



# SII SDK for Windows Application Programmer's Guide

Rev.01

## [Products]

RP-F10 Series

RP-G10 Series

RP-E10 Series

Seiko Instruments Inc.


Copyright©2024 by Seiko Instruments Inc.  
All rights reserved.

Microsoft® and Windows® are registered trademarks of Microsoft Corporation in the U.S., Japan, and other countries.

Bluetooth® is a registered trademark of Bluetooth SIG, Inc.

Seiko Instruments Inc. (hereinafter referred to as "SII") has prepared this manual for use by SII personnel, licensees, and customers. The information contained herein is the property of SII and shall not be reproduced in whole or in part without the prior written approval of SII.

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.

**SII**  is a trademark of Seiko Instruments Inc.

---

# Introduction

---

This manual describes "SII SDK for Windows" (hereinafter referred to as the "SDK") provided by Seiko Instruments Inc. (hereinafter referred to as "SII").  
The SDK runs on the printer driver stated in "Target Products".

## Target Products

---

The products covered by this manual are as follows.

Printer Mechanism	Printer Driver
RP-F10 Series RP-G10 Series RP-E10 Series	"SII Printer Driver for Windows"

# Notation in This Manual

---

The notation in this manual is described.

## Terms

The terms used in this manual are defined as below.

Term	Description
Technical Reference	Technical References shown as follows: ·RP-F10 SERIES THERMAL PRINTER TECHNICAL REFERENCE ·RP-G10 SERIES THERMAL PRINTER TECHNICAL REFERENCE ·RP-E10 SERIES THERMAL PRINTER TECHNICAL REFERENCE
Printer command	Instruction to control the printer, described in "Technical Reference".
ASB setting command (ASB: Automatic Status Back)	Printer command "Enable/Disable Automatic Status Back" See the chapter of "Command Functions" in "Technical Reference" for details of printer commands.
Printer status	Printer status information that can be retrieved by the SDK This information includes the status to respond to the printer command "Automatic Status Back Enable/Disable" and extended status. See "6.1 Printer Status List" for details of the printer status.

<b>Chapter 1</b>	<b>Overview</b>	<b>1-1</b>
1.1	Operating Conditions	1-1
<b>Chapter 2</b>	<b>Installation</b>	<b>2-1</b>
<b>Chapter 3</b>	<b>Win32 API</b>	<b>3-1</b>
3.1	Development Language	3-1
3.2	Library File	3-1
3.3	API List	3-2
3.4	API Details	3-4
	OpenMonPrinter	3-4
	CloseMonPrinter	3-5
	LockPrinter	3-6
	UnlockPrinter	3-7
	DirectIO	3-7
	DirectIOEx	3-9
	Reset	3-11
	GetStatus	3-12
	SetStatusBackFunction	3-13
	SetStatusBackWnd	3-14
	CancelStatusBack	3-15
	SetBarcodeDataBackFunction	3-16
	CancelBarcodeDataBack	3-18
	PowerOff	3-18
	GetCounter	3-19
	ResetCounter	3-20
	GetType	3-21
	GetPrnCapability	3-22
	OpenDrawer	3-23
	ControlFeature	3-24
	SendDataFile	3-25
	DirectSendRead	3-26
	GetProperty	3-27
	SetProperty	3-28
<b>Chapter 4</b>	<b>.NET API</b>	<b>4-1</b>
4.1	Development Language	4-1

4.2	Library File .....	4-1
4.3	Printer .....	4-2
4.3.1	API List.....	4-2
	Common API .....	4-2
	Specific API .....	4-2
4.3.2	Common Property .....	4-4
	LastError .....	4-4
	IsValid .....	4-4
4.3.3	Common Method.....	4-6
	OpenMonPrinter.....	4-6
	CloseMonPrinter .....	4-7
4.3.4	Specific Property .....	4-8
	Status .....	4-8
4.3.5	Specific Method.....	4-9
	LockPrinter.....	4-9
	UnlockPrinter.....	4-10
	DirectIOEx.....	4-10
	ResetPrinter .....	4-12
	SetStatusBack.....	4-13
	CancelStatusBack.....	4-13
	PowerOff .....	4-14
	GetCounter .....	4-14
	ResetCounter .....	4-15
	GetType .....	4-16
	GetPrnCapability .....	4-17
	OpenDrawer.....	4-18
	ControlFeature .....	4-19
	SendDataFile .....	4-20
	DirectSendRead.....	4-20
	GetProperty.....	4-22
	SetProperty .....	4-23
4.3.6	Event.....	4-24
	StatusCallback .....	4-24
4.4	Display .....	4-26
4.5	Barcode Scanner .....	4-27
4.5.1	API List.....	4-27
	Common API .....	4-27
	Specific API .....	4-27
4.5.2	Common Property .....	4-28

LastError .....	4-28
IsValid .....	4-28
4.5.3 Common Method .....	4-30
OpenMonPrinter .....	4-30
CloseMonPrinter .....	4-31
4.5.4 Specific Property .....	4-32
SetBarcodeDataBack .....	4-32
CancelBarcodeDataBack .....	4-33
4.5.5 Event .....	4-34
BarcodeDataCallback .....	4-34

---

<b>Chapter 5</b>	<b>Error Code List</b>	<b>5-1</b>
------------------	------------------------	------------

---

5.1 Error Code List .....	5-1
---------------------------	-----

---

<b>Chapter 6</b>	<b>Argument Information</b>	<b>6-1</b>
------------------	-----------------------------	------------

---

6.1 Printer Status List .....	6-1
6.2 Counter ID .....	6-2
6.3 Printer ID .....	6-3
6.4 Type ID .....	6-4
6.5 Target Drawers .....	6-4
6.6 Drawer Drive Time .....	6-5
6.7 Printer Function .....	6-5
6.8 Property ID .....	6-6
6.9 Barcode Data Structure .....	6-12
6.9.1 Win32 API .....	6-12
6.9.2 .NET API .....	6-13

---

<b>Chapter 7</b>	<b>Command Definition File</b>	<b>7-1</b>
------------------	--------------------------------	------------

---

7.1 Overview .....	7-1
7.2 Format .....	7-1
7.3 How to Use .....	7-3

---

# 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 to design the application independent of the port type. Also, some values of members in the private section of the DEVMODE can be retrieved or changed.

The SDK includes the following library files:

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

For usage sample of the SDK, see the sample program provided for each development language.

## 1.1 Operating Conditions

---

The operating conditions of the SDK basically follow the operating environment of the printer driver, and the settings and limitations of the printer. For details of the operating environment of the printer driver, see "SII Printer Driver for Windows User's Guide".

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 spool function is disabled.
- If all 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.)
  - When using RP-E10.
  - When using a single printer from multiple hosts through the TCP/IP connection.
  - When the data to send is divided more than once to send.

---

# Chapter 2 Installation

---

For the installation, see "SII Software Package for Windows Installation Guide".

---

# Chapter 3 Win32 API

---

This chapter describes the Win32 API.

## 3.1 Development Language

---

The following development language is covered.

- Visual C++

## 3.2 Library File

---

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

The library file has the following file name:

- SPSWL\_API.dll

The library file is stored in the Windows system folder.

Use the library file without moving it from the folder. Except in special cases, there is no need to set a path to the folder where the library file is saved.

If the library file is moved to another location, the library file will not be updated properly when upgrading the printer driver.

## 3.3 API List

The APIs implemented in the Win32 API are as follows.

API	Function Summary
<b>OpenMonPrinterA</b> *1 <b>OpenMonPrinterW</b> *1	Starts using the Win32 API and returns the API handle.
<b>CloseMonPrinter</b>	Ends using the Win32 API.
<b>LockPrinter</b>	Locks all data transmission and hardware reset requests from other processes to the printer.
<b>UnlockPrinter</b>	Unlocks the access prohibition (lock) from other processes by <b>LockPrinter</b> .
<b>DirectIO</b>	Sends and receives binary data. (Receive data does not include responses of the ASB setting command.)
<b>DirectIOEx</b>	Sends and receives binary data.
<b>Reset</b>	Performs hardware reset of the printer.
<b>GetStatus</b>	Gets the latest printer status.
<b>SetStatusBackFunction</b>	Registers the callback function called when a change in the printer status is detected.
<b>SetStatusBackWnd</b>	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 in the printer status is detected.
<b>CancelStatusBack</b>	Cancels the registration of the callback function called by <b>SetStatusBackWnd</b> or <b>SetStatusBackFunction</b> .
<b>SetBarcodeDataBackFunction</b>	Registers the callback function of the barcode scanner.
<b>CancelBarcodeDataBack</b>	Cancels the registration of the callback function called by <b>SetBarcodeDataBackFunction</b> .
<b>PowerOff</b>	Turns off the printer power.
<b>GetCounter</b>	Gets the maintenance counter.
<b>ResetCounter</b>	Initializes the maintenance counter.
<b>GetType</b>	Gets various IDs of the printer.
<b>GetPrnCapability</b>	Gets the printer information.
<b>OpenDrawer</b>	Drives the specified drawer.
<b>ControlFeature</b>	Controls functions of the printer other than printing.
<b>SendDataFileA</b> *1 <b>SendDataFileW</b> *1	Registers the contents of the command definition file in the SDK internal memory.
<b>DirectSendReadA</b> *1 <b>DirectSendReadW</b> *1	Sends transmission data corresponding to the specified command name, with respect to the command definition file registered by <b>SendDataFile</b> .
<b>GetProperty</b>	Gets a part of the print settings.
<b>SetProperty</b>	Changes a part of the print settings.

\*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 the suffix of 'A' or 'W' is omitted in the following descriptions.

## Caution

- ◆ See "RP-F10 SERIES Thermal Printer USER'S GUIDE" for details of the recommended barcode scanner and the barcode scanner setting.

## Reference

- For APIs of Display, see "SII SDK for Windows Application Programmer's Guide" for DSP-A01 for details.
- RP-G10 and RP-E10 do not support the barcode scanner or Display.

## 3.4 API Details

---

### Caution

- ◆ With Bluetooth connection, the response data while the printer is not connected cannot be retrieved.

## OpenMonPrinter

Starts using the Win32 API and returns the API handle.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

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

### Parameters

*i\_type*

Open type

For using the printer alone: 2

For using the printer connected to the barcode scanner: 4

*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 "5.1 Error Code List" for details of the error code.

### Remarks

- The number of API handles that can be retrieved simultaneously in 1 process is up to 8. The number is up to 126 in total processes.
- When the API handle retrieved by this API is no longer used, be sure to disable it by **CloseMonPrinter**.

- Set the connection destination of the printer driver to USB, Serial (including the COM port of the Bluetooth device), or TCP/IP.
- Specify 4 with *i\_type* when using **SetBarcodeDataBackFunction** or **CancelBarcodeDataBack**.
- This API succeeds even when the printer is not connected or the printer power is turned off.
- This API succeeds even when the barcode scanner is not connected

## CloseMonPrinter

Ends using the Win32 API.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
INT CloseMonPrinter(
    INT i_hdl )
```

### Parameters

*i\_hdl*

API handle

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

### Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

### Remarks

- When the API handle specified by this API is being used in another API, control of this API does not return until the processing is completed.
- All settings and data that have been associated with the API handle are discarded by this API.

# LockPrinter

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

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
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 time to wait for success of this API in milliseconds (ms).  
The valid range is from 3000 to 90000.

When the value is specified less than 3000, the time is set to 3000 ms.

When the value is specified more than 90000, the time is set to 90000 ms.

## Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

## Remarks

- When any APIs that directly access the printer are called from other processes during the time from calling this API until **UnlockPrinter** is called, those APIs fail.
- Only other processes are locked by this API. Therefore, calls from the process itself are not locked regardless of API handles or threads.
- By this API, multiple locking is possible up to 99 times with the same API handle. In order to release the lock, call **UnlockPrinter** the same number of times as for this API.
- When **LockPrinter** is being called by another process, calling this API fails.
- For using the TCP/IP connection, release the lock before the time of no transmission exceeds the TCP/IP receive timeout period. If it is locked, it may cause the data to be missed or the data sent from other hosts to interrupt the printer.  
For the receive timeout period, see "LAN communication settings" in "3.2.3 Setting Screen for Each Connection Method" described in the "SII Communication Setting Utility for Windows User's Guide".

## UnlockPrinter

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

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
INT UnlockPrinter(  
    INT i_hdl )
```

### Parameters

*i\_hdl*

API handle

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

### Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

### Remarks

- When **LockPrinter** has been called multiple times, this API must be called the same number of times as for **LockPrinter** in order to release the lock.

## DirectIO

Sends and receives binary data.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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>i_hdl</i>	API handle Specifies the API handle retrieved by <b>OpenMonPrinter</b> .
<i>i_wlen</i>	Size of data to send Specifies the size of data to send.
<i>i_wcmd</i>	Buffer of data to send Specifies the buffer in which the data to send is stored.
<i>io_rlen</i>	Size of data to receive Specifies the maximum length of data to be received from the printer. Specifies 0 when data retrieving is not necessary. When the control returns from the API, the retrieved receive data size is stored.
<i>o_rbuf</i>	Buffer of data to receive Specifies the buffer that stores the data to get.
<i>i_timeout</i>	Timeout period Specifies the time to wait for success of this API in milliseconds (ms). The valid range is from 3000 to 90000. When the value is specified less than 3000, the time is set to 3000 ms. When the value is specified more than 90000, the time is set to 90000 ms.
<i>i_flag</i>	Receive operation flag Specifies the flag to specify the receive operation from the following: 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 "5.1 Error Code List" for details of the error code.

## **Remarks**

- In the case of sending commands, data, and image data that do not allow interrupt of other data, call this API after calling **LockPrinter**. When **LockPrinter** is not called, data from other processes may interrupt.
- In the data to send, do not include the ASB setting command that disables the response. The API that gets the printer status will not work properly.
- Receive data does not include the response of the ASB setting command. To get data that includes the response of the ASB setting command, call **DirectIOEx**.
- This API can be aborted by **Reset**.
- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.
- For Bluetooth connection, do not include a printer command that initializes the printer other than the printer command "Initialize Printer" in the data to send. For printer initialization, see "Technical Reference". When performing a hardware reset, execute **Reset**.

- This API does not get response data from the barcode scanner. Use callback function registering in **SetBarcodeDataBackFunction** to get the response data from the barcode scanner.

## DirectIOEx

Sends and receives binary data.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 in which the data to send is stored.

*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 value is specified more than 4096, the size is set to 4096 bytes.  
Specify 0 when data retrieving is not necessary.  
When the control returns from the API, the received data size is stored.

*o\_rbuf*  
Buffer of data to receive  
Specifies the buffer that stores the data to get.

#### *i\_timeout*

Timeout period

Specifies the time to wait for success of this API in milliseconds (ms).

The valid range is from 3000 to 90000.

When the value is specified less than 3000, the time is set to 3000 ms.

When the value is specified more than 90000, the time is set to 90000 ms.

#### *i\_flag*

Receive operation flag

Specifies the flag to specify the receive operation from the following:

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 the data to receive from the following:

0: Gets the data excluding the response of the ASB setting command.

1: Gets the data including the response of the ASB setting command.

### **Return value**

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

### **Remarks**

- In the case of sending commands, data, and image data that do not allow interrupt of other data, call this API after calling **LockPrinter**. When **LockPrinter** is not called, data from other processes may interrupt.
- In the data to send, do not include the ASB setting command that disables the response. The API that gets the printer status will not work properly.
- This API can be aborted by **Reset**.
- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.
- For Bluetooth connection, do not include a printer command that initializes the printer other than the printer command "Initialize Printer" in the data to send. For printer initialization, see "Technical Reference". When performing a hardware reset, execute **Reset**.
- This API does not get response data from the barcode scanner. Use callback function registering in **SetBarcodeDataBackFunction** to get the response data from the barcode scanner.

# Reset

Performs hardware reset of the printer.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
INT Reset(  
    INT i_hdl )
```

## Parameters

*i\_hdl*

API handle

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

## Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

## Remarks

- Performs hardware reset of the printer using the communication protocol (without using printer commands).
- After called this API, wait a few seconds to send the data. If data transmission is performed right after calling this API, it may cause data missing.
- When this API is called, the following APIs are aborted.
  - **DirectIO**
  - **DirectIOEx**
  - **DirectSendRead**
- During the call of this API, the printer status response will be "No response". See "6.1 Printer Status List" for details of the printer status.
- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.
- For Bluetooth connection, when this API is executed but the printer is in the condition of no data accepting, this API succeeds, but the printer 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.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 a variable to store the printer status.

### Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the 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 called when a change in the printer status is detected.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
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 a variable in which the printer status is stored.

## Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

## Remarks

- When a callback function is registered by this API, the callback function is called with the current printer status.
- 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 reconnection to the printer is detected, the printer status is the status received last at the time.
- When this API is called in the state that the callback function is registered, the already registered function becomes disabled, and a new callback function is registered.
- Even when this API is called specifying the already registered and enabled callback function again, the immediate response of the printer status is performed.
- The callback function registered by this API is unregistered by the following APIs:
  - **CloseMonPrinter**
  - **CancelStatusBack**

- The following APIs cannot be called from the registered callback function with the same API handle:
  - **CloseMonPrinter**
  - **SetStatusBackFunction**
  - **SetStatusBackWnd**
  - **CancelStatusBack**
  - **SetBarcodeDataBackFunction**
  - **CancelBarcodeDataBack**
- The return value of the callback function is ignored.
- The time from receiving the printer status until 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 in the printer status is detected.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 that sends the click event.  
When NULL is specified, this API is canceled.

*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 "5.1 Error Code List" for details of the error code.

## Remarks

- When the button window handle is registered by this API, the click event is called with the current printer status.
- 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 reconnection to the printer is detected, the printer status is the status received last at the time.
- When this API is called in the state that the button window handle is registered, the already registered button window handle becomes disabled and a new button window handle is registered.
- Even when this API is called specifying the already registered and enabled button window handle again, the immediate response of the printer status is performed.
- This API is canceled by the following APIs:
  - **CloseMonPrinter**
  - **CancelStatusBack**
- The following APIs cannot be called from the processing routine of the called click event with the same API handle:
  - **CloseMonPrinter**
  - **SetStatusBackFunction**
  - **SetStatusBackWnd**
  - **CancelStatusBack**
  - **SetBarcodeDataBackFunction**
  - **CancelBarcodeDataBack**
- The return value of the click event is ignored.
- The time from receiving the printer status until calling the click event is not guaranteed.
- See "6.1 Printer Status List" for details of the printer status.

## **CancelStatusBack**

Cancels the registration of the callback function called by **SetStatusBackWnd** or **SetStatusBackFunction**.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
INT CancelStatusBack(  
    INT i_hdl )
```

### Parameters

*i\_hdl*

API handle

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

### Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

### Remarks

- This API succeeds even when registration has not been performed by either **SetStatusBackWnd** or **SetStatusBackFunction**.

## **SetBarcodeDataBackFunction**

Registers the callback function of the barcode scanner.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

### Syntax

```
INT SetBarcodeDataBackFunction(  
    INT i_hdl,  
    INT ( CALLBACK EXPORT *lpDataCB ) ( ST_BARDATA lpData ) )
```

### Parameters

*i\_hdl*

API handle

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

*lpDataCB*

Callback function address

Specifies the address of an application-defined callback function receiving the response data from the barcode scanner.

When NULL is specified, monitoring of the barcode scanner is canceled.

*lpData*

Barcode data structure

Specifies the address of the structure where the response data or size from the barcode scanner is to be stored.

See "6.9 Barcode Data Structure" for details of barcode data structure.

### Return value

On success: Returns 0.

On failure: Returns an error code.  
See "5.1 Error Code List" for details of the error code.

### **Remarks**

- The target of callback function is as follows.
  - Receipt of barcode data
  - Connection of barcode scanner
  - Disconnection of barcode scanner
- The barcode data structure is required to be defined in the user's application to use this API. See "6.9 Barcode Data Structure" for details of barcode data structure.
- The receive data is responded to in the binary form. The maximum receive data size is 64 KB. Exceeding the maximum receive data size is deleted the receive data in chronological order.
- When the callback function is registered by this API, the callback function is called with the connection state of the current barcode scanner.
- Even when the connection state of the barcode scanner is received, the callback function is not called when the barcode scanner connection state has not changed from when it was last received.
- When this API is called in the state with the callback function registered, the registered function is disabled and a new callback function is to be registered.
- Even when this API is called specifying the already registered and enabled callback function again, the immediate response of the barcode scanner connection state is performed.
- This API is canceled by the following APIs:
  - **CloseMonPrinter**
  - **CancelBarcodeDataBack**
- The following APIs cannot be called by the same API handle in the registered callback function:
  - **CloseMonPrinter**
  - **SetStatusBackFunction**
  - **SetStatusBackWnd**
  - **CancelStatusBack**
  - **SetBarcodeDataBackFunction**
  - **CancelBarcodeDataBack**
- The return value of the callback function is ignored.
- The time from receiving the barcode data until calling the callback function is not guaranteed.

## CancelBarcodeDataBack

Cancels the registration of the callback function called by **SetBarcodeDataBackFunction**.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

### Syntax

```
INT CancelBarcodeDataBack(  
    INT i_hdl )
```

### Parameters

*i\_hdl*

API handle

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

### Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

### Remarks

- This API succeeds even when registration has not been performed by either **SetBarcodeDataBackFunction**.

## PowerOff

Turns off the printer power.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 "5.1 Error Code List" for details of the error code.

## GetCounter

Gets the maintenance counter.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 of the maintenance counter to get.  
See "6.2 Counter ID" for specifiable values.

*o\_readcounter*  
Counter variable  
Specifies a variable to store the retrieved counter value.

### Return value

On success: Returns 0.  
On failure: Returns an error code.  
See "5.1 Error Code List" for details of the error code.

### Remarks

- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.

## ResetCounter

Initializes the maintenance counter.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 of the maintenance counter to initialize the counter value.  
See "6.2 Counter ID" for specifiable values.

### Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

### Remarks

- Confirm with **GetCounter** that the value of the counter ID specified by this API has been initialized.
- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.

# GetType

Gets various IDs of the printer.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
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  
3 is stored.

*o\_fontID*  
ID2  
2 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 "5.1 Error Code List" for details of the error code.

## Remarks

- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.

# GetPrnCapability

Gets the printer information.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
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 of the printer information to get.  
See "6.3 Printer ID" for specifiable values.

*io\_datsize*

Size of data to receive

Specifies the buffer size to store the printer information to get.  
When the control returns from the API, the data size of the retrieved printer information is stored.  
When the specified buffer size is smaller than the response size of the printer information to get, this API fails and the specified response size of the printer information is stored.

*o\_dat*

Buffer of data to receive

Specifies the buffer that stores the retrieved printer information.

## Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

## Remarks

- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.

# OpenDrawer

Drives the specified drawer.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
INT OpenDrawer(  
    INT i_hdl,  
    BYTE i_drawer,  
    BYTE i_pulse )
```

## Parameters

*i\_hdl*

API handle

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

*i\_drawer*

Drawer

Specifies the drawer to drive.

See "6.5 Target Drawer" for specifiable values.

*i\_pulse*

Drawer drive time

Specifies ON/OFF time of the drawer driving pulse. The ON time and the OFF time are specified as the same.

See "6.6 Drawer Drive Time" for specifiable values.

## Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

## Remarks

- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.
- As for the drawer control time, follow the specifications of your drawer.

## ControlFeature

Controls functions of the printer other than printing.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	-

### Syntax

```
INT ControlFeature(  
    INT i_hdl,  
    BYTE i_feature,  
    INT i_arg1,  
    INT i_arg2 )
```

### Parameters

*i\_hdl*

API handle

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

*i\_feature*

Printer function

Specifies the function of the printer to control.

See "6.7 Printer Function" for specifiable values.

*i\_arg1*

Argument 1

Specifies the argument for the specified printer function.

See "6.7 Printer Function" for specifiable values.

*i\_arg2*

Argument 2

Specifies the argument for the specified printer function.

See "6.7 Printer Function" for specifiable values.

### Return value

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

### Remarks

- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.

## SendDataFile

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

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 "5.1 Error Code List" for details of the error code.

### Remarks

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

## DirectSendRead

Sends transmission data corresponding to the command name specified in the command definition file that has been registered by **SendDataFile**.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 the command name registered by **SendDataFile**.

*i\_rtype*

Receive data type name

Specifies the type of data to receive from the following:

ASB: Stores only the response of the ASB setting command in the receive data.

Other: Stores responses other than the above 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 value is specified more than 4096, the size is set to 4096 bytes.

Specify 0 when data retrieving is not necessary.

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

*o\_rbuf*

Buffer of data to receive

Specifies the buffer that stores the data to get.

*i\_timeout*

Timeout period

Specifies the time to wait for success of this API in milliseconds (ms).

The valid range is from 3000 to 90000.

When the value is specified less than 3000, the time is set to 3000 ms.

When the value is specified more than 90000, the time is set to 90000 ms.

*i\_flag*

Receive operation flag

Specifies the flag to specify the receive operation from the following:

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 "5.1 Error Code List" for details of the error code.

## Remarks

- This API can be aborted by **Reset**.
- When **LockPrinter** is being called by another process, calling this API fails.
- When the printer is not connected or in a communication disabled state, this API fails.
- For Bluetooth connection, do not include a printer command that initializes the printer other than the printer command "Initialize Printer" in the data to send. For printer initialization, see "Technical Reference". When performing a hardware reset, execute **Reset**.

# GetProperty

Gets a part of the print settings.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

```
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 of the print settings to get.  
See "6.8 Property ID" for specifiable values.

*o\_dat*

Buffer of data to get

Specifies the buffer that stores the retrieved print settings.

When NULL is specified, this API fails and the specified data size of the print settings is stored in *io\_size*.

*io\_size*

Size of data to get

Specifies the buffer size to store the print settings.

When the control returns from the API, the data size of the retrieved print settings is stored.

When the specified buffer size is smaller than the data size of the print settings to get, this API fails and the specified data size of the print settings is stored.

### **Return value**

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

## **SetProperty**

Changes a part of the print settings.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### **Syntax**

```
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 of the print settings to be changed.

See "6.8 Property ID" for specifiable values.

*i\_dat*

Buffer of data to set

Specifies the buffer in which the print settings to be changed is stored.

*i\_size*

Size of data to set

Specifies the buffer size in which the print settings to be changed is stored.

**Return value**

On success: Returns 0.

On failure: Returns an error code.

See "5.1 Error Code List" for details of the error code.

---

# Chapter 4 .NET API

---

This chapter describes 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.

- SPSWL\_Cls.dll

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

## 4.3 Printer

---

### 4.3.1 API List

The APIs implemented in the .NET API are as follows.

- Namespace: SII.SDK.PrinterDevice
- Class name: StatusAPI

### Common API

Category	API	Function Summary
Property	<b>LastError</b>	Gets the error value of the last executed API.
Property	<b>IsValid</b>	Gets the call state of <b>OpenMonPrinter</b> .
Method	<b>OpenMonPrinter</b>	Starts using the .NET API.
Method	<b>CloseMonPrinter</b>	Ends using the .NET API.

### Specific API

Category	API	Function Summary
Property	<b>Status</b>	Gets the latest printer status.
Method	<b>LockPrinter</b>	Locks all data transmission and hardware reset requests from other processes to the printer.
Method	<b>UnlockPrinter</b>	Unlocks access prohibition (lock) from other processes by <b>LockPrinter</b> .
Method	<b>DirectIOEx</b>	Sends and receives binary data.
Method	<b>ResetPrinter</b>	Performs hardware reset of the printer.
Method	<b>SetStatusBack</b>	Starts the notification of the printer status by <b>StatusCallback</b> event.
Method	<b>CancelStatusBack</b>	Stops the notification of the printer status by <b>StatusCallback</b> event.
Method	<b>PowerOff</b>	Turns off the printer power.
Method	<b>GetCounter</b>	Gets the maintenance counter.
Method	<b>ResetCounter</b>	Initializes the maintenance counter.
Method	<b>GetType</b>	Gets various IDs of the printer.
Method	<b>GetPrnCapability</b>	Gets the printer information.
Method	<b>OpenDrawer</b>	Drives the specified drawer.
Method	<b>ControlFeature</b>	Controls functions of the printer other than printing.
Method	<b>SendDataFile</b>	Registers the contents of the command definition file in the SDK internal memory.
Method	<b>DirectSendRead</b>	Executes the command definition registered by <b>SendDataFile</b> .
Method	<b>GetProperty</b>	Gets a part of the print settings.
Method	<b>SetProperty</b>	Changes a part of the print settings.

Category	API	Function Summary
Event	<b>StatusCallback</b>	Notifies the returned printer status.

## Caution

- ◆ With Bluetooth connection, the response data while the printer is not connected cannot be retrieved.

## 4.3.2 Common Property

### LastError

Gets the error value of the last called API.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

#### Syntax

```
ErrorCode LastError { get; }
```

#### Initial value

SUCCESS

#### Remarks

- See "5.1 Error Code List" for details of the error code.

### IsValid

Gets the call state of **OpenMonPrinter**.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

#### Syntax

```
bool IsValid { get; }
```

#### Initial value

FALSE

### **Remarks**

- TRUE: **OpenMonPrinter** is successful.
- FALSE: **OpenMonPrinter** is not successful.

### 4.3.3 Common Method

## OpenMonPrinter

Starts using the .NET API.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

```
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 "5.1 Error Code List" for details of the error code.

### Remarks

- It can be opened up to 8 instances at the same time within 1 process combining each class in .NET API.
- When the .NET API is no longer used, be sure to call **CloseMonPrinter**.
- Set the connection destination of the printer driver to USB, Serial (including the COM port of the Bluetooth device), or TCP/IP.
- This API succeeds even when the printer is not connected or the printer power is turned off.

## CloseMonPrinter

Ends using the .NET API.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### **Syntax**

ErrorCode **CloseMonPrinter()**

### **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

### **Remarks**

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

## 4.3.4 Specific Property

### Status

Gets the latest printer status.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

#### Syntax

```
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.

## 4.3.5 Specific Method

### LockPrinter

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

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

#### Syntax

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

#### Parameters

*timeout*

Timeout period

Specifies the time to wait for success of this API in milliseconds (ms).

The valid range is from 3000 to 90000.

When the value is specified less than 3000, the time is set to 3000 ms.

When the value is specified more than 90000, the time is set to 90000 ms.

#### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

#### Remarks

- By this API, multiple locking is possible up to 99 times. In order to release the lock, call **UnlockPrinter** the same number of times as for this API.
- For using the TCP/IP connection, release the lock before the time of no transmission exceeds the TCP/IP receive timeout period. If it is locked, it may cause the data to be missed or the data sent from other hosts to interrupt the printer.  
For the receive timeout period, see "LAN communication settings" in "3.2.3 Setting Screen for Each Connection Method" described in the "SII Communication Setting Utility for Windows User's Guide".

## UnlockPrinter

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

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

ErrorCode **UnlockPrinter**()

### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

### Remarks

- When **LockPrinter** is called multiple times, this API must be called the same number of times as for **LockPrinter** in order to release the lock.

## DirectIOEx

Sends and receives binary data.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

With the syntax (a), the API gets the receive data as binary data from the printer after sending binary data.

With the syntax (b), the API gets the receive data as string data from the printer after sending binary data.

With the syntax (c), the API sends binary data.

### Syntax

(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 in which the data to send is stored.

*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 value is specified more than 4096, the size is set to 4096 bytes.

Specify 0 when data retrieving is not necessary.

When the control returns from the API, the received data size is stored.

*timeout*

Timeout period

Specifies the time to wait for success of this API in milliseconds (ms).

The valid range is from 3000 to 90000.

When the value is specified less than 3000, the time is set to 3000 ms.

When the value is specified more than 90000, the time is set to 90000 ms.

*readFlag*

Receive operation flag

Specifies the flag to specify the receive operation from the following:

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 the data to receive from the following:

0: Gets the data excluding the response of the ASB setting command.

1: Gets the data including the response of the ASB setting command.

### **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

### **Remarks**

- When this API succeeds, *data* in syntax (a) is resized to the actual receive data size within the limits of specified receive data size as *data*.
- In the data to send, do not include the ASB setting command that disables the response. The API that gets the printer status will not work properly.
- This API can be aborted by **ResetPrinter**.
- When 0x02 is included in the receive data, it is converted to 0x5f.

- For Bluetooth connection, do not include a printer command that initializes the printer other than the printer command "Initialize Printer" in the data to send. For printer initialization, see "Technical Reference". When performing a hardware reset, execute **ResetPrinter**.

## ResetPrinter

Performs a hardware reset of the printer.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

ErrorCode **ResetPrinter**()

### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

### Remarks

- Performs hardware reset of the printer using the communication protocol (without using printer commands).
- After called this API, wait a few seconds to send the data. If data transmission is performed right after calling this API, it may cause data missing.
- When this API is called, the following APIs are aborted.
  - **DirectIOEx**
  - **DirectSendRead**
- During the call of this API, the printer status response will be "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.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

ErrorCode **SetStatusBack**()

### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

### Remarks

- When the notification of the printer status is started by this API, **StatusCallback** event is raised with the current printer status.
- When the event handler is not registered on **StatusCallback** event, this API fails.
- Even when **StatusCallback** event is already started and this API is called again, the immediate response of the printer status is performed.
- **StatusCallback** event started by this API is stopped by the following APIs:
  - **CloseMonPrinter**
  - **CancelStatusBack**
- 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.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

ErrorCode **CancelStatusBack**()

### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

### **Remarks**

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

## **PowerOff**

Turns off the printer power.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### **Syntax**

ErrorCode **PowerOff**()

### **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

### **Remarks**

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

## **GetCounter**

Gets the maintenance counter.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

There are 2 types of syntax to get the maintenance counter: the syntax that uses the counter ID defined in CounterIndex, and the syntax that specifies it as a numerical value.

With the syntax (a), the API gets the counter value using the counter ID defined in CounterIndex. With the syntax (b), the API gets the counter value by specifying the counter ID as a numerical value.

## **Syntax**

- (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 of the maintenance counter to get.  
Specify (a) when using the counter ID defined in CounterIndex.  
Specify (b) when specifying the counter ID as a numerical value.  
See "6.2 Counter ID" for specifiable values.

*type*

Maintenance counter type

Specifies the type of the maintenance counter to get from the following:  
TRUE:    Integration counter  
FALSE:   Initializable counter

*data*

Counter variable

Specifies the variable to store the retrieved counter value.

## **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

## **ResetCounter**

Initializes the maintenance counter.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

There are 2 types of syntax to initialize the maintenance counter: the syntax that uses the counter ID defined in CounterIndex, and the syntax that specifies it as a numerical value.

With the syntax (a), the API initializes the counter value using the counter ID defined in CounterIndex.

With the syntax (b), the API initializes the counter value by specifying the counter ID as a numerical value.

## Syntax

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

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

## Parameters

*index*

Counter ID

Specifies the counter ID of the maintenance counter to initialize the counter value.  
Specify (a) when using the counter ID defined in CounterIndex.  
Specify (b) when specifying the counter ID as a numerical value.  
See "6.2 Counter ID" for specifiable values.

## Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

## Remarks

- Confirm with **GetCounter** that the value of the counter ID specified by this API has been initialized.

# GetType

Gets various IDs of the printer.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## Syntax

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

## Parameters

*typeId*

ID1

3 is stored.

*fontID*

ID2

2 is stored.

*exrom*  
 Reserved  
 Specify null.

*special*  
 Reserved  
 Specify null.

### **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

## **GetPrnCapability**

Gets the printer information.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

There are 2 types of syntax to get the printer information: the syntax that uses the printer ID defined in PrinterId, and the syntax that specifies it as a numerical value.

With the syntax (a), the API gets the printer information as binary data using the printer ID defined in PrinterId.

With the syntax (b), the API gets the printer information as string data using the printer ID defined in PrinterId.

With the syntax (c), the API gets the printer information as binary data by specifying the printer ID as a numerical value.

With the syntax (d), the API gets the printer information as string data by specifying the printer ID as a numerical value.

### **Syntax**

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

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

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

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

## **Parameters**

*id*

Printer ID

Specifies the printer ID of the printer information to get.

Specify (a) or (b) when using the printer ID defined in PrinterId. (a) gets the response format of the printer information as a binary code, and (b) gets it as an ASCII character string.

Specify (c) or (d) when specifying the printer ID as a numerical value. (c) gets the response format of the printer information as a binary code, and (d) gets it as an ASCII character string.

See "6.3 Printer ID" for specifiable values.

*data*

Buffer of data to receive

Specifies the buffer that stores the retrieved printer information.

## **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

# **OpenDrawer**

Drives the specified drawer.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

## **Syntax**

ErrorCode **OpenDrawer**(  
Drawer *drawer*,  
Pulse *pulse* )

## **Parameters**

*drawer*

Drawer

Specifies the drawer to drive.

See "6.5 Target Drawer" for specifiable values.

*pulse*

Drawer drive time

Specifies ON/OFF time of the drawer driving pulse. The ON time and the OFF time are specified as the same.

See "6.6 Drawer Drive Time" for specifiable values.

## **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

## **Remarks**

- As for the drawer control time, follow the specifications of your drawer.

## **ControlFeature**

Controls functions of the printer other than printing.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	-

## **Syntax**

```
ErrorCode ControlFeature(  
    Feature feature,  
    int arg1,  
    int arg2 )
```

## **Parameters**

*feature*

Printer function

Specifies the function of the printer to control.

See "6.7 Printer Function" for specifiable values.

*arg1*

Argument 1

Specifies the argument for the specified printer function.

See "6.7 Printer Function" for specifiable values.

*arg2*

Argument 2

Specifies the argument for the specified printer function.

See "6.7 Printer Function" for specifiable values.

## **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

## SendDataFile

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

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

### Syntax

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 "5.1 Error Code List" for details of the error code.

### Remarks

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

## DirectSendRead

Sends transmission data corresponding to the command name specified in the command definition file that has been registered by **SendDataFile**.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

With the syntax (a), the API gets the receive data as binary data from the printer after sending the transmission data corresponding to the command name specified in the command definition file that has been registered by **SendDataFile**.

With the syntax (b), the API gets the receive data as string data from the printer after sending the transmission data corresponding to the command name specified in the command definition file that has been registered by **SendDataFile**.

With the syntax (c), the API sends the transmission data corresponding to the command name specified in the command definition file that has been registered by **SendDataFile**.

## **Syntax**

- (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 the command name defined by **SendDataFile**.

*readType*

Receive data type name

Specifies the type of data to receive from the following:

ASB: Stores only the response of the ASB setting command in the receive data.

Other: Stores responses other than the above 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 time to wait for success of this API in milliseconds (ms).

The valid range is from 3000 to 90000.

When the value is specified less than 3000, the time is set to 3000 ms.

When the value is specified more than 90000, the time is set to 90000 ms.

*readFlag*

Receive operation flag

Specifies the flag to specify the receive operation from the following:

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 "5.1 Error Code List" for details of the error code.

## **Remarks**

- This API can be aborted by **ResetPrinter**.
- When 0x02 is included in the receive data, it is converted to 0x5f.

# GetProperty

Gets a part of the print settings.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

There are 2 types of syntax to get the print settings: the syntax that uses the property ID defined in PropertyId, and the syntax that specifies it as a numerical value.

With the syntax (a), the API gets a part of the print settings of the property using the property ID defined in PropertyId.

With the syntax (b), the API gets a part of the print settings of the property by specifying the property ID as a numerical value.

## Syntax

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

(b) 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 of the print settings to get.  
Specify (a) when using the property ID defined in PropertyId.  
Specify (b) when specifying the property ID as a numerical value.  
See "6.8 Property ID" for specifiable values.

*data*  
Buffer of data to get  
Specifies the buffer that stores the retrieved print settings.

*size*  
Size of data to get  
Specifies the buffer size that stores the print settings.  
When the control returns from the API, the data size of the retrieved print settings is stored.

## Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

# SetProperty

Changes a part of the print settings.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

There are 2 types of syntax to change the print settings: the syntax that uses the property ID defined in PropertyId, and the syntax that specifies it as a numerical value.

With the syntax (a), the API changes a part of the print settings using the property ID defined in PropertyId.

With the syntax (b), the API changes a part of the print settings by specifying the property ID as a numerical value.

## Syntax

(a) ErrorCode **SetProperty**(  
    IntPtr *devmode*,  
    PropertyId *id*,  
    byte[] *data*,  
    uint *size* )

(b) 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 of the print settings to be changed.

Specify (a) when using the property ID defined in PropertyId.

Specify (b) when specifying the property ID as a numerical value.

See "6.8 Property ID" for specifiable values.

*data*

Buffer of data to set

Specifies the buffer in which the print settings to be changed is stored.

*size*

Size of data to set

Specifies the buffer size in which the print settings to be changed is stored.

## Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

## 4.3.6 Event

### StatusCallback

Notifies the returned printer status.

Support printer

RP-F10	RP-G10	RP-E10
✓	✓	✓

#### Syntax

event StatusCallbackHandler **StatusCallback**

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

#### Parameters

*status*

Printer status variable

The printer status is stored.

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.
- 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.
- When reconnection to the printer is detected, the printer status is the status received last at the time.
- Call the following APIs to stop this API.
  - **CloseMonPrinter**
  - **CancelStatusBack**
- The following APIs cannot be called from the registered event handler.
  - **CloseMonPrinter**
  - **SetStatusBack**
  - **CancelStatusBack**
- 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.

## 4.4 Display

---

The APIs implemented in the .NET API are as follows.

- Namespace: SII.SDK.PrinterDevice
- Class name: DisplayAPI

For APIs of Display, see "SII SDK for Windows Application Programmer's Guide" for DSP-A01 for details.

### Reference

- RP-G10 and RP-E10 do not support Display.

## 4.5 Barcode Scanner

---

### 4.5.1 API List

The APIs implemented in the .NET API are as follows.

- Namespace: SII.SDK.PrinterDevice
- Class name: BarcodeScannerAPI

### Common API

Category	API	Function Summary
Property	<b>LastError</b>	Gets the error value of the last executed API.
Property	<b>IsValid</b>	Gets the call state of <b>OpenMonPrinter</b> .
Method	<b>OpenMonPrinter</b>	Starts using the .NET API.
Method	<b>CloseMonPrinter</b>	Ends using the .NET API.

### Specific API

Category	API	Function Summary
Method	<b>SetBarcodeDataBack</b>	Starts the notification of the barcode scanner changing state by <b>BarcodeDataCallback</b> event.
Method	<b>CancelBarcodeDataBack</b>	Stops the notification of the barcode scanner changing status by <b>BarcodeDataCallback</b> event.
Event	<b>BarcodeDataCallback</b>	Notifies the changing state of the response barcode scanner.

### Caution

- ◆ With Bluetooth connection, the response data while the printer is not connected cannot be retrieved.
- ◆ See "RP-F10 SERIES Thermal Printer USER'S GUIDE" for details of the recommended barcode scanner and the barcode scanner setting.

## 4.5.2 Common Property

### LastError

Gets the error value of the last called API.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

#### Syntax

```
ErrorCode LastError { get; }
```

#### Initial value

SUCCESS

#### Remarks

- See "5.1 Error Code List" for details of the error code.

### IsValid

Gets the call state of **OpenMonPrinter**.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

#### Syntax

```
bool IsValid { get; }
```

#### Initial value

FALSE

**Remarks**

- TRUE : **OpenMonPrinter** is successful.
- FALSE : **OpenMonPrinter** is not successful.

## 4.5.3 Common Method

### OpenMonPrinter

Starts using the .NET API.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

#### Syntax

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

#### Parameters

*type*  
Open type  
OpenType.TYPE\_BARCODE\_SCANNER (fixed)

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

#### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

#### Remarks

- It can be opened up to 8 instances at the same time within 1 process combining each class in .NET API.
- When the .NET API is no longer used, be sure to call **CloseMonPrinter**.
- Set the connection destination of the printer driver to USB, Serial (including the COM port of the Bluetooth device), or TCP/IP.
- This API succeeds even when the printer is not connected or the printer power is turned off.
- This API succeeds even when the barcode scanner is not connected.

## CloseMonPrinter

Ends using the .NET API.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

### **Syntax**

ErrorCode **CloseMonPrinter()**

### **Return value**

Returns an error code. See "5.1 Error Code List" for details of the error code.

### **Remarks**

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

## 4.5.4 Specific Property

### SetBarcodeDataBack

Starts the notification of the barcode scanner changing state by **BarcodeDataCallback** event.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

#### Syntax

ErrorCode **SetBarcodeDataBack**()

#### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

#### Remarks

- The barcode scanner changing states are as follows.
  - Receipt of barcode data
  - Connection of barcode scanner
  - Disconnection of barcode scanner
- When the notification of the barcode scanner changing state is started by this API, the **BarcodeDataCallback** event is raised with the current barcode scanner changing state.
- When the event handler is not registered in **BarcodeDataCallback** event, this API fails.
- Even when **BarcodeDataCallback** event is already started and this API is called, the immediate response of the barcode scanner changing state is performed.
- **BarcodeDataCallback** event started by this API is stopped by the following APIs:
  - **CloseMonPrinter**
  - **CancelStatusBack**
- The time from receiving the barcode data to raising the event is not guaranteed.

## CancelBarcodeDataBack

Stops the notification of the barcode scanner changing state by **BarcodeDataCallback**.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

### Syntax

ErrorCode **CancelBarcodeDataBack**()

### Return value

Returns an error code. See "5.1 Error Code List" for details of the error code.

### Remarks

- This API succeeds even when the notification of the barcode scanner changing state has not been started by **SetBarcodeDataBack**.

## 4.5.5 Event

### BarcodeDataCallback

Notifies the changing state of the response barcode scanner.

Support printer

RP-F10	RP-G10	RP-E10
✓	-	-

#### Syntax

event BarcodeDataCallbackHandler **BarcodeDataCallback**

delegate void BarcodeDataCallbackHandler (  
BarcodeData *barcodeData*)

#### Parameters

*barcodeData*

Buffer of data to receive

Stores the response data from the barcode scanner.

See "6.9 Barcode Data Structure" for the content of the response data to be stored.

#### Remarks

- The barcode scanner changing states are as follows.
  - Receipt of barcode data
  - Connection of barcode scanner
  - Disconnection of barcode scanner
- To start the notification of the barcode scanner changing state, call **SetBarcodeDataBack**.
- When the notification of the barcode scanner changing state is started by **SetBarcodeDataBack**, the event is raised with the current barcode scanner connection state.
- Register the event handler to handle the response barcode scanner changing state before starting the notification of the barcode scanner changing state.
- When the event handler is not registered, **SetBarcodeDataBack** fails.
- Even when the barcode scanner connection state is received, the event is not raised when the barcode scanner connection state has not changed from when it was last received.
- When reconnection to the barcode scanner is detected, the barcode scanner connection state is the state received last at the time.
- Call the following APIs to stop this API.
  - **CloseMonPrinter**
  - **CancelBarcodeDataBack**

- The following APIs cannot be called from the registered event handler.
  - **CloseMonPrinter**
  - **SetBarcodeDataBack**
  - **CancelBarcodeDataStatusBack**
- The time from receiving the barcode data to raising the event is not guaranteed.

---

# 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 because it is not reachable on the network, or the service is stopped during access.
ERR_PARAM	-90	Parameter is incorrect.
ERR_NOT_SUPPORT	-100	Function is not supported.
ERR_OFFLINE	-110	Printer is disconnected or offline.
ERR_NON_TARGETTED_DRIVER	-120	Specified printer driver is not supported.
ERR_DISK_FULL	-170	Printer is busy.
ERR_ENTRY_OVER	-190	Processing capacity is exceeded.
ERR_EXIST	-210	Existing module is called.
ERR_NOT_FOUND	-220	File cannot be found, or it is not 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 or SII Printer Software service has not been started.
ERR_LOCKED	-1000	Printer is locked.

Macro Definition (Constant)	Value	Description
ERR_UNLOCKED	-1010	<b>UnlockPrinter</b> was executed, even though 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 has been canceled.
ERR_UNKNOWN_PORT	-1070	Port is not supported.
ERR_INVALID_PRINTER_STATE	-1080	Printer status is abnormal.
ERR_BAD_ENVIRONMENT	-1090	Printer driver may not be installed normally.

# Chapter 6 Argument Information

This chapter describes the arguments.

## 6.1 Printer Status List

The corresponding bits of the printer status responses are as follows. In addition, please refer to the printer status processing of our sample program.

Printer Status	Corresponding Bit		Description
Voltage error	ASB_VP_ERR	0x00000000	No voltage error
		0x00000001	Voltage error
RP-F10, RP-G10 Hardware error	ASB_HARDWARE_ERR	0x00000000	No hardware error
		0x00000002	Hardware error
RP-E10 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
Auto cutter error	ASB_AUTOCUTTER_ERR	0x00000000	No auto cutter error
		0x00000008	Auto cutter error
Out-of-paper error	ASB_RECEIPT_END	0x00000000	No out-of-paper error
		0x00000010	Out-of-paper error
RP-E10 Paper-near-end	ASB_RECEIPT_NEAR_END	0x00000000	No paper-near-end
		0x00000020	Paper-near-end
RP-E10 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	Paper cover is closed
		0x00000080	Paper cover is open

Printer Status	Corresponding Bit		Description
FEED switch state	ASB_PAPER_FEED	0x00000000	FEED switch state is "Off"
		0x00000100	FEED switch state is "On"
Paper feed state	ASB_NOW_PRINTING	0x00000000	Stop
		0x00000400	Operating
Recovery waiting state	ASB_RETURN_WAITING	0x00000000	-
		0x00000800	In recovery waiting state
Drawer sensor state	ASB_DRAWER_KICK	0x00000000	Drawer sensor state is "L"
		0x00008000	Drawer sensor state is "H"
FLASH memory rewriting	ASB_FLASH_MEMORY_REWRITING	0x00000000	-
		0x00010000	FLASH memory rewriting
RP-E10 Peripheral device selection	ASB_PERIPHERAL_EQUIPMENT	0x00000000	Printer
		0x00020000	Other
Automatic recovery error	ASB_AUTORECOVER_ERR* <sup>1</sup>	0x00000000	No automatic recovery error
		0x20000000	Automatic recovery error
Unrecoverable error	ASB_UNRECOVER_ERR* <sup>1</sup>	0x00000000	No unrecoverable error
		0x40000000	Unrecoverable error
No response	ASB_NO_RESPONSE* <sup>1</sup>	0x00000000	Printer responds
		0x80000000	Not connected or communication error

\*1: It is an extended status for the response of the ASB setting command.

## 6.2 Counter ID

The counter IDs and descriptions are as follows.

Counter ID		Description	Initialization by ResetCounter
.NET API	.NET API Win32 API		
ROLL_FEED_LINES	20	Number of paper feed dot-lines (in 100 dot-lines)	Enable
ROLL_HEAD_CHARGE	21	Number of thermal head activation times (in 100 dot-lines)	Enable
PAPER_CUT	50	Number of drive times of auto cutter	Enable
OPERATION_TIME	70	Product drive time (in minutes)	Enable

Counter ID		Description	Initialization by ResetCounter
.NET API	.NET API Win32 API		
ROLL_FEED_LINES*1	148	Number of paper feed dot lines (Integrated value) (in 100 dot-lines)	Disable
ROLL_HEAD_CHARGE*1	149	Number of thermal head activation times (Integrated value) (in 100 dot-lines)	Disable
PAPER_CUT*1	178	Number of drive times of auto cutter (Integrated value)	Disable
OPERATION_TIME*1	198	Product drive time (Integrated value) (in minutes)	Disable

\*1: For the .NET API, specify *type* in **GetCounter** to get the counter value of the integration counter. See **GetCounter** of .NET API for details of the API.

## 6.3 Printer ID

The printer IDs and descriptions are as follows.

Printer ID		Description	Response Format
.NET API	.NET API Win32 API		
DEVICE	1	Printer model ID	Numerical value (1 byte) RP-F10, RP-G10: 0x25 RP-E10: 0x1a
TYPE	2	Type ID	Numerical value (1 byte) See "6.4 Type ID"
ROM_VER	3	ROM version ID	Numerical value (1 byte)
FW_VER_MAIN	65	Firmware version (main)	ASCII string "x.xx.xx"
MFG	66	Manufacturer	ASCII string "Seiko Instruments Inc."
MDL	67	Model name	ASCII string RP-F10 series: "SII RP-F10 Series." RP-G10 series: "SII RP-G10 Series." RP-E10 series: "SII RP-E10 Series."
FONT	69	Multi-language font type	ASCII string "KANJI JAPANESE"
FW_VER_BOOT	97	Firmware version (boot)	ASCII string "x.xx.xx"
FW_SUM_BOOT	98	Firmware checksum (boot)	Numerical value (2 bytes)
FW_SUM_MAIN	99	Firmware checksum (main)	Numerical value (2 bytes)
FW_SUM	100	Firmware checksum (main + boot)	Numerical value (2 bytes)

## 6.4 Type ID

The corresponding bits of responses for the type ID and descriptions are as follows.  
The type ID is a part of the printer information retrieved by the printer command "Send Printer ID".

Corresponding Bit	Description	
	RP-F10 RP-G10	RP-E10
0x01	0 : 58 mm 1 : 80 mm	0: Multi-byte code not supported 1: Multi-byte code supported
0x02	Reserved (Fixed to 1)	0: Autocutter not available 1: Autocutter available
0x04	Undefined (Fixed to 0)	
0x08	Undefined (Fixed to 0)	
0x10	Reserved (Fixed to 1)	Undefined (Fixed to 0)
0x20	Reserved (Fixed to 1)	Undefined (Fixed to 0)
0x40	Undefined (Fixed to 0)	
0x80	Undefined (Fixed to 0)	

The retrievable response is the sum of the above values.

Example: Response value for RP-F10 with 80 mm selected = 0x33

## 6.5 Target Drawers

The target drawers and descriptions are as follows.

Drawer		Description
.NET API	.NET API Win32 API	
DRAWER_1	1	Drives drawer 1
DRAWER_2	2	Drives drawer 2

## 6.6 Drawer Drive Time

The drawer drive time and descriptions are as follows.

Drawer Drive Time		Description
.NET API	.NET API Win32 API	
TIME_100	1	Drives drawer for 100 ms
TIME_200	2	Drives drawer for 200 ms
TIME_300	3	Drives drawer for 300 ms
TIME_400	4	Drives drawer for 400 ms
TIME_500	5	Drives drawer for 500 ms
TIME_600	6	Drives drawer for 600 ms
TIME_700	7	Drives drawer for 700 ms
TIME_800	8	Drives drawer for 800 ms

As for the drawer control time, follow the specifications of your drawer.

## 6.7 Printer Function

The printer function and descriptions are as follows.

Printer Function		Description	Argument 1	Argument 2
.NET API	.NET API Win32 API			
EXT_BUZZER	1	Sounds external buzzer	Buzzer pattern*1 0: Pattern 1 1: Pattern 2 2: Pattern 3 3: Pattern 4	Buzzer frequency*2 1 to 255 (times)

\*1: For the buzzer pattern, see "Technical Reference".

\*2: The external buzzer sounds will stop under one of the following conditions:

- When sounds for the set number of buzzers.
- When the cover is open.
- When the printer command "Stop External Buzzer" is executed.

## 6.8 Property ID

### Caution

- ◆ When calling **SetProperty**, specify the data size to 1 byte except for the logo keycode and custom commands.

The property IDs and descriptions are as follows.

Inside [ ] the parentheses indicates the timing when the processing is performed in option settings. See "3.3.7 Setting of Option" in "SII Printer Driver for Windows User's Guide" about the option settings.

Property ID		Description							
.NET API	.NET API Win32 API								
INIT	1	Initialize	0: Enabled 1: Disabled						
SPEED	2	Speed	<table><tr><th>Printer</th><th>Description</th></tr><tr><td>RP-F10 RP-G10</td><td>0 : High 1 : Middle (Quality) 2 : Middle (Silent)</td></tr><tr><td>RP-E10</td><td>0 : Middle (Silent)*<sup>1</sup> 1 : Low 2 : Middle (Quality) 3 : High</td></tr></table>	Printer	Description	RP-F10 RP-G10	0 : High 1 : Middle (Quality) 2 : Middle (Silent)	RP-E10	0 : Middle (Silent)* <sup>1</sup> 1 : Low 2 : Middle (Quality) 3 : High
			Printer	Description					
			RP-F10 RP-G10	0 : High 1 : Middle (Quality) 2 : Middle (Silent)					
RP-E10	0 : Middle (Silent)* <sup>1</sup> 1 : Low 2 : Middle (Quality) 3 : High								
MARGIN	3	Margin	0: Minimum margin 1: Minimum top margin 2: Minimum bottom margin 3: Maximum margin						
DENSITY	4	Density (percentage)	70 to 130						
DIRECTION	5	Direction	0: Forward 1: Backward						
REDUCTION	6	Reduction	0: None RP-F10, RP-G10 20 to 100: Scale (percentage) RP-E10 20 to 100: Scale (percentage)						
CUT	7	Paper cut	0: No cut 1: Full cut (by jobs) 2: Partial cut (by jobs) 3: Full cut (by pages) 4: Partial cut (by pages) 5: Partial cut (between pages)						

Property ID		Description	
.NET API	.NET API Win32 API		
MARK_FEED	8	Marked Paper Form Feed* <sup>2</sup>	0: Disabled 1: Each page 2: Each job
CUT_FEED	9	Feed to cut position	0: Enabled 1: Disabled
START_DOC_LOGO	10	[Print Start] Logo	0: None 1: Left 2: Center 3: Right
START_DOC_LOGO_KEY	11	[Print Start] Logo keycode (2 bytes)* <sup>3</sup>	RP-E10, RP-G10 Each byte 0x20 to 0x39 RP-E10 Each byte 0x30 to 0x39
START_DOC_DRAWER	12	[Print Start] Drawer* <sup>4</sup>	0: No use 1: Use drawer 1 2: Use drawer 2
START_DOC_DRAWER_ON	13	[Print Start] ON time (x 2 ms)	1 to 255
START_DOC_DRAWER_OFF	14	[Print Start] OFF time (x 2 ms)	1 to 255
START_DOC_CMD	15	[Print Start] Custom command (128 bytes)	Command data
START_DOC_BACK_FEED	16* <sup>5</sup>	[Print Start] Paper feed (back feed) (dot)	RP-E10, RP-G10 0 to -60 RP-E10 0 to -74
START_DOC_FEED	17* <sup>5</sup>	[Print Start] Paper feed (feed) (dot)	0 to 255
START_DOC_BUZZER	18	[Print Start] Buzzer* <sup>2</sup>	0: ON 1: OFF
START_DOC_EXT_BUZZER	110	[Print Start] External buzzer* <sup>6</sup>	0: Pattern 1 1: Pattern 2 2: Pattern 3 3: Pattern 4 255: OFF
START_DOC_EXT_BUZZER_RPT	111	[Print Start] External buzzer Frequency* <sup>6*7</sup>	1 to 255
START_PAGE_LOGO	20	[Page Start] Logo	0: None 1: Left 2: Center 3: Right

Property ID		Description	
.NET API	.NET API Win32 API		
START_PAGE_LOGO_KEY	21	[Page Start] Logo keycode (2 bytes)* <sup>3</sup>	RP-E10, RP-G10 Each byte 0x20 to 0x39 RP-E10 Each byte 0x30 to 0x39
START_PAGE_CMD	25	[Page Start] Custom command (128 bytes)	Command data
START_PAGE_BACK_FEED	26* <sup>5</sup>	[Page Start] Paper feed (back feed) (dot)	RP-E10, RP-G10 0 to -60 RP-E10 0 to -74
START_PAGE_FEED	27* <sup>5</sup>	[Page Start] Paper feed (feed) (dot)	0 to 255
START_PAGE_BUZZER	28	[Page Start] Buzzer* <sup>2</sup>	0: ON 1: OFF
START_PAGE_EXT_BUZZER	120	[Page Start] External buzzer* <sup>6</sup>	0: Pattern 1 1: Pattern 2 2: Pattern 3 3: Pattern 4 255: OFF
START_PAGE_EXT_BUZZER_RPT	121	[Print Start] External buzzer Frequency* <sup>6*7</sup>	1 to 255
END_PAGE_LOGO	30	[Page End] Logo	0: None 1: Left 2: Center 3: Right
END_PAGE_LOGO_KEY	31	[Page End] Logo keycode (2 bytes)* <sup>3</sup>	RP-E10, RP-G10 Each byte 0x20 to 0x39 RP-E10 Each byte 0x30 to 0x39
END_PAGE_CMD	35	[Page End] Custom command (128 bytes)	Command data
END_PAGE_BACK_FEED	36* <sup>5</sup>	[Page End] Paper feed (back feed) (dot)	RP-E10, RP-G10 0 to -60 RP-E10 0 to -74
END_PAGE_FEED	37* <sup>5</sup>	[Page End] Paper feed (feed) (dot)	0 to 255
END_PAGE_BUZZER	38	[Page End] Buzzer* <sup>2</sup>	0: ON 1: OFF

Property ID		Description	
.NET API	.NET API Win32 API		
END_PAGE_EXT_BUZZER	130	[Page Start] External buzzer* <sup>6</sup>	0: Pattern 1 1: Pattern 2 2: Pattern 3 3: Pattern 4 255: OFF
END_PAGE_EXT_BUZZER_RPT	131	[Print Start] External buzzer Frequency* <sup>6*7</sup>	1 to 255
END_DOC_LOGO	40	[Print End] Logo	0: None 1: Left 2: Center 3: Right
END_DOC_LOGO_KEY	41	[Print End] Logo keycode (2 bytes)* <sup>3</sup>	RP-E10, RP-G10 Each byte 0x20 to 0x39 RP-E10 Each byte 0x30 to 0x39
END_DOC_DRAWER	42	[Print End] Drawer* <sup>4</sup>	0: No use 1: Use drawer 1 2: Use drawer 2
END_DOC_DRAWER_ON	43	[Print End] ON time (x 2 ms)	1 to 255
END_DOC_DRAWER_OFF	44	[Print End] OFF time (x 2 ms)	1 to 255
END_DOC_CMD	45	[Print End] Custom command (128 bytes)	Command data
END_DOC_BACK_FEED	46* <sup>5</sup>	[Print End] Paper feed (back feed) (dot)	RP-E10, RP-G10 0 to -60 RP-E10 0 to -74
END_DOC_FEED	47* <sup>5</sup>	[Print End] Paper feed (feed) (dot)	0 to 255
END_DOC_BUZZER	48	[Print End] Buzzer* <sup>2</sup>	0: ON 1: OFF
END_DOC_EXT_BUZZER	140	[Page End] External buzzer* <sup>6</sup>	0: Pattern 1 1: Pattern 2 2: Pattern 3 3: Pattern 4 255: OFF
END_DOC_EXT_BUZZER_RPT	141	[Page End] External buzzer Frequency* <sup>6*7</sup>	1 to 255

Property ID		Description							
.NET API	.NET API Win32 API								
PAPER_SIZE	50	Paper size	<table><tr><th>Printer</th><th>Description</th></tr><tr><td>RP-F10 RP-G10</td><td>0: Letter 1: A4 2: 80 mm (72×3276 mm) 3: 58 mm (54×3276 mm) 5 to 255: Paper other than above*8</td></tr><tr><td>RP-E10</td><td>0: Letter 1: A4 2: 80 mm 3: 58 mm 5 to 255: Paper other than above*8</td></tr></table>	Printer	Description	RP-F10 RP-G10	0: Letter 1: A4 2: 80 mm (72×3276 mm) 3: 58 mm (54×3276 mm) 5 to 255: Paper other than above*8	RP-E10	0: Letter 1: A4 2: 80 mm 3: 58 mm 5 to 255: Paper other than above*8
Printer	Description								
RP-F10 RP-G10	0: Letter 1: A4 2: 80 mm (72×3276 mm) 3: 58 mm (54×3276 mm) 5 to 255: Paper other than above*8								
RP-E10	0: Letter 1: A4 2: 80 mm 3: 58 mm 5 to 255: Paper other than above*8								
ORIENTATION	51	Orientation	0: Portrait 1: Landscape						
COLOR_MODE	52	Color printing mode	0: System*9 1: Driver*9						
WATERMARK	53	Watermark	0: None 1: Upper left 2: Top center 3: Upper right 4: Left 5: Center 6: Right 7: Lower left 8: Bottom center 9: Lower right						
PRESET	60*10	Preset	<table><tr><th>Printer</th><th>Description</th></tr><tr><td>RP-F10 RP-G10</td><td>0: 80 mm Receipt 1: 58 mm Receipt 4: A4-&gt;80 mm Reduction 5: A4-&gt;58 mm Reduction 6: User setting*8</td></tr><tr><td>RP-E10</td><td>0: 80 mm Receipt Setting 1: 58 mm Receipt Setting 2: 80mm Marked Paper Setting 3: 58mm Marked Paper Setting 4: A4-&gt;80 mm Reduction Setting 5: A4-&gt;58 mm Reduction Setting 6: User setting*8</td></tr></table>	Printer	Description	RP-F10 RP-G10	0: 80 mm Receipt 1: 58 mm Receipt 4: A4->80 mm Reduction 5: A4->58 mm Reduction 6: User setting*8	RP-E10	0: 80 mm Receipt Setting 1: 58 mm Receipt Setting 2: 80mm Marked Paper Setting 3: 58mm Marked Paper Setting 4: A4->80 mm Reduction Setting 5: A4->58 mm Reduction Setting 6: User setting*8
Printer	Description								
RP-F10 RP-G10	0: 80 mm Receipt 1: 58 mm Receipt 4: A4->80 mm Reduction 5: A4->58 mm Reduction 6: User setting*8								
RP-E10	0: 80 mm Receipt Setting 1: 58 mm Receipt Setting 2: 80mm Marked Paper Setting 3: 58mm Marked Paper Setting 4: A4->80 mm Reduction Setting 5: A4->58 mm Reduction Setting 6: User setting*8								

\*1: When you are using the F / W version 1.02 or earlier, select the "Speed" except "Middle (Silent)".

\*2: Supported only by RP-E10.

\*3: Stored in the order of the higher digit to the lower digit.

- \*4: Driving the drawer both at Print Start and at Print End is not possible.
- \*5: For the same timing feed, the last setting is valid.
- \*6: Not supported in RP-E10.
- \*7: The external buzzer sounds will stop under one of the following conditions:
  - When sounds for the set number of buzzers.
  - When the cover is open.
  - When the printer command "Stop External Buzzer" is executed.
- \*8: It cannot be set. You can only get the value.
- \*9: For details of color printing mode, see "3.2 Paper/Quality" in "SII Printer Driver for Windows User's Guide".
- \*10: When this property ID is specified, some contents of other property IDs are ignored. For details of preset, see "3.3.4 Use of Preset" in "SII Printer Driver for Windows User's Guide".

## 6.9 Barcode Data Structure

---

Barcode data structures are as follows.

### 6.9.1 Win32 API

```
ST_BARDATA
{
    INT io_rlen,
    LPBYTE o_rdata,
    DWORD Reserved
}
ST_BARDATA, *PST_BARDATA;
```

#### Parameters

*io\_rlen*

Size of data to receive

The following receive data is stored when callback function is called.

<i>io_rlen</i>	Description
More than 0	The size of the received barcode data.
0	The barcode scanner is connected.
-1	The barcode scanner is not connected.

*o\_rdata*

Buffer of data to receive

Returns the address in the data array responded from the barcode scanner.

0 is entered in the data array of *o\_rdata* when *io\_rlen* is 0 or -1.

The receive data of *o\_rdata* is to be deleted at the timing when the callback function registered by a user next time is called. Copy the data of the receive data buffer to save the values.

*Reserved*

Reservation

## 6.9.2 .NET API

```
class BarcodeData
{
    int length;
    byte[] data;
    uint reserved
}
```

### Parameters

*length*

Size of data to receive

The following receive data is stored when callback function is called.

<i>length</i>	Description
More than 0	The size of the received barcode data.
0	The barcode scanner is connected.
-1	The barcode scanner is not connected.

*data*

Buffer of data to receive

Returns the data array responded from the barcode scanner.

The data array of *data* is to be empty when *length* is 0 or -1.

*Reserved*

Reservation

---

# Chapter 7 Command Definition File

---

## 7.1 Overview

---

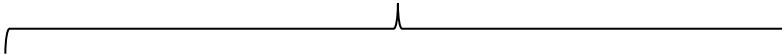
A command definition file is a text file (\*.txt) that describes transmission data in a predefined format and is saved in ANSI format and UNICODE format.

## 7.2 Format

---

The command definition file format is described.

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	Data in command name unit described according to the format. Be sure to describe it on 1 line for 1 command definition.
Command name	Describe an arbitrary command name to be specified by <b>DirectSendRead</b> . <ul style="list-style-type: none"><li>• Use ASCII characters except for "=" and "#". When any character other than ASCII characters is written, only the character is ignored.</li><li>• Command names are case-sensitive.</li><li>• Registerable command name is up to 33 bytes. When 34 bytes or more are written, 33 bytes are registered, and characters after 34 bytes are ignored.</li><li>• When describing the same command names in 1 command definition file, the first command name is registered. When the command name already registered is specified, it is ignored.</li><li>• Write "=" between the command name and the transmission data.</li></ul>

Name		Description
	Transmission data	<p>Describe printer commands and data to send to the printer.</p> <ul style="list-style-type: none"> <li>• In the transmission data, do not include the ASB setting command that disables the response. The API that gets the printer status will not work properly.</li> <li>• When specifying binary data for transmission data, describe the data in two-digit hexadecimal numbers. Separate each data with a 1-byte space. Do not include a printer command that initializes the printer other than the printer command "Initialize Printer" in the binary data. For printer initialization, see "Technical Reference". Execute <b>Reset</b> or <b>ResetPrinter</b> to perform hardware reset.</li> <li>• When specifying a string for transmission data, describe the data in ASCII characters. To specify the string, enclose it with " ".</li> <li>• The maximum transmission data size is 10240 bytes. However, the size when transmission data is specified as a string is the size after converting the specified string to binary data. Transmission data sizes of the command definition examples shown below are as follows: <ul style="list-style-type: none"> <li>• CmdName_1: 3 bytes</li> <li>• CmdName_2: 4 bytes</li> <li>• CmdName_3: 3 bytes</li> <li>• CmdName_4: 6 bytes</li> </ul> </li> </ul>
	Comment	<p>Describe the command definition etc.</p> <ul style="list-style-type: none"> <li>• Add "#" at the beginning of the comment.</li> <li>• There is no restriction on characters to be written.</li> <li>• Comments are optional.</li> </ul>

Examples of 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.

## Reference

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

## 7.3 How to Use

---

How to use the command definition file is described.

1. Create a command definition file.
2. Register the command definition file in the SDK internal memory by **SendDataFile**.
3. Specifying the command name by **DirectSendRead** executes the contents of the transmission data.

### Caution

- ◆ The registered command definition is discarded by **CloseMonPrinter**.
- ◆ When a new command definition file is registered after registering the command definition file, the registered command definition is overwritten.