Multics LISP
A Basic Guide for Current MACLISP Users

Revision 1
12/1/72

The purpose of this sheet is to enable people who are acquainted with the LISP found on the AI and MAC PDP-10's, to use the new Multics LISP. Since the basic goal of designing this LISP was compatibilty with the LISP found on the 10's there are only a few differences. These stem mostly from a different atom structure, a compacting garbage collector and the fact that the LISP resides in a different time-sharing system with a differently constructed file system and that consoles are different causing trivial but annoying keyboard considerations. The following instructions should enable one to sit down at a console and work with LISP.

1) There are three characters that are seen by Multics before they get to LISP and that have significance to the module that sees them. The character " # " erases the previous character. The " @ " kills the whole line. And the <back-slash> is the escape character (Remember that on a 2741, <back-slash> is represented as the EBCDIC character <cent-sign>.). See the MPM for a more thourough description.

2) ITS file names have two components while Multics file names are a single string without blanks but with components seperated by " . "'s. So that programs written for use on the 10's can run on Multics, it was decided that inside LISP, files would be referenced by two names and that these would be mapped into a dotted file name in the Multics file system. Thus whereas on the 10 one would say (UREAD FOO BAR) on Multics one would say (uread foo bar) and it would read from a segment called "foo.bar". Ufile works similarly. On ITS one has a "device" and a "sname" that specify the directory in which a file resides while on Multics there exists a tree of directories. It has been decided that the "device" name be of no significance while the "sname" position of the crunit (the "current unit") be the pathname of the directory to which one wants to refer. When one enters LISP the crunit is set to the current working directory. One may change this with the furction crunit or with additional arguments to uread or to uwrite. If one says (uread foo zoo dsk >udd>ap>d) to LISP then it will attempt to open the segment >udd>ap>d>foo.zoo for reading. If this succeeds then (status uread) will give the value (foo zoo dsk >udd>ap>d) and (crunit) will give the value (dsk >udd>ap>d). Relative pathnames such as <<Smith>r are allowed.

3) Control characters are used to give the LISP implemented on the 10's certain commands that have a real-time effect. To give the Multics implementation of LISP these same commands, send a QUIT signal to Multics (by hitting the ATTN key on a 2741, or BREAK on a 38, or Control O on an ARDS) and LISP will respond

12) Saved environments (see 6) for plnr, cnvr, doctor, grind, and trace are available. To use them type "lisp >udd>ap>Reed>installed>XXXX" to Multics, where XXXX is in the set (plnr, cnvr, doctor, grind, trace). Source for these programs is in the same directory. An editor stolen from the BBN PDP-1, is available in source form there also. Comments in the source file explain its use. A better editor may be available soon.

13) Mail questions or complaints to Reed, Moon, Sunguroff or Bricklin whose directories are in AutoProg. Or if any of us is logged in then send a message. Or our phones and office are:

Room 501
545 Tech. Sq.
Cambridge, Mass. 02139

617 253-6020
617 253-6013

14) More documentation will follow, but till then here is the current list of functions:

*	lsubr (0, n)
*$	lsubr (0, n)
*array	lsubr (3, n)
*function	fsubr
*rset	subr 1
+	lsubr (0, n)
+$	lsubr (0, n)
-	lsubr (1, n)
-$	lsubr (1, n)
/	lsubr (1, n)
/$	lsubr (1, n)
1+	subr 1
1+$	subr 1
1-	subr 1
1-$	subr 1
<	lsubr (2, n)
=	subr 2
>	lsubr (2, n)
CtoI	subr 1
ItoC	subr 1
abs	subr 1
add1	subr 1
alarmclock	subr 1
and	fsubr
append	lsubr (0, n)
apply	lsubr (2, 3)
arg	subr 1
args	lsubr (1, 2)
array	fsubr
arraydims	subr 1
ascii	subr 1
assoc	subr 2

eq	subr 2
equal	subr 2
err	fsubr
errframe	subr 1
errprint	subr 1
errset	fsubr
eval	lsubr (1, 2)
evalframe	subr 1
explode	subr 1
explodec	subr 1
exploden	subr 1
expt	subr 2
fix	subr 1
fixp	subr 1
flatc	subr 1
flatsize	subr 1
float	subr 1
floatp	subr 1
function	fsubr
gc	fsubr
gensym	lsubr (0, n)
get	subr 2
get_pname	subr 1
getl	subr 2
go	fsubr
greaterp	lsubr (2, n)
index	subr 2
intern	subr 1
ioc	fsubr
iog	fsubr
last	subr 1
length	subr 1
lessp	lsubr (2, n)
list	lsubr (0, n)
lsh	subr 2
make_atom	subr 1
maknam	subr 1
makoblist	subr 1
makreadtable	subr 1
map	lsubr (2, n)
mapc	lsubr (2, n)
mapcan	lsubr (2, n)
mapcar	lsubr (2, n)
mapcon	lsubr (2, n)
maplist	lsubr (2, n)
max	lsubr (2, n)
member	subr 2
memq	subr 2
min	lsubr (2, n)
minus	subr 1
minusp	subr 1
nconc	lsubr (0, n)
ncons	subr 1

throw	fsubr
times	lsubr (0, n)
tyi	lsubr (0, 1)
tyipeek	subr 0
tyo	subr 1
typep	subr 1
ufile	fsubr
uread	fsubr
uwrite	fsubr
xcons	subr 2
zerop	subr 1