Inter-Office Memorandum

To	Mesa Users	Date	October 17, 1977
From	Barbara Koalkin	Location	Palo Alto
Subject	Mesa 3.0 Debugger Update	Organization	SDD/SD

XEROX

Filed on: [MAXC]<MESA-DOC>MESADEBUGGER30.BRAVO

This release of the Mesa debugger introduces many changes of interest and importance to all Mesa programmers. Much work has been done to integrate the notion of configurations into the debugger and make this new information available to the user. The purpose of this memo is to make you aware of the changes that have taken place. More complete explanations may be found in the file <MESA-DOC>MESADEBUGGERDOCUMENT.EARS.

Configurations: new commands and changes to existing commands

The major impact of the new configuration scheme on the debugger is to allow the user to more closely specify the scope of the debugger's symbol search, thus saving on lookup time. The notion of the *current context* has been expanded to include the *current configuration* in addition to the *current module* and its corresponding frame within that configuration. The current context is used for determining where the debugger is to begin its symbol lookup. This applies for displaying variables as well as setting breakpoints, calling procedures, etc. Upon entering the debugger for the first time, the current configuration is the last configuration that was loaded and the current module is set to be its control module (if it exists) or else the first module of the configuration.

Note that in all cases an instance name is required when a configuration or module name is not unique.

CUrrent context

lists the name of the current module and its corresponding global frame address (and instance name if one exists) followed by the name of the current configuration.

List Configurations

begins with the last configuration loaded and lists the names (and instance names, if they exist) of all configurations that are loaded.

Display Configuration

lists the name of the current configuration followed by the names and the corresponding global frame addresses (and instance names) of all the modules in the current configuration.

SEt Root configuration *configname*

sets the current configuration to be *configname*, where *configname* is at the outermost level of its configuration. This command will be sufficient for simple configurations of only one level. It is also useful in getting you to the outermost level of nested configurations from which you may move "in" using the following command.

SEt Configuration *configname*

sets the current configuration to be *configname*, where *configname* is nested within the root configuration that is current. This command is useful for "jumping" further into the nested block structure of a configuration.

SEt Module context *modulename*

sets the current context to be *modulename* (which must be contained in the current configuration). Note that **SEt Octal Context** will continue to function as before and will update the configuration/module context as well as changing the current frame.

Reset context

restores the context which this instance of the debugger had upon entering for the first time.

Display Variable *var*

searches for the value of *var* but now limits the scope of its search to the current context. The lookup stops at the end of the call stack, whereas it previously continued its search by following the binding path (which no longer exists). It continues to look aside to global frames as well, but *the search is limited only to the current context.*

Find Variable *var*

looks through the *whole current configuration*, not only the current context, to find the value and module location of *var.*

Display GlobalFrameTable

displays each module name and its corresponding octal frame address, pc, codebase, and gfi.

String Utility Commands

Interpret STring *stringname startindex n*

begins with *startindex* and lists *n* (decimal) characters from *stringname.*

Ascii read *address n*

displays *n* (decimal) characters starting at *address.*

Note that null characters are ignored by the display software, and therefore do not appear. Control characters print as their corresponding alphanumeric preceeded by ↑.

Changes to Existing Commands

The **Bind** command has been taken out of the debugger completely, thus allowing the key letters for **Break Entry** and **Break Xit** to be simplified. The New command and the **STart** command will remain as before with the exception that **New** will no longer run any code.

Extended Capabilities

Esc buffers

The debugger tries to cut down on the amount of information that must be typed-in by saving default values that may be used or inspected by typing ESC. The following additional parameters are now being saved: configname, stringname, stringindex, and ascii read address.

More breakpoints - breakpoint indicator <>

It is now possible to set breakpoints on statement boundaries instead of just text line boundaries. The *indicator* <> will appear to the left of the source where the breakpoint is set, ie. IF foo THEN <>some statement; .

Small things you will notice

There have been improvements in the display of small records (and small fields of records) and the display of catch frames. In coordination with the language feature introducing PACKED ARRAYS, the debugger will print them as well. You will also notice stronger version checking when examining definitions modules.

If you use the optional window manager that is supplied with the system, you will notice that the current window no longer changes when the mouse is repositioned. To bring a window "on top" and make it current, position the mouse over it and click one of the mouse buttons. This makes it a bit more difficult to find "lost" windows, but there is less chance of a window disappearing accidently.

Documentation

The debugger documentation is now separate from the system document and can be found on <MESA-DOC>MESADEBUGGERDOCUMENT.EARS with an explanation of how to install the debugger and how to use the window manager found in Section 5.