

# How It Works: Master Boot Record (MBR)

[\[Home\]](#) [\[Q&A\]](#) [\[History\]](#) [\[Help\]](#) [\[SATA\]](#) [\[SCSI/1394\]](#) [\[HowItWorks\]](#) [\[Search\]](#)

[\[Hale's Test Software\]](#) [\[ATACT\]](#) [\[ATADEMO\]](#) [\[ATAMDT\]](#) [\[ATADRVR\]](#) [\[ADVDRVR\]](#)

[\[HIW Topics\]](#) [\[CHSxlat\]](#) [\[DOSboot\]](#) [\[FAQ\]](#) [\[FNF\]](#) [\[MBR\]](#) [\[OS2boot\]](#) [\[PartTable\]](#)

---

## Disassembly of a Master Boot Record (MBR)

This article is a disassembly of a Master Boot Record (MBR). The MBR is the sector at cylinder 0, head 0, sector 1 of a hard disk. An MBR is created by the FDISK program. The FDISK program of all operating systems must create a functionally similar MBR. The MBR is first of what could be many partition sectors, each one containing a four entry partition table.

At the completion of your system's Power On Self Test (POST), INT 19 is called. Usually INT 19 tries to read a boot sector from the first floppy drive. If a boot sector is found on the floppy disk, the that boot sector is read into memory at location 0000:7C00 and INT 19 jumps to memory location 0000:7C00. However, if no boot sector is found on the first floppy drive, INT 19 tries to read the MBR from the first hard drive. If an MBR is found it is read into memory at location 0000:7c00 and INT 19 jumps to memory location 0000:7c00. The small program in the MBR will attempt to locate an active (bootable) partition in its partition table. If such a partition is found, the boot sector of that partition is read into memory at location 0000:7C00 and the MBR program jumps to memory location 0000:7C00. Each operating system has its own boot sector format. The small program in the boot sector must locate the first part of the operating system's kernel loader program (or perhaps the kernel itself or perhaps a "boot manager program") and read that into memory.

INT 19 is also called when the CTRL-ALT-DEL keys are used. On most systems, CTRL-ALT-DEL causes an short version of the POST to be executed before INT 19 is called.

- [Where stuff is](#)
- [Summary of what this thing does](#)
- [Notes \(VERY IMPORTANT\)](#)
- [Entire MBR record in hex and ASCII](#)
- [Disassembly of the MBR](#)

## Where stuff is

- The MBR program code starts at offset 0000.
- The MBR messages start at offset 008b.
- The partition table starts at offset 01be.
- The signature is at offset 01fe.

## Summary of what this thing does

If an active partition is found, that partition's boot record is read into 0000:7c00 and the MBR code jumps to 0000:7c00 with SI pointing to the partition table entry that describes the partition being booted. The boot record program uses this data to determine the drive being booted from and the location of the partition on the disk.

If no active partition table entry is found, ROM BASIC is entered via INT 18. All other errors cause a system hang, see label HANG.

## Notes (VERY IMPORTANT)

1. The first byte of an active partition table entry is 80.  
This byte is loaded into the DL register before INT 13 is called to read the boot sector. When INT 13 is called, DL is the BIOS device number. Because of this, the boot sector read by this MBR program can only be read from BIOS device number 80 (the first hard disk). This is one of the reasons why it is usually not possible to boot from any other hard disk.
2. The MBR program uses the CHS based INT 13H AH=02H call to read the boot sector of the active partition. The location of the active partition's boot sector is in the partition table entry in CHS format. If the drive is >528MB, this CHS must be a translated CHS (or L-CHS, see my BIOS TYPES document). No addresses in LBA form are used (another reason why LBA does not solve the >528MB problem).

## Entire MBR record in hex and ASCII

```

OFFSET 0 1 2 3 4 5 6 7 8 9 A B C D E F *0123456789ABCDEF*
000000 fa33c08e d0bc007c 8bf45007 501ffbfc *.3.....|..P.P...*
000010 bf0006b9 0001f2a5 ea1d0600 00bebe07 *.....*
000020 b304803c 80740e80 3c00751c 83c610fe *......t....u.....*
000030 cb75efcd 188b148b 4c028bee 83c610fe *.u.....L.....*
000040 cb741a80 3c0074f4 be8b06ac 3c00740b *.t....t.....t.*
000050 56bb0700 b40ecd10 5eebf0eb febf0500 *V.....^.....*
000060 bb007cb8 010257cd 135f730c 33c0cd13 *..|...W.._s.3...*
```

```

000070 4f75edbe a306ebd3 bec206bf fe7d813d *Ou.....}.=*
000080 55aa75c7 8bf5ea00 7c000049 6e76616c *U.u.....|..Inval*
000090 69642070 61727469 74696f6e 20746162 *id partition tab*
0000a0 6c650045 72726f72 206c6f61 64696e67 *le.Error loading*
0000b0 206f7065 72617469 6e672073 79737465 * operating syste*
0000c0 6d004d69 7373696e 67206f70 65726174 *m.Missing operat*
0000d0 696e6720 73797374 656d0000 00000000 *ing system.....*
0000e0 00000000 00000000 00000000 00000000 *.....*
0000f0 TO 0001af SAME AS ABOVE
0001b0 00000000 00000000 00000000 00008001 *.....*
0001c0 0100060d fef83e00 00000678 0d000000 *.....x....*
0001d0 00000000 00000000 00000000 00000000 *.....*
0001e0 00000000 00000000 00000000 00000000 *.....*
0001f0 00000000 00000000 00000000 000055aa *.....U.*

```

## Disassembly of the MBR

This sector is initially loaded into memory at 0000:7c00 but it immediately relocates itself to 0000:0600.

```

                                BEGIN:                                NOW AT 0000:7C00,
RELOCATE

0000:7C00 FA                CLI                disable int's
0000:7C01 33C0               XOR                AX,AX                set stack seg to 0000
0000:7C03 8ED0               MOV                SS,AX
0000:7C05 BC007C            MOV                SP,7C00            set stack ptr to 7c00
0000:7C08 8BF4               MOV                SI,SP            SI now 7c00
0000:7C0A 50                 PUSH               AX
0000:7C0B 07                 POP                ES                ES now 0000:7c00
0000:7C0C 50                 PUSH               AX
0000:7C0D 1F                 POP                DS                DS now 0000:7c00
0000:7C0E FB                 STI
0000:7C0F FC                 CLD                clear direction
0000:7C10 BF0006            MOV                DI,0600            DI now 0600
0000:7C13 B90001            MOV                CX,0100            move 256 words (512
bytes)
0000:7C16 F2                 REPNZ
0000:7c00
0000:7C17 A5                 MOVSW
0000:7C18 EA1D060000        JMP                0000:061D            jmp to NEW_LOCATION

```

NEW\_LOCATION:

NOW AT 0000:0600

```

0000:061D BEBE07      MOV      SI,07BE          point to first table
entry
0000:0620 B304       MOV      BL,04           there are 4 table
entries

```

SEARCH\_LOOP1:

SEARCH FOR AN ACTIVE

ENTRY

```

0000:0622 803C80      CMP      BYTE PTR [SI],80  is this the active
entry?
0000:0625 740E       JZ       FOUND_ACTIVE      yes
0000:0627 803C00      CMP      BYTE PTR [SI],00  is this an inactive
entry?
0000:062A 751C       JNZ      NOT_ACTIVE        no
0000:062C 83C610      ADD      SI,+10           incr table ptr by 16
0000:062F FECB       DEC      BL               decr count
0000:0631 75EF       JNZ      SEARCH_LOOP1     jmp if not end of
table
0000:0633 CD18       INT      18              GO TO ROM BASIC

```

FOUND\_ACTIVE:

FOUND THE ACTIVE ENTRY

```

0000:0635 8B14       MOV      DX,[SI]         set DH/DL for INT 13
call
0000:0637 8B4C02     MOV      CX,[SI+02]     set CH/CL for INT 13
call
0000:063A 8BEE       MOV      BP,SI          save table ptr

```

SEARCH\_LOOP2:

MAKE SURE ONLY ONE

ACTIVE ENTRY

```

0000:063C 83C610      ADD      SI,+10           incr table ptr by 16
0000:063F FECB       DEC      BL               decr count
0000:0641 741A       JZ       READ_BOOT        jmp if end of table
0000:0643 803C00      CMP      BYTE PTR [SI],00  is this an inactive
entry?
0000:0646 74F4       JZ       SEARCH_LOOP2     yes

```

NOT\_ACTIVE:

MORE THAN ONE ACTIVE

ENTRY FOUND

0000:0648	BE8B06	MOV	SI,068B	display "Invld prttn tbl"
DISPLAY_MSG:		DISPLAY MESSAGE LOOP		
0000:064B	AC	LODSB		get char of message
0000:064C	3C00	CMP	AL,00	end of message
0000:064E	740B	JZ	HANG	yes
0000:0650	56	PUSH	SI	save SI
0000:0651	BB0700	MOV	BX,0007	screen attributes
0000:0654	B40E	MOV	AH,0E	output 1 char of message
0000:0656	CD10	INT	10	to the display
0000:0658	5E	POP	SI	restore SI
0000:0659	EBF0	JMP	DISPLAY_MSG	do it again
HANG:		HANG THE SYSTEM LOOP		
0000:065B	EBFE	JMP	HANG	sit and stay!
READ_BOOT:		READ ACTIVE PARTITION		
BOOT RECORD				
0000:065D	BF0500	MOV	DI,0005	INT 13 retry count
INT13RTRY:		INT 13 RETRY LOOP		
0000:0660	BB007C	MOV	BX,7C00	
0000:0663	B80102	MOV	AX,0201	read 1 sector
0000:0666	57	PUSH	DI	save DI
0000:0667	CD13	INT	13	read sector into
0000:7c00				
0000:0669	5F	POP	DI	restore DI
0000:066A	730C	JNB	INT13OK	jmp if no INT 13
0000:066C	33C0	XOR	AX,AX	call INT 13 and
0000:066E	CD13	INT	13	do disk reset
0000:0670	4F	DEC	DI	decr DI
0000:0671	75ED	JNZ	INT13RTRY	if not zero, try again
0000:0673	BEA306	MOV	SI,06A3	display "Errr ldng system"
0000:0676	EBD3	JMP	DISPLAY_MSG	jmp to display loop

INT13OK:

INT 13 ERROR

```

0000:0678 BEC206      MOV      SI,06C2      "missing op sys"
0000:067B BFFE7D      MOV      DI,7DFE      point to signature
0000:067E 813D55AA    CMP      WORD PTR [DI],AA55  is signature
correct?
0000:0682 75C7      JNZ      DISPLAY_MSG    no
0000:0684 8BF5      MOV      SI,BP         set SI
0000:0686 EA007C0000 JMP      0000:7C00      JUMP TO THE BOOT
SECTOR
    
```

WITH SI

POINTING TO

PART TABLE ENTRY

Messages here.

```

0000:0680 ..... 49 6e76616c *          Inval*
0000:0690 69642070 61727469 74696f6e 20746162 *id partition tab*
0000:06a0 6c650045 72726f72 206c6f61 64696e67 *le.Error loading*
0000:06b0 206f7065 72617469 6e672073 79737465 * operating syste*
0000:06c0 6d004d69 7373696e 67206f70 65726174 *m.Missing operat*
0000:06d0 696e6720 73797374 656d00.. ..... *ing system.      *
    
```

Data not used.

```

0000:06d0 .....00 00000000 *          .....*
0000:06e0 00000000 00000000 00000000 00000000 *.....*
0000:06f0 00000000 00000000 00000000 00000000 *.....*
0000:0700 00000000 00000000 00000000 00000000 *.....*
0000:0710 00000000 00000000 00000000 00000000 *.....*
0000:0720 00000000 00000000 00000000 00000000 *.....*
0000:0730 00000000 00000000 00000000 00000000 *.....*
0000:0740 00000000 00000000 00000000 00000000 *.....*
0000:0750 00000000 00000000 00000000 00000000 *.....*
0000:0760 00000000 00000000 00000000 00000000 *.....*
0000:0770 00000000 00000000 00000000 00000000 *.....*
0000:0780 00000000 00000000 00000000 00000000 *.....*
0000:0790 00000000 00000000 00000000 00000000 *.....*
0000:07a0 00000000 00000000 00000000 00000000 *.....*
0000:07b0 00000000 00000000 00000000 0000.... *.....*
    
```

The partition table starts at 0000:07be. Each partition table entry is 16 bytes. This table defines a single primary partition

which is also an active (bootable) partition.

```
0000:07b0 ..... 8001 * .....*
0000:07c0 0100060d fef83e00 00000678 0d000000 * .....x....*
0000:07d0 00000000 00000000 00000000 00000000 * .....*
0000:07e0 00000000 00000000 00000000 00000000 * .....*
0000:07f0 00000000 00000000 00000000 0000.... * .....*
```

The last two bytes contain a 55AAH signature.

```
0000:07f0 ..... 55aa * .....U.*
```

---

*This page was last updated on 06 May 2002.*