SEKONIC

OPTICAL MARK READER

Windows

API Reference manual

Index

1.Introduction

This manual explains the use of API (Application Programming Interface) for USB communications (ver.1.1/2.0/3.0) control of Sekonic's OMR, SR-2300/5500/3500/6000/6500, SR-1800, SR-3500/6500/8000HYBRID, and SR-11000.
(Please chack the specification of each model about usable USB version.)
The Microsoft Windows platform environment was utilized for developing efficient OMR control applications.
Regarding further details on the platform environment, please refer to the "Environment for Operation" section.
In order to control OMR, it is necessary to connect personal computer (PC) with the OMR by a USB cable.

OMR will operate according to control commands from the PC.
Functions will be created for each command (and response), so that they can be utilized as API reference.
This document provides detailed explanation on each function, and explains the parameters that can be transferred to each function. Actual usage examples are included for reference as well.

API usage can be classified into the following 4 categories:
(1) System control : interface initialization (connection) etc.
(2) Setting Parameters : conduct necessary mark-sheet reading settings.
(3) Action Command : Commands for reading and disposing paper, etc.
(4) Data Request : Collects data such as mark sheet reading results and OMR condition, etc.

Terminology and Trademarks

Microsoft, Windows, Windows8.1, Windows10, Visual C++, Visual Basic, Visual C# are the registered Trademark of Microsoft Corporation in the U.S.A. and / or other jurisdictions. Other company names and products names related Microsoft Corporation is the resistered Trademark of Microsoft Corporation.
Intel, Core, Pentium, Celeron are the registered Trademark of Intel Corporation in the U.S.A. and / or other jurisdictions. Other company names and products names related Intel Corporation is the resistered Trademark of Intel Corporation.
C++ Builder is a registered Trademark of Embarcadero Technologies, Inc in the U.S.A. and / or other jurisdictions.
Other Conpany names and Produc Names are trade Markd or registered Trademarks in Japan / or other jurisdictions.

Copyright

On the process of Decode/Encode of JPEG file, This API uses Library made by "Independent JPEG Group".

Caution

The contents of this manual are subject to change without notice.
SEKONIC will not be liable for any damages in any way about the contents of this manual even mis mentioned.
SEKONIC will not be liable in any kinds of damage caused by the use or failure to function of the product (Including but not limited to lost profit.and incidental, special, anc consequential damage).

2. APILibrary structure

The Program Structure to control SEKONIC OMR is as following,
The Program which you develop communicates with OMR via Device Driver by using New API Library (SkDvSRseries.dll) or Old API Library (OMRAPI.dll).　You can use both API Libraries same time.
Old API Library will be　no longer updated anymore.

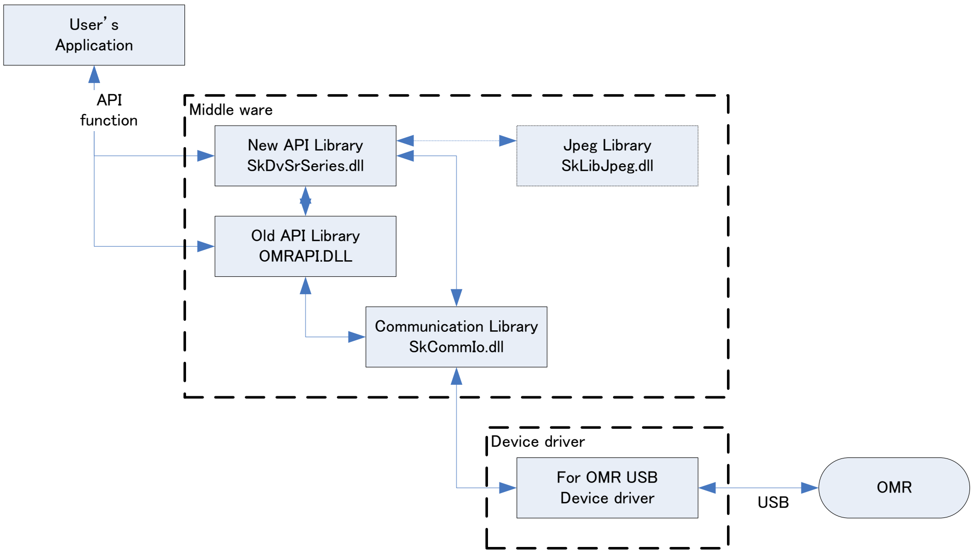

User's Application
API function
Middle ware
New API Library SkDvSrSeries.dll
Jpeg Library SkLibJpeg.dll
Old API Library OMRAPI.DLL
Communication Library SkCommIo.dll
Device driver
For OMR USB Device driver
USB
OMR

Fig 1 Program Structure

	Items	Description
1	User application	Application which developped by user or vendor.
2	New API Library (SkDvSrSeries.dll)	API Library for control of　SR-3500 / SR-6000 / SR-6500, SR-1800,SR-3500/6500/8000 HYBRID,SR-11000
3	Old API Library (OMRAPI.dll)	OMRAPI dll for control of SR-3500/SR-6000/SR-6500, SR-1800
4	Jpeg Library (SkLibJpeg.dll)	Library format for saving files in Jpeg for SR-3500/6500/8000 HYBRID.
5	Basic Communication Library (SkCommIo.dll)	Library for communication with USB Driver for OMR
6	USB Device Driver for OMR	Device Driver for communication with OMR
7	OMR	SR-3500 / SR-6000 / SR-6500, SR-1800, SR-3500/6500/8000 HYBRID, SR-11000

3.Provided files

API Library to control of SEKONIC OMR by USB communication is supplied by the following file.

3.1. APILibrary

API Library is 32bits Program.
Please develop the application in 32bits.
(1) In case of using New API Library,
User application can controls OMR by API Function of "SkDvSrSeries.dll".

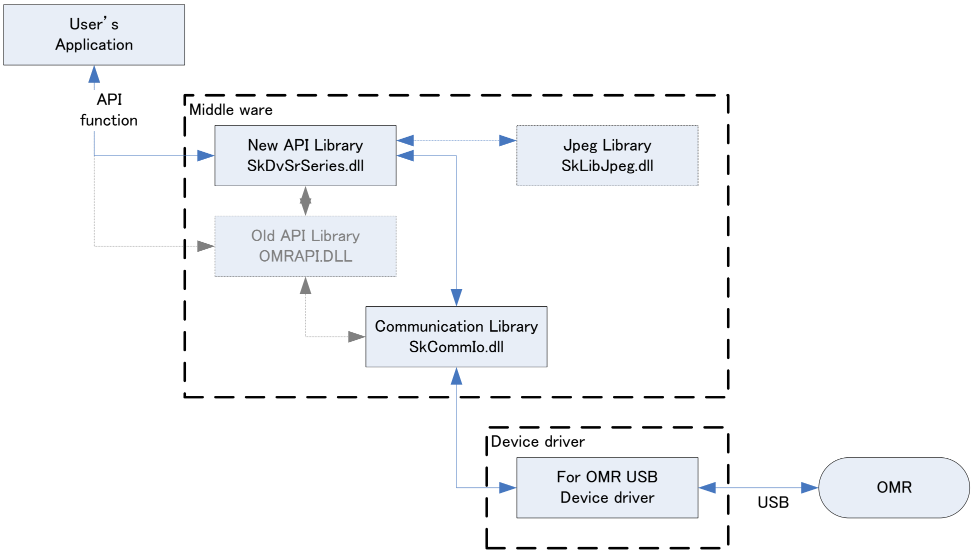

User's Application
API function
Middle ware
New API Library SkDvSrSeries.dll
Jpeg Library SkLibJpeg.dll
Old API Library OMRAPI.DLL
Communication Library SkCommIo.dll
Device driver
For OMR USB Device driver
USB
OMR

Fig 2 Program Structure by New API Library

New API Library can process the image data offered by SR-3500/6500/8000HYBRID and perform a continuous read by SR-11000.

Item	Files	description
Executable file	SkDvSrSeries.dll	The main body of Library. Dynamic link Library.
	SkCommIo.dll	DLL for access SkDvSrSeries.dll.
	SkLibJpeg.dll	DLL for save Image file as JPG.
Generating file for Execution	SkDvSrSeries.log	Generated when executing Library could be 4Kbyte at maximum.
	SkCommIo.log	
Header file	SkDvSr11000Api.h	Library's header file.
	SkDvSr3500ImgApi.h	Library's header file forimage.
	SkDvSrOptionApi.h	Library's header file for Bar Code Reader and Printer.
	SkDvSrBaseApi.h	Library's header file base for OMR
	SkDvSr11000Prm.h	Library header file only for continuous reading
	SkDvSrSeriesPrm.h	Header file defining Constant and Structure.
	SkDvSrSeriesErr.h	Header file defining Status Constant of SkDvSrSeries.dll
	SkCommIoErr.h	Header file defining Status Constant of SkCommIo.dll
Library	SkDvSrSeries.lib	Introducing Library for linking when using New type of API Library

(2) In the case of applications using " SkDvSr3500Img.dll ".
SkDvSr3500Img.dll in the CD is a wrapper DLL.

If an application uses SkDvSr3500Img.dll that was provided in past, you can control SkDvSrSeries.dll with this wrapper DLL.
If you develop an application newly, control SkDvSrSeries.dll directly without using the wrapper DLL.

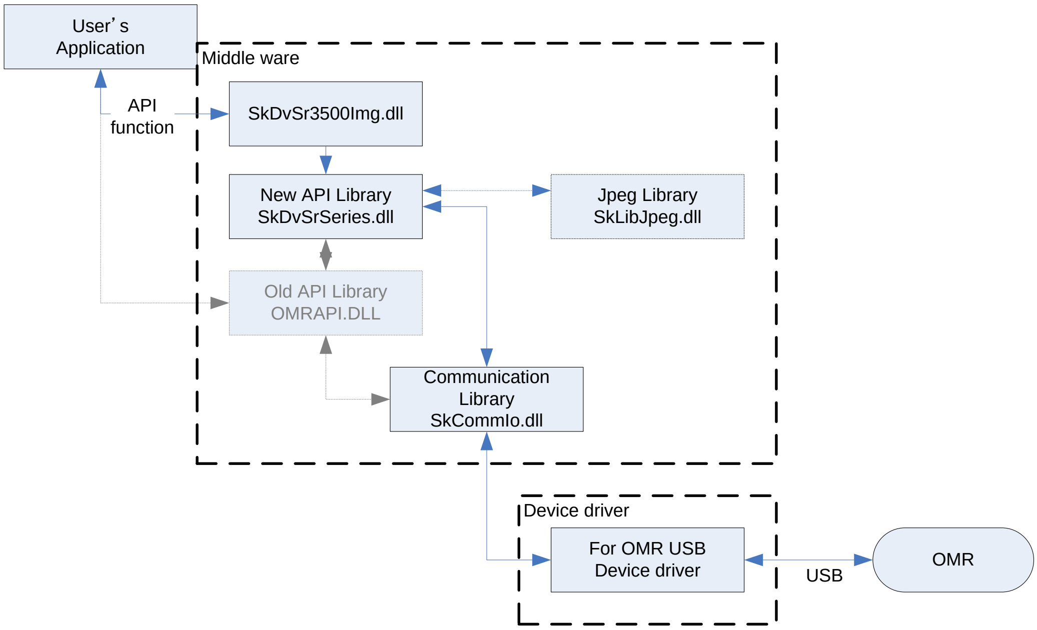

User's Application
Middle ware
API function
SkDvSr3500Img.dll
New API Library SkDvSrSeries.dll
Jpeg Library SkLibJpeg.dll
Old API Library OMRAPI.DLL
Communication Library SkCommIo.dll
Device driver
For OMR USB Device driver
USB
OMR

Fig 3 In the case of applications using " SkDvSr3500Img.dll "

(3) In case of using Old API Library,
User application controls OMR by calling API Functions of [OMRAPI.DLL].
[OMRAPI.DLL] is not included in the CD. Please use the old CD.

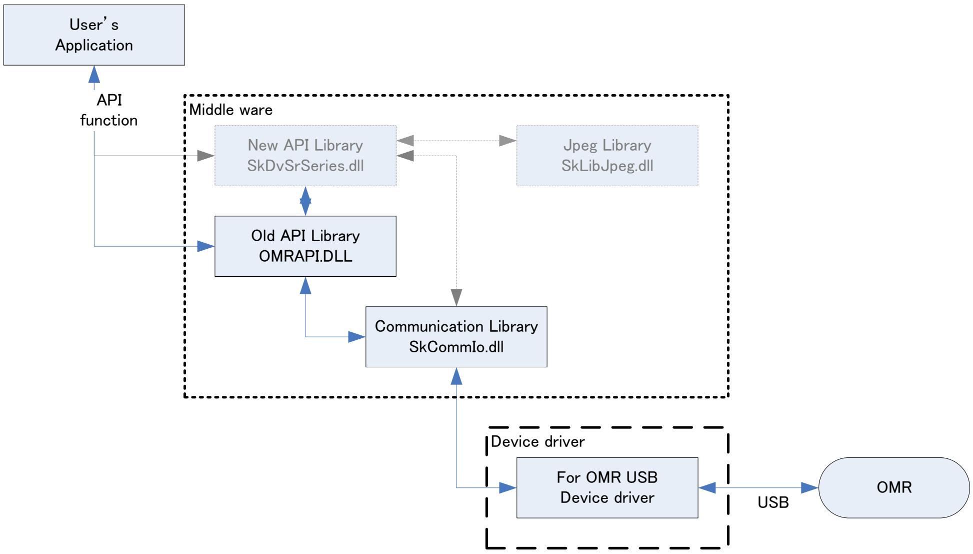

User’s Application
API function
Middle ware
New API Library SkDvSrSeries.dll
Jpeg Library SkLibJpeg.dll
Old API Library OMRAPI.DLL
Communication Library SkCommIo.dll
Device driver
For OMR USB Device driver
USB
OMR

Fig 4 Program structure using Old API Library

Old API Library cannot process image data and continuous read.
Former Ver. than Old type of API Library Ver4.1 was operated by only "OMRAPI.DLL", as from Ver.5.0, it needs "SkCommIo.dll". But processing quality is almost same with each other.

Items	File	description
Executable file	OMRAPI.DLL	Main body dynamic linking library with API function.
	SkCommIo.dll	Dynamic Link Library for "OMRAPI.DLL"
Generating file for Execution	SkCommIo.log	Generated when executing Library could be 4Kbyte at maximum.
Header file	OMRAPI.h	Header file of OMRAPI.DLL
Library	OMRAPI.LIB	The introducing library for linking when using OMR API.DLL

(4) In case of using NEW / OLD API Library both.
When processing SR-3500/6500HYBRID's image data and using Old API Library, Please use both Libraries (New & Old) at the same time.

Both libraries (new API/OLD API) need to execute application software or programs.

3.2. Device Driver

Hereunder is the USB device driver installer of SEKONIC CORPORATION
Execute the following file, to install the USB driver

DriverInstaller*.*.exe

1:".*" is the version of USB driver to be installed
*2:USB driver may be renewal

3.3. Re-distributable File

Re-distribution of the following filereading be allowed only with the develop the software.
Cannot distribute just only following files.

Driver installer*.*.exe
SkDvSrseries.dll
OMRAPI.DLL
SkCommIo.dll
(SkDvSr3500Img.dll)
SkLibJpeg.dll *:Caution

*:Caution

This Jpeg Library (SkLibJpeg.dll) is using the program of "Independent JPEG Group". In case re-distribute these dlls, need to mention these are using the program of "Independent JPEG Group"

4. Environment for Operation

4.1. Environment for Operation

This API Library can operate in the following environment.

OS
Microsoft Windows 8.1 32bit (x86) and 64bit (x64) Version.
Microsoft Windows 10 32bit (x86) and 64bit (x64) Version.
*: Operate in SYSWOW64 on 64bit PC
*: Not Supported virtual machine by Client Hyper-V
*: Microsoft support is finished, would remove it from the operating environment.

CPU
Intel Core i3 2GHz or greater, or equivalent CPU. (Recommendation: Core i5 or greater)

Hard disk space
vacant space more than 10MB.
Attn*: Not included of capacities for User application and Data filing.

Memory
1GBor larger. (Recommendation:2GB or larger)
*: System requirements are less than the OS.
*: 2GB or larger if you perform image capturing (Recommendation: 4GB or greater)
*:Depending on a type of processing performed by application software, there is a possibility that higher specification is required than the recommending environment.

Device Driver
Need installed provided Device Driver.

Interface
USB2.0 or USB3.0
Refer ro a command reference manual about usable USB version in each model.

Simultaneously connectable number of units
1 unit

Note
- Do not control OMR equipment by LCD display of OMR equipment while control via this API library.
- Do not use other application software while control OMR equipment via this API library. Capability of this API library may be drastically go down

4.2. Development tools

This DLL uses Windows standard API in internal.
As this DLL doesn't use other library third-party fixed, it can utilize it from usual Windows application development tool (Visual C++、Visual Basic、Borland C++ Builder etc…).

When use DLL from Development tool, please refer to each tool's manual or study aid.
Also, it can write DLL calling routine during coding by using Load Library Function of Win32API.

In actually writing codes, you write to include Header file and Library according the library you use.

When executing application, please put "OMRAPI.DLL / SkDvSrSeries.dll" and "SkCommIo.dll" into same folder as Execution file or Pass-fixed folder.

5.API Reference（New API library）

Here is "SkSrSeries.dll" (SkDvSrseries.dll) API reference.
Able to control SR-3500/6000/6500, SR-1800, SR-3500/6500/8000 HYBRID, SR-11000 by USB command.

5.1. Functions

Functions vary depending on model.

Functions		Description
Image capturing		For HYBRID series Method of this function is different by model, so usable API function is also different.
	SR-3500/6500 HYBRID	Standard image processing function.
	SR-8000 HYBRID	High speed image processing function.
Reading mode		Some models can perform the continuous reading and specific API function for this reading is used.
	Real time reading	For all models Processing of reading, data capturing and discharging paper is implemented by page
	Continuous reading	Proceed reading continuously without stop
Barcode reading		Read and recognize barcode (optional)
	SR-3500 series	For SR-3500/6000/6500, SR-6500HYBRID, SR-11000 Barcode recognition by dedicated barcode reader (optional)
	SR-8000HYBRID	For SR-8000 HYBRID Barcode recognition by CIS image capture (optional)

5.2. List of API

Pre-fixing of the name of API Function is based on the following charts.

API Pre-Fix	Description
SkDv_	Function which controls basically in this library and make a series of controls.
SkDv_Req	Function which communicate with Device and do controls. There is a case having some commands when communicating with Device.
SkDv_Get/SkDv_Set	Function which controls saving data inside this library. However, it would start communication in case there is un-captuable data unless doing communication with Device.
SkCd_	Functions managing one command. However, in case of demand for reading sheets, it would be separated because of several responses.

	Name	API	Description
Basic functions	Get Module info.	SkDv_GetModuleInfo	Get Name and Version of this library, Not needed to open if Library loaded.
	Open	SkDv_OpenSingle	Open control of OMR. Able to Respond in case of 1 unit connection only, and when there is Device already opening, returns the ERROR.
	Open with (OMRAPI)	SkDv_OpenWithOmrapi	Open when using "SkDvSrseries.dll" with " OMRAPI.dll".

	Name	API	Description
	Close	SkDv_Close	Execute when all handle is done. Open memories for each type of operations.
	Initialize Requirement	SkDv_ReqInit	Initialize the operation condition of Device. Image data and internal operation data initialized. Also Soft resetting is done toward Device.
	Get Device info.	SkDv_GetInfo	Get information of Ver. and available Functions.
	Get Operating mode	SkDv_GetMode	Get Operating mode. Operating mode is to be saved in this library.
	Set of Operating mode	SkDv_SetMode	Set Operation mode.Operation mode is to be saved in this library.
	Get Mark reading condition	SkDv_GetMarkConf	Get the condition of Mark reading operation. The condition of operation is saved in this library.
	Set of Mark reading condition	SkDv_SetMarkConf	Set the condition of Mark reading operation. The condition of operation is saved in this library.
	Request Status info.	SkCd_GetStatus	Get the Status of Device by communication.
	Request Sensor info.	SkCd_GetSensor	Get the sensor information. by communication.
	Request Reading (Mark only)	SkDv_ReqFeedMarkSheet	Read Marks. Not getting Image data.
	Request Eject	SkDv_ReqEjectForm	Eject the sheets.
	Request Clear Error	SkDv_ReqClearError	Communicate with Device to Clear Error.
	Request get Mark data	SkDv_ReqGetMarkData	Get Mark data by communication.
Expanded function	Request communication to get Device information.	SkDv_ReqGetInfo	Communicate with device to get infomation.
	Request communication to get Operation mode.	SkDv_ReqGetMode	Communicate with device to get Operation mode.
	Request communication to setting Operation mode	SkDv_ReqSetMode	Communicate with device to set up Operation mode.
	Request communication to get Mark condition	SkDv_ReqGetMarkConf	Communicate with device to get Operation condition of Mark reading.
	Request communication to set Mark condition.	SkDv_ReqSetMarkConf	Communicate with device to set Operation condition of Mark reading.

	Name	API	Description
Image Reading	Get Image reading condition	SkDv_GetImageConf	Get the condition of Image reading operation. The condition of operation is saved in this library.
	Set of Image reading condition	SkDv_SetImageConf	Set the condition of Image reading operation. The condition of operation is saved in this library.
	Get Image reading condition (extra)	SkDv_GetImageConfEx	Get the extra condition of Image reading operation. The condition of operation is aved in this library.
	Set of Image reading condition (extra)	SkDv_SetImageConfEx	Set the extra condition of Image reading operation. The condition of operation is saved in this library.
	Request communication to get Image condition.	SkDv_ReqGetImageConf	Communicating to get Image condition, to save it into Library.
	Request communication to set Image condition.	SkDv_ReqSetImageConf	Communicating to set Image condition, to save it into Library.
	Request Reading (Mark and Image)	SkDv_ReqScanForm	Reading Mark sheet, and capturing Image data.If the communication for getting Image/setting Condition/CIS standard data has not been done, the communication will be done before Reading.
	Request get sheet size	SkDv_GetFormSize	Getting the sheet size from Image data in Middleware.
	Set Information for detecting the elongation of the read image	SkDv_SetImage ElongationDetectInfo	Information for detecting the elongation of the read image is set.
	Get Information for detecting the elongation of the read image	SkDv_GetImage ElongationDetectInfo	Information for detecting the elongation of the image stored in the middleware is acquired.
	Saving the Image data	SkDv_SaveImageData	Saving the Image data in this library into File. Cutout position, rotation and file format can be specified.
	Saving the Image data by other threads	SkDv_SaveImageDataThread	Save the Image data which is keep in this library by other Threads.
	Confirm running Threads	SkDv_IsThreadRunning	Confirm other Threads made by the function of SkDv_SaveImageDataThread" has closed.
	Generating Image data	SkDv_CreateImage	Secure memory and store image data. Cutout position, rotation, and the like can be designated.
	Destroying Image data	SkDv_DestroyImage	Destroying the Image data generated at " SkDv_CreateImage"

	Name	API	Description
Continuous reading	Request ID for Set layout management	SkDv_ReqSetLayoutManage	Setup ID for perform window control
	Finish layout setup	SkDv_ReqSetLayoutManageTerminate	Finish layout setup (ID data setting, windows area setting) of window control.
	Request Form ID setting	SkDv_ReqSetLayoutId	ID of a form is set up and enables to set up window areas.
	Request window area setting	SkDv_ReqSetWindowArea	Setup window area.
	Clear Layout setting	SkDv_ReqClearLayout	Clear all contents of the layout setting
	Poll print & eject	SkDv_PollingPrintEject	Use for non multi-thread case like VB6.0 to call back print & eject function
	Register call back function of print & eject	SkDv_RegistPrintEjectCallback	Register the function to call back. Judge and print & eject based on registered function
	Call back function of print & eject	FN_SK_DV_CONTFEED_CALLBACK	Definition of call back function
	Get mark data for call back continuous reading	SkDv_GetContDataMarks_Now	Use only for call back function of print & eject. Get mark data.
	Get barcode data (read number) for call back continuous reading	SkDv_GetContDataBarcodesCount_Now	Use only for call back function of print & eject. Get number of read barcode.
	Get barcode data (data) for call back continuous reading	SkDv_GetContDataBarcodesData_Now	Use only for call back function of print & eject. Get data of specified number in read barcode.
	Request LCD Panel	SkDv_ReqSetPanelUserEnable	Specified character shows on the LCD display and prohibit operation by LCD button
	Request Panels switch statement	SkDv_ReqGetPanelUserSwitch	Get the state of the panel switch of OMR.
	Request Fixed printed characters for continuous reading	SkDv_ReqSetPrintFixString	Set the String(fixed) for Print
	Request printed sequence number for continuous reading	SkDv_ReqSetContFeedPrint	The consecutive numbers of continuation reading -- printing is set.
	Request start continuous reading	SkDv_ReqContFeedSheet	Request start Continuausn reading
	Request Stop Continuous Reading	SkDv_ReqContFeed_Cancel	Request stop Continuous reading
	Confirm Continuous reading	SkDv_IsContFeedRunning	Confirm continuation reading is operating.
	Confirm Continuous reading is finished	SkDv_IsContFeedDataFinished	Confirm continuous reading is finished, and all data is transferred completely
	Confirm Continuous reading data storage	SkDv_ExistDoneData	Confirm whether the data is stored after start continuous reading.
	Get feed condition	SkDv_GetContFeedCount	Get the number of sheets “worker thread” of continuous reading received.

	Prepare Continuous reading	SkDv_PrepareContData	The data which the “worker thread” of continuation reading received is acquired.
	Get the number of data	SkDv_GetContDataNumber	Get the current data number (number of sheets) of continus reading data
	Get mark data by Continuous reading	SkDv_GetContDataMarks	Get the mark data prepared by“SkDv_PrepareContData”
	Get Barcode data by continuous reading	SkDv_GetContDataBarcodes	Get the barcode data prepared by“SkDv_PrepareContData”
	Get the number of Barcode data by continuous reading	SkDv_GetContDataBarcodesCount	Get the number of barcode data prepared by“SkDv_PrepareContData”
	Get each barcode datas by continuous reading	SkDv_GetContDataBarcodesData	Get the each barcode datas prepared by “SkDv_PrepareContData”
	Save image file by continuous reading	SkDv_SaveContJpegImage	Save Jpeg file prepared by continuous reading data preparation as specified name.
	Get result by continuous reading	SkDv_GetContDataSheetResult	Get result data prepared by the“SkDv_PrepareContData”
	Get result of continuous reading	SkDv_GetContFeetResult	Get the result of continuous reading

	Name	API	Description
Option(Barcode)	Get reading condition of barcode	SkDv_GetBcrConf	Get the reading condition of barcode stored into library
	Set reading condition of barcode	SkDv_SetBcrConf	Set the reading condition of barcode stored into library
	Get reading condition of barcode (extra)	SkDv_GetBcrConfEx	Get the extra reading condition of barcode stored into library
	Set reading condition of barcode (extra)	SkDv_SetBcrConfEx	Set the extra reading condition of barcode stored into library
	Request to get read setting of Barcode	SkDv_ReqGetBcrConf	Get the reading condition of barcode.
	Request to set read setting of Barcode	SkDv_ReqSetBcrConf	Set the reading condition of barcode.
	Request get the number of barcodes	SkDv_ReqGetBcrDataCount	Get the number of barcodes read
	Request get the barcode datas	SkDv_ReqGetBcrData	Get each barcode datas
Option(Printer)	Get printer setting	SkDv_GetPrinterConf	Get printer setting into Library
	Set printer setting	SkDv_SetPrinterConf	Set printer setting into Library.
	Set printing string to device	SkDv_ReqPrintString	Set the print string into device.
	Get printing string from device	SkDv_ReqGetPrinterConf	Get the print string into device
	Request set printer setting to device	SkDv_ReqSetPrinterConf	Set the printer setting to device

5.3. Constant

5.3.1. Outline status code

Return value of API is based on "SkDvStatus" type.
SkDvStatus is defined as the following…

```
        typedef     DWORD   SkDvStatus;
```

In case of setting Status code "0xAABBCCCC", each AA,BB,CCCC mean the following.

<table>
<tr><th></th><th>Position of bit</th><th>hexadecimal number</th><th>Description</th></tr>
<tr><td></td><td>31～24</td><td>0xAA------</td><td>Fixed (0x20)</td></tr>
<tr><td></td><td>23～16</td><td>0x--BB----</td><td>Code representing generated Device and Library
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0x2032----</td><td>Status code of this library</td></tr>
<tr><td>0x2021----</td><td>Status code of lower communication library</td></tr>
<tr><td>0x2002----</td><td>Status code of Device</td></tr>
</table></td></tr>
<tr><td></td><td>15～ 0</td><td>0x----CCCC</td><td>Code of Device and Library</td></tr>
</table>

There are two type of status code of Device "ST1"and"ST2", prior one would be notified.
In Status code of Library, the prior 2byte is fixed to "0x2002", and the lower 2 byte is converted as follows and would be notified.
However, In case Status code (ST1 and ST2) = "00", it wouldn't notified as Status code of Library because it is normal.

<table>
<tr><th></th><th>Position of bit</th><th>binary digits</th><th>Description</th></tr>
<tr><td>1</td><td>15～11</td><td>0bDDDDD-----------</td><td>The first character of Status code (STA).
Converted as the following chart.
<table>
<tr><th>Converted value</th><th>Character of Status code</th><th>ASCII Code</th></tr>
<tr><td>0b00000</td><td>0</td><td>0x30</td></tr>
<tr><td>0b00001</td><td>A</td><td>0x41</td></tr>
<tr><td>≀</td><td>≀</td><td>≀</td></tr>
<tr><td>0b11010</td><td>Z</td><td>0x5A</td></tr>
<tr><td>0b11111</td><td>@</td><td>0x40</td></tr>
</table></td></tr>
<tr><td>2</td><td>10～ 7</td><td>0b-----EEEE-------</td><td>The second character of Status code (STA).
Converted as the following chart.
<table>
<tr><th>Converted value</th><th>Character of Status code</th><th>ASCII Code</th></tr>
<tr><td>0b0000</td><td>0</td><td>0x30</td></tr>
<tr><td>≀</td><td>≀</td><td>≀</td></tr>
<tr><td>0b1001</td><td>9</td><td>0x39</td></tr>
<tr><td>0b1111</td><td>@</td><td>0x40</td></tr>
</table></td></tr>
<tr><td>3</td><td>6～ 5</td><td>0b---------FF-----</td><td>Error description information.
Converted as same as the first character of Status code (STA).</td></tr>
<tr><td>4</td><td>4～ 0</td><td>0b-----------GGGGG</td><td>Error description information.
Converted as same as the first character of Status code (STA).</td></tr>
</table>

5.3.2. List of Status code

○[The Parts of Status code]

Constant	Value	Description
SK_STS_MODULE_MASK	0xffff0000	Bit mask of Module definition sector
SK_STS_MODULE	0x20310000	Main Module
SK_STS_MODULE_IO	0x20210000	The lower basic communication library
SK_STS_DEVICE_3500	0x20020000	Error from Device

○Status code of Library

Constant	Value	Description
SKDV_STS_SUCCESS	0x00000000	Normal
SKDV_STS_EXECUTE_ERR	0x20310001	WindowAPI Execution error
SKDV_STS_CREATE_EVENT_ERR	0x20310006	Create Event Error
SKDV_STS_CREATE_THREAD_ERR	0x20310007	Create Thread Error
SKDV_STS_USED_DEVICE	0x2031000E	Device is using
SKDV_STS_WAIT_TIMEOUT	0x2031000F	Timeout
SKDV_STS_HANDLE_ERR	0x20311001	Handle Error
SKDV_STS_MEM_ALLOC_ERR	0x20311002	Memory Allocation Error
SKDV_STS_LOAD_SUBLIB_ _ERR	0x20311003	Load Sub library Error
SKDV_STS_PARAM_ERR	0x20311010	Parameter Incorrect
SKDV_STS_PARAM_ERR1	0x20311011	Parameter 1 Incorrect
SKDV_STS_PARAM_ERR2	0x20311012	Parameter 2 Incorrect
SKDV_STS_PARAM_ERR3	0x20311013	Parameter 3 Incorrect
SKDV_STS_PARAM_ERR4	0x20311014	Parameter 4 Incorrect
SKDV_STS_PARAM_ERR5	0x20311015	Parameter 5 Incorrect
SKDV_STS_PARAM_ERR6	0x20311016	Parameter 6 Incorrect
SKDV_STS_PARAM_ERR7	0x20311017	Parameter 7 Incorrect
SKDV_STS_PARAM_ERR8	0x20311018	Parameter 8 Incorrect
SKDV_STS_PARAM_ERR9	0x20311019	Parameter 9 Incorrect
SKDV_STS_FUNCTION_ERR	0x2031101F	API Function Error
SKDV_STS_PARAM_ERR_SCANNINGSIDE	0x20311011	Parameter Error Back side reading
SKDV_STS_PARAM_ERR_SCAN_CTRL	0x20311012	Parameter Error Reading Control
SKDV_STS_PARAM_ERR_FORMLENGTH_MAX	0x20311014	Parameter Error MAX Form Length
SKDV_STS_PARAM_ERR_FORMLENGTH_MIN	0x20311015	Parameter Error MIN Form Length
SKDV_STS_PARAM_ERR_CIS_LIGHT	0x20311016	CIS Light Incorrect(+iFace)
SKDV_STS_PARAM_ERR_CIS_LIGHT_F	0x20311016	CIS Light error (F)
SKDV_STS_PARAM_ERR_CIS_LIGHT_B	0x20311017	CIS Light error(B)
SKDV_STS_PARAM_ERR_RESOLUTION	0x20311018	Resolution error
SKDV_STS_CIS_BASE_DATA_ERR	0x20311021	CIS Base data error
SKDV_STS_NO_DATA	0x20311100	No Data

Constant	Value	Description
SKDV_STS_ROTATION_ERR	0x20311102	Rotation Error
SKDV_STS_IMAGE_PROCESS_ERR	0x203111ff	Image Process Error
SKDV_STS_FILE_ERR	0x20311200	File Error
SKDV_STS_FILE_SAVE_ERR	0x20311201	File Save Error
SKDV_STS_FILE_LOAD_ERR	0x20311211	File Load Error
SKDV_STS_FILE_FORMAT_ERR	0x20311231	File Format Error
SKDV_STS_DV_IMAGE_ERR	0x20311301	Failure image data Transfer (from OMR device)
SKDV_STS_NO_IMAGE	0x20311311	No Image
SKDV_STS_OVER_LENGTH_IMAGE	0x20311312	Image data Overloaded
SKDV_STS_DIFF_IMAGE_WIDTH	0x20311313	Image Width Different
SKDV_STS_DIFF_IMAGE_HEIGHT	0x20311314	Image Height different
SKDV_STS_IMAGE_LENGTH_ERR	0x20311315	Image Length Error
SKDV_STS_IMAGE_ELONGATION_ERR	0x20311316	Image Elongation Error
SKDV_STS_RECVED_NAK_CMD	0x20312101	NAKreceived (Command Error)
SKDV_STS_RECVED_NAK_PRM	0x20312102	NAKreceived (Parameter Error)
SKDV_STS_RECVED_NAK_NON	0x20312103	NAKreceived (Non Support)
SKDV_STS_RECVED_NAK_OPE	0x20312104	NAKreceived (LCD panel operation)
SKDV_STS_RECVED_NAK_DOING	0x20312105	NAKreceived (Under Operation)
SKDV_STS_RECVED_NAK_UNKNOWN	0x2031210F	NAKreceived (Unknown)
SKDV_STS_RECV_ACK_TIMEOUT	0x20312200	Timeout (ACKreceived)
SKDV_STS_RECV_RES_TIMEOUT	0x20312201	Timeout (Response received)
SKDV_STS_DIFF_ACK_CMD	0x20312210	CMD code of ACK is Different
SKDV_STS_DIFF_RES_CMD	0x20312211	CMD code of response is Different
SKDV_STS_SHORT_RESPONSE_ERR	0x20312220	The length of response is short
SKDV_STS_LONG_RESPONSE_ERR	0x20312221	The length of response is long
SKDV_STS_RESPONSE_FORMAT_ERR	0x20312222	Incorrect response format
SKDV_STS_DIFF_PRM	0x20312223	Different Parameter
SKDV_STS_CALIB	0x20313000	Calibration Error

Constant	Value	Description
SKDV_STS_CALIB_CHK_BG_EVEN_BL_F_ERR	0x20313010	Different the data when LED off (Front)
SKDV_STS_CALIB_CHK_BG_EVEN_BL_B_ERR	0x20313011	Different the data when LED off (Back)
SKDV_STS_CALIB_CHK_BG_EVEN_WH_F_ERR	0x20313012	Different the data when LED On (Front)
SKDV_STS_CALIB_CHK_BG_EVEN_WH_B_ERR	0x20313013	Different the data when LED On (Back)
SKDV_STS_CALIB_CHK_BG_DIFF_F_ERR	0x20313014	Gap is big between LED ON / OFF (Front)
SKDV_STS_CALIB_CHK_BG_DIFF_B_ERR	0x20313015	Different Parameter
SKDV_STS_CALIB_CIS_BS_BLACK_EVEN_F_ERR	0x20313020	Calibration Error
SKDV_STS_CALIB_CIS_BS_BLACK_EVEN_B_ERR	0x20313021	Different the data when LED off (Front)
SKDV_STS_CALIB_CIS_BS_WHITE_EVEN_F_ERR	0x20313022	Different the data when LED off (Back)
SKDV_STS_CALIB_CIS_BS_WHITE_EVEN_B_ERR	0x20313023	Different the data when LED On (Front)
SKDV_STS_CALIB_CIS_BS_RED_EVEN_F_ERR	0x20313024	Different the data when LED On (Back)
SKDV_STS_CALIB_CIS_BS_RED_EVEN_B_ERR	0x20313025	Gap is big between LED ON / OFF (Front)
SKDV_STS_CALIB_CIS_BS_GREEN_EVEN_F_ERR	0x20313026	Different Parameter
SKDV_STS_CALIB_CIS_BS_GREEN_EVEN_B_ERR	0x20313027	Calibration Error
SKDV_STS_CALIB_CIS_BS_BLUE_EVEN_F_ERR	0x20313028	Different the data when LED off (Front)
SKDV_STS_CALIB_CIS_BS_BLUE_EVEN_B_ERR	0x20313029	Different the data when LED off (Back)
SKDV_STS_CALIB_CIS_BS_BLACK_UPPER_F_ERR	0x20313030	Different the data when LED On (Front)
SKDV_STS_CALIB_CIS_BS_BLACK_UPPER_B_ERR	0x20313031	Different the data when LED On (Back)
SKDV_STS_CALIB_CIS_BS_WHITE_RANGE_F_ERR	0x20313032	White Level calibration Range (Front)
SKDV_STS_CALIB_CIS_BS_WHITE_RANGE_B_ERR	0x20313033	White Level Calibration Range (Back)
SKDV_STS_CALIB_CIS_BS_RED_RANGE_F_ERR	0x20313034	Red Level calibration Range (Front)
SKDV_STS_CALIB_CIS_BS_RED_RANGE_B_ERR	0x20313035	Red Level calibration Range (Back)
SKDV_STS_CALIB_CIS_BS_GREEN_RANGE_F_ERR	0x20313036	Green Level calibration Range (Front)
SKDV_STS_CALIB_CIS_BS_GREEN_RANGE_B_ERR	0x20313037	Green Level calibration Range (Back)
SKDV_STS_CALIB_CIS_BS_BLUE_RANGE_F_ERR	0x20313038	Blue Level calibration Range (Front)

Constant	Value	Description
SKDV_STS_CALIB_CIS_BS_BLUE_RANGE_B_ERR	0x20313039	Blue Level calibration Range (Back)
SKDV_STS_CALIB_CHK_PL_UPPER_F_ERR	0x20313040	Platen detects too White (Front)
SKDV_STS_CALIB_CHK_PL_UPPER_B_ERR	0x20313041	Platen detects too White (Back)
SKDV_STS_CALIB_LS_UPPER_LEVEL_ERR	0x20313101	Left SkewLevel High
SKDV_STS_CALIB_LS_UNDER_LEVEL_ERR	0x20313102	Left Skew Level Low
SKDV_STS_USER_CANCEL	0x2031f001	User Cancel
SKIO_STS_SUCCESS	0x00000000	Normal.
SKIO_STS_EXECUTE_ERR	0x20210001	WindowAPI Execution error.
SKIO_STS_NONE_DEVICE	0x20210002	No suitable Device.
SKIO_STS_HANDLE_ERR	0x20210003	Handle error.
SKIO_STS_PARAM_ERR	0x20210004	Parameter error
SKIO_STS_MEM_ALLOC_ERR	0x20210005	Fail to secure memory
SKIO_STS_CREATE_EVENT_ERR	0x20210006	Fail to make event
SKIO_STS_USED_DEVICE	0x2021000e	Object Device is under use
SKIO_STS_WAIT_TIMEOUT	0x2021000f	Time out

○[Status code of Device]

Only toward Status code information. (ST1/ST2) , Constant is defined.
Description code is not defined. Please use by masking bit.

Constant	Value	Description
SKDV_STS_DEVICE_STA_MASK	0xffffff80	Status Mask from OMR device
SKDV_STS_DEVICE_ST_A1	0x20020880	"A1" Memory Error 1 (Internal Memory Error)
SKDV_STS_DEVICE_ST_A2	0x20020900	"A2" Memory Error 2 (Internal Memory Error)
SKDV_STS_DEVICE_ST_A3	0x20020980	"A3" Hopper Drive Error
SKDV_STS_DEVICE_ST_A4	0x20020A00	"A4" Download Error (Main Unit)
SKDV_STS_DEVICE_ST_A5	0x20020A80	"A5" Sensor Type Error
SKDV_STS_DEVICE_ST_A6	0x20020B00	"A6" Option Error
SKDV_STS_DEVICE_ST_A8	0x20020C00	"A8" Power Supply Error
SKDV_STS_DEVICE_ST_B1	0x20021080	"B1" Communication Error

Constant	Value	Description
SKDV_STS_DEVICE_ST_B2	0x20021100	"B2" Internal COM Error (No response Read Sensor)
SKDV_STS_DEVICE_ST_B3	0x20021180	"B3" Memory Error (in Read Sensor)
SKDV_STS_DEVICE_ST_B4	0x20021200	"B4" Adjust Value Error (Read Sensor)
SKDV_STS_DEVICE_ST_B5	0x20021280	"B5" Download Error (Read Sensor)
SKDV_STS_DEVICE_ST_B6	0x20021300	"B6" Internal Error (Read Sensor)
SKDV_STS_DEVICE_ST_B7	0x20021380	"B7" Version Error(Read Sensor)
SKDV_STS_DEVICE_ST_C1	0x20021880	"C1" Communication Error (Barcode Unit)
SKDV_STS_DEVICE_ST_C2	0x20021900	"C2" Internal COM Error (Barcode Unit)
SKDV_STS_DEVICE_ST_C3	0x20021980	"C3" Memory Error (Barcode Unit)
SKDV_STS_DEVICE_ST_C4	0x20021A00	"C4" Sensor Error (Barcode Unit)
SKDV_STS_DEVICE_ST_C5	0x20021A80	"C5" Download Error (Barcode Unit)
SKDV_STS_DEVICE_ST_C6	0x20021B00	"C6" Internal Error (Barcode Unit)
SKDV_STS_DEVICE_ST_C7	0x20021B80	"C7" Version Error (Barcode Unit)
SKDV_STS_DEVICE_ST_D1	0x20022080	"D1" Communication Error(Printer Unit)
SKDV_STS_DEVICE_ST_D2	0x20022100	"D2" Internal COM Error (Printer Unit)
SKDV_STS_DEVICE_ST_D3	0x20022180	"D3" Memory Error (Printer Unit)
SKDV_STS_DEVICE_ST_D4	0x20022200	"D4" Download Error (Printer Unit)
SKDV_STS_DEVICE_ST_D5	0x20022280	"D5" Internal Error (Printer Unit)
SKDV_STS_DEVICE_ST_D6	0x20022300	"D6" Version Error (Printer Unit)
SKDV_STS_DEVICE_ST_E1	0x20022880	"E1" Communication Error (Stacker Unit)
SKDV_STS_DEVICE_ST_E2	0x20022900	"E2" Internal COM Error (Stacker Unit)
SKDV_STS_DEVICE_ST_E3	0x20022980	"E3" Memory Error (Stacker Unit)
SKDV_STS_DEVICE_ST_E4	0x20022A00	"E4" Download Error (Stacker Unit)
SKDV_STS_DEVICE_ST_E5	0x20022A80	"E5" Internal Error (Stacker Unit)
SKDV_STS_DEVICE_ST_E6	0x20022B00	"E6" Version Error (Stacker Unit)

Constant	Value	Description
SKDV_STS_DEVICE_ST_E7	0x20022B80	"E7" Drive Error (Stacker)
SKDV_STS_DEVICE_ST_J1	0x20025080	"J1" Download Error (Image Sensor Unit)
SKDV_STS_DEVICE_ST_J2	0x20025100	"J2" Internal COM Error 1 (Image Sensor Unit)
SKDV_STS_DEVICE_ST_J3	0x20025180	"J3" Internal COM Error 1 (Image Sensor Unit)
SKDV_STS_DEVICE_ST_J4	0x20025200	"J4" Memory Error 1 (Image Sensor Unit / Flash Rom Read Error)
SKDV_STS_DEVICE_ST_J5	0x20025280	"J5" Memory Error 2 (Image Sensor Unit / Flash Rom Load Error)
SKDV_STS_DEVICE_ST_J6	0x20025300	"J6" Memory Error 3 (Image Sensor Unit / E2PROM Read Error)
SKDV_STS_DEVICE_ST_J7	0x20025380	"J7" Memory Error 4 (Image Sensor Unit / E2PROM Load Error)
SKDV_STS_DEVICE_ST_J8	0x20025400	"J8" IC Error (Image Sensor Unit)
SKDV_STS_DEVICE_ST_J9	0x20025480	"J9" CIS Error (Image Sensor Unit)
SKDV_STS_DEVICE_ST_K1	0x20025880	"K1" FPGA Error (Image Sensor Unit)
SKDV_STS_DEVICE_ST_K2	0x20025900	"K2" Power Supply Error (Image Sensor Unit)
SKDV_STS_DEVICE_ST_K3	0x20025980	"K3" Version Error (Image Sensor Unit)
SKDV_STS_DEVICE_ST_F5	0x20023280	"F5"Command error
SKDV_STS_DEVICE_ST_F6	0x20023300	"F6"Parameter Error
SKDV_STS_DEVICE_ST_F7	0x20023380	"F7"Protocol Error
SKDV_STS_DEVICE_ST_G1	0x20023880	"G1" Cover Open
SKDV_STS_DEVICE_ST_G2	0x20023900	"G2" Stacker Unit Cover Open
SKDV_STS_DEVICE_ST_H1	0x20024080	"H1" No Feed
SKDV_STS_DEVICE_ST_H2	0x20024100	"H2" Jam at Paper Feeding Detection Sensor
SKDV_STS_DEVICE_ST_H3	0x20024180	"H3" Jam at Reading Start Detection Sensor
SKDV_STS_DEVICE_ST_H4	0x20024200	"H4" Jam at Main Body Paper Discharge Detection Sensor
SKDV_STS_DEVICE_ST_H5	0x20024280	"H5" Main Body Sheet Interval Error
SKDV_STS_DEVICE_ST_I1	0x20024880	"I1" Jam at Printing Start Detection Sensor
SKDV_STS_DEVICE_ST_I2	0x20024900	"I2" Jam at Main Paper Discharge Sensor

Constant	Value	Description
SKDV_STS_DEVICE_ST_I3	0x20024980	"I3" Jam at Selected (Reject) Paper Discharge Sensor
SKDV_STS_DEVICE_ST_I4	0x20024A00	"I4" Jam at Upper Conveyance Path (Stacker)
SKDV_STS_DEVICE_ST_I5	0x20024A80	"I5" Jam at Lower Conveyance Path (Stacker)
SKDV_STS_DEVICE_ST_I6	0x20024B00	"I6" Stacker Sheet Interval Error
SKDV_STS_DEVICE_ST_P1	0x20028080	"P1" Back Surface Reading Unit Unconnected
SKDV_STS_DEVICE_ST_P2	0x20028100	"P2" Barcode Unit Unconnected
SKDV_STS_DEVICE_ST_P3	0x20028180	"P3" Printer Unit Unconnected
SKDV_STS_DEVICE_ST_P4	0x20028200	"P4" Select Stacker Unit Unconnected
SKDV_STS_DEVICE_ST_P5	0x20028280	"P5" Image Sensor Unit Unconnected
SKDV_STS_DEVICE_ST_Q1	0x20028880	"Q1" Sheet Empty
SKDV_STS_DEVICE_ST_Q2	0x20028900	"Q2" Double Feed Error
SKDV_STS_DEVICE_ST_Q3	0x20028980	"Q3" Left End Skew Error
SKDV_STS_DEVICE_ST_Q4	0x20028A00	"Q4" Mark Skew Error
SKDV_STS_DEVICE_ST_R1	0x20029080	"R1" Hopper Stops
SKDV_STS_DEVICE_ST_R2	0x20029100	"R2" Drowing Out Error
SKDV_STS_DEVICE_ST_R3	0x20029180	"R3" Timeout
SKDV_STS_DEVICE_ST_R4	0x20029200	"R4" Timing Mark Error
SKDV_STS_DEVICE_ST_R5	0x20029280	"R5" Setting Error
SKDV_STS_DEVICE_ST_R6	0x20029300	"R6" Memory Overflow
SKDV_STS_DEVICE_ST_R7	0x20029380	"R7" USB Connection Error
SKDV_STS_DEVICE_ST_R8	0x20029400	"R8"Sheet size setting Error
SKDV_STS_DEVICE_ST_R9	0x20029480	"R9"Sheet layout Error
SKDV_STS_DEVICE_ST_S2	0x20029900	"S2" Black Level Error
SKDV_STS_DEVICE_ST_S3	0x20029980	"S3" Read Sensor Stain Error
SKDV_STS_DEVICE_ST_T1	0x2002A080	"T1" Form Left In Hopper

Constant	Value	Description
SKDV_STS_DEVICE_ST_T2	0x2002A100	"T2" Form Left in Reading Sensor
SKDV_STS_DEVICE_ST_T3	0x2002A180	"T3" Form Left in End of Main Body
SKDV_STS_DEVICE_ST_T4	0x2002A200	"T4" Form Left in Printer Printing Detection Sensor
SKDV_STS_DEVICE_ST_T5	0x2002A280	"T5" Form Left in Main paper Discharge Sensor
SKDV_STS_DEVICE_ST_T6	0x2002A300	"T6" Form Left in Selected (Reject) Paper Discharge Sensor
SKDV_STS_DEVICE_ST_Z1	0x2002D080	"Z1" Density Adjust Skew Error
SKDV_STS_DEVICE_ST_Z2	0x2002D100	"Z2" Density Adjust Sheet Error
SKDV_STS_DEVICE_ST_Z3	0x2002D180	"Z3" Dirt Density Adjust Sheet
SKDV_STS_DEVICE_ST_Z4	0x2002D200	"Z4" Density Adjust Error
SKDV_STS_DEVICE_ST_Z5	0x2002D280	"Z5" RDPS Adjust Error
SKDV_STS_DEVICE_ST_Z6	0x2002D300	"Z6" Skew Sensor Adjust Error
SKDV_STS_DEVICE_ST_Z7	0x2002D380	"Z7" DF sensor Adjust Error

5.4. Structures

5.4.1. SK_DV_MODULE_INFO

Structure	Module infomation	

Syntax	typedef struct tag_SK_DV_MODULE_INFO
	{
	char szModel [SKDV_INFO_MODEL_LEN];
	char szVersion [SKDV_INFO_VER_LEN];
	}SK_DV_MODULE_INFO;	

Member	szModel	Character name strings of Library
	szVersion	Version information of Library
Description	Fold Library information	

5.4.2. SK_DEVICE_INFO

Structure	Basic Device infomation.	

Syntax	typedef struct tag_SK_DEVICE_INFO
	{
	char szGuid [SKDV_GUID_LEN+1];
	char szProduct [SKDV_PRODUCT_STR_MAX+1];
	char szSerialNo [SKDV_SERIAL_LEN+1];
	char szFirmVer [SKDV_FIRM_VER_STR_MAX+1];
	char szFirmSum [SKDV_FIRM_SUM_STR_MAX+1];
	char szHardVer [SKDV_HARD_VER_STR_MAX+1];
	DWORD dwOption;
	}SK_DEVICE_INFO;	

Member	szGuid	Original ID characters
	szProduct	Characters of product’s name
	szSerialNo	Characters of Serial number
	szFirmVer	Characters of firmware version
	szFirmSum	Characters of firmware checksum
	szHardVer	Character of hardware version
	dwOption	Refer to description "5.4.3.SK_DV_SR3500_INFO"
Description	Fold Device infomation. Used as member of Device information. Member except "dwOption" turn to be NULL of . edge	

5.4.3. SK_DV_SR3500_INFO , SK_DV_DEVICE_INFO

Structure	Device infomation	

Syntax	typedef struct tag_SK_DV_SR3500_INFO
	{
	SK_DEVICE_INFO Main;
	SK_DEVICE_INFO Reader[SKDV_FACE_NUM];
	SK_DEVICE_INFO Bcr;
	SK_DEVICE_INFO Printer;
	SK_DEVICE_INFO Stacker;
	SK_DEVICE_INFO Image1;
	SK_DEVICE_INFO Image2;
	}SK_DV_SR3500_INFO , SK_DV_DEVICE_INFO;	

Member	Main	Main body information
	Reader	Reading unit information
	Bcr	Bar code reader unit information.(Option)
	Printer	Printer unit information.(Option)
	Stacker	Select Stacker unit information.(Option)
	Image1	Image reading unit information.1 (only for OMR with image reading function)
	Image2	Image reading unit information.2 (only for OMR with image reading function)

Description

Fold various infomation of OMR.
Folded Items are different depend on connected products and option.

Fold members per products is the following. (✓: foldable / x:non-foldable)
Non-connected optional units are non-foldable.

	Member	SR-3500/6000/6500 SR-1800							SR-3500/6500 HYBRID						
		szGuid	szProduct	szSerialNo	szFirmVer	szFirmSum	szHardVer	dwOption	szGuid	szProduct	szSerialNo	szFirmVer	szFirmSum	szHardVer	dwOption
1	Main	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2	Reader[0]	✓	×	×	✓	✓	×	✓	✓	×	×	✓	✓	×	✓
3	Reader[1]	✓	×	×	✓	✓	×	✓	✓	×	×	✓	✓	×	✓
4	Bcr	✓	×	×	✓	✓	×	✓	×	×	×	×	×	×	×
5	Printer	✓	×	×	✓	✓	×	✓	×	×	×	×	×	×	×
6	Stacker	✓	×	×	✓	✓	×	✓	×	×	×	×	×	×	×
7	Image1	×	×	×	×	×	×	×	✓	×	×	✓	✓	×	✓
8	Image2	×	×	×	×	×	×	×	×	×	×	✓	✓	×	×

	Member	SR-11000							SR-8000 HYBRID						
		szGuid	szProduct	szSerialNo	szFirmVer	szFirmSum	szHardVer	dwOption	szGuid	szProduct	szSerialNo	szFirmVer	szFirmSum	szHardVer	dwOption
1	Main	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2	Reader[0]	✓	×	×	✓	✓	×	✓	✓	╲	╲	✓	✓	╲	✓
3	Reader[1]	✓	×	×	✓	✓	×	✓	✓	╲	╲	✓	✓	╲	✓
4	Bcr	✓	×	×	✓	✓	×	✓	╲	╲	╲	✓	✓	╲	✓
5	Printer	×	×	×	×	×	×	×	╲	╲	╲	╲	╲	╲	╲
6	Stacker	✓	×	×	✓	✓	×	✓	✓	╲	╲	✓	✓	╲	✓
7	Image1	×	×	×	×	×	×	×	╲	╲	╲	╲	╲	╲	╲
8	Image2	×	×	×	×	×	×	×	╲	╲	╲	✓	✓	╲	╲

Description

Each member's dwOption have followings meaning

Constant	Value	Description
SKDV_OPT_UNIT_MASK	0xffff0000	Unit bit mask
SKDV_OPT_UNIT_READER_F	0x80000000	Front reading unit – available
SKDV_OPT_UNIT_READER_B	0x40000000	Back reading unit - available
SKDV_OPT_UNIT_BCR	0x20000000	Bar code reader unit - available
SKDV_OPT_UNIT_PRINTER	0x10000000	Printer unit - available
SKDV_OPT_UNIT_STACKER	0x08000000	Select Stacker unit - available
SKDV_OPT_UNIT_IMAGE_1	0x04000000	Image reading unit 1 - available
SKDV_OPT_UNIT_IMAGE_2	0x02000000	Image reading unit 2 - available

○Reader

Constant	Value	Description
SKDV_OPT_SENSOR_TYPE_MASK	0x0000ff00	Bit mask per sensor variation
SKDV_OPT_SENSOR_TYPE_ERR	0x0000ff00	Error per sensor variation
SKDV_OPT_SENSOR_TYPE_RED_VIS	0x00000000	Visible light
SKDV_OPT_SENSOR_TYPE_INFRARED	0x00000100	Infrared light
SKDV_OPT_SENSOR_PITCH_MASK	0x000000ff	Bit mask of Sensor pitch
SKDV_OPT_SENSOR_PITCH_ERR	0x000000ff	Error of Sensor pitch
SKDV_OPT_SENSOR_PITCH_1P6INCH	0x00000001	1/6 inch
SKDV_OPT_SENSOR_PITCH_02INCH	0x00000002	0.2 inch
SKDV_OPT_SENSOR_PITCH_02INCH_S	0x00000003	0.2inch S
SKDV_OPT_SENSOR_PITCH_025INCH	0x00000004	0.25 inch
SKDV_OPT_SENSOR_PITCH_03INCH	0x00000005	0.3 inch
SKDV_OPT_SENSOR_PITCH_03INCH_F	0x00000006	0.3 inch F
SKDV_OPT_SENSOR_PITCH_6MM	0x00000007	6mm
SKDV_OPT_SENSOR_PITCH_02INCH_K	0x00000008	0.2 inch K
SKDV_OPT_SENSOR_PITCH_02INCH_SP	0x00000009	0.2 inch special
SKDV_OPT_SENSOR_PITCH_02INCH_C	0x0000000a	0.2 inch C

○Bcr

Constant	Value	Description
SKDV_OPT_BARCODE_MASK	0x000000ff	Bit mask of Bar code reader
SKDV_OPT_BARCODE_ERR	0x000000ff	Bar code reader non-connected
SKDV_OPT_BARCODE_V	0x00000001	Vertical feeding
SKDV_OPT_BARCODE_H	0x00000002	Horizontal feeding
SKDV_OPT_BARCODE_IMAGE	0x00000004	Barcode recognition from image.

○Printer

Constant	Value	Description
SKDV_OPT_PRINTER_CARTRIDGE	0x80000000	INK Cartridge available

○Image1

Constant	Value	Description
SKDV_OPT_IMAGE_FACE_MASK	0xf0000000	Bit mask of Face
SKDV_OPT_IMAGE_FACE_FRONT	0x80000000	Front
SKDV_OPT_IMAGE_FACE_BACK	0x40000000	Back
SKDV_OPT_IMAGE_FACE_BOTH	0xc0000000	Front & Back

5.4.4. SK_DV_SR3500_MODE , SK_DV_DEVICE_MODE

Structure	Device mode

Syntax

```
typedef struct tag_SK_DV_SR3500_MODE
{
        int             iFeedMode;
        int             iFeedTime;
       DWORD            dwDisableWarning;
        struct MarkSkew{
                int             iRow;
                int             iLevel;
        };
        int             iPanelOperation;
        struct Buzzer{
                int             iVol;
                int             iTone;
        };
        struct PSaving{
                int             iSleep;
                int             iStandby;
        };
}SK_DV_SR3500_MODE , SK_DV_DEVICE_MODE;
```

Member	iFeedMode	Not use(fix as 0(ZERO))
	iFeedTime	Not use(fix as 0(ZERO))
	dwDisableWarning	Set of Warning Error. Set per bit. Refer setting value at "Description".
	MarkSkew	Set detection column of Mark skew detection. Setting range is 0 to 155. "0" to be invalid.
	iRow	Detection levels of Mark skew. Setting range is 1 to 16.
	iLevel	Switch valid/invalid of panel operation. Refer setting value at "description".
	iPanelOperation	Set the volume and sound quality of Buzzer.
	Buzzer	The volume of Buzzer. Setting range is 0 to 5. "0" to be buzzer OFF.
	iVol	The sound volume of Buzzer. Setting range is 1 to 5.
	iTone	The sound tone of Buzzer Setting range is 1 to 3
	PSaving	Sleep timer. Switching time to turn Normal mode into Sleep mode. Setting range is 0 to 60 minutes. "0" to be invalid.
	iSleep	Sleep timer. Switching time to turn Sleep mode into Stand-by mode.set range is 0-60. 0 is to be invalid.
	iStandby	Stand-by timer. Switching time to turn Sleep mode into Stand-by mode. Set range is 0-60 0 is to be invalid.

Description

Fold various Operation mode of Device.

Use on Getting/Setting Function of Operation mode "SkDv_GetMode/ SkDv_SetMode "

○dwDisableWarning

Constant	Value	Description
SKDV_WARN_AUTO_REJECT	0x00010000	Auto Reject- Valid
SKDV_WARN_HOPPER_EMPTY	0x00020000	Detect Sheet empty- Valid
SKDV_WARN_TM_ERROR	0x00040000	Detect Mark error- Valid
SKDV_WARN_DF_ERROR	0x00080000	Detect Double feed- Valid
SKDV_WARN_LEFT_SKEW	0x00100000	Detect Left skew- Valid

○iPanelOperation

Constant	Value	Description
SKDV_DISABLE	0	Disable LCD panel operation
SKDV_ENABLE	1	Enable LCD panel operation

5.4.5. SK_DV_SR3500MARK_CONF , SK_DV_MARK_CONF

<table>
<tr><th>Structure</th><td colspan="2">Mark condition</td></tr>
<tr><th>Syntax</th><td colspan="2"><pre>typedef struct tag_SK_DV_SR3500MARK_CONF
{
            int   iBackSideReading;
            int   iColumns;
            int   iReadingMethod;
            int   iCtrlMultiple;
            int   iThicknessType;
}SK_DV_SR3500MARK_CONF , SK_DV_MARK_CONF;</pre></td></tr>
<tr><th rowspan="5">Member</th><td>iBackSideReading</td><td>Set Valid / Invalid of Back side reading. Refer to Setting value at "Description".</td></tr>
<tr><td>iColumns</td><td>Column to Read. Setting range is 1 to 48.
In only HYBRID model, 0 can be set. In that case, mark reading is not performed.</td></tr>
<tr><td>iReadingMethod</td><td>Reading method. Refer to Setting value at "Description".</td></tr>
<tr><td>iCtrlMultiple</td><td>Control Multiple. Valid in case Reading method is "Front edge/Rear edge controlling type". Setting range is as following.
"Front edge controlling type" : 1 to 9
"Rear edge controlling type" : 2 to 9</td></tr>
<tr><td>iThicknessType</td><td>Thickness of Sheet. Judge standard when to detect Double feed error.</td></tr>
<tr><th>Description</th><td colspan="2">Fold Mark Reading condition.

○iBackSideReading
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKDV_DISABLE</td><td>0</td><td>Disable – Back side</td></tr>
<tr><td>SKDV_ENABLE</td><td>1</td><td>Enable – Back side</td></tr>
</table>
○iReadingMethod
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKDV_READ_FRONT_EDGE</td><td>1</td><td>Front edge controlling type</td></tr>
<tr><td>SKDV_READ_REAR_EDGE</td><td>2</td><td>Rear edge controlling type</td></tr>
<tr><td>SKDV_READ_DIRECT</td><td>3</td><td>Read Direct type</td></tr>
<tr><td>SKDV_READ_FACOM</td><td>4</td><td>Read FACOM type</td></tr>
<tr><td>SKDV_READ_BETWEEN_MARK_NS</td><td>5</td><td>Read between Mark type
(Front edge blank reading – Unavailable)</td></tr>
<tr><td>SKDV_READ_BETWEEN_MARK</td><td>6</td><td>Read between Mark type
(Front edge blank reading – Available)</td></tr>
</table>
○iThicknessType
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKDV_THICKNESS_AUTO_DETECT</td><td>0</td><td>Automatic detection</td></tr>
<tr><td>SKDV_THICKNESS_55_KG</td><td>1</td><td>64g/m^2 (55kg)</td></tr>
<tr><td>SKDV_THICKNESS_72_KG</td><td>2</td><td>84g/m^2 (72kg)</td></tr>
<tr><td>SKDV_THICKNESS_90_KG</td><td>3</td><td>105g/m^2 (90kg)</td></tr>
<tr><td>SKDV_THICKNESS_110_KG</td><td>4</td><td>128g/m^2 (110kg)</td></tr>
<tr><td>SKDV_THICKNESS_135_KG</td><td>5</td><td>157g/m^2 (135kg)</td></tr>
</table></td></tr>
</table>

5.4.6. SK_DV_SR3500IMG_CONF , SK_DV_IMAGE_CONF

Structure	Image condition

Syntax

```
typedef struct tag_SK_DV_SR3500IMG_CONF
{
        int             iScanningSide;
        int             iScanCtrlType;
        int             iFormLengthMin;
        int             iFormLengthMax;
        int             iLightColor[SKDV_FACE_NUM];
        int             iResoType;
        int             iResoTypeY;
}SK_DV_SR3500IMG_CONF , SK_DV_IMAGE_CONF;
```

Member	iScanningSide	Set reading side. Refer to Setting value at "Description".
	iScanCtrlType	Non-usage
	iFormLengthMin	Non-usage
	iFormLengthMax	Non-usage
	iLightColor[0]	Set Front side light color. Refer to Setting value at"Description".
	iLightColor[1]	set Back side light color. Setting value is as same as iLightColor[0].
	iResoType	Set Optical resolution. Not able to set up per Front or Back side.
	iResoTypeY	Non-usage

Description

Fold Image condition.

SR-3500 HYBRID 系で使用可能。SR-8000 HYBRID 系でも制約はあるが使用可能。

For SR-3500HYBRID series. Available with condition for SR-8000HYBRID.

○iScanningSide

Constant	Value	Description
SKDV_SIDE_NONE	0	Non-reading image
SKDV_SIDE_BOTH	1	Double side
SKDV_SIDE_FRONT	2	Front side only
SKDV_SIDE_BACK	3	Back side only

○iLightColor

Light color is not able to set Front or Back side sepatrately. In case of setting either side "Color", it is needed to set another sides "Color" as same.

Constant	Value	Description
SKDV_LIGHT_COLOR	0	Color (3 colors)
SKDV_LIGHT_GRAYSCALE	1	Gray scale (single color)
SKDV_LIGHT_RED	2	Red (single color)
SKDV_LIGHT_GREEN	3	Green (single color)
SKDV_LIGHT_BLUE	4	Blue (single color)

○iResoType

Constant	Value	Description
SKDV_RESO_TYPE_300DPI	1	300dpi
SKDV_RESO_TYPE_200DPI	2	200dpi
SKDV_RESO_TYPE_150DPI	3	150dpi
SKDV_RESO_TYPE_100DPI	4	100dpi

5.4.7. SK_DV_IMAGE_CONF_EX

Structure	Image extra condition	

Syntax

```
typedef struct tag_SK_DV_IMAGE_CONF_EX
{
        int                 iSize;
        struct tagFace{
                int                 iEnable;
                int                 iColorType;
                int                 iEncoder;
                int                 iParam;
                int                 iRotate;
                int                 iResoMode;
                int                 iResolution;
        }Face[SKDV_FACE_NUM];
}SK_DV_IMAGE_CONF_EX;
```

Member	iSize	Set size of structure
	Face[SKDV_FACE_NUM]	Array for front and back side
	iEnable	Enable(SKDV_ENABLE) or Disable(SKDV_DISABLE) image reading
	iColorType	Set reading color. Refer to Setting value at "Description".
	iEncoder	Non-usage. BMP in one page reading. JPEG in continuous reading.
	iParam	Specify quality in image compression
	iRotate	Set image rotation
	iResoMode	Set resolution of image reading
	iResolution	Set resolution of image sending (100-600dpi)

Description

Fold image condition

Only for SR-8000HYBRID

○iEnable

Constant	Value	Description
SKDV_DISABLE	0	Disable image reading
SKDV_ENABLE	1	Enable image reading

○iColorType

Constant	Value	Description
SKDV_LIGHT_COLOR	0	Color (3-colors)
SKDV_LIGHT_GRAYSCALE	1	Grayscale (monochrome)
SKDV_LIGHT_RED	2	Red (plain color)
SKDV_LIGHT_GREEN	3	Green (plain color)
SKDV_LIGHT_BLUE	4	Blue (plain color)

○iEncoder

Non-usage. Only BMP in one page reading, and only JPEG in continuous reading

Constant	Value	Description
SKIM_ENCODE_BMP	0	BMP data (no-compressed)
SKIM_ENCODE_JPEG	2	JPEG data

○iResoMode

Reading speed and barcode recognition varies depending on the setting

Constant	Value	Description
SKDV_RESO_MODE_600DPI	0	600dpi
SKDV_RESO_MODE_300DPI	1	300dpi

5.4.8. SK_DV_SR3500_STATUS_CHAR , SK_DV_DEVICE_STATUS_CHAR

Structure	Device status code	

Syntax	typedef struct tag_SK_DV_SR3500_STATUS_CHAR
	{
	char ST1[SKDV_ST_LEN];
	char ST2[SKDV_ST_LEN];
	char DIF1;
	char DIF2;
	}SK_DV_SR3500_STATUS_CHAR , SK_DV_DEVICE_STATUS_CHAR;	

Member	ST1	Status information. Front side
	ST2	Status information. Back side
	DIF1	Error description information. Front side
	DIF2	Error description information. Back side
Description	Fold Status information of Device. Refer to "Command reference" in description.	

5.4.9. SK_DV_MARK_INFO

Structure	Mark data infomation.	

Syntax	typedef struct tag_SK_DV_MARK_INFO
	{
	int iType;
	int iRows;
	int iColumns;
	}SK_DV_MARK_INFO;	

Member	iType	Non-use
	iRows	Number of rows read
	iColumns	Number of columns read
Description	Fold read Mark data information.	

5.4.10. SK_DV_IMAGE_DATA_CONF

Structure	Image data condition

Syntax

```
typedef struct tag_SK_DV_IMAGE_DATA_CONF
{
            int                     iFace;
            RECT                    rectCutout;
            int                     iColorType;
            int                     iRotate;
            int                     iResolution;
            DWORD                   dwOption;
}SK_DV_IMAGE_DATA_CONF;
```

Member

iFace	Set front or back side

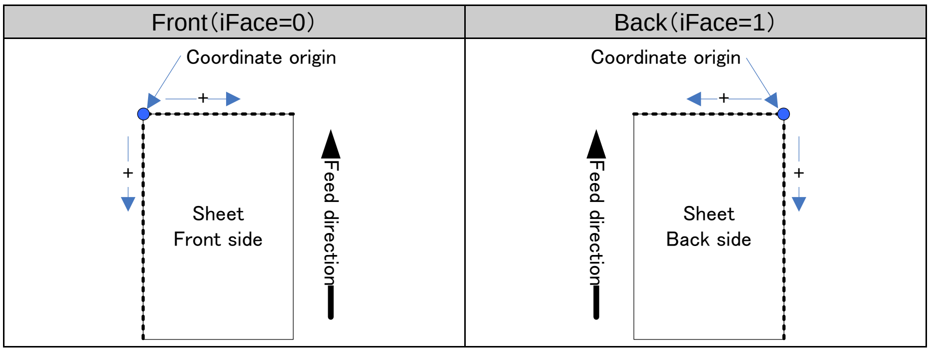
rectCutout	Set the Cutting position by degrees of 0.1mm.
	Coordinate System can be referred at "Description"
	Refer to RECT at "MFC library reference.
	typedef struct tagRECT {
	LONG left;
	LONG top;
	LONG right;
	LONG bottom;
	} RECT;

iColorType	Chosen from Black and White, Gray scale (8bit), Color (24bit). Refer to setting value at "Description"
iRotate	Rotation of Image. Chosen per 90 degrees.
iResolution	Optical resolution. Setting range is 50 to 300 dpi. Need to set the resolution of actual reading resolution. In case of setting the resolution higher than actual reading resolution, Image quality will be going down.
dwOption	Non-usage

Description

Fold Getting condition of Image data

○rectCutout



○iFace

Constant	Value	Description
SKDV_FRONT	0	Front side
SKDV_BACK	1	Back side

○iColorType

Constant	Value	Description
SKIM_IMAGE_WB	0	Black and White
SKIM_IMAGE_GRAY	1	Gray scale (8bit)
SKIM_IMAGE_COLOR	2	Color (24bit)

○iRotate

Constant	Value	Description
SKIM_IMAGE_ROTATE_0	0	Non-Rotation
SKIM_IMAGE_ROTATE_90	1	Clockwise – 90 degrees Rotation
SKIM_IMAGE_ROTATE_180	2	Clockwise –180 degrees Rotation
SKIM_IMAGE_ROTATE_270	3	Clockwise –270 degrees Rotation

5.4.11. SK_DV_IMAGE_FILE_CONF

<table>
<tr><th>Structure</th><th colspan="2">Image File Condition</th></tr>
<tr><td>Syntax</td><td colspan="2"><pre>
typedef struct tag_SK_DV_IMAGE_FILE_CONF
{
        int   iEncoder;
        int   iParam;
}SK_DV_IMAGE_FILE_CONF;
</pre></td></tr>
<tr><td rowspan="2">Member</td><td>iEncoder</td><td>File form. Refer to setting value at "Description".</td></tr>
<tr><td>iParam</td><td>File form parameter. Able to be appointed by file form.
In case of Non-usage file form, "0" is to be appointed.
Jpeg form: Set Image quality. Setting range is 1 to 100.</td></tr>
<tr><td>Description</td><td colspan="2">
<p>Set file format when saving readingned Image data into file.</p>
<p>○iEncoder</p>
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKIM_ENCODE_BMP</td><td>0</td><td>Bit map format</td></tr>
<tr><td>SKIM_ENCODE_GIF</td><td>1</td><td>GIF format</td></tr>
<tr><td>SKIM_ENCODE_JPEG</td><td>2</td><td>Jpeg format</td></tr>
<tr><td>SKIM_ENCODE_PNG</td><td>3</td><td>PNG format</td></tr>
<tr><td>SKIM_ENCODE_TIFF</td><td>4</td><td>Tiff format</td></tr>
</table>
<p>○[iParam] (iEncoder:SKIM_ENCODE_JPEG)
Can be set 1 – 100. 1 = Low resolution, 100 = High Resolution</p>
<p>○iParam （iEncoder : SKIM_ENCODE_TIFF）</p>
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKIM_ENCODE_TIFF_PRM_LZW</td><td>2</td><td>LZW compressed</td></tr>
<tr><td>SKIM_ENCODE_TIFF_PRM_NONE</td><td>6</td><td>Non Compressed</td></tr>
</table>
</td></tr>
</table>

5.4.12. SK_DV_IMG_ELGT_DETECT_CONF

<table>
<tr><td>Structure</td><td colspan="2">Image elongation detection setting</td></tr>
<tr><td>Syntax</td><td colspan="2">typedef struct tag_SK_DV_IMG_ELGT_DETECT_CONF
{
    bool   blEnabled;
    double   SheetLen;
    int   Threshold;
}SK_DV_IMG_ELGT_DETECT_CONF;</td></tr>
<tr><td>Member</td><td>blEnabled</td><td>Validity of image elongation detection.</td></tr>
<tr><td></td><td>SheetLen</td><td>The length of the sheet to be read.</td></tr>
<tr><td></td><td>Threshold</td><td>Threshold for detecting image elongation</td></tr>
<tr><td>Description</td><td colspan="2">
It sets the information necessary for detecting the image elongation.

Since "5.12. ImageElongationDetection" also has explanations and notes, so be sure to check it.

○blEnabled
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKIM_ENCODE_DETECT_ENABLED</td><td>true</td><td>Image elongation detection is enabled.</td></tr>
<tr><td>SKIM_ENCODE_DISABLE_DETECT</td><td>false</td><td>Image elongation detection is disabled.</td></tr>
</table>

○SheetLen
In the range of 110 to 355.6 mm, set the length of the sheet to be used in units of 0.1mm.

○Threshold
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKIM_ENCODE_IMG_ELGT_THR_10</td><td>100</td><td>Set the image elongation detection threshold to 10 mm.</td></tr>
<tr><td>SKIM_ENCODE_IMG_ELGT_THR_20</td><td>200</td><td>Set the image elongation detection threshold to 20 mm.</td></tr>
<tr><td>SKIM_ENCODE_IMG_ELGT_THR_30</td><td>300</td><td>Set the image elongation detection threshold to 30 mm.</td></tr>
</table>
</td></tr>
</table>

5.4.13. SK_LAYOUT_ID_PRM

Structure	Set Layout ID	

Syntax	typedef struct tag_SK_LAYOUT_ID_PRM
	{
	UCHAR ucIdData[SKDV_LAYOUT_ID_PTN_LEN];
	int iTmCount[SKDV_FACE_NUM];
	int iBarcodeCount;
	}SK_LAYOUT_ID_PRM;	

Member	ucIdData	Set mark pattern of ID window
	iTmCount	Set the number of timing marks both side.
	iBarcodeCount	Barcode not use=Fix as “0” Barcode use=”1” or ” 2”(QTY of BC), or”99” (to get the all barcodes)
Description	Store the setting of layout ID for setting of ID data setting.	

5.4.14. SK_WINDOW_PRM

Structure	Window setting		

Syntax	typedef struct tag_SK_WINDOW_PRM
	{
	int iFace;
	struct
	{
	int iStart;
	int iNumber;
	int iStep;
	}Col;
	struct
	{
	int iStart;
	int iNumber;
	int iStep;
	}Row;
	int iDirection;
	int iPartition;
	struct
	{
	int iSensitivity;
	int iDifference;
	}Level;
	}SK_WINDOW_PRM;		

Member	iFace		Set front / back side
	Col		Column
		iStart	Start column
		iNumber	Number of window
		iStep	Number of step
	Row		Row
		iStart	Start row
		iNumber	Number of window
		iStep	Number of step
	iDirection		direction
	iPartition		Partition This is not use in“SkDv_ReqSetLayoutManage” (Ignores to set)
	Level		Density
		iSensitivity	Detection sensitivity level
		iDifference	Differences the sensitivity level

Description

Window setting

○iDirection

Constant	Value	Description
SKDV_WIN_DIR_TL_DOWN	0	From the upper left to the bottom
SKDV_WIN_DIR_TR_DOWN	1	From the upper right to the bottom
SKDV_WIN_DIR_BL_UP	2	From the lower left to a top
SKDV_WIN_DIR_BR_UP	3	From the lower right to a top
SKDV_WIN_DIR_TL_RIGHT	4	From the upper left to right
SKDV_WIN_DIR_TR_LEFT	5	From the upper right to the left
SKDV_WIN_DIR_BL_RIGHT	6	From the lower left to right
SKDV_WIN_DIR_BR_LEFT	7	From the lower right to the left

○iPartition

Constant	Value	Description
SKDV_WIN_PART_OFF	0	Without partition
SKDV_WIN_PART_ON_START0	1	With Partition (0 to start)
SKDV_WIN_PART_ON_START1	2	With Partition (1 to start) When operation processing of a range checking etc. is performed, a beginnings can be specified in a start and the start from 1 from 0.

5.4.15. SK_WINDOW_CHECK

Structure	Window check

Syntax

```
typedef struct          tag_SK_WINDOW_CHECK
{
            int         iOption;
            struct
            {
                        int                 iMin;
                        int                 iMax;
            }MarkCount;
}SK_WINDOW_CHECK;
```

Member

iOption	Nor Mark permit
MarkCount	Check the number of marks
iMin	Minimum number of marks
iMax	Maximum number of of marks

Description

Structure that contains the check the number of marks.

○iOption

Constant	Value	Description
SKDV_WIN_CHK_NOMARK_ERR_ENABLE	0	No mark as an error
SKDV_WIN_CHK_NOMARK_ERR_DISABLE	1	No mark is not an error

5.4.16. SK_LAYOUT_OPTION

Structure	Set Layout option

Syntax

```
typedef struct          tag_SK_LAYOUT_OPTION
{
            int                     iType;
            union
            {
                        UCHAR   ucData[SKDV_LAYOUT_OPT_LEN];
                        struct
                        {
                                    DWORD               dwMin;
                                    DWORD               dwMax;
                        }Num;
            };
}SK_LAYOUT_OPTION;
```

Member

iType		Optional feature
ucData		Optional features parameter
Num		Range
	dwMin	Minimum
	dwMax	Maximum

Description

Structure that contains the layout option.

○iType
SKDV_LAYOUT_OPT_NONE/ SKDV_LAYOUT_OPT_ORDER_ID

In case setup the window, SKDV_LAYOUT_OPT_NONE/ SKDV_LAYOUT_OPT_MASK/ SKDV_LAYOUT_OPT_FIXED_COMP/ SKDV_LAYOUT_OPT_CHECK_DIGIT/ SKDV_LAYOUT_OPT_ASCENDING_ORDER/ SKDV_LAYOUT_OPT_RANGE_CHECK/ SKDV_LAYOUT_OPT_MASK_PART are available

Constant	Value	Description
SKDV_LAYOUT_OPT_NONE	0x00	Invalid (default)
SKDV_LAYOUT_OPT_ORDER_ID	0x01	The order check (at the time set the ID) of ID.
SKDV_LAYOUT_OPT_MASK	0x01	Mask (at the time set window) about a mark.
SKDV_LAYOUT_OPT_FIXED_COMP	0x02	Fixed mark
SKDV_LAYOUT_OPT_CHECK_DIGIT	0x03	Checkdigits
SKDV_LAYOUT_OPT_ASCENDING_ORDER	0x04	Range checking (ascending orders)
SKDV_LAYOUT_OPT_DESCENDING_ORDER	0x05	Range checking (descending order)
SKDV_LAYOUT_OPT_RANGE_CHECK	0x06	Range checking (not order)
SKDV_LAYOUT_OPT_MASK_PART	0x07	Mask setting(common to partition)

○ucData

Constant	Value	Description
SKDV_LAYOUT_OPT_LEN	16	Option data length [byte]

5.4.17. SK_LAYOUT_MANAGE_CONF

<table>
<tr><td>Structure</td><td colspan="2">ID management</td></tr>
<tr><td>Syntax</td><td colspan="2">typedef struct tag_SK_LAYOUT_MANAGE_CONF
{
BOOL blEnableId;
SK_WINDOW_PRM IdWindowPrm;
DWORD dwNgAction;
SK_LAYOUT_OPTION LayoutOpt;
}SK_LAYOUT_MANAGE_CONF;</td></tr>
<tr><td rowspan="4">Member</td><td>blEnableId</td><td>Layout ID switch</td></tr>
<tr><td>IdWindowPrm</td><td>ID window areas</td></tr>
<tr><td>dwNgAction</td><td>Judge NG operations</td></tr>
<tr><td>LayoutOpt</td><td>Option specified</td></tr>
<tr><td>Description</td><td colspan="2">Structure that contains ID management information.

○dwNgAction
Judge NG operations. Be used in combination.
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKDV_ACTION_SELECT</td><td>0x00000001</td><td>Paper ejection to select stacker.</td></tr>
<tr><td>SKDV_ACTION_STOP</td><td>0x00000002</td><td>Stop reading.</td></tr>
<tr><td>SKDV_ACTION_NOPRINT</td><td>0x00000004</td><td>Do not print.</td></tr>
</table>

○LayoutOpt.iType
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKDV_LAYOUT_OPT_NONE</td><td>0</td><td>Invalid (default)</td></tr>
<tr><td>SKDV_LAYOUT_OPT_ORDER_ID</td><td>1</td><td>ID order checks</td></tr>
</table>
</td></tr>
</table>

5.4.18. SK_LAYOUT_ID_CONF

<table>
<tr><td>Structure</td><td colspan="2">ID data</td></tr>
<tr><td>Syntax</td><td colspan="2">typedef struct tag_SK_LAYOUT_ID_CONF
{
SK_LAYOUT_ID_PRM LayoutIdPrm;
int iReject;
SK_LAYOUT_OPTION LayoutOpt;
}SK_LAYOUT_ID_CONF;</td></tr>
<tr><td rowspan="3">Member</td><td>LayoutIdPrm</td><td>Set the ID</td></tr>
<tr><td>iReject</td><td>Force select eject(Reject)</td></tr>
<tr><td>LayoutOpt</td><td>Options setting</td></tr>
<tr><td>Description</td><td colspan="2">Structure that contains ID data.

○iBackSideReading
<table>
<tr><th>Constant</th><th>Value</th><th>Description</th></tr>
<tr><td>SKDV_DISABLE</td><td>0</td><td>Not force reject.</td></tr>
<tr><td>SKDV_ENABLE</td><td>1</td><td>Force reject.</td></tr>
</table>
</td></tr>
</table>

5.4.19. SK_WINDOW_AREA_CONF

Structure	Window area	

Syntax	typedef struct tag_SK_WINDOW_AREA_CONF
	{
	SK_WINDOW_PRM WindowPrm;
	SK_WINDOW_CHECK WindowCheck;
	DWORD dwNgAction;
	SK_LAYOUT_OPTION LayoutOpt;
	}SK_WINDOW_AREA_CONF;	

Member	WindowPrm	Window areas
	WindowCheck	Window check
	dwNgAction	Operation while NG operation
	LayoutOpt	Options setting
Description	Store window areas information.	

5.4.20. SK_CONT_FEED_PRN_OPT

Structure	Printed options continuous reading	

Syntax	typedef struct tag_SK_CONT_FEED_PRN_OPT
	{
	int iDigits;
	int iStartNumber;
	}SK_CONT_FEED_PRN_OPT;	

Member	iDigits	Number of digits
	iStartNumber	Set the start number of Sequence
Description	Store printed options on continuous reading	

5.4.21. SK_DV_WINDOW_RESULT

Structure	Window read result	

Syntax	typedef struct tag_SK_DV_WINDOW_RESULT
	{
	int iResult;
	int iId;
	int iCount;
	}SK_DV_WINDOW_RESULT, *PSK_DV_WINDOW_RESULT;	

Member	iResult	Read result
	iId	ID recognition results
	iCount	The number of window
Description	Structure which stores the reading result information each window. ○iId	

Constant	Value	Description
SKDV_RESULT_ID_MIN	0	Minimum layout ID
SKDV_RESULT_ID_MAX	15	Maximum layout ID
SKDV_RESULT_ID_NUM	16	The number of layout ID
SKDV_RESULT_ID_NONE	0	ID when no ID is set
SKDV_RESULT_ID_UNKNOWN	99	ID mismatch

5.4.22. SK_DV_BARCODE_RESULT

Structure	Bar codes reading result structure	

Syntax	typedef struct tag_SK_DV_BARCODE_RESULT
	{
	int iResult;
	int iCount;
	}SK_DV_BARCODE_RESULT, *PSK_DV_BARCODE_RESULT;	

Member	iResult	Reading result
	iCount	The number of barcodes
Description	Stores the reading result of barcode.	

5.4.23. SK_SHEET_RESULT

Structure	Sheets read result

Syntax

```
typedef struct          tag_SK_SHEET_RESULT
{
            SkDvStatus            DvStatus;

            int                   iSheetCount;
            int                   iTmCount[SKDV_FACE_NUM];
            int                   iBarcodeCount;
            int                   iId;
            int                   iPrinted;
            int                   iStackedId;
            DWORD                 dwResult;
            int                   iFinish;
            int                   iIdResult;
            struct
            {
                        int                   iCount;

                        int                   iIndex;
                        int                   iPartIndex;
                        int                   iReason;
            }NgWindow;
}SK_SHEET_RESULT;
```

Member

DvStatus		Status of reading
iSheetCount		Total counts of form
iTmCount		The number of timing marks
iBarcodeCount		The number of barcode
iId		ID recognition results
iPrinted		Printing result
iStackedId		Paper Destination exit bin
dwResult		Notification data of a transport result
iFinish		Notice data of termination
iIdResult		ID recognition results
NgWindow		NG window detail
	iCount	The number of TOTAL NG Window
	iIndex	The first NG window number (0-)
	iPartIndex	The first NG partition number (0-)
	iReason	The first NG reason

Description

The structure which stores sheets read result.

○iID

Constant	Value	Description
SKDV_RESULT_ID_MIN	0	Minimum layout ID
SKDV_RESULT_ID_MAX	15	Maximum layout ID
SKDV_RESULT_ID_NUM	16	The number of layout ID
SKDV_RESULT_ID_NONE	0	ID at the time of ID setting nothing
SKDV_RESULT_ID_UNKNOWN	99	ID mismatch

○iPrinted

Constant	Value	Description
SKDV_RESULT_PRN_NONE	0	un-printing.
SKDV_RESULT_PRN_PRINTED	1	Printed

○iStackedId

Constant	Value	Description
SKDV_RESULT_STK_NONE	0	Not output the form
SKDV_RESULT_STK_MAIN	1	Output for main tray
SKDV_RESULT_STK_SELECTION	2	Output for select tray

SKDV_RESULT_STK_CNT	3	Total count

○dwResult

Constant	Value	Description
SKDV_RESULT_STS_OK	0x00000000	Normal
SKDV_RESULT_STS_ERROR_EJECT	0x00000004	Errors detect and eject form
SKDV_RESULT_STS_ERROR_STOP	0x00000002	Error detect and stop form
SKDV_RESULT_STS_WINDOW_NG	0x00000001	NG judge window
SKDV_RESULT_STS_ID_NG	0x00000008	ID judgment is NG
-	Except above	Other error

○NgWindow.iReason

Constant	Value	Description
SKDV_RESULT_REASON_OK	0	Normal
SKDV_RESULT_REASON_ MARK_COUNT_LOWER	1	There are few marks.
SKDV_RESULT_REASON_ MARK_COUNT_UPPER	2	There are many marks.
SKDV_RESULT_REASON_ MARK_DISAGREE	3	Fixed data mismatch
SKDV_RESULT_REASON_ CHECK_DIGIT_NG	4	Checkdigit NG
SKDV_RESULT_REASON_ ASCENDING_ORDER_NG	5	Range check(ascending)
SKDV_RESULT_REASON_ DESCENDING_ORDER_NG	6	Range check (descending)
SKDV_RESULT_REASON_ RANGE_CHECK_NG	7	Range check(not direction)
-	Except above	Other reason

○iFinish

Constant	Value	Description
SKDV_RESULT_FINISH_CONT	0	Continue
SKDV_RESULT_FINISH_DONE	1	Finish

○iIdResult

Constant	Value	Description
SKDV_RESULT_ID_OK	0	Normal
SKDV_RESULT_ID_TM_CNT_DIFF_F	1	The number of timing mark incorrect(front)
SKDV_RESULT_ID_TM_CNT_DIFF_B	2	The number of timing mark incorrect(back)
SKDV_RESULT_ID_TM_CNT_DIFF_FB	3	The number of timing mark incorrect(both)
SKDV_RESULT_ID_ID_NONE	4	All layout ID incorrect
SKDV_RESULT_ID_BC_CNT_DIFF	5	The number of Barcode incorrect
SKDV_RESULT_ID_ID_ORDER_NG	6	Order of Layout ID Incorrect
-	Except above	Other reason

5.4.24. SK_CONT_FEED_RESULT

Structure	Result of Continuous reading	

Syntax	
	```
	typedef struct		tag_SK_CONT_FEED_RESULT
	{
			SkDvStatus		DvStatus;

			int			iResult;
			struct
			{
				int			iRead
				int			iPrint;
				int			iEject;
			}Remain;
	}SK_CONT_FEED_RESULT;
	```

Member	DvStatus	Final status of continuous reading
	iResult	Notify finish data of continuous reading
	Remain	The number of forms not finished
	iRead	The number fo forms not reading
	iPrint	The number of forms not printing
	iEject	The number of forms not feed to tray
Description	Stores the result of continuous reading. ○iResult	

Constant	Value	Description
SKDV_CFR_NORMAL	0	Normal finish
SKDV_CFR_ERROR	1	Error finish
SKDV_CFR_WINDOW_NG	2	Finish by window judgement
SKDV_CFR_CANCEL	3	Finish by Cancel

5.4.25. SK_CONT_FEED_COUNT

Structure	Feed count of continuous reading	

Syntax	
	```
	typedef struct		tag_SK_CONT_FEED_COUNT
	{
			int			iFeed;
			int			iNormal;
			int			iPrint;
			int			iEject[SKDV_RESULT_STK_CNT];
			int			iUntreated;
	}SK_CONT_FEED_COUNT;
	```

Member	iFeed	Total feed count
	iNormal	The number of forms read normally
	iPrint	The number of forms print normally
	iEject	The number of forms output normally
	iUntreated	The number of forms not output
Description	Stores the number of continuous reading.	

5.4.26. SK_DV_OPT_PRN_CONF

Structure	Print setting

Syntax

```
typedef struct          tag_SK_DV_OPT_PRN_CONF
{
              int                    iEnable;
              int                    iStartPos;
              int                    iOrientation;
              int                    iFontSize;
              int                    iFontSpace;
              int                    iPrintMode;
}SK_DV_OPT_PRN_CONF;
```

Member		
Member	iEnable	Printer setting Valid or Invalid
	iStartPos	Print positions(mm)
	iOrientation	Print direction(0 degree’ or 180 degrees’)
	iFontSize	Printed font size
	iFontSpace	Printed characters spacings
	iPrintMode	Printing method

Description

Stores printer settings

○iEnable

Constant	Value	Description
SKDV_DISABLE	0	Invalid printing
SKDV_ENABLE	1	Valid printing

○iOrientation

Constant	Value	Description
SKDV_PRINT_ORIENT_0	1	Normal
SKDV_PRINT_ORIENT_180	2	180 degrees rotation

○iPrintMode

Constant	Value	Description
SKDV_PRINT_MODE_AFTER_FEED	1	Print after feed
SKDV_PRINT_MODE_FEED_AND_PRINT	2	Feed and print

5.4.27. SK_DV_OPT_BCR_CONF

Structure	Barcode reading condition

Syntax

```
typedef struct          tag_SK_DV_OPT_BCR_CONF
{
            int         iEnable;
            int         iReadingArea[SKDV_BCR_READING_AREA_MAX];
            DWORD       dwEnableBcType;

            struct tagCheckDigit{
                        int                     iCode39;
                        int                     iItf;
                        int                     iNw7;
                        int                     iReserved1;
                        int                     iReserved2;
                        int                     iReserved3;
            }CheckDigit;

            struct tagOption{
                        int                     iUpcA;
                        int                     iUpcE;
                        int                     iReserved1;
                        int                     iReserved2;
                        int                     iReserved3;
                        int                     iReserved4;
            }Option;
}SK_DV_OPT_BCR_CONF;
```

Member	iEnable		Barcode reading Invalid or Valid
	iReadingArea		Set the area of Barcode reading
	dwEnableBcType		Set the kind of barcode to read
	CheckDigit		Checkdigit
		iCode39	CODE39 checkdigit
		iItf	ITF checkdigit
		iNw7	NW7 checkdigit
		iReserved1-3	Not use
	Option		Option
		iUpcA	UPC-A output digit
		iUpcE	UPC-Eaddon system code
		iReserved1-4	Not use

Description

Barcode read condition

○iEnable

Constant	Value	Description
SKDV_DISABLE	0	Invalid barcode reading
SKDV_ENABLE	1	Valid barcode reading

○iReadingArea

・row1:"SKDV_BCR_READING_AREA_ENTIRE"to read all area.

・row1:SKDV_Disable (mm) and
row2:"SKDV_BCR_READING_AREA_ENTIRE"to read except disable area

Constant	Value	Description
SKDV_BCR_READING_AREA_ENTIRE	0	Read all area

○dwEnableBcType

Constant	Value	Description
SKDV_BCR_CODE39	0x00000001	CODE-39
SKDV_BCR_ITF	0x00000002	Interleaved 2 of 5 (ITF)
SKDV_BCR_NW7	0x00000004	NW-7
SKDV_BCR_JAN_EAN_UPC	0x00000008	JAN,EAN,UPC
SKDV_BCR_CODE128	0x00000010	Code-128
SKDV_BCR_INDUSTRIAL2OF5	0x00000020	Industrial 2 of 5
SKDV_BCR_COOP2OF5	0x00000040	COOP 2 of 5
SKDV_BCR_CODE93	0x00000080	CODE-93

○CheckDigit

Some kinds of barcode are available to use Checkdigit

・iCode39 / iItf

Constant	Value	Description
SKDV_BCR_CD_NONE	0	Checkdigit OFF
SKDV_BCR_CD_ENABLE	1	Checkdigit ON

・iNw7

Constant	Value	Description
SKDV_BCR_CD_NONE	0	Not check
SKDV_BCR_CD_NW7_M16	1	Modulus16
SKDV_BCR_CD_NW7_M11	2	Modulus11
SKDV_BCR_CD_NW7_M10W2	3	Modulus 10／2
SKDV_BCR_CD_NW7_M10W3	4	Modulus 10／3
SKDV_BCR_CD_NW7_7DR	5	7DR
SKDV_BCR_CD_NW7_WM11	6	Weight modulus11
SKDV_BCR_CD_NW7_RUNES	7	Runes check

○Option

Option is available depends on the kind of barcode

・iUpcA : UPC-A output digit

Constant	Value	Description
SKDV_BCR_CD_UPC_A_DIGIT_12	0	Output 12digits
SKDV_BCR_CD_UPC_A_DIGIT_13	1	Output 13digits

・iUpcE : UPC-E system code

Constant	Value	Description
SKDV_BCR_CD_UPC_E_NO_CODE	0	Not addon system code
SKDV_BCR_CD_UPC_E_ADD_CODE	1	Add-on system code

5.4.28. SK_DV_OPT_BCR_AREA

Structure	Set barcode reading area	

Syntax	typedef struct tag_BCR_AREA
	{
	RECT rectFrect;
	DWORD dwEnableBcType;
	int iCheckDigit;
	int iOption;
	}SK_DV_OPT_BCR_AREA;	

Member	rectFrect	Barcode reading area
	dwEnableBcType	Set type of barcode. One choice only (no multiple choices)
	iCheckDigit	Set check digit
	iOption	Set option

```
typedef struct tag_BCR_AREA
{
            RECT      rectFrect;
            DWORD     dwEnableBcType;
            int       iCheckDigit;
            int       iOption;
}SK_DV_OPT_BCR_AREA;
```

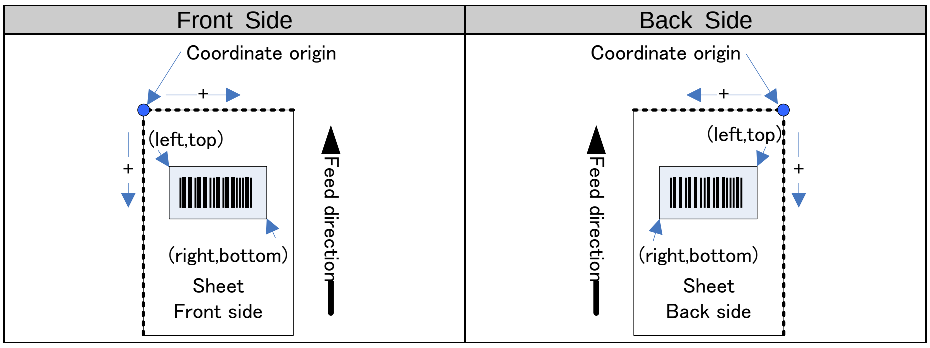
Description

○rectFrect

Set recognizing area (set in 0.1mm unit)

RECT refers MFC library reference

```
typedef struct tagRECT {
    LONG left;
    LONG top;
    LONG right;
    LONG bottom;
} RECT;
```



○dwEnableBcType

Set barcode type by bit unit

定数名 Constant	値 value	意味 Description
SKDV_BCR_CODE39	0x00000001	CODE-39
SKDV_BCR_ITF	0x00000002	Interleaved 2 of 5（ITF）
SKDV_BCR_NW7	0x00000004	NW-7
SKDV_BCR_JAN_EAN_UPC	0x00000008	Non usage
SKDV_BCR_CODE128	0x00000010	Code-128
SKDV_BCR_INDUSTRIAL2OF5	0x00000020	Non usage
SKDV_BCR_COOP2OF5	0x00000040	COOP 2 of 5
SKDV_BCR_CODE93	0x00000080	Non usage
SKDV_BCR_JAN_EAN_8	0x00000100	JAN, EAN (8 digits)
SKDV_BCR_JAN_EAN_13	0x00000200	JAN, EAN (13 digits)
SKDV_BCR_UPC_A	0x00000400	UPC-A
SKDV_BCR_UPC_E	0x00000800	UPC-E
SKDV_BCR_QR	0x01000000	QR Code

○iCheckDigit

Set check digit per barcode type

・CODE-39 / Interleaved 2 of 5（ITF）

Constant	Value	Description
SKDV_BCR_CD_NONE	0	No check
SKDV_BCR_CD_ENABLE	1	Check

・NW-7

Constant	Value	Description
SKDV_BCR_CD_NONE	0	No check
SKDV_BCR_CD_NW7_M16	1	Modulus 16
SKDV_BCR_CD_NW7_M11	2	Modulus 11
SKDV_BCR_CD_NW7_M10W2	3	Modulus 10 / 2
SKDV_BCR_CD_NW7_M10W3	4	Modulus 10 / 3
SKDV_BCR_CD_NW7_7DR	5	7 check DR
SKDV_BCR_CD_NW7_WM11	6	Weighted modulus 11
SKDV_BCR_CD_NW7_RUNES	7	Runes

○iOption

Set option per barcode type

・UPC-A output digit

Constant	Value	Description
SKDV_BCR_CD_UPC_A_DIGIT_12	0	Output in 12 digits
SKDV_BCR_CD_UPC_A_DIGIT_13	1	Output in 13 digits

・Add UPC-E system code

Constant	Value	Description
SKDV_BCR_CD_UPC_E_NO_CODE	0	Not to add system code
SKDV_BCR_CD_UPC_E_ADD_CODE	1	Add system code

5.4.29. SK_DV_OPT_BCR_CONF_EX

Structure	Barcode reading condition

Syntax

```
typedef struct tag_SK_DV_OPT_BCR_CONF_EX
{
     int                        iScanningSide;
     int                        iColorType;
     int                        iAreaNumber;
     SK_DV_OPT_BCR_AREA         Area[SKDV_BCR_AREA_MAX];
}SK_DV_OPT_BCR_CONF_EX;
```

Member	iScanningSide	Reading side
	iColorType	Set color of reading image
	iAreaNumber	Set number of barcode reading
	Area[SKDV_BCR_AREA_MAX]	Refer SK DV OPT BCR AREA Workable up to number set in iAreaNumber

Description

○iScanningSide

Constant	Value	Description
SKDV_SIDE_NONE	0	No reading image
SKDV_SIDE_BOTH	1	Non usage
SKDV_SIDE_FRONT	2	Only front side
SKDV_SIDE_BACK	3	Only back side

○iColorType

Constant	Value	Description
SKDV_LIGHT_COLOR	0	Non usage
SKDV_LIGHT_GRAYSCALE	1	Grayscale (monochrome)
SKDV_LIGHT_RED	2	Red (plain color)
SKDV_LIGHT_GREEN	3	Green (plain color)
SKDV_LIGHT_BLUE	4	Blue (plain color)

5.4.30. SK_DV_REQ_PRINT_EJECT

Structure	Print & Eject	

Syntax	typedef struct tag_SK_DV_REQ_PRINT_EJECT
	{
	int iStop;
	int iPrint;
	int iReject;
	char szString[SKDV_PRINT_BUFFER_SIZE];
	}SK_DV_REQ_PRINT_EJECT;	

Member	iStop	Stop
	iPrint	Print
	iReject	Reject
	szString[SKDV_PRINT_BUFFER_SIZE]	Set printing character when print is enabled. Printable character depends on models. Refer to commnd reference.
Description	Fold direction of print & eject in continuous reading ○iStop、iPrint、iReject	

Constant	Value	Description
SKDV_DISABLE	0	Disable
SKDV_ENABLE	1	Enable

5.5. API function - Basic function

5.5.1. SkDv_GetModuleInfo

Function	Get Name and Version of library.	
Prototype	void SkDv_GetModuleInfo(SK_DV_MODULE_INFO* pModuleInfo)	
Parameter	pModuleInfo	Set the address to fold infomation. Refer to Structure description "5.4.1.SK_DV_MODULE_INFO"
Return	Non	
Description	Not necessary to open this operation in advance in case library has loaded. After execute, Name and Version of this module are stored into "pModuleInfo"	

5.5.2. SkDv_OpenSingle

Function	Search OMR Device connected with USB, and enable to communication.	
Prototype	SkDvStatus SkDv_OpenSingle(SkDvHandle *phSkDevice)	
Parameter	phSkDevice	Set the pointer to receive open Device controlling handle.
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure (There is no operable Device, or Device is already connected.)
Description	To execute this Function, the preparations to keep memory to workspace and proceed advance communication with Device, then it enables various operations. Need to execute this procedure in advance of operation. In case more than one unit of OMR is connected by internal management, one OMR Device will be connected. In case of success, Device controlling handle is stored into "phSkDevice". If failed, NULL(0) is stored into "phSkDevice". By using this handle, various operations of this Middleware are proceeded. Attn: In case of use this Function, it is not able to use Old API library. "SkDv_OpenSingle" can correspond with only one OMR only, and return Error in case there is another operating Devices.	

5.5.3. SkDv_OpenWithOmrapi

Function	Search OMR Device -USB connected, and enables communication. Controlling by Old API library is also be available.	
Prototype	SkDvStatus　SkDv_OpenWithOmrapi(SkDvHandle *phSkDevice)	
Parameter	phSkDevice	Set the pointer to receive open Device controlling handle.
Return	SKDV_STS_SUCCESS	Succesful
	Except above	Failure (There is no operable Device, or Device is already connected.)
Description	By executing Old API "OMR_OpenDeviceUSB", Old API Function is to be available. Same as "SkDv_OpenSingle".	

5.5.4. SkDv_Close

Function	Execute when finishing all the operation, and release various memories to workspace.	
Prototype	SkDvStatus　SkDv_Close(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle.
Return	SKDV_STS_SUCCESS	Successful
	Except　above	Failure (There is no operable Device, or Device is already connected.)
Description	Attn: Use the controlling handle received by opening operation. Attn 2: In case of re-opening the same Device, controlling handle will change.	

5.5.5. SkDv_ReqInit

Function	Execute “Software reset” for device, and gets various operating condition from the device.	
Prototype	SkDvStatus　SkDv_ReqInit(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle.
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Reset Software is executed. Device operating condition, Image data and internal operating data are all to be initialized. In addition, get each operating condition and device information from the device.	

5.5.6. SkDv_GetInfo

Function	Get verious infomation of device controlling and infomation of function	
Prototype	SkDvStatus SkDv_GetInfo(SkDvHandle hSkDevice, SK_DV_SR3500_INFO* pDvInfo)	
Parameter	hSkDevice	Set Device controlling handle
	pDvInfo	Set the address of Structure to fold device infomation. Description can be referred at "5.4.3.SK_DV_SR3500_INFO"
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Proceed communicate with device and store information into “pDvInfo”in case device information has not got. Call “SkDv_ReqInit” or “SkDv_ReqGetInfo”in case communicate with device in advance.	

5.5.7. SkDv_GetMode

Function	Get Operating mode stored in Library.	
Prototype	SkDvStatus　SkDv_GetMode(SkDvHandle hSkDevice, SK_DV_SR3500_MODE* pDvMode)	
Parameter	hSkDevice	Set Device controlling handle
	pDvMode	Set the address of Structure to fold Operation mode. Description can be referred at "0. SK_DV_SR3500_MODE"
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Operate mode to get is saved in this library. Operation mode saved by this library is stored into Parameter "pDvMode". Communication with Device is not done. In case want to call in Device right after starting, please call Function "SkDv_ReqInit" or "SkDv_ReqGetMode".	

5.5.8. SkDv_SetMode

Function	Set Operation mode in Library	
Prototype	SkDvStatus　SkDv_SetMode(SkDvHandle hSkDevice, SK_DV_SR3500_MODE* pDvMode)	
Parameter	hSkDevice	Set Device controlling handle
	pDvMode	Description can be referred at "0. SK_DV_SR3500_MODE"
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	Fold the device information into library in case not connect with device and get the device information. "pDvMode" is checked as referring the description of device infomation, and error return in case the value has any problems, If there is no error, Operation mode is stored into Library. Operation mode can be set through "SkDv_ReqSetMode"　into device.	

5.5.9. SkDv_GetMarkConf

Function	Get Operating condition for mark reading from Library	
Prototype	SkDvStatus　SkDv_GetMarkConf(SkDvHandle hSkDevice, SK_DV_SR3500MARK_CONF* pDvMarkConf)	
Parameter	hSkDevice	Set Device controlling handle
	pDvMarkConf	Set the address to fold Operating condition for Mark reading. Description can be seen "5.4.5.SK_DV_SR3500MARK_CONF".
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Operating condition for Mark reading is stored into "pDvMarkConf". Call "SkDv_ReqInit" or "SkDv_ReqGetMarkConf" in case want to see the contents stored in the device	

5.5.10. SkDv_SetMarkConf

Function	Set operating condition for Mark reading into Library.	
Prototype	SkDvStatus　SkDv_SetMarkConf(SkDvHandle hSkDevice, SK_DV_SR3500MARK_CONF* pDvMarkConf)	
Parameter	hSkDevice	Set Device controlling handle
	pDvMarkConf	Set the address to fold Operating condition for Mark reading Description can be referred at "5.4.5.SK_DV_SR3500MARK_CONF".
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Fold the device information into library in case not connect with device and get the device information. "pDvMarkConf" is checked as referring the description of device infomation, and error return in case the value has any problems, If there is no error, operating condition for Mark reading is stored into Library. Operation mode can be set through "SkDv_ReqSetMarkConf"　into device.	

5.5.11. SkDv_ReqGetStatus

Function	Get Device status infomation by communication.	
Prototype	SkDvStatus　SkDv_ReqGetStatus(SkDvHandle hSkDevice, SK_DV_SR3500_STATUS_CHAR* pStatusChar)	
Parameter	hSkDevice	Set Device controlling handle
	pStatusChar	Set the address to fold Device status code infomation. Description can be referred at "5.4.8.SK_DV_SR3500_STATUS_CHAR".
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Use “request command for Error information” "DE" to get error information and status infomation. Device error is returned on Return value.	

5.5.12. SkDv_ReqGetSensor

Function	Get device sensor condition by communication.	
Prototype	SkDvStatus　SkDv_ReqGetSensor(SkDvHandle hSkDevice, DWORD* pdwSensor)	
Parameter	hSkDevice	Set Device controlling handle
	pdwSensor	Set the address of Variable to fold Sensor condition
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Use “request command for Sensor condition” "DS", to get error information and status infomation.　Sensor information are folded in "pdwSensor" ○pdwSensor	

Constant	Value	Description
SKDV_SENSOR_BIT_OUTPS	0x00000020	OUTPS
SKDV_SENSOR_BIT_RDPS	0x00000010	RDPS
SKDV_SENSOR_BIT_INPS	0x00000008	INPS
SKDV_SENSOR_BIT_PS0	0x00000004	PS0
SKDV_SENSOR_BIT_UPPS	0x00000002	UPPS
SKDV_SENSOR_BIT_DWPS	0x00000001	DWPS
SKDV_SENSOR_BIT_SKS	0x00001000	SKS
SKDV_SENSOR_BIT_MAIN_CVR	0x00000100	MAIN-CVR
SKDV_SENSOR_BIT_SPS3	0x00200000	SPS3
SKDV_SENSOR_BIT_SPS2	0x00100000	SPS2
SKDV_SENSOR_BIT_SPS1	0x00080000	SPS1
SKDV_SENSOR_BIT_SPS	0x00040000	SPS
SKDV_SENSOR_BIT_MPS	0x00020000	MPS
SKDV_SENSOR_BIT_P2PS	0x00010000	P2PS
SKDV_SENSOR_BIT_SRPS	0x10000000	SRPS
SKDV_SENSOR_BIT_MRPS	0x08000000	MRPS
SKDV_SENSOR_BIT_SPS5	0x04000000	SPS5
SKDV_SENSOR_BIT_SPS4	0x02000000	SPS4
SKDV_SENSOR_BIT_STK_CVR1	0x01000000	STK-CVR1

5.5.13. SkDv_ReqFeedMarkSheet

Function	Read Mark sheet.	
Prototype	SkDvStatus　SkDv_ReqFeedMarkSheet(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Read Mark sheet by Command "SF". Read the mark data Not read Image data. Connect with device in case not get the setting from the device	

5.5.14. SkDv_ReqEjectForm

<table>
<tr><td>Function</td><td colspan="2">Read Mark sheet, and capturing Image data.</td></tr>
<tr><td>Prototype</td><td colspan="2">SkDvStatus　SkDv_ReqEjectForm(SkDvHandle hSkDevice, int iDirection)</td></tr>
<tr><td rowspan="2">Parameter</td><td>hSkDevice</td><td>Set Device controlling handle</td></tr>
<tr><td>iDirection</td><td>Set the operation of eject sheet.
Setting value can be refered at "Description"</td></tr>
<tr><td rowspan="2">Return</td><td>SKDV_STS_SUCCESS</td><td>Successful</td></tr>
<tr><td>Except above</td><td>Failure</td></tr>
<tr><td>Description</td><td colspan="2">Eject the sheets by command of Ejection "ER".
If the stacker unit is not connected, switching of Main stacker and Select stacker is not work (SR3500HYBRID).
○iDirection
<table>
<tr><th>constant</th><th>value</th><th>description</th></tr>
<tr><td>SKDV_EJECT_MAIN</td><td>1</td><td>Ejecting the sheet at stand-by position into Main stacker
(instant operation)</td></tr>
<tr><td>SKDV_EJECT_SELECT</td><td>2</td><td>Ejecting the sheet at stand-by position into Select stacker
(instant operation)</td></tr>
<tr><td>SKDV_EJECT_MAIN_NEXT</td><td>3</td><td>Ejecting the sheet at stand-by position into Main stacker at the timing of next Reading command</td></tr>
<tr><td>SKDV_EJECT_SELECT_NEXT</td><td>4</td><td>Ejecting the sheet at stand-by position into Select stacker at the timing of next Reading command</td></tr>
</table></td></tr>
</table>

5.5.15. SkDv_ReqClearError

Function	Clear the Error on Device.	
Prototype	SkDvStatus　SkDv_ReqClearError(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	By Error clearing command "CE", clearing the Error happening in Device. However, in case of Non-clearable Error, it would return another massage of Error.	

5.5.16. SkDv_ReqGetMarks

Function	Get Mark data	
Prototype	SkDvStatus SkDv_ReqGetMarks(SkDvHandle hSkDevice, int iFace, SK_DV_MARK_INFO *pMarkInfo, char* pMarks, int* piBufSize)	
Parameter	hSkDevice	Set Device controlling handle
	iFace	Select front side or back side
	pMarkInfo	Set the address to fold Mark data Description can be referred at "5.4.9.SK_DV_MARK_INFO"
	pMarks	Set the address of buffer to fold Mark data Mark density is stored per 1 byte in the value of 0 to 16 In case you appointed "NULL", Mark data cannot be stored
	piBufSize	Set the byte size of buffer to fold Mark data Store data into the buffer appointed in “pMark” up to the buffer size After execution, the needed buffer size will be stored
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Use request code for Mark density data "MD", to get Mark data. Set the address of Mark infomation nto Parameter "pMarkInfo". Set the first address of buffer securing memories for unit number of Mark data, and the byte size of the buffer into "piBufSize"	

5.6. APIfunction - expanded Function

5.6.1. SkDv_ReqGetInfo

Function	Communicate to get Device infomation.	
Prototype	SkDvStatus　SkDv_ReqGetInfo(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Communicate to get Device infomation. and saves it into Library. Using commands is as following. -request command of Device infomation "DI" - request command of Model name "MN" - request command of Firmware version "FV" - Other - Closed command	

5.6.2. SkDv_ReqGetMode

Function	Get Operation mode.	
Prototype	SkDvStatus　SkDv_ReqGetMode(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Get operation mode, and saves it into Library. Using commands is as following. - Setting command of Warning error "WE" - Setting command of Panel operation "PO" - Setting command of Buzzer "BZ" - Setting command of Energy saving "ES"	

5.6.3. SkDv_ReqSetMode

Function	Set operating mode.	
Prototype	SkDvStatus　SkDv_ReqSetMode(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Set the opration mode in Library into Device. Command is as same as "SkDv_ReqGetMode"	

5.6.4. SkDv_ReqGetMarkConf

Function	Get mark condition	
Prototype	SkDvStatus　SkDv_ReqGetMarkConf(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Appinting Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Get Mark condition, and to store it into Library. Using command is as following. - Setting command for Reading column "NC" - Setting command for the Reading method "RM" - Setting command for Back side reading "BR" - Setting command for Sheet thickness "FT"	

5.6.5. SkDv_ReqSetMarkConf

Function	Set Mark condition.	
Prototype	SkDvStatus　SkDv_ReqSetMarkConf(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Set the Mark condition in Library into Device. Command is as same as "SkDv_ReqGetMarkConf"	

5.7. API function - Image reading

5.7.1. SkDv_GetImageConf

Function	Settling Operating condition for Image reading into Library. For SR-3500HYBRID siries (SR-8000HYBRID with constraint)	
Prototype	SkDvStatus SkDv_GetImageConf(SkDvHandle hSkDevice, SK_DV_SR3500IMG_CONF* pDvImageConf)	
Parameter	hSkDevice	Appointing Device controlling handle
	pDvImageConf	Appointing the address of Structure to fold Operating condition for Mark reading Structure description can be referred at "5.4.6.SK_DV_SR3500IMG_CONF".
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Operating condition for Image reading to get is saved in this Library. Operating condition for Image reading is folded in Parameter "pDvMarkConf". Communication with Device is not done. If you would like to get description settled in Device right after starting, Please call Function "SkDv_ReqInit" or "SkDv_ReqGetImageConf". This Function is only for Image reader equipment, not use for without Image reader.	

5.7.2. SkDv_SetImageConf

Function	Settling Operating condition for Image reading into Library. Foror SR-3500HYBRID siries (SR-8000HYBRID with constraint)	
Prototype	SkDvStatus SkDv_SetImageConf(SkDvHandle hSkDevice, SK_DV_SR3500IMG_CONF* pDvImageConf)	
Parameter	hSkDevice	Appointing Device controlling handle
	pDvImageConf	Appointing the address of Structure to fold Operating condition for Image reading Structure description can be referred at "5.4.6.SK_DV_SR3500IMG_CONF".
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	If getting Device info. has not been done, you can communicate with Device and fold the Device info. into library. Operation mode for Image reading appointed at Parameter "pDvImageConf" is checked as referring to Device info's description, and if the value has any problems, Error is returned. If there is no error, Operation mode is folded into Library. Setting description of Operation mode for Mark reading can be settle into Device by executing "Demand for Reading" or Function "SkDv_ReqImageConf" via communication. This Function is only for Image reader equipment, not use for without Image reader.	

5.7.3. SkDv_GetImageConfEx

Function	Get condition of image reading folded in library (For SR-8000HYBRID)	
Prototype	SkDvStatus SkDv_GetImageConfEx(SkDvHandle hSkDevice, SK_DV_IMAGE_CONF_EX* pDvImageConfEx)	
Parameter	hSkDevice	Appointing Device controlling handle
	pDvImageConfEx	Appointing the address of Structure to fold Operating condition for Image reading Structure description can be referred at "5.4.7SK_DV_IMAGE_CONF_EX".
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Operating condition for Image reading to get is saved in this Library. Operating condition for Image reading is folded in Parameter "pDvMarkConfEx". Communication with Device is not done. If you would like to get description settled in Device right after starting, Please call Function "SkDv_ReqGetImageConft" or "SkDv_ReqInit". This Function is only for SR-8000HYBRID series, not use for other readers.	

5.7.4. SkDv_SetImageConfEx

Function	Settling Operating condition for Image reading into Library. (For SR-8000HYBRID)	
Prototype	SkDvStatus SkDv_SetImageConf(SkDvHandle hSkDevice, SK_DV_IMAGE_CONF_EX* pDvImageConfEx)	
Parameter	hSkDevice	Appointing Device controlling handle
	pDvImageConfEx	Appointing the address of Structure to fold Operating condition for Image reading Structure description can be referred at "5.4.7SK_DV_IMAGE_CONF_EX".
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	If getting Device info. has not been done, you can communicate with Device and fold the Device info. into library. Operation mode for Image reading appointed at Parameter "pDvImageConfEx" is checked as referring to Device info's description, and if the value has any problems, Error is returned. If there is no error, Operation mode is folded into Library. Setting description of Operation mode for Mark reading can be settle into Device by executing "Demand for Reading" or Function "SkDv_ReqSetImageConf" via communication. This Function is only for SR-8000HYBRID series, not use for other readers.	

5.7.5. SkDv_ReqGetImageConf

Function	Communicating to get Image condition from Device.	
Prototype	SkDvStatus　SkDv_ReqGetImageConf(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Appointing Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Communicating to get Image condition, to save it into Library. Using command is as following. - Setting command for Image reading "IR" This Command is only for Image reader equipment, not use for without Image reader.	

5.7.6. SkDv_ReqSetImageConf

Function	Communicating to settle Image condition into Device.	
Prototype	SkDvStatus　SkDv_ReqSetImageConf(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Appointing Device controlling handle
Return value	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Settling the Image condition in Library into Device. Using command is as same as "SkDv_ReqGetImageConf" Function. This Command is only for Image reader equipment, not use for without Image reader.	

5.7.7. SkDv_ReqScanForm

Function	Reading Mark sheet, and capturing Image data.	
Prototype	SkDvStatus　SkDv_ReqScanForm(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Appointing Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Reading Mark sheet by Command "NS" If the communication for getting Image/setting Condition/CIS standard data has not been done, the communication will be done before Reading. Reading sheet and folding Image data into this library. Since only models with an image reading function, if not installed will not be used.	

5.7.8. SkDv_GetFormSize

Function	Getting the info of sheet size	
Prototype	SkDvStatus SkDv_GetFormSize(SkDvHandle hSkDevice, SIZE* pSizeForm)	
Parameter	hSkDevice	Appointing Device controlling handle

	pSizeForm	Appointing the address of Structure to fold the size of sheet. The measure of value is 0.1mm.
		SIZE can be referred at "MFC Library reference"
		typedef struct tagSIZE
		{
		LONG cx;
		LONG cy;
		} SIZE, *PSIZE, *LPSIZE;

Return	SKDV_STS_SUCCESS	Successful
	Except for the above.	Failure
Description	Getting the sheet size from Image data in Middleware. Width"cx" is the maximum width which Device can read, and Length"cy" is the length which Device has detected by sensor. Above "Image data in Middleware" means Image data captured by "SkDv_ReqScanForm" Function. This Command is only for Image reader equipment, not use for without Image reader.	

5.7.9. SkDv_SetImageElongationDetectInfo

Function	Information for detecting the elongation of the read image is set.	
Prototype	SkDvStatus SkDv_SetImageElongationDetectInfo(SkDvHandle hSkDevice, SK_DV_IMG_ELGT_DETECT_CONF* pElongationInfo)	
Parameter	hSkDevice	It specifies device controlling handle.
	pElongationInfo	It specifies the address of the structure to enter information for detecting the elongation of the image. The content of the structure can be referred at " 5.4.12 SK_DV_IMG_ELGT_DETECT_CONF"
Return	SKDV_STS_SUCCESS	Success
	Except for the above.	Failure
Description	Since "5.13Image Elongation Detection" also has explanations and notes,so be sure to check it. Information for detecting the elongation of the image in the middleware is set. The above "image in the middleware" means image data acquired by the "SkDv_ReqScanForm" function. In "blEnabled", by “true” or “false”, it specifies if image elongation detection is performed after image reading. In "SheetLen", specify the length of the sheet to be used in units of 0.1 mm. (Min:110, Max:355.6) In "Threshold", it specifies the maximum length that is not judged that the image became longer. (10mm, 20mm, and 30mm) In the image elongation detection, only the elongation is detected and the contraction is not detected. The image elongation detection is performed only on the image of the back side of the sheet. Therefore, under the condition that image enlongation detection is enabled, if it orders reading only front side, an error occurs. (SKDV_STS_PARAM_ERR_SCANNINGSIDE Parameter Error Back side reading) This command is only for HYBRID models and must not used for other models.	

5.7.10. SkDv_GetImageElongationDetectInfo

Function	Information for detecting the elongation of the image stored in the middleware is acquired.	
Prototype	SkDvStatus SkDv_GetImageElongationDetectInfo(SK_DV_IMG_ELGT_DETECT_CONF* pElongationInfo)	
Parameter	hSkDevice	Device controlling handle is specified.
	pElongationInfo	The address of the structure to enter information for detecting the elongation of the image is specified. The content of the structure can be referred at "5.4.12SK_DV_IMG_ELGT_DETECT_CONF"
Return	SKDV_STS_SUCCESS	Success
	Except for the above.	Failure
Description	Image elongation detection set by SkDv_SetImage Elongation DetectInfo is acquired. In “blEnabled”, a boolean value indicating if image elongation detection is performed is stored. In “SheetLen”, the length of the form to be subjected to image elongation detection is stored. In “Threshold”, a threshold value for performing image elongation detection is stored. This command is only for HYBRID models and must not used for other models.	

5.7.11. SkDv_SaveImageData

Function	Saving Image data appointed.	
Prototype	SkDvStatus SkDv_SaveImageData(SkDvHandle hSkDevice, SK_DV_IMAGE_DATA_CONF* pImageDataConf, SK_DV_IMAGE_FILE_CONF* pImageFileConf, char* pszFilename)	
Parameter	hSkDevice	Appointing Device controlling handle
	pImageDataConf	Appointing the address of Structure which appointed the Image data getting condition. The content of Structure can be referred at "5.4.10.SK_DV_IMAGE_DATA_CONF"
	pImageFileConf	Appointing the address of Structure which appointed the File form to save. The content of Structure can be referred at "5.4.11.SK_DV_IMAGE_FILE_CONF"
	pszFilename	Appointing the address of character string which folds Name of File to save.
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Saving the Image data in this library into File. Above "Image data in Middleware" means Image data captured by "SkDv_ReqScanForm" Function. In case saving is executed in resolution high than image, the quality of Image is to be rough than to saving is executed. And in case saving is Color but the image is grayscale, saved image is to be Grayscale. As for File name of these image file, it's required to check overwrite or forbidden words in advance before call this function. This Command is only for Image reader equipment, not use for without Image reader.	

5.7.12. SkDv_SaveImageDataThread

Function	Saving the appointed Image data by other threads	
Prototype	SkDvStatus WINAPI SkDv_SaveImageDataThread(SkDvHandle hSkDevice, SK_DV_IMAGE_DATA_CONF* pImageDataConf, SK_DV_IMAGE_FILE_CONF* pImageFileConf, char* pszFilename, FN_SK_DV_SAVEIMAGE_CALLBACK fnSaveImageCallback, LPVOID lpParameter)	
Parameter	hSkDevice	Appoint Device Control Handle
	pImageDataConf	Appoint the address of structure which appointed the setting condition of Image Data See "5.4.10.SK_DV_IMAGE_DATA_CONF" refer for the contents of structure.
	pImageFileConf	Appoint the address of structure which appoints the file type to be saved. See "5.4.11.SK_DV_IMAGE_FILE_CONF" refer for the contents of structure.
	pszFilename	Appoint the address of structure which appoints the name of save name
	fnSaveImageCallback	Appointing the pointer of Callback functions after finish Image Data saving. Not callback acceptable while "NULL"
	lpParameter	User Parameter hand over for Callback function
Return	SKDV_STS_SUCCESS	Successful
	Except for the Above	Failure
Description	Save the Image file which is keep in this library to the other Threads. The Function of "SkDv_SaveImageDataThread" would be finish prior to file saved completely. It can be operated by separated Threads for save Image Data (Front and Back side), but Data is holding in this API function in case not finished the previous Threads. Parameters "SkDv_SaveImageData" and "SkDv_CreateImage" can not save the Image Data until the following reading start in case execute this function. The Contents to be saved file is same as while use function of "SkDv_SaveImageData" No Callback executed while Parameter "fnSaveImageCallback"is "NULL". Then No information while save incorrectly. That's why, User application should have function to detect whether file saved correctly. Function "SkDv_CreateImage" has function to be judgment saved file is actuary made or not saved yet. It can be received the Error Status by the Parameter in case set the callback function in Parameter "fnSaveImageCallback" See "5.7.13FN_SK_DV_SAVEIMAGE_CALLBACK" for the reference of callback. This Function is only for Image reader equipment, not use for without Image reader.	

5.7.13. FN_SK_DV_SAVEIMAGE_CALLBACK

Function	Call Back function for Saving the Image Data by other Threads	
Prototype	BOOL (CALLBACK *FN_SK_DV_SAVEIMAGE_CALLBACK)(SK_DV_IMAGE_DATA_CONF* pImageDataConf, SK_DV_IMAGE_FILE_CONF* pImageFileConf, char* pszFilename, SkDvStatus Status, LPVOID lpParameter)	
Parameter	pImageDataConf	Appoint address of Structure which Appoints the setting of condition for the folding Image Data
	pImageFileConf	Appoint Address of the structure to appoint the saving extension (Format)
	pszFilename	Pointer which folded the image file name.
	Status	Folded Status code to be saved the image data files by other threads.
	lpParameter	Folded the user parameter which appointed SkDv_SaveImageDataThread
Return	TRUE	Saving the Image Data at other threads by the following setting
	FALSE	Finish the saving Image Data by other Threads.
Description	This function is to set into User Application to receive the result Data of File Saving by "SkDv_SaveImageDataThread" function. It would be better to add the error processing because Parameter "Status" folds the status code of results saving Image Data in other threads. Return "True" to save the Image Data Files by Parameter "pImageDataConf" "pImageFileConf" "pszFilename" (need change these contents) , Return "FALSE" to end other threads to save the file.	

5.7.14. SkDv_IsThreadRunning

Function	Confirm Threads created by "SkDv_SaveImageDataThread" function has closed.	
Prototype	BOOL WINAPI SkDv_IsThreadRunning(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Appoint Device controlling handle
Return	TRUE	Thread running saving image data files.
	FALSE	All threads have finished saving the image data files.
Description	Possibility to confirm other Threads (to be save Image files) made by the function of SkDv_SaveImageDataThread" has closed. Recommend to use this function in case use the function "SkDv_SaveImageDataThread"without callback. This Function is only for Image reader equipment, not use for without Image reader.	

5.7.15. SkDv_CreateImage

Function	Generating Image data in appointed range.	
Prototype	SkDvStatus　SkDv_CreateImage (SkDvHandle hSkDevice, SK_DV_IMAGE_DATA_CONF* pImageDataConf, BITMAPINFO** ppBitmapInfo, void** ppBitmapBits)	
Parameter	hSkDevice	Appointing Device controlling handle
	pImageDataConf	Appointing the address of Structure which appointed the Image data getting condition. The content of Structure can be referred at "5.4.10.SK_DV_IMAGE_DATA_CONF"
	ppBitmapBits	Appointing the address to get the address folding the bit data of Image data.
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Securing the memories folding Image data, and to return the address folding Image data in this library. As the memories are secured in this Function, you have to release the memories at "SkDv_DestroyImage" However, not able to get Image data unless executing "Demand for Reading". This Function is only for Image reader equipment, not use for without Image reader.	

5.7.16. SkDv_DestroyImage

Function	Destroying the Image data generated at "SkDv_CreateImage"	
Prototype	SkDvStatus　SkDv_DestroyImage (SkDvHandle hSkDevice, BITMAPINFO* pBitmapInfo)	
Parameter	hSkDevice	Appointing Device controlling handle
	pBitmapBits	Appointing the address folding the bit data gained at "SkDv_CreateImage".
Return	SKDV_STS_SUCCESS	Successful
	Except for the above	Failure
Description	Releasing the memories secured at "SkDv_CreateImage" However, Image data cannot be saved unless executing "Demand for Reading". This Function is only for Image reader equipment, not use for without Image reader.	

5.7.17. SkDv_ReqGetImgReadSheetLength

Function	Get Device image reading sheet length infomation by communication.	
Prototype	SkDvStatus SkDv_ReqGetImgReadSheetLength(SkDvHandle hSkDevice, int* pImgReadSheetLen)	
Parameter	hSkDevice	Set Device controlling handle
	pImgReadSheetLen	Please set the address of the variable for storing the image reading sheet length information. Setting value can be refered at "Description"
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Use request command for Set Image Reading Sheet Length "IL", to get Length infomation. Image reading sheet length information are folded in " pImgReadSheetLen ". In the image-featured models, available only models that "Setting of Reading" "Paper Size" in the menu in the main body panel is displayed. Non-compatible models are not set. ○pImgReadSheetLen	

constant	Value	description
SKDV_IMG_SHEET_14INCH	0	Image reading sheet Maximum length is 14inch.
SKDV_IMG_SHEET_A4	1	Image reading sheet Maximum length is A4 size.

5.7.18. SkDv_ReqSetImgReadSheetLength

Function	Get Device image reading sheet length infomation by communication.	
Prototype	SkDvStatus SkDv_ReqSetImgReadSheetLength(SkDvHandle hSkDevice, int ImgReadSheetLen)	
Parameter	hSkDevice	Set Device controlling handle
	ImgReadSheetLen	Please set the variable for storing the image reading sheet length information. Setting value can be refered at "Description"
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Use request command for Get Image Reading Sheet Length "IL", to set Length infomation. If the sheet length exceeds the specified size is read, the image reading length is up to specified size. In the image-featured models, available only models that "Setting of Reading" "Paper Size" in the menu in the main body panel is displayed. Non-compatible models are not set. ○ImgReadSheetLen	

constant	value	description
SKDV_IMG_SHEET_14INCH	0	Image reading sheet Maximum length is 14inch.
SKDV_IMG_SHEET_A4	1	Image reading sheet Maximum length is A4 size.

5.8. APIfunction - Continuous Reading

5.8.1. SkDv_ReqSetLayoutManage

Function	Set ID for window layout management	
Prototype	SkDvStatus SkDv_ReqSetLayoutManage(SkDvHandle hSkDevice, SK_LAYOUT_MANAGE_CONF* pLayManConf)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pLayManConf	Set the pointer of ID data setting
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure (No device connected, or device is connecting)
Description	"ID managed data setting [CMD:WM]" command is send to OMR. All registered layout setting of OMR is cleared, and set layout ID setting. Ex: setting of recognition of layout ID of Window controls, such as use / not use ID, and the contents of the ID window. In case window control is performed, there should runs first. It should perform " Finish the layout setup " function after all. (It is necessary to run before set "ID data sets" and "windows areas data sets") “IdWindowPrm.IPartition” of “pLayManConf”would be disregarded due to Partition division cannot be performed in ID window.	

5.8.2. SkDv_ReqSetLayoutManageTerminate

Function	Finish the layout setup (set ID data, set windows areas data sets)	
Prototype	SkDvStatus SkDv_ReqSetLayoutManageTerminate(SkDvHandle hSkDevice, int* piLimit)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	piLimit	Set the pointer to store the rectriction of continuous reading. “NULL” can be set. Not stored any restricted states set “NULL”
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure (No device connected, or device is connecting)
Description	"ID management data setting [CMD:WM]" command is send to OMR. Finish the setting of Window control set. Don't perform "SetLayoutID” and "SetWindowArea" after execution.	

5.8.3. SkDv_ReqSetLayoutId

Function	Request form ID setting and enables to set up window areas.	
Prototype	SkDvStatus SkDv_ReqSetLayoutId(SkDvHandle hSkDevice, SK_LAYOUT_ID_CONF* pLayIdConf, int* piIdIdx)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pLayIdConf	Set the pointer of ID data set the setting
	piIdIdx	Set the pointer to store the ID number (from 0). Null can be set . Null is not stored.
Return	SKDV_STS_SUCCESS	Succesful
	Except Above	Failure (No device connected, or device is connecting)
Description	"ID data setting [CMD:WL]" command is send to OMR. Set the ID pattern of ID window, and set the number of timing marks of the sheet. In case made a setting does not use ID, ID setting is disregarded. but have to call once. After performing it, "SetWIndowArea" can be set	

5.8.4. SkDv_ReqSetWindowArea

Function	Setup window area.	
Prototype	SkDvStatus SkDv_ReqSetWindowArea(SkDvHandle hSkDevice, SK_WINDOW_AREA_CONF* pWinAreaConf, int* piWinIdx)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pWinAreaConf	Set the pointer of window area to set the setting
	piWinIdx	Set the pointer to store the window number (from 0). Null can be set. “Null” window is not stored.
Return	SKDV_STS_SUCCESS	Succesful
	Except above	Failure (No device connected, or device is connecting)
Description	"Window domain data setting [CMD:WD]" command is send to OMR. The last “ID data setup" is performed	

5.8.5. SkDv_ReqClearLayout

Function	Clear all the contents of the layout setting.	
Prototype	SkDvStatus SkDv_ReqClearLayout(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set the pointer to receive the device Device controll handle
Return	SKDV_STS_SUCCESS	Succesful
	Except above	Failure (No device connected, or device is connecting)
Description	"Window information Clear [CMD:WC]" command is send to OMR. The contents of "ID data management settings", "ID data settings", and "windows areas data settings" are cleared.	

5.8.6. SkDv_PollingPrintEject

Function	Polling call back function for print & eject in continuous reading Only for SR-8000HYBRID	
Prototype	BOOL WINAPI SkDv_PollingPrintEject(SkDvHandle hSkDevice, FN_SK_DV_CONTFEED_CALLBACK fnContFeedCallBack, LPVOID lpParameter)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	fnContFeedCallBack	Set the call back function
	lpParameter	Set the parameter to feed call back function
Return	TRUE	Successful (sent print & eject direction)
	FALSE	Failure (Not sent print & ejet direction)
Description	By call this function, call call-back for print/eject specified by parameter from this function when mark data and barcode data have been received. In its call-back function, it is necessary to write a code that judges data and specify print/eject. Reading operation is performed continuously, so processing must have been completed in short time. (Refer to the command reference regarding the processing time) It must not be used SkDv_RegistPrintEjectCallback function together. Use when it is not compliant to multi thread like VB6.0. If calling this function is delayed, time to process in call-back function shortens, so it is necessary to call it frequently. It is also necessary to process data that has been read.	

5.8.7. SkDv_RegistPrintEjectCallback

Function	Registering call back function for print & eject in continuous reading Only for SR-8000HYBRID	
Prototype	BOOL WINAPI SkDv_RegistPrintEjectCallback(SkDvHandle hSkDevice, FN_SK_DV_CONTFEED_CALLBACK fnContFeedCallBack, LPVOID lpParameter)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	fnContFeedCallBack	Set the call back function Set NULL to release the registration
	lpParameter	Set the parameter to feed call back function
Return	TRUE	Successful
	FALSE	Failure
Description	Register the function if data from OMR is received. In its call-back function, it is necessary to write a code that judges data and specify print/eject. Reading operation is performed continuously, so processing must have been completed in short time. (Refer to the command reference regarding the processing time) This function is only for registration but not to call back. If mark data and barcode data is received during continuous reading, it call back from worker thread. It can’t be used with SkDv_PollingPrintEject function. It needs to be designed in thread-safe, since it is multi-thread function.	

5.8.8. FN_SK_DV_CONTFEED_CALLBACK

Function	Call back function to save image data file Only for SR-8000HYBRID	
Prototype	void (CALLBACK *FN_SK_DV_CONTFEED_CALLBACK)(int iSheetNumber, SK_DV_REQ_PRINT_EJECT* pPrintEject, LPVOID lpParameter)	
Parameter	iSheetNumber	The number of reading page is folded in. It can be used as necessary.
	pPrintEject	Set the direction to OMR in the print & eject structure, and finish call back function Refer to「5.4.30SK_DV_REQ_PRINT_EJECT」
	lpParameter	The parameter set by user in “5.8.6SkDv_PollingPrintEject”/” 5.8.7SkDv_RegistPrintEjectCallback” is folded in
Return	Non	
Description	A prototype of the callback function used for Continuous reading	

5.8.9. SkDv_GetContDataMarks_Now

Function	Get mark data to judge print & eject direction in continuous reading Only for SR-8000HYBRID	
Prototype	BOOL WINAPI SkDv_GetContDataMarks_Now(SkDvHandle hSkDevice, int iFace, SK_DV_MARK_INFO *pMarkResult, char* pMarks, int* piBufSize)	
Parameter		Refer to SkDv_GetContDataMarks
Return	TRUE	Successful
	FALSE	Failure
Description	Can be used in call back function for print & eject Refer to SkDv GetContDataMarks	

5.8.10. SkDv_GetContDataBarcodesCount_Now

Function	Get number of barcode to judge print & eject direction in continuous reading Only for SR-8000HYBRID	
Prototype	BOOL WINAPI SkDv_GetContDataBarcodesCount_Now(SkDvHandle hSkDevice, int* piReadCount)	
Parameter		Refer to SkDv_GetContDataBarcodesCount
Return	TRUE	Successful
	FALSE	Failure
Description	Can be used in call back function for print & eject Refer to SkDv_GetContDataBarcodesCount	

5.8.11. SkDv_GetContDataBarcodesData_Now

Function	Get barcode data to judge print & eject direction in continuous reading Only for SR-8000HYBRID	
Prototype	BOOL WINAPI SkDv_GetContDataBarcodesData_Now(SkDvHandle hSkDevice, int iIndex, char* pcBcType, char* pszBarcode, int* piDataLen)	
Parameter		Refer to SkDv_GetContDataBarcodesData
Return	TRUE	Successful
	FALSE	Failure
Description	Can be used in call back function for print & eject Refer to SkDv_GetContDataBarcodesData	

5.8.12. SkDv_ReqSetPanelUserEnable

Function	Set character to shows on the LCD display and prohibit operation by LCD buttons.	
Prototype	SkDvStatus SkDv_ReqSetPanelUserEnable(SkDvHandle hSkDevice, BOOL blEnable)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	blEnable	FALSE : Invalid. OMR controls LCD and panel switch. TRUE : Valid. LCD and panel switch can controll by application.
Return	SKDV_STS_SUCCESS	Succesful
	Except above	Failure (No device connected, or device is connecting)
Description	"Panel remote setting [CMD:PC]" command is send to OMR. LCD panel information disappears but can set the characters of LCD display of OMR by "Set Panel User String".	

5.8.13. SkDv_ReqGetPanelUserSwitch

Function	Get the state of the LCD panel button	
Prototype	SkDvStatus SkDv_ReqGetPanelUserSwitch(SkDvHandle hSkDevice, DWORD* pdwSwitch)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pdwSwitch	Set the pointer to store the condition of Panel switch
Return	SKDV_STS_SUCCESS	Succesful
	Except above	Failure (No device connected, or device is connecting)
Description	" Panel remote set [CMD:PC]" command is send to OMR. Display character can be set "remote settings" function is enabled.	

5.8.14. SkDv_ReqSetPanelUserString

Function	Set character for LCD display	
Prototype	SkDvStatus SkDv_ReqSetPanelUserString(SkDvHandle hSkDevice, char* pPanelStr)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pPanelStr	Set the characters for LCD display of OMR(Max 40 characters)
Return	SKDV_STS_SUCCESS	Succesful
	Except Above	Failure (No device connected, or device is connecting)
Description	"Panel remote set [CMD:PC]" command is send to OMR. Display character can be set "remote settings" enabled. The characters are set less than 40 characters to “pPanelStr,” margins are show as space. Exceeded characters are disregarded set over than 40 characters to the “pPanelStr”,	

5.8.15. SkDv_ReqSetPirntFixString

Function	Set the character (fixed) for printer.	
Prototype	SkDvStatus SkDv_ReqSetPrintFixString(SkDvHandle hSkDevice, char* pString)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pString	Set the fixed printing string(MAX 20 characters)
Return	SKDV_STS_SUCCESS	Succesful
	Except Above	Failure (No device connected, or device is connecting)
Description	"Printer setting [CMD:PR]" command is send to OMR. Font sizes etc. are set to "printer prints condition. Exceed characters are disregarded when set over 20 characters “pString”.	

5.8.16. SkDv_ReqSetContFeedPrint

Function	Set the print characters for continuous reading	
Prototype	SkDvStatus SkDv_ReqSetContFeedPrint (SkDvHandle hSkDevice, SK_CONT_FEED_PRN_OPT* pPrnOpt)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pPrnOpt	Set the pointer of Printing option which conditions are set
Return	SKDV_STS_SUCCESS	Succesful
	Except Above	Failure (No device connected, or device is connecting)
Description	Printer setting(CMD:PR) command is send to OMR	

5.8.17. SkDv_ReqContFeedSheet

Function	Continuation reading is begin	
Prototype	SkDvStatus SkDv_ReqContFeedSheet(SkDvHandle hSkDevice, DWORD dwReqData, int iErrReject)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	dwReqData	Set the data outputs format. Set "SKDV_CONT_FEED_REQ_NORMAL"
	iErrReject	Set the direction error sheet happen Set "SKDV_DISABLE(0)": mains paper eject or "SKDV_ENABLE (1)": select paper eject(rejecter)
Return	SKDV_STS_SUCCESS	Succesful
	Except Above	Failure (No device connected, or device is connecting)

Description

"Sheet continuous reading [CMD:CF]" command is send to OMR, and start continuous reading.

"Worker thread" is generates in the buffer, and receive mark data automatically.

○dwReqData

Constant	Value	Description
SKDV_CONT_FEED_REQ_NORMAL	0x0000	Normal Reading
SKDV_CONT_FEED_REQ_SHEET_12INCHES_OVER	0x6000	Sheet Size over 12 inches (workable to use sheet size setting)
SKDV_CONT_FEED_REQ_SHEET_12INCHES_BELOW	0x2000	Use sheet size setting
SKDV_CONT_FEED_REQ_SHEET_SHORT	0x3000	Sheet Size shorter 7.4 inches (Only for SR-8000HYBRID)
SKDV_CONT_FEED_REQ_PE_CMD	0x0800	Use print & eject command (Only for SR-8000HYBRID)
SKDV_CONT_FEED_REQ_LATE_STOP	0x0400	Stop in case print & eject is NG (Only for SR-8000HYBRID)
SKDV_CONT_FEED_REQ_LATE_PRINT	0x0200	Print in case print & eject is NG (Only for SR-8000HYBRID)
SKDV_CONT_FEED_REQ_LATE_REJECT	0x0100	Reject paper in case print & eject is NG (Only for SR-8000HYBRID)

"Worker thread" is automatically close all reading is completed (included error stop).

"SkDv_IsContFeedRunning" is for confirm "Worker thread" operating condition.

Do not call OMR and communication command except "kDv_ReqContFeed_Cancel"."

The function which may be used while a worker thread operates is as follows.

Constant	Description
SkDv_ReqContFeed_Cancel	Request cancel Continuous reading
SkDv_IsContFeedRunning	Check operation of continuous reading
SkDv_IsContFeedDataFinished	Check Finish contunious reading
SkDv_ExistDoneData	Continuation reading un-acquiring.Data authentication
SkDv_GetContFeedCount	Get the reading status (the number of sheets) by continuous reading
SkDv_PrepareContData	Prepare the continuous reading data
SkDv_GetContDataNumber	Get the current data number of continuous reading
SkDv_GetContDataMarks	Get the mark data of continuous reading
SkDv_GetContDataWindows	Get the window data of continous reading
SkDv_GetContDataBarcodes	Get the barcode data of continuous reading
SkDv_GetContDataSheetResult	Get the sheet result of continuous reading
SkDv_GetContFeedResult	Get the result of continuous reading

5.8.18. SkDv_ReqContFeed_Cancel

Function	Request stop(Cancel)continuous reading	
Prototype	SkDvStatus SkDv_ReqContFeed_Cancel(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
Return	SKDV_STS_SUCCESS	Succesful
	Except Above	Failure (No device connected, or device is connecting)
Description	"Canceling operations [CMD:CA]" command is send to OMR, Paused continuous reading. Continuous reading may not complete even return operation. Operating conditions can be check by "SkDv_IsConuFeedRunning"	

5.8.19. SkDv_IsContFeedRunning

Function	Check operation of continuous reading	
Prototype	BOOL SkDv_IsContFeedRunning(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
Return	TRUE	Thread for continuous reading is operating
	FALSE	Thread for continuous reading is completed
Description	This command can confirm “worker thread” is operating under execute” SkDvStatus SkDv_ReqContFeedSheet”. “Worker thread” is not end immediately execute “SkDvStatus SkDv_ReqContFeed_Cancel”.	

5.8.20. SkDv_IsContFeedDataFinished

Function	Confirm continuous reading is finished.	
Prototype	BOOL SkDv_IsContFeedDataFinished(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
Return	TRUE	Thread for continuous reading is operating
	FALSE	Thread for continuous reading is completed
Description	This command can confirm “Worker thread” finished completely, also all data can be received by “SkDv_PrepareContData” .The worker thread currently performed with the "continuation read requests" function is completed, TRUE response is returned when worker thread completely finished by “SkDv_IsContFeedRunning” and and all the data is erased from the ” SkDv_ExistDoneData”.	

5.8.21. SkDv_ExistDoneData

Function	Confirm continuous reading data is stored.	
Prototype	BOOL SkDv_ExistDoneData(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
Return	TRUE	With Continuous reading data
	FALSE	Without Continuous reading data
Description	The continuation reading data which is stored by “worker thread” that executed by “SlDv_ReqContFeedsheet”is exists in this library. It may return False when continuation reading data has not been stored even worker thread active.	

5.8.22. SkDv_GetContFeedCount

Function	Get the current situation total number of sheets	
Prototype	SkDvStatus SkDv_GetContFeedCount(SkDvHandle hSkDevice, SK_CONT_FEED_COUNT* pContFeedCount)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	pContFeedCount	Set the pointer to store the current situation of number of sheets
Return	SKDV_STS_SUCCESS	Succesful
	Except above	Failure (No device connected, or device is connecting)
Description	Get the current situation total number of sheets read executed “SkDvReq_contFeedSheet”	

5.8.23. SkDv_PrepareContData

Function	Prepare to receive the data “worker thread” received	
Prototype	SkDvStatus SkDv_PrepareContData(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
Return	SKDV_STS_SUCCESS	Response returns reading one sheet in continuation reading
	Except above	Failure (No device connected, or device is connecting)
Description	Prepares to receive the data “Worker thread” received. Continuous reading data is not stored, it keeps stand by till receive the data, When continuation readingned data is not stored yet, it stands by within this function until a worker thread receives data.	

5.8.24. SkDv_GetContDataNumber

Function	The current data number (number of sheets) of continuation reading data is acquired.	
Prototype	int SkDv_GetContDataNumber(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
Return	0	The value after performing "SkDv_PrepareContData"
	Except above	The value "SkDv_PrepareContData". (total read number of sheets +1)
Description	A return value will be set to 0 to execute "SkDv_GetContDataNumber " after execute "SkDv_ReqContFeedSheet". Count is up each execute "SkDv_PrepareContData", and proceed "Total read number of sheets +1".	

5.8.25. SkDv_GetContDataMarks

Function	Get the mark data prepared by "Sk_DvPrepareContData"	
Prototype	BOOL SkDv_GetContDataMarks(SkDvHandle hSkDevice, int iFace, SK_DV_MARK_INFO *pMarkResult, char* pMarks, int* piBufSize)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	iFace	Set Front side or Back side
	pMarkResult	Set the address to store the mark data. Refer to "SK_DV_MARK_INFO".
	pMarks	Set the address of buffer to store Mark data. Mark data is stored between 0 – 16 by each bytes. Marked data is not stored when NULL is set.
	piBufSize	Set the buffer size to store the mark data. Store the mark d ata into buffer set by pMarks. The appropriate buffers size i s stored after piBufSize is set.
Return	TRUE	Succesful
	FALSE	Failure (No Data or result is not normal)
Description	Get the mark data prepared by "Sk_DvPrepareContData".Content are the same as a "SkDv_ReqGetMarks".	

5.8.26. SkDv_GetContDataBarcodes

Function	Get barcode data prepared by SkDv_PrepareContData	
Prototype	BOOL SkDv_GetContDataBarcodes(SkDvHandle hSkDevice, SK_DV_BARCODE_RESULT *pBarcodeResult, char* pBarcodes, int* piBufSize)	
Parameter	hSkDevice	Set Device controlling handle
	pBarcodeResult	Set the address of Structure to fold barcode infomation.
	pBarcodes	Set the address of buffer to fold barcode data. Barcode data is lined “barcode type”, “the number of byte by each barcode data”
	piBufSize	Set the byte size of buffer to fold barcode data. Folding data into the buffer appointed in pBarcodes up to appointed size. After execution, the needed buffer size will be stored.
Return	TRUE	Successful
	FALSE	Failure
Description	Get barcode data prepared by SkDv_PrepareContData. Barcode data is lined “barcode type”, “the number of byte by each barcode data” and separated with “Comma (,)” To execute the data, ”SkDv_GetContDataBarcodesCount” to get the number of Barcodes, and “SkDv_GetContDataBarcodesData” to get the each barcode data.	

5.8.27. SkDv_GetContDataBarcodesCount

Function	Get the number of Barcode prepared by SkDv_PrepareContData	
Prototype	BOOL SkDv_GetContDataBarcodesCount(SkDvHandle hSkDevice, int* piReadCount)	
Parameter	hSkDevice	Set Device controlling handle
	piReadCount	Set the address of Structure to fold the number of barcode i nformation.
Return	TRUE	Successful
	FALSE	Failure
Description	Get the number of Barcode prepared by SkDv_PrepareContData Same as ”pBarcodeResult->iCount” by SkDv_GetContDataBarcodes	

5.8.28. SkDv_GetContDataBarcodesData

Function	Get each barcode data prepared by SkDv_PrepareContData	
Prototype	BOOL SkDv_GetContDataBarcodesData(SkDvHandle hSkDevice, int iIndex, char* pcBcType, char* pszBarcode, int* piDataLen)	
Parameter	hSkDevice	Set Device controlling handle
	iIndex	Set the index number of Barcode from “1”
	pcBcType	Set the address of Structure to fold the kind of barcode set by “iIndex”
	pBarcode	Set the address of Structure to fold the barcode data set by “iIndex”
	piBufSize	Set the buffer size to fold the barcode data set by “iIndex” Folding data is into the buffer appointed in pBarcodes up to appointed size. After execution, the needed buffer size will be stored.
Return	TRUE	Successful
	FALSE	Failure
Description	Get each barcode data prepared by SkDv_PrepareContData. Each data are same as get by “SkDv_GetContDataBarcodes”	

5.8.29. SkDv_SaveContJpegImage

Function	Save prepared JPEG image data in the appointed file name	
Prototype	BOOL WINAPI SkDv_SaveContJpegImage(SkDvHandle hSkDevice, int iFace, char* pszFilename)	
Parameter	hSkDevice	Set the pointer to receive the Device controll handle
	iFace	Appoint face of JPEG image to save
	pszFilename	Specify the JPEG file name to save in full path
return	TRUE	Successful
	FALSE	Failure Parameter error or no data
Description	To save JPEG image prepared by the function Can’t be used to convert other format except JPEG, to cut out image or rotate image Note that resolution info is not included in the JPEG file	

5.8.30. SkDv_GetContDataSheetResult

Function	Get the mark data prepared by SkDv_PrepareContData	
Prototype	BOOL SkDv_GetContDataSheetResult(SkDvHandle hSkDevice, SK_SHEET_RESULT *pSheetResult)	
Parameter	hSkDevice	Set Device controlling handle
	pSheetResult	Set the address of Structure to fold sheet result
Return	TRUE	Successful
	FALSE	Failure
Description	Get the mark data	

5.8.31. SkDv_GetContFeedResult

Function	Get the continuous read result	
Prototype	BOOL　SkDv_GetContFeetResult(SkDvHandle hSkDevice, SK_CONT_FEED_RESULT *pContFeedResult)	
Parameter	hSkDevice	Set Device controlling handle
	pContFeedResult	Set the address of Structure to fold read result
Return	TRUE	Successful
	FALSE	Failure
Description	Get the continuous read result	

5.9. API function - Option

5.9.1. SkDv_GetBcrConf

Function	Get the reading condition of barcode stored into library. (Barcode reading for SR-3500 series)	
Prototype	SkDvStatus SkDv_GetBcrConf (SkDvHandle hSkDevice, SK_DV_OPT_BCR_CONF* pBcrConf)	
Parameter	hSkDevice	Set Device controlling handle
	pBcrConf	Set the address of Structure to fold reading condition of b arcode
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Get the reading condition of barcode stored into library. This condition is stored into "pBcrConf". Not connect with Device. Call "SkDv_ReqInit" or "SkDv_ReqGetBcrConf" to get the setting into device	

5.9.2. SkDv_SetBcrConf

Function	Set the barcode reading setting into library. (Barcode reading for SR-3500 series)	
Prototype	SkDvStatus SkDv_SetBcrConf (SkDvHandle hSkDevice, SK_DV_OPT_BCR_CONF* pBcrConf)	
Parameter	hSkDevice	Set Device controlling handle
	pBcrConf	Set the address of Structure to fold reading condition of b arcode
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	Device information store into library. Connect with device and store into library in case not get the device information. Check barcode setting set by "pBcrConf" and response error accordingly. If not have any error, store the barcode reading setting into library. This setting can be set into device to execute "SkDv_ReqSetBcrConf"	

5.9.3. SkDv_GetBcrConfEx

Function	To get condition of barcode reading saved in the library Barcode reading for SR-8000HYBRID	
Prototype	SkDvStatus　SkDv_GetBcrConfEx (SkDvHandle hSkDevice, SK_DV_OPT_BCR_CONF_EX* pBcrConfEx)	
Parameter	hSkDevice	Appoint variable pointer to get device handler opened
	pBcrConfEx	Appoint structure pointer to fold the condition of barcode reading
Return	SKDV_STS_SUCCESS	Successful
	Other than above	Failure
Description	The condition of barcode reading to get is the one held in this library Fold the condition of barcode reading in the libary into parameter pBcrConfEx No communication with device Call the function of SkDv_ReqInit or SkDv_ReqGetBcrConf in advance, if you need to get the condition set in the device like immefiately after starting-up	

5.9.4. SkDv_SetBcrConfEx

Function	Set the condition of barcode reading in the library Barcode reading for SR-8000HYBRID	
Prototype	SkDvStatus　SkDv_SetBcrConfEx (SkDvHandle hSkDevice, SK_DV_OPT_BCR_CONF_EX* pBcrConfEx)	
Parameter	hSkDevice	Appoint variable pointer to get device handler opened
	pBcrConfEx	Appoint structure pointer to fold the condition of barcode reading
Return	SKDV_STS_SUCCESS	Successful
	Other than above	Failure
Description	Communicate with device and fold the information into the library if it is not obtained Return error if value is incorrect after checking the condition of barcode reading designated in parameter pBcrConfEx to the device information Fold the condition of barcode reading in the library if there is no error. The condition of barcode reading can be set in the device by communication by executing reading request or function of SkDv_ReqSetBcrConf	

5.9.5. SkDv_ReqGetBcrConf

Function	Request connect SR11000 to get the reading condition of Barcode	
Prototype	SkDvStatus　SkDv_ReqGetBcrConf (SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	Connect to SR11000 to get the reading condition of Barcode, and store it into library. Refer to command reference.　 Command [BC]	

5.9.6. SkDv_ReqSetBcrConf

Function	Request connect SR11000 to set the reading condition of Barcode	
Prototype	SkDvStatus　SkDv_ReqSetBcrConf (SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	Set reading condition of Barcode to device which stored in the library Command line is same as　「SkDv_ReqGetBcrConf」	

5.9.7. SkDv_ReqGetBcrDataCount

Function	Get the number of barcode read	
Prototype	SkDvStatus SkDv_ReqGetBcrDataCount(SkDvHandle hSkDevice, int* piReadCount)	
Parameter	hSkDevice	Set Device controlling handle
	piReadCount	Set the address to store the barcode data
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Use “request command barcode” “BD” to get the number of Barcode read correctly. It can not get the number of barcodes when continuously reading	

5.9.8. SkDv_ReqGetBcrData

Function	Get the (specified) barcode data	
Prototype	SkDvStatus SkDv_ReqGetBcrData(SkDvHandle hSkDevice, int iIndex, char* pcBcType, char* pszBarcode, int* piDataLen)	
Parameter	hSkDevice	Set Device controlling handle
	iIndex	Set the index number of Barcode
	pcBcType	Set valiable number for store the kinds of barcode type set by “iIndex”
	pszBarcode	Set the address to store the barcode data set by “iIndex”
	piDataLen	Set the size of buffer to store the barcode data set by “iIndex” Data is stored in the buffer set “pszBarcode”till the buffer size After execution, needed buffer size is stored.
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Use “request command barcode” “BD” to get the number of Barcode read correctly. It can not get the number of barcodes when continuously reading	

5.9.9. SkDv_GetPrinterConf

Function	Get printer setting from Library	
Prototype	SkDvStatus SkDv_GetPrinterConf(SkDvHandle hSkDevice, SK_DV_OPT_PRN_CONF* pPrnConf)	
Parameter	hSkDevice	Set Device controlling handle
	pPrnConf	Set the pointer to store the printer setting
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Printing condition is storing in the Library. These printing condition is stored “pPrnConf” Call “SkDv_ReqInit” or “SkDv_ReqGetPrinterConf” in case want to get these condition in device.	

5.9.10. SkDv_SetPrinterConf

Function	Set printer setting into library	
Prototype	SkDvStatus SkDv_SetPrinterConf(SkDvHandle hSkDevice, SK_DV_OPT_PRN_CONF* pPrnConf)	
Parameter	hSkDevice	Set Device controlling handle
	pPrnConf	Set the pointer stored printer setting
Return	SKDV_STS_SUCCESS	Successful
	Except above	Failure
Description	Set the printing condition into middleware.	

5.9.11. SkDv_ReqPrintString

Function	Set the print string into OMR device	
Prototype	SkDvStatus SkDv_ReqPrintString(SkDvHandle hSkDevice, char* pString)	
Parameter	hSkDevice	Set Device controlling handle
	pString	Set the address to want to print character
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	Set character to print	

5.9.12. SkDv_ReqGetPrinterConf

Function	Get printer setting	
Prototype	SkDvStatus SkDv_ReqGetPrinterConf(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	Get printing condition	

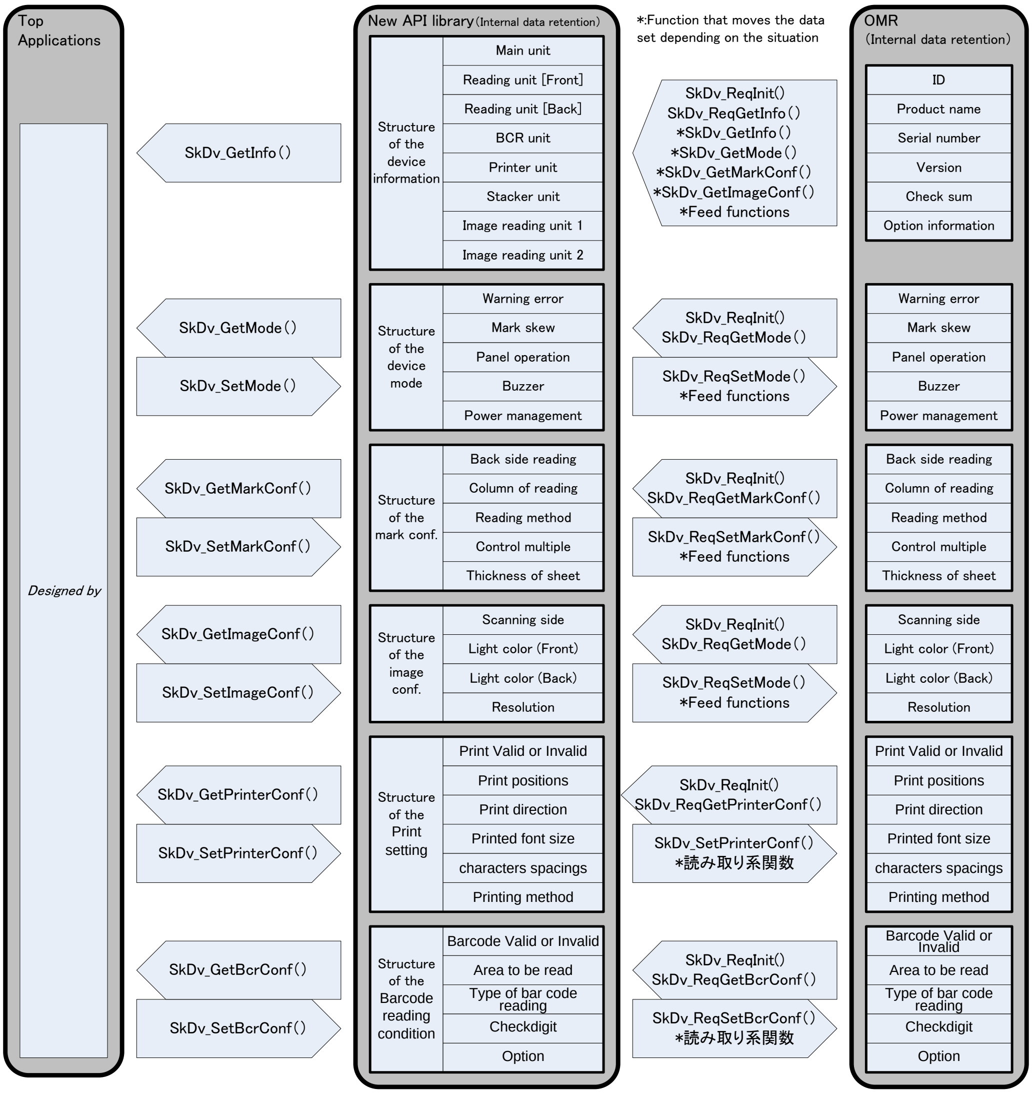
5.9.13. SkDv_ReqSetPrinterConf

Function	Set printer setting	
Prototype	SkDvStatus SkDv_ReqSetPrinterConf(SkDvHandle hSkDevice)	
Parameter	hSkDevice	Set Device controlling handle
Return	SKDV_STS_SUCCESS	Successful
	except above	Failure
Description	Set printer setting.	

5.10. Setting and Data

5.10.1. Data Transfer by executed API

Structure of the Device information / Structure of device Mode / Structure of the mark conf / Structure of the image conf / Structure of the Print setting / Structure of the Barcode setting would be retented in New API library / OMR, and Top applications depends on the structure of the applications. Each data transferred by the executed API

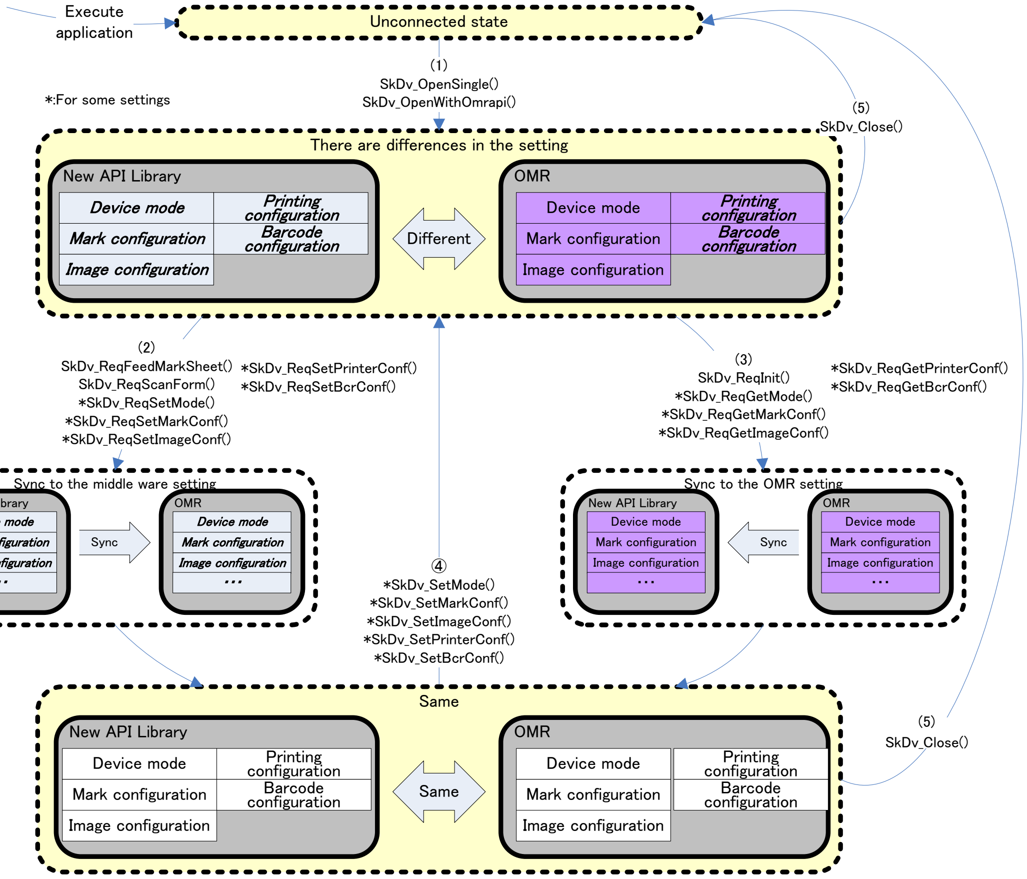


Call "SkDv_GetInfo", then New API would receive the device information by OMR and response to the Top applications.
Also, New API can set Device mode setting or Image conf setting automatically by Feed functions
(ex: SkDv_Req Set Mode) after set the setting by "SkDv_SetMode".

Note: If New API uses with Old API, New API may be change the setting old API set. Please refer 5.6.2 for the details

5.10.2. Change Settingdata by executed API

5 kinds of setting data (device Mode / mark conf / image conf / print conf /barcode conf) would be keep in New API library / OMR buffer both. There's different in the setting executed functions.



	Executed API	Description

1	SkDv_OpenSingle()
	SkDv_OpenWithOmrapi()
		When use these functions, the contents of API library has the default, and OMR has the setting retented (Difference)

2	SkDv_ReqFeedMarkSheet()
	SkDv_ReqScanForm()
	*SkDv_ReqSetMode()
	*SkDv_ReqSetMarkConf()
	*SkDv_ReqSetImageConf()
	*SkDv_ReqSetPrinterConf()
	*SkDv_ReqSetBcrConf()
		When execute feed functions, Setting keep in New API library would send to the OMR. The setting of OMR and API are corresponded.
		Additionally, setting keeps in New API library forward to the OMR to be execute enhance functions (ex:"SkDv_ReqSetMode" etc)

3	SkDv_ReqInit()
	*SkDv_ReqGetMode()
	*SkDv_ReqGetMarkConf()
	*SkDv_ReqGetImageConf()
	*SkDv_ReqGetPrinterConf()
	*SkDv_ReqGetBcrConf()
		When execute "SkDv_ReqInit" functions, Setting keeps in OMR would send to the New API Library. Then, the setting of OMR and API are corresponded.
		Additionally, setting keeps in OMR send to the New API Library to be execute enhance functions(ex: "SkDv_ReqGetMode" etc)

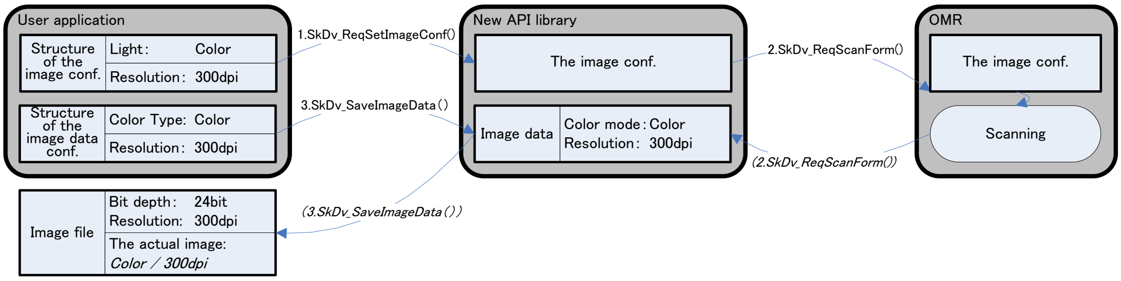
4	*SkDv_SetMode()
	*SkDv_SetMarkConf()
	*SkDv_SetImageConf()
	*SkDv_SetImageConfEx()
	*SkDv_SetPrinterConf()
	*SkDv_SetBcrConf()
	*SkDv_SetBcrConfEx()
		Contents between OMR and New API would be separated to call functions (ex: "SkDv_SetMode" etc) to change the contents of New API.

5	SkDv_Close()	When execute this functions, the contents of New API would be cleared.

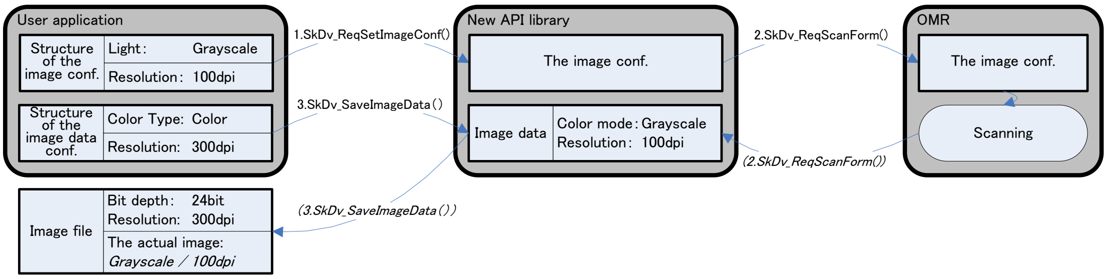
*: For some settings.

5.10.3. Image setting and saved Image file

This chapter image unit mounting machine is intended.
Image data would be stored in the new API library. It should be set same DPI and color setting front side image and back side image. If set Color / 300dpi image to get, this image can be changed to low resolution DPI, Grayscale, Black / White while use by "SkDv_SaveImageData".



But set Grayscale /low DPI image to get, actual image is low DPI image. Then resolution is low level even set "300dpi" by "SkDv_SaveImageData"



5.11. Examples

5.11.1. In case use NEW API

1. Enable the communication with OMR by Opening operating Function (SkDv_OpenSingle).

2.For Initialization ("SkDv_ReqInit"), various setting (Operation mode / Setting for Mark reading / Setting for Image reading) are get from OMR, and stored into API Library.
The setting of OMR & Library is to be corresponded to each other.

3. In case of changing the Operation mode, the values are to be stored into Structure of Operation mode, and handed to Setting Function of Operation mode ("SkDv_SetMode").
At this time, Operation mode is stored into Library.

4. In case of change the condition of Mark reading, the values are to be stored into structure of Mark setting condition, and handed to setting function for setting of mark condition ("SkDv_SetMarkConf"). At this time, Mark condition is stored into Library.

5. In case of change the Image reading condition, the values are to be stored into structure of setting of Image condition, and handed to setting for Image condition ("SkDv_SetImageConf").
At this time, Image condition is stored into Library.

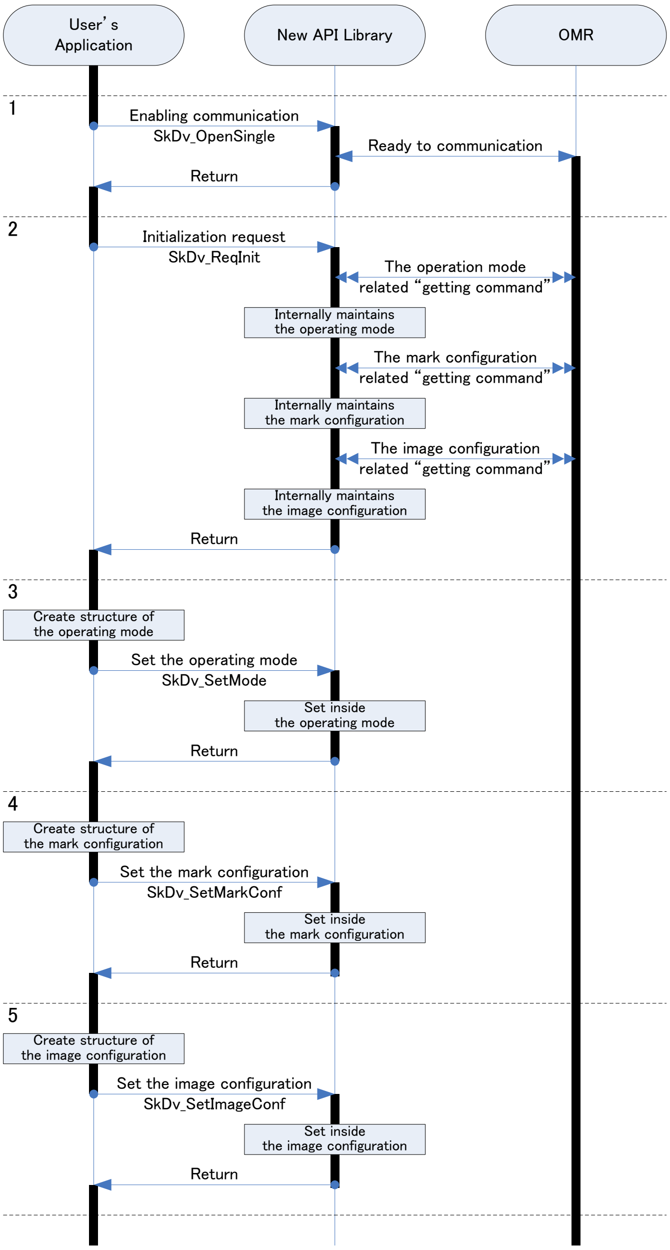

User's Application
New API Library
OMR
1
Enabling communication
SkDv_OpenSingle
Ready to communication
Return
2
Initialization request
SkDv_ReqInit
The operation mode related "getting command"
Internally maintains the operating mode
The mark configuration related "getting command"
Internally maintains the mark configuration
The image configuration related "getting command"
Internally maintains the image configuration
Return
3
Create structure of the operating mode
Set the operating mode
SkDv_SetMode
Set inside the operating mode
Return
4
Create structure of the mark configuration
Set the mark configuration
SkDv_SetMarkConf
Set inside the mark configuration
Return
5
Create structure of the image configuration
Set the image configuration
SkDv_SetImageConf
Set inside the image configuration
Return

Fig. **5** New API library control(1)

6. Calling the Demand Function ("SkDv_ReqscanForm") for Reading the sheets, Library operates the setting by each command in case the Operation mode / Mark condition / Image condition are changed.
Then, Sheets reading is operated [NS] and the Image data is to be stored into Library.

7. For get the Mark data, prepare the buffer to receive Mark data and the buffer to store Mark data, and call ("SkDv_ReqGetMarkData") to get Mark data. Library will respond the data given by mark density [MD].

8. For save Image into the file, set the value into Image data condition / Structure of Image file condition, and call ("SkDv_SaveImageData") for save Image data file.

(After the second sheet, the procedure will be repeated from "6" referred on the right chart.)
(Change operation setting can be done from "3" referred on the right chart.)

9. To finish Controll OMR, Call the Function("SkDv_Close") to Finish.

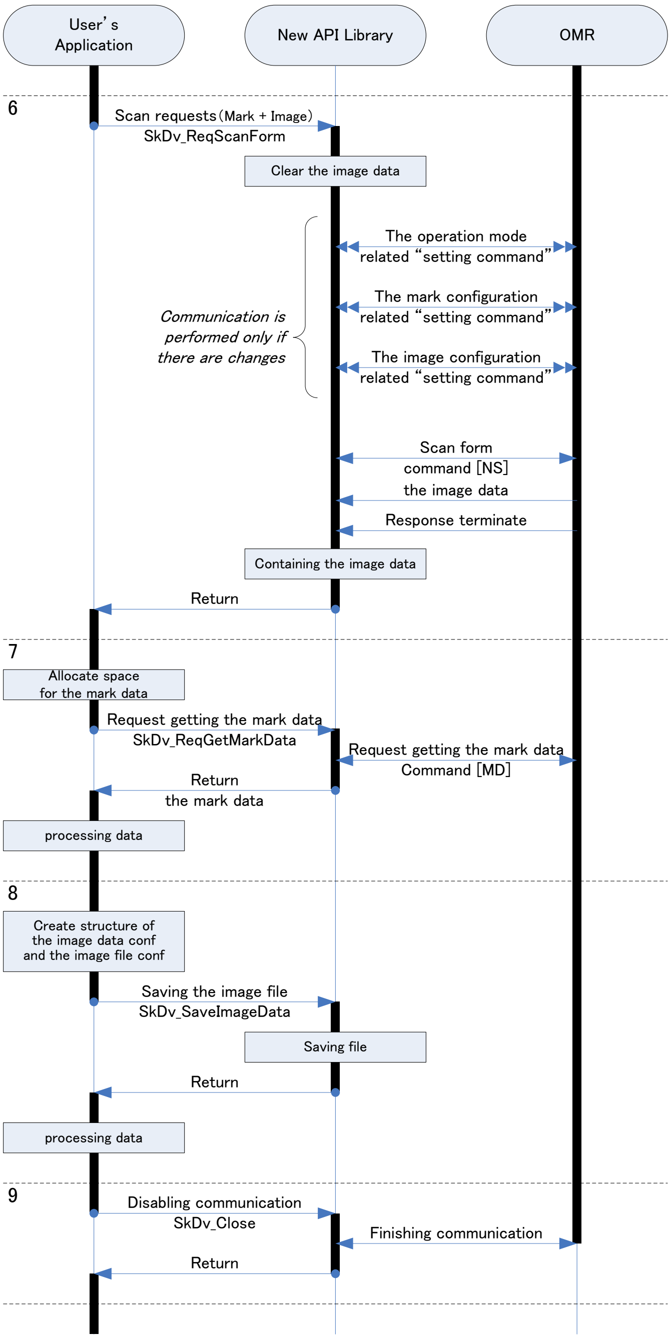

User' s Application
New API Library
OMR
6
Scan requests (Mark + Image)
SkDv_ReqScanForm
Clear the image data
Communication is performed only if there are changes
The operation mode related "setting command"
The mark configuration related "setting command"
The image configuration related "setting command"
Scan form command [NS]
the image data
Response terminate
Containing the image data
Return
7
Allocate space for the mark data
Request getting the mark data
SkDv_ReqGetMarkData
Request getting the mark data Command [MD]
Return
the mark data
processing data
8
Create structure of the image data conf and the image file conf
Saving the image file
SkDv_SaveImageData
Saving file
Return
processing data
9
Disabling communication
SkDv_Close
Finishing communication
Return

Fig. 6 New API library control(2)

5.11.2. Save Image Data by other threads

This chapter image unit mounting machine is intended.
New API library has additional function to save the image file (jpg, bmp…) by several threads due to improve the capability about DLL Library. Hereinafter is explained how to use it and how to work of it.
※It's necessary to use multi threads or callback function while use this function.

1.Call "SkDv_ReqReadingForm" to Reading Requests A), even capture image or not capture image.
Captured Image Data (Front and Back) stored in the buffers.

2. Call "SkDv_SaveImageDataThread" to be saved the image data into another threads.
"SkDv_SaveImageDataThread" has functions to transfer buffers (stored image) for another thread, and make new additional image buffers to fold for new image data.
After create new image buffers, "SkDv_SaveImageDataThread" create new threads and handle the contents of buffers to them.
At this time, Image data is still in buffer and not be saved yet.

3. Captured Image data is proceeding saving in another threads, Then it can be call "SkDv_ReqReadingForm" for next sheet. API received the callback when finish saving the image data by another threads.
"SkDv_SaveImageDataThread" still keep standby till finish "another threads"

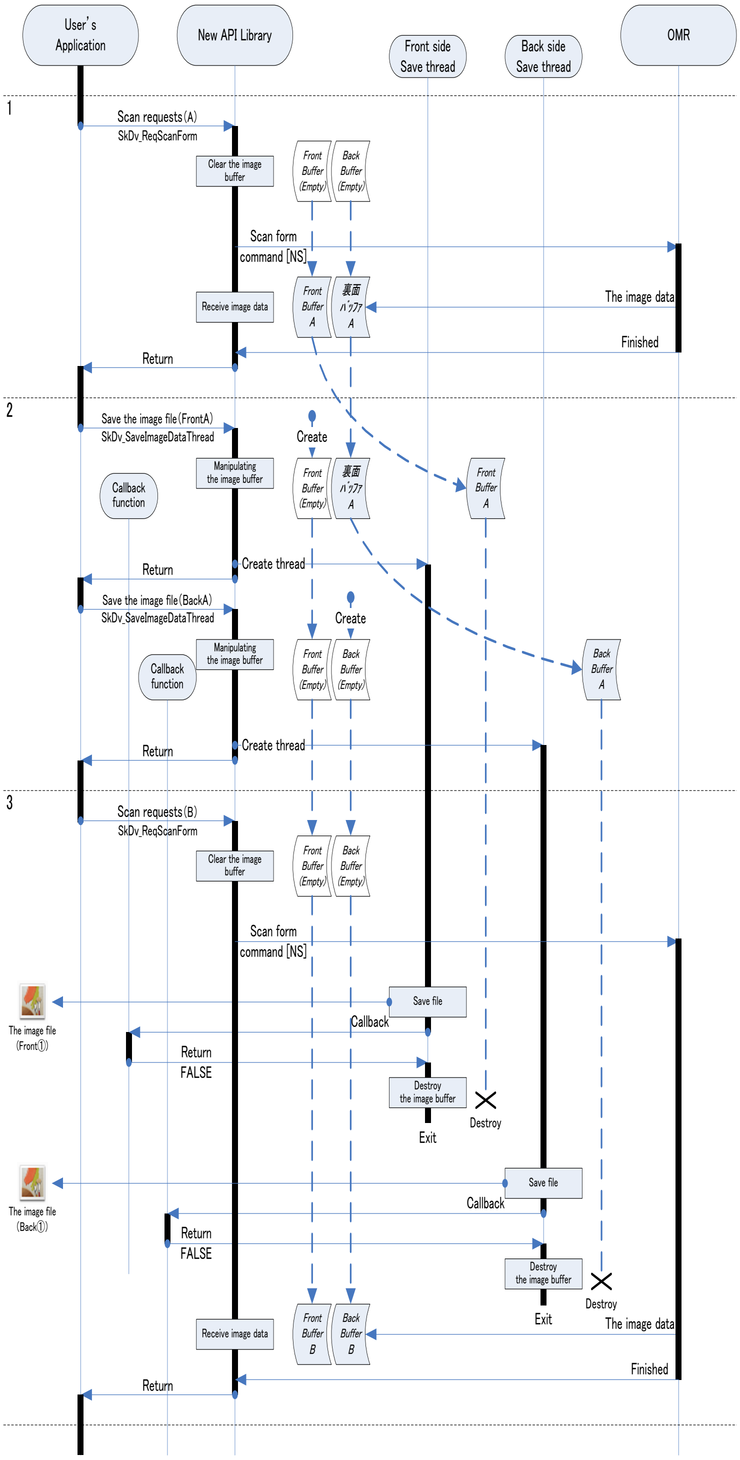


Fig. 7 Save Image file by other Threads

5.11.3. Read OMR form read without check the number of Timing Marks

This chapter is intended the products with continuous reading function.

1. Change into the state in which OMR and communication are possible with a startup function.

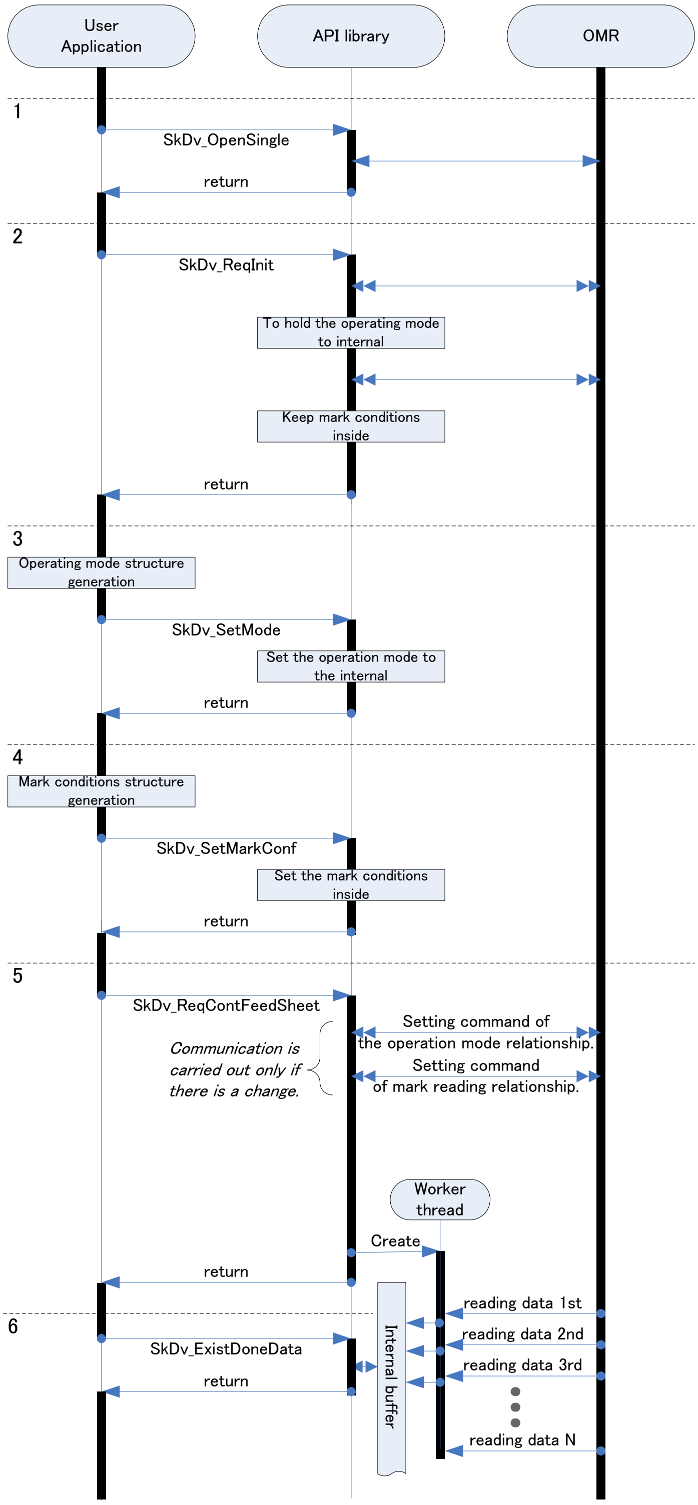
2. By the initialization requested function, acquisition of a mark reading setup, various mode of operation / various, various setup which are set as OMR is performed, and the contents are stored in a library.
This will be in the status where each settings are in agreement in OMR and a library.

3. In changing operation modes, each value is stored in an operation modes structure, and it passes an operation modes setting function by the argument. At this time, operation modes is stored in a library.

4. In changing conditions of mark-sensing, each value is stored in a mark condition structure, and it passes a marked setting function by an argument. At this time, mark condition are stored in a library.

5. In order to read a list, call a continuation read requests function.
A library sets up with each command, when operation modes / mark condition are changed. Thereby, a setting of a library and OMR is in agreement.
Then, a list is read with a continuation reading command and data is stored in an internals buffers.

6. In order to use data, it is necessary to perform a data present check function and to check that data exists.


User Application
API library
OMR
1
SkDv_OpenSingle
return
2
SkDv_ReqInit
To hold the operating mode to internal
Keep mark conditions inside
return
3
Operating mode structure generation
SkDv_SetMode
Set the operation mode to the internal
return
4
Mark conditions structure generation
SkDv_SetMarkConf
Set the mark conditions inside
return
5
SkDv_ReqContFeedSheet
Communication is carried out only if there is a change.
Setting command of the operation mode relationship.
Setting command of mark reading relationship.
Worker thread
Create
return
6
SkDv_ExistDoneData
return
Internal buffer
reading data 1st
reading data 2nd
reading data 3rd
reading data N

7. In order to obtain marked data, a preparing function for data acquisitions is performed.If a preparing function for data acquisitions is performed, data is stored in a buffer in an order from the first read list.

8. The buffer which stores the mark information structure object and marked data for receiving marked data is prepared, and it obtains with a continuation reading marked data get function. A library returns the data received with the marked concentrations data requests commands.
The received marked data is processed.

(6-8 is repeated after the 2nd sheet)

9. Perform a continuation reading result get functions for the check of the completion of continuation conveyance.

10. Call an end handling function for ending control of OMR, and end.

(For changing a setting, it carries out from 3)

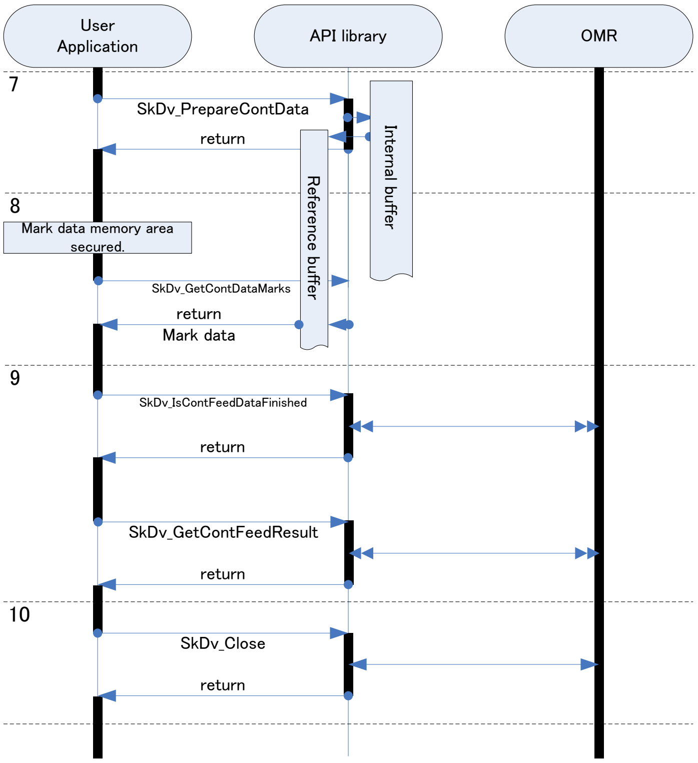


Fig. 8 Flow of processing which reads form, without checking TM number

5.11.4. Read OMR form(fixed the number of timing marks) without ID window

This chapter is intended the products with continuous reading function .

1. “SkDv_OpenSingle” to start up.

2. “SkDv_ReqInit” to request to get mark reading setup / image reading setup / operation mode setup… the contents are stored in a library.
This will to be corresponded with OMR and API library.

3. Change operation modes, “SkDv_SetMode” to change the setting and stored into API library.

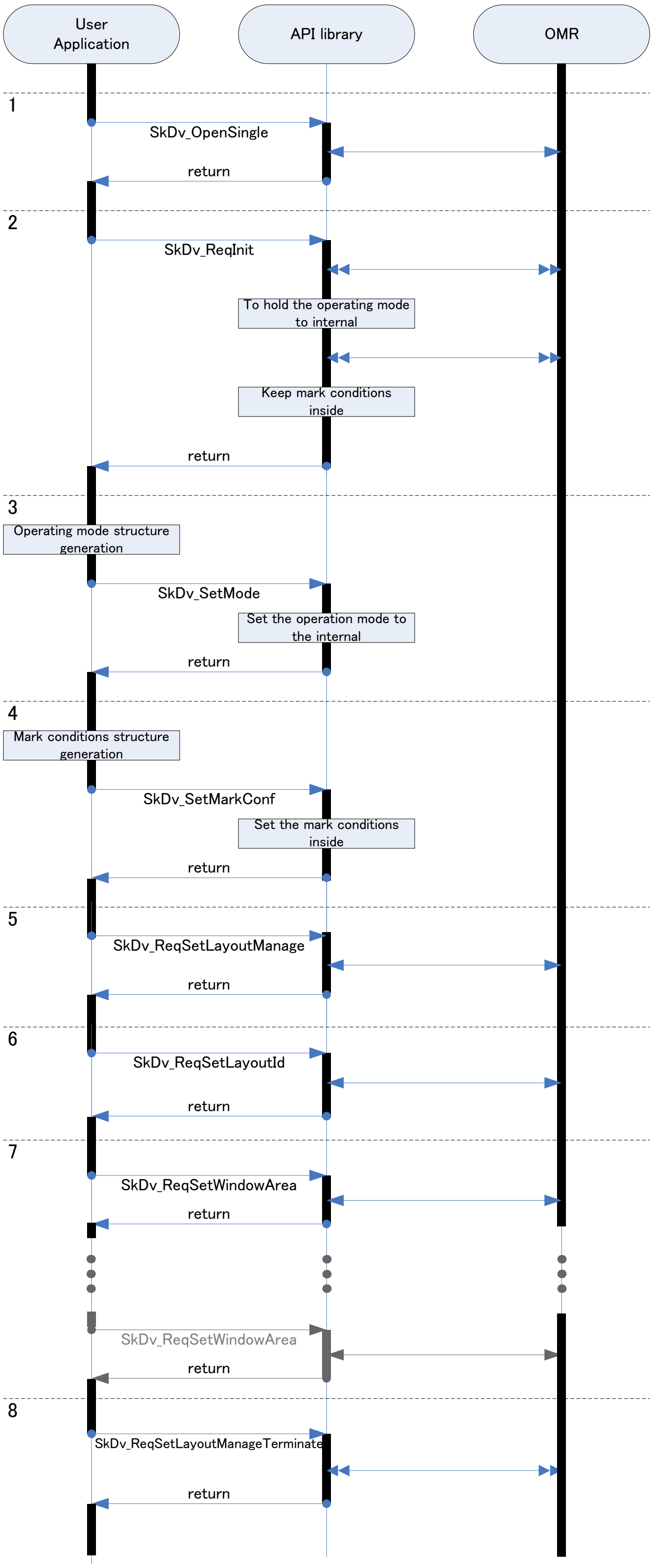
4. Change Mark read setting, “SkDv_SetMarkConf” to change the setting and stored into API library.

5. Setup window, “SkDv_ReqSetLayoutManage” to set IDsetting which does not use ID.

6. It’s necessary to “SkDv_ReqSetLayoutId” before to execute setup windows areas data. Not set this setting in case ID not use. ID setting is disregarded.
The number of timing marks is fixed this setting.

7. Setup a windows area. “SkDv_ReqSetWindowArea” to setup window setting. set up with identity data set function run immediately before.Only the number of the windows to be used is set up.

8. When ending a window setting,“SkDv_ReqSetLayoutManageTerminate” function.


User Application
API library
OMR
1
SkDv_OpenSingle
return
2
SkDv_ReqInit
To hold the operating mode to internal
Keep mark conditions inside
return
3
Operating mode structure generation
SkDv_SetMode
Set the operation mode to the internal
return
4
Mark conditions structure generation
SkDv_SetMarkConf
Set the mark conditions inside
return
5
SkDv_ReqSetLayoutManage
return
6
SkDv_ReqSetLayoutId
return
7
SkDv_ReqSetWindowArea
return
SkDv_ReqSetWindowArea
return
8
SkDv_ReqSetLayoutManageTerminate
return

9. “SkDv_ReqContFeedSheet” to read the form.
Linrary change the seting when operation mode / mark setting are changed. Then, the setting between OMR and API library are corresponced.
Data is stored in internals buffers.

10. In order to use data, it is necessary to use”SkDv_ExistDoneData” to confirm data exists.

11. In order to get mark data, use “SkDv_PrepareContData” prepare get mark data .

12. Use “SkDv_GetcontDataMarks” to get mark data

(10-12 repeat after the 2nd sheet)

13. “SkDv_IsContFeedDataFinished” to confirm feed finished.

14. Call “SkDv_Close” to be finish.
(To change setting, proceed from 3)
(To change window set, proceed out from 5.).

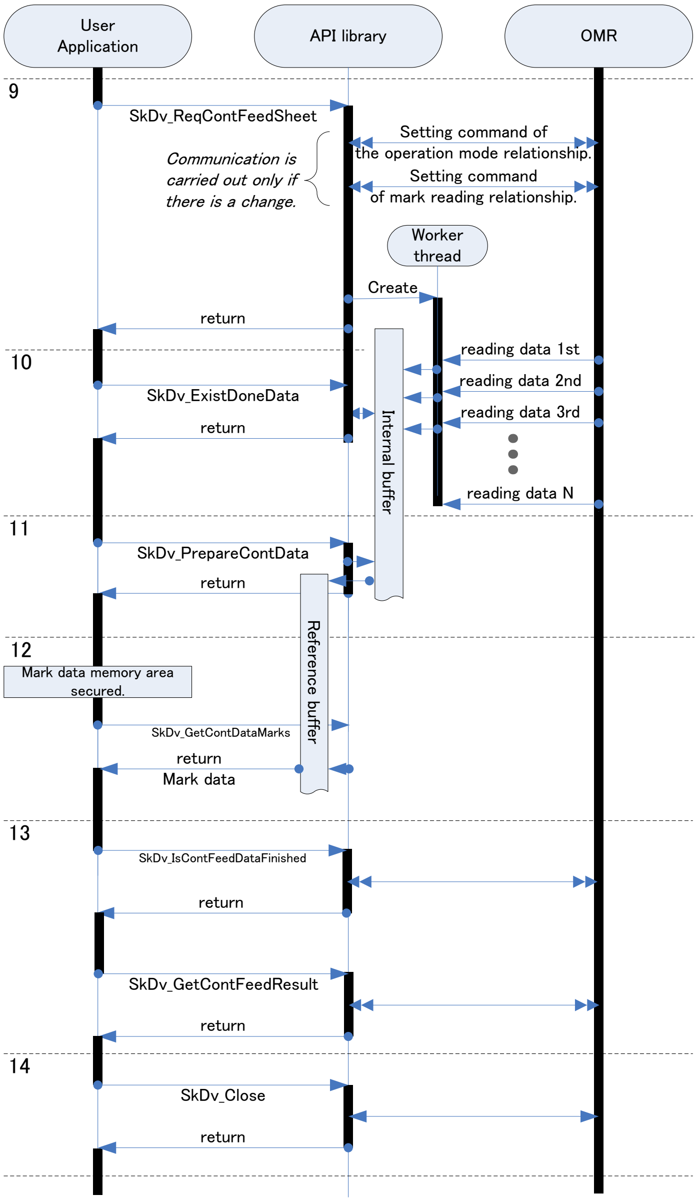
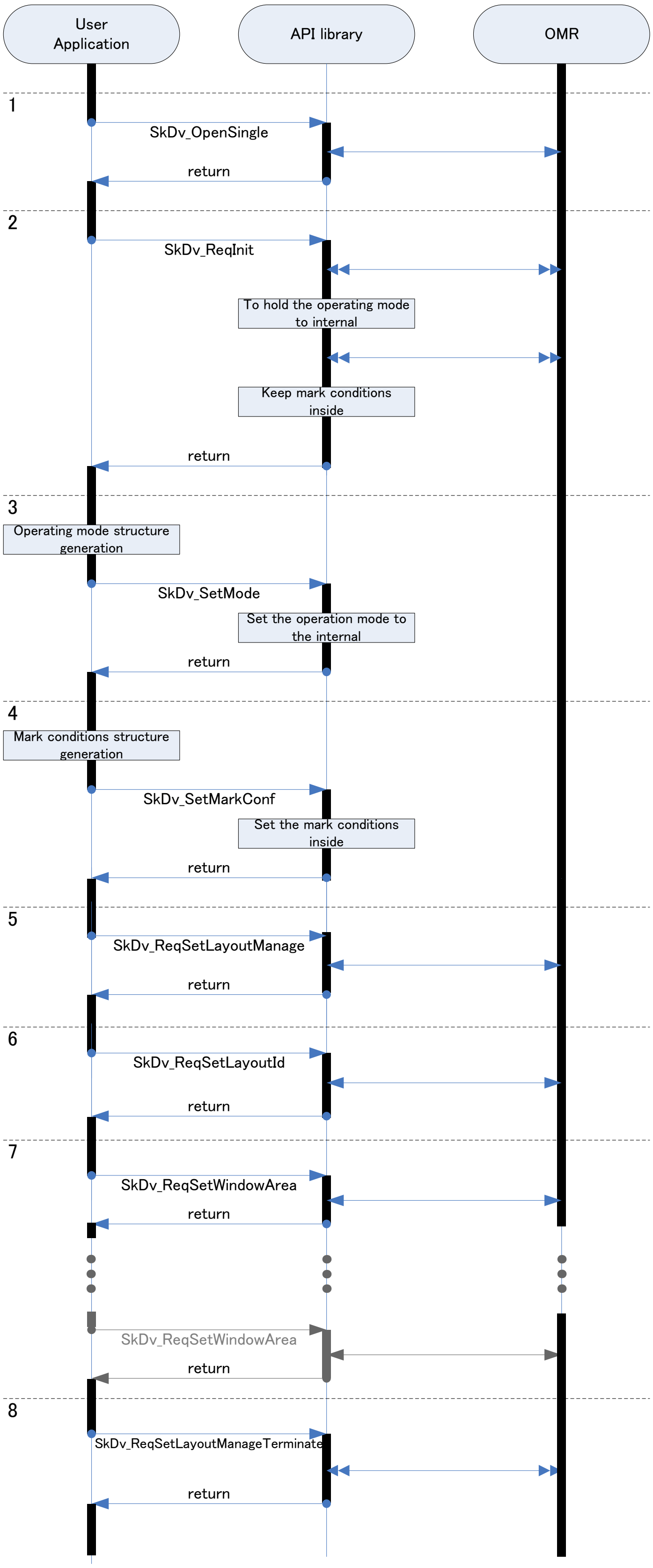


Fig. 9 Flow chart in case read one form(Fixed timing mark)without ID window

SEKONIC CORPORATION

5.11.5. Read several kinds of OMR forms(fixed timing marks) with ID Window

This chapter is intended the products with continuous reading function .


User Application
API library
OMR
1
SkDv_OpenSingle
return
2
SkDv_ReqInit
To hold the operating mode to internal
Keep mark conditions inside
return
3
Operating mode structure generation
SkDv_SetMode
Set the operation mode to the internal
return
4
Mark conditions structure generation
SkDv_SetMarkConf
Set the mark conditions inside
return
5
SkDv_ReqSetLayoutManage
return
6
SkDv_ReqSetLayoutId
return
7
SkDv_ReqSetWindowArea
return
SkDv_ReqSetWindowArea
return
8
SkDv_ReqSetLayoutManageTerminate
return

1. “SkDv_OpenSingle” to startup.

2. “SkDv_ReqInit” to request to get mark reading setup / image reading setup / operation mode setup… the contents are stored in a library.
This will to be corresponded with OMR and API library.

3. Change operation modes, “SkDv_SetMode” to change the setting and stored into API library.

4. Change Mark read setting, “SkDv_SetMarkConf” to change the setting and stored into API library.

5. To Setup window, “SkDv_ReqSetLayoutManage” to set IDsetting.

6. It’s necessary to “SkDv_ReqSetLayoutId” before execute setup windows areas data. Need to set the ID number use kinds of forms.
The number of timing marks is fixed this setting

7. Setup a windows area. “SkDv_ReqSetWindowArea” to setup window setting. set up with identity data set function run immediately before.Only the number of the windows to be used is set up.

8. When ending a window setting,“SkDv_ReqSetLayoutManageTerminate” function. (6 and 7 repeat to setup two or more ID, and, finally 8 is performed)

9. “SkDv_ReqContFeedSheet” to read the form.
Linrary change the seting when operation mode / mark setting are changed. Then, the setting between OMR and API library are corresponced. Data is stored in internals buffers.

10. In order to use data, it is necessary to use”SkDv_ExistDoneData” to confirm data exists.

11. In order to get mark data, use “SkDv_PrepareContData” prepare get mark data .

12. Use “SkDv_GetcontDataMarks” to get mark data

(10-12 repeat after the 2nd sheet)

13. “SkDv_IsContFeedDataFinished” to confirm feed finished.

14. Call “SkDv_Close” to be finish.
(To change setting, proceed from 3)
(To change window set, proceed out from 5.).　window are reset, only the number of required windows is repeated.

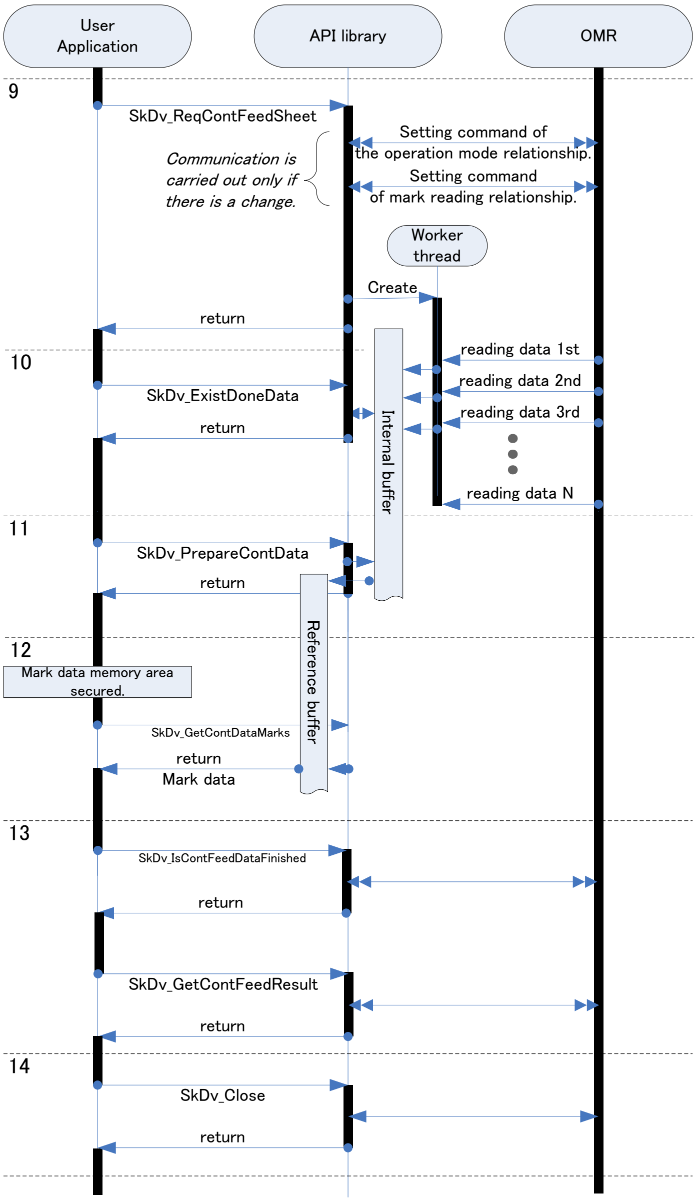
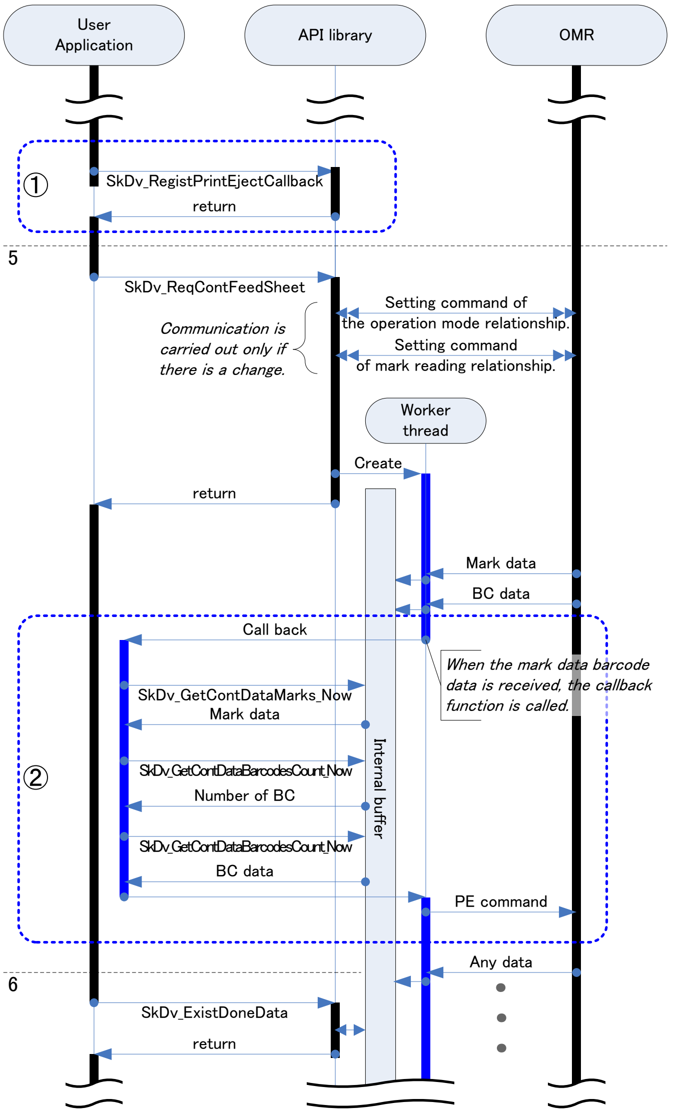


Fig. 10 Flow of processing which reads form which is several sorts as which TM number was decided with identifier window

5.11.6. Print & Eject during continuous reading (Only for SR-8000HYBRID)


User Application
API library
OMR
①
SkDv_RegistPrintEjectCallback
return
5
SkDv_ReqContFeedSheet
Setting command of the operation mode relationship.
Setting command of mark reading relationship.
Communication is carried out only if there is a change.
Worker thread
Create
return
Mark data
BC data
Call back
When the mark data barcode data is received, the callback function is called.
②
SkDv_GetContDataMarks_Now
Mark data
SkDv_GetContDataBarcodesCount_Now
Number of BC
SkDv_GetContDataBarcodesCount_Now
BC data
Internal buffer
PE command
Any data
6
SkDv_ExistDoneData
return

Refer to “5.11.3 Read OMR form read without check the number of Timing Marks” for 1 to 4

① Register call back function to print & eject

5. Call the function of continuous reading to read forms. In that case, use flag to use direction of print & eject.

Set library by each command in case operation mode/mark condition is altered. By this, setting of library and OMR is to be matched.

After that, read forms by command of continuous reading, and fold data into internal buffer.

Call back function is called if mark data and barcode data is received.

Get data of reading forms by specific function of acquiring data in the call back function.

Set the operation in the parameter of print & eject direction, and finish call back function.

By this, API library directs OMR to print & eject.

6 and followings, see the previous section

5.11.7. Print & Eject during continuous reading (Only for SR-8000HYBRID)

This is used for single thread program such as VB6.0 of not supporting muti thread.

Direction of print & eject is made in call back function. If calling this function is delayed, time to process in call-back function shortens, so it is necessary to call the function of SkDv_PollingPrintEject frequently. And, it is also necessary to proceed read data, so need to design to call function of SkDv_PollingPrintEject frequently.

Refer to “5.11.3 Read OMR form read without check the number of Timing Marks” for 1 to 4

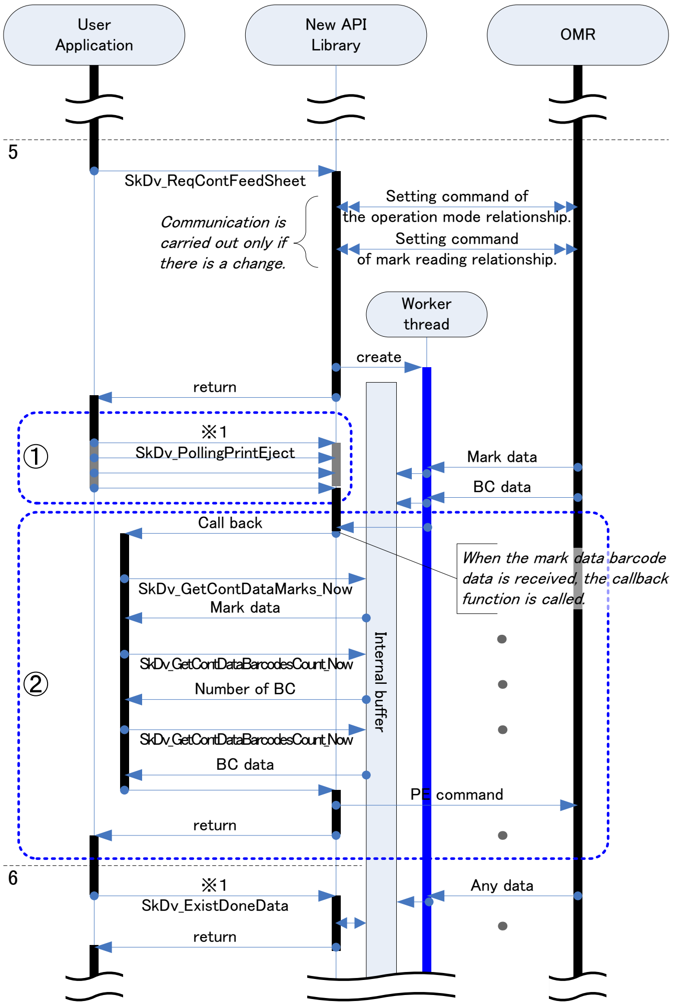
① Call back function is called if mark data and barcode data is received when the function of SkDv_PollingPrintEject is called to print & eject.

② Get data of reading forms by specific function of acquiring data in the call back function.

Set the operation in the parameter of print & egect direction, and finish call back function.

By this, API library directs OMR to print & eject.

*1:Need to design carefully considering that functions of SkDv_PollingPrintEject, SkDv_ExitstDoneData and acquiring continuous reading data are not processed alternately.


User Application
New API Library
OMR
5
SkDv_ReqContFeedSheet
Communication is carried out only if there is a change.
Setting command of the operation mode relationship.
Setting command of mark reading relationship.
Worker thread
create
return
①
※1
SkDv_PollingPrintEject
Mark data
BC data
Call back
When the mark data barcode data is received, the callback function is called.
SkDv_GetContDataMarks_Now
Mark data
②
SkDv_GetContDataBarcodesCount_Now
Number of BC
SkDv_GetContDataBarcodesCount_Now
BC data
Internal buffer
PE command
return
6
※1
SkDv_ExistDoneData
Any data
return

5.12. Printer setting

5.12.1. Procedure start printing

Printer or stacker mounting machine has printer function. Need to execute several API's same time to work .

① In case of set printing condition from the device, need to call “SkDv_ReqGetPrinterConf” and “SkDv_GetPrinterConf” to get printing condition.

② Set printing condition to “SK_DV_OPT_PRN_CONF”.

③ Set printing condition to the device by “SkDv_SetPrinterConf” and “SkDv_ReqSetPrinterConf” .

④ ～SR-6500 mode (Each sheet data transport to computer)～

After set printing characters call “SkDv_ReqPrintString” to set the setting to the device.

～SR-11000 mode(continuous feed to computer) ～

After set fixed characters, and call “SkDv_ReqSetPirntFixString” to set fixed characters into the device.

After set format of sequence number into “SK_CONT_FEED_PRN_OPT”, and call “SkDv_ReqSetContFeedPrint” to set sequence number into the device.
(Fixed character and sequence number are able to set independently.)

⑤ Printing start after paper feed order

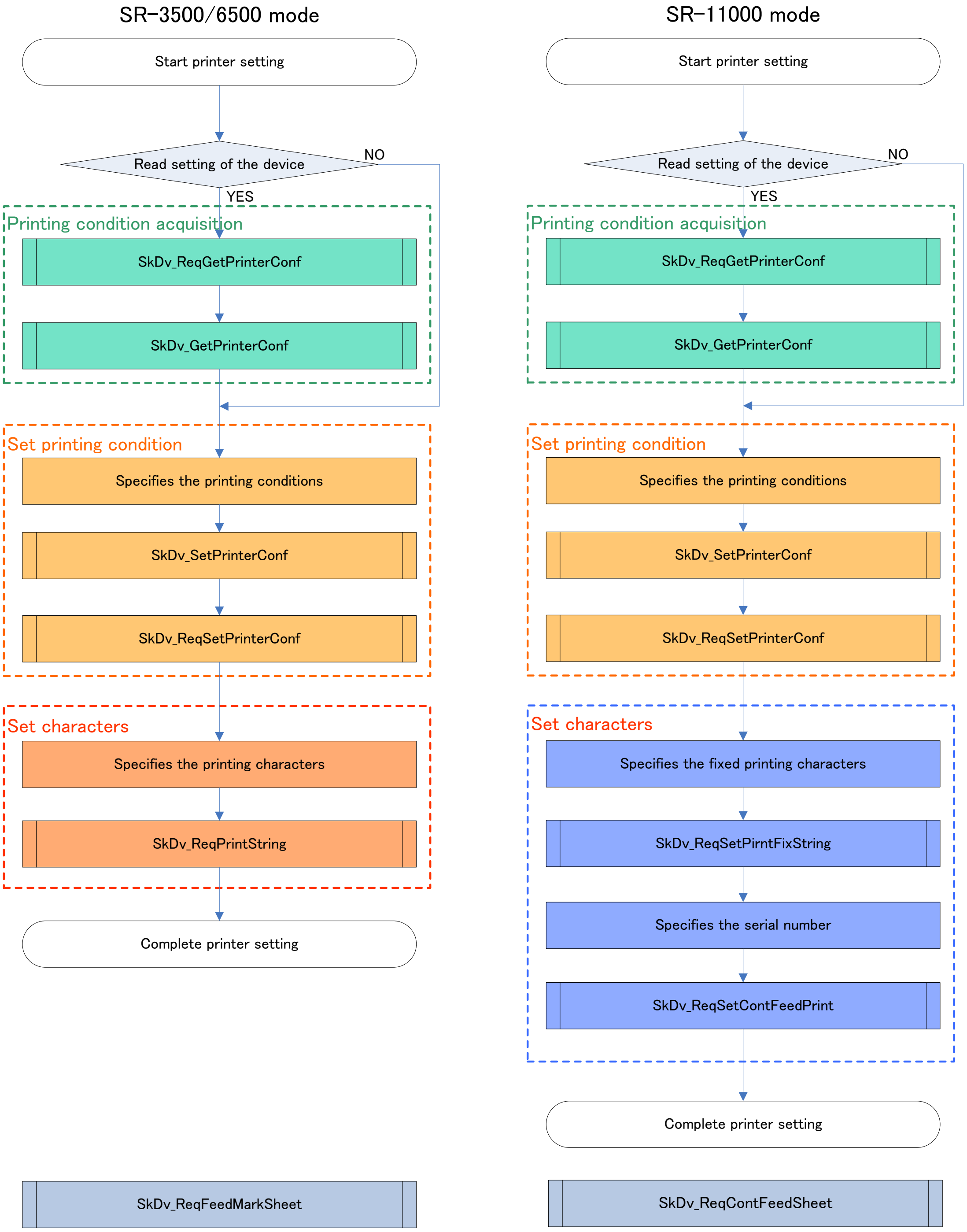
～SR-6500 mode～

“SkDv_ReqFeedMarkSheet”

～SR-11000 mode～

“SkDv_ReqContFeedSheet”

5.12.2. Flow diagram for printer setting



When you perform a read request function, it will be printed at the time of document transport.

When you run a continuous read request function, it will be printed at the time of document transport.

5.12.3. Structure for Print setting

There are 2 types of structures for printing, one is " SK_DV_OPT_PRN_CONF" and another one is" SK_CONT_FEED_PRN_OPT"

"SK_DV_OPT_PRN_CONF" has setting for "Printing or Non Printing", "Printing position", "Printing detection", "Size of printing character", and "Distance of character".
"SK_CONT_FEED_PRN_OPT" has setting for "Digit of sequence number", and "start number of sequence number".

Function that uses the structure		
SkDv_ReqSetContFeedPrint		
Structure	Member	Description
SK_DV_OPT_PRN_CONF	iEnable	Set "printing" or "non printing" [Print]: use print setting [Non print] :disregard print setting
	iStartPos	Set start position of printing. Start position can select between 0.00mm and 355.0mm. Print area settingThere's 3mm margin in the top side, so ctual printing will start from 3.0mm even set 0.00mm
	iOrientation	Set printing direction. This function can select 0 degrees, or 180degrees rotated.
	iFontSize	Set font size to print the character. Width of character can select between 3.2mm and 6.4mm. (SR-11000mode is able to select width between 4.0 and 96.0mm)
	iFontSpace	Set interval between character and character. Interval ca n select between 0.8 and 92.0mm.
	iPrintMode	Select "print after feed(read mark data)" or "print with fee d read mark data)".

Function that uses the structure		
SkDv_ReqSetContFeedPrint		
Structure	Member	Description
SK_CONT_FEED_PRN_OPT	iDigits	Set the digits of sequence number (MAX 8digits)
	iStartNumber	Set the start number for print sequence number

5.12.4. Setting of Character to Print

SR-11000 can print 42 characters in SR-6500 mode, (SR-11000 mode can print 20characters.
163 types of character print to mark sheet.

Print character can be set “SkDv_ReqPrintString” These characters are print our under “SkDv_ReqFeedMarkSheet” for SR-6500 mode.

In case of use the SR-11000, use “SkDv_ReqContFeedPrint” and “SkDv_ReqSetPirntFixString”

Printable characters

	QTY	Characters
Numeric	10	0123456789
Capital	26	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Small letter	26	abcdefghijklmnopqrstuvwxyz
Japanese Kana	55	ｱｲｳｴｵｶｷｸｹｺｻｼｽｾｿﾀﾁﾂﾃﾄﾅﾆﾇﾈﾉﾊﾋﾌﾍﾎﾏﾐﾑﾒﾓﾔﾕﾖﾗﾘﾙﾚﾛﾜｦﾝｧｨｩｪｫｬｭｮｯ
symbol	46	SP(space)!”#$%&’()*+,－./:;<=>?@[¥]^_ ■{}～○×△□｡｢｣､･ﾞ ﾟ

5.12.5. Print area setting

Start position can set between 0.0 and 355.0m. Start position is able to set each 1.0mm interval (0, 1.0, 2.0,…354.0, 355.0mm). However, there is a limitation of non printable area (3.0mm). Then, if set 0.0mm from this setting, actual printing will start from 3.0mm.

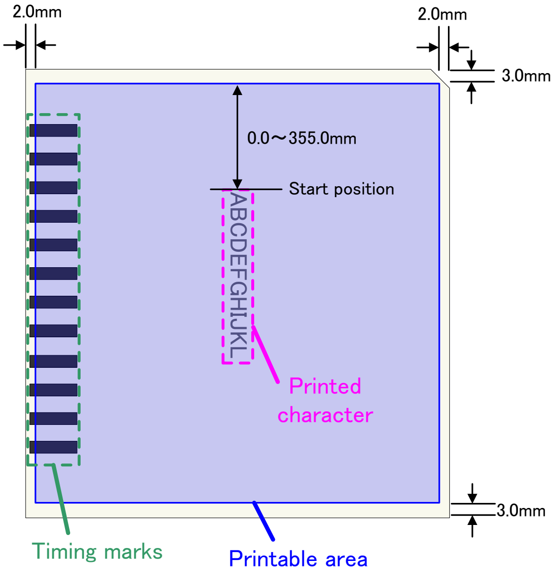


Fig 11 Printable area

In case adapt following 3 items, printing area will be different from standard area.

- Use printer unit.
- Print out after read marked data
- Sheet length over 233.0mm

In this case print will be started following position(Sheet length – 230.0 mm).

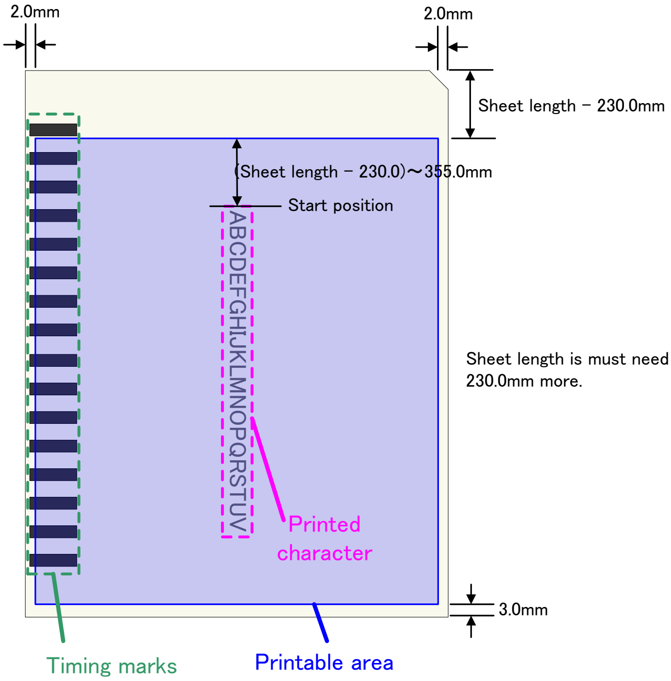


Fig 12 Printable area on specific condition.

5.12.6. Set print direction

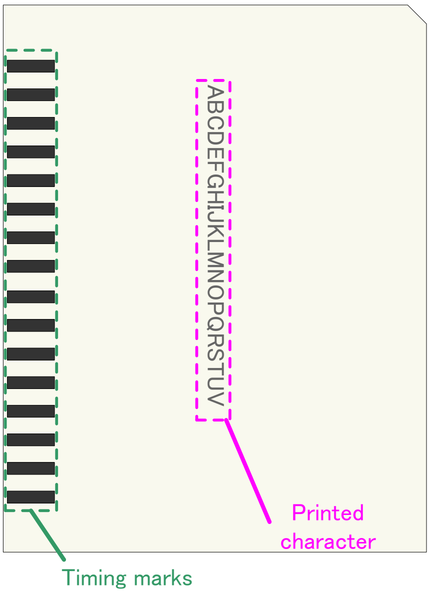


Fig 13 0 degrees

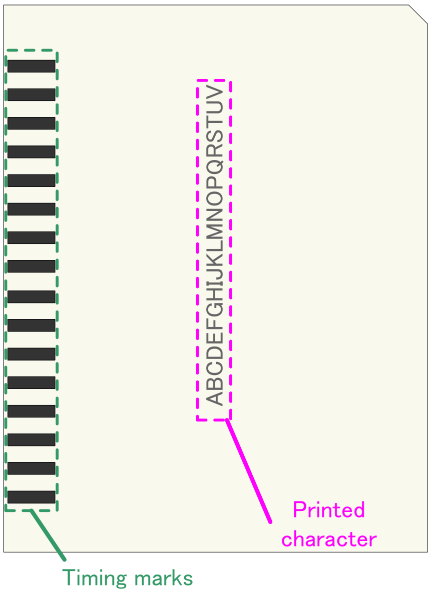


Fig 14 180 degrees

5.12.7. Character size

Width of character can select between 3.2 and 6.4mm.

(SR-11000 mode is able to set between 4.0 and 6.4mm)

The width of character can be increased in 0.8mm increments between 4.0 and 6.4mm.

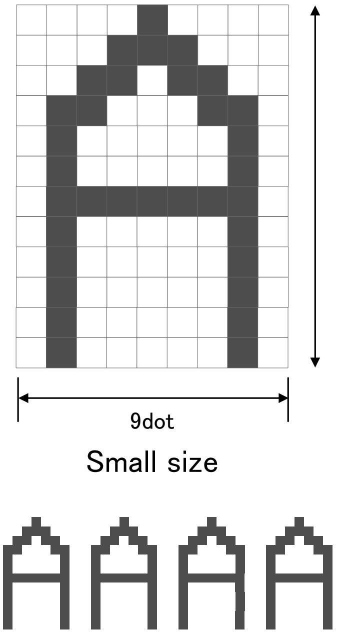


Fig 15 small print

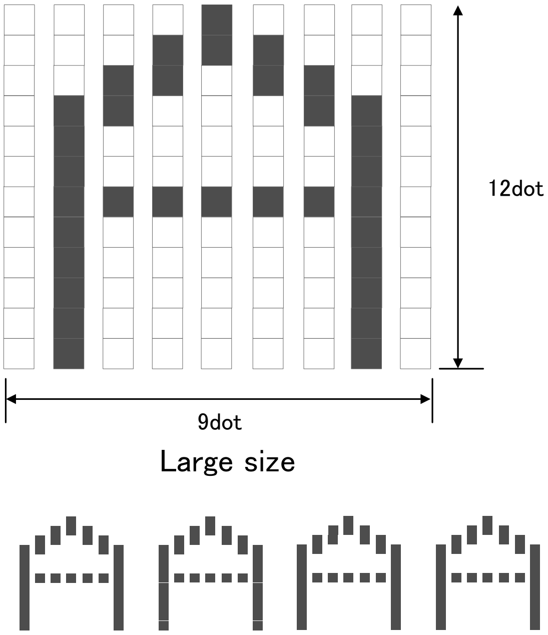


Fig 16 large print

5.12.8. Interval between characters

Set interval between characters. Interval can select between 0.8 and 92.0mm. The width of interval can be increased in 0.1mm increments between 4.0 and 92.0mm.

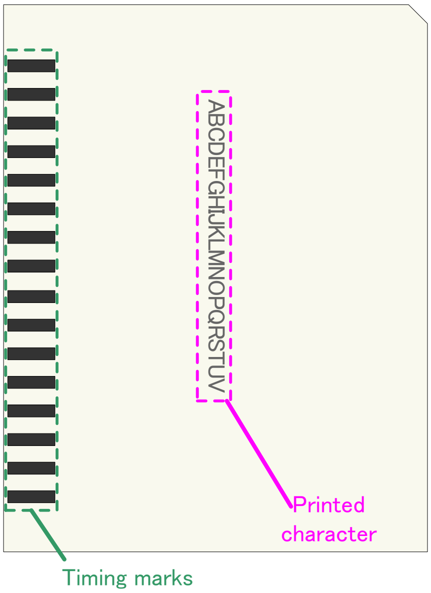


Fig 17 small interval

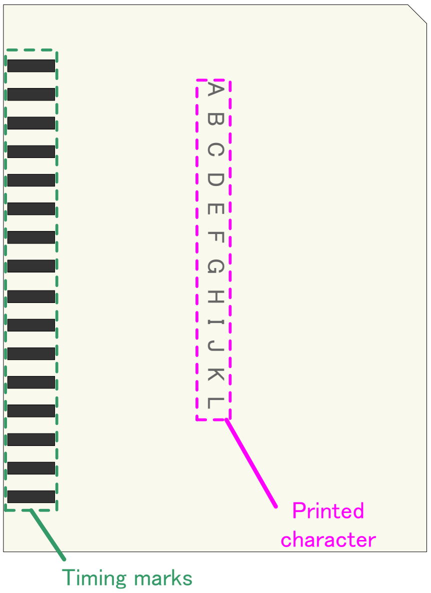


Fig 18 wide interval

5.12.9. Print Fixed strings or Sequence number

Fixed strings or sequence number can be print on the continuous reading (SR-11000mode)

examle) 3digits

… → 998 → 999 → 000 → 001 → …

Fixed strings can be set by ”SkDv_ReqSetPirntFixString”

Sequence numbercan be set by “SkDv_ReqSetContFeedPrint”

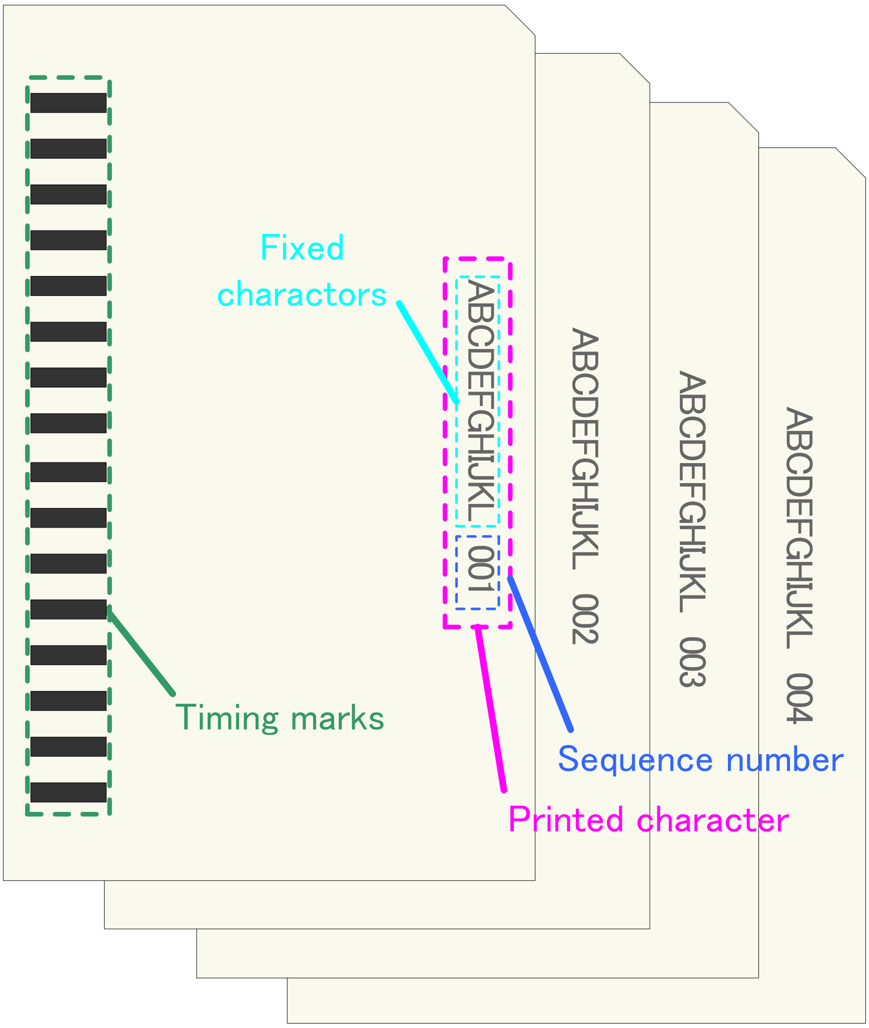


Fig 19 Fixed strings + Sequence number

5.13. Image Elongation Detection

Products equipped with the image reading function can detect the elongation of the image from which the sheet was read. (SR-3500 HYBRID, SR-6500 HYBRID)

Not available for SR-8000HYBRID series

5.13.1. Setting Of Image Elongation Detection Information

In order to detect the elongation of the image, the following three pieces of information are required.

・Validity of image elongation detection (enabled or disabled)

・The length of the sheet to be read.

・Threshold for judging that the image is stretched.

Store the above setting in the image elongation detection setting structure [5.3.11 SK_DV_IMG_ELGT_DETECT_CONF].

By performing image elongation detection information setting [5.6.7 SkDv_SetImageElongationDetectInfo] to middleware, image elongation can be detected. (It is not detected if image elongation detection is invalid.)

5.13.2. Method Of Image Elongation Detection

① Image elongation detection

The detected length (including margin) of image is compared with the set length of the length of the sheet to be read (including margin, to measure degree of elongation. If elongation exceeding the threshold is detected, it is regarded as an error.

② Rear end margin elongation detection

It detects the trailing edge of the acquired image (back side) and judges the margin length of the rear end of the image. If elongation exceeding the threshold is detected, it is regarded as an error

5.13.3. Notes On Image Elongation Detection

- Only the elongation is detected and the contraction is not detected.
- This detection is performed only on the image of the back side of the sheet. Therefore, under the condition that image enlongation detection is enabled, if it orders reading only front side, an error occurs.
  (SKDV_STS_PARAM_ERR_SCANNINGSIDE_Parameter Error Back side reading)
- The trailing edge of the form is detected from the read image. Therefore, if there is dirt or printing on the trailing edge, it may not be detected correctly. Since stain on the CIS or the pressure roller affects edge detection, please make sure to clean the parts before reading.
- In SR-6500 HYBRID, if the sheet size setting (see [5.6.16. SkDv_ReqSetImgReadSheetLength]) and the form size are mismatched, the image elongation amount may not be acquired correctly.

5.14. Layout Setting

This chapter is intended the products with continuous reading function .

5.14.1. Layout Configration Setup

A layout can be set up in continuous reading function.In order to set up a layout, it is necessary to execute multiples functions.The setup steps of the layout are as follows.

① Call “SkDv_ReqSetLayoutManage” and set the ID layout. (Which mark is ID, and How ID read.)
② Call “SkDv_ReqSetLayoutId” and set the details of ID. (details of ID, or How many timing marks, Action on errors etc.)
③ Call “SkDv_ReqSetWindowArea” and set window area. (Which area on the form uses as a window, or how read these window.)
④ Repeat 3 and set all window (MAX 80 window can be set on a ID)
⑤ Repeat 2 to 4and set all ID (MAX 16 ID’s)
⑥ Call “SkDv_ReqSetLayoutManageTerminate” to finish the setup

5.14.2. Flow chart to setup layout

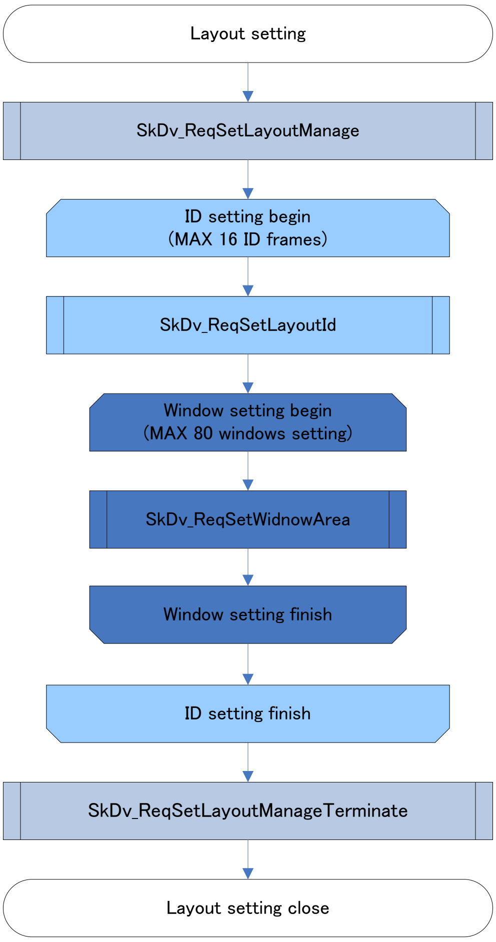


Fig 20 The flow chart of a diagram 16 layout setting procedure

5.14.3. Layout Setup

It has 3kinds of Parameter to setup the layout,
"SK_LAYOUT_MANAGE_CONF", "SK_LAYOUT_ID_CONF", and "SK_WINDOW_AREA_CONF”"

SK_LAYOUT_MANAGE_CONF is a structure to set how ID uses.
"SK_LAYOUT_ID_CONF" is a structure with ID value.
"SK_WINDOW_AREA_CONF" is a structure to set how window uses

Function that uses the structure		
SkDv_ReqSetLayoutManage		
Structure	Member	Description
SK_LAYOUT_MANAGE_CONF	blEnableId	Set up [not use] or [use] layout. If [use], use layout setup. If [not use], layout setup is disregarded. (However, errors checking by timing mark is performed. To setup the number of timing marks, use “LayoutIdPrm” of “SK_LAYOUT_ID_CONF“
	IdWindowPrm	Set the followings operation [read side] [the range of ID] [how ID read] [the direction of ID read] and [ID density level] uses by “SK_WINDOW_PRM”
	dwNgAction	Set operation when ID is NG. [select stacker -- paper ejection] or [a conveyance stop] either.
	LayoutOpt	Use “SK_LAYOUT_OPTION” and set up Option of ID layout (“No option” or “ID order*”)

(*) ID order checks is setting issue ERROR when it is not correspond order of ID.

Example 1.) 3ID’s are setup and they are named “0”, “1”, and “2”.
[transfer order]

OK 0 ->1 ->2	OK
NG 0 ->2	Error issued ID1 is jumped.
NG 1	Error issued ID0 / ID2 are over

Example 2.) Set 4kinds of ID’s are set, and they are named “3”, “5”, “7”, and “9”
[transfer order]

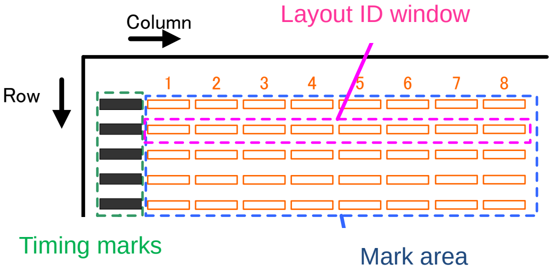
-3 ->5 ->7 ->9	
NG 3 ->5 ->9	Error issued ID 7 is jumped.
NG 5	Error issued ID3 / ID7 / ID9 are over.

Function that uses the structure		
SkDv_ReqSetLayoutId		
Structure	Member	Description
SK_LAYOUT_ID_CONF	LayoutIdPrm	Use “SK_LAYOUT_ID_PRM” and set the[Mark pattern setting ID window] and [the number of timing marks].
	iReject	Not use(fix as 0(ZERO))
	LayoutOpt	SK_LAYOUT_OPTION” and setupthe option of ID window. [not use] [mask marks] [fixed mark] [checkdigit] [check(Ascending order)] [check(descending order)] [check(no order)]

Function that uses the structure		
SkDv_ReqSetWindowArea		
Structure	Member	Description
SK_WINDOW_AREA_CONF	WindowPrm	Use “SK_WINDOW_PRM” and set [frontside/back side] [How ID reading] [direction of ID reading] [density level]
	WindowCheck	Use “SK_WINDOW_CHECK” and set[no mark acceptance]/[the number of minimum marks]/[the number of maximum marks]. These setting can give the judgement the number of marks in the window is correct. [No mark acceptance]is [accept], error is not appear even no mark in the window. Error appears in case [not accept]. [the number of minimum marks]/[the number of maximum marks] are set, error appears in casethe number of marks in window is smaller than[the number of minimum marks], or more than[the number of maximum marks]
	dwNgAction	Set operation when window is NG. [select stacker, paper eject] or [feed stop] either
	LayoutOpt	Use “SK_LAYOUT_OPTION” and set the option setting of window [not use] [mask marks] [fixed mark] [checkdigit] [check(Ascending order)] [check(descending order)] [check(no order)]

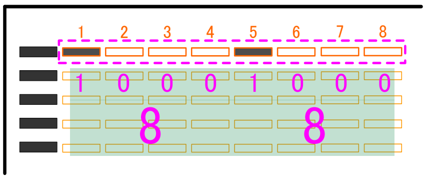
5.14.4. Setup Layout ID

In order to set up a layout, it is necessary to setup individual ID on a layout. ID is set “ucIdData“ of “SK_LAYOUT_ID_PRM”. “ucIdData” is 8bytes and “1” is Mark, and “0” is no mark



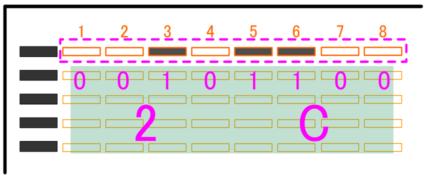
Layout ID

	ucIdData							
	1Byte	2Byte	3Byte	4Byte	5Byte	6Byte	7Byte	8Byte
Binary digit	1000	1000	0000	0000	0000	0000	0000	0000
Hexadecimal	8	8	0	0	0	0	0	0



Example ID setting(1)

	ucIdData							
	1Byte	2Byte	3Byte	4Byte	5Byte	6Byte	7Byte	8Byte
Binary digit	0010	1100	0000	0000	0000	0000	0000	0000
Hexadecimal	2	C	0	0	0	0	0	0

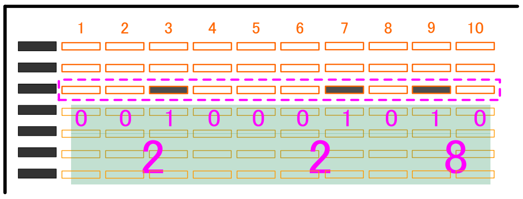


Example ID setting(2)

The position of ID can change a position and the range by changing the value of not corner of the form but the Col structure object within ID data management structure object (SK_LAYOUT_MANAGE_CONF), and a Row structure object.Detailed settings should look at the range specifications (excision) of 5.10.5 layouts. SK_LAYOUT_MANAGE_CONF5.14.6Layout Setting

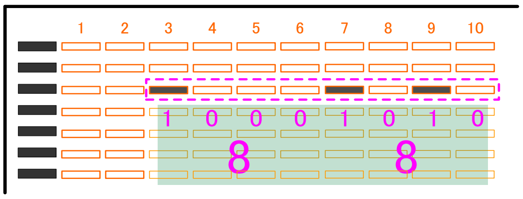
(*) Even if the same position has a mark of ID, it is recognized as the position and range of ID differing from each other as another ID.

	Row (Start)	Number of mark (Number)	Interval (Step)
Column	1	10	1
Row	3	1	1



ID setting which changed the position(1)

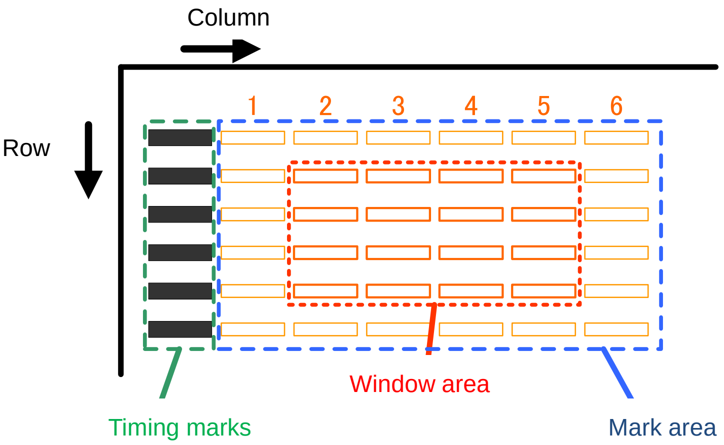
	Row (Start)	Number of mark (Number)	Interval (Step)
Column	3	8	1
Row	3	1	1



ID setting which changed the position(2)

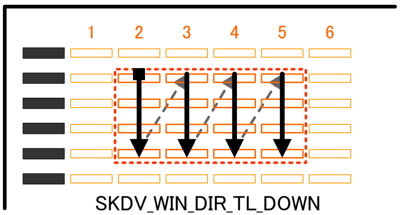
5.14.5. Layout read direction

It can be setup the destination to read layout. It can set “iDirection” parameter “SK_WINDOW_PRM”.

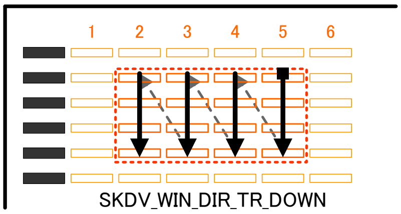


There are eight kinds of “iDirection”, and it defines as follows, respectively.

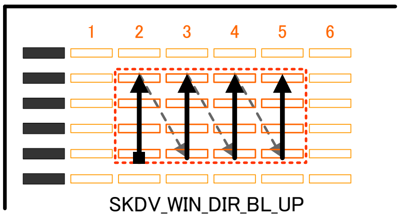
Constant	Value	Description
SKDV_WIN_DIR_TL_DOWN	0	From the up left to the bottom
SKDV_WIN_DIR_TR_DOWN	1	From the up right to the bottom
SKDV_WIN_DIR_BL_UP	2	From the low left to the top
SKDV_WIN_DIR_BR_UP	3	From the low right to the top
SKDV_WIN_DIR_TL_RIGHT	4	From the up left to right
SKDV_WIN_DIR_TR_LEFT	5	From the up right to the left
SKDV_WIN_DIR_BL_RIGHT	6	From the low left to right
SKDV_WIN_DIR_BR_LEFT	7	From the low right to the left



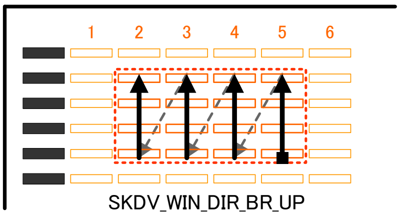
SKDV_WIN_DIR_TL_DOWN



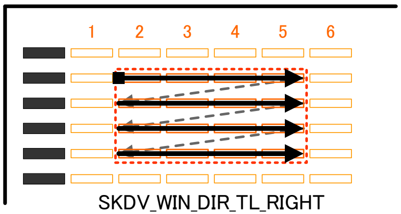
SKDV_WIN_DIR_TR_DOWN



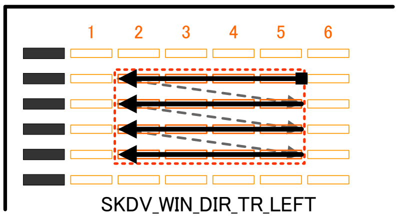
SKDV_WIN_DIR_BL_UP



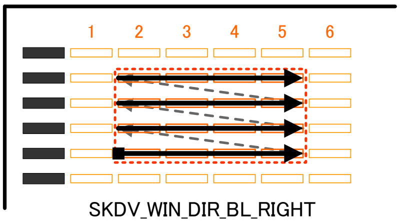
SKDV_WIN_DIR_BR_UP



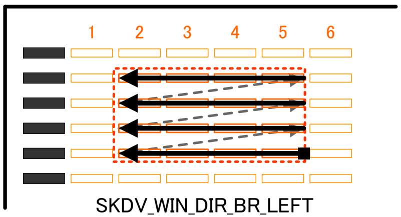
SKDV_WIN_DIR_TL_RIGHT



SKDV_WIN_DIR_TR_LEFT



SKDV_WIN_DIR_BL_RIGHT

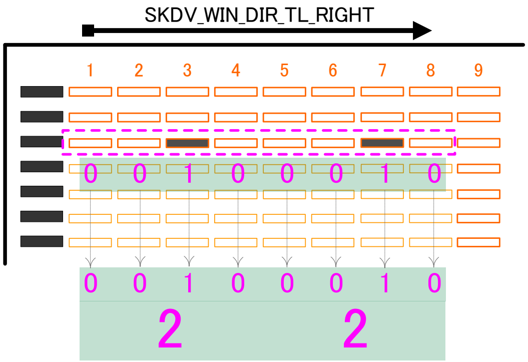


SKDV_WIN_DIR_BR_LEFT

When the layout reading directions differ, even if the same position has the mark of ID, it recognizes as another ID.

	Row (Start)	Number of mark (Number)	interval (Step)
Column	1	8	1
Row	3	1	1

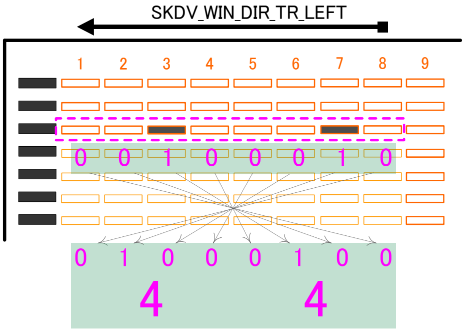
Layout read direction	From the upper left to the right



The example:ID setting which change position

	Row (Start)	Number of mark (Number)	interval (Step)
Column	1	8	1
Row	3	1	1

Layout read direction	From the upper right to the left sides



The example:ID ange position

5.14.6. Layout Setting

SR-11000, it can be set the area. This is called layout setting, and the area is called window.

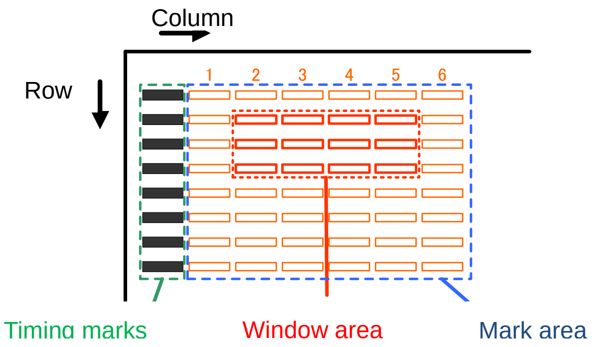
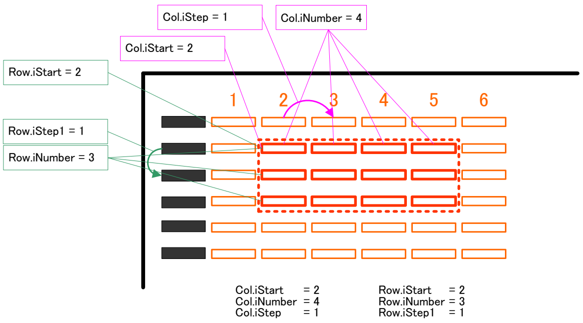


Image of window

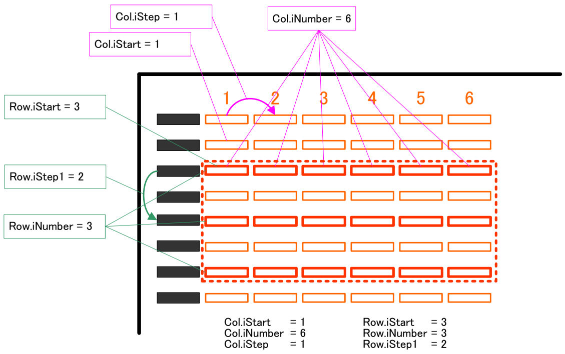
Area setting is controlled "SK_WINDOW_PRM"

	Row (Start)	Number of mark (Number)	Interval (Step)
Column	2	4	1
Row	2	3	1



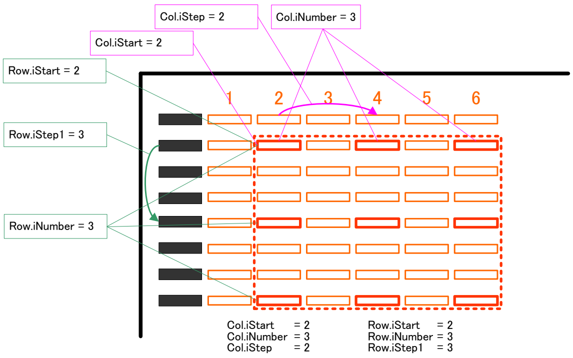
Example1

	Row (Start)	Number of mark (Number)	Interval (Step)
Column	1	6	1
Row	3	3	2



Example2

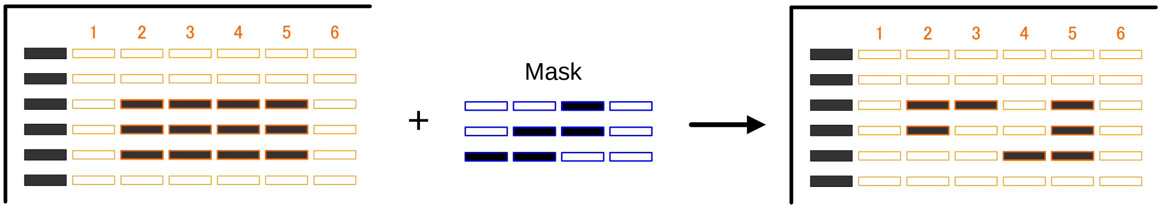
	Row (Start)	Number of mark (Number)	Interval (Step)
Column	2	3	2
Row	2	3	3



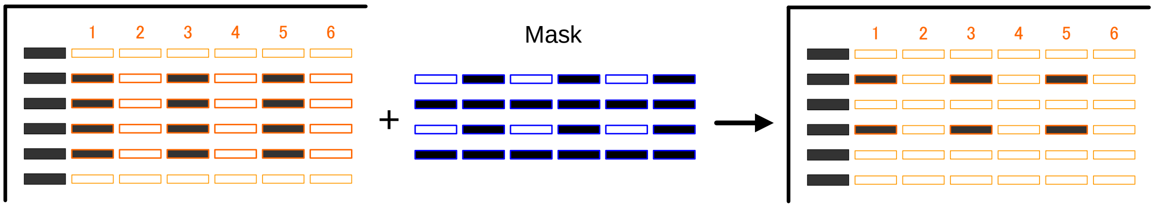
Example3

5.14.7. Mask Setting

MASK setup can be setup “SK_Layout“SKDV_LAYOUT_OPT_MASK” (SK_LAYOUT_OPTION).



Masked area removed from the window



Masked area removed from the window

5.14.8. Density level / Density gap

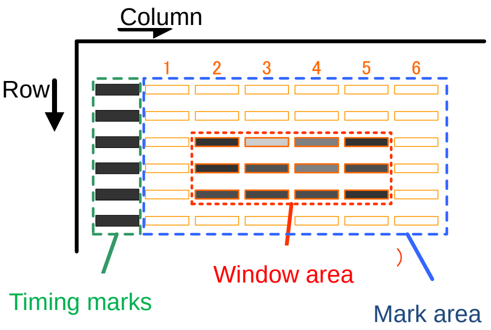
Density level setting (Sensitivity) can be set to pick up the marks in the each window. These setting can be set by each window.

All mark will be recognized if it corresponds to the following two conditions.

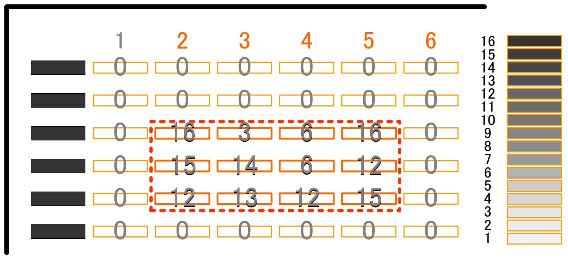
(1) Density level setting (Sensitivity) is lower than the actual mark density level(value).

(2) The marks whose Density level is higher than the value of density level setting(sensitivity) included the density level gap.

Specific examples are described below.

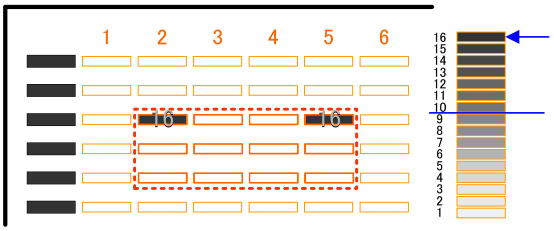


Mark sample



1	2	3	4	5	6
0	0	0	0	0	0
0	0	0	0	0	0
0	16	3	6	16	0
0	15	14	6	12	0
0	12	13	12	15	0
0	0	0	0	0	0

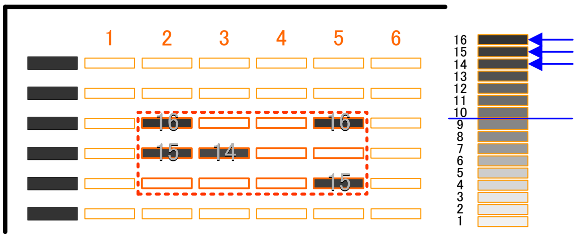
Density level of mark sample



Sensitivity = 10
Defference = 0

Density should detect over "level 10" and the differences against MAX level are "0"

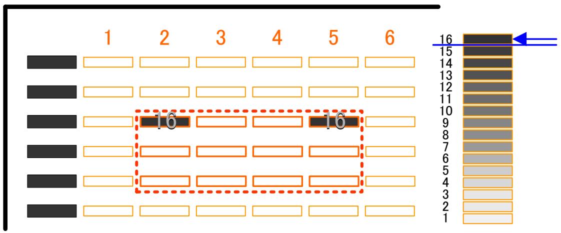
Mark recognition: in case sensitivity is 10±0



Sensitivity = 10
Defference = 3

Density should detect over "level 10" and the differences against MAX level are "3"

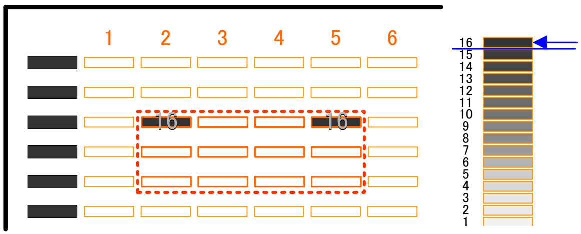
Mark recognition: in case sensitivity is 10±3



Sensitivity = 16
Defference = 0

Density should detect over "16" and the differences against MAX level are "0"

Mark recognition: in case sensitivity is 16±0



Sensitivity = 16
Defference = 3

Density should detect "level16 " and the differences against MAX level are "3"
= below 16 can not detect

Mark recognition: in case sensitivity is 16±3

OPTICAL MARK READER

SR-3500/6000/6500, SR-1800, SR-3500/6500/8000 HYBRID, SR-11000
API Reference Manual for Windows

Issued January 2021 REV11

http://www.sekonic.co.jp