PPC ROM POCKET GUIDE

This pocket guide is preliminary. A lack of time, and inadequate inputs did not permit a "fine tuned" pocket guide. For this reason, a preliminary version is printed on paper. This will allow you to write in notes to serve as a reminder for you to send in your inputs. We plan on reprinting this pocket guide on a high quality material that will make it durable for field use. Pick your favorite routine and rewrite the short form instructions for it. Make it short, complete, and easy to use. If every ROM user carefully wrote up just <u>one</u> routine for the pocket guide we would have a first class Pocket Guide in no time. <u>All</u> important details should be included. Tables, charts, text, and special notations are suitable and should be used as required for a short write-up. Registers and Flags used, input formats, function, etc. should be included. Keep in mind to make it short and complete. The guideline for a pocket guide is short, short, short.

Each routine has an abbreviated Technical Details box. Abbreviations used in this box are:

SRL:	Subroutines left	C:	Registers to copy
I?:	Interruptible?	INP:	If no printer
F:	Program file	-:	Variable
A/R:	As Required		

+K - <u>A</u>DDITIONAL <u>K</u>EY ASSIGNMENTS

May be used <u>after</u> MK (which will set flag 20 and clear 07) or <u>after</u> 1K (which will set both flags 20 and 07). The sequence prefix↑postfix↑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. If in a program it is more convenient not to use 1K, note that the sequence CF 20, SF 07, XEQ +K is completely equivalent to XEQ 1K. Do not use data registers 09-11 between uses of +K.

XROM: 10,03	SIZE: 012	SRL: 3	I? NO	F: MK	C: 61

-B - STORE PART OF L<u>B</u>

L- and -B permit construction of programs that can write (create) other programs. See L- regarding initialization of the byte-loading process. With a decimal code of the byte to be loaded in X, the instruction XROM **B** will load the corresponding byte into the next location of the prepared area. Execution of such a 'program-writing' program will terminate with the prompts for the terminating 'clean-up' operations as with **LB** .

XROM: 10,24   SIZE: 012   SRL: 4   I? YES   F: LB   C: 71

1K - FIRST KEY ASSIGNMENT

This non-prompting program is suitable for making a key assignment under program control. The sequence prefix↑postfix↑keycode, XEQ **1K** will check registers and add the desired key assignment. Subsequent key assignments can be made using **+K** , providing data registers 09-11 are not used in the meantime. See **+K** for more information on making key assignments under program control.

XROM: 10,02   SIZE: 012   SRL: 3   I? NO   F: MK   C: 61

2D - DECODE 2 BYTES TO DECIMAL

Evaluates the decimal equivalents for the last two bytes in X. Returns the decimal equivalents for the next-to-last byte in X, for the last byte in M. (Use X<>M to view.) Y, Z preserved; X, T lost.

XROM: 10,55   SIZE: 000   SRL: 6   I? YES   F: IF   C: 60

A? - ASSIGNMENT REGISTER FINDER

Returns the number of assignment registers used. If the top assignment register contains only a single assignment (in the left half of the register), the fractional part of the result in X is 0.5.

XROM: 10,10   SIZE: 000   SRL: 3   I? YES   F: LF   C: 59

AD - ALPHA DELETE LAST CHARACTER

Removes the last (rightmost) character from the alpha register. X, Y, Z, and T are lost, but L is preserved.

XROM: 10,18   SIZE: 000   SRL: 6   I? YES   F: ML   C: 64

AL - ALPHABETIZE X AND Y

Sorts two alpha strings in X and Y: X and Y in

alphabetic order upon return (lowest value in X). If X and Y are numeric, they are regarded as pointers to two data registers which are to be sorted, the lowest value to the register designated by X. Flag 10 is left set or clear, depending on whether an interchange was performed or not.

XROM: 10,37   SIZE: 000   SRL: 4   I? YES   F: VK   C: 63

AM - ALPHA TO MEMORY

This routine alpha stores (ASTO, ASHF, ASTO, etc.) the contents of the ALPHA register into data registers using a control number in X of the form bbb.eeeii. Example. Store ALPHA in R01 thru R04. 1.004 XEQ **AM** . Inverse operation is **MA** .

XROM: 20,53   SIZE: 005   SRL: 5   I? YES   F: NS   C: 16

Ab - ALPHA STORE b

In ROM the mode of interpreting the last two bytes of status register b (which comprise the current location of the program pointer) is different than when in RAM. In ROM all 4 nybbles specify the current address. In RAM the first nybble (mod 8) specifies a byte within a register specified by the last 3 nybbles (mod 1024). **Ab** provides an ASTO b with the ROM-mode interpretation, permitting ultra-fast ROM entry as an alternative to **XE** .

XROM: 10,61   SIZE: 000   SRL: 0   I? YES   F: IF   C: 60

BA - BARCODE ANALYZER

XEQ **BA**. See SCAN prompt. Use wand, scan any barcode. Requires printer. Prints tabular analysis of barcode including type, every byte in binary, hex and decimal. Computes checksum to verify if it read the barcode correctly.

XROM: 20,30   SIZE: 019   SRL: 4   I? YES   F: BA   C: 49

BC - BLOCK CLEAR

Stores 0's in defined block.

Input:	T: T	Output:	T: T
	Z: Z		Z: Z
	Y: Y		Y: Y
	X: bbb.eeell		X: 0
	L: L		L: final control word

```
XROM: 20,43  SIZE: var.  SRL: 5  I? yes  F: M2  C: 61
```

BD - BASE B TO BASE DECIMAL

Store base b in R06 (2<=b<=25). Key digits of base b number in alpha and XEQ "BD". Up to 14 digits may be input in alpha where each digit is in the range from 0 to b-1 inclusive. The base 10 result is left in X and alpha is left cleared. The inverse routine is TB.

```
XROM: 20,17  SIZE: 007  SRL: 5  I? yes  F: BD  C: 53
```

BE - BLOCK EXCHANGE

Store two block control words bbb.eeeii in X and Y and XEQ "BE". When the two blocks are of the same size the order of the inputs in Y and X is not important. Otherwise only the block control word in Y is tested and controls the loop.

```
XROM: 20,34  SIZE: var.  SRL: 5  I? yes  F: M2  C: 61
```

BI - BLOCK INCREMENT

This routine is a register block storing routine that will store numerical data in a sequence, as a constant, or zero. Three inputs are required: bbb.eeeii ENTER start value ENTER increment value XEQ BI . X shows last value stored when routine stops, Y contains ISG value, Z and T are cleared, LASTX contains increment value.

```
XROM: 10,44  SIZE: A/R  SRL: 5  I? YES  F: BL  C: 46
```

BL - BLDSPEC INPUTS FOR LB

This routine converts BLDSPEC numbers into HP-41 memory bytes for a synthetic text line. Key first BLDSPEC number, XEQ BL . See first of seven bytes 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.

```
XROM: 10,42  SIZE: 000  SRL: 4  I? YES  F: BL  C: 46
```

BM - BLOCK MOVE

Applies only to a block of consecutive registers.
```
Input:  T: T
        Z: 1st register in source block
        Y: 1st register in destination block
        X: number of registers in source block
```

```
XROM: 20,39  SIZE: var.  SRL: 5  I? yes  F: M2  C: 61
```

BR - BLOCK ROTATE

Applies only to a block of consecutive registers.
```
Input:  T: T
        Z: Z
        Y: 1st register in source block
        X: ± number of registers in source block
```

If X>0 the majority of block shifts to higher numbered registers. If X<0 the majority of block shifts to lower numbered registers. Amount of shift is by one register for each call to BR.

```
XROM: 20,40  SIZE: var.  SRL: 5  I? yes  F: M2  C: 61
```

BV - BLOCK VIEW

This routine uses a defined block input in X (bbb.eeeii) to view the non-zero contents of the defined registers in the format R: NNN...N. Prints if printer is connected and on. For PSE at each register set flag 09. Set flag 10 to stop at each register.

```
XROM: 20,07  SIZE: A/R  SRL: 4  I? YES  F: SR  C: 40
```

BX - BLOCK EXTREMA

Input bbb.eeeii in X. If flag 10 is set the block will be considered non-negative. If flag 10 is clear BX may return negative values as extrema. Output:
```
M: register number of max value = INT part of M
N: register number of min value = INT part of N
O: bbb.eeeii = original block control word
Y: maximum value in block
X: minimum value in block
```

```
XROM: 20,41  SIZE: var.  SRL: 5  I? yes  F: M2  C: 61
```

BΣ - BLOCK STATISTICS

Assign appropriate location of the Σ-registers.

Enter two block control words of the form bbb.eeeii in X and Y. When the two blocks are of the same size the order of the inputs in X and Y is not important. Otherwise only the block control word in Y is tested and controls the loop. Output in Σ-registers:

Σx Σx^2 Σy Σy^2 Σxy n=# reg. in block

XROM: 20,42 SIZE: var. SRL: 5 I? yes F: M2 C: 61

C? - CURTAIN FINDER

Returns the absolute decimal address of R00 (curtain location). X, Y preserved in Y, Z; Z, T, L lost.

XROM: 10,16 SIZE: 000 SRL: 6 I? YES F: ML C: 64

CA - COMPLEX ARITHMETIC

SIZE 018 minimum. GTO "CA" and XEQ 06 to initialzie the complex stack. The following functions are on the top two rows of keys.

a: X_c<>Y_c	b: $Y_c^{X_c}$	c: Pop	d: LAST Z	e: e^z
A: +	B: -	C: *	D: ÷	E: LN(z)
initialize (XEQ 06)		H: sin(z)	I: cos(z)	J: Push

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
11:X<>Y 12:Y↑X 13:POP 14:LAST Z 15:e↑z
16:POP(save LASTz)

XROM: 20,23 SIZE: 018 SRL: 4 I? yes F: CA C: 38

CB - COUNT BYTES

Counts bytes between any two instructions in RAM. Position program pointer to beginning of sequence (first instruction in display). In RUN mode do a RCL b. Position program pointer to end of sequence (first instruction after the sequence in the display) and a second RCL b in RUN mode. Then XEQ CB yields the byte count. When counting bytes in a program (first instruction to END) it may be more convenient to do the 2nd RCL b at the END of the program. In this case, add 3 to the byte count returned by CB. Note that total byte count for a program 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, k)

XROM: 10,50 SIZE: 000 SRL: 4 I? YES F: IF C: 60

CD - CHARACTER TO DECIMAL

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; in X; X, Z, T, and L are lost.

XROM: 10,35 SIZE: 000 SRL: 6 I? YES F: VM C: 60

CJ - CALENDAR DATE TO JULIAN DAY NUMBER

Clear flag 10 for Gregorian (modern) calendar. Set flag 10 for Julian calendar. Input date in format:

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

XROM: 20,21 SIZE: 000 SRL: 5 I? yes F: BD C: 53

CK - CLEAR KEY ASSIGNMENTS

All USER mode assignments revert to the standard keyboard. Assignment registers are returned to the pool of unused registers without any need to PACK. Global label assignments are dormant until the next program card is read in.

XROM: 10,06 SIZE: 000 SRL: 4 I? YES F: LF C: 59

CM - COMBINATIONS

To compute C(n,k) key n ENTER↑ k XEQ "CM". The number of combinations of n objects taken k at a time is left in X.

XROM: 20,20 SIZE: 000 SRL: 5 I? yes F: BD C: 53

CP - COLUMN PRINT FORMATTING

Aligns numeric values for printing columns of data.

A skip index in R06 keeps decimal points in place. Index is calculated as (Max # digits 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 CP. The number will then be added to the buffer in the correct column position.

XROM: 20,27   SIZE: 002   SRL: 5   I? YES   F: LG   C: 29

CU - CURTAIN UP

Adds the integer value (including sign) of the contents of X to the pointer in status register c to register R00. The sequence 10, XEQ CU raises the curtain 10 registers. Then the sequence -10, XEQ CU lowers it to its previous position. Y, Z preserved in X, Y; X, T, L lost.

XROM: 10,34   SIZE: 000   SRL: 6   I? INP   F: VM   C: 60

CV - CURVE FIT

SIZE 027. GTO "CV" and the following functions are on the top row of keys.

a: Σ-				e: Initialize
A: Σ+	B: Solve Type j	C: $\hat{Y}$	D: $\hat{X}$	E: Solve Best

Curve Types:	1. Linear	$y = bx + a$
	2. Exponential	$y = a*e^{bx}$ (a>0)
	3. Logarithmic	$y = b*\ln(x) + a$
	4. Power	$y = a*x^{b}$ (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

Registers used:

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

Store call # in R06 and XEQ "CV"
0:clear/init. 1: Σ+ 2:solve type j 3: Y 4: X
5:Solve Best Type 6: Σ-(when F10 set) 9:stop in CV

XROM: 20,08   SIZE: 027   SRL: 4   I? yes   F: CV   C: 42

CX - CURTAIN TO ABSOLUTE DECIMAL LOCATION IN X

Places hex version of x into the appropriate 3-nybble slot (the pointer to R00) in status register c. Values 17 through 192 (mod 1024) or greater than 512 (mod 1024) will result in "MEMORY LOST". Y preserved in X; X, Z, T, and L are lost.

XROM: 10,33   SIZE: 000   SRL: 5   I? INP   F: VM   C: 60

DC - DECIMAL TO CHARACTER

Appends to alpha contents the character corresponding to the decimal integer (mod 256) in X. Y, Z preserved in X, Y; X, T, L lost.

XROM: 10,11   SIZE: 000   SRL: 6   I? YES   F: LF   C: 59

DF - DECIMAL TO FRACTION

SIZE 011. Store display setting (0-9) in R07. Set flag 10 to display fraction in alpha or clear flag 10 for no display. Enter decimal in X and XEQ "DF". Reduced fraction is left in Y (numerator) and X (denominator) and in addition, if flag 10 was set the fraction is displayed in alpha.

XROM: 20,13   SIZE: 011   SRL: 4   I? yes   F: FR   C: 36

DP - DECIMAL TO PROGRAM POINTER

The inverse to PD . Converts a decimal byte address in X to a program pointer that via a STO b 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). Among possibilities offered by DP (but requiring great care) are execution of programs set up in data registers or the stack. Only Y is preserved; X, Z, T, L are lost.

XROM: 10,53   SIZE: 000   SRL: 5   I? YES   F: IF   C: 60

DR - DELETE RECORD

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

To delete the kth record key in k and XEQ "DR". R09 is automatically updated. Inverse routine is IR

XROM: 20,38 SIZE: var. SRL: 5 I? yes F: M2 C: 61

DS - DISPLAY SET

DS provides a capability similar to the HP-67/97 DSP function, which sets the number of decimal places to be displayed without changing the display mode type (FIX, ENG, or SCI). If FIX, SCI or ENG n is set then keying m XEQ "DS" changes the display mode to FIX, SCI, or ENG m.

XROM: 10,29 SIZE: 000 SRL: 6 I? YES F: VM C: 60

DT - DISPLAY TEST

Turns on all 12 commas for 1 second (execution can be interrupted 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.

XROM: 10,17 SIZE: 000 SRL: 6 I? NO F: ML C: 64

E? - .END. FINDER

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

XROM: 10,62 SIZE: 000 SRL: 5 I? YES F: IF 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.

XROM: 10,31 SIZE: 000 SRL: 0 I? YES F: VM C: 60

EX - EXPONENT OF X

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

XROM: 10,27 SIZE: 000 SRL: 6 I? YES F: VM C: 60

F? - FREE REGISTER FINDER

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 whenever the top assignment register contains only a single assignment in the left half of the register. See PK.

XROM: 10,04 SIZE: 000 SRL: 3 I? YES F: MK C: 61

FD - FIRST DERIVATIVE

SIZE 018. Calculates f'(a). Program function f(x). FD uses R10-R17. Store the following:
R10: function global label name
R11: pointer to register containing x
R12: step size
Store a in register pointed to by R11.
Flag F09 set selects quick approximation. F09 clear selects adaptive procedure. If F09 is clear then F10 controls a display option. Set F10 to view convergence of optimal step size. Estimate of first derivative is left in X and in addition, if F09 is clear an error estimate is left in Y.

XROM: 20,11 SIZE: 018 SRL: 4 I? yes F: IG C: 43

FI - FINANCIAL CALCULATIONS

SIZE 010. GTO "FI" and the following functions are on the top two rows of keys:

a: 12x	b: 12 ÷	c: C/D?	d: B/E?	e: Clear
A: n	B: $I	C: PV	D: PMT	E: FV
		H: CF	I: PF	J: Status

F08 set/clear = C/D. F09 set/clear = B/E. Set F10 (optional) to view convergence when solving for $I.
Store call # in R06 and XEQ "FI"
0:clear/init. 1:solve n 2:solve $I 3:solve PV
4:solve PMT 5:solve FV 12:stop in FI

XROM: 10,63 SIZE: 010 SRL: 4 I? yes F: FI C: 47

FL - FLAG INPUTS FOR LB

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

XROM: 10,43 SIZE: 000 SRL: 4 I? YES F: BL C: 46

FR - FRACTIONS

GTO " FR ". The following functions are on the top row of keys:

a: RN	b: GN	c: gcd	d: DF	e: NP
A: +	B: -	C: *	D: ÷	E: Reduce

Set F10 to display fractional forms in alpha. Use FIX 0 and clear flag 29.

Store call # in R06 and XEQ " FR "
1:+ 2:- 3:* 4:/ 5:Reduce Y/X 6:GCD(x,y)

XROM: 20,12 SIZE: 000 SRL: 4 I? yes F: FR C: 36

GE -GO TO .END.

Places program pointer at line 00 of the last program file in memory (the one that contains the .END. of program memory). If you switch to PGM mode, the number of available registers is displayed (as when GTO.. is executed). GE provides a handy way to move out of ROM back to RAM.

XROM: 10,60 SIZE: 000 SRL: 0 I? YES F: IF C: 60

GN - GAUSSIAN RANDOM NUMBER GENERATOR

Use any data register to hold the seeds, say register k. Store initial seed in Rk. Store mean in R06 and store standard deviation = s in R07. 68% ±1s 95% ±2s 99.7% ± 3s. Use DEGREES mode. Enter k in X and XEQ " GN" to generate two random numbers, one in X and one in Y. Pointer k is returned in Z and T. GN calls the related routine RN .

XROM: 20,15 SIZE: 001 SRL: 4 I? yes F: FR C: 36

HA - HIGH RESOLUTION HISTOGRAM 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 the bar is stored in R03. Partial character (ACCOL #) 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.

XROM: 20,25 SIZE: 006 SRL: 5 I? YES F: LG C: 29

HD - HIDE DATA REGISTERS

If size ≥ k+6, the sequence k, XEQ HD raises the curtain by k registers, and places in former R_k (R00 after the curtain is raised) an alpha constant used by UD to easily reinstate the curtain position before the shift by HD . Y, Z, T preserved in X, Y, Z; X, L lost.

XROM: 10,20 SIZE: X+6 SRL: 6 I? INP F: ML C: 64

HN - HEX TO NNN

Converts up to 14 hex characters in alpha to an 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 '?'. Unlike the inverse function NH , the status of Flag 10 is irrelevant. X, Y, Z are preserved in Y, Z, T.

XROM: 10,41 SIZE: 000 SRL: 6 I? YES F: NH C: 33

HP - HIGH RESOLUTION PLOT

Plots 1 to 9 user RAM functions simultaneously in high resolution in the X direction (7 plot points 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 functions in X and XEQ HP. See HP writeup for flag usage and other options.

XROM: 20,29 SIZE: 040 SRL: 3 I? YES F: MP C: 86

HS - HIGH RESOLUTION HISTOGRAM

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 R04. Character (ACCHR #) to print bar is stored in R03. Partial character (ACCOL #) to print portion of the bar is stored in R05. Store values in R03 to R05, place the number in X, XEQ HS and a bar is added to the print buffer.

XROM: 20,26 SIZE: 006 SRL: 5 I? YES F: LG C: 29

IF - INVERT FLAG

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

XROM: 10,49 SIZE: 000 SRL: 6 I? YES F: IF C: 60

IG - INTEGRATE

SIZE 030. Program function f(x). IG uses R10-R29 and flag 09. Store global label function name in R10. Set flag 10 to display iterations. Set display mode. Key in limits a ENTER↑ b and XEQ " IG ". Integral approximation is returned in X.

R10: function label name	R15: S_k
R11: counter k	R16: (b-a)/4
R12: u_1	R17: (a+b)/2
R13: $1 - u_i^2$	R18: M(k,0)
R14: 2^{1-k}	R19: M(k,1)
	⋮

XROM: 20,09 SIZE: 030 SRL: 4 I? yes F: IG C: 43

IP - INITIALIZE PAGE

Store 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.

XROM: 10,45 SIZE: 000 SRL: 3 I? YES F: BL C: 46

IR - INSERT RECORD

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, key in k and XEQ " IR ". R09 is updated. Inverse routine is DR

XROM: 20,37 SIZE: var. SRL: 5 I? yes F: M2 C: 61

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   Inverse routine is CJ

XROM: 20,22 SIZE: 000 SRL: 5 I? yes F: BD C: 53

L- -LOAD PART OF LB

L- and -B permit construction of programs than can write (create) other programs. Prepare the program area (identified by LBL '++') as for LB . In the program-writing program place the instruction XEQ L- to initialize the byte-loading process and return without prompting for bytes. See -B for the loading of individual bytes.

XROM: 10,23 SIZE: 012 SRL: 4 I? YES F: LB C: 71

LB - 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

FUNCTIONAL GROUPING OF ROM ROUTINES

ALPHA REGISTER
AM AD MA NC SU
VA

BLOCK OPERATIONS
BC BE BI BM BR
BV BX BΣ DR IR
M1 M2 M3

CURTAIN
C? CU CX HD UD
ΣC

DISPLAY
DS DT RD SD

KEY ASSIGNMENTS
+K 1K A? CK MK
PK RK SK VK

LOAD BYTES
-B BL FL L- LB
XL

MATHMATICS
BD BX BΣ CA CJ
CM CV DF EX FI
FD FR GN IG JC
M1 M2 M3 M4 M5
MT NP PM PR QR
RN SE SV TB UR
VM

MATRIX
BX BΣ M1 M2 M3
M4 M5

MEMORY
E? EP F? IP LF
ML MS OM PR PS
RX S? SM SX UR
VS Σ?

MISCELLANEOUS
CJ GE IF JC RF
T1 TN VF

NON-NORMALIZED NUMBERS
2D CD DC HN NH
NR NS XD

PERIPHERALS
BA CP HA HP HS
LG MP PO XE

PROGRAM POINTER
Ab CB DP PA PD
Rb Sb XE

RETURN STACK
LR RT SR

SORTS
AL S1 S2 S3

HP-41C STATUS REGISTER USEAGE (0 THRU 15 ABSOLUTE)

DISPLAY. REPRESENTATION (NAME)	6	5	4	3	2	1	0	← BYTE NUMBER
e	SHIFTED KEY ASSIGN.					SCR-ATCH	LINE NO.	015
d	USER FLAGS (0-29)				SYSTEM FLAGS (30-55)			014
c	ΣREG			COLD START	REG 00		.END.	013
b	3rd RTN	2nd RETURN		1st RETURN		ADDRESS POINTER		012
a	6th RETURN		5th RETURN		4th RETURN		3rd RTN	011
F	UNSHIFTED KEY ASSIGN.					SCRATCH		010
Q		TEMPORARY ALPHA SCRATCH						009
P		SCRATCH FOR ALPHA 25-28			ALPHA REGISTER 22-24			008
O		ALPHA REGISTER 15-21 CHAR.						007
N		ALPHA REGISTER 8-14 CHAR.						006
M		ALPHA REGISTER 1-7 CHAR.						005
L			STACK REGISTER L					004
X			STACK REGISTER X					003
Y			STACK REGISTER Y					002
Z			STACK REGISTER Z					001
T			STACK REGISTER T					000
	sign	← mantissa →				sign	exp	REG. NO.

R/S without an entry tells the program you're through. You can back up a byte by XEQ 03 (even if you've told the program you're through, providing you haven't begun to carry out the terminating instructions).

The prompt "SST, MORE +'S" indicates you don't 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.

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 most6). 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 LB instruction.

XROM: 10,22 SIZE: 012 SRL: 0 I? YES F: LB C: 71

LF -LOCATE FREE REGISTER BLOCK

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.

XROM: 10,05 SIZE: 000 SRL: 4 I? YES F: LF C: 59

LG - PPC LOGO

XEQ LG, ADV or PRBUF to print PPC Logo. SF12 for double wide. 21 columns. Use to identify PPC materials, business cards, "MEMBER PPC" with PPC being the LOGO, etc.

XROM: 20,24 SIZE: 000 SRL: 5 I? YES F: LG C: 29

LR - LENGTHEN RETURN STACK

Stores ≤5 return pointers (calls prior to that on LR) in data registers R_x and R_{x+1} in a format permitting reinstatement by SR . Allows call depths as deep as available data storage space permits, providing that successive calls on LR are separated by no more than 5 levels of subroutine nesting. For example, calls at the 5th and 10th levels of nesting (together with the associated calls on SR) would permit successful return from calls nested 16 deep (and would use only

4 data registers). Y, Z, T preserved in X, Y, Z; X, L lost.

XROM: 20,02 SIZE: 002 SRL: ∞ I? YES F: SR C: 40

M1 - MATRIX, INTERCHANGE TWO ROWS

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

To interchange rows i and j, key i ENTER↑ j XEQ " M1 ".

XROM: 20,33 SIZE: var. SRL: 4 I? yes F: M2 C: 61

M2 - MATRIX, MULTIPLY ROW BY CONSTANT

See M1 . To multiply row j by the constant k, key k ENTER↑ j XEQ " M2 ".

XROM: 20,31 SIZE: var. SRL: 4 I? yes F: M2 C: 61

M3 - MATRIX, ADD MULTIPLE OF ROW TO ANOTHER

See M1 . To add k times row i to row j, key j ENTER↑ i ENTER↑ k XEQ " M3 ". Row j will change. Row i will not change.

XROM: 20,32 SIZE: var. SRL: 4 I? yes F: M2 C: 61

M4 - MATRIX, REGISTER ADDRESS TO (i,j)

See M1 . Key in register r and XEQ " M4 ". Row number i is returned in Y, column number j is returned in X. Inverse routine is M5 .

XROM: 20,35 SIZE: var. SRL: 4 I? yes F: M2 C: 61

M5 - MATRIX, (i,j) TO REGISTER ADDRESS

See M1 . Key i ENTER↑ j and XEQ " M5 ". Register number r is returned in X. Inverse routine is M4 .

XROM: 20,36 SIZE: var. SRL: 5 I? yes F: M2 C: 61

MA - MEMORY TO ALPHA

This routine alpha recalls the contents of a block of data registers using a control number in X of the form bbb.eeeii. Inverse of AM . Using the AM example:

1.004 XEQ MA "brings back" the previously stored alpha register contents.

XROM: 20,54 SIZE: 005 SRL: 5 I? YES F: NS C: 16

MK - MAKE MAKE MULTIPLE KEY ASSIGNMENTS

XEQ MK first reports the number of registers available for assignments. (Should "NO ROOM" be reported, take corrective action before continuing, by 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 don't 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.

At times (whenever Flag 20 is clear) the MK program will recount key-assignment registers to ensure no overlaps or gaps occur.

XROM: 10,01 SIZE: 012 SRL: 3 I? NO F: MK C: 61

ML - MEMORY LOSS RESIZE TO 017

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.

XROM: 10,12 SIZE: - SRL: 0 I? YES F: ML C: 64

MP - MULTIPLE VARIABLE PLOT (1-9)

Plots 1 to 9 user RAM functions or numerical values 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, X max 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 functions in X and XEQ MP. See MP writeup for flag usage and other options.

XROM: 20,28 SIZE: 035 SRL: 3 I? YES F: MP C: 86

MS - MEMORY TO STACK

This routine recalls five data registers in sequence and stores them in X, Y, Z, T, and L. R06 must contain the lowest register of the five register block. To use: N STO 06, XEQ MS .

XROM: 10,48 SIZE: 005 SRL: 5 I? YES F: BL C: 46

MT - MANTISSA OF X

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

XROM: 10,28 SIZE: 000 SRL: 6 I? YES F: VM C: 60

NC -NTH CHARACTER

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 preserved in L; Y, Z preserved; T, L lost.

XROM: 10,38 SIZE: 000 SRL: 6 I? YES F: VK C: 63

NH - NNN TO HEX

Decodes an NNN in X to 14 hex characters in alpha. X, Y, Z are preserved; T, L lost. The characters for hexadecimal digits greater than 9 depend upon whether or not Flag 10 is set. See table below. The conversion is much faster when Flag 10 is set.

Flag 10:	Set	Clear
	:	A
	;	B
	<	C
	=	D

>	E
?	F

```
XROM: 10,40  SIZE: 000  SRL: 6  I? YES  F: HN  C: 33
```

NP - NEXT PRIME

Key integer n in Y and starting trial divisor d in X. d=2 or d is an odd number greater than 2. XEQ "NP" n is returned in Y and next divisor p is returned in X. Press R/S for the next factor.

```
XROM: 20,14  SIZE: 000  SRL: 5  I? yes  F: FR  C: 36
```

NR - NNN RECALL

Recalls NNN stored by NS in R_X and R_{X+1}. The NNN replaces the contents of X. Y, Z, T, L are preserved.

```
XROM: 20,50  SIZE: 002  SRL: 6  I? YES  F: NS  C: 16
```

NS - NNN STORE

Stores Y in R_X and R_{X+1} in a format that allows recall by NR of the NNN in Y. Y, Z, T preserved in X, Y, Z; X, L lost.

```
XROM: 20,49  SIZE: 002  SRL: 6  I? YES  F: NS  C: 16
```

OM - OPEN MEMORY

Places curtain at absolute decimal address 16 (just above the status registers). Used by various ROM routines to permit access to key assignments and program memory. OM returns with the former contents of status register c in X. Former X, Y, Z, L preserved in Y, Z, T, L; T lost.

```
XROM: 10,58  SIZE: 000  SRL: 5  I? YES  F: IF  C: 60
```

PA -PROGRAM POINTER ADVANCE

PA is a selectable byte jumper. 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 RCL b. With this pointer in Y and an integer decrement in X (a negative integer would be an increment--i.e., backward in program memory), XEQ PA yields a pointer in X moved the specified number of bytes in the specified direction: a plus integer in X moves the pointer downward in user memory--i.e., forward in the program. In RUN mode a STO b on the output of PA will then effect the actual byte jump. Switch to PRGM mode and see a line 00 display. Press SST to see where the pointer is located. (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 re-establish correct line numbers. Only the original X (the decrement) is preserved in Y; X, Z, T, L are lost.

```
XROM: 10,59  SIZE: 000  SRL: 0  I? YES  F: IF  C: 60
```

PD - PROGRAM POINTER TO DECIMAL

Converts a program pointer (from the last 2 bytes of status register b in RAM format) to a decimal byte address. See inverse DP regarding byte addresses.

```
XROM: 10,52  SIZE: 000  SRL: 5  I? YES  F: IF  C: 60
```

PK - PACK KEY ASSIGNMENT REGISTERS

If key assignments deleted leave a number of assignment registers with only one active key assignment, PACKing will not merge two half-registers into one. To recover as many registers as possible, XEQ PK after several assignment deletions.

```
XROM: 10,09  SIZE: 000  SRL: 4  I? YES  F: LK  C: 59
```

PM - PERMUTATIONS

P(n,k) = number of permutations of n objects taken k at a time. Key n ENTER↑ k and XEQ "PM". P(n,k) is returned in X.

```
XROM: 20,19  SIZE: 000  SRL: 5  I? yes  F: BD  C: 53
```

PR - PACK REGISTER

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.

Data Range	Base b	Position Numbers
0-1	2	1-30
0-2	3	1-19
0-3	4	1-15

0-4	5	1-13
0-6	7	1-11
0-9	10	1-10
0-13	14	1-8
0-20	21	1-7
0-36	37	1-6
0-99	100	1-5
0-214	215	1-4
0-1413	1414	1-3
0-99999	100000	1-2

XROM: 20,45 SIZE: var. SRL: 4 I? yes F: M2 C: 61

PO - PAPER OUT

This routine advances the paper in the 82153A Printer five times. When printer is not connected and on it may be used as a delay of about 1/5 second.

XROM: 20,51 SIZE: 000 SRL: 5 I? YES F: NS C: 16

PS - PAGE SWITCH

With flag 10 clear, put the name of the destination program in ALPHA, and place the two page numbers in Y and X, then XEQ **PS** from the keyboard or in a program. **PS** implements a switch from page Y to page X as follows. **PS** first stores information from status register c into absolute location 256 (decimal) of page Y, the bottom register of the switchable part of memory. The program then prompts for the user to switch page Y off and page X on. When **PS** is restarted 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.

XROM: 10,46 SIZE: 000 SRL: 0 I? YES F: BL C: 46

QR - QUOTIENT REMAINDER

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; L lost. Also, up to 14 characters in alpha will be preserved.

XROM: 10,54 SIZE: 000 SRL: 6 I? YES F: IF C: 60

RD - RECALL DISPLAY MODE

Restores an earlier display mode stored in R_X by **SD**.

(Actually restores the earlier state of flags 16 through 55.) X, Y, Z, T preserved in L, X, Y, Z; L lost.

XROM: 20,05 SIZE: 001 SRL: 6 I? YES F: SR C: 40

RF - RESET FLAGS

Stores hex 00 00 00 2C 02 80 00 in status register d; this resets flags to Master Clear status (except (except for FIX 2, rather than FIX 4). X, Y, Z, T, L are unchanged.

XROM: 10,13 SIZE: 000 SRL: 6 I? YES F: ML C: 64

RK - RK - REACTIVATE KEY ASSIGNMENTS

Transfers assignment bit maps stored in R_X and R_{X+1} (by **SK**) into status registers F and E. This reactivates the suspended assignments. Y, Z, T preserved in X, Y, Z; X, L lost (L contains original X incremented by 1).

XROM: 20,06 SIZE: 002 SRL: 6 I? YES F: SR C: 40

RN - RANDOM NUMBER GENERATOR

Use any data register to hold seeds, say register k. Store initial seed in Rk. To generate next random number r, 0<r<1, key in k and XEQ "**RN**".

XROM: 20,16 SIZE: 001 SRL: 5 I? yes F: FR C: 36

RT - RETURN ADDRESS TO DECIMAL

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.

XROM: 10,51 SIZE: 000 SRL: 5 I? YES F: IF C: 60

RX - RECALL FROM ABSOLUTE ADDRESS IN X

Only valid decimal addresses are 192 to 511. Content of register accessed is normalized (so USE WITH CARE) and that is the form returned in X. WARNING: if "NONEXISTENT" occurs, the alpha constant in X must be stored in status register c to regain the former curtain position. Avoid using RX with flag 25 set. Y, Z preserved; X, T, L lost; L will contain the old X - 16.

XROM: 10,57   SIZE: 000   SRL: 4   I? YES   F: IF   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 nybbles will be xF12, where x = 9, B, D, F for ports 1 through 4, respectively.

XROM: 20,52   SIZE: 000   SRL: 6   I? YES   F: NS   C: 16

S1 -STACK SORT

This routine arranges (sorts) the stack registers, X, Y, Z, and T in numerical order, lowest value in T, highest in X. If Flag 10 is set, the lowest value will be placed in X, highest in T.

XROM: 20,46   SIZE: 000   SRL: 5   I? YES   F: S1   C: 47

S2 - SMALL ARRAY SORT (≤ 32)

This routine sorts any block of registers in memory into ascending order (lowest numerical value in lowest register) using a block control number in X of the form bbb.eeeii. Optimum time is obtained if block is ≤ 32. For larger arrays use S3. Routine finishes with block control number in X. No ALPHA allowed.

XROM: 20,48   SIZE: A/R   SRL: 5   I? NO   F: S1   C: 47

S3 - LARGE ARRAY SORT (> 32)

This routine sorts a block of registers in memory from R03 up, into ascending order (lowest numerical value in lowest register) using a block control number in X of the form bbb.eee. For arrays of 32 or less use S2. No ALPHA allowed.

XROM: 20,47   SIZE: A/R   SRL: 4   I? YES   F: S1   C: 47

S? - SIZE FINDER

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

XROM: 10,15   SIZE: 000   SRL 5   I? YES   F: ML   C: 64

SD - STORE DISPLAY MODE

Saves a display mode in R_X (actually the status of flags 16 through 55). If display mode (any of flags 16 through 55) is then altered, the earlier mode can be restored via RD. Y, Z, T preserved in X, Y, Z; X, L lost.

XROM: 20,03   SIZE: 001   SRL: 6   I? YES   F: SR   C: 40

SE - SELECTION WITHOUT REPLACEMENT

Applies only to block of consecutive registers. First store the following:

R06: 1st register of selection block
R07: number of registers in block

As with RN, use any data register to hold seeds, say register k. Initialize Rk with starting seed. To make next selection key in k and XEQ "SE". R07 counts number of items remaining.

XROM: 20,56   SIZE: var.   SRL: 4   I? yes   F: SM   C: 26

SK - SUSPEND KEY ASSIGNMENTS

Stores assignment bit maps from status registers ⊦ 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; X, L lost (L contains original X incremented by 1).

XROM: 20,04   SIZE: 002   SRL: 5   I? YES   F: SR   C: 40

SM - STACK TO MEMORY

This routine 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 store the stack into R01-R05: 1 STO 06, XEQ SM. The inverse routine is MS.

XROM: 20,55   SIZE: 005   SRL: 5   I? YES   F: SM   C: 26

SR - SHORTEN RETURN STACK

Recalls 5 return pointers stored in R_X and R_{X+1} by LR, combining them in status registers a and b with the return to the routine calling SR. Every call on **SR** must be preceded (in time) by a call on **LR**. Y, Z, T preserved in X, Y, Z; X, L lost (L contains old X plus 1).

`XROM: 20,00   SIZE: 002   SRL: ∞   I? YES   F: SR   C: 60`

SU - SUBSTITUTE CHARACTER

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; T, L lost.

`XROM: 10,39   SIZE: 000   SRL: 6   I? YES   F: VK   C: 63`

SV - SOLVE

SIZE 010 minimum. Program f(x) function. **SV** uses R06-R09. Store f(x) global label name in R06. Select display setting. Set flag 10 to view iterations. Key stepsize ENTER↑ guess XEQ "**SV**".

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

`XROM: 20,10   SIZE: 010   SRL: 4   I? yes   F: IG   C: 43`

SX - STORE Y IN ABSOLUTE ADDRESS X

Only valid decimal addresses are 192 to 511. Content of register accessed is replaced by content of X. WARNING: If 'NONEXISTENT' occurs, the code in Z must be stored in status register c to regain the former curtain position.

`XROM: 10,56   SIZE: 000   SRL: 4   I? YES   F: IF   C: 60`

Sb - STORE b

Provides a STO b with the ROM-mode interpretation. This permits ultra-fast ROM entry. See **Ab**.

`XROM: 20,01   SIZE: 000   SRL:A/R I? YES   F: SR   C: 40`

T1 - BEEP ALTERNATIVE

This routine may be used in place of BEEP to produce a short sound burst to be used as an audio prompt.

`XROM: 10,47   SIZE: 000   SRL: 5   I? YES   F: BL   C: 46`

TB - BASE TEN TO BASE B

Store base b in R06 (2<=b<=19). Key base 10 number in X and XEQ "**TB**". Base b result is limited to 13 digits in alpha and will automatically be displayed if flag 10 is set. The stack is left cleared. Inverse routine is **BD**.

`XROM: 20,18   SIZE: 007   SRL: 4   I? yes   F: BD   C: 53`

TN - TONE N (0-127)

This routine is effectively a TONE IND instruction which will produce all 128 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. **TN** is most suitable for quick test/Demo. purposes.

`XROM: 10,32   SIZE: 000   SRL: 3   I? YES   F: VM   C: 60`

UD - UNCOVER DATA REGISTERS

XEQ **UD** uses the contents of R00 established by the last call on **HD** to lower the curtain to its previous position. Every call on **UD** must be preceded (in time) by a call on **HD**. Otherwise you'll get MEMORY LOST. (See **HD**.) X, Y, Z, T, L preserved.

`XROM: 10,08   SIZE: A/R   SRL: 6   I? YES   F: LF   C: 59`

UR - UNPACK REGISTER

R10: base b	R11: register pointer = j

To recall the number stored in position k in register J, key k in X and XEQ "**UR**". The unpacked number is returned in X. See also **PR** for list of bases and data ranges. Inverse routine is **PR**.

`XROM: 20,44   SIZE: var.   SRL: 5   I? yes   F: M2   C: 61`

VA - VIEW ALPHA

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 no stack registers are used.

`XROM: 10,07   SIZE: 000   SRL: 6   I? YES   F: LF   C: 59`

VF - VIEW FLAGS

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, Z, T, L are lost.

XROM: 20,58 SIZE: 000 SRL: 4 I? YES F: SM C: 26

VK - VIEW KEY ASSIGNMENTS

Defaults to 82143A Printer function PRKEYS if printer is connected. Otherwise, displays keycodes assigned, key by key from top to bottom, and from left to right of the keyboard. The consecutive displays show either one number or two for a single key: negative numbers correspond to shifted keys. On completion of VK the stack is cleared.

XROM: 10,36 SIZE: 000 SRL: 2 I? YES F: VK C: 63

VM - VIEW MANTISSA

Views full mantissa of number in X. Leaves stack intact, but alters L and alpha register.

XROM: 10,26 SIZE: 000 SRL: 5 I? NO F: VM 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.

XROM: 10,30 SIZE: 000 SRL: 6 I? YES F: VM 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, Z, and T.

XROM: 10,25 SIZE: 000 SRL: 5 I? YES F: LB C: 71

XE - XROM ENTRY

This routine first constructs a local return pointer (to the PPC ROM) between (in status register b) the program pointer in XE and the return to the RAM routine calling XE. It then proceeds to replace this return with what were the last two bytes in alpha register M at the time of entry to XE. These two bytes constitute a pointer to any ROM program line, 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 R_n is any data register you want to use. In the RAM program which is to enter the specified ROM line place the sequence CLA, ARCL n, XEQ XE. X, Y, Z, T and L are preserved and alpha is cleared.

XROM: 10,19 SIZE: 000 SRL: 5 I? YES F: ML C: 64

XL - XROM INPUTS FOR LB

This routine provides the LB or MK bytes for use in synthetic instructions or synthetic key assignments. An XROM number of the form AA, BB is converted by A ENTER B XEQ XL. The Y register will contain the B byte, the X register the A byte. Example. Assign PRA to key 31. PRA = XROM 29,08. XL ⇒ 167, 72. Assign with 167, ENTER, 72, ENTER, 31, XEQ TK. The printer is not required.

XROM: 20,57 SIZE: 000 SRL: 4 I? YES F: SM C: 26

Σ? - ΣREG FINDER

XEQ Σ? 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, but is subject to statistical commands (at the user's risk).

XROM: 10,14 SIZE: 000 SRL: 5 I? YES F: ML C: 64

ΣC - ΣREG CURTAIN EXCHANGE

Interchanges pointers in status register c to the statistical register block and to R00. Provided SIZE ≥ k+6, the sequence ΣREG k, XEQ ΣC makes the former R_k now R00, (but statistical registers commands should not be used: Σ? yields -k). A second XEQ ΣC re-establishes the former position of the curtain (R00 again becomes R_k). X, Y, Z preserved; T lost.

XROM: 10,21	SIZE: A/R	SRL: 6	I? INP	F: ML	C: 64

NOTES

PPC