Genera 8.3 Release Notes

Genera 8.3 Introduction and Highlights

Genera 8.3 is a minor release built upon Genera 8.1 and ECO's to Genera 8.1. Genera 8.3 includes not only all general bug fixes since Genera 8.1, but also the results of a special bug-fixing task force assigned to address customer-reported issues. CLIM 2.0, a new major release of CLIM, is also included.

Genera 8.3 supports the following Symbolics platforms and products:

- NXP1000, 3600, XL, MacIvory, and UX families

- for 3600 family and development Ivory machines — Color, Concordia, DNA, Joshua, KEE 3.1, and Statice.

- for delivery MacIvory machines — Genera 8.3 Delivery

Genera 8.3 provides the first Symbolics release of the draft proposed ANSI Common Lisp (referred to as *dpANS Common Lisp* in this document.)

Genera 8.3 supports all currently supported Sun and Macintosh system software versions, and adds support for the Macintosh Quadra 950, Apple System Software Version 7.1, and SunOS 4.1.2.

Genera 8.3 is distributed only on CD-ROM for Ivory-based platforms (NXP, XL, UX, and MacIvory); QIC-11 tapes will continue to be a distribution option for 3600-family machines.

Genera 8.3 Statice Runtime is now a CLOS-based implementation.

Genera 8.3 is the last release for the 3600-family machines.

Changes to Lisp in Genera 8.3

- The function **map-into** has been updated to conform to dpANS Common Lisp with respect to its treatment of the fill pointer of the target sequence. If the target sequence has a fill pointer, **map-into** now updates the fill pointer of the target sequence to be the number of times that the user function was invoked; in prior releases, **map-into** did not modify the value of the fill pointer.

- The bug that caused your machine to crash into the FEP if you created a zero length adjustable array and extended it using **vector-push-extend** has been fixed.

- In Genera 8.1, the dpANSI Common Lisp **loop** construct "for ... by" was incompletely implemented so that a form like

      (loop for x by 6 do (print x))

  did not work. This bug has been fixed.

- A bug in the table system where a table fills up with vacated slots without being rehashed, resulting in the error message `Too many collisions, can't happen`, has been fixed.

- In a previous release, using **format** with case-changing operations like `~(...~)` would suppress the mouse-sensitivity of items that were presented inside the case-changed region. In this release, this sensitivity is not suppressed.

- A minor bug in the interface between the Flavors system and the debugger has been fixed. When a generic function uses **:case** method-combination, and is called with an unrecognized subcase keyword, the debugger now correctly implements the proceed-option which offers to replace the unrecognized keyword with a correct one. Formerly, if the user selected this proceed option, a further error would result.

- A long-standing bug in which **:handle-condition-flavors-p** for condition flavors that were not yet "composed'" has been fixed. One implication of this is that **dbg:invoke-restart-handlers** used to ignore some possible options which it will now correctly detect.

- Using **clos:make-instance** in a **stack-let** form now produces a stack-allocated CLOS instance, just as using **make-instance** has always done for Flavors instances.

- In a previous release, tracing a CLOS method, for example by doing

      (TRACE (:FUNCTION (CLOS:METHOD *method signature*))),

  and then later trying to recompile the method would result in an error. This problem has been fixed.

- CLOS was not optimizing **clos:slot-value** on required arguments which were not used in specialization. For example:

      (defmethod m1 ((x c1) y)
            (slot-value y 's1))

  In Genera 8.3 the dispatch mechanism is used on arguments which are used with **clos:slot-value** as well as those which are used for dispatch.

- In Genera 8.1, the command Show Class Initargs did not show inherited initargs if a class had not yet been instantiated. This has been fixed.

- A bug in **flet** and **labels**, such that a macro named *f* from an outer environment was still visible while parsing declarations in the **flet** (or **labels**) body even when *f* was shadowed by the **flet** (or **labels**), has been fixed.

- The Common Lisp input notations `#1a"foo"` and `#1a#*10101` now work. **make-synonym-stream** now returns an object of type **future-common-lisp:synonym-stream** rather than a symbol.

- **flavor:remove-flavor** now correctly updates the CLOS data structures.

- Built-in classes have been expanded to work as types in **subtypep**. For example:
  ```
  (subtypep 'integer (find-class t)) => t
  ```

- The functions **cltl:function** and **cltl:apply** no longer improperly complain of an invalid function when supplied a symbol that is not defined as a function. Instead, they signal an undefined function error.

- In previous releases, the Genera compiler did not preserve **eq**-ness of constants, as required by dpANS Common Lisp. The **compile** function now properly preserves the identity of constants in the compiled code.

Changes to Zetalisp to Common Lisp Conversion Tools

- **zl:*throw**, **zl:pop**, **zl:with-input-from-string**, **zl:y-or-n-p**, and **zl:yes-or-no-p** are now converted in a way that better preserves their original semantics.

- The converters for **zl:get**, **zl:listp**, and **zl:nlistp** offer the same conversions as before, but in the opposite order, so that the semantics-preserving transformation is the default.

- The converter for **zl:check-arg** is now smarter about type predicates. For example, (ZL:CHECK-ARG X SYMBOLP "a symbol") and (ZL:CHECK-ARG X (ZL:SYMBOLP X) "a symbol") are now both converted to (CHECK-TYPE X SYMBOL).

- A bug in the conversion of string comparison operators in which functions erroneously used the the **:end1** keyword instead of being converted to use the **:end2** keyword has been fixed. The affected operators are:

zl:string-equal	**zl:string=**	**zl:string-compare**
zl:string-greaterp	**zl:string>**	**zl:string-exact-compare**
zl:string-lessp	**zl:string<**	
zl:string-not-equal	**zl:string≠**	
zl:string-not-greaterp	**zl:string≤**	
zl:string-not-lessp	**zl:string≥**	

Changes to the Command Processor in Genera 8.3

- The Expunge Directory CP command has the following four new keyword options:

  :More Processing
  {Default, Yes, No} Controls whether ****More**** processing at end of page is en-

abled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Enables you to direct your output. The default is the stream **standard-output***. Note that redirecting output to a printer can be particularly useful.

:Notify
{Yes, No, Query} Controls whether other users at the site are notified when multiple top-level directories are to be expunged by wildcard reference, for example, >*> or >**>. When such wildcards are not used (or when such wildcards are only used to specify wild sub-directories of one or more specifically named top-level directories, for example >Harry>*> or >Sally>**>), this option has no effect. By default the variable **si:*expunge-notification-default*** is set to :No.

:Delay After Notification
{a time interval} When notifications are sent to other users, controls the length of time between notification and expunge. During the delay, the process can be aborted with Control-Abort, which both cancels the expunge and sends a notification that the expunge has been cancelled. By default, the variable **si:*expunge-notification-delay-default*** is :No.

When Expunge Directory (m-X) is used to expunge multiple top-level directories by wildcard reference (such as >*> or >**>), the variable **si:*expunge-notification-default*** .controls whether other users at the site will be notified first. If a numeric argument is provided, the value of this variable is locally forced to be **:query**. By default, the variable **si:*expunge-notification-default*** is :No.

- The following variables are new:

si:*expunge-notification-default*   *Variable*

Controls the default of the :Notify keyword for the Expunge Directory CP command.

si:*expunge-notification-delay-default*   *Variable*

Controls the default of the :Delay After Notification keyword for the Expunge Directory CP command.

- The :Output Destination keyword can now also be used to discard the output of a command, by specifying "None" as the destination. This is useful when you want a command to perform its side-effects, but you are not interested in the command's typeout.

- The CP command Find Symbol has a new keyword option, :Verbose. If :Verbose is Yes, the names of all packages searched are printed, whether matching symbols are found in them or not. If verbose is No, only the names of those packages where matching symbols are found are printed. The default is No, and the mentioned default is Yes.

- The CP command Delete Directory has been enhanced to provide some additional error checking and better display of information about what action it will take. Because of its general nature, and in spite of these improvements, this command is still quite powerful and should continue to be used with great care.

- The Show Machine Configuration command has been changed to report the actual amount of physical memory on the MacIvory model 3. Previously, the command excluded those portions of physical memory reserved for the Boot ROM and communications area when it computed the size of memory on a model 3.

- The Copy World command now works over remote terminals. Using the keyword :Show Blocks Copied (which prints the number of blocks finished copying, every 100 blocks) is useful when using Copy World from a remote terminal.

- The Compare Directories command has been given a new keyword argument, :Ignore Versions:

:Ignore Versions   {Yes or No} The default is No. If Yes, then consider files with the same name and type to be the same even if they have different version numbers.

Changes to Zmail in Genera 8.3

- In a previous release, Zmail would occasionally try to expose one of its panes (usually the keyword pane) outside of the superior; if aborted, Zmail would never again expose the offending pane, leaving the filter menu in an unusable state. This bug has been fixed.

- The CP command Edit File now can be used to edit a file in Zmail by specifying the keyword argument :Editor Zmail (This keyword value is the default when the file to edit is a KBIN file.) From FSEdit, if you click right on a KBIN file and then click on Edit File, you can edit the file in Zmail. Similarly, if you click Meta-Mouse-Left on a KBIN file in a directory listing, you can use Zmail as the editor.

- The Zmail command Save Draft As Message (m-X) has been extended to accept numeric arguments. To specify a specific buffer, specify a numeric argument of 2. If the message has already been saved, Zmail does not resave it unless you specify a numeric argument of 4 (or c-U). The arguments are actually dealt with bit-wise, so an argument of 6 has the combined effect of an argument of 2 and an argument of 4. Save Draft As Message (m-X) now prefers the current sequence over the default sequence if the current sequence happens to be a buffer.

- The bug in the Decode ECO (m-X) Zmail command which caused it to erroneously complain about an incorrect end-to-end checksum when decoding an ECO whose content is a 16-bit wide binary file has been fixed.

- When you store your mail files on a UNIX host, Zmail is now careful to create a new version of the file, retaining a backup copy of the previous contents.

- The Zmail command Merge Keywords In Conversation (m-X) has been changed slightly to provide more flexibility in the case of the pop-up menu. When there is a conflict in the keywords, you are now asked not only which keywords to add and remove, but also whether you want to apply these changes to just the current message, to just unlabeled messages in the conversation, or to all messages in the conversation.

Changes to Zwei and Zmacs in Genera 8.3

- A long-standing bug in Zwei's mouse-sensitivity was fixed: If a multi-line expression is modified and the mouse is used to select the object represented by that expression from a line other than one which was modified, the object selected will now reflect what is currently in the buffer (rather than using stale information about what used to be in the buffer).

- Some Zwei pop-up windows that prompt the user for input were incorrectly exposed on the main console, rather than the console the user was sitting at. This typically happened when using the machine via X-Windows. Though we do not know if the problem is solved in the general case, one common source of this problem has been fixed.

- Some bugs in the self-documentation for the prefix commands Prefix Control (c-m-X), Prefix Meta (c-m-X), and Prefix Control Meta (c-m-X) that sometimes caused Zmacs to enter the debugger are fixed.

- When you try to select a file for editing using Find File (c-X c-F) but already have a version of the file in a buffer that is older than the newest version in the file system, you get a query about whether to revert the buffer to pick up the most recent version. This query now includes a "Compare" option. If you se-

lect this option, the buffer copy will be compared to the copy in the file system
and then you will be re-prompted about whether to revert the buffer.

- When you use ᴍ-ᴄ Finish Patch in Zmacs, if you have specified system depen-
  dencies for the patch but are not up to the proper patch level in those systems,
  you will now be queried interactively about whether you want to load patches
  rather than going straight into the debugger.

- A bug in Genera 8.2 (only) has been fixed in which ᴍ-ᴄ Add Patch (and related
  Zwei commands) would enter the debugger if the region being patched had un-
  balanced parens or contained no expressions at all.

- The patch facility in Zwei has been enhanced to automatically include the neces-
  sary conditionalization forms (for example, **#+3600** and **#+IMach**) when adding
  definitions to a patch from files in SCT modules which do not specify **:machine-
  type :all**.

Changes to Concordia in Genera 8.3

- The default for the Concordia Style Parameter **indexstyle** has been changed
  from Permuted to Multilevel. This means that indexes are no longer going to
  come out permuted, but rather as single column standard indexes. If you like
  permuted indexes and wish to restore the previous behavior, you can reset the
  Style Parameter **indexstyle**.

  IndexStyle

  Controls the style of index used by Concordia. It has two values, MultiLevel, the
  standard index, and Permuted, the style used in *Genera Workbook* and *Genera
  User's Guide* . The default is MultiLevel. To produce books that have Permuted
  indexes, reset this style parameter in your Book Design by adding (indexstyle
  permuted) to your **Initialize**-*your-book-design*-**LGP2** form.

- In earlier releases, typing Select Activity (activity name) Symbolics Concordia
  selected Concordia, but the menu and title panes were blank. This bug has been
  fixed; the menu and title panes are now properly displayed when Select Activity
  is used to create a Symbolics Concordia frame.

Graphic Editor Image Scaling Fixed

The **:scale** keyword in the Graphic Editor was replaced by **:scale-x** and **:scale-y** in
Concordia 3.1. This was an incompatible change affecting all scaled images dumped
out in the previous version of Concordia. (Images that were not scaled are unaf-
fected.) This has been fixed. Newly created images should be scaled using **:scale-x**
and **scale-y**, but old images having just a **:scale** attribute are still recognized.

Changes to MacIvory in Genera 8.3

- A bug that caused :Show Machine Configuration :Output Destination File *path-name* to blow up while trying to read the cursor position of the file has been fixed.

- MacIvory has been qualified for use in the Macintosh Quadra 950.

- The Logitech MouseMan 3-button mouse is now supported by MacIvory. See the section "Configuring the Logitech MouseMan for Use with a MacIvory" for instructions on how to setup the MouseMan so that it will be recognized by your MacIvory.

- MacIvory has been changed to use smaller fonts when running on the Macintosh 16" Color Display and similarly sized monitors (that is 832x624).

- The function **mtb:ez-sfgetfile** takes a **:types** keyword. Previously if you failed to supply the **:types** argument, no files were chosen by **mtb:_sfgetfile**. This has been fixed, if no **:types** argument is supplied, then any file type is allowed.

- The `MacIvory.lib` and `RPC.lib` libraries have been converted to THINK C 5.0. If you have your own Macintosh software that uses these libraries, you must convert that software to THINK C 5.0. If you have your own RPC clients or RPC servers that run on the Macintosh, you must also convert them to THINK C 5.0, which is an ANSI compliant C compiler. To do so, evaluate your **rpc:define-remote-c-program** forms to generate ANSI compatible source and header files. If you use the **:entries-only** and **:glue-only** suboptions in your RPC server, you must also use the new **:server-extern** suboption to generate a server functions prototypes header file which can be included in the source files of your server.

- The Macintosh toolbox interface functions **mtb:_Restart**, **mtb:_SetUpA5**, **mtb:_RestoreA5**, **mtb:_RAMSDOpen**, **mtb:_RAMSDClose**, **mtb:_SpeechOn**, **mtb:_SpeechOff**, **mtb:_SpeechRate**, **mtb:_SpeechPitch**, **mtb:_SpeechSex**, **mtb:_Reader**, and **mtb:_MacinTalk** have been removed.

  Most of these toolbox entries make up the MacinTalk interface. MacinTalk has not been supported by Apple for years and Apple has published a Technical Note urging their developers not to ship any code which used MacinTalk. In fact, trying to use MacinTalk under System 7 or on any Macintosh newer than a Macintosh II is almost certain to crash your Macintosh.

- In prior releases, a MacIvory would not recognize that the CD-ROM inserted in the player had been changed and would use the directory data of the old CD-ROM while trying to access the ISO 9660 filesystem on the new CD-ROM, resulting in unpredictable behavior. This bug has been fixed.

- The bug that prevented RTS/CTS flow control from being fully operational on a MacIvory serial stream has been fixed.

Configuring the Logitech MouseMan for Use with a MacIvory

You must be running MacIvory Support Software Version 4.2 to use the Logitech MouseMan with MacIvory. These instructions presume that you have already installed the MouseMan and its software according to Logitech's documentation.

1. Open the Mouse Key control panel.

2. Click on the Add button and select your copy of the Genera application.

3. Change the assignments for the middle and right mouse buttons to be Click.

    Repeat steps 2 and 3 for the Unassigned application as well.

4. Close the Control Panel.

You can now use all three buttons on the MouseMan just as you would use the buttons on a Symbolics mouse while in Genera. The MouseMan will continue to behave according to your previous configuration instructions outside the Genera application.

Installing Macintosh System 7.1 on a Macintosh with MacIvory

Version 4.2 of the MacIvory Support software, which is shipped with Genera 8.3, is fully compatible with Apple's Macintosh System Software 7.1. In order to upgrade a Macintosh on which MacIvory software has been installed to System 7.1, you must first remove the Genera fonts in the System file. To do so, follow these steps

1. If your Macintosh has a CD-ROM drive, insert the Genera 8.3 CD-ROM into the drive. A volume named MacIvory Support 4.2 will appear on your desktop and will open automatically.

2. If your Macintosh doesn't have a CD-ROM drive or you wish to use diskettes anyway, insert the diskette named MacIvory Installer into your Macintosh.

3. Double click on the Installer icon.

4. If the disk chosen by the Installer is incorrect, use the Switch Disk button to select the proper disk.

5. Click on the Customize button.

6. Press the Option key and keep it depressed throughout the remainder of this process. (On a Symolics keyboard, use the Meta key in place of the Option key.)

7. Use the scroll bar to find the item named MacIvory fonts for Sys 7.0.1 or earlier and select it by clicking on it. This item will be the only one that is not disabled (i.e., "grayed-out").

8. Click on the Remove button.

9. After the Installer reports that removal was successful, click on the Quit button.

Your Macintosh is now ready to be upgraded to System 7.1. After installing System 7.1, you must re-install the MacIvory Support software according to the instructions in the sections "Installing Macintosh Software From a CD-ROM" and "Installing the MacIvory Diskettes".

Changes for MacIvory Delivery in Genera 8.3

If you plan to load the serial system on the Delivery-Docex System in the 8.3 MacIvory Delivery world, you must increase your wired address space when you boot. Edit your boot file and add the line

```
Set Wired Addresses 37001206000
```

after the `Load World` command and before the `Start` command.

Changes to Utilities in Genera 8.3

Notifications behave differently on the NXP1000 and MacIvory and UX machines than on other Symbolics systems. On other systems, if a notification is generated before the system's console is available, the notification is delivered to the cold load stream. On the NXP1000 and MacIvory and UX machines, the notification is held until the console is available and is then delivered. The Show Notifications CP command can be used when in the cold load stream to display any pending notifications that are waiting for the creation of a console.

Changes to Utilities on the NXP1000

- A bug that sometimes caused the :Boot Machine command on the NXP1000 to fail before causing the machine to reboot has been fixed.

- To halt the netboot process on an NXP1000, press the NMI button on the front panel of the machine. This is the equivalent of `h-c-FUNCTION` on other Symbolics machines.

- On the NXP1000, `FUNCTION-SUSPEND` and the Emergency Break Activity in the System menu are inoperative unless the debug switch is in the DEBUG (that is, up) position. This is different from other Symbolics systems.

Changes to SCT

- The bug in SCT that caused some modules not to be loaded when the systems specified in their **:load-when-systems-loaded** clauses were finally loaded has been fixed. This bug prevented the files UNIX-LPD and LASERWRITER-VIA-APPLETALK from both being loaded into a 3600-family world after loading the UX-Support and MacIvory-Support systems.

- Several SCT functions check to see if the system directory has changed, and if it has, issue a message. Previously, this message was misleading. In Genera 8.3, the message has been clarified.

Changes to the File System

- The value of the **:block-size** property reported by **fs:directory-list** for Macintosh and ISO9660 file systems is now the number of bits in a block; in earlier releases, it was the number of bytes in a block.

- The value of the **:block-size** property reported by **fs:directory-list** for the FEP filesystem of an Ivory machine is now accurate; in earlier releases, it was the number of bits in a block on the FEP filesystem of a 3600-family machine.

Miscellaneous Changes to Utilities

- An incompatibility was inadvertently introduced that makes it impossible to net-boot an 8.3 or 8.2 world from an 8.1 Netboot Server. This incompatibility is resolved by a patch for Genera 8.1 netboot servers. This patch for Genera 8.1 is included in Genera 8.3 distribution, and should be loaded on all 8.1 netboot servers. The patch in the 8.3 distribution is called `sys:netboot;netboot-compatibility-for-8-1`.

- The function **tv:mouse-y-or-n-p** now offers both a `Yes` and a `No` option that can be clicked on. Formerly, you had to move the mouse off of the menu in order to respond `No` to such a query; this is still possible, but it is no longer the only way to respond `No`.

- A bug in **who-calls** that caused the **:package** keyword to be ignored if both **:package** and **:system** were specified has been fixed.

- A bug that caused **get-universal-time** to stop properly tracking time if it is not called at least once every 36 minutes has been fixed. When this bug triggered, consecutive calls to **get-universal-time** would always return the same value regardless of how much time elapsed between calls.

- The Document Examiner and other documentation viewing utilities tend to hold open file streams to documentation files longer than most other applications hold open files. They expect to refer to further documentation in those files. Sometimes due to temporary network problems or the rebooting of a file server host, these long-lived file streams would be lost without the documentation viewing software realizing it. When the user attempted to read further documentation from such a stream the error handler was invoked. The documentation viewing substrate has been made more robust against such problems. It will now attempt to re-open a stream that was lost, rather than issuing an error.

Changes to Hardcopy and Printer Support

- Previously, many Postscript files would not print on a printer spooled through a Unix host. This bug is fixed.

- Previously, printing in landscape mode placed the top of the image at the physical-right of the sheet of paper. This placed the image so that the holes of punched-paper were below the image, which is generally considered incorrect. In Genera 8.3, the top of the printed image is placed at the physical-left of the sheet of paper in landscape mode, so that the holes are at the top when using 3-hole paper.

- Previously, some characters in the Symbol Font were overprinted by the following character when hardcopied on Postscript printers. In Genera 8.3, such characters print correctly.

Changes to Network Software in Genera 8.3

- The Define Site CP Command now provides a default value for the secure-subnets namespace object based on the primary network and address of the local host. If the primary network is Internet, the secure-subnets field makes all hosts on the same official internet network be trusted. If the primary network is Chaos, the secure subnets field will make all hosts on the same chaos subnet be trusted. For example, if the local host is booted with the network address Internet|128.81.41.93, the secure subnets field in the site object will be set to Internet 128.81.0.0. If the local host is booted with the network address Chaos|24425, the secure subnets field in the site object will be set to CHAOS 51. For more information on Internet subnet numbers see the topics "IP/TCP Support for Subnetting" and "Format of Internet Addresses". For more information on chaos subnets, see the section "Chaosnet Addresses and Indices" and see the section "Format of Chaosnet Addresses".

- Previously, :Start X Screen Internet|*A.B.C.D* :Protocol TCP, where *A.B.C.D* is the address of an internet host, would sometimes fail to parse over a Telnet connection. This bug has been fixed.

- Some users who use Telnet to connect to Genera have observed that the first character of each new line is automatically uppercased. When this occurs, it is probably because their Telnet program is sending an unwanted ASCII NUL as a buffering character after each `CR/LF` sequence. Such a character is treated by Genera as a `Control-@`, which is the `Shift` prefix on a Telnet window. This has been fixed.

- Previously, the NFS client allowed access if any of the user, group, or world permission bits matched. In Genera 8.3, its behavior has been made consistent with Unix, in that only the most specific applicable permission bits are checked.

- In Genera 8.2 NXP, there were cases where a server could cause an outbound TCP connection to hang and the server eventually ran out of packet buffers. This bug has been fixed.

- Some NXP1000 users who have been using XDMCP to start their X-sessions with the NXP sometimes saw duplicate initial-windows displayed. This problem has been fixed.

- The RPC entry for starting an X Screen did not support adequare access controls for untrusted sites. This security hole has been closed.

- The following keyword options for the function **net:define-server** now both have a default value of **t**:

  :reject-unless-trusted
  The value of this keyword is **t** by default. It causes the server request to be rejected if the host requesting service is not trusted.

  :who-line
  The value of this keyword is **t** by default. It causes a message to be displayed in the status line while the server is active. It also causes the server to appear in the Peek active server display.

Changes to CD-ROM Support in Genera 8.3

- In prior releases, a MacIvory would not recognize that the CD-ROM inserted in the player had been changed and would use the directory data of the old CD-ROM while trying to access the ISO 9660 filesystem on the new CD-ROM, resulting in unpredictable behavior. This bug has been fixed.

Changes to the Graphics Facilities in Genera 8.3

Some keyword arguments were missing from the argument list declaration for **graphics:draw-circle**. The complete list of arguments is

graphics:draw-circle *center-x center-y radius* &key *(:inner-radius* **0***) (:start-angle* **0***)*
(:end-angle **graphics:2pi***) (:alu* **:draw***) (:filled* **t***) :color (:gray-level* **1***) :tile :stipple :re-*
turn-presentation :pattern (:opaque **t***) :mask (:mask-x* **0***) (:mask-y* **0***) :thickness (:scale-*
thickness **t***) (:line-end-shape* **:butt***) (:line-joint-shape* **:miter***) :dashed :dash-pattern*
(:initial-dash-phase **0***) (:draw-partial-dashes* **t***) :scale-dashes (:stream* ***standard-**
output**) (:rotation* **0***) :clockwise :join-to-path (:scale* **1***) (:scale-x* **1***) (:scale-y* **1***) :trans-*
lation :transform

Changes to X Client

- A bug where a pop-up notification sent to an X-window that is iconified caused the window system to be locked has been fixed.

- Previously, connecting to an X server without the Genera fonts loaded caused the corruption of default font mappings, which lead to problems when trying to use a non-X screen. This has been fixed by caching only device-font mappings in the default mapping table.

- In prior releases, if the Keyboard Control activity was selected from an X console whose keyboard was not recognized by Genera, the activity would get an error. This bug has been fixed; the Keyboard Control activity now issues a notification and refuses to be selected in this case.

Changes to Statice and CLOS

As a part of our continuing support for standards Symbolics is introducing a CLOS version of Statice with this release. This also furthers our goal of providing more portability in its products. A Flavors version of Statice will also be shipped with this release. In future releases the Flavors version will no longer be supported.

Both a CLOS version of Statice and a Flavors version of Statice are distributed with Genera 8.3. The two versions may not run on the same machine simultaneously. That is, the CLOS version of Statice or Statice-runtime many not be loaded on a client machine that is running the Flavors version of Statice or Statice-runtime, or vice versa.

Flavors Statice to CLOS Statice Conversion

CLOS Statice and Flavors Statice cannot be run simultaneously on the same client machine. If you have more than one Statice application running on a client machine, all applications must run the same version of Statice. You must convert and recompile the code that runs on client machines to use CLOS Statice.

It is not necessary to change the database and the Statice server to run CLOS Statice. A server will work with one client machine running Flavors Statice and a separate client machine running CLOS Statice.

To translate code using Flavors Statice to CLOS Statice, use one of the following commands:

```
m-X Convert Functions of Region
m-X Convert Functions of Buffer
m-X Convert Functions of Tag Table
```

When prompted `Conversion to use`, select Flavors Statice to CLOS Statice. This somewhat automatic conversion will do most of the work required to convert a program from Flavors Statice to CLOS Statice.

Each form in the program that can be converted to CLOS Statice results in an interactive query showing the old form in context in the editor buffer and one or more suggested replacement functions. You can then enter one of a variety of single character commands. Press HELP for a list of options. For further information, see the section "Getting Help with Conversion".

This is neither a conversion from Symbolics Common Lisp to portable Common Lisp nor is it a conversion from Flavors to CLOS. The Flavors Statice to CLOS conversion converts only Flavors Statice functions and macros; it does not touch the rest of the program. You should use the other conversion tools to change your code.

The general idea of a conversion is to convert an entity built with flavors to one made as a class with the same name. Flavors names and syntax are replaced with CLOS names and syntax wherever a correspondence exists. Flavors features that do not exist in CLOS are left in the program for you to convert by hand.

CLOS Statice is the default system loaded when you load Statice. CLOS Statice presumes that your code uses CLOS. CLOS Statice uses symbols that are not accessible in packages such as CL-USER and SCL. You should convert your code to CLOS prior to converting to CLOS Statice. See the section "Flavors to CLOS Conversion" before attempting to convert to CLOS Statice.

The conversion tool extracts information about your program, such as what method-combination type a generic function uses or what instance variables function is defined for a flavor, by looking in three places:

1. Forms that have already been converted during the same conversion operation.

2. Forms that have been read into editor buffers.

3. Flavor and generic function definitions that have been loaded into the world.

Consequently, you will get better results if the entire program is read into the editor or loaded into the world before converting any of it.

The new version of the program cannott be used at the same time as the old version, because it uses the same names with different meanings (a class is different from a flavor, a CLOS Statice attribute is different from a Flavors Statice attribute). However, if you move the new version into a new package before converting it you can avoid this problem. For information about moving your Flavors Statice program to another package, see the section "Package Conversion".

Mixed use of Flavors Statice and CLOS Statice is not supported at present. That is, a class base entity cannot inherit from a flavor based one, a flavor based entity cannot inherit from a class based one, CLOS generic functions cannot be used with Flavors methods, and Flavors generic functions cannot be used with CLOS methods nor with CLOS instances.

Loading Statice

Both a CLOS version of Statice and a Flavors version of Statice are available with Genera 8.3. The two versions may not run on the same machine simultaneously. By default, the CLOS version of Statice is not called Statice. To load the CLOS version of Statice, type

```
Load System Statice-Runtime
```

To load the Flavors version of Statice, type

```
Load System Flavors-Statice
```

CLOS Statice and CLOS Statice-runtime are both compatible with databases built with the existing Flavors Statice. The client portions of CLOS Statice and CLOS Statice-runtime are not compatible with the Flavors Statice or Flavors Statice-runtime client respectively. The CLOS version of Statice or Statice-runtime may not be loaded on a client machine that is running the Flavors version of Statice or Statice-runtime. A client machine running either CLOS Statice or CLOS Statice-runtime may access data on a server machine that is running Flavors Statice or Flavors Statice-runtime.

Also note the following changes to Statice in Genera 8.3:

* **statice::view-entity** now displays slot names in a column of appropriate width. Previously, too much space was allocated.

* Previously, you could occasionly enter the debugger while building a database due to Statice trying to update the area map during a page deallocate without a page allocator bound. In Genera 8.3 a safe allocator is automatically bound so that this no longer occurs.

Miscellaneous Changes in Genera 8.3

* :Copy World now works over IP-TCP. To use Internet band transfer, add the following service to your host namespace object:
```
BAND-TRANSFER TCP BAND-TRANSFER
```
Note that this feature is included in Genera 8.3 and Genera 8.2 NXP, but not in Genera 8.1. Internet band transfer will not work between and 8.1 machine and an 8.3 or 8.2 NXP machine.

- Source locators are fixed and work as described in the Genera documentation. See the section "How Programs Manipulate Definitions".

- Previously, the presence of a captive variable block caused the debugger to mis-report local variables. This has been fixed in Genera 8.3.

- Some system operations will automatically login a default user (usually Lisp-Machine) in order to read files from a file server, etc. when no user is logged in. Because this login operation can inhibit remote-login, the code that performs the automatic login has been changed to enable remote login only if no user was logged in. This will not affect normal operation when a user is already logged in.

- The Create Initial FEP Filesystem command now accepts any disk unit currently mounted. Previously, the abbreviated presentation type was evaluated only when the machine was booted. If a newly initialized disk was mounted, it was not accepted. This bug has been fixed.

- A bug in which describing certain symbols, such as **si:string-char-epsilon-input-stream**, crashed the machine has been fixed.

- New keyboard commands are available for scrolling while editing directories in FSEdit. These commands are the same keyboard commands available in the command processor.

```
c-V                     Scroll text forward.
m-V                     Scroll text backward.
Scroll                  Scroll text forward.
Meta-Scroll             Scroll text backward.
Control-Meta-Scroll     Scroll typeout window backward.
Control-Scroll          Scroll typeout window forward.
Super-Scroll            Scroll text right.
Super-Meta-Scroll       Scroll text left.
Control-L and Refresh   Return from typeout window scrolling
                        to directory editing without a query.
```

  A bug in Dynamic Windows that caused the incorrect computation of vertical cursor-advance when displaying menus with the keyword :**inter-row-spacing** other than zero has been fixed.

- A bug in the presentation subtype handling for the **alist-member** presentation-type was fixed (which implicitly affects the **boolean** presentation type). When **alist-member** presentations are visible, they only become mouse-sensitive in a call to **accept** if they are compatible with the type being accepted.

- Previously, decompressing from an input stream over FTP resulted in an error. This bug has been fixed.

- The FEP command Set Network Address has a new option *ip-secure-subnets* lists secure subnets. For example,

  ```
  :Set Network Address
  domino-0:internet|128.81.41.93;ip-secure-subnets=128.81.41.0
  ```

  Use commas to separate more than one subnet, or use the string *all* to allow access from any Internet host.

- A bug in which **case** was compiled incorrectly for clauses of the form (nil . *forms*) has been fixed.

- In previous releases, when in FSEdit, clicking Mouse-Right on a '.sab' file and then [Edit File (L)] would try to edit the file using Zmacs. In Genera 8.3, Concordia (if it is loaded) is used instead to edit such a file, just as the Edit File CP command would do.

- In Genera 8.1, when sending nested list structures from C to Lisp, the LIST remote type generated incorrect C code for calculating the size of the structure to be sent. This bug has been fixed in 8.3.

Notes and Clarifications for Genera 8.3

Function Q Update

The Options for FUNCTION Q were never adequately explained in *Genera Handbook,* specifically, the options available when you supply an argument, FUNCTION 0 Q were not documented. Here is the corrected documentation for FUNCTION Q.

FUNCTION Q

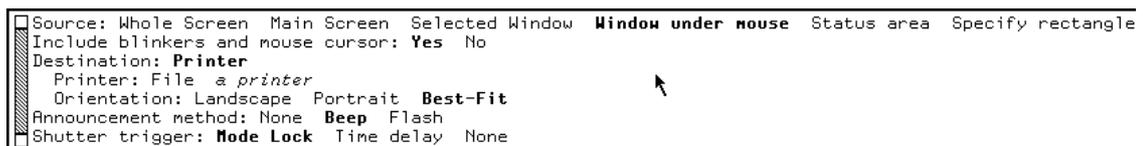|	
Function Q	Captures a screen image for hardcopying or inserting in a file.
Function *n* Q	(where *n* is any numeric argument, for example FUNCTION 0 Q) Displays a menu of options for capturing screen images. Note that the choices you make remain in effect for subsequent uses of FUNCTION Q.

```
Source: Whole Screen  Main Screen  Selected Window  Window under mouse  Status area  Specify rectangle
Include blinkers and mouse cursor: Yes  No
Destination: Printer
  Printer: File  a printer
  Orientation: Landscape  Portrait  Best-Fit
Announcement method: None  Beep  Flash
Shutter trigger: Mode Lock  Time delay  None
```

Figure 3. FUNCTION 0 Q

`Source:` options identify the area of the screen to capture:

Whole Screen	A bitmap image of the entire screen
	Note that a full-screen bitmap image includes a border around the actual screen image. If you do not want this extra whitespace around the image, select the Named Image option and edit the bitmap image using the "Refit Bounding Box Bitmap Editor Command".
Main Screen	All but the status area
Selected Window	The window of the selected activity (for example, Symbolics Concordia)
Window Under the Mouse	
	The window under the mouse cursor
	Use this option when you cannot use the mouse to specify a portion of the screen (for instance, to take a snapshot of a pop-up menu).
Status Area	The window at the bottom of the screen (including the mouse information line(s))
Specify Rectangle	Specify the area of the screen to be captured using the mouse
Window History	The history of the specified window

`Shutter Trigger:`

Mode Lock	Captures the image when the `MODE LOCK` key is pressed. This allows you to *set up* the screen a certain way, perhaps with a menu showing, or the mouse cursor in a certain place.
Time Delay	Allows you to specify a time delay before capturing the image. The default is 5 seconds.
None	Captures the image as soon as you click on `[Done]`. This is the default.

How to Delete and Expunge a File Left Open by an Aborted Restore Distribution

If your machine halts at an inopportune time, for instance when a file is open for writing during a Disk Restore, you might leave LMFS's data structures in an inconsistent state; `c-m-ABORT` or, in some circumstances, `c-ABORT` can do this, especially on Ivory machines. If the internal structure is inconsistent LMFS is not able to Delete and Expunge. This leaves the file marked with a `W`. Deleting and expunging seems to have no effect. The file has length 0, and you cannot open it because it is already open. `[Close All Files]` doesn't affect the file's status. Try a warm boot (of the server machine, of course). After a boot, LMFS's datastructures are

rebuilt. The file is still marked with a W because LMFS knows that it was in the middle of being written, but you can delete and expunge it normally.

Symbolics dpANS Common Lisp

Genera 8.3 contains our first release of an implementation of a Common Lisp that attempts to conform to the "draft proposed American National Standard for Information Systems---Programming Language---Common Lisp" (dpANS Common Lisp). There are still some unimplemented language features; see the section "Features of dpANS Common Lisp Not Yet Implemented for Genera". However, we believe that conformance is close enough to the draft proposed standard that its text (augmented by our list of differences) can and should be used as a reference document instead of any proprietary documentation that we might normally provide.

Overview of dpANS Common Lisp Documentation

In general, the design philosophy of the Symbolics implementation of dpANS Common Lisp has been a conservative one. Wherever possible, we have generally tried to avoid adding proprietary extensions in the standard Common-Lisp package. As a result, you should be able to use the same dpANS Common Lisp documentation as you would use for any other implementation. The main additional information you need is our documentation on how to access this facility (see the section "How dpANS Common Lisp Fits Into Genera") and our list of unimplemented features (see the section "Features of dpANS Common Lisp Not Yet Implemented for Genera").

How dpANS Common Lisp Fits Into Genera

In the Common-Lisp syntax, the packages named **common-lisp**, **cl**, and **lisp** denote the same package, which corresponds to functionality defined by CLtL. Likewise, the packages named **user** and **cl-user** denote the same package, which uses the **lisp** package. In this syntax, a package named **future-common-lisp** is available to refer to functionality defined by dpANS Common Lisp.

In the ANSI-Common-Lisp syntax, the name **lisp** denotes the package name containing CLtL-style functionality, but the names **common-lisp** and **cl** denote a new package which implements the dpANS Common Lisp.

Here is a table which shows the relationship of package names between the two syntaxes:

Global name	Common-Lisp name	ANSI-Common-Lisp name
future-common-lisp	**future-common-lisp**	**common-lisp** or **cl**
future-common-lisp-user	**future-common-lisp-user** or **fcl-user**	**common-lisp-user**
common-lisp or **cl** or **lisp**	**common-lisp** or **cl** or **lisp**	**lisp**
common-lisp-user or **cl-user**	**common-lisp-user** or **cl-user**	**user**

User-defined packages created while in the ANSI-Common-Lisp syntax will be visible to other syntaxes (for example, Common-Lisp, Zetalisp, and CLtL) except those which specifically screen out Genera packages (for example, CLtL-Only). Similarly, user-defined packages created in other syntaxes (for example, Common-Lisp and Zetalisp) will be visible in the ANSI-Common-Lisp package universe.

For an explanation of how package universes are managed, refer to the section "Packages Universes in dpANS CL".

Using dpANS Common Lisp From a File

When in Lisp Mode in Zmacs, you can use the command Set Lisp Syntax (m-X) to set the syntax of the buffer to ANSI-Common-Lisp. Or, you can edit the attribute list manually to something like the following:

```
;;; -*- Mode: Lisp; Syntax: ANSI-Common-Lisp; Package: CL-USER -*-
```

Then you can use Reparse Attribute List (m-X) to make sure that Zmacs has taken notice of your manual changes. Later, when the file is read into Zmacs anew, Zmacs will automatically put the file into the right syntax and package.

> **Note:** Unlike in syntax Common-Lisp, specifying package **cl-user** is *not* the same as specifying package **user**. (See the table of package name correspondences in the section "How dpANS Common Lisp Fits Into Genera".) Specifying package **cl-user** gives you the new functionality defined by dpANS CL, which is the correct thing in most cases. If you instead specify only package **user**, you will get the CLtL-compatible package; when this happens unintentionally, it can cause confusion, since the dpANS Common-Lisp functionality seems to be missing. If this happens, you should check your file attribute list to make sure you are using the right package.

To maximize code portability, you should add an **in-package** form immediately after the file attribute list, so that non-Symbolics systems that do not use the file attribute list will start on an even footing.

Here is an example of how such a program might be set up:

```
File CHEM:CODE;SYSDCL.LISP


 ;;; -*- Mode: LISP; Syntax: ANSI-Common-Lisp; Package: CL-USER -*-
 ;;; System declaration file for the CHEMISTRY system.


 (IN-PACKAGE :CL-USER)


 (SCT:DEFSYSTEM CHEMISTRY
     (:DEFAULT-PATHNAME "CHEM:CODE;"
      :JOURNAL-DIRECTORY "CHEM:PATCH;")
     (:SERIAL "PKGDCL" "MAIN"))


File CHEM:CODE;PKGDCL.LISP


;;; -*- Mode: LISP; Syntax: ANSI-Common-Lisp; Package: CL-USER -*-
;;; Package declaration file for the CHEMISTRY system.


(IN-PACKAGE :CL-USER)


(DEFPACKAGE CHEMISTRY
  (:NICKNAME "CHEM"))


File CHEM:CODE;MAIN.LISP


;;; -*- Mode: LISP; Syntax: ANSI-Common-Lisp; Package: CHEMISTRY -*-
;;; Main body of code implementing the CHEMISTRY system.
;;; Still a bit more coding to do here...


(IN-PACKAGE :CHEMISTRY)


(DEFUN MAIN () (ERROR "Not yet implemented."))
```

Using dpANS Common Lisp Interactively

At the Lisp Listener, you can use the command Set Lisp Context to globally
change the current context to ANSI-Common-Lisp. For example:

```
Set Lisp Context (a lisp syntax name [default Common-Lisp]) ANSI-Common-Lisp
```

Features of dpANS Common Lisp Not Yet Implemented for Genera

Setting the **&rest** parameter from **&key** initializers results in a compiler bug.

The scope of type declarations does not conform exactly to the description in
dpANS Common Lisp.

Package names are not currently case sensitive. This means, for example, that it is currently not possible to simultaneously have a package named "FOO" and a distinct package named "foo".

conditions:restart-bind does not work when the "handler" function does not throw. That is, it is useful for implementing **conditions:restart-case**, but not for implementing things like "debugger special commands."

The restarts **conditions:abort**, **conditions:store-value**, **conditions:muffle-warning**, **conditions:continue**, and **condtions:use-value** might not be available in all the places you expect them.

future-common-lisp:open-stream-p is currently only implemented for file streams.

The stream types are not always disjoint from the type **symbol**. One consequence is that value of the following variables is sometimes a symbol: ***debug-io***, ***standard-input***, ***error-output***, ***standard-output***, ***query-io***, and ***trace-output***.

Most of the new **pprint**-related format operations are not supported.

The following **pprint** substrate is largely still missing:

> ***print-lines***
> ***print-miser-width***
> ***print-pprint-dispatch***
> ***print-right-margin***
> **file-string-length**
> **copy-pprint-dispatch**
> **set-pprint-dispatch**
> **pprint-dispatch**
> **pprint-exit-if-list-exhausted**
> **pprint-fill**
> **pprint-indent**
> **pprint-linear**
> **pprint-logical-block**
> **pprint-newline**
> **pprint-pop**
> **pprint-tab**
> **pprint-tabular**

The function that is the value of the variable ***macroexpand-hook*** is not called in all situations where dpANS CL prescribes that it should be. In particular, it is not called in certain situations involving **setf** that are new since CLtL.

The **future-common-lisp:readtable-case** value of **:invert** for a readtable is not supported. (The other possible values of **:upcase**, **:downcase**, and **:preserve** are supported.)

Proper contagion of mixed floats and rationals in the following operations is not supported:

=	/=
<	<=
>	>=
MIN	**MAX**

equalp hash tables are only partly implemented. Because **equalp** tables depend on the proper contagion of mixed floats and rationals for operations in the previous item, you must not used mixed floats and rationals as if they were interchangeable keys (or key components) in **equalp** hash tables. Floating point contagion was not implemented because there was too much risk of breaking current user code.

future-common-lisp:stream-element-type returns type names in the LISP package, not the CL package. For example

```
(with-input-from-string (str
        (make-string 3 :element-type 'base-char))
            (stream-element-type str)) → LISP:STRING-CHAR
```

Some system methods for **future-common-lisp:describe** (such as the one for hash tables) do interactive querying.

future-common-lisp:function-lambda-expression sometimes returns "digested expressions" such as **si:digested-lambda** expressions.

future-common-lisp:load and **future-common-lisp:compile** ignore their **:print** argument, which implicitly means that ***load-print*** and ***compile-print*** do not have their desired effect.

Some symbols have properties that are indicated by symbols in the CL-LISP, USER or KEYWORD packages.

If you notice other missing features that are not listed here but are described in the dpANS Common Lisp specification, please feel free to mention their absence to a Symbolics representative so that we can be sure they are on our "to do" list for a future implementation.

Extensions That Might Go Away

Some aspects of Genera's implementation of dpANS Common Lisp may appear like features because they extend the implementation of Common Lisp beyond the point where it was designed, and because they mirror features of the pre-existing Common-Lisp implementation. Just because you observe an apparent extension to be present does not mean that it is intentionally present; before relying on such extensions, please consult the documentation.

This section describes some known situations where such pitfalls occur. You should not rely on the situations described in this section; your code should take care to avoid these situations to guard against problems that might be caused by future change. If you need these extensions, you are best off to call the SCL equivalents of the functions in question, both to remind yourself that the code is not portable and to insulate yourself from changes in the dpANS implementation.

- **future-common-lisp:apply** and **future-common-lisp:funcall** happen to permit you to call an array as a function.

- **future-common-lisp:adjust-array** permits you to adjust arrays that have not been declared to be **:adjustable**.

- **future-common-lisp:make-array** and **future-common-lisp:adjust-array** permit you to create arrays displaced to arrays that are not of the same type.

- **future-common-lisp:open** takes arguments other than those described by Common Lisp.

- **future-common-lisp:documentation** permits some non-standard documentation types.

If you notice other apparent extensions that are not marked here and not described in the dpANS Common Lisp specification, it is recommended that you contact a Symbolics representative before writing code that depends critically on such extensions.

CLIM 2.0 in Genera 8.3

No documentation available for section CLIM 2.0 Release Notes and Installation Guide.