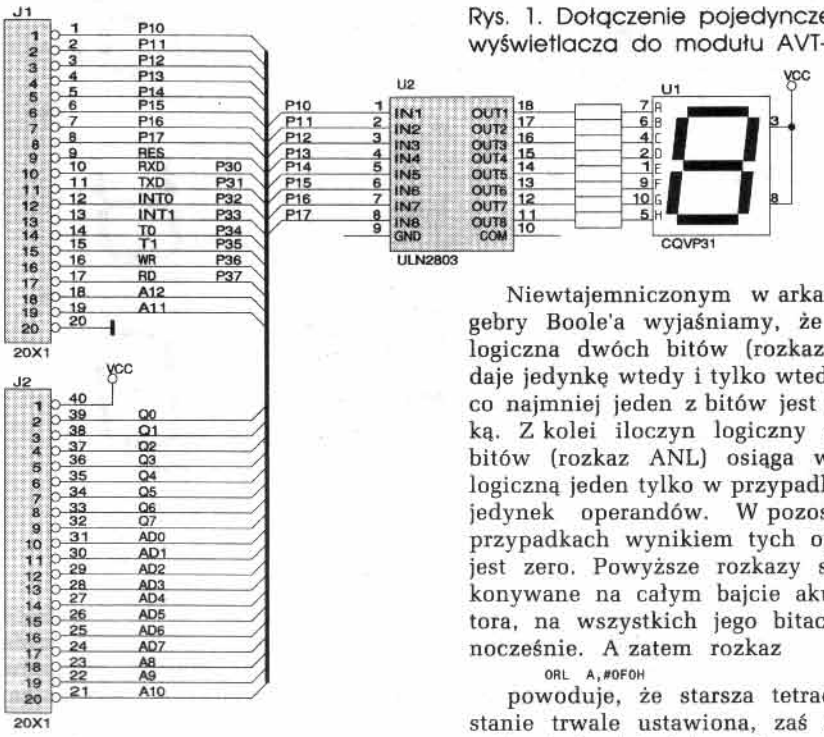
Wiemy już z poprzednich odcinków tego kursu jak dołączyć klawiaturę, a teraz zajmiemy się sterowaniem układów wyświetlacza.

Programowanie '51, cd. Sterowanie układami wyświetlania

Układy wyświetlania, które będą nas interesowały, to wszelkiego rodzaju wyświetlacze segmentowe, rastrowe i inne tego typu urządzenia zobrazowania informacji. W prostych systemach mikroprocesorowych stosowane układy wyświetlania nie są wyszukane, w tym również z powodu niewielkich możliwości samego systemu.

Na **rys. 1** pokazano układ jednocyfrowego wyswietlacza sterowanego bezpośrednio przez mikroprocesor. Do jego realizacji potrzebny jest tylko buforujący wzmacniacz prądu segmentów (ULN2803 - patrz USKA UA 2/94). Sterowanie tym wyswietlaczem polega na wysłaniu bitowej informacji na port P1. Z punktu widzenia programisty nie jest to taki wielki problem, aby poświęcać mu więcej uwagi.

Rzadko kiedy potrzebny jest nam jeden wskaźnik siedmiosegmentowy, z reguły chcemy mieć ich więcej. Układ z rys. 1 jest mało oszczędny, wykorzystuje osiem linii portu do prezentacji co najwyżej szesnastu różnych symboli - mowa oczywiście o symbolach czytelnych dla przeciętnego zjadacza chleba. Z tego powodu sięga się po transkodery. Najpopularniejszą grupą transkoderów jest seria TTL '47. Służą one do zmiany kodu BCD na kod wskaźnika siedmiosegmentowego. Tym sposobem dwa porty modułu AVT-222 potrafią wysterować cztery wskaźniki (**rys. 2**). Wartą przytoczenia jest krótka procedura składania kodów BCD dwóch liczb w jeden bajt. Będzie nam to potrzebne do przesłania tych danych na port. Zakładamy, że kody danych liczb znajdują się na młodszych tetradach rejestrów R0 i R1.



Rys. 1. Dołączenie pojedynczego wyświetlacza do modułu AVT-222.

```
        MOV  A,R1
        ORL  A,#0F0H
        SWAP  A
        MOV  R1,A
        MOV  A,R0
        ORL  A,#0F0H
        ANL  A,R1
```

Niewtajemniczonym w arkana algebry Boole'a wyjaśniamy, że suma logiczna dwóch bitów (rozkaz ORL) daje jedynkę wtedy i tylko wtedy, gdy co najmniej jeden z bitów jest jedynką. Z kolei iloczyn logiczny dwóch bitów (rozkaz ANL) osiąga wartość logiczną jeden tylko w przypadku obu jedynek operandów. W pozostałych przypadkach wynikiem tych operacji jest zero. Powyższe rozkazy są wykonywane na całym bajcie akumulatora, na wszystkich jego bitach jednocześnie. A zatem rozkaz

```
ORL  A,#0F0H
```

powoduje, że starsza tetrada zostanie trwale ustawiona, zaś zawartość młodszej będzie zachowana. Zapis 0F0H jest szesnastkowym zapisem liczby binarnej 11110000, zaś początkowe zero jest narzucone przez asemblery, które przeważnie niedopuszczają liczb zaczynających się na literę.

Rozkaz

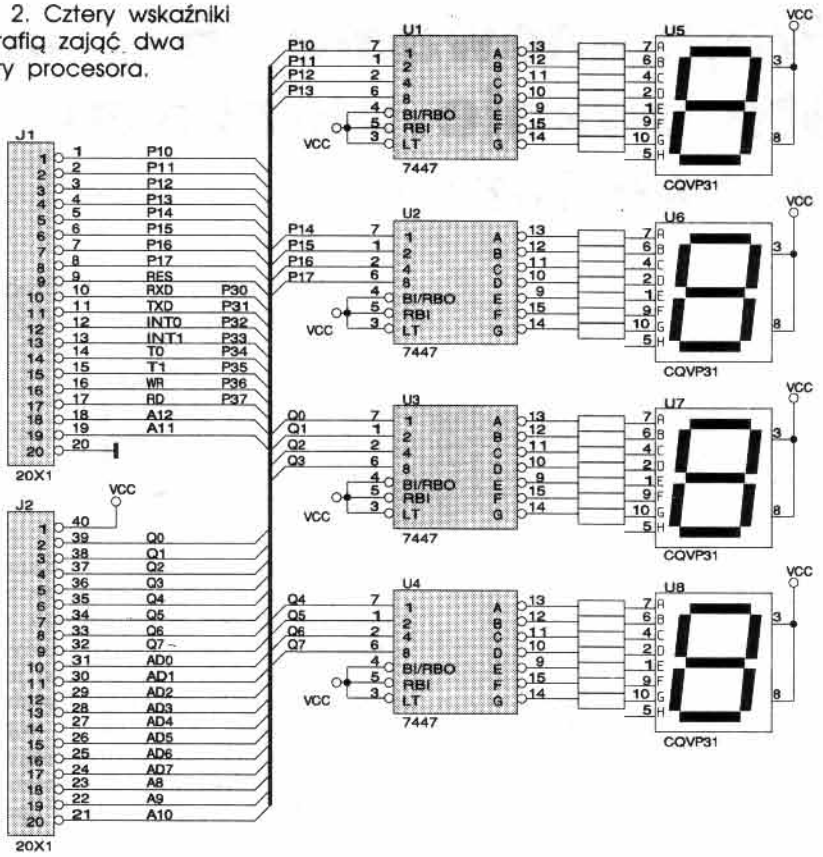
```
ANL  A,R1
```

wykorzystuje działanie sumy logicznej, która ustawiła bity nieistotnej tetrady. Jedynka w jednym z operan-

```
; WARTOSCI  ZMIENNYCH  I LITERALOW
R7REG           EQU  7
PART_SEC        EQU  30H
        ;  ADRES  KOMORKI  IRAM,
        ;  ZAWIERAJACEJ  CZESCI
        ;  ZLICZANYCH  SEKUND
PART_MSEC EQU   PART_SEC+1
SEC             EQU  PART_MSEC+1
MIN             EQU  SEC+1
NMBR_DISP       EQU  KEYB+1
ROREG           EQU  0
R1REG           EQU  1

        ORG  0
        AJMP  RESTART

        ORG  0BH
        AJMP  T0_SERVIS

        ORG  20H
RESTART:
        MOV  IE,#0
        MOV  IP,#0
        MOV  TMOD,#01H
        MOV  TH0,#03CH
        MOV  TL0,#0AFH
        MOV  PART_SEC,#10
        MOV  TCON,#00010000B
        SETB  IE.1
        SETB  IE.7
        MOV  NMBR_DISP,#4
;
;PROGRAM  GLOWNY
MAIN:
        SJMP  MAIN
; OBSLUGA  PRZERWANIA  T0
T0_SERVIS:
        PUSH  ACC
        PUSH  ROREG
        PUSH  R1REG
        NOP
        MOV  R1,TH0
        MOV  R0,TL0
        MOV  A,#017H
        CLR  C
        SUBB  A,R0
        MOV  R0,A
        MOV  A,#0FCH
        SUBB  A,R1
        MOV  R1,A
        MOV  TH0,R1
        MOV  TL0,R0
        DJNZ  PART_MSEC,T0_S1
        MOV  PART_MSEC,#200
        DJNZ  PART_SEC,T0_S1
        MOV  PART_SEC,#5
        MOV  A,SEC
        ADD  A,#1
        DA  A
        MOV  SEC,A
        MOV  A,#60H
        CJNE  A,SEC,T0_S1
        MOV  SEC,#0
        MOV  A,MIN
        ADD  A,#1
        DA  A
        MOV  MIN,A
T0_S1:
        MOV  R7,NMBR_DISP
        MOV  A,#11110111B
T0_SX4:
        RL  A
        DJNZ  R7,T0_SX4
        MOV  R1,A
        MOV  R7,NMBR_DISP
        MOV  R0,#MIN
T0_SX8:
        MOV  A,@R0
        SWAP  A
        DJNZ  R7,T0_SX5
        SJMP  T0_SX6
T0_SX5:
        MOV  A,@R0
        DJNZ  R7,T0_SX7
        SJMP  T0_SX6
T0_SX7:
        DEC  R0
        SJMP  T0_SX8
T0_SX6:
        ORL  A,#0F0H
        ANL  A,R1
        MOVX  @R0,A
        DJNZ  NMBR_DISP,T0_SX3
        MOV  NMBR_DISP,#4
T0_SX3:
        MOV  P1,#0FFH
        POP  R1REG
        POP  ROREG
        POP  ACC
        RETI
```

Listing 1

Rys. 2. Cztery wskaźniki potrafią zająć dwa porty procesora.



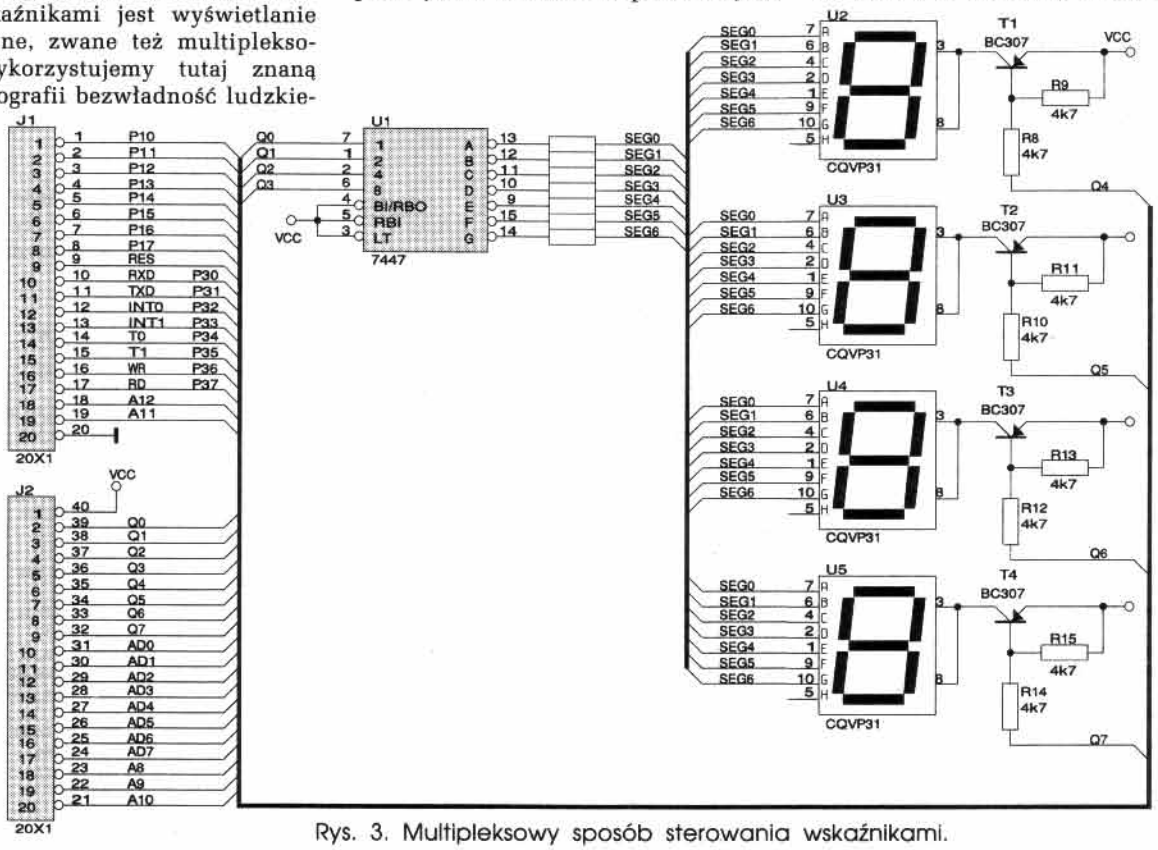
dów iloczynu logicznego ma wpływ na przepisanie do wyniku zawartości drugiego operandu, co daje w naszym przypadku połączenie obu interasujących nas terad w jeden bajt.

Innym sposobem sterowania wieloma wskaźnikami jest wyświetlanie sekwencyjne, zwane też multipleksowym. Wykorzystujemy tutaj znaną z kinematografii bezwładność ludzkiego oka, odbierającą szybko migoczące obrazy jako jeden ciągły oraz małą bezwładność fizycznych procesów włączania i wyłączania wskaźnika. Każdy z wyświetlaczy jest sterowany przez jeden krótki odstęp czasu. Jeśli czas obsługi wszystkich wyświetlaczy jest krótszy niż 20ms, migotanie znika. Ponieważ każdy wskaźnik jest zapalany tylko przez fragment okresu komutacji, w celu zachowania tej samej jasności, co przy sterowaniu ciągłym, musi on być zasilany zwiększonym prądem. Realizuje się to poprzez kilkukrotne zmniejszenie wartości rezystancji ograniczających prąd diod elektroluminescencyjnych.

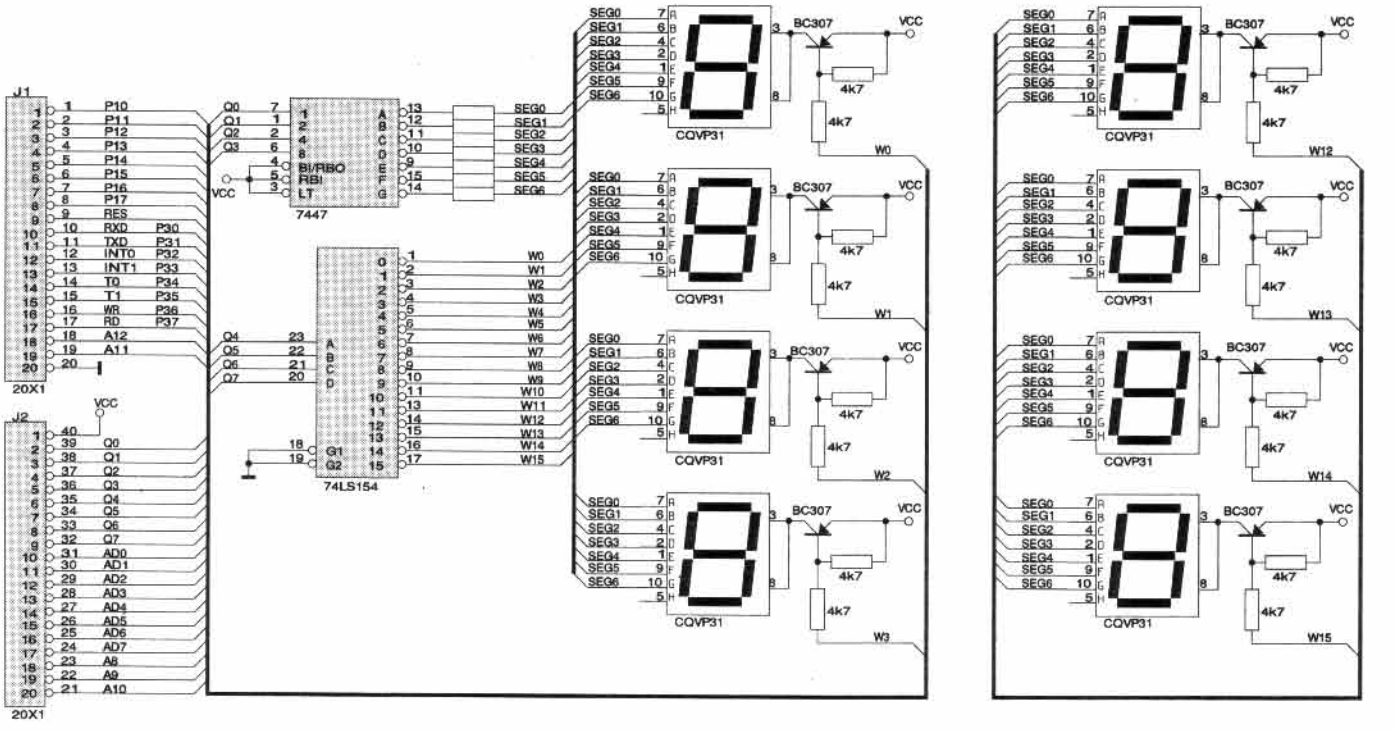
Na **rys. 3** pokazano multipleksowy układ sterowania czterema wyświetlaczami. Użycie transkodera '47 zmniejsza zapotrzebowanie układu z siedmiu linii na cztery sterujące segmentami. Pozostałe cztery linie portu Q służą do wybierania aktywnego wyświetlacza.

Na tak opracowanym układzie zbudujmy prosty timer odliczający czas od 0 do 100 minut z dokładnością jednej sekundy. W procedurze obsługi przerwania timera T0 zawrzemy metodę odliczania sekund i minut, które będą pamiętane w komórkach SEC i MIN. Okres wyświetlania znaku na jednym wyświetlaczu będzie okresem, jaki upływa pomiędzy jedną obsługą przerwania a drugą, przyjmijmy, że będzie to 1ms, czyli okres komutacji czterech cyfr wyniesie 4ms. **Listing 1** przedstawia wykonanie naszego zamiaru.

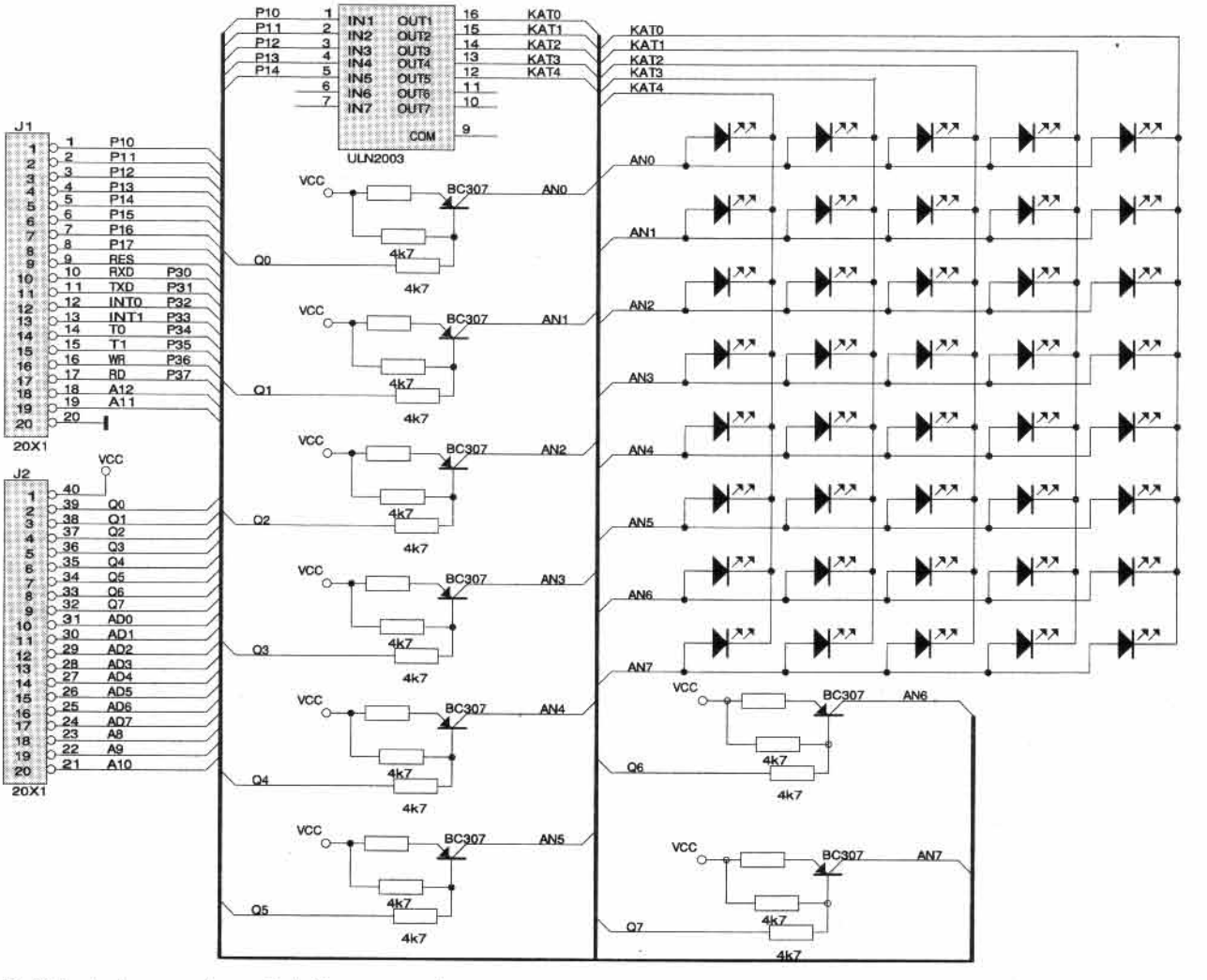
Jak widać, co tysiąc wywołań procedury T0_SERVIS zmienna SEC jest powiększana o 1, zaś co 60 zwiększenie zmiennej SEC powiększa



Rys. 3. Multipleksowy sposób sterowania wskaźnikami.



Rys. 4. Rozbudowa układu sterowania multipleksowego.



Rys. 5. Układ sterowania wyświetlacza matrycowego.

```
PART_MSEC   EQU  30H
            ; ADRES  KOMORKI
IRAM,
PART_SEC    EQU  PART_MSEC
W0          EQU  PART_SEC+1
COL_NMBR    EQU  W0+1
ROREG       EQU  0
R1REG       EQU  1
R2REG       EQU  2
T0_SERVIS:
            PUSH  ACC
            PUSH  ROREG
            PUSH  R1REG
            PUSH  R2REG
            NOP
            MOV  R1,TH0
            MOV  R0,TL0
            MOV  A,#017H
            CLR  C
            SUBB  A,R0
            MOV  R0,A
            MOV  A,#0FCH
            SUBB  A,R1
            MOV  R1,A
            MOV  TH0,R1
            MOV  TL0,R0
            DJNZ  PART_MSEC,T0_S1
            MOV  PART_MSEC,#10
            DJNZ  PART_SEC,T0_S1
            MOV  PART_SEC,#100
            MOV  A,W0
            INC  A
            CLR  ACC.7
            MOV  W0,A
T0_S1:
            MOV  P1,#0H
            MOV  R2,COL_NMBR
            MOV  A,#10000000B
T0_S3:
            RL  A
            DJNZ  R2,T0_S3
            MOV  R2,A
            MOV  DPTR,#ROM
            MOV  B,#5
            MOV  A,W0
            MUL  AB
            ADD  A,DPL
            MOV  DPL,A
            MOV  A,B          ;!
            ADDC  A,DPH
            MOV  DPH,A
            MOV
A,COL_NMBR
            DEC  A
            MOVC              ;"
A,@A+DPTR
            MOVX  @R0,A
            MOV  P1,R2REG
            DJNZ
COL_NMBR,T0_S2
            MOV
COL_NMBR,#5       ;#
T0_S2:
            POP  R2REG
            POP  R1REG
            POP  ROREG
            POP  ACC
            RETI              ;$
ROM:
;^@
            DB  11100011B
            DB  11101011B
            DB  11101011B     ;%
            DB  11101011B
            DB  11100011B
;^A
            DB  01111111B
            DB  01111111B
            DB  01111111B     ;&
            DB  01111111B
            DB  01111111B
;^B
            DB  10111111B
            DB  10111111B
            DB  10111111B     ;'
            DB  10111111B
            DB  10111111B
;^C
            DB  11011111B
            DB  11011111B
            DB  11011111B     ;(
            DB  11011111B
            DB  11011111B
            .  .
            .  .
            .  .
            DB  11111111B     ;.
            DB  11111111B
            DB  00000110B
            DB  11111111B
            DB  11111111B
                              ;/
            DB  11111111B
            DB  00001111B
            DB  11111111B
            DB  00001111B
            DB  11111111B
                              ;0
            DB  11101011B
            DB  00000000B
            DB  11101011B
            DB  00000000B
            DB  11101011B
                              ;1
            DB  10111001B
            DB  01101110B
            DB  00000000B
            DB  01101110B
            DB  10011101B
                              ;2
            DB  01111000B
            DB  10111000B
            DB  11000111B
            DB  00011101B
            DB  00011110B
                              ;3
            DB  11111111B
            DB  11111111B
            DB  11111111B
            DB  11111111B
            DB  11111111B
                              ;4
            DB  11111111B
            DB  00001111B
            DB  11101111B
            DB  11111111B
            DB  11111111B
                              ;5
            DB  11100111B
            DB  11100111B
            DB  11100111B
            DB  11100111B
            DB  11100111B
                              ;6
            DB  11111111B
            DB  11111000B
            DB  11111000B
            DB  11111000B
            DB  11111111B
                              ;7
            DB  00111111B
            DB  10011111B
            DB  11001111B
            DB  11111001B
            DB  11111100B
                              ;8
            DB  10000001B
            DB  01111110B
            DB  01111110B
            DB  01111110B
            DB  10000001B
                              ;9
            DB  11111111B
            DB  11111110B
            DB  00000000B
            DB  10111110B
            DB  11111111B
                              ;:
            DB  10011001B
            DB  01100110B
            DB  01100110B
            DB  01111110B
            DB  10111100B
                              ;;
            DB  11111101B
            DB  00000000B
            DB  11011101B
            DB  11101101B
            DB  11111101B
                              ;<
            DB  01110001B
            DB  01101110B
            DB  01101110B
            DB  01101110B
            DB  00001101B
                              ;>
            DB  10110001B
            DB  01101110B
            DB  01101110B
            DB  01101110B
            DB  10000001B
                              ;?
            DB  00001111B
            DB  01101111B
            DB  01111011B
            DB  01111101B
            DB  01111110B
                              ;@
            DB  10010001B
            DB  01101110B
            DB  01101110B
            DB  01101110B
            DB  10010001B
                              ;A
            DB  10000001B
            DB  01101110B
            DB  01101110B
            DB  01101110B
            DB  10011101B
                              ;B
            DB  11111111B
            DB  11100011B
            DB  11111111B
            DB  11100011B
            DB  11111111B
                              ;C
            DB  11111111B
            DB  11000011B
            DB  11111101B
            DB  11000111B
            DB  11111111B
                              ;D
            DB  11111111B
            DB  01111110B
            DB  10111101B
            DB  11011011B
            DB  11100111B
                              ;=
            DB  11011011B
            DB  11011011B
            DB  11011011B
            DB  11011011B
            DB  11011011B
                              ;>
            DB  11100111B
            DB  11011011B
            DB  10111101B
            DB  01111110B
            DB  11111111B
                              ;?
            DB  10001111B
            DB  01110111B
            DB  01110111B
            DB  01111000B
            DB  10111111B
                              ;@
            DB  11111111B
            DB  11111111B
            DB  11111111B
            DB  11111111B
            DB  11111111B
                              ;A
            DB  10000000B
            DB  01111011B
            DB  01111011B
            DB  01111011B
            DB  10000000B
                              ;B
            DB  10010001B
            DB  01101110B
            DB  01101110B
            DB  01101110B
            DB  00000000B
                              ;C
            DB  10111101B
            DB  01111110B
            DB  01111110B
            DB  01111110B
            DB  10000001B
                              ;D
            DB  10000001B
            DB  01111110B
            DB  01111110B
            DB  00000000B
            DB  01111110B
                              .  .
                              .  .
                              .  .
```

Listing 2

się wartość zmiennej MIN. Ażeby uprościć sobie sterowanie wyświetlaczem, wartości sekund i minut są pamiętane jako liczby BCD. Rozkaz dodawania ADD A,#1 dodaje liczby traktując je jako zapisane w kodzie naturalnym dwójkowym NB. W celu przekształcenia wyniku na liczbę BCD producent procesora przewidział rozkaz korekcji dziesiętnej DA A. Liczba BCD koduje cyfry zapisu dziesiętnego. Jako zapisana w jednym bajcie dzieli ona bajt na dwie dwie tetrady, z których młodsza reprezentuje jednostki, zaś starsza - dziesiątki. Np. dodając rozkazem ADD jedynkę do liczby 59H dostaniemy 5AH. Po korekcji dziesiętnej otrzymamy prawidłowy dla zapisu BCD wynik 60H.

Ktoś w tym miejscu zada pytanie, czy nie lepiej będzie, jeśli zamiast rozkazu ADD A,#1 użyjemy INC, który jest bezpośrednim rozkazem dodania jedynki do argumentu. Otóż popełnimy błąd, ponieważ rozkazy inkrementacji i dekrementacji są uproszczonymi rozkazami dodawania i odejmowania. Nie modyfikują one znaczników przeniesienia C i przeniesienia pomocniczego AC, a te znaczniki są potrzebne do wykonania poprawnej korekcji dziesiętnej. Po szczegóły odsyłamy do katalogów USKA µC 2-3/94.

Zmienna NMBR_DISP pamięta aktualny numer wyswietlacza. Przyjmuje ona wartości od 1 do 4. Z rys. 3 wynika, że wybór wyświetlacza polega na podaniu stanu niskiego na odpowiednie wyjście Q4-Q7. W oparciu o aktualną wartość NMBR_DISP następuje wybranie bieżącego wyswietlacza. Poprzez wykorzystanie kolejnego ustawienia zmiennych SEC i MIN w pamięci RAM procesora oraz bieżącej wartości NMBR_DISP zostaje wybrana właściwa tetrada. W ten sposób określiliśmy zawartość obu tetrad, które teraz należy złożyć w jeden bajt i wysłać na port Q modułu AVT-222.

Rozbudowa układu o dekoder wyboru wyświetlaczy pozwala na obsługę linijki 16 wyświetlaczy za pomocą jednego portu procesora. Nasza procedura ulegnie niewielkiej modyfikacji, polegającej na zliczaniu w zmiennej NMBR_DISP liczb od 0 do 15 i składaniu jej zawartości bezpośrednio z wartością wyświetlaną. Oczywiście, liczba komórek pamięci zajmowanych przez bufor wyświetlacza, przy zachowaniu tej samej konwencji zapisu, zwiększy się do 8. Przykładową konstrukcję takiego układu wyświetlania pokazuje w skrócie **rys. 4**.

Innym rodzajem wskaźnika jest wskaźnik matrycowy. Składa się on pojedynczych świecących punktów reprezentowanych przez diody LED umieszczone na planie macierzy. Sposób wewnętrznego połączenia diod LED przypomina sposób połączenia pokazany w poprzednim numerze EP w przykładowej klawiaturze matrycowej. Taki wskaźnik służy do czytelnego zobrazowania cyfr, liter i innych znaków.

Ze względu na wewnętrzną konstrukcję takiego wyświetlacza dopuszczalnym i jedynym sposobem sterowania jest sterowanie multipleksowe. **Rysunek 5** przedstawia schemat współpracy diodowej struktury matrycowej 5x8 z modułem AVT-222. **Listing 2** pokazuje procedurę sterowania taką strukturą. Program ten wyświetla co 1s kolejne znaki ASCII. Istotną jego nowością w stosunku do poprzedniego rozwiązania jest odwoływanie się do fragmentu pamięci programu za pomocą rozkazu MOVC z jednoczesnym wektorem przesunięcia w ramach wybranej grupy pięciobajtowej. W tej części programu znajduje się generator znaków ASCII, których kody są zapisywane w zmiennej W0. Samo wskazanie właściwej grupy bajtów polega na dodaniu do adresu ROM liczby, która jest iloczy-

nem kodu znaku i liczby 5. Koncepcja sterowania wyświetlaniem poszczególnych bajtów na im odpowiadających kolumnach wskaźnika jest podobna jak poprzednio. Ze względu na znaczną długość generatora znaków opublikowano tylko jego część. Krój tych znaków może być niedoskonały, wierzymy jednak, że zainteresowani Czytelnicy bez trudu poradzą sobie poprawieniem ich kształtów. Jeśli dobrze przyjrzeć się zapisowi bitowemu, zauważą oni, że w grupach pięciobajtowych zera tworzą zarys znaku. W standardzie ASCII pierwsze 32 znaki to znaki sterujące, które w naszym generatorze mają dowolne, semigraficzne odpowiedniki.

Na tym kończymy nasz elementarny kurs programowania '51. Sam typ procesora był tylko pretekstem do pokazania wybranych metod programowania mikroprocesorów w ogóle. Metody te są znane od czasów „dziadka" 8080, tylko konkretny sposób ich realizacji może się różnić, to oczywiście zależy od typu procesora i jego listy rozkazów. O pozostałych możliwościach tego i innych mikroprocesorów Czytelnicy dowiedzą się z lektury publikowanych projektów.

Mirosław Lach