SONY®

FeliCa

Technical Information

FeliCa Card User's Manual

Version 1.0

No. M617-E01-00

Introduction

This document describes the protocol specifications and the command specifications of any contactless IC card that utilizes FeliCa technology.

The purpose of this document is to provide basic information about the protocol specifications and the command specifications to customers who are engaged in the development of a Reader/Writer and application software that utilize FeliCa technology.

The objects of the descriptions in this document are the FeliCa-based contactless IC cards and IC chips sold by Sony Corporation.

For specific product names, see the following website:
http://www.sony.net/Products/felica/business/tech-support/index.html

This document contains no information about (a) security functions and security-related commands, (b) inspection / issuance specifications, and (c) specifications of individual products.

If you have any questions about the development of application software that is compatible with mobile FeliCa cards, please contact FeliCa Networks, Inc. (info-fn@FeliCaNetworks.co.jp)

The content of this document does not guarantee the correct operation of the System with all existing or future FeliCa cards.

Unless otherwise specified, the following notational conventions apply in this document:

- Numerical values are expressed in decimal notation.
- Hexadecimal values have "h" appended to each value (e.g., ffh).
- Binary values have "b" appended to each value (e.g., 0001b).
- Unless otherwise specified, the Byte order is Big Endian.

In this document, FeliCa Standard card is simply expressed as "card".

This document does not apply to FeliCa Lite series and FeliCa Plug series.

FeliCa technology refers to the following standards:

- JIS X 6319-4: Specification of implementation for integrated circuit(s) cards – Part 4: High speed proximity cards
- ISO/IEC 18092: Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol-1 (NFCIP-1)

Contents

(Blank page)

1 Overview

This chapter describes the general organization of this document.

Chapter 2 describes the communication protocol of FeliCa technology.

Chapter 3 describes the file system of FeliCa card.

Chapter 4 describes the general commands used by FeliCa card.

Note:

This document does not describe the detailed specifications of any security-related commands.

Chapter 5, Chapter 6 and Chapter 7 are placeholders for FeliCa card security, inspection and issuance specifications, all of which are beyond the scope of this document. For details of where those specifications are described, see the optional agreement.

2 Communication protocol

This chapter describes the communication protocol used for communication with FeliCa card and is organized as follows:

- Physical layer
  This layer defines the physical and electrical characteristics of data transfer.
- Data link layer
  This layer defines the data transfer method and the error detection scheme.
- Application layer
  This layer defines the specifications and functions of data strings to be handled as commands.

2.1 Physical layer

Table 2-1 shows the transmission characteristics of the physical layer of RF communication with FeliCa card.

Table 2-1: Transmission characteristics of physical layer of RF communication interface

Data transfer method	Half duplex, synchronous system
Carrier frequency	13.56MHz
Modulation method	ASK
Bit coding	Manchester code, MSB first
Data transfer rate	212kbps / 424kbps

2.2 Data Link layer

Data transfer between the Reader/Writer and the card is performed on a packet-by-packet basis, as defined in the data link layer. For definitions of fields in a packet and the packet structure, see Table 2-2 and Figure 2-1.

Table 2-2: Definition of fields in a packet

Field Name	Byte length	Definition
Preamble	6	(00 00 00 00 00 00)h
Sync code	2	(b2 4d)h
Data length (LEN)	1	Value of n (Byte length of packet data) + 1 (Byte length of LEN)
Packet data	n	Command packet data / Response packet data (to be defined on a command-by-command basis)
CRC	2	Checksum of data length and packet data based on CRC-CCITT (Big Endian) Initial value: 00 00h Generator polynomial: $X^{16} + X^{12} + X^5 + 1$

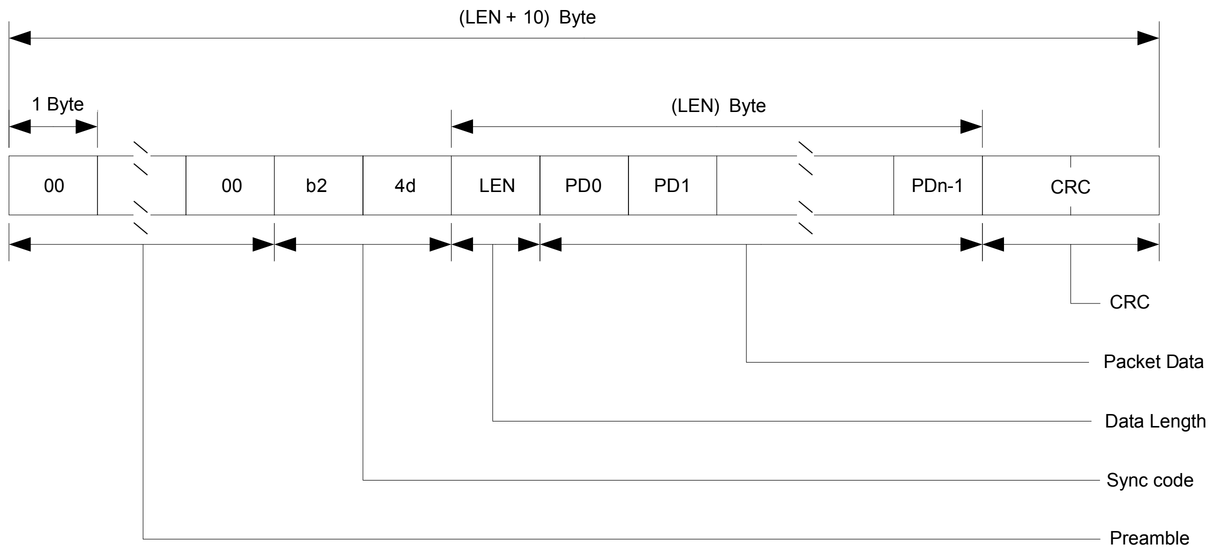


Figure 2-1: Packet Structure

2.3 Application layer

This section describes the rules applied to Packet Data (i.e., the data contained in a packet). It also describes the rules that govern how the parameters contained in the Packet Data are processed in accordance with the communication protocol.

In this document, the Packet Data received by the card is known as command packet, and the Packet Data transmitted from the card is known as response packet.

2.3.1 Command packet

A command packet consists of a command code (i.e., the first byte) followed by command data.

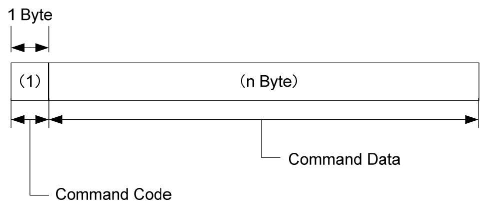


Figure 2-2: Command packet

<Command code>

The command code identifies the type of command. Table 2-3 shows the list of commands available. For detailed information about each command, see section “4.4 Command specifications”.

<Command data>

The command data are defined on a command-by-command basis. For information about the contents to be defined, see section “4.4 Command specifications”.

2.3.2 Response packet

A response packet consists of a response code (i.e., the first byte) and response data.

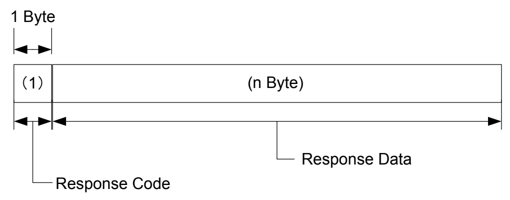


Figure 2-3: Response packet

<Response code>

The response code identifies the type of response. Table 2-3 shows the list of response codes corresponding to the commands available.

<Response data>

The response data are defined on a command-by-command basis. For information about contents to be defined, see the command descriptions in section “4.4 Command specifications”.

Table 2-3: List of commands

Command name	Command code	Response code	Function overview
Polling	00h	01h	Use this command to acquire and identify a card.
Request Service	02h	03h	Use this command to verify the existence of Area and Service.
Request Response	04h	05h	Use this command to verify the existence of a card.
Read Without Encryption	06h	07h	Use this command to read Block Data from a Service that requires no authentication.
Write Without Encryption	08h	09h	Use this command to write Block Data to a Service that requires no authentication.
Request System Code	0ch	0dh	Use this command to acquire the System Code registered to a card.
Authentication1	10h	11h	Use this command to authenticate a card.
Authentication2	12h	13h	Use this command to allow a card to authenticate a Reader/Writer.
Read	14h	15h	Use this command to read Block Data from a Service that requires authentication.
Write	16h	17h	Use this command to write Block Data to a Service that requires authentication.

2.3.3 Manufacture ID and Manufacture Parameter

This section describes Manufacture ID (IDm) and Manufacture Parameter (PMm). IDm and PMm can be acquired as the response data to the Polling command. Figure 2-4 shows the configuration of IDm and PMm.

IDm and PMm are set up only once at the time of card manufacture; no change is possible after completion of the setup. All the setup values are defined on a product-by-product basis.

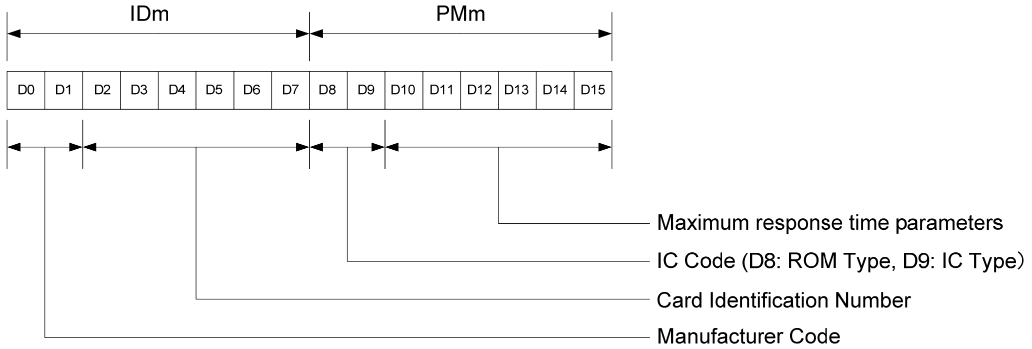

IDm
PMm
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
D14
D15
Maximum response time parameters
IC Code (D8: ROM Type, D9: IC Type)
Card Identification Number
Manufacturer Code

Figure 2-4: IDm and PMm

<Manufacture ID (IDm)>

Using Manufacture ID (IDm), the Reader/Writer identifies a card to be the counterpart of communication. If multiple Systems exist on a card, IDm is set to each of these Systems.

As shown in Figure 2-4, IDm consists of Manufacturer Code and Card Identification Number.

The upper 4 bits of the 1 Byte of data located at the top of Manufacturer Code indicate the System Number in the card. The System Number is automatically incremented by one (1) every time a System Separation is performed. The higher 4 bits of IDm of System0 is 0000b, so (for example), the higher 4 bits of IDm of System1 becomes 0001b.

<Manufacture Parameter (PMm)>

As shown in Figure 2-4, PMm consists of the information (2 Bytes) known as IC Code (this code is used to identify the product) and the Maximum Response Time (6 Bytes; this parameter is used to determine the timeout period of each command). For a detailed description of each Byte from D10 to D15, see the following section "<Maximum response time>".

IC Code consists of ROM Type and IC Type.

<Maximum response time>

Timeout time is determined based on the period of time necessary for processing commands. Therefore (and because this period of time depends on the status of the card as well as on the type and content of each command), the Reader/Writer must dynamically determine the timeout time. In FeliCa technology, the maximum response time is determined by using the lower 6 Bytes of the PMm parameter. The card provides this parameter to the Reader/Writer, enabling the Reader/Writer to dynamically determine the timeout time.

The Maximum Response Time Parameter is the data string of 6 Bytes configured as shown in Figure 2-5. The Reader/Writer makes reference to 1 Byte of data located in PMm that corresponds with the command, and determines the maximum response time using the calculation formula shown in Figure 2-6. Additionally, acquisition of PMm is made possible by the Polling command.

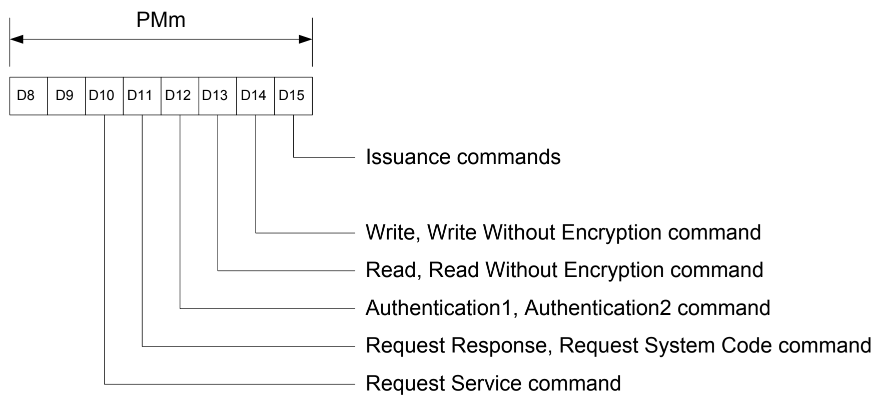
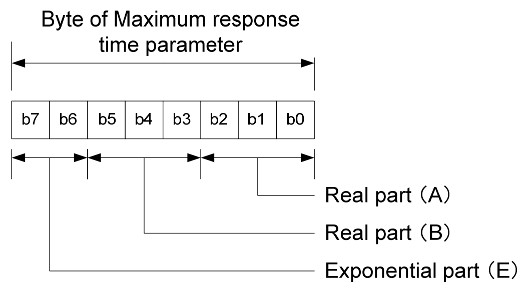

PMm
D8
D9
D10
D11
D12
D13
D14
D15
Issuance commands
Write, Write Without Encryption command
Read, Read Without Encryption command
Authentication1, Authentication2 command
Request Response, Request System Code command
Request Service command

Figure 2-5: Maximum response time parameters



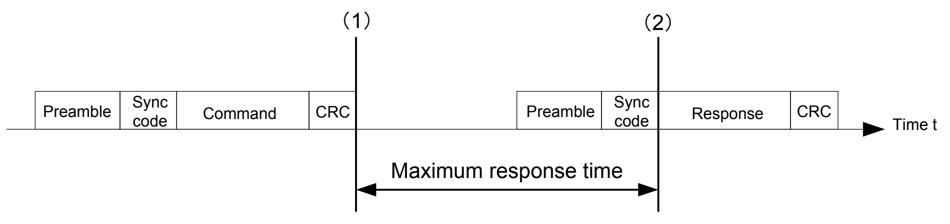
Timeout time [ms] = 0.3020 × [(B + 1) × n + (A + 1)] × 4^E

n : Number of blocks for Read, Write, Read Without Encryption and Write Without Encryption command
Number of Areas or Service for Request Service command
Sum total of Areas and Services for Authentication1 command
Zero (0) for other commands

Figure 2-6: Calculation formula for determination of maximum response time

In a FeliCa card, the process time of each command is measured based on the definition of intervals shown in Figure 2-7, and the value of the maximum response time is determined.

For the Polling command, a response time different from the ones for the other commands is defined. For details, see “2.3.4 Anti-collision ”.



(1) The point in time when transmission of all data in the command packet from the Reader/Writer is completed
(2) The point in time when transmission of the Sync code in the response packet from the card is completed

Figure 2-7: Definition of maximum response time

2.3.4 Anti-collision process

To identify a card, the Reader/Writer must poll an unspecified number of cards, by using the Polling command. If multiple cards exist within the range where communication between the Reader/Writer and the cards is possible, and if these cards respond to the Polling command simultaneously, however, the Reader/Writer is unable correctly to receive the responses returned from the cards. Therefore, FeliCa technology adopts a method known as Time Slot method to reduce the probability of collision between the responses returned from multiple cards.

<Time slot method>

Sections on the time axis divided at regular intervals are known as "time slots". Both the Reader/Writer and the card have the same number of (i.e., "n") time slots, and these slots are mutually synchronized. When a Polling command is received, the card selects a time slot in a random manner and then transmits a response to the Polling command only in the selected slot. When the Reader/Writer performs polling to cards under the previously-mentioned assumptions, it is expected that the cards return responses to the polling in a random manner in each time slot. This reduces the probability of collision between responses to a Polling command sent to multiple cards.

In FeliCa technology, the start time of the first time slot is known as "Response time (A)", and the width (i.e., duration) of the time slot is known as "Response time (B)". These response times are defined as follows:

- Response time (A) 2.417 [ms]
- Response time (B) 1.208 [ms]

The number of time slots (i.e., "n") to be shared between the Reader/Writer and the card are specified by the Polling command. For details, see “4.4.2 Polling ”.

Figure 2-8 shows an example of response times of the cards to the Polling command where the number of time slots is "4", and there are two cards within communication range of the Reader/Writer. This diagram shows the case where Card 1 selected slot #1 and Card 2 selected slot #3 of four time slots specified by the Reader/Writer.

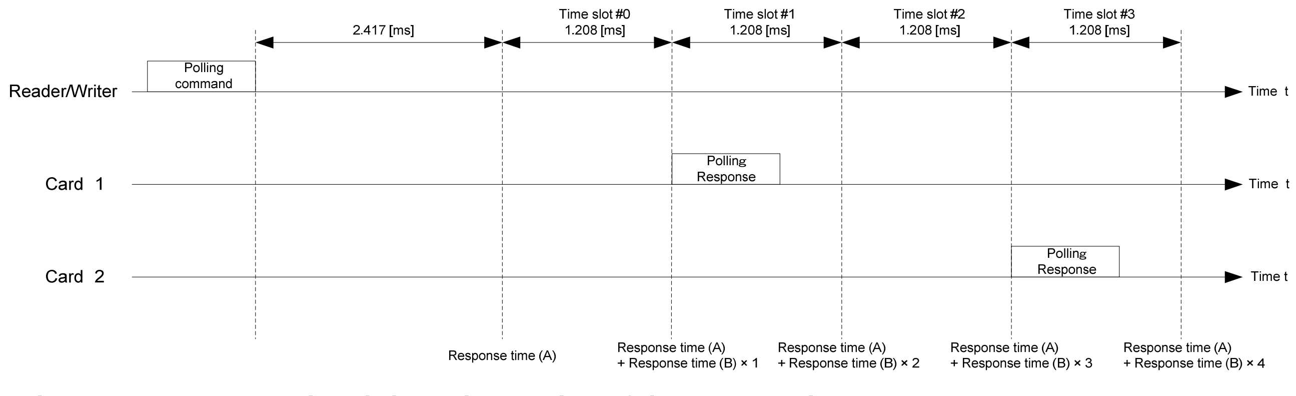


Figure 2-8: Response time (where the number of time slots = 4)

<Identification of communication destination by IDm>

When a response packet to the Polling command is received correctly, the Reader/Writer can acquire a parameter known as IDm, which is contained in the response packet to the Polling command. By setting IDm to the command packet, communication with a specific card becomes possible even if multiple cards are in proximity of the Reader/Writer. When a Polling command is received, each card refers to IDm and, if such a card detects that the command packet is not addressed to itself, the card returns no response.

3 File system

A basic concepts known as Area and Service are introduced to the file system. By defining Area and Service, users can design their original methods of utilizing / accessing non-volatile memory space.

This chapter describes the concept of file system.

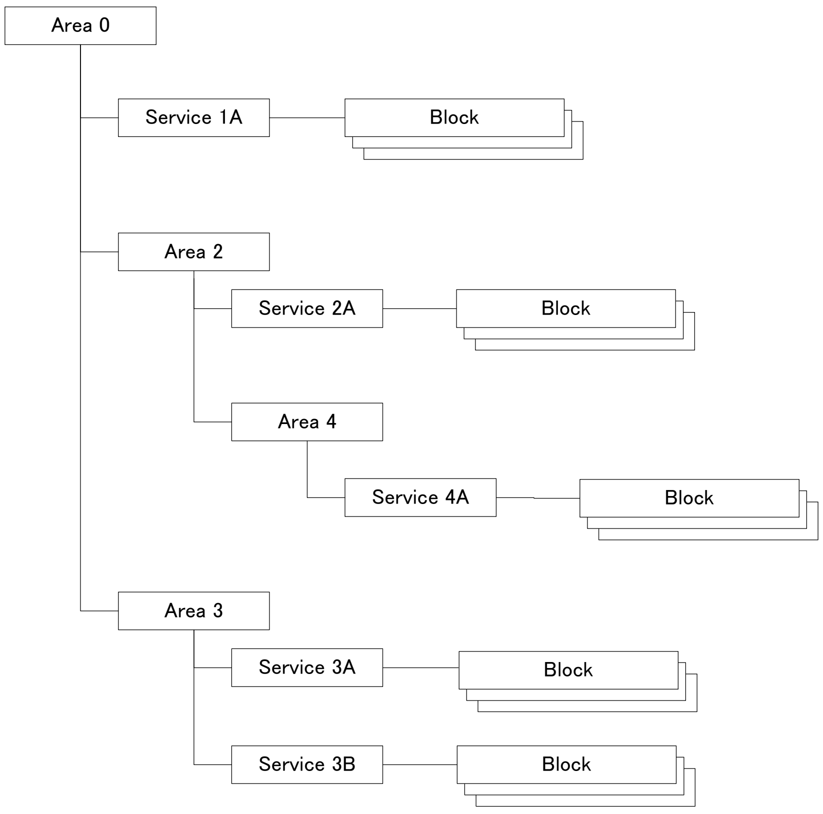


Figure 3-1: Concept diagram of file system

3.1 Block

In this file system, management of non-volatile memory space is performed with the minimum recording unit of 16 Bytes. This minimum recording unit is known as Block.

All the user data are stored to Block. Access to the memory space from the user is performed on a Block-by-Block basis. Therefore, it is necessary to divide the data into multiple Block to store user's data exceeding 16 Bytes. In addition to user's data, the management information of the file system and so on is stored in the Block.

All the management of the Block located in non-volatile memory space is performed by the file system. Therefore, the user does not need to perform any direct operation on the Block, which are accessed by using a mechanism known as Service (described later in this document).

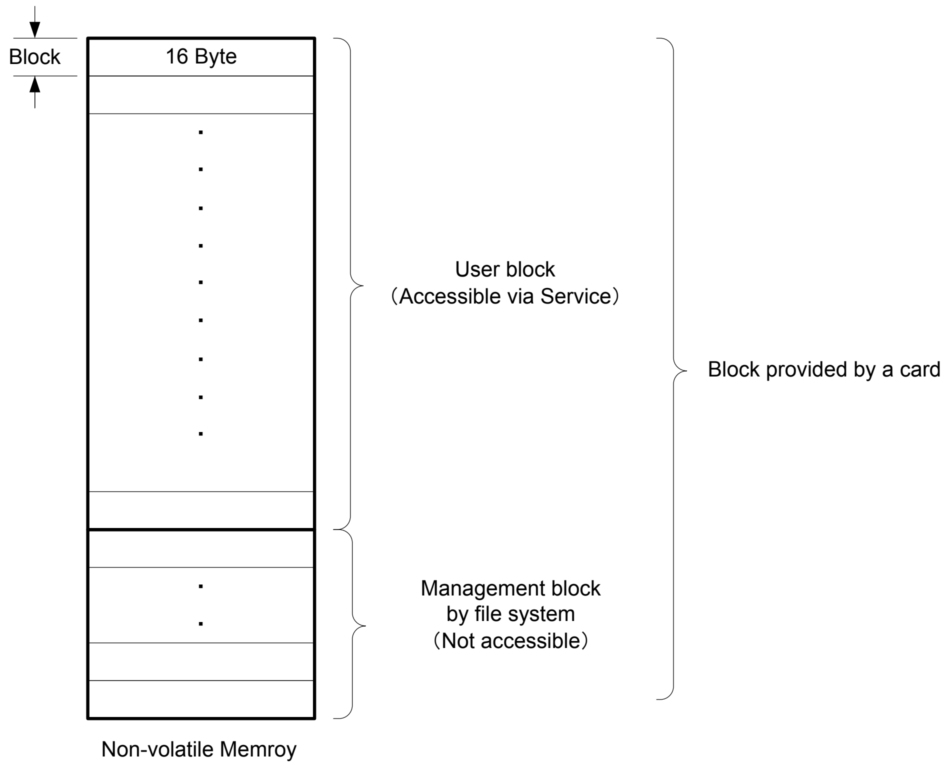


Figure 3-2: Blocks in non-volatile memory

3.2 Service

"Service" is a group of Block located on the file system. Service provides access control to the Block so grouped.

All access to the Block is performed by using "Service". Therefore, access to the Block in non-volatile memory becomes possible by registration of Service to the file system.

To access Block under management of a Service, first identify the Service with a code of 2 Bytes known as Service Code. Then, by using a 1-Byte number known as Block Number, specify a Block located in the range under management of the Service specified by the Service Code. Block Number starts from zero (0) within a Service.

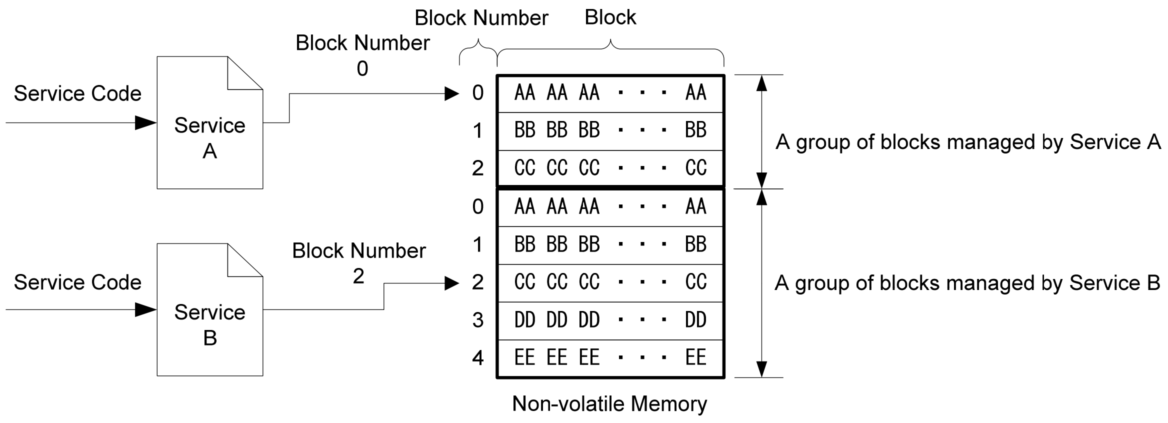


Figure 3-3: Image of access to Block by Service

There are three (3) types of Service with different access methods. For each type of Service, it is possible to set access attributes such as "Read / Write" or "Read Only", and "necessity of authentication before using the Service" to each of them.

- Random Service

  This is the general-purpose Service. This Service provides access to Block and can read and write data by specifying any Block.

- Cyclic Service

  This is the Service that assumes the role of log management. This Service provides access to Block and deletes the oldest data every time new data is written.

- Purse Service

  This is the Service that assumes the role of fee collection. This Service provides access to Block and automatically performs numerical operations on specific data in the Block.

The appropriate types of Service and access attribute are determined by Service Code. Different access attributes can be set, depending on the type of Service. To set the access attribute to a specific Service, see section “3.2.1 Random Service”, “3.2.2 Cyclic Service” or “3.2.3 Purse Service”.

<Service Code>

Service Code is determined with a number to be assigned at the discretion of the user and by the access attribute, and in accordance with the format shown in Figure 3-4.

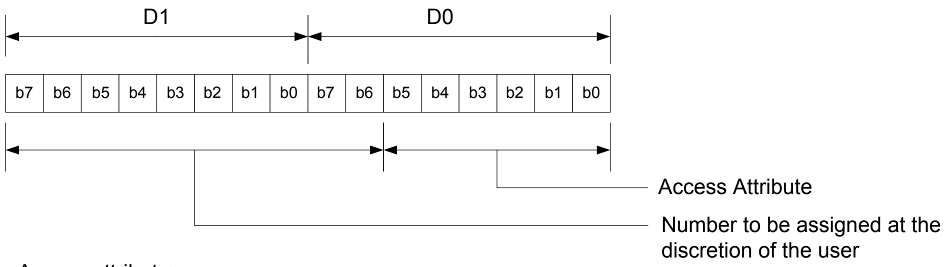
Example:

(1) Read / Write Access: Service Code of Random Service that requires authentication

(0001 0010 1100 1000)b = 12c8h

(2) Read / Write Access: Service Code of Random Service that requires no authentication

(0001 0010 1100 1001)b = 12c9h



<Access attribute>

001000b: Random Service (Read/Write access: Authentication required)
001001b: Random Service (Read/Write access: Authentication not required)
001010b: Random Service (Read Only access: Authentication required)
001011b: Random Service (Read Only access: Authentication not required)

001100b: Cyclic Service (Read/Write access: Authentication required)
001101b: Cyclic Service (Read/Write access: Authentication not required)
001110b: Cyclic Service (Read Only access: Authentication required)
001111b: Cyclic Service (Read Only access: Authentication not required)

010000b: Purse Service (Direct access: Authentication required)
010001b: Purse Service (Direct access: Authentication not required)
010010b: Purse Service (Cash-back access/Decrement access: Authentication required)
010011b: Purse Service (Cash-back access/Decrement access: Authentication not required)
010100b: Purse Service (Decrement access: Authentication required)
010101b: Purse Service (Decrement access: Authentication not required)
010110b: Purse Service (Read Only access: Authentication required)
010111b: Purse Service (Read Only access: Authentication not required)

Figure 3-4: Configuration and Access Attribute of Service Code

3.2.1 Random Service

Random Service is a general-purpose service that allows access to Block specified at the discretion of the user.

<Access attribute>

Random access is provided with four types of access attributes as listed in the following table:

Access Attribute	Description
Read / Write Access: authentication is necessary.	Both reading and writing of data is possible. Authentication of Service is necessary.
Read / Write Access: authentication is unnecessary.	Both reading and writing of data is possible. Authentication of Service is unnecessary.
Read Only Access: authentication is necessary.	Only reading of data is possible. Authentication of Service is necessary.
Read Only Access: authentication is unnecessary.	Only reading of data is possible. Authentication of Service is unnecessary.

<Structure of Block>

Any data can be stored to a Block.

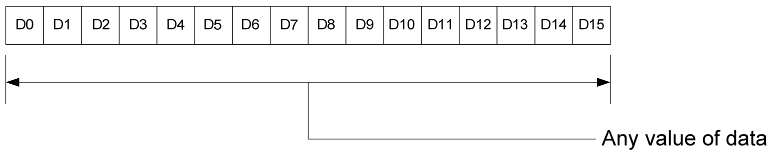


Figure 3-5: Block Data of Random Access

<Specifying a Block>

A Block can be specified by using a Block Number.

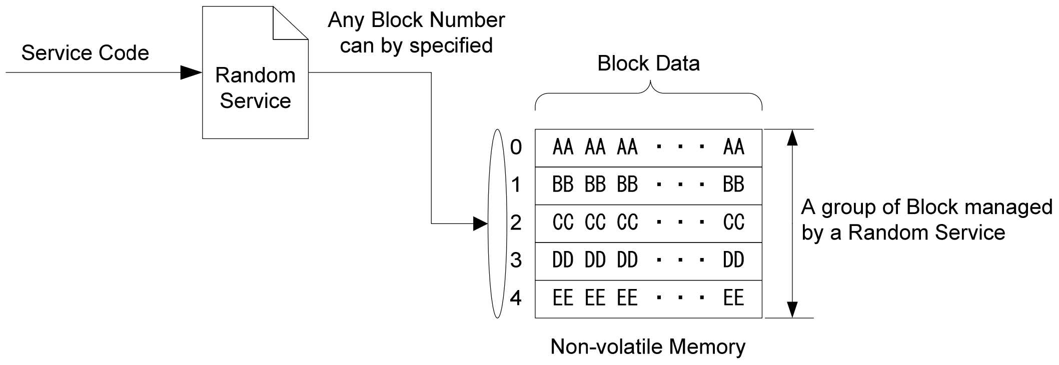


Figure 3-6: Specifying a Block in Random Service

3.2.2 Cyclic Service

Cyclic Service provides a special function when accessing Block associated with the "recording of logs" as the use case. In each case, data is written to the Block containing the oldest data. This method of data writing enables cyclic use of a group of Block.

While new data is automatically and sequentially overwrites the oldest data first, there is a risk of unintentional loss of existing data if the same data is repeatedly and indiscreetly written. To prevent this occurring, Cyclic Service has a function that compares the oldest data in the target Block with the data to be written. If both sets of data are identical, the command completes normally but the data in the target Block is not updated.

In FeliCa card, a single command can be used to write data simultaneously to multiple Block. In this case, each Block is handled as an independent data unit. In Cyclic Service, however, when a sequential data write to the same Cyclic Service is performed, such sequential Block Data are grouped together and handled as a single unit of data, with the following consequences:

- Data comparison to determine identity at the time of data writing is performed between such groups of sequential Block Data, not between the data in each Block.
- If the newly-grouped Block Data completely matches any older data that are stored across multiple Block, the older data are not updated.
- Even if a data log is not held within a single Block but distributed over several Block, the risk of unintentional loss of existing data can be avoided.

<Access Attribute>

In Cyclic Service, four access attributes are provided as listed in the following table:

Access attribute	Description
Read / Write Access: authentication is necessary.	Both reading and writing of data is possible. Authentication of Service is necessary.
Read / Write Access: authentication is unnecessary.	Both reading and writing of data is possible. Authentication of Service is unnecessary.
Read Only Access: authentication is necessary.	Only reading of data is possible. Authentication of Service is necessary.
Read Only Access: authentication is unnecessary.	Only reading of data is possible. Authentication of Service is unnecessary.

<Structure of Block>

Any data can be stored to a Block.

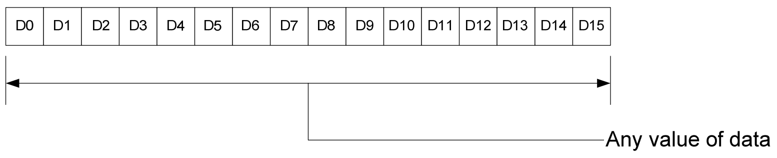


Figure 3-7: Structure of Block in Cyclic Service

<Specifying a Block>

When reading data, it is possible to specify any Block Number.

In Cyclic Service, the latest data is read from the Block when the Block Number is "0". It is possible to read the older data by increasing the value of Block Number.

When writing data, it is necessary always to specify "0" to the Block Number.

With the function of Cyclic Service, the oldest data at that time is automatically overwritten by the new data, and the group of Block is used in a cyclic manner. It is impossible to specify a destination Block of data writing.

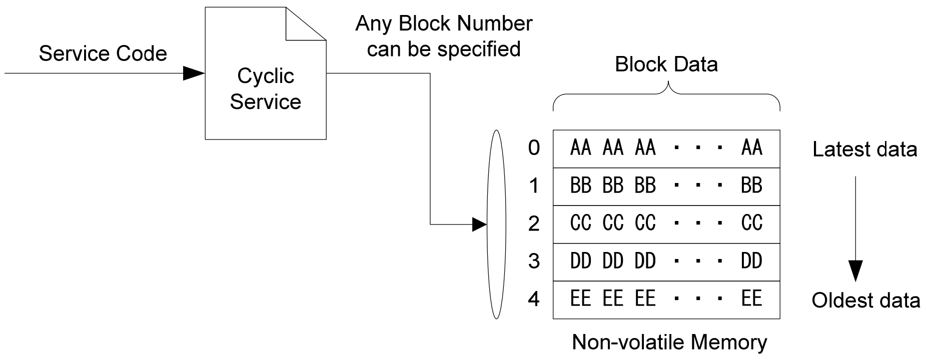


Figure 3-8: Specifying a Block in data read operation during Cyclic Service

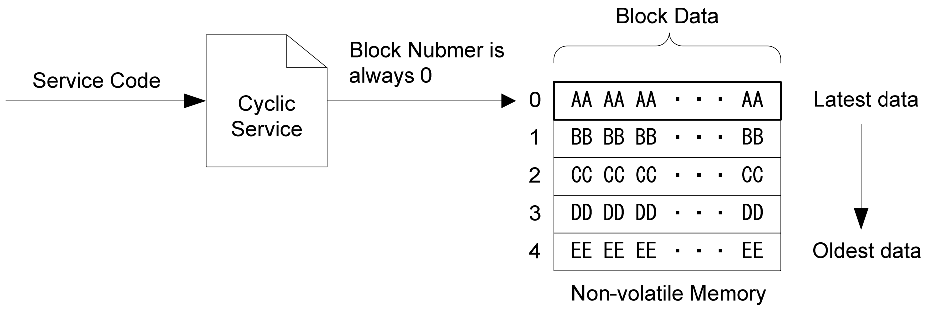


Figure 3-9: Specifying a Block in data write operation during Cyclic Service

3.2.3 Purse Service

Purse Service provides special functions when accessing Block associated with "fee collection" as a use case. For the Block under the management of this Service, the fields are defined as shown in Table 3-1. For the data stored to each field, it is possible to automatically perform numeric operations at the time of access using the functions described in the following list. Block List Element to be described later is used to specify an operation function. For details of the Block List Element, see “4.2 Access to Block”.

Decrement Function

With this function, the purse data is decremented by the specified value. At the same time, the value so decremented is stored to cash-back data. The value to be decremented is specified by the Block Data to be written.

Cash-back Function

Up to a ceiling of the value stored to cash-back data, the specified value is added to purse data (i.e., cash-back). When a cash-back operation is performed, the cash-back data is reset to zero ("0"), regardless of the value added to the purse data. The value to be added to the purse data is specified by the Block Data to be written.

In addition, a parameter known as "Execution ID" is available in Purse Service. During Purse Service operation, this parameter compares the Execution ID of Block Data to be written and the Execution ID of the Block Data at the data write destination. If Execution ID of both Block Data is identical, the data-write command normally completes but update (such as decrement, increment, etc.) of the data in the Block is not performed. Even if a command requesting data write to the same Purse Service was retransmitted because of communication difficulties, etc., this function prevents the data from being repeatedly decremented.

Table 3-1: Fields of Block in Purse Service

Field	Description
Purse data	This is the field to store data such as remaining value, etc.
Cash-back data	This is the field to store the value decremented from purse data.
User data	This is the field possible to store any of data.
Execution ID	This is the field to store the Execution ID when the Block was updated.

<Access Attribute>

In Purse Service, eight access attributes are provided, as listed in the following table. In Purse Service, the data structure of the Block to be written is defined on an attribute-by-attribute basis of access. For details of each Block structure, see the illustrations specified in "Structure of Block" column. In data-read operations, the data structure shown in Figure 3-10 is always used.

If letter "Y" is indicated in "Execution ID" column, comparison of Execution ID is performed before the data-write operation. If Execution ID is identical, no data writing is performed. If letter "N" is indicated in "Execution ID" column, comparison of Execution ID is not performed before the data-write operation. Data writing is performed even if Execution ID is identical.

Table 3-2: List of Access Attribute

Access Attribute	Description	Structure of Block	Execution ID
Direct access: authentication is necessary.	Both reading and writing of data is possible. Authentication of Service is necessary.	Figure 3-10	N
Direct access: authentication is unnecessary.	Both reading and writing of data is possible. Authentication of Service is unnecessary.	Figure 3-10	N
Cash-back / Decrement access: authentication is necessary.	Data write operation utilizing decrement function and cash-back function as well as ordinary data read operation are possible. Authentication of Service is necessary. Selection between decrement function and cash-back function is specified with the access mode having data structure known as "Block List".	Figure 3-11 or Figure 3-12	Y
Cash-back / Decrement access: authentication is unnecessary.	Data write operation utilizing decrement function and cash-back function as well as ordinary data read operation are possible. Authentication of Service is unnecessary. Selection between decrement function and cash-back function is specified with the access mode having data structure known as "Block List".	Figure 3-11 or Figure 3-12	Y
Decrement access: authentication is necessary.	Data write operation utilizing decrement function as well as ordinary data read operation are possible. Authentication of Service is necessary.	Figure 3-11	Y
Decrement access: authentication is unnecessary.	Data write operation utilizing decrement function as well as ordinary data read operation are possible. Authentication of Service is unnecessary.	Figure 3-11	Y
Read only access: authentication is necessary.	Only reading of data is possible. Authentication of Service is necessary.	Figure 3-10	–
Read only access: authentication is unnecessary.	Only reading of data is possible. Authentication of Service is unnecessary.	Figure 3-10	–

<Structure of Block>

Data types that can be stored and the store method of data differ, depending on the access attribute.

Note: The cash-back data and the purse data set to the Block Data are in Little Endian format.

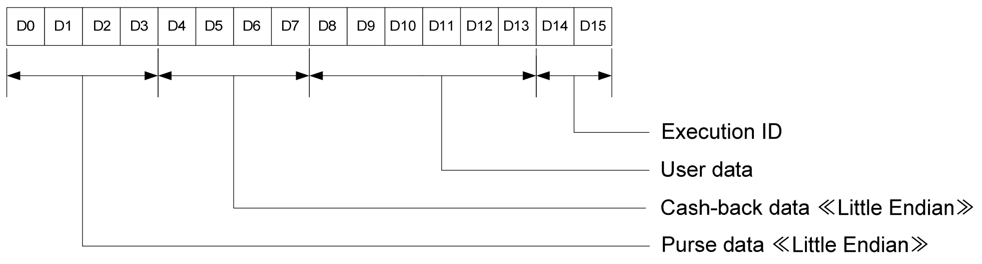

D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15
Execution ID
User data
Cash-back data ≪Little Endian≫
Purse data ≪Little Endian≫

Figure 3-10: Block Data in Direct Access (data read)

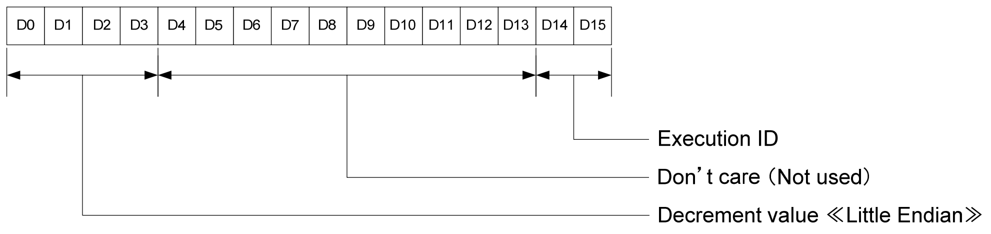

D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15
Execution ID
Don't care (Not used)
Decrement value ≪Little Endian≫

Figure 3-11: Block Data when used Decrement function

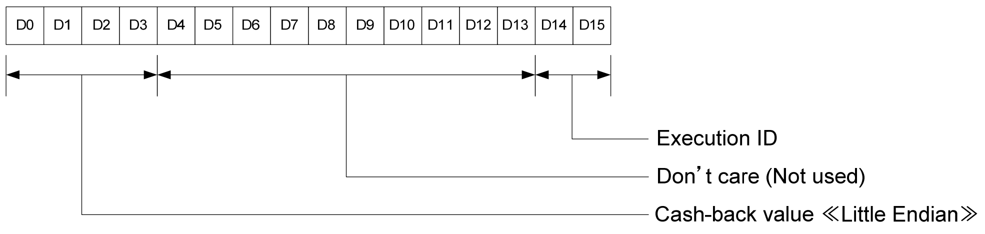

D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15
Execution ID
Don't care (Not used)
Cash-back value ≪Little Endian≫

Figure 3-12: Block Data when used Cash-back function

<Specifying a Block>

It is possible to specify a Block, by using any Block Number.

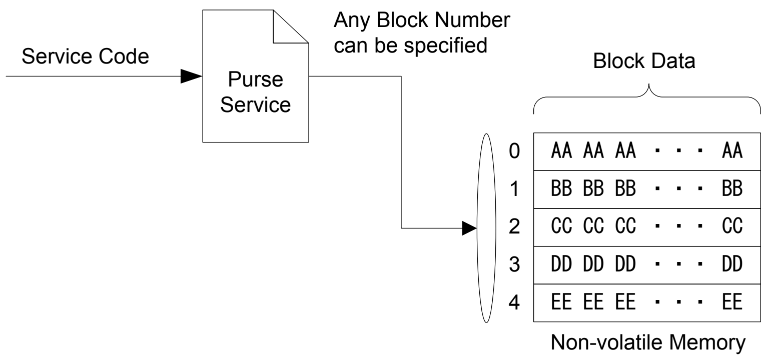


Figure 3-13: Specifying a Block in Purse Service

3.2.4 Overlap Service

In FeliCa card, management of Block Data located in non-volatile memory can be performed by using multiple Service Code. This allows the Block Data to be set up so that it requires authentication for Read / Write access, but does not require authentication for Read Only access. The process of managing shared Block Data by using multiple Service Code is known as "Overlap", and the Service that utilizes the Overlap process is known as "Overlap Service".

If you use an Overlap Service, you must take the following two restrictions into account:

- It is impossible to use Random Service together with Cyclic Service and Purse Service, or vice versa. For example, it is impossible to overlap Purse Service onto a Block under the management of Random Service.
- The Overlap operation is impossible if the number of Block in each group of shared Block differ. For example, it is impossible to overlap a Random Service with read / write access managing twenty Block onto a Random Service with Read Only access managing ten Block.

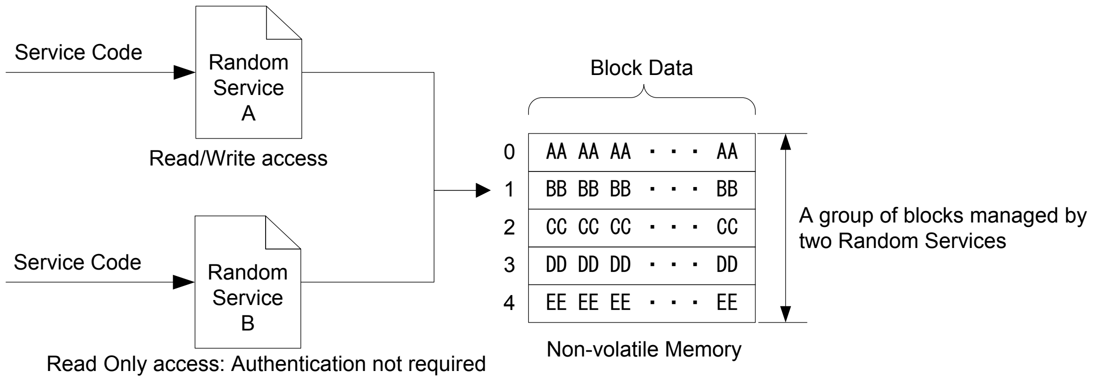


Figure 3-14: Image of operation of Overlap Service

3.3 Area

As described in the previous section, a concept of "Service" is used in the management of access control to Block. On the other hand, a concept of "Area" is used in the management of the remaining usable Block in non-volatile memory space or of assignment of Block to a Service.

All the Service are under the management of any of the Area. So when a Service is registered, Block to be managed by that Service are assigned from the Block under the management of the Area. It is also possible to manage Area nested inside one another. This allows the Block under the management of an Area to be assigned to and be managed by the other Area.

An Area is identified by a code of 2 Bytes known as Area Code. It is possible to assign one of the following attributes to an Area Code:

Creation of Area is possible

It is possible to create Sub-Area nested inside an Area (i.e., it is possible to create both "Area" and "Service").

Creation of Area is impossible

It is impossible to create Sub-Area nested inside an Area (i.e., it is possible to create "Service" only).

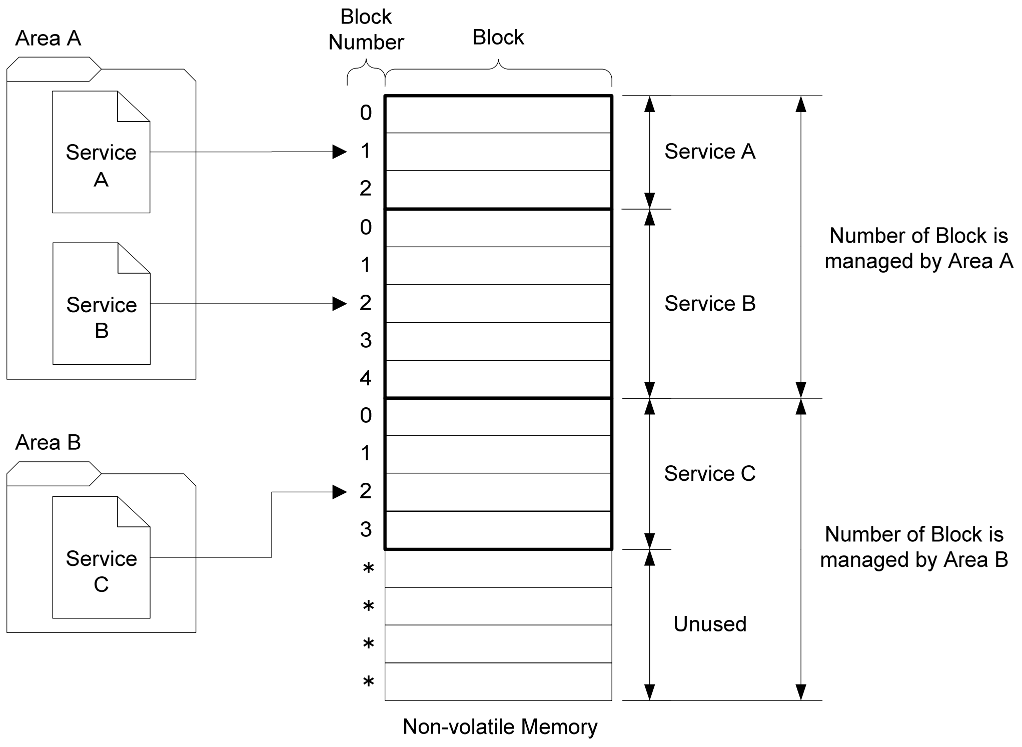
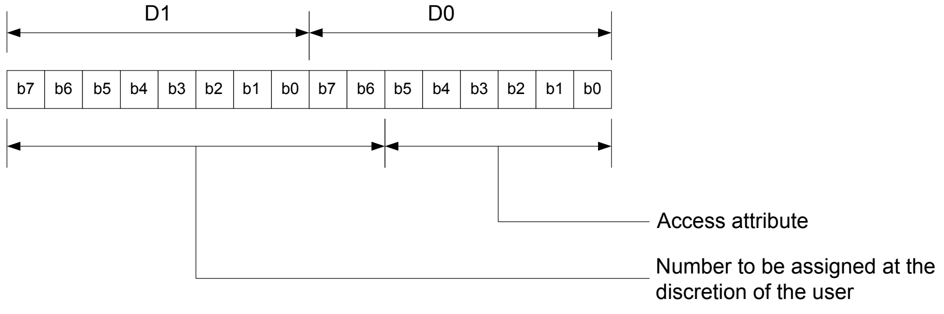


Figure 3-15: Image of management of Block by Area

<Area Code>

Depending on whether creation of Sub-Area nested inside an Area is necessary, the Area Code to be registered shall be specified in accordance with the format shown in Figure 3-16.



<Access attribute

000000b: Area（Creation of Sub-Areas is possible）

000001b: Area（Creation of Sub-Areas is impossible)

Figure 3-16: Structure and access attribute of Area Code

Example (1): Area Code = 0000h

In this case, (D1 D0) = (0000 0000 0000 0000)b, so registration of Sub-Area is possible for Area 0.

Example (2): Area Code = 12c0h

Here, (D1 D0) = (0001 0010 1100 0000)b, so registration of Sub-Area is possible for Area 12c0.

Example (3): Area Code = 12c1h

Here, (D1 D0) = (0001 0010 1100 0001)b, so registration of Sub-Area is impossible for Area 12c1.

3.3.1 Area 0

The relationship between Area and Service is the logical hierarchical structure shown in Figure 3-1. The Area to be the root of this hierarchical structure is known as "Area 0". Area 0 always exists in a card, and its Area Code is specified as 0000h.

3.4 System

A "System" is the normative unit to be handled as a single sheet of logical card. For a single sheet of card, it is possible to create multiple Systems with the procedure known as System Separation (see Figure 3-7).

For a card where multiple Systems exist as a result of the System Separation procedure, each System is known as System0, System1, …, System n. If System Separation is not performed (i.e., only a single System exists on the card), it is synonymous with the state where only System0 exists on the card.

<System Code>

Area / Service are provided with Area / Service Code as the parameters to identify themselves. In the same way, a System is provided with a code of 2 Bytes known as "System Code" for identification of itself. A Reader/Writer uses System Code to identify a card (i.e., System).

When identifying a card, the Reader/Writer must poll an unspecified number of cards with the Polling command. In this case, a System Code is specified as the parameter of the Polling command, and a System returns a response only when its System Code matches the System Code in the parameter of the Polling command at a preliminary stage of the anti-collision process. Even if the System of a card is divided, the Reader/Writer identifies each System as a single card unit. Therefore, the Reader/Writer can capture the destination System by specifying one of the System Code from System0 to System n in the Polling command.

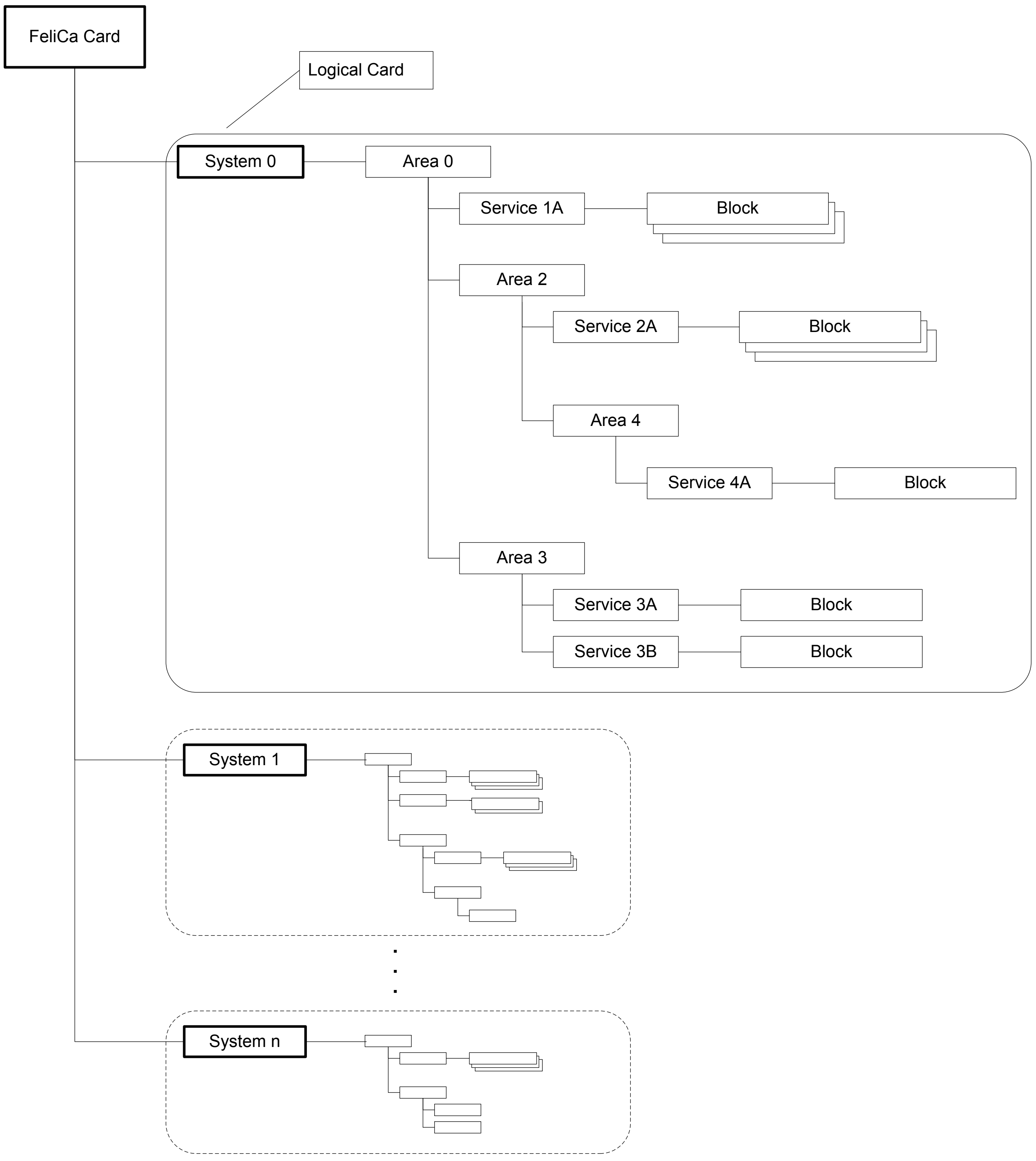


Figure 3-17: Concept diagram of file system containing Systems

3.4.1 Switching between Systems

In FeliCa technology, a function that "destination System of command packet returns a response in place of the System currently active" is available. Even when a System (i.e., System A) received a command packet addressed to the other System (i.e., System B) existing on the card, this function causes System B to return a response instead of System A. This is known as "Switching between Systems". When "Switching between Systems" occurs, the mode of the card transits to Mode0 and mutual authentication status becomes dissolved. Therefore, it is necessary to establish mutual authentication again after completion of Switching between Systems when accessing a Service that requires authentication.

The following two methods are available to perform Switching between Systems:

1. Switching between Systems by using the Polling command:

   Select a System by specifying System Code of the System you want to switch to the parameter of the Polling command.

2. Switching between Systems by specifying IDm:

   Select a System using a command that includes IDm in the parameter of command packet. In this case, set the value of IDm you want to switch to IDm of the command packet.

3.5 Logical layer structure

In sections "3.2 Service" and "3.3 Area", descriptions are provided mainly from the standpoint of how to manage Block located in non-volatile memory space. This section, however, describes the hierarchical structure of "Area" and "Service" in the file system.

A code of 2 Bytes, known as Service Code and unique in the System, is assigned to each of the Area and Service. Only one Service Code is assigned to a Service. To an Area, on the other hand, a range of Service Code is assigned. The Service Code located at the top of the range of Service Code so assigned to an Area is known as Area Code.

For example, let one Service Code such as 12c8h be assigned to a Service at the time of registration and a range of Service Code such as 12c0h to 3fffh be assigned to an Area. For an Area, the Service Code located at the top of assigned range (i.e., 12c0h) becomes the Area Code.

The hierarchical structure of Area / Service is logically determined by the magnitude relationship of Area Code / Service Code; the Area Code / Service Code having lower values take higher levels in the logical hierarchical structure. Parent-child relationship between an Area and a Service and between an Area and another Area is determined in the following manner:

- If the Service Code of a Service is included in the range of Service Code assigned to an Area, such an Area becomes the parent Area of the Service.
- If the range of Service Code assigned to Area B is included in the range of Service Code assigned to Area A, Area A becomes the parent Area of Area B.

Figure 3-18 shows an example of magnitude relationship of Area Code / Service Code, and Figure 3-19 shows an example of how the relationship of logical hierarchical structure corresponds to the magnitude relationship of Area Code / Service Code described in this section.

As described earlier in this section, the logical hierarchical structure of Area / Service in the file system is determined by two factors, i.e., (a) the magnitude relationship of Area Code / Service Code, and (b) the number of Block assigned to the Area.

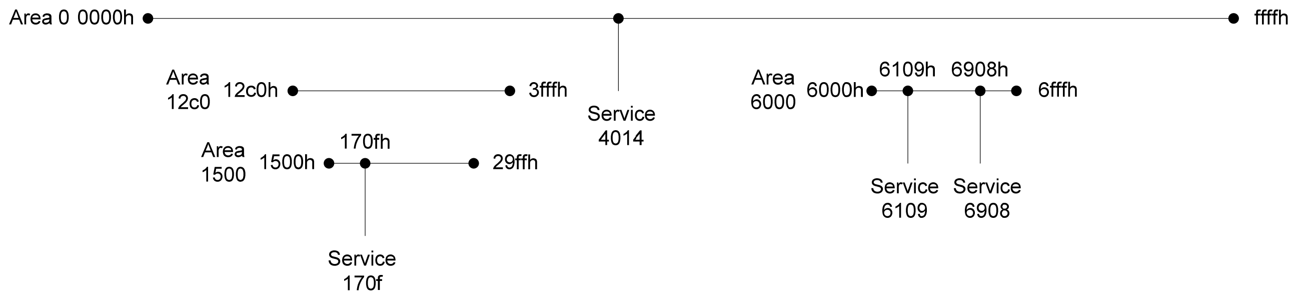


Figure 3-18: Magnitude relationship of Area / Service Code

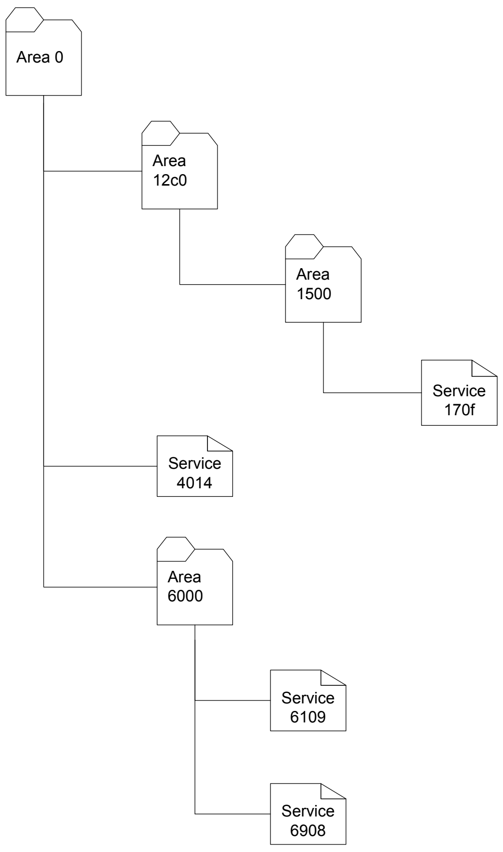


Figure 3-19: Logical hierarchical structure of file system

3.6 Protection of data

3.6.1 Data protectino function against power interruption

In FeliCa technology, it is guaranteed that the update of data located in non-volatile memory with a single command certainly results in either "totally updated" or "nothing updated". This is the function to maintain integrity of the data on non-volatile memory even if the update process was interrupted by shutting-off of the electrical power to IC. Data writing to User Block is handled as the qualified data only when writing data only when writing of all the data successfully completed. If data writing was interrupted by shutting-off of the electrical power to IC, the data being written is aborted and the data stored before such data writing is maintained.

In FeliCa technology, data writing with a single command is possible in various ways, as follows:

(1) to simultaneously write a Block Data to multiple Service, or

(2) to write multiple Block Data to a Service, or

(3) to write Block Data in a combined way of (1) and (2) in this list.

Even in such cases, this file system guarantees the synchronicity and inseparability (atomicity) of data writing. With this capability, it is possible to avoid the risk of inconsistency between data by processing fee collection and log writing with a single data write operation.

This data protection function is valid not only in writing Block Data of Service but in all the types of data writing to change the file system, such as "Area Registration", "Service Registration", and so on.

4 Command

This chapter describes the specifications of FeliCa card commands.

4.1 Acquisition / identification of card

This section describes how to acquire and identify a card (i.e., System) from the Reader/Writer.

To acquire a card from a Reader/Writer, the Reader/Writer calls (i.e., polls) an indefinite number of cards using the Polling command. To specify a desired card (i.e., System), the Reader/Writer uses a System Code described in “3.4 System” of this document.

When polling is performed with the Polling command, cards return IDm and PMm as the response to the command. After this, communication with only a specific card (i.e., System) becomes possible using the acquired IDm.

To identify the destination card of communication using IDm, see “2.3.4 Anti-collision process”. For details of the Polling command, see “4.4 Command specifications”.

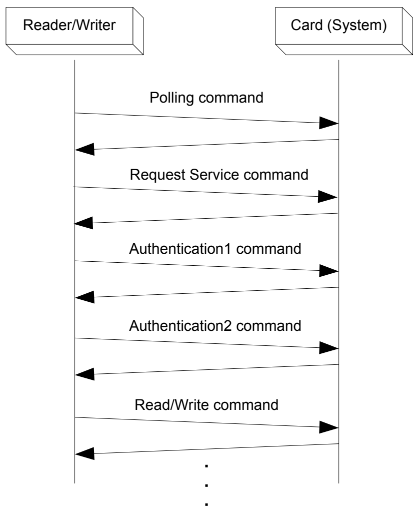
4.2 Access to Block

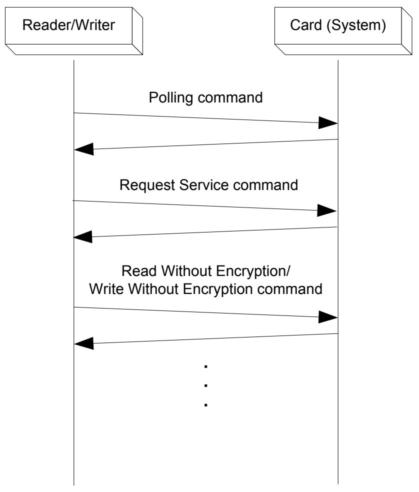
This section describes how to read Block from and write Block to FeliCa card.

To access a Block, the following commands are used: Read, Write, Read Without Encryption, Write Without Encryption.

Read / Write commands can be used for both Service that require authentication and that require no authentication at the time of access.

Read Without Encryption/Write Without Encryption commands, however, can be used only for Service that require no authentication at the time of access. To access a Service that requires authentication at the time of access, it is necessary to complete mutual authentication in advance, by using the Authentication1 command and the Authentication2 command.





Accessing a Service that requires authentication

1. Acquisition of a card (System)
   Transmit Polling command to acquire IDm as card identification information.

2. Verification of existence of the Service
   Transmit Request Service command to verify the existing of the Service, then acquire Key Version.

3. Mutual Authentication
   Transmit Authentication1 and Authentication2 to perform mutual authentication to access target Areas/Service.

4. Read and Write of block data
   Transmit Read command or Write command specifying Block List and Block Data (only for Write command) to read or write Block Data.

Accessing a Service that does not requires authentication

1. Acquisition of a card (System)
   Transmit Polling command to acquire IDm as card identification information.

2. Verification of existence of the Service
   Transmit Request Service command to verify the existing of the Service.

3. Read and Write of block data
   Transmit Read Without Encryption command or Write Without Encryption command specifying Area / Service Code List and Block List and Block Data (only for Write Without Encryption command) to read or write Block Data.

Figure 4-1: Example of command sequence

Additionally, to access a Block it is necessary to specify a Service using a Service Code and then to specify a Block using a Block Number. To perform the previously-described procedures using commands, data structures known as Area Code List / Service Code List as well as Block List are used.

4.2.1 Block List / Block List Element

To specify a Service and a Block Number to be the target of access, use a Block List. In the Block List, elements of data, each known as a Block List Element, are enumerated. Figure 4-2 to Figure 4-4 show the configurations of Block List and Block List Element.

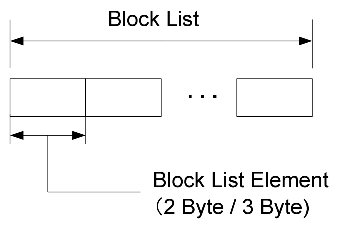


Figure 4-2: Block List

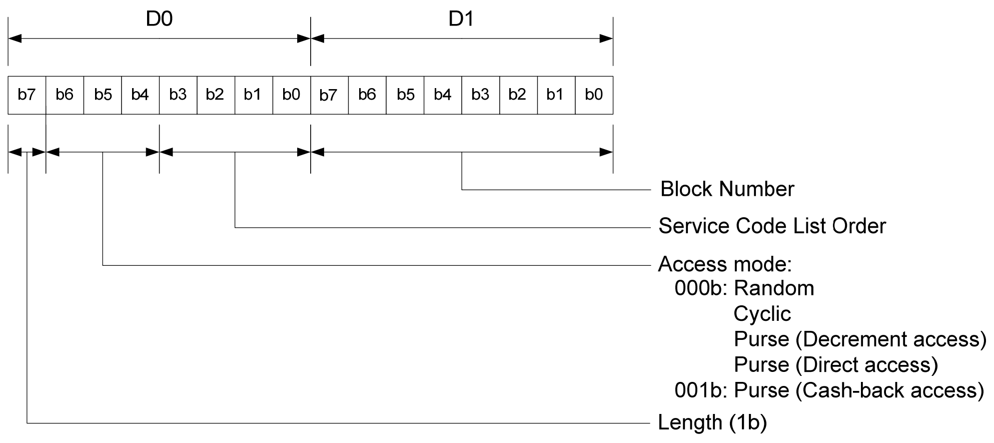


Figure 4-3: 2 Byte Block List Element

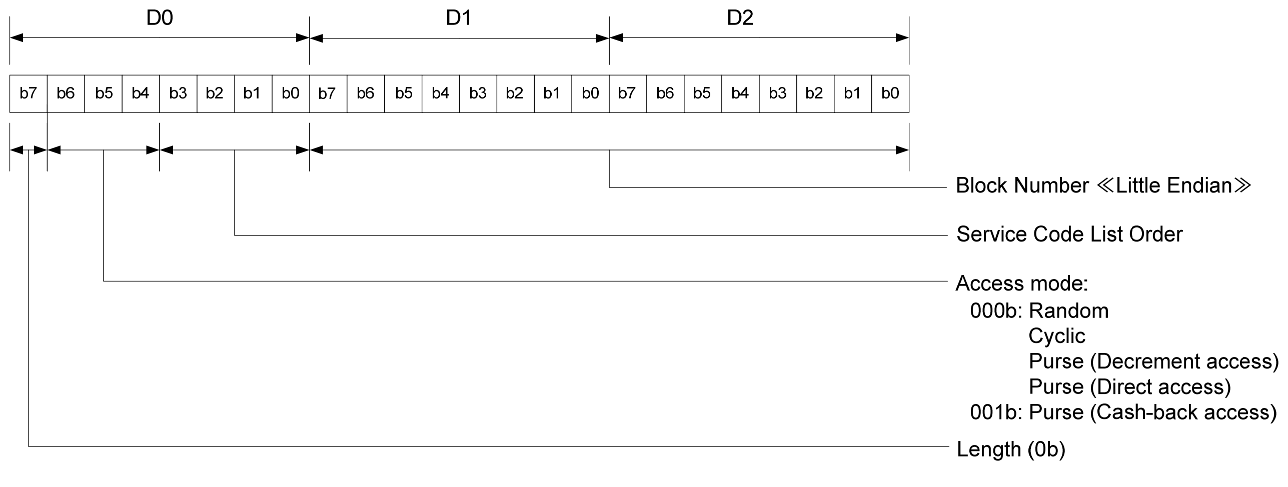

D0
D1
D2
b7 b6 b5 b4 b3 b2 b1 b0 b7 b6 b5 b4 b3 b2 b1 b0 b7 b6 b5 b4 b3 b2 b1 b0
Block Number ≪Little Endian≫
Service Code List Order
Access mode:
000b: Random
Cyclic
Purse (Decrement access)
Purse (Direct access)
001b: Purse (Cash-back access)
Length (0b)

Figure 4-4: 3 Byte Block List Element

To specify (in the Block List Element) the target Block of a data write operation, a combined description of [Block Number] and [Service Code List] is used; for example, "accesses nth Block (i.e., Block Number) of mth Service in the Service Code List". For the Read / Write command, the Service Code List referenced in the Block List Element means the Service Code List used by Authentication1 command. For the Read Without Encryption / Write Without Encryption command, the Service Code List referenced in the Block List Element means the Service Code List included in the command itself.

There are two types of Block List Element, that is, a 2-Byte Block List Element and a 3-Byte Block List Element. Co-existence of these two types of Block List Element in a Block List is possible. To specify a Block Number exceeding 256, a 3-Byte Block List Element shall be used.

The Block Data to be written to a Service, in addition, are enumerated in the parameter of the Write / Write Without Encryption command separated from the Block List. It is necessary, however, to set the order of Block List Element to be consistent with the order of corresponding Block Data. For details of how to store the Block List and the Block Data to a command packet, see “4.4 Command specifications”.

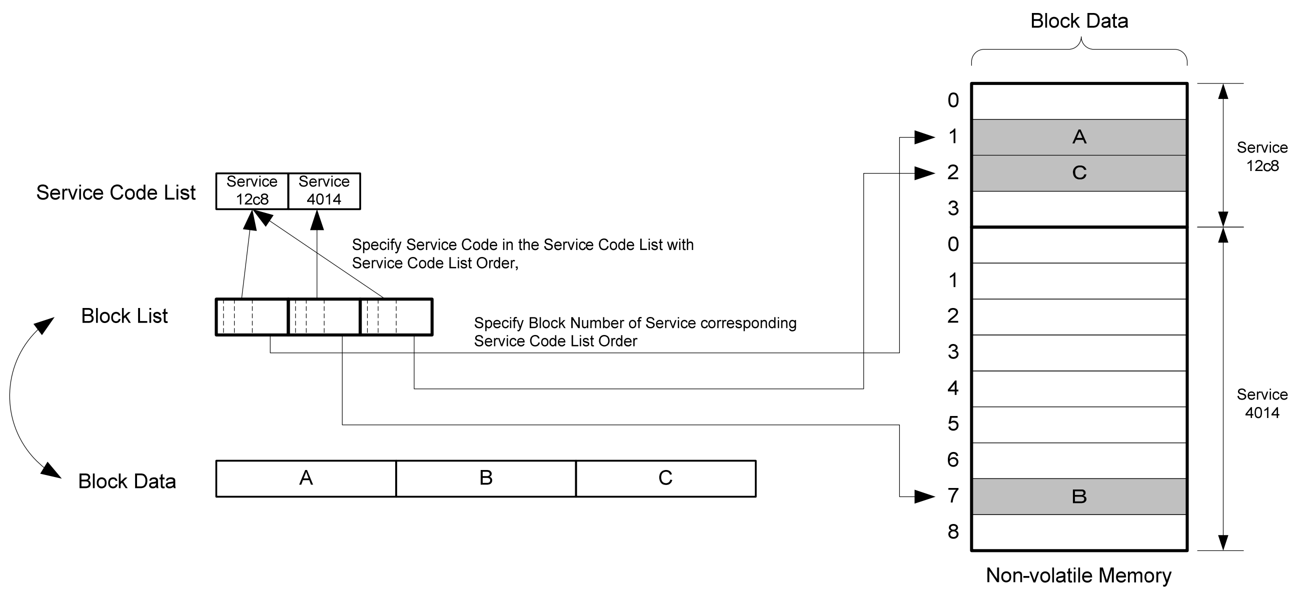

Block Data
Service Code List
Service 12c8
Service 4014
Specify Service Code in the Service Code List with Service Code List Order,
Block List
Specify Block Number of Service corresponding Service Code List Order
Block Data
A
B
C
0
1
2
3
0
1
2
3
4
5
6
7
8
A
C
B
Service 12c8
Service 4014
Non-volatile Memory

Figure 4-5: Relationship between Service Code List and Block List Element

It is necessary to set the following contents to the Block List Element with the format shown in Figure 4-3 and Figure 4-4.

Length

This specifies whether the Block List Element consists of 2 Bytes or 3 Bytes, as follows:

- 1b: this indicates that it is a Block List Element of 2 Bytes.
- 0b: this indicates that it is a Block List Element of 3 Bytes.

Access mode

In writing data to Purse Service of which access attribute is Cash-back / Aecrement Access, this specifies whether cash-back function or decrement function be used, as follows:

- 000b: Decrement function is used.
- 001b: Cash-back function is used.

  Note:

  In all cases other than Cash-back / Decrement Access of Purse Service, this is set to 000b.

Service Code List Order

This specifies the Service Code of Service to be the target of Block List Element with the order of Service Code in the Service Code List. In this case, let the order of Service located at the top of Service Code List be "0".

For example, if Service 12c8 is enumerated at the second position in Service Code List, its order in Service Code List is "1".

Block Number

This specifies that the access be performed to which Block in the Service specified by the order in Service Code List.

For details of how to set up the Block List / Block List Element, see also the setup examples shown on the following page.

4.2.2 Example of setting up a Block List

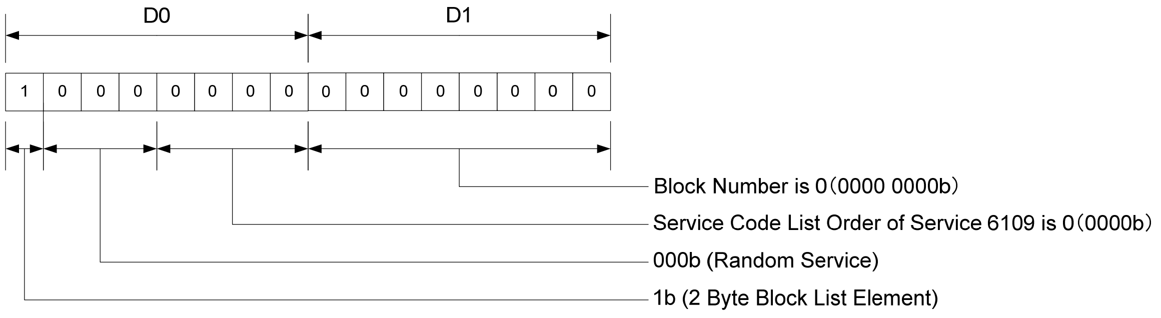
In this example, assume a simultaneous data write to Block Number 0 of Service 6109 and Block Number 5 of the same Service. The data to be written (to Block 0) is 01h x 16 and (to Block 5) is 02h x 16.

From its Service Code, it is possible to know that Service 6109 is the Service as described here:

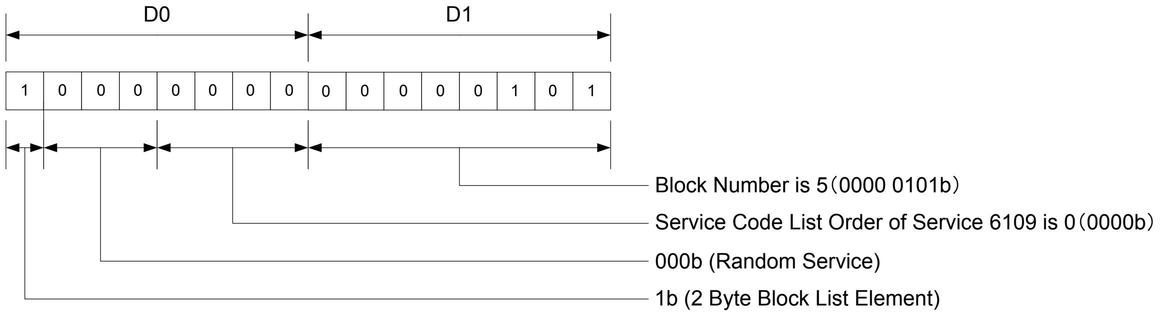
- Service 6109: Random Service with Read / Write access: no authentication is necessary.

  Service 6109 does not require authentication, so data is written to the Block by the Write Without Encryption command. The Block List Elements are as follows:

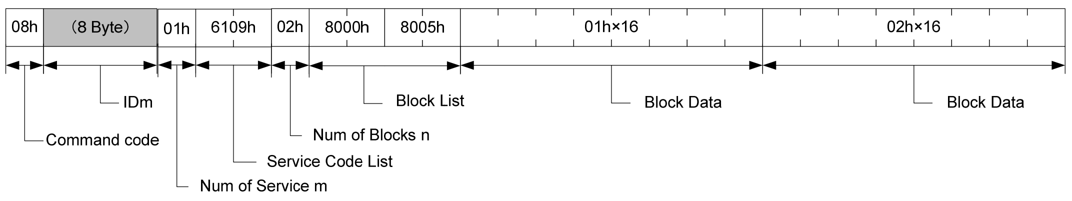
- Block Number 0 of Service 6109: 80 00h


D0
D1
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Block Number is 0（0000 0000b）
Service Code List Order of Service 6109 is 0（0000b）
000b (Random Service)
1b (2 Byte Block List Element)

- Block Number 5 of Service 6109: 80 05h


D0
D1
1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1
Block Number is 5（0000 0101b）
Service Code List Order of Service 6109 is 0（0000b）
000b (Random Service)
1b (2 Byte Block List Element)

The order of Block List Element and Block Data must be synchronized, so the packet data of the Write Without Encryption command shall become as follows:



4.3 Mode transition

In a System, there exist four modes, i.e., Mode0, Mode1, Mode2, and Mode3. The execution of commands provided by FeliCa card is restricted, depending on these modes. In addition, the transition between modes is triggered by specific commands from executable commands in each of these modes.

For example, the mode of card shall be Mode2 while executing the Read command to read data from a Service that requires authentication. To achieve this, it is necessary to perform several steps in the following order:

(1) Identify the System, by using the Polling command.

(2) To perform mutual authentication between the Reader/Writer and the card, transit the mode to Mode1, by using the Authentication1 command.

(3) Then transit the mode further to Mode2, by using the Authentication2 command.

The default mode immediately after electrical power is supplied to a card is Mode0. If the supply of electrical power is interrupted, the current mode is not maintained. The System is reset to Mode0 as soon as the electrical power is resumed.

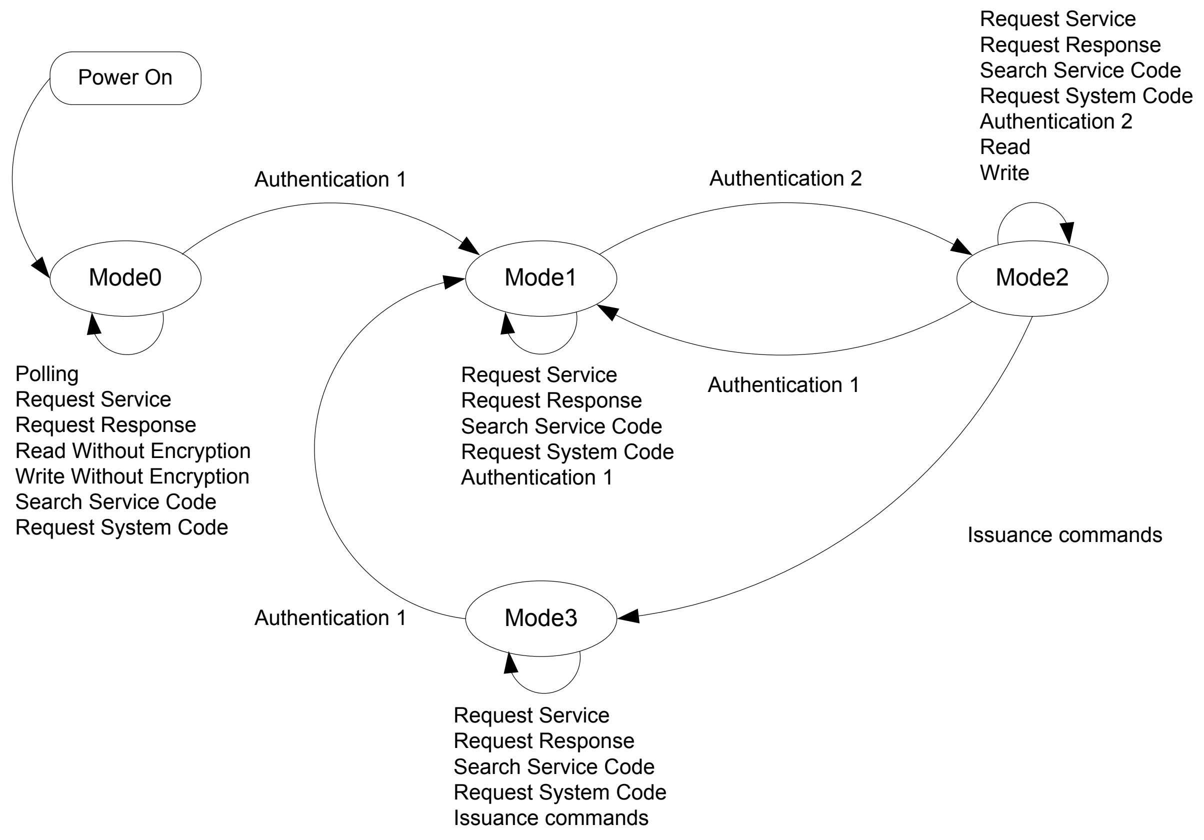


Figure 4-6: Mode transition diagram

4.4 Command specifications

4.4.1 Configuration of description

Each command interface is described in the following way:

<Summary>

Summarizes the functions of each command.

<Attribute>

Provides information about the modes executable by the command, mode transition after execution of command, encryption of packet, and switching between Systems, as follows:

- Execution condition
  Indicates whether the mode is executable by the command (i.e., either "Y", if it is, or "N", if it is not).
- Mode transition after execution of command
  Indicates the mode after execution of the command is successfully completed. If command execution fails, the mode remains unchanged.
- Encryption of packet
  Indicates whether encryption of command data or response data is performed at the time of data transmission / reception.
- Switching between Systems
  Indicates whether Switching between Systems using the command is possible.

Execution condition				**Mode transition after execution of command**	**Encryption of packet**	**Switching between Systems**
Mode0	**Mode1**	**Mode2**	**Mode3**			

<Requirement in returning response>

Describes the conditions under which a card should return some type of response to a command transmitted from the Reader/Writer. If the conditions are not satisfied, the card returns no response.

<Requirement for successful completion of command execution>

Describes the conditions required for the successful completion of command execution. Only when all the requirements enumerated here are satisfied, does the command become successfully completed.

<Packet structure>

COMMAND PACKET DATA

Describes the structure, parameter name, size, data, description, and other details (i.e., notes) of the command packet data at the time of command transmission (unit of size is represented in Bytes).

In command packet data, there exist parameters (such as Area Code, Service Code, and so on) for which Endian must be considered. For example, if notation ‹‹Little Endian›› exists in the "Note" column, the data must be set in Little Endian format.

RESPONSE PACKET DATA

Specifies the structure, parameter name, size, data, description, and other details (i.e., notes) of the packet data at the time the response is returned (unit of size is represented in Bytes).

In response packet data, there exist parameters (such as Area Code, Service Code, and so on) for which Endian must be considered. For example, if notation ‹‹Little Endian›› exists in the "Note" column, the data must be set in Little Endian format.

STATUS FLAG

For any command having Status Flag in its response packet data, describes the value of the Status Flag and the nature of the error associated with that flag.

<Special instruction>

Describes detailed information about the command, such as important notes to consider before using the command.

4.4.2 Polling command

<Summary>

- Use this command to acquire and identify a card.
- Acquisition of Manufacture ID (IDm) and Manufacture Parameter (PMm) is possible with this command.
- By specifying a Request Code, you can acquire System Code or communication performance of the System.
- By specifying a Time Slot, you can designate the maximum number of time slots possible to return responses (see "<Special instruction>").

<Attribute>

Execution condition				Mode transition after execution of command	Encryption of packet	Switching between Systems
Mode0	Mode1	Mode2	Mode3			
Y	N	N	N	N	N	Y

<Requirement in returning response>

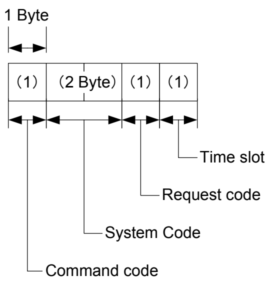
- Mode shall be Mode0.
- The data length of the received packet shall be the correct data length for the Polling command.
- The System specified by System Code shall exist in the card.

<Requirement for successful completion of command execution>

All the requirements for returning a response shall be satisfied.

<Packet structure>

COMMAND PACKET DATA

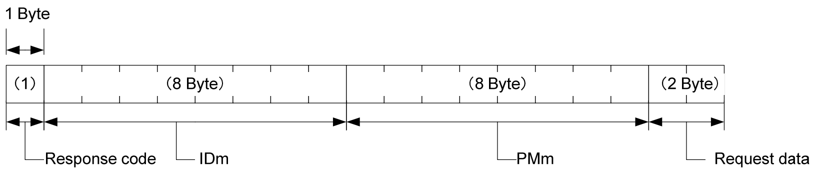


Parameter name	Size	Data	Description	Note
Command code	1	00h	Command code of Polling command	
System Code	2		System Code of card to be acquired	You can specify a wildcard. For details, see "<Special instruction>".
Request code	1		Designation of Request data. 00h: No request 01h: System Code request 02h: Communication performance request other: RFU	

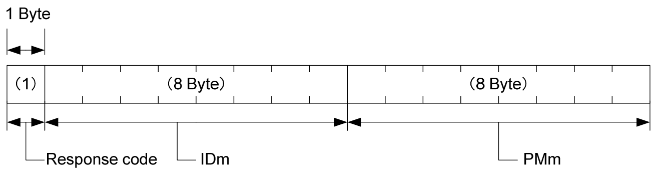
Parameter name	Size	Data	Description	Note
Time Slot	1		Designation of maximum number of slots possible to respond. (For details, see Table 4-1)	

RESPONSE PACKET DATA

- The case when request data is returned:



- The case when request data is not returned:



Parameter name	Size	Data	Description	Note
Response code	1	01h	Response code to Polling command	
IDm	8		Manufacture ID of acquired System	
PMm	8		Manufacture Parameter of acquired System	
Request data	0 or 2		Data that corresponds to the Request code (See Table 4-2).	Data is returned only when the request code of command packet is other than 00h, and corresponds to the request data.

STATUS FLAG

None

Table 4-1: Time slot specifications

Time slot	Maximum number of slots	Time slot possible to respond
00h	1	#0
01h	2	#0, #1
03h	4	#0, #1, #2, #3
07h	8	#0, #1, #2, #3, #4, #5, #6, #7
0fh	16	#0, #1, #2, #3, #4, #5, #6, #7, #8, #9, #10, #11, #12, #13, #14, #15

Table 4-2: Returned value to request data

Request code	Request data	Note
00h: No request	None	Request data is not returned.

Request code	Request data	Note
01h: System Code request	System Code	System Code of acquired System is returned. No request data is returned from the card that does not support request for System Code (the card operates in a manner as if 00h was specified).
02h: Requests communication performance	Communication performance	Communication performance is returned. See Table 4-3. For a card that does not support request for communication performance, no request data is returned (the card operates in a manner as if 00h was specified).

Table 4-3: Communication performance

D0	D1								Description
	b7	b6	b5	b4	b3	b2	b1	b0	
00h (other values are reserved)	-	-	-	-	-	-	-	x	0b: 212kbps communication is impossible. 1b: 212kbps communication is possible.
	-	-	-	-	-	-	x	-	0b: 424kbps communication is impossible. 1b: 424kbps communication is possible.
	-	-	-	-	-	0	-	-	0b: 848kbps communication is impossible. 1b: 848kbps communication is possible (reserved)
	-	-	-	-	0	-	-	-	0b: 1.6Mbps communication is impossible. 1b: 1.6Mbps communication is possible (reserved)
	-	0	0	0	-	-	-	-	Fixed value (other values are reserved)
	x	-	-	-	-	-	-	-	0b: communication rate automatic detection noncompliant. 1b: communication rate automatic detection compliant.

<Special instruction>

About specifying a wildcard for System Code:

- For System Code, you can specify a wildcard (ffh) for either the upper or lower 1 Byte, or for both the upper and lower Bytes. The Byte for which the wildcard is specified is regarded as an arbitrary value in the process of comparison with the System Code of System existing in the card. For example, if System Code of System0 was 0123h, the card returns a response when the System Code of the Polling command is 0123h (full matching), ff23h (the upper 1 Byte is a wildcard), 01ffh (the lower 1 Byte is a wildcard), or ffffh (both 2 Bytes are wildcards).
- If a card is divided into multiple Systems, comparison of System Code is done with System0 first, and after that, comparison of System Code is performed sequentially to Systems that follow System1. Therefore, if a wildcard is specified to both 2 Bytes of System Code, System0 always returns a response.
- By specifying a wildcard for both 2 Bytes (i.e., ffffh), you can make a card return a response to the Polling command, regardless of its System Code.

Specifying time slot

- Designation of the time slot of the Polling command may be selected from (00h / 01h / 03h / 07h / 0fh). In this case, the number of responses allowed for a card are (1 / 2 / 4 / 8 / 16), respectively. If 00h is designated, for example, the timing at which the card returns a response is only once. Therefore, all the cards can return a response start simultaneous returning of response. On the other hand, if 0fh is designated, sixteen timings are available for cards to return a response. It is expected that the probability of random timing selection increases and the probability of conflict decreases.

4.4.3 Request Service command

<Summary>

- Use this command to check for the existence of Area / Service specified by Area Code / Service Code.
- If the specified Area / Service exists, the card returns version information of the key known as "Key Version" (2 Bytes).
- If the specified Area / Service does not exist, the card returns ffffh as its Key Version.

<Attribute>

Execution condition				Mode transition after execution of command	Encryption of packet	Switching between Systems
Mode0	Mode1	Mode2	Mode3			
Y	Y	Y	Y	N	N	Y

<Requirement in returning response>

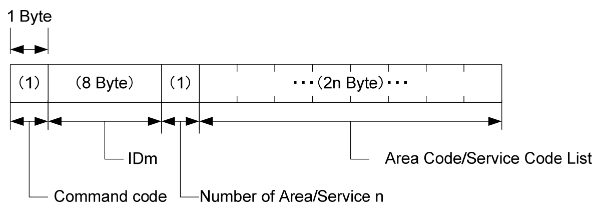
- The data length of received packet shall be the correct data length of the Polling command.
- The value of number of Area / Service of command packet data shall be within the specified range.

<Requirement for successful completion of command execution>

All the requirements for returning a response shall be satisfied.

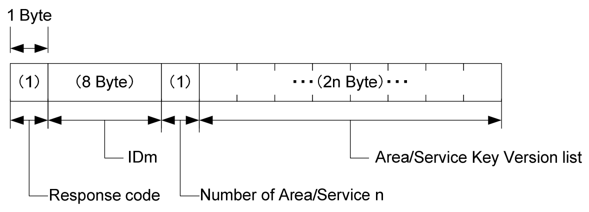
<Packet structure>

COMMAND PACKET DATA



Parameter name	Size	Data	Meaning	Note
Command code	1	02h	Command code of Request Service	
IDm	8		Manufacture ID of the System to be the target of the Request Service command	
Number of Area / Service	1	n	Number of Area / Service to be the target of acquisition of Key Version	1 ≤ n ≤ 32
Area Code ／Service Code List	2n		List of Area Code / Service Code to be the target of acquisition of Key Version	‹‹Little Endian›› See "<Special instruction>".

RESPONSE PACKET DATA



Parameter name	Size	Data	Description	Note
Response code	1	03h	Response code for the Request Service command	
IDm	8		Manufacture ID of System to be the target of the Request Service command	
Number of Area / Service	1	n	Number of Area / Service to be the target of acquisition of Key Version	1 ≤ n ≤ 32
Area / Service Key Version list	2n		List of acquired Key Versions	‹‹Little Endian›› See "<Special instruction>".

STATUS FLAG

None

<Special instruction>

- For Area Code / Service Code of a command packet, Area Code or Service Code of target of acquisition of Key Version are enumerated in Little Endian format. If System Key is the target of acquisition, specify ffffh in the command packet.
- The order of Key Versions in Area / Service Key Version list matches the order in Area Code / Service Code List.
- If the Area / Service specified by Area Code / Service Code in the command packet does not exist in the card, ffffh is returned at the corresponding location in the Area / Service Key Version list.

4.4.4 Request Response command

<Summary>

- Use this command to check whether a card exists.
- Current mode of the card is returned.

<Attribute>

Execution condition				Mode transition after execution of command	Encryption of packet	Switching between Systems
Mode0	Mode1	Mode2	Mode3			
Y	Y	Y	Y	No	No	Yes

<Requirement in returning response>

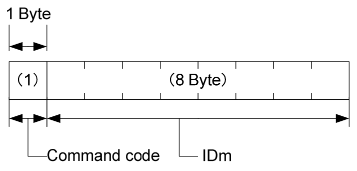
The data length of received packet shall be the correct data length of the Request Response command.

<Requirement for successful completion of command execution>

All the requirements for returning a response shall be satisfied.

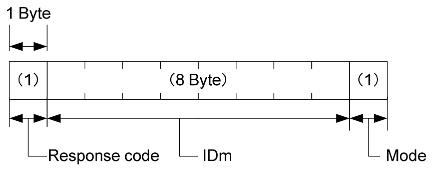
<Packet structure>

COMMAND PACKET DATA



Parameter name	Size	Data	Description	Note
Command code	1	04h	Command code for the Request Response command	
IDm	8		Manufacture ID of System to be the target of the Request Response command	

RESPONSE PACKET DATA



Parameter name	Size	Data	Description	Note
Response code	1	05h	Response code to the Request Response command	
IDm	8		Manufacture ID of System to be the target of the Request Response command	

Parameter name	Size	Data	Description	Note
Mode	1		Current mode of System 00h: Mode0 01h: Mode1 02h: Mode2 03h: Mode3	

STATUS FLAG

None

<Special instruction>

None

4.4.5 Read Without Encryption command

<Summary>

Use this command to read Block Data from a Service that requires no authentication.

<Attribute>

Execution condition				Mode transition after execution of command	Encryption of packet	Switching between Systems
Mode0	Mode1	Mode2	Mode3			
Y	N	N	N	N	N	N

<Requirement in returning response>

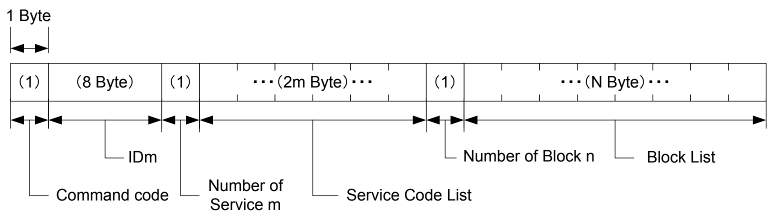
- The mode shall be Mode0.
- The data length of the received packet shall be the correct data length of the Read Without Encryption command.

<Requirement for successful completion of command execution>

- Number of Service shall be equal to or greater than "1" and equal to or less than "16".
- The number of Block shall be less than the maximum number of Block that can be read simultaneously.
- Each Block List Element shall satisfy the following conditions:
  - o The value of the order in the Service Code List does not exceed the number of Service.
  - o The access mode is 000b.
  - o The target specified by the Service Code is not an Area or a System.
  - o The Service specified in the Service Code List does exist in the System.
  - o The access attribute of the Service specified in the Service Code List is "no authentication required".
  - o The Block Number falls in the range of number of Block set to the specified Service.

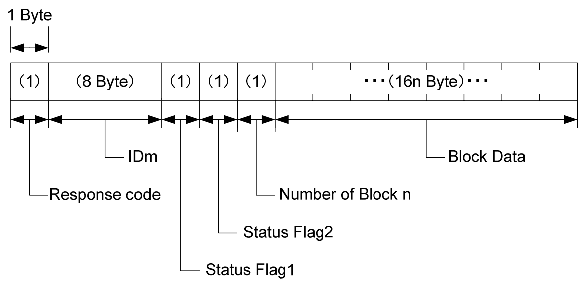
<Packet structure>

COMMAND PACKET DATA



Parameter name	Size	Data	Description	Note
Command code	1	06h	Command code of the Read Without Encryption command	
IDm	8		Manufacture ID of the System to be the target of the Read Without Encryption command	
Number of Service	1	m	Number of Service included in the Service Code List	1 ≤ m ≤ 16
Service Code List	2m		Service Code List	‹‹Little Endian››
Number of Block	1	n	Number of Block that can be read simultaneously	See "<Special instruction>".
Block List	N		List of Block List Element entries	2n ≤ N ≤ 3n

RESPONSE PACKET DATA



Parameter name	Size	Data	Description	Note
Response code	1	07h	Response code to the Read Without Encryption command	
IDm	8		Manufacture ID of the System to be the target of the Read Without Encryption command	
Status Flag1	1		See Status Flag.	
Status Flag2	1		See Status Flag.	
Number of Block	1	n	Number of Block read by the command	Provided only if Status Flag1 = 00h.
Block Data	16n		List of Block read by the command	Provided only if Status Flag1 = 00h.

STATUS FLAG

See “4.5 Status Flag”.

<Special instruction>

The maximum number of Block that can be read simultaneously differs, depending on the product.

4.4.6 Write Without Encryption command

<Summary>

Use this command to write Block Data to a Service that requires no authentication.

<Attribute>

Execution condition				Mode transition after execution of command	Encryption of packet	Switching between Systems
Mode0	Mode1	Mode2	Mode3			
Y	N	N	N	N	N	Y

<Requirement in returning response>

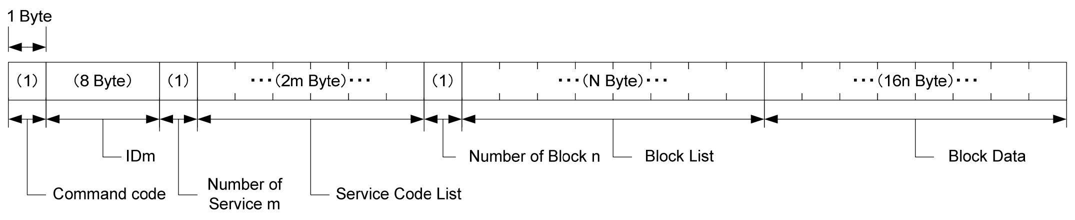
- The mode shall be Mode0.
- The data length of the received packet shall be the correct data length of the Write Without Encryption command.

<Requirement for successful completion of command execution>

- The number of Service shall be equal to or greater than "1" and equal to or less than "16".
- The number of Block shall be less than the maximum number of Block that can be written simultaneously.
- Each Block List Element shall satisfy the following conditions:
  - The value of the order in Service Code List does not exceed the number of Service.
  - The access mode is either 000b or 001b.
  - If 001b is specified to the access mode, the access attribute of the specified Service is "cash-back / decrement access without authentication" of Purse Service.
  - The target specified by Service Code List is not an Area or a System.
  - The Service specified in Service Code List exists in the System.
  - The access attribute of the Service specified in the Service Code List is not "read only".
  - The access attribute of the Service specified in Service Code List is "no authentication required".
  - The Block Number falls in the range of number of Block set to the specified Service.
  - If the specified Service is Cyclic Service, the following conditions are satisfied:
    - Block Number is "0".
    - To write data sequentially to the same Cyclic Service, the number of sequential data writes is within the range of number of Block set to the specified Cyclic Service.
  - If the specified Service is Purse Service, the following conditions are satisfied:
    - Purse data of the command packet data is less than the purse data of the specified purse Block Data.
    - Cash-back data of the command packet data is less than the cash-back data of the specified purse Block Data.
    - The value calculated by adding the cash-back data of the command packet data to the purse data of the specified purse Block Data does not exceed (ff ff ff ff)h.

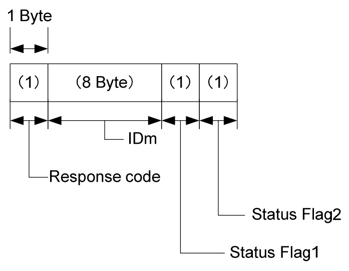
<Packet structure>

COMMAND PACKET DATA



Parameter name	Size	Data	Description	Note
Command code	1	08h	Command code of Write Without Encryption command	
IDm	8		Manufacture ID of the System to be the target of Write Without Encryption command	
Number of Service	1	m	Number of Service included in Service Code List	1 ≤ m ≤ 16
Service Code List	2m		Service Code List	‹‹Little Endian››
Number of Block	1	n	Number of Block to which data is written	See "<Special instruction>".
Block List	N		List of Block List Element entries	2n ≤ N ≤ 3n
Block Data	16n		List of Block Data	

RESPONSE PACKET DATA



Parameter name	Size	Data	Description	Note
Response code	1	09h	Response code to the Write Without Encryption command	
IDm	8		Manufacture ID of the System to be the target of the Write Without Encryption command	
Status Flag1	1		See Status Flag.	
Status Flag2	1		See Status Flag.	

STATUS FLAG

See “4.5 Status Flag”.

<Special instruction>

The maximum number of Block that can be written simultaneously differs, depending on the product.

4.4.7 Request System Code command

<Summary>

- Use this command to acquire System Code of the System located on a card.
- If a card is divided into multiple System, this command acquires System Code of all the System existing in the card.

<Attribute>

Execution condition				Mode transition after execution of command	Encryption of packet	Switching between Systems
Mode0	Mode1	Mode2	Mode3			
Y	Y	Y	Y	N	N	Y

<Requirement in returning response>

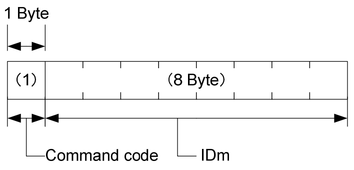
The data length of the received packet shall be the correct data length of the Request System Code command.

<Requirement for successful completion of command execution>

All the requirements in the returning response shall be satisfied.

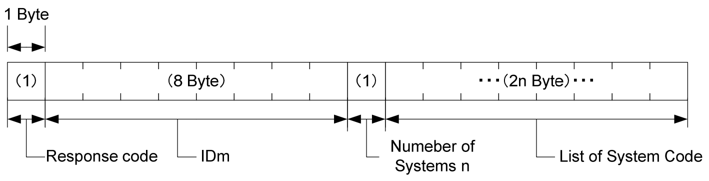
<Packet structure>

COMMAND PACKET DATA



Parameter name	Size	Data	Description	Note
Command code	1	0ch	Command code of the Request System Code command	
IDm	8		Manufacture ID of the System to be the target of the Request System Code command	

RESPONSE PACKET DATA



Parameter name	Size	Data	Description	Note
Response code	1	0dh	Response code to the Request System Code command	
IDm	8		Manufacture ID of System to be the target of the Request System Code command	

Parameter name	Size	Data	Description	Note
Number of System Code	1	n	Number of System existing in the card	
List of System Code	2n		List of System Code of the System existing in the card	System Codes are enumerated in ascending order starting from "0".

STATUS FLAG

None

<Special instruction>

None

4.4.8 Authentication1 command

<Summary>

Use this command to authenticate cards.

For details of this command, see the document to be disclosed in accordance with the separate agreement.

4.4.9 Authentication2 command

<Summary>

Use this command to allow a card to authenticate a Reader/Writer.

For details of this command, see the document to be disclosed in accordance with the separate agreement.

4.4.10 Read command

<Summary>

Use this command to read Block Data from a Service that requires authentication.

For details of this command, see the document to be disclosed in accordance with the separate agreement.

4.4.11 Write command

<Summary>

Use this command to write Block Data to a Service that requires authentication.

For details of this command, see the document to be disclosed in accordance with the separate agreement.

4.5 Status Flag

A Status Flag consists of Status Flag1 (1 Byte) and Status Flag2 (1 Byte).

Status Flag1 indicates the success or failure of processing in the card, as well as the location of Block or of Service where an error occurred.

Status Flag2 indicates the detailed content of Status Flag1.

4.5.1 Status Flag1

Status Flag1 indicates the success or failure of processing in the card, as well as the location of Block where an error occurred.

00h:	Process successfully completed.
ffh:	Error or warning independent from the Block List or the Service list (for detailed content, see Status Flag2).
xxh:	Indicates the Block Number where an error occurred (for detailed content, see Status Flag2).

Note:

Two types of error indication method are used, depending on the product.

[Location of Block, or Service, or setup value string specified in command packet is indicated]

When normal command processing failed and an error is returned in the response packet, the location of the Block, Service, or setup value string is written to Status Flag1 and returned to the Reader/Writer.

[Location of error is indicated by bit data]

When viewed on a bit-by-bit basis, each bit of Status Flag1 corresponds to the location in either the Block List or the Service list:

Bit0:	the 1st or the 9th location in Block List or Service list
Bit1:	the 2nd or the 10th location in Block List or Service list
Bit2:	the 3rd or the 11th location in Block List or Service list
Bit3:	the 4th or the 12th location in Block List or Service list
Bit4:	the 5th location in Block List or Service list
Bit5:	the 6th location in Block List or Service list
Bit6:	the 7th location in Block List or Service list
Bit7:	the 8th location in Block List or Service list

In this case, value of each bit signifies the following results:

0:	no error existed.
1:	an error existed.

In addition, Block / Service are checked in an ascending order of Block List / Service list. The first Block in which an error was detected during this checking is determined as the location of the error occurrence. Status Flag2 indicates the error information for that Block. Further error-checking is not performed for the other Block.

(Example) when Status Flag1 = 04h;

1st location of Block List or Service list:	no error exists.
2nd location of Block List or Service list:	no error exists.
3rd location of Block List or Service list:	an error exists. (Detailed information is stored to Status Flag2)
4th location of Block List or Service list:	not checked.
5th location of Block List or Service list:	not checked.
6th location of Block List or Service list:	not checked.
7th location of Block List or Service list:	not checked.
8th location of Block List or Service list:	not checked.

4.5.2 Status Flag2

Status Flag2 indicates the detailed content of Status Flag1. The content of this flag is divided into two major classes: i.e., the common specifications and the card-specific specifications. System designers are requested to use information based on the common specifications only, and not to incorporate information based on the card-specific specifications. The card-specific specifications are intended only for the purpose of debugging.

Common specifications (01h-7fh)

00h:	Normal completion of process
01h:	Result of calculation is less than zero at decrement of purse, or result of calculation exceeds 4 Bytes.
02h:	Data of the specified purse exceeds the value of cash-back data.
70h:	Memory error (fatal error).
71h:	Number of memory rewrites exceeds the upper limit (this is only a warning; data writing is performed as normal). The maximum number of rewrites differs depending on the product. In addition, Status Flag1 of some products is 00h or ffh.

Card-specific specifications (80h-ffh)

Use this status code to verify the application.

Only major error statuses are enumerated here. Conditions of error occurrence differ somewhat depending on the type of card. Therefore, it is not recommended to use card-specific details to determine error occurrence during card operations.

Use the card-specific specifications only to debug the application.

a1h:	Illegal Service number
a2h:	Illegal command packet (specified number of Block)
a3h:	Illegal Block List (specified order of Service)
a4h:	Illegal Service type
a5h:	Access is not allowed

a6h:	Illegal Service Code List
a7h:	Illegal Block List (access mode)
a8h:	Illegal Block Number (access to the specified data is inhibited)
a9h:	Data could not be written
aah:	Key could not be changed
abh:	Illegal parity
ach:	Illegal parameter
adh:	Service exists already
aeh:	Illegal System Code
afh:	Too many simultaneous cyclic write operations

5 Security

For security specifications, see the document to be disclosed based on the optional agreement.

6 Inspection

For inspection specifications, see the document to be disclosed based on the optional agreement.

7 Issuance

For issuance specifications, see the document to be disclosed based on the optional agreement.

8 Glossary

<A>

Access Attribute	The lowest 6 bits of each Area Code and Service Code. This value determines either (for Area Code) whether a Sub-Area can be defined or (for Service Code) how to access Block Data.
Access Mode	A value specified in the Block List Element. This value identifies the method of access to use when accessing Block Data.
Area	The concept of hierarchical management of Block Data. Area can contain Service and Sub-Area.
Area 0	The Area located at the highest hierarchical level of System. Each System can have only one Area 0.
Area Code	The value that uniquely identifies Area.

<B>

Big Endian	To record or transfer numerical data longer than 2 Bytes, such data is divided on a Byte-by-Byte basis. The method to sequentially record or transfer such data from the highest (i.e., most significant) Byte first is known as "Big Endian".
Block Data	1. Data to be written to or read from a Block. 2. Data to be stored to a Block.
Block List	The enumeration (i.e., the ordered array) of all Block List Element instances.
Block List Element	Data that specifies Service and Block Number to be the target of access.
Block Number	A value specified in the Block List Element. This value identifies the logical location of Block Data.

<C>

Cyclic Service	The Service that manages the deletion of the oldest set of data when new data is written (assuming that logs are in use).

<D>

Decrement Access	The method of access to decrement the specified value from purse data in Purse Service.
Direct Access	The method of access to overwrite the specified Block Data directly in Purse Service.

<I>

IC Code	The 2-Byte code that uniquely identifies each type of integrated chip (IC). IC Code comprises ROM Type (1 Byte) and IC Type (1 Byte).
IC Type	The 1-Byte code that uniquely identifies each hardware type.

<K>

Key Version	The value that identifies each key version.

<L>

Little Endian	To record or transfer numerical data longer than 2 Bytes, such data is divided on a Byte-by-Byte basis. The method to sequentially record or transfer such data from the lowest (i.e., least significant) byte first is known as "Little Endian".

<M>

Manufacture ID (IDm)	The value that comprises Manufacturer Code and card identification number. The Reader/Writer uses this value to identify each card with which to communicate.
Manufacture Parameter (PMm)	Card-specific information that is set by the card manufacturer.
Manufacturer Code	The upper 2 Bytes of Manufacture ID (IDm). This value identifies the manufacturer that assigned Manufacture ID (IDm) to the card.

<N>

No Response	The operation that halts communications without sending a response to the received command.

<O>

Overlap	The operation that enables multiple Service to share the same Block Data.
Overlap Service	A Service that shares the same Block Data with another service.

<P>

Purse Service	The Service that allows decrement operations where the stored data is regarded as a numerical value.

<R>

Random Service	The Service that enables read/write operations by specifying a Block.
ROM Type	The 1-Byte code that uniquely identifies the software (ROM) type of the same IC Type.

<S>

Service	The concept that identifies both the method of access to the Block Data and a set of Block Data.
Service Code	The value that uniquely identifies each Service.
Service Code List Order	A value specified in the Block List Element. This value specifies the target Service to access using an enumeration from the Service Code List.

Status Flag	The information that indicates the error status of a card, consisting of Status Flag1 and Status Flag2.
System	The logically-formatted domain that contains the FeliCa file management structure.
System Code	The value that uniquely identifies each System. System Code is assigned per service provider and per application.
System Number	The number that identifies each System located on a card.
System Separation	The operation that both logically divides the memory located on a card and creates multiple logical card functions (i.e., more than one System) on that card.

(Blank page)

Technical Information

FeliCa Card User's Manual Version 1.0

July 2010 First Edition FeliCa Business Division

Sony Corporation

No. M617-E01-00