Active @ UNDELETE Users Guide

Contents

Legal Statement	4
Active@ UNDELETE Overview	5
Getting Started with Active@ UNDELETE	
Active@ LINDELETE Views And Windows	6
Recovery Explorer View	7
Logical Drive Scan Result View	7
Physical Device Scan View	8
Search Results View	10
Application Log	11
Welcome View	11
Using Active@ UNDELETE Overview	13
Desever deleted Files and Folders	L 14
Recover deleted files did folders	14۲4 ۱۳
Scall a volume (Logical Drive) for deleted files	13
Scan for files by their signatures	10
Recover files and folders.	
Virtual RAID Assembly	
Decrypt recovered files	
Restore Partition	
Scan Disk (Physical Device)	
Scan for Deleted Partitions	
Edit or Clone Detected Partitions	24
Restore detected partition	
Using Scan Results	
Stop and Resume a scan	27
Filter detected partitions by certainty	
Work with scan results	29
File Preview	30
Search for deleted Files and Folders	31
File Filter Toolbar	33
Supported File Signatures	
Custom (user defined) file signature templates	
Custom Signatures Size Script	38
How to Use Wildcards	43
Application Preferences	44
Active@ UNDELETE Tools Overview	48
Hardware Diagnostic File	
Partition Manager Overview.	
Initialize Disk (Physical Device)	49
Create New Partition	50
Change Partition Attributes	51
Format Partition	51
Edit Boot Sectors	52
Edit Doct Sectors	53
Create Virtual Dartition	
Disk Editor Tool	۳-۲
Open objects in Disk Editor	۲C
Subject Navigation and Information	
Data Inspector	סכ ריז
Udia Inspector	
USIIIY DUUKIIIdiks	۵۵ دم
Searching in Disk Eulor	

	Editing with Disk Editor	.60
	Rollback Partition Changes	66
	Disk Image Overview	.66
	Create a Disk Image	. 67
	Open Disk Image	.69
	Virtual Partition (Logical drive clone)	.71
	Working with a corrupted RAID system	72
Activ	ve@ UNDELETE Wizards Overview	75
	Active@ UNDELETE Recovery Wizards Overview	. 75
	Recover Deleted Files Wizard	.75
	Recover Files by Signature Wizard	. 77
	Recover files from a deleted partitions wizard	. 80
	File Recovery Expert Wizard	. 82
	Restore a Deleted Partition Wizard	82
	Create a New Partition Wizard	. 84
	Create a virtual RAID Array Wizard	. 85
	Active@ UNDELETE Disk Image Wizards Overview	.87
	Create a Disk Image Wizard	. 87
	Open a Disk Image Wizard	88
	Verify a Disk Image Wizard	. 90
Data	a Recovery Concept Overview	93
Hard	lware and Disk Architecture	94
	Hardware and Disk Organization	.94
	Hard Disk Drive Basics	.94
	Master Boot Record (MBR)	. 96
	Partition Table	97
	Disk Arrays (RAID's) 1	102
	Logical Disk Manager (LDM) Overview	104
File	Logical Disk Manager (LDM) Overview	104 06
File	Logical Disk Manager (LDM) Overview	104 06 106
File	Logical Disk Manager (LDM) Overview	104 06 106 107
File	Logical Disk Manager (LDM) Overview	104 06 106 107 109
File	Logical Disk Manager (LDM) Overview	104 D6 106 107 109 110
File	Logical Disk Manager (LDM) Overview	104 06 106 107 109 110 116
File	Logical Disk Manager (LDM) Overview	104 D6 106 107 109 110 116 117
File	Logical Disk Manager (LDM) Overview	104 D6 106 107 109 110 116 117 117
File	Logical Disk Manager (LDM) Overview	104 106 107 109 110 116 117 117 122
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 116 117 117 122 124
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 116 117 122 124 131
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 116 117 122 124 131 133
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 116 117 122 124 131 133 133
File	Logical Disk Manager (LDM) Overview	104 106 107 109 110 116 117 122 124 131 133 133 137
File	Logical Disk Manager (LDM) Overview	104 106 107 109 110 117 122 124 131 133 133 137 138
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 116 117 122 124 131 133 133 133 137 138 138
File	Logical Disk Manager (LDM) Overview. 10 Systems. 10 Windows NT File System (NTFS). 10 NTFS Partition Boot Sector. 11 NTFS Master File Table (MFT). 11 Data Integrity and Recoverability with NTFS. 11 Data Integrity and Recoverability with NTFS. 11 Extended File System (exFAT). 12 Volume Layout. 11 exFAT Directory Structure. 12 exFAT Defined Directory Entries. 12 exFAT Cluster Heap. 13 File System (FAT). 14 FAT Partition Boot Sector. 14 FAT Root Folder. 14 FAT Root Folder. 14 FAT Folder Structure. 14 FAT32 Features. 14	104 D6 106 107 109 110 116 117 122 124 131 133 133 137 138 138 139
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 117 117 122 124 131 133 137 138 138 138 139 48
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 116 117 122 124 131 133 133 137 138 138 138 139 48 148
File	Logical Disk Manager (LDM) Overview. 10 Systems. 10 Windows NT File System (NTFS). 11 NTFS Partition Boot Sector. 11 NTFS Master File Table (MFT). 11 Data Integrity and Recoverability with NTFS. 12 Extended File System (exFAT). 12 Volume Layout. 11 exFAT Defined Directory Structure. 12 exFAT Defined Directory Entries. 12 exFAT Cluster Heap. 12 File System (FAT). 14 FAT Partition Boot Sector. 14 FAT Partition Boot Sector. 14 FAT File Allocation Table. 14 FAT Folder Structure. 14 FAT Folder Structure. 14 FAT Seatures. 14 Disk scan for deleted entries. 14 Define Clusters chain for the deleted entry. 14	104 D6 107 109 110 116 117 122 124 131 133 133 133 137 138 138 138 139 48 148 151
File	Logical Disk Manager (LDM) Overview	104 D6 107 109 110 116 117 122 124 131 133 133 137 138 138 138 139 48 148 151 153
File File Part	Logical Disk Manager (LDM) Overview. 10 Systems. 10 Windows NT File System (NTFS). 11 NTFS Partition Boot Sector. 11 NTFS Master File Table (MFT). 11 NTFS File Types. 12 Data Integrity and Recoverability with NTFS. 12 Extended File System (exFAT). 12 Volume Layout. 11 exFAT Directory Structure. 12 exFAT Defined Directory Entries. 12 exFAT Cluster Heap. 14 File System (FAT). 14 FAT Partition Boot Sector. 14 FAT File Allocation Table. 14 FAT Root Folder. 14 FAT Folder Structure. 14 FAT Solder Structure. 14 Disk scan for deleted entries. 14 Define clusters chain for the deleted entry. 14 Disk reaction recovery for the deleted entry. 14	104 106 107 109 110 116 117 122 124 131 133 133 137 138 138 139 48 148 151 153 55
File File Part	Logical Disk Manager (LDM) Overview. 10 Systems. 11 Windows NT File System (NTFS). 11 NTFS Partition Boot Sector. 11 NTFS Master File Table (MFT). 12 Data Integrity and Recoverability with NTFS. 12 Extended File System (exFAT). 12 Volume Layout. 11 exFAT Directory Structure. 12 exFAT Defined Directory Entries. 12 exFAT Cluster Heap. 12 File System (FAT). 12 FAT Partition Boot Sector. 14 FAT File Allocation Table. 14 FAT Folder Structure. 14 FAT Folder Structure. 14 FATS2 Features. 14 Disk scan for deleted entries. 14 Define clusters chain for the deleted entry. 14 Data Scan for deleted entries. 14 Data Scan for deleted entries. 14 Data Scan for deleted entries. 14 Data Scan for the deleted entry. 14 Data Bernel MBR. 14	104 106 107 109 110 117 122 124 131 133 137 138 138 139 48 151 153 55 156
File File Part	Logical Disk Manager (LDM) Overview. 10 Systems. 10 Windows NT File System (NTFS). 1 NTFS Partition Boot Sector. 1 NTFS Master File Table (MFT). 1 Data Integrity and Recoverability with NTFS. 1 Data Integrity and Recoverability with NTFS. 1 Extended File System (exFAT). 1 Volume Layout. 1 exFAT Directory Structure. 1 exFAT Defined Directory Entries. 1 exFAT Cluster Heap. 1 File System (FAT). 1 FAT Partition Boot Sector. 1 FAT Partition Boot Sector. 1 FAT Root Folder. 1 FAT Root Folder. 1 FAT Solder Structure. 1 FAT32 Features. 1 Disk scan for deleted entries. 1 Define clusters chain for the deleted entry. 1 Clusters chain recovery Process. 1 Damaged MBR. 1 Partition is deleted or Partition Table is damaged. 1	104 D6 107 109 110 117 122 124 131 133 137 138 138 138 138 138 138 138 155 156 158
File File Part	Logical Disk Manager (LDM) Overview. 10 Systems. 11 Windows NT File System (NTFS). 1 NTFS Partition Boot Sector. 1 NTFS Master File Table (MFT). 1 Data Integrity and Recoverability with NTFS. 1 Data Integrity and Recoverability with NTFS. 1 Extended File System (exFAT). 1 Volume Layout. 1 exFAT Directory Structure. 1 exFAT Defined Directory Entries. 1 exFAT Cluster Heap. 1 File System (FAT). 1 FAT Partition Boot Sector. 1 FAT Root Folder. 1 FAT Root Folder. 1 FAT32 Features. 1 Partition So the deleted entry. 1 Clusters chain recovery for the deleted entry. 1 Itom Recovery Process. 1 Damaged MBR. 1 Partition is deleted or Partition Table is damaged. 1	104 D6 107 109 110 117 122 124 131 133 133 133 133 137 138 138 139 48 151 153 55 156 158 160

Legal Statement

Copyright © 2013, LSOFT TECHNOLOGIES INC. All rights reserved. No part of this documentation may be reproduced in any form or by any means or used to make any derivative work (such as translation, transformation, or adaptation) without written permission from LSOFT TECHNOLOGIES INC.

LSOFT TECHNOLOGIES INC. reserves the right to revise this documentation and to make changes in content from time to time without obligation on the part of LSOFT TECHNOLOGIES INC. to provide notification of such revision or change.

LSOFT TECHNOLOGIES INC. provides this documentation without warranty of any kind, either implied or expressed, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. LSOFT may make improvements or changes in the product(s) and/or the program(s) described in this documentation at any time.

All technical data and computer software is commercial in nature and developed solely at private expense. As the User, or Installer/Administrator of this software, you agree not to remove or deface any portion of any legend provided on any licensed program or documentation contained in, or delivered to you in conjunction with, this User Guide.

LSOFT.NET logo is a trademark of LSOFT TECHNOLOGIES INC.

Active@ UNDELETE Overview

Active @ UNDELETE is a software application designed to help you restore your lost data from deleted files, folders or even partitions.

Main Features short list

- Recover deleted files and folders.
- Detect deleted partitions and restore them or recover data from them.
- Create a **Disk Image** for safe data restoration.
- Perform an Advanced Scan and organize the result using Scan Result View.
- Restore data from damaged RAID-system drives.
- Work and recover data form dynamic RAID.
- Manage existing partitions or create new once using **Partition Manager** tool.
- Edit disk content with the advanced **Disk Editor** tool.
- Preview files before restoring.
- Supports HDDs larger then 2TB.

List of supported File Systems

- NTFS
- FAT
- FAT32
- exFAT
- HFS+
- Ext2/Ext3/Ext4
- UFS
- Windows 8, Windows 7, Windows 2000, Windows 2003, Windows Server 2008, Windows XP, WinPE
- Administrators privileges required to install and run software
- Pentium processor or compatible
- 30 MB available on hard disk
- 64 MB of RAM or more
- Internet Explorer 8 or later, Mozilla Firefox 1.0 or later
- Mouse or other pointing device

Getting Started with Active@ UNDELETE

Active@ UNDELETE is designed to explore and browse all data storage devices on your computer in different ways to find and recover lost data. All information in the application is organized in tabbed views that provide easy access to information for different purposes.

New to Active@ UNDELETE?

To familiarize you with the *Active@ UNDELETE* workspace, read the following topics in this guide:

- Active@ UNDELETE Views And Windows on page 6
- Search for deleted Files and Folders on page 31
- *File Filter Toolbar* on page 33

Ready to Use?

- Recover deleted Files and Folders on page 14
- Restore Partition on page 23
- Virtual RAID Assembly on page 20
- Decrypt recovered files on page 21

Step-by-step guided tasks

- Active@ UNDELETE Recovery Wizards Overview on page 75
- *Restore a Deleted Partition Wizard* on page 82
- Create a New Partition Wizard on page 84
- Active(a) UNDELETE Disk Image Wizards Overview on page 87

Advanced Tools

- Partition Manager Overview on page 48
- Disk Editor Tool on page 54
- *Virtual RAID Assembly* on page 20

Active@ UNDELETE Views And Windows

All information in the application is organized in tabbed views. Four of the main views are:

<i>Recovery Explorer View</i> on page 7	The main (default) view of Active@ UNDELETE. In this view you can see all available Data Storage Devices and Logical Drives, Assembled RAIDs and opened Disk Images.
<i>Logical Drive Scan Result View</i> on page 7	The Drive Scan Result View displays all files detected after a logical drive scan.
<i>Physical Device Scan View</i> on page 8	Shows scan results made in context of Data Storage Device.
Search Results View on page 10	This view is used to display search results after the search in corresponded context.
Application Log on page 11	This log screen monitors each action taken by the application and displays messages, notifications and other service information.

Welcome View on page 11

Summary view with main tools, wizards and recent activity shortcuts.

To browse through each of these views, click on each tab in turn. You may also open a view from the **View** menu.

To close the current view at any time, press **CTRL+F4**. To open any closed view, select it from the **View** menu.

The status bar, at the bottom of the workspace shows the current status of the application or status of the activity in progress. When Active@ UNDELETE is idle and ready to perform an operation, the status displays "*Ready*".

To toggle the status bar click **View** > **Status Bar**.



Note: When you run Active@ UNDELETE, the application gathers information about disks and partitions available to the system. During this preliminary operation, the status bar displays "Initializing..." and application prevents most other operations from starting. **Application Log View** shows detailed information about the initialization stage.

To modify the information displayed in columns in a table list, right-click any column header and select or clear columns from context menu.

Recovery Explorer View

The main view in Active@ UNDELETE is **Recovery Explorer View**. The view tab label displays "My Computer".

This is the default view that you see after the application starts. It displays the hierarchical structure of all devices and drives, Virtual RAIDs and opened Disk Image. Scan Results appear here if you scan a device. To collapse or expand an item in this tree, click the arrow sign next to the item name.

Recovery Explorer shows its content in to modes:

Expert Device View (default)	At this mode, all available Data Storage Devices with hierarchy of partitions and Logical Drives are present; Use this mode for advanced features, such as Advanced Device Scan or Virtual Partition Management;
Simple Drive view	At this mode, only accessible Logical Drives are present;

Toggle between Simple Drive view and Expert Device view can be done by toolbar button.

To perform an action on any item (Data Storage Device, Drive etc.) select this and choose a command from:

- Toolbar at the top of the view;
- Menu Action;
- or from the right-click context menu.

To add an item to the Recovery Toolbox, select the check box next to the item.

The **Properties Panel** displays default properties for each selected item. Updates to these properties appear dynamically along with commands and activities performed in the workspace. To toggle the Properties Pane click **View** > **Properties** pane.

Logical Drive Scan Result View

Logical Drive (Partition) scan results view displays all files detected after a logical drive scan. To make the result easier to read, you may group detected files by:

- Extension;
- Application;
- Date (Created, Accessed and Modified);

Save Scan Results 🛛 Recover Checked	ecover All 📿 Search		
View 🗸 Group by 🗸 Filter by:	>		
Name File Count	Name St	atus Size Created	Accessed (Deleted) Attributes
🔺 📄 🍉 Major (D:)	🔄 🏐 !Downloads 🛛 H	ealthy 7.52 GB 19/02/12 11:10:12	07/03/12 12:26:05 RD
Downlo File Grouping by Date	📄 🏐 \$Extend Sy	vstem 20.1 MB 19/02/12 11:10:02	19/02/12 11:10:02 HSD
Extend Extensions and Associated	📄 🏐 DAEMON Tools Lite H	ealthy 25.8 MB 12/04/12 09:41:04	12/04/12 09:41:07 D
DAEMO Application	📄 🏐 Program Files 🛛 H	ealthy 0 bytes 19/02/12 21:16:29	12/04/12 10:58:30 D
▷ □ □ □ ▷ □ □ □ ○ □ □ □	The second secon	ealthy 1.75 GB 19/02/12 23:52:31	19/02/12 23:58:46 D

Figure 1: File grouping in Volume Scan Result

To make scan results easier to read, you may do the following:

- To sort the list by a column in ascending order, click the column header.
- To sort the list by the same column in descending order, click the column header a second time.
- To show a list that is reduced in size by a filter, select one of the preset options in the File Filter toolbar.
- To add an item to the Recovery Toolbox, select the check box next to the item.

File Filter Toolbar can be used to narrow down scan results. For more information see: *File Filter Toolbar* on page 33.

Advanced File Search can be used to enhance simple File Filtering with more searching criteria, such as File Attributes, File Size etc. See *Search for deleted Files and Folders* on page 31 for details.



Tip: It is recommended to save scan results for later use.

When you have found all files you looking for - proceed to *Recover files and folders* on page 19.

Physical Device Scan View

Physical Device scan view is used to review scan results, such as Partitions and Files Detected by Signature, made on Data Storage Device.

Scan 🔚 Save Scan Results	Open in Hex	Editor 🤘	Load Scan	Results	Resume	Scan 🧖 Crea	te Virtual Partition	Partition Filte	r 🧊 Remove	from Scan Result
View Group by	2						,	•		
\\.\PhysicalDrive4 - Partition View	Int	terrupted s	scan can be							
6.72 GB Unallocate 60.5 GB Primary NT	n_ex (G:) 59 MB Loi 1.	resu 98 GB Una	med M2_106 (H: 1.95 GB Logi	6.84 GB Ur	nallocated		lont (W:) 15.4 GB Primary NT	backup (J:) 372 GB Primary	NTFS	
Device Scan History			x	Name	File S, tem	n Status	Restore Status	Total Size	First Sector	Total Sectors
Name	Status	Partitions	Files				,		1	
▲ 🙀 \\.\PhysicalDrive4 scans				🥥 cleo (8:)) NTFS 🗋	Excellent	Can be Restored	6.84 GB	149774058	1434598
Disk Scan at 21/05/12 11:38:16 [1,0]	Incomplete	1	0		_					
Disk Scan at 21/05/12 11:39:42 [1,0]	Incomplete	1	0		S	Scan progress ind	icator -			
Disk Scan at 21/05/12 11:39:55 [0,0]	Incomplete	0	0		9	shows incomplete	e scan.			
Disk Scan at 21/05/12 11:40:29 [46,0]	Completed	46	0							

Figure 2: Interrupted Physical Device Scan



Tip: It is highly recommended to save scan results for later use: *Work with scan results* on page 29

Device View Control

In Device Scan view, scanned devices represented by **Device View Control**. For each selected scan, Device View control shows scan progress indicator: blue stripe means scan is incomplete and solid green stripe - scan is complete for selected range. All interrupted (incomplete) scans can be resumed by clicking **Resume** button in view's toolbar or by command **Resume Scan** in item context menu.

If detected partition is selected, its relative position and scanned size is also displayed on Device View Control indicating is this partition is recoverable or not.

Open in Hex Edi	tor 🍯	Load Sca	an Results	Resu	me Scan	R	Create Virtua	l Partition	Partition Filter	Remov	e from Scan Result
									-		
_ex (G:)	M	12_106 (H	t:)				lont (V	N:) ba	ackup (J:)		
9 MB Loj 1.98	GB Una 1	.95 GB Lo	gi 6.84 G	GB Unallocate	d (2 GB Primary N	ITFS	
					_	Use pa	rtition filter t	o narrow down	-		
				~	= 0		Scantes	suit		T 1 1 0	
Chattan Day		Cites 1	Name		Hie Syst	em	Status	Restore Status		Total Size	First Sector 10
Status Pa	rtitions	riles	BC	OT985E (22-)	FΔT		Excellent	Overlapped par	tition detected	1 41 MB	146132511
Incomplete 1				101000E (EEI)	1741		Excellent	Overlapped par	tition detected	1.41 MB	146455928
Incomplete 1		Scar	n progres	s indicator -	scan is		Excellent	Overlapped par	tition detected	1.41 MB	146152671
Incomplete 0		CO	mpleted	for selected	range		Excellent	Overlapped par	tition detected	1.41 MB	146415183
Completed 46		0		SK 03 (28:)	FAI		Excellent	Overlapped par	tition detected	1.41 MB	146149791
Excellent			🖉 🙆 DIS	SK_03 (37:)	FAT		Excellent	Overlapped par	tition detected	1.41 MB	146418063
Excellent			🔵 DR	RDOS (17:)	FAT		Excellent	Overlapped par	tition detected	201 MB	146104666
Excellent			🔵 DR	RDOS (33:)	FAT		Excellent	Overlapped par	tition detected	201 MB	146395978
Excellent			🥥 Lo	cal Disk (15:)	FAT		Bad	Overlapped par	tition detected	10.1 MB	145976268
Excellent			🔵 Lo	cal Disk (21:)	FAT		Excellent	Overlapped par	tition detected	1.41 MB	146129631
Excellent			🥥 Lo	cal Disk (32:)	FAT		Bad	Overlapped par	tition detected	10.1 MB	146312699
Excellent			🔵 Lo	cal Disk (45:)	FAT		Excellent	Overlapped par	tition detected	1.41 MB	146458845
Excellent			🕘 Lo	cal Disk (51:)	FAT		Bad	Overlapped pa	tition detected	10.1 MB	147841682
Bad			🛛 🥥 Lo	cal Disk (54:)	NTFS		Bad	Overlapped par	tition detected	512 MB	146917356
Excellent			🛛 🥥 Lo	cal Disk (55:)	NTFS		Bad	Overlapped par	tition detected	512 MB	147965612
Bad			🛛 🥥 M2	2_106 (10:)	FAT32		Excellent	Overlapped par	tition detected	1.95 GB	145677483
Excellent			🛛 🥯 NC	D NAME (11:)	FAT		Excellent	Overlapped par	tition detected	1.41 MB	145747661

Figure 3: Complete Physical Device Scan

Working with detected partitions

Detected partitions displayed with their status to be recovered and overall partition integrity. When partition *Recover Status* is "Can be recovered" then this partition can be restored as part of disk partitioning. To restore detected partition select it in view and click **Restore** button in view's toolbar or use **Restore** command from item's context menu.

If partition cannot be restored by any reason, data from this partition still can be recovered. To do so, partition must be scanned (as regular Logic Drive) and files needs to be selected individually and recovered to safe location.

n Results Open in Hex Editor 🧹	Load	Scan Results	Scan	Restore Par	tition Open in	Hex Edito)r	
,								
i les (I:) B Primary NT 259 MB Log 1.98 GB Una	M2_100 1.95 GB	5 (H:) Logi 6.84 GB	Unallocated		lont (W:) 15.4 GB Primary NT	backu 372 GE	p (J:) 3 Primary NTFS	
	× Name	-	rue stem	Status	Restore Status		Total Size	First
Status Partitions		DISK_03 (21:	Destars partition	indicator ob	artition of	detected	1.41 MB	
15	🗌 🥘 i	ISK_03 (30:	partition be restor	red (green) g	r not (red) artition (detected	1.41 MB	
L2 11:55:33 [46,0] Completed 46	ا 🎯	DRDOS (10:)	101	EXECUTION C	overapped partition of	detected	201 MB	
Excellent	i 🖉 🙆 🛙	DRDOS (26:)	FAT	Excellent	Overlapped partition of	detected	201 MB	
Excellent	🖉 🙆 l	Local Disk (14:)	FAT	Excellent	Overlapped partition of	detected	1.41 MB	
Excellent	🖉 🙆 l	Local Disk (25:)	FAT	Bad	Overlapped partition of	detected	10.1 MB	
Excellent	🥘 l	Loca Disk (38:)	FAT	Excellent	Overlapped partition of	detected	1.41 MB	
Excellent	🖉 🙆 l	Local Disk (44:)	FAT	Bad	Overlapped partition of	detected	10.1 MB	
Excellent	🖉 🕘 l	Local Disk (47:)	NTFS	Bad	Overlapped partition of	detected	512 MB	
Excellent	🖉 🙆 l	Local Disk (48:)	NTFS	Bad	Overlapped partition of	detected	512 MB	
Excellent	🖉 🍥 l	Local Disk (8:)	FAT	Bad	Overlapped partition of	detected	10.1 MB	
Excellent	I 🕘 I	M2_106 (3:)	FAT32	Excellent	Overlapped partition of	detected	1.95 GB	

Figure 4: Detected partition indicator

Working with Files detected by signatures

Files detected by signatures are shown under related Disk Scan item and combined in groups by signature.

Scan 📄 Sav	e Scan Results 🛛 👔	Load Sca	n Res	ults Recover	r Checked	Recover	N Fi	le Preview	Open in He	ex Editor	
🗍 View 🖕 Group by 🗸						Polativo pos	ition of				
\\.\PhysicalDrive4 - Partition	n View					detected	file	_			
6.72 GB Unallocated	alotoffiles (I:) 60.5 GB Primary NTFS	on_e 259 i	x (G: MB Lo) gic 1.98 GB Unallo	M2_106 (H) 1.95 CO Logi	ca 6.84 GB	Unallocate	d		lont (W:) 15.4 GB Primary NTFS	ł
Device Scan History		x	Nam	e î	/	Count	Status	Size	First Sector	Total Sectors	
Name		Status		E Found_14574561	17_356892.jpg		Deleted	349 KB	14574561	7 698	
▲ 🍓 \\.\PhysicalDrive4	scans			E Found_14574631	L5_220939.jpg		Deleted	216 KB	14574631	i 432	
a 💣 Disk Scan at 22	/05/12 18:12:05 [0,114]	Incomp		E Found_14/74674	47_291935.jpg		Deleted	285 KB	145746747	/ 571	
4 📄 🏐 Files Det	tected By Signatures			Found_14574731	18_171112.jpg		Deleted	167 KB	145747318	335	
📄 🖬 Bitm	ap Images			Found 14652059	97_9894.jpg		Deleted	9.66 KB	146520593	7 20	
Icon	Files			E Found_14652062	29_141647.jpg		Deleted	138 KB	146520629	277	
📄 🔤 JPEG	Images			Found_14652091	L7_182008.jpg		Deleted	178 KB	146520917	7 356	
				Found_14652128	35_415803.jpg		Deleted	406 KB	14652128	i 813	
				E Found_14652210	01_176398.jpg		Deleted	172 KB	14652210	L 345	
				E Found_14652245	53_395721.jpg		Deleted	386 KB	14652245	3 773	

Figure 5: File Detected by signatures

Search Results View

The **Search Results view** appears after you perform a *Search for deleted Files and Folders* on page 31. The top panel displays the results of the search in a list.

To make this list easier to read, you may do the following:

- To sort the list by a column in ascending order, click the column header.
- To sort the list by the same column in descending order, click the column header a second time.
- To show a list that is reduced in size by a filter, select one of the preset options in the File Filter toolbar.

View _ Group by _ Filter by:		Я				B I	🖏 Rec	over	🍋 File	Preview	0	pen in Hex Editor
Name	Size	Attributes	Created		Path					^		
🔲 🚳 DTCommonRes.dll	4.64 MB	AI	11/04/12 05	:53:54	D:\DAI	EMON	N Tools Lit	e				
DTGadget32.dll	299 KB	AI	11/04/12 05	:53:10	D:\DAI	EMON	V Tools Lit	e				
DTGadget64.dll	360 KB	AI	11/04/12 05	:53:10	D:\DAI	EMON	V Tools Lit	e				
📄 🚳 Engine.dll	3.57 MB	AI	11/04/12 05	:53:20	D:\DAI	EMON	V Tools Lit	e				
🔲 🚳 imgengine.dll	374 KB	AI	06/04/12 06	:22:56	D:\DAI	EMON	V Tools Lit	e				
🔲 🚳 AFK.dll	13.0 KB	AI	11/04/12 05	:52:28	D:\DAI	EMON	N Tools Lit	e\Lang				
🔲 🚳 ARA.dll	87.5 KB	AI	11/04/12 05	:52:26	D:\DAI	EMON	V Tools Lit	e\Lang				
🔲 🚳 BGR.dll	118 KB	AI	11/04/12 05	:52:28	D:\DA	EMON	V Tools Lit	e∖Lang				
🔲 🚳 BIH.dll	107 KB	AI	11/04/12 05	:52:28	D:\DAI	EMON	V Tools Lit	e\Lang				
CAT.dll	111 KB	AI	11/04/12 05	:52:30	D:\DA	EMON	V Tools Lit	e∖Lang				
🔲 🚳 CHS.dll	45.5 KB	AI	11/04/12 05	:52:24	D:\DAI	EMON	V Tools Lit	e\Lang				
🔲 🚳 CHT.dll	45.5 KB	AI	11/04/12 05	:52:24	D:\DAI	EMON	V Tools Lit	e\Lang				
CSY.dll	103 KB	AI	11/04/12 05	:52:24	D:\DAI	EMON	V Tools Lit	e\Lang				
DAN.dll	99.5 KB	AI	11/04/12 05	:52:28	D:\DA	EMON	V Tools Lit	e\Lang				
DEU.dll	122 KB	AI	11/04/12 05	:52:24	D:\DAI	EMON	V Tools Lit	e∖Lang				
ELL.dll	92.0 KB	AI	11/04/12 05	:52:28	D:\DA	EMON	V Tools Lit	e∖Lang				
	09 5 1/0	AT	11/0//12:05	52.24	D-\DAI		Tools Lit	al land				

To recover an item in this list, right-click the item and choose **Recover** from the context menu or click **Recover** button in toolbar.

To preview an item, select it and click **File Preview**.

To change search criteria and repeat the search at the same location, click Search Again.



Note: You can create a custom filter for this list. For more information see *File Filter Toolbar* on page 33.



Note: For information about how to start a search, see *Search for deleted Files and Folders* on page 31

Application Log

This log view monitors each action taken by the application and displays messages, notifications and other service information. Use the messages in this screen to observe and further understand the flow of the recovery process.

We recommend that you attach a copy of the log file to all requests made to our technical support group. The entries in this file will help us resolve certain issues.

To prepare a log file, turn on **Display Trace Events** and **Write Log on Disk** options in the **Preferences** dialog box.

It is best to save the log file to a physical disk that is different from the disk that holds the deleted data. By doing this, you reduce the risk of writing over the data that you are trying to recover.

Welcome View

This view contains main tools, wizards and recent activity shortcuts divided in groups.

Active 🥹	UI	NDELETE
Product version: 8.1.17 Administrator Rights: ∳ Granted Operating System: Windows 7, Ultimate		
Getting Started	۶	Here are some commonly used actions when working with Active@ UNDELETE - Data Recovery Toolbox on a regular basis.
Recently Used		Advanced Recovery Explorer Start Recovery Explorer in Advanced Mode.
Data Recovery Wizards		Access Physical Devices to detect Deleted or Damaged partitions and
Partition Management		Undelete Files Scan Volumes (Logical Drives) for deleted files and folders and files by their
Disk Image Management		signatures
Advanced Tools		Scan Disks for deleted or damaged partitions
Support		
Version Info		
Show this view on startup		Close

Using Active@ UNDELETE Overview

File recovery

Recover deleted Files and Folders on page 14

Scan Disk (Physical Device) on page 23

Scan for files by their signatures on page 17

This is one of the essential features of Active@ UNDLETE. To recover accidentally deleted files, simply scan the drive where they were deleted, then browse scan results in familiar Windowsexplorer like browser, search and filter results, select required files and recover them to safe location. You can preview scan results first to confirm that the detected files are exactly the once you need.



Note: For *DEMO* version recovered file size is limited to 64kb.

In some cases, you seek files from drives are not existing anymore - those partitions either deleted or overwritten by new one. It is still chance to recover some files in such condition! You have to located deleted partitions first and scan them as they are existing partitions and recover all detected files you need;

Active@ UNDELETE can find files by their unique format specification (signature) even if file can not be found in *Partition File Table*. For now, we can recognise various file formats:

- Microsoft Office Documents.
- Formatted Text files.
- Compressed Archives.
- Images and Camera Raw files.
- Music and Videos.
- QuickTime Multimedia files.

See *Supported File Signatures* on page 34 for complete list of default file signatures.

User can create **custom**, user defined **File Signature Templates** to be used to detect files during low level disk scan by customized file signatures. See *Custom (user defined) file signature templates* on page 35 for details.

Disassembled *RAID* array can be virtually recreated by Active@ UNDELETE and some of the files located on these array can be recovered;

Partition Management

Restore detected partition on page 25

Virtual RAID Assembly on page 20

You partition is gone? Accidentally deleted by user or by malicious software it is still chance it can be restored if not overwritten yet. Scan hard disk for deleted partition and use **Restore** command to get

	your partition back! We recommend you to restore your important data first;
Partition Manager Overview on page 48	By using small Partition Manager module in Active@ UNDELETE you can execute basic partition manipulation such as creation, formatting and delete. It can be useful during partition recovery operations;
Disk Images	
Disk Image Overview on page 66	We advice to create <i>Disk Image</i> of a drive you work with before any actual recovery or partition restoration. It may prevent loosing data in accidental writing of cumulative hardware malfunction;
Advanced Tools	
<i>Edit Boot Sectors</i> on page 52	For advanced operations, you can manipulate partition table and boot sector attributes by using template dialogs;
Disk Editor Tool on page 54	Advanced and integrated in Active@ UNDELETE environment disk editor - read and write data on low level.
Rollback Partition Changes on page 66	If all your manipulation with hard disk partitioning was made by using Active@ UNDELETE you can rollback (e.g. undo) all changes you have made in few clicks.
<i>File Preview</i> on page 30	To confirm that the file you have detected is exactly the file you seek, you can use <i>File Preview</i> feature before the actual recovery. It also helps to confirm file integrity first. Some restriction applies for <i>DEMO</i> version;

Recover deleted Files and Folders

After you can see partitions on a device, the file recovery process consists of three stages.

1. Scan Disk

Select Volume (Logical Drive) scan in Recovery Explorer and scan the chosen drive for deleted files.



See: Scan a Volume (Logical Drive) for deleted files on page 15

2. Analyze Scan Results

A Logical Drive scan result appears in the *Logical Drive Scan Result View* on page 7 where results can be reviewed and files selected for recovery.

File Grouping - detected files can be grouped for better analyzing by using the Group By drop-down menu in the toolbar. Detected files can be grouped by:

- File Extensions;
- By Associated Applications;
- By Date (Created Date, Modified Date and Accessed or Deleted Date);

Save Scan Results Recover	Checked	Recover	C Search				
Group by 🗸 Filter by:			5) B		
Name	Status	Size	e created	Date accessed	Attributes	ID	
🔺 🥅 🍉 Tutorial (K:)							
Image: Second	System	Userke	ecover button	to recover	HSD	11	
Image: Skiele	System	sele	ected files and	tolders	HSD	52	
[1] [1] _953649_	Deleted	U Dytes 14	-IVI8F-15 U5:U2:52	14-IVI8I-15 U5:02:52	D	39	
Files by Signatures	Deleted	4.27 MB			D	-1	
📄 📺 MSI80791.tmp	Deleted	0 bytes 14	-Mar-13 03:02:20	14-Mar-13 03:02:20	D	39	
🛛 📄 🥘 North	Healthy	837 KB 20	-Jan-13 12:13:14	22 Jan-13 09:47:22			
📄 🚞 System Volume Information	System	20.0 KB 22	-Jan-13 11:56:00	22-Jan-13 11:50:00	Files dete	ected by signatures	
Vacation	Healthy	297 KB 20	-Jan-13 12:13:46	23-Jan-13 09:47:33	- located	in this virtual folder	
AHCI.doc	Deleted	41.5 KB 23	-Jan-13 09:46:09	23-Jan-13 09:46:09	A	UU	
🔲 👜 AHCI.dot	Healthy	41.5 KB 23	13 09:46:09	23-Jan-13 09:46:09	Α	62	
🔲 🔛 Hydrangeas 😋	Healthy	581 KB 20	-Jan-13 1 23	20-Jan-13 12:12:23	A	38	
Hydrangeaa, or Healthy 581 KB 20-Jan-137 22 20-Jan-13 12:12:23 A 38 Deleted file [marked gray] - it can be recovered! System files and folders [marked dark red]							

Search and Filtering - detected files can be filtered by name, extension or deleted status by using the *File Filter Toolbar* on page 33. For more narrow results *Search for deleted Files and Folders* on page 31 can be used.

3. Recover Detected Files

You may recover damaged or deleted files and folders directly from the *Logical Drive Scan Result View* on page 7 or the *Search Results View* on page 10. See: *Recover files and folders* on page 19

Scan a Volume (Logical Drive) for deleted files

Scanning logical drives is a required step for recovering files and folders. During the scan all deleted (and existing) file and folders are detected. The results of a logical drive scan are displayed in a separate tabbed views: *Logical Drive Scan Result View* on page 7. To initiate a scan of a logical drive:

1. Select volume (logical drive)

In the **Recovery Explorer** view select a volume (logical drive).

- 2. Open the Scan Volume dialog box:
 - From the Recovery Explorer toolbar, click **Scan**.
 - Right-click the selected logical drive and click **Scan** from the context menu.

l acal Distr (Cr)	General Scan Options		
Unknown	✓ Ignore Errors	Use Advanced Scan Algorithm	
	Save scan results as scan competes		
NEW VOLU	Scan results location: D:\temp\scan_res	ults\drive_H_3990-6245_11D11365.scan	
perrot (F:)	Detect files by their signatures		
NITS	Name		Select All
lont (H:)	Image:		Clear All
NTFS	Formatted Text files		
Major (D:)	Compressed Archives Images and Camera Raw file		
NTFS	Bitmap Images	bmp	
	Canon CRW Raw Images	crw	
FAT32	Icon Files	ico	
	JPEG Images	jpg 🔻	
New Volume	Apply the same settings for all selected driv	25	
	· · · · · · · · · · · · · · · · · · ·		

Dialog parameters Multiple drive selection

Additional drives can be selected to scan on the **Logical Drives** list to be scanned simultaneously. At least one logical drive (volume) must be selected.

Ignore errors

Ignore Read and Write errors during the scan process.

Save scan results

If this option is on, a path must be specified where scan results with a unique name will be saved for each scanned drive. Provide valid path if you have this option selected.

Use Advanced Scan Algorithm

Select this option to apply advanced scan algorithm. However, event if scan results may contain more entries then with scan without this option overall scan process may take much more time.

Detect files by their signatures

Select this option to specify exact file types to be detected during the scan. With this option, device scan reads each disk sector trying to reconstruct any possible data related to unique file format.

Apply the same settings to all selected drives

All scan options above, can be selected for each drive individually or, when this check box is selected, to be the same for all selected logical drives.

Click Scan to initiate scan of selected logical drives (volumes) scan.

3. Scan selected volumes.

1	Processing 54%	9	-			_		?	x
	Scanning Volume Major (D:) Stop								
	Detected 262213 file(s) and 13739 folder(s)								
	Progress Proces	s Options	Dutput						
	Name	Status	Progress		Estimated	Elapsed	Files	Folders	
	Iont (H:)	Completed			00:00:00	00:00:10	34565	133	
	🚳 Major (D:)	Scanning	5	5% <mark>.</mark>	00:00:08	00:00:12	262213	13739	
1									
					Close th	is dialog w	hen exec	cution com	pletes
(L									

During the scan:

- To display or hide scanning events and progress details toggle **More\Less Info** button at any time.
- To terminate the scan process, click **Stop** at any time. Results may be not accurate or complete. After the scan completes you will see scan results in the *Logical Drive Scan Result View* on page 7.
- **4.** Review scan results

A Logical Drive scan result appears in the *Logical Drive Scan Result View* on page 7 where results can be reviewed and files selected for recovery.

Save Scan Results Recover	Checked		ver 📿 Sea	rch					
👔 Group by 🗸 Filter by:									
Name	Status	Size	e created	Date accessed	Attributes	ID			
🔺 🥅 🍉 Tutorial (K:)			Deserves but						
>	System	Use	Recover but	ton to recover	HSD	11			
Image: Second	System	s	elected files	and folders	HSD	52			
	Deleted	U bytes	14-IVI8F-15 05:02	:52 14-IVI8F-15 U5:U2:52	D	39			
Files by Signatures	Deleted	4.27 MB			D	-1			
MSI80791.tmp	Deleted	0 bytec	14-Mar-13 03:02	:20 14-Mar-13 03:02:20) D	39			
Image:	Healthy	837 KB	20-Jan-13 12:13	22 Jan-13 09:47:22					
📄 🚞 System Volume Information	System	20.0 KB	22-Jan-13 11:56	:00 22-Jan-13 11:50:00	Files det	ected by sig	natures		
Vacation	Healthy	297 KB	20-Jan-13 12:13	46 23-Jan-13 09:47:33	 located 	in this virtua	al folder		
AHCI.doc	Deleted	41.5 KB	23-Jan-13 09:46	09 23-Jan-13 09:46:09	A	UU			
🔲 🛄 AHCI.dot	Healthy	41.5 KB	23-> 13 09:46	09 23-Jan-13 09:46:09	Α	62			
📄 🔛 Hydrangeas 🛛 g	Healthy	581 KB	20-Jan-13	23 20-Jan-13 12:12:23	A	38			
Pryonanges, Q Preading Solide Booland Construction A System files and folders [marked dark red] - it can be recovered!									

(

Note: We recommend you to save scan results to designated location for later use - you can use saved scan results to save time on repeated scanning of same volume.

Scan for files by their signatures

Files on hard drive can be detected by their unique file signatures. Active@ UNDELETE can detect these files (see *Supported File Signatures* on page 34 for exact list of file types) during *Scan a Volume (Logical Drive) for deleted files* on page 15 or *Scan Disk (Physical Device)* on page 23. In first case, scanning will be limited by volume boundaries when by scanning physical disk, you can specify custom boundaries of disk surface to scan.

Volume (logical drive) Scan

During volume of scan you have to select file signatures on scan dialog and they will be detected (if any) among other deleted or live files on selected volume(s) only.



Note: See *Scan a Volume (Logical Drive) for deleted files* on page 15 for more information.

Physical Disk Scan

T.

Files by signatures can be also detected during scan of disk surface not limited by volume boundaries.





Note: See Scan Disk (Physical Device) on page 23 for more information.

Evaluate scan results

Detected files (if any) grouped in special virtual folder - *Files by Signatures*. Due to particular qualities of this algorithm, it is impossible to recover original file names, date and other attributes. To evaluate integrity of some of the detected files you can use *File Preview* on page 30 feature.

Group by 🗸 Filter by:			5		2 🔒	
Name	Status	Size	Date created	Date accessed	Attributes	ID
a 📄 🍉 Tutorial (K:)					-	
>	System	20.1 MB	20-Jan-13 12:05:01	20-Jan-13 12:05:01	HSD	11
Im	System	341 KB	20-Jan-13 12:45:51	20-Jan-13 12:45:51	HSD	52
	Deleted	0 bytes	14-Mar-13 03:02:32	14-Mar-13 03:02:32	D	39
🚺 🚺 🍙 Files by Signatures	Deleed	4.27 MB			D	-1
🔲 📄 0000012с.јрд	Doleted	164 KB				-1
🔲 🔛 0000adc8.jpg	Dele	581 KB				-1
🔲 🔛 0000b254.jpg	Deleted	138 KB				-1
📄 🔛 0000b36c.jpg	Deleted	136 KB				-1
🔲 🔛 0000b47c.jpg	Deleted	A KR				-1
🕅 🔛 0000b5a0.jpg	Files,	detected	by theirs signat	ures, will		-1
🔲 🔛 0000b664.jpg	be	collected	in one special f	older.		-1
🔲 🔛 0000b740.jpg						-1
🔲 🔛 0000b814.jpg	Deleted	99.2 KB				-1
🔲 🔛 0000fe04.jpg	Deleted	606 KB				-1



Note: Amount of selected files signatures directly impacts on total scan time.

Recover files and folders

You may recover damaged or deleted files and folders directly from the *Recovery Explorer View* on page 7, *Logical Drive Scan Result View* on page 7, *Physical Device Scan View* on page 8 and *Search Results View* on page 10. Recovering deleted files and folders is one of the essential features of Active@ UNDELETE. To recover detected files:

1. Select files

Select files in any view mentioned above using cursor selection (Use **Shift** or **Ctrl** keys for multiselection) or by setting check boxes for each individual file o folder.



Tip: You can set check boxes for all selected files or folders by pressing Space key.

2. Click Recover or Recover Checked.

Depending on you selection in view's toolbar or use context menu commands.

3. Enter destination path where recover detected path and other options as necessary.

Select Destinatoin Path to recover files and fok on the same drive of original file. Recover files to: d:\templ/recovered\	ders and file naming options. Beware to recover files
Naming options Use original file names (recommended) Rename files to: recovered_+(00001)	Existing files conflict resolution Generate unique file name (recommended) Ask before overwrite Overwrite without prompt Skip existing files
Options Create original folder (group) structure V Browse recovery destination folder, after recovery	Recover Named Streams
Create detailed log of recovered files (Impact Perfor V Use Disk Lock V Ignore Disk Lock Errors Restore Defaults	mance) Ignore Write Errors I Ignore Read Errors Recover Cancel

Naming options Use original file names

Names of detected files will be preserved only if no file with the same name already exists in the destination directory.

Rename files

All files will be renamed by their given specified file root name and added enumeration ID. File extensions remain intact.

Existing files conflict resolution

Unique file name

If a file with the same name exists in the destination folder, a file with a unique name will be generated to avoid overwriting.

Ask before overwrite

If a file with the same name already exists in the destination folder, the application will ask the user for a specific action to take.

Overwrite without prompt

All files will be overwritten in the event if they already exist in the destination folder.

Skip existing files

If a file with the same name exists in the destination folder, recovery of a new file will be skipped.

Additional Options

Create Folder Structure

When this option is selected files will be recovered with their original folder structures e.g. original folder hierarchy as it was on the storage source. In case files were organized in groups (date, file extensions, or by an associated application) then such groupings will be created by the folder structure in the location where the files will be recovered to.

Recover Name Streams

With this option on, files will be recovered with their original name streams.

Browse destination folder

Opens the destination folder in the default OS file browser.

Detailed Log

With this option on, the log file contains more detailed information about recovered files.

Use Disk Lock

The source disk will be locked during the file recovery process. It will be unlocked as soon as the process is completed.

Ignore Disk Lock Errors

With this option on, the file recovery process will continue even if locking of the source device fails.

Ignore Write Errors

No error messages will appear and all write errors will be ignored during the recovery process.

Ignore Read Errors

No error messages will appear and all read errors will be ignored during the recovery process.

4. Observe recovery process and verify recovered files in destination folder. Repeat recovery process if necessary.

Virtual RAID Assembly

To open the Virtual Disk Array Assembly dialog, do one of the following:

- From the Tools menu, choose Create Virtual RAID (RAID) command.
- From the Tools tab in Command Bar, choose Create Virtual RAID command.

Select Physical Devi boundaries if new Data Storage Devices availabl	ces from the list of a ded. le for RAID Assembl	available devices in o	correct or	ler and RAID Ty	pe. For each sele	ected device you can specify individually device		
Name	Status	Partitioning	Total Size	Total Sectors	Bytes/Sector			
Aller A	Ready Initialized	MBR (Basic)	932 GB	1953525168	512			
	Read Initialized	MBR (Dynamic)	932 GB	1953525168	512			
	Ready, In Vized	MBR (Basic)	932 GB	1953525168	512			
	Ready, Initian	MBR (Basic)	55.9 GB	117231408	512			
	Read Read	Data Storage D	evices	976773168	512			
6.72 GB L 60.5 GB P 259 I 1.98 C 1.95 C 6.84 GB Unallocated lont (W:) backup (J:) 15.4 GB P 372 GB Primary NTFS 2.49 MB 1 ata Storage I Selected if necessary. Clicking on partition will selected if necessary. Clicking on partition will set baoundaries to that partition 2456300658 343454721								
\\.\PhysicalDrive4 Re	\\\PhysicalDrive4 Ready, Initialize 10 Disk 466 GB 0 976773168 \\\PhysicalDrive4 Ready, Initialize 10 Disk 466 GB 0 976773168 \\\\PhysicalDrive4 Ready, Initialize 10 Disk 466 GB 0 976773168							
Local Disk (1:) 83.6 GB Primary L 134 GB Logical 1 138 GB Logical 1 104 GB Logical 472 GB Unallo 456390658 799845379 Size Select RAID type and Block Size Select RAID type and block size as required								
RAID Options					_			
RAID Type: Mirror (RAID 1)		Block Size:	512 bytes [Default] 🔻 See	ctors per Block:	1		
To create Mirror (RAID	1) select two Data	Storage devices.						
Reset						Create Virtual RAID Cancel Help		

To assemble virtual RAID follow the steps:

- **1.** Select Disks from list of Available Data Storage Devices by double-click or by using check mark.
- **2.** Specify the virtual array type.
- 3. To add disks, do one of the following:
 - Double-click a disk in the Available disks list to move it to the Selected disks list.
 - Click a disk in the Available disks list to select it. To move it to the Selected disks list, click Add.
- 4. To change the order of a disk in the Selected disks list, select it and click Move Up or Move Down
- 5. To remove a disk from the Selected disks list, do one of the following:
 - Double-click a disk in the Selected disks list.
 - Click a disk in the Selected disks list. To remove it, click **Remove**.
- 6. To remove all disks from the Selected disks list, click **Remove All**.
- 7. In Stripe block size, specify the stripe block size in kilobytes (Stripe and RAID-5 arrays only)

Decrypt recovered files

During the recovery of encrypted files to any destination that doesn't support encryption, Active@ UNDELETE creates temporary (*.EFS) files. These files can be decrypted later at any time by using the **File Decryption Tool**.

1. Open the Decrypt Files dialog:

- Use the command tools and select **Decrypt Files** from the main menu.
- From the **Tools** tab in the command bar, choose the **Decrypt Files** command.
- 2. Add temporary recovered encrypted files (*.efs) or open the Decrypted Files log (*.txt) created during recovery by using the **Add** button.

Select	Destinatoin Path to decrypt temporary *.EFS files and additional options if nec	essary.
Decrypt files to:	d:\temp\decrypted\	
Files:	Ad	Id temporary recovered files to decrypt
Options		Existing files conflict resolution
📃 Delete temp	orary files after decryption	Generate unique file name (rec
Browse reco	overy destionation folder, after decryption completes	Ask before overwrite
Create deta	iled log of decrypted files (Impact Performance)	Overwrite without prompt
🔽 Use Disk Loo	k 📝 Ignore Disk Lock Errors 📄 Ignore Write Errors 📝 Ignore Read Error	rs 🔘 Skip existing files
Restore Defaults]	Decrypt

Figure 6: Decrypt files dialog box

Options

Delete temporary files

All temporary recovered encrypted source files will be deleted after decryption.

Browse Destination

The folder where files will be decrypted will be opened by the default OS files browser.

Create Detailed Log

The log files will contain more detailed information about the forthcoming process.

Use Disk Lock

The source disk will be locked during the file recovery process. The disk will be unlocked as soon as the process is completed.

Ignore Disk Lock Errors

With this option on, the file recovery process will continue even if locking of the source device fails.

Ignore Write Errors

No error messages will appear and all write errors will be ignored during the recovery process.

Ignore Read Errors

No error messages will appear and all read errors will be ignored during the recovery process.

Existing files conflict resolution Generate unique file name

If a file with the same name exists in the destination folder, then a file with a unique name will be generated to avoid overwriting.

Ask before overwrite

If a file with a certain name already exists in the destination folder, the application will ask the user for a specific action to take.

Overwrite without prompt

All files will be overwritten even if they already exist in the destination folder.

Skip existing files

If a file with the same name already exists in the destination folder, recovery of that file will be skipped.

3. Set other options if necessary and then click the **Decrypt** button to complete the task.

Restore Partition

If you cannot see partitions on your device, or if you know that partitions are missing, you may first scan a device to find partitions. Restoring a deleted or damaged partition can be done in three stages:

- **1.** Scan a physical device for a deleted or damaged partition.
- 2. Evaluate the scan results and choose the partition to restore.
- **3.** Restore the partition.

Scan Disk (Physical Device)

A physical device is an installed hard disk, Flash card, external USB disk or any device that holds data. You may scan a device two ways:

Scan for Deleted Partitions on page 23

Detect Files by their signatures

Scan unallocated space on disk to detect deleted or damaged partition.

Besides detecting partitions, *Device Scan* can detect files by their unique file signature. To let files by signature to be detected, select **Detect files by their signatures** options when performing *Scan Disk (Physical Device)* on page 23.



Note: Scanning time directly correlated with number of file signatures selected.

Scan for Deleted Partitions

Detected partition can be scanned as any other Logical Drive for Files and Folder. You can scan detected partition to verify partition content before *partition restoration* or to be able to *recover (copy) files* to safe location if partition was deleted or damaged. To scan a physical device for deleted partitions:

- 1. In the **Recovery Explorer** select a disk (physical device) node. Open Scan Disks dialog box:
 - Click Scan button in view's toolbar
 - Click Scan command from context menu
 - Double-click and disk (physical device) node

2. Define scan range and other scan options if necessary

Advance Data Storage Dev and partitions, wi structure, based however it usuali	ed vice Scan processes the whole surface of the physical device searching for all possible logical drives bether they are existing, damaged or deleted. Scan reads each disk sector trying to reconstruct the on residual dues to the drive's system structures that remain on the disk surface. This is a slow prov raives much better results than Drive Scan.	(volumes) drive cess,
\\. \PhysicalDrive4 [976773	168 sectors]	
alotoffi on_ 6.72 GB 60.5 GB 259	M2_ lont (W backup (J:) 1.98 1.95 6.84 GB Unalloc 15.4 GB 372 GB Primary NTFS	2.49 MI
, i i i i i i i i i i i i i i i i i i i	149773995	
Scan entire disk	 Scan Unallocated areas only Select specific area to scan 	
Data Storage Device scan of Detected partitions File Sy Ignore errors	pptions stem lookup: INTFS I FAT32 I exFAT I FAT I HFS+ Ext2/Ext3/Ext4	
Save scan results as so	an completes	
Scan results location:	d:\temp\scan_results\	
	Scan Cancel	Help

Dialog options Multiple drive selection

Additional disks can be selected to scan on the **Physical Disks** list to be scanned simultaneously. At least one disk must be selected.

Scan area

Select scan area using predefined options: **Entire Disk**, **Unallocated Only** or **Specific Range** - use arrow markers to mark scan area;



Note: Scan area markers shown first and last sectors of scanning area. To enter exact start and end sectors to scan click on sector label and enter exact value in text field;

File System lookup

Select desired File System of a partitions to be detected;

Ignore Errors

Ignore disk Read/Write Errors;

Save Scan results

Enter path, where scan results will be saved as soon as scan completed;

Detect files by their signatures

Select this option to specify exact file types to be detected during the scan. With this option, device scan reads each disk sector trying to reconstruct any possible data related to unique file format.



Important: Turn this option off when you only want to detect and restore partition - it will significantly save your scanning time.

Apply the same settings to all selected devices

All scan options above, can be selected for each drive individually or, when this check box is selected, to be the same for all selected logical drives.

3. Click Scan to begin scan process

Edit or Clone Detected Partitions

It may be necessary for you to edit detected partition attributes directly when some attributes are detected incorrectly or need adjustments.

Any detected partition can be cloned (virtually copied) before manually altering partition attributes and properties. We recommend that you edit the clone rather than directly edit the original partition. Any detected partition can be cloned as any times as you want.

Clone detected partition

- 1. Select a detected partition in the **Recovery Explorer** tree.
- **2.** To clone the selected partition, do one of the following:
 - From the Recovery Explorer toolbar, click Clone Partition.
 - Right-click the selected partition and click **Clone** from the context menu.

Edit the boot sector template in detected partition

- 1. Select a detected partition in the Recovery Explorer tree.
- 2. To open the Edit Boot Sector Template dialog box, do one of the following:
 - From the Recovery Explorer toolbar, click **Edit Partition**.
 - Right-click the selected partition and click **Edit Partition** from the context menu.
- **3.** In the **Edit Boot Sector Template** dialog box, edit the *Primary* or *Copy Boot* sectors separately or simultaneously by entering values in designated fields.

Restore detected partition

We recommend that you restore a partition with a certainty status of "Acceptable" or higher.

Before you restore a partition, you may clone or edit the partition directly to adjust its properties.

Here are some rules to follow when restoring a partition:

Assigning a drive letter

- Be aware of the location of executable files or files required by the operating system. Many MS-DOS and Windows programs refer to a specific drive letter when describing a path to executable files.
- Drives A: and B: are usually reserved for floppy disk drives, but you can assign these letters to removable drives if the computer does not have a floppy disk drive.
- Hard disk drives in the computer receive letters C through Z, while mapped network drives are assigned drive letters in reverse order (Z through B).

Setting the partition as active

- You may set only a primary partition as active. You cannot set a logical drive (an extended partition) as active.
- To set a partition as active, the partition must have an MBR (Master Boot Record) as the first sector.
- A computer can only have one active partition per disk.
- The name commonly used for the partition that contains the start-up files is the boot partition. The name commonly used for the partition that contains the operating system files is the system partition.
- The system partition can never be part of a striped volume, spanned volume, or RAID-5 volume.
- The system partition must be a primary partition that has been marked as active for start-up purposes. It must be located on the disk that the computer accesses when starting up the system.
- There can be only one active system partition on a disk at a time.
- You may have multiple basic disks and each disk can have one active partition. However, the computer will only start from one specific disk. If you want to use another operating system, you must first mark its system partition as active before restarting the computer.
- You cannot mark an existing dynamic volume as active. However, you can convert a basic disk containing the active partition to a dynamic disk. After the disk is converted, the partition becomes

a simple volume that is active. If the active partition is not the current system or boot partition it becomes a simple volume and loses its entry in the partition table. Therefore it can no longer be active.

Extended partition

- A computer can only have one extended partition per physical disk device.
- You cannot create an extended partition on a disk if it already has four primary partitions.

Restore Partition

- 1. Select a detected partition in the *Physical Device Scan View* on page 8.
- 2. To open the Restore Partition dialog, do one of the following:
 - From the toolbar click the **Restore Partition** button or use the command action **Restore Partition** from the main menu.
 - Right-click the selected item and click the **Restore Partition** command from the context menu.

Revie \. \Physicall	w and confi Drive4	rm Partiti	on Restor	re optior	15.				
6.72 GB	alotoffi 60.5 GB	on_ 255	1.98 M	12_ ,95 6,	84 GB-Unalloca	lont (W 15.4 GB	backup (J:) 372 GB Primary NTFS		2.49 ME
artition Re Assign Driv	store Optior e Letter: F	ns > 🔻 🔳] Make re	estored p	partition Active	Crea	te Extended Partition all unallocated space set:Size:	Measure	in Sectors
								Restore	Cancel

Dialog Options Assign Drive Letter

To assign a drive letter to the recovered partition, select a letter from the drop-down list.

Make restored partition Active

To set this partition as active, check the **Make restored partition Active** check box.

Create Extended Partition

Before a partition is restored, unallocated space can be set as an extended partition by checking the **Create Extended Partition** check box.

Using Scan Results

After you have completed a device scan, a Scan Results branch appears in the Recovery Explorer tree. Detected partitions are listed in order of their certainty of recovery.



There are 12 attributes that define a partition. In some cases, the application cannot be certain that the found item actually is a partition. The rating in the order of certainty depends on how many attributes

are found and what condition they are in. You may perform the following actions on partitions in the Scan Results branch:

- Stop and Resume a scan on page 27
- Filter detected partitions by certainty on page 28
- Save and Load scan results

Stop and Resume a scan

To stop a physical device scan at any time, press **Stop**. After you stop a scan, a **Scan Results** branch appears in the **Recovery Explorer** tree.

🍞 backup (J:)	Ready	Logical Drive	NTFS	backup	372 GB
Onallocated Space			Unallocated		2.49 MB
a 🎒 Disk Scan Results					
a 🍓 \\.\PhysicalDrive4 scans					
a 🧠 Disk Scan at 10/06/12 12:47:31 [4,0]	Completed	Disk Scan			
🥥 m1 (2:)	Good	Detected Drive	NTFS	m1	2.23 GB
🥟 m1 (3:)	Not Bad	Detected Drive	NTFS	m1	399 GB
🥟 m1 (4:)	Poor	Detected Drive	NTFS	m1	398 GB
🥌 Local Disk (5:)	Bad	Detected Drive	NTFS		2.23 GB
ar Disk Scan at 10/06/12 12:48:45 [0,0]	Incomplete	Disk Scan			

The example above shows how incomplete scan results are indicated. An icon appears next to each node in the Scan Results branch.

Uncompleted Device Scan

An uncompleted (aborted) device scan can be resumed at any time.

🗷 Welcome to Active@ UNDELETE 🖾 💐 R	ecovery Explo	rer 🗵 😡 A	pplicatio	n Log View 🖾 💽 Partition Manager 🖾 🇞 \\. \PhysicalDrive4 scans 🗵
Scan 🔚 Save Scan Results 🧉	Load Scan I	Results	Recove	r Checked Resume Scan Add Virtual Partition F
View Group by View	ncomplete s	can can be		
\\.\PhysicalDrive4 - Partition View	resumed	from last		
	scanning	position.		
6.72 GB Unallocated 60.5 GB Primary NTI	on_ex	(G:)	Unalle	M2_106 (H:) Iont (W:) 15.4 GB Unallocated 15.4 GB Primar
	235 MB	Logit 1.98 GB	onanc	1.55 GB Logica 6.84 GB Onanocated
Device Scan History			×	Nan File System Status Restore Status Total Size
Name	Status	Partitions	Files	
▲ 🍓 \\.\PhysicalDrive4 scans				
a 🧠 Disk Scan at 10/06/12 12:47:31 [4,0]	Completed	4	0	
🥪 Local Disk (5:)	Bad			Incomplete Scan
🦢 m1 (2:)	Good			Indicator shows relative scan
🦢 m1 (3:)	Not Bad			boundaries and progress.
🥟 m1 (4:)	Poor			
Disk Scan at 10/06/12 12:48:45 [0,0]	Incomplete	0	0	· · · · · · · · · · · · · · · · · · ·

To resume a terminated scan:

- **1.** Select a device scan result under the **Scan Results** branch.
- **2.** To resume the scan, do one of the following:
 - From the toolbar, click the **Resume Scan** button.
 - Right-click the selected device scan and click **Resume Scan** from the context menu.

Completed Device Scan

A completed device scan cannot be resumed.

Scan Save Scan Results		oad Scan Resi	ilts 🖏	Recove	er Checked	Resume Scan
View _ Group by -						
\\.\PhysicalDrive4 - Partition View						
6.72 GB Unallocated 60.5 GB Prima) ry NTFS	on_ex (G:) 259 MB Log	ic 1.98 GB	Unallc	M2_106 (H:) 1.95 GB Logica	6.84 GB Unalloca
				~		
Device Scan History				x	l'ame	File System
Name	Sta	itus Pai	titions	Files		
▲ આ PhysicalDrive4 scans					Completed	scan of s
isk Scan at 10/06/12 12:47:31	. [4,0] Co	mpleted 4		0	unallocated	area.
Local Disk (5:)	Ba	d			🥟 m1 (3:)	NTFS
🥪 m1 (2:)	Go	od			🥥 m1 (4:)	NTFS
🥪 m1 (3:)	No	t Bad				
🥟 m1 (4:)	Po	or				
@ Disk Scan at 10/06/12 12:48:45	6 [0,0] Inc	omplete 0		0		

Filter detected partitions by certainty

After you complete a scan, detected partitions are listed in order of their certainty status based on attributes and validation level. To make a long list of partitions easier to read, remove partitions with a status of Bad and lower using a filter. To filter detected partitions:

- 1. In the *Physical Device Scan View* on page 8 select a scan result node with detected partitions.
- 2. Open the Filter Detected Partition dialog:
 - From the toolbar, click Partition Filter.
 - Right-click the partition and click Partition Filter from the context menu.
- 3. Set filter values in General or Advanced tabs and click Filter to apply selected filter criteria.

Filter Detected Partitions	5	
General Advanced		
Filter by Partition File Sys	stem	
VITES FAT32	🔽 exFAT 🛛 🔽 FAT	HFS+ Ext2/Ext3/Ext4
Filter by Partition Status		
[8] Excellent	[5] Acceptable	[2] Bad
[7] Very Good	[4] Not Bad	[1] Very Bad
📝 [6] Good	[3] Poor	
Filter by Partition Size		
Enter size range, MB:		
Reset		Filter Cancel

General Options Filter by Partition File System

Select the file system that will remain in the filtered partition list.

Filter by Status

Select the partition integrity statuses that will remain in the filtered partition list.

Filter by Size

To restrict the size of a partition to display, click the Filter by Partition Size check box and enter the lowest and highest partition size in MB.

neral Advanced		
Filter by NTFS Specific At	tributes	
Primary Boot Sector	Primary [\$MFT]	Copy of [\$MFT]
Copy Boot Sector	Primary [\$MFT Mirror]	Copy of [\$MFT Mirror]
Root Folder [.]	V Primary [\$LogFile]	Copy of [\$LogFile]
📝 Bitmap [\$Bitmap]	V Primary [\$Volume]	Copy of [\$Volume]
Boot Record [\$Boot]	Quick Formatted	
Filter by FAT Specific Attr	ibutes	
Primary Boot Sector	Copy Boot Sector	Root Folder [.]
✓ Primary [FAT]	Copy of [FAT]	

Advanced tab filtering will let you filter a partition with specific NTFS or FAT attributes.

Press Reset in the Filter Detected Partition dialog to cancel partition filtering.

Work with scan results

It can take a long time to run a default disk scan or a low level disk scan. Because you are dealing with a large volume of information, you might not be able to review all the data in one session.

So that you do not have to scan a partition again, you can save and re-use valuable scan results. You can save an entire **Scan Results** branch or make a separate save for each disk scan or save all scans set for a particular device.

Scan results are saved with the file extension .scaninfo.



Warning: Save a scan results file to a physical drive that is different from the drive that contains the original files.

Save Scan Results

- **1.** To save the entire **Scan Results** branch, select the branch.
- 2. To save a device node, select it under Scan Results.
- **3.** Right-click the selected node and click **Save Scan Result** from the context menu. The **Save Scan Result** dialog appears with the default path and a suggested file name.
- **4.** To change the file path, browse to a different folder.
- **5.** To change the file name, enter a name in the file name field.
- 6. Click Save.

Load Scan Results

- 1. To open the Load Scan Results dialog, do one of the following:
 - From the File menu, click **Open** > **Scan Result...**
 - Right-click the logical drive node and click **Load Scan Result** from the context menu.
 - If there is a **Scan Results** branch in the **Recovery Explorer** tree, right-click the **Scan Results** branch or right-click a **Scan Results** node and click **Load Scan Result** from the context menu.
- **2.** Browse to the folder that contains the scan result file and select the file.
- 3. Click Open.

The data from the scan results file appears in a Scan Results node in the **Recovery Explorer** tree.



Note: Loading scan results feature is not available in Active@ UNDELETE Lite edition. Please visit *http://www.active-undelete.com* to read more about Active@ UNDELETE Professional and Enterprise editions

Remove Scan Results

Data in the **Scan Results** branch is copied from the original physical device. You may remove any node – including detected partitions - from the Scan Results branch without harming the data on the original physical device.

To remove scan results:

- 1. To remove the entire **Scan Results** branch, select the branch.
- 2. To remove a device node, select it under Scan Results.
- 3. Right-click the selected node and click **Remove Scan Result** from the context menu.

The selected node is removed from the **Recovery Explorer** tree.

File Preview

File Preview allows you to view the contents of an image file (jpg, bmp, gif, png etc.) before you recover the file.



To open the File Preview panel from any view, do one of the following:

• Double-click an image file.

T

- Right-click an image file and click **File Preview** from the context menu.
- Select an image file and click File Preview from the main toolbar.

Note: If the preview file is not an image file, it appears in hexadecimal and text mode.

Search for deleted Files and Folders

To help you find deleted files in a long list of files from a scanned drive, you may search the list with specific search criteria and review results in a *Search Results View* on page 10.

- 1. Select a scanned logical drive or scanned detected partition
- 2. To open the Search for Files and Folders dialog box, do one of the following:
 - From the main toolbar, click **Search**.
 - Right-click the selected item and click **Search** from the context menu.
- **3.** Inter search criteria in Look for and other search options (if required) and click Search button to start searching in selected location.

	Gearch in: Major	(D:);		
Lo	ook for:			
General	Date	Size	File Attributes *	
V Re Ma Sei	cursive search i Itch case arch among exis	in subdirect	ories	
Sei	arch among del	eted only		
Restore Def	aults			Search Cancel

Figure 7: General Search Options

General Search Options

Recursive search in subdirectories

Use this option to search the root level of the drive and all sub folders. To search only the root folder, clear this check box.

Match case

To display files that match upper and lower case letters in the Look for field, select the Match case check box.

Search among existing only

To display only files that are not deleted, select the Search among existing only check box.

Search among deleted only

To display only files that are deleted or damaged, select the Search among deleted only check box.

To display files by a specified date, in the **Date Criteria** tab, in the **Date Type** drop-down list, choose a type and select a date range.

General	Date Siz	e File Attributes	s *		
By Create	Date				
🔘 Today	② Last 7 day	s 🔘 Last 30 days	Oustom Range:	21/05/12 -	22/05/12 🔻
By Modified	d Date				
🔘 Today	Last 7 day	s 💿 Last 30 days	Custom Range:	21/05/12 -	22/05/12 🔻
By Accesse	ed (Deleted) Dat	2			
🔘 Today	Last 7 day	vs 🔵 Last 30 days	Custom Range	21/05/12 -	22/05/12 🔻

Figure 8: Date Criteria

To display files by a specified file size, in the **Size** tab, select Small, Medium or Large, or specify the size range in KB.

General	Date	Size	File Attributes	
Any				
🔘 Small	(less then 10	00KB)		
🔘 Mediu	um (less then	1MB)		
🔘 Large	e (More then	1MB)		
Speci	ify size range	, bytes fro	om:	to:

Figure 9: File Size Criteria

To display files based on file attributes in, the **File Attributes** tab select file attributes that should be present (Include Files and Folders Attributes) or otherwise exempt (Exclude Files or Folders with Attributes) in search result.

General	Date Siz	File Attribu	tes *				
☑ Include Files or Folders with Attributes							
Deleted	ReadOnly	System	Hidden	Directory	Temporary	MFT	
Sparse	V Stream	Compressed	Archive	Encrypted	Normal	Resident	
\$Journal	\$LogFile						
Exclude File	s or Folders with	n Attributes					
Deleted	ReadOnly	System	Hidden	Directory	Temporary	MFT	
Sparse	Stream	Compressed	Archive	Encrypted	Normal	Resident	
\$Journal	\$LogFile						
ID range:	-						

Figure 10: File Attributes Criteria

To change all settings back to defaults, click **Restore Defaults**.

4. Click **Search** to start searching process.

To display disk image events and progress details, click **Details**. To terminate the searching process, click **Stop** at any time. In this case search results may be not accurate or complete. After the search is done, a *Search Results view* appears.



Note: You may repeat a search many times and refine the search criteria for better results.



Note: See *How to Use Wildcards* on page 43 for details how to set search criteria. You may use *File Filter Toolbar* on page 33 to improve search results.

After search complete, *Search Results View* on page 10 must appear with search results (if any for provided criteria). You can repeat steps form 1 to 4 for desired effect.

File Filter Toolbar

The File Filter toolbar contains commands that can help you organize files in a list.

Filter by:	» 🗊 E	8 😭 🔒
------------	-------	---------

By default, the results of a scan contain all files and folders. Use commands in the File Filter toolbar to make a large list of files smaller and easier to read.

You may use the File Filter toolbar in the following views:

- Recovery Explorer View
- Document View
- Search Result Views

The filtered result may be applicable over an entire list (for example, in Search Result View) of within a selected folder (for example in Recovery Explorer view and Document View).

Using File Filter Toolbar

- To display an unfiltered list, click **Show All Files and Folders**.
- To display only existing files and folders, click **Show only existing Files and Folders**.

- To display only deleted files and folders, click **Show only deleted Files and Folders**.
- To further reduce the size of a list, enter a pattern in File Filter field and press **ENTER**. The list displays only those files that match the pattern.

Supported File Signatures

List of supported File Signatures by file types

 Compiled HTML Help [*.chm]

Custom (user defined) file signature templates

Active@ UNDELETE comes with more than fifty predefined (internally programmed, very fast) file signatures to be used to detect particular files (MS Office Documents, many Image formats, ZIP archives, MP3, etc..*See complete list of supported file signatures*) during disk scan. However sometimes advanced users need to detect more specific file formats, not being defined in default signatures set.

Active@ UNDELETE offers advanced tools to define user's templates for signatures to be analyzed. Signatures can be defined using extended definition language RegExp (Regular Expressions).

To create custom file signature

T.

- 1. Click Tools > Preferences command to open Preferences dialog.
- 2. Open File Signatures group and use Add button to define new custom file signature or
- **3.** Click **Import** button to load (import) custom file signature from script file (*.ini format). See *Custom Signatures Size Script* on page 38 for details.

Note: You can edit your custom file signature template at any time by selecting your template in list and clicking **Edit** button or simply double-clicking on template's name.

2	Supported file signatures	(templates):		
General Settings	Name	File Extension	Algorithm	🚽 Add
Environment	 Microsoft Office Documents Formatted Text files Compressed Archives Images and Camera Raw files Music and Videos QuickTime Multimedia files 			Edit
Recovery Explorer	Miscellaneous			
Device Backups				
File Signatures	✓III File signatures version: 1.06.06		4	X Remove

Custom file signature template

Provide template name and brief description - for future references. Specify file extension of a file type you defining (optional). To completely define custom file signature template you need to enter Header (beginning of the file) and Footer (end of the file) criteria using RegExp syntax. Header criteria could be more then one and all of them must be met to consider beginning of the file. Footer's criteria could more then one too, but at least one of them must be met to consider end of file.

User defined file ter	mplates	a - real desire, stati		? <mark>- </mark>
To define custon Footer (optiona create such scrip	n file signatures enter al) search criteria. Siz ot.	Template Name, Description and File e Script is useful to determine actua	extension (optional). Specify file's H al file size. Read manual for more det	leader (required) and ail and samples how to
Template <u>N</u> ame:	template 1]	
Template <u>d</u> escription:	MIDI Audio			
File extension:	mid2			
Beginning of File Criteria	End of File Criteria			
You may provide seve	ral beginning of file (h	neader) criteria - all of them of ther	n must be met.	
	Criteria	a (RegEx expression)	Not earlier then	Not later then
1 MThd			0	0
2				
Reset			Save Cance	el Help

Dialog Options

Template name

Unique template name.

Template description

Brief template description (optional)

File extension

File extension for this template (optional)

Beginning of File Criteria

List of RegEx criteria, considered as beginning of file combined as AND statements. Not earlier then and Not later then specifies limits of defined criteria in the beginning of a file template.

End of File Criteria

End of files can be determined in two ways:

• By list of RegEx criteria, considered as end of file combined as OR statements. In case of missing file footer criteria, end of file will be taken by defined Maximum file size. Append after attribute specifies size of end of a file. File
User defined file templates	? ×
To define custom file signatures enter Template Name, Description and File extension (optional). Specify file's He Footer (optional) search criteria. Size Script is useful to determine actual file size. Read manual for more deta create such script.	ader (required) and il and samples how to
Template Name: template1	
Template description: MIDI Audio	
File extension: mid2	
Beginning of File Criteria End of File Criteria	
Use RegExp conditions to detect end of file Use script to detect end of file	2
If end of file criteria (footer) is not set - maxi critera - at least one of them must be met. Provide RegExp expression as end of file condition	al end of files
Maximum file size, bytes: 6535	
Criteria (RegEx expression)	Append After
1 FO_END	5
2	
Reset Save Cancel	Help

size is used in case of missing file end criteria. By default its 65535 bytes.

 By using simple script to calculate end of file. See *Custom Signatures Size Script* on page 38 for reference.

	0	$\mathbf{r} = \mathbf{r} \cdot \mathbf{r}$
	Template Name:	template 1
Te	emplate description:	MIDI or use script to
	File extension:	mid2 calculate file size
Be	ginning of File Criteria	
	0	Use RegExp conditions to detect end of file Ise script to detect end of file
C	Create script to deter	mine file size using simple command and condition operators. See Help for script reference.
	next:	
	temp =	read(dword, size)
	if (tem	p == "MThd") goto valid
	if (tem	p != "MTrk") goto exit
	valid:	
	size =	sum(size, 4)
	temp =	read(dword, size)
	size =	sum(size, 4)
	temp =	endian(dword, temp)
	size =	<pre>sum(size, temp)</pre>
	goto ne	xt

Defined custom file signatures templates are stored in INI files in user's selected locations and will be loaded at every consequent application starts. You can also import such custom signature template files created by other users by clicking Import button and specifying full path to custom file signatures template file in opened dialog. See *Custom Signatures Size Script* on page 38 for details.



Note: Also you can specify Custom File signature template from Volume (logical Drive) Scan dialog or Disk Scan dialog by clicking **Add** button near file signatures list.



Important: Regular Expressions can be used while defining signature headers and footers. Please check *RegExp syntax on a web for examples*.

Custom Signatures Size Script

Custom (User Defined) File Signatures are saved in text file and can be edited by using simple text editor (like notepad) or by using Active@ UNDELETE tool: *Custom (user defined) file signature templates* on page 35.

User defined template reference

- Empty lines and lines starting with semicolon are ignored
- Sections order and lines order in sections are not important
- Letter case is not important (except RegExp fields)

Section TEMPLATES - required and contains fields numbering from one;

TEMPLATE### - points to the section where signature template is described (numbered from one).

Section Template Header - required and contains fields:

BEGIN

required. Points to the section describing begin of the signature file

FOOTER

non required. Points to the section describing end of the signature file

MAX_SIZE

non required. Maximum file size to force file-end, if no file-end signature is detected. By default it is 64Kb

GROUP

non required. If missed - template goes to User Defined templates group by default

DESCRIPTION

non required. This is a descriptive name of user template being displayed on a screen

EXTENSION

non required. This is a file extension to be assigned and displayed

SCRIPT

1 7

non required. Refers to the section where size of the file being calculated

Note: If field SCRIPT is present, then field FOOTER is ignored in template header section.

Beginning of the file section

Section describing file beginning (required), contains fields of the same type:

<signature> = <offset start> | <offset end>

signature

expression (regular or RegExp-compatible). Expression max length is 1024 bytes

offset_start

acceptable minimal signature offset from the beginning of the file

offset_end

acceptable maximum signature offset from the beginning of the file



Note: If there are several fields listed in signature beginning, logical AND operation applied to confirm file start.

End of file section

Section describing file end (not required), contains fields of the same type:

<signature> [= <bytes_to_append>]

signature

expression (regular or RegExp-compatible). Expression max length is 1024 bytes

bytes_to_append

not required. How many bytes to append to the file after the signature is found



Note: If there are several fields listed in signature, logical OR operation applied to define file end.

File size calculation script

Section calculating file size (not required), contains operators of four types:

```
<result> = <command> (<argument>, <argument>)
<result> = <argument>
IF (<argument> <condition> <argument>) GOTO <label>
GOTO <label>
```

commands

READ, ENDIAN, SUM, SUB, MUL, DIV, SHR, SHL, AND, OR and XOR

Most of commands are the same as in assembler programming language, except:

READ - first argument - data type (size) to be read, second - offset from the beginning of the file

ENDIAN - first argument - data type (size), second - expression, which byte order will be swapped

First argument for commands READ and ENDIAN must be one of reserved data types: BYTE, WORD, DWORD, QWORD

argument

can be either a named variable or a constant

result

can be the only named variable

condition

can be one of : < <= => >! = (meaning is the same as in C++)

label

T

consists of label name followed by colon and it can precede any operator

Note:

- Label named EXIT has been reserved and instructs to complete the calculations
- Named variable SIZE has been reserved and keeps the file size
- Constants can be in Decimal form, Binary (followed by 'b'), Octal ('o'), and Hexadecimal ('h') or can be a text string

```
[TEMPLATES]
TEMPLATE1 = PRIMITIVE_HTML
TEMPLATE2 = PRIMITIVE_JPG
TEMPLATE3 = QBW_HEADER
TEMPLATE4 = CHM_HEADER
```

```
TEMPLATE5 = SWF HEADER
TEMPLATE6 = PST HEADER
TEMPLATE7 = MRW HEADER
TEMPLATE8 = MID HEADER
TEMPLATE9 = CAB HEADER
TEMPLATE10 = BMP HEADER
TEMPLATE11 = DJV HEADER
[PRIMITIVE HTML]
DESCRIPTION = Primitive HTML Signature
EXTENSION = html
BEGIN=HTML BEGIN
FOOTER=HTML FOOTER
MAX SIZE = \overline{6}55360
[HTML BEGIN] <html = 0 | 512 <head = 0 | 1024
[HTML FOOTER] </html> = 2
[PRIMITIVE JPG]
BEGIN=BEGIN.TEST.JPG
GROUP = Images and Camera RAW files
DESCRIPTION = Primitive JPG files
FOOTER=FOOTER-.TEST.JPG
EXTENSION = test.jpg
MAX SIZE = 3221225472
[BEGIN.TEST.JPG]
xFF xD8 xFF = 0 | 0
[FOOTER-.TEST.JPG]
\xFF\xD9
[DJV HEADER]
DESCRIPTION=DjVu Document
EXTENSION=djvu
BEGIN=DJV BEGIN
SCRIPT=DJV_SCRIPT
[DJV BEGIN]
AT&TFORM=0|0
[DJV SCRIPT]
  size = read(dword, 8)
  size = endian(dword, size)
 size = sum(size, 12)
[QBW HEADER]
DESCRIPTION=QuickBooks Document
EXTENSION=qbw
BEGIN=QBW BEGIN
SCRIPT=QBW SCRIPT
[QBW BEGIN]
MAUI=96|96
[QBW SCRIPT]
  data = read(dword, 36)
  temp = read(dword, 52)
  if (temp <= data) goto exit
  size = sum(temp, 1)
  size = shl(size, 10)
[CHM HEADER]
DESCRIPTION=Microsoft CHM Help
```

```
EXTENSION=chm
BEGIN=CHM BEGIN
SCRIPT=CHM_SCRIPT
[CHM BEGIN]
ITSF=0|0
[CHM SCRIPT]
  version = read(dword, 4)
  if (version == 0) goto exit
  header = read(dword, 8)
  if (header <= 1Ch) goto exit
  temp = read(qword, header)
  if (temp != 1FEh) goto exit
  temp = sum(header, 8)
  size = read(qword, temp)
  temp = sum(header, 10h)
  if (size > temp) goto exit
  size = 0
[SWF HEADER]
DESCRIPTION=Adobe Flash SWF
EXTENSION=swf
BEGIN=SWF BEGIN
SCRIPT=SWF SCRIPT
[SWF BEGIN]
FWS = \overline{0} \mid 0
[SWF SCRIPT]
  data = read(byte, 3)
  if (data <= 10h) goto exit
  size = read(dword, 4)
  if (size <= 8) goto exit
  size = 0
[PST HEADER]
DESCRIPTION = Outlook Archive
EXTENSION = pst
BEGIN = PST BEGIN
SCRIPT = PST SCRIPT
[PST BEGIN]
!BDN=0|0
[PST SCRIPT]
  data = read(byte, 10)
  if (data == OEh) goto valid
  if (data != 17h) goto exit
  size = read(dword, 184)
  goto exit
valid:
  size = read(dword, 168)
[MRW HEADER]
DESCRIPTION = Minolta Camera Images
EXTENSION = mrw
BEGIN = MRW BEGIN
SCRIPT = MR\overline{W} SCRIPT
[MRW BEGIN]
\x00MRM=0|0
[MRW SCRIPT]
```

```
data = read(dword, 4)
  if (data == 0) goto exit
  width = read(word, 24)
  if (width == 0) goto exit
  width = endian(word, width)
  height = read(word, 26)
  if (height == 0) goto exit
  height = endian(word, height)
  pixel = read(byte, 32)
  if (pixel == 0) goto exit
  pixel = mul(pixel, width)
  pixel = mul(pixel, height)
  pixel = div(pixel, 8)
  size = endian(dword, data)
  size = sum(size, pixel)
  size = sum(size, 8)
[MID HEADER]
DESCRIPTION = MIDI Audio
EXTENSION = mid
BEGIN = MID BEGIN
SCRIPT=MID SCRIPT
[MID BEGIN]
MThd=0|0
[MID SCRIPT]
next:
  temp = read(dword, size)
  if (temp == "MThd") goto valid
  if (temp != "MTrk") goto exit
valid:
  size = sum(size, 4)
  temp = read(dword, size)
  size = sum(size, 4)
  temp = endian(dword, temp)
  size = sum(size, temp)
  goto next
[CAB HEADER]
DESCRIPTION=Microsoft Compressed Archive CAB
EXTENSION=cab
BEGIN=CAB BEGIN
SCRIPT=CAB SCRIPT
[CAB BEGIN]
MSCF=0|0
[CAB SCRIPT]
  version = read(word, 24)
  if (version != 103h) goto exit
  folders = read(word, 26)
  folders = mul(folders, 8)
  folders = sum(folders, 36)
  files = read(word, 28)
  files = mul(files, 16)
  files = sum(files, folders)
  temp = read(dword, 16)
  if (temp < folders) goto exit
  temp = read(dword, 8)
  if (temp <= files) goto exit
  flags = read(word, 30)
  flags = and(flags, 4)
  if (flags == 0) goto skip
  flags = read(dword, 36)
```

```
if (flags != 20) goto skip
  flags = read(dword, 44)
  if (flags < temp) goto skip
 size = flags
 temp = read(dword, 48)
skip:
  size = sum(temp, size)
[BMP HEADER]
DESCRIPTION = Bitmap Images BMP
EXTENSION = bmp
BEGIN=BMP BEGIN
SCRIPT=BMP SCRIPT
[BMP BEGIN]
BM=0\overline{|}0
[BMP SCRIPT]
  width = read(dword, 12h)
  if (width == 0) goto exit
  height = read(dword, 16h)
  if (height == 0) goto exit
  pixel = read(word, 1ch)
  if (pixel == 1) goto valid
  if (pixel == 4) goto valid
  if (pixel == 8) goto valid
  if (pixel == 16) goto valid
  if (pixel == 24) goto valid
  if (pixel != 32) goto exit
valid:
  pixel = mul(pixel, width)
  pixel = mul(pixel, height)
  pixel = div(pixel, 1000b)
  rastr size = read(dword, 22h)
  if (rastr size < pixel) goto exit
  rastr offset = read(dword, 0Ah)
  if (rastr offset < 38) goto exit
  rastr offset = sum(rastr offset, rastr size)
  size = read(dword, 2)
  if (size >= rastr offset) goto exit
  size = 0
```

How to Use Wildcards

A *wildcard* is a character that can be used as a substitute for any of a class of characters in a search. Wildcard characters are often used in place of one or more characters when you do not know what the real character is or you do not want to enter the entire name. In Active@ UNDELETE three types of wildcard are used: star or asterisk(*), question mark (?) and number sign (#).

Wildcard characters are used in the *File Filter Toolbar* on page 33 and *Search for deleted Files and Folders* on page 31.

Examples of using wildcards:

Wildcard character	Example	Description
Asterisk (*)	docum*	Use the asterisk as a substitute for zero or more characters if you are looking for a file that you know what it starts with and you cannot remember the rest

Wildcard character	Example	Description
		of the file name. The example locates all files of any file type that begin with "docum" including documents.txt, document_01.doc and documentum.doc.
	docum*.doc	To narrow the search to a specific type of file, include the file extension. The example locates all files that begin with "docum" and have the file name extension .doc, such as document_01.doc and documentum.doc.
Question mark (?)	doc?.doc	Use the question mark as a substitute for a single character in a file name. In the example, you will locate the file docs.doc or doc1.doc but not documents.doc.
Number sign (#)	doc_###.doc	Use the number sign (also known as the pound or hash sign) as a substitute for a single number in a name. In the example, you will locate the file doc_012.doc or doc_211.doc but not doc_ABS.doc.

Application Preferences

You can change many of the settings that affect the application's behavior in the Preferences dialog.

To open the **Preferences dialog**, do one of the following:

- From the **Tools** menu, select **Preferences**.
- In the Application Command bar select Support tab, click Preferences.

See description of each tabbed preferences page below

General options for working with Active @ UNDELETE
Application Start
Show Welcome Page
Check for available updates at application start
V Show Splash Screen
Autoload last saved Session
Wizards When you use the software the next time, configuration settings are loaded and Wizards will display the last saved parameters. To restore configuration to the original settings (right after software being installed), please dick the button below. Reset to Defaults

General Settings Options Show Welcome page

Show\Hide welcome dialog at application start.

Check for available updates at application start

Each time when Active@ UNDELETE starts it will request for available update\upgrade and prompt for download if newer version is available for download.

Show splash screen

Enable\Disable splash screen at application start.

Auto load last saved session

When this option is on, at application start Active@ UNDELETE will load latest saved session state, such as environment configuration, opened *Disk Images* and *Scan Results*. See *Using Scan Results* on page 26

Reset wizards to default

Restores original wizard settings and page sequence to default state.

Change how Active @ UNDELETE look and feels
Display
 Show Context Help Panel Show Command Bar Toolbars style: Large Icons, with text
Miscelanious Image: Wight of the second se
Preview Panel

Environment Options Context help panel

Show\Hide left side context help panel. Context help will automatically changed when active view (tab) is changed to show related hints and brief description of every view.

Show command bar

Show\Hide right side command bar that contains shortcuts to most usable commands and actions.

Use sound

Enable\Disable application sound notifications.

Hexadecimal file preview

When this option is on, file preview by default will be always shown in hexadecimal mode without any attempt to load it as an image or a document.

Recovery Explorer
Show System Files
Show Not Ready Devices
V Show Logical Drive Scan Options dialog before scan
If this option is not set, dialog will appear only if press and hold CTRL button.
Application Log
Show System (DEBUG) Events
☑ Save Log events to the Disk
Default log path: D:\Active_Undelete\applog.txt

Recovery Explorer Options Show system files

Show\Hide system files in Recovery explorer. In most of the cases these files are not recoverable.

Show no ready device

Show\Hide devices that has not read state and can not be scanned.

Show Logical Drive scan dialog by default

When this option is OFF, double click logical drive (volume) node in Recover Explorer view will initiate scan with default (most usable) options. Only when CTRL button is pressed down at the same time or this option is off, Scan Volume dialog will appear and let you to change scan options.

Show system events

Show\Hide system events in application log.

Save log file to disk

Enable\Disable saving log entries to the file. Use Default log path to specify log file.

Select Physic been changed files you may n	titioning Bac al Disks to auto . That includes: D restore Device Pa	kup -create P elete, Cr rtitioning	artition Backup File (* eate, Modify Attribute using Rollback Part	.bpi) each time w s or Format Parti ition Changes T	hen Partitions Layout h tion. Using these back u iool.	ias Jp
Name	Serial Name	Size	Last Modified	History Count		
\\.\PhysicalDrive0	ST31000524AS	932 GB		0		
\\.\PhysicalDrive1	ST3500630AS	466 GB	19-Dec-12 16:33:47	10		
\\.\PhysicalDrive2	OCZ-VERTEX3	55.9 GB		0		
\\.\PhysicalDrive3	ST31000524AS	932 GB		0		
Backup Location: ects_De	evelopment\Activ	e Undelet	te 8\!Binaries\backups	\disk_ST3100052	4AS_6VPFYGMJ.bkp	

Device Backups Options Backup location

Define individually Physical Device (disk) backup file location. See *Rollback Partition Changes* on page 66 for details.

Supported file signatures (templates):			
Name	File Extension	Algorithm	•	Add
 Formatted Text files Adobe Acrobat Documents Rich Text Format Files XML Files HTML Files Compressed Archives Images and Camera Raw files Music and Videos QuickTime Multimedia files Miscellaneous User Defined Signature Types 	pdf rtf xml htm	Adobe Acrobat Text Document Text Document Text Document	E	Edit

File Signatures Options File signatures list

Review available (supported) file signatures. User defined file signatures (if any) are shown in separate group.

Add file signature

Click **Add** button to add user define file signature. See *Custom (user defined) file signature templates* on page 35.

Edit file signature

Click **Edit** button when custom file signature is selected or double click custom file signature node to open edit dialog.

Import Custom file signatures

Click **Import** button to import custom file signatures define in third party configuration file.

Active@ UNDELETE Tools Overview

Main Active@ UNDELETE tools are:

- Partition Manager Overview on page 48
- Disk Editor Tool on page 54
- Disk Image Overview on page 66

Hardware Diagnostic File

If you want to contact our technical support staff for help with file recovery, a file that contains a summary of your local devices is helpful. Active@ UNDELETE allows you to create a summary listing file in XML format. This data format is "human-readable" and can help our technical support staff analyze your computer configuration or point out disk failures.

To create a hardware diagnostic file from the File menu, click Save Hardware Info As...



Note: To save time when contacting our technical support staff, we highly recommend that you provide us with a hardware diagnostic file

Partition Manager Overview

Partition Manager is advanced Active@ UNDELETE tool, that allows you to perform disk partitioning tasks, such as creating partitions and volumes, formatting them, and assigning drive letters. Initialize raw disk, edit partition tables and more.

Most of these changes to disk partitioning are recorded in dedicated backup files thus at any time these changes could be rollback at certain point. See *Rollback Partition Changes* on page 66 for more information.

Create New Partition	Create Virtua	al Partition Open in H	ex Editor 🛛 👩	Partition Table	Rollback Pa	rtition Changes	Kill Disk	:		
View 🗸										
Name	Туре	Status File Syste	em Segment	Offset in Sectors	First Sector	Total Sectors	Total Size		\\.\PhysicalDrive0 - Prop	oerties X
▲	Fixed Disk	Ready, Initialized	()	·	0	1953525168	932 GB		Name	Value 🔺
TREMMENS (F:)	Primary	FAT32	() 2048	2048	221206528	105 GB		Unsaved	No
🥔 Local Disk (M:)	Primary	Unknow	/n (221208576	221208576	333938688	159 GB		General	
🥪 Local Disk (O:)	Primary	Unknow	/n (555147264	555147264	218122240	104 GB		Name	\\.\PhysicalDrive0
Ø Unallocated Space		Unalloc	ated (773269504	773269504	1180255631	563 GB		Device Key	ST1000DM003-9YN162
a 🍘 Logical Disk Manager 0	Virtual LDM				0	488921491	233 GB		Platform Name	\\.\PhysicalDrive0
(E:)	Primary	FAT32	() 0	0	374018048	178 GB		Product Name	ST1000DM003-9YN162
🧽 New Volume (L:)	Primary	NTFS	(374018048	374018048	1949696	952 MB		Product Revision	CC4D
🥟 PORTO (Q:)	Primary	FAT32	(375967744	375967744	112953747	53.9 GB		Serial Number	SIDURZYW Beeder Initialiaed
⊿	Fixed Disk	Ready, Initialized			0	1953525168	932 GB		Status	Ready, Initialized
a 🍘 LDM data partition	Data LDM	LDM da	ta () 63	63	1953523057	932 GB		4 Device Size	Fixed Disk
🥟 TREMOW (E:)	Dynamic	FAT32	63	3 1985	2048	374018048	178 GB		- Device Size	
🦱 New Volume (L:)	Dynamic	NTFS	6	374020033	374020096	1949696	952 MB	1	▼ [<]	•
Physical Data Storage Devices										x
ST 1000DM003-9YF GPT (Basic) 932 GB	IENS (F:) Primary FAT32	Local Disk (M:) 159 GB Primary Unkn	own	Disk (0:) B Primary Unknown	563 GB U	inallocated				
Fixed Disk ST31000524AS MBR (Dynamic) 932 GB	W (E:) Dynamic FAT3:	New Volume (L:) 952 MB Dynamic N 9.77	GB Unallocat 53.	RTO (Q:) 9 GB Dynamic f 689	9 GB Unalloca	ted				1.00 MB Primary LDM me
@ Stud Did										
MBR (Basic) 932 GB	(D:) Primary NTFS									1.71 MB Unallocated
Fixed Disk OCZ-VERTEX3 MBR (Basic) 55.9 GB	n Reserved (1: Primary NTFS	Local Disk (C:) 55.8 GB Primary NTFS								1.90 MB Unallocated
Fixed Disk ST3500630AS MBR (Basic)	alotof	files (I 259 MB 1.98 GB	M2_106 1.95 GB1 6.84 (3B Unallocated	lont (W	(:) backu	ıp (]:)			

To open Partition Manager click **Tools** > **Partition Manager**in main application menu or use shortcut **Ctrl** +**M** at any time when running Active@ UNDELETE.

The main features of Partition Manager are:

- Initialize Disk (Physical Device) on page 49
- Create New Partition on page 50
- Change Partition Attributes on page 51
- Format Partition on page 51
- *Edit Boot Sectors* on page 52
- *Edit Partition Table* on page 53
- Create Virtual Partition on page 54

Initialize Disk (Physical Device)

To make disk accessible for application it needs to be initialized first by one of the following partition style:

- Master Boot Record (MBR);
- GUID Partition Table

Danger: Do not initialize disk if you are about to recover lost data from it! Use *Scan Disk (Physical Device)* on page 23 to retrieve your files fist.

To initialize physical disk proceed as follows:

- 1. In Partition Manager or Recovery Explorer select a Disk (Physical Device) node.
- 2. To open the Initialize Disk dialog, do one of the following:
 - From the Partition Manager toolbar click Initialize button or use command Actions > Initialize... from main menu;
 - Right-click the selected item and click **Initialize...** command from the context menu.
- 3. Confirm disk selection and other options

Disk must be ir	iitialized to let Active@ U	NDELETE use	it.	
Name	Product Name	Sectors	Total Size	
\\.\PhysicalDrive0	ST1000DM003-9YN162	1953525168	932 GB	
Select partition style for t	his disk: 🔘 MBR (Master	Boot Record)	© <u>G</u> рт (G	UID Partition Table)
Create typical Bootstr	ap code 📃 Clear Primar	y Partition tab	le	
MARNING: The GPT recommended fordisk	partition style is not recog s larger then 2TB, or disks	nized by all p used onItani	revious versi um-based co	ons of Windows. It is mputers.
				OK <u>C</u> ancel

Figure 11: Initialize Disk dialog

Dialog options Partition style

T.

Select either *MBR* (Master Boot Record) or *GPT* (GUID Partition Table) partition style.

Note: GPT partition style is not supported by older versions of Windows. It is recommended for disks larger then 2TB. For all other purposes we recommend to use MBR partition style

Create typical bootstrap code

Default generic bootstrap code will be written if this option is on.

Clear Primary partition table

Primary partition table records will be cleared.



Warning: It is highly recommended to not clear primary partition table in case of restoring deleted or damaged disk partitioning.

4. Click **OK** to complete disk initialization.

Create New Partition

To create new partition (Logical Drive):

- **1.** In **Partition Manager** select a Disk (*Physical Device*) or *Unallocated space* node.
- 2. To open the Create New Partition dialog:
 - From the toolbar click **Create New Partition** button or use command **Actions** > **Create New Partition...** from main menu.
 - Right-click the selected item and click **Create New Partition** command from the context menu.
- 3. Adjust dialog options and click **Create** button to create new partition.

932 GB Unallocated	
 Partition Geometry Primary Partition Extended Partition 	Partition Attributes Mark partition as Active Assign Drive Letter F
Maximum Partition size: 932 GB	Fromat New Partition Volume label: New Volume
Sector offset: 2048 Size, MB: 259508	File System: FAT32 ▼ Allocation unit size: Default ▼ ✓ Perform a quick format
ew Logical Drive will be created in Primary Partition, starting from 2048	sector with size 253 GB [531473758 sectors].

Figure 12: Create Partition dialog

Partition Geometry Primary or Extended

Partition can be created as Primary partition (of number of available Primary partitions are not exceeded) or as Extended partition.

Sector Offset

First sector of created partition. It can be set exact by numerical value entered in text box or by moving left slider in **Device View** control;

Partition Size

Partition size can be set in megabytes or in sectors, depending on state of **Measure in Sectors** check box;

Partition Geometry Mark Partition as Active

Newly created partition will be set as Active Partition;

Assign Drive letter

For Primary Partition or Logical Drive on extended partition drive letter can be assigned from the list of available in the system drive letters;

Format Partition [optional] Volume label

Text label of partition (disk). This field can be blank

File System

Select file one of the supported file systems: FAT, FAT 32 or NTFS.

Unit Allocation Size

Depending on selected file system and total partition (disk) size available allocated unit size may be different. Default value of unit size is recommended.

Change Partition Attributes

To change Logical Drive (partition) attributes:

- **1.** In **Partition Manager**, select a *Logical Drive* (*Partition*) node.
- 2. To open the Partition Attributes dialog, do one of the following:

Chenge drive letter and volume label for selected drive.
Assign the following drive letter: F
Volume Label DATA
OK Cancel

Figure 13: Create Partition dialog

- From the Partition Manager toolbar click Change Attributes button or use command Actions > Change Attributes from main menu;
- Right-click the selected item and click **Change Attributes** from the context menu.

Format Partition

To format *Logical Drive* (*Partition*):

- **1.** In **Partition Manager** select a *Logical Drive* (Partition) node.
- 2. To open the Format Partition dialog:
 - From the toolbar click **Format** button or use command **Actions** > **Format...** from main menu.
 - Right-click the selected item and click **Format...** command from the context menu.
- **3.** Adjust dialog options and click **Format** button to format partition.

Format selected volume with selected File System and Allocation size unit. Volume label is optional.
Volume label: Iont
File System: FAT32 Allocation unit size: Default
Perform a quick format
Format Cancel

Figure 14: Format Partition dialog

Dialog Options

Volume label

Text label of partition (disk). This field can be blank

File System

Select file one of the supported file systems: FAT, FAT 32 or NTFS.

Unit Allocation Size

Depending on selected file system and total partition (disk) size available allocated unit size may be different. **Default** value of unit size is recommended.

Edit Boot Sectors

Primary Boot Sector and *Copy Boot Sector* (if applicable) can be edited and synchronized by individual fields. Active@ UNDELETE provide "suggested" boot sector with most appropriate values for reference.

To Edit (Synchronize) boot sectors:

- **1.** In **Partition Manager** or **Recovery Explorer** select a *Logical Drive* (*Partition*) node.
- 2. To open the Edit Boot Sectors dialog, do one of the following:
 - From the toolbar click Edit Boot Records button or use command Actions > Edit Boot Records... from main menu;
 - Right-click the selected item and click Edit Boot Records... command from the context menu.

3. Use radio buttons near the value fields to select and click **OK** button to confirm changes.

			Δ	Primary Boot Sector [Offset, bytes: 0]	Δ	Copy Boot Sector [Offset, bytes: 536707072]		Suggested Boot Sec	tor
00	JMP instruction	0	Δ	EB5B90	◎ 🛆	000000	۲	EB5290	 [
03	File System ID	0	\checkmark	4E54465320202020	<u> </u>	000000000000000000000000000000000000000		4E54465320202020	
0B	Bytes per sector	۲	\checkmark	0200		0000		0200	
DD	Sectors per cluster	۲	\checkmark	01		00		01	_
0E	Always 00		\checkmark	0000 Invalid field indic	ator	0000		0000	_
10	Always 00		\checkmark	000000	 ✓ 	000000		000000	
13	Always 00		\checkmark	0000	- 🗸	0000		0000	
15	Media descriptor	۲	\checkmark	F8		00	Δ	cented Value	_
16	Always 00	Г	Vali	d field	- 🗸	0000			
18	Sectors per track	L	van			0000		03F	
1A	Heads	\bigcirc		0020	0 🛆	0000	۲	00FF	
1C	Hidden Sectors	۲	\checkmark	000003F		0000000		000003F	
20	Always 00		\checkmark	0000000	- 🗸	0000000		00000000	
24	Always 00 80 00 80	۲	\checkmark	80008000		0000000		80008000	
				📝 Show offset in he	xadecimal	mode 📝 Show values in he	xadeci	mal mode 📃 Save on I	Dis

Figure 15: Synchronize Boot sectors dialog box

Edit Partition Table

You can edit *Disk System Records* (MFT, Boot sector etc.) by using specially designed forms.

To edit *Partition Table*:

- **1.** In **Recovery Explorer** or in **Partition Manager** select a *Physical Device*.
- 2. To open the Edit Partition Table dialog:
 - Use command Actions > Partition Table... from main menu;
 - Right-click the selected item and click **Partition Table** command from the context menu.
- 3. Change desired fields to appropriate values

View and edit master t	poot record				
00 Master bootstrap [first	t 32]: EB06000000000033C0FA8ED0BC007	CFB8E	D88BF48EC0BF267E06	57BF007EB9	
1B8 Disk Index: BF0	418E6				
1BC Reserved: 000	0				
1FE Signature (55AA): 55A	A				
Partition Table Entry #1		Partit	tion Table Entry #2		
1BE Active Partition (80):	00	1CE	Active Partition (80):	00	
1BF Start Head:	180	1CF	Start Head:	254	
1C0 Start Sector:	1	1D0	Start Sector:	63	
1C0 Start Cylinder:	877	1D0	Start Cylinder:	1023	=
1C2 File System [hex]:	07	1D2	File System [hex]:	OF	
1C3 End Head:	254	1D3	End Head:	254	
1C4 End Sector:	63	1D4	End Sector:	63	
1C4 End Cylinder:	1023	1D4	End Cylinder:	1023	
1C6 First Sector:	14100345	1D6	First Sector:	141002505	
1CA Partition size in sectors:	126902160	1DA P	artition size in sectors:	23117535	
Partition Table Entry #3		Partit	tion Table Entry #4		
1DE Active Partition (80):	80	1EE	Active Partition (80):	00	-
Reset				Show offset in hexadecimal mo	ode

Figure 16: Edit Partition Table dialog

- To discard all changes and restore all values to fields in the dialog, click **Reset**.
- To save all changes made in the dialog, click **Save**.

<mark>س</mark> ک

Warning:

Saving incorrect values might render the partition useless. You may not undo changes that you make in this dialog.

Create Virtual Partition

To create virtual partition in Active@ UNDELETE proceed as follows:

- 1. In Partition Manager or Recovery Explorer select a Disk (Physical Device) node.
- 2. To open the Create Virtual Partition dialog, do one of the following:
 - From the **Partition Manager** toolbar click **Create Virtual Partition** button or use command **Actions** > **Create Virtual Partition** from main menu;
 - Right-click the selected item and click **Create Virtual Partition** command from the context menu.

Disk Editor Tool

Overview

Disk Editor is advanced tool for viewing and editing sectors of *Physical Disks* or *Partitions and Volumes* and contents of any file type. Disk Editor uses a simple, low-level disk viewer which displays information in binary and text modes at the same time. You can use this view to analyze the contents of data storage structure elements such as:

- Hard disk drives (disks)
- Partitions
- Volumes (Logical drives)
- Files

To open any of these items in the editor:

- 1. In the **Recovery Explorer** tree pane or file pane select an item.
- **2.** Do one of the following:
 - From the Edit menu, click Open In Disk Editor.
 - Right-click the item and click **Open In Disk Editor** from the context menu.

Disk Editor shows detailed information about the selected object in the information panel on the left side of the view. The right panel displays the binary and text view of the file. After the Disk Editor view appears, you may browse through the content of the open item using the scroll bar, keyboard arrows or the mouse wheel.

Click either the binary area or the text area to focus on it. You may also use the Tab keyboard key to switch the focus between hexadecimal and text modes.



Warning: As with any advanced tool, use extreme caution with the **Disk Editor**. Changes that you make may affect disk structure integrity. You must be certain that the changes you make are in line with correct data structures before you save changes.

Disk Editor Preferences

Disk Editor memorize its state and when closed those settings are preserved. The settings saved are view options and geometry of windows.

Saving Changes

Unless stated otherwise, all modifications made in the Disk Editor are stored in memory. Changes are written to the drive when you click **Save**.

Open objects in Disk Editor

You can open a physical disk, a logical drive, or a partition from the *Recovery Explorer View* on page 7. If you performed scanning, you can also open a file from the list of found files.

In Disk editor you can view and edit following disk objects:

- Physical Disk
- Volume (Logical Drive)
- Partition
- File

To open an object:

- **1.** Select an object in a list of disk objects. You may select a physical drive, a partition, or a logical drive. If you performed scanning before, you can also select a file.
- 2. Click the Open in Disk Editor Button in a toolbar.
- **3.** Alternatively, right click on a disk object and select **Open in Disk Editor** from a context menu. You can also use the **Ctrl+H** shortcut.

You will see physical drives and partitions only if you switched to the *Expert Device* mode of **Recovery Explorer** view. Otherwise, only logical drives will be available.



Warning: As with any advanced tool, use extreme caution with the **Disk Editor**. Changes that you make may affect disk structure integrity. You must be certain that the changes you make are in line with correct data structures before you save changes.

Subject Navigation and Information

After you have opened an object with the **Disk Editor**, you may navigate by scrolling block by block, or by jumping directly to specific addresses. You may jump to disk system records such as the boot sector (primary and copy) or a partition table.

Use the **Navigate** button in the toolbar to jump to a specific area in the open object.

The selections that appear depend on the type of object that you are editing.

No matter what object is opened for editing, the first two menu items in the **Navigate** menu will be **Go to Offset** and **Go to Sector**.

Go to Offset

The **Go to Offset** menu opens a dialog allowing specification of an exact location (offset) in the disk to jump to.

You can use both decimal and hexadecimal values, preceding hexadecimal values with 0x. For example, to specify location 512 as a hexadecimal number, enter 0x200. There are also options to specify an offset from the beginning, from the current position, or from the end.

Next to the offset edit field there are two labels specifying the minimum and maximum allowed vales for offsets displayed as decimal numbers.

You can also open this dialog directly by using the shortcut **Ctrl+Shift+G**.

Go to Sector

This command allows jumping to the beginning of a specified sector or cluster.

There are two edit fields in this dialog that allow entering a desired location either as a sector number or a cluster number.

The **Cluster edit field** is available only for logical disks and greyed out for all other objects.

As with the offset dialog, you can also use both decimal and hexadecimal numbers.

Next to the edit field is the range of allowed values in brackets. Notice that not all sectors correspond to clusters, but every cluster corresponds to a particular sector.

You can enter either a sector value or a cluster value. Depending on which field is active, the dialog will use a sector or cluster. If you enter a number in the cluster edit field, a corresponding sector is displayed automatically.

You can also open this dialog directly using the shortcut **Ctrl+G**.

Back and Forward navigation

When you navigate to an access point through the **Navigate** menu or jump to a specific offset or sector, those addresses are stored in a stack. You can move backward and forward to the previous locations by using the **Back** and **Forward** commands located in the **Disk Editor Toolbar**.

Navigate a Physical Disk

To navigate to the disk system records of a physical disk, click on the **Navigate** button in the toolbar. Depending on the partition scheme and contents of the physical disk you are editing, the **Navigate** menu will contain different options.

Navigating basic disks

After the **Go to Offset** and **Go to Sector** items there is a **Partition Table** menu item which allows jumping to sector 0 of a physical disk. As you jump to the partition table, a *Master Boot Record* template is automatically selected.

If the disk is not empty, the names of the partitions and their system areas will be in sub menus below the **Partition Table** menu item.

Navigating dynamic disks

For dynamic disks the following system areas are available for direct access:

- LDM Private Header
- LDM Primary TOC Block
- LDM Backup TOC Block
- LDM VMDB Block
- LDM KLog
- LDM First VBLK Block

After each access point a sector number is specified in the brackets.

Navigate a Logical Drive

To navigate to the disk system records of a logical drive, click on the Navigate button in the toolbar.

Depending on the file system present in a logical drive, the navigation menu will have different access points.

FAT and FAT32 drives

- Boot Sector
- Boot Sector Copy (FAT32 only)
- FAT1
- FAT2
- Root Directory

NTFS drives

- Boot Sector
- Boot Sector Copy
- \$MFT
- \$MFT Mirror
- Arbitrary MFT record

HFS+ drives

- Volume Header
- Volume Header Copy

Ext2/Ext3 drives

Superblock

Some of the access points when used automatically select a corresponding template. For example, if a boot sector access point is selected, a boot sector template is applied to the boot sector offset.

Data Inspector

The **Data Inspector** is a small tool view window that provides the service of "inspecting" (or interpreting) data currently selected in the edit pane. The Data Inspector lets you view the type of data you have selected. This may help you interpret data as displayed in **Disk Editor**.

Data Inspector		x
Name	Value	
8 bit, binary	00100110	
ANSI character	&	
Unicode character		
8 bit, signed	38	
8 bit, unsigned	38	
16 bit, signed	806	
16 bit, unsigned	806	
32 bit, signed	806	
32 bit, unsigned	806	
64 bit, signed	281,474,976,711,462	
64 bit, unsigned	281,474,976,711,462	
DOS time		
Windows time	1601-11-22 18:44:57	
UNIX time	1970-01-01 00:13:26	

To open the Data Inspector, from the Disk Editor toolbar, choose View > Data Inspector;

To copy an interpreted data from Data Inspector as a text:

- 1. Right-click anywhere in the Data Inspector window.
- 2. Select Copy.

To switch between *little endian* and *big endian* representation:

- 1. Right-click anywhere in the Data Inspector window.
- 2. Select Big Endian

The Data Inspector window is dockable and its location can be changed by clicking on the window title and dragging it to the new location. If the Data Inspector window is sharing its space with other tool views, you can change its relative position by left clicking and dragging the window tab. You can close the window by clicking on the [X] button in the top right corner of the window and reopen it again using the **View** menu in the **Disk Editor Toolbar**.

Using Bookmarks

Bookmarks allow you to save the current cursor location and quickly return to it later on. You may also give a name to a bookmark to make orientation easier.

Bookmarks are shown in the tool window called Bookmarks. If the Bookmarks window is closed you can open it using the menu **View** > **Bookmarks**.

Bookmarks	x
🗆 👂 🗣 🔂 🗫	
Bookmark	Offset
🔽 🗖 Block A	2,683,645,746
🔽 🔲 MFT Mirror	8,282,853,376

Placing and removing a bookmark

Press **Ctrl+F2** in order add a bookmark. Alternatively, you can right click in the hex editor and select a command from a context menu. The bookmark position is shown with a light blue box and also added to the list of bookmarks in the **Bookmarks** window. To remove a bookmark, press **Ctrl+F2** while having the cursor over the position of that bookmark. You can also remove a bookmark from the **Bookmarks** window by selecting a bookmark button in a toolbar and clicking delete. The delete function may also be selected from a context menu.

Going to a bookmark

If you have defined bookmarks, pressing **F2** will move your current position to the next enabled bookmark in the list.

You can also right click a bookmark and select the **Next** bookmark command from a context menu. Another option is to double click a bookmark name in the **Bookmarks** window.

Editing bookmarks

Bookmarks are named automatically when they are placed. You can rename a bookmark in the Bookmarks window to give it some meaningful name. To do so make a single mouse click on the bookmark name and edit it. Press **Enter** to accept your changes or **Esc** to cancel editing and revert to the original name. You can also rename a bookmark by right clicking on it and selecting the **Rename** command from a context menu.

Sometimes instead of deleting a bookmark it is useful to temporarily disable it. A disabled bookmark will not be counted when moving to the next bookmark. Uncheck a bookmark in the Bookmarks window to disable it. To disable all bookmarks at once click **Disable** all bookmarks in a toolbar or select this command in a context menu.

Searching in Disk Editor

To search text or byte sequence in **Disk Editor**:

- Click Ctrl+F shortcut key or
- Use Find button in Disk Editor's toolbar then Find dialog will appear.

Find what			
Use:	Regular expressions	•	
Hex:			
Unicode:			
Find options			
Search direction:	Search down 🔻		
Search at offset	0	of 512	bytes block
Blocks start is determ During subsequent se	ined by current cursor po earch by F3/Ctrl+F3 the la	sition. Use 0x prefix for h ayout of defined blocks is	exadecimal values, preserved,
		Find Fin	

Search data by ANSI, Hex or Unicode pattern. To speed up the process you can ask to search only at given offset inside used-defined blocks. Regular expressions and wildcard are even greater expand search capabilities.

Search direction will specify search direction from current cursor position.

When using **Find All** command, list of all search entries will appear. Use this list to navigate between search result entries (if any) by double clicking on entry line.

Examples of using regular expressions:

^\d\d?\$ - match integers 0 to 99

^\S+\$ - match strings without white space

\b(mail|letter|correspondence)\b - match strings containing 'mail' or 'letter' or 'correspondence' but only match whole words i.e. not 'email'

&(?!amp;) - match ampersands but not &

\b(Eric | Eirik) \b - match Eric or Eirik

*.html - using a wildcard

Editing with Disk Editor

The **Disk Editor** allows you to edit the content of a selected part of an opened object. By default, the Disk Editor shows the content of an object in a *Read Only* mode that prevents accidental modifications. In *Edit* mode, you can change the content of the opened file or disk and all modifications are stored in memory. Changes are written to the drive when you click **Save**.

To toggle between *Read Only* and *Edit* modes, do one of the following:

- From the Disk Editor toolbar, choose **Edit** > **Allow Edit** content.
- Right-click in the edit pane choose Allow Edit content from the context menu.

When you copy selected text from the edit pane to the clipboard, you may store it there in one of two formats using the following commands:

- **Copy** selected data is copied into the clipboard as binary.
- Copy Formatted selected data is copied as formatted text suitable to paste into a text editor.

Working with selection

In order to select data in the **Disk Editor Area**, click and hold down the left mouse button and start dragging to select an area. The selected area background will be highlighted. Release the mouse to finish

selecting. You can select an area bigger than will fit into the screen by dragging the mouse beyond the top or bottom edge of the hex editor window.

The alternative way to make a selection is to define a beginning and an end of the block. This method might be more convenient when a large area has to be selected in order to simply select data in a particular range. Move the cursor to the position where you want the selection to start and do one of the following:

- Select the menu command Edit > Beginning of block from the Edit menu in the toolbar.
- Right click and select **Edit** > **Beginning of block** from a context menu.
- Press Ctrl+1.

Move the cursor to the end of the desired selection and set the end of a selection in a similar way. If you need to select all the data, you can use the Select All command instead.

Copying to the clipboard

Select an area of data as described above and either select the command **Edit** > **Copy** or press **Ctrl+C**. The selected area will be copied into the clipboard in binary format. If you later want to insert it into a text editor, use the **Copy Formatted** command instead. It will copy data as a formatted text.

Please note that you can copy a maximum of 1MB of data into the clipboard.

Pasting data from the clipboard

If you copied data into the clipboard, you can paste it into a different place by moving the cursor to the position where you want the data to be copied. Use the command **Edit** > **Paste** or press **Ctrl+V**.

If you copied a text into the clipboard in a text editor, it will be pasted into the **Disk Editor** as text. Otherwise, the data will be copied as binaries.

45	44	20) 44	45	4C	45	54	45	44	20	44	45	4C	45	54	ED	DELETED	DELET	
45	44	20	44	45	4C	45	54	45	44	20	44	45	4C	45	54	ED	DELETED	DELET	
45	44	29	11	15	10	45	54	15	11	20	11	15	10	15	54	ED.	PELETED	DELET	
45	44	2(F	ill Blo	ck										R	^	ELETED	DELET	
45	44	2(C 11									ELETED	DELET	
45	44	2(Ove	erwrite	selec	ted a	ata in ti	ne tollov	ving ra	ange:							ELETED	DELET	
45	44	2(1,2	99 - 1,	,609,	total	311 byt	es									ELETED	DELET	<mark>.</mark>
45	44	2(_	ELETED	DELET	
45	44	2(Tex	t:	D	ELETE	ED										ELETED	DELET	
45	44	2(0	r						ELETED	DELET	
45	44	2(Hex	value	es: E	nter p	attern	hex valı	les								ELETED	DELET	
45	44	2(ELETED	DELET	
45	44	2(OK			Cancel		ELETED	DELET	
45	44	2				_			_		_		_				ELETED	DELET	
45	44	20	44	45	4C	45	54	45	44	20	44	45	4C	45	54	ED	DELETED	DELET	
45	44	20	44	45	4C	45	54	45	44	20	44	45	4C	45	54	ED	DELETED	DELET	
45	44	20	44	45	4C	45	54	45	44	20	44	45	4C	45	54	ED	DELETED	DELET	
45	44	20	44	45	4C	45	54	45	44	20	44	45	4C	45	54	ED	DELETED	DELET	

Filling a selection

You can fill a selection with an arbitrary text or binary data. Make a selection first, then right click **Edit** > **Fill** block. The **Fill Block** dialog allows entering either text or hex value patterns which will be used to fill the selection. Patterns are used in a loop until the whole selection is filled. For example, if you need to fill a selection with 0 bytes, just enter 00 into the Hex values edit field. If you want fill it with an '**ERASED**' pattern, enter it as a text and it will be repeated as many times as necessary to fill the block.

Use Template Editing

You can edit system records (like boot sectors, MBR, MFT etc.) by using a template tool window. Template window is a small dockable window normally located to the left from main Disk Editor editing area. If it is not visible, you can turn it on by selecting toolbar menu **View** > **Templates**.

Templates			x	Offset	0 1	EF	ASCII	I
NTFS Boot Sector 🔹 🏠	F	0:000	0:000	000000000	EB 52	00 00	лRhNTFS	Τ
				000000010	00 00	3 09	ш?.я.иЕИ.	
Name	Offset	Value	Copy Value	000000020	00 00	00 00	Ъ.Ъ.]Ін	
JMP instruction	000	EB 52 90	EB 52 90	000000030	00 00	00 00	- %N	
OEM ID	003	NTFS	NTFS	000000040	RC 00	0 77	Thto:to-	
BIOS Parameter Block	00B			00000040	F-6 00	STA	цЕбд9;д92	1
Bytes per sector	00B	512	512	000000050	00 00	0 07	ъЗАЋРј. ыёА.	
Sectors per cluster	00D	8	8	000000060	8E D8 1)E 00	ЋШиёЋАЗЫЖ	
Reserved sectors	00E	0	0	000000070	10 E8	00 B4	.иS.hhj.Лљ.\$.r	
(always zero)	010	00 00 00	00 00 00	000000080	08 CD 1	0.66	.н.з.№яяљсf.¶ж@f	
(unused)	013	00 00	00 00	000000000	OF BG	6 OF	CERSURTHAN AF	
Media descriptor	015	248	248	0000000000	OF BO	00 01	. ICDB: 4B HAR.AI.	
(unused)	016	00 00	00 00	0000000A0	B/ C9	5 8A	·NIYDIJ .I'FA%EUJb	
Sectors per track	018	63	63	0000000В0	16 24	01 01	.\$.H.r.ЃыUЄu.цБ.	
Number of heads	01A	255	255	0000000000	74 04	00 66	t.юГ <mark>f</mark> `fўf	1
Hidden sectors	01C	164,120,040	164,120,040	000000000	03 06	66 6 A	f;,:fi	
(unused)	020	00 00 00 00	00 00 00 00	0000000000	00 66	00.00	fP Sfh B>	-
Signature	024	80 00 80 00	80 00 80 00	000000000000	00 00			
Total sectors	028	32,354,909	32,354,909	00000010	0F 85	ST UU	иіяь>"а.	
\$MFT cluster number	030	786,432	786,432	000000100	B4 42 8	5B 07	ґВЉ.\$<фН.fX[.	
\$MFTMirr cluster number	038	2,022,181	2,022,181	000000110	66 58	18 00	fXfX.л-f3Tf. ·	
Clusters per File Record Se	040	246	246	000000120	66 F7	37 36	fчсюВЉКf< РfБк.ч6	
Clusters per Index Block	044	1	1	000000130	1A 00 8	C 88	+ILЪ.\$ЪиАлМё	
Volume serial number	048	45 62 90 39	45 62 90 39 A6 90	000000140	01 02	0 66		
Checksum	050	0	0	000000140	01 02	00 00	п,њА А.	1
Bootstrap code	054	FA 33 C0 8E	FA 33 C0 8E D0 B	000000150	FF 06	6 61	яяояta	-
Signature (55 AA)	1FE	55 AA	55 AA	000000160	C3 A0 I	B FE	Гш.иы.иылю	
				000000170	B4 01	D 10	r. <p-,<.t.r.≫h.< td=""><td></td></p-,<.t.r.≫h.<>	
				000000180	EB F2	S1 64	лтГA disk read	
				000000100	20 65	1 00		

Applying a template

In order to apply a template to the desired offset, move the cursor to the location and use **Edit** menu command **Set Template position**. You can select this command either from Edit toolbar menu or from a context menu. The next step select a required template from the list box with template names in the toolbar of templates window.

NTFS Boot Sector 🔹 👔 👢	8	0:000	0:000
Select a template Master Boot Record	Offset	Value	Copy Value
NTFS Boot Sector	000	EB 52 90	EB 52 90
FAT Boot Sector	003	NTFS	NTFS
exFAT Boot Sector	00B		
HFS+ Volume Header	00B	512	512
Ext2/Ext3 Superblock	00D	8	8
FAT Directory Entry	00E	0	0
(always zero)	010	00 00 00	00 00 00
(unused)	013	00 00	00 00
Media descriptor	015	248	248
(unused)	016	00 00	00 00
Sectors per track	018	63	63
Number of heads	01A	255	255
Hidden sectors	01C	164,120,040	164,120,040
(unused)	020	00 00 00 00	00 00 00 00
Signature	024	80 00 80 00	80 00 80 00
Total sectors	028	32,354,909	32,354,909
\$MFT cluster number	030	786,432	786,432
\$MFTMirr cluster number	038	2,022,181	2,022,181
Clusters per File Record Se	040	246	246
Clusters per Index Block	044	1	1
Volume serial number	048	45 62 90 39	45 62 90 39 A6 90
Checksum	050	0	0
Bootstrap code	054	FA 33 C0 8E	FA 33 C0 8E D0 B
Signature (55 AA)	1FE	55 AA	55 AA

When you are jumping to particular system areas using **Navigate** menu, the corresponding template might be applied automatically. This is true for templates like boot sectors, MBR or MFT record but not all access points have a template associated with them.

The following templates are supported:

- MBR
- GUID Partition table
- NTFS boot sector
- NTFS MFT file record
- FAT boot sector
- FAT32 boot sector
- FAT directory entry
- exFAT boot sector
- exFAT directory entry
- HFS+ Volume header
- HFS+ Catalog Node
- HFS+ File Record
- Ext2/Ext3/Ext4 superblock
- Ext2/Ext3/Ext4 inode
- UFS superblock
- UFS inode
- LDM structures

As you edit data in Hex, ASCII or Unicode pane or in Templates window, modified data is fully synchronized between views. After each modification a template view is recalculated giving you an up-to-date interpretation of data.

Template Copy

The following templates have their copy:

- NTFS Boot Sector
- FAT32 Boot Sector
- HFS+ Volume Header
- Ext2/Ext3 super block
- LDM Private Header
- LDM TOC Block

In this case template window will have an additional column named Copy Value which contains the data from the copy record. Template copies are useful to compare record located in different locations using the same pattern, for example to compare a boot record with its copy.

In case of Copy template its location is set separately from a main record using the same pattern. If the main template and its copy are intersecting, the copy template data will be shown in template window but not highlighted in the main edit area.

Setting template position

In order to set a template position or change an existing one move the cursor to desired location and use Edit menu command Set Template position (or Set Template Copy Position for its copy).

Navigating to a system area which has an attached template using Navigate menu also changes template position.

In order to facilitate the movement between records located in sequence, use arrow buttons located in the template window toolbar next to the templates list. For example, if you are editing or viewing an MFT record you can easily move to the next or previous record using those buttons.

Another way to set a template position is to enter new offset directly into template offset edit field in the template window toolbar. One of those fields are used for entering an offset of the main record and another is for its copy. The format of offset used in offset field is <sector:>:<sector offset>. You don't need to specify sector offset if you want to move to the beginning of the sector. For example, you can simply enter 100 to go to sector 100 and template offset will be shown as 100:0, but if you need to specify 128 byte in sector 100, you have to enter 100:128.

Highlighting template fields

By default all individual fields of template record are highlighted in Disk Editor main area (in hexadecimal and ASCII columns only). This coloring highlighting can be disabled by clicking Toggle template fields coloring button in template window toolbar next to arrow buttons.

The colors used by template coloring are arbitrary and have no specific meaning, their main purpose is to make separate fields visible and distinguish from each other. Actually, a palette of several colors is chosen and colors are used in a circle. When you select a field in the template window, the current field is also highlighted in hex editing area with bold field frame.

When you move a mouse cursor above colored field in editing area, the name and value of the corresponding field is also shown in a tooltip.

Navigating around template fields

You can set the cursor (current position) to a particular field in a template by double clicking it. If you double click in Name, Offset or Value column, the position inside the main record is selected, but if you click inside Copy Value column, the navigation is performed to the field in template copy.

Please note, that in Edit mode double clicking inside of Value or Copy Value starts editing of the field instead of navigating to that field.

Editing using template

Double click in the **Value** or **Copy Value** column to start editing the field (make sure that **Allow Edit Content** is enabled).

Some of the fields are edited according to the mask and will not allow to enter invalid values. For example, you cannot enter the number bigger than 65535 when editing a 2-byte field or invalid date when editing a date.

To exit the editing of the field with saving the result of edit, press Enter or click to another field. To exit editing without saving the result and revert to original value, press Esc.

Some of the templates fields depend on other fields. When a template is selected, an initial parsing occurs. If some of the fields contain invalid values, the further parsing of the record might be not possible and parsing will be stopped at this point, resulting in incomplete record. As an example lets take an MFT record. The record header is always parsed, but if it contains invalid fields or update sequence, attributes will not be parsed. The same is true when parsing an attribute - if an error occurs, the further parse is canceled and no subsequent attributes are added to the record.

Furthermore, the whole set of fields for the template might depend on some field values. For example, FAT Directory Entry template will show a Short File Name Entry fields or Long File Name depending on the value of the flags.

Hyperlinks in templates

Many templates contain hyperlinks allowing navigate easily to important data points.

For example, MFT records contain links to first cluster in data runs and MBR provides links to partitions.

Active@ Disk Editor 3								
File View Window Help								
Save Sack S	orward	Edit 🛒 🔍	Find 🏹	Ravigat	te	>	Go to	
Templates		ð ×	0	Offset	0	1	2	
NTFS MFT File Record 🔻 👔 👢		6:000 0:000	0000	00000000	EВ	52	90	
Name	Offect	Value	0000	00000016	00	00	00	
Padding	202		0000	00000032	00	00	00	
Allocated size	292	822 607 872	0000	00000048	00	00	0C	
Real size	304	822,607,872	0000	00000064	F6	00	00	
Initialized size	312	822,607,872	0000	08000000	00	00	00	
⊿ \$DATA	320		0000	00000096	1F	1E	68	
▲ Data run	320		0000	00000112	54	46	53	
Size	320	0x33	0000	00000128	55	AA	75	
Cluster count	321	51,264	0000	00000144	18	68	1A	
First cluster	324	786,432	0000	00000160	9F	83	C4	
⊿ Data run	327		0000	00000176	OF	00	C1	
Size	327	0x43	0000	00000192	66	FF	06	
Cluster count	328	51,223	0000	00000208	4B	00	2B	
First cluster	331	<u>12,340,201</u>	0000	00000224	66	81	FB	
Data run	335		0000	00000221	68	07	BB	
	240			00000240	55	16	1.6	
Attribute \$80	360			00000236	20	10	TO	
		~ ~ ~ ~		00000272	20	10	23	
I				00000288	06	66		

Rollback Partition Changes

Some critical partition layout changes made to a physical device are backed up by default. Users can roll back these changes at any point by using the **Roll back Partition Changes** tool. These changes are:

- Initialize Disk
- Create Partition
- Format Partition
- Delete Partition

To open the Rollback Partition Changes dialog, do one of the following:

- From the Tools menu, choose the Roll Back Partition Changes command.
- From the Tools tab in Command Bar, choose the Roll Back Partition Changes command.
- For a selected physical device (disk) node use the context menu **Roll Back Partition Changes** command.

	Action type Date/Time Status	
10 M	Partition Created 18/05/12 15:38:49 Valid	
	🥥 🧔 Partition Deleted 18/05/12 15:40:36 Valid	
100000400		
100000000000000000000000000000000000000	😡 🐼 Partition Deleted 18/05/1215:43:12 Valid	
	Reg Partition Deleted 18/05/12 15:43:22 Valid	
1 Sty	Reg Partition Deleted 18/05/12 15:43:28 Valid	
	Partition Created 18/05/12 15:51:16 Valid	
T31000524AS	Partition Created 18/05/12 20:45:20 Valid	
151000524A5	Repeatition Deleted 21/05/12 10:18:29 Valid	
	Representation Deleted 21/05/12 10:18:37 Valid	
2Y	Representation Deleted 21/05/12 10:18:45 Valid	
	Partition Deleted 21/05/12 10:18:56 Valid	
T31000524AS	Partition Deleted 21/05/12 10:19:02 Valid	
CZ-VERTEX3	Date: Monday, 21 May, 2012 10:18:29; Partition deleted at sector: 745244675; Total Sectors: 218402816; File System Code: 6;	
	Source Backup File: d:\projects_Development\Active Undelete 8\!Binaries\backup	ips\disk_ST1000DM003-9YN
All changes made t 10:18:29. Rollbad	to device ST1000DM003-9YN162 will be reverted to modification 'Partition Delet Ik is final and cannot be undone.	ed', made at 21/05/12

To roll back changes made to a physical device, select a restore point in the chronologically ordered list and click the **Roll Back** button to complete the changes.

Disk Image Overview

Definition

Disk Image is a mirror of your logical drive or physical device that is stored in one file. This can be useful when you want to backup the contents of the whole drive, and restore it or work with it later. Before you start recovering the deleted files, it may be a good idea to create a Disk Image for this drive, if you have enough space at another drive. Why? Because if you do something wrong while recovering the files

(for example, recovering them onto the same drive could destroy their contents), you still will be able to recover these deleted files and folders from the Disk Image that you have wisely created.

Active@ UNDELETE provides extensive functionality to operate with Disk Images. You can create image of either Logical disk, Device or Partition. Save it as one large file or split on image chunks with size you prefer for later use.

When you creating Disk Image, it stores in at least two files: one is *Configuration file* with file extension .DIM and the second - actual image body file. If you decide to save disk image chopped on peaces (chunks) then image body files can be as many as its required to save data.

Here is an example: If you save a raw disk image with the name MyImage, the application creates a file named MyImage.dim. This is the configuration file. Data is stored in a file named MyImage.dim.001. If more than one file is created, the next file is named MyImage.dim.002, and so on. The data file can be split in several files – chunks that can be useful if you want to save the Disk Image on a CD or Data DVD.

When to use Disk Image

Raw disk images are very helpful in a data recovery. Here are some reasons why a raw disk image can be used for data recovery:

- Data recovery technologies are based on searching the unused space on a partition for traces of deleted, lost or damaged files and folders. So-called "unused space" on a partition is not recognized by the file system and is not saved to a regular disk image. However, this space does contain valuable data information and it is saved to a raw disk image.
- The uncompressed raw disk image file contains a sequence of sectors that is unchanged from the original. There are no headers or other application-specific identifiers added. As a result, the raw disk image can be viewed by any data rescue software as a mirror of your drive. If the integrity of the data on your live disk is questionable, you may want to experiment with the data on the partition image instead.
- If file size is an issue, a compressed raw image may be used. Active@ UNDELETE is an example of data recovery software which can work with both compressed and uncompressed raw images.
- Raw images have no regard for the file system type. During the raw disk image recording process, all sectors are backed up. An image of any partition can be restored by using Active@ Disk Image software.
- If you want the data from a file to be restored from the disk image to the same exact location as they were before, then use a raw disk image. A regular image saves all current data but restores files to different sectors, allowing the partition to shrink or grow, depending on the size of the replaced file. In a regular situation, you should not be concerned about partition size. If the partition size is important, however, a raw image is the solution.

Create a Disk Image

Using Active@ UNDELETE you can create a *Disk Image* of a *volume* (logical drive) or a *disk* (physical device). To create disk image:

- **1.** Open the Create Disk Image dialog
 - From the Recovery Explorer toolbar, click Create Disk Image button or use command Actions
 > Create Disk Image from main menu;
 - Right-click the selected item and click **Create Disk Image** command from the context menu.
 - From the Disk Images tab in Command Bar, choose Open Disk Image command;
- **2.** Create Disk Image dialog

Select Destinatoin Path where to save Disk Image. Use compression to save space.								
Destination:	D:\Virtual Images\Dynamic test\drive_W_3990-6245.dim							
Description:	Disk Image made by Active@ UNDELETE - Data Recovery	/ Toolbox						
Compression: Options	None [Raw Data] Store Disk Image as chunks:	4.7 GB DVD-5 💌						
📝 Replac	e existing Disk Image files 🛛 Ignore Read/Write Errors	Use Disk Lock 🗹 Ignore Disk Lock Errors						
Restore Def	aults	Create Image Cancel Help						

Dialog Options Multiple disk selection

Additional areas on other disks can be selected in the **Physical Disks** list to be processed simultaneously. At least one selection must be made to begin disk image creation.

Disk Control

Use markers that indicates the first and last sectors on this control to specify an area to image.

Destination

Provide location of *Disk Image configuration file*. To browse to the path, click the ellipsis button [...]. All Disk Image chunk files will be created in the same folder with DIM file.

Description

Enter a brief description about this disk image for future reference. Optional.

Replace existing disk image files

If this option is set, all chunk files will be replaced with a new once if their file names are the same.

Ignore R\W Errors

Ignore Read and Write errors during the disk image creation.

Lock Disk

Source disk will be locked until Disk Image creation is complete or aborted;

Ignore Disk Lock Errors

Any errors related to disk lock will be ignored;

Compression

Choose one of the following:

- None [Raw Data] No compression is applied, sectors are stored in raw format.
- **Fast** Sectors are compressed before storing to the file using a fast compression algorithm.
- **Medium** Sectors are compressed before storing to the file using a slow but more effective compression algorithm.
- High High level of compression;
- · Highest Highest possible compression level ill be used;

Store Disk Image in chunks

Select this check box to save the Disk Image as a series of files with a specified size. Choose the file size from the drop-down list. This option may be useful if you want to write the Disk Image to CD-ROMs or DVD-ROMs. By default this check box is cleared and the Disk Image is stored in one large file;

3. Click **Create Image** to initiate disk image creation process with selected parameters.

4. Create image

During the process:

- To display or hide scanning events and progress details toggle More\Less Info button at any time.
- To terminate the process, click **Stop** at any time. Results may be not accurate or complete.



Important: The Destination Path for a Disk Image file must always be on another drive. File systems such as FAT16 and FAT32 do not support file sizes larger than 2GB and 4GB respectively. With these file systems it is not possible to create a Disk Image file for a drive as it is likely to grow larger than the size limit. The solution in this case is to do one of the following:

- Use a Destination Path drive that is formatted using Windows NT, Windows 2000, Windows XP and using NTFS;
- Create a Disk Image that is split into chunks of an appropriate size, keeping within the limits set by the file system;

F

Tip: Use *Create a Disk Image Wizard* on page 87 for the same purpose.

Open Disk Image

You may open a *Disk Image* to browse for files and folders or to scan for deleted files and folders.

- **1.** To open the Open Disk Image dialog, do one of the following:
 - From the Disk Images tab in Command Bar, choose Open Disk Image command;
 - From the main toolbar, click **File** > **Open** > **Open Disk Image**.
 - From Welcome View, click Open Disk Image button in Default Actions group;
- 2. Open disk image using *Configuration file*



Use **Browse** button to locate .DIM (Disk Image Configuration) file. Once it selected, file ill be opened and presented with detailed preview of Disk Image information.

To open Disk I UNDELETE or si Click Compose Disk Image:	mage click Browse to select <i>Disk Image Configuratio</i> elect any other third pary Disk Image files such <i>Virtual</i> button to manualy assemble Disk Image from chunks. D:\Temp\di\drive_F_3990-6245.dim	(*.DIM) file created by Active@ Cor WMware. User frendly Disk Image Label				
Caption (Display Name):	Disk Image					
•						
Name	Value	Description				
Date Created	04/05/2012 16:50:26					
Creation Time	00:00:47					
Image Type	LSoft Disk Image					
Caption	Disk Image					
Description	Disk Image made by Active@ UNDELETE - Data Recovery Toolbox					
Image completed	No					
Image Validated	No					
Compression Level	Medium					
Media type	Fixed Disk					
Bytes per Sector	512	Click Customize to shange				
Sectors per Track	63	Dick Customize to change				
Tracks per Cylinder	255	Disk image attributes				
Checksum	0					
Disk Image Chunks						
File 1	D:\temp\di\drive_F_3990-6245.001					
	ОК Си	istomize Cancel Help				

Click **OK** to open disk image or click **Compose** button to alter disk image configuration (see next step).

3. Compose Disk Image [optional]

If there is no DIM file for Disk Image or to open third party Disk Images click **Compose** button.

Add Disk Image chunks and provide disk geometry attributes if necessary. Final Disk Image configuration can be saved in Disk Image Configuration (.DIM) file for later use.									
Caption (Disp	lay Name): Disk I	mage							
Image chunks:	D:\temp\di\driv	⊧_F_3990-6245.001			An A	id ove e Up Down			
Image Type	: LSoft Disk Image	•	Media Type: Fi:	xed Disk	v				
Bytes per Sec	tor: 512	Sector per Track: 63	Track per C	ylinder: 255					
					Save DIM File A	.s			
				ок	Cancel Hel	e de			

Dialog Options Caption

Enter any label to distinguish newly opened disk image among other devices and disks.

Disk Image Chunks

A Disk Image consists of one or many files, which contains actual image data. A Disk Image can be cut into several files (chunks) during creation for better space allocation. In this list you have to specify all these files which make ups the image. To Add a Disk Image chunk to the list click the **Add New** button and use browse for a file dialog to select a file. To Remove a Disk Image chunk, select this chunk in the list and click the **Remove** button. To modify the order of Disk Image chunks, select any chunk you wish to relocate and use the **Up** and **Down** buttons to move a selected chunk in the chunk stack.

Image Type

Select image type you about to open. Usually it assigned automatically, depending on Disk Image chunks added.

- Raw Disk Image Raw fragment of a disk;
- LSoft Disk Image Disk Image created by any LSoft Technology product;
- Virtual PC Disk Images from Virtual PC software;
- VMWare Image Disk Images from VMWare software;

Media Type

Select appropriate media type. Usually it assign automatically. Use **Fixed Disk** by default.

Bytes per Sector

Enter sector size in bytes;

Sectors per Track

Enter track size in sectors;

Tracks per Cylinder

Enter cylinder size in tracks;

Save DIM File as...

In case of manual composition of Disk Image properties you may save final configuration file for later use;

4. Confirm and open disk image

Click **OK** to open Disk Image. A disk image node appears in Recovery Explorer.



Important: Use Open a Disk Image Wizard on page 88 for the same purpose.

Virtual Partition (Logical drive clone)

A virtual logical partition is a copy (a clone) of a logical drive using a defined geometry that emulates a real logical drive or partition. If you have a logical drive that is recognized by Windows and you cannot access the data in that drive, you may be able to gain access to your data by creating a virtual partition copy.

Create A Virtual Partition

To create a virtual partition on a data storage device:

- Select a device item in **Recovery Explorer**.
- Right-click the selected item and click **Create Virtual Partition** command from the context menu.

Use the **Create Virtual Partition dialog** to set virtual partition options and complete the operation.

Create a Virtual Copy of An existing partition

1. In **Recovery Explorer**, select a logical drive or a partition and do one of the following:

- From the Recovery Explorer toolbar, click Create Virtual Copy.
- Right-click the selected item and click **Create Virtual Copy** from the context menu.
- 2. A partition copy appears under the corresponding physical device item.

Including the **Modify Partition** command, you can execute all tasks applicable to a logical drive on this drive copy.

Edit Virtual Partition Properties

To alter the properties of a virtual drive, do one of the following:

- 1. Select a virtual partition item in Recovery Explorer.
- 2. To open the Edit Boot Sector Template dialog, do one of the following:
 - From the Recovery Explorer toolbar, click Edit Partition.
 - Right-click the selected item and click Modify Partition from the context menu.
- **3.** In the **Edit Boot Sector Template dialog**, make changes to the *Boot Sector Primary* and *Boot Sector Copy* separately or simultaneously. See the *Edit Boot Sectors* on page 52 dialog for details
- 4. Click Save.

Working with a corrupted RAID system

If you have a corrupted *RAID* configuration and one or more drives in the array are damaged, you can combine the healthy drives together with the damaged drives in a virtual disk array. If the damaged drives are inaccessible, you can substitute a "dummy" drive as a replacement. Active@ UNDELETE simulates the RAID assembly and you can scan this virtual array as a logical device.

To assembly virtual RAID:

- 1. Open the Virtual Disk Array Assembly dialog
 - From the Tools menu, choose Create Virtual RAID (RAID) command.
 - From the Tools tab in Command Bar, choose Create Virtual RAID command.
| Select Physical Devi
boundaries if nee
Data Storage Devices availab | ces from the list of a
ded.
le for RAID Assembl | available devices in
y | correct ord | ler and RAID Ty | pe. For each sele | ected device you can specify individually device |
|--|---|---|---------------------------|-------------------------|--|--|
| Name | Status | Partitioning | Total Size | Total Sectors | Bytes/Sector | |
| Image: A state of the state | Ready Initialized | MBR (Basic) | 932 GB | 1953525168 | 512 | |
| All | Read Initialized | MBR (Dynamic) | 932 GB | 1953525168 | 512 | |
| | Ready. In Vized | MBR (Basic) | 932 GB | 1953525168 | 512 | |
| A standard stand
standard standard stand
standard standard stan
standard | Ready, Initian | MBR (Basic) | 55.9 GB | 117231408 | 512 | |
| | Read
Read | Data Storage [|)evices | 976773168 | 512 | |
| 6.72 GB L
60.5 GB P
ta Storage For each
selected if
set | 3. Array bour
device an exact
necessary. Clic
baoundaries to | 1
5 c 6.84 GB Unall
daries
boundaries can
king on partition
that partition | be
will ion C | ht (W:)
A GB P 372 G | up (J:)
B Primary NTFS
A
e
on Size | 2.49 MB
2.49 MB
array order can be changed if its
ssential for selected RAID type |
| Local Disk (T:)
83.6 GB Primary L
134 | ady, Initializa | d Disk 466 GE | l Disk (R:)
GB Logical | 0 97 | 4. RAID T | Type and Block 8.71 MB Unalloci |
| ffset, sector: 456390658 | 456390
Size, sectors: | 343454721 | 99845379 | | Select RAI
size | Size
D type and block
as required |
| AID Options
AID Type: Mirror (RAID 1) | | ▼ Block Size: | 512 bytes [l | Default] 🔻 Se | ctors per Block: | 1 |
| To create Mirror (RAID | 1) select two Data | Storage devices. | | | | |
| Reset | | - | | | | Create Virtual RAID Cancel Help |

Dialog Options

RAID Type

RAID type. See article: *Disk Arrays (RAID's)* on page 102 for information about how to select proper RAID type.

Block size

Size of stripe block in bytes. Applicable only for stripe or RAID-5 array types.

Sectors per Block

Size of a block in sectors

Offset

Offset of selected disk area from beginning if a disk in sectors.

Size

Size of selected area in sectors.

2. Select disks from list of Available Data Storage Devices by double-click or using check marks;

- Double-click a disk in the **Available disks** list to move it to the **Selected disks** list or select a disk in the **Available disks** list and click **Add** button.
- To change the order of a disk in the **Selected disks** list, select it and click **Move Up** or **Move Down**.
- To remove a disk from the Selected disks list double-click a disk in the Selected disks list or select disk and click Remove button.
- To remove all disks from the **Selected disks** list, click **Remove All**.
- **3.** Specify used disk range.
 - Drag sliders to desired position to mark used disk range or
 - Enter exact values in **Offset** and **Size** text boxes



Note: For each selected disk, used disk range can be set individually.

Tip: Click on disk partition or unallocated space to select entire partition or unallocated space to disk range for RAID assembly.

- 4. Specify RAID type, block size and sectors per block if necessary.
 - In **stripe block size** text box specify the stripe block size in kilobytes (Stripe and RAID-5 arrays only).
 - For RAID5 select a proper parity layout from drop-down list box. See *Disk Arrays (RAID's)* on page 102 for parity layout reference.
- 5. Click Create. The Processing... dialog appears.



Note: To display creation events and progress details, click **Details**.



Note: To terminate the creation process, click **Stop** at any time. Results may be not accurate or complete

If a virtual disk array is created successfully, a new node appears in **Recovery Explorer** tree.

If a virtual disk array is not created, or if it is created with errors, return to 1 and try again with different disks, or with a different disk order and RAID options.

Active@ UNDELETE Wizards Overview

Active@ UNDELETE Wizards are sets of step-by-step guided tools that help you to accomplish different recovery and disk management tasks.

Wizards can be started at any time from:

- Main application menu Wizards;
- From Welcome view or
- From Command Bar on lefts side of Main View.

Active@ UNDELETE has following wizards:

Data Recovery Wizards

- Recover Files by Signature Wizard on page 77
- Recover files from a deleted partitions wizard on page 80
- File Recovery Expert Wizard on page 82
- Recover Deleted Files Wizard on page 75

Partition Management Wizards

- Create a New Partition Wizard on page 84
- *Restore a Deleted Partition Wizard* on page 82

Advanced

• Create a virtual RAID Array Wizard on page 85

Disk image Wizards

- Create a Disk Image Wizard on page 87
- Verify a Disk Image Wizard on page 90
- Open a Disk Image Wizard on page 88

Active@ UNDELETE Recovery Wizards Overview

Data Recovery Wizards

- *Recover Files by Signature Wizard* on page 77
- Recover files from a deleted partitions wizard on page 80
- File Recovery Expert Wizard on page 82
- Recover Deleted Files Wizard on page 75

Recover Deleted Files Wizard

A wizard designed to recover accidentally deleted files from existing logical drives.

- 1. Select Logical Drives
 - Select at least one logical drive to scan for deleted files.
- 2. Scan

Scan selected logical drives for deleted files and folders. The scan can be stopped at any time.

3. Review volume scan results

Use the *File Filter Toolbar* on page 33 to narrow down search results. By default, only deleted **Files and Folders** are shown. To view all files detected on scanned devices, click the **Reset** filter to default button in the toolbar.

Files List: De Select de	etected Files and Folders sired Files for recovery using ch	eck marks. I	File Filter ca	an be used to narrow d	own files list.			1
Group by: Fil	e and Folders [Default] 🔻 🛛 Fi	lter by:	、 、			» D 🛛		
Name	^	Status	Size	Created	Accessed (Deleted)	Attributes	ID	
a 📄 🍥 alc	otoffii (I:)							
Sector	MSI1e tmp	Deleted	0 bytes	12/05/12 1 14:42	12/05/12 13:14:42	D	54550	
🗖 🗐	Data at a d Eile a san annu	and here	0 bytes	12/05/12 13:14	12/05/12 13:14:43	D	54550	
E 6	Detected Files can grou	ped by	0 bytes	03/03/12 10:41:50	93/03/12 10:41:50	D	54548	
	Date, The Type of Appli	cation	0 bytes	24/11/11 03:01:39	2 11 03:01:39	D	54518	
Þ 📄 🗳	temp	Healthy	341 MB	25/08/11 19:38:12	25/0 19:38:12	2 D	14251	
i 📄 🌍 lor	nt (W:)							
Þ 🔲 🖄	! Lost & Found !	Healthy	1.24 MB		Use File filter	to narrow se	can ⁻⁵	
Þ 🔲 📹	\$RECYCLE.BIN	System	325 MB	17/10/09 20:01:26	c re	sult	69	
	381856	Deleted	0 bytes	11/02/11 03:21:11	1		56	
	569497	Deleted	0 bytes	15/04/11 05:04:18	15/04/11 05:04:18	D	4462	
	942607	Deleted	0 bytes	10/05/11 05:57:32	10/05/11 05:57:32		4462	
	сору	Healthy	2.30 GB	05/03/09 12:52:35	26/04/11 13:47:0:	5 D	20976	
	Island	Deleted	1.35 MB	12/05/12/12/14/42	12/05/12 11:57:34	+ D	18410	
	MSI2ddbftmp	Deleted	0 bytes	22/09/12 15:14:45	12/03/12 15:14:45	D	4400	
	MSI/0277 tran	Deleted	0 bytes	08/03/12 08:54:58	08/02/12 09:54:58	D	4407	
	MSI72da5 tran	Deleted	0 bytes	00/03/12 03:02:50	00/03/12 03:02:50	D	4470	
	System Volume Informatio	n System	28.0 KB	16/12/08 11:18:11	26/03/12 17:51:3		27	
	Windows NT4	Deleted	345 MB	05/03/12 11:13:27	05/03/12 11:20:23	3 D	7777	
	Windows NT4 FAT	Deleted	0 bytes	05/03/12 11:20:23	05/03/12 11:20:23	D	4476	
	PartMan.exe	Deleted	3.82 MB	26/10/10 13:08:44	26/10/10 13:08:44	A	4454	
						🎾 47 of	88958 file(s)	50 file(s) and 18 folder(
Help								Next > Close

Select file(s) to recover and click **Next** to continue.

4. Recover Files

File Recovery Select required op	tions and click 'Next' to continue.		
	Recover files to: d:\temp\recovered\		
	Naming options		
	 Use original file names (recommended) 		
	Rename files to:		
	recovered_		+ (00001)
	Existing files conflict resolution		
	 Generate unique file name (recommended) 	Overwrite without prompt	
	Ask before overwrite	Skip existing files	
	Options		
	Create original folder (group) structure	Recover Named Streams	
	Selected 3 files and folders to recover.		
Help		< <u>Back</u> <u>Next</u> >	Close

Naming options Use original file names

Names of detected files will be preserved only if no file with that name exists in the destination directory.

Rename files

All files will be renamed by their given specified file root name and added enumeration ID. The file's extension remains intact.

Existing files conflict resolution Unique file name

If a file with the same name exists in the destination folder, files with a unique name will be generated to avoid overwriting.

Ask before overwrite

If a file with a certain name already exists in the destination folder, the application will ask the user what action to take.

Overwrite without prompt

All files will be overwritten in the event they already exist in the destination folder.

Skip existing files

If file with the same name exists in the destination folder, the recovery of a new file will be skipped.

Additional Options

Create Folder Structures

When this option is selected, files will be recovered with their original folder structures e.g. the original folder hierarchy as it was on the source storage device. In case the files were organized in groups (by date, file extensions, or an associated application), then such groupings will be created by the folder structure in the location where the files will be recovered to.

Recover Name Streams

With this option on, files will be recovered with their original named streams.

Verify default recovery options and click **Next** to continue.

5. Confirm Recovery

Review recovery options, destination path etc. and click **Recover** to start recovering files.

6. Complete wizard

Click to close the Wizard. After the recovery wizard has completed, you can open the destination folder to which the files were recovered. Use the default OS File Explorer or repeat the wizard again to scan another logical drive.



Note: All scan results will remain available after the wizard closes.

Recover Files by Signature Wizard

Some files has unique patterns, allowed them to be found by advanced scan process. This Wizard will guide you via simple steps to help you to detect files by File Signature. It will allow you to review and recover detected files. To run this Wizard - click Recover Files by Signature from the Wizards menu, or click Recover Files by Signature button in Tools Tab in Command Bar.

1. Select Logical Drives

Select at least one Logical Drive to scan for deleted files by File Signatures.

2. Select File Signatures to detect

Scan selected Logical Drives for deleted files and folders. Scan can be stopped at any time.

🕨 🔲 🏐 Microsoft Office Documents		Select All
👂 🗐 🏐 Formatted Text files	ſ	
4 🔲 🟐 Compressed Archives		Clear All
ZIP Archives	zip	
🔺 🔳 뉔 Images and Camera Raw files		=
🔲 📄 Bitmap Images	bmp	
🔽 📄 Canon CRW Raw Images	crw	
🔲 📄 Icon Files	ico	
JPEG Images	jpg	
🔲 📄 Konica Minolta Raw Images	mrw	
🔲 📄 Fuji FinePix Raw Images	raf	
TIFF Images	tif	
🔲 📄 Sony Alpha Raw Images	arw	
🔲 📄 Canon CR2 Raw Images	cr2	
Adobe Digital Negative	dna	*
.crw; .jpg;		



Note: Number of File Signatures impacts the scanning time.

3. Confirm and Scan

Review scan options and initiate scan process by clicking Scan button. The scan can be stopped at any time.

4. Review volume scan results

Use the *File Filter Toolbar* on page 33 to narrow down search results. By default, only deleted **Files and Folders** are shown. To view all files detected on scanned devices, click the **Reset** filter to default button in the toolbar.

es List: Detected Files and Fold Select desired Files for recovery of	lers using check marks. I	File Filter ca	n be used to narrow d	own files list.				ß
oup by: File and Folders [Default]	▼ Filter by:				8) (B		2 B
ne	Status	Size	Created	Accessed (Deleted)	Attri	butes	ID	
📄 🍛 alotoffii 🛛 (I:)				1	-			
🔲 🏐 MSI1e 🐂 👘 🕅	Deleted	0 bytes	12/05/12 1 14:42	12/05/12 13:14:42	D		54550	
		0 bytes	12/05/12 13:14	12/05/12 13:14:43	D		54550	
Detected Files ca	n grouped by	0 bytes	03/03/12 10:41:50	3/03/12 10:41:50	D		54548	
Date, File Type of	r Application	0 bytes	24/11/11 03:01:39	2 11 03:01:39	D		54518	
🖻 📄 🍓 temp	Healthy	341 MB	25/08/11 19:38:12	25/0 19:38:12	2 D		14251	
📄 🍛 lont (W:)				\sim				
▷ 📄 🍓 ! Lost & Found !	Healthy	1.24 MB		Use File filter	to nar	row sca	-5	
Image: Second S	System	325 MB	17/10/09 20:01:26	c re	sult		69	
	Deleted	0 bytes	11/02/11 03:21:11	1			56	
🔲 🍏 _569497_	Deleted	0 bytes	15/04/11 05:04:18	15/04/11 05:04:18	D		4462	
🔲 🏐 _942607_	Deleted	0 bytes	10/05/11 05:57:32	10/05/11 05:57:32	D		4462	
Þ 📄 🏐 copy	Healthy	2.56 GB	05/03/09 12:52:35	26/04/11 13:47:0	5 D		20976	
▷ 📄 🍓 island	Healthy	1.55 MB	05/03/09 12:50:24	12/05/12 11:57:34	I D		18410	
📄 🏐 MSI1e145.tmp	Deleted	0 bytes	12/05/12 13:14:43	12/05/12 13:14:43	D		4466	
🔲 🏐 MSI3ddbf.tmp	Deleted	0 bytes	23/08/11 10:29:02	23/08/11 10:29:02	D		4467	
📄 🏐 MSI49277.tmp	Deleted	0 bytes	08/03/12 08:54:58	08/03/12 08:54:58	D		4478	
📄 🏐 MSI72da5.tmp	Deleted	0 bytes	09/03/12 03:02:59	09/03/12 03:02:59	D		4464	
🖻 📄 🏐 System Volume Info	rmation System	28.0 KB	16/12/08 11:18:11	26/03/12 17:51:38	B HSE)	27	
D Windows NT4	Deleted	345 MB	05/03/12 11:13:27	05/03/12 11:20:23	B D		7777	
📄 🏐 Windows NT4 FAT	Deleted	0 bytes	05/03/12 11:20:23	05/03/12 11:20:23	D		4476	
PartMan.exe	Deleted	3.82 MB	26/10/10 13:08:44	26/10/10 13:08:44	А		4454	
					>	47 of 8	8 8958 file((s) 🗟 50 file(s) and 18 folde
Help								Next > Close

Select file(s) to recover and click **Next** to continue.

5. Recover Files

Recover files to: d:\temp\recovered\		
Naming options		
 Use original file names (recommended) 		
Rename files to:		
recovered_		+ (00001)
Existing files conflict resolution		
Generate unique file name (recommended)	Overwrite without prompt	
Ask before overwrite	Skip existing files	
Options		
Create original folder (group) structure	Recover Named Streams	
Selected 3 files and folders to recover.		

Naming options Use original file names

Names of detected files will be preserved only if no file with that name exists in the destination directory.

Rename files

All files will be renamed by their given specified file root name and added enumeration ID. The file's extension remains intact.

Existing files conflict resolution

Unique file name

If a file with the same name exists in the destination folder, files with a unique name will be generated to avoid overwriting.

Ask before overwrite

If a file with a certain name already exists in the destination folder, the application will ask the user what action to take.

Overwrite without prompt

All files will be overwritten in the event they already exist in the destination folder.

Skip existing files

If file with the same name exists in the destination folder, the recovery of a new file will be skipped.

Additional Options Create Folder Structures

When this option is selected, files will be recovered with their original folder structures e.g. the original folder hierarchy as it was on the source storage device. In case the files were organized in groups (by date, file extensions, or an associated application), then such groupings will be created by the folder structure in the location where the files will be recovered to.

Recover Name Streams

With this option on, files will be recovered with their original named streams.

Verify default recovery options and click **Next** to continue.

6. Confirm Recovery

Review recovery options, destination path etc. and click **Recover** to start recovering files.

7. Complete wizard

Click to close the Wizard. After the recovery wizard has completed, you can open the destination folder to which the files were recovered. Use the default OS File Explorer or repeat the wizard again to scan another logical drive.



Note: All scan results will remain available after the wizard closes.

Recover files from a deleted partitions wizard

In this wizard, unallocated spaces on data storage devices are scanned for deleted partitions. After partitions are detected, they should be scanned for files and folders.

1. Scan unallocated space

Select unallocated area by placing check marks in the data storage devices tree and click **Next** to continue.



Page Options File system lookup

Select File System of a partitions to be detected

2. Confirm and scan for deleted partitions

Review and confirm the unallocated space scan parameters and click the Scan button to start the scanning process. While the process is in progress, you can cancel it at any time by clicking Stop at the bottom of the screen.

3. Scan detected partitions

Review list of detected partitions and select at least one of them to scan for missing files.

Name	File System	Status	Restore Statu	Total Size	First Sector	Total Sectors	Serial Number	
📄 🥪 Local Disk (8:)	NTFS	Bad	Can be Re	15.7 MB	163750544	32129	3949-636F	
📄 🥔 folder_test (7:)	NTFS	Poor	Overlappe	388 GB	163723769	813049398		
📄 🥔 folder_test (6:)	NTFS	Not Bad	Overlappe	388 GB	163713061	813060106		
👽 🥪 folder_test (5:)	NTFS	Good	Can be Re	15.7 MB	163718415	32129	3949-636F	
👿 🥔 cleo (2:)	NTFS	Excellent	Can be Re	6.84 GB	149774058	14345981	398D-6D82	
📝 🥔 NEW VOLUME (3:)	FAT	Excellent	Can be Re	2.93 GB	157565520	6152864	6C4E-B6FF	
👽 🥪 MOORE (4:)	FAT32	Excellent	Can be Re	196 MB	163718415	401625	78B8-4FCC	

4. Review volume scan results

Use the *File Filter Toolbar* on page 33 to narrow down search results. By default, only deleted **Files and Folders** are shown. To view all files detected on scanned devices, click the **Reset** filter to default button in the toolbar.

Files List: De Select de	etected Files and Folders sired Files for recovery using ch	ieck marks. F	File Filter ca	an be used to narrow d	own files list.			× /
Group by: Fil	e and Folders [Default] 🔻 🛛 Fi	lter by:	、 、			» 🗈 🛛		
Name	^	Status	Size	Created	Accessed (Deleted)	Attributes	ID	
🔺 📄 🍥 ald	otoffii (I:)							
E 🕤	MSILe tmp	Deleted	0 bytes	12/05/12 1 14:42	12/05/12 13:14:42	D	54550	
🗖 🗇	Detected Files are seen	and here	0 bytes	12/05/12 13:14	12/05/12 13:14:43	D	54550	
Solution	Detected Files can grou Date File Type or Appli	ped by cation	0 bytes	03/03/12 10:41:50	93/03/12 10:41:50	D	54548	
	Date, The Type of Appli	cation	0 bytes	24/11/11 03:01:39	2 11 03:01:39	D	54518	
Þ 📃 🖄	temp	Healthy	341 MB	25/08/11 19:38:12	25/0 19:38:12	2 D	14251	
⊿ 📄 🌍 loi	nt (W:)				\frown			
Þ 🔲 🖄	! Lost & Found !	Healthy	1.24 MB		Use File filter	to narrow s	can ⁻⁵	
D 🗐 📹	\$RECYCLE.BIN	System	325 MB	1//10/09 20:01:26	c re:	sult	69	
	381856	Deleted	0 bytes	11/02/11 03:21:11	1. /04 //1 05 04 10	0	1162	
	569497	Deleted	0 bytes	15/04/11 05:04:18	15/04/11 05:04:18	D	4462	
	942007	Deleted	0 bytes	05/03/11 05:57:32	10/05/11 05:57:32	D	4402	
	copy	Healthy	2.30 GB	05/03/09 12:52:35	20/04/11 13:47:03		20970	
	Island	Deleted	1.33 IVID	12/05/09 12:30:24	12/05/12 11:57:54	+ D	10410	
	MSI2ddbftmp	Deleted	0 bytes	22/09/12 13:14:43	22/09/12 13:14:43	D	4400	
	MSI49277 tmp	Deleted	0 bytes	08/03/12 08:54:58	08/03/12 08:54:58	D	4407	
	MSI72da5 tmp	Deleted	0 bytes	09/03/12 03:02:59	09/03/12 03:02:59	D	4470	
	System Volume Informatio	n System	28.0 KB	16/12/08 11:18:11	26/03/12 17:51:38		27	
	Windows NT4	Deleted	345 MB	05/03/12 11:13:27	05/03/12 11:20:23	3 D	7777	
- <u>-</u>	Windows NT4 FAT	Deleted	0 bytes	05/03/12 11:20:23	05/03/12 11:20:23	D	4476	
	PartMan.exe	Deleted	3.82 MB	26/10/10 13:08:44	26/10/10 13:08:44	A	4454	
						🎾 47 of	f 88958 file(s)	50 file(s) and 18 folder(s)
Help								Next > Close

Select file(s) to recover and click **Next** to continue.

5. Recover Files

File Recovery Select required op	tions and click 'Next' to continue.		
	Recover files to: d:\temp\recovered\		
	Naming options		
	 Use original file names (recommended) 		
	Rename files to:		
	recovered_		+ (00001)
	Existing files conflict resolution		
	 Generate unique file name (recommended) 	Overwrite without prompt	
	Ask before overwrite	Skip existing files	
	Options		
	Create original folder (group) structure	Recover Named Streams	
	Selected 3 files and folders to recover.		
Help		< <u>Back</u> <u>Next</u> >	Close

Naming options Use original file names

Names of detected files will be preserved only if no file with that name exists in the destination directory.

Rename files

All files will be renamed by their given specified file root name and added enumeration ID. The file's extension remains intact.

Existing files conflict resolution Unique file name

If a file with the same name exists in the destination folder, files with a unique name will be generated to avoid overwriting.

Ask before overwrite

If a file with a certain name already exists in the destination folder, the application will ask the user what action to take.

Overwrite without prompt

All files will be overwritten in the event they already exist in the destination folder.

Skip existing files

If file with the same name exists in the destination folder, the recovery of a new file will be skipped.

Additional Options

Create Folder Structures

When this option is selected, files will be recovered with their original folder structures e.g. the original folder hierarchy as it was on the source storage device. In case the files were organized in groups (by date, file extensions, or an associated application), then such groupings will be created by the folder structure in the location where the files will be recovered to.

Recover Name Streams

With this option on, files will be recovered with their original named streams.

Verify default recovery options and click **Next** to continue.

6. Confirm Recovery

Review recovery options, destination path etc. and click **Recover** to start recovering files.

7. Complete wizard

Click to close the Wizard. After the recovery wizard has completed, you can open the destination folder to which the files were recovered. Use the default OS File Explorer or repeat the wizard again to scan another logical drive.



Note: All scan results will remain available after the wizard closes.

File Recovery Expert Wizard

The File Recover Expert wizard is a universal guided tool that allows the recovery of files from logical drives or data storage devices where files are detected by file signatures.

Select the logical drive or data storage device to be scanned for deleted files. Depending on your selection, use the following instructions:

Recover files from Logical Drives	See the <i>Recover Deleted Files Wizard</i> on page 75 for further instructions.
Recover files from Data Storage Devices	See Recover Files in <i>Recover Files by Signature Wizard</i> on page 77, <i>Recover files from a deleted partitions</i> <i>wizard</i> on page 80 for further instructions.

Restore a Deleted Partition Wizard

This wizard guides you through simple steps to help you to detect and restore deleted or damaged partitions. The Restore Partition wizard guides you through three processes:

- 1. Detecting deleted or damaged partitions.
- 2. Analyzing the content of a detected partition and optionally modifying its geometry.

3. Restoring the partition.

To start this wizard, do one of the following:

- From the wizards menu, click Restore Deleted Partitions.
- · Select the Partition Management tab in the command bar and click Restore Deleted Partitions

When the Restore Partition wizard starts for the first time, the first screen describes the process. Clear the "Show this page next time?" check box to avoid seeing this screen the next time you run this wizard.

1. Scan unallocated space

Select unallocated area by placing check marks in the data storage devices tree and click **Next** to continue.

Name	Status	Partitioning	Total Size	Total Sectors	Bytes/Sector	*
▲	Ready, Init	GPT (Basic)	932 GB	1953525168	512	
Vnallocated Space			105 GB	221208542	512	
🥟 Local Disk (M:)				333938688	512	
Eocal Disk (O:)		wore then one un	allocated are	218122240	512	E
👿 🥔 Unallocated Space		can be selected	at the time	1180255631	512	
▲	Ready, Init	MBR (Basic)	932 GB	1953525168	512	
🥟 Major (D:)			932 GB	1953519616	512	
📄 🥔 Unallocated Space			1.71 MB	3504	512	
▲	Ready, Init	MBR (Basic)	55.9 GB	117231408	512	
🥟 System Reserved (2:)		Assianina proper	File Svstem	filters may say	e 512	
🥪 Local Disk (C:)		scanning time ar	nd narrow do	wn final results	512	
📄 🍘 Unallocated Space	_		O MR	5885	512	
▲	Ready, Init	MBR (Basic)	466 GB	976773168	512	-
	Ready, Init p: 🔽 NTFS ated]	MBR (Basic)	466 GB IT ♥ FAT	976773168	512 2/Ext3/Ext4	

Page Options File system lookup

Select File System of a partitions to be detected

2. Confirm and scan for deleted partitions

Review and confirm the unallocated space scan parameters and click the Scan button to start the scanning process. While the process is in progress, you can cancel it at any time by clicking Stop at the bottom of the screen.

3. Review scan results

Select the partition to restore from the list of detected partitions and, if partition can be restored, click the **Next** button to continue.

Nam	ne		File System	Status	Restore Status	Total Size	First Sector	Total Sectors	Serial Number	
	TREMM	1ENS (2:)	FAT32	Excellent	Can be Restored	105 GB	2048	221206526	4FC6-1E86	
•	Device Map control marks selected partition with scaled geometry and indicates by green color (restorable) or by red if its not. Image: Color of the sector of the sec									
	Local Disk (M:) Local Disk (0:) 105 GB Unalloc 159 GB Primary Ui Obtected Drive: 105 GB, FAT32 [TREMMENS (2:)], Status: Excellent: Click Next to proceed with restoration of selected partition:									

4. Confirm Partition Recovery

Review and confirm the partition recovery and click the Restore button to restore the selected partition. If the action is successful, the restored partition will appear in the data storage device area of the Recovery Explorer.

Create a New Partition Wizard

This wizard guides you through simple steps to help you to create a new partition on a data storage device. When the **Restore Partition** wizard starts for the first time, the first screen describes the process. Clear the **Show this page next time?** check box to avoid seeing this screen the next time you run this wizard.

1. Select Unallocated Space

Select the unallocated space where the new partition must be created and click the Next button to continue.

Name	Status	Partitioning	Total Size	Total Sectors	Bytes/Sector	-
▲	Ready, Initiali	ized GPT (Basic)	932 GB	1953525168	512	
📄 🥔 Unallocated Space			105 GB	221208542	512	
🥪 Local Disk (M:)			159 GB	333938688	512	
🥪 Local Disk (O:)			104 GB	218122240	512	Ξ
🔽 🥏 Unallocated Space			563 GB	1180255631	512	
▲	Ready Initiali	ized MBR (Basic)	932 GB	1953525168	512	
🥟 Major (D:)			932 GB	1953519616	512	
📄 🥔 Unallocated Space	ſ	Only one unallocated s	B	3504	512	
▲	Ready, Initia	be selected to creat	te new B	117231408	512	
🥟 System Reserved (1:)		partition	B	204800	512	
🥪 Local Disk (C:)	l	'	В	117020672	512	
📄 🥔 Unallocated Space			1.90 MB	3888	512	
▲	Ready, Initiali	ized MBR (Basic)	466 GB	976773168	512	
🕅 📾 Unallocated Space			6.72 GB	14100344	512	Ŧ
<						- P

2. Select partition geometry (attributes)

\L \PhysicalDrive0 [1953525168 sectors]
Partition Geometry Maximum Partition size: 563 GB Offset, sector: 773269504 Size, MB: 211902 Measure in Sectors Use sliders set partition boundaries
New Logical Drive will be created in Primary Partition, starting from 773269504 sector with size 207 GB [433977156 sectors].

Page Options Offset

First sector of created partition. It can be set exact by numerical value entered in text box or by moving left slider in Device Map control;

Size

Partition size can be set in megabytes or in sectors, depending on state of **Measure in Sectors** check box;

Measure in sectors

Set this option on, to use sectors instead of megabytes as partition measurements;

3. Partition Attributes

Page Options Mark Partition as Active

Newly created partition will be set as Active Partition

Assign Drive letter

For Primary Partition or Logical Drive on extended partition drive letter can be assigned from the list of available in the system drive letters

4. Format Partition

This step is optional. Click set so **Do not format new partition** and click **Next** button to continue.

Page Options Volume Label

Enter distinctive volume label;

File System

Select on of the file system supported;

Allocation unit size

Allocation unit size depends on *File System* selected. Leave *Default* for mots of the cases;

5. Confirm Actions

Review and confirm new partition attributes and click **Create Partition** button to initiate creation process.

6. Complete

Click **Finish** to close the Wizard.

A new partition will appear for corresponded *disk* in **Recovery Explorer**.

Create a virtual RAID Array Wizard

This Wizard will guide you via simple steps to help you to re-assemble a damaged or disassembled RAID set to create a Virtual Disk Array. It will allow you to review and recover data located on the RAID set.

- To create a Virtual Disk Array you must specify the type of disk array (RAID type), disks and array geometry.
- You can manipulate the number and order of disks in the array.
- You can specify your own Virtual Disk Array geometry or accept the default values.

To run this Wizard - click **Create Virtual Array** from the **Wizards** menu, or click **Create Virtual RAID** button in Tools Tab of Command Bar.

1. Select Array Type

Select a RAID type to be reconstructed:

Composed of disk space located on several disks consecutively.
Stores data in stripes distributed on two or more disks.
Duplicates data identically on two disks.
Stores data in stripes distributed on three or more disks with parity control.

2. Select Array Disks

Choose disks to compose a Virtual Disk Array.

Use the **[Damaged Disk]** virtual device instead of the disk that is physically damaged (e.g. a non spinning disk), or is known to contains invalid information. Some RAIDs types (Mirror, RAID5) allow you to recover information even if one of the disks is lost this way.

3. Disk Options

Choose default geometry options or specify custom values.

Name	Status	Туре	Total Size	Selection Offset	Selection Size						
\\.\PhysicalDrive0	Ready, Initialized	Fixed Disk	932 GB	555147264	218122240						
\\.\PhysicalDrive1	Ready, Initialized	Fixed Disk	932 GB	396452498	112953747						
\\.\PhysicalDrive2	Ready, Initialized	Fixed Disk	932 GB	0	1953525168						
Damaged Disk	Ready	RAID-5 Temporal									
				Change di	sk order in						
				array if n	ecessary 📄	******					
TREMOW (E:) New	Volur	PORTO (Q		_							
178 GB Dynar 952	MB DV 9.77 GB L	53.9 GB D 689	GB Unallee	atad		1.00 MB Primai					
	396452498	50940	6240 Se	elect boundaries	on						
Office	2400 Gine ee		d	isk using marker	s G	Demonstration					
Uffset, sector: 39645	2498 Size, se	ctors: 112953/4/	_			emove Damaged Disk					
RAID Options											
	0		See. Ctab	uter [Defeult] _]	Cantana any Diadu						
RAID Type: RAID 5 (i	RAID Type: RAID 5 (left asynchronous) - Default V Block Size: 512 bytes [Default] V Sectors per Block: 1										
Device may be use	d instead of physica	s) - Detault select	at least thr rice.	ee physical devices	and provide Block	Size, The <i>Damaged</i>					

Page Option Offset

Address of selected area on current disk;

Size, sectors

Size of selected area on current disk;

Number of Tracks per Cylinder

Number of tracks in each cylinder on all platters making up a hard disk. For example, if a hard disk has four platters, each with 600 tracks, then there will be 600 cylinders, and each cylinder will consist of 8 tracks (assuming that each platter has tracks on both sides).

Number of Sectors per Track

A Sector is the smallest unit that can be accessed on a disk. The tracks are concentric circles around the disk and the sectors are segments within each circle. This value indicates how many sectors are on each track.

Stripe Block size

The Size of a block in kilobytes used for RAID creation. Applicable to RAID-0 and RAID-5 arrays. Standard values are 32Kb, 64Kb, 128Kb, 256Kb. If you are not sure - try all standard sizes consecutively and you will most likely find the proper one.

Arrange disks in the Virtual Disk Array using the **Up** and **Down** buttons. If you do not know the particular disk order, try all possible configurations: write down the current order, assemble the array and check the data in it. If the data is not accessible - try a different order until one works.

Some RAID types (Span, RAID5) require a certain stripe block size, thus you will need to specify it in Options box. If you are not sure of this value, you may try to find it in the Controller's configuration utility (Controller's BIOS), or you can try different block sizes and check the results. The most commonly used values are: 32kb, 64kb, 128kb.

4. Confirmation

Review and confirm parameters for the Virtual Disk Array to be created

Click the **Create** button to create the Virtual Disk Array.

5. Complete

Click the **Finish** button to close the Wizard if the RAID was reconstructed successfully, otherwise you will see error messages.

A New Data Storage Device and one or several drives (if detected) will appear in the list of devices and drives in the **Recovery Explorer.**

You can work with reconstructed RAID sets the same way as you work with a regular storage device or logical drive, i.e. scan device for deleted/damaged partitions, scan drives and search for files, recover/ copy files and folders to another safe location, etc...

Active@ UNDELETE Disk Image Wizards Overview

Active@ UNDELETE Disk Image Wizards are sets of step-by-step guided tools to to create, store and load Disk Images.

There are three Disk Image Wizards available:

Disk image Wizards

- Create a Disk Image Wizard on page 87
- Verify a Disk Image Wizard on page 90
- Open a Disk Image Wizard on page 88

Create a Disk Image Wizard

This wizard guides you through simple steps to create a Disk Image of a data storage device or a logical drive.

A Disk Image is a single file or a series of files that stores all the data from your logical drive or physical device as a mirror image. Having a Disk Image can be useful when you want to back up the contents of the whole drive, and restore it or work with it later.

When the Create Disk Image Wizard starts for the first time, the first screen describes the process. Clear the **Show this dialog next time?** check box to avoid seeing this screen the next time you create a Disk Image.

To start the Create Disk Image Wizard, do one of the following:

- From the Wizards menu, click Create Disk Image
- Select Disk Image tab in the Command Bar and click Create Disk Image
- 1. Select imaging area

Select a data storage device in the hierarchical device/partition tree and select the desired device area if necessary.



Note: By clicking on a partition item in the device map, control of the entire partition area will be selected.

2. Set Disk Image attributes

Destination:	D:\temp drive_L_FA23-5895.dim
Description:	Disk Image made by Active@ UNDELETE - Data Recovery Toolbox
ompression Options	Medium Store Disk Image as chunks: 4.7 GB DVD-5

Page Options Destination Path

The full path for the single Disk Image file. If you decide to store the Disk Image file in chunks, this path will be used to store all files. You have the option to use the default path, enter a new path or click **Browse** and navigate to the folder that will store the Disk Image.

Description

Enter a detailed description of the Disk Image you are about to create.

Compression

Choose one of the following:

- None [Raw Data] No compression is applied, sectors are stored in raw format.
- **Fast** Sectors are compressed before storing to the file using a fast compression algorithm.
- **Medium** Sectors are compressed before storing to the file using a slow but more effective compression algorithm.
- High High level of compression;
- Highest Highest possible compression level ill be used;

Store Disk Image as chunks:

Select this check box to save the Disk Image as a series of files with a specified size. Choose the file size from the drop-down list. This option may be useful if you want to write the Disk Image to CD-ROMs or DVD-ROMs. By default this check box is cleared and the Disk Image is stored in one large file;

Ignore R/W Errors

Any Read or Write errors will be ignored and process will continue if possible;

Use Disk Lock

Source disk will be locked until Disk Image creation is complete or aborted;

Ignore Disk Lock Errors

Any errors related to disk lock will be ignored;

3. Confirm actions

Review and confirm the disk image parameters and click the Create Disk Image button to start the disk image creation process. While the process is in progress, you can stop it at any time by clicking Stop at the bottom of the screen.

4. Complete

Click Finish to close the wizard when the disk image creation is complete.

You can work with a disk image in the same way as you work with a regular storage device or logical drive. You can:

- Scan it as a device for deleted or damaged partitions.
- Scan logical drives and search for files.
- Recover or copy files and folders to another safe location.

Open a Disk Image Wizard

This Wizard will guide you via simple steps to open a *Disk Image* that was previously created. You can open a Disk Image based on a configuration file or compose a Disk Image from raw chunks. These chunks may be created by third party software. After a Disk Image is opened you are able to work with it as you would work with a regular Logical Drive or Data Storage Device. You can scan an opened Disk Image, view its contents, and recover files and folders from the Disk Image.

To start the Open Disk Image Wizard - run the **Open Disk Image** menu command from the **Wizards** menu, or click the **Open Disk Image** button on the **Disk Image Tab** Command Bar on the left side.

1. Open Disk Image configuration file

A *Disk Image Configuration File* is a file, used to store all information about a created Disk Image including disk geometry and annotation labels. A Disk Image configuration File is created during the Create Disk Image procedure. You can select a Disk Image to be opened by specifying its Disk Image

Configuration File. Type the full path to this file in the edit box or use the **Browse** button to open a standard browse dialog to select this file.

You can skip this step in order to assemble a Disk Image manually from chunks supplying all necessary options yourself by clicking **Next** button

2. Compose Disk Image

Skip this step if disk image was opened using a configuration file (information is already entered), otherwise specify all parameters here manually.

Typically, a Disk Image Configuration File is used to open a Disk Image. This file contains necessary information about the Disk Image geometry, labels and other information. Nevertheless, a Disk Image can be open by specifying actual files (chunks) of an image and other options. This dialog can be also used to open raw Disk Images created by third party applications (such as WinHex for example)

Caption (Display Name): Disk Image	
Image chunks:	
D:\Temp\di\drive_G_3A3C-7C75.001	Add
	Remove
	Move Up
	Move Down
Disk Image Options	
Image Type: RAW Data (Binary) Disk Image Media Type: Fixed Disk	•
Bytes per Sector: 512 Sector per Track: 63 Track per Cylinder: 255	
Save D	IM File As

Page Options Caption

Enter any label to distinguish newly opened disk image among other devices and disks.

Disk Image Chunks

A Disk Image consists of one or many files, which contains actual image data. A Disk Image can be cut into several files (chunks) during creation for better space allocation. In this list you have to specify all these files which make ups the image. To Add a Disk Image chunk to the list click the **Add New** button and use browse for a file dialog to select a file. To Remove a Disk Image chunk, select this chunk in the list and click the **Remove** button. To modify the order of Disk Image chunks, select any chunk you wish to relocate and use the **Up** and **Down** buttons to move a selected chunk in the chunk stack.

Image Type

Select image type you about to open. Usually it assigned automatically, depending on Disk Image chunks added.

- Raw Disk Image Raw fragment of a disk;
- LSoft Disk Image Disk Image created by any LSoft Technology product;
- Virtual PC Disk Images from Virtual PC software;
- VMWare Image Disk Images from VMWare software;

Media Type

Select appropriate media type. Usually it assign automatically. Use **Fixed Disk** by default.

Bytes per Sector

Enter sector size in bytes;

Sectors per Track

Enter track size in sectors;

Tracks per Cylinder

Enter cylinder size in tracks;

Save DIM File as...

In case of manual composition of Disk Image properties you may save final configuration file for later use;

Click **Next** to continue.

3. Confirmation

Verify and confirm parameters for the disk image to be opened.

Click **Open Disk Image** to read the Disk Image structure and open the Disk Image.

4. Complete

Click **Finish** to close the Wizard.

A new storage device and one or several drives (if detected) will appear in the list of devices and drives in the **Recovery Explorer.**

You can work with an opened Disk Image the same way as you work with a regular storage device or logical drive, i.e. scan device for deleted/damaged partitions, scan drives and search for files, recover/ copy files and folders to another safe location, etc..

Verify a Disk Image Wizard

Disk image validation insures that a data storage disk image or a logical drive disk image is consistent internally and can be opened. We advise you to use this wizard to validate disk images created by third party applications.

To start the Verify Disk Image wizard:

Run the Verify Disk Image menu command from the Tools menu, or click the Validate Disk Image button on the disk image tab in the command bar on the left side.

When the Restore Partition wizard starts for the first time, the first screen describes the process. Clear the "*Show this page next time?*" check box to avoid seeing this screen the next time you run this wizard.

1. Open Disk Image configuration file

A Disk Image configuration File is a file, used to store all information about a created Disk Image including disk geometry and annotation labels. A Disk Image configuration File is created during the Create Disk Image procedure. You can select a Disk Image to be opened by specifying its Disk Image Configuration File. Type in the full path to this file in the edit box or use the browse button to open a 'Browse for file'' dialog and to select this file.

You can skip this step in order to assemble a Disk Image manually from chunks supplying all necessary options yourself.

2. Compose Disk Image

Skip this step if disk image was opened using a configuration file (information is already entered), otherwise specify all parameters here manually.

Typically, a Disk Image Configuration File is used to open a Disk Image. This file contains necessary information about the Disk Image geometry, labels and other information. Nevertheless, a Disk Image can be open by specifying actual files (chunks) of an image and other options. This dialog can be also used to open raw Disk Images created by third party applications (such as WinHex for example)

Caption (Display Nam	ne): Disk Image		
Image chunks:			
D:\Temp\di\drive_	_G_3A3C-7C75.001		Add
			Remove
			Move Up
			Move Down
Disk Image Options	ŝ		
Image Type: RA	AW Data (Binary) Disk Image	 Media Type: Fixed Disk 	•
Bytes per Sector:	512 Sector per Track:	63 Track per Cylinder: 255	
		Save	DIM File As

Page Options Caption

Enter any label to distinguish newly opened disk image among other devices and disks.

Disk Image Chunks

A Disk Image consists of one or many files, which contains actual image data. A Disk Image can be cut into several files (chunks) during creation for better space allocation. In this list you have to specify all these files which make ups the image. To Add a Disk Image chunk to the list click the **Add New** button and use browse for a file dialog to select a file. To Remove a Disk Image chunk, select this chunk in the list and click the **Remove** button. To modify the order of Disk Image chunks, select any chunk you wish to relocate and use the **Up** and **Down** buttons to move a selected chunk in the chunk stack.

Image Type

Select image type you about to open. Usually it assigned automatically, depending on Disk Image chunks added.

- Raw Disk Image Raw fragment of a disk;
- LSoft Disk Image Disk Image created by any LSoft Technology product;
- Virtual PC Disk Images from Virtual PC software;
- VMWare Image Disk Images from VMWare software;

Media Type

Select appropriate media type. Usually it assign automatically. Use **Fixed Disk** by default.

Bytes per Sector

Enter sector size in bytes;

Sectors per Track

Enter track size in sectors;

Tracks per Cylinder

Enter cylinder size in tracks;

Save DIM File as...

In case of manual composition of Disk Image properties you may save final configuration file for later use;

Click **Next** to continue.

3. Confirmation

Verify and confirm parameters for the disk image to be opened.

Click Verify Disk Image to read the Disk Image structure and initiaite verification process.

4. Complete

When verification is completed you will see verification report indicating current integrity of your Disk Image.

Click **Finish** to close the Wizard.

Data Recovery Concept Overview

To understand underlying mechanisms of data storage, organization and recovery, the following topics can give essential concepts:

Understanding Hardware and Disk Organization	Basic information about Hard Disk Drives (HDD) and low-level disk organization.
Understanding File System (FAT)	The FAT file system is a simple file system originally designed for small disks and simple folder structures. The FAT file system is named for its method of organization, the File Allocation Table, which resides at the beginning of the volume. To protect the volume, two copies of the table are kept, in case one becomes damaged. In addition, the file allocation tables and the root folder must be stored in a fixed location so that the files needed to start the system can be correctly located.
Understanding File System (NTFS)	The Windows NT file system (NTFS) provides a combination of performance, reliability, and compatibility not found in the FAT file system. It is designed to quickly perform standard file operations such as read, write, and search — and even advanced operations such as file-system recovery — on very large hard disks.
Understanding Recovery Process	Describes basic approaches and techniques of File and Folder recovery process.
Understanding Partition Recovery Process	Describes most common partition failures and techniques of their recovery.

Hardware and Disk Architecture

Hardware and Disk Organization

Here you can get some information about Hard Disk Drives (HDD) and low-level disk organization:

- Hard Disk Drive Basics on page 94
- Master Boot Record (MBR) on page 96
- Partition Table on page 97

Hard Disk Drive Basics

A hard disk is a sealed unit containing a number of *platters* in a stack. Hard disks may be mounted in a horizontal or a vertical position. In this description, the hard drive is mounted horizontally. Electromagnetic read/write *heads* are positioned above and below each platter. As the platters spin, the drive heads move in toward the center surface and out toward the edge. In this way, the drive heads can reach the entire surface of each platter.

Each disk consists of platters, rings on each side of each platter called tracks, and sections within each track called sectors. A sector is the smallest physical storage unit on a disk, almost always 512 bytes in size.

Figure below illustrates a hard disk with two platters. The remainder of this section describes the terms used on the figure.



Figure 17: Two plated hard disk

The cylinder/head/sector notation scheme described in this section is slowly being eliminated. All new disks use some kind of translation factor to make their actual hardware layout appear as something else, mostly to work with MS-DOS and Windows 95.

Tracks and Cylinders

On hard disks, the data are stored on the disk in thin, concentric bands called *tracks*. There can be more than a thousand tracks on a 3½ inch hard disk. Tracks are a logical rather than physical structure, and are established when the disk is low-level formatted. Track numbers start at 0, and track 0 is the outermost track of the disk. The highest numbered track is next to the spindle. If the disk geometry is being translated, the highest numbered track would typically be 1023. Next figure shows track 0, a track in the middle of the disk, and track 1023.

A *cylinder* consists of the set of tracks that are at the same head position on the disk. In a figure below, cylinder 0 is the four tracks at the outermost edge of the sides of the platters. If the disk has 1024 cylinders (which would be numbered 0-1023), cylinder 1023 consists of all of the tracks at the innermost edge of each side.

Most disks used in personal computers today rotate at a constant angular velocity. The tracks near the outside of the disk are less densely populated with data than the tracks near the center of the disk. Thus, a fixed amount of data can be read in a constant period of time, even though the speed of the disk surface is faster on the tracks located further away from the center of the disk.

Modern disks reserve one side of one platter for track positioning information, which is written to the disk at the factory during disk assembly. It is not available to the operating system. The disk controller uses this information to fine tune the head locations when the heads move to another location on the disk. When a side contains the track position information, that side cannot be used for data. Thus, a disk assembly containing two platters has three sides that are available for data.

Sectors and Clusters

Each track is divided into sections called *sectors*. A sector is the smallest physical storage unit on the disk. The data size of a sector is always a power of two, and is almost always 512 bytes.

Each track has the same number of sectors, which means that the sectors are packed much closer together on tracks near the center of the disk. Next figure shows sectors on a track. You can see that sectors closer to the spindle are closer together than those on the outside edge of the disk. The disk controller uses the sector identification information stored in the area immediately before the data in the sector to determine where the sector itself begins.



Figure 18: Clusters and sectors

As a file is written to the disk, the file system allocates the appropriate number of *clusters* to store the file's data. For example, if each cluster is 512 bytes and the file is 800 bytes, two clusters are allocated for the file. Later, if you update the file to, for example, twice its size (1600 bytes), another two clusters are allocated.

If contiguous clusters (clusters that are next to each other on the disk) are not available, the data are written elsewhere on the disk, and the file is considered to be *fragmented*. Fragmentation is a problem when the file system must search several different locations to find all the pieces of the file you want to read. The search causes a delay before the file is retrieved. A larger cluster size reduces the potential for fragmentation, but increases the likelihood that clusters will have unused space.

Using clusters larger than one sector reduces fragmentation, and reduces the amount of disk space needed to store the information about the used and unused areas on the disk.

The stack of platters rotate at a constant speed. The drive head, while positioned close to the center of the disk reads from a surface that is passing by more slowly than the surface at the outer edges of the disk. To compensate for this physical difference, tracks near the outside of the disk are less-densely populated with data than the tracks near the center of the disk. The result of the different data density is that the same amount of data can be read over the same period of time, from any drive head position.

The disk space is filled with data according to a standard plan. One side of one platter contains space reserved for hardware track-positioning information and is not available to the operating system. Thus, a disk assembly containing two platters has three sides available for data. Track-positioning data is written to the disk during assembly at the factory. The system *disk controller* reads this data to place the drive heads in the correct sector position.

Master Boot Record (MBR)

The Master Boot Record, created when you create the first partition on the hard disk, is probably the most important data structure on the disk. It is the first sector on every disk. The location is always track (cylinder) 0, side (head) 0, and sector 1.

The Master Boot Record contains the *Partition Table* on page 97 for the disk and a small amount of executable code. On x86-based computers, the executable code examines the Partition Table, and identifies the system partition. The Master Boot Record then finds the system partition's starting location on the disk, and loads an copy of its Partition Boot Sector into memory. The Master Boot Record then transfers execution to executable code in the Partition Boot Sector.



Note:

Although there is a Master Boot Record on every hard disk, the executable code in the sector is used only if the disk is connected to an x86-based computer and the disk contains the system partition.

Figure below shows a hex dump of the sector containing the Master Boot Record. The figure shows the sector in two parts. The first part is the Master Boot Record, which occupies the first 446 bytes of the sector. The disk signature (FD 4E F2 14) is at the end of the Master Boot Record code. The second part is the *Partition Table* on page 97.

Physical Sector: Cyl 0, Side 0, Sector 1

00000000:	00	33	C0	8E	D0	BC	00	7C	-	8B	F4	50	07	50	1F	FB	FC		
00000010:	BF	00	06	В9	00	01	F2	A5	-	ΕA	1D	06	00	00	ΒE	ΒE	07		
00000020:	B3	04	80	3C	80	74	0E	80	-	3C	00	75	1C	83	C6	10	FE		
0000030:	CB	75	EF	CD	18	8B	14	8B	-	4C	02	8B	ΕE	83	C6	10	FE		
00000040:	CB	74	1A	80	3C	00	74	F4	-	ΒE	8B	06	AC	3C	00	74	0B		
.t<.t 00000050:	56	.< BB	.t. 07	00	В4	0E	CD	10	-	5E	ΕB	FO	ΕB	FE	BF	05	00		
V 00000060:	^ BB	00	7C	В8	01	02	57	CD	-	13	5F	73	0C	33	C0	CD	13		
W 00000070:	••s 4F	3.3. 75	ED	ΒE	A3	06	ΕB	D3	-	ΒE	C2	06	BF	FE	7D	81	3D		
Ou 00000080:	55	.}. AA	.= 75	C7	8B	F5	EA	00	-	7C	00	00	49	6E	76	61	6C		
U.u 00000090:	I 69	nva 64	al 20	70	61	72	74	69	_	74	69	6F	6E	20	74	61	62	id par	rtition
tab 000000A0:	6C	65	00	45	72	72	6F	72	-	20	6C	6F	61	64	69	6E	67	le.Er:	ror
loading 000000B0:	20	6F	70	65	72	61	74	69	-	6E	67	20	73	79	73	74	65	opera	ating
syste 000000C0:	6D	00	4D	69	73	73	69	6E	-	67	20	6F	70	65	72	61	74	m.Miss	sing
000000D0:	69	6E	67	20	73	79	73	74	-	65	6D	00	00	80	45	14	15	ing	
system 000000E0:	.E 00	00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00		
000000F0:	00	00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00		
00000100:	00	00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00		
00000110:	00	00	00	00	00	00	00	00	_	00	00	00	00	00	00	00	00		
00000120:	00	00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00		
		•••	•••																

00000130:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
00000140:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
00000150:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
00000160:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
00000170:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
00000180:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
00000190:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
000001A0:	00 00	00	00	00	00	00	00	-	00	00	00	00	00	00	00	00
000001B0:	00 00	00	00	00	00	00	00	-	FD	4E	F2	14	00	00	80	01
000001C0:	01 00	06	0F	7F	96	ЗF	00	-	00	00	51	42	06	00	00	00
#.? 000001D0:	QB. 41 97	07	0F	FF	2C	90	42	-	06	00	AO	3E	06	00	00	00
A,.B 000001E0:	> C1 2D	05	0 F	FF	92	30	81	-	0C	00	AO	91	01	00	00	00
0 000001F0:	C1 93	01	0F	FF	A6	DO	12	-	0E	00	C0	4E	00	00	55	AA
	N.	.U.														



Important: Viruses Can Infect the Master Boot Record

Many destructive viruses damage the Master Boot Record and make it impossible to start the computer from the hard disk. Because the code in the Master Boot Record executes before any operating system is started, no operating system can detect or recover from corruption of the Master Boot Record. You can use, for example, the DiskProbe program on *Windows NT Workstation Resource Kit* CD to display the Master Boot Record, and compare it to the Master Boot Record shown above. There are also utilities on the Microsoft Windows Resource Kits that enable you to save and restore the Master Boot Record.



Tip:

For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com/

Partition Table

The information about primary partitions and an extended partition is contained in the Partition Table, a 64-byte data structure located in the same sector as the *Master Boot Record (MBR)* on page 96 (cylinder 0, head 0, sector 1). The Partition Table conforms to a standard layout that is independent of the operating system. Each Partition Table entry is 16 bytes long, making a maximum of four entries available. Each entry starts at a predetermined offset from the beginning of the sector, as follows:

- Partition 1 0x01BE (446)
- Partition 2 0x01CE (462)
- Partition 3 0x01DE (478)
- Partition 4 0x01EE (494)

The last two bytes in the sector are a signature word for the sector and are always 0x55AA.

The next figure is a printout of the Partition Table for the disk shown in a *Master Boot Record (MBR)* on page 96 earlier in this chapter. When there are fewer than four partitions, the remaining fields are all zeros.

																80	01	••
000001C0:	01	00	06	0 F	7F	96	ЗF	00	-	00	00	51	42	06	00	00	00	
#.?	••••	2Β																
000001D0:	41	97	07	ΟF	FF	2C	90	42	-	06	00	AO	ЗE	06	00	00	00	A,.B>
000001E0:	C1	2D	05	ΟF	FF	92	30	81	-	0C	00	AO	91	01	00	00	00	. – 0
000001F0:	C1	93	01	ΟF	FF	A6	DO	12	-	0E	00	C0	$4 \mathrm{E}$	00	00	55	AA	NU.

The following table describes each entry in the Partition Table. The sample values correspond to the information for partition 1.

Table 1: Partition Table Fields

Byte Offset	Field Length	Sample Value	Meaning
00	BYTE	0x80	Boot Indicator. Indicates whether the partition is the system partition. Legal values are: 00 = Do not use for booting. 80 = System partition.
01	BYTE	0x01	Starting Head.
02	6 bits	0x01	Starting Sector. Only bits 0-5 are used. Bits 6-7 are the upper two bits for the Starting Cylinder field.
03	10 bits	0x00	Starting Cylinder. This field contains the lower 8 bits of the cylinder value. Starting cylinder is thus a 10-bit number, with a maximum value of 1023.
04	BYTE	0x06	System ID. This byte defines the volume type. In Windows NT, it also indicates that a partition is part of a volume that requires the use of the HKEY_LOCAL_MACHIN \SYSTEM\DISK Registry subkey.
05	BYTE	0x0F	Ending Head.
06	6 bits	0x3F	Ending Sector. Only bits 0-5 are used. Bits 6-7 are the upper two bits for

Byte Offset	Field Length	Sample Value	Meaning
			the Ending Cylinder field.
07	10 bits	0x196	Ending Cylinder. This field contains the lower 8 bits of the cylinder value. Ending cylinder is thus a 10-bit number, with a maximum value of 1023.
08	DWORD	3F 00 00 00	Relative Sector.
12	DWORD	51 42 06 00	Total Sectors.

The remainder of this section describes the uses of these fields. Definitions of the fields in the Partition Table is the same for primary partitions, extended partitions, and logical drives in extended partitions.

Boot Indicator Field

The Boot Indicator field indicates whether the volume is the system partition. On x-86-based computers, only one primary partition on the disk should have this field set. This field is used only on x86-based computers. On RISC-based computers, the NVRAM contains the information for finding the files to load.

On x86-based computers, it is possible to have different operating systems and different file systems on different volumes. For example, a computer could have MS-DOS on the first primary partition and Windows 95, UNIX, OS/2, or Windows NT on the second. You control which primary partition (active partition in FDISK) to use to start the computer by setting the Boot Indicator field for that partition in the Partition Table.

System ID Field

For primary partitions and logical drives, the System ID field describes the file system used to format the volume. Windows NT uses this field to determine what file system device drivers to load during startup. It also identifies the extended partition, if there is one defined.

Value	Meaning
0x01	12-bit FAT primary partition or logical drive. The number of sectors in the volume is fewer than 32680.
0x04	16-bit FAT primary partition or logical drive. The number of sectors is between 32680 and 65535.
0x05	Extended partition. See section titled "Logical Drives and Extended Partitions," presented later in this chapter, for more information.
0x06	BIGDOS FAT primary partition or logical drive.
0x07	NTFS primary partition or logical drive.

Table 2: System ID field description

Figure presented earlier in this section, has examples of a BIGDOS FAT partition, an NTFS partition, an extended partition, and a 12-bit FAT partition.

If you install Windows NT on a computer that has Windows 95 preinstalled, the FAT partitions might be shown as unknown. If you want to be able to use these partitions when running Windows NT, your only option is to delete the partitions.

OEM versions of Windows 95 support the following four partition types for FAT file systems that Windows NT cannot recognize.

Value	Meaning
0x0B	Primary Fat32 partition, using interrupt 13 (INT 13) extensions.
0x0C	Extended Fat32 partition, using INT 13 extensions.
0x0E	Extended Fat16 partition, using INT 13 extensions.
0x0F	Primary Fat16 partition, using INT 13 extensions.

When you create a volume set or a stripe set, Disk Administrator sets the high bit of the System ID field for each primary partition or logical drive that is a member of the volume. For example, a FAT primary partition or logical drive that is a member of a volume set or a stripe set has a System ID value of 0x86. An NTFS primary partition or logical drive has a System ID value of 0x87. This bit indicates that Windows NT needs to use the HKEY_LOCAL_MACHINE\SYSTEM\DISK Registry subkey to determine how the members of the volume set or stripe set relate to each other. Volumes that have the high bit set can only be accessed by Windows NT.

When a primary partition or logical drive that is a member of a volume set or a stripe set has failed due to write errors or cannot be accessed, the second most significant bit is set. The System ID byte is set to C6 in the case of a FAT volume, or C7 in the case of an NTFS volume.

Note:

If you start up MS-DOS, it can only access primary partitions or logical drives that have a value of 0x01, 0x04, 0x05, or 0x06 for the System ID. However, you should be able to delete volumes that have the other values. If you use a MS-DOS-based low-level disk editor, you can read and write any sector, including ones that are in NTFS volumes.

On Windows NT Server, mirror sets and stripe sets with parity also require the use of the Registry subkey HKEY_LOCAL_MACHINE\SYSTEM\DISK to determine how to access the disks.

Starting and Ending Head, Sector, and Cylinder Fields

On x86-based computers, the Starting and Ending Head, Cylinder, and Sector fields on the start-up disk are very important for starting up the computer. The code in the Master Boot Record uses these fields to find and load the Partition Boot Sector.

The Ending Cylinder field in the Partition Table is ten bits long, which limits the maximum number of cylinders that can be described in the Partition Table to 1024. The Starting and Ending Head fields are one byte long, which limits this field to the range 0 - 255. The Starting and Ending Sector field is 6 bits long, limiting its range to 0 - 63. However, sectors start counting at 1 (versus 0 for the other fields), so the maximum number of sectors per track is 63.

Since current hard disks are low-level formatted with the industry standard 512-byte sector size, the maximum capacity disk that can be described by the Partition Table can be calculated as follows:

Substituting the maximum possible values yields:

512 x 63 x 1024 x 256 = 8,455,716,864 bytes or 7.8 GB

The maximum formatted capacity is slightly less than 8 GB.

However, the maximum cluster size that you can use for FAT volumes when running Windows NT is 64K, when using a 512 byte sector size. Therefore, the maximum size for a FAT volume is 4 GB.

If you have a dual-boot configuration with Windows 95 or MS-DOS, FAT volumes that might be accessed when using either of those operating systems are limited to 2 GB. In addition, Macintosh computers that are viewing volumes on a computer running Windows NT cannot see more than 2 GB. If you try to use a FAT volume larger than 2 GB when running MS-DOS or Windows 95, or access it from a Macintosh computer, you might get a message that there are 0 bytes available. The same limit applies to OS/2 system and boot partitions.

The maximum size of a FAT volume on a specific computer depends on the disk geometry, and the maximum values that can fit in the fields described in this section. The next table shows the typical size of a FAT volume when translation is enabled, and when it is disabled. The number of cylinders in both situations is 1024.

Translation mode	Number of heads	Sectors per track	Maximum size for system or boot partition
Disabled	64	32	1 GB
Enabled	255	63	4 GB



Note:

RISC-based computers do not have a limit on the size of the system or boot partitions.

If a primary partition or logical drive extends beyond cylinder 1023, all of these fields will contain the maximum values.

Relative Sectors and Number of Sectors Fields

For primary partitions, the Relative Sectors field represents the offset from the beginning of the disk to the beginning of the partition, counting by sectors. The Number of Sectors field represents the total number of sectors in the partition. For a description of these fields in extended partitions, see the section *Logical Drives and Extended Partitions*.

Windows NT uses these fields to access all partitions. When you format a partition when running Windows NT, it puts data into the Starting and Ending Cylinder, Head, and Sector fields only for backward compatibility with MS-DOS and Windows 95, and to maintain compatibility with the BIOS interrupt (INT) 13 for start-up purposes.

Logical Drives and Extended Partitions

When more than four logical disks are required on a single physical disk, the first partition should be a primary partition. The second partition can be created as an extended partition, which can contain all the remaining unpartitioned space on the disk.



Note:

A primary partition is one that can be used as the system partition. If the disk does not contain a system partition, you can configure the entire disk as a single, extended partition.

Some computers create an EISA configuration partition as the first partition on the hard disk.

Windows NT detects an extended partition because the System ID byte in the Partition Table entry is set to 5. There can be only one extended partition on a hard disk.

Within the extended partition, you can create any number of logical drives. As a practical matter, the number of available drive letters is the limiting factor in the number of logical drives that you can define.

When you have an extended partition on the hard disk, the entry for that partition in the Partition Table (at the end of the Master Boot Record) points to the first disk sector in the extended partition. The first sector of each logical drive in an extended partition also has a Partition Table, which is the last 66 bytes of the sector. (The last two bytes of the sector are the end-of-sector marker.)

These are the entries in an extended Partition Table:

- The first entry is for the current logical drive.
- The second entry contains information about the next logical drive in the extended partition.
- Entries three and four are all zeroes.

This format repeats for every logical drive. The last logical drive has only its own partition entry listed. The entries for partitions 2-4 are all zeroes.

The Partition Table entry is the only information on the first side of the first cylinder of each logical drive in the extended partition. The entry for partition 1 in each Partition Table contains the starting address for data on the current logical drive. And the entry for partition 2 is the address of the sector that contains the Partition Table for the next logical drive.

The use of the Relative Sector and Total Sectors fields for logical drives in an extended partition is different than for primary partitions. For the partition 1 entry of each logical drive, the Relative Sectors field is the sector from the beginning of the logical drive that contains the Partition Boot Sector. The Total Sectors field is the number of sectors from the Partition Boot Sector to the end of the logical drive.

For the partition 2 entry, the Relative Sectors field is the offset from the beginning of the extended partition to the sector containing the Partition Table for the logical drive defined in the Partition 2 entry. The Total Sectors field is the total size of the logical drive defined in the Partition 2 entry.



If a logical drive is part of a volume set, the Partition Boot Sector is at the beginning of the first member of the volume set. Other members of the volume set have data where the Partition Boot Sector would normally be located.



Tip:

For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com/

Disk Arrays (RAID's)

Redundant array of independent disks (RAID) is a storage technology that combines multiple disk drive components into a logical unit. Data is distributed across the drives in one of several ways called "RAID levels", depending on what level of redundancy and performance (via parallel communication) is required.

RAID Types

RAID-0 This technique has striping but no redundancy of data. It offers the best performance but no fault-tolerance.
 RAID-1 This type is also known as disk mirroring and consists of at least two drives that duplicate the storage of data. There is no striping. Read performance is improved since either disk can be read at the same time. Write performance is the same as for single disk storage. RAID-1 provides

	the best performance and the best fault-tolerance in a multi-user system.
RAID-2	This type uses striping across disks with some disks storing error checking and correcting (ECC) information. It has no advantage over RAID-3.
RAID-3	This type uses striping and dedicates one drive to storing parity information. The embedded error checking (ECC) information is used to detect errors. Data recovery is accomplished by calculating the exclusive OR (XOR) of the information recorded on the other drives. Since an I/O operation addresses all drives at the same time, RAID-3 cannot overlap I/O. For this reason, RAID-3 is best for single-user systems with long record applications.
RAID-4	This type uses large stripes, which means you can read records from any single drive. This allows you to take advantage of overlapped I/O for read operations. Since all write operations have to update the parity drive, no I/O overlapping is possible. RAID-4 offers no advantage over RAID-5.
RAID-5	This type includes a rotating parity array, thus addressing the write limitation in RAID-4. Thus, all read and write operations can be overlapped. RAID-5 stores parity information but not redundant data (but parity information can be used to reconstruct data). RAID-5 requires at least three and usually five disks for the array. It's best for multi-user systems in which performance is not critical or which do few write operations.

Parity [·]	Tables
---------------------	--------

Left Synchronous			
0	5	6	Ρ
1	4	Р	11
2	Р	7	10
Р	3	8	9
Left Asynchronous			
0	3	6	Р
1	4	Р	9
2	Р	7	10
Р	5	8	11
Right Synchronous			
Р	5	6	11
0	Ρ	7	10

Right Synchronous			
1	4	Р	9
2	3	8	Р
Right Asynchronous			
Р	3	6	9
0	Р	7	10
1	4	Ρ	11
2	5	8	Ρ

Logical Disk Manager (LDM) Overview

Dynamic disks provide features that basic disks do not, such as the ability to create volumes that span multiple disks (spanned and striped volumes), and the ability to create fault tolerant volumes (mirrored and RAID-5 volumes). All volumes on dynamic disks are known as dynamic volumes.

There are five types of dynamic volumes:

Simnle

Simple	A dynamic volume made up of disk space from a single dynamic disk. A simple volume can consist of a single region on a disk or multiple regions of the same disk that are linked together. If the simple volume is not a system volume or boot volume, you can extend it within the same disk or onto additional disks. If you extend a simple volume across multiple disks, it becomes a spanned volume. You can create simple volumes only on dynamic disks. Simple volumes are not fault tolerant, but you can mirror them to create mirrored volumes on computers running the Windows 2000 Server or Windows Server 2003 families of operating systems.
Spanned	A dynamic volume consisting of disk space on more than one physical disk. You can increase the size of a spanned volume by extending it onto additional dynamic disks. You can create spanned volumes only on dynamic disks. Spanned volumes are not fault tolerant and cannot be mirrored.
Striped	A dynamic volume that stores data in stripes on two or more physical disks. Data in a striped volume is allocated alternately and evenly (in stripes) across the disks. Striped volumes offer the best performance of all the volumes that are available in Windows, but they do not provide fault tolerance. If a disk in a striped volume fails, the data in the entire volume is lost. You can create striped volumes only on dynamic disks. Striped volumes cannot be mirrored or extended.
Mirrored	A fault-tolerant volume that duplicates data on two physical disks. A mirrored volume provides data

redundancy by using two identical volumes, which are called mirrors, to duplicate the information contained on the volume. A mirror is always located on a different disk. If one of the physical disks fails, the data on the failed disk becomes unavailable, but the system continues to operate in the mirror on the remaining disk. You can create mirrored volumes only on dynamic disks on computers running the Windows 2000 Server or Windows Server 2003 families of operating systems. You cannot extend mirrored volumes.

A fault-tolerant volume with data and parity striped intermittently across three or more physical disks. Parity is a calculated value that is used to reconstruct data after a failure. If a portion of a physical disk fails, Windows recreates the data that was on the failed portion from the remaining data and parity. You can create RAID-5 volumes only on dynamic disks on computers running the Windows 2000 Server or Windows Server 2003 families of operating systems. You cannot mirror or extend RAID-5 volumes. In Windows NT 4.0, a RAID-5 volume was known as a striped set with parity.

Mirrored and RAID-5 volumes are fault tolerant and are available only on computers running Windows 2000 Server, Windows 2000 Advanced Server, Windows 2000 Datacenter Server, or the Windows Server 2003 family of operating systems. You can, however, use a computer running Windows XP Professional to remotely create mirrored and RAID-5 volumes on these operating systems.

Regardless of whether the dynamic disk uses the master boot record (MBR) or GUID partition table (GPT) partition style, you can create up to 2,000 dynamic volumes, although the recommended number of dynamic volumes is 32 or less.

For information about how to manage dynamic volumes, see *Manage dynamic volumes*.

RAID-5

File Systems

Windows NT File System (NTFS)

The Windows NT file system (NTFS) provides a combination of performance, reliability, and compatibility not found in the FAT file system. It is designed to quickly perform standard file operations such as read, write, and search — and even advanced operations such as file-system recovery — on very large hard disks.

Formatting a volume with the NTFS file system results in the creation of several system files and the Master File Table (MFT), which contains information about all the files and folders on the NTFS volume.

The first information on an NTFS volume is the Partition Boot Sector, which starts at sector 0 and can be up to 16 sectors long. The first file on an NTFS volume is the Master File Table (MFT).

partition boot Master File Table sector	system files	file area
---	-----------------	-----------

Figure 19: Layout of NTFS volume after formatting

See the next sections for more information about NTFS:

- NTFS Partition Boot Sector on page 107
- NTFS Master File Table (MFT) on page 109
- NTFS File Types on page 110
- Data Integrity and Recoverability with NTFS on page 116

The NTFS file system includes security features required for file servers and high-end personal computers in a corporate environment. The NTFS file system also supports data access control and ownership privileges that are important for the integrity of critical data. While folders shared on a Windows NT computer are assigned particular permissions, NTFS files and folders can have permissions assigned whether they are shared or not. NTFS is the only file system on Windows NT that allows you to assign permissions to individual files.

The NTFS file system has a simple, yet very powerful design. Basically, everything on the volume is a file and everything in a file is an attribute, from the data attribute, to the security attribute, to the file name attribute. Every sector on an NTFS volume that is allocated belongs to some file. Even the file system metadata (information that describes the file system itself) is part of a file.

Encryption	The Encrypting File System (EFS) provides the core file encryption technology used to store encrypted files on NTFS volumes. EFS keeps files safe from intruders who might gain unauthorized physical access to sensitive, stored data (for example, by stealing a portable computer or external disk drive).
Disk quotas	Windows 2000 supports disk quotas for NTFS volumes. You can use disk quotas to monitor and limit disk-space use.
Reparse points	Reparse points are new file system objects in NTFS that can be applied to NTFS files or folders. A file or folder that contains a reparse point acquires additional behaviour not present in the underlying file system. Reparse points are used by many of the

What's New in NTFS5 (Windows 2000)

	new storage features in Windows 2000, including volume mount points.
Volume mount points	Volume mount points are new to NTFS. Based on reparse points, volume mount points allow administrators to graft access to the root of one local volume onto the folder structure of another local volume.
Sparse files	Sparse files allow programs to create very large files but consume disk space only as needed.
Distributed link tracking	NTFS provides a link-tracking service that maintains the integrity of shortcuts to files as well as OLE links within compound documents.



Tip:

For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://msdn.microsoft.com

NTFS Partition Boot Sector

Next table describes the boot sector of a volume formatted with NTFS. When you format an NTFS volume, the format program allocates the first 16 sectors for the boot sector and the bootstrap code.

Byte Offset	Field Length	Field Name
0x00	3 bytes	Jump Instruction
0x03	LONGLONG	OEM ID
0x0B	25 bytes	BPB
0x24	48 bytes	Extended BPB
0x54	426 bytes	Bootstrap Code
0x01FE	WORD	End of Sector Marker

On NTFS volumes, the data fields that follow the BPB form an extended BPB. The data in these fields enables Ntldr (NT loader program) to find the master file table (MFT) during startup. On NTFS volumes, the MFT is not located in a predefined sector, as on FAT16 and FAT32 volumes. For this reason, the MFT can be moved if there is a bad sector in its normal location. However, if the data is corrupted, the MFT cannot be located, and Windows NT/2000 assumes that the volume has not been formatted.

The following example illustrates the boot sector of an NTFS volume formatted while running Windows 2000. The printout is formatted in three sections:

- Bytes 0x00– 0x0A are the jump instruction and the OEM ID (shown in bold print).
- Bytes 0x0B–0x53 are the BPB and the extended BPB.
- The remaining code is the bootstrap code and the end of sector marker (shown in bold print).

Physical Sector: Cyl 0, Side 1, Sector 1

```
00000000: EB 52 90 4E 54 46 53 20 - 20 20 20 00 02 08 00

00 .R.NTFS .....

00000010: 00 00 00 00 F8 00 00 - 3F 00 FF 00 3F 00 00 00 ....?..?..

00000020: 00 00 00 80 00 80 00 - 4A F5 7F 00 00 00 00

00 .....J....

00000030: 04 00 00 00 00 00 00 00 - 54 FF 07 00 00 00 00

00 .....T....
```

00000040: F6 00 00 00 01 00 00 00 - 14 A5 1B 74 C9 1B 74 1Ct..t. 00000050: 00 00 00 00 FA 33 C0 8E - D0 BC 00 7C FB B8 C0 07....3...... 00000060: 8E D8 E8 16 00 B8 00 0D - 8E C0 33 DB C6 06 0E 00000070: 10 E8 53 00 68 00 0D 68 - 6A 02 CB 8A 16 24 00 B4 ...s.h...hj.... \$.. 00000080: 08 CD 13 73 05 B9 FF FF - 8A F1 66 0F B6 C6 40 66s.....f...@f 00000090: OF B6 D1 80 E2 3F F7 E2 - 86 CD C0 ED 06 41 66 OFAf. 000000A0: B7 C9 66 F7 E1 66 A3 20 - 00 C3 B4 41 BB AA 55 8A ..f..f. ...A..U. 000000B0: 16 24 00 CD 13 72 OF 81 - FB 55 AA 75 09 F6 C1 01 .\$...r...U.u.... 000000c0: 74 04 FE 06 14 00 C3 66 - 60 1E 06 66 A1 10 00 66 t.....f`..f. 000000D0: 03 06 1C 00 66 3B 06 20 - 00 0F 82 3A 00 1E 66 6Af;fj 000000E0: 00 66 50 06 53 66 68 10 - 00 01 00 80 3E 14 00 00 .fP.sfh....>... 00000100: B4 42 8A 16 24 00 16 1F - 8B F4 CD 13 66 58 5B 07 .B..\$....fx[.. 00000110: 66 58 66 58 1F EB 2D 66 - 33 D2 66 0F B7 0E 18 00 fxfx.-f3.f..... 00000120: 66 F7 F1 FE C2 8A CA 66 - 8B D0 66 C1 EA 10 F7 36 f.....f..f....6 00000130: **1A 00 86 D6 8A 16 24 00 - 8A E8 C0 E4 06 0A CC B8**\$.... 00000140: 01 02 CD 13 0F 82 19 00 - 8C CO 05 20 00 8E CO 66 00000160: C3 A0 F8 01 E8 09 00 A0 - FB 01 E8 03 00 FB EB FE 00000170: **B4 01 8B F0 AC 3C 00 74 - 09 B4 0E BB 07 00 CD 10**<.t.... 00000180: EB F2 C3 OD OA 41 20 64 - 69 73 6B 20 72 65 61 64A disk read 00000190: 20 65 72 72 6F 72 20 6F - 63 63 75 72 72 65 64 00 error occurred. 000001A0: OD OA 4E 54 4C 44 52 20 - 69 73 20 6D 69 73 73 69 ..NTLDR is missi 000001B0: 6E 67 00 0D 0A 4E 54 4C - 44 52 20 69 73 20 63 6F ng...NTLDR is co 000001c0: 6D 70 72 65 73 73 65 64 - 00 0D 0A 50 72 65 73 73 mpressed...Press 000001D0: 20 43 74 72 6C 2B 41 6C - 74 2B 44 65 6C 20 74 6F Ctrl+Alt+Del to 000001E0: 20 72 65 73 74 61 72 74 - OD OA 00 00 00 00 00 00 restart.....

The following table describes the fields in the BPB and the extended BPB on NTFS volumes. The fields starting at 0x0B, 0x0D, 0x15, 0x18, 0x1A, and 0x1C match those on FAT16 and FAT32 volumes. The sample values correspond to the data in this example.

Table 3	5: BIOS Parameter	DIUCK dilu	Extended BIO	5 Falalle	er block fields	

Table 2: PIAS Davameter Pleak and Extended PIAS Davameter Pleak Fields

Byte Offset	Field Length	Sample Value	Field Name
0x0B	WORD	0x0002	Bytes Per Sector
0x0D	BYTE	0x08	Sectors Per Cluster
0x0E	WORD	0x0000	Reserved Sectors
0x10	3 BYTES	0x000000	always 0
0x13	WORD	0x0000	not used by NTFS
0x15	BYTE	0xF8	Media Descriptor
0x16	WORD	0x0000	always 0
0x18	WORD	0x3F00	Sectors Per Track
0x1A	WORD	0xFF00	Number Of Heads
0x1C	DWORD	0x3F000000	Hidden Sectors
0x20	DWORD	0x0000000	not used by NTFS
Byte Offset	Field Length	Sample Value	Field Name
-------------	--------------	---------------------	---
0x24	DWORD	0x80008000	not used by NTFS
0x28	LONGLONG	0x4AF57F0000000000	Total Sectors
0x30	LONGLONG	0x04000000000000000	Logical Cluster Number for the file \$MFT
0x38	LONGLONG	0x54FF070000000000	Logical Cluster Number for the file \$MFTMirr
0x40	DWORD	0xF6000000	Clusters Per File Record Segment
0x44	DWORD	0x01000000	Clusters Per Index Block
0x48	LONGLONG	0x14A51B74C91B741C	Volume Serial Number
0x50	DWORD	0x0000000	Checksum

Protecting the Boot Sector

Because a normally functioning system relies on the boot sector to access a volume, it is highly recommended that you run disk scanning tools such as Chkdsk regularly, as well as back up all of your data files to protect against data loss if you lose access to a volume.



Tip:

For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com microsoft Developers Network (MSDN) http://www.microsoft.com

NTFS Master File Table (MFT)

Each file on an NTFS volume is represented by a record in a special file called the master file table (MFT). NTFS reserves the first 16 records of the table for special information. The first record of this table describes the master file table itself, followed by a MFT *mirror record*. If the first MFT record is corrupted, NTFS reads the second record to find the MFT mirror file, whose first record is identical to the first record of the MFT. The locations of the data segments for both the MFT and MFT mirror file are recorded in the boot sector. A duplicate of the boot sector is located at the logical center of the disk.

The third record of the MFT is the log file, used for file recovery. The seventeenth and following records of the master file table are for each file and directory (also viewed as a file by NTFS) on the volume.



Figure 20: Simplified illustration of the MFT structure

The master file table allocates a certain amount of space for each file record. The attributes of a file are written to the allocated space in the MFT. Small files and directories (typically 1500 bytes or smaller), such as the file illustrated in next figure, can entirely be contained within the master file table record.

Standard information	File or directory name	Security descriptor	Data or index		
-------------------------	------------------------------	------------------------	---------------	--	--

Figure 21: MFT Record for a Small File or Directory

This design makes file access very fast. Consider, for example, the FAT file system, which uses a file allocation table to list the names and addresses of each file. FAT directory entries contain an index into the file allocation table. When you want to view a file, FAT first reads the file allocation table and assures that it exists. Then FAT retrieves the file by searching the chain of allocation units assigned to the file. With NTFS, as soon as you look up the file, it's there for you to use.

Directory records are housed within the master file table just like file records. Instead of data, directories contain index information. Small directory records reside entirely within the MFT structure. Large directories are organized into B-trees, having records with pointers to external clusters containing directory entries that could not be contained within the MFT structure.



For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com/

NTFS File Types

- NTFS File Attributes on page 111
- NTFS System Files on page 112
- NTFS Multiple Data Streams on page 113
- NTFS Compressed Files on page 114
- NTFS Encrypted Files (Windows 2000 only) on page 114
- NTFS Sparse Files (Windows 2000 only) on page 115

NTFS File Attributes

The NTFS file system views each file (or folder) as a set of file attributes. Elements such as the file's name, its security information, and even its data, are all file attributes. Each attribute is identified by an attribute type code and, optionally, an attribute name.

When a file's attributes can fit within the MFT file record, they are called resident attributes. For example, information such as filename and time stamp are always included in the MFT file record. When all of the information for a file is too large to fit in the MFT file record, some of its attributes are non-resident. The non-resident attributes are allocated one or more clusters of disk space elsewhere in the volume. NTFS creates the Attribute List attribute to describe the location of all of the attribute records.

Next table lists all of the file attributes currently defined by the NTFS file system. This list is extensible, meaning that other file attributes can be defined in the future.

Attribute Type	Description
Standard Information	Includes information such as timestamp and link count.
Attribute List	Lists the location of all attribute records that do not fit in the MFT record.
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 8.3, case-insensitive name for the file. Additional names, or hard links, required by POSIX can be included as additional file name attributes.
Security Descriptor	Describes who owns the file and who can access it.
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.
Object ID	A volume-unique file identifier. Used by the distributed link tracking service. Not all files have object identifiers.
Logged Tool Stream	Similar to a data stream, but operations are logged to the NTFS log file just like NTFS metadata changes. This is used by EFS.
Reparse Point	Used for volume mount points. They are also used by Installable File System (IFS) filter drivers to mark certain files as special to that driver.
Index Root	Used to implement folders and other indexes.
Index Allocation	Used to implement folders and other indexes.
Bitmap	Used to implement folders and other indexes.
Volume Information	Used only in the \$Volume system file. Contains the volume version.
Volume Name	Used only in the \$Volume system file. Contains the volume label.

NTFS System Files

NTFS includes several system files, all of which are hidden from view on the NTFS volume. A *system file* is one used by the file system to store its metadata and to implement the file system. System files are placed on the volume by the Format utility.

System File	File Name	MFT Record	Purpose of the File
Master file table	\$Mft	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.
Master file table 2	\$MftMirr	1	A duplicate image of the first four records of the MFT. This file guarantees access to the MFT in case of a single-sector failure.
Log file	\$LogFile	2	Contains a list of transaction steps used for NTFS recoverability. Log file size depends on the volume size and can be as large as 4 MB. It is used by Windows NT/2000 to restore consistency to NTFS after a system failure.
Volume	\$Volume	3	Contains information about the volume, such as the volume label and the volume version.
Attribute definitions	\$AttrDef	4	A table of attribute names, numbers, and descriptions.
Root file name index	\$	5	The root folder.
Cluster bitmap	\$Bitmap	6	A representation of the volume showing which clusters are in use.

Table 4: Metadata Stored in the Master File Table

System File	File Name	MFT Record	Purpose of the File
Boot sector	\$Boot	7	Includes the BPB used to mount the volume and additional bootstrap loader code used if the volume is bootable.
Bad cluster file	\$BadClus	8	Contains bad clusters for the volume.
Security file	\$Secure	9	Contains unique security descriptors for all files within a volume.
Upcase table	\$Upcase	10	Converts lowercase characters to matching Unicode uppercase characters.
NTFS extension file	\$Extend	11	Used for various optional extensions such as quotas, reparse point data, and object identifiers.
		12–15	Reserved for future use.

NTFS Multiple Data Streams

NTFS supports multiple data *streams*, where the stream name identifies a new data attribute on the file. A handle can be opened to each data stream. A data stream, then, is a unique set of file attributes. Streams have separate opportunistic locks, file locks, and sizes, but common permissions.

This feature enables you to manage data as a single unit. The following is an example of an alternate stream:

```
myfile.dat:stream2
```

A library of files might exist where the files are defined as alternate streams, as in the following example:

library:file1
 :file2
 :file3

A file can be associated with more than one application at a time, such as Microsoft [®] Word and Microsoft [©] WordPad. For instance, a file structure like the following illustrates file association, but not multiple files:

program:source_file
 :doc_file
 :object_file
 :executable_file

To create an alternate data stream, at the command prompt, you can type commands such as:

echo text>program:source file

more <program:source file

Important:

T

When you copy an NTFS file to a FAT volume, such as a floppy disk, data streams and other attributes not supported by FAT are lost.

NTFS Compressed Files

Windows NT/2000 supports compression on individual files, folders, and entire NTFS volumes. Files compressed on an NTFS volume can be read and written by any Windows-based application without first being decompressed by another program. Decompression occurs automatically when the file is read. The file is compressed again when it is closed or saved. Compressed files and folders have an attribute of **C** when viewed in Windows Explorer.

Only NTFS can read the compressed form of the data. When an application such as Microsoft[®] Word or an operating system command such as **copy** requests access to the file, the compression filter driver decompresses the file before making it available. For example, if you copy a compressed file from another Windows NT/2000–based computer to a compressed folder on your hard disk, the file is decompressed when read, copied, and then recompressed when saved.

This compression algorithm is similar to that used by the Windows 98 application DriveSpace 3, with one important difference — the limited functionality compresses the entire primary volume or logical volume. NTFS allows for the compression of an entire volume, of one or more folders within a volume, or even one or more files within a folder of an NTFS volume.

The compression algorithms in NTFS are designed to support cluster sizes of up to 4 KB. When the cluster size is greater than 4 KB on an NTFS volume, none of the NTFS compression functions are available.

Each NTFS data stream contains information that indicates whether any part of the stream is compressed. Individual compressed buffers are identified by "holes" following them in the information stored for that stream. If there is a hole, NTFS automatically decompresses the preceding buffer to fill the hole.

NTFS provides real-time access to a compressed file, decompressing the file when it is opened and compressing it when it is closed. When writing a compressed file, the system reserves disk space for the uncompressed size. The system gets back unused space as each individual compression buffer is compressed.

NTFS Encrypted Files (Windows 2000 only)

The Encrypting File System (EFS) provides the core file encryption technology used to store encrypted files on NTFS volumes. EFS keeps files safe from intruders who might gain unauthorized physical access to sensitive, stored data (for example, by stealing a portable computer or external disk drive).

EFS uses symmetric key encryption in conjunction with public key technology to protect files and ensure that only the owner of a file can access it. Users of EFS are issued a digital certificate with a public key and a private key pair. EFS uses the key set for the user who is logged on to the local computer where the private key is stored.

Users work with encrypted files and folders just as they do with any other files and folders. Encryption is transparent to the user who encrypted the file; the system automatically decrypts the file or folder when the user accesses. When the file is saved, encryption is reapplied. However, intruders who try to access the encrypted files or folders receive an "Access denied" message if they try to open, copy, move, or rename the encrypted file or folder.

To encrypt or decrypt a folder or file, set the encryption attribute for folders and files just as you set any other attribute. If you encrypt a folder, all files and subfolders created in the encrypted folder are automatically encrypted. It is recommended that you encrypt at the folder level.

NTFS Sparse Files (Windows 2000 only)

A sparse file has an attribute that causes the I/O subsystem to allocate only meaningful (nonzero) data. Nonzero data is allocated on disk, and non-meaningful data (large strings of data composed of zeros) is not. When a sparse file is read, allocated data is returned as it was stored; non-allocated data is returned, by default, as zeros.

NTFS deallocates sparse data streams and only maintains other data as allocated. When a program accesses a sparse file, the file system yields allocated data as actual data and deallocated data as zeros.

NTFS includes full sparse file support for both compressed and uncompressed files. NTFS handles read operations on sparse files by returning allocated data and sparse data. It is possible to read a sparse file as allocated data and a range of data without retrieving the entire data set, although NTFS returns the entire data set by default.

With the sparse file attribute set, the file system can deallocate data from anywhere in the file and, when an application calls, yield the zero data by range instead of storing and returning the actual data. File system application programming interfaces (APIs) allow for the file to be copied or backed as actual bits and sparse stream ranges. The net result is efficient file system storage and access. Next figure shows how data is stored with and without the sparse file attribute set.



Without sparse file attribute set

C-

Important:

Allocated

If you copy or move a sparse file to a FAT or a non-Windows 2000 NTFS volume, the file is built to its originally specified size. If the required space is not available, the operation does not complete.

Tip:

For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com

Data Integrity and Recoverability with NTFS

NTFS is a recoverable file system that guarantees the consistency of the volume by using standard transaction logging and recovery techniques. In the event of a disk failure, NTFS restores consistency by running a recovery procedure that accesses information stored in a log file. The NTFS recovery procedure is exact, guaranteeing that the volume is restored to a consistent state. Transaction logging requires a very small amount of overhead.

NTFS ensures the integrity of all NTFS volumes by automatically performing disk recovery operations the first time a program accesses an NTFS volume after the computer is restarted following a failure.

NTFS also uses a technique called cluster remapping to minimize the effects of a bad sector on an NTFS volume.

Important:

If either the master boot record (MBR) or boot sector is corrupted, you might not be able to access data on the volume.

Recovering Data with NTFS

NTFS views each I/O operation that modifies a system file on the NTFS volume as a transaction, and manages each one as an integral unit. Once started, the transaction is either completed or, in the event of a disk failure, rolled back (such as when the NTFS volume is returned to the state it was in before the transaction was initiated).

To ensure that a transaction can be completed or rolled back, NTFS records the suboperations of a transaction in a log file before they are written to the disk. When a complete transaction is recorded in the log file, NTFS performs the suboperations of the transaction on the volume cache. After NTFS updates the cache, it commits the transaction by recording in the log file that the entire transaction is complete.

Once a transaction is committed, NTFS ensures that the entire transaction appears on the volume, even if the disk fails. During recovery operations, NTFS redoes each committed transaction found in the log file. Then NTFS locates the transactions in the log file that were not committed at the time of the system failure and undoes each transaction suboperation recorded in the log file. Incomplete modifications to the volume are prohibited.

NTFS uses the Log File service to log all redo and undo information for a transaction. NTFS uses the redo information to repeat the transaction. The undo information enables NTFS to undo transactions that are not complete or that have an error.



Important:

NTFS uses transaction logging and recovery to guarantee that the volume structure is not corrupted. For this reason, all system files remain accessible after a system failure. However, user data can be lost because of a system failure or a bad sector.

Cluster Remapping

In the event of a bad-sector error, NTFS implements a recovery technique called cluster remapping. When Windows 2000 detects a bad-sector, NTFS dynamically remaps the cluster containing the bad sector and allocates a new cluster for the data. If the error occurred during a read, NTFS returns a read error to the calling program, and the data is lost. If the error occurs during a write, NTFS writes the data to the new cluster, and no data is lost.

NTFS puts the address of the cluster containing the bad sector in its bad cluster file so the bad sector is not reused.



Important:

Cluster remapping is *not* a backup alternative. Once errors are detected, the disk should be monitored closely and replaced if the defect list grows. This type of error is displayed in the Event Log.



For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com microsoft Developers Network (MSDN) http://www.microsoft.com

Extended File System (exFAT)

Extended File System (exFAT) is a successor of FAT family of file systems (FAT12/16/32). It has similar design though renders many significant improvements:

- Larger volume and file size limits
- Native Unicode file names
- Bigger boot area allowing a larger boot code
- Better performance
- Time zone offset support
- OEM parameters support

exFAT vs. FAT32 Comparison

Feature	FAT32	exFAT
Maximum Volume Size	8 TB*	128 PB
Maximum File Size	4 GB	16 EB
Maximum Cluster Size	32 KB **	32 MB
Maximum Cluster Count	228	232
Maximum File Name Length	255	255
Date/Time resolution	2 s	10 ms
MBR Partition Type Identifier	0x0B, 0x0C	0x07

Notice: Windows cannot format FAT32 volumes bigger than 32GB, though itsupports larger volumes created by third party implementations; 16 TB is the maximum volume size if formatted with 64KB cluster



Notice: According to Microsoft KB184006 clusters cannot be 64KB or larger, though some third party implementations support up to 64KB.

Volume Layout

Offset, sectors	Size, sectors	Block	Comments
Main Boot Region			
0	1	Boot Sector	
1	8	Extended Boot Sectors	
9	1	OEM Parameters	
10	1	Reserved	
11	1	Boot Checksum	
Backup Boot Region			
12	1	Boot Sector	

Offset, sectors	Size, sectors	Block	Comments
13	8	Extended Boot Sectors	
21	1	OEM Parameters	
22	1	Reserved	
23	1	Boot Checksum	
FAT Region			
24	FatOffset - 24	FAT Alignment	Boot Sectors contain FatOffset
FatOffset	FatLength	First FAT	Boot Sectors contain FatOffset and FatLength
FatOffset + FatLength	FatLength	Second FAT	For TexFAT only
Data Region			
FatOffset + FatLength * NumberOfFats	ClusterHeapOffset – (FatOffset + FatLength * NumberOfFats)	Cluster Heap Alignment	
ClusterHeapOffset	ClusterCount * 2^SectorsPerClusterShift	Cluster Heap	
ClusterHeapOffset + ClusterCount * 2^SectorsPerClusterShift	VolumeLength – (ClusterHeapOffset + ClusterCount * 2^SectorsPerClusterShift)	Excess Space	

Navigate to detailed volume specification using following links:

- *Boot Sector* on page 118
- *Extended Boot Sector* on page 120
- OEM Parameters on page 120
- *Boot Checksum* on page 121
- *File Allocation Table (FAT)* on page 121

Boot Sector

Offset	Size	Description	Comments
0 (0x00)	3	JumpBoot	0xEB7690
3 (0x03)	8	FileSystemName	"EXFAT "
11 (0x0B)	53	MustBeZero	
64 (0x40)	8	PartitionOffset	In sectors; if 0, shall be ignored
72 (0x48)	8	VolumeLength	Size of exFAT volume in sectors
80 (0x50)	4	FatOffset	In sectors
84 (0x54)	4	FatLength	In sectors. May exceed the required space in order to align the second FAT

Offset	Size	Description	Comments
88 (0x58)	4	ClusterHeapOffset	In sectors
92 (0x5C)	4	ClusterCount	2^32-11 is the maximum number of clusters could be described
96 (0x60)	4	RootDirectoryCluster	
100 (0x64)	4	VolumeSerialNumber	
104 (0x68)	2	FileSystemRevision	as MAJOR.minor, major revision is high byte, minor is low byte; currently 01.00
106 (0x6A)	2	VolumeFlags (see below)	
108 (0x6C)	1	BytesPerSectorShift	Power of 2. Minimum 9 (512 bytes per sector), maximum 12 (4096 bytes per sector)
109 (0x6D)	1	SectorsPerCluster Shift	Power of 2. Minimum 0 (1 sector per cluster), maximum 25 – BytesPerSectorShift, so max cluster size is 32 MB
110 (0x6E)	1	NumberOfFats	2 is for TexFAT only
111 (0x6F)	1	DriveSelect	Extended INT 13h drive number; typically 0x80
112 (0x70)	1	PercentInUse	0100 – percentage of allocated clusters rounded down to the integer 0xFF – percentage is not available
113 (0x71)	7	Reserved	
120 (0x78)	390	BootCode	
510 (0x1FE)	2	BootSignature	0xAA55
512 (0x200)	2^BytesPerSectorShift - 512	ExcessSpace	Not used

Table 5: Volume Flags

Offset	Size	Field
0	1	ActiveFat 0 - First FAT and Allocation Bitmap are active, 1 - Second .
1	1	VolumeDirty (0-clean, 1-dirty)
2	1	MediaFailure (0 – no failures reported or they already marked

		as BAD clusters) 1- some read/ write operations failed)
3	1	ClearToZero (no meaning)
4	12	Reserved

Extended Boot Sector

Offset	Size	Description	Comments
0 (0x00)	2^BytesPerSectorShift - 4	ExtendedBootCode	
2^BytesPerSectorShift - 4	4	ExtendedBootSignature	0xAA550000

Whole sector is used for boot code except last 4 bytes used for signature in each sector. If Extended Boot Sector is not used, it should be filled with 0x00. Extended signature must be preserved.

OEM Parameters

Offset	Size	Description	Comments
0 (0x00)	48	Parameters[0]	
432 (0x1B0)	48	Parameters[9]	
480 (0x01E0)	2^BytesPerSectorShift - 480	Reserved	

OEM parameters are ignored by Windows but can be used by OEM implementations. OEMs can define their own parameters with unique GUIDs. All unused Parameters fields must be described as unused by GUID_NULL in ParameterType.

This structure must be preserved during exFAT formatting, except in the case of secure wipe.

Table 6: OEM Parameter Record

Offset	Size	Description	Comments
0x00	16	ParameterType	OEM defined GUID , GUID_NULL indicate that parameter value is not used
0x10	32	ParameterValue	OEM specific

```
#define OEM_FLASH_PARAMETER_GUID 0A0C7E46-3399-4021-90C8-FA6D389C4BA2
struct
{
    GUID OemParameterType; //Value is OEM_FLASH_PARAMETER_GUID
    UINT32 EraseBlockSize; //Erase block size in bytes
    UINT32 PageSize;
    UINT32 NumberOfSpareBlocks;
    UINT32 tRandomAccess; //Random Access Time in nanoseconds
    UINT32 tProgram; //Program time in nanoseconds
    UINT32 tReadCycle; // Serial read cycle time in nanoseconds
    UINT32 tWriteCycle; // Write Cycle time in nanoseconds
```

```
UCHAR Reserved[4];
}
FlashParameters;
```

Boot Checksum

This sector contains a repeating 32-bit checksum of the previous 11 sectors. The checksum calculation excludes VolumeFlags and PercentInUse fields in Boot Sector (bytes 106, 107, 112). The checksum is repeated until the end of the sector. The number of repetitions depends on the size of the sector.

```
UINT32 BootChecksum(const unsigned char data[], int bytes)
{
  UINT32 checksum = 0;
  for (int i = 0; i < bytes; i++)
   {
    if (i == 106 || i == 107 || i == 112)
        continue;
    checksum = (checksum << 31) | (checksum >> 1) + data[i];
    }
  return checksum;
}
```

File Allocation Table (FAT)

File Allocation Table (FAT) may contain 1 or 2 FATs, as defined in NumberOfFats field. ActiveFat field in VolumeFlags in the Main Boot Sector determines which FAT is active.

The first cluster is cluster 2, as in FAT32. Each FatEntry represents one cluster

In exFAT, FAT is not used for tracking an allocation; an Allocation Bitmap is used for this purpose. FAT is only used for keeping chains of clusters of fragmented files. If a file is not fragmented, FAT table does not need to be updated. A Stream Extensions Directory Entry should be consulted to determine if the FAT chain is valid or not. If FAT chain is not valid, it does not need to be zeroed.

Offset	Size	Description	Comments
0 (0x00)	4	FatEntry[0]	Media type (should be 0xFFFFFFF8)
4 (0x04)	4	FatEntry[1]	Must be 0xFFFFFFFF
8 (0x08)	4	FatEntry[2]	First cluster
(ClusterCount + 1) * 4	4	FatEntry[ClusterCount + 1]	Last cluster
(ClusterCount + 2) * 4	Remainder of sector	ExcessSpace	

Valid values of FAT entries:

0x0000002

ClusterCount +1 (max 0xFFFFFF6) – next cluster in the chain

0xFFFFFFF7

bad cluster

0xFFFFFF8

media descriptor

0xFFFFFFFF

end of file (EOF mark)

Value 0x00000000 does not mean the cluster is free, it is an undefined value.

The second FAT table (presents only in TexFAT) is located immediately after the first one and has the same size.

exFAT Directory Structure

exFAT uses tree structure to describe relationship between files and directories. The root of the directory tree is defined by directory located at RootDirectoryCluster. Subdirectories are single-linked to there parents. There is no special (.) and (..) directories pointing to itself and to parent like in FAT16/FAT32.

Each directory consists of a series of directory entries. Directory entries are classified as critical/benign and primary/secondary as follows:

- Primary Directory Entries
- Critical Primary Entries
- Benign Primary Entries
- Secondary Directory Entries
- Critical Secondary Entries
- Benign Secondary Entries

Critical entries are required while benign entries are optional. Primary directory entries correspond to the entries in file system and describe main characteristics. Secondary directory entries extend the metadata associated with a primary directory entry end follow it. A group of primary/secondary entries make up a directory entry set describing a file or directory. The first directory entry in the set is a primary directory entry. All subsequent entries, if any, must be secondary directory entries.

Each directory entry derives from Generica Directory Entry template. Size of directory entry is 32 bytes.

Table 7: Generic Directory Entry Template

Offset	Size	Description	Comments
0 (0x00)	1	EntryType (see below)	
1 (0x01)	19	CustomDefined	
20 (0x14)	4	FirstCluster	0 – no cluster allocation 2ClusterCount+1 – cluster index
24 (0x18)	8	DataLength	In bytes

Table 8: Enty Types description

Bits	Size	Description	Comments
0-4	5	Code	
5	1	Importance	0 – Critical entry, 1 – Benign entry
6	1	Category	0 – Primary entry, 1 – Secondary entry
7	1	In use status	0 – Not in use, 1 – In use

EntryType can have the following values:

- **0x00** End Of Directory marker. All other fields in directory entry are invalid. All subsequent directory entries are also End Of Directory markers
- **0x01-0x7F** (InUse = 0). All other fields in this entry are not defined
- **0x81-0xFF** (InUse = 1). Regular record with all fields defined.

Offset	Size	Description	Comments
0 (0x00)	1	EntryType	
1 (0x01)	1	SecondaryCount	Number of secondary entries which immediately follow this primary entry and together comprise a directory entry set. Valid value is 0255
2 (0x02)	2	SetChecksum	Checksum of all directory entries in the given set excluding this field. See EntrySetCheckSum().
4 (0x04)	2	GeneralPrimaryFlags (se below)	e
6 (0x06)	14	CustomDefined	
20 (0x14)	4	FirstCluster	
24 (0x18)	8	DataLength	
Bits	Size	Description	Comments
0	1	AllocationPossible	0-not possible (FirstCluster and DataLength undefined), 1-possible
1	1	NoFatChain	0-FAT cluster chain is valid 1-FAT cluster chain is not used (contiguous data)
2	14	CustomDefined	

Table 9: Generic Primary Directory Entry Template

All critical primary directory entries are located in root directory (except file directory entries). Benign primary directory enries are optional. If one benign primary entry is not recognized, all directory entry set is ignored.

```
// data points to directory entry set in memory
UINT16 EntrySetChecksum(const unsigned char data[], int secondaryCount)
{
  UINT16 checksum = 0;
  int bytes = (secondaryCount + 1) * 32;
  for (int i = 0; i < bytes; i++)
  {
</pre>
```

```
if (i == 2 || i == 3)
   continue;
   checksum = (checksum << 15) | (checksum >> 1) + data[i];
}
return checksum;
}
```

exFAT Defined Directory Entries

Main exFAT Diectory entries defined in table below:

Table 10: Defined Directory Entries list

EntryType	Primary	Critical	Code	Directory Entry Name
0x81	boolean: yes	boolean: yes	1	Allocation Bitmap
0x82	boolean: yes	boolean: yes	2	Up-case Table
0x83	boolean: yes	boolean: yes	3	Volume Label
0x85	boolean: yes	boolean: yes	5	File
0xA0	boolean: yes	boolean: no	0	Volume GUID
0xA1	boolean: yes	boolean: no	1	TexFAT Padding
0xA2	boolean: yes	boolean: no	2	Windows CE Access Control Table
0xC0	boolean: no	boolean: yes	0	Stream Extension
0xC1	boolean: no	boolean: yes	1	File Name

Read about Directory entries below:

- Allocation Bitmap Directory Entry on page 124
- Up-Case Table Directory Entry on page 125
- Volume Label Directory Entry on page 125
- File Directory Entry on page 126
- Volume GUID Directory Entry on page 128
- TexFAT Padding Directory Entry on page 128
- Windows CE Access Control Table Directory Entry on page 129
- Stream Extension Directory Entry on page 129
- File Name Directory Entry on page 130

Allocation Bitmap Directory Entry

Offset	Size	Description	Comments
0 (0x00)	1	Entry type	0x81
1 (0x01)	1	BitmapFlags (see below)	Indicates which Allocation Bitmap the given entry describes
2 (0x02)	18	Reserved	
20 (0x14)	4	First Cluster	
24 (0x18)	8	Data Length	

Table 11: Bitmap Flags

Bits	Size	Description	Comments
0	1	BitmapIdentifier	0 – 1st bitmap, 1 - 2nd bitmap
1	7	Reserved	

The number of bitmaps and therefore a number of Bitmap Allocation entries is equal to the number of FATs. In case of TexFAT two FATs are used and bit 0 of Flags indicates which bitmap and FAT are referred.

The First Allocation Bitmap shall be used in conjunction with the First FAT and the Second Allocation Bitmap shall be used with the Second FAT. ActiveFat field in Boot Sector defines which FAT and Allocation Bitmap are active.

Bitmap size in bytes must be a number of clusters in the volume divided by 8 and rounded up.

Up-Case Table Directory Entry

Offset	Size	Description	Comments
0 (0x00)	1	Entry type	0x82
1 (0x01)	3	Reserved1	
4 (0x04)	4	TableChecksum	Up-case Table checksum
8 (0x08)	12	Reserved2	
20 (0x14)	4	FirstCluster	
24 (0x18)	8	DataLength	

The checksum is calculated against DataLength bytes of Up-case Table according to the following code:

```
UINT32 UpCaseTableChecksum(const unsigned char data[], int bytes)
{
  UINT32 checksum = 0;
  for (int i = 0; i < bytes; i++)
    checksum = (checksum << 31) | (checksum >> 1) + data[i];
  return checksum;
}
```

Volume Label Directory Entry

Offset	Size	Description	Comments
0 (0x00)	1	Entry type	0x83
1 (0x01)	1	CharacterCount	Length in Unicode characters (max 11)
2 (0x02)	22	VolumeLabel	Unicode string
24 (0x18)	8	Reserved	

If volume is formatted without a label, the Volume Label Entry will be present but Entry Type will be set to 0x03 (not in use).

File Directory Entry

File directory entry describes files and directories. It is a primary critical directory entry and must be immediately followed by 1 Stream Extension directory entry and from 1 to 17 File Name directory entries. Those 3-19 directory entries comprise a directory entry set describing a single file or a directory.

Offset	Size	Description	Comments
0 (0x00)	1	Entry type	0x85
1 (0x01)	1	SecondaryCount	Must be from 2 to 18
2 (0x02)	2	SetChecksum	
4 (0x04)	2	FileAttributes (see below)	
6 (0x06)	2	Reserved1	
8 (0x08)	4	CreateTimestamp	
12 (0x0C)	4	LastModifiedTimestamp	
16 (0x10)	4	LastAccessedTimestamp	
20 (0x14)	1	Create10msIncrement	0199
21 (0x15)	1	LastModified10msIncreme	er û 199
22 (0x16)	1	CreateTimezoneOffset	Offset from UTC in 15 min increments
23 (0x17)	1	LastModifiedTimezoneOff	eooffset from UTC in 15 min increments
24 (0x18)	1	LastAccessedTimezoneOf	fs ef fset from UTC in 15 min increments
25 (0x19)	7	Reserved2	

Table 12: File Attributes

Bits	Size	Attribute	Comments
0	1	ReadOnly	
1	1	Hidden	
2	1	System	
3	1	Reserved1	
4	1	Directory	
5	1	Archive	
6	10	Reserved2	

Table 13: Timestamp Format

Bits	Size	Description	Comments
0-4	5	Seconds (as number of 2-second intervals)	029 29 represents 58 seconds
5-10	6	Minutes	059
11-15	5	Hour	023

Bits	Size	Description	Comments
16-20	5	Day	131
21-24	4	Month	112
25-31	7	Year (as offset from 1980)	0 represents 1980

Timestamp format records seconds as 2 seconds intervals, so 10ms increments are used to increase precision from 2 seconds to 10 milliseconds. The valid values are from 0 to 199 in 10ms intervals which are added to correspondent timestamp. Timestamp is recorded in local time.

Time zone offset is expressed in 15 minutes increments.

TimezoneOffset field	TZ Offset	Time Zone	Comments
128 (0x80)	UTC	Greenwich Standard Time	
132 (0x84)	UTC+01:00	Central Europe Time	
136 (0x88)	UTC+02:00	Eastern Europe Standard Time	
140 (0x8C)	UTC+03:00	Moscow Standard Time	
144 (0x90)	UTC+04:00	Arabian Standard Time	
148 (0x94)	UTC+05:00	West Asia Standard Time	
152 (0x98)	UTC+06:00	Central Asia Standard Time	
156 (0x9C)	UTC+07:00	North Asia Standard Time	
160 (0xA0)	UTC+08:00	North Asia East Standard Time	
164 (0xA4)	UTC+09:00	Tokyo Standard Time	
168 (0xA8)	UTC+10:00	West Pacific Standard Time	
172 (0xAC)	UTC+11:00	Central Pacific Standard Time	
176 (0xB0)	UTC+12:00	New Zealand Standard Time	
180 (0xB4)	UTC+13:00	Tonga Standard Time	
208 (0xD0)	UTC-12:00	Dateline Standard Time	
212 (0xD4)	UTC-11:00	Samoa Standard Time	
216 (0xD8)	UTC-10:00	Hawaii Standard Time	
220 (0xDC)	UTC-09:00	Alaska Standard Time	
224 (0xE0)	UTC-08:00	Pacific Standard Time	
228 (0xE4)	UTC-07:00	Mountain Standard Time	
232 (0xE8)	UTC-06:00	Central Standard Time	

TimezoneOffset field	TZ Offset	Time Zone	Comments
236 (0xEC)	UTC-05:00	Eastern Standard Time	
240 (0xF0)	UTC-04:00	Atlantic Standard time	
242 (0xF2)	UTC-03:30	Newfoundland Standard Time	
244 (0xF4)	UTC-03:00	Greenland Standard Time	
248 (0xF8)	UTC-02:00	Mid-Atlantic Standard Time	
252 (0xFC)	UTC-01:00	Azores Standard Time	

Volume GUID Directory Entry

In following table presented a benign primary directory entry and may not present in a file system.

Offset	Size	Description	Comments
0 (0x00)	1	EntryType	0xA0
1 (0x01)	1	SecondaryCount	Must be 0x00
2 (0x02)	2	SetChecksum	
4 (0x04)	2	GeneralPrimaryFlags (See below)	
6 (0x06)	16	VolumeGuid	All values are valid except null GUID {00000000-0000-0000-0000-00
22 (0x16)	10	Reserved	

Table 15: Primary Flags Definitions

Bits	Size	Description	Comments
0	1	AllocationPossible	Must be 0
1	1	NoFatChain	Must be 0
2	14	CustomDefined	

TexFAT Padding Directory Entry

Offset	Size	Description	Comments
0 (0x00)	1	EntryType	0xA1
1 (0x01)	31	Reserved	



Remember:

exFAT 1.00 does not define TexFAT Padding directory entry. TexFAT Padding directory entries are only valid in the first cluster of directory and occupy every directory entry of the cluster. The implementations should not move TexFAT Padding directory entries.

Windows CE Access Control Table Directory Entry

Offset	Size	Description	Comments
0 (0x00)	1	EntryType	0xA2
1 (0x01)	31	Reserved	

(**(**-

exFAT 1.00 does not define Windows CE Access Control Table Directory Entry.

Offset	Size	Description	Comments
0 (0x00)	1	EntryType	0xC0
1 (0x01)	1	GeneralSecondaryFlags (see below)	
2 (0x02)	1	Reserved1	
3 (0x03)	1	NameLength	Length of Unicode name contained in subsequent File Name directory entries
4 (0x04)	2	NameHash	Hash of up-cased file name
6 (0x06)	2	Reserved2	
8 (0x08)	8	ValidDataLength	Must be between 0 and DataLength
16 (0x10)	4	Reserved3	
20 (0x14)	4	FirstCluster	
24 (0x18)	8	DataLength	For directories maximum 256 MB

Stream Extension Directory Entry

Remember:

Table 16: Secondary Flags Definitions

Bits	Size	Description	Comments
0	1	AllocationPossible	Must be 1
1	1	NoFatChain	
2	14	CustomDefined	

Stream Extension directory entry must immediately follow the File directory entry in the set. It could be only one Stream Extension entry in the set. If NoFatChain flag is set, all allocated clusters are contiguous.

The NameHash field facilitates the purpose of fast file name comparison and is performed on up-cased file name. NameHash verify against a mismatch, however matching hashes cannot guarantee the equality of file names. If name hashes match, a subsequent full name comparison must be performed.

```
// fileName points to up-cased file name
UINT16 NameHash(WCHAR *fileName, int nameLength)
{
    UINT16 hash = 0;
```

```
unsigned char *data = (unsigned char *)fileName;
for (int i = 0; i < nameLength * 2; i++)
   hash = (hash << 15) | (hash >> 1) + data[i];
return hash;
}
```

ValidDataLength determines how much actual data written to the file. Implementation shall update this field as data has been written. The data beyond the valid data length is undefined and implementation shall return zeros.

File Name Directory Entry

Offset	Size	Description	Comments
0 (0x00)	1	EntryType	0xC1
1 (0x01)	1	GeneralSecondaryFlags (see below)	
2 (0x02)	30	FileName	

Table 17: Secondary Flags Definitions

Bits	Size	Description	Comments
0	1	AllocationPossible	Must be 0
1	1	NoFatChain	Must be 0
2	14	CustomDefined	

File Name directory entries must immediately follow the Steam Extension directory entry in the number of NameLength/15 rounded up. The maximum number of File Name entries is 17, each can hold up to 15 Unicode characters and the maximum file name length is 255. Unused portion of FileName field must be set to 0x0000.

Table 18: Invalid File Name Characters

Character Code	Character	Description
0x0000 – 0x001F		Control codes
0x0022	N	Quotation mark
0x002A	*	Asterisk
0x002F	1	Forward slash
0x003A	:	Colon
0x003C	<	Less than
0x003E	>	Greater than
0x003F	?	Question mark
0x005C	/	Back slash
0x007C		Vertical bar

exFAT Cluster Heap

The cluster heap is a set of clusters which hold data in exFAT. It contains:

- Root Directory
- Files
- Directories
- Allocation Bitmap on page 131
- *Up-case Table* on page 131

The allocation status of clusters in cluster heap is tracked by Bitmap Allocation Table which itself located inside the cluster heap.

Allocation Bitmap

Allocation Bitmap keeps track of the allocation status of clusters. FAT does not serve this purpose as in FAT16/FAT32 file system. Allocation Bitmap consists of a number of 8 bit bytes which can be treated as a sequence of bits. Each bit in bitmap corresponds to a data cluster. If it has a value of 1, the cluster is occupied, if 0 - the cluster is free. The least significant bit of bitmap table refers to the first cluster, i.e. cluster 2.

Offset	Size	Description	Comments
0x00	1	1st byte	Clusters 2-9
0x01	1	2nd byte	Clusters 10-17
0x02	1	3rd byte	Clusters 18-25

Bitmap allocation table resides in cluster heap and referred by Bitmap Directory entry in root directory.

In TexFAT could be 2 Bitmap Allocation tables, otherwise there will be only one bitmap. The NumberOfFats field in Boot Sectors determines the number of valid Allocation Bitmap directory entries in the root directory and the number of Allocation Bitmaps.

Up-case Table

Up-case table contains data used for conversion from lower-case to upper-case characters. File Name Directory Entry uses Unicode characters and preserves case when storing file name. exFAT itself is case insensitive, so it needs to compare file names converted to the upper-case during search operations.

Normally Up-case table is located right after Bitmap Allocation table but can be placed anywhere is the cluster heap. It has a corresponding primary critical directory entry in the root directory.

Up-case Table is an array of Unicode characters, an index of which represents the Unicode characters to be up-cased and the value is the target up-cased character. The Up-case Table shall contain at least 128 mandatory Unicode mappings. If implementation supports only mandatory 128 characters it may ignore the rest of Up-case Table. When up-casing file names such implementation shall up-case only characters from the mandatory 128 characters set and leave other characters intact. When comparing file names which are different only by characters in non-mandatory set, those file names shall be treated as equal.

Index	Value	Comments
0x0000	0x0000	
0x0001	0x0001	
0x0002	0x0002	

Index	Value	Comments
0x0041	0x0041	'A' is mapped into itself (identity mapping)
0x0042	0x0042	'B' is mapped into itself
0x061	0x041	`a' is mapped into `A' (non-identity mapping)
0x062	0x0042	'b' is mapped into 'B'

Up-case Table can be written in compressed format where the series of identity mappings is represented with 0xFFFF followed by the number of identity mappings.

Mandatory First 128 Up-case Table Entries

Index | Table Entries

0000 - 0000 0001 0002 0003 0004 0005 0006 0007 0008 0009 000A 000B 000C 000D 000E 000F - 0010 0011 0012 0013 0014 0015 0016 0017 0018 0019 001A 001B 001C 001D 0010 001E 001F - 0020 0021 0022 0023 0024 0025 0026 0027 0028 0029 002A 002B 002C 002D 0020 002E 002F - 0030 0031 0032 0033 0034 0035 0036 0037 0038 0039 003A 003B 003C 003D 0030 003E 003F - 0040 0041 0042 0043 0044 0045 0046 0047 0048 0049 004A 004B 004C 004D 0040 004E 004F 0050 - 0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 005A 005B 005C 005D 005E 005F 0060 - 0060 0041 0042 0043 0044 0045 0046 0047 0048 0049 004A 004B 004C 004D 004E 004F 0070 - 0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 005A 007B 007C 007D 007E 007F

Remember:

Non-identity mappings are highlighted in **bold**.

Mandatory First 128 Up-case Table Entries in compressed format

Index | Table Entries

0000 - FFFF 0061 0041 0042 0043 0044 0045 0046 0047 0048 0049 004A 004B 004C 004D 004E 0010 - 004F 0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 005A FFFF 0005

The first highlighted group describes that first 0x0061 characters (0x0000-0x0060) have identity mappings. The next character after it (0x0061) maps to 0x0041 etc. until the next compressed group is encountered.



T CF

Remember:

The first highlighted **in bold** group describes that first 0x0061 characters (0x0000-0x0060) have identity mappings. The next character after it (0x0061) maps to 0x0041 etc. until the next compressed group is encountered.

File System (FAT)

The FAT file system is a simple file system originally designed for small disks and simple folder structures. The FAT file system is named for its method of organization, the File Allocation Table, which resides at the beginning of the volume. To protect the volume, two copies of the table are kept, in case one becomes damaged. In addition, the file allocation tables and the root folder must be stored in a fixed location so that the files needed to start the system can be correctly located.

A volume formatted with the FAT file system is allocated in clusters. The default cluster size is determined by the size of the volume. For the FAT file system, the cluster number must fit in 16 bits and must be a power of two.

Partition Boot Sector	FAT1	FAT2 (duplicate)	Root folder	Other folders and all files.

Figure 22: FAT file system volume organization

See the next sections for more information about FAT:

- FAT Partition Boot Sector on page 133
- FAT File Allocation Table on page 137
- FAT Root Folder on page 138
- FAT Folder Structure on page 138
- FAT32 Features on page 139

Main differences between FAT12, FAT16, FAT32

- FAT12 file system contains 1.5 bytes per cluster within the file allocation table.
- FAT16 file system contains 2 bytes per cluster within the file allocation table.
- FAT32 file system includes 4 bytes per cluster within the file allocation table.

FAT Partition Boot Sector

The Partition Boot Sector contains information that the file system uses to access the volume. On x86based computers, the Master Boot Record use the Partition Boot Sector on the system partition to load the operating system kernel files.

Next table describes the fields in the Partition Boot Sector for a volume formatted with the FAT file system.

Table 19: S	ystem ID	field	description
-------------	----------	-------	-------------

Byte Offset (in hex)	Field Length	Sample Value	Meaning
00	3 bytes	EB 3C 90	Jump instruction
03	8 bytes	MSDOS5.0	OEM Name in text
ОВ	25 bytes		BIOS Parameter Block
24	26 bytes		Extended BIOS Parameter Block
3E	448 bytes		Bootstrap code
1FE	2 bytes	0x55AA	End of sector marker

Byte Offset	Field Length	Sample Value	Meaning
0х0В	WORD	0x0002	Bytes per Sector. The size of a hardware sector. For most disks in use in the United States, the value of this field is 512.
0x0D	BYTE	0×08	Sectors Per Cluster. The number of sectors in a cluster. The default cluster size for a volume depends on the volume size and the file system.
0x0E	WORD	0x0100	Reserved Sectors. The number of sectors from the Partition Boot Sector to the start of the first file allocation table, including the Partition Boot Sector. The minimum value is 1. If the value is greater than 1, it means that the bootstrap code is too long to fit completely in the Partition Boot Sector.
0x10	BYTE	0x02	Number of file allocation tables (FATs). The number of copies of the file allocation table on the volume. Typically, the value of this field is 2.
0x11	WORD	0x0002	Root Entries. The total number of file name entries that can be stored in the root folder of the volume. One entry is always used as a Volume Label. Files with long filenames use up multiple entries

Table 20: BIOS Parameter Block and Extended BIOS Parameter Block Fields

Byte Offset	Field Length	Sample Value	Meaning
			per file. Therefore, the largest number of files in the root folder is typically 511, but you will run out of entries sooner if you use long filenames.
0x13	WORD	0x0000	Small Sectors. The number of sectors on the volume if the number fits in 16 bits (65535). For volumes larger than 65536 sectors, this field has a value of 0 and the Large Sectors field is used instead.
0x15	BYTE	0xF8	Media Type. Provides information about the media being used. A value of 0xF8 indicates a hard disk.
0x16	WORD	0xC900	Sectors per file allocation table (FAT). Number of sectors occupied by each of the file allocation tables on the volume. By using this information, together with the Number of FATs and Reserved Sectors, you can compute where the root folder begins. By using the number of entries in the root folder, you can also compute where the user data area of the volume begins.
0x18	WORD	0x3F00	Sectors per Track. The apparent disk geometry in use when the disk was low-level formatted.
0x1A	WORD	0x1000	Number of Heads. The apparent disk

Byte Offset	Field Length	Sample Value	Meaning
			geometry in use when the disk was low-level formatted.
0x1C	DWORD	3F 00 00 00	Hidden Sectors. Same as the Relative Sector field in the Partition Table.
0x20	DWORD	51 42 06 00	Large Sectors. If the Small Sectors field is zero, this field contains the total number of sectors in the volume. If Small Sectors is nonzero, this field contains zero
0x24	BYTE	0x80	Physical Disk Number. This is related to the BIOS physical disk number. Floppy drives are numbered starting with 0x00 for the A disk. Physical hard disks are numbered starting with 0x80. The value is typically 0x80 for hard disks, regardless of how many physical disk drives exist, because the value is only relevant if the device is the startup disk.
0x25	BYTE	0x00	Current Head. Not used by the FAT file system.
0x26	ВҮТЕ	0x29	Signature. Must be either 0x28 or 0x29 in order to be recognized by Windows NT.
0x27	4 bytes	CE 13 46 30	Volume Serial Number. A unique number that is created when you format the volume.
0x2B	11 bytes	NO NAME	Volume Label. This field was used to store the volume

Byte Offset	Field Length	Sample Value	Meaning
			label, but the volume label is now stored as special file in the root directory.
0x36	8 bytes	FAT16	System ID. Either FAT12 or FAT16, depending on the format of the disk.



For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://msdn.microsoft.com

FAT File Allocation Table

The FAT file system is named for its method of organization, the file allocation table, which resides at the beginning of the volume. To protect the volume, two copies of the table are kept, in case one becomes damaged. In addition, the file allocation tables must be stored in a fixed location so that the files needed to start the system can be correctly located.

The file allocation table contains the following types of information about each cluster on the volume (see example below for FAT16):

- Unused (0x0000)
- Cluster in use by a file
- Bad cluster (0xFFF7)
- Last cluster in a file (0xFFF8-0xFFFF)

There is no organization to the FAT folder structure, and files are given the first available location on the volume. The starting cluster number is the address of the first cluster used by the file. Each cluster contains a pointer to the next cluster in the file, or an indication (0xFFFF) that this cluster is the end of the file. These links and end of file indicators are shown below.



Figure 23: Example of File Allocation Table

This illustration shows three files. The file File1.txt is a file that is large enough to use three clusters. The second file, File2.txt, is a fragmented file that also requires three clusters. A small file, File3.txt, fits completely in one cluster. In each case, the folder entry (see *folder entry* for details) points to the first cluster of the file.



Tip:

For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://msdn.microsoft.com

FAT Root Folder

The root folder contains an entry for each file and folder on the root. The only difference between the root folder and other folders is that the root folder is on a specified location on the disk and has a fixed size (512 entries for a hard disk, number of entries on a floppy disk depends on the size of the disk).

See FAT Folder Structure on page 138 topic for details about folder organization.



For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com/

FAT Folder Structure

Folders have set of 32-byte *Folder Entries* for each file and sub-folder contained in the folder (see example figure below).

The *Folder Entry* includes the following information:

- Name (eight-plus-three characters)
- Attribute byte (8 bits worth of information, described later in this section)
- Create time (24 bits)
- Create date (16 bits)
- Last access date (16 bits)
- Last modified time (16 bits)
- Last modified date (16 bits.)
- Starting cluster number in the file allocation table (16 bits)
- File size (32 bits)

There is no organization to the FAT folder structure, and files are given the first available location on the volume. The starting cluster number is the address of the first cluster used by the file. Each cluster contains a pointer to the next cluster in the file, or an indication (0xFFFF) that this cluster is the end of the file. See *File Allocation Table* for details.

The information in the folder is used by all operating systems that support the FAT file system. In addition, Windows NT can store additional time stamps in a FAT folder entry. These time stamps show when the file was created or last accessed and are used principally by POSIX applications.

Because all entries in a folder are the same size, the attribute byte for each entry in a folder describes what kind of entry it is. One bit indicates that the entry is for a sub folder, while another bit marks the entry as a volume label. Normally, only the operating system controls the settings of these bits.

A FAT file has four attributes bits that can be turned on or off by the user — archive file, system file, hidden file, and read-only file.

File names on FAT Volumes

Beginning with Windows NT 3.5, files created or renamed on FAT volumes use the attribute bits to support long file names in a way that does not interfere with how MS-DOS or OS/2 accesses the volume. Whenever a user creates a file with a long file name, Windows creates an eight-plus-three name for the file. In addition to this conventional entry, Windows creates one or more secondary folder entries for the file, one for each 13 characters in the long file name. Each of these secondary folder entries stores a corresponding part of the long file name in Unicode. Windows sets the volume, read-only, system, and hidden file attribute bits of the secondary folder entry to mark it as part of a long file name. MS-DOS and OS/2 generally ignore folder entries with all four of these attribute bits set, so these entries are effectively invisible to these operating systems. Instead, MS-DOS and OS/2 access the file by using the conventional eight-plus-three file name contained in the folder entry for the file.

Figure below shows all of the folder entries for the file Thequi~1.fox, which has a long name of The quick brown.fox. The long name is in Unicode, so each character in the name uses two bytes in the folder entry. The attribute field for the long name entries has the value 0x0F. The attribute field for the short name is 0x20.

	(and la	st)	y													
	0×42	u	N	, r	ì			' 1	f)	0×0F	0×00	check sum	×	(
l	0×0	000	0×F	FFF	0×F	FFF	0×F	FFF	0×F	FFF	0×0	000	0×F	FFF	0×F	FFF
ſ	0×01		Г	ł	1		,				1	0×0F	0×00	check sum	u u	I
l	i			;	1	4			')	0×0	000		r	• • •	•
ĺ	т	н	E	Q	U	I	~	1	F	0	×	0×20	NT	Cr	reate Tin	ne
	Create	Date	Last A Da	locess ite	0×0	000	Last M Tir	odified ne	Last M Da	odified ite	First (Cluster		File	Size	

1st long entry

Tip:

Figure 24: Example of Folder Entries for the long file name



For more detailed information see resource kits on Microsoft's web site http://www.microsoft.com/windows/reskits/webresources/default.asp or Microsoft Developers Network (MSDN) http://www.microsoft.com microsoft Developers Network (MSDN) http://www.microsoft.com

FAT32 Features

The following topics describe the FAT32 file system.

- File System Specifications on page 139
- Boot Sector and Bootstrap Modifications on page 139
- FAT Mirroring on page 143
- Partition Types on page 145

File System Specifications

FAT32 is a derivative of the File Allocation Table (FAT) file system that supports drives with over 2GB of storage. Because FAT32 drives can contain more than 65,526 clusters, smaller clusters are used than on large FAT16 drives. This method results in more efficient space allocation on the FAT32 drive.

The largest possible file for a FAT32 drive is 4GB minus 2 bytes.

The FAT32 file system includes four bytes per cluster within the file allocation table. Note that the high 4 bits of the 32-bit values in the FAT32 file allocation table are reserved and are not part of the cluster number.

Modifications	Description
Reserved Sectors	FAT32 drives contain more reserved sectors than FAT16 or FAT12 drives. The number of reserved sectors is usually 32, but can vary.
Boot Sector Modifications	Because a FAT32 BIOS Parameter Block (BPB), represented by the <i>BPB</i> structure, is larger than a standard BPB, the boot

Boot Sector and Bootstrap Modifications

Modifications	Description
	record on FAT32 drives is greater than 1 sector. In addition, there is a sector in the reserved area on FAT32 drives that contains values for the count of free clusters and the cluster number of the most recently allocated cluster. These values are members of the <i>BIGFATBOOTFSINFO</i> structure which is contained within this sector. These additional fields allow the system to initialize the values without having to read the entire file allocation table.
Root Directory	The root directory on a FAT32 drive is not stored in a fixed location as it is on FAT16 and FAT12 drives. On FAT32 drives, the root directory is an ordinary cluster chain. The A_BF_BPB_RootDirStrtClus member in the BPB structure contains the number of the first cluster in the root directory. This allows the root directory to grow as needed. In addition, the BPB_RootEntries member of BPB is ignored on a FAT32 drive.
Sectors Per FAT	The A_BF_BPB_SectorsPerFAT member of <i>BPB</i> is <i>always</i> zero on a FAT32 drive. Additionally, the A_BF_BPB_BigSectorsPerFat and A_BF_BPB_BigSectorsPerFatHi members of the updated BPB provide equivalent information for FAT32 media.

BPB (FAT32)

The BPB for FAT32 drives is an extended version of the FAT16/FAT12 BPB. It contains identical information to a standard BPB, but also includes several extra fields for FAT32 specific information.

This structure is implemented in Windows OEM Service Release 2 and later.

A	BF E	BPB	0	STRUC		
-	- <u>A</u>	BF	BPB	BytesPerSector	DW	?
	A	BF	BPB	SectorsPerCluster	DB	?
	A	BF	BPB	ReservedSectors	DW	?
	A	BF	BPB	NumberOfFATs	DB	?
	A	BF	BPB	RootEntries	DW	?
	A	ΒF	BPB	TotalSectors	DW	?
	A	BF	BPB	MediaDescriptor	DB	?
	A	BF	BPB	SectorsPerFAT	DW	?
	A	BF	BPB	SectorsPerTrack	DW	?
	A	BF	BPB	Heads	DW	?
	A	BF	BPB	HiddenSectors	DW	?
	A	BF	BPB	HiddenSectorsHigh	DW	?
	A	BF	BPB	BigTotalSectors	DW	?
	A	BF	BPB	BigTotalSectorsHigh	DW	?
	A	BF	BPB	BigSectorsPerFat	DW	?
	A	BF	BPB	BigSectorsPerFatHi	DW	?
	A	BF	BPB	ExtFlags	DW	?
	A	BF	BPB	FS Version	DW	?
	A	BF	BPB	RootDirStrtClus	DW	?

	A BF BPE	B RootDirStrtClusHi	DW	?		
	A BF BPE	3 FSInfoSec	DW	?		
	A BF BPE	3 ⁻ BkUpBootSec	DW	?		
	A BF BPE	- Reserved	DW	6	DUP	(?
А	BF BPB	ENDS				

A BF BPB BytesPerSector

The number of bytes per sector.

A_BF_BPB_SectorsPerCluster

The number of sectors per cluster.

A_BF_BPB_ReservedSectors

The number of reserved sectors, beginning with sector 0.

A_BF_BPB_NumberOfFATs

The number of File Allocation Tables.

A_BF_BPB_RootEntries

This member is ignored on FAT32 drives.

A_BF_BPB_TotalSectors

The size of the partition, in sectors.

A_BF_BPB_MediaDescriptor

The media descriptor. Values in this member are identical to standard BPB.

A_BF_BPB_SectorsPerFAT

The number of sectors per FAT.

Note: This member will always be zero in a FAT32 BPB. Use the values from **A_BF_BPB_BigSectorsPerFat** and **A_BF_BPB_BigSectorsPerFatHi** for FAT32 media.

A_BF_BPB_SectorsPerTrack

The number of sectors per track.

A_BF_BPB_Heads

The number of read/write heads on the drive.

A_BF_BPB_HiddenSectors

The number of hidden sectors on the drive.

A_BF_BPB_HiddenSectorsHigh

The high word of the hidden sectors value.

A_BF_BPB_BigTotalSectors

The total number of sectors on the FAT32 drive.

A_BF_BPB_BigTotalSectorsHigh

The high word of the FAT32 total sectors value.

A_BF_BPB_BigSectorsPerFat

The number of sectors per FAT on the FAT32 drive.

A_BF_BPB_BigSectorsPerFatHi

The high word of the FAT32 sectors per FAT value.

A_BF_BPBExtFlags

Flags describing the drive. Bit 8 of this value indicates whether or not information written to the active FAT will be written to all copies of the FAT. The low 4 bits of this value contain the 0-based FAT number of the Active FAT, but are only meaningful if bit 8 is set. This member can contain a combination of the following values.

Value	Description
BGBPB_F_ActiveFATMsk (000Fh)	Mask for low four bits.
BGBPB_F_NoFATMirror (0080h)	Mask indicating FAT mirroring state. If set, FAT mirroring is disabled. If clear, FAT mirroring is enabled.

* Bits 4-6 and 8-15 are reserved.

A_BF_BPB_FS_Version

The file system version number of the FAT32 drive. The high byte represents the major version, and the low byte represents the minor version.

A_BF_BPB_RootDirStrtClus

The cluster number of the first cluster in the FAT32 drive's root directory.

A_BF_BPB_RootDirStrtClusHi

The high word of the FAT32 starting cluster number.

A_BF_BPB_FSInfoSec

The sector number of the file system information sector. The file system info sector contains a *BIGFATBOOTFSINFO* structure. This member is set to 0FFFFh if there is no FSINFO sector. Otherwise, this value must be non-zero and less than the reserved sector count.

A_BF_BPB_BkUpBootSec

The sector number of the backup boot sector. This member is set to 0FFFFh if there is no backup boot sector. Otherwise, this value must be non-zero and less than the reserved sector count.

A_BF_BPB_Reserved

Reserved member.

BIGFATBOOTFSINFO (FAT32)

Contains information about the file system on a FAT32 volume. This structure is implemented in Windows OEM Service Release 2 and later.

```
BIGFATBOOTFSINFO STRUC

bfFSInf_Sig DD ?

bfFSInf_free_clus_cnt DD ?

bfFSInf_next_free_clus DD ?

bfFSInf_resvd DD 3 DUP (?)

BIGFATBOOTFSINFO ENDS
```

bfFSInf_Sig

The signature of the file system information sector. The value in this member is FSINFOSIG (0x61417272L).

bfFSInf_free_clus_cnt

The count of free clusters on the drive. Set to -1 when the count is unknown.

bfFSInf_next_free_clus

The cluster number of the cluster that was most recently allocated.

bfFSInf_resvd

Reserved member.

FAT Mirroring

On all FAT drives, there may be multiple copies of the FAT. If an error occurs reading the primary copy, the file system will attempt to read from the backup copies. On FAT16 and FAT12 drives, the first FAT is always the primary copy and any modifications will automatically be written to all copies. However, on FAT32 drives, FAT mirroring can be disabled and a FAT other than the first one can be the primary (or "active") copy of the FAT.

Mirroring is enabled by clearing bit 0x0080 in the **extdpb_flags** member of a FAT32 Drive Parameter Block (DPB) structure, **DPB**.

Mirroring	Description
When Enabled (bit 0x0080 clear)	With mirroring enabled, whenever a FAT sector is written, it will also be written to every other FAT. Also, a mirrored FAT sector can be read from any FAT.
	A FAT32 drive with multiple FATs will behave the same as FAT16 and FAT12 drives with multiple FATs. That is, the multiple FATs are backups of each other.
When Disabled (bit 0x0080 set)	With mirroring disabled, only one of the FATs is active. The active FAT is the one specified by bits 0 through 3 of the extdpb_flags member of <i>DPB</i> . The other FATs are ignored. Disabling mirroring allows better handling of a drive with a bad sector in one of the FATs. If a bad sector exists, access to the damaged FAT can be completely disabled. Then, a new FAT can be built in one of the inactive FATs and then made accessible by changing the active FAT value in extdpb_flags .

DPB (FAT32)

The DPB was extended to include FAT32 information. Changes are effective for Windows 95 OEM Service Release 2 and later.

DPB	STRUC		
	dpb drive	DB	?
	dpb_unit	DB	?
	dpb sector size	DW	?
	dpb_cluster_mask	DB	?
	dpb_cluster_shift	DB	?
	dpb_first_fat	DW	?
	dpb fat count	DB	?
	dpb root entries	DW	?
	dpb first sector	DW	?
	dpb_max_cluster	DW	?
	dpb fat size	DW	?
	dpb_dir_sector	DW	?
	dpb_reserved2	DD	?
	dpb media	DB	?
ifde	ef NOTFAT32		
	dpb_first_access	DB	?
else	9		
	dpb reserved	DB	?
endi	if —		
	dpb_reserved3	DD	?

```
dpb next free
                          DW
                                 ?
   dpb_free_cnt
                          DW
                                 ?
ifndef NOTFAT32
   extdpb free cnt hi DW
                                 ?
   extdpb_flags DW
extdpb_FSInfoSec DW
                                 ?
                                ?
   extdpb BkUpBootSec DW
                                ?
   extdpb first sector DD
                                ?
   extdpb_max_cluster DD
                                ?
   extdpb_fat_size
extdpb_root_clus
extdpb_next_free
                         DD ?
                         DD ?
                         DD
                                 ?
endif
```

DPB ENDS

dpb_drive

The drive number (0 = A, 1 = B, and so on).

dpb_unit

Specifies the unit number. The device driver uses the unit number to distinguish the specified drive from the other drives it supports.

dpb_sector_size

The size of each sector, in bytes.

dpb_cluster_mask

The number of sectors per cluster minus 1.

dpb_cluster_shift

The number of sectors per cluster, expressed as a power of 2.

dpb_first_fat

The sector number of the first sector containing the file allocation table (FAT).

dpb_fat_count

The number of FATs on the drive.

dpb_root_entries

The number of entries in the root directory.

dpb_first_sector

The sector number of the first sector in the first cluster.

dpb_max_cluster

The number of clusters on the drive plus 1. This member is undefined for FAT32 drives.

dpb_fat_size

The number of sectors occupied by each FAT. The value of zero indicates a FAT32 drive. Use the value in **extdpb_fat_size** instead.

dpb_dir_sector

The sector number of the first sector containing the root directory. This member is undefined for FAT32 drives.

dpb_reserved2

Reserved member. Do not use.

dpb_media

Specifies the media descriptor for the medium in the specified drive.

reserved

Reserved member. Do not use.
dpb_first_access

Indicates whether the medium in the drive has been accessed. This member is initialized to -1 to force a media check the first time this DPB is used.

dpb_reserved3

Reserved member. Do not use.

dpb_next_free

The cluster number of the most recently allocated cluster.

dpb_free_cnt

The number of free clusters on the medium. This member is 0FFFFh if the number is unknown.

extdpb_free_cnt_hi

The high word of free count.

extdpb_flags

Flags describing the drive. The low 4 bits of this value contain the 0-based FAT number of the Active FAT. This member can contain a combination of the following values.

Value	Description
BGBPB_F_ActiveFATMsk (000Fh)	Mask for low four bits.
BGBPB_F_NoFATMirror (0080h)	Do not mirror active FAT to inactive FATs.

Bits 4-6 and 8-15 are reserved.

extdpb_FSInfoSec

The sector number of the file system information sector. This member is set to 0FFFFh if there is no FSINFO sector. Otherwise, this value must be non-zero and less than the reserved sector count.

extdpb_BkUpBootSec

The sector number of the backup boot sector. This member is set to 0FFFFh if there is no backup boot sector. Otherwise, this value must be non-zero and less than the reserved sector count.

extdpb_first_sector

The first sector of the first cluster.

extdpb_max_cluster

The number of clusters on the drive plus 1.

extdpb_fat_size

The number of sectors occupied by the FAT.

extdpb_root_clus

The cluster number of the first cluster in the root directory.

extdpb_next_free

The number of the cluster that was most recently allocated.

Partition Types

The following are all the valid partition types and their corresponding values for use in the **Part_FileSystem** member of the *s_partition* structure.

Table 21: Partition Types

Value	Description
PART_UNKNOWN (00h)	Unknown
PART_DOS2_FAT (01h)	12-bit FAT
PART_DOS3_FAT (04h)	16-bit FAT. Partitions smaller than 32MB.
PART_EXTENDED (05h)	Extended MS-DOS Partition
PART_DOS4_FAT (06h)	16-bit FAT. Partitions larger than or equal to 32MB.
PART_DOS32 (0Bh)	32-bit FAT. Partitions up to 2047GB.
PART_DOS32X (0Ch)	Same as PART_DOS32 (0Bh), but uses Logical Block Address Int 13h extensions.
PART_DOSX13 (0Eh)	Same as PART_DOS4_FAT (06h), but uses Logical Block Address Int 13h extensions.
PART_DOSX13X (0Fh)	Same as PART_EXTENDED (05h), but uses Logical Block Address Int 13h extensions.

s_partition (FAT32)

```
s_partition STRUC

Part_BootInd DB ?

Part_FirstHead DB ?

Part_FirstSector DB ?

Part_FirstTrack DB ?

Part_FileSystem DB ?

Part_LastHead DB ?

Part_LastSector DB ?

Part_LastTrack DB ?

Part_LastTrack DB ?

Part_StartSector DD ?

Part_NumSectors DD ?

s partition ENDS
```

Part_BootInd

Specifies whether the partition is bootable or not. This value could be set to PART_BOOTABLE (80h), or PART_NON_BOOTABLE(00h). The first partition designated as PART_BOOTABLE is the boot partition. All others are not. Setting multiple partitions to PART_BOOTABLE will result in boot errors.

Part_FirstHead

The first head of this partition. This is a 0-based number representing the offset from the beginning of the disk. The partition includes this head.

Part_FirstSector

The first sector of this partition. This is a 1-based, 6-bit number representing the offset from the beginning of the disk. The partition includes this sector. Bits 0 through 5 specify the 6-bit value; bits 6 and 7 are used with the **Part_FirstTrack** member.

Part_FirstTrack

The first track of this partition. This is an inclusive 0-based, 10-bit number that represents the offset from the beginning of the disk. The high 2 bits of this value are specified by bits 6 and 7 of the **Part_FirstSector** member.

PartFileSystem

Specifies the file system for the partition.

Table 22: Acceptable values

Value	Description
PART_UNKNOWN(00h)	Unknown.
PART_DOS2_FAT(01h)	12-bit FAT.
PART_DOS3_FAT(04h)	16-bit FAT. Partition smaller than 32MB.
PART_EXTENDED(05h)	Extended MS-DOS Partition.
PART_DOS4_FAT(06h)	16-bit FAT. Partition larger than or equal to 32MB.
PART_DOS32(0Bh)	32-bit FAT. Partition up to 2047GB.
PART_DOS32X(0Ch)	Same as PART_DOS32(0Bh), but uses Logical Block Address Int 13h extensions.
PART_DOSX13(0Eh)	Same as PART_DOS4_FAT(06h), but uses Logical Block Address Int 13h extensions.
PART_DOSX13X(0Fh)	Same as PART_EXTENDED(05h), but uses Logical Block Address Int 13h extensions.

Part_LastHead

The last head of the partition. This is a 0-based number that represents the offset from the beginning of the disk. The partition includes the head specified by this member.

Part_LastSector

The last sector of this partition. This is a 1-based, 6-bit number representing offset from the beginning of the disk. The partition includes the sector specified by this member. Bits 0 through 5 specify the 6-bit value; bits 6 and 7 are used with the **Part_LastTrack** member.

Part_LastTrack

The last track of this partition. This is a 0-based, 10-bit number that represents offset from the beginning of the disk. The partition includes this track. The high 2 bits of this value are specified by bits 6 and 7 of the **Part_LastSector** member.

Part_StartSector

Specifies the 1-based number of the first sector on the disk. This value may not be accurate for extended partitions. Use the **Part_FirstSector** value for extended partitions.

Part_NumSectors

The 1-based number of sectors in the partition.

Note:

Values for head and track are 0-based. Sector values are 1-based. This structure is implemented in Windows OEM Service Release 2 and later.

File Recovery Process

File recovery process can be briefly described as drive or folder scanning to find deleted entries in Root Folder (FAT) or Master File Table (NTFS) then for the particular deleted entry, defining clusters chain to be recovered and then copying contents of these clusters to the newly created file.

Different file systems maintain their own specific logical data structures, however basically each file system:

- Has a list or catalogue of file entries, so we can iterate through this list and entries, marked as deleted
- Keeps for each entry a list of data clusters, so we can try to find out set of clusters composing the file

After finding out the proper file entry and assembling set of clusters, composing the file, read and copy these clusters to another location.

Step by Step with examples:

- *Disk scan for deleted entries* on page 148
- *Define clusters chain for the deleted entry* on page 151
- Clusters chain recovery for the deleted entry on page 153

However, not every deleted file can be recovered, there are some assumptions, for sure:

- First, we assume that the file entry still exists (not overwritten with other data). The less the files have been created on the drive where the deleted file was resided, the more chances that space for the deleted file entry has not been used for other entries.
- Second, we assume that the file entry is more or less safe to point to the proper place where file clusters are located. In some cases (it has been noticed in Windows XP, on large FAT32 volumes) operating system damages file entries right after deletion so that the first data cluster becomes invalid and further entry restoration is not possible.
- Third, we assume that the file data clusters are safe (not overwritten with other data). The less the write operations have been performed on the drive where deleted file was resided, the more chances that the space occupied by data clusters of the deleted file has not been used for other data storage.

Important:

As general advices after data loss:

1. DO NOT WRITE ANYTHING ONTO THE DRIVE CONTAINING YOUR IMPORTANT DATA THAT YOU HAVE JUST DELETED ACCIDENTALLY! Even data recovery software installation could spoil your sensitive data. If the data is really important to you and you do not have another logical drive to install software to, take the whole hard drive out of the computer and plug it into another computer where data recovery software has been already installed or use recovery software that does not require installation, for example recovery software which is capable to run from bootable floppy.

2. DO NOT TRY TO SAVE ONTO THE SAME DRIVE DATA THAT YOU FOUND AND TRYING TO RECOVER! When saving recovered data onto the same drive where sensitive data is located, you can intrude in process of recovering by overwriting FAT/MFT records for this and other deleted entries. It's better to save data onto another logical, removable, network or floppy drive.

Disk scan for deleted entries

Disk Scanning is a process of low-level enumeration of all entries in the *Root Folders* on FAT12, FAT16, FAT32 or in Master File Table (MFT) on NTFS, NTFS5. The goal is to find and display deleted entries.

In spite of different file/folder entry structure for the different file systems, all of them contain basic file attributes like name, size, creation and modification date/time, file attributes, existing/deleted status, etc...

Given that a drive contains root file table and any file table (MFT, root folder of the drive, regular folder, or even deleted folder) has location, size and predefined structure, we can scan it from the beginning to the end checking each entry, if it's deleted or not and then display information for all found deleted entries.



Note:

Deleted entries are marked differently depending on the file system. For example, in FAT any deleted entry, file or folder has been marked with ASCII symbol **229** (**OxE5**) that becomes first symbol of the *entry*. On NTFS deleted entry has a special attribute in file header that points whether the file has been deleted or not.

Example of scanning folder on FAT16

```
1. Existing folder MyFolder entry (long entry and short entry)
Offset
       0 1 2 3 4 5 6 7 8 9 A B C D E F
_____
                                  _____
0003EE20 41 4D 00 79 00 46 00 6F 00 6C 00 0F 00 09 64 00 AM.y.F.o.l...d.
0003EE40 4D 59 46 4F 4C 44 45 52 20 20 20 10 00 4A C4 93 MYFOLDER ..JA"
0003EE50 56 2B 56 2B 00 00 C5 93 56 2B 02 00 00 00 00 0 V+V+..A<sup>V</sup>V+....
2. Deleted file MyFile.txt entry (long entry and short entry)
0003EE60 E5 4D 00 79 00 46 00 69 00 6C 00 0F 00 BA 65 00 aM.y.F.i.l...?e.
0003EE70 2E 00 74 00 78 00 74 00 00 00 00 00 FF FF FF FF
                                                 ..t.x.t....yyyy
0003EE80 E5 59 46 49 4C 45 20 20 54 58 54 20 00 C3 D6 93 aYFILE TXT .AO"
0003EE90 56 2B 56 2B 00 00 EE 93 56 2B 03 00 33 B7 01 00 V+V+..i<sup>v</sup>V+..3<sup>.</sup>.
4. Existing file Setuplog.txt entry (the only short entry)
0003EEA0 53 45 54 55 50 4C 4F 47 54 58 54 20 18 8C F7 93
                                                 SETUPLOGTXT
.??"
                            47 2B 07 00 8D 33 03 00
0003EEB0
        56 2B 56 2B 00 00 03 14
                                                 V+V+....G+...?3...
0003EEC0
        . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . .
```

This folder contains 3 entries, one of them is deleted. First entry is an existing folder **MyFolder**. Second one is a deleted file **MyFile.txt** Third one is an existing file **Setuplog.txt**.

First symbol of the deleted file entry is marked with **E5** symbol, so Disk Scanner can assume that this entry has been deleted.

Example of scanning folder on NTFS5 (Windows 2000):

For our drive we have input parameters:

- Total Sectors 610406
- Cluster size 512 bytes
- One Sector per Cluster
- MFT starts from offset 0x4000, non-fragmented
- MFT record size 1024 bytes
- MFT Size 1968 records

Thus we can iterate through all 1968 MFT records, starting from the absolute offset 0x4000 on the volume looking for the deleted entries. We are interested in MFT entry 57 having offset 0x4000 + 57 * 1024 = 74752 = 0x12400 because it contains our recently deleted file "My Presentation.ppt"

Below MFT record number 57 is displayed:

Offset 0 1 2 3 4 5 6 7 8 9 A B C D E F

00012400	46	49	4C	45	2A	00	03	00	9C	74	21	03	00	00	00	00	FILE*?
L !	47	~~	00	~~	20	~~	~~	00	D 0	01	~~	~~	~~	~ 4	~~	~~	
00012410	4/	00	02	00	30	00	00	00		01	00	00	00	04	00	00	G00
00012420	1.0	00	00	00	00	00	00	00	05	00	03	00	00	00	00	00	•••••
00012430	10	00	00	00	10	00	00	00	20	00 E 2	00	00 72	10	00	00 C1	00	
00012440	40	20	00 2D	00 D0			00	00	20	22	20	AS NO	10	Е I 101		01	
00012450	20	50		72	48	E9	CU C1	01	20	BF	20	AU	19	E.T		01	.U+OHEA.A? .HA.
00012460	20	53		A3	10	E.T	00	01	20	00	00	00	00	00	00	00	SI?.NA
00012470	00	00	00	00	00	00	00	00	00	00	00	00	02	01	00	00	• • • • • • • • • • • • • • • • • •
00012480	20	00	00	00	70	00	00	00	00	00	00	00	00	00	00	00	· · · · · · · · · · · · · · · · · · ·
00012490	50	00	00	00	10	00	00	00	00	00	00	00	00	00	05	00	7
000124A0	3A 20	50 52	00	00 73	10 10	00 101		00	20	50	00	72	10	00 101	05	00	
00012460	20	22	םם מח	AS NO	10 10	ЕТ 1		01	20	22	עע סס	AS NO	10	면 1 단 1		01	SIT.MA. SIT.MA.
00012400	20	22		A3	10	F 1		01	20	22		A3	10	F I		01	SI?.MA. SI?.MA.
00012400	20	00	00	00	00	00	00	00	00	00	40	00	50	00	50	00	 м v р
000124E0	52	00	45	00	53	00	00 75	00	31	02	-1D 217	00	50	00	50	00	
00012420	54	00	- 3	00	८ म	00	7 म	00	30	00		00	80	00	0.0	00	Tion0 ^
00012510	00	00	00	00	00	00	02	00	68	00	00	00	18	00	01	00	h
00012510	05	00	00	00	00	00	05	00	20	53	סס	A 3	18	00 F 1	C1	01	SY2 nA
00012520	20	53	סס	23 23	18	00 ਸ1	C1	01	20	53	סס	<u>23</u>	18	ਸ 1 ਜ	C1	01	SY2 nA SY2 nA
00012540	20	53	סס	A3	18	 ਸ1	C1	01	00	00	00	00	00	00	00	00	SY2 nA
00012550	00	00	00	00	00	00	00	00	20	00	00	00	00	00	00	00	
00012560	13	01	4D	00	79	00	20	00	50	00	72	00	65	00	73	00	
M.v.	.P.r	.e.s	з.								. –						
00012570	65	00	6E	00	74	00	61	00	74	00	69	00	6F	00	6E	00	e.n.t.a.t.i.o.n.
00012580	2E	00	70	00	70	00	74	00	80	00	00	00	48	00	00	00	
p.p.t.	^ I	H															
00012590	01	00	00	00	00	00	04	00	00	00	00	00	00	00	00	00	
000125A0	6D	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00	
m	@																
000125B0	00	DC	00	00	00	00	00	00	00	DC	00	00	00	00	00	00	
.U	U.																
000125C0	00	DC	00	00	00	00	00	00	31	6E	EB	C4	04	00	00	00	.UlneA
000125D0	FF	FF	FF	FF	82	79	47	11	00	00	00	00	00	00	00	00	уууу,уG
000125E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000125F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	03	00	
00012600	00		00	00	00	00	00	00	00	00	00	00	00	00	00	00	

MFT Record has pre-defined structure. It has a set of attributes defining any file of folder parameters.

MFT Record begins with standard File Record Header (first bold section, offset 0x00):

- "FILE" identifier (4 bytes)
- Offset to update sequence (2 bytes)
- Size of update sequence (2 bytes)
- \$LogFile Sequence Number (LSN) (8 bytes)
- Sequence Number (2 bytes)
- Reference Count (2 bytes)
- Offset to Update Sequence Array (2 bytes)
- Flags (2 bytes)
- Real size of the FILE record (4 bytes)
- Allocated size of the FILE record (4 bytes)
- File reference to the base FILE record (8 bytes)
- Next Attribute Id (2 bytes)

The most important information for us in this block is a file state: deleted or in-use. If Flags (in red color) field has bit 1 set, it means that file is in-use. In our example it is zero, i.e. file is deleted.

Starting from 0x48, we have **Standard Information** Attribute (second bold section):

- File Creation Time (8 bytes)
- File Last Modification Time (8 bytes)
- File Last Modification Time for File Record (8 bytes)
- File Access Time for File Record (8 bytes)
- DOS File Permissions (4 bytes) 0x20 in our case Archive Attribute

Following standard attribute header, we have **File Name** Attribute belonging to DOS name space, short file names, (third bold section, offset 0xA8) and again following standard attribute header, we have **File Name** Attribute belonging to Win32 name space, long file names, (third bold section, offset 0x120):

- File Reference to the Parent Directory (8 bytes)
- File Modification Times (32 bytes)
- Allocated Size of the File (8 bytes)
- Real Size of the File (8 bytes)
- Flags (8 bytes)
- Length of File Name (1 byte)
- File Name Space (1 byte)
- File Name (Length of File Name * 2 bytes)

In our case from this section we can extract file name, "My Presentation.ppt", File Creation and Modification times, and Parent Directory Record number.

Starting from offset 0x188, there is a non-resident Data attribute (green section).

- Attribute Type (4 bytes) (e.g. 0x80)
- Length including header (4 bytes)
- Non-resident flag (1 byte)
- Name length (1 byte)
- Offset to the Name (2 bytes)
- Flags (2 bytes)
- Attribute Id (2 bytes)
- Starting VCN (8 bytes)
- Last VCN (8 bytes)
- Offset to the Data Runs (2 bytes)
- Compression Unit Size (2 bytes)
- Padding (4 bytes)
- Allocated size of the attribute (8 bytes)
- Real size of the attribute (8 bytes)
- Initialized data size of the stream (8 bytes)
- Data Runs ...

In this section we are interested in Compression Unit size (zero in our case means non-compressed), Allocated and Real size of attribute that is equal to our file size (0xDC00 = 56320 bytes), and Data Runs (see the next topic).

Define clusters chain for the deleted entry

To define clusters chain we need to scan drive, going through one by one all file (NTFS) clusters or free (FAT) clusters belonging (presumably) to the file until we reach the file size equals to the total size of the selected clusters. If the file is fragmented, clusters chain will be composed of several extents in case of NTFS or we take clusters bypassing occupied ones in case of FAT.

Location of these clusters can vary depending on file system. For example, file deleted on FAT volume has its first cluster in its Root entry, the other clusters can be found in File Allocation Table. On NTFS each file

has _DATA_ attribute that describes "data runs". Disassembling data runs to "extents" for each extent we have start cluster offset and number of clusters in extent, so enumerating extents, we can compose file's cluster chain.

You can try to define clusters chain manually, using low-level disk editors, however it's much simpler to use data recovery tools, like Active@ UNDELETE.

Example of defining clusters chain on FAT16

Lets continue examine an example for deleted file **MyFile.txt** from the previous topic.

The folder, we scanned before contains a record for this file:

Offset	0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F	
0003EE60	E5	4D	00	79	00	46	00	69	00	6C	00	0F	00	BA	65	00	aM.y.F.i.l?e.
0003EE70	2E	00	74	00	78	00	74	00	00	00	00	00	FF	FF	FF	FF	t.x.tyyyy
0003EE80	E5	59	46	49	4C	45	20	20	54	58	54	20	00	C3	D6	93	aYFILE TXT .AO"
0003EE90	56	2B	56	2B	00	00	EE	93	56	2B	03	00	33	B7	01	00	V+V+i"V+3·

We can calculate size of the deleted file based on root entry structure. Last four bytes are 33 B7 01 00 and converting them to decimal value (changing bytes order), we get 112435 bytes. Previous 2 bytes (03 00) are the number of the first cluster of the deleted file. Repeating for them the conversion operation, we get number 03 - this is the start cluster of the file.

What we can see in the File Allocation Table at this moment?

Offset	0	1	2	3	4	5	6	7	8	9	A	В	С	D	Ε	F	
00000200	F8 09	FF 00	FF 0A	FF 00	FF 0B	FF 00	00 0C	00	00 0D	00	00 FF	00 FF	00	00	08 00	00	оуууууvv
00000220	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	••••••••••••••••

Zeros! And it is good in our case - it means that these clusters are free, i.e. most likely our file was not overwritten by other file's data. Now we have chain of clusters 3, 4, 5, 6 and ready to recover it.

Some explanations:

- we started looking from offset 6 because each cluster entry in FAT16 takes 2 bytes, our file starts from 3rd cluster, i.e. 3*2=6.
- we considered 4 clusters because cluster size on our drive is 32 Kb, our file size is 112, 435 bytes, i.e.
 3clusters*32Kb = 96Kb plus a little bit more.
- we assumed that this file was not fragmented, i.e. all clusters were located consequently. We need 4 clusters, we found 4 free consecutive clusters, so this assumption sounds reasonable, although in real life it may be not true.



Note:

There are a lot of cases where the file's data can not be successfully recovered, because clusters chain can not be defined. Most of them occur when you write another data (files, folders) on the same drive where deleted file located. You'll see these warnings while recovering data using, for example Active@ UNDELETE.

Example of defining clusters chain on NTFS

When recovering on NTFS part of DATA attribute called Data Runs give us location about file clusters. In most cases DATA attribute is stored inside MFT record, so if we found MFT record for the deleted file, most likely we'll be able to determine cluster's chain.

In example below DATA attribute is marked with a green color. Data Runs inside, marked as Bold.

Offset	0	1	2	3	4	5	6	7	8	9	A	В	С	D	Е	F	
00012580	2E	00	70	00	70	00	74	00	80	00	00	00	48	00	00	00	p.p.t.€H
00012590	01	00	00	00	00	00	04	00	00	00	00	00	00	00	00	00	
000125A0	6D	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00	m@
000125B0	00	DC	00	00	00	00	00	00	00	DC	00	00	00	00	00	00	.UU
000125C0	00	DC	00	00	00	00	00	00	31	6E	EB	C4	04	00	00	00	.U lneA
000125D0	FF	FF	FF	FF	82	79	47	11	00	00	00	00	00	00	00	00	уууу,уG

Data Runs need to be decrypted. First byte (0x31) shows how many bytes are allocated for the length of the run (0x1 in our case) and for the first cluster offset (0x3 in our case). Next, we take one byte (0x6E) that points to the length of the run. Next, we pick up 3 bytes pointing to the start cluster offset (0xEBC404). Changing bytes order we get first cluster of the file 312555 (equals 0x04C4EB). Starting from this cluster we need to pick up 110 clusters (equals 0x6E). Next byte (0x00) tells us that no more data runs exist. Our file is not fragmented, so we have the only one data run.

Lets check, isn't there enough information about the file data? Cluster size is 512 bytes. We have 110 clusters, 110*512 = 56320 bytes. Our file size was defined as 56320 bytes, so we have enough information now to recover the file clusters.

Important:

1. DO NOT WRITE ANYTHING ONTO THE DRIVE CONTAINING YOUR IMPORTANT DATA THAT YOU HAVE JUST DELETED ACCIDENTALLY!Even data recovery software installation could spoil your sensitive data. If the data is really important to you, and you do not have another logical drive to install software to, take whole hard drive out of the computer and plug into another computer where data recovery software has been already installed.

2. DO NOT TRY TO SAVE ONTO THE SAME DRIVE DATA THAT YOU FOUND AND TRYING TO RECOVER!While saving recovered data onto the same drive where sensitive data was located, you can intrude in process of recovering by overwriting FAT records for this and other deleted entries. It's better to save data onto another logical, removable, network or floppy drive.

Clusters chain recovery for the deleted entry

After clusters chain is defined, automatically or manually, the only task left is to read and save contents of the defined clusters to another place verifying their contents.

We have a chain of clusters; we can calculate each cluster offset from the beginning of the drive, using standard formulas. After that we copy amount of data equals to the cluster size, starting from the calculated offset into the newly created file. For the last one we copy not all cluster, but reminder from the file size minus number of copied clusters multiplied by cluster size.

Formulas for calculating cluster offset could vary depending on file system.

To calculate, for example, offset of the cluster for FAT we need to know:

- Boot sector size
- Number of FAT supported copies
- Size of one copy of FAT
- Size of main root folder
- Number of sectors per cluster
- Number of bytes per sector

On the NTFS, we have linear space so we can calculate cluster offset simply as cluster number multiplied by cluster size.

Example of recovery clusters chain on FAT16

Lets continue examine an example for deleted file **MyFile.txt** from the previous topics.

By now we have chain of clusters 3, 4, 5, 6 ready for recovering. Our cluster consists of 64 sectors, sector size is 512 bytes, so cluster size is: 64*512 = 32,768 bytes = 32 Kb First data sector is 535 (we have 1 boot sector, plus 2 copies of FAT by 251 sectors each, plus root folder 32 sectors, total 534 occupied by system data sectors). Clusters 0 and 1 do not exist, so first data cluster is 2. Cluster number 3 is next to cluster 2, i.e. is located 64 sectors behind the first data sector (535). i.e. 535 + 64 = 599 sector, equal offset of 306,668 byte from the beginning of the drive (0x4AE00).

With a help of low-level disk editor on the disk we can see our data starting with offset 0x4AE00, or 3 cluster, or 599 sector:

Offset	0	1	2	3	4	5	6	7	8	9	A	В	С	D	Ε	F	
0004AE00	47	55	49	20	6D	6F	64	65	20	53	65	74	75	70	20	68	GUI mode Setup h
0004AE10	61	73	20	73	74	61	72	74	65	64	2E	0 D	0A	43	ЗA	5C	as startedC:\
0004AE20	57	49	4E	4E	54	5C	44	72	69	76	65	72	20	43	61	63	WINNT\Driver Cac

All we need to do is just copy 112,435 bytes starting from this place because clusters chain is consecutive. If it was not - we would need to re-calculate offset of each found cluster, and copy 3 times by 64*512 = 32768 bytes starting from each cluster offset, and then from the last cluster copy reminder: 14,131 bytes that is calculated as 112,435 bytes - (3 * 32768 bytes).

Example of recovery clusters chain on NTFS

In our example we just need to pick up 110 clusters starting from the cluster 312555.

Cluster size is 512 byte, so the offset of the first cluster would be **512** * **312555** = **160028160** = **0x0989D600**

Offset	0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F	
0989D600	D0	CF	11	Е0	A1	B1	1A	E1	00	00	00	00	00	00	00	00	ÐÏ.à;±.á
0989D610	00	00	00	00	00	00	00	00	ЗE	00	03	00	FΕ	FF	09	00	þÿ
0989D620	06	00	00	00	00	00	00	00	00	00	00	00	01	00	00	00	
0989D630	69	00	00	00	00	00	00	00	00	10	00	00	6B	00	00	00	ik
0989D640	01	00	00	00	FΕ	FF	FF	FF	00	00	00	00	6A	00	00	00	þÿÿÿj
0989D650	FF	FF	FF	$\mathbf{F}\mathbf{F}$	FF	FF	FF	FF	FF	$\mathbf{F}\mathbf{F}$	FF	FF	FF	FF	FF	FF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u>

Here is our data. What's left to do is just reading from this point 110 clusters (56320 bytes) and then copy them to another location. Data recovery is complete now.



Important:

DO NOT SAVE ONTO THE SAME DRIVE DATA THAT YOU FOUND AND TRYING TO RECOVER! process of recovering by overwriting FAT records for this and other deleted entries. It's better to save data onto another logical, removable, network or floppy drive.

Partition Recovery Process

System Boot Process

In some cases, the first indication of a problem with hard drive data is a refusal of the machine to perform a bootstrap startup. For the machine to be able to start properly, the following conditions must apply:

- Master Boot Record (MBR) exists and is safe
- Partition Table exists and contains at least one active partition

If the above is in place, executable code in the MBR selects an active partition and passes control there, so it can start loading the standard files (COMMAND.COM, NTLDR, ...) depending on the file system type on that partition.

If these files are missing or corrupted it will be impossible for the OS to boot - if you have ever seen the famous "NTLDR is missing ..." error, you understand the situation.

When using Active@ UNDELETE, the recovery software accesses the damaged drive at a low level, bypassing the standard system boot process (this is the same as if you instructed the computer to boot from another hard drive). Once the computer is running in this recovery environment, it will help you to see all other files and directories on the drive and allow you to copy data to a safe place on another drive.

Partition Visibility

A more serious situation exists if your computer will start and cannot see a drive partition or physical drive (see Note below). For the partition or physical drive to be visible to the Operating System the following conditions must apply:

- Partition/Drive can be found via Partition Table
- Partition/Drive boot sector is safe

If the above conditions are true, the OS can read the partition or physical drive parameters and display the drive in the list of the available drives.

If the file system is damaged (Root, FAT area on FAT12/FAT16/FAT32, or system MFT records on NTFS) the drive's content might not be displayed and we might see errors like "MFT is corrupted", or "Drive is invalid" ... If this is the case it is less likely that you will be able to restore your data. Do not despair, as there may be some tricks or tips to display some of the residual entries that are still safe, allowing you to recover your data to another location.

Partition recovery describes two things:

- 1. Physical partition recovery. The goal is to identify the problem and write information to the proper place on the hard drive so that the partition becomes visible to the OS again. This can be done using manual Disk Editors along with proper guidelines or using recovery software, designed specifically for this purpose. *Active@ Partition Recovery* software implements this approach.
- **2. Virtual partition recovery.** The goal is to determine the critical parameters of the deleted/damaged/ overwritten partition and render it open to scanning in order to display its content. This approach can be applied in some cases when physical partition recovery is not possible (for example, partition boot sector is dead) and is commonly used by recovery software. This process is almost impossible to implement it manually. *Active@ UNDELETE, Active@ UNERASER* software both implement this approach.



Note: If your computer has two operating systems and you choose to start in Windows 95/98 or ME, these operating systems cannot see partitions that are formatted for NTFS. This is normal operation for these operating systems. To view NTFS partitions, you must be in a Windows NT/2000/XP environment.

Other Partition Recovery Topics

These topics related to the recovery of partitions apply to any file system:

- Damaged MBR on page 156
- Partition is deleted or Partition Table is damaged on page 158
- Partition Boot Sector is damaged on page 160
- Missing or Corrupted System Files on page 163

For these topics the following disk layout will be used:

🖀 Disk Administrat	or				_ 🗆 ×
Partition <u>T</u> ools <u>V</u> iew	w <u>O</u> ptions	<u>H</u> elp			
					<u></u>
🖃 Disk 0	C:	H:	D:	E:	
4605 MB	NTFS 2502 MB	FAT 298 MB	FAT 102 MB	NTFS 102 MB	Free Spac 1608 MB
Primary partition	n 📃 Log	ical drive			
Free space in extend	ed partition	1600 MB			

The figure shows a system with two primary partitions (C:(NTFS) and H:(FAT)) and one extended partition having two logical drives (D: (FAT) and E:(NTFS))

Damaged MBR

The Master Boot Record (MBR) will be created when you create the first partition on the hard disk. It is very important data structure on the disk. The Master Boot Record contains the Partition Table for the disk and a small amount of executable code for the boot start. The location is always the first sector on the disk.

The first 446 (0x1BE) bytes are MBR itself, the next 64 bytes are the Partition Table, the last two bytes in the sector are a signature word for the sector and are always 0x55AA.

For our disk layout we have MBR:

```
Physical Sector: Cyl 0, Side 0, Sector 1
           33 C0 8E D0 BC 00 7C FB 50 07 50 1F FC BE 1B 7C
000000000
                                                               3AZ??.|
uP.P.u?.|
000000010
           BF 1B 06 50 57 B9 E5 01 F3 A4 CB BE BE 07 B1 04
                                                               ?..PW?a.o¤E??.
±.
000000020
           38 2C 7C 09 75 15 83 C6 10 E2 F5 CD 18 8B 14 8B
8, |.u.??.aoI.<.<
000000030 EE 83 C6 10 49 74 16 38
                                    2C 74 F6 BE 10 07 4E AC
i??.It.8,to?..N¬
000000040 3C 00 74 FA BB 07 00 B4
                                    OE CD 10 EB F2 89 46 25
                                                               <.tu»..?.I.eo%F
000000050 96 8A 46 04 B4 06 3C 0E
                                    74 11 B4 0B 3C 0C 74 05
SF.?.<.t.?.<.t.
00000060
           3A C4 75 2B 40 C6 46 25 06 75 24 BB AA 55 50 B4
                                                               :Au+@?F%.u$»?
UP?
```

000000070	41	CD	13	58	72	16	81	FB	55	AA	75	10	F6	C1	01	74	AI.Xr.?uU?
u.oA.t 000000080 f 2	0в	8A	ΕO	88	56	24	C7	06	A1	06	ΕB	1E	88	66	04	BF	.Sa?V\$C.?.e.?
000000090 v # < N	0A	00	В8	01	02	8B	DC	33	С9	83	FF	05	7F	03	8B	4E	? <u3e?< td=""></u3e?<>
000000000 >?}U	25	03	4E	02	CD	13	72	29	ΒE	46	07	81	3E	FE	7D	55	%.N.I.r)?F.?
0000000B0 ou??'.e	AA	74	5A	83	ΕF	05	7F	DA	85	F6	75	83	ΒE	27	07	EB	?tZ?i.#U
0000000C0 0000000D0	8A D5	98 4F	91 74	52 E4	99 33	03 C0	46 CD	08 13	13 EB	56 B8	0A 00	E8 00	12 00	00 00	5A 00	EB 00	S?'R [™] .FV.eZe
0000000E0	56	33	F6	56	56	52	50	06	53	51	ΒE	10	00	56	8B	F4	
00000000F0 d.r	50	52	В8	00	42	8A	56	24	CD	13	5A	58	8D	64	10	72	PR?.BSV\$I.ZX?
000000100 o^AetT	0A	40	75	01	42	80	C7	02	E2	F7	F8	5E	C3	ΕB	74	49	.@u.B€C.a?
000000110 partition	6E	76	61	6C	69	64	20	70	61	72	74	69	74	69	6F	6E	nvalid
000000120 loa	20	74	61	62	6C	65	00	45	72	72	6F	72	20	6C	6F	61	table.Error
000000130 s	64	69	6E	67	20	6F	70	65	72	61	74	69	6E	67	20	73	ding operating
000000140 op	79	73	74	65	6D	00	4D	69	73	73	69	6E	67	20	6F	70	ystem.Missing
000000150 system	65	72	61	74	69	6E	67	20	73	79	73	74	65	6D	00	00	erating
000000160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000170	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000180 <u.w<ob< td=""><td>00 5</td><td>00</td><td>00</td><td>8B</td><td>FC</td><td>1E</td><td>57</td><td>8B</td><td>F5</td><td>СВ</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td></td></u.w<ob<>	00 5	00	00	8B	FC	1E	57	8B	F5	СВ	00	00	00	00	00	00	
000000190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001B0 €.	00	00	00	00	00	00	00	00	A6	34	1F	ΒA	00	00	80	01	· · · · · · · ¦ 4 . ? · ·
0000001C0 >?@2N	01	00	07	FΕ	7F	3E	3F	00	00	00	40	32	4E	00	00	00	?#
0000001D0 d#2N.¦P	41	ЗF	06	FΕ	7F	64	7F	32	4E	00	A6	50	09	00	00	00	A?.?#
0000001E0 W.fa8	41	65	ΟF	FΕ	BF	4A	25	83	57	00	66	61	38	00	00	00	Ae.??J%?
0000001F0	00	00 .U	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	

What will happen if the first sector has been damaged (by virus, for example)?

Lets overwrite the first 16 bytes with zeros.

 000000000
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00
 00

When we try to boot after hardware testing procedures, we see just blank screen without any messages. It means the piece of code at the beginning of the MBR could not be executed properly. That's why even

error messages could not be displayed. However, if we boot from the floppy, we can see FAT partition, files on it and we are able to perform standard operations like file copy, program execution... It happens because in our example only part of the MBR has been damaged which does not allow the system to boot properly. However, the partition table is safe and we can access our drives when we boot from the operating system installed on the other drive.

What will happen if sector signature (last word 0x55AA) has been removed or damaged?

Lets write zeros to the location of sector signature.

When we try to boot now, we see an error message like "Operating System not found".

Thus the first thing if computer does not boot is to run Disk Viewer and check the first physical sector on HDD, whether it looks like valid MBR or not:

- · check, may be it's filled up with zeros or any other single character
- check whether error messages (like you can see above "Invalid partition table"...) are present or not
- check whether disk signature (0x55AA) is present or not

The simplest way to repair or re-create MBR is to run Microsoft's standard utility called FDISK with a parameter **/MBR**, like

A:\> FDISK.EXE /MBR

FDISK is a standard utility included in MS-DOS, Windows 95, 98, ME.

If you have Windows NT / 2000 / XP, you can boot from start-up floppy disks or CD-ROM, choose repair option during setup, and run **Recovery Console**. When you are logged on, you can run **FIXMBR** command to fix MBR.

Also you can use third party MBR recovery software or if you've created MBR backup, restore it from there (Active@ Partition Recovery has such capabilities).

What will happen if the first sector is bad/unreadable?

Most likely we'll get the same black screen, which we got when trying to boot. When you try to read it using Disk Viewer/Editor you should get an error message saying that sector is unreadable. In this case recovery software is unable to help you to bring HDD back to the working condition, i.e. physical partition recovery is not possible. The only thing that can be done is to scan and search for partitions (i.e. perform virtual partition recovery), and in case if something is found - display them and give the user an opportunity to save important data to another location. Software, like Active@ UNDELETE, Active@ UNDELETE, Active@ UNDELETE, will help you here.

Partition is deleted or Partition Table is damaged

The information about primary partitions and extended partition is contained in the Partition Table, a 64byte data structure, located in the same sector as the Master Boot Record (cylinder 0, head 0, sector 1). The Partition Table conforms to a standard layout, which is independent of the operating system. The last two bytes in the sector are a signature word for the sector and are always 0x55AA.

For our disk layout we have Partition Table:

```
Physical Sector: Cyl 0, Side 0, Sector 1
0000001B0
€
```

80 01

0000001C0 01 00 07 FE 7F 3E 3F 00 00 00 40 32 4E 00 00 00 ...?# >?...@2N.... 0000001D0 41 3F 06 FE 7F 64 7F 32 4E 00 A6 50 09 00 00 00 A?.?# d#2N.\P..... 0000001E0 41 65 0F FE BF 4A 25 83 57 00 66 61 38 00 00 00 Ae.??J%? W.fa8... 0000001F0 00 00 00 00 00 00 00 00 00 00 00 00 55 AAU?

We can see three existing and one empty entries:

- Partition 1, offset 0x01BE (446)
- Partition 2, offset 0x01CE (462)
- Partition 3, offset 0x01DE (478)
- Partition 4 empty, offset 0x01EE (494)

Each Partition Table entry is 16 bytes long, making a maximum of four entries available. Each partition entry has fields for Boot Indicator (BYTE), Starting Head (BYTE), Starting Sector (6 bits), Starting Cylinder (10 bits), System ID (BYTE), Ending Head (BYTE), Ending Sector (6 bits), Ending Cylinder (10 bits), Relative Sector (DWORD), Total Sectors (DWORD).

Thus the MBR loader can assume the location and size of partitions. MBR loader looks for the "active" partition, i.e. partition that has Boot Indicator equals 0x80 (the first one in our case) and passes control to the partition boot sector for further loading.

Lets consider the situations which cause computer to hang up while booting or data loss.

1. What will happen if no partition has been set to the Active state (Boot Indicator=0x80)?

Lets remove Boot Indicator from the first partition:

000001в0															00	01	
000001C0	01	00	07	FΕ	7F	ЗE	ЗF	00	00	00	40	32	$4 \mathrm{E}$	00	00	00	?#>?@2N

When we try to boot now, we see an error message like "Operating System not found". It means that the loader cannot determine which partition is system and active to pass control to.

2. What will happen if partition has been set to the Active state (Boot Indicator=0x80) but there are no system files on that partition?

(it could happen if we had used for example FDISK and selected not the proper active partition).

Loader will try to boot from there, fails, try to boot again from other devices like floppy, and if fails to boot again, we'll see an error message like "Non-System Disk or Disk Error".

3. What will happen if partition entry has been deleted?

If it has been deleted, next two partitions will move one line up in the partition table.

```
Physical Sector: Cyl 0, Side 0, Sector 1
0000001B0
                                         80 00
                                                . . . . . . . . . . . . . .
€.
0000001C0
        41 3F 06 FE 7F 64 7F 32 4E 00 A6 50 09 00 00 00
                                               A?.?#
d#2N.¦P.....
0000001D0
        41 65 OF FE BF 4A 25 83 57 00 66 61 38 00 00 00
                                               Ae.??J%?
W.fa8...
        0000001E0
 . . . . . . . . . . . . . . . .
```

If we try to boot now, the previous second (FAT) partition becomes the first and the loader will try to boot from it. And if it's not a system partition, we'll get the same error messages.

4. What will happen if partition entry has been damaged?

Let's write zeros to the location of the first partition entry.

```
Physical Sector: Cyl 0, Side 0, Sector 1
0000001B0
                                         80 00
                                               . . . . . . . . . . . . . .
€..
0000001C0
        . . . . . . . . . . . .
0000001D0 41 3F 06 FE 7F 64 7F 32 4E 00 A6 50 09 00 00 00
                                              A?.?#
d#2N.¦P.....
0000001E0 41 65 0F FE BF 4A 25 83 57 00 66 61 38 00 00 00
                                              Ae.??J%?
W.fa8...
....U?
```

If we try to boot now, the MBR loader will try to read and interpret zeros (or other garbage) as partition parameters and we'll get an error message like "Missing Operating System".

Thus, the second step in partition recovery is to run Disk Viewer and to make sure that the proper partition exists in the partition table and has been set as active.

How can recovery software help you in the above-mentioned scenarios?

- **1.** Discover and suggest you to choose the partition to be active (even FDISK does so).
- **2.** Discover and suggest you to choose the partition to be active.
- **3.** Perform a free disk space scan to look for partition boot sector or remaining of the deleted partition information in order to try to reconstruct Partition Table entry for the deleted partition.
- **4.** Perform all disk space scan to look for partition boot sector or remaining of the damaged partition information in order to try to reconstruct Partition Table entry for the damaged partition entry.

Why partition boot sector is so important?

Because if recovery software finds it, all necessary parameters to reconstruct partition entry in the Partition Table are there. (see *Partition Boot Sector is damaged* on page 160 topic for details).

What would happen if partition entry had been deleted then recreated with other parameters and re-formatted?

In this case, instead of the original partition entry we would have a new one and everything would work fine except that later on we could recall that we had some important data on the original partition. If you've created MBR, Partition Table, Volume Sectors backup (for example, Active@ Partition Recovery and Active@ UNERASER can do it) before, you can virtually restore it back and look for your data (in case if it has not been overwritten with new data yet). Some advanced recovery tools also have an ability to scan disk surface and try to reconstruct the previously deleted partition information from the pieces of left information (i.e. perform virtual partition recovery). However it is not guaranteed that you can recover something.

Partition Boot Sector is damaged

The Partition Boot Sector contains information, which the file system uses to access the volume. On personal computers, the Master Boot Record uses the Partition Boot Sector on the system partition to load the operating system kernel files. Partition Boot Sector is the first sector of the Partition.

For our first NTFS partition we have boot sector:

ctor 0 EB	r: (1 5B	Cyl 2 90	0, 3 4E	Sic 4 54	de 1 5 46	6 53	Secto 7 20	or 1 8 20	L 9 20	A 20	B 00	C 02	D 01	E 00	F 00	e[?NTFS
00	00	00	00	00	F8	00	00	3F	00	FF	00	3F	00	00	00	
• y • 00	00	00	00	80	00	80	00	ЗF	32	4E	00	00	00	00	00	€.€.?
5B '	43	01	00	00	00	00	00	1F	19	27	00	00	00	00	00	
02 iFA	00 .GA	00	00	08	00	00	00	10	EC	46	C4	00	47	C4	0C	
00	00 . u3 2	00 A	00	00	00	00	00	00	00	00	00	00	FA	33	C0	
8E	D0	BC	00	7C	FB	в8	C0	07	8E	D8	C7	06	54	00	00	Z??. u?
00	C7	06	56	00	00	00	C7	06	5B	00	10	00	в8	00	0D	.C.VC.
8E EPS	C0	2B	DB	E8	07	00	68	00	0D	68	66	02	СВ	50	53	ZA
51	52	06	66	A1	54	00	66	03	06	1C	00	66	33	D2	66	QR.f?
0F	в7	0E	18	00	66	F7	F1	FE	C2	88	16	5A	00	66	8B	. 'f?n?
D0	66	C1	EA	10	F7	36	1A	00	88	16	25	00	A 3	58	00	?fAe.?6?.%.?
A1	18	00	2A	06	5A	00	40	3в	06	5B	00	76	03	A1	5B	?*.Z.@;.[.v.?
00	50	в4	02	8B	16	58	00	в1	06	D2	E6	0A	36	5A	00	.P?.<.X.
8B	CA	86	E9	8A	36	25	00	в2	80	CD	13	58	72	2A	01	<e†es6%.?< td=""></e†es6%.?<>
06	54	00	83	16	56	00	00	29	06	5в	00	76	0в	C1	E0	.T.?.V).
05	8C	C2	03	D0	8E	C2	EB	8A	07	5A	59	5B	58	С3	BE	.?A.?
59	01	EB	80	BE	Е3	01	EB	03	BE	39	01	E8	09	00	BE	Y.e.?a.e.?
AD	01	E8	03	00	FB	EB	FE	AC	3C	00	74	09	в4	0E	BB	eue?
07	00	CD	10	EB	F2	СЗ	1D	00	41	20	64	69	73	6B	20	I.eoAA
72	65	61	64	20	65	72	72	6F	72	20	6F	63	63	75	72	read error
72	65	64	2E	0D	0A	00	29	00	41	20	6В	65	72	6E	65	red).A
6C	20	66	69	6C	65	20	69	73	20	6D	69	73	73	69	6E	l file is
67	20	66	72	6F	6D	20	74	68	65	20	64	69	73	6B	2E	g from the
0D	0A	00	25	00	41	20	6В	65	72	6E	65	6C	20	66	69	%.A kernel
6C	65	20	69	73	20	74	6 F	6F	20	64	69	73	63	6F	6E	le is too
74	69 1 2 2	67	75	6F	75	73	2E	0D	0A	00	33	00	49	6E	73	
65	72	74	20	61	20	73	79	73	74	65	6D	20	64	69	73	ert a
6В	65	74	74	65	20	61	6E	64	20	72	65	73	74	61	72	kette and
74	0D	0A	74	68	65	20	73	79	73	74	65	6D	2E	0D	0A	tthe
	Cto: 0 EB 00 .y. 00 5B 02 iFA 00 8E PS 51 0F D0 8E PS 51 0F D0 8B 06 05 59 AD 07 72 6C 67 0D 6C 74 .3. 65 6B 74	Ctor: 0 0 1 EB 5B 00 00 .y.? 00 00 5B 43 02 00 iFA.GA 00 07 8E D0 00 C7 8E C0 EPS 51 52 0F B7 D0 66 A1 18 00 50 8B CA 06 54 05 8C 59 01 AD 01 07 00 72 65 72 65 6C 20 07 20 00 00 00 50 00 72 65 72 65	Ctor: Cyl 0 1 2 EB 5B 90 00 00 00 5B 43 01 02 00 00 FA.GA. 00 07 06 8E D0 BC 00 C7 06 8E C0 2B 51 52 06 0F B7 0E D0 66 C1 A1 18 00 00 50 B4 8B CA 86 06 54 00 05 8C C2 59 01 EB AD 01 E8 07 00 CD 72 65 61 72 65 61 72 65 61 72 65 64 6C 20 66 67 20 66 60 67 20 66 60 72 60 60 67 72 60 60 60 60 72 60 72 70 60 60 72 70 72 60 72 70 72 60 72 70 72 60 72 70 72 70 72 74 60 72 70 72 70 72 70 72 70 72 70 72 70 72 70 72 70 72 70 72 70 72 70 72 70 70 70 70 70 70 70 70 70 70 70 70 70	Ctor: Cyl 0, 0 1 2 3 EB 5B 90 4E 00 00 00 00 5B 43 01 00 00 00 00 00 FA GA 00 00 00 C7 06 56 8E C0 2B DB 51 52 06 66 0F B7 0E 18 D0 66 C1 EA A1 18 00 2A 00 50 B4 02 8B CA 86 E9 06 54 00 83 05 8C C2 03 59 01 EB 08 AD 01 E8 03 05 8C C2 03 59 01 EB 08 AD 01 E8 03 07 00 CD 10 72 65 61 64 72 65 61 64 72 65 64 2E 6C 20 66 69 67 20 66 72 0D 0A 00 25 6C 65 20 69 74 69 67 75 	ctor: Cyl 0, Sid EB 5B 90 4E 54 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 100 60 C1 EA 10 00 60 C1 EA 10 00 60 C1 EA 10 01 66 C1 EA 10 01 66 E9 8A 06 01 EB 03 10 EB 02 01 EB 03 00 02 65 61 64 20 02 65 64 2E	ctor: Cyl 0, Side 1 0 1 2 3 4 5 EB 5B 90 4E 54 46 00 00 00 00 00 80 00 5B 43 01 00 00 00 00 5B 43 01 00 00 00 00 5B 43 01 00 00 00 00 00 00 00 00 00 00 00 174 00 00 00 00 00 00 18 00 2A 06 5A 00 66 10 66 C1 EA 10 F7 A1 18 00 2A 06 5A 00 50 B4 02 8B 16 8B CA 86 E9 8A 36 00 50 B4 02 8B 16 8B <td< td=""><td>ctor: Cyl 0, Side 1, 5 0 1 2 3 4 5 6 EB 5B 90 4E 54 46 53 00 00 00 00 80 00 80 90 00 00 00 80 00 80 5E 43 01 00 00 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 1SE D0 BC 00 7C FB B8 00 C7 06 56 01 00 66 F7 D0 66 C1 EA 10 F7 36 A1 18 00 2A 16 58 65 0</td><td>ctor: Cyl 0, Side 1, Sectored 1 2 3 4 5 6 7 EB 5B 90 4E 54 46 53 20 00 00 00 00 00 80 00 80 00 00 00 00 00 00 80 00 00 00 5B 43 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 07 06 56 00 00 00 00 00 07 06 56 00 00 00 00 00 67 06 56 00 00 00 00 00 66 C1 EA 10 F7 36 1A A1 18 00 2A 06 5A 00 40 00 50 B4 02 8B 16 58 00 8B CA 86 E9 8A 36 25 00 06 54 00 83 16 56 00 00 05 8C C2 03 D0 8E C2 EB 59 01 EB 08 BE E3 01 EB AD 01 E8 03 00 FB EB FE 07 00 CD 10 EB F2 C3 1D 72 65 61 64 2E 0D 0A 00 29 6C 20 66 72 6F 6D 20 74 0F 72 20 66 72 6F 73 73 2E 3.Ins 6C 65 20 69 73 20 74 6F 74 69 67 75 6F 75 73 2E .3.Ins 65 72 74 20 61 20 73 79 65 72 74 20 61 20 73 79<!--</td--><td>ctor: Cy1 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 EB 5B 90 4E 54 46 53 20 20 00 00 00 00 00 80 00 80 00 3F 00 00 00 00 00 00 00 00 3F 5B 43 01 00 00 00 00 00 10 102 00 00 00 00 00 00 00 10 1FA.GA. 00 00 00 00 00 00 00 00 10 C7 06 56 00 00 00 66 03 00 C7 06 56 00 00 66 03 01 E8 02 88 16 58 00 B1 00 50 B4 02 88 16</td><td>ctor: Cy1 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 EB 5B 90 4E 54 46 53 20 20 20 00 00 00 00 80 00 80 00 3F 32 5B 43 01 00 00 00 00 00 00 1F 19 02 00 00 00 00 00 00 00 10 EC 00</td></td></td<> <td>ctor: Cyl 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 A EB 5B 90 4E 54 46 53 20 20 20 20 00 00 00 00 80 00 3F 00 FF 00 00 00 00 00 00 00 10 EC 46 5B 43 01 00 00 00 00 10 EC 46 00</td> <td>ctor: Cyl 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 A B EB 5B 90 4E 54 46 53 20 20 20 20 00 00 00 00 00 00 3F 32 4E 00 y. ?. ? 00 00 00 00 00 1F 19 27 00 00 00 00 00 00 00 10 1E 19 27 00 00 00 00 00 00 00 00 10 1E 19 27 00 00 00 00 00 00 00 00 10 1E 10 1 1 10 1 10</td> <td>ctor: Cyl 0, Side 1, Sector 1 8 9 A B C 0 00 00 00 00 00 00 00 00 00 00 00 3F 00 00 00 3F 00 00 3F 00 00 00 3F 00 00 00 3F 00</td> <td>ctor: Cyl 0, Side 1, Sector 1 0</td> <td>ctor: Cyl 0, Side 1, Sector 1 0 A B C 0</td> <td>ctor: Ctor: Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 A B C 0 00</td>	ctor: Cyl 0, Side 1, 5 0 1 2 3 4 5 6 EB 5B 90 4E 54 46 53 00 00 00 00 80 00 80 90 00 00 00 80 00 80 5E 43 01 00 00 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 1SE D0 BC 00 7C FB B8 00 C7 06 56 01 00 66 F7 D0 66 C1 EA 10 F7 36 A1 18 00 2A 16 58 65 0	ctor: Cyl 0, Side 1, Sectored 1 2 3 4 5 6 7 EB 5B 90 4E 54 46 53 20 00 00 00 00 00 80 00 80 00 00 00 00 00 00 80 00 00 00 5B 43 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 1FA.GA. 00 00 00 00 00 00 00 00 00 07 06 56 00 00 00 00 00 07 06 56 00 00 00 00 00 67 06 56 00 00 00 00 00 66 C1 EA 10 F7 36 1A A1 18 00 2A 06 5A 00 40 00 50 B4 02 8B 16 58 00 8B CA 86 E9 8A 36 25 00 06 54 00 83 16 56 00 00 05 8C C2 03 D0 8E C2 EB 59 01 EB 08 BE E3 01 EB AD 01 E8 03 00 FB EB FE 07 00 CD 10 EB F2 C3 1D 72 65 61 64 2E 0D 0A 00 29 6C 20 66 72 6F 6D 20 74 0F 72 20 66 72 6F 73 73 2E 3.Ins 6C 65 20 69 73 20 74 6F 74 69 67 75 6F 75 73 2E .3.Ins 65 72 74 20 61 20 73 79 65 72 74 20 61 20 73 79 </td <td>ctor: Cy1 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 EB 5B 90 4E 54 46 53 20 20 00 00 00 00 00 80 00 80 00 3F 00 00 00 00 00 00 00 00 3F 5B 43 01 00 00 00 00 00 10 102 00 00 00 00 00 00 00 10 1FA.GA. 00 00 00 00 00 00 00 00 10 C7 06 56 00 00 00 66 03 00 C7 06 56 00 00 66 03 01 E8 02 88 16 58 00 B1 00 50 B4 02 88 16</td> <td>ctor: Cy1 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 EB 5B 90 4E 54 46 53 20 20 20 00 00 00 00 80 00 80 00 3F 32 5B 43 01 00 00 00 00 00 00 1F 19 02 00 00 00 00 00 00 00 10 EC 00</td>	ctor: Cy1 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 EB 5B 90 4E 54 46 53 20 20 00 00 00 00 00 80 00 80 00 3F 00 00 00 00 00 00 00 00 3F 5B 43 01 00 00 00 00 00 10 102 00 00 00 00 00 00 00 10 1FA.GA. 00 00 00 00 00 00 00 00 10 C7 06 56 00 00 00 66 03 00 C7 06 56 00 00 66 03 01 E8 02 88 16 58 00 B1 00 50 B4 02 88 16	ctor: Cy1 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 EB 5B 90 4E 54 46 53 20 20 20 00 00 00 00 80 00 80 00 3F 32 5B 43 01 00 00 00 00 00 00 1F 19 02 00 00 00 00 00 00 00 10 EC 00	ctor: Cyl 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 A EB 5B 90 4E 54 46 53 20 20 20 20 00 00 00 00 80 00 3F 00 FF 00 00 00 00 00 00 00 10 EC 46 5B 43 01 00 00 00 00 10 EC 46 00	ctor: Cyl 0, Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 A B EB 5B 90 4E 54 46 53 20 20 20 20 00 00 00 00 00 00 3F 32 4E 00 y. ?. ? 00 00 00 00 00 1F 19 27 00 00 00 00 00 00 00 10 1E 19 27 00 00 00 00 00 00 00 00 10 1E 19 27 00 00 00 00 00 00 00 00 10 1E 10 1 1 10 1 10	ctor: Cyl 0, Side 1, Sector 1 8 9 A B C 0 00 00 00 00 00 00 00 00 00 00 00 3F 00 00 00 3F 00 00 3F 00 00 00 3F 00 00 00 3F 00	ctor: Cyl 0, Side 1, Sector 1 0	ctor: Cyl 0, Side 1, Sector 1 0 A B C 0	ctor: Ctor: Side 1, Sector 1 0 1 2 3 4 5 6 7 8 9 A B C 0 00

0000001E0 00 17 00 5C 4E 54 4C 44 52 20 69 73 20 63 6F 6D ...\NTLDR is com 0000001F0 70 72 65 73 73 65 64 2E 0D 0A 00 00 00 00 55 AA pressed.....U?

The printout is formatted in three sections:

- Bytes 0x00– 0x0A are the jump instruction and the OEM ID (shown in bold print).
- Bytes 0x0B–0x53 are the BIOS Parameter Block (BPB) and the extended BPB. This block contains such essential parameters as:
 - Bytes Per Sector (WORD, offset 0x0B),
 - Sectors Per Cluster (BYTE, offset 0x0D),
 - Media Descriptor (BYTE, offset 0x15),
 - Sectors Per Track (WORD, offset 0x18),
 - Number of Heads (WORD, offset 0x1A),
 - Hidden Sectors (DWORD, offset 0x1C),
 - Total Sectors (LONGLONG, offset 0x28), etc...
- The remaining code is the bootstrap code (that is necessary for the proper system boot) and the end of sector marker (shown in bold print).

This sector is so important on NTFS, for example, duplicate of the boot sector is located on the disk.

Boot Sector for FAT looks different, however its BPB contains parameters similar to the above mentioned. There is no extra copy of this sector stored anywhere, so recovery on FAT is as half as less successful than on NTFS.

What will happen if Partition Boot Sector is damaged or bad/unreadable?

Lets fill up with zeros several lines of Partition Boot Sector:

000000000	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00			
000000060 A.ZOC.T	8E	DO	BC	00	7C	FΒ	В8	C0	07	8E	D8	C7	06	54	00	00	Ζ??.	. u?	

If we try to boot, we'll see "Non System Disk" or "Disk Error..". After we fail to load from it and from floppy, partition becomes not bootable.

Because a normally functioning system relies on the boot sector to access a volume, it is highly recommended that you run disk-scanning tools such as Chkdsk regularly, as well as back up all of your data files to protect against data loss in case you lose access to the volume.

Tools like Active@ Partition Recovery and Active@ UNERASER allow you to create backup of MBR, Partition Table and Volume Boot Sectors so that if for some reason it fails to boot, you can always restore your partition information and have an access to files/folders on that partition.

What to do if this sector is damaged?

• If we do have backup of the whole disk or MBR/Boot Sectors we can try to restore it from there.

- If we do not have backup, in case of NTFS we could try to locate a duplicate of Partition Boot Sector and get information from there.
- If duplicate boot sector is not found, only virtual partition recovery might be possible if we can determine critical partition parameters such as Sectors per Cluster, etc..

How can we fix NTFS boot sector using standard Windows NT/2000/XP tools?

On NTFS copy of boot sector is stored at the middle or at the end of the Volume.

You can boot from start-up floppy disks or CD-ROM, choose repair option during setup, and run **Recovery Console**. When you are logged on, you can run **FIXBOOT** command to try to fix boot sector.

How can recovery software help you in this situation?

- It can backup MBR, Partition Table and Boot Sectors and restore them in case of damage
- It can try to find out duplicate boot sector on the drive and re-create the original one or perform virtual data recovery based on found partition parameters
- Some advanced techniques allow assuming drive parameters even if duplicate boot sector is not found (i.e. perform virtual partition recovery) and give the user virtual access to the data on the drive to be able to copy them to the safer location.

Missing or Corrupted System Files

For Operating System to boot properly, system files required to be safe.

In case of Windows 95 / 98 / ME, these files are *msdos.sys*, *config.sys*, *autoexec.bat*, *system.ini*, *system.dat*, *user.dat*, etc.

In case of Windows NT / 2000 / XP these files are: *NTLDR*, *ntdetect.com*, *boot.ini*,located at the root folder of the bootable volume, Registry files (i.e., *SAM*, *SECURITY*, *SYSTEM* and *SOFTWARE*), etc.

If these files have been deleted, corrupted, damaged by virus, Windows will be unable to boot. You'll see error messages like "NTLDR is missing ...".

So, the next step in recovery process is to check the existence and safety of system files (for sure, you won't able to check them all, but you must check at least *NTLDR*, *ntdetect.com*, *boot.ini* which cause most of problems).

To do it in Windows 95 / 98 / ME - you can boot in *Command Prompt* Mode, or from the bootable floppy and check system files in the command line or with a help of third party recovery software.

To do it in Windows NT / 2000 / XP, you can use Emergency Repair Process, Recovery Console or third party recovery software.

Emergency Repair Process

To proceed with Emergency Repair Process, you need Emergency Repair Disk (ERD). This disk is recommended to create after you install and customize Windows. To create it, use the "Backup" utility from System Tools. You can use the ERD to repair damaged boot sector, damaged MBR, repair or replace missing or damaged NT Loader (NTLDR) and ntdetect.com files.

If you do not have an ERD, the emergency repair process can attempt to locate your Windows installation and start repairing your system, but it may not be able to do so.

To run the process, boot from Windows bootable disks or CD, and choose Repair option when system suggests you to proceed with installation or repairing. Then press **R**to run Emergency Repair Process and choose Fast or Manual Repair option. Fast Repair is recommended for most users, Manual Repair - for Administrators and advanced users only.

If the emergency repair process is successful, your computer will automatically restart and you should have a working system

Recovery Console

Recovery Console is a command line utility similar to MS-DOS command line. You can list and display folder content, copy, delete, replace files, format drives and perform many other administrative tasks.

To run Recovery Console, boot from Windows bootable disks or CD and choose Repair option, when system suggests you to proceed with installation or repairing and then press **C** to run Recovery Console. You will be asked to which system you want to log on to and then for **Administrator**'s password, and after you logged on - you can display drive's contents, check the existence and safety of critical files and, for example, copy them back if they have been accidentally deleted.

Recovery Software

Third party recovery software in most cases does not allow you to deal with system files due to the risk of further damage to the system, however you can use it to check for the existence and safety of these files, or to perform virtual partition recovery.