

DP-DS820/DP-DS820 (A)  
Status API (DS820Stat.dll, DS820Stat64.dll)

Command Reference

Ver. 1.01\_D

Dai Nippon Printing Co., Ltd.

September 29, 2017

Revision History Chart		First Edition: 2016/02/29			Page 1/1	
Revised Item	Type of Revision	Revision No.	Document No.	Revision Date	Approved	Designer
	• New	0.1.0.0	0.10	2/29/2016		
P5	• Get media class has changed to the function that does not release the standby state.	0.1.0.0	0.10_1	3/18/2016		
P6 P42,43	• Add the following API functions. - Set Gamma Table - Get Gamma Table Checksum - Clear Data Table	0.2.0.0	0.20	5/27/2016		
P22 P14 P24 P29,31	The following sections have been changed. - Get Media Class Table of media type string has been added. - Get Printer Counter Value Table of “operation of counter L/A/B/P count-up” has been updated. - Set USB iSerialNumber availability setting. Initial value changed. “disable” -> “enable” The following notation has been changed. - RX media -> SD media - HDM media -> PP media	0.2.0.0	0.20_1	6/24/2016		
P40,41 P6,42	- The return value of the panorama printing start function has been added. - Continuous panorama printing setting function has been added.	1.0.0.0	1.00_D	9/13/2016		
P4	- Exclusive control for BiDi communication processing was added. (Windows 10 update program troubleshooting. Internal processing change.)	1.0.1.0	1.01_D	9/29/2017		

## Table of Contents

Introduction.....	4
API function .....	5
DLL Initialize .....	7
Get Printer Version Information .....	7
Get Printer Sensor Information .....	8
Get Printer Resolution.....	9
Get Printer Media Code .....	10
Get Printer Status .....	11
Get Printer Counter Value .....	13
Clear Printer Counter Value .....	15
Set Printer Counter Value (P) .....	15
Get the Number of Free Image Buffers .....	16
Get Remaining Print Quantity .....	16
Get the Media Counter of Remaining Sheets .....	17
Half Size Conversion Media Counter of Remaining Sheets.....	18
Get initial media count .....	19
Get Media Color Offset Value of the Lot .....	20
Get Media Lot Information .....	21
Get Media Class .....	22
Get RF-ID reserve data .....	22
Get Printer Serial Number.....	23
Set USB iSerialNumber availability setting.....	24
Get USB iSerialNumber availability setting.....	24
Set Firmware Update Mode.....	25
Write Firmware Data .....	25
Clear Color Data.....	26
Write Color Data .....	26
Set Color Data Version .....	27
Get Color Data Version .....	28
Get Color Data Version <Type designation> .....	28
Get Color Data Version <Type, Media designation> .....	29
Get Color Data Checksum .....	29
Get Color Data Checksum <Type designation> .....	30
Get Color Data Checksum <Type, Media designation> .....	31
Cutter Control.....	32
Full Cutter Set-up .....	33
Full Cutter Set-up Extended .....	35
Set Print Speed.....	37
Set Standby Mode Transition Time .....	38
Get Standby Mode Transition Time .....	38
Set Media End Keep Mode .....	39
Get Media End Keep Mode .....	39
Panorama Printing Start Check.....	40
Continuous Panorama Prints settings.....	42
Overcoat Finishing Control .....	43
Set Gamma Table.....	44
Get Gamma Table Checksum .....	45
Clear Data Table .....	45
Common Set Command .....	46
Common Get Command .....	46
Note for Sample Program .....	47

## **Introduction**



The copyrights for this document are the property of the holders of rights. Reproduction of any or all of the contents is prohibited.



The contents of this document are subject to change without prior notice.



Microsoft and Windows are registered trademarks of Microsoft Corporation valid in the USA and other countries.



You cannot resale/use the DP-DS820 without destination number (A) in the U.S.

### **Application Scope**

This document is a command reference manual for the DP-DS820/DP-DS820 (A) Printer Status API.

### **Supported Operating Systems and Operating Environment**

This API runs on Windows 7, Windows 8, and windows 10.

In Windows 10 environment, please use DLL of ver.1.0.1.0 or later.

## API function

A table of the API functions is shown below.

Also, if an API function is used while the printer is in Standby mode, the table indicates whether the Standby will be released or not.

### ■Notes

- If you are using C language, please include "DS820Stat.h", and add DS820Stat.lib to the link library. When running this, put in "DS820Stat.dll"(for 32BitOS) as a reference pass. (For 64bitOS, use "DS820Stat64.dll")
- When using VB.Net, since the pointer cannot be given directly from VB to the DLL function argument, a wrapper function for VB.Net with the same name as the DLL has been prepared. The definition is written in the DS820Stat.vb file, so add the file to the VB project to use it.  
Both DS820Stat.vb are defined such that they can be used and switched between 32⇔64 Bit OS environments. For the 32 bit OS, make the first line of the DS820Stat.vb "#Const x64=False", and for the 64 bit OS make the first line "#Const x64=True".

	API function name		Standby state	Point of notice
DLL Initialize	CvInitialize()	PortInitialize()	Not release	
Get Printer Version Information	CvGetVersion()	GetFirmwVersion()	Not release	
Get Printer Sensor Information	CvGetSensorInfo()	GetSensorInfo()	Release	
Get Printer Resolution	CvGetResolutionH() CvGetResolutionV()	GetResolutionH() GetResolutionV()	Release	
Get Printer Media Code	CvGetMedia()	GetMedia()	Not release	
Get Printer Status	CvGetStatus()	GetStatus()	Not release	
Get Printer Counter Value	CvGetCounterL() CvGetCounterA() CvGetCounterB()	GetCounterL() GetCounterA() GetCounterB() GetCounterP() GetCounterMatte() GetCounterM()	Release	
Clear Printer Counter Value	CvSetClearCounterA() CvSetClearCounterB()	SetClearCounterA() SetClearCounterB() SetClearCounterM()	Release	
Set Printer Counter Value(P)		SetCounterP()	Release	
Get the Number of Free Image Buffers	CvGetFreeBuffer()	GetFreeBuffer()	Release	
Get Remaining Print Quantity	CvGetPQTY()	GetPQTY()	Release	
Get the Media Counter of Remaining Sheets	CvGetMediaCounter()	GetMediaCounter()	Not release	
Half Size Conversion Media Counter of Remaining Sheets		GetMediaCounterH()	Not release	
Get initial media count		GetInitialMediaCount()	Not release	
Get Media Color Offset Value of the Lot	CvGetMediaColorOffset()	GetMediaColorOffset()	Release	
Get Media Lot Information	CvGetMediaLotNo()	GetMediaLotNo()	Not release	
Get media class		GetRfidMediaClass()	Not release	
Get RF-ID reserve data		GetRfidReserveData()	Release	
Get Printer Serial Number	CvGetSerialNo();	GetSerialNo()	Not release	
Set USB iSerialNumber availability setting		SetUSBSerialEnable()	Release	
Get USB iSerialNumber availability setting		GetUSBSerialEnable()	Release	
Set Firmware Update Mode	CvSetFirmwUpdateMode()	SetFirmwUpdateMode()	Release	
Write Firmware Data	CvSetFirmwDataWrite()	SetFirmwDataWrite()	Release	VB.Net Wrapper function

	API function name		Standby state	Point of notice
Clear Color Data	CvSetColorDataClear()	SetColorDataClear()	Release	
Write Color Data	CvSetColorDataWrite()	SetColorDataWrite()	Release	VB.Net Wrapper function
Set Color Data Version	CvSetColorDataVersion()	SetColorDataVersion()	Release	VB.Net Wrapper function
Get Color Data Version	CvGetColorDataVersion()	GetColorDataVersion()	Release	
Get Color Data Version <Type designation>		GetColorDataVersionRes()	Release	
Get Color Data Version <Type,Media designation>		GetColorDataVersionResEX() ( )	Release	
Get Color Data Checksum	CvGetColorDataChecksum()	GetColorDataChecksum()	Release	
Get Color Data Checksum <Type designation>		GetColorDataChecksumRes()	Release	
Get Color Data Checksum <Type,Media designation>		GetColorDataChecksumResEX() ( )	Release	
Cutter Control Command		SetCutterMode()	Release	
Full Cutter Set-up		SetFullCutterSetup()	Release	
Full Cutter Set-up Extension		SetFullCutterSetupEx()	Release	
Set Print Speed		SetPrintSpeed()	Release	
Set Standby Mode Transition Time		SetStandbyTime()	Release	
Get Standby Mode Transition Time		GetStandbyTime()	Release	
Set Media End Keep Mode		SetEndKeepMode()	Release	
Get Media End Keep Mode		GetEndKeepMode()	Release	
Panorama Printing Start Check		GetPanoramaPrintable()	Release	
Continuous Panoramic Prints settings		SetContPanorama()	Release	
Overcoat finish		SetOvercoatFinish()	Release	
Set Gamma Table		SetGammaTable()	Release	
Get Gamma Table Checksum		GetGammaTableChecksum()	Release	
Clear Data Table		SetDataTableClear()	Release	
Common Set Command	CvSetCommand()	SetCommand()		VB.Net Wrapper function
Common Get Command	CvGetCommandEX()	GetCommandEX()		VB.Net Wrapper function

**DLL Initialize**

---

[Format]	long PortInitialize(LPWSTR p); long CvInitialize(LPWSTR p);	
[Parameter]	p:	Pointer to the port name WSTR (2 byte Unicode)
[Return]	Successful: Failure:	Port number -1
[Note]	Initializes API and returns the port number. If more than one port are used, please get each port number by generating the command repeatedly.	
[Sample Coding]	<div>&lt; Visual C &gt;</div> <pre>long DS820; if(( DS820 = PortInitialize( L"USB001" )) &lt; 0 ){     // error }</pre> <div>&lt; VB.NET &gt;</div> <pre>Dim DS820 As Integer DS820 = PortInitialize("USB001"); If DS820 &lt; 0 Then GoTo Error</pre>	

**Get Printer Version Information**

---

[Format]	long GetFirmwVersion( long lPortNum, LPSTR p ); long CvGetVersion( long lPortNum, LPSTR p );	
[Parameter]	lPortNum: p:	Port number Pointer to the receiving buffer
[Return]	Successful: Failure:	The number of characters received in buffer p -1
[Note]	Receives the printer version information in the buffer.	
	<div>[Version Information]</div> <ul style="list-style-type: none"> <li>• DS820A xx.xx (For the United States)</li> <li>• DS820 xx.xx (For other countries)</li> </ul> <div>xx.xx indicates the version.</div>	
[Sample Coding]	<div>&lt; Visual C &gt;</div> <pre>char rbuf[256]; if( GetFirmwVersion( DS820, (LPSTR)rbuf ) &gt; 0 ){     // Next process }</pre> <div>&lt; VB.NET &gt;</div> <pre>Dim s As String = New String("", 255) Dim i As Integer i = GetFirmwVersion(DS820, s) If i &gt; 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"</pre>	
[Version Information]	DP-DS820 **. **	Firmware version information

## Get Printer Sensor Information

---

[Format]	long GetSensorInfo( long lPortNum, LPSTR p ); long CvGetSensorInfo( long lPortNum, LPSTR p );	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
[Return]	Successful:	The number of characters received in buffer p
	Failure:	-1
[Note]	Receives value of each sensor (through AD converter) in the buffer.	
[Sample Coding]	<div>&lt; Visual C &gt;</div> <pre>char rbuf[256]; if( GetSensorInfo( DS820, (LPSTR)rbuf ) &gt; 0 ){     // Next process }</pre> <div>&lt; VB.NET &gt;</div> <pre>Dim s As String = New String("", 255) Dim i As Integer i = GetSensorInfo(DS820, s) If i &gt; 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"</pre>	

### [Sensor Information for]

HDT-***;	Head temperature	
MDT-***;	Media temperature	
PMK-***;	Paper mark	
RML-***;	Ribbon mark left	(Unused,"000" always returns to value)
RMC-***;	Ribbon mark center	(Unused)
RMR-***;	Ribbon mark right	(Unused)
PSZ-***;	Paper size	(Unused,"000" always returns to value)
PNT-***;	Paper notch	(Unused,"000" always returns to value)
PJM-***;	Paper jam	(Unused,"000" always returns to value)
PED-***;	Paper end	
PET-***;	Paper empty	(Unused,"000" always returns to value)
HDV-***;	Head voltage	
HMD-***;	Humidity	
GSR-***;	Color Sensor (Red)	
GSG-***;	Color Sensor (Green)	
GSB-***;	Color Sensor (Blue)	



**Get Printer Resolution**

---

[Format]	long GetResolutionH( long lPortNum ); long CvGetResolutionH( long lPortNum ); long GetResolutionV( long lPortNum ); long CvGetResolutionV( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	Horizontal, or Vertical resolution (dpi)
	Failure:	-1
[Note]	GetResolutionH(),CvGetResolutionH() returns Horizontal resolution (dpi). GetResolutionV(),CvGetResolutionV() returns Vertical resolution (dpi).	
[Sample Coding]	<div>&lt; Visual C &gt;</div> <pre> long rh; if(( rh = GetResolutionH( DS820 )) &gt;= 0 ){     // Returns Horizontal resolution to rh } </pre> <div>&lt; VB.NET &gt;</div> <pre> Dim i As Integer i = GetResolutionH(DS820) If i &gt;= 0 Then Text1.Text = Str(i) Else Text1.Text = "ERROR!" </pre>	

## Get Printer Media Code

---

[Format]            long GetMedia( long lPortNum, LPSTR p );  
                      long CvGetMedia( long lPortNum, LPSTR p );

[Parameter]        lPortNum:            Port number  
                      p:                    Pointer to the receiving buffer

[Return]            Successful:            The number of characters received by buffer p  
                      Failure:                -1

[Note]              Receives media code in the buffer.

[Sample Coding]    < Visual C >  
                      char rbuf[256];  
                      if( GetMedia( DS820, (LPSTR)rbuf ) > 0 ){  
                                          // Next process  
                      }  
  
                      < VB.NET >  
                      Dim s As String = New String("", 255)  
                      Dim i As Integer  
                      i = GetMedia(DS820, s)  
                      If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"

[Media code]        Media code is defined by 5 decimal bytes. The return value for the Get Media Code command is referred to as the following 5 bytes (ASCII numeric).

4th byte (0n000)	Paper Type	3rd, 2nd byte (00nn0)	Paper Size	1st byte (0000n)	Positioning Mark
00000	Normal Paper	00500	8x10	00000	No Mark
01000	Sticker	00510	8x12	00001	With Mark
		00600	A4		(Back Print)

### Example

Paper Size	Paper Type	(Size: Width x Height)	Media Code
8x10	Normal paper	(207.0 x 257.0 mm)	00500
8x12	Normal paper	(207.0 x 308.0 mm)	00510
A4	Normal paper	(214.0 x 300.0 mm)	00600

**Get Printer Status**

---

[Format]                DWORD GetStatus( long lPortNum );  
                           DWORD CvGetStatus( long lPortNum );

[Parameter]           lPortNum:            Port number

[Return]               Successful:            Status  
                           Failure:                STATUS\_ERROR

[Note]                This returns the printer status.  
                           Bit position of the status is defined in DS820Stat.h by macro.(DS820Stat.vb for Visual Basic)  
                           Meanings of the symbols are as follows:

GROUP_USUALLY	[Usual Operation]	Group identification bit
GROUP_SETTING	[Setting Error]	Group identification bit
GROUP_HARDWARE	[Hardware Error]	Group identification bit
GROUP_SYSTEM	[System Error]	Group identification bit
GROUP_FLSHPROG	[Rewriting Mode]	Group identification bit
STATUS_ERROR	Status Receiving Error	
STATUS_USUALLY_IDLE	Idle	
STATUS_USUALLY_PRINTING	Printing	
STATUS_USUALLY_PAPER_END	Paper End	
STATUS_USUALLY_RIBBON_END	Ribbon End	
STATUS_USUALLY_COOLING	Head Cooling Down	
STATUS_USUALLY_MOTCOOLING	Motor Cooling Down	
STATUS_USUALLY_STANDBY_MODE	Standby Mode	
STATUS_SETTING_COVER_OPEN	Cover Open	
STATUS_SETTING_PAPER_JAM	Paper Jam	
STATUS_SETTING_RIBBON_ERR	Ribbon Error (Detect Error, Ribbon break)	
STATUS_SETTING_PAPER_ERR	Paper Definition Error	
STATUS_SETTING_DATA_ERR	Data Error (Illegal command)	
STATUS_SETTING_SCRAPBOX_ERR	Scrap Box Error	
STATUS_HARDWARE_ERR01	Head Voltage Error	
STATUS_HARDWARE_ERR02	Head Position Error	
STATUS_HARDWARE_ERR03	Power Supply Fan Stopped	
STATUS_HARDWARE_ERR04	Cutter Error (Cut jamming etc)	
STATUS_HARDWARE_ERR05	Pinch Roller Position Error	
STATUS_HARDWARE_ERR06	Abnormal Head Temperature	
STATUS_HARDWARE_ERR07	Abnormal Media Temperature	
STATUS_HARDWARE_ERR08	Ribbon Tension Error	
STATUS_HARDWARE_ERR09	RF-ID Module Error	
STATUS_HARDWARE_ERR10	Abnormal Motor Temperature	
STATUS_SYSTEM_ERR01	System Error	
STATUS_FLSHPROG_IDLE	Idling for receiving rewriting data	
STATUS_FLSHPROG_DATA_ERR1	Transfer Data Error	
STATUS_FLSHPROG_DEVICE_ERR1	Device Error	
STATUS_FLSHPROG_OTHERS_ERR1	Other Error	

[Sample Coding]

&lt; Visual C &gt;

```

include "DS820Stat.h"
long stat;

stat = GetStatus( DS820 );
if( stat & GROUP_USUALLY ){ // Usual Operation Status Group
    switch( stat ){
        case STATUS_USUALLY_IDLE: ;
        case STATUS_USUALLY_PRINTING: ;
        case STATUS_USUALLY_PAPER_END: ;
        :
    }
}
if( stat & GROUP_SETTING ){ // Setting Error Status Group
    switch( stat ){
        case STATUS_SETTING_COVER_OPEN: ;
        :
        :
    }
}
}

```

&lt; VB.NET &gt;

```

Dim stat as Integer

stat = GetStatus(DS820)
If stat And GROUP_USUALLY Then          ' Usual Operation Status Group
    Select Case stat
        Case STATUS_USUALLY_IDLE: Text1.Text = "IDLE"
        Case STATUS_USUALLY_PRINTING: Text1.Text = "PRINTING"
        Case STATUS_USUALLY_PAPER_END: Text1.Text = "PAPER_END"
        Case STATUS_USUALLY_RIBBON_END: Text1.Text = "RIBBON_END"
        Case STATUS_USUALLY_COOLING: Text1.Text = "COOLING"
    End Select
ElseIf stat And GROUP_SETTING Then      ' Setting Error Status Group
    ' Operation Error ( )
ElseIf stat And GROUP_HARDWARE Then    ' Hardware Error Status Group
    ' Hardware Error ( )
ElseIf stat And GROUP_SYSTEM Then      ' System Error Status Group
    ' System Error ( )
End If

```

**Get Printer Counter Value**

---

[Format]            long GetCounterL( long lPortNum );  
                      long CvGetCounterL( long lPortNum );  
                      long GetCounterA( long lPortNum );  
                      long CvGetCounterA( long lPortNum );  
                      long GetCounterB( long lPortNum );  
                      long CvGetCounterB( long lPortNum );  
                      long GetCounterP( long lPortNum );  
                      long GetCounterMatte( long lPortNum );  
                      long GetCounterM( long lPortNum );

[Parameter]        lPortNum:            Port number

[Return]            Successful:            Counter Value  
                      Failure:                -1

[Note]              GetCounterL(), CvGetCounterL()   Returns Life Counter Value.  
                      GetCounterA(), CvGetCounterA()   Returns Counter A Value.  
                      GetCounterB(), CvGetCounterB()   Returns Counter B Value.  
                      GetCounterP( long lPortNum )   Returns Counter P Value  
                      GetCounterMatte()   Returns Matte Counter Value.  
                      GetCounterM()   Returns Counter M Value.

Each counter is counted up 1 for each photo printed.  
 When printing multi-cut image, it will be counted up 2 when the second image is printed.

Counter P is initialized when power on.   Random setting is OK with SetCounterP().  
 Counter P value is countered according to each discharge of image.

When overcoat finish is matte print, Matte Counter and Counter M will be counted up (Life Counter and Counter A/B are counted up also).

Counter M is clearable with SetClearCounterM() function.  
 Specification of count up is the same as that of the above-mentioned Life Counter and Counter A/B.

[Sample Coding]    < Visual C >  
                      long counter;  
                      if(( counter = GetCounterL( DS820 )) >= 0 ){  
                                  // Returns Life Counter Value to counter  
                      }

< VB.NET >  
 Dim c As Integer  
 c = GetCounterL(DS820)  
 If c >= 0 Then Text1.Text = Str(c) Else Text1.Text = "ERROR!"

Operation of Counter L/A/B/P count up

Count timing of counter is after cutting a print picture correctly.  
When an error occurs, a counter does not go up.

	Print Size		Counter L/A/B/ Matte/M	Counter P
Single-Cut	8x10		+1	+1
	8x12		+1	+1
	8x4		+1	+1
	8x5		+1	+1
	8x6		+1	+1
	8x8		+1	+1
	A5 Format		+1	+1
	A4 Format		+1	+1
Multi-Cut	8x4x2	1st Image	---	+1
		2nd Image	+1	+1
	8x5x2	1st Image	---	+1
		2nd Image	+1	+1
	8x6x2	1st Image	---	+1
		2nd Image	+1	+1
	A5x2	1st Image	---	+1
		2nd Image	+1	+1
	8x4x3	1st Image	---	+1
		2nd Image	---	+1
		3rd Image	+1	+1
2inch-Cut	8x4	1st Sheet	---	+1
		2nd Sheet	+1	+1
	8x6	1st Sheet	---	+1
		2nd Sheet	---	+1
		3rd Sheet	+1	+1
	8x8	1st Sheet	---	+1
		2nd Sheet	---	+1
		3rd Sheet	---	+1
		4th Sheet	+1	+1
	8x10	1st Sheet	---	+1
		2nd Sheet	---	+1
		3rd Sheet	---	+1
		4th Sheet	---	+1
		5th Sheet	+1	+1
	8x12	1st Sheet	---	+1
		2nd Sheet	---	+1
		3rd Sheet	---	+1
		4th Sheet	---	+1
		5th Sheet	---	+1
		6th Sheet	+1	+1

**Clear Printer Counter Value**

---

[Format]	BOOL SetClearCounterA( long lPortNum ); BOOL CvSetClearCounterA( long lPortNum );  BOOL SetClearCounterB( long lPortNum ); BOOL CvSetClearCounterB( long lPortNum );  BOOL SetClearCounterM( long lPortNum );	
[Parameter]	PortNum:	Port number
[Return]	Successful:	True
	Failure:	False
[Note]	SetClearCounterA(), CvSetClearCounterA() Clears Counter A. SetClearCounterB(), CvSetClearCounterB() Clears Counter B. SetClearCounterM() Clears Counter M.	
[Sample Coding]	<pre>&lt; Visual C &gt; if( SetClearCounterA( DS820 )){     // Counter A was cleared }  &lt; VB.NET &gt; If SetClearCounterA(DS820) &lt;&gt; 0 Then     ' Counter A was cleared EndIf</pre>	

**Set Printer Counter Value (P)**

---

[Format]	BOOL SetCounterP( long lPortNum、 long lCounter );	
[Parameter]	lPortNum:	Port number
	lCounter:	Set counter value
[Return]	Successful:	TRUE
	Failure:	FALSE
[Note]	SetCounterP() Sets Counter value P. Set random number to Counter P. When power off, it is initialized to 0.	
[Sample Coding]	<pre>&lt; Visual C &gt; if( SetCounterP( DS820, 100)){     // Set Counter P }  &lt; VB.NET &gt; If SetCounterP(DS820, 100) &lt;&gt; 0 Then     ' Set Counter P EndIf</pre>	

## Get the Number of Free Image Buffers

---

[Format]	long GetFreeBuffer( long lPortNum ); long CvGetFreeBuffer( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	The number of free print buffers
	Failure:	-1
[Note]	Printer will return the number of free image buffers.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> long bn; if(( bn = GetFreeBuffer( DS820 )) >= 0 ){ <i>// Returns the number of free buffers to bn</i> }	
	<b>&lt; VB.NET &gt;</b> Dim bn As Integer bn = GetFreeBuffer(DS820)	

## Get Remaining Print Quantity

---

[Format]	long GetPQTY( long lPortNum ); long CvGetPQTY( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	Remaining number of images to be printed
	Failure:	-1
[Note]	Returns the number of images remaining to be printed.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> long number; if(( number = GetPQTY( DS820 )) >= 0 ){ <i>// Returns the number of remaining prints to number</i> }	
	<b>&lt; VB.NET &gt;</b> Dim number As Integer number = GetPQTY(DS820)	



## Get the Media Counter of Remaining Sheets

---

[Format]            long GetMediaCounter( long lPortNum );  
                      long CvGetMediaCounter( long lPortNum );  
                      long GetMediaCounter\_R( long lPortNum );

[Parameter]        lPortNum:            Port number

[Return]            Successful:            The number of remaining sheets  
                      Failure:                -1

[Note]              Printer will return the number of sheets remaining in the printer.

With operating conditions of media, the actual number of sheets which can be printed, and media tag number of sheets may be different. Detection of a media end should use together with the ribbon or paper end status of GetStatus() function.

■Initial value of number of sheets for each media

Media	number of sheets default Value	number of printable sheets
8x10	130	130
8x12	110	110
A4	110	110

[Sample Coding]    < Visual C >  
 long number;  
 if(( number = GetMediaCounter( DS820 )) >= 0 ){  
                      // Returns the number of remaining sheets to *number*  
 }  
 }

< VB.NET >  
 Dim number As Integer  
 number = GetMediaCounter(DS820)

## Half Size Conversion Media Counter of Remaining Sheets

---

[Format]	long GetMediaCounterH( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	The Media Counter of Remaining Sheets which has been converted to Half size.
	Failure:	-1
[Explanation]	When you load the 8x10 size media to printer, it will return the number of remaining sheets which has been converted to 8x5 size. Or when you load the 8x12 size media to printer, it will return the number of remaining sheets which has been converted to 8x6 size. Or when you load the A4 size media to printer, it will return the number of remaining sheets which has been converted to A5 size.	

Depending on media use conditions, there may be a difference between the media tag quantity and the actual printable quantity.

To detect the media end, use this in conjunction with the ribbon end and paper end GetStatus() function.

### ■Initial number of sheets of each media

Media	number of sheets default Value	number of printable sheets
8x10	260	260
8x12	220	220
A4	220	220

[Sample Coding]	< Visual C >
	<pre>long number; if(( number = GetMediaCounterH( DS820 )) &gt;= 0 ){     //Returns the number of remaining sheets to <i>number</i> }</pre>
	< VB.NET >
	<pre>Dim number As Integer number = GetMediaCounterH(DS820)</pre>

**Get initial media count**

---

[Format]	long GetInitialMediaCount( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	Initial media count
	Failure:	-1
[Explanation]	printer will return the initial media count.	
[Sample Coding]	< Visual C >	
	<pre> long number; if(( number = GetInitialMediaCount ( DS820 )) &gt;= 0 ){     // Returns the number of initial media count to <i>number</i> } </pre>	
	< VB.NET >	
	<pre> Dim number As Integer number = GetInitialMediaCount (DS820) </pre>	

**Get Media Color Offset Value of the Lot**

---

[Format]                    long GetMediaColorOffset( long lPortNum );  
                              long CvGetMediaColorOffset( long lPortNum );

[Parameter]              lPortNum:                    Port number

[Return]                    Successful:                    Media color offset value of the lot  
                              Failure:                        -1

[Note]                      Printer will return the offset value.

[Sample Coding]          < Visual C >  
                              long offset;  
                              if(( offset = GetMediaColorOffset( DS820 )) >= 0 ){  
                                                                  // Returns the value to *offset*  
                              }

< VB.NET >  
 Dim offset As Integer  
 offset = GetMediaColorOffset(DS820)

[Example]                    In the case where offset = 169082895 (decimal) --> 0x0A14000F (hex), the offset values for each color are defined as below.

Color	Offset value
Yellow	10 (0x0A)
Magenta	20 (0x14)
Cyan	0 (0x00)
Op	15 (0x0F)

**Get Media Lot Information**

---

[Format]	long GetMediaLotNo( long lPortNum, LPSTR p ); long CvGetMediaLotNo( long lPortNum, LPSTR p );	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Note]	Printer will return the information stored in RF-ID tag of the media.	
[Sample Coding]	<div>&lt; Visual C &gt;</div> <pre>char rbuf[256]; if( GetMediaLotNo( DS820, (LPSTR)rbuf ) &gt; 0 ){     // Next process }</pre> <div>&lt; VB.NET &gt;</div> <pre>Dim s As String = New String("", 255) Dim i As Integer i = GetMediaLotNo(DS820, s) If i &gt; 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"</pre>	
[Media tag information]	ML*****	User-specific information <16Bytes>

**Get Media Class**

---

[Format]	long GetRfidMediaClass( long lPortNum, LPSTR p );	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Note]	The media class data recorded in the RF-ID tag is returned. (ASCII character of four digits)	

## •Media Class Information

Receive string	Description
0001	Digital (SD) media.
0003	Pure Premium (PP) media.

[Sample Coding]	<b>&lt; Visual C &gt;</b> char rbuf[256]; if(GetRfidMediaClass( DS820, (LPSTR)rbuf ) > 0 ){ // Next process }
	<b>&lt; VB.NET &gt;</b> Dim s As String = New String("", 255) Dim i As Integer i = GetRfidMediaClass(DS820, s) If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"

**Get RF-ID reserve data**

---

[Format]	long GetRfidReserveData ( long lPortNum, LPSTR p, DWORD dwPage );	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
	dwPage:	Page of reserve data
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Note]	The reserve data recorded in the RF-ID tag is returned. (ASCII character of four digits)	

[Sample Coding]	<b>&lt; Visual C &gt;</b> char rbuf[256]; if(GetRfidReserveData ( DS820, (LPSTR)rbuf , 1) > 0 ){ // Page 1 // Next process }
	<b>&lt; VB.NET &gt;</b> Dim s As String = New String("", 255) Dim i As Integer i = GetRfidReserveData (DS820, s, 2) 'Page 2 If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"

**Get Printer Serial Number**

---

[Format]	long GetSerialNo( long lPortNum, LPSTR p ); long CvGetSerialNo( long lPortNum, LPSTR p );
[Parameter]	lPortNum: Port number p: Pointer to the receiving buffer
[Return]	Successful: The number of characters received by buffer p Failure: -1
[Note]	Printer will return the printer serial number.
[Sample Coding]	<div>&lt; Visual C &gt;</div> <pre>char rbuf[256]; if( GetSerialNo( DS820, (LPSTR)rbuf ) &gt; 0 ){     // Next process }</pre> <div>&lt; VB.NET &gt;</div> <pre>Dim s As String = New String("", 255) Dim i As Integer i = GetSerialNo(DS820, s) If i &gt; 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"</pre>

**Set USB iSerialNumber availability setting**

---

[Format]	BOOL SetUSBSerialEnable( long lPortNum,    DWORD dwEnable );	
[Parameter]	lPortNum:	Port number
	dwEnable :	Enable the USB iSerialNumber (1) Disable the USB iSerialNumber (0)
[Return]	Successful:	True
	Failure:	False
[Explanation]	Sets the USB iSerialNumber availability setting.	
	When you enable this setting, serial number is the product serial number of the product-specific information.	
	The default value for this setting is "enabled (1)". Settings are stored in the printer. (It is valid even when you turn off the printer.)	
[Attention]	This configuration change will be effective when the power of the printer is turned on again.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> if( SetUSBSerialEnable(DS820, 1) ){                    // Enable the USB iSerialNumber // Successful }	
	<b>&lt; VB.NET &gt;</b> if SetUSBSerialEnable(DS820, 0) Then                    ' Disable the USB iSerialNumber ' Successful End if	

**Get USB iSerialNumber availability setting**

---

[Format]	long GetUSBSerialEnable( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	The USB iSerialNumber is enable. (1) The USB iSerialNumber is disable. (0)
	Failure:	-1
[Explanation]	printer will return the USB iSerialNumber availability setting.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> if( GetUSBSerialEnable(DS820) == 1 ){ // use }	
	<b>&lt; VB.NET &gt;</b> if GetUSBSerialEnable (DS820) = 1 Then ' use End If	



## Set Firmware Update Mode

---

[Format]	BOOL SetFirmwUpdateMode( long lPortNum ); BOOL CvSetFirmwUpdateMode( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	True
	Failure:	False
[Note]	Changes the printer mode to firmware update mode.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> if( SetFirmwUpdateMode(DS820) > 0 ){ // Next process }	
	<b>&lt; VB.NET &gt;</b> if ( SetFirmwUpdateMode(DS820) > 0 ) Then ' Next Process End if	

## Write Firmware Data

---

[Format]	BOOL SetFirmwDataWrite( long lPortNum, LPSTR lpData, DWORD dwDataLen ); BOOL CvSetFirmwDataWrite( long lPortNum, LPSTR lpData, DWORD dwDataLen );	
[Parameter]	lPortNum:	Port number
	lpData:	Pointer to the buffer where the data is to be rewritten
	dwDataLen:	The number of characters of the data
[Return]	Successful:	True
	Failure:	False
[Note]	Sends data to rewrite firmware to the printer. The data is supplied by Motorola S format file. When this command is issued, a buffer (approx. 5M byte) is necessary for reading the file and for storing the data temporarily. After completion of data writes, the printer is rebooted automatically, and Update Mode is reset.	
	<b>&lt;About VB.Net wrapper function&gt;</b> When using VB.Net, since the pointer cannot be given directly from VB to the DLL function argument, a wrapper function for VB.Net with the same name as the DLL has been prepared. The definition is written in the DS820Stat.vb file, so add the file to the VB project to use it.	
[Sample Coding]	<b>&lt; VB.NET &gt;</b> Dim fd(5000000) Dim c as Long, n As Long  c=0 FileOpen(1, fname, OpenMode.Binary, OpenAccess.Read) FileLength = LOF(1) For n = 0 To FileLength - 1 FileGet(1, fd(c)) c = c + 1 Next n FileClose(1)  SetFirmwDataWrite( DS820, fd, c )	

## Clear Color Data

---

[Format]	BOOL SetColorDataClear( long lPortNum ); BOOL CvSetColorDataClear( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	True
	Failure:	False
[Note]	Clear color control data stored in the printer. When rewriting color control data, please first clear color control data with this command.	
[Sample Coding]	< Visual C > <pre>if(SetColorDataClear ( DS820 )){     // Color control data successfully cleared }</pre>	
	< VB.Net > <pre>If SetColorDataClear(DS820) &lt;&gt; 0 Then     ' Color control data successfully cleared EndIf</pre>	

## Write Color Data

---

[Format]	BOOL SetColorDataWrite( long lPortNum, LPSTR lpData, DWORD dwDataLen ); BOOL CvSetColorDataWrite( long lPortNum, LPSTR lpData, DWORD dwDataLen );	
[Parameter]	lPortNum:	Port number
	lpData:	Pointer to the buffer where the color data is to be rewritten
	dwDataLen:	The number of characters of the data
[Return]	Successful:	True
	Failure:	False
[Note]	Sends color data to rewrite. Before sending the new color data with this command, please first clear existing color data.	
	Color data is provided in an original format binary file, and when this command is issued, a buffer (approx. 100k byte) is necessary for reading the file and for storing the data temporarily.	
[Sample Coding]	<About VB.Net wrapper function> When using VB.Net, since the pointer cannot be given directly from VB to the DLL function argument, a wrapper function for VB.Net with the same name as the DLL has been prepared. The definition is written in the DS820Stat.vb file, so add the file to the VB project to use it.	
	< VB.Net > <pre>Dim fd(100000) Dim c as Long, n As Long  c=0 FileOpen(1, fname, OpenMode.Binary, OpenAccess.Read) FileLength = LOF(1) For n = 0 To FileLength - 1     FileGet(1, fd(c))     c = c + 1 Next n FileClose(1)  SetColorDataWrite( DS820, fd, c )</pre>	

**Set Color Data Version**

---

[Format]	BOOL SetColorDataVersion( long lPortNum, LPSTR lpData, DWORD dwDataLen ); BOOL CvSetColorDataVersion( long lPortNum, LPSTR lpData, DWORD dwDataLen );	
[Parameter]	lPortNum:	Port number
	lpData:	Pointer to the buffer where the color data version is stored
	dwDataLen:	The number of characters of the data
[Return]	Successful:	True
	Failure:	False
[Note]	Sets the color control data version. After rewriting color control data, please set the version name as the file name of the color control data supplied.  <About VB.Net wrapper function> When using VB.Net, since the pointer cannot be given directly from VB to the DLL function argument, a wrapper function for VB.Net with the same name as the DLL has been prepared. The definition is written in the DS820Stat.vb file, so add the file to the VB project to use it.	
[Sample Coding]	<pre> &lt; VB.Net &gt; Dim fname As String ' Store version name in fname SetColorDataVersion( DS820, fname, Len(fname) ) </pre>	

**Get Color Data Version**

---

[Format]	long GetColorDataVersion( long lPortNum, LPSTR p ); long CvGetColorDataVersion( long lPortNum, LPSTR p );	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Note]	Printer will return the version name in the buffer.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> char rbuf[256]; if(GetColorDataVersion( DS820, (LPSTR)rbuf ) > 0 ){ // Next process }	
	<b>&lt; VB.NET &gt;</b> Dim s As String = New String("", 255) Dim i As Integer i = GetColorDataVersion(DS820, s) If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"	

**Get Color Data Version <Type designation>**

---

[Format]	long GetColorDataVersionRes( long lPortNum, LPSTR p , DWORD dwType);	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
	dwType:	Type designation      setting value: 0,1,2,3
		Version name of the color data for 300dpi print (0)
		Version name of the color data for 600dpi print (1)
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
	Version name of the color data for slow speed print (2)	
	Version name of the color data for high density print (3)	
	Version name of the color data depends on the loaded media type.	
[Explanation]	Printer will return checksum of the version name of specified type in the buffer.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> char rbuf[256]; if(GetColorDataVersionRes ( DS820, (LPSTR)rbuf , 1 ) > 0 ){ // Get Color Data Version for 600dpi // Next process }	
	<b>&lt; VB.NET &gt;</b> Dim s As String = New String("", 255) Dim i As Integer i = GetColorDataVersionRes(DS820, s , 0) 'Get Color Data Version for 300dpi If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"	

**Get Color Data Version <Type, Media designation>**

---

[Format]	long GetColorDataVersionResEX( long lPortNum, LPSTR p, DWORD dwType, DWORD dwMedia );	
[Parameter]	lPortNum:	Port Number
	p:	Pointer to the receiving buffer
	dwType:	Type designation      setting value: 0,1,2,3 Version name of the color data for 300dpi print (0) Version name of the color data for 600dpi print (1) Version name of the color data for low speed print (2) Version name of the color data for high density print (3)
	dwMedia:	SD Media (1) PP Media (3)
[Return]	True:	The number of characters received by buffer p
	False:	-1
[Note]	Printer will return the version name in the buffer.	
[Sample Coding]	< Visual C >	
	char p[256]; if(GetColorDataVersionResEX ( DS820, (LPSTR)p, 1, 3 ) > 0 ){ // Get Version for 600dpi // process }	
	< VB.NET >	
	Dim s As String = New String("", 255) Dim i As Integer i = GetColorDataVersionResEX(DS820, s, 0, 3) ' Get Version for 300dpi If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"	

**Get Color Data Checksum**

---

[Format]	long GetColorDataChecksum( long lPortNum, LPSTR p ); long CvGetColorDataChecksum( long lPortNum, LPSTR p );	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Note]	Printer will return checksum of the color data in the buffer.	
[Sample Coding]	< Visual C >	
	char rbuf[256]; if(GetColorDataChecksum( DS820, (LPSTR)rbuf ) > 0 ){ // Next process }	
	< VB.NET >	
	Dim s As String = New String("", 255) Dim i As Integer i = GetColorDataChecksum(DS820, s) If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"	

**Get Color Data Checksum <Type designation>**

---

[Format]	long GetColorDataChecksumRes( long lPortNum, LPSTR p , DWORD dwType);	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
	dwType:	Type designation      setting value : 0,1,2,3
		Checksum of the color data for 300dpi print (0)
		Checksum of the color data for 600dpi print (1)
		Checksum of the color data for slow speed print (2)
[Return]		Checksum of the color data for high density print (3)
		Checksum of the color data depends on the loaded media type.
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Explanation]	Printer will return checksum of the color data of specified type in the buffer.	
[Sample Coding]	< Visual C >	
	<pre> long lType; char rbuf[256]; if(GetColorDataChecksumRes ( DS820, (LPSTR)rbuf , lType ) &gt; 0 ){     // Next process } </pre>	
[Sample Coding]	< Visual Basic >	
	Dim s As String = New String("", 255)	
	Dim i As Integer	
	i = GetColorDataChecksumRes (DS820, s, 1)	
	If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"	

**Get Color Data Checksum <Type, Media designation>**

---

[Format]	long GetColorDataChecksumResEX( long lPortNum, LPSTR p, DWORD dwType, DWORD dwMedia);	
[Parameter]	lPortNum:	Port number
	p:	Pointer to the receiving buffer
	dwType:	Type designation      setting value : 0,1,2,3
		Checksum of the color data for 300dpi print (0)
		Checksum of the color data for 600dpi print (1)
		Checksum of the color data for slow speed print (2)
		Checksum of the color data for high density print (3)
	dwMedia:	SD Media (1)
		PP Media (3)
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Explanation]	Printer will return checksum of the color data of specified type in the buffer.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> long lType; char rbuf[256]; if(GetColorDataChecksumResEX ( DS820, (LPSTR)rbuf , 0, 3 ) > 0 ){ // Next process }	
	<b>&lt; Visual Basic &gt;</b> Dim s As String = New String("", 255) Dim i As Integer i = GetColorDataChecksumResEX (DS820, s, 1, 3 ) If i > 0 Then Text1.Text = VB.Left(s, i) Else Text1.Text = "ERROR!"	

## Cutter Control

---

[Format]	BOOL	SetCutterMode( long lPortNum, DWORD dwMode );
[Parameter]	lPortNum: dwMode:	port number cutter mode selection
[Return]	Successful: Failure:	True False
[Explanation]	<p>This is to designate the cutter operation.  The cutter operation designation has a macro definition of DS820Stat.h (DS820Stat.vb for VB).  The symbols have the following meanings:</p> <p>CUTTER_MODE_STANDARD Standard cutter operation  CUTTER_MODE_NONSCRAP Non-scrap cutter operation  CUTTER_MODE_2INCHCUT 2inch cut operation  CUTTER_MODE_PANORAMA Panorama Print operation (with white borders between prints)</p> <p>The default value for this setting is "CUTTER_MODE_STANDARD".  This setting will return to the initial value the power is turned OFF</p>	
[Attention]	<p>The cutter control command sets the operation before the image data is sent. The command is valid for 1 image.  After performing the designated cut for the printed image, the printer will return to its standard cut operation.  2inch cut operation is effective only in paper size (8x4), (8x6), (8x8),(8x10),or (8x12).  In the following case, panorama printing supports sizes.  8x10 ⇒ 8x20 (2 sheets ) ⇒ 8x30(3 sheets)  8x12 ⇒ 8x24 (2 sheets ) ⇒ 8x36(3 sheets)  A4 ⇒ A4x2 (2 sheets ) ⇒ A4x3(3 sheets)</p> <p>In the following cases, specified by this command is disabled.</p> <ul style="list-style-type: none"> <li>▪ If two inches cut setting of the printer driver is "Yes".</li> <li>▪ When it is used full cutter setup command or full cutter setup extended command.</li> </ul>	
[Example]	<pre>&lt; Visual C &gt; if ( SetCutterMode(DS820, (DWORD)CUTTER_MODE_NONSCRAP) == True ) {     // Success }  &lt; VB.Net &gt; if SetCutterMode(DS820, CUTTER_MODE_NONSCRAP) &lt;&gt; False Then     ' Success End If</pre>	



**Full Cutter Set-up**

[Format]	BOOL	SetFullCutterSetup( long lPortNum, long lSize1, long lSize2, long lSize3, long lSize4, long lSize5, long lSize6);		
[Parameter]	lPortNum:	Port number		
	lSize1:	Image1 cut size	(setting value: 0, 20 - )	
	lSize2:	Image2 cut size	(setting value: 0, 20 - )	
	lSize3:	Image3 cut size	(setting value: 0, 20 - )	
	lSize4:	Image4 cut size	(setting value: 0, 20 - )	
	lSize5:	Image5 cut size	(setting value: 0, 20 - )	
	lSize6:	Image6 cut size	(setting value: 0, 20)	
[Return]	Successful:	True		
	Failure:	False		
[Explanation]	With the media sizes shown in the table below, it will cut to the designated sizes to make from 1 to the corresponding maximum cutting sheet number.			
	The cut size can be set in 0.1-inch increments in the 2 inches to the maximum size range. The total size should not exceed the maximum size.			

When setting the image1 to '0', the printer will be the normal size print operation.  
And ignore the image 2-4 arguments.

When the cutting sheet number is one sheet, please set '0' to the cut size of the image 2-6.  
When the cutting sheet number is two sheets, please set '0' to the cut size of the image 3-6.  
When the cutting sheet number is three sheets, please set '0' to the cut size of the image 4-6.  
When the cutting sheet number is four sheets, please set '0' to the cut size of the image 5-6.  
When the cutting sheet number is five sheets, please set '0' to the cut size of the image 6.

If you set the out-of-range cut size, it is returned "FALSE".

■ Media size and cut size setting range

Media size	Maximum cutting sheet number	Maximum size	Number of images	Cut size setting range
8x10	5 sheets	10 inches	1	20~100
			2	20~80
			3	20~60
			4	20~40
			5	20
8x12	6 sheets	12 inches	1	20~120
			2	20~100
			3	20~80
			4	20~60
			5	20~40
A4	5 sheets	11.7 inches	6	20
			1	20~117
			2	20~97
			3	20~77
			4	20~57
			5	20~37

[Attention]	Send this command before the start print command is sent.
	This command is only valid once for each image. The printer will return to the normal size print operation after each image is printed.
	If the total cut size set using this command is larger than the image size sent to the printer, this command is disabled and printing is done with the media size (normal size) sent to the printer.

When this command used, the following specification is disabled.

- cutter control command

If both the full cutter set-up expansion command and this command are used together, the last setting used will be given priority.

[Sample Coding]

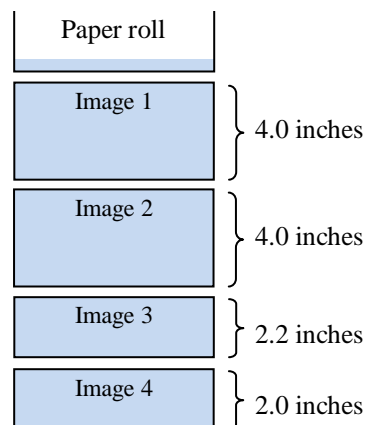
< Visual C >

```
if((SetFullCutterSetup( DS820, 40, 40, 20, 20, 0, 0 )) == True){
    // Successful
}
```

< Visual Basic >

```
If SetFullCutterSetup(DS820, 40, 40, 20, 20, 0, 0) = True Then
    ' Successful
End If
```

\* If using the sample coding, it will print the following result.



**Full Cutter Set-up Extended**

---

[Format]	BOOL	SetFullCutterSetupEX(long lPortNum, long lCutSize1, long lCutSize2, long lCutSize3, long lCutSize4, long lCutSize5, long lCutSize6, long lDustSize );	
[Parameter]	lPortNum:	Port number	
	lCutSize1:	Image1 cut size	Setting value: 0, 20-
	lCutSize2:	Image2 cut size	Setting value: 0, 20-
	lCutSize3:	Image3 cut size	Setting value: 0, 20-
	lCutSize4:	Image4 cut size	Setting value: 0, 20-
	lCutSize5:	Image5 cut size	Setting value: 0, 20-
	lCutSize6:	Image6 cut size	Setting value: 0, 20
	lDustSize:	Intermediate scrap cut size	Setting value: 0, 12-22
[Return]	Successful:	True	
	Failure:	False	
[Explanation]	<p>Full Cutter Set-up Extended command that makes it possible to set the size of an intermediate scrap cut. The scrap size can be set in 0.01-inch increments in the 0.12 to 0.22 inches range. When setting the intermediate scrap cut size to a valid value other than 0, pay attention that the total cut size calculated with the formula below does not exceed the media maximum size.</p> <p style="text-align: center;">Total cut size = (Total of each image cut size) + ((Intermediate scrap cut size/10) x (Number of images-1))</p> <p>Example: The following formula is obtained when the media size is 8x12 inches (120), there are 5 images of 2 inches each (setting value = 20), and the intermediate scrap is 0.22 inches (setting value = 22).</p> <p style="text-align: center;">(20+20+20+20+20+0)+((22/10)x(5-1)) = 108.8 ≤ 120</p> <p>When intermediate scrap cut size is set to 0, refer to the table in Full Cutter Set-up for the media size and cut size setting ranges for each image.</p> <p>If values outside the setting range above are set for the cut sizes, “FALSE” is returned.</p>		
[Attention]	<p>If the total cut size set using this command is larger than the image size sent to the printer, this command is disabled and printing is done with the media size (normal size) sent to the printer.</p> <p>When this command used, the following specification is disabled.</p> <ul style="list-style-type: none"><li>▪ cutter control command</li></ul> <p>If both the full cutter set-up command and this command are used together, the last setting used will be given priority.</p>		

[Sample Coding]

&lt; Visual C &gt;

```

if((SetFullCutterSetUpEX( DS820, 20, 20, 20, 20, 20, 0, 12 )) == True){
    //Success
}

```

&lt; Visual Basic &gt;

Dim Result As Long

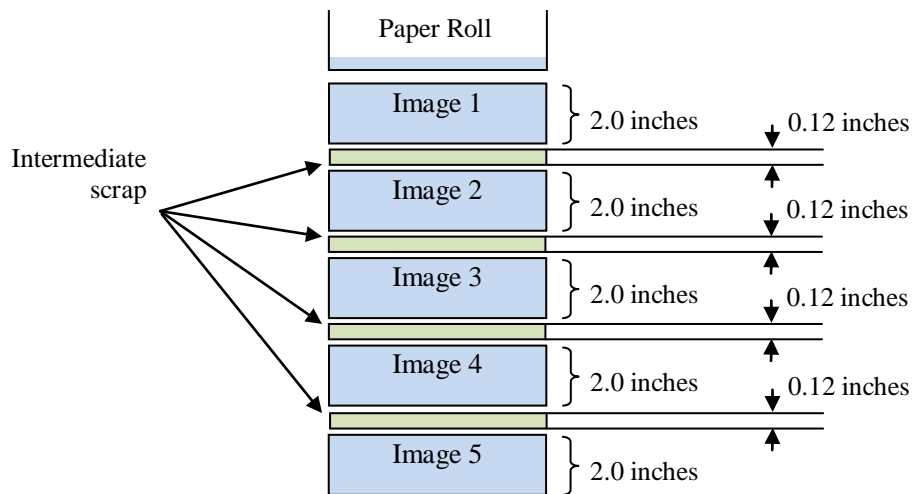
Result = SetFullCutterSetUpEX (DS820, 20, 20, 20, 20, 20, 0, 12)

If Result = True Then

'Success

End If

\* If using the sample coding, it will print the following result..



**Set Print Speed**

---

[Format]	BOOL SetPrintSpeed(long lPortNum, long lPrintSpeed);	
[Parameter]	lPortNum:	Port Number
	lPrintSpeed :	Print Speed
		0: Normal printing (Default)
		1: Reserved
		2: Reserved
[Return]		3: High density printing
	Successful:	True
	Failure:	False
[Explanation]	Printing is carried out at the designated print speed.	
[Attention]	The printer will revert to the conventional operation after each image is printed.	
	This command must be designated for each image.	
[Sample Coding]	< Visual C >	
	if( SetPrintSpeed (DS820, 3) ){ // High density printing	
	// Successful	
	}	
	< VB.NET >	
	if SetPrintSpeed (DS820, 0) Then // Normal printing	
	' Successful	
	End if	

## Set Standby Mode Transition Time

---

[Format]	BOOL	SetStandbyTime( long lPortNum, long lTime );
[Parameter]	lPortNum: lTime:	Port number Standby mode transition time (Setting value:0, 1 - 99)
[Return]	Successful: Failure:	True False
[Explanation]	<p>Set the time to transition to standby mode in 1 minute increments. (Up to 99 minutes maximum) When the idle state continues for a set time, it transitions to the standby mode.</p> <p>It becomes the specification the printer does not transition to the standby mode, if you set 00. If you set the out-of-range transition time, it is returned "FALSE".</p> <p>The default value for this setting is 10-minute. Settings are stored in the printer. (It is valid even when you turn off the printer.)</p>	
[Attention]	When changing to a time shorter than 10-minute, please check the information listed in the “Power Consumption” section of the Printer Product Specification Manual before proceeding.	
[Sample Coding]	<p>&lt; Visual C &gt;</p> <pre>if((SetStandbyTime( DS820, 0 )) == FALSE ){     //Error }</pre> <p>&lt; Visual Basic &gt;</p> <pre>Dim Result As Long Result = SetStandbyTime(DS820, 99) If Result = FALSE Then GoTo Error</pre>	

## Get Standby Mode Transition Time

---

[Format]	long GetStandbyTime( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful: Failure:	Standby mode transition time -1
[Explanation]	printer will return the Standby mode transition time.	
[Sample Coding]	<p>&lt; Visual C &gt;</p> <pre>long time; if(( time = GetStandbyTime( DS820 )) &gt;= 0 ){     // Next process }</pre> <p>&lt; VB.NET &gt;</p> <pre>Dim time As Integer time = GetStandbyTime(DS820)</pre>	

## Set Media End Keep Mode

---

[Format]	BOOL	SetEndKeepMode( long lPortNum、 long lMode );
[Parameter]	lPortNum:	Port number
	lMode:	Keep Mode Setting (Setting value:0, 1)
[Return]	Successful:	True
	Failure:	False
[Explanation]	<p>After the Media End has occurred, when you open and close the door or you restart the printer, it is set to determine whether or not to hold the Media End.</p> <p>Media End (ribbon end, paper end) and Media Errors (Ribbon error, paper jam) are eligible.</p> <p>When you set '0' to the keep mode setting, it becomes the media end non keep mode.</p> <p>When you set '1' to the keep mode setting, it becomes the media end keep mode.</p> <p>When you set the out-of-range keep mode, it is returned "FALSE".</p> <p>The factory preset value for this setting is "keep mode(1)".</p> <p>Settings are stored in the printer. (It is valid even when you turn off the printer.)</p>	
[Sample Coding]	<b>&lt; Visual C &gt;</b> <pre>if((SetEndKeepMode ( DS820, 0 )) == FALSE ){     //Error }</pre>	
	<b>&lt; VB.NET &gt;</b> <pre>Dim Result As Integer Result = SetEndKeepMode (DS820, 1) If Result = FALSE Then GoTo Error</pre>	

## Get Media End Keep Mode

---

[Format]	long	GetEndKeepMode ( long lPortNum );
[Parameter]	lPortNum:	Port number
[Return]	Successful:	Keep Mode Setting
	Failure:	-1
[Explanation]	printer will return the Media End Keep Mode setting value. ( 1(Keep) or 0(Non keep) )	
[Sample Coding]	<b>&lt; Visual C &gt;</b> <pre>long mode; if((mode = GetEndKeepMode ( DS820 )) &gt;= 0 ){     // Next process }</pre>	
	<b>&lt; VB.NET &gt;</b> <pre>Dim mode As Integer mode = GetEndKeepMode (DS820)</pre>	

## Panorama Printing Start Check

---

[Format]                    long            GetPanoramaPrintable( long lPortNum );

[Parameter]            lPortNum:                Port number

[Return]                Successful:                Printer status indicating whether or not panorama printing is possible  
 Failure:                 STATUS\_COMM\_ERROR

[Explanation]           This command returns the printer status indicating whether or not panorama printing is possible.  
 The values for each status are defined in DS820Stat.h by macro.(DS820Stat.vb for VB.NET)  
 Meanings of the symbols are as follows:

Return value	Status
STATUS_PRINTABLE	Panorama printing can be started.
STATUS_HIGHTEMP_HEAD	High head temperature.
STATUS_HIGHHUMIDITY *1	High humidity.
STATUS_LOWTEMP_MEDIA *2	Low media temperature.
STATUS_OTHER_STATE	Other status (printing, cooling, error, etc.) Details can be obtained with the Get Printer Status command.
STATUS_COMM_ERROR	Communication error

\*1: Supported by DP-DS820 firmware version 01.01 or later.

\*2: Supported by DP-DS820 firmware version 01.03 or later.

[Attention]              •When the head temperature is high, send the panoramic prints data without waiting for the status to change to show printing is possible. (The same as for normal printing, it will start cooling, and printing will start when the head temperature has dropped.)

•When the humidity is high, the print quality of the panoramic prints may deteriorate, so we don't recommend printing in high-humidity situations.

•When the media temperature is low, don't perform the continuous panoramic prints. (It may occur the problem such as a paper jam according to the image.)

(For details regarding panorama printing with a white border, please refer to "Panorama\_withSpace\_E.doc")

(Please refer to the continuous panorama SDK for continuous panorama printing)



[Sample Coding]

## &lt; Visual C &gt;

```

long      stat;
stat = GetPanoramaPrintable( DS820 );
if(stat == STATUS_LOWTEMP_MEDIA){
    // please do not run the continuous panoramic prints
}else if(stat == STATUS_HIGHHUMIDITY){
    // run of the continuous panoramic prints is not recommended
}else if((stat == STATUS_PRINTABLE) || (stat == STATUS_HIGHTEMP_HEAD)){
    // please start the panoramic prints
}else{
    // checking printer status, and others
}

```

## &lt; VB.NET &gt;

```

Dim stat As Integer
stat = GetPanoramaPrintable( DS820 );
If stat = STATUS_LOWTEMP_MEDIA Then
    ' please do not run the continuous panoramic prints
Else If stat = STATUS_HIGHHUMIDITY Then
    ' run of the continuous panoramic prints is not recommended
Else If (stat = STATUS_PRINTABLE) Or (stat = STATUS_HIGHTEMP_HEAD) Then
    ' please start the panoramic prints
Else
    ' checking printer status, and others
End If

```

**Continuous Panorama Prints settings**

---

[Format]	BOOL SetContPanorama( long lPortNum, DWORD dwContinuous, DWORD dwOverlap );	
[Parameter]	lPortNum:	Port number
	dwContinuous:	Continuous Panoramic Prints specification.
	dwOverlap:	Overlap width.
[Return]	Successful:	True
	Failure:	False
[Explanation]	This function sets the operation of continuous panoramic prints. (Please refer to the continuous panorama SDK for continuous panorama printing)	
	The continuous panoramic prints specification has a macro definition of DS820Stat.h. (DS820Stat.vb for VB.NET) The symbols have the following meanings:	
	CONT_PANORAMA_PRINT:	Continuous Panoramic Prints has specified.
	CONT_PANORAMA_LAST:	Last image of Continuous Panoramic Prints, or normal printing. (Continuous Panoramic Prints cancelation)
[Attention]	Continuous Panoramic Prints specification is only valid for the first image and second image. If you set this parameter for the third image, the specified parameter will be ignored, and the paper will be cut.	
	The overlap width for the first image printed with Continuous Panoramic Prints is also applied to the second image and later on, but is disabled if a different value is set for the second or later images. Also, if you set this value to less than the recommended value, the print quality of the overlapped area will deteriorate.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> <pre>if( SetContPanorama( DS820, CONT_PANORAMA_PRINT, 200 ) { // Continuous Panoramic Prints settings     // Successful }</pre>	
	<b>&lt; VB.NET &gt;</b> <pre>If SetContPanorama( DS820, CONT_PANORAMA_PRINT, 200 ) Then 'Continuous Panoramic Prints settings     ' Successful End If</pre>	

**Overcoat Finishing Control**

---

[Format]	BOOL SetOvercoatFinish ( long lPortNum, DWORD ovcoat );	
[Parameter]	lPortNum:	Port number
	ovcoat:	Overcoat finishing mode selection
[Return]	Successful:	True
	Failure:	False
[Explanation]	This is to designate the overcoat finishing operation.	
	The overcoat finishing mode designation has a macro definition of DS820Stat.h.(DS820Stat.vb for VB.NET)	
	The symbols have the following meanings:	
	OVERCOAT_FINISH_GLOSSY	glossy
	OVERCOAT_FINISH_MATTE1	matte 1
	OVERCOAT_FINISH_MATTE2	matte 2 (reserve)
	OVERCOAT_FINISH_MATTE3	matte 3 (reserve)
	OVERCOAT_FINISH_FINEMATTE	fine matte
	OVERCOAT_FINISH_LUSTER	luster
	OVERCOAT_FINISH_PMATTE11	partial matte (matte)
	OVERCOAT_FINISH_PMATTE12	partial matte (fine-matte)
	OVERCOAT_FINISH_PMATTE13	partial matte (luster)
	*The initial value is "OVERCOAT_FINISH_GLOSSY".	
[Attention]	The overcoat finishing control command sets the operation before the image data is sent.	
	The command is valid for 1 image.	
	After performing the designated overcoat finishing for the printed image, the printer will return to its glossy finishing operation.	
	When this command used, the following specification is disabled.	
	• The printer driver overcoat finish setting is other than "glossy".	
[Sample Coding]	< Visual C >	
	<pre> if ( SetOvercoatFinish (DS820, OVERCOAT_FINISH_MATTE) == True ) {     // Success } </pre>	
	< VB.NET >	
	<pre> If SetOvercoatFinish (DS820, OVERCOAT_FINISH_MATTE) = True Then     ' Success End If </pre>	

**Set Gamma Table**

[Format]                BOOL SetGammaTable( long lPortNum, LPBYTE lpGammaTable, long lDataLen );

[Parameter]           lPortNum:            Port number  
                          lpGammaTable:    Pointer to the buffer where the gamma table data  
                          lDataLen:           Gamma table data size.(Unit:Byte)

[Return]                Successful:            True  
                          Failure:                False

[Explanation]        Sets the gamma table data.  
                          For information on the format of the gamma table data, please refer to the table below.

Type	Item	Explanation
Header (8byte)	Data unit size	0002h fixed
	Row	0100h fixed
	Plane	0004h fixed
	Checksum	xxxxh (sum value of expansion and gamma table data)
Expansion Data (8byte)	Mode	0000h fixed
	Rsv1	0000h fixed
	Rsv2	0000h fixed
	Rsv3	0000h fixed
Gamma Table Data (2048byte)	Y[0]...Y[255] M[0]...M[255] C[0]...C[255] OP[0]...OP[255]	0000h~FFFFh

[Attention]            Because the gamma table data is lost when the printer is turned OFF or reverts to stand-by mode, it should be sent just before printing.

[Sample Coding]     **< Visual C >**  
 BYTE tableData[2064];  
 ...  
 if(SetGammaTable( DS820, tableData, 2064 ) > 0 ){  
                          // successful  
 }  
  
**< VB.NET >**  
 Dim tableData(2064) As Byte  
 ...  
 if SetGammaTable( DS820, tableData, 2064 ) > 0 Then  
                          ' successful  
 End If

**Get Gamma Table Checksum**

---

[Format]	long GetGammaTableChecksum( long lPortNum, LPSTR p );	
[Parameter]	lPortNum:	Port number
	lpRcvBuf:	Pointer to the receiving buffer
[Return]	Successful:	The number of characters received by buffer p
	Failure:	-1
[Explanation]	Printer will return checksum of the gamma table data in the buffer. However, if the gamma table data is not set at the printer, it will receive a 4-character space character string "    " (CR<0Dh> terminus, 4-byte padding).	
[Sample Coding]	<b>&lt; Visual C &gt;</b> char sum[256]; if(GetGammaTableChecksum ( DS820, sum ) > 0 ){ // successful }	
	<b>&lt; VB.NET &gt;</b> Dim sum As String = New String(" ", 256) if GetGammaTableChecksum ( DS820, sum ) > 0 Then ' successful End If	

**Clear Data Table**

---

[Format]	BOOL SetDataTableClear( long lPortNum );	
[Parameter]	lPortNum:	Port number
[Return]	Successful:	True
	Failure:	False
[Explanation]	Clear the data of gamma table stored in the printer.	
[Sample Coding]	<b>&lt; Visual C &gt;</b> if(SetDataTableClear( DS820 )){ // successful }	
	<b>&lt; VB.NET &gt;</b> if SetDataTableClear( DS820 ) Then ' successful End If	

## Common Set Command

---

[Format]	BOOL SetCommand( long lPortNum, LPSTR lpCmd, DWORD dwCmdLen ); BOOL CvSetCommand( long lPortNum, LPSTR lpCmd, DWORD dwCmdLen );	
[Parameter]	lPortNum:	Port number
	lpCmd:	Pointer to the buffer where the command is stored
	dwCmdLen:	The number of characters of the command
[Return]	Successful:	True
	Failure:	False
[Note]	Sends the command to the printer  <About VB.Net wrapper function> When using VB.Net, since the pointer cannot be given directly from VB to the DLL function argument, a wrapper function for VB.Net with the same name as the DLL has been prepared. The definition is written in the DS820Stat.vb file, so add the file to the VB project to use it.	

## Common Get Command

---

[Format]	long GetCommandEX( long lPortNum, LPSTR lpCmd, DWORD dwCmdLen, LPSTR lpRetBuff, DWORD dwRetBuffSize ); long CvGetCommandEX( long lPortNum, LPSTR lpCmd, DWORD dwCmdLen, LPSTR lpRetBuff, DWORD dwRetBuffSize );	
[Parameter]	lPortNum:	Port number
	lpCmd:	Pointer to the buffer where the command is stored
	dwCmdLen:	The number of characters of the command
	lpRetBuff:	Pointer to the buffer to store receipt data
	dwRetBuffSize:	Available size of receiving buffer
[Return]	Successful:	The number of bytes received (Receipt data by RetBuff)
	Failure:	-1
[Note]	After the command is sent to the printer, receipt data is stored in the buffer.  <About VB.Net wrapper function> When using VB.Net, since the pointer cannot be given directly from VB to the DLL function argument, a wrapper function for VB.Net with the same name as the DLL has been prepared. The definition is written in the DS820Stat.vb file, so add the file to the VB project to use it.	

## Note for Sample Program

The Status API sample program is a program made with VB.Net, which gets the printer information using the Printer Status API. How to use it is explained below.

If you run the sample program, the screen will appear. By clicking the buttons, you can get each type of information. An example of when you click the LifeCounter button is shown below.

When the LifeCounter button is clicked, the program below will be run, and with the status API function GetCounterL() the life counter value will be retrieved from the printer and displayed on the screen.

```
***** Get Life Counter
Private Sub Command7_Click()
Dim c As Long

    c = GetCounterL(DS820)
    If c >= 0 Then Text2.Text = Str(c) Else Text2.Text = "ERROR!"

End Sub
```

