EPSON

PX-8
COMMAND SUMMARY

H8492025-0
Y203990062

PX-8
COMMAND SUMMARY

NOTICE:

* All rights reserved. Reproduction of any part of this manual in any form whatsoever without EPSON's express written permission is forbidden.
* The contents of this manual are subject to change without notice.
* All efforts have been made to ensure the accuracy of the contents of this manual.
  However, should any errors be detected, EPSON would greatly appreciate being informed of them.
* The above notwithstanding, EPSON can assume no responsibility for any errors in this manual or their consequences.

CP/M Commands

Built-in commands

DIR
Lists the names of all files on the currently logged in drive.

DIR dr:
Lists the names of all files on the specified drive.

DIR dr:afn
Lists the names of all files satisfying the specified ambiguous file name (afn) on the display screen.

ERA dr:ufn or ERA dr:afn
Erases a specific file or all files that satisfy a specific ambiguous file name from the specified disk device.

REN dr:ufn1 = ufn2
Changes the name of a file which is on the specified drive from ufn2 to ufn1.

SAVE n dr:ufn
Saves the contents of the specified number of pages of the transient program area to a disk device under the specified file name. (Each page consists of 256 bytes of memory).

TYPE dr:ufn
Displays the contents of the specified ASCII file (a file stored in ASCII codes) in the specified disk device.

USER n
Allows different users to specify their own logical directories on the same disk device. n is the user number (an integer from 0 to 15) which specifies a logical directory.

Transient commands

CONFIG
The CONFIG command sets the system environment of the PX-8 and determines various default settings.

FILINK
The FILINK program transfers files between PX-8 and another computer (such as another PX-8, QX-10, etc.) through the RS-232C communication line.

PIP

Activates the PIP command.

PIP d:=s:[afn / ufn]

Copies the specified file(s) from drive s to drive d.

PIP [LST: / PUN: / CON:] =s:ufn

Transfers the specified file from drive S to the specified device.

PIP d:ufn=RDR:

Transfers data from RDR: to the specified file.

PIP d:newname. typ=s1:ufn1, s2:ufn2, ...

Concatenates the specified files to generate a new file.

PIP parameters

[B] (Block)

When specified, PIP performs block mode transfer.

[Dn] (Delete)

When specified, PIP deletes characters exceeding column n of each line.

[E] (Echo)

When specified, data transferred is also output to the console.

[F] (Form feed)

When specified, form feed characters (ØCH) are removed from data transferred.

[Gn]

When specified, data can be transferred from another user area.

[H] (Hex format)

When specified, PIP checks to confirm that data transferred is in Intel HEX format. (If not, operation is terminated.)

[I] (Ignore null)

When specified, PIP ignores null records (ØØH) and checks to confirm that data transferred is Intel HEX format. (If not, operation is terminated.)

[L] (Lower case)

When specified, all upper case characters are coverted to lower case characters.

[N] (Line number)

When specified, line numbers are added to the beginning of each line. Specifying "N2" adds zeros to the beginning of each line number.

[O] (Object files)

When specified, PIP ignores the physical end of file code (1AH) during concatenation and transfer. This makes it possible to transfer files other than ASCII files.

[Pn] (Page eject)

When specified, PIP makes a page eject every n lines. When n is 1 or omitted, 60 is assumed.

[Qstring ^Z] (Quit)

When specified, PIP quits transfer when the specified string is detected.

[R] (Read)

Makes it possible to transfer .SYS files. (The [W] parameter is set automatically when [R] is specified.)

[Sstring ^Z] (Start)

When specified, PIP searches the data to be transferred for the specified string, then starts transfer from the point at which it is detected.

[Tn] (Tab)

When specified, the tab width for transfer is set to n columns.

[U] (Upper case)

When specified, all lower case characters are converted to upper case characters.

[V] (Verify)

When specified, each file is verified as it is copied.

[W] (Write in R/O)
When specified, the R/O attribute is ignored for destination files.

[Z] (Zero parity)
When specified, parity bits in data received are reset to zero.

STAT
Displays the amount of free space on the disk in the currently logged-in drive and other drives which have been at least once used, as well as their R/W attribute.

STAT dr:
Displays the amount of free space on the disk in the specified drive.

STAT dr:filename.typ
Displays the size and attributes of the specified file.

STAT dr:filename.typ $S
Displays the size and attributes of the specified file in detail.

STAT dr:filename.typ $R/O
Sets the specified file to read only.

STAT dr:filename.typ $R/W
Makes it possible to read or write the specified file.

STAT dr:filename.typ $SYS
Sets the SYS attribute for the specified file.

STAT dr:filename.typ $DIR
Sets the DIR attribute for the specified file.

(Device related formats)

STAT DEV:
Displays the current physical-to-logical device assignments (that is, the contents of IOBYTE).

STAT VAL:
Displays instructions for specifying the operand of the STAT command.

STAT DSK:

Displays the status of the currently logged-in drive and other drives which have been used at least once.

STAT USR:

Displays the current user number and user numbers which have active files on the current disk.

STAT dr: = R/O

Sets the specified drive to read only. The read only state remains effective until a cold or warm start is made.

STAT logical: = physical:

Assigns the specified physical device to the specified logical device.

SUBMIT filename parameters

Executes the commands in command procedure file "filename.SUB" using parameters.

TERM

The TERM program makes it possible to connect the PX-8 to a host computer through a RS-232C communication line for use as a terminal.

XSUB

Extends the function of SUBMIT.

BASIC Commands and statements

ABS

FORMAT ABS(X)

PURPOSE Returns the absolute value of expression X.

EXAMPLE A = ABS(−1Ø)

ALARM

FORMAT ALARM [<date>, <time>, <string> [,W]]

PURPOSE Specifies the alarm or wake time.

EXAMPLE ALARM "Ø1/31/84", "1Ø:ØØ:ØØ", "Staff meeting"
ALARM "**/**/**", "Ø8:ØØ:ØØ";
"A:MORNING" + chr$(13), W

ALARM$

FORMAT ALARM$ (<function>)

PURPOSE Returns information about the ALARM setting.

EXAMPLE PRINT ALARM$(1)

ASC

FORMAT ASC(X$)

PURPOSE Returns the numeric value which is the ASCII code for the first character of string X$.

EXAMPLE A = ASC("A")

ATN

FORMAT ATN(X)

PURPOSE Returns the arc tangent in radians of X.

EXAMPLE A = ATN(X)

AUTO

FORMAT AUTO [<line number>][,[<increment>]]

PURPOSE Initiates automatic generation of program line numbers.

EXAMPLE AUTO
AUTO 1ØØ, 5Ø

AUTO START

FORMAT AUTO START <auto start string>

PURPOSE Sets the auto start string.

EXAMPLE AUTO START"A:MELODY" + CHR$(13)

BEEP

FORMAT BEEP [<duration>]

PURPOSE Sounds the PX-8's built-in speaker.

EXAMPLE BEEP ON
BEEP OFF
BEEP 100

CALL

FORMAT CALL <variable name> [(<argument list>)]

PURPOSE Starts execution of a machine language subroutine.

EXAMPLE CALL J(X)

CDBL

FORMAT CDBL(X)

PURPOSE Converts numeric expression X to a double precision number.

EXAMPLE A# = CDBL(X!)

CHAIN

FORMAT CHAIN [MERGE] <file name>[,[<line number exp>][,ALL][,DELETE<range>]]

PURPOSE Calls the BASIC program designated by <file name> and passes variables to it from the calling program.

EXAMPLE CHAIN "SAMPLE"
CHAIN "A:SAMPLEZ3",,ALL
CHAIN MERGE "SUB",100

CHR$

FORMAT CHR$(J)

PURPOSE Returns the character whose ASCII code equals the value of integer expression J.

EXAMPLE A$ = CHR$(65)

CINT

FORMAT CINT(X)

PURPOSE Rounds the decimal portion of numeric expression X to the nearest whole number and returns the equivalent integer value.

EXAMPLE A% = CINT(5.6)

CLEAR

FORMAT CLEAR[[<dummy>][,[<upper memory limit>][,<stack area size>]]]

PURPOSE Clears all numeric and string variables. When options are specified, also reserves an area in memory for machine language programs and sets the stack area size.

EXAMPLE 1Ø CLEAR ,&HC4ØØ

CLOSE

FORMAT CLOSE[[#]<file number>[,[#]<file number...>]]

PURPOSE Terminates access to files.

EXAMPLE CLOSE #3

CLS

FORMAT CLS

PURPOSE Clears the currently selected virtual screen.

EXAMPLE CLS

COMMON

FORMAT COMMON<list of variables>

PURPOSE Passes variables to a CHAINed program.

EXAMPLE COMMON B, A$( )

CONT

FORMAT CONT

PURPOSE Resumes execution of a program interrupted by STOP, END or the STOP key.

EXAMPLE CONT

COPY

FORMAT COPY

PURPOSE Outputs the contents of the LCD screen to the printer.

EXAMPLE
```
COPY
```

COS

FORMAT COS(X)

PURPOSE Returns the cosine of angle X, where X is in radians.

EXAMPLE
```
A# = COS(1.570796326794897)
```

CSNG(X)

FORMAT CSNG(X)

PURPOSE Converts numeric expression X to a single precision number.

EXAMPLE
```
A! = CSNG(16%)
```

CSRLIN

FORMAT CSRLIN [(<function>)]

PURPOSE Returns the vertical character coordinate of the cursor on the virtual screen or the vertical position of the first line of the screen window in the vertical screen.

EXAMPLE
```
A% = CSRLIN
```

CVI/CVS/CVD

FORMAT
```
CVI (<2-byte string>
CVS (<4-byte string>
CVD(<8-byte string>
```

PURPOSE Converts ASCII representations of BCD code to numeric values.

EXAMPLE
```
PRINT CVI (CHR$(5) + CHR$(0))
PRINT CVS (CHR$(0) + CHR$(0)
          + CHR$(32) + CHR$(131))
PRINT CVD (CHR$(0) + CHR$(0)
          + CHR$(0) + CHR$(0)
          + CHR$(0) + CHR$(0)
          + CHR$(32) + CHR$(131)
```

DATA

FORMAT DATA<list of constants>

PURPOSE Stores numeric and string constants which are substituted into variables by the READ statement.

EXAMPLE DATA PX,8,EPSON

DATE$

FORMAT As a statement
DATE$ = "<MM>/<DD>/<YY>"
As a variable
X$ = DATE$

PURPOSE As a statement, sets the date of the PX-8's calendar clock. As a variable, returns the date of the PX-8's built-in clock.

EXAMPLE DATE$ = "Ø1/28/84"
A$ = DATE$

DAY

FORMAT As a statement
DAY = <W>
As a variable
X% = DAY

PURPOSE As a statement, sets the day of the week of the PX-8's built-in clock. As a variable, returns the day of the week from the PX-8's built-in clock.

EXAMPLE DAY = 6
A% = DAY

DEF FN

FORMAT DEF FN<name>(<parameter list>)
= <function definition>

PURPOSE Defines and names user-written functions.

EXAMPLE DEF FNA(X, Y) = X*3/(Y + 2)

DEFINT/SNG/DBL/STR

FORMAT DEF INT <range(s) of letters>
DEF SNG <range(s) of letters>
DEF DBL <range(s) of letters>
DEF STR <range(s) of letters>

PURPOSE Declares type of variables as integer, single precision, double precision, or string.

EXAMPLE DEFINT I-N, W-Z

DEF USR

FORMAT DEF USR[<digit>] = <integer expression>

PURPOSE Used to specify the starting addresses of user-written machine language subroutines.

EXAMPLE DEF USR1 = &HCØØØ

DELETE

FORMAT DELETE [<line number 1>][– <line number 2>]

PURPOSE Deletes specified lines from a BASIC program.

EXAMPLE DELETE 4Ø
DELETE 4Ø-1ØØ
DELETE -4Ø

DIM

FORMAT DIM <list of subscripted varialbles>

PURPOSE Specifies the maximum range of array subscripts and allocates space for storage of array variables.

EXAMPLE DIM G%(25), F%(25)

DSKF

FORMAT DSKF (<drive name>)

PURPOSE Returns the number of kilobytes of free space in specified disk device.

EXAMPLE A% = DSKF("A:")

EDIT

FORMAT EDIT <line number>

PURPOSE Enters the EDIT mode at the specified line.

EXAMPLE EDIT 4Ø

END

FORMAT END

PURPOSE Terminates program execution, closes all files, and returns BASIC to the command level.

EXAMPLE END

EOF

FORMAT EOF (<file number>)

PURPOSE Returns a value indicating whether the end of a sequential file has been reached during sequential input.

EXAMPLE IF EOF(1) THEN 1ØØ

ERASE

FORMAT ERASE <list of variables>

PURPOSE Cancels array definitions made with the DIM statement.

EXAMPLE ERASE A, B

ERL

FORMAT ERL

PURPOSE Returns the line number of a command/statement causing an error during program execution.

EXAMPLE B=ERL

ERR

FORMAT ERR

PURPOSE Returns the error code of errors occurring during command or satement execution.

EXAMPLE A=ERR

ERROR

FORMAT ERROR <integer expression>

PURPOSE Simulates the occurrence of BASIC errors or allows the user to define his own error codes.

EXAMPLE ERROR 255

EXP

FORMAT EXP (X)

PURPOSE Returns the value of the natural base *e* to the power of X.

EXAMPLE A=EXP(X)

FIELD

FORMAT FIELD[#]<file number>,<field width> AS<string variable>,<field width>AS <string variable>,...

PURPOSE Assigns positions in a random file buffer for use as variables.

EXAMPLE
```
FIELD 1, 20 AS N$, 10 AS ID$, 40 AS ADD$
```

FILES

FORMAT FILES [<ambiguous file name>]

PURPOSE Displays the names of files satisfying the <ambiguous file name>.

EXAMPLE
```
FILES "L???????.BAS"
FILES "E:"
FILES "A:D???.*"
```

FIX

FORMAT FIX(X)

PURPOSE Returns the integer portion of numeric expression X.

EXAMPLE
```
A=FIX(X)
```

FOR...NEXT

FORMAT
```
FOR <variable> = <expression 1>TO
<expression 2>
[STEP <expression 3>]
    :
NEXT [<variable>][,<variable>...]
```

PURPOSE Allows the series of instructions between FOR and NEXT to be repeated a given number of times.

EXAMPLE
```
FOR I=1 TO 100 STEP 4
    A%=CSRLIN
    :
NEXT I
```

FRE

FORMAT
```
FRE(X)
FRE(X$)
```

PURPOSE Returns the number of bytes of memory which are not being used by BASIC.

EXAMPLE
```
PRINT FRE(0)
PRINT FRE(A$)
```

GET

FORMAT GET [#]<file number>[,<record number>]

PURPOSE Reads a record from a random disk file.

EXAMPLE GET#1, X

GOSUB...RETURN

FORMAT
```
GOSUB <line number>
      :
      :
RETURN
```

PURPOSE Used for branching to and returning from subroutines.

EXAMPLE
```
GOSUB <line number>
      :
      :
RETURN
```

GOTO or GO TO

FORMAT GOTO <line number>
GO TO <line number>

PURPOSE Transfers program execution to the program line specified by <line number>.

EXAMPLE GOTO 200

HEX$

FORMAT HEX$(X)

PURPOSE Returns a character string representing the hexadecimal value of X.

EXAMPLE PRINT HEX$ (44323)

IF...THEN[...ELSE]/IF...GOTO

FORMAT IF <logical expression>

THEN	<statement>	[ELSE	<statement>	]
	<line No.>		<line No.>	
GOTO <line No.>				

PURPOSE Changes the flow of program execution according to the result of a logical expression.

EXAMPLE IF A=B THEN PRINT "A=B" ELSE PRINT "A< >B"

INKEY$

FORMAT INKEY$

PURPOSE Checks the keyboard buffer and returns one character (or a null string if no key has been pressed).

EXAMPLE A$ = INKEY$

INP

FORMAT INP (J)

PURPOSE Returns one byte of data from machine port J.

EXAMPLE A = INP(176)

INPUT

FORMAT

INPUT[;] [<prompt string> | ; | ,] <list of variables>

PURPOSE Allows values to be substituted into variables from the keyboard during program execution.

EXAMPLE INPUT "NAME";A$

INPUT

FORMAT INPUT # <file number>, <variable list>

PURPOSE Reads data into variables from a sequential file.

EXAMPLE INPUT #1, A$, B$, C$

INPUT$

FORMAT INPUT$(X[, [#]<file number>])

PURPOSE Reads a string of X characters from the keyboard buffer or file opened under <file number>.

EXAMPLE A$ = INPUT$(1)
A$ = INPUT$(10, #1)

INSTR

FORMAT INSTR([J,]X$,Y$)

PURPOSE Searches for string Y$ in string X$ and returns a number indicating the position at which it was found.

EXAMPLE A = INSTR(X$, "ABC")

INT

FORMAT INT(X)

PURPOSE Subtracts the decimal portion of X from X and returns the integer value which is the result.

EXAMPLE A = INT(−B/3)

KEY

FORMAT KEY<n>,<X$>

PURPOSE Defines the functions of programmable function keys.

EXAMPLE KEY 2, "LIST" + CHR$(13)

KEY LIST/KEY LLIST

FORMAT KEY LIST
KEY LLIST

PURPOSE Outputs a list of the programmable function key definitions to the display or printer.

EXAMPLE KEY LIST

KILL

FORMAT KILL <file descriptor>

PURPOSE Deletes the disk device file specified by <file descriptor>.

EXAMPLE KILL "FILE3.BAS"
KILL "D:SAMPLE1.BAS"

LEFT$

FORMAT LEFT$ (X$, J)

PURPOSE Returns a string of J characters from the left end of string X$.

EXAMPLE A$ = LEFT$ (X$, 5)

LEN

FORMAT LEN (X$)

PURPOSE Returns the number of characters in string X$.

EXAMPLE A = LEN (X$)

LET

FORMAT [LET]<variable> = <expression>

PURPOSE Assigns the value of an expression to a variable.

EXAMPLE
```
PI = 3.14159
LET PI = 3.14159
```

LINE

FORMAT LINE [[STEP] (X1, Y1)]-[STEP](X2, Y2)[,[<function code>][,[B[F]][,<line style>]]]

PURPOSE Draws a straight line between two specified points.

EXAMPLE
```
LINE (Ø,Ø)-(5ØØ,3ØØ)
LINE -STEP (2Ø,2Ø)
LINE (25,25)-(5ØØ,2ØØ),4,,&HAAA
```

LINE INPUT

FORMAT LINE INPUT[;][<prompt string>;] <string variable>

PURPOSE Assigns character strings entered from the keyboard during program execution to variables.

EXAMPLE
```
LINE INPUT "ENTER NAME (LAST, FIRST);A$
```

LINE INPUT

FORMAT LINE INPUT #<file number>,<string variable>

PURPOSE Reads lines of data into variables from a sequential disk file.

EXAMPLE
```
LINE INPUT #1, A$
```

LIST

FORMAT
```
LIST[*][<line number>][-][<line number>]
LIST[*][<file descriptor>,][<line number>][-][<line number>]
```

PURPOSE Lists BASIC program lines on the display or printer.

EXAMPLE
```
LIST
LIST -5Ø
LIST 5Ø-
LIST 5Ø-2ØØ
LIST "LPTØ:"
```

LLIST

FORMAT LLIST [*][<line number>][-][<line number>]

PURPOSE Outputs a program list to the printer.

EXAMPLE LLIST 1000 - 2000

LOAD

FORMAT LOAD <file descriptor>[,R]

PURPOSE Loads a program into memory.

EXAMPLE LOAD "LNINPT"
LOAD "B:LNINPT.BAS"

LOC

FORMAT LOC (<file number>)

PURPOSE Returns the random access file record number following that used by the last GET or PUT statement, or the number of file sectors read/written since a sequential file was opened.

EXAMPLE A = LOC(1)

LOCATE

FORMAT LOCATE [<X>][,[<Y>][,<cursor switch>]

PURPOSE Moves the cursor to specified virtual screen coordinates.

EXAMPLE LOCATE 1,1,0

LOF

FORMAT LOF(<file number>)

PURPOSE Returns the size of a file in sectors.

EXAMPLE A = LOF(1)

LOG

FORMAT LOG(X)

PURPOSE Returns the natural logarithm of X.

EXAMPLE PRINT LOG(2.7812818)

LOGIN

FORMAT LOGIN <program area no.>[,R]

PURPOSE Switches to the specified BASIC program area. If R is specified, executes the program in that area.

EXAMPLE
```
LOGIN 2
LOGIN 3,R
```

LPOS

FORMAT LPOS (X)

PURPOSE Returns the current position of the buffer pointer in the printer output buffer. (X is a dummy argument.)

EXAMPLE
```
A=LPOS(X)
```

LPRINT

FORMAT LPRINT [<list of expressions>]

PURPOSE Outputs data to a printer connected to the PX-8.

EXAMPLE
```
LPRINT "EPSON PX-8"
```

LPRINT USING

FORMAT LPRINT USING <format string>;<list of expressions>

PURPOSE Outputs data to a printer in a specific format.

EXAMPLE
```
LPRINT USING "# # # #";A;B
```

LSET/RSET

FORMAT LSET <string variable> = <string expression>
RSET <string variable> = <string expression>

PURPOSE Prepares character data for storage in a random access file by moving it into a random file buffer.

EXAMPLE
```
LSET A$=B$
```

MENU

FORMAT MENU

PURPOSE Returns BASIC to the BASIC program menu.

EXAMPLE
```
MENU
```

MERGE

FORMAT MERGE <file descriptor>

PURPOSE Merges a program from a file (disk device, disk image RAM or COMØ:) with the program currently in memory.

EXAMPLE
```
MERGE "TEST1"
MERGE "COMØ:"
```

MID$

FORMAT As a statement
MID$ (<string expl>,n[,m]) = <string exp2>
As a function
MID$ (X$,J[,K])

PURPOSE As a statement, replaces characters from position n in <string expl> with the first m characters of <string exp2>. As a function, returns K characters from the middle of X$, starting with character J.

EXAMPLE
```
MID$ (A$,5,7) = B$
A$ = MID$("ABCDEFG",3,3)
```

MKI$/MKS$/MKD$

FORMAT MKI$ (<integer expression>)
MKS$ (<single precision expression>)
MKD$ (<double precision expression>)

PURPOSE Converts numeric values to strings for storage in random access files.

EXAMPLE
```
A$ = MKI$(X%)
A$ = MKS$(X!)
A$ = MKD$(X#)
```

MOUNT

FORMAT MOUNT

PURPOSE Reads the microcassette tape directory into memory and prepares the microcassette drive for access as a disk device.

EXAMPLE
```
MOUNT
```

NAME

FORMAT NAME <old file name> AS <new file name>

PURPOSE Changes the name of a disk device file.

EXAMPLE
```
NAME "SAMPLE1.BAS"
AS "SAMPLE2.BAS"
```

NEW

FORMAT NEW

PURPOSE Deletes the program in the currently selected memory area and clears all variables.

EXAMPLE NEW

OCT$

FORMAT OCT$ (X)

PURPOSE Returns a character string representing the octal value of X.

EXAMPLE A$ = OCT$(9999)

ON ERROR GOTO

FORMAT ON ERROR GOTO [<line number>]

PURPOSE Causes program execution to branch to the first line of an error handling subroutine when an error occurs.

EXAMPLE ON ERROR GOTO 1000

ON...GOSUB/ON...GOTO

FORMAT ON <numeric expression> GOTO <list of line numbers>
ON <numeric expression> GOSUB <list of line numbers>

PURPOSE Branches to one of several specified program line numbers depending on the value returned for <expression>.

EXAMPLE ON A GOSUB 100,200,500,1000

OPEN

FORMAT OPEN "<mode>",[#]<file number>, <file descriptor>,[<record length>]

PURPOSE Opens a disk file or other device for input or output.

EXAMPLE OPEN"O",#1,"CLIENTS.DAT"

OPTION BASE

FORMAT OPTION BASE $\left|\begin{matrix}0\\1\end{matrix}\right|$

PURPOSE Declares the minimum value for array subscripts.

EXAMPLE OPTION BASE 1

OPTION COUNTRY

FORMAT OPTION COUNTRY <character string>

PURPOSE Specifies the international character set to be used for keyboard input/output, LCD display, and output to the printer.

EXAMPLE OPTION COUNTRY "U"
OPTION COUNTRY "england"

OPTION CURRENCY

FORMAT OPTION CURRENCY <string expression>

PURPOSE Changes the currency symbol.

EXAMPLE OPTION CURRENCY "@"

OUT

FORMAT OUT <integer expression 1>,<integer expression 2>

PURPOSE Outputs the value of <integer expression 2> to the machine output port specified in <integer expression 1>.

PCOPY

FORMAT PCOPY <program area no.>

PURPOSE Copies the program in the current program area to another program area.

EXAMPLE PCOPY 3

PEEK

FORMAT PEEK (J)

PURPOSE Returns one byte of data from memory location J.

EXAMPLE A = PEEK (&HEØØØ)

POINT

FORMAT POINT (horizontal position, vertical position)

PURPOSE Returns the setting of the display dot at the specifie graphic screen location.

EXAMPLE PRINT POINT (1Ø,1Ø)

POKE

FORMAT POKE <integer expression 1>,
<integer expression 2>

PURPOSE Writes the data byte specified by <integer expression 2> to the memory address specified by <integer expression 1>.

EXAMPLE POKE &HCØØ1,A

POS

FORMAT POS (<file no.>)

PURPOSE Returns the current position of the buffer pointer in the file output buffer.

EXAMPLE PRINT POS (1)

POWER

FORMAT POWER OFF[,RESUME]
POWER[<duration>]
POWER CONT

PURPOSE Turns off the power or sets the auto power-off function.

EXAMPLE POWER OFF, RESUME
POWER 1Ø
POWER CONT

PRESET

FORMAT PRESET [STEP](X,Y)[,<function code>]

PURPOSE Resets the dot at the specified graphic display coordinates.

EXAMPLE PRESET (X,Y)
PRESET STEP (10,10),1

PRINT

FORMAT PRINT [<list of expression>]

PURPOSE Outputs data to the LCD screen.

EXAMPLE PRINT "Name is ";A$
PRINT X,Y

PRINT USING

FORMAT PRINT USING "<format string>", <list of expressions>

PURPOSE Outputs data to the LCD screen in the format specified in <"format string">.

EXAMPLE
```
PRINT USING "\          \";A$;B$;C$
PRINT USING "####.##";A;B;C
```

PRINT

FORMAT PRINT # <file number>,
[<list of expressions>]

PURPOSE Writes data to a sequential file.

EXAMPLE
```
PRINT #1,A$;";";B$
```

PRINT # USING

FORMAT PRINT # <file number>,USING<format string>;
<list of expression>

PURPOSE Writes data to a sequential file in a specific format.

EXAMPLE
```
PRINT #1 USING "#####.##";A;B;C
```

PSET

FORMAT PSET [STEP](X,Y)[,<function code>]

PURPOSE Sets the dot at the specified graphic coordinates.

EXAMPLE
```
PSET (A,B)
PSET STEP (5,-5),4
```

PUT

FORMAT PUT [#]<file number>[,<record number>]

PURPOSE Writes a data record to a random access file.

EXAMPLE
```
PUT #1,X
```

RANDOMIZE

FORMAT RANDOMIZE [<expression>]

PURPOSE Reinitializes the random number generator.

EXAMPLE
```
RANDOMIZE
RANDOMIZE VAL (RIGHT$ (TIME$,2))
```

READ

FORMAT READ <list of variables>

PURPOSE Reads values from DATA statements and substitutes them into variables.

EXAMPLE READ A$,B$,C$

REM

FORMAT REM <remark>
' <remark>

PURPOSE Used to insert explanatory remarks into a program.

EXAMPLE ' REGRESSION ROUTINE

REMOVE

FORMAT REMOVE

PURPOSE Writes the microcassette directory to tape and disables read and write access to the microcassette drive.

EXAMPLE REMOVE

RENUM

FORMAT RENUM [[<new line number>][,[<old line number>][,<increment>]]]

PURPOSE Renumbers the lines of programs.

EXAMPLE RENUM
RENUM 300,50
RENUM 1000,900,20

RESET

FORMAT RESET

PURPOSE Resets the READ ONLY condition after a floppy disk in a disk drive has been replaced.

EXAMPLE RESET

RESTORE

FORMAT RESTORE [<line number>]

PURPOSE Resets the pointer which keeps track of the last item read from DATA statements.

EXAMPLE RESTORE
RESTORE 1000

RESUME

FORMAT RESUME
RESUME Ø
RESUME NEXT
RESUME <line number>

PURPOSE Used to continue program execution after completion of an error processing routine.

EXAMPLE RESUME 1ØØ

RIGHT$

FORMAT RIGHT$ (X$,J)

PURPOSE Returns J characters from the right end of string X$.

EXAMPLE A$ = RIGHT$ ("abcdefg",3)

RND

FORMAT RND[(X)]

PURPOSE Returns a random number between 0 and 1.

EXAMPLE A = RND

RUN

FORMAT RUN [<line number>]
RUN <file descriptor>[,R]

PURPOSE Starts execution of a program.

EXAMPLE RUN 3ØØ
RUN "B:SAMPLE",R

SAVE

FORMAT SAVE <file descriptor>[,A]
SAVE <file descriptor>[,P]

PURPOSE Saves the program in memory to a disk file or the RS-232C interface.

EXAMPLE SAVE"ADDRESS.DAT"
SAVE"COMØ:"

SCREEN

FORMAT SCREEN [<mode>]
[,[<virtual screen>]
[,[<function key switch>]
[,[<boundary character>]
[WIDTH [<no. columns>]
[,[<no. lines 1>][,<no. lines 2>]]]]]]

PURPOSE Sets the screen mode and screen parameters.

EXAMPLE SCREEN 2,1,Ø,"E" WIDTH 2Ø,2Ø,2Ø

SCREEN

FORMAT SCREEN (<horizontal position>,<vertical position>)

PURPOSE Returns the ASCII code corresponding to the character at the specified screen location.

EXAMPLE A = SCREEN(5,5)

SGN

FORMAT SGN (X)

PURPOSE Returns the sign of X.

EXAMPLE A = SGN(X)

SIN

FORMAT SIN(X)

PURPOSE Returns the sine of X.

EXAMPLE A = SIN(X)

SOUND

FORMAT SOUND <pitch>,<duration>

PURPOSE Outputs a tone of the specified pitch and duration from the speaker.

EXAMPLE SOUND 1ØØØ,1ØØ

SPACE$

FORMAT SPACE$(J)

PURPOSE Returns a string of spaces of a specified length.

EXAMPLE A$ = "AAA" + SPACE$(10) + "CCC"

SPC

FORMAT SPC(J)

PURPOSE Outputs a string of J spaces to the display or printer.

EXAMPLE PRINT SPC(10);A$

SQR

FORMAT SQR(X)

PURPOSE Returns the square root of X.

EXAMPLE PRINT SQR(2#)

STAT

FORMAT STAT [<program area no.>]
STAT ALL

PURPOSE Displays the status of BASIC program areas.

EXAMPLE STAT
STAT 1
STAT ALL

STOP

FORMAT STOP

PURPOSE Terminates program execution and returns BASIC to the command level.

EXAMPLE STOP

STOP KEY

FORMAT STOP KEY | ON |
OFF |

PURPOSE Disables or reenables the [STOP] key.

EXAMPLE STOP KEY ON
STOP KEY OFF

STR$

FORMAT STR$(X)

PURPOSE Returns a string representation of the value of X.

EXAMPLE A$ = STR$(123)

STRING$

FORMAT STRING$(J,K)
STRING$(J,X$)

PURPOSE Returns a string of J characters.

EXAMPLE PRINT STRING$(10,65)
PRINT STRING$(10,"A")

SWAP

FORMAT SWAP <variable 1>,<variable 2>

PURPOSE Exchanges the values of the variables specified in <variable 1> and <variable 2>

EXAMPLE SWAP A$,B$

SYSTEM

FORMAT SYSTEM

PURPOSE Terminates BASIC operation and returns control to the CP/M operating system.

EXAMPLE SYSTEM

TAB

FORMAT TAB(J)

PURPOSE Moves the cursor (print head) to character position J on the display screen (print line).

EXAMPLE PRINT TAB(10);"ABC"
LPRINT TAB(10);"ABC"

TAN

FORMAT TAN(X)

PURPOSE Returns the tangent of X.

EXAMPLE A = TAN(3.1416/4)

TAPCNT

FORMAT As a statement
TAPCNT = J
As a variable
J = TAPCNT

PURPOSE Sets or reads the PX-8's microcassette drive counter.

EXAMPLE TAPCNT = 0
PRINT TAPCNT

TIME$

FORMAT As a statement
TIME$ = "<HH>:<MM>:<SS>"
As a variable
X$ = TIME$

PURPOSE Sets or reads the PX-8's built-in clock.

EXAMPLE TIME$ = "15:35:ØØ"
PRINT TIME$

TITLE

FORMAT TITLE[<program area name>][,P]

PURPOSE Sets the name and edit attribute of the currently selected BASIC program area.

EXAMPLE TITLE"FINAL",P
TITLE""

TRON/TROFF

FORMAT TRON
TROFF

PURPOSE Starts or stops the trace mode of program execution.

EXAMPLE TRON
TROFF

USR

FORMAT USR[<digit>](argument)

PURPOSE Calls a machine language subroutine defined by a DEF USR statement.

EXAMPLE A = USRØ(B)

VAL

FORMAT VAL(X$)

PURPOSE Converts a string composed of numeric ASCII characters into a numeric value.

EXAMPLE A = VAL("123")

VARPTR

FORMAT VARPTR(<variable name>)
VARPTR(# <file number>)

PURPOSE Returns the address of the specified variable or file buffer.

EXAMPLE PRINT HEX$(VARPTR(A))
PRINT HEX$(VARPTR(1))

WAIT

FORMAT WAIT <port number>,J[,K]

PURPOSE Suspends program execution until a specified bit pattern is developed at the specified machine input port.

WHILE...WEND

FORMAT
```
WHILE <expression>
    :
[<loop statements>]
    :
WEND
```

PURPOSE Repeats the series of instructions included between WHILE and WEND as long as the result of the specified expression is TRUE.

EXAMPLE
```
WHILE X≦1ØØ
    :
WEND
```

WIDTH

FORMAT
```
WIDTH [<no. of columns>][,[<no. of lines 1>][,<no. of lines 2>]]
WIDTH <file descriptor>,<no. of columns>
WIDTH LPRINT <no. of columns>
```

PURPOSE Sets the column width of the virtual screens or other specified device or file.

EXAMPLE
```
WIDTH 2Ø,2Ø,2Ø
WIDTH "LPTØ:",8Ø
WIDTH #1,8Ø
WIDTH LPRINT 4Ø
```

WIND

FORMAT WIND[<counter value> | ON | OFF]

PURPOSE Turns the microcassette drive motor on or off, winds the tape until the specified counter value is reached, or rewinds the tape to the beginning and resets the counter to 0.

EXAMPLE
```
WIND
WIND ON
WIND OFF
WIND 3ØØØ
```

WRITE

FORMAT WRITE[<list of expressions>]

PURPOSE Displays data on the LCD screen.

EXAMPLE WRITE A$,B$,C$

WRITE

FORMAT WRITE # <file number>,<list of expressions>

PURPOSE Writes data to a sequential disk file using the format of the WRITE statement.

EXAMPLE WRITE # 1,A$,B$

Error Codes and Error Messages

Error Message	Code	Description
Bad file mode	54	A statement or function was used with a file of the wrong type.
Bad file descriptor	64	An illegal file name was specified in a LOAD, SAVE, or KILL command or an OPEN statement (for example, a file name with too many characters).
Bad file number	52	A statement or command references a file that has not been opened, or the file number specified in an OPEN statement was outside of the range of file numbers that was specified when BASIC was started.
Bad record number	63	The record number specified in a PUT or GET statement was either zero or greater than the maximum allowed.
Can't continue	17	An attempt was made to resume execution of a program when continuation was not possible.
Communication buffer overflow	28	The receive buffer overflowed during receipt of data via the RS-232C interface.
Device fault	25	The level of the signal on the DSR or DCD line became low during input from the RS-232C interface after the DSR receive check or DCD check had been set to ON (by option ''c'' of the communications format specification in the OPEN''I'' sttement executed to open the interface).

Error Message	Code	Description
Device I/O error	57	An error occurred involving input or output to a peripheral device.
Device time out	24	An input or output device was not ready when an input or output operation was attempted.
Device unavailable	68	An attempt was made to access a drive which did not contain a floppy disk; or, the RS-232C interface was not available.
Direct statement in file	66	A program line without a line number was encountered during execution of a LOAD or MERGE command; or, an attempt was made to LOAD a data file or machine language program.
Disk full	61	All storage on the disk is in use.
Disk read error	70	A read error occurred while data was being read from a disk device.
Disk write error	71	A write error occurred while data was being written to a disk device.
Disk write protect	69	An attempt was made to write data to a disk which was protected by a write protect tab; an attempt was made to write data to a disk drive without executing the RESET command after replacing the disk in that drive; or, an attempt was made to write data to a ROM capsule.
Division by zero	11	An operation was encountered which included division by zero.
Duplicate Definition	10	A variable array was defined more than once.

Error Message	Code	Description
FIELD overflow	50	A FIELD statement attempted to allocate more bytes in a random file buffer than were specified for that buffer when the file was opened.
File already exists	58	The new file name specified in a NAME statement is already being used with another file on the disk.
File already open	55	An OPEN''O'' statement was executed for a file which was already open, or a KILL command was executed for a file that was open.
File not found	53	The file name specified in a LOAD, KILL, NAME, or OPEN statement did not exist on the disk in the accessed drive.
FOR without NEXT	26	A FOR statement was encountered without a corresponding NEXT.
Illegal direct	12	A statement that is illegal in the direct mode (such as DEF FN) was entered as a direct mode command.
Illegal function call	5	A statement or function was incorrectly specified.
Input past end	62	An INPUT statement was executed for an empty file or after all data in the file had been read (to avoid this error, use the EOF function to detect the end of the file.) Or, the STOP key was pressed while input from an RS-232C interface was pending with INPUT# INPUT$, or the like.
Internal error	51	An internal malfunction occurred in BASIC.
Line buffer overflow	23	An attempt was made to input a line that contains too many characters.

Error Message	Code	Description
Missing operand	22	An expression contains an operator without a following operand; a required parameter is missing from the AUTO START or LOCATE commands.
NEXT without FOR	1	A NEXT statement was encountered without a corresponding FOR statement.
No RESUME	19	No RESUME statement was included in an error processing routine.
Out of DATA	4	A READ statement was executed when there was no unread data remaining in the program's DATA statements.
Out of memory	7	Memory available is insufficient for processing required.
Out of string space	14	Insufficient memory space is available for storage of characters in string variables.
Overflow	6	A numeric value was encountered whose magnitude exceeds the limits prescribed by PX-8 BASIC (if underflow occurs, zero is assumed and execution continues without error).
RESUME without error	20	A RESUME statement was encountered outside of an error processing routine.
RETURN without GOSUB	3	A RETURN statement was encountered which did not correspond to a previously executed GOSUB statement.
String formula	16	The complexity of a string operation is too great.
String too long	15	An attempt was made to create a string whose length exceeds 255 characters.
Subscript out of range	9	The subscript specified in a statement referencing an array element is outside the range permitted for that array.
Syntax error	2	A statement does not conform to the syntax rules of PX-8 BASIC.

Error Message	Code	Description
Tape access error	72	An access error occurred during an attempt to access a microcassette file.
Too many files	67	An attempt was made to create a new disk file after all directory entries were full.
Type mismatch	13	A string expression was used where a numeric expression is required, or vice versa.
Undefined line number	8	A non-existent line number was specified in one of the following commands or statements: EDIT, GOTO, GOSUB, RESTORE, RUN, RENUM
Undefined user function	18	A call was made to an undefined user function.
Unprintable error	21	No error message has been assigned to the error condition which exists; this message also applies to error codes 27, 31-49, 56, 59, 60, 65, and 73-255.
WEND without WHILE	30	A WEND statement was encountered without a corresponding WHILE.
WHILE without WEND	29	A WHILE statement was encountered without a corresponding WEND.

Console Escape Sequences

Control Code	Function
ESC "%"	Access CGROM directly
ESC 243	Arrow key code
ESC 246	Buffer clear key
ESC "C"	Character table
ESC 246	Clear keyboard buffer
ESC "*"	Clear screen
ESC 245	CTRL key code
ESC 215	Cursor find
ESC 243	Cursor key code
ESC "="	Cursor position set
ESC 214	Cursor type select
ESC "P"	Dump screen
ESC "T"	Erase to end of line
ESC "Y"	Erase to end of screen
ESC 210	Display characters on real screen
ESC 208	Display mode set
ESC 198	Dot line write
ESC 213	End locate
ESC 215	Find cursor
ESC 177	Function key code returned
ESC 176	Function key string returned
ESC 211	Function key display select
ESC "C"	International Character Sets
ESC 161	INS LED off
ESC 160	INS LED on
ESC 242	Key repeat interval time
ESC 240	Key repeat on/off
ESC 241	Key repeat start time
ESC 244	Key code scroll
ESC 247	Key shift set
ESC "T"	Line erase
ESC 198	Line dot draw
ESC 213	Locate end of screen
ESC 212	Locate top of screen

Control Code	Function
ESC 125	Non secret
ESC 165	NUM LED off
ESC 164	NUM LED on
ESC 199	PSET/PRESET
ESC 242	Repeat interval time for keys
ESC 240	Repeat on/off for keys
ESC 241	Repeat start time for keys
ESC "*"	Screen clear
ESC 209	Screen display select
ESC "P"	Screen dump
ESC 213	Screen window end
ESC "Y"	Screen erase
ESC 212	Screen window top
ESC 145	Scroll down
ESC 244	Scroll key code
ESC 148	Scroll step
ESC 149	Scroll mode
ESC 144	Scroll up
ESC 151	Screen down n lines
ESC 150	Scroll up n lines
ESC 123	Secret mode
ESC 125	Secret mode cancel
ESC 214	Select cursor type
ESC 209	Select virtual screen
ESC 211	Select function key display
ESC 247	Shift key set
ESC 163	CAPS LED off
ESC 162	CAPS LED on
ESC 212	Top locate
ESC 224	User defined character

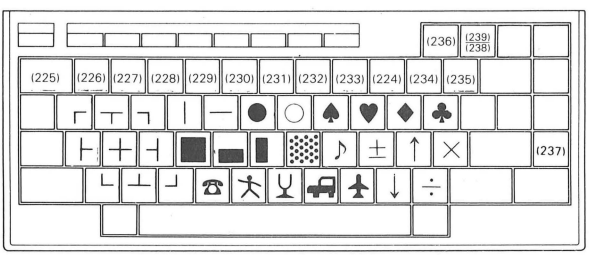
Device Assignments

Device name	Physical device	I/O	Note
A	RAM disk	I/O	
B	ROM capsule	I	
C			
D	Floppy disk device	I/O	
E			
F			
G			
H	Microcassette drive	I/O	

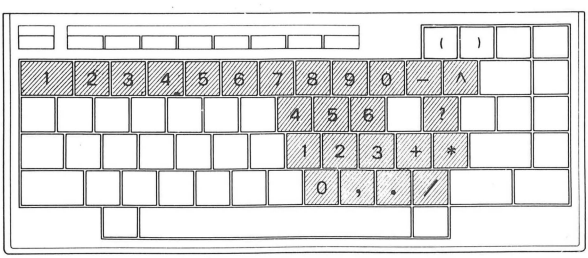
Note:
The assignments shown above can be changed by the CONFIG command.

Keyboard Layout

Graphic mode



Numeric mode



ASCII Code Table

Hex. No.	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Hex. No.	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	0000	0	16	SP 32	48	@ 64	P 80	` 96	p 112	128	144	160	176	192	208	224	240
1	0001	1	17	! 33	1 49	A 65	Q 81	a 97	q 113	129	145	161	177	193	209	225	241
2	0010	2	18	" 34	2 50	B 66	R 82	b 98	r 114	130	146	162	178	194	210	226	242
3	0011	3	19	# 35	3 51	C 67	S 83	c 99	s 115	131	147	163	179	195	211	227	243
4	0100	4	20	$ 36	4 52	D 68	T 84	d 100	t 116	132	148	164	180	196	212	228	244
5	0101	5	21	% 37	5 53	E 69	U 85	e 101	u 117	133	149	165	181	197	213	229	245
6	0110	6	22	& 38	6 54	F 70	V 86	f 102	v 118	134	150	166	182	198	214	230	246
7	0111	7	23	' 39	7 55	G 71	W 87	g 103	w 119	135	151	167	183	199	215	231	247
8	0100	8	24	(40	8 56	H 72	X 88	h 104	x 120	136	152	168	184	200	216	232	248
9	1001	9	25	) 41	9 57	I 73	Y 89	i 105	y 121	137	153	169	185	201	217	232	249
A	1010	10	26	* 42	: 58	J 74	Z 90	j 106	z 122	138	154	170	186	202	218	234	250
B	1011	11	27	+ 43	; 59	K 75	[91	k 107	{ 123	139	155	171	187	203	219	235	251
C	1100	12	28	, 44	< 60	L 76	\ 92	l 108	| 124	140	156	172	188	204	220	236	252
D	1101	13	29	- 45	= 61	M 77	] 93	m 109	} 125	141	157	173	189	205	221	237	253
E	1110	14	30	. 46	> 62	N 78	^ 94	n 110	~ 126	142	158	174	190	206	222	238	254
F	1111	15	31	/ 47	? 63	O 79	_ 95	o 111	127	143	159	175	191	207	223	239	255

NOTES:

1. *$(0)_D$ through $(31)_D$ are control characters.*
2. *$(32)_D$ through $(127)_D$ are ASCII characters.*
3. *Characters displayed for codes E0 to FF can be defined by the user.*
4. *For further details see section 2.6.2, "Control Characters of the BASIC reference manual" and the User's Manual.*

Differences between the USASCII character set and the character sets of other countries are as shown below.

Country / Dec. Code	United States	France	Germany	England	Denmark	Sweden	Italy	Spain	Norway
35	#	#	#	£	#	#	#	₧	#
36	$	$	$	$	$	¤	$	$	¤
64	@	à	§	@	É	É	@	@	É
91	[	°	Ä	[	Æ	Ä	°	¡	Æ
92	\	Ç	Ö	\	Ø	Ö	\	Ñ	Ø
93	]	§	Ü	]	Å	Å	é	¿	Å
94	^	^	^	^	Ü	Ü	^	^	Ü
96	`	`	`	`	é	é	ù	`	é
123	{	é	ä	{	æ	ä	à	¨	æ
124	|	ù	ö	|	ø	ö	ò	ñ	ø
125	}	è	ü	}	å	å	è	}	å
126	~	¨	ß	~	ü	ü	ì	~	ü

EPSON OVERSEAS MARKETING LOCATIONS

EPSON AMERICA, INC.
2780 Lomita Blvd.
Torrance, CA 90505 U.S.A.
Phone: (213) 539-9174
Telex: 182412

EPSON UK LTD.
Dorland House
388 High Road,
Wembley, Middlesex, HA9 6UH, U.K.
Phone: (01) 902-8892
Telex: 8814169

EPSON DEUTSCHLAND GmbH
Am Seestern 24
4000 Düsseldorf 11
F.R. Germany
Phone: (0211) 5952-0
Telex: 8584786

EPSON ELECTRONICS (SINGAPORE) PTE. LTD.
No.1 Maritime Square, #02-19
World Trade Centre
Singapore 0409
Phone: 2786071/2
Telex: 39536

EPSON ELECTRONICS TRADING LTD.
Room 411, Tsimshatsui Centre,
East Wing, 66, Mody Road
Tsimshatsui Kowloon, Hong Kong
Phone: 3-694343/4
3-7213427
3-7214331/3
Telex: 34714

EPSON ELECTRONICS TRADING LTD. TAIWAN BRANCH
1,8F K.Y. Wealthy Bldg. 206, Nanking
E. Road, Sec, 2, Taipei, Taiwan, R.O.C.
Phone: 536-4339
536-3567
Telex: 24444

EPSON FRANCE S. A.
114, Rue Marius Aufan
92300 Levallois-perret
France
Phone: (1) 758-77-00
Telex: 614169

EPSON AUSTRALIA PTY. LTD.
Unit 3, 17 Rodborough Road
Frenchs Forest, NSW 2086
Australia
Phone: (02) 452-5222
Telex: (71) 75052

EPSON

Printed in Japan 84.08-10