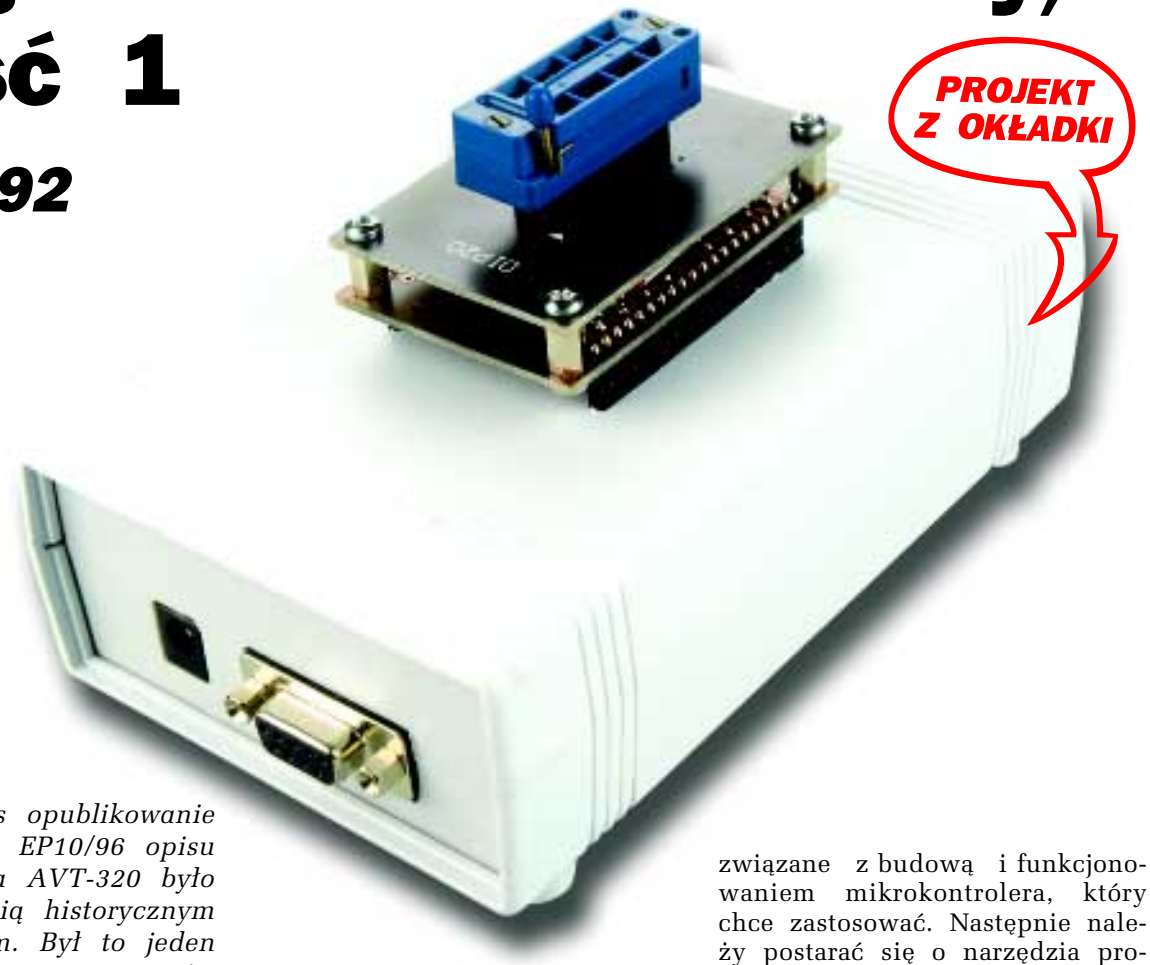
Punch Programator uniwersalny, część 1

AVT-5092



Dla nas opublikowanie w EP10/96 opisu programatora AVT-320 było z pewnością historycznym wydarzeniem. Był to jeden z pierwszych programatorów umożliwiających programowanie mikrokontrolerów '51 z pamięcią Flash. Teraz przedstawiamy jego godnego następcę - programator, który nazwaliśmy Punch, co w języku angielskim oznacza perforator do kart papierowych - niegdyś nośników programów i danych dla komputerów. ***Rekomendacje****: programator jest podstawowym przyrządem w pracowni elektronika, a więc ten opis może zainteresować większość naszych Czytelników.*

W artykule przedstawiono projekt uniwersalnego programatora elementów półprzewodnikowych działającego w oparciu o skryptowy język poleceń FEMTO. Programator umożliwia programowanie elementów, które zawiera jego biblioteka, a także wprowadzenia własnych procedur. Obecnie dostępne są skrypty dla takich elementów jak mikrokontrolery Atmela: AT89C1051/2051/4051, AT89C51/52/55, AVR-ów AT90S2313, AT90S8515, AT90S8535 i pokrewnych, a także szeregowych pamięci EEPROM z interfejsem I^2C.

Każdy, kto poważnie myśli o konstruowaniu własnych urządzeń wykorzystujących mikroprocesory oraz chciałby samodzielnie pisać dla nich oprogramowanie, musi przygotować się do tego przedsięwzięcia.

Musi przede wszystkim „zdobyć" dane techniczne i informacje związane z budową i funkcjonowaniem mikrokontrolera, który chce zastosować. Następnie należy postarać się o narzędzia programistyczne (kompilatory) oraz programator.

Karty katalogowe mikrokontrolerów oraz oprogramowanie narzędziowe można znaleźć w Internecie - chociażby na stronach producentów danego podzespołu.

Programator trzeba niestety kupić, ale można go także wykonać samemu. Wiele mikrokontrolerów można programować po zamontowaniu w systemie za pośrednictwem interfejsu ISP. Programatory, w których wykorzystuje się ten sposób programowania, są bardzo proste w wykonaniu. Kilka z nich opisaliśmy na łamach EP.

Jeżeli jednak trzeba będzie zaprogramować element, który nie ma takiej magistrali, albo w urządzeniu znajdzie się inny programowalny element, np. pamięć EPROM, konieczne jest posiadanie standardowego programatora.

To właśnie zainspirowało mnie do opracowania własnego programatora.

Dobry programator, czyli jaki?

Moim zdaniem dobry programator powinien charakteryzować się następującymi cechami:
- niezawodnością,
- dużą liczbą programowanych typów elementów,
- niską ceną,

a także poręcznością, łatwością obsługi, dostępnością serwisu i pomocy ze strony producenta oraz estetycznym wyglądem. Niestety, niektóre z tych wymagań trudno ze sobą pogodzić. Rozbudowane programatory, bogate w zaawansowane opcje, są bardzo drogie, a ze wsparciem producentów - szczególnie zagranicznych - różnie bywa. Mając to na uwadze, może warto pokusić się o skonstruowanie własnego programatora, który nie będzie konkurencyjny z produktami renomowanych firm, będzie jednak prosty, uniwersalny i w miarę niedrogi?

Opis układu

Efektem dwóch lat prób i nieustającego dopingu ze strony zespołu Elektroniki Praktycznej jest programator, którego schemat pokazano na **rys. 1**. Jego najważniejsze parametry są następujące:
- napięcie programujące ustawiane programowo w przedziale od 3 do 13 V,
- napięcie zasilania programowanego elementu ustawiane programowo w przedziale od 2 do 7 V,
- najkrótszy możliwy czas trwania impulsu potrzebnego do programowania elementu wynosi 100 ns,
- zewnętrzne napięcie zasilania programatora powinno mieć wartości: +16 VDC lub 12 VAC,
- możliwość samodzielnego wykonania prostych adapterów przeznaczonych dla różnego typu obudów programowanych elementów,
- możliwość samodzielnego pisania skryptów w języku poleceń FEMTO dla nowych elementów (skrypty są „czystymi" plikami tekstowymi) - ograniczenia w przystosowaniu programatora do obsługi nowych elementów wiążą się jedynie z jego możliwościami technicznymi (patrz wyżej),
- współpraca programatora z komputerem PC sterującym jego pracą.

Programator służy do programowania elementów z równoległą, 8-bitową magistralą danych lub z magistralą szeregową, np. I^2C. Pojemność pamięci programowanych elementów może mieć 64 kB lub więcej. Sygnały wyjściowe programatora przypisane są na stałe do wyprowadzeń jego zewnętrznych gniazd JP1 i JP2, toteż do programowania konkretnego elementu należy przygotować adapter służący do połączenia wyprowadzeń odpowiednich gniazd z wyprowadzeniami poszczególnych nóżek elementu. Samodzielne przygotowanie adapterów nie jest trudne ani specjalnie kosztowne.

Dla omówienia budowy urządzenia i istoty języka FEMTO należy przypomnieć schemat w czasie programowania typowego elementu. Najlepiej zrobić to na stosunkowo prostym przykładzie, np. równoległej pamięci EPROM 2764.

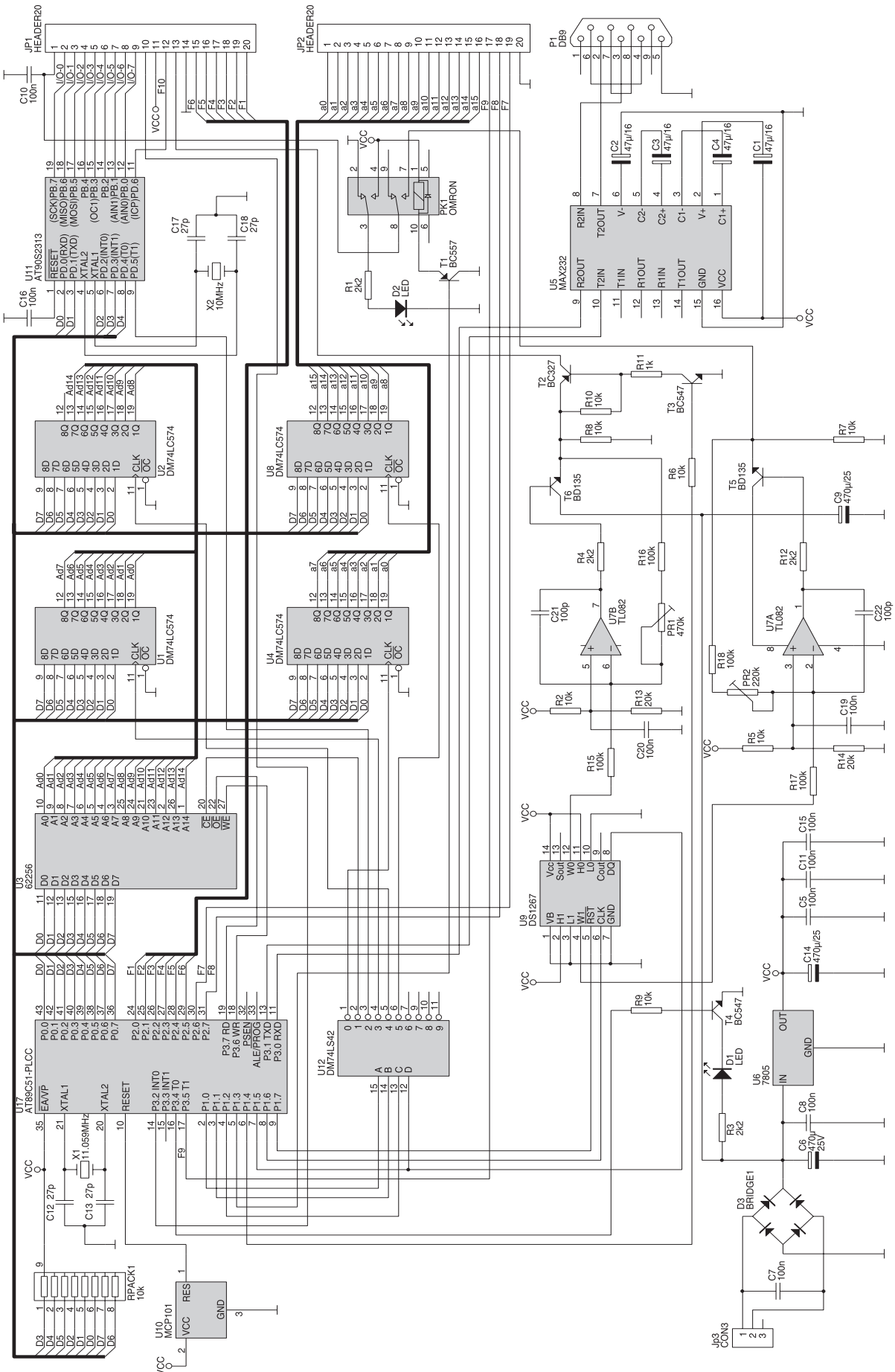
W procesie programowania jest szereg pętli składających się z kilku prostych czynności. Najpierw należy podać napięcie zasilające (np. +5V) na odpowiednie wyprowadzenia EPROM-u. Następnie na wyprowadzenia adresowe podawany jest adres komórek w matrycy pamięci, które będą programowane, a na magistralę danych jest przesłany bajt, który ma być wpisany do komórek. Następnie na wyprowadzeniu napięcia programującego powinno pojawić się napięcie o wartości np. 12,75 V. Impuls na odpowiednim wejściu sterującym wymusi otwarcie buforów magistrali danych EPROM-u i przyjęcie zapisywanej danej. Wreszcie finał, czyli wygenerowanie impulsu zapisującego, np. na wejściu CS EPROM-u, o czasie trwania 50 µs i polaryzacji ujemnej, przepisze bajt danych z magistrali EPROM-u do jego komórek pamiętających. Powtarzając te elementarne czynności odpowiednią liczbę razy, można zapisać całą wewnętrzną matrycę pamiętającą elementu. Oczywiście, w rzeczywistości wszystko jest trochę bardziej skomplikowane. Dane techniczne elementu określają parametry poszczególnych przebiegów na wyprowadzeniach elementu w czasie programowania i zależności czasowe między nimi. Należy także pamiętać o procedurze sprawdzenia, czy wszystkie dane zostały prawidłowo zapisane w matrycy elementu. Jednak ogólna zasada pozostaje taka jak opisana wyżej.

Jeżeli programator może wytworzyć odpowiednie przebiegi elektryczne i podać je na właściwe nóżki programowanego elementu, cała operacja zakończy się sukcesem. Zadaniem części sprzętowej programatora, o schemacie z rys. 1, jest właśnie generowanie odpowiednich impulsów i przesyłanie ich na odpowiednie wyprowadzenia programowanego elementu. Jednak do sterowania programatorem niezbędny jest także program, który nadzoruje działanie części sprzętowej programatora. Oprogramowanie sterujące zainstalowane w komputerze zajmuje się także magazynowaniem i przesyłaniem danych, które mają być zapisane w programowanym elemencie oraz komunikuje się z użytkownikiem.

Oprócz części sprzętowej i programu sterującego jest niezbędny jeszcze trzeci element tego systemu, czyli skrypt zapisany w języku FEMTO. W pliku skryptu zawarte są zarówno polecenia dla programu sterującego, np.: prześlij dane do programatora, zainicjuj programowanie, sprawdź poprawność programowania, jak i szczegółowe rozkazy dla części sprzętowej, np. włącz zasilanie, wygeneruj na odpowiednim wyprowadzeniu impuls, ustaw kolejny adres, a także informacje dla użytkownika w postaci komunikatów. Relacje pomiędzy tymi trzema częściami składającymi się na programator można przedstawić następująco:

skrypt
↓
program sterujący
↑↓
część sprzętowa

Współdziałanie tych trzech części systemu pozwala zaprogramować układ, a także przystosować programator do obsługi nowych typów układów.



Rys. 1. Schemat elektryczny programatora

Schemat przedstawiony na rys. 1 sprawia być może wrażenie skomplikowanego, lecz w istocie - za pomocą prostych układów, jakimi są rejestry zatrzaskowe czy wzmacniacze operacyjne - realizuje podstawowe zadania części sprzętowej: podawanie na programowany element właściwych kombinacji stanów logicznych, generację odpowiednich napięć (programującego i zasilania), a także komunikację z programem sterującym, który rezyduje w PC-cie.

Układy U4, U8, U11 sterowane przez mikrokontroler służą do podawania sygnałów na element programowany. Układy U1, U2 związane są z pamięcią RAM U3, w której m.in. przechowywane są dane zapisywane i odczytywane. Do generowania napięć: programującego i zasilania służą układy U7 i U9, także sterowane przez procesor. Komunikacja z komputerem sterującym jest możliwa dzięki interfejsowi RS232 zbudowanemu na U5.

Wszystkie sygnały i napięcia wyprowadzone są na złącza JP1 i JP2. Są to m.in.: 8 linii portu danych I/O0...I/O7, 16 linii adresowych A0...A15, 10 linii portów oznaczonych symbolami F1...F10. Zsumowanie liczby potrzebnych linii wskazuje, że zastosowany procesor AT89C52 nie jest w stanie ich obsłużyć własnymi portami i musi „podeprzeć się" dodatkowymi układami. Wszystkie są podłączone do wspólnej magistrali danych obsługiwanej przez port P0 procesora U17. Każdy z tych układów ma swoje wejście aktywujące połączone z wyprowadzeniami multipleksera U12. Dzięki temu procesor, wybierając po kolei każdy z układów, może do niego zapisać dane lub je odczytać, wykorzystując w tym celu jeden port P0 i cztery linie portu P1 sterujące multiplekserem. W ten sposób wystawianych jest 16 bitów adresu potrzebnych np. w czasie programowania EPROM-u. Tak samo realizowany jest dostęp do wewnętrznej pamięci RAM programatora (U3), w której przechowywane są dane.

Niektórych Czytelników może dziwić obecność w układzie drugiego procesora oznaczonego jako U11. Element ten odpowiada za realizację kilku zadań. Pełni m.in. rolę bramy wejścia-wyjścia magistrali danych, obsługuje port F10, na który można wysłać precyzyjnie odmierzone impulsy o czasie od 100 ns do 6,5 ms, przechowuje także w swojej wewnętrznej pamięci EEPROM dane konfiguracyjne programatora. Każdą z tych funkcji można powierzyć osobnemu układowi scalonemu, jednak jest korzystniej, gdy wykonuje je tylko jedna kostka zajmująca dużo mniej miejsca.

Zazwyczaj do zaprogramowania wielu typów elementów potrzebne jest napięcie programujące o wartości znacznie przekraczającej +5 V. Są także elementy wymagające w czasie programowania podwyższenia napięcia zasilania np. do +6,5 V. Z tego powodu wartość obu tych napięć może być zmieniana i ustawiana programowo.

Do realizacji tego zadania wykorzystano w programatorze podwójny potencjometr elektroniczny typu DS1267 oznaczony na schemacie symbolem U9. Procesor, wysyłając dane do potencjometru liniami portów P1.5...P1.7, może wymusić na wyprowadzeniu potencjometru pełniącego rolę suwaka napięcie z przedziału 0...5 V. To jednak nie wystarcza, ponieważ do programowania potrzebne są znacznie wyższe napięcia. W celu obejścia tego ograniczenia zastosowano wzmacniacze napięcia stałego. Spójrzmy na schemat: napięcie z wyjścia suwaka W0 U9...12 jest podawane poprzez opornik R15 na wejście odwracające wzmacniacza U7B. Po wzmocnieniu napięcie trafia do wtórnika emiterowego T6, natomiast poziom wzmocnienia układu wzmacniacz - tranzystor określa pętla sprzężenia zwrotnego R16, PR1 i rezystor R15. Z emitera tranzystora napięcie programujące jest podawane poprzez tranzystor odcinający T2 na złącze JP1. Tranzystor odcinający jest potrzebny, aby w czasie wkładania lub wyjmowania z podstawki programowanego elementu na jego wyprowadzeniach nie pojawiały się potencjały mogące doprowadzić do uszkodzenia. W podobny sposób wytwarzane jest napięcie zasilające. Tym razem elementem odcinającym nie jest tranzystor, a przekaźnik PK1, do którego drugiej pary styków dołączona jest dioda LED D2 sygnalizująca stan programowania.

Układ U5 jest zwykłym konwerterem poziomów sygnałów RS232 na poziom TTL. Program sterujący komunikuje się z częścią sprzętową, korzystając z portu COM komputera, którym są przesyłane rozkazy sterujące i dane. Po konwersji poziomów sygnały są przekazywane do wyprowadzeń RxD i TxD procesora.

Oprogramowanie procesora sterującego częścią sprzętową programatora

Układ stanowiący część sprzętową programatora nie jest jedynie biernym wykonawcą poleceń programu sterującego, ale posiada pewną autonomię. Przede wszystkim jest wyposażony w programowy interpreter języka FEMTO. Kody poleceń dotyczą elementarnych działań na wyprowadzeniach, np. zmiany poziomu z niskiego na wysoki na wyprowadzeniu F1, operacji logicznych, np. porównań stanów na wyprowadzeniach I/O0... I/O7 z ostatnio programowanym bajtem danych oraz dostępem do wewnętrznych rejestrów niezbędnych do pracy części programującej, do których ma także dostęp program sterujący w PC-cie. Takim rejestrem jest licznik adresu, którego 16 bitów wyprowadzonych jest na złącze JP2 i oznaczonych symbolami A0...A15. W rzeczywistości licznik ten składa się z 4 bajtów, a wartość bitów pozostałych bajtów można za pomocą poleceń przepisywać np. do wyprowadzeń F1...F10.

Kolejną grupą dostępnych z zewnątrz rejestrów są rejestry wskaźników dostępu do buforów pamięci RAM części sprzętowej. Wskaźnik określa po prostu adres, do którego można się odwołać.

Jakie to są rejestry (bufory)? Programator dzieli wewnętrzną pamięć RAM (U3) na trzy obszary. W pierwszym lokowane są kody rozkazów języka FEMTO interpretowane w czasie pracy programatora. Jak to zostało opisane w części dotyczącej opisu działania, kody określają elementarne czynności, jakie część sprzętowa musi wykonać np. w czasie zapisu danych do pamięci EPROM lub

w czasie weryfikacji itd. Drugi obszar pamięci jest przeznaczony na dane wejściowe. Jest to po prostu wydzielona część pamięci RAM, w której gromadzone są dane przesłane z komputera do części sprzętowej i przeznaczone do zapisu w programowanym elemencie. W trzecim obszarze zorganizowano bufor danych wyjściowych. Jest to obszar gromadzenia odczytanych danych z programowanego elementu przed przesłaniem ich do komputera PC.

Każdy z tych obszarów ma własne wskaźniki określające jego położenie i bieżący adres. Obszar kodu posiada jeden wskaźnik. Najpierw będzie z niego korzystał program sterujący, który prześle z PC-ta kody rozkazów ze skryptu. Następnie wskaźnik zostanie wyzerowany. Gdy rozpocznie się programowanie, kontrolę nad nim przejmie procesor części sprzętowej, realizując kolejne rozkazy pętli programowania. Bufory danych mają po dwie pary wskaźników. Jednym zarządza wyłącznie program sterujący, natomiast drugim procesor części sprzętowej. Dzięki temu możliwe jest przesyłanie danych, gdy toczą się jeszcze operacje programowania. Czytelnik tego opisu może zapytać: co się stanie, gdy wskaźniki dotrą do końca obszaru buforów? Wówczas przestawiane są od nowa na początek swojego bufora. Dzięki temu możliwe jest programowanie elementów o pojemności pamięci przekraczającej pojemność zastosowanej pamięci RAM, czyli 32 k. Po prostu, gdy zapisana zostanie do programowanego elementu część danych z bufora, na zwolnione miejsce wpisywane są nowe dane przesyłane z komputera.

Osobom, które zechcą skonstruować własny programator, opierając się na tym projekcie, należy opisać współpracę procesora głównego części sprzętowej z procesorem U11. Ze względu na brak odpowiedniej liczby wyprowadzeń U11, komunikacja z nim odbywa się z wykorzystaniem jedynie 5 linii magistrali danych D0...D4. Czterema młodszymi przesyłane są połówki bajtów danych, natomiast linia D4 (zależnie od sytuacji) służy do sygnalizacji gotowości do transmisji lub określa, która część bajtu jest aktualnie transmitowana. Tak jak w przypadku innych układów, procesor sygnalizuje chęć nawiązania kontaktu poprzez ustawienie poziomu niskiego na wyprowadzeniu multipleksera U12, dołączonego do linii portu PD5 układu U11.

Zastosowanie jako układu U11 szybkiego procesora AT90S2313 z rodziny AVR pozwoliło na programowe generowanie krótkich, ale precyzyjnie odmierzonych impulsów, które mogą pojawiać się na wyjściu F10. Wytworzenie takich impulsów przez procesor rodziny '51 jest niemożliwe, ponieważ jak wynika z zasady jego pracy, impulsy taktujące wytwarzane są na podstawie sygnału generatora kwarcowego przez podział jego częstotliwości przez 12. Nawet wykonując następujące bezpośrednio po sobie rozkazy, ustawienia któregoś z portów naprzemian: na poziomie wysokim i następnie niskim, jesteśmy w stanie wygenerować impuls o czasie trwania nie krótszym niż 1,09 µs, co wynika z częstotliwości własnej zastosowanego kwarcu X1. Takie impulsy lub ich wielokrotność można uzyskać na wyprowadzeniach F1...F9. Najczęściej są one wystarczające.

Tam jednak, gdzie potrzebne są krótkie impulsy o małym błędzie czasu trwania, należy użyć wyprowadzenia F10 obsługiwanego przez AT90S2313. Zaletą procesorów AVR jest ich szybkość działania wynikająca z tego, że cykl rozkazowy w większości przypadków jest równy jednemu okresowi sygnału generatora kwarcowego. Procesory te są co najmniej 12 razy szybsze od ich odpowiedników z rodziny '51. Dodatkowo, dzięki wykorzystaniu skoku pośredniego, adresowanego rejestrem Z, można wytworzyć dokładnie impuls o czasie trwania 100 ns lub jego wielokrotności.

Realizacja programowa generatora impulsów dodatnich o krótkim czasie trwania jest następująca: w pamięci programu procesora AVR wpisana jest tablica zawierająca np. 256 razy powtórzony rozkaz ustawienia portu PD6 (w programatorze obsługuje wyjście F10) na poziom wysoki. Bezpośrednio za tablicą powinien być rozkaz ustawiający PD6 na poziom niski. Ponieważ oba roz-

WYKAZ ELEMENTÓW

Rezystory
R1, R3, R4, R12: 2,2kΩ
R2, R5...R10: 10kΩ
R11: 1kΩ
R13, R14: 20kΩ
R15...R18: 100kΩ
RPACK1: drabinka rezystorów 10kΩ
PR1: potencjometr wieloobrotowy 470kΩ
PR2: potencjometr wieloobrotowy 220kΩ

Kondensatory
C1...C4: 47µF/16V
C5, C7, C8, C10, C11, C15, C16, C19, C20: 100nF
C6, C9: 470µF/25V
C12, C13, C17, C18: 27pF
C14: 470µF/16V
C21, C22: 100pF

Półprzewodniki
D1, D2: diody LED: czerwona i zielona ϕ3 lub 5mm z oprawkami
D3: mostek prostowniczy 1A/50V
T1: BC557
T2: BC327
T3, T4: BC547
T5, T6: BD135
U1, U2, U4, U8: 74LS574SMD
U3: pamięć RAM 62256 SMD
U5: MAX232
U6: 7805
U7: TL082
U9: DS1267
U10: MCP101 (lub podobny)
U11: AT90S2313 SMD zaprogramowany
U12: 74LS42SMD
U17: AT89C52 zaprogramowany PLCC

Różne
JP1, JP2: szpilki do złącz zaciskanych na taśmie HEADER20
JP3: CON3 gniazdo zasilania wlutowywane do płytki
P1: DB9 gniazdo kątowe żeńskie wlutowywane do płytki
PK1: przekaźnik 5V miniaturowy
X2: 10MHz
X1: 11,059MHz
Dwustronna płytka drukowana programatora
Jednostronna płytka drukowana „connect board" złącza dla wymiennych adapterów
Obudowa typu Z50
Podstawka PLCC44
Podstawka DIP16
Styki precyzyjne ϕ0,8mm
Taśma 20-żyłowa 20cm
Wkręty stożkowe M3
Złącza zaciskane na taśmę 20-żyłową

Uwaga! *Wszystkie oporniki i kondensatory nie-elektrolityczne typu SMD 1206*

kazy potrzebują do realizacji tylko jednego okresu oscylatora, organizując skok do tablicy bliżej lub dalej jej końca, można wytworzyć impuls będący wielokrotnością 100 ns. Adres skoku należy wyliczyć przed jego realizacją i zapisać w rejestrze Z. Sam skok wykonywany jest poleceniem IJMP (*indirect jump*). W ten sposób stosując kwarc 10 MHz można wygenerować impulsy o czasie trwania dokładnie 0,1 µs...25,6 µs. Oczywiście, sposób ten zajmuje sporo miejsca w pamięci programu (256 razy powtórzony ten sam rozkaz), ale jest prosty i skuteczny. Do generacji dłuższych impulsów wykorzystywane są programowe pętle opóźniające.

Ryszard Szymaniak, AVT
ryszard.szymaniak@ep.com.pl

Opis języka i ewentualnych zmian jest dostępny na stronie http://www.aries-rs.com.pl/femto.

Wzory płytek drukowanych w formacie PDF są dostępne w Internecie pod adresem: ***http://www.ep.com.pl/?pdf/grudzien02.htm*** *oraz na płycie CD-EP12/2002B w katalogu* ***PCB****.*