Wii Optical Disc Drive Guidelines

Version 1.46

Wii™ Optical Disc Drive Guidelines

Version 1.46 (For Nintendo GameCube™) + Revisions for Wii

Table of Contents

Figures

Revision History

Version	Revision Date	Description of Revisions
1.46	9/29/06	• Updates for Wii. For provisional supply of information. Plan to move to Disc Drive Library Programming Manual.
1.46	4/21/04 5/28/04	• Added table of error emulation methods for each development instrument. • Added Chapter 6 “Other Cautions” and 6.1 regarding the description of Nintendo GameCube optical media.
1.45	11/28/03	• Made the restriction in paragraph 5.3 a recommendation, that the total section size should be less than 4 megabytes.
1.44	2/5/03	• Changed the instructions for distinguishing discs.
	1/9/03	• Added a warning about problems with the boot program.
1.43	11/15/02	• Revised and added a note to paragraph 3.2.5 “Fatal Error”. • Rewrote paragraphs 3.6 “Error-handling During Audio Streaming” and 3.7 “How to Emulate Errors”. • Revised paragraph 4.7.5 “Secure Memory for FST”. • Revised paragraph 5.4 “Banner File for Game Disc Information and Displaying it in IPL Main Menu”.
1.42	8/09/02	• Replaced disc number with game version as an example of a wild card in paragraph 4.2. • Revised paragraph 4.5 to reflect updates to the AMCDDK. • Revised paragraph 3.7 to reflect added functionality in the optical disc emulator. • Revised message for switching Game Discs in paragraph 4.4 • Added note to paragraph 3.6. • Added explanation to note in paragraph 3.2.
1.41	3/29/02	• Changed recommended message for Disc Cover Open error.
1.4	2/20/02	• Added sections related to the multi-disc function
1.3	12/3/01	• Added paragraph 3.6.
1.2.2	8/31/01	• Added notes to paragraphs 2.2 and 2.4, • Revised paragraph 2.6
1.2.1	8/3/01	• Revised messages in paragraphs 2.2.4 and 2.2.5 • Added paragraph 2.3 • Changed Game Disc to song at end of paragraph 2.6 • Revised paragraph 3.1 • Added paragraph 3.5
1.01	7/13/01	• Released by NOA

1 Overview

This document offers guidelines when using the Wii optical disc drive to design games that are user-friendly with regard to Game Disc access. This document differs from the "Revolution Optical Disc Drive Library (DVD.pdf)" document, in that it is meant not only for the programmers, but also for the designers (who design the screen layout for error messages) and the like, and for the bug testers. For information that is more specific to programming, see this section in the *Revolution Optical Disc Drive Library (DVD)*.

2 About the Game Discs

2.1 Game Disc Structure

The structure of each Game Disc can be broadly divided into two components:

- The program part (the `.dol` file)
- The user file part

The dol file is the game program itself, converted from the elf file when *ndrun* is executed. Because the format of the elf file built by the program will differ slightly depending on the linker, the elf file is converted into a custom-format dol file for storage on the Game Disc. The dol file is loaded and executed by the IPL.

The user file part is the component that is treated as files by the program part of the Game Disc. It does not matter how these files are used (for data, for re-locatable modules, etc.).

2.2 Game Disc ID

Each Game Disc has a region for storing its unique ID. This unique ID is comprised of the following four elements:

- Game code

  Each game has its own specific code. The code is set by the NOA Lot Check Department.
- Company code

  Each publisher has their own specific code. This code is obtained through the NOA Lot Check Department for the US market and NOE Lot Check Department for the European market.
- Game Disc number

  Every Game Disc for each game is allocated a Game Disc number. The first Game Disc of a game is numbered "0" and subsequent Game Discs are numbered 1, 2, 3…
- Game version

  Every version of the game is allocated a game version. It does not matter whether the game has been released or not.

The Optical Disc Driver uses the Game Disc ID to determine whether the Game Disc is a correct Game Disc.

3 Guidelines for Error-handling

3.1 Error-Handling Policy with Optical Disc Drive Device Driver

The error-handling policy of our optical disc drive device driver is “polling”. No complicated error-handling routine is needed. All the game developer needs to do is to poll the “error type” and show the proper message on the screen.

3.2 Error Types that Developers Need to Resolve

There are four “error types” that developers need to resolve.

- ~~Disc Cover Open Error~~ (Wii has no disc cover)
- No Disc Error
- Wrong Disc Error
- Retry Error
- Fatal Error

A description of each error type follows.

Note: These errors are generated when the issued command is not processed normally for some reason. They will not occur unless a command is issued. For example, when no command has been issued, a No Disc error will not occur, even if a disc is ejected. Additionally, the developer does not need to inform the game player about these status changes when they do not generate errors. In other words, even if the game player ejects the disc during gameplay, you do not need to display this message until the Game Disc is accessed.

3.2.1 ~~Disc Cover Open Error~~ (Wii has no disc cover)

~~When the game player opens the Disc Cover of the optical disc drive, this error will occur. Display the following message instructing the user to close the Disc Cover:~~

~~“The Disc Cover is open. If you want to continue the game, please close the Disc Cover.”~~

3.2.2 No Disc Error

This error occurs when the optical disc drive cannot find a Wii Disc. Display the following message and instruct the user to insert the appropriate disc:

“Please insert the <Game Title> Game Disc.”

This error may also occur in the following situations:

- The Wii Disc is inserted upside down.
- A disc that is not a Wii Disc, for example if a 12cm CD, is inserted.
- An extremely dirty Wii Disc is inserted.

If the game has multiple Game Discs, please display some message like:

"Please insert <Game Title> Game Disc 1"

to clearly indicate which Game Disc must be inserted.

3.2.3 Wrong Disc Error

This error occurs when the wrong Game Disc has been placed in the drive. Display the following message and instruct the user to insert the correct Game Disc:

“Please insert the <Game Title> Game Disc.”

Please be explicit when specifying which Game Disc to insert, when there are several discs in a set. For example:

“Please insert <Game Title> Game Disc 1.”

3.2.4 Retry Error

This error occurs when there is dust, fingerprints, etc. on the Game Disc and it cannot be accessed. Display the following message instructing the user to read the Instruction Booklet. (The Instruction Booklet tells the user to wipe the surface of the Game Disc with a soft cloth and then restart the game.)

“The Game Disc could not be read. Please read the Wii operations manual for more information.”

If a disc is inserted after a disc has been ejected, the device driver automatically performs a retry.

3.2.5 Fatal Error

This error occurs when a problem is detected that may make it impossible for the optical disc drive to recover. Display the following message instructing the user to read the operations manual.

“An error has occurred. First press the Eject Button and eject the disc. Then turn the Wii console power off and refer to the Wii operations manual for further instructions.”

There may be a problem with the optical disc drive or the Game Disc, so you need to stop the game after displaying an error message.

Note: When there is a Fatal error, we recommend that you disable RESET during the presentation of the error message. (Give priority to the display of Fatal error message).

3.3 Changing Error Messages

The error messages in Chapter 3.2 “Error Types that Developers Need to Resolve” are just examples of these messages. You may revise these messages, as long as the meaning is accurately conveyed to the user.

However, messages like Retry error and Fatal error reference the Wii operations manual for procedures to handle their associated errors. In order to avoid user confusion, do not make any major alterations to these messages.

3.4 How to Get the Error Type

This section explains how an application can retrieve the optical disc error type.

The error type shows the device driver's status. The device driver status can be obtained by calling the function `DVDGetDriveStatus`.

Note: As mentioned in Chapter 3.2 "Error Types that Developers Need to Resolve", errors only occur when commands cannot be processed properly. So, no errors will occur if no commands have been executed. For example, even if the disc is ejected, the function `DVDGetDriveStatus` will not return `DVD_STATE_NO_DISK` (described below) if there is no command being executed. The functions `DVDGetFileInfoStatus` and `DVDGetCommandBlockStatus` behave in the same manner.

The following list indicates states the function will return when an error has occurred.

- ~~`DVD_STATE_COVER_OPEN`~~ `(Wii has no disc cover)`
- `DVD_STATE_NO_DISK`
- `DVD_STATE_WRONG_DISK`
- `DVD_STATE_RETRY`
- `DVD_STATE_FATAL_ERROR`

There are other states that can be returned from the function `DVDGetDriveStatus()`. They are not mentioned here because they do not indicate errors, and this document focuses on error handling. Refer to the *Revolution SDK Function Reference Manual* for details on other states.

Note: For a game that has multiple Game Discs, DVD_STATE_MOTOR_STOPPED is returned when Game Discs are exchanged. See Chapter 4 "Games with Multiple Game Discs" for details.

Below is pseudo-code for handling errors. For an operational demo, see the DVD sample demo, "errorhandling" in the Revolution SDK.

Code 1 - Pseudo-code for Error-handling

```
DVDReadAsync();
do{
    status = DVDGetDriveStatus();
    switch (status){
      case DVD_STATE_NO_DISK:
        show_message_for_no_disk_on_screen();
        break;
      case DVD_STATE_WRONG_DISK:
        show_message_for_wrong_disk_on_screen();
        break;
      case DVD_STATE_RETRY:
        show_message_for_retry_on_screen();
        break;
      case DVD_STATE_FATAL_ERROR:
        show_message_for_fatal_error_on_screen();
        break;
    }
} while (status != DVD_STATE_END) && (status != DVD_STATE_FATAL_ERROR);
```

In order to display an appropriate error message when a problem occurs, this routine calls the function `DVDGetDriveStatus` every frame and polls the state.

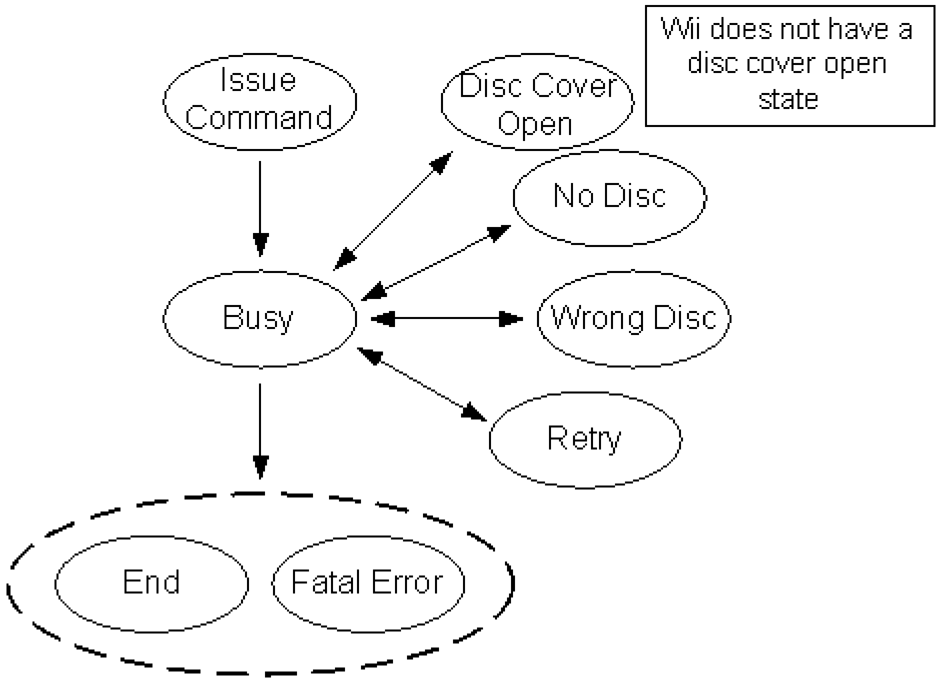
The APIs that access the optical disc drive only return when processing is completed or if a fatal error occurs. “Return” here means that **callback** is called (**callback** is specified with an asynchronous function), or API ends processing and returns a value (with a synchronous function).

You can also use the function `DVDGetFileInfoStatus` or `DVDGetCommandBlockStatus` to get the error state. These return the states of the designated file info and command block. The function `DVDGet-DriveStatus` returns the state of the currently executing request.

3.5 State Diagram

The following diagram shows how the state (error type) changes.

Figure 1 - State Changes



An explanation of Figure 1 follows:

- From {No Disc, Wrong Disc, Retry} to Busy

  This state change occurs when the disc is inserted. Note that the state does not change when the disc is ejected.

- ~~High-speed Change of Status due to Interrupt~~ (Wii does not have one)

  ~~For example, if the Disc Cover is closed, the interrupt will take place and be processed internally. The device driver changes states so polling is not required, making the processing minimal.~~

- Calibration Time when disc is inserted

  Please note that when the disc is inserted, the calibration processing of the optical disc drive takes time. For example, if in the `WRONG_DISK` state, it takes time to change discs and then a driver checks the Disc ID.

3.6 ~~Error-handling During Audio Streaming~~ (Not required for Wii)

~~Use the same procedure that is described in Chapter 3.4 “How to Get the Error Type”. Note that with this procedure, there are times when an error cannot be detected. If no error can be detected, there is no need to use some other method in search for one.~~

3.6.1 ~~Error-handling During Preparation for Audio Streaming~~ (Not required for Wii)

~~Before the start of audio streaming, preparations must be made by executing the `DVDPrepareStream` function. If an error occurs during this preparation process, the error can be detected by using the procedure described in Chapter 3.4. That is to say, by calling the `DVDGetDriveStatus` function.~~

~~When an error is detected at this time, it is essential to notify the game player.~~

3.6.2 ~~Error-handling After Starting Audio Streaming~~ (Not required for Wii)

~~The `DVDGetDriveStatus` function cannot detect errors that occur after starting audio streaming. But there is no need to use some other method to detect errors and no need to notify the game player if one is detected.~~

3.6.3 ~~Restarting Audio Streaming After it has Been Halted Due to an Error~~ (Not required for Wii)

~~If an error occurs during audio streaming the audio stream will halt, and it will not automatically start again, even if the cause of the error has been removed. The developer is free to decide whether in such cases to execute the `DVDPrepareStream` function again and restart the audio stream.~~

~~Consider the situation where the Disc Cover is opened while an audio stream is executing. When that happens the audio stream will halt. It is up to the developer to decide whether or not the audio stream should start up again when the Disc Cover is closed.~~

3.7 How to Emulate Errors

Developers must check that appropriate error-handling routines (e.g., displaying appropriate error messages on the TV) are executed in their program. All of the different kinds of errors (Wrong Disc, Retry, and Fatal Error) can be emulated using NDEV. For details see the NDEV manual.

~~DDH~~

~~All of the errors can be emulated using AMCDDK Version.3.0.2 patch #2 or later. For details, read the AMCDDK manual.~~

~~NPDP-GDEV~~

~~All of the errors can be emulated using Version 1.01 or later. For details, read the NPDP-GDEV manual.~~

~~If you will be using the NPDP-GDEV with an inserted NPDP cartridge, read the section below concerning NPDP.~~

~~**NPDP**~~

~~All of the errors can be emulated using Version 1.0.003 or later firmware. For Retry Error and Fatal Error, especially, refer to the table below:~~

Table 1 - Error Emulation Method for Each Development Instrument

	Retry Error	**Fatal Error (Note 1)**	**Wrong Disk Error**	**No Disk Error**
NDEV~~NPDP GDEV~~	OdemSry (RETRY button)	OdemSry (FATAL button)	~~OdemSry (Cover Open)~~ OdemSry (Select DLF File) ~~OdemSry (Cover Close)~~ OdemSry (Eject) OdemSry (Load)	~~OdemSry (Cover Open)~~ OdemSry (No Disk) ~~OdemSry (Cover Close)~~ OdemSry (Eject) OdemSry (Load)
~~NPDP with NPDP GBOX~~	~~Error Button (Note 1)~~	~~UP Button + Error Button~~ ~~(NPDP firmware v 1.0.003 and later (Note 1)) (Note 3)~~	~~Open/Close Button~~ ~~UP Button/DOWN Button~~ ~~Open/Close Button~~	~~Open/Close Button~~ ~~UP Button/DOWN Button (Select No Disk)~~ ~~Open/Close Button~~
~~NPDP with NPDP Console~~	~~Error Button (Note 1)~~	~~Disk Change Button + Error Button~~ ~~(NPDP firmware v 1.0.003 and later (Note 1)) (Note 3)~~	~~Open/Close Button~~ ~~Disk Change Button~~ ~~Open/Close Button~~	~~Open/Close Button~~ ~~Disk Change Button (Select No disk)~~ ~~Open/Close Button~~
RVT-R Reader ~~NR Reader~~	None	None	Exchange disk (Note 2)	Remove disk
~~DDH~~	~~>dirtydisk~~	~~>breakdrive~~	~~>loadrun myapp.elf~~ ~~>opencover~~ ~~>changedisk myapp.elf editor "-dn=1"~~ ~~>closecover~~	~~>loadrun myapp.elf~~ ~~>opencover~~ ~~>removedisk~~ ~~>closecover~~

Note 1: Error appears when you hold down the button until the next disc access.

Note 2: Use different disc IDs for the disc after exchange and the disc before exchange.

~~**Note 3:** Occasionally a Retry error will be generated if buttons are not pressed in this order:~~

~~Press and hold down UP button or Disk Change button, then press Error button.~~

4 Games with Multiple Game Discs (For Reference)

Please contact the NOA Licensing Department in advance, when considering games with several Game Discs.

At the present time, such discussions are only for two-disc games, for which both discs are sold at the same time. Please hold separate discussions with the Nintendo Licensing Department for games with three or more Game Discs, or for the marketing of separate Game Discs.

If more than one Game Disc is needed for a single game, then the Game Discs must be exchanged at some appropriate place in the game. The exchange is done by specifying the Game Disc ID for the next Game Disc (see Chapter 2.2 “Game Disc ID”).

The rest of this chapter explains the different types of exchange patterns for multi-disc games and how to specify Game Discs. It provides a broad description of the procedure for exchanging Game Discs, and gives an example of a message displayed to instruct the player to exchange Game Discs.

4.1 Game Disc exchange Process Patterns

With multi-disc games, the Game Disc exchange process can be broadly classified into two patterns, depending on whether or not the dol program part (Chapter 2.1 “Game Disc structure”) of the newly inserted Game Disc is loaded and executed after Game Discs are exchanged. As an example, consider the case where Game Disc 1 is removed and replaced with Game Disc 2.

- **A When loading/running a Game Disc 2 dol file**

  After the switch from Game Disc 1 to Game Disc 2 is complete, please run the restart process. Restarting will start the dol program on Game Disc 2.

- **B When not loading/running a Game Disc 2 dol file**

  After the switch from Game Disc 1 to Game Disc 2 is complete, it will be possible to access the Game Disc 2 user files (Chapter 2.1 “Game Disc structure”). There is nothing particular that has to be done after the switch.

  In this case, the system can be classified in two ways: to allow Game Disc 2 to start up by itself or not.

  - **B1 When allowing individual start up by Game Disc 2**

    It is conceivable that the programs on Game Disc 1 and Game Disc 2 could be the same, and only the user file sections would be different. There may be a prompt to return to Game Disc 1, depending upon the state of progress of the game in the Memory Card.

  - **B2 When not allowing individual start up by Game Disc 2**

    It is conceivable that Game Disc 2 could contain only data files. In this case, please display a message like "This is a data only Game Disc. Please insert <Game Title> Game Disc 1.", and give the user appropriate instructions when in individual start-up mode.

We don't really recommend B2 above. That's because it is possible that there could be Game Disc switching all the time, when a game is being run. Unless there is a particular reason for needing a data-only Game Disc, we feel it is nicer to the user to have the same dol file on the second Game Disc as on the first Game Disc, and make it possible to start-up on the second Game Disc by itself.

Comparing A and B1, B1 requires no loading of the dol program section on Game Disc 2, and the shift to the second Game Disc will be somewhat faster. If the dol program portion of the first and second Game Discs can be made the same, it would probably be better to choose B1.

Note that in either case, we recommend making it possible to save data before switching Game Discs, for the following reasons.

- Data loss can be avoided if there is a Game Disc read error after the switch.
- It will make debugging easier by saving the data to the Memory Card before exchanging the Game Discs. When you have a bug either during or after the Game Disc exchange, you can easily reproduce the bug using the saved data. If you don't save the data before the Game Disc exchange, you will have to begin game play from the Game Disc 1 again, to reproduce the bug.

Caution

File structures that were open prior to the exchange will contain old data, so please do not use such data after the exchange. Even if Game Disc 1 and Game Disc 2 have files with the same name, please be sure to re-open them. To be safe, we recommend closing all open files before switching.

4.2 Specifying Game Discs at Time of Exchange

The Game Disc ID (see Chapter 2.2 “Game Disc ID”) is used for specifying Game Discs at the time of an exchange. For details, refer to the DVDChangeDisk* function in the *Revolution SDK Function Reference Manual*.

With the `DVDChangeDisk*` function, wildcards can be specified for Game Disc ID components. Thus, if 0xff is specified for the Game Version, then no matter which version of the game is inserted during the exchange, it will be treated as a correct Game Disc and the exchange will be approved, as long as the other three Game Disc ID components (i.e., the Game Code, Company Code and Disc Number) are correct.

In the following section, we offer some precautionary statements concerning wild cards.

4.2.1 Wild Card Usage Restrictions (Non-Game Versions)

Wild cards can only be used on game versions (not on Game Codes, Company Codes or Game Disc numbers). The reason is that when wild cards are used, the operation check combinations become enormous and cause a lot of problems.

If you are nevertheless considering the use of wild cards for the elements other than game version, you must discuss the matter with Nintendo in advance.

4.2.2 Wild Card Usage Restrictions (Game Versions)

When using wild cards for game versions, there are advantages and disadvantages to using and not using them (when given explicit instructions). The following are some examples. Please review them and select one or the other for use.

A When the Game Version is Explicitly Specified

- Advantages

  The corresponding Game Disc versions have a one-to-one correspondence, so it is sufficient to debug in that particular combination.
- Disadvantages

  When there are future version changes, it will be necessary to upgrade the version on both Game Discs. In other words, if Game Disc 1 specifies Game Disc 2 version 0, and Game Disc 2 was modified to version 1 after it was sold, then once Game Disc 1 had been modified, the versions on both Game Discs would have to be upgraded and resubmitted.

When a group of friends have their own copies of the same game and the versions are different and two friends switch Game Discs, they will not work properly when Game Disc 1 and Game Disc 2 are exchanged.

An identifying mark will be required on the label so that the version can be identified.

B When a Wild Card is Used in a Game Version

- Advantages

  Independent version upgrades can be released at future dates on their own Game Discs.

  Game Discs with different versions will keep running when they are used together, so friends can switch Game Discs without any problems.

- Disadvantages

  Care is required when releasing future version upgrades. In other words, when upgrading for example, Game Disc 2, it will be necessary to assure that it can be switched with all of the Game Disc 1 versions that are out circulating in the market.

4.3 The Game Disc Exchange Procedure

Below is an outline of the procedure that takes place from the time the player exchanges Game Discs until the Wii can access the new Game Disc.

1. The Optical Disc Drive motor is stopped.
2. The optical disc drive motor is confirmed to be stopped, and a message is displayed telling the game player to exchange Game Discs (see Chapter 4.4 “Messages and Error Processing when Switching Game Discs”).
3. ~~The system waits until the Disc Cover is opened by the player and then closed.~~ (Wii has no disc cover)
4. After the exchange, the Game Disc ID is checked to determine whether it is the proper Game Disc.
5. The Game Disc's File Symbol Table (FST) is loaded and access is enabled.

All of these procedures, except for the message display performed in (2), are processed by the DVD-ChangeDisk function (See Chapter 4.6 “APIs for multi-disc games”).

You can determine the duration of the message displayed in Step 2, by using the DVDGetDriveStatus function to check the status of the Device Driver. You check the status of the Device Driver using the same method you use for error handling (see Chapter 3.4 “How to get the error type”). When the optical disc drive motor is stopped, the optical disc driver's status is DVD_STATE_MOTOR_STOPPED. Only display the message in Step 2 when the optical disc driver is in this state.

If the exchanged Game Disc is not the proper Game Disc, or if there is no Game Disc in the optical disc drive, then display an appropriate message like those shown in Chapter 4.4 “Messages and Error Processing when Switching Game Discs”. The DVDChangeDisk* function can cancel the Game Disc exchange process by calling the DVDCancel function.

4.4 Messages and Error Processing when Switching Game Discs

This section provides an example of a message to be displayed to the game player when Game Discs are being exchanged. These sample displays are like the ones in Chapter 3.3 "Changing Error Messages," and they may be modified to communicate their intent to the game player.

- Instructing the player to exchange Game Discs

  While the Device Driver is in the DVD_STATE_MOTOR_STOPPED state, display this message, instructing the game player to insert the next appropriate Game Disc:

  "Please insert <Game Title> Game Disc 2."

If the game is a multi-disc game, be sure to provide clear instructions about which Game Disc should be inserted next.

When you issue the DVDChangeDisk function, perform the same error handling that you would with the DVDRead function. For details, see Chapter 3.2 “Error Types that Developers Need to Resolve” (If the specified Game Disc is not inserted, then a Wrong Disk error should be generated.)

4.5 Multi-disc Emulation

Currenly, multi-disc emulation can be only be executed on NDEV.

4.6 APIs for Multi-disc Games

The following APIs apply to multi-disc games. For details about each API refer to the *Revolution SDK Function Reference Manual*.

- DVDChangeDisk* (DVDChangeDisk as well as DVDChangeDiskAsync)
- DVDCompareDiskID
- DVDGenerateDiskID
- DVDGetCurrentDiskID

4.7 Cautions Regarding Multi-disc Games

4.7.1 Every Game Disc Should Boot on its Own

(Same content as Wii Programming Guideline, Version 1.00, paragraph 3.3 “[Only For Multiple Disc Games] Independent Startup of All Discs [Required].”)

Even if startup of the application from a particular Wii Game Disc is not expected, always make sure that some kind of program is executed when the disc is started independently, even if it is just a warning message. For example, if the inserted Wii Game Disc is one of a set of two Wii Game Discs and the inserted disc is one that cannot be played standalone, have available a program that displays a message such as “This is <*game title*> Game Disc 2. Please insert <*game title*> Game Disc 1 and restart power.”

Not doing so may cause the user to think that there is malfunction in either the Wii console or the Wii Game Disc when a Wii Game Disc is not designed for independent startup.

~~Please design every Game Disc so it can boot up on its own when inserted in the Nintendo GameCube. Have the Game Disc boot on its own -- even if the game's execution program cannot start up without the execution program that is on some other Game Disc -- and then display a message instructing the game player to exchange Game Discs (see Chapter 4.7.3 “Display for Game Disc exchanges”)~~

4.7.2 Provide Support for the Exchange with any Game Disc

Make sure the system does not lock-up, no matter what Game Disc is exchanged. For example, if Game Disc 3 is only supposed to operate when it is booted after Game Disc 2, do not let the system lock-up if it is exchanged after Game Disc 1 instead.

4.7.3 Display for Game Disc Exchanges

If an exchange of discs is necessary, please display instructions to this effect on the screen. See Section 4.4 “Messages and Error Processing when Switching Game Discs” to read about the contents of this screen message.

4.7.4 Use Labels and Banners to Distinguish Game Discs

Print a label for each Game Disc, so game players can differentiate Game Discs and exchange them properly when instructed to do so. For details on banners and icons displayed by the Wii console, see "Icon and Banner Creation Tool."

4.7.5 Secure Memory for FST

When a normal Game Disc is exchanged, the File Symbol Table (FST) is overwritten by what is on the new Game Disc. If the FST of the new Game Disc is smaller than the FST of the Game Disc that was swapped out, then there is no problem. But if the new FST is larger, then it cannot be loaded. An ASSERT message will be output, and the system will lock up.

To prevent this from happening, with a game that has multiple discs, be sure to secure a memory space of adequate size for the FST. When using DDH, request the required FST size from each disc and specify each value in `FstMaxLen`. When using NPDP-GDEV, register the group of discs in the `mdf` file. When this is done, the appropriate FST size will be set automatically.

For further details, read the various development environment manuals.

5 Other Guidelines Regarding Accessing the Optical Disc Drive

5.1 Game Startup Screen

After the dol file is loaded by the boot program, data and program files are read from the Game Disc. At this time, in order to shorten the time the screen is blank as much as possible, do not read any data files. Use only the dol file to display the first screen.

Also, the dol file should use the error-handling routines described in Chapter 3.4 "How to Get the Error Type". If there are no error-handling routines (including necessary data) built into the dol file, there is no way to notify the user if some kind of problem occurs with the read before the error-handling routine's data is loaded.

~~Warning:~~ (Not a concern with Wii)

~~There is a bug in the boot program that causes the system to halt under certain circumstances when the game program is booted from the Main Menu screen. Specifically, if the Disc Cover is opened just at the right moment while the screen fades out, the screen will remain faded out and the system will halt. The timing has to be just right for this to happen, so the rate of occurrence is very low. However, you should be aware of this when testing the error handling of your game program. It is acceptable to ignore this bug.~~

~~Take the following steps if a problem arises and you want to determine whether it is due to this bug or to some bug in the game program:~~

1. ~~Try closing the Disc Cover. If the game program boots up, the problem was caused by a bug in the game program.~~
2. ~~Wait 10 seconds after closing the Disc Cover and then press the START button. If the game program boots up, the problem was caused by the boot program bug. (This can be ignored.)~~
3. ~~If the game program does not start up when you perform Step 2, then the problem was caused by a bug in the game program.~~

~~This boot program bug does not occur when games are started from the Nintendo GameCube™ logo (in other words, booted up without going through the IPL Main Menu screen).~~

5.2 Various Causes of Changes in Game Disc Access Time

(Same as Wii Programming Guideline, Version 1.00, paragraph 2.1 "Prohibition of Dependency on Device Specifications [Information]. Plan to add "Prohibition of Dependency on Device Specifications) [Required]" to the disc drive chapter, as a disc drive related topic.)

Avoid designing a program with a dependency on the device specifications of the Wii console or its peripherals, since this may result in a program malfunction or crash.

There will be some specification variance between devices. Performance will also decrease as the device ages or a disc becomes damaged.

Specifically, the devices here refer to the disc drive, disc, Wii Remote, Wii system memory, SD Memory Card, and Nintendo GameCube Memory Card. See the corresponding chapters for details.

~~If a problem occurs with Game Disc access, the hardware will carry out a retry internally. This means that even if a read has succeeded, it may have been the result of going through several retries.~~

~~Also, the calibration process requires a large amount of time, as mentioned in Chapter 3.5 "State Diagram". Therefore, when a game player opens the Disc Cover while reading, much longer time is required than just one read.~~

~~Furthermore, due to individual differences, the access may be completed slightly faster than with the particular optical disc drive the developer is using.~~

~~Because Game Disc access time can vary for all of these various reasons, do not design processes that depend on access time.~~

5.3 dol File Restrictions

Be aware that following restriction applies to the dol file (see Chapter 2.1 “Game Disc structure”).

- Addresses that can be used by dol files are below 0x8070_0000

  Addresses from 0x8070_0000 and on are reserved by the system so dol files can only use addresses below this. Note that this restriction also applies to BSS section.

In addition, we recommend that the total file size, excluding the BSS and SBSS sections (data region for uninitialized variables) be less than 4 megabytes. This is recommended to reduce the read time when starting up.

5.4 ~~Banner File for Game Disc Information and Displaying it in IPL Main Menu~~ (Wii displays separate icons and banners)

~~On the Nintendo GameCube, when the game is not started immediately and is started after going into the IPL Main Menu once, disc information is displayed on the screen. Banner files are used in this situation.~~

~~In order to display disc information, the following information must be specified in the banner file:~~

~~`u8 image[2 * DVD_BANNER_WIDTH * DVD_BANNER_HEIGHT]`~~
~~`(DVD_BANNER_WIDTH = 96; DVD_BANNER_HEIGHT=32)`~~

> ~~Banner format image data.~~
>
> ~~The size is 96x32 (the same as the memory card banner), but the only format that can be used is RGB5A3.~~
>
> ~~Identical data is used on the menu screen and on the gameplay screen.~~
>
> ~~There is no problem if the data does not match the memory card banner (although you can use the same data).~~

~~`u8 shortTitle[32]`~~

> ~~The title of the game displayed on the Main Menu screen.~~

~~`u8 shortMaker[32]`~~

> ~~The name of the maker displayed on the Main Menu screen.~~
>
> ~~You can include just the maker name, or the maker name and the production date (the year).~~

~~`u8 longTitle[64]`~~

> ~~The title of the game displayed during gameplay.~~

~~`u8 longMaker[64]`~~

> ~~The name of the maker displayed on the gameplay screen.~~
>
> ~~You can include just the maker name, or the maker name and the production date (the year).~~

~~`u8 comment[128]`~~

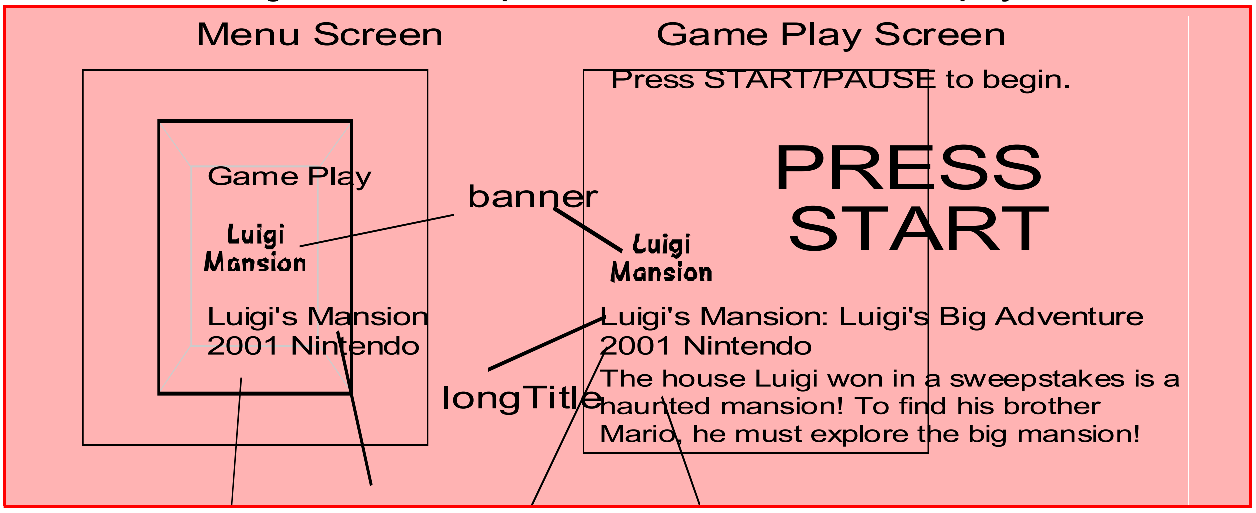
> ~~Comments displayed on the game play screen.~~

~~Example: Luigi’s Mansion~~

~~`shortTitle`~~	~~Luigi’s Mansion~~
~~`shortMaker`~~	~~2001 Nintendo~~
~~`longTitle`~~	~~Luigi’s Mansion: Luigi’s Big Adventure~~

~~longMaker~~	~~2001 Nintendo~~
~~comment~~	~~The house Luigi won in a sweepstakes is a haunted mansion!~~
	~~To find his brother Mario, he must explore the big mansion!~~

~~**Figure 2 - An example of Game Disc information display**~~


Menu Screen
Game Play Screen
Game Play
Luigi Mansion
Luigi's Mansion
2001 Nintendo
banner
longTitle
Press START/PAUSE to begin.
PRESS START
Luigi Mansion
Luigi's Mansion: Luigi's Big Adventure
2001 Nintendo
The house Luigi won in a sweepstakes is a haunted mansion! To find his brother Mario, he must explore the big mansion!

~~The characters that can be used are the same as those that can be used by the FONT API. Game Discs intended for the Japanese market can accommodate Level 1 Shift-JIS characters as well as ASCII and single-byte ("hankaku") katakana characters. Game Discs intended for the non-Japanese market can accommodate ANSI 8-bit (WinLatin1) characters. Line feed codes can also be used in comments. However, if you use many line breaks or characters with wide character widths, there is a possibility that the comment will not be displayed completely within the frame. In such cases, you would need to revise your text to make it fit completely within the frame.~~

~~On development tools, the Nintendo GameCube startup program displays the information in the `opening.bnr` file stored in the Game Disc's root directory. You can create `opening.bnr` by executing the `makebanner.exe` tool or the `makebanner2.exe` tool (specific to Europe) included with the Nintendo GameCube SDK in the `/X86/bin` directory.~~

~~`makebanner.exe` creates banner files in the BNR1 format, which can be used in any region. `makebanner2.exe` creates files in the BNR2 format, which can include game information in six standard languages (English, German, French, Spanish, Italian and Dutch). Do not use `makebanner2.exe` for game discs destined for Japan or the U.S. The reason is because the IPLs of the Japanese and American versions of the Nintendo GameCube do not support the BNR2 format. On the other hand, we recommend that you use `makebanner2.exe` for Game Discs destined for Europe.~~

~~The banner file structure is defined in `/dolphin/dvd/DVDBanner.h`.~~

~~You can use the listdemo program (located in `build/demos/carddemo`) to check the Game Disc banner and game information located in the `opening.bnr` file.~~

5.5 Avoid Infinite Loop of Hardware/Audio Streaming Under the Following Conditions

~~(~~Same as Wii Programming Guideline, Version 1.00, paragraph 3.2 "Infinite Loops During Multiple Data Streaming Prohibited [Required]". Wii does not perform hardware audio streaming.)

Avoid infinite looping of simultaneous multiple instances of data streaming.

Data streaming is defined as the method of reading data sequentially as it is used, as opposed to a burst transfer where necessary data is loaded all at once from the disc before it is used. One good example of data streaming is movie playback.

Similarly, if frequent non-sequential access is necessary over a long period of time, try to make the access sequential or, in any case, prevent an infinite loop.

Also, seek noise created during simultaneous playback can be reduced significantly by placing files that are played simultaneously as close to each other as possible.

When the drive enters an infinite loop while constantly moving the drive head at high frequency, this may adversely affect the drive if the user leaves the game in this state.

If the application violates this requirement, the following steps may be taken to prevent such an occurrence.

- Interleave the data file and the audio file into a single file and play back the audio portion.
- Interleave multiple files for data streaming into a single file so that they can be accessed sequentially.

Another method to prevent infinite loops is to stop data streaming after several loops.

~~Avoid an infinite loop in the following situations:~~

~~1. Hardware/Audio Streaming while Data Streaming~~

~~2. Synchronous Playback of more than one data stream~~

~~**Note:** The necessary data is read from the Game Disc all at once and then used as required. This method is defined as Burst Transfer. The data is read when required, one by one in order, and is not read all at once. This method is defined as Data Streaming. A typical example of data streaming transfer would be the playback of a movie.~~

~~The reason that we need to avoid infinite loops is because the player may leave the Nintendo GameCube unattended, and frequent drive accesses may be occurring. Nintendo GameCube's optical disc drive moves the head frequently when accessing the Game Disc. There is a possibility that this may cause unnecessary wear and tear on the optical disc drive. Games should avoid this situation.~~

~~Other than "1" and "2" above, if there is a need for prolonged access that is not sequential with high frequency, either make the access sequential or prevent it from entering an infinite loop.~~

~~While the optical disc drive is seeking data, a noise is generated that some users may find objectionable. To limit this noise, reduce the number of seeks that are generated during synchronous playback considerably by placing those files that are played back synchronously close to each other. Place these files as close to each other as possible.~~

5.5.1 Examples of Problems and How to Solve Them

1. There is a demo that plays different audio files as BGM while playing back a movie file. It is possible that this demo will enter an infinite loop.

   Solution 1: Compile the movie file and the audio file into one file by interleaving them, and play back the audio portion using software. After this is done, access to the optical disc drive will be sequential.

   Solution 2: Stop BGM after the fade-out of the third loop. By doing this, only the movie is played back, and access to the optical disc drive will be sequential.

2. If two movies are played back synchronously there is a possibility that the program will enter an infinite loop.

   Solution 1: Compile the two movie files into one file by interleaving them. By doing this, access to the optical disc drive will be sequential.

   Solution 2: Switch both movies to a static image after three loops. This will prevent an infinite loop.

3. There is an artificial intelligence type dialog game. There is a portion of this game in which voice data is read consecutively, while the audio file is played as BGM. This might continue infinitely, unless there is an input to the controller.

   Solution: If there is no input to the controller for 5 minutes, make characters in the game go to sleep, so that they will no longer talk. By doing this, access to the optical disc drive will be only for playing back BGM.

6 Other Cautions

6.1 Describing Wii Optical Media

When describing Wii optical media to a game player, such as in a game message or in the Instruction Booklet, please use “Game Disc”, “Disc”, or “disc” instead of “Disk”.