Microsoft Windows

BCDEdit Reference

June 13, 2007

Abstract

In Windows Vista® and later versions, the boot configuration data (BCD) store contains the boot configuration parameters and controls the computer's boot environment. BCDEdit is a Windows Vista command-line tool that can be used to add, delete, edit, and modify data in a BCD store. This paper is a reference for BCDEdit commands, data types, and well-known identifiers.

This information applies for the following operating systems:

Windows Server® 2008
Windows Vista

The current version of this paper is maintained on the Web at:

http://www.microsoft.com/whdc/system/platform/firmware/bcdeditref.mspx

References and resources discussed here are listed at the end of this paper.

Contents

Introduction 3
Terminology 4
BCDEdit Commands 5
- bootems 6
- bootdebug 6
- bootsequence 7
- copy 8
- create 8
- createstore 10
- dbgsettings 10
- debug 12
- default 12
- delete 12
- deletevalue 13
- displayorder 14
- ems 15
- emssettings 15
- enum 16
- export 17
- import 17
- set 18
- store 19
- sysstore 20
- timeout 20
- toolsdisplayorder 20
- v 21

BCDEdit Identifiers 22
BCDEdit Data Formats 23
BCDEdit Data Types 23
- All Entry Types 24
- Boot Applications 25
- Windows Boot Manager 26
- Windows Boot Loader 26
- Memory Diagnostic Application 28
- Resume Application 28

Firmware Boot Manager....28
Ntldr....29
Boot Sector Application....29
Device Additional Options....29
Custom Data Types....29
Resources....29

Disclaimer

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Introduction

The boot configuration data store (BCD) contains boot configuration parameters and controls how the operating system is booted for Windows Vista® and later versions. With earlier versions of Windows®, the way in which boot configuration data was handled depended on the system's firmware:

- For BIOS-based systems, boot configuration data was contained in a text file named Boot.ini.
- For Extensible Firmware Interface (EFI)–based systems, boot configuration data was contained in nonvolatile RAM (NVRAM).

BCD abstracts the underlying firmware and provides a common programming interface that can be used to manipulate the boot environment for all systems running Windows Vista or later versions of Windows. Every such system has a system BCD store that contains the data that controls the boot environment. Systems can have additional BCD stores, but only one store at a time can be designated as the system store.

Unlike Boot.ini, BCD stores data in a binary format and cannot be edited with a text editor. Instead, Windows Vista provides several ways to access a computer's BCD stores:

- The Windows Vista user interface (UI)

  MSConfig and the Shell's Control Panel System application provide end users with access to a limited subset of the data in the BCD system store, including the Windows Boot Manager's time-out setting, and the debug and safe-mode settings.

- BCDEdit

  BCDEdit is a command-line editor—included with Windows Vista and later versions—that provides complete access to all BCD stores on the system. BCDEdit can be used to create or delete BCD data stores, designate a new system store, and add, delete, or modify the data in an individual store.

- The BCD Windows Management Instrumentation (WMI) provider

  This component exposes an API that provides management tools with complete access to all BCD stores on the system.

This paper is a complete reference for the Windows Vista BCDEdit commands, identifiers, formats, and data types. For a quick reference while using BCDEdit, you can run the **/?** help command.

- To display a list of commands, run:

  bcdedit.exe /?

- To display details for a particular command, run:

  bcdedit.exe /? *command*

  For example, to display information about the **/createstore** command, run:

  bcdedit.exe /? /createstore

- To display a list of well-known boot entry identifiers, run:

  bcdedit.exe /? id

- To display a list of formats, run:

  bcdedit.exe /? formats

For a general discussion of the BCD store and the Windows boot environment, see "Boot Configuration Data in Windows Vista" on the WHDC Web site. For more information on the BCD WMI API, see "BCD Reference" on MSDN.

Note: You should run BCDEdit from a command window with elevated privileges. To do so:

- On the **Start** menu, click **All Programs, Accessories**, and then **Command Prompt**.
- Right-click **Command Prompt** and click **Run as administrator** on the context menu.
- A **User Account Control** dialog box appears. Click **Continue** to run the command window with administrative privileges.

Terminology

The following list defines the key BCDEdit terms that are used in this paper.

BCD store
A binary file that contains boot configuration data in Windows Vista and later versions. Boot applications use the system BCD store—which is on the system partition—during the boot process. You can also create additional BCD stores in separate files by using the **/export** command or by copying a BCD file.

boot application
A boot entry that represents a boot environment application, such as Windows Boot Manager, the Windows boot loader, or the Windows resume-from-hibernate application.

boot entry
An object in the BCD store. BCD stores can contain several types of boot entries, including the following boot applications:

- Boot Manager, which controls boot flow. In a dual-boot system, Boot Manager displays a boot selection menu to the user.
- The Windows boot loader, which loads a particular version or configuration of Windows Vista or later versions of Windows.
- Ntldr, which is the boot loader for versions of Windows earlier than Windows Vista.
- The resume application, which restores Windows to its running state when a computer resumes from hibernation.
- The memory diagnostics application, which runs a set of memory diagnostics.

Boot entries can also be used for other purposes, such as the global RAM defect list or the global boot loader settings.

command
A BCDEdit command. Commands are arguments to BCDEdit that consist of a '/' character followed by the command name. Most commands also have one or more arguments that are listed following the command. To run a command, type:

```
bcdedit /command [arguments...]
```

boot loader
A loader that loads Windows Vista and later versions of Windows.

data type/element
One or more data values that each boot entry has. Because elements have two characteristics—a name and an associated data type, they are also referred to as data types. For example, Boot Manager has a **Timeout** element that controls how long Boot Manager waits before automatically selecting the default boot entry.

firmware boot manager
On EFI-based systems, the entry in NVRAM that the firmware uses to locate Windows Boot Manager.

identifier
An associated globally unique identifier (GUID) that each boot entry has and that BCDEdit uses as an identifier for the entry.

memory diagnostic application
A boot environment tool that runs memory diagnostic tests, often referred to as Memdiag.

NtldrA
The legacy Windows boot loader, which loads versions of Windows earlier than Windows Vista.

resume application
A boot application that handles the resume-from-hibernation operation.

system BCD store
The BCD store that Windows Boot Manager uses to control boot flow.

well-known identifiers
Identifiers for commonly used boot entries that serve as readable aliases for the full GUIDs. For example, Boot Manager's well-known identifier is **{bootmgr}**. There are also virtual identifiers, where the associated GUID can vary from one boot to the next. For example, **{current}** is the well-known identifier for the currently booted operating system. For a complete list of well-known identifiers, see "BCDEdit Identifiers" later in this paper.

Windows Boot Manager
A boot environment application that initiates the boot process. With a multiboot system, Boot Manager displays an operating system selection menu. It uses the BCD store to locate the Windows loader to continue loading a specific version of Windows.

BCDEdit Commands

The following table summarizes the BCDEdit commands by category. The sections that follow the table provide detailed references for each command, presented in alphabetical order.

Command	Description
Commands that operate on a store	
/createstore	Creates a new empty BCD store.
/export	Exports the contents of the system BCD store to a specified file.
/import	Restores the state of the system BCD store from a specified file.
Commands that operate on boot entries in a store	
/copy	Makes copies of boot entries.
/create	Creates new boot entries.
/delete	Deletes boot entries.

Command	Description
Commands that operate on elements	
/deletevalue	Deletes elements from a boot entry.
/set	Creates or modifies a boot entry's elements.
Commands that control output	
/enum	Lists the boot entries in a store.
Commands that control Boot Manager	
/bootsequence	Specifies a one-time boot sequence.
/default	Specifies the default boot entry.
/displayorder	Specifies the order in which Boot Manager displays its menu.
/timeout	Specifies the Boot Manager Timeout value.
/toolsdisplayorder	Specifies the order in which Boot Manager displays the tools menu.
Commands that control Emergency Management Services	
/bootems	Enables or disables Emergency Management Services (EMS) for a specified boot application.
/ems	Enables or disables EMS for an operating system boot entry.
/emssettings	Specifies global EMS parameters.
Commands that control debugging	
/bootdebug	Enables or disables boot debugging for a boot application.
/dbgsettings	Specifies global debugger parameters.
/debug	Enables or disables kernel debugging for an operating system boot entry.
Commands that modify other commands	
/store	Specifies the BCD store upon which a command acts.
/v	Displays boot entry identifiers in full, rather than using well-known identifiers.

/bootems

Enables or disables EMS for a specified boot application:

bcdedit [**/store** *filename*] **/bootems** [*id*] **{ ON** | **OFF }**

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Optional. The identifier of the boot application to be modified. The default value is the current operating system entry.

ON | OFF

Required. **ON** enables EMS, and **OFF** disables EMS.

Example

The following command enables EMS for Boot Manager:

```
bcdedit /bootems {bootmgr} ON
```

Remarks

This command runs without errors for any boot entry, but affects only boot applications.

/bootdebug

Enables or disables the boot debugger for a specified boot entry:

bcdedit [**/store** *filename*] **/bootdebug** [*id*] **{ ON** | **OFF }**

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Optional. The identifier of the boot entry to be modified. The default value is the current operating system entry.

ON | OFF

Required. **ON** enables boot debugging, and **OFF** disables boot debugging.

Examples

The following command enables boot debugging for the Windows boot loader for the current operating system:

```
bcdedit /bootdebug ON
```

The following command disables boot debugging for Boot Manager:

```
bcdedit /bootdebug {bootmgr} OFF
```

Remarks

This command runs without errors for any boot entry, but affects only boot applications.

/bootsequence

Specifies the boot entries and display order for a one-time boot sequence:

bcdedit [**/store** *filename*] **/bootsequence** *id* [...] [**/addfirst** | **/addlast** | **/remove**]

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id **[...]**

Required. A list of identifiers for the entries to be added or removed. You must specify at least one entry. To specify multiple entries, list the identifiers on the command line in the order in which they should appear in the boot sequence, separated by a space.

/addfirst | /addlast | /remove

Optional. You can specify one of the commands from this set. They apply to only a single boot entry, so if you use them, the identifier list must contain only one value.

/addfirst

Adds the specified boot entry to the beginning of the one-time boot sequence. If the boot entry is already in the sequence, it is moved to the beginning.

/addlast

Adds the specified boot entry to the end of the one-time boot sequence. If the identifier is already in the sequence, it is moved to the end.

/remove

Removes the specified boot entry from the one-time boot sequence. If the one-time boot sequence has only one entry, then the one-time boot sequence value is deleted from the Boot Manager entry. If the specified boot entry is not in the one-time boot sequence, the **/bootsequence** command has no effect.

Examples

The following command specifies a one-time boot sequence with three entries. The first two are Windows boot loaders, identified by their GUIDS, followed by the well-known identifier for Ntldr:

```
bcdedit /bootsequence {802d5e32-0784-11da-bd33-000476eba25f}
{cbd971bf-b7b8-4885-951a-fa03044f5d71} {ntldr}
```

The following command adds a Windows boot loader entry, specified by its GUID, to the end of the one-time boot sequence:

```
bcdedit /bootsequence {802d5e32-0784-11da-bd33-000476eba25f}
/addlast
```

Remarks

This command creates a display order to be used only for the next boot. By default, the boot sequence specified by the identifier list replaces the existing sequence. To modify an existing sequence, use the **/addfirst | /addlast | /remove** arguments.

This command is similar to **/displayorder**, except that a one-time boot sequence is used only once: the next time the system is booted. After that has occurred, the system reverts to the regular display order.

/copy

Creates a copy of a specified boot entry:

bcdedit [**/store** *filename*] **/copy** *id* **/d** *description*

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Required. The identifier of the boot entry to be copied.

/d *description*

Required. A string that contains the description to be associated with the new boot entry.

Example

The following command creates a copy of Windows boot loader entry in the system BCD store:

```
bcdedit /copy {cbd971bf-b7b8-4885-951a-fa03044f5d71} /d "Copy
of entry"
```

Remarks

This command creates a new GUID for the copy. When the command returns, BCDEdit displays the new GUID in the command window.

/create

Creates a new boot entry:

bcdedit [**/store** *filename*] **/create** [*id*] **/d** *description* [**/application** *apptype* | **/inherit** [*apptype*] | **/inherit DEVICE** | **/device**]

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Optional. The new boot entry identifier:

- If you set *id* to a well-known identifier, such as **{ntldr}**, you cannot use the **/application**, **/inherit**, and **/device** options.
- If you set *id* to a GUID that does not correspond to a well-known identifier, you must use **/application**, **/inherit**, or **/device** to specify the boot entry type. BCEdit checks the GUID to ensure that it is not equal to the identifier of an existing entry.
- If you do not specify an identifier, BCDEdit creates a new GUID to serve as the entry's identifier. You must use **/application**, **/inherit**, or **/device** to specify the boot entry type.

/d *description*

Required. A string that contains the new boot entry's description.

/application *apptype*

Optional. If *id* is not set to a well-known identifier, the option that is used to specify the new boot entry as an application entry, with an application type of *apptype*, which must be set to one of the values in the following table:

Apptype	Description
BOOTSECTOR	The boot sector application
OSLOADER	The Windows boot loader
RESUME	A resume application

You cannot use the **/application** option for other types of applications. Instead, you must set *id* to the application's well-known identifier.

/inherit [*apptype*]

Optional. If *id* is not set to a well-known identifier, the option that is used to specify the new boot entry as an inherit entry that can be inherited by the application type specified by *apptype*. The application type can be one of the values in the following table:

Apptype	Description
BOOTMGR	Boot Manager
BOOTSECTOR	The boot sector application
FWBOOTMGR	The firmware boot manager
MEMDIAG	The memory diagnostics application
NTLDR	Ntldr
OSLOADER	The Windows boot loader
RESUME	The resume application

If you do not specify an *apptype* value, any boot entry can inherit the new boot entry.

/inherit DEVICE

Optional. If *id* is not set to a well-known identifier, the option that is used to specify the new boot entry as an inherit entry that only a device options boot entry can inherit.

/device

Optional. If *id* is not set to a well-known identifier, the option that is used to specify the new boot entry as an additional device options entry.

Examples

The following command creates an Ntldr boot entry:

```
bcdedit /create {ntldr} /d "Legacy OS Loader"
```

The following command creates a RAM disk additional options boot entry:

```
bcdedit /create {ramdiskoptions} /d "Ramdisk options"
```

The following command creates a new Windows boot loader entry:

```
bcdedit /create /d "Windows Vista" /application osloader
```

The following command creates a new debugger settings boot entry:

```
bcdedit /create {dbgsettings} /d "Debugger Settings"
```

Remarks

If you set *id* set to a well-known identifier, such as **{ntldr}** or **{dbgsettings}**, you cannot use the **/application**, **/inherit**, and **/device** options. Those options are already defined for well-known identifiers. If you set *id* to something other than a well-known identifier or if you do not specify an *id* value, you must specify the boot entry's inheritance characteristics by using one of the **/application**, **/inherit**, or **/device** options.

/createstore

Creates a new empty BCD store:

bcdedit /createstore [*filename*]

Parameters

filename

Optional. The file name of the new BCD store. If the file name contains spaces, it must be enclosed in quotation marks ("").

- If you do not specify a file name, BCDEdit creates a new empty system BCD store.
- If you specify just the file name, BCDEdit creates the file in the current default folder.
- To have the file placed in a specific folder, set *filename* to the fully qualified path. You can use environment variables as part of the path. The path must end in a valid file name, such as c:\temp\mystore. The command fails if the path ends in the name of a folder, such as c:\temp, or the name of an existing file.

Example

The following command creates a BCD store named C:\Data\BCD:

```
bcdedit /createstore C:\DATA\BCD
```

/dbgsettings

Sets or displays the global debugger settings:

bcdedit [**/store** *filename*] **/dbgsettings** [*debugtype* [**DEBUGPORT:***port*] [**BAUDRATE:***baud*] [**CHANNEL:***channel*] [**TARGETNAME:***targetname*] **/start** *startpolicy* **/noemux**]

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

debugtype
Optional. The type of debugger. *debugtype* can be set one of the following:

SERIAL
1394
USB

There is no default debugging type. If you omit *debugtype*, **/dbgsettings** displays the current settings.

DEBUGPORT:*port*
Optional. If *debugtype* is set to SERIAL, the option that is used to specify which serial port to use as the debugging port. Set *port* to 1 for COM1, and so on.

BAUDRATE:*baud*
Optional. If *debugtype* is set to SERIAL, the option that is used to specify the baud rate to be used for debugging. Set *baud* to 57600 for a baud rate of 57,600, and so on. Valid baud rates are 9600, 19200, 38400, 57600, and 115200, and the default value is 9600. If you assign any other value to *baud*, BCDEdit returns an “Invalid baud rate” error.

CHANNEL:*channel*
Optional. If *debugtype* is set to 1394, the option that is used to specify the 1394 channel to be used for debugging. Set *channel* to the appropriate 1394 channel.

TARGETNAME:*targetname*
Optional. If *debugtype* is set to USB, the option that is used specifies the USB target name to be used for debugging.

/start *startpolicy*
Optional. The debugger start policy for all debugger types. *startpolicy* can be set to one of the following:

ACTIVE
AUTOENABLE
DISABLE

By default, *startpolicy* is set to ACTIVE.

/noumex
Optional. The option that causes the kernel debugger to ignore user-mode exceptions.

Examples

The following command displays the current global debugger settings:

```
bcdedit /dbgsettings
```

The following command sets the global debugger settings to serial debugging over COM1 at 115,200 baud:

```
bcdedit /dbgsettings SERIAL DEBUGPORT:1 BAUDRATE:115200
```

The following command sets the global debugger settings to 1394 debugging on channel 23:

```
bcdedit /dbgsettings 1394 CHANNEL:23
```

The following command sets the global debugger settings to use USB debugging with a target named DEBUGGING:

```
bcdedit /dbgsettings USB TARGETNAME:DEBUGGING
```

To set an individual global debugger setting, use the **/set** command, as follows:

```
bcdedit /set {dbgsettings} debugtype value
```

Remarks

The **/debugsettings** command does not enable or disable the debugger. You must use **/debug** for that purpose.

/debug

Enables or disables the kernel debugger for a specified boot entry:

bcdedit [**/store** *filename*] **/debug** [*id*] **{ ON** | **OFF }**

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Optional. The identifier of the boot entry for which kernel debugging is to be enabled or disabled. This value can be set only to Windows boot loader entries. By default, *id* is set to **{current}**.

ON | **OFF**

Required. **ON** enables kernel debugging, and **OFF** disables kernel debugging.

Examples

The following command enables kernel debugging for the current Windows boot loader entry:

```
bcdedit /debug ON
```

The following command disables kernel debugging for the specified Windows boot loader entry:

```
bcdedit /debug {cbd971bf-b7b8-4885-951a-fa03044f5d71} OFF
```

/default

Specifies the boot entry to be used by default if the user does not select an entry before the time out expires:

bcdedit [**/store** *filename*] **/default** *id*

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Required. The default boot entry's identifier.

Examples

The following command sets the specified Windows boot loader as the default Boot Manager entry:

```
bcdedit /default {cbd971bf-b7b8-4885-951a-fa03044f5d71}
```

The following command sets Ntldr as the default boot entry:

```
bcdedit /default {ntldr}
```

/delete

Deletes a boot entry from a BCD store:

bcdedit [**/store** *filename*] **/delete** *id* [**/f**] [**/cleanup** | **/nocleanup**]

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Required. The identifier of the boot entry to be deleted.

/f

Optional. Force deletion. You must use this option to delete boot entries that have a well-known identifier. It is not required for other boot entries.

/cleanup | /nocleanup

Optional. These options specify whether the boot entry should be removed from the display order. The default value is **/cleanup**.

/cleanup

Removes the boot entry from the display order. If you delete an operating system boot loader entry, the associated resume-from-hibernation boot entry is also deleted, as long as it is not referenced by any other operating system loaders.

/nocleanup

Deletes the specified boot entry without removing it from the display order.

Examples

The following command deletes the specified Windows boot loader entry from the system BCD store and from the display order:

```
bcdedit /delete {cbd971bf-b7b8-4885-951a-fa03044f5d71}
```

The following command deletes the specified Windows boot loader entry from the system BCD store and from the display order:

```
bcdedit /delete {cbd971bf-b7b8-4885-951a-fa03044f5d71}
/cleanup
```

The following command deletes the specified Windows boot loader entry from the system BCD store without removing the boot entry from the display order:

```
bcdedit /delete {cbd971bf-b7b8-4885-951a-fa03044f5d71}
/nocleanup
```

The following command deletes the Ntldr boot entry from the system BCD store:

```
bcdedit /delete {ntldr} /f
```

/deletevalue

Deletes an element from a boot entry in a BCD store:

bcdedit [**/store** *filename*] **/deletevalue** [*id*] *datatype*

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Optional. The identifier of the boot entry that owns the element to be deleted. By default, *id* is set to **{current}**.

datatype
Required. The element to be deleted. The elements and data formats that are available for the various boot entry types are listed in “BCDEdit Data Types” later in this paper.

Examples

The following command deletes the **bootsequence** element from the Boot Manager entry:

```
bcdedit /deletevalue {bootmgr} bootsequence
```

The following command deletes the Windows Preinstallation Environment (WinPE) element from the current operating system boot entry:

```
bcdedit /deletevalue winpe
```

The following command deletes the WinPE element from the specified Windows boot loader entry:

```
bcdedit /deletevalue {cbd971bf-b7b8-4885-951a-fa03044f5d71}
winpe
```

Remarks

For more information on BCD elements and data formats, see “BCDEdit Data Types” and “BCDEdit Data Formats” later in this paper.

/displayorder

Specifies the Boot Manager’s display order:

bcdedit [**/store** *filename*] **/displayorder** *id* [...] [**/addfirst** | **/addlast** | **/remove**]

Parameters

/store *filename*
Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id [...]
Required. A list of identifiers for the entries to be added or removed. You must specify at least one entry. To specify multiple entries, list the identifiers on the command line in the order in which they should appear in the boot sequence, separated by a space.

/addfirst | /addlast | /remove
Optional. You can specify one of the commands from this set. They apply to only a single boot entry, so if you use one of these options, the identifier list must contain only one value.

/addfirst
Adds the specified boot entry to the beginning of the display order. If the boot entry is already in the display order, it is moved to the beginning.

/addlast
Adds the specified boot entry to the end of the display order. If the identifier is already in the display order, it is moved to the end.

/remove
Removes the specified boot entry from the display order. If the display order has only one entry, then the display order value is deleted from the Boot Manager boot entry. If the specified boot entry is not in the in the display order, the **/displayorder** command has no effect.

Examples

The following command creates a display order that consists of two Windows loader boot entries, identified by their GUIDS, followed by Ntldr:

```
bcdedit /displayorder {802d5e32-0784-11da-bd33-000476eba25f}
{cbd971bf-b7b8-4885-951a-fa03044f5d71} {ntldr}
```

The following command adds a Windows boot loader entry to the end of the existing display order:

```
bcdedit /displayorder {802d5e32-0784-11da-bd33-000476eba25f}
/addlast
```

/ems

Enables or disables EMS for a specified Windows boot loader entry:

bcdedit [**/store** *filename*] **/ems** [*id*] **{ ON** | **OFF }**

Parameters

/store *filename*
Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id
Optional. The identifier of the boot entry to be modified. *id* can be set only to Windows boot loader boot entries. By default, *id* is set to **{current}**.

ON | OFF
Required. **ON** enables EMS, and **OFF** disables EMS.

Example

The following command enables EMS for the current Windows boot loader entry:

```
bcdedit /ems ON
```

/emssettings

Sets the global EMS settings for the system:

bcdedit [**/store** *filename*] **/emssettings BIOS** | **EMSPORT:***port* [**EMSBAUDRATE:***baudrate*]

Parameters

/store *filename*
Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

BIOS | EMSPORT:*port*
Required. Options that specify the EMS configuration:

- **BIOS.** Uses BIOS settings for the EMS configuration. This option works only on systems that have BIOS support for EMS.
- **EMSPORT**. Uses the specified serial port. To specify a port value, set *port* to 1 for COM1, and so on.

EMSBAUDRATE:*baudrate*
Optional. The baud rate to use for the specified serial port. To use EMSBAUDRATE, you must also set the **EMSPORT** option to specify the serial port. Set *baudrate* to 57600 for a baud rate of 57,600, and so on. By default, *baudrate* is set to 9600. Do not use this option if you have set the **BIOS** option.

Examples

The following command sets the EMS parameters to BIOS settings:

```
bcdedit /emssettings BIOS
```

The following command sets the EMS parameters to use COM2 at 115,200 baud:

```
bcdedit /emssettings EMSPORT:2 EMSBAUDRATE:115200
```

Remarks

The **/emssettings** command does not enable or disable EMS. Use **/ems** for that purpose.

/enum

Lists boot entries in a specified BCD store:

bcdedit [**/store** *filename*] **/enum** [*type* | *id*] [**/v**]

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

type

Optional. The type of boot entries to be listed. *type* can be set to one of the values in the following table. The values are not case sensitive. By default, type is set to ACTIVE.

Type	Description
ACTIVE	All boot entries in the store in Boot Manager display order.
ALL	All boot entries.
BOOTAPP	All boot environment applications.
BOOTMGR	Boot Manager.
FIRMWARE	All firmware applications.
INHERIT	All inherit entries.
OSLOADER	All operating system boot entries.
RESUME	All resume-from-hibernation boot entries.

id

Optional. The identifier of the boot entry to be listed. If *id* is specified, then **/enum** lists only that object. Otherwise, **/enum** lists all boot entries that are consistent with the *type* setting.

/v

Optional. Display boot entry identifiers in full, rather than using names for well-known identifiers.

Examples

The following command lists all Windows boot loader entries:

```
bcdedit /enum OSLOADER
```

The following command lists all Boot Manager entries:

```
bcdedit /enum BOOTMGR
```

The following command lists all firmware entries:

```
bcdedit /enum FIRMWARE
```

The following command lists only the default boot entry:

```
bcdedit /enum {default}
```

The following command lists only the specified operating system boot entry:

```
bcdedit /enum {b38a9fc1-5690-11da-b795-e9ad3c5e0e3a}
```

Remarks

/enum is the default BCDEdit command. Running BCDEdit without arguments is equivalent to running **bcdedit /enum ACTIVE**.

/export

Exports the contents of the system store to a specified file, which can be used later to restore the state of the system store:

bcdedit /export *filename*

Parameters

filename

The name of the file that is to contain the exported store. If *filename* contains spaces, it must be enclosed in quotation marks (""):

- If you specify just the file name, BCDEdit creates the file in the current default folder.
- To have the file placed in a specific folder, set *filename* to the fully-qualified path. The path must end in a valid file name, such as c:\temp\mystore. If the path ends in the name of a folder (such as c:\temp) or the name of an existing file, the command fails.
- You can use valid environment variables in the path. For example, if %TEMP% is defined as c:\Temp, setting *filename* to %TEMP%\MyStore creates an exported store named MyStore in c:\Temp.

Example

The following command exports the system store to C:\Data\BCD Backup:

```
bcdedit /export "C:\Data\BCD Backup"
```

Remarks

This command can be used only to export data from the system store. The system store itself is not affected.

/import

Restores the state of the system store by importing data from a backup BCD store previously created by the **/export** command:

bcdedit /import *filename*

Parameters

filename

Required. The name of the file that is imported into the system store. If *filename* contains spaces, it must be enclosed in quotation marks (""):

- If you specify just the file name, BCDEdit looks for the file in the current default folder.
- To import from a file in a specific folder, set *filename* to the fully-qualified path. The path must end in a valid file name, such as c:\temp\mystore. If the path ends in the name of a folder (such as c:\temp) or the name of an existing file, the command fails.

- You can use valid environment variables in the path. For example, if %TEMP% is defined as c:\Temp, setting *filename* to %TEMP%\MyStore imports a store named MyStore in c:\Temp.

Example

The following command imports the data from C:\Data\BCD Backup into the system store:

```
bcdedit /import "C:\Data\BCD Backup"
```

Remarks

/import deletes any existing boot entries in the system store before importing the data from the backup BCD store.

This command can be used only to import data into the system store.

/set

Creates a new element or modifies an existing element for a specified boot entry:

bcdedit [**/store** *filename*] **/set** [*id*] *datatype value* [**/addfirst** | **/addlast** | **/remove**]

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id

Optional. The identifier of the boot entry to be modified. By default, *id* is set to **{current}**.

datatype

Required. The data type of the element to be created or modified. The elements and data types that are available for the various boot entry types are listed in "BCDEdit Data Types" later in this paper.

value

Required. The value to be assigned to the element. The format of *value* depends on *datatype*. The format associated with each data type is listed in "BCDEdit Data Types," and the details of the formats are given in "BCDEdit Data Formats" later in this paper.

Beta Disclaimer: The following options are available only on Windows Server 2008, Beta 3, and later.

/addfirst | /addlast | /remove

Optional. If the element to be set is a list, you can specify one of the following commands. They apply to only a single list element, so if you use one of these options, *value* must contain only a single object.

/addfirst

Adds the specified element to the beginning of the list. If the boot entry is already in the list, it is moved to the beginning.

/addlast

Adds the specified element to the end of the list. If the identifier is already in the list, it is moved to the end.

/remove

Removes the specified element from the list. If the list has only one entry, then the element is deleted from the Boot Manager boot entry. If the specified value is not in the in the list, the **/displayorder** command has no effect.

Examples

The following command sets the application device to partition C for the specified operating system boot entry:

```
bcdedit /set {cbd971bf-b7b8-4885-951a-fa03044f5d71} device
partition=C:
```

The following command sets the application path to \windows\system32\winload.exe for the specified operating system boot entry:

```
bcdedit /set {cbd971bf-b7b8-4885-951a-fa03044f5d71} path
\windows\system32\winload.exe
```

The following command sets the NX policy to OptIn for the current operating system boot entry:

```
bcdedit /set nx optin
```

Remarks

For more information on how to use the **/set** command, see "BCDEdit Data Types" and "BCDEdit Data Formats" later in this paper.

/store

Specifies the BCD store upon which a command should act:

bcdedit /store *filename* [...]

Parameters

filename

The file name of a BCD store. If *filename* contains spaces, it must be enclosed in quotation marks (""):

- If you specify just the file name, BCDEdit looks for the file in the current default folder.
- To refer to a file placed in a specific folder, set *filename* to the fully-qualified path. The path must end in a valid file name, such as c:\temp\mystore. If the path ends in the name of a folder)such as c:\temp) or the name of an existing file, the command fails.
- You can use valid environment variables in a path. For example, if %TEMP% is defined as c:\Temp, setting *filename* to %TEMP%\MyStore uses the store named MyStore in c:\Temp.

Example

The following command lists the active boot entries in the specified data store file:

```
bcdedit /store C:\DATA\BCD /enum ACTIVE
```

Remarks

You can run **bcdedit /store** as a standalone command; it is equivalent to running **bcdedit /enum ACTIVE**.

You cannot use **/store** with the **/createstore**, **/import**, and **/export** commands.

If a BCD command does not use the **/store** argument, then *filename* is set to the system store.

Beta Disclaimer: The following command is available only on Windows Server 2008, Beta 3, and later.

/sysstore

Specifies the partition that contains the system store, for EFI-based systems:

bcdedit /sysstore *partition*

Parameters

partition

Required. The partition, such as C:, that contains the system store.

Example

The following command sets the system store to the C: partition:

```
bcdedit /sysstore C:
```

Remarks

An EFI system typically has only EFI system partition (ESP), and the system BCD store is located on that partition. However, EFI systems can have multiple ESPs on multiple hard drives, only one of which can contain the system BCD store. In that case, you must use the /**sysstore** command to inform Boot Manager which ESP contains the system BCD store.

This command is not used with BIOS-based systems.

/timeout

Specifies how long Boot Manager should wait before selecting the default boot entry:

bcdedit [**/store** *filename*] **/timeout** *timeout*

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed earlier in this paper.

timeout

Required. How long Boot Manager should wait, in seconds, before selecting the default boot entry.

Example

The following command sets the Boot Manager's time out to 30 seconds:

```
bcdedit /timeout 30
```

Remarks

To specify the default boot entry, run the **/default** command.

/toolsdisplayorder

Specifies the entries and display order that Boot Manager should use for the tools menu:

bcdedit [**/store** *filename*] **/toolsdisplayorder** *id* [...] [**/addfirst** | **/addlast** | **/remove**]

Parameters

/store *filename*

Optional. The BCD store to be used. The default value is the system store. **/store** is discussed later in this paper.

id ***[...]***

Required. A list of identifiers for the tools to be added to or removed from the tools display. You must specify at least one identifier. Additional identifiers must be separated by spaces.

/addfirst | /addlast | /remove

Optional. You can specify one of the commands from this set. They apply to only a single tool, so the identifier list must contain only one value.

/addfirst

Adds the specified tool to the beginning of the tools display order. If the tool is already in the list, it is moved to the top of the list.

/addlast

Adds the specified tool to the end of the tools display order. If the tool is already in the list, it is moved to the end of the list.

/remove

Removes the specified tool from the tools display order. If the list contains only one tool, BCDEdit deletes the tools display order from the Boot Manager boot entry. If the specified tool is not in the in the list, **/toolsdisplayorder** has no effect.

Examples

The following command sets two tools boot entries and the memory diagnostic in the Boot Manager's tools display order:

```
bcdedit /toolsdisplayorder {802d5e32-0784-11da-bd33-
000476eba25f} {cbd971bf-b7b8-4885-951a-fa03044f5d71} {memdiag}
```

The following command adds the specified tool to the end of the tools display order:

```
bcdedit /toolsdisplayorder {802d5e32-0784-11da-bd33-
000476eba25f} /addlast
```

Remarks

Boot Manager displays a list of available operating systems and a list of tools. By default, the only tool is the memory diagnostics application. You can use **/toolsdisplayorder** to add other tools to the list.

/v

Display all identifiers in full:

bcdedit /v ...

Example

The following command lists the active boot entries in the system store with all boot entry identifiers displayed in full:

```
bcdedit /enum ACTIVE /v
```

Remarks

GUIDs are used to identify all boot entries. However, by default, BCDEdit displays the readable form of well-known identifiers, such as **{current}** or **{bootmgr}**. **/v** directs BCDEdit to display all identifiers as GUIDs. Running **bcdedit /v** by itself is equivalent to running **bcdedit /enum ACTIVE /v**.

BCDEdit Identifiers

Many of the BCDEdit commands require *id* values to uniquely identify boot entries in the store. An *id* value is the string form of a GUID, which has the following value, where each 'x' corresponds to a hexadecimal digit:

{xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx}

The values of each digit vary for different GUIDS, but braces at the beginning and the end of the string are required and the dashes must be in the indicated locations in the string. The string cannot contain any white space. For example:

{d2b69192-8f14-11da-a31f-ea816ab185e9}

Several commonly used boot entries can also be identified by well-known identifiers, which are simpler to use than the corresponding GUIDs. BCDEdit displays well-known identifiers in output unless you include a **/v** argument. In that case, BCDEdit displays the GUID. The following table contains the well-known identifiers.

Well-Known Identifiers

Identifier	Description
{badmemory}	The global RAM defect list. This list can be inherited by any boot application boot entry.
{bootloadersettings}	A collection of global settings that should be inherited by all Windows boot loader entries.
{bootmgr}	Windows Boot Manager.
{current}	A virtual identifier that represents the currently running operating system.
{dbgsettings}	The global debugger settings. Any boot application entry can inherit these settings.
{default}	A virtual identifier that represents the default boot entry.
{emssettings}	The global EMS settings. These settings can be inherited by any boot application entry.
{fwbootmgr}	The firmware boot manager boot entry. This is the Boot Manager entry that is stored in NVRAM for EFI-based systems.
{globalsettings}	A collection of global settings that should be inherited by all boot application entries.
{memdiag}	The memory diagnostic application.
{ntldr}	The Windows legacy loader, Ntldr. It is used for versions of Windows earlier than Windows Vista.
{ramdiskoptions}	Additional options required for RAM disk devices.
{resumeloadersettings}	A collection of global settings that should be inherited by all Windows resume-from-hibernation application entries.

BCDEdit Data Formats

The following table describes the data format values that can be assigned to the **/set** command's *datatype* argument.

Format	Description
Boolean	A Boolean value that can be set to TRUE or FALSE. You can also use the following values instead of TRUE or FALSE: ▪ TRUE: 1, ON, YES ▪ FALSE: 0, OFF, NO,
device	A device data type that can be set to one of the following: BOOT PARTITION=*drive* FILE=[*parent*]*path* RAMDISK=[*parent*]*path*,*optionsid* The options for these types are as follows: ▪ *drive.* Required. A drive letter with a colon but without a trailing backslash, such as c:. ▪ *parent*. Required. Can be set to either [BOOT] or a drive letter with a colon but no trailing backslash. The square brackets are a required part of the syntax, not the usual convention indicating an optional value. ▪ *path.* Required. A path to the file from the root of the parent device. ▪ *optionsid.* Optional. The identifier of the device options boot entry that contains the system deployment image (SDI) options for the RAM disk. This value is usually set to **{ramdisksdioptions}**.
enum	The data type that takes a value from an enumerated list. For example, the NX data type can be set to one of four enumerated values: OPTIN, OPTOUT, ALWAYSON, or ALWAYSOFF.
id	The identifier for a boot entry. This is typically the string form of the GUID that is associated with a boot entry. Commonly used boot entries have well-known IDs that can be used in place of a GUID. For example, the well-known ID for the current operating system is **{current}**.
integer	A 64-bit integer.
list	A boot entry identifier list that contains one or more boot entry identifiers separated by spaces. The list should not be enclosed in quotation marks.
string	A literal string. If it contains spaces, it should be surrounded by quotation marks ("").

BCDEdit Data Types

The **/set** and **/deletevalue** commands require a *datatype* argument, followed by a value for the data type. Most data types are identified by a name, but you can also define custom data types. Each data type is associated with a data format, such as string or integer. The available data formats are listed in the previous section, "BCDEdit Data Formats".

For example, the Windows boot loader has an NX element that can be set to one of four enumerated values: OPTIN, OPTOUT, ALWAYSON, or ALWAYSOFF. The following **/set** command sets the NX policy to OptIn for the current operating system:

```
bcdedit /set {current} NX OPTIN
```

This section is a complete reference for the data types associated with the various boot entry types. To display information about data types when you are using BCDEdit, type the following command:

```
bcdedit.exe /? TYPES Apptype
```

where *Apptype* is one of the items in the following table:

Apptype	Description
BOOTAPP	All boot applications
BOOTMGR	Boot Manager
BOOTSECTOR	The boot sector application
CUSTOMTYPES	Custom data types
DEVOBJECT	Device objects
FWBOOTMGR	The firmware boot manager
MEMDIAG	The memory diagnostics application
NTLDR	Ntldr
OSLOADER	The Windows boot loader
RESUME	The resume application

The sections in the following table list data types and formats that are used by the different boot entry types.

Entry type	Description
All Entry Types	Data types that apply to any boot entry types.
Boot application types	
Boot Applications	Data types that apply to all boot applications.
Boot Manager	Data types that apply only to Boot Manager.
Windows Boot Loader	Data types that apply only to the Windows boot loader.
Memory Diagnostic Application	Data types that apply only to the memory diagnostic application.
Resume Application	Data types that apply only to resume applications.
Firmware Boot Manager	Data types that apply only to the firmware boot manager.
Ntldr	Data types that apply only to Ntldr.
Boot Sector Application	Data types that apply only to boot sector applications.
Other types	
Device Additional Options	Data types that apply to device additional options.
Custom Data Types	How to define custom data types.

All Entry Types

The data types in the following table are valid for any boot entry.

Data type name	Format	Description
PATH	string	The path to the application.
DEVICE	device	The device on which the application resides.
DESCRIPTION	string	A boot entry's description.
INHERIT	list	A space-delimited list of boot entries to be inherited.

Boot Applications

The data types in the following table apply to all boot applications. Data types that are specific to a particular application are listed in the following sections.

Type name	Format	Description
Display		
GRAPHICSMODEDISABLED	Boolean	TRUE disables graphics mode.
NOVESA	Boolean	TRUE disables VESA display modes.
Debugging		
BAUDRATE	integer	The baud rate for serial debugging.
BOOTDEBUG	Boolean	TRUE enables the boot debugger.
CHANNEL	integer	The channel for 1394 debugging.
DEBUGADDRESS	integer	The address of the serial port for serial debugging.
DEBUGPORT	integer	The serial port number for serial debugging.
DEBUGSTART	enum	The debug start type: ACTIVE, AUTOENABLE, or DISABLE.
DEBUGTYPE	enum	The debugging type: SERIAL, 1394, or USB.
NOUMEX	Boolean	TRUE causes user-mode exceptions to be ignored.
TARGETNAME	string	The target name for USB debugging.
Memory		
BADMEMORYACCESS	Boolean	TRUE enables an application to use the memory described by the bad memory list.
BADMEMORYLIST	integerlist	A space-delimited list of page frame numbers that describe faulty memory in the system.
TRUNCATEMEMORY	integer	A physical memory address. All memory at or above the specified address is disregarded.
Emergency Management Services		
BOOTEMS	Boolean	TRUE enables EMS.
EMSBAUDRATE	integer	The EMS baud rate
EMSPORT	integer	The EMS serial port number.
Devices and Hardware		
CONFIGACCESSPOLICY	enum	The access policy: DEFAULT or DISALLOWMMCONFIG.
FIRSTMEGABYTEPOLICY	enum	First megabyte policy: USENONE, USEALL, or USEPRIVATE.
LOCALE	string	The boot application's locale.
NOUMEX	Boolean	TRUE causes user-mode exceptions to be ignored.
Recovery		
RECOVERYENABLED	Boolean	TRUE enables the recovery sequence.
RECOVERYSEQUENCE	list	A space-delimited list of identifiers that defines the recovery sequence.
Verification		
TESTSIGNING	Boolean	TRUE enables prerelease test code signing certificates.

Windows Boot Manager

The following table lists the types that apply only to Windows Boot Manager. They can be used in addition to the standard boot applications types.

Data type name	Format	Description
Boot		
BOOTSEQUENCE	list	A space-separated list of identifiers that defines a one-time boot sequence.
DEFAULT	id	The default boot entry identifier.
TIMEOUT	integer	The Boot Manager's wait time, in seconds, after which Boot Manager selects the default boot entry.
Resume		
RESUME	Boolean	TRUE indicates that a resume operation should be attempted.
RESUMEOBJECT	id	The resume application identifier.
Display		
DISPLAYBOOTMENU	Boolean	TRUE enables the boot menu display.
DISPLAYORDER	list	A space-separated list of identifiers that defines Boot Manager's display order.
TOOLSDISPLAYORDER	list	A space-separated list of identifiers that defines the Boot Manager tools display order.

Windows Boot Loader

The types in the following table can be used only for Windows boot loader entries. They can be used in addition to the standard boot application types.

Data type name	Format	Description
Boot types		
BOOTLOG	Boolean	TRUE enables the system initialization log.
BOOTSTATUSPOLICY	enum	Boot status policy: DISPLAYALLFAILURES, IGNOREALLFAILURES, IGNORESHUTDOWNFAILURES, or IGNOREBOOTFAILURES.
LASTKNOWNGOOD	Boolean	TRUE enables the system to boot to the last known good configuration.
NOCRASHAUTOREBOOT	Boolean	TRUE disables automatic restart on crash.
QUIETBOOT	Boolean	TRUE disables the boot screen display.
RESUMEOBJECT	id	The identifier for the resume application that is associated with this operating system.
SAFEBOOT	enum	The safe boot option: MINIMAL, NETWORK, or DSREPAIR.
SAFEBOOTALTERNATESHELL	Boolean	TRUE specifies that the alternate shell should be used when the system is booted into Safe mode.
STAMPDISK	Boolean	Enables stamping of RAW disks during a WinPE boot. Available only on Windows Server 2008 Beta 3 and later.
SOS	Boolean	TRUE displays additional boot information.
WINPE	Boolean	TRUE enables the computer to boot to WinPE.

Data type name	Format	Description
Debugging and performance types		
DBGTRANSPORT	string	The file name for a private debugger transport.
DEBUG	Boolean	TRUE enables kernel debugging.
PERFMEM	integer	The size, in megabytes, of the buffer to be allocated for performance data logging.
Drivers, kernel, and system root types		
DRIVERLOADFAILUREPOLICY	enum	Driver load failure policy: FATAL or USEERRORCONTROL.
EMS	Boolean	TRUE enables kernel EMS.
KERNEL	string	The file name for a private kernel.
OSDEVICE	device	The device that contains the system root.
SYSTEMROOT	string	The fully-qualified path to the system root folder. It cannot contain environment variables.
Hardware abstraction layer (HAL) types		
DETECTHAL	Boolean	TRUE enables HAL and kernel detection.
HAL	string	The file name for a private HAL.
HALBREAKPOINT	Boolean	TRUE enables the special HAL breakpoint.
KERNEL	string	The file name for a private kernel.
Memory types		
INCREASEUSERVA	integer	The size of the user-mode address space for 32-bit versions of Windows. The default value is 2 GB. To specify a larger value, set INCREASEUSERVA to the size of the address space, in MB. The valid range for INCREASEUSERVA is 2048 to 3072. This data type is not used for 64-bit versions of Windows.
NOLOWMEM	Boolean	TRUE disables the use of low memory.
NX	enum	NX options: OPTIN, OPTOUT, ALWAYSON, or ALWAYSOFF.
PAE	enum	PAE options: DEFAULT, FORCEENABLE, or FORCEDISABLE.
REMOVEMEMORY	integer	The amount of memory to be removed from the total memory available to Windows.
Options		
ADVANCEDOPTIONS	Boolean	TRUE enables advanced options.
LOADOPTIONS	string	Any additional load options that are not covered by other data types.
OPTIONSEDIT	Boolean	TRUE enables the options editor.
Processors and APICs types		
CLUSTERMODEADDRESSING	integer	The maximum number of processors to include in a single Advanced Programmable Interrupt Controller (APIC) cluster.
CONFIGFLAGS	integer	Processor-specific configuration flags.
MAXPROC	Boolean	TRUE reports the maximum number of processors in the system.
NUMPROC	integer	The number of processors to be used.
ONECPU	Boolean	TRUE forces only the boot CPU to be used.
RESTRICTAPICCLUSTER	integer	The largest APIC cluster number that the system can use.
USEPHYSICALDESTINATION	Boolean	TRUE forces the physical APIC to be used.

Data type name	Format	Description
VESA, PCI, and VGA types		
MSI	enum	Message signaled interrupt (MSI) settings: DEFAULT or FORCEDISABLE.
USEFIRMWAREPCISETTINGS	Boolean	TRUE uses BIOS-configured PCI resources.
VGA	Boolean	TRUE forces the VGA display driver to be used.

Memory Diagnostic Application

The types in the following table apply only to memory diagnostic application boot entries. They can be used in addition to the standard boot application types.

Data type name	Format	Description
PASSCOUNT	integer	The number of iterations that to run.
TESTMIX	enum	The text mix: BASIC or EXTENDED.

Resume Application

The types in the following table apply to boot entries for the resume application. Boot application types can also apply to boot entries for the resume application.

Resume Application Types

Type name	Format	Description
Hibernation file		
FILEDEVICE	device	The device that contains the hibernation file.
FILEPATH	string	The path of the hibernation file.
Other		
ASSOCIATEDOSDEVICE	device	A Microsoft MS-DOS® device with a resume application.
CUSTOMSETTINGS	Boolean	TRUE allows resume loader to use custom settings.
PAE	enum	PAE settings: DEFAULT, FORCEENABLE, or FORCEDISABLE.

Firmware Boot Manager

The types in the following table apply only to boot entries for the firmware boot manager. There are no additional BCDEdit options for the firmware boot manager.

Data type name	Format	Description
Boot		
BOOTSEQUENCE	list	A space-separated list of identifiers that defines a one-time boot sequence.
DEFAULT	id	The default boot entry's identifier.
TIMEOUT	integer	The firmware boot manager's wait time, in seconds, after which Boot Manager selects the default boot entry.
Display		
DISPLAYORDER	list	A space-separated list of boot entry identifiers that defines the firmware boot manager's display order.

Ntldr

Ntldr is the legacy boot loader for earlier versions of Windows. Boot configuration options for Ntldr are defined in the boot.ini file. There are no additional BCDEdit options for Ntldr.

Boot Sector Application

Boot sector application allows the Windows Boot Manager to start a 16-bit real-mode loader application associated with a non-Windows operating system. There are no additional BCDEdit options for boot sector applications.

Device Additional Options

The types in the following table apply to device additional options boot entries.

Data type name	Format	Description
EXPORTASCD	Boolean	TRUE enables exporting the RAM disk as a CD.
RAMDISKIMAGELENGTH	integer	The RAM disk image length.
RAMDISKIMAGEOFFSET	integer	The RAM disk image offset.
RAMDISKSDIDEVICE	device	The device on which the SDI file is located.
RAMDISKSDIPATH	string	The path to the SDI file.
RAMDISKTFTPCLIENTPORT	integer	The RAM disk WIM file's Trivial File Transfer Protocol (TFTP) client port.

Custom Data Types

An element in a BCD store consists of two parts:

- A 4-byte integer that serves as the elements identifier. Options in BCDEdit such as DEBUG, BAUDRATE, or LOCALE are "friendly equivalents" to the integer that represents that element.
- A data payload. The type of data in the payload is encoded in the identifier.

A custom data type is a mechanism for specifying an option that does not have a friendly name equivalent. A custom type has the following format:

custom:xxxxxxxx
This is a custom type, with the format and meaning encoded in the specified 8-digit hex number.

Resources

Boot Configuration Data in Windows Vista
http://go.microsoft.com/fwlink/?LinkId=93005

BCD Reference
http://go.microsoft.com/fwlink/?LinkId=93006