Jak z ZX SPECTRUM zrobić komputer?

ZMIENIAMY ARCHITEKTURĘ KOMPUTERA

Wojciech Apel

Artykuł ten stanowi kontynuację tematu udoskonalenia mikrokomputera ZX Spectrum rozpoczętego w numerze 1/1987 naszego kwartalnika. Proponowane w niniejszym artykule przeróbki mikrokomputera ZX Spectrum mają na celu zmianę stosunkowo sztywnej struktury sprzętu. Polegają one na dołączeniu dodatkowych stron (bloków) pamięci tak, aby konfiguracja tego systemu dała się łatwo zaadaptować do dowolnego układu.

Przypatrzmy się mapie pamięci ZX Spectrum. Od adresów 0 do 3FFFH procesor „widzi" pamięć ROM zawierającą rezydujący tam interpreter języka BASIC. Powyżej, tj. od 4000H do 7FFFH zlokalizowana jest pamięć RAM o pojemności 16 KB, w której znajduje się obraz ekranu, zmienne systemowe i rozwijająca się w górę treść programu w języku BASIC (rys. 1). Pamięć zawierającą treść ekranu nazwiemy VRAM (Video-RAM). Komputery 48K są wyposażone w opcjonalną pamięć RAM 32K umieszczoną w przestrzeni adresowej od 8000H do FFFFH (podobną strukturę stosuje się w komputerach MERITUM czy Laser). Struktura taka posiada jednak pewne wady.

Pierwsza z nich to wyróżnienie pewnego obszaru RAM, jako pamięci obrazu. Obszar ten należy do przestrzeni adresowej procesora, a więc umieszczanie w nim programów w języku wewnętrznym mikroprocesora jest możliwe. Jednak wtedy na ekranie pojawiają się tzw. śmieci nie będące żadną informacją dla użytkownika. Przykładem tego może być znany program kopiujący COPY-COPY.

Drugą wadą takiej struktury jest umiejscowienie pamięci ROM od adresu 0. Po sygnale RESET (np. po włączeniu zasilania) procesor startuje wykonując program od początku przestrzeni adresowej (tj. od adresu 0) — powinien więc tam znajdować się program uruchamiający komputer. W bardziej rozbudowanych rozwiązaniach stosuje się w tym miejscu małe pamięci ROM, w których znajduje się program ładujący system operacyjny z urządzenia zewnętrznego. Po załadowaniu i uruchomieniu systemu operacyjnego odłącza on tę małą pamięć i odsłania istniejącą pod nim pamięć RAM. Ta mała pamięć ROM nosi nazwę BOOTSTRAP. Rozwiązanie takie stosuje się np. w mikrokomputerach pracujących pod kontrolą systemu operacyjnego CP/M. System ten wykorzystuje komórki od 0 do 255, jako zmienne systemowe i posiada możliwość definiowania *restartów* dla programów uruchamianych pod jego kontrolą — konieczna jest więc pamięć RAM.

Proponowane niżej przeróbki Spectrum mają na celu usunięcie najistotniejszych wad jego struktury, co pozwoli na pracę pod kontrolą systemu CP/M. Oto wymagania stawiane mikrokomputerowi po przeróbce:

1. Wszystkie modyfikacje nie mogą zmienić struktury mikrokomputera w sposób istotny, tzn. musi istnieć możliwość pracy z normalnym interpreterem języka BASIC zapisanym w firmowej pamięci ROM.

2. Wszystkie programy napisane na mikrokomputer ZX Spectrum muszą pracować na przerobionym komputerze. Założenie to obejmuje także programy napisane w języku wewnętrznym komputera.

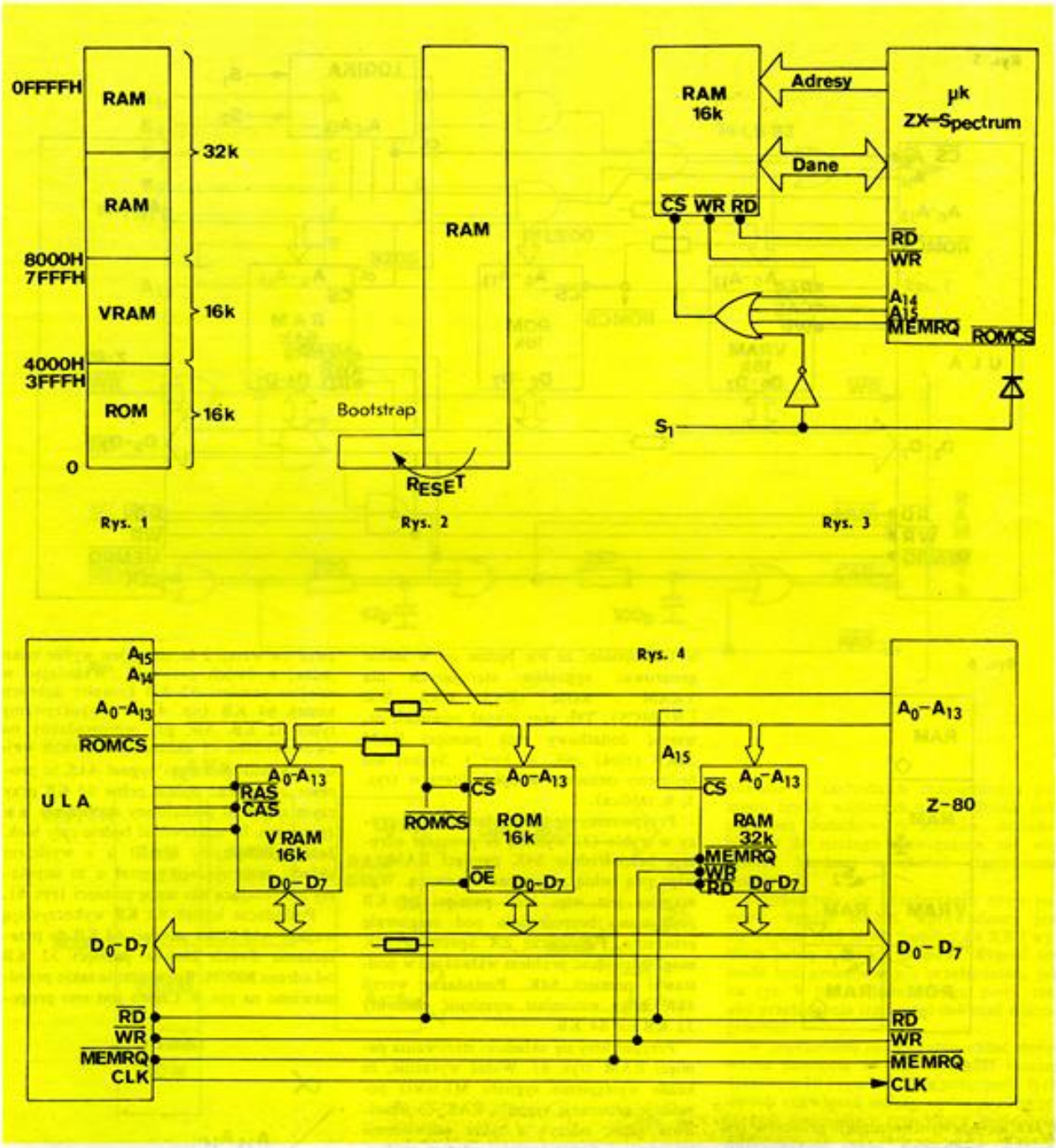
3. Wszystkie zmiany muszą zmieścić się w normalnej obudowie komputera i muszą być zasilane ze standardowego zasilacza.

4. Wszystkie sygnały na złączu krawędziowym muszą być zgodne z standardowym ZX Spectrum. Gwarantuje to możliwość pracy ze wszystkimi interfejsami fabrycznymi i amatorskimi.

5. Możliwość pracy pod kontrolą systemu CP/M, lub wszystkich innych systemów bazujących na mikroprocesorze Z-80 (np. ISIS), a nie wymagających więcej, niż 64 KB pamięci RAM. Pozwala to wykorzystać dużą część bogatego oprogramowania pracującego pod tymi systemami.

6. Dostęp do przerwań maskowalnych i niemaskowalnych.

Przedyskutujmy teraz podłączenie pamięci RAM tak, aby obejmowała obszar od 0 do 3FFFH. W mikrokomputerze ZX Spectrum przewidziano możliwość wyłączenia pamięci ROM. Na złączu krawędziowym dostępny jest sygnał ROMCS. Po podaniu na tą linię logicznej jedynki pamięć ROM przestaje być widoczna dla procesora. Ten sam sygnał może załączyć dodatkową pamięć RAM zgłaszającą się pod tym adresem. Przedstawiono to 3 (zakładamy tu istnienie pamięci RAM 16KB oznaczonej w opisie karcianym symbolem „pik"). W ten prosty sposób, korzystając



Rys. 1 Rys. 2 Rys. 3

Rys. 4

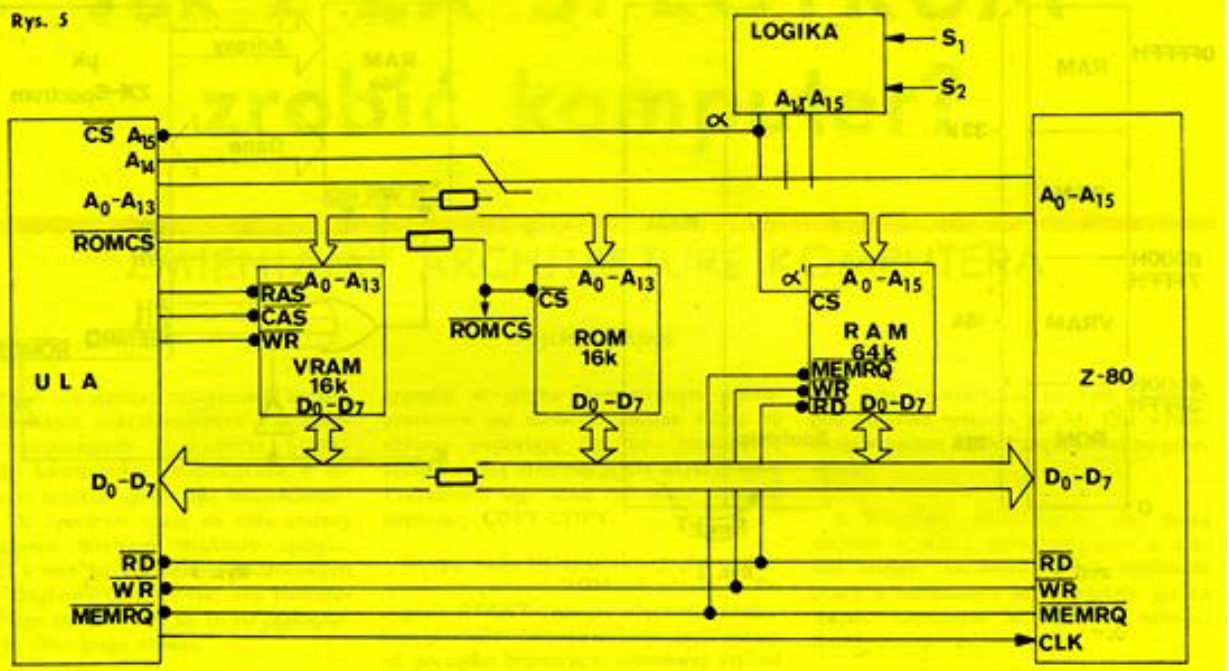
z sygnału ROMCS wyłączyliśmy ROM. Podobne rozwiązanie zastosowali konstruktorzy INTERFACE I. Jak się jednak dalej okaże, można rozwiązać ten problem zupełnie inaczej.

Spróbujmy przedyskutować teraz problem podłożenia pamięci RAM w miejsce drugiej ćwiartki przestrzeni adresowej mikroprocesora (oznaczonej „kier"). Problem nie jest łatwy i aby go rozwiązać, należy przypatrzeć się pracy mikroprocesora w II i III ćwiartce pamięci.

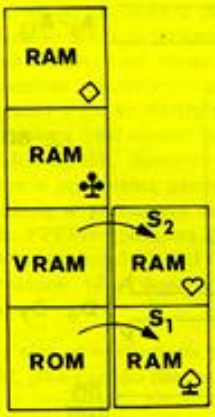
W III ćwiartce mikroprocesor adresuje pamięć RAM, a ta zgłasza się w cyklu RD (READ-odczyt) lub WR (WRITE-zapis). Jednocześnie w mikrokomputerze zachodzi drugi, całkowicie niezależny proces wyświetlania obrazu. Procesor graficzny (ULA) adresuje pamięć VRAM, a ta zgłasza się w cyklu RD. Możliwe jest więc zgłaszanie się w tej samej chwili pamięci VRAM i procesora lub pamięci RAM. Problem konfliktów na magistrali rozwiązany jest za pomocą rezystorów, które rozdzielają szynę danych na szynę procesora i szynę ULA (patrz rys. 4). Rezystory te pozwalają na to, aby w mikrokomputerze ZX Spectrum przebiegały jednocześnie i niezależnie dwa procesy: proces tworzenia obrazu przez ULA i praca mikroprocesora operującego na obszarach pamięci RAM „trefl" i „karo".

Przypatrzmy się cyklowi dostępu do pamięci w II ćwiartce przestrzeni adresowej. Procesor w stanie T1 wystawia adres, który informuje ULA o żądaniu dostępu do pamięci VRAM. W chwili, gdy ULA adresuje pamięć procesor nie może otrzymać do niej dostępu. Zostaje więc zatrzymany w stanie T2 do chwili, gdy ULA zwolni swoją część magistrali. ULA wygeneruje wtedy sygnały sterujące pamięć VRAM (RAS, CAS, WR) dla adresów procesora. Dostępy procesora do VRAM w czasie wygaszania sygnału wizji odbywają się bezkonfliktowo.

Rys. 5



Rys. 6



Taka metoda synchronizacji procesów powoduje wydłużenie czasu wykonywania jednego z nich. W ten sposób można wytłumaczyć fakt, że programy w języku maszynowym umieszczone w przestrzeni VRAM pracują o około 20% wolniej.

Z rozważań tych wynika, że gdy mikroprocesor nie adresuje pamięci VRAM, to jego praca nie jest niczym zakłócana, a procesor graficzny (ULA) kontroluje pamięć VRAM jako niezależny proces. Podłączenie dodatkowej pamięci RAM „kier" jest możliwe wtedy, gdy pod adresami II ćwiartki przestrzeni adresowej nie będzie zgłaszała się pamięć VRAM. Można to uzyskać traktując nóżkę A15 układu ULA jako sygnał CS. Sygnał ten zablokuje ULA w taki sposób, że nie będzie on w stanie generować sygnałów sterujących dla VRAM i ROM (RAS, CAS, WR i ROMCS). Ten sam sygnał powinien otwierać dodatkowy blok pamięci RAM 32KB (bloki „pik" i „kier"). Sygnał ten będziemy oznaczać w opisie literą α (rys. 5, 6, tablica).

Przypatrzmy się tablicy funkcji α. Z pracy w trybie (3) wynika, że procesor adresuje bezpośrednio 64K pamięci RAM, a więc swą pełną przestrzeń adresową. Wymagany jest więc blok pamięci 64 KB podłączony bezpośrednio pod magistralę procesora. Posiadacze ZX Spectrum 16K mogą rozwiązać problem wkładając w podstawki pamięci 64K. Posiadacze wersji 48K mogą natomiast wymienić elementy 32 KB na 64 KB.

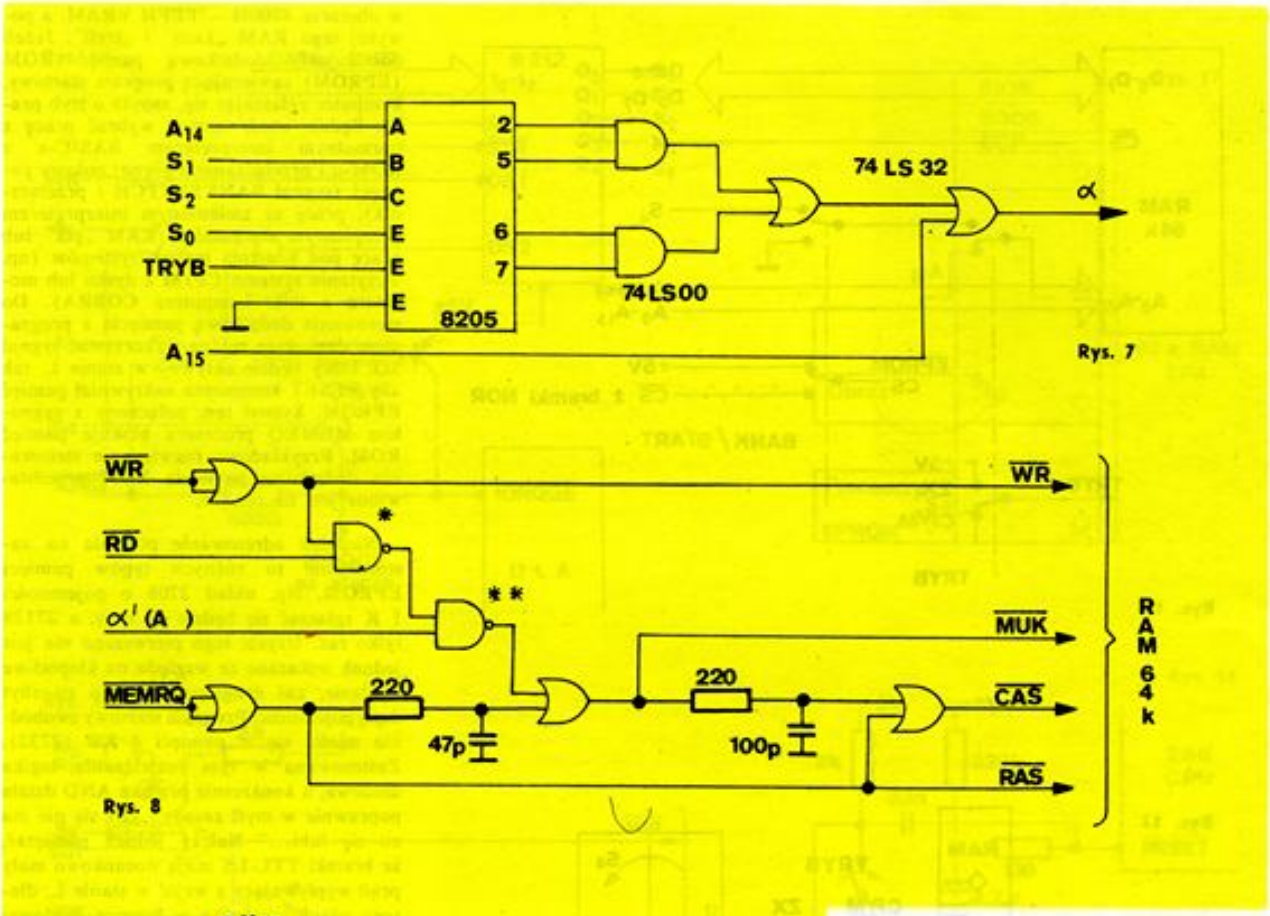
Przypatrzmy się układowi sterowania pamięci RAM (rys. 8). Widać wyraźnie, że każde wystąpienie sygnału MEMRO powoduje generację sygnału RAS, co umożliwia zapis, odczyt, a także odświeżenie tej pamięci. Cykle zapisu i odczytu są rozpoznawane na bramce * pod warunkiem, że pamięć ta jest wybrana (A15 = 1 na bramce **). Powstaje sygnał MUX przełączający multipleksery, a potem sygnał CAS odblokowujący bufory pamięci. Odświeżanie powoduje tylko generację sygnału RAS, ponieważ w takim cyklu pamięciowym nie są generowane sygnały RD i WR. Jak widać, sygnał A15 wykorzystany jest w ZX Spectrum jako CS bloku 32 KB. W wersji 48K producent umieszcza uszkodzone pamięci 64 KB, wybierając je tak, aby komplet 8 kostek miał te same dobre połówki (32 KB). Wyboru tych połówek można dokonać za pomocą odpowiednich zwor (w wersji 2 możliwy jest wybór tylko jednej z dwóch połówek). Wkładając w miejsce pamięci 32 KB komplet dobrych kostek 64 KB (np. 4164) wykorzystamy tylko 32 KB. Ale, gdy wprowadzimy na zworę (nóżka 11 układu IC 26, czyli wejście 3A multipleksera) sygnał A15, to procesor adresować będzie pełne 64 KB przy czym sygnał α podłączony do bramki ** (patrz rys. 8) uaktywniać będzie cały blok. Jeżeli podłączymy sygnał α z wyjściem układu generującego sygnał α to uzyskamy interesującą nas mapę pamięci (rys. 6).

Posiadacze kostek 64 KB wykorzystują wejście A15 bloku pamięci 64 KB do przełączania dwóch bloków pamięci 32 KB od adresu 8000H. Rozwiązanie takie przedstawiono na rys. 9. Często jest ono propo-

Tablica 1

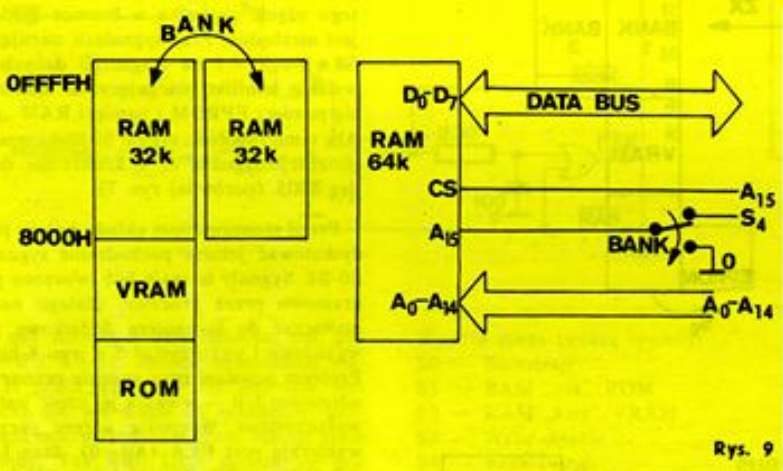
α

S_2 S_2 \ A_{15} A_{14}	0 0	0 1	1 1	1 0	
0 0	0	0	1	1	(1)
0 1	1	0	1	1	(2)
1 1	1	1	1	1	(3)
1 0	0	1	1	1	(4)
	I ćw	II ćw	IV ćw	III ćw	

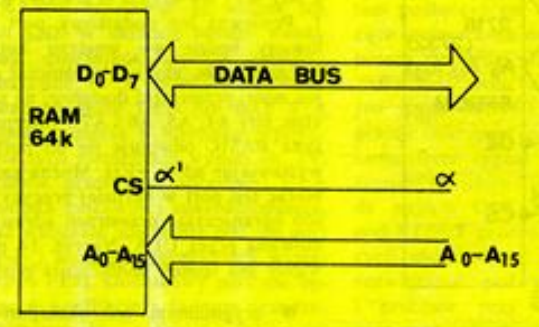


Rys. 7

Rys. 8



Rys. 9



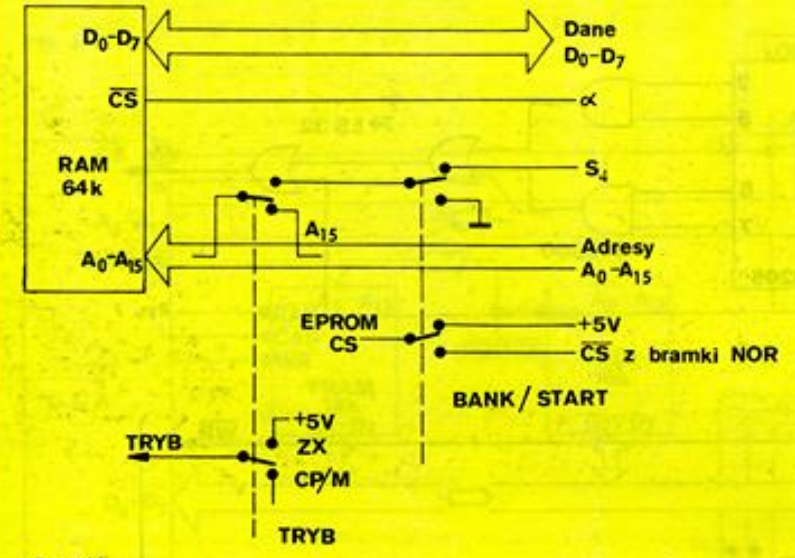
Rys. 10

nowane w zachodnich czasopismach pomimo braku większych zalet. Można jednak, bez dodatkowych kosztów, dobudować je do naszego rozwiązania, tak aby jeszcze bardziej zwiększyć elastyczność systemu.

Przypatrzmy się przełączaniu stron pamięci. Sygnał BANK wprowadzany jest na wejście A15 bloku pamięci 64 KB i wybiera jedną z dwóch połówek. Sygnał ten może być generowany z przełącznika, jak na rys. 9 lub z dodatkowego portu tak, aby przełączenia tego mógł dokonać mikroprocesor.

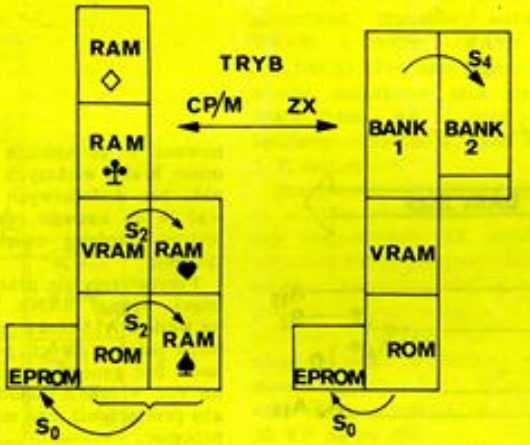
W poprzednim rozwiązaniu układ sterowania pamięcią 64 KB wyglądał inaczej (por. rys. 10 i rys. 11). Dla połączenia tych dwóch rozwiązań należy zastosować przełącznik zmieniający tryb pracy komputera (nazwane tu CP/M<−>ZX). Przełączanie wejścia A15 pamięci odbywa się ręcznie. Jednocześnie na wejście CS pamięci podawany jest sygnał α, który w przypadku zablokowania dekodera 8205 sygnałem TRYB jest równy A15. Sygnał BANK można podłączyć tak, aby mógł być przełączany zarówno przez mikroprocesor, jak i ręcznie. Przełączanie ręczne jest możliwe, gdy S4 jest w stanie wysokim (patrz rys. 9), a przełączanie programowe, gdy przełącznik jest w odpowiedniej pozycji. Uzyskaliśmy w ten sposób mapę pamięci przedstawioną na rys. 12.

Jeżeli po załączeniu komputera sygnały sterujące S1 i S2 będą wyzerowane, to otrzymamy konfigurację normalnego ZX Spectrum, tzn. od 0 do 3FFFH ROM,

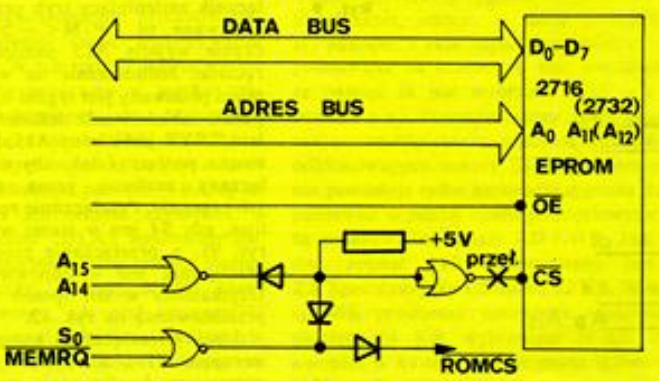


Rys. 11

Rys. 12



Rys. 13

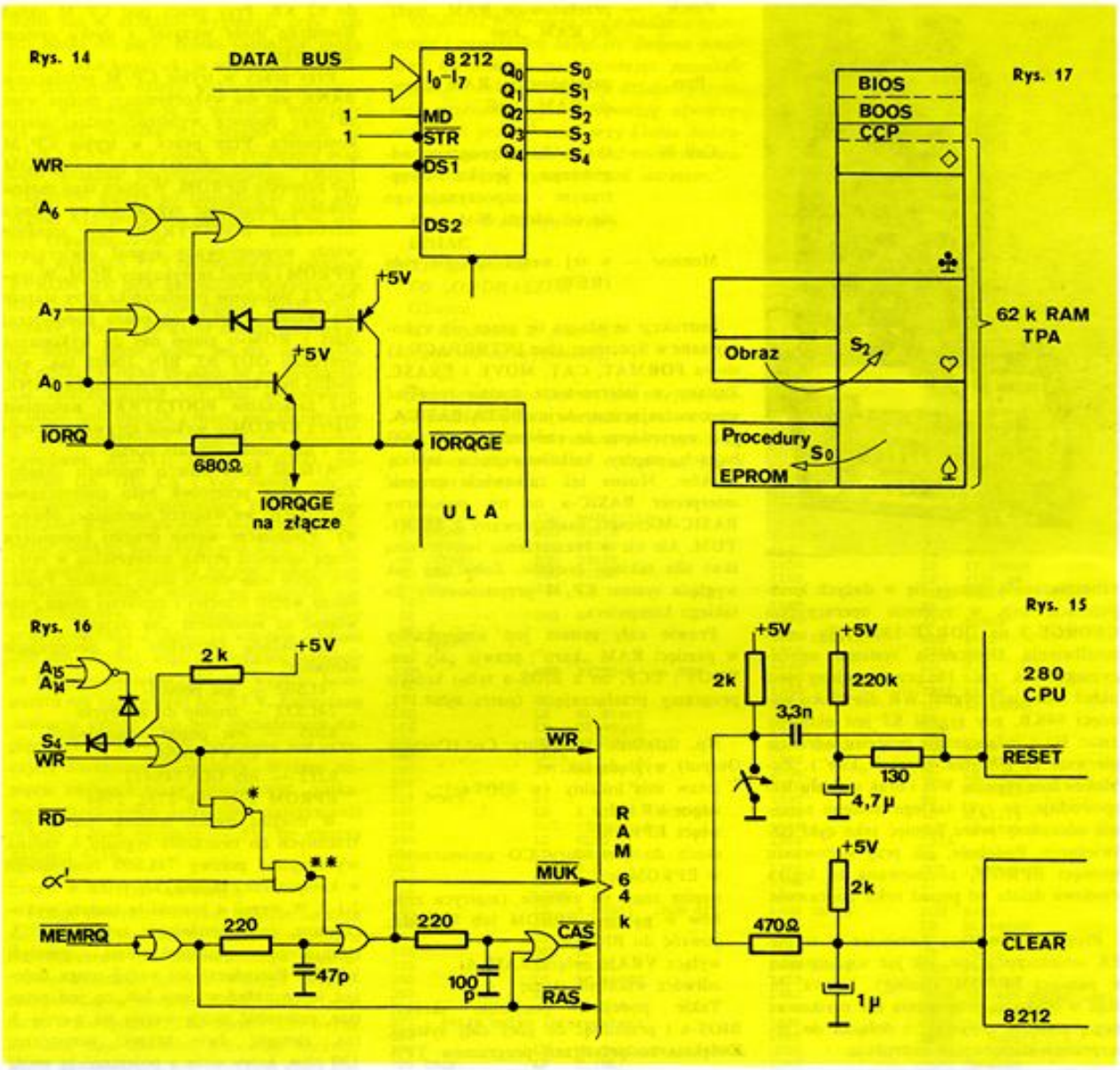


w obszarze 4000H—7FFFH VRAM, a powyżej tego RAM „karo" i „trefl". Jeżeli dobudujemy dodatkową pamięć ROM (EPROM) zawierającą program startowy, komputer zgłaszając się, zapyta o tryb pracy. Będzie wtedy można wybrać pracę z normalnym interpreterem BASIC-a z ROM-u i przełączaniem górnej połowy pamięci (sygnał BANKSWITCH i przełącznik), pracę ze zmienionym interpreterem przepisanym do pamięci RAM „pik" lub pracę pod kontrolą innych systemów (np. wczytaniu systemu CP/M z dysku lub monitora z mikrokomputera COBRA). Do sterowania dodatkową pamięcią z programem startowym można wykorzystać sygnał SO, który będzie aktywny w stanie L, tak aby RESET komputera uaktywniał pamięć EPROM. Sygnał ten, połączony z sygnałem MEMRQ procesora blokuje pamięć ROM. Przykładowe rozwiązanie sterowania dodatkową pamięcią ROM przedstawione jest na rys. 13.

Niepełne adresowanie pozwala na zastosowanie tu różnych typów pamięci EPROM. Np. układ 2708 o pojemności 1 K zgłaszać się będzie 16 razy, a 27128 tylko raz. Użycie tego pierwszego nie jest jednak wskazane ze względu na kłopotliwe zasilanie, zaś drugiego, z uwagi na zbyt dużą pojemność. Program startowy swobodnie mieści się w pamięci 4 KB (2732). Zastosowana w tym rozwiązaniu logika diodowa, a konkretnie bramka AND działa poprawnie w myśl zasady: „gdy się nie ma co się lubi..." Należy jednak pamiętać, że bramki TTL-LS mają stosunkowo mały prąd wypływający z wejść w stanie L, dlatego użycie opornika w bramce diodowej jest niezbędne. Przy sygnałach sterujących S0 = 0 i S1 = 1 na magistrali danych powstanie konflikt polegający na zgłaszaniu się pamięci EPROM i pamięci RAM „pik". Aby temu zapobiec, sygnał S0 blokuje układ generacji sygnału α, a konkretnie dekoder 8205 (porównaj rys. 7).

Przed montowaniem układu należy przedyskutować jeszcze pochodzenie sygnałów S0-S4. Sygnały te mają być tworzone programowo przez procesor, dlatego należy podłączyć do komputera dodatkowy port wyjściowy i wykorzystać 5 z jego 8 bitów. Problem powstaje przy analizie przestrzeni adresowej I/O — większa jej część jest już wykorzystana. Wszystkie adresy parzyste wybierają port ULA (A0 = 0). Stan L na A2 wybiera ZX Printer i inne drukarki, a bity A3 i A4 sterują pracą INTERFACE 1. Ponieważ ten dodatkowy port zainstalowany będzie we wnętrzu komputera, wskazane jest, aby nie ograniczał on i tak już małej przestrzeni dostępnej na zewnątrz (tzn. bity A1, A5, A6 i A7). Interpreter języka BASIC odwołuje się do portu ULA wystawiając adres FEH. Można zatem wybierać ten port w bardziej precyzyjny sposób ograniczając przestrzeń adresową zajmowaną przez ULA. Na rys. 14 przedstawiony jest sposób podłączenia portu 8212.

W oryginalnym mikrokomputerze ZX Spectrum port ULA zgłasza się w cyklu



I/O gdy A0 = 0; w przerobionym zaś, gdy A0 = 0 i A7 = 1. Sposób wymuszenia stanu H przez tranzystor nie jest zbyt elegancki, ale jak widać na schemacie, producent mikrokomputera także stosuje takie sposoby. Tak więc ULA zgłasza się pod adresami 1XXXXXXOB, a nasz port pod 00XXXXXOB; adresować go można np. 3EH lub 62D. W układzie bardzo ważna jest dioda obniżająca napięcie załączające tranzystor — w praktyce najlepiej zastosować tu diodę LED lub BAP 813, gdyż spadek napięcia w kierunku przewodzenia na tych elementach wynosi ok. 2V. Ograniczenie przestrzeni adresowej ULA nie zmniejsza możliwości mikrokomputera, ponieważ wszystkie odwołania pod ten port w ROM-ie i w innych programach wystawiają adres FEH. Dodatkowy port nie będzie więc w konflikcie z żadnym urządzeniem zewnętrznym.

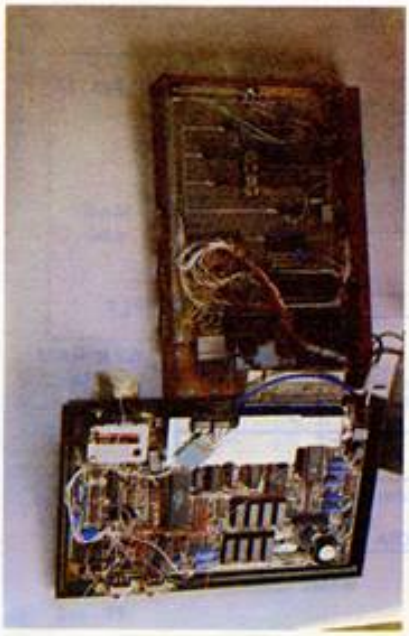
Wyjścia portu tworzą sygnały:

S0 — Bootstrap
S1 — RAM „pik"/ROM
S2 — RAM „kier"/VRAM
S3 — Write-disable
S4 — Bank-switch

Pozostałe nie są wykorzystane i można tam podłączyć np. dodatkowe diody świecące umieszczone na płycie czołowej komputera lub drugi głośniczek.

Po załączeniu komputera generowany jest sygnał RESET zerujący licznik programu mikroprocesora. W rozbudowanym komputerze sygnał ten powinien zerować sygnały sterujące S0—S4. Port 8212 posiada wejście CLEAR, ale podłączenie go pod RESET procesora nie da zadowalających efektów. Na rys. 15 przedstawiono rozwiązanie pozwalające zerować port i procesor przy załączeniu komputera oraz zerować procesor przyciskiem. Rezystor 470 ohm spowoduje zerowanie 8212 przy dłuższym załączeniu przycisku. Analizując ten układ należy pamiętać, że komputer może pracować poprawnie tylko wtedy, gdy sygnał CLEAR z 8212 ustąpi.

Po przepisaniu interpretera BASIC-a do pamięci RAM „pik" stwierdzono, że pierwsze pięć komórek zmieniło swe wartości. Okazuje się, że interpreter zapisany w ROM-ie niszczy sam siebie — twórcy oprogramowania ułatwili sobie zadanie zakładając, że ich dzieło zapisane będzie na stałe w pamięci ROM. I tak na przykład przejście przez procedurę SKIP-CONSTANS (33F7H) niszczy 5 pierwszych komórek pamięci RAM „pik", a niektóre funkcje edytora niszczą matryce znaków graficznych. Tak więc, aby nie przerabiać za bardzo interpretera BASIC-a, do układu dorobiono sprzętowe zabezpieczenie przed zapisem w pamięci RAM „pik". Podobne



zabezpieczenie stosuje się w dużych komputerach (np. w systemie operacyjnym GEORGE-3 na ODRZE-1305) dla uniemożliwienia zniszczenia systemu operacyjnego. Na rys. 16 przedstawiony jest układ blokujący sygnał WR dla bloku pamięci 64KB, gdy sygnał S3 jest aktywny (stan H) i jednocześnie procesor adresuje pierwsze 16 KB (tzn. pamięć „kier"). Zablokowanie sygnału WR i brak sygnału RD spowoduje, że cykl takiego dostępu zostanie odczytany przez pamięć jako cykl odświeżania. Podobnie, jak przy sterowaniu pamięci EPROM, zastosowana tu logika diodowa działa od ponad roku poprawnie.

Program inicjujący komputer w trybie ZX umieszczony jest, jak już wspomniano w pamięci EPROM (listing). Usuwa on błąd w obsłudze przerwania nie maskowanego, zmienia *copyright* i dołącza do interpretera następujące instrukcje:

Pusch — przeładowuje RAM „trefl" do RAM „kier"

Pop — przeładowuje RAM „kier" do RAM „trefl"

Call W — skacze do podprogramu napisanego w języku wewnętrznym rozpoczynającego się od adresu W

Monitor — w tej wersji nic nie robi (RET)

Instrukcje te włącza się przez nie wykorzystane w Spectrum (bez INTERFACE 1) słowa FORMAT, CAT, MOVE i ERASE. Zmiany w interpreterze można rozwijać wprowadzając instrukcje z BETA-BASICA, pod warunkiem, że zmieszczą się w 700 bajtach między kalkulatorem, a tablicą znaków. Można też całkowicie zmienić interpreter BASIC-a na np. popularny BASIC-Microsoft zaadaptowany z MERITUM. Ale nie w rozszerzeniu interpretera tkwi siła takiego systemu. Zobaczmy jak wygląda system CP/M przystosowany do takiego komputera.

Prawie cały system jest umieszczony w pamięci RAM „karo"; prawie cały tzn. BDOS i CCP, bo z BIOS-a tylko krótkie programy przełączające (patrz rys. 17).

Np. działanie procedury Co (Console Output) wygląda tak:

ustaw stos lokalny (w BIOS-ie);
włącz VRAM;
włącz EPROM;
skocz do procedury CO umieszczonej w EPROM-ie;
napisz znak na ekranie (matryca znaków w pamięci EPROM lub VRAM);
powróć do BIOS-a;
wyłącz VRAM, włącz RAM A;
odtwórz wskaźnik stosu;

Takie podejście pozwoliło skrócić BIOS-a i przesunąć do góry cały system. Zwiększa to przestrzeń programów TPA do 62 KB. Przy pracy pod CP/M układ Bootstrap może wczytać z dysku system operacyjny.

Przy pracy w trybie CP/M przełącznik BANK jest nie wykorzystany, można więc za jego pomocą wybierać rodzaj startu komputera. Przy pracy w trybie CP/M komputer może zgłaszać się pamięcią ROM lub pamięcią EPROM. Wyboru tego można dokonać odłączając sygnał S0 od układu sterowania BOOTSTRAP. Nie zostanie wtedy wygenerowany sygnał otwierający EPROM i sygnał zamykający ROM. W trybie ZX położenie przełącznika przy starcie ma podobny wpływ na pracę komputera; start z ROM-u zmusi nas do wykonania instrukcji OUT 62, BIN 10001 tak, aby można było korzystać z przełącznika BANK bez załączania BOOTSTRAP, natomiast start z EPROM-u wykona sprawdzenie trybu i sam ustali sygnały portu.

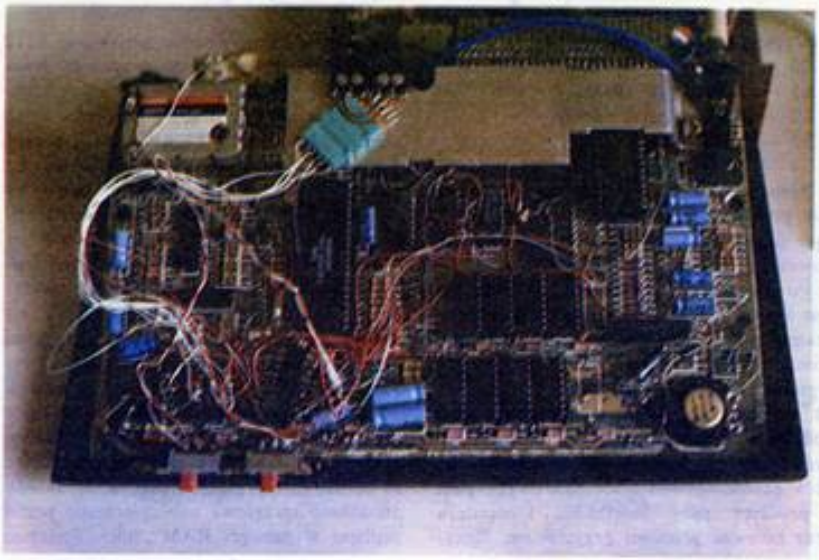
A teraz kilka słów o montażu układu. Założeniem przeróbek było zmieszczenie wszystkiego we wnętrzu normalnej obudowy. Posiadacze wersji drugiej komputera mogą umieścić płytkę uniwersalną w miejscu, gdzie inne wersje mają radiator. Posiadacze wersji trzeciej i czwartej mogą rozwiązać to montażem „na pająka".

Do układu potrzebne są następujące elementy:

74LS02 — jest polski!
74LS32 — trudny do zdobycia
8205 — jest polski odpowiednik UCY74S405
8212 — lub UCY74S412
EPROM 2716 lub 2732, 2764

W miejsce dwóch bramek NAND potrzebnych do tworzenia sygnału α można wykorzystać połowę 74LS00 istniejącego w komputerze. Można, ale tylko w wersji 2 i 3. W wersji 4 bramki te zostały wykorzystane, jako opóźnienie sygnału MUX sterującego multiplekserami pamięci VRAM. Posiadacze tej wersji mogą dołożyć jeden układ scalony lub, co jest prostsze, przerobić swoją wersję na wersję 3, tzn. zastąpić dwie bramki rezystorem 130 ohm, który wraz z pojemnością wejść stanowi takie samo opóźnienie. Teraz przecinając kilka ścieżek uwolnimy te dwie bramki. Kłopotliwe połączenie pod układem IS 24 wykonane od strony elementów można usunąć nawiercając wiertłem 0,8 mm otwór koło nóżki 12 układu IS 24.

Montaż można rozpocząć od uruchomienia portu wyjściowego. Jedynym miejscem dla tej dużej kostki jest nałożenie jej na kostki 4116 pod kątem, tak aby końcówki zasilania można było przylutować do szyn doprowadzających 5V i 0V do pamięci. Pozostałe nóżki można powyginać do góry. Szyna danych procesora, do której należy podłączyć ten port, dostępna jest na opornikach R1-R8. Dla tranzystora zamykającego ULA najlepszym miejscem będzie mikroprocesor. Można położyć go płasko, a jego emiter podłączyć do nóżki 11 IC 2 czyli do Vcc. Pozostałe kostki można nałożyć na układy scalone, wluto-



wane już w układzie, wyginając przy tym ich nóżki do góry. Nóżki zasilające mogą służyć, jako konstrukcja wsporczar. Wszystkie połączenia należy wykonywać przewodami wlutowując je w przeloty istniejące na płytce. Ścieżkę A15 trzeba odciąć od ULA i RAM przy nóżce 20 procesora oraz koło gniazda magnetofonowego (MIC). Następnie należy go zmostkować tak, aby sygnał A15 procesora dostępny był na złączu krawędziowym.

W wersjach 3 i 4 zainstalowanie pamięci EPROM jest dość kłopotliwe. Optymalnym dla niej miejscem jest pamięć ROM. Podstawkę pamięci EPROM można nalutować na ROM mimo tego, że ma ona tylko 24 nóżki. Wynika to stąd, że rozmieszczenie interesujących nas linii adresowych, a także linii danych i masy jest identyczne w obu elementach. Zgadza się ułożenie końcówek A0-A11, D0-D7 i GND, pozostałe tzn. OE, CS i Vcc należy odgiąć i doprowadzić odpowiednie sygnały przewodami. Kłopotliwe może być jedynie zdobycie cienkiej podstawki pod EPROM, tak aby kostka ta zmieściła się pod radiatorem.

Należy zwrócić uwagę na zasilanie mikrokomputera. Po podłączeniu układów prąd pobierany z zasilacza wzrasta o około 200 mA. Należy więc tak ustawić napięcie za transformatorem, aby na wejściu komputera nie było więcej niż 9,5 V. Fabryczne zasilacze często dają pod obciążeniem napięcie około 12—14 V, co często jest przyczyną uszkodzeń komputera. Wyższe napięcie zasilania może spowodować „zatkanie" przetwornicy (chwilowe zatrzymanie pracy) i brak napięcia — 5V, co niszczy pamięci 4116.

OD AUTORA

Pisząc powyższy artykuł miałem na celu zachęcenie posiadaczy mikrokomputera ZX Spectrum do samodzielnego przerabiania własnego sprzętu. Pomysł przeróbek powstał w sierpniu 1985 roku, a prototyp w listopadzie. Tak długi czas i liczne dyskusje nad układem pozwoliły uniknąć wielu kłopotów z uruchomieniem. Opis ten, w pełni oddając tok mojego rozumowania, może innym ułatwić dokonanie identycznych lub tylko częściowych przeróbek swoich mikrokomputerów. Tych, którzy zdecydują się na wszystkie proponowane przeze mnie przeróbki zapraszam do wspólnego oprogramowania. Dostrzegłem cztery dziedziny, którymi warto się zainteresować:

1) Zmiany interpretera ZX Spectrum — to dla tych, którzy lubią rozszerzania BASIC-a i dobrze znają ROM;

2) Przenoszenie oprogramowania z innych komputerów (MERITUM, ARMSTAD, LASER, COBRA);

3) Adaptacja systemu CP/M 2.2, CP/M+ oraz ISIS ze stacją dysków $5\,^1/_4$ cala;

4) Stworzenie nowego systemu operacyjnego dopasowanego do sprzętu i służącego do badań nad systemami operacyjnymi.

Osobiście interesują mnie badania systemowe i zamierzam zająć się dwoma ostatnimi kierunkami pozostawiając pozostałe innym. W celu koordynacji działań wszystkich przerabiających proponuję utworzenie sekcji przerabiaczy przy klubie mikrokomputerowym ENIAC. Od zainteresowanych oczekuję zgłoszeń pod adresem:

Klub Mikrokomputerowy
ENIAC
sekcja sprzętowa
DS „ONDRASZEK"
Gliwice,
ul. Kujawska 2, pok. 427
lub pod adresem redakcji.

Przyjemnie jest mi w tym miejscu złożyć podziękowania wszystkim moim kolegom, którzy swoimi radami przyczynili się do powstania zarówno prototypu, jak i tego artykułu.

```
 10 Switc:      EQU  %00111110
 20 ERRNR:      EQU  23610
 30 NMIRE:      EQU  #5CB0
 40 CHARS:      EQU  23606
 50 NEW:        EQU  #11B7
 60 CR:         EQU  13
 70
 80             ORG  0
 90
100 Reset:
110
120             ORG  23755+5  ;w REM
130
140 Boost:      DI
150             LD   HL,Start
160             LD   DE,#8000
170             LD   BC,DlugB
180             LDIR
190             JP   #8000
200
210 Start:      DI
220             LD   A,%0001  ;ROM on
230             OUT  (Switc),A
240             LD   HL,0
250             LD   DE,#4000
260             LD   B,D
270             LD   C,E
280             LDIR
290             LD   A,%0011  ;ROM off
300             OUT  (Switc),A
310             DEC  DE
320             DEC  HL
330             EX   DE,HL
340             LD   BC,#4000
350             LDDR
360
370 ; petla zmian interpretera
380
390             LD   HL,TabZm-Start+#8000
400 Loop:       LD   C,(HL)
410             INC  HL
420             LD   B,(HL)
430             LD   A,B
440             CP   C
450             JR   Z,Exit
460             INC  HL
470             LD   E,(HL)
480             INC  HL
490             LD   D,(HL)
500             INC  HL
510             LDIR
520             JR   Loop
530 Exit:       LD   A,%1011  ;blokada on
540             OUT  (Switc),A
550             JP   Reset
560
570 ;*********************************
580 ;
590 ; Tablica zmian interpretera
600 ;
610 ; organizacja
620 ;       defw dlugosc bloku1
630 ;       defw adres bloku1
640 ;       defb bajty do wymiany
650 ;       ....
660 ;       defb bajty do wymiany
670 ;       defw 0  ;straznik
680 ;
690 ;*********************************
700
710
720 TabZm:
730             DEFW D1Zm1
740             DEFW #1539
750 x1:         EQU  #1539-$
760             DEFM " Przerwanie"
770             DEFB CR
780             DEFM "   niemaskowane"
790             DEFB "!"+128
800 D1Zm1:      EQU  $+x1-#1539
810
820             DEFW D1Zm2
830             DEFW #1295
840 x2:         EQU  #1295-$
850             CALL InNMI
860             XOR  A
870 D1Zm2:      EQU  $+x2-#1295
880
890             DEFW D1Zm3
900             DEFW #06D
910 x3:         EQU  #06D-$
920             DEFB #28
930 D1Zm3:      EQU  $+x3-#06D
940
950             DEFW D1Zm4
960             DEFW #04AA
970 x4:         EQU  #04AA-$
980 InNMI:      EQU  $+x4
990             LD   HL,ObNMI
1000            LD   (NMIRE),HL
1010            LD   HL,NEW
1020            LD   A,%00011 ;WRenable
1030            OUT  (Switc),A
1040            LD   (5+1),HL
1050            LD   A,%01011 ;WRdisable
1060            OUT  (Switc),A
1070            LD   DE,NowyC
1080            RET
1090 D1Zm4:     EQU  $+x4-#04AA
1100
1110            DEFW D1Zm5
1120            DEFW #386E
1130 x5:        EQU  #386E-$
1140 ObNMI:     EQU  $+x5
1150            LD   IY,ERRNR
1160            LD   (IY),28
1170            CALL #16C5
1180            EI
1190            JP   #1303
1200 ObCal:     EQU  $+x5
1210            CALL #1E99
1220            LD   H,B
1230            LD   L,C
1240            CALL #162C
1250            LD   IY,ERRNR
1260            EI
1270            RET
1280 ObPus:     EQU  $+x5
1290            LD   HL,#8000
1300            LD   DE,#4000
1310 obpus:     LD   BC,#4000
1320            DI
1330            LD   A,%1111
1340            OUT  (Switc),A
1350            LDIR
1360            LD   A,%1011
1370            OUT  (Switc),A
1380            EI
1390            RET
1400 ObPop:     EQU  $+x5
1410            LD   HL,#4000
1420            LD   DE,#8000
1430            JR   obpop
1440 NowyC:     EQU  $+x5
1450            DEFB " "+128
1460            DEFM "© 1985 ENIAC'owe"
1470            DEFM " Spectrum"
1480            DEFB CR
1490            DEFM "  Wersja 1.1 "
1500            DEFM "  1986-01-0"
1510            DEFB "5"+128
1520 ObMon:     EQU  $+x5
1530            RET
1540 D1Zm5:     EQU  $+x5-#386E
1550
1560            DEFW D1Zm6
1570            DEFW #11F
1580 x6:        EQU  #11F-$
1590            DEFB "C","a","l","l"+128
1600            DEFM "Monito"
1610            DEFB "r"+128
1620            DEFB "P","u","e","h"+128
1630            DEFB "P","o","p"+128
1640 D1Zm6:     EQU  $+x6-#11F
1650
1660            DEFW 3
1670            DEFW #1B14
1680            DEFB 3
1690            DEFW ObCal
1700
1710            DEFW 3
1720            DEFW #1B06
1730            DEFB 0
1740            DEFW ObMon
1750
1760            DEFW 3
1770            DEFW #1B0A
1780            DEFB 0
1790            DEFW ObPus
1800
1810            DEFW 3
1820            DEFW #1B10
1830            DEFB 0
1840            DEFW ObPop
1850
1860            DEFW 0
1870 DlugB:     EQU  $-Start
```