A TEMPORARY UPDATE FOR LISP USERS

This short system note is intended to fill the information vacuum since the publication of A.I. memo 4157 in March 1968. In addition to removing many bugs and inadequacies, I have added some new features which are described on the succeeding pages. Some are in the current LISP, all are in NLISP.

On about Dec 1 1968 NLISP will become the standard system. When this happens, "NEW LISP" will go away—use henceforward either E-LISP or C-LISP.

NOTE WELL! Items 4) and 7) below document changes not upward compatible in RECO with old argument and multiplication usage.
The "E" feature of the MIDAS linking macros does not operate as advertised, and furthermore is superfluous. A new macro "EXECUTE" (option #6) is installed, which signals a closed subroutine (exited by a POPP P,) to be executed once immediately following the completion of the linking of the subprogram in which it occurs. This allows the MIDAS user to do automatically a little post-allocation initialization, which would include, but is not limited to, changing some code of the already loaded LISP system.

2) ERRSET now works as advertised: errset [S] is list [eval [S]] if no error occurs while evaluating S and is NIL otherwise. However, if the function ERR is called while evaluating S, then the error recovery routines are entered (but not error-message is printed on the console); and if ERR is given an argument, then this argument is returned by ERRSET instead of NIL. If a second argument is given to ERRSET, then the error-message print-out is suppressed during the evaluation of the first argument.

Whenever an error occurs in an evaluation under the control of ERRSET, LISP is re-initialized (as much as is possible) including the resetting of the I/O switches. Since there is no A-list in MIDAS-LISP, a policy attempt is made to restore the value cells of those atoms having higher-level bindings (see LISP write-up, get, set, specific value cell, variable bindings). Just after the re-initialization is completed, and just before the main top-level loop is re-entered, the following evaluation takes place: eval [ERRSET]. Thus ERRSET provides a means for a user to add his own 24 month to the re-initialization process.
Some errors are of such a drastic nature that `ERRSET` cannot be guaranteed to recover from them, and they always translate to the re-initialization:

(a) PDL or Special PDL overflow
(b) No free storage or full word space left
(c) LG typed as console. LG typed is a quit signal which is caught by `ERRSET`; LG is unconditional.
(d) Illegal instruction execution. A typical bug leading to this error is a REPLACD which clutters some part of the system.

3) `DELETE` (and `DELR`) has an optional third argument which is taken as an upper limit on the number of deletions to be made. `DELETE [a; b]` deletes only the first occurrence of `a` in the list `b`, while `DELETE [a; b; lo]` deletes the first `lo` such occurrences. `DELETE` works by copying or REPLACING, and returns a pointer to the deleted list.

4) `READ` (and `READCH`) has an optional argument, in order to allow end-of-file detection. The ordinary action upon reading a `LE` (the EOF character for mac tape) on an auxiliary device is to close the input file, reset the job console for input, and wait for an expression to be typed. However, `READ [IP]` will close the file and return `IP` as the value of `READ`; if no EOF is encountered in reading the next expression of the input file, then `READ [IP]` functions just like `READ [I]`.

5) The character `<alt mode>` does not need a slash preceding it to be read in by `LISP`; but on print-out a slash will be inserted to distinguish it from `<dollar>`. The characters `*` and `-` may be used like any other single-character atom. Note: `-5` and `+A` are numbers while `/5` and `4A` read-in as alphabetic atoms.
6) UREAD, UWRITE, and UFILE have default options in lieu of arguments, just like TECO. However, if UREAD is given any arguments, the first two are interpreted as filename1 and filename2; if a third is present it is device name; and if a fourth is present it is user name. Similarly, if UWRITE has any arguments, the first is taken as device name and the second, if present, as user name.

7) The method of opening and closing the A/D and D/A multiplexer has been changed: Two arguments are now required, the first for the A/D input multiplexer and the second for the D/A output multiplexer. In each case, the meaning of the argument is as follows:
   NIL - leave open-close status unchanged
   0  - close device
   1  - open device in normal mode (image)
   2  - open device in fast mode (ASCII)

8) NFIX can also read in block mode: nfix[...anymame2...] will assume, for 0≤i≤n, that anyname2 has an x-value and anyname2+i has a y-value, and will replace the latter by nfix[i;y]. LISP numbers are used both coming and going, but generally small numbers will suffice.
   One can write a small SAP routine to fill the first n words of some array for input to NFIX, if the points to be read are related in some reasonable fashion. In each case the left-half holds the x-value, the right-half the y-value, and the return value is in the right-half.
   NSET is used to set the conditions for NFIX, regardless of which mode is used. An error occurs if the first argument to NFIX is neither an array nor a number.
Some Display Features.

The 340 Display may be used in T.S. LISP as an ordinary output device, just like the line printer, the plotter, UTAPE and disc devices, and the teletype. In turns on and turns off this feature. Of course, one cannot simultaneously and at the same time be outputting to the display and displaying from a DISLIST (by means of IF).

DISLIST is used as before, except that the relevant information is found as an *array* property instead of "SUBR". The system tries to be clever in order to avoid display memory protect violations, regardless of how the loser loses his arrays.

10) Lu is no longer used to halt a plot started by IPL or by plot[0], but only for halting the top level output to the plotter (as LE is used for the line printer). The careful plotter-nik will say plot[zo0] to close his schemes.

11) The new disk units are available for I/O usage as device Dsk. Alternately one can use PKn to write on a specific disk pack.

12) Accumulator 13 is not used or referenced by LISP. One may safely design obscure uses for it with LISP and MPPAS sub-programs.

13) The length of the OBLIST may be prescribed at assembly time with minimal effort. The length of the OBLIST in LISP's with version number > 91 is set to 777. If you have slow reads because of atom verbosity, contact John.

14) NOUPO, NOCHK, NORET, BAKGAG, GCTWA, *NOPOINT, *RESET all set accessible switches, namely their own special value cells. Thus itochk[0] accomplishes the same as set [Nock; e].

There are a number of global variables critically pertinent to system operation—a brief catalog follows on the next three pages. This is primarily for convenience since nothing essentially new is documented therein.
**ATOM**

**BASE**

Ready to be used when printing out fixed point numbers. Used by PRINT

**GCMKL**

cml [GCMKL] is a list of pairs, each pair a special array cell and the number of cells that the corresponding array occupancy

**CHRCT**

Number of character space remaining in current output line, PRINT enforces a minimum output line length of LINES (q.v.), and when CHRCT reaching zero, a carriage return is emitted and CHRCT reset to LINES

**BPROC**

BRENV.

When the system is initially loaded, all memory from BPROC up to the top of the core allotment is available as a kind of superuser storage (not part of Free Space or Full Used Space). Subroutines loaded by LAC are generally stored in the lower end of the space, and Arrays are dynamically stored at the higher end. BPROC is updated by LAC to indicate the lowest currently available cell of this space, and BRENV is updated by the array handler and garbage collector to indicate the highest currently available cell.

**DISCNT**

When DISANO is used to prepare an array of Text for 370 Display, DISCNT serves the same dynamic function for pseudo-output to the 370 as CHRCT does for regular output, say to the line printer.

**FSLP**

The number of carriage returns inserted by DISANO since the last DISCNT call after the
number of lines of text inserted in the array to be displayed.

AISLIST

Our means of displaying information on the 390 display screen — the creation of "display arrays" — each such array is a sequence of instructions to the 390 (see manual for usage). AISLIST is a list of the special array calls of most display arrays we want to run. In Time Sharing version, the display is not actually viewed until the IF Switch is on (OFF).

ERLUST

When a non-ERLUST error is detected, control passes to an initialization routine, just be for re-entering the top level main loop. Among other things, [EVAL, ERLUST] is performed, thus allowing the user to add his own test calls to the error recovery process.

IBASE

radix to be used; when converting a number-like string on input. May be any value from 1 to 36, except that any number-like string having non-decimal digits must begin with either "+" or "-". Examples:

IF IBASE is 8, then "47" is read in as 39.
IF IBASE is 12, then "47" is read in as 16.
IF IBASE is 16, then "+1A" is read in as 27.

"1A", "+1A", or are illegal, or strings:
While "+", "-", "A1", "B-2", "1A": are acceptable.

Herein, "x" and "y" are user as meta characters.
Typing LN (control-x) from the console, or (more ill)
evolving a in [x] sets the x-switch to ON.
The x-switch is merely the value of the x-switch
Type (variant x): EA- x-switch x (control-x) has a corresponding control character when turned off.
A

meaning of suite
not used by system

B

turn on line printer to receive PRINT output

D

turn on output of garbage collector statistics (one set of values from collection)

F

enable displaying from display arrays of BFSLIST

N

in time sharing, turn on the DIS device to receive PRINT output. (Physically, the DIS device is the same as the 3 teletype, and one cannot use them simultaneously)

P

Turn on the plotter to receive PRINT output.

Q

Let READ receive its input from the currently selected input file, rather than the job console

R

enable the correctly output file to receive PRINT output.

W

disable the job console from receiving PRINT output
15) **Relocatable Arrays**

Arrays are stored on the property list of atoms as the "array" property. Instead of "subr" property, thus arrays are automatically garbage-collected when nothing in the system points to them. Array space is the area between bporg and the top of core, and bporg is reset from time to time reflecting the varying amount of space required. Calling "(getsp n)" will insure that there is at least n free cells between bporg and bpend. [Bporg holds the address of the first cell currently available for binary programs — bpend holds the current address of the last cell.] The function "array" is used as before, but all entries are zeroed automatically. To remove an array without removing its atom header, use "remq" in a fashion similar to "remq". Both "array" and "getsp" will return nil if the requested space just cannot be obtained.

All references to arrays now are checked to see that the cell accessed is indeed a part of the array mentioned. Thus mistakes such as storing over the end of the array, or cluttering some code in bporg area, are prevented. This feature takes very little extra time, but may be inhibited by executing "(nochk t)" and disinhibited by "(nochk nil)".

16) **Garbage Collector goodies**

The garbage collector has a few other little goodies in addition to the array relocating features: a truly worthless atom (twa) is defined to be one with a trivial property list and which is pointed to by nothing alive in the system except the oblist. An internal switch, the "gctwa" switch, if non-nil will cause the garbage collector to remove all twas from the oblist. Executing "(gctwa s)" will set the switch to the value of s; since this feature adds about 15% in time to a gc, the normal mode of the switch is nil. As a variant, executing "(gctwa)" will force the twa removal phase to happen during the immediately ensuing garbage collection.

Whenever the gc finds that bporg is so much less than bpend that two or more blocks of core could be returned to the time-sharing system, then it will do so, first compactifying arrays and relocating them downwards. "(nochet t)" inhibits this feature, which is disinhibited in the usual way.

17) **Using Lisp print while in dot**

When debugging Lisp programs with dot, often sees a pointer typed out, instead of the beautiful fully-parenthesized style of print output. Typing P $x$ will cause dot to enter a part of Lisp which prints out the currently open cell in LISP format, and return to dot when done. [Actually, "$p" is a symbol in the job symbol table equal to "print", P$SYM"]
A quick look at some new LAP features

(a) LAP now communicates with the DDT symbol table for the LAP job in which it is running. Thus all operation codes are defined in LAP by means of the DDT op-decoder. The "symbols" pseudo-op, described in A.I. memo #152, Pop-6 LAP, page 4, works quite well now.

(b) If there are any undefined referenced symbols in a LAP assembly, then the DDT symbol table is interrogated to try to make a definition. If a value is successfully pulled back from DDT and used, a warning message is printed: "DDT symbols" followed by the ones found. Otherwise, a message "undefined symbols" is printed followed by the recalculating ones.

(c) In both the above-discussed features, if a LAP symbol has characters not legal for MIDs symbols, then those characters are converted to "o".

(d) If a non-diagnostic error occurs during assembly, LAP records as much as possible and aborts assembly. If the assembly occurred while reading from an auxiliary device, then the rest of the LAP subroutines is skipped over so that other routines may assemble successfully.

(e) REmLAP works well, needs no argument, and performs no dangerous relocations. One may even re-load LAP after remapping.

(f) Three new pseudo-ops are installed:
   BLOCK n   assembles a block of n zeros
   (ASCII SF) does a PRINT of the S-expression of with ASCII characters, just as MIDS directly does.
   (SIXBIT SF) Diffs, but with sixbit characters
The ASCII and SIXBIT forms may be used as assembly word constituents, in which case they point to the first location of the ASCII or six-bit string. For example:

(NL
    MOVEI 1 (ASCII (LIST)))

would assemble as

MOVEI 1, +1
ASCII (LIST)

With reasonable allocation for free storage and full word space, LISP should assemble about 40 to 70 instructions per second of CPU time.

19) PATOM is a new routine which returns non-nil only for pointers into the free storage area which are not pointing to stores. Using PATOM, routines such as MEMBER, EQUAL, SUBST, etc. no longer get hopelessly tied-up when a list has some pointers out of the free storage area.

20) If an auxiliary device runs out of space while a file is being written on it by LISP, then LISP prints on the console an appropriate comment and allows the user the same recourse as with TECO.