# HOTLINE!

Bulletin 3
September 25, 1987

*HOTLINE!* is published periodically by the Customer Support group of Xerox Artificial Intelligence Systems to assist its customers in using the Lyric Release of Xerox Lisp. We will be covering a variety of topics and answering questions that are most frequently asked of Customer Support. We'll also include suggestions to help you get started in Lyric as well as announcements of known problems that you may encounter.

Feel free to make copies of individual bulletin pages and insert them in the appropriate place(s) in your Interlisp Reference Manual, Lisp Library Modules manual or other relevant manual. The documentation reference at the end of each topic can be used as a filing guide.

For more information on the questions or problems addressed in this or other bulletins please call us toll-free in the Continental United States 1-800-228-5325 (or in California 1-800-824-6449). Customer Support can also be reached via the Arpanet by sending mail to AISUPPORT.PASA@Xerox.com., or by writing to:

Xerox AIS Customer Support
250 North Halstead Street
P.O. Box7018
Pasadena, CA 91109-7018
M/S 5910-432

## In this issue

In response to user requests we have decided to have *HOTLINE!* cover all supported releases of XAIE, instead of Lyric only. Supported releases include Koto and Lyric. Each item now contains a "Release" field for any item that is release specific.

The following topics are covered in this issue:

- How to close open streams
- Saving macros in files
- NAME COMMANDS spontaneous redefinition
- Interfacing an 1186 to a VAX 11/780 and Sun 3/160
- Exporting Symbols from Packages
- Making Tedit Read Only
- Over-riding the default compiler
- Changing the Default Executive type

Terminology used in the *HOTLINE!* bulletin:

AR – Action Request, a Xerox problem tracking number (e.g. AR 8321)

IRM – Interlisp Reference Manual

UG – User's Guide

# How to close open streams

**Topic** How to close open streams that are not bound to an atom

**Release** Lyric

**Keywords** Streams, Files

**Question** How do you close open streams that are not bound to an atom?

**Background** In prior releases of Interlisp-D the system treated the name of an open stream as a synonym for the stream open on the file. In the Lyric relase, a filename is no longer a unique name for an open stream. If you open a stream to a file and do not bind it to an atom, you will not be able to close the stream by calling CLOSEF on the filename.

**Answer** Answer: In order to close a stream not bound to an atom first call in the Interlisp exec :

(INSPECT (OPENP))

Select "Inspect" from the pop-up menu that appears. Your file should appear in the inspector in the form:

#<IO stream 45,7561 on {dsk}<lispfiles>foo

(CLOSEF

and then shift select the file you want to close from the inspector window into the CLOSEF expression. It will look something like:

(CLOSEF (\VAG2 45Q 7561Q)

Type in the trailing ")" and the file will be closed.

# Saving macros in files

**Topic** Saving macros in files

**Release** Lyric

**Keywords** Macros, FilePackage

**Question** How do I save Koto macro definitions in Lyric?

**Background** In Interlisp-D, macros are saved on property lists. The DMACRO property identifies Interlisp-D macros. The MACRO property identifies "implementation independent" macros and is used to save macros defined with the Koto function DEFMACRO. Common Lisp does not allow a symbol to have both a macro and function definition; you may either (DEFUN FOO --) or (DEFMACRO FOO --), but not both. In Lyric, this restriction also applies to Interlisp macros defined by DEFMACRO.

**Answer** Use the FileManager command FUNCTIONS to save macros defined with DEFMACRO. Use the FileManager command MACROS to save all other types of Interlisp macros.

**Reference** Lyric release Notes, page 11.

# NAME COMMANDS spontaneous redefinition

**Topic** NAME COMMANDS has a mysterious habit of spontaneous redefinition.

**Release** Lyric

**Keywords** DEFCOMMAND, NAME COMMANDS

**Problem** If you use NAME COMMANDS in Lyric, you will discover that they have a mysterious habit of spontaneous redefinition. There is no easy way to give a specific case where you can reproduce it, since you never know when it might happen.

**Workaround** This is a bug in the NAME command. The SUBPAIR should have a T as its last argument (saying to copy the entire structure.) That is because the output of :history commands are used directly in the event, and a subsequent fix or other history operation can change it.

Here's a patch:

In the XCL exec window type the following:

```
(in-package "INTERLISP")

(defcommand "NAME" (command-name &optional arguments
&rest event-spec)
   "NAME command-name [arguments] [event-spec] defines
   new command"
   (cl:unless (listp arguments)
            (setq event-spec (cons arguments event-spec))
            (setq arguments nil))
    (let ((events (find-history-events event-spec lispxhistory))
       (argnames (for i from 1 as x in arguments collect (pack*
       (quote arg) i))))
   (cl:eval `(defcommand (,command-name :history) , argnames
            (subpair ',argnames (list ,@ argnames)
                ',(subpair arguments argnames(events-input
                    (events-input events) t))))))
```

# Interfacing an 1186 to a VAX 11/780 and Sun 3/160

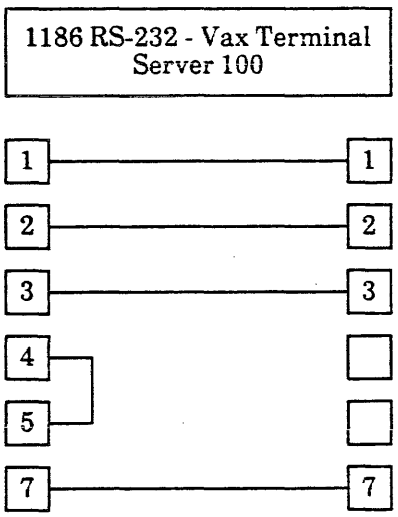
**Topic** Interfacing an 1186 RS-232 port to a Vax 11/780 and Sun 3/160 port A

**Release** Koto, Lyric

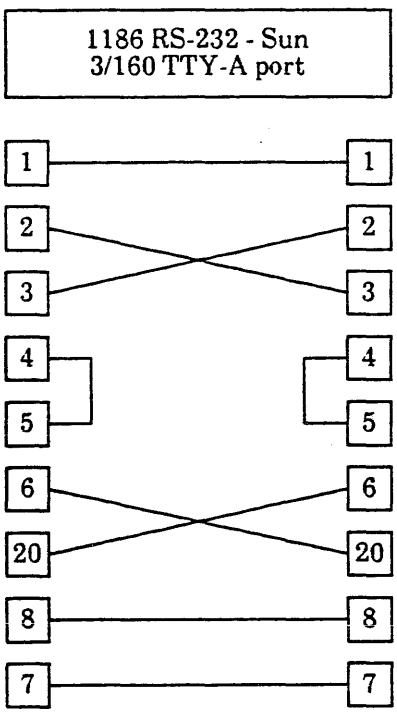
**Keywords** RS-232, Vax 11/780, Sun 3/160

**Question** How do I correctly interface my 1186 to a Vax 11/780 and a Sun 3/160?

**Answer** For connecting to a Vax Terminal Server 100, the following cable should be used:




For connecting to a Sun 3/160 TTY-A port, use the following cable:




**Reference** 1186 Users Guide, Section 7. Library Modules, RS-232

# Exporting Symbols from Packages

**Topic** Exporting Symbols from Packages

**Release** Lyric

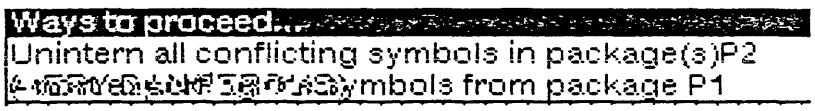
**Keywords** Packages, Export, Use-package, Unuse-package, Name-conflicts, Unintern

**Question** How do I EXPORT a symbol from a package (say P1) in a case where it is in name-conflict with another package (say P2) that uses the package P1?

**Background** If you try to load a module that exports a symbol (e.g. X) that is name conflict with another package you will get a break:

In IL:RESOLVE EXPORT-CONFLICT:
Exporting these symbols from the P1 package:
X
results in name conflict with packages(s):
P2

The PROCEED menu in the BREAK window for the conflicting - symbols error under EXPORT comes out to be garbaged, as shown below:




One of the choices offered, to UNINTERN all the conflicting symbols in P2, doen't seem to do anything when picked: it gets you out of the BREAK, but it does not unintern any symbol and it does not do the export.

**Answer** In the XCL Exec window, do UNUSE-PACKAGE P1, do the EXPORT, then call USE-PACKAGE P1 again. When you do the USE-PACKAGE P1, you get the same error, but this time it will be under USE-PACKAGE. Also, the PROCEED menu in the BREAK window is not garbaged, and picking the UNINTERN-FROM-P2 choice works and the USE-PACKAGE succeeds. Notice that the USE-PACKAGE menu has warnings like "VERY DANGEROUS" attached

**Reference** AR #9029.

# Making Tedit Read Only

**Problem** In calling the Tedit function to bring up a Tedit window, I cannot get the Tedit property READONLY to work.

**Release** Koto, Lyric

**Keywords** Tedit, READONLY

**Example** (TEDIT 'myfile NIL NIL '(READONLY T))

**Symptom** After calling the example, users will find that they can still edit the Tedit window's contents.

**Workaround** (SETQ mystream (OPENTEXTSTREAM NIL (CREATEW) NIL NIL '(READONLY T)))

(TEDIT.GET (TEXTOBJ mystream) myfile)

**Reference** AR# 5606

# Over-riding the default complier

**Topic** Over-riding the default compiler for an individual file.

**Release** Lyric

**Keywords** MAKEFILE, CLEANUP, FILETYPE.

**Question** How do I over-ride the default compiler on an individual file?

**Background** The Lyric release has two compilers - the Interlisp Compiler and the XCL Compiler. The default compiler for the MAKEFILE and CLEANUP functions is determined by the function named by the variable *DEFAULT-CLEANUP-COMPILER*.

This system-wide default can be over-ridden for individual files.

**Answer** To over-ride the default compiler for a file, set the FILETYPE property on the root name of the file to either :TCOMPL, :BCOMPL, or :COMPILE-FILE

**Example** Suppose the value of *DEFAULT-CLEANUP-COMPILER* is CL:COMPILE-FILE (the XCL Compiler), and the user needs to compile the file TEST with the Interlisp Compiler.

Then type the following at the Interlisp Exec:

```
(PUTPROP 'TEST 'FILETYPE ':TCOMPL)
```

Or, at the XCL Exec:

```
(IL:PUTPROP 'IL:TEST 'IL:FILETYPE ':TCOMPL)
```

**Reference** Lyric Release Notes, pages 29-30, "Section 17.2 Storing Files."

# Changing the Default Executive type

**Topic** Changing the Default Executive type

**Release** Lyric

**Keywords** EXEC

**Question** How can I get the default executive to be of type INTERLISP?

**Background** In Lyric, the user is provided with several types of Execs which are primarily distinguished by their default settings for package and readtable. New Execs can be created by selecting the Exec menu item from the background pop-up menu. Initially the type of Exec created by default is XCL, which uses the XCL readtable and XCL-USER package. The user may reset this to INTERLISP, COMMON-LISP or OLD-INTERLISP.

**Answer** In the current executive, evaluate:

(XCL:SET-DEFAULT-EXEC-TYPE "IL")

This function can also be added to your init file.

**Reference** Reference: Lyric Release Notes, pages 13-15 and Appendix A, The EXEC, page A-20