żądaną wersję językową. Jeśli odpowiedni katalog znajduje się na dysku, to program będzie się z nami komunikował w wybranym języku. Stworzenie pliku lokalizacji nie jest trudne. Wystarczy edytor tekstu (np: CED, AmiTekst) i dobre chęci. Wygląd pliku lokalizacji można podejrzeć („*Locale/catalogs/polski/programator_at_v2.catalog*“). Załóżmy, że chcemy stworzyć niemiecką wersję programu. W tym celu najlepiej skopiować plik „*Locale/catalogs/polski/programator_at_v2.catalog*“ do „*Locale/catalogs/deutsh/programator_at_v2.catalog*“ i odpowiednio go zmodyfikować (pamiętamy oczywiście o ustawieniu wybranego języka w preferencjach).

Tab. 2. Występowanie bitów konfiguracyjnych

Typ procesora	Bit konfiguracyjny
AT89S8252	SPIEN
AT90S1200	RCENT, SPIEN
AT90S2313	FSTRT, SPIEN
AT90S8515	FSTRT, SPIEN

Funkcje bitów konfiguracyjnych:
RCENT - ustawienie konfiguruje wejście generatora jako RC, skasowanie generator kwarcowy,
SPIEN - ustawienie umożliwia programowanie procesora w układzie przez SPI, skasowanie blokuje programowanie przez SPI,
FSTRT - ustawienie skraca czas wewnętrznego impulsu zerującego z 16 na 1,1ms (typowo przy Uzas +5V)



Rys. 7. Okno potwierdzenia.

Co należy wiedzieć o pliku lokalizacji?

Najważniejsze to:
- Interpretowane są teksty zawarte w cudzysłowie występujące po etykiecie.
- Symbol po znaku podkreślenia określa skrót klawiszowy.
- Etykieta ułatwia orientację w pliku i musi się kończyć znakiem „:“ (dwukropka).
- Kolejność etykiet jest dowolna i nie wszystkie muszą występować (można zmodyfikować część tekstów). W takim przypadku pod niewystępujące etykiety zostaną podstawione domyślne teksty.
- Jeśli etykieta powtórzy się, to pod uwagę będzie brana ostatnia.

Wyjaśnijmy te niezbyt oczywiste objaśnienia na przykładzie:
Linia:
TXG_LOAD: „_Load“
będzie zinterpretowana jako:
- gadżet „Load“,
- skrót klawiszowy „L“.

Linia:
TXG_WRITE: „Z_apisz procesor“
będzie zinterpretowana jako:
- gadżet „Zapisz procesor“,
- skrót klawiszowy „a“.

Linia:
TXG_QUIT: „Exit“
będzie zinterpretowana jako:
- gadżet „Exit“,
- bez skrótu klawiszowego.

Inaczej jest, gdy znak podkreślenia jest przedostatnim w interpretowanym tekście.
Linia:
Etykieta: „Długość_3“
będzie zinterpretowany jako:
- gadżet „Długość“,
- skrót klawiszowy „3“.

W tekstach należy zwrócić uwagę na:
- znaki „|“ pomiędzy niektórymi tekstami, które muszą wystąpić, ponieważ oddzielają poszczególne opcje w *requesterach*,
- sekwencje „\n“ formatującą wydruk, oznaczającą nic innego jak kod znaku *Enter*.

Brak pliku lokalizacji powoduje ustawienie polskiej wersji językowej. Gdy w katalogu „*Locale/catalogs/polski/*“ znajdzie się plik z polską wersją językową, to będzie on zinterpretowany. Umożliwia to zmianę skrótów klawiszowych czy wręcz wszystkich dostępnych komunikatów, gadżetów i menu.

Zmiany

Opisana tu wersja programatora nosi numer 2.2. W wersji 2.0 brak jest układu US8, linia A14 (gnd dla 89Cx051) jest na stałe podłączona do masy. Dioda D4 jest podłączona do pinu 4 US3. Taka wersja poprawnie współpracuje z oprogramowaniem w wersji 2.5, ale w procesorze AT89C55 można obsłużyć tylko 16 z 20kB pamięci. W wersji sprzętu 2.1 wystąpił błąd, który ujawnia się podczas programowania niektórych procesorów (zwłaszcza w obudowach z 40 wyprowadzeniami). W V2.2 błąd ten jest usunięty. Użytkownicy poprzednich wersji sprzętu muszą odciąć połączenie pinów 4 i 5 US2 z pinem 11 US2. Piny 4 i 5 US2 podłączamy do pinu 6 US7. Jeśli załączaniem zasilania steruje kontaktron (wersja hardware V2.0), należy go usunąć, a na jego miejsce zastosować układ z tranzystorem T3 (na płytce V2.0 jest miejsce na T3 i R10). Jeśli ktoś nie ma ochoty na przecinanie ścieżek, wystarczy usunąć R7, a w tooltypes ikony (parametry) ustawić *DEVICE_TEST=NO*.

Sławomir Skrzyński, AVT
slawomir.skrzynski@ep.com.pl

Pamiętajcie, że w razie jakichkolwiek kłopotów z uruchomieniem czy użytkowaniem programatora możecie zwrócić się o pomoc do autora. Czekam także na propozycje i uwagi dotyczące urządzenia i programu. Jeśli macie propozycje urządzeń dla Amigi, piszcie na adres redakcji.

Wzory płytek drukowanych w formacie PDF są dostępne w Internecie pod adresem: ***http://www.ep.com.pl/?pdf/luty01.htm*** *oraz na płycie CD-EP02/2001B w katalogu* ***PCB****.*

WYKAZ ELEMENTÓW

Rezystory
R1..R3: 68Ω
R4..R6, R15, R18: 47kΩ
R7: 100Ω
R8: 2,2kΩ
R9, R12, R13: 470Ω
R10, R11, R14: 510Ω
R16, R19, R21: 10kΩ
R17, R20, R22: 4,7kΩ
RP1.8: R-PACK 8*64Ω obudowa DIL
RP2.8: R-PACK 8*1kΩ obudowa DIL
RP3.8: R-PACK 8*10kΩ obudowa SIL

Kondensatory
C1A: 220µF/25V
C1B: 220µF/16V
C2A, C2B, C3, C7..C12: 100nF
C4, C13: 47µF
C5, C6: 27pF

Półprzewodniki
D1: 1N4007
D2..D4, D7: 1N4148
D5: LED
D6: nie występuje w tej wersji urządzenia
T1..T3: BC308
T4: BC238
US1A: 7812
US1B: 7805
US2: 74HC132
US3: 74HCT245
US4..US6, US8: 4094
US7: 74LS06

Różne
Q: 4MHz
Podstawka TEXTOOL 40 pin 0,3-0,6 cala
Obudowa KM-60
Zasilacz 15..18V/200mA
Złącze DB25M
Złącze ZFC16
Przewód taśmowy ok. 50 cm