Inter-Office Memorandum

To	Mesa Users	Date	May 31, 1978
From	Barbara Koalkin	Location	Palo Alto
Subject	Debugger - Extended Features	Organization	SDD/SD

XEROX

Filed on: [IRIS]<MESA>DOC>XDF.BRAVO

There are three extended features available with the Mesa 4.0 debugger on an experimental basis. One of these provides access to FTP to allow a user to retrieve files from remote locations from within the debugger. Another new debugger command allows a user to invoke a special debugging package. The window manager (**WindEx**) has been expanded in an attempt to extend the user interface to the debugger. Each feature is described below in further detail. We encourage feedback on these features and remind you that due to their experimental nature, they are subject to change.

Debugger FTP

The Mesa 4.0 debugger has the capability to invoke a subset of the standard FTP commands from within the debugger environment without having to exit to the Alto Executive. The debugger's FTP command is built on top of the standard Mesa FTP package and therefore any comments and/or problems regarding FTP itself should be addressed to the SDD Communications Group. `↑Ftp (control-F)` enables the following commands (see the FTP documentation for further details):

`Close connection [confirm]`

closes the currently open FTP connection.

`DElete filename` **filename**

tries to delete **filename** from the local disk, regardless of whether the file is in use. If it finds it impossible to delete the file due to *some of its own pointers that would be left dangling*, it does not allow you to do so (e.g., you cannot delete XDEBUG.IMAGE). *Beware of your own references!*

`DUmp to remote file` **dumpfile**

bundles together a group of files from the local file system into a 'dump format' file and stores the result as **dumpfile**. FTP asks you for the names of the local files to be included. Terminate the dump by typing a carriage return (**CR**).

`Free pages`

tells you how many free pages are left on your disk.

`LIst remote file designator` **filelist**

lists all files in the remote host corresponding to **filelist**. This must conform to the file naming conventions on the remote host. You may designate multiple files by the use of "*" expansion.

`LOad from remote file` **dumpfile**

performs the inverse operation of `DUmp`, unbundling a dump-format file in the remote system and storing the constituent files in the local file system.

`Open connection` **host, directory**

opens a connection to the FTP Server in the specified **host** and (optional) specified **directory.**

`Quit [confirm]`

takes you out of FTP mode and returns you to the debugger command processor. The debugger closes your connection when you leave FTP, if you have forgotten to do so.

`Retrieve filename` **filename**

transfers **filename** from the remote host to the local host. The filename must conform to the file naming conventions on the remote host. You may designate multiple files by the use of "*" expansion if the remote server supports them (currently Maxc and IFS do). Note that the byte count is printed out truncated to 16 bits.

`Store filename` **filename**

transfers **filename** from the local host to the remote host. Alto filename conventions apply to the local file; "*" expansion is not supported.

Warning: *Be careful not to change any files out from under the program you are debugging; the debugger makes no provisions for checking this when you ask to delete or update a local file!*

In order for you to be able to use the FTP commands, you must retrieve the file FETCH.BCD and load it into your debugger. This may be done when you are installing your debugger, in the same way as installing the window manager. Once you are inside the debugger nub, before you install, type `New -` FETCH which will load the FTP package. You may later invoke FTP at any time by typing `control-F` to the debugger. If you wish to load the FTP package at some later time, simply enter the debugger nub (`control-D`) and then load FETCH. If you do not have the FTP package loaded, you may still use the `Delete`, `Free pages`, and `Quit` commands. However trying to use any of the other FTP commands will give you the message "`-- FTP not installed`".

Debugger User Procedures

The new `↑UserProc` (`control-U`) command allows you to load a debugging package into the debugger and invoke it at any time simply by typing `control-U`. The mechanism for loading is the same as for loading the window manager. Simply enter the debugger nub; then do a `>New` **YourOwnFileName**, followed by an (optional) `Start`. Your program must

IMPORT **DebugUtilityDefs** and make a call to:

```
    AddCommand: PROCEDURE [tag: STRING, proc: PROCEDURE].
```

which expects the name of the command and the procedure to call when this command is invoked.

Internally, things work as follows: When a user makes a call to **AddCommand**, the procedure gets added to the list of user procs that have already been loaded. When the ↑UserProc command is invoked, the debugger looks to see if there have been any procedures loaded. If there is only one, it will be invoked automatically. If several user procs have been loaded, typing "?" displays a list of the available commands. The message " !No user procs." is displayed if the debugger can't find any procedures that have been loaded.

The debugger gives you added help in gaining more access to the information it already knows about your programs. Taking advantage of the configuration format for grouping modules, the debugger's configuration EXPORTS all of the debugger's interfaces: **DebugBreakptDefs**, **DebugConfigDefs**, **DebugFtpDefs**, **DebuggerDefs**, **DebugInterpretDefs**, **DebugMiscDefs**, and **DebugUtilityDefs**. A user program can get access to any of the debugger's PUBLIC procedures simply by IMPORTing the definitions modules of the procedures that you want to use. When writing your own debugging routines, look carefully at some of the utility routines that the debugger already provides (eg., **ModuleNameToFrame**, **FrameToModuleName**, **SREAD**, etc.). You should also look at the **<MesaLib>** directory for UserProcs that other Mesa users have already written and debugged.

Warning: *The Mesa Group makes no guarantees about the stability of these interfaces. Use at your own risk!*

Window manager

The new window manager (**WindEx**) has the ability to handle long files (over 64K characters), set breakpoints by selecting, position a file in a window using a character index, and (optionally) use the keyset as an input device. Several bugs have been fixed and performance has been improved. The documentation that follows is the complete documentation for **WindEx**.

Loading WindEx

To load **WindEx**, execute a New - Start sequence on the file WINDEX.BCD before installing the debugger. Alternately, you may enter the debugger nub at some later time (using control-D) and load **WindEx**. You may also load **WindEx** from the command line by typing XDebug WindEx/I to the Alto Executive which loads **WindEx** when installing the debugger (additionally, you can use the L switch to load **WindEx** with code links to save space).

Current window

Using **WindEx**, the concept of "the current window" is clearly visible. A window is current until the cursor is moved to another window and a mouse button is clicked. As a result, a window is not repainted until it is made current. After any action is taken, the current window is repainted and its selection is updated.

Text area

Selections are made by depressing either the RED or YELLOW mouse buttons while in the text area. RED selects a character and extends selections by characters; YELLOW selects a word and extends selections by words. Characters typed into a scratch window are automatically selected as they are typed.

Scroll bar

The scrolling commands are activated by moving the cursor into the scroll bar (left margin of a window) and clicking some combination of mouse buttons. In all cases, scrolling is activated when the mouse buttons are released. Moving out of the scroll bar before releasing the buttons returns you to text selection mode without repositioning the file. The thermometer in the scroll bar shows the current position of the window in the file. The positioning commands are as follows:

scrolling up [RED button]
moves the line next to the cursor to the top of the window.

relative scrolling [YELLOW button]
moves to the position in the file corresponding to the relative position of the cursor in the scroll bar (also called "thumbing").

scrolling down [BLUE button]
causes the line at the top of the window to be moved next to the cursor.

normalize selection [YELLOW and BLUE buttons]
causes the line containing the current selection to be moved next to the cursor.

Menu Commands

When the BLUE mouse button is pressed in the text area of a window, the **WindEx** menu appears and the cursor changes to a left arrow. Select a menu command by pointing at it (causing it to video reverse) and releasing the mouse button. If you do not wish to execute the command, release the cursor outside the region of the menu. In all cases (except where otherwise noted below) clicking the RED mouse button causes the command to be executed; clicking the BLUE mouse button resets the window to its previous state. After seeing the menu, if you do not wish to execute a menu command, just move the cursor away from the menu and release the BLUE mouse button. The menu commands are as follows:

Create

creates a new scratch window (that accepts keyboard input) at the place selected by clicking RED. Note that a maximum of four scratch windows may exist at the same time.

Destroy

destroys the window selected by moving the bullseye cursor into a window and clicking RED. Note that windows belonging to the debugger cannot be destroyed by the user.

Move

changes the position of the current window; the window sticks to the cursor as it is moved around. Clicking RED positions the upper-left hand corner of the window to the cursor location; clicking BLUE returns the window to its previous position.

Grow

changes the size of the current window; the lower right-hand corner of the window sticks to the cursor and the window turns gray. Clicking RED fixes the size of the window (subject to the minimum size restriction); clicking BLUE resets the window to its previous size.

Load

loads a file into the window selected by clicking RED, using the selection of the current window as a filename. Alternately, if you type a filename terminated by the escape character (ESC) into a scratch window, the window is automatically loaded with that file. Note that the user cannot load into the window containing DEBUG.TYPESCRIPT or any other windows belonging to the debugger.

Stuff It

takes the selection of the current window and stuffs it into the input stream of the window selected by clicking RED. The lower left function key on Alto II keyboards (FL4) stuffs the current selection into the default window (the debugger's typescript).

Find

finds the selection of the current window in the window selected by clicking RED. The search begins at the end of the selection of the window being searched. If the search is successful, the text becomes the new selection and is scrolled to the top of the window; otherwise, the selection remains the same.

Set Brk

uses the selection of the current window to set a breakpoint. If you select the word "PROCEDURE", a breakpoint is set on the entry to the procedure; if you select the word "RETURN", a breakpoint is set on the exit of the procedure; otherwise a breakpoint is set at the closest statement enclosing the selection. Confirmation is given by the selection being moved to the place at which the breakpoint is actually set. The window must contain the source file for a module in the current configuration; in the case of multiple instances of a module, the name of the current context must be the same as the source file.

Clr Brk

clears the breakpoint or tracepoint as specified above.

Set Trc

sets a tracepoint as specified above. Confirmation is given by the selection being moved to the place at which the tracepoint is actually set.

Set Pos

takes the selection of the current window as a character index and positions the file in the window selected by clicking RED to that character position.

Keys On/Off

activates/deactivates input from the keyset as described below.

When **WindEx** is actively working on a command, the cursor is in the shape of hourglass. When it is done with the current task, the cursor returns to its normal shape.

Keyset

The keyset can be used as an additional source of text input to **WindEx**. Keyset chords are described as octal numbers with **1B** corresponding to the rightmost key and **20B** to the leftmost. The keyset forms a chord by **OR**ing all keys depressed and returns the chord when all keys are released. *Be careful not to have any books on the keyset when it is active!*

Keyset Commands

The keyset commands involve stuffing characters and selections into the default window (which is the DEBUG.TYPESCRIPT window when **WindEx** is loaded with the debugger). In all cases, the default window is made the current window at the end of the command. The keyset commands are as follows:

1B stuffs the selection of the current window.

2B stuffs a carriage return (**CR**).

3B stuffs the selection of the current window followed by a **CR**.

4B stuffs an escape (**ESC**).

10B stuffs a delete (**DEL**).

20B stuffs a backspace (**BS**).

Keyset Input

Holding down the RED and YELLOW mouse buttons puts the keyset into text input mode. For each chord typed, the corresponding character is put into the input stream of the current window. The characters supported are **'A..'Z** (**1B..32B**), **'+** (**33B**), **BS** (**37B**). Attached is a label that can be taped to your keyset and used as a guide to the appropriate codes.

A	 x	J	. x . x .	S	x . . x x
B	. . . x .	K	. x . x x	T	x . x . .
C	. . . x x	L	. x x . .	U	x . x . x
D	. . x . .	M	. x x . x	V	x . x x .
E	. . x . x	N	. x x x .	W	x . x x x
F	. . x x .	O	. x x x x	X	x x . . .
G	. . x x x	P	x	Y	x x . . x
H	. x . . .	Q	x . . . x	Z	x x . x .
I	. x . . x	R	x . . x .	+	x x . x x

BS DEL ESC CR STUFF