BBP11-24L / BBP11-34L

THERMAL TRANSFER BAR CODE PRINTER

TSPL2 PROGRAMMING LANGUAGE MANUAL

BBP11-24L / BBP11-34L
Bar Code Printer Programming Manual

Update History

Date	Content	Editor
2007/7/13	Revise some typos	Phil
2007/12/25	Revise FREAD$() example	Camille
2008/4/10	Add update history list	Camille
2009/1/17	Add GAPDETECT, WRITE, LTRIM$(), RTRIM$(), TRIM$(), INSTR(), INPUTFILTER, INPUTPREFIX and INPUTSUFFIX commands.	Ken
2009/2/11	Brady changes	MVP

BBP11-24L / BBP11-34L
Bar Code Printer Programming Manual

BRADY
WHEN PERFORMANCE MATTERS MOST™

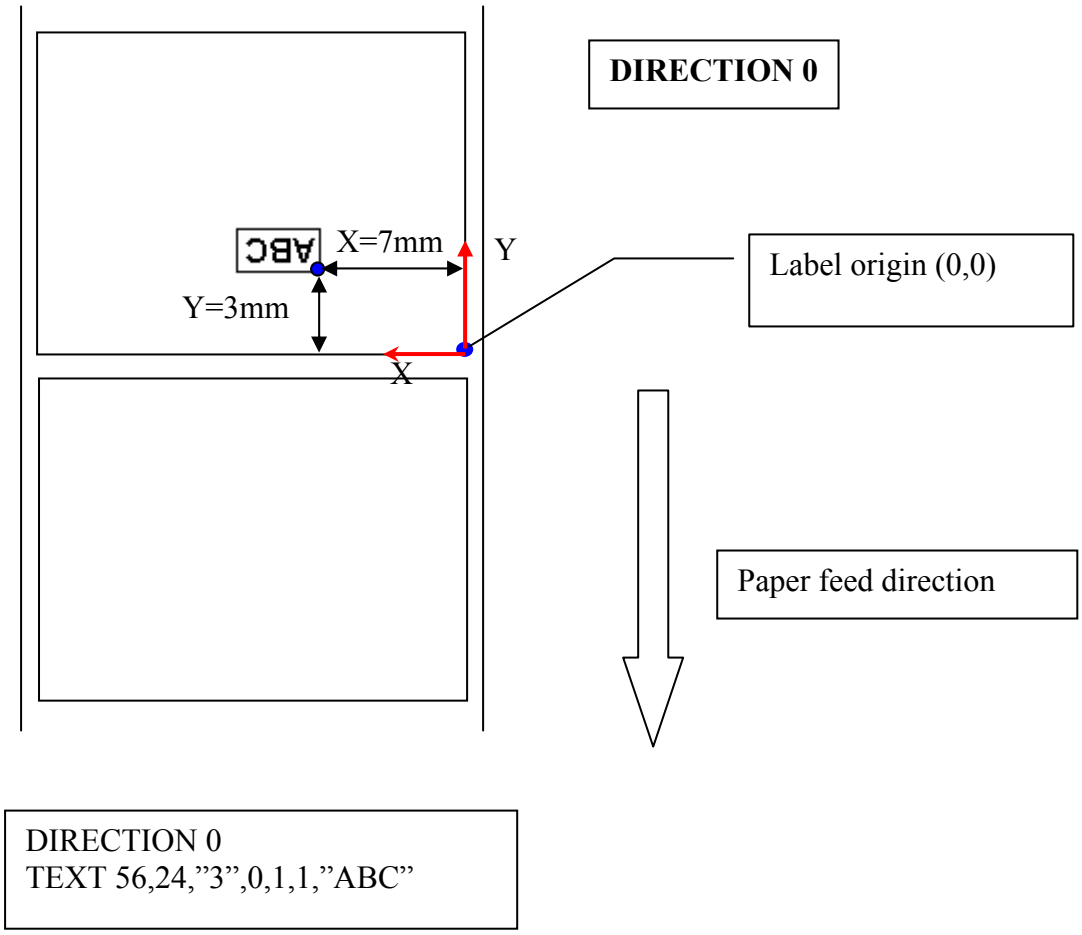
TABLE OF CONTENTS

Document Conventions{ TC "Document Conventions" }

This manual uses the following typographic conventions.

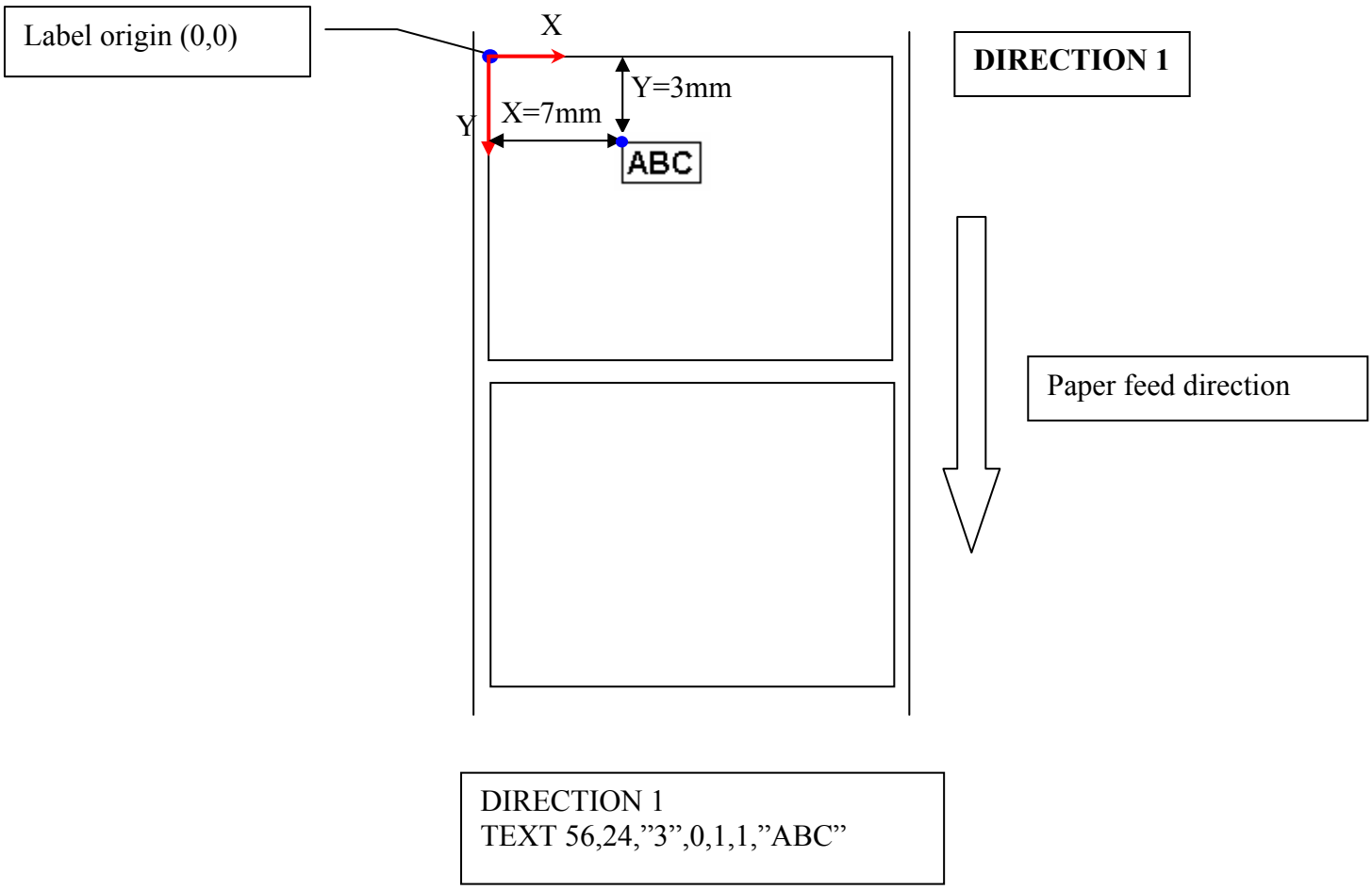
Convention	Description
[expression list]	Items inside square brackets are optional, expression maximum length 2*1024 bytes;
<ESC>	ESCAPE (ASCII 27), control code of status polling command returns the printer status immediately.
~	(ASCII 126), control code of status polling command, returns the printer 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, except the 2D bar code commands.
Note: 203 DPI: 1 mm = 8 dots	Arial font in bold and italic type is used for note.

Object Position Calculation{ TC “Object Position Calculation“}



DIRECTION 0
TEXT 56,24,”3”,0,1,1,”ABC”

203 DPI, 1mm=8 dots
300 DPI, 1mm=12 dots



DIRECTION 1
TEXT 56,24,”3”,0,1,1,”ABC”

Printer Models List

Series	Models
BBP11-24L series	1. BBP11-24LENG
	2. BBP11-24LFR
	3. BBP11-24LGER
	4. BBP11-24LUK
	5. BBP11-24LNL
	6. BBP11-24LIT
BBP11-34L series	7. BBP11-34LENG
	8. BBP11-34LFR
	9. BBP11-34LGER
	10. BBP11-34LUK
	11. BBP11-34LNL
	12. BBP11-34LIT

Setup and System Commands{ TC “Setup and System Commands “}

● SIZE{ XE "SIZE" }{ TC “SIZE”}

Description

This command defines the label width and length.

Syntax

(1) English system (inch)
SIZE m,n
(2) Metric system (mm)
SIZE m mm,n mm
(3) Dot measurement
SIZE m dot,n dot
This command is only supported in v6.27 and later firmware.

Parameter	Description
m	Label width (inch or mm)
n	Label length (inch or mm)

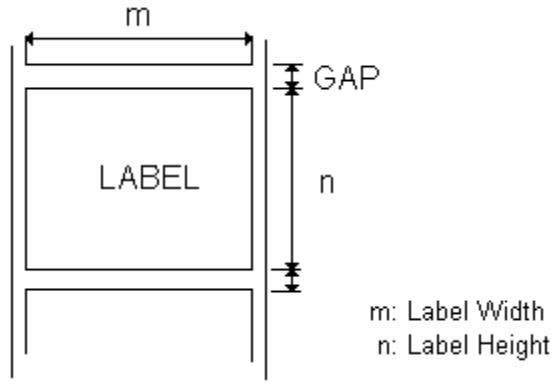
Note:

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”.

Max. width	106mm	108mm
BBP11-24L series		**X**
BBP11-34L series	**X**	

Example

(1) English system (inch)
SIZE 3.5, 3.00
(2) Metric system (mm)
SIZE 100 mm, 100 mm



See Also

GAP, BLINE

● GAP { XE "GAP" }{ TC “GAP”}

Description

Defines the gap distance between two labels

Syntax

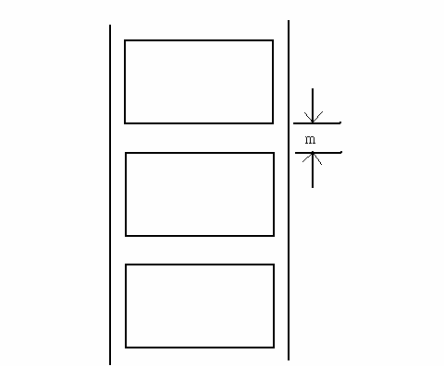
(1). English system (inch)
GAP m,n
(2) Metric system (mm)
GAP m mm,n mm

Parameter	Description
m	The gap distance between two labels $0 \le m \le 1$ (inch), $0 \le m \le 25.4$ (mm)
n	The offset distance of the gap n ≤ label length (inch or mm)
0,0	Continuous label.

Note: For metric system, 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.
Ex: In DOS mode,
C:\>copy con lpt1 <Enter>
GAP 2 mm,0 <Enter>
<Ctrl>+<Z> <Enter>

Example

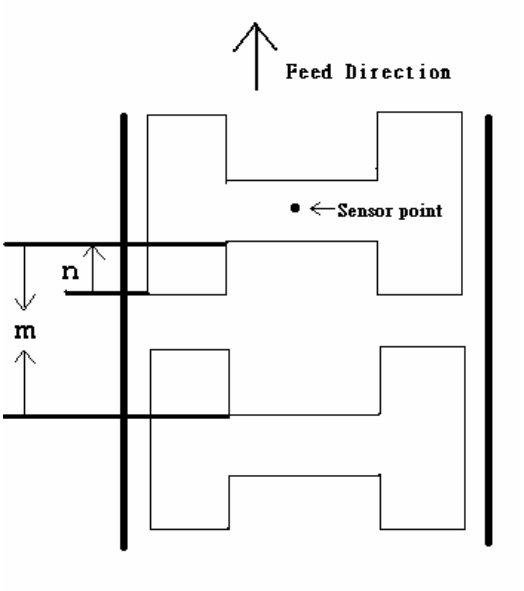
Normal gap
(1). English system (inch)
GAP 0.12,0
(2) Metric system (mm)
GAP 3 mm,0
(3). Continuous label
GAP 0,0



Special gap

(1). English system (inch)
GAP 0.30, 0.10

(2). Metric system (mm)
GAP 7.62 mm, 2.54 mm



See Also

SIZE, BLINE

● GAPDETECT{ TC “GAPDETECT”}

Description

Feeds 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.

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

Syntax

GAPDETECT [x, y]

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

See Also

GAP, SIZE

● BLINE{ XE “BLINE” }{ TC “BLINE”}

Description

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

Syntax

(1) English system (inch)
BLINE m,n
(2) Metric system (mm)
BLINE m mm,n mm

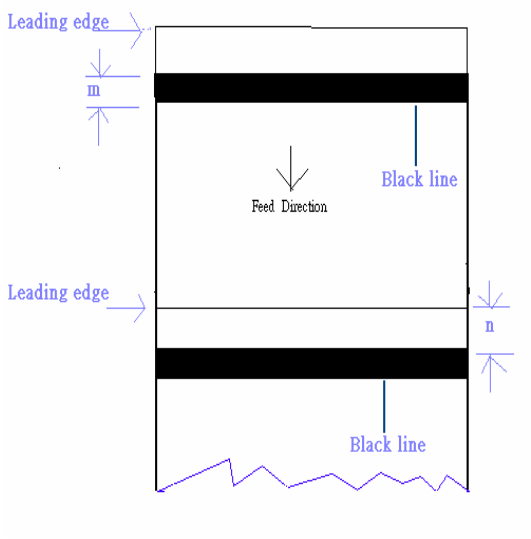
Parameter	Description
m	The height of black line either in inch or mm. $0 \leq m \leq 1$ (inch), $0 \leq m \leq 25.4$ (mm)
n	The extra label feeding length. $0 \leq n \leq$ label length
0,0	Continuous label.

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.
Ex : In DOS mode,
C :\>copy con lpt1 <Enter>
BLINE 2 mm,0 <Enter>
<Ctrl>+<Z> <Enter>

Example

(1) English system (inch)
BLINE 0.20,0.50

(1) Metric system (mm)
BLINE 5.08 mm,12.7 mm



See Also

SIZE, GAP

● OFFSET{ XE “OFFSET” }{ TC “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

(2) English system (inch)
OFFSET m
(3) Metric system (mm)
OFFSET m mm

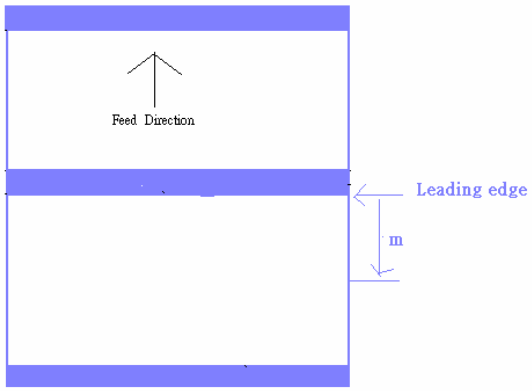
Parameter	**Description**
m	The offset distance (inch or mm) -1 ≤ m ≤ 1(inch)

CAUTION: Impropriety offset value may cause paper jam.

Example

(1) English system (inch)
OFFSET 0.5

(2) Metric system (mm)
OFFSET 12.7 mm



See Also

SIZE, GAP, SET PEEL, SET CUTTER

● SPEED{ XE “SPEED” } { TC “SPEED”}

Description

This command defines the print speed.

Syntax

SPEED n

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

Model / IPS	2	3	4	5
BBP11-24L series	**x**	**x**	**x**	**x**
BBP11-34L series	**x**	**x**		

Example

SPEED 10

See Also

DENSITY

● DENSITY{ XE “DENSITY” }{ TC “DENSITY”}

Description

This sets the printing darkness.

Syntax

DENSITY n

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

Example

DENSITY 7

See Also

DENSITY

● DIRECTION and Mirror Image{ XE “DIRECTION and Mirror Image” }{ TC “DIRECTION and Mirror Image”}

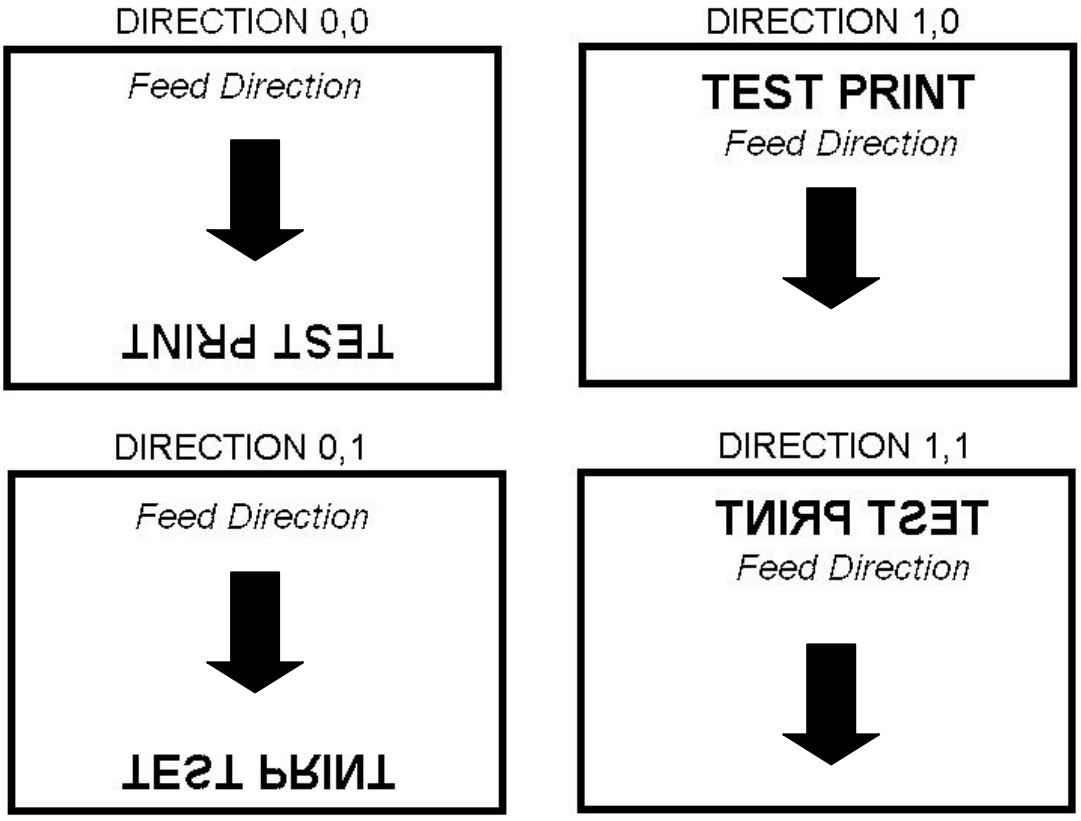
Description

This command defines the printout direction and mirror image. This will be memorized 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

DIRECTION 0[,0]

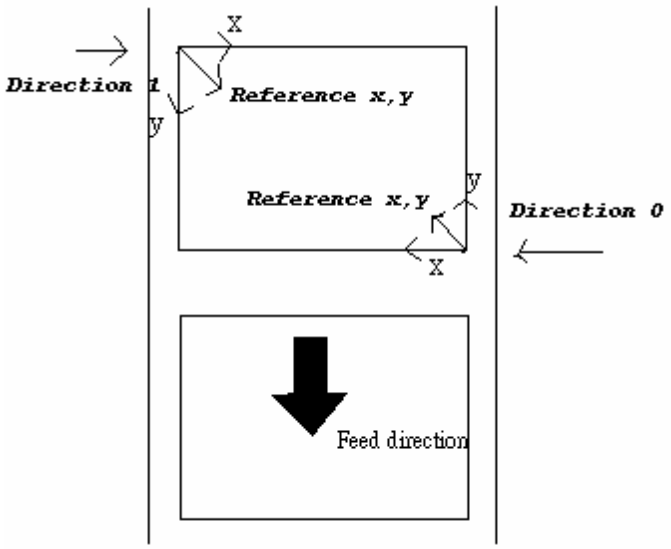
See Also

REFERENCE

● REFERENCE{ XE "REFERENCE" }{ TC "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

REFERENCE 10,10

See Also

DIRECTION

● SHIFT { XE "SHIFT" }{ TC "SHIFT"}

Description

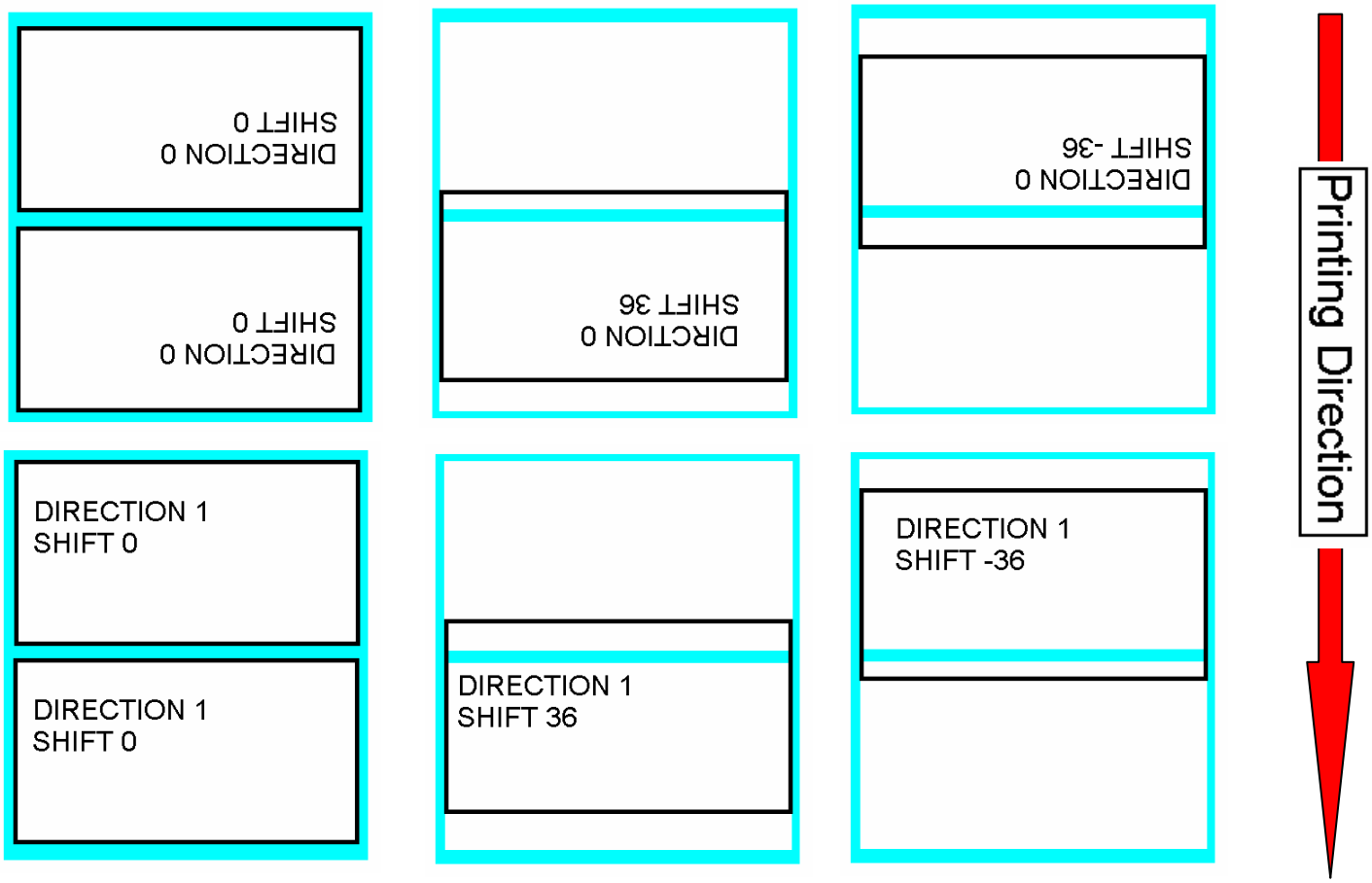
This command moves the label vertical position. A positive value moves the label further from the printing direction; a negative value moves towards. For a visual representation, see next page.

Syntax

SHIFT n

Parameter	Description
n	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




```
SIZE 4,2.5
GAP 2 mm,0
DIRECTION 0
SHIFT 30
OFFSET 0
CLS
TEXT 400,200,”3”,0,1,1,”DIRECTION 0”
TEXT 400,250,”3”,0,1,1,”SHIFT 30”
BOX 10,0,780,490,8
PRINT 3,1
```

See Also

OFFSET, REFERENCE

● COUNTRY{ XE “COUNTRY” }{ TC “COUNTRY”}

Description

This command orients the keyboard for use in different countries via defining special characters on the BBP11-SK series portable LCD keyboard (option).

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

COUNTRY 001

See Also

CODEPAGE, ~!I

● CODEPAGE{ XE “CODEPAGE” }{ TC “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 further. 7-bit code page name USA: USA BRI: British GER: German FRE: French DAN: Danish ITA: Italian SPA: Spanish SWE: Swedish SWI: Swiss 8-bit code page number 437: United States 850: Multilingual 852: Slavic 860: Portuguese 863: Canadian/French 865: Nordic 857: Turkish Windows code page 1250: Central Europe 1252: Latin I 1253: Greek 1254: Turkish

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

.

Example

CODEPAGE 437

See Also

COUNTRY, SET COM1, ~!I

● CLS{ XE "CLS" }{ TC "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
CLS

See Also
SIZE, GAP, BLINE

● FEED{ XE "FEED" }{ TC "FEED"}

Description

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

Syntax

FEED n

Parameter	Description
n	unit: dot $1 \leq n \leq 9999$

Example

FEED 40

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

See Also

BACKFEED, SIZE, GAP, BLINE, HOME, FORMFEED

● BACKFEED & BACKUP { XE " BACKFEED & BACKUP " }{ TC " BACKFEED & BACKUP "}

Description

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

Syntax

BACKFEED n

Parameter	Description
n	unit: dot $1 \leq n \leq 9999$

Example

BACKFEED 40

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

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

See Also

FEED, SIZE, GAP, BLINE, HOME, FORMFEED

● FORMFEED{ XE “FORMFEED” }{ TC “FORMFEED”}

Description

This command feeds label to the beginning of next label.

Syntax

FORMFEED

Parameter	Description
None	N/A

Example

```
SIZE 4,2.5
GAP 0 mm,0
SPEED 4
DENSITY 7
DIRECTION 0
OFFSET 0.00
REFERENCE 0,0
SET PEEL OFF
SET CUTTER OFF
SET COUNTER @0 +1
@0=”000001”
FORMFEED
CLS
BOX 1,1,360,65,12
TEXT 25,25,”3”,0,1,1,”FORMFEED COMMAND TEST”
TEXT 25,80,”3”,0,1,1,@0
PRINT 3,1
```

See Also

FEED, SIZE, GAP, BLINE, HOME, BACKFEED

● HOME { XE “HOME” }{ TC “HOME”}

Description

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

Syntax

HOME

Parameter	Description
None	N/A

Example

```
SIZE 4,2.5
GAP 2 mm,0
SPEED 4
DENSITY 7
DIRECTION 0
OFFSET 0.00
REFERENCE 0,0
SET PEEL OFF
SET CUTTER OFF
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

● PRINT{ XE "PRINT" }{ TC "PRINT"}

Description

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

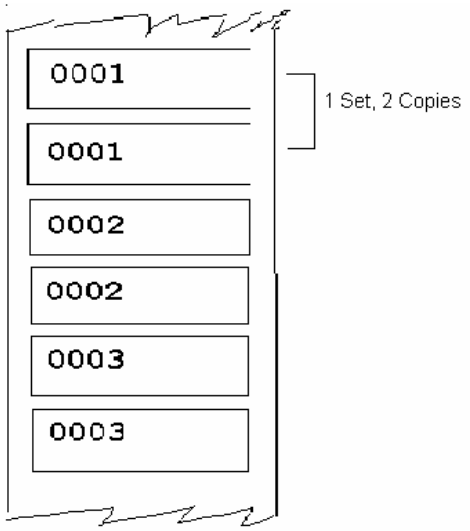
Syntax

PRINT m [,n]

Parameter	Description
m	Specifies how many sets of labels will be printed. 1 ≤ m ≤ 999999999 If m=-1, printer will print the last label content for n copies.
n	Specifies how many copies should be printed for each particular label set. 1 ≤ n ≤ 999999999

Example

```
SIZE 60 mm, 20 mm
SET COUNTER @1 1
@1=”0001”
CLS
TEXT 10,10,”3”,0,1,1,@1
PRINT 3,2
PRINT –1,2
```



See Also

SET COUNTER, INPUT, DOWNLOAD

● SOUND{ TC "SOUND" }{ XE "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

```
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
```


● CUT{ TC "CUT" }{ XE "CUT" }

Description

This command activates the cutter to immediately cut the labels without back feeding the label.

Syntax

CUT

Parameter	Description
None	N/A

Example

```
SIZE 3,3
GAP 0 mm,0
CLS
DIRECTION 0
REFERENCE 0,0
OFFSET 0.00 mm
SET CUTTER OFF
SET TEAR OFF
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

● LIMITFEED{ TC "LIMITFEED" }{ XE "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 (inch, the English system)
LIMITFEED n mm (mm, the metric system)

Parameter	Description
n	inch or mm

Remark

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

● SELFTEST{ TC "SELFTEST" }{ XE "SELFTEST" }

Description

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

Syntax

SELFTEST

Example

SELFTEST

Label Formatting Commands{ TC “Label Formatting Commands “}

● BAR{ XE "BAR" }{ TC “BAR”}

Description

This command draws a bar on the label format.

Syntax

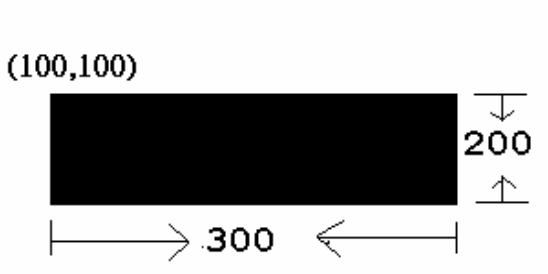
BAR x, y, width, height

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 12mm at 4” width. Bar height over 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

```
SIZE 4,2.5
GAP 0,0
SPEED 6
DENSITY 8
DIRECTION 0
CLS
BAR 100, 100, 300, 200
PRINT 1,1
```



See Also

BOX

- **BARCODE{ XE "BARCODE" }{ TC “BARCODE”}**

Description

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

- Code 128 (switching code subset automatically)
- Code 128M (switching code subset manually)
- EAN 128 (switching code subset automatically)
- Interleaved 2 of 5
- Interleaved 2 of 5 with check digit
- Code 39 standard
- Code 39 full ASCII
- Code 39 full ASCII with check digit
- Code 93
- EAN 13
- EAN 13 with 2 digits add-on
- EAN 13 with 5 digits add-on
- EAN 8
- EAN 8 with 2 digits add-on
- EAN 8 with 5 digits add-on
- Codabar
- Postnet
- UPC-A
- UPC-A with 2 digits add-on
- UPC-A with 5 digits add-on
- UPC-E
- UPC-E with 2 digits add-on
- UPC-E with 5 digits add-on
- MSI
- PLESSEY
- China POST
- ITF14
- EAN14

Syntax

BARCODE X, Y, ”code type”, height, human readable, rotation, narrow, wide, “code”

Parameter	**Description**
X	Specify the x-coordinate bar code on label
Y	Specify the y-coordinate bar code on label
Code type	
128	Code 128, switching code subset A, B, C automatically
128M	Code 128, switching code subset A, B, C manually.

Control code	A	B	C
096	FNC3	FNC3	NONE
097	FNC2	FNC2	NONE
098	SHIFT	SHIFT	NONE
099	CODE C	CODE C	NONE
100	CODE B	FNC4	CODE B
101	FNC4	CODE A	CODE A
102	FNC1	FNC1	FNC1
103	Start (CODE A)		
104	Start (CODE B)		
105	Start (CODE C)		

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.

EAN128	Code 128, switching code subset A, B, C automatically
25	Interleaved 2 of 5
25C	Interleaved 2 of 5 with check digits
39	Code 39 full ASCII
39C	Code 39 full ASCII with check digit
39S	Code 39 standard
93	Code 93
EAN13	EAN 13
EAN13+2	EAN 13 with 2 digits add-on
EAN13+5	EAN 13 with 5 digits add-on
EAN8	EAN 8
EAN8+2	EAN 8 with 2 digits add-on
EAN8+5	EAN 8 with 5 digits add-on
CODA	Codabar
POST	Postnet
UPCA	UPC-A
UPCA+2	UPC-A with 2 digits add-on
UPCA+5	UPC-A with 5 digits add-on
UPCE	UPC-E
UPCE+2	UPC-E with 2 digits add-on

UPCE+5	UPC-E with 5 digits add-on
CPOST	China post code
MSI	MSI code
MSIC	
PLESSEY	PLESSEY code
ITF14	ITF 14 code
EAN14	EAN 14 code

Height	Bar code height (in dots)
Human readable	0: not readable
	1: human readable
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	-	-	-	-
25	-	10x	10x	5x	-
25C	-	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	-	-	-	-	-

` code number the maximum number
of digits of bar code content

Barcode type	Maximum bar code length
128	-
EAN128	-
25	-
25C	-
39	-
39C	-
93	-
EAN13	12
EAN13+2	14
EAN13+5	17
EAN 8	7
EAN 8+2	9
EAN 8+5	12
CODA	-
POST	5,9,11
UPCA	11
UPCA+2	13
UPCA+5	16
UPCE	6
UPCE+2	8
UPCE+5	11
CPOST	-
MSI	-
MSIC	
PLESSY	-
ITF14	13
EAN14	13

Example

BARCODE 100,100,”39”,96,1,0,2,4,”1000”
BARCODE 10,10,”128M”,48,1,0,2,2,”!104!096ABCD!101EFGH”
(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.)

Model \ Barcode Type	128	EAN128	25	25C	39 for TSPL2	39 for TSPL	39 for PLUS	39C for TSPL2	39C for TSPL	39C for PLUS	39S	93	EAN13	EAN13+2	EAN13+5	EAN 8	EAN 8+2	EAN 8+5	CODA	POST	UPCA	UPCA+2	UPCA+5	UPCE	UPCE+2	UPCE+5	CPOST	MSI	MSIC	PLESSY	ITF14	EAN14
BBP11-24L series	X	X	X	X	X			X			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
BBP11-34L series	X	X	X	X	X			X			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

● BITMAP{ XE "BITMAP" }{ TC “BITMAP”}

Description

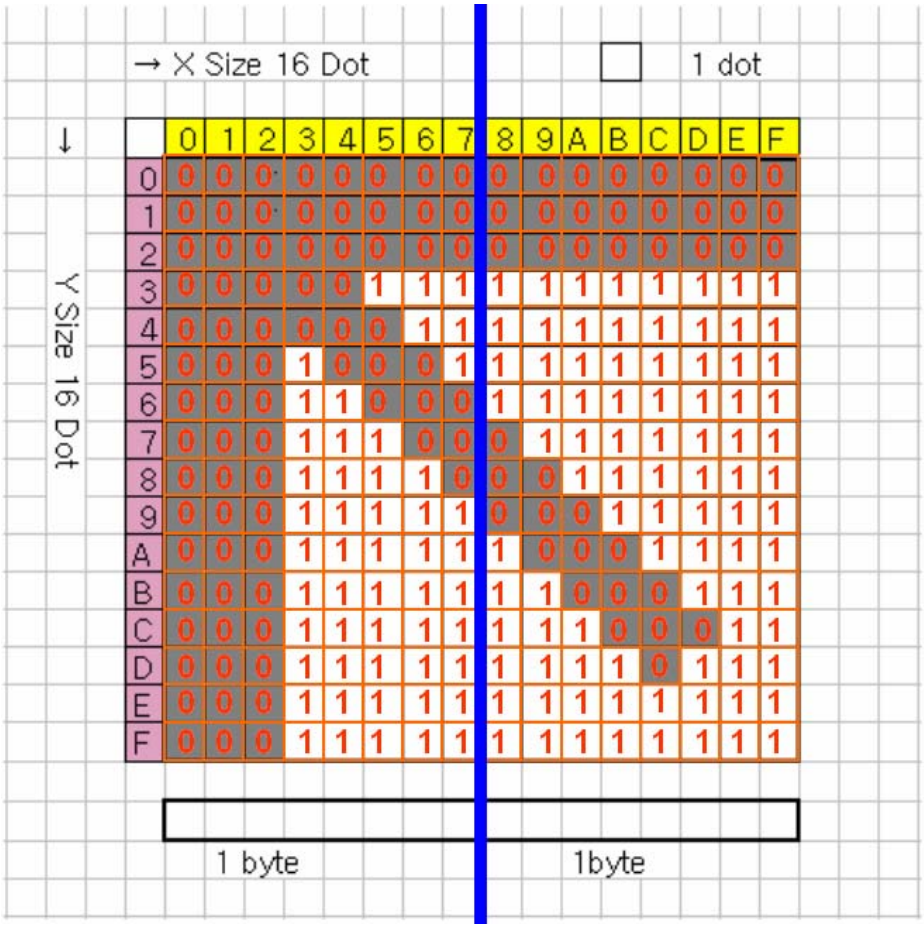
This command draws bitmap images (as opposed to BMP graphic files).

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



ROW (Y- axis)	L-Byte		R-Byte	
	Binary	Hexadecimal	Binary	Hexadecimal
0	00000000	00	00000000	00
1	00000000	00	00000000	00
2	00000000	00	00000000	00
3	00000111	07	11111111	FF
4	00000011	03	11111111	FF
5	00010001	11	11111111	FF
6	00011000	18	11111111	FF
7	00011100	1C	01111111	7F
8	00011110	1E	00111111	3F
9	00011111	1F	00011111	1F
A	00011111	1F	10001111	8F
B	00011111	1F	11000111	C7
C	00011111	1F	11100011	E3
D	00011111	1F	11110111	F7
E	00011111	1F	11111111	FF
F	00011111	1F	11111111	FF

Ex:

```
SIZE 4,2
GAP 0,0
CLS
BITMAP 200,200,2,16,0,        □□□□□□□□□□-?????□□
PRINT 1,1
```

Hexadecimal	ASCII
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	SIZE 4,2 GAP 0,0 CLS BITMAP 200,200,2,16,0, □□□□□□□□□□-?????□□ PRINT 1,1

See Also

PUTBMP, PUTPCX

● BOX{ XE "BOX" }{ TC “BOX”}

Description

This command draws rectangles on the label.

Syntax

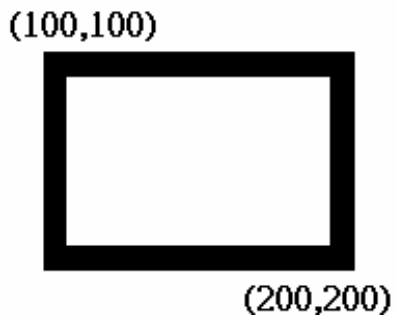
BOX X_start, Y_start, X_end, Y_end, line thickness

Parameter	Description
X_start	Specify x-coordinate of upper left corner (in dots)
Y_start	Specify y-coordinate of upper left corner (in dots)
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)

Note: ***200 DPI: 1 mm = 8 dots***
300 DPI: 1 mm = 12 dots
Recommended max. thickness of box is 12mm 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

```
SIZE 4,2.5
GAP 0,0
SPEED 6
DENSITY 8
DIRECTION 0
CLS
BOX 100,100,200,200,5
PRINT 1,1
```



See Also

BAR

● DMATRIX { TC "DMATRIX" }{ XE "DMATRIX" }

Description

This command is used to define the DataMatrix 2D bar code. Currently, only ECC200 error correction is supported.

Syntax

DMATRIX x, y, width, height, [xm,row,col], expression

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)
xm	Module size (in dots)
row	Symbol size of row: 10 to 144
col	Symbol size of col: 10 to 144

Example

```
SIZE 3,3
GAP 0,0
SPEED 4
DENSITY 8
DIRECTION 0
REFERENCE 0,0
OFFSET 0.00
SET CUTTER OFF
SET TEAR ON
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
```


● ERASE{ XE "ERASE" }{ TC “ERASE”}

Description

This command clears a specified region in image buffer.

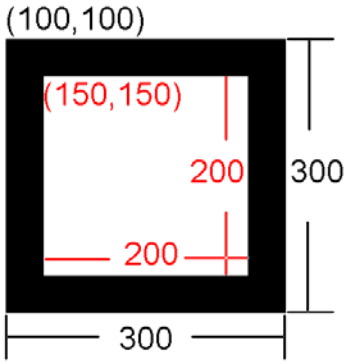
Syntax

ERASE 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	The region width in x-axis direction (in dots)
Y_height	The region height in y-axis direction (in dots)

Example

```
SIZE 4,2.5
GAP 0,0
SPEED 6
DENSITY 8
DIRECTION 0
CLS
BAR 100, 100, 300, 300
ERASE 150,150,200,200
PRINT 1,1
```



See Also

CLS

● MAXICODE{ XE "MAXICODE" }{ TC “MAXICODE”}

Description

This command defines a 2D Maxicode.

Syntax

MAXICODE x, y, mode, [class, country, post, Lm,] “message”

For mode 2 or 3:
MAXICODE x, y, mode, class, country, postal code, ”low priority message"
If country is 840, the postal code is in 99999,9999 format.
For other countries, the code is up to 6 alphanumeric characters.

For mode 4,5,6
MAXICODE x, y, mode, [Lm], "message"

* AIM special format is supported, see page 23 in the spec.

Note: Mode 6 is not supported in TSPL2 printers firmware.

Parameter	**Description**
x	X-coordinate of the staring 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 Mode3: (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.)
message	Barcode content

Example

```
SIZE 4,2
GAP 0,0
CLS
SPEED 4
DENSITY 8
OFFSET 0.00
REFERENCE 0,0
SET CUTTER OFF
SET TEAR ON

REM *****Mode 2 For USA*****
MAXICODE 110,100,2,300,840,06810,7317,"DEMO 2 FOR USA MAXICODE"
```



```
TEXT 100,520,"3",0,2,2,"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 100,520,"3",0,2,2,"Mode 3 For CANADA"
PRINT 1,1

REM *****MODE4*****
CLS
MAXICODE 110,100,4,"DEMO 4 FOR MAXICODE"
MAXICODE 600,100,4,19,DEMO 4 FOR MAXICODE
TEXT 100,520,"3",0,2,2,"Mode 4 FOR MAXICODE"
PRINT 1,1

REM *****MODE 5*****
CLS
MAXICODE 110,100,5,"DEMO 5 FOR MAXICODE"
MAXICODE 600,100,5,19,DEMO 5 FOR MAXICODE
TEXT 100,520,"3",0,2,2,"DEMO 5 FOR MAXICODE"
PRINT 1
```


● PDF417 { XE "PDF417" }{ TC “PDF417”}

Description

This command defines a PDF417 2D barcode.

Syntax

PDF417 x, y, width, height, rotate, [option], expression

Parameter	Description
x	X-coordinate of starting point (in dots)
y	Y-coordinate of starting point (in dots)
width	Expected width (in dots)
height	Expected height (in dots)
rotate	Rotation counterclockwise.
0:	No rotation
90:	90 degrees
180:	180 degrees
270:	270 degrees
expression	Barcode text or string expression to be printed.
[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 (without double quote), 1≤m≤2048

Example


```
SIZE 3,3
GAP 0.12,0
CLS
SPEED 6
DENSITY 8
DIRECTION 1
REFERENCE 0,0

REM *****WITHOUR OPTIONS*****
CLS
PDF417 50,50,400,200,0,"Without Options"
PRINT 1,1

REM *****OPTION:E4*****
CLS
PDF417 50,50,400,200,0,E4,"Error correction level:4"
PRINT 1,1

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


REM *****OPTION:E4 W4 H4 R25*****
CLS
PDF417 50,50,600,600,0,E4,W4,H4,R25,"Error correction level:4
Module Width 4 dots
Bar Height 4 dots
Maximum Number of Rows: 25 Rows
"
PRINT 1,1

REM *****OPTION:E4 W4 H4 R40 C3*****
CLS
PDF417 50,50,600,600,0,E4,W4,H4,R40,C3,"Error correction level:4
Module Width 4 dots
Bar Height 4 dots
Maximum Number of Rows: 40 Rows
Maximum number of columns: 3 Cols
```



```
"
PRINT 1,1

REM *****OPTION:E4 W4 H4 R40 C4 T0*****
CLS
PDF417 50,50,600,600,0,E4,W4,H4,R40,C4,T0,"Error correction level:4
Module Width 4 dots
Bar Height 4 dots
Maximum Number of Rows: 40 Rows
Maximum number of columns: 4 Cols
Truncation:0
"
PRINT 1,1

REM *****OPTION:E4 W4 H4 R40 C4 T1*****
CLS
PDF417 50,50,900,900,0,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

REM *****OPTION:E4 W4 H4 R40 C4 T0 L169*****
CLS
PDF417 50,50,900,900,0,E4,W4,H4,R40,C4,T0,L169,Error correction level:4
Module Width 4 dots
Bar Height 4 dots
Maximum Number of Rows: 40 Rows
Maximum number of columns: 4 Cols
Truncation:0
Expression length:167
PRINT 1,1

REM *****OPTION:E4 W4 H4 R40 C4 T1 L169*****
CLS
PDF417 50,50,900,900,0,E4,W4,H4,R40,C4,T1,L169,Error correction level:4
Module Width 4 dots
Bar Height 4 dots
Maximum Number of Rows: 40 Rows
Maximum number of columns: 4 Cols
Truncation:1
Expression length:169
PRINT 1,1

REM *****OPTION:P0 E4 W4 H4 R40 C4 T1 L169*****
CLS
```


PDF417 50,50,900,900,0,P0,E4,W4,H4,R40,C4,T1,L169,Error correction level:4
Module Width 4 dots
Bar Height 4 dots
Maximum Number of Rows: 40 Rows
Maximum number of columns: 4 Cols
Truncation:1
Expression length:169
PRINT 1,1

REM *****OPTION:P0 E4 M0 W6 H6 R60 C4 T0 L283*****
SIZE 3,2
CLS
PDF417 50,50,900,600,0,P0,E4,M0,W6,H6,R60,C4,T0,L283,Data compression method: P0
Error correction level: E4
Center pattern in barcode area: M0
Human Readable: No
Module Width 6 dots: W6
Bar Height 6 dots: H6
Maximum Number of Rows: 60 Rows: R60
Maximum number of columns: 4 Cols: C4
Truncation:0: T0
Expression length:283: L283
PRINT 1,1

REM *****OPTION:P1 E4 M1 U100,500,10 W4 H4 R60 C4 T1 L297*****
CLS
PDF417 50,50,900,600,0,P1,E4,M1,U100,500,10,W6,H6,R60,C4,T1,L297,Data compression method: P1
Error correction level: E4
Center pattern in barcode area: M1
Human Readable: Yes: U100,300,10
Module Width 6 dots: W6
Bar Height 6 dots: H6
Maximum Number of Rows: 60 Rows: R60
Maximum number of columns: 4 Cols: C4
Truncation:1: T1
Expression length:297: L297
PRINT 1,1

● PUTBMP{ XE "PUTBMP" }{ TC “PUTBMP”}

Description

This command prints BMP format images.

Syntax

PUTBMP X, Y, ”filename”

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.

Example

```
C:\BMP-PCX>dir
 Volume in drive C has no label.
 Volume Serial Number is CCF4-7BE4

 Directory of C:\BMP-PCX

11/02/2009  13:02    <DIR>          .
11/02/2009  13:02    <DIR>          ..
11/02/2009  13:00               139 BRADY.BMP
11/02/2009  13:01                64 BRADY.pcx
               2 File(s)            203 bytes
               2 Dir(s)   3.577.835.520 bytes free

C:\BMP-PCX>COPY CON LPT1
DOWNLOAD "BRADY.BMP",139,^Z
        1 file(s) copied.

C:\BMP-PCX>COPY BRADY.BMP/B LPT1
        1 file(s) copied.

C:\BMP-PCX>COPY CON LPT1
SIZE 4,2.5
GAP 0,0
CLS
PUTBMP 100,100,"BRADY.BMP"
PRINT 1,1
^Z
        1 file(s) copied.

C:\BMP-PCX>
```

See Also

DOWNLOAD, BITMAP, PUTPCX

● PUTPCX{ XE "PUTPCX" }{ TC “PUTPCX”}

Description

This command prints PCX format images.
TSPL2 language supports 256-color PCX format graphics.

Model	Support	
	2 – color PCX	256 – color PCX
BBP11-24L series	**X**	**X**
BBP11-34L series	**X**	**X**

Syntax

PUTPCX X, Y, ”filename”

Parameter	**Description**
X	The x-coordinate of the PCX image
Y	The y-coordinate of the PCX image
filename	The downloaded PCX filename. Case sensitive

Example

```
C:\BMP-PCX>dir
 Volume in drive C has no label.
 Volume Serial Number is CCF4-7BE4

 Directory of C:\BMP-PCX

11/02/2009  13:02    <DIR>          .
11/02/2009  13:02    <DIR>          ..
11/02/2009  13:00               139 BRADY.BMP
11/02/2009  13:01                64 BRADY.pcx
               2 File(s)            203 bytes
               2 Dir(s)   3.570.733.056 bytes free

C:\BMP-PCX>COPY CON LPT1
DOWNLOAD "BRADY.pcx",64,^Z
        1 file(s) copied.

C:\BMP-PCX>COPY BRADY.PCX/B LPT1
        1 file(s) copied.

C:\BMP-PCX>COPY CON LPT1
SIZE 4,2.5
GAP 0,0
CLS
PUTPCX 100,100,"BRADY.PCX"
PRINT 1,1
^Z
        1 file(s) copied.

C:\BMP-PCX>
```

See Also

DOWNLOAD, BITMAP, PUTPCX

● QRCODE{ XE "QRCODE" }{ TC “QRCODE”}

Description

This command prints QR code

Syntax

QRCODE X, Y, ECC Level, cell width, mode, rotation, [model, mask,]"Data string”

Parameter	Description
X	The upper left corner x-coordinate of the QR code
Y	The upper left corner y-coordinate of the QR code
ECC level	Error correction recovery level
L	7%
M	15%
Q	25%
H	30%
cell width	1~10
mode	Auto / manual encode
A	Auto
M	Manual
rotation	
0	0 degree
90	90 degree
180	180 degree
270	270 degree
model	
M1	(default), original version
M2	enhanced version
mask	S0~S8, default is S7
Data string	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):

	Model 1 (Version 14-L)	Model 2 (Version 40-L)

1). Numeric data:	1,167 characters	7,089 characters
2). Alphanumeric data:	707 characters	4,296 characters
3). 8-bit byte data:	486 characters	2,953 characters
4). Kanji data:	299 characters	1,817 characters

*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.
*If “!” is in the data string and follows by “N”, “A”, “B”, “K” then it will be switched to specified encodable character set.

Example

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"

(1) Auto mode example

a. General data string

```
SIZE 4,2.5
GAP 0.12,0
CLS
QRCODE 10,10,H,4,A,0,"ABCabc123"
QRCODE 160,160,H,4,A,0,"123ABCabc"
QRCODE 310,310,H,4,A,0,"□□□ABCabc123"
PRINT 1,1
```

b. Data string including <Enter> character (0Dh, 0Ah)

```
SIZE 4,2.5
GAP 0.12,0
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,"□□□<Enter>
ABC<Enter>
abc<Enter>
123"
PRINT 1,1
```

c. Data string concatenation (Must be used with DOWNLOAD ... EOP command)

```
DOWNLOAD "DEMO.BAS"
SIZE 4,2.5
CAP 0.12,0
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,"□□□ABCabc123"+"1234"+"abcd"
PRINT 1,1
EOP
DEMO
```

d. Data string including double quote (") character, please use \["] instead of

```
SIZE 4,2.5
CAP 0.12,0
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,"\["]□□□\["]ABCabc123"
PRINT 1,1
```

(3) Manual mode

a. General data string□

```
SIZE 4,2.5
CAP 0.12,0
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□□□!AABC!B0006abc123"
PRINT 1,1
```

b. Data string including <Enter> character, <Enter> is an 8-bit byte data

```
SIZE 4,2.5
CAP 0.12,0
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□□□!B0002<Enter>
```



```
!AABC!B0010<Enter>
abc<Enter>
123"
  PRINT 1,1
```

c. Data string concatenation (Must be used with DOWNLOAD ... EOP command)

```
DOWNLOAD “A.BAS”
SIZE 4,2.5
CAP 0.12,0
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□□□!AABC!B0014abc123"+"1234"+"abcd"
PRINT 1,1
EOP
A
```

d. Data string including double quote (“) character, please use \["] instead of

```
SIZE 4,2.5
CAP 0.12,0
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□□□!B0010\["]ABCabc123"
PRINT 1,1
```


● REVERSE{ XE "REVERSE" }{ TC “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

```
SIZE 4,2.5
GAP 0,0
SPEED 6
DENSITY 8
DIRECTION 0
CLS
TEXT 100,100,"3",0,1,1,"REVERSE"
REVERSE 90,90,128,40
PRINT 1,1
```

● TEXT{ XE "TEXT" }{ TC “TEXT”}

Description

This command prints text on label

Note:

(1). Font “0” and “ROMAN.TTF” internal True Type Fonts are available in TSPL2 language printers

Syntax

TEXT X, Y, ”font”, rotation, x-multiplication, y-multiplication, “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
Rotation	The rotation angle of text
0	No rotation
90	90 degrees, in clockwise direction
180	180 degrees, in clockwise direction
270	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.*

Note:

1. ***If there is any double quote (“) within the text, please change it to \[“].***
2. ***Font “0” and “ROMAN.TTF” internal True Type Fonts are available in TSPL2 language printers.***
3. ***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.***

MODEL	Font Type									
	0	1	2	3	4	5	6	7	8	ROMAN.TTF
BBP11-24L series	X	X	X	X	X	X	X	X	X	X
BBP11-34L series	X	X	X	X	X	X	X	X	X	X

Example

```
SIZE 3,2
GAP 0,0
CLS
TEXT 100,100,”5”,0,1,1,”\[“]DEMO FOR TEXT\[“]”
TEXT 100,200,”ROMAN.TTF”,0,1,20,”\[“]True Type Font Test Print\[“]”
PRINT 1,1
```


Status Polling Commands (RS-232){ TC “Status Polling Commands (RS-232)”}

● <ESC>!?{ XE “<ESC>!?” }{ TC “<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.

Bit	**Status**
0	Head opened
1	Paper jam
2	Out of paper
3	Out of ribbon
4	Pause
5	Printing
6	Cover opened (option)
	Environment Temperature over range (option)

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

Syntax

<ESC>!?

See Also

<ESC>!R

● <ESC>!R{ XE "<ESC>!R" }{ TC "<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

Parameter	Description
N/A	N/A

See Also

<ESC>!?

● ~!@ { XE "~!@" }{ TC "~!@"}

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

~!@

Parameter	Description
None	N/A

Example

~!@

● ~!A { XE "~!A" }{ TC "~!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

Parameter	Description
None	N/A

Example

~!A

See Also

FILES

● ~!C { XE "~!C" }{ TC "~!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.

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

Syntax

~!C

Parameter	Description
None	N/A

Example

~!C

See Also

YEAR, MONTH, DATE, WEEK, HOUR, MINUTE, SECOND, @YEAR, @MONTH, @DATE, @DAY, @HOUR, @MINUTE, @SECOND

● ~!D { XE “~!D” }{ TC “~!D”}

Description

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

Syntax

~!D

Parameter	Description
None	N/A

Example

~!D

● ~!F{ XE “~!F” }{ TC “~!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

Parameter	Description
None	N/A

Example

~!F

See Also

FILES

● ~!I{ XE “~!I” }{ TC “~!I”}

Description

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

Syntax

~!I

Parameter	Description
None	N/A

Example

~!I

See Also

COUNTRY, CODEPAGE

● ~!T{ XE “~!T” }{ TC “~!T”}

Description

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

Printer Type	Returned String
BBP11-24L	BBP11-24
BBP11-34L	BBP11-34

Syntax

~!T

Parameter	**Description**
None	N/A

Example

~!T

See Also

~!I, ~!F

Message Translation Protocols{ TC “Message Translation Protocols “}

● ~#{ XE "~#" }{ TC "~#"}

Description

The beginning identifier (~#) of the prompt message is sent from the printer to the BBP11-SK portable keyboard. The ending identifier is ~&.

@0 following the ending identifier ~& is used to instruct keyboard to display the prompt in the first line of LCD display.

@1 following the ending identifier ~& is used to instruct keyboard to display the prompt in the first line of LCD display.

If @0 or @1 are not present, prompt string will be displayed in first line of LCD and input data will be displayed in second line of LCD.

Syntax

~#Prompt~&[@0]
~#Prompt~&[@1]

Parameter	**Description**
None	N/A

Example

```
DOWNLOAD "A.BAS"
OUT "~#BBP11-SK~&@0"
OUT "~#Testing~&@1"
EOP
 A
```

See Also

INPUT, OUT

Commands for Windows Driver{ TC “Commands for Windows Driver “}

● !B{ XE "!B" }{ TC "!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

● !J{ XE "!J" }{ TC “!J”}

Description

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

Syntax

!Jnnnn

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

Example

!J0100

See Also

FEED

● !N{ XE "!N" }{ TC “!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

File Management Commands{ TC “File Management Commands “}

● DOWNLOAD{ XE "DOWNLOAD" }{ TC "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 to 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.

Maximum numbers of file saved in Flash memory:

256 files

Model	Maximum numbers of file saved in		
	DRAM	FLASH	Ext. FLASH
BBP11-24L	50	256	Depends on SD card capacity
BBP11-34L	50	256	Depends on SD card capacity

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.

Hold the FEED key and power on the switch. The LED color will be changed as following pattern.

Orange → red (5 blinks) → orange (5 blinks) → green (5 blinks) → solid green (for firmware version before V3.37)

Orange → red (5 blinks) → orange (5 blinks) → green (5 blinks) → green and orange (5 blinks) → red and orange (5 blinks) → solid green (V3.37)

Release the FEED key while LED becomes solid green to prevent the printer from running “AUTO.BAS”.

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:

(1). Ffilenames are case sensitive.
(2). File extensions must be ".BAS"
(3). Filenames must be in 8.3 format.
(4). 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.

2. Download a data file
   DOWNLOAD [n,]"FILENAME", DATA SIZE, DATA CONTENT…where

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)

Note:

(1). For text data files, CR (carriage return) 0x0D and LF (Line Feed) 0x0A is the separator of data.
(2). 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.

	Support			
	DOWNLOAD "filename"	MOVE	DOWNLOAD F,"filename"	DOWNLOAD E,"filename"
BBP11-24L series	**X**	**X**	**X**	**X**
BBP11-34L series	**X**	**X**	**X**	**X**

Example

The example program listed below will download to printer SDRAM.

```
DOWNLOAD "EXAMPLE.BAS"
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 100,100,"3",0,1,1,"EXAMPLE PROGRAM"
PRINT 1
EOP
```

Note: 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:

1. ***Call the main filename***

```
C:\>COPY CON LPT1<ENTER>
EXAMPLE<ENTER>
<CTRL><Z>
C:\>
```

2. ***Use Run command to start the program***

```
C:\>COPY CON LPT1<ENTER>
RUN "EXAMPLE.BAS"<ENTER>
<CTRL><Z>
C:\>
```

Below is an example of downloading data file.

```
DOWNLOAD "DATA",20,COMPUTER<Enter>
2001<Enter>
21<Enter>
```

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

See Also

EOP, RUN, PUTBMP, PUTPCX, INPUT

● EOP{ XE “EOP”}{ TC “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

```
DOWNLOAD “DEMO.BAS”
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 100,100,”3”,0,1,1,”DEMO PROGRAM”
PRINT 1
EOP
```

See Also

DOWNLOAD, EOP, INPUT

● FILES{ XE "FILES" }{ TC “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

Follow the steps below to print out (or list through RS-232) files saved in printer memory using the DOS environment through serial port or parallel port connection.

C:\>MODE COM1 96,N,8,1<ENTER>
C:\>COPY CON COM1<ENTER>
FILES<ENTER>
<CTRL><Z><ENTER>
C:\>

Or

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, KILL

● KILL{ XE "KILL" }{ TC “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.

Model	Support			
	KILL “*”	KILL “*” MOVE	KILL F,”*”	KILL E,”*”
BBP11-24L series	**X**		**X**	**X**
BBP11-34L series	**X**		**X**	**X**

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.

Note:

(1). If optional parameter n is not specified, firmware will delete the file in DRAM.

Syntax example

1. KILL “FILENAME”
2. KILL “*.PCX”
3. KILL “*”
4. KILL F,“FILENAME”
5. KILL E,“*.PCX”

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

● MOVE{ XE "MOVE" }{ TC “MOVE”}

Description

This command moves downloaded files from DRAM to FLASH memory.

Syntax

MOVE

Parameter	Description
N/A	N/A

See Also

DOWNLOAD, EOP

● RUN{ XE "RUN" }{ TC “RUN”}

Description

This command executes a program resident in the printer memory
This command is available for TSPL2 language printers only.

Syntax

RUN “FILENAME.BAS”

Example

C:\>COPY CON LPT1<ENTER>
RUN “DEMO.BAS”<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

See Also

DOWNLOAD, EOP

BASIC Commands and Functions{ TC “BASIC Commands and Functions “}

● ABS( ){ XE "ABS( )" }{ TC "ABS( )"}

Description

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

Syntax

```
ABS (-100)
ABS (-99.99)
ABS (VARIABLE)
```

Example

```
DOWNLOAD “TEST.BAS”
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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$(ABS(C))
PRINT 1
EOP
```

See Also

DOWNLOAD, EOP

● ASC( ){ XE "ASC( )" }{ TC "ASC( )"}

Description

This function returns the ASCII code of the character.

Syntax

ASC (“A”)

Example

```
DOWNLOAD “TEST.BAS”
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
CODE1=ASC(“A”)
TEXT 100,100,”3”,0,1,1,STR$(CODE1)
PRINT 1
EOP
```

See Also

DOWNLOAD, EOP, STR$()

● CHR$( ){ XE "CHR$( )" }{ TC "CHR$( )"}

Description

This function returns the character with the specified ASCII code.

Syntax

CHR$(n)

Parameter	Description
n	The ASCII code

Example

```
DOWNLOAD “TEST.BAS”
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
A=65
WORD$=CHR$(A)
TEXT 100,100,”3”,0,1,1,WORD$
PRINT 1
EOP
```

See Also

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

● END{ XE "END" }{ TC "END"}

Description

This command states the end of program.

Syntax

END

Example

```
DOWNLOAD "DEMO.BAS"
SIZE 4,2
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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
```

See Also

DOWNLOAD, EOP

● EOF( ) { XE "EOF( )" }{ TC "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

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

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.0,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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
```

See Also

DOWNLOAD, EOP, OPEN, READ, SEEK

● OPEN { XE "OPEN" }{ TC "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.

Syntax

OPEN “Filename”, File handle

Parameter	Description
Filename	The file downloaded in the printer memory
File handle	Either 0 or 1.

Example

If a file by the name of “DATA” is to be downloaded,
The file format contains:

```
DOWNLOAD "DATA1",56,COMPUTER
2000
12
MOUSE
500
13
KEYBOARD
300
100

DOWNLOAD "DATA2",56,Computer
3000
32
Mouse
900
93
Keyboard
700
700
```

Save the above contents of data under the file name of “DATA”. Follow the steps below to download data to the printer

C:\>COPY DATA/B LPT1

If a file by name of “DEMO.BAS is to be downloaded, the file format contains:

DOWNLOAD "DEMO.BAS"


```
SIZE 3,1
GAP 0,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
I=1
Y=100
GOSUB OpenData
:Start
CLS
TEXT 10,10,"3",0,1,1,"*****OPEN COMMAND TEST*****"
ITEM$=""
READ 0,ITEM$,P,Q
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$(Q)+"="+STR$(P*Q)
Y=Y+100
PRINT 1
Y=100
IF EOF(0)=1 THEN GOSUB OpenData
IF EOF(0)=0 THEN GOTO Start
END
:OpenData
IF I=1 THEN OPEN "DATA1",0
IF I=2 THEN OPEN "DATA2",0
SEEK 0,0
IF I>2 THEN END
I=I+1
RETURN
EOP
DEMO
```

Saving the above contents of data under the file name of **“**DEMO**”.**

Follow the steps below to download data to the printer

<under MS-DOS mode>:

```
C:\>COPY DEMO/B LPT1
```

Execute DEMO.BAS in printer:

```
C:\>COPY CON LPT1
DEMO
<Ctrl><Z>
```

The above example instructs the printer to open the file “DATA1” and “DATA2” with same file handle of 0, and read items from the file.

See Also

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

● WRITE{ XE "WRITE" }{ TC "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$()

● READ{ XE "READ" }{ TC "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

```
DOWNLOAD "DATA1",20,COMPUTER
2000
12

DOWNLOAD "DATA2",16,Mouse
900
93

DOWNLOAD "DEMO.BAS"
SIZE 3,1
GAP 0,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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
```

See Also

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

● SEEK{ XE "SEEK" }{ TC “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

```
DOWNLOAD “DATA”,12,1234567890

DOWNLOAD “TEST.BAS”
SIZE 3,1
GAP 0,0
DENSITY 8
SPEED 3
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
```

See Also

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

● LOF( ){ XE “LOF( )” }{ TC “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

```
DOWNLOAD “DATA1”,10,1234567890

DOWNLOAD “DATA2”,15,ABCDEFGHIJKLMNO

DOWNLOAD “LofTest.BAS”
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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
```

See Also

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

● FREAD$( ){ XE “FREAD$( )” }{ TC “FREAD$( )”}

Description

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

Syntax

FREAD$ (file handle, byte)

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

Example

```
DOWNLOAD “DATA1”,10,1234567890

DOWNLOAD “DATA2”,15,ABCDEFGHIJKLMNO

DOWNLOAD “OPEN2.BAS”
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
  OPEN “DATA1”,0
  OPEN “DATA2”,1
  SEEK 0,0
  SEEK 1,0
   Y$=FREAD$(0,6)
   Z$=FREAD$(1,6)
  TEXT 10,260,”3”,0,1,1,”FREAD$(0,6) IS: “+Y$
  TEXT 10,320,”3”,0,1,1,”FREAD$(1,6) IS: “+Z$
  PRINT 1
EOP
```

See Also

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

● FOR…NEXT{ XE “FOR…NEXT” } LOOP{ TC “FOR…NEXT”}

Description

Loop is used to execute one or more lines of program repetitively. A loop counter value specifics the number of executions. Nested loop 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
NEXT

Parameter	Description
variable	The variable name ~~is~~ (up to 8 characters)
start	Integer or floating point numbers
end	Integer of floating point numbers
increment	Integer or floating point, positive or negative.

Example

```
DOWNLOAD “LOOP.BAS”
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 1
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
LOOP
```

See Also

DOWNLOAD, EOP

● IF…THEN…ELSE…ENDIF { XE "IF…THEN…ELSE…ENDIF" } LOOP{ TC "IF…THEN…ELSE…ENDIF"}

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 □eeping one single line in less than 255 characters.

Parameter	**Description**
condition	Available relational operator: <, >, =, <=, >=
statement	Only one statement is available in

Example


```
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.12,0
SPEED 4
DENSITY 8
DIRECTION 1
REFERENCE 0,0
OFFSET 0.00
SET CUTTER OFF
SET PEEL OFF
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)
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 “IFTHEN.BAS”
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
A=50
B=5
C$=""
D$=""

:L1
IF A>100 THEN GOTO L1 ELSE A=A+10
C$=STR$(A)+" IS SMALLER THAN 100"
TEXT 100,10,"4",0,1,1,C$
PRINT 1
END

:L2
A=A+B
D$=STR$(A)+" IS LARGER THAN 100"
TEXT 100,100,"4",0,1,1,D$
PRINT 1
GOTO L1
EOP
```

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 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.

Limitations:
The total numbers of nested IF …THEN …ELSE statement in a program can not exceed than 40.
The total numbers of nested IF …THEN …ELSE, FOR…NEXT, GOSUB RETURN in a program can not exceed than 40 loops.

See Also

DOWNLOAD, EOP

● GOSUB…RETURN{ XE "GOSUB…RETURN" }{ TC "GOSUB…RETURN”}

Description

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

```
DOWNLOAD "GOSUB1.BAS"
SIZE 3,3
GAP 0,0
DENSITY 8
SPEED 4
DIRECTION 0
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
```

See Also

DOWNLOAD, EOP, END, GOTO

● GOTO{ XE "GOTO" }{ TC "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
None	N/A

Example

```
DOWNLOAD "GOTO1.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 1
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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
```

See Also

DOWNLOAD, EOP, END, GOSUB…RETURN

● INP$( ){ XE "INP$( )" }{ TC "INP$( )"}

Description

One byte is received from a serial port through this function.

Syntax

INP$(n)

Parameter	Description
n	1: com1 port in printer

Example

```
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
T$=""
FOR I=1 TO 5
  T$=T$+INP$(1)
NEXT
TEXT 100,100,"4",0,1,1,"INP$(1)="+T$
PRINT 1
EOP
DEMO
12345
```

See Also

DOWNLOAD, EOP, END, INPUT, GOSUB…RETURN, GOTO

● INPUT{ XE "INPUT" }{ TC "INPUT"}

Description

This command receives data through serial port. This command is used with portable keyboard BBP11-SK.

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 prompt string is shown on keyboard LCD screen. 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

```
DOWNLOAD "INPUT1.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF

:START
  CLS
  A$=""
  TEXT 20,50,"3",0,1,1,"INPUT and BBP11-SK Test"
  INPUT "CODE 39 :",C39$
  INPUT "EAN 13:",12,E13$

  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
```

See Also

DOWNLOAD, EOP, END, GOTO

● INPUTFILTER{ XE "INPUTFILTER" }{ TC "INPUTFILTER"}

Description

This command alters the method by which INPUT and related commands receive information. The corresponding TCF command is

INPUT FILTER = 0 or 1

with 0 corresponding to OFF.

Syntax

SET INPUTFILTER Setting
INPUTPREFIX “Prefix”
INPUTSUFFIX “Suffix”

Parameter	**Description**
Setting	ON or OFF only
Prefix	INPUT command will begin reading after the designated prefix
Suffix	INPUT command will stop reading at the designated suffix

OFF is the default setting. In this mode, information being fed via the INPUT command stops immediately at the carriage feed OA OD (seen as a new line, produced via the computer “ENTER” key.)

Example:

SET INPUTFILTER ON
INPUTPREFIX “3”
INPUTSUFFIX “7”
INPUT A$

The user subsequently enters the string: 123456789

In this example, the A$ variable will be stored as 456

See Also

INPUT, INPUTPREFIX, INPUTSUFFIX, DOWNLOAD

● INPUTPREFIX{ XE "INPUTPREFIX" }{ TC "INPUTPREFIX"}

Description

This command alters the method by which INPUT and related commands receive information. The prerequisite for use of this command is INPUTFILTER. The corresponding TCF command is

DEFAULT INPUT PREFIX = “setting”

Syntax

INSTR$([Start,] Start string, End String)

Parameter	Description
Start (optional)	string, integer or float point variable to be used
Start string	origin for the INSTR measurement
End String	ending point for the INSTR ending measurement.

Example:

A$=”blank blank blank blank [[HELLO]] blank blank”
INSTR=(A$,”[[“,”]]”)

In this example, INSTR will be equal to 7.

See Also

INPUT, INPUTFILTER, INPUTSUFFIX, DOWNLOAD

● INPUTSUFFIX{ XE "INPUTSUFFIX" }{ TC "INPUTSUFFIX"}

Description

This command alters the method by which INPUT and related commands receive information. The prerequisite for use of this command is INPUTFILTER. The corresponding TCF command is

DEFAULT INPUT SUFFIX = “setting”

Syntax

INSTR$([Start,]Start string, End String)

Parameter	**Description**
Start (optional)	string, integer or float point variable to be used
Start string	origin for the INSTR measurement
End String	ending point for the INSTR ending measurement.

Example:

A$=”blank blank blank blank [[HELLO]] blank blank”
INSTR=(A$,”[[“,”]]”)

In this example, INSTR will be equal to 7.

See Also

INPUT, INPUTFILTER, INPUTPREFIX, DOWNLOAD

● REM{ XE "REM" }{ TC "REM"}

Description

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

Syntax

REM

Example

```
REM ******************************
REM This is a demonstration program*
REM ******************************
DOWNLOAD "REMARK.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 1
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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
```

See Also

DOWNLOAD, EOP, END

● OUT{ XE "OUT" }{ TC "OUT"}

Description

This command sends data through the printer serial port.

Syntax

OUT “prompt”, variable

Parameter	Description
prompt	Prompt which is shown on LCD screen.
Variable	The output message

Example

```
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
PRICE$="123456"
OUT "PRICE:",PRICE$
EOP
```

See Also

DOWNLOAD, EOP, END, ~#…~&

● GETKEY( ){ XE "GETKEY( )" }{ TC "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.

Model	PAUSE	FEED
BBP11-24L	X	1
BBP11-34L	X	1

Syntax

GETKEY()

Example

```
DOWNLOAD “DEMO4.BAS”
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
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

● INT( ){ XE "INT( )" }{ TC "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

```
DOWNLOAD "DEMO.BAS"
SIZE 4,2
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
REM **** To round up or down****
INPUT "Number:",Num
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
```

See Also

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

● LEFT$( ){ XE "LEFT$( )" }{ TC "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

```
DOWNLOAD "STR1.BAS"
SIZE 3.00,3.00
GAP 0.08,0.00
SPEED 4.0
DENSITY 8
SET CUTTER OFF
DIRECTION 0
REFERENCE 0,0
CLS
A$="BARCODE PRINTER DEMO PRINTING"
C$=LEFT$(A$,10)
TEXT 10,10,"3",0,1,1,A$
TEXT 10,100,"3",0,1,1,"10 LEFT 10 CHARS: "+C$
PRINT 1
EOP
```

See Also

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

● LEN( ){ XE "LEN( )" }{ TC "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

```
DOWNLOAD "DEMO.BAS"
SIZE 3.00,3.00
GAP 0.08,0.00
SPEED 4.0
DENSITY 8
SET CUTTER OFF
DIRECTION 0
REFERENCE 0,0
CLS
A$="BRADY WORLD WIDE"
B=LEN(A$)
TEXT 10,10,"3",0,1,1,A$
TEXT 10,50,"3",0,1,1,"STRING LENGTH="+STR$(B)
PRINT 1
EOP
```

See Also

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

● MID$( ){ XE "MID$( )" }{ TC "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 *m*th characters in the string. 1 <= m <= string length
n	The number of characters to return.

Example

```
DOWNLOAD "DEMO.BAS"
SIZE 3.00,3.00
GAP 0.08,0.00
SPEED 4.0
DENSITY 8
SET CUTTER OFF
DIRECTION 0
REFERENCE 0,0

CLS
A$="BRADY WORLDWIDE"
E$=MID$(A$,11,10)
TEXT 10,10,"3",0,1,1,A$
TEXT 10,200,"3",0,1,1,"10 MIDDLE CHARS: "+E$
PRINT 1
EOP
```

See Also

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

● RIGHT$( ){ XE "RIGHT$( )" }{ TC "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

```
DOWNLOAD "DEMO.BAS"
SIZE 3.00,3.00
GAP 0.08,0.00
SPEED 4.0
DENSITY 8
SET CUTTER OFF
DIRECTION 0
REFERENCE 0,0

CLS
A$="BRADY WORLDWIDE"
D$=RIGHT$(A$,10)
TEXT 10,10,"3",0,1,1,A$
TEXT 10,150,"3",0,1,1,"10 RIGHT CHARS: "+D$
PRINT 1
EOP
```

See Also

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

● LTRIM$(){ XE " LTRIM$()" }{ TC " LTRIM$()"}

Description

This command removes leading spaces from a string variable.

Syntax

LTRIM$(variable)

Parameter	Description
variable	string, integer or float point variable

Example:

A$=“ Sample”
B$=LTRIM$(A$)

In this example, B$ is now equal to “Sample”

See Also

RTRIM$(), TRIM$()

● RTRIM$(){ XE " RTRIM$()" }{ TC " RTRIM$()"}

Description

This command removes trailing spaces from a string variable.

Syntax

RTRIM$(variable)

Parameter	Description
variable	string, integer or float point variable

Example:

A$="Sample ”
B$=LTRIM$(A$)

In this example, B$ is now equal to “Sample”

See Also

LTRIM$(), TRIM$()

● TRIM$(){ XE " TRIM$()" }{ TC " TRIM$()"}

Description

This command removes both leading and trailing spaces from a string variable.

Syntax

TRIM$(variable)

Parameter	Description
variable	string, integer or float point variable

Example:

A$=“ Sample ”
B$=LTRIM$(A$)

In this example, B$ is now equal to “Sample”

See Also

LTRIM$(), TRIM$()

● INSTR(){ XE " INSTR()" }{ TC " INSTR()"}

Description

This command the extracts a length of a given string.

Syntax

INSTR$([Start,]Start string, End String)

Parameter	Description
Start (optional)	string, integer or float point variable to be used
Start string	origin for the INSTR measurement
End String	ending point for the INSTR ending measurement.

Example:

A$=”blank blank blank blank BEGIN HELLO END blank blank”
B$=INSTR(A$,”BEGIN“,”END”)

In this example, B$ will be equal to 7 (1 leading space, 5 character spaces, 1 trailing space)

See Also

● STR$( ){ XE "STR$( )" }{ TC "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

```
DOWNLOAD "DEMO.BAS"
SIZE 3.00,3.00
GAP 0,0.00
SPEED 4.0
DENSITY 8
SET CUTTER OFF
DIRECTION 0
REFERENCE 0,0
CLS
A$="BRADY WORLDWIDE"
F=100
G=500
H$=STR$(F+G)
TEXT 10,10,"3",0,1,1,A$
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
```

See Also

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

● VAL( ){ XE "VAL( )" }{ TC "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

```
DOWNLOAD "DEMO.BAS"
SIZE 3.00,3.00
GAP 0.00,0.00
SPEED 4.0
DENSITY 8
SET CUTTER OFF
DIRECTION 0
REFERENCE 0,0
CLS
 A$="BRADY WORLDWIDE"
 F$="100"
 G$="500"
 H=VAL(F$)+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
```

See Also

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

● BEEP{ XE "BEEP" }{ TC "BEEP"}

Description

This command issues a beep sound on portable keyboard. Printer sends the string 0x07 to BBP11-SK portable keyboard.

Syntax

BEEP

Parameter	Description
None	N/A

Example

```
DOWNLOAD “DEMO.BAS”
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
BEEP
INPUT “Text1 =”,TEXT1$
TEXT 100,100,”3”,0,1,1,TEXT1$
PRINT 1
EOP
```


Device Reconfiguration Commands{ TC “Device Reconfiguration Commands “}

● SET COUNTER{ XE "SET COUNTER" } { TC "SET COUNTER"}

Description

Counters can be a real counter or a variable.
This setting sets the counter number in program and their 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 51 counters available (@0~@50) in the printer.
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

```
SIZE 3,3
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET COUNTER @1 1
@1="00001"
SET COUNTER @2 5
@2="AB000001"
CLS
TEXT 50,50,"3",0,1,1,@1
BARCODE 50,100,"39",48,1,0,2,4,@2
PRINT 2,1
```

See Also

PRINT, TEXT, BARCODE

● SET CUTTER{ XE "SET CUTTER" }{ TC "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.

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

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

See Also

OFFSET, PRINT, SET PARTIAL_CUTTER

● SET PARTIAL_CUTTER { XE "SET PARTIAL_CUTTER" }{ TC "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.

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

Example

```
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

● SET BACK { XE "SET BACK" }{ TC "SET BACK"}

Description

This setting is used after SET CUTTER function.
This function prevents label backfeeding after a cut. Backfeeding after a cut backfeed when cut off

Syntax

SET BACK OFF/ON

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

Example

```
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

● SET KEY1, SET KEY2, SET KEY3{ XE "SET KEY1, SET KEY2, SET KEY3" }{ TC "SET KEY1, SET KEY2, SET KEY3"}

Description

This setting is used to enable/disable the KEY1/KEY2/KEY3 function. The default function of KEY1 is "MENU" key, KEY2 is "PAUSE" key and KEY3 is "FEED" key. Before setting KEY1/KEY2/KEY3 function otherwise, please disable KEY1/KEY2/KEY3 first. The setting will remain resident in the printer even when the printer is power off.

Model	KEY1
BBP11-24L series	FEED
BBP11-34L series	FEED

Syntax

SET KEY1 ON/OFF
SET KEY2 ON/OFF
SET KEY3 ON/OFF

Parameter	**Description**
ON	Enable KEY1 as MENU function
	Enable KEY2 as PAUSE function
	Enable KEY3 as FEED function
OFF	Disable KEY1 as MENU function
	Disable KEY2 as PAUSE function
	Disable KEY3 as FEED function

Note: The setting will remain in the printer even if the printer is power off.

Example

```
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

CLS
```



```
: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

● SET LED1, SET LED2, SET LED3{ XE " SET LED1, SET LED2, SET LED3" }{ TC "SET LED1, SET LED2, SET LED3"}

Description

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

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

Model	LED1	LED2	LED3	LED2 & LED3
BBP11-24L series	GREEN	GREEN	RED	ORANGE
BBP11-34L series	GREEN	GREEN	RED	ORANGE

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

Syntax

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

Example

```
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
```


● SET PEEL{ XE “SET PEEL” }{ TC “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.

Syntax

SET PEEL ON/OFF

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

Example

```
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

● SET TEAR

Description

This command is used to enable/disable feeding of labels to gap/black mark position for tearing off.

This setting will be saved in printer memory when turning off the power.

Syntax

SET TEAR ON/OFF

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

```
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

● SET GAP { XE "SET GAP" }{ TC “SET GAP”}

Description

This setting sets the gap sensor emission sensitivity. The printer initiates automatic gap sensor calibration 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 using when the Black Mark is the separation in front of the label and which can’t be detected by Black Mark sensor. The parts of the media where can be passed through by GAP sensor are defined to be the printable area, otherwise there will be defined to the GAP of the media.
OBVERSE	Disable the “SET GAP REVERSE” function.

Printer model	Range	SET GAP REVERSE SET GAP OBVERSE SET GAP AUTO
BBP11-24L & BBP1-34L	0~31 (Gap) 0~3 (Bline)	X

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

● SET HEAD{ XE "SET HEAD" } { TC “SET HEAD”}

Description

This setting is used to enable/disable head open sensor. If the head open sensor is closed, 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

● SET RIBBON{ XE "SET RIBBON" } { TC “SET RIBBON”}

Description

This setting is used to enable/disable ribbon sensor detection. (Thermal Transfer Printing/Thermal Direct 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.

Syntax

SET RIBBON ON /OFF

Parameter	Description
ON	Thermal transfer printing
OFF	Thermal direct printing

Example

```
REM ***Direct printing****
SIZE 4,4
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
SET RIBBON OFF
CLS
BARCODE 100,100,"39",48,1,0,2,5,"CODE 39”
PRINT 1
```


● SET COM1{ XE "SET COM1" }{ TC “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.

● SET PRINTKEY { XE “SET PRINTKEY”}{ TC “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

Execute:

```
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 SET PRINTKEY AUTO	1.) PRINT 2	Label 1~2
	2.) Press FEED key	Label 3~4

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

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

● SET REPRINT { XE “SET REPRINT”}{ TC “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

● PEEL { XE “PEEL” }{ TC “PEEL”}

Description

This command obtains the status of the peel-off sensor. This attribute is read only.

Syntax

PEEL

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

Example

```
DOWNLOAD “DEMO.BAS”
SIZE 4,1
GAP 0,0
SPEED 4
DENSITY 8
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
```


● LED1, LED2, LED3{ XE " LED1, LED2, LED3" }{ TC "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.

Model	LED1	LED2	LED3	LED2 & LED3
BBP11-24L series	GREEN	GREEN	RED	ORANGE
BBP11-34L series	GREEN	GREEN	RED	ORANGE

Syntax

LEDm=n

Parameter	**Description**
m	m=1, LED1 m=2, LED2 m=3, LED3
n	0: turn off LED 1: light on LED

Example

```
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
```


● KEY1, KEY2, KEY3{ XE " KEY1, KEY2, KEY3" }{ TC “KEY1, KEY2, KEY3”}

Description

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

Model	KEY0	KEY1	KEY2	KEY3	KEY4	KEY5	KEY6
BBP11-24L series		FEED					
BBP11-34L series		FEED					

Syntax

KEYm=n

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

Example

```
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
```


Printer Global Variables{ TC "Printer Global Variables" }

- **@LABEL{ XE "@LABEL" }{ TC "@LABEL"}**

Description

This variable counts how many pieces of labels have been printed. This attribute cannot be initialized if the printer is reset, and 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

```
DOWNLOAD "DEMO.BAS"
SIZE 4,2.5
GAP 2 mm,0
SPEED 6
DENSITY 12
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
```


● YEAR{ XE "YEAR" }{ TC “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

```
DOWNLOAD "SetYear.BAS"
REM *****Set Year Parameter to RTC*****
YEAR=05
EOP
SetYear

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

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

REM *****Print*****
TEXT 10,10,"5",0,1,1,"YEAR1="+YEAR$
TEXT 10,110,"5",0,1,1,"YEAR2="+STR$(Y)
TEXT 10,210,"5",0,1,1,"YEAR3="+STR$(YEAR)
PRINT 1
EOP
DEMO
```

See Also

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

● MONTH{ XE "MONTH" }{ TC “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

```
DOWNLOAD "SetMonth.BAS"
REM *****Set Month Parameter to RTC*****
MONTH=05
EOP
SetMonth

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

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

REM *****Print*****
TEXT 10,10,"5",0,1,1,"MONTH1="+MONTH$
TEXT 10,110,"5",0,1,1,"MONTH2="+STR$(M)
TEXT 10,210,"5",0,1,1,"MONTH3="+STR$(MONTH)
PRINT 1
EOP
DEMO
```

See Also

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

● DATE{ XE "DATE" }{ TC “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

```
DOWNLOAD "SetDate.BAS"
REM *****Set Date Parameter to RTC*****
DATE=30
EOP
SetDate

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

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

REM *****Print*****
TEXT 10,10,"5",0,1,1,"DATE1="+DATE$
TEXT 10,110,"5",0,1,1,"DATE2="+STR$(D)
TEXT 10,210,"5",0,1,1,"DATE3="+STR$(DATE)
PRINT 1
EOP
DEMO
```

See Also

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

● WEEK{ XE "WEEK" }{ TC “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

Write attribute: WEEK=3
Read attribute: A=WEEK
Range: 1(Sunday)~7(Saturday)

Example

```
DOWNLOAD "SetWeek.BAS"
REM *****Set Week Parameter to RTC*****
WEEK=6
EOP
SetWeek

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

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

REM *****Print*****
TEXT 10,10,"5",0,1,1,"WEEK1="+WEEK$
TEXT 10,110,"5",0,1,1,"WEEK2="+STR$(W)
TEXT 10,210,"5",0,1,1,"WEEK3="+STR$(WEEK)
PRINT 1
EOP
DEMO
```

See Also

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

● HOUR{ XE "HOUR" }{ TC “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

```
DOWNLOAD "SetHour.BAS"
REM *****Set Hour Parameter to RTC*****
HOUR=11
EOP
SetHour

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

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

REM *****Print*****
TEXT 10,10,"5",0,1,1,"HOUR1="+HOUR$
TEXT 10,110,"5",0,1,1,"HOUR2="+STR$(H)
TEXT 10,210,"5",0,1,1,"HOUR3="+STR$(HOUR)
PRINT 1
EOP
DEMO
```

See Also

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

● MINUTE{ XE "MINUTE" }{ TC “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

```
DOWNLOAD "SetMinute.BAS"
REM *****Set Minute Parameter to RTC*****
MINUTE=59
EOP
SetMinute

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

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

REM *****Print*****
TEXT 10,10,"5",0,1,1,"MINUTE1="+MINUTE$
TEXT 10,110,"5",0,1,1,"MINUTE2="+STR$(MIN)
TEXT 10,210,"5",0,1,1,"MINUTE3="+STR$(MINUTE)
PRINT 1
EOP
DEMO
```

See Also

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

● SECOND{ XE "SECOND" }{ TC “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

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

DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

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

REM *****Print*****
TEXT 10,10,"5",0,1,1,"SECOND1="+SECOND$
TEXT 10,110,"5",0,1,1,"SECOND2="+STR$(SEC)
TEXT 10,210,"5",0,1,1,"SECOND3="+STR$(SECOND)
PRINT 1
EOP
DEMO
```

See Also

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

● @YEAR{ XE "@YEAR" }{ TC “@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

```
REM *****Set @YEAR*****
@YEAR=”05”

REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,”5”,0,1,1,”@YEAR”
TEXT 310,10,”5”,0,1,1,@YEAR
PRINT 1
```

See Also

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

● @MONTH{ XE “@MONTH” }{ TC “@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

```
REM *****Set @MONTH*****
@MONTH="12"

REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@MONTH"
TEXT 310,10,"5",0,1,1,@MONTH
PRINT 1
```

See Also

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

● @DATE{ XE "@DATE" }{ TC “@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

```
REM *****Set @DATE*****
@DATE="31"

REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@DATE"
TEXT 310,10,"5",0,1,1,@DATE
PRINT 1
```

See Also

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

● @DAY{ XE "@DAY" }{ TC “@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

Write attribute: @DAY=”3”

Read attribute: @DAY

Range: 1(Sunday)~7(Saturday)

Example

```
REM *****Set @DAY*****
@DAY="5"

REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@DAY"
TEXT 310,10,"5",0,1,1,@DAY
PRINT 1
```

See Also

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

● @HOUR{ XE "@HOUR" }{ TC “@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

```
REM *****Set @HOUR*****
@HOUR=”23”

REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,”5”,0,1,1,”@HOUR”
TEXT 310,10,”5”,0,1,1,@HOUR
PRINT 1
```

See Also

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

● @MINUTE{ XE “@MINUTE” }{ TC “@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

```
REM *****Set @MINUTE*****
@MINUTE="59"

REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@MINUTE"
TEXT 310,10,"5",0,1,1,@MINUTE
PRINT 1
```

See Also

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

● @SECOND{ XE "@SECOND" }{ TC “@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

```
REM *****Set @SECOND*****
@SECOND="59"

REM *****Print*****
SIZE 3,3
GAP 0,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@SECOND"
TEXT 310,10,"5",0,1,1,@SECOND
PRINT 1
```

See Also

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

W.H.Brady N.V
Industriepark C3,
Lindestraat 20
B9240 Zele- Belgium

Website:
www.bradyeurope.com