New Technical Notes
Macintosh

®

Developer Support

CD-ROM Driver Calls

Devices

Written by: Chris Brown, Neil Day, and Brian Bechtel May 1993

This Technical Note discusses the public interface to the Apple CD-ROM driver, which currently supports the AppleCD SC, AppleCD SC Plus/AppleCD 150, and the AppleCD 300. This information supercedes the "Macintosh CD-ROM Device Driver" chapter of AppleCD SC Developer's guide available through APDA. If you're writing special purpose application software that needs to access the audio or data portions of a CD-ROM directly, this note will be of interest to you.

Topics

- Status & Control calls for the Apple CD-ROM Driver.
- Differences between CD SC, CD SC Plus/CD 150, and CD 300.

This Technical Note describes version 4.0.x of the Apple CD-ROM driver. Previous versions of the Apple CD-ROM driver are documented in the *AppleCD SC Developers Guide* from APDA. Version 4.0 of the Apple CD-ROM extension was the first version to support dual-speed operation and multiple-session Photo CDs on the AppleCD 300. Version 4.0 also supports single-session Photo CDs on all other Apple CD-ROM drives.

Version 4.0 had problems with certain CD-ROMs that used encryption technology to restrict access to files. When the customer was given a decryption key to gain access to a file, the file would not always appear in the Finder. Version 4.0.1 was produced to fix this problem. This version was only tested on the Macintosh IIvx and Performa 600. Version 4.0.2 was tested on all Macintosh computers. A few changes were made to ensure compatibility across the Macintosh product line. Version 4.0.2 corrected problems when sending audio commands using block addressing only. Audio calls using minute-second-frame addressing continued to work correctly.

Audio compact discs and CD-ROM discs conform to two standards, called the Red Book and the Yellow Book. The Red book specifies audio standards; the Yellow book specifies additional standards for CD-ROM. “Red Book” is the common name of the “Compact Disc Digital Audio Standard” standard, CEI IEC 908. When a disc conforms to the red book standard, it will usually have “digital audio” printed below the “disc” logo. Most music CDs conform to this standard. “Yellow Book” is the standard for CD-ROM, ISO 10149. When a disc conforms to the yellow book, it will usually say “data storage” beneath the “disc” logo.

There are several other standards usually associated with CD-ROM technology. The "Green Book" standard defines CD-I (Compact Disc-Interactive); "Orange Book" defines write-once compact discs; and "Blue Book" defines LaserDisc.

You can get the Red Book and Yellow Book from

ANSI
Attn: Sales
1430 Broadway
New York, NY 10018
(212) 642-4900

Red Book:	CEI IEC 908
Yellow Book:	ISO 10149:1989

You can get the Green Book from

American CD-I Association
11111 Santa Monica, Suite 750
Los Angeles, CA 90025
(213) 444-6619

The other standards are available only to Sony/Philips licensees. Contact Sony or Philips for licensing details.

Beginning with version 4.0 of the Apple CD-ROM software, several new Gestalt selectors are installed by Foreign File Access and various file system translators. Each selector is a version number, as defined in Inside Macintosh volume VI page VI-3-34. The list includes:

Apple Photo Access	kpcd
Foreign File Access	ufox
High Sierra File Access / ISO 9660 File Access	hscd (If High Sierra File Access and/or ISO 9660 File Access is installed.)
Audio CD Access	aucd

To call the Macintosh driver, you need to understand the possible addressing formats for audio blocks on a CD. The three addressing formats are:

1) logical block
2) absolute minute, second and frame
3) track number.

An audio CD contains up to 74 minutes of sound (By decreasing the inter-track gap and other tricks, you can put more than 74 minutes of sound on a CD. Doing this is only marginally within the standard. Such CDs may not be playable on all drives.). These minutes are divided into 60 seconds. Each second contains 75 frames. There is a two second lead-in area at the beginning of the disc which contains table of contents information about the tracks on that CD. You can have a maximum of 99 tracks on a CD.

You can address sound down to the frame level; that is, down to the 1/75th of a second. You can address sound by specifying an absolute block number (e.g., start playing at absolute block 1,234,567 from the start of the disc), or by absolute minute, second, and frame number (e.g., start playing at minute 42, second 30, frame 15 on the disc) or by logical track number (e.g., start playing at track 2.)

The driver requires that the logical track number or absolute minute-second-frame numbers be specified in BCD (e.g., the decimal number 12 is stored as hex 0x0012, not as 0x000C.)

Optical Positioning Type

Many calls require an optical pick-up positioning type. This is used to specify what kind of address is being used in other parameters of a given call. There are three different optical pick-up positioning types:

Type	Description
0x0000	32-bit Logical Block Address
0x0001	An AMIN-ASEC-AFRAME descriptor that gives the running time from the beginning of the disc. (in BCD)
0x0002	A track number (in BCD).

Track Control Field

Many calls return a control field. This field describes the format of the current track, and has the following values:

bits 3210	function
00x0	2 audio channels without pre-emphasis
00x1	2 audio channels with pre-emphasis
10x0	4 audio channels without pre-emphasis
10x1	4 audio channels with pre-emphasis
01x0	data track
01x1	reserved
11xx	reserved
xx0x	digital copy prohibited
xx1x	digital copy permitted

Audio Play Mode

Many calls require a play mode. This field describes how to play the audio track. This field has the following values:

bits 3210	effect
0000	muting on (no audio)
0001	right channel through right channel only
0010	left channel through right channel only
0011	left and right channel through right channel only
0100	right channel through left channel only
0101	right channel through left and right channel
0110	right channel through left channel, left channel through right channel
0111	right channel through left channel, left and right channel through right channel
1000	left channel through left channel only
1001	left channel through left channel, right channel through right channel (stereo)
1010	left channel through left and right channel
1011	left channel through left channel, left and right channel through right channel
1100	left and right channel through left channel
1101	left and right channel through left channel, right channel through right channel
1110	left and right channel through left channel, left channel through right channel
1111	left and right channel through left channel, left and right channel through right channel (monaural)

In the absolute minute-second-frame address, the ranges are as follows:

Minutes	00 to 99 (effectively, from 00 to 75)
Seconds	00 to 59
Frames	00 to 74

Making a High Level Driver Call

The Apple CD-ROM driver does not conform to the normal design of a driver. In particular, the Apple CD-ROM driver uses Status calls to change control settings of the device, and uses control calls to get status information from the device. The high-level `Control` and `Status` calls weren't designed with this in mind. The glue for `Control` does not fill in the `csParamPtr` field with the results of any changes due to a `Control` call, and the glue for `Status` does not use the `csParam` field passed in as part of the call it creates on the stack. Do not use high level `Control` and `Status` calls to access the Apple CD-ROM driver.

Making a Low Level Driver Call

The low-level calls require that you pass in two parameters; a pointer to a parameter block and a flag indicating synchronous or asynchronous completion. For now, the driver always is synchronous, so always pass false as the second parameter. You must fill in the parameter block yourself. You may find it easier to define your own custom parameter block variants for the various calls. Some example MPW C code to get audio status information is as follows:

```
#include <Devices.h>

short  drvRefNum;          /* set up somewhere else */

/* this is a custom version of the CntrlParam parameter
** block defined on page II-181 and II-183 or in
** Devices.h
*/
typedef struct {
       QElemPtr     qLink;
       int          qType;
       int          ioTrap;
       Ptr          ioCmdAddr;
       ProcPtr      ioCompletion;
       OsErr        ioResult;
       StringPtr    ioNamePtr;
       int          ioVRefNum;
       int          ioRefNum;
       int          csCode;
/* Everything below this replaces: short csParam[11] */
       unsigned char status;
       unsigned char play;
       unsigned char control;
       unsigned char minute;
       unsigned char second;
       unsigned char frame;
       char   unused[16];         /* for the rest */
} CDCntrlParam;

/*
** GetAudioStuff
**     requires one input, the driver reference number that we
**     got by calling PBOpen().
**     fills in six parameters with appropriate information
**     from the driver call.
**     We're assuming that only one CD player is attached.
**     To generalize this code, fill in the ioVRefNum dynamically.
*/
OSErr
GetAudioStuff(status, play, control, minute, second, frame)
unsigned char *status, *play, *control, *minute, *second, *frame;
{
       OSErr  result;
       CDCntrlParam myPB;

       myPB.ioCompletion = 0;
       myPB.ioVRefNum = 1;        /* we're assuming only 1 volume */
       myPB.ioRefNum = gDrvRefNum;      /* determined elsewhere */
       myPB.csCode = 107;

       result = PBControl(&myPB, false);
       if (result == noErr)
       {
              *status = myPB.status;
              *play = myPB.play;
              *control = myPB.control;
              *minute = myPB.minute;
              *second = myPB.second;
              *frame = myPB.frame;
       }
       return(result);
}
```

Driver Summary

Name

.AppleCD

Supported Control Calls

csCode	Function Name	Description
Standard Driver Calls		
1	KillIO	Interrupt asynchronous driver operations.
5	VerifyTheDisc	Issue the SCSI VERIFY command
6	FormatTheDisc	Issue the SCSI FORMAT UNIT command
7	EjectTheDisc	Issue the SCSI EJECT DISC command
21	GetDriveIcon	Return 'ICN#' information for the drive hardware
22	GetMediaIcon	Return 'ICN#' information for the drive media
23	DriveInfo	Return drive characteristics information
65	accRun	Driver-specific needTime code
Special CD-ROM Control Calls		
76	ModifyPostEvent	Enable/disable PostEvent()s for non-HFS discs.
79	SetBlockSize	Modify block size of device
80	UserEject	Enable/disable the button on the front of the CD-ROM device
81	SetPollFreq	Modify period of the VBL/dNeedTime task execution
Audio track Control Calls		
100	ReadTOC	Return a disc's Table Of Contents (TOC) information
101	ReadQ	Return Q Subcode information
102	ReadHeader	Return 4-byte Header information for a specified block
103	AudioTrackSearch	Search disc for a specified track
104	AudioPlay	Cause drive to play a given range of audio tracks
105	AudioPause	Cause drive to enter/exit the Hold Track State
106	AudioStop	Specify an address to cause the CD-ROM to stop
107	AudioStatus	Request the play status information from a drive

108	AudioScan	Cause drive to fast-forward/reverse
109	AudioControl	Set volume level for drive (Not available on CD SC)
110	ReadMCN	Return Media Catalog Number (Not available on CD SC)
111	ReadISRC	Return International Standard Recording Code (Not available on CD SC)
112	ReadAudioVolume	Return Audio Volume Control Data (Not available on CD SC)
113	GetSpindleSpeed	Return the current spindle speed
114	SetSpindleSpeed	Set the spindle speed
115	ReadAudio	Return digital audio data (CD 300 only)
116	ReadAllSubcodes	Return subcodes while playing audio (CD 300 only)

Supported Status Calls

csCode	Function Name	Description
8	DriveStatus	Provides drive/disc information about a specified SCSI ID.
95	Get2KOffset	Returns last Prime call's offset from start of 2K physical block.
96	GetDriveType	Returns the type of CD drive at a given SCSI ID.
97	WhoIsThere	Returns a bitmap of SCSI IDs the CD-ROM driver controls.
98	GetBlockSize	Returns the block size of the disc at a given SCSI ID.

Control Call Descriptions

KillIO (1)

The purpose of this call is to allow the interruption of asynchronous device driver operations. All calls to this driver are assumed to be synchronous so KillIO just returns noErr.

Input Parameters:

csCode	1
csParam[]	Don't care.

Status Return Codes:

noErr	This is the only expected status return code.

VerifyTheDisc (5)

The purpose of this call is to ensure the validity of the data on the disc. CD-ROMs cannot be changed (they are read-only, as the name implies). When this Control call is made, as long as a disc is inserted in the drive, the CD-ROM driver will return noErr. If no disc is inserted in the drive, the CD-ROM driver will return offLinErr.

Input Parameters:

csCode	5

csParam[]	Don’t care.

Status Return Codes:

noErr	The entire disc was successfully verified with no errors detected.
offLinErr	No disc inserted at the specified SCSI ID.

FormatTheDisc (6)

The purpose of this call is to format and initialize the CD-ROM disc for use in the Macintosh. However, a CD-ROM cannot be formatted after it has been pressed, so this command always returns a controlErr status code.

Input Parameters:

csCode	6
csParam[]	Don’t care.

Status Return Codes:

controlErr	This control call is not valid for this device.

EjectTheDisc (7)

The purpose of this call is to cause a disc eject at a specified SCSI ID. The specified SCSI device is sent a SCSI Eject Disc command to eject the disc. But first, a SCSI Prevent/Allow Medium Removal command is issued to reactivate the external Eject button on the front panel of the CD-ROM drive. If the Prevent/Allow Medium Removal command is not executed to re-allow the manual eject button, the Eject Disc command will fail with a “Check Condition” status code. The dNeedTime flag in the specified SCSI ID’s DCE is set so that new disc insertions will be periodically checked for (see the accRun control call for more details). Finally, the drive queue entry associated with this drive is marked as being Off-Line and Ejected.

Input Parameters:

csCode	7
csParam[]	Don’t care.

Status Return Codes:

noErr	The entire disc was successfully ejected.
offLinErr	No disc inserted at the specified SCSI ID.

GetDriveIcon (21)

The purpose of this call is to return ‘ICN#’-style data that is typically displayed either during the disc formatting process or by the Startup Disk Control Panel module. The drive icon is the same as the media icon in this driver. There is no true drive icon because there is no way of determining if an internal or external drive should represented. The icons always look like this:

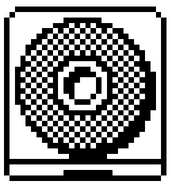


Figure 1: Icon returned by GetDriveIcon and GetMediaIcon Control calls

Input Parameters:

csCode	21
csParam[0-10]	Don’t care.

Output Produced:

csParam[0-1]	Address of ICN# and name string

Status Return Codes:

noErr	This is the only possible status return code.

GetMediaIcon (22)

The purpose of this call is to return ‘ICN#’ -style data that is displayed whenever the disc media is shown. This icon can be seen on the desktop or when the “Get Info” (⌘-I) command from the Finder is executed under System 7. See figure 1 for a picture of the icon returned.

Input Parameters:

csCode	22
csParam[0-10]	Don’t care.

Output Produced:

csParam[0-1]	Address of ICN# and name string

Status Return Codes:

noErr	This is the only possible status return code.

DriveInfo (23)

The purpose of this call is to return information describing drive characteristics for the specified SCSI ID. The values to be returned are defined in Inside Macintosh, volume V, pages 470-471. The value returned by the Apple CD-ROM driver is $0000 $0B01. The low order byte indicates unspecified drive type. The high order byte of the low word indicates external drive (this is set even if the drive is actually mounted internally), SCSI drive, removable drive, and secondary drive.

Input Parameters:

csCode	23
csParam[0-10]	Don’t care.

Output Produced:

csParam[0] (word)	$0000
csParam[1] (word)	$0B01

Status Return Codes:

noErr	This is the only possible status return code.

accRun (65)]

There is no need for an application to issue this call. This call is made on a regular basis by the operating system, based on the dNeedsTime flag in the dCtlFlags entry in the Device Control Entry (DCE) for the Apple CD-ROM driver. The purpose of this call is to perform operations required by the driver on an “as needed” basis. Since the CD-ROM device is a removable media device and does not generate interrupts when a disc is inserted, the only way to detect the fact that a disc has been inserted is to occasionally poll all devices that the driver last determined did not have a disc inserted. Almost all Macintosh programs make a call to the MacOS routine SystemTask(), and one of the functions of that routine is to occasionally make some CPU time available to all devices that request it. This is done by setting the dNeedTime flag in the Device Control Entry (DCE) for a given device.

Testing for disc insertions is the main function provided by this Control call. accRun also handles some other functions on a one-time basis. During the Macintosh boot process if the driver is installed before the System file is read in, the driver cannot install its custom CD-ROM Disk Switch subroutine. This is because when the System file is read in and installed it will overwrite the low memory global that defines the address of the standard Disk Switch routine. So, the installation of our custom Disk Switch routine is held off until the first invocation of SystemTask() to make sure that the System file has already been loaded and the standard Disk Switch routine has been initialized.

Input Parameters:

csCode	65
csParam[]	Don’t care.

Status Return Codes:

noErr	This is the only possible status return code.

ModifyPostEvent (76)

The purpose of this call is to allow an application to modify the driver’s behavior when non-HFS discs are inserted into the CD-ROM drive. The default action of the driver is to post a disc-inserted event for all discs that are inserted into the CD-ROM drive, regardless of whether or not the disc actually contains valid partition information and HFS file system. There may be some reason why the driver should not post the disc-inserted event and this call allows that "feature" to be turned off. When this feature is turned off for a specified SCSI ID, it will stay off until it is reactivated (via another call to ModifyPostEvent) or until the machine is restarted.

Input Parameters:

csCode	76	
csParam[0]	0	Do not issue disc-insert events for non-HFS CD-ROMs
	≠ 0	Issue disc-insert events for non-HFS CD-ROM discs
csParam[1-10]		Reserved for future use - must be zero (0).

Status Return Codes:

noErr	This is the only possible status return code.

ChangeBlockSize (79)

The purpose of this call is to allow an application to modify the "block size" the drive is currently using to read data with. The standard HFS file system block size is 512 bytes/block. Applications and file system translators dealing with non-HFS file systems may require other block sizes to read their data correctly. The Apple CD SC can handle block sizes of 256, 512, 1024, 2048, 2336 and 2340 bytes. In addition, the CD SC+ drive can handle a block size of 2056 bytes. In addition, the CD 300 can handle block sizes of 2352, 2646, and 2647. The CD 300 cannot handle a block size of 256.

For those block sizes greater than 2048 bytes, special consideration is given to the DCR bit in the Error Recovery Parameters page to enable/disable layered ECC. For the CD SC, the bit is set to 1 to disable ECC. Without this, accessing Mode 2 data blocks with a block size greater than 2048 would create a Check Condition error response by the drive. For all other drives, if there is a CD-I or XA disc inserted, the ECC will be enabled. For all other discs or if there is no media inserted, the ECC will be turned off.

Input Parameters:

csCode	79
csParam[0]	Block size the drive should use from now on
csParam[1-10]	Reserved for future use - must be zero (0).

Status Return Codes:

noErr	The block size change completed successfully.
paramErr	An invalid block size was specified.

UserEject (80)

The control call `UserEject` enables or disables the eject button on the drive, as well as enabling or disabling the `Eject (7)` call described earlier.

This call provides a method to disable/enable the manual eject button for a specified SCSI ID. Global variables are maintained for each SCSI ID that reflect the current state of the manual eject button. If the button is enabled, the driver sets the `dNeedTime` flag in the appropriate `DCE`, telling the driver to periodically poll the device to see whether a disc has been ejected, or inserted. This is the only way the driver has to ensure the validity of its global variables in this situation. When a disc is inserted, the button setting reverts back to the default setting.

Input Parameters:

csCode	80	
csParam[0]	1	Enable the manual eject button.
	≠ 1	Disable the manual eject button. {default}
csParam[1-10]		Reserved for future use - must be zero (0).

Status Return Codes:

noErr	The button was successfully enabled/disabled.
offLinErr	No disc inserted in the drive.

SetPollFreq (81)

The purpose of this call is to allow an application to specify a different polling frequency (in Macintosh time ticks) than the driver default value (100). This polling frequency is the value used for determining how often the accRun Control call is issued, as well as the polling frequency for a VBL task that is started during the execution of the driver's custom Disk Switch routine.

Input Parameters:

csCode	81
csParam[0]	Number of Macintosh ticks to be used for polling frequency
csParam[1-10]	Reserved for future use - must be zero (0).

Status Return Codes:

noErr	This is the only possible status return code.

ReadTOC (100)

The purpose of this call is to allow an application to receive various types of Table Of Contents (TOC) data from the drive. There are six varieties of the ReadTOC call.

csParam+0 (word)	Description
1	CD SC, CD SC+, CD 150: Transfers the first and last user track number (in BCD) of the first or only session. CD 300: Transfers the first track number of the first (or only) session and the last track number of the last (or only) session.

2	CD SC, CD SC+, CD 150: Transfers the starting address of the Lead Out area of the first (or only) session in BCD MIN-SEC-FRAME format. CD 300: Transfers the starting address of the Lead Out area of the last (or only) session in BCD MIN-SEC-FRAME format
3	Transfers the starting address of each track in a range of specified tracks, starting from the specified starting track and proceeding in ascending order until the specified buffer is filled or the data for the last track on the disc is transferred, whichever is less. Four bytes of information are transferred to describe the starting address. Only a CD 300 will be able to access TOC information beyond the first session on a multi-session disc.
4	Transfers the entire table of contents for a disc (all track entries and points A0, A1 and A2) to the specified buffer. (Not available on CD SC)
5	Transfers information about the number of sessions on the disc and the location of the last session. (CD 300 Only)
6	Transfers all Q-subcode entries in the lead-in areas of a disc (including TOC information), starting from the specified session number and proceeding in ascending order until the specified buffer is filled or the data for the last Q-subcode entry on the disc is transferred, whichever is less. (CD 300 Only)

For transfer types 1, 2, and 5, the TOC information is returned directly in the caller's csParam[] array. For types 3, 4, and 6, the caller is required to specify a buffer to place the information into. In the case of a type 4 transfer, the buffer must be 512 bytes long.

Input Parameters:

csCode	100

For Type 1:

csParam[0]	The transfer type value 1.
csParam[1-10]	Reserved for future use - must be zero (0).

For Type 2:

csParam[0]	The transfer type value 2.
csParam[1-10]	Reserved for future use - must be zero (0).

For Type 3:

csParam[0]	The transfer type value 3.

csParam[1-2]	The address of a buffer to return the requested information.
csParam[3]	The size of the buffer specified in csParam[1-2], in bytes.
csParam[4]	The MSByte (the upper 8 bits) of this 16-bit word must contain the starting track number, in BCD.

For Type 4 (Not available on CD SC):

csParam[0]	The transfer type value 4.
csParam[1-2]	The address of a buffer to return the requested information.

For Type 5 (CD 300 Only):

csParam[0]	The transfer type value 5.
csParam[1-10]	Reserved for future use - must be zero (0).

For Type 6 (CD 300 Only):

csParam[0]	The transfer type value 6.
csParam[1-2]	The address of a buffer to return the requested information.
csParam[3]	The size of the buffer specified in csParam[1-2], in bytes.
csParam[4]	The MSByte (the upper 8 bits) of this 16-bit word must contain the starting session number, in BCD.

Output Produced:

Type 1:

csParam[0]:15-8	MSByte contains first user track number (in BCD)
csParam[0]:7-0	LSByte contains last user track number (in BCD)

Type 2:

csParam[0]:15-8	MSB contains the MIN field of a MIN-SEC-FRAME descriptor that describes the Lead Out starting address (in BCD).
csParam[0]:7-0	LSB contains the SEC field of the Lead Out starting address descriptor.
csParam[1]:15-8	MSB contains the FRAME field of the Lead Out starting address descriptor.

Type 3:

The user must specify a return buffer address and the size of the buffer. The buffer is filled with 4-byte entries for each track in the requested range. Each entry is of the form:

Entry (byte offset)		Contents
0	Bits 7-4:	Reserved
		Bits 3-0: Control Field (see Track Control Field at the beginning of this document for more information)

1	The MIN field of a MIN-SEC-FRAME descriptor that describes the start of the associated track.
2	The SEC field of the track starting address descriptor.
3	The FRAME field of the track starting address descriptor.

Note: no entries will be returned for data tracks on CD-I discs.

Type 4:

This option is not valid for the CD SC drive. The user must specify a return buffer address. The buffer must be 512 bytes long, and is filled with the entire table of contents, which is returned in the form:

Entry (byte offset)	Contents
0	Drive Status.
1-5	TOC entry for point A0 (first track number).
6-10	TOC entry for point A1 (last track number).
11-15	TOC entry for point A2 (address of beginning of lead out area).
16-20	TOC entry for track 1.
.	.
.	.
.	.
506-510	TOC entry for track 99.

Each five byte entry is of the form:

Entry (byte offset)	Contents
0	Bits 7-4: Reserved
	Bits 3-0: Control Field (see Track Control Field at the beginning of this document for more information)
1	Point or track number.
2	PMIN.
3	PSEC.
4	PFRAME.

Type 5 (CD 300 Only):

csParam[0]	The first session number (in BCD).
csParam[1]	The last session number (in BCD).
csParam[2]	First track number of last session (in BCD).
csParam[3]:15-8	Bits 7-4:Reserved
	Bits 3-0: Control Fieldof first track in last session. (see Track Control Field at the beginning of this document for more information)
csParam[3]:7-0	The MIN field of a MIN-SEC-FRAME descriptor that describes the start of the first track of the last session (in BCD).
csParam[4]:15-8	The SEC field of the track starting address descriptor (in BCD).
csParam[4]:7-0	The FRAME field of the track starting address descriptor (in BCD).

Type 6 (CD 300 Only):

The user must specify a return buffer address and the size of the buffer. The buffer is filled with a 4-byte header and 8-byte entries for each Q-subcode entry in the requested range. The first three entries are for points A0, A1 and A2 and are followed by the track entries for that session. After that, the Q-subcode entries are sorted in ascending order using the ADR field and within each ADR group by ascending order of the POINT field. This pattern is repeated for each session on the disc.

The 4-byte header is of the form:

Header (byte offset)	Contents
0-1	Total number of TOC data bytes transferred excluding these two length bytes.
2	First session number on the disc (in BCD).
3	Last session number on the disc (in BCD).

Each TOC entry is of the form:

Entry (byte offset)	Contents
0	The session number (in BCD).
1	Bits 7-4: ADR field, i.e., Q-subcode mode field Bits 3-0: Control Field (see Track Control Field at the beginning of this document for more information)
2	TNO - Track number.
3	POINT. In some early CD-300 units, the values for Entries 2 and 3 are swapped. This is due to a firmware bug, which was corrected on later drives. You should always OR the two values together to always receive the proper value.
4	MIN.
5	SEC.
6	FRAME.
7	ZERO.
8	PMIN.
9	PSEC.
10	PFRAME.

The values come directly from the disc and will be in either BCD or hexadecimal format depending on the field. The session number, ADR/Control and POINT fields are in hexadecimal. All other fields are in BCD. For more information on the format of the TOC, please refer to the “Yellow Book” specification for CD-ROM discs, ISO 10149.

Status Return Codes:

noErr	TOC data successfully retrieved.
paramErr	An invalid TOC type specified. This can happen due to the TOC type being out of range, or due to the call not being supported by the drive in question (e.g. type 6 call issued to AppleCD SC drive.)
offLinErr	No disc in drive.

ReadQ (101)

The purpose of this call is to allow an application access to the Q subcode information for the current track. The current track is the track that the optical pickup is currently over on a disc.

Input Parameters:

csCode	101
csParam[0-10]	Unused on input. Q Subcode data are returned here.

Output Produced:

csParam[0]:[15-8]	MSB contains Control Field information. (see Track Control Field at the beginning of this document for more information)
csParam[0]:[7-0]	LSB contains current track number (in BCD).
csParam[1]:[15-8]	MSB contains the current index number in the current track.
csParam[1]:[7-0]	LSB contains the MIN field of a MIN-SEC-FRAME descriptor, which describes the relative running time from the beginning of the track.
csParam[2]:[15-8]	MSB contains the SEC field for a MIN-SEC-FRAME descriptor.
csParam[2]:[7-0]	LSB contains the FRAME field for a MIN-SEC-FRAME descriptor.
csParam[3]:[15-8]	MSB contains the AMIN field of an AMIN-ASEC-AFRAME descriptor, which describes the relative running time from the beginning of the disc.
csParam[3]:[7-0]	LSB contains the ASEC field of an AMIN-ASEC-AFRAME descriptor.
csParam[4]:[15-8]	MSB contains the AFRAME field of an AMIN-ASEC-AFRAME descriptor.
csParam[4]:[7-0]	LSB unused.

Status Return Codes:

noErr	Q subcode data successfully retrieved.

ReadHead (102)

The control call `ReadHead` returns the absolute minute-second-frame address and play mode information about a specified logical block. If a requested block is in a CD-Audio section of the disc, a `paramErr` will be returned. An application program can find out what the mode of a block is more easily with this command than with the `ReadExtended` command.

Input Parameters:

csCode	102
csParam[0-1]	32-bit logical block address to retrieve Header information for.
csParam[2-10]	Reserved for future use - must be set to zero.

Output Produced:

csParam[0]:[15-8]	MSB contains the AMIN field of an AMIN-ASEC-AFRAME descriptor, which describes the absolute running time from the beginning of the disc.
csParam[0]:[7-0]	LSB contains the ASEC field of an AMIN-ASEC-AFRAME descriptor.

csParam[1]:[15-8]	MSB contains the AFRAME field of an AMIN-ASEC-AFRAME descriptor.
csParam[1]:[7-0]	LSB contains the Play Mode of this AMIN-ASEC-AFRAME descriptor. For a description of possible play modes, please see the “Audio Play Modes” at the beginning of this document.

Status Return Codes:

noErr	Header information successfully retrieved.

AudioTrackSearch (103)

The purpose of this call is to position the optical pick-up at the specified audio address. In addition, you can also tell the drive either to start to play at the given address, or to pause at this address until an AudioPlay command is executed and in what Play Mode it should output the audio information in.

Input Parameters:

csCode	103
csParam[0]	The optical pick-up positioning address type (see “Optical Positioning Types” at the beginning of this document for more information.)
csParam[1-2]	32-bit value representing the search address required by the corresponding type value in csParam[0] (see above).
csParam[3]	A boolean value indicating what the drive should do when positioning of the pick-up is complete: 0 Enter a Hold Track state (“pause”) ≠ 0 Start playing the audio track, according the specified Play Mode.
csParam[4]	The “Play Mode” . (see “Audio Play Modes” at the beginning of this document for more information.)
csParam[5-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Optical pick-up successfully positioned.
paramErr	Invalid type, Play Mode or Search Address value specified.

AudioPlay (104)

The purpose of this call is to position the optical pick-up at the specified audio address and begin playback of the audio at that address. It can be similar to the AudioTrackSearch command that automatically begins playing at the specified Search Address (see the documentation on the AudioTrackSearch call for more detail.) However, this command can also be used to specify a “Stop Address”. The use for this would be to position the optical pick-up with an AudioTrackSearch command to indicate the starting position to begin playback and then specify where the playback should stop by issuing an AudioPlay command to mark the stopping address. Playback then continues from the position originally marked by AudioTrackSearch until the position indicated by AudioPlay.

Input Parameters:

csCode	104
csParam[0]	The optical pick-up positioning address type (see “Optical Positioning Types” at the beginning of this document for more information.)
csParam[1-2]	32-bit value representing the search address required by the corresponding type value in csParam[0].
csParam[3]	A boolean value indicating whether the address specified in csParam[1-2] is a starting address or a stopping address. 0 The address specified is a Starting Address. ≠ 0 The address specified is a Stopping Address.
csParam[4]	The “Play Mode” (see “Audio Play Modes” at the beginning of this document for more information.)
csParam[5-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Operation successful.
paramErr	Invalid type, Play Mode or Search Address value specified.

AudioPause(105)

The purpose of this call is to cause the CD-ROM drive to either enter a Hold Track (“Pause”) or Play state. This command is only valid after the issuance of an Audio Track Search, or Audio Play command.

Input Parameters:

csCode	105
csParam[0-1] (long)	A boolean value that tells the CD-ROM drive either to pause or continue. 0 Release the “pause” state and continue playing at the same Q subcode address before the last AudioPause. 1 Enter a Hold Track (“Pause”) state.
csParam[2-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	CD-ROM device successfully paused or released from pause.
paramErr	Invalid pause value specified.

AudioStop (106)

The purpose of this call is to specify an address to the CD-ROM drive at which it should end the current Play operation.

Input Parameters:

csCode	106
csParam[0]	The optical pick-up positioning address type (see “Optical Positioning Types” at the beginning of this document for more information.)
csParam[1-2]	32-bit value representing the search address required by the corresponding type value in csParam[0].
csParam[3-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Optical pick-up successfully positioned.
paramErr	Invalid type or Search Address value specified.

AudioStatus (107)

The purpose of this call is to return the current audio play status information and the starting Q subcode address for the current track. The audio play status information includes the Play Mode, and Status and Control information for the current track.

Input Parameters:

csCode	107
csParam[0-10]	Unused on input. Audio Status data are returned here.

Output Produced:

csParam[0]:15-8	The MSB contains the Status Field. 0 Audio Play operation in progress 1 CD-ROM device currently in Hold Track (“Pause”) state 2 MUTING-ON operation in progress 3 Audio Play completed 4 Error occurred during audio play operation 5 Audio play operation not requested
csParam[0]:7-4	Reserved.
csParam[0]:3-0	Play Mode. (see “Audio Play Modes” at the beginning of this document for more information.)
csParam[1]:15-12	Reserved.
csParam[1]:11-8	The LSNibble contains the Control Field.(see Track Control Field at the beginning of this document for more information)
csParam[1]:7-0	The LSB contains the AMIN field of an AMIN-ASEC-AFRAME descriptor that describes the current Q-subcode.
csParam[2]:15-8	The MSB contains the ASEC field of an AMIN-ASEC-AFRAME descriptor for the current Q-subcode.

csParam[2]:7-0	The LSB contains the AFRAME field of an AMIN-ASEC-AFRAME descriptor for the current Q-subcode.

Status Return Codes:

noErr	Status information successfully read.
offLinErr	No disc in drive.
controlErr	An error occurred in the Audio Status command execution.

AudioScan (108)

The purpose of this call is to request the CD-ROM drive to perform a fast-forward or fast-reverse operation starting from a specified address.

Input Parameters:

csCode	108
csParam[0]	The optical pick-up positioning address type (see "Optical Positioning Types" at the beginning of this document for more information.)
csParam[1-2]	32-bit value representing the search address required by the corresponding type value in csParam[0].
csParam[3]	The direction the scan should take place: 0 Scan in the fast-forward direction $\neq 0$ Scan in the fast-reverse direction
csParam[4-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Optical pick-up successfully positioned.
paramErr	Invalid type or Search Address value specified.

AudioControl (109) [not available on AppleCD SC]

The purpose of this call is to set the output volume for the drive's audio channels. Each channel can be given a value from zero to 255, zero indicating that the channel output is muted, 255 indicating maximum volume. Note that this command is not available on the CD SC.

Input Parameters:

csCode	109
csParam[0]:15-8	MSB Left channel volume (0-FFh).
csParam[0]:7-0	LSB Right Channel volume (0-FFh).
csParam[1-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Audio volume successfully set.
paramErr	on AppleCD SC, this call is not supported.

ReadMCN (110) [not available on AppleCD SC]

The purpose of this call is to allow an application access to the Media Catalog Number for the disc. The Media Catalog Number consists of the UPC/Bar Code information for the disc. Input Parameters:

csCode	110
csParam[0-10]	Unused on input. MCN data are returned here.

Output Produced:

csParam[0]:[15-8]	MSB contains MCVal bit (bit 15). If this bit is set to one, then the Media Catalog Number data was found and the rest of the returned fields are valid. If this bit is set to zero, then the Media Catalog Number data was not found and the rest of the returned fields are invalid.
csParam[0]:[7-0]	LSB contains MSB of the Media Catalog Number (UPC/Bar Code) data if the MCVal bit is set.
csParam[1-7]	Contains the rest of the Media Catalog Number (UPC/Bar Code) data if the MCVal bit is set. The LSB of csParam[7] contains the LSB of the Media Catalog Number data.

Status Return Codes:

noErr	Media Catalog Number data successfully retrieved.

ReadISRC (111) [not available on AppleCD SC]

The purpose of this call is to allow an application access to the International Standard Recording Code information for a specified track on a disc. For more information about the ISRC, please refer to ISO 10149.

Input Parameters:

csCode	111
csParam[0]	A track number (in BCD).
csParam[1-10]	Unused on input. ISRC data are returned here.

Output Produced:

csParam[0]:[15-8]	MSB contains TCVal bit (bit 15). If this bit is set to one, then the ISRC data was found for the specified track and the rest of the

	returned fields are valid. If this bit is set to zero, then the ISRC data was not found for the specified track and the rest of the returned fields are invalid.
csParam[0]:[7-0]	LSB contains MSB of the ISRC data for the specified track if the the TCVal bit is set.
csParam[1-7]	Contains the rest of the ISRC data if the TCVal bit is set. The LSB of csParam[7] contains the LSB of the ISRC data.

Status Return Codes:

noErr	ISRC data successfully retrieved.

ReadAudioVolume (112) [not available on AppleCD SC]

The purpose of this call is to return the current audio volume control data for the drive. This call is not available on the CD SC.

Input Parameters:

csCode	112
csParam[0-10]	Unused on input. Audio Volume data are returned here.

Output Produced:

csParam[0]:15-8	The MSB contains the left channel volume (0-FFh).
csParam[0]:7-0	The LSB contains the right channel volume (0-FFh).

Status Return Codes:

noErr	Status information successfully read.
offLinErr	No disc in drive.
ioErr	An error occurred in the ReadAudioVolume command execution.

GetSpindleSpeed (113)

The purpose of this call is to return the current rotational speed of the drive. This call always returns 00 on the Apple CD SC and AppleCD SC Plus/CD 150.

Input Parameters:

csCode	113
csParam[0-10]	Unused on input. Speed information is returned here.

Output Produced:

csParam[0]:	The current rotational speed: 00 = Normal speed FF = Maximum speed (AppleCD 300 only)

Status Return Codes:

noErr	Speed information successfully read.
ioErr	An error occurred in the GetSpindleSpeed command execution.

SetSpindleSpeed (114)

The purpose of this call is to set the rotational speed of the drive. This call will return paramErr for an AppleCD SC or AppleCD SC Plus/CD 150 if the input parameter is not 00 (normal speed.)

Input Parameters:

csCode	114
csParam[0]	Speed setting: 00 = Normal speed (i.e. 150 KB/sec) FF = Maximum speed (i.e. 300 KB/sec)
csParam[1-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Rotational speed successfully set.
paramErr	An invalid speed setting was specified.
ioErr	An error occurred in the SetSpindleSpeed command execution.

ReadAudio (115) [AppleCD 300 only]

The purpose of this call is to read digital audio and/or subcode data from the disc into a specified buffer. The starting audio address may be specified in one of three formats (see the “Audio Play Modes” at the beginning of this document for more information.) There are four possible formats for the audio data that is returned:

Type values	Description
0	Audio data only —2352 bytes per audio block.
1	Audio data with Q-subcode—2352 audio bytes, 10 bytes Q-subcode, and 6 zero bytes = 2368 bytes per block.
2	Audio data with all subcode data (P~W)—2352 audio bytes and 96 subcode bytes = 2448 bytes per block.
3	All subcode data only (P~W)—96 subcode bytes per block.

The position of an audio block cannot be determined exactly. Thus, two ReadAudio calls with the same starting address will not necessarily return the same data in the same position. Typically the data will be shifted up to 1.5 frames (3528 bytes). To read large sections of audio data using multiple ReadAudio calls and not have a gap are overlap in the audio data, the ReadAudio calls must occur frequently enough to prevent the internal drive buffer from overflowing. If the Macintosh is too slow, a controlErr will be returned along with the audio data. The controlErr indicates that the audio data just received has a gap or overlap in it.

For type 2 and type 3, the P and Q subcodes are accurately aligned to the CD block boundary, but the R through W subcodes may be offset by one packet (i.e. 24 subcode bytes) from the P and Q subcodes. Thus, in any 96-byte subcode data from block *n,* the last 24 bytes contain R through W subcodes from the next block *(n+1)*.

The current volume setting has no effect on the audio data that is returned. When using ReadAudio with type 3, the audio is played through any connected speakers. The audio can be muted to avoid hearing anything while reading the subcode data. Data is transferred using the I/O style previously set by the ChooseIOStyle driver call. For types 0, 1, and 2 the data rate can be adjusted by changing the drive speed prior to calling ReadAudio. For type 3 only, subcode data is always transferred at an average throughput of 150K bytes per second.

Input Parameters:

csCode	115
csParam[0]	Audio Type (see above)
csParam[1-2]	The address of the buffer to return the requested information.
csParam[3]	The starting address type (see “Optical Positioning Types” at the beginning of this document for more information.)
csParam[4-5]	32-bit value representing the start address required by the corresponding type value in csParam[3].
csParam[6]	The number of consecutive audio blocks to return.
csParam[7-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Audio data successfully retrieve.
paramErr	An invalid audio type, address type, or audio address specified.
controlErr	An error occurred in the ReadAudio command execution.

ReadAllSubcodes (116) [AppleCD 300 only]

The purpose of this call is to read consecutive blocks of subcode information (P~W) as the drive is playing audio. When one of the commands is issued that begins playing audio, the subcode information corresponding to the audio blocks is stored in a buffer within the drive. ReadAllSubcodes retrieves the subcode information from this buffer. If the number of subcode blocks requested exceeds the quantity currently in the buffer, the control call waits until more subcode blocks are available before returning. An option allows any subcode data currently in the buffer to be purged, allowing the host to re-synchronize to the latest subcode data. This call is only avalilable on the CD 300.

If the Macintosh does not send ReadAllSubcode requests frequently enough, a controlErr will be returned indicating that the internal drive buffer overflowed and some subcode data was lost.

Note: The R through W subcodes that are returned are accurately aligned to the CD block boundary, but the P and Q subcodes may be offset by one packet (i.e. 24 subcode bytes) from the R through W subcodes. Thus, in any 96-byte subcode data from block *n,* the first 24 bytes may contain the P and Q subcodes from the previous block *(n+1).*

Input Parameters:

csCode	116
csParam[0-1]	The address of the buffer to return the requested subcode blocks.

csParam[2]	A boolean value indicating if the current subcode information in the drive buffer should be purged before retrieving any subcode data. 0 retrieve buffered subcode blocks ≠ 0 delete any blocks in the subcode buffer & wait for new data
csParam[3]	The number of consecutive subcode blocks (96 bytes per block) to return.
csParam[4-10]	Reserved for future use - must be set to zero.

Status Return Codes:

noErr	Audio data successfully retrieve.
controlErr	An error occurred in the ReadAllSubcodes command execution.

Status Call Descriptions

DriveStatus(8)

The purpose of this call is to return information about the drive/disc as described in Inside Macintosh, Volume II, page 215.

Input Parameters:

csCode	8
csParam[0-10]	Not defined on entry. On exit, contains information that maps to the `DrvSts` record/structure.

Status Return Codes:

noErr	This is the only possible status return code.

Get2KOffset (95)

The purpose of this call is to return information to the caller that allows subsequent Prime calls to be aligned to the 2K physical block size of the CD-ROM. This can be used in applications (e.g. QuickTime) to increase streaming performance by avoiding data requests that span across a 2K disc block. This call returns the number of bytes between the beginning of the 2K physical block and the first byte read in the previous Prime call. Possible values are 0, 512, and 768. For example, if a Prime call causes a read of the last 512 bytes within a 2K physical block, this call will return 3 x 256 = 768 bytes. This is the offset from the beginning of the 2K block to the start of the first data byte read. Note that this call assumes that the file system actually makes a Prime call and that a file system cache did not fill the read request.

Input Parameters:

csCode	95
csParam[0-1]	Not defined on entry. On exit, it contains the offset between the beginning of the 2K physical block and the first byte of the last read call.
csParam[2-10]	Not used during this Status call.

Status Return Codes:

noErr	The offset for the specified SCSI ID was returned.
statusErr	Prime was not previously called.

GetDriveType (96)

The purpose of this call is to return to the caller the type of CD-ROM drive located at the specified SCSI ID.

Input Parameters:

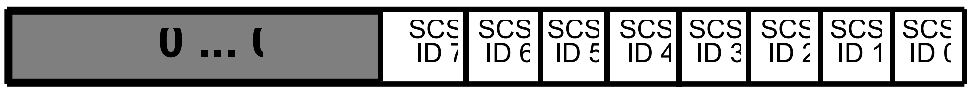
csCode	96
csParam[0]	Not defined on entry. On exit, it contains a code indicating the type of CD-ROM drive located at the specified SCSI ID. These codes are defined as follows: 1 Drive is an AppleCD SC 2 Drive is an AppleCD SC Plus / AppleCD 150 3 Drive is an AppleCD 300
csParam[1-10]	Not used during this Status call.

Status Return Codes:

noErr	The drive type for the specified SCSI ID was returned.
statusErr	The specified SCSI ID is not a CD-ROM drive.

WhoIsThere (97)

The purpose of this call is to return to the caller a list of devices that the CD-ROM driver is responsible for. It returns in csParam[0] a bit array, each member of which has a value of one if an Apple CD-ROM drive exists at the corresponding SCSI ID or a value of zero if it does not. The order of the bit array is the same as how Motorola orders bits in a byte; that is, the LSB (least significant bit) is the right-most bit, or the 20 bit. Since a maximum of 8 SCSI devices can be on the SCSI bus (addresses 0-7), the bit array takes up a byte, and will be located in the lower half of csParam[0], as shown here:


0 ... (
SCS ID 7
SCS ID 6
SCS ID 5
SCS ID 4
SCS ID 3
SCS ID 2
SCS ID 1
SCS ID (

Input Parameters:

csCode	97
csParam[0]	Not defined on entry. On exit, it contains the above bit array.
csParam[1-10]	Not used during this Status call.

Status Return Codes:

noErr	This is the only possible status return code.

This call only works for a specified driver. Many third-party drivers have the same internal driver name as the Apple CD-ROM driver. (e.g. “.AppleCD”) In order to detect all possible CD-ROM drives supported by all possible CD-ROM drivers, you should use the following sample code instead:

In C:

```
#define dRAMBased          0x0040

unsigned char WhereCDs()
{
 unsigned char where = 0;    /* assume no drives handled at beginning */
 short     scsiAddress;
 short     drvrRefNum;
 DCtlHandle    aDCtlEntry;
 StringPtr   aDriverName;
 StringPtr   appleDriverName = "\p.AppleCD";

 for (scsiAddress = 0; scsiAddress < 7; scsiAddress++) {
   drvrRefNum = -33 - scsiAddress;

   aDCtlEntry = GetDCtlEntry(drvrRefNum);
   if (aDCtlEntry != nil) {
     if ((**aDCtlEntry).dCtlFlags & dRAMBased)
       aDriverName = (StringPtr) ((*(DCtlPtr)aDCtlEntry).dCtlDriver+18);
     else
       aDriverName = (StringPtr)((**aDCtlEntry).dCtlDriver+18);
     if ( EqualString(aDriverName, appleDriverName, false, false) == true )
       where |= (1 << scsiAddress);
   }
 }
 return where;
}
```

In Pascal:

```
FUNCTION WhereCDs: Byte;

CONST
   kDriverName = '.AppleCD';  { Only recognize drivers whose names are '.AppleCD' }

 VAR
   where: LONGINT;
   scsiAddr: INTEGER;
   drvrRefNum: INTEGER;
   aDCtlEntry: DCtlHandle;
   aDriverName: StringPtr;
```

```
BEGIN
 where := 0;

 FOR scsiAddr := 0 TO 7 DO
   BEGIN
     drvrRefNum := -33 - scsiAddr;

     aDCtlEntry := GetDCtlEntry(drvrRefNum);   { Get the driver's control entry }
     IF aDCtlEntry <> NIL THEN                 { Is there a driver there? }
       BEGIN
       { Go find the driver's name }

         WITH aDCtlEntry^^ DO
           IF BAND(dCtlFlags, $0040) <> 0 THEN { Is it handle (RAM) based? }
             aDriverName := StringPtr(LONGINT(Handle(dCtlDriver)^) + 18)
           ELSE                                { It must be pointer (ROM) based. }
             aDriverName := StringPtr(LONGINT(dCtlDriver) + 18);

       { Is it named '.AppleCD'? }
         IF aDriverName^ = kDriverName THEN
           BSET(where, scsiAddr);
       END;
   END;

 WhereCDs := where;
END;
```

GetBlockSize (98)

The purpose of this call is to return to the caller the current block size of the CD-ROM drive. If an application has set the block size to 512, but the driver has set the actual block size to 2048 and is doing the 2048 byte to 512 byte conversion, this call will return 512 as the current block size.

Input Parameters:

csCode	98
csParam[0]	Not defined on entry. On exit, it contains the block size of the disc inserted in the specified SCSI ID.
csParam[1-10]	Not used during this Status call.

Status Return Codes:

noErr	The block size of the disc in the specified SCSI ID was returned.
offLinErr	The specified SCSI ID does not contain a disc.

Further Reference:

- AppleCD SC Developer's Guide, Second Edition (APDA A7G0023/C)
- Apple CD-ROM Handbook (Addison-Wesley)
- Inside Macintosh Volume II - The Device Manager
- Inside Macintosh Volume VI - The File Manager
- CD-ROM and the Macintosh Computer, on the Developer CD series
- ISO 10149 Information technology — Data interchange on read-only 120 mm optical data disks (CD-ROM) (“Yellow Book”)
- CEI IEC 908 Compact Disc digital audio system (“Red Book”)