MCI String Commands Contents

About MCI Commands

MCI commands divide into the following categories:

* *System commands* are interpreted directly by MCI rather than being relayed to a device.

* *Required commands* are supported by all MCI devices.

* *Basic commands* are optional commands. If a device uses a basic command, it must respond to all options for that command. If a device does not use a basic command, it will return "Action not available for this device."

* *Extended commands* are specific to a device type or device class, for example, videodisc players. These commands contain both unique commands and extensions to the required and basic commands.

Creating a Command String

There are three components associated with each command string: the command, the name of the device receiving the command, and the command arguments. A command string has the following form:

```
command   device_name   arguments
```

These components contain the following information:

* The *command* includes a command from the system, required, basic, or extended command set. Examples of commands include **open**, **close**, and **play**.

* The *device_name* designates the target of the *command*. MCI accepts the names of MCI device types and names of device elements for the *device_name*. An example of a name of device type is **cdaudio**. Other device types are listed in the topic "About MCI Device Types."

* The *arguments* specify the flags and parameters used by the *command*. The flags in the argument consist of the key words supported by the MCI command. Parameters consist of variable numeric or string data which apply to either the MCI command or flag. For example, the **play** command uses the arguments **from** *position* and **to** *position* to indicate the positions to start and end playing. (The key words used as flags are indicated by bold font and variables parameters are indicated by the italic font.) You can list the flags used with a command in any order. When you use a flag that has a parameter associated with it, you must supply a value for the parameter.

As an example, the following command string instructs the CD audio player "cdaudio" to play the disc sequence located between 5000 milliseconds and 15000 milliseconds from the start of the disc:

```
play cdaudio from 5000 to 15000
```

Unspecified command arguments assume a default value. For example, if the flag *from* was unspecified in the previous example, the CD audio player would start playing at the current position.

Using MCI Data Types

You can use the following data types for the parameters in a string command:

* Strings--String data types are delimited by leading and trailing white space and quotation marks ("). MCI removes single quotation marks from a string. If you want to put a quotation mark in a string, use a set of two quotation marks where you want to embed your quotation mark. If you want to use an empty string, use two quotation marks to delimit the empty string.

* Signed long integers--Signed long integer data types are delimited by leading and trailing white space. Unless otherwise specified, integers can be positive or negative. If using negative integers, do not put any white space between the negative sign and the first digit.

* Rectangles--Rectangle data types are an ordered list of four signed short values. White space delimits this data type as well as separates each integer in the list.

About MCI Device Types

Your application identifies an MCI device by specifying an MCI *device type*. A device type indicates the physical type of device. The following table lists the MCI device types currently defined:

Device Type	Description
cdaudio	CD audio player
dat	Digital audio tape player
digitalvideo	Digital video in a window (not GDI based)
MMMovie	Multimedia movie player
other	Undefined MCI device
overlay	Overlay device (analog video in a window)
scanner	Image scanner
sequencer	MIDI sequencer
vcr	Videotape recorder or player
videodisc	Videodisc player
waveaudio	Audio device that plays digitized waveform files

Available MCI Device Drivers

The system software includes device drivers and command sets for the device types that are fundamental to many multimedia presentations. The system software includes the following MCI device drivers:

Device Type	Device Driver	Description
cdaudio	MCICDA.DRV	An MCI device driver for controlling compact disc audio.
MMMovie	MCIMMP.DRV	An MCI device driver for playing multimedia movie files.
sequencer	MCISEQ.DRV	An MCI device driver for playing MIDI audio files.
videodisc	MCIPIONR.DRV	An MCI device driver for controlling the Pioneer LD-V4200 videodisc player.
waveaudio	MCIWAVE.DRV	An MCI device driver for playing and recording waveform audio files.

The [mci] section of the SYSTEM.INI file lists the device types installed in the system. If you have a particular device type installed more than once, the device type names in the SYSTEM.INI file have integers appended to them. This creates unique names for each MCI device type entry. For example, if the "cdaudio" device type is installed twice, the names "cdaudio" and "cdaudio1" are used to create unique names for each occurrence of the device type. Each name usually refers to a different CD audio player in the system.

Opening a Device

Before using a device, you must initialize it with the **open** command. The number of devices you can have open depends on the amount of available memory. The syntax for the **open** command has the following form:

open *device_name* **shareable type** *device_type* **alias** *alias*

The parameters for the **open** command are:

Parameters	Description
device_name	Specifies the destination device or an MCI element (file) name.
shareable	Allows applications to share a common device or device element.
type *device_type*	Specifies the device type when the *device_name* refers to an MCI element.
alias *alias*	Specifies a replacement name for the device.

MCI classifies device drivers as *compound* and *simple*. Compound device drivers use a *device element*<196>a media element associated with a device<196>during operation. For most compound device drivers, the device element is the source or destination data file. For these file elements, the element name references a file and its path. Compound devices include animation devices, waveform audio devices, and MIDI sequencers.

Simple device drivers do not require a device element for playback. Simple devices include CD audio devices and videodisc devices.

Opening Simple Devices

Simple devices require only the *device_name* for operation. You don't need to provide any additional information (such as a name of a data file) to open these devices. For these devices, substitute the name of a device type obtained from the [mci] section of SYSTEM.INI for the *device_name*. For example, you can open a videodisc device with "open videodisc1."

If you want to open a specific simple device, you can use the filename of the device driver for the *device_name*. Opening a device with the filename will make an application device dependent and can prevent the application from running if the system configuration changes. When you specify the filename of the device driver, you can optionally include the .DRV extension but do not include the path to the file.

Opening Compound Devices

Depending upon your needs, there are three ways you can open a compound device:

* By specifying just the device type
* By specifying both the element name and the device type
* By specifying just the element name

To determine the capabilities of a device, you can open a device by specifying only the device type. When opened this way, most compound devices will only let you determine their capabilities and close them. For example, you can open the sequencer with "open sequencer."

You can specify the device type either by using the device type from the SYSTEM.INI file or by using the filename of the device driver. Opening a device with the filename of the device driver will make an application device dependent and can prevent the application from running if the system configuration changes. If you specify the filename of the device driver, you can optionally include the .DRV extension but do not include the path to the file.

To associate a device element with a particular device, you must specify the element name and device type. In the **open** command, substitute the element name for the *device_name*, add the **type** flag, and substitute the name of the device you want to use for *device_type*. This combination lets your application specify the MCI device it needs to use. For example, you can open a device element of the waveaudio device with "open train.wav type waveaudio."

To associate a default MCI device with a device element, you can specify just an element name. In this case, MCI uses the extension of the element name to select the default device from the list in the [mci extensions] section of the SYSTEM.INI file. The entries in the [mci extensions] section use the following form:

file extension = device type

MCI implicitly uses the device type if the extension is found. The following fragment shows a typical [mci extensions] section:

```
[mci extensions] wav=waveaudio
mid=sequencer
rmi=sequencer
mmm=mmmovie
```

With these definitions, MCI opens the default waveaudio device if you use "open train.wav."

Using the Shareable Flag

The **shareable** flag lets multiple applications or tasks access the same device (or element) and device instance concurrently. If your application opens a device or device element without the **shareable** flag, no other application can access it simultaneously. If your application opens a device or device element as shareable, other applications can also access it by also opening it as shareable. The shared device or device element gives each application the ability to change the parameters governing the operating state of the device or device element. Each time a device or device element is opened as shareable, MCI returns a unique device ID even though the IDs refer to the same instance.

If you make a device or device element shareable, your application should not make any assumptions about the state of a device. When working with shared devices, your application might need to compensate for changes made by other applications using the same services.

If a device can service only one open instance it will fail an open with the **shareable** flag.

While most compound device elements are not shareable, you can open multiple elements (where each element is unique), or you can open a single element multiple times. If you open a single file element multiple times, MCI creates an independent instance for each of them. Each file element you open this way must have a unique name. The **alias** flag lets you use a unique name for each element.

Using the Alias Flag

The **alias** flag specifies an alternate name for the given device. The alias provides a shorthand notation for compound devices with lengthy pathnames, and it lets you open multiple instances of the same file. If your application creates a device alias, it must use the alias rather than the device name for all subsequent references.

Opening New Device Elements

If you wish to create a new device element without specifying an element name, you can specify **new** as a *device_name*. MCI does not save a new file element until you save it with the **save** command. When creating a new file, you must include a device alias with the **open** command. The following commands open a new waveaudio device element, start and stop recording, save the file element, and close the device element:

```
open new type waveaudio alias capture
record capture
stop capture
save capture orca.wav
close capture
```

Closing a Device

The **close** command releases access to a device or device element. To help MCI manage the devices, your application must explicitly close each device or device element when it is finished with it.

Shortcuts and Variations for MCI Commands

The MCI string interface lets you use several shortcuts when working with MCI devices.

* Using All as a Device Name
* Using ! to Combine the Device Type and Device Element Name
* Automatic Open
* Automatic Close

Using All as a Device Name

You can specify **all** as a *device_name* for any command that does not return information. When you specify **all**, MCI sequentially sends the command to all devices opened by the current application. For example, "close all" closes all open devices and "play all" starts playing all devices opened by the task. Because MCI sequentially sends the commands to the MCI devices, there is a delay between when the first device receives the command and when the last device receives the command.

Using ! to Combine the Device Type and Device Element Name

You can eliminate the **type** flag in the **open** command if you combine the device type with the device element name. MCI recognizes this combination when you use the following syntax:

device_type ! *element_name*

The exclamation mark separates the device type from the element name. The following example opens the right.wav element with the waveaudio device:

```
open waveaudio ! right.wav
```

Automatic Open

If MCI cannot identify the *device_name* of a command as an open device, MCI tries to automatically open the specified device. The following items apply to devices automatically opened:

* Automatic open works only with the command-string interface.

* Automatic open does not let your application specify the **type** flag. Without the device type, MCI determines the device type from the [mci extensions] section of the WIN.INI file. If you want to use a specific device, you can combine the device type name with the device element name using the exclamation mark.

* Automatic open will fail for commands that are specific to custom device drivers. For example, the command to unlock the front panel of the Pioneer videodisc player will fail an automatic open. The command for unlocking the front panel is specific only to this videodisc player.

* A device automatically opened will not respond to a command that uses **all** as a device name.

Automatic Close

MCI automatically closes any device automatically opened using the command-string interface. MCI closes a device when the command completes, when you abort the command, when you request notification in a subsequent command, or when MCI detects a failure.

Using the Notify Flag

The **notify** flag directs the device to post an MM_MCINOTIFY message when the device completes an action. Your application must have a window procedure to process the MM_MCINOTIFY message for notification to have any effect. The *Programmer's Workbook* includes examples of window procedures that process the MM_MCINOTIFY message.

While the results of a notification are application dependant, the application's window procedure can act upon four possible conditions associated with the notify message:

* The notification will occur when the notification conditions are satisfied.
* The notification can be superseded.
* The notification can be aborted.
* The notification can fail.

A successful notification occurs when the conditions required for initiating the callback are satisfied and the command completed without interruption.

A notification is superseded when the device has a notification pending and you send it another notify request. When a notification is superseded, MCI resets the callback conditions to correspond to the notify request of the new command.

A notification is aborted when you send a new command that prevents the callback conditions set by a previous command from being satisfied. For example, sending the stop command cancels a notification pending for the "play to 500 notify" command. If your command interrupts a command that has a notification pending, and your command also requests notification, MCI will abort the first notification immediately and respond to the second notification normally.

A notification fails if a device error occurs while a device is executing the MCI command. For example, a notification fails when a hardware error occurs during a play command.

Using the Wait Flag and Break Command

When a command uses the **wait** flag, MCI returns control to the calling application when the target device completes the command. You can cancel a wait operation by pressing a break key. By default, MCI defines this key as <MS>CTRL+BREAK. (You can redefine this key with the **break** command.) When you cancel the wait operation, if possible, MCI returns control to the application and does not interrupt the command associated with the **wait** flag. For example, breaking the command "play to 500 wait" cancels the wait operation without interrupting the play operation.

Obtaining Information From MCI Devices

Every device responds to the **capability**, **status**, and **info** commands. These commands obtain information about the device. For example, your application can determine if a videodisc requires a device element by using the command "capability videodisc compound file." (This example returns **false**.) The flags listed for the required and basic commands provide a minimum amount of information about a device. Many devices supplement the required and basic flags with extended flags to provide additional information about the device.

When you request information with the **capability**, **status**, or **info** command, the argument list can contain only one flag requesting information. The string interface can only return one string or value in response to a command requesting information.

Using the Play Command

The **play** command starts playing a device. Without any flags, the **play** command starts playing from the current position and plays until the command is halted or until the end of the media or file is reached. For example, "play cdaudio" starts playing an audio disc from the position where it was stopped.

Most devices that support the **play** command also support the **from** and **to**. These flags indicate the position at which the device should start and stop playing. For example, "play cdaudio from 0" plays the audio disc from the beginning of the first track. The units assigned to the position value depend on the device. For example, the position is normally specified in frames for CAV videodiscs, and in milliseconds for digital audio.

As an extended command, devices add flags to use the capabilities of a particular device. For example, the **play** command for videodisc devices adds the flags **fast**, **slow**, **reverse**, and **scan**.

Using the Stop, Pause, and Resume Commands

The **stop** command suspends the playing or recording of a device. Many devices include the basic command **pause**, which also suspends these sessions. The difference between **stop** and **pause** depends on the device. Usually **pause** suspends operation but leaves the device ready to resume playing or recording immediately.

Using **play** or **record** to restart a device will reset the **to** and **from** positions specified before the device was paused or stopped. Without the **from** flag, these commands reset the start position to the current position. Without the **to** flag, they reset the end position to the end of the media. If you want to continue playing or recording but want to stop at a position previously specified, use the **to** flag with these commands and repeat the position value.

Some devices include the **resume** command to restart a paused device. This command does not change the **to** and **from** positions specified with the **play** or **record** command which preceded the **pause** command.

break (System)

Syntax: **break *item***

Specifies a key to abort a **wait** command. One of the following *items* modifies **break**:

on *virtual_key*

Specifies the *virtual_key* code of the that aborts the **wait**. When the key is pressed, the device returns control to the application. If possible, the command continues execution. Substitute a Windows virtual key code for *virtual_key*.

off

Disables the current break key.

break (System)

Command	Arguments
break *device_name*	**{off | on** *virtual key code***}**
	[notify]
	[wait]

sound (System)

Syntax: **sound**

The device name field of this command specifies a sound from the [sounds] section of WIN.INI to play. If it is not found, MCI uses the SystemDefault sound.

sound (System)

Command	Arguments
sound *sound_name*	**[notify]**
	[wait]

sysinfo (System)

Syntax: **sysinfo *item***

Obtains MCI system information. One of the following items modifies **sysinfo**:

installname

Returns the name listed in the SYSTEM.INI file used to install the device.

quantity

Returns the number of MCI devices listed in the SYSTEM.INI file of the type specified in the device name field. The device name must be a standard MCI device type. Any digits after the name are ignored. The special device name **all** returns the total number of MCI devices in the system.

quantity open

Returns the number of open MCI devices of the type specified in the device name field. The device name must be a standard MCI device type. Any digits after the name are ignored. The name **all** returns the total number of MCI devices in the system that are open.

name *index*

Returns the name of an MCI device. The *index* ranges from 1 to the number of devices of that type. If **all** is specified for the device name, *index* ranges from 1 to the total number of devices in the system.

name *index open*

Returns the name of an open MCI device. The *index* ranges from 1 to the number of open devices of that type. If **all** is specified for the device name, *index* ranges from 1 to the total number of open devices in the system.

sysinfo (System)

Command	Arguments
sysinfo *device_name*	**{installname** **| name** *index* **| name** *index***open** **| quantity** **| quantity open}** **[notify]** **[wait]**

capability (Required)

Syntax: **capability *item***

Requests information about a particular capability of a device. While other capabilities are defined for specific devices and device types, the following *items* are always available:

can eject

Returns **true** if the device can eject the media.

can play

Returns **true**if the device can play.

can record

Returns **true** if the device supports recording.

can save

Returns **true** if the device can save data.

compound device

Returns **true** if the device requires an element name.

device type

Returns one of the following:

animation	**scanner**
cdaudio	**sequencer**
dat	**vcr**
digitalvideo	**videodisc**
other	**waveaudio**
overlay	

has audio

Returns **true** if the device supports audio playback.

has video

Returns **true** if the device supports video.

uses files

Returns **true** if the element of a compound device is a file pathname.

capability (Required)

Command	Arguments
capability *device_name*	**{can eject** **| can play** **| can record** **| can save** **| compound device** **| device type** **| has audio** **| has video** **| uses files}** **[notify]** **[wait]**

close (Required)

Syntax: **close**

When sent to a simple device, closes the device. When sent to a compound device element, closes the element.

close (Required)

Command	Arguments
close *device_name*	**[notify]**
	[wait]

info (Required)

Syntax: **info *item***

Fills a user-supplied buffer with information. The following *item* modifies info:

product

Returns a null-terminated string with description of the hardware associated with a device. This usually includes the manufacturer and model information.

info (Required)

Command	Arguments
info *device_name*	**[product]**
	[notify]
	[wait]

open (Required)

Syntax: **open *items***

Initializes the device. The following optional *items* modify open:

alias *device_ alias*

Specifies an alternate name for the given device. If specified, it must be used for subsequent references to the device.

shareable

Initializes the device or element as shareable. Subsequent attempts to open it fail unless you specify **shareable** in both the original and later **open** commands. MCI returns an error if it is already open and not shareable.

type *device_ type*

Specifies the device type of a device element. As an alternative to **type**, MCI can use the [mci extension] entries in the SYSTEM.INI file to select the device based on the extension used by the device element.

open (Required)

Command	Arguments
open *device_name*	**[alias** *device_alias***]**
	[shareable]
	[type *device_type***]**
	[notify]
	[wait]

status (Required)

Syntax: **status *item***

Obtains status information for the device. One of the following *items* modifies status:

mode

Returns the current mode of the device. All devices can return: **not ready**, **paused**, **playing**, and **stopped**. The modes **open**, **parked**, **recording**, and **seeking** are device dependent.

ready

Returns **true** if the device is ready.

status (Required)

Command	Arguments
status *device_name*	**[mode | ready]**
	[notify]
	[wait]

Load (Basic)

Syntax: **load**

Load a device element from disk. The following optional *item* modifies **load**:

filename

Specifies the source path and file.

Load (Basic)

Command	Arguments
load *device_name*	**{***file_name***}**
	[notify]
	[wait]

pause (Basic)

Syntax: **pause**

Stop playing.

pause (Basic)

Command	Arguments
pause *device_name*	**[notify]**
	[wait]

play (Basic)

Syntax: **play *items***
Start playing the device. The following optional *items* modify **play**:

from *position*
Specifies the position to start playing. If **from** is omitted, the play starts from the current position.

to *position*
Specifies the position to stop playing. If **to** is omitted, the play stops at the end of the media.

play (Basic)

Command	Arguments
play *device_name*	**[from***position***]**
	[to *position***]**
	[notify]
	[wait]

record (Basic)

Syntax: **record *items***

Start recording data. All data recorded after a file is opened is discarded if the file is closed without saving it. The following optional *items* modify **record**:

insert

Specifies that new data is added to the device element at the current position.

from *position*

Specifies the position to start recording. If **from** is omitted, the device starts recording at the current position.

to *position*

Specifies the position to stop recording. If **to** is omitted, the device records until a **stop** or **pause** command is received.

overwrite

Specifies that new data will replace data in the device element.

record (Basic)

Command	Arguments
record *device_name*	**[from** *position***]**
	[to *position***]**
	[insert | overwrite]
	[notify]
	[wait]

resume (Basic)

Syntax: **resume**

Resumes playing or recording on a paused device.

resume (Basic)

Command	Arguments
resume *device_name*	**[notify]**
	[wait]

save (Basic)

Syntax: **save *item***

Saves the MCI element. The following optional *item* modifies **save**:

filename

Specifies the destination path and file.

save (Basic)

Command	Arguments
save *device_name*	**[***file_name***]**
	[notify]
	[wait]

seek (Basic)

Syntax: **seek *item***

Moves to the specified position and stops. One of the following is required for *item*:

to *position*

Specifies the position to stop the seek.

to start

Specifies to seek to the start of the media or device element.

to end

Specifies to seek to the end of the media or device element.

seek (Basic)

Command	Arguments
seek *device_name*	**{to** *position* **| to start | to end}**
	[notify]
	[wait]

set (Basic)

Syntax: **set *items***
Sets the various control *items*:

audio all off
Disables audio output.

audio all on
Enables audio output.

audio left off
Disables audio output.

audio left on
Enables audio output.

audio right off
Disables audio output.

audio right on
Enables audio output.

door closed
Loads the media and closes the door if possible.

door open
Opens the door and ejects the media if possible.

time format milliseconds
Sets the time format to milliseconds. All position information is this format after this command. You can abbreviate milliseconds as **ms**.

video off
Disables video output.

video on
Enables video output.

set (Basic)

Command	Arguments
set *device_name*	**[audio all off** **| audio all on** **| audio left off** **| audio left on** **| audio right off** **| audio right on** **| video off** **| video on]** **[door closed | door open]** **[time format milliseconds | time format ms]** **[notify]** **[wait]**

status (Basic)

Syntax: **status *item***

Obtains status information for the device. One of the following *items* modifies **status**:

current track
Returns the current track.

length
Returns the total length of the media.

length track *track_number*
Returns the length of the track specified by *track_number*.

number of tracks
Returns the number of tracks on the media.

position
Returns the current position.

position track *track_number*
Returns the position of the start of the track specified by *track_number*.

start position
Returns the starting position of the media or device element.

time format
Returns the time format.

status (Basic)

Command	Arguments
status *device_name*	**{current track** **| length** **| length track** *track_number* **| mode** **| number of tracks** **| position** **| position track** *track_number* **| ready** **| start position** **| time format}** **[notify]** **[wait]**

stop (Basic)

Syntax: **stop**
Stops the device.

stop (Basic)

Command	Arguments
stop *device_name*	**[notify]**
	[wait]

capability (Animation)

Syntax: **capability *item***

Requests information about the capabilities of the graphics driver. The *item* is one of the following:

can eject
Returns **true** if the device can eject the media. The MCIMMP movie player returns **false**.

can play
Returns **true** if the device can play. The MCIMMP movie player returns **true**.

can record
Returns **false**. Animation and movie player devices cannot record.

can reverse
Returns **true** if the animation or movie player device can play in reverse.

can save
Returns **false**. Animation and movie player devices cannot save data.

can stretch
Returns **true** if the device can stretch frames to fill a given display rectangle. The MCIMMP movie player returns **false**.

compound device
Returns **true** if the device requires an element name.

device type
Animation and movie player devices return **animation**.

fast play rate
Returns fast play rate in frames per second.

has audio
Returns **true** if the device supports audio playback. The MCIMMP movie player has audio.

has video
Returns **true**. Animation and movie devices are video devices.

normal play rate
Returns normal play rate in frames per second.

slow play rate
Returns the slow play rate in frames per second.

uses files
Returns **true** if the element of a compound device is a file pathname.

uses palettes
Returns **true** if the device uses palettes. The MCIMMP movie player returns **true**.

windows
Returns the number of windows the device can support. The MCIMMP movie player returns 8.

capability (Animation)

Command	Arguments
capability *device_name*	**{can eject** **| can play** **| can record** **| can reverse** **| can save** **| can stretch** **| compound device** **| device type** **| fast play rate** **| has audio** **| has video** **| normal play rate** **| slow play rate** **| uses files** **| uses palettes** **| windows}** **[notify]** **[wait]**

close (Animation)

Syntax: **close**

Closes a device element and any resources associated with it.

close (Animation)

Command	Arguments
close *device_name*	**[notify]**
	[wait]

info (Animation)

Syntax: **info *item***

Fills a user-supplied buffer with information. One of the following optional *item* modifies **info**:

file

Returns the name of the file used by the animation device or movie player in a null-terminated string.

product

Returns the product name and model of the current device in a null-terminated string. The MCIMMP movie player returns **Microsoft Multimedia Movie Player**.

window text

Returns the caption of the window used by the device.

info (Animation)

Command	Arguments
info *device_name*	**[file | product | window text]**
	[notify]
	[wait]

open (Animation)

Syntax: **open *items***
Initializes the animation device. The following optional *items* modify **open**:

alias *device_ alias*
Specifies an alternate name for the animation or movie player element. If specified, it must also be used for subsequent references.

nostatic
Indicates that the device should reduce the number of static (system) colors in the palette. Reducing the number of static colors increases the number of colors controlled by the animation. The MCIMMP movie player reduces the static colors to black and white while in the foreground.,

parent *hwnd*
Specifies the window handle of the parent window.

shareable
Initializes a device element as shareable. Subsequent attempts to open it fail unless you specify **shareable** in both the original and subsequent **open** commands. MCI returns an invalid device error if it is already open and not shareable. The MCIMMP movie player does not allow shared files.

style *style_type*
Indicates a window style.

style child
Opens a window with a child window style.

style overlapped
Opens a window with an overlapped window style.

style popup
Opens a window with a popup window style.

type *device_ type*
Specifies the device type of the device element. MCI reserves MMMovie for the movie player device type. As an alternative to **type**, MCI can use the [mci extension] entries in the SYSTEM.INI file to select the controlling device based on the extension used by the device element.

open (Animation)

Command	Arguments
open *device_name*	**[alias** *device_alias***]**
	[nostatic]
	[parent *hwnd***]**
	[shareable]
	[style child **| style overlapped** **| style popup]**
	[type *device_type***]**
	[notify]
	[wait]

pause (Animation)

Syntax: **pause**

Pauses playback of the animation if it is playing. If the animation is stopped, **pause** displays the animation if it is not visible and in the foreground.

pause (Animation)

Command	Arguments
pause *device_name*	**[notify]**
	[wait]

play (Animation)

Syntax: **play *items***
Starts playing the animation sequence. The following optional *items* modify **play**:

fast
Plays the animation sequence at a fast rate.

from *position*
Specifies the frame at which to start playing. If **from** is omitted, play starts at the current frame.

to *position*
Specifies the frame at which to stop playing. If **to** is omitted, play stops at the end frame.

reverse
Indicates that the play direction is backwards.

scan
Plays the animation sequence as fast as possible without disabling video.

slow
Plays the animation sequence at a slow rate.

speed *fps*
Plays the animation sequence at the specified speed *fps*. Speed is specified in frames per second.

play (Animation)

Command	Arguments
play *device_name*	**[fast | slow | speed** *fps***]**
	[from *position***]**
	[reverse]
	[to *position***]**
	[scan]
	[notify]
	[wait]

put (Animation)

Syntax: **put *items***

Defines the area of the source image and destination window used for display. One of the following *items* modify **put**:

destination

Sets the whole window as the destination window.

destination at *rectangle*

Specifies a rectangle for the area of the window used to display the image. The *rectangle* coordinates are relative to the window origin and are specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner, and the coordinates *X2*, *Y2* specify the width and height of the rectangle. When an area of the display window is specified, and the device supports stretching, the source image is stretched to the destination offset and extent.

source

Selects the whole image for display in the destination window.

source at *rectangle*

Specifies a rectangle for the image area used for display. The *rectangle* coordinates are relative to the image origin and are specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner, and the coordinates *X2*, *Y2* specify the width and height of the rectangle. When an area of the source image is specified, and the device supports stretching, the source image is stretched to the destination offset and extent.

put (Animation)

Command	Arguments
put *device_name*	**{destination** **| destination at** *destination_rectangle* **| source** **| source at** *source_rectangle***}** **[notify]** **[wait]**

realize (Animation)

Syntax: **realize *item***

Tells the device to select and realize its palette into a display context of the displayed window. One of the following *items* modifies **realize**:

background

Realizes the palette as a background palette. The MCIMMP movie player does not support this option.

normal

Realizes the palette normally. The MCIMMP movie player does not support this option.

realize (Animation)

Command	Arguments
realize *device_name*	**{background | normal}**
	[notify]
	[wait]

resume (Animation)

Syntax: **resume**

Resumes playing. The MCIMMP movie player does not support this option.

resume (Animation)

Command	Arguments
resume *device_name*	**[notify]**
	[wait]

seek (Animation)

Syntax: **seek *item***

Moves to the specified position and stops. One of the following is required for *item*:

to *position*

Specifies the position to stop the seek.

to start

Specifies to seek to the start of the device element.

to end

Specifies to seek to the end of the device element.

seek (Animation)

Command	Arguments
seek *device_name*	**[to** *position* **| to start | to end]**
	[notify]
	[wait]

set (Animation)

Syntax: **set *items***
Sets the various control *items*:

audio all off
Disables audio output.

audio all on
Enables audio output.

audio left off
Disables output to the left audio channel.

audio left on
Enables output to the left audio channel.

audio right off
Disables output to the right audio channel.

audio right on
Enables output to the right audio channel.

time format frames
Sets the time format to frames. All position information is specified in frames following this command. When the device is opened, frames is the default mode.

time format milliseconds
Sets the time format to milliseconds. All position information is this format after this command. You can abbreviate milliseconds as **ms**. The MCIMMP movie player does not support this option.

video off
Disables video output. The MCIMMP movie player does not support this option.

video on
Enables video output. The MCIMMP movie player does not support this option.

set (Animation)

Command	Arguments
set *device_name*	**[audio all off** **| audio all on** **| audio left off** **| audio left on** **| audio right off** **| audio right on** **| video off** **| video on]** **[time format milliseconds** **| time format ms** **| time format frames]** **[notify]** **[wait]**

status (Animation)

Syntax: **status *item***

Obtains status information for the device. One of the following *items* modifies **status**:

current track
Returns the current track. The MCIMMP movie player returns 1.

forward
Returns **true** if the play direction is forward or if the device is not playing.

length
Returns the total number of frames.

length track *track_number*
Returns the total number of frames in the track specified by *track_number*.

media present
Returns **true** if the media is inserted in the device; otherwise it returns **false**.

mode
Returns **not ready**, **paused**, **playing**, **seeking**, or **stopped** for the current mode.

number of tracks
Returns the number of tracks on the media. The MCIMMP movie player returns 1.

palette handle
Returns the handle of the palette used for the animation in the low-order word of the return value.

position
Returns the current position.

position track *number*
Returns the position of the start of the track specified by *number*.

ready
Returns **true** if the device is ready.

speed
Returns the current speed of the device in frames per second.

start position
Returns the starting position of the media or device element.

time format
Returns the current time format.

window handle
Returns the handle of the window used for the animation in the low-order word of the return value.

status (Animation)

Command	Arguments
status *device_name*	**{current track** **| forward** **| length** **| length track** *track_number* **| media present** **| mode** **| number of tracks** **| palette handle** **| position** **| position track** *track_number* **| ready** **| speed** **| start position** **| time format** **| window handle}** **[notify]** **[wait]**

step (Animation)

Syntax: **step *item***

Step the play one or more frames forward or reverse. The default action is to step one frame forward. The following *items* modify **step**:

by *frames*

Indicates the number of frames to step.

reverse

Step the frames in reverse.

step (Animation)

Command	Arguments
step *device_name*	**[by** *frames***]**
	[reverse]
	[notify]
	[wait]

stop (Animation)

Syntax: **stop**

Stops playing.

stop (Animation)

Command	Arguments
stop *device_name*	**[notify]**
	[wait]

update (Animation)

Syntax: **update *item***

Repaints the current frame into the specified display context. The following *items* modifies **update**:

at *rectangle*
Specifies the clipping rectangle.

hdc *hdc*
Specifies the handle of the display context to paint.

update (Animation)

Command	Arguments
update *device_name*	**[at** *update_rect***]**
	[hdc *hdc***]**
	[notify]
	[wait]

where (Animation)

Syntax: **where**

Obtains the rectangle specifying the source or destination area. One of the following *items* modifies **where**:

destination

Requests the destination offset and extent.

source

Requests the source offset and extent.

where (Animation)

Command	Arguments
where *device_name*	**{destination | source}**
	[notify]
	[wait]

window (Animation)

Syntax: **window *item***

Tells the animation or movie player to use a given window to display the images instead of the default window created by the driver. By default, these devices should create a window when opened but should not display it until they receive the **play** command. Applications providing window handles should manage the display issues that result when the window is sized or when the window handle is switched during play. Several flags manipulate the window. Since the **status** command can obtain the handle to the current display window, you can use the standard window functions instead. The following *items* modify **window**:

handle *window_handle*
Specifies the handle of the destination window used as an alternate to the default window.

handle default
Specifies that the animation device or movie player should set the current display window back to the driver's default window.

state hide
Hides the current display window.

state iconic
Displays the window as iconic.

state maximized
Maximizes the current display window.

state minimize
Minimizes the specified window and activates the top-level window in the window-manager's list.

state minimized
Minimizes the current display window.

state no action
Displays a display window in its current state. The window that is currently active remains active.

state no activate
Displays a display window in its most recent size and state. The window that is currently active remains active.

state normal
Activates and displays the current display window in its original size and position.

state show
Shows the current display window.

text *caption*
Specifies the *caption* for the display window.

window (Animation)

Command	Arguments
window *device_name*	**[handle** *window_handle* **| handle default]** **[state hide** **| state iconic** **| state maximized** **| state minimize** **| state minimized** **| state no action** **| state no activate** **| state normal** **| state show** **[text** *caption_text***]** **[notify]** **[wait]**

capability (CD Audio)

Syntax: **capability *item***

Requests information about the capabilities of the CD audio device. One of the following *items* is required:

can eject
Returns **true** if the CD audio device can eject the media.

can play
Returns **true** if the CD audio device can play the media.

can record
Returns **false**. CD audio devices cannot record.

can save
Returns **false**. CD audio devices cannot save data.

compound device
Returns **false**. CD audio devices are simple devices.

device type
Returns CD audio.

has audio
Returns **true**.

has video
Returns **false**. CD audio devices do not support video.

uses files
Returns **false**. Simple devices do not use files.

capability (CD Audio)

Command	Arguments
capability *device_name*	**{can eject** **| can play** **| can record** **| can save** **| compound device** **| device type** **| has audio** **| has video** **| uses files}**

close (CD Audio)

Syntax: **close**

Closes the device.

close (CD Audio)

Command	Arguments
close *device_name*	**[notify]**
	[wait]

info (CD Audio)

Syntax: **info *item***

Fills a user-supplied buffer with information. One of the following optional *items* modifies **info**:

product

Returns the product name and model of the current audio device. The MCICDA device driver returns **CD Audio Player**.

info (CD Audio)

Command	Arguments
info *device_name*	**[product]**
	[notify]
	[wait]

open (CD Audio)

Syntax: **open *items***

Initializes the device. MCI reserves cdaudio for the compact disc audio device type. The following optional *items* modify **open**:

alias *device_ alias*

Specifies an alternate name for the given device. If specified, it must also be used for subsequent references.

shareable

Initializes the device as shareable. Subsequent attempts to open it fail unless you specify **shareable** in both the original and subsequent **open** commands. MCI returns an error if it is already open and not shareable.

open (CD Audio)

Command	Arguments
open *device_name*	**[alias** *device_alias***]**
	[shareable]
	[notify]
	[wait]

pause (CD Audio)

Syntax: **pause**

Pauses playing. This is the same as the **stop** command for the MCICDA device driver.

pause (CD Audio)

Command	Arguments
pause *device_name*	**[notify]**
	[wait]

play (CD Audio)

Syntax: **play *items***
Starts playing audio. The following optional *items* modify **play**:

from *position*
Specifies the position to start playing. If **from** is omitted, the play starts at the current position. If the **from** position is greater than the end position of the disc, or if the **from** position is greater than the **to** position, MCI returns an error.

to *position*
Specifies the position to stop playing. If the **to** position is greater than the length of the disc, MCI returns an error.

play (CD Audio)

Command	Arguments
play *device_name*	**[from** *position***]**
	[to *position***]**
	[notify]
	[wait]

resume (CD Audio)

Syntax: **resume**

Resumes playing of a paused device. The MCICDA device driver does not support this option.

resume (CD Audio)

Command	Arguments
resume *device_name*	**[notify]**
	[wait]

seek (CD Audio)

Syntax: **seek *item***

Moves to the specified location on the disc. If already playing or recording, the device stops. One of the following optional *items* modifies **seek**:

to *position*

Specifies the destination position for the seek. If it is greater than the length of disc, MCI returns an out-of- range error.

to start

Specifies seek to the start of the audio data on the disc.

to end

Specifies seek to the end of the audio data on the disc.

seek (CD Audio)

Command	Arguments
seek *device_name*	**[to** *position* **| to start | to end]**
	[notify]
	[wait]

set (CD Audio)

Syntax: **set *items***
Sets the various control *items*:

audio all off
Disables audio output.

audio all on
Enables audio output.

audio left off
Disables output to the left audio channel.

audio left on
Enables output to the left audio channel.

audio right off
Disables output to the right audio channel.

audio right on
Enables output to the right audio channel.

door closed
Retracts the tray and closes the door if possible.

door open
Opens the door and ejects the tray if possible.

time format milliseconds
Sets the time format to milliseconds. All position information is in this format after this command. You can abbreviate milliseconds as **ms**.

time format msf
Sets the time format to minutes, seconds, and frames. When time or position values are used, they are expressed as *mm*:*ss*:*ff*, where *mm* is minutes, *ss* is seconds, and *ff* is frames. This is the default format for CD audio. On input, *ff* can be omitted if 0, and *ss* can be omitted if both it and *ff* are 0. For example, 3, 3:0, and 3:0:0 are valid ways to express 3 minutes. These fields have the following maximum values: Minutes 99 Seconds 59 Frames 74

time format tmsf
Sets the time format to tracks, minutes, seconds, and frames. When time or position values are used, they are expressed as *tt*:*mm*:*ss*:*ff*, where *tt* is tracks, *mm* is minutes, *ss* is seconds, and *ff* is frames. All position information is in this format after this command. On input, *ff* can be omitted if 0, *ss* can be omitted if both it and *ff* are 0, and *mm* can be omitted if it, *ss*, and *ff* are 0. For example, 3, 3:0, 3:0:0, 3:0:0:0 all specify track 3. These fields have the following maximum values: Tracks 99 Minutes 99 Seconds 59 Frames 74

set (CD Audio)

Command	Arguments
set *device_name*	**[audio all off** **| audio all on** **| audio left off** **| audio left on** **| audio right off** **| audio right on** **| video off** **| video on]** **[door closed | door open]** **[time format milliseconds** **| time format ms** **| time format msf** **| time format tmsf]** **[notify]** **[wait]**

status (CD Audio)

Syntax: **status *item***

Obtains status information for the device. One of the following *items* modifies **status**:

current track
Returns the current track.

length
Returns the total length of the disc.

length track *track_number*
Returns the length of the track specified by *track_number*.

media present
Returns **true** if the disc is inserted in the drive; otherwise it returns **false**.

mode
Returns **not ready**, **open**, **paused**, **playing**, **seeking**, or **stopped** for the current mode of the device.

number of tracks
Returns the number of tracks on the disc.

position
Returns the current position.

position track *track_no*
Returns the starting position of the track specified by *track_no*.

ready
Returns **true** if the device is ready.

start position
Returns the starting position of the disc or device element.

time format
Returns the current time format.

status (CD Audio)

Command	Arguments
status *device_name*	**{current track** **| length** **| length track** *track_number* **| media present** **| mode** **| number of tracks** **| position** **| position track** *track_number* **| ready** **| start position** **| time format}** **[notify]** **[wait]**

stop (CD Audio)

Syntax: **stop**

Stops playing.

stop (CD Audio)

Command	Arguments
stop *device_name*	**[notify]**
	[wait]

capability (MIDI Sequencer)

Syntax: **capability *item***

Requests information about the capabilities of the MIDI sequencer. One of the following *items* is required:

can eject
Returns **false**. Sequencers cannot eject.

can play
Returns **true**.

can record
Returns **true** if the sequencer can record MIDI data. The MCISEQ sequencer cannot record and returns **false**.

can save
Returns **true** if the sequencer can save MIDI data. The MCISEQ sequencer cannot save data and returns **false**.

compound device
Returns **true**; sequencers are compound devices.

device type
Returns sequencer.

has audio
Returns **true**. Sequencers support playback.

has video
Returns **false**. Sequencers do not support video.

uses files
Returns **true**. Sequencers use files for operation.

capability (MIDI Sequencer)

Command	Arguments
capability *device_name*	**{can eject** **| can play** **| can record** **| can save** **| compound device** **| device type** **| has audio** **| has video** **| uses files}** **[notify]** **[wait]**

close (MIDI Sequencer)

Syntax: **close**

Closes the sequencer element and the port and file associated with it.

close (MIDI Sequencer)

Command	Arguments
close *device_name*	**[notify]**
	[wait]

info (MIDI Sequencer)

Syntax: **info *item***

Fills a user-supplied buffer with information. One of the following optional *item* modifies **info**:

product

Returns the product name of the sequencer. The MCISEQ sequencer returns **MIDI Sequencer**.

info (MIDI Sequencer)

Command	Arguments
info *device_name*	**[product]**
	[notify]
	[wait]

open (MIDI Sequencer)

Syntax: **open *items***
Initializes the sequencer. The following optional *items* modify **open**:

alias *device_ alias*
Specifies an alternate name for the sequencer element. If specified, it must also be used for subsequent references.

shareable
Initializes the sequencer element as shareable. Subsequent attempts to open it fail unless you specify shareable in both the original and subsequent **open** commands. MCI returns an invalid device error if it is already open and not shareable. Files cannot be shared when using the MCISEQ sequencer.

type *device_ type*
Specifies the device type of the device element. MCI reserves sequencer for the MIDI sequencer device type. As an alternative to **type**, MCI can use the [mci extension] entries in the SYSTEM.INI file to select the sequencer based on the extension used by the device element.

open (MIDI Sequencer)

Command	Arguments
open *device_name*	**[alias** *device_alias***]**
	[shareable]
	[type *device_type***]**
	[notify]
	[wait]

pause (MIDI Sequencer)

Syntax: **pause**
Pauses playing.

pause (MIDI Sequencer)

Command	Arguments
pause *device_name*	**[notify]**
	[wait]

play (MIDI Sequencer)

Syntax: **play *items***
Starts playing the sequencer. The following optional *items* modify **play**:

from *position*
Specifies the position to start playing. If **from** is omitted, play starts at the current position.

to *position*
Specifies the position to stop playing. If **to** is omitted, play stops at the end of the file.

play (MIDI Sequencer)

Command	Arguments
play *device_name*	**[from** *position***]**
	[to *position***]**
	[notify]
	[wait]

record (MIDI Sequencer)

Syntax: **record *items***

Starts recording MIDI data. All data recorded after a file is opened is discarded if the file is closed without saving it. (The MCISEQ sequencer does not support recording.) The following optional *items* modify **record**:

insert
Specifies that new data is added to the device element.

from *position*
Specifies the position to start recording. If **from** is omitted, the device starts recording at the current position.

to *position*
Specifies the position to stop recording. If **to** is omitted, the device records until a **stop** or **pause** command is received.

overwrite
Specifies that new data will replace data in the device element.

record (MIDI Sequencer)

Command	Arguments
record *device_name*	**[from** *position***]**
	[to *position***]**
	[insert | overwrite]
	[notify]
	[wait]

resume (MIDI Sequencer)

Syntax: **resume**

Resumes playing or recording. The MCISEQ sequencer does not support this option.

resume (Sequencer)

Command	Arguments
resume *device_name*	**[notify]**
	[wait]

save (MIDI Sequencer)

Syntax: **save *item***

Saves the MCI element. (The MCISEQ sequencer does not support this option.) The following item *modifies* **save**:

filename

The *filename* specifies the destination path and file.

save (MIDI Sequencer)

Command	Arguments
save *device_name*	**[***file_name***]**
	[notify]
	[wait]

seek (MIDI Sequencer)

Syntax: **seek *item***
Moves to the specified position in the file. One of the following *items* is required:

to *position*
Specifies the final position for the seek.

to start
Specifies to seek to the start of the sequence.

to end
Specifies to seek to the end of the sequence.

seek (MIDI Sequencer)

Command	Arguments
seek *device_name*	**[to** *position* **| to start | to end]**
	[notify]
	[wait]

set (MIDI Sequencer)

Syntax: **set *items***
Sets the various control *items*:

audio all off
Disables audio output. The MCISEQ sequencer does not support this option.

audio all on
Enables audio output. The MCISEQ sequencer does not support this option.

audio left off
Disables output to the left audio channel. The MCISEQ sequencer does not support this option.

audio left on
Enables output to the left audio channel. The MCISEQ sequencer does not support this option.

audio right off
Disables output to the right audio channel. The MCISEQ sequencer does not support this option.

audio right on
Enables output to the right audio channel. The MCISEQ sequencer does not support this option.

master MIDI
Sets the MIDI sequencer as the synchronization source. Synchronization data is sent in MIDI format. The MCISEQ sequencer does not support this option.

master none
Inhibits the sequencer from sending synchronization data. The MCISEQ sequencer does not support this option.

master SMPTE
Sets the MIDI sequencer as the synchronization source. Synchronization data is sent in SMPTE format. The MCISEQ sequencer does not support this option.

offset *time*
Sets the SMPTE offset *time*. The offset is the beginning time of a SMPTE based sequence. The *time* is expressed as *hh*:*mm*:*ss*:*ff*, where *hh* is hours, *mm* is minutes, *ss* is seconds, and *ff* is frames.

port *port_number*
Sets the MIDI port receiving the MIDI messages. This command will fail if the port you are trying to open is being used by another application.

port mapper
Sets the MIDI mapper as the port receiving the MIDI messages. This command will fail if the MIDI mapper or a port it needs is being used by another application.

port none
Disables the sending of MIDI messages. This command also closes a MIDI port.

slave file
Sets the MIDI sequencer to use file data as the synchronization source. This is the default.

slave MIDI

Sets the MIDI sequencer to use incoming data MIDI for the synchronization source. The sequencer recognizes synchronization data with the MIDI format. The MCISEQ sequencer does not support this option.

slave none
Sets the MIDI sequencer to ignore synchronization data.

slave SMPTE
Sets the MIDI sequencer to use incoming MIDI data for the synchronization source. The sequencer recognizes synchronization data with the SMPTE format. The MCISEQ sequencer does not support this option.

tempo *tempo_value*
Sets the tempo of the sequence according to the current time format. For a ppqn-based file, the *tempo_value* is interpreted as beats per minute. For a SMPTE-based file, the *tempo_value* is interpreted as frames per second.

time format milliseconds
Sets the time format to milliseconds. All position information is specified as milliseconds following this command. The sequence file sets the default format to ppqn or SMPTE. You can abbreviate milliseconds as **ms**.

time format song pointer
Sets the time format to song pointer (sixteenth notes). This can only be performed for a sequence of division type ppqn.

time format SMPTE 24
Sets the time format to SMPTE 24 frame rate. All position information is specified in SMPTE format following this command. The sequence file sets the default format to ppqn or SMPTE.

time format SMPTE 25
Sets the time format to SMPTE 25 frame rate. All position information is specified in SMPTE format following this command. The sequence file sets the default format to ppqn or SMPTE.

time format SMPTE 30
Sets the time format to SMPTE 30 frame rate. All position information is specified in SMPTE format following this command. The sequence file sets the default format to ppqn or SMPTE.

time format SMPTE 30 drop
Sets the time format to SMPTE 30 drop frame rate. All position information is specified in SMPTE format following this command. The sequence file sets the default format to ppqn or SMPTE.

set (MIDI Sequencer)

Command	Arguments
set *device_name*	**[audio all off** **| audio all on** **| audio left off** **| audio left on** **| audio right off** **| audio right on** **| video off** **| video on]** **[master MIDI** **| master none** **| master SMPTE]** **[offset** *hmsf_value***]** **[port** *port_number* **| port mapper** **| port none]** **[slave file** **| slave MIDI** **| slave none** **| slave SMPTE]** **[tempo** *tempo_value***]** **[time format milliseconds** **| time format ms** **| time format smpte 24** **| time format smpte 25** **| time format smpte 30** **| time format smpte 30 drop** **| time format song pointer]** **[notify]** **[wait]**

status (MIDI Sequencer)

Syntax: **status *item***

Obtains status information for the MIDI sequencer. One of the following *items* modifies **status**:

current track
Returns the current track number. The MCISEQ sequencer returns 1.

division type
Returns one of the following file division type: **PPQN**, **SMPTE 24 frame**, **SMPTE 25 frame**, **SMPTE 30 drop frame**, or **SMPTE 30 frame**. Use this information to determine the format of the MIDI file, and the meaning of tempo and position information.

length
Returns the length of a sequence in the current time format. For ppqn files, this will be song pointer units. For SMPTE files, this will be expressed as *hh*:*mm*:*ss*:*ff*, where *hh* is hours, *mm* is minutes, *ss* is seconds, and *ff* is frames.

length track *track_number*
Returns the length of the sequence in the current time format. For ppqn files, this will be song pointer units. For SMPTE files, this will be expressed as *hh*:*mm*:*ss*:*ff*, where *hh* is hours, *mm* is minutes, *ss* is seconds, and *ff* is frames.

master
Returns **midi**, **none**, or **smpte** depending on the type of synchronization set.

media present
The sequencer returns **true**.

mode
Returns not ready, paused, playing, seeking, or stopped.

number of tracks
Returns the number of tracks. MCISEQ returns 1.

offset
Returns the offset of a SMPTE-based file. The offset is the start time of a SMPTE based sequence. The time is returned as *hh*:*mm*:*ss*:*ff*, where *hh* is hours, *mm* is minutes, *ss* is seconds, and *ff* is frames.

port
Returns the MIDI port number assigned to the sequence.

position
Returns the current position of a sequence in the current time format. For ppqn files, this will be song pointer units. For SMPTE files, this will be in colon form *hh*:*mm*:*ss*:*ff*, where *hh* is hours, *mm* is minutes, *ss* is seconds, and *ff* is frames.

position track *track_number* Returns the current position of the track specified by *track_number* in the current time format. For ppqn files, this will be song pointer units. For SMPTE files, this will be in colon form *hh*:*mm*:*ss*:*ff*, where *hh* is hours, *mm* is minutes, *ss* is seconds, and *ff* is frames. The MCISEQ sequencer returns 0.

ready
Returns **true** if the device is ready.

slave
Returns **file**, **midi**, **none**, or **smpte** depending on the type of synchronization set.

start position
Returns the starting position of the media or device element.

tempo
Returns the current tempo of a sequence in the current time format. For files with ppqn format, the tempo is in beats per minute. For files with SMPTE format, the tempo is in frames per second.

time format
Returns the time format.

status (MIDI Sequencer)

Command	Arguments
status *device_name*	**{current track** **| division type** **| length** **| length track** *track_number* **| master** **| media present** **| mode** **| number of tracks** **| offset** **| port** **| position** **| position track** *track_number* **| ready** **| slave** **| start position** **| tempo** **| time format}** **[notify]** **[wait]**

stop (MIDI Sequencer)

Syntax: **stop**

Stops playing.

stop (MIDI Sequencer)

Command	Arguments
stop *device_name*	**[notify]**
	[wait]

capability (Videodisc)

Syntax: **capability *item***

Reports the capabilities of the device. The return information is for the type of disc inserted unless **CAV** or **CLV** is used to override the format. If no disc is present then information is returned for **CAV** discs. The following optional *items* modify **capability**:

can eject
Returns **true** if the device can eject the disc. The MCIPIONR device returns **true**.

can play
Returns **true** if the device supports playing. The MCIPIONR device returns **true**.

can record
Returns **true** if the video device can record. The MCIPIONR device returns **false**.

can reverse
Returns **true** if the device can play in reverse, **false** otherwise. CLV discs return **false**.

can save
Returns **false**. MCI videodisc players cannot save data.

CAV
When combined with other items, **CAV** specifies that the return information applies to CAV format discs. This is the default if no disc is inserted.

CLV
When combined with other items, **CLV** specifies that the return information applies to CLV format discs.

compound device
Returns **false**. MCI videodisc players are simple devices.

device type
Returns videodisc.

fast play rate
Returns the standard fast play rate of the player in frames per second. Returns 0 if the device cannot play fast.

has audio
Returns **true** if the videodisc player has audio.

has video
Returns **true**.

normal play rate
Returns the normal play rate in frames per second. Returns 0 for CLV discs.

slow play rate
Returns the standard slow play rate in frames per second. Returns 0 if the device cannot play slow.

uses files
Returns **false**. Simple devices do not use files.

capability (Videodisc)

Command	Arguments
capability *device_name*	**{can eject** **| can play** **| can record** **| can reverse** **| can save** **| compound device** **| device type** **| fast play rate** **| has audio** **| has video** **| normal play rate** **| slow play rate** **| uses files}** **[CAV | CLV]** **[notify]** **[wait]**

close (Videodisc)

Syntax: **close**

Closes the device.

close (Videodisc)

Command	Arguments
close *device_name*	**[notify]**
	[wait]

escape (Videodisc)

Syntax: **escape *item***

Sends custom information to a device. The following *item* modifies **escape**:

command_string

Specifies the custom information sent to the device.

escape (Videodisc)

Command	Arguments
escape *device_name*	**{***command_string***}**
	[notify]
	[wait]

info (Videodisc)

Syntax: **info *item***

Fills a user-supplied buffer with information. The following optional *item* modifies **info**:

product

Returns the product name of the device that the peripheral is controlling. The MCIPIONR device returns **Pioneer LD-V4200.**

info (Videodisc)

Command	Arguments
info *device_name*	**[product]**
	[notify]
	[wait]

open (Videodisc)

Syntax: **open *items***

Initializes the device. MCI reserves videodisc for the videodisc device type. The following optional *items* modify **open**:

alias *device_ alias*

Specifies an alternate name for the given device. If specified, it must also be used for subsequent references.

shareable

Initializes the device as shareable. Subsequent attempts to open it fail unless you specify **shareable** in both the original and subsequent **open** commands. MCI returns an invalid device error if it is already open and not shareable.

open (Videodisc)

Command	Arguments
open *device_name*	**[alias** *device_alias***]**
	[shareable]
	[notify]
	[wait]

pause (Videodisc)

Syntax: **pause**

Stops playing. For CAV discs the video frame will freeze. For CLV discs, the player is stopped.

pause (Videodisc)

Command	Arguments
pause *device_name*	**[notify]**
	[wait]

play (Videodisc)

Syntax: **play *items***
Starts playing. The following optional *items* modify **play**:

fast
Indicates that the device should play faster than normal. To determine the exact speed on a particular player, use the **status speed** command. To specify the speed more precisely, use the **speed** flag.

slow
Indicates that the device should play slower than normal. To determine the exact speed on a particular player, use the **status speed** command. To specify the speed more precisely, use the **speed** flag. **Slow** applies only to CAV discs.

from *position*
Specifies the position to start playing. The default positions are in frames for CAV discs and in hours, minutes, and seconds for CLV discs. If **from** is omitted, play starts at the current position.

to *position*
Specifies the position to stop playing. The default positions are in frames for CAV discs and in hours, minutes, and seconds for CLV discs. If **to** is omitted, the play stops at the end of the disc.

reverse
Sets the play direction to backwards. This applies only to CAV discs.

scan
Indicates the play speed is as fast as possible, possibly with audio disabled. This applies only to CAV discs.

speed *integer*
Specifies the rate of play in frames per second (for example, speed 15 means 15 frames per second). This applies only to CAV discs.

play (Videodisc)

Command	Arguments
play *device_name*	**[fast | slow | speed** *fps***]**
	[from *position***]**
	[reverse]
	[scan]
	[to *position***]**
	[notify]
	[wait]

resume (Videodisc)

Syntax: **resume**

Resumes playing of a paused device. The MCIPIONR device driver does not support this command.

resume (Videodisc)

Command	Arguments	
resume *device_name*	**[notify]**	
		[wait]

seek (Videodisc)

Syntax: **seek *item***

Searches using fast forward or fast reverse with video and audio off. The following optional *items* modify **seek**:

reverse

Indicates the seek direction on CAV discs is backwards. This modifier is invalid if **to** is specified.

to *position*

Specifies the end position to stop the seek. If **to** is not specified, the seek continues until the end of the disc is reached.

to start

Specifies to seek to the start of the disc.

to end

Specifies to seek to the end of the disc.

seek (Videodisc)

Command	Arguments
seek *device_name*	**[reverse | to** *position* **| to start | to end]**
	[notify]
	[wait]

set (Videodisc)

Syntax: **set *items***
Sets the various control *items*:

audio all off
Disables audio output.

audio all on
Enables audio output.

audio left off
Disables output to the left audio channel.

audio left on
Enables output to the left audio channel.

audio right off
Disables output to the right audio channel.

audio right on
Enables output to the right audio channel.

door open
Opens the door and ejects the tray, if possible.

door closed
Retracts the tray and closes the door, if possible.

time format frames
Sets the position format to frames on CAV discs. All position information is specified in this format following this command. This is the default for CAV discs.

time format hms
Sets the time format to hours, minutes, and seconds. When time or position values are used, they are expressed as *h*:*mm*:*ss* where *h* is hours, *mm* is minutes, and *ss* is seconds. All position information is specified in this format following this command. On input, *h* may be omitted if 0, and *mm* may be omitted if both it and *h* are 0. This is the default for CLV discs.

time format milliseconds
Sets the position format to milliseconds. All position information is specified in this format following this command. You can abbreviate milliseconds as **ms**.

time format track
Sets the position format to tracks. All position information is specified in this format following this command.

video off
Turns the video off.

video on
Turns the video on.

set (Videodisc)

Command	Arguments
set *device_name*	**[audio all off** **| audio all on** **| audio left off** **| audio left on** **| audio right off** **| audio right on** **| video off** **| video on]** **[door closed | door open]** **[time format milliseconds** **| time format ms** **| time format frames** **| time format hms** **| time format track]** **[notify]** **[wait]**

spin (Videodisc)

Syntax: **spin *item***

Starts the disc spinning or stops the disc from spinning. One of the following *items* modifies **status**:

down

Stops the disc from spinning.

up

Starts the disc spinning.

spin (Videodisc)

Command	Arguments
spin *device_name*	**[down | up]**
	[notify]
	[wait]

status (Videodisc)

Syntax: **status *item***

Obtains status information for the device. One of the following *items* modifies **status**:

current track
Returns the current track (chapter) number.

disc size
Returns either 8 or 12 to indicate the size of the loaded disc in inches.

forward
Returns **true** if the play direction is forward or if the device is not playing; **false** if the play direction is backward.

length
Returns the total length of the disc.

length track *track_number* Returns the length of the track (chapter) specified by *track_number*.

media present
Returns **true** if a disc is inserted in the device, **false** otherwise.

media type
Returns either **CAV**, **CLV**, or **other** depending on the type of videodisc.

mode
Returns not ready, open, paused, parked, playing, seeking, or stopped.

number of tracks
Returns the number of tracks (chapters) on the disc. The MCIPIONR device does not support this option.

position
Returns the current position.

position track *track_number*
Returns the position of the start of the track (chapter) specified by *track_number*. The MCPIONR device does not support this option.

ready
Returns **true** if the device is ready.

side
Returns 1 or 2 to indicate which side of the disc is loaded.

speed
Returns the current speed in frames per second. The MCIPIONR videodisc player does not support this option.

start position
Returns the starting position of the disc.

time format
Returns the time format.

status (Videodisc)

Command	Arguments
status *device_name*	**{current track** **| disc size** **| forward** **| length** **| length track** *track_number* **| media present** **| media type** **| mode** **| number of tracks** **| position** **| position track** *track_number* **| ready** **| side** **| speed** **| start position** **| time format}** **[notify]** **[wait]**

step (Videodisc)

Syntax: **step *items***

Step the play one or more frames forward or backward. The default action is to step one frame forward. The **step** command applies only to CAV discs. The following *items* modifies **step**:

by *frames*

Specifies the number of *frames* to step. If a negative value is used, the **reverse** flag is ignored.

reverse

Step backward.

step (Videodisc)

Command	Arguments
step *device_name*	**[by]** *position*
	[reverse]
	[notify]
	[wait]

stop (Videodisc)

Syntax: **stop**

Stop playing.

stop (Videodisc)

Command	Arguments
stop *device_name*	**[notify]**
	[wait]

capability (Video Overlay)

Syntax: **capability *item***

Requests information about the capabilities of the video overlay device. The *item* is one of the following:

can eject
Returns **false**. Video overlay devices have no media to eject.

can freeze
Returns true if the device can freeze data in the frame buffer.

can play
Returns false.

can record
Returns false.

can save
Returns **true** if the device can save the current contents of the frame buffer to disk.

can stretch
Returns **true** if the device can stretch frames to fill a given display rectangle.

compound device
Returns **true** if the device requires an element name. Video overlay devices which support multiple windows may be compound devices.

device type
Returns overlay.

has audio
Returns **false** if the device supports audio playback.

has video
Returns **true**. Video overlay devices are video devices.

uses files
Returns **true** if the element of a compound device is a file pathname.

windows
Returns the number of simultaneous display windows the device can support.

capability (Video Overlay)

Command	Arguments
capability *device_name*	**{can eject** **| can freeze** **| can play** **| can record** **| can save** **| can stretch** **| compound device** **| device type** **| has audio** **| has video** **| uses files** **| windows}** **[notify]** **[wait]**

close (Video Overlay)

Syntax: **close**

Closes a video overlay element and any resources associated with it.

close (Video Overlay)

Command	Arguments	
close *device_name*	**[notify]**	
		[wait]

freeze (Video Overlay)

Syntax: **freeze *item***

Disables video acquisition to the frame buffer. This is supported only if **capability can freeze** returns **true**. The following optional *item* modifies **freeze**:

at *rectangle*

Specifies the rectangular region that will have video acquisition disabled. Irregular acquisition regions can be specified using the **freeze** and **unfreeze** commands. (Some video overlay devices will place limitations on the complexity of the region used for acquisition.) The *rectangle* region is relative to the video buffer origin and is specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner and the coordinates *X2*, *Y2* specify the width and height of the rectangle.

freeze (Video Overlay)

Command	Arguments
freeze *device_name*	**[at** *rectangle***]**
	[notify]
	[wait]

info (Video Overlay)

Syntax: **info *item***

Fills a user-supplied buffer with information. The following optional *item* modifies **info**:

file

Returns the name of the file used by the video overlay device.

product

Returns the product name and model of the current video overlay device.

window text

Returns the caption of the window used by the video overlay device.

info (Video Overlay)

Command	Arguments
info *device_name*	**[product | file | window text]**
	[notify]
	[wait]

load (Video Overlay)

Syntax: **load *item***

Loads the contents of the video buffer in a device specific format. The following optional *items* modify **load**:

filename

Specifies the file and pathname used to load the data.

at *rectangle*

Specifies a rectangle relative to the video buffer origin. The *rectangle* is specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner and the coordinates *X2*, *Y2* specify the width and height of the rectangle.

load (Video Overlay)

Command	Arguments
load *device_name*	**[***file_name***]**
	[at *buffer_rectangle***]**
	[notify]
	[wait]

open (Video Overlay)

Syntax: **open *items***
Initializes the video overlay device. The following optional *items* modify **open**:

alias *device_ alias*
Specifies an alternate name for the device element. If specified, it must also be used for subsequent references.

parent *hwnd*
Specifies the window handle of the parent window.

shareable
Initializes the device element as shareable. Subsequent attempts to open it fail unless you specify **shareable** in both the original and subsequent **open** commands. MCI returns an error if it is already open and not shareable.

style *style_type*
Indicates a window style.

style child
Opens a window with a child window style.

style overlapped
Opens a window with an overlapped window style.

style popup
Opens a window with a popup window style.

type *device_ type*
Specifies the device type of the device element. MCI reserves overlay for the video overlay device type. As an alternative to **type**, MCI can use the [mci extension] entries in the SYSTEM.INI file to select the controlling device based on the extension used by the device element.

open (Video Overlay)

Command	Arguments
open *device_name*	**[alias** *device_alias***]**
	[parent *hwnd***]**
	[shareable]
	[style *style_type* **| style child** **| style overlapped** **| style popup]**
	[type *device_type***]**
	[notify]
	[wait]

put (Video Overlay)

Syntax: **put *items***

Defines the source, destination, and frame windows. One of the following *items* modify **put**:

destination

Sets the whole window as the destination window.

destination at *rectangle*

Specifies a rectangle for the area of the window used to display the image. The *rectangle* coordinates are relative to the window origin and are specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner, and the coordinates *X2*, *Y2* specify the width and height of the rectangle. When an area of the display window is specified, and the device supports stretching, the source image is stretched to the destination offset and extent.

frame

Specifies that the whole video buffer is used to capture the video image.

frame at *rectangle*

Specifies a rectangle for the area of the video buffer used to capture the video image. The *rectangle* coordinates are relative to the video buffer origin and are specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner, and the coordinates *X2*, *Y2* specify the width and height of the rectangle. If the device supports stretching, the video image is stretched to the frame extent.

source

Selects the whole video buffer for display in the destination window.

source at *rectangle*

Specifies a rectangle for the video buffer area used to obtain the image for display. The *rectangle* coordinates are relative to the video buffer origin and are specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner, and the coordinates *X2*, *Y2* specify the width and height of the rectangle. When an area of the source image is specified, and the device supports stretching, the source image is stretched to the destination offset and extent.

video

Selects the whole input video source for display in the destination window.

video at *rectangle*

Specifies a rectangle for the input video source area used to obtain the image for display. The *rectangle* coordinates are relative to the video origin and are specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner, and the coordinates *X2*, *Y2* specify the width and height of the rectangle. When an area of the source image is specified, and the device supports stretching, the source image is stretched to the destination offset and extent.

put (Video Overlay)

Command	Arguments
put *device_name*	**{destination	at** *destination_rectangle* **| frame** **| frame at** *frame_rectangle* **| source** **| source at** *source_rectangle* **| video** **| video at** *video_rectangle***}** **[notify]** **[wait]**

save (Video Overlay)

Syntax: **save *item***

Saves the contents of the video buffer in a device specific format. The following optional *item* modifies **save**:

filename

Specifies the file and pathname used to save the data.

at *rectangle*

Specifies a rectangle relative to the video buffer origin. The *rectangle* is specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner and the coordinates *X2*, *Y2* specify the width and height of the rectangle.

save (Video Overlay)

Command	Arguments
save *device_name*	**[***file_name***]**
	[at *buffer_rectangle***]**
	[notify]
	[wait]

set (Video Overlay)

Syntax: **set *items***
Sets the various control *items*:

audio all off
Video overlay devices do not support this option.

audio all on
Video overlay devices do not support this option.

audio left off
Video overlay devices do not support this option.

audio left on
Video overlay devices do not support this option.

audio right off
Video overlay devices do not support this option.

audio right on
Video overlay devices do not support this option.

time format milliseconds
Video overlay devices do not support this option.

video off
Disables video output.

video on
Enables video output.

set (Video Overlay)

Command	Arguments
set *device_name*	**[audio all off** **| audio all on** **| audio left off** **| audio left on** **| audio right off** **| audio right on** **| video off** **| video on]** **[time format milliseconds | time format ms]** **[notify]** **[wait]**

status (Video Overlay)

Syntax: **status *item***

Obtains status information for the device. One of the following *items* modifies **status**:

media present

Returns true.

mode

Returns **not ready**, **recording**, or **stopped** for the current mode.

ready

Returns **true** if the video overlay device is ready.

window handle

Returns the handle of the window used for the video overlay display in the low word of the return value.

status (Video Overlay)

Command	Arguments
status *device_name*	**{current track** **| length** **| length track** *track_number* **| media present** **| mode** **| number of tracks** **| position** **| position track** *track_number* **| ready** **| start position** **| time format** **| window handle}** **[notify]** **[wait]**

unfreeze (Video Overlay)

Syntax: **unfreeze *item***

Enables the frame buffer to acquire video data. This is supported only if **capability can freeze** returns **true**. The following optional *item* modifies **unfreeze**:

at *rectangle*

Specifies a rectangle relative to the video buffer origin. The *rectangle* is specified as *X1 Y1 X2 Y2*. The coordinates *X1*, *Y1* specify the top, left corner and the coordinates *X2*, *Y2* specify the width and height of the rectangle.

unfreeze (Video Overlay)

Command	Arguments
unfreeze *device_name*	**[at** *freeze_rectangle***]**
	[notify]
	[wait]

where (Video Overlay)

Syntax: **where**

Obtains the rectangle specifying the source, destination, or frame area. One of the following *items* modifies **where**:

destination
Requests the offset and extent of the destination rectangle.

frame
Requests the offset and extent of the frame buffer rectangle.

source
Requests the offset and extent of the source rectangle.

video
Requests the offset and extent of the video rectangle.

where (Video Overlay)

Command	Arguments
where *device_name*	**{destination | frame | source | video}**
	[notify]
	[wait]

window (Video Overlay)

Syntax: **window *item***

Tells the video overlay device to use a given window for display instead of the default window created by the driver. By default, video overlay devices should create a window when opened but should not display it until the device receives the **play** command. Applications providing window handles should manage the display issues that result when the window is sized or when the window handle is switched during play. Since the **status** command can obtain the handle to the current display window, you can use the standard window functions instead. The following *items* modify **window**:

handle *window_handle*
Specifies the handle of the destination window used as an alternate to the default window.

handle default
Specifies that the video overlay device should create and manage its own window. This flag can be used to set the display back to the driver's default window.

state hide
Hides the current display window.

state iconic
Displays the window as an icon.

state maximized
Maximizes the current display window.

state minimize
Minimizes the specified window and activates the top-level window in the window-manager's list.

state minimized
Minimizes the current display window.

state no action
Displays the display window in its current state. The window currently active remains active.

state no activate
Displays the display window in its most recent size and state. The window that is currently active remains active.

state normal
Displays the current display window as it was created.

state show
Shows the current display window.

text *caption*
Specifies the *caption* for the display window.

window (Video Overlay)

Command	Arguments
window *device_name*	**[handle** *window_handle* **| handle default]** **[state hide** **| state iconic** **| state maximized** **| state minimize** **| state minimized** **| state no action** **| state no activate** **| state normal** **| state show]** **[text** *caption_text***]** **[notify]** **[wait]**

capability (Waveform Audio)

Syntax: **capability *item***

Requests information about the capabilities of the waveform audio driver. One of the following *items* modify **capability**:

can eject
Returns **false**. Waveform audio devices have no media to eject.

can play
Returns **true** if the device can play. The waveform audio device returns true if an output device is available.

can record
Returns **true** if the device can record.

can save
Returns **true** if the waveform audio device can save data.

compound device
Returns **true**; waveform audio devices are compound devices.

device type
Returns waveaudio.

has audio
Returns true.

has video
Returns **false**. Waveform audio devices do not support video.

inputs
Returns the total number of input devices.

outputs
Returns the total number of output devices.

uses files
Returns **true**. Waveform audio devices use files for operation.

capability (Waveform Audio)

Command	Arguments
capability *device_name*	**{can eject** **| can play** **| can record** **| can save** **| compound device** **| device type** **| has audio** **| has video** **| inputs** **| outputs** **| uses files}** **[notify]** **[wait]**

close (Waveform Audio)

Syntax: **close**

Closes the device element and any resources associated with it.

close (Waveform Audio)

Command	Arguments
close *device_name*	**[notify]**
	[wait]

cue (Waveform Audio)

Syntax: **cue *item***

Prepares for playing or recording. The **cue** command does not have to be issued prior to playing or recording. However, depending on the device, it might reduce the delay associated with the **play** or **record** command. This command fails if playing or recording is in progress. The *item* is one of the following:

input
Prepares for recording.

output
Prepares for playing. This is the default.

cue (Waveform Audio)

Command	Arguments
cue *device_name*	**[input | output]**
	[notify]
	[wait]

delete (Waveform Audio)

Syntax: **delete *items***

Deletes a data segment from the MCI element. The following optional *items* modify **delete**:

from *position*

Specifies the position to start deleting data. If **from** is omitted, deletion starts at the current position.

to *position*

Specifies the position to stop deleting data. If **to** is omitted, deletion stops at the end of the file or waveform.

delete (Waveform Audio)

Command	Arguments
delete *device_name*	**[from** *position***]**
	[to *position***]**
	[notify]
	[wait]

info (Waveform Audio)

Syntax: **info *item***

Fills a user-supplied buffer with information. One of the following *items* modifies **info**:

input

Returns the description of the current waveform audio input device. Returns **none** if an input device is not set. The MCIWAVE driver returns **Wave Audio Input and Output Device.**

file

Returns the current filename.

output

Returns the description of the current waveform audio output device. Returns **none** if an output device is not set. The MCIWAVE driver returns **Wave Audio Input and Output Device.**

product

Returns the description of the current waveform audio output device. The MCIWAVE driver returns **Wave Audio Input and Output Device.**

info (Waveform Audio)

Command	Arguments
info *device_name*	**[file** **| input** **| output** **| product]** **[notify]** **[wait]**

open (Waveform Audio)

Syntax: **open *items***
Initializes the device. The following *items* are optional:

alias *device_ alias*
Specifies an alternate name for the given device. If specified, it must also be used the alias for references.

buffer *buffer_ size*
Sets the size in seconds of the buffer used by the waveform audio device. The default size of the buffer is set when the waveform audio device is installed or configured. Typically the buffer size is set to 4 seconds.

shareable
Initializes the device element as shareable. Subsequent attempts to open it fail unless you specify **shareable** in both the original and subsequent **open** commands. MCI returns an error if it is already open and not shareable. The MCIWAVE device does not support shared files.

type *device_ type*
Specifies the device type of a device element. MCI reserves waveaudio for the waveform audio device type. As an alternative to **type**, MCI can use the [mci extension] entries in the SYSTEM.INI file to select the controlling device based on the extension used by the device element.

open (Waveform Audio)

Command	Arguments
open *device_name*	**[alias** *device_alias***]**
	[buffer *buffer_size***]**
	[shareable]
	[type *device_type***]**
	[notify]
	[wait]

pause (Waveform Audio)

Syntax: **pause**

Pauses playing or recording.

pause (Waveform Audio)

Command	Arguments
pause *device_name*	**[notify]**
	[wait]

play (Waveform Audio)

Syntax: **play *items***
Starts playing audio. The following optional *items* modify **play**:

from *position*
Specifies the position to start playing. If **from** is omitted, play starts at the current position.

to *position*
Specifies the position to stop playing. If **to** is omitted, play stops at the end of the file or waveform.

play (Waveform Audio)

Command	Arguments
play *device_name*	**[from** *position***]**
	[to *position***]**
	[notify]
	[wait]

record (Waveform Audio)

Syntax: **record *items***

Starts recording audio. All data recorded after a file is opened is discarded if the file is closed without saving it. The following optional *items* modify **record**:

insert

Specifies that new data is added to the device element.

from *position*

Specifies the position to start recording. If **from** is omitted, the device starts recording at the current position.

to *position*

Specifies the position to stop recording. If **to** is omitted, the device records until a **stop** or **pause** command is received.

overwrite

Specifies that new data will replace data in the device element. The MCIWAVE driver does not support this option.

record (Waveform Audio)

Command	Arguments
record *device_name*	**[from** *position***]**
	[to *position***]**
	[insert | overwrite]
	[notify]
	[wait]

resume (Waveform Audio)

Syntax: **resume**

Resumes playing or recording of a paused device.

resume (Waveform Audio)

Command	Arguments
resume *device_name*	**[notify]**
	[wait]

save (Waveform Audio)

Syntax: **save *item***

Saves the MCI element in its current format. The following *item* modifies **save**:

filename

Specifies the file and pathname used to save data.

save (Waveform Audio)

Command	Arguments
save *device_name*	**[***file_name***]**
	[notify]
	[wait]

seek (Waveform Audio)

Syntax: **seek *item***

Moves to the specified location in the file. One of the following items modify **seek**:

to *position*
Specifies the stop position.

to start
Specifies to seek to the start of the first sample.

to end
Specifies to seek to the end of the last sample.

seek (Waveform Audio)

Command	Arguments
seek *device_name*	**[to** *position* **| to start | to end]**
	[notify]
	[wait]

set (Waveform Audio)

Syntax: **set *items***
Sets the various control *items*:

alignment *integer*
Sets the alignment of data blocks relative to the start of waveform data passed to the waveform audio device. The file is saved in this format.

any input
Use any input that supports the current format when recording. This is the default.

any output
Use any output that supports the current format when playing. This is the default.

audio all off
Disables audio output. The MCIWAVE driver does not support this option.

audio all on
Enables audio output. The MCIWAVE driver does not support this option.

audio left off
Disables output to the left audio channel. The MCIWAVE driver does not support this option.

audio left on
Enables output to the left audio channel. The MCIWAVE driver does not support this option.

audio right off
Disables output to the right audio channel. The MCIWAVE driver does not support this option.

audio right on
Enables output to the right audio channel. The MCIWAVE driver does not support this option.

bitspersample *bit_count*
Sets the number of bits per sample played or recorded. The file is saved in this format.

bytespersec *byte_rate*
Sets the average number of bytes per second played or recorded. The file is saved in this format.

channels *channel_count*
Sets the channels for playing and recording. The file is saved in this format.

format tag *tag*
Sets the format type for playing and recording. The file is saved in this format.

format tag pcm
Sets the format type to PCM for playing and recording. The file is saved in this format.

input *integer*
Sets the audio channel used as the input.

output *integer*
Sets the audio channel used as the output.

samplespersec *integer*

Sets the sample rate for playing and recording. The file is saved in this format.

time format bytes
Sets the time format to bytes. All position information is specified as bytes following this command.

time format milliseconds
Sets the time format to milliseconds. All position information is specified as milliseconds following this command. This is the default. You can abbreviate milliseconds as **ms**.

time format samples
Sets the time format to samples. All position information is specified as samples following this command.

set (Waveform Audio)

Command	Arguments
set *device_name*	**[alignment** *block_alignment***]** **[any input]** **[any output]** **[audio all off** **| audio all on** **| audio left off** **| audio left on** **| audio right off** **| audio right on** **| video off** **| video on]** **[bitspersample** *bit_count***]** **[bytespersec** *byte_rate***]** **[channels** *channel_count***]** **[format tag** *tag* **| format tag pcm]** **[input** *device_number***]** **[output** *device_number***]** **[samplespersec** *sample_rate***]** **[time format milliseconds** **| time format ms** **| time format bytes** **| time format samples]** **[notify]** **[wait]**

status (Waveform Audio)

Syntax: **status *item***

Obtains status information for the device. One of the following *items* modifies **status**:

alignment
Returns the block alignment of data in bytes.

bitspersample
Returns the bits per sample.

bytespersec
Returns the average number of bytes per second played or recorded.

channels
Returns the number of channels set (1 for mono, 2 for stereo).

current track
Returns the index for the current track. The MCIWAVE device returns 1.

format tag
Returns the format tag.

input
Returns the input set. If one is not set, the error returned indicates that any device can be used.

length
Returns the total length of the waveform.

length track *track_number*
Returns the length of the specified track.

level
Returns the current audio sample value.

media present
Returns true.

mode
Returns **not ready**, **paused**, **playing**, **stopped**, **recording**, or **seeking** for the device mode.

number of tracks
Returns the number of tracks. The MCIWAVE device returns 1.

output
Returns the currently set output. If no output is set, the error returned indicates that any device can be used.

position
Returns the current position.

position track *track_number*
Returns the position of the track specified by *track_number*. The MCIWAVE device returns 0.

ready

Returns **true** if the device is ready.

samplespersec
Returns the number of samples per second played or recorded.

start position
Returns the starting position of the media or device element.

time format
Returns the current time format.

status (Waveform Audio)

Command	Arguments
status *device_name*	**{alignment** **| bitspersample** **| bytespersec** **| channels** **| current track** **| format tag** **| input** **| length** **| length track** *track_number* **| level** **| media present** **| mode** **| number of tracks** **| output** **| position** **| position track** *track_number* **| ready** **| samplespersec** **| start position** **| time format}** **[notify]** **[wait]**

stop (Waveform Audio)

Syntax: **stop**

Stops playing or recording.

stop (Waveform Audio)

Command	Arguments
stop *device_name*	**[notify]**
	[wait]

System Commands

The following list summarizes the system commands. MCI supports these commands directly rather than passing them to MCI devices.

Command	Description
break	Sets a break key for an MCI device.
sound	Play sounds from the [sounds] section of the WIN.INI file.
sysinfo	Returns information about MCI devices.

Required Commands

The following list summarizes the required commands. All devices support these commands.

Command	Description
capability	Obtains the capabilities of a device.
close	Closes the device.
info	Obtains textual information from a device.
open	Initializes the device.
status	Obtains status information from the device.

Basic Commands

In addition to the required commands, each device supports a set of commands specific to its device type. Where possible, these type-specific commands are identical between types. When type-specific commands are common to multiple devices, they are considered basic commands. For example, the basic **play** command is identical for videodisc and videotape devices.

Although these commands are optional for a device, if a device supports a command it must respond to the basic options defined for it. The options provide a minimum set of capabilities for most devices. If an option does not apply to device, the device must return "Action not available for this device."

The following list summarizes the basic commands.

Command	Description
load	Recalls data from a disk file.
pause	Stops playing.
play	Starts transmitting output data.
record	Starts recording input data.
resume	Resumes playing or recording on a paused device.
save	Saves data to a disk file.
seek	Seeks forward or backward.
set	Sets the operating state of the device.
status	Obtains status information about the device. (The flags for this command supplement the flags for the command in the required command group.)
stop	Stops playing.

Extended Commands

MCI devices can have additional commands or extend the definition of the required and basic commands. While some extended commands only apply to a specific device driver, most of them apply to all devices of a particular type. For example, the MIDI sequencer command set extends the **set** command to add time formats that are needed by MIDI sequencers. The following table lists the extended commands not listed in the overviews of the basic and required commands.

Command	Command Set	Description
cue	Waveform Audio	Prepares for playing or recording.
delete	Waveform Audio	Deletes a data segment from the MCI element.
escape	Videodisc	Sends custom information to a device.
freeze	Video Overlay	Disables video acquisition to the frame buffer.
put	Animation	Defines the source, destination, and frame windows.
	Video Overlay	
realize	Animation	Tells the device to select and realize its palette into a display context of the displayed window.
spin	Videodisc	Starts the disc spinning or stops the disc from spinning.
step	Animation	Step the play one or more frames forward or reverse.
	Videodisc	
unfreeze	Video Overlay	Enables the frame buffer to acquire video data.
update	Animation	Repaints the current frame into the display context.
where	Animation	Obtains the rectangle specifying the source, destination,
	Video Overlay	or frame area.
window	Animation	Tells the device to use a given window for display
	Video Overlay	instead of the default window.

Animation Commands

The following list summarizes the animation MCI commands.

Command	Description
capability	Obtains the capabilities of a device.
close	Closes the device.
info	Obtains textual information from a device.
open	Initializes the device.
pause	Pauses the device.
play	Starts transmitting output data.
put	Defines the source, destination, and frame windows.
realize	Tells the device to select and realize its palette into a display context of the displayed window.
set	Sets the operating state of the device.
status	Obtains status information from the device.
step	Step the play one or more frames forward or reverse.
stop	Stops playing.
update	Repaints the current frame into the display context.
where	Obtains the rectangle specifying the source or destination area.
window	Tells the device to use a given window for display instead of the default window.

CD Audio Commands

The following list summarizes the CD audio MCI commands.

Command	Description
capability	Obtains the capabilities of a device.
close	Closes the device.
info	Obtains textual information from a device.
open	Initializes the device.
pause	Stops playing.
play	Starts playing the device.
resume	Resumes playing or recording on a paused device.
seek	Seeks forward or backward.
set	Starts the disc spinning or stops the disc from spinning.
status	Obtains status information from the device.
stop	Stops playing.

MIDI Sequencer Commands

The following list summarizes the MIDI sequencer MCI commands.

Command	Description
capability	Obtains the capabilities of a device.
close	Closes the device.
info	Obtains textual information from a device.
open	Initializes the device.
pause	Stops playing.
play	Starts transmitting output data.
record	Starts recording input data.
resume	Resumes playing or recording on a paused device.
save	Saves data to a disk file.
seek	Seeks forward or backward.
set	Starts the disc spinning or stops the disc from spinning.
status	Obtains status information from the device.
stop	Stops playing.

Videodisc Command Overview

The following list summarizes the videodisc MCI commands.

Command	Description
capability	Obtains the capabilities of a device.
close	Closes the device.
escape	Sends custom information to a device.
info	Obtains textual information from a device.
open	Initializes the device.
pause	Stops playing.
play	Starts playing the device.
resume	Resumes playing or recording on a paused device.
seek	Seeks forward or backward.
set	Starts the disc spinning or stops the disc from spinning.
spin	Starts the disc spinning or stops the disc from spinning.
status	Obtains status information from the device.
step	Step the play one or more frames forward or reverse.
stop	Stops playing.

Video Overlay Command Overview

The following list summarizes the video overlay MCI commands.

Command	Description
capability	Obtains the capabilities of a device.
close	Closes the device.
freeze	Disables video acquisition to the frame buffer.
info	Obtains textual information from a device.
load	Recalls data from a disk file.
open	Initializes the device.
put	Defines the source, destination, and frame windows.
save	Saves data to a disk file.
set	Starts the disc spinning or stops the disc from spinning.
status	Obtains status information from the device.
unfreeze	Enables the frame buffer to acquire video data.
where	Obtains the rectangle specifying the source, destination, or frame area.
window	Tells the device to use a given window for display instead of the default window.

Waveform Audio Command Overview

The following list summarizes the waveform audio MCI commands.

Command	Description
capability	Obtains the capabilities of a device.
close	Closes the device.
cue	Prepares for playing or recording.
delete	Deletes a data segment from the MCI element.
info	Obtains textual information from a device.
open	Initializes the device.
pause	Stops playing.
play	Starts transmitting output data.
record	Starts recording input data.
resume	Resumes playing or recording on a paused device.
save	Saves data to a disk file.
seek	Seeks forward or backward.
set	Starts the disc spinning or stops the disc from spinning.
status	Obtains status information from the device.
stop	Stops playing.