SII

RP-E10 Series
Communication Library
Application Programmer's Guide

U00127039409

Seiko Instruments Inc.

Introduction

This document describes the Communication library (hereinafter referred to as "Communication library") running on the printer driver (hereinafter referred to as "printer driver") for the RP-E10 series provided by Seiko Instruments Inc.

Symbols

This section describes symbols used in this document.

> ### Caution
> ◆ Notes and limitations are described.

Target Printer Drivers

The following printer driver is supported by the Communication library.

- RP-E10 series printer driver

Terms

This section describes terms used in this document.

Definition	Description
Technical reference	RP-E10 SERIES THERMAL PRINTER TECHNICAL REFERENCE.
ASB Setting command (ASB: Automatic Status Back)	Printer command "Automatic Status Back Enable/Disable". For details, refer to the Technical reference of the printer.
POS printer status	POS printer's status information retrievable by the Communication library. This information includes the status to respond for the printer command "Automatic Status Back Enable/Disable" and some extended statuses. For details, see "6.1 POS Printer Status List".
Printer driver user's guide	RP-E10 Series Printer Driver User's Guide.

Disclaimer

Seiko Instruments Inc. shall not be liable for any damages that may occur either directly or indirectly from use of this product.

Seiko Instruments Inc. shall not be liable for any damages or losses caused by or related to improper use of this product, improper handling without adherence to this document, repair or modification from third-party other than our personnel, and so forth.

Chapter 7   Command Definition File   7-1

Chapter 1   Overview

1.1 Introduction

This chapter describes the overview of the Communication library.

The Communication library is a dynamic library to directly control printers for developers.

The Communication library is provided with the printer driver, and uses the printer driver to work. The Communication library must be installed separately from the printer driver.

You can use the Communication library to directly control printers in an application development and design the application independent of the port type.

Also, it is possible to retrieve or change values of some private DEVMODE setting items.

See sample programs for each language which are provided as usage examples of the Communication library.

1.2 Operating Conditions

It basically follows the operating environment of the printer driver, and the use conditions and limitations of the memory switch.

Refer to the Printer driver user's guide.

In addition, the following operating conditions must be met.

- Installation of .NET Framework Version 2.0 or later.
- For use in the serial communication, [Flow Control] must be "Hardware".
  (See the Technical reference for details.)
- The bidirectional support function must be enabled.
  (See the Printer driver user's guide for details about how to set it.)
- The printer pool function must be disabled.
  (See the Printer driver user's guide for details about how to set it.)
- The printer must not be disabled in the printer command "Peripheral Equipment Selection".
  (See the Technical reference for details.)
- If both of the following are applicable, use a shared printer.
  (If a shared printer is not used, it may cause the data sent from other hosts to interrupt the printer.)
  1. When using a single printer from multiple hosts through the TCP/IP connection.
  2. When the data to send is divided more than once to send.

Chapter 2   Installation

2.1 Installation Methods

The Communication library is installed with the printer driver. Refer to the Printer driver user's guide for installation method.

Chapter 3   Win32 API

3.1 Overview

This chapter describes the Communication library for Win32 development environment (Win32 API).

3.2 Development Language

The following development language is covered.

- Visual C++

3.3 Library File

The Communication library has the following file name.

- SiiRpe1Api.dll

The Communication library file is stored in the Windows system folder.

Use the Communication library without moving it from the folder. In this case, you do not have to set a path to the folder containing the Communication library.

When the Communication library file is moved to another location, the Communication library could not be updated properly during version up of the printer driver.

3.4 API List

The following APIs are implemented in the Communication library.

API	Brief Description of the Function
RpOpenMonPrinterA[*1] RpOpenMonPrinterW[*1]	Starts using the Communication library in the specified printer and returns the API handle.
RpCloseMonPrinter	Ends using the Communication library with the specified API handle.
RpLockPrinter[*2]	Locks all data transmission and hardware reset requests from other processes to the printer.
RpUnlockPrinter	Unlocks the access prohibition (lock) from other processes by RpLockPrinter.
RpDirectIO[*2,*3]	Sends/Receives binary data. (Receive data does not include responses of the ASB Setting command)
RpDirectIOEx[*2,*3]	Sends/Receives binary data.
RpResetPrinter[*2,*3]	Resets the printer.
RpGetStatus	Retrieves the latest POS printer status.
RpSetStatusBackFunction	Registers the callback function to be called when a change of the POS printer status is detected.
RpSetStatusBackWnd	Registers the window handle of a button for which the click event is called when a change of the POS printer status is detected, and the variable to set the POS printer status.
RpCancelStatusBack	Unregisters the callback function which was executed in RpSetStatusBackWnd and RpSetStatusBackFunction.
RpPowerOff[*2,*3]	Turns the printer power off.
RpGetCounter[*2,*3]	Retrieves the specified maintenance counter.
RpResetCounter[*2,*3]	Initializes the specified maintenance counter.
RpGetType[*2,*3]	Retrieves the type ID and font type of the printer.
RpGetPrnCapability[*2,*3]	Retrieves the specified printer information.
RpOpenDrawer[*2,*3]	Opens the specified drawer.
RpSendDataFileA[*1] RpSendDataFileW[*1]	Registers the command using the specified command definition file.
RpDirectSendReadA[*1,*2,*3] RpDirectSendReadW[*1,*2,*3]	Executes a command registered by RpSendDataFile.
RpGetProperty	Retrieves the specified property ID.
RpSetProperty	Changes the specified property ID.

*1: Specify arguments of strings by MBCS (MultiByte Character Set) or Unicode. Use API added the suffix 'A' for MBCS or 'W' for Unicode. Note that a suffix of 'A' or 'W' is omitted in the following descriptions.
*2: When RpLockPrinter was called from another process, this API fails.
*3: When there is a print job in the spooler, or any disconnection or communication failure with the printer occurs, this API fails.

3.5 API Details

◢ Caution ◣

◆ For Bluetooth connection, when the connection is once disconnected, part of response data may not be retrieved.

◆ For Bluetooth connection, response data of the disconnected printer cannot be retrieved.

RpOpenMonPrinter

Starts using the Communication library in the specified printer and returns the API handle.

INT RpOpenMonPrinter(

INT i_type,

LPCTSTR i_prt)

Parameters

i_type

Open type

2 (fixed)

i_prt

Name of the printer that uses the Communication library

Specifies the printer name (friendly name).

Return value

Returns the API handle to identify the printer for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- The number that printers open per process is up to eight.
- When the API handle retrieved in this API is not used, be sure to close it by RpCloseMonPrinter.
- When the printer driver connects to FILE, this API fails.
- This API succeeds even when the printer is not connected to the system or the printer power is turned off.

RpCloseMonPrinter

Ends using the Communication library with the specified API handle.

INT RpCloseMonPrinter(

INT i_hdl)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- RpSetStatusBackFunction and RpSetStatusBackWnd stop monitoring the POS printer status.
- When the API handle called with this API is used in another API, this API is not executed until it is completed.
- The command registered by RpSendDataFile is discarded.

RpLockPrinter

Locks all data transmission and hardware reset requests from other processes to the printer.

INT RpLockPrinter(

INT i_hdl,

DWORD i_timeout)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_timeout

Timeout period

Specifies the timeout period in msec (millisecond).

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- This API allows to exclusively occupy the data transmission and reset control to the printer. To release it, use RpUnlockPrinter.
- When an API which performs data transmission or reset control to the printer from another process is used after the calling of this API until RpUnlockPrinter execution, it will fail.
- An occupation by this API is valid within the process. Therefore, during the occupation, an API can directly access the device from another thread in the same process.
- The number of times this API is repeatedly executed with an already occupied API handle is up to 99 times. To release it, execute RpUnlockPrinter the same times as for this API.
- The range of i_timeout is from 3000 ms to 90000 ms. When the value is less than 3000 ms, it is corrected to 3000 ms and when the value is more than 90000 ms, it is corrected to 90000 ms.

RpUnlockPrinter

Unlocks the access prohibition (lock) from other processes by RpLockPrinter.

INT RpUnlockPrinter(

INT i_hdl)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- Unlocks the lock of the printer set by RpLockPrinter.
- When RpLockPrinter is called more than one time, this API must be called the same time as for RpLockPrinter to release the lock.

RpDirectIO

Sends/Receives binary data.

    INT RpDirectIO(
        INT i_hdl,
        BYTE i_wlen,
        LPBYTE i_wcmd,
        LPBYTE io_rlen,
        LPBYTE o_rbuf,
        DWORD i_timeout,
        BOOL i_flag)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_wlen

Size of data to send

Specifies the size of data to send.

i_wcmd

Buffer of data to send

Specifies the buffer to store the data to send.

io_rlen

Size of data to receive

Specifies the maximum length of data to be received from the printer.

When no data retrieval is needed, specify "0".

When the API returns, the size of the retrieved receive data is stored.

o_rbuf

    Buffer of data to receive

        Specifies the buffer to store the data to retrieve.

i_timeout

    Timeout period

        Specifies the timeout period in msec (millisecond).

i_flag

    Receive operation flag

        Specifies one of the following flags for the receive operation.

        TRUE: Continues receiving until anything is received or timeout occurs.

        FALSE: Continues receiving until the data of read size is received or timeout occurs.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- This API is aborted by RpResetPrinter.
- The range of i_timeout is from 3000 ms to 90000 ms. When the value is less than 3000 ms, it is corrected to 3000 ms and when the value is more than 90000 ms, it is corrected to 90000 ms.
- When multiple processes use the Communication library and they use this API to send divided data more than one time, unexpected output from other processes may interrupt before completion of transmission. When outputting such Printer commands and data that do not allow interrupting of other data, especially for image data, be sure to use this API after calling RpLockPrinter.
- Receive data does not include responses of the ASB Setting command. When responses including ones of the ASB Setting command are retrieved, execute RpDirectIOEx.
- Do not include the data which disables the ASB Setting command in the binary data to send. Otherwise, an API which retrieves POS printer status does not work properly.
- For Bluetooth connection, do not include a printer command "Hardware Reset" or "Printer Reset" in the binary data to send.
  When executing hardware reset, use RpResetPrinter.

RpDirectIOEx

Sends/Receives binary data.

```
INT RpDirectIOEx(
    INT i_hdl,
    DWORD i_wlen,
    LPBYTE i_wcmd,
    LPDWORD io_rlen,
    LPBYTE o_rbuf,
    DWORD i_timeout,
    BOOL i_flag,
    BYTE i_op )
```

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_wlen

Size of data to send

Specifies the size of data to send.

i_wcmd

Buffer of data to send

Specifies the buffer to store the data to send.

io_rlen

Size of data to receive

Specifies the maximum length of data to be received from the printer.

When no data retrieval is needed, specify "0".

When the API returns, the size of the receive data is stored.

o_rbuf

Buffer of data to receive

Specifies the buffer to store the data to retrieve.

i_timeout

Timeout period

Specifies the timeout period in msec (millisecond).

i_flag

Receive operation flag

Specifies one of the following flags for the receive operation.

TRUE: Continues receiving until anything is received or timeout occurs.

FALSE: Continues receiving until the data of read size is received or timeout occurs.

i_op

Receive target option

Specifies one of the following options for data to receive.

0: Retrieves data excluding responses of the ASB Setting command

1: Retrieves data including responses of the ASB Setting command

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- This API is aborted by RpResetPrinter.
- The range of i_timeout is from 3000 ms to 90000 ms. When the value is less than 3000 ms, it is corrected to 3000 ms and when the value is more than 90000 ms, it is corrected to 90000 ms.
- When multiple processes use the Communication library and they use this API to send divided data more than one time, unexpected output from other processes may interrupt before completion of transmission. When outputting such Printer commands and data that do not allow interrupting of other data, especially for image data, be sure to use this API after calling RpLockPrinter.
- Size of data to receive is up to 4096 bytes. When an exceeding data size is specified, data for 4096 bytes is set.
- Do not include the data which disables the ASB Setting command in the binary data to send. Otherwise, an API which retrieves POS printer status does not work properly.
- For Bluetooth connection, do not include a printer command "Hardware Reset" or "Printer Reset" in the binary data to send.
  When executing hardware reset, use RpResetPrinter.

RpResetPrinter

Resets the printer.

INT RpResetPrinter(

INT i_hdl)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- Resets the printer using the communication protocol (without using printer commands).
- When this API is called during execution of RpDirectIO, RpDirectIOEx, or RpDirectSendRead, these APIs are aborted.
- After performing hardware reset with this API, wait for a few seconds until the printer reset process completes before outputting any data. Data output during the hardware reset may cause skipped data.
- During execution of this API, POS printer status responds disconnected state.
- When executing this function but the printer is in data unaccepting state, the printer resetting may not be correctly completed and garble character may occur.

RpGetStatus

Retrieves the latest POS printer status.

INT RpGetStatus(

INT i_hdl,

LPDWORD o_status)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

o_status

POS printer status variable

Specifies the variable to store the POS printer status.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- When reconnection of the printer is detected, the POS printer status received last is returned.
- When the POS printer status cannot be retrieved in connected condition, this API fails.

- For details of the POS printer status, see "6.1 POS Printer Status List".

RpSetStatusBackFunction

Registers the callback function to be called when a change of the POS printer status is detected.

  INT RpSetStatusBackFunction(

   INT i_hdl,

   INT (CALLBACK EXPORT *lpStatusCB) (DWORD o_st))

Parameters

  i_hdl

   API handle

    Specifies the API handle retrieved by RpOpenMonPrinter.

  lpStatusCB

   Callback function address

    Specifies the address of an application-defined callback function to receive the POS printer status.

  o_st

   POS printer status variable

    Specifies the variable to store the POS printer status.

Return value

  Returns 0 for success.

  Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- The callback function registered by this API is unregistered by RpCancelStatusBack and RpCloseMonPrinter. It is also unregistered by specifying NULL for lpStatusCB.
- APIs in the Communication library cannot be called with the same API handle from the registered callback function.
- When the POS printer status cannot be retrieved in connected condition, this API fails.
- When reconnection is detected, the POS printer status received last is returned.
- Even when the POS printer status is received, the callback function is not called when the POS printer status has not changed from when it is last received.
- When the callback function is registered with this API, the callback function is called with the current POS printer status.
- When the callback function is already registered and this API is called again, the registered API becomes invalid and the new callback function is registered.
- Even when this API is called again specifying the already registered valid callback function, the POS printer status response is performed immediately after it.
- The return value of the callback function is ignored.

- The time between receiving the POS printer status and calling the callback function is not guaranteed.
- For details of the POS printer status, see "6.1 POS Printer Status List".

RpSetStatusBackWnd

Registers the window handle of a button for which the click event is called when a change of the POS printer status is detected, and the variable to set the POS printer status.

    INT RpSetStatusBackWnd(

        INT i_hdl,

        HANDLE i_Wnd,

        LPDWORD i_status)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_Wnd

Window handle

Specifies the window handle of the button to send the click event.

i_status

POS printer status variable

Specifies the variable to store the POS printer status.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- This API is unregistered by RpCancelStatusBack and RpCloseMonPrinter. It is also unregistered by specifying NULL for the window handle of the button.
- When the POS printer status cannot be retrieved in connected condition, this API fails.
- When reconnection is detected, the POS status received last is returned.
- Even when the POS printer status is received, the click event is not called when the POS printer status has not changed from when it is last received.
- When the window handle of a button is registered by this API, the click event with the current POS printer status is called.
- When the callback function is registered with this API, the click event is called with the current POS printer status.
- When this API is called again when the window handle of a button is already registered, it is disabled and the one of the new button is registered.

- Even when this API is called again specifying the already registered valid window handle of a button, the POS printer status response is performed immediately after it.
- The return value of the click event is ignored.
- The time between receiving the POS printer status and calling the click event is not guaranteed.
- For details of the POS printer status, see "6.1 POS Printer Status List".

RpCancelStatusBack

Unregisters the callback function which was executed in the RpSetStatusBackWnd and RpSetStatusBackFunction.

INT RpCancelStatusBack(

INT i_hdl)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- This API succeeds even when neither RpSetStatusBackWnd nor RpSetStatusBackFunction are registered.

RpPowerOff

Turns the printer power off.

INT RpPowerOff(

INT i_hdl)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- When this API is executed, the power off operation is performed on the printer.

RpGetCounter

Retrieves the specified maintenance counter.

INT RpGetCounter(

    INT i_hdl,

    WORD i_readno,

    LPDWORD o_readcounter)

Parameters

i_hdl

    API handle

        Specifies the API handle retrieved by RpOpenMonPrinter.

i_readno

    Counter ID

        Specifies the counter ID to retrieve.

o_readcounter

    Counter variable

        Specifies the variable to store the counter value to retrieve.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- For details of the counter ID, see "6.2 Counter ID".

RpResetCounter

Initializes the specified maintenance counter.

INT RpResetCounter(

INT i_hdl,

WORD i_readno)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_readno

Counter ID

Specifies the counter ID to initialize.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- For details of the counter ID, see "6.2 Counter ID".
- Check that the counter ID value specified by this API is initialized by using RpGetCounter.

RpGetType

Retrieves the type ID and font type of the printer.

INT RpGetType(

INT i_hdl,

LPBYTE o_typeID,

LPBYTE o_font,

LPBYTE o_exrom,

LPBYTE o_special)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

o_typeID

Type ID variable

Specifies the variable to store the type ID.

o_font

Font type variable

Specifies the variable to store the font type.

o_exrom

Reserved

Specifies NULL.

o_special

Reserved

Specifies NULL.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- For details of the type ID, see "6.3 Type ID".
- For details of the font type, see "6.4 Font Type".

RpGetPrnCapability

Retrieves the specified printer information.

INT RpGetPrnCapability(

INT i_hdl,

BYTE i_id,

LPBYTE io_datsize,

LPBYTE o_dat)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_id

Printer ID

Specifies the printer ID to retrieve.

io_datsize

Size of data to receive

Specifies the size of the buffer to store the printer ID to retrieve.

When the API returns, retrieved size of data to receive is stored.

o_dat

Buffer of data to receive

Specifies the buffer to store the value of printer ID to retrieve.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- For details of the printer ID, see "6.5 Printer ID".
- When the size specified by io_datsize is smaller than the response size of the specified printer ID or NULL is specified for o_dat, API fails and the response size of the specified printer ID is stored in io_datsize.

RpOpenDrawer

Opens the specified drawer.

    INT RpOpenDrawer(
        INT i_hdl,
        BYTE i_drawer,
        BYTE i_pulse)

Parameters

i_hdl

    API handle

        Specifies the API handle retrieved by RpOpenMonPrinter.

i_drawer

    Target drawer

        Specifies the drawer to use.

i_pulse

    Drawer kick time

        Specifies the drawer kick time for the target drawer.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- For details of the target drawer, see "6.6 Target Drawer".
- For details of the drawer kick time, see "6.7 Drawer Kick Time".

RpSendDataFile

Registers the command using the specified command definition file.

    INT RpSendDataFile(

        INT i_hdl,

        LPCTSTR i_fname)

Parameters

i_hdl

    API handle

        Specifies the API handle retrieved by RpOpenMonPrinter.

i_fname

    Command definition file name

        Specifies the name of a command definition file created in the predefined format.

<u>**Return value**</u>

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

<u>**Remarks**</u>

- For details of the command definition file, see "Chapter 7 Command Definition File".
- The command registered by this API is discarded by calling RpCloseMonPrinter.

RpDirectSendRead

Executes a command registered by RpSendDataFile.

INT RpDirectSendRead(

INT i_hdl,

LPCTSTR i_cname,

LPCTSTR i_rtype,

LPDWORD io_rlen,

LPBYTE o_rbuf,

DWORD i_timeout,

BOOL i_flag)

<u>**Parameters**</u>

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_cname

Command name

Specifies a command name defined in RpSendDataFile.

i_rtype

Receive data type name

ASB: Stores only the responses of the ASB Setting command in the receive data.

Other: Stores the other responses from the printer in the receive data.

io_rlen

   Size of data to receive

      Specifies the maximum length of data to be received from the printer.

      When no data retrieval is needed, specify "0".

      When the API returns, the size of the retrieved receive data is stored.

o_rbuf

   Buffer of data to receive

      Specifies the buffer to store the data to retrieve.

i_timeout

   Timeout period

      Specifies the timeout period in msec (millisecond).

i_flag

   Receive operation flag

      Specifies one of the following flags for the receive operation.

      TRUE: Continues receiving until anything is received or timeout occurs.

      FALSE: Continues receiving until the data of read size is received or timeout occurs.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- Do not include the data which disables the ASB Setting command in the binary data to send. Otherwise, an API which retrieves POS printer status does not work properly.
- This API is aborted by RpResetPrinter.
- The range of i_timeout is from 3000 ms to 90000 ms. When the value is less than 3000 ms, it is corrected to 3000 ms and when the value is more than 90000 ms, it is corrected to 90000 ms.
- Size of data to receive is up to 4096 bytes. When an exceeding data size is specified, data for 4096 bytes is set.
- For Bluetooth connection, do not include a printer command "Hardware Reset" or "Printer Reset" in the binary data to send.
  When executing hardware reset, use RpResetPrinter.

RpGetProperty

Retrieves the content of the specified property ID.

```
INT RpGetProperty(
    INT i_hdl,
    LPDEVMODE i_devmode,
    BYTE i_pid,
    LPBYTE o_dat,
    LPDWORD io_size )
```

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_devmode

Devmode address

Specifies the Devmode address.

i_pid

Property ID

Specifies the property ID to retrieve.

o_dat

Buffer of data to retrieve

Specifies the buffer to store the content of the property ID to retrieve.

io_size

Size of data to retrieve

Specifies the maximal length of data to retrieve.

When the API returns, the size of the retrieved data is stored.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

- For details of the property ID, see "6.8 Property ID".
- When the size specified by io_size is smaller than the response size of the specified property ID or NULL is specified for o_dat, the response size of the specified property ID is stored in io_size.

RpSetProperty

Changes the content of the specified property ID.

INT RpSetProperty(

INT i_hdl,

LPDEVMODE i_devmode,

BYTE i_pid,

LPBYTE o_dat,

LPDWORD i_size)

Parameters

i_hdl

API handle

Specifies the API handle retrieved by RpOpenMonPrinter.

i_devmode

Devmode address

Specifies the Devmode address.

i_pid

Property ID

Specifies the property ID to change.

o_dat

Buffer of data to set

Specifies the buffer to store the content of the property ID to change.

i_size

Size of data to set

Specifies the buffer size to store the content of the property ID to change.

Return value

Returns 0 for success.

Returns an error code for failure. See "Chapter 5 Error Code List" for details.

Remarks

- For details of the property ID, see "6.8 Property ID".

Chapter 4 .NET API

4.1 Overview

This chapter describes the Communication library for .NET development environment (.NET API).

4.2 Development Language

The following development languages are covered.

- Visual Basic .NET
- Visual C#

4.3 Library File

The Communication library has the following file name.

- SiiRpe1ClassLib.dll

The Communication library file is stored in the Global Assembly Cache (GAC) folder.

4.4 API List

The following APIs are implemented in the Communication library.

- Namespace: SiiPrinterSdk
- Class name: StatusAPI

Category	API	Brief Description of the Function
Property	Status	Retrieves the latest POS printer status.
Property	LastError	Retrieves the error value of the last executed API.
Property	IsValid	Retrieves the open state.
Method	OpenMonPrinter	Starts using the Communication library in the specified printer.
Method	CloseMonPrinter	Ends using the Communication library.
Method	LockPrinter	Locks all data transmission and hardware reset requests from other processes to the printer.
Method	UnlockPrinter	Unlocks the access prohibition (lock) from other processes by LockPrinter.
Method	DirectIOEx	Sends/Receives binary data.
Method	ResetPrinter	Resets the printer.
Method	SetStatusBack	Registers the callback method to be called when a change of the POS printer status is detected.
Method	CancelStatusBack	Unregisters the callback method which was executed in the SetStatusBack.
Method	PowerOff	Turns the printer power off.
Method	GetCounter	Retrieves the specified maintenance counter.
Method	ResetCounter	Initializes the specified maintenance counter.
Method	GetType	Retrieves the type ID and font type of the printer.
Method	GetPrnCapability	Retrieves the specified printer information.
Method	OpenDrawer	Opens the specified drawer.
Method	SendDataFile	Registers the transmission data using the specified command definition file.
Method	DirectSendRead	Executes a command defined in SendDataFile.
Method	GetProperty	Retrieves the specified property ID.
Method	SetProperty	Changes the specified property ID.
Event	StatusCallback	Event to process the responded POS printer status.

4.5 Property

Status

Retrieves the latest POS printer status.

    SII.Driver.PosPrinter.ASB Status { get; }

Initial value

    0

Remarks

- To retrieve a failure of this property, use LastError. In case of a failure, the value of this property is not defined.
- For details of retrievable values, see "6.1 POS Printer Status List".
- For details, see RpGetStatus in "Chapter 3 Win32 API".

LastError

Retrieves the error value of the last executed API.

    SII.Driver.PosPrinter.ErrorCode LastError { get; }

Initial value

    SUCCESS

Remarks

For details of retrievable values, see "Chapter 5 Error Code List".

IsValid

Retrieves the open state.

> bool IsValid { get; }

Initial value

FALSE

Remarks

One of the following values are retrieved.

- TRUE: Already opened state.
- FALSE: Not opened state.

4.6 Method

OpenMonPrinter

Starts using the Communication library in the specified printer.

ErrorCode OpenMonPrinter(

OpenType type,

string name)

Parameters

type

Open type

TYPE_PRINTER: Specifies the printer driver name to open the printer (fixed).

name

Name of the printer that uses the Communication library

Specifies the printer name (friendly name) to output.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

- The number that printers open per process is 1 only.
- When the Communication library is not used, be sure to call CloseMonPrinter.
- When the printer driver connects to FILE, this API fails.
- This API succeeds even when the printer is not connected to the system or the printer power is turned off.

CloseMonPrinter

Ends using the Communication library.

ErrorCode CloseMonPrinter ()

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details, see RpCloseMonPrinter in "Chapter 3 Win32 API".

LockPrinter

Locks all data transmission and hardware reset requests from other processes to the printer.

ErrorCode LockPrinter(
    int timeout)

Parameters

timeout

Timeout period

Specifies the timeout period in msec (millisecond).

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details, see RpLockPrinter in "Chapter 3 Win32 API".

UnlockPrinter

Unlocks access prohibition (lock) from other processes by LockPrinter.

ErrorCode UnlockPrinter()

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details, see RpUnlockPrinter in "Chapter 3 Win32 API".

DirectIOEx

Sends/receives binary data.
Retrieves the receive data as binary data from the printer.

ErrorCode DirectIOEx(

byte[] cmd,

ref byte[] data,

int timeout,

bool readFlag,

byte option)

Sends/receives binary data.
Retrieves the receive data as string data from the printer.

ErrorCode DirectIOEx(

byte[] cmd,

out string data,

int timeout,

byte option)

Sends binary data.

ErrorCode DirectIOEx(

byte[] cmd,

int timeout)

Parameters

cmd

Buffer of data to send

Specifies the buffer to store the data to send.

data

Buffer of data to receive

Specifies the buffer to store the data to retrieve.

timeout

Timeout period

Specifies the timeout period in msec (millisecond).

readFlag

Receive operation flag

Specifies one of the following flags for the receive operation.

TRUE: Continues receiving until anything is received or timeout occurs.

FALSE: Continues receiving until the data of read size is received or timeout occurs.

option

Receive target option

Specifies one of the following options for data to receive.

0: Retrieves data excluding responses of the ASB Setting command

1: Retrieves data including responses of the ASB Setting command

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

- When this API succeeds, ref byte[] data is resized to the size of the receive data, with an upper limit of the size specified before the call.
- Any 0x02 included in the receive data is converted to 0x5f.
- For details, see RpDirectIOEx in "Chapter 3 Win32 API".

ResetPrinter

Resets the printer.

ErrorCode ResetPrinter()

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details, see RpResetPrinter in "Chapter 3 Win32 API".

SetStatusBack

Registers the callback method to be called when a change of the POS printer status is detected.

ErrorCode SetStatusBack()

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details, see RpSetStatusBackFunction in "Chapter 3 Win32 API".

CancelStatusBack

Unregisters the callback method which was executed in the SetStatusBack.

ErrorCode CancelStatusBack()

Return value

For details of the return value, see "Chapter 5 Error Code List".

For details, see RpCancelStatusBack in "Chapter 3 Win32 API".

PowerOff

Turns the printer power off.

ErrorCode PowerOff()

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details, see RpPowerOff in "Chapter 3 Win32 API".

GetCounter

Retrieves the specified maintenance counter.

ErrorCode GetCounter(

CounterIndex index,

bool type,

out int data)

ErrorCode GetCounter(

byte index,

out int data)

Parameters

index

Counter ID

Specifies the counter ID to retrieve or the value of a counter ID defined in

SII.Driver.PosPrinter.CounterIndex.

type

Type of maintenance counter

One of the following types of maintenance counter to retrieve

TRUE: Integrated counter

FALSE: Initializable counter

data

    Counter variable

        Specifies the variable to store the counter value to retrieve.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details of the counter ID, see "6.2 Counter ID".

For details, see RpGetCounter in "Chapter 3 Win32 API".

ResetCounter

Initializes the specified maintenance counter.

    ErrorCode ResetCounter(

        CounterIndex index)

    ErrorCode ResetCounter(

        byte index)

Parameters

index

    Counter ID

        Specifies the counter ID to initialize or the value of a counter ID defined in

        SII.Driver.PosPrinter.CounterIndex.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details, see RpResetCounter in "Chapter 3 Win32 API".

GetType

Retrieves the type ID and font type of the printer.

ErrorCode GetType(

out byte typeId,

out byte font,

out byte exrom,

out byte special)

Parameters

typeId

Type ID variable

Specifies the variable to store the type ID.

font

Font type variable

Specifies the variable to store the font type.

exrom

Reserved

Specifies NULL.

special

Reserved

Specifies NULL.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details of the type ID, see "6.3 Type ID".

For details of the font type, see "6.4 Font Type".

For details, see RpGetType in "Chapter 3 Win32 API".

GetPrnCapability

Retrieves the specified printer information.
Retrieves the response data as binary data.

    ErrorCode GetPrnCapability(

        byte id,

        out byte[] data)

Retrieves the specified printer information.
Retrieves the response data as string data.

    ErrorCode GetPrnCapability(

        byte id,

        out string data)

Parameters

id

Printer ID

    Specifies the printer ID to retrieve.

data

Buffer of data to receive

    Specifies the buffer to store the value of printer ID to retrieve.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details of the printer ID, see "6.5 Printer ID".

For details, see RpGetPrnCapability in "Chapter 3 Win32 API".

OpenDrawer

Opens the specified drawer.

ErrorCode OpenDrawer(
Drawer drawer,
Pulse pulse)

Parameters

drawer

Target drawer

Specifies the value of target drawer to open, defined in SII.Driver.PosPrinter.Drawer.

pulse

Drawer kick time

Specifies the value of drawer kick time, defined in SII.Driver.PosPrinter.Drawer.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details of the target drawer, see "6.6 Target Drawer".

For details of the drawer kick time, see "6.7 Drawer Kick Time".

For details, see RpOpenDrawer in "Chapter 3 Win32 API".

SendDataFile

Registers the transmission data using the specified command definition file.

ErrorCode SendDataFile(
string fileName)

Parameters

fileName

Command definition file name

Specifies the name of a command definition file created in the predefined format.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details of the command definition file, see "Chapter 7 Command Definition File".

For details, see RpSendDataFile in "Chapter 3 Win32 API".

DirectSendRead

Executes a command defined in SendDataFile.
Retrieves the receive data as binary data from the printer.

ErrorCode DirectSendRead(

string cmdName,

string readType,

ref byte[] data,

int timeout,

bool readFlag)

Executes a command defined in SendDataFile.
Retrieves the receive data as string data from the printer.

ErrorCode DirectSendRead(

string cmdName,

string readType,

out string data,

int timeout)

Executes a command defined in SendDataFile.

ErrorCode DirectSendRead(

string cmdName,

string readType,

int timeout)

Parameters

cmdName

Command name

Specifies a command name defined in SendDataFile.

readType

Receive data type name

ASB: Stores only the responses of the ASB Setting command in the receive data.

Other: Stores the other responses from the printer in the receive data.

data

Buffer of data to receive

Specifies the buffer to store the data to retrieve.

timeout

Timeout period

Specifies the timeout period in msec (millisecond).

readFlag

Receive operation flag

Specifies one of the following flags for the receive operation.

TRUE: Continues receiving until anything is received or timeout occurs.

FALSE: Continues receiving until the data of read size is received or timeout occurs.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

- Any 0x02 included in the receive data is converted to 0x5f.
- For details, see RpDirectSendRead in "Chapter 3 Win32 API".

GetProperty

Retrieves the content of the specified property ID.

    ErrorCode GetProperty(
        IntPtr devmode,
        byte id,
        byte[] data,
        ref uint size)

Parameters

devmode

DevMode address

Specifies the Devmode address.

id

Property ID

Specifies the property ID to retrieve.

data

Buffer of data to retrieve

Specifies the buffer to store the content of the property ID to retrieve.

size

Size of data to retrieve

Specifies the maximal length of data to retrieve.

When the API returns, the size of the retrieved data is stored.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

For details of the property ID, see "6.8 Property ID".

For details, see RpGetProperty in "Chapter 3 Win32 API".

SetProperty

Changes the content of the specified property ID.

ErrorCode SetProperty(

IntPtr devmode,

byte id,

byte[] data,

uint size)

Parameters

devmode

DevMode address

Specifies the Devmode address.

id

Property ID

Specifies the property ID to change.

data

Buffer of data to set

Specifies the buffer to store the content of the property ID to change.

size

Size of data to set

Specifies the size of the buffer to store the content of the property ID data to change.

Return value

For details of the return value, see "Chapter 5 Error Code List".

Remarks

- For details of the property ID, see "6.8 Property ID".
- For details, see RpSetProperty in "Chapter 3 Win32 API".

4.7 Event

StatusCallback

Event to process the responded POS printer status.

event StatusCallbackHandler StatusCallback

delegate void StatusCallbackHandler(

ASB status)

Parameters

status

POS printer status variable

Specifies the variable to store the POS printer status.

Remarks

- For details of the POS printer status, see "6.1 POS Printer Status List".
- For details, see RpSetStatusBackFunction in "Chapter 3 Win32 API".

Chapter 5   Error Code List

5.1 Error Code List

Major error codes are as follows.

Macro Definition (Constant)	Value	Description
SUCCESS	0	Success
ERR_TYPE	-10	Open type parameter error
ERR_OPENED	-20	Specified printer has already been opened.
ERR_NO_PRINTER	-30	Specified printer driver does not exist.
ERR_HANDLE	-60	API handle value is incorrect.
ERR_TIMEOUT	-70	Timeout or busy state occurs.
ERR_ACCESS	-80	Printer cannot be accessed.
ERR_PARAM	-90	Parameter is incorrect.
ERR_NOT_SUPPORT	-100	Function is not supported.
ERR_OFFLINE	-110	Printer is disconnected or offline.
ERR_NOT_SII	-120	Printer driver is not supported.
ERR_DISK_FULL	-170	Printer is busy state.
ERR_ENTRY_OVER	-190	Maximum processing capacity is exceeded.
ERR_EXIST	-210	Existing module is called.
ERR_NOT_FOUND	-220	File cannot be found. Or, it may not be registered.
ERR_WORKAREA_NO_MEMORY	-260	Specified memory size is insufficient.
ERR_WORKAREA_FAILED	-280	Memory cannot be reserved.
ERR_EXEC_FUNCTION	-310	Function is not available because it is used by other thread or process.
ERR_SPL_NOT_EXIST	-350	Spooler service has not been started.
ERR_LOCKED	-1000	Printer is locked.
ERR_UNLOCKED	-1010	Printer is not locked.
ERR_INVALID_DATA	-1020	Invalid data is specified.
ERR_READ_FAULT	-1030	Data cannot be received from printer.
ERR_WRITE_FAULT	-1040	Data cannot be sent to printer.
ERR_CANCELLED	-1050	Function is canceled.
ERR_PRINTER_HAS_JOBS_QUEUED	-1060	Printer has queued job.
ERR_UNKNOWN_PORT	-1070	Port is not supported.
ERR_INVALID_PRINTER_STATE	-1080	Printer status is abnormal.

Macro Definition (Constant)	Value	Description
ERR_BAD_ENVIRONMENT	-1090	Printer driver installation may be abnormal.

Chapter 6   Argument Information

6.1 POS Printer Status List

```
= Caution =
◆ The POS printer status includes the status to respond for the ASB Setting command
  and some extended statuses.
◆ Values not described in the table below are reserved.
```

There are the following responses for the POS printer status.

Status	Response Value		Description
Voltage error	ASB_VP_ERR	0x00000000	No voltage error
		0x00000001	Voltage error
Head error/Voltage initialization error	ASB_HEAD_ERR	0x00000000	No head error/voltage initialization error
		0x00000002	Head error/voltage initialization error
Head temperature error	ASB_HEAD_TEMPERATUR_ERR	0x00000000	No head temperature error
		0x00000004	Head temperature error
Autocutter error	ASB_AUTOCUTTER_ERR	0x00000000	No autocutter error
		0x00000008	Autocutter error
Out-of-paper error	ASB_RECEIPT_END	0x00000000	No out-of-paper error
		0x00000010	Out-of-paper error
Paper-near-end	ASB_RECEIPT_NEAR_END	0x00000000	No paper-near-end
		0x00000020	Paper-near-end

Status	Response Value		Description
Paper jam error while detecting mark	ASB_MARK_PAPER_JAM_ERR	0x00000000	No paper jam error while detecting mark
		0x00000040	Paper jam error while detecting mark
Cover open error	ASB_COVER_OPEN	0x00000000	Cover is closed
		0x00000080	Cover is opened
FEED switch state	ASB_PAPER_FEED	0x00000000	FEED switch state = off
		0x00000100	FEED switch state = on
Printing	ASB_NOW_PRINTING	0x00000000	Stop
		0x00000400	Printing
Recovery waiting state	ASB_RETURN_WAITING	0x00000000	-
		0x00000800	Recovery waiting state
Drawer sensor state	ASB_DRAWER_KICK	0x00000000	Drawer sensor state = "Low"
		0x00008000	Drawer sensor state = "High"
FLASH memory rewriting	ASB_FLASH_MEMORY_REWRITING	0x00000000	-
		0x00010000	FLASH memory rewriting
Peripheral device selection	ASB_PERIPHERAL_EQUIPMENT	0x00000000	Printer
		0x00020000	Other
Automatic recovery error	ASB_AUTORECOVER_ERR[1]	0x00000000	No automatic recovery error
		0x20000000	Automatic recovery error
Unrecoverable error	ASB_UNRECOVER_ERR[1]	0x00000000	No unrecoverable error
		0x40000000	Unrecoverable error
No response	ASB_NO_RESPONSE[1]	0x00000000	Printer responds
		0x80000000	Disconnection or communication error

*1: Extended state to the responses of the ASB Setting command.

The retrievable response value is the addition of the above value. However, the value turns 0x80000000 for No response.

6.2 Counter ID

There are the following responses for the counter ID.

Counter ID			Description
ROLL_FEED_LINES	20	Initializable	Number of dot lines for paper feed (unit: 100-dot line)
ROLL_HEAD_CHARGE	21	Initializable	Thermal head activation time (unit: 100-dot line)
PAPER_CUT	50	Initializable	Number of autocutter drive times
OPERATION_TIME	70	Initializable	Product drive time (unit: minute)
ROLL_FEED_LINES	148	Integrated value	Number of dot lines for paper feed (unit: 100-dot line)
ROLL_HEAD_CHARGE	149	Integrated value	Thermal head activation time (unit: 100-dot line)
PAPER_CUT	178	Integrated value	Number of autocutter drive times
OPERATION_TIME	198	Integrated value	Product drive time (unit: minute)

6.3 Type ID

There are the following responses for the type ID.

Response Value	Description
0x01	0: Multi-byte code not supported 1: Multi-byte code supported
0x02	0: Autocutter not available 1: Autocutter available
0x04	Fixed to zero
0x08	Fixed to zero
0x10	Fixed to zero
0x20	Fixed to zero
0x40	Fixed to zero
0x80	Fixed to zero

The retrievable response value is the addition of the above value.

6.4 Font Type

There are the following responses for the font type.

Response Value	Description
2	Japanese kanji(JIS)

6.5 Printer ID

There are the following available printer IDs with their responses.

Printer ID	Description		Response Format
1	Printer model ID	0x1a	HEX code
2	Type ID	See "6.3 Type ID".	HEX code
3	ROM version ID	Depends on ROM version	HEX code
65	Firmware version (main)	"X.XX.XX"	ASCII string
66	Manufacturer	"Seiko Instruments Inc."	ASCII string
67	Model name	"SII RP-E10 Series."	ASCII string
69	Multi-language font type	For Japanese: "KANJI JAPANESE"	ASCII string

6.6 Target Drawer

There are the following available target drawers.

	Target Drawer	Description
1	SII_RP_DRAWER_1	Opens drawer 1
2	SII_RP_DRAWER_2	Opens drawer 2

6.7 Drawer Kick Time

There are the following available drawer kick times.

	Drawer Kick Time	Description
1	SII_RP_ PULSE_100	Drives the drawer in 100 millisecond
2	SII_RP_ PULSE_200	Drives the drawer in 200 millisecond
3	SII_RP_ PULSE_300	Drives the drawer in 300 millisecond
4	SII_RP_ PULSE_400	Drives the drawer in 400 millisecond
5	SII_RP_ PULSE_500	Drives the drawer in 500 millisecond
6	SII_RP_ PULSE_600	Drives the drawer in 600 millisecond
7	SII_RP_ PULSE_700	Drives the drawer in 700 millisecond
8	SII_RP_ PULSE_800	Drives the drawer in 800 millisecond

6.8 Property ID

Caution

◆ For a property ID not described below, the value set in the printer driver is valid.

◆ Specify the data size to 1 byte except for custom command for using RpSetProperty.

◆ The phrase in square brackets ([ ]) indicates the timing of corresponding process.

There are the following available property IDs with their contents.

Property ID	Description	
1	Initialize	0: Enabled 1: Disabled
2	Speed*1	0： Middle (Silent) 1： Low 2： Middle (Quality) 3： High
3	Margins	0: Minimum Margin 1: Minimum Top Margin 2: Minimum Bottom Margin 3: Maximum Margin
4	Density (percentage)	70 to 130
5	Direction	0: Front to Back 1: Back to Front
6	Reduction	0: None 25 to 100: Scale (percentage)
7	Paper Cut	0: None 1: Full cut (Each job) 2: Partial cut (Each job) 3: Full cut (Each page) 4: Partial cut (Each page) 5: Partial cut (Between pages)
8	Marked Paper Form Feed	0: Disabled 1: Each page 2: Each job
9	Feed to Cut Position	0: Enabled 1: Disabled
10	[Print Start] Logo	0: Disabled 1: Left 2: Center 3: Right
11	[Print Start] Logo Keycode	0 to 99
12	[Print Start] Drawer*2	0: Disabled 1: Drawer 1 2: Drawer 2

Property ID		Description
13	[Print Start] ON Time (×2 ms)	1 to 255
14	[Print Start] OFF Time (×2 ms)	1 to 255
15	[Print Start] Custom Command. (128 bytes)	The Printer command
16[*3]	[Print Start] Paper feed (backward feed) (dot)	0 to 74
17[*3]	[Print Start] Paper feed (feed) (dot)	0 to 255
18	[Print Start] Buzzer	0: ON 1: OFF
20	[Page Start] Logo	0: Disabled 1: Left 2: Center 3: Right
21	[Page Start] Logo Keycode	0 to 99
25	[Page Start] Custom Cmd. (128 bytes)	The Printer command
26[*3]	[Page Start] Paper feed (backward feed) (dot)	0 to 74
27[*3]	[Page Start] Paper feed (feed) (dot)	0 to 255
28	[Page Start] Buzzer	0: ON 1: OFF
30	[Page End] Logo	0: Disabled 1: Left 2: Center 3: Right
31	[Page End] Logo Keycode	0 to 99
35	[Page End] Custom Cmd. (128 bytes)	The Printer command
36[*3]	[Page End] Paper feed (backward feed) (dot)	0 to 74
37[*3]	[Page End] Paper feed (feed) (dot)	0 to 255
38	[Page End] Buzzer	0: ON 1: OFF
40	[Print End] Logo	0: Disabled 1: Left 2: Center 3: Right
41	[Print End] Logo Keycode	0 to 99
42	[Print End] Drawer[*2]	0: Disabled 1: Drawer 1 2: Drawer 2

Property ID	Description	
43	[Print End] ON Time (×2 ms)	1 to 255
44	[Print End] OFF Time (×2 ms)	1 to 255
45	[Print End] Custom Cmd. (128 bytes)	The Printer command
46[*3]	[Print End] Paper feed (backward feed) (dot)	0 to 74
47[*3]	[Print End] Paper feed (feed) (dot)	0 to 255
48	[Print End] Buzzer	0: ON 1: OFF
50	Paper Size	0: Letter 1: A4 2: 80 mm 3: 58 mm 4 to 255: Paper except for the above[*4]
51	Orientation	0: Portrait 1: Landscape
52	Color Printing Mode	0: System[*5] 1: Driver[*5]
53	Watermark	0: None 1: Top Left 2: Top Center 3: Top Right 4: Left 5: Center 6: Right 7: Bottom Left 8: Bottom Center 9: Bottom Right
60[*6]	Preset	0: 80mm Receipt Setting 1: 58mm Receipt Setting 2: 80mm Marked Paper Setting 3: 58mm Marked Paper Setting 4: A4->80mm Reduction Setting 5: A4->58mm Reduction Setting 6: User Setting

*1: When you are using the F / W version 1.02 or earlier, select the "Speed" except "Middle (Silent)".
*2: The drawer at specific timing is not available simultaneously.
*3: When setting feed and backward feed at same timing, the last setting is valid.
*4: Cannot be set. Retrieval only.
*5: See "3.3.1 Setting of Paper/Quality" in the Printer driver user's guide for details.
*6: When specified this property, some data of other property IDs is ignored. See "3.4.4 Use of Preset" in the Printer driver user's guide.

Chapter 7   Command Definition File

7.1 Format

A command definition file is created in the following format.

Command definition

- One command definition must be specified in one line.
- Multiple commands must be specified in multiple lines.
- Comments are optional.

    [Command name 1] = [Printer command]#[Comment 1]

    [Command name 2] = [Printer command]#[Comment 2]

    [Command name 3] = [Printer command]#[Comment 3]

    •
    •
    •

Example of command definitions:

    CmdName_1="SII"#Comments1

    CmdName_2="SII" 0A#Comments2

    CmdName_3=53 49 49 0A

[Command name]

- Specifies the command name for data to send to the printer on the left of "=".
- Use ASCII characters except for "=" and "#" as command name.
- When characters other than ASCII characters are found in command name, they are ignored.
- Command name is case-sensitive.
- Registerable command name is up to 33 bytes. When 34 bytes or more is used for command name specification, up to 33 bytes are registered, and subsequent characters are ignored.
- When command name which is already registered is specified, it is ignored.

[Printer command]

- Specifies the Printer command to the printer on the right of "=".
- When you include strings in the Printer command, enclose them with "".
- When you use binary data as the Printer command, describe it in two-digit hexadecimal numbers.
  Separate each number with a space.
- Use ASCII characters for specifying the Printer command with the characters.
- Data size of the Printer command is up to 10240 bytes. Specify the Printer command up to 10240 bytes.
  However, when you use strings to specify the Printer command, they are converted to binary data to determine the actual size.
  In the above example of command definitions, the size of data to send is 3 bytes for CmdName_1, 4 bytes for CmdName_2, and 4 bytes for CmdName_3.

[Comment]

- Specify "#" at the beginning of the comment to describe.
- There is no limitation of characters used for comment.

7.2 Limitation

There are the following limitations on the command definition file.

- Readable file size is up to 4 GB.
- Available file format is ANSI and Unicode formats.
- The number of registerable command is limited by the available system memory.