```
                                page    40,132
                        ;_____________________________________________________________________
                        ;       TBOOT is part of Partition Table Tiny Editor (PTTE).
                        ;       Copyright (C) 2021 A.C.M. Daas <http://daas.info>
                        ;       Partition Table Tiny Editor 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.
                        ;
                        ;       This program 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/>.
                        ;_____________________________________________________________________
                        ; Basic Master Boot Record code with conditional use of extended int 13h.
                        ; It does some more validity checks than standard on an active CHS partition.

                        ; Master boot record routine at CYL=0, HEAD=0, SECT=1
                        ; Upon execution reg values are: DL=drive, IP=7C00h, CS=0000h
0000                    tboot   segment
                                assume  cs:tboot,ds:tboot

7C00                            org     7C00h
= 7C00                  loadadr equ     $
0600                            org     600h
                        ; Initialize stack
0600  33 C9             mbr:    xor     cx,cx
0602  FA                        cli             ;prevent interrupts during stack relocate
0603  8E D1                    mov     ss,cx   ;set stack segment
0605  BC 7C00 R                 mov     sp,offset loadadr      ;set stack to before this area
0608  FB                        sti             ;allow interrupts again
```

```
                                ; Copy this code from 7C00h to 0600h, to free up area for chain boot
0609  8E D9                            mov     ds,cx   ;set source segment
060B  8B F4                            mov     si,sp   ;and offset
060D  8E C1                            mov     es,cx   ;set destination segment
060F  BF 0600 R                        mov     di,offset mbr  ;and offset
0612  B5 01                            mov     ch,1h   ;256 words
0614  FC                               cld             ;increment string operations
0615  F3/ A5                           rep movsw       ;move this code from address 7C00h to 600h
0617  E9 905D R                        jmp     near ptr cont+(mbr-loadadr)   ;near jump will do nicely

061A  43 6F 70 79 72 69                db      'Copyright (C) 2021 Ton Daas',0
      67 68 74 20 28 43
      29 20 32 30 32 31
      20 54 6F 6E 20 44
      61 61 73 00

                                ; partition CHS type list
= 0001                          fat12   equ     01h     ;11h if hidden, <32Mb
= 0004                          fat16   equ     04h     ;14h if hidden, >32Mb <500Mb
= 0005                          extend  equ     05h     ;15h if hidden, extended partition
= 0006                          fat16b  equ     06h     ;16h if hidden, >32Mb <2Gb
= 000B                          fat32   equ     0Bh     ;1Bh if hidden, <2Gb
= 0010                          hidden  equ     10h     ;hidden bit for above types
= 0080                          bootflg equ     80h

                                ; types that have hidden equivalent
0636  01 04 05 06 0B            dostype db      fat12,fat16,extend,fat16b,fat32      ;types that use chs value
= 0005                          chslen  equ     $-dostype

                                ; Subroutine to convert CHS to LBA and subtract first LBA
                                ; On entry AL= Head, DX= Cyl & Sector, CX= tracksize, BP= cylsize
                                ; On exit AX:DX= sector difference with first LBA, CX and BP preserved
063B                            chs2lba proc    near
```

```
                                        mul     cl      ;multiply with # sectors/track (result fits word)
                                        xchg    di,ax   ;mov # full track sectors to DI
                                        mov     al,dl   ;copy low byte cyl & sector (has high cyl bits)
                                        mov     ah,0
                                        shl     ax,1    ;shift 2 cyl bits into AH
                                        shl     ax,1
                                        mov     al,dh   ;get low byte cyl in AL
                                        and     dx,3Fh  ;isolate sector value
                                        dec     dx      ;make first sector base 0
                                        add     di,dx   ;add to full track sectors
                                        mul     bp      ;convert Cyl to LBA in AX:DX (may require 3 bytes)
                                        add     ax,di   ;add converted head+sector
                                        adc     dx,0    ;take care of carry
                                        sub     ax,[si+8]       ;subtrack first LBA
                                        sbb     dx,[si+10]      ;as dword
                                        ret
                                chs2lba endp

                                ; Check bootflags and look for an active partition
                                cont:   mov     di,offset table
                                        mov     cx,4
                                        mov     ax,bootflg      ;bootflag in AL, AH=0
                                check:  or      ah,[di] ;check empty bootflag field
                                        jz      next    ;active flag clear?
                                        xor     al,ah   ;check valid bootflag and make future flags invalid
                                        jnz     inval   ;invalid bootflag field?
                                        mov     si,di   ;save table entry offset
                                        cbw             ;clear ah again
                                next:   lea     di,[di+16]
                                        loop    check   ;not all 4 partitions done?
                                        test    al,bootflg      ;if an active partition was found,
                                        jz      chktyp  ;then skip exit to basic and continue

                                ; Start ROM Basic or return to PC BIOS for next boot device
                                        int     18h     ;exit to ROM Basic
```

```
                                ; Analize partition for CHS type
067C  8A 44 04                  chktyp: mov     al,[si+4]       ;get partition type, is it unused?
067F  84 C0                             test    al,al
0681  74 52                             jz      noboot  ;exit, as unused partition is not bootable
0683  24 EF                             and     al,not hidden  ;unhide if hidden CHS type
0685  BF 0636 R                         mov     di,offset dostype
0688  B1 05                             mov     cl,chslen
068A  F2/ AE                            repne   scasb   ;if type requires CHS boot,
068C  74 5E                             je      rdchs   ;then continue load with CHS

                                ; For all other types check for extended int13 support
068E  BB 55AA                           mov     bx,55AAh        ;fill with request signature
0691  B4 41                             mov     ah,41h  ;get extended int 13 support info; DL still has drive
0693  CD 13                             int     13h
0695  72 55                             jc      rdchs   ;extension not found
0697  81 FB AA55                        cmp     bx,0AA55h       ;signature, AH=major version, DH=extension ver.
069B  75 4F                             jne     rdchs   ;requested support not installed
069D  F6 C1 01                          test    cl,01h  ;bit0=1 if int13,AH=42h supported
06A0  74 4A                             jz      rdchs   ;API subset not supported

                                ;Read bootrecord with extended int13 using LBA value
06A2  8B DC                             mov     bx,sp   ;get bootsector load address
06A4  B9 0005                           mov     cx,5    ;set retry count=5
06A7  56                        rtrylb: push    si      ;save si
06A8  33 C0                             xor     ax,ax
                                ;build address request packet
06AA  50                                push    ax      ;sector LBA 4th word (=0)
06AB  50                                push    ax      ;sector LBA 3rd word (=0)
06AC  FF 74 0A                          push    [si+10] ;sector LBA 2nd word
06AF  FF 74 08                          push    [si+8]  ;sector LBA low word
06B2  06                                push    es      ;set buffer segment
06B3  53                                push    bx      ;and buffer offset
06B4  40                                inc     ax      ;set sector count=1
06B5  50                                push    ax      ;number of sectors (max.(7F)
```

```
141      06B6  B0 10                        mov     al,10h  ;packet size
142      06B8  50                          push    ax      ;high byte reserved (=0)
143      06B9  8B F4                       mov     si,sp   ;DS:SI points to request address packet
144      06BB  B4 42                       mov     ah,42h  ;extended disk read; DL has drive number
145      06BD  CD 13                       int     13h
146      06BF  72 04                       jc      skp_ck
147                                  ; Check sector count read, as C is not set if sector not found error
148      06C1  83 7C 02 01                 cmp     word ptr[si+2],1       ;also need to check actual count
149      06C5  8D 64 0E               skp_ck: lea     sp,[si+14]     ;purge address request packet-1w from stack
150      06C8  58                          pop     ax      ;restore AX with last word of packet (=0)
151      06C9  5E                          pop     si      ;restore initial SI
152      06CA  73 73                       jnc     readok  ;if count < 1 then return C=1, else C=0
153      06CC  CD 13                       int     13h     ;reset drive
154      06CE  E2 D7                       loop    rtrylb  ;try again until count exhausted
155      06D0  BE 0778 R              rdfail: mov     si,offset err_msg
156      06D3  EB 0F                       jmp     short tty
157
158      06D5  BE 0798 R              noboot: mov     si,offset mis_msg
159      06D8  EB 0A                       jmp     short tty
160
161      06DA  BE 0758 R              inval:  mov     si,offset inv_msg
162      06DD  AC                          lodsb
163
164                                  ; routine to write message text to screen
165                                  ; entrypoint is tty (or wrtty if AL already has first character to write)
166                                  ; on entry ds:si points to message to write (null terminates routine)
167      06DE  B3 07                  wrtty:  mov     bl,7    ;white
168      06E0  B4 0E                       mov     ah,0Eh  ;write teletype to active page
169      06E2  CD 10                       int     10h     ;AL=character, BL=foreground color
170      06E4  AC                     tty:    lodsb
171      06E5  3C 00                       cmp     al,0
172      06E7  75 F5                       jnz     wrtty   ;end on nul
173      06E9  4E                          dec     si
174      06EA  EB F8                       jmp     short tty      ;infinite loop on last 0
175
```

```
                                ; Get traditional drive parameters and define cylsize and tracksize from it
06EC  52                        rdchs:  push    dx      ;save drive
06ED  06                                push    es      ;save ES as some BIOS will destroy it
06EE  B4 08                             mov     ah,8h   ;get drive parameters
06F0  CD 13                             int     13h     ;DL= # drives, DH= last head, CX= last cyl/sec
06F2  07                                pop     es
06F3  5B                                pop     bx      ;restore drive in BL
06F4  8A C6                             mov     al,dh   ;get last head in AL
06F6  81 E1 003F                        and     cx,3Fh  ;filter track size in CX
06FA  F6 E1                             mul     cl      ;get cylinder size less one track
06FC  03 C1                             add     ax,cx   ;add one track to get total cylinder size
06FE  95                                xchg    bp,ax   ;move cylinder size to BP

                                ; Convert first CHS to LBA and check with first LBA
06FF  8A 44 01                          mov     al,[si+1]       ;get partition first head
0702  8B 54 02                          mov     dx,[si+2]       ;get partition first cyl & sec
0705  8A F8                             mov     bh,al   ;save first head in BH, BL still has drive
0707  E8 063B R                         call    chs2lba
070A  0B C2                             or      ax,dx   ;if not both words 0 (LBA values unequal),
070C  75 CC                             jnz     inval   ;then quit

                                ; Convert last CHS to LBA and subtract with first LBA
070E  8A 44 05                          mov     al,[si+5]       ;get partition last head
0711  8B 54 06                          mov     dx,[si+6]       ;get partition last cyl & sec
0714  E8 063B R                         call    chs2lba         ;compute LBA size-1
0717  72 C1                             jb      inval   ;if negative amount then quit

                                ; Check against LBA size (may be above CHS size if out of range for last CHS)
0719  2B 44 0C                          sub     ax,[si+12]      ;dword result should have negative value
071C  1B 54 0E                          sbb     dx,[si+14]
071F  73 B9                             jnb     inval   ;if CHS size-1 not below LBA size then quit

                                ; Read bootrecord of active partition using CHS values
0721  8B D3                             mov     dx,bx   ;get saved drive and first head in DX
0723  8B 4C 02                          mov     cx,[si+2]       ;set CX to first sector & cyl
```

```
0726  8B DC                        mov     bx,sp   ;set ES:BX to buffer address 7C00h

                                  ; Read AL mod 10 sector
                                  ; AL/10= function (2-read), AL mod 10= sector count
0728  B0 15                        mov     al,21   ;read partition bootsector
072A  B4 08                        mov     ah,100h shr 5  ;initialize retry count to 5
072C  8B F8                     int13r: mov     di,ax   ;save request and current retry count
072E  D4 0A                        aam             ;split value in request in AH and sector count in AL
0730  CD 13                        int     13h     ;do disk request
0732  73 0B                        jnc     readok  ;if no error, continue validity check
0734  B4 00                        mov     ah,0    ;else reset disk system
0736  CD 13                        int     13h     ;perform disk reset
0738  97                          xchg    ax,di   ;reload request and retry count
0739  D0 E4                        shl     ah,1    ;do untill bit shifts out into carry
073B  73 EF                        jnc     int13r  ;if no carry yet, then retry
073D  EB 91                        jmp     short rdfail

                                  ; Check for valid boot record signature
073F  81 BF 01FE AA55           readok: cmp     word ptr[bx+(bootid-mbr)],0AA55h
0745  75 8E                        jne     noboot
0747  FF D3                        call    bx      ;execute partitions boot record
                                  ;upon exit DS:SI points to booted partition table entry
0749  E9 067C R                    jmp     chktyp

074C     0C [                      db      158h-($-mbr) dup (0)   ;fill unused space
                  00
                     ]

0758  49 6E 76 61 6C 69        inv_msg db      'Invalid partition table'
      64 20 70 61 72 74
      69 74 69 6F 6E 20
      74 61 62 6C 65
076F     09 [                      db      32-($-inv_msg) dup (0) ;allow enough space for message patching
                  00
                     ]
```

```
246
247     0778  45 72 72 6F 72 20      err_msg db     'Error loading operating system'
248           6C 6F 61 64 69 6E
249           67 20 6F 70 65 72
250           61 74 69 6E 67 20
251           73 79 73 74 65 6D
252     0796     02 [                        db     32-($-err_msg) dup (0) ;allow enough space for message patching
253                     00
254                        ]
255
256     0798  4D 69 73 73 69 6E      mis_msg db     'Missing operating system'
257           67 20 6F 70 65 72
258           61 74 69 6E 67 20
259           73 79 73 74 65 6D
260     07B0     05 [                        db     1B5h-($-mbr) dup (0)   ;fill unused space
261                     00
262                        ]
263
264     07B5  58 78 98                       db     low offset inv_msg,low offset err_msg,low offset mis_msg
265     07B8     06 [                NT_sign db     6 dup (0)      ;Windows NT signature
266                     00
267                        ]
268
269     07BE     40 [                table   db     64 dup (0)
270                     00
271                        ]
272
273     07FE  AA55                   bootid  dw     0AA55h
274     0800                         tboot   ends
275                                          end
```