Index wersja polska

Casio PB-700 - data formats

BASIC tokens

$80	SIN	$90	ASC(	$A0	GOTO	$B0	ELSE	$C0	CLEAR	$D0	SYSTEM
$81	COS	$91	LEN(	$A1	GOSUB	$B1	STEP	$C1	PROG	$D1	CLS
$82	TAN	$92	VAL(	$A2	RETURN	$B2	THEN	$C2	PUT	$D2	
$83	EXP	$93	PI	$A3	FOR	$B3	TO	$C3	GET	$D3	
$84	ASN	$94	RND	$A4	NEXT	$B4	USING	$C4	VERIFY	$D4	
$85	ACS	$95		$A5	IF	$B5	TAB(	$C5	CHAIN	$D5	
$86	ATN	$96		$A6	STOP	$B6	ALL	$C6	SAVE	$D6	
$87	LOG	$97		$A7	INPUT	$B7	DATA	$C7	LOAD	$D7	
$88	LGT	$98		$A8	READ	$B8	REM	$C8	PASS	$D8	
$89	SQR	$99	INKEY$	$A9	RESTORE	$B9	LET	$C9	NEW	$D9	
$8A	ABS	$9A	CHR$(	$AA	END	$BA	ANGLE	$CA	LIST	$DA	
$8B	FRAC	$9B	STR$(	$AB	DRAW(	$BB	BEEP	$CB	LLIST	$DB	
$8C	INT	$9C	LEFT$(	$AC	DRAWC(	$BC	DIM	$CC	RUN	$DC	
$8D	SGN	$9D	RIGHT$(	$AD	LOCATE	$BD	ERASE	$CD	DELETE	$DD	
$8E	POINT(	$9E	MID$(	$AE	PRINT	$BE	TRON	$CE	EDIT	$DE	
$8F	ROUND(	$9F	MOD	$AF	LPRINT	$BF	TROFF	$CF	CONT	$DF	

There aren't any undocumented keywords.

BASIC program structure

BASIC line begins with a line number stored in 2 bytes (four 4-bit words) in packed decimal format, ends with an end marker $FF. BASIC keywords are stored as single byte tokens, numeric values as strings of characters, colons used as statements separators as $FE.
Example:

```
1234 FOR I=1 TO 49 STEP 1: NEXT I
34 12 A3 49 3D 31 B3 34 39 B1 31 FE A4 49 FF
```

File formats

A file consists of a header segment followed by one or more data segments.

Structure of the file header segment

```
 1 byte         character 'H' - the header segment identifier
 1 byte         file type
 8 bytes        file name, padded with spaces to 8 characters
 3 bytes        3 spaces, reserved for the file name extension?
 8 bytes        password, data are inverted (xor FF) and padded with FFs
12 bytes        parameters - information specific to individual file type,
                seem to be ignored when loading
 1 byte         checksum - the two's complement of the sum of all preceding
                bytes modulo 256 (sum of all bytes + checksum = 0x00),
                ignored, not even read from the tape when loading
 1 byte        end marker 0xF1, ignored, not even read from the tape
 1 byte        fixed value 0x00, ignored, not even read from the tape
```

Example:

```
    48                                      character 'H'
    D0                                      file type: PROGRAM
    50 41 53 53 20 20 20 20                 file name: PASS
    20 20 20                                file name extension
    BD BA AB BE FF FF FF FF                 password: BETA
    00 00 00 00 00 00 04 00 00 00 00 00     parameters
    F1                                      checksum
    F1                                      end marker
    00                                      byte 0x00
```

PROGRAM files

Header of the PROGRAM file

```
    file type       0xD0
    parameters+6    the least significant byte of the program length
    parameters+7    the most significant byte of the program length
```

Structure of the PROGRAM data segment

```
     1 byte        character 'D' - the data segment identifier
  xxxx bytes       the BASIC program
     1 byte        end marker 0xF0
```

Example:

```
    44
    10 00 A3 49 3D 31 B3 34 FE A4 49 FF     10 FOR I=1 TO 4: NEXT I
    20 00 A0 31 30 FF                       20 GOTO 10
    F0
```

ALL PROGRAMS files (saved with SAVE ALL)

Header of ALL PROGRAMS file

```
    file type       0xC1
```

Structure of the ALL PROGRAMS data segment

```
     1 byte        character 'D' - the data segment identifier
  xxxx bytes       list of 10 BASIC programs, each program terminated with 0xE0,
                   an empty program is stored as 0xE0 alone
     1 byte        end marker 0xF0
```

Example:

```
44
00 10 AE FF E0                          P0: 1000 PRINT
10 00 A2 FF E0                          P1: 10 RETURN
E0 E0 E0 E0 E0 E0 E0 E0
F0
```

ASCII PROGRAM files (saved with SAVE,A)

Each BASIC line is stored in a separate data segment.

Header of the ASCII PROGRAM file

```
file type       0x30
```

Structure of the ASCII PROGRAM data segment

```
   1 byte         character 'D' - the data segment identifier
   1 byte         file type = 0x30
   1 byte         fixed value = 0x00
   2 bytes        length of the program line, the least significant byte
                  first, seems to be ignored when loading
xxxx bytes        the BASIC line in the form of a string of ASCII characters
                  terminated by a CR character (code 0x0D)
   1 byte         checksum - the two's complement of the sum of all preceding
                  bytes modulo 256 (sum of all bytes + checksum = 00),
                  ignored, not even read from the tape when loading
   1 byte         end marker 0xF1, ignored, not even read from the tape
   1 byte         fixed value 0x00, ignored, not even read from the tape
```

Example:

```
280 A$=INKEY$
44 30 00 0F 00 20 32 38 30 20 41 24 3D 49 4E 4B 45 59 24 0D 50 F1 00
```

Structure of the last ASCII PROGRAM data segment

```
   1 byte         character 'D' - the data segment identifier
   1 byte         file type = 0x30
   4 bytes        fixed values 0xFF, 0x01, 0x00, 0x0D
   1 byte         checksum - the two's complement of the sum of all preceding
                  bytes modulo 256 (sum of all bytes + checksum = 00),
                  ignored, not even read from the tape when loading
   1 byte         end marker 0xF1, ignored, not even read from the tape
   1 byte         fixed value 0x00, ignored, not even read from the tape
   1 byte         end marker 0xF0, ignored, not even read from the tape
```

Example:

```
44 30 FF 01 00 0D 7F F1 00 F0
```

VARIABLES files (saved with PUT)

Each variable is stored in a separate data segment. No variable names are stored.

Header of the VARIABLES file

```
    file type       0x24
```

Structure of the VARIABLES data segment

```
     1 byte         character 'D' - the data segment identifier
     1 byte         file type = 0x24
     1 byte         fixed value = 0x00
     2 bytes        length of the variable, the least significant byte first,
                    seems to be ignored when loading
  xxxx bytes        the contents of the variable in the form of a string of ASCII
                    characters terminated by a CR character (code 0x0D),
                    a numerical variable begins with a space (code 0x20)
     1 byte         checksum - the two's complement of the sum of all preceding
                    bytes modulo 256 (sum of all bytes + checksum = 00),
                    ignored, not even read from the tape when loading
     1 byte         end marker 0xF1, ignored, not even read from the tape
     1 byte         fixed value 0x00, ignored, not even read from the tape
```

Examples:

```
    a string variable containing "PIOTR":
    44 24 00 06 00 50 49 4F 54 52 0D F7 F1 00

    a numerical variable containing 3.141592654:
    44 24 00 0D 00 20 33 2E 31 34 31 35 39 32 36 35 34 0D 28 F1 00
```

Structure of the last VARIABLES data segment

```
     1 byte         character 'D' - the data segment identifier
     1 byte         file type = 0x24
     4 bytes        fixed values 0xFF, 0x01, 0x00, 0x0D
     1 byte         checksum - the two's complement of the sum of all preceding
                    bytes modulo 256 (sum of all bytes + checksum = 00),
                    ignored, not even read from the tape when loading
     1 byte         end marker 0xF1, ignored, not even read from the tape
     1 byte         fixed value 0x00, ignored, not even read from the tape
     1 byte         end marker 0xF0, ignored, not even read from the tape
```

Example:

```
    44 24 FF 01 00 0D 8B F1 00 F0
```
