Norden Logic

ESC POS	Command Set ESC/POS	**Programming API**
Rev. 1.0.13	Date: 27.01.2026	

Norden Logic

Product History		
Version	Rev. Date	Description
1.0	September 2019	First command set release
1.0.4	November 2019	New commands added
1.0.5	January 2020	New commands plus cutter support
1.0.6	April 2020	New Languages enabled
1.0.7	August 2020	New commands: QR code, NVBitmap flash, Real-Time status callback, new Font combination attributes (ESC ! n)
1.0.8	November 2024	New command: Power management Updated: serial baud rates Requires firmware 1.22 or higher
1.0.9	December 2024	Added M23 print head types
1.0.10	August 2025	Max gray level changed to 20 Korean language enabled Default code page changed to CP437 XON/XOFF flow control improved
1.0.11	November 2025	Added Seiko LTPV345 head types Added PT486HLV head types New commands for uploading and using additional or custom fonts
1.0.12	December 2025	QR encoding function updated
1.0.13	January 2026	Added label printer JX2R39BQ Added 80mm with cutter PT72EP New commands for label printing

2.2.1 LF 9
2.2.2 CR 9
2.2.3 ESC J 10
2.2.4 ESC d 11
2.2.5 ESC 3 11
2.2.6 ESC 2 12
2.2.7 ESC l 13
2.2.8 ESC Q 15
2.2.9 ESC $ 16
2.2.10 ESC ! 17
2.2.11 ESC a 18
2.2.12 ESC m 19
2.2.13 FS & 20
2.2.14 FS . 20
2.2.15 International 8-bit and 16-bit Unicode 21
2.2.16 DoubleByte Character Sets 23
2.2.17 ESC * 23
2.2.18 GS v 0 25
2.2.19 HT 27
2.2.20 ESC D 27
2.2.21 GS h 29
2.2.22 GS w 30
2.2.23 GS k 31
2.2.24 ESC @ 32
2.2.25 GS (33
2.2.26 ESC M 34
2.2.27 DLE DC4 35
2.2.28 DLE EOT 36
2.2.29 DLE ENQ 37
2.2.30 ESC Z 38

2.2.31 GS H 38
2.2.32 ESC i 39
2.2.33 ESC n 40
2.2.34 GS a 41
2.2.35 FS q 42
2.2.36 FS p 44
2.2.37 GS (k Size/Version 45
2.2.38 GS (k Error Level 46
2.2.39 GS (k Encode Buffer 47
2.2.40 GS (k Print QR 49
2.2.41 DLE SYN 50
2.2.42 FS u Font Bin File 51
2.2.43 ESC t 52
2.2.44 GS ENQ SOH 0 Buffer 53
2.2.45 GS ACK 54
2.2.46 GS ENQ EOT n 55
3.1 Standard Language Fonts 55

Norden Logic

1. Product Description

The Norden Logic range NL02x of new flexible and powerful ARM based printer controllers support the following ESC/POS commands.

Norden Logic

2. Command Reference

2.1 ESC/POS Command Overview

ASCII	Hex format	Explanation
LF	0A	Print and feed paper
CR	0D	Carriage return
ESC J	1B 4A n	Print and feed paper n dots
ESC d	1B 64 n	Print and feed paper n lines
ESC 3	1B 33 n	Set line spacing to n dots
ESC 2	1B 32	Set line spacing to default values
ESC l	1B 6C n	Set left margin
ESC Q	1B 51 n	Set right margin
ESC $	1B 24 nL nH	Set absolute print position
ESC !	1B 21 n	Set character printing mode
ESC a	1B 61 n	Set print alignment
ESC m	1B 6D n	Set font grayscale
FS &	1C 26	Select double-byte character mode
FS .	1C 2E	Cancel double-byte character mode
International 8-bit and 16-bit Unicode		Printing international character sets
DoubleByte Character Sets		Printing DoubleByte character sets

ESC *	1B 2A m xL xH d [0] ... d [k]	Select bit-image mode
GS v 0	1D 76 30 m xL xH yL yH d [0] ... d [k]	Print raster image
HT	09	Move to next horizontal tab
ESC D	1B 44 d [0] ... d [k] 0	Set the horizontal tab positions
GS h	1D 68 n	Set barcode height
GS w	1D 77 n	Set barcode width
GS k	1D 6B mn d [0] ... d [k]	Print barcode
ESC @	1B 40	Initialize printer
GS (	1D 28 n m	Set serial communications parameters
ESC M	1B 4D	Set font size
DLE DC4	10 14	Clear print buffer
DLE EOT	10 04	Query printer status
DLE ENQ	10 05	Query firmware version
ESC Z	1B 5A n	Set Print Head Type
GS H	1D 48 n	Show or Hide Bar Code Text
ESC i	1B 69	Full Cut (NL024)
ESC n	1B 6E	Partial Cut (NL024)
GS a	1D 61 n	Set/Cancel status callback
FS q	1C 71 n 0 xL xH yL yH d [0] ... d [k]	Save NVbitmap to flash
FS p	1C 70 n m	Print NVbitmap from flash

GS (k Size/Version	1D 28 6B pL pH 31 43 n	Set QR code size/version
GS (k Error Level	1D 28 6B pL pH 31 45 n	Set QR code error correct level
GS (k Encode Buffer	1D 28 6B pL pH 31 50 30 d1...dk	QR code transfer data to encode buffer
GS (k Print QR	1D 28 6B pL pH 31 51 30	Print QR code from encode buffer
DLE SYN	10 16	Put controller into sleep mode (FW 1.22 or higher)
FS u	1C 75 n 0 lenH lenL d0..dk	Upload font bin file - up to 8 font libs
ESC t	1B 74 n	Set active uploaded font with ID
GS ENQ SOH 0 Buffer	1D 05 01 30 lenL lenH d0…dn	Save label byte data in 2k byte buffer
GS ACK	1D 06	Go to next label home position
GS ENQ EOT n	1D 05 04 n	Print n amounts of labels from label buffer

2.2 Commands

2.2.1 LF

Command	Print and feed paper
Codes	ASCII: LF DEC: 10 HEX: 0A
Description	Print the data in the printer buffer, then feed paper for one line according to the current line space settings. After printing, the print position moves to the beginning of the line.
Parameters	
Defaults	
Notes	

Example	* **********
	* Print new line
	* ********** */
	void printLF (void)
	{
	uint8_t buf[1] = {0x0A}; //LF
	Write(buf,1);
	}

2.2.2 CR

Command	Carriage return
Codes	ASCII: CR DEC: 13 HEX: 0D
Description	Adjust the print position to the starting position of this line without line feed
Parameters	

Defaults	
Notes	

Example	* **
	* Print move to starting position
	* ** */
	void printCR (void)
	{
	uint8_t buf[1] = {0x0D}; //CR
	Write(buf,1);
	}

2.2.3 ESC J

Command	Print and feed paper n dots
Codes	ASCII: ESC J n DEC: 27 74 n HEX: 1B 4A n
Description	Will print the contents of the print buffer feeds the paper n points.
Parameters	0≤ n ≤ 255
Defaults	
Notes	When the print buffer is empty, only the n feed points (0.125mm / point). After printing is completed, this command sets the print starting position to the beginning of the line.

Example	/* **
	* Description
	* *** */
	void printAndFeedNDot (uint8_t ucDotNum)
	{
	uint8_t buf[3] = {0x1B,0x4A,0x00}; //ESC J n
	buf[2] = ucDotLineNum;
	Write(buf,3);
	}

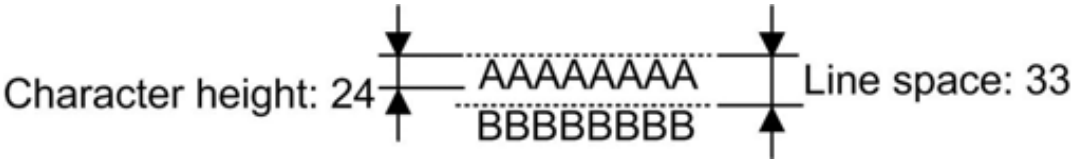
2.2.4 ESC d

Command	Print and feed paper n lines
Codes	ASCII: ESC d n DEC: 27 100 n HEX: 1B 64 n
Description	We will print the contents of the print buffer and feeds the paper n lines.
Parameters	0≤ n ≤ 255
Defaults	
Notes	When the printer buffer is empty the paper will feed for n lines but not print. The line spacing is set by ESC 2 or ESC 3. After printing, the print position moves to the beginning of the line.

Example	void printAndFeedNFontLine (unsigned char ucFontLineNum)
	{
	unsigned char buf[3]= {0x1B,0x64,0x00};//ESC d n
	buf[2] = ucFontLineNum;
	Write(buf,3);
	}

2.2.5 ESC 3

Command	Set line spacing to n dots

Codes	ASCII: ESC 3 n DEC: 27 51 n HEX: 1B 33 n
Description	Set line spacing to n dots
Parameters	0≤ n ≤ 255
Defaults	n=33
Notes	Line space is shown as follows: If the maximum character height exceeds the specified line space in a line, the line spacing will be automatically set to that maximum height. The line space will be reset to the default value 33 dots, if ESC 2 is executed, ESC @ command is executed, printer is reset or printer is turned off 

Example	/
	**********
	** Function name: lineSpacingSet
	** Descriptions: This function is used to set the line space.
	** input parameters:
	ucDotLineNum: dots of line space; n = 0 ~ 255, default value: n = 33.
	** Returned value: none
	**********/
	void lineSpacingSet (uint8_t ucDotLineNum)
	{
	uint8_t buf[3] = {0x1B,0x33,0x00};//ESC 3 n
	buf[2] = ucDotLineNum;
	Write(buf,3);
	}

2.2.6 ESC 2

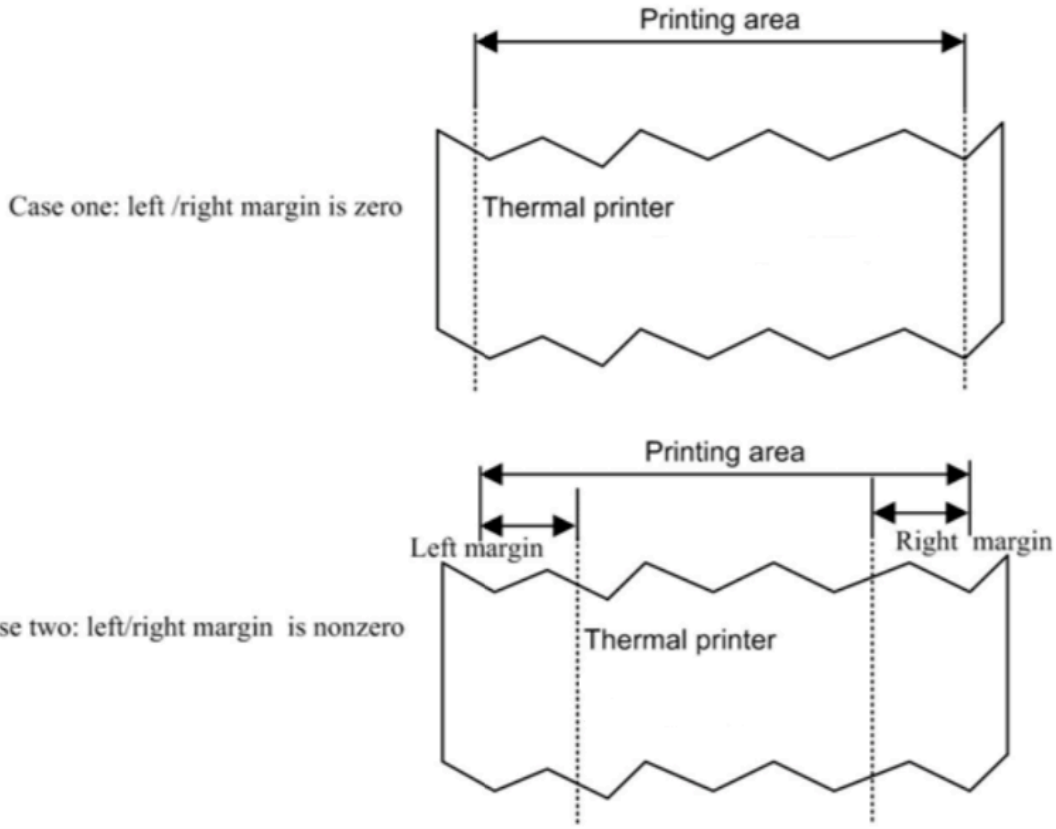
Command	Set line spacing default values

Codes	ASCII: ESC 2 DEC: 27 50 HEX: 1B 32
Description	Set line spacing to default values
Parameters	
Defaults	
Notes	

Example	/
	**********
	** Function name: lineSpacingDefaults
	** Descriptions: This function is used to set the line space.
	** Returned value: none
	**********/
	void lineSpacingDefaults (void)
	{
	uint8_t buf[2] = {0x1B,0x32};//ESC 2
	Write(buf,2);
	}

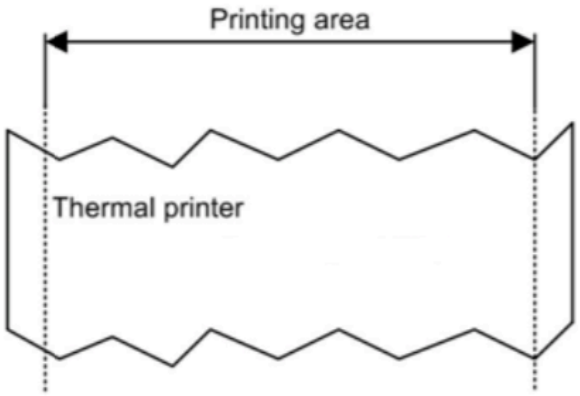
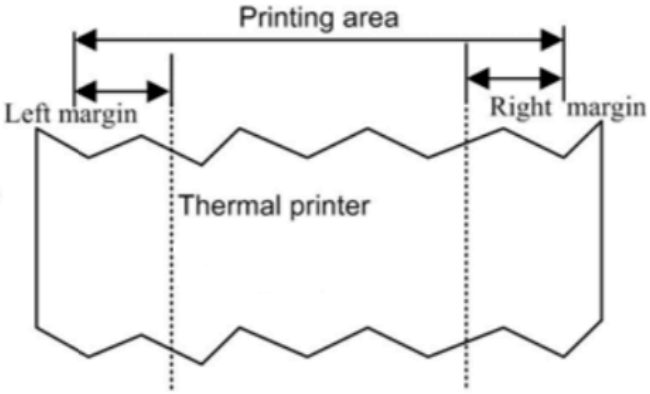
2.2.7 ESC l

Command	Set left margin
Codes	ASCII: ESC l n DEC: 27 108 n HEX: 1B 6C n
Description	Allows for setting a left margin. For example 8 would start printing 8mm from the left paper printing area. **For aligning bar codes use this function. For example if the bar code is desired to be more centred. *After setting this function for bar codes make sure to set the left margin back to 0.*** The margin settings are effective until ESC @ command is executed, printer is reset or printer is turned off

Parameters	58mm print heads: 0≤n≤47, and 0≤(left margin + right margin)≤47 80mm print heads: 0≤n≤71, and 0≤(left margin + right margin)≤71
Defaults	n = 0
Notes	The left margin position indicates the left edge position of the printing range. 

Example	void setLeftMargin (uint8_t ucDots)
	{
	uint8_t buf[3] = {0x1B,0x6C,0x00};//ESC l n
	buf[2] = ucDots;
	Write(buf,3);
	}

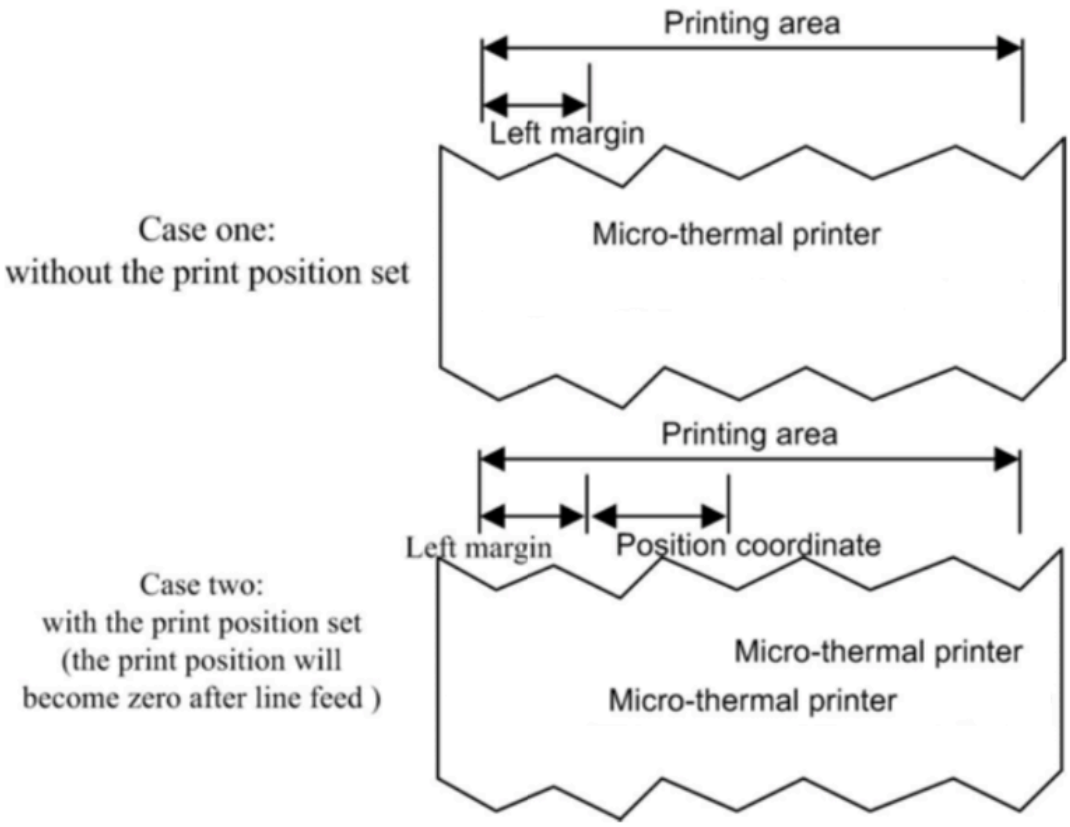
2.2.8 ESC Q

Command	Set right margin
Codes	ASCII: ESC Q n DEC: 27 81 n HEX: 1B 51 n
Description	Allows for setting a right margin. For example 8 would start printing 8mm from the right side paper printing area. The margin settings are effective until ESC @ command is executed, printer is reset or printer is turned off.
Parameters	58mm print heads: 0≤n≤48, and 0≤(left margin + right margin)≤48 80mm print heads: 0≤n≤72, and 0≤(left margin + right margin)≤72
Defaults	n = 48 for 58mm n = 72 for 80mm
Notes	The right margin position indicates the right edge of the printing range. Case one: left /right margin is zero  Case two: left/right margin is nonzero 

Example	void setRightMargin (uint8_t ucDots)
	{
	uint8_t buf[3] = {0x1B,0x51,0x00};//ESC Q n
	buf[2] = ucDots;
	Write(buf,3);
	}

2.2.9 ESC $

Command	Set absolute print position
Codes	ASCII: ESC $ nL nH DEC: 27 36 nL nH HEX: 1B 24 nL nH
Description	Moves the print position to a location in a distance of (nL + nH × 256) dots from the starting position for printing. This command only affects one line. The print position is the starting position of printing again after line feed.
Parameters	0≤nL≤255, 0≤nH≤255
Defaults	

Notes	The starting position for printing will be the left margin 

Example	/
	**********
	** Function name: absolutePrintPos
	** input parameters:
	ucLNum and ucHNum: 0≤nL≤255, 0≤nH≤255
	** Returned value: none
	**********/
	void absolutePrintPos (uint8_t ucLNum,uint8_t ucHNum)
	{
	uint8_t buf[4] = {0x1B,0x24,0x00,0x00};//ESC $ nL nH
	buf[2] = ucLNum;
	buf[3] = ucHNum;
	Write(buf,4);
	}

2.2.10 ESC !

Command	Set character printing mode

Codes	ASCII: ESC ! n DEC: 27 33 n HEX: 1B 21 n
Description	Supported modes are: normal, bold, double width, double height, inverse, underlined, italic and combination of these attributes using the bit values and OR bit operator
Parameters	n = xxxx xxxx -> total 8 bits Bit Description 0 normal 1 italic 2 reserve 3 bold 4 double height 5 double width 6 reverse 7 underline
Defaults	n = 0
Notes	This command now supports font attribute combination. Requires firmware June 2020 or newer. The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void fontTypeSet (uint8_t ucFontType)
	{
	uint8_t buf[3] = {0x1B,0x21,0x00};//ESC ! n
	buf[2] = ucFontType; //0x02l0x80 (italic + underline)
	Write(buf,3);
	}

2.2.11 ESC a

Command	Set print alignment
Codes	ASCII: ESC a n DEC: 27 97 n HEX: 1B 61 n

Description	Sets the print alignment left, center and right. Valid for all items about to be printed
Parameters	n = 0 Left n = 1 Center n = 2 Right
Defaults	n = 0
Notes	The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void setPrintAlignment (uint8_t ucType)
	{
	uint8_t buf[3] = {0x1B,0x61,0x00};//ESC a n
	buf[2] = ucType;
	Write(buf,3);
	}

2.2.12 ESC m

Command	Set font grayscale
Codes	ASCII: ESC m n DEC: 27 109 n HEX: 1B 6D n
Description	There are 20 levels supported (0 to 20) to satisfy different color depth requirements for different thermal paper, where ”0” is the lightest and “20” is the darkest.
Parameters	0≤ n ≤ 20
Defaults	n = 4
Notes	The smaller the gray level value the faster the print. However, a good combination should be found for each combination of paper, power supply and print head. The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void setFontGrayScale (uint8_t ucScale)
	{
	uint8_t buf[3] = {0x1B,0x6D,0x00};//ESC m n
	buf[2] = ucScale;
	Write(buf,3);
	}

2.2.13 FS &

Command	Select double-byte character mode
Codes	ASCII: FS & DEC: 28 38 HEX: 1C 26
Description	Sets the font mode into double-byte language character sets. E.g. Chinese GBK and Japanese JISx0208
Parameters	
Defaults	
Notes	The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void setDbMode (void)
	{
	uint8_t buf[2] = {0x1C,0x26};//FS &
	Write(buf,2);
	}

2.2.14 FS .

Command	Cancel double-byte character mode
Codes	ASCII: FS . DEC: 28 46 HEX: 1C 2E

Description	Cancels double-byte language character set mode.
Parameters	
Defaults	
Notes	After execution of this command all characters are again ASCII and Unicode - 8 bit, UTF8, UTF16. The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void cancelDbMode (void)
	{
	uint8_t buf[2] = {0x1C,0x2E};//FS .
	Write(buf,2);
	}

2.2.15 International 8-bit and 16-bit Unicode

Command	Printing international character set
Codes	
Description	The NL02x controllers make it very easy to work with international character sets. No need to set charter set mode and code pages. You can write directly the 8-bit character or the 16-bit Unicode to the controller to get the desired character. We have stored a huge number of language fonts in the controller's flash. There is also space for factory installed custom language fonts if needed.

Parameters	Basic Latin and Latin-1 in range of 0x00h - 0xFFh can be send as single bite Unicode Range: Basic latin (unicode >= 0x0020UL && unicode <= 0x007FUL) Part of latin1 & latin extended A (unicode >= 0x00A0UL && unicode <= 0x017FUL) Part of Latin extended B unicode >= 0x01A0UL && unicode <= 0x01CFUL) Part of Latin extended B (unicode >= 0x01F0UL && unicode <= 0x01FFUL) Part of Latin extended B (unicode >= 0x0210UL && unicode <= 0x021FUL) Part of Latin Extended Additional (unicode >= 0x1EA0UL && unicode <= 0x1EF9UL)
Defaults	
Notes	**The controller supports over 170 languages**. If the needed character set is not in this list please contact us. Please see Appendix 3.1 for language fonts listings.

Example	printStr("[00h-7Fh]");
	printLF();
	printStr("ABCDabcd1234%&/!)(.-_;\x80\x24\x40\xa3\xa5\xa9\xae");
	printLF();
	unsigned char
	buf2[]="\x01\53\x01\x35\x01\x07\x01\x26\x01\x76\x01\x08\x01\x19\x01\x0e\x01\x7d\x01\x4e\x01\x5c\x01\x7f\x01\x51\x01\x62\x01\x32\x01\x4a\x01\x6f";
	printStr("[100h-17Fh]");
	printLF();
	printStr((const char*)buf2);
	printLF();

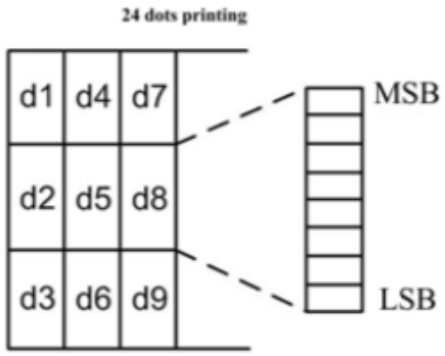
2.2.16 DoubleByte Character Sets

Command	Printing DoubleByte character sets
Codes	
Description	Printing DoubleByte fonts always involves sending the command FS & before sending any DoubleByte characters to the controller and always ends with calling the command FS . - this must also be done when switching from one DoubleByte character set to another. You can not send e.g. Japanese character and mix them with e.g. Korean. You would first need to end the Japanese DoubleByte “session” with FS . and then start a new Korean “session” with FS &
Parameters	Use of the typical DoubleByte hex codes for the used font
Defaults	
Notes	**The controller supports over 170 languages**. If the needed character set is not in this list please contact us. Please see Appendix 3.1 for language fonts listings.

Example	printStr("[DoubleByte - Chinese]");
	printLF();
	setDbMode();
	unsigned char
	buf5[]="\xB7\xC9\xC1\xF7\xD6\xB1\xCF\xC2\xC8\xFD\xC7\xA7\xB3\xDF\xA3\
	xAC\xD2\xC9\xCA\xC7\xD2\xF8\xBA\xD3\xC2\xE4\xBE\xC5\xCC\xEC";
	printStr((const char*)buf5);
	printLF();
	printLF();
	cancelDbMode();

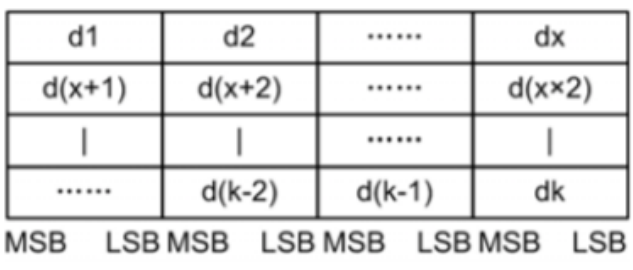
2.2.17 ESC *

Command	Select bit-image mode
Codes	ASCII: ESC * m xL xH d[0]…d[k] DEC: 27 42 m xL xH d[0]…d[k] HEX: 1B 2A m xL xH d[0]…d[k]

Description	Bit-image mode using m for the number of dots specified by xL and xH: m mode horizontal scale vertical scale 33 24 dots double density x1 x1 xL, xH represent a bit image in the horizontal direction as (xL+xHx256) dots d[k] represents the bit image data in column format This function requires a lot of byte padding and alignment from the host before sending the bitmap image to the controller. We recommend using GS v 0 for image printing.
Parameters	58mm 1≤ xL + xH ×256 ≤384 0≤ d[k] ≤0xff m = 33 k = (xL+xHx256)x3 in mode m 33 80mm 1≤ xL + xH ×256 ≤576 0≤ d[k] ≤0xff m = 33 k = (xL+xHx256)x3 in mode m 33
Note	 Dot image data (bit image)
Notes	data [d]k specifies a bit printed to 1 and not printed to 0. If the bit image exceeds one line of print area, the excess part will be ignored. The bit image is only stored in the print buffer and is not printed. When the print command is received, the printing starts. The printer buffer will be cleared when the printing is complete. If the image to be printed is too high, please split it into several images that the height is 24 dots (m = 33) and print them respectively. After filling up the image data, additional information can also be filled in the print buffer to print with the image. A print command must be executed e.g. ESC J in order to start printing and empty the print buffer.

Example	No C example available. We recommend GS v 0

2.2.18 GS v 0

Command	Print raster image
Codes	ASCII: GS v 0 m xL xH yL yH d[0] ... d[k] DEC: 29 118 48 m xL xH yL yH d[0] ... d[k] HEX: 1D 76 30 m xL xH yL yH d[0] ... d[k]
Description	This functions prints a vertical raster bit image m mode horizontal scale vertical scale 0,48,0x30 normal x1 x1 xL, xH specifies (xL + xH × 256) bytes in horizontal direction yL, yH specifies (yL + yH × 256) dots in vertical direction [d]k specifies the bit image data (raster format)
Parameters	58mm m = 0 or m=48 or m = 0x30 (ASCII,DEC,HEX) 1≤xL + xH × 256 ≤ 48 1≤yL ≤255,1≤yH ≤255 0≤d≤255 80mm m = 0 or m=48 or m = 0x30 (ASCII,DEC,HEX) 1≤xL + xH × 256 ≤ 72 1≤yL ≤255,1≤yH ≤255 0≤d≤255
Note	

Notes	Bitmaps must be byte aligned as to the width can be divided by 8. For example you have a bitmap 228x159, the width needs to be padded to 232 bits in order to get a full byte value 232/8 = 29 Bytes. You would then multiply 29x159 to get an array of 4611 Bytes. If that is not the case the bitmap must be padded. When this command is executed, data will be transmitted and printing started. No other print command is required. After the image is printed the command sets the print position to the beginning of the line and clears the print buffer.

Example	/**********
	** Function name: picturePrintH
	** Descriptions: This function is used to print raster bit image.
	** input parameters: ucType: raster bit image modes as follows:
	** PICTURE_TYPE_NORMAL normal 0
	** usHSize: horizontal byte number
	** usVSize: vertical dot number
	** pucDataBuf: starting address of the image buffer
	** Returned value: none
	**********/
	void picturePrintH (unsigned char ucType,
	unsigned short usHSize,
	unsigned short usVSize,
	unsigned char * pucDataBuf)
	{
	unsigned char buf[8]= {0x1D,0x76,0x30,0x00,0x00,0x00,0x00,0x00};//GS v 0
	buf[3] = ucType;
	buf[4] = usHSize & 0xFF; // xL
	buf[5] = (usHSize >> 8) & 0xFF; // xH
	buf[6] = usVSize & 0xFF; // yL
	buf[7] = (usVSize >> 8) & 0xFF; // yH
	Write(buf,8);
	unsigned long dSize = (unsigned long)usHSize;
	dSize *= ((unsigned long)usVSize);
	if (NULL != pucDataBuf)
	{
	Write(pucDataBuf,dSize);
	}
	else
	{
	unsigned long i;
	uint8_t empty = 0;
	for (i=0;i<dSize;i++)
	Write(&empty, 1);
	}
	}

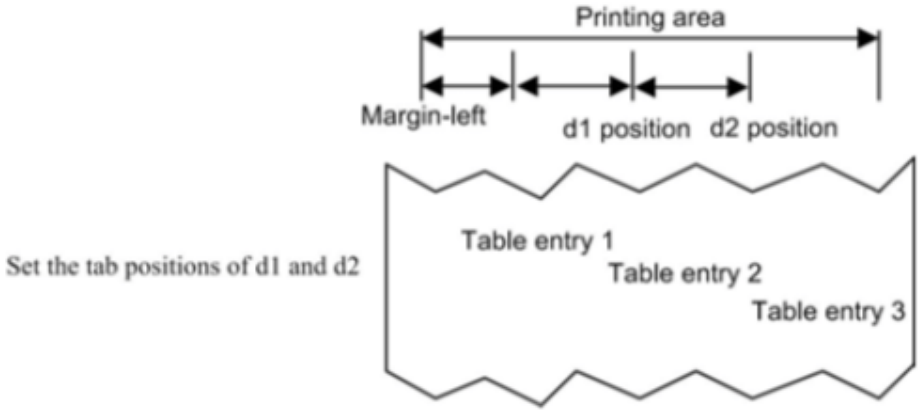
2.2.19 HT

Command	Move to next horizontal tab
Codes	ASCII: HT DEC: 9 HEX: 09
Description	Move the print position to the next defined tab position
Parameters	
Defaults	
Notes	Tab positions are set using ESC D. If no tab position is defined, the default, then this command will simply do a LF command.

Example	/**********
	** Function name: horizontalTab
	** Descriptions: This function is used to move the printing position to the next tab position in horizontal.
	** input parameters: none
	** Returned value: none
	**********/
	void horizontalTab (void)
	{
	unsigned char buf[1]= {0x09}; //HT
	Write(buf,1);
	}

2.2.20 ESC D

Command	Set the horizontal tab positions
Codes	ASCII: ESC D [d]k NULL DEC: 27 68 [d]k 0 HEX: 1B 44 [d]k 00

Description	d1 d2 d3…dk are horizontal tab position (Unit: 8 dots) NULL is the stop character/terminator 9 tab positions are supported When this command is used, any previous horizontal tab settings will be cancelled. k is not transmission data to the printer. Transmit [d]k in ascending order and place a NULL terminator at the end. When dk is less than or equal to dk-1, horizontal tab setting is finished, and the following data will be processed as normal data. The tab position can be adjusted using the HT command. When the left margin is changed, the tab position is also changed. Horizontal tab position settings are effective until the next ESC @ command is executed, printer is reset or printer is turned off.
Parameters	0≤k≤8 58mm 1≤d≤46 (d1<d2< ..... dk) 80mm 1≤d≤70 (d1<d2< ..... dk)
Defaults	[d]k= 0, no horizontal tab
Notes	

Example	/*****
	** Function name: horizontalTabSet
	
	** input parameters: pucNBuf: array of column position
	** ucDataNum: number of tabs
	** Returned value: none
	**
	
	*****/
	void horizontalTabSet (unsigned char * pucNBuf, unsigned char ucDataNum)
	{
	uint8_t buf[2] = {0x1B,0x44};//ESC D
	uint8_t term[1]= {0x00}; //null terminator signal for function
	Write(buf,2);
	Write(pucNBuf,ucDataNum);
	Write(term,1);
	}

2.2.21 GS h

Command	Set barcode height
Codes	ASCII: GS h n DEC: 29 104 n HEX: 1D 68 n
Description	Sets the height of a one-dimensional barcode in n dots

Parameters	12 ≤n≤ 128
Defaults	n = 48
Notes	The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void barcodeHeightSet(unsigned char ucHeight)
	{
	unsigned char hbuf[3] = {0x1D,0x68,0x00}; //GS h n
	hbuf[2] = ucHeight;
	Write(hbuf,3);
	}

2.2.22 GS w

Command	Set barcode width
Codes	ASCII: GS w n DEC: 29 119 n HEX: 1D 77 n
Description	Sets the width of a one-dimensional barcode in n dots

Parameters	1 ≤n≤ 4
Defaults	n = 2
Notes	The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void barcodeWidthSet(unsigned char ucWidth)
	{
	unsigned char wbuf[3] = {0x1D,0x77,0x00};//GS w n
	wbuf[2] = ucWidth;
	Write(wbuf,3);
	}

2.2.23 GS k

Command	Print barcode
Codes	ASCII: GS k m n d[0]…d[k] DEC: 29 107 m n d[0]…d[k] HEX: 1D 6B m n d[0]…d[k]
Description	This command prints one-dimensional bar codes of the types listed in the listing below. m is the encoding mode n is the encoding length such as: n = k +1, number of bytes of barcode data d specifies barcode data m type length d range in decimal 0x41 UPCA 11≤k≤12 48≤d≤57 0x42 UPC-E 11≤k≤12 48≤d≤57 0x43 EAN13 11≤k≤13 48≤d≤57 0x44 EAN8 1≤k≤7 48≤d≤57 0x45 CODE39 1≤k≤12 48≤d≤57,65≤d≤90,d = 32,36,37,43,45,46,47 0x46 ITF25 1≤k 12 48≤d≤57 0x47 CODABAR 1≤k≤12 48≤d≤57,65≤d≤90,d = 36,43,45,46,47,58 0x48 CODE93 1≤k≤12 48≤d≤57,65≤d≤90,d = 32,36,37,43,45,46,47 0x49 CODE128A 1≤k≤12 0≤d≤127

Parameters	
Defaults	
Notes	The CODE128 default selection is of type CODE128A. CODE128B and CODE128C are not supported CODE128A does not support the input of function characters FNC1 ~ FNC4. The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	/**********
	** Bar code system macro used in barcodePrint()
	**********/
	#define BARCODE_SYS_UPCA 0
	#define BARCODE_SYS_UPCE 1
	#define BARCODE_SYS_EAN13 2
	#define BARCODE_SYS_EAN8 3
	#define BARCODE_SYS_CODE39 4
	#define BARCODE_SYS_ITF25 5
	#define BARCODE_SYS_CODABAR 6
	#define BARCODE_SYS_CODE93 7
	#define BARCODE_SYS_CODE128 8
	
	void barcodePrint(unsigned char ucBarcodeSys,
	unsigned char *pucCodeBuf,
	unsigned char ucCodeLen)
	{
	unsigned char buf[4]= {0x1D,0x6B,0x00,0x00};//GS k m n d[k]
	buf[2] = (ucBarcodeSys+65); //ASCII A,B….
	buf[3] = ucCodeLen;
	Write(buf,3);
	Write(pucCodeBuf, ucCodeLen);
	}

2.2.24 ESC @

Command	Initialize printer

Codes	ASCII: ESC @ DEC: 27 64 HEX: 1B 40
Description	Initialize the printer Clears the print buffer Restores default values
Parameters	
Defaults	
Notes	

Example	void printerSoftInit(void)
	{
	uint8_t buf[2] = {0x1B,0x40}; //ESC @
	Write(buf,2);
	}

2.2.25 GS (

Command	Set serial communications parameters
Codes	ASCII: GS (n m DEC: 29 40 n m HEX: 1D 28 n m
Description	The default serial port settings are: Baud rate: 115200 Parity: None Flow Control: Software Flow Control XON/XOFF Data length: 8 bit

Parameters	n = 0 9600 bps n = 1 19200 bps n = 2 38400 bps n = 3 57600 bps n = 4 115200 bps n = 5 256000 bps *FW 1.22 or higher n = 6 375000 bps *FW 1.22 or higher m=0 No Flow Control m=1 Software Flow Control (xon/xoff) m=2 Hardware Flow Control: RTS/busy=1 Host Stop Send Data , RTS/busy=0 Host Can Send Data
Defaults	n = 4 115200bps m = 1 xon/xoff
Notes	The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void setComMode (unsigned char ucComSet, unsigned char ucFlowSet)
	{
	  uint8_t buf[4] = {0x1B,0x28,0x00,0x00};//GS (n m
	  buf[2] = ucComSet; // 0,1,2,3,4
	  buf[3] = ucFlowSet; //0,1,2
	  Write(buf,4);
	
	}

2.2.26 ESC M

Command	Set font size
Codes	ASCII: ESC M n DEC: 27 77 n HEX: 1B 4D n
Description	Allows to set font sizes depending of which mode the controller is in

Parameters	Double-byte mode (e.g. Japanese, ... Chinese) n=1 fontsize = 8x16 ASCII fontsize = 16x16 Chinese, Japanese, ... n=2 fontsize = 12x24 ASCII fontsize = 24x24 Chinese, Japanese, ... Latin byte mode (ASCII based languages) n=1, fontsize=8x16 n=2, fontsize= 12x24
Defaults	n = 2 Latin byte mode
Notes	The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void setFontMode (unsigned char ucFont)
	{
	uint8_t buf[3] = {0x1B,0x4D,0x00};//ESC M n
	buf[3] = ucFont // 0,1,2
	Write(buf,3);
	}

2.2.27 DLE DC4

Command	Clear print buffer
Codes	ASCII: DLE DC4 DEC: 16 28 HEX: 10 14
Description	Clears and resets the print buffer
Parameters	
Defaults	

Notes	

Example	void clearPrintBuffer (void)
	{
	uint8_t buf[2] = {0x10,0x14};//DLE DC4
	Write(buf,2);
	}

2.2.28 DLE EOT

Command	Query printer status
Codes	ASCII: DLE EOT DEC: 16 4 HEX: 10 04
Description	Returns the paper status and temperature of the printer in 2 lines 1st line: 0 = No Paper 1 = Paper Ok 2nd line: xxC = temperature in Celsius
Parameters	
Defaults	
Notes	

Example	void getPrintStatus (void)
	{
	uint8_t buf[2] = {0x10,0x04};//DLE EOT
	Write(buf,2);
	}

2.2.29 DLE ENQ

Command	Query firmware version
Codes	ASCII: DLE ENQ DEC: 16 5 HEX: 10 05
Description	Returns information about the firmware in 4 lines 1st line: controller type 2nd line: firmware version 3rd line: firmware date 4th line: print head family type index e.g. 0, 1, 2, 3
Parameters	
Defaults	Family index 0. n= 0 MM58_FTP628MCL101 58mm 6stb 7.2V designs n=1 MM80_FTP638MCL101 80mm 5stb 7.2V designs n=2 MM58_LTP02_245_13 58mm 1stb line 7.2V designs n=3 MM58_LTP02_245_C1 58mm 1stb line 4.0V designs n=4 MM58_CUTTER_384 58mm NL024B 7.2V/24V designs n=5 MM80_CUTTER_576 80mm NL024B - PT72EP n=6 MM58_M23_DH_H69 58mm 1stb line 7.2V design n=7 MM58_M23_SX 58mm 1stb line 4.0V design n=8 MM80_SEIKO_LTPV345 80mm print head n=9 MM58_PT486HLV 58mm 6stb line low power design n=10 MM58_JX2R39BQ 58mm 6stb 3.5V-9.5V label printer NL024B
Notes	Example of a return: 1st line: NL022 2nd line: V1.10 3rd line: 19-09-2019 4rd line: 3

Example	void getFirmwareInfo (void)
	{
	uint8_t buf[2] = {0x10,0x05};//DLE EOT
	Write(buf,2);
	}

2.2.30 ESC Z

Command	Set Print Head Type
Codes	ASCII: ESC Z n DEC: 27 90 n HEX: 1B 5A n
Description	This command configures the controller for the indicated print head type or compatible design. The selection is saved into flash and need not be done every time.
Parameters	n= 0 MM58_FTP628MCL101 58mm 6 stb 7.2V compatible designs n=1 MM80_FTP638MCL101 80mm 5stb 7.2V compatible designs n=2 MM58_LTP02_245_13 58mm 1stb line 7.2V designs n=3 MM58_LTP02_245_C1 58mm 1stb line 4.0V designs n=4 MM58_CUTTER_384 58mm NL024 chip 7.2V/24V designs n=5 MM80_CUTTER_576 80mm NL024B - PT72EP n=6 MM58_M23_DH_H69 58mm 1stb line 7.2V design n=7 MM58_M23_SX 58mm 1stb line 4.0V design n=8 MM80_SEIKO_LTPV345 80mm print head n=9 MM58_PT486HLV 58mm 6stb line low power design n=10 MM58_JX2R39BQ 58mm 6stb 3.5V-9.5V label printer NL024B
Defaults	Family index 0 n= 0 MM58_FTP628MCL101 58mm 6 stb 7.2V compatible designs
Notes	

Example	void setPrinterHead(unsigned char ucHeadIndex)
	{
	unsigned char buf[3] = {0x1B,0x5A,0x00};//ESC Z n
	buf[2] = ucHeadIndex;
	Write(buf,3);
	}

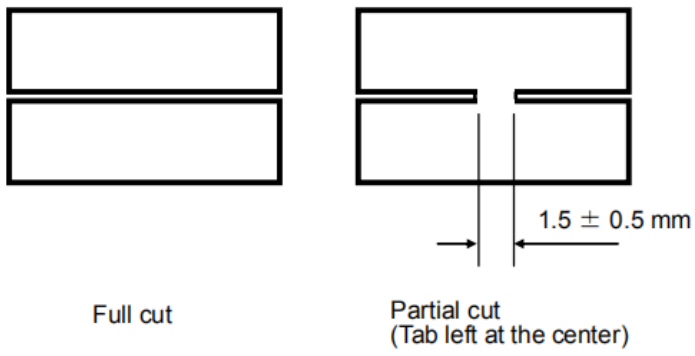
2.2.31 GS H

Command	Show or Hide Bar Code Text

Codes	ASCII: GS H n DEC: 29 72 n HEX: 1D 48 n
Description	This command allows you to Hide or Show bar code text. The default is show bar code text
Parameters	n = 0 Hide Bar Code Text n = 1 Show Bar Code Text *Default
Defaults	Show Text n = 1
Notes	

Example	void barcodeTextYesNo(unsigned char ucYesNo)
	{
	  unsigned char hbuf[3] = {0x1D,0x48,0x00};//GS H n
	  hbuf[2] = ucYesNo; //0,1
	  Write(hbuf,3);
	}

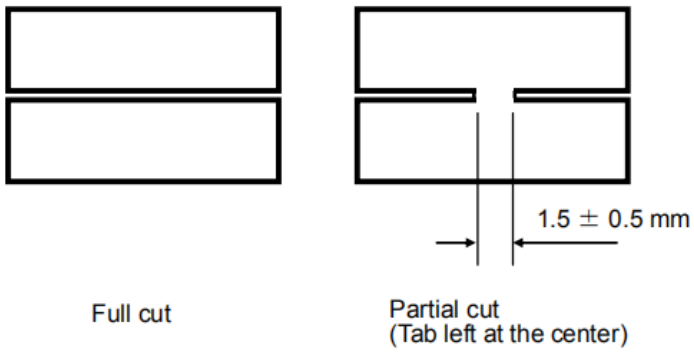
2.2.32 ESC i

Command	Full Cut
Codes	ASCII: ESC i DEC: 27 105 HEX: 1B 69
Description	

Parameters	
Defaults	
Notes	This command requires the NL024 controller and thermal printer hardware with cutter

Example	void fullCut(void)
	{
	uint8_t buf[2] = {0x1B,0x69};//ESC i
	Write(buf,2);
	}

2.2.33 ESC n

Command	Partial Cut
Codes	ASCII: ESC n DEC: 27 110 HEX: 1B 6E
Description	
Parameters	
Defaults	
Notes	This command requires the NL024 controller and thermal printer hardware with cutter

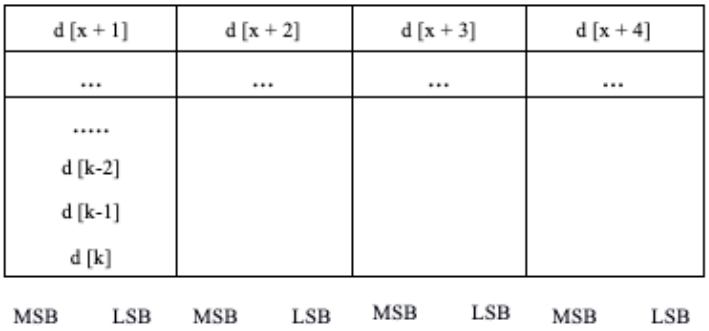
Example	void partialCut(void)
	{
	
	uint8_t buf[2] = {0x1B,0x6E};//ESC n
	Write(buf,2);
	
	}

2.2.34 GS a

Command	Set/Cancel status callback
Codes	ASCII: GS a n DEC: 29 97 n HEX: 1D 61 n
Description	This command allows to configure automatic callback from the printer controller to the host via the communication link for requested status information in real-time.
Parameters	set /cancel printer status callback. Bit description. Bits can be combined using OR bit operator n = xxxx xxxx -> total 8 bits Bit Description 0 reserved 1 reserved 2 paper out 3 overheat 4 reserved 5 reserved 6 reserved 7 reserved
Defaults	n=0 Default is no callback
Notes	If either bit 2, 3 or both are set then automatic callback is enabled

Example	void setPrinterCallBack(unsigned char ucCallBack)
	{
	unsigned char buf[3] = {0x1D,0x61,0x00};//GS a n
	buf[2] = ucCallBack; //0x04 enable paper out callback
	Write(buf,3);
	}

2.2.35 FS q

Command	Save NVbitmap to flash
Codes	ASCII: FS q n m xL xH yL yH d [0] ... d [k] DEC: 28 113 n m xL xH yL yH d [0] ... d [k] HEX: 1C 71 n m xL xH yL yH d [0] ... d [k]
Description	The printer controller can store up to 4 bitmaps of max size 256x256 in internal flash memory. When you store a new bitmap at index location 1…4 any previous bitmap in flash at that location will be deleted and replaced by the new bitmap data at that index location. The number of bytes of type xL, xH in horizontal direction is obtained as xL + xH × 256. One byte can store eight pixels. The number of dots yL, yH in vertical direction is obtained as = yL + yH × 256 The data stored in the flash should be arranged in transverse mode. Suppose xL = 4, xH = 0, yL = 4, yH = 0
	
Parameters	1≤ n ≤4 m = 0
Defaults	

Notes	A stable power supply must be guaranteed during bitmap transfer otherwise it may result in flash corruption! Bitmaps must be byte aligned as to the width can be divided by 8. For example you have a bitmap 228x159, the width needs to be padded to 232 bits in order to get a full byte value 232/8 = 29 Bytes. You would then multiply 29x159 to get an array of 4611 Bytes. If that is not the case the bitmap must be padded. Suppose a bitmap of 16x16 dot matrix, will have a width of 2 bytes and 16 dots high and the total effective length of 2x16 = 32 data bytes. If we want to store it as the third (3) index flash download the hex command would look something like this: 1C 71 03 00 02 00 10 00 02 02 02 02 04 04 04 04 08 08 08 08 10 10 10 10 02 02 02 02 04 04 04 04 08 08 08 08 10 10 10 10 02

Example

```
/*******************************************************************************************
** Function name:          downloadNVBitmap
** Descriptions:           This function is used to load bitmaps into flash memory
** input parameters:       ucLocation: bit image location in flash:
**                         n: 1, 2, 3, 4
**                         ucAlign: m in general use 0
**                         usHSize: horizontal byte number
**                         usVSize: vertical dot number
**                         pucDataBuf: starting address of the image buffer
** Returned value:         none
*******************************************************************************************/
void downloadNVBitmap (unsigned char ucLocation,
                       unsigned char ucAlign,
                       unsigned short usHSize,
                       unsigned short usVSize,
                            unsigned char * pucDataBuf)
{
unsigned char buf[8]= {0x1D,0x71,0x00,0x00,0x00,0x00,0x00,0x00};//FS q
     buf[2] = ucLocation
     buf[3] = ucAlign;
      buf[4] = usHSize & 0xFF;           // xL
      buf[5] = (usHSize >> 8) & 0xFF;  // xH
      buf[6] = usVSize & 0xFF;           // yL
      buf[7] = (usVSize >> 8) & 0xFF;  // yH
      Write(buf,8);
      unsigned long dSize = (unsigned long)usHSize;
      dSize *= ((unsigned long)usVSize);
      if (NULL != pucDataBuf)
      {
            Write(pucDataBuf,dSize);
      }
      else
      {
            unsigned long i;
            uint8_t empty = 0;
            for (i=0;i<dSize;i++)
                  Write(&empty, 1);
      }
}
```

2.2.36 FS p

Command	Print NVbitmap from flash

Codes	ASCII: FS p n m DEC: 28 112 n m HEX: 1C 70 n m
Description	Print available bitmaps directly from flash memory. Only the index of the bitmap in the flash needs to be passed and a valid bitmap needs to be at that location. Valid index values are n: 1…4 m = 0 needs to be passed, meaning printing in normal mode (length:width = 1:1)
Parameters	1≤ n ≤4 m = 0
Defaults	
Notes	As indicated in the Download NVbitmap function, the max size of a bitmap can not exceed 256x256.

Example	void printNVBitmap(unsigned char ucBitmapIndex, unsigned char ucMode)
	{
	unsigned char buf[4] = {0x1C,0x70,0x00,0x00};//FS p n m
	buf[2] = ucBitmapIndex; //index range 1,2,3,4
	buf[3] = ucMode; //mode value is 0 for 1:1 normal mode bitmap
	Write(buf,4);
	}

2.2.37 GS (k Size/Version

Command	Set QR code size/version
Codes	ASCII: GS (k 03 00 cn fn n DEC: 29 40 107 03 00 49 67 n Hex: 1D 28 6B 03 00 31 43 n
Description	Sets the QRcode version(size): cn=0x31(fixed value) fn=0x43(fixed value) n: specifies the QR module size range

Parameters	n=0 //module size 21->53
	n=1 //module size 57->121
	n=2 //module size 125->177

Defaults	n=0 cn=0x31 fn=0x43
Notes	The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void setQRSizeVersion(unsigned char ucModuleSizeRange)
	{
	unsigned char buf[8] = {0x1D,0x28,0x6B,0x03,0x00,0x31,0x43,0x00};
	buf[7] = ucModuleSizeRange; //0x00,0x01,0x02
	Write(buf,8);
	}

2.2.38 GS (k Error Level

Command	Set QR code error correct level
Codes	ASCII: GS (k 03 00 cn fn n DEC: 29 40 107 03 00 49 69 n Hex: 1D 28 6B 03 00 31 45 n
Description	Set the error correction level for QR Code: cn=0x31(fixed value) fn=0x45(fixed value) n: specifies the QR code correction level
Parameters	n=0 // 7% of codewords can be restored n=1 // 15% of codewords can be restored n=2 // 25% of codewords can be restored n=3 // 30% of codewords can be restored
Defaults	n=3 cn=0x31 fn=0x45
Notes	QR Code employs Reed-Solomon error correction to generate a series of error correction codewords. The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	void setQRcodeErrorCorrection(unsigned char ucErrorCorrectionLevel)
	{
	unsigned char buf[8] = {0x1D,0x28,0x6B,0x03,0x00,0x31,0x45,0x00};
	buf[7] = ucErrorCorrectionLevel; //0x00,0x01,0x02,0x03
	Write(buf,8);
	}

2.2.39 GS (k Encode Buffer

Command	QR code transfer data to encode buffer
Codes	ASCII: GS (k pL pH cn fn data1—> datak DEC: 29 40 107 pL pH 49 80 data1—> datak Hex: 1D 28 6B pL pH 31 50 data1—> datak
Description	Transfers the data in data1--->datak to the encode buffer and encodes it as a QR code. [pL,pH] is the data length, it is a 2 byte (WORD) value The high byte is represented by pH The low byte is represented by pL If we have: [pL , pH] = [0x03, 0x01], then the data length would give the value = 0x0103 (Hex) = 259(DEC) cn=0x31(fixed value) fn=0x50(fixed value)
Parameters	Data length restriction: NL022: 4≤(pH×256 + pL)≤110 NL025, NL024B: 4≤(pH×256 + pL)≤384
Defaults	cn=0x31 fn=0x50

Notes	The data in data1 —> datak can have only the following characters Numerical: “0”~ “9” Alphanumeric: “0” ~ “9“, “A” ~ “Z“, SP, $, %, *, +,–, . , /, : 8bit data: 00H~ FFH Example 1: 1D 28 6B 05 00 31 50 31 32 33 34 35 1D 28 6B 05 00 (frame header) 31 50 (cn, fn) 31 32 33 34 35 (data: data1-->data5) Example 2: 1D 28 6B 06 00 31 50 31 32 33 34 35 36 1D 28 6B 06 00 (frame header) 31 50 (cn, fn) 31 32 33 34 35 36 (data: data1-->data6) The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example

```
/*******************************************************************************************
** Function name:        transferQRcodeData
** Descriptions:         This function is used to transfer data and QR encode
** input parameters:
**                       usBufferSize: size of buffer
**                       pucDataBuf: starting address of the data buffer
** Returned value:       none
*******************************************************************************************/
void transferQRcodeData ( unsigned short usbufferSize,
                                                    unsigned char * pucDataBuf)
{
unsigned char buf[7]= {0x1D,0x28,0x6B,0x00,0x00,0x31,0x50};

    unsigned short usCompleteBufferwithFunc;

    usCompleteBufferwithFunc =  usBufferSize+2;

    //we need to add 2 bytes for the control function parameters for pL and pH
    //to the passed usBufferSize
    // they represent: cn=0x31, fn=0x50

     buf[3] = usCompleteBufferwithFunc & 0xFF; // pL
     buf[4] = (usCompleteBufferwithFunc >> 8) & 0xFF;// pH
     Write(buf,7);

     if (NULL != pucDataBuf)
     {
          Write(pucDataBuf,usbufferSize);
     }
     else
     {
          unsigned long i;
          uint8_t empty = 0;
          for (i=0;i<usbufferSize;i++)
               Write(&empty, 1);
     }
}
```

2.2.40 GS (k Print QR

Command	Print QR code from encode buffer
Codes	ASCII: GS (k 03 00 cn fn m DEC: 29 40 107 03 00 49 80 48 Hex: 1D 28 6B 03 00 31 51 30

Description	Print the encoded QR code data from the encode buffer cn=0x31(fixed value) fn=0x51(fixed value) m=0x30(fixed value)
Parameters	
Defaults	cn=0x31 fn=0x51 m=0x30
Notes	Before you can run this command you must set up the QR code with the other QR code commands in particular the transfer QR code data to the encode buffer command: GS (k pL pH cn fn m data1--->datak The settings of this command are effective until the next ESC @ command is executed, printer is reset or printer is turned off.

Example	/**********
	** Function name: printQRcodeData
	** Descriptions: This function is used to print the QR code buffer
	** input parameters: none
	** Returned value: none
	**********/
	void printQRcodeData (void)
	{
	unsigned char buf[8] = {0x1D,0x28,0x6B,0x03,0x00,0x31,0x51,0x30};
	Write(buf,8);
	}

2.2.41 DLE SYN

Command	Put controller into sleep mode
Codes	ASCII: DLE SYN DEC: 16 22 HEX: 10 16
Description	Puts the controller in low power sleep mode
Parameters	

Defaults	
Notes	Send any command to put controller back into active mode. Then send print command and data.

Example	void enterSleepMode(void)
	{
	unsigned char buf[2] = {0x10,0x16};//DLE SYN
	Write(buf,2);
	}

2.2.42 FS u Font Bin File

Command	Upload font bin file - up to 8 font libs
Codes	ASCII: FS u n 0 lenH lenL d0..dk DEC: 34 165 n 0 lenH lenL d0..dk HEX: 1C 75 n 0 lenH lenL d0..dk n=ID for font file (0x11..0x18) lenH+lenL = size of bin file d0..dk = binary font data
Description	Allows uploading of own font file for use in printing. Bitmap glyphs should be 12x24 and character mapping starting at 0x20.
Parameters	n=ID for font file (0x11..0x18) lenH+lenL = size of bin file d0..dk = binary font data
Defaults	
Notes	You can find our font collection at printercontroller.com

Example	void uploadFont (unsigned char fontID,
	unsigned char lenH,
	unsigned char lenL,
	unsigned int fontSize,
	unsigned char * pucDataBuf)
	{
	unsigned char buf[6]= {0x1C,0x75,0x00,0x00,0x00,0x00};//FS u
	buf[2] = fontID; //0x13
	buf[4] = lenH;
	buf[5] = lenL;
	Write(buf,6);
	
	if (NULL != pucDataBuf)
	{
	Write(pucDataBuf,fontSize);
	}
	else
	{
	unsigned long i;
	uint8_t empty = 0;
	for (i=0;i<fontSize;i++)
	Write(&empty, 1);
	}
	}

2.2.43 ESC t

Command	Set active uploaded font with ID
Codes	ASCII: ESC t n DEC: 33 164 n HEX: 1B 74 n n = ID for font file (0x11..0x18) or 0 = default (CP437) 1 = CP1250
Description	Activates uploaded own font file or sets default internal fonts. Bitmap glyphs should be 12x24 and character mapping starting at 0x20.

Parameters	n=ID for font file (0x11..0x18)
Defaults	
Notes	You can find our font collection at printercontroller.com

Example	void setActiveFont(unsigned char ucID)
	{
	unsigned char buf[3] = {0x1B,0x74,0x00};//ESC t n
	buf[2] = ucID;
	Write(buf,3);
	}

2.2.44 GS ENQ SOH 0 Buffer

Command	Save label byte data to chip 2k buffer
Codes	ASCII: GS ENQ SOH 0 lenL lenH d0…dn DEC: 29 05 01 48 lenL lenH d0…dn HEX: 1D 05 01 30 lenL lenH d0…dn lenL+lenH = size of byte data (max label buffer is 2048 bytes) d0..dk = binary label data
Description	Create the label data as a binary array of bytes and send it to the NL02x
Parameters	lenL+lenH = size of byte data (max label buffer is 2048 bytes) d0..dk = binary label data
Defaults	

Notes	

Example	void saveLabelData (unsigned char lenL,
	unsigned char lenH,
	unsigned int LabelDataSize,
	unsigned char * pucDataBuf)
	{
	unsigned char buf[6]= {0x1D,0x05,0x01,0x30,0x00,0x00};//GS ENQ SOH 0
	buf[4] = lenL;
	buf[5] = lenH;
	Write(buf,6);
	
	if (NULL != pucDataBuf)
	{
	Write(pucDataBuf,fontSize);
	}
	else
	{
	unsigned long i;
	uint8_t empty = 0;
	for (i=0;i<fontSize;i++)
	Write(&empty, 1);
	}
	}

2.2.45 GS ACK

Command	Go to next label home position
Codes	ASCII: GS ACK DEC: 29 06 HEX: 1D 06
Description	Go to home position of the next label

Example	void setLabelHome(void)
	{
	unsigned char buf[2] = {0x1D,0x06};//GS ACK
	Write(buf,2);
	}

2.2.46 GS ENQ EOT n

Command	Print n amounts of labels from label buffer
Codes	ASCII: GS ENQ EOT n DEC: 29 05 04 n HEX: 1D 05 04 n
Description	Print n amount of labels from the label buffer
Parameters	n=amount of labels to print from label buffer data

Example	void setActiveFont(unsigned char ucNumLabels)
	{
	unsigned char buf[4] = {0x1D,0x05,0x04,0x00};//GS ENQ EOT n
	buf[3] = ucNumLabels;
	Write(buf,4);
	}

3. Appendix

3.1 Standard Language Fonts

Please see the tables on the following pages.
Country code Indexes marked RED indicates not yet activated.

Family	Area	No.	country	Language	ISO-8859
	Europe	1	Britain(United Kingdom)	English	
		2	lreland		
	North America	3	English	English	
		4	Canada	English,French	
		5	Belize		
		6	Jamaica		

		7	Trinidad and Tobago	English	
		8	Bahamas		
		9	Antigua and Barbuda		
		10	Dominica		
		11	St.Vincent		
		12	St.Lucia		
		13	Grenada		
		14	St.Kitts-Nevis		
	South America	15	Guyana	English	
	Oceania	16	Australia	English	
		17	New Zealand		
		18	Tonga		
		19	Fiji		
		20	Palau		
		21	Solomon		
		22	Vanuatu		
		23	Kiribati		
		24	Nauru		
		25	Marshall Islands		
Latin_English	Africa	26	South Africa	English、Afrikaans	ISO-8859-1
		27	Zimbabwe	English	
		28	Gambia		
		29	Sierra Leone		
		30	Liberia		
		31	Ghana		
		32	Nigeria		
		33	Uganda		
		34	Zambia		
		35	Malawi		
		36	Seychelles		
		37	Mauritius		
		38	Botswana		
		39	Namibia		
		40	Lesotho		
Latin_French	Europe	41	France	French	ISO8859-15

		42	Belgium	French、Dutch	
		43	Monaco	French、Italian	
	North America	44	Haiti	French	
	Africa	45	Senegal	French	
		46	Mali		
		47	Burkina Faso		
		48	Guinea		
		49	cote dlvoire		
		50	Togo		
		51	Benin		
		52	Niger		
		53	Cameroon		
		54	Chad		
		55	Central African Republic		
Latin_French	Africa	56	Djibouti	French	
		57	Burundi		
		58	Republic of Democratic Congo		
		59	Congo		
		60	Gabon		
		61	Comoros		
		62	Madagascar		
Latin_Spanish	Europe	63	Spain	Spanish、Catalan	ISO8859-1 ISO8859-15
		64	Andorra	Spanish	
	North America	65	Mexico	Spanish	
		66	Guatemala		
		67	Costa Rica		
		68	Panama		
		69	Dominican Republic		
		70	El Salvador		
		71	Honduras		
		72	Nicaragua		
		73	Puerto Rico		
		74	Cuba		
	South America	75	Venezuela	Spanish	

	America	76	Colombia		
		77	Peru		
		78	Argentina		
		79	Ecuador		
		80	Chile		
		81	Uruguay		
		82	Paraguay		
		83	Bolivia		
	Africa	84	Equatorial New Guinea		
		85	Ceuta and Melilla		
Latin_ Portuguese	Europe	86	Portugal	Portuguese	ISO8859-1 ISO8859-15
	North America	87	Brazil		
	Africa	88	Cape Verde		
		89	Guinea-Bissau		
		90	Sao Tome and Principe		
		91	Angola		
		92	Mozambique		
Europe	Europe	93	Germany	German	ISO8859-1 ISO8859-15
		94	Switzerland	German、French	
		95	Austria	German	
		96	Luxembourg	German、French	
		97	Liechtenstein	German	
Latin_ Dutch	Europe	98	Holland	Dutch	ISO8859-1 ISO8859-15
	North America	99	Surinam		
Latin_ Northern Europe	Europe	100	Denmark	Danish	ISO8859-1-10
		101	Norway	Norwegian language	
		102	Sweden	Swedish	
		103	Faroes, The	Faroese	
		104	Greenland	Greenlandic	
		105	Iceland	Icelandic	
		106	Finland	Finnis、Swedish	ISO8859-13、-15
		107	Estonia	Estonian	ISO8859-4、

		108	Latvia	Latvian	-13 ISO8859-4、-13
		109	Lithuania	Lithuanian	ISO8859-4、-13
Latin_ Central Europe	Europe	110	Czech	Czech	ISO8859-2
		111	Slovakia	Slovak language	ISO8859-2
		112	Poland	Polish	ISO8859-2、-16
		113	Hungary	Hungarian	ISO8859-2、-16
		114	Romania	Romanian	ISO8859-16
		115	Slovenia	Slovenian	ISO8859-2、-16
		116	Croatia	Croatian	ISO8859-2、-16
Latin_ Southern Europe	Europe	117	Italy	Italian language	ISO8859-1 ISO8859-16
		118	San Marino		
		119	Vatican		
		120	Turkey	Turkish	ISO8859-9
		121	Malta	Maltese	ISO8859-3、-9
		122	Albania	Albanian	ISO8859-1、-16
Latin_ Southeast Asia	Asia	123	Vietnam	Vietnamese	ISO8859-1
		124	Malaysia	Malay	
		125	Brunei		
		126	Indonesia	Indonesian	
		127	East Timor		
		128	Philippines，The	EnglisTagalog	
Latin Africa	Africa	129	Kenya	Swahili	ISO8859-1
		130	Tanzania		
Cyrillic_ Eastern Europe	Europe	131	Russia	Russian	ISO8859-5
		132	Byelorussia 或 Belarus		
		133	Ukraine	Russian、Ukrainian	
		134	Bulgaria	Bulgarian	
		135	Moldova	Russian	
		136	F.R.Yugoslavia	Serbian language	

		137	Barbados	Serbian language	
		138	Macedonia	Macedonian	
Cyrillic _Asia	Asia	139	Azerbaijan	Azerbaijani	ISO8859-5
		140	Kirghizstan	Kyrgyz	
		141	Tajikistan	Tajikistan Language	
		142	Turkmenistan	Turkmenistan Language	
		143	Uzbekistan	Uzbek	
		144	Kazakhstan	Kazakh	
		145	Mongolia	Mongolian	
Greek	Europe	146	Greece	Greek	ISO8859-7
		147	Cyprus		
Arabic	Africa	148	Egypt	Arabic	ISO8859-6
		149	Tunisia		
		150	Libya		
		151	Morocco		
		152	Algeria		
		153	Sudan, The		
		154	Somalia		
		155	Djibouti		
		156	Mauritania		
Arabic	Asia	157	Syria	Arabic)	ISO8859-6
		158	United Arab Emirates, The		
		159	Lebanon		
		160	Yemen		
		161	Kuwait		
		162	Qatar		
		163	Bahrain		
		164	Oman		
		165	Jordan		
		166	Iraq		
		167	Saudi Arabia		
		168	Palestine		
		169	Iran	Persian	
		170	Pakistan	Urdu language 、 Arabic	

		171	Afghanistan	Pashto	
Hebrew	Asia	172	Israel	Hebrew	ISO8859-8
Thai	Asia	173	Thailand	Thai	ISO8859-11
Chinese	Asia	174	China	Chinese	
Japanese	Asia	175	Japan	Japanese	
Korean	Asia	176	Korea	Korean	