SONY®

NFC

Development Guidelines for NFC/FeliCa-Enabled Devices and Applications

Version 1.4

No. M830-E01-40

Introduction

In this document, NFC Forum Devices (i.e., devices that are manufactured according to NFC Forum specifications) and devices that utilize FeliCa technology are referred to collectively as NFC/FeliCa-enabled devices.

This document is written mainly for developers of NFC/FeliCa-enabled devices and developers of applications for NFC/FeliCa-enabled devices.

The purpose of this document is to provide important information about improving the interoperability between NFC/FeliCa-enabled devices and between those devices and other devices.

For specifications and terminology defined by the NFC Forum, visit the following website:

https://nfc-forum.org/

For information about FeliCa technology, visit the following website:

https://www.sony.net/Products/felica/business/tech-support/index.html

This document does not describe the specifications of individual NFC/FeliCa-enabled devices. For information about individual specifications, contact the suppliers of the devices.

This document covers some aspects of mobile phones having Mobile FeliCa IC chips. It does not, however, describe all the functions of Mobile FeliCa IC chips. If you have any questions when developing applications for Mobile FeliCa, contact FeliCa Networks, Inc. at the following address:

info-fn@FeliCaNetworks.co.jp

This document provides guidelines for developing general Near Field Communication (NFC) systems that support NFC/FeliCa-enabled devices. Use the information in this document as reference to develop the systems that accommodate your operating environment and applications.

The content of this document does not guarantee the correct operation of all current and future products.

The content of this document does not guarantee interoperability between all products.

Contents

1 Overview

This chapter describes the general structure of this document, other reference documents, and notational conventions.

Chapter 2 describes general precautions to be noted.

Chapter 3 describes additional precautions to be noted for Android OS applications, and respective countermeasure examples.

Chapter 4 describes additional precautions to be noted for Windows OS applications, and respective countermeasure examples.

Chapter 5 describes additional precautions to be noted for iOS applications, and respective countermeasure examples.

1.1 Reference documents

Readers of this document are recommended to also read the documents listed in the following table. In this document, the documents that are issued by the NFC Forum and ISO/IEC are referred to as [DIGITAL], [ACTIVITY], [T3T], [REQ], and [ISO/IEC 18092].

Table 1-1: Reference documents

Issuer	Document name
Sony[*1]	Format Distinction Sequence Design Guidelines
	FeliCa Card User’s Manual Excerpted Edition
	FeliCa Technology Code Descriptions
NFC Forum[*2]	NFC Digital Protocol Technical Specification 2.2 [DIGITAL]
	NFC Activity Technical Specification 2.1 [ACTIVITY]
	Type 3 Tag Technical Specification 1.1 [T3T]
	NFC Forum Device Requirements 2.1 [REQ]
ISO/IEC	ISO/IEC 18092: Information technology - Telecommunications and information exchange between systems - Near Field Communication - Interface and Protocol-1 (NFCIP-1) [ISO/IEC 18092]

[*1] https://www.sony.net/Products/felica/business/tech-support/index.html

[*2] https://nfc-forum.org/

1.2 Notational conventions

This section describes the notation used in this document.
The following notational conventions apply in this document, unless otherwise specified:

Binary values	"b" is appended to a binary value (e.g., 0101b).
Hexadecimal values	"h" is appended to a hexadecimal value (e.g., FFFFh).
Decimal values	Nothing is appended to a decimal value.

In a bit string, the most significant bit is at the leftmost end and the least significant bit is at the rightmost end.
The Byte order is Big Endian, unless otherwise specified.
In figures, Byte strings and bit strings are denoted as shown in Figure 1-1, Figure 1-2, and Figure 1-3.

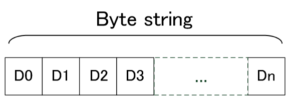

Figure 1-1 : Graphic notation of Byte string

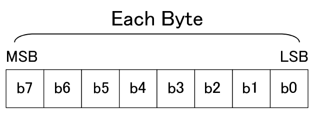

Figure 1-2 : Graphic notation of bit string

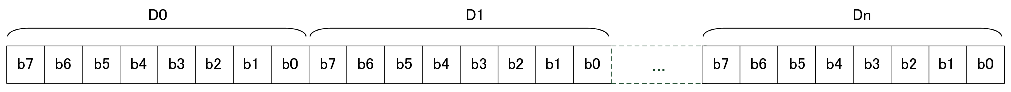

Figure 1-3: Graphic notation of Byte string and bit string

When referring to specific Bytes or bit in figures, the following notation is used:

upper 2 Bytes	Indicates 2 Bytes, from D0 to D1, inclusive. Unless otherwise specified, D0 is the most significant Byte.
D0-D15	Indicates 16 Bytes from D0 to D15, inclusive.
upper 6 bits	Indicates 6 bits from b7 to b2, inclusive.
b5-b3	Indicates 3 bits from b5 to b3, inclusive.

2 Precautions

This chapter describes the precautions to be noted when developing NFC/FeliCa-enabled devices, and IC chips, drivers, operating systems, middleware, applications, servers, etc., running in NFC/FeliCa-enabled devices.

The following lists different types of NFC/FeliCa-enabled devices:

- NFC Forum Devices, which are based on the specifications and requirements defined by the NFC Forum.
- Devices that are developed according to the specifications defined by the NFC Forum (including those that do not fully satisfy the requirements for NFC Forum Devices).
- Reader/Writer utilizing FeliCa technology.
- Contactless IC cards utilizing FeliCa technology, including devices with functions that are equivalent to those of contactless IC cards, such as Mobile FeliCa products.

The NFC Forum defines the specifications and requirements for NFC Forum Devices. Note that some NFC/FeliCa-enabled devices do not conform to these specifications and requirements. There are also NFC/FeliCa-enabled devices that do not need to fully satisfy the requirements for NFC Forum Devices because of their usage conditions (devices that are used exclusively as Reader/Writer devices, for example). Note also that such devices (especially Reader/Writer) could have been developed before the NFC Forum defined the specifications and requirements. Such Reader/Writer do not take any specific requirements to interact with NFC Forum Devices into account.

This chapter describes the points to note for developing devices so that the interoperability can be improved between NFC Forum Devices and between an NFC Forum Device and a non-NFC Forum Device.

NFC Forum Devices have the following two modes:

- Poll mode: the mode in which an NFC Forum Device sends a command and receives a corresponding response.
- Listen mode: the mode in which an NFC Forum Device receives a command and sends a corresponding response.

In this document, the Reader/Writer and an NFC Forum Device in Poll mode are called Pollers. A contactless card and an NFC Forum Device in Listen mode are called Listeners.

When developing a Reader/Writer, see section 2.1.1 "Precautions for developing Pollers".

When developing an NFC Forum Device, see section 2.1.1 "Precautions for developing Pollers" and section 2.1.2 "Precautions for developing Listeners".

2.1 Mode for responding to SENSF_REQ (Polling)

A Poller sends the SENSF_REQ command (known as the Polling command, in FeliCa technology) to capture a Listener compliant with NFC-F. The SENSF_REQ command contains parameters such as System Code and Request Code. For details of the SENSF_REQ command, see [DIGITAL].

When a Poller attempts to capture a Listener compliant with NFC-F, the Poller might send the SENSF_REQ command with System Code set to FFFFh. When an NFC Forum Device in Listen Mode or a contactless IC card receives the SENSF_REQ command with System Code set to FFFFh, the upper two Bytes of NFCID2 (known as Manufacture ID (IDm) in FeliCa technology) in the SENSF_RES response (Polling response) it returns are one of the pairs listed in Table 2-1.

Table 2-1: Upper two Bytes of NFCID2 (IDm)

Type of device		Upper 2 Bytes of NFCID2
NFC Forum Device	When the device is compliant with NFC-F, in Listen Mode, and supports the NFC-DEP Protocol	01h, FEh
	When the device is compliant with NFC-F, in Listen Mode, and supports the Card Emulation mode based on Type 3 Tag Platform.	Other than 01h, FEh
Contactless IC card		Other than 01h, FEh

NFC Forum Devices might support both the NFC-DEP Protocol and the Type 3 Tag Platform simultaneously. In this case, requirement 6.2.1.13 in [ACTIVITY] stipulates that an NFC Forum Device returns a response as follows according to the parameters in the SENSF_REQ command:

- The Device responds as the NFC-DEP Protocol when System Code of the command is FFFFh and Request Code is 00h (the upper two Bytes of NFCID2 are 01h and FEh).
- The Device responds as the Type 3 Tag Platform when System Code of the command matches System Code of the Type 3 Tag Platform.

For details of the requirements for NFC Forum Devices in regard to the SENSF_REQ command, see [ACTIVITY], [DIGITAL], and [REQ].

2.1.1 Precautions for developing Pollers

We recommend that Pollers compliant with NFC-F perform polling according to the "Format Distinction Sequence Design Guidelines".

A Poller should send the SENSF_REQ command with System Code set to FFFFh and Request Code set to 00h and then send additional SENSF_REQ commands with System Code set to a value other than FFFFh as necessary. The Poller can receive a response from a contactless IC card or NFC Forum Device as the Type 3 Tag Platform by sending a SENSF_REQ command with System Code set to a value other than FFFFh. Note here that the connected NFC Forum Device might also support the Type 3 Tag Platform even if a response (the upper two Bytes of NFCID2 are 01h and FEh) indicating that the NFC-DEP Protocol is supported is received in response to the SENSF_REQ command.

If the Poller only intends to communicate with contactless IC cards and the Poller knows the exact value for System Code, the Poller does not need to send the SENSF_REQ command with System Code set to FFFFh in the first place.

If you want to communicate with a contactless IC card or an NFC Forum Device in Card Emulation Mode based on the Type 3 Tag Platform and you cannot identify System Code, it is recommended to send the SENSF_REQ command with System Code set to FFFFh and Request Code set to 01h.

Table 2-2 shows the recommended values for the parameters to be specified in the first SENSF_REQ command to be sent.

Table 2-2: Recommended values for the parameters to be specified in the first SENSF_REQ command to be sent

Target of communication	System Code	Request Code
Other than below	FFFFh	00h
Cards only (when the exact value for System Code is known)	Other than FFFFh	Any
Cards only (when the exact value for System Code is not known)	FFFFh	01h

When middleware, a driver, or an operating system provides applications with the function for sending the SENSF_REQ command, make sure that the applications can specify System Code and Request Code.

2.1.2 Precautions for developing Listeners

If an NFC Forum Device in Listen Mode only supports either the NFC-DEP Protocol or the Type 3 Tag Platform, process the received SENSF_REQ command according to the NFC Forum Device requirements.

2.2 NFCID2 (IDm) changes dynamically

The SENSF_RES response, a response to the SENSF_REQ command, contains NFCID2 (IDm). The value of NFCID2 may or may not be unique for each card. For details of IDm, see the "FeliCa Technology Code Descriptions".

When an NFC Forum Device in Listen Mode returns the SENSF_RES response, NFCID2 in the response might dynamically change after the operating field disappears. NFCID2 might dynamically change in case of not only NFC Forum Device but also other devices or cards.For example, if the upper two Bytes of NFCID2 are one of the pairs of Bytes listed in Table 2-3, NFCID2 might dynamically change. The shown pairs of Bytes, however, are not the only cases of NFCID2 changes. Even if the upper two Bytes of NFCID2 are one of the pairs in Table 2-3, NFCID2 might not change dynamically.

Table 2-3: Cases when NFCID2 might dynamically change

Upper 2 Bytes of NFCID2	Description
01h, FEh	The succeeding 6 Bytes are random numbers. Specified in [ISO/IEC 18092] and [DIGITAL].
02h, FEh	The succeeding 6 Bytes can be any. Specified in [DIGITAL].
03h, FEh	The succeeding 6 Bytes conform to the number system containing Data Format Code. The lower 4 Bytes might dynamically change in some cases such as NFC Dynamic Tag.

Figure 2-1 shows an example of how NFCID2 changes dynamically.

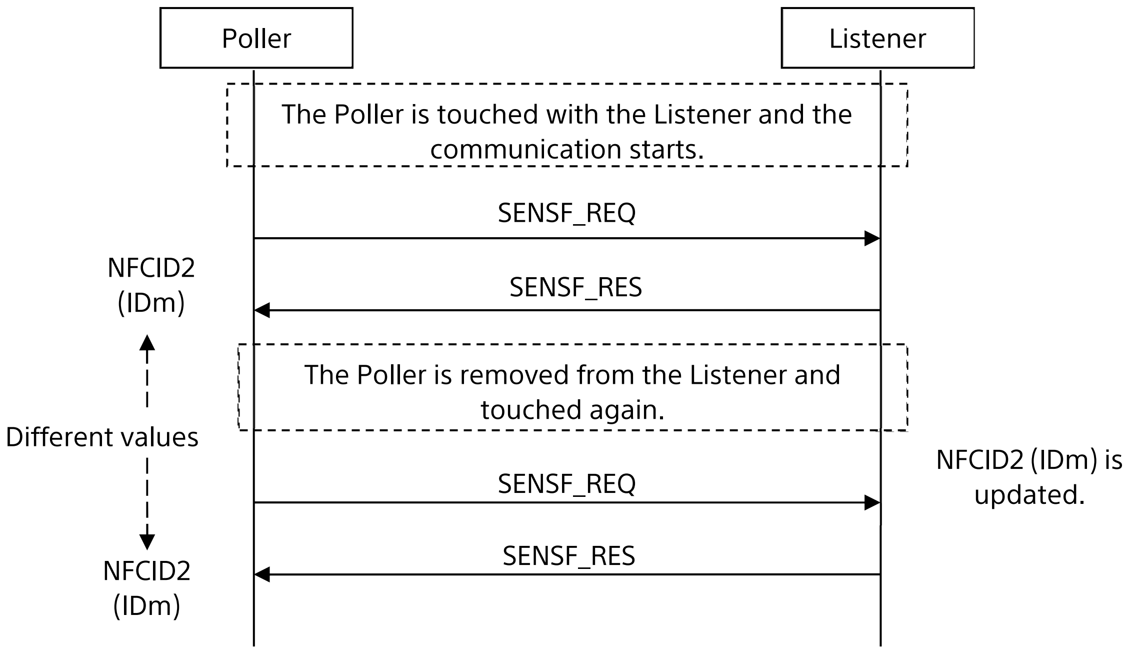
Poller
Listener
The Poller is touched with the Listener and the communication starts.
SENSF_REQ
NFCID2 (IDm)
SENSF_RES
The Poller is removed from the Listener and touched again.
Different values
SENSF_REQ
NFCID2 (IDm) is updated.
SENSF_RES
NFCID2 (IDm)

Figure 2-1: Example of how NFCID2 (IDm) changes dynamically

2.2.1 Precautions for developing Pollers

When you develop the applications, middleware, or drivers for Pollers, we recommend that you take into account that NFCID2 might dynamically change.

2.3 Communication fails when both sides become Pollers

If two NFC/FeliCa-enabled devices simultaneously enter Poll Mode, they both generate a carrier, disabling communications. NFC Forum Devices that support Poll Mode use the RF Collision Avoidance (RFCA) function to prevent each other from generating carriers. For details of RFCA, see [ACTIVITY].

When a Reader/Writer without RFCA comes in close contact with an NFC Forum Device (that supports both Poll Mode and Listen Mode) with RFCA, both devices might become Pollers depending on which device generates a carrier first.

- When the Reader/Writer generates a carrier first (becomes a Poller) and then the NFC Forum Device attempts to generate a carrier, the RFCA of the NFC Forum Device prevents it from becoming a Poller.
- When the NFC Forum Device generates a carrier first (becomes a Poller) and then the Reader/Writer attempts to generate a carrier, the Reader/Writer also becomes a Poller because it does not support RFCA.

The second case might occur when, for example, the Reader/Writer temporarily suspends the generation of a carrier during the processing and the counterpart NFC Forum Device starts to generate a carrier because it does not detect the carrier from the Reader/Writer. Figure 2-2 shows an example sequence.

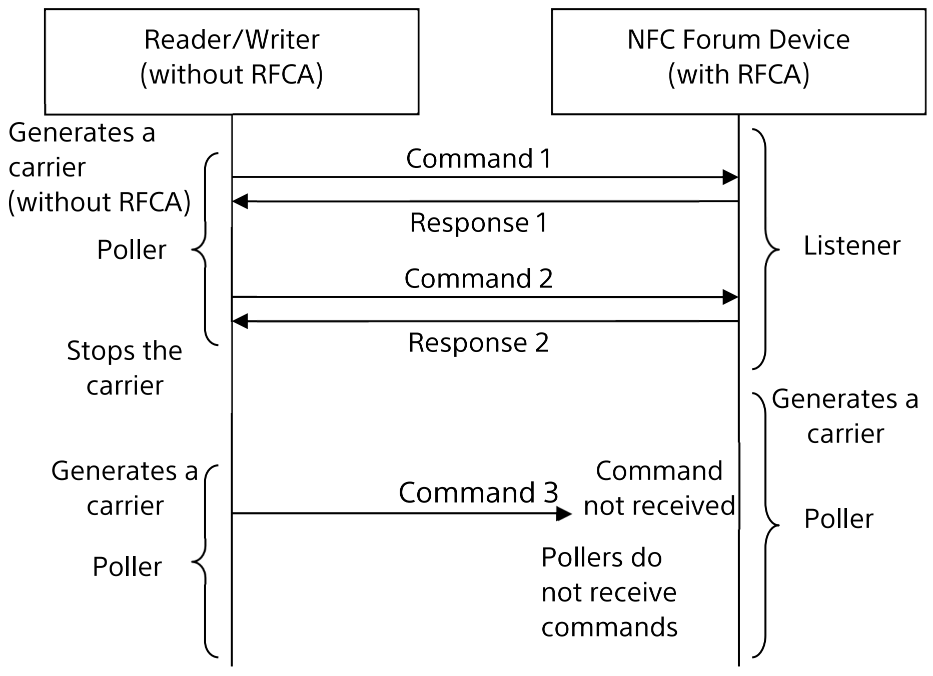

Figure 2-2: Example sequence when two devices become Pollers

2.3.1 Precautions for developing Pollers

Possible solutions are as follows for preventing both sides from becoming Pollers causing communication failures.

- Both sides support RFCA.
- Do not suspend the generation of a carrier during processing.
- If a device needs to suspend the generation of a carrier during processing, the device should keep resending commands (SENSF_REQ command for NFC-F) after it resumes the generation of the carrier until the counterpart NFC Forum Device becomes the Listener again. Note that the length of time required for an NFC Forum Device to become a Listener again depends on the implementation.

2.4 Shutdown processing of NFC Forum Devices

When an NFC Forum Device operates as the Card Emulation mode based on Type 3 Tag Platform , if the carrier from the counterpart Reader/Writer disappears, the IC chip that enables the NFC function might shut down. In this case, during the shutdown procedure, the NFC Forum Device might not operate as the Listener. Whether the IC chip performs shutdown and the length of the shutdown procedure depend on the implementation of the IC chip.

For example, if a Reader/Writer temporarily suspends the generation of a carrier during processing, the IC chip in the counterpart NFC Forum Device might start to shut down. At this time, even if the Reader/Writer resumes generation of a carrier and sends a command, the IC chip in the counterpart NFC Forum Device might not respond to the command when it is performing shutdown processing or being started (see Figure 2-3).

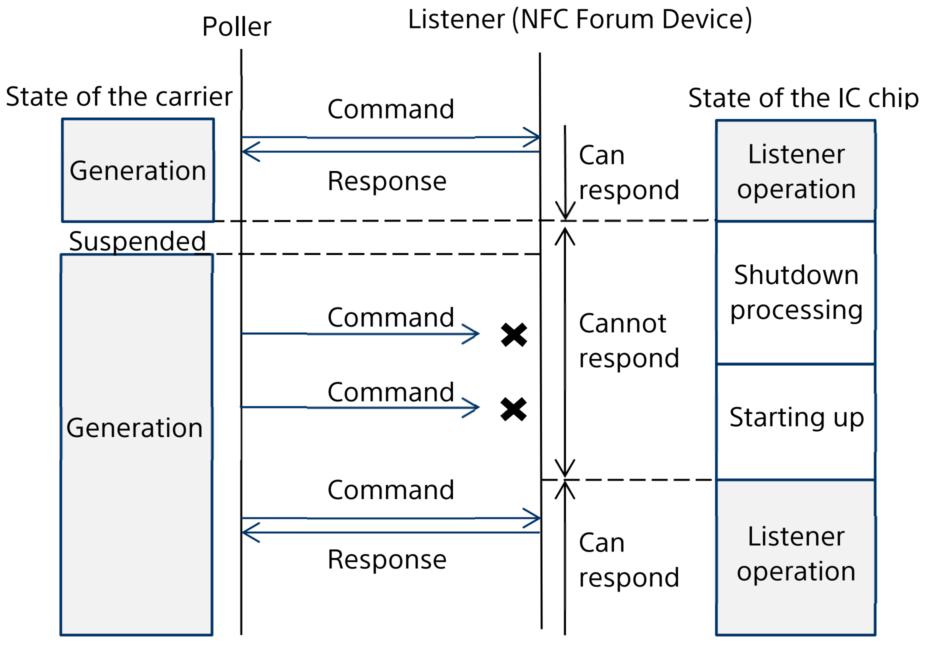

Figure 2-3: Case where the IC chip does not respond due to shutdown processing

2.4.1 Precautions for developing Pollers

We recommend that you should not suspend the generation of a carrier during processing.

If a device needs to suspend the generation of a carrier during processing, we recommend that the device keep resending commands after it resumes the generation of the carrier until the counterpart NFC Forum Device becomes able to respond to the commands.

The time required for an NFC Forum Device as a Listener to perform shutdown processing (upon detecting suspension of carrier generation) depends on the implementation. For example:

Exemplary time for the IC chip to perform shutdown processing (from start to end) = 30.0 ms

According to requirement 8.7.4.1 in [DIGITAL], the NFC Forum Device as a Listener is able to receive a command within 20.0 ms after receiving a carrier (Listener start-up time). Therefore, if the NFC Forum Device as a Listener detects suspension of carrier generation and starts shutdown processing, the exemplary time after it detects a carrier again and is ready to receive a command is 50.0 ms.

According to requirement 7.1.1.2 in [ACTIVITY], the NFC Forum Device as a Poller must send a command 20.4 ms after generating a carrier. This time is the Listener start-up time plus a margin.

Therefore, if the Poller suspends generation of a carrier temporarily, and then resumes generation of a carrier to continue processing, it is recommended to continue to send the command for at least 50.4 ms after suspension of carrier generation (or for at least 20.4 ms after resuming carrier generation if the carrier generation is suspended for more than 30.0 ms) and to send the command at least once after that time has elapsed.

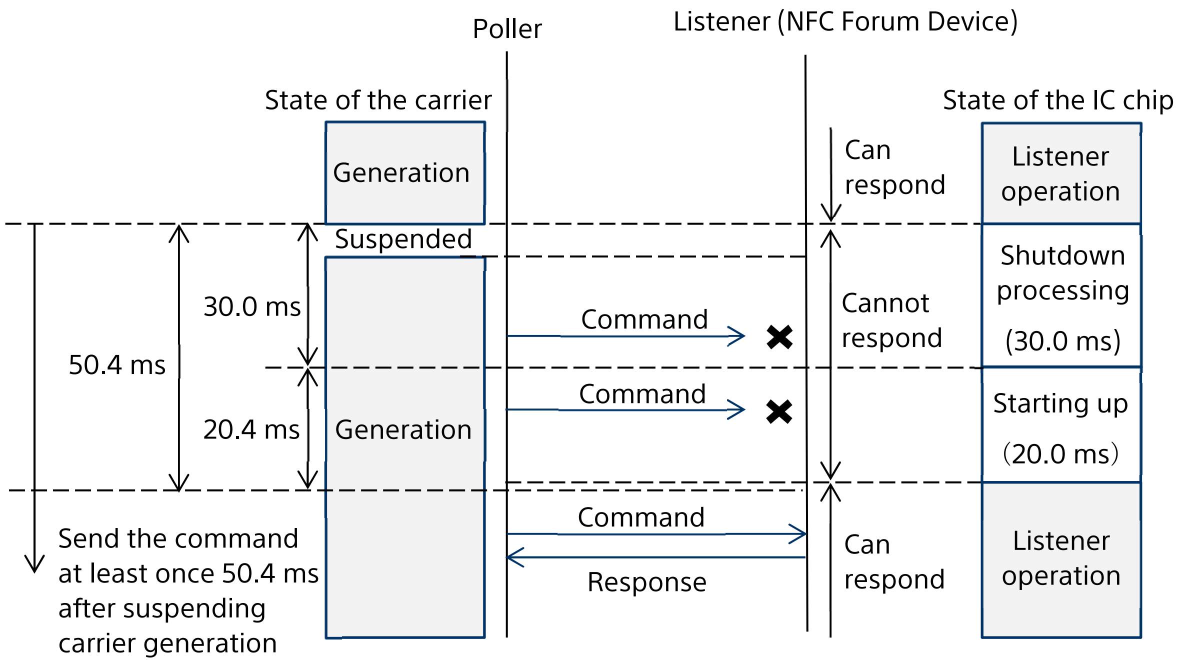

Figure 2-4: Command retransmission example (when carrier generation is suspended for no more than 30.0 ms)

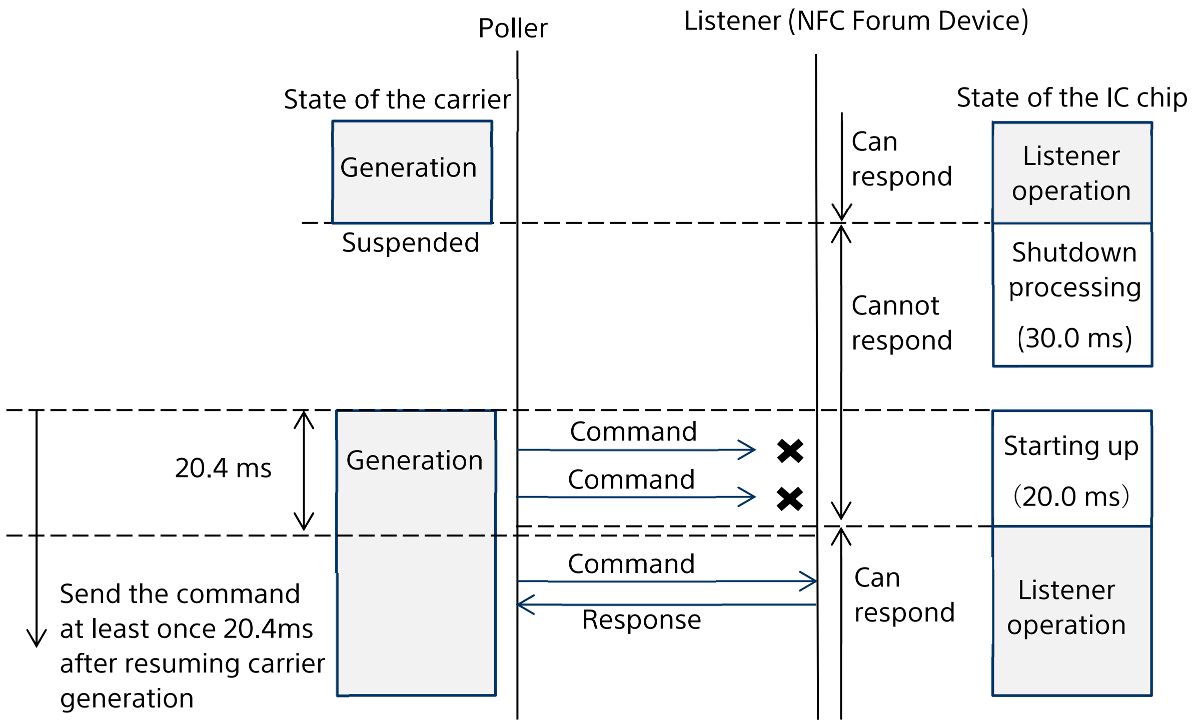

Figure 2-5: Command retransmission example (when the carrier generation is suspended for 30.0 ms or longer)

2.5 Mutual authentication is canceled when the presence of a card is checked

The middleware or a driver of a Poller might send the SENSF_REQ command to a card independent of the command transmission and reception instructions of an application so that the middleware or the driver can determine whether the counterpart card (Listener) still exists in the operating field.

When mutual authentication is performed for a contactless IC card based on FeliCa technology (by executing the Authentication1 command or the Authentication1 v2 command), Mode of the card changes to the one other than Mode0. If Mode is not Mode0, the card does not respond to the SENSF_REQ command.

A contactless IC card based on FeliCa technology sometimes has multiple logical card units named System, each having an own System Code and NFCID2. System created first, at the time of card manufacture, is known as "System 0". From System 0, "System 1" can be created, then "System 2", and so on in ascending order. After mutual authentication is performed for System other than System 0, if the card receives the SENSF_REQ command with System Code set to FFFFh (wildcard), System of the card changes to System 0 (initial System) and the card returns a response to SENSF_REQ. At this time, Mode of the card changes to Mode0 and mutual authentication is canceled.

For details of card Mode and Switching between Systems, see the "FeliCa Card User’s Manual Excerpted Edition".

Contactless IC cards based on FeliCa technology and NFC Forum Devices supporting Listen Mode of NFC-F accept the SENSF_REQ command, so the SENSF_REQ command with System Code set to FFFFh is sometimes used to check whether a Listener compliant with NFC-F exists (presence check). If, however, the middleware or a driver of a Poller sends the SENSF_REQ command with System Code set to FFFFh to check for the presence of a card while an application is performing processing that needs mutual authentication with a contactless IC card, the mutual authentication with the card is canceled as follows and the processing of the application might fail:

- When mutual authentication is performed with System 0, Switching between Systems does not occur and no SENSF_RES response is returned. As a result, the middleware or the driver of the Poller might determine that the card is absent, cancel the generation of a carrier, and report an error to the application. Once the generation of a carrier is canceled, Mode of the card returns to Mode0 and mutual authentication is canceled.
- When mutual authentication is performed with System other than System 0, Switching between Systems occurs. For this reason, Mode of the card returns to Mode0 and mutual authentication is canceled.

In either case, the application cannot continue its processing. Figure 2-6 and Figure 2-7 show the example sequences for these cases.

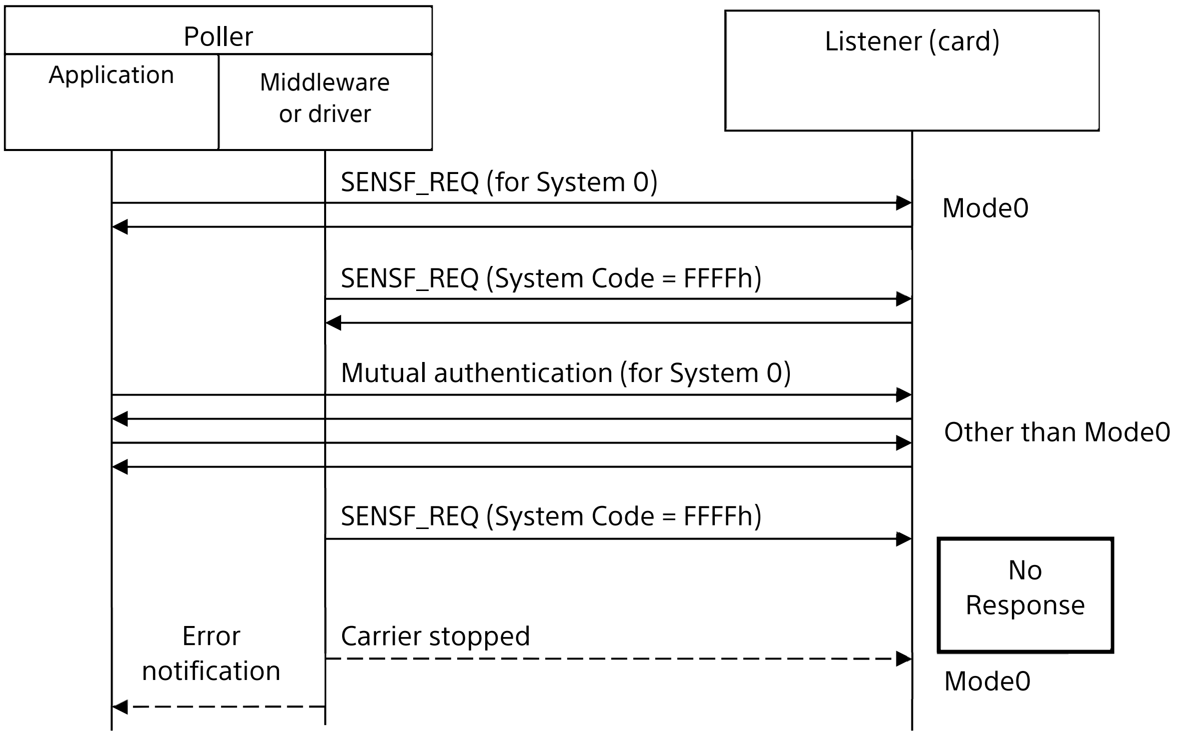

Figure 2-6: Example sequence when the Poller mistakenly determines that the card is absent

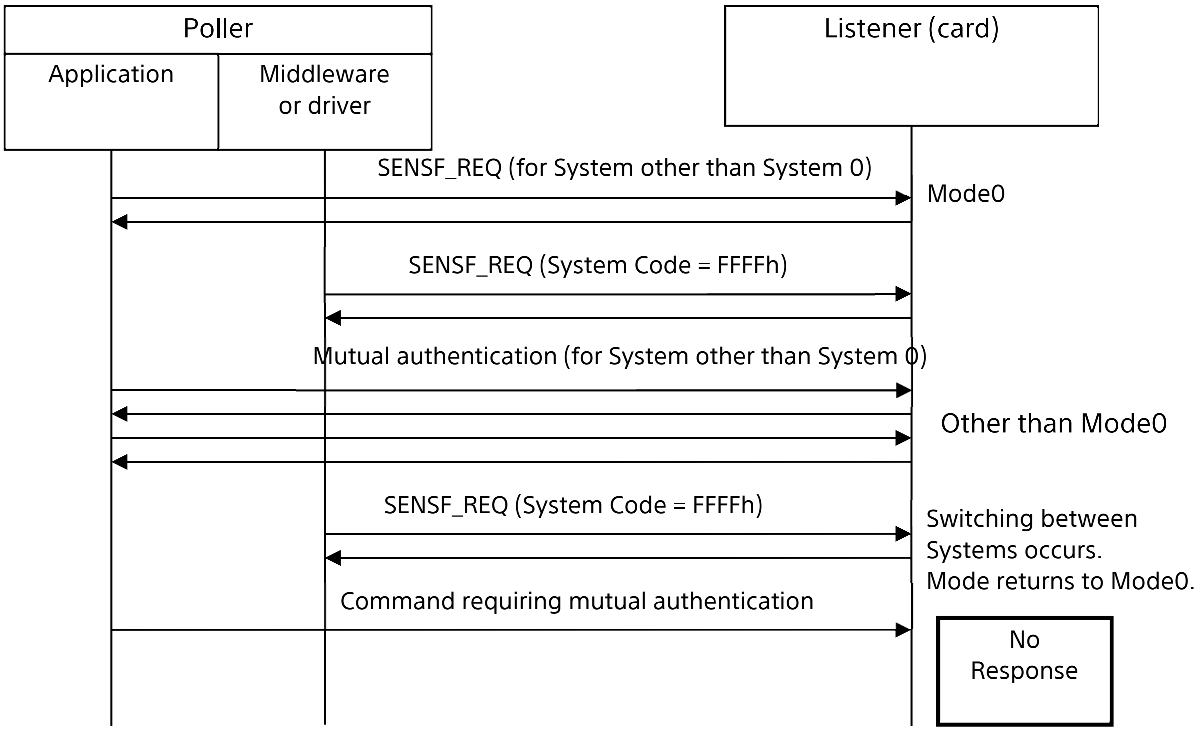
Poller
Application
Middleware or driver
Listener (card)
SENSF_REQ (for System other than System 0)
Mode0
SENSF_REQ (System Code = FFFFh)
Mutual authentication (for System other than System 0)
Other than Mode0
SENSF_REQ (System Code = FFFFh)
Switching between Systems occurs. Mode returns to Mode0.
Command requiring mutual authentication
No Response

Figure 2-7: Example sequence when mutual authentication is canceled due to Switching between Systems

2.5.1 Precautions for developing Pollers

We recommend that middleware and drivers should not check for the presence of counterpart cards (Listeners) independent of the processing of applications.

If there are requirements to check for the presence of cards, provide the applications with the ability to suppress that check by middleware or drivers.

- For example, provide the application with the APIs to suppress the checking for the presence of cards by middleware or drivers. The application uses the APIs to suppress that check while they perform necessary processing for mutual authentication. If an operating system needs to check for the presence of cards as part of its specifications, specify the APIs for suppressing that check as part of the specifications of the operating system.

While an application suppresses the checking of presence of cards, middleware or any driver cannot detect whether a card disappears or is replaced by another card. After the application cancels the suppression of the checking for the presence of cards, the middleware or the driver performs checking and then detects a missing card or a replaced card.

2.6 Timeout time for FeliCa commands

As Type 3 Tag commands, the POLLING command (SENSF_REQ command, known as the Polling command in FeliCa technology), the CHECK command (Read Without Encryption command), and the UPDATE command (Write Without Encryption command) are defined in [T3T].
The maximum response time for the CHECK command and the UPDATE command can be calculated from the SENSF_RES response, which is the response to the SENSF_REQ command. The maximum value of the maximum response time is approximately 2.475 s when $\mathrm{MRTI}_{\mathrm{CHECK}}$ (15th Byte of SENSF_RES) is FFh and Number of Block of the CHECK command is 15.

Some contactless IC cards based on FeliCa technology support commands other than those mentioned in the previous paragraph. The maximum response time for those other commands can be calculated by checking the corresponding Byte in SENSF_RES for each command. For example, when the 11th Byte of SENSF_RES is FFh and Number of Node of the Request Service command is 32, the maximum response time is approximately 5.104 s. This is the maximum value of the maximum response time calculated from SENSF_RES. For details of the formula for calculating the maximum response time for contactless IC cards based on FeliCa technology, see the "FeliCa Card User’s Manual Excerpted Edition".

2.6.1 Precautions for developing Pollers

In the IC chips, drivers, and middleware of Pollers, make sure that applications can specify at least up to 5.104 s as the timeout time for the commands of the Type 3 Tag Platform.

2.7 Accessing multiple System

Figure 2-8 shows an example internal configuration of an NFC Forum Device. The NFC Controller is responsible for transmitting data over the NFC interface. The Device Host is the Execution Environment that is responsible for the total management of the NFC Forum Device and peripherals. The Secure Element provides security.

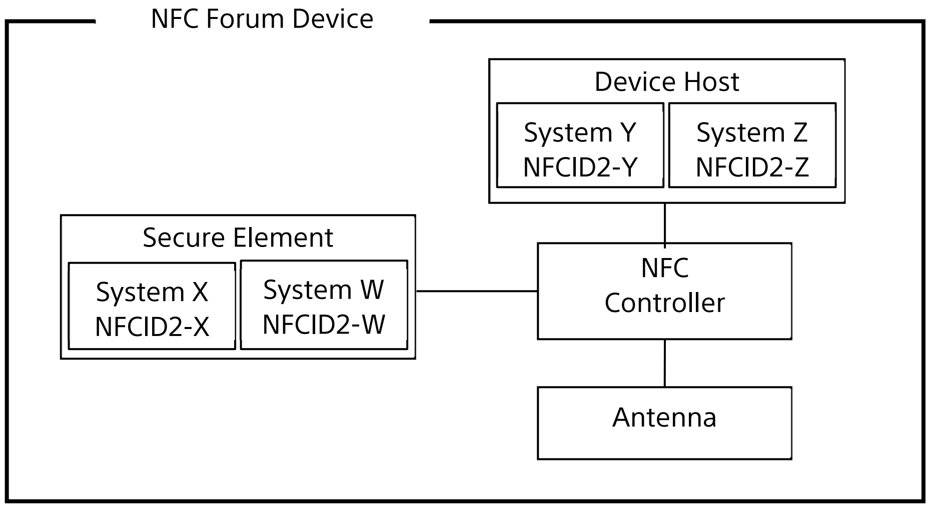

Figure 2-8: Example internal configuration of an NFC Forum Device

If an NFC Forum Device supports the Type 3 Tag Platform, its card emulation function is implemented by the Secure Element or the Device Host. In some cases, both the Secure Element and the Device Host provide the card emulation function. Each component may also have multiple Systems (logical card units). If an NFC Forum Device contains multiple components that provide the card emulation function, how the device internally routes the commands of the Type 3 Tag Platform depends on the implementation.

Some implementations might route commands other than the SENSF_REQ command only to the last component that responded to the SENSF_REQ command. If a Poller sends commands in the following sequence to such an NFC Forum Device, the result might not be the one expected by the Poller:

1) The Poller sends the SENSF_REQ command specified with System Code X for the card emulation function to be provided by the Secure Element. The Poller obtains NFCID2-X as NFCID2 for that System
2) The Poller sends the CHECK command specified with NFCID2-X. The command is transferred from the NFC Controller to the Secure Element. The Poller receives a response.
3) The Poller sends the SENSF_REQ command specified with System Code Y for System to be provided by the Device Host. The Poller obtains NFCID2-Y as NFCID2 for that System.
4) The Poller sends the CHECK command specified with NFCID2-Y. The command is transferred from the NFC Controller to the Device Host. The Poller receives a response.
5) The Poller sends the UPDATE command specified with NFCID2-X. The command is transferred from the NFC Controller to the Device Host. NFCID2-X differs from the value of NFCID2 expected by the Device Host. Therefore, the NFC Forum Device returns No Response.

2.7.1 Precautions for developing Pollers

Pollers must send only the commands for last System that responds to the SENSF_REQ command. If a Poller needs to send commands for any other System, the Poller must first send the SENSF_REQ command with System Code for that System and receive a response even if the Poller knows NFCID2 for that System.

For instance, for the previous example, we recommend performing the following procedure:

1) The Poller sends the SENSF_REQ command specified with System Code X for the card emulation function to be provided by the Secure Element.
   The Poller obtains NFCID2-X as NFCID2 for that System.
2) The Poller sends the CHECK command specified with NFCID2-X. The command is transferred from the NFC Controller to the Secure Element.
   The Poller receives a response.
3) The Poller sends the SENSF_REQ command specified with System Code Y for System to be provided by the Device Host.
   The Poller obtains NFCID2-Y as NFCID2 for that System.
4) The Poller sends the CHECK command specified with NFCID2-Y. The command is transferred from the NFC Controller to the Device Host.
   The Poller receives a response.
5) The Poller sends the SENSF_REQ command specified with System Code X for the card emulation function to be provided by the Secure Element.
   The Poller obtains NFCID2-X as NFCID2 for that System.
6) The Poller sends the UPDATE command specified with NFCID2-X. The command is transferred from the NFC Controller to the Secure Element.
   The Poller receives a response.

NOTE For NFC Forum Device, the NFCID2 value of a System can be completely independent of any of the values in the other Systems., even if there are several System between Secure Element and Device Host, and within the same Secure Element or Device Host.

When a Poller accesses more than one System, it is recommended that it not be implemented to branch using relationship of NFCID2 in the NFC Forum device.

2.8 Using multiple technologies for access

[ACTIVITY] defines the Listen Mode state machine for NFC Forum Devices. This state machine does not allow NFC Forum Devices to respond to the commands of other technologies after having responded to the commands of one technology (NFC-A, NFC-B, NFC-F, or NFC-V).

Figure 2-9 shows corresponding example.

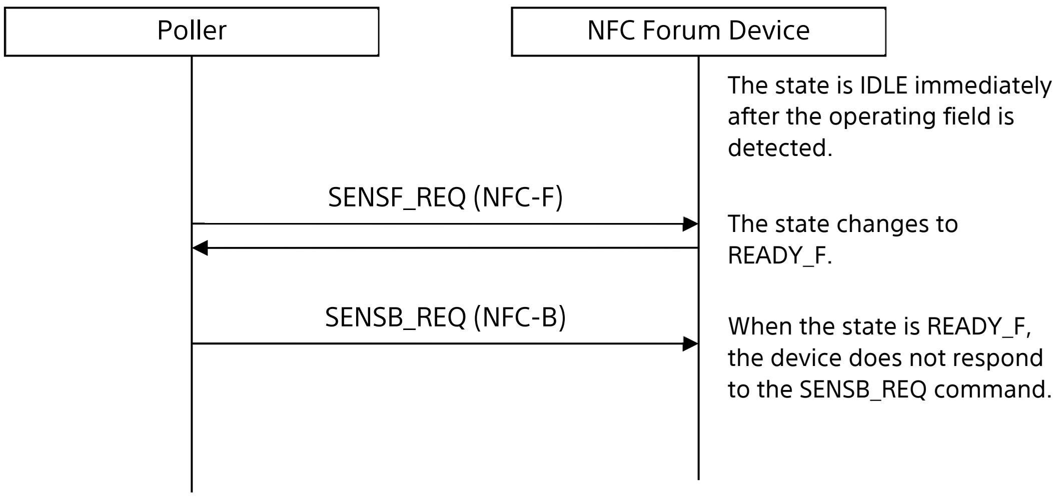
Poller
NFC Forum Device
The state is IDLE immediately after the operating field is detected.
SENSF_REQ (NFC-F)
The state changes to READY_F.
SENSB_REQ (NFC-B)
When the state is READY_F, the device does not respond to the SENSB_REQ command.

Figure 2-9: Example of NFC Forum Device not responding to command of another technology

When the operating field disappears, the Listen Mode state machine is terminated. Later, when the NFC Forum Device re-enters Listen Mode, the new state machine allows the NFC Forum Device again to respond to commands from any technology.

2.8.1 Precautions for developing Pollers

When a Poller uses one technology to capture a Listener, we recommend that the Poller completes the processing for that technology and stops the carrier without using the commands of other technologies. If the target technology is predetermined, we recommend that Pollers perform polling for that technology alone. If middleware, a driver, or an operating system provides applications with the function for performing polling and capturing Listeners, make sure that the applications can specify the technology to be used for polling.

Figure 2-10 shows an example sequence for a Poller when it uses multiple technologies to access NFC Forum Devices in Listen Mode.

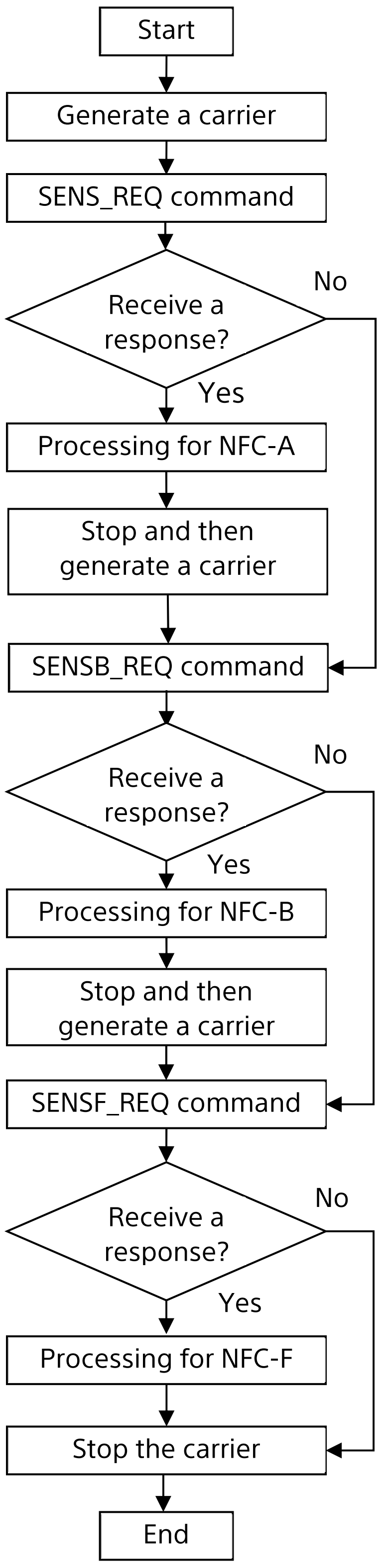

Figure 2-10: Example of Poller using multiple technologies to access NFC Forum Devices

3 Precautions in the Android OS

This chapter describes additional precautions for applications on the Android OS, and respective countermeasure examples.

The program code example described in this chapter is for reference only and does not guarantee correct operation or mutual connectivity.

The description in this chapter is based on the API specifications (API level 30) for the Android OS as of January 2021. The latest API specifications are shown in the following website provided by Google:

https://developer.android.com/reference

In this chapter, a device running the Android OS is called an Android device. The NFC function of the Android OS is created by referring to the specifications defined in the NFC Forum, so an Android device having the NFC function can be regarded as an NFC/FeliCa-enabled device.

3.1 Operation of the NFC function in the Android OS

This chapter describes the general specifications and operation of the NFC function in the Android OS.

3.1.1 Whether the NFC function is installed

Not all Android devices have the NFC function. Android devices that have the NFC function support the Reader/Writer function based on NFC-F.

3.1.2 Turning the NFC function on or off

In an Android device having the NFC function, the user can turn the NFC function on and off by using the setting application.

3.1.3 Operation of the NFC function

When the NFC function is set to On and the screen is unlocked, a general Android device waits for a communication target to show up in the communication range by switching between Poll Mode and Listen Mode irrespective of whether applications are on.

In Poll Mode, the device performs the Technology Detection Activity defined in [Activity] for all applicable technologies. Upon detecting a communication target, the following operations are performed:

1) If it is Android 9 or earlier, the device checks whether the target supports the NFC-DEP Protocol, and if it does, the device starts communication according to the NFC-DEP Protocol.
2) For Android 10 or later, or if the target does not support the NFC-DEP protocol, the device checks whether the target is an NDEF tag (contactless IC card having NDEF data, which hereinafter includes devices having the same function as a contactless IC card), and it tries to read the NDEF data if the target is an NDEF tag.
3) After this processing is finished, control is passed to an application using a mechanism called an "intent".

In Listen Mode, the device waits for communication from the Poller for all applicable technologies. Upon receiving communication from the Poller, the device starts communication according to the installed Secure Element, NFC-DEP Protocol, or host-based card emulation (HCE) depending on the request from the Poller and what the device supports. Then, an application is called as required.

3.2 Specifying a technology and protocol

As described earlier in this chapter, in an Android device, the Android OS determines the technology and protocol used for detection of and communication with a communication target in NFC communication. For this reason, if the communication target supports multiple technologies or protocols, communication can start with an unintended technology or protocol. Examples of a communication target that supports multiple technologies or protocols include another Android device, a contactless IC card that supports multiple technologies such as NFC-A and NFC-F, etc.

3.2.1 Setting reader mode (Android 4.4 or later)

To prevent the device from communicating with the communication target using an unintended technology or protocol, an active application may call the enableReaderMode method in the android.nfc.NfcAdapter class to set the NFC controller in the Android device to reader mode. Reader mode allows you to restrict the technology or protocol used for communication in Poll Mode. Also, reader mode does not allow communication in Listen Mode. This method is supported in API level 19 (Android 4.4) or later.

The following example shows the code that restricts the technology used for communication to NFC-F to prevent reading the NDEF data:

```
import android.nfc.NfcAdapter;
...
   NfcAdapter nfcAdapter = NfcAdapter.getDefaultAdapter(this);
   int flags = (FLAG_READER_NFC_F | FLAG_READER_SKIP_NDEF_CHECK);
   nfcAdapter.enableReaderMode(this, callback, flags, NULL);
```

"callback" is an object that includes the onTagDiscovered method called when the Android OS finds a communication target.

To end reader mode, call the disableReaderMode method as follows:

```
   nfcAdapter.disableReaderMode(this);
```

The technology can also be specified in the enableForegroundDispatch method, but this method configures the intent filter. The intent filter is applied after the Android OS determines the technology or protocol used for communication. Note that the enableForegroundDispatch method cannot restrict the technology or protocol used for communication.

3.3 Suppressing card presence checking

When the Android OS detects an NFC-F communication target in Poll Mode, which is a contactless IC card, it continues to send the SENSF_REQ (Polling) command with System Code set to FFFFh periodically to an application to check the presence of the contactless IC card even after it passes control. If no response to the SENSF_REQ command is received, the Android OS assumes that the contactless IC card no longer exists in the operating field, and stops generating a carrier. This might result in the problems described in section 2.5 "Mutual authentication is canceled when the presence of a card is checked".

The Android OS automatically checks the presence of the contactless IC card if the next command is not sent for a predetermined period of time (default: 125 ms) after the previous command and response exchange.

3.3.1 Setting reader mode (Android 4.4 or later)

By using reader mode, you can specify the interval at which the Android OS checks the presence of a contactless IC card. An active application can set reader mode by calling the enableReaderMode method in the android.nfc.NfcAdapter class. This method is supported in API level 19 (Android 4.4) or later.

The following code sample shows an interval setting of 120 seconds (120,000 ms) for contactless IC card presence checking:

```
import android.nfc.NfcAdapter;
import android.os.Bundle;
...
    NfcAdapter nfcAdapter = NfcAdapter.getDefaultAdapter(this);
    int flags = (FLAG_READER_NFC_F | FLAG_READER_SKIP_NDEF_CHECK);
    Bundle extras = new Bundle();
    extras.putInt(NfcAdapter.EXTRA_READER_PRESENCE_CHECK_DELAY, 120000);
    nfcAdapter.enableReaderMode(this, callback, flags, extras);
```

In the same way as in section 3.2.1 "Setting reader mode (Android 4.4 or later)", the code restricts the technology used for communication to NFC-F to prevent reading the NDEF data.

To suppress checking of presence of a contactless IC card indefinitely, you must set the interval of contactless IC card presence checking to a value larger than the maximum command transmission interval. In a system where the server generates commands, you can set a time a little longer than the server communication timeout time.

Once the presence of a contactless IC card is no longer checked, the Android OS cannot detect when the contactless IC card left the operating field. It is recommended to provide a function that suspends processing when a button on the screen is tapped because processing is not suspended automatically when the card is out of communication range.

To end reader mode, call the disableReaderMode method as follows:

```
    nfcAdapter.disableReaderMode(this);
```

3.3.2 Sending the Request Response command

In an Android OS before Android 4.4, you cannot use the method described in section 3.3.1 "Setting reader mode (Android 4.4 or later)". To suppress checking of presence of a contactless IC card, you must send the next command within a predetermined period of time (default: 125 ms) after the last command and response exchange. If a command to be sent next cannot be prepared, checking of presence of a contactless IC card can be suppressed by sending the following commands continuously within a predetermined period of time:

- Request Response command for a FeliCa Standard card
- Polling command for a FeliCa Lite-S card.

4 Precautions in the Windows OS

This chapter describes additional precautions for applications on Windows PCs and respective countermeasure examples.

NOTE In this chapter, “Windows PC” refers to any device running a Windows OS.

An application on a Windows PC in this document refers to an application managed and run in a Windows PC that can operate as an NFC/FeliCa-enabled device by connecting a Reader/Writer or equivalent device.

NOTE The program code sample in this chapter is for reference only and does not guarantee correct operation or mutual connectivity.

4.1 NFC function in a Windows PC

There are several ways to develop an application in a Windows PC that communicates with a contactless IC card, as follows:

- Use the PC/SC interface.
- Use the Proximity API.
- Use a proprietary API provided by the Reader/Writer vendor and others.

  The applicable API and its function differ depending on the Reader/Writer connected to the Windows PC.

  This chapter provides an example of how to use the PC/SC interface-based NFC function.

4.2 Suppressing NFC communication by the OS, drivers, etc.

On a Windows PC, multiple applications may be able to perform NFC communication simultaneously. Also, the OS or driver may perform NFC communication in the background. If the OS, driver or another application performs NFC communication after performing mutual authentication with a contactless IC card, mutual authentication can be canceled (as in section 2.5 "Mutual authentication is canceled when the presence of a card is checked") when the technology is switched or the carrier is suspended temporarily. To prevent this, NFC communication by the OS, driver or another application must be suppressed.

The method to suppress NFC communication by the OS, driver or another application depends on the Windows PC or Reader/Writer environment. Refer to the documentation for the API to be used.

4.3 Specifying a technology and protocol for the PC/SC interface

In the PC/SC interface, you can specify a technology and protocol by issuing the Manage Session, Transparent Exchange and Switch Protocol APDU commands by using the SCardControl function.

The following sample code restricts the technology to NFC-F and sends the SENSF_REQ command.

In this example, the Switch Protocol command specifies FeliCa as "type" and layer 2 as "layer".

```
#include <winscard.h>
. . .
const BYTE StartTransparentSession[]
   = {0xFF,0xC2, 0x00, 0x00, 0x02, 0x81, 0x00, 0x00};
const BYTE SwitchProtocol[]
   = {0xFF,0xC2, 0x00, 0x02, 0x04, 0x8F, 0x02, 0x03, 0x02, 0x00};
const BYTE TurnONtheRF[]
   = {0xFF,0xC2, 0x00, 0x00, 0x02, 0x84, 0x00, 0x00};
const BYTE TransparentExchange[]
   = {0xFF, 0xC2, 0x00, 0x01, 0x08,
      0x95, 0x06, 0x06, 0x00, 0xFF, 0xFF, 0x01, 0x03, 0x00};
const BYTE TurnOFFtheRF[]
   = {0xFF, 0xC2, 0x00, 0x00, 0x02, 0x83, 0x00, 0x00};
const BYTE EndTransparentSession[]
   = {0xFF, 0xC2, 0x00, 0x00, 0x02, 0x82, 0x00, 0x00};

SCardControl(handle, SCARD_CTL_CODE(3500),
   StartTransparentSession, 0x08, buffer, sizeof(buffer), &length);
SCardControl(handle, SCARD_CTL_CODE(3500),
   SwitchProtocol, 0x0A, buffer, sizeof(buffer), &length);
SCardControl(handle, SCARD_CTL_CODE(3500),
   TurnONtheRF, 0x08, buffer, sizeof(buffer), &length);
SCardControl(handle, SCARD_CTL_CODE(3500),
   TransparentExchange, 0x0E, buffer, sizeof(buffer), &length);
SCardControl(handle, SCARD_CTL_CODE(3500),
   TurnOFFtheRF, 0x08, buffer, sizeof(buffer), &length);
SCardControl(handle, SCARD_CTL_CODE(3500),
   EndTransparentSession, 0x08, buffer, sizeof(buffer), &length);
```

NOTE The code does not take into account error processing and communication interval.

For details of the APDUs handled by the PC/SC interface, refer to the following specification document provided by the PCSC WorkGroup:

“Interoperability Specification for ICCs and Personal Computer Systems
Part 3. Supplemental Document for Contactless ICCs”

When accessing the card by using SCardConnect and SCardTransmit, you cannot specify a technology and protocol by using a PC/SC function. In this case, check if a utility or configuration tool is provided by the device driver vendor.

5 Precautions for iOS

This chapter describes additional precautions for iOS applications, and respective countermeasure examples.

The program code example described in this chapter is for reference purposes only; it does not guarantee correct operation or mutual connectivity.

The description in this chapter is based on the API specifications of iOS as of December 2020. The latest API specifications are shown in the following website provided by Apple:

https://developer.apple.com/documentation/

In this chapter, a device running iOS is called an iOS device. Because the NFC function of iOS is created by referring to the specifications defined in the NFC Forum, an iOS device having the NFC function can be regarded as an NFC/FeliCa-enabled device.

5.1 Operation of the NFC function in iOS

This chapter describes the general specifications and operation of the NFC function in iOS.

5.1.1 Whether the NFC function is installed

Not all iOS devices have the NFC function. For example, the following are iOS devices, but:

- iPhone 7 (or later) has the NFC function and supports the Reader/Writer function.
  For it to send and receive any commands, however, it must have iOS 13 or later installed.
- iPhone 8 (or later) has NFC-F's card emulation function.
  In Japan, this function is also available on iPhone 7 and iPhone 7 Plus.

5.1.2 Operation of the NFC function

In an iOS device having the NFC function, the application can do NFC communication using the API.

An iOS device with "Background Tag Reading" enabled and Express Card settings enabled waits for a communication target to show up in the communication range by switching between Poll Mode and Listen Mode irrespective of whether applications are on.

In Poll Mode of "Background Tag Reading", the device performs the Technology Detection Activity defined in [Activity] for all applicable technologies. Upon detecting a communication target, the device checks whether the target is an NDEF tag, and it tries to read the NDEF data if the target is an NDEF tag. After this processing is finished, control is passed to an application associated with the read URL scheme.

5.2 Specifying a technology and protocol

On an iOS device, iOS detects the communication target in NFC communication. You can specify the technology and protocol used for communication from the application.

The specific sequence is as follows:

1) NFC-A /B/F detect (REQA, REQB, FFFF Polling, etc.).
2) If a communication target is detected, check that System identifiers of the technology, protocol, and application specified in the application match System Code, etc. that were set when the application was built.
3) If they all match, Session is started in the application.

The following sample code shows how to limit the technology to NFC-F and detect a contactless smart card:

```
var session: NFCTagReaderSession?
...
    session = NFCTagReaderSession(pollingOption: .iso18092, delegate: self)
    session?.alertMessage = "Hold your iPhone near a NFC-F tag."
    session?.begin()
```

After iOS detects a contactless IC card and establishes Session, the application can issue any command.

The session times out 20 seconds after the card is detected, however, so communication must be completed within 20 seconds after the card is detected.

5.3 Card presence checking

iOS detects the NFC-F communication target in Poll Mode, but does not check the presence of the card after Session is established.

iOS does not check the presence of the card after detecting the NFC-F communication target and establishing a session.

Therefore, the issue described in section 2.5 "Mutual authentication is canceled when the presence of a card is checked" is not confirmed.

5.4 Timeout time for FeliCa commands

In iOS, the command timeout of sendFeliCaCommand API is fixed at about 300 ms.

We recommend that you prevent the FeliCa command or commands from exceeding the iOS command timeout, by referring to section 2.6 "Timeout time for FeliCa commands”.

Appendix A Glossary

This appendix describes the main terms that are used in this document.

A.1 Glossary

<A>

APDU

Application Protocol Data Unit. The data unit for sending and receiving communication commands and responses, or its protocol.

<D>

Device Host

An Execution Environment responsible for the overall management of an NFC Forum Device and any peripherals.

<L>

Listener

An NFC Forum Device in Listen Mode, or a card.

Listen Mode

The mode in which an NFC Forum Device receives a command and sends a corresponding response.

<N>

NFC-DEP Protocol

The half-duplex block transmission protocol defined in NFC Digital Protocol and based on ISO/IEC 18092.

NFC Controller

The logical entity responsible for transmission data over the NFC radio interface. NFC Controller is also known as Contactless Front-end (CLF).

NFC Forum Device

A device that supports at least one communication protocol for at least one communication mode defined by the NFC Forum specifications.

<P>

PC/SC

Personal Computer/Smart Card. A specification for smart-card integration into computing environments. The standard for using IC cards or Reader/Writer on a computer.

Poller

An NFC Forum Device in Poll Mode, or Reader/Writer.

Poll Mode

The mode in which an NFC Forum Device sends a command and receives a corresponding response.

Proximity API

The API for accessing a proximity device, such as an NFC device, for the Windows Universal Windows Platform (UWP). Designed originally for store apps. Added first in Windows 8.

<S>

Secure Element

A component in a device providing security required to support various business models. A SE can exist in any form factor such as universal integrated circuit card (UICC), Embedded SE, memory card, etc.

<T>

Type 3 Tag

A contactless tag or smart card that supports NDEF and can be accessed by a device that implements the tag operation defined in the Type 3 Tag Technical Specification.

Type 3 Tag Platform

A set of technical components that supports NFC-F technology as the underlying layer of Type 3 Tag.

NFC

Development Guidelines for NFC/FeliCa-Enabled Devices and Applications and Applications

Version 1.4

January 2015	First Edition	FeliCa Business Division
May 2021	Revision	

Sony Corporation

No. M830-E01-40