Inter-Office Memorandum

To	Mesa Users	Date	October 17, 1977
From	John Wick	Location	Palo Alto
Subject	Mesa 3.0 System Update	Organization	SDD/SD

XEROX

Filed on: [MAXC]<MESA-DOC>MESASYSTEM30.BRAVO

This memo outlines changes made in the Mesa system code since the last release (May 13, 1977).

Names in square brackets refer to sections of the *Mesa System Documentation* which have been updated. More details can be found there and in other documentation accompanying this release.

Binding

Modules (and configurations) are now bound as they are loaded; a separate binding call is no longer required.

BitBlt

Since all Altos have been converted to microcode version 23, we have removed software BitBlt from the system.

Configuration Instantiation

Because the language does not yet support the NEW operation on configurations containing more than one module, the system procedure FrameDefs.New must be used. It takes a filename (which usually ends with ".bcd.") and returns a GlobalFrameHandle; the result should be LOOPHOLEd into a PROGRAM or a POINTER TO FRAME of the appropriate type.

Definitions

Because the new binder requires all procedures and signals to appear in definitions files, several new modules have been added to the system. Those of interest to users include BFSDefs, DirectoryDefs, FrameDefs, MiscDefs, and TrapDefs. [Section 2]

Disk Streams

Procedures have been implemented which create byte or word streams directly from file names and access options, rather than from FileHandles. A ModifyIndex procedure is available for adjusting a StreamIndex forward or backward. [Streams]

Free Storage Package

Blocks added to a zone are now linked together, and an interface has been provided which will automatically release blocks containing no allocated nodes (by calling a user-supplied procedure). The procedure PruneHeap must be called explicitly to accomplish this for the system's free storage heap. [Storage]

Interrupts

EnableInterrupts and DisableInterrupts have been implemented in microcode. Definitions of these instructions have been added to ProcessDefs. Note that EnableInterrupts now guarantees that (at least) one instruction will be executed following the enable opcode before the current process is suspended. [Process]

To allow the keyboard process (and others) to run preemptively during code swapping, interrupts are no longer turned off while a code trap is being processed. For this to work, *all* code which runs at interrupt level (i.e., preemptively) *must be locked in memory.* You can use the procedures LockCode and UnlockCode defined in FrameDefs.

KeyStreams

It is now possible to change the character code assignments for the keyboard, keyset, and mouse. Cursor tracking can also be selectively disabled. Optionally, a procedure can be supplied which will be called whenever the current KeyStream is waiting for input. [Streams]

Loading

The sequence of commands used to load a program (formerly New, Bind, Start) has been shortened to New, Start. While the parameters and results of these commands remain unchanged (for single modules), there are two important differences.

The New command does not execute any code; Start causes the initialization code and the main body of the module to be executed (there is no implicit STOP inbetween). The net effect is that the command sequence contains the same number of starts. (This also applies to NEWS and STARTS contained in programs. See the *Mesa Language Manual* for more details.)

The loader has been extended to handle configurations containing several modules which have been prebound by the the new Mesa binder. (This allows all of the configuration's global frames to be allocated compactly, without breakage.) If the modules require a specific initialization sequence, the configuration must include a CONTROL module. The global frame of the control module is returned by the New command; when it is started (with the Start command), it should start other modules of the configuration. If no control module has been designated for the configuration, the New command will return a null global frame. In this case, modules will be started only as a result of taking a start trap (see below).

MakeImage

Facilities have been added for processing configuration descriptions. The symbolsToImage option of MakeImage is now a noop; all packaging of symbol tables should be controlled by the binder. The definitions of cleanup procedures have been moved from NovaOps to ImageDefs. (In anticipation of conversion to the Dstar, all dependencies on NovaOps should be removed.) Machine codes for StopMesa, AbortMesa, and PuntMesa have been added to ImageDefs. [Images]

Start Trap

A trap will now occur when a transfer to a module which has not been initialized is attempted. The trap handler will start the module, first checking that its control module (if any) has been started; it will then complete the transfer. Note that modules which are started by the trap handler cannot take parameters or return results; a control module must perform initialization in this case.

UnNew

A kernal function which deletes a module instance has been implemented; it is defined in FrameDefs. If there are no other instances of the module in existence, the module's code (and symbols) are released. Note that this procedure does not check for other modules bound to the one being deleted. *Beware of dangling references!*

Unsigned Compare

The unsigned compare instruction (USC) has been removed from the microcode. A procedural implementation has been provided; it is defined in InlineDefs.

Internal Changes

The following changes are internal to the implementation and do not affect user's source code. They may affect performance, however.

Allocation Trap

The allocation trap handler now takes a single parameter: the original destination of the transfer. In the case of a local procedure call, the microcode fabricates a procedure descriptor from the entry point number and the information in the current global frame.

Code Segment Prefix

The format of the second word of the prefix proceeding the entry vector has been revised so that it describes all external transfer types in the global frame. This allows signals and frames to be bound just like procedures. It also enforces the restriction of no more than 127 procedures per module.

Global Frame Table

The current length of the global frame table is now kept in an entry of the system dispatch table. This allows systems to be built with GFT's of less than the maximum size (currently, 511 entries). The standard Mesa configuration allows for 255 entries.

Kernal Function Calls

Several new kernal functions have been added as a result of other changes. Most entries of public interest are now defined as inlines in definitions modules (e.g. FrameDefs).

Procedure and Signal Descriptors

The fields of procedure descriptors have been rearranged to produce better code in generating them. The internal value of a signal has been changed from an address to a structure paralleling procedure descriptors (a global frame index and "entry" number). The values of signals are no longer stored in the global frames of the modules defining them.

Signaller

The signaller has been recoded to improve its execution time. Processing a signal has been speeded up by a factor of from two to nine, depending on the disposition of the signal (reject, unwind, or resume).

Static Link

To achieve a better instruction mix, the static link of local procedures and catch frames now points to the first local variable of the enclosing frame, rather than to the beginning of that frame.

Distribution:
Mesa Users
Mesa Group