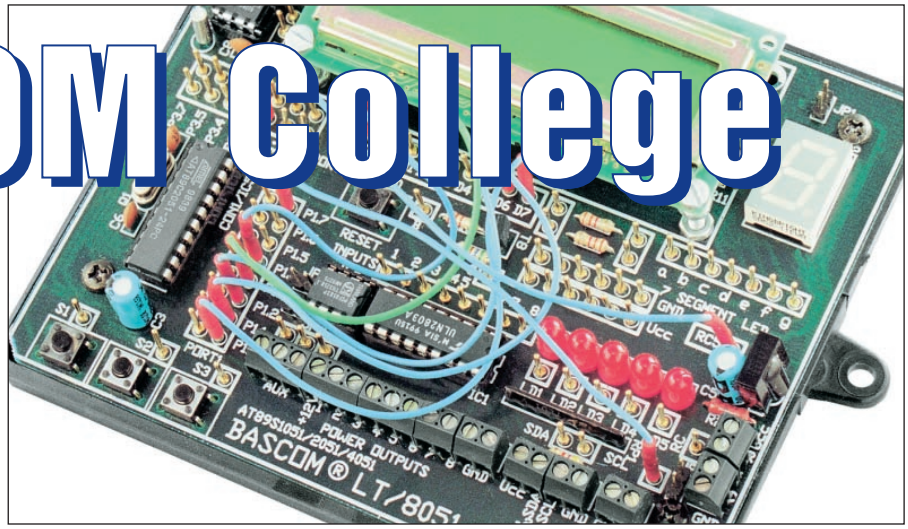
BASCOM College

Ćwiczenie 8



Uwaga!
Następne spotkanie w ramach BASCOM College będzie już spotkaniem dwunastym, ostatnim w ramach regularnego kursu. W styczniowym numerze oprócz ostatniego wykładu przedstawimy pytania sprawdzające. Osoby, które wykażą się znajomością BASCOM-a otrzymają zapowiadane dyplomy. Zakończenie kursu nie oznacza jednak, że BASCOM znika z łamów EdW. Fakultatywne zajęcia będą trwać nadal, a ich podstawą będą mniej lub bardziej regularne artykuły redakcyjne oraz projekty nadsyłane przez Czytelników.

Programowanie szeregowych pamięci EEPROM

W tym miesiącu spotykamy się znowu, tym razem w laboratorium. Do przerobienia mamy cała masę materiału, który możemy podzielić na dwie części. Pierwsza z nich to temat "na żądanie", sygnalizowany już na wykładzie w ubiegłym miesiącu. W ekspresowym tempie zbudujemy sobie aż trzy urządzenia służące kompleksowej obsłudze szeregowych pamięci EEPROM. Abstrahując od wartości użytkowej urządzeń, które wykonamy, temat jest ciekawy sam w sobie. Zauważmy bowiem, że znaleźliśmy całkiem nowe zastosowanie dla naszego BASCOM-a, zastosowanie którego nie przewidział nawet jego Twórca. Do naszego pakietu dodamy bowiem nową funkcję: będzie on pracował jako samodzielny programator pamięci EEPROM i to bez wyposażania go w jakikolwiek dodatkowy hardware. Do wszelkich operacji na pamięciach EEPROM wystarczy nam bowiem nasza płytka testowa, no może po maleńkiej i odwracalnej przeróbce.

Wykonamy trzy ćwiczenia, będące jednocześnie konstruowaniem pełnosprawnych i użytecznych programatorów EEPROM:

1. "Ręczny" programator umożliwiający zapisanie dowolnej wartości pod dowolny, wskazany adres pamięci.

2. Programator automatyczny, zapełniający wskazany obszar pamięci danymi umieszczonymi w tabeli.

3. Kopiarkę pamięci EEPROM umożliwiającą przeniesienie zwartości jednej pamięci do drugiej.

Po zakończeniu "ujarzmiania" pamięci EEPROM przejdziemy do drugiej części ćwiczeń, chyba znacznie ciekawszej od pierwszej. Naszym zadaniem będzie zbudowanie dwóch urządzeń testowych: cyfrowego termometru z układem DS1820 i "rdzenia" programu obsługi najprostszego zamka - blokady do urządzeń elektronicznych z tabletkami DS1990. Jednak tym razem nie obejdzie się bez pewnej rozbudowy posiadanego przez nas hardware'u. Podam Wam dwie konfiguracje sprzętowe: minimalną i maksymalną, które umożliwią Wam przeprowadzenie ćwiczeń:

1. Konfiguracja maksymalna:
- jedna lub kilka tabletek DALLAS DS1990
- czytnik TOUCH MEMORY
- jeden lub kilka układów DS1820

2. Konfiguracja minimalna:
- jeden układ DS1820

Wszystkie potrzebne elementy znajdują się w ofercie handlowej AVT, a także są dostępne w większości sklepów z częściami elektronicznymi. Wybór zestawu części do przeprowadzenia ćwiczeń zależy tylko od Waszych planów na przyszłość. Dysponując zaledwie jednym układem DS1820 będziemy mogli wykonać wszystkie dzisiejsze ćwiczenia z części drugiej, ale praca nie będzie należeć do najwygodniejszych. Jeżeli zatem mamy zamiar w przyszłości wykonać np. zamek elektroniczny, którego otwarcie "sposobem" będzie absolutnie niemożliwe, to radziłbym zakupić czytnik TOUCH MEMORY i co najmniej jedna tabletkę DALLAS DS1990. Po wykonaniu ćwiczeń elementy te posłużą Wam bowiem do budowy konkretnego układu użytkowego.

Zacznijmy od pierwszej grupy ćwiczeń, związanej z obsługą pamięci EEPROM. Przygotowałem dla Was trzy proste programy, które możemy wykorzystać do ćwiczeń, a także w dowolny sposób modernizować, dostosowując je do własnych potrzeb. Wszystkie programy mogą zostać wykorzystane bez konieczności programowania procesora, wyłącznie w środowisku emulatora sprzętowego BASCOM-a 8051 połączonego płytką testową AVT2500. A zatem, zaczynamy!

"Ręczne" zapisywanie danych w pamięci EEPROM, pod wskazanym adresem

Pierwszy program, którego treść pokazana jest na **listingu 1** jest bardzo podobny do programu, który napisaliśmy podczas przerabiania ćwiczenia z obsługi magistrali I^2C. Umożliwia on zapisanie dowolnej wartości (oczywiście z zakresu jednego bajta, czyli 0 ... 255) pod dowolny adres szeregowej pamięci EEPROM. Działanie tego programu zostało szczegółowo opisane w komentarzach. Warto jedynie zwrócić uwagę, że każdy zapis poddawany jest natychmiastowej weryfikacji. Daje to nam absolutną pewność, że pamięć została zapisana zgodnie z naszymi intencjami.

Dane, które mamy zapisać w pamięci EEPROM w celu ich późniejszego wykorzystania przez program obsługujący procesor nie pochodzą zwykle z "naszej głowy", ale są wynikiem mniej lub bardziej skomplikowanych obliczeń matematycznych, wykonywanych

najczęściej za pomocą arkuszy kalkulacyjnych. Mozolne ręczne "wklepywanie" 255 liczb do pamięci byłoby nie tylko czynnością żmudną, ale z całą pewnością prowadzącą do powstania nieuchronnych pomyłek. Najwyższy zatem czas, aby praktycznie wykorzystać wiadomości zdobyte na dzisiejszym wykładzie i dane przeznaczone do zapisania w pamięci EEPROM umieścić wstępnie w tabeli, z której następnie będą odczytywane za pomocą instrukcji READ i umieszczane w pamięci.

```
'Listing 1

$sim                        'praca w symulacji
Config Sda = P3.5           'konfiguracja magistrali I²C
Config Scl = P3.7           'konfiguracja magistrali I²C

Declare Sub Read_eeprom(adres As Byte, Test As Byte)
                            'deklaracja podprogramu
    odczytu pojedynczej komórki pamięci EEPROM
Declare Sub Write_eeprom(adres As Byte, Value As Byte)
                            'deklaracja podprogramu
 zapisu do  pojedynczej komórki pamięci EEPROM
 Dim Adres As Byte, Value As Byte
                            'deklaracja zmiennej
określającej adres w pamięci i zapisywaną wartość
Dim Test As Byte            'deklaracja zmiennej pomocniczej

Do
Input "Podaj adres w pamięci EEPROM  ", Adres
'zapytanie o adres, pod który ma być zapisana informacja
Input "Podaj wartość danej do zapisania  ", Value
'zapytanie o wartość, jak  ma być umieszczona
                            w podanej komórce pamięci
Call Write_eeprom Adres, Value
'wezwanie podprogramu zapisania wskazanej
                    komórki pamięci podaną wartością
Call Read_eeprom Adres, Test      'kontrolny odczyt
                            zapisanej uprzednio wartości

  If Value = Test Then 'jeżeli wynik porównania
   wartości podanej z odczytaną jest pomyślny, to:
    Print "OK"                   'wyświetl na
                        ekranie terminala napis "OK."
  End If                        'koniec warunku
Loop
'Poniższe podprogramy zostały już opisane podczas
przerabiania ćwiczeń z obsługi magistrali I²C.
Sub Read_eeprom(adres As Byte, Value As Byte)
  I2cstart
  I2cwbyte 160
  I2cwbyte Adres
  I2cstart
  I2cwbyte 161
  I2crbyte Test, 9
  I2cstop
End Sub

Sub Write_eeprom(adres As Byte, Value As Byte)
   I2cstart
   I2cwbyte 160
   I2cwbyte Adres
   I2cwbyte Value
   I2cstop
   Waitms 10
   End Sub
```

Ulokowanie danych w tabeli jest szczególnie łatwe w przypadku korzystania z arkusza kalkulacyjnego. Obliczone wartości możemy bowiem zapisać w "czystym" pliku tekstowym i stamtąd przenieść je przez clipboard do tekstu programu pisanego w MCS BASIC.

Analizy sposobu działania drugiego programu, który po uruchomieniu automatycznie przepisze dane z tabeli do pamięci EEPROM umieszczonej w podstawce na płytce testowej AVT2500, nie musimy chyba szczegółowo omawiać. Podobnie jak w pierwszym programie, dane zapisywane w pamięci poddawane są weryfikacji. Tylko że w procesie automatycznego zapisu nic z tego nie wynika. Po prostu program nie wyświetli na ekranie terminala napisu potwierdzającego poprawny zapis i będzie pracował dalej, jakby nic się nie stało. Jeżeli zatem program ten ma Wam służyć do czegoś więcej, niż do demonstracji możliwości pakietu BASCOM8051, to warto go uzupełnić o bardziej rozbudowaną procedurę zabezpieczającą przed zapisaniem do pamięci fałszywych danych.

```
'Listing 2
$sim
Config Sda = P3.5
Config Scl = P3.7

Declare Sub Read_eeprom(adres As Byte, Test As Byte)
Declare Sub Write_eeprom(adres As Byte, Value As Byte)
Dim Adres As Byte, Value As Byte
Dim Test As Byte
Dim R As Byte
Restore Randomdata
For R = 0 To 255
Read Value
Call Write_eeprom R, Value
Call Read_eeprom R, Test
If Value = Test Then
Print Value; "  "; Test; "  OK"
End If
Next R
End

Sub Read_eeprom(adres As Byte, Value As Byte)
  I2cstart
  I2cwbyte 160
  I2cwbyte Adres
  I2cstart
  I2cwbyte 161
  I2crbyte Test, 9
  I2cstop
End Sub

Sub Write_eeprom(adres As Byte, Value As Byte)
   I2cstart
   I2cwbyte 160
   I2cwbyte Adres
   I2cwbyte Value
   I2cstop
   Waitms 10
   End Sub

Randomdata:
Data 128 , 124 , 152 , 247 , 72 , 240 , 147 , 153 ,
231 , 80 , 40 , 98 , 220 ,
Data 22 , 31 , 25 , 115 , 241 , 39 , 18 , 247 , 38 ,
127 , 95 , 121 , 85 , 234,
Data 150 , 217 , 8 , 5 , 101 , 58 , 242 , 192 , 148 ,
99 , 93 , 135 , 54 , 216 ,
Data 201 , 75 , 16 , 82 , 221 , 137 , 251 , 47 , 118 , 64 , 154,
Data 41 , 115 , 208 , 234 , 201 , 241 , 105 , 212 ,
136 , 113 , 199 , 165 , 135 ,
Data 22 , 20 , 51 , 24 , 210 , 58 , 97 , 39 , 42 , 254
, 196 , 20 , 93 , 111 , 5,
Data 216 , 65 , 141 , 237 , 32 , 79 , 212 , 241 , 14 ,
134 , 223 , 158 , 53 ,
Data 163 , 118 , 60 , 53 , 21 , 190 , 140 , 111 , 224
, 198 , 58 , 199 , 222 ,
Data 211 , 40 , 148 , 102 , 95 , 37 , 173 , 187 , 121
, 134 , 120 , 114 , 207 ,
Data 195 , 241 , 71 , 126 , 14 , 246 , 41 , 178 , 224
, 35 , 197 , 14 , 118 ,
Data 74 , 21 , 18 , 76 , 112 , 165 , 196 , 103 , 127 ,
166 , 28 , 42 , 61 , 62 ,
Data 39 , 132 , 68 , 101 , 84 , 246 , 131 , 181 , 163
, 0 , 8 , 16 , 46 , 146 ,
Data 105 , 58 , 138 , 139 , 157 , 216 , 227 , 89 , 158
, 187 , 81 , 160 , 42 ,
Data 157 , 55 , 150 , 16 , 253 , 42 , 83 , 59 , 219 ,
123 , 241 , 24 , 114 , 119
Data 93 , 157 , 242 , 103 , 26 , 177 , 92 , 115 , 166
, 29 , 91 , 86 , 82 , 201,
Data 21 , 10 , 110 , 59 , 126 , 115 , 70 , 201 , 39 ,
197 , 24 , 129,
Data 201 , 39 , 197 , 24 , 129 , 46 , 47 , 215 , 191 ,
240 , 166 , 204 , 243,
Data 117 , 32 , 5 , 81 , 224 , 9 , 4 , 176 , 170 , 101
, 96 , 117 , 86 , 153,
Data 50 , 25 , 110 , 226 , 247 , 101 , 86 , 146 , 218
, 227 , 72 , 172 , 238 ,
Data 1 , 100 , 174 , 194
```

Kolejnym problemem, na jaki możemy napotkać podczas wykorzystywania w naszej pracy pamięci szeregowych EEPROM, jest konieczność zbadania zawartości pamięci zaprogramowanej w wyniku działania jakiegoś programu lub dostarczonej przez producenta i skopiowanie jej zawartości. Program realizujący czynność kopiowania danych z jednej pamięci do drugiej jest banalnie prosty, ale w pierwszej chwili możemy natknąć się na problemy sprzętowe. Na naszej płytce testowej jest miejsce na jedną pamięć szeregową EEPROM, a co z drugą pamięcią do której lub z której będziemy kopiować dane? Na szczęście nie ma sytuacji bez wyjścia: popatrzmy na schemat naszej płytki AVT2500. Mamy tam miejsce na jeszcze jeden układ scalony w obudowie 8-pinowej, na zegar czasu rzeczywistego typu PCF8583. Jeżeli wnikliwie przyjrzymy się schematowi, to zauważymy z pewnością, że rozmieszczenie pinów SDA i SCL jest w tym układzie identyczne jak w typowej pamięci szeregowej EEPROM. Różnica polega na tym, że w układzie zegara mamy dostępne tylko jedno wejście adresowe: A0 na pinie 3, a piny 1 i 2 wykorzystane są do dołączenia do układu rezonatora kwarcowego. Na płytce pin A0 jest na stałe dołączony do plusa zasilania. Co z tego wynika? A no to, że bez najmniejszych przeszkód możemy umieścić drugiego EEPROM-a w podstawce przeznaczonej na zegar czasu rzeczywistego, a piny 1 i 2 po prostu zewrzeć do masy za pomocą cienkiego drucika włożonego w podstawkę razem z układem scalonym. Dołączony do podstawki kwarc i kondensator nie będą w niczym przeszkadzać, w żadnym przypadku nie grozi im też uszkodzenie. Konieczne przeróbki pokazane są na **rysunku 1**, a po ich wykonaniu uzyskamy możliwość zainstalowania na naszej płytce drugiej pamięci EEPROM, o adresach 162 i 163.

Na **listingu 3** pokazana jest treść programu, którego zadaniem jest przekopiowanie zawartości pamięci umieszczonej w podstawce IC2 do pamięci, która zagarnęła podstawkę przeznaczoną zwykle na zegar RTC. Program jest tak prosty, że możemy pozostawić go bez komentarza, a wymagający Czytelnicy mogą jedynie rozbudować go o procedury sprawdzania poprawności kopiowania.

Do pierwszego programu umożliwiającego "ręczne" programowanie pamięci trudno mieć jakiekolwiek zastrzeżenia. Jego bardzo wolna praca, spowodowana opóźnieniami wnoszonymi przez emulator sprzętowy, nie powinna przy tym trybie pracy sprawić najmniejszych kłopotów. Natomiast zarówno drugi, jak i trzeci program będą pracować w emulacji sprzętowej rozpaczliwie wolno. Przeprowadziłem stosowne próby i z procesorem PENTIUM III 600MHz zainstalowanym w komputerze, czas kopiowania pamięci wyniósł nieco ponad 1,5 minuty! Jeżeli zatem będziemy mieli zamiar kopiować większą ilość pamięci, to może warto by było zaprogramować procesor i użyć go do obsługi naszej kopiarki.

Powyższe przykłady nie wyczerpują, rzecz jasna wszystkich możliwości, jakie dają nam operacje wykonywane na pamięciach EEPROM bezpośrednio z poziomu BASCOM'a. Sama możliwość odczytania zawartości pamięci może oddać w wielu przypadkach nieocenione usługi. Za chwilę zajmiemy się układami 1WIRE i będziemy odczytywać numery seryjne "magicznych" tabletek DALLAS, a następnie zapisywać je i odczytywać z pamięci EEPROM. Podczas wykonywania tych czynności łatwo o wystąpienie błędu w oprogramowaniu. Mając możliwość natychmiastowego odczytania zawartości pamięci możemy stwierdzić, czy przynajmniej proces zapisu danych przebiegał prawidłowo.

1WIRE

No nareszcie, zaczynamy pracę z jedną z najbardziej interesujących rodzin układów scalonych, jakie kiedykolwiek zostały skonstruowane. Stosowanie tych układów zawsze budziło, częściowo tylko uzasadnione lęki wśród początkujących elektroników, uważających "Dallasy" za przejaw działalności sił nieczystych i czarnej magii. Za chwilę przekonamy się, jak bardzo ta opinia była niesłuszna i że będziemy w stanie "dobrać się" do wszystkich możliwości oferowanych przez układy 1WIRE za pomocą tylko trzech, prostych w obsłudze poleceń języka MCS BASIC.

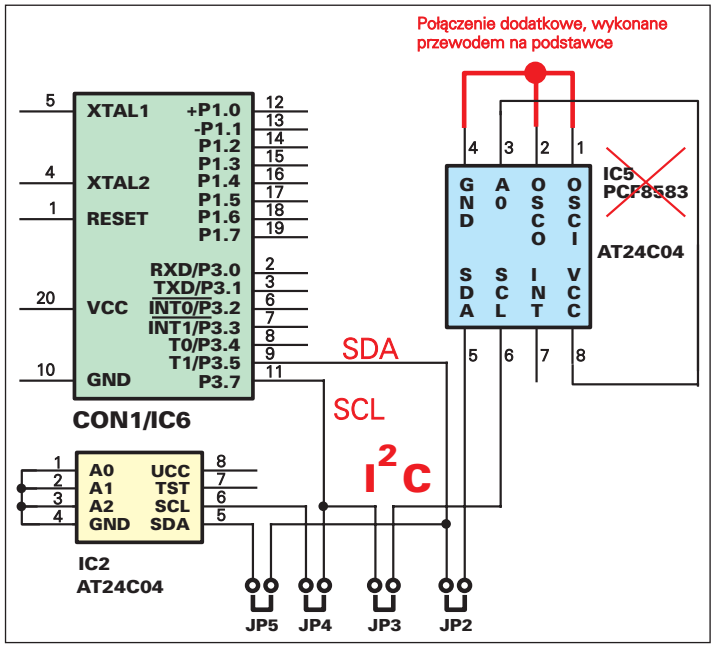
Do wykonania ćwiczeń potrzebne będą elementy wymienione w pierwszej części artykułu oraz procesor typu 89CX051. Niestety, żadna z operacji związanych z obsługą magistrali 1WIRE nie jest możliwa do przeprowadzenia w emulacji sprzętowej i wszystkie ćwiczenia będziemy musieli wykonać dysponując zaprogramowanym uprzednio procesorem. Czytnik 1WIRE TOUCH MEMORY dołączmy do naszej płytki testowej tak, jak zostało pokazane na **rysunku 4**. Jeżeli nie posiadamy takiego czytnika, to możemy zastąpić go prowizorycznie dwoma odcinkami przewodu, do których będziemy dołączać tabletki DS1990 lub termometru DS1820. Informacje o pracy programu kierowane będą na wyświetlacz alfanumeryczny LCD.

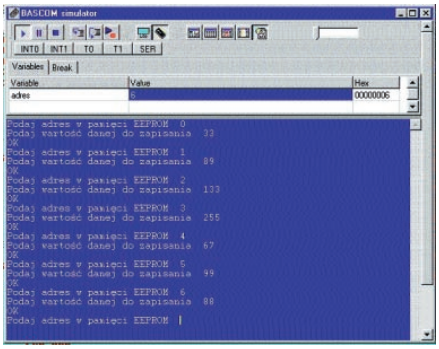
Do końca dzisiejszych zajęć pozostało już niewiele czasu i dlatego zdążymy przerobić tylko jedno ćwiczenie, ale za to mające zasadnicze znaczenie dla opanowania techniki posługiwania się układami 1WIRE. Nauczymy się odczytywać numery seryjne tych układów, co w najbliższej przyszłości umożliwi nam skonstruowanie kilku ciekawych urządzeń.

Na tym etapie nauki jest całkowicie obojętne, jakimi układami 1WIRE aktualnie dysponujemy. Może to być równie dobrze tabletka DS1990 jak i cyfrowy termometr DS1820. Każdy bowiem z tych układów posiada swój indywidualny i niepowtarzalny numer seryjny, który odczytywany jest w identyczny sposób. Wynika z tego jedno, ciekawe spostrzeżenie: każdy układ wyprodukowany przez firmę DALLAS i pracujący z magistralą 1WIRE może być elektronicznym kluczem, niezależnie od pełnionych przez niego zasadniczej funkcji. Oczywiście, stosowanie w tej roli układów DS1990 jest najwygodniejsze, głównie ze względu na ich obudowę, ułatwiającą dołączanie ich do czytników.

Przygotowałem dla Was krótki program, który może posłużyć jako baza do dalszych

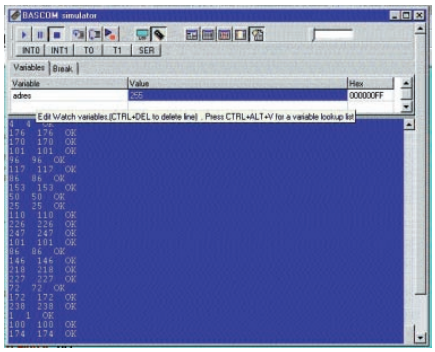
Rys. 1



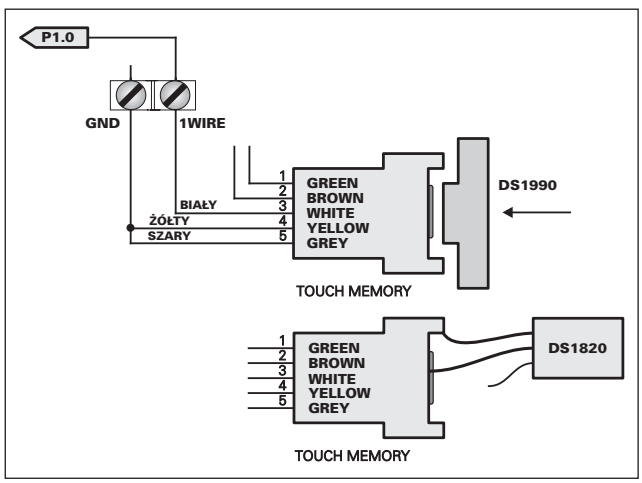


Rys. 2

Rys. 3



Rys. 4




```
'Listing 3
$sim
Config Sda = P3.5
Config Scl = P3.7

Declare Sub Read_eeprom(adres As Byte, Test As Byte)
Declare Sub Write_eeprom(adres As Byte, Value As Byte)
Dim Adres As Byte, Value As Byte
Dim Test As Byte
Dim R As Byte

For R = 0 To 255
Call Read_eeprom R, Value: Print Value
Call Write_eeprom R, Value
Next R
End


Sub Read_eeprom(adres As Byte, Value As Byte)
  I2cstart
  I2cwbyte 160
  I2cwbyte Adres
  I2cstart
  I2cwbyte 161
  I2crbyte Value, 9
  I2cstop
End Sub

Sub Write_eeprom(adres As Byte, Value As Byte)
   I2cstart
   I2cwbyte 162
   I2cwbyte Adres
   I2cwbyte Value
   I2cstop
   Waitms 10
End Sub
```

doświadczeń z układami 1WIRE, a także jako podstawa do napisanie konkretnych programów użytkowych. Chciałbym, abyście przemyśleli dokładnie jego treść i zastanowili się, jakie bardzo, ale to bardzo ciekawe urządzenie można zbudować wykorzystując jego składniki i oczywiście, dodając nowe. Zadaniem tego programu jest odczytanie z przyłożonego do czytnika układu 1WIRE ośmiu bajtów jego numeru seryjnego. Tak naprawdę, to numer seryjny składa się tylko z siedmiu bajtów (w tym jeden bajt identyfikujący grupę układów) i jednego bajtu zawierającego sumę kontrolną, umożliwiającą zweryfikowanie poprawności odczytu. Dla ułatwienia możemy jednak nazywać odczytaną z układu liczbę 64 bitową numerem seryjnym. Po odczytaniu numeru program zapisuje go w pamięci EEPROM, weryfikuje ten zapis i wyświetla kolejne bajty numeru na ekranie wyświetlacza LCD. Zastanówcie się teraz, jakie urządzenie możemy zbudować dodając jeszcze kilka - kilkanaście linijek programu? Urządzenie to może być wyjątkowo spektakularnym pokazem możliwości stosowania układów 1WIRE, głównie tabletek DS1990.

Zbigniew Raabe
e-mail: **zbigniew.raabe@edw.com.pl**

```
'Listing 4
Config 1wire = P1.0  'wskazanie kompilatorowi,
  do którego pinu dołączona jest magistrala 1WIRE
Config Lcd = 16 * 1a
Dim Number(8) As Byte
Dim R As Byte
Declare Sub Read_numbers
Declare Sub Write_eeprom(r As Byte, Number As
Byte)
Declare Sub Read_eeprom(r As Byte, Number As
Byte)
Cls
Cursor Off

Do                      'początek pętli programowej
1wreset                 'inicjalizacja magistrali 1WIRE
If Err = 1 Then         'jeżeli żaden układ nie
                        odpowiedział na wezwanie, to:
Cls                     'wyczyść wyświetlacz
Lcd "No device!"        'wyświetl stosowny komunikat
Waitms 25                       'zaczekaj 250 ms
Else                    'w przeciwnym wypadku:
Cls
Call Read_numbers   'wezwij podprogram czytania
      numeru seryjnego dołączonego do magistrali
                                        układu 1WIRE
End If                  'koniec warunku

Loop

Sub Read_numbers
1wwrite &H33        'żądanie podania przez
          odnaleziony układ jego numeru seryjnego

For R = 1 To 8                  'ośmiokrotnie:
  Number(r) = 1wread():         'odczytaj z
     magistrali 1WIRE kolejne bajty numeru układu
Next R
Cls
Lcd "Device found!"  'wyświetl komunikat o
                                zgłoszeniu się układu

For R = 1 To 8                  'ośmiokrotnie:
Call Write_eeprom R, Number(r)      'zapisz do pamięci
  EEPROM kolejne bajty numeru  seryjnego układu
Next R

For R = 1 To 8                  'ośmiokrotnie:
Call Read_eeprom R, Number(r)   'dla kontroli
                odczytaj zapisany w pamięci numer
Next R

For R = 1 To 8
'ośmiokrotnie
Cls
'wyczyść ekran wyświetlacza
Waitms 255
'zaczekaj 255ms
Lcd Number(r)
'wyświetl kolejne bajty numeru seryjnego układu
Wait 1
'zaczekaj 1 sekundę
Next R
End Sub


Sub Write_eeprom (eeprom_adres Byte, Value As
Byte)
   I2cstart
   I2cwbyte 160
   I2cwbyte R
   I2cwbyte Number
   I2cstop
   Waitms 10
   End Sub

Sub Read_eeprom (eeprom_adres As Byte, Value
As Byte)
   I2cstart
   I2cwbyte 160
   I2cwbyte R
   I2cstart
   I2cwbyte 161
I2crbyte Number, Nack
End Sub
```