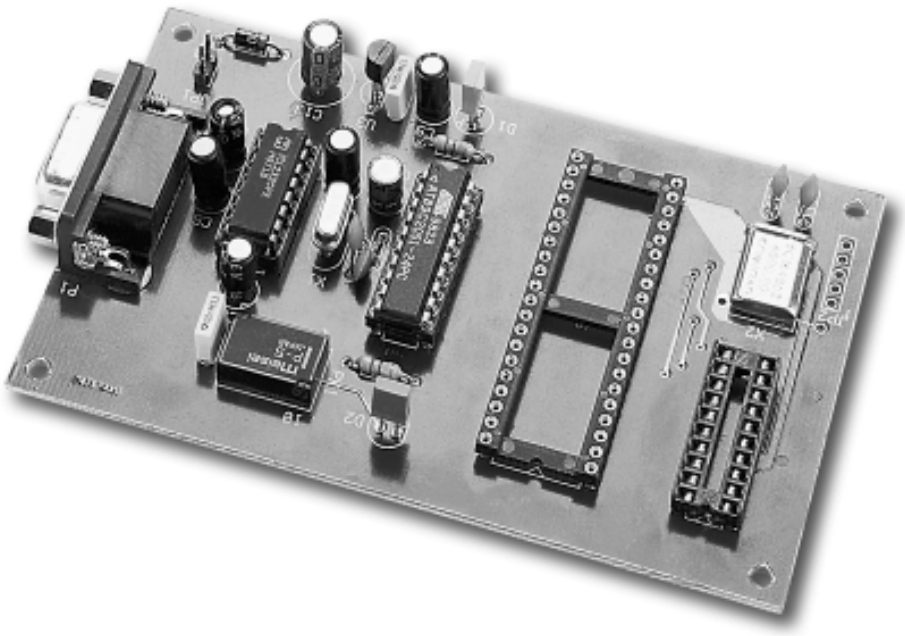
Programator procesorów AVR, część 1

kit AVT-812



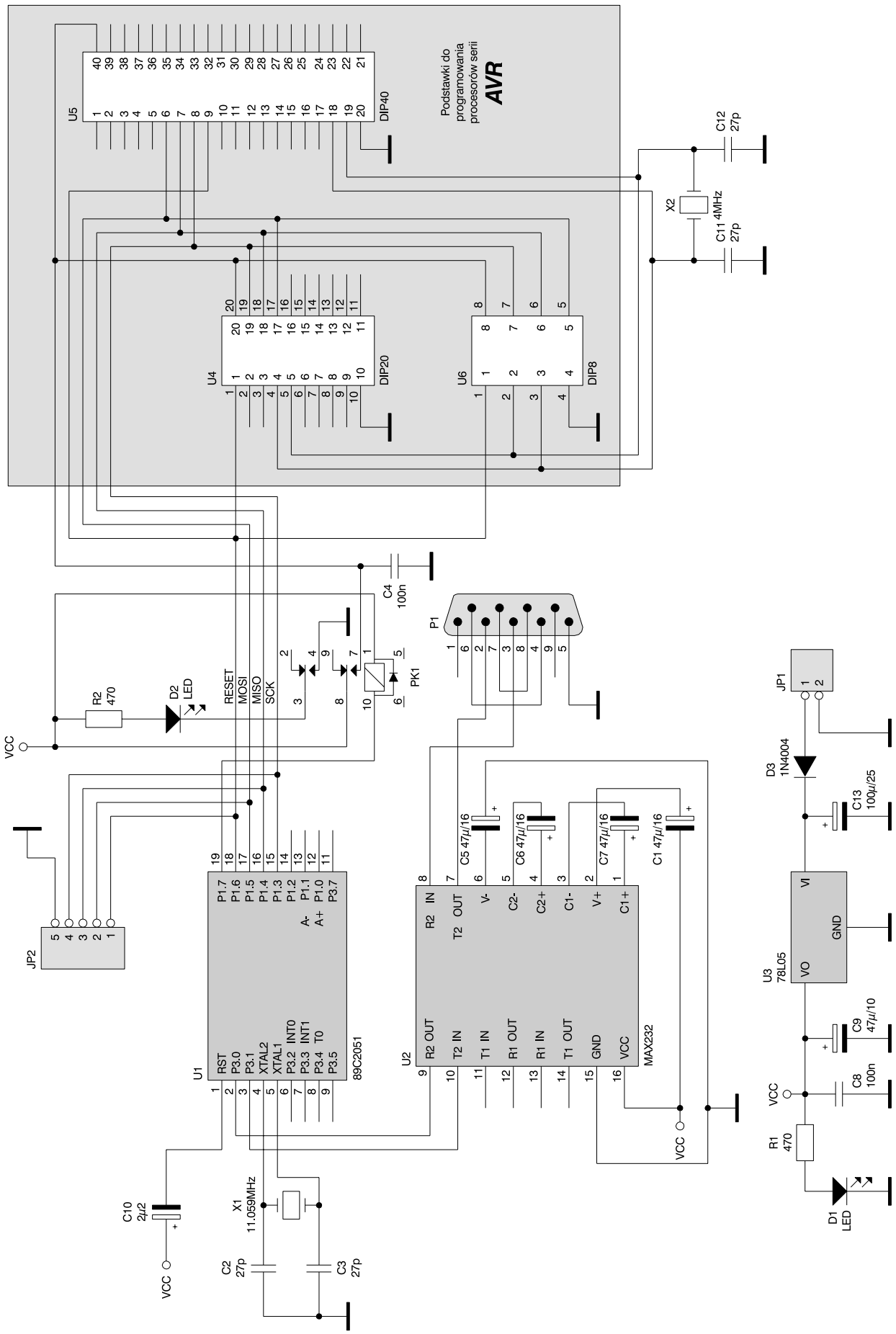
Procesory jednoukładowe zrobiły prawdziwą karierę w świecie elektroniki. Sukces ten wiąże się z rozwojem elektronicznego sprzętu powszechnego użytku. Im urządzenia stawały się łatwiejsze w użyciu, bardziej sprawne i niezawodne, tym bardziej rosło zapotrzebowanie na elementy sterujące ich pracą, czyli mikrokontrolery. Dotyczy to także układów automatyki przemysłowej. Dziś trudno spotkać urządzenie elektroniczne bez inteligentnego sterownika, będącego dalekim krewnym dużych komputerów.

Rozwój elekreonicznego sprzętu powszechnego użytku i możliwość ulokowania w nim swoich wyrobów stanowiła silny bodziec dla wielu firm zajmujących się produkcją układów wielkiej skali integracji.

Początkowo na rynku dominowały układy z grupy 8049, następnie słynny 8051 produkowany najpierw przez Intela oraz mikrosterowniki firmy Motorola. W latach 90. pojawiły się nowe rodzaje procesorów, nierzadko bardzo wyspecjalizowanych i zminiaturyzowanych.

Firma Atmel zaistniała na rynku najpierw ze swoją odmianą sterownika '51, w którym zastąpiono niewygodną, kasowaną ultrafioletem pamięć EPROM, elektrycznie programowaną pamięcią FLASH EEPROM. Taki sposób zapisu kodu programu do pamięci sterownika jest bardzo wygodny, zwłaszcza na etapie pracy nad programem i w produkcji małoseryjnej. W przypadku polskiego rynku można stwierdzić, że procesory te znalazły na nim swoje miejsce. Od ponad dwóch lat Atmel promuje nową rodzinę procesorów ochrzczonych wspólnym mianem AVR.

Procesory wchodzące w skład rodziny różnią się wielkością i możliwościami, jednak kilka cech pozostaje wspólnych. Najważniejszą z nich jest oparcie wewnętrznej budowy sterowników na architekturze RISC, bazującej na uproszczonej liście rozkazów wykonywanych najczęściej podczas jednego cyklu zegarowego. Powoduje to znaczne przyśpieszenie pracy układu, w którym jeden cykl zegara taktującego odpowiada jednemu cyklowi rozkazowemu. Określenie „uproszczona lista rozkazów“ wcale nie oznacza, że jest ona krótka, ponieważ wzbogacono ją o cały zestaw skoków warunkowych i trybów adresowania. W związku z tym wszystkie procesory wyposażone są w rozbudowany zestaw rejestrów uniwersalnych bezpośrednio współpracujących z akumulatorem, co dodatko-



Rys. 1. Schemat elektryczny urządzenia.

wo zwiększa szybkość działania układów.

Producent określa moc obliczeniową sterowników na równą 1MIPS przy częstotliwości zegara 1MHz. Maksymalna częstotliwość zegara dla większości typów procesorów AVR zawiera się w przedziale 10..24MHz. Kolejną cechą wspólną jest wyposażenie prawie wszystkich typów sterowników w wewnętrzne pamięci RAM i EEPROM oraz sprzętowy zegar watchdoga. Wszystkie procesory posiadają szeregowy interfejs SPI umożliwiający w prosty sposób ich programowanie oraz zapisywanie i odczytywanie danych do i z wewnętrznej pamięci EEPROM. Linie portów wyjściowych procesorów pozwalają na bezpośrednie sterowanie różnych układów zewnętrznych, np. diod LED, ponieważ w stanie niskim potrafią przyjąć prąd o wartości nawet 20mA. Konstruktorzy dużo uwagi poświęcili redukcji mocy pobieranej przez sterowniki, co umożliwia ich stosowanie w sprzęcie zasilanym bateryjnie.

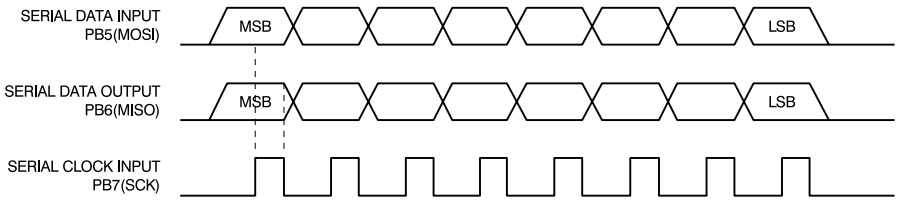
Pobór prądu w czasie normalnej pracy wynosi przeciętnie kilka miliamperów, natomiast w czasie uśpienia, gdy podtrzymywana jest zawartość wewnętrznych rejestrów i aktywny jest tylko wewnętrzny zegar watchdoga, pobór prądu wynosi jedynie 50µA.

Kolejną interesującą cechą wszystkich procesorów jest możliwość zakończenia trybu uśpienia poprzez podanie odpowiedniego poziomu napięcia na linii portu P3 i wygenerowanie przerwania.

Mamy nadzieję, że ten krótki wstęp zachęci Was do bliższego poznania procesorów AVR. Aby ułatwić Wam nieco to zadanie, w drugiej części artykułu omówimy nieco bardziej szczegółowo możliwości poszczególnych układów tej rodziny, a teraz przejdziemy do prezentacji konstrukcji programatora, opisywanego w artykule.

Opis układu

Schemat elektryczny programatora pokazano na **rys. 1**. Jego budowa jest bardzo prosta, ale można za jego pomocą zaprogramować praktycznie każdy rodzaj procesora AVR. Wynika to z faktu zastosowania w nim szeregowego interfejsu SPI.



Rys. 2. Przebieg sygnałów w czasie transmisji.

Wszystkie sterowniki z rodziny AVR mogą być programowane na dwa sposoby. Pierwszy z nich, który można określić jako tradycyjny, wykorzystuje do wymiany danych pomiędzy programatorem, a programowanym procesorem jeden z jego portów, a kilka dodatkowych sygnałów podawanych na linie pozostałych portów steruje całym procesem. Sposób ten wymaga zaangażowania wielu linii danych. Jeżeli programator ma obsłużyć procesory w różnych obudowach i o różnej liczbie wyprowadzeń, trzeba się liczyć z koniecznością stosowania adapterów lub multipleksowaniem wyprowadzeń programatora, w zależności od typu aktualnie programowanego procesora.

Drugi sposób wiąże się z wykorzystaniem w każdym typie procesora specjalnego szeregowego interfejsu o zredukowanej liczbie wyprowadzeń, w tym przypadku trzech. Zastosowanie do programowania jedynie trzech „drutów" - SPI (tak naprawdę dochodzi jeszcze zasilanie, linia RESET oraz doprowadzenia zegara) bardzo upraszcza procedurę programowania, a w pewnych warunkach umożliwia przeprogramowanie procesora nawet wtedy, gdy znajduje się w systemie, w którym pracuje.

Interfejs SPI składa się z następujących linii sygnałowych: linii zegara synchronizującego transfer informacji SCK, linii danych wejściowych MOSI i linii danych wyjściowych MISO. Przebieg sygnałów na tych trzech liniach w czasie transmisji pokazano na **rys. 2**.

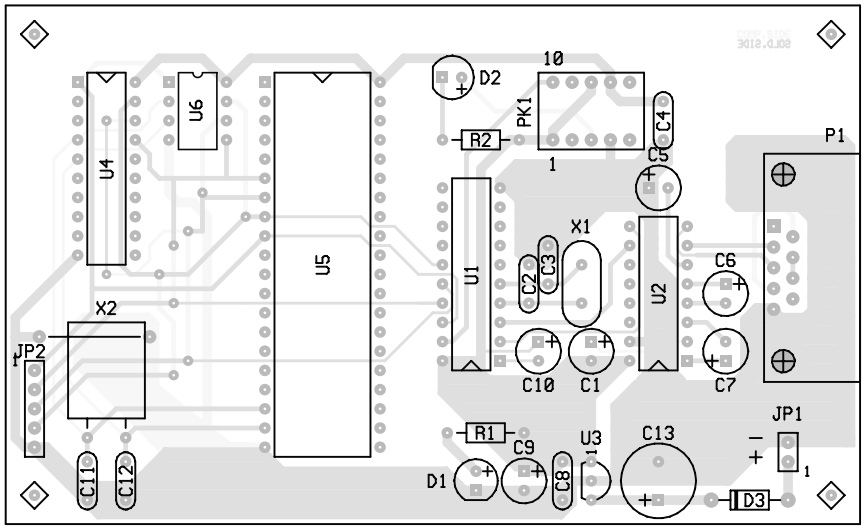
Programowanie polega na wysłaniu linią MOSI kodów sterujących i ewentualnie danych, które określą sposób w jaki ma się zachować programowany układ. Kody są to 3 lub 4 bajty wysyłane bit po bicie, z najstarszym bitem jako pierwszym. Odczytane dane procesor wysyła w ten sam sposób linią MISO, z najstarszym bitem jako pierwszym.

Kody sterujące wpisywane są do procesora podczas narastającego zbocza zegara SCK, natomiast dane pojawiające się na linii MISO mogą być odczytane podczas opadającego zbocza impulsu zegarowego. Sygnały na liniach MOSI i MISO mogą się zmieniać jedynie podczas stanu niskiego na linii zegarowej SCK.

Procesory AVR wyposażone w interfejs SPI reagują na kilka kodów sterujących. Ich format w przypadku procesora 90S2313 pokazano w **tab. 1**.

W celu rozpoczęcia korzystania z interfejsu SPI należy spełnić kilka prostych warunków. Przed podaniem napięcia zasilającego trzeba podać na wyprowadzenia procesora RESET i SCK stan niski oraz dołączyć do wyprowadzeń XTAL rezonator kwarcowy o nominalnej częstotliwości lub podać sygnał taktujący na wyprowadzenie XTAL1. Po włączeniu zasilania należy odczekać 20ms, a następnie wysłać cztery bajty rozkazu *Programing enable*. Od tego momentu procesor znajduje się w trybie programowania. Zapisanie do pamięci procesora nowego kodu programu wymaga wcześniejszego wykasowania zawartości pamięci FLASH, co nastąpi po wysłaniu rozkazu *Chip erase*, a następnie odczekaniu 10ms na zakończenie operacji kasowania pamięci.

Jednocześnie z pamięcią FLASH wykasowane zostaną wszystkie dane zapisane w pamięci EEPROM procesora. Zapis danych do pamięci programu wykonuje się za pomocą rozkazu Write Program Memory. Litery *a, b* oznaczają wyrażony binarnie adres komórki pamięci, do której zostanie dokonany zapis. Liczba znaczących bitów w przypadku procesorów o większej pojemności



Rys. 3. Rozmieszczenie elementów na płytce drukowanej.

pamięci FLASH będzie oczywiście większa niż w przykładzie odnoszącym się do procesora 90S2313. Ponieważ format rozkazów procesorów AVR jest 16-bitowy, a przesyłane bajty danych są 8-bitowe (4 bajt rozkazu oznaczony literami „i"), identyfikator „H" określa, która połówka kodu jest aktualnie transmitowana. Starsza i młodsza połówka kodu rozkazu zapisywane są pod jednakowym adresem *a, b*.

W czasie odczytu pamięci procesora adresowanie z wykorzystywaniem bitu „H" jest identyczne jak podczas zapisu. Różnica polega na tym, że po wysłaniu 3 bajtów linią MOSI, odczytywany bajt danych pojawi się na linii MISO.

Do zapisu i odczytu danych do i z pamięci EEPROM procesora służą rozkazy *Write EEPROM Memory* i *Read EEPROM Memory*, a cała wymiana danych przebiega podobnie jak w przypadku pamięci programu. Różnica polega na tym, że przed nowym zapisem nie ma konieczności czyszczenia pamięci rozkazem *Chip erase*.

Zaadresowana komórka EEPROM jest automatycznie czyszczona przed zapisem nowych danych. Następnie trzeba odczekać 4ms przed wysłaniem kolejnego kodu interfejsem SPI.

Rozkaz *Write Lock Bits* pozwala zaprogramować bity zabezpieczeń chroniące obie pamięci przed możliwością doprogramowania nowych danych, a także przed ich odczytaniem. Bity są aktywne, gdy przyjmują wartość 0. Bity mogą być skasowane jedynie w wyniku działania rozkazu *Chip erase*. Rozkaz *Read Device Code* odczytuje kod typu procesora. Zakończenie sesji programowania wymaga wyłączenia zasilania oraz pozostawienia wyprowadzenia RESET na poziomie wysokim, gdy zasilanie zostanie włączone ponownie.

Dzięki temu, że programator wykorzystuje w swoim działaniu interfejs SPI, jego budowa może być stosunkowo prosta. Zasadniczym elementem urządzenia jest procesor 89C2051, który kontroluje przepływ danych liniami interfejsu oraz włącza i wyłącza przekaźnik PK1 dołączający napięcie zasilania do programowanego procesora. Pracą programatora steruje komputer poprzez standardową linię RS dołączaną do gniazda P1. Parametry transmisji portem RS to: szybkość 9600 bodów, 8 bitów danych, brak bitu parzystości oraz 1 bit stopu. Układ scalony U2 dokonuje konwersji poziomów logicznych sygnałów do standardu TTL.

W założeniu programator miał być jak najprostszym urządzeniem współpracującym z zewnętrznym komputerem klasy PC. Z tego powodu oprogramowanie procesora U1 potrafi jedynie obsługiwać interfejs SPI oraz rozróżnia 3 rozkazy sterujące, co zupełnie wystarczy, aby prawidłowo zapisać i odczytać dane z wszystkich procesorów AVR.

Każdy z rozkazów składa się z kilku bajtów danych. Najpierw wysyłany jest znak litery (w formacie ASCII) określający rodzaj rozkazu, następnie jego parametry, dane, a na końcu bajt sumy kontrolnej. Bajt ten powstaje poprzez wykonanie operacji XOR na wszystkich kolejnych bajtach rozkazu z wyłączeniem oczywiście samego bajtu sumy kontrolnej. Po prawidłowym wykonaniu każdej operacji programator potwierdza ten fakt wysyłając w odpowiedzi literę „A". Wysłanie jakiegokolwiek innego znaku lub brak odpowiedzi powinien być interpretowany przez komputer sterujący jako błąd.

Rozkazy sterujące i opis poszczególnych bajtów:

1.Rozkaz ustawienia parametrów programatora:

„S"*abk*0*c*

„S" - kod ASCII (53h) identyfikatora rozkazu

ab - dwa bajty określające adres początkowy, od którego programator rozpocznie odczytywanie lub zapisywanie danych do procesora

k - parametr określający sposób pracy programatora. Bajt ten może przyjmować następujące wartości:

0 - następne rozkazy odczytu lub zapisu będą dotyczyły pamięci programu procesora

1 - następne rozkazy będą dotyczyły pamięci EEPROM procesora

2 - programator powinien uaktywnić bity zabezpieczające programowanego procesora

FFh - programator powinien się zresetować

0 - bajt o stałej wartości równy zero

c - bajt sumy kontrolnej

2.Rozkaz odczytu danych:

„R"*xc*

„R" - kod ASCII (52h) identyfikatora rozkazu

x - liczba bajtów danych, które mają być odczytane z pamięci procesora

c - bajt sumy kontrolnej

3.Rozkaz zapisu danych do pamięci procesora:

„W"*xd...dc*

„W" - kod ASCII (57h) identyfikatora rozkazu

x - liczba bajtów danych *d*, które mają być zapisane do pamięci procesora (FLASH lub EEPROM, co zależy od paramet-

ru *k* wysłanego we wcześniejszym rozkazie ustawienia parametrów)

d - bajty danych, których liczba została określona parametrem *x*. Liczba danych może być dowolna lecz nie większa niż 32

c - bajt sumy kontrolnej

Programator po odebraniu rozkazu ustawienia parametrów, przede wszystkim ustawia swój wewnętrzny licznik danych zgodnie z wartością przekazaną parametrami *ab*. Licznik ten po każdym zapisie lub odczycie pamięci procesora AVR jest zwiększany o jeden. Z tego powodu ustawienie początkowego adresu, np. podczas odczytu całej dostępnej pamięci FLASH procesora, można przeprowadzić tylko raz, wysyłając na początku rozkaz S*abk*0*c*. Programator zapamiętuje także parametr *k* i do czasu jego zmiany wszelkie odczyty lub zapisy będą dotyczyć wybranego typu pamięci.

Potwierdzenie wykonania rozkazu przez programator w przypadku rozkazu odczytu jest nieco zmodyfikowane. Po wysłaniu kodu litery „A" (41h) programator wysyła także odczytane dane w liczbie określonej w rozkazie odczytu parametrem *x*, dołączając na końcu bajt sumy kontrolnej *c*. Po wykonaniu każdego rozkazu odczytu danych lub zapisu, programator przez ok. 0,6s podtrzymuje w stanie załączenia przekaźnik PK1. Jeżeli w tym czasie nie zostanie odebrany kolejny rozkaz, styki przekaźnika zostaną rozłączone i programowany procesor AVR bez obaw można wyjąć z podstawki. Wysłanie polecenia resetu programatora powoduje natychmiastowe rozłączenie styków i przejście programatora w stan oczekiwania na kolejny rozkaz.

Przykładowa sekwencja rozkazów, dotycząca zapisania do pamięci FLASH nowego programu dla sterownika AVR, a potem sprawdzenia czy zapis został dokonany poprawnie, może wyglądać następująco:

S, 0, 0, 0, 0, *c* (ustawienie początkowego adresu zapisu na 00h i typu pamięci na FLASH)

W,20h,*d,...,d,c* (zapis bloku 32 bajtów)... (zapis kolejnych bloków danych)

W,32,*d,...,d,c*

S,0,0,FFh,*c* (reset programatora)

S,0,0,0,0,*c*

R,20h,*c* (odczyt danych z pamięci programowanego procesora w celu weryfikacji)

R,20h,*c*

S,0,0,FFh,*c* (reset programatora)

Podany opis powinien okazać się wystarczający do napisania własnego programu sterującego pracą programatora. Wszystkim, którzy nie mają ochoty tworzyć go samodzielnie proponujemy jego funkcjonalną, prostą wersję pracującą w środowisku Windows95 (wchodzi w skład kitu).

Montaż i uruchomienie

Konstrukcja mechaniczna programatora jest bardzo prosta. Na płytce drukowanej (rozmieszczenie elementów na **rys. 3**) oprócz innych części znajdują się także podstawki do osadzania procesorów AVR na czas programowania. Zastosowanie 3 typów podstawek pozwala programować niemal wszystkie typy układów. Dodatkowo sygnały interfejsu SPI wyprowadzone są na gniazdo JP2. Procesor przed rozpoczęciem programowania lub czytania należy umieścić w podstawce z odpowiadającą mu liczbą styków (na fotografii pokazana jest starsza wersja programatora, jedynie z dwoma typami podstawek). Wcześniej płytkę programatora należy połączyć z odpowiednim gniazdem portu RS komputera standardowym kablem oraz zasilić napięciem stałym o wartości 8..12V dołączanym do gniazda JP1.

Po uruchomieniu na komputerze programu sterującego i rozpoczęciu czytania lub zapisu procesora AVR, przekaźnik na płytce zostanie załączony, co sygnalizuje zapalenie się diody D2. Po zakończeniu programowania dioda zgaśnie i procesor może być wyjęty z podstawki. Jednocześnie może być programowany tylko jeden procesor. W programatorze najlepiej użyć podstawek ze złączami precyzyjnymi, które nie ulegną zniszczeniu na skutek częstego wkładania i wyjmowania programowanych procesorów.

Ryszard Szymaniak, AVT

Tab. 1.

Kod sterujący	Bajt1	Bajt2	Bajt3	Bajt4
Programing enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx
Chip erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx
Read Program Memory	0010 H000	xxxx xxaa	bbbb bbbb	oooo oooo
Write Program Memory	0100 H000	xxxx xxaa	bbbb bbbb	iiii iiii
Read EEPROM Memory	1010 0000	xxxx xxxx	xbbb bbbb	oooo oooo
Write EEPROM Memory	1100 0000	xxxx xxxx	xbbb bbbb	iiii iiii
Write Lock Bits	1010 1100	111x x21x	xxxx xxxx	xxxx xxxx
Read Device Code	0011 0000	xxxx xxxx	xxxx xxbb	oooo oooo

gdzie:
a, b binarnie określony adres pamięci z 'a' oznaczającymi starsze bity
H bit określający czy chodzi o młodszy [0] czy starszy [1] bajt kodu programu (dane zapisywane są do pamięci programu jako bajty, natomiast w ALU przetwarzane jako 16-bitowe słowa),
o dana odczytywana z wyjścia MISO
i dana zapisywana do pamięci
1, 2 bity zabezpieczenia przed odczytem
x ustawienie tak oznaczonego bitu nie jest istotne

WYKAZ ELEMENTÓW

Rezystory
R1, R2: 470Ω
Kondensatory
C1, C5, C6, C7, C10: 47µF/16V
C2, C3, C11, C12: 27pF
C8, C4: 100nF
C10: 2,2µF
C13: 100µF/25V
Półprzewodniki
D1, D2: LED czerwona i zielona
D3: 1N4004
U1: 89C2051 zaprogramowany
U2: MAX232
U3: 78L05
Różne
PK1: przekaźnik miniaturowy 5V typu OMRON
P1: złącze DB9 żeńskie do druku
X1: 11,059MHz
X2: 4MHz
U4: precyzyjna podstawka DIP20
U5: precyzyjna podstawka DIP40
U6: precyzyjna podstawka DIP8