AMERICAN INTERACTIVE MEDIA

AIM Technical Note #62

Living with Reality: Workarounds for Remaining Bugs in CD-RTOS 1.1

Kitty van Laan

February 14, 1991

In the state currently reached in CD-I players, many of the older problems have been resolved and a very workable situation has been created. However a number of issues remain: the actual available implementations may deviate from Green Book definitions or the Green Book definitions are ambiguous. The time has come for title productions to face these items and plan strategies to work around them, rather than to rely on fixes in a future version of the player or its software. This note describes some of the points that have to be taken into account and suggests workarounds whenever possible. The notion of a workaround is such that the solution that is suggested is usable towards compatible titles: discs that utilize these workarounds will work well on future players, even after the player bugs have been fixed.

Table of Contents

Living with Reality: Workarounds for Remaining Bugs in CD-RTOS 1.1

1. Introduction

CD-I players have gone through an evolution of various versions of hardware and software leading to the 1.1 release that we have today. There is a fundamental philosophical difference between earlier updates and the current one. While the current version is certainly not the last, it is the last version* that we will find in our development and test platforms before the consumer launch. Also, this version of CD-I players is currently being sold commercially, and there is no plan to do any retrofits to these players. There is always the risk that future discs that rely on later refinements would not play correctly on players that are out in the field, thus violating the compatibility principle that distinguishes CD-I from other, more computer-oriented technologies. Where these discrepancies are merely cosmetic, anticipation of future improvements might still be the best title strategy. However, when functionality is affected, workarounds become mandatory for titles in the AIM catalog. This is not only for the abstract "good of CD-I," but it is also pragmatic because there is no way that a title can make it through the test process and be released commercially if it shows functional defects on the only available platforms.

* The "last version," as used in this note, refers to the 1.1 version of CD-RTOS. There will be a final update to this revision level that solves some of the problems of the pre-release currently available. Throughout this note, items affected by this revision plan are marked by an asterisk.

2. Reference Versions.

During CD-I development, many versions of development software, libraries, "include files," utilities and players have been used. Now is the time to make a "clean sweep" and make sure that all discs that are part of the AIM catalog are using the same version of the software and run on the same "reference" player.

In regard to the software, the reference version should be CD-RTOS 1.1, based on OS-9 version 2.4 together with a corresponding set of libraries. The reference player is the Philips 180 with CD-RTOS 1.1 ROMS and the DSP ROM update that AIM will receive in April, 1991. The latter component is required to fix a problem of losing sectors when seeking while playing sound maps. Since there is no known workaround for this problem, we have to accept the fact that titles that are affected by this particular bug can be completely tested only at the eleventh hour before launch when this particular fix is made available. In Appendix A, there is an overview of how the various versions of the reference package can be identified.

The remainder of this note describes the three categories of problems. It starts with cosmetic flaws and their recommended workarounds; this is followed by functional problems. The avoidance of functional problems is mandatory for AIM titles. Finally, there are some grey areas in the Green Book where an interpretation is given of how best to proceed.

3. Cosmetic Problems.

Some of the problems are merely cosmetic. The esthetics of the imagery could be affected or the quality of the audio might not be optimal. Some of these problems are due to the hardware and have no known workarounds. Others can be avoided by following a proper procedure. Since the effects of flaws in this area are not considered show stoppers, the observation of the workarounds is encouraged, but not at all cost. AIM does not consider these workarounds to be mandatory. They are subject to normal cost/benefits trade-offs.

3.1. Video. The player image is off-center by roughly 12 pixels.

This is a timing problem in the chip that is used to generate the video. Future versions might have correct timing, and/or manufacturers might want to compensate for it by delaying the sync signal so that the active video portion of the lines is properly centered.

Workaround: Reduce the safety area by 12 pixels on the right hand side. So, instead of using the center 320 pixels, restrict the area for critical information down to 308 pixels, offset to the left.

Warning: Do not try to compensate for the shift in software by shifting the whole image to the left. Once manufacturers fix the hardware problem, your disc would exhibit imagery shifted to the left.

3.2. Video. Poor CVBS quality in non-interlace mode.

The current version of the 380 player generates an interference pattern in the subcarrier frequency when connected to a Comb-filter TV set. This problem is solved in 400 and 200 series of players.

Workaround: Take your audio on a non-comb filter TV set.

3.3. Video. Matte function could result in incorrect rightmost pixel.

The current hardware resets the matte flag to FALSE before the last pixel of the current line, instead of doing so after termination of the whole line. This could result in an incorrect rightmost pixel.

Workaround: Make the plane and matte arrangement so that the FALSE setting of the flag corresponds to the desired display. Alternatively, make the right most pixel in your images black.

3.4 Audio. Mute circuitry active at beginning and end of sounds.

This affects both sound maps and audio played directly from disc. The player contains "soft mute" circuitry that has a time constant associated with it. This suppresses audio at the very beginning and end of a sound.

Workaround: Make sure that the first and last 50 msec of a sound map or file do not contain vital information. If necessary, pad the audio with silence or other appropriate sound.

3.5 Audio. Deemphasis not always activated by the decoder.

When a sound map interrupts another sound map, the deemphasis is not activated for the interrupting sound map. This leads to an incorrect frequency characteristic.

Workaround: Always use "Emphasis off" in audio production for sound map audio. (See FFGS Note)

3.6 Video. Non-atomic, dual video plane update

The display logic scans the memory for both video planes in perfect synchronization. The video driver, however, cannot write to the hardware in an atomic operation. This can result in flashes when Plane B image coding methods are modified from DCP-A while plane B is visible.

Workaround: See ADM Technical Note #03.

3.7 Audio. DC offset at end of single sound maps.

In the 180 players, single sound maps have a slight DC offset at the end. This causes a kind of mild "thump" sound to be audible.

Workaround: Don't worry about it. It is minor and will go away.

3.8 Audio. SC_Atten causes some interference in the audio.

In the 180 players, there is slight audio interference when SC_Atten is invoked while a sound map is playing. This seems to depend on the timing. This problem is still under investigation.

Workaround: Do not modify attenuation level while audio is playing from a sound map. If your application requires this functionality, you might accept the cosmetic problem as non-disturbing or try experimenting with the timing relationship between the SM_Out and SC_Atten.

3.9 Audio. Seamless channel switching by SS_CChan impossible.

The functionality offered by SS_CChan is fine for switching channels that are unrelated (e.g., language choices). However, it is not possible to achieve seamless channel switching this way, because of the activation or mute circuitry during the switch.

Workaround: For applications requiring seamless channel switching, audio has to be routed through memory, filling sound maps from the disc, and the switch can be performed by feeding the proper sound map to the audio processor.

3.10 Audio. Extraneous sound at the beginning of RTF audio.

When RTF audio is played and a looped C mono sound map interrupts the RTF audio, the next time an SS_Play is started with RTF audio, a short piece of the sound map is heard right before the RTF audio.

Workaround: Still under investigation. One suggestion might be to "flush" the audio processor buffer with a two-sector A stereo sound map containing silence. However this has not yet been confirmed. Please contact the author if you have any suggestions for workarounds.

4. Functional Problems.

This category contains problems that are real bugs that could affect the functionality of an application, rather than cosmetic issues. Some of these are hardware related, and some of them are system software bugs. **Application of a workaround is mandatory for AIM titles**, since the resulting defects are considered to be unacceptable for consumer quality titles.

4.1 CDFM. Sync-pattern sensitivity.

Due to a problem in the CDIC chip that is used in the CD-interface, both explicit seeks (ISSeek, SS_Seek) and implicit seeks (ISRead, SS_Raw and SS_Play) are not reliable if a sync pattern (see FFGB page II-20) exists within 2 sectors of a seek point in the descrambled data.

Workaround: During the authoring stage, prevent the synchronization pattern from existing in the relevant data area. AIM has created a test program that checks for this pattern.

4.2 Video. CLUT reload function for banks 2 & 3.

A chip problem in the VSR prevents reliable access to CLUT banks 2 and 3 from path 1, once those banks have been accessed for loading entries from path 0 in CLUT8

Workaround: There are two alternatives for dealing with this problem:

4.2.1 When using CLUT 8 mode to reload CLUT from the DCP that is associated with Plane A, always put the reload instructions for bank 0 and 1 at the end, instead of the more natural sequence of putting banks 2 and 3 at the end.

4.2.2 Avoid using the CLUT8 mode for loading the CLUT. Instead, use CLUT 7 for both planes to load the CLUT entries, and only then change the image coding method to CLUT8, if required. (See AIM Technical Note #46).

4.3 Video. Matte instruction used at x=0 affects whole video plane.

When the ICF is changed by a matte instruction while X location = 0 (See FFGB, page V-84), then the ICF affects the whole video plane.

This is irrespective of which LCT is used.

Workaround: Use the LCT instruction "Load Image Contribution Factor" (FFGB V-80) to set the desired value of ICF for the leftmost pixel.

4.4. CPU instructions BSET, BCLR and BCHG.

When the following instructions are used in post increment addressing mode, an old version of the CPU will not execute them correctly.

```
BSET      register,  (An)+
BCLR      register,  (An)+
BCHG      register,  (An)+
```

Workaround: Use the following combination of instructions:

```
Bxxx      register,  (An)
ADDQ.L    #1,  (An)
```

It has been verified that the Motorola compiler never generates these instructions in post increment addressing modes.

4.5 CDFM. SS_CChan does not work correctly.

SS_CChan cannot be used to change channels if play has been started with no channels selected. This is because the current version of the DSP changes channels only upon retrieving a selected sector. When the play has been started with no channels selected, the opportunity for changing never exists

Workaround: Do not start a play with no channels selected. Alternatively, avoid using SS_CChan altogether, and route audio data through memory, setting the right information into sound maps, and create a seamless channel switch instead.

4.6 CDFM. Disc access sometimes returns with an error.

When the disc is accessed for the first time, the system sometimes returns with an error.

Workaround: Implement an error checking routine and retry at least once.

4.7 CDFM. Seek after partial sector read fails.*

If the internal CDFM buffer contains any remaining characters when an SS_Seek is performed, the SS_Seek returns an error. This happens in the case of a read of a partial sector.

Workaround: Always read a full sector when doing an ISRead. This is also recommended from a performance point of view.

4.8 CDFM. Pause/Continue problematic.*

Two problems are related to the Pause/Continue of a real-time file or CD-DA. The first is that the file position pointer is not updated dynamically during SS_Play, or even upon execution of an SS_Pause. Thus, there is no position information available to the application. This prevents accurate synchronization of video together with audio (lyrics, synchronized light shows, etc.). Even more severe is another bug that causes the random loss of sectors when Pause and Continue are activated in quick succession

Workaround: Avoid Pause Continue altogether. Translate a user request for a pause into an SS_Abort function call. SS_Pos now returns the correct information. Instead of a Continue, set up a new play from the position as indicated by SS_Pos (or GS_Pos() in the C-bindings).

4.9 CDFM. When play ends on an audio sector that has an EOR in it, the audio from that sector is not heard.*

Workaround: Make sure this situation never occurs in the disc image.

---

* This problem will be corrected by the final 1.1 release.

- *Pad the audio file with an additional sector of silence. This is not always possible if audio is part of a larger file that needs to be played contiguously across the EOR in other cases.*
- *Modify the disc building script so that the EOR is contained in a different (non-audio or dummy) sector*

5. Incomplete Definitions.

Despite best efforts to make the 1.1 release of the Green Book fully current, some issues have slipped through because it has not been possible to reach agreement on exact phrasing in a timely fashion. The items mentioned in this part of the note constitute the author's best interpretation of the most likely future developments in this area. As such, compatibility cannot be guaranteed on such a loose and informal basis. At best, these issues can grow into a *de facto* standard that may ultimately be absorbed into a future release of the Green Book.

5.1 Sound map done signal timing vaguely defined.

In the current version of the Green Book, there is no description of exactly when the "sound map done" signal is generated. Thus, the application has a "window of uncertainty" of a duration equal to the time it takes to play back the current sector of the sound map. This makes the generation of contiguous audio from sound maps questionable.

Interpretation: The signal is sent at the moment the transfer of the last sector of the sound map to the actual audio processor is completed. Due to local buffering in the audio processor, this gives a time window equivalent to the play back time of one sector of audio for the application to react with a new SM_Out to achieve contiguous audio operation.

5.2 SM_Stat information varies in different player architectures.

Interpretation: The status information that is obtained from this function call pertains to the next sector to be transferred from memory to the local audio processor buffer. However, player architectures vary in that some effectively have a single sector audio processor buffer, while others (including the Philips reference player) have a two sector buffer. So, for true portability, the SM_Stat information cannot be used for synchronization purposes between audio and video presentations. These have to be timer driven. The only practical use of the SM_Stat information is for buffer regulation purposes in an SM_Out environment.

5.3 Front panel controls unspecified.

One serious omission in the current Green Book is that the controls that are accessible to the user, both on the front panel, as well as on a remote control, are not available to the application in any standardized way. This is especially important for discs with linear controls (like music titles). In the appendix, we are publishing the way in which the Philips 600 and 200 series of players implement this feature in the hope that wide distribution of this knowledge might cause *de facto* standardization in this area.

Recommendation: *Use this functionality as a feature only in parallel to screen-based functionality. In other words, make sure that your application runs even if these functions are not available on a base case player.*

APPENDIX A

PHILIPS Hasselt	Functional specification on change request of player control keys	20/11/90
ENGINEERING IMS Software		J. Wuyts

1. Introduction :

This document describes the functional specification for the 'player control keys'.

2. Purpose :

1. The 'player control keys' should be made available for an CDI-application "green keys". The remaining keys are only available for the player shell.

2. The function keys on the keyboard can be used as control keys.

3. Specification :

There are 2 kinds of applications:

- System indepencent applications:
  They can only read green keys from the key driver.

- System depencent applications: e.g. player shell
  They are able to read green and non-green keys from the key driver.

3.1 System indepencent applications

A. Green Keys :

All system indepencent applications can only read the following green keys from the player control key driver:

- Codes for keygroup 0:

Function:	Code:
Play	30
Stop	31
Pause	32
Next Track	33
Previous Track	34
Search backward	35
Search forward	36

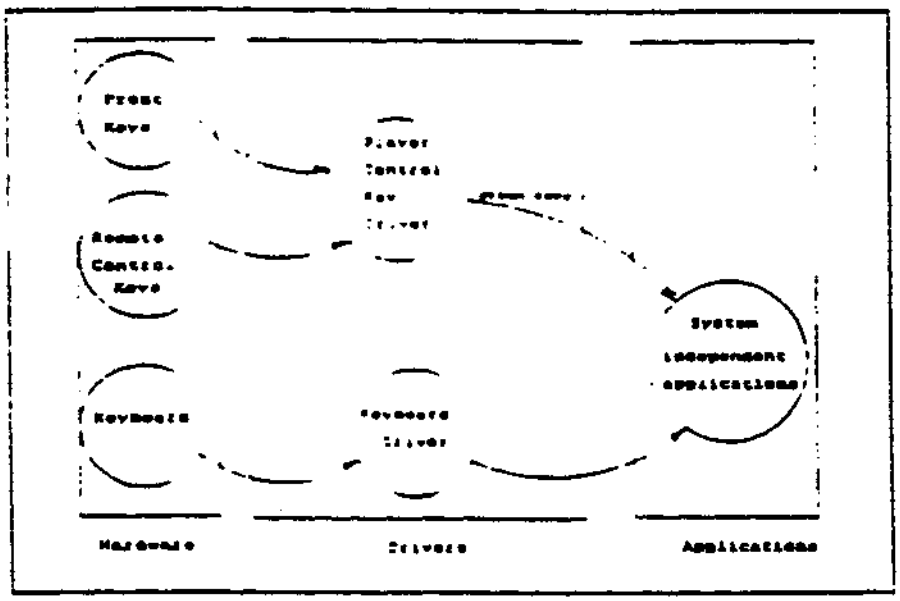
- Codes for keygroup 1:

Function:	Code:
Numeric pad :	10
	11
	12
	13
	14
	15
	16
	17
	18
	19

These key codes that green are not pusned into the buffer of the key driver that means that they are not available for applications

An example in the audio-domain will clarify the meaning of these functions :

Function:	Description:
Play:	Starts play or replay from the beginning of the actual track
Stop:	For stopping play
Pause:	For interrupting playing and restarting from the interrupted position
Next Track:	For selecting the next track
Previous Track:	For selecting the previous track
Search backwards:	For fast search (backwards) to a particular passage during play
Search forwards:	For fast search (forwards) to a particular passage during play



B. DSD's :

When keygroup 8 is used in special entry (number 11) has to be provided in the DSD:

e.g. : 11: keys:KG="1,2,3,4,5,6,7,8":

in general : 11: keys:KG=string

Keygroups defined by string:

1 : Programmable function keys
2 : Alphanumeric keys
3 : Special keys
4 : Cursor control keys
5 : Formatting keys
6 : Labelled function keys
7 : Numeric keys
8 : Player control keys
All : All keygroups

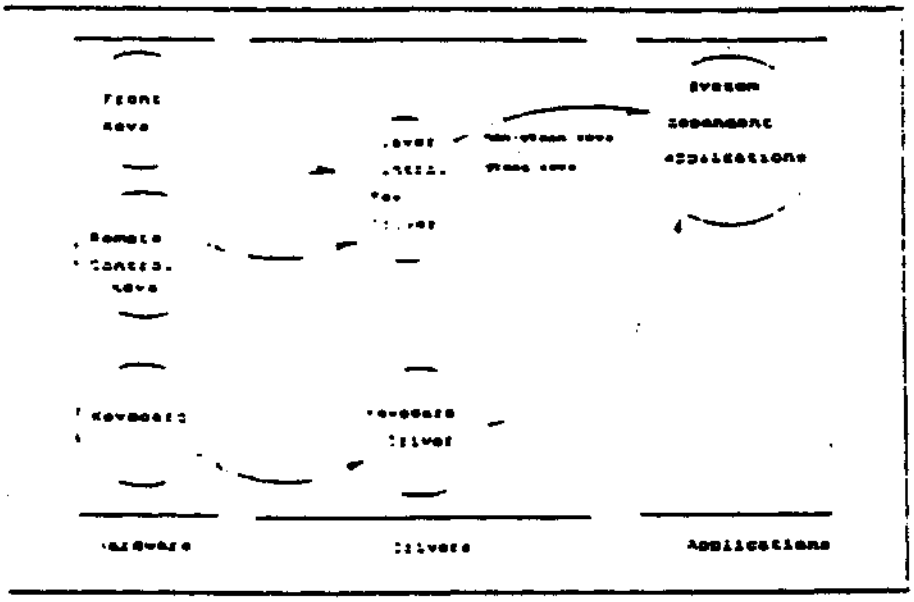
C. Interface mechanism:

There is a group of functions available for the applications to interface with the drivers:

- ISReac = Read string
- KB_Read = Read Keyboard Event
- KB_Ssig = Send Signal on Keyboard Data Ready
- KB_Relea = Release Device
- KB_Repeat = Set Keyboard Latency and Repeat Times
- KB_Stat = Determine Status of Keyboard
- KB_Ready = Check for Data Ready

For more details on this functions cfr. the Green Book Chapter VII.2.3.6 (Keyboard Input Functions).

3.2 System dependent applications (player shell)



The interface mechanism behaves exactly the same for system dependent as for system independent applications: the same interface functions, with the same parameters, etc. But there is an extension provided for system dependent applications. Two extra interface functions makes it possible to read all the keys (also non-green keys):

- The player shell can enable this extension by calling the function ' PS_INIT '. After this, all keys (green or not green) are pushed into the buffer of the key driver.

- Calling PS_DEINIT disables this extension : only green keys are pushed into the buffer (cfr. system independent applications).

Remark :

- After calling PS_DEINIZ, it's possible that non-green keys remained in the buffer as the driver didn't clear the buffer.

- Running player shell simultaneous with an other application will cause unexpected effects: When the player shell calls PS_INIZ, the application reads also non-green keys as they are pushed into the key driver buffer.

3.3 KEYBOARD DRIVER

The function keys of the keyboard have the following meaning:

Key:	Function:	Code:
F1	Play	80
F2	Stop	81
F3	Pause	82
F4	Next Track	83
F5	Previous Track	84
F6	Search backwards	85
F7	Search forwards	86

3.4 APPENDIX

The following 2 functions are available for system dependent applications:

PS_INIT - Non-green keys allowed

PS_INIT causes that non-green keys are pushed into the buffer. Only system depencent applications (e.g. player shell) are allowed to make this call.

Input: d0.w = Path number
d1.w = SS_KB setstatcode
d2.w = PS_INIT

Output: None

PS_DEINIT - Non-green keys not allowed

PS_DEINIT causes that non-green keys are not pushed into the buffer. Only system depencent applications (e.g. player shell are allowed to make this call.

Input: d0.w = Path number
d1.w = SS_KB setstatcode
d2.w = PS_DEINIT

Output: None

APPENDIX B

The following list serves, together with the program **pmod**, as an identifier mechanism for version 1.1 of the CD-I libraries and include files, as well as version 3.2 of the Microware cross compiler libraries. This list is merely a summary of the output of **pmod** run on all the appropriate files. Each entry in the list shows the size of the file, the calculated cyclic redundancy check (CRC) for the file, and the name of the file, all separated by colons. The program **pmod** is available for Sun, Macintosh, and OS9. (The difference between the Sun include files and the OS9/Macintosh include files is due only to the different end of line characters on the operating systems.)

Size:CRC:	Name
Library files	
: 17297:F40163:	:cdi.l
:2253:EA824E:	:cdisys.l
:4526:3E71DF:	:cio.l
: 39927:12F134:	:clib.l
: 46599:A7A537:	:clib020.l
: 46545:95D6B0:	:clib020h.l
: 46453:60C81D:	:clib020n.l
: 39773:D9B2A0:	:clibn.l
:1274:5B2776:	:cstart.r
: 10975:B4F634	:math.l
:4858:D0FA74:	:math881.l
: 19657: 7EC58:	:sys.l
:3184:C745B7:	:termlib.l
: 14081:408867.	:usr.l
Macintosh/OS9 include files	
:4013:4143E9:	:cdfm.h
:7385:16D5A8:	:cdi.h
: 746:487845:	:csd.h
: 902:55178F:	:ctype.h
: 728:EB9F32:	:dir.h
:1444:E5F2D6:	:direct.h
:7110:E14479:	:errno.h
: 729: 59DF6:	:events.h
: 826:513EBC.	:float.h
: 483: 59636:	:limits.h
: 308:10E4D1:	:math.h
: 123:DAAB4D:	:memory.h
: 577:C71987:	:modes.h
:5764:75BB53:	:module.h
:1702:984D4C:	:path.h
:3929:5E87D3:	:procid.h
:9465:145FF7:	:rbf.h
:2321:E4CB71:	:sbf.h
:2662:5063EC:	:scf.h
: 140:34355E:	:setjmp.h
:4776:3FC30F:	:setsys.h
:6259:E05607:	:sg_codes.h
:6956:22C797:	:sgstat.h
: 378:D4F06F:	:signal.h
:1822:BDDFA0:	:stdio.h

: 201:9303DA:	:strings.h
:1451:6FFA69:	:sysio.h
: 201:A2E12D:	:termcap.h
:1358:16587E:	:time.h
: 146:FC5872:	:types.h
: 16987:DDF555:	:ucm.h

Sun include files

4013:7A3B60	:cdfm.h
7385:F20663	:cdi.h
746:A8818E	:csd.h
902:BB1E23	:ctype.h
728:CF826B	:dir.h
1444:91DE97	:direct.h
7110:1A177A	:errno.h
729:C79BC3	:events.h
826:EB967C	:float.h
483:7FB029	:limits.h
308:BD7603	:math.h
123:CF85C8	:memory.h
577:25755E	:modes.h
5764:A4B3A3	:module.h
1702:4A3585	:path.h
3929:1027B1	:procid.h
9465:B85014	:rbf.h
2321:3330EF	:sbf.h
2662:C8B6FB	:scf.h
140:A014BF	:setjmp.h
4776:F8E445	:setsys.h
6259:1B45BC	:sg_codes.h
6956:9F8561	:sgstat.h
378:F1BC06	:signal.h
1822:D95D5A	:stdio.h
201:63F230	:strings.h
1451:EAD7DC	:sysio.h
201:18C204	:termcap.h
1358:F422D8	:time.h
146:A13A7E	:types.h
16987:65701C	:ucm.h