NTFS Boot sector

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Jump Instruction			OEM ID								Bytes/ Sector		Sect/ clust	res	
10	0x000000			unused		Media desc	0x0000		Sect / track		Number heads	Hidden Sectors				
20	unused								Total Sectors							
30	Logical Cluster of $MFT								Logical Cluster of $MFTMirr							
40	Clust / File record segment			Clusters / Index Block					Volume Serial Number							
50	Checksum			Boot Code												
60	Boot Code															
.																
.																
1E0																
1F0	Boot Code														55	AA

Key

Sect / Clust - Sectors per cluster

res - reserved, note that the terms reserved, unused and 0x00 are specified by Microsoft, the difference between reserved and unused is not specified. However it should be noted that the blocks specified as all zeros have defined meaning within FAT boot sectors.

media desc - Media descriptor, legacy from DOS, 0xF8 indicates fixed disk, 0xF0 a HD 3.5inch floppy.

BIOS Parameter Block (BPB)
Extended BPB
Boot code
End of sector marker

reference: http://technet.microsoft.com/en-us/library/cc976796.aspx

NTFS files

File	Name	$MFT record #	Description
$Mft	Master File Table	0	Contains one base file record for each file and folder on an NTFS volume. If the allocation information for a file or folder is too large to fit within a single record, other file records are allocated as well.
$MftMirr	MFT mirror	1	Guarantees access to the MFT in case of a single-sector failure. It is a duplicate image of the first four records of the MFT.
$LogFile	Log file	2	Contains information used by NTFS for faster recoverability. The log file is used by Windows Server 2003 to restore metadata consistency to NTFS after a system failure. The size of the log file depends on the size of the volume, but you can increase the size of the log file by using the Chkdsk command.
$Volume	Volume	3	Contains information about the volume, such as the volume label and the volume version.
$AttrDef	Attribute definitions	4	Lists attribute names, numbers, and descriptions.
.	Root file name index	5	The root folder.
$Bitmap	Cluster bitmap	6	Represents the volume by showing free and unused clusters.
$Boot	Boot sector	7	Includes the BPB used to mount the volume and additional bootstrap loader code used if the volume is bootable.
$BadClus	Bad cluster file	8	Contains bad clusters for a volume.
$Secure	Security File	9	Contains unique security descriptors for all files within a volume.
$Upcase	Upcase table	10	Converts lowercase characters to matching Unicode uppercase characters.
$Extend	NTFS extension file	11	Used for various optional extensions such as quotas, reparse point data, and object identifiers.
		12-15	Reserved for future use.

source: http://technet.microsoft.com/en-us/library/cc781134(WS.10).aspx

Some $MFT entry attributes

ID	Attribute Type	Description
0x10	Standard Information	Includes information such as time stamp and link count.
0x20	Attribute List	Lists the location of all the attribute records that do not fit in the MFT record.
0x30	File Name	A repeatable attribute for both long and short file names. The long name of the file can be up to 255 Unicode characters. The short name is the MS-DOS-readable, 8.3, case-insensitive name for the file. Additional names, or hard links, required by POSIX can be included as additional file name attributes.
0x40	Object ID	A volume-unique file identifier. Used by the link tracking service. Not all files have object identifiers.
0x50	Security Descriptor	Shows information about who owns the file and who can access the file.
0x60	Volume Name	Used only in the $Volume system file. Contains the volume label.
0x70	Volume Information	Used only in the $Volume system file. Contains the volume version.
0x80	Data	Contains file data. NTFS allows multiple data attributes per file. Each file typically has one unnamed data attribute. A file can also have one or more named data attributes, each using a particular syntax.
0x90	Index Root	Used to implement folders and other indexes.
0xA0	Index Allocation	Used to implement folders and other indexes.
0xB0	Bitmap	Used to implement folders and other indexes.
0xC0	Reparse Point	Used for directory junction points and volume mount points. They are also used by file system filter drivers to mark certain files as special to that driver.
0x100	Logged Tool Stream	Similar to a data stream, but operations on a logged tool stream are logged to the NTFS log file just like NTFS metadata changes. Used by EFS.

source: http://technet.microsoft.com/en-us/library/cc976808.aspx
http://msdn.microsoft.com/en-us/library/bb470038

File Record Segment Header

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	F	I	L	E	Update Seq array offset		Update Seq array size		$LogFile Sequence Number							
1	Seq no			Hard Link Count		1st attrib offset		Flags	Used size of file record				Allocated size of file record			
2	File reference to base file record								Next attrib ID				MFT Record No			
3	default location of update seq array (size determined by seq size)						Reserved for update sequence array?									
	Reserved for sequence array?								Common location of 1st attrib							

Resident Attribute Header

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Type ID				Attribute Length				Form code	name len	Name offset		flags		Attrib ID	
1	Content length				Content offset		unused									

Form code
0x00 = Resident
0x01 = Non resident

Flags
0x00FF = Compressed
0x8000 = Sparse
0x4000 = Encrypted

Non Resident Attribute Header

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Type ID				Attribute Length				Form code	name len	Name offset		flags		Attrib ID	
10	Start virtual cluster number								Ending virtual cluster number							
20	Runlist offset		Compression unit size		0x0000				Allocated size of attribute content (physical size)							
30	Actual size of attribute content (logical size)								Initialized size of attribute content							
40	Common start of Data runlists															

Attrib ID starts from zero
Virtual cluster numbers are used when a MFT record is fragmented

$Standard_Information

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Date Created*								Date Modified							
10	Date MFT record modified								Date Accessed							
20	Flags				Max Versions				Version Num				Class ID			
30	Owner ID				Security ID				Quota Charged							
40	Update Sequence Number															

*Time values are Microsoft FILETIME, 100 nanoseconds since January 1, 1601 UTC

flags (used for both $Standard_Information and $File_Name

Bit	Hex	Meaning	Bit	Hex	Meaning
0	0x0001	Read only	8	0x0100	Temporary
1	0x0002	Hidden	9	0x0200	Sparse File
2	0x0004	System	A	0x0400	Reparse Point
3	0x0008		B	0x0800	Compressed
4	0x0010		C	0x1000	Offline
5	0x0020	Archive	D	0x2000	Not Indexed
6	0x0040	Device	E	0x4000	Encrypted
7	0x0080	Normal	F	0x8000	

Source: http://msdn.microsoft.com/en-us/library/aa365535(v=VS.85).aspx

$File_Name

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Parent Directory								Date Created							
10	Date Modified								Date MFT Modified							
20	Date Accessed								Logical file size							
30	Size on disk								Flags*				Reparse value			
40	Name len	Name type	Name (variable length)													

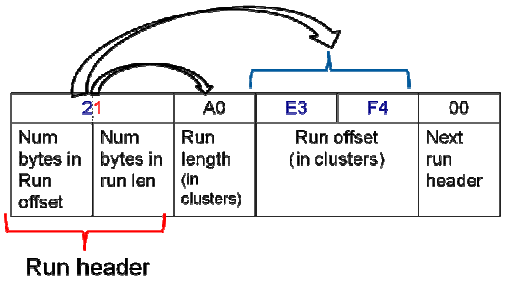
Name types

Value	Description
0	POSIX (unicode, case sensitive)
1	Win32 (unicode, case insensitive)
2	DOS (8.3 ASCII, case insensitive)
3	Win32 7 DOS (when Win32 fits in DOS space)

$Data (Standard Header with data run, may be resident or non resident, non resident shown here)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Type ID 0x80				Attribute Length				Form code	name len	Name offset		flags		Atrib ID	
10	Start virtual cluster number								Ending virtual cluster number							
20	Runlist offset		Compression unit size		0x0000				Allocated size of attribute content (physical file size)							
30	Actual size of attribute content (logical file size)								Initialized size of attribute content							
40	Data runlists															

Data runlists



$ATTRIBUTE_LIST entry (one entry per attribute in the record, including attributes that precede the list).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	Type ID				Record Length		Atrib name len	Attrib name offset	Lowest VCN							
	$MFT Record number						Seq num		Reserved		Start of name (if present)					

Source: http://msdn.microsoft.com/en-us/library/bb470038%28v=vs.85%29.aspx