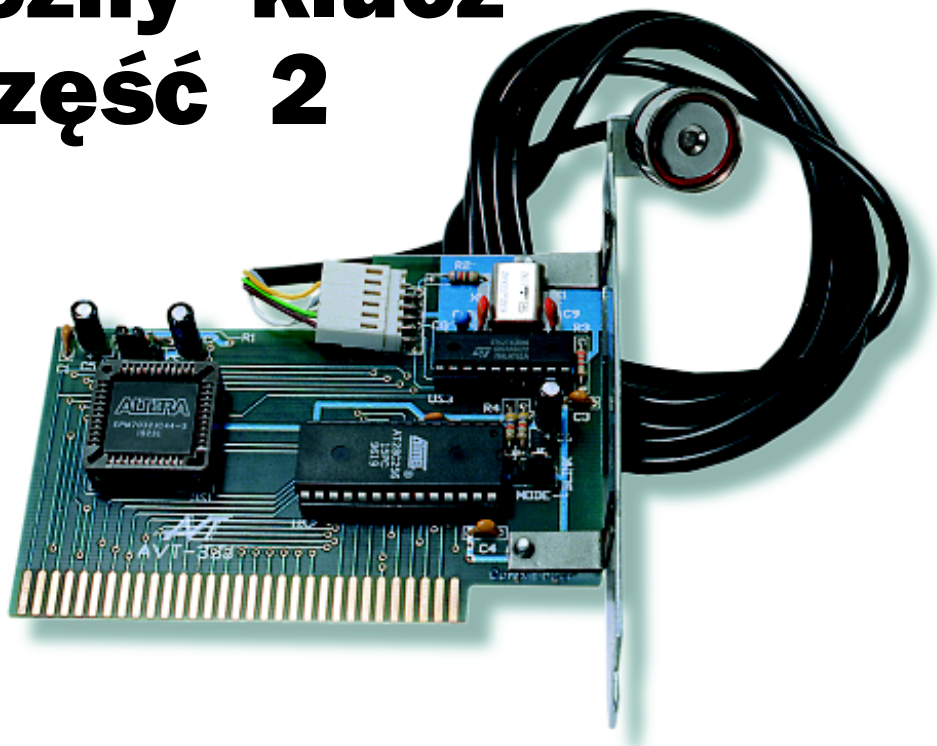
Elektroniczny klucz do PC, część 2

kit AVT-330



W poprzedniej części artykułu opisaliśmy konstrukcję karty. Dzięki zastosowaniu układów programowalnych jest ona maksymalnie uproszczona, pozostaje tylko jeden kłopot - układy te należy właściwie oprogramować. Ta część artykułu poświęcona jest omówieniu sposobu rozbudowy standardowego systemu BIOS w komputerze PC.

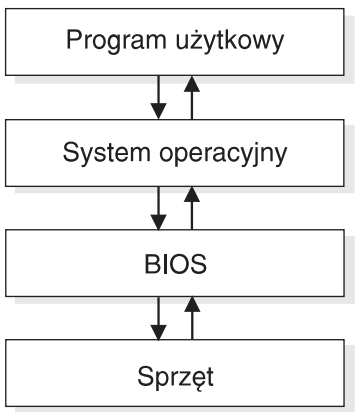
Najistotniejszym elementem, umożliwiającym prawidłową pracę elektronicznego klucza do PC, jest procesor ST62T60B (US3). Podstawowym zadaniem tego mikrokontrolera jest zapewnienie prawidłowej współpracy karty z pastylkami DS1990 (odczyt kodu pastylki i sterowanie świeceniem diody w czytniku) oraz przechowywanie w pamięci EEPROM kodów umożliwiających zdjęcie blokady komputera. Oprogramowanie tego układu jest zbliżone do zastosowanego w kicie AVT-294, nie będzie więc tu szerzej omawiane.

Sposób pracy automatu synchronicznego zbudowanego na układzie EPM7032 (US1) został przedstawiony w pierwszej części, w punkcie „Opis układu".

Poniżej omówione zostanie oprogramowanie układu EPROM 27C256 (US2), będącego rozszerzeniem standardowego BIOS-u (BIOS Extension).

BIOS i BIOS Extension w komputerze PC

BIOS (Basic Input Output System) jest programem umieszczonym w pamięci typu ROM na płycie głównej komputera. System ten zawiera dziesiątki funkcji i procedur umożliwiających pracę komputera oraz zapewniających prawidłową współpracę pomiędzy elementami komputera pochodzącymi od różnych producentów. Większość z tych procedur jest udostępniona programiście poprzez złożony system przerwań, z części nie można jednak korzystać, gdyż może to zakłócić poprawną pracę komputera lub po prostu nie ma takiej potrzeby. Można zaryzykować stwierdzenie, że BIOS jest głównym i uniwersalnym (bo zawsze działa) elementem pośredniczącym pomię-



Rys. 4. Zalecany sposób wykorzystania procedur BIOS-u przez programy użytkownika.

Listing 1. Makro wypisujące tekst na ekranie z uwzględnieniem wybranego języka (polski/angielski).

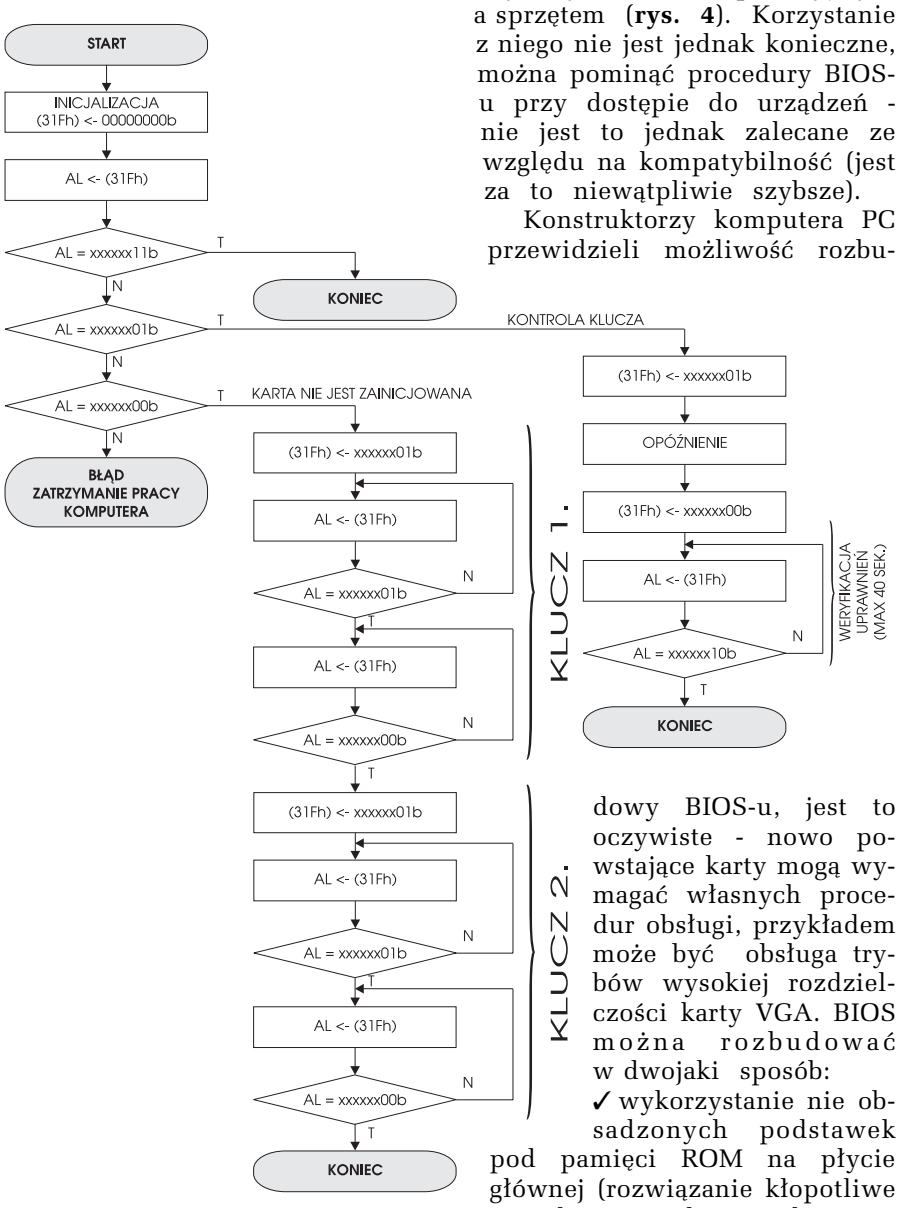
```
write MACRO napis
LOCAL NastepnyZnak1, NastepnyZnak,
KoniecTekstu, OminAng, JezPL
; D2 w|31f   == 1|-> PL
;           == 0|-> ANG
   xor c1,c1
   mov ch,c1
   mov ax,31fh
   mov dx,ax
   in al,dx
   and al,4  ; zerowanie wszystkich bitów
             ; oprócz D2
   cmp al,4
   je OminAng

; pisz po ang
   push cx
   ; w|ds segment jest OK
NastepnyZnak1:
   lea si,tekst
   add si,cx
   lodsb
   cmp al,'#'
   je KoniecTekstu
   mov ah,0eh
   int 10h
   pop cx
   inc cx
   push cx
   jmp NastepnyZnak1

; omin angielski tekst
OminAng:
   lea si,tekst
   add si,cx
   lodsb
   inc cx
   cmp al,'#'    ; znak oddzielający tekst
                 ; w|języku angielskim i|polskim
   jne OminAng

JezPL:
   push cx   ; w|ds segment jest OK
NastepnyZnak:
   lea si,tekst
   add si,cx
   lodsb
   cmp al,'$'    ; znacznik końca tekstu
   je KoniecTekstu
   mov ah,0eh
   int 10h
   pop cx
   inc cx
   push cx
   jmp NastepnyZnak
KoniecTekstu:
   pop ax
   ENDM

   .......
; przykładowy napis
t1 db 'tu wstawiamy tekst w|jezyku angielskim'
   db '#a tu tekst w|jezyku polskim','$'
```

dzy systemem operacyjnym a sprzętem (**rys. 4**). Korzystanie z niego nie jest jednak konieczne, można pominąć procedury BIOS-u przy dostępie do urządzeń - nie jest to jednak zalecane ze względu na kompatybilność (jest za to niewątpliwie szybsze).

Konstruktorzy komputera PC przewidzieli możliwość rozbudowy BIOS-u, jest to oczywiste - nowo powstające karty mogą wymagać własnych procedur obsługi, przykładem może być obsługa trybów wysokiej rozdzielczości karty VGA. BIOS można rozbudować w dwojaki sposób:

✓ wykorzystanie nie obsadzonych podstawek pod pamięci ROM na płycie głównej (rozwiązanie kłopotliwe w realizacji, gdyż uzależnione od konkretnej płyty głównej, nie ma w tym przypadku ogólnych reguł);

✓ zainstalowanie pamięci ROM na dodatkowej karcie - w opisywanym projekcie wybrano tę metodę.



Rys. 5. Algorytm działania rozszerzenia BIOS-u.

Dodatkowa pamięć ROM, w której zapisany jest BIOS extension, jest poszukiwana przez standardowy BIOS podczas testu inicjalizującego komputer (POST - Power On Self Test) w przestrzeni adresowej od C8000h do E0000h, z krokiem 2KB. Aby rozszerzenie BIOS-u było odnalezione, muszą być spełnione dwa warunki.

1. Nagłówek rozszerzenia BIOS-u musi mieć następującą postać:
   - **bajt 0** 55h;
   - **bajt 1** AAh;
   - **bajt 2** długość programu rozszerzenia BIOS-u w 512-bajtowych blokach;
   - **bajt 3** pierwszy bajt kodu - do tego miejsca przekazywane jest sterowanie przez dalekie wywołanie (tzn., że powrót do procedur standardowego BIOS-u musi być zrealizowany przez daleki powrót: retf).
2. Suma modulo 100h wszystkich bajtów rozszerzenia BIOS-u musi być równa 0 - istotna jest, zadeklarowana w drugim bajcie nagłówka, długość programu, a nie fizyczna wielkość pamięci ROM.

Nie spełnienie powyższych warunków spowoduje zignorowanie procedur zawartych w pamięci ROM.

Na zakończenie tego teoretycznego wstępu jeszcze jedna praktyczna uwaga. Sterowanie do rozszerzenia BIOS-u jest przekazywane po zainicjowaniu standardowych wektorów przerwań, jednak nie można zagwarantować, że inne rozszerzenia BIOS-u (np. kart VGA) zostały zainicjowane. Należy więc unikać stosowania tego typu odwołań (np. nie powinno korzystać się z przerwań obsługujących tryby graficzne VGA).

Algorytm

Na **rys. 5** przedstawiono algorytm programu zapisanego w pamięci EPROM karty. Pierwszym krokiem jest ustalenie stanu bitów B0 i B1 odczytanych z portu 31Fh. Możliwe są tu następujące sytuacje:

B1B0	znaczenie
0 0	karta nie jest zainicjowana (w pamięci EEPROM nie są zapisane żadne kody pastylek DS1990);
0 1	karta jest zainicjowana, przeprowadzona będzie weryfikacja uprawnień;
1 0	kombinacja błędna;
1 1	karta jest nieaktywna (wyłączona z poziomu systemu operacyjnego) lub nie będzie przeprowadzana weryfikacja uprawnień (po zresetowaniu komputera);

UWAGA: *Weryfikacja uprawnień przeprowadzana jest tylko jeden raz po włączeniu komputera (mikrokontroler generuje wtedy na bitach B1 i B0 odpowiednio stany 0 i 1). Nie zmusza to użytkownika do ciągłego przykładania klucza, jeżeli zresetowanie komputera było konieczne (mikrokontroler zwraca wtedy: B1=1 i B0=1).*

Na **listingu 1** jest pokazane makro wypisujące tekst adresowany zmienną `napis`. Makro to jest wykorzystywane do wypisywania komunikatów w wybranym języku (ustawianym jumperem LANGUAGE), przy czym ciąg znaków musi być zakończony znakiem '$', a elementem separującym tekst w języku angielskim od tekstu w języku polskim jest znak '#', znaki te wybrano ze względu na niezbyt częste stosowanie, można oczywiście zastosować inny znak. Do wypisywania znaków na monitorze jest wykorzystane przerwanie 10h (Video and Screen Services), procedura 0Eh (Write Character in Teletype Mode).

Kompletny program, wykorzystujący makro z list. 1, jest pokazany na **listingu 2**. Program bezpośrednio implementuje algorytm z rys. 5. Omówienia wymagać może jedynie pętla oczekiwania na przyłożenie pastylki DS1990 do czytnika. Dzięki wykorzystaniu przerwania 1Ah (System Timer and Clock Services), procedury 00h (Read System-Timer Time Counter) jest możliwe uniezależnienie czasu oczekiwania na przyłożenie pastylki od szybkości komputera.

Paweł Zbysiński

W kolejnym numerze przybliżymy procedury obsługi karty z poziomu systemu operacyjnego.

Listing 2. Kompletny program rozszerzenia BIOS-u.

```
poczatek:
; copyright
   write Copyr
   mov dx,31fh
   mov al,0
   out dx,al ; 31fh <- 0
   in al,dx
   and al,3  ; zerowanie bitów oprocz D1 i|D2
   cmp al,3
   jne kk1
   jmp koniec    ; Klucz OK
kk1: cmp al,1
   jne kk2
   jmp Krok3_0   ; D1=0 i|D0=1
kk2: cmp al,0
   je Krok2_0    ; D1=0 i|D0=0
   jmp Blad1 ; D1=1 i|D0=9

Krok2_0:
Krok2:
   write t2  ; system nie zainicjowany
   mov dx,31fh
   mov al,1
   out dx,al
Krok2_2:     ; I|klucz
   in al,dx
   and al,3
   cmp al,1
   jne Krok2_2

   write t5  ; inicjalizacja
   mov dx,31fh
Krok2_22:    ; czekaj na 0
   in al,dx
   and al,3
   cmp al,0
   jne Krok2_22

   write t3
   mov dx,31fh
Krok2_3:     ; II klucz
   in al,dx
   and al,3
   cmp al,1
   jne Krok2_3

   write t5  ; inicjalizacja
   mov dx,31fh
Krok2_32:    ; czekaj na 0
   in al,dx
   and al,3
   cmp al,0
   jne Krok2_32

   write t4
   jmp koniec


; D1=0 i|D0=0
Krok3_0:
   mov ax,31fh
   mov dx,ax
   mov al,1
   out dx,al
   mov cx,16000  ; 1|opóźnienie 0
del:
   loop del
   xor al,al
   out dx,al

; beep
   mov al,07h
   mov ah,0eh
   int 10h

   write t1  ; proszę przyłożyć klucz...
   write t7  ; rysuj pasek

; pasek
   mov ah,0  ; petla 40 sek
   int 1ah   ; Read System Timer
   push dx   ; zachowaj początkową wartość
             ; timera
Krok3_01:
   mov ah,0  ; petla 500 ms
   int 1ah   ; Read System Timer
   push dx   ; zachowaj początkową wartość
             ; timera
   mov ah,0eh
   mov al,' '
   int 10h

   ; odczyt klucza
Krok3_1:
   mov dx,31fh
   in al,dx
   and al,3
   cmp al,2  ; czekaj az (D1=1 i|D0=0)
   jne Krok3_2
   pop dx
   pop dx
   jmp koniec    ; klucz prawidłowy

Krok3_2:
   mov ah,0
   int 1ah
   pop cx    ; do CX stara wartość timera
   push cx
   sub dx,cx ; DX <- nowy odczyt - stary odczyt
   cmp dx,10
   jl Krok3_1
   pop dx    ; wyrównanie stosu pętli 500 ms
             ; koniec petli 500 ms

   mov ah,0
   int 1ah
   pop cx    ; do CX stara wartość timera
   push cx
   sub dx,cx ; DX <- nowy odczyt - stary odczyt
   cmp dx,790    ; ok. 40 sek
   jl Krok3_01
   pop dx    ; wyrównanie stosu pętli 40 sek

   mov ax,31fh
   mov dx,ax
   mov al,2
   out dx,al

; system stop!!! = znak STOP
   xor cl,cl
   xor ch,ch
   push cx
   ; w|ds segment jest OK
NastZnak:
   lea si,ZnakStop
   add si,cx
   lodsb
   cmp al,'$'
   je KonTekst
   mov ah,0eh
   int 10h
   pop cx
   inc cx
   push cx
   jmp NastZnak
KonTekst:
   pop ax
   write t6

; beep x|2
   mov al,07h
   mov ah,0eh
   int 10h

   mov cx,5000   ; opóźnienie
opoz2:
   loop opoz2


   mov al,07h
   mov ah,0eh
   int 10h


zawies0:
   jmp zawies0

Blad1:
   ; wystapil stan D1=1 i|D0=0
   write Blad1_t
zawies:      mov al,07h
   mov ah,0eh
   int 10h
   jmp zawies    ; pętla nieskończona
                 ; zawiesza komputer

koniec:
   retf


;
*****************************************************************
CR EQU 0Dh
LF EQU 0Ah

Copyr        db  CR,LF,'BIOS Extension v.1.0',
CR,LF,'Security system for PC (C) 1997
btc',CR,LF,CR,LF
   db        '#',CR,LF,'Rozszerzenie BIOS
v.1.0',CR,LF,'System zabezpieczenia komputera
(C) 1997 btc',CR,LF,CR,LF,'$'
Blad1_t      db  'Error at address 31Fh',CR,LF
   db        '#Niedozwolony stan bitow
w|31Fh',CR,LF,'$'
t1 db        'Insert key....',CR,LF,CR,LF
   db        '#Prosze przylozyc klucz do
czytnika....',CR,LF,CR,LF,'$'
t2 db        'System not initialized',CR,LF
   db        '   - insert first key'
   db        '#System zabezpieczenia nie zostal
zainicjowany',CR,LF
   db        '   - prosze przylozyc pierwszy
klucz','$'
t3 db        '   - insert second key'
   db        '#   - prosze przylozyc drugi
klucz','$'
t4 db        CR,LF,'Security system is
activated',CR,LF
   db        '#',CR,LF,'System zabezpieczenia
jest aktywny',CR,LF,'$'
t5 db        ' ..initialization',CR,LF
   db        '# ..inicjalizacja',CR,LF,'$'
t6 db        'Computer in standby mode - press
RESET',CR,LF
   db        '#Komputer zostal zablokowany
przez system zabezpieczenia - prosze wcisnac
RESET',CR,LF,'$'
t7 db        '
',CR
   db        '#
',CR,'$'

ZnakStop db  CR,LF,CR,LF,CR,LF
   db        ' TU  ZNAJDUJE SIĘ ZNAK ',CR,LF
   db        ' STOP ZBUDOWANY ZE ZNAKÓW ',CR,LF
   db        ' SEMIGRAFICZNYCH ',CR,LF
   db        ' NIESTETY NIEDRUKOWALNYCH',CR,LF
   db        ' JEGO WYGLĄD ',CR,LF
   db        ' JEST ZAPREZENTOWANY ',CR,LF
   db        ' NA STR. 59 ',CR,LF
   db        '',CR,LF,CR,LF,'$'

       ends code
end start
```