BootIt 2

Extended Master Boot Record Specification
Revision 971207

And

Related Information

Printed April 18, 2021

Extended Master Boot Record

The Extended Master Boot Record (EMBR) is defined as the area of a hard disk from physical cylinder 0, head 0, sector 2, to (and including) the last physical sector of cylinder 0, head 0. The actual area available to the EMBR depends upon the BIOS mapping of the logical CHS and reserved sectors. The minimum availability requirements for the EMBR is sectors 3 through 17.

Master Partition Table

The purpose of the master partition table (MPT) is to over come the current limitation of the partition table. The MPT resides in the EMBR.

The MPT includes a header and partition details which includes a partition name. The partition detail entries are in a new format not used by the standard partition table. The MPT header contains information that will allow the table to be extended or revised with minimal or no programming changes.

Master Boot File Table

Immediately following the MPT is the Master Boot File Table (MBFT). The purpose of this table is to provide a common area for boot file names to be referenced.

The MBFT contains a header followed by one or more detail entries. The detail entries include the segment which should be used to load the boot file and which segment/offset the first byte of code begins. It also includes a parameter string which is to be used by a file system boot loader.

Master Driver Table

Immediately following the MBFT there *may be* a master driver table. The main purpose of the master driver table (MDT) is to provide a way to extend the system BIOS.

The MDT includes a header, driver details, and driver options. The MDT header contains information that will allow the table to be extended or revised with minimal or no programming changes. The driver details contain information about the driver including a name, where to find it, and whether to load it. The driver options provides a way to configure the driver.

EMBR Initiator (EMBRI)

The EMBRI resides in the MBR. It is the initial code responsible for loading of the EMBR and EMBRL. Before control is given to the EMBRL it must set SS:SP to 0:400h. It must also set ES and DS equal to the EMBR segment.

EMBR Loader (EMBRL)

All physical hard drives with an EMBR need a default driver that begins in the sector following all tables. The default driver on physical hard drive 0 is called the Extended Master Boot Record Loader. The default driver on all other hard drives will remain dormant until that hard drive becomes physical hard drive 0.

Control is transferred to the EMBRL with the registers DS and ES equal to the EMBR segment, SS:SP to 0:400h, and all other registers undefined.

As mentioned before, the EMBRL is located on physical hard drive 0. It is the driver responsible for processing the MPT and MDT. The EMBRL is a driver and is included in the MDT. Therefore, a MDT must exist on

physical hard drive 0. The EMBRL must begin in the sector following the last sector containing any portion of a table.

The EMBRL option string in the MDT must be at least 15 bytes long. See the following table.

Byte Number	Description
0	Key Delay Time (seconds) to wait for user to press space bar or 0 to automatically load the EMBRM..
1	Physical hard drive number that contains the partition to boot. HD 80h = '0'
2	Drive Number of Drive to come ready. If drive is not ready at boot, pause 5 seconds and reboot.
3	Boot option flag1: ASCII values >= '0'. Bit 1=Swap Floppy disk drives. Bit 2=Auto boot floppy if present, Bit 3=Force EMBRM
4-14	Reserved
15-X	Open

EMBR Manager (EMBRM)

The EMBRM is the code that manages the EMBR and interfaces with the user. It has the responsibility of maintaining the integrity of the EMBR.

The EMBRM is located by searching the MPT for the first entry that is marked as the EMBRM. The search begins with Hard Drive 0 and increments through each hard drive until the EMBRM partition is found.

EMBR Layout

Offset	Length	Description
0	4	"EMBR" ID=Extended Master Boot Record
4	1	EMBR Revision (For all revision fields low 5 bits changed means backwards compatibility. high 3 bits means not backwards compatible.
5	1	Minimum SPT needed to load EMBR. bit 7 & 6 reserved.
6	1	Sectors prior to code sector. bit 7 & 6 reserved.
7	1	Reserved.
8	2	Sector size in Bytes
10	2	Last Cylinder Number (0..X, X<1024)
12	1	Last Head Number
13	1	Number of Sectors Per Track
14	4	Reserved
18	3	"MPT" ID=Master Partition Table
21	1	Master Partition Revision Level
22	1	Number of Partition Entries
23	5	Reserved
28	???	Partition Entries
	4	"MBFT" ID=Master Boot File Table
	1	MBFT Revision Level
	1	Number of MBFT Entries
	???	MBFT Entries
	3	"MDT" ID=Master Driver Table if MDT exists.
	1	Driver Table Revision Level
	1	Number of Driver Entries
	11	Reserved
	???	Driver Entries
	???	Possible unused area
	???	EMBRL/Code - Sector Aligned

Partition Entries Layout

Offset	Length	Description
0	2	Partition Flag: bit 15=boot this partition; bit 14=EMBR manager; bit 13=Bootable; bit 12=MultiBoot Partition, bit 11=Partition Bootable from HD0 only, bits 10-0=reserved.
2	4	Starting LBA of partition.
6	4	Ending LBA of partition.
10	1	Boot File Table Entry Number (binary).
11	1	File System Type
12	16	Partition Name (ZTerm-15 chars max.)

MBFT Entries Layout

Offset	Length	Description
0	16	Boot File Name (ZTerm-15 chars max.).
17	2	Load Segment
19	2	First Code byte Offset
21	2	First Code byte Segment
23	1	Number of Sectors needed to load
24	1	Reserved
25	2	Reserved
27	1	Parameter String Allocated Length (bits 7&6=reserved)
28	?	Parameter String (Z-Term if < Allocated Length)

Driver Entry Layout

Offset	Length	Description
0	1	Driver Flag: bit 7=Install/load; bit 6=Proven flag; bit 5=EMBRL flag, bit 4=Required flag, 3-0=Reserved.
1	1	Reserved.
2	1	Driver Size (in sectors)
3	1	Reserved
4-7	4	LBA where driver is located *. (logical if within first CH)
8-13	6	Reserved
14	16	Driver Name (ZTerm-15 char max.)
30	1	Parameter String Allocated Length (bits 7&6=reserved)
31-?	?	Parameter String (Z-Term if < Allocated Length)

*Note: The LBA should reside in a partition or the EMBR. The EMBR should only be used for drivers that add system support that is required to successfully boot the system.

Programming

Drivers

The purpose of this section is to explain how to write drivers that are compatible with the MDT. This consists of a few program responsibilities and conforming to the driver layout on the next page.

The LBA entry in the MDT is a logical LBA if the value is less than the number of sectors per track. If this is the case, LBA 0 is the first sector beyond the sector that contains the last byte of the MDT.

The EMBRL is responsible for activating the driver installation routine. The loader performs the following:

- Load the driver to a memory area at offset 0.
- Perform a far call to the driver at xxxx:0003h with DS:BX pointing to the driver option string. All other registers are undefined.
- Save the pointer returned by the driver to its uninstall routine.
- If the driver returned a non-zero pointer to an uninstall routine then the loader will call that routine prior to loading the boot record of the operating system to boot..

The following list contains the driver program responsibilities.

- Install the resident portion of the driver some place in memory. It must also make other programs aware of the area in memory it resides. For instance, using/updating the BIOS memory size value at 413h is a good choice.
- Handle multiple load requests. The EMBR loader does not keep track of what drivers are loaded. Every driver will be loaded for every entry found in all MDTs.
- Use the DS:BX pointer to access the option string for that driver. The first byte of the string is the maximum length allowed for the string.
- Keep the code/data size of the entire driver under 64K and initialize segment registers as needed. Do not write outside the 64K block given to the driver except to install the resident portion of the driver.
- Return control back (retf) to loader with the direction flag, DS, and SS:SP unchanged. Also, ES:BX needs to return either NULL or a pointer to an uninstall routine.
- The uninstall routine must return control (retf) with the direction flag, DS, and SS:SP unchanged.

Driver Program Layout

Offset	Length	Description
0	2	AA55h Flag.
2	1	Allocation Size in 512 byte blocks.
3	3	Start of Code - Jump to Initialization Routine.
6	2	"DF"=Driver Flag (optional*)
8	2	Allocation Size in Bytes. (optional*)
10	1	Default Parameter Allocation Length (bits 7&6 reserved) (optional*)
11	1	Reserved (optional*)
12	??	Code.

* If *any* optional fields are used then all optional fields *must* be used.

File System Boot Sectors

This sections explains how the EMBRL loads a file system boot sector.

The EMBRL loads the boot sector of a partition to 0:7C00h. It will pass the drive number of the partition in DL. The stack will also contain the drive number along with a far pointer to the partition entry and MBFT. At this point these entries are stand alone and no longer part of the larger MPT and MBFT.

To determine if a EMBRL was used, first check if AL equals DFh. Next check to make sure DL matches the hard drive number on the stack. If all checks pass then assume an EMBRL has loaded the partition. The stack will look like this:

```
                HDNo.
                PtrSegMPTEntry
                PtrOfsMPTEntry
                PtrSegMBFTEntry
SP ->           PtrOfsMBFTEntry
```

You can then use the information in the partition entry and MBFT to boot the correct operating system. Use the load segment to load the boot file and the first code byte segment/offset to transfer control to the boot file. The parameters are specific to a file system boot loader.

If the file system boot sector and all OS's under it recognize the EMBR and ignore the MBR, place a DFh byte value to offset 509 of the boot sector.

Miscellaneous

All entries use the standard PC-ASCII format.

Acronyms

- **EMBR:** Extended Master Boot Record.
- **EMBRI:** Extended Master Boot Record Initiator.
- **EMBRL:** Extended Master Boot Record Loader.
- **EMBRM:** Extended Master Boot Record Manager.
- **MBFT:** Master Boot File Table.
- **MDT:** Master Driver Table.
- **MPT:** Master Partition Table.
- **OS:** Operating System.

Revisions

961014 Initial Release
961130 Added the drive to come ready byte to EMBRL parameter string and clarification on drivers/partitions.
970101 Added boot option flag 1 to the EMBRL parameter string.
970101 Changed bit 12 of the MPT Flag from reserved to multi-boot indicator. (MPT Revision 2).
970809 Changed bit 11 of the MPT Flag from reserved to HD0 boot indicator. (MPT Revision 3).
971207 Changed entry to file system boot sector to point to MBFT entry instead of the whole table.