MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PROJECT MAC

A.I. Memo. No. 116A.
(Corrected edition Memo.116)

April, 1967.

PDP-6 LISP (LISP 1.6) Revised.(January 1967).

PDP-6 LISP (LISP 1.6)

This is a mosaic description of PDP-6 LISP, intended for readers familiar with the LISP 1.5 Programmer's Manual or who have used LISP on some other computer. Many of the features, such as the display, are subject to change. Thus, consult a PDP-6 systems programmer for any differences which may exist between LISP of Oct. 14, 1966 and present LISP on the system tape.

SOME DISTINCTIVE CHARACTERISTICS

Top-level input is to EVAL; there is no EVALQUOTE.
Thus LISP's listen loop may be described by

```
(PROG NIL
     A      (TERPRI)
            (PRINT  (EVAL (READ)))
            (GO A))
```

One types ATOM followed by a space to get the value of ATOM, or (FN (QUOTE arg1) (QUOTE arg2)...) to apply a function to explicit arguments. For this reason, users frequently arrange their top-level functions to be FEXPRs or FSUBRs to avoid the necessity of quoting explicit arguments. Alternatively, they may SETQ atoms to frequently-used or unwieldy argument lists, and apply their functions to these atoms.

(EQUAL fixed-point-number floating-point-number) is NIL.
(ZEROP 0.0) is NIL. (EVAL T) is T. (EVAL NIL) is NIL.
T and F do not exist.

There is no a-list. Current bindings of interpreted variables and free compiled variables are stored in the CDR of the variable's VALUE property, which is also the variable's SPECIAL cell.
Their values may be changed with SET and SETQ; and CSETQ do not exist. Higher-level bindings of these variables are stored on the special pushdown list.
All interpreted variables and free compiled variables are automatically SPECIAL and provide complete communication between compiled and interpreted functions. COMMON does not exist.

Flags are not allowed; elements on a property list of an atom are expected to be paired.

MAP, MAPCAR, etc. assume the first argument is the function, and the second is the list of arguments to which it is to be applied.

There is no DEFINE; defining of functions is usually done with DEFPROP.

The available input-output devices are the on-line teletype, the DECtapes, the vidisector, the display and the line printer. For their use, see "Input-output".

One may allocate as much as one wishes (within limits) of each type of LISP storage each time LISP is started. See "Allocator" for how to use this feature.

The characters space, tab and comma are treated identically and are used to separate items on a list, and to distinguish dot-notation periods from periods denoting floating-point or decimal numbers. A sequence of spaces, commas, or tabs is equivalent to a single one. Spaces, commas, or tabs adjacent to parentheses have no effect.

Object	Indicator	Relevant Information
ABS	SUBR	Absolute value.
ADD1	SUBR	Result is fixed or floating, same as argument.
Allocator		There are five storage areas in LISP: a/ Free Storage: holds s-expressions b/ Full Word Space: holds character strings of print names, floating point and large fixed-point numbers (see "Numbers"). c/ Binary Program Space: holds compiled functions and arrays. d/ Special Pushdown List: holds higher-level bindings of all special variables. e/ Regular Pushdown List: holds return addresses for sub-routine calls, bindings of all local variables, and is also used by various internal routines.

When LISP is loaded, the Allocator types out ALLOC? and waits for the user to type Y (yes) or N (no). If Y is typed, the Allocator responds with MEMTOP= and waits for the user to type an octal number (ended with a carriage return) which LISP will take as the highest avaiable register of memory. The Allocator then similarly requests typed-in parameters for the size of Full Word Space. Binary Program Space, Special Pushdown List, and Regular Pushdown List. Free Storage is given all remaining space. For any typed-in number, carriage return alone may be typed in and a standard value will be taken:

MEMTOP	3717Ø (hopefully this will	
FULL WDS	4ØØ	change soon)
BIN.PROG.SP.	1ØØØ	
SPEC.PDL	1ØØØ	
REG.PDL	1ØØØ	

If N is typed as the response to ALLOC? all standard values are taken. In case of error, RUBOUT will type an X and cancel the current value. The Allocator is wiped out by the first garbage collection; thus a restarted LISP may not be reallocated.

AND	FSUBR	Returns NIL or last non-Nil argument; does not evaluate arguments past first NIL.
APPEND	LSUBR	Appends together any number of arguments, copying the top level of all but the last of them. (APPEND) is NIL and (APPEND X) is EQ to X.
*APPEND	SUBR	APPEND of two arguments.
APPLY	LSUBR	(APPLY fn (args)) or (APPLY fn (args) alist).
ARRAY	FSUBR	(ARRAY name par dim1 dim2 ...) sets up name as an array (actually as a SUBR). par should be T to protect array elements from garbage collection; NIL otherwise. (name indx1 indx2 ...) returns the value of the specified element. (STORE (name indx1 indx2 ...) value) sets the element to value.

(NSTORE (name indx1 indx2 ...) number) deposits the low-order 18 bits of the number in the array. Both STORE and NSTORE evaluate their second argument before their first. An array may have no more than five dimensions; indices run from Ø to dim-1.

ARG SUBR See LSUBR.

ASSOC SUBR Uses EQ.

BAKGAG SUBR (BAKGAG T) enables a backtrace of any LISP error except pushdown list overflow; (BAKGAG NIL) disables it. A backtrace is printed as a series of function calls as determined from the regular pushdown list, most recent first:

fn1-fn2	fn1 calls fn2
fn1-EVALARGS	arguments being evaluated preparatory to calling fn1
fn1-ENTER	fn1 being evaluated
?-fn1	? represents an internal routine

BASE VALUE Current radix of fixed-point number output; may be modified with SETQ.

Binary Program Space Storage area in LISP for compiled functions and arrays; usually is 1ØØØ registers long.

BOOLE LSUBR (BOOLE n a b c ...) causes a 36-bit bitwise Boolean operation to be performed on a and b, on the result and c, etc. The number n selects the operation as follows:

n	operation	n	operation
0	0	10	$\overline{a}.\overline{b}$
1	a.b	11	a≡b
2	$\overline{a}$.b	12	$\overline{a}$
3	b	13	$\overline{a}$ vb
4	a.$\overline{b}$	14	$\overline{b}$
5	a	15	a v$\overline{b}$
6	a v$\overline{b}$	16	$\overline{a}$ v$\overline{b}$
7	a vb	17	1

BPEND VALUE Highest location available for use as Binary Program Space.

BPORG VALUE Current lowest unused location in Binary Program Space.

CAR SUBR Left half. CAR of an atom is the "pointer" - 1.

CDR SUBR Right half. CDR of an atom is its property list.

CAAR, CADR etc. through CDDDDR

SUBRS any combination of up to 4 D's and A's.

CHRCT VALUE Number of character spaces remaining in current output line. If CHRCT is Ø and LISP is to output a character, it first outputs a carriage-return--line-feed and resets CHRCT to LINEL (q.v.).

Compiler The Compiler is a collection of SUBRs not available in regular PDP-6 LISP, but rather in a special version on the System tape as the file LISP COMP. To compile functions from DECtape, load the file LISP COMP, type (UREAD name1 name2) on tape number tapeno, select DECtape output if desired, not forgetting to specify an output tape with (UWRITE tapeno), type (CMP1) and then select DECtape input. (See "Input-Output"). When the Compiler is finished, file the output if DECtape output was selected. The Compiler outputs each function in LAP-readable machine language. (See "LAP").

COND	FSUBR	"COND pairs" of predicate and value may have other than two elements. If the predicate evaluates non-NIL, the remaining elements are evaluated in CAR-to-CDR order, and the value of the last is the value of the COND. If there is only one element, its value, if non-NIL, is the value of the COND. If no predicate is true, the value of the COND is NIL.
CONS	SUBR	One of the five elementary functions of LISP.
Current Problems		Compiled GOs and RETURNs happen when encountered; interpreted GOs and RETURNS happen when evaluation of the functions they are contained in (except PROG) is completed. Arguments to GO are repeatedly evaluated until atomic when interpreted, but are evaluated only once when compiled. Compiled GOs and RETURNS must appear explicitly in the PROG to which they apply. GOs and RETURNS occasionally confuse the compiler when appearing in compositions of functions other than PROG, COND, AND, OR, and NOT. Variables used free in a functional context must be declared SPECIAL or else the compiler will mistake them for undefined EXPRs or SUBRs.
DECtapes		To select DECtape input, specify a file by using the function UREAD, then type /Q or evaluate (IOC Q) or set the value of ↑Q to T. To deselect DECtape input, type /S or evaluate (IOC S) or set the value of ↑Q to NIL. To select DECtape output, specify an output tape with UWRITE, type /R or evaluate (IOC R) or set the value of ↑R to T. To file output use UFILE. To deselect DECtape output, type /T or evaluate (IOC T) or set the value of ↑R to NIL. See UFILE, UKILL, UREAD, UWRITE.
DEFPROP	FSUBR	(DEFPROP atom property indicator); value is atom.

DEPOSIT	SUBR	Causes the fixed-point number which is its second argument to be deposited in the absolute machine location which is its first argument.
DIFFERENCE *DIF	LSUBR SUBR	arg1 - arg2 - arg3 ... (DIFFERENCE) is Ø. DIFFERENCE of two arguments.
DISAD	SUBR	See "Display".
DISCNT	VALUE	See "Display".
DISINI	SUBR	See "Display".
DISLIST	VALUE	See "Display".

Display

The display is different from other output devices in that PRINT and related functions cannot send outputs to the display. Instead, text is displayed from LISP arrays which are on the DISLIST. To place text on the display, first perform (DISINI (QUOTE arrayname)) which initializes element Ø of the array to a character mode entry and the remaining elements to escape characters, destroying the previous contents of the array. Put this array on the DISLIST with a (SETQ DISLIST (LIST GET (QUOTE arrayname) QUOTE SUBR))). More than one array can be put on the DISLIST, but if they have all been DISINI'ed, they will overprint on the display. To avoid this, or to use the display in other than character mode use NSTORE (q.v.). To put text in an array, (DISAD(QUOTE arrayname) par arg) in effect performs a PRIN1 or PRINC (according as par is non-NIL or NIL) of arg into the specified array. In other words, the text of an s-expression is appended to the text currently in such an array. DISCNT is for the display what CHRCT is for all other output devices; when it reaches zero, DISAD inserts a carriage-return-line-feed in the text being put in an array and resets it to LINEL.

EQ	SUBR	Will work for atoms and positive fixed-point numbers less than about $4ØØØ_{1Ø}$; otherwise use EQUAL.

EQUAL	SUBR	Floating point numbers are EQUAL only if they are exactly equal. To compare a fixed to a floating--point number, first float the former by performing(PLUS fixed-point number Ø.Ø).
ERR	LSUBR	Takes 0 or 1 argument. Causes a non-printing LISP error condition: the argument of ERR (or NIL, if there is none) is returned as the value of the most recent ERRSET.
ERRSET	FSUBR	(ERRSET (fn arg ...)) has the value NIL (but see ERR, above) if an error occurs while evaluating (fn args ...), and the value of (list (fn args ...)) otherwise. Optional second argument T inhibits error message printout.
EVAL	LSUBR	If a second argument is given it is used instead of the present "p a-list". (EVAL number) is that number. In entering a function, arguments are evaluated from left to right. (EVAL atom) is (CDR (GET (QUOTE atom) (QUOTE VALUE))), unless there is no VALUE property or the above computation returns the pseudo-atom UNBOUND, in which case a LISP error occurs and the offending atom is printed, followed by the message "UNBOUND VARIABLE - EVAL." (EVAL x) where x is non-atomic and (CAR x) is atomic proceeds as follows: the interpreter determines (GETL (QUOTE CAR x) (QUOTE(SUBR EXPR FSUBR FEXPR LSUBR MACRO))), or, if this is NIL, (EVAL (CAR x)). If x has no VALUE property, a LISP error occurs and x is printed with the message "UNDEFINED FUNCTION." Having found a function, it then evaluates its arguments if appropriate and applies the function to them. When (CAR x) is non-atomic it immediately evaluates the list of arguments i.e., (CDR x). It then compares (CAAR x) to LAMBDA, LABEL and FUNARG, taking appropriate action in each case. Failing this, it applies (EVAL (CAR x)) to the members of (CDR x).

EXAMINE	SUBR	Argument is number, which is taken as absolute machine address; value is contents of said address as a fixed point number.
EXPLODE	SUBR	Argument is s-expression; value is list of atoms whose print names are single characters, which concatenated would form the print name of the argument. For example, (EXPLODE (QUOTE FOO)) has the value (F 0 0). EXPLODE like PRIN1, inserts slashes, so (EXPLODE (QUOTE (QUOTE FOO/BAR)) PRIN1's as (F 0 0 // / B A R) or PRINC's as (F 0 0 / B A R).
EXPLODEC	SUBR	EXPLODEC is to EXPLODE as PRINC is to PRIN1. Example: (EXPLODEC (QUOTE FOO/BAR)) would PRIN1 as (F 0 0 / B A R) or PRINC as (F 0 0 B A R).
EXPR		Indicator for s-expression function.
FEXPR		Indicator for s-expression special form. Its lambda-list may have one or two members; the second is bound to the current "a-list."
FIX	SUBR	Argument is a fixed or floating-point number; value is truncated fixed-point value of argument.
FIXNUM		Indicator for a fixed-point number. See "Numbers".
FIX1A	SYM	(PUSHJ P FIX1A) Turns the actual fixed point number in 1 into a LISP number, q.v.
FLATSIZE	SUBR	(LENGTH (EXPLODE arg))
FLONUM		Indicator for floating-point number. See "Numbers".
Free Storage		Storage area in LISP for s-expressions. Takes all memory which is free after the four other storage areas have been allocated. See "Allocator".

FSUBR		Indicator for special-form subroutine; the next pointer on the property list points to the first instruction in the routine. To use the current "a-list" a FSUBR must call *AMAKE, which leaves a pointer to same in 2.
Full Word Space		Storage area in LISP for character strings of print names, floating-point numbers, and negative fixed-point numbers, and large positive fixed-point numbers.
FUNARG		One of the three indicators the interpreter looks for when evaluating a non-atomic s-expression with a non-atomic CAR.
FUNCTION		To the interpreter, has the same effect as QUOTE. To the compiler, causes compilation of its argument.
*FUNCTION		Causes FUNARG binding when encountered; otherwise identical to FUNCTION.
GC	SUBR	Takes no arguments, causes a garbage collection, and returns NIL. To cause typeout of garbage collector statistics, type /D To inhibit this typeout, type /C
GENSYM	SUBR	GØØØ1, GØØØ2, etc.
GET	SUBR	(LAMBDA (A B) (COND((NULL (CDR A)) NIL) ((EQ (CADR A) B) (CADDR A)) ((GET (CDDR A) B)))) A typical use. (GET (QUOTE atom) (QUOTE indicator)), would return the property if it was found or NIL. If the property could be NIL, an amiguity could result in the second case.

GETL	SUBR	Similar to GET but second argument is a list of indicators. The value returned is either NIL (indicating no property with any of the desired indicators was found) or such that (CAR (GETL ...)) is the indicator, (CADR (GETL...)) is the property, and (CDDR (GETL ...)) is the rest of the property list. GETL like GET stops at the first satisfactory pair on the property list.
GO	FSUBR	If argument is non-atomic, it is repeatedly evaluated until it is atomic. However, see "Current Problems".
GREATERP	LSUBR	Takes any number of arguments; returns T only if arg1 > arg2 > arg3 . . .; returns NIL otherwise. GREATERP of two arguments.
*GREAT	SUBR	
IBASE	VALUE	Radix of fixed-point number input. May be modified with SETQ.

Input-Output

The input-output devices available to LISP are the on-line Teletype, the DECtapes, the vidisector, the display, and the line printer. Input from the Teletype and the DECtapes is through the function READ; typing /Q or evaluating (IOC Q) or SETQ'ing the atom ↑Q to T will cause DECtape input to be selected; no further s-expressions will be read from the Teletype until an end-of-file is read from tape, /S is typed, (IOC S) is evaluated, or ↑Q is SETQ'ed to NIL. Input from the vidisector is through the function VIDI, q.v. There are similar switches for the Teletype, the DECtapes, and the line printer on output, which are shown in the summary below. Output to the display is from arrays in memory through the functions DISINI, DISAD and the value DISLIST. See "Display". The DECtape routines must be given names of input files and numbers of output tapes before DECtape input-output can be done. For this purpose, the functions UREAD, UWRITE, UFILE and UKILL are provided. See"DECtapes".

Summary:

Device	Input through	Select	Deselect	Output through	Select	Deselect
Teletype READ	(Default option)			PRINT	#V	#W
DECtape READ	#Q		#S	PRINT	#R	#T
Line printer				PRINT	#B	#E
Display				See "Display"		
Vidisector VIDI						

When the above control characters are typed, they are not seen by READ, but are seen by READCH. See also IOC, IOG Display, DECtapes, Teletype, Line Printer, and VIDI.

INTERN	SUBR	Argument is pointer to atom structure; puts said atom on OBLIST and returns (probably new) atom pointer.
INUM	SYM	The largest fixed-point number which may be represented by no more list structure than a "pointer" one larger than itself.
IOC	FSUBR	Identical to typing the characters in its arguments with the control key held. Its value is its argument list.
IOG	SUBR	Saves up the current state of all i-o switches, performs IOC of the value of its first argument, evaluates its second argument which is of the form (FUNCTION args) then restores the i-o switches to their former states.
LABEL		One of the indicators which EVAL looks for when evaluating an s-expression with a non-atomic CAR.
LAMBDA		One of the three indicators EVAL looks for when evaluating an s-expression whose CAR is non-atomic.

LAP

LAP is not part of the LISP on the system tape, but rather as a file of s-expressions on the system tape as file @ LAP. (LAP name indicator), where the indicator is SUBR, FSUBR, or LSUBR, causes LAP to call READ repeatedly, each time reading one tag or storage word. An atom is taken as a tag, except for NIL which indicates the end of the function being read; a non-atomic s-expression is taken as a storage word in either the format (Inst Acc Adr) or the format (Inst Acc Adr Indx). Inst should be a PDP-6 instruction mnemonic, optionally suffixed @ (e.g. HLRZ@) for the indirect bit. Acc should be a number from Ø - 17, or P, the regular pushdown pointer. Adr may be a numeric machine address, a tag in that function, a negative number, one of certain symbols for entry points to LISP internal routines, or a list in one of the following forms: (QUOTE atom) for a pointer to atom; (SPECIAL atom) for a pointer to atom's special cell (which is the CDR of the VALUE property); (E atom) for a pointer to the name of a function being called; or (C w x y z) for a pointer to a constant, i.e., a storage word containing (w x y z). Indx is an optional left-half quantity, such as an index register specification, of the same form as Adr. The best way to get a hand-coded function into the system is with LAP. A SUBR may have no more than 5 arguments, the values of which are placed in accumulators 1 through 5 when the SUBR is called. A pointer to the argument list of an FSUBR is in 1. The arguments of an LSUBR are on the regular pushdown list, last on top, and - (number of arguments) is in 6. Accumulators 1 through 7 are the ONLY accumulators available for use within a subroutine, which is expected to return its value in accumulator 1. (PUSHJ P *AMAKE) will return a pointer to the "a-list" in accumulator 2.

Thus, a typical usage of LAP to get a hand-coded into the system would be:

```
(LAP ABS SUBR)
(PUSHJ P NUMVAL)
(MOVMS Ø 1)
(JCALL 2 (E MAKNUM))
NIL
```

Four UUO (trap) instructions are available for LAP: CALL, JCALL, CALLF and JCALLF. These are to be used for function calls in the form (CALL n (E fn)) where n is as follows: Ø through 5, calling a SUBR or EXPR with Ø to 5 arguments; 16 octal, calling LSUBR with arguments on pushdown list and

- (number of arguments) in accumulator 6;
17 octal, calling an FSUBR or FEXPR with pointer to argument list in 1.
When one of these UUOs is first executed, the UUO handler will call the interpreter if calling an EXPR or FEXPR, otherwise in the case of a CALLF or JCALLF will execute a PUSHJ or JRST, respectively, to the specified function, and in the case of CALL or JCALL will execute the PUSHJ or JRST but in additon will change the UUO to a PUSHJ or JRST unless (NOUUO T) has been evaluated.
The F forms are necessary to call functions whose names are computed; the J forms save code in the case of (RETURN (fn ...)).

LAST SUBR
```
(LAMBDA (A)
        (COND  ((ATOM  (CDR A))A)
               ((LAST  (CDR A)))))
```

LENGTH SUBR
```
(LAMBDA (A)
        (COND  ((ATOM A)  ∅)
               ((ADD1 (LENGTH  (CDR A))))))
```

LESSP LSUBR — Takes any number of arguments, returns T only if arg1 <arg2 <arg3 ... etc., and NIL otherwise.

*LESS SUBR — LESSP of two arguments.

Line printer — Output to the line printer may be selected by typing /B or setting ↑B to T, and deselected by typing /E or setting ↑B to NIL.

LINEL VALUE — That number of characters which will be output before the LISP output functions will insert a carriage-return-line-feed and reset CHRCT to LINEL. LINEL may be modified by SETQ.

LIST FSUBR — (LAMBDA (A) (MAPCAR (FUNCTION EVAL) A))

LSH SUBR — Logical left shift of the value of the first argument by n places, where n is the value of the second argument, which may be negative.

LSUBR — Indicator for subroutine function which may take any number of arguments which are placed on the regular push down list, last on top. The negative number of arguments is placed in accumulator 6. In an s-expression function, the notation (LAMBDA var ...)

where the "LAMBDA-list" is an atom, indicates that the variable var is to be bound to the number of arguments, and the values of the arguments may be gotten by calling (ARG 1) (ARG 2),etc.

MACDMP LSUBR — With no arguments, causes a garbage collection, zeroes the free storage list, waits for all output to finish, and transfers control to MACDMP. If an argument is given, the PRINC of it is given to MACDMP as a command.

MACRO — Indicator of a MACRO property, simulated by the interpreter and expanded at compile time. The property should be a function of one argument. When a call to this function is encountered, the list which is the function call (i.e., CAR of it is the function name) is fed to the macro definition as the one argument. Then the MACRO property function is expanded, for example:

```
(DEFPROP CONSCONS
         (LAMBDA  (A)
                  (COND  ((NULL  (CDDR A)) (CADR A))
                         ((LIST  (QUOTE CONS)
                                 (CADR A)
                                 (CONS (CAR A)
                                       (CDDR A))))))
         MACRO)
```

would cause (CONSCONS A B C) to expand first, to (CONS A (CONSCONS B C)), then to (CONS A (CONS B (CONSCONS C))), and finally to (CONS A (CONS B C)), which is what would be interpreted or compiled.

MAKNAM SUBR — Argument is list of atoms whose print names are single characters (actually it takes the first character of each print name); value is pointer to s-expression which if printed out, would be the concatenation of the single characters which were its arguments.

MAKNUM SUBR — Turns the pointer which is its first argument into a LISP fixed - or floating-point number, according as whether its second argument evaluates to FIXNUM or FLONUM.

MAPC	SUBR	First argument is function; second is argument list. Returns NIL, does no CONSes.
MAPCAR	SUBR	First argument is function; argument is second.
MAPLIST	SUBR	Function is first argument; argument list is second.
MEMBER	SUBR	Uses EQUAL
MEMQ	SUBR	Like MEMBER, But uses EQ.
MINUS	SUBR	- (argument).
MINUSP	SUBR	Returns T if argument is less than Ø.
NCONC	LSUBR	Takes any number of arguments. (NCONC) is NIL. (NCONC x) is EQ to x.
NCONS	SUBR	(LAMBDA (A) (CONS A NIL))
NIL	VALUE	False value of predicates, explicitly tested for by COND. (EVAL NIL) is NIL (NIHIL ex nihilo). (MAKNUM NIL (QUOTE FIXNUM) is Ø. NIL ends lists.
NOT	SUBR	Identical in value to NULL.
NOUUO	SUBR	(NOUUO T) prohibits the UUO handler from changing CALLs and JCALLs to PUSHJs and JRSTs, enabling tracing of the functions so called. (NOUUO NIL) restores that ability.
NSTORE	FSUBR	(NSTORE (name indx1 indx2 ...) number) deposits the low-order 18 bits of number in the specified cell in the array name. NSTORE evaluates its second argument first.
NULL	SUBR	Identical in value to NOT.
NUMBERP	SUBR	Returns T if its argument is a number; NIL otherwise.

Numbers — Positive fixed-point numbers less than about 4ØØØ decimal are represented by "pointers" one greater than their value and no further list structure. Floating-point numbers and

other fixed-point numbers are represented by atom structures whose property list contains a pointer to either one indicator FIXNUM or the indicator FLONUM, and a pointer to a word in Full Word Space containing the actual number. For numeric input, see READ. Numeric output for floating-point is decimal; otherwise, is in radix BASE, and if that radix is 1Ø decimal, LISP will end the number with a decimal point unless *NOPOINT has been set to T. The arithmetic functions (except MINUS) take any number of arguments, and use fixed-point arithmetic until the first floating-point operand is encountered, at which time the current result and all succeeding fixed-point operands are floated, and the result is floating-point.

NUMVAL	SYM	(PUSHJ P NUMVAL) with a LISP number in 1 returns a machine number in 1.
OBLIST	VALUE	The object list, a list of buckets of atoms.
OPS	FEXPR	or FSUBR Part of LAP. Takes pairs of arguments of the form sym value, gives each sym a SYM property of value.
OR	FSUBR	Takes any number of arguments; returns first non-NIL argument or NIL; does not evaluate arguments past the one it returns.
PLUS	LSUBR	arg1 + arg2 + arg3 ...
PNAME		Indicator for print name property. The property is a list of pointers to words in full word space containing the actual character string.
PRINC	SUBR	Prints any s-expression; does not insert slashes before characters which must be quoted with a slash on type-in. PRINC does not print a space before or after what it prints.
PRINT	SUBR	Identical to (PROG2 (TERPRI) (PRIN1 arg) PRINC (QUOTE /))).

PRIN1	SUBR	Prints any s-expression; inserts slashes before characters which would otherwise be syntactically incorrect as part of an atom's print name, such as +, (, etc.
PROG	FSUBR	Evaluates each member of its argument, disregarding its value. Enables the writing of ALGOL-like programs in LISP.
PROG2	SUBR	Returns the value of the second of fewer than seven arguments.
PUTPROP	SUBR	Puts the value of its second argument on the property list of the value of its first argument with the indicator which is the value of its third argument.
Quit		A user may quit at any time by typing /G(bell) or evaluating (IOC G). This returns LISP to the top level instantly without, however, emptying input or output buffers.
QUOTE	FSUBR	Except that it is an FSUBR, identical to CAR.
QUOTIENT	LSUBR	arg1 / arg2 / arg3 / arg4 ...
READ	SUBR	Reads one s-expression from the currently selected input device; see "Input-Output". A number without a decimal point is assumed to be in radix BASE, a number with digits only to the left of a decimal point is assumed to be a fixed-point decimal number; a number with digits on both sides of a decimal point is assumed to be a decimal floating-point number. Typing a rubout will rub out the last s-expression if the last character typed was space,), comma, or tab; will rub out the last (if (was the last character typed; otherwise will rub out the last atom fragment typed. Any character which would otherwise be syntactically incorrect as part of an atom's print name, such as an initial digit or (or ., may be quoted with the character /. / itself may be quoted this way. The various characters

		which control LISP's input-output switches may also be quoted this way, but will still have their usual effect. Unless so quoted, these characters are not seen by READ.
READCH	SUBR	Reads one character from the selected input device; see "Input-Output". The various characters which control LISP's input-output switches are seen and treated as any other characters, but still have their usual effects.
READLIST	SUBR	Similar to MAKNAM q.v., but automatically INTERNs any atoms appearing in the resulting s-expression.

Regular Push Down List Storage area in LISP containing returns for function calls and bindings of all local variables. Normally contains 1ØØØ registers.

REMLAP	FEXPR	Part of LAP. REMOBs all functions associated with LAP. With an optional argument of T, removed the SYM property of all atoms which possess it.
REMOB	FSUBR	Take any number of atomic arguments; removes them from the OBLIST and returns NIL.
REMPROP	SUBR	Removes the property with the indicator which is the value of its second argument from the atom which is the value of its first argument. Returns T if the property was there, and NIL otherwise.
RETURN	SUBR	See "Current Problems".
REVERSE	SUBR	Reverses the top level of a list, using CONS.
RPLACA	SUBR	Replaces CAR of the value of its first argument with the value of its second argument.
RPLACD	SUBR	Replaces CDR of the value of its first argument with the value of its second argument.
SASSOC	SUBR	Uses EQ.
SET	SUBR	Causes the value of its second argument to be placed, with RPLACD, in CDR of the VALUE property of the value of its first argument.

SETQ	FSUBR	(SET (QUOTE arg1) arg2).
SETTIME	SUBR	Sets (TIME) counter to value of its argument.
SPEAK	SUBR	Takes no arguments, returns current value of CONS counter.
SPECBIND	SYM	(PUSHJ P SPECBIND) (Ø Ø var1) (Ø n var2)... causes the current binding of the special variables var1 and var 2 to be saved up on the Special Push Down List, and causes var1 to be bound to NIL and var2 to be bound to the contents of accumulator n.
SPECIAL	FEXPR	Part of the compiler. Declares all of the atoms which are in its list of arguments to be SPECIAL. However, all free variables in compiled functions are automatically special. There is no UNSPECIAL; use (REMPROP (QUOTE atom) (QUOTE SPECIAL).
Special Push Down List		Storage are in LISP where higher-level bindings of special variables are stored. Normally 1ØØØ registers long.
SPECSTR	SYM	(PUSHJ P SPECSTR) restores most recent batch of special variable bindings.
STORE	FSUBR	(STORE (name indx1 indx2 ...) value) sets the value of the specified cell in the array name to the value value. See "ARRAY". Evaluated its second argument first.
SUBR		Indicator for subroutine function property. Property is pointer to first instruction in routine.
SUBST	SUBR	Uses CONSes. Use (SUBST Ø Ø arg) to copy the value of arg.
SUB1	SUBR	Value is fixed or floating, same as argument.
SYM		Indicator for symbol value property. Used by LAP.
T	VALUE	True value of predicates. (EVAL T) is T (Veritas numquam perit).

Teletype		Teletype input is automatically selected when no other input is selected. Output is initially selected, but may be deselected by typing ↑W by evaluating (IOC W) or by setting the value of ↑W to T. It may be reselected by typing ↑V evaluating (IOC V), or by setting the value of ↑W to NIL.
TERPRI	SUBR	Takes no arguments; prints a carriage-return-line-feed, and returns NIL.
TIME	SUBR	Returns, as a fixed-point number, the number of sixtieths of a second LISP has been running since the last call to SETTIME, plus the last argument to SETTIME.
TIMES	LSUBR	arg1 * arg2 * arg3 . . .
TYO	SUBR	Outputs the character whose ASCII value is the value of its one argument.
UFILE	FSUBR	Files all DECtape output which has happened since the last UWRITE or UFILE and files it as arg1 arg2. Has value of current tape number, or, if file command failed, NIL.
UKILL	LSUBR	There is currently room in LISP for only two DECtape file directories. UKILL reclaims the space used by the directory of the tape whose number is its argument if an argument is given, otherwise of the current tape. Value is the argument or the current tape number.
UREAD	FSUBR	Takes two or three arguments. First two are file name (if file has one name, first argument must be @); third is tape number. If third argument is not given, current tape number is assumed. The current tape number is initially 2. The value of UREAD is the resultant current tape number.
UWRITE	LSUBR	Specifies an output DECtape, causing previous unfiled output to that tape to be forgotten. If no argument is given, the current tape is assumed. Returns either its argument or to the current tape number. If the specified tape is full, the error message "TAPE FULL" will be printed.

VALUE		Indicator for value property. The value cell/special cell is CDR of the VALUE property.
VIDI	SUBR	Takes two fixed-point numbers as arguments; returns number which is a function of brightness at the point (arg1, arg2) on the vidissector. (TVB).
Vidissector q.v.	Input from the vidissector is through the function VIDI.	
XCONS	SUBR	(CONS arg2 arg1)
ZEROP	SUBR	(ZEROP Ø.Ø is NIL).
*AMAKE	SYM	(PUSHJ P *AMAKE) returns a pointer to the current "a-list" in 2.
*LCALL	SYM	(JSP 3 *LCALL) in an LSUBR which has been called by the interpreter puts the number of arguments in 1, and a pointer to the arguments on the top of push down list.
*NOPOINT	VALUE	When set to T, inhibits the printing of decimal points after typeout of decimal numbers. Setting *NOPOINT to NIL restores this typeout.
*RSET	SUBR	(*RSET T) inhibits the rebinding of special variables to top-level values when a LISP error occurs; (*RSET NIL) permits this rebinding.