The ICEBOX.ROM is an ongoing project by Geir Isene - a continuation of the work laid down by Ángel Martin. He created the Toolbox.rom, a compilation of very useful MCODE routines, some written by himself and many written by several other MCODE experts. It is especially aimed at NoV users (see [http://www.clonix41.org)](http://www.clonix41.org)

This work is licensed under the GNU General Public License version 3.
This Quick Reference Guide is for ICEBOX.ROM version 1H, aka "Height".

Some functions inherited from the Toolbox.rom are still cryptic to me. If you have information that can help me fill out the question marks throughout this manual, please let me know.

XROM	*NAME*	*SHORT DESCRIPTION*	*SOURCE*
04,01	**>244**	**Convert HEX to 244 format**	**Geir Isene, ICEBOX.ROM**
Input:	10 bit hex word in 442-format in Alpha		
Output:	10 bit hex word in 244 format in Alpha		
Converts a 10 bit hex number from 442 format to 244 format.			
Example: The 10 bit word 1111111111 (binary) can be seen as 11 1111 1111 (244 format) or 1111 1111 11 (442 format). The hex values would be 3FF or FF3 respectively.			
04,02	**>442**	**Convert HEX to 442 format**	**Geir Isene, ICEBOX.ROM**
Input:	10 bit hex word in 244-format in Alpha		
Output:	10 bit hex word in 442 format in Alpha		
Converts a 10 bit hex number from 244 format to 442 format.			
Example: The 10 bit word 1111111111 (binary) can be seen as 11 1111 1111 (244 format) or 1111 1111 11 (442 format). The hex values would be 3FF or FF3 respectively.			
04,03	**A<>R**	**Swap Alpha with Registers**	**Geir Isene, ICEBOX.ROM**
Input:	N/A		
Output:	N/A		
This function swaps the content of the Alpha register with four register in RAM memory (the normal register). You can tell the function which registers you want as target register by setting flags 0-3. Please refer to the function X<>F to learn what the combinations of flags means.			
Example: Setting flags 0 and 1 will make register 03 the starting register for swapping. The four register that is swapped with Alpha are always consecutive (so in the example above, register 03-06 will be used).			
04,04	**aNN<>X**	**Absolute ex. Reg. with X**	**Ángel Martin, Toolbox.rom**
Input:	Content to be replaced with Reg NN in X, Reg# in Y		
Output:	Content of Reg NN in X		
Exchanges content in X with absolute register address given in Y without normalization.			
04,05	**aNRCL**	**Absolute NRCL**	**Ken Emery, MCODE for beginners**
Input:	Reg# in X		
Output:	Content of Reg# in X		
Recalls the contents of the registers without normalization, but more powerful because it uses the absolute address instead of the register number as input in X.			
Thus it is possible to recall anything from main memory, including status registers (from 0 to 17), buffers and Key Assignment areas, and even Extended-Memory registers.			

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,06	**aNSTO**	**Absolute NSTO**	**Ángel Martin, Toolbox.rom**

Input: Content to be stored in X, Reg# in Y

Output: N/A

See aNRCL

04,07	**BCD>BIN**	**Binary to BCD**	**Ken Emery, MCODE for beginners**

Input: Binary (hex as NNN) in X

Output: BCD (decimal) in X

Converts between binary and BCD. Used internally also as subroutines for other functions.

04,08	**BIN>BCD**	**BCD to Binary**	**Ken Emery, MCODE for beginners**

Input: Binary (hex as NNN) in X

Output: BCD (decimal) in X

Converts between BCD and binary. Used internally also as subroutines for other functions.

04,09	**BLNG?**	**Buffer Length Finder**	**W&W GmbH, RAMBOX ROM**

Input: Buffer id# in X

Output: Length of buffer in X

Returns the length in registers of the buffer which id# is provided in X. Buffers are created by different modules (CCD, Advantage, Plotter, etc) for temporary or permanent data storage, and it's beyond the scope of this manual to provide further details on their creation and properties. The following table (necessarily incomplete) lists some of the buffers known:

Buffer id#	*Module/Eprom*	*Reason*
1	David Assembler	MCODE Labels already existing
2	David Assembler	MCODE Labels referred to
3	Eramco RSU-1B	ASCII file pointers
4	Eramco RSU-1A	Data File Pointers
5	CCD Module, Advantage	Seed, Word Size, Matrix Name
6	Extended IL (Skwid)	Accessory ID of current device
7	Extended IL (Skwid)	Print Cols, number & width
8	Complex Stack	Ángel Martin's 41Z ROM
10	Time Module	Alarms information
11	Plotter Module	Data and barcode parameters
12	IL Development, CMT-200	IL buffer and monitoring
13	CMT-300	Status Info
14	Advantage	INTEG & SOLVE scratch
15*	Mainframe	Key Assignments
*) KA area isn't really a buffer.		

04,10	**BUF>R**	**Buffer to Register**	**Ángel Martin, Toolbox.rom**

Input: Buffer ID in X

Output: 0.XXX in X (where XXX is buffer length -1)

Saves buffer with buffer ID in X to registers 00-(N-1) and returns in X 0.(N-1) so that a buffer with a length of 4 is saved in registers 00, 01, 02 and 03 with 0.003 returned in X. This makes it easy to do a WRTRX. See the David Assembler manual for more information.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,11	**CALLXM**	**Call program in EM**	**Ross Wentworth, PPCJ,V12N3 p48**

Input: N/A

Output: N/A

Transfers program execution to a program in Extended Memory which global label name is stored in Alpha. The program can be anywhere in EM, but its entire length must be contained within a single XM module (or the XM included in the XF/M module). All GTO's must also be precompiled before hand, or the execution will fail.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,12	**CHKROM**	**Check ROM**	**HP Co., HP-IL Devel ROM**

Input: N/A

Output: N/A

This function tests the ROM with XROM number in X, to verify whether the value of its checksum word is correct. Input value is the XROM number, and the result is a message with the words "OK" or "BAD". The ROM id# is also shown while performing the calculation. (Sum of all word values, MOD 256).

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,13	**CLEM**	**Clear EM**	**Hakan Thorngren, PPC,JV13N2 p14**

Input: N/A

Output: N/A

Clears ALL Extended Memory.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,14	**CLMM**	**Clear Main Memory**	**Ángel Martin, Toolbox.rom**

Input: N/A

Output: N/A

Clears ALL of Main Memory.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,15	**CSST**	**Continuous SST**	**Phi Trinh, PPCJ,V9N7 p49**

Input: N/A

Output: N/A

Sequentially displays the program steps of the program pointed at by the Program Counter (PC). It's equivalent to using the SST key multiple times, and thus its name.

The delay between lines shown can be adjusted by pressing any keyboard key, see the original source for further details. To use it, position first the PC (Program Counter) at the target location (using GTO or similar).

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,16	**CUR?**	**Curtain location finder**	**Ángel Martin, Toolbox.rom**

Input: N/A

Output: Curtain address in X (NNN), curtain address in Aplha (hex)

Returns the absolute address of the curtain (separation between program and data registers).

The general equation is:

Total Registers = Data Regs + Program Regs,

Where: Total Regs=512 on the CV and CX models.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,17	**FDATA**	**Function Data**	**Klaus Huppertz, Prisma, Jan-90**

Input: Function name in Alpha (prompt)

Output: FAT address and XROM value in Alpha

Shows the FAT address and XROM value (the one used for key assignments) of the function input into the function's Alpha prompt. It works equally for mainframe functions, User Code programs in RAM, and MCODE functions in ROM.

Despite being an Alpha prompt function when invoked from the keyboard, FDATA is also programmable: when in a program, the function name will be taken from the Alpha register!

04,18	**FINDB**	**Find Buffer**	**Ángel Martin, Toolbox.rom**

Input: Buffer ID in X

Output: Absolute address of buffer and size of first free register (buf#.size)

Buffer ID will be saved to LastX if it exists or 0 if nonexistant (for testing).

If X register = 0, returns first free register in buffer area to X so ".n" can be appended and MAKEB called to create a new buffer.

04,19	**FLNG?**	**Disk File Length**	**Unknown, MMEPROM**

Input: N/A

Output: Length of mass storage file in X

Returns to X the length in registers of the (primary) mass storage file which name is specified in Alpha. If no HP-IL is present on the system an error message will be shown.

04,20	**FREG?**	**Free Registers Finder**	**Ken Emery, MCODE for beginners**

Input: N/A

Output: Number of free registers in X

Returns to X the number of available (free) program registers in Main Memory. No input value is required.

04,21	**GETN**	**Restore main memory**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A ("NONEXISTENT" if HEPAX data file "N" is not present)

Restores main memory from a file named "N" in HEPAX ram (must be created manually or by the function SAVEN first). This function calls HGETA with the parameter "N" in Alpha.

04,22	**GROM**	**Goto ROM address**	**Geir Isene, ICEBOX.ROM**

Input: ROM address in NNN in X

Output: N/A

Jumps directly to the ROM address given in X (in NNN). DO NOT USE THIS unless you know what you are doing.

This is a dangerously powerful function.

04,23	**HEX>NNN**	**Code**	**Ken Emery, MCODE for beginners**

Input: HEX value in Alpha

Output: NNN in X

This ia an improved version of the well-known CODE functions. The function is well-known and has been around for a long time, included already in the PPC ROM (routine "HN").

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,24	**HEX>VSM**	**Hex to VASM Oct**	**Ken Emery, MCODE for beginners**

Input: HEX value in Alpha

Output: VASM octal address

Routine to convert ROM address from HEX to the VASM Octal format used by HP. Input fields are automatically separated by the function, and the keyboard only admits numbers appropriate of the origin base (Hex or Octal).

04,25	**HEXIN**	**Hex Input**	**Hakan Thorgren, PPCJ,V13N4 p13**

Input: N/A

Output: NNN in X

Direct entry of Non-normalized numbers using its byte's HEX codes. Similar to CODE or HEX>NNN but interactive. HEXIN allows for a prompt message, if the alpha register contains any string before the function is executed. Enables only the keys of the HEX keyboard (0-9 and A-F).

04,26	**HXENTRY**	**Hex Entry**	**Ken Emery, MCODE for beginners**

Input: N/A

Output: NNN in X, HEX value in Alpha

Direct entry of Non-normalized numbers using its byte's HEX codes. Similar to CODE or HEX>NNN but interactive. HXENTRY stores the input code into Alpha as well as returning the NNN into X. Enables only the keys of the HEX keyboard (0-9 and A-F).

04,27	**KACLR**	**Clear Key Assignments**	**HaJo David. PPCJ,V12N4 p24**

Input: OK or OKALL in Alpha

Output: N/A

Clears all key assignments presently configured on the USER keyboard. Very similar to CLKEYS function of the X-Functions module, but with added functionality: it requires the literal string "OK" in the alpha register to perform the clearing. If the string "OKALL" is found, then not only the KA registers but all the buffers will be cleared as well.

04,28	**KALNG?**	**A Registers size finder**	**W&W GmbH, RAMBOX ROM**

Input: N/A

Output: Length of Key Assignment area in X

Returns the length in registers of the Key Assignment area in RAM memory. (Note that this cannot be done with BLNG? above, using 15 in X).

04,29	**KAPCK**	**Pack Key Assignments**	**HaJo David, PPCJ,V12N4 p24**

Input: N/A

Output: N/A

Packs the key assignments registers area of the 41 RAM memory. This can recover some registers held up for key assignments by the calculator but not being used, which frequently occurs after de-assigning keys.

04,30	**LKAOFF**	**Suspends Local KA**	**Ross Cooling, PPCJ,V13N2 p37**

Input: N/A

Output: N/A

LKAOFF Suspends the local key assignment, that is those in the first two rows un-shifted (A-J), plus the first row shifted (a-e). This permits the usage of these keys as local labels within a program, and thus not being overwritten by their global assignment.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,31	**LKAON**	**Reactivates Local KA**	**Ross Cooling, PPCJ,V13N2 p37**

Input: N/A

Output: N/A

LKAON reactivates the Key assignments suspended by LKAOFF. These two functions should be used together to temporarily suspend and then reactivate the local assignments.

04,32	**MAKEB**	**Make Buffer**	**Ángel Martin, Toolbox.rom**

Input: Buffer ID and size (in registers) (#.SIZE)

Output: ?

Given buf#.size (in regs.) in X, this function will create the buffer. If buffer ID exists, it will destroy and recreate the buffer and move content from old to new buffer (if the new buffer is smaller than the old buffer, only the first "new" n registers will be moved.

04,33	**MNFR**	**Mainframe Function**	**Clifford Stern, PPCJ,V12N3 p37**

Input: Three digit representative of mainframe function (prompt)

Output: N/A

This function prompts for a three-digit input representative of any mainframe function, as per the codes contained in the HEX Byte tables. Note that some values will invoke strange synthetic routines.

The following table shows some of the functions and their corresponding suffixes. Note how MFN conveniently accesses many of the non-programmable mainframe functions.

Suffix	MFN Function
000	CAT _
006	SIZE _ _ _
002	DEL _ _ _
003	CLP _
010	PACK
015	ASN _

04,34	**N100**	**Write h100 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “100” into address 4100. This is only useful if you have a NoV-32 or a NoV-64 module by Diego Diaz (see his web site http://www.clonix41.org for more information on these modules.

For the NoV-32, this function will activate HEPAX RAM bank #0.

For the NoV-64, this function will activate ROM Bank #1 and HEPAX RAM bank #0.

ICEBOX.ROM will naturally reside in ROM Bank #1 of a NoV-64. This is why I have not included any N20X functions as that would switch the pointer away from ICEBOX.ROM in the middle of the function (sawing off the branch that it sits on :)

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,35	**N101**	**Write h101 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “100” into address 4100. This is only useful if you have a NoV-32 or a NoV-64 module.

For the NoV-32, this function will activate HEPAX RAM bank #1.

For the NoV-64, this function will activate ROM Bank #1 and HEPAX RAM bank #1.

See N100 for more information.

04,36	**N102**	**Write h102 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “102” into address 4100. This is only useful if you have a NoV-64 module. This function will activate ROM Bank #1 and HEPAX RAM bank #2.

See N100 for more information.

04,37	**N103**	**Write h103 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “103” into address 4100. This is only useful if you have a NoV-64 module. This function will activate ROM Bank #1 and HEPAX RAM bank #3.

See N100 for more information.

04,38	**N200**	**Write h200 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “200” into address 4100. This is only useful if you have a NoV-64 module. This function will activate ROM Bank #2 and HEPAX RAM bank #0.

See N100 for more information.

04,39	**N201**	**Write h201 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “201” into address 4100. This is only useful if you have a NoV-64 module. This function will activate ROM Bank #2 and HEPAX RAM bank #1.

See N100 for more information.

04,40	**N202**	**Write h202 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “202” into address 4100. This is only useful if you have a NoV-64 module. This function will activate ROM Bank #2 and HEPAX RAM bank #2.

See N100 for more information.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,41	**N203**	**Write h203 to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: N/A

Write the hex value of “203” into address 4100. This is only useful if you have a NoV-64 module. This function will activate ROM Bank #2 and HEPAX RAM bank #3.

See N100 for more information.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,42	**NBS**	**NoV Block Switch**	**Geir Isene, ICEBOX.ROM**

Input NS prompts for the NoV bank number (100-103,200-203 – see N100)

Output: N/A (“DATA ERROR” if input value is not in the ranges 000-003, 100-103,200-203. “NON EXISTENCE” if HGETA is not found, “CALC OFF” if ROM block is switched)

This function switches the block to the configuration you enter at the prompt and then restores Main Memory to the file named “N” in the new HEPAX RAM block. If you switch the ROM block (the first of the three digits you enter at the prompt is different than the current value at the address 4100), it gives a brief message, “CALC OFF” to remind you that you must turn the calc off and on again for the ROM block switch to take effect.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,43	**NX**	**Write X to addr. 4100**	**Geir Isene, ICEBOX.ROM**

Input: Number (000, 001, 002, 003, 100, 101, 102, 103, 200, 201, 202 or 203) in X

Output: N/A

Write the value in X into address 4100. See N100 for more information.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,44	**N?**	**Write addr. In 4100 to X**	**Geir Isene, ICEBOX.ROM**

Input: N/A

Output: Number (100, 101, 102, 103, 200, 201, 202 or 203) in X

Write the value of address 4100 to X. See N100 for more information.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,45	**NNN>HEX**	**Decode**	**Clifford Stern, MCODE for beg.**

Input: NNN in X

Output: HEX value in Alpha

This is an improved version of the well-known DECODE functions. The function is well-known and has been around for a long time, included already in the PPC ROM (routine “NH”).

NNN>HEX will decode the NNN in X into the HEX code in Alpha, and (contrary to other implementations of this function) without leading zeros (i.e. no left-padding).

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,46	**NRCLX**	**Recall**	**Ken Emery, MCODE for beginners**

Input: Register number in X

Output: Register content in X

The content of register N (given in X) is returned to X.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,47	**OSREV?**	**Show OS revision**	**Ángel Martin, Toolbox.rom**

Input: N/A

Output: OS rev#

Shows HP-41 Operating System revision number.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,48	**PGTRAIL**	**Page Trail**	**Ángel Martin, Toolbox.rom**

Input: Page#

Output: ROM revision

Gives the complete ROM revision string for the page given in X.

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,49	**POPADR**	**Pop Address**	**Hakan Thorngren, Toolbox.rom**

Input: N/A

Output: N/A

Pops last return address from return stack

04,50	**R>BUF**	**Register to Buffer**	**Ángel Martin, Toolbox.rom**

Input: N/A

Output: N/A

Saves registers to Buffer. This function gets its information from REG 00 (the used as the “Header” of the buffer with the format in NNN as:

```
Bit:      | 13 | 12 | 11 | 10 | 09 | 08 | 07 | 06 | 05| |04 | 03 | 02 | 01 | 00 |
Content:  | ID  | ID |  SIZE   |  buffer data --------------------------------------->|
```

So, to save registers 01-07 as a buffer with ID #5, simply load in the Alpha register:

55080000000000 (first the buffer ID# twice, then the size, then 0's – you may use them for something else)

Then XEQ “HEX>NNN” and STO 00 before the R>BUF, and voilá the buffer is saved.

On power ON, the calculator will zero out the buffers' first ID# and let the various ROMs reclaim them (by putting back the first ID#. If no ROM reclaims a particular buffer, it is lost.

See the David Assembler manual for more information.

04,51	**RAMEDIT**	**Ram Editor**	**Hakan Thorgren, PPCJ V13 N4 p26**

Input: Start address in decimal in X or address in NNN in X or from PC (Program Counter)

Output: N/A

This function sets the calculator in RAM Editor mode. When invoked from the keyboard, it can take the start absolute address either from the decimal value stored in X, or from a right-justified NNN with the binary address in it. When invoked from a program it takes it from the current position of the program counter.

In either case, the display shows the register and nybble being edited, as well as the contents of the complete register. The cursor can be moved to the left and right with the USER and PRGM keys respectively, and the current digit where it’s positioned on will blink on the display.

Direct editing is possible using the redefined hex keyboard. Continuing to scroll in either direction shifts the cursor to the beginning or end of the register (indicated with a short warning tone), but doesn’t move up or down to the adjacent registers. Use the “+” and “–“ keys to actually move to the following or previous registers.

The input sequence terminates by pressing R/S or the back arrow key, which exits the RAM editing mode.

04,52	**RAND**	**Random number generator**	**Ken Emery, MCODE for beginners**

Input: Seed in X (0 - 1)

Output: Random number (0 - 1) in X

Simple random number generator. For a 41CX or C/CV with Time module use the following program to generate the first seed, then use RAND for fast random numbers:

```
01 LBL "RNG"         05 1 E49       09 LN1+X
02 DATE              06 *           10 R-D
03 TIME              07 PI          11 FRC
04 +                 08 MOD         12 END
```

XROM NAME SHORT DESCRIPTION SOURCE

04,53 ROMCAT ROM CATalog J.D.Dodin, Au Fond de la HP-41

Input: XROM number in X

Output: Catalog listing

Lists the functions on the module which XROM number is in X. Once the module is finished, the listing continues with all the other modules plugged in on pages with higher number than the first one.

04,54 RROM Read ROM Geir Isene, ICEBOX.ROM

Input: ROM address in X (NNN)

Output: ROM address word in Y (NNN)

Takes an address in X (in NNN format - use HEX>NNN to take an address in ALPHA and convert it to NNN format in X) and returns the NNN value from that address in Y (use NNN>HEX to get the hex value in ALPHA).

The address in the X register is incremented by one (makes it easy to view the ROM instruction-by-instruction).

04,55 SAVEN Save main memory Geir Isene, ICEBOX.ROM

Input: N/A

Output: N/A (“NONEXISTENT” if HEPAX data file “N” is not present)

Saves main memory from a file named “N” in HEPAX ram. This function calls HSAVEA with the parameter “N” in Alpha.

04,56 SHOWB Show Buffer Ángel Martin, Toolbox.rom

Input: N/A

Output: N/A

Puts a numeric list of Buffer IDs in Alpha each seperated by a space.

04,57 SROM Search ROM Geir Isene, ICEBOX.ROM

Input: ROM address (NNN) in X, WORD to search for in Y (NNN)

Output: ROM address where WORD is found in Y (or end of page)

Executing SROM will start the search immediately after the address you entered into X. It will return the address into X where it finds the first occurrence of the search word. You can then execute SROM again to find the next occurrence etc.

SROM will stop when it reaches the end of the block and return the start address of the next block (by again executing SROM, it will continue into the next block).

04,58 ST<>R Swap Stack with Registers Geir Isene, ICEBOX.ROM

Input: Content of Stack (including Last X), flags determine target register addresses

Output: Content in target registers swapped to Stack

This function swaps the content of the Stack registers with five registers in RAM memory (the normal register). You can tell the function which registers you want as target register by setting flags 0-3. Please refer to the function X<>F to learn what the combinations of flags means.

Example: Setting flags 2 and 3 will make register 12 the starting register for swapping. The five register that is swapped with Stack are always consecutive (so in the example above, register 12-16 will be used).

XROM	NAME	SHORT DESCRIPTION	SOURCE
04,59	**SUMROM**	**Sum ROM**	**George Ioannou, DF V3 N1 p10**
Input:	Page address in X		
Output:	ROM checksum (written into address FFF of page)		
Calculates the ROM Checksum and writes its value into the last word of the page being summed. Prompting function requests the page address (8 to F), to be input on the blinking field.			
04,60	**VSM>HEX**	**VASM Oct to HEX**	**Ken Emery, MCODE for beginners**
Input:	0		
Output:	0		
Routine to convert ROM address from the VASM Octal format used by HP to HEX. Input fields are automatically separated by the function, and the keyboard only admits numbers appropriate of the origin base (Hex or Octal).			
04,61	**WROM**	**Write ROM**	**Geir Isene, ICEBOX.ROM**
Input:	ROM address (NNN) in X, WORD to write in Y (NNN)		
Output:	N/A		
Takes an address in X and the value to write to that address in Y (both in NNN). This can only be used to write to EPROM RAM. This makes it possible to write 101h into the address 4100 to make the second HEPAX RAM block active.			
04,62	**XQ>XR**	**XEQ to XROM**	**W&W GmbH, RAMBOX ROM**
Input:	User code program name in Alpha		
Output:	N/A		
For a user code program which name is in Alpha, this function changes all the global XEQ lines calling other programs in the q-RAM space, converting them into their XROM equivalent. Use it once the function allocation and FAT is completed, as it will refer to the XROM and function numbers, instead to performing a label search based on the actual name. Execution of the program will be much faster, as the mentioned search will be avoided.			
04,63	**XROM**	**XEQ ROM**	**Clifford Stern, PPCJ,V12N3 p37**
Input:	XROM number (prompt)		
Output:	N/A		
A very special prompting function. Allows direct entry of any function included in a plug-in module, by introducing its XROM number first and then the function number. This allows access to ROM header functions, such us "–Sandbox 3d", (XROM 08,00). Note that while XROM is not programmable, the function called can be entered into a program, thus it isn't necessary that the ROM be present to introduce its corresponding functions.			

ICEBOX.ROM home page: http://isene.com/isene.cgi?hp-41

If you want other functions in the ICEBOX.ROM, please e-mail me at g@isene.com and ask for functionality – your wish may come true :-)

Geir Isene

Oslo, 2010-11-25