Microsoft®

Multimedia

Standards Update

Motion Picture Expert Group Command Set for the Media Control Interface

September 29, 1994
Revision: 1.0

Information in this document is subject to change without notice and does not represent a commitment on the part of Microsoft Corporation. The software described in this document is furnished under license agreement or nondisclosure agreement. The software may be used or copied only in the accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or nondisclosure agreement.

This standards update is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESSED OR IMPLIED IN THIS DOCUMENT.

Table of Contents

Overview

Preface

In late 1993 and early 1994, several hardware and software companies approached Microsoft expressing interest in creating an industry standard for the Motion Picture Expert Group (MPEG) Standard on PCs. Microsoft had already identified MPEG as promising technology for a variety of applications and was debating internally how to implement it under Microsoft® Windows™. Commercial MPEG products for PCs were starting to appear, and people at Microsoft and in the software industry wanted a standard to encourage the adoption of MPEG on the PC.
As a result of all this, Jim Anderson of Jazz Multimedia created the Open PC MPEG Consortium. Jim held meetings, organized working sessions, and recruited participants. Now, more than 50 companies participate in the consortium, including Microsoft. This document is a result of this consortium.

Thanks

Microsoft thanks Jim Anderson for organizing the consortium, chairing meetings, and faxing information. Microsoft also thanks Partha Srivinasan and Prem Nath of Mediamatics for writing this command set, educating the consortium participants, working with Microsoft on the technical issues, and reaching consensus from a diverse group of people.

Introduction

This document describes how to play MPEG files using Windows. Microsoft hopes software developers use the MPEG command set to develop multimedia applications. Additionally, Microsoft hopes PC hardware manufacturers provide hardware and drivers to support this command set. As a result, users can be sure their applications will work on the MPEG add-on software or hardware they have.

This document describes only the linear playback of MPEG data. Because there might be additional technical questions surrounding MPEG on PCs, Microsoft will address other requirements in future standards.

Where to Look for Information

This document is based on several other standards. This MPEG command set is largely a subset of the "Digital Video Command Set for the Media Control Interface" (Microsoft Part No. 098-37538). You can find an electronic version of this document on the Microsoft Developers' Network CD under "Specs and Strategy."

For more information about the Media Control Interface (MCI), see the Microsoft Windows 3.1 Software Development Kit.

Versions of This Document

Version	Date	Comments
1.00	Sep. 10, 16, 20, 29, 1994	Changes from .99 version: added seek frameref frame, updatevideodc at rect. Where Video becomes where source. Info audio and info video items become status audio and video items. Removed info audio samplerate (use status audio samplespersec). Rectangle definition. Format. HRD file mods

MPEG Command Set for MCI

Definitions

The MCI driver supports a subset of the MCI Digital Video (DV) command set with some minor additions. All the commands support the **notify**, **test** and **wait** modifiers. The proposed name for the device is **MpegVideo**. Use the following modifier with the **open** command:

```
open compton.mpg type MpegVideo
```

Use the following entires the SYSTEM.INI file and WIN.INI file

SYSTEM.INI Entries

```
[mci]
..
AVIVideo=mciavi.drv
..
MpegVideo=mymci.drv
..
```

WIN.INI Entries

```
[mci extensions]
mpg = MpegVideo
```

The driver will support the **digitalvideo** device type.

For information about the relevant keywords and their meanings, see "Digital Video Command Set for the Media Control Interface" (Microsoft Part No. 098-37538).

This document is an appendix to the “Microsoft Multimedia Standards Update.” You can find an electronic version of this document on the Microsoft Developers’ Network CD under "Specs and Strategy."

The MPEG command set controls system bitstreams conforming to the standard - ISO/IEC JTC 1 11172 “Information Technology - Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbits/s.”

MCI Mpeg Video Command Set

The following list of supported commands are presented in the same order as in the DV specifications. New items are indicated in bold. New commands are indicated in underline.

NOTE The drivers must recognize and reply to all capability queries. All mandatory reply values are indicated in the following list. Any values not specified are optional. If you reply TRUE to any of the optional capabilities, such as "uses palettes," make sure you provide any additional messages required to support such a capability. You can reply FALSE to any of the unspecified capabilities, such as "can freeze." Most devices will support the "can stretch" capability. All commands must support the notify, wait and test flags.

capability *items*		MCI_GETDEVCAPS	
	can eject		0
	can freeze		
	can lock		
	can play		1
	can record		0
	can reverse		
	can save		
	can stretch [input]		
	can test		1
	compound device		1
	device type		**digitalvideo**
	has audio		1
	has still		
	has video		1
	maximum play rate		
	minimum play rate		
	uses files		1
	uses palettes		
	windows		
capture *items*		MCI_CAPTURE	
	as *pathname*		
close		MCI_CLOSE	
configure		MCI_CONFIGURE	
copy *items*		MCI_COPY	
createvideodc *items*		MCI_CREATEDC	
	at *rect*		
cue *items*		MCI_CUE	
	to *position*		
	noshow		
freeze		MCI_FREEZE	
info *items*		MCI_INFO	
	file		
	product		
	version		

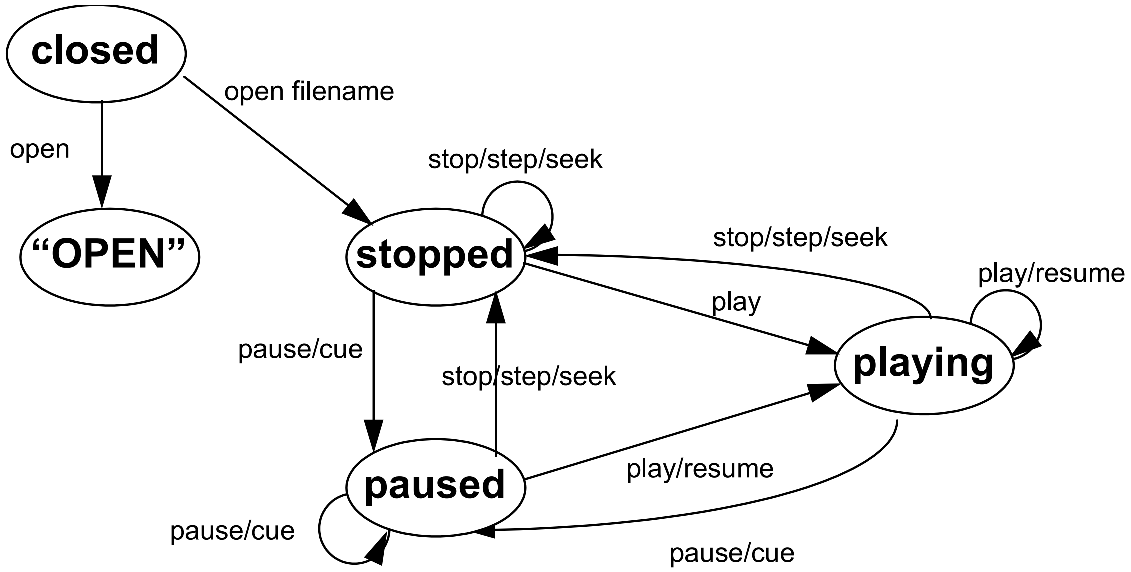
	window text	
open *items*		MCI_OPEN
	alias *alias*	
	parent *hwnd*	
	shareable	
	style *StyleValue*	
	type **MpegVideo**	
pause		MCI_PAUSE
play *items*		MCI_PLAY
	from *position*	
	to *position*	
	repeat	
put *items*		MCI_PUT
	at *rect*	
	destination	
	window	
	window client	
releasevideodc hdc *hDC*		MCI_RELEASEDC
resume		MCI_RESUME
seek *items*		MCI_SEEK
	offset *bytes*	
	to *position*	
	to end	
	to start	
	frameref *frame*	
set *items*		MCI_SET
	// no setting allowed for speed factor - it depends on clip	
	time format *format*	
		frames and milliseconds
setaudio *items*		MCI_SETAUDIO
	off	
	on	
	stream to *number*	
setvideo *items*		MCI_SETVIDEO
	off	
	on	
	stream to *number*	
signal *items*		MCI_SIGNAL
	at *position*	
	cancel	
	every *interval*	
	return position	

uservalue *id*

status *items*		MCI_STATUS	
	audio		
	audio bitrate		
	audio samplespersec		should be 48000, 44100 or 32000, or sampes per sec.
	audio stream		
	audio streams		
	length		
	mode		
	nominal frame rate		
	position		
	ready		
	speed		
	start position		
	time format		
	video		
	video brush		
	video stream		
	video streams		
	video bitrate		
	video maxbitrate		
	window handle		
	window visible		
	window maximized		
	window minimized		
step *items*		MCI_STEP	
	by *frames*		
	key		
stop		MCI_STOP	
update *items*		MCI_UPDATE	
	at *rect*		
	hdc *hdc*		
	paint		
updatevideodc *items*		MCI_UPDATEDC	
	hdc *hDC*		
	at *rect*		
unfreeze			
where *items*		MCI_WHERE	
	destination		
	destination max		
	source		
	source max		
	source min		
	window		

	window max	
window *items*		MCI_WINDOW
	handle *hwnd*	
	handle default	
	state *ShowValue*	
	text *caption*	

State Transition Model for the Driver



Notes:
1. In the "open" state, only query commands are supported.
2. The MCI command "close" always returns to the closed state. Arrows have been omitted for clarity.

Notes on the New Calls

status *items* MCI_INFO

audio bitrate

Bitrate of the audio stream that is the current stream. e.g 192000

audio streams

Number of audio stream in the system stream. Zero if system stream contains no audio stream.

video streams

Number of video streams in the system stream. Zero if system stream contains no video stream.

video bitrate

Bitrate of the video stream that is the current stream.

video maxbitrate

Maximum video bitrate accepted by the device (for example, 1295000 bits/sec).

where *items* MCI_WHERE

source min

Minimum resolution video that can be displayed (for example, 160x120 pixels).

Optional Commands

To determine if one of these calls are supported, send the command with the test flag.

step *items* MCI_STEP

key

The added item "key", was desired for the **step** command. This would be a modifier to the basic step, such that the step from the current position is measured in "key frames." For example, **step** “key” advances from the current position to the next key frame. In the case of MPEG, key frames are actually I frames. For example, **step** “by 10 key” advances from the current position 10 I frames.

seek *items* MCI_SEEK

offset *bytes*
frameref *frame*

The added item ”offset” in the seek command would seek to a position offset from the beginning of the file, an absolute byte position at which to begin decoding. The application must ensure that this is a position in the MPEG bitstream from which the device can start decoding. This will be file specific. The added item “frameref” is the actual frame number represented by the byte position, allowing the driver can know what frame it is currently at. For example, if seeking frame 120, but the previous I frame was 110 at byte position 100,000, the following command is issued:

“seek to 120 offset 100000 frameref 110”

The following group of commands is also optional. However, if you support one, then you must support them all. Failing on a **createvideodc** call means that none of these calls are available.

status *items* MCI_STATUS

video brush

Returns a windows brush (HBRUSH) to be used by the application to indicate areas where video should be displayed. Areas without the HBRUSH color contain the graphics data that was painted into them. HBRUSH is used instead of **status** "video key color" or **status** "video key index" so the driver can use the color privately, isolating the application from it.

This HBRUSH would be used to paint areas of the VideoDC. The VideoDC acts as a combined graphics rendering surface and control surface.

createvideodc *items* ***MCI_CREATEDC***

at *rect*

Creates a device context (DC), on which the application can draw the sprite. The application uses the HBRUSH returned from the **status** "video brush" command to paint areas where video will be displayed. If the items "at *rect*" is specified, the VideoDC is created of the size *rect*.

A rectangle consists of four integer values denoting the coordinates of the upper-left pixel and the width and height of the rectangle. If *rect* is not specified, the VideoDC has the same dimensions as the video window. For example, a rectangle with dimensions 10, 20, 150, 100, creates a video DC whose upper left corner is at pixel 10, line 20 and whose lower right corner is at pixel 160, line 120 of the video window. . The origin of *rect* is at the upper left corner of the video DCr.
the origin

updatevideodc *items* ***MCI_UPDATEDC***

hdc *hDC*

at *rec*

After the application draws the sprite/text into the VideoDC, this function updates the contents of the VideoDC in the video window. The areas of the VideoDC that contain the HBRUSH color display video, and the rest displays the contents of the VideoDC.

releasevideodc hdc *hDC* ***MCI_RELEASEDC***

Releases the VideoDC obtained by the application using CreateVideoDC.