Software Developer's Manual

FBPL Command Reference

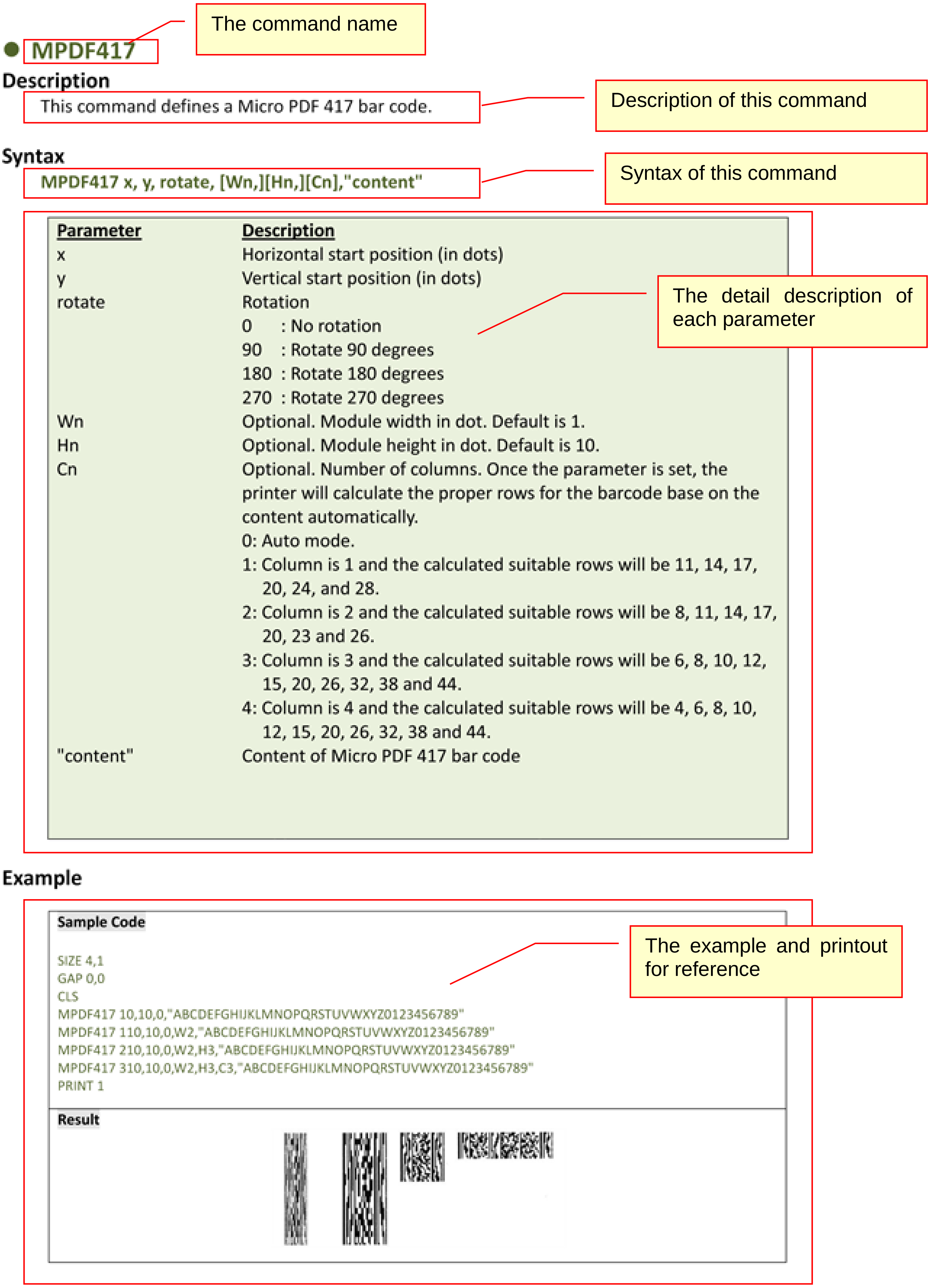
TD-4520TN/4420TN

TD-4650TNWB/4750TNWB/4650TNWBR/ 4750TNWBR

Version 3.00

TABLE OF CONTENTS

1 How to Read



The command name

● MPDF417

Description

This command defines a Micro PDF 417 bar code.

Description of this command

Syntax

MPDF417 x, y, rotate, [Wn,][Hn,][Cn],"content"

Syntax of this command

Parameter	Description
x	Horizontal start position (in dots)
y	Vertical start position (in dots)
rotate	Rotation 0 : No rotation 90 : Rotate 90 degrees 180 : Rotate 180 degrees 270 : Rotate 270 degrees
Wn	Optional. Module width in dot. Default is 1.
Hn	Optional. Module height in dot. Default is 10.
Cn	Optional. Number of columns. Once the parameter is set, the printer will calculate the proper rows for the barcode base on the content automatically. 0: Auto mode. 1: Column is 1 and the calculated suitable rows will be 11, 14, 17, 20, 24, and 28. 2: Column is 2 and the calculated suitable rows will be 8, 11, 14, 17, 20, 23 and 26. 3: Column is 3 and the calculated suitable rows will be 6, 8, 10, 12, 15, 20, 26, 32, 38 and 44. 4: Column is 4 and the calculated suitable rows will be 4, 6, 8, 10, 12, 15, 20, 26, 32, 38 and 44.
"content"	Content of Micro PDF 417 bar code

The detail description of each parameter

Example

Sample Code

```
SIZE 4,1
GAP 0,0
CLS
MPDF417 10,10,0,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
MPDF417 110,10,0,W2,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
MPDF417 210,10,0,W2,H3,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
MPDF417 310,10,0,W2,H3,C3,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
PRINT 1
```

Result

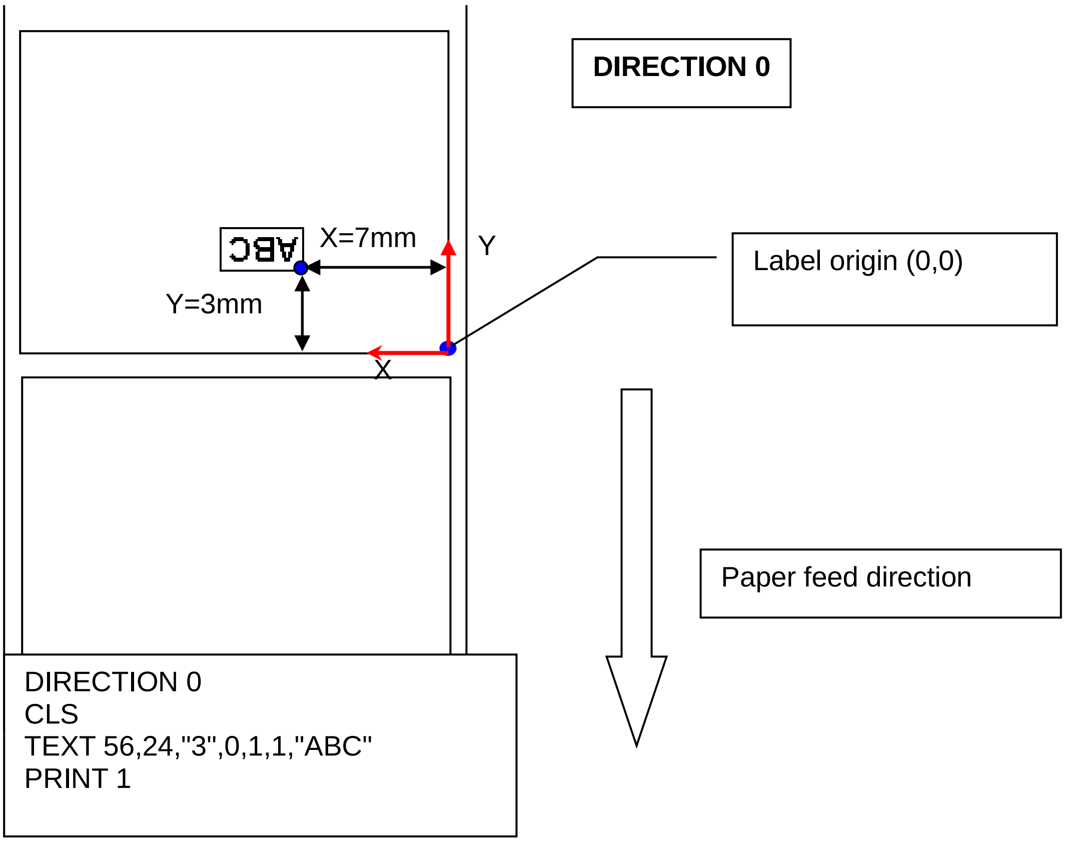
The example and printout for reference

2 Document Conventions

This manual uses the following typographic conventions.

Convention	Description
[expression list]	Items inside square brackets are optional, expression maximum length 2*1024 bytes.
<ESC>	ASCII 27, control code of status polling command returns/runs the printer status immediately.
~	ASCII 126, control code of status polling command returns the printer status only when the printer is ready.
Space	ASCII 32, characters will be ignored in the command line.
"	ASCII 34, beginning and ending of expression.
CR, LF	ASCII 13, ASCII 10, denotes end of command line.
NULL	ASCII 0, supported in the expression.
Note: ***203 DPI: 1 mm = 8 dots***	The font in bold and italic type is used for note.

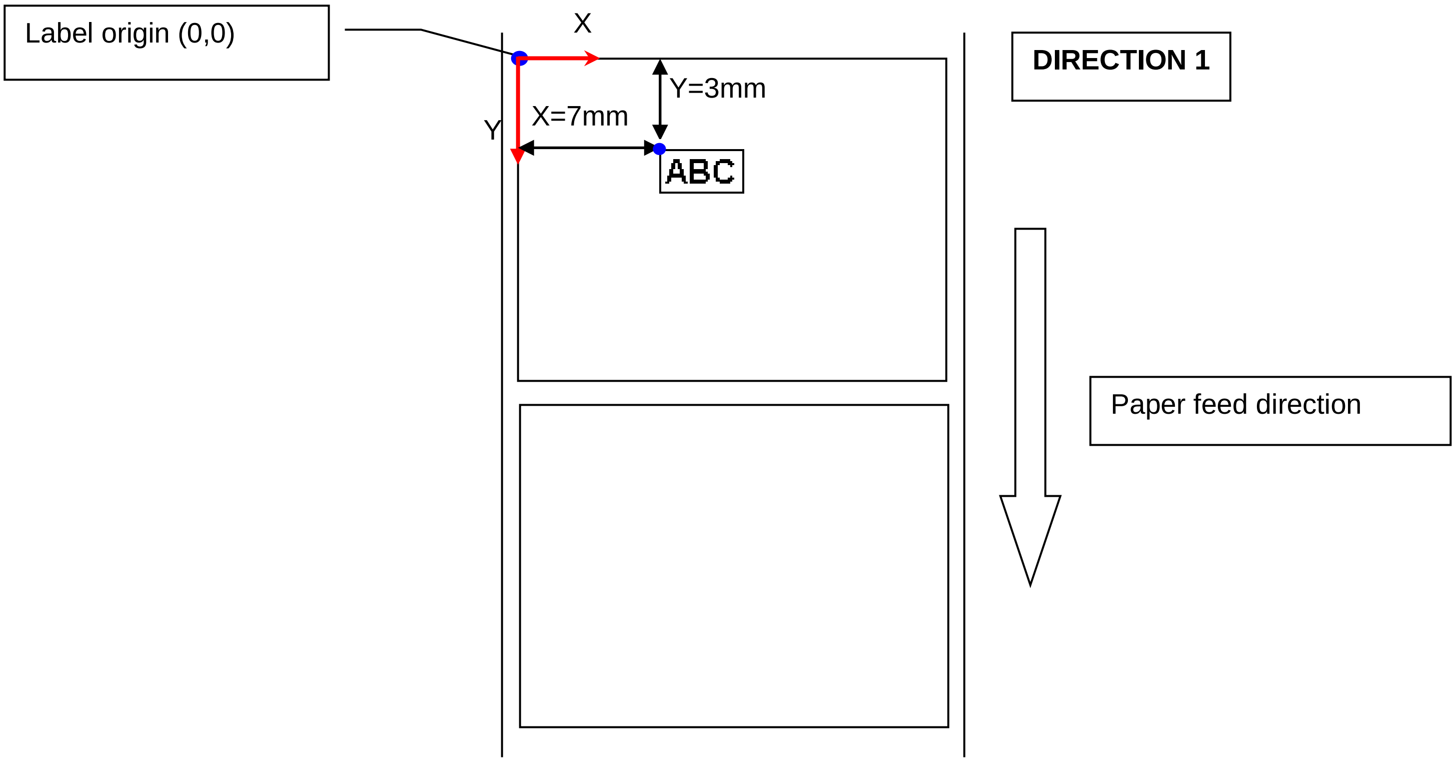
3 Object Position Calculation



```
DIRECTION 0
CLS
TEXT 56,24,"3",0,1,1,"ABC"
PRINT 1
```

Note :

1. ***203 DPI, 1mm=8 dots ; 300 DPI, 1mm=11.8 dots***
2. ***Only integer portion will be used. Ex. 2 mm = 23.6 dots then 23 dots will be used.***



```
DIRECTION 1
CLS
TEXT 56,24,"3",0,1,1,"ABC"
PRINT 1
```

4 Printer Model List

Series	Models
TD-4T series	TD-4420TN/ 4520TN
	TD-4650TNWB/ 4750TNWB/ 4650TNWBR/ 4750TNWBR

5 Setup and System Commands

5.1 SIZE

Description

This command defines the label width and length.

Syntax

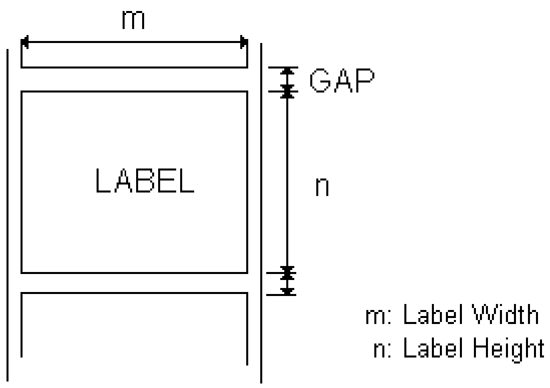
SIZE m[,n]	English system (inch)
SIZE m mm[,n mm]	Metric system (mm)
SIZE m dot[,n dot]	Dot measurement

Parameter	**Description**
M	Label width (inch/ mm/ dot)
[N]	Label length (inch/ mm/ dot)

Note :

- ***200 DPI : 1 mm = 8 dots***
  300 DPI : 1mm = 12 dots
- ***For metric and dot systems, there must be a space between parameter and "mm" or "dot".***

Example

Sample Code	**Result**
- English system (inch): **SIZE 3.5,3.00** - Metric system (mm): **SIZE 100 mm,100 mm**	

See Also

GAP, BLINE

5.2 GAP

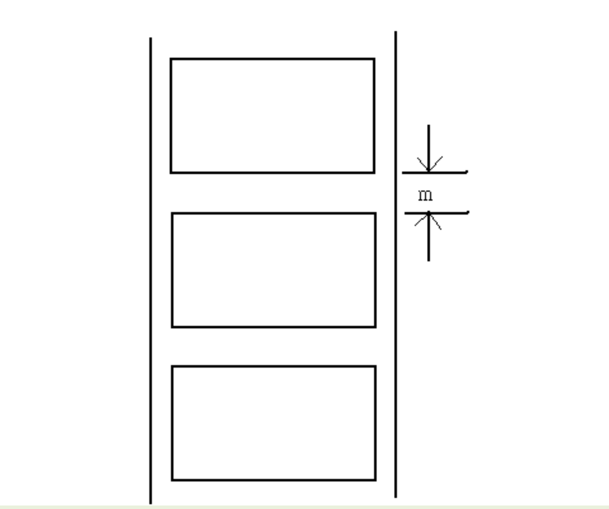
Description

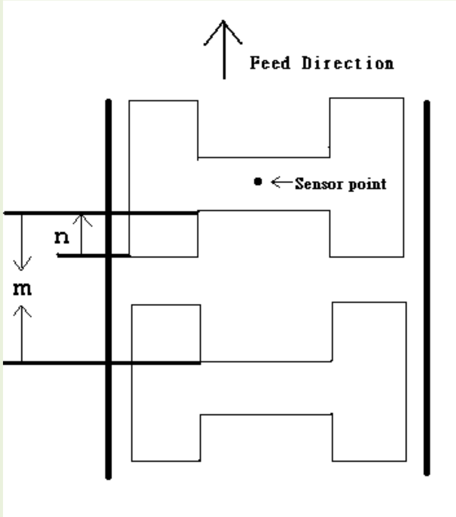
Defines the gap distance between two labels.

Syntax

GAP m,n	English system (inch)
GAP m mm,n mm	Metric system (mm)
GAP m dot,n dot	Dot measurement

Parameter	Description
M	The gap distance between two labels 0 ≤ m ≤1 (inch), 0 ≤ m ≤ 25.4 (mm) 0 ≤ m ≤5 (inch), 0 ≤ m ≤ 127 (mm)
N	The offset distance of the gap n ≤ label length (inch or mm)
0, 0	Continuous label

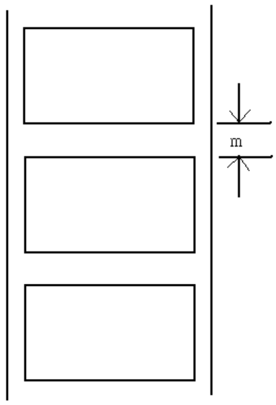
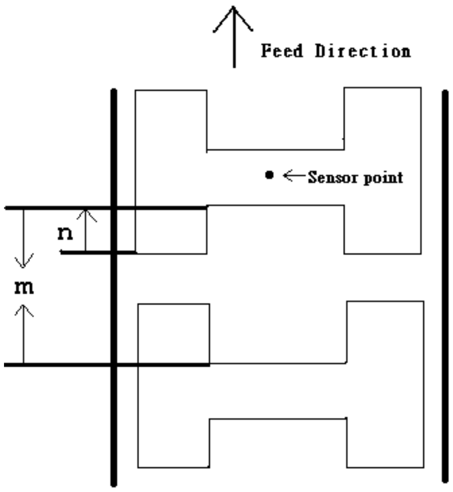




Note :

- ***200 DPI : 1 mm = 8 dots***
  300 DPI : 1mm = 12 dots
- ***For metric and dot systems, there must be a space between parameter and mm.***
- ***When the sensor type is changed from "Black Mark" to "GAP", please send the "GAP" command to the printer first.***

Example

Sample Code	Result
Normal gap - English system (inch): **GAP 0.12,0** - Metric system (mm): **GAP 3 mm,0 mm** - Continuous label: **GAP 0,0**	**Normal gap** 
Special gap - English system (inch) **GAP 0.30,0.10** - Metric system (mm) **GAP 7.62 mm,2.54 mm**	**Special gap** 

See Also

SIZE, BLINE

5.3 GAPDETECT

Description

This command feeds the paper through the gap sensor in an effort to determine the paper and gap sizes, respectively. This command references the user’s approximate measurements. If the measurements conflict with the actual size, the GAPDETECT command will not work properly. This calibration method can be applied to the labels with pre-printed logos or texts.

Syntax

GAPDETECT [x,y]

Parameter	Description
X	Paper length (in dots)
Y	Gap length (in dots)

Note:
If the x, y parameters are ignored then the printer will calibrate and determine the paper length and gap size automatically.

See Also

GAP, SIZE, BLINEDETECT, AUTODETECT

5.4 BLINEDETECT

Description

This command feeds the paper through the black mark sensor in an effort to determine the paper and black mark sizes, respectively. This command references the user's approximate measurements. If the measurements conflict with the actual size, the BLINEDETECT command will not work properly. This calibration method can be applied to the labels with pre-printed logos or texts.

Syntax

BLINEDETECT [x,y]

Parameter	Description
x	Paper length (in dots)
y	Gap length (in dots)

Note:
If the x, y parameters are ignored then the printer will calibrate and determine the paper length and gap size automatically.

See Also

GAP, SIZE, GAPDETECT, AUTODETECT

5.5 AUTODETECT

Description

This command feeds the paper through the gap/black mark sensor in an effort to determine the paper and gap/black mark sizes, respectively. This command references the user’s approximate measurements. If the measurements conflict with the actual size, the AUTODETECT command will not work properly. This calibration method can be applied to the labels with pre-printed logos or texts.

Syntax

AUTODETECT [x,y]

Parameter	Description
x	Paper length (in dots)
y	Gap length (in dots)

Note:

- ***If the x, y parameters are ignored then the printer will calibrate and determine the paper length and gap/black mark size automatically.***
- ***When using this command, the printer will detect the label by the proper sensor type so please don’t set the command GAP or BLINE in your program.***

See Also

GAP, SIZE, GAPDETECT, BLINEDETECT

5.6 BLINE

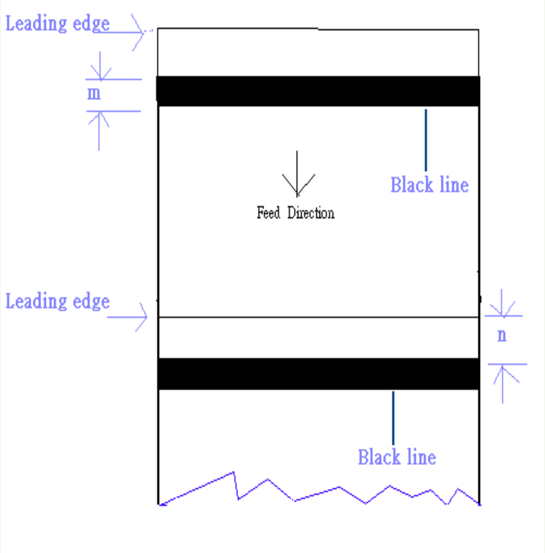
Description

This command sets the height of the black line and the user-defined extra label feeding length each form feed takes.

Syntax

BLINE m,n	English system (inch)
BLINE m mm,n mm	Metric system (mm)
BLINE m dot,n dot	Dot measurement

Parameter	Description
m	The height of black line either in inch or mm 0 ≤ m ≤ 1 (inch), 0 ≤ m ≤ 25.4 (mm) 0 ≤ m ≤5 (inch), 0 ≤ m ≤ 127 (mm)
n	The extra label feeding length 0 ≤ n ≤ label length
0,0	Continuous label



Note:

- ***For metric system, there must be a space between parameter and mm.***
- ***When the sensor type is changed from "GAP" to "Black Mark", please send the "BLINE" command to the printer first.***
- ***200 DPI : 1 mm = 8 dots***
  300 DPI : 1mm = 12 dots

Example

Sample Code

- English system (inch):

```
BLINE 0.20,0.50
```

- Metric system (mm):

```
BLINE 5.08 mm,12.7 mm
```

See Also

SIZE, GAP

5.7 OFFSET

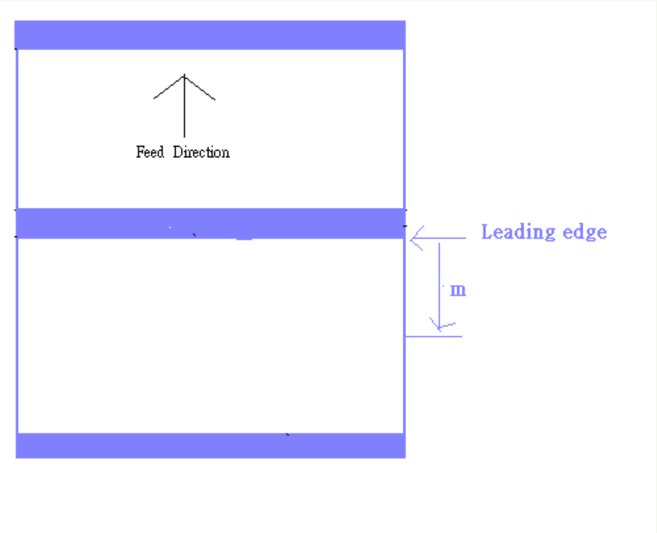
Description

This command defines the selective, extra label feeding length each form feed takes, which, especially in peel-off mode and cutter mode, is used to adjust label stop position, so as for label to register at proper places for the intended purposes. The printer back tracks the extra feeding length before the next run of printing.

Syntax

OFFSET m	English system (inch)
OFFSET m mm	Metric system (mm)
OFFSET m dot	Dot measurement

Parameter	**Description**
m	The offset distance (inch or mm) -1 ≤ m ≤ 1 (inch)



CAUTION:

- ***Impropriety offset value may cause paper jam.***
- ***For metric system, there must be a space between parameter and mm.***
- ***200 DPI : 1 mm = 8 dots***
  300 DPI : 1mm = 12 dots

Example

Sample Code

- English system (inch):

```
OFFSET 0.5
```

- Metric system (mm):

```
OFFSET 12.7 mm
```

See Also

SIZE, GAP, SET PEEL, SET CUTTER

5.8 SPEED

Description

This command defines the print speed.

Syntax

SPEED n

Parameter	**Description**
n	Printing speed in inch per second

Model / IPS	1	1.5	2	2.5	3	3.5	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
TD-4420TN			**V**		**V**		**V**	**V**	**V**												
TD-4520TN			**V**		**V**		**V**	**V**													
TD-4650TNWB, TD-4650TNWBR		**V**	**V**		**V**		**V**	**V**	**V**	**V**	**V**										
TD-4750TNWB, TD-4750TNWBR		**V**	**V**		**V**		**V**	**V**	**V**												

Example

Sample code

```
SPEED 10
```

See Also

DENSITY

5.9 DENSITY

Description

This command sets the printing darkness.

Syntax

DENSITY n

Parameter	Description
n	0~15 0: specifies the lightest level 15: specifies the darkest level

Note:
Default DENSITY setting is 8.

Example

Sample code

```
DENSITY 7
```

5.10 DIRECTION and Mirror Image

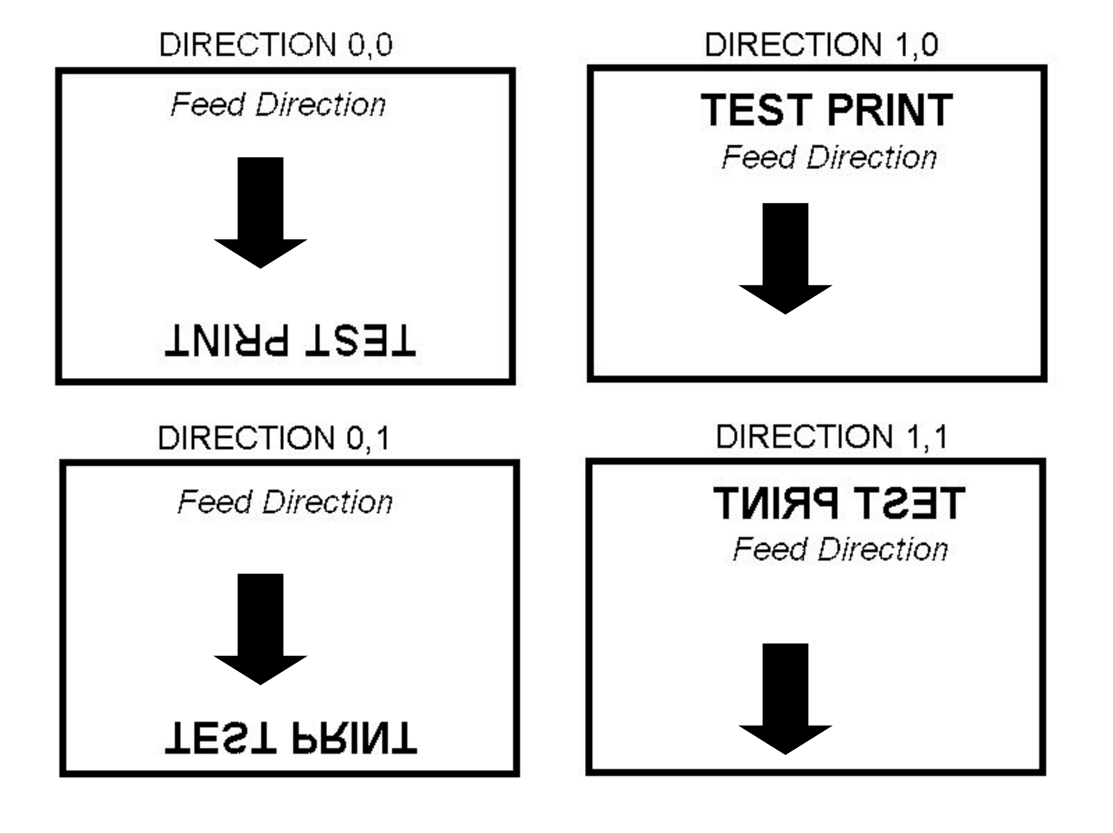
Description

This command defines the printout direction and mirror image. This will be stored in the printer memory.

Syntax

DIRECTION n[,m]

Parameter	Description
n	0 or 1. Please refer to the illustrations below
m	0: Print normal image 1: Print mirror image



Example

Sample code

- **DIRECTION 0**
- **DIRECTION 0,1**

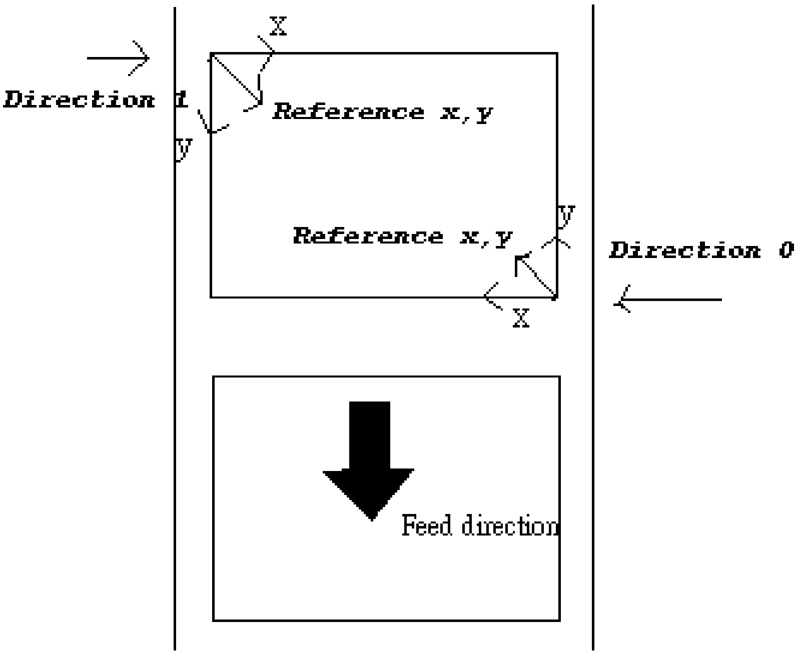
See Also

REFERENCE

5.11 REFERENCE

Description

This command defines the reference point of the label. The reference (origin) point varies with the print direction, as shown:



Syntax

REFERENCE x, y

Parameter	Description
x	Horizontal coordinate (in dots)
y	Vertical coordinate (in dots)

Note:
200 DPI: 1 mm = 8 dots
300 DPI: 1 mm = 12 dots

Example

Sample code

```
REFERENCE 10,10
```

See Also

DIRECTION

5.12 SHIFT

Description

This command moves the label's horizontal and vertical position. A positive value moves the label further from the printing direction; a negative value moves the label towards the printing direction.

Syntax

```
SHIFT [x,] y
```

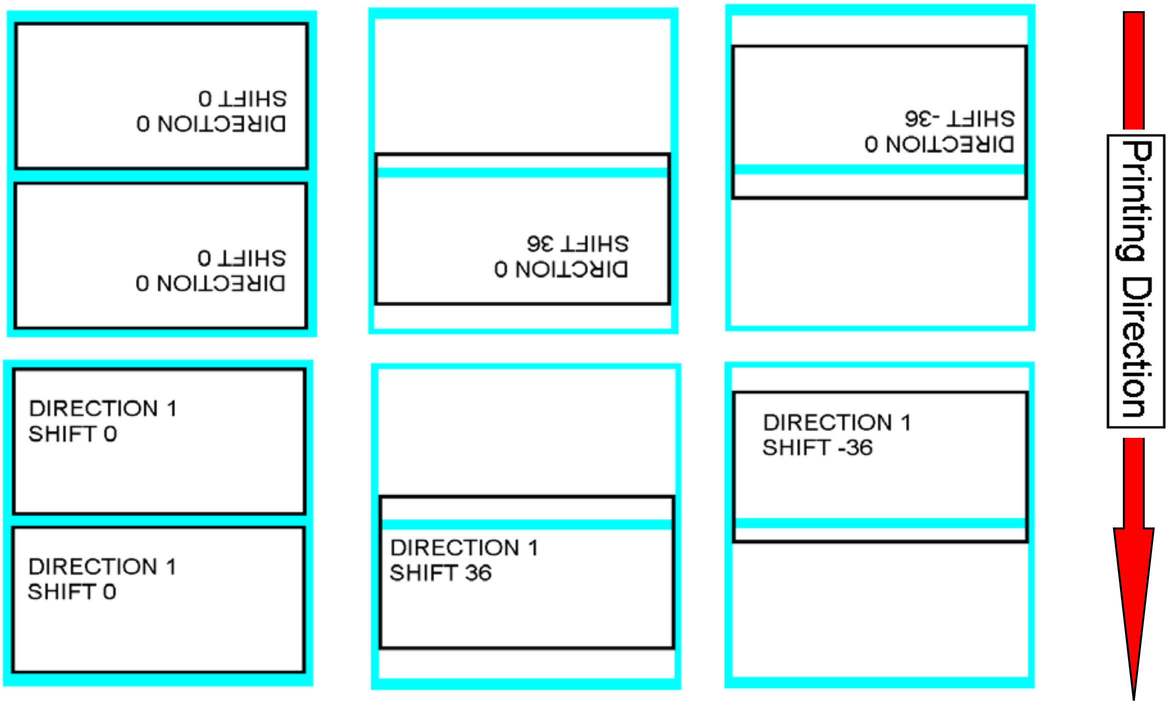
Parameter	Description
x	Optional. The maximum value is 1 inch. For 200 dpi printers, the range is –203 to 203; for 300 dpi printers, the range is –300 to 300. The unit is dot.
Y	The maximum value is 1 inch. For 200 dpi printers, the range is –203 to 203; for 300 dpi printers, the range is –300 to 300. The unit is dot.

Example

Sample Code

```
SIZE 4,2.5
GAP 2 mm,0
DIRECTION 0
SHIFT 36
OFFSET 0
CLS
TEXT 400,200, "3",0,1,1, "DIRECTION 0"
TEXT 400,250, "3",0,1,1, "SHIFT 36"
BOX 10,0,780,490,8
PRINT 3,1
```

Result



See Also

OFFSET, REFERENCE

5.13 COUNTRY

Description

This command orients the keyboard for use in different countries via defining special characters on the external device (keyboard etc).

Syntax

COUNTRY n

Parameter	Description
n	001: USA 002: Canadian-French 003: Spanish (Latin America) 031: Dutch 032: Belgian 033: French (France) 034: Spanish (Spain) 036: Hungarian 038: Yugoslavian 039: Italian 041: Switzerland 042: Slovak 044: United Kingdom 045: Danish 046: Swedish 047: Norwegian 048: Polish 049: German 055: Brazil 061: English (International) 351: Portuguese 358: Finnish

Example

Sample Code

```
COUNTRY 001
```

See Also

CODEPAGE, ~!I

5.14 CODEPAGE

Description

This command defines the code page of international character set.

Syntax

CODEPAGE n

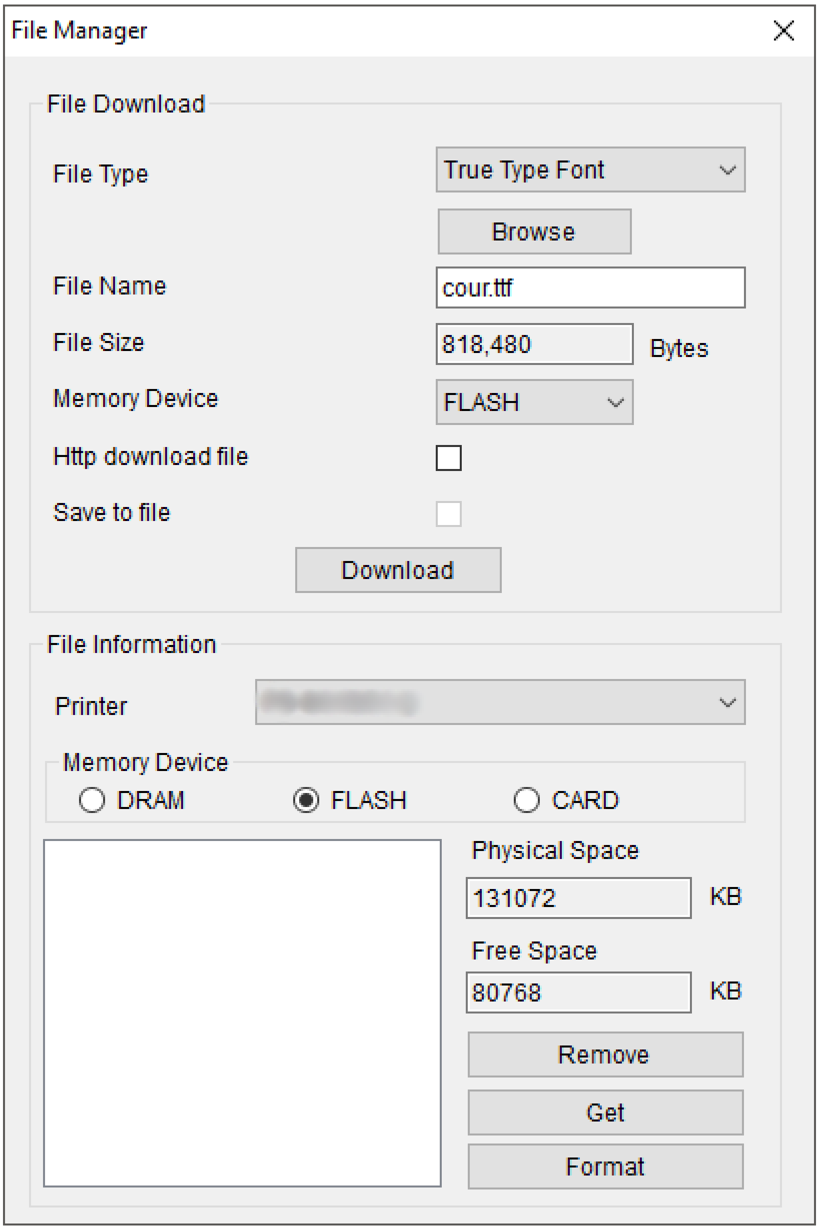
Parameter	Description
n	Name or number of code page, which can be divided into 7-bit code page and 8-bit code page.

7-bit code page		8-bit code page		Windows code page		ISO code page	
n	Name	n	Name	n	Name	n	Name
USA	USA	**437**	United States	**1250**	Central Europe	**8859-1**	Latin 1
BRI	British	**737**	Greek	**1251**	Cyrillic	**8859-2**	Latin 2
GER	German	**850**	Multilingual	**1252**	Latin I	**8859-3**	Latin 3
FRE	French	**851**	Greek 1	**1253**	Greek	**8859-4**	Baltic
DAN	Danish	**852**	Slavic	**1254**	Turkish	**8859-5**	Cyrillic
ITA	Italian	**855**	Cyrillic	**1255**	Hebrew	**8859-6**	Arabic
SPA	Spanish	**857**	Turkish	**1256**	Arabic	**8859-7**	Greek
SWE	Swedish	**860**	Portuguese	**1257**	Baltic	**8859-8**	Hebrew
SWI	Swiss	**861**	Icelandic	**1258**	Vietnam	**8859-9**	Turkish
		862	Hebrew	**932**	Japanese Shift-JIS	**8859-10**	Latin 6
		863	Canadian/French	**936**	Simplified Chinese GBK	**8859-15**	Latin 9
		864	Arabic	**949**	Korean		
		865	Nordic	**950**	Traditional Chinese Big5		
		866	Russian	**UTF-8**	UTF 8		
		869	Greek 2				

Note:
DATA LENGTH determines 7-bit or 8-bit communications parameter.

Example

Download the COUR.TTF into printer by BPM



File Manager

File Download

File Type: True Type Font

File Name: cour.ttf

File Size: 818,480 Bytes

Memory Device: FLASH

Http download file □

Save to file □

File Information

Printer: [redacted]

Memory Device: ○ DRAM ● FLASH ○ CARD

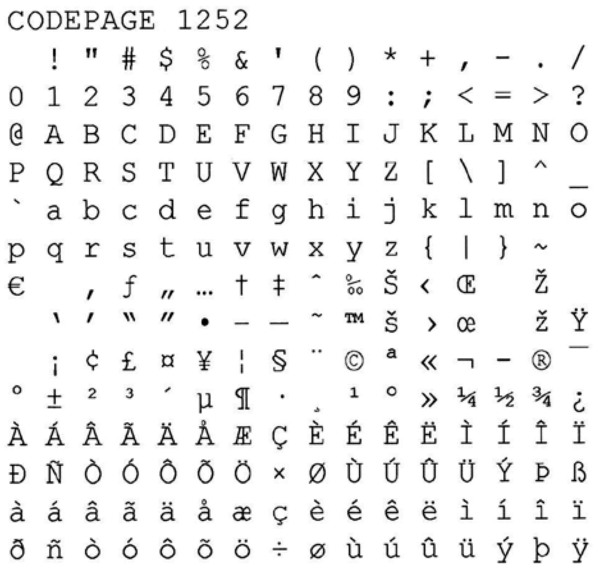
Physical Space: 131072 KB

Free Space: 80768 KB

Sample Code

```
DOWNLOAD "TEST.BAS"

str1$ = " "
J = 0
y = 50

CODEPAGE 1252
SIZE 4,3
GAP 0,0
DIRECTION 1
CLS
TEXT
10,10,"COUR.TTF",0,12,12,"CODEPAGE
1252"
FOR I=32 TO 255
str1$=str1$+CHR$(I) + " "
J=J+1
IF J=16 THEN GOSUB drawTEXT
NEXT

PRINT 1
END

drawTEXT:
TEXT 10,y,"COUR.TTF",0,12,12,str1$
str1$=" "
J=0
y=y+40
RETURN
EOP
TEST
```

Result



```
CODEPAGE 1252
  ! " # $ % & ' ( ) * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [ \ ] ^ _
` a b c d e f g h i j k l m n o
p q r s t u v w x y z { | } ~
€   ‚ ƒ „ … † ‡ ˆ ‰ Š ‹ Œ   Ž
  ‘ ’ “ ” • – — ˜ ™ š › œ   ž Ÿ
  ¡ ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ - ® ¯
° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿
À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï
Ð Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß
à á â ã ä å æ ç è é ê ë ì í î ï
ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ
```

See Also

COUNTRY, ~!I

5.15 CLS

Description

This command clears the image buffer.

Syntax

CLS

Parameter	Description
None	N/A

Note:
This command must be placed after SIZE command.

Example

Sample code

```
CLS
```

See Also

SIZE, GAP, BLINE

5.16 FEED

Description

This command feeds label with the specified length. The length is specified by dot.

Syntax

FEED n

Parameter	Description
n	unit: dot 1 ≤ n ≤ 9999

Note:
200 DPI: 1 mm = 8 dots
300 DPI: 1 mm = 12 dots

Example

Sample code

```
FEED 40
```

See Also

BACKFEED, SIZE, GAP, BLINE, HOME, FORMFEED

5.17 BACKFEED & BACKUP

Description

This command feeds the label in reverse. The length is specified by dot.

Syntax

BACKFEED n	FBPL printers only

Parameter	Description
n	unit: dot 1 ≤ n ≤ 9999

Note:
200 DPI: 1 mm = 8 dots
300 DPI: 1 mm = 12 dots

CAUTION:
Impropriety back feed value may cause paper jam or wrinkle.

Example

Sample code

- FBPL printers

```
BACKFEED 40
```

See Also

FEED, SIZE, GAP, BLINE, HOME, FORMFEED

5.18 FORMFEED

Description

This command feeds label to the beginning of next label.

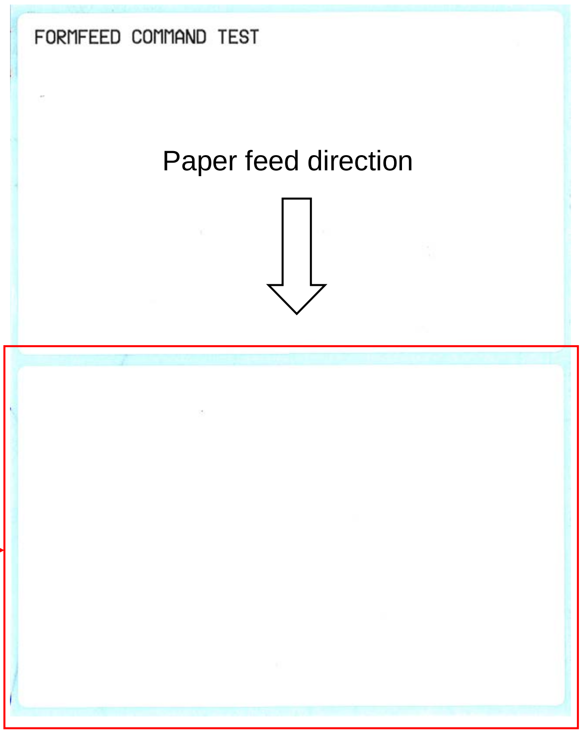
Syntax

FORMFEED

Parameter	Description
None	N/A

Note:
This command must be placed after SIZE command.

Example

Sample code	Result
SIZE 4,2.5 GAP 2 mm,0 DIRECTION 1 FORMFEED CLS TEXT 25,25, "3",0,1,1, "FORMFEED COMMAND TEST" PRINT 1,1	

See Also

FEED, SIZE, GAP, BLINE, HOME, BACKFEED

5.19 HOME

Description

This command will feed label until the internal sensor has determined the origin. Size and gap of the label should be defined before using this command.

Syntax

```
HOME
```

Parameter	Description
None	N/A

For FBPL programming printer: Feed label to origin position

Example

Sample code

```
SIZE 4,2.5
GAP 2 mm,0
SET COUNTER @0 +1
@0="000001"
HOME
CLS
BOX 1,1,360,65,12
TEXT 25,25, "3",0,1,1, "HOME COMMAND TEST"
TEXT 25,80, "3",0,1,1,@0
PRINT 3,1
```

See Also

FEED, SIZE, GAP, BLINE, FORMFEED

5.20 PRINT

Description

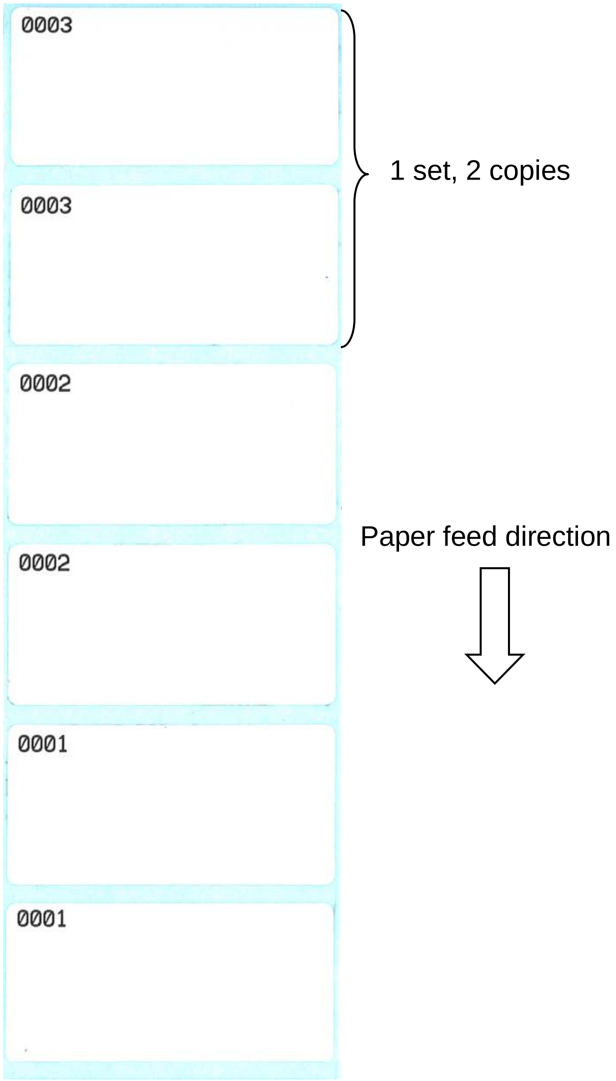
This command prints the label format currently stored in the image buffer.

Syntax

PRINT m[,n]

Parameter	Description
m	Specifies how many sets of labels will be printed. 1 ≤ m ≤ 999999999
n	Specifies how many copies should be printed for each particular label set. 1 ≤ n ≤ 999999999

Example

Sample code	Result
SIZE 50 mm,25 mm **GAP 3 mm,0** **DIRECTION 1** **SET COUNTER @1 1** **@1="0001"** **CLS** **TEXT 10,10, "3",0,1,1,@1** **PRINT 3,2**	

See Also

SET COUNTER, INPUT, DOWNLOAD

5.21 SOUND

Description

This command controls the sound frequency of the beeper. There are 10 levels of sounds. The timing control can be set by the "interval" parameter.

Syntax

SOUND level,interval

Parameter	Description
level	Sound level: 0~9
interval	Sound interval: 1~4095

Example

Sample code

```
· SOUND 5,200
· SOUND 3,200
· SOUND 3,200
· SOUND 4,200
· SOUND 2,200
· SOUND 2,200
· SOUND 1,200
· SOUND 2,200
· SOUND 3,200
· SOUND 4,200
· SOUND 5,200
```

5.22 CUT

Description

This command activates the cutter to immediately cut the labels without back feeding the label.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
CUT
```

Parameter	Description
None	N/A

Example

Sample code

```
SIZE 3,3
GAP 0,0
CLS
BOX 0,0,866,866,5
TEXT 100,100, "5",0,1,1, "FEED & CUT"
TEXT 100,200, "5",0,1,1, "300 DPI"
PRINT 1,1
FEED 260
CUT
```

See Also

SET CUTTER, SET BACK, SET PARTITAL_CUTTER

5.23 LIMITFEED

Description

If the gap sensor is not set to a suitable sensitivity while feeding labels, the printer will not be able to locate the correct position of the gap. This command stops label feeding and makes the red LED flash if the printer does not locate gap after feeding the length of one label plus one preset value.

Syntax

LIMITFEED n[,minpaper,maxgap]	English system (inch)
LIMITFEED n mm[,minpaper mm,maxgap mm]	Metric system (mm)
LIMITFEED n dot[,minpaper dot,maxgap dot]	Dot measurement

Parameter	**Description**
N	The maximum length for sensor detecting
Minpaper	The minimum length of paper
Maxgap	The maximum length of gap

Note:

- ***The setting will remain resident in memory.***
- ***For metric system, there must be a space between parameter n and mm.***
- ***The default value is 10 inches when printer initializes.***

Example

Sample code

- English system (inch)

```
LIMITFEED 12
```

5.24 SELFTEST

Description

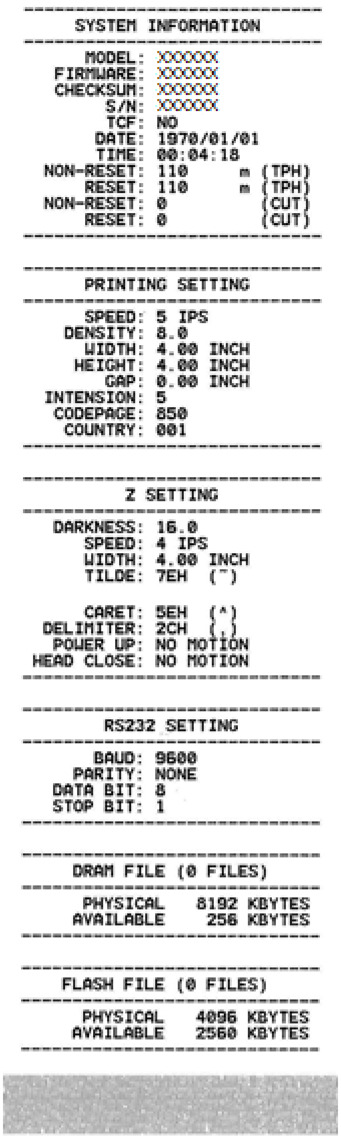
At this command, the printer will print out the printer information.

Syntax

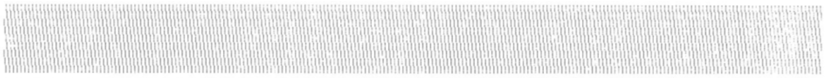
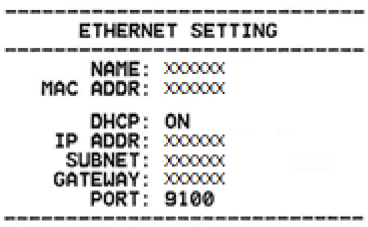
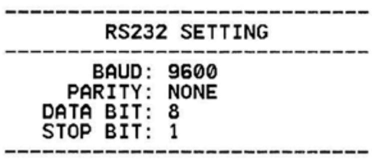
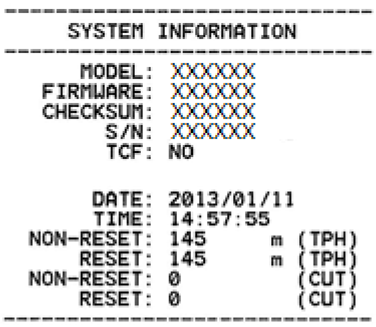
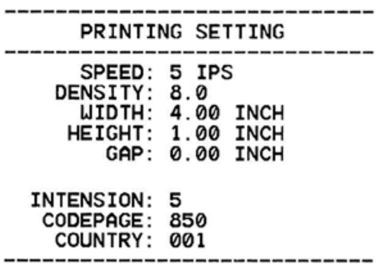
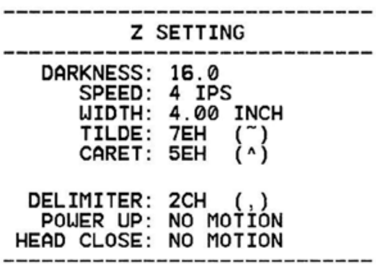
SELFTEST [page]

Parameter	Description
page	**omitted**: Print a self-test page with whole printer information. **PATTERN**: Print a pattern to check the status of print head heat line. **ETHERNET**: Print a self-test page with Ethernet settings. **RS232**: Print a self-test page with RS-232 settings. **SYSTEM**: Print a self-test page with printer settings. **Z**: Print a self-test page with emulated language settings.

Example

Sample code	Result
SELFTEST	

```
------------------------------
      SYSTEM INFORMATION
------------------------------
       MODEL: XXXXXX
    FIRMWARE: XXXXXX
    CHECKSUM: XXXXXX
         S/N: XXXXXX
         TCF: NO
        DATE: 1970/01/01
        TIME: 00:04:18
   NON-RESET: 110      m (TPH)
       RESET: 110      m (TPH)
   NON-RESET: 0          (CUT)
       RESET: 0          (CUT)
------------------------------

------------------------------
       PRINTING SETTING
------------------------------
       SPEED: 5 IPS
     DENSITY: 8.0
       WIDTH: 4.00 INCH
      HEIGHT: 4.00 INCH
         GAP: 0.00 INCH
   INTENSION: 5
    CODEPAGE: 850
     COUNTRY: 001
------------------------------

------------------------------
          Z SETTING
------------------------------
    DARKNESS: 16.0
       SPEED: 4 IPS
       WIDTH: 4.00 INCH
       TILDE: 7EH  (~)

       CARET: 5EH  (^)
   DELIMITER: 2CH  (,)
    POWER UP: NO MOTION
  HEAD CLOSE: NO MOTION
------------------------------

------------------------------
        RS232 SETTING
------------------------------
        BAUD: 9600
      PARITY: NONE
    DATA BIT: 8
    STOP BIT: 1
------------------------------

------------------------------
     DRAM FILE (0 FILES)
------------------------------
      PHYSICAL    8192 KBYTES
     AVAILABLE     256 KBYTES
------------------------------

------------------------------
    FLASH FILE (0 FILES)
------------------------------
      PHYSICAL    4096 KBYTES
     AVAILABLE    2560 KBYTES
------------------------------
```

SELFTEST PATTERN	
SELFTEST ETHERNET	 ----------------------------- ETHERNET SETTING ----------------------------- NAME: xxxxxx MAC ADDR: xxxxxx DHCP: ON IP ADDR: xxxxxx SUBNET: xxxxxx GATEWAY: xxxxxx PORT: 9100 -----------------------------
SELFTEST RS232	 ----------------------------- RS232 SETTING ----------------------------- BAUD: 9600 PARITY: NONE DATA BIT: 8 STOP BIT: 1 -----------------------------
SELFTEST SYSTEM	 ----------------------------- SYSTEM INFORMATION ----------------------------- MODEL: XXXXXX FIRMWARE: XXXXXX CHECKSUM: XXXXXX S/N: XXXXXX TCF: NO DATE: 2013/01/11 TIME: 14:57:55 NON-RESET: 145 m (TPH) RESET: 145 m (TPH) NON-RESET: 0 (CUT) RESET: 0 (CUT) -----------------------------
SELFTEST PRINTER	 ----------------------------- PRINTING SETTING ----------------------------- SPEED: 5 IPS DENSITY: 8.0 WIDTH: 4.00 INCH HEIGHT: 1.00 INCH GAP: 0.00 INCH INTENSION: 5 CODEPAGE: 850 COUNTRY: 001 -----------------------------
SELFTEST Z	 ----------------------------- Z SETTING ----------------------------- DARKNESS: 16.0 SPEED: 4 IPS WIDTH: 4.00 INCH TILDE: 7EH (~) CARET: 5EH (^) DELIMITER: 2CH (,) POWER UP: NO MOTION HEAD CLOSE: NO MOTION -----------------------------

5.25 EOJ

Description

Let the printer wait until process of commands (before EOJ) be finished then go on the next command.

Syntax

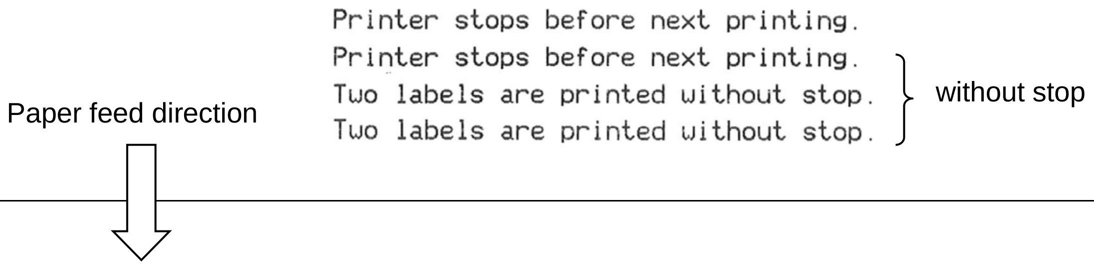
```
EOJ
```

Example

Sample Code

```
SIZE 4,0.2
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10,"3",0,1,1,"Two labels are printed without stop."
PRINT 1
PRINT 1

SIZE 4,0.2
GAP 0,0
CLS
TEXT 10,10,"3",0,1,1,"Printer stops before next printing."
PRINT 1
EOJ
PRINT 1
```

Result



```
Printer stops before next printing.
Printer stops before next printing.
Two labels are printed without stop.
Two labels are printed without stop.
```

5.26 DELAY

Description

Let the printer wait specific period of time then go on next command.

Syntax

DELAY ms

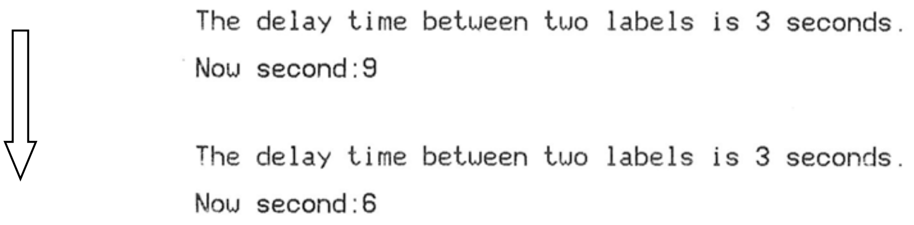
Parameter	Description
ms	The specific period of time. Unit is millisecond. 1000 ms = 1 second.

Example

Sample Code

```
SIZE 4,0.7
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10,"3",0,1,1,"The delay time between two labels is 3 seconds."
TEXT 10,60,"3",0,1,1,"Now second:" +@SECOND
PRINT 1
DELAY 3000
PRINT 1
```

Result



```
The delay time between two labels is 3 seconds.
Now second:9

The delay time between two labels is 3 seconds.
Now second:6
```

5.27 DISPLAY

Description

This command can show the image, which is in printer’s image buffer, on LCD panel.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
DISPLAY IMAGE/OFF/CLS/forecolor,backcolor/x,y,width,height/x,y, "bmpfile"/x,y, "font","content"
```

Parameter	Description
IMAGE	Show the image in printer’s image buffer on LCD panel.
OFF	Disable this function.
CLS	Show the background color and clear the items in printer’s image buffer on LCD panel
forecolor,backcolor	Set the color (decimal) for item and background in printer’s image buffer on LCD panel
x,y,width,height	Draw the bar in printer’s image buffer on LCD panel
x,y,”bmpfile”	Show the .bmp in printer’s image buffer on LCD panel
x,y,”font”,”content”	Show the text in printer’s image buffer on LCD panel

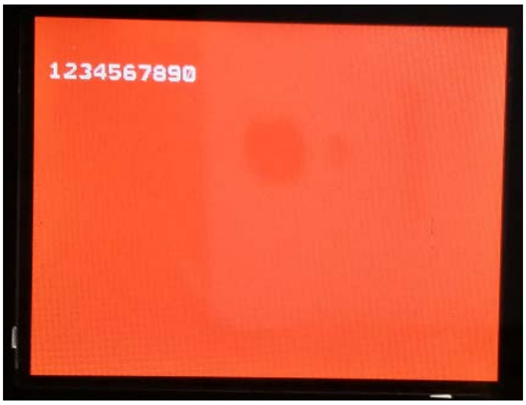
forecolor	RGB color code for text or bar (decimal)
backcolor	RGB color code for background (decimal)
x	Horizontal multiplication
y	Vertical multiplication
width	frame width
height	frame height
bmpfile	.bmp file name
font	Font name
content	Content of text string

Note:
This command only can be performed on the printer with LCD display.

Example

Sample code	Result
CLS TEXT 1,10, "1",0,1,1, "Image on LCD" TEXT 1,30, "1",0,1,1, "1234567890" DISPLAY IMAGE DELAY 5000 DISPLAY OFF	

```
CLS
DISPLAY 15128749,16711680
DISPLAY CLS
DISPLAY 10,30, "1","1234567890"
DELAY 5000
DISPLAY OFF
```



5.28 INITIALPRINTER

Description

This command can restore printer settings to defaults.

Syntax

```
INITIALPRINTER
```

Parameter	Description
None	N/A

Example

Sample code

```
INITIALPRINTER
```

5.29 MENU

Description

This command can design user's own menu with a database resident on the printer.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

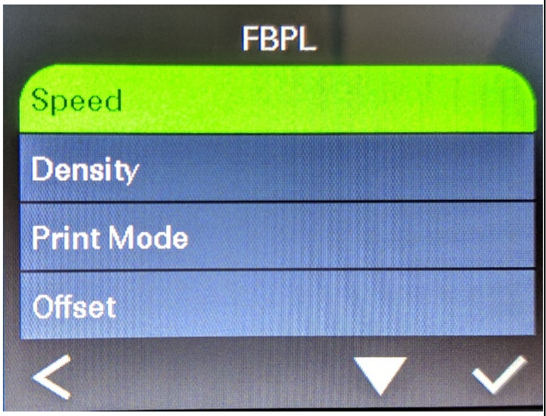
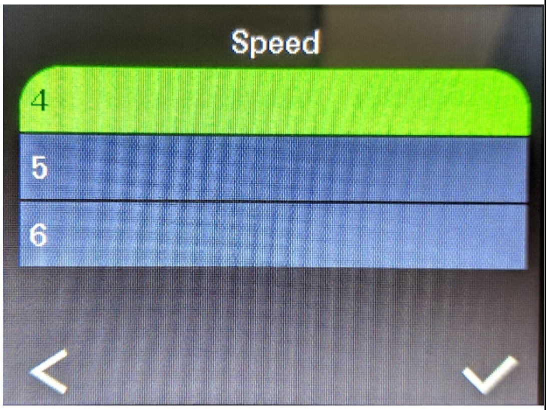
Syntax

MENU title$, list$, selected

Parameter	Description
title$	The title string is shown on LCD screen.
list$	List of items, separated by CRLF.
selected	It must be a variable to get the result of selection. When selected is 0, the operator has hit ESC (USB keyboard) or MENU button.

Note:
-This command only can be performed on the printer with LCD display.

Example

Sample code	Result
DOWNLOAD F,"FBPL" Speed Density Print Mode Offset Country EOP DOWNLOAD F,"Speed" 4 5 6 EOP DOWNLOAD F,"Density" 6 7 8 9 10 11 12 EOP DOWNLOAD F,"Print Mode" NONE TEAR OFF PEEL OFF CUT OFF EOP DOWNLOAD F,"Country" 007 031 033	 

```
034
045
EOP

DOWNLOAD F,"DEMO.BAS"

DPI =
VAL(GETSETTING$("SYSTEM","INFORMATION","DPI"
))

:MAINLOOP
OPEN "FBPL",0
LIST$ = FREAD$(0, LOF("FBPL"))
CLOSE 0
MENU "FBPL", LIST$, OPTION$

IF LEN(OPTION$) = 0 THEN END

IF OPTION$ = "Speed" THEN SETTING$ =
GETSETTING$("CONFIG","FBPL","SPEED")
IF OPTION$ = "Density" THEN SETTING$ =
GETSETTING$("CONFIG","FBPL","DENSITY")
IF OPTION$ = "Print Mode" THEN SETTING$ =
GETSETTING$("CONFIG","FBPL","PRINT MODE")
IF OPTION$ = "Offset" THEN SETTING$ =
GETSETTING$("CONFIG","FBPL","OFFSET")
IF OPTION$ = "Country" THEN SETTING$ =
GETSETTING$("CONFIG","FBPL","COUNTRY CODE")

IF LOF(OPTION$) <> 0 THEN
        OPEN OPTION$,0
        LIST$ = FREAD$(0, LOF(OPTION$))
        CLOSE 0
        MENU OPTION$, LIST$, SETTING$
ELSE
        IF OPTION$ = "Offset" THEN INPUT "Offset",
SETTING$
ENDIF

IF LEN(SETTING$) <> 0 THEN
        IF OPTION$ = "Speed" THEN SPEED
VAL(SETTING$)
        IF OPTION$ = "Density" THEN DENSITY
VAL(SETTING$)
        IF OPTION$ = "Print Mode" THEN GOSUB
SET_PRINT_MODE
        IF OPTION$ = "Offset" THEN OFFSET
VAL(SETTING$) / DPI
        IF OPTION$ = "Country" THEN GOSUB
SET_COUNTRY
ENDIF

GOTO MAINLOOP

:SET_PRINT_MODE
IF SETTING$ = "NONE" THEN SET TEAR OFF
IF SETTING$ = "TEAR OFF" THEN SET TEAR ON
IF SETTING$ = "PEEL OFF" THEN SET PEEL ON
IF SETTING$ = "CUT OFF" THEN SET CUTTER ON
RETURN

:SET_COUNTRY
IF SETTING$ = "007" THEN COUNTRY 007
IF SETTING$ = "031" THEN COUNTRY 031
IF SETTING$ = "033" THEN COUNTRY 033
IF SETTING$ = "034" THEN COUNTRY 034
IF SETTING$ = "045" THEN COUNTRY 045
RETURN

EOP

RUN "DEMO.BAS"
```

6 Label Formatting Commands

6.1 BAR

Description

This command draws a bar on the label format.

Syntax

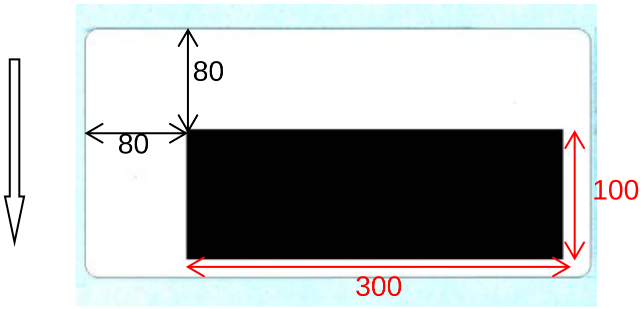
BAR x,y,width,height

Parameter	**Description**
x	The upper left corner x-coordinate (in dots)
y	The upper left corner y-coordinate (in dots)
width	Bar width (in dots)
height	Bar height (in dots)

Note:

- ***200 DPI : 1 mm = 8 dots***
  300 DPI : 1 mm = 12 dots
- ***Recommended max. bar height is 12 mm at 4" width. Bar height over 12 mm may damage the power supply and affect the print quality.***
- ***Max. print ratio is different for each printer model. Desktop and industrial printer print ratio is limited to 20% and 30% respectively.***

Example

Sample code	**Result**
SIZE 50 mm,25 mm **GAP 3 mm,0** **DIRECTION 1** **CLS** **BAR 80,80,300,100** **PRINT 1,1**	

See Also

BOX

6.2 BARCODE

Description

This command prints 1D barcodes. The available barcodes are listed below:

Code Type	Description	Narrow : Width					Max. data length
		1:1	1:2	1:3	2:5	3:7	
128	Code 128, switching code subset automatically.	V					
128M	Code 128, switching code subset manually.	V					
EAN128	EAN128, switching code subset automatically.	V					
EAN128M	EAN128M, switching code subset automatically.	V					
25	Interleaved 2 of 5.		V	V	V		Length is even
25C	Interleaved 2 of 5 with check digit.		V	V	V		Length is odd
25S	Standard 2 of 5.		V	V	V		
25I	Industrial 2 of 5.		V	V	V		
39	Code 39, switching standard and full ASCII mode automatically.		V	V	V		
39C	Code 39 with check digit.		V	V	V		
93	Code 93.			V			
EAN13	EAN 13.	V					12
EAN13+2	EAN 13 with 2 digits add-on.	V					14
EAN13+5	EAN 13 with 5 digits add-on.	V					17
EAN8	EAN 8.	V					7
EAN8+2	EAN 8 with 2 digits add-on.	V					9
EAN8+5	EAN 8 with 5 digits add-on.	V					12
CODA	Codabar.		V	V	V		
POST	Postnet.	V					5, 9, 11
UPCA	UPC-A.	V					11
UPCA+2	UPC-A with 2 digits add-on.	V					13
UPA+5	UPC-A with 5 digits add-on.	V					16
UPCE	UPC-E.	V					6
UPCE+2	UPC-E with 2 digits add-on.	V					8
UPE+5	UPC-E with 5 digits add-on.	V					11
MSI	MSI.		V	V	V		
MSIC	MSI with check digit.		V	V	V		
PLESSEY	PLESSEY.		V	V	V		
CPOST	China post.					V	
ITF14	ITF14.		V	V	V		13
EAN14	EAN14.	V					13
11	Code 11.		V	V	V		
TELEPEN	Telepen..		V	V	V		
TELEPENN	Telepen number.		V	V	V		
PLANET	Planet.	V					
CODE49	Code 49.	V					
DPI	Deutsche Post Identcode.		V	V	V		11
DPL	Deutsche Post Leitcode.		V	V	V		13
LOGMARS	A special use of Code 39.		V	V	V		

Syntax

BARCODE X,Y, "code type",height,human readable,rotation,narrow,wide,[alignment,] "content "

<table>
<tr><th>Parameter</th><th></th><th>Description</th></tr>
<tr><td>X</td><td></td><td>Specify the x-coordinate bar code on the label</td></tr>
<tr><td>Y</td><td></td><td>Specify the y-coordinate bar code on the label</td></tr>
<tr><td>code type</td><td></td><td></td></tr>
<tr><td></td><td>128</td><td>Code 128, switching code subset A, B, C automatically</td></tr>
<tr><td></td><td>128M</td><td>Code 128, switching code subset A, B, C manually
<table>
<tr><th>Control code</th><th>A</th><th>B</th><th>C</th></tr>
<tr><td>096</td><td>FNC3</td><td>FNC3</td><td>NONE</td></tr>
<tr><td>097</td><td>FNC2</td><td>FNC2</td><td>NONE</td></tr>
<tr><td>098</td><td>SHIFT</td><td>SHIFT</td><td>NONE</td></tr>
<tr><td>099</td><td>CODE C</td><td>CODE C</td><td>NONE</td></tr>
<tr><td>100</td><td>CODE B</td><td>FNC4</td><td>CODE B</td></tr>
<tr><td>101</td><td>FNC4</td><td>CODE A</td><td>CODE A</td></tr>
<tr><td>102</td><td>FNC1</td><td>FNC1</td><td>FNC1</td></tr>
<tr><td>103</td><td colspan="3">Start (CODE A)</td></tr>
<tr><td>104</td><td colspan="3">Start (CODE B)</td></tr>
<tr><td>105</td><td colspan="3">Start (CODE C)</td></tr>
</table>
<b><i>Use "!" as a starting character for the control code followed by three control codes. If the start subset is not set, the default starting subset is B.</i></b></td></tr>
<tr><td></td><td>EAN128</td><td>Code 128, switching code subset A, B, C automatically</td></tr>
<tr><td></td><td>EAN128M</td><td>Code 128, switching code subset A, B, C manually</td></tr>
<tr><td></td><td>25</td><td>Interleaved 2 of 5</td></tr>
<tr><td></td><td>25C</td><td>Interleaved 2 of 5 with check digits</td></tr>
<tr><td></td><td>25S</td><td>Standard 2 of 5</td></tr>
<tr><td></td><td>25I</td><td>Industrial 2 of 5</td></tr>
<tr><td></td><td>39</td><td>Code 39 full ASCII for FBPL printers
Auto switch full ASCII and standard code 39 for <b><i>PLUS</i></b> models</td></tr>
<tr><td></td><td>39C</td><td>Code 39 full ASCII with check digit for FBPL printers
Auto switch full ASCII and standard code 39 for <b><i>PLUS</i></b> models</td></tr>
<tr><td></td><td>39S</td><td>Code 39 standard for FBPL printers</td></tr>
<tr><td></td><td>93</td><td>Code 93</td></tr>
<tr><td></td><td>EAN13</td><td>EAN 13</td></tr>
<tr><td></td><td>EAN13+2</td><td>EAN 13 with 2 digits add-on</td></tr>
<tr><td></td><td>EAN13+5</td><td>EAN 13 with 5 digits add-on</td></tr>
<tr><td></td><td>EAN8</td><td>EAN 8</td></tr>
<tr><td></td><td>EAN8+2</td><td>EAN 8 with 2 digits add-on</td></tr>
<tr><td></td><td>EAN8+5</td><td>EAN 8 with 5 digits add-on</td></tr>
<tr><td></td><td>CODA</td><td>Codabar</td></tr>
<tr><td></td><td>POST</td><td>Postnet</td></tr>
<tr><td></td><td>UPCA</td><td>UPC-A</td></tr>
<tr><td></td><td>UPCA+2</td><td>UPC-A with 2 digits add-on</td></tr>
<tr><td></td><td>UPCA+5</td><td>UPC-A with 5 digits add-on</td></tr>
<tr><td></td><td>UPCE</td><td>UPC-E</td></tr>
<tr><td></td><td>UPCE+2</td><td>UPC-E with 2 digits add-on</td></tr>
<tr><td></td><td>UPCE+5</td><td>UPC-E with 5 digits add-on</td></tr>
<tr><td></td><td>CPOST</td><td>China post code</td></tr>
<tr><td></td><td>MSI</td><td>MSI code</td></tr>
<tr><td></td><td>MSIC</td><td>MSI with check digit</td></tr>
<tr><td></td><td>PLESSEY</td><td>PLESSEY code</td></tr>
<tr><td></td><td>ITF14</td><td>ITF 14 code</td></tr>
<tr><td></td><td>EAN14</td><td>EAN 14 code</td></tr>
<tr><td></td><td>11</td><td>Code 11</td></tr>
<tr><td></td><td>TELEPEN</td><td>Telepen code</td></tr>
</table>

TELEPENN	Telepen code. Number only
PLANET	Planet code
CODE49	Code 49
DPI	Deutsche Post Identcode
DPL	Deutsche Post Leitcode

Height Bar code height (in dots)

human readable 0: not readable
1: human readable aligns to left
2: human readable aligns to center
3: human readable aligns to right

rotation 0 : No rotation
90 : Rotate 90 degrees clockwise
180 : Rotate 180 degrees clockwise
270 : Rotate 270 degrees clockwise

narrow Width of narrow element (in dots)

wide Width of wide element (in dots)

	narrow : wide 1:1	**narrow : wide 1:2**	**narrow : wide 1:3**	**narrow : wide 2:5**	**narrow : wide 3:7**
128	10x	-	-	-	-
EAN128	10x	-	-	-	-
EAN128M	10x				
25	-	10x	10x	5x	-
25C	-	10x	10x	5x	-
25S		10x	10x	5x	
25I		10x	10x	5x	
39	-	10x	10x	5x	-
39C	-	10x	10x	5x	-
93	-	-	10x	-	-
EAN13	8x	-	-	-	-
EAN13+2	8x	-	-	-	-
EAN13+5	8x	-	-	-	-
EAN 8	8x	-	-	-	-
EAN 8+2	8x	-	-	-	-
EAN 8+5	8x	-	-	-	-
CODA	-	10x	10x	5x	-
POST	1x	-	-	-	-
UPCA	8x	-	-	-	-
UPCA+2	8x	-	-	-	-
UPCA+5	8x	-	-	-	-
UPCE	8x	-	-	-	-
UPCE+2	8x	-	-	-	-
UPCE+5	8x	-	-	-	-
CPOST	-	-	-	-	1x
MSI	-	-	10x	-	-
MSIC			10x		-
PLESSY	-	-	10x	-	-
ITF14	-	10x	10x	5x	-
EAN14	8x	-	-	-	-
11	-	10x	10x	5x	-

alignment Specify the alignment of barcode
0 : default (Left)
1 : Left
2 : Center
3 : Right

content

Content of barcode

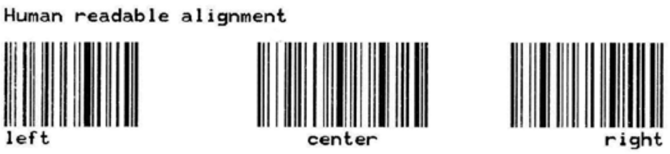
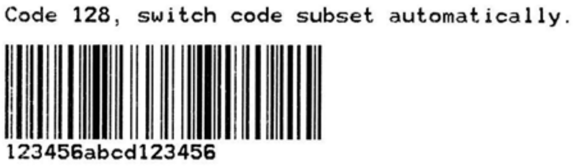
Please note that the maximum number of digits of bar code content.

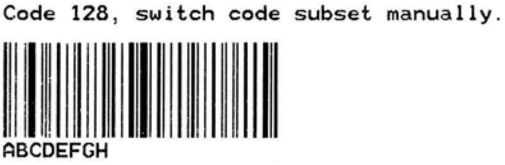
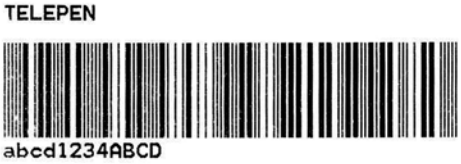
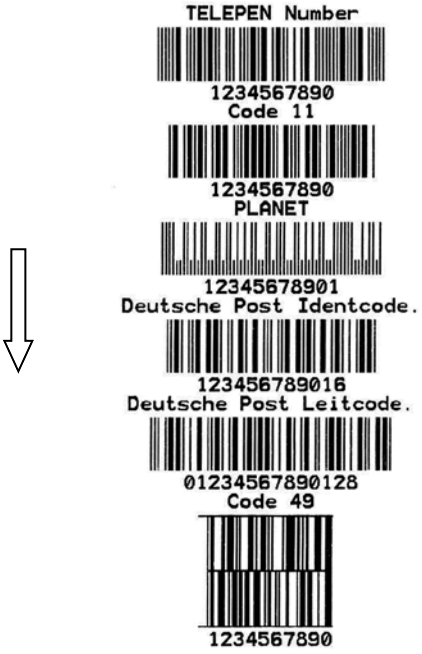
Code Type	Character sets	Max. data length
128	See Character set for CODE128.	-
128M	See Character set for CODE128.	-
EAN128	See Character set for CODE128.	-
EAN128M	See Character set for CODE128.	-
25	0123456789	Length is even.
25C	0123456789	Length is odd.
25S	0123456789	
25I	0123456789	
39 I	0123456789[Space]ABCDEFGHIJKLMNOPQRSTUVWXYZ-.$/+%	-
39 I Full ASCII	0123456789[Space]ABCDEFGHIJKLMNOPQRSTUVWXYZ!#$%&’()*+,-./:;⇔?@[¥]^_`abcdefghijklmnopqrstuvwxyz{|}~	-
93	0123456789[Space]ABCDEFGHIJKLMNOPQRSTUVWXYZ!#$%&’()*+,-./:;⇔?@[¥]^_`abcdefghijklmnopqrstuvwxyz{|}~	-
EAN13	0123456789	12
EAN13+2	0123456789	14
EAN13+5	0123456789	17
EAN8	0123456789	7
EAN8+2	0123456789	9
EAN8+5	0123456789	12
CODA	0123456789-$:/.+	-
POST	0123456789	5, 9, 11
UPCA	0123456789	11
UPCA+2	0123456789	13
UPA+5	0123456789	16
UPCE	0123456789	6
UPCE+2	0123456789	8
UPE+5	0123456789	11
MSI	0123456789	-
MSIC	0123456789	-
PLESSEY	0123456789	-
CPOST	0123456789	-
ITF14	0123456789	13
EAN14	0123456789	13
11	0123456789-	-
TELEPEN	ASCII 0 to 127	30
TELEPENN	0123456789	60
PLANET	0123456789	38
CODE49	ASCII 0 to 127	81
DPI	0123456789	11
DPL	0123456789	13
LOGMARS	0123456789[Space]ABCDEFGHIJKLMNOPQRSTUVWXYZ-.$/+%	-

Character set for CODE 128

Value	128A	128B	128C	Value	128A	128B	128C	Value	128A	128B	128C
0	space	space	00	36	D	D	36	72	BS	h	72
1	!	!	01	37	E	E	37	73	HT	i	73
2	“	“	02	38	F	F	38	74	LF	j	74
3	#	#	03	39	G	G	39	75	VT	k	75
4	$	$	04	40	H	H	40	76	FF	l	76
5	%	%	05	41	I	I	41	77	CR	m	77
6	&	&	06	42	J	J	42	78	SO	n	78
7	‘	‘	07	43	K	K	43	79	SI	o	79
8	(	(	08	44	L	L	44	80	DLE	p	80
9	)	)	09	45	M	M	45	81	DC1	q	81
10	*	*	10	46	N	N	46	82	DC2	r	82
11	+	+	11	47	O	O	47	83	DC3	s	83
12	,	,	12	48	P	P	48	84	DC4	t	84
13	-	-	13	49	Q	Q	49	85	NAK	u	85
14	.	.	14	50	R	R	50	86	SYN	v	86
15	/	/	15	51	S	S	51	87	ETB	w	87
16	0	0	16	52	T	T	52	88	CAN	x	88
17	1	1	17	53	U	U	53	89	EM	y	89
18	2	2	18	54	V	V	54	90	SUB	z	90
19	3	3	19	55	W	W	55	91	ESC	{	91
20	4	4	20	56	X	X	56	92	FS	|	92
21	5	5	21	57	Y	Y	57	93	GS	}	93
22	6	6	22	58	Z	Z	58	94	RS	~	94
23	7	7	23	59	[	[	59	95	US	DEL	95
24	8	8	24	60	¥	¥	60	96	FNC 3	FNC 3	96
25	9	9	25	61	]	]	61	97	FNC 2	FNC 2	97
26	:	:	26	62	^	^	62	98	Shift B	Shift A	98
27	;	;	27	63	_	_	63	99	Code C	Code C	99
28	<	<	28	64	NUL	`	64	100	Code B	FNC4	Code B
29	=	=	29	65	SOH	a	65	101	FNC 4	Code A	Code A
30	>	>	30	66	STX	b	66	102	FNC 1	FNC 1	FNC 1
31	?	?	31	67	ETX	c	67	103	Start Code A		
32	@	@	32	68	EOT	d	68	104	Start Code B		
33	A	A	33	69	ENQ	e	69	105	Start Code C		
34	B	B	34	70	ACK	f	70				
35	C	C	35	71	BEL	g	71				

Example

Sample Code	**Result**
SIZE 4,1 **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 10,10, "2",0,1,1, "Human readable alignment"** **BARCODE 10,50, "128",100,1,0,2,2,"left"** **BARCODE 310,50, "128",100,2,0,2,2,"center"** **BARCODE 610,50, "128",100,3,0,2,2,"right"** **PRINT 1**	
SIZE 4,1 **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 10,10, "2",0,1,1, "Code 128, switch code subset automatically. "** **BARCODE 10,50, "128",100,1,0,2,2, "123456abcd123456"** **PRINT 1**	 Code 128, switch code subset automatically. 123456abcd123456

SIZE 4,1 GAP 0,0 DIRECTION 1 CLS TEXT 10,10, "2",0,1,1, "Code 128, switch code subset manually." BARCODE 10,50, "128M",100,1,0,2,2, "!104!096ABCD!101EFGH" PRINT 1 ***Note:*** ***The above example of code 128M encoded with CODE B start character. The next character will be the code 128 function character FNC3 which is then followed by the ABCD characters and EFGH characters encoded as CODE A subset.***	
SIZE 4,1 GAP 0,0 DIRECTION 1 CLS TEXT 10,10, "2",0,1,1, "TELEPEN" BARCODE 10,50, "TELEPEN",100,1,0,2,6, "abcd1234ABCD" PRINT 1	
SIZE 4,4 GAP 0,0 DIRECTION 1 CLS TEXT 400,26, "2",0,1,1,2, "TELEPEN Number" BARCODE 400,50, "TELEPENN",60,2,0,2,6,2, "1234567890" TEXT 400,136, "2",0,1,1,2, "Code 11" BARCODE 400,160, "11",60,2,0,2,6,2, "1234567890" TEXT 400,246, "2",0,1,1,2, "PLANET" BARCODE 400,270, "PLANET",60,2,0,2,2,2, "12345678901" TEXT 400,356, "2",0,1,1,2, "Deutsche Post Identcode." BARCODE 400,380, "DPI",60,2,0,2,6,2, "12345678901" TEXT 400,466, "2",0,1,1,2, "Deutsche Post Leitcode. " BARCODE 400,490, "DPL",60,2,0,2,6,2, "123456789012" TEXT 400,576, "2",0,1,1,2, "Code 49" BARCODE 400,600, "CODE49",60,2,0,2,2,2, "1234567890" PRINT 1	 TELEPEN Number 1234567890 Code 11 1234567890 PLANET 12345678901 Deutsche Post Identcode. 12345678901 6 Deutsche Post Leitcode. 01234567890128 Code 49 1234567890

6.3 TLC39

Description

This command draws TLC39, TCIF Linked Bar Code 3 of 9, barcode.

Syntax

```
TLC39 x,y,rotation,[height,]narrow,]wide,]cellwidth,]cellheight,] "ECI number,Serial number & additional data"
```

Parameter	Description
x	Specify the x-coordinate
y	Specify the y-coordinate
rotation	0 : No rotation 90 : Rotate 90 degrees clockwise 180 : Rotate 180 degrees clockwise 270 : Rotate 270 degrees clockwise
height	Height of Code39 in dots (Default is 40)
narrow	Width of narrow element of Code39 in dots (Default is 2)
wide	Width of wide element of Code39 in dots (Default is 4)
cellwidth	Width of cell of MicroPDF417 in dots (Default is 2)
cellheight	Height of cell of MicroPDF417 in dots (Default is 4)
ECI number	Must be 6 digits which is used to generate Code39
Serial number & additional data	Alphanumeric is for Micro-PDF417

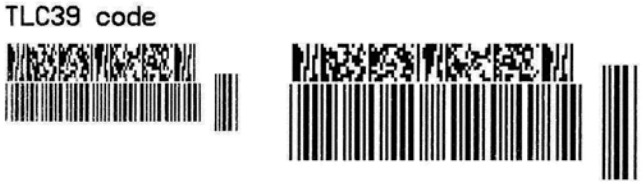
Note:
Comma (") is necessary between ECI number and Serial number & additional data.

Example

Sample Code

```
SIZE 4,1.2
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "TLC39 code"
TLC39 10,50,0, "123456,SN00000001,00601,01501"
TLC39 310,50,0,80,3,6,3,4, "123456,SN00000001,00601,01501"
PRINT 1
```

Result



6.4 BITMAP

Description

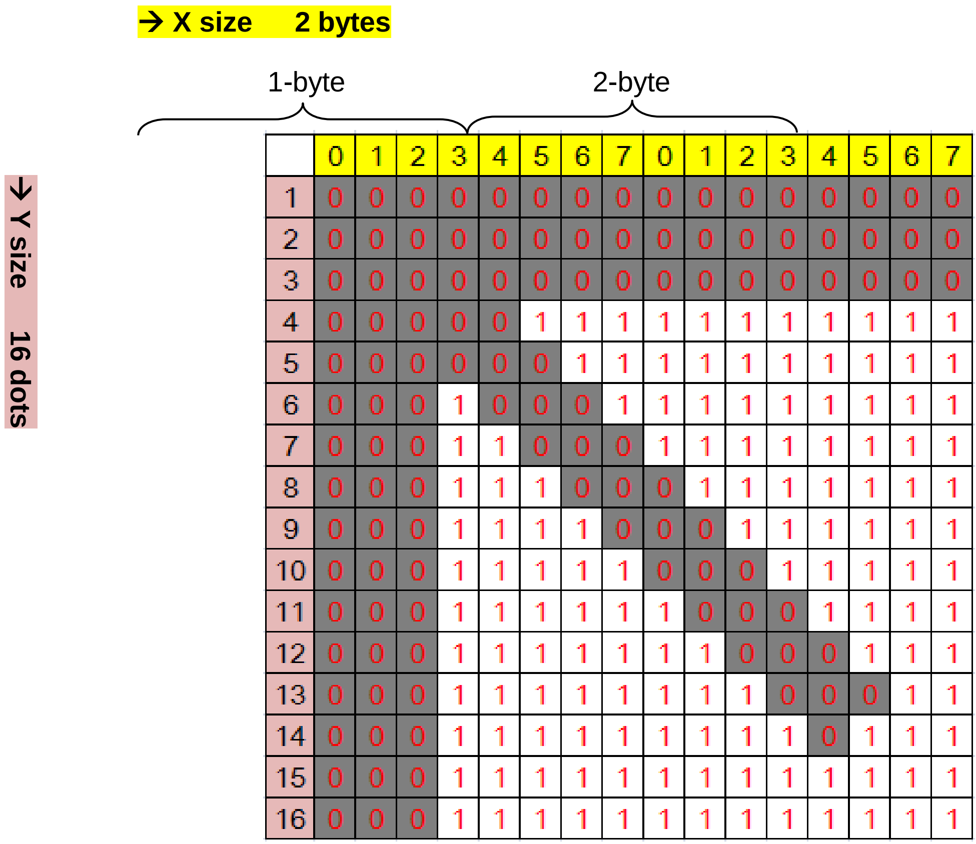
This command draws bitmap images (as opposed to BMP graphic files).

Syntax

BITMAP X,Y,width,height,mode,bitmap data…

Parameter	Description
X	Specify the x-coordinate
Y	Specify the y-coordinate
width	Image width (in bytes)
height	Image height (in dots)
mode	Graphic modes listed below: 0: OVERWRITE 1: OR 2: XOR
bitmap data	Bitmap data

Example



	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
5	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
6	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1
7	0	0	0	1	1	0	0	0	1	1	1	1	1	1	1	1
8	0	0	0	1	1	1	0	0	0	1	1	1	1	1	1	1
9	0	0	0	1	1	1	1	0	0	0	1	1	1	1	1	1
10	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1
11	0	0	0	1	1	1	1	1	1	0	0	0	1	1	1	1
12	0	0	0	1	1	1	1	1	1	1	0	0	0	1	1	1
13	0	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1
14	0	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1
15	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
16	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

	X – axis			
Y- axis	1-byte		2-byte	
	Binary	Hexadecimal	Binary	Hexadecimal
1	00000000	00	00000000	00
2	00000000	00	00000000	00
3	00000000	00	00000000	00
4	00000111	07	11111111	FF
5	00000011	03	11111111	FF
6	00010001	11	11111111	FF
7	00011000	18	11111111	FF
8	00011100	1C	01111111	7F
9	00011110	1E	00111111	3F
10	00011111	1F	00011111	1F
11	00011111	1F	10001111	8F
12	00011111	1F	11000111	C7
13	00011111	1F	11100011	E3
14	00011111	1F	11110111	F7
15	00011111	1F	11111111	FF
16	00011111	1F	11111111	FF

Sample Code (ASCII)	Hexadecimal	Result
SIZE 4,2 GAP 0,0 CLS BITMAP 200,200,2,16,0, □————————————□□□□-?- ????□ PRINT 1,1	53 49 5A 45 20 34 2C 32 0D 0A 47 41 50 20 30 2C 30 0D 0A 43 4C 53 0D 0A 42 49 54 4D 41 50 20 32 30 30 2C 32 30 30 2C 32 2C 31 36 2C 30 2C 00 00 00 00 00 00 07 FF 03 FF 11 FF 18 FF 1C 7F 1E 3F 1F 1F 1F 8F 1F C7 1F E3 1F E7 1F FF 1F FF 0D 0A 50 52 49 4E 54 20 31 2C 31 0D 0A	

See Also

PUTBMP, PUTPCX

6.5 BOX

Description

This command draws rectangles on the label.

Syntax

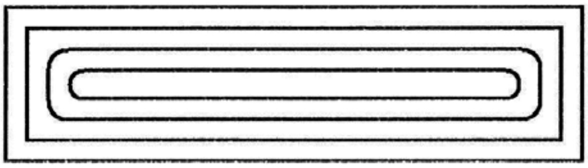
BOX x,y,x_end,y_end,line thickness[,radius]

Parameter	Description
x	Specify x-coordinate of upper left corner (in dots)
y	Specify y-coordinate of upper left corner (in dots)
x_end	Specify x-coordinate of lower right corner (in dots)
y_end	Specify y-coordinate of lower right corner (in dots)
line thickness	Line thickness (in dots)
radius	Optional. Specify the round corner. Default is 0.

Note:

- ***200 DPI : 1 mm = 8 dots***
  300 DPI : 1 mm = 12 dots
- ***Recommended max. thickness of box is 12 mm at 4" width. Thickness of box larger than 12 mm may damage the power supply and affect the print quality. Max. print ratio is different for each printer model. Desktop and industrial printer print ratio is limited to 20% and 30% respectively.***

Example

Sample code	Result
SIZE 4,1.1 CLS BOX 60,60,610,210,4 BOX 80,80,590,190,4 BOX 100,100,570,170,4,20 BOX 120,120,550,150,4,20 PRINT 1	

See Also

BAR

6.6 CIRCLE

Description

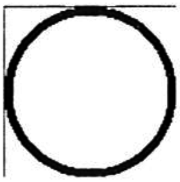
This command draws a circle on the label.

Syntax

CIRCLE X_start,Y_start,diameter,thickness

Parameter	Description
X_start	Specify x-coordinate of upper left corner (in dots)
Y_start	Specify y-coordinate of upper left corner (in dots)
diameter	Specify the diameter of the circle (in dots)
thickness	Thickness of the circle (in dots)

Example

Sample code	Result
SIZE 80 mm,30 mm GAP 0,0 DIRECTION 1 CLS BAR 250,20,100,1 BAR 250,20,1,100 CIRCLE 250,20,100,5 PRINT 1	

6.7 ELLIPSE

Description

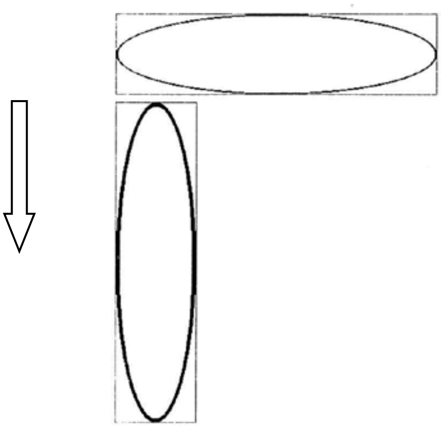
This command draws an ellipse on the label.

Syntax

ELLIPSE x,y,width,height,thickness

Parameter	Description
x	Specify x-coordinate of upper left corner (in dots)
y	Specify y-coordinate of upper left corner (in dots)
width	Specify the width of the ellipse (in dots)
height	Specify the height of the ellipse (in dots)
thickness	Thickness of the ellipse (in dots)

Example

Sample code	Result
SIZE 4,3 **GAP 0,0** **DIRECTION 1** **CLS** **BOX 10,10,410,110,1** **ELLIPSE 10,10,400,100,2** **BOX 10,120,110,520,1** **ELLIPSE 10,120,100,400,5** **PRINT 1**	

6.8 CODABLOCK F mode

Description

This command draws CODABLOCK F mode barcode.

Syntax

CODABLOCK x,y,rotation,[row height,]module width,] "content"

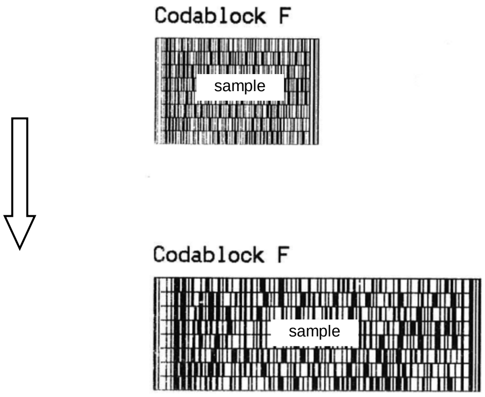
Parameter	Description
x	Specify the x-coordinate
y	Specify the y-coordinate
rotation	0 : No rotation
	90 : Rotate 90 degrees clockwise
	180 : Rotate 180 degrees clockwise
	270 : Rotate 270 degrees clockwise
row height	The height of individual row equals to row height x module width (Default is 8)
module width	Width of narrow element of CODABLOCK in dots (Default is 2)
content	content of CODABLOCK bar code

Example

Sample Code

```
SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "Codablock F"
CODABLOCK 10,50,0, "With the "At your side." spirit in mind. the Brother Group aims to continually create value."
PRINT 1
CLS
TEXT 10,10, "3",0,1,1, "Codablock F"
CODABLOCK 10,50,0,16,1, "With the "At your side." spirit in mind. the Brother Group aims to continually create value."
PRINT 1
```

Result



6.9 DMATRIX

Description

This command defines a DataMatrix 2D bar code. Currently, only ECC200 error correction is supported.

Syntax

DMATRIX x,y,width,height,[c#,x#,r#,a#,row,col,] "content"

Parameter	Description
x	Horizontal start position (in dots)
y	Vertical start position (in dots)
width	The expected width of barcode area (in dots)
height	The expected height of barcode area (in dots)
c#	Escape sequence control character (decimal digit) Ex. C126 means ~

(1) ~X is shift character for control characters.

~X	Hex	ASCII	~X	HEX	ASCII	~X	HEX	ASCII	~X	HEX	ASCII
~@	00	NUL	~H	08	BS	~P	10	DLE	~X	18	CAN
~A	01	SOH	~I	09	HT	~Q	11	DC1	~Y	19	EM
~B	02	STX	~J	0A	LF	~R	12	DC2	~Z	1A	SUB
~C	03	ETX	~K	0B	VT	~S	13	DC3	~[	1B	ESC
~D	04	EOT	~L	0C	FF	~T	14	DC4	~¥	1C	FS
~E	05	ENQ	~M	0D	CR	~U	15	NAK	~]	1D	GS
~F	06	ACK	~N	0E	SO	~V	16	SYN	~^	1E	RS
~G	07	BEL	~O	0F	SI	~W	17	ETB	~_	1F	US

(2) ~1 means FNC1.

(3) ~dNNN creates ASCII decimal value NNN for a codeword. Must be 3 digits. 000 ~ 255.

(4) ~ in data is encoded by ~~.

Parameter	Description
X#	Module size (in dots)
r#	Rotation 0 : No rotation 90 : Rotate 90 degrees clockwise 180 : Rotate 180 degrees clockwise 270 : Rotate 270 degrees clockwise
a#	0 : Square (default) 1 : Rectangle
row	Symbol size of row: 10 to 144
col	Symbol size of col: 10 to 144
content	Content of DataMatrix 2D bar code

Note:
For standard symbol sizes for DataMatrix 2D barcode, please refer to below list.

Square			Rectangle
10 x 10	26 x 26	72 x 72	8 x 18
12 x 12	32 x 32	80 x 80	8 x 32
14 x 14	36 x 36	88 x 88	12 x 26
16 x 16	40 x 40	96 x 96	12 x 36
18 x 18	44 x 44	104 x 104	16 x 36
20 x 20	48 x 48	120 x 120	16 x 48
22 x 22	52 x 52	132 x 132	
24 x 24	64 x 64	144 x 144	

Example

Sample code	Result
SIZE 4,3 GAP 0,0 DIRECTION 1 CLS DMATRIX 10,110,400,400, "DMATRIX EXAMPLE 1" DMATRIX 310,110,400,400,x6, "DMATRIX EXAMPLE 2" DMATRIX 10,310,400,400,x8,18,18, "DMATRIX EXAMPLE 3" PRINT 1,1	
Sample code for FNC SIZE 4,1 GAP 0,0 CLS DIRECTION 1 DMATRIX 100,50,100,100,c126,x6,18,18, "~1241sPn~110sLot~130sQty " PRINT 1	
Sample code in rectangular shape SIZE 4,1 GAP 0,0 DIRECTION 1 CLS DMATRIX 100,110,600,600,a1,"DMATRIX EXAMPLE 1" PRINT 1,1	

6.10 ERASE

Description

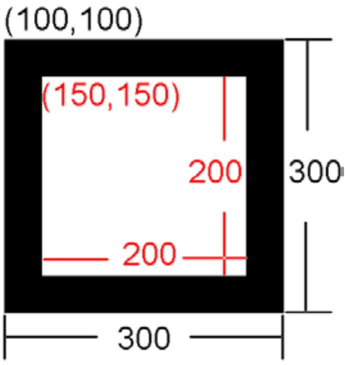
This command clears a specified region in the image buffer.

Syntax

```
ERASE x,y,x_width,y_height
```

Parameter	Description
x	The x-coordinate of the starting point (in dots)
y	The y-coordinate of the starting point (in dots)
x_width	The region width in x-axis direction (in dots)
y_height	The region height in y-axis direction (in dots)

Example

Sample code	Result
SIZE 4,2.5 GAP 0,0 DIRECTION 1 CLS BAR 100,100,300,300 ERASE 150,150,200,200 PRINT 1,1	

See Also

CLS

6.11 MAXICODE

Description

This command defines a 2D Maxicode.

Syntax

MAXICODE x,y,mode,[class,country,post,Lm,] "content"	
MAXICODE x,y,mode,class,country,postal code, "content"	For mode 2 or 3, If country is 840, the postal code is in 99999,9999 format. For other countries, the code is up to 6 alphanumeric characters.
MAXICODE x,y,mode,[Lm,] "content"	For mode 4 or 5, AIM special format is supported.

Parameter	Description
x	X-coordinate of the starting point (in dot)
y	Y-coordinate of the starting point (in dot)
mode	2,3,4,5
class	Class of service, 3-digit number (for mode 2,3)
country	Country code, 3-digit number (for mode 2,3)
post	Post code (for mode 2,3) Mode 2(USA): 5-digit + 4-digit number Mode 3(Canada): 6 alphanumeric post code included by double quotes.
Lm	Expression length (double quote is ignored) , 1≤m≤138, (this parameter is just for mode 4 and 5)
content	Content of 2D Maxicode ***Note:*** ***If parameter Lm is used, double quotes (") are unnecessary.***

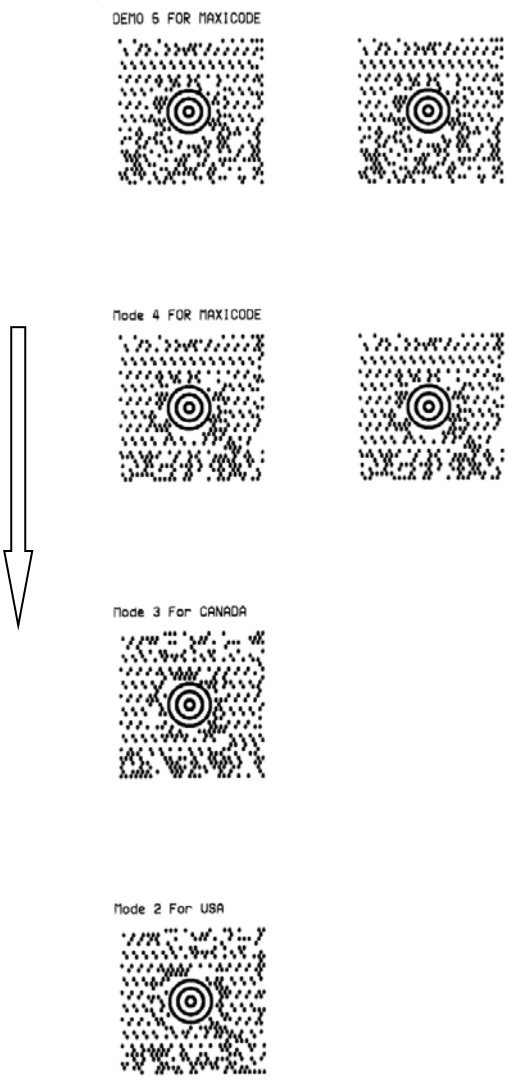
Example

Sample code

```
SIZE 4,2
GAP 0,0
DIRECTION 1
CLS

REM *****Mode 2 For USA*****
MAXICODE 110,100,2,300,840,06810,7317, "DEMO 2 FOR USA MAXICODE"
TEXT 100,50, "3",0,1,1, "Mode 2 For USA"
PRINT 1,1

REM *****Mode 3 For Canada*****
CLS
MAXICODE 110,100,3,300,863, "107317","DEMO 3 FOR CANADA MAXICODE"
TEXT 100,50, "3",0,1,1, "Mode 3 For CANADA"
PRINT 1,1

REM *****MODE4*****
CLS
MAXICODE 110,100,4, "DEMO 4 FOR MAXICODE"
MAXICODE 600,100,4,L19,DEMO 4 FOR MAXICODE
```

```
TEXT 100,50, "3",0,1,1, "Mode 4 FOR MAXICODE"
PRINT 1,1

REM *****MODE 5*****
CLS
MAXICODE 110,100,5, "DEMO 5 FOR MAXICODE"
MAXICODE 600,100,5,L19,DEMO 5 FOR MAXICODE
TEXT 100,50, "3",0,1,1, "DEMO 5 FOR MAXICODE"
PRINT 1
```

Result



DEMO 5 FOR MAXICODE

Mode 4 FOR MAXICODE

Mode 3 For CANADA

Mode 2 For USA

6.12 PDF417

Description

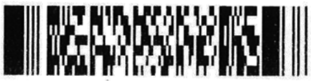
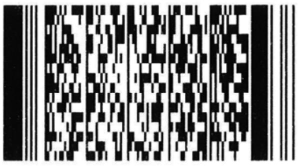
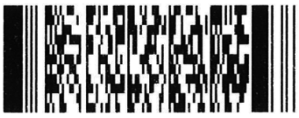
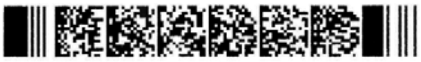
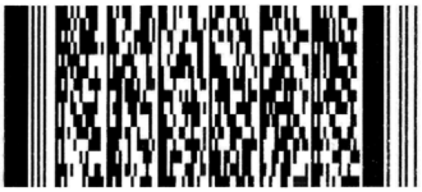
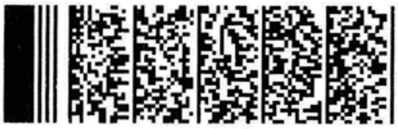
This command defines a PDF417 2D bar code.

Syntax

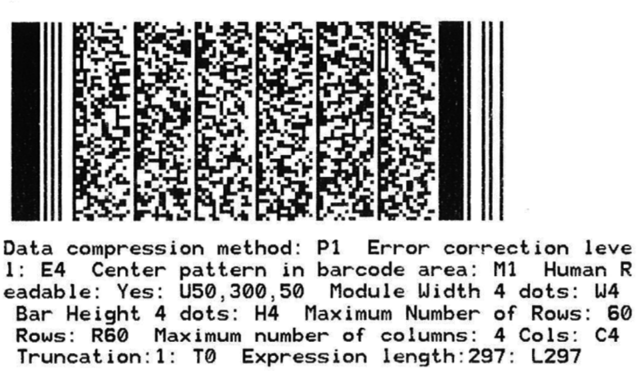
PDF417 x,y,width,height,rotate,[option], "content"

Parameter		Description
x		X-coordinate of starting point (in dot)
y		Y-coordinate of starting point (in dot)
width		Expected width (in dots)
height		Expected height (in dots)
rotate		Rotation counterclockwise 0 : No rotation 90 : Rotate 90 degrees 180 : Rotate 180 degrees 270 : Rotate 270 degrees
option		
	P	Data compression method 0: Auto encoding 1: Binary mode
	E	Error correction level (Range: 0~8)
	M	Center pattern in barcode area 0: The pattern will print upper left justified the area 1: The pattern is printed middle of area
	Ux,y,c	Human readable x: Human readable characters in the specified x-coordinate y: Human readable characters in the specified y-coordinate c: Maximum characters of human readable character per line
	W	Module width in dot (Range: 2~9)
	H	Bar height in dot (Range: 4~99)
	R	Maximum number of rows
	C	Maximum number of columns
	T	Truncation 0: Not truncated 1: Truncated
	Lm	Expression length, 1≤m≤2048 (without " for content)
content		Content of PDF417 2D bar code ***Note:*** ***If parameter Lm is used, double quotes (") are unnecessary for content.***

Example

Sample code	Result
SIZE 4,1 GAP 0,0 DIRECTION 1 REM *****WITHOUR OPTIONS***** CLS PDF417 50,50,400,200,0, "Without Options" PRINT 1,1	
SIZE 4,1.5 GAP 0,0 DIRECTION 1 REM *****OPTION:E3***** CLS PDF417 50,50,400,200,0,E3, "Error correction level:3" PRINT 1,1 REM *****OPTION:E4***** CLS PDF417 50,50,400,200,0,E4, "Error correction level:4" PRINT 1,1	 
SIZE 4,1.5 GAP 0,0 DIRECTION 1 REM *****OPTION:E4 W4***** CLS PDF417 50,50,600,600,0,E4,W4, "Error correction level:4 module width 4 dots" PRINT 1,1 REM *****OPTION:E4 W4 H4***** CLS PDF417 50,50,600,600,0,E4,W4,H4, "Error correction level:4 module width 4 dots bar height 4 dots" PRINT 1,1	 
SIZE 4,1.5 GAP 0,0 DIRECTION 1 REM *****OPTION:E4 W4 H4 R40 C4 T1***** CLS PDF417 50,50,800,800,0,E4,W4,H4,R40,C4,T1, "Error correction level:4 Module Width 4 dots Bar Height 4 dots Maximum Number of Rows:5 Rows Maximum number of columns:90 Cols Truncation:1" PRINT 1,1	

```
SIZE 4,2.5
GAP 0,0
DIRECTION 1

REM *****OPTION:P1 E4 M1
U50,300,50,W4,H4,R60,C4,T0,L297*****
CLS
PDF417
50,50,900,600,0,P1,E4,M1,U50,300,50,W4,H4,R60
,C4,T0,L297,Data compression method: P1
Error correction level: E4
Center pattern in barcode area: M1
Human Readable: Yes: U50,300,50
Module Width 4 dots: W4
Bar Height 4 dots: H4
Maximum Number of Rows: 60 Rows: R60
Maximum number of columns: 4 Cols: C4
Truncation:1: T0
Expression length:297: L297
PRINT 1,1
```



```
Data compression method: P1  Error correction leve
l: E4  Center pattern in barcode area: M1  Human R
eadable: Yes: U50,300,50  Module Width 4 dots: W4
 Bar Height 4 dots: H4  Maximum Number of Rows: 60
 Rows: R60  Maximum number of columns: 4 Cols: C4
 Truncation:1: T0  Expression length:297: L297
```

6.13 AZTEC

Description

This command defines a AZTEC 2D bar code.

Syntax

AZTEC x,y,rotate,[size,]ecp,]flg,]menu,]multi,]rev,] "content"

AZTEC x,y,rotate,size,ecp,flg,menu,multi,rev,bytes,content

Parameter	**Description**
x	Horizontal start position (in dots)
y	Vertical start position (in dots)
rotate	Rotation 0 : No rotation 90 : Rotate 90 degrees 180 : Rotate 180 degrees 270 : Rotate 270 degrees
size	Element module size (1 to 20), default is 6
ecp	Error control (& symbol size/type) parameter 0 : default error correction level 1 to 99 : minimum error correction percentage 101 to 104 : 1 to 4-layer Compact symbol 201 to 232 : 1 to 32-layer Full-Range symbol 300 : a simple Aztec “Rune”
flg	0 : input message is straight bytes 1 : input uses "<Esc>n" for FLG(n), "<Esc><Esc>" for "<Esc>"
menu	Menu symbol (0 : no, 1 : yes), default is 0
multi	Number of symbols (1 to 26), default is 6
rev	Output to be reversed (0 : no, 1 : yes), default is 0
bytes	Length of content
content	Content of AZTEC 2D bar code ***Note:*** ***If parameter bytes is used, double quotes (") are unnecessary.***

Example

Sample Code	**Result**
SIZE 4,2 **GAP 0,0** **CLS** **AZTEC 10,10,0,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"** **AZTEC 210,10,0,4,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"** **AZTEC 410,10,0,4,1,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"** **AZTEC 610,10,0,4,1,0,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"** **AZTEC 10,310,0,4,1,0,0,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"** **AZTEC 210,310,0,4,1,0,0,1,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"** **AZTEC 410,310,0,4,1,0,0,1,1,"ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"** **AZTEC 610,310,0,4,1,0,0,1,1,10,1234567890** **PRINT 1**	

6.14 MPDF417

Description

This command defines a Micro PDF 417 bar code.

Syntax

MPDF417 x,y,rotate,[Wn,][Hn,][Cn,] "content"

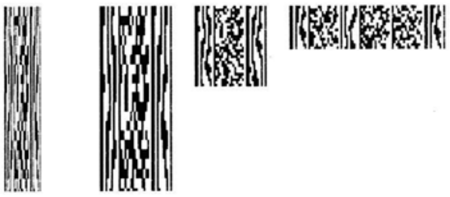
Parameter	Description
x	Horizontal start position (in dots)
y	Vertical start position (in dots)
rotate	Rotation 0 : No rotation 90 : Rotate 90 degrees 180 : Rotate 180 degrees 270 : Rotate 270 degrees
Wn	Optional. Module width in dot. Default is 1.
Hn	Optional. Module height in dot. Default is 10.
Cn	Optional. Number of columns. Once the parameter is set, the printer will calculate the proper rows for the barcode base on the content automatically. 0: Auto mode. 1: Column is 1 and the calculated suitable rows will be 11, 14, 17, 20, 24, and 28. 2: Column is 2 and the calculated suitable rows will be 8, 11, 14, 17, 20, 23 and 26. 3: Column is 3 and the calculated suitable rows will be 6, 8, 10, 12, 15, 20, 26, 32, 38 and 44. 4: Column is 4 and the calculated suitable rows will be 4, 6, 8, 10, 12, 15, 20, 26, 32, 38 and 44.
Content	Content of Micro PDF 417 bar code

Example

Sample Code

```
SIZE 4,1
GAP 0,0
CLS
MPDF417 10,10,0, "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 "
MPDF417 110,10,0,W2, "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 "
MPDF417 210,10,0,W2,H3, "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 "
MPDF417 310,10,0,W2,H3,C3, "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 "
PRINT 1
```

Result



6.15 PUTBMP

Description

This command prints BMP format images. The grayscale printing is for direct thermal mode only. Support 1-bit (monochrome) and 8-bit (256-color) BMP graphic only.

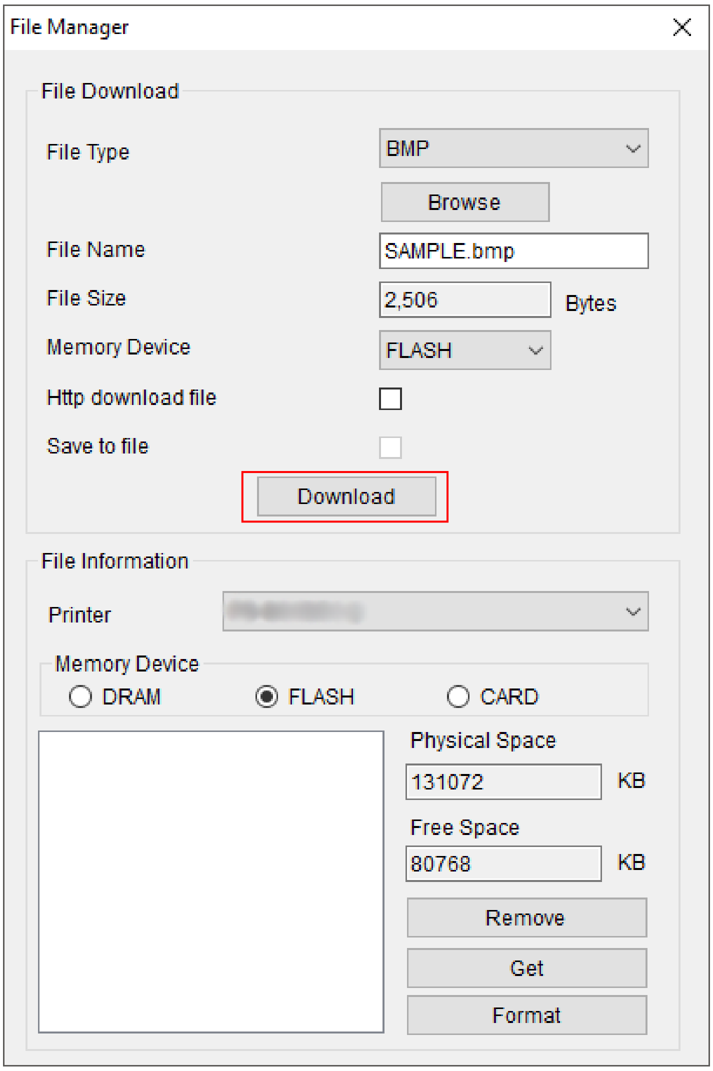
Syntax

PUTBMP x,y, "filename" [, bpp][, contract]

Parameter	Description
x	The x-coordinate of the BMP format image
y	The y-coordinate of the BMP format image
filename	The downloaded BMP filename (Support ZPL *.GRF)
bpp	Optional. Bits per pixel of grayscale graphic. Default is 1. 1: 1-bit (monochrome) graphic 8: 8-bit (256-color) graphic
contrast	Optional. Contrast of grayscale graphic. Default is 80. Suggested range is from 60 to 100.

Example

Downloading BMP file into printer by DiagTool



File Manager

File Download

File Type: BMP

File Name: SAMPLE.bmp

File Size: 2,506 Bytes

Memory Device: FLASH

Http download file ☐

Save to file ☐

File Information

Printer: [redacted]

Memory Device: ○ DRAM ● FLASH ○ CARD

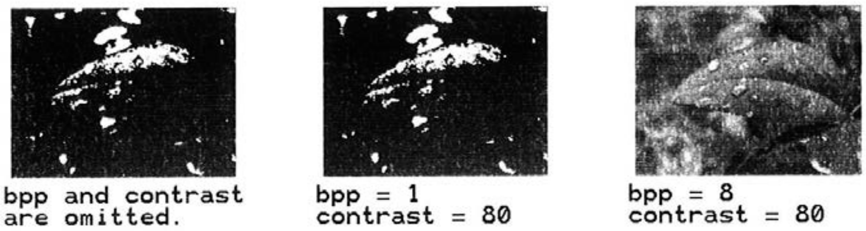
Physical Space: 131072 KB

Free Space: 80768 KB

Sample Code

```
SPEED 2
DENSITY 3
SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS
PUTBMP 10,10,"SAMPLE.BMP"
BLOCK 10,180,240,100,"2",0,1,1,"bpp and contrast are omitted."
PUTBMP 300,10, "SAMPLE.BMP",1,80
BLOCK 300,180,240,100,"2",0,1,1, "bpp = 1
contrast = 80"
PUTBMP 590,10, "SAMPLE.BMP",8,80
BLOCK 590,180,240,100,"2",0,1,1,"bpp = 8
contrast = 80"
PRINT 1
```

Result



```
bpp and contrast     bpp = 1              bpp = 8
are omitted.         contrast = 80        contrast = 80
```

Sample Code

```
SIZE 2,2
GAP 0,0
CLS
PUTBMP 10,10, "SAMPLE.GRF"
PRINT 1
```

See Also

DOWNLOAD, BITMAP, PUTPCX

6.16 PUTPCX

Description

This command prints PCX format images. FBPL language supports 256-color PCX format graphics.

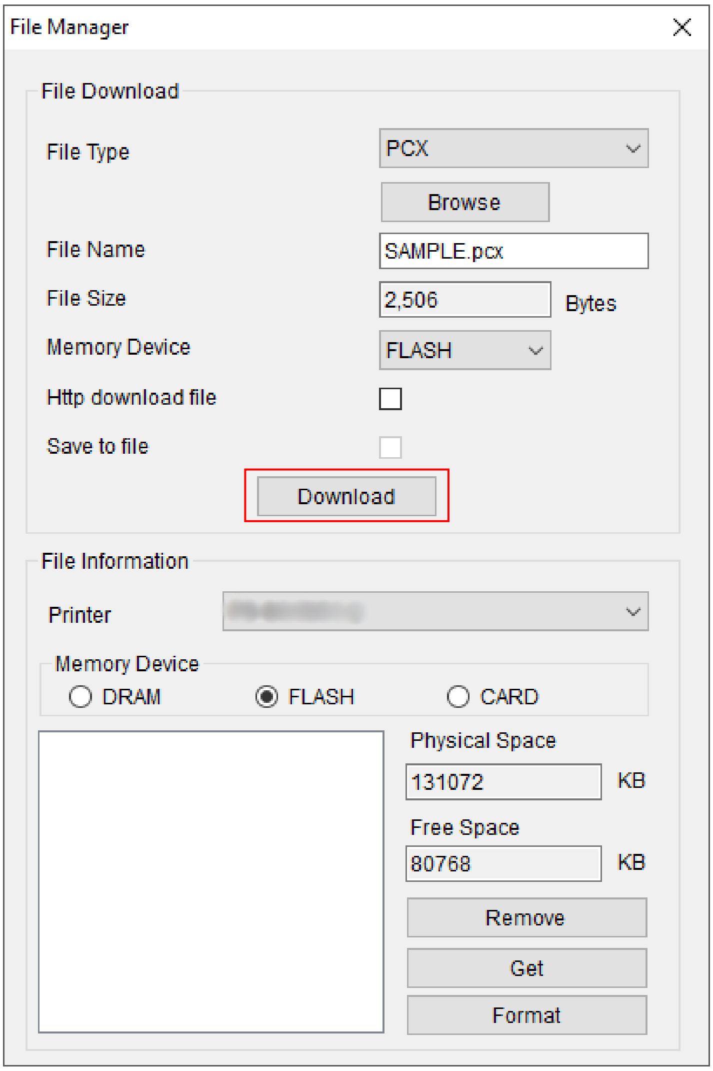
Syntax

PUTPCX x,y, "filename"

Parameter	Description
x	The X-coordinate of the PCX format image
y	The Y-coordinate of the PCX format image
filename	The downloaded PCX file name ***(Case sensitive)***

Example

Downloading PCX file into printer by DiagTool



File Manager

File Download

File Type: PCX

File Name: SAMPLE.pcx

File Size: 2,506 Bytes

Memory Device: FLASH

Http download file □

Save to file □

File Information

Printer: [illegible]

Memory Device: ○ DRAM ● FLASH ○ CARD

Physical Space: 131072 KB

Free Space: 80768 KB

Sample Code
SPEED 2 DENSITY 3 SIZE 4,1.5 GAP 0,0 DIRECTION 1 CLS PUTBMP 10,10, "SAMPLE.PCX" PRINT 1
Result

See Also

DOWNLOAD, BITMAP, PUTPCX

6.17 QRCODE

Description

This command prints QR code.

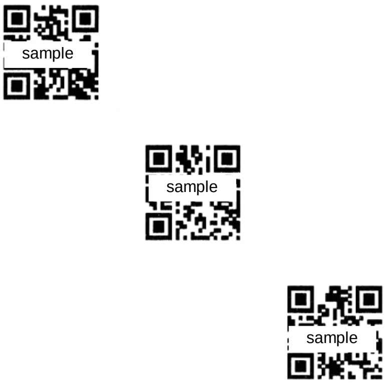
Syntax

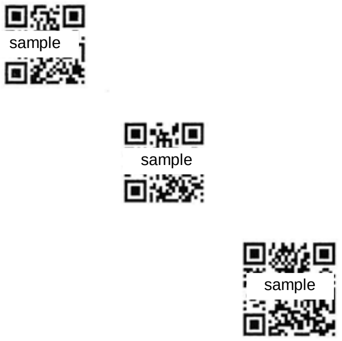
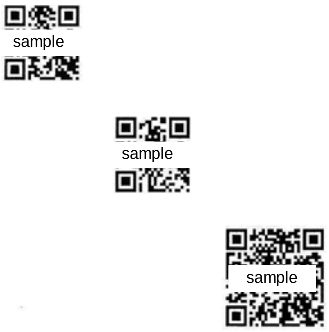
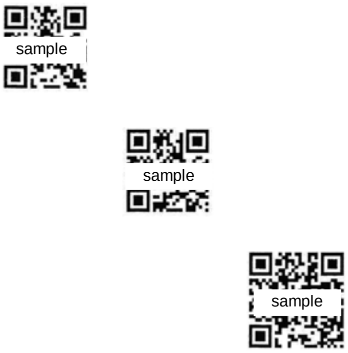
QRCODE x,y,ECC Level,cell width,mode,rotation,[justification,]model,]mask,]area] "content"

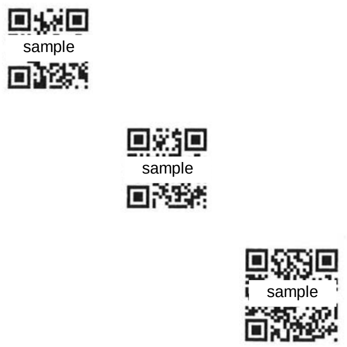
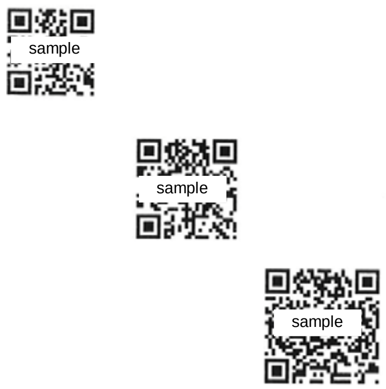
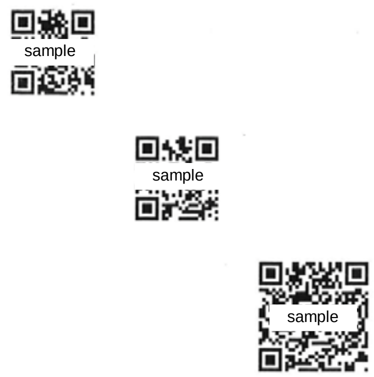
<table>
<tr><th><u>Parameter</u></th><th><u>Description</u></th></tr>
<tr><td>x</td><td>The upper left corner x-coordinate of the QR code</td></tr>
<tr><td>y</td><td>The upper left corner y-coordinate of the QR code</td></tr>
<tr><td>ECC level</td><td>Error correction recovery level
L : 7%
M : 15%
Q : 25%
H : 30%</td></tr>
<tr><td>cell width</td><td>1~10</td></tr>
<tr><td>mode</td><td>Auto / manual encode
A : Auto
M : Manual</td></tr>
<tr><td>rotation</td><td>0 : 0 degree
90 : 90 degree
180 : 180 degree
270 : 270 degree</td></tr>
<tr><td>[justification]</td><td>Barcode justification (J1 to J9 valid; refer to "Sample code" example below)</td></tr>
<tr><td>[model]</td><td>M1: (default), original version
M2: enhanced version (Almost smart phone is supported by this version.)</td></tr>
<tr><td>[mask]</td><td>S0~S8, default is S7</td></tr>
<tr><td>[area]</td><td>Maximum size of barcode area (Xdots; ex: X100)</td></tr>
<tr><td>content</td><td>The encodable character set is described as below,
Encodable character set:
1) Numeric data: (digits 0~9)
2) Alphanumeric data
Digits 0-9
Upper case letters A-Z
Nine other characters: space, $ % * + - . / :)
3) 8-bit byte data
JIS 8-bit character set (Latin and Kana) in accordance with JIS X 0201
4) Kanji characters
Shift JIS values 8140_{HEX} –$9FFC_{HEX}$ and $E040_{HEX}$ –$EAA4_{HEX}$. These are values shifted from those of JIS X 0208. Refer to JIS X 0208 Annex 1 Shift Coded Representation for detail.
Data characters per symbol (for maximum symbol size):
<table>
<tr><th></th><th>Model 1 (Version 14-L)</th><th>Model 2 (Version 40-L)</th></tr>
<tr><td>Numeric data</td><td>1,167 characters</td><td>7,089 characters</td></tr>
<tr><td>Alphanumeric data</td><td>707 characters</td><td>4,296 characters</td></tr>
<tr><td>8-bit byte data</td><td>486 characters</td><td>2,953 characters</td></tr>
<tr><td>Kanji data</td><td>299 characters</td><td>1,817 characters</td></tr>
</table>
*If "A" is the first character in the data string, then the following data after"A" is alphanumeric data.
*If "N" is the first character in the data string, then the following data after "N" is numeric data.
*If "B" is the first character in the data string, then the following 4 digits after "B" is used to specify numbers of data. After the 4 digits is the number of bytes of binary data to be encoded.
*If "K" is the first character in the data string, then the following data after "K" is Kanji data.</td></tr>
</table>

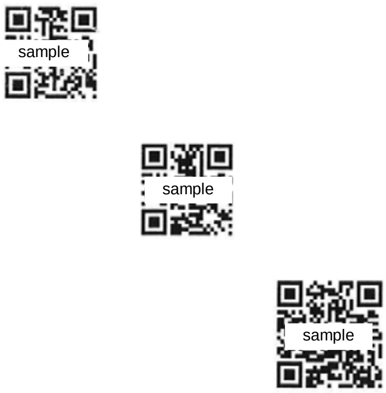
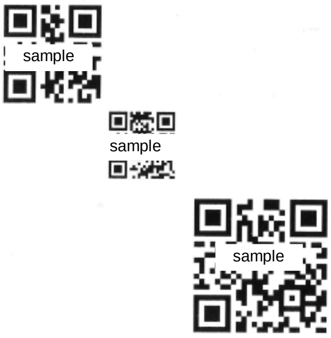
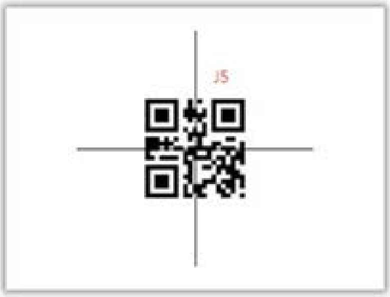
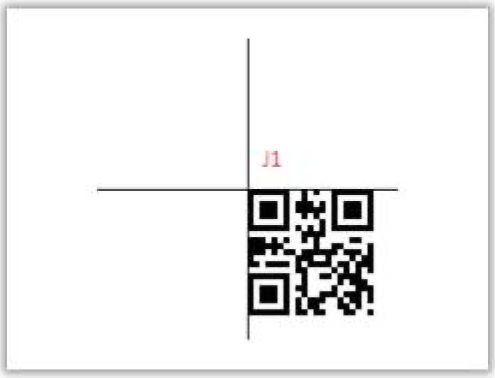
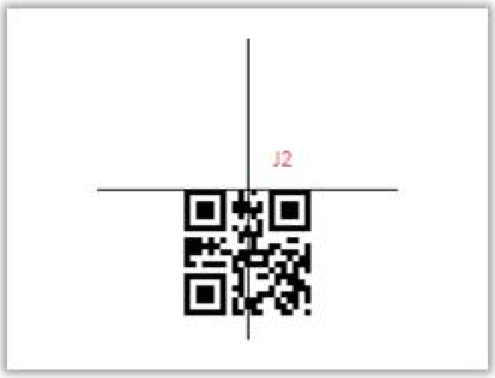
	*If "!" is in the data string and follows by "N", "A", "B", "K" then it will be switched to specified encodable character set. Manual mode example: **QRCODE 100,10,L,7,M,0,M1,S1, "ATHE FIRMWARE HAS BEEN UPDATED"** (Where A: Alphanumeric data) **QRCODE 100,10,M,7,M,0,M1,S2, "N123456"** (Where N: Numeric data) **QRCODE 100,10,Q,7,M,0,M1,S3, "N123456!ATHE FIRMWARE HAS BEEN UPDATED"** (Where N: Numeric data ; !:Transfer char ; A: Alphanumeric data) **QRCODE 100,10,H,7,M,0,M1,S3, "B0012Product name"** (where B: Binary data ; 0012: 12 bytes) **QRCODE 100,10,M,7,M,0,M1,S3, "K"** (Where K: Kanji data) Auto mode example: **QRCODE 100,10,M,7,A,0, "THE FIRMWARE HAS BEEN UPDATED"**

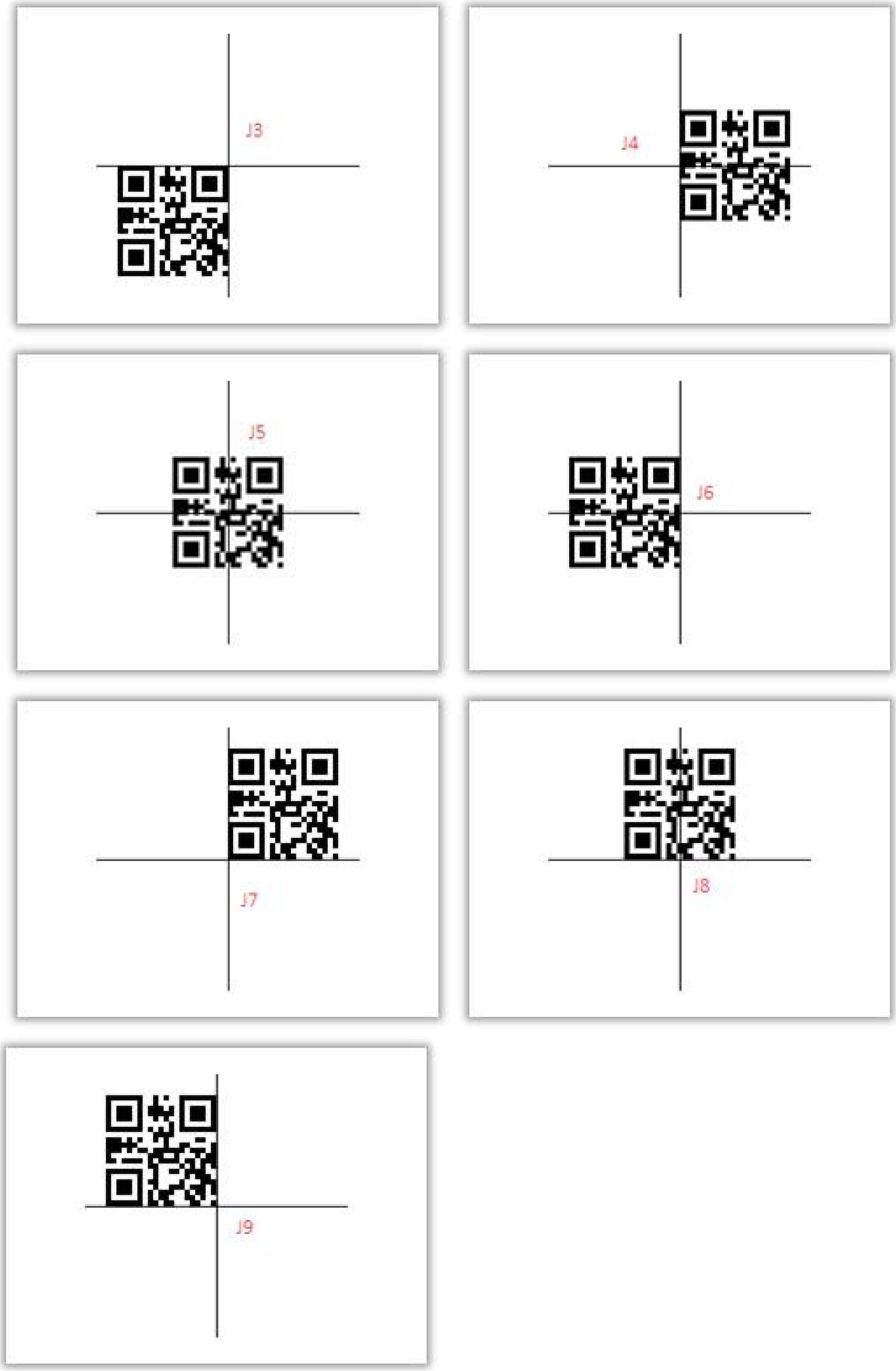
Example

Sample code	Result
Auto mode example	
General data string **SIZE 4,2.5** **GAP 0,0** **DIRECTION 1** **CLS** **QRCODE 10,10,H,4,A,0, "ABCabc123"** **QRCODE 160,160,H,4,A,0, "123ABCabc"** **QRCODE 310,310,M,4,A,0,M2, "Printer ABCabc123"** **PRINT 1,1**	

Data string including <Enter> character (0Dh, 0Ah) **SIZE 4,2.5** **GAP 0,0** **DIRECTION 1** **CLS** **QRCODE 10,10,H,4,A,0, "ABC<Enter>** **abc<Enter>** **123 "** **QRCODE 160,160,H,4,A,0, "123<Enter>** **ABC<Enter>** **abc"** **QRCODE 310,310,H,4,A,0, "Printer<Enter>** **ABC<Enter>** **abc<Enter>** **123"** **PRINT 1,1**	
Data string concatenation (Must be used with DOWNLOAD ... EOP command) **DOWNLOAD "DEMO.BAS"** **SIZE 4,2.5** **GAP 0,0** **DIRECTION 1** **CLS** **QRCODE 10,10,H,4,A,0, "ABCabc123" +STR$(1234)** **QRCODE 160,160,H,4,A,0, "123ABCabc" +"1234"** **QRCODE 310,310,H,4,A,0,** **"PrinterABCabc123"+"1234"+"abcd"** **PRINT 1,1** **EOP** **DEMO**	
Data string including double quote (") character, please use ¥["] instead of **SIZE 4,2.5** **GAP 0,0** **DIRECTION 1** **CLS** **QRCODE 10,10,H,4,A,0, "ABC¥["]abc¥["]123"** **QRCODE 160,160,H,4,A,0, "123¥["]ABC¥["]abc"** **QRCODE 310,310,H,4,A,0,** **"¥["]Printer¥["]ABCabc123"** **PRINT 1,1**	

Manual mode	
General data string SIZE 4,2.5 GAP 0,0 DIRECTION 1 CLS QRCODE 10,10,H,4,M,0, "AABC!B0003abc!N123" QRCODE 160,160,H,4,M,0, "N123!AABC!B0003abc" QRCODE 310,310,H,4,M,0, "K Printer!AABC!B0006abc123" PRINT 1,1	
Data string including <Enter> character, <Enter> is an 8-bit byte data SIZE 4,2.5 GAP 0,0 DIRECTION 1 CLS QRCODE 10,10,H,4,M,0,"AABC!B0007<Enter> abc<Enter> !N123" QRCODE 160,160,H,4,M,0,"N123!B0002<Enter> !AABC!B0005<Enter> abc" QRCODE 310,310,H,4,M,0, "K Printer!B0002<Enter> !AABC!B0010<Enter> abc<Enter> 123" PRINT 1,1	
Data string concatenation (Must be used with DOWNLOAD ... EOP command) DOWNLOAD "A.BAS" SIZE 4,2.5 GAP 0,0 DIRECTION 1 CLS QRCODE 10,10,H,4,M,0,"AABC!B0006abc123!N"+STR$(1234) QRCODE 160,160,H,4,M,0,"N123!AABC!B0007abc"+"1234" QRCODE 310,310,H,4,M,0, "K Printer!AABC!B0014abc123"+" 1234"+"abcd" PRINT 1,1 EOP A	

Data string including double quote (") character, please use ¥["] instead of **SIZE 4,2.5** **GAP 0,0** **DIRECTION 1** **CLS** **QRCODE 10,10,H,4,M,0, "AABC!B0005¥["]abc¥["]!N123"** **QRCODE 160,160,H,4,M,0, "N123!B0001¥["]!AABC!B0004¥["]abc"** **QRCODE 310,310,H,4,M,0, "B0001¥["]!K Printer!B0010¥["]ABCabc123"** **PRINT 1,1**	
Smart phone data string **DOWNLOAD "A.BAS"** **SIZE 3,3** **GAP 0,0** **DIRECTION 1** **CLS** **QRCODE 10,10,H,7,M,0,M2,S7,"Aabcd"** **QRCODE 170,170,H,4,M,0, M2,"B0008 日本語"** **QRCODE 300,300, L, 8, M, 0, M2,"B0026http://www.brother.com"** **PRINT 1,1** **EOP** **A**	
Data string for parameter [justification] & [area] **SIZE 4,2.5** **GAP 0,0** **DIRECTION 1** **CLS** **BAR 60,120,200,1** **BAR 160,20,1,200** **QRCODE 160,120,H,10,A,0,X100,J5,"123456789"** **PRINT 1,1**	
For other [justification] results (J1~J9)  	


J3
J4
J5
J6
J7
J8
J9

6.18 RSS

Description

This command is used to draw a RSS bar code on the label format.

Syntax

RSS x,y, "sym",rotate,pixMult,sepHt, "content"
RSS x,y, "RSSEXP",rotate,pixMult,sepHt,segWidth, "content"
RSS x,y, "UCC128CCA",rotate,pixMult,sepHt,linHeight, "content"
RSS x,y, "UCC128CCC",rotate,pixMult,sepHt,linHeight, "content"

Parameter	Description
x	X-coordinate
y	Y-coordinate
sym	Symbology type:

RSS14	RSS14
RSS14T	RSS14 Truncated
RSS14S	RSS14 Stacked
RSS14SO	RSS14 Stacked Omnidirectional
RSSLIM	RSS Limited
RSSEXP	RSS Expanded
UPCA	UPC-A
UPCE	UPC-E
EAN13	EAN-13
EAN8	EAN-8
UCC128CCA	UCC/EAN-128 & CC-A/B
UCC128CCC	UCC/EAN-128 & CC-C

rotate	Rotation (0, 90, 180, and 270 valid)
pixMult	Module width in dot (1 to 10 valid) The following barcode height is calculated by printer.

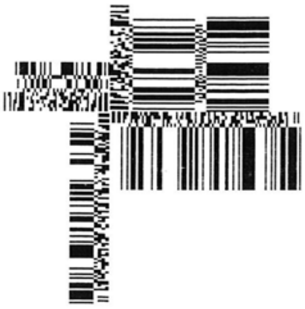
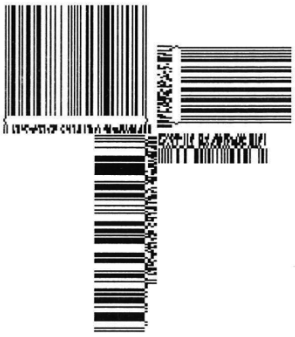
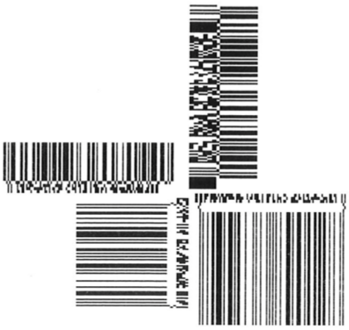
RSS14	33 × pixMult
RSS14T	13 × pixMult.
RSS14S	13 × pixMult.
RSS14SO	33 × pixMult.
RSSLIM	13 × pixMult.
RSSEXP	33 × pixMult.
EAN8	60 × pixMult.
EAN13	74 × pixMult.
UPCA	74 × pixMult.
UPCE	74 × pixMult.

sepHt	Separator row height (1 and 2 valid) pixMult times sepHt is the real separator row height. It is calculated by printer.
segWidth	Segment width of RSS expanded (even 2 to 22 valid)
linHeight	UCC/EAN-128 height in dot (1 to 500 valid)
content	Barcode content or string expression Content of UPCE must be: *00abc0000hij = abhijc, where c = 0-2 *00abc00000ij = abcij3 *00abcd00000j = abcdj4 *00abcde0000j = abcdej where j = 5-9

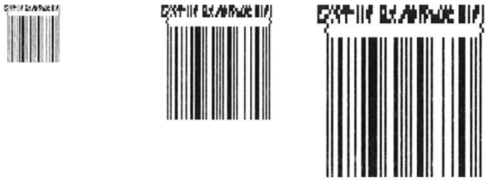
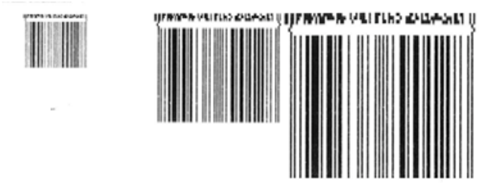
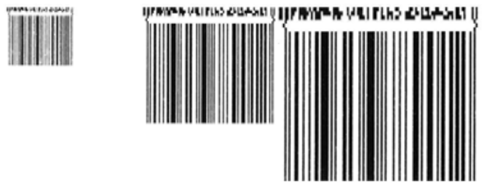
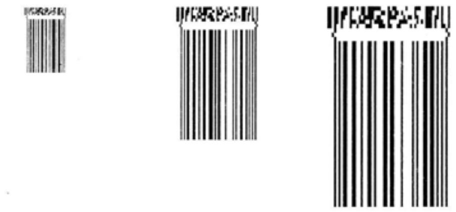
Note:

- ***200 DPI: 1 mm = 8 dots***
  300 DPI: 1 mm = 12 dots
- ***Recommended max. height of reversed black area is 12 mm at 4 " width. Height of reversed area that is larger than 12 mm may damage the power supply and affect the print quality.***
- ***Max. print ratio is different for each printer model. Desktop and industrial printer print ratio is limited to 20% and 30% respectively.***

Example

Sample code	Result
SIZE 100 mm,100 mm GAP 0,0 DIRECTION 1 CLS RSS 300,300, "RSS14",0,2,2, "1234567890|ABCDEFG" RSS 300,300,"RSS14T",90,2,2,"1234567890|ABCDEFG" RSS 300,300,"RSS14S",180,2,2,"1234567890|ABCDEFG" RSS 300,300, "RSS14SO",270,2,2, "1234567890|ABCDEFG" PRINT 1,1	
SIZE 100 mm,100 mm GAP 0,0 DIRECTION 1 CLS RSS 300,300, "RSSLIM",0,2,2, "1234567890|ABCDEFG" RSS 300,300, "RSSEXP",90,2,2,22, "1234567890|ABCDEFG" RSS 300,300, "UPCA",180,2,2, "1234567890|ABCDEFG" RSS 300,300, "UPCE",270,2,2, "000|ABCDEFG" PRINT 1,1	
SIZE 100 mm,100 mm GAP 0,0 DIRECTION 1 CLS RSS 300,300,"EAN13",0,2,2,"123456789012|ABCDEFG" RSS 300,300,"EAN8",90,2,2,"1234567|ABCDEFG" RSS 300,300,"UCC128CCA",180,2,2,25,"1234567890|ABCDEFG" RSS 300,300,"UCC128CCC",270,2,2,25,"1234567890|ABCDEFG" PRINT 1,1	

SIZE 100 mm, 100 mm GAP 0,0 DIRECTION 1 CLS RSS 300,10, "RSSEXP",90,2,2,12, "8110106141411234562891101201212085010048000214025610048000310123191000" PRINT 1	
Example of UPCE mode	
SIZE 4,1 GAP 0,0 DIRECTION 1 CLS REM UPCE Rule 1: *00abc0000hij* = abhijc, where c = 0-2 RSS 10,10,"UPCE",0,2,2,"001200000456|ABCDEFG" RSS 210,10,"UPCE",0,2,2,"001210000456|ABCDEFG" RSS 410,10,"UPCE",0,2,2,"001220000456|ABCDEFG" PRINT 1	
SIZE 4,1 GAP 0,0 DIRECTION 1 CLS REM UPCE Rule 2: *00abc00000ij* = abcij3 RSS 10,10,"UPCE",0,2,2,"001230000045|ABCDEFG" PRINT 1 SIZE 4,1 CLS REM UPCE Rule 3: *00abcd00000j* = abcdj4 RSS 10,10,"UPCE",0,2,2,"001234000005|ABCDEFG" PRINT 1	
SIZE 4,1 GAP 0,0 DIRECTION 1 CLS REM UPCE Rule 4: *00abcde0000j* = abcdej where j = 5-9 RSS 10,10,"UPCE",0,2,2,"001234500005|ABCDEFG" RSS 160,10,"UPCE",0,2,2,"001234500006|ABCDEFG" RSS 310,10,"UPCE",0,2,2,"001234500007|ABCDEFG" RSS 460,10,"UPCE",0,2,2,"001234500008|ABCDEFG" RSS 610,10,"UPCE",0,2,2,"001234500009|ABCDEFG" PRINT 1	

Example of barcode height of EAN8 EAN13 UPCA and UPCE.	
SIZE 4,2 GAP 0,0 DIRECTION 1 CLS RSS 10,10,"EAN8",0,1,1,"1234567|ABCDEFG" RSS 210,10, "EAN8",0,2,1,"1234567|ABCDEFG" RSS 410,10, "EAN8",0,3,1,"1234567|ABCDEFG" PRINT 1	
SIZE 4,2 GAP 0,0 DIRECTION 1 CLS RSS 10,10,"EAN13",0,1,1,"123456789012|ABCDEFG" RSS 210,10,"EAN13",0,2,1,"123456789012|ABCDEFG" RSS 410,10,"EAN13",0,3,1,"123456789012|ABCDEFG" PRINT 1	
SIZE 4,2 GAP 0,0 DIRECTION 1 CLS RSS 10,10,"UPCA",0,1,1,"12345678901|ABCDEFG" RSS 210,10,"UPCA",0,2,1,"12345678901|ABCDEFG" RSS 410,10,"UPCA",0,3,1,"12345678901|ABCDEFG" PRINT 1	
SIZE 4,2 GAP 0,0 DIRECTION 1 CLS RSS 10,10,"UPCE",0,1,1,"001200000456|ABCDEFG" RSS 210,10,"UPCE",0,2,1,"001210000456|ABCDEFG" RSS 410,10,"UPCE",0,3,1,"001220000456|ABCDEFG" PRINT 1	

6.19 REVERSE

Description

This command reverses a region in image buffer.

Syntax

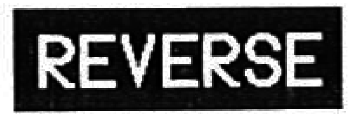
REVERSE x_start,y_start,x_width,y_height

Parameter	Description
x_start	The x-coordinate of the starting point (in dots)
y_start	The y-coordinate of the starting point (in dots)
x_width	X-axis region width (in dots)
y_height	Y-axis region height (in dots)

Note:

- ***200 DPI : 1 mm = 8 dots***
  300 DPI : 1 mm = 12 dots
- ***Recommended max. height of reversed black area is 12mm at 4" width. Height of reversed area that is larger than 12 mm may damage the power supply and affect the print quality.***
- ***Max. print ratio is different for each printer model. Desktop and industrial printer print ratio is limited to 20% and 30% respectively.***

Example

Sample code	Result
SIZE 4,2.5 **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 100,100,"3",0,1,1,"REVERSE"** **REVERSE 90,90,128,40** **PRINT 1,1**	

6.20 DIAGONAL

Description

This command is used to draw a diagonal.

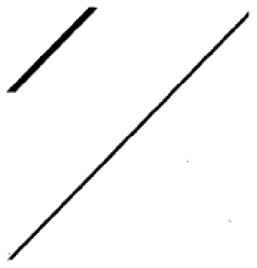
Syntax

DIAGONAL x1, y1, x2, y2, thickness

Parameter	Description
x1	The x1-coordinate of the starting point (in dots)
y1	The y1-coordinate of the starting point (in dots)
x2	The x2-coordinate of the ending point (in dots)
y2	The y2-coordinate of the ending point (in dots)
thickness	Thickness of diagonal

Note:
200 DPI : 1 mm = 8 dots
300 DPI : 1 mm = 12 dots

Example

Sample code	Result
SIZE 4,2.5 **GAP 0,0** **DIRECTION 1** **CLS** **DIAGONAL 50, 200, 200, 50, 16** **DIAGONAL 50, 500, 500, 50, 8** **PRINT 1,1**	

6.21 TEXT

Description

This command prints text on label.

Syntax

TEXT x,y, " font ",rotation,x-multiplication,y-multiplication,[alignment,] " content "

Parameter	Description
x	The x-coordinate of the text
y	The y-coordinate of the text
font	Font name

0	Monotye CG Triumvirate Bold Condensed, font width and height is stretchable
1	8 x 12 fixed pitch dot font
2	12 x 20 fixed pitch dot font
3	16 x 24 fixed pitch dot font
4	24 x 32 fixed pitch dot font
5	32 x 48 dot fixed pitch font
6	14 x 19 dot fixed pitch font OCR-B
7	21 x 27 dot fixed pitch font OCR-B
8	14 x25 dot fixed pitch font OCR-A
ROMAN.TTF	Monotye CG Triumvirate Bold Condensed, font width and height proportion is fixed.
1.EFT	EPL2 font 1
2.EFT	EPL2 font 2
3.EFT	EPL2 font 3
4.EFT	EPL2 font 4
5.EFT	EPL2 font 5
A.FNT	ZPL2 font A
B.FNT	ZPL2 font B
D.FNT	ZPL2 font D
E8.FNT	ZPL2 font E8
F.FNT	ZPL2 font F
G.FNT	ZPL2 font G
H8.FNT	ZPL2 font H8
GS.FNT	ZPL2 font GS

Parameter	Description
rotation	The rotation angle of text 0 : No rotation 90 : degrees, in clockwise direction 180 : degrees, in clockwise direction 270 : degrees, in clockwise direction
x-multiplication	Horizontal multiplication, up to 10x Available factors: 1~10 For "ROMAN.TTF" true type font, this parameter is ignored. For font "0", this parameter is used to specify the width (point) of true type font. 1 point=1/72 inch.
y-multiplication	Vertical multiplication, up to 10x Available factors: 1~10 For true type font, this parameter is used to specify the height (point) of true type font. 1 point=1/72 inch. For *.TTF font, x-multiplication and y-multiplication support floating value.

alignment Optional. Specify the alignment of text.
0 : Default (Left)
1 : Left
2 : Center
3 : Right

content Content of text string

Note:

- ***Font "0" and "ROMAN.TTF" internal True Type Fonts are available in FBPL language printers.***
- ***If there is any double quote (") within the text, please change it to ¥["].***
- ***If font "0" is used, the font width and font height is stretchable by x-multiplication and y-multiplication parameter. It is expressed by pt (point). 1 point=1/72inch.***
- ***EPL2 and ZPL2 fonts are also supported.***

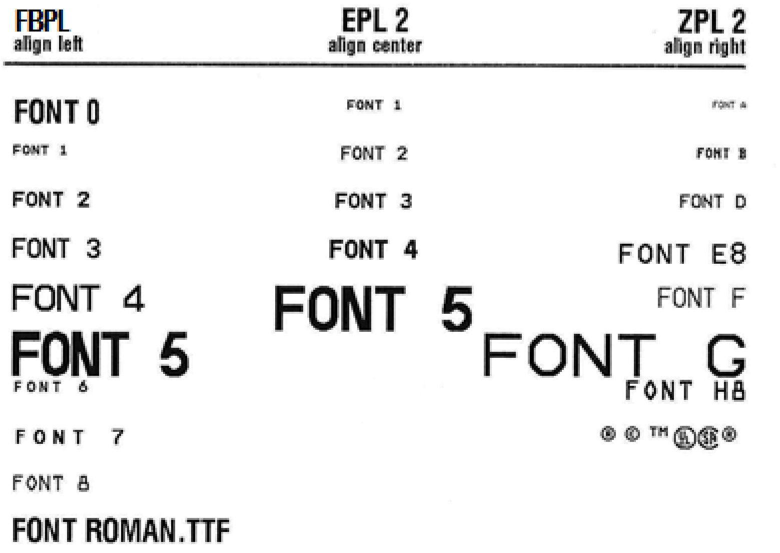
MODEL	Font Type									
	0	1	2	3	4	5	6	7	8	ROMAN.TTF
FBPL language printers	V	V	V	V	V	V	V	V	V	V

Example

Sample Code

```
SIZE 4,3
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10,"0",0,12,12,"FBPL"
TEXT 10,40,"0",0,8,8,"align left"
BAR 0,70,800,4
TEXT 10,110,"0",0,12,12,"FONT 0"
TEXT 10,160,"1",0,1,1,"FONT 1"
TEXT 10,210,"2",0,1,1,"FONT 2"
TEXT 10,260,"3",0,1,1,0,"FONT 3"
TEXT 10,310,"4",0,1,1,0,"FONT 4"
TEXT 10,360,"5",0,1,1,0,"FONT 5"
TEXT 10,410,"6",0,1,1,1,"FONT 6"
TEXT 10,460,"7",0,1,1,1,"FONT 7"
TEXT 10,510,"8",0,1,1,1,"FONT 8"
TEXT 10,560,"ROMAN.TTF",0,12,12,"FONT ROMAN.TTF"

TEXT 400,10,"0",0,12,12,2,"EPL 2"
TEXT 400,40,"0",0,8,8,2,"align center"
TEXT 400,110,"1.EFT",0,1,1,2,"FONT 1"
TEXT 400,160,"2.EFT",0,1,1,2,"FONT 2"
TEXT 400,210,"3.EFT",0,1,1,2,"FONT 3"
TEXT 400,260,"4.EFT",0,1,1,2,"FONT 4"
TEXT 400,310,"5.EFT",0,1,1,2,"FONT 5"

TEXT 800,10,"0",0,12,12,3,"ZPL 2"
TEXT 800,40,"0",0,8,8,3,"align right"
TEXT 800,110,"A.FNT",0,1,1,3,"FONT A"
TEXT 800,160,"B.FNT",0,1,1,3,"FONT B"
TEXT 800,210,"D.FNT",0,1,1,3,"FONT D"
TEXT 800,260,"E8.FNT",0,1,1,3,"FONT E8"
TEXT 800,310,"F.FNT",0,1,1,3,"FONT F"
TEXT 800,360,"G.FNT",0,1,1,3,"FONT G"
TEXT 800,410,"H8.FNT",0,1,1,3,"FONT H8"
TEXT 800,460,"GS.FNT",0,1,1,3,"ABCDEF"
PRINT 1
```

Result



FBPL
align left

FONT 0
FONT 1
FONT 2
FONT 3
FONT 4
FONT 5
FONT 6
FONT 7
FONT 8
FONT ROMAN.TTF

EPL 2
align center

FONT 1
FONT 2
FONT 3
FONT 4
FONT 5

ZPL 2
align right

FONT A
FONT B
FONT D
FONT E8
FONT F
FONT G
FONT H8
® © ™ UL SA ®

6.22 BLOCK

Description

This command prints paragraph on label.

Syntax

BLOCK x,y,width,height, "font",rotation,x-multiplication,y-multiplication,[space,]align,]fit,] "content"

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>x</td><td>The x-coordinate of the text</td></tr>
<tr><td>y</td><td>The y-coordinate of the text</td></tr>
<tr><td>width</td><td>The width of block for the paragraph in dots</td></tr>
<tr><td>height</td><td>The height of block for the paragraph in dots</td></tr>
<tr><td>font</td><td>Font name
<table>
<tr><td><b>0</b></td><td>Monotye CG Triumvirate Bold Condensed, font width and height is stretchable</td></tr>
<tr><td><b>1</b></td><td>8 x 12 fixed pitch dot font</td></tr>
<tr><td><b>2</b></td><td>12 x 20 fixed pitch dot font</td></tr>
<tr><td><b>3</b></td><td>16 x 24 fixed pitch dot font</td></tr>
<tr><td><b>4</b></td><td>24 x 32 fixed pitch dot font</td></tr>
<tr><td><b>5</b></td><td>32 x 48 dot fixed pitch font</td></tr>
<tr><td><b>6</b></td><td>14 x 19 dot fixed pitch font OCR-B</td></tr>
<tr><td><b>7</b></td><td>21 x 27 dot fixed pitch font OCR-B</td></tr>
<tr><td><b>8</b></td><td>14 x25 dot fixed pitch font OCR-A</td></tr>
<tr><td><b>ROMAN.TTF</b></td><td>Monotye CG Triumvirate Bold Condensed, font width and height proportion is fixed.</td></tr>
<tr><td><b>1.EFT</b></td><td>EPL2 font 1</td></tr>
<tr><td><b>2.EFT</b></td><td>EPL2 font 2</td></tr>
<tr><td><b>3.EFT</b></td><td>EPL2 font 3</td></tr>
<tr><td><b>4.EFT</b></td><td>EPL2 font 4</td></tr>
<tr><td><b>5.EFT</b></td><td>EPL2 font 5</td></tr>
<tr><td><b>A.FNT</b></td><td>ZPL2 font A</td></tr>
<tr><td><b>B.FNT</b></td><td>ZPL2 font B</td></tr>
<tr><td><b>D.FNT</b></td><td>ZPL2 font D</td></tr>
<tr><td><b>E8.FNT</b></td><td>ZPL2 font E8</td></tr>
<tr><td><b>F.FNT</b></td><td>ZPL2 font F</td></tr>
<tr><td><b>G.FNT</b></td><td>ZPL2 font G</td></tr>
<tr><td><b>H8.FNT</b></td><td>ZPL2 font H8</td></tr>
<tr><td><b>GS.FNT</b></td><td>ZPL2 font GS</td></tr>
</table>
</td></tr>
<tr><td>rotation</td><td>The rotation angle of text
0 : No rotation
90 : degrees, in clockwise direction
180 : degrees, in clockwise direction
270 : degrees, in clockwise direction</td></tr>
<tr><td>x-multiplication</td><td>Horizontal multiplication, up to 10x
Available factors: 1~10
For "ROMAN.TTF" true type font, this parameter is ignored.
For font "0", this parameter is used to specify the width (point) of true type font. 1 point=1/72 inch.</td></tr>
</table>

y-multiplication	Vertical multiplication, up to 10x Available factors: 1~10 For true type font, this parameter is used to specify the height (point) of true type font. 1 point=1/72 inch. For *.TTF font, x-multiplication and y-multiplication support floating value.
[space]	Add or delete the space between lines (in dots)
[align]	Text alignment. 0 : default (Left) 1 : Left 2 : Center 3 : Right
[fit]	Shrink the text so that it fits in the block 0 : No shrink (default) 1 : Shrink
content	Data in block. The maximum data length is 4092 bytes.

Note:

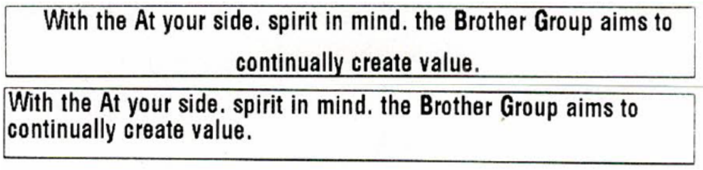
- ***Font "0" and "ROMAN.TTF" internal True Type Fonts are available in FBPL language printers.***
- ***If there is any double quote (") within the text, please change it to ¥["].***
- ***If font "0" is used, the font width and font height is stretchable by x-multiplication and y-multiplication parameter. It is expressed by pt (point). 1 point=1/72inch.***
- ***¥[R] means carriage return character 0x0D.***
- ***¥[L] means line feed character 0x0A.***
- ***EPL2 and ZPL2 fonts are also supported.***

Example

Sample Code

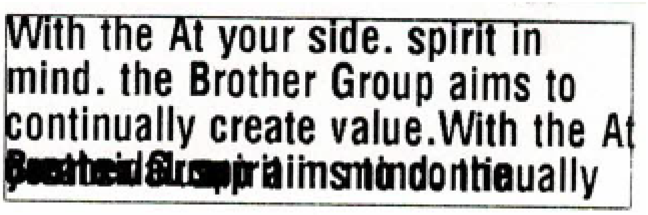
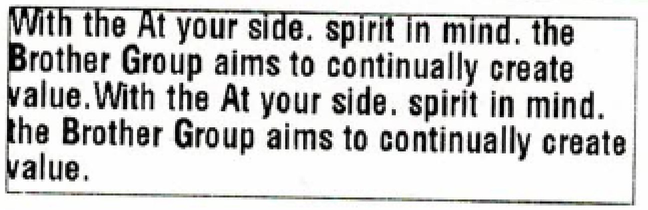
```
SIZE 4,0.5
GAP 0,0
DIRECTION 1
CLS
BOX 10,10,800,100,2
BLOCK 15,15,790,90, "0",0,8,8," With the At your side. spirit in mind. the Brother Group aims to continually create value."
PRINT 1
CLS
BOX 10,10,800,100,2
BLOCK 15,15,790,90,"0",0,8,8,20,2," With the At your side. spirit in mind. the Brother Group aims to continually create value."
PRINT 1
```

Result



With the At your side. spirit in mind. the Brother Group aims to
continually create value.

With the At your side. spirit in mind. the Brother Group aims to
continually create value.

Sample Code for [fit] Parameter	Result
DATA$ = "With the At your side. spirit in mind. the Brother Group aims to continually create value. With the At your side. spirit in mind. the Brother Group aims to continually create value." SIZE 4,1.5 GAP 0,0 DIRECTION 1 CLS BLOCK 20,20,500,170,"0",0,10,10,0,0,1,DATA$ BOX 20,20,500+20,170+20,2 PRINT 1 SIZE 4,1.5 GAP 0,0 DIRECTION 1 CLS BLOCK 20,20,500,170,"0",0,10,10,0,0,0,DATA$ BOX 20,20,500+20,170+20,2 PRINT 1	 With the At your side. spirit in mind. the Brother Group aims to continually create value.With the At [illegible]  With the At your side. spirit in mind. the Brother Group aims to continually create value.With the At your side. spirit in mind. the Brother Group aims to continually create value.

7 Status Polling and Immediate Commands

These commands support RS-232, USB and Ethernet.

7.1 <ESC>!?

Description

This command obtains the printer status at any time, even in the event of printer error. An inquiry request is solicited by sending an <ESC> (ASCII 27, escape character) as the beginning control character to the printer. A one byte character is returned, flagging the printer status. A 0 signifies the printer is ready to print labels.

Syntax

<ESC>!?

Hex Receive	Printer Status
00	Normal
01	Head opened
02	Paper Jam
03	Paper Jam and head opened
04	Out of paper
05	Out of paper and head opened
08	Out of ribbon
09	Out of ribbon and head opened
0A	Out of ribbon and paper jam
0B	Out of ribbon, paper jam and head opened
0C	Out of ribbon and out of paper
0D	Out of ribbon, out of paper and head opened
10	Pause
20	Printing
80	Other error

See Also

<ESC>!S

7.2 <ESC>!C

Description

This command restarts the printer and omits to run AUTO.BAS. The beginning of the command is an ESCAPE character (ASCII 27).

Syntax

<ESC>!C

Note:
When printer receives this command, printer will restart itself no matter AUTO.BAS exists or not.

See Also

<ESC>!Q

7.3 <ESC>!D

Description

This command is used to disable immediate command, ex. *<ESC>!R <RSC>!?* <ESC>!C and so on, which is starting by <ESC>!. The beginning of the command is an ESCAPE character (ASCII 27).

Syntax

<ESC>!D

See Also

~!E

7.4 <ESC>!O

Description

This command is using to cancel the PAUSE status of printer. The beginning of the command is an ESCAPE character (ASCII 27).

Syntax

<ESC>!O

See Also

<ESC>!P

7.5 <ESC>!P

Description

This command is using to PAUSE the printer. The beginning of the command is an ESCAPE character (ASCII 27).

Syntax

<ESC>!P

See Also

<ESC>!O

7.6 <ESC>!Q

Description

This command restarts the printer and omits to run AUTO.BAS. The beginning of the command is an ESCAPE character (ASCII 27).

Syntax

<ESC>!Q

Note:
If there is no AUTO.BAS inside the printer, the printer will not restart itself.

See Also

<ESC>!C

.

7.7 <ESC>!R

Description

This command resets the printer. The beginning of the command is an ESCAPE character (ASCII 27). The files downloaded in memory will be deleted. This command cannot be sent in dump mode.

Syntax

<ESC>!R

See Also

<ESC>!?

7.8 <ESC>!S

Description

This command obtains the printer status at any time, even in the event of printer error. An inquiry request is solicited by sending an <ESC> (ASCII 27, escape character) as the beginning control character to the printer. 8 bytes will be returned, flagging the printer status.

Syntax

<ESC>!S

Response Format

<STX>[4-byte status]<ETX><CR><LF>

Status Byte #1: message

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Hex	ASCII	Char	Meaning
0	1	0	0	0	0	0	0	40	64	@	Normal
0	1	1	0	0	0	0	0	60	96	`	Pause
0	1	0	0	0	0	1	0	42	66	B	Backing label
0	1	0	0	0	0	1	1	43	67	C	Cutting
0	1	0	0	0	1	0	1	45	69	E	Printer error
0	1	0	0	0	1	1	0	46	70	F	Form feed
0	1	0	0	1	0	1	1	4B	75	K	Waiting to press print key
0	1	0	0	1	1	0	0	4C	76	L	Waiting to take label
0	1	0	1	0	0	0	0	50	80	P	Printing batch
0	1	0	1	0	1	1	1	57	87	W	Imaging

Status Byte #2: warning

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Hex	ASCII	Char	Meaning
0	1	0	0	0	0	0	0	40	64	@	Normal
0	1	0	0	0	0	0	1	41	65	A	Paper low
0	1	0	0	0	0	1	0	42	66	B	Ribbon low
0	1	0	0	0	1	0	0	44	68	D	Reversed
0	1	0	0	1	0	0	0	48	72	H	Receive buffer bull
0	1	1	0	0	0	0	0	60	96	`	Reversed

Status Byte #3: error

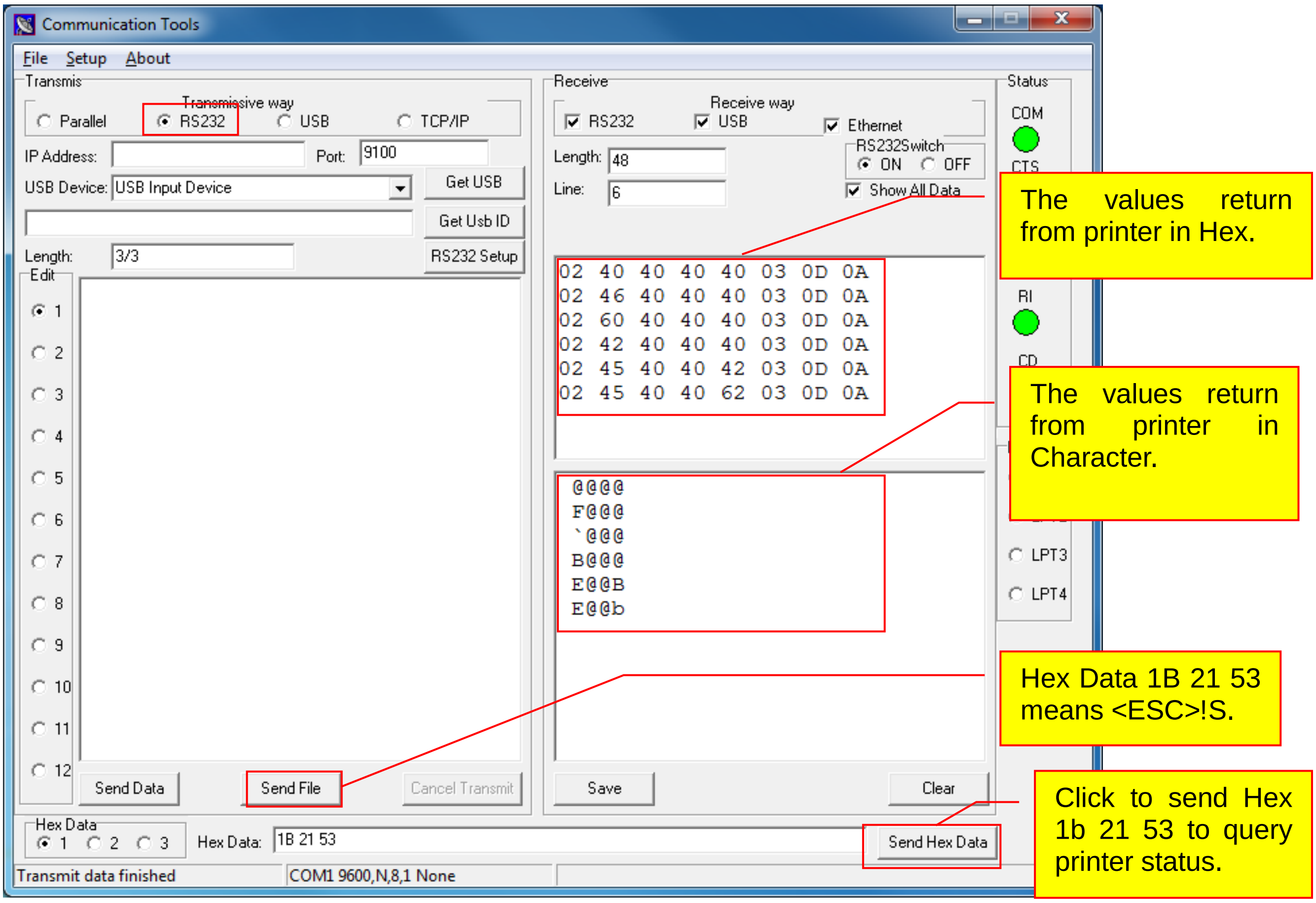
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Hex	ASCII	Char	Meaning
0	1	0	0	0	0	0	0	40	64	@	Normal
0	1	0	0	0	0	0	1	41	65	A	Print head overheat
0	1	0	0	0	0	1	0	42	66	B	Stepping motor overheat
0	1	0	0	0	1	0	0	44	68	D	Print head error
0	1	0	0	1	0	0	0	48	72	H	Cutter jam
0	1	0	1	0	0	0	0	50	80	P	Insufficient memory

Status Byte #4: error

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Hex	ASCII	Char	Meaning
0	1	0	0	0	0	0	0	40	64	@	Normal
0	1	0	0	0	0	0	1	41	65	A	Paper empty
0	1	0	0	0	0	1	0	42	66	B	Paper jam
0	1	0	0	0	1	0	0	44	68	D	Ribbon empty
0	1	0	0	1	0	0	0	48	72	H	Ribbon jam
0	1	1	0	0	0	0	0	60	96	`	Print head open

Example

Test <ESC>!S by CommTool via RS-232 port.



Communication Tools

Transmissive way: ○ Parallel ● RS232 ○ USB ○ TCP/IP

IP Address: Port: 9100

USB Device: USB Input Device

Length: 3/3

Receive way: ☑ RS232 ☑ USB ☑ Ethernet

RS232Switch: ● ON ○ OFF

☑ Show All Data

Length: 48

Line: 6

```
02 40 40 40 40 03 0D 0A
02 46 40 40 40 03 0D 0A
02 60 40 40 40 03 0D 0A
02 42 40 40 40 03 0D 0A
02 45 40 40 42 03 0D 0A
02 45 40 40 62 03 0D 0A
```

```
@@@@
F@@@
`@@@
B@@@
E@@B
E@@b
```

Hex Data: ● 1 ○ 2 ○ 3

Hex Data: 1B 21 53

Transmit data finished

COM1 9600,N,8,1 None

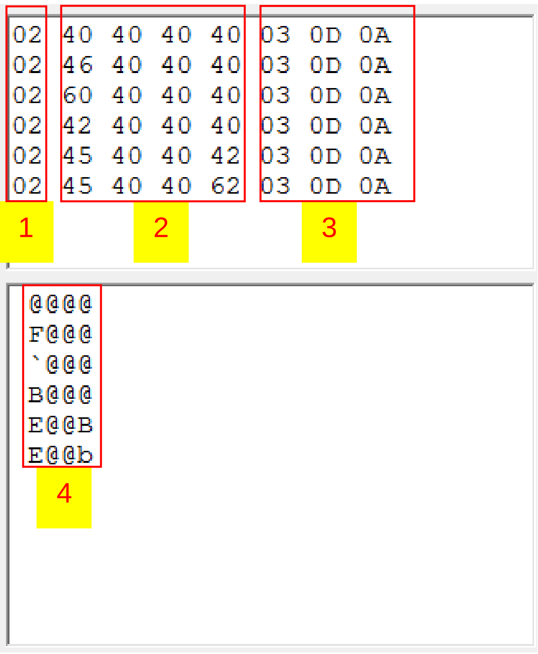
The values return from printer in Hex.

The values return from printer in Character.

Hex Data 1B 21 53 means <ESC>!S.

Click to send Hex 1b 21 53 to query printer status.

Result



```
02 40 40 40 40 03 0D 0A
02 46 40 40 40 03 0D 0A
02 60 40 40 40 03 0D 0A
02 42 40 40 40 03 0D 0A
02 45 40 40 42 03 0D 0A
02 45 40 40 62 03 0D 0A
```

```
@@@@
F@@@
`@@@
B@@@
E@@B
E@@b
```

Item	Meaning
1	The start character of returned value.
2	The 4-byte status in Hex.
3	The end characters of returned value.
4	4-byte status in characters. @@@@: The printer is normal for use. F@@@: The printer is feeding label. `@@@: Printer is in PAUSE mode. B@@@: The printer is backing label. E@@B: Printer is in error “Paper Jam”. E@@b: Printer is in error “Paper Jam” & “Head open”.

See Also

<ESC>!?

7.9 <ESC>!F

Description

This command is using to feed a label. This function is the same as to press the FEED button. The beginning of the command is an ESCAPE character (ASCII 27).

Syntax

<ESC>!F

7.10 <ESC>!.

Description

This command can cancel all printing files. The beginning of the command is an ESCAPE character (ASCII 27).

Syntax

```
<ESC>!.
```

7.11 ~!@

Description

This command inquires the mileage of the printer. The integer part of mileage is returned (the decimal part of mileage is not return) to the PC in ASCII characters. The ending character of mileage is 0x0D.

Syntax

```
~!@
```

Example

```
~!@
```

7.12 ~!A

Description

This command inquires the free memory of the printer. The number of bytes of free memory is returned in decimal digits, with 0x0d as ending code of PC.

Syntax

```
~!A
```

Example

```
~!A
```

See Also

FILES

7.13 ~!C

Description

This command inquires the presence of Real Time Clock. One byte is return from the printer, indicating whether or not the RTC is installed.

Syntax

~!C

Return value	Description
0	RTC is not installed.
1	RTC is installed.

Example

```
~!C
```

7.14 ~!D

Description

This command enters the printer into DUMP mode. In DUMP mode, the printer outputs code directly without interpretation.

Syntax

~!D

Example

```
~!D
```

7.15 ~!E

Description

This command is used to enable immediate command, ex. *<ESC>!R <RSC>!?* <ESC>!C and so on, which is starting by <ESC>!.

Syntax

~!E

Example

~!E

See also

<ESC>!D

7.16 ~!F

Description

This command inquires all about files resident in the printer memory, and fonts installed in the memory module. The filename are returned in ASCII characters. Each file name ends with 0x0D. The ending character is 0x1A. Entering this command multiple times will cycle through the files resident on memory.

Syntax

```
~!F
```

Example

```
~!F
```

See Also

FILES

7.17 ~!I

Description

The command inquires the code page and country setting of the printer.

Syntax

~!I

The returned information is given in the following format:

code page, country code
ex : 8 bit: 437, 001
7 bit: USA, 001

Regarding the code pages and country codes supported by the printer, please refer to the **CODEPAGE** and **COUNTRY** command respectively.

Example

~!I

See Also

COUNTRY, CODEPAGE

7.18 ~!T

Description

This command inquires the model name and number of the printer. This information is returned in ASCII characters.

Syntax

~ !T

Example

~!T

8 Commands for Windows Driver

8.1 !B

Description

This command stores bitmap image data in the memory. Behind the nnn is the bitmap data.

Syntax

!Bnnn

Parameter	Description
nnn	The number of bytes of image data sent from PC to printer, expressed in 3 decimal digits.

Example

!B100

See Also

BITMAP

8.2 !J

Description

This command prints bitmap data at the specified position (in y-direction).

Syntax

!Jnnnn

Parameter	Description
nnn	Print image at the specified position in y-direction. The position is expressed in 4 decimal digits.

Example

!J0100

See Also

FEED

8.3 !N

Description

This command prints a specified number of labels.

Syntax

!Nnnn

Parameter	Description
nnn	Specifies the number of copies to be printed.

Example

!N001

9 File Management Commands

9.1 DOWNLOAD

Description

"DOWNLOAD" is a header of the file that is to be saved in the printer’s memory. The downloaded files can be divided into two categories: program files and data files (including text data files, PCX graphic files and bitmap font files) The detailed descriptions regarding the download syntax for different files are as follows:

Maximum numbers of file saved in DRAM:

50 files for FBPL language printers

Maximum numbers of file saved in Flash memory:

256 files for FBPL language printers

If "AUTO.BAS" exists in the printer memory, it will be automatically executed upon printer startup. To disable the auto execution function, please follow the procedures below.

Ignore AUTO.BAS
For two buttons desktop printer series Hold down the **PAUSE** and **FEED** buttons and turn on the printer power. Do not release the buttons until the three LEDs flash in turn. Printer will Ignore AUTO.BAS and initialize the printer.
For one button desktop printer series Hold the FEED key and power on the switch. Release the FEED key while LED becomes solid green to prevent the printer from running "AUTO.BAS". The LED color will be changed as following pattern: Orange → red (5 times) → orange (5 times) → green (5 times) → green and orange (5 times) → red and orange (5 times) → solid green
For three buttons industrial printer series Hold the FEED key and power on the switch. The ERROR LED will be on. Printer is now ready to use.
For six or two buttons industrial printer series Hold the PAUSE and FEED keys and power on the switch. "AUTO.BAS" will not be executed after printer initialization, and will now be ready for use. Alternatively, hold the PAUSE key and power on the switch. After sensor calibration, the "AUTO.BAS" will not be executed. Printer is now ready for use.

Syntax

1. **Download a program file:**

DOWNLOAD [n,] "FILENAME.BAS"

Parameter	**Description**
n	Specify memory used to save downloaded files. ***N is ignored***: Download files to DRAM only. If you would like to save the files from DRAM to Flash memory before turning off power, issue the MOVE command to printer. ***F***: Download files to main board flash memory. ***E***: Download files to expansion memory module.
FILENAME.BAS	The filename resident in printer memory.

Note:

- ***Filenames are case sensitive.***
- ***File extensions must be ".BAS"***
- ***Filenames must be in 8.3 format.***
- ***It should use with EOP command.***
- ***If memory is not specified, all files will be downloaded to DRAM.***

- ***The priority of AUTO.BAS in each memory device:***
  DRAM > CARD (Ext. FLASH) > FLASH
- ***No Battery is used to back up files in DRAM. Which will be lost in the event printer power is lost.***

2. **Download a data file:**

DOWNLOAD [n,] “FILENAME”,DATA SIZE,DATA CONTENT…

Parameter	**Description**
n	Specify the memory location to save downloaded files. ***N is ignored***: Download files to DRAM only. If you would like to save the files from DRAM to Flash memory before turning off power, issue the MOVE command to printer. ***F***: Download files to main board flash memory. ***E***: Download files to expansion memory module.
FILENAME	The name of data file that will remain resident in the printer memory (case sensitive).
DATA SIZE	The actual size in bytes of the data file (without header)
DATA CONTENT	The data which will be downloaded into printer.

Note:

- ***For text data files, CR (carriage return) 0x0D and LF (Line Feed) 0x0A is the separator of data.***
- ***If memory is not specified, all files will be downloaded to DRAM.***
- ***No Battery is used to back up files in DRAM. Which will be lost in the event printer power is lost.***
- ***When writing a download program, "DOWNLOAD" header must be placed in the beginning of file, and "EOP" must be placed at the end of program.***
- ***To run the program, call the main filename without BAS extension or use RUN command to start the download program.***

Example

Sample code (The example program listed below will download to printer SDRAM.)

```
DOWNLOAD "EXAMPLE.BAS"
SIZE 4,4
GAP 0,0
DIRECTION 1
SET TEAR ON
CLS
TEXT 100,100, "3",0,1,1, "EXAMPLE PROGRAM"
PRINT 1
EOP
```

See Also

EOP, RUN, PUTBMP, PUTPCX, INPUT, FILES, ~!F

9.2 EOP

Description

End of program. To declare the start and end of BASIC language commands used in a program, DOWNLOAD “FILENAME.BAS” must be added in the first line of the program, and “EOP” statement at the last line of program.

Syntax

EOP

Example

Sample code (The example program listed below will download to printer SDRAM.)

```
DOWNLOAD "DEMO.BAS"
SIZE 4,4
GAP 0,0
DIRECTION 1
SET TEAR ON
CLS
TEXT 100,100, "3",0,1,1, "DEMO PROGRAM"
PRINT 1
EOP
```

See Also

DOWNLOAD, INPUT, FILES, ~!F

9.3 FILES

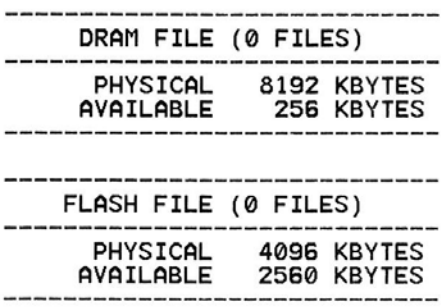
Description

This command prints out the total memory size, available memory size and files lists (or lists the files through RS-232) in the printer memory (both FLASH memory and DRAM).

Syntax

FILES

Example

Sample code	Result
FILES	 ------------------------------ DRAM FILE (0 FILES) ------------------------------ PHYSICAL 8192 KBYTES AVAILABLE 256 KBYTES ------------------------------ ------------------------------ FLASH FILE (0 FILES) ------------------------------ PHYSICAL 4096 KBYTES AVAILABLE 2560 KBYTES ------------------------------

See Also

~!F, KILL

9.4 KILL

Description

This command deletes a file in the printer memory. The wild card (*) will delete all files resident in specified DRAM or FLASH memory.

Syntax

KILL [n], “FILENAME”

Parameter	Description
n	Specify the memory location that files will be deleted. ***N is ignored***: Kill files saved in DRAM. ***F***: Kill files from main board flash memory. ***E***: Kill files from expansion memory module.
FILENAME	The name of data file that will delete in the printer memory (case sensitive)

Note:

- ***If optional parameter n is not specified, firmware will delete the file in DRAM.***
- ***Syntax example***

KILL "FILENAME"	***: Delete the specify file in DRAM.***
KILL "*.PCX"	***: Delete all PCX files in DRAM.***
KILL "* "	***: Delete all files in DRAM.***
KILL F, "FILENAME"	***: Delete the specify file in FLASH.***
KILL E, " *.PCX "	***: Delete all PCX file in extension memory card.***

Model	Support		
	KILL "*"	KILL "*" MOVE	KILL F, "*"
FBPL programming printer	**V**		**V**

Example

Users can use printer SELFTEST utility to list printer configurations and files saved in the printer memory, or use the FILES command to print the downloaded file list in printer. Follow the steps below to delete files in the printer memory via parallel port connection.

```
C:¥>COPY CON LPT1<ENTER>
FILES<ENTER>
<CTRL><Z><ENTER>
C:¥>COPY CON LPT1<ENTER>
KILL "DEMO.BAS "   <ENTER>
<CTRL><Z><ENTER>
C:¥>COPY CON LPT1<ENTER>
FILES<ENTER>
<CTRL><Z><ENTER>
```

Note: <ENTER> stands for PC keyboard "ENTER" key. <CTRL><Z> means to hold PC keyboard "CTRL" key then press the PC keyboard <Z> key

See Also

~!F, FILES

9.5 MOVE

Description

This command moves downloaded files from DRAM to FLASH memory.

Syntax

MOVE

See Also

DOWNLOAD, EOP

9.6 RUN

Description

This command executes a program resident in the printer memory. It is available for FBPL language printers only.

Syntax

```
RUN "FILENAME.BAS"
```

Note:
** This command can be replaced to filename that without typing ".BAS".*

Example

Sample code	Result
DOWNLOAD "DEMO.BAS" SIZE 4,4 GAP 0,0 DIRECTION 1 SET TEAR ON CLS TEXT 100,100, "3",0,1,1, "DEMO PROGRAM" PRINT 1 EOP RUN "DEMO.BAS"	
DOWNLOAD "DEMO.BAS" SIZE 4,4 GAP 0,0 DIRECTION 1 SET TEAR ON CLS TEXT 100,100, "3",0,1,1, "DEMO PROGRAM" PRINT 1 EOP DEMO	

See Also

DOWNLOAD, EOP

10 BASIC Commands and Functions

10.1 ABS( )

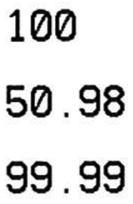
Description

This function returns the absolute value of an integer, floating point or variable.

Syntax

ABS (VARIABLE)

Example

Sample code	Result
DOWNLOAD "TEST.BAS" SIZE 4,4 GAP 0,0 DIRECTION 1 SET TEAR ON CLS A=ABS(-100) B=ABS(-50.98) C=-99.99 TEXT 100,100, "3",0,1,1,STR$(A) TEXT 100,150, "3",0,1,1,STR$(B) TEXT 100,200, "3",0,1,1,STR$(ABSI) PRINT 1 EOP RUN "TEST.BAS"	 100 50.98 99.99

See Also

DOWNLOAD, EOP

10.2 ASC( )

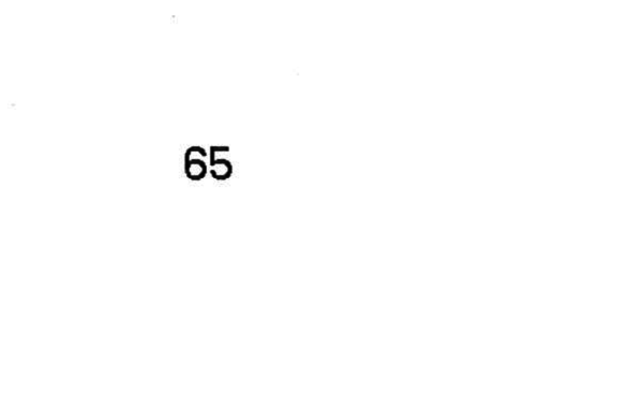
Description

This function returns the ASCII code of the character.

Syntax

ASC (" A ")

Example

Sample code	Result
DOWNLOAD "TEST.BAS" SIZE 4,4 GAP 0,0 DIRECTION 1 SET TEAR ON CLS CODE1=ASC(" A ") TEXT 100,100, " 3 ",0,1,1,STR$(CODE1) PRINT 1 EOP RUN "TEST.BAS"	 65

See Also

DOWNLOAD, EOP, STR$()

10.3 CHR$( )

Description

This function returns the character with the specified ASCII code.

Syntax

```
CHR$(n)
```

Parameter	Description
n	The ASCII code

Example

Sample code	Result
DOWNLOAD "TEST.BAS" SIZE 4,4 GAP 0,0 DIRECTION 1 SET TEAR ON CLS A=75 WORD$=CHR$(A) TEXT 100,100, "3",0,1,1,WORD$ PRINT 1 EOP RUN "TEST.BAS"	K

See Also

DOWNLOAD, EOP, STR$(), ASC$()

10.4 XOR$( )

Description

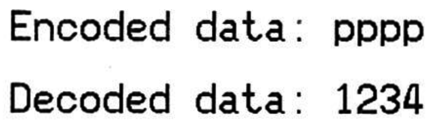
This command can encode the original data to a new data by logic XOR.

Syntax

XOR$(data$,password$)

Parameter	Description
data$	The original data needs to be encoded by password$.
Password$	This parameter will be used to create the new data.

Example

Sample code	Result
data$="1234" password$="ABCD" encoded$=XOR$(data$,password$) deconded$=XOR$(encoded$,password$) SIZE 4,0.5 GAP 0,0 CLS TEXT 10,10,"3",0,1,1, "Encoded data: "+encoded$ TEXT 10,60, "3",0,1,1, "Decoded data: "+deconded$ PRINT 1	 Encoded data: pppp Decoded data: 1234

10.5 END

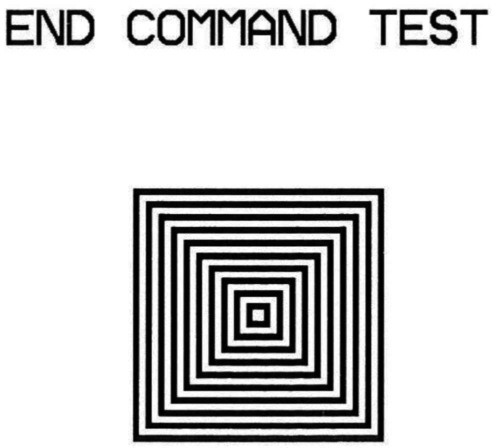
Description

This command states the end of program.

Syntax

```
END
```

Example

Sample code	Result
DOWNLOAD "DEMO.BAS" SIZE 4,2 GAP 0,0 DIRECTION 1 CLS TEXT 200,60, "4",0,1,1, "END COMMAND TEST" X=300 Y=200 X1=500 Y1=400 GOSUB DR_LINE PRINT 1 END :DR_LINE FOR I=1 TO 100 STEP 10 BOX X+I,Y+I,X1-I,Y1-I,5 NEXT RETURN EOP DEMO	 END COMMAND TEST

See Also

DOWNLOAD, EOP, GOSUB

10.6 EOF( )

Description

This function is used to detect an opened download file to see whether it has reached the end of file.

Syntax

EOF (File Handle)

Parameter	Description
File handle	Either 0 or 1

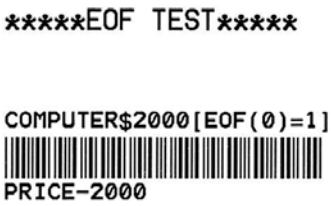
Return value	Description
None-zero	End of file
0	Not end of file

Example

Sample code

```
DOWNLOAD "DATA",16,COMPUTER
2000

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.0,0
DIRECTION 1
CLS
OPEN "DATA",0
SEEK 0,0
Y=110
TEXT 10,10, "3",0,1,1, "*****EOF TEST*****"
:A
Temp$=""
READ 0,ITEM$,P
TEXT 10,Y,"2",0,1,1,ITEM$+"$" +STR$(P)+"[EOF(0)="+STR$(EOF(0))+"]"
BARCODE 10,Y+25,"39",40,1,0,2,4,"PRICE-"+STR$(P)
Y=Y+100
IF EOF(0)=0 THEN GOTO A
PRINT 1
EOP
DEMO
```

Result



```
*****EOF TEST*****

COMPUTER$2000[EOF(0)=1]
PRICE-2000
```

See Also

DOWNLOAD, EOP, OPEN, READ, SEEK

10.7 OPEN

Description

This command opens a downloaded file and establishes the file handle. Up to two file handles are supported, thus only up to two files can be opened simultaneously. The file to be opened should be downloaded prior to using this command. When opening a file, the firmware will search automatically to see if the file exists in the on board flash memory or extended memory card. ***If the file doesn't exist, the printer will create this file in the onboard FLASH.***

Syntax

OPEN [memory ID,] "filename",file handle

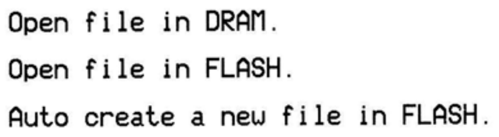
<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>[memory ID]</td><td>Optional. Open the file in specific memory device.
<table>
<tr><th>ID</th><th>Memory device</th></tr>
<tr><td>Omitted</td><td>DRAM</td></tr>
<tr><td>F</td><td>FLASH</td></tr>
<tr><td>E</td><td>CARD</td></tr>
</table>
</td></tr>
<tr><td>filename</td><td>The file downloaded in the printer memory</td></tr>
<tr><td>file handle</td><td>Either 0 or 1</td></tr>
</table>

Example

Sample code

```
DOWNLOAD "DATA.DAT",18,Open file in DRAM.
DOWNLOAD F, "DATA.DAT",19,Open file in FLASH.
DOWNLOAD "TEST.BAS"
data1$=""
data2$=""
data3$=""
OPEN "DATA.DAT",0
READ 0,data1$
CLOSE 0
OPEN F, "DATA.DAT",0
READ 0,data2$
CLOSE 0
KILL F, "*"
OPEN "NEW.DAT",0
SEEK 0,0
WRITE 0, "Auto create a new file in FLASH."
SEEK 0,0
READ 0,data3$
CLOSE 0
SIZE 4,1
GAP 0,0
CLS
TEXT 10,10,"3",0,1,1,data1$
TEXT 10,60,"3",0,1,1,data2$
TEXT 10,110,"3",0,1,1,data3$
PRINT 1
EOP
TEST
```

Result



```
Open file in DRAM.
Open file in FLASH.
Auto create a new file in FLASH.
```

See Also

DOWNLOAD, EOP, READ, WRITE, SEEK, CLOSE

10.8 CLOSE

Description

Close the file handle which is open by command OPEN.

Syntax

CLOSE file handle

Parameter	Description
file handle	Either 0 or 1

Example

See the example in command OPEN.

10.9 WRITE

Description

This command writes data to a downloaded data file. Two files can be open simultaneously, by virtue of printer support for two file handles.

Syntax

WRITE file handle,variables

Parameter	Description
file handle	0 or 1
variables	string, integer or float point variable

See Also

READ, DOWNLOAD, EOP, OPEN, EOF, LOF, SEEK, FREAD$()

10.10 READ

Description

This command reads data from downloaded data file.

Syntax

READ file handle,variables

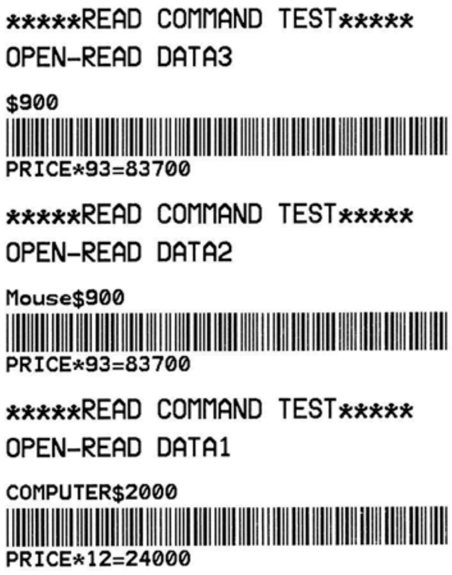
Parameter	Description
file handle	0 or 1
variables	string, integer or float point variable

Example

Sample code

```
DOWNLOAD "DATA1",20,COMPUTER
2000
12
DOWNLOAD "DATA2",16,Mouse
900
93
DOWNLOAD "DEMO.BAS"
SIZE 3,1
GAP 0,0
DIRECTION 1
I=0
Y=100
OPEN "DATA1",0
OPEN "DATA2",1
SEEK 0,0
SEEK 1,0
:Start
CLS
TEXT 10,10,"3",0,1,1,"*****READ COMMAND TEST*****"
TEXT 10,50,"3",0,1,1,"OPEN-READ DATA"+STR$(I+1)
ITEM$=""
READ I,ITEM$,P,Q
TEXT 10,Y, "2",0,1,1,ITEM$+"$" +STR$(P)
BARCODE 10,Y+25, "39 ",40,1,0,2,4, "PRICE* "+STR$(Q)+ "= "+STR$(P*Q)
Y=Y+100
PRINT 1
Y=100
IF I<=1 THEN
IF EOF(I)=1 THEN
I=I+1
GOTO Start
ELSE
GOTO Start
ENDIF
ELSE
END
ENDIF
EOP
DEMO
```

Result



```
*****READ COMMAND TEST*****
OPEN-READ DATA3
$900
PRICE*93=83700

*****READ COMMAND TEST*****
OPEN-READ DATA2
Mouse$900
PRICE*93=83700

*****READ COMMAND TEST*****
OPEN-READ DATA1
COMPUTER$2000
PRICE*12=24000
```

See Also

DOWNLOAD, EOP, OPEN, EOF, LOF, SEEK, FREAD$()

10.11 SEEK

Description

This command shifts the specified file pointer to a certain position.

Syntax

SEEK file handle,offset

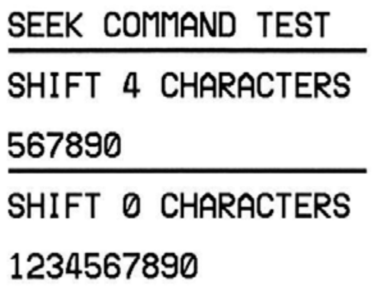
Parameter	Description
file handle	0 or 1
offset	the offset characters which are shifted to a new position

Example

Sample code

```
DOWNLOAD "DATA",12,1234567890
DOWNLOAD "TEST.BAS"
SIZE 4,1.5
GAP 0,0
DIRECTION 1
REFERENCE 0,0
CLS
OPEN "DATA",0
SEEK 0,4
READ 0,Num$
TEXT 100,10,"3",0,1,1,"SEEK COMMAND TEST"
BAR 100,40,300,4
TEXT 100,60,"3",0,1,1,"SHIFT 4 CHARACTERS"
TEXT 100,110,"3",0,1,1,Num$
BAR 100,140,300,4
SEEK 0,0
READ 0,Num$
TEXT 100,160,"3",0,1,1,"SHIFT 0 CHARACTERS"
TEXT 100,210,"3",0,1,1,Num$
PRINT 1
EOP
TEST
```

Result



```
SEEK COMMAND TEST
SHIFT 4 CHARACTERS
567890
SHIFT 0 CHARACTERS
1234567890
```

See Also

DOWNLOAD, EOP, OPEN, READ, EOF, LOF, FREAD$()

10.12 LOF( )

Description

This function returns the size of the specified file.

Syntax

LOF("FILENAME")

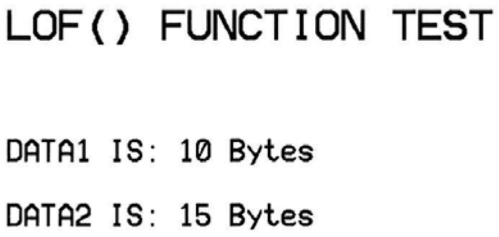
Parameter	Description
FILENAME	The file downloaded in the printer memory.

Example

Sample code

```
DOWNLOAD "DATA1",10,1234567890

DOWNLOAD "DATA2",15,ABCDEFGHIJKLMNO

DOWNLOAD "LofTest.BAS"
SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS
OPEN "DATA1",0
OPEN "DATA2",1
TEXT 10,20,"4",0,1,1,"LOF() FUNCTION TEST"
J=LOF("DATA1")
K=LOF("DATA2")
TEXT 10,140,"3",0,1,1,"DATA1 IS: "+STR$(J)+"Bytes"
TEXT 10,200,"3",0,1,1,"DATA2 IS: "+STR$(K)+"Bytes"
PRINT 1
EOP
LofTest
```

Result



```
LOF() FUNCTION TEST


DATA1 IS: 10 Bytes

DATA2 IS: 15 Bytes
```

See Also

DOWNLOAD, EOP, OPEN, READ, EOF, SEEK, FREAD$()

10.13 LOC( )

Description

This function returns the current read/write position within an open file.

Syntax

LOC(file handle)

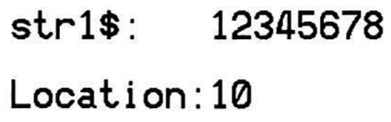
Parameter	Description
file handle	0 or 1

Example

Sample code

```
DOWNLOAD "DATA.DAT",30,12345678
12345678
12345678

DOWNLOAD "TEST.BAS"
str1$ = ""
location = 0
OPEN "DATA.DAT",0
READ 0,str1$
location = LOC(0)
CLOSE 0
SIZE 4,1
GAP 0,0
CLS
TEXT 10,10,"3",0,1,1,"str1$:     "+str1$
TEXT 10,60,"3",0,1,1,"Location:"+STR$(location)
PRINT 1
EOP
TEST
```

Result



```
str1$:    12345678
Location:10
```

10.14 FREAD$( )

Description

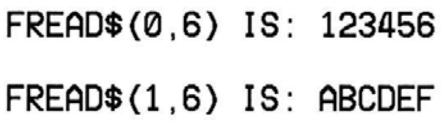
This function reads a specified number of bytes of data from a file.

Syntax

FREAD$ (file handle,byte)

Parameter	Description
file handle	0 or 1
byte	Number of bytes to be read

Example

Sample code	Result
DOWNLOAD "DATA1",10,1234567890 DOWNLOAD "DATA2",15,ABCDEFGHIJKLMNO DOWNLOAD "OPEN2.BAS" SIZE 4,1 GAP 0,0 DIRECTION 1 CLS OPEN "DATA1",0 OPEN "DATA2",1 SEEK 0,0 SEEK 1,0 Y$=FREAD$(0,6) Z$=FREAD$(1,6) TEXT 10,100,"3",0,1,1,"FREAD$(0,6) IS: " +Y$ TEXT 10,150,"3",0,1,1,"FREAD$(1,6) IS: " +Z$ PRINT 1 EOP OPEN2	 FREAD$(0,6) IS: 123456 FREAD$(1,6) IS: ABCDEF

See Also

DOWNLOAD, EOP, OPEN, READ, EOF, LOF(), SEEK

10.15 PUT

Description

One byte is appended into file.

Syntax

```
PUT file handle,var1$[, var2$][,var3$][, …]
PUT file handle,var1[, var2][,var3][, …]
PUT file handle,var1$[, var2$][,var3][, …]
```

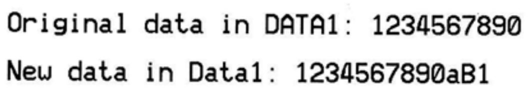
Parameter	Description
file handle	0 or 1
var$	Data is a character
var	Data is ASCII value

Example

Sample code

```
DOWNLOAD "DATA1",10,1234567890
DOWNLOAD "TEST.BAS"
str1$ = ""
str2$ = ""
OPEN "DATA1",0
SEEK 0,0
READ 0,str1$
PUT 0,"a","B",49
SEEK 0,0
READ 0,str2$
CLOSE 0

SIZE 4,0.5
GAP 0,0
CLS
TEXT 10, 10,"3",0,1,1,"Original data in DATA1: "+str1$
TEXT 10, 60,"3",0,1,1,"New data in Data1: "+str2$
PRINT 1
EOP
TEST
```

Result



```
Original data in DATA1: 1234567890
New data in Data1: 1234567890aB1
```

See Also

DOWNLOAD, EOP, OPEN, READ, EOF, LOF(), SEEK, GET

10.16 GET

Description

Get one byte from file.

Syntax

GET file handle,var1$[,var2$][,var3$][, …]
GET file handle,var1[,var2][,var3][, …]
GET file handle,var1$[,var2$][,var3][, …]

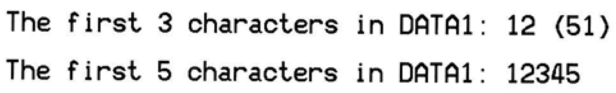
Parameter	Description
file handle	0 or 1
var$	Get a character
var	Get ASCII value

Example

Sample code

```
DOWNLOAD "DATA1",10,1234567890
DOWNLOAD "TEST.BAS"
a$=""
b$=""
c=0
d$=""
e$=""
OPEN "DATA1",0
SEEK 0,0
GET 0,a$,b$,c
SEEK 0,0
FOR I=1 TO 5
GET 0,d$
e$=e$+d$
NEXT

SIZE 4,0.5
GAP 0,0
CLS
TEXT 10,10,"3",0,1,1,"The first 3 characters in DATA1: "+ a$+b$+" ("+STR$(c)+")"
TEXT 10,60,"3",0,1,1,"The first 5 characters in DATA1: "+e$
PRINT 1
EOP
TEST
```

Result



```
The first 3 characters in DATA1: 12 (51)
The first 5 characters in DATA1: 12345
```

See Also

DOWNLOAD, EOP, OPEN, READ, EOF, LOF(), SEEK, PUT

10.17 COPY

Description

Copy the existed file from CARD to FLASH.

Syntax

COPY [memory ID of source,] "filename of source",[memory ID of new file,] "new filename"

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>memory ID of source</td><td>Optional.<table><tr><th>ID</th><th>Memory device</th></tr><tr><td>Omitted</td><td>DRAM</td></tr><tr><td>F</td><td>FLASH</td></tr><tr><td>E</td><td>CARD</td></tr></table></td></tr>
<tr><td>source filename</td><td>The file in CARD which you want to copy to on board FLASH.</td></tr>
<tr><td>Memory ID of new file</td><td>Optional.<table><tr><th>ID</th><th>Memory device</th></tr><tr><td>Omitted</td><td>DRAM</td></tr><tr><td>F</td><td>FLASH</td></tr></table></td></tr>
<tr><td>new filename</td><td>The new filename you want to use in the on board FLASH.</td></tr>
</table>

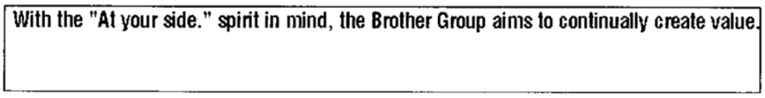
Example

Sample Code

```
DOWNLOAD "DATA_D.DAT",105, With the "At your side." spirit in mind. the Brother Group aims to continually create value.
DOWNLOAD "TEST.BAS"
KILL F,"*"
COPY "DATA_D.DAT",F,"DATA_F.DAT"
OPEN "DATA_F.DAT",0
SEEK 0,0
data$=FREAD$(0,LOF("DATA_F.DAT"))
CLOSE 0
SIZE 4,0.5
GAP 0,0
CLS
BOX 10,10,800,100,2
BLOCK 15,15,790,90,"0",0,8,8,20,2,data$
PRINT 1
EOP
TEST
```

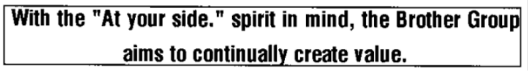
Result

203 dpi



With the "At your side." spirit in mind, the Brother Group aims to continually create value.

300 dpi



With the "At your side." spirit in mind, the Brother Group
aims to continually create value.

See Also

DOWNLOAD, EOP, OPEN, FREAD$(), EOF, LOF(), SEEK, CLOSE

10.18 FOR…NEXT LOOP

Description

Loop is used to execute one or more lines of program repetitively. A loop counter value specifies the number of executions. Nested loops are allowed (up to 39 nested loops) in this printer. Jumping out in the middle of the FOR…NEXT loop is prohibited.

Syntax

```
FOR variable = start TO end STEP increment
    statement; start < end
    [EXITFOR]
NEXT
```

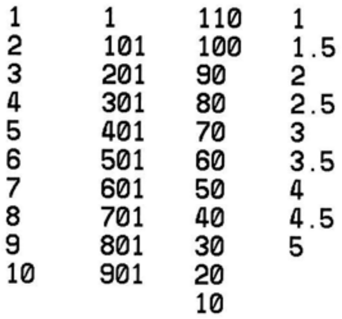
Parameter	Description
variable	The variable name (up to 8 characters)
start	Integer or floating point numbers
end	Integer of floating point numbers
increment	Integer or floating point, positive or negative
EXITFOR	Exit for loop

Example

Sample code

```
DOWNLOAD "TEST.BAS"
SIZE 4,2.5
GAP 0,0
CLS
FOR I=1 TO 10 STEP 1
TEXT 100,10+30*(I-1),"3",0,1,1,STR$(I)
NEXT

FOR I=1 TO 1000 STEP 100
TEXT 200,10+((I-1)/10)*3,"3",0,1,1,STR$(I)
NEXT

FOR I=110 TO 10 STEP -10
TEXT 300,10+(ABS(I-110))*3,"3",0,1,1,STR$(I)
NEXT

FOR I=1 TO 5 STEP 0.5
IF I-INT(I)=0 THEN Y=10+60*(I-1) ELSE Y=Y+30
TEXT 400,Y,"3",0,1,1,STR$(I)
NEXT

PRINT 1
EOP
TEST
```

Result



```
1     1     110   1
2     101   100   1.5
3     201   90    2
4     301   80    2.5
5     401   70    3
6     501   60    3.5
7     601   50    4
8     701   40    4.5
9     801   30    5
10    901   20
            10
```

See Also

DOWNLOAD, EOP

10.19 WHILE…WEND

Description

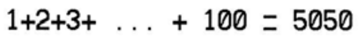
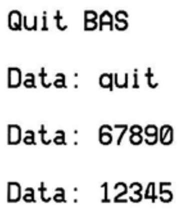
Executes a series of statements as long as a given condition is True. Nested loops are allowed (up to 39 nested loops) in this printer.

Syntax

WHILE *condition*
[*statement*]
WEND

Parameter	**Description**
condition	Available relational operator: <, >, =, <=, >=, <> ***Relational operator <>, not equal, was supported.**
Statement	One or more statements executed while condition is True.

Example

Sample Code	Result
DOWNLOAD "TEST.BAS" I=0 TOTAL=0 WHILE I<100 I=I+1 TOTAL=TOTAL+I WEND SIZE 4,0.5 GAP 0,0 CLS TEXT 10,10, "3",0,1,1, "1+2+3+ ... + 100 = " +STR$(TOTAL) PRINT 1 EOP TEST	 1+2+3+ ... + 100 = 5050
DOWNLOAD "TEST.BAS" data$ ="" SIZE 4,0.3 GAP 0,0 DIRECTION 1 INPUT "Data: ",data$ WHILE data$ <> "Quit" CLS TEXT 10,10, "3",0,1,1, "Data: "+data$ PRINT 1 INPUT "Data: ",data$ WEND CLS TEXT 10,10, "3",0,1,1, "Quit BAS" PRINT 1 EOP TEST 12345 67890 quit Quit	 Quit BAS Data: quit Data: 67890 Data: 12345

10.20 DO…LOOP

Description

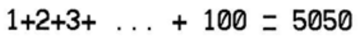
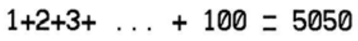
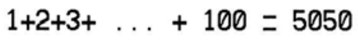
Repeats a block of statement while a condition is True.

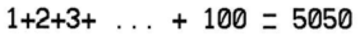
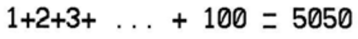
Syntax

```
DO
    [statement]
    [EXITDO]
    [statement]
LOOP

DO WHILE condition
    [statement]
    [EXITDO]
    [statement]
LOOP

DO UNTIL condition
    [statement]
    [EXITDO]
    [statement]
LOOP

DO
    [statement]
    [EXITDO]
    [statement]
LOOP WHILE condition

DO
    [statement]
    [EXITDO]
    [statement]
LOOP UNTIL condition
```

Parameter	Description
condition	Available relational operator: <, >, =, <=, >=, <>
	*Relational operator <>, not equal.
Statement	One or more statements executed while condition is True.
EXITDO	Exit loop

Example

Sample Code	Result
DOWNLOAD "TEST.BAS" I=0 TOTAL=0 DO I=I+1 TOTAL=TOTAL+I IF I=100 THEN EXITDO LOOP SIZE 4,0.5 GAP 0,0 CLS TEXT 10,10, "3",0,1,1, "1+2+3+ ... + 100 = " + STR$(TOTAL) PRINT 1 EOP TEST	 1+2+3+ ... + 100 = 5050
DOWNLOAD "TEST.BAS" I=0 TOTAL=0 DO WHILE I<=100 TOTAL=TOTAL+I I=I+1 LOOP SIZE 4,0.5 GAP 0,0 CLS TEXT 10,10, "3",0,1,1, "1+2+3+ ... + 100 = " + STR$(TOTAL) PRINT 1 EOP TEST	 1+2+3+ ... + 100 = 5050
DOWNLOAD "TEST.BAS" I=0 TOTAL=0 DO UNTIL I>100 TOTAL=TOTAL+I I=I+1 LOOP SIZE 4,0.5 GAP 0,0 CLS TEXT 10,10, "3",0,1,1, "1+2+3+ ... + 100 = " + STR$(TOTAL) PRINT 1 EOP TEST	 1+2+3+ ... + 100 = 5050

DOWNLOAD "TEST.BAS" I=0 TOTAL=0 DO TOTAL=TOTAL+I I=I+1 LOOP WHILE I<101 SIZE 4,0.5 GAP 0,0 CLS TEXT 10,10, "3",0,1,1, "1+2+3+ ... + 100 ="+ STR$(TOTAL) PRINT 1 EOP TEST	 1+2+3+ ... + 100 = 5050
DOWNLOAD "TEST.BAS" I=0 TOTAL = 0 DO TOTAL = TOTAL + I I=I+1 LOOP UNTIL I>100 SIZE 4,0.5 GAP 0,0 CLS TEXT 10,10, "3",0,1,1, "1+2+3+ ... + 100 = " + STR$(TOTAL) PRINT 1 EOP TEST	 1+2+3+ ... + 100 = 5050

10.21 IF…THEN…ELSE…ENDIF LOOP

Description

Use IF…THEN block to execute one or more statements conditionally. Either a single-line syntax or multiple-line “block” syntax can be used.

Syntax

IF *condition* THEN *statement*

Note the single-line form of IF …THEN does not use an ENDIF statement.

Or

```
IF condition THEN
    Statements
ENDIF
```

Or

```
IF condition THEN
    Statements
ELSE
    Statements
ENDIF
```

Or

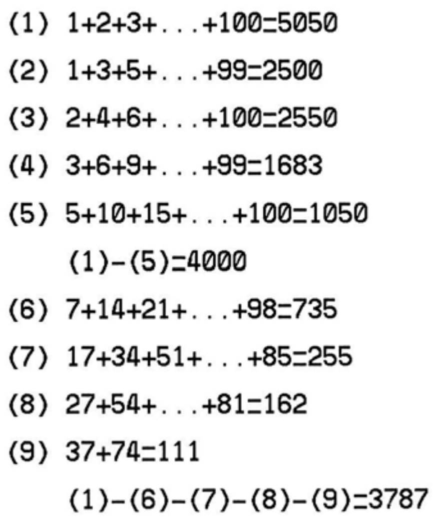
```
IF condition 1 THEN
    Statement block 1
ELSEIF condition 2 THEN
    Statement block 2
    . . .
ELSEIF condition n THEN
    Statement block n
ENDIF
```

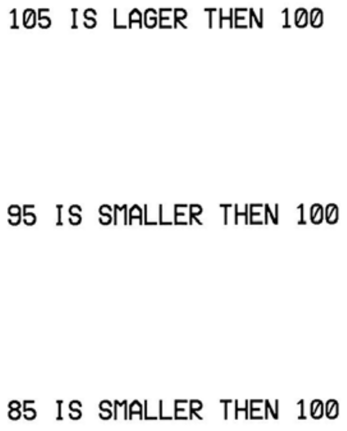
****The syntax of IF…THEN…ELSE requires that the command be typed in one single line in less than 255 characters.***

Parameter	**Description**
condition	Available relational operator: <, >, =, <=, >=, <> ****Relational operator <>, not equal.***
Statement	Only one statement is available in

Example

Sample Code	Result

DOWNLOAD "DEMO.BAS"
SIZE 4,4
GAP 0,0
DIRECTION 1
CLS
A=0
B=0
C=0
D=0
E=0
F=0
G=0
H=0
J=0
K=0
L=0
FOR I=1 TO 100
IF I-INT(I/1)*1=0 THEN A=A+I
IF I-INT(I/2)*2=1 THEN B=B+I ELSE C=C+I
IF I-INT(I/3)*3=0 THEN
D=D+I
ENDIF
IF I-INT(I/5)*5=0 THEN
E=E+I
ELSE
F=F+I
ENDIF
IF I-INT(I/7)*7=0 THEN
G=G+I
ELSEIF I-INT(I/17)*17=0 THEN
H=H+I
ELSEIF I-INT(I/27)*27=0 THEN
J=J+I
ELSEIF I-INT(I/37)*37=0 THEN
K=K+I
ELSE
L=L+I
ENDIF
NEXT
TEXT 100,110,"3",0,1,1,"(1) 1+2+3+...+100="+STR$(A)
TEXT 100,160,"3",0,1,1,"(2) 1+3+5+...+99="+STR$(B)
TEXT 100,210,"3",0,1,1,"(3) 2+4+6+...+100="+STR$(C)
TEXT 100,260,"3",0,1,1,"(4) 3+6+9+...+99="+STR$(D)
TEXT 100,310,"3",0,1,1,"(5) 5+10+15+...+100="+STR$(E)
TEXT 100,360,"3",0,1,1, "   (1)-(5)= "+STR$(F)
TEXT 100,410,"3",0,1,1,"(6) 7+14+21+...+98="+STR$(G)
TEXT 100,460,"3",0,1,1,"(7) 17+34+51+...+85=” +STR$(H)
	
	(1) 1+2+3+...+100=5050
	(2) 1+3+5+...+99=2500
	(3) 2+4+6+...+100=2550
	(4) 3+6+9+...+99=1683
	(5) 5+10+15+...+100=1050
	(1)-(5)=4000
	(6) 7+14+21+...+98=735
	(7) 17+34+51+...+85=255
	(8) 27+54+...+81=162
	(9) 37+74=111
	(1)-(6)-(7)-(8)-(9)=3787

TEXT 100,510,"3",0,1,1,"(8) 27+54+...+81="+STR$(J) TEXT 100,560,"3",0,1,1,"(9) 37+74="+STR$(K) TEXT 100,610,"3",0,1,1," (1)-(6)-(7)-(8)-(9)=”+STR$(L) PRINT 1,1 EOP	
DOWNLOAD F, "TEST.BAS" SIZE 4,1 GAP 0,0 DIRECTION 1 CLS A=85 B=10 :START IF A<100 THEN GOTO L1 ELSE GOTO L2 :L1 CLS TEXT 100,10,"3",0,1,1,STR$(A) + " IS SMALLER THEN 100" PRINT 1 A=A+B GOTO START ENDIF :L2 CLS TEXT 100,10,"3",0,1,1,STR$(A) + "IS LAGER THEN 100" PRINT 1 EOP TEST	 105 IS LAGER THEN 100 95 IS SMALLER THEN 100 85 IS SMALLER THEN 100

Note:

If the result of the expression is nonzero, the statement following THEN will be executed. If the result of the expression is zero, and the statement following the ELSE is present, it will be executed. Otherwise the next line of statement is executed.

If there are block of statements in IF…THEN …ELSE, ENDIF must be used at the end of the IF…THEN …ELSE statement.

<u>Limitations:</u>
The total numbers of nested IF …THEN …ELSE statement in a program cannot exceed 40.

The total numbers of nested IF …THEN …ELSE, FOR…NEXT, GOSUB RETURN in a program cannot exceed 40 loops.

See Also

DOWNLOAD, EOP

10.22 GOSUB…RETURN

Description

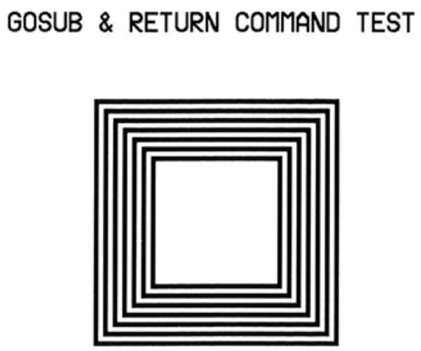
This command will branch to a subroutine, executing statements until “RETURN” is reached.

Syntax

```
GOSUB LABEL
    statement
END

:LABEL
    statement
RETURN
```

Parameter	Description
LABEL	Beginning of the subroutine. The maximum length of the label is 8 characters.

Example

Sample code	Result
DOWNLOAD "GOSUB1.BAS" SIZE 4,3 GAP 0,0 DIRECTION 1 CLS TEXT 10,10,"3",0,1,1,"GOSUB & RETURN COMMAND TEST" GOSUB DR_BOX PRINT 1 END :DR_BOX FOR I=21 TO 81 STEP 10 BOX 80+I,80+I,80+300-I,80+300-I,5 NEXT RETURN EOP GOSUB1	 GOSUB & RETURN COMMAND TEST

See Also

DOWNLOAD, EOP, END, GOTO

10.23 GOTO

Description

This command is used to branch to a specified label. The label cannot exceed 8 characters in length.

Syntax

```
GOTO LABEL

:LABEL
```

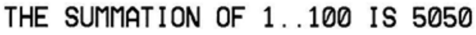
Parameter	Description
LABEL	Beginning of the point. The maximum length of the label is 8 characters.

Example

Sample code

```
DOWNLOAD "GOTO1.BAS"
SIZE 4,3
GAP 0,0
DIRECTION 1
CLS
A=0
TOTAL=0
:START
IF A<100 THEN
GOTO SUM
ELSE
GOTO PRTOUT
ENDIF
:SUM
A=A+1
TOTAL=TOTAL+A
GOTO START
:PRTOUT
B$="THE SUMMATION OF 1..100 IS "+STR$(TOTAL)
TEXT 10,100, "3",0,1,1,B$
PRINT 1
END
EOP
```

Result



```
THE SUMMATION OF 1..100 IS 5050
```

See Also

DOWNLOAD, EOP, END, GOSUB…RETURN

10.24 INP$( )

Description

One byte is received from communication port.

Syntax

INP$(n)

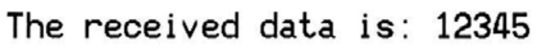
Parameter	Description
N	1 : com1 port in printer

Example

Sample code

```
DOWNLOAD "TEST.BAS"

T$=""
FOR I=1 TO 5
T$=T$+INP$(1)
NEXT

SIZE 4,0.5
GAP 0,0
CLS
TEXT 10,10, "3",0,1,1, "The received data is: "+T$
PRINT 1
EOP
TEST
12345
```

Result



```
The received data is: 12345
```

See Also

INP()

10.25 INP( )

Description

One byte (ASCII value) is received from communication port.

Syntax

INP(n)

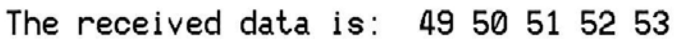
Parameter	Description
n	1 : com1 port in printer

Example

Sample code

```
DOWNLOAD "TEST.BAS"

147sci=0
str$=""

FOR I=1 TO 5
147sci=INP(1)
str$=str$+" " +STR$(147sci)
OUT  147sci
NEXT

SIZE 4,0.5
GAP 0,0
CLS
TEXT 10,10, "3",0,1,1, "The received data is: "+str$
PRINT 1
EOP
TEST
12345
```

Result



```
The received data is:  49 50 51 52 53
```

See Also

INP$()

10.26 LOB( )

Description

This function returns the size of data in receiving buffer.

Syntax

LOB ()

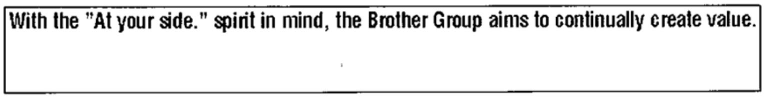
Example

Sample Code

```
DOWNLOAD "TEST.BAS"

DATA$=""

WHILE LOB()<>0
DATA$=DATA$+INP$(1)
WEND

SIZE 4,0.5
GAP 0,0
CLS
BOX 10,10,800,100,2
BLOCK 15,15,790,90, "0",0,8,8,DATA$
PRINT 1
EOP
TEST
With the "At your side." spirit in mind. the Brother Group aims to continually create value.
```

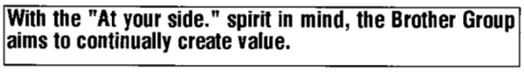
Result

203 dpi:



With the "At your side." spirit in mind, the Brother Group aims to continually create value.

300 dpi:



With the "At your side." spirit in mind, the Brother Group
aims to continually create value.

See Also

INP$(), WHILE … WEND

10.27 INPUT

Description

This command receives data sent externally.

Syntax

INPUT ["Prompt string", number of digits], variables

The comma also can be replaced by semicolon, such as:

INPUT ["Prompt string"; number of digits]; variables

Parameter	Description
Prompt string	The maximum length of prompt string is 20 characters
Number of digits	Maximum number of characters is 255
Variables	The variable to receive input data

Example

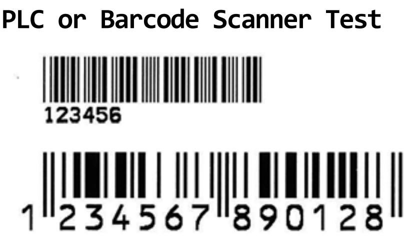
Sample code as template

```
DOWNLOAD F,"TEXT.BAS"
SIZE 4,3
GAP 0,0
DIRECTION 1

:START
INPUT "CODE 39 : ",C39$
INPUT "EAN 13: ",12,E13$
CLS
TEXT 20,50, "3",0,1,1, PLC or Barcode Scanner Test
BARCODE 20,100, "39",48,1,0,2,5,C39$
BARCODE 20,200, "EAN13",48,1,0,4,4,E13$
PRINT 1
GOTO START
EOP
```

Sample code as PLC or Barcode Scanner

```
TEXT
123456
123456789012
```

Result



PLC or Barcode Scanner Test

See Also

DOWNLOAD, EOP, END, GOTO

10.28 PREINPUT

Description

This command can define the start character for command INPUT.

Syntax

```
PREINPUT var$
PREINPUT CHR$(n)
```

Parameter	Description
var$	The specific character or string in front of data.
N	n = 1 ~ 255

Example

```
PREINPUT "<"
PREINPUT CHR$(2)
```

See also

POSTINPUT, INPUT, SET FILTER

10.29 POSTINPUT

Description

This command can define the end character for command INPUT.

Syntax

```
POSTINPUT var$
POSTINPUT CHR$(n)
```

Parameter	Description
var$	The specific character or string in end of data.
N	n = 1 ~ 255

Example

```
POSTINPUT ">"
POSTINPUT CHR$(3)
```

See also

PREINPUT, INPUT, SET FILTER

10.30 SET FILTER ON/OFF

Description

This command is using to enable/disable commands PREINPUT and POSTINPUT.

Syntax

```
SET FILTER ON/OFF
```

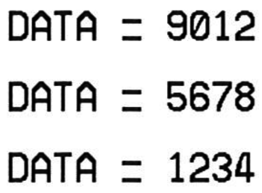
Parameter	Description
ON	Enable PREINPUT and POSTINPUT
OFF	Disable PREINPUT and POSTINPUT

Example

Sample Code

```
DOWNLOAD "TEST.BAS"

PREINPUT "<="
POSTINPUT "=>"
SET FILTER ON

START:
INPUT "DATA",data1$
SIZE 4,0.25
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "DATA = "+data1$
PRINT 1
GOTO START
EOP
TEST
<=1234=><=5678=><=9012=>
```

Result



```
DATA = 9012

DATA = 5678

DATA = 1234
```

See also

PREINPUT, POSTINPUT, INPUT

10.31 REM

Description

Comment. Prefix is "REM", which will be ignored by the printer.

Syntax

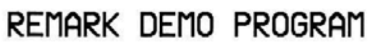
REM

Example

Sample code

```
REM *********************************
REM This is a demonstration program*
REM *********************************
DOWNLOAD “REMARK.BAS”
SIZE 4,3
GAP 0,0
DIRECTION 1
CLS
TEXT 50,50, "3",0,1,1, "REMARK DEMO PROGRAM"
REM TEXT 50,100, "3",0,1,1, "REMARK DEMO PROGRAM"
PRINT 1,1
EOP
REMARK
```

Result



REMARK DEMO PROGRAM

See Also

DOWNLOAD, EOP, END

10.32 OUT

Description

This command returns data through the specific port.

Syntax

OUT [port] "prompt",variable
OUT [port] "prompt";variable

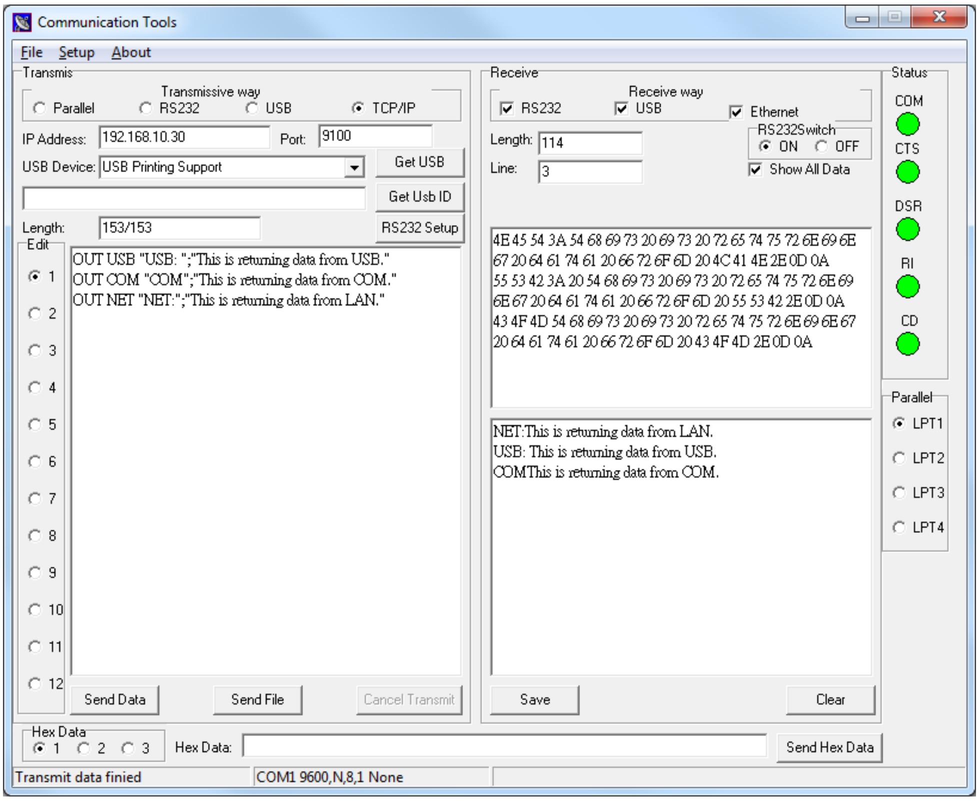
Parameter	Description
port	Optional. Specified the port for returning data/string. Default is returning the data/string from the port which is sending data to printer. **COM**: Returning data/string from COM port. **USB**: Returning data/string from USB port. **NET**: Returning data/string from LAN port.
Prompt	Prompt string.
Variable	The output message.
,	The "*prompt*" and "*variable*" are separated by <0x0D><0x0A>.
;	The "*variable*" comes behind "*prompt*" directly.

Example

Sample Code

```
OUT USB "USB: ";"This is returning data from USB. "
OUT COM "COM"; "This is returning data from COM. "
OUT NET "NET: ";"This is returning data from LAN. "
```

Result



Communication Tools

Transmissive way: ○ Parallel ○ RS232 ○ USB ● TCP/IP

IP Address: 192.168.10.30 Port: 9100

USB Device: USB Printing Support

Length: 153/153

Edit: ● 1

```
OUT USB "USB: ";"This is returning data from USB."
OUT COM "COM";"This is returning data from COM."
OUT NET "NET:";"This is returning data from LAN."
```

Receive way: ☑ RS232 ☑ USB ☑ Ethernet

RS232Switch: ● ON ○ OFF ☑ Show All Data

Length: 114

Line: 3

```
4E 45 54 3A 54 68 69 73 20 69 73 20 72 65 74 75 72 6E 69 6E
67 20 64 61 74 61 20 66 72 6F 6D 20 4C 41 4E 2E 0D 0A
55 53 42 3A 20 54 68 69 73 20 69 73 20 72 65 74 75 72 6E 69
6E 67 20 64 61 74 61 20 66 72 6F 6D 20 55 53 42 2E 0D 0A
43 4F 4D 54 68 69 73 20 69 73 20 72 65 74 75 72 6E 69 6E 67
20 64 61 74 61 20 66 72 6F 6D 20 43 4F 4D 2E 0D 0A
```

```
NET:This is returning data from LAN.
USB: This is returning data from USB.
COMThis is returning data from COM.
```

Parallel: ● LPT1 ○ LPT2 ○ LPT3 ○ LPT4

Hex Data: ● 1 ○ 2 ○ 3

Transmit data finied COM1 9600,N,8,1 None

10.33 OUTR

Description

This command sends data through RS-232 port only.

Syntax

OUTR "prompt",variable
OUTR "prompt";variable

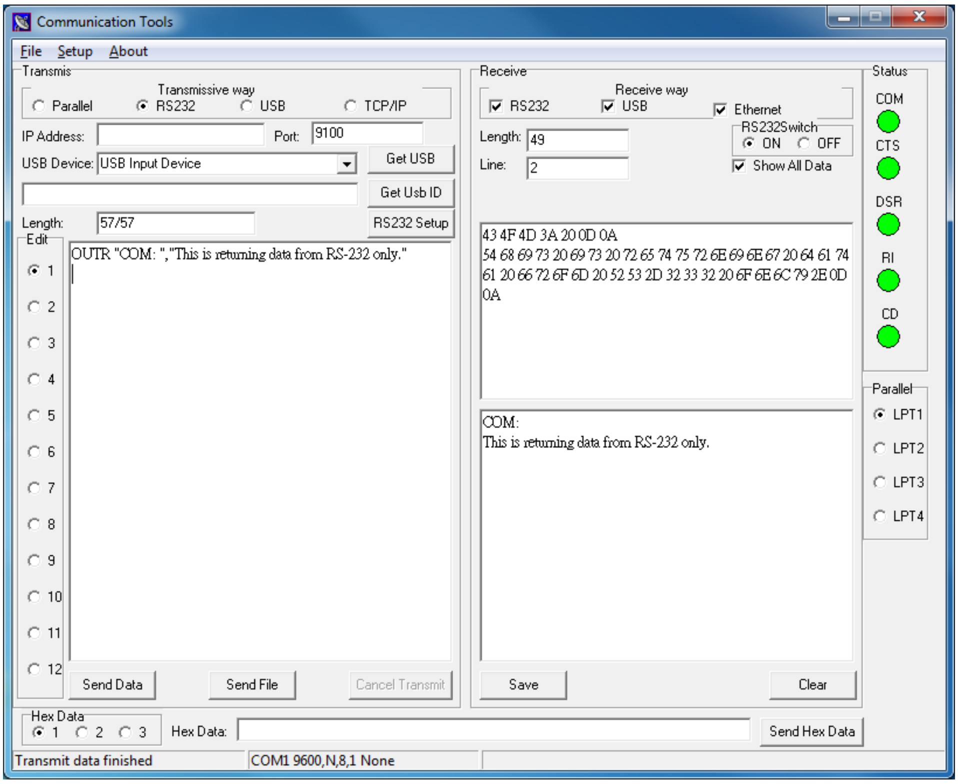
Parameter	Description
prompt	Prompt string.
Variable	The output message.
,	The "*prompt*" and "*variable*" are separated by <0x0D><0x0A>.
;	The "*variable*" comes behinds "*prompt*" directly.

Example

Sample Code

```
OUTR "COM: "," This is returning data from RS-232 only."
```

Result



Transmissive way: ○ Parallel ● RS232 ○ USB ○ TCP/IP

IP Address: Port: 9100

USB Device: USB Input Device

Length: 57/57

Edit: ● 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 ○ 8 ○ 9 ○ 10 ○ 11 ○ 12

```
OUTR "COM: ","This is returning data from RS-232 only."
```

Receive way: ☑ RS232 ☑ USB ☑ Ethernet

RS232Switch: ● ON ○ OFF

Length: 49

Line: 2

☑ Show All Data

```
43 4F 4D 3A 20 0D 0A
54 68 69 73 20 69 73 20 72 65 74 75 72 6E 69 6E 67 20 64 61 74
61 20 66 72 6F 6D 20 52 53 2D 32 33 32 20 6F 6E 6C 79 2E 0D
0A
```

```
COM:
This is returning data from RS-232 only.
```

Parallel: ● LPT1 ○ LPT2 ○ LPT3 ○ LPT4

Hex Data: ● 1 ○ 2 ○ 3 Hex Data:

Transmit data finished | COM1 9600,N,8,1 None

10.34 GETKEY( )

Description

This command is used to get the status of the PAUSE and FEED keys. This command waits until either key is pressed, whereupon 0 is returned if PAUSE key is pressed and 1 is returned if FEED key is pressed.

Syntax

GETKEY()

PAUSE	FEED
0	1

Example

Sample code

```
DOWNLOAD "DEMO4.BAS"
SIZE 4,3
GAP 0,0
CLS
:START
A=GETKEY()
IF A=0 THEN GOTO PAUSEB
IF A=1 THEN GOTO FEEDB
:PAUSEB
CLS
TEXT 50,10, "4",0,1,1, "PAUSE key is pressed !"
PRINT 1
GOTO START
:FEEDB
CLS
TEXT 50,10, "4",0,1,1, "FEED key is pressed !"
PRINT 1
EOP
```

See Also

DOWNLOAD, EOP, END, GOTO

10.35 INT( )

Description

This function truncates a floating point number.

Syntax

INT (n)

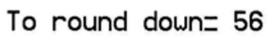
Parameter	Description
n	Positive or negative integer, floating point number or mathematical expression

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
INPUT "Number: ",Num
CLS
REM **** To round up or down****
N=INT(Num+0.5)
IF N>Num THEN
TEXT 50,100, "3",0,1,1, "To round up= " +STR$(N)
ELSE
TEXT 50,100, "3",0,1,1, "To round down= " +STR$(N)
ENDIF
PRINT 1
EOP
56.2
```

Result



```
To round down= 56
```

See Also

DOWNLOAD, EOP, END, ABS(), ASC(), STR$()

10.36 LEFT$( )

Description

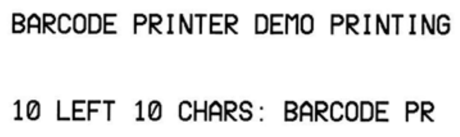
This function returns the specified number of characters down from the initial character of a string.

Syntax

LEFT$ (X$, n)

Parameter	Description
X$	The string to be processed
n	The number of characters to be returned

Example

Sample code	Result
DOWNLOAD "TEST.BAS" **SIZE 4,1** **GAP 0,0** **DIRECTION 1** **A$="BARCODE PRINTER DEMO PRINTING"** **C$=LEFT$(A$,10)** **CLS** **TEXT 10,10,"3",0,1,1,A$** **TEXT 10,100,"3",0,1,1, "10 LEFT 10 CHARS: " +C$** **PRINT 1** **EOP** **TEST**	 BARCODE PRINTER DEMO PRINTING 10 LEFT 10 CHARS: BARCODE PR

See Also

DOWNLOAD, EOP, END, RIGHT$(), MID$(), LEN(), STR$()

10.37 LEN( )

Description

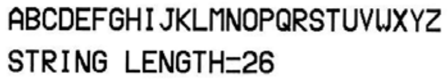
This function returns the length of a string.

Syntax

LEN (string)

Parameter	Description
string	The string whose length is to be measured.

Example

Sample Code	Result
DOWNLOAD "DEMO.BAS" SIZE 4,1 GAP 0,0 DIRECTION 1 A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ" B=LEN(A$) CLS TEXT 10,10, "3",0,1,1,A$ TEXT 10,50, "3",0,1,1,"STRING LENGTH=" +STR$(B) PRINT 1 EOP DEMO	 ABCDEFGHIJKLMNOPQRSTUVWXYZ STRING LENGTH=26

See Also

DOWNLOAD, EOP, END, LEFT$(), LEN(), RIGHT$(), MID$(), STR$(), VAL()

10.38 MID$( )

Description

This function retrieves the specified number of characters down from the *m*th character of a string.

Syntax

MID$(string,m,n)

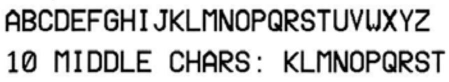
Parameter	Description
string	The string to be processed
m	The beginning of mth characters in the string 1 <= m <= string length
n	The number of characters to return

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
E$=MID$(A$,11,10)
CLS
TEXT 10,10, "3",0,1,1,A$
TEXT 10,200, "3",0,1,1,"10 MIDDLE CHARS: "+E$
PRINT 1
EOP
DEMO
```

Result



```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
10 MIDDLE CHARS: KLMNOPQRST
```

See Also

DOWNLOAD, EOP, END, LEFT$(), LEN(), RIGHT$(), STR$(), VAL()

10.39 RIGHT$( )

Description

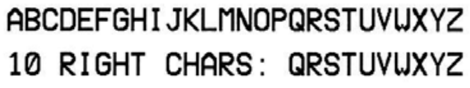
This function returns a specified number of characters up from the end of a string.

Syntax

RIGHT$ (X$,n)

Parameter	Description
X$	The string to be processed
n	The number of characters to be returned from the right side (end) of the string

Example

Sample code	Result
DOWNLOAD "DEMO.BAS" **SIZE 4,1** **GAP 0,0** **DIRECTION 1** **A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"** **D$=RIGHT$(A$,10)** **CLS** **TEXT 10,10,"3",0,1,1,A$** **TEXT 10,150,"3",0,1,1, "10 RIGHT CHARS: "+D$** **PRINT 1** **EOP** **DEMO**	 ABCDEFGHIJKLMNOPQRSTUVWXYZ 10 RIGHT CHARS: QRSTUVWXYZ

See Also

DOWNLOAD, EOP, END, LEFT$(), LEN(), MID$(), STR$(), VAL()

10.40 STR$( )

Description

This function converts a specified value or expression into corresponding string of characters.

Syntax

STR$ (n)

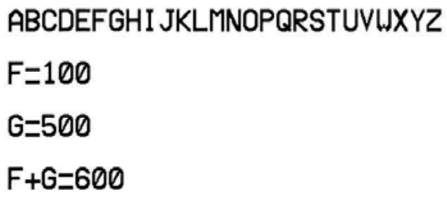
Parameter	Description
n	An integer, floating point number or mathematical expression

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
F=100
G=500
H$=STR$(F+G)
CLS
TEXT 10,10, "3",0,1,1,A$
TEXT 10,60, "3",0,1,1, "F=" +STR$(F)
TEXT 10,110, "3",0,1,1, "G=" +STR$(G)
TEXT 10,160, "3",0,1,1, "F+G=" +H$
PRINT 1
EOP
DEMO
```

Result



```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
F=100
G=500
F+G=600
```

See Also

DOWNLOAD, EOP, END, LEFT$(), LEN(), RIGHT$(), MID$(), VAL()

10.41 STRCOMP( )

Description

Returns -1, 0, or 1, based on the result of a string comparison.

Syntax

STRCOMP(str1$,str2$[,comp])

Parameter	Description
str1$	Required. Any valid string expression.
Str2$	Required. Any valid string expression.
Comp	Optional. Specifies the type of string comparison. 0: Binary comparison. Default. 1: Textual comparison. The comparison is **case-insensitive**.

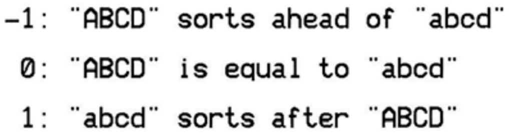
Condition	Return value
str1$ sorts ahead of str2$	-1
str1$ is equal to str2$	0
str1$ sorts after str2$	1

Example

Sample Code

```
DOWNLOAD "TEST.BAS"
STR1$ = "ABCD"
STR2$ = "abcd"

result1 = STRCOMP(STR1$,STR2$)
result2 = STRCOMP(STR1$,STR2$,1)
result3 = STRCOMP(STR2$,STR1$)

SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 100,10,"3",0,1,1,STR$(result1)+": ¥["]" +STR1$+"¥["] sorts ahead of ¥["]" +STR2$+"¥["]"
TEXT 100,60,"3",0,1,1," " +STR$(result2)+": ¥["]"+STR1$+"¥["] is equal to ¥["]"+STR2$+"¥["]"
TEXT 100,110,"3",0,1,1," "+STR$(result3)+": ¥["]"+STR2$+"¥["] sorts after ¥["]"+STR1$+"¥["]"
PRINT 1
EOP
TEST
```

Result



```
-1: "ABCD" sorts ahead of "abcd"
 0: "ABCD" is equal to "abcd"
 1: "abcd" sorts after "ABCD"
```

See Also

INSTR()

10.42 INSTR ( )

Description

Returns an integer specifying the start position of the first occurrence of one string within another.

Syntax

INSTR ([start,]str1$,str2$)

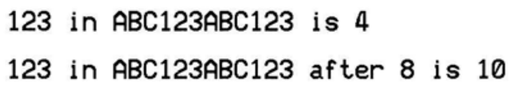
Parameter	Description
start	Optional. Numeric expression that sets the starting position for each search. If omitted, search begins at the first character position. The stat index is 1 – based.
Str1$	Required. String expression being searched.
Str2$	Required. String expression sought.

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
string$="ABC123ABC123"
searchfor$="123"
starpos=8

temp1=INSTR(string$,searchfor$)
temp2=INSTR(starpos,string$,searchfor$)

str1$=searchfor$+"in "+string$+"is "+STR$(temp1)
str2$=searchfor$+"in "+string$+"after"+STR$(starpos)+ " is "+STR$(temp2)

SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1,str1$
TEXT 10,60, "3",0,1,1,str2$
PRINT 1
EOP
DEMO
```

Result



```
123 in ABC123ABC123 is 4
123 in ABC123ABC123 after 8 is 10
```

See Also

STRCOMP()

10.43 TRIM$( )

Description

Removes both leading and trailing blank spaces or specific characters from a string.

Syntax

TRIM$ (str$[,list$])

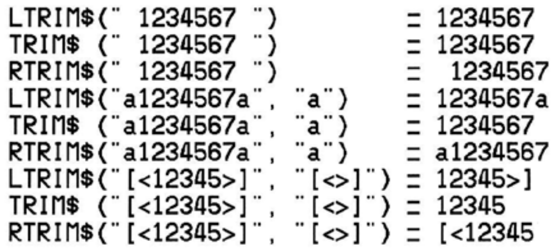
Parameter	Description
str$	The string that will be trimmed.
List$	Optional. The specific characters in list$ will be removed.

Example

Sample Code

```
DOWNLOAD "DEMO.BAS"
data1$="1234567"
data2$="a1234567a"
data3$="[<12345>]"

SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS
TEXT 50,020,"3",0,1,1,"LTRIM$(¥["]" +data1$+" ¥["])              = " +LTRIM$(data1$)
TEXT 50,050,"3",0,1,1,"TRIM$ (¥["]" +data1$+"¥["])               = " +TRIM$(data1$)
TEXT 50,080,"3",0,1,1,"RTRIM$(¥["]" +data1$+"¥["])               = " +RTRIM$(data1$)
TEXT 50,110,"3",0,1,1,"LTRIM$(¥["]" +data2$+ "¥["], ¥["]a¥["])      = " +LTRIM$(data2$,"a")
TEXT 50,140,"3",0,1,1,"TRIM$ (¥["]" +data2$+ "¥["], ¥["]a¥["])      = " +TRIM$(data2$,"a")
TEXT 50,170,"3",0,1,1,"RTRIM$(¥["]" +data2$+ "¥["], ¥["]a¥["])      = " +RTRIM$(data2$,"a")
TEXT 50,200,"3",0,1,1,"LTRIM$(¥["]" +data3$+ "¥["], ¥["][<>]¥["]) = " +LTRIM$(data3$,"[<>]")
TEXT 50,230,"3",0,1,1,"TRIM$ (¥["]" +data3$+ "¥["], ¥["][<>]¥["]) = " +TRIM$(data3$,"[<>]")
TEXT 50,260,"3",0,1,1,"RTRIM$(¥["]" +data3$+ " ¥["], ¥["][<>]¥["]) = " +RTRIM$(data3$,"[<>]")
PRINT 1
EOP
DEMO
```

Result



```
LTRIM$(" 1234567 ")            = 1234567
TRIM$ (" 1234567 ")            = 1234567
RTRIM$(" 1234567 ")            =  1234567
LTRIM$("a1234567a", "a")       = 1234567a
TRIM$ ("a1234567a", "a")       = 1234567
RTRIM$("a1234567a", "a")       = a1234567
LTRIM$("[<12345>]", "[<>]")    = 12345>]
TRIM$ ("[<12345>]", "[<>]")    = 12345
RTRIM$("[<12345>]", "[<>]")    = [<12345
```

See Also

LTRIM$(), RTRIM$()

10.44 LTRIM$( )

Description

Removes leading blank space from a string.

Syntax

LTRIM$ (str$[,list$])

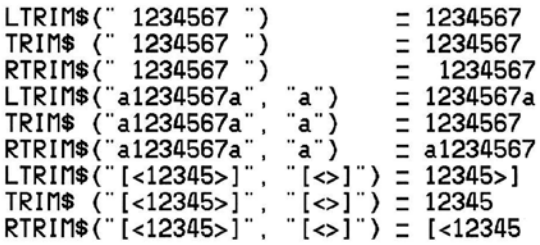
Parameter	Description
str$	The string that will be trimmed.
List$	Optional. The specific characters in list$ will be removed.

Example

Sample Code

```
DOWNLOAD "DEMO.BAS"
data1$="1234567"
data2$="a1234567a"
data3$="[<12345>]"

SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS
TEXT 50,020,"3",0,1,1,"LTRIM$(¥["]" +data1$+" ¥["])               =" +LTRIM$(data1$)
TEXT 50,050,"3",0,1,1,"TRIM$ (¥["]" +data1$+" ¥["])               =" +TRIM$(data1$)
TEXT 50,080,"3",0,1,1,"RTRIM$(¥["]" +data1$+" ¥["])               =" +RTRIM$(data1$)
TEXT 50,110,"3",0,1,1,"LTRIM$(¥["]" +data2$+" ¥["],¥["]a¥["])     =" +LTRIM$(data2$,"a")
TEXT 50,140,"3",0,1,1,"TRIM$ (¥["]" +data2$+" ¥["],¥["]a¥["])     =" +TRIM$(data2$,"a")
TEXT 50,170,"3",0,1,1,"RTRIM$(¥["]" +data2$+" ¥["],¥["]a¥["])     =" +RTRIM$(data2$,"a")
TEXT 50,200,"3",0,1,1,"LTRIM$(¥["]" +data3$+" ¥["],¥["][<>]¥["])  =" +LTRIM$(data3$,"[<>]")
TEXT 50,230,"3",0,1,1,"TRIM$ (¥["] " +data3$+" ¥["],¥["][<>]¥["]) =" +TRIM$(data3$,"[<>]")
TEXT 50,260,"3",0,1,1,"RTRIM$(¥["]" +data3$+" ¥["],¥["][<>]¥["])  =" +RTRIM$(data3$,"[<>]")
PRINT 1
EOP
DEMO
```

Result



```
LTRIM$(" 1234567 ")         = 1234567
TRIM$ (" 1234567 ")         = 1234567
RTRIM$(" 1234567 ")         =  1234567
LTRIM$("a1234567a", "a")    = 1234567a
TRIM$ ("a1234567a", "a")    = 1234567
RTRIM$("a1234567a", "a")    = a1234567
LTRIM$("[<12345>]", "[<>]") = 12345>]
TRIM$ ("[<12345>]", "[<>]") = 12345
RTRIM$("[<12345>]", "[<>]") = [<12345
```

See Also

TRIM$(), RTRIM$()

10.45 RTRIM$( )

Description

Removes trailing blank space from a string.

Syntax

RTRIM$ (str$ [, list$])

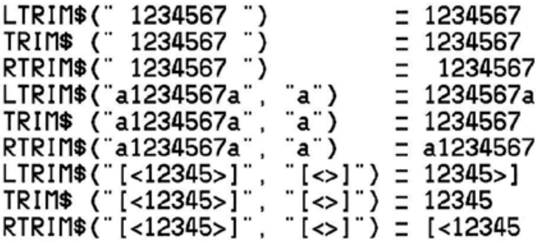
Parameter	Description
str$	The string that will be trimmed.
List$	Optional. The specific characters in list$ will be removed.

Example

Sample Code

```
DOWNLOAD "DEMO.BAS"
data1$="1234567"
data2$="a1234567a"
data3$="[<12345>]"

SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS
TEXT 50,020,"3",0,1,1, "LTRIM$(¥["]" +data1$+" ¥["])              = " +LTRIM$(data1$)
TEXT 50,050,"3",0,1,1, "TRIM$ (¥["]" +data1$+" ¥["])              = " +TRIM$(data1$)
TEXT 50,080,"3",0,1,1, "RTRIM$(¥["]" +data1$+" ¥["])              = " +RTRIM$(data1$)
TEXT 50,110,"3",0,1,1, "LTRIM$(¥["]" +data2$+" ¥["], ¥["]a¥["])      = " +LTRIM$(data2$,"a")
TEXT 50,140,"3",0,1,1, "TRIM$ (¥["]" +data2$+" ¥["], ¥["]a¥["])      = " +TRIM$(data2$,"a")
TEXT 50,170,"3",0,1,1, "RTRIM$(¥["]" +data2$+" ¥["], ¥["]a¥["])      = " +RTRIM$(data2$,"a")
TEXT 50,200,"3",0,1,1, "LTRIM$(¥["]" +data3$+" ¥["], ¥["][<>]¥["]) = " +LTRIM$(data3$,"[<>]")
TEXT 50,230,"3",0,1,1, "TRIM$ (¥["]" +data3$+" ¥["], ¥["][<>]¥["]) = " +TRIM$(data3$,"[<>]")
TEXT 50,260,"3",0,1,1, "RTRIM$(¥["]" +data3$+" ¥["], ¥["][<>]¥["]) = " +RTRIM$(data3$,"[<>]")
PRINT 1
EOP
DEMO
```

Result



```
LTRIM$(" 1234567 ")           = 1234567
TRIM$ (" 1234567 ")           = 1234567
RTRIM$(" 1234567 ")           =  1234567
LTRIM$("a1234567a", "a")      = 1234567a
TRIM$ ("a1234567a", "a")      = 1234567
RTRIM$("a1234567a", "a")      = a1234567
LTRIM$("[<12345>]", "[<>]")   = 12345>]
TRIM$ ("[<12345>]", "[<>]")   = 12345
RTRIM$("[<12345>]", "[<>]")   = [<12345
```

See Also

TRIM$(), LTRIM$()

10.46 TEXTPIXEL( )

Description

Returns the width of the text string in dot.

Syntax

TEXTPIXEL (cont$,font$,size)

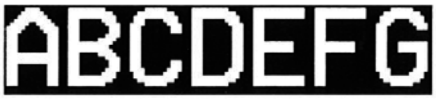
Parameter	Description
cont$	The content of text string.
Font $	The font type. Please refer to the parameter ***font*** in command TEXT.
Size	The font size. Please refer to the parameter x-multiplication in command TEXT.

Example

Sample code

```
DOWNLOAD "TEST.BAS"

str$="ABCDEFG"
font$="3"
fontsize=3
strwidth=TEXTPIXEL(str$,font$,fontsize)

SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10,font$,0,fontsize,fontsize,str$
REVERSE 8,8,strwidth,72
PRINT 1
EOP
TEST
```

Result



ABCDEFG

See Also

TEXT, BARCODEPIXEL()

10.47 BARCODEPIXEL( )

Description

Returns the width of barcode in dot.

Syntax

BARCODEPIXEL (cont$, sym$, narrow, wide)

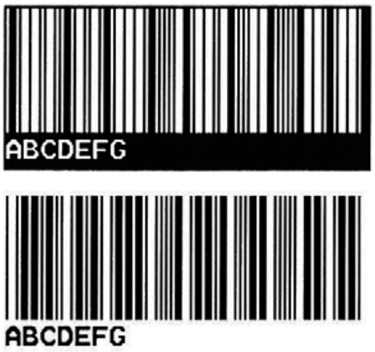
Parameter	Description
cont$	The content of barcode.
Sym $	Barcode type. Please refer to the parameter ***code type*** in command BARCODE.
Narrow	The width of narrow bar. Please refer to the parameter ***narrow*** in command BARCODE.
Wide	The width of wide bar. Please refer to the parameter ***wide*** in command BARCODE.

Example

Sample code

```
DOWNLOAD "TEST.BAS"

cont$="ABCDEFG"
sym$="39"
narrow=2
wide=6
codewidth=BARCODEPIXEL(cont$,sym$,narrow,wide)

SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS
BARCODE 10,10,sym$,100,1,0,narrow,wide,cont$
REVERSE 8,8,codewidth+8,132
BARCODE 10,160,sym$,100,1,0,narrow,wide,cont$
PRINT 1
EOP
TEST
```

Result



See Also

BARCODE, TEXTPIXEL()

10.48 VAL( )

Description

This function converts numeric characters into corresponding integer or floating point number.

Syntax

VAL ("numeric character")

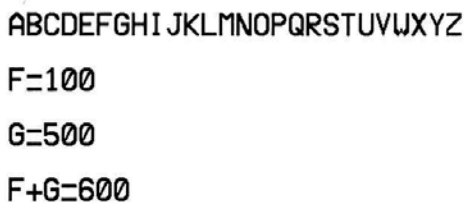
Parameter	Description
numeric character	" 0~9","."

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
F$="100"
G$="500"
CLS
H=VAL(F$)+VAL(G$)
I$=STR$(H)
TEXT 10,10, "3",0,1,1,A$
TEXT 10,60, "3",0,1,1, "F=" +F$
TEXT 10,110, "3",0,1,1, "G=" +G$
TEXT 10,160, "3",0,1,1, "F+G=" +I$
PRINT 1
EOP
DEMO
```

Result



```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
F=100
G=500
F+G=600
```

See Also

DOWNLOAD, EOP, END, LEFT$(), LEN(), RIGHT$(), MID$(), STR$()

10.49 NOW$( )

Description

Returns the current date and time according to the setting of your printer. The returned value always uses with commands FORMAT$().

Syntax

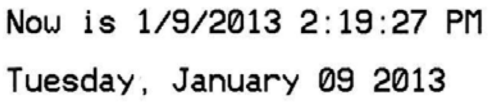
NOW$( )

Example

Sample code

```
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "Now is " +NOW$( )
TEXT 10,60, "3",0,1,1,FORMAT$(NOW$(),"Long Date")
PRINT 1
```

Result



```
Now is 1/9/2013 2:19:27 PM
Tuesday, January 09 2013
```

See Also

FORMAT$()

10.50 NOW

Description

Returns the total days since A.D. 1900. This global variable always uses with commands FORMAT$() and DATEADD().

Syntax

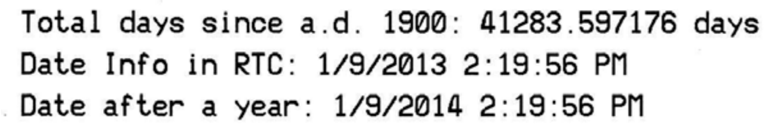
```
NOW
```

Example

Sample Code

```
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "Total days since a.d. 1900: " +STR$(NOW)+ " days"
TEXT 10,50, "3",0,1,1, "Date Info in RTC: " +FORMAT$(NOW, "General Date")
TEXT 10,90, "3",0,1,1, "Date after a year: " +FORMAT$(DATEADD("yyyy",1,NOW), "General Date")
PRINT 1
```

Result



```
Total days since a.d. 1900: 41283.597176 days
Date Info in RTC: 1/9/2013 2:19:56 PM
Date after a year: 1/9/2014 2:19:56 PM
```

See Also

FORMAT$(), DATEADD(), NOW

10.51 FORMAT$( )

Description

Returns the current date, time, number and number value according to the setting of your printer.

Syntax

FORMAT$(expression[,style$])

Parameter	**Description**
expression	Required. Any valid expression.
Style$	Optional. A valid named or user-defined format string expression.

Predefined date/time formats	**Description**
General Date	Shows date and time.
Long Date	Uses the Long Date format.
Medium Date	Uses the dd-mmm-yy format.
Short Date	Uses the Short Date format.
Long Time	Shows the hour, minute, second, and “AM” or “PM” using the h:mm:ss format.
Medium Time	Shows the hour, minute, and “AM” or “PM” using the “hh:mm AM/PM” format.
Short Time	Shows the hour and minute using the hh:mm format.

User-defined date/time formats	**Description**
c	Display the date as ddddd and display the time as ttttt, in that order.
d	Display the day as a number without a leading zero (1 – 31).
dd	Display the day as a number with a leading zero (01 – 31).
ddd	Display the day as an abbreviation (Sun – Sat).
dddd	Display the day as a full name (Sunday – Saturday).
ddddd	Display a date serial number as a complete date (including day, month, and year), formatted according to your system’s short date format setting. The default short date format is m/d/yyyy.
dddddd	Display the date as a complete date (including day, month, and year), formatted according to the long date setting recognized by your system. The default long date format is dddd, mmmm dd, yyyy.
w	Display the day of the week as a number (1 for Sunday through 7 for Saturday).
ww	Display the week of the year as a number (1 – 53).
m	Display the month as a number without a leading zero (1 – 12). If m immediately follows h or hh, the minute rather than the month is displayed.
mm	Display the month as a number with a leading zero (01 – 12). If mm immediately follows h or hh, the minute rather than the month is displayed.
mmm	Display the month as an abbreviation (Jan – Dec).
mmmm	Display the month as a full month name (January – December).
q	Display the quarter of the year as a number (1 – 4).
y	Display the day of the year as a number (1 – 366).
yy	Display the year as a 2-digit number (00 – 99).
yyyy	Display the year as a 4-digit number (100 – 9999).
h	Display the hour as a number without leading zeros (0 – 23).
hh	Display the hour as a number with leading zeros (00 – 23).
n	Display the minute as a number without leading zeros (0 – 59).
nn	Display the minute as a number with leading zeros (00 – 59).
s	Display the second as a number without leading zeros (0 – 59).
ss	Display the second as a number with leading zeros (00 – 59).

ttttt	Display a time as a complete time (including hour, minute, and second). The default time format is h:mm:ss AM/PM.
AM/PM	Display an uppercase AM with any hour before noon; display an uppercase PM with any hour between noon and 11:59 P.M.
am/pm	Display a lowercase AM with any hour before noon; display a lowercase PM with any hour between noon and 11:59 P.M.
A/P	Display an uppercase A with any hour before noon; display an uppercase P with any hour between noon and 11:59 P.M.
a/p	Display a lowercase A with any hour before noon; display a lowercase P with any hour between noon and 11:59 P.M.
AMPM	AMPM can be either uppercase or lowercase, but the case of the string displayed matches the string as defined by your system settings.
\	Display the next character in the format string.
“string”	Display the string inside the double quotation marks.

Number formats	Description
General Number	Displays the number as entered, with no rounding and no commas.
Currency	Displays the number with a dollar sign, comma (if appropriate), and two digits to the right of the decimal point. Shows negative numbers inside parentheses.
Fixed	Displays the number with at least one digit to the left of the decimal separator and two digits to the right. Does not show comma.
Standard	Displays the number with at least one digit to the left of the decimal separator and two digits to the right and commas (if appropriate).
Percent	Multiplies the value by 100 and displays the result with two digits to the right of the decimal point and a percent sign at the end.
Scientific	Uses standard scientific notation.
Yes/No	Any nonzero numeric value is Yes. Zero is No.
True/False	Any nonzero numeric value is True. Zero is False.
On/Off	Any nonzero numeric value is On. Zero is Off.

User-defined number formats	Description
0	Digit placeholder. Displays a digit or a zero.
#	Digit placeholder. Displays a digit or nothing.
.	Decimal placeholder.
%	Percent placeholder. Multiplies the expression by 100.
,	Thousand separator.
E- E+ e- e+	Scientific format.
¥	Display the next character in the format string.
"ABC"	Display the string inside the double quotation marks.

Different formats for different number values	Description
One section only	The format expression applies to all values.
Two section	The first section applies to positive values and zeros; the second applies to negative values.
Three sections	The first section applies to positive values, the second applies to negative values, and the third applies to zeros.

See Also

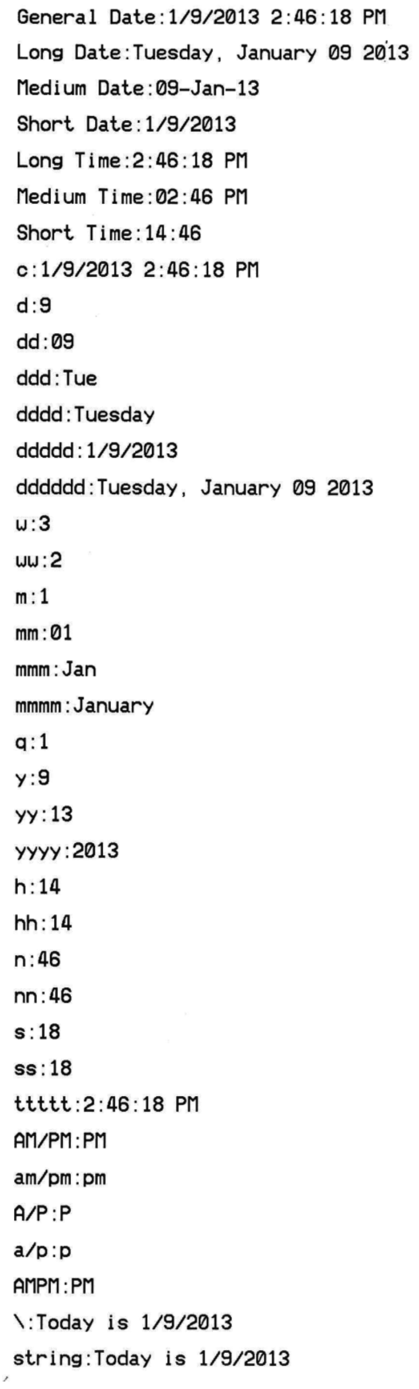
NOW$(), DATEADD(), NOW

Example

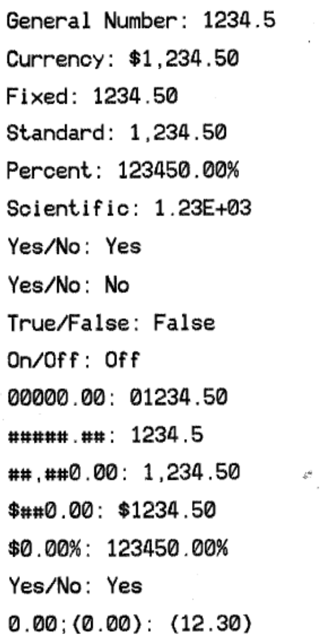
Sample Code

```
SIZE 800 dot,1900 dot
GAP 0,0
DIRECTION 1
CLS
TEXT 15,10, "3",0,1,1, "General Date: "+FORMAT$(NOW,"General Date")
TEXT 15,60, "3",0,1,1, "Long Date: "+FORMAT$(NOW,"Long Date")
TEXT 15,110, "3",0,1,1, "Medium Date: "+FORMAT$(NOW,"Medium Date")
TEXT 15,160, "3",0,1,1, "Short Date: "+FORMAT$(NOW,"Short Date")
TEXT 15,210, "3",0,1,1, "Long Time: "+FORMAT$(NOW,"Long Time")
TEXT 15,260, "3",0,1,1, "Medium Time: "+FORMAT$(NOW,"Medium Time")
TEXT 15,310, "3",0,1,1, "Short Time: "+FORMAT$(NOW,"Short Time")
TEXT 15,360, "3",0,1,1, "c: "+FORMAT$(NOW,"c")
TEXT 15,410, "3",0,1,1, "d: "+FORMAT$(NOW,"d")
TEXT 15,460, "3",0,1,1, "dd: " +FORMAT$(NOW,"dd")
TEXT 15,510, "3",0,1,1, "ddd: " +FORMAT$(NOW,"ddd")
TEXT 15,560, "3",0,1,1, "dddd: " +FORMAT$(NOW,"dddd")
TEXT 15,610, "3",0,1,1, "ddddd: " +FORMAT$(NOW,"ddddd")
TEXT 15,660, "3",0,1,1, "dddddd: " +FORMAT$(NOW,"dddddd")
TEXT 15,710, "3",0,1,1, "w: " +FORMAT$(NOW,"w")
TEXT 15,760, "3",0,1,1, "ww: " +FORMAT$(NOW,"ww")
TEXT 15,810, "3",0,1,1, "m: " +FORMAT$(NOW,"m")
TEXT 15,860, "3",0,1,1, "mm: " +FORMAT$(NOW,"mm")
TEXT 15,910, "3",0,1,1, "mmm: " +FORMAT$(NOW,"mmm")
TEXT 15,960, "3",0,1,1, "mmmm: " +FORMAT$(NOW,"mmmm")
TEXT 15,1010, "3",0,1,1, "q: " +FORMAT$(NOW,"q")
TEXT 15,1060, "3",0,1,1, "y: " +FORMAT$(NOW,"y")
TEXT 15,1110, "3",0,1,1, "yy: " +FORMAT$(NOW,"yy")
TEXT 15,1160, "3",0,1,1, "yyyy: " +FORMAT$(NOW,"yyyy")
TEXT 15,1210, "3",0,1,1, "h: " +FORMAT$(NOW,"h")
TEXT 15,1260, "3",0,1,1, "hh: " +FORMAT$(NOW,"hh")
TEXT 15,1310, "3",0,1,1, "n: " +FORMAT$(NOW,"n")
TEXT 15,1360, "3",0,1,1, "nn: " +FORMAT$(NOW,"nn")
TEXT 15,1410, "3",0,1,1, "s: " +FORMAT$(NOW,"s")
TEXT 15,1460, "3",0,1,1, "ss: " +FORMAT$(NOW,"ss")
TEXT 15,1510, "3",0,1,1, "ttttt: " +FORMAT$(NOW,"ttttt")
TEXT 15,1560, "3",0,1,1, "AM/PM: " +FORMAT$(NOW,"AM/PM")
TEXT 15,1610, "3",0,1,1, "am/pm: " +FORMAT$(NOW,"am/pm")
TEXT 15,1660, "3",0,1,1, "A/P: " +FORMAT$(NOW,"A/P")
TEXT 15,1710, "3",0,1,1, "a/p: " +FORMAT$(NOW,"a/p")
TEXT 15,1760, "3",0,1,1, "AMPM: " +FORMAT$(NOW,"AMPM")
TEXT 15,1810, "3",0,1,1, "¥:" +FORMAT$(NOW,"To¥da¥y i¥s ddddd")
TEXT 15,1860, "3",0,1,1, "string: " +FORMAT$(NOW,"To¥da¥y i¥s ddddd")
PRINT 1
```

Result



```
General Date:1/9/2013 2:46:18 PM
Long Date:Tuesday, January 09 2013
Medium Date:09-Jan-13
Short Date:1/9/2013
Long Time:2:46:18 PM
Medium Time:02:46 PM
Short Time:14:46
c:1/9/2013 2:46:18 PM
d:9
dd:09
ddd:Tue
dddd:Tuesday
ddddd:1/9/2013
dddddd:Tuesday, January 09 2013
w:3
ww:2
m:1
mm:01
mmm:Jan
mmmm:January
q:1
y:9
yy:13
yyyy:2013
h:14
hh:14
n:46
nn:46
s:18
ss:18
ttttt:2:46:18 PM
AM/PM:PM
am/pm:pm
A/P:P
a/p:p
AMPM:PM
\:Today is 1/9/2013
string:Today is 1/9/2013
```

Sample Code	Result
SIZE 800 dot,850 dot GAP 0,0 DIRECTION 1 CLS TEXT 15,10, "3",0,1,1, "General Number: "+FORMAT$(1234.5,"General Number") TEXT 15,60, "3",0,1,1, "Currency: "+FORMAT$(1234.5,"Currency") TEXT 15,110, "3",0,1,1, "Fixed: "+FORMAT$(1234.5,"Fixed") TEXT 15,160, "3",0,1,1, "Standard: "+FORMAT$(1234.5,"Standard") TEXT 15,210, "3",0,1,1, "Percent: "+FORMAT$(1234.5,"Percent") TEXT 15,260, "3",0,1,1, "Scientific: "+FORMAT$(1234.5,"Scientific") TEXT 15,310, "3",0,1,1, "Yes/No: "+FORMAT$(1234.5,"Yes/No") TEXT 15,360, "3",0,1,1, "Yes/No: "+FORMAT$(0,"Yes/No") TEXT 15,410, "3",0,1,1, "True/False: "+FORMAT$(0,"True/False") TEXT 15,460, "3",0,1,1, "On/Off: "+FORMAT$(0,"On/Off") TEXT 15,510, "3",0,1,1, "00000.00: "+FORMAT$(1234.5,"00000.00") TEXT 15,560, "3",0,1,1, "#####.##: "+FORMAT$(1234.5,"#####.##") TEXT 15,610, "3",0,1,1, "##,##0.00: "+FORMAT$(1234.5,"##,##0.00") TEXT 15,660, "3",0,1,1, "$##0.00: "+FORMAT$(1234.5,"$##0.00") TEXT 15,710, "3",0,1,1, "$0.00%: "+FORMAT$(1234.5,"0.00%") TEXT 15,760, "3",0,1,1, "Yes/No: "+FORMAT$(-12.3,"Yes/No") TEXT 15,810, "3",0,1,1, "0.00;(0.00): "+FORMAT$(-12.3,"0.00;(0.00)") PRINT 1	 General Number: 1234.5 Currency: $1,234.50 Fixed: 1234.50 Standard: 1,234.50 Percent: 123450.00% Scientific: 1.23E+03 Yes/No: Yes Yes/No: No True/False: False On/Off: Off 00000.00: 01234.50 #####.##: 1234.5 ##,##0.00: 1,234.50 $##0.00: $1234.50 $0.00%: 123450.00% Yes/No: Yes 0.00;(0.00): (12.30)

10.52 DATEADD()

Description

Returns a date after which a specified time/date interval has been added. The returned value always uses with commands FORMAT$().

Syntax

DATEADD(interval$,number,date)

Parameter	Description
interval$,	The time/date interval for adding. It can be one of following values.

Interval$	The interval unit of parameter interval$
"yyyy"	Year.
"q"	Quarter.
"m"	Month.
"y"	Day of year.
"d"	Day.
"w"	Weekday.
"ww"	Week of year.
"h"	Hour.
"n"	Minute.
"s"	Second.

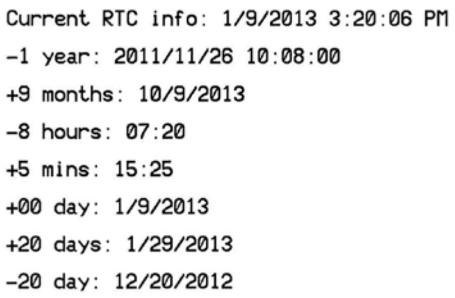
Number	The number of interval$ for adding.
Date	The date which is used to add the interval$. Date format: "yyyy/mm/dd" Time format: "hh:nn:ss"

Example

Sample Code 1

```
SIZE 4,2
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "Current RTC info: " +NOW$()
TEXT 10,60, "3",0,1,1, "-1 year: " +FORMAT$(DATEADD("yyyy",-1, " 11/26/2012 10:08:00"), "yyyy/mm/dd hh:nn:ss")
TEXT 10,110, "3",0,1,1, "+9 months: " +FORMAT$(DATEADD("m",9,NOW), "Short Date")
TEXT 10,160, "3",0,1,1, "-8 hours: " +FORMAT$(DATEADD("h",-8,NOW), "Short Time")
TEXT 10,210, "3",0,1,1, "+5 mins: " +FORMAT$(DATEADD("n",5,NOW), "Short Time")
TEXT 10,260, "3",0,1,1, "+00 day: " +FORMAT$(NOW, "Short Date")
TEXT 10,310, "3",0,1,1, "+20 days: " +FORMAT$(DATEADD("d",20,NOW), "Short Date")
TEXT 10,360, "3",0,1,1, "-20 day: " +FORMAT$(DATEADD("d",-20,NOW), "Short Date")
PRINT 1
```

Result 1

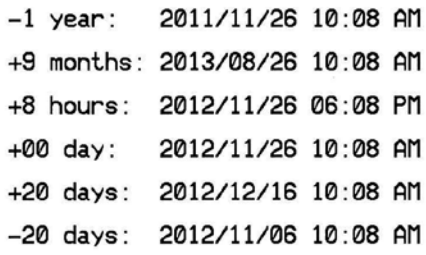


```
Current RTC info: 1/9/2013 3:20:06 PM
-1 year: 2011/11/26 10:08:00
+9 months: 10/9/2013
-8 hours: 07:20
+5 mins: 15:25
+00 day: 1/9/2013
+20 days: 1/29/2013
-20 day: 12/20/2012
```

Sample Code 2

```
SIZE 4,2
GAP 0,0
DIRECTION 1
CLS
TEXT 10,60, "3",0,1,1, "-1 year:     "+FORMAT$(DATEADD("yyyy", -1, "11/26/2012 10:08"),"yyyy/mm/dd hh:nn AM/PM")
TEXT 10,110,"3",0,1,1,"+9 months: "+FORMAT$(DATEADD("m",9,"11/26/2012 10:08"),"yyyy/mm/dd hh:nn AM/PM")
TEXT 10,160,"3",0,1,1,"+8 hours:   "+FORMAT$(DATEADD("h", +8,"11/26/2012 10:08"),"yyyy/mm/dd hh:nn AM/PM")
TEXT 10,210,"3",0,1,1,"+00 day:     "+FORMAT$("11/26/2012 10:08:00","yyyy/mm/dd hh:nn AM/PM")
TEXT 10,260,"3",0,1,1,"+20 days:    "+FORMAT$(DATEADD("d",20,"11/26/2012 10:08"),"yyyy/mm/dd hh:nn AM/PM")
TEXT 10,310,"3",0,1,1,"-20 days:    "+FORMAT$(DATEADD("d", -20,"11/26/2012 10:08"),"yyyy/mm/dd hh:nn AM/PM")
PRINT 1
```

Result 2



```
-1 year:    2011/11/26 10:08 AM
+9 months:  2013/08/26 10:08 AM
+8 hours:   2012/11/26 06:08 PM
+00 day:    2012/11/26 10:08 AM
+20 days:   2012/12/16 10:08 AM
-20 days:   2012/11/06 10:08 AM
```

10.53 FSEARCH()

Description

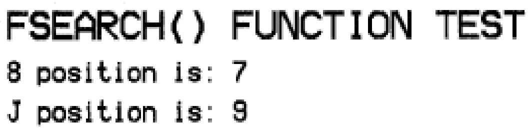
This function returns the position of a string.

Syntax

FSEARCH(file handle, STR$)

Parameter	Description
file handle	0 or 1
STR$	Required. Any valid string expression.

Example

Sample Code	Result
DOWNLOAD "DATA1",10,1234567890 DOWNLOAD "DATA2",15,ABCDEFGHIJKLMNO DOWNLOAD "Test.BAS" SIZE 4,1.5 GAP 0,0 DIRECTION 1 CLS OPEN "DATA1",0 OPEN "DATA2",1 TEXT 10,90,"4",0,1,1,"FSEARCH() FUNCTION TEST" A=FSEARCH(0,"8") B=FSEARCH(1,"J") TEXT 10,140,"3",0,1,1,"8 position is:"+STR$(A) TEXT 10,180,"3",0,1,1,"J position is:"+STR$(B) PRINT 1 EOP Test	 FSEARCH() FUNCTION TEST 8 position is: 7 J position is: 9

10.54 RECORDSET$ ( )

Description

This function returns a value from a table. Table is represented in a grid format, tabular form in rows and columns. Please refer to following table format on example.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

RECORDSET$(TABLE$, ROW, COLUMN [, DELIMITER])

Parameter	Description
TABLE$	Table name
ROW	Number of row
COLUMN	Number (or name) of column
DELIMITER	Optional. Set the delimiter of table. The default is 09H <Tab>

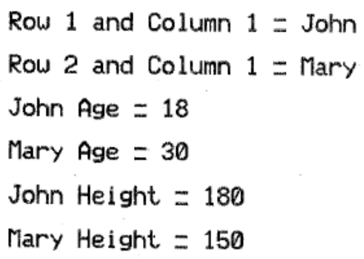
Note: The Row is always a number. But the column can be a number or name

Example

Sample Code 1:

```
DOWNLOAD F,"TEST.CSV",75,3
Name,Age,Height,Weight
John,18,180,80
Mary,30,150,50
Mark,65,170,65


DOWNLOAD F,"TEST.BAS"
CLOSE 0
SIZE 4,2
GAP 0,0
DIRECTION 1
CLS
TEXT 100,50,"3",0,1,1,"Row 1 and Column 1 = " +
RECORDSET$("TEST.CSV", 1, 1, ASC(","))
TEXT 100,100,"3",0,1,1,"Row 2 and Column 1 = " +
RECORDSET$("TEST.CSV", 2, 1, ASC(","))
TEXT 100,150,"3",0,1,1,"John Age = " +
RECORDSET$("TEST.CSV", 1, 2,ASC(","))
TEXT 100,200,"3",0,1,1,"Mary Age = " +
RECORDSET$("TEST.CSV", 2, 2,ASC(","))
TEXT 100,250,"3",0,1,1,"John Height = " +
RECORDSET$("TEST.CSV",1,"Height", ASC(","))
TEXT 100,300,"3",0,1,1,"Mary Height = " +
RECORDSET$("TEST.CSV",2,"Height", ASC(","))
PRINT 1

EOP
TEST
```

Result



```
Row 1 and Column 1 = John
Row 2 and Column 1 = Mary
John Age = 18
Mary Age = 30
John Height = 180
Mary Height = 150
```

Table format (TEST.CSV)

Number of rows	3			
Name of column	Name	Age	Height	Weight
Row 1	John	18	180	80
Row 2	Mary	30	150	50
Row 3	Mark	65	170	65
	Column 1	Column 2	Column 3	Column 4

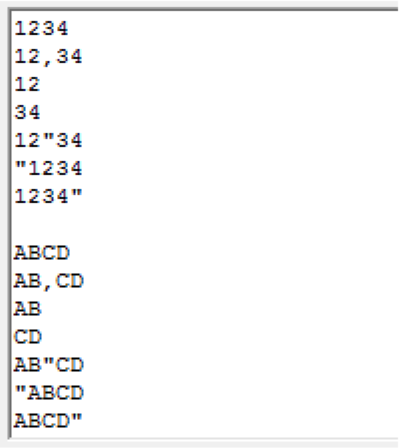
Sample Code 2:

```
DOWNLOAD "TEST.CSV",121,6,
Number,String
1234,ABCD
"12,34","AB,CD"
"12
34","AB
CD"
"12""34","AB""CD"
"""1234","""ABCD"
"1234""","ABCD"""


OUT RECORDSET$("TEST.CSV", 1, "Number", ASC(","))
OUT RECORDSET$("TEST.CSV", 2, 1, ASC(","))
OUT RECORDSET$("TEST.CSV", 3, 1, ASC(","))
OUT RECORDSET$("TEST.CSV", 4, 1, ASC(","))
OUT RECORDSET$("TEST.CSV", 5, 1, ASC(","))
OUT RECORDSET$("TEST.CSV", 6, 1, ASC(","))

OUT ""

OUT RECORDSET$("TEST.CSV", 1, "String", ASC(","))
OUT RECORDSET$("TEST.CSV", 2, 2, ASC(","))
OUT RECORDSET$("TEST.CSV", 3, 2, ASC(","))
OUT RECORDSET$("TEST.CSV", 4, 2, ASC(","))
OUT RECORDSET$("TEST.CSV", 5, 2, ASC(","))
OUT RECORDSET$("TEST.CSV", 6, 2, ASC(","))
```

Table format (TEST.CSV)

Number of rows	6	
Name of column	Number	String
Row 1	1234	ABCD
Row 2	12,34	AB,CD
Row 3	12 34	AB CD
Row 4	12"34	AB"CD
Row 5	"1234	"ABCD
Row 6	1234"	ABCD"
	Column 1	Column 2

Return



```
1234
12,34
12
34
12"34
"1234
1234"

ABCD
AB,CD
AB
CD
AB"CD
"ABCD
ABCD"
```

11 Device Reconfiguration Commands

11.1 SET COUNTER

Description

Counters can be a real counter or a variable. This setting sets the counter number in the program and its increments. There are three different types of counters: digit (0~9~0), lower case letter (a~z~a) or upper case letter (A~Z~A).

Syntax

```
SET COUNTER @n step
@n= "Expression "
```

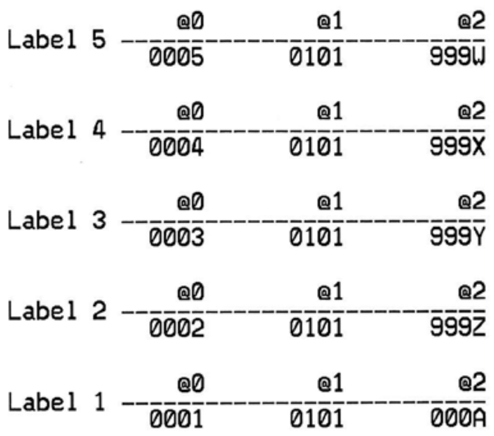
Parameter	Description
@n	n: counter number. There are 61 counters available (@0 ~ @60) in the printer. @0 to @50 will be cleared while restarting the printer. @51 to @60 will be stored in printer until the printer is restored to factory default.
Step	The increment of the counter, can be positive or negative. -999999999 <= step <= 999999999 *If the counter is used as a fixed variable, please set the increment to 0.*
Expression	Initial string. String length is 101 bytes

Example

Sample Code

```
SET COUNTER @0 +1
SET COUNTER @1 +0
SET COUNTER @2 -1
SET COUNTER @3 1

@0=" 0001"
@1=" 0101"
@2=" 000A"
@3=" 1"

SIZE 4,0.5
GAP 0,0
DIRECTION 1
CLS
TEXT 600,10," 3",0,1,1,3," @0          @1           @2"
TEXT 600,30," 3",0,1,1,3, « Label"   +@3+"   -------------------------“
TEXT 600,50,"3",0,1,1,3,@0+"        "   +@1+"        " +@2
PRINT 5
```

Result



```
             @0         @1         @2
Label 5 --------------------------------
           0005       0101       999W

             @0         @1         @2
Label 4 --------------------------------
           0004       0101       999X

             @0         @1         @2
Label 3 --------------------------------
           0003       0101       999Y

             @0         @1         @2
Label 2 --------------------------------
           0002       0101       999Z

             @0         @1         @2
Label 1 --------------------------------
           0001       0101       000A
```

See Also

PRINT, TEXT, BARCODE

11.2 SET CUTTER

Description

This setting activates or deactivates the cutter and defines how many printed labels is to be cut at one time. This setting will be saved in printer memory after turning off the power.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB)

Syntax

```
SET CUTTER OFF/BATCH/pieces
```

Parameter	Description
OFF	Disable cutter function.
BATCH	Set printer to cut label at the end of printing job.
Pieces	Set number of printing labels per cut. 0<= pieces <=65535

Example

Sample code	Result
SIZE 3,3 GAP 0,0 SET CUTTER OFF SET PEEL OFF CLS TEXT 50,50, "3",0,1,1, "SET CUTTER OFF" PRINT 3	The cutter function is disabling.
SET CUTTER BATCH CLS TEXT 50,50, "3",0,1,1, "SET CUTTER BATCH" PRINT 3,2	The cutter cuts once after 6 labels are printed.
SET CUTTER 1 CLS TEXT 50,50, "3",0,1,1, "SET CUTTER 1" PRINT 3,2	The cutter cuts every label.
CLS TEXT 50,50, "3",0,1,1, "SET CUTTER 2" PRINT 3,2	The cutter cuts every 2 labels.

See Also

OFFSET, PRINT, SET PARTIAL_CUTTER

11.3 SET PARTIAL_CUTTER

Description

This setting activates or deactivates the cutter and defines how many printed labels is to be cut at one time. This setting will be saved in printer memory after turning off the power. This function prevents label back feeding after a cut.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB)

Syntax

SET PARTIAL_CUTTER OFF/BATCH/Pieces

Parameter	Description
OFF	Disable cutter function.
BATCH	Set printer to cut label at the end of printing job.
Pieces	Set number of printing labels per cut. 0<= pieces <=65535

Note: This command is supported for the printer that have cutter module.

Example

Sample code

```
REM **SET PARTIAL_CUTTER FUNCTION OFF EXAMPLE PROGRAM**
SIZE 3,1
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET PARTIAL_CUTTER OFF
CLS
TEXT 50,50, "3",0,1,1, "SET PARTIAL_CUTTER OFF"
PRINT 3
REM ***This program cuts once at the batch***
SET PARTIAL_CUTTER BATCH
CLS
TEXT 50,50, "3",0,1,1, "SET PARTIAL_CUTTER BATCH"
PRINT 3,2
REM ***This program cuts every label***
SET PARTIAL_CUTTER 1
CLS
TEXT 50,50, "3",0,1,1, " SET PARTIAL_CUTTER 1"
PRINT 3,2
REM ***This program cuts 2 label***
SET PARTIAL_CUTTER 2
CLS
TEXT 50,50, "3",0,1,1, "SET PARTIAL_CUTTER 2"
PRINT 3,2
```

See Also

OFFSET, PRINT, SET CUTTER

11.4 SET BACK

Description

This setting is used after SET CUTTER function. This function prevents label backfeeding after a cut. (Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
SET BACK OFF/ON
```

Parameter	Description
OFF	Disable back function.
ON	Enable back function.

Example

Sample code

```
REM **SET BACK FUNCTION OFF EXAMPLE PROGRAM**
SIZE 3,1
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 1
REFERENCE 0,0
SET CUTTER 1
SET BACK OFF
CLS
TEXT 50,50, "3",0,1,1, "SET BACK OFF "
PRINT 3
CLS
SET CUTTER 1
SET BACK ON
TEXT 50,50, "3",0,1,1, "SET BACK ON "
PRINT 3
```

See Also

OFFSET, PRINT, SET CUTTER

11.5 SET KEYn

Description

This setting is used to enable/disable the KEYn function. Before setting KEYn function, please disable the default function of KEYn first. The setting will remain resident in the printer even when the printer is power off.

Syntax

SET KEYn ON/OFF/DEFAULT/MENU/PAUSE/PRINT m/FEED/BACKFEED/FORMFEED/CUT/INPUT "string "

Parameter	**Description**
n	0, 1, 2, 3, 4, 5, 6
ON	Enable KEYn function
OFF	Disable KEYn function
DEFAULT	Resume KEYn defaule function
MENU	Set to "MENU " key
PAUSE	Set to "PAUSE " key
PRINT m	Set to "PRINT " key m: Set number of printing labels per print. (0 < m < 32000)
FEED	Set to "FEED " key that can manual control the feeding distance
BACKFEED	Set to "BACKFEED " key that can manual control the backfeeding distance
FORMFEED	Set to "FORMFEED " key that will feed the label under the format. Ex: If format is "size 4,6, it will feed 6 ".
CUT	Set to "CUT " key
INPUT "string "	Send the command by press key (ex: **SET KEY1 INPUT "CONFIG " + CHR$(13) + CHR$(10)**)

The default function of KETn id as listed below:

Model	KEY0	KEY1	KEY2	KEY3	KEY4	KEY5	KEY6
TD-4T series		FEED					

Example

Sample code

```
DOWNLOAD “DEMO.BAS”
SIZE 3,1
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET KEY1 OFF
SET KEY2 OFF
SET KEY3 OFF
KEY1=0
KEY2=0
KEY3=0

:START
IF KEY1=1 THEN
CLS
TEXT 100,10, “3”,0,1,1, “KEY1 (MENU key) is pressed!! “
PRINT 1,1
```

```
ELSEIF KEY2=1 THEN
CLS
TEXT 100,10, “3”,0,1,1, “KEY2 (PAUSE key) is pressed!! “
PRINT 1,1
ELSEIF KEY3=1 THEN
CLS
TEXT 100,10, “3”,0,1,1, “KEY3 (FEED key) is pressed!! “
TEXT 100,60, “3”,0,1,1, “End of test”
PRINT 1,1
SET KEY1 ON
SET KEY2 ON
SET KEY3 ON
END
ENDIF
GOTO START
EOP
DEMO
```

See Also

OFFEST, PRINT

11.6 SET LEDn

Description

This setting is used to control LED on/off function.

Syntax

```
SET LED1 ON/OFF
SET LED2 ON/OFF
SET LED3 ON/OFF
```

LED no.	**Default Function**
LDE1	Power on/off
LED2	Printer on-line/off-line
LED3	Error/normal

Parameter	**Description**
ON	Enable LED*n* function
OFF	Disable LED*n* function

The default function of LED1, LED2 and LED3 id as listed below:

Model	LED1	LED2	LED3	LED4	LED5	LED6	LED7	LED2 & LED3
TD-4T series **Note: For this series, the LED1=LED2**	GREEN	GREEN	RED					ORANGE

Example

Sample code

```
DOWNLOAD "DEMO4.BAS "
SET LED1 OFF
SET LED2 OFF
SET LED3 OFF
FOR I=1 TO 100
LED1=0
LED2=0
LED3=0
IF I-INT(I/2)*2=0 THEN
LED1=1
ELSEIF I-INT(I/3)*3=0 THEN
LED2=1
ELSE
LED3=1
ENDIF
NEXT
LED1=1
LED2=1
LED3=0
SET LED1 ON
SET LED2 ON
SET LED3 ON
EOP
DEMO4
```

11.7 SET PEEL

Description

This setting is used to enable/disable the self-peeling function. The default setting for this function is off. When this function is set on, the printer stops after each label printing, and does not print the next label until the peeled label is taken away. This setting will be saved in printer memory when turning off the power. (Available for TD-4420TN/4520TN/4650TNWB/4750TNWB)

Syntax

SET PEEL ON/OFF

Parameter	Description
ON	Enable the self-peeling function
OFF	Disable the self-peeing function

Example

Sample code

```
REM ***SELF-PEELING FUNCTION ON***
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL ON
CLS
TEXT 50,100, "3 ",0,1,1, "SELF-PEELING FUNCTION TEST "
PRINT 5
```

See Also

OFFEST, PRINT

11.8 SET REWIND

Description

This setting is used to enable/disable the internal rewind function. This setting will be saved in printer memory when turning off the power.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

SET REWIND ON/OFF/RS232

Parameter	Description
ON	Enable the internal rewind function
OFF	Disable the internal rewind or external rewind module function
RS232	Enable the external rewind module function (via RS-232 port/ pull high signal)

Example

Sample code

```
REM ***REWIND FUNCTION ON***
SIZE 4,4
GAP 0.12,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET REWIND ON
CLS
TEXT 50,100, "3 ",0,1,1, "REWIND FUNCTION TEST "
PRINT 500
```

See Also

OFFEST, PRINT

11.9 SET TEAR

Description

This command is used to send the label to the tear position up to the gap/black mark. This setting will be saved in printer memory when turning off the power.

Syntax

SET TEAR ON/OFF (FBPL language printers only)

Parameter	Description
ON	The label gap will stop at the tear off position after print.
OFF	The label gap will NOT stop at the tear off position after print. The beginning of label will be aligned to print head.

Example

Sample code

```
REM ***TEAR FUNCTION ON***
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
SET TEAR ON
CLS
TEXT 50,100, "3 ",0,1,1, "TEAR FUNCTION TEST "
PRINT 1
```

See Also

SET PEEL, SET CUTTER

11.10 SET GAP

Description

This setting sets the gap sensor emission sensitivity. The printer initiates automatic gap sensor calibration when the PAUSE key is held down while powering up. This function may cease to work if the thickness of the backing paper and that of label with backing paper are not of appreciable difference to the sensor, or when there are pre-printed marks or patterns on the label. In such case, users must calibrate the gap sensor manually by this command through trial-and-error method to attain the proper setting. This setting will be saved in printer memory when turning off the power.

Syntax

SET GAP n/AUTO/OFF/0,/REVERSE/OBVERSE

Parameter	**Description**
N	Gap sensor light emission strength. Available range is listed as below. 0 is the lowest sensitivity
AUTO	The printer will feed 2 or 3 labels to calibrate the gap. If the label is continuous, the printer will feed label to limit 10~20 inches to confirm if the label is continuous.
OFF	Disable the SET GAP AUTO function.
0,	Automatically calibrate the gap size.
REVERSE	This function is used when the Black Mark is the separation in the front of the label and which can't be detected by the Black Mark sensor. The parts of the media which can be passed through by GAP sensor are defined to be the printable area, otherwise it will be defined to the GAP of the media.
OBVERSE	Disable the "SET GAP REVERSE " function.

Printer model	Gap Sensor Range	Black Mark Sensor Range	SET GAP REVERSE
			SET GAP OBVERSE
			SET GAP AUTO

TD-4420TN/4520TN	0~15	0~3	V
TD-4650TNWB/4650TNWBR/ 4750TNWB/4750TNWBR	0~15	0~3	V

Note:

**** When in "SET HEAD OFF " mode, the function "SET GAP AUTO " doesn't work even the printer head is opened and closed, but it can work when power on the printer.***

Example

The example below is operated in DOS environment via the parallel port connection to setup the label size, gap distance and sensor sensitivity.

```
C:¥>COPY CON LPT1<ENTER>
SIZE 4,2.5<ENTER>
GAP 0.12,0<ENTER>
SET GAP 1<ENTER>
<CTRL><Z><ENTER>
C:¥>
```

Note:
<ENTER> stands for keyboard "ENTER " key. In the above example, please press "ENTER " key instead of typing <ENTER> in the above example. <CTRL> stands for keyboard “Ctrl” key.

Troubleshooting:
Press the FEED key to test. Does printer stop at the same position on each label without the error light blinking? If not, adjust the setting to a larger number. When adjusting this setting, begin from 0 and then on to higher values incrementally.

See Also

SIZE, GAP, BLINE

11.11 SET BLINE

Description

This setting is using to reverse/obverse the sensor function.

Syntax

SET BLINE REVERSE/OBVERSE

Parameter	Description
REVERSE	Reverse the sensor function. Redefine the reflective area is black line and non-reflective part is paper. (Normally, reflective part is paper and non-reflective part is black line.)
OBVERSE	Disable the “SET BLINE REVERSE” function.

11.12 SET HEAD

Description

This setting is used to enable/disable head open sensor. If the head open sensor is turned off, an open printer head will not return an error message. This setting will be saved in printer memory.

Syntax

SET HEAD ON /OFF

Parameter	Description
ON	Turn on the "HEAD OPEN " sensor
OFF	Turn off the "HEAD OPEN " sensor

Example

```
SET HEAD ON
SET HEAD OFF
```

11.13 SET RIBBON

Description

This setting is used to enable/disable ribbon sensor detection. (Thermal Transfer Printing/Direct Thermal Printing) Printer will detect the presence of a ribbon to determine using either direct thermal or thermal transfer printing upon printer startup. This setting will NOT be saved in printer memory.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

SET RIBBON ON/OFF/INSIDE/OUTSIDE

Parameter	Description
ON	Thermal transfer printing
OFF	Direct Thermal Printing

Example

Sample Code

```
REM *****Disable ribbon detection sensor for direct thermal printing.
SET RIBBON OFF
SIZE 4,1
GAP 0,0
CLS
TEXT 10,10, " 3 "  ,0,1,1, "  Direct thermal printing. "
PRINT 1

REM *****Enable ribbon detection sensor for thermal transfer printing.
SET RIBBON ON
SIZE 4,1
GAP 0,0
CLS
TEXT 10,10, " 3 "  ,0,1,1, "  Thermal transfer printing. "
PRINT 1
```

11.14 SET ENCODER

Description

This setting is used to enable/disable ribbon encoder sensor detection.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

SET ENCODER ON/OFF

Parameter	Description
ON	Enable ribbon encoder sensor.
OFF	Disable ribbon encoder sensor.

Example

SET ENCODER ON
SET ENCODER OFF

11.15 SET RIBBONEND

Description

This setting is used to enable/disable ribbon-end sensor detection.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

SET RIBBONEND ON/OFF

Parameter	Description
ON	Enable ribbon-end sensor.
OFF	Disable ribbon-end sensor.

Example

SET RIBBONEND ON
SET RIBBONEND OFF

11.16 SET COM1

Description

This setting defines communication parameters for printer serial port.

Syntax

SET COM1 baud,parity,data,stop

Parameter	Description
baud	Baud rate, available baud rates are as listed : 24: 2400 bps 48: 4800 bps 96: 9600 bps 19: 19200 bps 38: 38400 bps 57: 57600 bps 115: 115200 bps
parity	Parity check N: No parity check E: Even parity check O: Odd parity check
Data	Data bit 8: 8 bits data 7: 7 bits data
stop	Stop bit 1: 1 stop bit 2: 2 stop bits

Example

The parallel port is used to setup the printer serial port in this example via MS-DOS mode.

```
C:¥>COPY CON LPT1<ENTER>
SET COM1 19,N,8,1<ENTER>
<CTRL><Z><ENTER>
C:¥>
```

Note:

<ENTER> stands for PC keyboard "ENTER" key. <CTRL><Z> means to hold PC keyboard "CTRL" key then press the PC keyboard <Z> key.

11.17 SET PRINTKEY

Description

This command will print one label and feed label gap to tear bar position for tearing away. Press FEED button to print the next label or batch of labels. If label content includes serial text or barcode, it will change the serial number accordingly. This setting will be saved in printer memory.

Syntax

SET PRINTEKY OFF/ON/AUTO/<num>

Parameter	Description
OFF	Disable this function
ON	Enable this function
AUTO	Enable this function
<num>	Numbers of labels will be printed if FEED button is pressed.

Example

Sample code

```
SIZE 4,2.5
GAP 0.12,0
SET PRINTKEY ON
SET COUNTER @0 1
@0= "0001"
CLS
TEXT 10,10, "5",0,1,1,@0
PRINT 1
```

Execute:

Syntax	Receive “PRINT m”	Print Out
SET PRINTKEY ON or	1.) PRINT 2	Label 1~2
SET PRINTKEY AUTO	2.) Press FEED key	Label 3~4

Syntax	Receive “PRINT m,n”	Print Out
SET PRINTKEY ON or	1.) PRINT 1,2	Label 1, Label 1
SET PRINTKEY AUTO	2.) Press FEED key	Label 2, Label 2

Syntax	Receive “PRINT -1,n”	Print Out
SET PRINTKEY ON or	1.) PRINT -1,2	Label 1, Label 1
SET PRINTKEY AUTO	2.) Press FEED key	Label 1, Label 1

Syntax	Receive “PRINT m”	Print Out
SET PRINTKEY 5	1.) PRINT 2	Label 1~2
	2.) Press FEED key	Label 3~7
Syntax	**Receive “PRINT m,n”**	**Print Out**
SET PRINTKEY 5	1.) PRINT 1,2	Label 1, Label 1
	2.) Press FEED key	Label 2~6

Syntax	Receive “PRINT –1,n”	Print Out
SET PRINTKEY 5	1.) PRINT -1,2	Label 1, Label 1
	2.) Press FEED key	Label 1, Label 1

11.18 SET REPRINT

Description

This command will disable/enable a reprinting attempt subsequent to a “no paper”, “no ribbon” or “carriage open” error.

Syntax

SET REPRINT OFF/ON

Parameter	Description
OFF	Disable this function
ON	Enable this function

Example

```
SET REPRINT ON
```

11.19 SET FEED_LEN

Description

This command can set the feeding length when FEED key is pressed. This setting will be memorized by printer. The initialized value is the label length.

Syntax

SET FEED_LEN n

Parameter	Description
n	The feeding length in dot.

Example

Sample code

```
SET FEED_LEN 100
```

Result

The feeding length is 100 dots when you press the FEED button after this setting.

11.20 GETSENSOR()

Description

This command is used to get the sensor status/AD value. We can use it to check the sensor function.

Syntax

GETSENSOR(sensor$[,intension])

Parameter	Description
sensor$	Sensor type.

GAP	Gap sensor
BLINE	Black mark sensor
RIBBON	Ribbon-end sensor
PEEL	Peeler sensor
HEAD UP	Thermal print head open sensor
HEAD TEMP	The temperature of print head
HEAD VOLT	The voltage of print head
BATTERY VOLT	The voltage of battery
BATTERY CAP	The capacity of battery

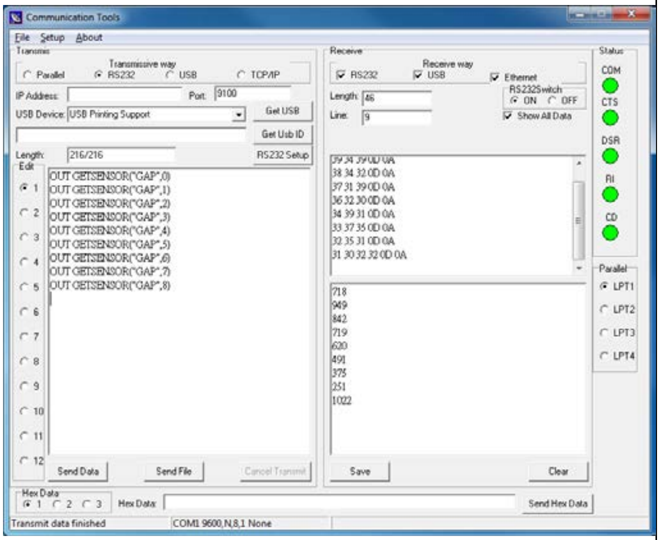
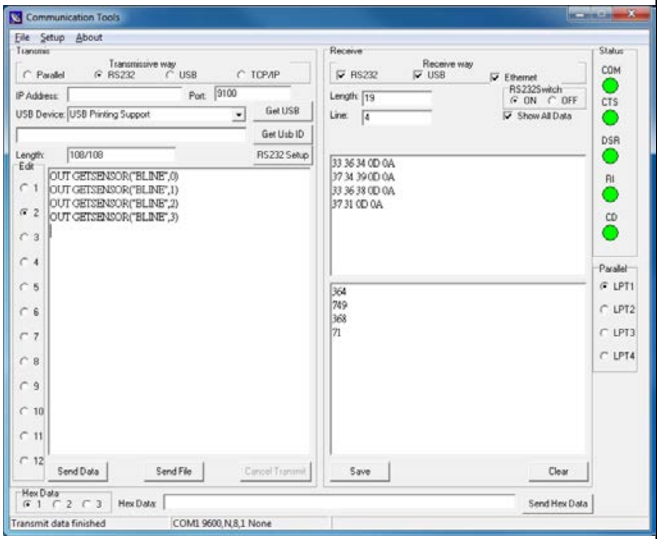
intension	Sensor intension.

Gap intension	Please refer to SET GAP for gap sensor range of different model.
BLINE intension	Please refer to SET GAP for black mark sensor range of different model.
RIBBON intension	0 ~ 3
PEEL sensor intension	Ignored
HEAD UP sensor intension	Ignored
HEAD TEMP	Ignored
HEAD VOLT	Ignored

Returned value

Gap	Return the AD value of gap sensor
BLINE	Return the AD value of black mark sensor
RIBBON	Return the AD value of ribbon sensor
PEEL	The return value will be either 0 or 1 0: Paper is not on the sensor 1: Paper is on the sensor
HEAD UP	The return value will be either 0 or 1 0: print head module is close 1: print head module is open
HEAD TEMP	Return the temperature of thermal print head
HEAD VOLT	Return the voltage of thermal print head

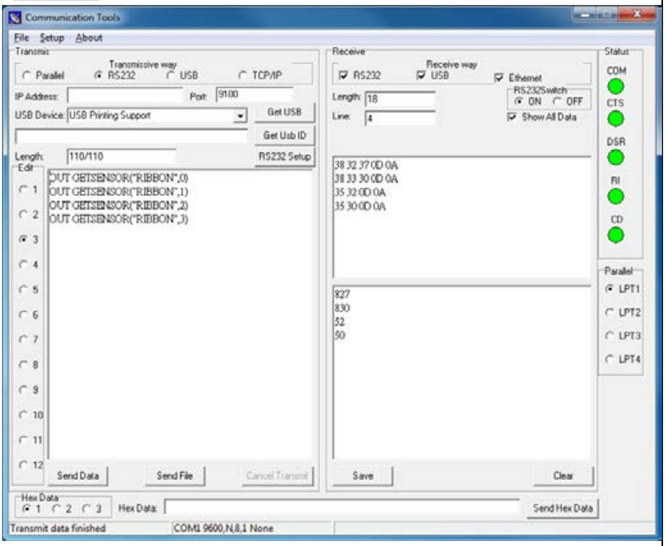
Example (*Use CommTool to get sensor status via RS-232.*)

Sample code	Sample code
OUT GETSENSOR("GAP",0) OUT GETSENSOR("GAP",1) OUT GETSENSOR("GAP",2) OUT GETSENSOR("GAP",3) OUT GETSENSOR("GAP",4) OUT GETSENSOR("GAP",5) OUT GETSENSOR("GAP",6) OUT GETSENSOR("GAP",7) OUT GETSENSOR("GAP",8)	OUT GETSENSOR("BLINE",0) OUT GETSENSOR("BLINE",1) OUT GETSENSOR("BLINE",2) OUT GETSENSOR("BLINE",3)
Result	**Result**
 Length: 216/216 OUT GETSENSOR("GAP",0) OUT GETSENSOR("GAP",1) OUT GETSENSOR("GAP",2) OUT GETSENSOR("GAP",3) OUT GETSENSOR("GAP",4) OUT GETSENSOR("GAP",5) OUT GETSENSOR("GAP",6) OUT GETSENSOR("GAP",7) OUT GETSENSOR("GAP",8) Receive Length: 46 Line: 9 39 34 39 0D 0A 38 34 32 0D 0A 37 31 39 0D 0A 36 32 30 0D 0A 34 39 31 0D 0A 33 37 35 0D 0A 32 35 31 0D 0A 31 30 32 32 0D 0A 718 949 842 719 620 491 375 251 1022 Transmit data finished COM1 9600,N,8,1 None	 Length: 108/108 OUT GETSENSOR("BLINE",0) OUT GETSENSOR("BLINE",1) OUT GETSENSOR("BLINE",2) OUT GETSENSOR("BLINE",3) Receive Length: 19 Line: 4 33 36 34 0D 0A 37 34 39 0D 0A 33 36 38 0D 0A 37 31 0D 0A 364 749 368 71 Transmit data finished COM1 9600,N,8,1 None
***If the returned valued is changed in different sensor intension, we can say the sensor is functional.**	***If the returned valued is changed in different sensor intension, we can say the sensor is functional.**

Sample code

```
OUT GETSENSOR("RIBBON",0)
OUT GETSENSOR("RIBBON",1)
OUT GETSENSOR("RIBBON",2)
OUT GETSENSOR("RIBBON",3)
```

Result



Communication Tools

Transmit: Transmissive way ○ Parallel ● RS232 ○ USB ○ TCP/IP; IP Address: ; Port: 9100; USB Device: USB Printing Support; Length: 110/110; Edit ● 3

```
OUT GETSENSOR("RIBBON",0)
OUT GETSENSOR("RIBBON",1)
OUT GETSENSOR("RIBBON",2)
OUT GETSENSOR("RIBBON",3)
```

Receive: Receive way ☑ RS232 ☑ USB ☑ Ethernet; Length: 18; Line: 4; RS232Switch ● ON ○ OFF; ☑ Show All Data

```
38 32 37 0D 0A
38 33 30 0D 0A
35 32 0D 0A
35 30 0D 0A
```

```
827
830
52
50
```

Parallel ● LPT1; Hex Data ● 1

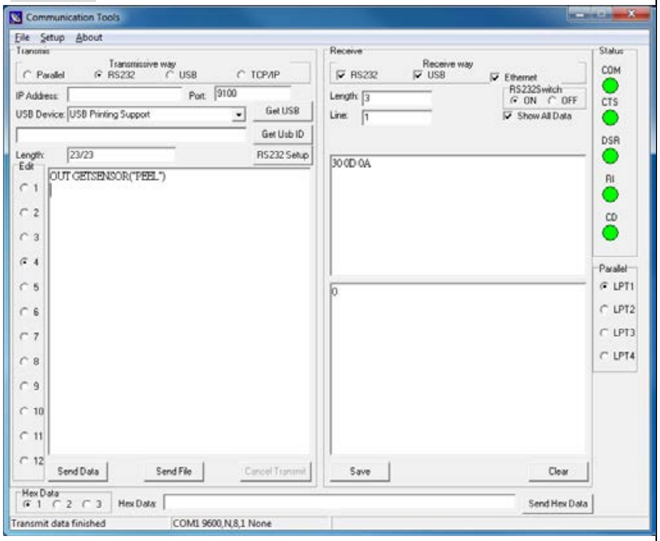
Transmit data finished — COM1 9600,N,8,1 None

***If the returned valued is changed in different sensor intension, we can say the sensor is functional.**

Sample code

```
OUT GETSENSOR("PEEL")
```

Result



Communication Tools

Transmit: Transmissive way ○ Parallel ● RS232 ○ USB ○ TCP/IP; Port: 9100; USB Device: USB Printing Support; Length: 23/23; Edit ● 4

```
OUT GETSENSOR("PEEL")
```

Receive: Receive way ☑ RS232 ☑ USB ☑ Ethernet; Length: 3; Line: 1; RS232Switch ● ON ○ OFF; ☑ Show All Data

```
30 0D 0A
```

```
0
```

Parallel ● LPT1; Hex Data ● 1

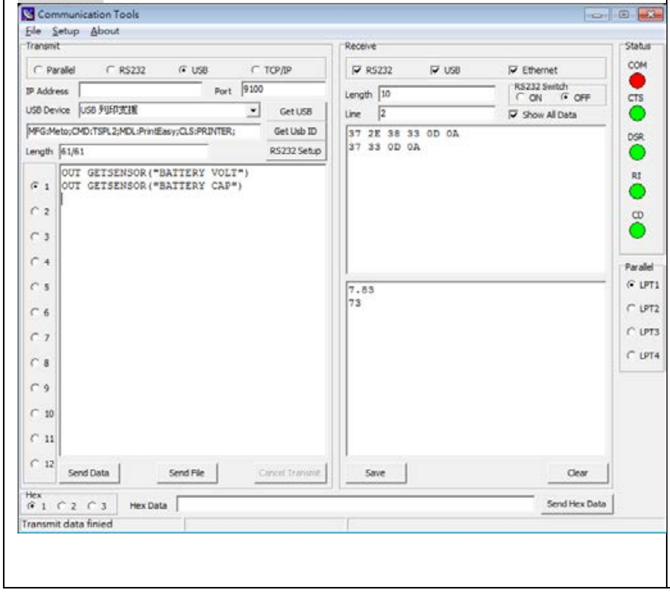
Transmit data finished — COM1 9600,N,8,1 None

***0: Paper is not on the sensor. 1: Paper is on the sensor.**

Sample code

```
OUT GETSENSOR("BATTERY VOLT")
OUT GETSENSOR("BATTERY CAP")
```

Result



Communication Tools

Transmit: ○ Parallel ○ RS232 ● USB ○ TCP/IP; IP Address: ; Port: 9100; USB Device: USB 列印支援; MFG:Meto;CMD:TSPL2;MDL:PrintEasy;CLS:PRINTER;; Length: 61/61; ● 1

```
OUT GETSENSOR("BATTERY VOLT")
OUT GETSENSOR("BATTERY CAP")
```

Receive: ☑ RS232 ☑ USB ☑ Ethernet; Length: 10; Line: 2; RS232 Switch ○ ON ● OFF; ☑ Show All Data

```
37 2E 38 33 0D 0A
37 33 0D 0A
```

```
7.83
73
```

Parallel ● LPT1; Hex ● 1

Transmit data finied

11.21 GETSETTING$()

Description

This command is used to get printer settings.

Syntax

GETSETTING$ (app$,sec$,key$[,default$])

app$	sec$	key$	Comment
SYSTEM	INFORMATION	DPI	Return printer resolution
		MODEL	Return printer model name
		SERIAL	Return Printer serial number
		VERSION	Return Printer firmware version
		CHECKSUM	Return Printer firmware checksum
	RECORD	MILAGE	Return printed mileage
		CUT COUNTER	Return cutter cuts
FILE	DRAM	CAPACITY	Return the total capacity of DRAM
		AVAILABLE	Return the available capacity of DRAM
	FLASH	CAPACITY	Return the total capacity of FLASH
		AVAILABLE	Return the available capacity of FLASH
	CARD	CAPACITY	Return the total capacity of CARD
		AVAILABLE	Return the available capacity of CARD
		INSTALLED	Return the status of card. 1: installed; 0: none installed.
CONFIG	NET	MAC ADDRESS	Return MAC address
		IP ADDRESS	Return IP address
		SUBNET MASK	Return Subnet Mask
		DEFAULT GATEWAY	Return default gateway
		RAW PORT	Return raw port
		NAME	Return printer name
		PRIMARY DNS	Return primary DNS
		SECONDARY DNS	Return secondary DNS
	WLAN	MAC ADDRESS	Return MAC address
		IP ADDRESS	Return IP address
		SUBNET MASK	Return Subnet Mask
		DEFAULT GATEWAY	Return default gateway
		RAW PORT	Return raw port
	COM1	BAUD RATE	Return baud rate of COM port
		DATA BIT	Return data bit of COM port
		PARITY	Return parity of COM port
		STOP BIT	Return stop bit of COM port
	SENSOR	SENSOR TYPE	Return the current sensor type
		CARRIAGE	Return the status of head open sensor.
		GAP INTENSION	Return intension of gap sensor.
		BLINE INTENSION	Return intension of black mark sensor.
		CONTINUOUS INTENSION	Return intension of continuous sensor.
	FBPL	PRINT MODE	Return pos-print action.
		DENSITY	Return print density
		PAPER SIZE	Return paper size
		GAP SIZE	Return gap size
		BLINE SIZE	Return black mark size
		DIRECTION	Return printing direction
		MIRROR	Return mirror status.
		RIBBON	Return ribbon status.
		REPRINT	Return reprint status.
		PAPER WIDTH	Return paper width
		LIMIT FEED	Return maximum length for sensor calibration.
		OFFSET	Return OFFSET value.
		REFERENCE X	Return REFERENCE X value.
		REFERENCE Y	Return REFERENCE Y value.
		SHIFT X	Return SHIFT X value.
		SHIFT Y	Return SHIFT Y value.
		SPEED	Return print speed.
		COUNTRY CODE	Return COUNTRY code.
		CODEPAGE	Return CODEPAGE.

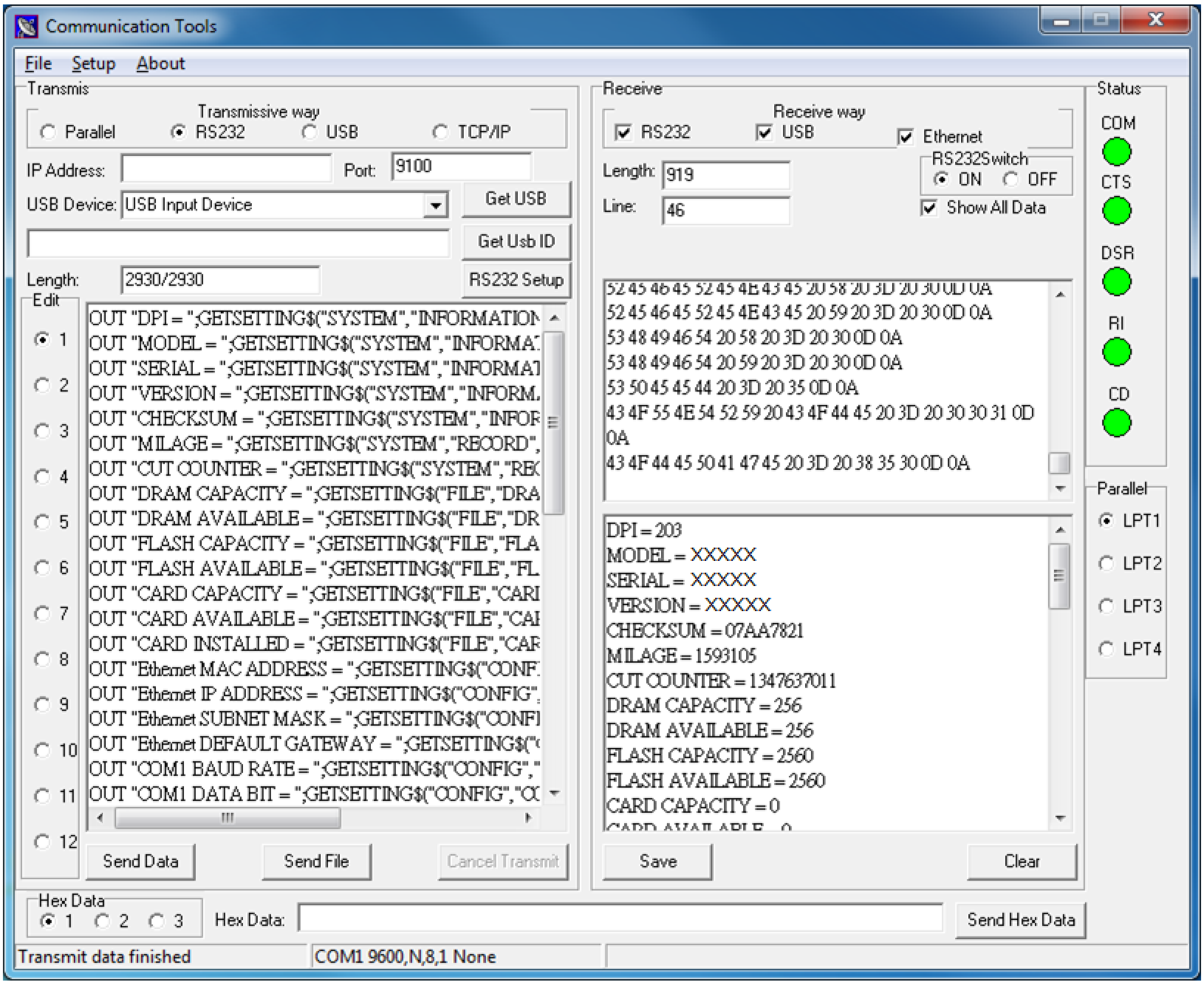
Parameter	Description
default$	Optional. Expression containing the value to return if no value is set in the key$ setting. If omitted, default is assumed to be a zero-length string ("").

Example

Sample code (Use CommTool to get printer settings via RS-232.)

```
OUT "DPI = ";GETSETTING$("SYSTEM","INFORMATION","DPI")
OUT " MODEL =";GETSETTING$("SYSTEM ", " INFORMATION ", " MODEL")
OUT "SERIAL = ";GETSETTING$("SYSTEM","INFORMATION","SERIAL")
OUT "VERSION = ";GETSETTING$("SYSTEM", "INFORMATION", "VERSION")
OUT "CHECKSUM = ";GETSETTING$("SYSTEM","INFORMATION","CHECKSUM")
OUT "MILAGE = ";GETSETTING$("SYSTEM", "RECORD", "MILAGE")
OUT "CUT COUNTER = ";GETSETTING$("SYSTEM","RECORD","CUT COUNTER")
OUT "DRAM CAPACITY = ";GETSETTING$("FILE", "DRAM", "CAPACITY")
OUT "DRAM AVAILABLE = ";GETSETTING$("FILE","DRAM","AVAILABLE")
OUT "FLASH CAPACITY = ";GETSETTING$("FILE", "FLASH", "CAPACITY")
OUT "FLASH AVAILABLE = ";GETSETTING$("FILE","FLASH","AVAILABLE")
OUT "CARD CAPACITY = ";GETSETTING$("FILE", "CARD", "CAPACITY")
OUT "CARD AVAILABLE = ";GETSETTING$("FILE","CARD","AVAILABLE")
OUT "CARD INSTALLED = ";GETSETTING$("FILE", "CARD", "INSTALLED")
OUT "Ethernet MAC ADDRESS = ";GETSETTING$("CONFIG", "NET", "MAC ADDRESS")
OUT "Ethernet IP ADDRESS = ";GETSETTING$("CONFIG", "NET", "IP ADDRESS")
OUT "Ethernet SUBNET MASK = ";GETSETTING$("CONFIG", "NET", "SUBNET MASK")
OUT "Ethernet DEFAULT GATEWAY = ";GETSETTING$("CONFIG","NET","DEFAULT GATEWAY")
OUT "Ethernet PRIMARY DNS = ";GETSETTING$("CONFIG","NET","PRIMARY DNS")
OUT "Ethernet SECONDARY DNS = ";GETSETTING$("CONFIG","NET","SECONDARY DNS")
OUT "COM1 BAUD RATE = ";GETSETTING$("CONFIG", "COM1", "BAUD RATE")
OUT "COM1 DATA BIT = ";GETSETTING$("CONFIG","COM1","DATA BIT")
OUT "COM1 PARITY = ";GETSETTING$("CONFIG", "COM1", "PARITY")
OUT "COM1 STOP BIT = ";GETSETTING$("CONFIG", "COM1", "STOP BIT")
OUT "SENSOR TYPE = ";GETSETTING$("CONFIG","SENSOR","SENSOR TYPE")
OUT "CARRIAGE = ";GETSETTING$("CONFIG", "SENSOR", "CARRIAGE")
OUT "GAP INTENSION = ";GETSETTING$("CONFIG","SENSOR","GAP INTENSION")
OUT "BLINE INTENSION = ";GETSETTING$("CONFIG", "SENSOR", "BLINE INTENSION")
OUT "CONTINUOUS INTENSION = ";GETSETTING$("CONFIG","SENSOR","CONTINUOUS INTENSION")
OUT "PRINT MODE = ";GETSETTING$("CONFIG", "FBPL", "PRINT MODE")
OUT "DENSITY = ";GETSETTING$("CONFIG","FBPL","DENSITY")
OUT "PAPER SIZE = ";GETSETTING$("CONFIG", "FBPL", "PAPER SIZE")
OUT "GAP SIZE = ";GETSETTING$("CONFIG","FBPL","GAP SIZE")
OUT "BLINE SIZE = ";GETSETTING$("CONFIG", "FBPL", "BLINE SIZE")
OUT "DIRECTION = ";GETSETTING$("CONFIG","FBPL","DIRECTION")
OUT "MIRROR = ";GETSETTING$("CONFIG","FBPL","MIRROR")
OUT "RIBBON = ";GETSETTING$("CONFIG", "FBPL", "RIBBON")
OUT "REPRINT = ";GETSETTING$("CONFIG","FBPL","REPRINT")
OUT "PAPER WIDTH = ";GETSETTING$("CONFIG", "FBPL", "PAPER WIDTH")
OUT "LIMIT FEED = ";GETSETTING$("CONFIG","FBPL","LIMIT FEED")
OUT "OFFSET = ";GETSETTING$("CONFIG","FBPL","OFFSET")
OUT "REFERENCE X = ";GETSETTING$("CONFIG", "FBPL", "REFERENCE X")
OUT "REFERENCE Y = ";GETSETTING$("CONFIG","FBPL","REFERENCE Y")
OUT "SHIFT X = ";GETSETTING$("CONFIG", "FBPL", "SHIFT X")
OUT "SHIFT Y = ";GETSETTING$("CONFIG","FBPL","SHIFT Y")
OUT "SPEED = ";GETSETTING$("CONFIG", "FBPL", "SPEED")
OUT "COUNTRY CODE = ";GETSETTING$("CONFIG","FBPL","COUNTRY CODE")
OUT "CODEPAGE = ";GETSETTING$("CONFIG","FBPL","CODEPAGE")
```

Result



Transmissive way: ○ Parallel ● RS232 ○ USB ○ TCP/IP

IP Address:

Port: 9100

USB Device: USB Input Device

Length: 2930/2930

Edit: ● 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7 ○ 8 ○ 9 ○ 10 ○ 11 ○ 12

```
OUT "DPI = ";GETSETTING$("SYSTEM","INFORMATION
OUT "MODEL = ";GETSETTING$("SYSTEM","INFORMA
OUT "SERIAL = ";GETSETTING$("SYSTEM","INFORMAT
OUT "VERSION = ";GETSETTING$("SYSTEM","INFORM.
OUT "CHECKSUM = ";GETSETTING$("SYSTEM","INFOR
OUT "MILAGE = ";GETSETTING$("SYSTEM","RECORD",
OUT "CUT COUNTER = ";GETSETTING$("SYSTEM","REC
OUT "DRAM CAPACITY = ";GETSETTING$("FILE","DRA
OUT "DRAM AVAILABLE = ";GETSETTING$("FILE","DR
OUT "FLASH CAPACITY = ";GETSETTING$("FILE","FLA
OUT "FLASH AVAILABLE = ";GETSETTING$("FILE","FL
OUT "CARD CAPACITY = ";GETSETTING$("FILE","CARI
OUT "CARD AVAILABLE = ";GETSETTING$("FILE","CAI
OUT "CARD INSTALLED = ";GETSETTING$("FILE","CAR
OUT "Ethemet MAC ADDRESS = ";GETSETTING$("CONF
OUT "Ethemet IP ADDRESS = ";GETSETTING$("CONFIG",
OUT "Ethemet SUBNET MASK = ";GETSETTING$("CONFI
OUT "Ethemet DEFAULT GATEWAY = ";GETSETTING$("
OUT "COM1 BAUD RATE = ";GETSETTING$("CONFIG","
OUT "COM1 DATA BIT = ";GETSETTING$("CONFIG","C
```

Receive way: ☑ RS232 ☑ USB ☑ Ethernet

RS232Switch: ● ON ○ OFF

☑ Show All Data

Length: 919

Line: 46

```
52 45 46 45 52 45 4E 43 45 20 58 20 3D 20 30 0D 0A
52 45 46 45 52 45 4E 43 45 20 59 20 3D 20 30 0D 0A
53 48 49 46 54 20 58 20 3D 20 30 0D 0A
53 48 49 46 54 20 59 20 3D 20 30 0D 0A
53 50 45 45 44 20 3D 20 35 0D 0A
43 4F 55 4E 54 52 59 20 43 4F 44 45 20 3D 20 30 30 31 0D
0A
43 4F 44 45 50 41 47 45 20 3D 20 38 35 30 0D 0A
```

```
DPI = 203
MODEL = XXXXX
SERIAL = XXXXX
VERSION = XXXXX
CHECKSUM = 07AA7821
MILAGE = 1593105
CUT COUNTER = 1347637011
DRAM CAPACITY = 256
DRAM AVAILABLE = 256
FLASH CAPACITY = 2560
FLASH AVAILABLE = 2560
CARD CAPACITY = 0
CARD AVAILABLE = 0
```

Parallel: ● LPT1 ○ LPT2 ○ LPT3 ○ LPT4

Hex Data: ● 1 ○ 2 ○ 3

Hex Data:

Transmit data finished | COM1 9600,N,8,1 None

Sample code(NET, WLAN)

```
OUT "Ethernet DEFAULT RAW PORT = ";GETSETTING$("CONFIG","NET","RAW PORT")


OUT "WLAN MAC ADDRESS = ";GETSETTING$("CONFIG", "WLAN", "MAC ADDRESS")
OUT "WLAN IP ADDRESS = ";GETSETTING$("CONFIG","WLAN","IP ADDRESS")
OUT "WLAN SUBNET MASK = ";GETSETTING$("CONFIG", "WLAN", "SUBNET MASK")
OUT "WLAN DEFAULT GATEWAY = ";GETSETTING$("CONFIG","WLAN","DEFAULT GATEWAY")
OUT "WLAN DEFAULT RAW PORT = ";GETSETTING$("CONFIG", "WLAN", "RAW PORT")
OUT "NET Name = ";GETSETTING$("CONFIG", "NET", "NAME")
```

11.22 SET USBHOST

Description

This command can set the USB host for the usage of PC USB keyboard and USB HID interface scanner. (Available for TD-4420TN/4520TN/4650TNWB/4650TNWBR/4750TNWB/4750TNWBR)

Syntax

```
SET USBHOST KEYBOARD/SCANNER
```

Parameter	**Description**
KEYBOARD	USB keyboard (Enable the prompt shown on LCD)
SCANNER	USB scanner (Disable the prompt shown on LCD)

Note:

- ***This command is for the model which has USB HOST connector.***

Example

Sample code

```
SET USBHOST KEYBOARD
DOWNLOAD "A.BAS "
LOOP:
SIZE 4,2
GAP 0,0
CLS
INPUT A$
TEXT 50,50, "0 ",0,20,20,A$
PRINT 1
GOTO LOOP
EOP
A.BAS
```

11.23 SET AUTORUN

Description

This command redefines the BAS file which can be run automatically while switching on the printer. Default is AUTO.BAS.

Syntax

SET AUTORUN "filename"

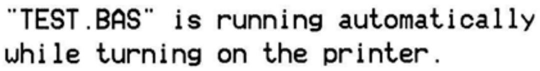
Parameter	Description
filename	The file will be defined to AUTO-RUN file. Default is AUTO.BAS.

Example

Sample Code

```
REM *****Step1: Send the following command to redefine the auto-run file from
"AUTO.BAS" to "TEST.BAS"
SET AUTORUN "TEST.BAS"

REM *****Step2: Send the following commands to download "TEST.BAS" file into printer.
DOWNLOAD F, "TEST.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
BLOCK 10,10,600,200, "3",0,1,1,12, "¥[ "]TEST.BAS¥[ "] is running automatically while
turning on the printer. "
PRINT 1
EOP

REM *****Step3: Turn off and on the printer to run "TEST.BAS" automatically.
```

Result



"TEST.BAS" is running automatically
while turning on the printer.

11.24 SET VERIFIER

Description

This command is used to add a data at start printing or end printing then return it via RS-232 port for verifier (scanner) application.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
SET VERIFIER ON/OFF
VERIFYTRIGGER CHR$()+CHR$()…
VERIFYEND CHR$()+CHR$()…
```

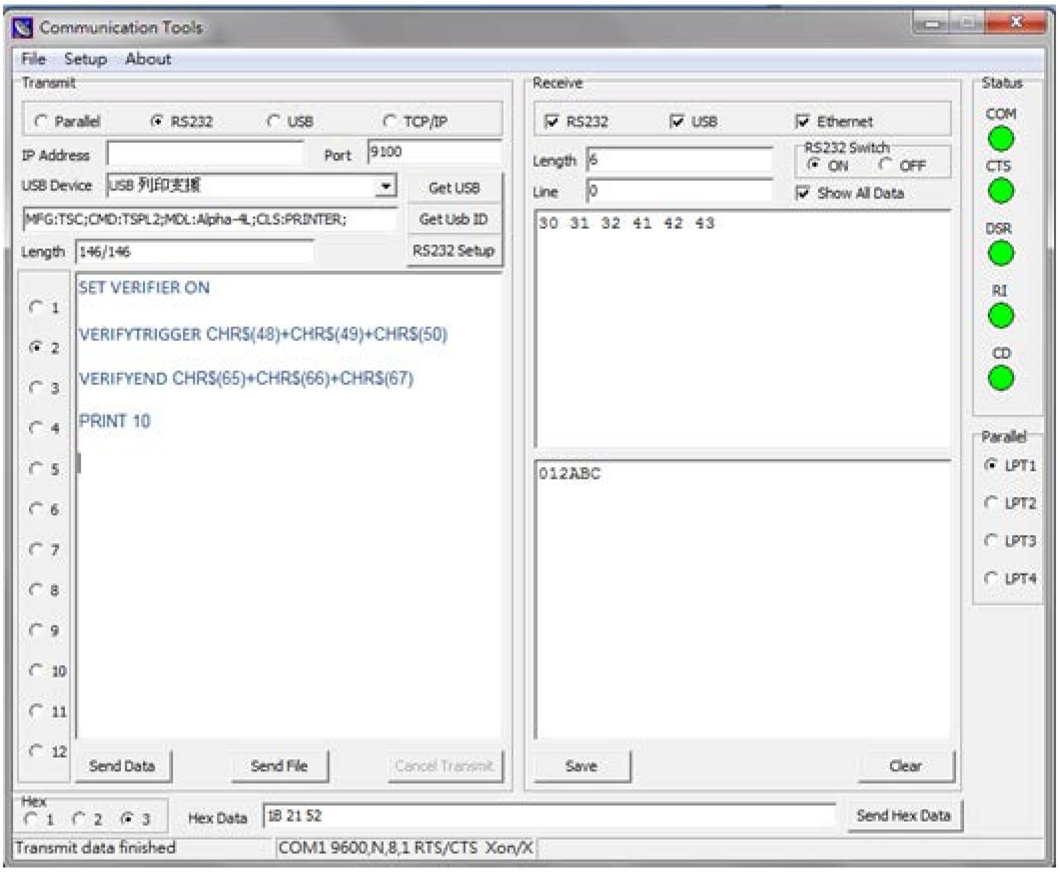
Parameter	Description
ON	Enable verifier function
OFF	Disable verifier function
VERIFYTRIGGER	Add data at start printing
VERIFYEND	Add data at end printing

Example

Sample Code

```
SET VERIFIER ON
VERIFYTRIGGER CHR$(48)+CHR$(49)+CHR$(50)
VERIFYEND CHR$(65)+CHR$(66)+CHR$(67)
PRINT 10
```

Result



Port: 9100
USB Device: USB 列印支援
MFG:TSC;CMD:TSPL2;MDL:Alpha-4L;CLS:PRINTER;
Length: 146/146

```
SET VERIFIER ON
VERIFYTRIGGER CHR$(48)+CHR$(49)+CHR$(50)
VERIFYEND CHR$(65)+CHR$(66)+CHR$(67)
PRINT 10
```

Length: 6
Line: 0

```
30 31 32 41 42 43
```

```
012ABC
```

Hex Data: 1B 21 52

Transmit data finished | COM1 9600,N,8,1 RTS/CTS Xon/X

11.25 SET RESPONSE

Description

This command can response issue automatically.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

SET RESPONSE ["Job ID",] ON/OFF/BATCH

Parameter	Description
["Job ID"]	Optional. Set job ID. Default is Null
ON	Enable this function
OFF	Disable this function. Default is OFF
BATCH	Response at the end of printing job

Response Syntax

{Status,######,ID}

Status

[Hex Receive]
00 Normal
01 Head opened
02 Paper Jam
03 Paper Jam and head opened
04 Out of paper
05 Out of paper and head opened
08 Out of ribbon
09 Out of ribbon and head opened
0A Out of ribbon and paper jam
0B Out of ribbon, paper jam and head opened
0C Out of ribbon and out of paper
0D Out of ribbon, out of paper and head opened
10 Pause
20 Printing
80 Other error
#####: 00001 ~ 99999

Example

Sample Code

```
SET RESPONSE ON
SIZE 4,2
GAP 0,0
PRINT 3

{00,00001}{00,00002}{00,00003}
```

Sample Code

```
SET RESPONSE "ID1",ON
SIZE 4,2
GAP 0,0
PRINT 3,2

{00,00001,ID1}{00,00002,ID1}{00,00003,ID1}{00,00004,ID1}{00,00005,ID1}{00,00006,ID1}
```

Sample Code

```
SET RESPONSE "CCCC ",BATCH
SIZE 4,2
GAP 0,0
PRINT 3,2

{00,00006,CCCC}
```

11.26 SET DAYLIGHT_SAVE

Description

This command is used to set daylight saving time.

Syntax

```
SET DAYLIGHT_SAVE ON/OFF
SET DAYLIGHT_SAVE "Start", "End"
```

Parameter	**Description**
ON	Enable function
OFF	Disable function (Default)
“Start”	The time will be increased 1 hour from "Start time"
“End”	The time will be reduced 1 hour (return) from "End time"
	Month "JAN","FEB","MAR","APR","MAY","JUN","JUL","AUG","SEP","OCT","NOV","DEC" "JANUARY", "FEBRUARY", "MARCH", "APRIL", "MAY", "JUNE", "JULY", "AUGUST", "SEPTEMBER", "OCTOBER", "NOVEMBER", "DECEMBER" Week "SUN","MON","TUE","WED","THU","FRI","SAT" "SUNDAY", "MONDAY", "TUESDAY", "WEDNESDAY", "THURSDAY", "FRIDAY", "SATURDAY" Which Week "FIRST","SECOND", "THIRD","FOURTH","LAST" "1ST", "2ND", "3RD", "4TH", "LAST" Date 1~31 Time 0:00~23:00

Example

Sample Code

```
SET DAYLIGHT_SAVE ON
SET DAYLIGHT_SAVE OFF
SET DAYLIGHT_SAVE "MAR 1 4:00","NOV 1 5:00"
SET DAYLIGHT_SAVE "MAR FIRST SUN 2:00", "NOV LAST SUN 3:00"
```

11.27 PEEL

Description

This command obtains the status of the peel-off sensor. This attribute is read only.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB)

Syntax

```
PEEL
```

Return Value	Description
0	Paper is not on top of peel sensor
1	Paper is on top of peel sensor

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
SET PEEL OFF
SET KEY1 OFF
SET LED1 OFF
SET LED3 OFF
:START
LED1=0
LED3=0
IF KEY1=1 THEN GOTO A
GOTO START
:A
LED1=1
CLS
TEXT 10,10, "3",0,1,1, "PEEL Function Test!! "
PRINT 1,1
:B
LED1=0
IF PEEL=1 THEN
LED3=1
GOTO B
ELSE
CLS
TEXT 10,10, "3",0,1,1, "The label is removed from the PEEL sensor!! "
PRINT 1,1
GOTO START
ENDIF
EOP
DEMO
```

11.28 LED1, LED2, LED3

Description

This command is used to control LED on/off. This attribute is write-only. Specify 1 to light on LED and 0 to turn off LED. Before using this command, be sure to cancel the default LED functions. Please refer to the SET LED command.

Syntax

LED*m* = *n*

Return Value	Description

m	m=1, LED1
	m=2, LED2
	m=3, LED3

n	0: turn off LED 1: light on LED

Model	LED1	LED2	LED3	LED4	LED5	LED6	LED7	LED2 & LED3
TD-4T series **Note: For this series, the LED1=LED2**	GREEN	GREEN	RED					ORANGE

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.12,0
SPEED 4
DENSITY 8
DIRECTION 1
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
SET LED1 OFF
SET LED2 OFF
SET LED3 OFF
LED1=0
LED2=1
LED3=0
EOP
DEMO
```

11.29 KEY1, KEY2, KEY3

Description

This command reads the status of KEY1, KEY2 and KEY3.

Model	KEY0	KEY1	KEY2	KEY3	KEY4	KEY5	KEY6
TD-4T series		FEED					

Syntax

KEY*m* = *n*

Key	Return Value
KEY1 (MENU)	0: released 1: pressed
KEY2 (PAUDE)	0: released 1: pressed
KEY3 (FEED)	0: released 1: pressed

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 3,1
GAP 0,0
SPEED 4
DENSITY 8
DIRECTION 1
REFERENCE 0,0
SET LED1 OFF
SET KEY1 OFF
LED1=0
:START
IF KEY1=1 THEN
LED1=1
CLS
TEXT 100,10, "3",0,1,1, "KEY FUNCTION TEST"
PRINT 1,1
ELSE
LED1=0
ENDIF
GOTO START
EOP
DEMO
```

12 Printer Global Variables

12.1 @LABEL

Description

This variable counts how many pieces of labels have been printed. This attribute cannot be initialized if the printer is reset, but will be retained if the printer power is turned off.

Syntax

Write attribute: @LABEL=n or @LABEL= "n"
Read attribute: A=LABEL or A$=STR$(LABEL)

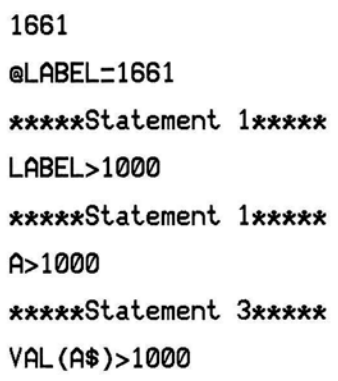
Parameter	Description
n	Number of labels printed. 0<=n<=999999999

Example

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,2.5
GAP 0,0
DIRECTION 1
CLS
TEXT 10,50, "3",0,1,1,@LABEL
TEXT 10,100, "3",0,1,1,
"@LABEL="+STR$(LABEL)
TEXT 10,150, "3",0,1,1, "*****Statement 1*****"
IF LABEL>1000 THEN
TEXT 10,200, "3",0,1,1, "LABEL>1000"
ELSE
TEXT 10,200, "3",0,1,1, "LABEL<1000"
ENDIF
TEXT 10,250, "3",0,1,1, "*****Statement 1*****"
A=LABEL
IF A>1000 THEN
TEXT 10,300, "3",0,1,1, "A>1000"
ELSE
TEXT 10,300, "3",0,1,1, "A<1000"
ENDIF
TEXT 10,350, "3",0,1,1, "*****Statement 3*****"
A$=STR$(LABEL)
IF VAL(A$)>1000 THEN
TEXT 10,400, "3",0,1,1, "VAL(A$)>1000"
ELSE
TEXT 10,400, "3",0,1,1, "VAL(A$)<1000"
ENDIF
PRINT 1,1
EOP
DEMO
```

Result



```
1661
@LABEL=1661
*****Statement 1*****
LABEL>1000
*****Statement 1*****
A>1000
*****Statement 3*****
VAL(A$)>1000
```

12.2 YEAR

Description

This variable reads/writes the year data via the Real Time Clock (RTC). Four-digit year formats are supported by RTC.

Syntax

Write attribute: YEAR = 02
Read attribute: A = YEAR
Range: 00~50 = 2000~2050; 51~99 = 1951~1999

Example

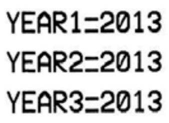
Sample code

```
DOWNLOAD "SetYear.BAS"
REM *****Set Year Parameter to RTC*****
YEAR=13
EOP
SetYear
```

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS

REM *****Read YEAR parameter from RTC*****
YEAR$=STR$(YEAR)
Y=YEAR

REM *****Print*****
TEXT 10,10, "3",0,1,1, "YEAR1="+YEAR$
TEXT 10,50, "3",0,1,1, "YEAR2="+STR$(Y)
TEXT 10,90, "3",0,1,1, "YEAR3="+STR$(YEAR)
PRINT 1
EOP
DEMO
```

Result



```
YEAR1=2013
YEAR2=2013
YEAR3=2013
```

See Also

~!C, MONTH, DATE, DAY, HOUR, MINUTE, SECOND

12.3 MONTH

Description

This variable reads/writes the month data via the Real Time Clock (RTC). Two-digit (01~12) month formats are supported by RTC.

Syntax

Write attribute: MONTH = 01
Read attribute: A = MONTH
Range: 01~12

Example

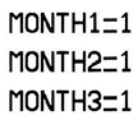
Sample code

```
DOWNLOAD "SetMonth.BAS"
REM *****Set Month Parameter to RTC*****
MONTH=01
EOP
SetMonth
```

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS

REM *****Read Month parameter form RTC*****
MONTH$=STR$(MONTH)
M=MONTH

REM *****Print*****
TEXT 10,10, "3",0,1,1, "MONTH1="+MONTH$
TEXT 10,50, "3",0,1,1, "MONTH2="+STR$(M)
TEXT 10,90, "3",0,1,1, "MONTH3="+STR$(MONTH)
PRINT 1
EOP
DEMO
```

Result



```
MONTH1=1
MONTH2=1
MONTH3=1
```

See Also

~!C, MONTH, DATE, DAY, HOUR, MINUTE, SECOND

12.4 DATE

Description

This variable reads/writes the date data via the Real Time Clock (RTC). Two-digit (01~31) date formats are supported by RTC.

Syntax

Write attribute: DATE = 12
Read attribute: A = DATE
Range: 01~31

Example

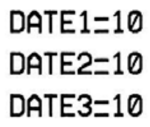
Sample code

```
DOWNLOAD "SetDate.BAS"
REM *****Set Date Parameter to RTC*****
DATE=10
EOP
SetDate
```

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS

REM *****Read Date parameter form RTC*****
DATE$=STR$(DATE)
D=DATE

REM *****Print*****
TEXT 10,10, "3",0,1,1, "DATE1="+DATE$
TEXT 10,50, "3",0,1,1, "DATE2="+STR$(D)
TEXT 10,90, "3",0,1,1, "DATE3="+STR$(DATE)
PRINT 1
EOP
DEMO
```

Result



```
DATE1=10
DATE2=10
DATE3=10
```

See Also

~!C, MONTH, DATE, DAY, HOUR, MINUTE, SECOND

12.5 WEEK

Description

This variable reads/writes the day of the week data via the Real Time Clock (RTC), which is represented by one single digit (1~7).

Syntax

Read attribute: A = WEEK

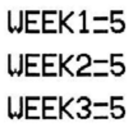
Range: 1(Sunday)~7(Saturday)

Example

Sample code

```
DOWNLOAD "DEMO.BAS "
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS

REM *****Read Week parameter form RTC*****
WEEK$=STR$(WEEK)
W=WEEK

REM *****Print*****
TEXT 10,10, "3",0,1,1, "WEEK1="+WEEK$
TEXT 10,50, "3",0,1,1, "WEEK2="+STR$(W)
TEXT 10,90, "3",0,1,1, "WEEK3="+STR$(WEEK)
PRINT 1
EOP
DEMO
```

Result



```
WEEK1=5
WEEK2=5
WEEK3=5
```

See Also

~!C, MONTH, DATE, DAY, HOUR, MINUTE, SECOND

12.6 HOUR

Description

This variable reads/writes the hour data via the Real Time Clock (RTC). The 24-hour-day system (00~23) is supported by RTC.

Syntax

Write attribute: HOUR = 12
Read attribute: A = HOUR
Range: 00~23

Example

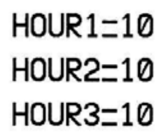
Sample code

```
DOWNLOAD "SetHour.BAS"
REM *****Set Hour Parameter to RTC*****
HOUR=10
EOP
SetHour
```

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS

REM *****Read Hour parameter form RTC*****
HOUR$=STR$(HOUR)
H=HOUR

REM *****Print*****
TEXT 10,10, "3",0,1,1, "HOUR1="+HOUR$
TEXT 10,50, "3",0,1,1, "HOUR2="+STR$(H)
TEXT 10,90, "3",0,1,1, "HOUR3="+STR$(HOUR)
PRINT 1
EOP
DEMO
```

Result



```
HOUR1=10
HOUR2=10
HOUR3=10
```

See Also

~!C, MONTH, DATE, DAY, HOUR, MINUTE, SECOND

12.7 MINUTE

Description

This variable reads/writes the minute data via the Real Time Clock (RTC). Two-digits (00~59) minute format is supported by RTC.

Syntax

Write attribute: MINUTE = 12
Read attribute: A = MINUTE
Range: 00~59

Example

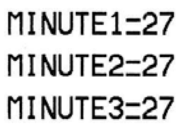
Sample code

```
DOWNLOAD "SetMinute.BAS"
REM *****Set Minute Parameter to RTC*****
MINUTE=27
EOP
SetMinute
```

Sample code

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS

REM *****Read Minute parameter form RTC*****
MINUTE$=STR$(MINUTE)
MIN=MINUTE

REM *****Print*****
TEXT 10,10, "3",0,1,1, "MINUTE1="+MINUTE$
TEXT 10,50, "3",0,1,1, "MINUTE2="+STR$(MIN)
TEXT 10,90, "3",0,1,1, "MINUTE3="+STR$(MINUTE)
PRINT 1
EOP
DEMO
```

Result



```
MINUTE1=27
MINUTE2=27
MINUTE3=27
```

See Also

~!C, MONTH, DATE, DAY, HOUR, MINUTE, SECOND

12.8 SECOND

Description

This variable reads/writes the second data via the Real Time Clock (RTC). Two-digits (00~59) second format is supported by RTC.

Syntax

Write attribute: SECOND = 12
Read attribute: A = SECOND
Range: 00~59

Example

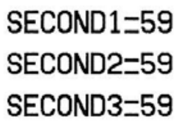
Sample code

```
DOWNLOAD "SetSecond.BAS"
REM *****Set Second Parameter to RTC*****
SECOND=59
EOP
SetSecond
```

Sample code

```
DOWNLOAD "DEMO.BAS "
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS

REM *****Read Second parameter form RTC*****
SECOND$=STR$(SECOND)
SEC=SECOND

REM *****Print*****
TEXT 10,10, "3",0,1,1, "SECOND1="+SECOND$
TEXT 10,50, "3",0,1,1, "SECOND2="+STR$(SEC)
TEXT 10,90, "3",0,1,1, "SECOND3="+STR$(SECOND)
PRINT 1
EOP
DEMO
```

Result



```
SECOND1=59
SECOND2=59
SECOND3=59
```

See Also

~!C, MONTH, DATE, DAY, HOUR, MINUTE, SECOND

12.9 @YEAR

Description

This variable reads/writes the year data via the Real Time Clock (RTC). Two-digit year formats are supported by RTC. @YEAR global variable can be accessed directly without using BASIC language functions.

Syntax

Write attribute: @YEAR = "01"
Read attribute: @YEAR
Range: 00~99

Example

Sample code	Result
REM *****Set @YEAR***** **@YEAR="05"** **REM *****Print******* **SIZE 4,1** **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 10,10, "3",0,1,1, "@YEAR"** **TEXT 210,10, "3",0,1,1, @YEAR** **PRINT 1**	 @YEAR 2005

See Also

~!C, @MONTH, @DATE, @DAY, @HOUR, @MINUTE, @SECOND

12.10 @MONTH

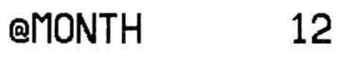
Description

This variable reads/writes the month data via the Real Time Clock (RTC). Two-digits (01~12) month formats are supported by RTC. @MONTH global variable can be accessed directly without using BASIC language functions.

Syntax

Write attribute: @MONTH = "01"
Read attribute: @MONTH
Range: 01~12

Example

Sample code	**Result**
REM *****Set @MONTH***** **@MONTH="12"** **REM *****Print******* **SIZE 4,1** **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 10,10, "3",0,1,1, "@MONTH"** **TEXT 210,10, "3",0,1,1,@MONTH** **PRINT 1**	 @MONTH 12

See Also

~!C, @YEAR, @DATE, @DAY, @HOUR, @MINUTE, @SECOND

12.11 @DATE

Description

This variable reads/writes the date data via the Real Time Clock (RTC). Two-digits (01~31) date formats are supported by RTC. @DATE global variable can be accessed directly without using BASIC language functions.

Syntax

Write attribute: @DATE = "12"
Read attribute: @DATE
Range: 01~31

Example

Sample code	Result
REM *****Set @DATE***** @DATE="31" REM *****Print***** SIZE 4,1 GAP 0,0 DIRECTION 1 CLS TEXT 10,10, "3",0,1,1, "@DATE" TEXT 210,10, "3",0,1,1,@DATE PRINT 1	 @DATE 31

See Also

~!C, @YEAR, @MONTH, @DAY, @HOUR, @MINUTE, @SECOND

12.12 @DAY

Description

This variable reads/writes the day of the week data via the Real Time Clock (RTC), which is represented by one single digit (1~7). @DAY global variable can be accessed directly without using BASIC language functions.

Syntax

Read attribute: @DAY
Range: 1(Sunday)~7(Saturday)

Example

Sample code	**Result**
REM *****Print***** **SIZE 4,1** **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 10,10, "3",0,1,1, "@DAY"** **TEXT 210,10, "3",0,1,1,@DAY** **PRINT 1**	 @DAY 7

See Also

~!C, @YEAR, @MONTH, @DATE, @HOUR, @MINUTE, @SECOND

12.13 @HOUR

Description

This variable reads/writes the hour data via the Real Time Clock (RTC). The 24-hour-day system (00~23) is supported by RTC. @HOUR global variable can be accessed directly without using BASIC language functions.

Syntax

Write attribute: @HOUR = "12"
Read attribute: @HOUR
Range: 00~23

Example

Sample code	Result
REM *****Set @HOUR***** **@HOUR="23"** **REM *****Print******* **SIZE 4,1** **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 10,10, "3",0,1,1, "@HOUR"** **TEXT 210,10, "3",0,1,1,@HOUR** **PRINT 1**	 @HOUR 23

See Also

~!C, @YEAR, @MONTH, @DATE, @DAY, @MINUTE, @SECOND

12.14 @MINUTE

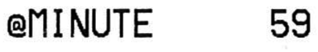
Description

This variable reads/writes the minute data via the Real Time Clock (RTC). The two-digits (00~59) minute format is supported by RTC. @MINUTE global variable can be accessed directly without using BASIC language functions.

Syntax

Write attribute: @MINUTE = "12"
Read attribute: @MINUTE
Range: 00~59

Example

Sample code	**Result**
REM *****Set @MINUTE***** **@MINUTE="59"** **REM *****Print******* **SIZE 4,1** **GAP 0,0** **DIRECTION 1** **CLS** **TEXT 10,10, "3",0,1,1, "@MINUTE"** **TEXT 210,10, "3",0,1,1,@MINUTE** **PRINT 1**	 @MINUTE 59

See Also

~!C, @YEAR, @MONTH, @DATE, @DAY, @HOUR, @SECOND

12.15 @SECOND

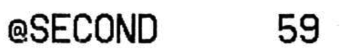
Description

This variable reads/writes the second data via the Real Time Clock (RTC). The Two-digit (00~59) second format is supported by RTC. @SECOND global variable can be accessed directly without using BASIC language functions.

Syntax

Write attribute: @SECOND = "12"
Read attribute: @SECOND
Range: 00~59

Example

Sample code	Result
REM *****Set @SECOND***** @SECOND = "59" REM *****Print***** SIZE 4,1 GAP 0,0 DIRECTION 1 CLS TEXT 10,10, "3",0,1,1, "@SECOND" TEXT 210,10, "3",0,1,1,@SECOND PRINT 1	 @SECOND 59

See Also

~!C, @YEAR, @MONTH, @DATE, @DAY, @HOUR, @MINUTE

12.16 _MODEL$

Description

This variable can be read only. It includes the information of printer’s model name.

Syntax

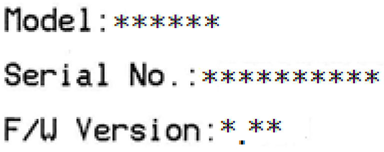
_MODEL$

Example

Sample code

```
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "Model: " + _MODEL$
TEXT 10,60, "3",0,1,1, "Serial No.: " + _SERIAL$
TEXT 10,110, "3",0,1,1, "F/W Version: " + _VERSION$
PRINT 1
```

Result



```
Model:******
Serial No.:***********
F/W Version:*.**
```

See Also

_SERIAL$, _VERSION$

12.17 _SERIAL$

Description

This variable can be read only. It includes the information of printer's serial number.
The printer's serial number must be programmed into printer at factory.

Syntax

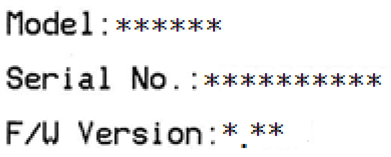
```
_SERIAL$
```

Example

Sample code

```
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "Model: " + _MODEL$
TEXT 10,60, "3",0,1,1, "Serial No.: " + _SERIAL$
TEXT 10,110, "3",0,1,1, "F/W Version: " + _VERSION$
PRINT 1
```

Result



```
Model:******
Serial No.:**********
F/W Version:*.**
```

See Also

_MODEL$, _VERSION$

12.18 _VERSION$

Description

This variable can be read only. It includes the information of printer's firmware version.

Syntax

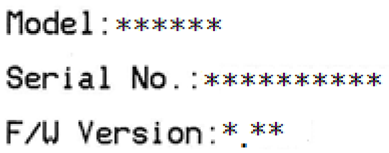
_SERIAL$

Example

Sample code

```
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
TEXT 10,10, "3",0,1,1, "Model: " + _MODEL$
TEXT 10,60, "3",0,1,1, "Serial No.: " + _SERIAL$
TEXT 10,110, "3",0,1,1, "F/W Version: " + _VERSION$
PRINT 1
```

Result



```
Model:******
Serial No.:**********
F/W Version:*.**
```

See Also

_MODEL$, _VERSION$

13 Wi-Fi Module Setting Commands

13.1 WLAN OFF

Description

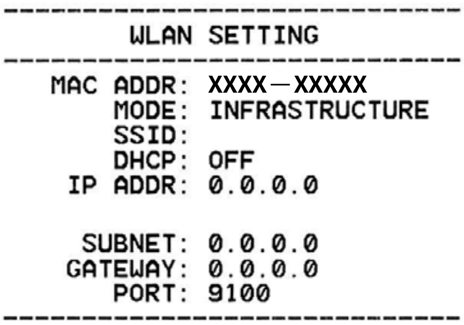
This command is used to disable Wi-Fi module. Restart the printer is necessary.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
WLAN OFF
```

Note:
We can use command SELFTEST WLAN to check the status of WLAN. The SSID is empty.

Example

Sample code	Result
WLAN OFF SELFTEST WLAN	

```
------------------------------
         WLAN SETTING
------------------------------
   MAC ADDR: XXXX－XXXXX
       MODE: INFRASTRUCTURE
       SSID:
       DHCP: OFF
    IP ADDR: 0.0.0.0

     SUBNET: 0.0.0.0
    GATEWAY: 0.0.0.0
       PORT: 9100
------------------------------
```

See Also

WLAN SSID

13.2 WLAN SSID

Description

This command is used to set the SSID of your wireless network into Wi-Fi module. Restart the printer is necessary.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

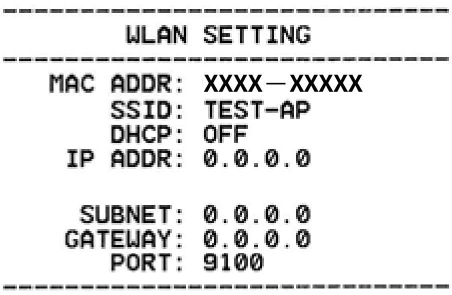
Syntax

WLAN SSID "ssid"

Parameter	Description
ssid	It is the SSID of your wireless network.

Note:
SSID is case-sensitive. The maximum length is 32 bytes.

Example

Sample code	Result
WLAN SSID "TEST-AP" **SELFTEST WLAN**	 ------------------------------ WLAN SETTING ------------------------------ MAC ADDR: XXXX—XXXXX SSID: TEST-AP DHCP: OFF IP ADDR: 0.0.0.0 SUBNET: 0.0.0.0 GATEWAY: 0.0.0.0 PORT: 9100 ------------------------------

See Also

WLAN OFF

13.3 WLAN WPA

Description

This command is used to set WPA security mode. This command only can be set but not be checked. Restart the printer is necessary.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
WLAN WPA OFF
WLAN WPA "key"
```

Parameter	Description
OFF	Disable WPA security mode.
Key	The network security key. 8 to 63 characters. Key = Passphrase or Pre-Shared Key (Passphrase is a string containing between 8 and 63 characters) (Pre-Shared Key is a 32-byte key, formatted as hexadecimal number)

Example

Sample code

```
WLAN WPA OFF
WLAN WPA "123456789"
```

13.4 WLAN WEP

Description

This command is used to set WEP security mode. This command only can be set but not be checked. Restart the printer is necessary.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
WLAN WEP OFF
WLAN WEP n, "key"
```

Parameter	Description
OFF	Disable WEP security mode.
N	The index of key. 1 to 4.
Key	The encryption key. 5 or 13 characters or 10 or 26 hexadecimal digits.

Example

Sample code

```
WLAN WEP OFF
WLAN WEP 1, "ABCDE"
WLAN WEP 2, "ABCDE"
WLAN WEP 3, "ABCDE"
WLAN WEP 4, "4142434445"
```

13.5 WLAN DHCP

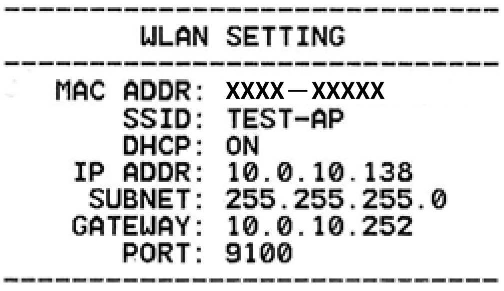
Description

This command is used to set the printer to get the IP address from DHCP server. Restart the printer is necessary.
(Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

WLAN DHCP

Example

Sample code	Result
WLAN SSID "TEST-AP" **WLAN WPA "123456789"** **WLAN DHCP** **WLAN PORT 9100** **SELFTEST WLAN**	 ------------------------------ WLAN SETTING ------------------------------ MAC ADDR: XXXX—XXXXX SSID: TEST-AP DHCP: ON IP ADDR: 10.0.10.138 SUBNET: 255.255.255.0 GATEWAY: 10.0.10.252 PORT: 9100 ------------------------------

See Also

WLAN IP

13.6 WLAN IP

Description

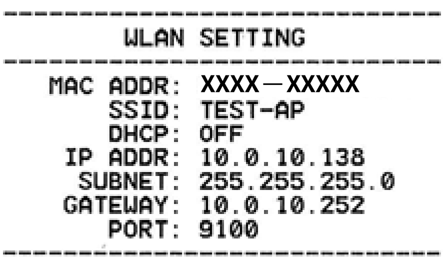
This command is used to set the specific static IP address to printer. Restart the printer is necessary. (Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

WLAN IP "ip", "mask", "gateway"

Parameter	Description
ip	IP address.
Mask	Subnet mask.
Gateway	Default gateway.

Example

Sample code	Result
WLAN SSID "TEST-AP" **WLAN WPA "123456789"** **WLAN IP "10.0.10.138", "255.255.255.0", "10.0.10.252"** **WLAN PORT 9100** **SELFTEST WLAN**	 ----------------------------- WLAN SETTING ----------------------------- MAC ADDR: XXXX—XXXXX SSID: TEST-AP DHCP: OFF IP ADDR: 10.0.10.138 SUBNET: 255.255.255.0 GATEWAY: 10.0.10.252 PORT: 9100 -----------------------------

See Also

WLAN DHCP

13.7 WLAN PORT

Description

This command is used to specify the PORT number of Wi-Fi module. Restart the printer is necessary. (Available for TD-4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

```
WLAN PORT number
```

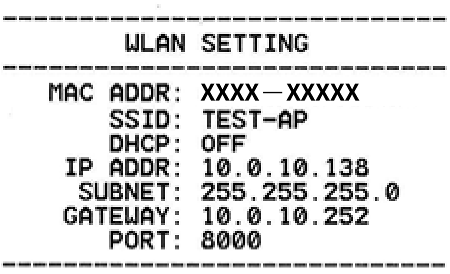
Parameter	Description
number	Base raw port number. Default is 9100.

Example

Sample code

```
WLAN SSID "TEST-AP"
WLAN WPA "123456789"
WLAN IP "10.0.10.138", "255.255.255.0","10.0.10.252"
WLAN PORT 8000
SELFTEST WLAN
```

Result



```
-----------------------------
         WLAN SETTING
-----------------------------
   MAC ADDR: XXXX—XXXXX
       SSID: TEST-AP
       DHCP: OFF
    IP ADDR: 10.0.10.138
     SUBNET: 255.255.255.0
    GATEWAY: 10.0.10.252
       PORT: 8000
-----------------------------
```

14 Internal Ethernet Setting Commands

14.1 NET DHCP

Description

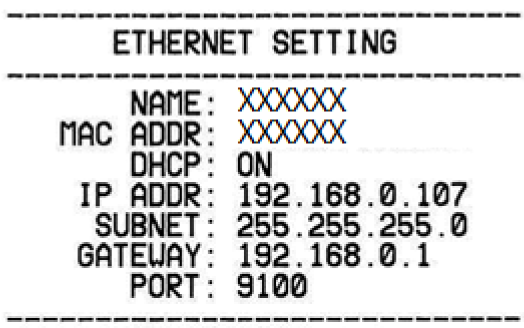
This command is used to set the printer to get the IP address from DHCP server. Printer will restart itself while setting this command.

(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

NET DHCP

Example

Sample code	Result
NET DHCP SELFTEST ETHERNET	 ----------------------------- ETHERNET SETTING ----------------------------- NAME: XXXXXX MAC ADDR: XXXXXX DHCP: ON IP ADDR: 192.168.0.107 SUBNET: 255.255.255.0 GATEWAY: 192.168.0.1 PORT: 9100 -----------------------------

See Also

NET IP

14.2 NET IP

Description

This command is used to set the specific IP address to printer. Printer will restart itself while setting this command.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

NET IP "ip","mask","gateway"

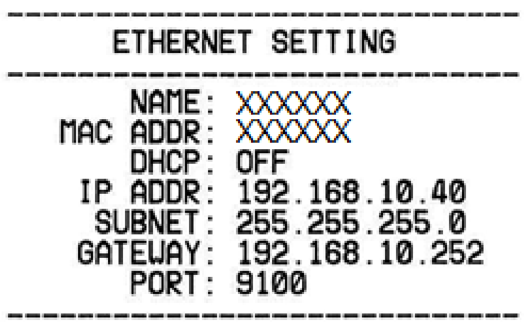
Parameter	Description
ip	IP address
mask	Subnet mask
gateway	Default gateway

Example

Sample code

```
NET IP "192.168.10.40","255.255.255.0","192.168.10.252"
SELFTEST ETHERNET
```

Result



```
------------------------------
      ETHERNET SETTING
------------------------------
       NAME: XXXXXX
   MAC ADDR: XXXXXX
       DHCP: OFF
    IP ADDR: 192.168.10.40
     SUBNET: 255.255.255.0
    GATEWAY: 192.168.10.252
       PORT: 9100
------------------------------
```

See Also

NET DHCP

14.3 NET PORT

Description

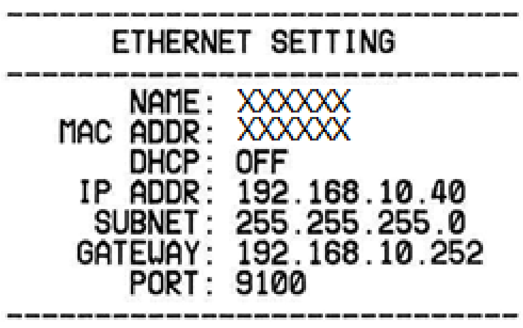
This command is used to specify the PORT number of Ethernet. Printer will restart itself while setting this command.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

NET PORT number

Parameter	Description
number	Base raw port number. Default is 9100.

Example

Sample code	Result
NET PORT 9100 SELFTEST ETHERNET	 ------------------------------ ETHERNET SETTING ------------------------------ NAME: XXXXXX MAC ADDR: XXXXXX DHCP: OFF IP ADDR: 192.168.10.40 SUBNET: 255.255.255.0 GATEWAY: 192.168.10.252 PORT: 9100 ------------------------------

14.4 NET NAME

Description

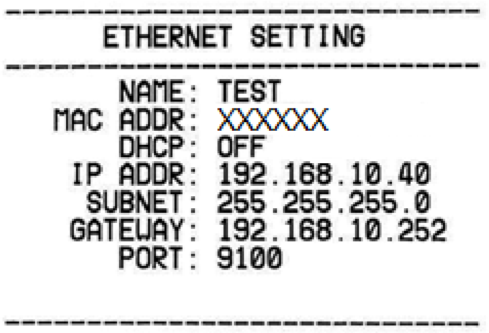
This command is used to set the printer server name.
(Available for TD-4420TN/4520TN/4650TNWB/4750TNWB/4650TNWBR/4750TNWBR)

Syntax

NET NAME "printerserver"

Parameter	Description
printerserver	The specific name of printer server.

Example

Sample code	Result
NET NAME "TEST" **SELFTEST ETHERNET**	 ETHERNET SETTING NAME: TEST MAC ADDR: XXXXXX DHCP: OFF IP ADDR: 192.168.10.40 SUBNET: 255.255.255.0 GATEWAY: 192.168.10.252 PORT: 9100

15 RFID READ

15.1 RFID WRITE

Description

This command reads or writes to an RFID tag.

IMPORTANT:
Make sure the label length matches the physical length of the installed media.

Syntax

RFID READ,A,B,C,D,E,"Read Data:"
RFID WRITE,A,B,C,D,E,data

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>RFID <b>READ</b>
RFID <b>WRITE</b></td><td><b>READ</b> = read the tag
<b>WRITE</b> = write to the tag</td></tr>
<tr><td>A</td><td><table>
<tr><td><b>READ</b> only</td><td>unlock password</td><td>0 = read without unlock.
1 to FFFFFFFF in hex = read and unlock the data block so it can be overwritten later.</td></tr>
<tr><td><b>WRITE</b> only</td><td>lock password</td><td>0 = write without lock.
1 to FFFFFFFF in hex = write and lock the data block to prevent it from being overwritten.</td></tr>
</table></td></tr>
<tr><td>B</td><td>A letter specifying the representation format of the field data.
A = ASCII
B = Binary
D = decimal
H = Hex</td></tr>
<tr><td>C
Start block</td><td>Specifies the number of blocks to read. This option is valid only for the read operation. (0 to FFFF in hex)</td></tr>
<tr><td>D
Size</td><td>Read /Write data size from 1 to n in decimal number.

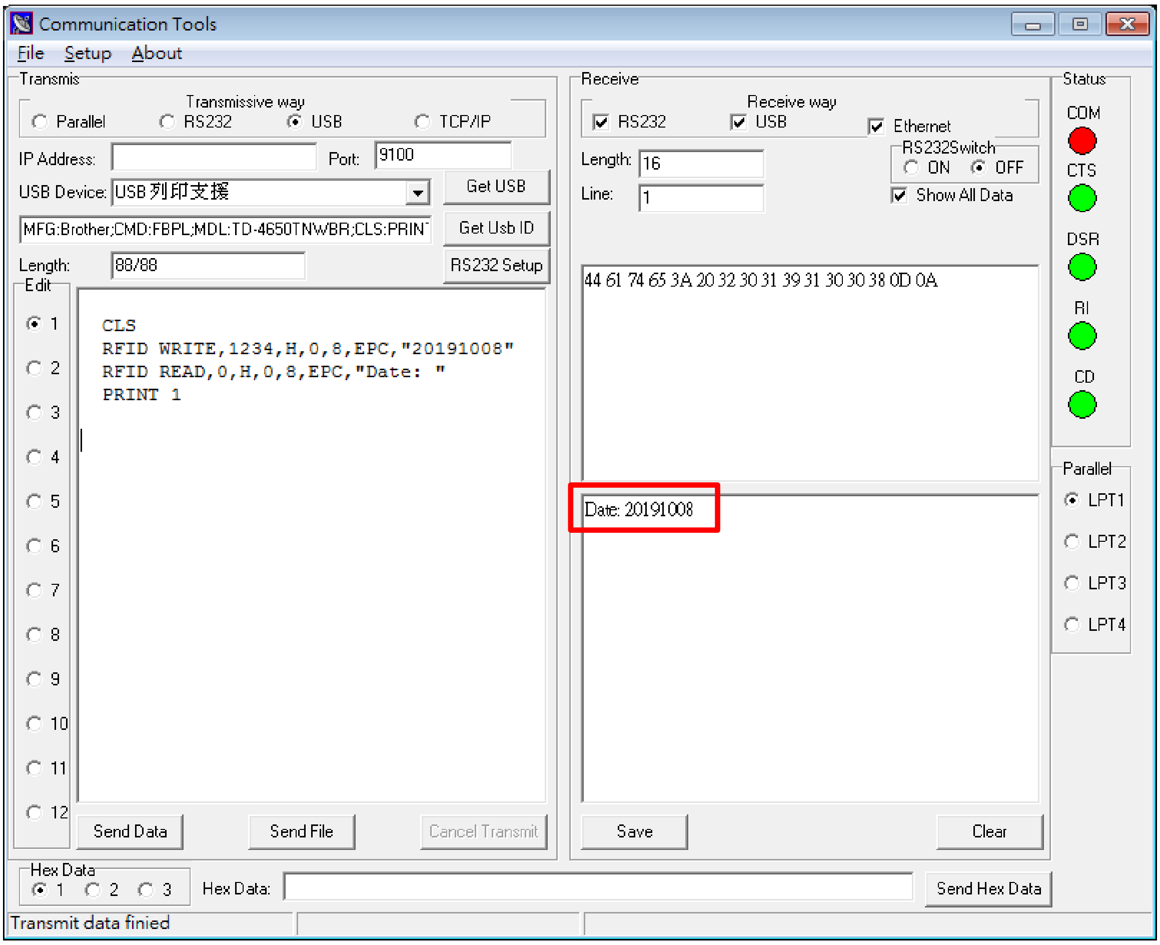
<b>Note:</b>
- <b>When using WRITE, if the "size" is larger than the "data", it will be padded with 0 in front of the data to read.</b>
- <b>When using READ, if the "size" is larger than the WRITE "data", it will be padded with 0 in back of the data to read.</b>
- <b>Refer to the example 3 of sample code.</b></td></tr>
<tr><td>E
Memory bank</td><td>EPC - EPC 12 bytes data area
TID - Tag identification 8 bytes area (currently not applicable for RFID WRITE)
USR - User 32 bytes area
ACS - 4 bytes access code area
KIL - 4 bytes kill code area
PC - 2 bytes PC code area (Gen 2 tags only)</td></tr>
<tr><td><b>"Read Data:"</b>
or
<b>data</b></td><td><b>READ</b> = [prompt of data]
<b>WRITE</b> = content of data string

<b>Note:</b>
- <b>RFID WRITE supported "string" or basic variable (e.g. VAR$)</b>
- <b>[ ] = Optional parameter</b></td></tr>
</table>

Example

Sample code (1)

```
CLS
RFID WRITE,0,H,0,12,EPC,"123456789012"
RFID READ,0,H,0,12,EPC,"Read Data: "
PRINT 1
```



Transmissive way: ○ Parallel ○ RS232 ● USB ○ TCP/IP

IP Address: Port: 9100

USB Device: USB 列印支援

MFG:Brother;CMD:FBPL;MDL:TD-4650TNWBR;CLS:PRIN

Length: 88/88

```
CLS
RFID WRITE,1234,H,0,8,EPC,"20191008"
RFID READ,0,H,0,8,EPC,"Date: "
PRINT 1
```

Receive way: ☑ RS232 ☑ USB ☑ Ethernet

RS232Switch: ○ ON ● OFF

Length: 16

Line: 1

☑ Show All Data

```
44 61 74 65 3A 20 32 30 31 39 31 30 30 38 0D 0A
```

```
Date: 20191008
```

Parallel: ● LPT1 ○ LPT2 ○ LPT3 ○ LPT4

Hex Data: ● 1 ○ 2 ○ 3 Hex Data:

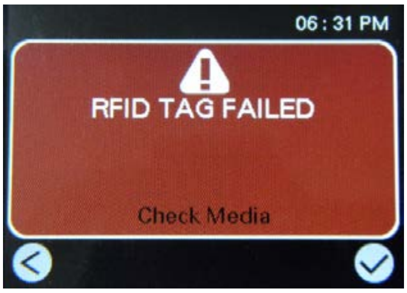
Transmit data finied

Sample code (2)

This programming example writes a data with lock password into an RFID tag and reads the written data with a prompt.

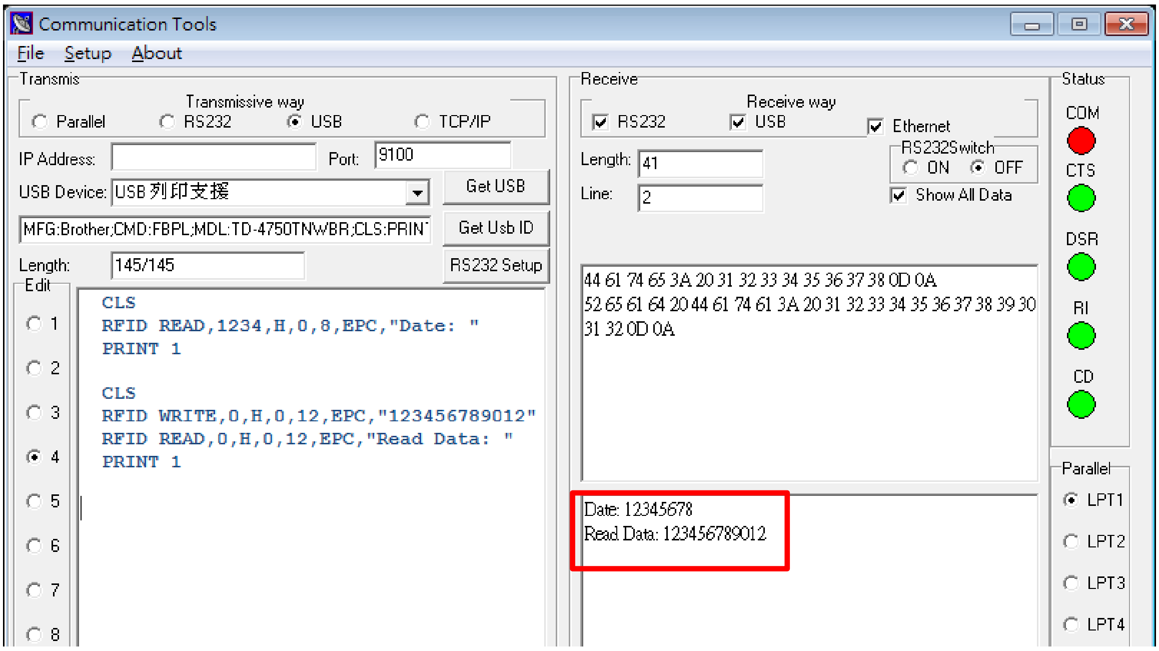
```
CLS
RFID WRITE,1234,H,0,8,EPC,"20191008"
RFID READ,0,H,0,8,EPC,"Date: "
PRINT 1
```

Note:
For this locked RFID tag, it cannot be overwritten data without using RFID READ unlock password command. If you re-send the RFID WRITE command, the printer LCD will be shown as below,



If you need to overwrite this locked tag, please use RFID READ unlock command as following programming example, to unlock password for the RFID tag so it can be overwritten later.

```
CLS
RFID READ,1234,H,0,8,EPC,"Date: "
PRINT 1

CLS
RFID WRITE,0,H,0,12,EPC,"123456789012"
RFID READ,0,H,0,12,EPC,"Read Data: "
PRINT 1
```



Transmissive way: ○ Parallel ○ RS232 ● USB ○ TCP/IP

IP Address: Port: 9100

USB Device: USB 列印支援

MFG:Brother;CMD:FBPL;MDL:TD-4750TNWBR;CLS:PRIN

Length: 145/145

Edit: ○ 1 ○ 2 ○ 3 ● 4 ○ 5 ○ 6 ○ 7 ○ 8

```
CLS
RFID READ,1234,H,0,8,EPC,"Date: "
PRINT 1

CLS
RFID WRITE,0,H,0,12,EPC,"123456789012"
RFID READ,0,H,0,12,EPC,"Read Data: "
PRINT 1
```

Receive way: ☑ RS232 ☑ USB ☑ Ethernet

RS232Switch: ○ ON ● OFF

☑ Show All Data

Length: 41

Line: 2

```
44 61 74 65 3A 20 31 32 33 34 35 36 37 38 0D 0A
52 65 61 64 20 44 61 74 61 3A 20 31 32 33 34 35 36 37 38 39 30
31 32 0D 0A
```

```
Date: 12345678
Read Data: 123456789012
```

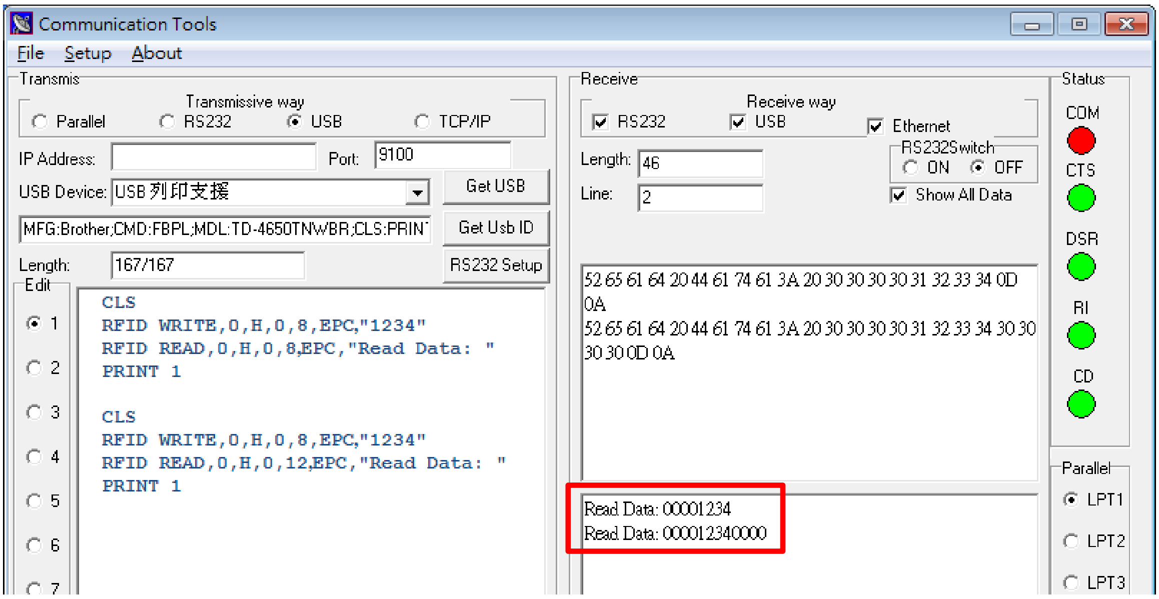
Status: COM 🟥 CTS 🟩 DSR 🟩 RI 🟩 CD 🟩

Parallel: ● LPT1 ○ LPT2 ○ LPT3 ○ LPT4

Sample code (3)

When using WRITE, if the "size" is larger than the "data", it will be padded with 0 in front of the data to read. When using READ, if the "size" is larger than the WRITE "data", it will be padded with 0 in back of the data to read.

```
CLS
RFID WRITE,0,H,0,8,EPC,"1234"
RFID READ,0,H,0,8,EPC,"Read Data: "
PRINT 1

CLS
RFID WRITE,0,H,0,8,EPC,"1234"
RFID READ,0,H,0,12,EPC,"Read Data: "
PRINT 1
```



Transmissive way: ○ Parallel ○ RS232 ● USB ○ TCP/IP

IP Address: Port: 9100

USB Device: USB 列印支援

MFG:Brother;CMD:FBPL;MDL:TD-4650TNWBR;CLS:PRIN

Length: 167/167

Edit: ● 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6 ○ 7

```
CLS
RFID WRITE,0,H,0,8,EPC,"1234"
RFID READ,0,H,0,8,EPC,"Read Data: "
PRINT 1

CLS
RFID WRITE,0,H,0,8,EPC,"1234"
RFID READ,0,H,0,12,EPC,"Read Data: "
PRINT 1
```

Receive way: ☑ RS232 ☑ USB ☑ Ethernet

RS232Switch: ○ ON ● OFF

☑ Show All Data

Length: 46

Line: 2

```
52 65 61 64 20 44 61 74 61 3A 20 30 30 30 30 31 32 33 34 0D
0A
52 65 61 64 20 44 61 74 61 3A 20 30 30 30 30 31 32 33 34 30 30
30 30 0D 0A
```

```
Read Data: 00001234
Read Data: 000012340000
```

Parallel: ● LPT1 ○ LPT2 ○ LPT3

15.2 RFIDDETECT

Description

This command performs RFID calibration.

brother