# Moduł Hosta USB, część 2

## AVT–983

*Cieszące się niezmiernie długo popularnością dyskietki 1,44 MB, których czytniki jeszcze do dziś są instalowane w komputerach, chyba powoli będą jednak odchodziły w zapomnienie. Do długotrwałego przechowywania danych wyparły je płytki CDROM i DVD, do chwilowego zapisania danych, np. w celu przeniesienia ich z komputera na komputer, już od dłuższego czasu służą pendrive'y. Wbudowana w nich pamięć półprzewodnikowa o pojemności dochodzącej do kilku gigabajtów nie daje szans dyskietce. Od niedawna, za sprawą układów Vinculum, pamięci pendrive można w bardzo prosty sposób wykorzystywać nawet w najprostszych systemach mikroprocesorowych.*

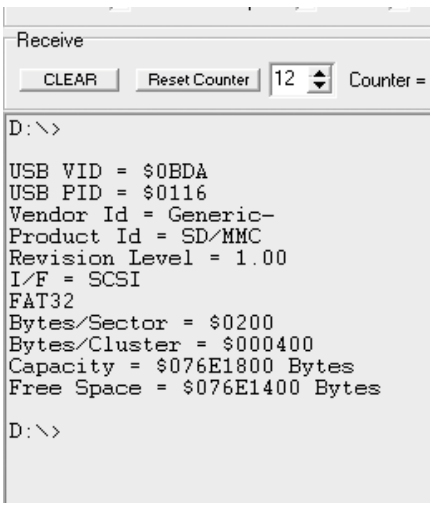
***Rekomendacje:***
*nowe układy rodziny Vinculum z pewnością szybko zdobędą popularność, tak jak stało się to w przypadku układów FT232 i FT245. Chętni ich stosowania nie mogą nie zrobić modułu Hosta USB.*

**PODSTAWOWE PARAMETRY**

- Płytka o wymiarach 56x44 mm
- Zasilanie: +5 V, dostępne napięcie +3,3 V na pinach modułu
- Gniazdo: USB Host typu A
- Interfejsy: UART, SPI, FIFO wybierane dwoma zworkami, drugi interfejs USB dostępny na pinach modułu
- Wskazania o stanie modułu za pomocą diod LED
- Współpraca z pamięciami masowymi z systemem plików FAT
- Komunikacja za pomocą kilkunastu prostych komend przypominających komendy DOS

### Komunikacja z wykorzystaniem komputerowego terminala

Stosując komendy rozszerzone, komunikację modułu Hosta USB z komputerowym terminalem można zrealizować w najprostszy sposób wykorzystując interfejs UART. Do tego celu niezbędny będzie programator układów Vinculum, który jak już wiemy, pełni jedynie funkcję konwertera USB<–>RS232. Zainstalowano dla niego sterowniki wirtualnego portu COM. Za pomocą programatora można podać napięcie zasilania dla modułu Hosta USB. Zgodnie z tab. 1, aby wybrać interfejs UART, zworki JP1 i JP2 mogą być założone lub zdjęte. Do portu USB został dołączony pendrive o pojemności 128 MB. Komunikacja będzie się odbywać za pomocą komend rozszerzonych. Domyślna prędkość transmisji wynosi 9600 bodów, 1 bit startu, 1 bit stopu, brak bitu parzystości z włączoną kontrolą transmisji za pomocą sygnałów RTS/CTS. RTS to linia żądania nadawania, a CTS to linia gotowości do wysłania danych. Aby zrezygnować ze sprzętowej kontroli transmisji, można linie RTS i CTS połączyć ze sobą. Wtedy do transmisji będą wykorzystywane tylko linie TXD i RXD. W terminalu należy ustawić identyczne parametry transmisji. Przy poprawnej komunikacji, po umieszczeniu dysku w złączu USB modułu, w oknie terminala powinien pojawić się znak zachęty *D:\>*. Po wysłaniu rozkazu *IDD* zosta-

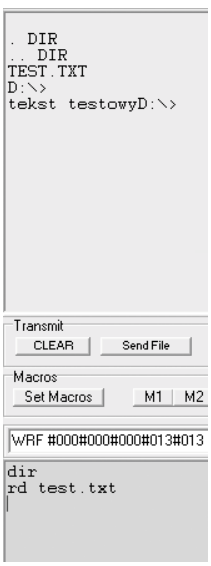



**Rys. 5. Informacje o dysku wyświetlone po wysłaniu rozkazu *IDD***


**Rys. 6. Informacja wyświetlona po założeniu katalogu *TEST***


**Rys. 7. Informacja wyświetlona po założeniu do zapisu pliku *TEKST.TXT***


**Rys. 8. Zapisanie danych do pliku *TEST.TXT* przy użyciu komendy WRF**

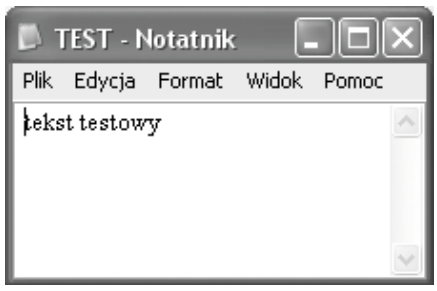



**Rys. 9. Odczytanie zawartości pliku *TEST.TXT* przy użyciu komendy *RD***

ną wyświetlone informacje o dysku (**rys. 5**). Można odczytać z tych informacji pojemność dysku, wielkość sektora oraz typ systemu plików, w tym przypadku jest nim FAT32. W pierwszej kolejność w ramach testu został założony katalog *TEST* (**rys. 6**) z wykorzystaniem komendy *MKD*. Listę katalogów i plików można wyświetlić wysyłając komendę *DIR*. Następnie po wejściu do katalogu *TEST* (komendą *CD*) zostaje założony do zapisu plik *TEKST.TXT* (**rys. 7**). Do założenia pliku otwartego do zapisu wykorzystano komendę *OPW*. Do zapisu danych do pliku *TEST.TXT* wykorzystano komendę *WRF* (**rys. 8**), której jednym z parametrów jest informacja o liczbie zapisywanych bajtów. Składa się ona z 4 bajtów zapisanych szesnastkowo. Po zapisaniu do pliku przykładowego stringu *„tekst testowy”*, plik jest zamykany z wykorzystaniem komendy *CLF*. Do odczytu zawartości pliku *TEST.TXT* wykorzystano komendę *RD* (**rys. 9**). Jak można się przekonać, obsługa pamięci USB jest bardzo prosta. Zapisane pliki można również odczytywać umieszczając dysk USB w komputerze (**rys. 10**). W ten sposób można bez większych problemów odczytać dane za pośrednictwem komputera, po zapisaniu ich przez inne urządzenie.

## Komunikacja z wykorzystaniem interfejsu UART

Na **rys. 11** przedstawiono przykładowy sposób podłączenia mikrokontrolera do interfejsu UART modułu Host USB. Z modułem komunikuje się mikrokontroler AVR ATmega88. Linie interfejsu RS232 zostały pokazane w **tab. 10**. Do komunikacji wykorzystano tylko linie RXD i TXD. Linie kontroli transmisji CTR i RTS zostały ze sobą połączone. Oczywiście jest możliwe, aby tymi liniami sterował mikrokontroler. Do mikrokontrolera został dołączony również wyświetlacz LCD, na którym będą prezentowane informacje z działania programu. Program sterujący mikrokontrolerem został napisany w Bascom AVR i będzie realizował identyczne operacje




**Rys. 10. Odczytanie zawartości pliku *TEST.TXT* po włożeniu dysku USB w komputerze**

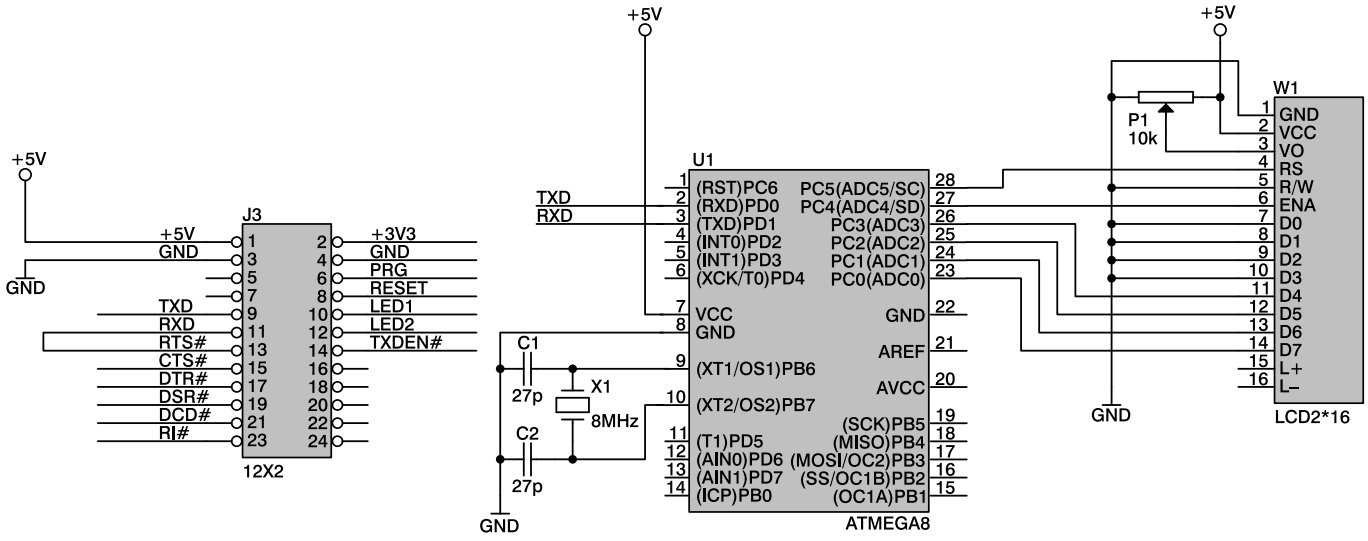
**Tab. 10. Sygnały interfejsu UART**

| Nazwa | Typ | Opis |
|---|---|---|
| TXD | Wyjście | Dane wysyłane |
| RXD | Wejście | Dane odbierane |
| RTS# | Wyjście | Żądanie nadawania |
| CTS# | Wejście | Gotowość do wysyłania danych |
| DTR# | Wyjście | Gotowość do odbierania danych |
| DSR# | Wejście | Odbiornik gotowy do odbioru danych |
| DCD# | Wejście | Odbiór fali nośnej |
| RI# | Wejście | Wskaźnik wywołania |
| TXDEN | Wyjście | Odblokowuje transmisję przez RS485 |

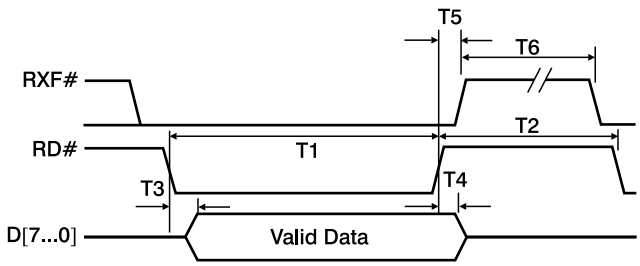
jak w przypadku testowania modułu poprzez terminal. Będzie więc zakładany katalog, a w nim plik, w którym zostaną zapisane dane. Następnie utworzony plik zostanie odczytany i jego zawartość zostanie wyświetlona na wyświetlaczu LCD. Aby moduł pracował z interfejsem UART, zgodnie z tab. 1 należy odpowiednio ustawić zworki JP1 i JP2. Do portu USB został dołączony również pendrive o pojemności 128 MB. Komunikacja będzie się odbywać za pomocą skróconych komend, zalecanych przy współpracy z mikrokontrolerem. Komendy skrócone są prostsze i sto-

**Tab. 11. Sygnały interfejsu równoległego FIFO**

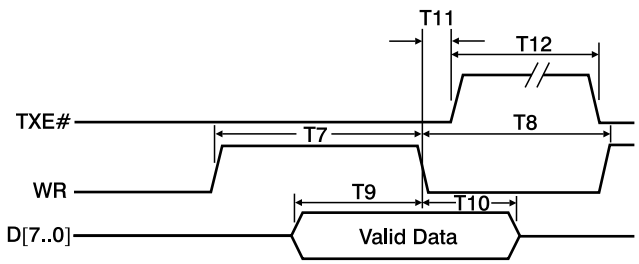
| Nazwa | Typ | Opis |
|---|---|---|
| D0 | We/Wy | Bit 0 magistrali danych D |
| D1 | We/Wy | Bit 1 magistrali danych D |
| D2 | We/Wy | Bit 2 magistrali danych D |
| D3 | We/Wy | Bit 3 magistrali danych D |
| D4 | We/Wy | Bit 4 magistrali danych D |
| D5 | We/Wy | Bit 5 magistrali danych D |
| D6 | We/Wy | Bit 6 magistrali danych D |
| D7 | We/Wy | Bit 7 magistrali danych D |
| RXF# | Wyjście | Kiedy stan wysoki, nie można czytać danych z FIFO. Kiedy stan niski dane są w FIFO. Dane można odczytać strobując linię RD#. |
| TXE# | Wyjście | Kiedy stan wysoki, nie można zapisywać danych do FIFO. Kiedy stan niski można zapisywać dane do FIFO. Dane można zapisywać strobując linię WR. |
| WR# | Wejście | Dane z linii D0...D7 są zapisywane do FIFO kiedy sygnał WR zmienia stan z wysokiego na niski. |
| RD# | Wejście | Dane z FIFO pojawiają się na liniach D0...D7 kiedy sygnał RD# zmienia stan z wysokiego na niski. |




**Rys. 11. Przykładowy sposób dołączenia mikrokontrolera do interfejsu UART modułu Host USB**

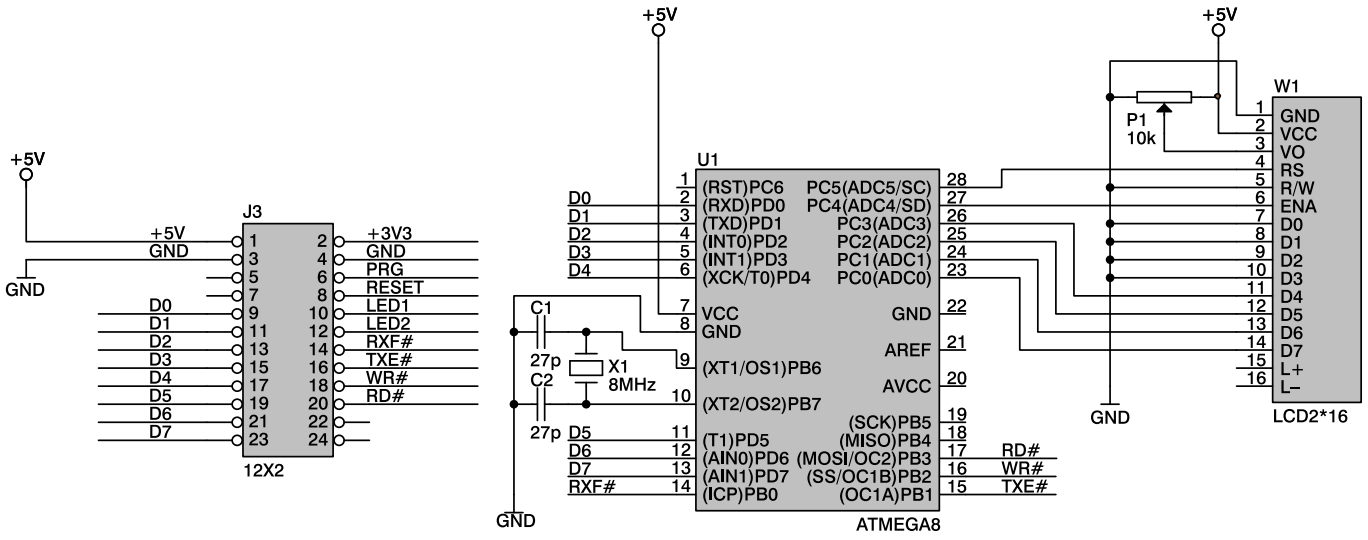



**Rys. 12. Przebiegi występujące podczas odczytu danych z FIFO**




**Rys. 13. Przebiegi występujące podczas zapisu danych do FIFO**

sując je przesyła się mniej danych. Domyślna prędkość transmisji wynosi 9600 bodów, 1 bit startu, 1 bit stopu, brak bitu parzystości, z włączoną kontrolą transmisji danych za pomocą sygnałów RTS/CTS. Również domyślnie wykonywany jest rozszerzony zestaw komend. Na **list. 1** przedstawiono program testowy komunikujący się z modułem Hosta USB. W programie wykorzystano buforową transmisję przez UART, zarówno danych odbieranych, jak i wysyłanych. W pierwszej kolejności oczekuje się na zgłoszenie się dysku USB (oczekiwany jest odbiór znaków *D:\>*). Jeśli dysk zostanie wykryty, wysyłany jest rozszerzony rozkaz *SCS*, który powoduje przełączenie modułu do komunikacji ze skróconymi komendami. Po jego wysłaniu, dysk USB będzie się zgłaszał znakiem „>”. Znak „>” będzie również świadczyć o poprawności wykonania danej komendy. Jego otrzymanie jest sprawdzane po każdej komendzie. Jeśli nie zostanie on otrzymany, zgłaszany jest błąd. W dalszej kolejności komunikacja z dyskiem odbywa się za pomocą instrukcji *Print* i *Input*. Zakładany jest katalog, następnie plik z danymi. Na końcu programu następuje odczyt zawartości pliku i jego wyświetlenie na wyświetla-




**Rys. 14. Schemat dołączenia modułu Host USB do mikrokontrolera z wykorzystaniem interfejsu równoległego FIFO**

czu LCD. Procedura *Odp* sprawdza poprawność wykonania komend. Jak widać założenie katalogu i pliku jest bardzo proste. Nie jest przy tym wymagana duża liczba operacji. Do obsługi dysku wystarczy nawet niewielki mikrokontroler, który nie musi posiadać dużo pamięci i wyprowadzeń.

## Komunikacja z wykorzystaniem interfejsu równoległego FIFO

Komunikacja z wykorzystaniem portu równoległego FIFO znacznie przyspiesza transmisję danych w odniesieniu do interfejsów szeregowych. Interfejs równoległy wymusza jednak stosowanie kilkunastu linii mikrokontrolera. W module Host USB interfejs równoległy FIFO można aktywować, zakładając tylko zworkę JP2. W **tab. 11** pokazano opis linii interfejsu równoległego FIFO. Na **rys. 12** pokazano przebiegi podczas odczytu danych z FIFO. Nieodebrane dane są wskazywane niskim stanem linii RXF. Na **rys. 13** pokazano przebiegi podczas zapisu danych do FIFO. Zapełnienie FIFO danymi jest wskazywane stanem wysokim linii TXE. Interfejs FIFO modułu Host USB działa identycznie, jak interfejs FIFO układu FT245. Na **rys. 14** pokazano przykładowy schemat dołączenia modułu Host USB do mikrokontrolera z wykorzystaniem interfejsu równoległego FIFO.

**Marcin Wiązania, EP**
**marcin.wiazania@ep.com.pl**

**List. 1. Program testowy wykorzystujący interfejs UART do komunikacji z pamięcią USB**

```
'Program wykorzystujacy interfejs UART do komunikacji z pamiecia USB
'z wykorzystaniem ukladu VNCIL (Vinculum)
'Zaklozony zostaje katalog w ktorym nastepnie tworzony jest plik.
'Do pliku zapisywany jest tekst, ktory podczas odczytu wyswietlany jest na LCD
'Program mozna rowniez wykorzystac do komunikacji mikrokontrolera z pamiecia USB
'z wykorzystaniem interfejsu USB Slave jak FT232 lub FT245.
'Marcin Wiązania
'marcin.wiazania@ep.com.pl
$prog &HFF , &HFF , &HDF , &HF9                          'konfiguracja bitow FUSE
$regfile = „m88def.dat"                                  'informuje kompilator o pliku dyrektyw
wykorzystywanego mikrontrolera
$crystal = 8000000                                       'informuje kompilator o czestotliwo-
sci rezonatora kwarcowego
$baud = 9600                                             'informuje kompilator o predkosci
transmisji
$hwstack = 40                                            'konfiguracja wartosci stosow
$swstack = 40                                            'konfiguracja wartosci stosow
$framesize = 40                                          'konfiguracja wartosci stosow
Config Lcd = 16 * 2                                      'konfiguracja organizacji znakow wy-
swietlacza LCD
Config Lcdpin = Pin , Db4 = Portc.3 , Db5 = Portc.2 , Db6 = Portc.1 , Db7 = Portc.0 , E = Portc.4
, Rs = Portc.5       'konfiguracja pinow mikrokontrolara do ktorych dolaczone zostaly linie wy-
swietlacza
Config Serialin = Buffered , Size = 50                   'konfiguracja bufora wejsciowego
Config Serialout = Buffered , Size = 50                  'konfiguracja bufora wejsciowego
Declare Sub Odp                                          'procedura sprawdzania poprawnosci
wykonania komendy
Dim Znaki As String * 40                                 'definicja zmiennej znaki typu string
Wait 2                                                   'opoznienie 2 sekundy
Echo Off                                                 'wylaczenia echa dla instrukcji input
Enable Interrupts                                        'odblokowanie przerwan globalnych
Cursor Off                                               'wylaczenie kursora
Cls                                                      'czysc lcd
Lcd „Brak dysku"                                         'wyswietlenie komunikatu
Do                                                       'poczatek petli
  Input Znaki                                            'odczyt danych z UART
Loop Until Znaki = „D:\>"                                'petla wykonywana az odebrane zostana
znaki D:\>
Print „SCS" ; : Print Chr(&H0d);                         'przelacznie na skrocony zestaw ko-
mend
Call Odp                                                 'sprawdzenie wykonania komendy
Cls                                                      'czysc lcd
Lcd „Dysk aktywny"                                       'wyswietlenie komunikatu
Wait 1                                                   'opoznienie 1 sekundy
Cls                                                      'czysc lcd
Lcd „MKD TEST"                                           'wyswietlenie komunikatu
Print Chr(&H0d);                                         'wyslanie znaku CR w ramach testu
Call Odp                                                 'sprawdzenie wykonania komendy
Print Chr(&H06) ; : Print Chr(&H20) ; : Print „TEST" ; : Print Chr(&H0d);       'zalozenie kata-
logu TEST
Call Odp                                                 'sprawdzenie wykonania komendy
Wait 1                                                   'opoznienie 1 sekundy
Cls                                                      'czysc lcd
Lcd „CD TEST"                                            'wyswietlenie komunikatu
Print Chr(&H02) ; : Print Chr(&H20) ; : Print „TEST" ; : Print Chr(&H0d);       'wejscie do kata-
logu TEST
Call Odp                                                 'sprawdzenie wykonania komendy
Wait 1                                                   'opoznienie 1 sekundy
Cls                                                      'czysc lcd
Lcd „OPW TEST.TXT"                                       'wyswietlenie komunikatu
Print Chr(&H09) ; : Print Chr(&H20) ; : Print „TEST.TXT" ; : Print Chr(&H0d);       'zalozenie
pliku do zapisu TEXT.TXT
Call Odp                                                 'sprawdzenie wykonania komendy
Wait 1                                                   'opoznienie 1 sekundy
Cls                                                      'czysc lcd
Lcd „WRF TEST.TXT"                                       'wyswietlenie komunikatu
Print Chr(&H08) ; : Print Chr(&H20) ; : Print Chr(&H00) ; : Print Chr(&H00);       'zapis do pli-
ku TEXT.TXT
Print Chr(&H00) ; : Print Chr(&H0d) ; : Print Chr(&H0d);     'tekstu majacego 13 znakow o tresci
Print „tekst testowy" ; : Print Chr(&H0d);               'tekst testowy
Call Odp                                                 'sprawdzenie wykonania komendy
Call Odp                                                 'sprawdzenie wykonania komendy
Wait 1                                                   'opoznienie 1 sekundy
Cls                                                      'czysc lcd
Lcd „CLF TEST.TXT"                                       'wyswietlenie komunikatu
Print Chr(&H0a) ; : Print Chr(&H20) ; : Print „TEST.TXT" ; : Print Chr(&H0d);       'zamkniecie
pliku TEXT.TXT do zapisu
Call Odp                                                 'sprawdzenie wykonania komendy
Wait 1                                                   'opoznienie 1 sekundy
Cls                                                      'czysc lcd
Lcd „RD TEST.TXT"                                        'wyswietlenie komunikatu
Print Chr(&H04) ; : Print Chr(&H20) ; : Print „TEST.TXT" ; : Print Chr(&H0d);       'odczyt za-
wartosci pliku TEXT.TXT
Input Znaki                                              'odczyt danych z UART
Lowerline                                                'druga linia LCD
Lcd Left(znaki , 13)                                     'wyswietlenie tesktu odczytanego
z pliku TEXT.TXT
Home                                                     'pierwsze linia LCD
Do                                                       'nieskonczona petla do-loop
Loop                                                     'koniec nieskonczonej petli
End                                                      'koniec programu
Sub Odp                                                  'podprogram sprawdzania poprawnosci
wykonania komendy
  Input Znaki                                            'odczyt danych z UART
  If Znaki <> „>" Then                                   'jesli znak inny niz > to
    Cls                                                  'czysc LCD
    Lcd „Blad komendy"                                   'wyswietlenie komunikatu
    Do                                                   'nieskonczona
    Loop                                                 'petla do-loop
  End If
End Sub                                                  'koniec procedury
```