Zasilacz laboratoryjny sterowany cyfrowo, część 3

Cyfrowy zasilacz laboratoryjny jest sterowany programem napisanym w języku PIC BASIC, podczas gdy program w Visual BASIC służy do wyświetlania obrazu panelu sterującego na ekranie komputera. Szczegóły związane z działaniem tych programów przedstawiamy w artykule.

Jak to robią inni? ELEKTOR ELECTRONICS

Uwaga!
Źródło programu sterującego pracą zasilacza oraz kompilator Basica znajdą Czytelnicy EPo/oL na płycie CD-EP5/2002B.

Na **rys. 5** przedstawiono sieć działań programu mikrosterownika. Na początku uruchamia się krótka sekwencja inicjalizacyjna, kasująca ustalone wartości do zera i konfigurująca kilka wyprowadzeń mikrosterownika. Następnym krokiem jest pomiar aktualnych wartości napięcia i prądu, objęty już pętlą główną programu. Wszystkie pozostałe części programu następują kolejno w pętli. Jedyne rozgałęzienia zachodzą w czasie odczytywania stanu przycisków. Procedura odczytu stanu przycisku jest, jak widać z sieci działań, dość nużąca. Polega na odczytywaniu stanu wszystkich kolejnych przycisków i zapamiętywaniu przez mikrokontroler kodów tych przycisków, które zostały naciśnięte. W końcu na zapytanie „czy jakiś przycisk jest wciśnięty?“ mikrokontroler sprawdza, czy któryś z przycisków został rzeczywiście naciśnięty. Jeśli tak, jest uruchamiane rozgałęzienie, kodujące zwiększenie lub zmniejszenie odpowiedniej ustalonej wartości, dopóki wartość ta mieści się w dozwolonym zakresie. Wtedy zostaje wyświetlony nowo ustalony punkt. Funkcję automatycznego powtarzania skanowania przycisków uzupełnia półsekundowy czas opóźnienia. Jeżeli nie został naciśnięty żaden przycisk, program powraca na początek głównej pętli do ponownego pomiaru napięcia i prądu.

Program w BASIC-u

Listing kodu źródłowego mikrosterownika jest przedstawiony na **list. 1**. Program ten, napisany w PIC BASIC 1.3, można pobrać z *www.pic-basic.de* (pliki z działu *Download* tej strony publikujemy także na płycie CD-EP5/2002B). Można tam także znaleźć więcej informacji na temat tego języka, a także najnowszą wersję jego kompilatora.

PIC BASIC pozwala pisać programy mikrosterowników łatwo i szybko. Ułatwia także ich kompilację i programowanie mikrosterowników. W czasie pisania tego artykułu informacje te były dostępne jedynie w języku niemieckim.

Budowa programu jest następująca: najpierw deklaruje się wszystkie używane w programie zmienne. Jest 13 zmiennych o wielkości jednego bajta i dwie wielkości dwóch bajtów, w RAM mikrosterownika zajmują razem 17 bajtów (a dalszych 12 PIC BASIC rezerwuje jako część roboczą). Następnie rozpoczyna się pierwsza część programu, nazwana „inicjalizacją“. Etykieta *Start* wyznacza punkt wejściowy głównej pętli. Listing zawiera liczne komentarze i szczegółowy opis programu nie jest konieczny. Jednak poniżej zamieszczono kilka kolejnych uwag.

Przetwornik A/C

Polecenie *ADW A2, 5380, 0, Meas_Voltage* służy do przeprowadzenia konwersji analogowo-cyfrowej i wpisania zmierzonej wartości do zmiennej *Meas_Voltage* (zmierzone napięcie). Współczynnik skali może zostać zmieniony przez zmianę wartości 5380. Układ został jednak tak zaprojektowany, ze taka zmiana nie jest zwykle potrzebna.

Podprogramy w asemblerze

Podprogram *Format* ma długość 30 bajtów i zawiera kilka podprogramów (także w języku PIC BASIC), potrzebnych do użycia innych poleceń BASIC, formatujących wartości mierzone do wyświetlenia. Sztuczka ta umożliwia zaoszczędzenie sporo cennej pamięci programu.

Ośmiobajtowy podprogram RS232E jest napisany także w asemblerze. Ustawia on sygnał

Rys. 2. Listing programu w języku PIC BASIC

```
'PIC-BASIC-1.3-Quelltext
'Netzteil 25V, 2,5A (oder 20V, 1A)
VarB Lh1, Lh2, Lh3, Lh5, Lh6, Lh7, Uwert, Iwert, y
VarB Knopfnummer, Flag, AufrufZähler, Bitmuster
VarW MessSpannung, MessStrom

'----------------------------
'Hauptprogramm
Init:
CV Uwert, Iwert  'Uwert bei jedem Neustart auf "0"
Inc Iwert        'Iwert bei jedem Neustart auf "10mA"
'Inc Iwert      'Zeile für 1A (bei 2,5A ersatzlos weglassen)
Low A3           'ADW-Ausgang auf "0" setzen
High B2          'CTS: keine Empfangsbereitschaft

Start:
'Spannung und Strom messen
Low A4   'Mux auf U
ADW A2, 5380, 0, MessSpannung  'Spannungsmessung
MessSpannung = MessSpannung Shr 1   'entspricht: Wert / 2
High A4   'Mux auf I
ADW A2, 5380, 0, MessStrom  'Strommessung
MessStrom = MessStrom Shr 1   'entspricht: Wert / 2
'MessStrom = MessStrom Shr 2   'entspricht / 4   'Zeile für 1A-Version

'Messwerte formatieren
Call Format

'Messwerte auf dem LCD anzeigen
LCD B5, " ", Lh1, Lh2, ",", Lh3, "V    ", Lh5, ",", Lh6, Lh7, "A "

'Meßwerte über RS232 senden
SerOut B3, 9600, "D", #MessSpannung, #MessStrom, 13

'Wenn vorhanden, neue Sollwerte über RS232 empfangen
Call RS232E

'die Sollwerte über DAW ausgeben
PWM A1, Uwert, 64   'Spannung einstellen (250 = 25V)
PWM A0, Iwert, 64   'Strom einstellen (250 = 2,5A bzw. 200 = 1A
                    'für 1A-Version)

'Knöpfe scannen
Einsprung:
Flag = %00010000    'nur Bit 4 auf "H" (wird bei gedrücktem
                    'Knopf zurückgesetzt)
CV AufrufZähler, Knopfnummer   'Variablen auf "0" setzen
Call ButtonScan
Call ButtonScan
Call ButtonScan
Call ButtonScan
Call ButtonScan
Call ButtonScan
Call ButtonScan
Call ButtonScan
Branch Knopfnummer, Start, S2, S3, S4, S5, S6, S7, S8, S9
                'wenn kein Knopf
' gedrückt wird zum Label "Start" springen
S2:
If Uwert > 240 Then Skip
'If Uwert > 190 Then Skip   'Zeile für 1A-Version
Uwert = Uwert + 10
Goto Uwert_anzeigen

S3:
If Uwert < 10 Then Skip
Uwert = Uwert - 10
Goto Uwert_anzeigen

S4:
If Uwert > 249 Then Skip
'If Uwert > 199 Then Skip   'Zeile für 1A-Version
Inc Uwert
Goto Uwert_anzeigen

S5:
If Uwert < 1 Then Skip
Dec Uwert
Goto Uwert_anzeigen

S6:
If Iwert > 240 Then Skip
'If Iwert > 190 Then Skip   'Zeile für 1A-Version
Iwert = Iwert + 10
Goto Iwert_anzeigen

S7:
If Iwert < 10 Then Skip
Iwert = Iwert - 10
Goto Iwert_anzeigen

S8:
If Iwert > 249 Then Skip
'If Iwert > 198 Then Skip 2  'Zeile für 1A-Version
Inc Iwert
'Inc Iwert  'Zeile für 1A (bei 2,5A ersatzlos weglassen)
Goto Iwert_anzeigen

S9:
If Iwert < 1 Then Skip
'If Iwert < 2 Then Skip 2  'Zeile für 1A-Version
Dec Iwert
'Dec Iwert  'Zeile für 1A (bei 2,5A ersatzlos weglassen)

Iwert_anzeigen:
'y = Iwert Shr 1    'entspricht/2 Zeile für 1A (bei 2,5A ersatzlos
                    'weglassen)
LCD B5, "     ", #Iwert, "0mA"
'LCD B5, "     ", #y, "0mA"  'Zeile für 1A-Version
Pause 500
Goto Einsprung

Uwert_anzeigen:
LCD B5, "     ", #Uwert, "00mV"
Pause 500
Goto Einsprung

Sub ButtonScan
  LookUp AufrufZähler, %11111110, %11111101, %11111011, %11110111,
%11101111, %11011111, %10111111, %01111111, Bitmuster
  EXPo B5, Bitmuster, 0  'Dieser Befehl nutzt die gleichen Controller-
Pin's
' wie das LCD. Durch den Wert "0" wird "Bitmuster" am LCD vorbeigeschoben.
  Inc AufrufZähler
  PBI %00010000 = Flag   'nur Bit 4 von Port B einlesen
  If Flag <> 0 then Skip
   'nächsten Befehl überspringen wenn Knopf nicht gedrückt
  Knopfnummer = AufrufZähler  'Knopfnummer merken
EndSub


'Das folgende Basic-Unterprogramm "Einlesen" wird nur aus dem
'Assembler-Unterprogramm "RS232E" aufgerufen

Sub Einlesen
  SerIn B0, 9600, #Uwert, #Iwert
  Uwert = Uwert Min 250   'auf 25 Volt begrenzen  'Zeile für 2,5A
  'Uwert = Uwert Min 200  'auf 20 Volt begrenzen  'Zeile für 1A-Version
  Iwert = Iwert Min 250   'auf 2,5 Ampere begrenzen  'Zeile für 2,5A
  'Iwert = Iwert Min 200  'auf 1 Ampere begrenzen  'Zeile für 1A-Version
  Y = 1          'Schleife sofort verlassen
Endsub

Ass Format
   ;Spannung formatieren
  MOVF 24,W
  MOVWF HWERT2
  MOVF 23,W
  MOVWF 21
  MOVLW 2
  Call Packer
  MOVWF 27
  MOVLW 4
  Call Packer
  MOVWF 28
  MOVLW 6
  Call Packer
  MOVWF 29
    ;Strom formatieren
  MOVF 26,W
  MOVWF HWERT2
  MOVF 25,W
  MOVWF 21
  MOVLW 2
  Call Packer
  MOVWF 30
  MOVLW 4
  Call Packer
  MOVWF 31
  MOVLW 6
  Call Packer
  MOVWF 32
  Return

Packer:     ;spart Wiederholung der Zeilen, insges. 8 Byte
Programmspeicher
  MOVWF FSR
  CALL SOSS
  MOVF LWERT1,W
EndAss

Ass RS232E
  CLRF 35       ; Clrf Y (= R35)
RS232:
  BCF PB,2      ; CTS: Empfangsbereitschaft
  BTFSS PB,0    ; RxT Pintest
  Call Einlesen ; Basic-Unterprogramm aufrufen
  DECFSZ 35,F   ;
  GOTO RS232    ; Schleife 255 mal durchlaufen (als Zeitfenster)
  BSF PB,2      ; CTS: keine Empfangsbereitschaft
EndAss
```

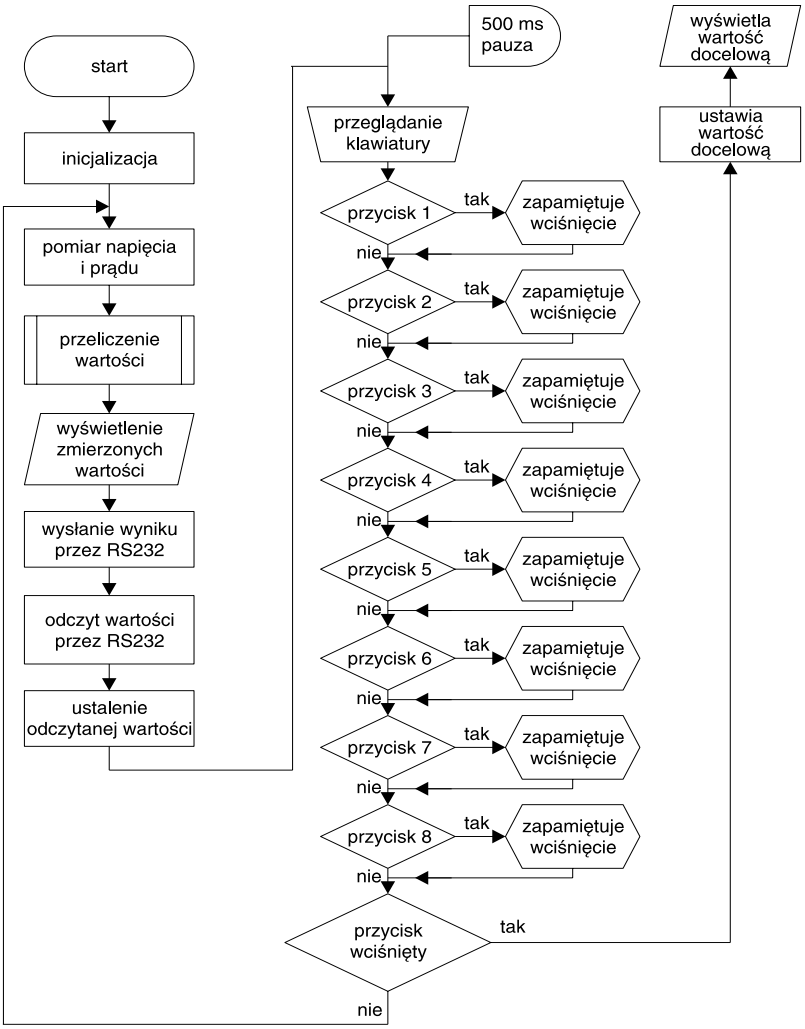
CTS w stanie wysokim i czeka przez chwilę, aby sprawdzić, czy z komputera są wysyłane dane. Jeśli tak, program w asemblerze przywołuje podprogram BASIC *Read*, służący do odczytywania napływających danych.

Wszystkie pozostałe części programu są napisane wyłącznie w BASIC-u. Cały skompilowany program ma objętość 1009 bajtów (wersja 2,5A) lub 1021 bajtów (wersja 1A) i niemal zapełnia pamięć programu układu PIC16F84. Te wiersze programu, które zależnie od wersji zasilacza wymagają zmiany, są oznaczone w listingu programu BASIC.

Jeśli chce się umożliwić działanie kontrolnego licznika zegarowego (*watchdog timer*) w mikrosterowniku, to słowo konfiguracyjne na wyjściu z asemblera musi zostać zmienione na:

```
CONFIG B'11111111110101'
```

W pętlę główną i w pętlę skanowania przycisków musi także



Rys. 5. Sieć działań programu mikrosterownika

zostać wstawiona instrukcja CLRWDT. Pętla skanowania trwa ponad 500ms, a pętla główna około 780ms. Przy wartościach ustawianych w rejestrze *Option* w czasie startu, kasowanie kontrolnego licznika zegarowego następuje po 2,3s. Czas ten wystarcza dla obu pętli (wystarczyłaby nawet połowa). Najprostszym miejscem do wprowadzenie instrukcji CLRWDT jest tuż po etykiecie *Entry*. Następujący po niej kod zostanie przesunięty o jedno miejsce, co w tym przypadku jest bez znaczenia.

Protokół interfejsu

Pakiet danych, które zasilacz wysyła przez swój interfejs, ma następującą strukturę:

```
Duuuuuiiiiiż
```

Najpierw jest wysyłany znak D, potem pięciocyfrowe wartości napięcia i prądu, a na końcu znak powrotu karetki. Najmniej znacząca cyfra napięcia oznacza 10mV, a natężenia prądu 1mA. Decydują one o dziesięciokrotnie większej rozdzielczości niż pokazywana przez wyświetlacz. Pierwszymi cyframi napięcia i prądu zawsze są zera.

Przy przesyłaniu do zasilacza ustalonych wartości, przesłane muszą być i napięcie i natężenie prądu, jedno natychmiast po drugim. Można wysłać do trzech cyfr każdej wartości:

```
uuużiiiż
```

Po każdej z wartości musi zostać wysłany znak niecyfrowy (np. znak powrotu karetki). W każdej wersji zasilacza wartość napięcia 20V musi zostać wysłana w postaci 200, a wartość natężenia prądu 1A w postaci 100 dla silniejszego modelu i 200 dla słabszego.

Program w Visual BASIC

Specjalnie napisany dla tego urządzenia program sterujący działa pod Windows 95 lub 98. Jego interfejs sterowania imituje panel czołowy zasilacza (**rys. 6**). Napięcie i natężenie prądu ustala się w nim tak samo jak w rzeczywistym zasilaczu za pomocą ośmiu przycisków. Zwolnienie klawisza myszy skutkuje wysłaniem wartości do zasilacza. Ustalane i aktualne napięcie i natężenie prądu są wyświetlane na pozorowanym wyświetlaczu ciekłokrystalicznym. Po kliknięciu w okienku programu poza przyciskiem, pokazuje się okienko ustalania. Można w nim konfigurować wersję zasilacza (2,5A lub 1A) oraz port komunikacyjny danych (COM 1 do 4). Jeżeli mysz jest połączona z COM1, może się zdarzyć, że program nie będzie poprawnie działał przez COM3, jak się to często w komputerach zdarza.

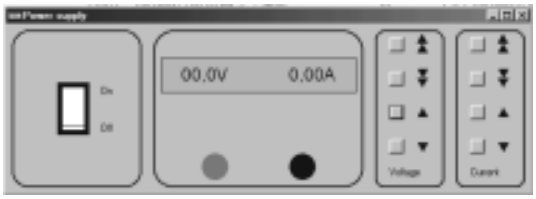
Po prawej stronie okienka konfiguracyjnego są podane nazwy plików logu i sterowania. Kliknięcie na sąsiednim przycisku „Start“ skutkuje możliwością odczytu bądź zapisu pliku. Plik zawiera wyświetlaną wielkość i datę jej ostatniej zmiany (w przypadku pliku logu), albo mającej nastąpić zmiany (w przypadku przygotowanego pliku sterowania). W najprostszym przypadku plik sterowania może zostać utworzony z pliku logu przez modyfikację daty.

Poniższy przykład rekordu jednego wiersza danych pokazuje format, używany w plikach logu i sterowania:

```
#2000-08-20 14:35:53#,"04.9V","0.97A"
```

Pomiędzy dwoma symbolami „#“ mieści się data (w formacie międzynarodowym) i czas zmiany, która została lub ma zostać dokonana. Czas i dwie dane elektryczne w rekordzie są rozdzielone przecinkami. Ułatwia to przetwarzanie i na przykład graficzną prezentację zawartości pliku za pomocą programu arkusza kalkulacyjnego.

Opracował R. Pagel, EE



Rys. 6. Widok okna programu sterującego zasilaczem z poziomu PC