The Master Boot Record (MBR) and Why it is Necessary?

If you have arrived here through a search engine, and there's no menu to the left click here!

When you turn on your PC, the processor attempts to begin the process of processing data. But, since the system memory is empty, the processor doesn't really have anything to execute, or even begin to know where to look for it. To ensure that the PC will always boot regardless of the BIOS code, both chip and BIOS manufacturers developed their code so that the processor, once turned on, always starts executing at the same place, FFFF0h.

Similarly, every hard disk must have a consistent "starting point" where key information is stored about the disk, such as the number of partitions and what type they are. There also must be someplace where the BIOS can load the initial boot program that starts the process of loading the operating system. The place where this information is stored is called the *master boot record* (*MBR*), also referred to as the *master boot sector* or even just the *boot sector*. Do not confuse the master boot sector with **volume boot sectors**, which are indeed different.

The master boot record is always located at cylinder 0, head 0, and sector 1, the first sector on the disk. This is the consistent starting point that the disk will always use. When a computer starts and the BIOS boots the machine, it will always look at this first sector for instructions and information on how to proceed with the boot process and load the operating system. The master boot record contains the following structures:

- **Master Partition Table:** This small bit of code that is referred to as a table contains a complete description of the partitions that are contained on the hard disk. When the developers designed the size of this master partition table, they left just enough room for the description of four partitions, hence the four partition (four physical partitions) limit. For this reason, and no other, a hard disk may only have four true partitions, also called *primary or physical partitions*. Any additional partitions must be logical partitions that are linked to (or are part of) one of the primary partitions. One of these partitions is marked as active, indicating that it is the one that the computer should used to continue the boot process.

- **Master Boot Code:** The master boot record is the small bit of computer code that the BIOS loads and executes to start the boot process. This code, when fully executed, transfers control to the boot program stored on the boot (active) partition to load the operating system.

Obviously, due to the great importance of the information stored in the master boot record, should it become damaged or corrupted in some way, serious data loss often occurs. The master boot code is the first program executed when you turn on your PC, and is often the target of a virus. In order to understand the Master Boot Record and the Master Boot Code buried within it, it might be helpful to understand the sequence of events that occur when your computer starts.

Although often overlooked as a diagnostic tool, the Boot Sequence performed by your computer can often tell you what may be wrong with the hardware itself should you be experiencing problems, even moderate ones. The following will take you through the System Boot sequence step by step. If you

notice any thing unusual through each sequence of events, then you only need to look at the step where the delay appears to occur or back one step.

What occurs during the System Boot Sequence?

The system BIOS, a small bit of code inserted in the BIOS prom chip located on your computers motherboard, is what starts the computer running when you turn it on. Below we have outlined the typical sequence of events that occur during your computers startup process, although they will vary by the manufacturer of your hardware, BIOS, and the peripherals you have in the PC. Again, this is a typical sequence of events that occur when you turn on your computers power switch:

1. The internal power supply turns on, initializes and then takes a few moments to generate reliable power for the rest of the computer. If the power received by the motherboard's chipset, and subsequently the processor, is not within expected parameters, the chipset will generate a reset signal to the processor in the same fashion as if you were to touch the reset button. This will continue until the motherboard receives a Power Good signal from the power supply or you turn the system off because of a failed power supply.

2. After a Power Good signal is received, or after the reset button is released and there is confirmation of reliable power, the processor will be ready to start executing. When the processor first starts, it really has no idea what to do next as there is nothing at all in the memory to execute. Of course the processor designers are aware of this, so they pre-program the processor to always look at the same place in the system, the BIOS ROM, for the small bit of startup code to begin the boot process. This is typically located at memory location FFFF0h, or right at the end of the system memory. Developers locate it there in the event the size of the ROM has to be changed so as to prevent compatibility problems. Since there are only 16 bytes from there to the end of conventional memory area, this location contains just a "jump" instruction telling the processor where to go to find the real BIOS startup program.

3. The BIOS performs the **power-on self test (POST)**. If there are any fatal errors, the boot process stops. If the POST is successful, the BIOS calls INT 19 (Interrupt 19) and then proceeds to look for devices attached to the motherboard.

4. The BIOS code begins its search by looking for a video card, more particularly, its looking for the video card's built in BIOS program, (normally found at location C000h in memory) and if found, runs it. The system BIOS executes the video card BIOS, which in turn initializes the video card. Most modern video cards will display information on the screen about the video card, which is why on some modern PC's you usually see something on the screen about the video card before you see the messages from the system BIOS itself.

5. Once video has been enabled, the BIOS begins searching for other devices that may have their own ROM and whether that ROM has its own BIOS code. Normally, the floppy drive is located at 0000:7C00, and the IDE/ATA hard disk BIOS will be found at C8000h. If a floppy and/or hard drive is found, their codes are executed. If, during this INT 19 process, any other device BIOS's are found, they are executed as well.

6. The BIOS displays its **startup screen**, which provides some key information about the BIOS as well as other system information.

7. As the boot sequence continues, the BIOS continues to perform additional tests on the system.

Depending upon the system manufacturer, this will usually include a memory count. The BIOS will generally display an error message on the screen if it encounters an error when it counts installed memory. You will find these error messages and their explanations in our **Support Center** in the **Motherboard and BIOS** section or you can **click here**. (Clicking here will open a new window)

8. During the next phase of the BIOS startup process, it performs somewhat of an inventory of the hardware installed in the system, and then communicates or interrogates it to ensure that the hardware is functioning. Most modern BIOS's have automatic settings to collect information such as memory timing, based on what kind of memory it finds. Today's BIOS's dynamically set hard drive parameters and access modes, and will display a message on the screen for each drive they detect and configure in this way. It will also search for and label logical devices such as COM and LPT ports. Note: If the BIOS supports the Plug and Play standard, and the feature is enabled, this is the point at which it will detect and configure Plug and Play devices and display a message on the screen for each one found.

9. During the final phase of the POST and BIOS boot process, the BIOS will display a summary screen with your system's configuration. While early machines, 486 through Pentium II, were fairly accurate about the system configuration information, later machines using the later Pentium III and AMD processors can have some inaccurate information. This is usually related to BIOS setup issues involving processor information. Checking this data can be helpful in diagnosing setup problems, although it can be hard to see because sometimes it flashes on the screen and then scrolls off the top.

10. Once the BIOS finishes what it needs to do, it begins searching for a drive to boot an operating system. All BIOS's contain a setting that controls this search sequence for a boot drive. Most are set to first look for a bootable floppy disk, and if one is not found then proceed to a hard disk, which is usually the C: drive. Some BIOS's permit you to boot from your CD-ROM drive or other devices such as a SCSI (Small Computer System Interface), depending on the boot sequence selected. Once the BIOS identifies its target boot drive, the it looks for boot information to start the operating system boot process. If it is searching a hard disk, it looks for a master boot record at cylinder 0, head 0, sector 1, the first sector on the disk. If it is searching a floppy disk, it looks at the same address on the floppy disk for a volume boot sector.

11. Once the boot sector is found and its contents or data verified, the BIOS starts the process of booting the operating system by using the information in the boot sector. If this is a floppy disk boot sector, the information is read into memory at location 0000:7c00. INT 19 goes to memory location 0000:7c00 to continue the process. If no boot sector is found on the floppy drive, INT 19 moves to the next bootable drive in the list provided by the motherboard BIOS, usually a hard drive, and then attempts to read the MBR. If a Master Boot Record is found, it is read into memory at location 0000:7c00 and INT 19 jumps to memory location 0000:7c00 the same as was the case with the floppy. At this point, the BIOS attempts to move control of the computer from the BIOS to the actual operating system.

    Next, the small program in the Master Boot Record will attempt to locate an active (bootable) partition in the hard drives partition table. If such a partition is found, the boot sector of that partition is also read into memory at location 0000:7C00 and then MBR program itself jumps to memory location 0000:7C00. Keep in mind that each operating system has its own boot sector format. The next step involves the small program in the boot sector locating the first part of the operating system's kernel loader program, or in some cases the kernel itself or perhaps a boot

manager program, and read then that into memory. For you Windows NT and Windows 2000 fans, this kernel loader is referred to as NTLDR. You will find a **description of the DOS boot process here**.

12. If no boot device of any type can be found, the system will display an error message and stop. The specific error message is depends on the BIOS developer and/or the computer's manufacturer, and can be anything from a rather clear "No boot device" to the very cryptic "NO ROM BASIC - SYSTEM HALTED". This will also happen if you have a bootable hard disk partition but forget to set it active. Believe it or not, you can partition a drive, format it and install the operating system and never realize that there is problem until the first start of that operating system occurs.

This entire process is referred to as a "cold boot" (since the machine was off, or cold, when it started). A "warm boot" also known as a "soft boot" is the same thing except it occurs when the machine is rebooted using the Ctrl + Alt + Del keys. In this case the POST is skipped and the boot process continues at roughly step 8 above. As a side note, INT 19 is also called when the CTRL-ALT-DEL keys are used. On most systems, Ctrl + Alt + Del causes a soft-boot or shorten version of the POST to be executed before INT 19 is called.

Many things can damage the Master Boot Record, therefore it might be helpful for you to know where certain portions of the code reside in the MBR, should you have to recover the MBR manually.

You will find that:

- 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

If an active partition is found, regardless of where it is found, that partition's boot record is read into memory at 0000:7c00 and then the MBR code is added to 0000:7c00 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 exact location of the partition on the disk. If no active partition table entry can be found, the boot process enters ROM BASIC via INT 18. All other errors will cause the system to hang or stop.

Notes (Extremely 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. Of course, there are exceptions to every rule, but those are beyond the scope of this exercise.

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. 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                       allow int's
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                     move MBR from 0000:7c00
0000:7C17 A5            MOVSW                        to 0000:0600
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 PARITION 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 systm"
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.*
```