PPC ROM POCKET GUIDE
KEYBOARD FUNCTIONS (Default Assignments)

This Pocket Guide is the short form of the 500-page ROM Users Manual. This, the HEX Card and the PPC ROM are available at PPC.

PPC, the Personal Programming Center, publishes a monthly 48-page joint computer/calculator journal. PPC is a California non-profit public benefit corporation dedicated to the advancement and application art of personal programming.

PPC is the oldest, most experienced worldwide users group of personal programmers having printed more than 15 million pages on personal programming calculators and computers.
Each routine has an abbreviated Technical Details box. Abbreviations used in this box are:

- **S**: SIZE
- **R**: Registers Used
- **F**: Flags Used
- **SL**: Subroutines Left
- **C**: Registers to Copy
- **A**: Alpha Registers
- **m**: Minimum
- **A/R**: As Required
- **N/A**: Not Applicable
- **(R)**: All Flags Restored

*Maximum pending RTN’s when routine is called.

---

**+K** — ADDITIONAL **KEY ASSIGNMENT**

May be used after **MK** (which will set flag 20 and clear 07) or after **1K** (which will set both flags 20 and 07). The sequence prefix ENTER postfix ENTER keycode, XEQ **+K** will make the corresponding key assignment. If flag 07 is clear, it will then prompt for subsequent key assignments; otherwise it returns after making only one assignment. Both **1K** and **+K** permit key assignments under program control. Note that the sequence CF 20, SF07, XEQ **+K** is completely equivalent to XEQ **1K**. Do not use data registers 09-11 between uses of **+K**. Not interruptible. Disrupts I/O Buffer (Timer Alarms), see warning under **LF**.

```
| S: 012 | R: 6-11, A | F: 7-10, 20, 25 | SL: 3 | C: 61 |
```

---

**-B** — STORE PART OF LB

**L** must be used for initialization of the byte-loading process. With a decimal code of the byte to be loaded in X, XEQ **-B**. Will load the corresponding byte into the next available location after LBL “+ +”. CF 09 to terminate byte loading with prompting for clean up operations. CF 08 to switch to manual byte loading.

```
| S: 012 | R: 6-11, A | F: 8, 9, 29 | SL: 4 | C: 71 |
```

---

**1K** — FIRST **KEY ASSIGNMENT**

A non-prompting key assignment program. The sequence prefix ENTER postfix ENTER keycode, XEQ **1K**, will make one key assignment. For more assignments, see **+K**. Not interruptible. Disrupts I/O Buffer (Timer Alarms), see warning under **LF**.

```
| S: 012 | R: 6-11, A | F: 7-10, 20, 25 | SL: 3 | C: 61 |
```
**2D** — DECODE 2 BYTES TO DECIMAL  XROM 10,55
Evaluates the decimal equivalents for the last two bytes in X. Returns the decimal equivalent for the next-to-last byte in X, for the last byte in M. (Use X < > M to view.) Y, Z preserved.

```
| S: 000 | R: A | F: None | SL: 6 | C: 60 |
```

**A?** — ASSIGNMENT REGISTER FINDER ?  XROM 10,10
Returns the number of assignment registers used. Key assignments by ASN may give a register count 1/2 too high. If timer alarms are present they will be disrupted and file count will be inaccurate. See alarm warning under **LF**.

```
| S: 000 | R: A | F: 10 | SL: 3 | C: 59 |
```

**AD** — ALPHA DELETE LAST CHARACTER  XROM 10,18
Removes the last (rightmost) character from the alpha register. There must be no nulls in the string. L is preserved.

```
| S: 000 | R: A | F: None | SL: 6 | C: 64 |
```

**AL** — ALPHABETIZE X AND Y  XROM 10,37
Sorts two alpha strings in X and Y in alphabetic order (upon return lower value in X). If X and Y are numeric, they are regarded as pointers to two data registers which are to be sorted, the lower value to the register designated by X. Flag 10 is left set or clear to indicate whether an interchange was performed or not.

```
| S: 000 | R: A | F: 10,25 | SL: 4 | C: 63 |
```

**AM** — ALPHA TO MEMORY  XROM 20,53
ASTOres then clears all or part of contents of the ALPHA register into data registers using an ISG control number in X of the form bbb.eeeii. Inverse operation is **MA**. (not synthetic)

```
| S: 004m | R: A/R A | F: None | SL: 5 | C: 16 |
```
Ab — ALPHA STORE b

Provides an ASTO b with the ROM-mode interpretation, permitting ultra-fast ROM entry as an alternative to XE.

In ROM the mode of interpreting the last two bytes of status register b (which comprise the program pointer) is different than when in RAM. In ROM the first nibble specifies a 4k page of ROM, and the last 3 nibbles specify an address within that page. In RAM the first nibble (mod 8) specifies a byte within a register specified by the last 3 nibbles (mod 1024).

Stack is preserved.

| S: 000 | R: None | F: None | SL: 0 | C: 60 |

Ba — BARCODE ANALYZER

Prompts for wand scan of barcode. Prints tabular analysis of barcode including type, with every byte in binary, hex, decimal, and the printer character (< 128). Computes and prints checksum if > 2 bytes.

| S: 019 | R: 0-18, A | F: 7-10, 12, 21 | SL 4 | C: 49 |

BC — BLOCK CLEAR

Stores 0's in defined block using an ISG control number bbb.eeeeii in X. Y, Z, and T are preserved.

| S: A/R | R: A/R | F: None | SL: 5 | C: 61 |

BD — BASE b TO BASE DECIMAL

Store base b in RO6 (2 = b = 25). Input up to 14 digits of base b number in alpha (each digit must be less than b) and XEQ BD. The base 10 result is left in X and alpha is cleared. The inverse routine is TB. Keyboard function A will XEQ BD after positioning to BD, TB, PM, CM, CJ, or JC.

| S: 007 | R: 6, A | F: None | SL: 5 | C: 53 |

BE — BLOCK EXCHANGE

Input two ISG block control numbers bbb.eeeeii in X and Y and XEQ BE. For two blocks not of the same size, note that the
block control number in Y is tested and controls the loop. Z, L are preserved.

| S: A/R | R: A/R | F: None | SL: 5 | C: 61 |

**BI** — **BLOCK INCREMENT**

Stores numerical data (as a sequence, constant, or zero) in a block of registers. Input:

- Z: Control word (bbb.eeeii) bbb.eeeii ENTER
- Y: Starting Data Value Start value ENTER
- X: Data Increment Value Increment value XEQ BI.

X, T are preserved in L, T.

| S: A/R | R: None | F: None | SL: 5 | C: 46 |

**BL** — **BLDSPEC INPUTS FOR LB**

Converts seven BLDSPEC numbers into seven decimal byte numbers for a synthetic text line. Key first BLDSPEC number, XEQ BL. See corresponding byte in X (0-255). Key second BLDSPEC number, R/S, see second byte. Continue with R/S until all seven bytes are obtained. To use LB start with 247, followed by seven bytes. To print: Text line, RCL M, ACSPEC, PRBUF. Related routines are FL and XL. These are not intended to be used as subroutines.

| S: 000 | R: M, N, O | F: None | SL: N/A | C: 46 |

**BM** — **BLOCK MOVE**

Moves a block of consecutive registers into another block of consecutive registers. Input:

- Z: 1st register number in source block
- Y: 1st register number in destination block
- X: Number of registers in block

| S: A/R | R: None | F: None | SL: 5 | C: 61 |

**BR** — **BLOCK ROTATE**

Rotates the content of data in a block of consecutive registers.
Input:

Y: 1st register number of block
X: ± number of registers in block

If X > 0 the block shifts to higher numbered registers
If X < 0 the block shifts to lower numbered registers
Each call to BR shifts by one register.

<table>
<thead>
<tr>
<th>S: A/R</th>
<th>R: A/R</th>
<th>F: None</th>
<th>SL: 5</th>
<th>C: 61</th>
</tr>
</thead>
</table>

**BV — BLOCK VIEW**

Allows viewing (or printing if able) of non-zero register contents of a block defined using a control word bbb.eeeii in X. Set flag 09 to PSE at each register. Set flag 10 to stop at each register.

<table>
<thead>
<tr>
<th>S: A/R</th>
<th>R: A/R</th>
<th>F: 9, 10, 21, 25, 29</th>
<th>SL: 4</th>
<th>C: 40</th>
</tr>
</thead>
</table>

**BX — BLOCK EXTREMA**

Finds the maximum or minimum of a block of registers defined using a control word bbb.eeeii in X. Set flag 10 to use absolute values. Output:

M: INT part is max value register number
N: INT part is min value register number
O: Original block control word
Y: Max value of block
X: Min value of block

<table>
<thead>
<tr>
<th>S: A/R</th>
<th>R: M,N,O</th>
<th>F: 10</th>
<th>SL: 5</th>
<th>C: 61</th>
</tr>
</thead>
</table>

**BS — BLOCK STATISTICS**

Calculates statistical sums for two register blocks defined by control words bbb.eeeii in X and Y. For two blocks not of the same size the block control word in Y controls the loop. Output in Σ-registers:

Σx Σx² Σy Σy² Σxy n=#reg. in block

<table>
<thead>
<tr>
<th>S: A/R</th>
<th>R: ΣREG</th>
<th>F: none</th>
<th>SL: 5</th>
<th>C: 61</th>
</tr>
</thead>
</table>

© PPC 1981

Create this function using LB or MK inputs 162, 128. Use this function to determine if PPC ROM is present. It has no effect if present, gives NON EXISTENT or clears flag 25 if absent.

<table>
<thead>
<tr>
<th>S: 000</th>
<th>R: none</th>
<th>F: 25</th>
<th>SL: 5</th>
<th>C: 61</th>
</tr>
</thead>
</table>

*PPC ROM POCKET GUIDE*
C? — CURTAIN FINDER
Returns the absolute decimal address of R00 (curtain location). X, Y preserved in Y, Z.

| S: 000 | R: A | F: (R) | SL: 6 | C: 64 |

CA — COMPLEX ARITHMETIC
GTO CA and XEQ 06 to initialize the complex stack. The keyboard functions are:

- a: X<>Y
- b: Y^X
- c: Pop
- d: Last Z
- e: e^z

Initialize (XEQ 06)

| A: + | B: − | C: × | D: ÷ | E: ln(z) |
| H: sin(z) | I: cos(z) | J: Push |

Register usage:

- R06: function #
- R07: Last Z (IM)
- R08: Last Z (RE)
- R09: Ptr to Stk

RAD mode must be used.

Store call # in R06 and XEQ CA

| 1: + | 2: − | 3: × | 4: ÷ | 5: ln(z) | 6: Init. Stack |
| 7: cosh(y), sinh(y) | 8: sin(z) | 9: cos(z) | 10: PUSH |
| 16: POP(save Last Z) |

CB — COUNT BYTES
Counts bytes between any two instructions in RAM after placing their program pointers in X and Y. The count includes the first instruction, but not the last. Obtain program pointer with RCL b when positioned at the instruction in RUN mode. Also, the decimal address of the first pointer returns in Y, the last pointer's returns in L. Note that an END adds 3 to the byte count. A program byte count divided by 112 and rounded up to the nearest integer yields the number of tracks required to record the segment on magnetic cards. (RCL b = 144, 124, key)

| S: 000 | R: A | F: None | SL: 4 | C: 60 |
**CD** — **CHARACTER TO DECIMAL**  XROM 10,35

Decodes last byte of alpha register (rightmost character displayed) into a decimal number returned in X. Up to 14 characters (the rightmost 14) will be preserved in the alpha register, **including** the decoded character if flag 10 is set. Otherwise the decoded character is deleted. X, Y, Z preserved in Y, Z, T.

<table>
<thead>
<tr>
<th>S: 000</th>
<th>R: A</th>
<th>F: 10</th>
<th>SL: 6</th>
<th>C: 60</th>
</tr>
</thead>
</table>

**CJ** — **CALENDAR DATE TO JULIAN DAY**  XROM 20,21

Clear flag 10 for Gregorian (modern) calendar. Set flag 10 for Julian calendar. The keyboard function E will XEQ **CJ** after positioning to BD, TB, PM, CM, CJ, or JC. Valid from March 1, 0 AD (1BC). Input date in format:

- Z: year  The JDN is returned in X.
- Y: month  DOW = (JDN + 1) MOD 7  (SUN = 0)
- X: day  The inverse routine is JC.

<table>
<thead>
<tr>
<th>S: 000</th>
<th>R: None</th>
<th>F: 10</th>
<th>SL: 5</th>
<th>C: 53</th>
</tr>
</thead>
</table>

**CK** — **CLEAR KEY ASSIGNMENTS**  XROM 10,06

All USER mode assignments revert to the standard keyboard. Assignment registers become free registers. Global label assignments are dormant until the next program or status card is read in. X is preserved. Clears I/O buffers (Timer Alarms).

<table>
<thead>
<tr>
<th>S: 000</th>
<th>R: A</th>
<th>F: (R)</th>
<th>SL: 4</th>
<th>C: 59</th>
</tr>
</thead>
</table>

**CM** — **COMBINATIONS**  XROM 20,20

To compute C(n,k): n ENTER k XEQ **CM**. The number of combinations of n objects taken k at a time is left in X. The keyboard function D will XEQ **CM** after positioning to BD, TB, PM, CM, CJ, or JC.

<table>
<thead>
<tr>
<th>S: 000</th>
<th>R: None</th>
<th>F: None</th>
<th>SL: 5</th>
<th>C: 53</th>
</tr>
</thead>
</table>

**CP** — **COLUMN PRINT FORMATTING**  XROM 20,27

Aligns numeric values for printing columns of data. A skip index keeps decimal points in place and is calculated as (Max # digits

*PPC ROM POCKET GUIDE*  7
left of dec. pt.) \(-1\) for F29 clear, and (Max # digits . . .) \(-1\) + (No. commas in largest number in the column) for F29 set. Set display mode, status of F29, place skip index in R06, place the number in X and XEQ \(\text{CP}\). The number will then be added to the buffer in the correct column position.

\[ S: 007 \quad R: 6 \quad F: \text{None} \quad SL: 5 \quad C: 29 \]

---

**CU**—**CURTAIN UP**

XROM 10,34

Adds the signed integer value of the contents of X to the R00 pointer (in status register C). The sequence 10, XEQ \(\text{CU}\) raises the curtain 10 registers. Then the sequence \(-10, \text{XEQ \(\text{CU}\)}\) lowers it to its previous position. Y, Z preserved in X, Y; \(\Sigma\)-reg absolute location is unchanged. Not interruptible with printer. See important warnings under \(\text{CX}\).

\[ S: 000 \quad R: A \quad F: (R) \quad SL: 6 \quad C: 60 \]

---

**CV**—**CURVE FIT**

XROM 20,08

GTO \(\text{CV}\) and the keyboard functions are:

- a: \(\Sigma\) –
- e: Initialize
- A: \(\Sigma\) +
- B: Solve Type j
- C: \(\wedge\)
- D: \(\wedge\)
- E: Solve Best

Curve Types:

1. Linear \(y = bx + a\)
2. Exponential \(y = ae^{bx} (a > 0)\)
3. Logarithmic \(y = b \ln(x) + a\)
4. Power \(y = ax^b (a > 0)\)

Key x ENTER y and press A for each data pair.

Key B returns: \(X: b\) \(Y: a\) \(Z: r\)

Key E returns: \(X: j\) \(Y: b\) \(Z: a\) \(T: r\)

- R06: fct. #
- R07: cv type
- R08: b, x
- R09: a, y
- R10: r
- R11: \(\Sigma\)xln(y)
- R12: \(\Sigma\)yln(x)
- R13: \(\Sigma\)x
- R14: \(\Sigma\)x^2
- R15: \(\Sigma\)y
- R16: \(\Sigma\)y^2
- R17: \(\Sigma\)xy
- R18: \(\Sigma\)n
- R19: \(\Sigma\)ln(x)
- R20: \(\Sigma\)ln(x)^2
- R21: \(\Sigma\)ln(y)
- R22: \(\Sigma\)ln(y)^2
- R23: \(\Sigma\)ln(x)ln(y)
- R24: n
- R25: best r
- R26: best cv type

Store call # in R06 and XEQ \(\text{CV}\)

\[ \wedge \quad \wedge \]

0: clear/init. 1: \(\Sigma\) + 2: solve type \(j\) 3: \(Y\) 4: \(X\)
5: Solve Best Type 6: Σ – (when F10 set)  9: stop in

| S: 027 | R: 6-26 | F: 8-10 | SL: 4 | C: 42 |

CX — CURTAIN TO DECIMAL ADDRESS  XROM 10,33 IN X

Places hex version of X into the appropriate 3-nibble slot (the pointer to R00) in status register c. **WARNING:** Values 17 through 192 (mod 1024) or greater than 512 (mod 1024) may result in "MEMORY LOST." Y preserved in X, Y; T contains old c. Σ-reg absolute location is not changed. Not interruptible if printer is attached.

| S: 000 | R: A | F: (R) | SL: 5 | C: 60 |

DC — DECIMAL TO CHARACTER  XROM 10,11

Similar to Extended Functions XTOA. Appends to alpha contents the character corresponding to the decimal integer (mod 256) in X. Y, Z preserved in X, Y; (L contains old X + 256).

| S: 000 | R: A | F: (R) | SL: 6 | C: 59 |

DF — DECIMAL TO FRACTION  XROM 20,13

Store display accuracy (0-9) in R07. Input decimal in X and XEQ DF. Reduced fraction is left in Y (numerator) and X (denominator). In addition, if flag 10 was set, the fraction is displayed in alpha. The keyboard function d will XEQ DF after positioning to FR, DF, NP, GN, or RN.


DP — DECIMAL TO PROGRAM POINTER  XROM 10,53

The inverse to PD. Converts a decimal byte address in X to a RAM program pointer that via a STO b (be sure to return to RAM first!) can resume program operation where desired. See Ab regarding program pointers. Byte addresses begin with 0 at the bottom of status register memory through 111; this is followed by a 'hole' from 112 to 1343; then follows user memory from 1344 to 3583 (for the full complement of user registers). Y is preserved.

| S: 000 | R: A | F: None | SL: 5 | C: 60 |
**DR** — **DELETE RECORD**

Store:
R07: 1st register of entire file
R08: number of registers per record
R09: number of records in the file.

To delete the kth record, input k in X and XEQ **DR**. R09 is automatically updated. Inverse routine is **IR**.

```
S: 010  R: None  F: None  SL: 5  C: 61
```

**DS** — **DISPLAY SET**

If FIX, SCI or ENG n is set then keying m XEQ **DS** changes the display mode to FIX, SCI, or ENG m. X, Y, Z, T are preserved in L, X, Y, Z.

```
S: 000  R: A  F: None  SL: 6  C: 60
```

**DT** — **DISPLAY TEST**

Turns on all 12 commas for 1 PSE (execution can be interrupted only at this point), then all display segments and annunciators except for the comma tails. Review the display, then push the PRGM switch and R/S to clean up. X, Y, and Z are preserved.

```
S: 000  R: A  F: (R)  SL: 6  C: 64
```

**E?** — **.END. FINDER ?**

Returns the absolute decimal address of the register containing the .END. in its last 3 bytes (bytes 02-00) by decoding the appropriate pointer digits in status register c. X, Y preserved in Y, Z.

```
S: 000  R: A  F: None  SL: 5  C: 60
```

**EP** — **ERASE PROGRAM MEMORY**

As long as flag 14 is clear and there is a program labeled “/ /” (as described below) in RAM, XEQ **EP** will clear programs following that labeled “/ /”.

```
LBL “/ /”
RCL b
END or RTN
```

(followed by at least 6 bytes before .END.)
If flag 14 is set or if there is no program "//" as described, all programs will be erased. XEQ EP should be followed by PACK to reinstate CATalog 1.

If keyboard assigned global labels are erased by EP, use ASN to clear keys, or read any program or status card. X is preserved.

| S: 000 | R: A | F: 14, 25 | SL: 0 | C: 60 |

**EX** — EXponent OF **X**

XROM 10,27

Replaces X by its exponent portion, −99 to +99. The old X is saved in L; Y, Z, and T are preserved.

| S: 000 | R: A | F: None | SL: 6 | C: 60 |

**F?** — FREE REGISTER FINDER ?

XROM 10,04

Returns the number of free (available) registers between the last used assignment register and the .END. register. There will be a non-zero fractional part 0.5 and flag 10 will be set whenever the top assignment register contains only a single assignment in the left half of the register. See PK. If I/O buffers (like Timer Alarms) are present they will be disrupted and the count will be inaccurate. See warnings under LF.

| S: 000 | R: A | F: 10 | SL: 3 | C: 61 |

**FD** — FIRST DERIVATIVE

XROM 20,11

Calculates f’(X). Store the following:

- R10: global label name of function
- R11: pointer to register containing X (function input)
- R12: step size

Flag 09 set selects quick approximation. F09 clear selects adaptive procedure. If F09 is clear, set F10 to view convergence. Estimate of first derivative is left in X and if F09 is clear an error estimate is left in Y. Maximum accuracy is 6½ digits. The keyboard function D will XEQ FD after positioning to IG, SV, or FD.

| S: 018 | R: 10-17 | F: 9, 10 | SL: 4 | C: 43 |
FINANCIAL CALCULATIONS

GTO   and the keyboard functions are:

\[ a: 12x \quad b: 12 \div \quad c: C/D? \quad d: B/E? \quad e: \text{Clear} \]
\[ A: n \quad B: \% i \quad C: PV \quad D: \text{PMT} \quad E: FV \]
\[ H: \text{CF} \quad I: \text{PF} \quad J: \text{Status} \]

Register usage:

R01 through R05 hold the values defined by the keys above, plus R06: Fcn call #, R07: \% i as decimal. F08 set/clear = C/D. F09 set/clear = B/E. Set F10 to view convergence when solving for \% i. Accuracy for \% i is set by display mode.

Store call # in R06 and XEQ   

0: clear/init. 1: solve \( n \) 2: solve \% \( i \) 3: solve \( PV \) 
4: solve \( \text{PMT} \) 5: solve \( FV \) 12: stop in

FLAG INPUTS FOR

Provides the seven decimal numbers for a synthetic text line from the flags set inputs. To use XEQ , key first flag set, R/S. If \( X \) is negative, key next flag, R/S. If TONE sounds, record byte number, R/S. Key flags set in order of lowest to highest. Use 56 as last input if required. To use start with 247, followed by seven bytes. In a program use sequence: Text line, RCL M, STO d. Related routines are and .

FRACTIONS

GTO   and the keyboard functions are:

\[ a: \text{GN} \quad b: \text{RN} \quad c: \text{GCD} \quad d: \text{DF} \quad e: \text{NP} \]
\[ A: + \quad B: - \quad C: \times \quad D: \div \quad E: \text{Reduce} \]

Set F10 and clear F29 to display fractional forms in alpha.

Store call # in R06 and XEQ   

1: + 2: - 3: \times 4: \div 5: \text{Reduce } Y/X 6: \text{GCD}(x,y)

GO TO .END.

provides a handy way to move out of ROM back to RAM by placing the program pointer at line 00 of the program file.
that contains the .END. of program memory. X, Y, Z, L are preserved.

S: 000  R: A  F: (R)  SL: 0  C: 60

**GN** — **GAUSSIAN RANDOM NUMBER**  XROM 20,15
**GENERATOR**

Input:

R06: Mean of distribution
R07: Standard deviation of distribution
X: Pointer to register with seed.

Generates two random numbers (in X and Y) with Gaussian distribution. Use DEG mode, X is preserved in Z, T. The keyboard function b will XEQ **GN** after positioning to FR, DF, NP, GN, or RN. **GN** calls **RN** as a subroutine.

S: 008m  R: 6, 7+  F: None  SL: 4  C: 36

**HA** — **HIGH RESOLUTION HISTOGRAM**  XROM 20,25
**WITH AXIS**

Prints a bar-chart bar whose height is proportional to the input in X, where X is between Ymin (stored in R00) and Ymax (stored in R01). Plot width (1 to 168 columns) is integer part; and column position of the axis (1 to 168) is fractional part of R02. Bar is printed from axis to value, either up or down. Character (ACCHR #) to print portion of bar is stored in R05. Store values in R00 to R03 and R05, place the number in X, XEQ **HA** and a bar is added to the print buffer. Returns Fix 4 mode.

S: 006  R: 0-5  F: 12, 13  SL: 5  C: 29

**HD** — **HIDE DATA REGISTERS**  XROM 10,20

Input k, XEQ **HD** to raise the curtain by k registers, and place in Rₖ (R00 after the curtain is raised) an alpha constant used by **UD** to quickly lower the curtain. Y, Z, T preserved in X, Y, Z; Σ-reg is set to R00 (former Rₖ). Not interruptible if a printer is attached.

S: k+6  R: 0, A  F: (R)  SL: 6  C: 64
**HN** — **HEX TO NNN**  
XROM 10,41

Converts up to 14 hex characters in alpha to a non-normalized number (NNN) in X. Leading zeros need not be entered, and the space character can be used instead of zero. There are two acceptable characters for each hex digit greater than 9: 'A' is equivalent to ';', 'B' to ';', 'C' to '<', 'D' to '='; 'E' to '>', 'F' to '?'. (The status of flag 10 is irrelevant.) X, Y, Z are preserved in Y, Z, T.

S: 000  R: A  F: (R)  SL: 6  C: 33

**HP** — **HIGH RESOLUTION PLOT**  
XROM 20,29

Plots 1 to 9 user RAM functions simultaneously in high resolution in the X direction (7 plot points per printed line). Store Ymin in RO0, Ymax in RO1, plot width (1 to 168 columns) in RO2, Xmin in RO8, Xmax in RO9, X increment in RO10. Function names (global labels 6 char’s or less) are stored in RO15 to RO23. Place the number of functions in X and XEQ HP. See HP writeup for flag usage and other options.

S: 040  R: 0-39, A  F: 4-10  SL: 3  C: 86

**HS** — **HIGH RESOLUTION HISTOGRAM**  
XROM 20,26

Prints a bar-chart bar whose height is proportional to the input in X where 0 ≤ X ≤ 1. Plot width (1 to 168 columns) is stored in RO4. Character (ACCHR #) to print bar is stored in RO3. Partial character (ACCOL #) to print portion of the bar is stored in RO5. Store values in RO3 to RO5, place the number in X, XEQ HS, and a bar is added to the print buffer.

S: 009  R: 6-8  F: None  SL: 5  C: 29

**IF** — **INVERT FLAG**  
XROM 10,49

Toggles flag (changes state: clear to set or vice versa) specified in X (0 through 55). Y, Z preserved in X, Y.

S: 000  R: A  F: A/R  SL: 6  C: 60

**IG** — **INTEGRATE**  
XROM 20,09

Store f(x) global name in R10. Set flag 10 to view iterations. Set display mode to indicate desired accuracy of result. Key in limits.
a ENTER b, XEQ IG. Integral approximation is returned in X.

Register usage:

R10: function label name  R15: S_k
R11: counter k  R16: (b-a)/4
R12: u_i  R17: (b+a)/2
R13: 1 - u_i^2  R18: M(k,0)
R14: 2^{1-k}  R19: M(k,1)

The keyboard function B will XEQ IG after positioning to IG, SV, or FD. IG uses R10-R29.

---

**IP** — INITIALIZE PAGE  

Stores information from status register c in absolute location 256 (decimal), the bottom register of the external portion of memory. This stored information will be used by PS to restore the proper pointers when the page is switched back on line. If you decide not to switch the page off line immediately after executing IP, execute PS (without actually switching pages — just R/S in response to the switching prompt) to clear location 256. X, Y are preserved. Σ-reg is set to 000 absolute.

---

**IR** — INSERT RECORD  

Store:

R07: 1st register of entire file
R08: number of registers per record
R09: number of records in the file

To make room to insert a new kth record, input k and XEQ IR. R09 is updated. Inverse routine is DR.

---

**JC** — JULIAN DAY NUMBER TO CALENDAR DATE  

Clear flag 10 for Gregorian (modern) calendar. Set flag 10 for Julian calendar. Key JDN in X and XEQ JC. Date is returned in stack as:

Z: year  
Y: month  
X: day of month

Valid from March 1, 0 AD (1BC). Inverse routine is CJ. The
keyboard function e will XEQ JC after positioning to BD, TE, PM, CM, CJ, or JC.

S: 000 R: None F: 10 SL: 5 C: 53

LOAD PART OF LB

Prepare the program area (identified by LBL "++") as for LB. Then the program-writing program uses the instruction XROM L- to initialize the byte-loading process and return without prompting for bytes. (See LB for the loading of individual bytes.)

S: 012 R: 6-11 A F: 8,9,25,50 SL: 4 C: 71

LOAD BYTES

Permits the loading of arbitrary bytes into program memory, to facilitate synthetic coding. At location in program where bytes are to be loaded, key in LBL "++, ++, ++, ++,..., XROM LB. The number of +'s should be at least n+6, where n is the smallest multiple of 7 not less than the number of bytes you intend to load using LB. With the program pointer somewhere in this keyed-in sequence, switch to RUN mode and press R/S. The prompt "HEX/DEC INPUT" which precedes the prompt for byte #1 reminds you that either hex or decimal input is acceptable, depending only upon whether the mode is ALPHA or not, at the time you key in an input. You can change the mode at will during the sequence. Press R/S after each entry. Pressing R/S without an entry tells the program you are through. You can back up a byte by using a negative entry or by XEQ 03 (even if you have told the program you are through, providing you have not begun to carry out the terminating instructions).

The prompt "SST, MORE +'S" indicates you do not have enough +'s to load even one byte. Pushing SST once will get you to LBL "++, ++", where in PRGM mode you can insert more +'s and restart. A reprompt can be obtained by XEQ 01.

The terminating procedure is prompted. The SST is in RUN mode. The DEL is in PRGM mode. Following your loaded bytes, there may be some final +'s (at most 6). The X-register contains a number of the form p.00q. The termination prompt called for DEL 00p. Positioned at the first of any final +'s in PRGM mode, DEL 00q will remove these excess +'s and the XROM instruction.

S: 012 R: 6-11 A F: 8,9,22,23,25,29 SL: 0 C: 71

16 PPC ROM POCKET GUIDE
**LF** — LOCATE FREE REGISTER BLOCK XROM 10,05

Add 16.016 to returned value to get bbb.eee where bbb and eee are the absolute decimal addresses of unused registers between the last used assignment register (if any) and the register containing .END. If flag 10 is set on completion of **LF**, the left half of the bbb register is occupied by a key assignment.

**WARNING:** Do not execute **LF** if I/O buffer exists (e.g., Timer Alarms). They will be disrupted and the free register block indices will be incorrect. Use the Save Alarms routine from PPC CJ V9N4P17.

| S:000 R:A F:10 SL:4 C:59 |

**LG** — PPC LOGO XROM 20,24

XEQ **LG**, ADV or PRBUF to print PPC Logo. SF 12 for double wide. 21 columns (can fit 10 char/line w/std logo; 18 char/line w/wide logo; half of these with wide char). Stack is preserved.

| S:000 R:M,N,O F:None SL:5 C:29 |

**LR** — LENGTHEN RETURN STACK XROM 20,02

Stores ≤ 5 return pointers (calls prior to that on **LR**) in data registers Rx and Rx+1 in a format permitting reinstatement by **SR**. Allows call depths as deep as available data storage space permits, provided that successive calls on **LR** are separated by no more than 5 levels of subroutine nesting. Y, Z, T are preserved in X, Y, Z (L contains old x plus 1).

| S:002m R:A/R,A F:None SL:A/R C:40 |

**M1** — MATRIX, INTERCHANGE XROM 20,33

TWO ROWS

Store:

R07: start reg. of matrix
R08: # columns in matrix

To interchange rows i and j, input i ENTER j XEQ **M1**.

| S:009m R:7,8+ F:None SL:4 C:61 |
**M2** — **MATRIX, MULTIPLY ROW BY CONSTANT**

See **M1**. To multiply row \(j\) by the constant \(k\), input \(k\) ENTER \(j\) XEQ **M2**.

| S: 009m | R: 7,8+ | F: None | SL: 4 | C: 61 |

**M3** — **MATRIX, ADD MULTIPLE OF ROW TO ANOTHER**

See **M1**. To add \(k\) times row \(i\) to row \(j\), input \(j\) ENTER \(i\) ENTER \(k\) XEQ **M3**. Row \(j\) will change. Row \(i\) will not change.

| S: 009m | R: 7,8, A | F: None | SL: 4 | C: 61 |

**M4** — **MATRIX, REGISTER ADDRESS TO \((i,j)\)**

See **M1**. Input register \(r\) and XEQ **M4**. Row number \(i\) is returned in \(Y\), column number \(j\) is returned in \(X\). Inverse routine is **M5**.

| S: 009m | R: 7,8, +0 | F: None | SL: 4 | C: 61 |

**M5** — **MATRIX, \((i,j)\) TO REGISTER ADDRESS**

See **M1**. Input \(i\) ENTER \(j\) and XEQ **M5**. Register number \(r\) is returned in \(X\). Inverse routine is **M4**. \(Z, T\) are preserved in \(Y, Z\).

| S: 009m | R: 7,8+ | F: None | SL: 5 | C: 61 |

**MA** — **MEMORY TO ALPHA**

Alpha recalls the contents of a block of data registers using an ISG control number in \(X\) of the form bbb.eeeii. Inverse of **AM**. \(L, Y, Z, T\) are preserved.

| S: 005m | R: A/R, A | F: None | SL: 5 | C: 16 |

**MK** — **MAKE MULTIPLE KEY ASSIGNMENTS**

Clear I/O Buffer (Timer Alarms) before using **MK**. See warning under **LF**.
XEQ **MK** first reports the number of registers available for assignments. (Should "NO ROOM" be reported, take corrective action before continuing, pressing R/S if your corrective action did not move the program pointer.) Then the program prompts "PRE|POST|KEY." Key in decimal equivalent of the first byte (prefix), ENTER, decimal equivalent of the second byte (postfix), ENTER, user keycode, R/S. After each successful assignment, the program will either prompt for another assignment or indicate no more available room with the message "DONE, NO MORE." (In the latter case, proceed as with the "NO ROOM" message.) If you do not wish to make any more assignments, there is no termination procedure; simply go on to whatever subsequent task you intended.

The messages "KEY TAKEN" followed by "KEYCODE?" is suggesting you select another keycode, since the one you specified is already in use. Your options are: (1) enter another keycode and press R/S; (2) enter zero, indicating your desire to restart the assignment in process, and press R/S; (3) delete the previous assignment to the key you want to use and press R/S.

The messages "NO SUCH KEY" followed by "KEYCODE" indicate that your keycode entry is illegal. Options (1) and (2) in the preceding paragraph apply to these error messages.

Error messages above and on/off will clear flag 20 and the program will recount key assignment registers to ensure no overlaps or gaps occur. Between key assignments, if PACK or SIZE are used, clear flag 20 before continuing assignments.

Not interruptible

| S: 012 | R: 6-11, A | F: 7-10,20,25 | SL: 3 | C: 61 |

**ML** — MEMORY LOST RESIZE TO 017 **XROM 10,12**

*ML* should only be executed immediately after MASTER CLEAR, when the curtain and .END. are at their MEMORY LOST positions. XEQ **ML** is essentially equivalent to XEQ SIZE 017, making more registers available for loading in long programs. Returns in FIX 2 mode. X, Y are preserved. Does not work on CX.

| S: N/A | R: A | F: (R) | SL: 0 | C: 64 |

**MP** — MULTIPLE VARIABLE PLOT (1-9) **XROM 20,28**

Plots 1 to 9 user RAM functions or numerical values simul-
Simultaneously in standard resolution (1 plot point per printed line).
Store Ymin in R00, Ymax in R01, plot width (1 to 168 columns)
in R02, Xmin in R08, Xmax in R09, X increment in R10. Function
names (global labels 6 char's or less) are stored in R15 to R23.
Place the number of function in X and XEQ MP. See MP
writeup for flag usage and other options.

| S: 035 | R: 0-34,M,N,0 | F: 4-10 | SL: 3 | C: 86 |

**M** — MEMORY TO STACK

XROM 10,48
Recalls five data registers in sequence and stores them in X,
Y, Z, T, and L. RO6 must contain the location of lowest register
of the five register block. To use: N STO 06, XEQ **MS**. (If N=0, then finish with RCL IND 06.) Inverse routine is **SM**.

| S: N+5 | R: A/R | F: None | SL: 5 | C: 46 |

**M** — MANTISSA OF X

XROM 10,28
Replaces X by its mantissa. Old X to L; Y, Z, T preserved.

| S: 000 | R: A | F: None | SL: 6 | C: 60 |

**N** — NTH CHARACTER

XROM 10,38
Extracts the Nth character (1<N<10) from the right end of alpha
register. This extracted character becomes the new contents
of X and the alpha register.
Old X to L; Y, Z preserved.

| S: 000 | R: A | F: 25 | SL: 6 | C: 63 |

**N** — NNN TO HEX

XROM 10,40
Decodes a non-normalized number (NNN) in X to 14 hex
characters in alpha. X, Y, Z are preserved. The characters for hex-
decimal digits greater than 9 depend on flag 10:

Flag 10 (set): : ; < = > ?
(clear): A B C D E F (slower)

| S: 000 | R: A | F: 10 | SL: 6 | C: 33 |
**NP** — **NEXT PRIME**

XROM 20,14

Input integer \( n \) in \( Y \) and starting trial divisor \( d \) in \( X \). (\( d = 2 \) or \( d \) is an odd number greater than 2.) \( \text{XEQ NP} \). \( n \) is returned in \( Y \) and next divisor \( p \) is returned in \( X \). Press R/S for the next factor. The keyboard function \( \text{e} \) will \( \text{XEQ NP} \) after positioning to \( \text{FR}, \text{DF}, \text{NP}, \text{GN}, \text{or RN} \).

| S: 000 | R: None | F: None | SL: 5 | C: 36 |

**NR** — **NNN RECALL**

XROM 20,50

Recalls non-normalized number (NNN) stored by \( \text{NS} \) in \( R_x \) and \( R_{x+1} \). The NNN replaces the contents of \( X \). \( Y, Z, T, L \) are preserved.

| S: 002m | R: A/R, A | F: None | SL: 6 | C: 16 |

**NS** — **NNN STORE**

XROM 20,49

Stores \( Y \) and \( R_x \) and \( R_{x+1} \) in a format that allows recall by \( \text{NR} \) of the NNN. \( Y, Z, T \) preserved in \( X, Y, Z \); (L contains old \( X \) plus 1).

| S: 002m | R: A/R, A | F: None | SL: 6 | C: 16 |

**OM** — **OPEN MEMORY**

XROM 10,58

Places curtain at absolute decimal address 16 (just above the status registers). \( \text{OM} \) returns with the former contents of status register \( c \) in \( X \). \( X, Y, Z, L \) preserved in \( Y, Z, T, L \).

| S: 000 | R: A | F: (R) | SL: 5 | C: 60 |

**PA** — **PROGRAM POINTER ADVANCE**

XROM 10,59

\( \text{PA} \) is a selectable byte jumper (not programmable). In PRGM mode position the program pointer to the line from which you want to byte jump. In RUN mode fetch the pointer with a \( \text{RCL b} \). Then key \( n \) and \( \text{XEQ PA} \) to yield a program pointer in \( X \) moved by \( n \) bytes: a plus integer \( n \) moves the pointer downward in user memory, i.e., forward in the program. A \( \text{STO b} \) will effect the actual byte jump. Switch to PRGM mode and see a line 00 display and press SST. (This SST from line 00 does not advance the program pointer. The HP-41 merely displays what it believes to be...
line 01.) A GTO. (any line) or a BST will reestablish correct line numbers. X is preserved in Y.

S: 000 R: A F: (R) SL: 0 C: 60

**PD** — **PROGRAM POINTER TO DECIMAL**  XROM 10,52

Converts a program pointer in X (from the last 2 bytes of status register b in RAM format) to a decimal byte address. Inverse routine is **DP**. Y is preserved.

S: 000 R: A F: None SL: 5 C: 60

**PK** — **PACK KEY ASSIGNMENT**  XROM 10,09

REGISTERS

To recover as many registers as possible, XEQ **PK** after several key assignment deletions. Interruptible, but execution must be completed. Returns number of assignment registers used in X. Flag 25 may end up set, so CF 25 after **PK** for safety. (See Warning on Page 17, under **LF**.)

S: 000 R: A F: 9, 10, 25 SL: 4 C: 59

**PM** — **PERMUTATIONS**  XROM 20,19

\[ P(n,k) = \text{number of permutations of } n \text{ objects taken } k \text{ at a time.} \] Input \( n \) ENTER \( k \) and XEQ **PM**. \( P(n,k) \) is returned in X. Z and T are preserved in Y, Z. The keyboard function C will XEQ **PM** after positioning to **BD**, **TB**, **PM**, **CM**, **CJ**, or **JC**.

S: 000 R: None F: None SL: 5 C: 53

**PR** — **PACK REGISTER**  XROM 20,45

Initialize: R10: base b R11: register pointer = j. To store the number n in position k in register Rj, key \( n \) ENTER \( k \) XEQ **PR**. Inverse routine is **UR**. Note Rj should be cleared-initialized prior to its first use.

<table>
<thead>
<tr>
<th>Data Range</th>
<th>Base b</th>
<th>Position Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>2</td>
<td>1-30</td>
</tr>
<tr>
<td>0-2</td>
<td>3</td>
<td>1-19</td>
</tr>
<tr>
<td>0-3</td>
<td>4</td>
<td>1-15</td>
</tr>
<tr>
<td>0-4</td>
<td>5</td>
<td>1-13</td>
</tr>
<tr>
<td>0-6</td>
<td>7</td>
<td>1-11</td>
</tr>
<tr>
<td>0-9</td>
<td>10</td>
<td>1-10</td>
</tr>
<tr>
<td>0-13</td>
<td>14</td>
<td>1-8</td>
</tr>
<tr>
<td>0-20</td>
<td>21</td>
<td>1-7</td>
</tr>
<tr>
<td>Data Range</td>
<td>Base b</td>
<td>Position Numbers</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>------------------</td>
</tr>
<tr>
<td>0-36</td>
<td>37</td>
<td>1-6</td>
</tr>
<tr>
<td>0-99</td>
<td>100</td>
<td>1-5</td>
</tr>
<tr>
<td>0-214</td>
<td>215</td>
<td>1-4</td>
</tr>
<tr>
<td>0-1413</td>
<td>1414</td>
<td>1-3</td>
</tr>
<tr>
<td>0-99999</td>
<td>100000</td>
<td>1-2</td>
</tr>
</tbody>
</table>

S: 012m  R: 10,11 +  F: None  SL: 4  C: 61

PO — PAPER OUT  XROM 20,51

Advances the paper five times if the printer is enabled, else it may be used as a delay of about 1/5 second.

S: 000  R: None  F: None  SL: 5  C: 16

PS — PAGE SWITCH  XROM 10,46

Input:

Y: module number to turn off
X: module number to turn on
Alpha: destination program name

Execution of PS from the keyboard or in a program implements a switch from page Y to page X by first storing information from status register c into absolute location 256 (decimal) of page Y, the bottom register of the switchable part of memory. Then if flags 10 and 24 are clear, a user prompt appears to switch page Y off and page X on.

Flag 10 is set, 24 is clear: two tones sound for switching page Y off and X on.

Flags 10 and 24 are set: a RTN is executed.

If PS continues, then location 256 of page X, which was previously initialized by IP or PS, is recalled and placed in status register c. Execution is then transferred to the destination program. X, Alpha are preserved in Y, Z. Σ-reg is set to address 256 absolute. Works on 41C only.

S: 000  R: A  F: 10, 24(R)  SL: 0  C: 46

QR — QUOTIENT REMAINDER  XROM 10,54

Replaces Y and X by (Y-Y mod X)/X (the quotient) and Y mod X (the remainder). Z, T are preserved; old X to L. Also, up to
m— RECALL DISPLAY MODE  XROM 20,05
Restores an earlier display mode stored by SD in Rx (the register designated by the number in X). Restores the earlier state of flags 16 through 55. X, Y, Z, T preserved in L, X, Y, Z.

m— RESET FLAGS  XROM 10,13
Stores hex 00 00 00 2C 02 80 00 in status register d; this resets flags to Master Clear status (except for FIX 2, rather than FIX 4). X, Y, Z, T, L are unchanged. Sets flags 26, 28, 29, 38, 40.

m— REACTIVATE KEY ASSIGNMENTS  XROM 20,06
Transfers assignment bit maps stored in R_x and R_{x+1} (by SK) into status registers I- and e. This reactivates the suspended assignments. Y, Z, T preserved in X, Y, Z; (L contains old X plus 1).

m— RETURN ADDRESS TO DECIMAL  XROM 10,51
Converts a RAM return address (first return address, bytes 03 and 02 — third and fourth from the end of status register b) placed in X (by a RCL b for example) to a decimal byte address. See DP regarding legal byte addresses.
**RX** — **RECALL FROM ABSOLUTE ADDRESS IN X**

Only valid decimal addresses are 192 to 511 (64-511 with extended functions). **Content of register accessed is normalized** (so USE WITH CARE) and that is the form returned in X.

**WARNING:** If "NONEXISTENT" occurs restore the curtain (status register c) by ENTER, SST (in PRGM mode), R/S. Avoid using **RX** with flag 25 set. Y, Z preserved; **(L contains old X minus 16)**.

```
| S: 000 | R: A  | F: (R) | SL: 4  | C: 60 |
```

**Rb** — **RECALL b**

Yields an address in the PPC ROM permitting identification of the port it is plugged into. XEQ **Rb**, XEQ **NH**; the last 4 nibbles will be xF12, where X = 9, B, D, F for ports 1 through 4, respectively. X, Y, Z, L are preserved in Y, Z, T, L.

```
| S: 000 | R: None | F: None | SL: 6  | C: 16 |
```

**S1** — **STACK SORT**

Arranges (sorts) the stack registers, X, Y, Z, and T in numerical order, lowest value in T, highest in X. Set flag 10 to place the lowest value in X, highest in T. L is preserved, flag 10 cleared.

```
| S: 000 | R: None | F: 10   | SL: 5  | C: 47 |
```

**S2** — **SMALL ARRAY SORT (N≤32)**

Sorts any block of registers in memory into numerical order (lowest value in lowest register) using a block control number in X of the form bbb.eeeii. Optimum for arrays ≤32, or for non-contiguous registers. X is saved. Not interruptible.

```
| S: 003m | R: 1,2,A+ | F:10   | SL: 5  | C: 47 |
```

**S3** — **LARGE ARRAY SORT (N > 32)**

Sorts a block of registers in memory (from R03 up) into numerical order (lowest value in lowest register) using a block control

**PPC ROM POCKET GUIDE**

25
number in X of the form bbb.eee. For arrays \( \leq 32 \) or for non-contiguous registers use \( \text{S2} \). \( X \) is saved. Not interruptible.

\[
\begin{array}{llllll}
S: & 003m & R: & 1,2,A+ & F: & 10 \\
SL: & 4 & C: & 47
\end{array}
\]

\( \text{S?} \) — SIZE FINDER ?

XROM 10,15

Returns number of data registers (above curtain) in \( X \). \( X \), \( Y \), \( Z \) preserved in \( Y \), \( Z \).

\[
\begin{array}{llllll}
S: & 000m & R: & A & F: & (R) \\
SL: & 5 & C: & 64
\end{array}
\]

\( \text{SD} \) — STORE DISPLAY MODE

XROM 20,03

Saves a display mode in \( R_x \) (actually the status of flags 16 through 55). The earlier mode can be restored via \( \text{RD} \). \( X \), \( Y \), \( Z \), \( T \) preserved in \( L \), \( X \), \( Y \), \( Z \).

\[
\begin{array}{llllll}
S: & 001m & R: & A/R, A & F: & \text{None} \\
SL: & 6 & C: & 40
\end{array}
\]

\( \text{SE} \) — SELECTION WITHOUT REPLACEMENT

XROM 20,56

Store:

\( R06: \) 1st register of selection block
\( R07: \) number of registers in block (consecutive)

\( \text{SE} \) calls routine \( \text{RN} \). Register \( k \) holds the random number seed. To make selection, key in \( k \) and \( \text{XEQ SE} \). The selected value is returned in \( X \). \( R07 \) counts number of items remaining (as the block is rearranged).

\[
\begin{array}{llllll}
S: & 008m & R: & 6,7+ & F: & \text{None} \\
SL: & 4 & C: & 26
\end{array}
\]

\( \text{SK} \) — SUSPEND KEY ASSIGNMENTS

XROM 20,04

Stores assignment bit maps from status registers \( \text{I-} \) and \( e \) into \( R_x \) and \( R_{x+1} \), respectively, and clears both of these status registers. Without these bit maps, the HP-41 assumes no user key assignments are in effect. \( Y \), \( Z \), \( T \) preserved in \( X \), \( Y \), \( Z \); (\( L \) contains old \( X \) plus 1). Restore key assignments with \( \text{RK} \) or by reading in a program or status card.

\[
\begin{array}{llllll}
S: & 002m & R: & A/R, A & F: & \text{None} \\
SL: & 5 & C: & 40
\end{array}
\]

26

PPC ROM POCKET GUIDE
--- STACK TO MEMORY XROM 20,55

Stores the stack registers X, Y, Z, T, and L into a continuous block of five registers, the lowest register number being stored in R06. To use, input n, STO 06, XEQ SM. Inverse routine is MS.

--- SHORTEN RETURN STACK XROM 20,00

Recalls 5 return pointers stored in Rx and Rx+1 (by LR) combining them in status registers a and b with the return to the routine calling SR. Y, Z, T are preserved in X, Y, Z (L contains old X plus 1).

--- SUBSTITUTE CHARACTER XROM 10,39

With a single character in Y and an integer in the range of 1 through 10 in X (and with at most 13 characters in alpha), XEQ SU replaces the x-th character (counting from the right) of alpha by the character in Y. X, Y, Z preserved in L, X, Y.

--- SOLVE FOR ROOTS XROM 20,10

Store f(x) global label name in R06. Set display mode. Set flag 10 to view iterations. Input stepsize, ENTER, input guess, XEQ SV. (NOTE: if stepsize = 0 then default is 1 percent of guess.) Register usage:

R06: function label name
R07: X_i
R08: f(x_i)
R09: dx_i

The keyboard function C will XEQ SV after positioning to IG, SV, or FD.

--- STORE Y IN ABSOLUTE ADDRESS X XROM 10,56

Only valid decimal addresses are 192 to 511 (64-511 with extended functions). Contents of Y is placed in register addressed by X.
WARNING: If 'NONEXISTENT' occurs, the code in Z must be stored in status register c to regain the former curtain position (by SST in PRGM, then R/S). Y, Z are preserved in X, Y (L contains old x minus 16).

S: 000  R: A  F: (R)  SL: 4  C: 60

**Sb** — **STORE b**

XROM 20,01

Provides a STO b with the ROM-mode interpretation. This permits ultra-fast ROM entry. See AB. Stack is preserved.

S: 000  R: None  F: None  SL: A/R  C: 40

**T1** — **BEEP ALTERNATIVE**

XROM 10,47

Used in place of BEEP to produce a short sound burst (0.6 sec.) of 13 tones. Stack is preserved.

S: 000  R: None  F: None  SL: 5  C: 46

**TB** — **BASE TEN TO BASE B**

XROM 20,18

Store base b in R06 (2 ≤ b ≤ 19). Input base 10 number and XEQ TB. Base b result is limited to 13 digits in alpha and will be displayed if flag 10 is set. *A missing single quote indicates overflow.* The stack is cleared. Inverse routine is BD. The keyboard function B will XEQ TB after positioning to BD, TB, PM, CM, CJ, or JC.

S: 007  R: 6, A  F: 10,25  SL: 4  C: 53

**TN** — **TONE N (0-127)**

XROM 10,32

Effectively a TONE IND X instruction for synthetic tones. Place the TONE number in X, XEQ TN. Valid inputs: 0-127 for TONE, 128-255 for TONE IND, 256-383 TONE, etc. Y, Z are saved in X, Y.

S: 000  R: A  F: 14, 25  SL: 4  C: 60

**UD** — **UNCOVER DATA REGISTERS**

XROM 10,08

Uses the contents of R00 (established by HD) to lower the curtain to its previous position. *If R00 has not been initialized,
MEMORY LOST results. Use of **UD** after editing or **PACK** will lose CAT 1. Stack including L is preserved. ΣREG set to 01.

| S: 002 | R: 0, A | F: None | SL: 6 | C: 59 |

**UR** — **UNPACK REGISTER**

XROM 20,44

Store:

R10: base b

R11: register pointer = j

To recall the number stored in position k in register j. Input k in X and XEQ **UR**. The unpacked number is returned in X. Inverse routine is **PR**. Y is preserved in T.

| S: 012m | R: 10,11 | F: None | SL: 5 | C: 61 |

**VA** — **VIEW ALPHA**

XROM 10,07

Like AVIEW, but **VA** never causes the program to stop. If the printer is plugged in, turned on, and enabled by flag 21 being set, then the alpha register is also printed. The status of flag 21 remains unchanged and stack is preserved.

| S: 000 | R: (A) | F: 25 | SL: 6 | C: 59 |

**VF** — **VIEW FLAGS**

XROM 20,58

XEQ **VF** displays the numbers of those flags which are set in groups of four (although the last group may contain less than four). X and Y are preserved.

| S: 000 | R: A | F: (R) | SL: 4 | C: 26 |

**VK** — **VIEW KEY ASSIGNMENTS**

XROM 10,36

Defaults to printer function PRKEYS if printer is connected. Otherwise displays key codes assigned key by key from top to bottom, and from left to right (negative numbers correspond to shifted keys). If used in a program, first CLD or CF50. Stack is cleared.

| S: 000 | R: A | F: 21 | SL: 2 | C: 63 |
**VM — VIEW MANTISSA**

Views full mantissa of number in X. Leaves stack intact, but alters L and alpha register. *Not interruptible.*

| S: 000 | R: A | F: 21 | SL: 5 | C: 60 |

**VS — VERIFY SIZE**

Insert the sequence n, XROM VS, FC?C 25, PROMPT in a program to verify that the SIZE is at least n. If it is not, the message "RESIZE ≥ n" will be displayed following a quick tone. X, Y, Z and T are preserved in L, X, Y, and Z. Alpha is not disturbed if SIZE is sufficient.

| S: 000 | R: A | F: 25 | SL: 6 | C: 60 |

**XD — HEX TO DECIMAL**

Convert two hex digits — no more, no less — from the alpha register to the decimal equivalent (0 through 255). X is preserved in Y.

| S: 000 | R: A | F: None | SL: 4 | C: 71 |

**XE — XROM ENTRY**

XE Allows entry into ROM at any point using a pointer from the last two bytes of alpha register M, set up by the user prior to calling XE. A RTN activates this ROM entry address, and any subsequent END or RTN in the ROM program returns the pointer to the RAM program which called XE.

To set up the pointer in alpha register M, prior to program execution, position the pointer at the desired ROM line entry and manually perform the sequence (in RUN mode) CLA, RCL b, STO M, ASTO n, where Rn is any data register you want to use. In the RAM program which is to enter the specified ROM line place the sequence ARCL n, XEQ XE. Stack is preserved and alpha is cleared.

| S: 000 | R: A | F: 14 | SL: 4 | C: 64 |

**XL — XROM INPUTS FOR LB**

Provides two LB or MK bytes for user of synthetic instruc-
tions or synthetic key assignments. An XROM number of the form AA, BB is converted by AA ENTER BB XEQ \(*XL\). The Y register will contain the postfix bytes, and the X register the prefix byte. Z is preserved.

$$\Sigma ?$$ — \(\Sigma\)REG FINDER ?

XEQ \(*?\) yields the number of the first register of the statistical block of 6 registers. If the result is negative, the statistical block lies below the curtain (see \(*C\)). X is preserved in Y.

$$\Sigma C$$ — \(\Sigma\)REG CURTAIN EXCHANGE

Interchanges pointers in status register c to the statistical register block and to R00. Provided SIZE \(\geq k+6\), the sequence \(\Sigma\)REG k, XEQ \(*C\) makes the former \(R_k\) now R00, (but statistical register commands should not be used: \(*?\) yields \(-k\)). A second XEQ \(*C\) re-establishes the former position of the curtain (R00 again becomes \(R_k\)). X, Y, Z preserved, not interruptible if printer is attached.
NOTE: The tone durations shown here are based on ROM configuration NFL (HP-41CX). Other machines will have a few tones of different durations.
<table>
<thead>
<tr>
<th>+K</th>
<th>CX</th>
<th>LR</th>
<th>RX</th>
</tr>
</thead>
<tbody>
<tr>
<td>-B</td>
<td>DC</td>
<td>M1</td>
<td>Rb</td>
</tr>
<tr>
<td>1K</td>
<td>DF</td>
<td>M2</td>
<td>S1</td>
</tr>
<tr>
<td>2D</td>
<td>DP</td>
<td>M3</td>
<td>S2</td>
</tr>
<tr>
<td>A?</td>
<td>DR</td>
<td>M4</td>
<td>S3</td>
</tr>
<tr>
<td>AD</td>
<td>DS</td>
<td>M5</td>
<td>S?</td>
</tr>
<tr>
<td>AL</td>
<td>DT</td>
<td>MA</td>
<td>SE</td>
</tr>
<tr>
<td>AM</td>
<td>E?</td>
<td>MK</td>
<td>SK</td>
</tr>
<tr>
<td>Ab</td>
<td>EP</td>
<td>ML</td>
<td>SM</td>
</tr>
<tr>
<td>BA</td>
<td>EX</td>
<td>MP</td>
<td>SR</td>
</tr>
<tr>
<td>BC</td>
<td>F?</td>
<td>MS</td>
<td>SU</td>
</tr>
<tr>
<td>BD</td>
<td>FD</td>
<td>MT</td>
<td>SV</td>
</tr>
<tr>
<td>BE</td>
<td>FI</td>
<td>NC</td>
<td>SX</td>
</tr>
<tr>
<td>BI</td>
<td>FL</td>
<td>NH</td>
<td>Sb</td>
</tr>
<tr>
<td>BL</td>
<td>FR</td>
<td>NP</td>
<td>T1</td>
</tr>
<tr>
<td>BM</td>
<td>GE</td>
<td>NR</td>
<td>TB</td>
</tr>
<tr>
<td>BR</td>
<td>GN</td>
<td>NS</td>
<td>TN</td>
</tr>
<tr>
<td>BV</td>
<td>HA</td>
<td>OM</td>
<td>UD</td>
</tr>
<tr>
<td>BX</td>
<td>HD</td>
<td>PA</td>
<td>UR</td>
</tr>
<tr>
<td>BS</td>
<td>HN</td>
<td>PD</td>
<td>VA</td>
</tr>
<tr>
<td>C?</td>
<td>HP</td>
<td>PK</td>
<td>VF</td>
</tr>
<tr>
<td>CA</td>
<td>HS</td>
<td>PM</td>
<td>VK</td>
</tr>
<tr>
<td>CB</td>
<td>IF</td>
<td>PR</td>
<td>VM</td>
</tr>
<tr>
<td>CD</td>
<td>IG</td>
<td>PO</td>
<td>VS</td>
</tr>
<tr>
<td>CJ</td>
<td>IP</td>
<td>PS</td>
<td>XD</td>
</tr>
<tr>
<td>CK</td>
<td>IR</td>
<td>QR</td>
<td>XE</td>
</tr>
<tr>
<td>CM</td>
<td>JC</td>
<td>RD</td>
<td>XL</td>
</tr>
<tr>
<td>CP</td>
<td>L-</td>
<td>RF</td>
<td>Σ?</td>
</tr>
<tr>
<td>CU</td>
<td>LB</td>
<td>RK</td>
<td>ΣC</td>
</tr>
<tr>
<td>CV</td>
<td>LF</td>
<td>RN</td>
<td>RT</td>
</tr>
</tbody>
</table>