

Thermal 1/1.7x

Head Firmware Summary

719-00017A-A-PR01

01 Jan, 2003

Creo Inc.
3700 Gilmore Way
Burnaby, BC
Canada V5G 4M1
(604) 451-2700

Copyright 2003 Creo Inc. All rights reserved.

creo™

Thermal 1/1.7x
Head Firmware Summary

CONTENTS

1. Overview....1

2. Thermal Head Firmware....1

3. MPE Head Interface....3

4. MCE Head Interface....3

5. MPE/Head Compatibility....3
   - *5.1 fheadcal* Parameter Location in the MPE....4

6. Head NVS Configuration....4
   - 6.1 Handling Head NVS Parameters in the MPE....5
     - 6.1.1 Handling Generic Parameters in the MPE....6
   - 6.2 Handling Head NVS Parameters in the MCE....6
     - 6.2.1 Handling Generic Parameters in the MCE....7
   - 6.3 Handling STROKE and SCORR Tables....8
   - 6.4 Handling the STATUS Sub-System....9

7. MPE NVS Synchronization and Upgrade....9
   - 7.1 MPE ‘nvs save all creo’ Command....9
   - 7.2 MPE ‘nvs upgrade head <previous version>’ Command....9

8. MCE NVS Synchronization and Upgrade....11

9. MPE Problems and Solutions....11
   - 9.1 hdm Name Corruption in the MPE....11
   - 9.2 Temporary Changes to hdm Block Saved in MPE....11
   - 9.3 Firmware Upgrade and Head Installation Problems on MPE....12

10. Measuring Head Power....12
    - 10.1 Head Power Full....12
    - 10.2 Head Power Sum....12

11. Thermal Head Version Information....13

1. OVERVIEW

This document outlines some of the major facets of head firmware, focusing particularly on the NVS and changes made to the *hdm* block in version 1.74. It provides a general explanation of how the head and host interact, outlines firmware changes made since head firmware version 1.38, and discusses issues you may encounter when using the thermal head.

This document outlines head operation on both MPE and MCE platforms. However, most of the compatibility issues associated with the MPE are not present in the MCE. Because there are fewer compatibility issues relating to the MCE, this document focuses more on MPE platforms.

We are continually striving to make head firmware easy to use, while increasing functionality and reliability. In order to make this process as painless as possible, we make an effort to provide timely documentation of changes through firmware release notes, PSNs, and direct communications. It is our hope that by providing documentation we can help everyone understand why we make changes, and how to utilize change effectively. This way we can all better serve our customers, providing them with higher quality and better reliability, in a timely manner.

2. THERMAL HEAD FIRMWARE

The thermal 1/1.7x head firmware runs on a 16-bit TMS320C50 digital signal processor, located on the head controller board. The code space consists of two banks of 64K (16-bit) of RAM. One bank of the code has not been implemented. The code space is loaded from the flash during bootup. The data space consists of two banks of 64K (16-bit) of RAM, 512K (8-bit) of flash memory, and 32K (8-bit) of NVS memory. Both data banks have been implemented. The flash contains the firmware and LCA files.

The TH1/1.7x head does not have its own interface. Therefore, all head access happens through the host, and roll-out of head changes are highly host-dependent. Figure 1, on the next page, shows the MPE-head interface and firmware from the head side.

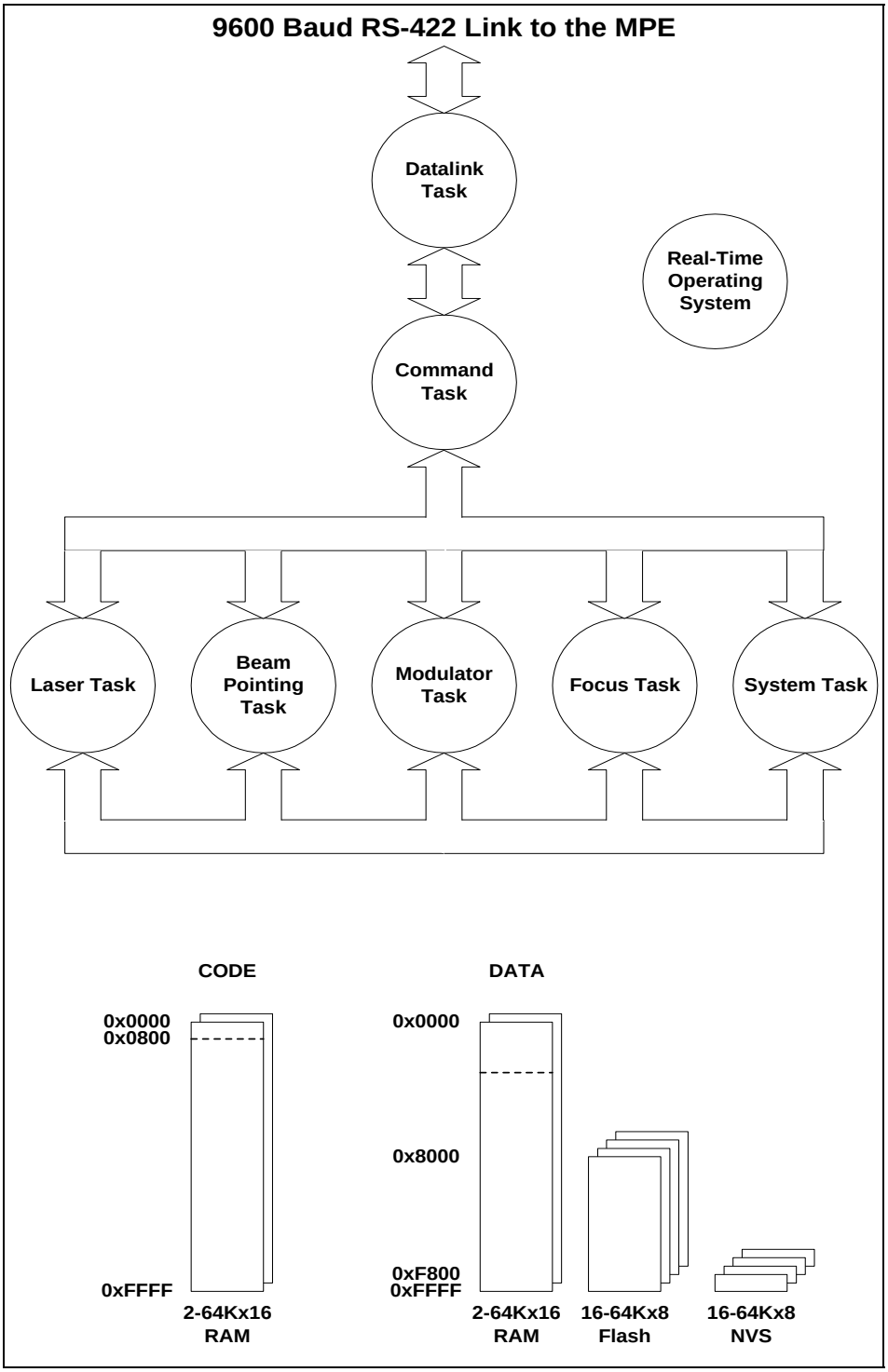


Figure 1: **Head interface and code structure**

3. MPE HEAD INTERFACE

The MPE firmware contains several sub-systems, or modules. These modules contain the code to control different parts of the device. For example, the drum module contains the code to control the drum, and the carriage module contains the code to control the carriage. Similarly, the thermal module contains the code that interacts with the thermal head.

The modules combine to create a single MPE build. Therefore, if you need a new version of a specific module, you must upgrade to an MPE firmware version that contains that module version.

Because MPE firmware is constantly being developed and released, many versions of the thermal module are present in the field. These different versions of the thermal module will cause the MPE firmware to interact with the head differently. Therefore, the ability to access much of the head functionality depends not only on the version of head firmware that you are using, but also the thermal module you are using.

In order to determine which version of the thermal module you are using, type **`>> list version`**. You will see a line that reads "Thermal software version: X.XX". The X's denote the specific version of the thermal module that you are using.

4. MCE HEAD INTERFACE

The MCE is Creo's next-generation device platform. In many ways, it is a major departure from our MPE-based platforms. To learn more about the MCE, and how it differs from the MPE, refer to the online documentation on TechPlanet. The documents below are a good starting point.

- MPE and MCE Firmware Differences (http://techplanet.creo.com/docs/firmwareDiff3.pdf)
- Mapping of MPE and MCE Commands (http://techplanet.creo.com/docs/MPE_commands.xls)
- MCE Firmware Page (http://techplanet.creo.com/output/computer_to_plate/trendsetter_3244/nav/mce_firmware.htm)
- MPE and MCE Hardware Differences (http://techplanet.creo.com/output/computer_to_plate/trendsetter_3244/content/docs/hardwareDiff.pdf)

The MCE does not use the thermal module. In an effort to simplify the head-to-host interface, the MCE employs the Creo Exposure Head (CEH) interface. This interface was intentionally designed to leave most of the details of head operation to the head, implementing a core set of commands that allow the head and host to interact when necessary.

Since the TH1.x head is an older product, it is unable to take advantage of much of the modular functionality contained in CEH. To compensate for the older head technology, CEH simulates a standard TH1.x interface when used with a TH1.x head. This means that most of the core commands and functionality of the head appear to be very similar to those contained in the older thermal module, although significant differences exist in the underlying platform.

5. MPE/HEAD COMPATIBILITY

The thermal module and thermal head firmware is developed simultaneously, because much of the head functionality depends on the thermal module. Therefore, different versions of the thermal module access different functionality in the head firmware. This is not to say that you must use only these versions of head and MPE firmware together on a certain device. In fact, some devices may not have any release versions of MPE firmware containing an up-to-date thermal module. Rather, this means that the changes implemented in head firmware won't become visible unless the MPE firmware contains the correlated thermal module.

Significant amounts of new functionality are added in each version of head firmware. Figure 2, on the next page, summarizes this new functionality, starting at version 1.56, along with the minimum version of the thermal module required to access that new functionality. **If either the head or MPE firmware fail to meet the minimum version requirements, then you will be unable to access new functionality**.

Note: Version 1.9x is not yet released, and is included here only for future reference.

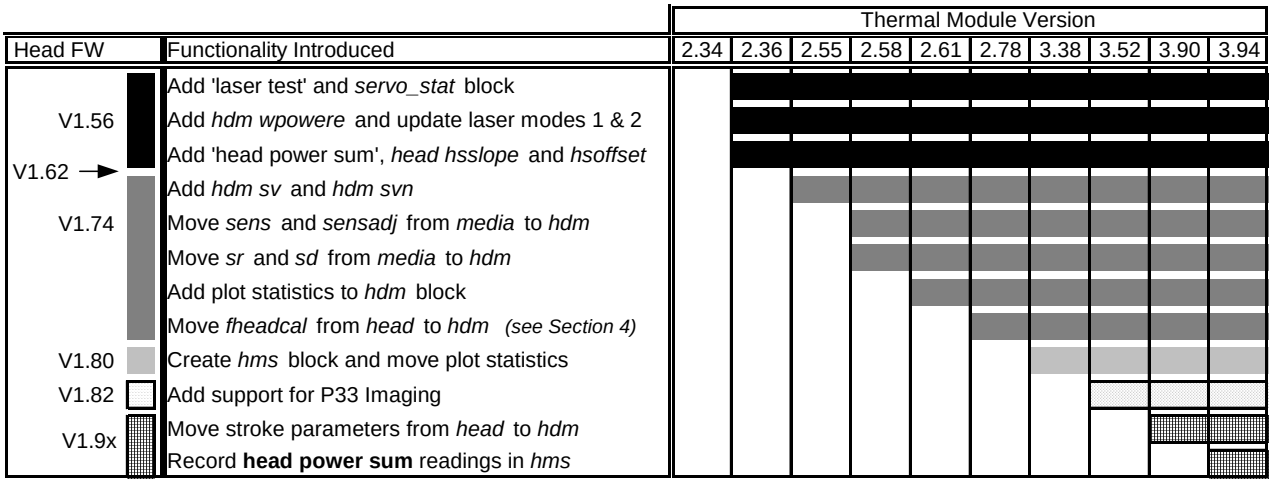


Figure 2: **Functionality introduced in thermal module**

5.1 *fheadcal* Parameter Location in the MPE

The *fheadcal* parameter was moved from the *head* block to the *hdm* block in version 2.78 of the thermal module and version 1.74 of the thermal head firmware, as shown in Figure 2. Hence, the active value of *fheadcal*, which is used when imaging, will differ depending on the firmware you are using. Table 1 summarizes the active *fheadcal* location for all possible firmware combinations.

Table 1: **Active location of** `fheadcal`

	Thermal module < 2.78	**Thermal module ≥ 2.78**
Head firmware ≤ V1.62	*`head fheadcal`*	*`head r9`*
Head firmware ≥ V1.74	*`head fheadcal`*	*`hdm <index> fheadcal`*

When using thermal module version 2.78 or higher with head firmware version 1.62 or older, type **`>> set head r9 <value>`** to set *fheadcal* manually. **Changing the value stored in *hdm*** will not have the desired effect, and **will cause name corruption.** In this case, when you **type `>> set hdm <index>`,** you will not be able to see *fheadcal*. However, you will be able to see *fheadcal* by typing **`>> set hdm <index> fheadcal`**. Remember, though, that unless you have thermal head firmware version 1.74 or higher, you must not attempt to access *fheadcal* in the *hdm* block.

6. HEAD NVS CONFIGURATION

The thermal head contains copies of all head NVS parameters. As of head firmware version 1.83, the *head*, *hdp*, *hdm*, *hms*, and *servo_stat* blocks comprise the head NVS. Here are some notes on the parameter blocks and their usage:

- The *head* parameters are used mostly by the thermal head.
- The *hdp* parameters are used only by the host (no exceptions). As the name head-dependent parameters (*hdp*) implies, the *hdp* parameters are dependent on the thermal head and are thus saved with the head.
- The head's *hdm* and host's *media* parameters are closely related. As the name head-dependent media (*hdm*) implies, the *hdm* parameters are dependent on the thermal head and are thus saved with the head. The media parameters are dependent on the media and are thus saved with the host.

Other parameters are stored in the head NVS as well. These parameters include the STROKE and SCORR tables, the head state parameters, and the head status (or mask) parameters.

6.1 Handling Head NVS Parameters in the MPE

The *head*, *hdm*, *hdp*, *servo_stat*, and *hms* parameter blocks have parallel structures in the MPE and head. This permits easier back-up, restoration, and updating of the parameter blocks. By contrast, the STROKE and SCORR tables, the head state parameters, and the head status (or mask) parameters do not have MPE copies, and are therefore handled differently. In particular, these parameters are more difficult to back-up and restore.

Each head NVS block has an 'active' copy, stored in RAM, and a 'saved' copy, stored in the actual NVS. Those parameter blocks with parallel structures in the MPE also have 'active' and 'saved' copies in the MPE. The head NVS configuration when using an MPE platform is shown in Figure 3.

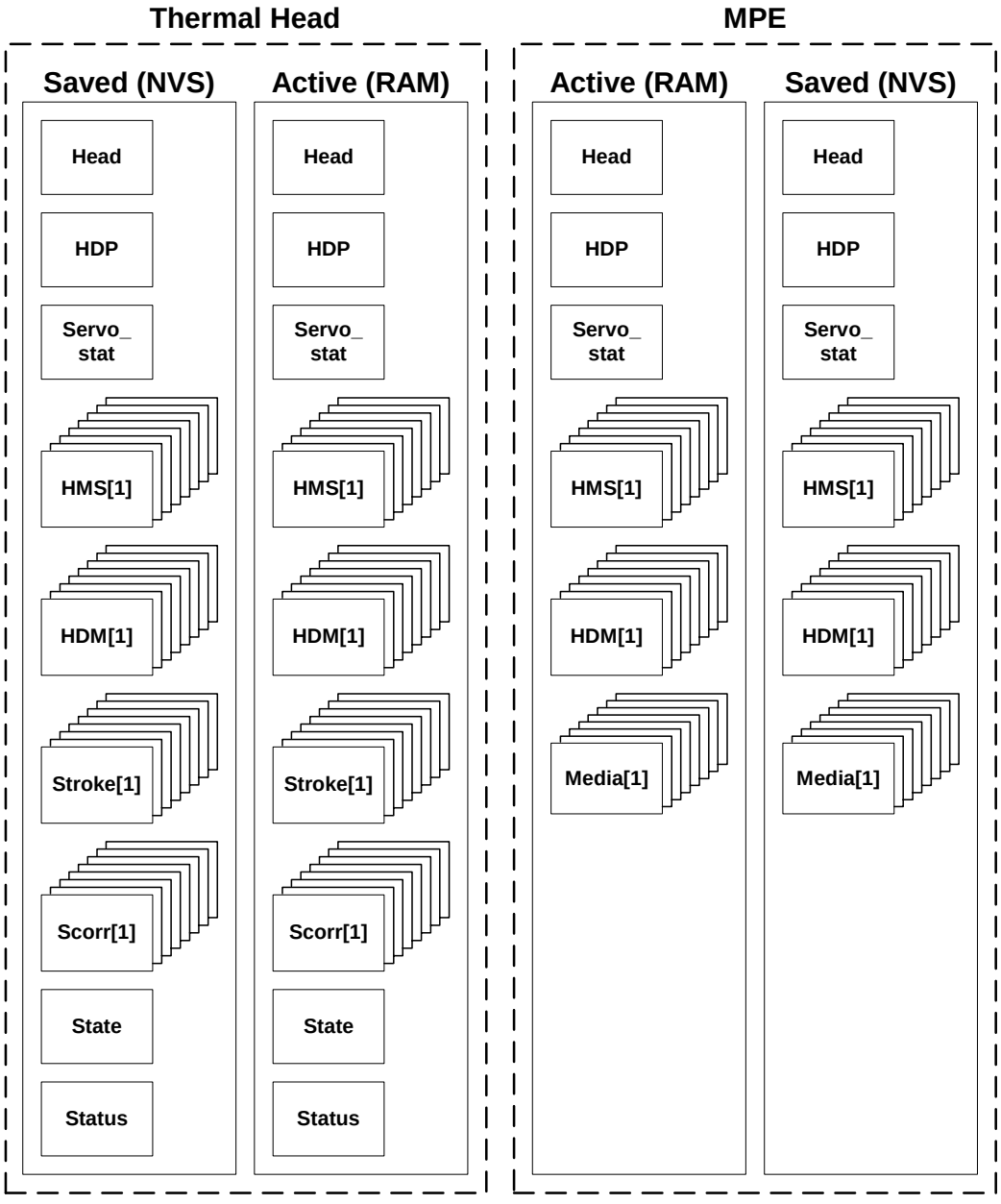


Figure 3: **Head/MPE NVS configuration**

6.1.1 Handling Generic Parameters in the MPE

When you type **`>> set <sub-system>`**, you will see three values displayed for each parameter: active, saved, and default. The default value is hard-coded, and does not play an active part at run-time. The **active and saved values displayed on the terminal are the active and saved values in the MPE**. This means that **the active and saved values in the head are not visible**, although they are used inside the head. Figure 4 shows the relationship between the head and MPE NVS sub-systems, as well as the data travel that occurs during execution of various commands.

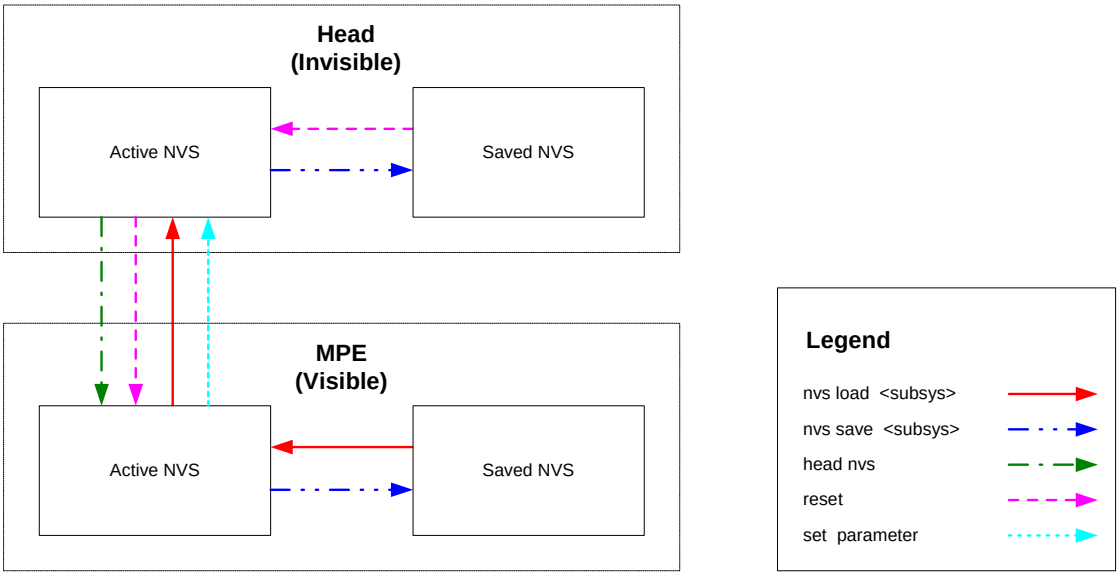


Figure 4: **Head and MPE NVS relationship**

6.2 Handling Head NVS Parameters in the MCE

Unlike the MPE, the MCE does not have a parallel NVS structure for the *head*, *hdp*, *hdm*, *servo_stat*, and *hms* parameter blocks. The values are stored in the head only, and no saved copy exists in the MCE. The head NVS configuration when using an MCE platform is shown in Figure 5, on the next page.

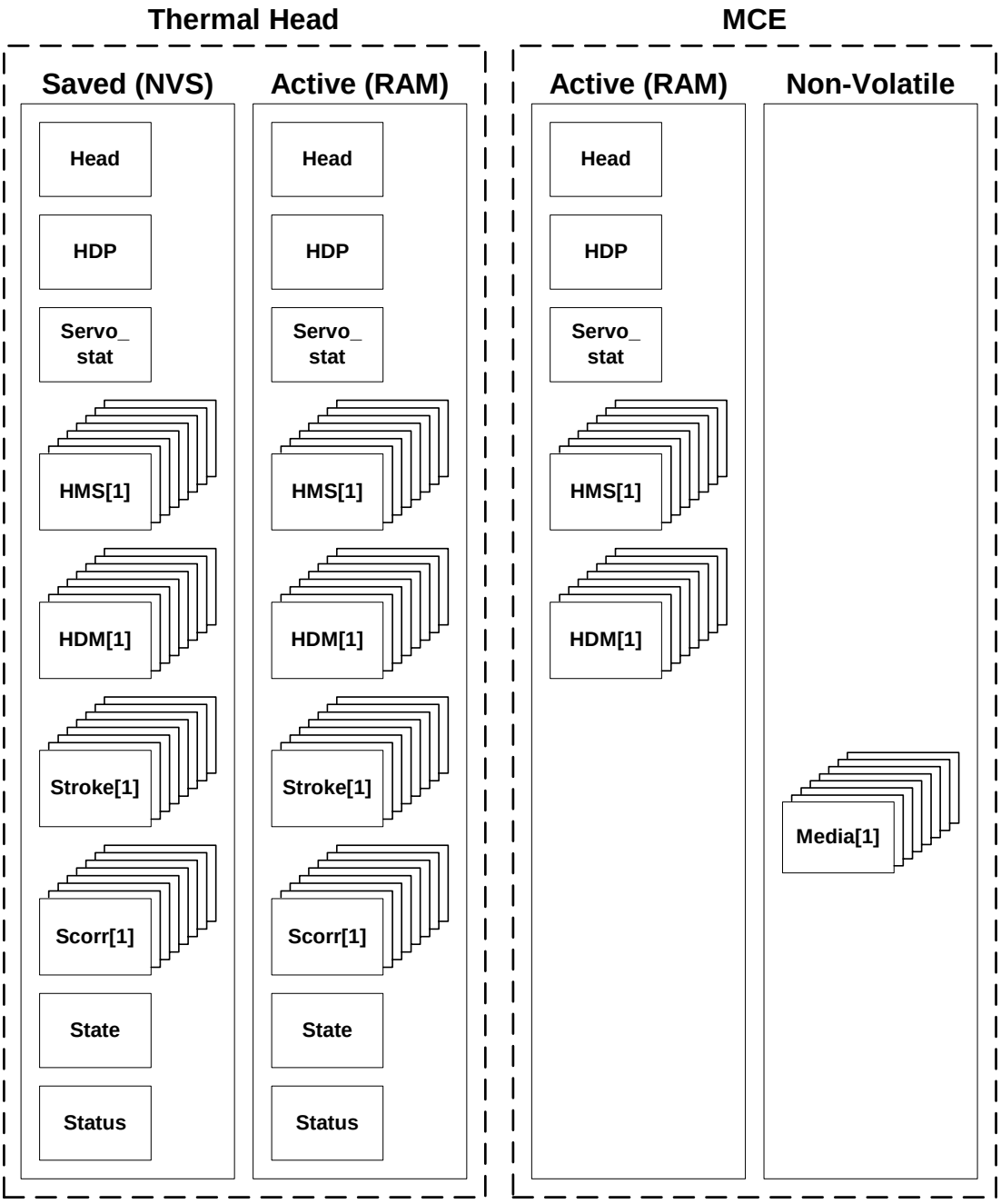


Figure 5: **Head/MCE NVS configuration**

6.2.1 Handling Generic Parameters in the MCE

The MCE parameter-handling differs significantly from the MPE parameter-handling. The first and most significant difference is that **all parameters are automatically saved when they are entered**. Head parameters are saved in the head, and MCE parameters are saved in the MCE. What this means is that there is no distinction between ‘active’ and ‘saved’ values for a parameter.

For host parameters, a single, non-volatile copy of each parameter exists in the MCE. This is why Figure 5 shows only one, non-volatile copy of the *media* sub-system.

As in the MPE, the **active value displayed on the terminal is the active value in the MCE**. This means that **the active and saved values in the head are not visible**, although they are used inside the head. Figure 6 shows the relationship between the head and MPE NVS sub-systems, as well as the data travel that occurs during execution of various commands.

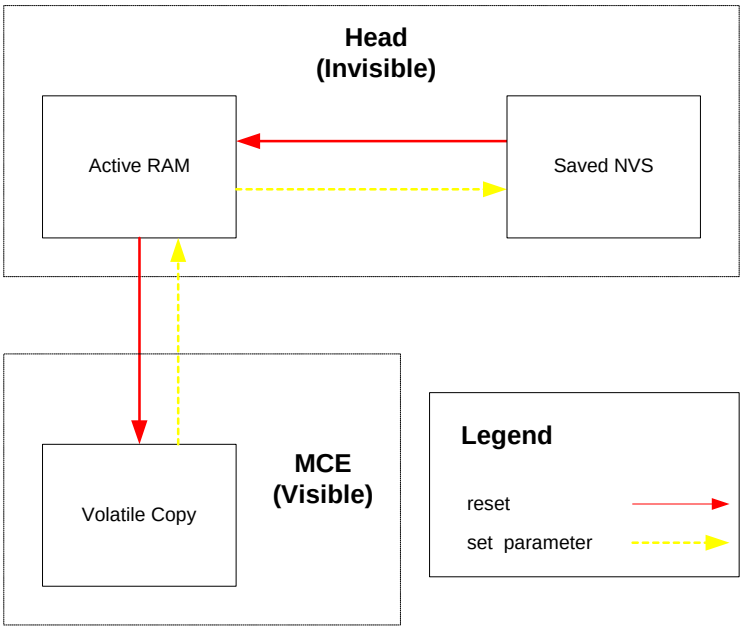


Figure 6: **Head and MCE NVS relationship**

As Figure 6 shows, the head parameters are loaded from the head to the MCE on start-up. Because no copy of the head parameters exists on the MCE, these parameters may not be viewed or changed if the head is offline. The head parameters will simply become invisible unless a head is attached and online with the MCE device.

To account for the situation when a head is offline because it has no firmware loaded, there is one head-dependent parameter stored in non-volatile memory in the MCE. The *hmgr type* parameter stores the head type, and is updated from the *head type* parameter under normal operation.

6.3 Handling STROKE and SCORR Tables

The active SCORR and STROKE tables correspond to the active media. The host sets the active media in the head. When setting the active media, the host performs the following steps:

- The *media* index is determined by input parameters from the user or workstation.
- The *hdm* index is determined by matching the *name* and (optionally) *color* fields of the selected media block with the *name* and *color* fields in the hdm block. The *hdm* index may differ from the *media* index.
- The STROKE and SCORR table index is identical to the *hdm* index.
- The *hms* index is identical to the *hdm* index.
- The thermal head provides up to sixteen copies of the *hdm*, *hms*, STROKE, and SCORR parameters. However, some hosts only access eight of the copies.

Other useful commands include:

- **(stroke|scorr)**: display the current STROKE or SCORR table
- **(stroke|scorr) <index>**: display the appropriate value in the current STROKE or SCORR table
- **(stroke|scorr) <index> <value>**: modify the appropriate value in the current STROKE or SCORR table
- **stroke init <value>**: initialize the whole STROKE table to **<value>** (default is 33 if no value specified)
- **scorr init <value>**: initialize the whole SCORR table to **<value>** (default is 0 if no value specified)
- **(stroke|scorr) save**: save all STROKE or SCORR tables (not just the active table)

6.4 Handling the STATUS Sub-System

The STATUS sub-system contains information on the status bits available on the thermal head. These status bits are commonly referred to as the 'head mask'. To view all status bits on the thermal head, type **>> head mask**. A '!' character is used to identify status bits that are alarmed, and a '*' character is used to indicate status bits that are masked. Because some status bits are translated to generic error messages, the **head mask** command can help to pinpoint more specific error conditions.

To mask or unmask status bits, type **>> head mask (on|off) <number1> <number2>** … You can mask or unmask multiple status bits at one time. To save the current mask configuration, type **>> head mask save**.

7. MPE NVS SYNCHRONIZATION AND UPGRADE

When upgrading head firmware, or installing a new head, it is crucial to initialize and synchronize the head NVS. Since the head NVS parameters control some of the most basic head functionality, a device may fail to image correctly if the correct procedure is not followed. In this section we address two commands that have caused confusion in the past, and explain when and why they are used.

7.1 MPE 'nvs save all creo' Command

The >> **nvs save all creo** command works in much the same way as the >> **nvs save <subsys>** command shown in Figure 4. The only distinction is that >> **nvs save <subsys>** saves the parameters stored in the specified sub-system only, whereas >> **nvs save all creo** saves every parameter in the entire NVS system. Typically, when saving parameters in only one or two sub-systems, you use >> **nvs save <subsys>**, but if you need to save parameters in a larger number of sub-systems, you use >> **nvs save all creo**.

When a new head is installed on a device, the reset functionality shown in Figure 4 is invoked. During reset, the 'saved' values in the head are copied to the 'active' slots in the head and MPE, but not to the 'saved' slot in the MPE. We consider the head properly synchronized when all of the 'active' and 'saved' copies of every variable are the same. In order to synchronize the head and MPE NVS, **you must type >> nvs save all creo.**

7.2 MPE 'nvs upgrade head <previous version>' Command

The >> **nvs upgrade head <previous version>** command implements changes in the head NVS block for each version of head firmware. This command is contained in the thermal module; hence, it only implements NVS changes contained in that version of the thermal module. So, if you **type** >> **nvs upgrade head 1.62** on a device with thermal module 2.61, *sv, svn, sens, sensadj, sr, sd,* and the plot statistics will be set up in the *hdm* block, but *fheadcal* will not. Refer to Figure 2 for a summary of the NVS changes contained in each version of the thermal module and implemented by >> **nvs upgrade head <previous version>**.

You should use the >> **nvs upgrade head <previous version>** command in two situations. These are:

1. You have just upgraded the head firmware.
2. You have just upgraded the MPE firmware to a version containing thermal module 2.78 or higher, under both of these conditions:
    a. The head firmware was upgraded to version 1.74 after the head was installed on this device, but before the MPE was upgraded. Heads installed with version 1.74 head firmware do not need to have their NVS upgraded.
    b. The previous MPE firmware contained a thermal module prior to 2.78.

    In this case, use '1.62' as the previous version number when running >> **nvs upgrade head <previous version>**.

You do not need to run >> **`nvs upgrade head <previous version>`** on a head that was integrated using version 1.74 firmware. After upgrading the MPE and/or installing a new head, you can tell if the head is configured correctly for version 1.74 firmware by typing >> **`set hdm <index>`**. Check the *sr*, *sd*, and *fheadcal* parameters—if these parameters are within expected operating ranges, then the head is configured correctly. If one or more of these parameters are incorrect, you will need to run >> **`nvs upgrade head <previous version>`**, as described above.

Figure 7 shows the >> **`nvs upgrade head <previous version>`** command's functionality.

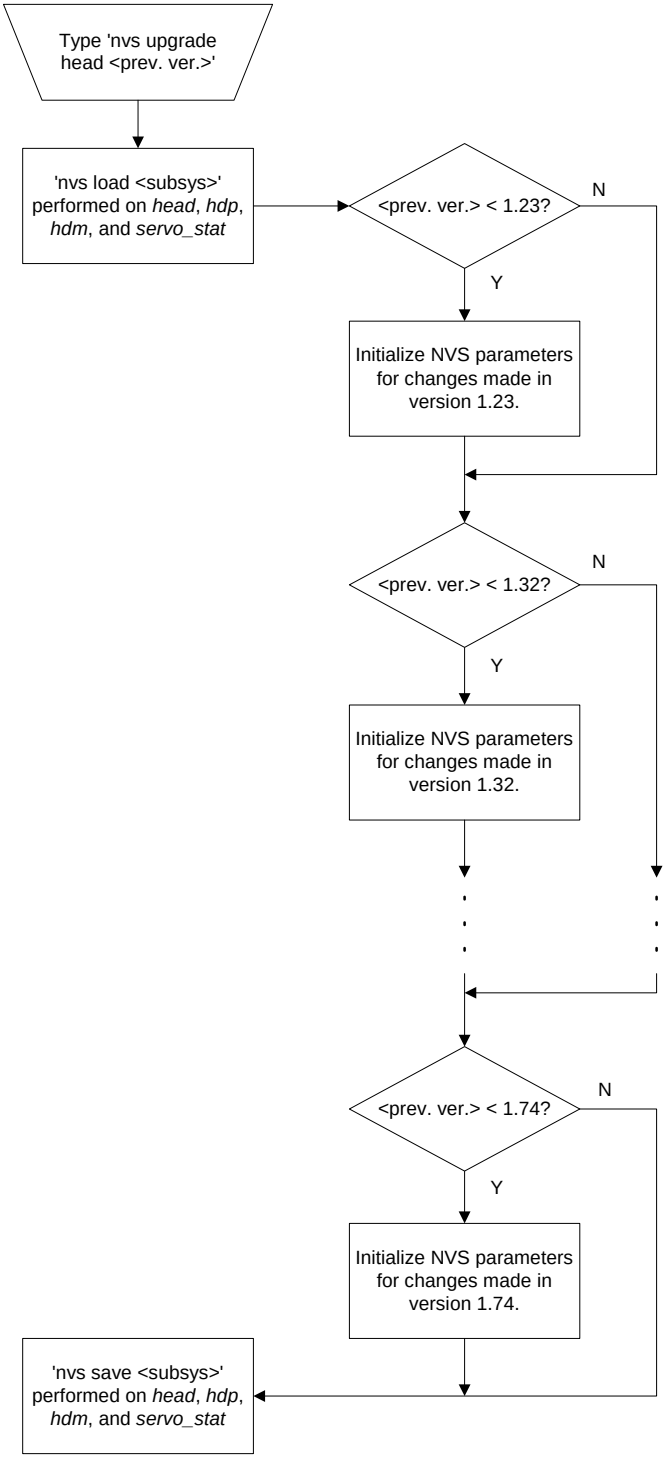


Figure 7: **'nvs upgrade head' functionality**

8. MCE NVS SYNCHRONIZATION AND UPGRADE

When upgrading head firmware, or installing a new head, it is necessary to initialize and synchronize the head NVS. Since the head NVS parameters control some of the most basic head functionality, a device may fail to image correctly if the correct procedure is not followed. The synchronization and upgrade procedures differ significantly between the MCE and MPE, because of the significant differences between head parameter handling on the two platforms.

Whenever a head is installed on an MCE device, all head parameters in the host will be automatically synchronized. Because there is no distinction between active and saved copies, it is not necessary or possible to explicitly save parameters. This is the first significant difference between the MCE and MPE.

In the MPE firmware, there is a command to automatically upgrade NVS when new firmware is loaded. This will automatically populate any new parameters, copy values from one location to another, and save all of the changes. No similar functionality exists in the MCE. In the future, as needs arise, we will develop upgrade scripts to perform any necessary NVS upgrades. Currently, no head NVS upgrades are required in the MCE, because the TH1.x head is so new to this MCE platform. There are no NVS changes between the two versions of head firmware supported by the MCE.

9. MPE PROBLEMS AND SOLUTIONS

Some problems in handling *hdm* parameters have arisen on the MPE platform. In particular, many problems arose with the release of the version 1.74 head firmware. It is important to remember that each upgrade problem is distinct, and must be handled differently.

9.1 hdm Name Corruption in the MPE

When operating a device with thermal module version 2.61 to 3.30, with head firmware version 1.62 or older, the *hdm name* parameter sometimes becomes corrupted. A single changed character indicates name corruption. In order to permanently eliminate this problem, head firmware must be upgraded to version 1.74 using the standard upgrade procedure. Then, once the upgrade is complete, re-enter the correct media name by typing >> **`set hdm <index> name "<name>"`** followed by >> **`nvs save hdm`**.

Name corruption occurs during retrace, when we increment and save the *hdm retrace* parameter. When head firmware version 1.62 or older is being used, no space is allocated in the head NVS for the *retrace* parameter, since it was introduced in head firmware version 1.74 (see Figure 2). When the active value is written to the saved value in the head, the head doesn't know where this new *retrace* value should go, so it writes it to the next logical address—the next *hdm* block's *name*—causing corruption.

In order to prevent name corruption from occurring, we added a version check in thermal module version 3.32. The version check confirms that we are using head firmware version 1.74 or higher before updating and saving the *retrace* parameter.

9.2 Temporary Changes to hdm Block Saved in MPE

When operating a device with thermal module version 2.61 to 3.32, with head firmware version 1.74, all of the active values in the *hdm* block are saved during a plot. In this case, only the head NVS 'saved' value is updated, and the MPE 'saved' value is left untouched. So, on reset, it will appear as if the 'active' and 'saved' values disagree. To prevent imaging problems, manually set any hdm changes back to their original values, and then type >> **`nvs save hdm`** before leaving a device.

The NVS saving occurs during plotting, when we increment and save the plot statistics in the *hdm* block. We saved the statistics with every plot to ensure that we would be able to retain the data, for informational and debugging purposes. The problem is that the head saves the whole *hdm* block at the same time (we cannot save individual parameters within the *hdm* block). So, when we saved the plot statistics, we inadvertently saved all of the other *hdm* values to the head at the same time. From Figure 4, you can see why the 'active' and 'saved'

values appear to disagree—the head's 'saved' values are invisible, even though they are copied to the MPE's 'active' slot on reset.

In order to fix this problem, we stopped saving the plot statistics altogether in thermal module 3.34. Then, in thermal module 3.38, we added the new *hms* parameter block to store plot statistics by media, removing the plot statistics from the *hdm* block. This allows us to save and track the plot statistics over time, without risking problems in the *hdm* block. The *hms* block will become available following the next release of head firmware.

9.3 Firmware Upgrade and Head Installation Problems on MPE

Unfortunately, device problems frequently occur following firmware upgrades or after installing heads with new firmware versions. In many of these cases, NVS issues result when new parameters are improperly initialized and synchronized. The firmware upgrade procedures are well documented in the firmware release notes for both the MPE and the thermal head, and should be followed closely in all cases. However, it is important to underscore two points:

1. The head firmware and thermal module must remain compatible at all times.
2. When both MPE and thermal head firmware are being upgraded, the MPE firmware must be upgraded first.

In order to ensure that an upgrade is successful, be sure to refer to the firmware release notes to determine how to proceed. You may need to sequentially upgrade both MPE and head firmware in order to bring the MPE firmware to the desired level. At each step it is important to ensure that the device is configured correctly, upgrading the NVS as needed.

When installing a head, it is very important that any necessary MPE upgrades are performed before the head is installed on the device. Following the procedure outlined in the firmware release notes, perform whatever upgrades are necessary to install the appropriate MPE firmware, and ensure that the NVS is configured properly. If the head was properly configured in the factory, and the MPE was upgraded correctly, you should be able to plug it in and have the device plot with no problems.

10. MEASURING HEAD POWER

10.1 Head Power Full

Prior to head firmware version 1.56, the preferred measurement for head output power was the **`head power full`** command. The values from **`head power full`** were used to calculate the required exposure when setting up the reading. However, due to the nature of the output vs. measuring curve, this method can be quite inaccurate (up to ±20% or more) and inadequate for exposure-setting requirements, although still preferable to using the *wpower* setting. Since **`head power full`** was inadequate, a new method, the **`head power sum`** reading, was incorporated into head firmware version 1.56.

10.2 Head Power Sum

If you are dealing with any thermal head using version 1.56 head firmware, measure the output laser power of the head by typing **`>> head power sum`**. The returned result will be two values: laser power in watts and laser leakage in watts. These values are the most accurate to help you calculate the exposure, and the measured values should be directly input in the exposure calculating formula.

Note: In order to ensure the accuracy of the **`head power sum`** reading, the *head bpgain, hsslope,* and *hsoffset* parameters must be set up properly. In particular, any time that *bpgain* is changed, *hsslope* and *hsoffset* must be recalibrated. Failure to do so could result in false power readings from the head.

11. THERMAL HEAD VERSION INFORMATION

The following table cross-references the Thermal Head Firmware Version with the MPE Thermal Module required. Note that Thermal Head Firmware Version entries that are **bold** are the supported releases. Also note that all released MCE firmware will function with head firmware V1.80 or higher, and no minimum MCE requirements exist. Information about which versions of head firmware are compatible with device control (MPE or MCE) firmware can be found on the Controlled Release Website. Thermal head firmware can be found on the Thermal Head 1 Firmware Release Summary site.

Thermal Head Firmware Version	MPE Thermal Module	Notes
1.00	1.56	
1.03	1.62, 2.00	
1.05	1.62, 2.00	
1.11	2.06, 2.07, 2.08, 2.09	Beta release of Thermal Head firmware version 1.12.
1.12	2.06, 2.07, 2.08, 2.09	Adds support for more than one media type.
1.13	2.06, 2.07, 2.08, 2.09	Corrects bugs in Thermal Head firmware version 1.12.
1.14	2.06, 2.07, 2.08, 2.09	Corrects bugs in Thermal head firmware version 1.13.
1.15	2.06, 2.07, 2.08, 2.09	Corrects bugs in Thermal Head firmware version 1.14.
1.16	2.06[1], 2.07, 2.08, 2.09, 2.10	Adds support for the 10-0892C version of the Thermal Head Controller.
1.17	2.06[2], 2.07, 2.08, 2.09, 2.10	Beta release of Thermal Head firmware version 1.16 that supports 40W Thermal Heads.
1.18	2.06[3], 2.07, 2.08, 2.09, 2.10	Corrects bugs in Thermal Head firmware version 1.16.
1.18b	2.06[4], 2.07, 2.08, 2.09, 2.10	Corrects bugs in Thermal Head firmware version 1.18.
1.19	2.06[5], 2.07, 2.08, 2.09, 2.10	Beta release of Thermal Head firmware version 1.18 that supports 40W Thermal Heads.
1.20	2.23	Beta release of Thermal Head firmware version 1.21.
1.21	2.23	Adds support for Staccato and Bipolar.
1.22	2.23	Corrects bugs in Thermal Head firmware version 1.21.
1.23	2.23	Corrects bugs in Thermal Head firmware version 1.22.
1.24	2.30	Beta release of Thermal Head firmware version 1.30.
1.30	2.30	Adds support for the active chiller and laser standby.
1.31	2.30	Beta release of Thermal Head firmware version 1.34.
1.32	2.30	Corrects bugs in Thermal Head firmware version 1.30.
1.32.6	2.30	Corrects a hardware problem that resulted in the error message “Timeout waiting for LASER OK”.
1.33	2.34	Beta release of Thermal Head firmware version 1.34.
1.34	2.34	Adds support for laser mode 3 and enhanced safety loop handling.

[1]Thermal Module 2.06 only supports the 10-0892B version of the Thermal Head Controller.
[2] See footnote 1.
[3] See footnote 1.
[4] See footnote 1.
[5] See footnote 1.

1.35	2.34	Beta release of Thermal Head firmware version 1.36.
1.36	2.34	Corrects bugs in Thermal Head firmware version 1.34.
1.37	2.34	Development work only.
1.38	2.34	Corrects a hardware problem that resulted in the error message “Timeout waiting for LASER OK”.
1.56	2.34, but higher versions preferred	Improved diagnostics, focus and bug fixes
1.62	2.34, but higher versions preferred	Bug fixes and added functionality to allow shutter sensor calibration in the field.
1.73	2.34 but higher versions strongly recommended	Improved diagnostics, code optimization, reorganized NVS, added plot statistic storage.
1.74	2.34 but higher versions strongly recommended	Fixed head power pulse and stroke profile bugs in version 1.73.
1.80	2.36 but higher versions strongly recommended	Support Entry Imaging heads, move media statistics collection to *hms* sub-system.
1.82	2.36 but higher versions strongly recommended	Support revised P33 Imaging plan.
1.83	2.36 but higher versions strongly recommended	Fix bug that resulted in rare, intermittent NVS corruption.