Obsługa wyświetlaczy graficznych w Bascom, część 1

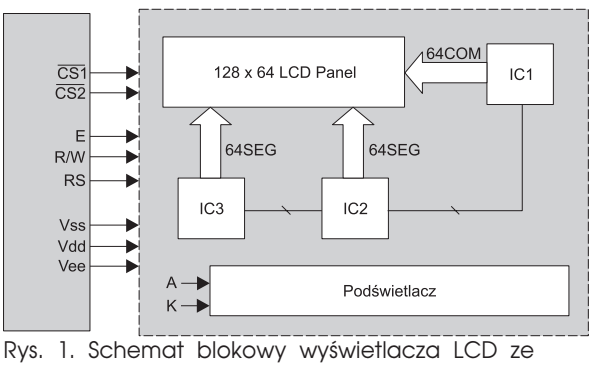
W tym odcinku przedstawimy dwa przykłady obsługi wyświetlaczy graficznych o rozdzielczości 128*64 pikseli, z kontrolerami typu KSx i T6963C. Obsługa wyświetlacza z kontrolerem SED jest identyczna jak wyświetlacza z kontrolerem KSx, więc i z tego typu wyświetlaczami nie powinno być problemów.

Wyświetlacze graficzne są coraz tańsze i z tego względu da się zauważyć ich coraz częstsze wykorzystywanie. Nie są to elementy trudne w obsłudze, w porównaniu z popularnymi wyświetlaczami alfanumerycznymi dają większe możliwości. Na wyświetlaczu graficznym można wyświetlić nie tylko znaki alfanumeryczne dowolną, przygotowaną czcionką, ale także elementy graficzne. Obsługa wyświetlaczy graficznych w Bascom tak, jak wyświetlaczy alfanumerycznych jest dosyć prosta. Bez większych problemów można z tego typu wyświetlaczami budować dosyć ambitne urządzenia. Przykładowo, w przypadku miernika temperatury, można zmierzoną temperaturę w określonym czasie wyświetlać dodatkowo na wykresie. W Bascomie jest możliwość obsługi wyświetlacza z trzema popularnymi sterownikami: KS0107B, KS0108B lub T6963C oraz SED.

Bascom oferuje wiele instrukcji, które ułatwiają obsługę wyświetlaczy graficznych. Dostępne są instrukcje, które umożliwiają zapalenie wybranego piksela, narysowanie linii, okręgu lub wyświetlenie grafiki. Dla wyświetlaczy graficznych nie posiadających własnego generatora znaków, możliwe jest załączenie w programie pliku z fontami, którego znaki można na nich wyświetlać. Dodatkowo, w Bascomie, pracę z wyświetlaczami graficznymi ułatwiają zawarte w nim narzędzia, którymi są: *Graphic Converter* – umożliwia dokonanie konwersji pliku z grafiką na postać akceptowaną przez wyświetlacz oraz *Font Editor* – umożliwia przygotowanie własnego lub modyfikację istniejącego pliku z fontami (znakami), które można wyświetlić na wyświetlaczu graficznym. Przedstawimy dwa przykłady. Pierwszy, będzie dotyczył wyświetlacza graficznego ze sterownikiem KSx w ramach, którego zaprezentujemy działanie instrukcji rysujących linie, koła, wyświetlających znaki z zaimportowanego pliku z fontami oraz wyświetlającymi grafikę (również z prostą animacją). Narysowany zostanie także wykres funkcji trygonometrycznych sin oraz cos. Drugi przykład będzie prezentował aspekty obsługi wyświetlacza z kontrolerem T6963C. W ramach tego przykładu zostanie przedstawiony program realizujący analogowy zegar (ze wskazówkami) z datownikiem. Jak można się spodziewać i jak się okaże, realizacja tego typu zegara w Bascom, może być w miarę prosta. W wyświetlaczach graficznych, których obsługa odbywa się za pomocą Bascoma, może być problem dołączenia ich do mikrokontrolera i ich konfiguracji. Dodatkowo większość wyświetlaczy graficznych potrzebuje ujemnego napięcia dla układu regulacji kontrastu. Zatem przedstawiamy także sposób dołączenia wyświetlaczy do mikrokontrolera, jego konfiguracji oraz wytworzenia ujemnego napięcia dla obwodów regulacji kontrastu.

Wyświetlacze z kontrolerem KSx

Wyświetlacze z kontrolerem KS0107B lub KS0108B mają 8–bitowy interfejs komunikacyjny, 8192 bitową pamięć RAM oraz są zasilane napięciem 5 V. Na **rys. 1** przedstawiono schemat blokowy wyświetlacza JM12864a, który wykorzystano w ramach tego przykładu. Może on zawierać kontroler KS0107B lub KS0108B, ma rozdzielczość 128x64 pikseli oraz podświetlenie za pomocą diod LED. Warto wiedzieć,

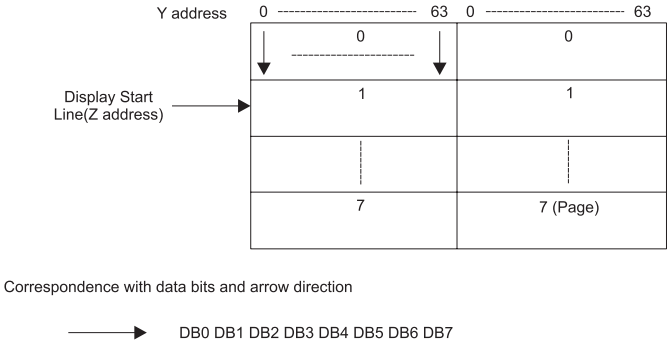


Rys. 1. Schemat blokowy wyświetlacza LCD ze sterownikiem KS0107B

List. 1.

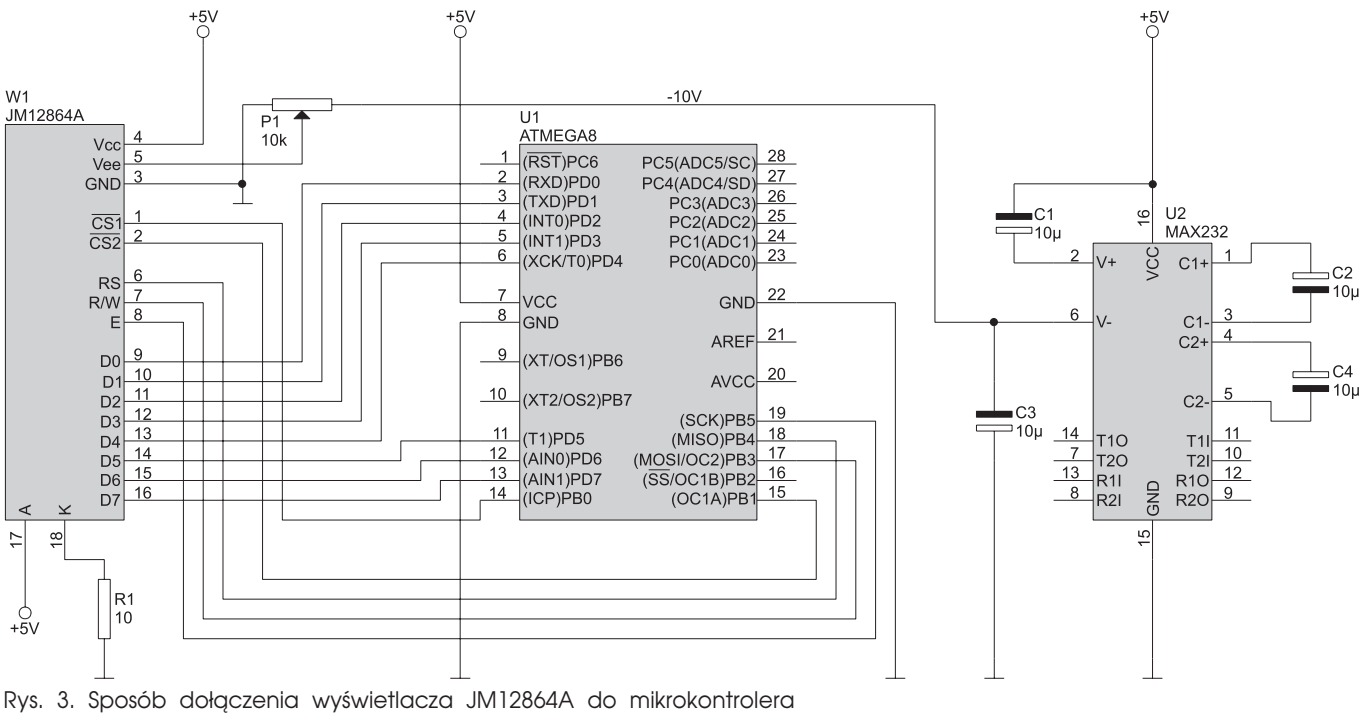
```
'Przyklad obslugi wyswietlacza graficznego JM12864A (128*64) z kontrolerem KS0107B lub KS0108B
'W programie pokazano dzialanie instrukcji przeznaczonych dla wyswietlaczy graficznych jak:
'wyswietlenie tekstu, rysowanie linii, okregu, punktu na podstawie ktorego obliczone i narysowane zostaly wykresy SIN oraz COS
'Poprzez wyswietlenie dwoch roznych bitmap pokazana zostala prosta animacja.
'Marcin Wiazania
'marcin.wiazania@ep.com.pl

$lib „glcdKS108.lib”                                    'zalaczenie biblioteki obslugujacej wyswietlacz z kontrolerem KS
$lib „FP_Trig.lib”                                      'zalaczenie biblioteki funkcji trygonometrycznych

$crystal = 4000000                                      'wartosc czestotliwosci oscylatora taktujacego mikrokontroler
$regfile = „m8def.dat”                                  'wskazanie pliku z definicjami rejestrow dla mikrokontrolera ATmega8

Config Graphlcd = 128 * 64sed , Dataport = Portd , Controlport = Portb , Ce = 0 , Ce2 = 1 , Cd = 4 , Rd = 3 , Reset = 2 , Ena-
ble = 5       'konfiguracja rodzaju oraz wyprowadzen wyswietlacza graficznego

'port danych dolaczony do Portd
'linia CS1 dolaczona do Portb.0
'linia CE2 dolaczona do Portb.1
'linia CD dolaczona do Portb.4
'linia RD dolaczona do Portb.3
'linia RESET brak
'linia ENABLE dolaczona do portb.5

$eeprom                                                 'poczatek zapisu danych do pamieci EEPROM
Rys2:                                                   'etykieta wskazujaca na rysunek 2
$bgf „rysunek2.bgf”                                     'zapis do EEPROM podczas programowania plik rysunek2.bgf
$data                                                   'poczatek pamieci programu mikrokontrolera

Declare Sub Rys_fun(byval R As Byte , Byval Poz As Byte)    'procedura rysujaca funkcje trygonometryczne - par poz wskazuje na
pozycje na LCD a par R na rodzaj funkcji tryg

Dim I As Byte                                           'zmienna licznikowa
Dim S As Word                                           'zmienna licznikowa
Dim W As Single                                         'zmienna pomocnicza przy obliczaniu wartosci funkcji tryg
Dim Y As Integer                                        'zmienna wskazujaca pozycje Y wyswietlacza
Dim X As Word                                           'zmienna wskazujaca pozycje X wyswietlacza

Cls                                                     'czyszczenie LCD
Setfont Font8x8                                         'wybranie rodzaju fontu (czcionki) dla wyswietlacza graficznego
Lcdat 2 , 22 , „Elektronika”                            'wyswietlenie tekstu
Lcdat 3 , 27 , „Praktyczna”                             'wyswietlenie tekstu
Lcdat 5 , 40 , „Bascom” , 1                             'wyswietlenie tekstu
Lcdat 6 , 52 , „AVR” , 1                                'wyswietlenie tekstu
Line(10 , 5) -(118 , 5) , 1                             'wyswietlenie linii - rysowanie prostokatu
Line(118 , 5) -(118 , 51) , 1                           'wyswietlenie linii - rysowanie prostokatu
Line(118 , 51) -(10 , 51) , 1                           'wyswietlenie linii - rysowanie prostokatu
Line(10 , 51) -(10 , 5) , 1                             'wyswietlenie linii - rysowanie prostokatu

For I = 14 To 120 Step 10                               'petla wykonywana az I bedzie wieksze od 120
  Circle(i , 58) , 4 , 1                                'rysowanie kilku okregow
Next I                                                  'zwiekszenie o jeden wartosci I

Wait 2                                                  'czekaj 2 sekundy
Cls                                                     'czysci LCD
Lcdat 1 , 1 , „sin(x)” , 1                              'wyswietlenie nazwy rysowanej funkcji
Call Rys_fun(1 , 19)                                    'rysowanie funkcji sin(x)
Lcdat 5 , 1 , „cos(x)” , 1                              'wyswietlenie nazwy rysowanej funkcji
Call Rys_fun(0 , 51)                                    'rysowanie funkcji con(x)
Wait 2                                                  'czekaj 2 sekundy

Cls                                                     'czyszczenie LCD
Do                                                      'poczatek nieskonczonej petli Do-loop
  Showpic 0 , 0 , Rys1                                  'wyswietlenie rysunku z pamieci programu umieszczonego pod etykie-
ta Rys1
  I = Rnd(256)                                          'w I zapisywana wartosc losowa od 0 do 255
  Waitms I                                              'czekaj czas okreslony wartoscia I
  Waitms 100                                            'czekaj 100 ms
  Showpice 0 , 0 , Rys2                                 'wyswietlenie rysunku z pamieci EEPROM umieszczonego pod etykieta
Rys2
  I = Rnd(256)                                          'w I zapisywana wartosc losowa od 0 do 255
  Waitms I                                              'czekaj czas okreslony wartoscia I
  Waitms 100                                            'czekaj 100 ms
Loop
End                                                     'koniec programu

Sub Rys_fun(byval R As Byte , Byval Poz As Byte)        'poczatek proceduty rysujacej funkcje sin i cos
  For S = 0 To 640 Step 2                               'petla wykonywana az S osiagnie wartosc 640, S zmienia sie co 2
    W = S                                               'przypisanie wartosci S do W
    W = Deg2rad(w)                                      'zamiana stopni zapisanych w W na radiany
    If R = 1 Then                                       'jesli R = 1 to
      W = Sin(w)                                        'obliczenie wartosci funkcji sin
    Else                                                'w przeciwnym razie
      W = Cos(w)                                        'obliczenie wartosci funkcji cos
    End If
    W = W * 9                                           'zwiekszenie o 9 wartosci obliczonej funkcji
    Y = Int(w)                                          'zwrocenie do Y wartosci calkowitej obliczonej funkcji
    Y = Poz - Y                                         'obliczenie pozycji rysowania funkcji
    X = S / 5                                           'obliczenie wartosci wspolrzednej X
    Line(x , Y) -(x , Y) , 1                            'rysowanie obliczonego punktu na LCD
  Next I                                                'zwiekszenie o jeden wartosci I
End Sub                                                 'koniec procedury

$include „font8x8.font”                                 'zalaczenie pliku ze znakami dla LCD
'$include „font16x16.font”                               'zalaczenie innego pliku ze znakami dla LCD

Rys1:                                                   'etykieta wskazujaca na rysunek 1
$bgf „rysunek1.bgf”                                     'zapis do rysunku rysunek2.bgf do pamieci programu programowania
```

że wyświetlacz tego typu nie ma własnego generatora znaków – można go zrealizować na drodze programowej. Ekran tego wyświetlacza jest podzielony na dwie części, z których każda dzielona jest na 8 tak zwanych stron, co przedstawiono na **rys. 2**. Ponieważ w Bascomie do obsługi tego typu wyświetlacza wykorzystać można gotowe instrukcje, więc trzeba nie wiedzieć, w jaki sposób odbywa się jego obsługa. Na **rys. 3** przedstawiono schemat dołączenia wyświetlacza graficznego JM12864a do mikrokontrolera. Linie D0...D7 wyświetlacza są liniami danych, natomiast linie E, R/W, RS, /CS1, /CS2 są liniami kontrolnymi. Wykorzystywany wyświetlacz ma podświetlenie, którego prąd jest ograniczany przez rezystor R1. Od wartości tego rezystora zależy jasność podświetlenia. By wyświetlacz działał

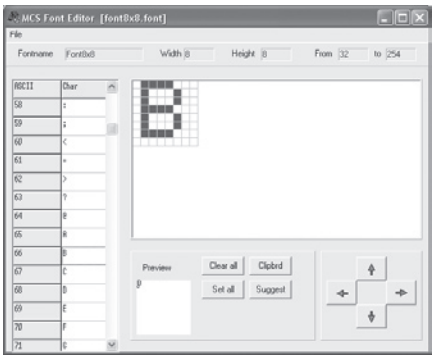


Rys. 2. Organizacja pamięci obrazu w sterowniku KS0107B

poprawnie, na wejście obwodu regulacji kontrastu (V_{ee}), należy podać ujemne napięcie bliskie –10 V. Takie napięcie wytwarzane przez przetwornice zawartą w układzie U2 jest podawane poprzez potencjometr regulacji kontrastu P1. Najprostszym, choć nie jedynym sposobem uzyskania napięcia ujemnego –10 V jest zastosowanie przetwornicy pojemnościowej, której niewielka wydajność prądowa w tym przypadku w zupełności wystarcza. Jest dostępnych wiele różnych scalonych przetwornic pojemnościowych. W taką przetwornice wyposażony jest popularny i dosyć tani układ interfejsowy (U2), który wykorzystano w tym przykładzie. Przetwornica zawarta w MAX232 wytwarza napięcia ok. ±10 V. Na linii V– (k. 6) jest dostępne napięcie –10 V, które poprzez P1 jest podawane na linie V_{ee} wyświetlacza. Oczywiście do wytworzenia ujemnego napięcia można wykorzystać inne układy jak na przykład MAX680, którego schemat aplikacyjny przedstawiono w dalszej części artykułu. Na **list. 1** przedstawiono przykład konfiguracji i wykorzystania instrukcji umożliwiających obsługę wyświetlacza graficznego z kontrolerem KSx. Na początku programu załączone zostają biblioteki *glcdKS108.lib* wymagana do obsługi wyświetlacza z kontrolerem KSx oraz *FP_Trig.lib*, która jest biblioteką funkcji trygonometrycznych, wykorzystywanych w programie. Wyświetlacz graficzny można skonfigurować instrukcją *Config Graphlcd*. Jako pierwszy parametr należy podać typ wyświetlacza połączony z jego rozdzielczością. Wykorzystywany wyświetlacz miał rozdzielczość 128*64 pikseli, a ponieważ jego obsługa jest podobna do obsługi wyświetlaczy z kontrolerem SED, więc należy taki typ wyświetlacza wybrać. Tak więc pierwszy parametr skonfigurowano jako *128*64sed*. W Bascomie możliwa jest obsługa wyświetlaczy graficznych o rozdzielczościach do 240*128 pikseli. Pozostałe parametry instrukcji konfiguracyjnej dotyczą linii portów komunikujących się z wyświetlaczem. W przypadku wyświetlacza z kontrolerem KSx, który nie ma wejścia zerującego, a instrukcja konfiguracyjna wymaga podania linii Reset, należy podać nieużywaną linię mikrokontrolera. Dalej w programie po dyrektywie *$eeprom* zapisywany jest w pamięci EEPROM mikrokontrolera pierwszy rysunek graficzny, który będzie wyświetlany. Jest on identyfikowany etykietą *Rys2*. W programie jest dostępna jedna procedura, która w zależności od parametrów rysuje na wyświetlaczu wykres funkcji sin lub cos. Utworzonych zostało także 5 zmiennych, z których jedna jest typu Single, co było wymagane przy obliczaniu funkcji trygonometrycznych. Pierwszą instrukcją odnoszącą się do wyświetlacza jest instrukcja *Cls*, która czyści jego ekran. Ponieważ wyświetlacz zarówno z kontrolerem KS, jak i SED nie ma wbudowanego generatora znaków, więc został on w Bascomie zrealizowany programowo. Informacje o znakach są przechowywane w odrębnych plikach z rozszerzeniem *.font*. W Bascom można znaleźć dwa takie przykładowe pliki ze znakami o wielkości 8x8 i 16x16 pikseli. W przykładowym programie, w jego końcowej części, zaimportowano dyrektywą *$include* plik *font8x8.font*, w którym są znaki o wielkości 8x8 pikseli. Bascom ma dodatkowy program *Font Editor* (**rys. 4**), którym można tworzyć fonty z własnymi znakami jak i modyfikować już istniejące. W

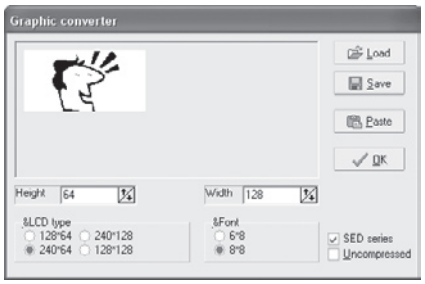


Rys. 3. Sposób dołączenia wyświetlacza JM12864A do mikrokontrolera



Rys. 4. Widok okna programu Font Editor

celu używania zaimportowanego pliku ze znakami, należy go ustawić jako domyślny poprzez użycie instrukcji *Setfont*, w której parametrem jest nazwa fontu. W tego typu wyświetlaczach (z kontrolerem KSx i SED) do wyświetlenia tekstu służy instrukcja *Lcdat*. Pierwszymi dwoma parametrami tej instrukcji są pozycje tekstu, przy czym parametr pierwszy odnosi się do pozycji poziomej w pikselach, a drugi do pionowej, ale nie w pikselach, lecz w liniach. Trzecim parametrem jest tekst (stała) lub zmienna, której zawartość ma być wyświetlana. Ostatni, opcjonalny parametr, określa czy tekst będzie odwrócony tzn. znaki będą przedstawiane przez zgaszone piksele a nie przez zapalone. Przy braku tego parametru lub gdy ma wartość 0, znaki będą wyświetlane normalnie, a przy 1 będą wyświetlane z inwersją. W przykładzie, dwa teksty zostają wyświetlone bez inwersji, a dwa kolejne z inwersją. Instrukcja *Line* umożliwia narysowanie linii. Pierwsze dwa parametry określają współrzędne x, y początku linii, a kolejne dwa – współrzędne x, y jej końca. Ostatni parametr określa kolor linii. Przy wartości 0, linia jest reprezentowana przez zgaszone piksele, a przy 1 przez zapalone. W przykładzie, za pomocą instrukcji *Line* został narysowany prostokąt otaczający wyświetlony tekst. W pętli *For*, wykonywanej aż *I* osiągnie wartość większą od 120, wykonywana jest kilkakrotnie instrukcja *Circle*, która rysuje okrąg. Pierwsze dwa parametry określają współrzędne x, y środka okręgu, trzeci parametr jest promieniem, a ostatni tak, jak w przypadku *Line* określa kolor. Przy kolorze o wartości 1, okrąg będzie reprezentowany przez zapalone piksele. Po odczekaniu 2 sekund, ekran LCD jest czyszczony oraz wywoływana jest dwa razy procedura *Rys_fun*, która rysuje funkcję sin oraz cos. Pierwszy parametr określa, która funkcja ma być rysowana. Przy wartości 1, rysowana jest funkcja sin(x), a przy 0 funkcja cos(x). Drugi parametr określa pozycję (w pionie) rysowanej funkcji na wyświetlaczu. Dodatkowo, przez zastosowanie instrukcji *Lcdat*, narysowane funkcje zostały podpisane. Procedura *Rys_fun*, rysująca funkcję sin lub cos, ma prostą budowę. Obliczanie wartości wybranej funkcji odbywa się w pętli *For*, wykonywanej aż *S* osiągnie wartość 640. W pierwszej kolejności, wartość *S* (która reprezentuje stopnie) po przepisaniu do *W* zamieniana jest na radiany za pomocą funkcji *Deg2rad*. W zależności od parametru *R* obliczana jest wartość funkcji sin lub cos. Obliczona wartość funkcji po przemnożeniu zamieniana jest na wartość całkowitą, a po skalowaniu i obliczeniu pozycji jest za pomocą instrukcji *Line* (która w tym przypadku rysuje tylko jeden punkt) wyświetlana w postaci punktu. Po wykonaniu procedury *Rys_fun* dla sin oraz cos, zostaną narysowane dwie sinusoidy, z których jedna będzie przesunięta względem drugiej o 90 stopni. Na wyświetlaczu graficznym można także wyświetlić dowolny rysunek, który jest zapisany w pamięci EEPROM lub Flash mikrokontrolera. W przykładzie zapisano dwa rysunki różniące się szczegółami. Poprzez naprzemienne wyświetlanie tych rysunków uzyskano prostą animację wyświetlanej postaci. Pierwszy rysunek, o czym była mowa, został zapisany w pamięci EEPROM mikrokontrolera, natomiast drugi w pamięci Flash, zaraz za danymi dotyczących fontu. Rysunek zapisany z pamięci Flash identyfikowany jest przez etykietę *Rys1*. Załączane pliki graficzne powinny mieć rozszerzenie *.bgf*. W Bascom znajduje się program *Graphic Converter* (**rys. 5**), który umożliwia konwersje plików graficznych zapisanych w formacie BMP na format BGF. Należy pamiętać, że konwertowana bitmapa powinna być czarno–biała i mieć wielkość zgodną z wielkością ekranu wyświetlacza. W tym przypadku powinna to być bitmapa o wielkości 128x64 pikseli. Do dołączenia pliku z grafiką, zarówno do pamięci Flash jak



Rys. 5. Program Graphic Converter

i EEPROM, służy dyrektywa *$bgf*, której parametrem jest plik z grafiką w formacie BGF. Po odczekaniu w programie kolejnych 2 sekund, ekran LCD jest czyszczony i program przechodzi do wyświetlania w nieskończonej pętli zapisanych rysunków. Do wyświetlenia rysunku z pamięci Flash służy instrukcja *Showpic*. Jej dwa pierwsze parametry określają współrzędne x, y górnego lewego rogu rysunku. Ostatnim jej parametrem jest etykieta pod którą znajduje się wyświetlany rysunek. Do wyświetlenia drugiego rysunku, który został zapisany w pamięci EEPROM, służy instrukcja *Showpice*, która ma identyczne parametry jak *Showpic*. Etykietą instrukcji *Showpice* będzie *Rys2*, czyli rysunek zapisany w pamięci EEPROM. Rysunki są wyświetlane naprzemiennie w pętli z opóźnieniem w pewnym sensie losowanym przez funkcję *Rnd*. Wartość losowana może wynosić od 0 do 255 ms. Poprzez wyświetlanie naprzemiennie dwóch nieco różniących się rysunków uzyskano „postać" losowo poruszającą ustami. Dla wyświetlaczy graficznych z kontrolerem KSx i SED dedykowane są jeszcze instrukcję *Glcdcmd* oraz *Glcddata*. Instrukcja *Glcdcmd* umożliwia wysłanie do wyświetlacza rozkazu, a instrukcja *Glcddata* danej. Oczywiście wszystkie przedstawione dane w tym punkcje odnoszą się także do wyświetlaczy graficznych z kontrolerem SED, przy wykorzystaniu którego zamiast biblioteki *glcdKS108.lib* należy załączyć bibliotekę *glcdsed.lib*. Przyłączenie wyświetlacza graficznego z kontrolerem SED do mikrokontrolera jest podobne jak wyświetlacza z kontrolerem KSx, z tym, że należy dodatkowo wykorzystać nieużywaną w tym przypadku linię Reset.

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