NoV-64 user manual

(For version 08r)

Contents

Introduction.

Since its introduction in October 2008 NoV-64 has become the preferred option for many HP-41 calculator enthusiasts. Top of the range Clonix modules family, the NoV-64 allows any user to gain full advantage of the whole(1) series of ROM Pacs available for the HP-41 system, as well as a perfect entry point into the amazing M-code programming world; and all that at a very small fraction of the cost.

This includes some of the most expensive devices built for this machine like the HEPAX, as well as some others that, although not that expensive at the time they were in production, have become collector items with outrageous prices.

The NoV-64 not only replicates Advanced HEPAX module but also extends its functionality way beyond the original module features. Below is a comparative table to show the main differences on PC interfacing, ROM and RAM between both devices.

	Advanced HEPAX	NoV-64
ROM:	16K (used by HEPAX code)	16K for HEPAX + 32K for user
RAM:	16K (erased when unplugged)	64K (non volatile RAM)
PC Int:	Via HP-IL	Via USB programmer or HP-IL

Internal structure.

To accomplish its features NoV-64 uses a microcontroller which includes 64K Flash ROM. The internal code required to operate the module is placed in the lowermost 4K while the remaining 60K are divided in three blocks (16Kwords each). Block #0 holds the HEPAX emulation code, this block is always active and it's auto-allocated in the lower available page by HEPAX module at power ON, in the same way the real HEPAX module does.

Blocks #1 & #2 are available for the user to place ROM images using the *ClonixConfig.exe* utility. When active, a ROM block is placed into pages #C to #F. Any empty page into a ROM Block can be used to plug a physical ROM module into the corresponding port.

Four non-volatile RAM chips are used for a total of 64K. Every chip holds 4 continuous pages (16K) and can be assigned to pages #8 to #B or #C to #F. Pages #8 to #B will always have a RAM chip assigned to as this is necessary for HEPAX to operate. When a RAM chip is assigned to pages #C to #F (32K operation) no ROM block can be active. See configuration details below for further info on 16K RAM and 32K RAM operation modes.

In order to keep full compatibility with HP-41 system, it is required to power cycle the calculator to change the active ROM block. Just the same way as if the user is about to insert or remove physical ROM modules.

On the contrary, RAM blocks are switched on the fly, since they are always placed in the same page range and always fill 4 pages.

WARNING!: Note that for 32K RAM operation, **NO** physical ROM modules(2) can be plugged.

Features.

Since NoV-64 is built using non volatile RAM, all memory pages, RAM and ROM, can be used to store any ROM image page and its contents will be preserved even if/when the module is unplugged.

This allows the user full control over 24 pages (16 RAM + 8 ROM). The higest storage capacity for a single module ever, and all of this including HEPAX functionality, which is claimed to be the "Holy Grail" of the HP-41 system.

NoV-64 adds a new feature to its memory size and HEPAX emulation. The unique RAM to Flash dumping tool which represents the simplest way to get a .ROM file out of your M-code or FOCAL programs writen into your RAM pages.

It also adds the ability to erase any previous image existent in page #F of the **ROM Block 2** in order to allow the user the possibility of dumping a new image from RAM without the need of reconfiguring the whole module.

Configuration.

NoV-64 is basically a RAM-Box including HEPAX emulation. Please refer to HEPAX manual, available at "www.hp41.org" for details on the functionality of HEPAX module.

The contents of NoV-64 ROM can be selected by the user with *ClonixConfig.exe* configuration utility and programmed with the USB programmer, according to user's needs. Please refer to *ClonixConfig user manual* for details on the procedure.

Once programmed the NoV-64 will show up its basic configuration: RAM block 0 is enabled at pages #8 to #B and ROM block 1 is enabled at pages #C to #F.

In its previous implementation (ver. 08p) NoV-64 allows a wide range of configuration possibilities, all of them included Adv. HEPAX emulation with 16K RAM.

Current NoV-64 software (ver. 08r) can be operated both in 16K RAM and 32K RAM configurations. Obviously, when 32K RAM operation is selected, ROM cannot be allowed since the whole range of addresses for external ROM in the HP-41 system (pages #8 to #F) are used by RAM memory.

To modify the configuration of the NoV-64, user must enter the appropriate **control word** at address H'4100. This can be done manually by means of "HEXEDIT" HEPAX command, (or any other ROM command which allows RAM writing), or with the recently available ICEBOXnn from Geir Isene, wich allows user command level (FOCAL) to reconfigure such **control word**. Please refer to Geir's web at "www.isene.com" for further details on this great tool.

The user should become familiar with configuration structure and rules for **control word** at H'4100 described in the following chapters.

Control Word structure: h'FHL.

In order to control 16K and 32K RAM operation modes the **control word** must held values according to the following criteria.

Please note that RAM to Flash dumping process is also triggered by the control word. This will be detailled in the corresponding section.

"F"= Two higher bits (b9 & b8), designates the active "Flash ROM Block"
0: No Flash ROM block enabled. (This is required to alow 32K RAM operation)
1 or 2: Enabled Flash ROM Block 1 or 2 (This disables 32K RAM operation)
3: Triggers RAM to Flash dumping or Flash erasing procedure.
See **PC interfacing** below for details on these new features.

"H"=Higher nibble of lower eight bits (b7 - b4), designates the RAM chip active for upper RAM block (pages #C to #F)
0: No RAM is active at pages #C to #F (16K RAM operation)
1, 2 or 3; RAM chip #1, 2 or 3 is active in the upper RAM block (32K RAM operation)
Bits b7 and b6 will be erased so no more values are allowed.

"L"=Lower nibble of lower eight bits (b3 - b0), designates the RAM chip active for lower RAM block (pages #8 to #B)
0, 1, 2 or 3; RAM chip #0, 1, 2 or 3 is active in the lower RAM block (both 16 and 32K RAM operation)
Bits b3 and b2 will be erased so no more values are allowed.

Normalization.

Normalization is achieved in order to avoid any invalid user entry to cause a system malfuction due to conflict among different pages of ROM and/or RAM.

The steps required to perform normalization are detailed below following the real order in which they are applied. These are automatic, so the user doesn't have to worry about it. It is explained here to help users understand why some values may appear different from the typed ones.

In case F=3, triggers RAM to Flash dump or Flash erasing if enabled. Then **control word** recovers its previous value. See **PC Interfacing**

In case F=1 or 2, H will become 0.

In case H or L >3 they'll become "mod 4".

In case H=L, H will become 0.

In case H<L and both are positive. Their values will be swapped.

Example 1: user types in "132", registered value will be "102".
F is kept as 1, H is zeroed since F is not zero and L remains as entered =2.

Example 2: user enters "053", registered value "031".
F is kept, H becomes “5 mod 4”=1, then H and L are swapped to keep H>L.

Valid configuration word list:

The following tables list the allowed values in the control word both for 16K RAM and 32K RAM olperation modes.

Note that although the user can type any value into **configuration word**, only these listed bellow will be kept. Any other value will either be normalized to one of the valid list or will perform a Flash operation (dump or erase) in which case the previous vaule will be restored after completion.

16K RAM operation.

Value	Active ROM	Active RAM
H'000	None	Chip # 0 in the Lower RAM block (pages #8 to #B)
H'001	None	Chip # 1 in the Lower RAM block (pages #8 to #B)
H'002	None	Chip # 2 in the Lower RAM block (pages #8 to #B)
H'003	None	Chip # 3 in the Lower RAM block (pages #8 to #B)
H'100	Block 1	Chip # 0 in the Lower RAM block (pages #8 to #B)
H'101	Block 1	Chip # 1 in the Lower RAM block (pages #8 to #B)
H'102	Block 1	Chip # 2 in the Lower RAM block (pages #8 to #B)
H'103	Block 1	Chip # 3 in the Lower RAM block (pages #8 to #B)
H'200	Block 2	Chip # 0 in the Lower RAM block (pages #8 to #B)
H'201	Block 2	Chip # 1 in the Lower RAM block (pages #8 to #B)
H'202	Block 2	Chip # 2 in the Lower RAM block (pages #8 to #B)
H'203	Block 2	Chip # 3 in the Lower RAM block (pages #8 to #B)

32K RAM operation.

Value	Active RAM Upper Block (#C - #F)	Active RAM Lower Block (#8 - #B)
H'010	Chip # 1	Chip # 0
H'020	Chip # 2	Chip # 0
H'030	Chip # 3	Chip # 0
H'021	Chip # 2	Chip # 1
H'031	Chip # 3	Chip # 1
H'032	Chip # 3	Chip # 2

PC Interfacing.

Since more NoV-64 users are developing growing interest in M-code programming, the need of a useful and simple way to get their M-code programs moved from the NoV-64 RAM to a PC file in .ROM format has also become a priority.

The HP-41 system is not renowned by its communication features. Proprietary I/O peripherals like the Wand, IR printer, etc. and HP-IL are definitely not the most easy way to handle comms with current PC technology.

Although HP-IL system is able to connect with a PC, the solution is bulky and expensive, and require a dedicated old fashioned PC with ISA bus. A quite more "elegant" solution is currenty available from J-F Garnier and the PIL-Box, which exceeds by far the simple requirement of dumping a RAM page into a .ROM file.

To accomplish this basic task, without the need of any extra device, the NoV-64 code includes and improves a feature previously developed to its predecessors NoVRAM and NoV-32.

Again, the **control word** is the key to access this feature. By entering H'3PR at address H'4100, RAM page pointed by "PR" will be dumped into Flash ROM. Nibble "R" stands for RAM chip number, and "P" stands for Page number within this chip.

WARNING!: To properly produce the .ROM image from a RAM page, **NO** ROM image can be present at page #F in Flash ROM Block 2. If the user has placed any ROM image in this page during the configuration procedure, or if a previous RAM to Flash dumping has been performed, no action will be taken and the memory contents will remain uncahnged. The word h'0FD (Flash dumping Disabled) will show up in address h'4101, to let the user know such circumstance and **control word** will keep its previous contents. A physical module may be inserted in port 4, this won't interfere with the RAM dumping process. If this was the case, and the module in port 4 is an 8K module, it must be removed before activating ROM block 2. Otherwise the newly created ROM image will conflict with the physical module. Card Reader or 4K modules can be kept in port 4 as they don't interfere with ROM image in page #F.

Allowed values for "P" & "R" are 0 to 3, thus RAM chip containig the page wich is goint to be dumped become the active RAM chip and fills pages #8 to #B. Values of "P" designates the page using the obvious scheme: 0=page #8, 1=page #9, 2=page #A and 3=page #B. Values other than these will be normalized before the dumping process begins.

It will take about 2.5 seconds for the NoV-64 dumping routine to burn the RAM image into its Flash ROM. Once finished, the **control word** will return to its previou value.

Once the above procedure has been accomplished, a .ROM file replica of our RAM is present into NoV-64 Flash ROM and therefore it is suitable to be converted into a PC file by reading your NoV-64 module with the USB Programmer. Note that the ROM image dumped from RAM is also fully functional and can be activated by enabling Flash ROM Block 2 in the **control word**.

To create the corresponding .ROM file, remove your NoV-64 from the calculator and plug it in the USB programmer. Run "microbrn.exe" and click the "Read" button, then click "Save" and choose an appropriate name for the file, no longer than 8 characters, and don't forget the .HEX extension. Close "microbrn.exe" after the .HEX file is saved.

Run "RAM2ROM4.exe" and enter the filename you choose in the step before. This will extract your ROM image from the .HEX file and builds a .ROM file which you will more likely want to share with the rest of the world.

At this point you may want to run *ClonixConfig.exe* in case you want any other .ROM image file into page #F. You may also leave your ROM image in page #F as it is fully operative.

Eventually, you may need to recover the Flash ROM area into page #F (ROM bank 2) in order to make it available for a RAM to Flash dumping. Should this was the case, just enter H'3FF into **control word** and page #F will be erased. Allowing a new RAM to be dumped. Note that this procedure only make sense if page #F already contains a ROM image. Otherwise, entering H'3FF in the **control world** will behave as a RAM to Flash dump command and will be normalized to H'333.

HEPAX File System.

Some users may still find the HEPAX FILE SYSTEM (FS) useful for their needs. If this is your case, please read the following carefuly.

Although NoV-64 fully emulates Adv. HEPAX, some of the unique features related to 32K RAM mode operation go beyond what the HEPAX module was designed to.

While 32K operation in real HEPAX is possible by means of HEPAX MEMORY modules, the 32K FS created by HEPAX is always structured in the same way: Adv. HEPAX RAM is placed on pages #8 to #B and the HEPAX MEMORY module is assigned to pages #C to #F.

NoV-64 introduces a far more flexible way to create 32K contiguous RAM areas, allowing a 16K block (chip) to be assigned either to pages #8 to #B or #C to #F.

This flexibility may however lead to an irregular FS structure. For example, assumes a clean (RAM erased) NoV-64. At power ON, HEPAX will create a FS (16K pages #8 to #B) on chip 0, the user may then sets control word to h‘001 and power cycle the HP-41; this will generate another identical FS on chip 1.

If control word is now set to h’010 (32K), chip 0 will be assigned to pages #8 to #B which is consistent with the FS previously generated on it. But chip 1 will be assigned to pages #C to #F and its FS structure which was also created to pages #8 to #B will no longer be consistent with its actual memory allocation. Therefore HEPAX FS chain is interrupted and HEPROOM will only show 2,610 registers corresponding to the FS on chip 0. Command HEPAX 002 however, will show HEPAX RAM from page #8 to #F, but FS is truncated after page #B.

NOTE: If you use your HEPAX RAM in a ROM-like fashion, i.e. out of the HEPAX FS, none of the following will apply. (see HEPAX manual for further info on how to get your HEPAX RAM out of the FS)

To properly create a 32K FS user must simulate the process as if it was about to be created inserting a physical HEPAX MEMORY module. To achieve this, first select the chip which is about to be placed in pages #8…#B, configure it in a 16K mode (control word h'00L) and power cycle your calc, then select the desired 32K config (h'0HL); pages #C…#F should be erased (see CLRAM HEPAX command) prior to power cycle the HP-41. This will ensure a fully functional 32K HEPAX FS and HEPROOM command will show 5,222 registers.

Please refer to the HEPAX manual for detailed info on the HEPAX FS, its structure, functionality and related commands.

WARNING!: Note that in some occasions attempt to run your HP-41 with a heavily damaged 32K RAM FS may lead to MEMORY LOST or not power ON. Should this happen, please remove the NoV-64 module from the calculator, power cycle or reset (ENTER+ON) it and reinsert NoV-64. The user should be aware of the different chip assignements used in the FS to preserve its integrity.

Crash Recovery Function.

Considering the amount of time required to write a 4K ROM, the idea of developing a method to recover from a severe crash without the need of erasing the whole memory was certainly quite appealing.

The CRF method takes advantage of the "Reset" feature of the HP-41: With calculator OFF and NoV-64 inserted, hold down ENTER key while pressing ON twice in quick sucession.

Your calculator will turn ON, HEPAX will be re-allocated into page #C temporarily. All HEPAX functions will be accessible. RAM will be read-disabled, (thus avoiding any polling-point conflicts), but write-enabled, so you can HEXEDIT, and erase any/all xFF4-xFFA contents, while keeping the rest of your work safe.

Note that the control word at H'4100 allows you to select the RAM chip you're gonna work with. Just keep in mind that since HEPAX is temporarily allocated into page #C, you should only select control words H'000, H'001, H'002 and H'003.

Once you've cleaned the offending areas, you can power cycle your HP-41 normally, the system will get back to life and you can get back to work on your favourite program(s).

The CRF may help if you want to avoid the total erasing procedure described in the following point.

RAM erasing procedure.

Should you decide to set your NoV-64 back to clean factory status, follow the procedure described herein.

Please remove you NoV-64, plug it in your USB programmer and run *microbrn.exe.* Click “Load” and select “CLR_RAM4.HEX”, then click “Program”.

You can also program the "CLR_RAM.HEX" by selecting it on the *ClonixConfig.exe* Windows configuration utility.

Once programming is finished, insert NoV-64 into any port and wait for about 25 seconds for the erasing procedure to complete. After that your HP-41 should show up a [CLR OK] message on the display; or [NO CLR]. The first one indicates that RAM is successfuly erased, you can now run *ClonixConfig* and configure your NoV-64 as usual.

If [NO CLR] shows up, please repeat the process and make sure you have your HP-41 running on a fresh set of alkalines. If the problem persists, please drop me a line to clonix41@gmail.com.

RAM & ROM shadowing.

This release of NoV-64 software includes the shadowing feature in a CY-like fashion. So when you plug a physical ROM Pac module into your HP-41, any RAM or ROM page at the module's addressing space will become hidden until the module is removed.

Please take into account that some RAM and ROM boxes do also include this feature and, therefore, the use of such RAM/ROM boxes along with NoV-64 is deprecated.

Note that the RAM contents will be preserved but, as usual, HEPAX file system may be affected (broken chain) if the module is inserted without removing its addresses from the HEPAX file system's chain. See HEPAX manual for details on the HEPAX file system chain structure.

RAMTOG command.

HEPAX allows the user a means to protect the contents of a given page by toggling a Write Protected flag. The flag itself is not readable so you can only toggle it, hence the name of the command. A message on the display will let you know the status after toggle is performed.

RAMTOG command on FS pages, must be avoided, as stated in the HEPAX manual (page 66), otherwise the FS chain will be broken, with subsequent data loss.

Since RAM page assignements does not change in real HEPAX (whether or not HEPAX MEMORY modules are used) RAM write protection status can be assigned to the page this RAM is assigned to.

NoV-64 on the contrary, can shift RAM pages from pages #8…#B to pages #C…#F and vice-versa. So the contents which is intended to be protected may be addressed into a different page after a configuration change.

Therefore a different scheme has been built within NoV-64 to ensure that every RAM page keeps its protection status regardless the configuration selected by user, even if HP-41 is turned OFF.

WARNING!: Note that RAM protection status of all RAM pages will be erased when NoV-64 is removed from its port.

NOTES:
(1) Peripheral modules (IR Printer, HP-IL, TIMER & ExtFunc/Memory module) cannot be replicated due their unique characteristics.

(2) System modules (IR Printer, TIMER, HP-IL* and Ext Memory) can be plugged along with the NoV-64 even in 32K RAM mode, since they does not conflict with its page mapping. (*) Please see HP-IL compatibility limitations in the HEPAX manual.