Mikrokontrolery? To takie proste...

Część 15
Procesor 8051/2 – obsługa i praca z wewnętrzną pamięcią programu

W dzisiejszym odcinku traktującym o mikrokontrolerach 8051/52 zapoznamy się z możliwościami i sposobami obsługi i programowania tych wersji kostek które posiadają wewnętrzną pamięć programu typu EPROM. Ze względu na to, że w ostatnich latach, cena układów w takich wersjach bardzo spadła, nawet kilkanaście razy, oraz pojawiło się wiele mutacji procesorów C51/52 z pamięcią reprogramowalną typu „Flash" EEPROM, stały się one dostępne dla większości hobbystów – amatorów techniki mikroprocesorowej. Jeżeli nawet nie zamierzasz wkrótce korzystać z dobrodziejstw procesora w wersji z wewnętrzną, reprogramowalną pamięcią programu, to przedstawiona w artykule garść informacji z pewnością, przyczyni się do większego oswojenia się z popularną '51-ką.

Pamiętam jeszcze czasy, a było to prawie dekadę temu, kiedy to cena procesora 87C51 była tak duża, że aby go zdobyć, musiałem sporo odkładać ze studenckiego stypendium. Kiedy wreszcie udało mi się kupić wymarzoną kostkę, środków ostrożności nie było nigdy za wiele. A to nosiło się układ w folii aluminiowej, a przed wyjęciem wyrównywało się swój – ludzki potencjał dotykając kaloryfera lub rury wodociągowej, a to sprawdzało się zmontowaną płytkę drukowaną przed włożeniem drogocennego układu kilka razy. Wszystko po to aby przypadkiem nie uszkodzić delikatnej struktury mikroprocesora. Dla przykładu podam, że wówczas przy cenie procesora 87C51 równej około 500 tys. złotych (50,- nowych złotych), wersja bez pamięci EPROM lub wersje z pamięcią stałą ROM, kosztowały około 20...30 tys. (2...3 nowe złote), czyli około 20 razy mniej. Nie wspominam tutaj o procesora takich jak np. 87C52 z wewnętrzną pamięcią EEPROM, których zdobycie było nie lada trudnością, a jeżeli już były to trzeba było za nie zapłacić ponad milion starych złotych, czyli prawie 1/3 ówczesnej przeciętnej pensji!

Tabela 1

Symbol handlowy	Opis
80C51	wersja z wewnętrzną pamięcią programu typu ROM, której zawartość jest nieznana z naszego punktu widzenia, toteż układ możemy wykorzystać do pracy tylko z dołączoną zewnętrzną pamięcią np. EPROM do której zapiszemy nasz program (wtedy pamięć ROM jest wyłączona – nieaktywna).
80C31	wersja procesora bez wewnętrznej pamięci programu. Mikrokontroler w tej wersji może pracować tylko z dołączoną zewnętrzną pamięcią jak dla 80C51.
87C51	wersja z wbudowaną pamięcią EPROM – 4kB. Obudowa mikroprocesora posiada okienko kwarcowe, dzięki któremu możliwe jest kasowanie zawartości tej pamięci, co umożliwia wielokrotne programowanie całego układu.
89C51	wersja procesora z kasowaną elektrycznie pamięcią EEPROM – 4kB. Ponieważ w tej wersji cała pamięć programu EEPROM może być kasowana bardzo szybko – za pomocą tylko 1 impulsu, procesory w tej wersji nazywa się typu „Flash".
80C52	jest to procesor identyczny z 8051 tyle, że posiada dodatkowy 3-ci programowalny licznik/timer (nazywany jako „T2"). Reszta jek dla 80C51 – patrz wyżej.
80C32	jak dla 80C31 z uwzględnieniem „T2".
87C52	jak dla 87C51 z uwzględnieniem „T2", wewn. pamięć programu ma 8kB.
89C52	jak dla 89C51 z uwzględnieniem „T2", wewn. pamięć programu ma 8kB.

Teraz, kiedy na rynku aż roi się od mikroprocesorów w różnych wersjach, ceny układów znacznie spadły. Obecnie za kwotę 15 zł można kupić popularną '51-kę z pamięcią EEPROM „Flash", a kostki tzw. „ROM less", czyli bez wewnętrznej pamięci programu lub z fabrycznym ROMem, spotka się za kilka złotych. I właśnie ze względu na to, że różnica pomiędzy cenami układów jest taka mała, warto zainteresować się takim właśnie wersjami tych jakże popularnych kostek.

W jednym z pierwszych odcinków szkoły mikroprocesorowej, opisywałem kilka najpopularniejszych obecnie wersji procesorów oraz ich przybliżone ceny. Ponieważ było to ponad rok temu, obecne ceny tych układów są jeszcze niższe. Dla przypomnienia zamieszczam informacje dotyczące najpopularniejszych obecnie mikrokontrolerów serii MCS-51 (patrz **tabela 1**).

Zanim przejdę do omówienia sposobów programowania i weryfikacji pamięci wewnętrznej programu mikrokontrolerów '51, powinienem Ci uzmysłowić drogi Czytelniku, że do wykonania tej operacji będzie potrzebny, oprócz dobrej woli, także „programator", w dodatku nie byle jaki, bo potrafiący programować procesory rodziny MCS-51.

Na rodzimym rynku można znaleźć sporo urządzeń tego typu, kosztujących od kilkuset złotych do kilkudziesięciu! Nie oznacza to że aby zaprogramować procesor trzeba wybrać urządzenie najdroższe, chodzi o to aby znaleźć te tańsze, potrafiące jednak bez żadnych przystawek, fachowo nazywanych adapterami, programować chociaż podstawowe procesowy z rodziny MCS-51, a więc: 87C51, 87C52, 98C51, 98C52. Na szczęście większość amatorskich programatorów oferowanych przez drobnych rodzimych wytwórców, spełnia te wymagania, a ich cena nie zwala z nóg przeciętnego „zjadacza chleba" który interesuje się techniką mikroprocesorową i chce dokształcić się w tej jakże interesującej dziedzinie wiedzy.

Niestety chęć ujarzmienia kostek z wewnętrzna pamięcią programu, oprócz posiadania programatora, wymaga także posiadania komputera PC. Tak więc „ręczniakom" należą się w tym miejscu przeprosiny, lecz musicie zdawać sobie sprawę drodzy koledzy, że aby posiąść stosowną wiedzę, każdy z nas jest zdolny do wielu wyrzeczeń. Tak też było w moim przypadku, aczkolwiek jeszcze kilka ładnych lat temu, kiedy królowały komputery PC typu XT oraz AT, kupno jednego z nich było nie lada wysiłkiem dla całej mojej rodziny. W chwili obecnej komputer PC wystarczający do obsługi programatora procesorów nawet w najmniejszej konfiguracji powinien mieć procesor co najmniej 80286, 1MB pamięci RAM, oraz jakikolwiek twardy dysk z wolnymi ok. 2,5MB przestrzeni. Takiej konfiguracji praktycznie nie spotka my już na rynku, z pomocą przyjść musza więc giełdy, gdzie proponowany zestaw w nieco lepszej (z procesorem 80386) konfiguracji można nabyć na 200..300 zł. Do tego należy dokupić jeszcze używany monitor mono za około 50 zł i można zabrać się do programowania. Ważne jest aby przy takiej konfiguracji komputera program obsługi wybranego przez Ciebie programatora potrafił pracować w trybie tekstowym, bowiem zainstalowanie systemu Windows tylko dla celów programowania '51-ek na tym etapie wiedzy nie ma za bardzo sensu.

Na szczęście większość oprogramowania na dostępne na naszym rynku programatory pracuje w środowisku tekstowym MS-DOS, i posiada bardzo ograniczone wymagania sprzętowe co do komputera, toteż z instalacją nabytego urządzenia nie powinno być większych problemów.

Na pocieszenie pragnę poinformować, że w ofercie handlowej AVT znajduje się idealny do naszych potrzeb programator przeznaczony specjalnie dla rodziny procesorów MCS-51. Urządzenie posiada kod handlowy AVT-320 i sprzedawane jest w postaci zestawu do samodzielnego montażu (wersja /B) lub jako zmontowane (wersja /C). Bardziej zaawansowani i wytrwali elektronicy mogą też nabyć samą płytkę drukowaną wraz z kilkoma układami scalonymi opracowanymi specjalnie dla potrzeb tego urządzenia (zestaw /A). Dołączona do zestawu dyskietka zawiera prosty ale funkcjonalny program obsługi programatora. Urządzenie współpracuje z komputerem typu PC (począwszy od poczciwego XT na szybkich Pentiumach skończywszy) poprzez port szeregowy RS232C.

Wszystkich zainteresowanych zachęcam do lektury artykułu na ten temat, który ukazał się z naszym bratnim piśmie – „Elektronice Praktycznej" w numerach 9,10 i 11/97, a jest autorstwa niżej podpisanego.

Obsługa pamięci programu

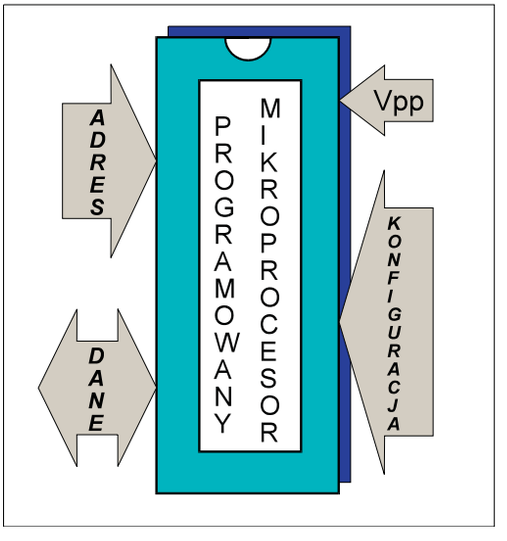
W tym miejscu powinienem wyjaśnić że w tytule tej części artykułu chodziło mi o wspominaną wcześniej sprawę programowania pamięci wewnętrznej procesora. Ale na samym programowaniu się nie kończy, bowiem po tej operacji trzeba zapisaną pamięć sprawdzić – czyli fachowo mówiąc „zweryfikować". Istnieje jeszcze kilka dodatkowych operacji, które jednak omówię w dalszej części artykułu.

Zanim przejdę do omówienia sposobów programowania pamięci programu, poinformować niektórych z Was, drodzy Czytelnicy, że w zasadzie podane niżej informacje nie są niezbędne do posługiwania się kontrolerami z wbudowaną pamięcią programu i korzystania z programatorów tych kostek. Jednak aby w pełni zrozumieć zasadę działania i dodatkowe funkcje procesora, warto znać te wiadomości, a praktyka przyniesie z pewnością mniej niemiłych niespodzianek.

Oto kilka informacji, które pozwolą zrozumieć Ci wstępnie w jaki sposób można zaprogramować wewnętrzną pamięć programu procesora.

1. Wiesz, już w jaki sposób procesor wykonuje swój program, i że do tego z jakiej pamięci (zewnętrznej czy wewnętrznej) odczytywany jest program, służy końcówka /EA (pin 31). W przypadku kiedy wyprowadzeni to jest zwarte do masy procesor pobiera rozkazy z zewnętrznej pamięci programu o adresach 0000h...FFFFh, czyli maksymalnie z 64kB (65536 bajtów). W przypadku kiedy zewrzemy to wyprowadzenie do plusa zasilania (+5V) uaktywniona zostanie wewnętrzna pamięć programu (w kostkach 87C51, 89C51 lub podobnych) i kolejne rozkazy będą pobierane właśnie z niej.
2. Fizycznie wewnętrzną pamięć programu można wyobrazić sobie jako wbudowany w procesor układ reprogramowalnej pamięci EPROM (87C51) lub EEPROM (np. 89C51), który za pośrednictwem zewnętrznych końcówek procesora może być zaprogramowany przez urządzenie zewnętrzne – programator.
3. Ponieważ procesor posiada te same i niezmienne wyprowadzenia (40 dla omawianych kostek 8051/C51,87C51, 89C51/C52) programowanie i weryfikacja wewnętrznej pamięci programu odbywa się z wykorzystaniem tych samych wyprowadzeń, tylko że w tzw. „trybie programowania". W trybie tym procesor znajduje się poza układem macierzystym (tym w którym ma pracować) a umieszczony jest w pro-

Rys. 1. Ogólny sposób na programowanie procesora



gramatorze, który w odpowiedni sposób sterując pewnymi wyprowadzeniami kostki wprowadza ją w ten właśnie tryb.

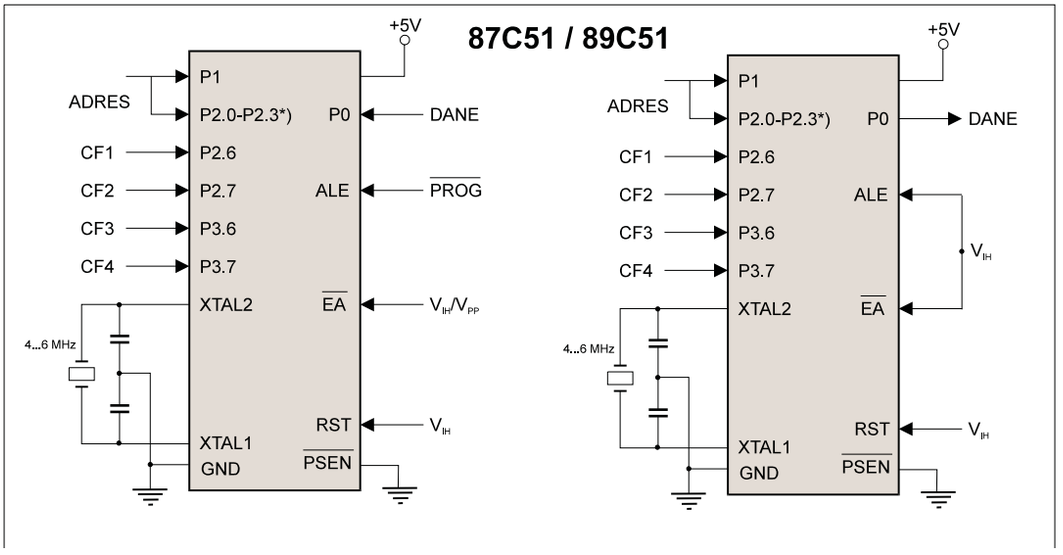
4. Skoro powiedziałem o tym że wewnętrzną pamięć programu można wyobrazić sobie jako wbudowany chip pamięci EPROM/EEPROM, to oznacza to że do „dobrania się" do niej muszą służyć:
   - linie adresowe, których liczba zależy od wielkości pamięci programu
   - linie danych : w procesorach takich jak MCS-51 będzie ich oczywiście 8
   - dodatkowe linie konfiguracyjne – sterujące zapisem i odczytem tej pamięci. Obrazowo pokazano to na **rysunku 1**
5. Wśród tych ostatnich – linii sterujących znajduje się także tzw. linia Vpp – czyli linia napięcia programującego. Z reguły napięcie to jest wyższe od napięcia zasilania procesora i wynosi:
   - 12,75 V dla większości układów z pamięcią EPROM, np. 87C51/2
   - 12V dla większości układów z pamięciami EEPROM (Flash EEPROM), np. 89C51/2. Istnieją także wersje programowane napięciem 5V (np. procesory Atmela o oznaczeniach 89C51-XX-5, gdzie XX oznacza maksymalną częstotliwość pracy układu w MHz).

   W zamierzchłych czasach istniały także wersje procesorów 8751 które programowano napięciem 21V, podobnie jak pamięci EPROM wykonywane kiedyś w technologii MOS (obecnie CMOS), ale to przeszłość i takich wersji układów na rynku się nie spotyka.
6. Dzięki stosownemu wysterowaniu wspomnianych końcówek procesora w trybie programowania, a następnie poprzez podanie napięcia programującego Vpp stosowna zaadresowana przez programator komórka w wewnętrznej pamięci programu zostaje zaprogramowana. Po obniżeniu napięci Vpp do wartości napięcia zasilającego Vcc (+5V) programator może zweryfikować zapisany bajt. Tak z grubsza odbywa się każdy cykl zapisu i sprawdzenia poprawności zaprogramowanej komórki.
7. Do prawidłowego programowania procesora potrzebny jest także dołączony do końcówek XTAL1 i XTAL2 rezonator kwarcowy, tak aby procesor mógł „oddychać" podczas programowania. Już wiesz przecież że bez sygnału zegarowego procesor jest „martwy" jak człowiek bez krwi. Wartość częstotliwości rezonansowej kwarcu nie jest w tym przypadku istotna, ważne jest aby zawierała się ona w następujących granicach:
   - 4...6 MHz dla układów z pamięcią EPROM (87C51/C52)
   - 4...20 MHz dla układów z pamięcią EEPROM/ Flash (89C51/C52)
8. Ktoś może w tym momencie zapytać: „...No dobrze, procesor pracując w trybie z zewnętrzną pamięcią programu może czytać rozkazy spod adresów 0000h...FFFFh, a ile jest tej pamięci wewnętrznej programu?...", a no tyle ile podałem w tabeli 1.

   I tak dla poszczególnych kostek :
   - 87C51, 89C51 jest to 4kB (4096 bajtów), adresy 0000h...0FFFh
   - 87C52, 89C52 jest to 8kB (8192 bajty), adresy 0000h...1FFFh
9. Inne pytanie – „...a co się stanie, kiedy to np. pracując w trybie z wewnętrzną pamięcią programu (końcówka /EA zwarta do Vcc) program dojdzie do końca obszaru wewnętrznej pamięci programu (np. do adresu 0FFFh dla kostki 87C51), co będzie wtedy, skąd będą pobierane dalsze rozkazy?...". Jeżeli tak się stanie i procesor dojdzie do końca tej pamięci to dalsze rozkazy będą pobierane z zewnętrznej pamięci programu dołączonej do procesora w tradycyjny (jak w naszym komputerku edukacyjnym AVT-2250) sposób.
   - Na **rysunku 2** pokazano sposób dołączenia sygnałów : adresowych, danych i sterujących w tym napięcia Vpp podczas programowania wewnętrznej pamięci programu kostek. W przypadku programowania procesorów w wersji z pamięci programu EEPROM „Flash" typu 89C51 /C52 na rysunku będą pewne różnice, a mianowicie: rezonator kwarcowy może być z zakresu 4...20 MHz
   - W przypadku układów 87C52 i 89C52 adres komórki do zaprogramowania podawany jest na linie portów : P1 – LSB adresu oraz P2.0 – P2.4 : MSB adresu
   - Przy programowaniu układów 87C51/52 na wejście sterujące /PROG (końcówka ALE) podczas programowania danej komórki pamięci podaje się 25 impulsów ujemnych (od Vcc do masy) o czasie trwania min. 100µs i przerwie ok. 10µs.
   - Przy programowaniu kostek 89C51 i 89C52 na wejście /PROG wystarczy podać 1 impuls ujemny o czasie trwania ok. 100µs, to wystarczy aby zaprogramować bajt w wewnętrznej pamięci programu tej kostki

10. Warto wiedzieć, że zapisaną, wewnętrzną pamięć programu można zabezpieczyć przed odczytem przez osoby niepowołane. Służą temu tzw. „Security Bits", czyli bity zabezpieczające, których odpowiednie „przepalenie" uniemożliwia odczytanie zawartości pamięci (nie bójcie się, nie robi się tego „zapalniczką", chodzi mi tu o ich zaprogra-

Rys. 2. Sposób dołączenia sygnałów sterujących podczas programowania i weryfikacji układów 87C51/52.



Uwaga *): W przypadku kostki 87C52, wykorzystana jest dodatkowa linia adresowa – końcówka P2.4, ze względu na większą – 8kB pamięć programu.

mowanie). Zabezpieczenia zawartości programu może być często użyteczne, kiedy np. mamy zamiar oferować osobom drugim swój zaprogramowany mikroprocesor pracujący w mniej lub bardziej wymyślnym urządzeniu, sprzedając go i nie chcąc jednocześnie aby ktoś skopiował nasz pomysł i powielił w setkach tysięcy egzemplarzy.

I tu kryje się istotna zaleta procesorów z wewnętrzną pamięcią programu. Otóż zauważmy, że w przypadku umieszczenia programu w zewnętrznej pamięci EPROM, praktycznie każdy ma do niej dostęp, i może korzystając z programatora pamięci EPROM odczytać jej zawartość, w celu późniejszego skopiowania. Takie postępowanie jest oczywiście niezgodne z prawem, ale kto jest w stanie dochodzić swoich praw, szczególnie, że program może zostać sprytnie zmodyfikowany przez programistę-pirata w sposób uniemożliwiający późniejsze udowodnienie mu jego winy. Najistotniejsze jest to że w takich sytuacjach nasz, często opracowywany miesiącami pomysł zostanie błyskawicznie skradziony i powielony!

11. I tu z pomocą przychodzi wewnętrzna pamięć programu i bity ją zabezpieczające. Otóż raz zapisany i zabezpieczony program w procesorze jest nie do odczytania! Nie ma sposobu aby program taki odczytać jak ze zwykłej pamięci EPROM. Jeżeli ty właśnie jesteś autorem tego programu, to i tak wszystko w porządku, bo przechowujesz gdzieś, zapewne w komputerze, kopię programu oraz listing źródłowy. A potencjalny pirat? – ten musi obejść się smakiem, bo i tak nic nie wskóra, a program będzie zabezpieczony przed jego ingerencją.
12. Pytanie: „...No tak, ale skoro zabezpieczymy już ten program, to jak go potem usunąć? Zwyczajnie. W układach z pamięcią EEPROM robi się to tak jak w przypadku pamięci EPROM, korzystając z kwarcowego, przezroczystego okienka. Procesor umieszcza się po prostu w kasowniku ultrafioletowym i po około 15 minutach jest po wszystkim, kostka jest „czysta" i gotowa do ponownego zaprogramowania i użycia.

Tabela 2

Tryb dla 87C51/C52	RST	PSEN	ALE PROG	EA/ Vpp	CF1 P2.6	CF2 P2.7	CF3 P3.6	CF4 P3.7
Zapis danych	1	0	0*	Vpp	0	1	1	1
Odczyt danych	1	0	1	1	0	0	1	1
Bity zabezpieczające B1	1	0	0*	Vpp	1	1	1	1
Bity zabezpieczające B2	1	0	0*	Vpp	1	1	0	0
Prog. tabeli szyfrującej	1	0	0*	Vpp	0	1	0	1
Odczyt sygnatury układu	1	0	1	1	0	0	0	0

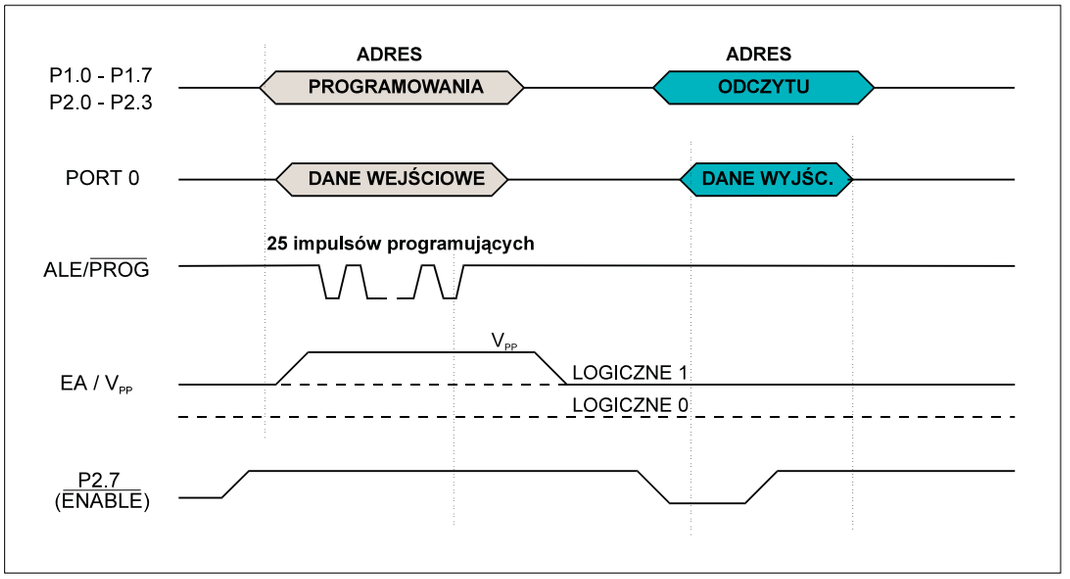
Uwagi:
a) 0* oznacza że należy podać 25 impulsów ujemnych o czasie trwania ok. 100µs i czasie przerwy min. 10µs
b) Vpp = 12,75 V ± 0,25V
c) Vcc = 5V ± 10% (dla programowania i weryfikacji)

Tabela 3

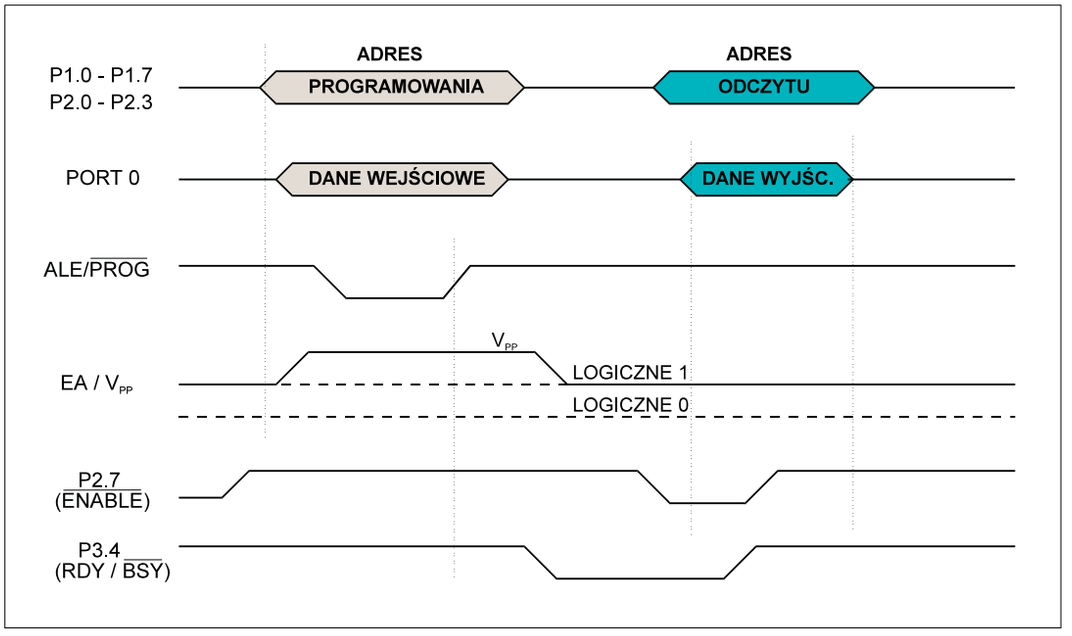
Tryb dla 89C51/C52	RST	PSEN	ALE PROG	EA/ Vpp	CF1 P2.6	CF2 P2.7	CF3 P3.6	CF4 P3.7
Zapis danych	1	0	0*	Vpp	0	1	1	1
Odczyt danych	1	0	1	1	0	0	1	1
Bity zabezpieczające B1	1	0	0*	Vpp	1	1	1	1
Bity zabezpieczające B2	1	0	0*	Vpp	1	1	0	0
Bity zabezpieczające B3	1	0	0*	Vpp	0	1	1	1
Kasowanie pamięci programu	1	0	0*	Vpp	0	1	1	1
Odczyt sygnatury układu	1	0	1	1	0	0	0	0

Uwaga:
a) 0* oznacza że należy podać 1 impuls ujemny o czasie trwania ok. 100µs
b) Vpp = 12 V ± 0,25V lub 5V ± 0,25V dla wersji 89C51/C52 – XX – 5 (patrz tekst)
c) procedura kasowania pamięci wymaga podania impulsu na wejście /PROG o czasie trwania 10 ms.

Rys. 3. Zależności czasowe podczas programowania proceseów a pamięcią EPROM (87C51, 87C52).



Inaczej jest w przypadku nowocześniejszych układów z pamięcią EEPROM „Flash". Tutaj nie jest potrzebne promieniowanie ultrafioletowe. Kasowanie wewnętrznej pamięci programu odbywa się w programatorze. Wysterowując końcówki sterujące, podając napięcie Vpp (patrz **tabela 3**) oraz impuls ujemny o czasie trwania ok. 10 ms, powodujemy skasowanie całej zawartości pamięci programu. Po takiej operacji układ jest gotowy do ponownego zaprogramowania i użycia.



Rys. 4. Zależności czasowe podczas programowania procesorów a pamięcią EEPROM/Flash (89C51, 89C52).

W zależności od ustawienia sygnałów sterujących oznaczonych na rysunku 2 jako CF1...CF4 oraz dodatkowych ALE i EA, RST i PSEN można uzyskać kilka funkcji programowania lub weryfikacji wewnętrznej pamięci programu. Wszystkie dozwolone kombinacje przedstawiają:

- dla kostek 87C51/ C52 – **tabela 2**
- dla kostek 89C51/ C52 – **tabela 3**

W dalszej części artykułu dokładnie objaśnię znaczenie poszczególnych pozycji tabel 2 i 3 podczas operacji programowania wewnętrznej pamięci mikrokontrolerów.

13. Wątpliwość: „... No tak ale przecież istnieje w liście rozkazów procesora instrukcja MOVC A,@A+DPTR, dzięki której możliwe jest odczytanie każdego bajtu z wewnętrznej czy zewnętrznej pamięci programu, co wtedy...? Odpowiadam: i na to jest rada. Otóż producenci procesorów umieścili dodatkowy bit zabezpieczający, którego „przepalenie" (zaprogramowanie) powoduje zablokowanie tej instrukcji, w wypadku kiedy ktoś próbuje wykonać ją z obszaru zewnętrznej pamięci programu – zastanów się dlaczego jest to dobry sposób na zabezpieczenie?
14. I na koniec jeszcze jedna informacja o zabezpieczeniach. Otóż niektórzy producenci procesorów rodziny MCS-51, prawie wszyscy produkujący układy w wersji z pamięcią EPROM stosują dodatkowe zabezpieczenia w postaci tzw. tablicy szyfrującej (ang. Encryption Table). Fizycznie jest to wydzielona część pamięci EPROM, o rozmiarze przeważnie równym 1, 2 lub 4 krotności 16 bajtów. Zaprogramowanie tabeli szyfrującej sekwencją 16, 32 lub 64 bajtów (zależnie od wersji układu) po uprzednim zaprogramowaniu wewnętrznej pamięci programu, powoduje, to, że w przypadku nie zabezpieczenia procesora bitami zabezpieczającymi, odczytywany przez potencjalnego hackera każdy bajt programu będzie wynikiem operacji EXNOR ("Exclusive NOR") faktycznego bajtu programu z kolejnym (modulo wielkość tabeli szyfrującej) bajtem tabeli szyfrującej. Dzięki temu bez znajomości zawartości tabeli enkrypcji (która po zaprogramowaniu nie jest dostępna) nie jest praktycznie możliwe rozkodowanie programu przez osobę nie mającą dostępu do zawartości tabeli szyfrującej. Oczywiście autor programu posiada takową kopię i wie co trzeba zrobić z odczytanym programem aby doprowadzić go do stanu „używalności".

W ostatnich czasach, ze względu na niepotrzebną często komplikację, większość producentów procesorów rodziny MCS-51 odeszła od koncepcji stosowania tabeli szyfrującej i stosuje dodatkowe bity zabezpieczające, które pokrótce opiszę w dalszej części artykułu.

Na **rysunku 3** i **4** przedstawiłem zależności czasowe pomiędzy sygnałami sterującymi podczas programowania procesorów z pamięciami EPROM (87C51/C52) i EEPROM „Flash" (89C51/C52).

Charakterystyka pamięci EPROM/EEPROM procesorów

Układy 87C51/C52 wyposażone są w pamięć EPROM wykonaną w technologii CMOS, programowana tzw. algorytmem szybkim „Quick-Pulse Programming". Algorytm ten polega na podaniu napięcia Vpp o wysokości 12,75V (na wejście EA/Vpp) a następnie podanie na wejście ALE procesora serii 25 impulsów ujemnych o czasie trwania 100µs (stan L) i przerwie min. 10µs (stan H).

W wypadku procesorów 89C51/ C52 napięcie Vpp może mieć jedną z dwóch wartości: 12V i 5V w zależności od wersji procesora (patrz tekst wyżej).

Mikrokontrolery posiadają tzw. sygnatury, dzięki którym możliwa jest autoidentyfikacja układu przez obsługujący programator. Fizycznie są to pojedyncze komórki ROM wbudowane w procesor z zapisanymi bajtami mówiącymi o producencie układu, jego typie oraz wersji wykonania. Są to:

a) dla układów z pamięcią EPROM
   - (adres: 030h) = 15h, oznacza producenta (w tym wypadku jest to Philips)

Tabela 4

Tryb	LB1	LB2	LB3	Rodzaj zabezpieczenia
1	U	U	U	Program nie zabezpieczony
2	P	U	U	Ignorowane są instrukcje MOVC, wykonywane z pamięci zewnętrzne programu, wejście EA jest zatrzaśnięte podczas "resetu" procesora", zablokowane jest dalsze programowanie pamięci Flash
3	P	P	U	Tak jak w trybie 2 z zablokowaną możliwością weryfikacji programu
4	P	P	P	Tak jak w trybie 3 z zablokowaną możliwością pobierania rozkazów z zewnętrznej pamięci programu.

LB1 - bit nr 1, LB2 - bit nr 2, LB3 - bit nr 3
P - bit zaprogramowany
U - bit niezaprogramowany

- (adres: 031h) = 92h, oznacza układ 87C51, podobnie dla 87C52 jest to 97h

b) dla układów z pamięcią EEPORM / Flash (np. producenta – firmy Atmel)
- (adres: 030h) = 1Eh, oznacza producenta w tym przypadku Atmel)
- (adres: 031h) = 51h, oznacza układ 89C51
- (adres: 032h) = FFh oznacza napięcie Vpp=12V, =05h oznacza Vpp=5V.

Ze względu na różnorodność typów układów jak i braku jednolitego standardu wśród producentów, podane wartości mogą się zmieniać. Należy więc je traktować jako informacyjne. Na zakończenie wspomnę że dostęp do sygnatury odbywa się poprzez zaadresowanie (adres podany w nawiasie) sygnatury tak jak to się odbywa w przypadku weryfikacji pamięci programu, z tą różnicą, że odmienny jest układ sygnałów sterujących – patrz tabele 2 i 3.

Do prawidłowego zaprogramowania kości potrzebny jest dołączony zewnętrzny oscylator kwarcowy (patrz rys.2) Powodem zastosowania tego elementu jest fakt, że podczas programowania, pracuje licznik wewnętrznego adresu procesora oraz odbywa się transfer danych z rejestrów portów procesora do matrycy pamięci programu.

Podczas programowania, adres danej komórki EPROM / EEPROM podawany jest przez programator na port P1 procesora (młodsza część adresu – LSB) oraz na część pinów portu P2. Przy układach 'C51 wyposażonych w 4 kB pamięci programu są to linie P2.0... P2.3, a w kostkach 'C52 dodatkowo sterowana jest linia P2.4. W ten sposób dzięki 12 liniom adresowym (A0...A11) dla 'C51 oraz 13 liniom adresowym (A0...A12) dla kostek 'C52 możliwe jest zaadresowanie całej wewnętrznej pamięci programu.

Jeżeli chodzi o dane to programator podaje je na port P0 procesora. Następnie ustawione zostają (zgodnie z tabelą 2 dla układów z EPROM oraz tabelą 3 dla układów z EEPROM Flash) sygnały RST i PSEN oraz wybrana zostaje konfiguracja pinów CF1...CF4 określających zgodnie z tymi tabelami operacje na procesorze jaka ma być właśnie wykonana. Przy programowaniu sekwencja poziomów logicznych sygnałów CF1...CF4 będzie równa: 0-1-1-1.

Następnie (patrz zależności czasowe na rys.3 i 4) programator podaje napięcie Vpp na końcówkę EA procesora, po czym po krótkiej chwili, kiedy napięcie to narośnie do odpowiedniej wartości programator
– dla układów z EPROM : generuje szereg impulsów programujących (25) o parametrach jak podałem wcześniej w artykule
– dla układów z EEPROM jest to 1 impuls o określonym czasie trwania (zazwyczaj jest to 100µs)

Następnie programator obniża do pierwotnej wartości Vcc wartość napięcia programującego Vpp, po czym wystawia poziom niski na linię CF2, co powoduje że zapisana przed chwila dana jest wystawiana, tym razem przez procesor na linie portu P0 celem weryfikacji (odczytu) przez urządzenie programujące.

W układach 89C51/52 programator może monitorować stan programowania komórki za pośrednictwem dodatkowej linii P3.4 procesora. Otóż po zapisaniu danej w pamięci wewnętrznej programu, procesor sygnalizuje to pojawieniem się stanu niskiego na tej linii, co może odczytać programator i w ten sposób skrócić niezbędny czas impulsu programującego, podawanego na wejście /PROG procesora.

Programowanie tablicy szyfrującej („Encryption Table") w procesorach z pamięcią EPROM odbywa się podobnie, jak w przypadku programowania pamięci programu, lecz inna jest kombinacja sygnałów sterujących CF1...CF4 (patrz tabela 2).

Podobnie wygląda programowanie bitów zabezpieczających, z tą różnicą, że do przepalenia danego bitu wystarczy kombinacja sygnałów CF1..CF4 oraz jak poprzednio cykl programujący z Vpp (jak poprzednio). Linie adresowe oraz danych nie mają w tym momencie znaczenia.

W tabeli 4 przedstawione są efekty przepalania kolejnych bitów zabezpieczających. W przypadku układów 87C51/C52 mamy do czynienia tylko z dwoma bitami LB1 i LB2. Przepalenie LB1 zabezpiecza układ przed przyszłym programowaniem, czy raczej „doprogramowaniem" tej części pamięci, która nie została wcześniej podczas programowania zapisana (nie musimy przecież programować całej pamięci programu, a tylko tyle ile ma nasz program. Przepalenie bitu LB2 powoduje zablokowanie możliwości weryfikacji zaprogramowanej pamięci programu.

Dlatego należy pamiętać, **że bity zabezpieczające programuje się w zależności od potrzeb, ale zawsze na końcu całego procesu programowania!**

W układach z 89C51/C52 możliwe jest elektryczne kasowanie całej zawartości wewnętrznej pamięci programu poprzez podanie 10 milisekundowego impulsu programującego przy pozostałych sygnałach CF1...CF4 ustawionych jak podano w tabeli 3.

W wypadku układów z EPROM kasowanie pamięci może być wykonane tylko za pośrednictwem promieni ultrafioletowych (podobnie jak w typowych kostkach EPROM) o długości fali najlepiej około 4000A (400 nm).

Do kasowania układów najlepiej jest stosować fabryczne kasownika EPROM lub samodzielnie wykonaną lampę kasująca np. ze świetlówki pracującej w podanym zakresie fal.

W tym miejscu chce przestrzec niektórych z Was o możliwości zastosowania w roli lampy kasującej występujących na naszym rynku małych świetlówek do tzw. „sprawdzania banknotów". Niestety nie nadają się one do kasowania struktur EPROM ze względu na nieodpowiednie widmo promieniowania.

Trzeba zatem nabyć specjalną świetlówkę, najlepiej miniaturową (o mocy 4...8W) emitująca stosowne promieniowanie. Cechą charakterystyczną właściwej dla naszych celów świetlówki jest całkowita przezroczystość rurki — czyli brak luminoforu. Jeżeli natraficie w sklepie na taki egzemplarz, pracujący w zakresie ultrafioletu, to z pewnością kasowanie układów za pomocą takiej lampy będzie udane.

Pamiętajcie tylko aby własnoręcznie wykonany „kasownik" zamknąć w obudowie, w przeciwnym przypadku ostre promieniowanie ultrafioletowe może uszkodzić wzrok!

Jeżeli ktoś chciałby spróbować samodzielnie wykonać chociażby najprostszy programator i zaprogramować procesor na podstawie niniejszego artykułu, zalecam sięgniecie do literatury [1] i [2]. Uprzedzam jednak że ze względu na różnorodność parametrów czasowych w przebiegach z rys.3 i 4 w zależności od producenta układu oraz jego wersji, nie jest to zadanie łatwe, a przynajmniej nie da się zrobić na kolanie. Można bowiem drogi często procesor po prostu uszkodzić. Dlatego jeszcze raz zalecam korzystanie z gotowych programatorów lub złożenie samodzielne proponowanego w artykule programatora AVT.

Sławomir Surowiński

Literatura:

[1] – 80C51 Based 8-bit Microcontrollers, katalog Philips IC20

[2] — Microcontrollers DataBook, Atmel 1995/97