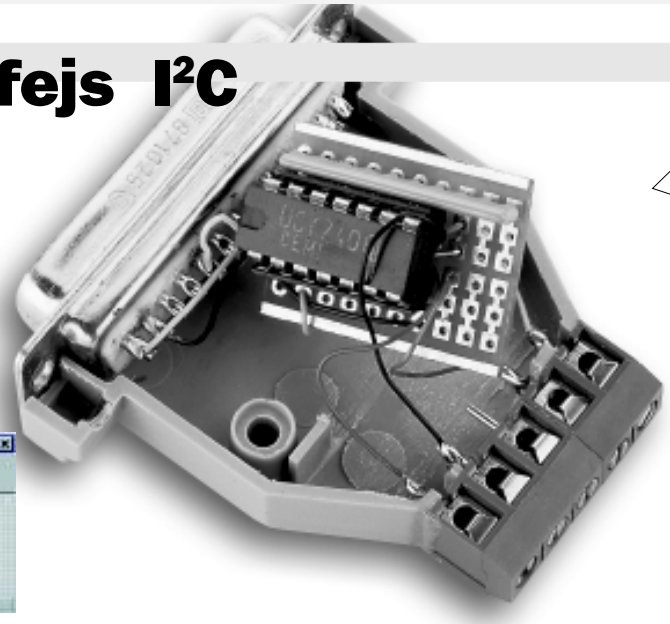
Dział "Projekty Czytelników" zawiera opisy projektów nadesłanych do redakcji EP przez Czytelników. Redakcja nie bierze odpowiedzialności za prawidłowe działanie opisywanych układów, gdyż nie testujemy ich laboratoryjnie, chociaż sprawdzamy poprawność konstrukcji.

Prosimy o nadsyłanie własnych projektów z modelami (do zwrotu). Do artykułu należy dołączyć podpisane ***oświadczenie, że artykuł jest własnym opracowaniem autora i nie był dotychczas nigdzie publikowany****. Honorarium za publikację w tym dziale wynosi 250,- zł (brutto) za 1 stronę w EP. Przysyłanych tekstów nie zwracamy. Redakcja zastrzega sobie prawo do dokonywania skrótów.*

Prosty interfejs I²C



Nowa stawka 250zł/str.

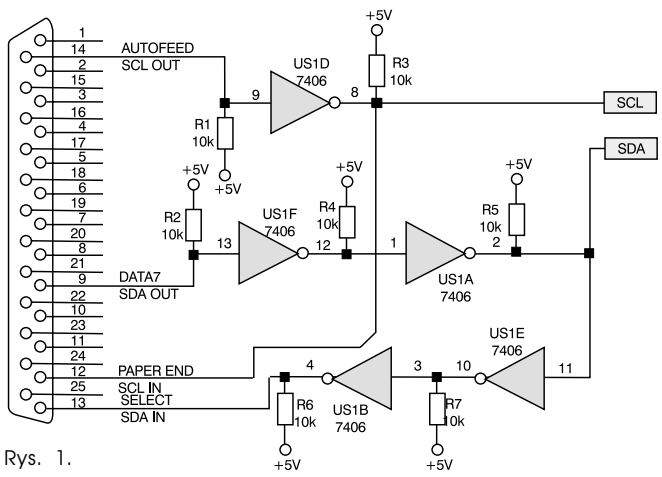
Swego czasu opracowałem bardzo prosty interfejs magistrali I²C do komputera klasy PC. Korzystając z łamów EP chciałbym przedstawić Czytelnikom projekt tego urządzenia.

Projekt 056

Opis ogólny

Interfejs składa się z dwu części. Pierwszą, sprzętową, jest układ elektroniczny. Jest on tak prosty, że w zasadzie zasługuje na miano miniukładu. Zbudowany jest tylko z dwóch elementów, nie licząc oczywiście wtyczki, płytki drukowanej i przewodów. Są to: układ negatora z otwartym kolektorem 74LS06 oraz drabinka rezystorowa 8x10kΩ. Schemat elektryczny układu przedstawiono na **rys. 1**.

Drugą częścią interfejsu jest oprogramowanie sterujące interfejsem. Ze względu na prostotę części sprzętowej, główny ciężar obsługi magistrali I²C spada właśnie na oprogramowanie. Zostało ono przygotowane w języku C++, w nie najnowszej już niestety jego wersji dla Windows 3.11.



Rys. 1.

Opis urządzenia i oprogramowania

W projekcie interfejsu wykorzystano równoległy port komputera. Układ interfejsu korzysta z rejestru danych, rejestru wyjściowego (sterującego) i rejestru wejściowego portu. Wykorzystanie linii poszczególnych rejestrów pokazano w **tab. 1**.

Takie wykorzystanie rejestrów portu umożliwia korzystanie również z komputerów, które nie są wyposażone w dwukierunkowy port drukarkowy.

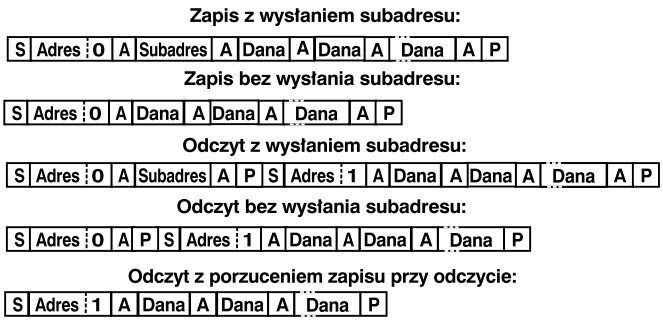
W zasadzie część sprzętowa interfejsu mógłaby zostać pominięta, jednakże buforuje ona port komputera, zabezpieczając go tym samym przed ewentualnym uszkodzeniem oraz powoduje sprzętową negację sygnału SCL OUT.

Program sterujący powstał jako program uniwersalny, nie przeznaczony do jakichś konkretnych zastosowań. Może on wraz z interfejsem służyć między innymi jako tester magistrali I²C (po odłączeniu procesora sterującego od tej magistrali), tester układów wyposażonych w interfejs I²C, jako kopiarka zawartości pamięci, programator pamięci EEPROM itp., a więc w zastosowaniach raczej serwisowych, np. sprzętu RTV. W odróżnieniu od oprogramowania typowych układów zbudowanych z wykorzystaniem mikrokomputerów jednoukładowych (najczęściej pochodnych 8051), program ten umożliwia na przykład przechowywanie przez bardzo długi czas wyników odczytów pamięci lub innych układów w postaci zbiorów na dysku, ich łatwą modyfikację czy ponowny zapis. Jest więc na pewno bardziej elastyczny, jeśli chodzi o udoskonalenia i modyfikacje.

Ponieważ jest to program w przeważającej części (bez samej obsługi magistrali I²C

Tab. 1.

SCL OUT	AUTOFEED	Control LPT (037Ah)	bit 1
SCL IN	PAPER END	In LPT (0379h)	bit 5
SDA OUT	DATA 7	Out LPT (0378h)	bit 7
SDA IN	SELECT	In LPT (0379h)	bit 4



Rys. 2.

napisanej w asemblerze) napisany w języku wysokiego poziomu (Borland C++ ver.3.1 dla Windows), to jest on względnie wolny. Prędkość przesyłania danych może wynosić do kilku tysięcy bitów na sekundę. Zależy ona również od typu komputera. Nie ma to jednak większego znaczenia z dwóch względów:

- po pierwsze, pojemność pamięci układów dołączanych poprzez magistralę I²C wynosi do 2kB (może być oczywiście większa, jednakże program obsługuje na razie pamięci o takiej pojemności), co powoduje, że czas transmisji wynosi co najwyżej kilka sekund;
- po drugie, to program generuje przebieg zegarowy, pracując w trybie MASTER.

Ponieważ czasy opóźnienia pomiędzy zboczami impulsów SDA i SCL oraz oczekiwania na ACK są ustalane przez pętle programowe, to czas ich wykonania zależy od typu komputera, o czym wspomniano powyżej. Dla określenia tych czasów w programie przyjęto odpowiednio 5 i 50 pętli. Są to wartości zapewniające poprawne działanie programu z komputerem AMD486 DX-4 100MHz. Gdyby czasy te okazały się za krótkie, należy za pomocą dowolnego edytora tekstowego utworzyć zbiór o nazwie *i2c.cfg* i wpisać do niego odpowiednie opóźnienia w następujący sposób (przykład dla 10 i 100 pętli):

```
10
100
sa
ssa
bc
sh
```

Akceptowane są liczby z przedziału 0..65535. Przypuszczalnie operacja taka jednak nie będzie konieczna, gdyż nie zauważyłem specjalnych różnic w czasie działania programu „pod“ Windowsami 3.11 na komputerze z procesorem Pentium 100MHz, 8MB RAM, zaś jako ciekawostkę mogę dodać, że program wyraźnie „zwolnił“, pracując z procesorem Pentium 133MHz, 16MB RAM „pod“ systemem Windows NT.

Symbole umieszczone poniżej stałych określających czasy opóźnień odpowiadają kolejnym parametrom zapamiętywanym w zbiorze, odpowiednio:

sa - *slave address*: adres urządzenia podporządkowanego
ssa - *slave subaddress*: subadres komórki;
bc - *byte count*: liczba bajtów przesyłanych;
sh - *show*: widoczne klawisze, jeśli 1.

Dane te są zapamiętywane, jeśli tylko istnieje na dysku zbiór *i2c.cfg* (nie jest on tworzony automatycznie). Wszystkie dane powinny być zapisane dziesiętnie. Program obsługuje standardową transmisję poprzez złącze I²C: 7- bitowy adres urządzenia (możliwość zaadresowania do 127 układów) i 11-bitowy subadres, który może być wysyłany lub nie (możliwość adresowania 2048 komórek układu). Jak wspomniano powyżej, pracuje on w trybie MASTER, to znaczy zawsze rozpoczyna i kończy transmisję oraz generuje sygnał zegarowy. Formaty transmisji realizowane przez program przedstawia **rys. 2**, zaś sposób przesłania pojedynczego bajtu przypomniano na **rys. 3**.

Ekranowy interfejs graficzny użytkownika jest typowy dla Windows - program jest sterowany za pomocą menu oraz posiada dodatkowo tzw. szybkie klawisze, odpowiadające poszczególnym opcjom menu. A oto bliższy opis tych opcji, a co za tym idzie możliwości programu.

Ustawianie parametrów

Ustawienia parametrów transmisji dokonujemy w okienku Parametry. W okienku tym można ustawić:

Port komunikacyjny

Port drukarkowy - LPT1 lub LPT2. Port ten przy starcie programu jest wykrywany automatycznie poprzez odczyt odpowiedniego pola adresowego BIOS-u. Jeśli odczytany w ten sposób adres portu wynosi 0x378, to port ten zostaje nazwany portem LPT1, jeśli zaś odczytanym adresem jest 0x278, port zostaje nazwany LPT2.

Parametry transmisji

Adres - oznacza adres urządzenia (układu) na magistrali I²C, do którego (lub z którego) odczytujemy dane. Do wyboru mamy 88 standardowych adresów. Po ręcznym wpisaniu adresu możemy wybrać adres dowolny. Pole to przyjmuje wartości dziesiętnie, chyba, że jako pierwsze znaki wpiszemy „0x“ (wówczas stosuje się zapis szesnastkowy).

Obok pola wybory adresu znajduje się klawisz *Układy*. Jego naciśnięcie powoduje wyświetlenie listy układów, posiadających wybrany w okienku adres. Możliwe jest zidentyfikowanie po adresie 242 najbardziej popularnych układów.

Subadres - przy przesyłaniu danych do układu określa on adres komórki, od której będą zapisywane dane. W tym przypadku subadres określa również adres komórki bufora, od którego dane będą z niego odczytywane.

Przy odczytywaniu danych z układu określa on adres komórki, od której dane będą odczytywane z układu, a także adres komórki bufora, od którego odczytywane dane będą do niego wpisywane. Podobnie jak pole *Adres*, pole to przyjmuje wartości dziesiętnie lub szesnastkowe. Ponieważ przesłanie subadresu większego od 255 w pewien sposób zakłóca przesłanie adresu (część subadresu staje się wtedy częścią bajtu adresowego) i może być w razie pomyłki przyczyną błędu transmisji (program pyta o potwierdzenie przesyłania subadresu większego od 255).

Prześlij - dotyczy subadresu. Umożliwia przesyłanie lub nie (nie wszystkie układy wymagają takiego przesłania) subadresu do układu.

Ilość bajtów - określa wielkość przesyłanego bloku. Podobnie jak pola Adres i Subadres przyjmuje wartości dziesiętnie lub szesnastkowe poprzedzone znakami „0x“. Dane nie zawierające się w przedziale <subadres - subadres+liczba bajtów> nie ulegają zmianie, tak w buforze, jak i w układzie.

Porzuć zapis przy odczycie - normalnie program realizuje format transmisji właściwy odczytowi z układów posiadających wiele komórek wewnętrznych (jak np. pamięci) z autoinkrementacją adresu. Format ten wymaga przy odczycie układu przesłania najpierw adresu z ustawionym bitem kierunku (zapis), a następnie subadresu itd. Tak więc najmłodszy bit określa tutaj kierunek transmisji i nie może być używany jako bit adresowy. Taka sytuacja występuje dla większości układów z interfejsem I²C - wszystkich układów posiadających parzyste adresy. Istnieją jednakże układy, które można tylko odczytywać, jak np. SAA4700, SAF1135 i które na domiar złego posiadają nieparzyste adresy, czyli najmniej znaczący bit bajtu adresowego jest rzeczywiście najmniej znaczącym bitem adresu. Zatem próba zaadresowania takiego układu do zapisu, czyli wyzerowanie najmniej znaczącego bitu adresu spowoduje brak reakcji układu (bo to nie on będzie adresowany!!!). Opcja ta umożliwia zatem zrezygnowanie z wysyłania jako pierwszego bitu adresowego z wyzerowanym bitem kierunku. Zatem przy dodatkowym wyłączeniu przesyłania subadresu program realizuje transmisję jak na ostatniej pozycji rys. 2.

Zapis typu EEPROM - ze względu na długi czas zapi-

su informacji, pamięci EEPROM nie mają możliwości przyjmowania w jednym cyklu transmisji przy zapisie dowolnej liczby bajtów, jak ma to miejsce np. dla pamięci RAM. Liczba bajtów akceptowanych w jednym cyklu zapisu waha się pomiędzy 1 (np. SDA2516) a, w pewnych warunkach, 8 (np. ST24C02A). Próby przesyłania większej liczby bajtów powodują albo brak potwierdzenia odbioru danych przez pamięć i brak zapisu albo, co gorsza, błędny zapis. Również jest wymagany pewien czas na zapisanie przesyłanego bloku danych przed przesłaniem kolejnego. Może on wynosić dla niektórych pamięci aż 40ms.

Zatem, aby zapisać poprawnie informacje w pamięci EEPROM, wybranie tej opcji umożliwia przesyłanie w jednym cyklu transmisji jednego bajtu pod wskazany subadres, odczekanie 30ms, zwiększenie subadresu, kolejne przesłanie jednego bajtu itd., aż do końca całego bloku. Zatem zapisanie 2kB informacji w tym trybie może trwać aż 61,5 s!!!

Test złącza (systemu)

Test złącza (systemu) polega na adresowaniu po kolei wszystkich ewentualnych urządzeń (układów), począwszy od posiadającego adres 0x02 aż do adresu 0xfe i oczekiwaniu na ewentualne potwierdzenie transmisji. Jeśli jakikolwiek układ wyśle takowe potwierdzenie, jego adres zostaje zapamiętany, a następnie wyświetlony po zakończeniu testu. Jeśli w teście nie zostanie wykryty żaden układ, program wyświetla stosowny komunikat. Podczas testu nie są wpisywane do testowanych układów żadne dane.

Jeśli test był zakończony sukcesem (to znaczy zidentyfikowano jakikolwiek adres układu na magistrali), w okienku pojawia się adres (adresy) urządzenia, które odpowiedziały na zaadresowanie. Jeśli klikniemy dwukrotnie myszką na interesujący nas adres, program wyświetli listę układów, które taki adres mogą posiadać.

Test złącza trwa około jednej sekundy (na komputerze AMD486 DX4/100MHz).

Zapis do urządzenia

Zapis do urządzenia polega na przesłaniu za pomocą magistrali I²C zawartości bufora, poczynając od adresu początkowego, określonego w polu *Subadres* w okienku *Parametry*, do urządzenia (układu) o adresie ustalonym w polu *Adres* w okienku *Parametry*. Przesyłanych jest zawsze tyle bajtów, ile określono w polu *Ilość bajtów* w okienku *Parametry*. Dane w układzie są umieszczane począwszy od adresu określonego w polu *Subadres*.

Jeśli w trakcie zapisu nastąpi jakikolwiek błąd, jak również zapis zakończy się sukcesem, fakt ten jest sygnalizowany odpowiednim komunikatem.

Odczyt z urządzenia

Odczyt z urządzenia polega na przesłaniu magistralą I²C zawartości układu o adresie ustalonym w polu *Adres* w okienku *Parametry*, od adresu początkowego określonego w polu *Subadres*, do bufora od początkowego adresu umieszczonego również w polu *Subadres*. Przesyłanych (odczytywanych) jest tyle bajtów, ile określa pole *Ilość bajtów*. Jeśli w trakcie odczytu nastąpi jakikolwiek błąd, fakt ten sygnalizowany jest odpowiednim komunikatem. Jeśli odczyt zakończy się sukcesem, pojawi się okienko z nową, odczytaną zawartością bufora. Jak wspomniano powyżej, efekty transmisji układ - komputer mogą być przechowywane na dysku komputera, jak również zbiory dyskowe mogą być przesyłane do układu. Umożliwiają to kolejne opcje menu.

Odczyt ze zbioru

Odczyt ze zbioru polega na wpisaniu do bufora całej zawartości zbioru (jego nazwę wprowadza się za pomocą okienka służącego do tego celu), o ile tylko jego rozmiar nie jest większy od rozmiaru bufora (2048 bajtów). Cała poprzednia zawartość bufora jest przy tym kasowana.

Jeśli rozmiar zbioru jest większy od rozmiaru bufora, wczytywanie nie odbywa się, a fakt ten jest sygnalizowany odpowiednim komunikatem.

Zapis do zbioru

Zapis do zbioru polega na zapisaniu na dysku (w zbiorze, którego nazwę wprowadza się w wyświetlonym okienku) pewnej liczby bajtów bufora, począwszy od adresu 0x00 w sposób następujący: od adresu 0x00 do adresu określonego w polu *Subadres* w okienku *Parametry* są wpisywane zera, zaś od adresu określonego w polu *Subadres* do adresu *Subadres+Ilość bajtów* są zapisane dane z bufora. Liczba bajtów odpowiada zawartości pola pod tytułem *Ilość bajtów* w okienku *Parametry*.

Program umożliwia również wykonywanie specjalnych operacji na buforze, zawierającym odczytane dane lub dane do przesłania. Są to: porównanie zawartości bufora ze zbiorem, szukanie wartości w buforze, wyświetlenie zawartości bufora, edycja bufora i czyszczenie bufora.

Porównanie zawartości bufora ze zbiorem dyskowym

Porównanie to polega na porównaniu zawartości bufora oraz wybranego z okna dialogowego zbioru dyskowego. Porównanie zachodzi w obszarze adresowym od adresu 0x000 do 0x000 + rozmiar zbioru. Wszystkie rozbieżności pomiędzy buforem a zbiorem są wyświetlane, łącznie z wyszczególnieniem adresu komórki, w której wystąpiła niezgodność. Jeśli takich różnic nie ma, to fakt ten jest sygnalizowany odpowiednim komunikatem.

Szukanie wartości

Szukanie wartości w buforze polega na odnajdywaniu zadanej wartości w zadanym polu adresowym. Ustawienia odpowiednich parametrów operacji dokonujemy w okienku *Szukaj w buforze*. Możliwe do ustawienia są:

Adres początkowy - określa od jakiego adresu w buforze wartość ma być poszukiwana.

Adres końcowy - określa do jakiego adresu wartość ma być poszukiwana.

Wartość szukana - określa poszukiwaną wartość. Wartość ta może być maksymalnie 4-bajtowa.

Wszystkie wyżej wymienione pola przyjmują wpisywaną wartość szesnastkowo. Nie należy więc dopisywać na początku wartości „0x".

Program nie kontroluje poprawności wprowadzanych danych poza przypadkami:

1. Adres początkowy jest większy od końcowego.
2. Adres końcowy jest większy od 2047 (tj. 7ff szesnastkowo).

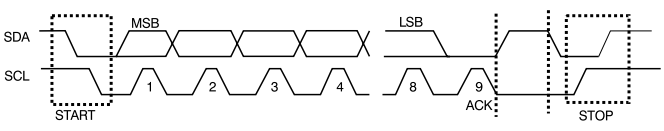
W obu tych przypadkach program przyjmuje ustalone wartości: adres początkowy jako 0x0000, adres końcowy jako 0x07ff. Źle wprowadzona wartość szukana jest traktowana jako zero.

Znalezienie lub nie poszukiwanej danej jest odpowiednio sygnalizowane. Zgłaszane jest zawsze pierwsze pojawienie się danej, licząc od adresu początkowego.

Wyświetlenie zawartości bufora

Zawartość bufora jest wyświetlona w specjalnym okienku. Nie ma możliwości zmiany zawartości bufora przy pomocy okienka wyświetlania zawartości bufora.

Pod okienkiem zawartości bufora jest wyświetlany rozmiar bufora, który standardowo wynosi 2048 bajtów oraz wielkość ostatnio wczytanego zbioru. W danej



Rys. 3. Obsługa odbioru danych przez SLAVE'a

chwili jest wyświetlanych 256 bajtów bufora, zgrupowanych w szesnaście linii. Zawartość bufora możemy przeglądać klawiszami *Up*, *Down* lub korzystając z paska przewijania.

Edycja zawartości bufora

Edycji zawartości bufora dokonujemy w przeznaczonym do tego celu okienku. Po jego wywołaniu nieaktywne stają się klawisze (i odpowiadające im opcje menu) odczytu z układu do bufora, odczytu zbioru do bufora, wyświetlania zawartości bufora, ponownej jego edycji oraz czyszczenia bufora. Sposób wyświetlania bufora przy edycji jest typowy - tak jak dla edytorów binarnych. Typowa jest również obsługa okienka. Możemy poruszać się jedynie w polu ograniczonym danymi, przy czy za pomocą klawisza TAB możemy przeskakiwać pomiędzy danymi szesnastkowymi, a danymi ASCII. Podobnie jak przy wyświetlaniu zawartości bufora, w danej chwili mamy podgląd 256 bajtów bufora. Bufor możemy przeglądać klawiszami Left, Right, Up, Down, Home, End oraz PgDn i PgUp lub za pomocą paska przewijania. Możliwe jest również stosowanie myszy.

Wszystkie zmiany dokonywane w okienku edycji bufora są od razu uwzględniane, tak że nie musimy zapamiętywać nowej zawartości bufora. Edycję kończymy naciśnięciem przycisku *OK*.

Czyszczenie zawartości bufora

Czyszczenie zawartości bufora polega na wpisaniu zer do wszystkich komórek bufora. Ponieważ zawartość bufora jest przy tym tracona bezpowrotnie, program prosi o potwierdzenie operacji.

Wyświetl układy

Opcja ta powoduje wyświetlenie biblioteki wszystkich rozpoznawanych przez program układów z interfejsem I²C. Układy te są wyświetlane w postaci uporządkowanej alfabetycznie listy. Jeśli klikniemy dwukrotnie na wybranym z listy układzie, w okienku z prawej strony listy ukażą się wszystkie adresy, które układ może posiadać na magistrali. Możliwe jest również szukanie układu według nazwy wpisanej w odpowiednim okienku.

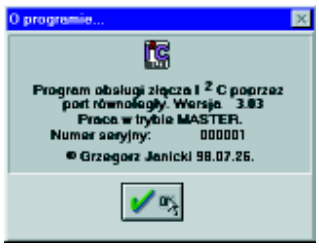
Program zawiera również pomoc, której treść niewiele odbiega od powyższego tekstu.

Montaż i uruchomienie

Jak już wspomniano powyżej, niezwykła prostota układu spowodowała, że został on zbudowany na kawałku płytki uniwersalnej. Kolejność i sposób montażu jest tutaj zupełnie dowolny. Zależy on oczywiście od rozmiarów płytki i przewidzianego sposobu rozmieszczenia elementów. Ponieważ układ nie zawiera żadnych elementów regulacyjnych, trudno jest mówić o jakimkolwiek uruchamianiu go. Zmontowany ze sprawdzonych elementów musi działać od razu. Całą obsługę interfejsu I²C zapewnia oprogramowanie.

Ponieważ płytka z układem zajmuje bardzo mało miejsca, zdecydowałem się umieścić ją wewnątrz obudowy DB25. W tym celu wyciąłem niepotrzebne przegródki z wnętrza wtyczki, zaś płytkę umieściłem w środku. Ponieważ jest ona lekka oraz nie ma zbyt dużej swobody ruchu, uznałem, że „umocowanie" jej na samych przewodach połączeniowych w zupełności wystarczy. Szyny sygnałowe I²C oraz zasilanie są wyprowadzone na złącze śrubowe ARK i przyklejone do wtyczki klejem Poxipol.

Działanie programu zostało przetestowane w pełni jedynie dla układów pamięci z interfejsem I²C, jednakże już to pozwalało stwierdzić jego elastyczność. Na



przykład, pamięć PCF8583 pracowała poprawnie zarówno z pełnym formatem transmisji (rys. 2, pozycja 1,3), jak i z pozostałymi formatami, przy czym, jeśli porzucano w jakikolwiek sposób przesyłanie subadresu, transfer danych zachodził od komórki o adresie 0.

Ewentualnych zainteresowanych tym programem, jak również jego rozbudową czy przystosowaniem do „bardziej konkretnych" aplikacji, proszę o kontakt za pośrednictwem Redakcji. Program jest dostarczony na dyskietce w wersji instalacyjnej, w postaci trzech zbiorów: *install.exe*, *i2c.scr* oraz *i2c.ex$*. Zbiory te najlepiej przekopiować na dysk twardy i w okienku programu instalacyjnego wpisać nazwę katalogu, do którego docelowo ma być przeniesiony program. Po zainstalowaniu we wskazanym katalogu znajdą się zbiory: *i2c.exe* - program obsługi złącza, *i2c.hlp* - pomoc do tego programu oraz *bwcc.dll* - biblioteka niezbędna do działania programu.

Program został skompilowany z opcjami, wymagającymi do jego działania przynajmniej procesora 80386 z koprocesorem arytmetycznym lub 486DX.

Grzegorz Janicki

Bibliografia

[1] Elektronika Praktyczna 9/93.
[2] Elektronika Praktyczna 10/94.
[3] Elektronika Praktyczna 3/98.
[4] Serwis Elektroniki 2/96.
[5] Serwis Elektroniki 3/96.
[6] Serwis Elektroniki 6/96.
[7] Pomoc do kompilatora Borland C w.3.1 dla Windows.