SD Specifications
Part E7
Wireless LAN Simplified Addendum

Version 1.10

March 25, 2014

Addendum to:

SD Specifications
Part E7 iSDIO Simplified Specification
Version 1.10
March 25, 2014

Technical Committee
SD Card Association

Revision History

Date	Version	Changes compared to previous issue
March 25, 2014	1.10	The first release of iSDIO Wireless LAN Simplifed Addendum Following use cases are defined for wireless LAN applications upon an iSDIO Card. - Server Upload - Peer-to-Peer File Transfer - DLNA - PTP

To the extent this proposed specification, which is being submitted for review under the IP Policy, implements, incorporates by reference or refers to any portion of versions 1.0 or 1.01 of the SD Specifications (including Parts 1 through 4), adoption of the proposed specification shall require Members utilizing the adopted specification to obtain the appropriate licenses from the SD-3C, LLC, as required for the utilization of those portion(s) of versions 1.0 or 1.01 of the SD Specifications.
For example, implementation of the SD Specifications in a host device under versions 1.0 or 1.01 and under the adopted specification requires the execution of a SD Host Ancillary License Agreement with the SD-3C, LLC; and implementation of the SD Specifications under versions 1.0 or 1.01 and under the proposed specification in a SD Card containing any memory storage capability (other than for storage of executable code for a controller or microprocessor within the SD Card) requires the execution of a SD Memory Card License Agreement with the SD-3C, LLC.

Release of SD Simplified Specification/Addendum

The following conditions apply to the release of the SD Simplified Specification/Addendum by the SD Card Association. The Simplified Specification/Addendum is a subset of the complete version of SD Specification/Addendum which is owned by the SD Card Association.

Conditions for publication

Publisher and Copyright Holder:

SD Card Association
2400 Camino Ramon, Suite 375
San Ramon, CA 94583 USA
Telephone: +1 (925) 275-6615,
Fax: +1 (925) 886-4870
E-mail: office@sdcard.org

Notes:

This Simplified Specification/Addendum is provided on a non-confidential basis subject to the disclaimers below. Any implementation of the Simplified Specification/Addendum may require a license from the SD Card Association or other third parties.

Disclaimers:

The information contained in the Simplified Specification/Addendum is presented only as a standard Specification/Addendum for SD Cards and SD Host/Ancillary products and is provided "AS-IS" without any representations or warranties of any kind. No responsibility is assumed by the SD Card Association for any damages, any infringements of patents or other right of the SD Card Association or any third parties, which may result from its use. No license is granted by implication, estoppel or otherwise under any patent or other rights of the SD Card Association or any third party. Nothing herein shall be construed as an obligation by the SD Card Association to disclose or distribute any technical information, know-how or other confidential information to any third party.

Conventions Used in This Document

Naming Conventions

- Some terms are capitalized to distinguish their definition from their common English meaning.
- Words not capitalized retain their common English meaning.

Numbers and Number Bases

- Hexadecimal numbers are written with a lower case "h" suffix, e.g., FFFFh and 80h.
- Binary numbers are written with a lower case "b" suffix (e.g., 10b).
- Binary numbers larger than four digits are written with a space dividing each group of four digits, as in 1000 0101 0010b.
- All other numbers are decimal.

Key Words

- May: Indicates flexibility of choice with no implied recommendation or requirement.
- Shall: Indicates a mandatory requirement. Designers shall implement such mandatory requirements to ensure interchangeability and to claim conformance with the specification.
- Should: Indicates a strong recommendation but not a mandatory requirement. Designers should give strong consideration to such recommendations, but there is still a choice in implementation.

Application Notes

Some sections of this document provide guidance to the host implementers as follows:

Application Note: This is an example of an application note.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

1.1 Overview

iSDIO Wireless LAN Card ("Card") is an Intelligent SDIO Combo Card that add a wireless LAN module to a NAND memory module (which may be removable as a micro SD card), and furthermore both modules in the Card are able to communicate with each other by simple instructions from a Host.
In one use case, a user records data with a Host that has a Card and then sends the data to other Hosts with the Card via a wireless LAN without an Access Point (AP). In another use case, a user sends data by a Card to a smart phone, a PC, or other network devices via wireless LAN. In a third use case, a user sends data by a Card to a server via Access Point.

1.2 Prerequisite Specifications of iSDIO Wireless LAN Card

Reference documents are described in Appendix A.
iSDIO Wireless LAN Card shall or may support following specifications.

- A Card shall support SD Memory commands defined in [SDPart1] and support either one or both of
  - Function Extension commands defined in [SDPart1] and
  - SDIO commands defined in [SDIO].

- A Card shall support the SD File System defined in [SDPart2].

- A Card shall support the Station mode of IEEE 802.11g defined in [IEEE802.11] at a minimum. In addition,
  - A Card shall support the WPS 2.0 defined in [WPS] to connect an AP.
  - A Card shall support WPA2 defined in [IEEE802.11] at a minimum for Wireless Network Security.

- A Card may support the Access Point mode of IEEE 802.11g defined in [IEEE802.11] at a minimum. In addition,
  - A Card shall support at WPA2 at a minimum as Wireless Network Security if a Card supports the Access Point mode.

- A Card may support the Wi-Fi Direct defined in [WiFiDirect].

- A Card shall support the ability to send and receive an HTTP message which complies with HTTP/1.1 defined in [HTTP] and [HTTPS] including the support of
  - TCP/IP as Network and Transport Layer defined in [TCP] and [IP], and
  - SSL 3.0/TLS 1.0 defined in [SSL] and [TLS] as Security Layer with the following cipher suite at a minimum.
    - "TLS_RSA_WITH_AES_128_CBC_SHA" (2048-bit and shorter RSA key are supported at a minimum)

1.3 Use Cases

This Wireless LAN Addendum Specification enables a Card to realize four typical use cases, Server Upload, DLNA, a Peer-to-Peer File Transfer and PTP.

1.3.1 Server Upload

This specification enables a Card to send an HTTP request message attaching data and to receive an HTTP response message via an AP.
This specification requires a Host to control a Card to upload a file to a Host-vendor's service or 3rd party service. A case that a Card uploads a file to a Card-vendor's service or 3rd party service "without" Host control is out of scope in this specification.

The followings are the basic steps for this use case, and the corresponding commands and status information are defined in this specification to realize the use case. Note that the basic steps are typical, and a Host implementer is able to realize this use case in other ways by using the commands and status information defined in this specification.

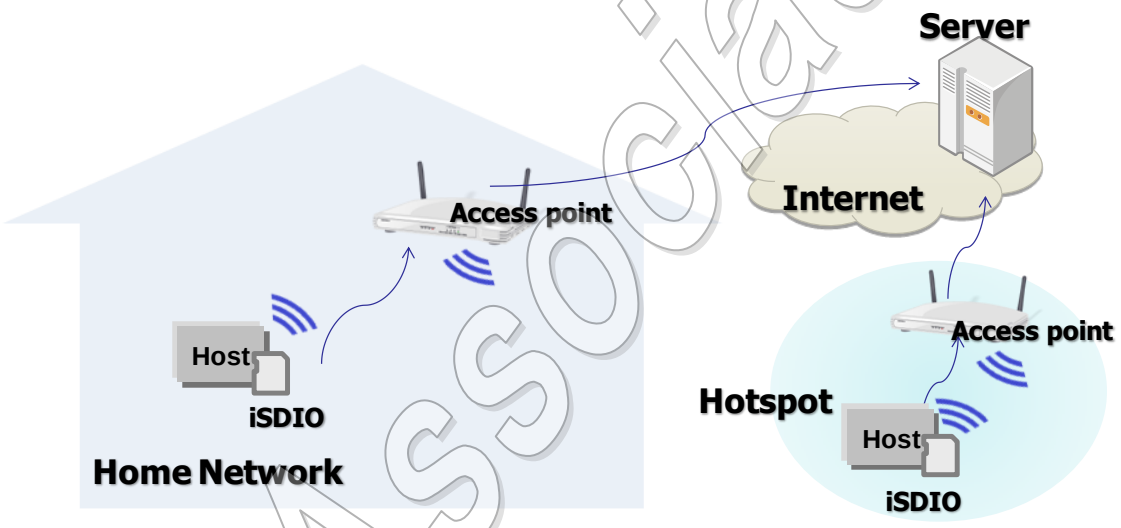


Figure 1-1 : Server Upload use case

Basic Steps

(1) A Card connects an AP.
(2) A Host creates an HTTP request message and optionally stores the message as a file into the Card.
(3) The Card sends the HTTP request message to the specified server. The Card may attach a file to the message.
(4) The Card receives an HTTP response message from the server and optionally stores the message as a file into the Card.
(5) The Host reads the HTTP response message from the Card.

1.3.2 DLNA

This specification enables a Card to support use cases defined by DLNA (Digital Living Network Alliance). A Card supports DLNA M-DMS (Mobile Digital Media Server) device class and M-DMU (Mobile Digital Media Uploader) device class. In addition, a Card supports STA or Wi-Fi Direct.

Since many digital consumer devices support DLNA, the use of DLNA will allow a Card to send and receive data from digital devices.
This use case is to communicate with other devices supporting network access. A Card can connect other devices through home network and/or Wi-Fi Direct and can upload data if they have a storage device like a PC, NAS (Network Attached Storage), or other digital devices supporting iSDIO Card.

The DLNA use cases have three transfer scenarios through two DLNA device classes and one DLNA device capability (M-DMS, M-DMU, +PU+). The first scenario is a content transfer from a Card to a Card or a DMS device. The second scenario is a content push playback from a Card to a DMR (Digital Media Renderer) device. And the third scenario is a content distribution from a Card to a DMP (Digital Media Player) device on demand.

The followings are the basic steps for the first scenario, and the corresponding commands and status information are defined in this specification to realize the use case. Note that the basic steps are typical, and a Host implementer is able to realize this use case in other ways by using the commands and status information defined in this specification.

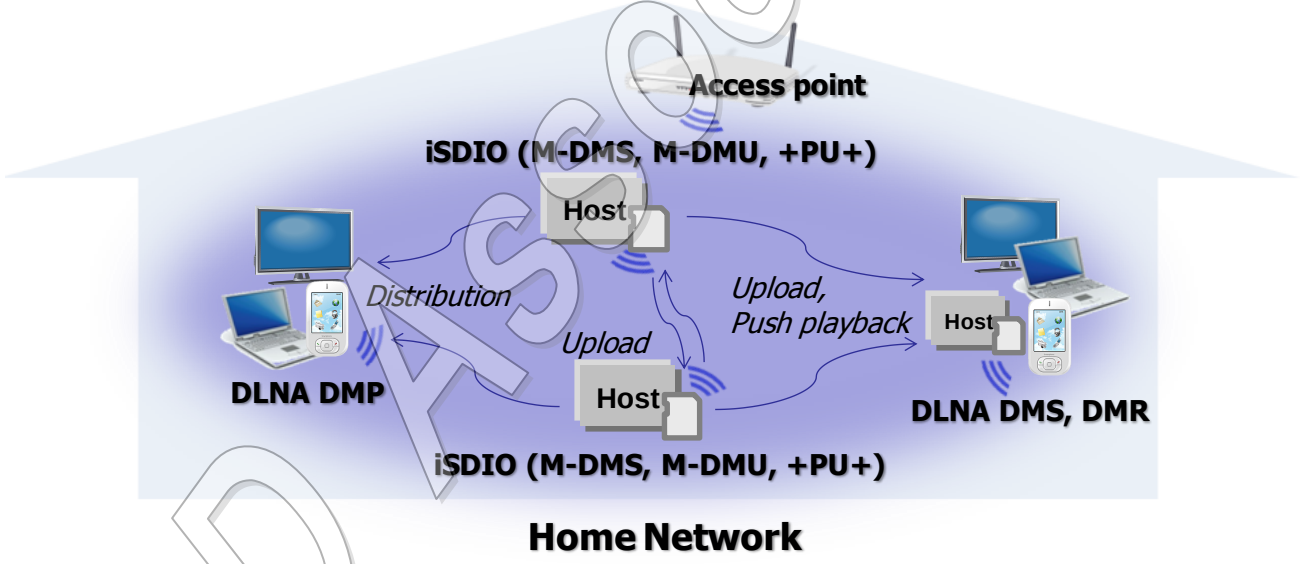


Figure 1-2 : DLNA use case

Basic Steps

(1) A Sender Card creates wireless LAN connection with a Receiver Card or Receiver DLNA device via an access point or Wi-Fi Direct group.
(2) A Sender Card selects one or more files to be sent to a Receiver Card or Receiver DLNA device.
(3) A Sender Card sends the selected files to a selected Receiver Card or Receiver DLNA device.

1.3.3 Peer-to-Peer File Transfer

This specification enables a Card to support the file transfer for Peer-to-Peer networks through the wireless LAN Infrastructure mode (AP and STA).
This use case is that a Card (a Sender Card) sends a file to another Card (a Receiver Card), where

- Alternatively a PC or a Smart Phone is able to be a Sender or a Receiver if the application on the PC or the Smart Phone supports the communication protocol defined in this specification.
- A case that a PC or a Smart Phone communicates with a Card via a communication protocol not defined in this specification is out of scope in this specification.

The followings are the basic steps for this use case, and the corresponding commands and status information are defined in this specification to realize this use case. Note that the basic steps are typical, and a Host implementer is able to realize this use case in other ways by using the commands and status information defined in this specification.

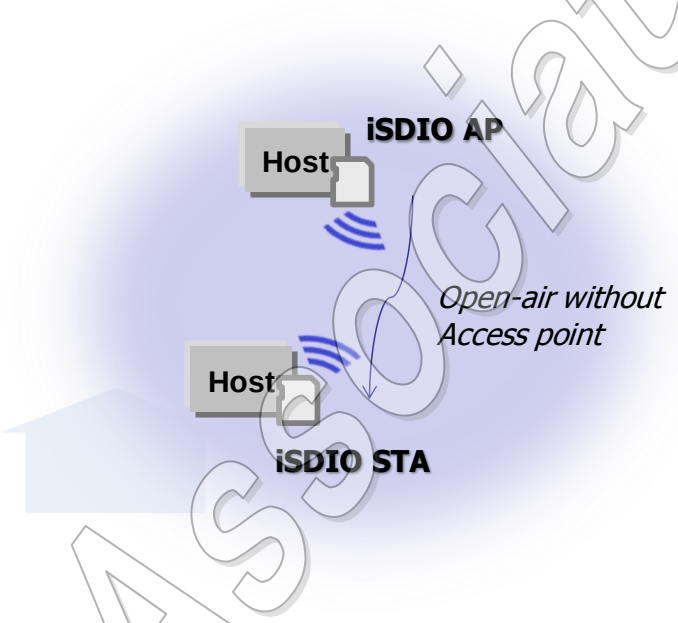


Figure 1-3 : Peer to Peer use case

Basic Steps

(1) A Sender Card connects to one or more Receiver Cards in wireless LAN.
(2) A Sender Card selects one or more files to be sent to Receiver Cards.
(3) A Sender Card selects one or more Receiver Cards to be sent the selected files.
(4) A Sender Card sends the selected files to the selected Receiver Cards. Optionally each Receiver Card is able to select files to be received before they are sent.

1.3.4 PTP

This specification enables a Card to support use cases defined by PTP. A Card supports a proprietary PTP application by the PTP Pass-Through framework.

The PTP use cases have three transfer scenarios for PTP devices. The first scenario prints a picture to DPS printer. The second scenario transfers the content to PC. One more scenario, the PTP Pass-Through mode provides the PTP application framework to a proprietary PTP application.

The followings are the basic steps for the first scenario, and the corresponding commands and status information are defined in this specification to realize the use case. Note that the basic steps are typical, and a Host implementer is able to realize this use case in other ways by using the commands and status information defined in this specification.

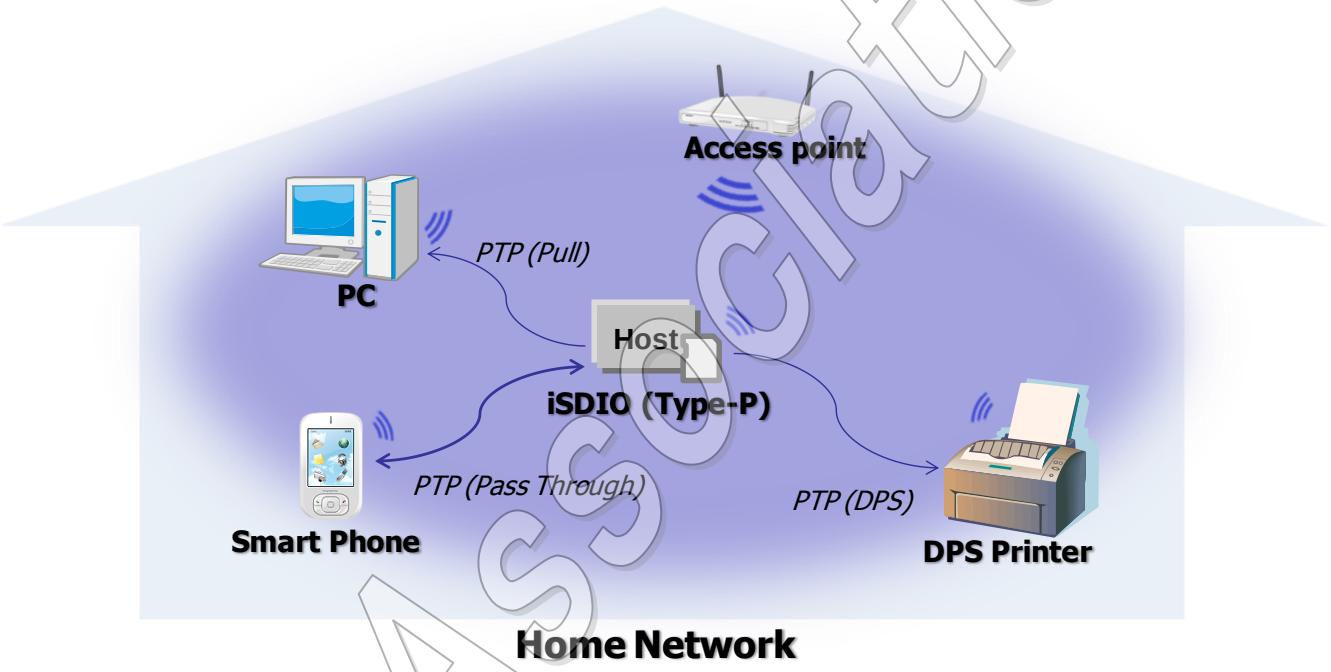


Figure 1-4 : PTP use case

Basic Steps

(1) A Card creates wireless LAN connection with a DPS Printer device via an access point or Wi-Fi Direct group.
(2) A Card selects one or more files to be sent to a DPS Printer device.
(3) A Card sends the selected files to a selected DPS Printer device.

1.4 Card Type

This Wireless LAN Addendum Specification defines three types of Card described in Figure 1-5. The iSDIO Wireless LAN Card shall support one of the 5(five) configurations below. On the other hand, a Host supports the one configuration in accordance with its application. See **6.3 iSDIO Wireless LAN Card Type** for details.

(1) Type-W only
(2) Type-D only
(3) Type-P only
(4) Type-W and Type-D
(5) Type-W and Type-P

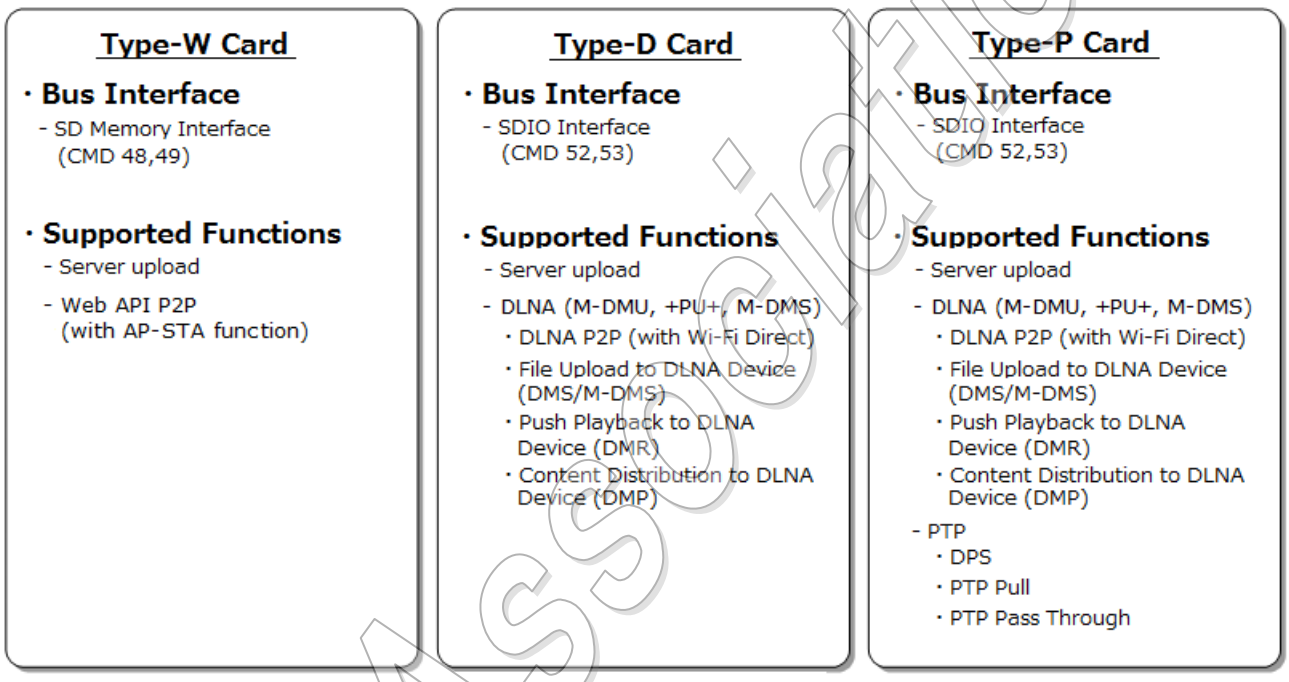


Figure 1-5 : Wireless LAN Card Types

Note that the Type-P specification includes the Type-D all specification.

1.5 Endian

- Wireless LAN Card takes the **little endian** representation.
- Multiple-byte numerical values in a description field shall be recorded in the **little endian** form. e.g.) the 32-bit hexadecimal number ‘12345678h’ shall be recorded as ‘78h’, ‘56h’, ‘34h’ and ‘12h’ in this order.

2. iSDIO Wireless LAN System Model and Functions

2.1 iSDIO Wireless LAN Interface Model

Figure 2-1 shows the "iSDIO Wireless LAN Card Interface and Network Interface" between a Card and related devices/services, and Figure 2-2 shows "iSDIO Wireless LAN Card Interface and Card System Model".

"Card Interface" (Figure 2-1) is for interactions between a Host and a Card, and SD Memory commands ([SDPart1]) are used for a Host to read/write data in a "NAND Memory module" (Figure 2-2). A Card may have an embedded "NAND Memory module" or may have a (micro) SD Card slot to insert an external (micro) SD Memory Card.

In addition, this specification defines an interface for a Host to control the "Wireless LAN module" (Figure 2-2) via the iSDIO Register which is accessed by CMD52/53 ([SDIO]) or CMD48/49 ([SDPart1]). In accordance with instructions from a Host via the iSDIO Register, the Wireless LAN module connects to a wireless LAN and communicates connected devices or servers (e.g. PC, Smart Phone, another iSDIO Card, Server) via "Network Interface" (Figure 2-1). A Card shall support the HTTP protocol as the Network Interface.

For a Host to access to the iSDIO Register via the Card Interface, see **3. iSDIO Wireless LAN Register Definition**, and to communicate via the Network Interface, see **4. iSDIO Wireless LAN Commands**.

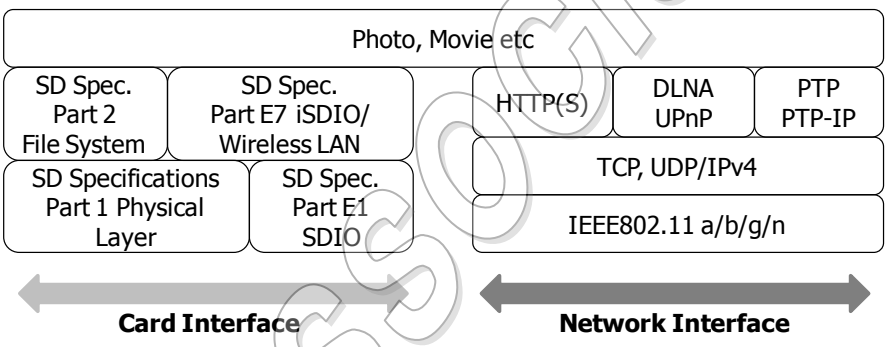


Figure 2-1 : iSDIO Wireless LAN Card Interface and Network Interface

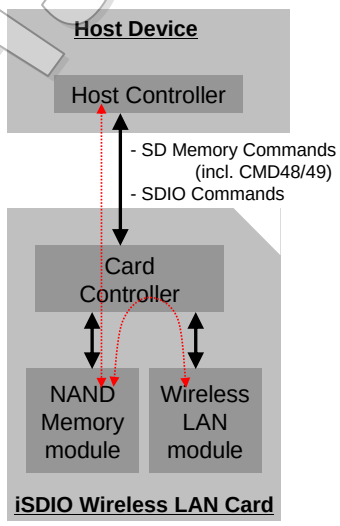


Figure 2-2 : iSDIO Wireless LAN Card Interface and Card System Model

2.2 Wireless LAN State Diagram

Figure 2-3 shows the state diagram of the wireless LAN. A state of the current wireless LAN is stored as "WLAN" in the Status Register (**3.4 iSDIO Status Register for Wireless LAN**) and a Host is able to read it.

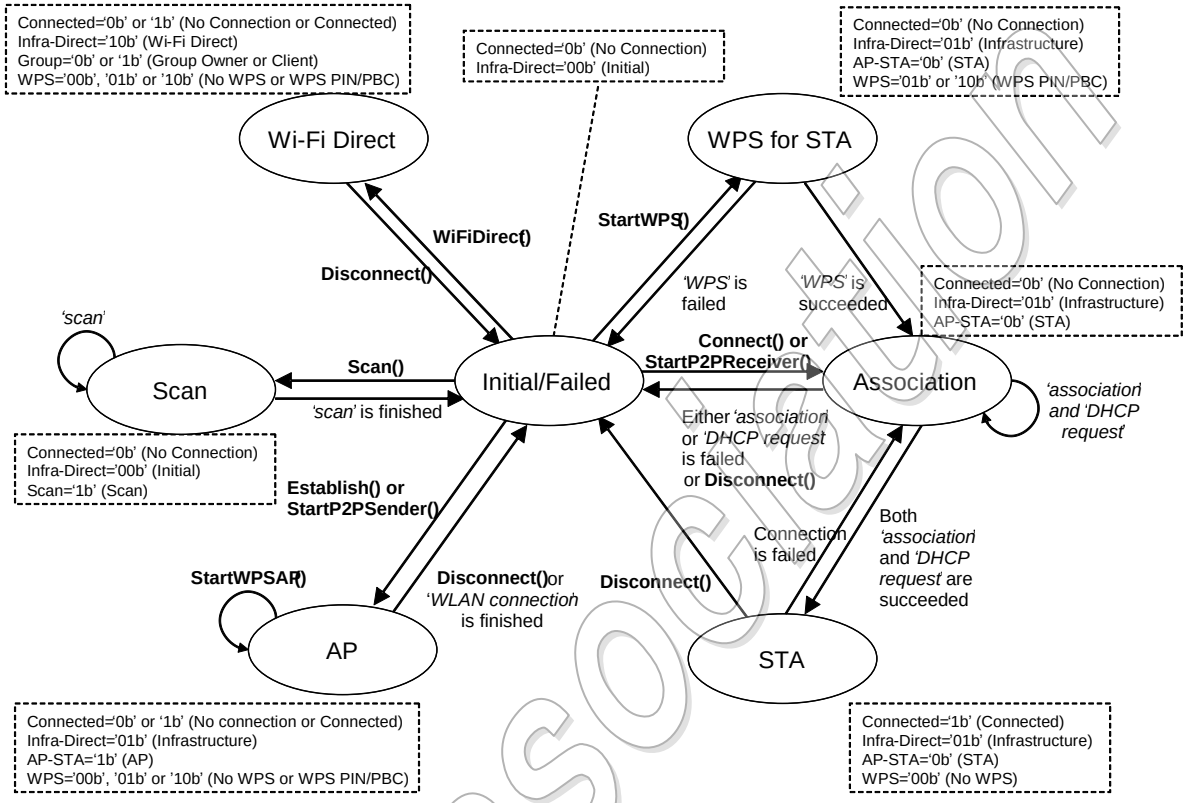


Figure 2-3 : Wireless LAN State Diagram

2.3 Server Upload

This specification enables a Host to connect to any servers of the Host-vendor's choice, while the protocol to interact with each server is different. This specification then requires the Host to implement such a protocol. A Card shall send an HTTP request message prepared by a Host and receive an HTTP response message from a server, and then the Host interprets the HTTP response message received from the Card.

See **4.4 Server Upload Commands** for the interface for a Host to control a Card.

2.3.1 HTTP Request Message

For an HTTP request message, a Host shall prepare an HTTP request message and may store the message as a file ("message file"). The HTTP request message shall contain an HTTP Request Line and an HTTP message header and may contain an HTTP message body (optional in [HTTP]). Then the Host requests the Card to send the HTTP request message to the specified server with the "SendHTTPMessageByRegister" or "SendHTTPSSLMessageByRegister" command, or the "SendHTTPMessageByFile" or "SendHTTPSSLMessageByFile" command if the message file is created.

In an HTTP request message, an HTTP Request Line and an HTTP message header shall comply with the [HTTP], and an HTTP message body depends on the service. The following is an example of an HTTP request message.

```
POST /hoge/ HTTP/1.1
Content-Type: text/plain
User-Agent: XXXXXXXX
Host: hogehoge.com
Content-Length: 19

THIS IS AN EXAMPLE.
```

Note that a Host should not add any HTTP message headers to specify the HTTP Client (i.e. Card) behavior, and the Card should behave as if such HTTP message headers were not included. Followings are examples of such HTTP message headers.

- Expect: 100-continue
- Transfer-Encoding: chunked
- Connection: Keep-Alive

Additionally, the "SendHTTPFileByRegister" and "SendHTTPSSLFileByRegister", "SendHTTPFileByFile" (for sending message file) and "SendHTTPSSLFileByFile" (for sending message file) commands shall have a function to replace pre-defined characters ("<!--WLANSDFILE-->" in [ISO 8859-1]) with the specified file, in addition to the functions of "SendHTTPMessageByRegister", "SendHTTPSSLMessageByRegister", "SendHTTPMessageByFile" and "SendHTTPSSLMessageByFile" commands respectively. They are used to attach a file (attach file) to a message file. The following is an example of an HTTP request message to attach a file.

```
POST /hoge/ HTTP/1.1
Content-Type: image/jpeg
User-Agent: XXXXXXXX
Host: hogehoge.com
Content-Length: 1234567

<!--WLANSDFILE-->
```

A Card creates the HTTP request message below by attaching the specified file to the above HTTP request message.

```
POST /hoge/ HTTP/1.1
Content-Type: image/jpeg
User-Agent: XXXXXXXX
Host: hogehoge.com
Content-Length: 1234567

<Attach file is inserted here by a Card>
```

2.3.2 HTTP Response Message

A Card shall receive an HTTP response message of the HTTP request message prepared by the "SendHTTPMessageByRegister", "SendHTTPFileByRegister", "SendHTTPSSLMessageByRegister" or "SendHTTPSSLFileByRegister" command. The HTTP response message shall contain an HTTP Response Line and an HTTP message header, and may contain an HTTP message body (optional in [HTTP]). A Host is able to read the HTTP response message via the Response Data Register Port (See **2.2.2 iSDIO Register** in [iSDIO]).

Alternatively a Card shall receive an HTTP response message of the HTTP request message

prepared by the "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile" or "SendHTTPSSLFileByFile" command and shall store the HTTP response message as a file to the NAND memory module. The HTTP response message shall contain an HTTP Response Line and an HTTP message header, and may contain an HTTP message body (optional in [HTTP]). The file is stored under the RESPONSE directory and its filename is specified by the "iSDIO command sequence id" (See **2.2.2.2 Command Write Data** in [iSDIO]), which is differently assigned among commands by a Host and has one of values from "00000000" to "FFFFFFFF" hexadecimal. A Host will read the file and interpret the HTTP response message sent from the server.
See also **2.7 Directory and File Structure** as for the directory and the file formats.

2.4 DLNA Applications

This specification introduces a DLNA content transfer feature.
As described in Table 2-1, the +UP+ Card becomes the M-DMU device, and DMS and -UP- features are implemented on M-DMS device. The +PU+ is the push controller device capability for iSDIO Card.
DLNA feature supports three content transfer applications by two device classes and one device capability (M-DMS, M-DMU and +PU+). The first +UP+ application provides a content transfer from Card to Card or DMS device. The second +PU+ application provides a content push playback from Card to DMR device. The third DMS application provides a content distribution from Card to DMP device on demand. Moreover, DMS application supports -UP- application too.
The iSDIO Card and the DLNA devices connect each other via a wireless LAN access point, or connect via a Wi-Fi Direct network group.

See **4.5 DLNA Commands** for the interface for a Host to control a Card.
See **Appendix D** for the DLNA upload transfer sequence.

Application Name	DLNA Device Class /Capability	Target DLNA device (Receiver)	Functions
+UP+	M-DMU	M-DMS, DMS	1. The +UP+ connects to a wireless LAN access point or Wi-Fi Direct network. 2. The +UP+ may select the receiver M-DMS/DMS discovered by UPnP device discovery. 3. The +UP+ provides selected file to the selected M-DMS/DMS.
+PU+	+PU+	DMR	1. The +PU+ connects to a wireless LAN access point or Wi-Fi Direct network. 2. The +PU+ may select the receiver DMR discovered by UPnP device discovery. 3. The +PU+ provides selected file to the selected DMR.
DMS	M-DMS	M-DMP, DMP	1. The DMS connects to a wireless LAN access point or Wi-Fi Direct network. 2. The DMS broadcasts an advertisement packet. 3. The DMP sends a response packet to the UPnP device discovery. 4. The DMS distributes contents requested from DMP.
-UP-		+UP+	1. The -UP- connects to a wireless LAN access point or Wi-Fi Direct network. 2. The -UP- broadcasts an advertisement packet. 3. The -UP- sends a response packet to the UPnP device discovery. 4. The -UP- receives file from the +UP+.

Table 2-1 : DLNA Applications

Note that +UP+ is defined by the DLNA guideline as an upload controller. On the other hand, -UP- is not defined by the DLNA guideline, but -UP- is defined by this specification as a receiver device of +UP+ for the sake of convenience.

Note that a -UP- device shall support the "CreateObject" action of Content Directory Service in this specification.

2.5 Peer-to-Peer File Transfer

This specification defines a simple Peer-to-Peer File Transfer Application. As described in Table 2-2, the Sender Card becomes an AP and a DHCP Server, and Receivers connect to the Sender Card as the AP. Then the Sender sends files in the Sender Card to the Receivers.
See **4.6 P2P File Transfer Commands** for the interface for a Host to control a Card.
See **Appendix C** for the interface between a Sender Card and a Receiver Card.
See **Appendix D** for the Peer-to-Peer File Transfer sequence.

Application Name	WLAN	DHCP	Functions
P2P Sender	Infrastructure AP	DHCP Server	1. The Sender provides wireless LAN for the Receivers. 2. The Sender becomes the DHCP Server and assigns the IP Address for the Receivers. The Sender notifies Sender's IP Address as most preferred DNS Server's IP Address to the Receivers, as if the Sender was a DNS Server via DHCP. 3. The Sender may select the Receivers to access. 4. The Sender provides the FILELIST file to the Receivers. 5. The Sender provides files in the FILELIST file to the Receivers.
P2P Receiver	Infrastructure STA	DHCP Client	1. The Receiver connects wireless LAN and gets the Receiver's IP Address and the Sender's IP Address as most preferred DNS Server's IP Address. 2. The Receiver sends the access request to the Sender. 3. The Receiver receives the FILELIST file from the Sender. 4. The Receiver may select the files in the FILELIST file. 5. The Receiver receives the files in the FILELIST file from the Sender.

Table 2-2 : Peer-to-Peer File Transfer Applications

2.6 PTP Applications

This specification introduces a PTP-IP content transfer feature.
As described in Table 2-3, a Card becomes PTP responder. PTP feature supports three applications. The first application is DPS, it provides picture print from Card to DPS printer. The second application is PTP Pull, it provides a content transfer from Card to PC. The third application is PTP Pass-Through, the detail of this application depends on the control of the Host.
The iSDIO Card and the PTP-IP devices connect each other via a wireless LAN access point, or connect via a Wi-Fi Direct network group.

See **4.7 PTP Commands** for the interface for a Host to control a Card.
See **Appendix D** for the PTP transfer sequence.

Application Name	Responder/Initiator	Target Device	Functions
DPS	Responder	DPS Printer	1. The responder connects to a wireless LAN access point or Wi-Fi Direct network. 2. The responder starts device discovery process to search initiators that is operating as DPS printer. 3. The responder may select a target printer discovered in the device discovery process. 4. The responder requests the selected printer to connect. 5. The printer as an initiator connects to the responder. 6. The responder provides objects to the printer.
PTP Pull	Responder	PC	1. The responder connects to a wireless LAN access point or Wi-Fi Direct network. 2. The responder starts advertisement as PTP-IP device. 3. Initiators search responders via device discovery process. 4. An initiator connects to the responder. 5. The initiator requests objects. 6. The responder provides the requested objects to the initiator.
PTP Pass Through	Responder	(not fixed)	1. The responder connects to a wireless LAN access point or Wi-Fi Direct network. 2. The responder starts advertisement as PTP-IP device. 3. Initiators search responders via device discovery process. 4. An initiator connects to the responder. 5. The initiator requests operations. 6. The responder responds the requests depending on the control of the Host.

Table 2-3 : PTP Applications

In DPS mode, a Card shall carry out the device discovery process (the search of initiators which is DPS printer) inside all the time while it operates with the mode. The detail of device discovery process for DPS mode is described in [DPS over IP].

In PTP Pull mode and PTP Pass-Through mode, both MTP/IP (described in [MTP/IP]) and Multicast DNS (described in [Multicast DNS]) shall be used as the device discovery process (the advertisement as a PTP-IP device to initiators). These processes are not active at the start of the mode, and are invoked by a Host by issuing the "PTP_StartAdvertisement" command. A Card shall operate two methods at the same time.

Note that in PTP Pull mode, a Card is given only the permission to read of the file system of the NAND memory module. Therefore a Card shall not support the PTP operation which changes the content of the NAND memory module.

Note that following definitions about PTP applications (e.g. registers, commands) assume that all the PTP applications support only one PTP session on one PTP-IP connection.

Note that in all PTP modes, a Card shall interpret Probe Request packets from an initiator and respond them internally. Additionally a Card should send Probe Request packets to the initiator to know whether PTP-IP connection is still alive. The details of sending Probe Request packets (interval, timing, etc.) depend on implementation of a Card.

2.7 Directory and File Structure

Figure 2-4 shows the directory and file structure of an iSDIO Wireless LAN Card. The "DCIM" directory is defined in [DCF]. In addition to this, this specification defines the "SD_WLAN" directory which contains the "RESPONSE" directory, "CONFIG" file, "FILELIST" file and "CDSDB" directory, which are defined in the following sections.

In addition to that, the following restrictions shall be applied to the files and the directories handled by the iSDIO Wireless LAN commands in this specification.

- The length of the filename shall be equal or less than 8-characters for the name part and 3-characters for the extension part.
- The length of the directory name shall be equal or less than 8-characters.
- The character of the filename and directory name shall be "0" to "9", "A" to "Z" or "_" ("30h" to "39h", "41h" to "5Ah" or "5Fh" in ASCII character respectively).

Also the following restrictions on the file path, which may be specified in an argument of an iSDIO Wireless LAN command or in the FILELIST file, shall be applied.

- A file path shall be the absolute path from the root directory of a Card.
- A directory separator in a file path shall be "/" ("2Fh" ASCII character).

Any configuration files or configuration utilities related to the Wireless LAN Addendum shall be stored in locations that are accessible to the host device.

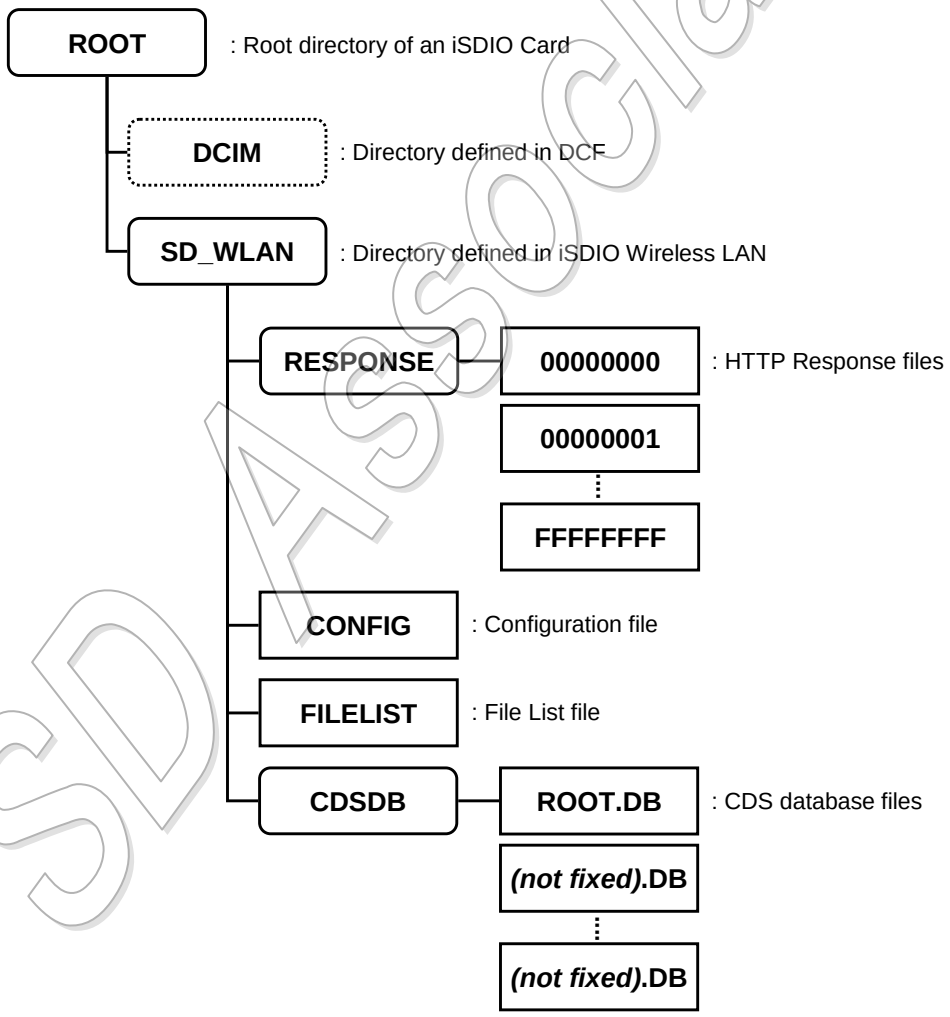


Figure 2-4 : iSDIO Wireless LAN Directory and File Structure

2.7.1 RESPONSE Directory

Under the RESPONSE directory, a Card may store an HTTP response message as a file. The HTTP response message is a response to an HTTP request message, which is issued by "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile", or "SendHTTPSSLFileByFile" commands. The HTTP response message file shall contain an HTTP response line and an HTTP message header, and may contain an HTTP message body. The filename is specified by the "iSDIO command sequence id" (See **2.2.2.2 Command Write Data** in [iSDIO]), which is assigned by a Host and has one of values from "00000000" to "FFFFFFFF" in the hexadecimal number.
Note that a Card doesn't delete files under the RESPONSE directory and a Host is assumed to manage them.

2.7.2 CONFIG File

A Host may create a Configuration file under the "SD_WLAN" directory and its format is defined in **5 iSDIO Wireless LAN Configuration File**.
If a Card supports the Configuration file, the Card parses the file and sets the specified parameters when a wireless LAN connection or an application is started. Even when a Card supports the file but a Host does not create the file, the default value shall be set to the Card.
If a Card doesn't support the Configuration file, the Card ignores the file even if it is prepared at the specified location.
A Host is able to know whether the Configuration file is supported by reading the "Configuration File Support" in the iSDIO Capability Register. (See **2.2.2.4 iSDIO Capability Register** in [iSDIO])

2.7.3 FILELIST File

In the Peer-to-Peer File Transfer in the Infrastructure, the Sender Card creates a list of files to be sent to Receivers and stored the list as a FILELIST file under the "SD_WLAN" directory. The Receiver is required to get the list and store it as a FILELIST file under the "SD_WLAN" directory before requesting the file download. See **2.5 Peer-to-Peer File Transfer** for the Peer-to-Peer File Transfer. Note that this file may not exist if a Card doesn't support the Peer-to-Peer File Transfer in the Infrastructure.

The first line in the FILELIST file shall have the identifier "WLANSD_FILELIST" described in [ISO 8859-1] and shall be terminated by the line break character [CR][LF] ("0D0Ah"), as shown below.

```
WLANSD_FILELIST[CR][LF]
```

In the following lines, zero or more filenames (including file paths) and the corresponding file sizes in bytes shall be described in [ISO 8859-1].
Optionally, the last modified date and time of a file ("YYYY-MM-DDTHH:MM:SS" format), the thumbnail's filename (including file path) and the vendor unique filename (including file path) may be described in this order in addition to the filename and file size in [ISO 8859-1]. These are delimited by a comma character ("2Ch") in a line, and each line shall be terminated by the line break character [CR][LF] ("0D0Ah"), as shown below.

```
filename,file_size,Date_Time,thumbnail's_filename,vendor_unique_filename[CR][LF]
```

Note that blank lines shall not be contained in the file.

See also the example below. In the first line, only the filename, the file size, the date and time and

the thumbnail's filename are described. In the second line, only the filename, the file size and the vendor unique filename are described. In the third line, the filename, the file size and the date and time are described.

(example of FILELIST file)

```
WLANSD_FILELIST
/DCIM/100XXXXX/YYYY0001.JPG,123456,2011-04-21T15:52:00,/THUMB/ZZZZ0001.JPG,
/DCIM/100XXXXX/YYYY0002.JPG,234567,,,/META/MMMM0001.TXT
/DCIM/100XXXXX/YYYY0003.JPG,345678,2011-04-21T15:55:10,,
```

2.7.4 CDSDB Directory

Under the CDSDB directory, a content distribution database used in DLNA M-DMS mode is stored as files. These files make a hierarchy of the content distribution database by referring each other. Here, one file constitutes one container class described in [DLNA].
A Host shall issue the "DLNA_SetUpdateID" command to validate database files placed under this directory.
In the processing of the "DLNA_AcceptBrowse" command, a Card uses these files to generate a response of the Browse request. The format of each file is defined in this specification.

Note that content distribution database files are created by a Host or other devices (e.g. PC), not a Card.
Note that all files stored in this directory shall belong to the hierarchy of the content database. A Host shall not exclude a specific file from the hierarchy by not referring to the file from all other files.
Note that this directory may be empty or not exist, depending on implementation of a Host.

2.7.4.1 CDS Database Files

The format of the files under the CDSDB directory is described in this section. All files shall be named using extension "DB". Especially, the file used for the root container shall be named as "ROOT.DB". Basically, the description rule of each parameter complies with [DLNA]. Therefore, each file shall be described in [UTF-8].

The first line in the CDS database file shall have the identifier "WLANSD_CDS_DATABASE" as shown below.

```
WLANSD_CDS_DATABASE[CR][LF]
```

In the second line, the this_container's_title is described as shown below.

```
this_container's_title[CR][LF]
```

this_container's_title specifies the title of the container constituted from this file. The 'dc:title' metadata of this container shall be set to this value.

In the third line, the parent_database's_filename is described as shown below. If a file is for root container, this line shall be ignored. (It may be blank line.)

```
parent_database's_filename[CR][LF]
```

parent_database's_filename specifies the filename of the database file which constitutes parent container of this container. The file path shall not be included.

In the following lines, the object list of this container is constituted. The interpretation of each line is distinguished by a top letter. If a file ends at the third line, it means the container is empty.
The parameters are delimited by a comma character ("2Ch") in a line, and each line is terminated by the line break character [CR][LF] ("0D0Ah"), as shown below.

A line beginning from "C" ("43h") indicates a container object.

```
C,database's_filename,container's_title[CR][LF]
```

database's_filename specifies the filename of the database file which constitutes this container object. The file path shall not be included.
container's_title specifies the title of this container object. The 'dc:title' metadata of this object shall be set to this value. This value shall be the same as described in the second line of the database file specified at database's_filename.

A line beginning from "I" ("49h") indicates an item object with one resource.

```
I,resource's_filename,file_size,MIME-
type,profileID,Date_Time,item's_title[CR][LF]
```

resource's_filename specifies the filename of the entity for this resource. The file path shall be included.
file_size specifies the size of a file appointed in 'filename'. The 'res@size' metadata of this resource shall be set to this value.
MIME-type specifies the MIME-type of this resource. The 'res@protocolInfo' metadata of this resource shall be constructed using this value.
profileID specifies the profileID of this resource. The 'res@protocolInfo' metadata of this resource shall be constructed using this value.
Date_Time specifies the date and time to apply to this resource. This field shall be described using the "CCYY-MM-DDThh:mm:ss" format. The 'dc:date' metadata of this object shall be set to this value.
item's_title specifies the title of this item object. The 'dc:title' metadata of this object shall be set to this value.

A line beginning from "R" ("52h") indicates the other resource of the previous item object. If the previous line doesn't specify an item object, this line shall be ignored.

```
R,resource's_filename,file_size,MIME-type,profileID[CR][LF]
```

resource's_filename specifies the filename of the entity for this resource. The file path shall be included.
file_size specifies the size of a file appointed in 'filename'. The 'res@size' metadata of this resource shall be set to this value.
MIME-type specifies the MIME-type of this resource. The 'res@protocolInfo' metadata of this resource shall be constructed using this value.
profileID specifies the profileID of this resource. The 'res@protocolInfo' metadata of this resource shall be constructed using this value.

Note that if a comma character ("2Ch") is used in any title fields, a Card shall not treat it as a delimiter. Because every title fields are defined as the last field of each line, a Card can distinguish whether a comma character shall be treated as a delimiter or not.
Note that HTML encoding is not required on these files. For example, "&" ("26h") shall be used to describe an ampersand character. (It shall not be replaced to "&". The replacement shall be processed by a Card internally.)
Note that blank line shall not be contained in the file, except the line which describes the parent_database's_filename of root container.

See also the example below. It describes a container object that contains four child objects. The first one and the second one are container objects. The third one is an item object which consists of single resource. The last one is an item object which consists of three resources. In addition, it is assumed that the root container includes this container as its child object.

(example of CDS database file)

```
WLANSD_CDS_DATABASE
Sample Container
ROOT.DB
C,SUN_MOON.DB,The sun & The moon
C,FLOWER.DB,Flowers
I,/DCIM/100XXXXX/SSSS0001.JPG,123456,image/jpeg,JPEG_SM,2011-04-
21T15:55:10,Single Resource
I,/DCIM/100XXXXX/SSSS0002.JPG,123456,image/jpeg,JPEG_SM,2011-04-
21T15:55:10,Three Resources
R,/DCIM/100XXXXX/MMMM0002.JPG,234567,image/jpeg,JPEG_MED
R,/DCIM/100XXXXX/LLLL0002.JPG,345678,image/jpeg,JPEG_LRG
```

3. iSDIO Wireless LAN Register Definition

3.1 iSDIO Type Support Code for Wireless LAN

Table 3-1 shows the Wireless LAN-specific structure of the "iSDIO Type Support Code" (0108h in FBR) defined in **2.2.1.2.1 Additional definitions in FBR for iSDIO** in [iSDIO].

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
108h in FBR	-	-	-	-	-	**iSDIO Wireless LAN Type-P Support**	**iSDIO Wireless LAN Type-D Support**	**iSDIO Wireless LAN Type-W Support**

Table 3-1 : iSDIO Type Support Code for Wireless LAN

iSDIO Wireless LAN Type-W Support specifies whether the Card supports the Type-W Card function defined in **6.3 iSDIO Wireless LAN Card Type**. One of the following values shall be set.
0b: Not Supported
1b: Supported
iSDIO Wireless LAN Type-D Support specifies whether the Card supports the Type-D Card function defined in **6.3 iSDIO Wireless LAN Card Type**. One of the following values shall be set.
0b: Not Supported
1b: Supported
iSDIO Wireless LAN Type-P Support specifies whether the Card supports the Type-P Card function defined in **6.3 iSDIO Wireless LAN Card Type**. One of the following values shall be set.
0b: Not Supported
1b: Supported

Note that in case of Type-P card, both Type-D bit and Type-P bit shall be set.
Note that at least one of the defined types in the above table shall be supported.

3.2 Command Response Status for Wireless LAN

This section defines the Wireless LAN-specific "response status" defined in **2.2.2.1.1.1 Command Response Status** in [iSDIO]. From "00h" to "BFh" are reserved in [iSDIO], in addition this specification defines the following "response status" for the Process Failed status.
C0h: SSL ServerHello error (The specified cipher suite by a client may not be supported in a server. However, this error may occur by the other reasons.)
C1h: SSL Server Certificate Verification error (The server certificate is invalid when it is verified by the specified root certificate.)
C2h: PTP-IP cancelled (The command processing was aborted by an initiator generated cancel. The Card may not complete the data transfer.)
C3h to FFh: reserved

3.3 Total Card Power (TCP) for Wireless LAN

This section defines the Wireless LAN-specific power consumption requirement. iSDIO Wireless LAN Card which requires power over 0.72W shall support at least either one of 010b (1.44W with max 400mA on VDD1, max 200mA on VDD2), 011b (1.80W with max 400mA on VDD1, max 200mA on VDD2) and 100b (1.80W with max 500mA on VDD1) as TCP settings.
If iSDIO Wireless LAN Card doesn't require power over 0.72W, it may not support the above TCP modes.

3.4 iSDIO Status Register for Wireless LAN

Table 3-2 shows a structure of the Application Status for Wireless LAN in the iSDIO Status Register. See **2.2.2.1 iSDIO Status Register** in [iSDIO] for the Application Status.

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Type
0500h	DLNA Status (Interrupt)	-	-	-	-	**CDR**	**CBR**	**DLU**	**ULR**	R/O
0501h	P2P Status (Interrupt)	-	-	-	-	-	-	**FLU**	**ILU**	R/W
0502h	PTP Status (Interrupt)	-	-	**CIL**	**DPI**	**CPI**	**RPC**	**RPD**	**RPO**	R/O
0503h	Reserved for Vendor (Interrupt)	-	-	-	-	-	-	-	-	
0504h	Application									R/O
0505h	Reserved									R/O
0506h	WLAN	Connected	**Infra-Direct**		AP-STA	Group	**WPS**		**Scan**	R/O
0507h		-	-	-	-	-	-	-	-	R/O
0508h - 0527h	SSID									R/O
0528h	Encryption Mode									R/O
0529h	Signal Strength									R/O
052Ah	Channel									R/O
052Bh - 052Fh	Reserved									
0530h - 0535h	MAC Address									R/O
0536h - 053Fh	Reserved									
0540h - 054Fh	ID									R/O
0550h - 0553h	IP Address									R/O
0554h - 0557h	Subnet Mask									R/O
0558h - 055Bh	Default Gateway									R/O
055Ch - 055Fh	Preferred DNS Server									R/O
0560h - 0563h	Alternate DNS Server									R/O
0564h	Proxy Server	-	-	-	-	-	-	-	**PSE**	R/O
0565h - 056Fh	Reserved									
0570h - 0571h	Date									R/O
0572h - 0573h	Time									R/O
0574h	HTTP Status	**HPC**	**HTTP Progress**							R/O

0575h	Power Save Management	-	-	-	-	-	-	-	**PSM**	R/O
0576h	File System Management	-	-	-	-	-	-	-	**FIM**	R/W
0577h	PTP Transfer Status	-	**PTP Progress**							R/O
0578h - 057Bh	PTP Cancelled TransactionID									R/W
057Ch	PTP Error Status									R/O
057Dh - 05BFh	Reserved									
05C0h - 05FFh	Reserved for Vendor									

Table 3-2 : Application Status for Wireless LAN

DLNA Status:

ULR (Upload Requested) specifies whether UPnP Action CreateObject is detected. This read-only bit shall be updated by a Card while operating in DLNA M-DMS mode. (The start and end timing of this mode is managed by issuing of "DLNA_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not requested (Default)
1b: Requested
If a Host detects this bit is set to '1b', a Host shall issue the "DLNA_GetUploadInformation" command and confirm whether the upload is acceptable or not. Then, a Host shall issue the "DLNA_AcceptUpload" command to notify a Card of the confirmation result.
Note that this field is changed to '0b' by the "DLNA_AcceptUpload" command. Note that this field is always '0b' if a Card doesn't support DLNA M-DMS mode.

DLU (Device List Update) specifies whether device list is updated. This read-only bit shall be updated by a Card while operating in DLNA M-DMU mode or DLNA +PU+ mode. (The start and end timing of this mode is managed by issuing of "DLNA_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not updated (Default)
1b: Updated
If a Host detects this bit is set to '1b', a Host is able to get the updated list by issuing the "DLNA_GetDeviceList" command.
Note that this field is changed to '0b' by the "DLNA_GetDeviceList" command. Note that this field is always '0b' if a Card supports neither DLNA M-DMU mode nor DLNA +PU+ mode.

CBR (Content Browse Requested) specifies whether UPnP Action Browse (or similar one such as Search) is detected. This read-only bit shall be updated by a Card while operating in DLNA M-DMS mode. (The start and end timing of this mode is managed by issuing of "DLNA_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not requested (Default)
1b: Requested
If a Host detects this bit is set to '1b', a Host shall issue the "DLNA_AcceptBrowse" command to notify whether the request is acceptable or not.
Note that this field is changed to '0b' by the "DLNA_AcceptBrowse" command. Note that this field is always '0b' if a Card doesn't support DLNA M-DMS mode.

CDR (Content Distribution Requested) specifies whether HTTP GET request is detected. This read-only bit shall be updated by a Card while operating in DLNA M-DMS mode. (The start and end timing of this mode is managed by issuing of "DLNA_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not requested (Default)

1b: Requested
If a Host detects this bit is set to '1b', a Host shall issue "DLNA_AcceptDistribution" command to notify whether the request is acceptable or not.
Note that this field is changed to '0b' by the "DLNA_AcceptDistribution" command. Note that this field is always '0b' if a Card doesn't support DLNA M-DMS mode. Note that this field shall be valid in M-DMS mode only. In +PU+ mode, HTTP GET request shall be handled by only a Card itself internally.

If either bit of this register changes from '0b' to '1b', a Card shall set ASU (Application Status Update) bit of iSDIO Status to '1b'. If ASU_ENA bit of iSDIO Int Enable is '1b', an interrupt shall be issued.
Note that a Card shall respond to the requests without notification of the event to a Host, in case of the following situations.

- The database files for the content distribution database are not valid. (CBR/CDR)
- It is clear that the request shall be handled as an error. (ULR/CBR/CDR)

P2P Status:

FLU (File List Update) specifies whether the File List has been received for the P2P Receiver in Peer-to-Peer File Transfer. One of the following values shall be set.
0b: Not updated (Default)
1b: Updated
Note that this field is changed to '0b' by the "Disconnect" command, and is able to be reset to '0b' when accessed by a Host. Note that this field is always '0b' if a Card doesn't support Peer-to-Peer File Transfer.
ILU (ID List Update) specifies whether the ID List is updated for the P2P Server in Peer-to-Peer File Transfer. One of the following values shall be set.
0b: Not updated (Default)
1b: Updated
Note that this field is reset to '0b' by the "Disconnect" command and by the "ReadIDList" command, or a Host is able to be reset to '0b'. Note that this field is always '0b' if a Card doesn't support Peer-to-Peer File Transfer.

If either bit of this register changes from '0b' to '1b', a Card shall set ASU (Application Status Update) bit of iSDIO Status to '1b'. If ASU_ENA bit of iSDIO Int Enable is '1b', an interrupt shall be issued.

PTP Status:

RPO (Received PTP Operation data) specifies whether an Operation Request packet is received. This read-only bit shall be updated by a Card while operating in DPS mode or PTP Pass-Through mode. (The start and end timing of this mode is managed by issuing of "PTP_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not received (Default)
1b: Received
If a Host detects this bit is set to '1b', a Host is able to get the received data by issuing the "PTP_ReceiveOperation" command.
Note that this field is changed to '0b' by the "PTP_ReceiveOperation" command. Note that this field is always '0b' if a Card doesn't support PTP mode.
RPD (Received PTP Data) specifies whether PTP data is received. This read-only bit shall be updated by a Card during the data-out phase after issuing of the "PTP_SetReceiveDataInformation" command with setting 'mode' to "00h". One of the following values shall be set.
0b: Not received (Default)
1b: Received
If a Host detects this bit is set to '1b', a Host is able to get the received data by issuing the

"PTP_ReceiveData" command.
Note that this field is changed to '0b' by the "PTP_ReceiveData" command. Note that when CRU (Command Response Update) is set to '1b' with the completion of the "PTP_ReceiveData" command, this field may be still set to '1b'. It means that the following data for the next "PTP_ReceiveData" command is ready. Note that this field is always '0b' if a Card doesn't support PTP mode.
RPC (Received PTP Cancel Data) specifies whether an initiator generated cancel is detected. A Host can know that a Card received Cancel packet via Command/Data connection or Event connection. (The trigger is only first one.) This read-only bit shall be updated by a Card while operating in DPS mode or PTP Pass-Through mode. (The start and end timing of this mode is managed by issuing of "PTP_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not detected (Default)
1b: Detected
If a Host detects this bit is set to '1b', a Host is able to get the TransactionID of cancelled transaction by reading "PTP Cancelled TransactionID".
Note that this field is changed to '0b' by setting "PTP Cancelled TransactionID" register to "FFFFFFFFh". Note that this field is always '0b' if a Card doesn't support PTP mode.
CPI (Connected PTP-IP) specifies whether PTP-IP connection is established. This read-only bit shall be updated by a Card while operating in each PTP mode. (The start and end timing of this mode is managed by issuing of "PTP_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not detected (Default)
1b: Detected
Note that this field is always '0b' if a Card doesn't support PTP mode.
DPI (Disconnected PTP-IP) specifies whether PTP-IP connection is lost. Especially before connection establishment, this field specifies that connection cannot be established. (e.g. network error) This read-only bit shall be updated by a Card while operating in each PTP mode. (The start and end timing of this mode is managed by issuing of "PTP_SwitchMode" command with appropriate argument which specifies target mode.) One of the following values shall be set.
0b: Not detected (Default)
1b: Detected
If a Host detects this bit is set to '1b', a Host shall exit from PTP mode by issuing the "PTP_SwitchMode" command. A Host can know the reason why this field is set to '1b' by reading "PTP Error Status".
Note that this field is always '0b' if a Card doesn't support PTP mode.
CIL (Changed Initiator List) specifies whether the initiator list is updated. This read-only bit shall be updated by a Card in the following situations.

1. In DPS mode. (The start and end timing of this mode is managed by issuing of "PTP_SwitchMode" command with appropriate argument which specifies target mode.) In this situation, this bit specifies about PTP Initiator List.
2. In PTP Pull mode and PTP Pass-Through mode, while the device discovery process is carried out. (Between the issuing of the "PTP_StartSearch" command, and the issuing of the "PTP_StartAdvertisement" command.) In this situation, this bit specifies about PTP Initiator List.
3. In PTP Pull mode and PTP Pass-Through mode, while particular advertisement process is carried out. (Between two times of the issuing of the "PTP_StartAdvertisement" command as described in the note of 4.7.18) In this situation, this bit specifies about PTP Rejected Initiator List.

One of the following values shall be set.
0b: Not updated (Default)
1b: Updated
If a Host detects this bit is set to '1b', a Host is able to get the updated list by issuing the

"PTP_GetInitiatorList" command (for the situations 1 and 2) or the "PTP_GetRejectedInitiatorList" command (for the situation 3).
Note that this field is changed to '0b' by the "PTP_GetInitiatorList" command and the "PTP_GetRejectedInitiatorList" command. Note that this field is always '0b' if a Card doesn't support PTP mode.

If either bit of this register changes from '0b' to '1b', a Card shall set ASU (Application Status Update) bit of iSDIO Status to '1b'. If ASU_ENA bit of iSDIO Int Enable is '1b', an interrupt shall be issued.

Application specifies current Application status. This value is an integer and one of the following values shall be set.
00h: No Application (Default)
01h: P2P Sender Application
02h: P2P Receiver Application
03h: DLNA M-DMU mode
04h: DLNA +PU+ mode
05h: DLNA M-DMS mode
06h: PTP Pull mode
07h: DPS mode
08h: PTP PassThrough mode
Note that the application values are exclusive.

WLAN specifies the current wireless LAN status as defined below. See Figure 2-3 as for the Wireless LAN State Diagram.

Connected specifies whether a Card has a wireless LAN connections with other devices. One of the following values shall be set.
0b: No Connection (Default)
1b: Connected
Note that in case of an AP, this value shall be '1b' if the AP connects to one or more STAs, and otherwise '0b'.

Infra-Direct specifies the wireless LAN connection scheme. One of the following values shall be set.
00b: Initial (Default)
01b: Infrastructure
10b: Wi-Fi Direct

AP-STA specifies whether a Card is in AP mode or STA mode in the Infrastructure connection. This value is valid only in case of Infrastructure (i.e. "Infra-Direct" is '01b'). One of the following values shall be set.
0b: STA (Default)
1b: AP

Group specifies whether a Card is the Group Owner or the Group Client in the Wi-Fi Direct connection. This value is valid only in case of Wi-Fi Direct (i.e. "Infra-Direct" is '10b'). One of the following values shall be set.
0b: Group Client (Default)
1b: Group Owner

WPS specifies a WPS mode. One of the following values shall be set.
00b: No WPS (Default)
01b: WPS with PIN
10b: WPS with PBC

Scan specifies whether a Card is scanning for wireless LAN APs. One of the following values shall be set.
0b: No Scan (Default)
1b: Scanning

SSID specifies SSID of an AP when a Card is in AP mode or SSID of an AP that a Card is connected to when the Card is in STA mode. The value is a string of upto 32 ASCII characters. If the length of the name is shorter than 32, the unused bytes are filled with "00h". Default value is all zero.
In the case of Wi-Fi Direct, SSID is created automatically through negotiation.

Encryption Mode specifies the network authentication and the data encryption for the current wireless LAN connection. This value is an integer and one of the following values shall be set.
00h: Open System and no encryption (Default)
01h: Open System and WEP
02h: Shared Key and WEP
03h: WPA-PSK and TKIP
04h: WPA-PSK and AES
05h: WPA2-PSK and TKIP
06h: WPA2-PSK and AES
Note that "06h" shall be set in the case of Wi-Fi Direct.

Signal Strength specifies the current signal strength in the percentage. This value is an integer and shall be "00h" (low) to "64h" (high) inclusive (0 to 100).

Channel specifies the current channel for wireless LAN. This value is an integer and one of the following values shall be set.
00h: No connection (Default)
01h to FFh: Channel number (e.g. 01h: Channel 1, 02h: Channel 2, etc.)

MAC Address specifies Mac Address of the iSDIO Wireless LAN Card. This value is a 6-byte integer (big endian). e.g. "00h", "11h", "22h", "AAh", "BBh" and "CCh" are stored in this order for "00-11-22-AA-BB-CC".

ID specifies ID of this Card for Peer-to-Peer File Transfer. This value is a string of upto 16 ASCII characters, and is able to be set by the "SetID" command or by the Configuration file. If the length of ID is shorter than 16 bytes, the unused bytes are filled with "00h". If this value is not configured, this value is filled with "00h".

IP Address specifies the current IP Address, and it may be different from what is set in the Configuration file if the DHCP is enabled. This value is a 4-byte integer (big endian). e.g. "C0h", "A8h", "00h" and "01h" are stored in this order for "192.168.0.1".

Subnet Mask specifies the current Subnet Mask, and it may be different from what is set in the Configuration file if the DHCP is enabled. This value is a 4-byte integer (big endian). e.g. "C0h", "A8h", "00h" and "01h" are stored in this order for "192.168.0.1".

Default Gateway specifies the current Default Gateway, and it may be different from what is set in the Configuration file if the DHCP is enabled. This value is a 4-byte integer (big endian). e.g. "C0h", "A8h", "00h" and "01h" are stored in this order for "192.168.0.1".

Preferred DNS Server specifies the current Preferred DNS Server, and it may be different from what is set in the Configuration file if the DHCP is enabled. This value is a 4-byte integer (big endian). e.g. "C0h", "A8h", "00h" and "01h" are stored in this order for "192.168.0.1".

Alternate DNS Server specifies the current Alternate DNS Server, and it may be different from that is set in the Configuration file if the DHCP is enabled. This value is a 4-byte integer (big endian). e.g. "C0h", "A8h", "00h" and "01h" are stored in this order for "192.168.0.1".

Proxy Server:

PSE (Proxy Server Enabled) specifies which Proxy Server specified in the Configuration file is enabled. One of the following values shall be set.
0b: Proxy is disabled (Default)
1b: Proxy is enabled

Date specifies the current date in a Card. A Host should set a starting value in a Card and the Card counts up from the specified value by the Card's internal clock. This value is an integer (little endian) and the format is as follows; Bits 0 to 4: Day of month, valid value range 1 to 31 inclusive. Bits 5 to 8: Month of year, 1 = January, valid value range 1 to 12 inclusive. Bits 9 to 15: Count of years from 1980, valid value range 0 to 127 inclusive (1980 to 2107). Default value is all zero.

Time specifies the current time in a Card. A Host should set a starting value in a Card and the Card counts up from the specified value by the Card's internal clock. This value is an integer (little endian) and the format is as follows; Bits 0 to 4: 2-second count, valid value range 0 to 29 inclusive (0 to 58 seconds). Bits 5 to 10: Minutes, valid value range 0 to 59 inclusive. Bits 11 to 15: Hours, valid value range 0 to 23 inclusive. The valid time range is from Midnight 00:00:00 to 23:59:58. Default value is all zero. Note that this field is read by a Host to confirm the value is set correctly in a Card, and this value may not be accurate enough to be used by a Host's application.

HTTP Status:

HTTP Progress specifies the progress of the download or the upload of HTTP messages in percentage. This value is an integer and shall be "00h" (Default) to "64h" inclusive (0 to 100). Note that it depends on a Card implementation how to calculate the value.
HPC (HTTP Processing) specifies whether a Card is processing (i.e. downloading or uploading) HTTP message(s) or not. One of the following values shall be set.
0b: No Processing (Default)
1b: Processing

Note that "SendHTTPMessageByRegister", "SendHTTPFileByRegister", "SendHTTPSSLMessageByRegister", "SendHTTPSSLFileByRegister", "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile", "SendHTTPSSLFileByFile", "DLNA_StartUpload", "DLNA_PushPlayback", "DLNA_AcceptUpload", "DLNA_AcceptDistribution" and "GetFile" commands affect to this "HTTP Status".

Power Save Management:

PSM (Power Save Mode) specifies whether a Card is in Power Save mode or not. One of the following values shall be set.
0b: Power Save Mode Off (Default)
1b: Power Save Mode On
Note that it is upto a Card implementation how to realize the power saving, and that if a Card doesn't support the power save mode, the Card shall set '0b'.

File System Management:

FIM (FS Information Modified) specifies whether the Host ensures the file system information would be modified or not. iSDIO Wireless LAN Card has file system function in it. In order to achieve this function, the Card has to check the file system information (Master Boot Record, Partition Table, Partition Boot Sector, FS Info of FAT32, and Boot Region of exFAT) and recognize some file system parameters (Sector Size, Cluster Size, Format Layout, etc.). This file system information may be modified by a format operation executed by Host. Therefore, if the Host can notify the timing of this modification to the Card, it may be helpful for the Card implementation and increase the Card performance. For that reason, FIM should be set as '1b', during the period when the Host can ensure the file system information would not be modified by format operation.

Other than that it shall always be set as ‘0b’. Using this bit is optional for Host. If the Host doesn’t use this bit, the Host shall keep this bit as ‘0b’.
0b: FS Information may be modified (Default). In this case, Host can execute format operation.
1b: FS Information shall not be modified. In this case, Host shall not execute format operation.
During the period of setting FIM as ‘1b’, the Card assumes the file system information would not be modified.
Note that if the Host cannot ensure the format operation timing, the Host shall not use this flag. And while Host’s file system access is prohibited by exclusive control specified in **6.1.3 Command Attribute for Exclusive Control** section, the Host shall not modify FIM.
Here, the file system information doesn’t include File Allocation Table, Allocation Bitmap and/or directory entries. This means that the Host can modify File Allocation Table, Allocation Bitmap and/or directory entries even if FIM is set as ‘1b’. Moreover, the following fields included in the file system information can be modified similarly even if FIM is set as ‘1b’.

- All reserved fields
- "Free Cluster Count" and "Next Free Cluster" in FS Info of FAT32
- "VolumeFlags" and "PercentInUse" in Boot Region of exFAT

Figure 3-1 shows an example of access sequence including FIM settings.

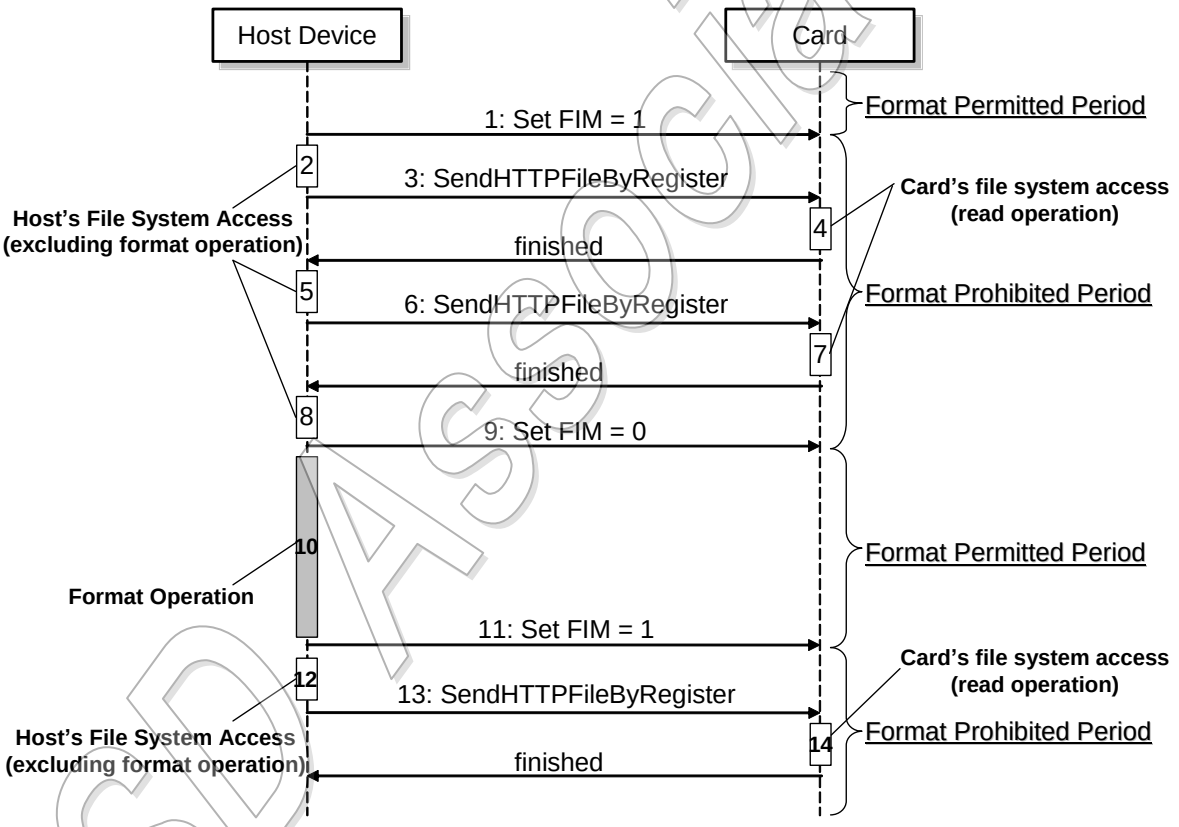


Figure 3-1 : Example of Access Sequence

By using FIM flag, Card can recognize "Format Permitted Period" and "Format Prohibited Period". Host shall not execute format operation during this "Format Prohibited Period". Therefore, if Host wants to format the Card, Host shall set FIM flag to 0 (step9) and set FIM flag to 1 again after format operation (step11).

Here, Card may assume that format operation shall not be executed and FS Information shall not

be modified during "Format Prohibited Period". Therefore, Card may not read and check FS Information at step7, because FS Information shall not be modified from step 4. This may reduce some flash memory access and achieve high performance operation in the Card. Note that Card should assume FS Information may have been modified at step4 and step14 though those are included in "Format Prohibited Period", because those are the first command after "Format Permitted Period".

The values not defined above are reserved.
Reserved fields shall be filled with "00h".

PTP Transfer Status:

PTP Progress specifies the progress of the PTP transfer in percentage. This value is an integer and shall be "00h" (Default) to "64h" inclusive (0 to 100). Note that it depends on a Card implementation how to calculate the value.

Note that the "PTP_SendFile" command and the "PTP_ReceiveFile" command affect to this "PTP Transfer Status".

PTP Cancelled TransactionID specifies the TransactionID specified in the initiator generated cancel. The value of this field is valid only when RPC bit of "PTP Status" is set to '1b'. This value is a 4-byte unsigned integer and one of the following values shall be set.
00000000h to FFFFFFFEh: Cancelled TransactionID
FFFFFFFFh: No cancellation (Default)
Note that even if a Host does nothing after confirmation of this register, a Host shall set this register to "FFFFFFFFh" immediately to clear RPC bit of "PTP Status". In addition, a Card shall store at least one cancellation to accept close cancellations.

PTP Error Status specifies the error status in the PTP-IP application. The value of this field is valid only when DPI bit of "PTP Status" is set to '1b'. This value is an integer and one of the following values shall be set.
00h: No error (Default)
01h: Wireless LAN error
02h: TCP/IP error
03h: PTP-IP Probe error
04h: Other PTP-IP error
Note that if a Card detects an error and cannot continue PTP-IP processing, a Card shall set this field to other than "00h". The detailed criteria depend on the implementation of each Card.

3.5 iSDIO Capability Register for Wireless LAN

Table 3-3 shows the structure of the iSDIO Capability Register for Wireless LAN. A Host is able to determine the supported functions in a Card that are optionally defined in this specification. Note that the area from "0600h" to "060Bh" is commonly defined in **2.2.2.4 iSDIO Capability Register** in [iSDIO].

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Type
0600h	*iSDIO Common Specification*	*Major Version Number*				*Minor Version Number*				
0601h	*iSDIO Application Specification*	*Major Version Number*				*Minor Version Number*				*R/O*
0602h	*Command Write Update Necessity*	-	-	-	-	-	-	-	*CWN*	*R/O*
0603h	*Command Response Status Queue Support*	-	-	-	*Number of Command Response Status in Queue*					*R/O*
0604h - 0607h	*Max Size of Command Write Data*									*R/O*
0608h - 060Bh	*Max Size of Command Response Data*									*R/O*
060Ch - 063Fh	*Reserved*									
0640h	WLAN Support	-	-	-	-	-	-	**5GHz**	**2.4 GHz**	R/O
0641h		-	-	-	-	**WPS_AP**	**WPS_STA**	**AP**	**WiFi Direct**	R/O
0642h	P2P File Transfer Support	-	-	-	-	-	-	-	**P2P**	R/O
0643h	DLNA Support	-	-	-	-	-	**DMS**	**DPU**	**DMU**	R/O
0644h	ByFile Support	-	-	-	-	-	-	-	**BFS**	R/O
0645h	Configuration File Support	-	-	-	-	-	-	-	**CFF**	R/O
0646h	Encryption Mode Support	-	-	-	-	-	-	-	-	R/O
0647h		-	-	**W2A**	**W2T**	**WPA**	**WPT**	**SWE**	**OWE**	R/O
0648h	2.4GHz Channel Support	-	-	**C14**	**C13**	**C12**	**C11**	**C10**	**C9**	R/O
0649h		**C8**	**C7**	**C6**	**C5**	**C4**	**C3**	**C2**	**C1**	R/O
064Ah	5GHz Channel Support	-	-	-	-	-	-	-	-	R/O
064Bh		-	**C161**	**C157**	**C153**	**C149**	**C140**	**C136**	**C132**	R/O
064Ch		**C128**	**C124**	**C120**	**C116**	**C112**	**C108**	**C104**	**C100**	R/O
064Dh		**C64**	**C60**	**C56**	**C52**	**C48**	**C44**	**C40**	**C36**	R/O
064Eh	Proxy Server Support	-	-	-	-	-	-	-	**PSS**	R/O
064Fh	PTP Support								**PTP**	R/O
0650h - 07BFh	Reserved									
07C0h - 07FFh	Reserved for Vendor									

Table 3-3 : iSDIO Capability Register for Wireless LAN

iSDIO Common Specification specifies the version number of the iSDIO Common Specification.
0001 0000b: iSDIO Specification Version 1.00

iSDIO Application Specification specifies the version number of the Wireless LAN Addendum.
0001 0000b: Wireless LAN Addendum Version 1.00
0001 0001b: Wireless LAN Addendum Version 1.10 (this version)

Command Write Update Necessity:
CWN: Refer to [iSDIO].

Command Response Status Queue Support:
Number of Command Response Status in Queue: Refer to [iSDIO].

Max Size of Command Write Data: Refer to [iSDIO].
This value is an integer. The value shall be equal to or more than 1024 if the "BFS" bit of the "ByFile Support" is set to '1b' and shall be equal to or more than 4096 if the "BFS" bit of the "ByFile Support" is set to '0b'.

Max Size of Command Response Data: Refer to [iSDIO].
This value is an integer. The value shall be equal to or more than 1024 if the "BFS" bit of the "ByFile Support" is set to '1b' and shall be equal to or more than 16384 if the "BFS" bit of the "ByFile Support" is set to '0b'.

WLAN Support:
2.4GHz specifies whether the 2.4GHz band is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
5GHz specifies whether the 5GHz band is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
WiFiDirect specifies whether the Wi-Fi Direct is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
AP specifies whether the AP of the Infrastructure is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
WPS_STA specifies whether the WPS for the STA is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
WPS_AP specifies whether the WPS for the AP is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

P2P File Transfer Support:
P2P specifies whether the Peer-to-Peer File Transfer defined in **2.5 Peer-to-Peer File Transfer** is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

DLNA Support:

DMU (M-DMU Support) specifies whether DLNA M-DMU is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

DPU (+PU+ Support) specifies whether DLNA +PU+ feature is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

DMS (M-DMS Support) specifies whether DLNA M-DMS is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

ByFile Support:

BFS specifies whether all of "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile", "SendHTTPSSLFileByFile" and "SetCertificateByFile" commands are supported. One of the following values shall be set.
0b: Not supported
1b: Supported

Configuration File Support:

CFF specifies whether Configuration File defined in **5. iSDIO Wireless LAN Configuration File**. One of the following values shall be set.
0b: Not supported
1b: Supported

Encryption Mode Support:

OWE (Open System and WEP) specifies whether the specified network authentication and data encryption are supported in the AP. One of the following values shall be set.
0b: Not supported
1b: Supported

SWE (Shared Key and WEP) specifies whether the specified network authentication and data encryption are supported in the AP. One of the following values shall be set.
0b: Not supported
1b: Supported

WPT (WPA-PSK and TKIP) specifies whether the specified network authentication and data encryption are supported in the AP. One of the following values shall be set.
0b: Not supported
1b: Supported

WPA (WPA-PSK and AES) specifies whether the specified network authentication and data encryption are supported in the AP. One of the following values shall be set.
0b: Not supported
1b: Supported

W2T (WPA2-PSK and TKIP) specifies whether the specified network authentication and data encryption are supported in the AP. One of the following values shall be set.
0b: Not supported
1b: Supported

W2A (WPA2-PSK and AES) specifies whether the specified network authentication and data encryption are supported in the AP. This value shall be ‘1b’ because it is mandated in this version.
Note that in the case of Wi-Fi Direct, only W2A mode is effective.

2.4GHz Channel Support:

C1 specifies whether Channel 1 is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
C2 specifies whether Channel 2 is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
...
C14 specifies whether Channel 14 is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

5GHz Channel Support:

C36 specifies whether Channel 36 is supported. One of the following values shall be set.
0b: Not supported
1b: Supported
...
C161 specifies whether Channel 161 is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

Proxy Server Support:

PSS specifies whether the Proxy Server configured by the Configuration File in **5. iSDIO Wireless LAN Configuration File** is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

PTP Support:

PTP specifies whether the PTP mode is supported. One of the following values shall be set.
0b: Not supported
1b: Supported

The values not defined above are reserved.
Reserved fields shall be filled with "00h".

3.6 iSDIO Response Data for Wireless LAN

This subsection defines the Response Data for Wireless LAN that is returned to a Host by the Command Response Data (See **2.2.2.3 iSDIO Command Response Data** in [iSDIO]) via the Response Data Register Port.

3.6.1 WPS List

Table 3-4 shows the structure of the WPS List which is the response to the "StartWPS" command. The WPS List contains both the SSID name and the Network Key of the connected AP. This data is stored in the iSDIO Command Response Data.

Size (byte)	Name	Short Description	Type
32	SSID	SSID of the AP	R/O
64	Network Key	Network Key for the AP	R/O

Table 3-4 : WPS List

SSID specifies the SSID of the AP. The value is a string of upto 32 ASCII characters, and if the length of the name is shorter than 32, the unused bytes are filled with "00h".

Network Key specifies the Network Key for the AP. This value is a string of upto 64 ASCII characters, and if the length of the key is shorter than 64, the unused bytes are filled with "00h".

3.6.2 Wireless LAN SSID List

Table 3-5 shows the structure of the Wireless LAN SSID List Data which is the response to the "Scan" command. The Wireless LAN SSID List Data contains the list of scanned SSIDs, MAC Addresses, Signal Strengths and Encryption Modes. This data is stored in the iSDIO Command Response Data.
Note that if the "Number of SSIDs" is "00h", this list consists of the 1-byte "Number of SSIDs" field and a 3-byte reserved field.

Size (byte)	Name	Short Description	Type
1	Number of SSIDs	Number of SSIDs in the SSID List	R/O
3	Reserved		R/O
32	SSID #1	SSID of the AP	R/O
6	MAC Address #1	MAC Address of the AP	R/O
1	Signal Strength #1	Signal Strength of the AP	R/O
1	Encryption Mode #1	Encryption Mode of the AP	R/O
4	Reserved		R/O
...	...	...	R/O
32	SSID #n_3		R/O
6	MAC Address #n_3		R/O
1	Signal Strength #n_3		R/O
1	Encryption Mode # n_3		R/O
4	Reserved		R/O

Table 3-5 : Wireless LAN SSID List

Number of SSIDs specifies the number of SSIDs of the scanned APs in the list.

SSID specifies the SSID of the scanned AP. The value is a string of upto 32 ASCII characters, and if the length of the name is shorter than 32, the unused bytes are filled with "00h".

MAC Address specifies MAC Address of the scanned AP. This value is a 6-byte integer (big endian). e.g. "00h", "11h", "22h", "AAh", "BBh" and "CCh" are stored in this order for "00-11-22-AA-BB-CC".

Signal Strength specifies the signal strength of the scanned AP as a percentage. This value is an integer and shall be "00h" (low) to "64h" (high) inclusive (0 to 100).

Encryption Mode specifies the network authentication and the data encryption of the scanned AP. This value is an integer and one of the following values shall be set.
00h: No encryption

01h: WEP
02h: WPA
03h: WPA2

The values not defined above are reserved.
Reserved fields shall be filled with "00h".

3.6.3 HTTP Response Data

Table 3-6 shows the structure of the HTTP Response Data which is the response to the "SendHTTPMessageByRegister", "SendHTTPFileByRegister", "SendHTTPSSLMessageByRegister" and "SendHTTPSSLFileByRegister" commands. The HTTP Response Data contains the HTTP response message containing the HTTP Response Line, the HTTP message header and the HTTP message body. This data is stored in the iSDIO Command Response Data.

Size (byte)	Name	Short Description	Type
variable	HTTP Response Data	The data of the HTTP response message	R/O

Table 3-6 : HTTP Response Data

HTTP Response Data contains the received HTTP response message containing the HTTP Response Line, the HTTP message header and the HTTP message body. This value is binary data.

3.6.4 DLNA Device List

Table 3-7 shows the structure of the DLNA Device List Data which is the response to the "DLNA_GetDeviceList" command of DLNA function. The DLNA Device List Data contains the list of UPnP devices detected during the UPnP device discovery. In DLNA M-DMU mode, this list includes DMS/M-DMS devices. In DLNA +PU+ mode, this list includes DMR devices. This data is stored in the iSDIO Command Response Data.
Note that if the "Number of devices" is "00h", this list consists of the 1-byte "Number of devices" field and a 3-byte reserved field.

Size (byte)	Name	Short Description	Type
1	Number of devices	Number of detected devices	R/O
3	Reserved		R/O
1	Length of UDN #1	(L_{UDN1})	R/O
3	Reserved		R/O
L_{UDN1}	UDN #1	UDN of device #1	R/O
0, 1, 2 or 3	Padding #U1	Padding size is either 0 (for L_{UDN1} mod4=0), 1 (for L_{UDN1} mod4=3), 2 (for L_{UDN1} mod4=2), 3 (for L_{UDN1} mod4=1)	R/O
1	Length of friendlyName #1	(L_{fN1})	R/O
3	Reserved		R/O
L_{fN1}	friendlyName #1	friendlyName of device #1	R/O
0, 1, 2 or 3	Padding #F1	Padding size is either 0 (for L_{fN1} mod4=0), 1 (for L_{fN1} mod4=3), 2 (for L_{fN1} mod4=2), 3 (for L_{fN1} mod4=1)	R/O
...	...	...	R/O
1	Length of UDN #n	(L_{UDNn})	R/O
3	Reserved		R/O
L_{UDNn}	UDN #n	UDN of device #n	R/O
0, 1, 2 or 3	Padding #Un	Padding size is either 0 (for L_{UDNn} mod4=0), 1 (for L_{UDNn} mod4=3), 2 (for L_{UDNn} mod4=2), 3 (for L_{UDNn} mod4=1)	R/O
1	Length of friendlyName #n	(L_{fNn})	R/O
3	Reserved		R/O
L_{fNn}	friendlyName #n	friendlyName of device #n	R/O
0, 1, 2 or 3	Padding #Fn	Padding size is either 0 (for L_{fNn} mod4=0), 1 (for L_{fNn} mod4=3), 2 (for L_{fNn} mod4=2), 3 (for L_{fNn} mod4=1)	R/O

Table 3-7 : DLNA Device List

Number of devices specifies the number of UPnP Devices in the list.

Length of UDN specifies the length of the UDN in bytes. The value is in 1-byte integer, and should be 5 to 68 inclusive.

UDN specifies UDN of a device. The value is string of ASCII characters and it is not terminated with null character.

Length of friendlyName specifies the length of the friendlyName in bytes. The value is in 1-byte unsigned integer, and should be set from 0 to 252.

friendlyName specifies the friendlyName of a device. The value is string encoded in UTF-8 and it is not terminated with null character.

Reserved fields shall be filled with "00h".
Padding bytes shall be filled with "00h".

3.6.5 DLNA Upload Information Data

Table 3-8 shows the structure of the DLNA Upload Information Data which is the response to the "DLNA_GetUploadInformation" DLNA command. The DLNA Upload Information Data contains the information of the content upload requested. The values are received as the metadata of CreateObject request. This data is stored in the iSDIO Command Response Data.

Size (byte)	Name	Short Description	Type
8	Size	Size of the content	R/O
2	Length of Title	(L_{Title})	R/O
2	Reserved		R/O
L_{Title}	Title	Title of the content	R/O
0, 1, 2 or 3	Padding	Padding size is either 0 (for L_{Title} mod4=0), 1 (for L_{Title} mod4=3), 2 (for L_{Title} mod4=2), 3 (for L_{Title} mod4=1)	R/O
2	Length of MIME-Type	(L_{Type})	R/O
2	Reserved		R/O
L_{Type}	MIME-Type	MIME-Type of the content	R/O
0, 1, 2 or 3	Padding	Padding size is either 0 (for L_{Type} mod4=0), 1 (for L_{Type} mod4=3), 2 (for L_{Type} mod4=2), 3 (for L_{Type} mod4=1)	R/O
2	Length of profileID	(L_{ID})	R/O
2	Reserved		R/O
L_{ID}	profileID	profileID of the content	R/O
0, 1, 2 or 3	Padding	Padding size is either 0 (for L_{ID} mod4=0), 1 (for L_{ID} mod4=3), 2 (for L_{ID} mod4=2), 3 (for L_{ID} mod4=1)	R/O

Table 3-8 : DLNA Upload Information

Size specifies the size of the content. The value is in 8-byte unsigned integer. A Card shall get this value as res@size metadata in CreateObject request. If there is no res@size in the request, a Card shall use "FF FF FF FF FF FF FF FFh" (Maximum Value) for this value.

Length of Title specifies the length of Title in bytes. The value is a 2-byte integer.

Title specifies the title of the content. The value is a string encoded in UTF-16LE and it is not terminated with null character. To create this data, a Card shall convert dc:title metadata of CreateObject request from UTF-8 to UTF-16LE.

Length of MIME-Type specifies the length of MIME-Type in bytes. The value is a 2-byte integer.

MIME-Type specifies the MIME-Type of content. The value is a string of ASCII characters and it is not terminated with null character. A Card shall get this value from res@protocolInfo metadata in CreateObject request. (e.g. "image/jpeg")

Length of profileID specifies the length of profileID in bytes. The value is a 2-byte integer.

profileID specifies profileID of the content. The value is a string of ASCII characters and it is not terminated with null character. A Card shall get this value from res@protocolInfo metadata in CreateObject request. (e.g. "JPEG_SM")

Reserved fields shall be filled with "00h".
Padding bytes shall be filled with "00h".

Note that details of CreateObject and metadata are described in [DLNA].

3.6.6 Wireless LAN ID List

Table 3-9 shows the structure of the Wireless LAN ID List Data which is the response to the "ReadIDList" command for Peer-to-Peer File Transfer. The Wireless LAN ID List Data contains the list of Receiver Card IDs which request to connect the Sender Card. This data is stored in the iSDIO Command Response Data.
Note that the Receiver ID is removed from this list when a Host requests to deselect a Receiver ID in this list by the "DeselectMAC" command.
Also note that if the "Number of Receiver IDs" is "00h", this list consists of the 1-byte "Number of Receiver IDs" field and a 3-byte reserved field.

Size (byte)	Name	Short Description	Type
1	Number of Receiver IDs	Number of Receiver Card IDs in the list	R/O
3	Reserved		R/O
16	Receiver ID #1	ID of a Receiver Card which requests to connect to the Sender Card	R/O
6	Receiver MAC Address #1	MAC Address of a Receiver Card which requests to connect to the Sender Card	R/O
1	Receiver Permission #1	Receiver's Access Permission	R/O
1	Reserved		R/O
...	...	...	R/O
16	Receiver ID #n_4		R/O
6	Receiver MAC Address # n_4		R/O
1	Receiver Permission # n_4		R/O
1	Reserved		R/O

Table 3-9 : Wireless LAN ID List

Number of Receiver IDs specifies the number of Receiver Card IDs in the list. This value is an integer and shall be 0 to 10 inclusive.

Receiver ID specifies ID of a Receiver Card which requests to connect to the Sender Card. The value is a string of upto 16 ASCII characters and if the length of the name is less than 16, the unused bytes are filled with "00h".

Receiver MAC Address specifies MAC Address of a Receiver Card which requests to connect to the Sender Card. This value is a 6-byte integer (big endian). e.g. "00h", "11h", "22h", "AAh", "BBh" and "CCh" are stored in this order for "00-11-22-AA-BB-CC".

Receiver Permission specifies whether the Receiver Card is permitted to access the Sender Card. This value is an integer and one of the following values shall be set.
00h: Not Permitted (Default)
01h: Access Permitted

The values not defined above are reserved.
Reserved fields shall be filled with "00h".

3.6.7 PTP Information Data

Table 3-10 shows the structure of the PTP Information Data which is the response to the "PTP_ReceiveOperation" command and the "PTP_SetReceiveDataInformation" command of PTP function. The PTP Information Data contains the payload of a PTP-IP packet (Operation Request or Start Data). This data is stored in the iSDIO Command Response Data.

Size (byte)	Name	Short Description	Type
variable	Payload	Payload of the received PTP-IP packet	R/O

Table 3-10 : PTP Information Data

Payload specifies the payload of the received PTP-IP packet. A Card shall remove the PTP-IP header ("Length" and "Packet Type") from the PTP-IP packet to generate the payload.

3.6.8 PTP Received data

Table 3-11 shows the structure of the PTP Received Data which is the response to the "PTP_ReceiveData" command of PTP function. The PTP Received Data contains the PTP data received in a data-out phase. This data is stored in the iSDIO Command Response Data.

Size (byte)	Name	Short Description	Type
variable	Data	Received PTP Data or partial PTP data	R/O

Table 3-11 : PTP Received Data

Data specifies the PTP data or partial PTP data that received in the data-out phase. A Card shall interpret the PTP-IP Data packets to reconstruct the PTP data.

Note that a Card can divide the PTP data into multiple partial data depending on limitation of the "Max Size of Command Response Data". Therefore, a Host may be required to issue the "PTP_ReceiveData" command multiple times. Note that a Host can know the total size of reconstructed data from the response data of the "PTP_SetReceiveDataInformation" command.

3.6.9 PTP Initiator List

Table 3-12 shows the structure of the PTP Initiator List Data which is the response to the "PTP_GetInitiatorList" command of PTP function. The PTP Initiator List Data contains the list of initiators detected during the UPnP device discovery in PTP mode. In DPS mode, the list contains DPS printers. In PTP Pull mode or PTP Pass-Through mode the list contains devices which a Host specified by the "PTP_StartSearch" command. This data is stored in the iSDIO Command Response Data.
Note that if the "Number of initiators" is "00h", this list consists of the 1-byte "Number of initiators" field and a 3-byte reserved field.

Size (byte)	Name	Short Description	Type
1	Number of initiators	Number of detected initiators	R/O
3	Reserved		R/O
1	Length of UDN #1	(L_{UDN1})	R/O
3	Reserved		R/O
L_{UDN1}	UDN #1	UDN of initiator #1	R/O
0, 1, 2 or 3	Padding #U1	Padding size is either 0 (for L_{UDN1} mod4=0), 1 (for L_{UDN1} mod4=3), 2 (for L_{UDN1} mod4=2), 3 (for L_{UDN1} mod4=1)	R/O
1	Length of friendlyName #1	(L_{fN1})	R/O
3	Reserved		R/O
L_{fN1}	friendlyName #1	friendlyName of initiator #1	R/O
0, 1, 2 or 3	Padding #F1	Padding size is either 0 (for L_{fN1} mod4=0), 1 (for L_{fN1} mod4=3), 2 (for L_{fN1} mod4=2), 3 (for L_{fN1} mod4=1)	R/O
...	...	...	R/O
1	Length of UDN #n	(L_{UDNn})	R/O
3	Reserved		R/O
L_{UDNn}	UDN #n	UDN of initiator #n	R/O
0, 1, 2 or 3	Padding #Un	Padding size is either 0 (for L_{UDNn} mod4=0), 1 (for L_{UDNn} mod4=3), 2 (for L_{UDNn} mod4=2), 3 (for L_{UDNn} mod4=1)	R/O
1	Length of friendlyName #n	(L_{fNn})	R/O
3	Reserved		R/O
L_{fNn}	friendlyName #n	friendlyName of initiator #n	R/O
0, 1, 2 or 3	Padding #Fn	Padding size is either 0 (for L_{fNn} mod4=0), 1 (for L_{fNn} mod4=3), 2 (for L_{fNn} mod4=2), 3 (for L_{fNn} mod4=1)	R/O

Table 3-12 : PTP Initiator List

Number of initiators specifies the number of initiators in the list.

Length of UDN specifies the length of the UDN in bytes. The value is in 1-byte integer, and should be 5 to 68 inclusive.

UDN specifies UDN of an initiator. The value is string of ASCII characters and it is not terminated with null character. e.g. "uuid:00000000-0000-0000-0000-000000000000".
Note that if a Host applies this value for the "PTP_StartAdvertisement" command as the GUID, the Host shall convert this value to an array of 1-byte unsigned integer in the responsibility of the Host application.

Length of friendlyName specifies the length of the friendlyName in bytes. The value is in 1-byte unsigned integer, and should be set from 0 to 252.

friendlyName specifies the friendlyName of an initiator. The value is string encoded in UTF-8 and it is not terminated with null character.

Reserved fields shall be filled with "00h".

Padding bytes shall be filled with "00h".

Note that this list consists of only information obtained in the UPnP device discovery process. On the other hand, the PTP Rejected Initiator List described below consists of only information obtained via the PTP-IP connection establishment sequence.

3.6.10 PTP Rejected Initiator List

Table 3-13 shows the structure of the PTP Rejected Initiator List Data which is the response to the "PTP_GetRejectedInitiatorList" command of PTP function. The PTP Rejected Initiator List Data contains the list of initiators rejected by the Card after issuing of the "PTP_StartAdvertisement" command with setting 'reject' to "01h". This data is stored in the iSDIO Command Response Data. Note that if the "Number of initiators" is "00h", this list consists of the 1-byte "Number of initiators" field and a 3-byte reserved field.

Size (byte)	Name	Short Description	Type
1	Number of initiators	Number of rejected initiators	R/O
3	Reserved		R/O
16	GUID #1	GUID of initiator #1	R/O
1	Length of Friendly Name #1	(L_{FN1})	R/O
3	Reserved		R/O
L_{FN1}	Friendly Name #1	Friendly Name of initiator #1	R/O
0, 1, 2 or 3	Padding #F1	Padding size is either 0 (for L_{FN1} mod4=0), 1 (for L_{FN1} mod4=3), 2 (for L_{FN1} mod4=2), 3 (for L_{FN1} mod4=1)	R/O
...	...	...	R/O
16	GUID #n	GUID of initiator #n	R/O
1	Length of Friendly Name #n	(L_{FNn})	R/O
3	Reserved		R/O
L_{FNn}	Friendly Name #n	Friendly Name of initiator #n	R/O
0, 1, 2 or 3	Padding #Fn	Padding size is either 0 (for L_{FNn} mod4=0), 1 (for L_{FNn} mod4=3), 2 (for L_{FNn} mod4=2), 3 (for L_{FNn} mod4=1)	R/O

Table 3-13 : PTP Rejected Initiator List

Number of initiators specifies the number of initiators in the list.

GUID specifies GUID of an initiator. The value is 16-byte array of unsigned integer. e.g. "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00h".

Length of Friendly Name specifies the length of the Friendly Name in bytes. The value is in 1-byte integer, and should be set from 0 to 78.

Friendly Name specifies the Friendly Name of an initiator. The value is string encoded in UTF-16LE and it is not terminated with null character.

Reserved fields shall be filled with "00h".
Padding bytes shall be filled with "00h".

Note that this list consists of only information obtained via the PTP-IP connection establishment sequence. On the other hand, the PTP Initiator List described above consists of only information obtained in the UPnP device discovery process.

4. iSDIO Wireless LAN Commands

4.1 Overview

This chapter defines the iSDIO Wireless LAN commands, which are registered in the iSDIO Command Write Data via the Command Write Register Port from a Host to a Card. The commands are categorized below depending on usage.

- WLAN Commands for the connection to the wireless LAN
- Common Commands for the management of the Card etc.
- Server Upload Commands for the sending and receiving HTTP messages
- DLNA Commands for DLNA File Transfer
- P2P File Transfer Commands for the Peer-to-Peer File Transfer Application
- PTP Commands for PTP Object Transfer

Table 4-1 shows all commands defined in this specification and their details are defined in the following sections.

Command ID	Command Name	Short Description	Ref.
0000h	Reserved		
0001h	Scan()	Scan wireless LAN and get the list of available SSIDs	4.2.1
0002h	Connect(ssid, networkKey)	Connect to wireless LAN AP as STA	4.2.2
0003h	Establish(ssid, networkKey, encMode)	Establish wireless LAN as AP	4.2.3
0004h	WiFiDirect(wpsMode, pin)	Connect via Wi-Fi Direct	4.2.4
0005h	StartWPS(ssid, wpsMode, pin)	Start WPS for STA and connect to wireless LAN AP	4.2.5
0006h	StartWPSAP(wpsMode, pin)	Start WPS for AP *(This command is issued when the Establish() command is being executed)*	4.2.6
0007h	Disconnect()	Terminate the application and disconnect wireless LAN	4.2.7
0008h - 0010h	Reserved		
0011h	SetCurrentTime(currentDate, currentTime)	Set the current time from a Host to a Card	4.3.1
0012h	Abort(sequenceID)	Abort the process of the specified iSDIO command	4.3.2
0013h	ReadResponse(sequenceID)	Read the specified Response Data from the Response Data Register Port	4.3.3
0014h	SetPowerSaveMode(powerMode)	Set the Power Save mode to a Card (on/off)	4.3.4
0015h	SetChannel(channelNum)	Set the channel to a Card for AP	4.3.5
0016h - 0020h	Reserved		
0021h	SendHTTPMessageByRegister(hostName, message)	Send an HTTP message by the Command Write Data	4.4.1
0022h	SendHTTPFileByRegister(hostname, appendFileName, message)	Send an HTTP message by the Command Write Data appending a file	4.4.2
0023h	SendHTTPSSLMessageByRegister(host	Send an HTTP message by the Command	4.4.3

	Name, message)	Write Data via SSL	
0024h	SendHTTPSSLFileByRegister(hostName, appendFileName, message)	Send an HTTP message by the Command Write Data via SSL appending a file	4.4.4
0025h	SendHTTPMessageByFile(hostName, messageFileName, headerRemoval)	Send an HTTP message stored in a message file	4.4.5
0026h	SendHTTPFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Send an HTTP message stored in a message file appending a data file	4.4.6
0027h	SendHTTPSSLMessageByFile (hostName, messageFileName, headerRemoval)	Send an HTTP message stored in a message file via SSL	4.4.7
0028h	SendHTTPSSLFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Send an HTTP message stored in a message file via SSL appending a data file	4.4.8
0029h	SetCertificate(certificate)	Set a root certificate for HTTPS connections	4.4.9
002Ah	SetCertificateByFile(certificateFileName)	Set a root certificate file for HTTPS connections	4.4.10
002Bh - 0060h	Reserved		
0061h	DLNA_SetDeviceInformation(parameterID, parameterValue)	Set device information of M-DMS from a Host to a Card	4.5.1
0062h	DLNA_SwitchMode(modeID)	Switch DLNA mode	4.5.2
0063h	DLNA_GetDeviceList()	Get a device list from a M-DMU Card to a Host	4.5.3
0064h	DLNA_StartUpload(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	Start a content transfer for M-DMS or DMS	4.5.4
0065h	DLNA_PushPlayback(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	Notify DMR to start playback	4.5.5
0066h	DLNA_GetUploadInformation()	Get the information of upload content	4.5.6
0067h	DLNA_AcceptUpload(accept, targetFileName)	Send a target filename from a Host to M-DMS Card	4.5.7
0068h	DLNA_SetUpdateID(valid, UpdateID)	Set UpdateID for M-DMS	4.5.8
0069h	DLNA_AcceptBrowse(accept)	Respond to a current Browse request	4.5.9
006Ah	DLNA_AcceptDistribution(accept)	Start a content transfer for M-DMP or DMP	4.5.10
006Bh - 0080h	Reserved		
0081h	StartP2PSender(ssid, networkKey, encMode)	Establish wireless LAN and start the P2P Sender application.	4.6.1
0082h	StartP2PReceiver(ssid, networkKey)	Connect to wireless LAN and start the P2P Receiver application.	4.6.2
0083h	GetFile(requestFileName, saveFileName)	Get a file from the Sender Card and store it in the Receiver Card	4.6.3
0084h	ReadIDList()	Read the ID List of the Receiver Cards from the Response Data Register Port	4.6.4
0085h	SelectMAC(mac)	Select the specified MAC Address of a Receiver Card which is allowed to connect the Sender Card	4.6.5
0086h	DeselectMAC(mac)	Deselect the specified MAC Address of a Receiver Card which is not allowed to	4.6.6

		connect the Sender Card	
0087h	SetID(id)	Set the ID of the Receiver Card	4.6.7
0088h -00FFh,	Recerved		
0100h	PTP_SetDeviceInformation(parameterID, parameterValue)	Set information of the PTP device	4.7.1
0101h	PTP_SetHiddenObjectType(objectTypeList)	Set the object types to hide	4.7.2
0102h	PTP_SetHiddenDirectory(directoryList)	Set the directories to hide	4.7.3
0103h	PTP_SwitchMode(modeID)	Switch PTP mode	4.7.4
0104h	PTP_ReceiveOperation()	Get an Operation Request packet	4.7.5
0105h	PTP_SendResponse(data)	Send an Operation Response packet	4.7.6
0106h	PTP_SetSendDataInformation(mode, TransactionID, TotalDataLength)	Set up the data-in phase	4.7.7
0107h	PTP_SendData(data)	Send the PTP data	4.7.8
0108h	PTP_SendFile(targetFileName)	Send the PTP data from a file	4.7.9
0109h	PTP_SetReceiveDataInformation(mode)	Set up the data-out phase	4.7.10
010Ah	PTP_ReceiveData()	Receive the PTP data	4.7.11
010Bh	PTP_ReceiveFile(targetFileName)	Store the PTP data as a file	4.7.12
010Ch	PTP_CancelData()	Cancel the current PTP transfer	4.7.13
010Dh	PTP_SendEvent(data)	Send an Event packet	4.7.14
010Eh	PTP_StartSearch(target)	Start the search of initiators	4.7.15
010Fh	PTP_GetInitiatorList()	Get a list of the initiators	4.7.16
0110h	DPS_ConnectPrinter(targetUDN)	Connect to a DPS printer	4.7.17
0111h	PTP_StartAdvertisement(reject, GUID)	Start the advertisement to initiators	4.7.18
0112h	PTP_GetRejectedInitiatorList()	Get a list of the rejected initiators	4.7.19
0113h -DFFFh	Reserved		
E000h - FFFFh	Reserved for Vendors		

Table 4-1 : iSDIO Wireless LAN Command List

4.2 WLAN Commands

4.2.1 Scan()

A Host requests to scan connectable wireless LAN. The connectable SSID list (SSID List) is available from the Response Data Register Port as a result of the command process.

4.2.1.1 Argument

None

4.2.1.2 Response Data

The SSID List is available from the Response Data of the Command Response Data (See **2.2.2.3 iSDIO Command Response Data** in [iSDIO]). The format of the SSID List is defined in **3.6.2 Wireless LAN SSID List**.

4.2.2 Connect(ssid, networkKey)

A Host requests to connect the specified AP in the wireless LAN as the STA. The command process is finished when the Card has connected the AP successfully.

4.2.2.1 Argument

4.2.2.1.1 Argument 1

Name: ssid
Type: ASCII character
Size: Variable
Description: The 'ssid' is the SSID of the AP to be connected. This value is not terminated by a null character.

4.2.2.1.2 Argument 2

Name: networkKey
Type: ASCII character
Size: Variable
Description: The 'networkKey' is the network key of the corresponding SSID. This value is not terminated by a null character.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") if encryption is not used for the WLAN connection.

4.2.2.2 Response Data

None

4.2.3 Establish(ssid, networkKey, encMode)

A Host requests to establish wireless LAN as the AP. A Card shall be the DHCP server, and shall handle the HTTP request/response messages defined in **Appendix C** in accordance with the sequence defined in **D.4 Peer-to-Peer File Transfer Sequence** with the Card's accepting any other Card's request (without the Host's operation). The command process is finished when the Card has initiated the AP successfully or an error.

4.2.3.1 Argument

4.2.3.1.1 Argument 1

Name: ssid
Type: ASCII character
Size: Variable
Description: The 'ssid' is the name of SSID to be established. This value is not terminated by a null character.

4.2.3.1.2 Argument 2

Name: networkKey
Type: ASCII character
Size: Variable
Description: The 'networkKey' is the network key of the corresponding SSID. This value is not

terminated by a null character.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") if encryption is not used for the WLAN connection.

4.2.3.1.3 Argument 3

Name: encMode
Type: integer
Size: 1-byte
Description: The 'encMode' denotes the network authentication and the data encryption scheme for the AP.
00h: Open System and no encryption
01h: Open System and WEP (Optional for a Card)
02h: Shared Key and WEP (Optional for a Card)
03h: WPA-PSK and TKIP (Optional for a Card)
04h: WPA-PSK and AES (Optional for a Card)
05h: WPA2-PSK and TKIP (Optional for a Card)
06h: WPA2-PSK and AES (Mandatory for a Card)
Note that supported combinations of network authentication and data encryption depend on the Card implementation, and a Host needs to check the "Encryption Mode Support" in the Capability Register. See **3.5 iSDIO Capability Register for Wireless LAN**.

4.2.3.2 Response Data

None

4.2.4 WiFiDirect(wpsMode, pin)

A Host requests to connect other devices via the Wi-Fi Direct. All Wi-Fi Direct devices are able to operate as either a device or an access point. The Wi-Fi Direct devices negotiate when they first connect to determine which device acts as an access point. The command process is finished when the Card has connected the Wi-Fi Direct device successfully. This command doesn't require the additional "Connect" command.

4.2.4.1 Argument

4.2.4.1.1 Argument 1

Name: wpsMode
Type: integer
Size: 1-byte
Description: The 'wpsMode' denotes the WPS mode.
01h: WPS with PIN
02h: WPS with PBC

4.2.4.1.2 Argument 2

Name: pin
Type: Numeric character
Size: 4-byte or 8-byte
Description: The 'pin' is the pin code for WPS with PIN.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") and even if set this value is ignored for the WPS with PBC.

4.2.4.2 Response Data

None

4.2.5 StartWPS(ssid, wpsMode, pin)

A Host requests to start the WPS with PIN method or with PBC method to the AP in the wireless LAN and connect the AP. The command process is finished when the Card has connected the AP successfully after the WPS process. This command doesn't require the additional "Connect" command.

4.2.5.1 Argument

4.2.5.1.1 Argument 1

Name: ssid
Type: ASCII character
Size: Variable
Description: The 'ssid' is the SSID of the AP to be connected. This value is not terminated by a null character.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") and even if set this value is ignored for the WPS with PBC.

4.2.5.1.2 Argument 2

Name: wpsMode
Type: integer
Size: 1-byte
Description: The 'wpsMode' denotes the WPS mode.
01h: WPS with PIN
02h: WPS with PBC

4.2.5.1.3 Argument 3

Name: pin
Type: Numeric character
Size: 4-byte or 8-byte
Description: The 'pin' is the pin code for WPS when using PIN.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") and even if set this value is ignored for the WPS with PBC.

4.2.5.2 Response Data

The 32-byte SSID in ASCII character and the 64-byte network key in ASCII character (total of 96 bytes) of the connected AP are available in this order as WPS List from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

4.2.6 StartWPSAP(wpsMode, pin)

A Host requests to start the WPS with PIN method or with PBC method to the STA in the wireless LAN. The command process is finished when the Card finished the WPS process to the STA. This command should be issued after the "Establish" command is issued (i.e. after an AP is initiated by a Card).

4.2.6.1 Argument

4.2.6.1.1 Argument 1

Name: wpsMode
Type: integer
Size: 1-byte
Description: The 'wpsMode' denotes the WPS mode.
01h: WPS with PIN

02h: WPS with PBC

4.2.6.1.2 Argument 2

Name: pin
Type: Numeric character
Size: 4-byte or 8-byte
Description: The 'pin' is the pin code for WPS when using a PIN.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") and even if set this value is ignored for the WPS with PBC.

4.2.6.2 Response Data

None

4.2.7 Disconnect()

A Host requests to disconnect or terminate the AP and terminate the current Application if it exists. In addition, current command processing requiring the WLAN connection is terminated.
If the P2P Receiver Application is running, a Card shall send the HTTP request message to notify the termination (HTTP(END)) defined in **C.2.6 HTTP(END)**.
The command process is finished when the Card has disconnected or terminated the AP.

4.2.7.1 Argument

None

4.2.7.2 Response Data

None

4.3 Common Commands

4.3.1 SetCurrentTime(currentDate, currentTime)

A Host requests to set the current time to a Card. The Card counts up from the specified value by the Card's internal clock, and the time is used for a Card to set the time stamp of a file or a directory when the Card creates, modifies or accesses it.

4.3.1.1 Argument

4.3.1.1.1 Argument 1

Name: currentDate
Type: integer (little endian)
Size: 2-byte
Description: The 'currentDate' is the date to be set to a Card. The format is as follows; Bits 0 to 4: Day of month, valid value range 1 to 31 inclusive. Bits 5 to 8: Month of year, 1 = January, valid value range 1 to 12 inclusive. Bits 9 to 15: Count of years from 1980, valid value range 0 to 127 inclusive (1980 to 2107).

4.3.1.1.2 Argument 2

Name: currentTime
Type: integer (little endian)
Size: 2-byte
Description: The 'currentTime' is the time to be set to a Card. The format is as follows; Bits 0 to 4: 2-second count, valid value range 0 to 29 inclusive (0 to 58 seconds). Bits 5 to 10: Minutes, valid value range 0 to 59 inclusive. Bits 11 to 15: Hours, valid value range 0 to 23 inclusive. The

valid time range is from Midnight 00:00:00 to 23:59:58.

4.3.1.2 Response Data
None

4.3.2 Abort(sequenceID)
A Host requests to terminate the specified iSDIO command processing. When a Card receives this command, the Card should behave as if the current processed command was not issued. However, as for a command which creates a file in the NAND memory module (i.e. "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile", "SendHTTPSSLFileByFile", "DLNA_AcceptUpload", "GetFile" and "PTP_ReceiveFile" commands), the Card doesn't have to remove the file even though the process is not completely finished. This means the file may not have all of the data to be stored. Therefore, a Host should treat this incomplete file appropriately after abort process. Moreover, note that the Card shall keep the consistency of file system metadata (File Allocation Table, Directory Entry, etc.).
This command is processed prior to the current processed command if accepted, and the Command Response Status for this command is not registered in the Command Response Status Queue but Command Response Status for the aborted command is changed to "04h: Process Terminated". That is, any command statuses are not removed from the queue though this command is issued.
If the multiple commands are processed and one command is aborted, the other commands are not affected and continue to be processed.

Depending on an issued timing of this command, this command may be ignored and the target command to be aborted may be processed completely. That is, Command Response Status for the target command may become "03h: Process Succeeded" or "80h to FFh: Process Failed". Note that processing time of "Abort" command shall be kept even in this case.

Note that it is strongly recommended to use "Disconnect" command to terminate the wireless LAN connection or the Peer-to-Peer File Transfer Application.

4.3.2.1 Argument

4.3.2.1.1 Argument 1
Name: sequenceID
Type: integer
Size: 4-byte
Description: The 'sequenceID' is the iSDIO command sequence id of the command, which is specified by a Host when the command is issued. This value shall be "00 00 00 00h" to "FF FF FF FFh".

4.3.2.2 Response Data
None

4.3.3 ReadResponse(sequenceID)
A Host requests to read a Response Data corresponding to the issued command by specifying the iSDIO command sequence id. The Response Data is available from the Response Data Register Port unless the buffer is overflow. This command is valid only when the specified command has been processed successfully and is in the Command Response Status Queue.
In addition, this command is processed prior to the current processed command, and the Command

Response Status for this command is not registered in the Command Response Status Queue. That is, any command statuses are not removed from the queue though this command is issued.

4.3.3.1 Argument

4.3.3.1.1 Argument 1

Name: sequenceID
Type: integer
Size: 4-byte
Description: The 'sequenceID' is the iSDIO command sequence id of the command, which is specified by a Host when the command is issued. This value shall be "00 00 00 00h" to "FF FF FF FFh".

4.3.3.2 Response Data

The Response Data depending on the specified iSDIO command (See **2.2.2.3 iSDIO Command Response Data** in [iSDIO]).

Note that by default the Response Data for the Command Response Status #1 in the first command in the Command Response Status Queue is mapped to the Response Data Register Port. And as for the other Command Response Statuses (#2 to #8), this command needs to be issued to map the specified Response Data to the Response Register Port (If a Host issues only one iSDIO command in the Command Write Data without using the queue scheme, the Host is unnecessary to use this command). This command is also able to be issued to map the Response Data for the Command Response Status #1.
Also note that even though this command is issued before the reading of the current Response Data is completed, the new Response Data is able to be read from the beginning of the data. That is, if the same command as current one is specified by this command before the reading of the current Response Data is completed, the Response Data is able to be read from the beginning.

4.3.4 SetPowerSaveMode(powerMode)

A Host requests to change the Power Save mode of a Card. If the mode is 'on', a Card works in Power Save mode (It depends on a Card implementation how to realize the power saving in a Card however some other performance such as the data transfer throughput may become lower in this mode depending on the Card).

4.3.4.1 Argument

4.3.4.1.1 Argument 1

Name: powerMode
Type: integer
Size: 1-byte
Description: The 'powerMode' denotes the Power Save mode.
00h: Power Save Mode Off
01h: Power Save Mode On

4.3.4.2 Response Data

None

4.3.5 SetChannel(channelNum)

A Host requests to set the channel in the AP to a Card.

Note that if a Card doesn't support the selected channel, this command is rejected. A Host needs to check the "Channel Support" in the Capability Register. See **3.5 iSDIO Capability Register for Wireless LAN**.

4.3.5.1 Argument

4.3.5.1.1 Argument 1

Name: channelNum
Type: integer
Size: 1-byte
Description: The 'channelNum' denotes the number of the Channel.
00h: Channel number is set automatically
01h: Channel 1
...
0Eh: Channel 14
24h: Channel 36
...
A1h: Channel 161

4.3.5.2 Response Data

None

4.4 Server Upload Commands

4.4.1 SendHTTPMessageByRegister(hostName, message)

A Host requests to send an HTTP request message to the specified server. It is assumed that a Host prepares the HTTP request message containing the HTTP Request Line, the HTTP message header and the HTTP message body (optional in [HTTP]).
Then, a Card receives the HTTP response message containing the HTTP Response Line, the HTTP message header and the HTTP message body (optional in [HTTP]) from the server.
See also **2.3 Server Upload**.

4.4.1.1 Argument

4.4.1.1.1 Argument 1

Name: hostName
Type: ASCII character
Size: Variable
Description: The 'hostName' is the host name which the HTTP request message is sent to and the HTTP response message is received from. The port number "80" is used unless a proxy server is used. This value is not terminated by a null character.

4.4.1.1.2 Argument 2

Name: message
Type: Binary data
Size: Variable
Description: The 'message' is the HTTP request message containing the HTTP Request Line, the HTTP message header and the HTTP message body.

4.4.1.2 Response Data

The received HTTP response message is available from the Response Data of the Command

Response Data (See **2.2.2.3 iSDIO Command Response Data** in [iSDIO]).

4.4.2 SendHTTPFileByRegister(hostName, appendFileName, message)

A Host requests to send an HTTP request message to the specified server, attaching the file (append file). It is assumed that a Host prepares the HTTP request message containing the HTTP Request Line, the HTTP message header and the HTTP message body (optional in [HTTP]), and in addition a Card replaces the pre-defined characters ("<!--WLANSDFILE-->" in [ISO 8859-1]) in the message with a specified file when the message is sent to the server.
Then, a Card receives the HTTP response message containing the HTTP Response Line, the HTTP message header and the HTTP message body (optional in [HTTP]) from the server.
See also **2.3 Server Upload**.

4.4.2.1 Argument

4.4.2.1.1 Argument 1

Name: hostName
Type: ASCII character
Size: Variable
Description: The 'hostName' is the host name which the HTTP request message is sent to and the HTTP response message is received from. The port number "80" is used unless the proxy server is used. This value is not terminated by a null character.

4.4.2.1.2 Argument 2

Name: appendFileName
Type: ASCII character
Size: Variable
Description: The 'appendFileName' is the filename and the file path from the root directory of the append file in the NAND Memory module to be sent to a server. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.4.2.1.3 Argument 3

Name: message
Type: Binary data
Size: Variable
Description: The 'message' is the HTTP request message containing the HTTP Request Line, the HTTP message header and the HTTP message body.

4.4.2.2 Response Data

The received HTTP response message is available from the Response Data of the Command Response Data (See **2.2.2.3 iSDIO Command Response Data** in [iSDIO]).

4.4.3 SendHTTPSSLMessageByRegister(hostName, message)

The definition of this command is same as that of the "SendHTTPMessageByRegister(hostName, message)" except that this command requests a Card to send an HTTP request message via SSL. The port number "443" is used unless the proxy server is used.
It is recommended the root certificate should be set by the "SetCertificate" or "SetCertificateByFile" command before issuing this command. If this command is issued without setting the root certificate, a Card processes this command without the verification of the server certificate by the root certificate.
If the server certificate is invalid in the verification by the root certificate, the command processing will fail (i.e. Response Status becomes "85h: SSL certification error").

4.4.4 SendHTTPSSLFileByRegister(hostName, appendFileName, message)

The definition of this command is same as that of the "SendHTTPFileByRegister(hostName, appendFileName, message)" except that this command requests a Card to send an HTTP request message via SSL.
The port number "443" is used unless a proxy server is used.
It is recommended the root certificate should be set by the "SetCertificate" or "SetCertificateByFile" command before issuing this command. If this command is issued without setting the root certificate, a Card processes this command without the verification of the server certificate by the root certificate.
If the server certificate is invalid in the verification by the root certificate, the command processing will fail (i.e. Response Status becomes "85h: SSL certification error").

4.4.5 SendHTTPMessageByFile(hostName, messageFileName, headerRemoval)

A Host requests to send an HTTP request message to the specified server. It is assumed that a Host stores the HTTP request message as a file (message file) in the NAND Memory module, containing the HTTP Request Line, the HTTP message header and the HTTP message body (optional in [HTTP]) before issuing the command.
Then, a Card receives the HTTP response message containing the HTTP Response Line, the HTTP message header and the HTTP message body (optional in [HTTP]) from the server, and stores it as a file (response file) whose file name is "iSDIO command sequence id" under the "RESPONSE" directory (This operation overwrites the existing file if it exists). If a Host requests to remove the HTTP Request Line and the HTTP message header, only the HTTP message body is stored as a response file.
See also **2.3 Server Upload**.
Note that the restriction specified by "Max Size of Command Response Data" in the Capability Register is not applied to the size of the response file created by this command.

4.4.5.1 Argument

4.4.5.1.1 Argument 1

Name: hostName
Type: ASCII character
Size: Variable
Description: The ‘hostName’ is the host name which the HTTP request message is sent to and the HTTP response message is received from. The port number "80" is used unless the proxy server is used. This value is not terminated by a null character.

4.4.5.1.2 Argument 2

Name: messageFileName
Type: ASCII character
Size: Variable
Description: The ‘messageFileName’ is the filename and the file path from the root directory of the message file in the NAND Memory module, containing the HTTP Request Line, the HTTP message header and the HTTP message body. e.g. "/REQUEST/MESSAGE1.TXT". This value is not terminated by a null character.

4.4.5.1.3 Argument 3

Name: headerRemoval
Type: integer
Size: 1-byte
Description: The ‘headerRemoval’ denotes whether the HTTP header is removed.
00h: The HTTP response message is stored as is.
01h: Only HTTP message body is stored by removing HTTP header.

4.4.5.2 Response Data

None

4.4.6 SendHTTPFileByFile(hostName, messageFileName, appendFileName, headerRemoval)

The Host requests to send an HTTP request message to the specified server, attaching the file (append file). It is assumed that a Host stores the HTTP request message as a file (message file) in the NAND Memory module, containing the HTTP Request Line, the HTTP message header and the HTTP message body (optional in [HTTP]) before issuing the command. And in addition the Card replaces the pre-defined characters (i.e. "<!--WLANSDFILE-->" in [ISO 8859-1]) in the message with a specified file when the message is sent to the server.
Then, the Card receives the HTTP response message containing the HTTP Response Line, the HTTP message header and the HTTP message body (optional in [HTTP]) from the server, and stores it as a file (response file) whose file name is "iSDIO command sequence id" under the "RESPONSE" directory (This operation overwrites the existing file if it exists). If the Host requests to remove the HTTP Request Line and the HTTP message header, only the HTTP message body is stored as a response file.
See also **2.3 Server Upload**.
Note that the restriction specified by "Max Size of Command Response Data" in the Capability Register is not applied to the size of the response file created by this command.

4.4.6.1 Argument

4.4.6.1.1 Argument 1

Name: hostName
Type: ASCII character
Size: Variable
Description: The 'hostName' is the host name that the HTTP request message is sent to and the HTTP response message is received from. The port number "80" is used unless the proxy server is used. This value is not terminated by a null character.

4.4.6.1.2 Argument 2

Name: messageFileName
Type: ASCII character
Size: Variable
Description: The 'messageFileName' is the filename and the file path from the root directory of the message file in the NAND Memory module, containing the HTTP Request Line, the HTTP message header and the HTTP message body. e.g. "/REQUEST/MESSAGE1.TXT". This value is not terminated by a null character.

4.4.6.1.3 Argument 3

Name: appendFileName
Type: ASCII character
Size: Variable
Description: The 'appendFileName' is the filename and the file path from the root directory of the append file in the NAND Memory module to be sent to a server. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.4.6.1.4 Argument 4

Name: headerRemoval
Type: integer
Size: 1-byte
Description: The 'headerRemoval' denotes whether the HTTP header is removed.

00h: The HTTP response message is stored as is.
01h: Only HTTP message body is stored by removing HTTP header.

4.4.6.2 Response Data
None

4.4.7 SendHTTPSSLMessageByFile(hostName, messageFileName, headerRemoval)
The definition of this command is same as that of the "SendHTTPMessageByFile (hostName, messageFileName, headerRemoval)" except that this command requests the Card to send an HTTP request message via SSL.
The port number "443" is used unless the proxy server is used.
It is recommended the root certificate should be set by the "SetCertificate" or "SetCertificateByFile" command before issuing this command. If this command is issued without setting the root certificate, the Card processes this command without the verification of the server certificate by the root certificate.
If the server certificate is invalid from the verification of the root certificate, the command processing shall fail (i.e. Response Status becomes "85h: SSL certification error").

4.4.8 SendHTTPSSLFileByFile(hostName, messageFileName, appendFileName, headerRemoval)
The definition of this command is same as that of the "SendHTTPFileByFile (hostName, messageFileName, appendFileName, headerRemoval)" except that this command requests a Card to send an HTTP request message via SSL.
The port number "443" is used unless the proxy server is used.
It is recommended the root certificate should be set by the "SetCertificate" or "SetCertificateByFile" command before issuing this command. If this command is issued without setting the root certificate, the Card processes this command without the verification of the server certificate by the root certificate.
If the server certificate is invalid from the verification of the root certificate, the command processing shall fail (i.e. Response Status becomes "85h: SSL certification error").

4.4.9 SetCertificate(certificate)
Set a root certificate for HTTPS connections. After processing of this command, the Card shall use the given certificate for any HTTPS connection. If a root certificate has already been set, the Card shall invalidate the current one and set the given one.

4.4.9.1 Argument

4.4.9.1.1 Argument 1
Name: certificate
Type: Binary data
Size: Variable
Description: The ‘certificate’ is X.509 binary encoded with DER (Distinguished Encoding Rules). ([X.509])
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h"). If omitted, the Card shall invalidate the current certificate and return to the default behavior.

4.4.9.2 Response Data
None

Note that the detailed behaviors of HTTPS depend on the implementation of a Card (e.g. supported encryption method, confirmation level of the certificate, etc.). Also it depends on the specification of the destination host.

4.4.10 SetCertificateByFile(certificateFileName)

Set a root certificate file stored in the NAND Memory module for HTTPS connections. After processing of this command, the Card shall use the given certificate for any HTTPS connection. If a root certificate has already been set, the Card shall invalidate the current one and set the given one.

4.4.10.1 Argument

4.4.10.1.1 Argument 1

Name: certificateFileName
Type: ASCII character
Size: Variable
Description: The 'certificateFileName' is the filename and the file path from the root directory of the certificate file in the NAND Memory module. e.g. "/ABCDEF.CRT". This value is not terminated by a null character. The certificate file is X.509 binary encoded with DER (Distinguished Encoding Rules). ([X.509])
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h"). If omitted, the Card shall invalidate the current certificate and return to the default behavior.

4.4.10.2 Response Data

None

Note that the detailed behaviors of HTTPS depend on the implementation of the Card (e.g. supported encryption method, confirmation level of the certificate, etc.). Also it depends on the specification of the destination host.

4.5 DLNA Commands

4.5.1 DLNA_SetDeviceInformation(parameterID, parameterValue)

Set some contents of the Device Description for M-DMS can be changed by a Host before the start of DLNA M-DMS mode. If these contents aren't set before issuing the other DLNA commands, the default values defined by each Card shall be used. This command is used with the application status "No Application" only. Otherwise, a Card shall set "02h: Command Rejected" to Response Status.

4.5.1.1 Argument

4.5.1.1.1 Argument 1

Name: parameterID
Type: integer
Size: 1-byte
Description: The 'parameterID' denotes the Parameter ID.
00h: friendlyName

4.5.1.1.2 Argument 2

Name: parameterValue
Type: ASCII character

Size: Variable
Description: The 'parameterValue' denotes the Parameter Value. The requirement (e.g. length, format, etc.) of each value for each parameterID conforms to the DLNA guideline. This value is not terminated by a null character.

4.5.1.2 Response Data
None

4.5.2 DLNA_SwitchMode(modeID)
Switch the application status between "No Application" and each DLNA mode (use as start/stop). If a Host issues this command when a Card is in one of the DLNA mode (M-DMU, +PU+ and M-DMS), modeID shall be set as "No Application" only. That is, direct mode switching between a DLNA mode and the other DLNA mode is prohibited.

4.5.2.1 Argument

4.5.2.1.1 Argument 1
Name: modeID
Type: integer
Size: 1-byte
Description: The 'modeID' denotes DLNA mode defined below.
00h: No Application (exit DLNA mode)
01h: DLNA M-DMU mode
02h: DLNA +PU+ mode
03h: DLNA M-DMS mode

4.5.2.2 Response Data
None

When a Card switches mode, a Card shall set '0b' to ULR/DLU/CBR/CDR bit of DLNA Status and HPC bit of HTTP Status.

4.5.3 DLNA_GetDeviceList()
Get the device list of detected DMS/M-DMS/DMR devices with the UPnP discovery process. In M-DMU mode, DMS/M-DMS device list is available. In +PU+ mode, DMR device list is available. In other modes, the Card shall set "02h: Command Rejected" to Response Status.
If a Host issues this command in the state when the DLU bit of DLNA Status is '0b', a Card shall return the list same as the previous one. Especially in the initial state, a Card shall return an empty list.
After processing of this command, a Card shall clear the DLU bit of DLNA Status.

4.5.3.1 Argument
None

4.5.3.2 Response Data
The device list is available from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

4.5.4 DLNA_StartUpload(targetUDN, targetFileName, notifyFileName, MIMEType,

profileID)

Start uploading a target file to a target DMS/M-DMS device. A Card invokes the CreateObject action of the UPnP. If the CreateObject action is succeeded, the Card starts a file transfer with HTTP POST. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of the file transfer. This command is used only in M-DMU mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.5.4.1 Argument

4.5.4.1.1 Argument 1

Name: targetUDN
Type: ASCII character
Size: Variable
Description: The 'targetUDN' is UDN of the target device from the Response Data of the "DLNA_GetDeviceList" command. e.g. "uuid:00000000-0000-0000-0000-000000000000". This value is not terminated by a null character.

4.5.4.1.2 Argument 2

Name: targetFileName
Type: ASCII character
Size: Variable
Description: The 'targetFileName' is the filename and the file path from the root directory to upload to a target device. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.5.4.1.3 Argument 3

Name: notifyFileName
Type: UTF-16LE character
Size: Variable
Description: The 'notifyFileName' shall be used to notify the target file name to the target DMS/M-DMS device via the DLNA protocol. This argument shall be used only for the notification purpose by the Card. It is sent as UTF-16LE character. And it is strongly recommended for the Host to set the original file name without the file name extension in this argument. This value is not terminated by a null character. Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h"). If omitted, the Card shall use the file name without the file name extension of 'targetFileName' for the notification.
By using this argument, the Host can specify any name other than the 'targetFileName'. For example, if the Host wants to upload a file other than 8.3 format file name, the Host can upload it after conversion to the temporary 8.3 format file name and specify the original file name by using 'notifyFileName'.

4.5.4.1.4 Argument 4

Name: MIMEType
Type: ASCII character
Size: Variable
Description: The 'MIMEType' is MIME-Type of the corresponding profile ID. e.g. "image/jpeg". This value is not terminated by a null character.

4.5.4.1.5 Argument 5

Name: profileID
Type: ASCII character
Size: Variable
Description: The 'profileID' is profile ID of the target content item. The detail of the profile ID conforms to the DLNA Guideline. e.g. "JPEG_SM". This value is not terminated by a null

character.

4.5.4.2 Response Data

None

In the upload process of the target file, a Card shall convert the ‘notifyFileName’ from UTF-16LE to UTF-8 and set it as dc:title metadata of the CreateObject request.
To inform the target DMS/M-DMS device of the last modified date and time of the target file, a Card shall invoke CreateObject action with dc:date that is described as optional metadata in [DLNA]. Here, a Card shall set the last modified date and time timestamp of the target file to dc:date.
To ensure Card to Card transfer with DLNA functions, a Card shall invoke CreateObject action with res@size that is described as optional metadata in [DLNA].
To indicate HTTP POST status and progress, a Card shall set the HPC bit and HTTP Progress of HTTP Status.
Note that any type of content may be transferred to a -UP- device by this command, whether or not the content is playable on the -UP- device.

4.5.5 DLNA_PushPlayback(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)

Notify a target DMR device to start playback of the target content. A Card invokes the SetAVTransportURI action and the Play action of UPnP. If the actions are succeeded, the Card waits for an HTTP GET request to start a file transfer. In normal processing of this command, a Card shall keep Response Status as "01h: Command Processing" until completion of the file transfer. This command is used in only +PU+ mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.5.5.1 Argument

4.5.5.1.1 Argument 1

Name: targetUDN
Type: ASCII character
Size: Variable
Description: The ‘targetUDN’ is UDN of the target device from the Response Data of the "DLNA_GetDeviceList" command. e.g. "uuid:00000000-0000-0000-0000-000000000000". This value is not terminated by a null character.

4.5.5.1.2 Argument 2

Name: targetFileName
Type: ASCII character
Size: Variable
Description: The ‘targetFileName’ is the filename and the file path from the root directory to play back on a target device. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.5.5.1.3 Argument 3

Name: notifyFileName
Type: UTF-16LE character
Size: Variable
Description: The ‘notifyFileName’ shall be used to notify the target file name to the target DMR device via the DLNA protocol. This argument shall be used only for the notification purpose by the Card. It is sent as UTF-16LE character. And it is strongly recommended for the Host to set

the original file name without the file name extension in this argument. This value is not terminated by a null character. Note that this value is able to be omitted ("Length of Argument" is set to '00 00 00 00h'). If omitted, the Card shall use the file name without the file name extension of 'targetFileName' for the notification.
By using this argument, the Host can specify any name other than the 'targetFileName'. Typically, it is assumed that the specified value is used as a title of the target content on the target DMR.

4.5.5.1.4 Argument 4

Name: MIMEType
Type: ASCII character
Size: Variable
Description: The 'MIMEType' is MIME-Type of the corresponding profile ID. e.g. "image/jpeg". This value is not terminated by a null character.

4.5.5.1.5 Argument 5

Name: profileID
Type: ASCII character
Size: Variable
Description: The 'profileID' is profile ID of the target content. Detail of the profile ID conforms to the DLNA Guideline. e.g. "JPEG_SM". This value is not terminated by a null character.

4.5.5.2 Response Data

None

In the push playback process of the target file, a Card shall convert 'notifyFileName' from UTF-16LE to UTF-8 and set it as dc:title metadata of SetAVTransportURI request.
To indicate HTTP GET status and progress, a Card shall set the HPC bit and HTTP Progress of HTTP Status.
Note that if the "Abort" command is issued during processing this command, a Card may need more than 3 seconds to finish post processing such as the Stop action of UPnP. Even so, a Card shall finish processing of the "Abort" command in 3 seconds and shall release the permission to read the file system. Such post processing should be implemented as internal processes of a Card. In addition, aborting the post processing should be considered too. For example, if the "DLNA_SwitchMode" command is issued, a Card shall finish all internal processing about the DLNA +PU+ mode appropriately.

4.5.6 DLNA_GetUploadInformation()

Get the information of the content upload requested. It is assumed that the values are received as metadata in CreateObject request. A Card shall keep this information during the ULR bit of DLNA Status is set. This command is used only in M-DMS mode with ULR bit of DLNA Status is set. In other modes, a Card shall set "02h: Command Rejected" to Response Status.

4.5.6.1 Argument

None

4.5.6.2 Response Data

The information of upload content is available from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

4.5.7 DLNA_AcceptUpload(accept, targetFileName)

Start receiving upload content. A Card creates new file using given filename and writes the received content data (This operation overwrites the existing file if exists). In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of the file transfer. This command is used only in M-DMS mode with the ULR bit of DLNA Status set. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.
A Host shall issue this command after completion of the "DLNA_GetUploadInformation" command. Otherwise, a Card shall set "02h: Command Rejected" to the Response Status (A Host shall not omit issuing of "DLNA_GetUploadInformation" command).
After processing of this command, a Card shall clear the ULR bit of the DLNA Status.

4.5.7.1 Argument

4.5.7.1.1 Argument 1

Name: accept
Type: integer
Size: 1-byte
Description: The 'accept' is defined below.
00h: A Host does not accept the request (e.g. Writable area is not big enough)
01h: A Host accepts the request.

4.5.7.1.2 Argument 2

Name: targetFileName
Type: ASCII character
Size: Variable
Description: The 'targetFileName' is the filename and the file path from the root directory to store received content data by Card. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.
Note that if 'accept' is set to "00h" (not accept), this argument has not effect and this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h").

4.5.7.2 Response Data

None

If a Card detects dc:date metadata in the CreateObject request and it includes both date and time information, a Card shall use the value as the last modified date and time timestamp of the created file (internal date and time information which is set by the "SetCurrentTime" command should be used to the other timestamps). If no dc:date is detected or dc:date doesn't have enough information, a Card may use internal date and time information for all timestamps.
To indicate HTTP POST status and progress, a Card shall set the HPC bit and HTTP Progress of HTTP Status.
Note that a Card shall wait for this command before issuance of CreateObject response. In addition, the time-out processing about issuance of CreateObject response shall be implemented on a Card. In time-out processing, a Card shall clear the ULR bit of the DLNA Status.

4.5.8 DLNA_SetUpdateID(valid, UpdateID)

Set UpdateID for the content distribution database of M-DMS. In addition, notify whether the database files are valid or not. A Card shall keep these settings until this command is reissued. At the processing of exiting the DLNA M-DMS mode, a Card shall not revoke them. This command is used with the application status "No Application" only. Otherwise, a Card shall set "02h: Command Rejected" to Response Status.

4.5.8.1 Argument

4.5.8.1.1 Argument 1

Name: valid
Type: integer
Size: 1-byte
Description: The 'valid' is defined below.
00h: The database files are invalid
01h: The database files are valid

4.5.8.1.2 Argument 2

Name: UpdateID
Type: unsigned integer
Size: 4-byte
Description: The 'UpdateID' is the UpdateID for the content distribution database. The SystemUpdateID state variable shall be set to this value. The detail of the SystemUpdateID state variable is described in [DLNA]. Note that even if 'valid' is set to "00h", this argument shall not be omitted. (The UpdateID for the empty database shall be set.)

4.5.8.2 Response Data

None

Note that if a Host switches application to the DLNA M-DMS mode without issuing this command beforehand, the database files shall be invalidated and the SystemUpdateID state variable shall be set to zero.

4.5.9 DLNA_AcceptBrowse(accept)

Notify whether a Host accepts the Browse request indicated at the CBR bit of the DLNA Status or not. If accepted, the content of response shall depend on the corresponding database file. This command is used only in M-DMS mode with the CBR bit of the DLNA Status set. In other modes, a Card shall set "02h: Command Rejected" to Response Status.
After processing of this command, a Card shall clear the CBR bit of the DLNA Status.

4.5.9.1 Argument

4.5.9.1.1 Argument 1

Name: accept
Type: integer
Size: 1-byte
Description: The 'accept' is defined below.
00h: A Host does not accept the request
01h: A Host accepts the request

4.5.9.2 Response Data

None

Note that a Card may read not only the database file but also other related files. For example, a Card may analyze each content data to generate optional metadata not written in database file. Note that a Card shall wait this command before issuance of Browse response. In addition, the time-out processing about issuance of Browse response shall be implemented on a Card. In the time-out processing, a Card shall clear the CBR bit of the DLNA Status.

4.5.10 DLNA_AcceptDistribution(accept)

Notify whether a Host accepts the HTTP GET request indicated at the CDR bit of the DLNA Status or not. If accepted, the Card starts a file transfer after the HTTP response. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of the file transfer. This command is used only in M-DMS mode with the CDR bit of the DLNA Status set. In other modes, a Card shall set "02h: Command Rejected" to Response Status. After processing of this command, a Card shall clear the CDR bit of the DLNA Status.

4.5.10.1 Argument

4.5.10.1.1 Argument 1

Name: accept
Type: integer
Size: 1-byte
Description: The 'accept' is defined below.
00h: A Host does not accept the request
01h: A Host accepts the request

4.5.10.2 Response Data

None

To indicate HTTP POST status and progress, a Card shall set the HPC bit and HTTP Progress of HTTP Status.
Note that a Card shall wait for this command before issuance of the HTTP response. In addition, the time-out processing about issuance of the HTTP response shall be implemented on a Card. In the time-out processing, a Card shall clear the CDR bit of the DLNA Status.

4.6 P2P File Transfer Commands

4.6.1 StartP2PSender(ssid, networkKey, encMode)

In the Peer-to-Peer File Transfer application, for a Sender Card, a Host requests to start the Sender Application (**2.5 Peer-to-Peer File Transfer**). The Sender Application shall establish wireless LAN as the AP in the Infrastructure, be the DHCP server, and handle the HTTP request/response messages defined in **Appendix C** in accordance with the sequence defined in **D.4 Peer-to-Peer File Transfer Sequence**. The command process is finished when the Card has initiated the AP successfully.

4.6.1.1 Argument

4.6.1.1.1 Argument 1

Name: ssid
Type: ASCII character
Size: Variable
Description: The 'ssid' is the name of SSID to be established. This value is not terminated by a null character.

4.6.1.1.2 Argument 2

Name: networkKey
Type: ASCII character
Size: Variable

Description: The 'networkKey' is the network key of the corresponding SSID. This value is not terminated by a null character.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") if the encryption is not used for the wireless LAN connection.

4.6.1.1.3 Argument 3

Name: encMode
Type: integer
Size: 1-byte
Description: The 'encMode' denotes the network authentication and the data encryption scheme for the AP.

- 00h: Open System and no encryption
- 01h: Open System and WEP (Optional for a Card)
- 02h: Shared Key and WEP (Optional for a Card)
- 03h: WPA-PSK and TKIP (Optional for a Card)
- 04h: WPA-PSK and AES (Optional for a Card)
- 05h: WPA2-PSK and TKIP (Optional for a Card)
- 06h: WPA2-PSK and AES (Mandatory for a Card)

Note that supported combinations of network authentication and data encryption depend on a Card implementation, and a Host needs to check the "Encryption Mode Support" in the Capability Register. See **3.5 iSDIO Capability Register for Wireless LAN**.

4.6.1.2 Response Data

None

4.6.2 StartP2PReceiver(ssid, networkKey)

In the Peer-to-Peer File Transfer application, for a Receiver Card, a Host requests to start the Receiver Application (**2.5 Peer-to-Peer File Transfer**). The Receiver Application shall connect the specified AP (the Sender Card) in the wireless LAN as the STA in Infrastructure, and handle the HTTP request/response messages defined in **Appendix C** in accordance with the sequence defined in **D.4 Peer-to-Peer File Transfer Sequence**. The command process is finished when the Card has received the HTTP response message for the HTTP(ID) defined in **C.2.1 HTTP(ID)** successfully.

4.6.2.1 Argument

4.6.2.1.1 Argument 1

Name: ssid
Type: ASCII character
Size: Variable
Description: The 'ssid' is the SSID of the AP to be connected. This value is not terminated by a null character.

4.6.2.1.2 Argument 2

Name: networkKey
Type: ASCII character
Size: Variable
Description: The 'networkKey' is the network key of the corresponding SSID. This value is not terminated by a null character.
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") if the encryption is not used for the WLAN connection.

4.6.2.2 Response Data

None

4.6.3 GetFile(requestFileName, saveFileName)

In the Peer-to-Peer File Transfer application, for a Receiver Card, a Host requests to download a file from the Sender Card and store it as a file with the specified location and filename in the Receiver Card. The filename and the file path of the download file are identified by the downloaded File List file from the Sender Card. The Receiver Card shall send the HTTP request message to download the specified file (HTTP(FILE)) defined in **C.2.5 HTTP(FILE)**. Then the downloaded file shall be stored as a file with the specified filename and at the specified location in the Receiver Card (This operation overwrites the existing file if it exists).

4.6.3.1 Argument

4.6.3.1.1 Argument 1

Name: requestFileName
Type: ASCII character
Size: Variable
Description: The 'requestFileName' is the filename and the file path from the root directory of the download file in the Sender Card. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.6.3.1.2 Argument 2

Name: saveFileName
Type: ASCII character
Size: Variable
Description: The 'saveFileName' is the filename and the file path from the root directory of the file to be stored in the Receiver Card. e.g. "/RECEIVE/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.6.3.2 Response Data

None

4.6.4 ReadIDList()

In the Peer-to-Peer File Transfer application, a Host requests the Sender Card to read the ID List containing the IDs of the Receiver Cards. The ID List is available from the Response Data Register Port as a result of the command process.

4.6.4.1 Argument

None

4.6.4.2 Response Data

The ID List is available from the Response Data of the Command Response Data (See **2.2.2.3 iSDIO Command Response Data** in [iSDIO]). The format of the ID List is defined in **3.6.6 Wireless LAN ID List**.

4.6.5 SelectMAC(mac)

In the Peer-to-Peer File Transfer application, for a Sender Card, a Host requests to select a MAC Address of a Receiver Card and allow the Receiver Card to access the Sender Card. A Host is able to select one or more MAC Addresses of Receiver Cards by issuing this command in multiple times. The Sender Card shall send the HTTP request message (HTTP(ACCEPT)) defined in **C.2.2 HTTP(ACCEPT)**.

4.6.5.1 Argument

4.6.5.1.1 Argument 1

Name: mac
Type: integer (big endian)
Size: 6-byte
Description: The 'mac' is the MAC Address of a Receiver Card. e.g. "00h", "11h", "22h", "AAh", "BBh" and "CCh" are stored in this order for "00-11-22-AA-BB-CC".

4.6.5.2 Response Data

None

4.6.6 DeselectMAC(mac)

In the Peer-to-Peer File Transfer application, for a Sender Card, a Host requests to deselect a MAC Address of a Receiver Card and remove the MAC Address of the Receiver Card from the ID List. A Host is able to deselect one or more MAC Addresses of Receiver Cards by issuing this command in multiple times. The Sender Card shall send the HTTP request message (HTTP(REJECT)) defined in **C.2.3 HTTP(REJECT)**.

4.6.6.1 Argument

4.6.6.1.1 Argument 1

Name: mac
Type: integer (big endian)
Size: 6-byte
Description: The 'mac' is the MAC Address of a Receiver Card. e.g. "00h", "11h", "22h", "AAh", "BBh" and "CCh" are stored in this order for "00-11-22-AA-BB-CC".

4.6.6.2 Response Data

None

4.6.7 SetID(id)

In the Peer-to-Peer File Transfer application, for a Receiver Card, a Host requests to set an ID of a Receiver Card.

4.6.7.1 Argument

4.6.7.1.1 Argument 1

Name: id
Type: ASCII character
Size: Variable
Description: The 'id' is the ID of a Receiver Card. This value is not terminated by a null character.

4.6.7.2 Response Data

None

4.7 PTP Commands

4.7.1 PTP_SetDeviceInformation(parameterID, parameterValue)

Set some information of the PTP responder. These values can be changed by a Host before the start of PTP mode. If these contents aren't set before issuing the other PTP commands, the default values defined by each Card shall be used. This command is used with the application status "No Application" only. Otherwise, a Card shall set "02h: Command Rejected" to Response Status.

4.7.1.1 Argument

4.7.1.1.1 Argument 1

Name: parameterID
Type: integer
Size: 1-byte
Description: The 'parameterID' denotes the Parameter ID.
- 00h: friendlyName
- 01h: manufacturer
- 02h: modelDescription
- 03h: modelName
- 04h: serialNumber
- 05h: UDN
- 06h: deviceVersion

4.7.1.1.2 Argument 2

Name: parameterValue
Type: ASCII character
Size: Variable
Description: The 'parameterValue' denotes the Parameter Value. The requirement (e.g. length, format, etc.) of each value for each parameterID conforms to corresponding specifications. This value is not terminated by a null character.

4.7.1.2 Response Data

None

Note that if the 'parameterID' is set to '00h'-'05h', a Card shall directly apply the 'parameterValue' to the corresponding field (of same name) in the device description of MTP/IP. A Card can assume the format of the value is UTF-8.
Note that if the 'parameterID' is set to '01h', '03h', '04h' or '06h', a Card shall apply the 'parameterValue' to the corresponding field ("Manufacturer", "Model", "SerialNumber" and "DeviceVersion") in the DeviceInfo dataset. A Card shall convert the format of the value to UTF-16LE.
Note that if the 'parameterID' is set to '00h', a Card shall apply the 'parameterValue' to "Responder Friendly Name" in the Init Command ACK packet. A Card shall convert the format of the values to UTF-16LE.
Note that if the 'parameterID' is set to '05h', a Card shall apply the 'parameterValue' to "Responder GUID" in the Init Command ACK packet. A Card shall get the 16-byte UUID value from the UDN and generate the array of 1-byte unsigned integer as GUID.

4.7.2 PTP_SetHiddenObjectType(objectTypeList)

Set the object types which a Host intends to hide. A Host can request to hide files corresponding to the specified object type before the start of PTP Pull mode. The object type is specified by a file

extension. In the processing of PTP Pull mode, a Card shall hide the files which have the file extension specified by a Host. At the processing of exiting the PTP Pull mode, a Card shall not revoke them. If a list has been already set, the Card shall invalidate the current one and set the given one. This command is used with the application status "No Application" only. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.2.1 Argument

4.7.2.1.1 Argument 1

Name: objectTypeList
Type: PTP Object Type List
Size: Variable
Description: The 'objectTypeList' is the list of extensions of the object type which the Host intends to hide. The format is described in Table 4-2.

Size (byte)	Name	Short Description	Type
1	Number of types	Number of object types	R/O
3	Reserved		R/O
4	Type #1	Extension of object type #1	R/O
...	...	...	R/O
4	Type #n	Extension of object type #n	R/O

Table 4-2 : PTP Object Type List

Number of types specifies the number of the object types which the host intends to hide.

Type specifies an extension of the object type which the host intends to hide. (e.g. "JPG", "TIF") The value is string of ASCII characters and it is not terminated with null character. If the string is shorter than 4 bytes, the rest shall be filled with "00h".

Reserved fields shall be filled with "00h".
Note that if the "Number of types" is "00h", this list consists of the 1-byte "Number of types" field and a 3-byte reserved field.

4.7.2.2 Response Data

None

Note that if this command is not issued before the start of PTP Pull mode, every types of files will be disclosed to the initiator.

4.7.3 PTP_SetHiddenDirectory(directoryList)

Set the directory paths which a Host intends to hide. A Host can request to hide specific directories before the start of PTP Pull mode. In the processing of PTP Pull mode, a Card shall hide the directories specified by a Host. At the processing of exiting the PTP Pull mode, a Card shall not revoke them. If a list has been already set, the Card shall invalidate the current one and set the given one. This command is used with the application status "No Application" only. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.3.1 Argument

4.7.3.1.1 Argument 1

Name: directoryList
Type: PTP Directory List
Size: Variable
Description: The 'directoryList' is the list of the directories which the Host intends to hide. The format is described in Table 4-3.

Size (byte)	Name	Short Description	Type
1	Number of directories	Number of directory paths	R/O
3	Reserved		R/O
1	Length of Directory #1	(L_{D1})	R/O
3	Reserved		R/O
L_{D1}	Directory #1	Path of directory #1	R/O
0, 1, 2 or 3	Padding #D1	Padding size is either 0 (for L_{D1} mod4=0), 1 (for L_{D1} mod4=3), 2 (for L_{D1} mod4=2), 3 (for L_{D1} mod4=1)	R/O
...	...	...	R/O
1	Length of Directory #n	(L_{Dn})	R/O
3	Reserved		R/O
L_{Dn}	Directory #n	Path of directory #n	R/O
0, 1, 2 or 3	Padding #Dn	Padding size is either 0 (for L_{Dn} mod4=0), 1 (for L_{Dn} mod4=3), 2 (for L_{Dn} mod4=2), 3 (for L_{Dn} mod4=1)	R/O

Table 4-3 : PTP Directory List

Number of directories specifies the number of the directories which the host intends to hide.

Length of Directory specifies the length of Directory in bytes. The value is in 1-byte unsigned integer.

Directory specifies a path of the directory which the host intends to hide. The path shall be an absolute path from the root directory. (e.g. "/DCIM/100XXXXX") The value is string of ASCII characters and it is not terminated with null character.

Reserved fields shall be filled with "00h".
Padding bytes shall be filled with "00h".
Note that if the "Number of directories" is "00h", this list consists of the 1-byte "Number of directories" field and a 3-byte reserved field.

4.7.3.2 Response Data

None

Note that if this command is not issued before the start of PTP Pull mode, all directories in the Card will be disclosed to the initiator.

4.7.4 PTP_SwitchMode(modeID)

Switch the application status between "No Application" and each PTP mode (use as start/stop). If a Host issues this command when a Card is in one of the PTP mode (PTP Pull, DPS and PTP Pass-Through), modeID shall be set as "No Application" only. That is, direct mode switching between a PTP mode and the other PTP mode is prohibited.

4.7.4.1 Argument

4.7.4.1.1 Argument 1

Name: modeID
Type: integer
Size: 1-byte
Description: The 'modeID' denotes PTP mode defined below.
00h: No Application (exit PTP mode)
01h: DPS mode
02h: PTP Pull mode
03h: PTP Pass-Through mode

4.7.4.2 Response Data

None

When a Card switches mode, a Card shall set "PTP Status", "PTP Transfer Status", "PTP Cancelled TransactionID" and "PTP Error Status" to default value.
Note that a Host shall not access the NAND memory module of the Card while the Card is operating in PTP Pull mode.
When a Card exits PTP Pull mode or PTP Pass-Through mode, the Card shall finish the advertisement processes. At that time, the Card should send ssdp:byebye (for [MTP/IP]) and goodbye packets (for [Multicast DNS]) if possible.

4.7.5 PTP_ReceiveOperation()

Get the payload of the Operation Request packet from the initiator. It is assumed that the response data for this command will be interpreted by the PTP responder implemented in the Host. This command is used in DPS mode or PTP Pass-Through mode with RPO bit of PTP Status is set. In other modes, a Card shall set "02h: Command Rejected" to Response Status.
After processing of this command, a Card shall clear the RPO bit of the PTP Status.

4.7.5.1 Argument

None

4.7.5.2 Response Data

The PTP Information Data is available from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

4.7.6 PTP_SendResponse(data)

Send an Operation Response packet to the initiator. The payload is given as the argument of this command and the Card shall add a PTP-IP header ("Length" and "Packet Type") and send it. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used in DPS mode or PTP Pass-Through mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.6.1 Argument

4.7.6.1.1 Argument 1

Name: data
Type: Binary data
Size: Variable
Description: The 'data' is the payload for the Operation Response packet.

4.7.6.2 Response Data

None

Note that after issuing of the "Abort" command to the processing of this command, the Host should assume that the Card may not be able to continue the current PTP-IP session. In a normal case, it is assumed that the "Abort" command is not applicable during the processing of this command, because the size of data to transfer in the processing of this command is small.

4.7.7 PTP_SetSendDataInformation(mode, TransactionID, TotalDataLength)

Set the method to specify the following data, and send a Start Data packet to the initiator. A Card shall generate the Start Data packet using specified parameters and send it. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used in DPS mode or PTP Pass-Through mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.7.1 Argument

4.7.7.1.1 Argument 1

Name: mode
Type: integer
Size: 1-byte
Description: The 'mode' is defined below.
00h: The data to transfer will be specified by the "PTP_SendData" command.
01h: The data to transfer will be specified by the "PTP_SendFile" command.

4.7.7.1.2 Argument 2

Name: TransactionID
Type: unsigned integer
Size: 4-byte
Description: The 'TransactionID' is the TransactionID to be used in the data-in phase started by this command. A Card shall apply this value to"TransactionID" of the Start Data packet, Data packets and End Data packet.

4.7.7.1.3 Argument 3

Name: TotalDataLength
Type: unsigned integer
Size: 8-byte
Description: The 'TotalDataLength' is the total size of the data to be sent in the data-in phase started by this command. A Card shall apply this value to "Total Data Length" of the Start Data packet.

4.7.7.2 Response Data

None

4.7.8 PTP_SendData(data)

Set the data for the data-in phase, and send a Data packet to the initiator. A Card shall generate the Data packet using specified data and send it. In order to avoid the limitation of "Max Size of Command Write Data", a Host is able to divide the data into multiple partial data and deliver by multiple issuing of this command. Therefore, a Card shall reconstruct those partial data and send it as single Data packet. Especially if "Total Data Length" of the Start Data packet is more than the maximum size that can be applied to single Data packet, a Card is able to divide into multiple Data packets. (The maximum size is decided by the Card within the limits described in [PTP-IP].) After sending data of size specified by "Total Data Length" of the Start Data packet, a Card shall send an End Data packet (or treat the last Data packet as the End Data packet). In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used after issuing of the "PTP_SetSendDataInformation" command with setting 'mode' to "00h" in DPS mode or PTP Pass-Through mode. In other modes or situations, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.8.1 Argument

4.7.8.1.1 Argument 1

Name: data
Type: Binary data
Size: Variable
Description: The 'data' is the data or partial data to be sent in current data-in phase.

4.7.8.2 Response Data

None

Note that a Card shall complete transfer of specified data in the processing of current issuing of this command. Although the construction of each PTP-IP packets depends on the implementation of each Card, a Card shall not buffer all or part of the specified data and wait the following data to transfer together.
Note that if the total size of sent data and specified data exceeds "Total Data Length" of the Start Data packet, a Card shall set "81h: argument error" to the Response Status and shall not transfer specified data.
Note that if a Card detects an initiator generated cancel for current data phase, the Card shall finish the data phase. The end processing of the data phase is carried out in the processing of the "PTP_CancelData" command whether it was in the act of the processing of this command or not. Especially in the processing of this command, the Card shall suspend data transfer immediately and finish with the Response Status "C2h: PTP-IP cancelled". And then, the Card waits for the "PTP_CancelData" command.

4.7.9 PTP_SendFile(targetFileName)

Set the data for the data-in phase, and send a Data packet to the initiator. A Card shall generate the Data packet using the content of specified file and send it. If "Total Data Length" of the Start Data packet is more than the maximum size that can be applied to single Data packet, a Card is able to divide into multiple Data packets. (The maximum size is decided by the Card within the limits described in [PTP-IP].) After sending data of size specified by "Total Data Length" of the Start Data packet, a Card shall send an End Data packet (or treat the last Data packet as the End Data packet). In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used after issuing of the "PTP_SetSendDataInformation" command with setting 'mode' to "01h" in DPS mode or PTP Pass-Through mode. In other modes or situations, a Card shall set "02h: Command Rejected" to the

Response Status.

4.7.9.1 Argument

4.7.9.1.1 Argument 1

Name: targetFileName
Type: ASCII character
Size: Variable
Description: The 'targetFileName' is the filename and the file path from the root directory of the file to be sent as the object data. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.7.9.2 Response Data

None

To indicate transfer status and progress, a Card shall set PTP Progress of PTP Transfer Status. Note that if the size of specified file is not equal to "Total Data Length" of the Start Data packet, a Card shall set "81h: argument error" to the Response Status and shall not transfer any data. Note that if a Card detects an initiator generated cancel for current data phase, the Card shall finish the data phase. The end processing of the data phase is carried out in the processing of the "PTP_CancelData" command whether it was in the act of the processing of this command or not. Especially in the processing of this command, the Card shall suspend data transfer immediately and finish with the Response Status "C2h: PTP-IP cancelled". And then, the Card waits for the "PTP_CancelData" command.

4.7.10 PTP_SetReceiveDataInformation(mode)

Set the method to process the following data, and get the payload of the Start Data packet from the initiator. It is assumed that the response data for this command will be interpreted by the PTP responder implemented in the Host. If the Start Data packet is not received before issuing of this command, a Card shall wait the packet in processing of this command. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used in DPS mode or PTP Pass-Through mode. In other modes, a Card shall set "02h: Command Rejected" to Response Status.

4.7.10.1 Argument

4.7.10.1.1 Argument 1

Name: mode
Type: integer
Size: 1-byte
Description: The 'mode' is defined below.
00h: The data received will be processed by the "PTP_ReceiveData" command.
01h: The data received will be processed by the "PTP_ReceiveFile" command.

4.7.10.2 Response Data

The PTP Information Data is available from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

Note that a Card shall receive PTP-IP packets without cooperating with each command. For example, it shall be considered in the implementation of a Card that some Data packets following the Start Data packet may arrive before the Host issues this command.

4.7.11 PTP_ReceiveData()

Get the data received in the data-out phase. This command is used after issuing of the "PTP_SetReceiveDataInformation" command with setting 'mode' to "00h" in DPS mode or PTP Pass-Through mode with RPD bit of PTP Status is set. In other modes or situations, a Card shall set "02h: Command Rejected" to Response Status.
After processing of this command, a Card shall clear the RPD bit of the PTP Status.

4.7.11.1 Argument

None

4.7.11.2 Response Data

The PTP Received Data is available from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

Note that a Card can divide an PTP-IP data into multiple partial data. Therefore, a Host may be required to issue this command multiple times to deliver a PTP-IP data.
Note that if a Card detects an initiator generated cancel for current data phase, the Card shall finish the data phase. The end processing of the data phase is carried out in the processing of the "PTP_CancelData" command whether it was in the act of the processing of this command or not. Especially in the processing of this command, the Card shall suspend data transfer immediately and finish with the Response Status "C2h: PTP-IP cancelled". And then, the Card waits for the "PTP_CancelData" command.

4.7.12 PTP_ReceiveFile(targetFileName)

Store the data received in the data-out phase to the NAND memory module. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used after issuing of the "PTP_SetReceiveDataInformation" command with setting 'mode' to "01h" in DPS mode or PTP Pass-Through mode. In other modes or situations, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.12.1 Argument

4.7.12.1.1 Argument 1

Name: targetFileName
Type: ASCII character
Size: Variable
Description: The 'targetFileName' is the filename and the file path from the root directory to use for the file to store the data. e.g. "/DCIM/100XXXXX/YYYY1234.JPG". This value is not terminated by a null character.

4.7.12.2 Response Data

None

To indicate transfer status and progress, a Card shall set PTP Progress of PTP Transfer Status.
Note that if a Card detects an initiator generated cancel for current data phase, the Card shall finish the data phase. The end processing of the data phase is carried out in the processing of the "PTP_CancelData" command whether it was in the act of the processing of this command or not. Especially in the processing of this command, the Card shall suspend data transfer immediately

and finish with the Response Status "C2h: PTP-IP cancelled". And then, the Card waits for the "PTP_CancelData" command.

4.7.13 PTP_CancelData()

Request the post processing of uncompleted PTP-IP packet transfer. If a Card is sending a packet, a Card shall send dummy data till the end of the packet. If a Card is receiving a packet, a Card shall receive remaining data of the packet and discard the received data. Especially in a data phase, a Card shall push forward the sequence until the end of the phase. (In a data-in phase, a Card shall send the End Data packet after the Data packet last sent. In a data-out phase, a Card shall receive the following Data packets and the End Data packet, and discard them.) In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used in DPS mode or PTP Pass-Through mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.
After processing of this command, a Card shall clear the RPD bit of PTP Status.

4.7.13.1 Argument

None

4.7.13.2 Response Data

None

Note that it is assumed that this command is used for both of responder generated cancel and initiator generated cancel. Specifically, the following situations are assumed.

1. Responder Generated Cancel in Data-In phase
   - 1.1 After the completion of the "PTP_SendData" command. But the total size of transferred data does not achieve "Total Data Length" specified in the Start Data packet.
   - 1.2 After the "Abort" command to the processing of the "PTP_SendData" command.
   - 1.3 After the "Abort" command to the processing of the "PTP_SendFile" command.
   - 1.4 After the "Abort" command to the processing of this command.
2. Responder Generated Cancel in Data-Out phase
   - 2.1 After the completion of the "PTP_ReceiveData" command. But the total size of transferred data does not achieve "Total Data Length" specified in the Start Data packet.
   - 2.2 After the "Abort" command to the processing of the "PTP_ReceiveData" command.
   - 2.3 After the "Abort" command to the processing of the "PTP_ReceiveFile" command.
   - 2.4 After the "Abort" command to the processing of this command.
3. Initiator Generated Cancel in Data-In phase
   - 3.1 After the "PTP_SendData" command finished with the Response Status "C2h: PTP-IP cancelled".
   - 3.2 After the "PTP_SendFile" command finished with the Response Status "C2h: PTP-IP cancelled".
   - 3.3 After this command finished with the Response Status "C2h: PTP-IP cancelled".
   - 3.4 After an initiator generated cancel which is detected in the state that the Card is not processing any of "PTP_SendData", "PTP_SendFile" or this command . (The "PTP Cancelled TransactionID" specifies the TransactionID of current transaction.)
4. Initiator Generated Cancel in Data-Out phase
   - 4.1 After the "PTP_ReceiveData" command finished with the Response Status "C2h: PTP-IP cancelled".
   - 4.2 After the "PTP_ReceiveFile" command finished with the Response Status "C2h: PTP-IP cancelled".
   - 4.3 After this command finished with the Response Status "C2h: PTP-IP cancelled".
   - 4.4 After an initiator generated cancel which is detected in the state that the Card is not

processing any of "PTP_ReceiveData", "PTP_ReceiveFile" or this command . (The "PTP Cancelled TransactionID" specifies the TransactionID of current transaction.)

Note that if a Card detects an initiator generated cancel for current data phase, the Card shall finish the data phase. The end processing of the data phase is carried out in the processing of this command.
Especially in the processing of this command, the Card shall suspend data transfer immediately and finish with the Response Status "C2h: PTP-IP cancelled". And then, the Card waits for the this command again.

In a data-out phase, a Host may issue this command after issuing of the "PTP_SendResponse" command (to send Transaction Cancelled). Additionally in both data-in phase and data-out phase, a Host may issue this command after issuing of the "PTP_SendEvent" command (to send CancelTransaction).

4.7.14 PTP_SendEvent(data)

Send an Event packet to the initiator. The payload is given as the argument of this command and a Card shall add a PTP-IP header ("Length" and "Packet Type") and send it. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until completion of data transfer. This command is used in DPS mode or PTP Pass-Through mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.14.1 Argument

4.7.14.1.1 Argument 1

Name: data
Type: Binary data
Size: Variable
Description: The ‘data’ is the payload for the Event packet.

4.7.14.2 Response Data

None

Note that after issuing of the "Abort" command to the processing of this command, the Host should assume that the Card may not be able to continue the current PTP-IP session. In a normal case, it is assumed that the "Abort" command is not applicable during the processing of this command, because the size of data to transfer in the processing of this command is small.

4.7.15 PTP_StartSearch(target)

Start UPnP device discovery process to search initiators. A Card shall start search using the search target specified by a Host, and generate the list of the devices internally. This process shall be finished by issuing of the "PTP_StartAdvertisement" command. This command is used in PTP Pull mode or PTP Pass-Through mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.15.1 Argument

4.7.15.1.1 Argument 1

Name: target
Type: ASCII character
Size: Variable

Description: The 'target' is the search target of the device discovery process. It is assumed that the Card applies the value to ST (Search Target) of M-SEARCH. (e.g. "upnp:rootdevice", "urn:schemas-upnp-org:device:MediaServer:1") This value is not terminated by a null character.

4.7.15.2 Response Data

None

Note that there is no specification that specifies the typical search target for the process started by this command. It depends on the implementation of the Host application. If the Host as a responder has no necessity to search specified initiators, there is no necessity to issue this command.
Note that in DPS mode, a Card shall search DPS printers and generate the list automatically.

4.7.16 PTP_GetInitiatorList()

Get the list of the initiators detected with the UPnP device discovery process. Generally, this command is used in DPS mode. In PTP Pull mode and PTP Pass-Through mode, this command is used while the device discovery process is carried out, (Between the issuing of the "PTP_StartSearch" command, and the issuing of the "PTP_StartAdvertisement" command.) In other modes or situations, a Card shall set "02h: Command Rejected" to the Response Status.
If a Host issues this command in the state when the CIL bit of PTP Status is '0b', a Card shall return the list same as the previous one. Especially in the initial state, a Card shall return an empty list.
After processing of this command, a Card shall clear the CIL bit of PTP Status.

4.7.16.1 Argument

None

4.7.16.2 Response Data

The PTP Initiator List Data is available from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

4.7.17 DPS_ConnectPrinter(targetUDN)

Request the initiator in the target printer to establish PTP connection. A Card shall invoke the ConnectionRequest action of the UPnP serviced on the specified printer and be ready to respond PTP-IP connection request from the initiator. In normal processing of this command, a Card shall keep the Response Status as "01h: Command Processing" until establishment of PTP-IP connection. This command is used only in DPS mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.17.1 Argument

4.7.17.1.1 Argument 1

Name: targetUDN
Type: ASCII character
Size: Variable
Description: The 'targetUDN' is UDN of the target printer from the Response Data of the "PTP_GetInitiatorList" command. e.g. "uuid:00000000-0000-0000-0000-000000000000". This value is not terminated by a null character.

4.7.17.2 Response Data

None

Note that the detail of the ConnectionRequest action is described in [DPS over IP].
Note that a Card shall set the CPI bit of PTP status to '1b' in the processing of this command.

4.7.18 PTP_StartAdvertisement(reject, GUID)

Start the device discovery processes to advertise to initiators. A Card shall start advertisement and be ready to respond PTP-IP connection request from the initiator. A Host can specify the target initiator with its GUID. If a GUID is specified, a Card shall reject all Init Command Request with other GUID. This command is used in PTP Pull mode or PTP Pass-Through mode. In other modes, a Card shall set "02h: Command Rejected" to the Response Status.

4.7.18.1 Argument

4.7.18.1.1 Argument 1

Name: reject
Type: integer
Size: 1-byte
Description: The 'reject' is defined below.
00h: The Card will accept an Init Command Request according to the 'GUID'.
01h: The Card shall reject all Init Command Request regardless of its GUID.
Note that "00h" is specified for this value in typical case. On the other hand, "01h" is specified to list initiators which attempts to access. If "01h" is specified for this value, the Card shall generate the PTP Rejected Initiator List. And then the Host can get the list by issuing the "PTP_GetRejectedInitiatorList" command.

4.7.18.1.2 Argument 2

Name: GUID
Type: array of 1-byte unsigned integer
Size: 16-byte
Description: The 'GUID' is the GUID of an initiator which the Host permits to connect. e.g. "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00h".
Note that this value is able to be omitted ("Length of Argument" is set to "00 00 00 00h") if a Host does not select an initiator. (Only the initiator which requests first is permitted to connect.)

4.7.18.2 Response Data

None

Note that if the 'GUID' is omitted, a Card shall accept all initiators except the "anonymous". And if "FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FFh" is specified to the 'GUID', a Card shall accept all initiators including the "anonymous".

Note that if a Host issues this command with setting 'reject' to "01h", it is assumed that the Host will issue this command again with setting 'reject' to "00h" after issuing of the "PTP_GetRejectedInitiatorList" command. At the processing of this command which is issued later, the Card shall finish previous advertisement processes (for rejection) once, and start the new processes (for acceptance). At that time, the Card should send ssdp:byebye (for [MTP/IP]) and goodbye packets (for [Multicast DNS]).

4.7.19 PTP_GetRejectedInitiatorList()

Get the list of the initiators which attempted to connect and was rejected. This command is used in PTP Pull mode or PTP Pass-Through mode, between two times of the issuing of the

"PTP_StartAdvertisement" command as described in the note of 4.7.18. In other modes or situations, a Card shall set "02h: Command Rejected" to the Response Status.
If a Host issues this command in the state when the CIL bit of PTP Status is '0b', a Card shall return the list same as the previous one. Especially in the initial state, a Card shall return an empty list.
After processing of this command, a Card shall clear the CIL bit of PTP Status.

4.7.19.1 Argument
None

4.7.19.2 Response Data
The PTP Rejected Initiator List Data is available from the Response Data of the Command Response Data (**3.6 iSDIO Response Data for Wireless LAN**).

4.8 iSDIO Wireless LAN Command Availability

Table 4-4 shows whether each command is allowed in each mode. "YES" means that the command is able to be issued in the mode, and "N/A" means the command is rejected in the mode.
Note that in case of an AP, "before connection" corresponds to "before AP is started" and "after connection" corresponds to "after AP is started".

Command ID	Command Name	No Application		DLNA M-DMU	DLNA +PU+	DLNA M-DMS	P2P Sender	P2P Receiver	DPS	PTP Pull	PTP Pass Through
		before connection	after connection								
0001h	Scan()	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0002h	Connect(ssid, networkKey)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0003h	Establish(ssid, networkKey, encMode)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0004h	WiFiDirect(wpsMode, pin)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0005h	StartWPS(ssid, wpsMode, pin)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0006h	StartWPSAP(wpsMode, pin)	YES	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0007h	Disconnect()	N/A	YES	N/A	N/A	N/A	YES	YES	N/A	N/A	N/A
0011h	SetCurrentTime(currentDate, currentTime)	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES
0012h	Abort(sequenceID)	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES
0013h	ReadResponse(sequenceID)	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES
0014h	SetPowerSaveMode(powerMode)	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES
0015h	SetChannel(channelNum)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0021h	SendHTTPMessageByRegister(hostName, message)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0022h	SendHTTPFileByRegister(hostName, appendFileName, message)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0023h	SendHTTPSSLMessageByRegister(hostName, message)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0024h	SendHTTPSSLFileByRegister(hostName, appendFileName, message)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0025h	SendHTTPMessageByFile(hostName, messageFileName, headerRemoval)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0026h	SendHTTPFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0027h	SendHTTPSSLMessageByFile (hostName, messageFileName, headerRemoval)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0028h	SendHTTPSSLFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

0029h	SetCertificate(certificate)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
002Ah	SetCertificateByFile(certificateFileName)	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0061h	DLNA_SetDeviceInformation(parameterID, parameterValue)	YES	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0062h	DLNA_SwitchMode(modeID)	N/A	YES	YES	YES	YES	N/A	N/A	N/A	N/A	N/A
0063h	DLNA_GetDeviceList()	N/A	N/A	YES	YES	N/A	N/A	N/A	N/A	N/A	N/A
0064h	DLNA_StartUpload(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	N/A	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0065h	DLNA_PushPlayback(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A	N/A	N/A
0066h	DLNA_GetUploadInformation()	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A	N/A
0067h	DLNA_AcceptUpload(accept, targetFileName)	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A	N/A
0068h	DLNA_SetUpdateID(valid, UpdateID)	YES	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0069h	DLNA_AcceptBrowse(accept)	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A	N/A
006Ah	DLNA_AcceptDistribution(accept)	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A	N/A
0081h	StartP2PSender(ssid, networkKey, encMode)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0082h	StartP2PReceiver(ssid, networkKey)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0083h	GetFile(requestFileName, saveFileName)	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A
0084h	ReadIDList()	N/A	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A
0085h	SelectMAC(mac)	N/A	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A
0086h	DeselectMAC(mac)	N/A	N/A	N/A	N/A	N/A	YES	N/A	N/A	N/A	N/A
0087h	SetID(id)	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0100h	PTP_SetDeviceInformation(parameterID, parameterValue)	YES	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0101h	PTP_SetHiddenObjectType(objectTypeList)	YES	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0102h	PTP_SetHiddenDirectory(directoryList)	YES	YES	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
0103h	PTP_SwitchMode(modeID)	N/A	YES	N/A	N/A	N/A	N/A	N/A	YES	YES	YES
0104h	PTP_ReceiveOperation()	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
0105h	PTP_SendResponse(data)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
0106h	PTP_SetSendDataInformation(mode, TransactionID, TotalDataLength)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
0107h	PTP_SendData(data)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
0108h	PTP_SendFile(targetFileName)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
0109h	PTP_SetReceiveDataInformation(mode)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
010Ah	PTP_ReceiveData()	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
010Bh	PTP_ReceiveFile(targetFileName)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES

010Ch	PTP_CancelData()	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
010Dh	PTP_SendEvent(data)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	YES
010Eh	PTP_StartSearch(target)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	YES
010Fh	PTP_GetInitiatorList()	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	YES	YES
0110h	DPS_ConnectPrinter(target UDN)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	N/A	N/A
0111h	PTP_StartAdvertisement(reject, GUID)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	YES
0112h	PTP_GetRejectedInitiatorList()	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	YES	YES

Table 4-4 : iSDIO Wireless LAN Command Availability

5. iSDIO Wireless LAN Configuration File

5.1 Overview

This chapter defines the Configuration File for Wireless LAN iSDIO Card. This file is stored as the "CONFIG" file under the "SD_WLAN" directory, and it is a text file encoded in [ISO 8859-1]. It is assumed a Host reads and edits this file initially, and then a Card initiates a wireless LAN or an Application based on the configuration settings described in this file.

5.2 Format

Properties

Every property has a name and a value, delimited by an equals sign (=). The name appears to the left of the equals sign. Available names and values are defined in this specification. The order of each property in the section is not defined.

```
name=value
```

Sections

Properties may be grouped into arbitrarily named sections. The section name appears on a line by itself, in square brackets ([and]). All properties after the section declaration are associated with that section. There is no explicit "end of section" delimiter; sections end at the next section declaration, or the end of the file. Sections shall not be nested. In this version, "[WLANSD]" is defined for the configuration related with the wireless LAN in this specification, and "[Vendor]" is defined for the vendor use. Other section names are reserved for future use. The order of each section in the file is not defined.

```
[section]
```

Comments

Semicolons (;) indicate the start of a comment. Comments continue to the end of the line. Everything between the semicolon and the end of the line is ignored.

```
; comment text
```

Others

Each line is terminated by the line break character *[CR][LF]* ("0D0Ah").
A property line (*name=value*) or a property value (*value*) is able to be omitted, and in this case, the default value defined in this specification is set by a Card.
A Blank line is allowable and shall be ignored.
See the example below.

(example of Configuration file)

```
[WLANSD];Configuration for Wireless LAN
;Any comment can be inserted.
ID=SMITH'S_CARD
DHCP_Enabled=YES
IP_Address=
Subnet_Mask=
Default_Gateway=
Preferred_DNS_Server=
```

```
Alternate_DNS_Server=
Proxy_Server_Enabled=YES
Proxy_Server_Name=hogehoge.com
Port_Number=8080

[Vendor];Configuration for Vendor Unique
```

5.3 Configuration for Wireless LAN

For the Wireless LAN Configuration, the section "[WLANSD]" is used in the Configuration, and the properties defined in this section are able to be set.
Note that a name or a value which is not defined in this specification is reserved for future use.

ID specifies ID of this Card. The value is equal to or less than16 ASCII characters. e.g. "SMITH'S_CARD". The default value is an empty string.

DHCP_Enabled specifies whether the DHCP Client is enabled or not; "YES": Enabled (Default), "NO": Disabled.

IP_Address specifies IP address when the DHCP Client is disabled. e.g. "192.168.0.10". Default value is the "0.0.0.0". Note that this setting is ignored if the "DHCP_Enabled=YES".

Subnet_Mask specifies subnet mask when the DHCP Client is disabled. e.g. "255.255.255.0". Default value is the "0.0.0.0". Note that this setting is ignored if the "DHCP_Enabled=YES".

Default_Gateway specifies default gateway when the DHCP Client is disabled. e.g. "192.168.0.1". Default value is the "0.0.0.0". Note that this setting is ignored if the "DHCP_Enabled=YES".

Preferred_DNS_Server specifies preferred DNS server when the DHCP Client is disabled. e.g. "192.168.0.1". Default value is the "0.0.0.0". Note that this setting is ignored if the "DHCP_Enabled=YES".

Alternate_DNS_Server specifies alternate DNS server when the DHCP Client is disabled. e.g. "192.168.0.2". Default value is the "0.0.0.0". Note that this setting is ignored if the "DHCP_Enabled=YES".

Proxy_Server_Enabled specifies whether the Proxy Server is enabled or not; "YES": Enabled, "NO": Disabled (Default).

Proxy_Server_Name specifies the name of the Proxy Server. e.g. "hogehoge.com", "123.123.0.1". Default value is the empty string. Note that this setting is ignored if the "Proxy_Server_Enabled=NO".

Port_Number specifies the port number of the Proxy Server. e.g. "8080". Default value is "8080". Note that this setting is ignored if the "Proxy_Server_Enabled=NO".

5.4 Configuration for Vendor Use

The "[Vendor]" section is reserved for vendor usage and is not defined in this section.

6. iSDIO Wireless LAN Miscellaneous

6.1 iSDIO Wireless LAN Exclusive Control

6.1.1 Overview

This subsection defines exclusive control for an iSDIO Wireless LAN Card. An iSDIO Wireless LAN Card has a file system function in the Card in order to provide some intelligent combo Card functions such as SendHTTPFileByRegister(). In these functions, the Card may read and write the file system metadata (ex. FAT, directory entry) internally. Therefore, exclusive control between Host's file system and Card's file system is needed. This subsection specifies the basic policy and rules for the exclusive control.

6.1.2 Basic Policy

The basic policy for the exclusive control is as follows.

(1) Host's file system may access the Card with CMD17, 18, 24, 25, 32, 33 and 38 during the period when an iSDIO Wireless LAN Command is not issued.

(2) When the Card is processing iSDIO Wireless LAN Command, the Host's file system access may be prohibited depending on the iSDIO Wireless LAN Command.

- If the iSDIO Wireless LAN Command accesses the flash memory stored in the Card via the Card's file system, the Host's file system access shall be prohibited (See Table 6-1 about the target iSDIO Wireless LAN Command).

(3) Host's file system shall flush file system metadata cache (ex. FAT, directory entry) stored in Host's memory to the Card before issuing the above iSDIO Wireless LAN Command.

- In order to avoid confliction between the Host's file system metadata cache and Card's file system metadata cache, the Host shall ensure that it flushes its cache to the Card before issuing the iSDIO Wireless LAN Command which accesses the flash memory via Card's file system.
- The target iSDIO Wireless LAN Commands are all commands which access the flash memory in the Card via Card's file system (See Table 6-1 about the target iSDIO Wireless LAN Command).

(4) Host's file system shall update its file system metadata cache after finishing some iSDIO Wireless LAN Commands which write file system metadata.

- Some iSDIO Wireless LAN Commands may modify file system metadata (ex. FAT, directory entry) stored in the flash memory in the Card. In order to avoid conflict between the Host's file system metadata cache and the file system metadata stored in the flash memory, Host shall update its cache by reading it out from the Card (See Table 6-1 about the target iSDIO Wireless LAN Command).

Figure 6-1 shows an example of the exclusive control in the case of a SendHTTPFileByRegister() command. This command is the target command of the above policy (2) and (3), because it accesses the flash memory in the Card via Card's file system and it reads file system metadata.

Figure 6-1 : Example of the Exclusive Control (SendHTTPFileByRegister())

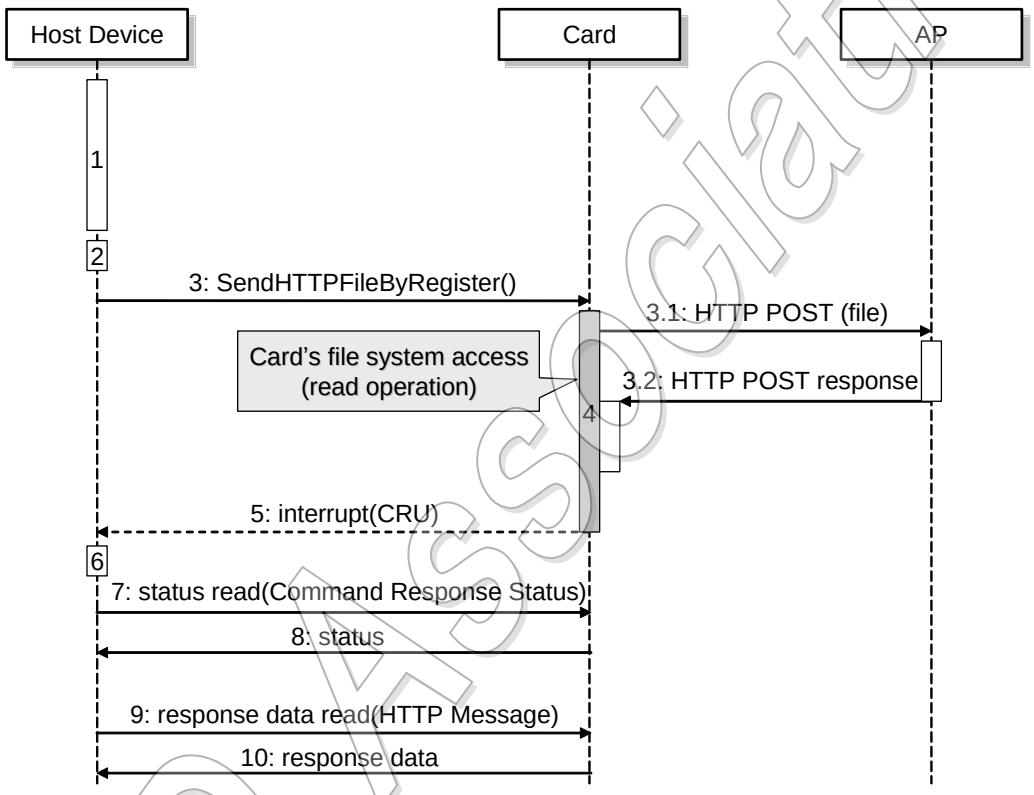
In step 1, the Host's file system may read and write the file system metadata in order to determine the target file to be uploaded to Server (**policy (1)**).

And in step 2, the Host shall flush the file system metadata cache (ex. FAT, directory entry) stored in the Host's memory to the Card, because SendHTTPFileByRegister() accesses file system metadata via Card's file system (**policy (3)**).

During the period of step 4 (from issuing SendHTTPFileByRegister() to receiving interrupt of CRU), the Host's file system shall not access to the Card, because the Card's file system accesses the file system metadata internally (**policy (2)**).

At the time of step 6, the Host's file system doesn't have to update its file system metadata cache, because Card's file system access during SendHTTPFileByRegister() is read operation (**policy (4)**). After this timing (step 6), the Host's file system may read and write the file system metadata.

In order to clarify the above timing for each access sequence, this specification defines the occurrence of the Card's file system access and its access type (read or write) for each iSDIO



Wireless LAN Command (See **6.1.3 Command Attribute for Exclusive Control**).

6.1.3 Command Attribute for Exclusive Control

Table 6-1 shows the command attributes for exclusive control.

Command ID	Command	Card's File System Access	Access Type
0001h	Scan()	No	-
0002h	Connect(ssid, networkKey)	No	-
0003h	Establish(ssid, networkKey, encMode)	No	-
0004h	WiFiDirect(wpsMode, pin)	No	-

0005h	StartWPS(ssid, wpsMode, pin)	No	-
0006h	StartWPSAP(wpsMode, pin)	No	-
0007h	Disconnect()	No	-
0011h	SetCurrentTime(currentDate, currentTime)	No	-
0012h	Abort(sequenceID)	N/A (*1)	N/A (*1)
0013h	ReadResponse(sequenceID)	No	-
0014h	SetPowerSaveMode(powerMode)	No	-
0015h	SetChannel(channelNum)	No	-
0021h	SendHTTPMessageByRegister(hostName, message)	No	-
0022h	SendHTTPFileByRegister(hostName, appendFileName, message)	Yes	Read
0023h	SendHTTPSSLMessageByRegister(hostName, message)	No	-
0024h	SendHTTPSSLFileByRegister(hostName, appendFileName, message)	Yes	Read
0025h	SendHTTPMessageByFile (hostName, messageFileName, headerRemoval)	Yes	Write
0026h	SendHTTPFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Yes	Write
0027h	SendHTTPSSLMessageByFile (hostName, messageFileName, headerRemoval)	Yes	Write
0028h	SendHTTPSSLFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Yes	Write
0029h	SetCertificate(certificate)	No	-
002Ah	SetCertificateByFile(certificateFileName)	Yes	Read
0061h	DLNA_SetDeviceInformation(parameterID, parameterValue)	No	-
0062h	DLNA_SwitchMode(modeID)	No	-
0063h	DLNA_GetDeviceList()	No	-
0064h	DLNA_StartUpload(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	Yes	Read
0065h	DLNA_PushPlayback(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	Yes	Read
0066h	DLNA_GetUploadInformation()	No	-
0067h	DLNA_AcceptUpload (accept, targetFileName)	Yes	Write
0068h	DLNA_SetUpdateID(valid, UpdateID)	No	-
0069h	DLNA_AcceptBrowse(accept)	Yes	Read

006Ah	DLNA_AcceptDistribution(accept)	Yes	Read
0081h	StartP2PSender(ssid, networkKey, encMode)	See **6.1.4 Exceptions**	
0082h	StartP2PReceiver(ssid, networkKey)		
0083h	GetFile(requestFileName, saveFileName)	Yes	Write
0084h	ReadIDList()	See **6.1.4 Exceptions**	
0085h	SelectMAC(mac)		
0086h	DeselectMAC(mac)		
0087h	SetID(id)	No	-
0100h	PTP_SetDeviceInformation(parameterID, parameterValue)	No	-
0101h	PTP_SetHiddenObjectType(objectTypeList)	No	-
0102h	PTP_SetHiddenDirectory(directoryList)	No	-
0103h	PTP_SwitchMode(modeID)	No	-
0104h	PTP_ReceiveOperation()	No	-
0105h	PTP_SendResponse(data)	No	-
0106h	PTP_SetSendDataInformation(mode, TransactionID, TotalDataLength)	No	-
0107h	PTP_SendData(data)	No	-
0108h	PTP_SendFile(targetFileName)	Yes	Read
0109h	PTP_SetReceiveDataInformation(mode)	No	-
010Ah	PTP_ReceiveData()	No	-
010Bh	PTP_ReceiveFile(targetFileName)	Yes	Write
010Ch	PTP_CancelData()	No	-
010Dh	PTP_SendEvent(data)	No	-
010Eh	PTP_StartSearch(target)	No	-
010Fh	PTP_GetInitiatorList()	No	-
0110h	DPS_ConnectPrinter(targetUDN)	No	-
0111h	PTP_StartAdvertisement(reject, GUID)	No	-
0112h	PTP_GetRejectedInitiatorList()	No	-

*1: It complies to the attribute of target command of Abort() command.

Table 6-1 : Command Attribute for Exclusive Control

[Card's File System Access]

Yes: This command accesses file system metadata via Card's file system. When the Card is processing this command, the Host's file system access shall be prohibited (**policy (2)**). And the Host's file system shall flush the file system metadata cache (ex. FAT, directory entry) stored in the Host's memory before issuing this command (**policy (3)**).

No: This command does **not** access file system metadata via Card's file system. When the

Card is processing this command, the Host's file system access shall **not** be prohibited (**policy (2)**).
And the Host's file system does **not** have to flush the file system metadata cache (ex. FAT, directory entry) stored in the Host's memory before issuing this command (**policy (3)**).

[Access Type]

Write: This command may modify the file system metadata via the Card's file system access. The Host's file system shall update its file system metadata cache after finishing this command (**policy (4)**).

Read: This command does **not** modify the file system metadata via the Card's file system access. The Host's file system does **not** have to update its file system metadata cache after finishing this command (**policy (4)**).

6.1.4 Exceptions

Basically, almost all the iSDIO Wireless LAN Commands satisfy the basic policy of exclusive control described in section "6.1.2 Basic Policy". However, in case of P2P File Transfer Command, some exceptions exist (for more details about access sequence of P2P File Transfer Command, see **Appendix D**).

[1. StartP2PSender()]

StartP2PSender() instructs the starting of the Peer-to-Peer File Transfer Sender Application. Therefore, the Peer-to-Peer File Transfer Sender Application continues to work after finishing of StartP2PSender(). And this application may access the file system metadata via the Card's file system.

This means the Host's file system access shall be prohibited during and after StartP2PSender() by the following exception rule.

Exception Rule1:

- During the period from issuing StartP2PSender() to finishing of Disconnect(), the Host's file system access shall be prohibited. During this period, ReadIDList(), SelectMAC() and DeselectMAC() may be issued.
- After finishing the above period in which Host's file system access is prohibited, Host's file system does not have to update its file system metadata cache, because Card's file system access is a read operation.

[2. StartP2PReceiver()]

StartP2PReceiver() instructs the starting of the Peer-to-Peer File Transfer Receiver Application. Therefore, the Peer-to-Peer File Transfer Receiver Application continues to work after finishing of StartP2PReceiver(). And this application may access file system metadata via the Card's file system.

This means the Host's file system access shall be prohibited during and after StartP2PReceiver() by the following exception rule.

Exception Rule2:

- During the period from issuing StartP2PReceiver() to the notification of FLU (File List Update) specified in iSDIO Status Register for Wireless LAN, the Host's file system access shall be prohibited.
- After the above notification, the Host's file system shall update its file system metadata cache, because the Card's file system access is a write operation.

At the version 1.10 of this specification, following exception for PTP use case is added.

[3. PTP Pull mode]

Issuing of the PTP_SwitchMode command with modeID "02h: PTP Pull mode" instructs the starting of the PTP Pull mode application. Therefore, the PTP Pull Application continues to work until issuing of the PTP_SwitchMode command with modeID "00h: No Application". And this application may access file system metadata via the Card's file system.

This means the Host's file system access shall be prohibited during and after issuing of the PTP_SwitchMode command with modeID "02h: PTP Pull mode" by the following exception rule.

Exception Rule3:

- During the period from issuing of the PTP_SwitchMode command with modeID "02h: PTP Pull mode" to issuing of the PTP_SwitchMode command with modeID "00h: No Application", the Host's file system access shall be prohibited.
- After finishing the above period in which Host's file system access is prohibited, Host's file system does not have to update its file system metadata cache, because Card's file system access is a read operation.

6.2 iSDIO Wireless LAN Processing Time

6.2.1 Overview

This subsection defines the maximum required time for each process. A Card shall be implemented according to these definitions, and a Host vendor is recommended to use the defined values with enough margins for the timeout values of the Host application.

6.2.2 Processing Time

6.2.2.1 Command Writing Time

This specification does not define any timeout value when a Host writes the iSDIO Command Write Data from the Command Write Register Port. That is, the Card keeps receiving the iSDIO Command Write Data until the Host finishes or aborts writing.

6.2.2.2 Response Reading Time

This specification doesn't define any timeout value when a Host reads the iSDIO Command Response Data from the Response Data Register Port. That is, the Card keeps preparing the iSDIO Command Response Data until the Host issues the next command or the Host issues the "ReadResponse" command.

6.2.2.3 Command Registration Time

The required time after a Host finishes writing the iSDIO Command Write Data before a Card registers the Command Response Status (i.e. the status whether the issued command is accepted or rejected is registered) in the Status Register shall be equal or less than 100 milliseconds.

6.2.2.4 Command Processing Time

The required time after a Card starts processing the issued command before a Card finishes processing and prepares the iSDIO Command Response Data (if the command has) is defined in Table 6-2.

Command ID	Command	Maximum Processing Time
0001h	Scan()	20 seconds
0002h	Connect(ssid, networkKey)	10 seconds
0003h	Establish(ssid, networkKey, encMode)	10 seconds
0004h	WiFiDirect(wpsMode, pin)	120 seconds (for PBC) Not defined (for PIN)
0005h	StartWPS(ssid, wpsMode, pin)	120 seconds (for PBC) Not defined (for PIN)
0006h	StartWPSAP(wpsMode, pin)	120 seconds (for PBC) Not defined (for PIN)
0007h	Disconnect()	10 seconds
0011h	SetCurrentTime(currentDate, currentTime)	1 second
0012h	Abort(sequenceID)	1 second (Note)
0013h	ReadResponse(sequenceID)	1 second
0014h	SetPowerSaveMode(powerMode)	1 second
0015h	SetChannel(channelNum)	1 second
0021h	SendHTTPMessageByRegister(hostName, message)	Not defined
0022h	SendHTTPFileByRegister(hostName, appendFileName, message)	Not defined
0023h	SendHTTPSSLMessageByRegister(hostName, message)	Not defined
0024h	SendHTTPSSLFileByRegister(hostName, appendFileName, message)	Not defined
0025h	SendHTTPMessageByFile (hostName, messageFileName, headerRemoval)	Not defined
0026h	SendHTTPFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Not defined
0027h	SendHTTPSSLMessageByFile (hostName, messageFileName, headerRemoval)	Not defined
0028h	SendHTTPSSLFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Not defined
0029h	SetCertificate(certificate)	1 second
002Ah	SetCertificateByFile(certificateFileName)	3 seconds
0061h	DLNA_SetDeviceInformation(parameterID, parameterValue)	1 second
0062h	DLNA_SwitchMode(modeID)	1 second
0063h	DLNA_GetDeviceList()	1 second
0064h	DLNA_StartUpload(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	Not defined
0065h	DLNA_PushPlayback(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	Not defined
0066h	DLNA_GetUploadInformation()	1 second
0067h	DLNA_AcceptUpload (accept, targetFileName)	Not defined

0068h	DLNA_SetUpdateID(valid, UpdateID)	1 second
0069h	DLNA_AcceptBrowse(accept)	Not defined
006Ah	DLNA_AcceptDistribution(accept)	Not defined
0081h	StartP2PSender(ssid, networkKey, encMode)	10 seconds
0082h	StartP2PReceiver(ssid, networkKey)	Not defined
0083h	GetFile(requestFileName, saveFileName)	Not defined
0084h	ReadIDList()	1 second
0085h	SelectMAC(mac)	1 second
0086h	DeselectMAC(mac)	1 second
0087h	SetID(id)	1 second
0100h	PTP_SetDeviceInformation(parameterID, parameterValue)	1 second
0101h	PTP_SetHiddenObjectType(objectTypeList)	1 second
0102h	PTP_SetHiddenDirectory(directoryList)	1 second
0103h	PTP_SwitchMode(modeID)	1 second
0104h	PTP_ReceiveOperation()	1 second
0105h	PTP_SendResponse(data)	Not defined
0106h	PTP_SetSendDataInformation(mode, TransactionID, TotalDataLength)	Not defined
0107h	PTP_SendData(data)	Not defined
0108h	PTP_SendFile(targetFileName)	Not defined
0109h	PTP_SetReceiveDataInformation(mode)	Not defined
010Ah	PTP_ReceiveData()	1 second
010Bh	PTP_ReceiveFile(targetFileName)	Not defined
010Ch	PTP_CancelData()	Not defined
010Dh	PTP_SendEvent(data)	Not defined
010Eh	PTP_StartSearch(target)	1 second
010Fh	PTP_GetInitiatorList()	1 second
0110h	DPS_ConnectPrinter(targetUDN)	Not defined
0111h	PTP_StartAdvertisement(reject, GUID)	1 second
0112h	PTP_GetRejectedInitiatorList()	1 second

Table 6-2 : Maximum Command Processing Time

Note:
If this "Abort" command is issued to abort one of the following commands to read a file in the NAND Memory module, "Maximum Processing Time" of "Abort" command shall be 3 seconds.

- SendHTTPFileByRegister
- SendHTTPSSLFileByRegister
- SetCertificateByFile
- DLNA_StartUpload

- DLNA_PushPlayback
- DLNA_AcceptBrowse
- DLNA_AcceptDistribution
- StartP2PSender
- PTP_SendFile

If this "Abort" command is issued to abort one of the following commands to write a file in the NAND Memory module, "Maximum Processing Time" of "Abort" command shall be 30 seconds.

- SendHTTPMessageByFile
- SendHTTPFileByFile
- SendHTTPSSLMessageByFile
- SendHTTPSSLFileByFile
- DLNA_AcceptUpload
- GetFile
- StartP2PReceiver
- PTP_ReceiveFile

6.3 iSDIO Wireless LAN Card Type

6.3.1 Overview

This subsection defines the three Card Types for iSDIO Wireless LAN. The Type-W Card supports the Function Extension command access by using CMD48 and CMD49 defined in [SDPart1] and the Web API-base Peer-to-Peer File Transfer function. The Type-D Card supports the SDIO command access by using CMD52 and CMD53 defined in [SDIO] and the DLNA function. The Type-P Card supports the SDIO command access by using CMD52 and CMD53 defined in [SDIO] and the PTP function. The iSDIO Wireless LAN Card shall support one of the 5(five) configurations below.

(1) Type-W only
(2) Type-D only
(3) Type-P only
(4) Type-W and Type-D
(5) Type-W and Type-P

6.3.2 Supported Functions

Type-W Card, Type-D Card and Type-P Card supports the functions as defined in Table 6-3.

Supported Function	Type-W Card Support	Type-D Card Support	Type-P card Support
SD Memory command	Mandatory	Mandatory	Mandatory
Function Extension command (CMD48 and CMD49)	Mandatory	N/A	N/A
SDIO command	N/A	Mandatory	Mandatory
Server Upload	Mandatory	Mandatory	Mandatory
DLNA	N/A	Mandatory	Mandatory
Peer-to-Peer File Transfer	Mandatory	N/A	N/A
PTP	N/A	N/A	Mandatory

Table 6-3 : Supported Functions in Type-W, Type-D and Type-P

6.3.3 Supported Commands

Type-W Card, Type-D Card and Type-P Card supports the commands as defined in Table 6-4.

Command ID	Supported Command	Type-W Card Support	Type-D Card Support	Type-P Card Support	Note
0001h	Scan()	Mandatory	Mandatory	Mandatory	
0002h	Connect(ssid, networkKey)	Mandatory	Mandatory	Mandatory	
0003h	Establish(ssid, networkKey, encMode)	Mandatory	Optional	Optional	*1
0004h	WiFiDirect(wpsMode, pin)	Optional	Mandatory	Mandatory	*2
0005h	StartWPS(ssid, wpsMode, pin)	Mandatory	Mandatory	Mandatory	
0006h	StartWPSAP(wpsMode, pin)	Optional	Optional	Optional	*3
0007h	Disconnect()	Mandatory	Mandatory	Mandatory	
0011h	SetCurrentTime(currentDate, currentTime)	Mandatory	Mandatory	Mandatory	

0012h	Abort(sequenceID)	Mandatory	Mandatory	Mandatory	
0013h	ReadResponse(sequenceID)	Mandatory	Mandatory	Mandatory	
0014h	SetPowerSaveMode(powerMode)	Mandatory	Mandatory	Mandatory	
0015h	SetChannel(channelNum)	Mandatory	Optional	Optional	*1
0021h	SendHTTPMessageByRegister(hostName, message)	Mandatory	Mandatory	Mandatory	
0022h	SendHTTPFileByRegister(hostName, appendFileName, message)	Mandatory	Mandatory	Mandatory	
0023h	SendHTTPSSLMessageByRegister(hostName, message)	Mandatory	Mandatory	Mandatory	
0024h	SendHTTPSSLFileByRegister(hostName, appendFileName, message)	Mandatory	Mandatory	Mandatory	
0025h	SendHTTPMessageByFile (hostName, messageFileName, headerRemoval)	Mandatory	Optional	Optional	*4
0026h	SendHTTPFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Mandatory	Optional	Optional	*4
0027h	SendHTTPSSLMessageByFile (hostName, messageFileName, headerRemoval)	Mandatory	Optional	Optional	*4
0028h	SendHTTPSSLFileByFile (hostName, messageFileName, appendFileName, headerRemoval)	Mandatory	Optional	Optional	*4
0029h	SetCertificate(certificate)	Mandatory	Mandatory	Mandatory	
002Ah	SetCertificateByFile(certificateFileName)	Mandatory	Optional	Optional	*4
0061h	DLNA_SetDeviceInformation(parameterID, parameterValue)	N/A	Mandatory	Mandatory	*8
0062h	DLNA_SwitchMode(modeID)	N/A	Mandatory	Mandatory	*5
0063h	DLNA_GetDeviceList()	N/A	Mandatory	Mandatory	*5
0064h	DLNA_StartUpload(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	N/A	Mandatory	Mandatory	*6
0065h	DLNA_PushPlayback(targetUDN, targetFileName, notifyFileName, MIMEType, profileID)	N/A	Mandatory	Mandatory	*7
0066h	DLNA_GetUploadInformation()	N/A	Mandatory	Mandatory	*8
0067h	DLNA_AcceptUpload (accept, targetFileName)	N/A	Mandatory	Mandatory	*8
0068h	DLNA_SetUpdateID(valid, UpdateID)	N/A	Mandatory	Mandatory	*8
0069h	DLNA_AcceptBrowse(accept)	N/A	Mandatory	Mandatory	*8
006Ah	DLNA_AcceptDistribution(accept)	N/A	Mandatory	Mandatory	*8
0081h	StartP2PSender(ssid, networkKey, encMode)	Mandatory	N/A	N/A	*9
0082h	StartP2PReceiver(ssid, networkKey)	Mandatory	N/A	N/A	*9
0083h	GetFile(requestFileName, saveFileName)	Mandatory	N/A	N/A	*9
0084h	ReadIDList()	Mandatory	N/A	N/A	*9
0085h	SelectMAC(mac)	Mandatory	N/A	N/A	*9
0086h	DeselectMAC(mac)	Mandatory	N/A	N/A	*9
0087h	SetID(id)	Mandatory	N/A	N/A	*9

0100h	PTP_SetDeviceInformation(parameterID, parameterValue)	N/A	N/A	Mandatory	*10
0101h	PTP_SetHiddenObjectType(objectTypeList)	N/A	N/A	Mandatory	*10
0102h	PTP_SetHiddenDirectory(directoryList)	N/A	N/A	Mandatory	*10
0103h	PTP_SwitchMode(modeID)	N/A	N/A	Mandatory	*10
0104h	PTP_ReceiveOperation()	N/A	N/A	Mandatory	*10
0105h	PTP_SendResponse(data)	N/A	N/A	Mandatory	*10
0106h	PTP_SetSendDataInformation(mode, TransactionID, TotalDataLength)	N/A	N/A	Mandatory	*10
0107h	PTP_SendData(data)	N/A	N/A	Mandatory	*10
0108h	PTP_SendFile(targetFileName)	N/A	N/A	Mandatory	*10
0109h	PTP_SetReceiveDataInformation(mode)	N/A	N/A	Mandatory	*10
010Ah	PTP_ReceiveData()	N/A	N/A	Mandatory	*10
010Bh	PTP_ReceiveFile(targetFileName)	N/A	N/A	Mandatory	*10
010Ch	PTP_CancelData()	N/A	N/A	Mandatory	*10
010Dh	PTP_SendEvent(data)	N/A	N/A	Mandatory	*10
010Eh	PTP_StartSearch(target)	N/A	N/A	Mandatory	*10
010Fh	PTP_GetInitiatorList()	N/A	N/A	Mandatory	*10
0110h	DPS_ConnectPrinter(targetUDN)	N/A	N/A	Mandatory	*10
0111h	PTP_StartAdvertisement(reject, GUID)	N/A	N/A	Mandatory	*10
0112h	PTP_GetRejectedInitiatorList()	N/A	N/A	Mandatory	*10

Table 6-4 : Supported Commands in Type-W,Type-D and Type-P

Note:
If the "*1" commands are supported, the "AP" bit in "WLAN Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.
If the "*2" command is supported, the "WiFiDirect" bit in "WLAN Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.
If the "*3" command is supported, the "WPS_AP" bit in "WLAN Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.
If the all of "*4" commands are supported, the "BFS" bit in "ByFile Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.
If the all of "*5" and "*6" commands are supported, "DMU" bit in "DLNA Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.
If the all of "*5" and "*7" commands shall be supported, the "DPU" bit in "DLNA Support" (**3.5 iSDIO Capability Register for Wireless LAN**) is set to ‘1b’.
If the all of "*5" and "*8" commands are supported, the "DMS" bit in "DLNA Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.
If the all of "*9" commands are supported, the "P2P" bit in "P2P File Transfer Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.
If the all of "*10" commands are supported, the "PTP" bit in "PTP Support" (**3.5 iSDIO Capability Register for Wireless LAN**) shall be set to ‘1b’.

Note that if a Host issues an optional command that a Card doesn’t support, the command is rejected by the Card.

6.4 iSDIO PTP Cancellation

6.4.1 Overview

The Figure 6-2 and Figure 6-3 show the typical command sequence of the PTP transaction cancellation.
The [PTP-IP] specification defined the PTP transaction sequence diagram from the start data packet to the end data packet. The start data packet includes the total data size in the packet header. If a device does not send the data as specified data size then a device issues the cancel processing.
The PTP transaction cancels by the "PTP_CancelData" command, "Abort" command and Cancel Event.
Note that the described sequences are typical, and a Host implementer is able to realize the sequences in other ways.

6.4.2 Cancellation of Data-In

The cancellation sequence of the Data-In phase is described below and in Figure 6-2.

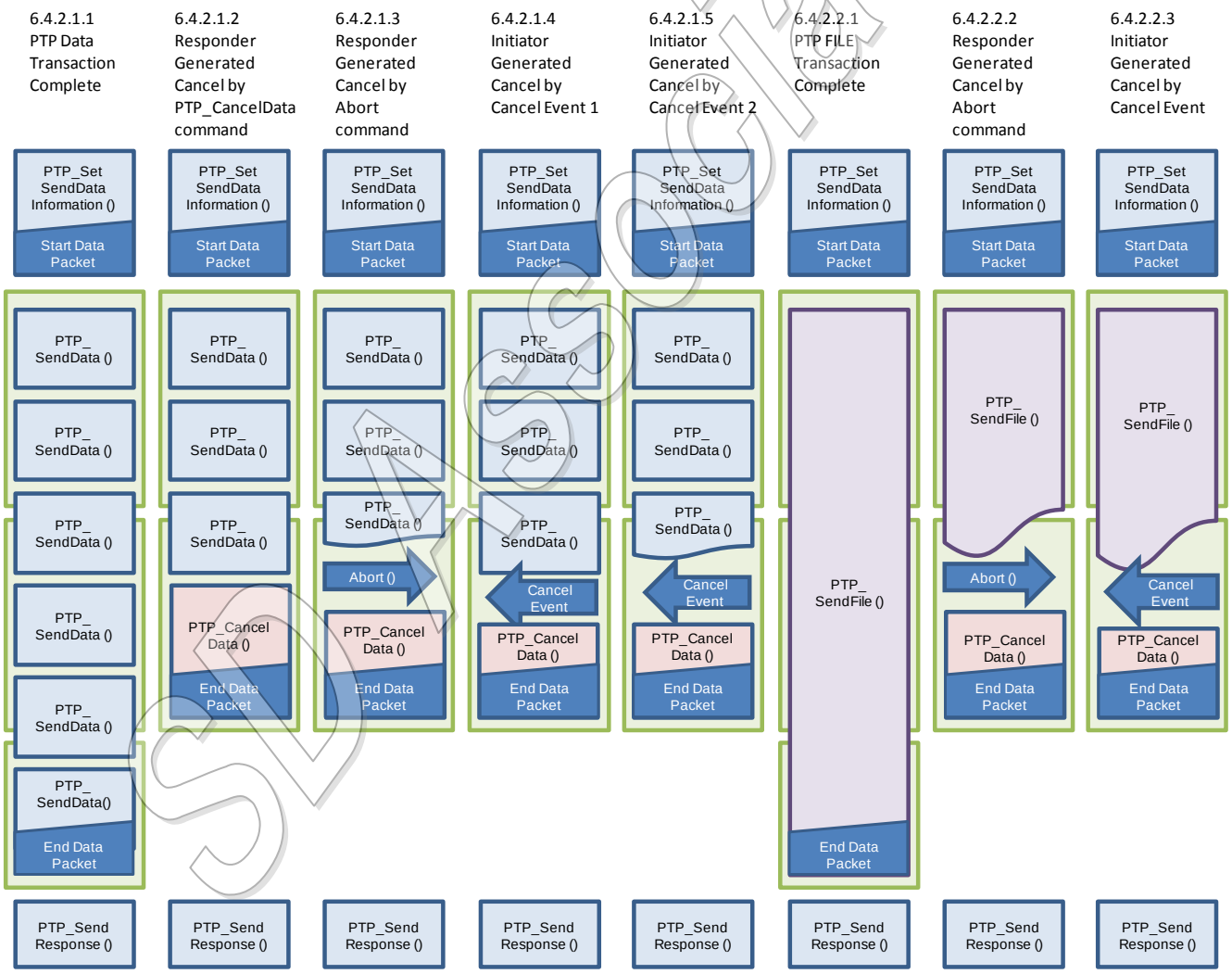


Figure 6-2 : PTP cancellation diagram of Data-In

Note that the green box in Figure 6-2 indicates the data packet from a Card to an initiator device.
The size of data in each data packet depends on the implementation of the Card.
Note that the Host application issues the "Abort" command for the cancellations of the iSDIO command.
Note that the Card receives the Cancel Event from the initiator device for cancellations of the PTP transaction.

6.4.2.1 PTP Transaction by Data

6.4.2.1.1 PTP Data Transaction Complete

This case describes successful completion of the PTP Data-In transaction between the initiator device and the responder device.

6.4.2.1.2 Responder Generated cancel by PTP_CancelData command (Data)

In this case, the PTP Data-In transaction is cancelled by the "PTP_CancelData" command in the transferring of the data packet of PTP transaction.
The Card shall send the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.2.1.3 Responder Generated cancel by Abort command (Data)

In this case, the PTP Data-In transaction is cancelled by the "Abort" command in the processing of the "PTP_SendData" command.
The Card shall send the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.2.1.4 Initiator Generated cancel by Cancel Event 1 (Data)

In this case, the Card receives the cancel event from the initiator device in the transferring of the data packet of PTP transaction.
The Card shall send the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.2.1.5 Initiator Generated cancel by Cancel Event 2 (Data)

In this case, the Card receives the cancel event from the initiator device in the transferring of the data packet of PTP transaction and in the processing of the "PTP_SendData" command.
The Card shall send the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.2.2 PTP Transaction by File

6.4.2.2.1 PTP File Transaction Complete

This case describes successful completion of the file transfer of PTP Data-In transaction between the initiator device and the responder device.

6.4.2.2.2 Responder Generated cancel by Abort command (File)

In this case, the file transfer of PTP Data-In transaction is cancelled by the "Abort" command in the processing of the "PTP_SendFile" command.
The Card shall send the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.2.2.3 Initiator Generated cancel by Cancel Event (File)

In this case, the Card receives the cancel event from the initiator device in the transferring of the data packet of PTP transaction and in the processing of the "PTP_SendFile" command.
The Card shall send the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.3 Cancellation of Data-Out

The cancellation sequence of the Data-Out phase is described below and in Figure 6-3.

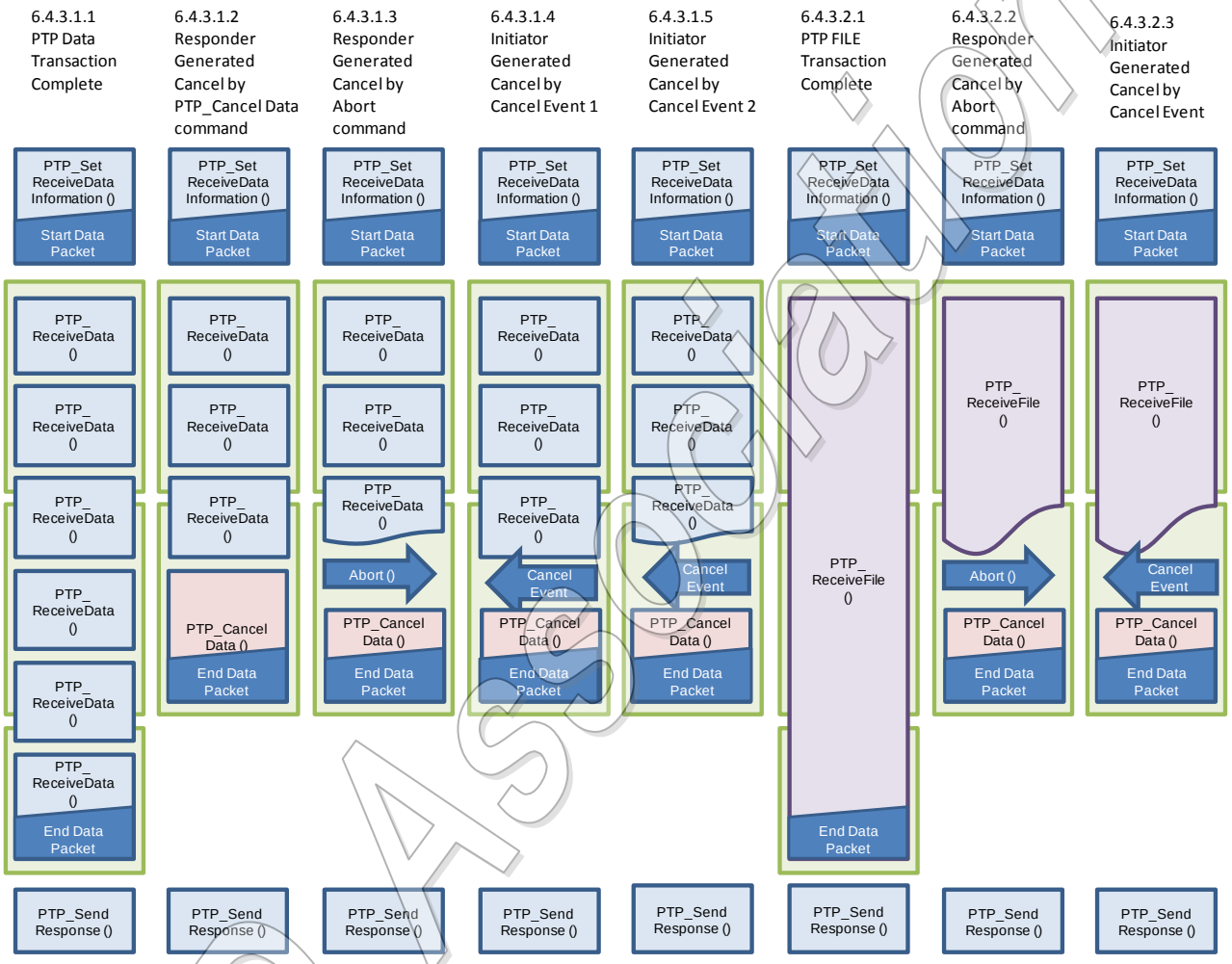


Figure 6-3 : PTP cancellation diagram of Data-Out

Note that the green box in Figure 6-3 indicates the one data packet from an initiator device to a Card. The size of data in each data packet depends on the implementation of the initiator device.
Note that the Host application issues the "Abort" command for the cancellations of the iSDIO command.
Note that the Card receives the Cancel Event from the initiator device for cancellations of the PTP transaction.

6.4.3.1 PTP Transaction by Data

6.4.3.1.1 PTP Data Transaction Complete

This case describes successful completion of the PTP Data-Out transaction between the initiator device and the responder device.

6.4.3.1.2 Responder Generated cancel by PTP_CancelData command (Data)

In this case, the PTP Data-Out transaction is cancelled by the "PTP_CancelData" command in the transferring of the data packet of PTP transaction.
The Card shall receive the remaining data (dummy data) from the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.3.1.3 Responder Generated cancel by Abort command (Data)

In this case, the PTP Data-Out transaction is cancelled by the "Abort" command in the processing of the "PTP_ReceiveData" command.
The Card shall receive the remaining data (dummy data) from the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.3.1.4 Initiator Generated cancel by Cancel Event 1 (Data)

In this case, the Card receives the cancel event from the initiator device in the transferring of the data packet of PTP transaction.
The Card shall receive the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.3.1.5 Initiator Generated cancel by Cancel Event 2 (Data)

In this case, the Card receives the cancel event from the initiator device in the transferring of the data packet of PTP transaction and in the processing of the "PTP_ReceiveData" command.
The Card shall receive the remaining data (dummy data) from the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.3.2 PTP Transaction by File

6.4.3.2.1 PTP File Transaction Complete

This case describes successful completion of the file transfer of PTP Data-Out transaction between the initiator device and the responder device.

6.4.3.2.2 Responder Generated cancel by Abort command (File)

In this case, the file transfer of PTP Data-Out transaction is cancelled by the "Abort" command in the processing of the "PTP_ReceiveFile" command.
The Card shall receive the remaining data (dummy data) from the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

6.4.3.2.3 Initiator Generated cancel by Cancel Event (File)

In this case, the Card receives the cancel event from the initiator device in the transferring of the data packet of PTP transaction and in the processing of the "PTP_ReceiveFile" command.
The Card shall receive the remaining data (dummy data) to the initiator device until the specified data size in the header of data packet in the processing of the "PTP_CancelData" command.

Appendix A (Normative) : Reference

A.1 Reference

This specification refers the following documents.
Note that this specification requires at least the functions defined in the normative references to realize the application and functions defined in this specification.

[SDPart1]	SD Specifications Part 1 Physical Layer Specification Version 4.10 or later
[SDPart2]	SD Specifications Part 2 File System Specification Version 3.00 or later
[SDIO]	SD Specifications Part E1 SDIO Specification Version 4.00 or later
[iSDIO]	SD Specifications Part E7 iSDIO Specification Version 1.10
[IEEE802.11]	IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications (IEEE Std 802.11™-2007), and Amendment 5: Enhancements for Higher Throughput (IEEE Std 802.11n™-2009)
[WiFiDirect]	Wi-Fi Peer-to-Peer (P2P) Technical Specification Version 1.1
[WPS]	Wi-Fi Simple Configuration Technical Specification v2.0.0
[WPA]	Wi-Fi Protected Access (WPA) Enhanced Security Implementation Based on IEEE P802.11i standard Version 3.1 August, 2004
[HTTP]	RFC2616
[HTTPS]	RFC2818
[SSL]	The SSL Protocol Version 3.0
[TLS]	RFC2246 and RFC3268
[TCP]	RFC793
[IP]	RFC791
[DHCP]	RFC2131
[ISO 8859-1]	ISO/IEC 8859-1
[DLNA]	DLNA Networked Device Interoperability Guidelines expanded: August 2009
[X.509]	RFC3280
[UTF]	ISO/IEC10646
[DCF]	Design rule for Camera File system: DCF Version 2.0 (Edition 2010)
[PTP-IP]	"Picture Transfer Protocol" over TCP/IP networks CIPA DC-005-2005
[MTP/IP]	Network Association MTP Extension
[DPS over IP]	Implementation Guidelines for DPS over IP CIPA DCG-006-2012
[Multicast	Multicast DNS draft-cheshire-dnsext-multicastdns-15

DNS]
[UPnP] UPnP Device Architecture 1.0

Appendix B (Normative) : Special Terms

B.1 Terminology

Card	An SD card compliant with Wireless LAN Addendum specifications
Host	An SD host device compliant with Wireless LAN Addendum specifications
[iSDIO] Command	A command issued from a Host to a Card via the iSDIO Command Write Register Port
[iSDIO] Status	Status information read from a Card for a Host via the iSDIO Status Register
[iSDIO] Command Write Data	The data format for a Host to issue an iSDIO Command to a Card
[iSDIO] Command Response Data	The data format for a Host to read the Response Data corresponding to the issued iSDIO Command from a Card
[iSDIO] Status Register	The register for a Host to read/write the iSDIO Status from/to a Card
Command Write Register Port	The data port for a Host to write the iSDIO Command Write Data
Response Data Register Port	The data port for a Host to read from the iSDIO Command Response Data
Capability Register	The register for a Host to read the capability of a Card
CMD48	A Function Extension command to read the data in the iSDIO Register for the iSDIO Card, which is defined in [SDPart1]
CMD49	A Function Extension command to write the data in the iSDIO Register for the iSDIO Card, which is defined in [SDPart1]
CMD52	An SDIO command to read and write the one-byte data in the iSDIO Register for the iSDIO Card, which is defined in [SDIO]
CMD53	An SDIO command to read and write the multi-byte data in the iSDIO Register for the iSDIO Card, which is defined in [SDIO]
Type-W Card	An iSDIO Wireless LAN Card which supports the Function Extension command and the Web API-base Peer-to-Peer File Transfer function, in addition to the SD Memory command.
Type-D Card	An iSDIO Wireless LAN Card which supports the SDIO command and the DLNA function, in addition to the SD Memory command.
Type-P Card	An iSDIO Wireless LAN Card which supports the SDIO command and the DLNA function and PTP function, in addition to the SD Memory command.
SSID List	The list of SSIDs of available Access Points
Configuration File	The configuration data which is stored in the NAND Memory module

Peer-to-Peer File Transfer	The file transfer scheme in the Infrastructure mode of wireless LAN
P2P Sender	A Card to send a file to a P2P Receiver in the Peer-to-Peer File Transfer
P2P Receiver	A Card to receive a file from a P2P Sender in the Peer-to-Peer File Transfer
ID	An identifier of a Card
ID List	The list of IDs for Receiver Cards in the Peer-to-Peer File Transfer
File List	The list of available files stored in the Sender Card for the Receiver Cards in the Peer-to-Peer File Transfer
UTF-16LE	UTF-16 Little Endian without BOM

B.2 Abbreviations

NAS	Network Attached Storage
AP	Access Point mode
STA	Station mode
+UP+	Upload controller
M-DMU	Mobile Digital Media Uploader
M-DMS	Mobile Digital Media Server
DMS	Digital Media Server
+PU+	Push controller
UPnP	Universal Plug & Play
DMR	Digital Media Renderer
CDS	Content Directory Service
DER	Distinguished Encoding Rules
PTP	Picture Transfer Protocol
MTP	Media Transfer Protocol
DPS	Digital Photo Solutions for Imaging Devices
GUID	Globally Unique Identifier
UUID	Universally Unique Identifier

Appendix C (Normative) : Peer-to-Peer File Transfer Web API

C.1 Overview

This appendix defines the Peer-to-Peer File Transfer Web APIs between a Sender Card and a Receiver Card for the Peer-to-Peer File Transfer Application. See also the protocol for the Peer-to-Peer File Transfer Application described in **D.4 Peer-to-Peer File Transfer Sequence**.

C.2 P2P Web APIs

C.2.1 HTTP(ID)

To inform the Receiver ID (*ReceiverID, e.g. SMITH'S_CARD*) and the Receiver MAC Address (*ReceiverMACaddress, e.g. 001122AABBCC for "00-11-22-AA-BB-CC"*) to the Sender, the Receiver Card sends the HTTP request message to the Sender Card's IP Address (*SenderIPaddress, e.g. 192.168.0.1*). In the HTTP request message, the Request URI is set to "/command.cgi?op=0&id=*ReceiverID*&mac=*ReceiverMACaddress*".

```
GET /command.cgi?op=0&id=ReceiverID&mac=ReceiverMACaddress HTTP/1.1
Host: SenderIPaddress
```

Then the Sender Card returns the HTTP response message containing the "OK" message in the HTTP response body, which is encoded in ISO-8859-1 ([HTTP]). See the example below.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
Content-Length: 2

OK
```

C.2.2 HTTP(ACCEPT)

In accordance with the "SelectMAC" command, to inform the acceptance for the Receiver Card to access the Sender Card, the Sender Card sends the HTTP request message to the Receiver Card's IP Address (*ReceiverIPaddress, e.g. 192.168.0.2*). In the HTTP request message, the Request URI is set to "/command.cgi?op=1".

```
GET /command.cgi?op=1 HTTP/1.1
Host: ReceiverIPaddress
```

Then the Receiver Card returns the HTTP response message containing the "OK" message in the HTTP response body, which is encoded in ISO-8859-1 ([HTTP]). Note that the Receiver Card needs to be an HTTP server to receive the HTTP(ACCEPT) request.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
Content-Length: 2
```

```
OK
```

Note that when the access is requested from the Receiver Card by the HTTP(FILELIST) (**C.2.4 HTTP(FILELIST)**) including the Receiver ID and the MAC Address, the Sender Card doesn't need to send this HTTP(ACCEPT) to the Receiver Card even though the access is allowed.

C.2.3 HTTP(REJECT)

In accordance with the "DeselectMAC" command, to inform the reject for the Receiver Card to access the Sender Card, the Sender Card sends the HTTP request message to the Receiver Card's IP Address (*ReceiverIPaddress, e.g. 192.168.0.2*). In the HTTP request message, the Request URI is set to "/command.cgi?op=2".

```
GET /command.cgi?op=2 HTTP/1.1
Host: ReceiverIPaddress
```

Then the Receiver Card returns the HTTP response message containing the "OK" message in the HTTP response body, which is encoded in ISO-8859-1 ([HTTP]). The Receiver Card finishes the P2P Receiver Application after receiving the HTTP response message. Note that the Receiver Card needs to be an HTTP server to receive the HTTP(REJECT) request.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
Content-Length: 2

OK
```

Note that when the access is requested from the Receiver Card by the HTTP(FILELIST) (**C.2.4 HTTP(FILELIST)**) including the Receiver ID and the MAC Address, the Sender Card doesn't need to send this HTTP(REJECT) to the Receiver Card even though the access is disallowed.

C.2.4 HTTP(FILELIST)

To download the FILELIST file from the Sender Card, the Receiver Card sends the HTTP request message to the Sender Card's IP Address (*SenderIPaddress, e.g. 192.168.0.1*). In the HTTP request message, the Request URI is set to "/command.cgi?op=3".

```
GET /command.cgi?op=3 HTTP/1.1
Host: SenderIPaddress
```

If the Sender Card accepts the request from the Receiver Card, the Sender Card returns the HTTP response message containing the Sender's FILELIST file, which is encoded in ISO-8859-1 ([HTTP]) and whose MIME type is "text/plain". See an example below. The HTTP message body (excluding HTTP message header) is stored in the "FILELIST" file under the "SD_WLAN" directory in the Receiver Card.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
```

```
Content-Length: XXXX

WLANSD_FILELIST
/DCIM/100XXXXX/YYYY0001.JPG,123456,2011-04-21T15:52:00,/THUMB/ZZZZ0001.JPG,
/DCIM/100XXXXX/YYYY0002.JPG,2345678,,,/META/MMMM0001.TXT
/DCIM/100XXXXX/YYYY0003.JPG,345678,2011-04-21T15:55:10,,
```

If the Sender Card rejects the request from the Receiver Card, the Sender Card returns the HTTP response message containing the "REJECTED" message encoded in ISO-8859-1 ([HTTP]). See an example below. The HTTP message body (excluding HTTP message header) is stored in the "FILELIST" file under the "SD_WLAN" directory in the Receiver Card.

Then the Receiver Card finishes the P2P Receiver Application.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
Content-Length: 8

REJECTED
```

Before the Sender Card is instructed from a Host on whether to accept or reject the request from the Receiver Card, the Sender Card returns the HTTP response message containing the "SELECTING" message encoded in ISO-8859-1 ([HTTP]). See an example below.
Then the Receiver Card continues to send the HTTP request message until the Sender Card selects or rejects.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
Content-Length: 9

SELECTING
```

If the Sender Card accepts the request from the Receiver Card but the Sender's FILELIST file is not prepared, the Sender Card returns the HTTP response message containing the "NOFILELIST" message encoded in ISO-8859-1 ([HTTP]). See an example below. The HTTP message body (excluding HTTP message header) is stored in the "FILELIST" file under the "SD_WLAN" directory in the Receiver Card.
Then the Receiver Card finishes the P2P Receiver Application.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
Content-Length: 10

NOFILELIST
```

Optionally for the Receiver Card, to download the FILELIST file from the Sender Card, the Receiver Card sends the HTTP request message to the Sender Card's IP Address (*SenderIPaddress, e.g. 192.168.0.1*) with the Receiver ID (*ReceiverID, e.g. SMITH'S_CARD*) and the Receiver MAC

Address (*ReceiverMACaddress, e.g. 001122AABBCC for "00-11-22-AA-BB-CC"*)), without sending HTTP(ID) (**C.2.1 HTTP(ID)**) in advance. In the HTTP request message, the Request URI is set to "/command.cgi?op=3&id=*ReceiverID*&mac=*ReceiverMACaddress*". See the example below.

```
GET /command.cgi?op=3&id=ReceiverID&mac=ReceiverMACaddress HTTP/1.1
Host: SenderIPaddress
```

C.2.5 HTTP(FILE)

To download the file from the Sender Card, the Receiver Card sends the HTTP request message to the Sender's IP Address (*SenderIPaddress, 192.168.0.1*) specifying the file path (*filepath, e.g. DCIM/100XXXXX*) and the filename (*filename, e.g. YYYY1234.JPG*) in accordance with the received FILEISIT file. See the example below.

```
GET /filepath/filename HTTP/1.1
Host: SenderIPaddress
```

Then the Sender Card returns the HTTP response message containing the file requested by the Receiver Card. In the HTTP response message, an appropriate MIME type may be set by the Sender Card, and even if the MIME type is omitted, it is recommended the Receiver Card receive the HTTP response message. See the example below.

```
HTTP/1.1 200 OK
Content-Type: image/jpeg
Content-Length: YYYY

< JPEG data is inserted here >
```

If the Receiver Card sends the HTTP request message to download a file without acceptance by the Sender Card, the Sender Card returns the HTTP response message with 4xx class of status code. See the example below.

```
HTTP/1.1 403 Forbidden
```

C.2.6 HTTP(END)

A Receiver Card sends the HTTP request message to the Sender's IP Address (*SenderIPaddress, e.g. 192.168.0.1*) for notifying that the download is finished. In the HTTP request message, the Request URI is set to "/command.cgi?op=4". See the example below.

```
GET /command.cgi?op=4 HTTP/1.1
Host: SenderIPaddress
```

Then the Sender Card returns the HTTP response message containing the "OK" message encoded in ISO-8859-1 ([HTTP]). See the example below.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=iso-8859-1
Content-Length: 2

OK
```

Note that "op" values from ‘100’ to ‘199’ (decimal) ("op=100" to "op=199") are reserved for vendor use, and other "op" values than those defined in this specification are reserved for future use.

Appendix D (Normative) : Sequence Diagrams

D.1 Overview

This appendix shows sequence diagrams for the Server Upload Application and the Peer-to-Peer File Transfer Application (Infrastructure). Note that the described sequences are typical, and a Host implementer is able to realize the sequences in other ways.

D.2 Server Upload Sequence

A sequence diagram for the Server Upload Application is described below and in Figure D - 1.

< Steps >

[WPS] - *Optional*

1. (Host to Card) A Host requests a Card to start the WPS by issuing the "StartWPS" command via the Command Write Register Port.
   - 1.1 (Card to AP) A Card starts WPS to the AP.
     - 1.1.1 (Card to AP) The Card receives the SSID name and the network key.
   - 1.2 (Card to AP) The Card requests the Association to the AP.
   - 1.3 (Card to AP) If accepted, the Card requests IP address assignment by the DHCP.
   - 1.4 (Host to Card) The Host reads the "Command Response Status" in the Status Register and confirms the WPS has been finished successfully.
   - 1.5 (Card to Host) The Host reads the SSID and the corresponding network key via the Response Data Register Port.

Note that the [WPS] is optional for the Host if the following [No WPS] is used.

[No WPS] – *Optional*

[SSID Scan] – *Optional*

2. (Host to Card) The Host requests the Card to scan for a connectable wireless LAN by issuing the "Scan" command via the Command Write Register Port.
   - 2.1 (Card to AP) The Card scans for connectable wireless LANs.
     - 2.1.1 (AP to Card) The Card creates the SSID LIST which contains the connectable SSID names etc.
   - 2.2 (Card to Host) The Host reads the "Command Response Status" in the Status Register and confirms the scan has finished successfully.
   - 2.3 (Card to Host) The Host reads the SSID LIST via the Response Data Register Port.

Note that the [SSID Scan] is optional for the Host if the SSID and the corresponding network key are already known by the Host.

3. (Host to Card) The Host requests the Card to connect the AP by issuing the "Connect" command via the Command Write Register Port.
   - 3.1 (Card to AP) The Card requests the Association to the AP.
   - 3.2 (Card to AP) If accepted, the Card requests IP address assignment by the DHCP.
   - 3.3 (Card to Host) The Host reads the "Command Response Status" in the Status Register and confirms the connection has finished successfully.

Note that the [No WPS] is optional for the Host if the previous [WPS] is used.

[HTTP]

4. (Host to Card - *Optional*) For the "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile" or "SendHTTPSSLFileByFile" command, the Host creates a file on the NAND Memory module that contains the HTTP request message to be sent to a

server.
For the "SendHTTPMessageByRegister", "SendHTTPFileByRegister", "SendHTTPSSLMessageByRegister" or "SendHTTPSSLFileByRegister" command, this step may be omitted because the HTTP request message is stored in a Host itself.
Note that the message depends on a service and is not defined in this specification.

5. (Host to Card) The Host requests the Card to send the HTTP request message to the specified Host by issuing the "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile", "SendHTTPSSLFileByFile", "SendHTTPMessageByRegister", "SendHTTPFileByRegister", "SendHTTPSSLMessageByRegister" or "SendHTTPSSLFileByRegister" command via the Command Write Register Port.
   5.1 (Card to AP) The Card sends the HTTP request message. For the "SendHTTPFileByFile", "SendHTTPSSLFileByFile", "SendHTTPFileByRegister" or "SendHTTPSSLFileByRegister" command, the specified file is attached to the HTTP request message.
      5.1.1 (Card to AP) The Card receives the corresponding HTTP response message. For the "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile" or "SendHTTPSSLFileByFile" command, the message is stored in a file on the NAND Memory module.
   5.2 (Card to Host) The Host reads the "Command Response Status" in the Status Register and confirms the download has finished successfully.
   5.3 (Card to Host) For the "SendHTTPMessageByRegister", "SendHTTPFileByRegister", "SendHTTPSSLMessageByRegister" or "SendHTTPSSLFileByRegister" command, the Host reads the HTTP response message via the Response Data Register Port.
   For the "SendHTTPMessageByFile", "SendHTTPFileByFile", "SendHTTPSSLMessageByFile" or "SendHTTPSSLFileByFile" command, the Host reads the HTTP response message from the stored file.

Note that this sequence may be repeated depending on the service or the Host design.

6. (Host to Card) A Host requests the Card to disconnect wireless LAN by issuing the "Disconnect" command via the Command Write Register Port.
   6.1 (Card to Host) The Host reads the "Command Response Status" in the Status Register and confirms the disconnection has finished successfully.

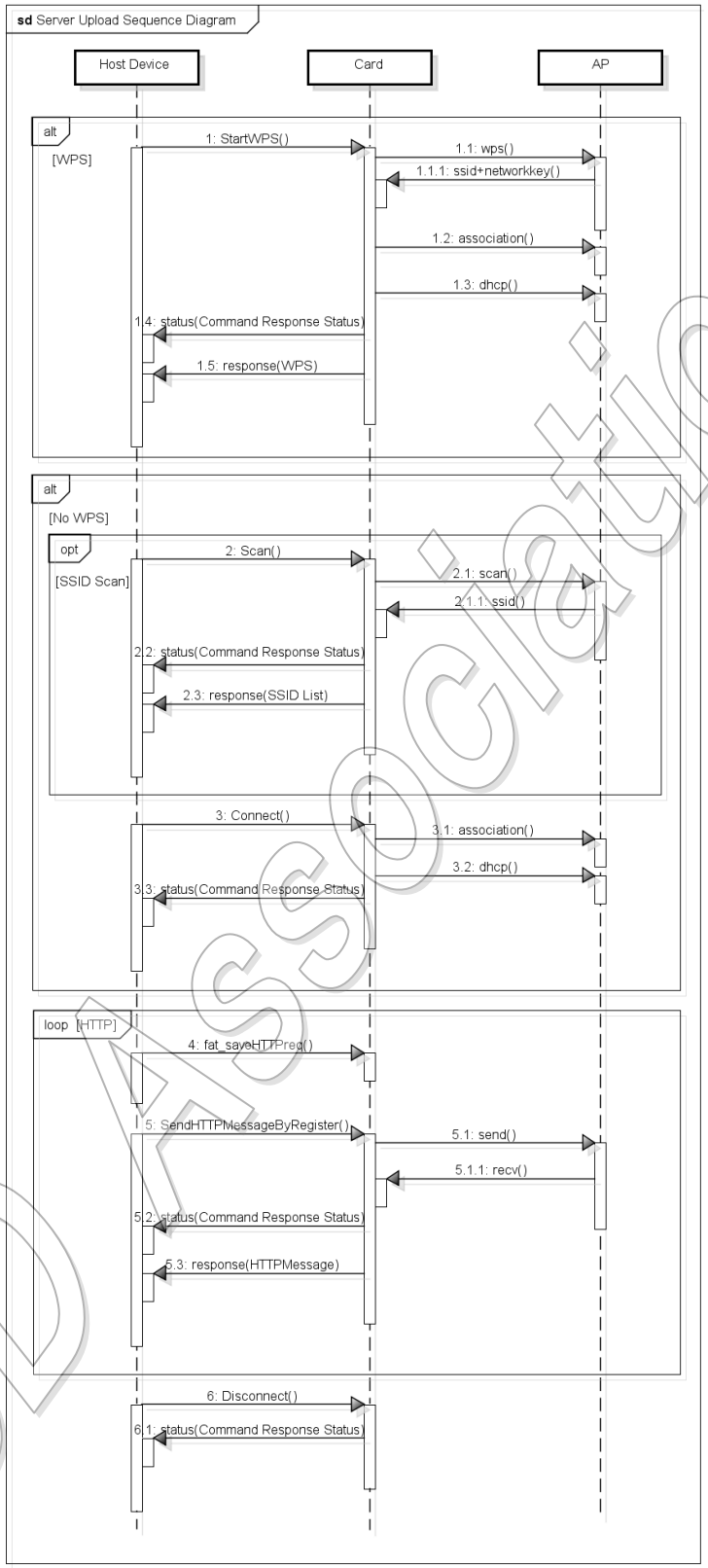


Figure D - 1 : Server Upload Sequence Diagram

D.3 DLNA Transfer Sequence

D.3.1 Card to Card Sequence Diagram

The sequence diagram for the Device Communication Content Transfer Application from Card to Card is described below and in Figure D - 2.

< Steps >

Create wireless LAN connection

[+UP+, -UP-]

A Card connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by the "WiFiDirect" command.

Content viewing

[+UP+]

1. (+UP+ Host to +UP+ Card) A +UP+ Host application requests to get a stored content data by SD Memory commands.
   1.1 (+UP+ Card to +UP+ Host) A +UP+ Card returns the specified content data.

DLNA +UP+ (File upload)

[-UP-]

2. (-UP- Host to -UP- Card) A -UP- Host application set device information for M-DMS by the "DLNA_SetDeviceInformation" command.
   2.1 (-UP- Card to -UP- Host) A -UP- Card prepares the data for the device description.

3. (-UP- Host to -UP- Card) A -UP- Host application calls the "DLNA_SwitchMode" command with M-DMS argument.
   3.1 (-UP- Card to -UP- Host) An M-DMS service starts in a receiver Card.

4. (-UP- Card to +UP+ Card) An M-DMS service sends an advertisement (NOTIFY) packet to the network.
   The following HTTP message is an example of this process.

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age = seconds until advertisement expires
LOCATION: URL for UPnP description for root device
NT: search target
NTS: ssdp:alive
USN: advertisement UUID
```

[+UP+]

5. (+UP+ Host to +UP+ Card) A +UP+ Host application sends the "DLNA_SwitchMode" command with the M-DMU argument.
   5.1 (+UP+ Card to +UP+ Host) An M-DMU service starts in a sender Card.

6. (+UP+ Card to -UP- Card) An M-DMU service sends a discovery (M-SEARCH) packet to the network.
   The following HTTP message is an example of this process.

```
M-SEARCH * HTTP/1.1
```

```
HOST: 239.255.255.250:1900
MAN: "ssdp:discover"
MX: seconds to delay response
ST: search target
```

6.1 (-UP- Card to +UP+ Card) An M-DMS service sends a responding packet to a +UP+ Card. The following HTTP message is an example of this process.

```
HTTP/1.1 200 OK
CACHE-CONTROL: max-age = seconds until advertisement expires
LOCATION: URL for UPnP description for root device
ST: search target
USN: advertisement UUID
```

6.2 (+UP+ Card to +UP+ Host) A +UP+ Card changes the DLU (Device List Update) register.

7. (+UP+ Host to +UP+ Card) A +UP+ Host application sends the "DLNA_GetDeviceList" command for receiving a -UP- Card list.
   7.1 (+UP+ Card to +UP+ Host) A +UP+ application receives a list of available -UP- Cards.

8. (+UP+ Host to +UP+ Card) A +UP+ Host application sends the "DLNA_StartUpload" command for uploading to a selected -UP- Card.
   8.1 (+UP+ Card to -UP- Card) A +UP+ Card invokes the CreateObject Action of the -UP- Card.
      8.1.1 (-UP- Card to -UP- Host) A -UP- Card changes the ULR (Upload Requested) register.
      8.1.2 (-UP- Host to -UP- Card) A -UP- Host application sends the "DLNA_GetUploadInformation" command.
      8.1.3 (-UP- Card to -UP- Host) A -UP- Card returns the information.
      8.1.4 (-UP- Host to -UP- Card) A -UP- Host application calls the "DLNA_AcceptUpload" command.
      8.1.5 (-UP- Card to +UP+ Card) A -UP- Card returns the CreateObject response.
   8.2 (+UP+ Card to -UP- Card) The selected file is transferred to a -UP- Card by HTTP POST method. The -UP- Card creates a file in order to store received content data into.
      8.2.1 (-UP- Card to -UP- Host, +UP+ Card to +UP+ Host) Each Card changes a HTTP progress register. A -UP- Card writes the received content data to the NAND Memory module.
      8.2.2 (-UP- Card to +UP+ Card) The -UP- Card sends HTTP POST response to the +UP+ Card.
      8.2.3 (-UP- Card to -UP- Host) The -UP- Card finishes the processing of the "DLNA_AcceptUpload" command.
      8.2.4 (+UP+ Card to +UP+ Host) The +UP+ Card finishes the processing of the "DLNA_StartUpload" command.

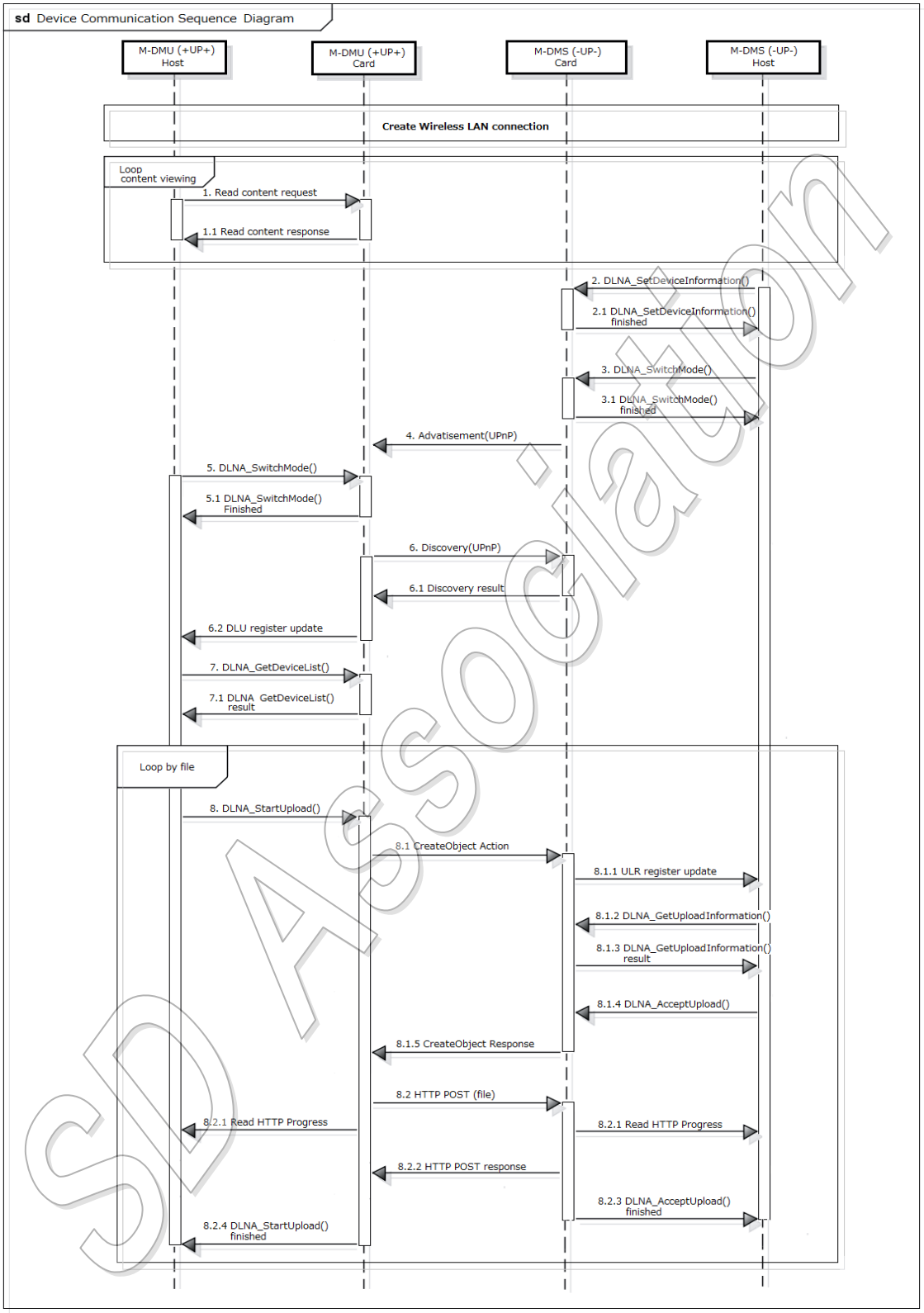


Figure D - 2 : Card to Card DLNA Upload Sequence Diagram

D.3.2 Card to DLNA Device Sequence Diagram

The sequence diagram for the Device Communication Content Transfer Application from a Card to a DLNA -UP- device (M-DMS/DMS) is described below and in Figure D - 3.

< Steps >

Create a wireless LAN connection

[-UP-]

A -UP- device connects for a wireless LAN connection via an access point.

[+UP+]

A +UP+ Card connects for a wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by the "WiFiDirect" command.

Content viewing

[+UP+]

1. (+UP+ Host to +UP+ Card) A +UP+ Host application requests to get a stored content data by SD Memory commands.
   1.1 (+UP+ Card to +UP+ Host) A +UP+ Card returns the specified content data.

DLNA +UP+ (File upload)

[-UP-]

2. (-UP- device to +UP+ Card) An M-DMS/DMS service sends an advertisement (NOTIFY) packet to the network.

[+UP+]

3. (+UP+ Host to +UP+ Card) A +UP+ Host application calls the "DLNA_SwitchMode" command with the M-DMU argument.
   3.1 (+UP+ Card to +UP+ Host) An M-DMU service starts in the sender Card.

4. (+UP+ Card to -UP- device) The M-DMU service sends a discovery (M-SEARCH) packet to the network.
   4.1 (-UP- device to +UP+ Card) An M-DMS/DMS service sends a responding packet to the sender Card.
   4.2 (+UP+ Card to +UP+ Host) The +UP+ Card changes the DLU (Device List Update) register.

5. (+UP+ Host to +UP+ Card) The +UP+ Host application sends the "DLNA_GetDeviceList" command to receive the receiver -UP- device list.
   5.1 (+UP+ Card to +UP+ Host) The +UP+ application receives a list of available -UP- device.

6. (+UP+ Host to +UP+ Card) The +UP+ Host application sends the "DLNA_StartUpload" command for uploading to the selected receiver -UP- device.
   6.1 (+UP+ Card to -UP- device) The +UP+ Card invokes the CreateObject Action of the -UP- device.
   6.2 (-UP- device to +UP Card) The -UP- device returns the CreateObject response.
   6.3 (+UP+ Card to -UP- device) The selected file is transferred to the -UP- device via the HTTP POST method.
   6.4 (+UP+ Card to +UP+ Host) The +UP+ Card changes a HTTP progress register.
   6.5 (-UP- device to +UP+ Card) The -UP- device sends HTTP POST response to the +UP+ Card.
   6.6 (+UP+ Card to +UP+ Host) The +UP+ Card finishes the processing of the "DLNA_StartUpload" command.

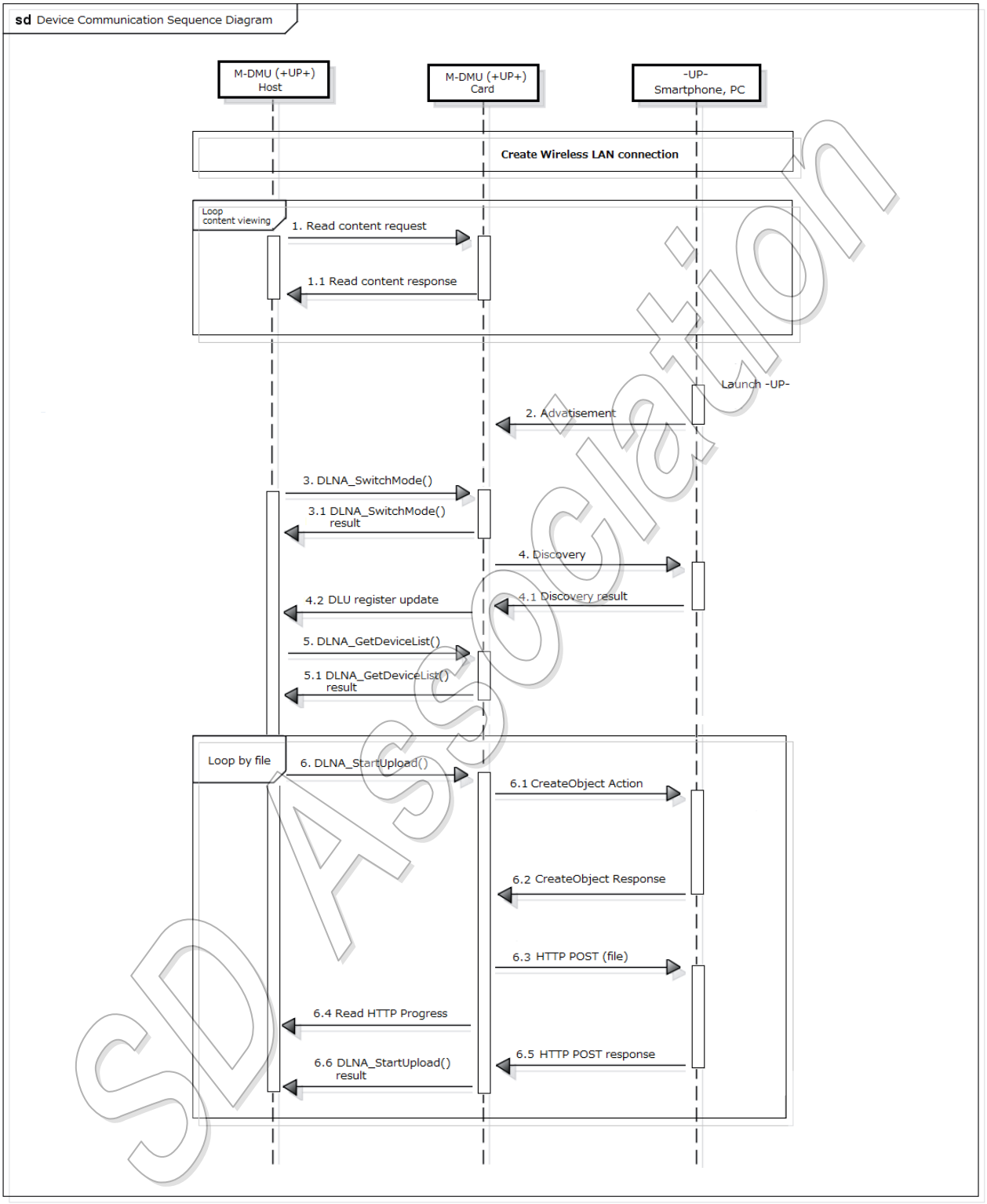


Figure D - 3 : Card to DLNA device DLNA Upload Sequence Diagram

D.3.3 Content Distribution Sequence Diagram

The sequence diagram for content distribution from a Card to a DMP device is described below and in Figure D - 4.

< Steps >

Create the Content database

[M-DMS]

An M-DMS Host application creates the content distribution database files on the Card.
Note that the content distribution database files aren't created by a Card automatically and a Host has to create them.

Create wireless LAN connection

[M-DMS]

An M-DMS Card connects for a wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by "WiFiDirect" command.

[DMP]

A DMP device connects for a wireless LAN connection via an access point.

Advertisement

[M-DMS]

1. (M-DMS Host to M-DMS Card) The M-DMS Host application sends the "DLNA_SetUpdateID" command with the UpdateID of the content distribution database.
   1.1 (M-DMS Card) The M-DMS Card receives the UpdateID of the content distribution database.
2. (M-DMS Host to M-DMS Card) The M-DMS Host application sends the "DLNA_SwitchMode" command with the M-DMS argument.
   2.1 (M-DMS Card to M-DMS Host) The M-DMS service starts in the sender Card.
3. (M-DMS Card to DLNA devices) The M-DMS service sends an advertisement (NOTIFY) packet to the network.
4. (DMP device to M-DMS Card) A DMP device sends a discovery (M-SEARCH) packet to the network.
   4.1 (M-DMS Card to DMP device) The M-DMS Card service sends a responding packet to the DMP device.

Content Browsing

[DMP]

5. (DMP device to M-DMS Card) A DMP device invokes the Browse Action of an M-DMS Card.
   5.1 (M-DMS Card to M-DMS Host) The M-DMS Card changes the CBR (Content Browse Requested) register.
   5.2 (M-DMS Host to M-DMS Card) The M-DMS Host application sends the "DLNA_AcceptBrowse" command.
   5.3 (M-DMS Card to DMP device) The M-DMS Card returns a Browse response.
   5.4 (M-DMS Card to M-DMS Host) The M-DMS Card finishes the "DLNA_AcceptBrowse" command.

Content Distribution

[DMP]

6. (DMP device to M-DMS Card) DMP device requests to get a selected file by HTTP GET method.
   6.1 (M-DMS Card to M-DMS Host) An M-DMS Card changes the CDR (Content Distribution Requested) register.

6.2 (M-DMS Host to M-DMS Card) An M-DMS Host application sends the "DLNA_AcceptDistribution" command.
6.3 (M-DMS Card to M-DMS Host) An M-DMS Card changes a HTTP progress register.
6.4 (M-DMS Card to DMP device) The M-DMS Card sends an HTTP GET response to the DMP device.
6.5 (M-DMS Card to M-DMS Host) The M-DMS Card finishes the processing of the "DLNA_AcceptDistribution" command.

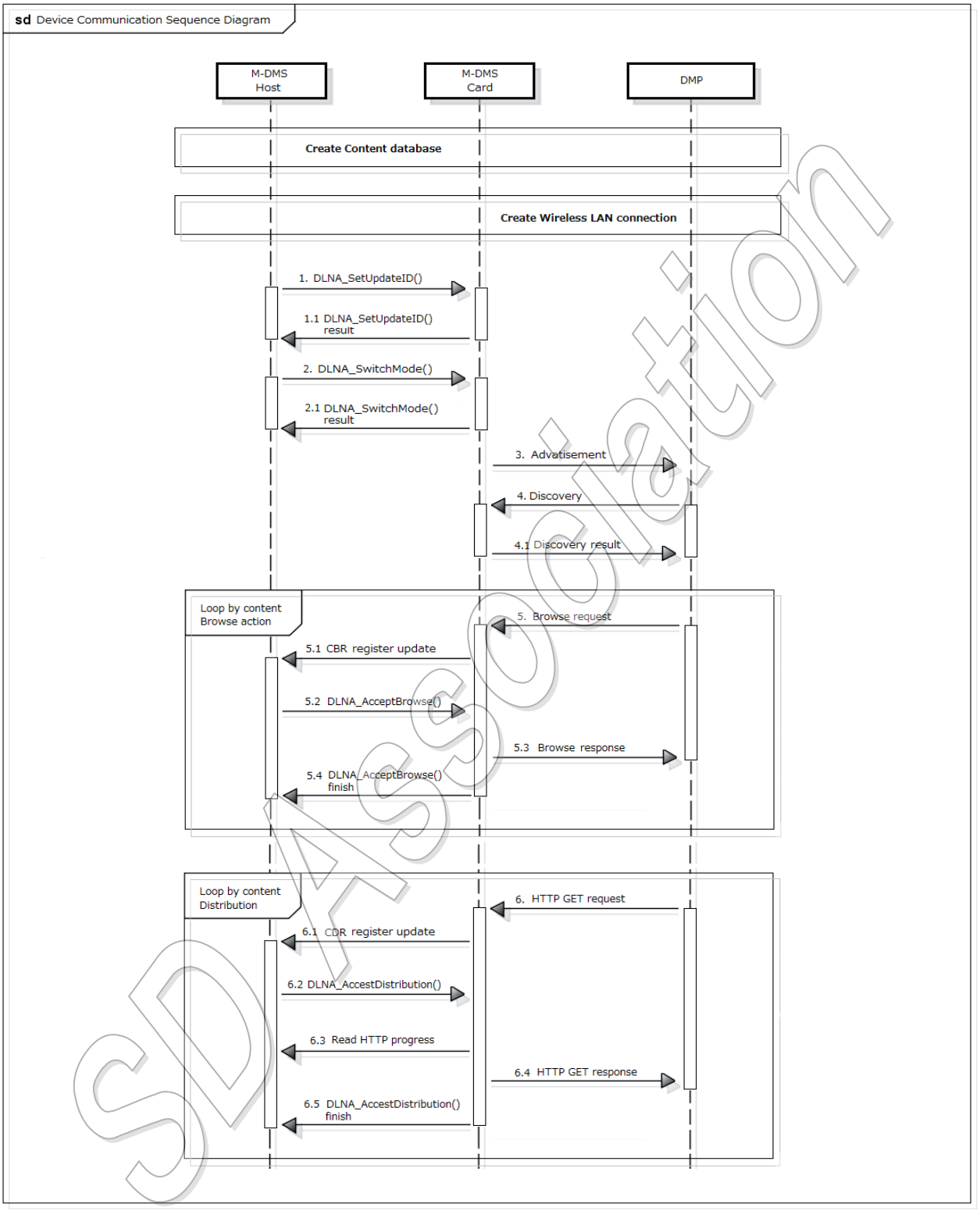


Figure D - 4 : Content Distribution Sequence Diagram

D.3.4 Push Playback Sequence Diagram

The sequence diagram for the push playback from Card to DMR device is described below and in Figure D - 5.

< Steps >

Create a wireless LAN connection

[+PU+]

A +PU+ Card connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by "WiFiDirect" command.

[DMR]

A DMR device connects for wireless LAN connection via an access point.

Content viewing

[+PU+]

1. (+PU+ Host to +PU+ Card) A +PU+ Host application requests to get a stored content data.
   1.1 (+PU+ Card to +PU+ Host) The +PU+ Card returns the specified content data.

DLNA +PU+ (push playback)

[DMR]

2. A DMR service sends an advertisement (NOTIFY) packet to the network.

[+PU+]

3. (+PU+ Host to +PU+ Card) A +PU+ Host application calls the "DLNA_SwitchMode" command with the +PU+ argument.
   3.1 (+PU+ Card to +PU+ Host) The +PU+ service starts in the sender Card.

4. (+PU+ Card to DMR device) The +PU+ service sends a discovery (M-SEARCH) packet to the network.
   4.1 The DMR service sends a responding packet to the +PU+ Card.
   4.2 (+PU+ Card to +PU+ Host) The +PU+ Card changes the DLU (Device List Update) register.

5. (+PU+ Host to +PU+ Card) A +PU+ Host application sends the "DLNA_GetDeviceList" command to get the DMR device list.
   5.1 (+PU+ Card to +PU+ Host) The +PU+ application receives a list of available DMR devices.

6. (+PU+ Host to +PU+ Card) The +PU+ Host application sends the "DLNA_PushPlayback" command for playing on a selected DMR device.
   6.1 (+PU+ Card to DMR device) The +PU+ Card invokes the SetAVTransportURI Action of the DMR device.
   6.2 (DMR device to +PU+ Card) The DMR device returns the SetAVTransportURI response.
   6.3 (+PU+ Card to DMR device) The +PU+ Card invokes the Play Action of the DMR device.
   6.4 (DMR device to +PU+ Card) The DMR device returns the Play response.
   6.5 (DMR device to +PU+ Card) The selected file is transferred to the DMR device by the HTTP GET method.
   6.6 (+PU+ Card to +PU+ Host) The +PU+ Card changes the HTTP progress register.
   6.7 (+PU+ Card to DMR device) The +PU+ Card sends the HTTP GET response to the DMR device.

(+PU+ Card to +PU+ Host) The +PU+ Card finishes the processing of the "DLNA_PushPlayback" command.

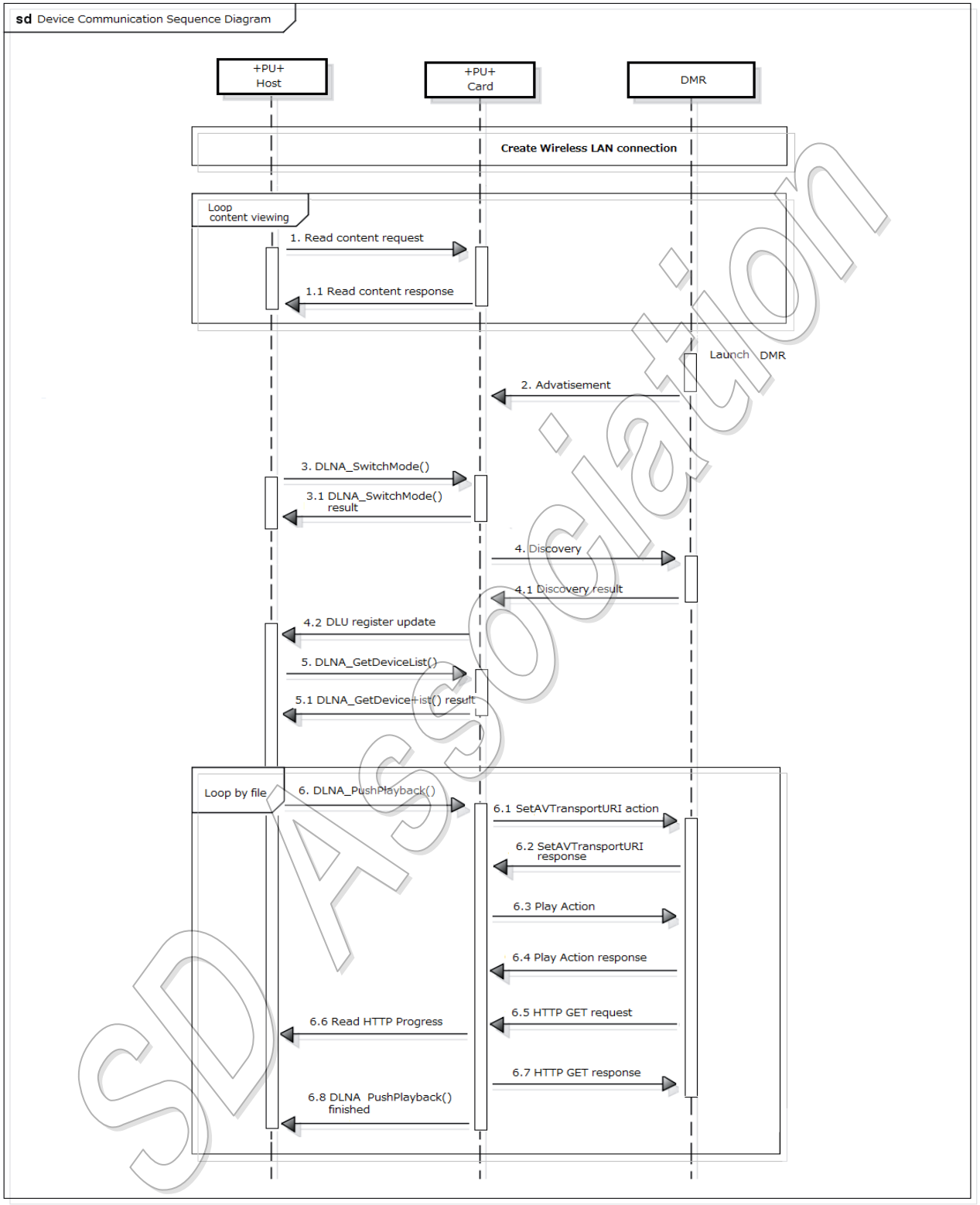


Figure D - 5 : Push Playback Sequence Diagram

D.4 Peer-to-Peer File Transfer Sequence

The sequence diagram for the Peer-to-Peer File Transfer Application (Infrastructure) is described below and in Figure D - 6.

< Steps >

[Sender]

1. (Sender Host to Sender Card) A Sender Host creates a FILELIST file on the NAND Memory module, to inform the Card which files are available for a Receiver Card.
2. (Sender Host to Sender Card) The Sender Host sets the SSID name, the network key and the authentication scheme to establish a wireless LAN connection by issuing the "StartP2PSender" command via the Command Write Register Port.
   2.1 (Sender Card to Sender Host) The Sender Host reads the "Command Response Status" in the Status Register and confirms the application has been successfully initiated.

[Receiver]

3. (Receiver Host to Receiver Card) A Receiver Host requests a Receiver Card to connect the Sender Card by issuing the "StartP2PReceiver" command via the Command Write Register Port.
   Note that it is assumed the Receiver is able to know the Sender's SSID and its network key by some other method.
   3.1 (Receiver Card to Sender Card) The Receiver Card requests the Association to the Sender Card.
   3.2 (Receiver Card to Sender Card) If accepted, the Receiver Card requests an IP address assignment from the DHCP server. The Receiver Card is able to know assigned IP Address and also know the Sender's IP Address as most preferred DNS Server as if the Sender was a DNS Server via DHCP.
   3.3 (Receiver Card to Sender Card) The Receiver Card sends the HTTP request message to notify the Receiver ID to the notified Sender's IP Address. The HTTP request message is defined in **C.2.1 HTTP(ID)**.

[Sender]

   3.3.1 (Sender Card to Sender Host) The Sender Host reads the "ID List Update" in the Status Register and confirms whether the ID List is updated.

[Receiver]

   3.3.2 (Sender Card to Receiver Card) The Receiver Card receives the HTTP response message.
   3.4 (Sender Card to Sender Host) The Sender Host reads the "Command Response Status" in the Status Register and confirms the application has been successfully initiated.

[Sender]

4. (Sender Host to Sender Card) If updated, the Sender Host issues the "ReadIDList" command via the Command Write Register Port.
   4.1 (Sender Card to Sender Host) Then the Sender Host reads the ID List via the Response Data Register Port.

5. (Sender Host to Sender Card) In the list, the Sender Host selects one or more Receiver's IDs by issuing the "SelectMAC" command via the Command Write Register Port.
   5.1 (Sender Card to Receiver Card) The Sender Card sends the HTTP request message to notify the acceptance to the specified Receiver. The HTTP request message is defined in **C.2.2 HTTP(ACCEPT)**. Note that the Receiver Card needs to be an HTTP server to

receive the HTTP(ACCEPT) request.

5'. (Sender Host to Sender Card) In the list, the Sender Host may deselect one or more Receiver's IDs by issuing the "DeselectMAC" command via the Command Write Register Port.

5'.1 (Sender Card to Receiver Card) The Sender Card sends the HTTP request message to notify the reject to the specified Receiver. The HTTP request message is defined in **C.2.3 HTTP(REJECT)**. Note that the Receiver Card needs to be an HTTP server to receive the HTTP(REJECT) request.

[Receiver]

5.1.1 (Receiver Card to Sender Card) If accepted, the Receiver Card sends the HTTP request message to download the FILELIST file in the Sender Card. The HTTP request message is defined in **C.2.4 HTTP(FILELIST)**. The Receiver Card receives the FILELIST file and stores it as the FILELIST file under the "SD_WLAN" directory on the NAND Memory module in the Receiver Card.

5.1.1.1 (Sender Card to Receiver Card) The Sender Card sends the FILELIST file to the Receiver Card.

5.1.1.1.1 (Receiver Card to Receiver Host) The Receiver Host reads the "File List Update (FLU)" in the Status Register and confirms the FILELIST file has been successfully downloaded.

5.2 (Sender Card to Sender Host) The Sender Host reads the "Command Response Status" in the Status Register and confirms the ID selection has successfully finished.

6. (Receiver Host to Receiver Card) The Receiver Host reads the "FILELIST" file on the NAND Memory module.

[Receiver: Thumbnail] - *Optional*

7. (Receiver Host to Receiver Card) The Receiver Host requests a Receiver Card to download the specified thumbnail file (according to the FILELIST file) from the Sender Card by issuing the "GetFile" command via the Command Write Register Port.

7.1 (Receiver Card to Sender Card) The Receiver Card sends the HTTP request message to download the specified thumbnail file from the Sender Card. The HTTP request message is defined in **C.2.5 HTTP(FILE)**.

7.1.1 (Sender Card to Receiver Card) The Receiver Card receives the HTTP response message and the file in the message is stored in the Card.

7.2 (Receiver Card to Receiver Host) The Receiver Host reads the "Command Response Status" in the Status Register and confirms the download has successfully finished.

Note that this routine may be repeated depending on the number of thumbnail files to be downloaded, and that the download of the thumbnail files is optional for the Application and it depends on the Host design whether this process is done.

[Receiver: File]

8. (Receiver Host to Receiver Card) The Receiver Host requests the Receiver Card to download the specified file (according to the FILELIST file) from the Sender Card by issuing the "GetFile" command via the Command Write Register Port.

8.1 (Receiver Card to Sender Card) The Receiver Card sends the HTTP request message to download the specified file from the Sender Card. The HTTP request message is defined in **C.2.5 HTTP(FILE)**.

8.1.1 (Sender Card to Receiver Card) The Receiver Card receives the HTTP response message and the file in the message is stored in the Card.

8.2 (Receiver Card to Receiver Host) The Receiver Host reads the "Command Response Status" in the Status Register and confirms the download has been finished successfully.

Note that this routine may be repeated depending on the number of files to be downloaded.

[Receiver]

9. (Receiver Host to Receiver Card) After the download for all of the target files has been completed, the Receiver Host requests the Receiver Card to disconnect the wireless LAN by issuing the "Disconnect" command via the Command Write Register Port.
   9.1 (Receiver Card to Receiver Host) The Receiver Host reads the "Command Response Status" in the Status Register and confirms the application has been successfully finished.
   9.2 (Receiver Card to Sender Card) The Receiver Card sends the HTTP request message to notify the finish of the download to the Sender Card. The HTTP request message is defined in **C.2.6 HTTP(END)**.

[Sender]

   9.2.1 (Sender Card to Sender Card) If all connected Receiver Cards have finished the download, the Sender Card is able to finish the wireless LAN establishment.

10. (Sender Card to Sender Host) A Sender Host reads the "WLAN" in the Status Register and confirms whether wireless LAN establishment is finished.

11. (Sender Host to Sender Card) If not finished, a Sender Host requests to finish the Application by issuing the "Disconnect" command via the Command Write Register Port.

    11.1 (Sender Card to Sender Host) The Sender Host reads the "Command Response Status" in the Status Register and confirms the application has been successfully finished.

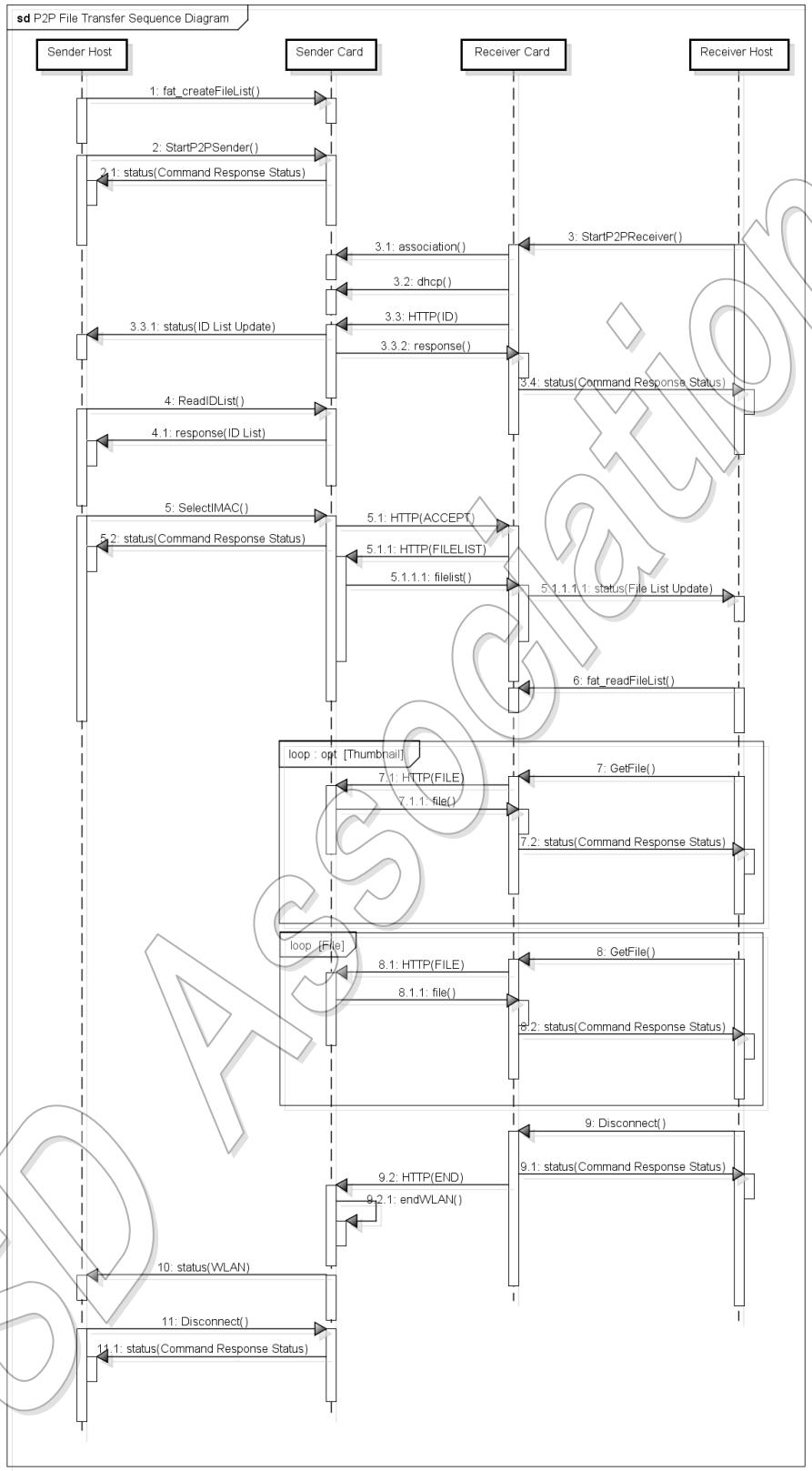


Figure D - 6 : Peer-to-Peer File Transfer Sequence Diagram

D.5 Wi-Fi Direct Sequence

The sequence diagram for the Wi-Fi Direct command is described below and in Figure D - 7.

< Steps >

[Host #1 and Host #2]

1. (Host #1 to Card #1) The Host application sends the "WiFiDirect" command.
   1.1 (Host #2 to Card #2) The Host application sends the "WiFiDirect" command.

[Card #1 and Card #2]

2. (Card #1 and Card #2) The Card starts Wi-Fi Direct device discovery process for the scan/find phase.
3. (Card #1 and Card #2) Card #1 and Card #2 select a group owner and a client by a negotiation process.
4. (Card #1 and Card #2) A client as WPS enrollee invokes a WPS process to a group owner as WPS registrar.
5. (Card #1 and Card #2) A client invokes the WPA process to a group owner.

[Host #1 and Host #2]

6. (Card #1 to Host #1) The Card completes the "WiFiDirect" command.
   6.1 (Card #2 to Host #2) The Card completes the "WiFiDirect" command.

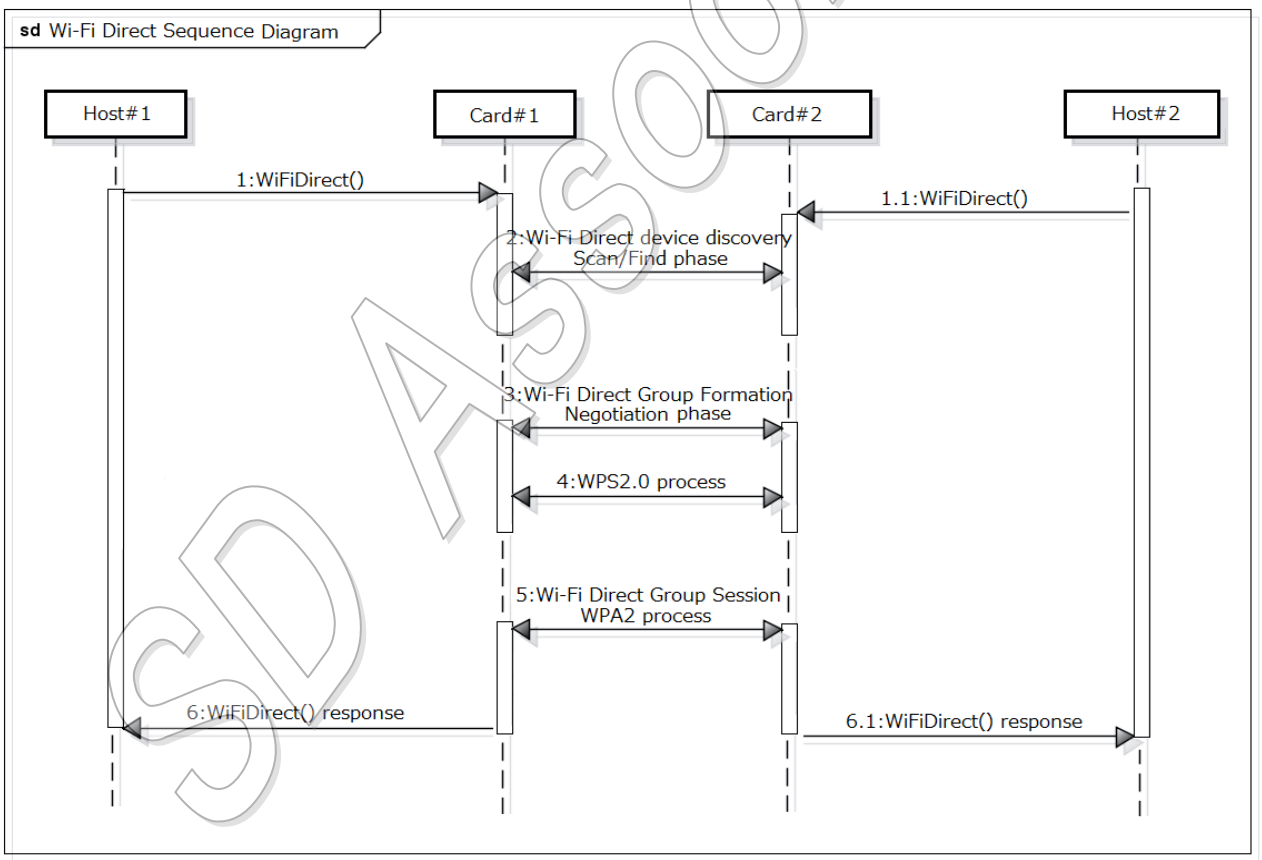


Figure D - 7 : Wi-Fi Direct Sequence Diagram

D.6 PTP Transfer Sequence

D.6.1 PTP software structure

The software structure diagram for the PTP transfer Application is described in Figure D - 8.

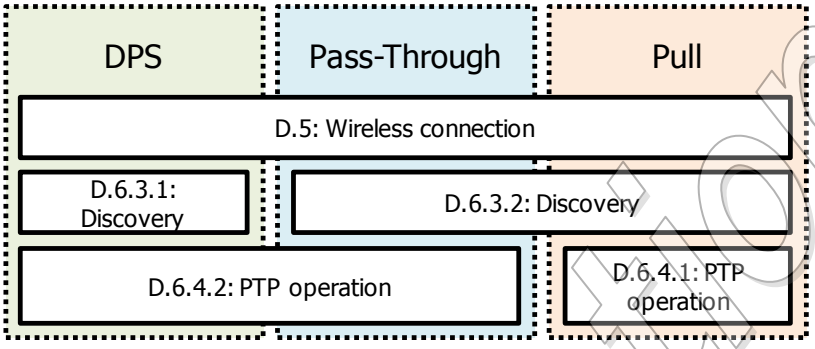


Figure D - 8 : PTP software structure

D.6.2 Wireless connection

A Card connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by "WiFiDirect" command.
The sequence diagram for the "WiFiDirect" command is described in **D.5 Wi-Fi Direct Sequence**.

D.6.3 Device Discovery

D.6.3.1 DPS mode device discovery

In DPS mode, a Card shall carry out the UPnP device discovery process inside all the time while it operates with the DPS mode. The sequence diagram for the DPS discovery from Card to DPS is described below and in Figure D - 9.

< Steps >
Create wireless LAN connection.

[DPS Printer]
A DPS Printer connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by the "WiFiDirect" command.

[Card]
A Card connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by the "WiFiDirect" command.

1. (Host to Card) The Host application calls the "PTP_SwitchMode" command with DPS argument.
   1.1 (Card to Host) The Host application receives the result information of the "PTP_SwitchMode" command.

2. (Card to DPS Printer) The Card sends a discovery packet to the network.
   2.1 (DPS Printer to Card) The DPS Printer device service sends a responding packet to the Card.
   2.2 (Card to Host) The Card changes the CIL (Changed Initiator List) register.

3. (Host to Card) The Host application sends the "PTP_GetInitiatorList" command to receive the receiver DPS printer device list.
   3.1 (Card to Host) The Host application receives a list of available DPS printer device.

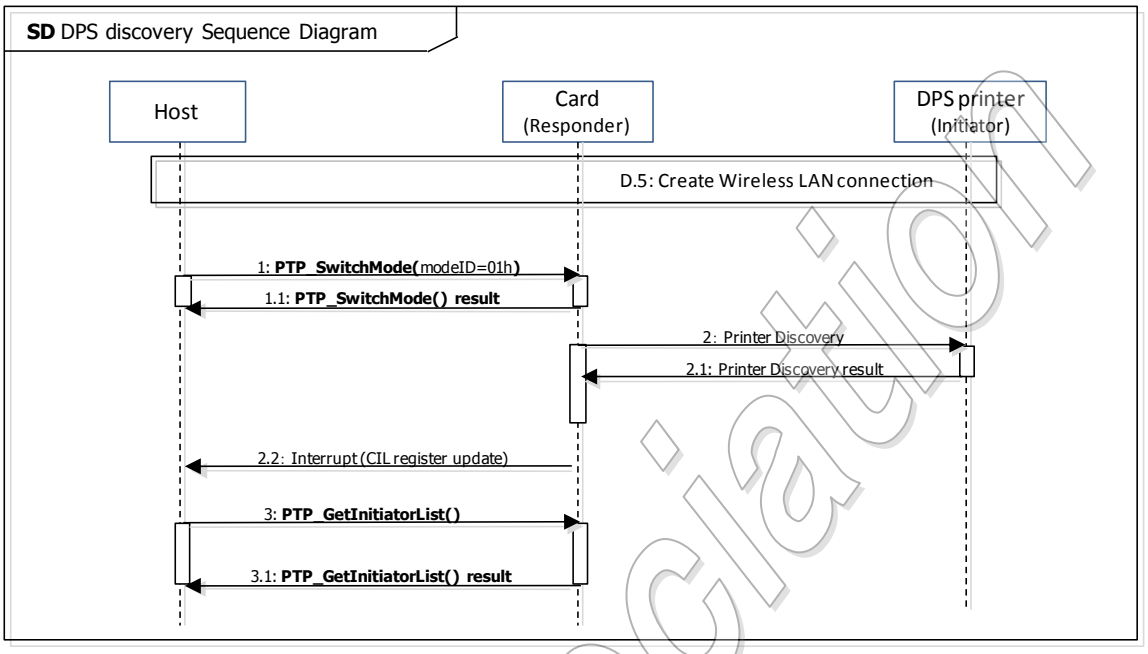


Figure D - 9 : DPS mode device discovery

D.6.3.2 PTP Pull and PTP Pass-Through mode device discovery

In PTP Pull mode and PTP Pass-Through mode, both MTP/IP (described in [MTP/IP]) and Multicast DNS (described in [Multicast DNS]) shall be used as the device discovery process (the advertisement as a PTP-IP device to initiators). The sequence diagram for the PTP Pull mode and PTP Pass-Through mode discovery from the initiator device as the target device to the Card as responder device is described below and in Figure D - 10.

< Steps >
Create wireless LAN connection.

[Target initiator device]
A Target initiator device connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by the "WiFiDirect" command.

[Card]
A Card connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by the "WiFiDirect" command.

1. (Host to Card) The Host application calls the "PTP_SetDeviceInformation" command.
   1.1 (Card to Host) The Host application receives the result information of the "PTP_SetDeviceInformation" command.

2. (Host to Card) The Host application calls the "PTP_SwitchMode" command with PTP Pull and PTP Pass-Through mode argument.
   2.1 (Card to Host) The Host application receives the result information of the "PTP_SwitchMode" command.

Note that if the Host has no necessity to search initiators, there is no necessity to execute following commands sequence.

3. (Host to Card) The Host application calls the "PTP_StartSearch" command.
   3.1 (Card to Target) The Card sends a discovery packet to the network.
   3.2 (Card to Host) The Card returns the "PTP_StartSearch" result.
   3.3 (Card to Host) The Card changes the CIL (Changed Initiator List) register.
   3.4 (Host to Card) The Host application calls the "PTP_GetInitiatorList" command.
   3.5 (Card to Host) The Host application receives the result information of the "PTP_GetInitiatorList" command.

4. (Host to Card) The Host application calls the "PTP_StartAdvertisement" command with 'reject' and 'GUID' argument.
   4.1 (Card to Host) The Card sends an advertise packet to the network.
   4.2 (Card to Host) The Host application receives the result information of the "PTP_StartAdvertisement" command.

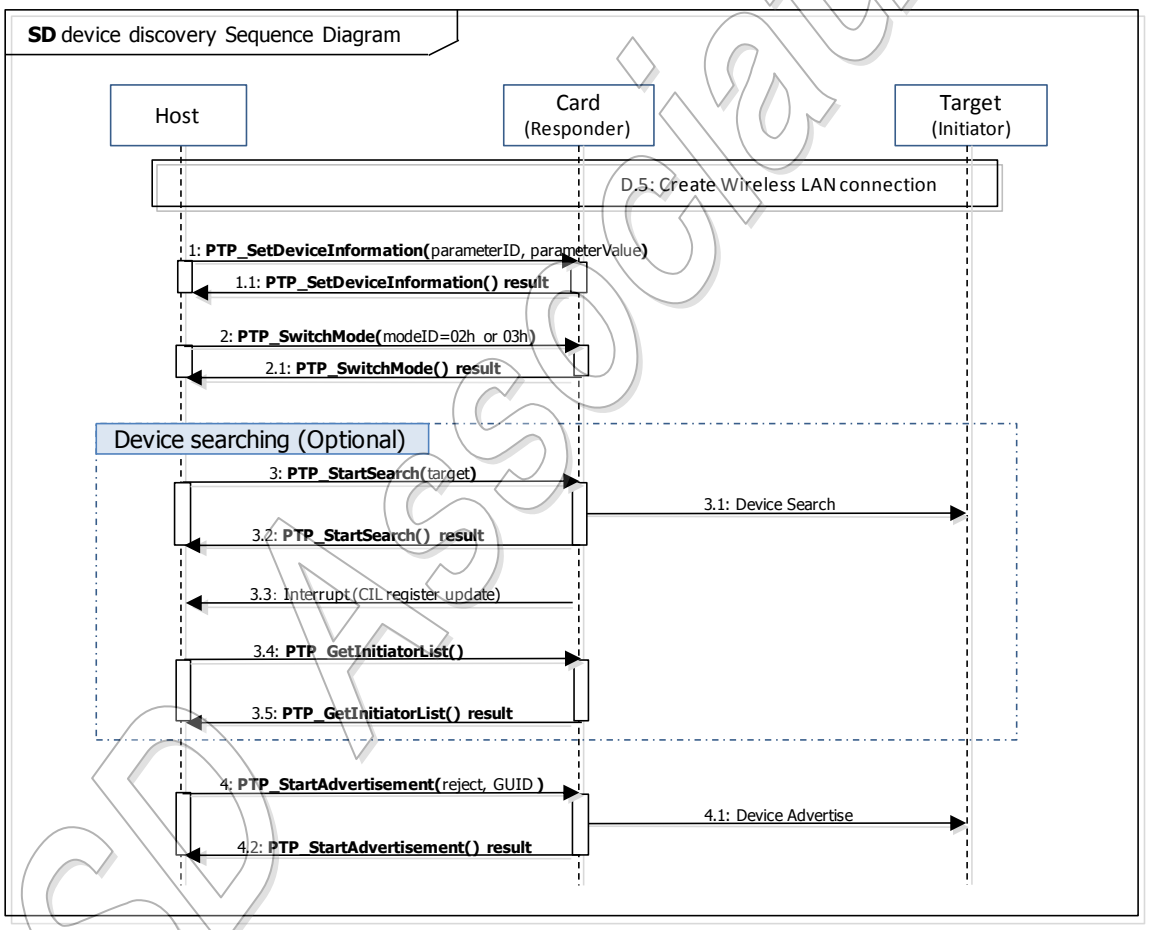


Figure D - 10 : PTP Pull and PTP Pass-Through mode device discovery

D.6.4 PTP operation

A PTP command divides as three phases. The first phase is the PTP command request received from the initiator device. The second phase is the PTP data transfer from the responder device to the initiator device, or from the initiator device to the responder device. The third phase is the PTP command response sends to the initiator device.

D.6.4.1 PTP Pull mode request and response

In PTP Pull mode, a Card receives the PTP request from the initiator device and a Card sends the PTP response to the initiator device. The sequence diagram of PTP Pull mode is described below and in Figure D - 11.

< Steps >

1. (Target to Card) The Target initiator device sends the PTP request packet to the Card.

2. (Card to Target) The Card sends the Start Data packet to the Target initiator device.
   2.1 (Card to Target) The Card sends the Data packet to the Target initiator device.
   2.2 (Card to Target) The Card sends the End Data packet to the Target initiator device.

3. (Card to Target) The Card sends the PTP response packet to the Target initiator device.

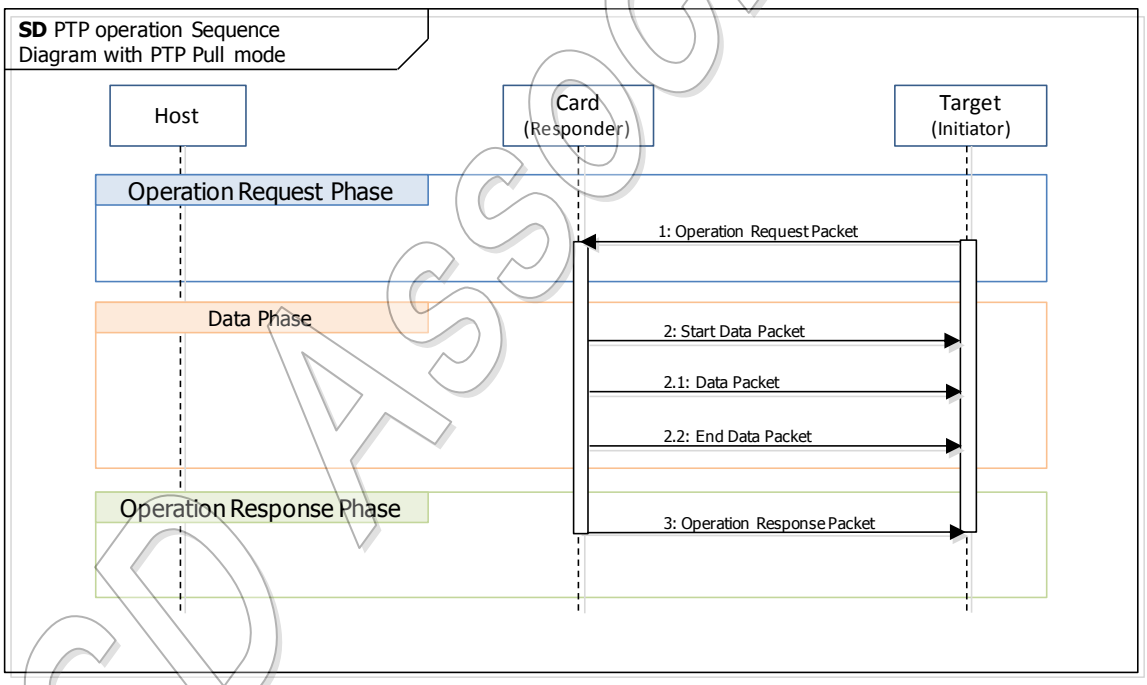


Figure D - 11 : PTP Pull mode PTP operation

D.6.4.2 DPS and PTP Pass-Through mode request and response

In DPS mode and PTP Pass-Through mode, a Host receives the PTP request from the initiator device through the Card. The sequence diagram of DPS mode and PTP Pass-Through mode are described below and in Figure D - 12.

< Steps >

1. (Target to Card) The Target initiator device sends the PTP request packet to the Card.
   1.1 (Card to Host) The Card changes the RPO (Received PTP Operation data) register.
   1.2 (Host to Card) The Host application calls the "PTP_ReceiveOperation" command.
   1.3 (Card to Host) The Host application receives the result information of the "PTP_ReceiveOperation" command with the PTP operation request.

2. (Host to Card) The Host application calls the "PTP_SendResponse" command with the response data.
   2.1 (Card to Target) The Card sends the PTP response packet to the Target.
   2.2 (Card to Host) The Host application receives the result information of the "PTP_SendResponse" command.

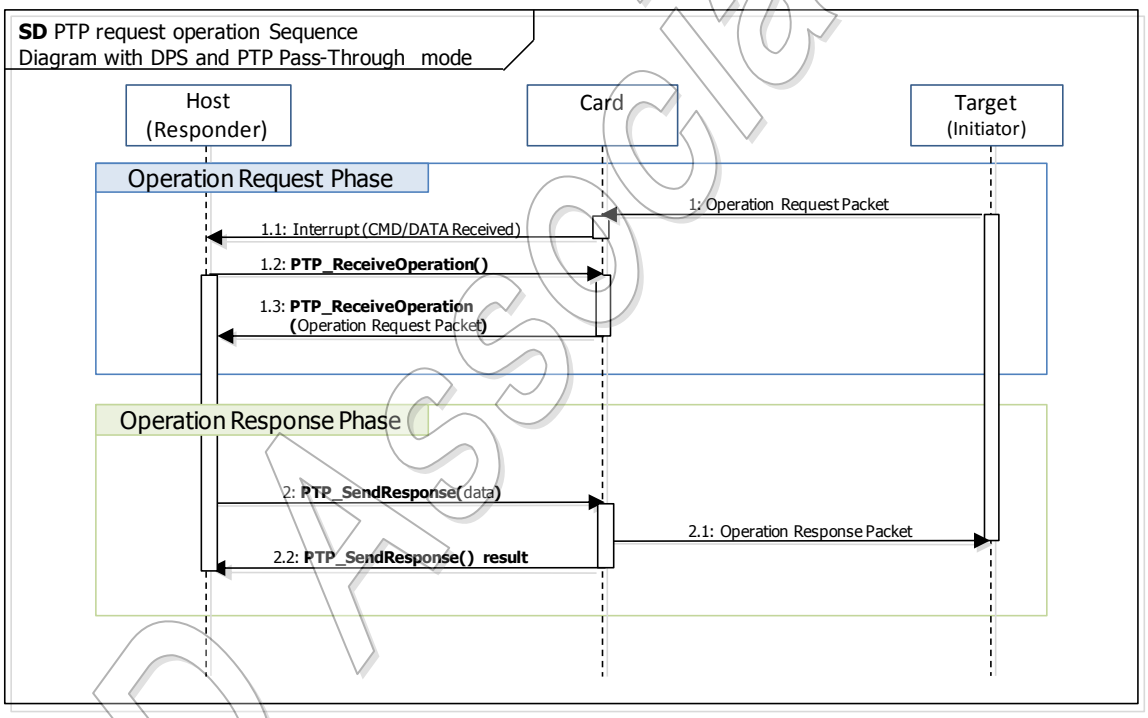


Figure D - 12 : DPS and PTP Pass-Through mode PTP operation

D.6.4.3 PTP data-in

In DPS mode and PTP Pass-Through mode, a content data transfers from the Host to the Target initiator device, and the sequence diagram is described below and in Figure D - 13.

< Steps >

1. (Target to Card) The Target initiator device sends the PTP request packet to the Card.
   - 1.1 (Card to Host) The Card changes the RPO (Received PTP Operation data) register.
   - 1.2 (Host to Card) The Host application calls the "PTP_ReceiveOperation" command.
   - 1.3 (Card to Host) The Host application receives the result information of the "PTP_ReceiveOperation" command with the PTP operation request.

2. (Host to Card) The Host application calls the "PTP_SetSendDataInformation" command with the mode parameter, TransactionID and total data size.
   - 2.1 (Card to Target) The Card sends the Start Data packet to the Target.
   - 2.2 (Card to Host) The Host application receives the result information of the "PTP_SetSendDataInformation" command.
   - 2.3 (Host to Card) The Host application calls the "PTP_SendData" command with the transfer data.
   - 2.4 (Card to Target) The Card sends the Data packet to the Target.
   - 2.5 (Card to Host) The Host application receives the result information of the "PTP_SendData" command.
   - 2.6 (Host to Card) The Host application calls the "PTP_SendData" command with the transfer data.
   - 2.7 (Card to Target) The Card sends the Data packet to the Target.
   - 2.8 (Card to Host) The Host application receives the result information of the "PTP_SendData" command.
   - 2.9 (Host to Card) The Host application calls the "PTP_SendData" command with the transfer data.
   - 2.10 (Card to Target) The Card sends the End Data packet to the Target.
   - 2.11 (Card to Host) The Host application receives the result information of the "PTP_SendData" command.

   Note that if total data size is more than "Max Size of Command Write Data" in iSDIO Capability register, the data shall be fragmented.

3. (Host to Card) The Host application calls the "PTP_SendResponse" command with the response data.
   - 3.1 (Card to Target) The Card sends the PTP response packet to the Target.
   - 3.2 (Card to Host) The Host application receives the result information of the "PTP_SendResponse" command.

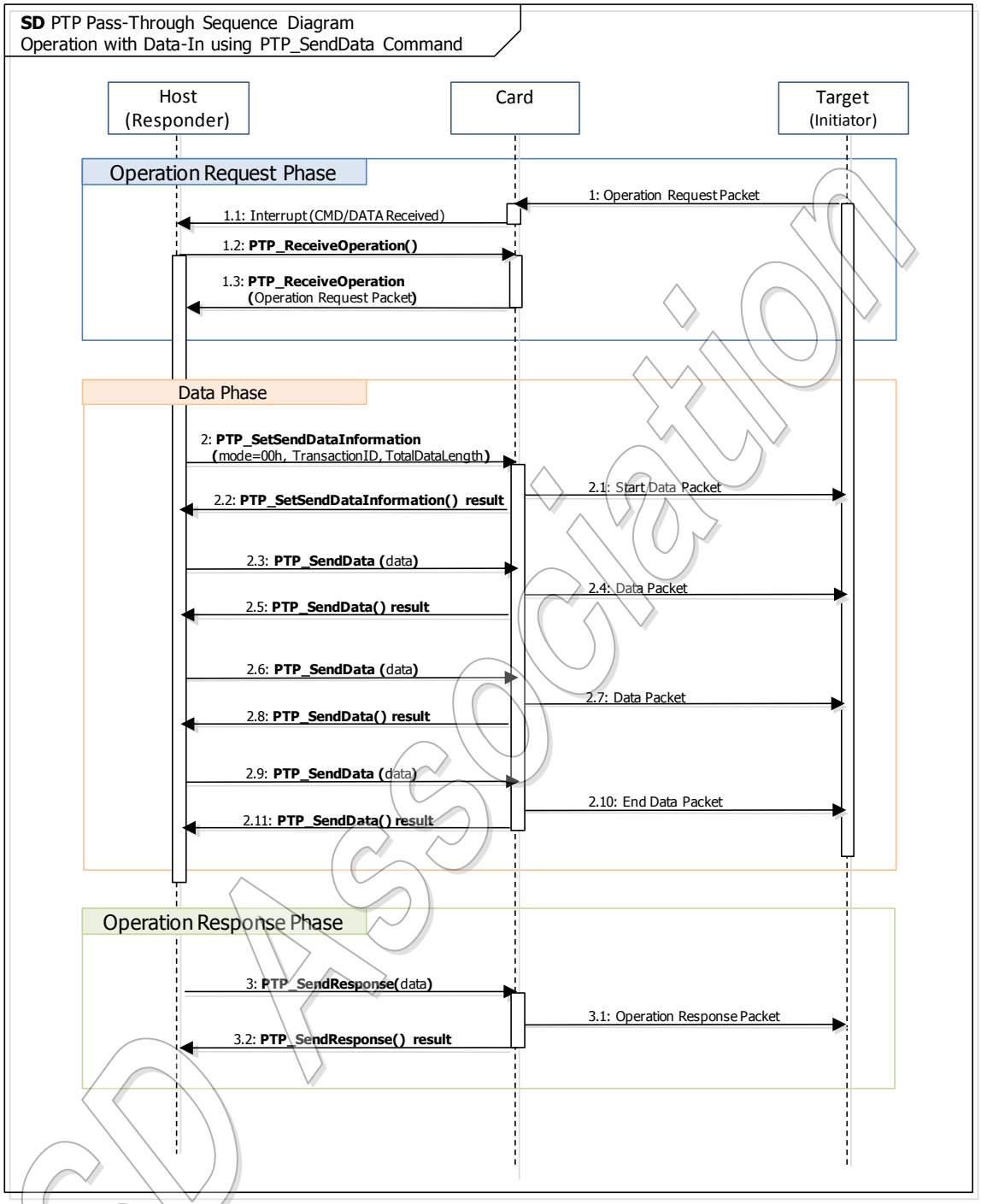


Figure D - 13 : PTP data-in Sequence Diagram

D.6.4.4 PTP data-in (file)

In DPS mode and PTP Pass-Through mode, a content file transfers from the Card to the Target initiator device, and the sequence diagram is described below and in Figure D - 14.

< Steps >

1. (Target to Card) The Target initiator device sends the PTP request packet to the Card.
   1.1 (Card to Host) The Card changes the RPO (Received PTP Operation data) register.
   1.2 (Host to Card) The Host application calls the "PTP_ReceiveOperation" command.
   1.3 (Card to Host) The Host application receives the result information of the "PTP_ReceiveOperation" command with the PTP operation request.

2. (Host to Card) The Host application calls the "PTP_SetSendDataInformation" command with the mode parameter, TransactionID and total data size.
   2.1 (Card to Target) The Card sends the Start Data packet to the Target.
   2.2 (Card to Host) The Host application receives the result information of the "PTP_SetSendDataInformation" command.
   2.3 (Host to Card) The Host application calls the "PTP_SendFile" command with the file path.
   2.4 (Card to Target) The Card sends the Data packet to the Target.
   2.5 (Card to Target) The Card sends the Data packet to the Target
   2.6 (Card to Target) The Card sends the End Data packet to the Target
   2.7 (Card to Host) The Host application receives the result information of the "PTP_SendFile" command.

3. (Host to Card) The Host application calls the "PTP_SendResponse" command with the response data.
   3.1 (Card to Target) The Card sends the PTP response packet to the Target.
   3.2 (Card to Host) The Host application receives the result information of the "PTP_SendResponse" command.

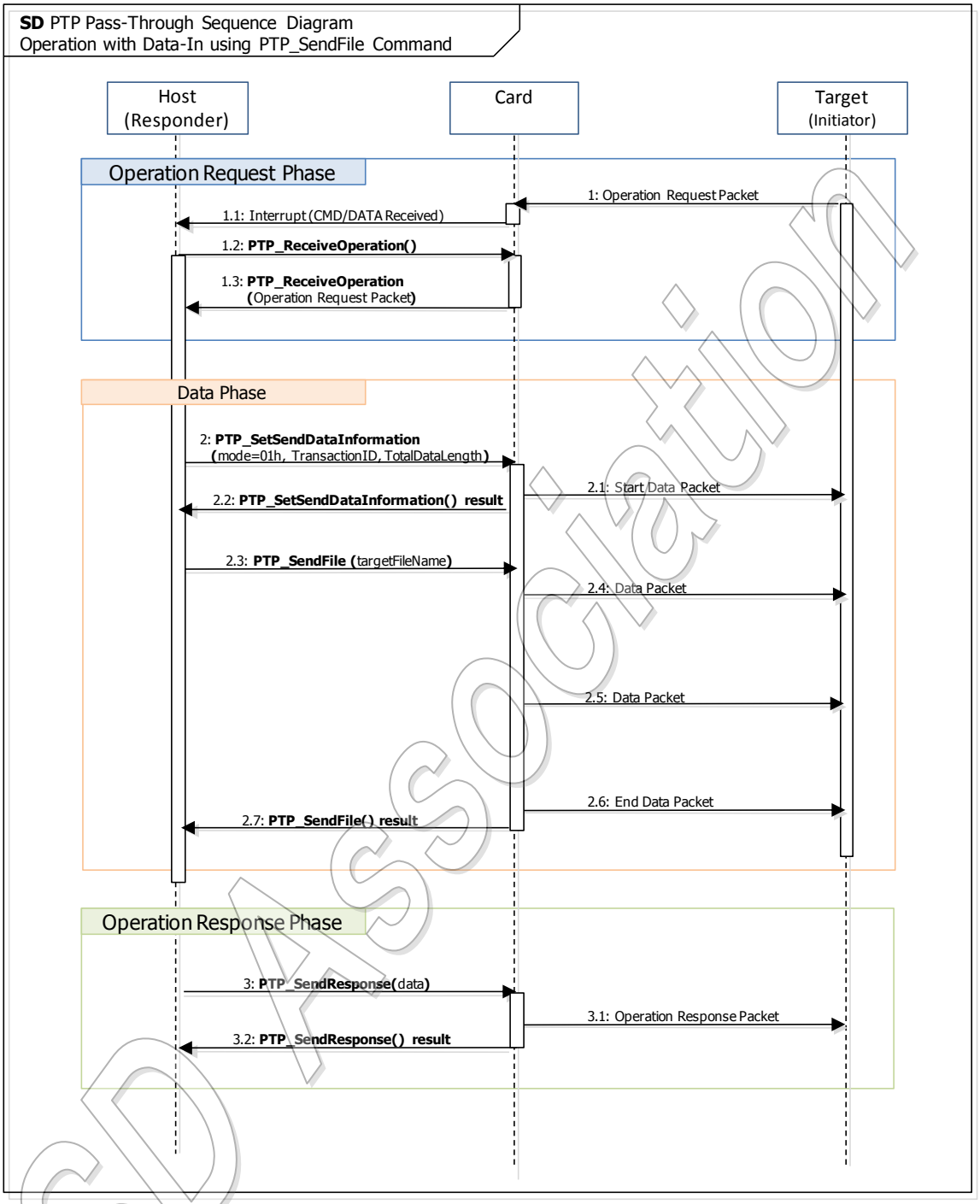


Figure D - 14 : PTP data-in (file) Sequence Diagram

D.6.4.5 PTP data-out

In DPS mode and PTP Pass-Through mode, a content data transfers from the Target initiator device to the Card, and the sequence diagram is described below and in Figure D - 15.

< Steps >

1. (Target to Card) The Target initiator device sends the PTP request packet to the Card.
   - 1.1 (Card to Host) The Card changes the RPO (Received PTP Operation data) register.
   - 1.2 (Host to Card) The Host application calls the "PTP_ReceiveOperation" command.
   - 1.3 (Card to Host) The Host application receives the result information of the "PTP_ReceiveOperation" command with the PTP operation request.

2. (Host to Card) The Host application calls the "PTP_SetReceiveDataInformation" command with the mode parameter.
   - 2.1 (Target to Card) The Card receives the Start Data packet from the Target.
   - 2.2 (Card to Host) The Host application receives the result information of the "PTP_SetReceiveDataInformation" command.
   - 2.3 (Target to Card) The Card receives the Data packet from the Target
   - 2.4 (Card to Host) The Card changes the RPD (Received PTP Data) register.
   - 2.5 (Host to Card) The Host application calls the "PTP_ReceiveData" command.
   - 2.6 (Target to Card) The Card receives the Data packet from the Target.
   - 2.7 (Card to Host) The Host application receives the result information of the "PTP_ReceiveData" command with the received Data.
   - 2.8 (Target to Card) The Card receives the Data packet from the Target.
   - 2.9 (Card to Host) The Card changes the RPD (Received PTP Data) register.
   - 2.10 (Host to Card) The Host application calls the "PTP_ReceiveData" command.
   - 2.11 (Target to Card) The Card receives the Data packet from the Target.
   - 2.12 (Card to Host) The Host application receives the result information of the "PTP_ReceiveData" command with the received Data.
   - 2.13 (Target to Card) The Card receives the End Data packet from the Target.
   - 2.14 (Card to Host) The Card changes the RPD (Received PTP Data) register.
   - 2.15 (Host to Card) The Host application calls the "PTP_ReceiveData" command.
   - 2.16 (Card to Host) The Host application receives the result information of the "PTP_ReceiveData" command with the received Data.

   Note that if total data size is more than "Max Size of Command Response Data" in iSDIO Capability register, the data shall be fragmented by the card.

3. (Host to Card) The Host application calls the "PTP_SendResponse" command with the response data.
   - 3.1 (Card to Target) The Card sends the PTP response packet to the Target.
   - 3.2 (Card to Host) The Host application receives the result information of the "PTP_SendResponse" command.

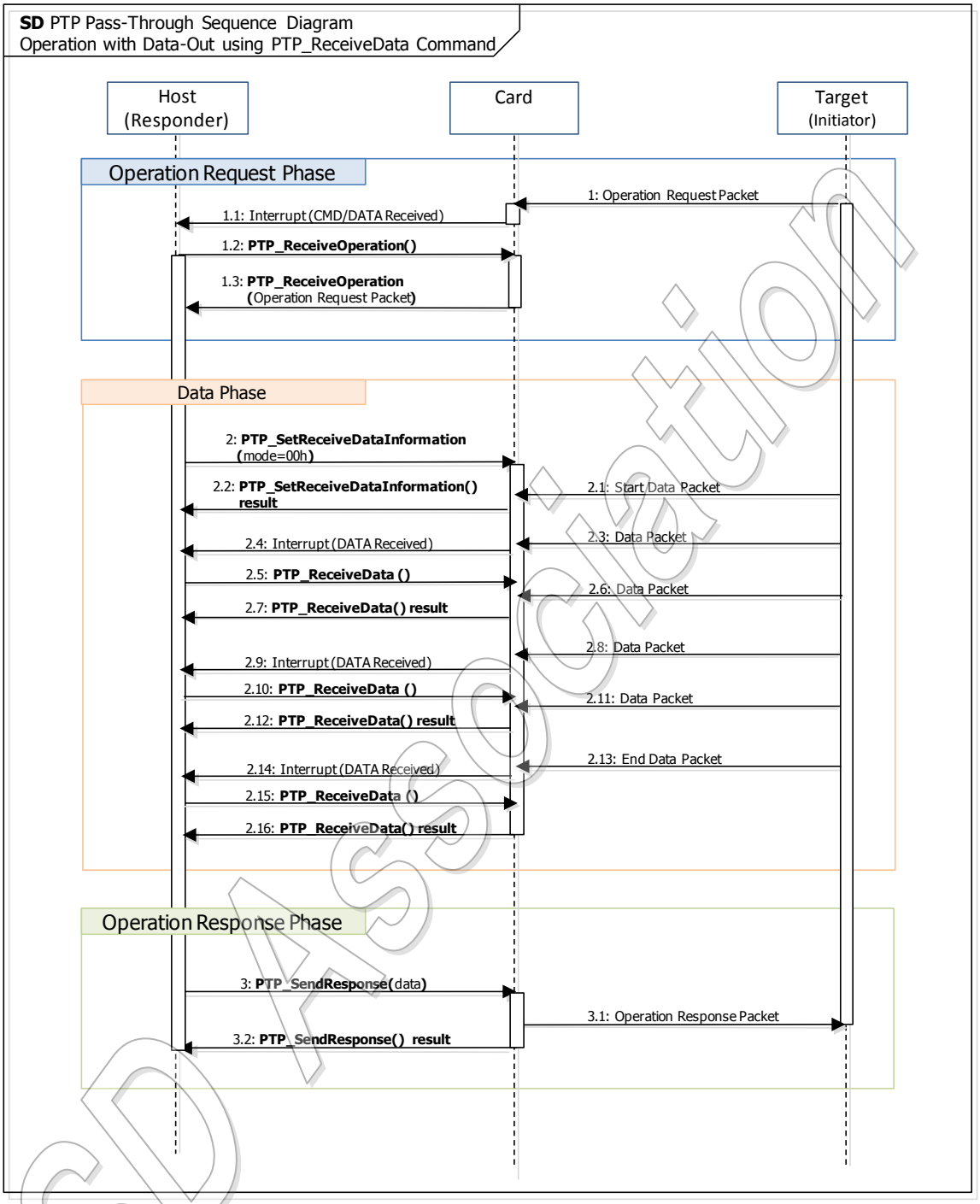


Figure D - 15 : PTP data-out Sequence Diagram

D.6.4.6 PTP data-out (file)

In DPS mode and PTP Pass-Through mode, a content file transfers from the Target initiator device to the Card, and the sequence diagram is described below and in Figure D - 16.

< Steps >

1. (Target to Card) The Target initiator device sends the PTP request packet to the Card.
   1.1 (Card to Host) The Card changes the RPO (Received PTP Operation data) register.
   1.2 (Host to Card) The Host application calls the "PTP_ReceiveOperation" command.
   1.3 (Card to Host) The Host application receives the result information of the "PTP_ReceiveOperation" command with the PTP operation request.

2. (Host to Card) The Host application calls the "PTP_SetReceiveDataInformation" command with the mode parameter.
   2.1 (Target to Card) The Card receives the Start Data packet from the Target.
   2.2 (Card to Host) The Host application receives the result information of the "PTP_SetReceiveDataInformation" command.
   2.3 (Target to Card) The Card receives the Data packet from the Target.
   2.4 (Host to Card) The Host application calls the "PTP_ReceiveFile" command.
   2.5 (Target to Card) The Card receives the Data packet from the Target.
   2.6 (Target to Card) The Card receives the Data packet from the Target.
   2.7 (Target to Card) The Card receives the Data packet from the Target.
   2.8 (Target to Card) The Card receives the End Data packet from the Target.
   2.9 (Card to Host) The Host application receives the result information of the "PTP_ReceiveFile" command.

3. (Host to Card) The Host application calls the "PTP_SendResponse" command with the response data.
   3.1 (Card to Target) The Card sends the PTP response packet to the Target.
   3.2 (Card to Host) The Host application receives the result information of the "PTP_SendResponse" command.

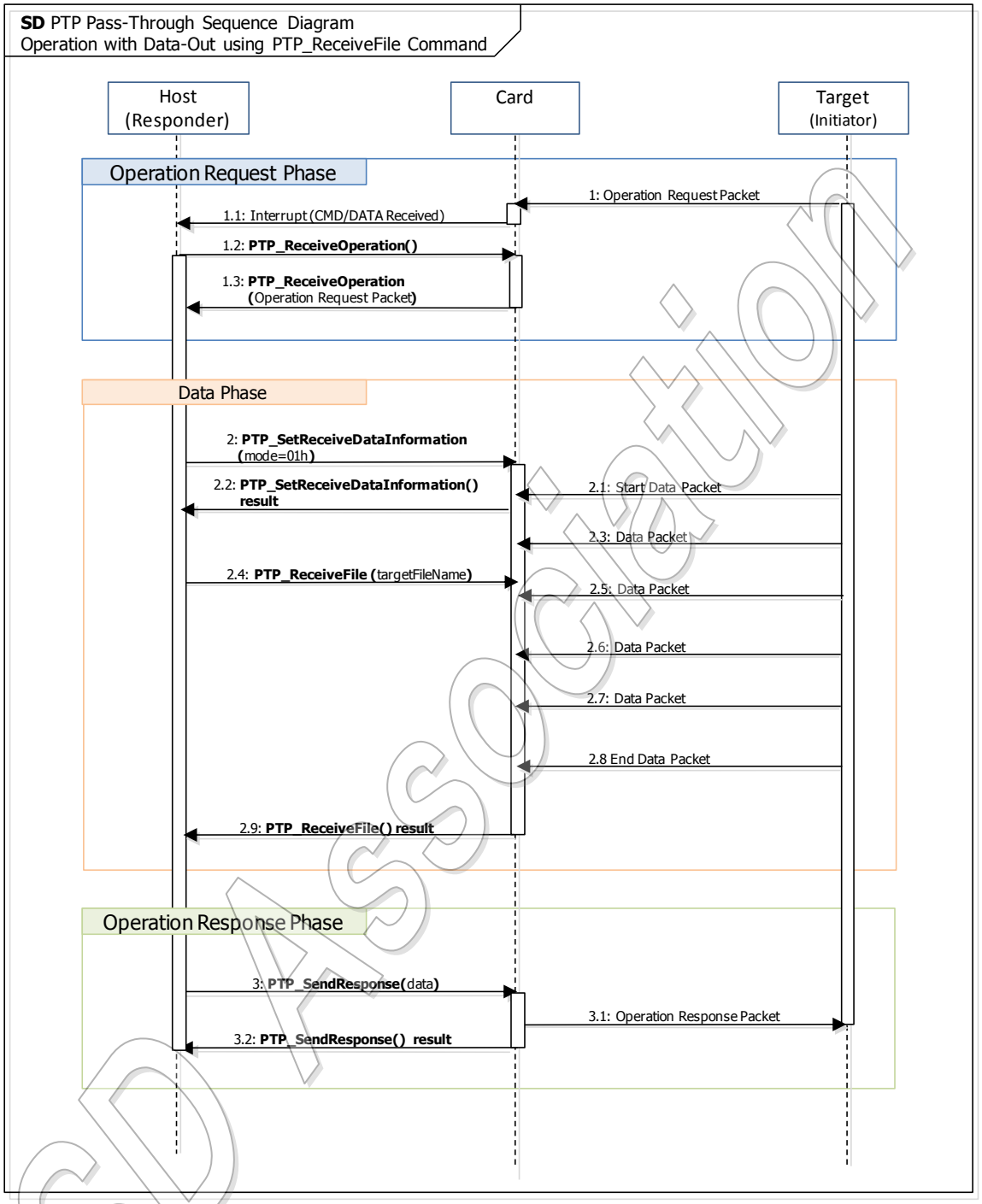


Figure D - 16 : PTP data-out (file) Sequence Diagram

D.6.4.7 Operation with Event, Probe and Cancel

The sequence diagram for the PTP Pass-Through Application from Card to Target device data transfer is described below and in Figure D - 17.

< Steps >

[Event Phase]

1. (Host to Card) The Host application calls the "PTP_SendEvent" command with the argument of data.
   1.1 (Card to Target) The Card sends the "Event Packet" to the target device.
   1.2 (Host to Card) The Host application receives the result information of the "PTP_SendEvent" command.

[Probe Phase Responder to Initiator]

1. (Card to Target) The Card sends the "Probe Request Packet" to the Target device.
   1.1 (Target to Card) The Card receives the "Probe Response Packet" from the Target device.

[Probe Phase Initiator to Responder]

1. (Target to Card) The Target device sends the "Probe Request Packet" to the Card.
   1.1 (Card to Target) The Target device receives the "Probe Response Packet" from the Card.

[Cancel Phase]

1. (Target to Card) The Target device sends the "Cancel Packet" with the "Event connection" to the Card.
   1.1 (Card to Host) The Card changes the RPC (Received PTP Cancel data) register.
   1.2 (Card to Host) The Card updates the PTP Cancelled TransactionID register for cancel operation.
2. (Target to Card) The Target device sends the "Cancel Packet" with the "Command and Data connection" to the Card.

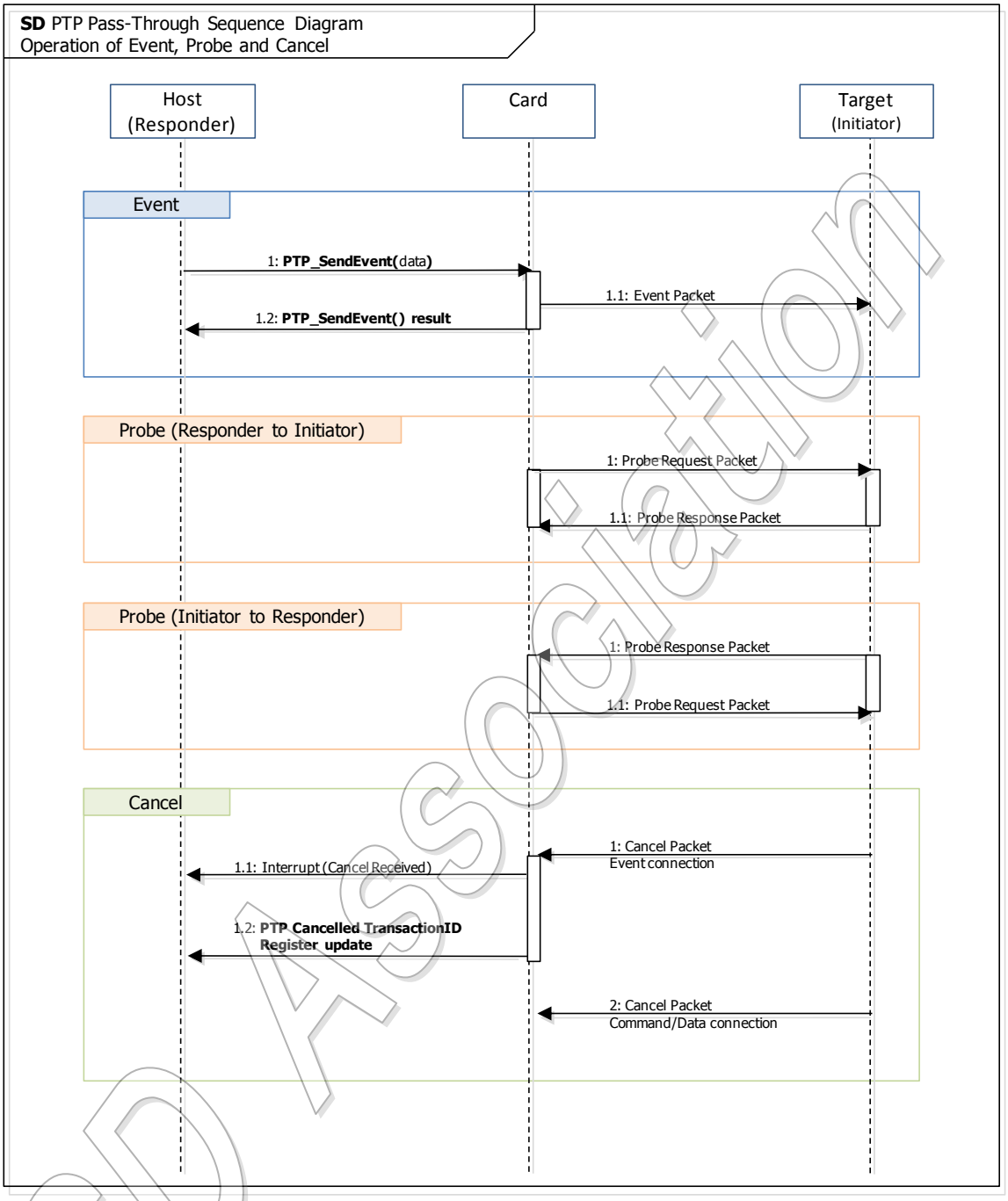


Figure D - 17 : Event, Probe and Cancel Sequence Diagram

D.6.5 DPS mode sequence

The sequence diagram for the DPS printing Application from Card to Printer is described below and in Figure D - 18.

< Steps >

A Card connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by "WiFiDirect" command.
The sequence diagram for the Wi-Fi Direct command is described in **D.5 Wi-Fi Direct Sequence**.

A Card shall carry out the UPnP device discovery process inside all the time while it operates with the DPS mode. The sequence diagram for the DPS discovery from Card to DPS is described below and in **D.6.3.1 DPS mode device discovery**.

1. (Host to Card) The Host application calls the "DPS_ConnectPrinter" command.
   1.1 (Card to Target) The Card creates the PTP-IP connection with the Target device.
   1.2 (Target to Card) The Card receives the result information of PTP-IP connection.
   1.3 (Card to Host) The Card changes the CPI (Connected PTP-IP) register.
   1.4 (Card to Host) The Host application receives the result information of the "DPS_ConnectPrinter" command.

A Host receives the PTP request from the initiator device through the Card. The sequence diagram of DPS mode is described below and in **D.6.4.2 DPS and PTP Pass-Through mode request and response**.

A content data transfers from the Host to the Target initiator device, and the sequence diagram is described below and in **D.6.4.3 PTP data-in** and **D.6.4.4 PTP data-in (file)**.

2. (Host to Card) The Host application calls the "PTP_SwitchMode" command with the Exit argument.
   2.1 (Card to Host) The Host application receives the result information of the "PTP_SwitchMode" command.

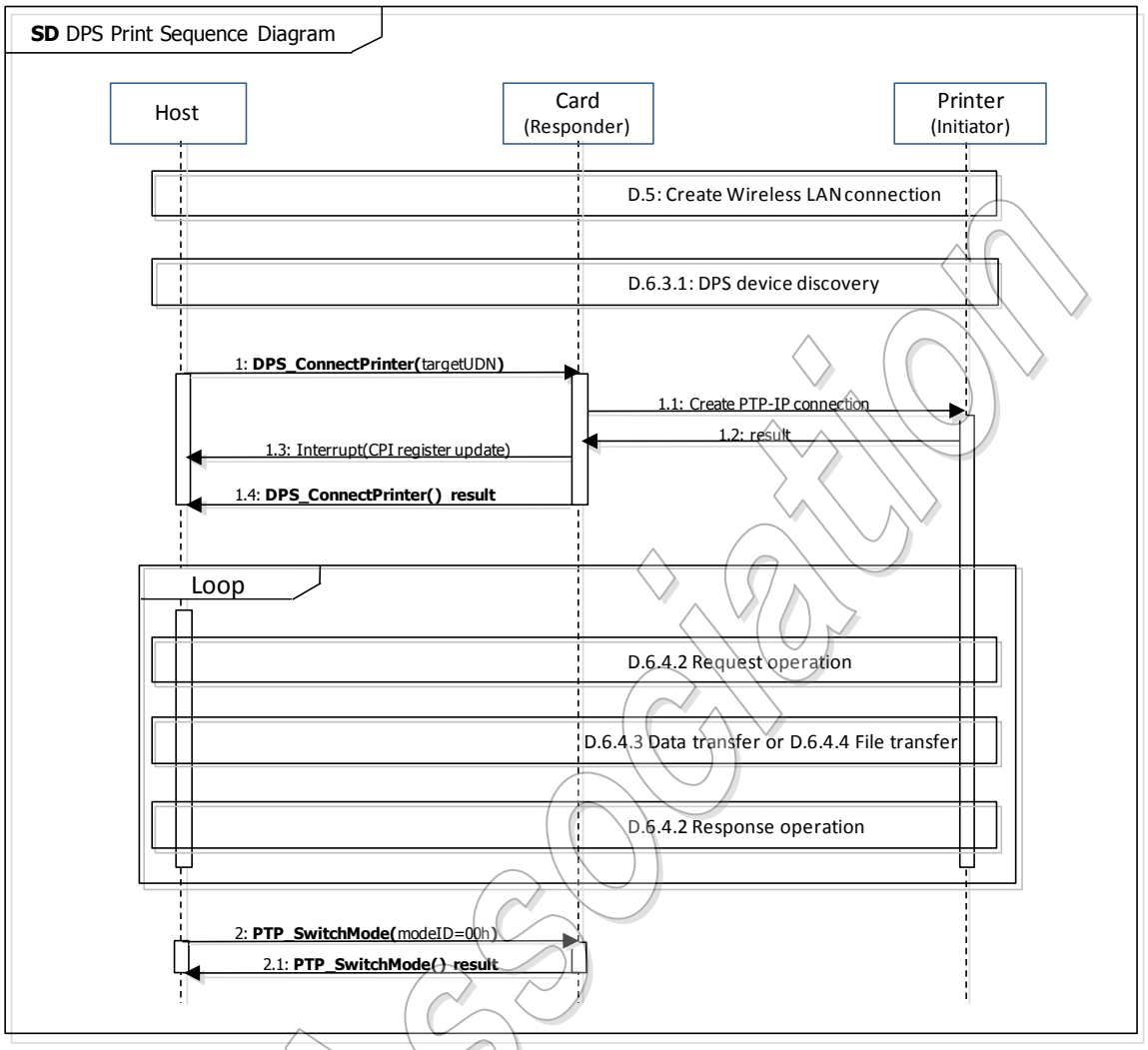


Figure D - 18 : DPS mode Sequence Diagram

D.6.6 PTP Pull Sequence Diagram

The sequence diagram for the PTP Pull Application from Card to PC data transfer is described below and in Figure D - 19.

< Steps >

A Card connects for wireless LAN connection via an access point. Or a Wi-Fi Direct network connection is automatically created by "WiFiDirect" command.
The sequence diagram for the Wi-Fi Direct command is described in **D.5 Wi-Fi Direct Sequence**.

In PTP Pull mode, MTP/IP (described in [MTP/IP]) and Multicast DNS (described in [Multicast DNS]) shall be used as the device discovery process (the advertisement as a PTP-IP device to initiators). The sequence diagram for the PTP Pull mode discovery from the initiator device as the target device to the Card as responder device is described in **D.6.3.2 PTP Pull and PTP Pass-Through mode device discovery**.

A Card receives the PTP request from the initiator device and a Card sends the PTP response to the initiator device. The sequence diagram of PTP Pull mode is described in **D.6.4.1 PTP Pull mode request and response**.

1. (Host to Card) The Host application calls the "PTP_SwitchMode" command with the Exit argument.
   1.1 (Card to Host) The Host application receives the result information of the "PTP_SwitchMode" command.

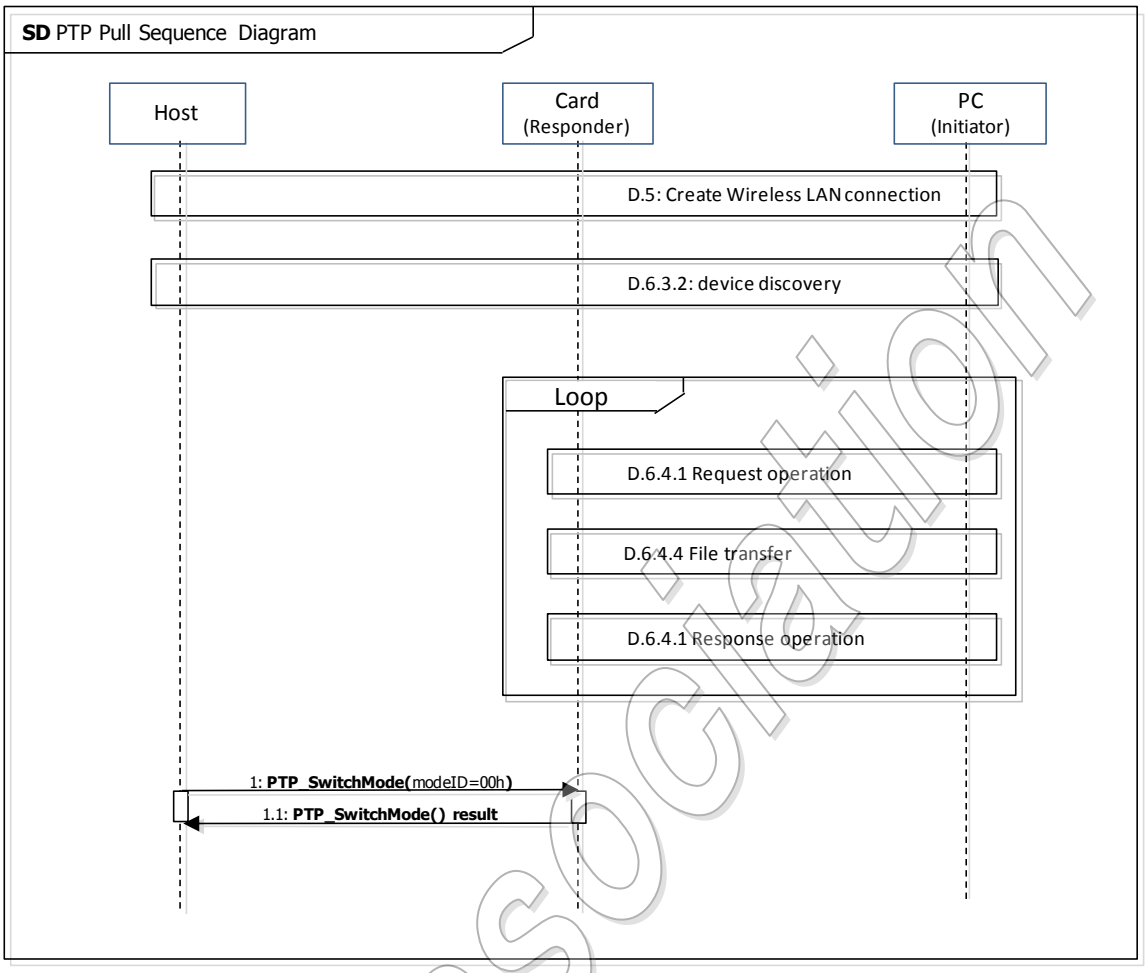


Figure D - 19 : PTP Pull Sequence Diagram