SII SDK for Windows
Application Programmer's Guide

Rev.02

[Products]

MP-B20 Series

Seiko Instruments Inc.

Rev.01 February 2017
Rev.02 October 2020

SII reserves the right to make changes without notice to the specifications and materials contained herein and shall not be responsible for any damages (including consequential) caused by reliance on the materials presented, including but not limited to typographical, arithmetic, or listing errors.

Introduction

This document describes the "SII SDK for Windows" for MP-B20 Series (hereinafter referred to as "SDK") running on the "SII Printer Driver for Windows" for MP-B20 Series (hereinafter referred to as "printer driver") provided by Seiko Instruments Inc.

Symbols

This section describes symbols used in this document.

Caution

◆ Notes and limitations are described.

Target Printer Driver

The following printer driver is supported by the SDK.

- "SII Printer Driver for Windows" for MP-B20 Series

Terms

This section describes terms used in this document.

Terms	Description
Printer command	Command for controlling the printer described in "MP-B20 Series Thermal Printer Technical Reference".
ASB setting command (ASB: Automatic Status Back)	Printer command "Automatic Status Back Enable / Disable". See "Chapter 6 Command Functions" in "MP-B20 Series Thermal Printer Technical Reference" for details.
Printer status	Printer's status information can be retrieved by the SDK. This information includes the status to respond for the printer command "Automatic Status Back Enable / Disable" and some extended statuses. See "6.1 Printer Status List" for details of the printer status.

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 1 Overview

This chapter describes the overview of the SDK.

The SDK includes the library file to directly control printers, provided for developers. And also, the SDK is provided with the printer driver and uses the printer driver to work.

Using the SDK allows to directly control printers in an application development and design the application independent of the port type. Also, it is possible to get or change values of some private DEVMODE setting items.

The SDK includes the following library files:

- SDK for Win32 development environment (hereinafter, Win32 API)
- SDK for .NET development environment (hereinafter, .NET API)

For details of specific usages of the SDK, see sample programs provided for each language.

1.1 Operating Conditions

The SDK’s operating conditions basically follows the operating environment of the printer driver, and the use conditions and limitations of the memory switch. For details of the operating environment, see "SII Printer Driver for Windows User's Guide" for MP-B20 Series.

In addition, the following operating conditions must be met.

- When using the .NET API, the SDK requires .NET Framework Version 2.0 or later. When .NET Framework is uninstalled from the computer, .NET API cannot be used.
- All functions of the SDK are only available when the bidirectional support function is enabled and the printer pool function is disabled.

Chapter 2 Installation

For the SDK installation methods, see "SII Printer Driver for Windows User's Guide" for MP-B20 Series.

Chapter 3 Win32 API

This chapter describes the overview of the Win32 API.

3.1 Development Language

The following development language is covered.

- Visual C++

3.2 Library File

The library file of Win32 API is a dynamic link library format.

The library file has the following file name.

- SiiMpb2Api.dll

The library file is stored in the Windows system folder.

Use the library file without moving it from the folder. In this case, there is no need to set a path to the folder containing the library file.

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

3.3 API List

The following APIs are implemented in the Win32 API.

API	Function Summary
OpenMonPrinterA[*1] **OpenMonPrinterW**[*1]	Starts using the Win32 API and returns the API handle.
CloseMonPrinter	Ends using the Win32 API.
LockPrinter[*2]	Locks the data transmission and reset control to the printer.
UnlockPrinter	Unlocks the data transmission and reset control to the printer.
DirectIO[*2, *3]	Sends / receives binary data. (Receive data does not include responses for the ASB setting command)
DirectIOEx[*2, *3]	Sends / receives binary data.
Reset[*2, *3]	Performs hardware reset of the printer.
GetStatus	Gets the latest printer status.
SetStatusBackFunction	Registers the callback function to be called when a change of the printer status is detected.
SetStatusBackWnd	Registers the window handle of a button for which the click event is called and the variable to set the printer status when a change of the printer status is detected.
CancelStatusBack	Unregisters the callback function which was executed in **SetStatusBackWnd** and **SetStatusBackFunction.**
PowerOff[*2, *3]	Turns the printer power off.
GetCounter[*2, *3]	Gets the maintenance counter.
ResetCounter[*2, *3]	Initializes the maintenance counter.
GetType[*2, *3]	Gets the various IDs of the printer.
GetPrnCapability[*2, *3]	Gets the printer information.
SendDataFileA[*1] **SendDataFileW**[*1]	Registers the contents of the command definition file to the memory in the SDK.
DirectSendReadA[*1, *2, *3] **DirectSendReadW**[*1, *2, *3]	Executes a command registered by **SendDataFile**.
GetProperty	Gets a part of the print setting contents.
SetProperty	Changes a part of the print setting contents.

*1: Specify arguments of strings by MBCS (MultiByte Character Set) or UNICODE (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 **LockPrinter** 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.

3.4 API Details

Caution

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

OpenMonPrinter

Starts using the Win32 API and returns the API handle.

INT **OpenMonPrinter**(
INT *i_type*,
LPCTSTR *i_prt*)

Parameters

i_type
Open type
2 (fixed)

i_prt
Name of the printer that uses the Win32
Specifies the printer name (friendly name).

Return value

On success: Returns the API handle to identify the printer.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- The number of API handles that can be retrieved at the same time by 1 process is up to 8. The total number of the entire process is up to 32.
- When the API handle retrieved in this API is not used, be sure to close it by **CloseMonPrinter**.
- When the printer driver connects except for USB or Bluetooth, 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 Win32 API.

INT **CloseMonPrinter**(
INT *i_hdl*)

Parameter

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- When the API handle called with this API is used in another API, this API control is not returned until it is completed.
- All settings and data that have been associated with the API handle specified by this API are discarded.

LockPrinter

Locks the data transmission and reset control to the printer.

INT **LockPrinter**(
INT *i_hdl*,
DWORD *i_timeout*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

i_timeout
Timeout period
Specifies the waiting period for success in msec (millisecond).
The range 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.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- Locks the data transmission and reset control to the printer by this API. To release it, use **UnlockPrinter**.
- 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 **UnlockPrinter** execution, it will fail.
- A lock by this API is valid within the process. Therefore, during the lock, 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 locked API handle is up to 99 times. To release it, execute **UnlockPrinter** the same times as for this API.

UnlockPrinter

Unlocks the data transmission and reset control to the printer.

INT **UnlockPrinter**(
INT *i_hdl*)

Parameter

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

Return value

On success:	Returns 0.
On failure:	Returns an error code. See "Chapter 5 Error Code List" for details of error code.

Remarks

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

DirectIO

Sends / receives binary data.

INT **DirectIO**(
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 **OpenMonPrinter**.

i_wlen
Size of data to send
Specifies the size of data to send.

i_wcmd
Buffer of data to send
Specifies the buffer that stored the data to send.

io_rlen
Size of data to receive
Specifies the maximum length of data to be received from the printer.
Specifies 0 when there is no need to get the data.
When the API returns, the size of the retrieved receive data is stored.

o_rbuf
Receive data buffer
Specifies the buffer that stores the data to get.

i_timeout
Timeout period
Specifies the waiting period for success in msec (millisecond).
The range 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.

i_flag
Receive operation flag
Specifies one of the following flags for the receive operation.
TRUE: Continues receiving until any data is received or timeout occurs.
FALSE: Continues receiving until the receive data size is received or timeout occurs.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- This API is aborted by **Reset**.
- For the commands, data and image data that do not allow interrupting of other data before completion of transmission, use this API after the calling **LockPrinter**. When **LockPrinter** is not called, the data from other process may interrupt.

- Receive data does not include responses for the ASB setting command. When responses including ones of the ASB setting command are retrieved, execute **DirectIOEx**.
- Do not include the data that disables the ASB setting command in this API. Otherwise, the API that gets printer status does not work properly.
- For Bluetooth connection, do not include a printer command "Hardware Reset" or "Printer Reset" in the data to send. Use **Reset** when hardware resetting the printer.

DirectIOEx

Sends / Receives binary data.

INT **DirectIOEx**(
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 **OpenMonPrinter**.

i_wlen
Size of data to send
Specifies the size of the data to send.

i_wcmd
Buffer of data to send
Specifies the buffer that stored the data to send.

io_rlen
Size of data to receive
Specifies the maximum length of data to be received from the printer.
The maximum receive data size is 4096 bytes.
When the specified value is more than 4096 bytes, it is corrected to 4096 bytes.
Specifies 0 when there is no need to get the data.
When the API returns, the size of the received data is stored.

o_rbuf
Buffer of data to receive
Specifies the buffer that stores the data to get.

i_timeout
Timeout period
Specifies the waiting period for success in msec (millisecond).
The range 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.

i_flag

Receive operation flag

Specifies one of the following flags for the receive operation.

TRUE: Continues receiving until any data is received or timeout occurs.

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

i_op

Receive target option

Specifies one of the following options for data to receive.

0: Gets the data excluding responses for the ASB setting command.

1: Gets the data including responses for the ASB setting command.

Return value

On success: Returns 0.

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

Remarks

- This API is aborted by **Reset**.
- For the commands, data and image data that do not allow interrupting of other data before completion of transmission, use this API after the calling **LockPrinter**. When **LockPrinter** is not called, the data from other process may interrupt.
- Do not include the data that disables the ASB setting command in this API. Otherwise, the API that gets 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. Use **Reset** when hardware resetting the printer.

Reset

Performs hardware reset of the printer.

INT **Reset**(
INT *i_hdl*)

Parameter

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- Performs hardware reset of the printer using the communication protocol (without using printer commands).
- When this API is called, the following APIs are aborted.
  - **DirectIO**
  - **DirectIOEx**
  - **DirectSendRead**
- After executing this API, wait for a few seconds to send the data. If data transmission is performed right after executing this API, it may cause data skipping.
- During execution of this API, the printer status responds "No response". See "6.1 Printer Status List" for details of the printer status.
- For Bluetooth connection, when executing this API in the state that the printer is unaccepting data, this API succeeds, but the reset is not executed until the printer is ready to print.
  And in the meantime, data transmission cannot be performed.

GetStatus

Gets the latest printer status.

INT **GetStatus**(
INT *i_hdl*,
LPDWORD *o_status*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

o_status
Printer status variable
Specifies the variable to store the printer status.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- When reconnection to the printer is detected, the printer status is the status received last at the time.
- See "6.1 Printer Status List" for details of the printer status.

SetStatusBackFunction

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

INT **SetStatusBackFunction**(
INT *i_hdl*,
INT (CALLBACK EXPORT **lpStatusCB*) (DWORD *o_st*))

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

lpStatusCB
Callback function address
Specifies the address of an application-defined callback function that receives the printer status.
When NULL is specified, monitoring of the printer status is canceled.

o_st
Printer status variable
Specifies the variable in which the printer status is stored.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- When the callback function is registered with this API, the callback function is called with the current printer status.
- The callback function registered by this API is unregistered by the following APIs.
  - **CancelStatusBack**
  - **CloseMonPrinter**

- APIs in the Win32 API cannot be called with the same API handle from the registered callback function.
- When reconnection to the printer is detected, the printer status is the status received last at the time.
- Even when the printer status is received, the callback function will not be called when the printer status has not changed from when it was last received.
- When this API is called in that state that the callback function is already registered, the registered function becomes invalid and new callback function is registered.
- Even when this API is called again specifying the already registered valid callback function, the printer status response is performed immediately after it.
- The return value of the callback function is ignored.
- The time between receiving the printer status and calling the callback function is not guaranteed.
- See "6.1 Printer Status List" for details of the printer status.

SetStatusBackWnd

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

INT **SetStatusBackWnd**(
INT *i_hdl*,
HANDLE *i_Wnd*,
LPDWORD *o_status*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

i_Wnd
Window handle
Specifies the window handle of the button to send the click event.
When NULL is specified, unregisters this API.

o_status
Printer status variable
Specifies the variable to store the printer status.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- This API is unregistered by the following APIs.
  - **CancelStatusBack**
  - **CloseMonPrinter**
- When reconnection to the printer is detected, the printer status is the status received last at the time.
- Even when the printer status is received, the click event will not be called when the printer status has not changed from when it was last received.
- When the window handle of the button is registered with this API, the click event is called with the current printer status.
- When this API is called in that state that the window handle of the button has been registered, the registered window handle becomes invalid and new window handle is registered.
- Even when this API is called again specifying the already registered valid window handle of a button, the printer status response is performed immediately after it.
- The return value of the click event is ignored.
- The time between receiving the printer status and calling the click event is not guaranteed.
- See "6.1 Printer Status List" for details of the printer status.

CancelStatusBack

Unregisters the callback function which was executed in **SetStatusBackWnd** and **SetStatusBackFunction.**

INT **CancelStatusBack**(
INT *i_hdl*)

Parameter

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- This API succeeds even when neither **SetStatusBackWnd** nor **SetStatusBackFunction** are registered.

PowerOff

Turns the printer power off.

INT **PowerOff**(
INT *i_hdl*)

Parameter

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

GetCounter

Gets the maintenance counter.

INT **GetCounter**(
INT *i_hdl*,
WORD *i_readno*,
LPDWORD *o_readcounter*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

i_readno
Counter ID
Specifies the counter ID to get.
See "6.2 Counter ID" for possible values for specification.

o_readcounter
Counter variable
Specifies the variable to store the counter value to get.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

ResetCounter

Initializes the maintenance counter.

INT **ResetCounter**(
INT *i_hdl*,
WORD *i_readno*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

i_readno
Counter ID
Specifies the counter ID to initialize.
See "6.2 Counter ID" for possible values for specification.

Return value

On success:	Returns 0.
On failure:	Returns an error code. See "Chapter 5 Error Code List" for details of error code.

Remarks

- Confirm that the counter ID value specified by this API is initialized by using **GetCounter**.

GetType

Gets the various IDs of the printer.

INT **GetType**(
INT *i_hdl*,
LPBYTE *o_typeID*
LPBYTE *o_fontID*
LPBYTE *o_exrom*
LPBYTE *o_special*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

o_typeID
ID1
1 (fixed) is stored.

o_fontID
ID2
2 (fixed) is stored.

o_exrom
Reserved
Specify NULL.

o_special
Reserved
Specify NULL.

Return value

On success: Returns 0.

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

GetPrnCapability

Gets the printer information.

INT **GetPrnCapability**(
INT *i_hdl*,
BYTE *i_id*,
LPBYTE *io_datsize*,
LPBYTE *o_dat*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

i_id
Printer ID
Specifies the printer ID to get.
See "6.3 Printer ID" for possible values for specification.

io_datsize
Size of data to receive
Specifies the size of the buffer to store the printer ID to get.
When the API returns, retrieved size of data to receive is stored.
When the specified buffer size is smaller than the response size of the retrieved printer ID, this API fails and the response size of the retrieved printer ID is stored.

o_dat
Buffer of data to receive
Specifies the buffer to store the value of printer ID to get.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

SendDataFile

Registers the contents of the command definition file to the memory in the SDK.

INT **SendDataFile**(
INT *i_hdl*,
LPCTSTR *i_fname*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

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

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

Remarks

- See "Chapter 7 Command Definition File" for details of the command definition file.
- The command data registered by this API is discarded by **CloseMonPrinter**.

DirectSendRead

Executes a command registered by **SendDataFile**.

INT **DirectSendRead**(
INT *i_hdl*,
LPCTSTR *i_cname*,
LPCTSTR *i_rtype*,
LPDWORD *io_rlen*,
LPBYTE *o_rbuf*,
DWORD *i_timeout*,
BOOL *i_flag*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

i_cname
Command name
Specifies a command name registered by **SendDataFile**.

i_rtype

Receive data type name

Specifies one of the following types for data to receive.
ASB: Stores only the responses for 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.
The maximum receive data size is 4096 bytes.
When the specified value is more than 4096 bytes, it is corrected to 4096 bytes.
Specifies 0 when there is no need to get the data.
When the API returns, the size of the retrieved receive data is stored.

o_rbuf

Buffer of data to receive

Specifies the buffer that stores the data to get.

i_timeout

Timeout period

Specifies the waiting period for success in msec (millisecond).
The range 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.

i_flag

Receive operation flag

Specifies one of the following flags for the receive operation.
TRUE: Continues receiving until any data is received or timeout occurs.
FALSE: Continues receiving until the receive data size is received or timeout occurs.

Return value

On success: Returns 0.

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

Remarks

- Do not include the data that disables the ASB setting command in this API. Otherwise, the API that gets printer status does not work properly.
- This API is aborted by **Reset**.

GetProperty

Gets a part of the print setting contents.

INT **GetProperty**(
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 **OpenMonPrinter**.

i_devmode
Devmode address
Specifies the Devmode address.

i_pid
Property ID
Specifies the property ID to get.
See "6.4 Property ID" for possible values for specification.

o_dat
Buffer of data to get
Specifies the buffer to store the content of the property ID to get.
When specifies NULL, this API fails and the response size of the specified printer ID is stored to *io_size*.

io_size
Size of data to get
Specifies the maximal length of data to get.
When the API returns, the size of the retrieved data is stored.
When the specified size is smaller than the response size of the property ID, this API fails and the response size of the specified property ID is stored.

Return value

On success: Returns 0.
On failure: Returns an error code.
See "Chapter 5 Error Code List" for details of error code.

SetProperty

Changes a part of the print setting contents.

INT **SetProperty**(
INT *i_hdl*,
LPDEVMODE *i_devmode*,
BYTE *i_pid*,
LPBYTE *i_dat*,
LPDWORD *i_size*)

Parameters

i_hdl
API handle
Specifies the API handle retrieved by **OpenMonPrinter**.

i_devmode
Devmode address
Specifies the Devmode address.

i_pid
Property ID
Specifies the property ID to change.
See "6.4 Property ID" for possible values for specification.

i_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

On success: Returns 0.

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

Chapter 4 .NET API

This chapter describes the overview of the .NET API.

4.1 Development Language

The following development languages are covered.

- Visual Basic .NET
- Visual C#

4.2 Library File

The library file of .NET API is a class library format.

The library file has the following file name.

- SiiMpb2ClassLib.dll

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

4.3 API List

The following APIs are implemented in the .NET API.

- Namespace: Sii.SDK.MPosPrinter
- Class name: StatusAPI

Category	API	Function Summary
Property	**Status**	Gets the latest printer status.
Property	**LastError**	Gets the error value of the last executed API.
Property	**IsValid**	Gets the call status of **OpenMonPrinter**.
Method	**OpenMonPrinter**	Starts using the .NET API.
Method	**CloseMonPrinter**	Ends using the .NET API.
Method	**LockPrinter**	Locks the data transmission and reset control to the printer.
Method	**UnlockPrinter**	Unlocks the data transmission and reset control to the printer.
Method	**DirectIOEx**	Sends / Receives binary data.
Method	**ResetPrinter**	Performs hardware reset of the printer.
Method	**SetStatusBack**	Starts the notification of the printer status by **StatusCallback** event.
Method	**CancelStatusBack**	Stops the notification of the printer status by **StatusCallback** event.
Method	**PowerOff**	Turns the printer power off.
Method	**GetCounter**	Gets the maintenance counter.
Method	**ResetCounter**	Initializes the maintenance counter.
Method	**GetType**	Gets the various IDs of the printer.
Method	**GetPrnCapability**	Gets the printer information.
Method	**SendDataFile**	Registers the contents of the command definition file to the memory in the SDK.
Method	**DirectSendRead**	Executes a command registered by **SendDataFile**.
Method	**GetProperty**	Gets a part of the print setting contents.
Method	**SetProperty**	Changes a part of the print setting contents.
Event	**StatusCallback**	Notifies the returned printer status.

4.4 API Details

Caution

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

4.4.1 Property

Status

Gets the latest printer status.

SII.SDK.MposPrinter.ASB Status { get; }

Initial value

ASB.NO_RESPONSE

Remarks

- When reconnection to the printer is detected, the printer status is the status received last at the time.
- When **isValid** is FALSE, a correct value is not available.
- See "6.1 Printer Status List" for details of the printer status.

LastError

Gets the error value of the last executed API.

SII.SDK.MposPrinter.ErrorCode LastError { get; }

Initial value

SUCCESS

Remarks

- See "Chapter 5 Error Code List" for details of error code.

IsValid

Gets the call status of **OpenMonPrinter**.

bool **IsValid** { get; }

Initial value

FALSE

Remarks

- TRUE: On success for **OpenMonPrinter**.
- FALSE: Out of success for **OpenMonPrinter**.

4.4.2 Method

OpenMonPrinter

Starts using the .NET API.

ErrorCode **OpenMonPrinter**(
OpenType *type*,
string *name*)

Parameters

type
Open type
OpenType.TYPE_PRINTER (fixed)

name
Name of the printer that uses the .NET API
Specifies the printer name (friendly name).

Return value

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

Remarks

- The number of API handle that can be retrieved by 1 process is 1.
- When the .NET API is not used, be sure to call **CloseMonPrinter**.
- When the printer driver connects except for USB or Bluetooth, 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 .NET API.

ErrorCode **CloseMonPrinter**()

Return value

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

Remarks

- All settings and data that have been associated with the API specified by this API are discarded.

LockPrinter

Locks the data transmission and reset control to the printer.

ErrorCode **LockPrinter**(
int *timeout*)

Parameters

timeout
Timeout period
Specifies the waiting period for success in msec (millisecond).
The range 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.

Return value

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

Remarks

- Locks the data transmission and reset control to the printer by this API. To release it, use **UnlockPrinter**.
- The number of repetitions this API is up to 99 times. To release it, execute **UnlockPrinter** the same times as for this API.

UnlockPrinter

Unlocks the data transmission and reset control to the printer.

ErrorCode **UnlockPrinter**()

Return value

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

Remarks

- When **LockPrinter** is called more than one time, this API must be called the same time as for **LockPrinter** to release the lock.

DirectIOEx

(a) API: After sending binary data, gets the receive data as binary data from the printer.
(b) API: After sending binary data, gets the receive data as string data from the printer.
(c) API: Sends binary data.

(a) ErrorCode **DirectIOEx**(
byte[] *cmd*,
ref byte[] *data*,
int *timeout*,
bool *readFlag*,
byte *option*)

(b) ErrorCode **DirectIOEx**(
byte[] *cmd*,
out string *data*,
int *timeout*,
byte *option*)

(c) ErrorCode **DirectIOEx**(
byte[] *cmd*,
int *timeout*)

Parameters

cmd
Buffer of data to send
Specifies the buffer that stored the data to send.

data
Buffer of data to receive
Specifies the buffer that stores the data to get.
The maximum receive data size is 4096 bytes.
When the specified value is more than 4096 bytes, it is corrected to 4096 bytes.
Specifies 0 when there is no need to get the data.
When the API returns, the size of the received data is stored.

timeout
Timeout period
Specifies the waiting period for success in msec (millisecond).
The range 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.

readFlag
Receive operation flag
Specifies one of the following flags for the receive operation.
TRUE: Continues receiving until any data is received or timeout occurs.
FALSE: Continues receiving until the receive data size is received or timeout occurs.

option
Receive target option
Specifies one of the following options for data to receive.
0: Gets the data excluding responses for the ASB setting command.
1: Gets the data including responses for the ASB setting command.

Return value

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

Remarks

- This API is aborted by **ResetPrinter**.
- Do not include the data that disables the ASB setting command in this API. Otherwise, the API that gets 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. Use **ResetPrinter** when hardware resetting the printer.
- 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.

ResetPrinter

Performs hardware reset of the printer.

ErrorCode **ResetPrinter**()

Return value

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

Remarks

- Performs hardware reset of the printer using the communication protocol (without using printer commands).
- When this API is called, the following APIs are aborted.
  - **DirectIOEx**
  - **DirectSendRead**
- After executing this API, wait for a few seconds to send the data. If data transmission is performed right after executing this API, it may cause data skipping.
- During execution of this API, the printer status responds "No response". See "6.1 Printer Status List" for details of the printer status.

SetStatusBack

Starts the notification of the printer status by **StatusCallback** event.

ErrorCode **SetStatusBack**()

Return value

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

Remarks

- **StatusCallback** event started by this API is stopped by the following APIs:
  - **CancelStatusBack**
  - **CloseMonPrinter**
- When the event handler is not registered on **StatusCallback** event, this API fails.
- When reconnection to the printer is detected, the printer status is the status received last at the time.
- When the notification of the printer status is started by this API, **StatusCallback** event is raised with the current printer status.
- When this API is called in that state that the callback function is already registered, the registered method becomes invalid and new callback function is registered.
- Even when **StatusCallback** event is already started and this API is called again, the immediate response of the printer status is performed.
- The time from receiving the printer status to raising the event is not guaranteed.
- See "6.1 Printer Status List" for details of the printer status.

CancelStatusBack

Stops the notification of the printer status by **StatusCallback** event.

ErrorCode **CancelStatusBack**()

Return value

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

Remarks

- This API succeeds even when the notification of the printer status has not been started by **SetStatusBack**.

PowerOff

Turns the printer power off.

ErrorCode **PowerOff**()

Return value

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

Remarks

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

GetCounter

Gets the maintenance counter.
(a) API: Gets the maintenance counter with the counter ID defined in CounterIndex.
(b) API: Gets the maintenance counter by specifying ID.

(a) ErrorCode **GetCounter**(
CounterIndex *index*,
bool *type*,
out int *data*)

(b) ErrorCode **GetCounter**(
byte *index*,
out int *data*)

Parameters

index
Counter ID
Specifies the counter ID to get.
Specify (a) when using the counter ID defined in SII.SDK.MposPrinter.CounterIndex.
Specify (b) when specifying the counter ID.
See "6.2 Counter ID" for possible values for specification.

type
Type of maintenance counter
One of the following types of maintenance counter to get:
TRUE: Integrated counter
FALSE: Initializable counter

data
Counter variable
Specifies the variable to store the counter value to get.

Return value

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

ResetCounter

Initializes the maintenance counter.
(a) API: Initializes the maintenance counter with the counter ID defined in CounterIndex.
(b) API: Initializes the maintenance counter by specifying ID.

(a) ErrorCode **ResetCounter**(
CounterIndex *index*)

(b) ErrorCode **ResetCounter**(
byte *index*)

Parameters

index
Counter ID
Specifies the counter ID to initialize.
Specify (a) when using the counter ID defined in SII.SDK.MposPrinter.CounterIndex.
Specify (b) when specifying the counter ID.
See "6.2 Counter ID" for possible values for specification.

Return value

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

Remarks

- Confirm that the counter ID value specified by this API is initialized by using **GetCounter**.

GetType

Gets the various IDs of the printer.

ErrorCode **GetType**(
out byte *typeId*,
out byte *fontID*,
out byte *exrom*,
out byte *special*)

Parameter

typeId
ID1
1 (fixed) is stored.

fontID
ID2
2 (fixed) is stored.

exrom
Reserved
Specify NULL.

special
Reserved
Specify NULL.

Return value

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

GetPrnCapability

Gets the printer information.
(a) API: Gets the response data as binary data.
(b) API: Gets the response data as string data.

(a) ErrorCode **GetPrnCapability**(
byte *id*,
out byte[] *data*)

(b) ErrorCode **GetPrnCapability**(
byte *id*,
out string *data*)

Parameters

id

Printer ID

Specifies the printer ID to get.
Specify (a) when the response format of the printer ID to be retrieved is HEX code.
Specify (b) when the response format of the printer ID to be retrieved is ASCII character strings.
See "6.3 Printer ID" for possible values for specification and response format.

data

Buffer of data to receive

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

Return value

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

SendDataFile

Registers the contents of the command definition file to the memory in the SDK.

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

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

Remarks

- See "Chapter 7 Command Definition File" for details of the command definition file.
- The command data registered by this API is discarded by **CloseMonPrinter**.

DirectSendRead

(a) API: After executing **SendDataFile**, gets the receive data as binary data from the printer.
(b) API: After executing **SendDataFile**, gets the receive data as string data from the printer.
(c) API: Executes a command registered by **SendDataFile**.

(a) ErrorCode **DirectSendRead**(
string *cmdName*,
string *readType*,
ref byte[] *data*,
int *timeout*,
bool *readFlag*)

(b) ErrorCode **DirectSendRead**(
string *cmdName*,
string *readType*,
out *string data*,
int *timeout*)

(c) ErrorCode **DirectSendRead**(
string *cmdName*,
string *readType*,
int *timeout*)

Parameters

cmdName

Command name

Specifies a command name defined by **SendDataFile**.

readType

Receive data type name

Specifies one of the following types for receive data.
ASB: Stores only the responses for 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 that stores the data to get.

timeout

Timeout period

Specifies the waiting period for success in msec (millisecond).
The range 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.

readFlag

Receive operation flag

Specifies one of the following flags for the receive operation.
TRUE: Continues receiving until any data is received or timeout occurs.
FALSE: Continues receiving until the receive data size is received or timeout occurs.

Return value

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

Remarks

- Do not include the data that disables the ASB setting command in this API. Otherwise, the API that gets printer status does not work properly.
- This API is aborted by **ResetPrinter**.
- For Bluetooth connection, do not include a printer command "Hardware Reset" or "Printer Reset" in the binary data to send. Use **ResetPrinter** when hardware resetting the printer.
- Any 0x02 included in the receive data is converted to 0x5f.

GetProperty

Gets a part of the print setting contents.

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 get.
See "6.4 Property ID" for possible values for specification.

data
Buffer of data to get
Specifies the buffer to store the content of the property ID to get.

size
Size of data to get
Specifies the maximal length of data to get.
When the API returns, the size of the retrieved data is stored.

Return value

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

SetProperty

Changes a part of the print setting contents.

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.
See "6.4 Property ID" for possible values for specification.

data
Buffer of data to set
Specifies the buffer that stored the content of the property ID to change.

size

Size of data to set

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

Return value

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

4.4.3 Event

StatusCallback

Notifies the returned printer status.

event StatusCallbackHandler **StatusCallback**

delegate void **StatusCallbackHandler**(
ASB *status*)

Parameters

status
Printer status variable
Specifies the variable to store the printer status.
See "6.1 Printer Status List" for details of the printer status.

Remarks

- To start the notification of the printer status, call **SetStatusBack**.
- When the notification of the printer status is started by **SetStatusBack**, the event is raised with the current printer status.
- Register the event handler to handle the response printer status before starting the notification of the printer status.
- When the event handler is not registered, **SetStatusBack** fails.
- Call the following APIs to stop this API.
  - **CancelStatusBack**
  - **CloseMonPrinter**
- APIs in the .NET API cannot be called with the same instance from the registered event handler.
- When reconnection to the printer is detected, the printer status is the status received last at the time.
- Even when the printer status is received, the event is not raised when the printer status has not changed from when it was last received.
- The time from receiving the printer status to raising the event is not guaranteed.
- See "6.1 Printer Status List" for details of the printer status.

Chapter 5 Error Code List

This chapter describes the error codes.

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 being 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.

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

Chapter 6 Argument Information

This chapter describes the arguments.

6.1 Printer Status List

The followings are responses for the printer status. In addition, please refer to the printer status processing of our sample program.

Printer status	Response Value		Description
Voltage error	ASB.VP_ERR	0x00000000	No voltage error
		0x00000001	Voltage error
Hardware error	ASB.HARDWARE_ERR	0x00000000	No hardware error
		0x00000002	Hardware error
Head temperature error	ASB.HEAD_TEMPERATUR_ERR	0x00000000	No head temperature error
		0x00000004	Head temperature error
Out-of-paper error	ASB.RECEIPT_END	0x00000000	No out-of-paper error
		0x00000010	Out-of-paper error
FEED switch state	ASB.PAPER_FEED	0x00000000	FEED switch status = "Off"
		0x00000100	FEED switch status = "On"
Paper feed state	ASB.NOW_PRINTING	0x00000000	Stop
		0x00000400	Operating
Recovery waiting status	ASB.RETURN_WAITING	0x00000000	-
		0x00000800	Recovery waiting status
FLASH memory rewriting	ASB.FLASH_MEMORY_REWRITING	0x00000000	-
		0x00010000	FLASH memory rewriting

Printer status	Response Value		Description
Battery remaining capacity level *1	ASB.BATTERY_LOW	0x00100000	Low: 0x00100000
	ASB.BATTERY_MIDDLE	0x00200000	Middle: 0x00300000
	ASB.BATTERY_FULL	0x00400000	Full: 0x00700000
Battery error	ASB.BATTERY_ERR	0x00000000	No battery error
		0x00800000	Battery error
Automatic recovery error	ASB.AUTORECOVER_ERR*2	0x00000000	No automatic recovery error
		0x20000000	Automatic recovery error
Unrecoverable error	ASB.UNRECOVER_ERR*2	0x00000000	No unrecoverable error
		0x40000000	Unrecoverable error
No response	ASB.NO_RESPONSE*2 *3	0x00000000	Printer responds
		0x80000000	Disconnection or communication error

*1: Verify the response value from Full -> Middle -> Low in order. (For example, in Full, the response values of Middle and Low are set.) When all of these 3 response values are cleared, this means that the battery remaining capacity is none.
*2: Extended status to the responses for the ASB setting command.
*3: For wireless communication, detection of "No response" may take some time.

6.2 Counter ID

There are the following responses for the counter ID.

Counter ID		Description	ResetCounter / Initialization by ResetCounter
ROLL_FEED_LINES	20	Number of dot lines for paper feed (unit: 100-dot line)	Enable
ROLL_HEAD_CHARGE	21	Thermal head activation time (unit: 100-dot line)	Enable
OPERATION_TIME	70	Product drive time (unit: minute)	Enable
ROLL_FEED_LINES	148	Number of dot lines for paper feed (unit: 100-dot line)	Disable (Integrated value)
ROLL_HEAD_CHARGE	149	Thermal head activation time (unit: 100-dot line)	Disable (Integrated value)
OPERATION_TIME	198	Product drive time (unit: minute)	Disable (Integrated value)

6.3 Printer ID

There are the following printer IDs with their responses.

Printer ID	Description		Response Format
1	Printer model ID	0x21	HEX code
2	Type ID	0x30	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 MP-B20 Series.	ASCII string
69	Multi-language font type	(For Japanese) KANJI JAPANESE	ASCII string
97	Firmware version (boot)	x.xx.xx	ASCII string
98	Firmware checksum (boot)	2 bytes checksum	HEX code
99	Firmware checksum (main)	2 bytes checksum	HEX code
100	Firmware checksum (main + boot)	2 bytes checksum	HEX code

6.4 Property ID

Caution

- Specify the data size to 1 byte except for custom command for using **SetProperty**.
- The phrase in square brackets ([ ]) indicates the timing of corresponding process.

There are the following property IDs with their contents.

Property ID	Description	
1	Initialize	0: Enabled 1: Disabled
3	Margins	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 20 to 100: Scale (percentage)
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
15	[Print Start] Custom Command (128 bytes)	Command data
17*1	[Print Start] Paper feed (feed) (dot)	0 to 255
20	[Page Start] Logo	0: Disabled 1: Left 2: Center 3: Right
21	[Page Start] Logo Keycode	0 to 99
25	[Page Start] Custom Command (128 bytes)	Command data
27*1	[Page Start] Paper feed (feed) (dot)	0 to 255
30	[Page End] Logo	0: Disabled 1: Left 2: Center 3: Right

Property ID	Description	
31	[Page End] Logo Keycode	0 to 99
35	[Page End] Custom Command (128 bytes)	Command data
37*1	[Page End] Paper feed (feed) (dot)	0 to 255
40	[Print End] Logo	0: Disabled 1: Left 2: Center 3: Right
41	[Print End] Logo Keycode	0 to 99
45	[Print End] Custom Command (128 bytes)	Command data
47*1	[Print End] Paper feed (feed) (dot)	0 to 255
50	Paper Size	0: Letter 1: A4 3: 58 mm 4 to 255: Paper except for the above*2
51	Orientation	0: Portrait 1: Landscape
52	Color Printing Mode	0: System*3 1: Driver*3
60*4	Preset	1: 58 mm Receipt Setting 5: A4->58 mm Reduction Setting 6: User Setting

*1: When setting feed same timing, the last setting is valid.
*2: Cannot be set. Only get the value.
*3: For details, see "3.2 Paper/Quality" in "SII Printer Driver for Windows User's Guide" for MP-B20 Series.
*4: When specified this property, some data of other property IDs is ignored. For details, see "3.4.4 Use of [Preset]" in "SII Printer Driver for Windows User's Guide" for MP-B20 Series.

Chapter 7 Command Definition File

7.1 Overview

Command definition file is the text file (*.txt) which transmission data is described in the predefined format and saved in the ANSI format or UNICODE format.

7.2 Format

This section describes the command definition file format.

Command definition

[Command name 1] = [Transmission data 1]#[Comment 1]
[Command name 2] = [Transmission data 2]#[Comment 2]
[Command name 3] = [Transmission data 3]#[Comment 3]
:
:

Name		Description
Command definition		The command name unit data specified based on the format. Be sure to specify one command definition in one line.
	Command name	Specify the arbitrary command name designated by **DirectSendRead**. ・ Use ASCII characters except for "=" and "#" . When characters other than ASCII characters are specified, they are ignored. ・ Command name is case-sensitive. ・ Registerable command name is up to 33 bytes. When 34 bytes or more is specified, up to 33 bytes are registered, and subsequent characters are ignored. ・ When specifies same command name in the command definition file, the first command name is registered, When command name which is already registered is specified, it is ignored. ・ Specify "=" between the command name and the transmission data.

Name		Description
	Transmission data	Specifies the printer command and data to send to the printer. ・ When use binary data as transmission data, specify it in two-digit hexadecimal numbers. Separate each number with a 1-byte space. Do not include the printer command "Hardware Reset" or "Printer Reset" in the binary data. Use **Reset** or **ResetPrinter** when hardware resetting the printer. ・ Specify ASCII characters to use strings as transmission data. When specifies strings, enclose them with "". ・ The maximum transmission data size is 10240 bytes. However, when use strings as transmission data, they are converted to binary data to determine the actual size. The transmission data size of the command definitions as an example shown below are as follows: ・ CmdName_1: 3 bytes ・ CmdName_2: 4 bytes ・ CmdName_3: 3 bytes ・ CmdName_4: 6 bytes
	Comment	Describe the description of the command definition etc. ・ Add "#" at the beginning of the comment to describe. ・ There is no limitation of characters described. ・ Comments are optional.

Examples Command definition

```
CmdName_1="SII"#Comments1
CmdName_2=53 49 49 0A
CmdName_3=1D 56 00#Feed the paper to cut position.
CmdName_4=1B 40 "SII" 0A#Initialize the printer, the strings "SII" and line feed.
```

Caution

- ◆ The maximum command definition file size is 4GB.
- ◆ The number of registerable command definition depends on the available system memory.

7.3 How to Use

This section describes how to use the command definition file.

1. Create a command definition file.
2. Register the command definition file by **SendDataFile** to the memory in the SDK.
3. When specifies the command name by **DirectSendRead**, the contents of the transmission data is executed.

Caution

- The registered command data is discarded by **CloseMonPrinter**.
- After registration of the command definition file, when a new command definition file is registered, the content of the command definition file is overwritten.

Seiko Instruments Inc.
1-8, Nakase, Mihama-ku, Chiba-shi,
Chiba 261-8507, Japan
Print System Division
Telephone:+81-43-211-1106
Facsimile:+81-43-211-8037

Seiko Instruments USA Inc.
Thermal Printer Div.
21221 S. Western Avenue, Suite 250, Torrance, CA 90501, USA
Telephone:+1-310-517-7778 Facsimile:+1-310-517-7779

Seiko Instruments GmbH
Siemensstrasse 9, D-63263 Neu-lsenburg, Germany
Telephone:+49-6102-297-0 Facsimile:+49-6102-297-222
info@seiko-instruments.de

Seiko Instruments (H.K.) Ltd.
4-5/F, Wyler Center 2,200 Tai Lin Pai Road, Kwai Chung, N.T., Kowloon, Hong Kong
Telephone:+852-2494-5160 Facsimile:+852-2424-0901