DECUS
PROGRAM LIBRARY

DECUS NO. 85

TITLE The LISP Implementation for the PDP-1 Computer

AUTHOR L. Peter Deutsch and Edmund C. Berkeley

COMPANY

DATE March 1964

FORMAT LISP

Appendix 5

The LISP Implementation for the PDP-1 Computer

L. Peter Deutsch and Edmund C. Berkeley

TABLE OF CONTENTS

Part I

Part II

Part I

1. Introduction

In October 1963 a system for implementing LISP on the PDP-1 computer was finished by L. Peter Deutsch. This system was further improved in March 1964 by adding:

-- variable length of push-down list;
-- variable quantity of combined storage;
-- optional machine language subroutines;

and is here called Basic PDP-1 LISP. It uses a minimum of some 2000 (decimal) registers out of 4096 registers in a one-core PDP-1 computer; it may use 16,361 registers in a four-core PDP-1 computer.

Basic PDP-1 LISP is presented in considerable detail in this appendix for the following reasons:

-- the structure of a system for programming LISP on any computer is thereby revealed;
-- if changes are to be implemented, they can be easily linked with the existing system.

In a one-core PDP-1 computer with 4096 registers, as many as 4070 registers may be assigned to regular LISP, and only 23 reserved for the read-in routine (namely, from 7751 to 7777, octal).

With the system described here, additional LISP functions can be defined and included in the system and later used when desired. Or if desired, additional functions can be programmed in machine language and these can be inserted compatibly with the system.

Punched tapes for placing this LISP system on the PDP-1 computer are available through DECUS, the Digital Equipment Corporation Users Organization, Maynard, Mass.

In the following, it is assumed that the reader has a fairly good working knowledge of: (1) LISP (which may be obtained from the "LISP 1.5 Programmer's Manual," 1962); (2) the machine language codes for the PDP-1 computer (which may be obtained from the computer manual supplied by Digital Equipment Corporation); and (3) the program assembly language MACRO, in which the sym-

bolic tapes are written (a description may be obtained in two manuals published by Digital Equipment Corporation).

2. Functions and Properties included in Basic PDP-1 LISP

The functions and properties included in Basic PDP-1 LISP are shown in Table 1. These functions and properties together constitute a basic subset of the functions and properties of the LISP interpreter for the IBM 7090, as stated in the LISP 1.5 Programmer's Manual.

In order to obtain other LISP functions and properties as may be desired for any particular purpose, see Sections 4 and 5 below.

Table 1

FUNCTIONS AND PROPERTIES OF BASIC PDP-1 LISP

A. Functions Identical with the Corresponding IBM 7090 LISP Functions

ATOM	LIST	PROG
CAR	LOGAND	QUOTE
CDR	LOGOR	READ
COND	MINUS	RETURN
CONS	NULL	RPLACA
EVAL	NUMBERP	RPLACD
GENSYM	PLUS	SASSOC
GO	PRINT	SETQ
		TERPRI

B. Functions Somewhat Different from the Corresponding 7090 Functions

EQ	This works both on atoms and on numbers
GREATERP	This tests for X greater than Y, not for X greater than or equal to Y.
STOP	This is equivalent to PAUSE in 7090 LISP. It takes a numerical argument which appears in the accumulator when the computer halts.
PRINT X	This prints the atom X without the extra space at the end. Its value is NIL.

C. Functions Which Have No Analog in 7090 Functions

XEQ — This provides for putting into storage a named machine language subroutine, which can be referred to and used by the PDP-1 LISP interpreter. It also provides for executing single specified machine language instructions.

The SUBR (XEQ C A I) executes the machine language instruction C, with A in the accumulator and I in the in-out register; and returns a value in the form of (a i P) where a is the new value of the accumulator after execution, i is the new value of the in-out register after execution, and P is T if the instruction skipped, and NIL if the instruction did not skip.

LOC X — This gives the machine register in which the atom or list X begins; its value is the location.

Of the foregoing functions, COND, LIST, PROG, SETQ, PLUS, TIMES, LOGAND, LOGOR, and QUOTE are FSUBRs and the remainder are SUBRs.

D. The following special form is available and is identical with the corresponding form in 7090 LISP:

LAMBDA

E. The following permanent objects exist in the Basic PDP-1 LISP system:

OBLIST	the current list of atomic symbols
NIL	F has been replaced by NIL
T	
EXPR	
SUBR	
FEXPR	
FSUBR	
APVAL	

F. Miscellaneous

The print names of atomic symbols are not part of property lists. A quick examination of listings of the system will show exactly where the print names are.

Doing a CDR of an atom is permissible and will get the atom's property list. Doing a CAR of an atom may very easily wreck the system.

QUOTE should be used in place of 7090 FUNCTION. This may re-

quire a bit of extra care in defining functions with functional arguments.

It is advisable to use PROG to avoid recursion wherever possible, even though it may take more space.

3. Use of these Functions and Suggested Test Sequences

How to use these functions is briefly explained here.

As soon as the basic PDP-1 LISP system is read into the computer, control stops at register 4. Turn up sense switch 5 for typewriter input; press CONTINUE; and the system enters a waiting loop which causes lamps to light in the program counter, looking like 1335. At this point, the LISP system is ready for manual typewriter input. As soon as the operator types, for example:

```
(CAR (QUOTE (A B C D)))
```

together with a final space at the end of the last right parenthesis, the computer takes control of the typewriter, impulses a carriage return, and then types out:

```
A
```

which of course is the correct answer. Similarly, for the other suggested test sequences in Table 2 below.

Table 2

SUGGESTED TEST SEQUENCES

Input	Response
(CAR (QUOTE (A B C D)))	A
(CDR (QUOTE (A B C D)))	(B C D)
OBLIST	The interpreter will type out a complete list of the atomic symbols stored within it.
(LIST (QUOTE (A B C D)))	((A B C D))

NIL	NIL
(CDR NIL)	(APVAL NIL)
(CAR (QUOTE (T.NIL)))	T
(CONS (ATOM (CDR T)) (LIST (GENSYM) (GENSYM)))	(NIL G00001 G00002)
(COND (EQ T NIL) (STOP 1)) (T (EQ (PLUS 1 1) 2)))	T
(PROG (U) (PRINT NIL) (TERPRI) (PRINT T) (SETQ U T) (RETURN U))	NIL T T
(RPLACD (QUOTE CAAR) (QUOTE (EXPR (LAMBDA (X) (CAR (CAR X))))))	CAAR
(CAAR (QUOTE ((A))))	A
(STOP 2)	Computer stops and puts 2 in the accumulator.
(PRIN1 (QUOTE CAR))	CAR, with no punctuation before or after; the value of PRIN1 is NIL.
(PRINT X)	Prints out the value of X; the value of (PRINT X) is X.
(TERPRI)	Prints a carriage return; the value of (TERPRI) is NIL.
(LOC NIL)	2651; this is the register where the NIL atom starts.
(LOC (QUOTE COND))	2725; this is the register where the COND atom starts.
(LOGAND 6 7 3)	2
(LOGOR 12 3 15)	17

```
(RPLACA (QUOTE (NIL X Y))          ((A B) X Y)
    (QUOTE (A B)))
```

Suppose the computer contains DDT -- DDT is short for "Digital Equipment Corp. Debugging Tape"; its starting register is 6000, and in one of its customary forms it uses registers 5540 to 7750. Then, if the highest storage register of LISP is below 5540, the instruction:

```
(XEQ 606000 0 0)
```

transfers control to DDT, and puts zero in the accumulator and in the in-out register.

If there is the following subroutine stored in the computer:

```
5500    dzm 5507
5501    idx 5507
5502    lac 5507
5503    dpy'
5504    sma
5505    jmp 5501
5506    jmp 2241
5507    (being used for storage)
```

and LISP is below 5500, then:

```
(XEQ 605500 0 0)
```

Will cause a horizontal line to be drawn on the scope from the origin to the x-axis positive limit, and then control will be returned to LISP.
NIL will be typed out.
2241 is the register called "prx" in the macro symbolic.

4. Auxiliary Functions Which May Be Defined with LISP Expressions

Any of the functions listed below in Table 3 can be put into the system at will, as follows: Prepare a punched tape listing of it. Insert tape into the reader. Turn on the reader. Turn down Sense Switch 5. Thereupon the computer will read in the

tape. The typewriter, when the reading in is accomplished, will type back the name of the inserted function.

Many other functions besides those listed in Table 3 may be inserted.

Table 3

AUXILIARY LISP FUNCTIONS

ABSOLUTE VALUE

```
(RPLACD (QUOTE ABSVAL) (QUOTE (EXPR (LAMBDA (X) (COND ((GREATERP
    0 X) (MINUS X)) (T X))))))
```

AND

```
(RPLACD (QUOTE AND) (QUOTE (FEXPR (LAMBDA (X A) (PROG NIL N (COND
    ((NULL X) (RETURN T)) ((NULL (EVAL (CAR X) A)) (RETURN NIL)))
    (SETQ X (CDR X)) (GO N))))))
```

ASSOC

```
(RPLACD (QUOTE ASSOC) (QUOTE (EXPR (LAMBDA (X Y) (COND ((EQUAL
    (CAAR Y) X) (CAR Y)) (T (ASSOC X (CDR Y)))))))))
```

CAAR

```
(RPLACD (QUOTE CAAR) (QUOTE (EXPR (LAMBDA (X) (CAR (CAR X))))))
```

CADR

```
(RPLACD (QUOTE CADR) (QUOTE (EXPR (LAMBDA (X) (CAR (CDR X))))))
```

CDAR

```
(RPLACD (QUOTE CDAR) (QUOTE (EXPR (LAMBDA (X) (CDR (CAR X))))))
```

CDDR

```
(RPLACD (QUOTE CDDR) (QUOTE (EXPR (LAMBDA (X) (CDR (CDR X))))))
```

CSET

```
(RPLACD (QUOTE CSET) (QUOTE (EXPR (LAMBDA (X Y) (RPLACD X (LIST
    (QUOTE APVAL) Y))))))
```