# GRUB Stage1

```
7C00 EB48          JMP     7C4A          ; Jump (short) over BPB data
7C02 90            NOP                   ;  area to main body of code.

7C4A FA            CLI

; In the 0.94 and 0.95 code, an OR  DL,80 instruction is inserted here
; when GRUB is installed in an MBR as a "workaround for buggy BIOSes.."
; which don't pass the boot drive byte correctly. If GRUB is installed
; as the Linux Boot Sector, a value of 00 is used instead of 80, which
; effectively makes it a NOP.  This code (80 CA 80) causes all offsets
; after it to shift by 3 bytes, so all relative jumps below would be
; different for these versions; even though its the same code!
;
; For the GNU GRUB 0.97 code, its programmers substituted the following
; as their "workaround for buggy BIOSes.." using the test  dl,0x80
; instruction, etc. (as shown here):
; 7C4B  90          nop               ; These 'nops' are prob. for
; 7C4C  90          nop               ;   anticipated future changes!
; 7C4D F6C280       test dl,0x80      ; Check if DL is masked correctly.
; 7C50 7502         jnz  0x7c54       ;   If not, then assume it's a
; 7C52 B280         mov  dl,0x80      ;   bogus value and set DL to 80.
; Thus, all the instructions below this line would be shifted by 9 bytes
; for version 0.97 (7C4Bh + 9 = 7C54h).


7C4B EA507C0000    JMP     0000:7C50  ; Long Jump to the next instruction
                                      ; because some bogus BIOSes jump to
                                      ; 07C0:0000 instead of 0000:7C00.

7C50 31C0          XOR     AX,AX
7C52 8ED8          MOV     DS,AX
7C54 8ED0          MOV     SS,AX
7C56 BC0020        MOV     SP,2000
7C59 FB            STI
7C5A A0407C        MOV     AL,[7C40]     ; <<<<<<<< Boot Drive
7C5D 3CFF          CMP     AL,FF
7C5F 7402          JZ      7C63
7C61 88C2          MOV     DL,AL
7C63 52            PUSH    DX
7C64 BE767D        MOV     SI,7D76       ; --> "GRUB "
7C67 E83401        CALL    7D9E          ; Display GRUB ID on screen.

7C6A F6C280        TEST    DL,80
7C6D 7454          JZ      7CC3
7C6F B441          MOV     AH,41         ; Function 41h of INT13
7C71 BBAA55        MOV     BX,55AA
7C74 CD13          INT     13            ; Test for INT13 Extensions

7C76 5A            POP     DX
7C77 52            PUSH    DX
7C78 7249          JB      7CC3
7C7A 81FB55AA      CMP     BX,AA55
7C7E 7543          JNZ     7CC3
7C80 A0417C        MOV     AL,[7C41]        <<<< Force LBA mode byte
7C83 84C0          TEST    AL,AL

; At this point, SuSE Linux 9.1 added a JS instruction to jump to
; the code at 7CC3. Why? Neither 0.94 nor 0.95 have this! There are
; already 3 jumps above (7C6D, 7C78, 7C7E) and 2 below (7C8A, 7CBC)
; to this same location. Is there a problem with "TEST  AL,AL" here?
```

```
7C85 7505          JNZ     7C8C
7C87 83E101        AND     CX,+01
7C8A 7437          JZ      7CC3

; LBA mode begins here:
; ====================
7C8C 668B4C10   *  MOV     ECX,[SI+10]

7C90 BE057C        MOV     SI,7C05        <<<<<<<<  Setup "Disk Packet"
                                                    for Extended Read
7C93 C644FF01      MOV     BYTE PTR [SI-01],01

7C97 668B1E447C  * MOV     EBX,[7C44]     <<<<<  Location of Stage2 code
                                                   from the beginning of
                                                the partition (the offset
                                                is in number of sectors).

7C9C C7041000      MOV     WORD PTR [SI],0010
7CA0 C744020100    MOV     WORD PTR [SI+02],0001

7CA5 66895C08   *  MOV     [SI+08],EBX

7CA9 C744060070    MOV     WORD PTR [SI+06],7000

7CAE 6631C0     *  XOR     EAX,EAX

7CB1 894404        MOV     [SI+04],AX

7CB4 6689440C   *  MOV     [SI+0C],EAX

7CB8 B442          MOV     AH,42          ; Function 42h of INT13
7CBA CD13          INT     13             ; Extended Read (using
                                          ; Disk Address Packet).
7CBC 7205          JB      7CC3           ; If LBA not supported,
                                          ; go to CHS mode only.

7CBE BB0070        MOV     BX,7000
7CC1 EB7D          JMP     7D40

7CC3 B408          MOV     AH,08          ; Function 08 of INT13
7CC5 CD13          INT     13             ; Get Drive Parameters

7CC7 730A          JNB     7CD3

7CC9 F6C280        TEST    DL,80          ; Tests if HDD exists.
7CCC 0F84F300   *  JZ      7DC3           ; Therefore, this jump is
; never taken unless grub was installed on and running from a floppy
; disk.  And only then will you find more executable code at 7DC3.
; In the source code file (stage1.S), you'll find this short comment
; about the extra code: "Kinda sneaky, huh?"

7CD0 E98D00        JMP     7D60           ; There was an HDD Error!

7CD3 BE057C        MOV     SI,7C05           <<<<<< "Disk Packet"
7CD6 C644FF00      MOV     BYTE PTR [SI-01],00
7CDA 6631C0     *  XOR     EAX,EAX

; Save number of heads:
7CDD 88F0          MOV     AL,DH
7CDF 40            INC     AX
7CE0 66894404   *  MOV     [SI+04],EAX
7CE4 31D2          XOR     DX,DX
```

```
7CE6 88CA          MOV     DL,CL
7CE8 C1E202     *  SHL     DX,02
7CEB 88E8          MOV     AL,CH
7CED 88F4          MOV     AH,DH

; Save number of cylinders:
7CEF 40            INC     AX
7CF0 894408        MOV     [SI+08],AX
7CF3 31C0          XOR     AX,AX
7CF5 88D0          MOV     AL,DL
7CF7 C0E802     *  SHR     AL,02

; Save number of sectors:
7CFA 668904     *  MOV     [SI],EAX

7CFD 66A1447C   *  MOV     EAX,[7C44]     <<<<<< Location of Stage2 code
                                                 from the beginning of
                                              the partition (the offset
                                              is in number of sectors).
7D01 6631D2     *  XOR     EDX,EDX
7D04 66F734     *  DIV     WORD PTR [SI]       ; Double word here.

7D07 88540A        MOV     [SI+0A],DL

7D0A 6631D2     *  XOR     EDX,EDX
7D0D 66F77404   *  DIV     WORD PTR [SI+04]    ; Double word here.

7D11 88540B        MOV     [SI+0B],DL
7D14 89440C        MOV     [SI+0C],AX
7D17 3B4408        CMP     AX,[SI+08]
7D1A 7D3C          JGE     7D58            ; There was a Geometry Error!
7D1C 8A540D        MOV     DL,[SI+0D]

7D1F C0E206     *  SHL     DL,06

7D22 8A4C0A        MOV     CL,[SI+0A]
7D25 FEC1          INC     CL
7D27 08D1          OR      CL,DL
7D29 8A6C0C        MOV     CH,[SI+0C]
7D2C 5A            POP     DX
7D2D 8A740B        MOV     DH,[SI+0B]
7D30 BB0070        MOV     BX,7000
7D33 8EC3          MOV     ES,BX
7D35 31DB          XOR     BX,BX
7D37 B80102        MOV     AX,0201         ; Function 02 of INT13
7D3A CD13          INT     13              ; Read 1 sector into Memory

7D3C 722A          JB      7D68            ; There was a Read Error!

7D3E 8CC3          MOV     BX,ES
7D40 8E06487C      MOV     ES,[7C48]       ; <<<<<<<< WORD [0800 hex]
                                           ; Note: 800:0000 = 0000:8000
7D44 60         *  PUSHA

7D45 1E            PUSH    DS
7D46 B90001        MOV     CX,0100
7D49 8EDB          MOV     DS,BX
7D4B 31F6          XOR     SI,SI
7D4D 31FF          XOR     DI,DI
7D4F FC            CLD
7D50 F3A5          REP     MOVSW

7D52 1F            POP     DS
```

```
7D53 61         *  POPA

; This is where we jump to the next stage of the code which GRUB loaded
; from the HDD into Memory locations 0000:8000 hex and following:

7D54 FF26427C      JMP     [7C42]           ; WORD    <<< 8000 hex.
                                            ; "stage2_address".

7D58 BE7C7D        MOV     SI,7D7C          ; --> "Geom Error"
7D5B E84000        CALL    7D9E             ; Display it on screen.
7D5E EB0E          JMP     7D6E             ; Finish it and 'lock-up'

7D60 BE817D        MOV     SI,7D81          ; --> "Hard Disk Error"
7D63 E83800        CALL    7D9E             ; Display it on screen.
7D66 EB06          JMP     7D6E             ; Finish it and 'lock-up'

7D68 BE8B7D        MOV     SI,7D8B          ; --> "Read Error"
7D6B E83000        CALL    7D9E             ; Display it on screen.
7D6E BE907D        MOV     SI,7D90          ; (For displaying " Error")
7D71 E82A00        CALL    7D9E             ; Finish it and 'lock-up'
7D74 EBFE          JMP     7D74             ; Locks-up execution in an
                                            ; infinite loop!  You must
                                            ;   reboot your computer!
```