```
                                page    40,132
                        ;_____________________________________________________________________________
                        ;       T0DOS Boot Menu is part of T0DOS. Copyright (C) 2021 Ton Daas
                        ;       T0DOS is free software: you can redistribute it and/or modify
                        ;       it under the terms of the GNU General Public License as published by
                        ;       the Free Software Foundation, either version 3 of the License,
                        ;       or any later version.
                        ;
                        ;       T0DOS is distributed in the hope that it will be useful,
                        ;       but WITHOUT ANY WARRANTY; without even the implied warranty of
                        ;       MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
                        ;       See the GNU General Public License for more details.
                        ;
                        ;       You should have received a copy of the GNU General Public License
                        ;       along with this program.  If not, see <https://www.gnu.org/licenses/>.
                        ;_____________________________________________________________________________
                        ; T0DOS Boot menu. Loaded and executed from T0DOS
                        ; In edit mode config parameters are saved back in first sector of own program.
                        ; It is assumed that minimal DOS file (FCB) support is available.
                        ; The following DOS INT 21h functions are called:
                        ; AH=00h (exit), 0Fh (open), 10h (close), 1Ah (sta), 22h (r-write), 4Ch (exit)
                        ; The following special BIOS function is called:
                        ; INT 13h, AH=19h      ;a tailored command for interaction with T0DOS
0000                    t0dosmnu segment
=                       version equ     '21.9'

= 0008                  bs      equ     08h
= 0009                  tab     equ     09h
= 000D                  cr      equ     0Dh
= 001B                  escape  equ     1Bh

005C                            org     5Ch
005C                    FCB1    label   byte    ;warning! last byte may overlay parameter area

0080                            org     80h
```

```
0080  ???? ????           timer   dw      ?,?
0084  ??                 expired db      ?
0085     08 [             charbuf db      8 dup (?)
                  ??
                      ]


                                ; Check for any parameters on command line
0100                            org     100h
                                assume  cs:t0dosmnu,ds:t0dosmnu, es:t0dosmnu
0100  8C D2                     start:  mov     dx,ss   ;if loaded from T0DOS, SS=0, otherwise SS>0
0102  85 D2                             test    dx,dx   ;are we loaded directly from T0DOS?
0104  74 08                             jz      chk_bf  ;if yes, go and interpret AL value
0106  33 D2                             xor     dx,dx   ;select diskette A:
0108  B4 19                             mov     ah,19h  ;request reboot if started from DOS under T0DOS
010A  CD 13                             int     13h     ;if T0DOS installed, we do not return after int
010C  EB 1E                             jmp     short sav_vm   ;skip, since AL and BP have no meaning

                                ; Check current status of AL=bootflag and reflect in settings
                                ; SS:[BP] points at partition table
010E  B9 8004                   chk_bf: mov     cx,8004h
0111  22 E8                             and     ch,al   ;isolate bootflag
0113  88 2E 08D0 R                      mov     bootfl,ch       ;save current setting
0117  74 13                             jz      sav_vm  ;if not set, use saved defaults
0119  D2 E8                             shr     al,cl   ;convert table index to partition index
011B  24 03                             and     al,3h   ;remove bootflag
011D  FE C0                             inc     al      ;convert to base 1
011F  A2 021F R                         mov     byte ptr def_sel,al    ;overrule default selection
0122  80 0E 0222 R 80                   or      byte ptr sel_atr,80h   ;force select attribute flashing
0127  C6 06 021E R 00                   mov     byte ptr timeout,0     ;disable timer

                                ; Save current video settings on stack and switch to mode 3, page 1
012C  B4 0F                     sav_vm: mov     ah,0Fh  ;get current video mode
012E  CD 10                             int     10h
0130  8A E7                             mov     ah,bh   ;move current page also in AX
```

```
0132  50                        push    ax      ;save current mode and page on stack
0133  3C 07                     cmp     al,7    ;if mode 7 (mono),
0135  74 0D                     je      init_p  ;then keep this mode
0137  3C 03                     cmp     al,3    ;if mode 3 (co80),
0139  74 09                     je      init_p  ;then keep this mode
013B  3C 02                     cmp     al,2    ;if mode 2 (bw80)
013D  74 05                     je      init_p  ;them keep this mode
013F  B8 0003                   mov     ax,3h   ;else switch to mode 3
0142  CD 10                     int     10h     ;set video mode 3 if no compatible mode
0144  B8 0501           init_p: mov     ax,501h ;set to page 1
0147  8A D8                     mov     bl,al   ;copy page # also in bl
0149  CD 10                     int     10h     ;select page 1
014B  B4 03                     mov     ah,3h
014D  CD 10                     int     10h     ;get cursor info
014F  80 CD 20                  or      ch,20h
0152  B4 01                     mov     ah,1h
0154  CD 10                     int     10h     ;turn off cursor

                        ; Clear screen
0156  8A 3E 0220 R              mov     bh,auth_a       ;set bh to background attr
015A  33 C9                     xor     cx,cx   ;set upper left corner to row 0 and col 0
015C  BA 184F                   mov     dx,79+24 shl 8 ;set lower right corner to last char
015F  B8 0600                   mov     ax,0600h
0162  CD 10                     int     10h     ;clear screen by scrolling active page up
0164  86 FB                     xchg    bh,bl   ;set page # and attrib regs
0166  E8 043D R                 call    sh_auth ;do some advertizing
0169  E9 0648 R                 jmp     seldrw

                        ;-----------------------------------------------------------------------------
                        ; Configuration data
016C  30 20 49 42 4D 20 dscr    db      '0 IBM PC-DOS 1.10 (from track 0)  '
      50 43 2D 44 4F 53
      20 31 2E 31 30 20
      28 66 72 6F 6D 20
      74 72 61 63 6B 20
```

```
                30 29 20 20
= 0022                          dscr_ln equ     $-dscr                          ;'
018E  31 20 [                           db      '1 ',dscr_ln-2 dup (' ')      ;needs non blank to prevent loop
                20
                    ]

01B0  32 20 [                           db      '2 ',dscr_ln-2 dup (' ')
                20
                    ]

01D2  33 20 [                           db      '3 ',dscr_ln-2 dup (' ')
                20
                    ]

01F4  34 20 [                           db      '4 ',dscr_ln-2 dup (' ')
                20
                    ]


0216     08 [                   passwrd db      8 dup (' ')
                20
                    ]

021E  00                        timeout db      0
021F  00                        def_sel db      0       ;default selection index

0220  07                        auth_a  db      07h     ;white on black
0221  70                        win_atr db      70h     ;window black on white
0222  0F                        sel_atr db      0Fh     ;select white on black
= 00B7                          bu_size equ     $-dscr  ;area size for backup
                                ;-----------------------------------------------------------------------------
                                ; Fixed data
0223  00 54 30 44 4F 53        fcb     db      0,'T0DOSMNUCOM'
      4D 4E 55 43 4F 4D
022F  54 30 2D 44 4F 53        author  db      'T0-DOS Boot Menu ver: 21.9 (C) Ton Daas'
```

```
                20 42 6F 6F 74 20
                4D 65 6E 75 20 76
                65 72 3A 20 32 31
                2E 39 20 28 43 29
                20 54 6F 6E 20 44
                61 61 73
= 0027                          auth_l  equ     $-author
= 1814                          auth_p  equ     24 shl 8 +(80-auth_l)/2       ;center on last row

0256  20 DA C4 BF 20            win_top db      ' ÚÄ¿ '
025B  20 B3 20 B3 20            win_mid db      ' ³ ³ '
0260  20 C3 C4 B4 20            win_sep db      ' ÃÄ´ '
0265  20 C0 C4 D9 20            win_bot db      ' ÀÄÙ '
= 0004                          win_bor equ     $-win_bot-1

026A  20 43 68 6F 6F 73         sel_hd  db      ' Choose partition                    '
      65 20 70 61 72 74
      69 74 69 6F 6E 20
      20 20 20 20 20 20
      20 20 20 20 20 20
      20 20 20 20 20 20
= 0024                          win_wth equ     $-sel_hd
028E  20 30 2D 34 2C 20         sel_ft  db      ' 0-4, Enter or Space: Start choice  '
      45 6E 74 65 72 20
      6F 72 20 53 70 61
      63 65 3A 20 53 74
      61 72 74 20 63 68
      6F 69 63 65 20 20
02B2  20 18 2F 19 3A 20                 db      ' □/□: Change partition choice       '
      43 68 61 6E 67 65
      20 70 61 72 74 69
      74 69 6F 6E 20 63
      68 6F 69 63 65 20
      20 20 20 20 20 20
02D6  20 45 73 63 3A 20                 db      ' Esc: Request BIOS boot next device '
```

```
                52 65 71 75 65 73
                74 20 42 49 4F 53
                20 62 6F 6F 74 20
                6E 65 78 74 20 64
                65 76 69 63 65 20
02FA  20 54 61 62 3A 20                 db      ' Tab: Edit partition descriptions   '
      45 64 69 74 20 70
      61 72 74 69 74 69
      6F 6E 20 64 65 73
      63 72 69 70 74 69
      6F 6E 73 20 20 20
= 0004                          sel_ftl equ     ($-sel_ft)/win_wth
031E  20 45 6E 74 65 72         sel_pw  db      ' Enter password?:    '
      20 70 61 73 73 77
      6F 72 64 3F 3A 20
      20 20 20
0333  6F 72 20 77 61 69         sel_to  db                          'or wait    sec.'
      74 20 20 20 20 73
      65 63 2E

0342  20 50 61 73 73 77        edt_hd  db      ' Password:           Timeout:    sec.'
      6F 72 64 3A 20 20
      20 20 20 20 20 20
      20 20 54 69 6D 65
      6F 75 74 3A 20 20
      20 20 73 65 63 2E
0366  18 2F 19 3A 20 53        edt_ft  db      '□/□: Select row ³ +/-/0: Select time'
      65 6C 65 63 74 20
      72 6F 77 20 B3 20
      2B 2F 2D 2F 30 3A
      20 53 65 6C 65 63
      74 20 74 69 6D 65
038A  45 6E 74 65 72 3A                 db      'Enter: Edit row ³ Tab: Edit password'
      20 45 64 69 74 20
      72 6F 77 20 B3 20
```

```
                54 61 62 3A 20 45
                64 69 74 20 70 61
                73 73 77 6F 72 64
03AE            28 6C 65 61 64 69               db      '(leading ! or space inhibits choice)'
                6E 67 20 21 20 6F
                72 20 73 70 61 63
                65 20 69 6E 68 69
                62 69 74 73 20 63
                68 6F 69 63 65 29
03D2            46 33 2D 46 38 3A               db      'F3-F8: Colors ³ Esc:Quit ³ F10:Save '
                20 43 6F 6C 6F 72
                73 20 B3 20 45 73
                63 3A 51 75 69 74
                20 B3 20 46 31 30
                3A 53 61 76 65 20
= 0004                          edt_ftl equ     ($-edt_ft)/win_wth

= 0514                          win_pos equ     5 shl 8 + (80-win_wth-win_bor)/2      ;center on screen

= 0616                          win_hdp equ     win_pos+100h+win_bor/2 ;position of header lines
= 0621                          win_epw equ     win_hdp+11
= 0628                          win_spw equ     win_hdp+18
= 0633                          win_to  equ     win_hdp+29
= 0817                          win_des equ     win_hdp+201h

= 0006                          row_des equ     2+high win_hdp ;first row for descriptions
= 000C                          row_ft  equ     6+row_des      ;first row for footer

= 0015                          ofs_bto equ     21
= 0017                          fld_des equ     low win_des
                                ;-----------------------------------------------------------------------------
                                ; display character ds:[si] CX times starting pos. DX, page BH, attrib. BL
03F6                            dspl_ch proc    near
03F6  50                                push    ax
03F7  B4 02                             mov     ah,02h
```

```
0246  03F9  CD 10                    int     10h     ;place cursor
0247  03FB  AC                       lodsb
0248  03FC  B4 09                    mov     ah,09h
0249  03FE  CD 10                    int     10h     ;write cl characters
0250  0400  02 D1                    add     dl,cl
0251  0402  58                       pop     ax
0252  0403  C3                       ret
0253  0404                    dspl_ch endp
0254
0255                          ; display CX lines of window of width=AL with 5 chars from DS:[SI] at pos.=DX
0256  0404                    dspl_ln proc    near
0257  0404  52                       push    dx
0258  0405  51                       push    cx
0259  0406  B9 0001                   mov     cx,1
0260  0409  E8 03F6 R                 call    dspl_ch
0261  040C  E8 03F6 R                 call    dspl_ch
0262  040F  B1 24                     mov     cl,win_wth
0263  0411  E8 03F6 R                 call    dspl_ch
0264  0414  B1 01                     mov     cl,1
0265  0416  E8 03F6 R                 call    dspl_ch
0266  0419  E8 03F6 R                 call    dspl_ch
0267  041C  59                       pop     cx
0268  041D  5A                       pop     dx
0269  041E  FE C6                    inc     dh
0270  0420  BE 025B R                 mov     si,offset win_mid
0271  0423  E2 DF                     loop    dspl_ln
0272  0425  C3                       ret
0273  0426                    dspl_ln endp
0274
0275                          ; display string of CX characters from DS:[SI] starting row DH, row DL
0276                          ; routine to emulate int 10h, function 1300h (not present on old pc's)
0277  0426                    dspltxt proc    near
0278  0426  56                       push    si
0279  0427  8B F5                    mov     si,bp   ;get pointer in si
0280  0429  51                       push    cx
```

```
281      042A  52                        push    dx      ;save cursor position
282      042B  51                  wrs_lp: push    cx
283      042C  B9 0001                     mov     cx,1
284      042F  E8 03F6 R                   call    dspl_ch
285      0432  59                          pop     cx
286      0433  E2 F6                       loop    wrs_lp
287      0435  5A                          pop     dx
288      0436  B4 02                       mov     ah,2h   ;set cursor back to beginning
289      0438  CD 10                       int     10h
290      043A  59                          pop     cx
291      043B  5E                          pop     si
292      043C  C3                          ret
293      043D                      dspltxt endp
294
295      043D                      sh_auth proc    near
296      043D  53                          push    bx
297      043E  8A 1E 0220 R                mov     bl,auth_a
298      0442  BD 022F R                   mov     bp,offset author
299      0445  BA 1814                     mov     dx,auth_p
300      0448  B9 0027                     mov     cx,auth_l
301      044B  E8 0426 R                   call    dspltxt
302      044E  5B                          pop     bx
303      044F  C3                          ret
304      0450                      sh_auth endp
305
306      0450                      sh_win  proc    near
307      0450  BA 0514                     mov     dx,win_pos
308      0453  BE 0256 R                   mov     si,offset win_top
309      0456  B9 0002                     mov     cx,2    ;top and header rows
310      0459  E8 0404 R                   call    dspl_ln
311      045C  BE 0260 R                   mov     si,offset win_sep
312      045F  B9 0006                     mov     cx,6    ;separator and 5 descr. rows
313      0462  E8 0404 R                   call    dspl_ln
314      0465  BE 0260 R                   mov     si,offset win_sep
315      0468  B9 0005                     mov     cx,1+sel_ftl   ;separator and footer rows
```

```
316      046B  E8 0404 R                  call    dspl_ln
317      046E  41                         inc     cx      ;only bottom border row to finish
318      046F  BE 0265 R                  mov     si,offset win_bot
319      0472  E8 0404 R                  call    dspl_ln ;complete select window
320      0475  C3                         ret
321      0476                     sh_win  endp
322
323      0476                     fmt_des proc    near
324      0476  B9 0022                     mov     cx,dscr_ln
325      0479  B2 17                       mov     dl,fld_des
326      047B  8A C6                       mov     al,dh
327      047D  2C 06                       sub     al,row_des
328      047F  F6 E1                       mul     cl
329      0481  05 016C R                   add     ax,offset dscr
330      0484  95                         xchg    bp,ax
331      0485  E8 0426 R                  call    dspltxt ;display part description
332      0488  C3                         ret
333      0489                     fmt_des endp
334
335                               ; calculate remaining time and show on screen. Return with CF if expired
336      0489                     timecal proc
337      0489  52                         push    dx
338      048A  B4 00                      mov     ah,0
339      048C  CD 1A                      int     1Ah     ;read timer, returns DX:CX dword of count (18.2/sec)
340      048E  A1 0080 R                  mov     ax,timer
341      0491  2B C2                      sub     ax,dx
342      0493  8B 16 0082 R               mov     dx,timer+2
343      0497  1B D1                      sbb     dx,cx
344      0499  72 42                      jc      tim_ex  ;timer expired
345      049B  05 0009                    add     ax,9    ;add « sec for rounding
346      049E  B9 0005                    mov     cx,5
347      04A1  F7 E1                      mul     cx
348      04A3  B9 005B                    mov     cx,91
349      04A6  F7 F1                      div     cx      ;al now has remaining seconds in binary
350      04A8  EB 0D                      jmp     short to_0
```

```
                                
04AA  52                    fmt_to: push    dx
04AB  A0 021E R                     mov     al,timeout
04AE  B1 20                         mov     cl,' '
04B0  F6 06 0222 R 80               test    sel_atr,80h
04B5  74 02                         jz      to_dsp
04B7  B1 30                 to_0:   mov     cl,'0'
04B9  BA 0633               to_dsp: mov     dx,win_to
04BC  D4 0A                         aam
04BE  0D 3030                       or      ax,'00'
04C1  80 FC 30                      cmp     ah,'0'
04C4  75 08                         jne     to_hi
04C6  B4 20                         mov     ah,' '
04C8  3C 30                         cmp     al,'0'
04CA  75 02                         jne     to_hi
04CC  8A C1                         mov     al,cl
04CE  86 C4                 to_hi:  xchg    al,ah
04D0  A3 0085 R                     mov     word ptr charbuf,ax
04D3  BD 0085 R                     mov     bp,offset charbuf
04D6  B9 0002                       mov     cx,2    ;characters to write
04D9  E8 0426 R                     call    dspltxt ;display timer
04DC  F8                            clc
04DD  5A                    tim_ex: pop     dx
04DE  C3                            ret
04DF                        timecal endp
                                
04DF                        readtxt proc    near
04DF  B4 02                         mov     ah,02h  ;set cursor position
04E1  CD 10                         int     10h
04E3  51                            push    cx
04E4  B4 03                         mov     ah,03h  ;Read cursor type and position
04E6  CD 10                         int     10h
04E8  80 E5 DF                      and     ch,not 20h      ;Set cursor on mask.
04EB  B4 01                         mov     ah,01h  ;set cursor type
04ED  CD 10                         int     10h
```

```
386      04EF  59                        pop     cx
387      04F0  B4 00                rtwkey: mov     ah,0
388      04F2  CD 16                        int     16h     ;wait and read keystroke
389      04F4  3C 1B                        cmp     al,escape
390      04F6  74 41                        je      rtcanc
391      04F8  3C 0D                        cmp     al,cr
392      04FA  74 3E                        je      rtend
393      04FC  3C 08                        cmp     al,bs
394      04FE  74 1F                        je      rtback
395      0500  3C 20                        cmp     al,' '
396      0502  72 EC                        jb      rtwkey  ;no other control characters permitted
397      0504  77 05                        ja      rtinc   ;jump if non blank entered
400      0506  80 FE 06                     cmp     dh,high win_hdp       ;is this the header row?
401      0509  74 E5                        je      rtwkey  ;space in password not allowed
402      050B  3A D5                rtinc:  cmp     dl,ch   ;last position reached?
403      050D  73 E1                        jnb     rtwkey
404      050F  80 F9 28                     cmp     cl,low win_spw ;password in selection window?
405      0512  75 03                        jne     rtwrc
406      0514  AA                           stosb           ;save in buffer
407      0515  B0 2A                        mov     al,'*'  ;show dummy if in sel menu
408      0517  B4 0E                rtwrc:  mov     ah,0Eh  ;write character to screen, move cursor
409      0519  CD 10                        int     10h
410      051B  FE C2                        inc     dl      ;new screenposition
411      051D  EB D1                        jmp     short rtwkey
412
413      051F  3A D1                rtback: cmp     dl,cl   ;is cursor on start position?
414      0521  76 CD                        jna     rtwkey  ;ignore backspace if so
415      0523  B4 0E                        mov     ah,0Eh  ;back cursor one position
416      0525  CD 10                        int     10h
417      0527  B0 20                        mov     al,' '  ;write space, move cursor
418      0529  CD 10                        int     10h
419      052B  B0 08                        mov     al,bs   ;back cursor one position again
420      052D  CD 10                        int     10h
```

```
0531  80 F9 28                    cmp     cl,low win_spw ;password in selection window?
0534  75 BA                       jne     rtwkey
0536  4F                          dec     di      ;ignore last char in buffer
0537  EB B7                       jmp     short rtwkey

0539  F9                  rtcanc: stc
053A  9C                  rtend:  pushf
053B  51                          push    cx
053C  52                          push    dx
053D  B4 03                       mov     ah,03h  ;Read cursor type and position
053F  CD 10                       int     10h
0541  80 CD 20                    or      ch,20h  ;turn off cursor
0544  B4 01                       mov     ah,01h  ;set cursor type
0546  CD 10                       int     10h
0548  5A                          pop     dx
0549  59                          pop     cx
054A  9D                          popf
054B  C3                          ret
054C                      readtxt endp

054C                      chk_des proc    near
054C  8B F8                       mov     di,ax   ;save al
054E  B4 22                       mov     ah,dscr_ln
0550  F6 E4                       mul     ah
0552  97                          xchg    di,ax   ;restore al
0553  80 BD 016E R 21             cmp     dscr+2[di],'!'
0558  C3                          ret
0559                      chk_des endp

0559                      chk_pw  proc    near
0559  52                          push    dx
055A  BD 031E R                   mov     bp,offset sel_pw
055D  BA 0616                     mov     dx,win_hdp
0560  B9 0015                     mov     cx,ofs_bto
```

```
0563  E8 0426 R                    call    dspltxt ;display header row with password prompt
0566  BF 0085 R                    mov     di,offset charbuf
0569  B2 28                       mov     dl,low win_spw ;position cursor at place to edit
056B  8A CA                       mov     cl,dl   ;save start position in CL
056D  8A EA                       mov     ch,dl
056F  80 C5 08                    add     ch,8    ;last position in CH
0572  E8 04DF R                   call    readtxt
0575  73 0E                       jnc     pw_cr
0577  BD 026A R             pw_esc: mov     bp,offset sel_hd      ;restore normal header
057A  B2 16                       mov     dl,low win_hdp
057C  B9 0024                     mov     cx,win_wth     ;length
057F  E8 0426 R                   call    dspltxt ;display normal header again
0582  F9                          stc
0583  EB 18                       jmp     short pw_ret

0585  B9 008D R             pw_cr:  mov     cx,offset charbuf+8
0588  2B CF                       sub     cx,di   ;remaining unused characters
058A  76 04                       jbe     pw_cmp
058C  B0 20                       mov     al,' '
058E  F3/ AA                      rep stosb      ;blank remaining characters
0590  BE 0216 R              pw_cmp: mov     si,offset passwrd
0593  BF 0085 R                   mov     di,offset charbuf
0596  B9 0008                     mov     cx,8
0599  F3/ A6                      repe cmpsb             ;compare password
059B  75 DA                       jne     pw_esc
059D  5A                    pw_ret: pop     dx
059E  C3                          ret
059F                        chk_pw  endp

                            ; Edit routine to change partition descriptions or password
                            ; On entry:    es:bp=address of string in memory, dh=partition index
                            ;              bh=page number, bl=attribute
                            ; On exit:     al=escape char if edit was canceled, ah=?
                            ;              current screen textrow needs refresh when canceled
059F                        edit    proc    near
```

```
0059F  52                       push    dx
05A0  51                        push    cx
05A1  BF 016E R                 mov     di,offset dscr+2
05A4  8A C6                     mov     al,dh
05A6  2C 06                     sub     al,row_des      ;al has partition index
05A8  B5 22                     mov     ch,dscr_ln
05AA  F6 E5                     mul     ch
05AC  03 F8                     add     di,ax
05AE  80 ED 02                  sub     ch,2
05B1  B2 19                     mov     dl,fld_des+2
05B3  EB 0A                     jmp     short edstrt

05B5  52                editpw: push    dx
05B6  51                        push    cx
05B7  BF 0216 R                 mov     di,offset passwrd
05BA  BA 0621                   mov     dx,win_epw
05BD  B5 08                     mov     ch,length passwrd
05BF  8A CA             edstrt: mov     cl,dl   ;save cursor start position
05C1  02 E9                     add     ch,cl   ;save cursor end position
05C3  8A D5                     mov     dl,ch   ;set cursor to end of text
05C5  FE CA             edflnb: dec     dl      ;point to last character
05C7  3A D1                     cmp     dl,cl   ;is cursor at start position?
05C9  72 0C                     jb      edeot   ;exit if before start position
05CB  B4 02                     mov     ah,02h  ;set cursor position
05CD  CD 10                     int     10h
05CF  B4 08                     mov     ah,08h  ;read character & attribute
05D1  CD 10                     int     10h
05D3  3C 20                     cmp     al,' '  ;is it a space?
05D5  74 EE                     je      edflnb  ;find last non blank
05D7  FE C2             edeot:  inc     dl      ;position cursor at place to edit
05D9  E8 04DF R                 call    readtxt
05DC  72 11                     jc      edcanc
05DE  8A D1                     mov     dl,cl   ;cursor back to first position
05E0  B4 02             edschr: mov     ah,02h
05E2  CD 10                     int     10h     ;set cursor position
```

```
05E4  B4 08                     mov     ah,08h
05E6  CD 10                     int     10h     ;read character
05E8  AA                        stosb           ;save character
05E9  FE C2                     inc     dl      ;increment cursor position
05EB  3A D5                     cmp     dl,ch   ;last character read?
05ED  72 F1                     jb      edschr
05EF  59                edcanc: pop     cx
05F0  5A                        pop     dx
05F1  C3                        ret
05F2                    edit    endp

                        ;Color changing routine. F3-F8
05F2                    color   proc    near
05F2  B1 04                     mov     cl,4
05F4  80 FC 3E                  cmp     ah,3Eh  ;F4 key?
05F7  77 19                     ja      col_fg  ;>F4
05F9  A0 0220 R                 mov     al,auth_a
05FC  72 06                     jb      col_bg  ;F3 key
05FE  04 10                     add     al,10h
0600  24 7F                     and     al,7Fh
0602  EB 06                     jmp     short col_au

0604  D2 C0               col_bg: rol     al,cl
0606  04 10                     add     al,10h
0608  D2 C8                     ror     al,cl
060A  A2 0220 R           col_au: mov     auth_a,al
060D  E8 043D R                 call    sh_auth
0610  EB 35                     jmp     short col_ex

0612  80 FC 40             col_fg: cmp     ah,40h  ;F6 key?
0615  77 17                     ja      col_hi  ;>F6
0617  72 08                     jb      col_tx  ;F5 key
0619  80 C3 10                  add     bl,10h
061C  80 E3 7F                  and     bl,7Fh
061F  EB 07                     jmp     short col_wa
```

```
561
562      0621  D2 C3                col_tx: rol     bl,cl
563      0623  80 C3 10                    add     bl,10h
564      0626  D2 CB                        ror     bl,cl
565      0628  88 1E 0221 R          col_wa: mov     win_atr,bl
566      062C  EB 19                        jmp     short col_ex
567
568      062E  A0 0222 R             col_hi: mov     al,sel_atr
569      0631  80 FC 42                     cmp     ah,42h  ;F8 key?
570      0634  72 08                        jb      col_ht  ;F7
571      0636  D0 C0                        rol     al,1    ;save flashing bit
572      0638  04 20                        add     al,20h
573      063A  D0 C8                        ror     al,1    ;restore flashing bit
574      063C  EB 06                        jmp     short col_sa
575
576      063E  D2 C0                 col_ht: rol     al,cl
577      0640  04 10                        add     al,10h
578      0642  D2 C8                        ror     al,cl
579      0644  A2 0222 R             col_sa: mov     sel_atr,al
580      0647  C3                    col_ex: ret
581      0648                        color   endp
582                                  ;-----------------------------------------------------------------------------
583                                  ; Draw menu an fill in fixed header and footer data
584
585      0648  8A 1E 0221 R          seldrw: mov     bl,win_atr
586      064C  E8 0450 R                     call    sh_win  ;draw empty menu except bottom row
587      064F  BD 026A R                     mov     bp,offset sel_hd
588      0652  B9 0024                       mov     cx,win_wth
589      0655  BA 0616                       mov     dx,win_hdp
590      0658  E8 0426 R                     call    dspltxt ;display header row
591      065B  BD 028E R                     mov     bp,offset sel_ft
592      065E  B6 0C                         mov     dh,row_ft
593      0660  E8 0426 R             sfd_n2: call    dspltxt
594      0663  03 E9                         add     bp,cx   ;increment to next data
595      0665  FE C6                         inc     dh
```

```
0667  80 FE 10                    cmp     dh,row_ft+sel_ftl
066A  72 F4                      jb      sfd_n2

                                 ; Calculate elapsed system-timer value and display header and time message
066C  32 C0                      xor     al,al
066E  02 06 021E R               add     al,timeout      ;get initial timeout value
0672  A2 0084 R                  mov     expired,al
0675  74 2A                      jz      sel_dm
0677  98                         cbw
0678  B9 005B                    mov     cx,91
067B  F7 E1                      mul     cx
067D  B9 0005                    mov     cx,5
0680  F7 F1                      div     cx
0682  8B F8                      mov     di,ax
0684  B4 00                      mov     ah,0
0686  CD 1A                      int     1Ah     ;get system timer in CX:DX
0688  03 D7                      add     dx,di
068A  83 D1 00                   adc     cx,0
068D  89 16 0080 R               mov     timer,dx        ;save elapsed timer value
0691  89 0E 0082 R               mov     timer+2,cx
0695  BD 0333 R                  mov     bp,offset sel_to
0698  B9 000F                    mov     cx,win_wth-ofs_bto
069B  BA 062B                    mov     dx,win_hdp+ofs_bto
069E  E8 0426 R                  call    dspltxt ;display header timeout message

                                 ; Display menu items
06A1  B6 06                      sel_dm: mov     dh,row_des      ;first row and char for selection list
06A3  E8 0476 R                  dminxt: call    fmt_des
06A6  FE C6                              inc     dh      ;next display row
06A8  80 FE 0B                           cmp     dh,row_des+5
06AB  72 F6                              jb      dminxt

                                 ; set cursor and hilite selection
06AD  B6 06                              mov     dh,row_des      ;index of first part
06AF  02 36 021F R                       add     dh,def_sel      ;add default partition index
```

```
06B3  53                  sel_sc: push    bx      ;save list attribute and page
06B4  8A 1E 0222 R                mov     bl,sel_atr      ;select attribute
06B8  E8 0476 R                   call    fmt_des ;redisplay with hilite
06BB  5B                          pop     bx      ;restore list attribute
06BC  80 3E 0084 R 00             cmp     expired,0
06C1  74 59                       je      sel_gk

                          ; Wait timeout while no key pressed
06C3  B4 01               sel_wk: mov      ah,1   ;read status
06C5  CD 16                       int     16h
06C7  75 40                       jnz     sel_bt
06C9  E8 0489 R                   call    timecal
06CC  73 F5                       jnc     sel_wk  ;loop while no timeout yet
06CE  8A C6               sel_cr: mov     al,dh
06D0  2C 06                       sub     al,row_des      ;al now has partition table index
06D2  EB 6F 90                    jmp     sel_pn

06D5  80 FE 06            sel_up: cmp     dh,row_des      ;are we on first row?
06D8  76 42                       jbe     sel_gk  ;if yes, ignore keystroke
06DA  E8 0476 R                   call    fmt_des ;remove hilite from current
06DD  FE CE               up_up:  dec     dh      ;move cursor up one row
06DF  8A C6                       mov     al,dh
06E1  2C 06                       sub     al,row_des
06E3  E8 054C R                   call    chk_des ;compare first char of description with '!'
06E6  77 CB                       ja      sel_sc  ;if yes, go and set cursor
06E8  80 FE 06                    cmp     dh,row_des
06EB  77 F0                       ja      up_up
06ED  EB 08                       jmp     short dn_dn

06EF  80 FE 0A            sel_dn: cmp     dh,4+row_des
06F2  73 28                       jae     sel_gk
06F4  E8 0476 R                   call    fmt_des ;remove hilite
06F7  FE C6               dn_dn:  inc     dh
06F9  8A C6                       mov     al,dh
06FB  2C 06                       sub     al,row_des
```

```
06FD  E8 054C R                   call   chk_des ;compare first char of description with '!'
0700  77 B1                       ja     sel_sc
0702  80 FE 0A                    cmp    dh,4+row_des
0705  72 F0                       jb     dn_dn
0707  EB D4                       jmp    short up_up

0709  52                    sel_bt: push   dx      ;save screenrow
070A  BE 025B R                   mov    si,offset win_mid      ;point at char to fill area with
070D  BA 062B                     mov    dx,win_hdp+ofs_bto     ;position to start blanking
0710  B9 000F                     mov    cx,win_wth-ofs_bto     ;size of area to fill
0713  E8 03F6 R                   call   dspl_ch ;clear wait timeout message
0716  C6 06 0084 R 00             mov    expired,0      ;flag timeout canceled
071B  5A                          pop    dx      ;restore screenrow
071C  B4 00                 sel_gk: mov    ah,0    ;keyboard read
071E  CD 16                       int    16h
0720  3D 4800                     cmp    ax,72 shl 8    ;is up key pressed?
0723  74 B0                       je     sel_up
0725  3D 5000                     cmp    ax,80 shl 8    ;is down key pressed?
0728  74 C5                       je     sel_dn
072A  3C 0D                       cmp    al,cr   ;is enter pressed?
072C  74 A0                       je     sel_cr
072E  3C 20                       cmp    al,' '  ;is space pressed?
0730  74 9C                       je     sel_cr
0732  3C 1B                       cmp    al,escape      ;escape pressed?
0734  F9                          stc
0735  74 23                       je     sel_ex
0737  3C 09                       cmp    al,tab  ;is tab key pressed
0739  74 22                       je     sel_tb
073B  3C 34                       cmp    al,'4'
073D  77 DD                       ja     sel_gk
073F  2C 30                       sub    al,'0'
0741  72 D9                       jc     sel_gk
0743  74 07                 sel_pn: jz     sel_0   ;0 key pressed
0745  E8 054C R                   call   chk_des ;compare first char of description with '!'
0748  77 10                       ja     sel_ex
```

```
701      074A  EB D0                        jmp     short sel_gk
702
703      074C  80 3E 0216 R 20       sel_0:  cmp     passwrd,' '
704      0751  74 07                         je      sel_ex
705      0753  50                            push    ax
706      0754  E8 0559 R                     call    chk_pw
707      0757  58                            pop     ax
708      0758  72 C2                         jc      sel_gk
709      075A  E9 089D R             sel_ex: jmp     run_ex
710
711      075D  80 3E 0216 R 20       sel_tb: cmp     passwrd,' '
712      0762  74 05                         je      savebu
713      0764  E8 0559 R                     call    chk_pw
714      0767  72 B3                         jc      sel_gk
715
716                                  ; Save backup of current configuration
717      0769  BE 016C R             savebu: mov     si,offset dscr
718      076C  BF 08E3 R                     mov     di,offset bu_area
719      076F  B9 00B7                       mov     cx,bu_size
720      0772  F3/ A4                        rep movsb       ;backup current configuration
721
722                                  ;-Edit mode------------------------------------------------------------------------
723                                  ; Show edit window
724      0774  E8 0450 R             txt_mu: call    sh_win  ;draw empty window
725                                  ;       mov     si,offset win_bot
726                                  ;       dec     dh      ;start one row higher
727                                  ;       inc     cx      ;only bottom row
728                                  ;       call    dspl_ln
729                                  ;       mov     cl,win_wth+win_bor
730                                  ;       mov     bl,auth_a
731                                  ;       call    dspl_ch ;clear unused row with char CH at pos DX
732                                  ;       mov     bl,win_atr
733
734                                  ; Fill in fixed data
735      0777  BD 0342 R                     mov     bp,offset edt_hd
```

```
736      077A  BA 0616                      mov     dx,win_hdp
737      077D  B9 0024                      mov     cx,win_wth
738      0780  E8 0426 R                    call    dspltxt
739      0783  B6 0C                        mov     dh,row_ft
740      0785  83 C5 24                txt_n2: add     bp,win_wth
741      0788  E8 0426 R                    call    dspltxt
742      078B  FE C6                        inc     dh
743      078D  80 FE 10                     cmp     dh,row_ft+edt_ftl
744      0790  72 F3                        jb      txt_n2
745
746                                    ; Display password and timeout values
747      0792  BD 0216 R                    mov     bp,offset passwrd
748      0795  BA 0621                      mov     dx,win_epw
749      0798  B9 0008                      mov     cx,8
750      079B  E8 0426 R                    call    dspltxt
751      079E  E8 04AA R                    call    fmt_to
752
753                                    ; Display menu items
754      07A1  B6 06                        mov     dh,row_des      ;first row for partition info
755      07A3  E8 0476 R               cvttbl: call    fmt_des
756      07A6  FE C6                        inc     dh      ;next display row
757      07A8  80 FE 0A                     cmp     dh,row_des+4
758      07AB  76 F6                        jbe     cvttbl
759
760                                    ; Set cursor to initial field of partition selected
761      07AD  B6 06                        mov     dh,row_des      ;index of first part
762      07AF  02 36 021F R                 add     dh,def_sel      ;add default partition index
763
764                                    ; Highlite cursor field
765      07B3  53                      mnu_hs: push    bx      ;save list attribute and page
766      07B4  8A 1E 0222 R                 mov     bl,sel_atr      ;select attribute
767      07B8  E8 0476 R                    call    fmt_des
768      07BB  5B                           pop     bx      ;restore list attribute
769      07BC  EB 3F                        jmp     short mnu_gk
770
```

```
07BE  3D 3D00              mnu_tf: cmp     ax,3D00h        ;key F3?
07C1  72 11                        jb      mnu_tu
07C3  3D 4200                      cmp     ax,4200h        ;key F8?
07C6  77 0C                        ja      mnu_tu
07C8  80 EE 06                     sub     dh,row_des
07CB  88 36 021F R                 mov     def_sel,dh
07CF  E8 05F2 R                    call    color
07D2  EB A0                        jmp     txt_mu

07D4  3D 4800              mnu_tu: cmp     ax,4800h        ;up key pressed?
07D7  75 0C                        jne     mnu_td
07D9  80 FE 06                     cmp     dh,row_des
07DC  76 1F                        jbe     mnu_gk
07DE  E8 0476 R                    call    fmt_des
07E1  FE CE                        dec     dh
07E3  EB CE                        jmp     short mnu_hs

07E5  3D 5000              mnu_td: cmp     ax,5000h        ;down key pressed?
07E8  75 0C                        jne     mnu_cr
07EA  80 FE 0A                     cmp     dh,row_des+4
07ED  73 0E                        jnb     mnu_gk
07EF  E8 0476 R                    call    fmt_des
07F2  FE C6                        inc     dh
07F4  EB BD                        jmp     short mnu_hs

07F6  3C 0D                mnu_cr: cmp     al,cr   ;Enter key?
07F8  75 49                        jne     mnu_es
07FA  E8 059F R                    call    edit
07FD  B4 00                mnu_gk: mov     ah,0    ;keyboard read
07FF  CD 16                        int     16h
0801  3C 09                        cmp     al,tab  ;tab key pressed?
0803  75 B9                        jne     mnu_tf
0805  E8 05B5 R                    call    editpw
0808  EB F3                        jmp     short mnu_gk
```

```
806      080A  8A 26 021E R          chk_to: mov     ah,timeout
807      080E  3C 2B                          cmp     al,'+'
808      0810  74 23                          je      to_inc
809      0812  3C 2D                          cmp     al,'-'
810      0814  74 14                          je      to_dec
811      0816  3C 30                          cmp     al,'0'
812      0818  75 E3                          jne     mnu_gk
813      081A  80 0E 0222 R 80                or      sel_atr,80h     ;set flashing bit on
814      081F  B4 00                          mov     ah,0
815      0821  88 26 021E R          to_ext: mov     timeout,ah
816      0825  E8 04AA R             to_fmt: call    fmt_to
817      0828  EB 89                          jmp     short mnu_hs
818
819      082A  80 26 0222 R 7F       to_dec: and     sel_atr,7Fh
820      082F  FE CC                          dec     ah
821      0831  79 EE                          jns     to_ext
822      0833  EB F0                          jmp     short to_fmt
823
824      0835  80 26 0222 R 7F       to_inc: and     sel_atr,7Fh
825      083A  FE C4                          inc     ah
826      083C  80 FC 64                       cmp     ah,100
827      083F  72 E0                          jb      to_ext
828      0841  EB E2                          jmp     short to_fmt
829
830                                  ; restore configuration from backup and return to selection menu
831      0843  3C 1B                 mnu_es: cmp     al,escape
832      0845  75 0E                          jne     mnu_10
833      0847  BE 08E3 R                      mov     si,offset bu_area
834      084A  BF 016C R                      mov     di,offset dscr
835      084D  B9 00B7                        mov     cx,bu_size
836      0850  F3/ A4                         rep movsb
837      0852  E9 0648 R                      jmp     seldrw
838
839      0855  3D 4400               mnu_10: cmp     ax,4400h        ;key F10?
840      0858  75 B0                          jne     chk_to
```

```
841      085A  8A C6                        mov     al,dh
842      085C  2C 06                        sub     al,row_des
843      085E  74 9D                        jz      mnu_gk
844      0860  E8 054C R                    call    chk_des ;compare first char of selection with '!'
845      0863  76 98                        jbe     mnu_gk  ;do not allow blank or specials as default selection
846      0865  A2 021F R                    mov     def_sel,al
847
848                                  ; save configuration data to program file
849      0868  BA 0100 R                    mov     dx,offset start        ;set to start of our code
850      086B  B4 1A                        mov     ah,1Ah
851      086D  CD 21                        int     21h     ;set disk tranfer address to this code
852      086F  BA 005C R                    mov     dx,offset fcb1
853      0872  BE 0223 R                    mov     si,offset fcb
854      0875  8B FA                        mov     di,dx
855      0877  B9 000C                      mov     cx,12
856      087A  F3/ A4                       rep movsb
857      087C  B4 0F                        mov     ah,0Fh
858      087E  CD 21                        int     21h     ;open file
859      0880  0A C0                        or      al,al
860      0882  75 10                        jnz     fl_clo
861      0884  33 C0                        xor     ax,ax
862      0886  A3 007D R                    mov     word ptr fcb1+33,ax    ;set relative record number to 0
863      0889  A2 007F R                    mov     fcb1+35,al     ;uses only 1 byte, second byte overlays 80h
864      088C  C7 06 006A R 0200            mov     word ptr fcb1+14,512   ;set record size rounded to sector size
865      0892  B4 22                        mov     ah,22h  ;do random write of record unbuffered
866      0894  CD 21                  fl_clo: int     21h
867      0896  B4 10                        mov     ah,10h  ;close file
868      0898  CD 21                        int     21h
869      089A  E9 0648 R                    jmp     seldrw
870
871                                  ; restore original video settings
872      089D  73 02                  run_ex: jnc     rstvid
873      089F  B0 10                        mov     al,10h  ;selection was Esc, so use an undefined partition
874      08A1  5B                     rstvid: pop     bx      ;get old video mode and page
875      08A2  50                           push    ax      ;save selection
```

```
08A3  8A C7                     mov     al,bh   ;page to al
08A5  B4 05                     mov     ah,5
08A7  CD 10                     int     10h     ;select old page
08A9  B4 03                     mov     ah,03h  ;Read cursor type.
08AB  CD 10                     int     10h
08AD  80 E5 DF                  and     ch,not 20h      ;Set cursor on mask.
08B0  B4 01                     mov     ah,01h          ;set cursor type
08B2  CD 10                     int     10h
08B4  8A C3                     mov     al,bl   ;get old video mode
```

```
08B6  0C 80                     or      al,80h  ;do not clear video
08B8  B4 00                     mov     ah,0    ;set video mode
08BA  CD 10                     int     10h
08BC  58                skip_v: pop     ax      ;restore selection value in al

                        ; Return request to T0DOS to start requested partition
                        ; On exit with int 21, AH=4C, return code in AL:
                        ; bits 3-0 partition, if bit 7=1 then bit 6= bootflag else use default
08BD  8C D2                     mov     dx,ss   ;if loaded from T0DOS, SS=0
08BF  85 D2                     test    dx,dx   ;are we loaded directly from T0DOS?
08C1  75 1C                     jnz     go_dos  ;exit without boot if loaded under DOS
08C3  FE C8                     dec     al      ;make selection 0 origin
08C5  78 18                     js      go_dos  ;was selection 0? then exit to DOS
08C7  8A 26 0222 R              mov     ah,sel_atr      ;get (blinking=) new bootflag value in bit 7
08CB  80 E4 80                  and     ah,80h  ;isolate bit 7
08CE  80 FC 00                  cmp     ah,0    ;need bootflag to be set/reset?
08D0                            org     $-1
08D0  ??                bootfl  db      ?
08D1  74 05                     je      def_bt  ;if bit 7 is changed, we need to explitly change it
08D3  F9                        stc             ;set request bit
```

```
08D4  D0 DC                     rcr     ah,1    ;rotate bits in place
08D6  0A C4                     or      al,ah   ;combine in AL
08D8  B4 4C             def_bt: mov     ah,4Ch  ;exit with return code
08DA  CD 21                     int     21h     ;if request is honoured, we will not return
08DC  E9 012C R                 jmp     sav_vm  ;in case the request is not honoured (not bootable)

08DF  B4 00             go_dos: mov     ah,0
08E1  CD 21                     int     21h

                        ; backup buffer
08E3     B7 [           bu_area db      bu_size dup (?)
                  ??
                      ]

099A                    t0dosmnu ends
                                end     start
```