problem type: Performance
Subject: Want faster GETPROP
subsystem: microcode

GETPROP could be open coded faster. It is time-critical for a number of user functions. GETPROP would be faster if PUTPROP put new properties on the front of symbols PList instead of the back.

---
subsystem: microcode

TYPENAME is too slow, and is used by TYPENAMEP. Want primitive in microcode which is JNTYPENAMEP [alpha, beta, offset], like DTEST except jumps instead of traps if type doesn't match.

---
subsystem: compiler

Some of the initial constants and global variables can now be expanded inline for faster execution. Do stats on system, and look at GLOBALVAR references of functions which show up in profile.

****
Date: 12 Feb. 1982 8:53 am PST (Friday)
From: Moran.PA
Subject: CLOSEF problem

CLOSEF doesn't seem to work on the Dolphin in functions that work perfectly well on Maxc. My style is the following:

```
(LAMBDA (F)...(OUTFILE F)...(PRINTOUT NIL...)...(CLOSEF F)...)
```

When it gets to CLOSEF, it says that file val-of-F is not open. However, calling CLOSEALL in the break does close file val-of-F, which really was open and which did indeed get all the printout intended.

Tom

Date: 1 March 1982 7:20 pm PST (Monday)
From: Bobrow.PA

The compiler seems to me to be much too verbose. Cannot there be a flag so that

it only says things when either a) there is a free variable b) an undefined function called

Currently I scan my dribble file for these, which are most useful. But so little wheat and so much chaff.

****
Date: 19 March 1982 9:59 am PST (Friday)
From: VanLehn.PA
Subject: HELPFLG bug
To: Lispbug↑.pa
cc: VanLehn
Reply-To: VanLehn

LISPX seems to be rebinding HELPFLG so that SETQ's at top level have no effect. I can't force errors to break by setting it to BREAK!, which makes debugging under errorsets damn near impossible.

Kurt

****

Date: 19 Mar 1982 1006-PST
From: Neil Goldman <GOLDMAN at USC-ISIF>
Subject: CONSTANT
To: MASINTER at PARC-MAXC

In the macro for CONSTANT, I propose that the evaluation of the form be done under ERRORSET protection, with an error causing the thing to be treated as a DEFERREDCONSTANT, just as is done when the value fails to pass the CONSTANTOK test.

neil
****
Subject: SEPRCASE/GETBRK bug
To: LispCore↑
If a character X is defined as a macro in readtable Y, then (SYNTAXP X 'BREAK Y) is NIL, but X is not in (GETBRK Y). I don't know whether this is a bug or a feature, but in any case it means that SEPRCASE treats such a char as an alphabetic, and FINDCALLERS and friends will miss when the target atom is preceded by the macro character.

****
Date: 8 Apr 1982 1932-PST
Subject: SPECVARS and the compiler.
From: RBATES at USC-ISIB

The manual on page 18.21 states "Whenever bcompl or brecompile encounter a block declaration they rebind retfns, specvars ... to their top level value". This is NOT true as far as SPECVARS are concerned. The reason is the function LOCALVARS in COMP seeing that LOCALVARS are T sets SPECVARS to SYSSPECVARS without checking what the current value of SPECVARS is. This bug has been around since Sept 1976! These two functions with the coms show off the bug:

```
(FOO (LAMBDA NIL (PROG (X) (FUM]
(FUM (LAMBDA NIL (NILL X)]
(SETQ BUGCOMS '((FNS FOO FUM)
            (SPECVARS X)
            (BLOCK (FOOBLOCK FOO FUM (ENTRIES FOO]
```

/Ray
****
Date: 19 Apr 1982 0956-PST
Subject: DECLARE:
From: RBATES at USC-ISIB

I noticed that DECLARE: don't get compiled away (the function DECLARE: always get called), but DECLARE does get compiled away. This problem has been around awhile. Also that the example on the end of page 23.16:

```
(FOR X IN Y (DECLARE: (LOCALVARS X)) -- )
```

doesn't work.

/Ray
****

Date: 19 MAY 1982 1535-PDT
From: KAPLAN
Subject: \TAKEINTERRUPT skeleton on AINTERRUPT

I defined a skeleton for the \TAKEINTERRUPT macro on AINTERRUPT.

It has dummy calls to 2 primitives, one for checking whether \INTERRUPTABLE is T everywhere above its lowest binding (which is predictably NIL by the

client of \TAKEINTERRUPT). I can simulate this with a stack search using a constant stack pointer, but this probably should be done at a lower level.

The other is a function for calling INTERRUPTED (I called it \CALLINTERRUPT). I don't think this can be the same as \CAUSEINTERRUPT. Maybe it is sufficient simply to branch to the keyboard context here.

I noticed that there are 2 global variables used mark that an interrupt is pending, \InterruptChar (used in the keyboard handler) and \INTCHAR, used in INTERRUPTED. \CAUSEINTERRUPT clears \InterruptChar and sets \INTCHAR. I don't quite understand this--is it temporary cause we haven't committed to WIND?

--Ron
****
Date: 28 MAY 1982 0003-PDT
From: KAPLAN
Subject: Bug in BCOMPL/BRECOPILE
To: MASINTER

I noticed that BRECOMPILE and BCOMPL setup LOCALVARS slightly differently.

BCOMPL initializes it to SYSLOCALVARS. BRECOMPILE does that only if LOCALVARS is T. If it is not T, it sets it to (UNION SYSLOCALVARS LOCALVARS).

Do you understand this? Which is correct, or are both correct?

If the UNION makes sense, should it also happen for SPECVARS?

--Ron
****
Date: 15 JUN 1982 2210-PDT
From: MASINTER.PA
Subject: INSPECT scrolling
To: lispsupport

If you put up an inspect window, and then change radix (e.g., from RADIX(8) to RADIX(10)), you get inconsistant output when you scroll the window.

Larry
****
We need to have an updated suite of tests to give to the technicians for hardware checkout.

Date: 18 June 1982 9:18 am PDT (Friday)
FROM: MANN
Subject:Runmicrotest.cm/replacement

Can you send us a message about Runmicrotest.cm or a suitable replacement to use in checking out the Dorado's as we discussed the other day. We do need this to do a good verification that the machines we ship run all the emulators.
****
Interlisp-D I believe has never been verified to run the DIsrael Y-FUN test of spaghetti stack operation.

Date: 23 Jun 1982 1332-EDT
From: DISRAEL at BBNG
To: masinter at PARC-MAXC

(1) YY2 is the Interlisp version of the standard fixed point (recursion) operator in the lambda calculus. As written in lambda calculus form, it looks like this: LAMBDA F. (LAMBDA X. F (X X)) (LAMBDA X. F (X X)). You apply it to a functional - a function that

returns a function as value - and then apply the result to, say a number. So if you give YY2 the factorial functional - everyone's favorite example : LAMBDA FUN. (LAMBDA N. IF N = 0, 1, ELSE TIMES N (FUN N-1)) - you get something which when applied to 3 yields 6. It does this bit of magic by unwinding an internal lambda expression 4 times (in this case). So it simulates recursion by as much iteration as you need. Note: the definition of FACTFUN is not syntactically recursive, and if one defines FACT as (YY2 'FACTFUN), then FACT is also not recursively defined.

One can actually think of YY2 (never mind why it's called that and not Y) as the limit (the least fixed point) of an infinite series, starting with Y-0 (which see) and getting on by applying Yn to G to get Yn+1. (Moreover Yn = (G 'Yn) - so every Y is a fixed point of G).

Z is another, slightly unstandard recursion operator - written in lambda calculus form as follows: LAMBDA F. ((LAMBDA X. F (LAMBDA Z. (X X) Z)) (LAMBDA X. F (LAMBDA Z. (X X) Z)). Again Z of FACTFUN is a non-recursively defined FACTORIAL applicable to numbers.

(3) As for COMBOY - it's the Y-type recursion operator written out purely in terms of the two (so-called) primitive combinators. K (LAMBDA X. (LAMBDA Y. X)) and S (LAMBDA X. (LAMBDA Y. (LAMBDA Z. ((X Z) (Y Z))))). But there is a bug in the code: as it stands, it is not applicable to numbers - only to functions; e.g. not to 3 but to LAMBDA.() 3 - and of course TIMES, SUB1, etc. barf at these.

(4) Speaking of combinators; BB is functional composition (in disguise), SKIAPPLY is function application and SKIAPPLY2 is "APPLY" - again in disguise. (It takes a function and an arg as arguments; SKIAPPLY takes a function and returns a function which is the argument function to SKIAPPLY primed for application. (Baroque, eh??). WW takes a function and produces a function which when applied to an argument, produces a version of a two-placed function whose two arguments are identified. (So SQUARE is WW applied to TIMES.)

(5) F and J are weird functionals of purely theoretical interest (unlike the others which are, as you'll surely allow, of immense practical import). J is a function provably equal to I (LAMBDA X. X); but which is, unlike I, provably non-normalizable. I, moreover, is provably the only fixed point of F. (I think J, like COMBOY, may require functions as arguments all the way down.)

To TEST:

(APPLY* (YY2 'FACTFUN) 3) will do nicely to compute (factorial 3). (The same goes for (APPLY* (Z 'FACTFUN) 3).) You can go (APPLY* (FACTFUN 'FACTORIAL) 3) - where FACTORIAL is the regular recursively defined factorial function. And, since YY2 (or Z) are fixed point operators (so F = YF) you can go (APPLY* (APPLY* 'FACTFUN (YY2 'FACTFUN)) 3). ETC...

****
Date: 27 June 1982 5:53 pm PDT (Sunday)
From: vanMelle.PA
Subject: incompatible changes

incompatible changes for whenever we feel like introducing an incompatible change:

Rearrange InterfacePage so that IFPFaultHi is even-aligned.

Rearrange DataTypeDescriptors so that DTDFREE, DTDCNT0 and DTDNEXTPAGE are all in the same quadword.

Make htfind xor the hiloc of the datum when computing the hash probe.

****
Date: 30 JUN 1982 0755-PDT
From: SPROULL
Subject: Dolphin experience

[This was a long report on Bob's experience with the Dolphin. I have exerpted the problems which I think are still relevant]

- - - - - - - - - - - - - - - - - - - -
The bad news

My view is that Interlisp is sinking of its own weight, and the move onto a personal computer has shown the hulk in alarming vividness. This section presents a brief justification of this view and offers possibilities for remedies. The problem can be fixed, but it may be costly.

The problem, as I see it, is that Interlisp has never had a clean internal structure of the system (as separate from the language): features and packages have accreted, wired into the existing maze to create a tighter maze. It's now so bad that a good programmer who encounters an Interlisp system is at a loss for what to do for a good long time. Few if any of the "interfaces" in Interlisp correspond to things he recognizes from other environments. He has to seek out facilities one by one; his intuition for where to look is often wrong; and he remains worried about deep interactions among various parts of the system.

This problem has been made worse by the move onto a personal computer. I think to a great extent, new facilities are added to Interlisp using the same rather low standards of interface definition that have characterized Interlisp so far. For example, while I think the facilities provided by the Interlisp-D stuff for graphics are mostly OK, the interfaces can be substantially improved.

I feel the "system" part of Interlisp needs a thorough overhaul. I favor the "open system" approach in which a very few low-level primitives are built in, and the rest is done with packages that can be separated and that have well-defined interfaces.

I realize that an undertaking such as this is a big one. However, I believe that a great deal of the detailed functionality of Interlisp can be retained (even much of the code can be retained), but the interfaces need to be redesigned in light of the tremendous evolution of the system. It's remarkable that they've remained useful as long as they have, but they need overhaul. I see two hopeful signs:

1. To the extent possible, new work should be done in the form of LISPUSERS packages. I'm delighted, for example, to see the new editor done in this way. (But I suspect the interface between it and the rest of the system could be improved if the system were improved.) I think it's important that some (perhaps all) of these packages be distributed in source-file form so that users can actually understand what they do.

2. The new Common Lisp effort is an opportunity to redesign many of these functions. Perhaps a long-range plan might be to put Common Lisp up on the Dolphin. Should the Xerox group be contributing to the design of Common Lisp, especially in those areas where it has the most expertise (e.g., user I/O)?

Another approach is to invest some effort in restructuring Interlisp. I think it might be worth a few days' effort to estimate the difficulty of this problem and the improvements that could be reasonably expected.

(long paragraph)

> To summarize, I'd like a complete, consistent "graphics package" at the low level. Some of the pieces are there (bitblt, line, etc.), but I don't see the structure. It appears to be a complex set of stuff with complex inter-relations. There is no clear description of what a bitmap is, or what a display stream is, or what a window is. This needs to be cleaned up.
>
> More to the present point, however, is that I find the manual almost completely lacking is discussion of concepts. To pick an example from the current manual, consider the file package. What is a "file" from the point of view of Lisp? What is its purpose? What does it contain? What is the distinction between what is remembered inside the Lisp virtual memory and what is retained on the disk? And so on and on . . . While the file package might have once started out so simple that none of these questions arose, it's long past that point now.
>
> The new graphics stuff definitely needs a good deal such concept documentation. In the manual, old and new, there's too much emphasis on functions and not enough on structure and concepts.
>
> The best manual would result from a system restructuring The Interlisp language should form a distinct part of the manual -- some sections
> of the current manual are salvageable in this respect. If the interface between the language and the packages were cleaned up, this section would describe the interface. I, for one, would benefit enormously from a self-contained section that describes the language without reference to any of the system stuff.

****
Date: 8 July 1982 8:25 am PDT (Thursday)
From: VanLehn.PA
Subject: main data space overflow
....
I've tried to find the storage leaks using COUNTDOWN, MAPATOMS and so forth. So far, the only circularities I've come across are on LISPXHISTORY. I need better leak-finding tools. One would be to do the mark & sweep part of garbage collection, including the freelist as "accessible" during the mark. The list that the sweep delivers is therefore all the cons cells that got leaked. By looking through the ones with atomic cars and cdrs, I could probably figure out from the pnames where the leaks came from.

4. Of course, having found all the lost storage, it could be put back on the freelist, saving me a reload (but probably still taking 20 minutes). Since there is plenty of array space around, the mark & sweep could be written simply, using its own array to hold the mark bits. I don't see any reason to do the "copy" part of a "stop & copy" since swapping doesn't seem to be a problem on the Dorado (according to temporal intuition and control T).

5. If the mark's bit array is the screen bitmap, one could probably learn alot about the storage use and maybe the chronology of the leaks by seeing not

only where in vmem the leaks are, but watching the mark propagate out from specific atoms. A quick calculation has it that marking the current Mds, assuming it's mostly cons cells (95% in my program), would take a 900 by 900 bitmap.

I'd be willing to help code such a tool, or any other.

Any tools or diagnostic ideas would be welcomed with extreme enthusiasm.

Date: 8 JUL 1982 1155-PDT
From: ROACH.PA
Subject: DEFINEQ and MACROs
To: LISPSUPPORT
cc: ROACH

Dear Interlisp Support,

I think MACROs, DEFINEQ, and the Interlisp compiler interact incorrectly. I would like to define macros such as DEFEXPR, DEFFEXPR, etc. that can appear in files and will expand out into DEFINEQ forms which go on to be compiled like other DEFINEQ forms. I've been told by Ronald Kaplan that this won't work, and in fact, it doesn't. What is Interlisp's problem? I might point out that Maclisp does allow you to do this sort of thing. This is a pretty serious deficiency on Interlisp's part.

Secondly, instead of compiling expressions in the order in which they occur, the Interlisp compiler gathers functions definitions into a separate group from all other expressions. This is also a bug. (Again Maclisp does the right thing.) Compiled forms ought to load in the same order in which the uncompiled forms loaded.

I am hopeful that action will be taken on these problems.

****

Date: 27 JUL 1982 0935-PDT
From: KAPLAN.PA

. . .

We probably also ought to implement a device info interface that could, for example, tell how many pages are left on the disk, in an ifs directory, or in the ifs as a whole.

****

00348 00024 UU
Date: 10 Aug. 1982 8:48 am PDT (Tuesday)
From: VanLehn.PA
Subject: MARKASCHANGED
To: lispsupport
cc: VanLehn

MARKASCHANGED apparently doesn't inform the compiler that the function needs to be compiled. Reference manual doesn't specify whether it does or not. Obviously, it should tell the compiler to recompile.

****

Date: 26-AUG-82 11:46:38 PST
Subject: AWFUL CODE FROM CREATE USING
To: LispSupport

(create FOO using X) when FOO is a RECORD translates as a forest of CONS of CARs of CDRs rather than COPY. Produces awfully large chunks of code that doesn't even run fast. Bug?

****

Date: 8-Sep-82 14:03:40 PDT (Wednesday)
From: Masinter.PA
Subject: Re: RAISE for files

Users want to be able to read a file in lower case as if it were in upper case.

Why don't we put a translation table into READ tables? We already have them for FILEPOS and FFILEPOS etc. This would make a lot of sense. The cost is relatively small.

Date: 9-Sep-82 16:18:21 PDT (Thursday)
From: Masinter.PA
Subject: RESETLST vs RESETFORM
To: Bobrow
cc: LispSupport, Masinter.PA

This should be in the manual, or maybe we should fix RESETFORM.

Example: this doesn't work:
(RESETFORM (DEFPRINT 'A 'FOO) stuff)

This is what DOES work:

```
(RESETLST (RESETSAVE NIL (LIST 'DEFPRINT 'A (DEFPRINT 'A 'FOO)))
         stuff)
**********************
```

Date: 10-Sep-82 8:48:59 PDT (Friday)

There sentiment for making TY not elide comments

Date: 14 Sept. 1982 9:41 am PDT (Tuesday)
From: JonL.pa

I've never used TY, but if it does the obvious thing, then one might expect that TY* would be the command which doesn't elide comments (e.g., PF and PF*?)
****

We need to have an inbound CHAT server so you can CHAT to a machine from a remote terminal over the PUP and NS Ethernet.

We need to handle UNIX filenames better
We need device synonyms and pseudo-devices
We need to be able to delete 1100 {DSK} files without building the whole map

We need to document the facilities for doing Binary i/o for bitmaps, integers, floats. AOUT/AIN.

Compiler: [14 NOV 1981 1815-PST] (FUNCTION (LAMBDA --)) expression used as a value to be stored into a record slot. The compiler did produce a suitable subfunction, but then compiled as the value of the (FUNCTION --) expression the free varaible NEWVALUE.

We need to fix Sandbarring

Storage management:
   We need to fix the GC so that it collects items which are only held as keys of hash-tables. (CLISPARRAY in particular).

we need unwindprotect

many system functions have Names which can conflicts with user fns

We need to unify the handling of meta, blank keys

we need to clean up GLOBALVARS situation

We need more diagnostics and system tests

We need to fix it so that expanding macros doesn't take so long

* FUNARG doesn't work
* MASTERSCOPE:
      WHO USES FILE FREE causes funny error messages

Date: 16 SEP 1982 1802-PDT
From: BURTON.PA
Subject: compiler bug

The bind merge optimzation getts carried away with the function DSPDESTINATION on <LISPCORE>WIND>LLDISPLAY and binds the variable \INTERRUPTABLE with the variable DS. The effect is that the \DTEST is called in a context that is uninterruptable leading to a call to RAID when DSPDESTINATION is given a bad argument such as (DSPDESTINATION 123 (DSPCREATE)).

Date: 20 SEP 1982 1954-PDT
From: SHEIL.PA
Subject: Compiler bug - EVERY

Attempting to compile the expression (EVERY xxx (FUNCTION ATOM)) generates the compiler warning message (ATOM: Too many args for macro). Code seems to be OK but the message is distracting, especially if the code has come from a type? from either a record or DECLtype.

Beau

Date: 24-Sep-82 8:47:08 PDT (Friday)
From: Masinter.PA
Subject: Re: Problem with open leaf files -- and patch
In-reply-to: BOBROW's message of 21 SEP 1982 1517-PDT
To: LISPSUPPORT
cc: Bobrow, Stefik

Danny's patch to FINDOPENFILE seems to get around the immediate problem, but some more permanent fixes are needed. Ron and I talked about these problems for a while; I thought I would send out some notes on our conversation and some additional thoughts.

There are currently three separate problem areas in the current system:

a) READ/PRINT given file names which are not fully qualified scans the directory every time, which is TERRIBLY SLOW
b) for LEAF files, the file CANNOT BE FOUND by INFILEP/OUTFILEP/FINDFILE if it is already open
c) There are a number of inconsistancies having to do with the use of DIRECTORIES and the error mechanism to implement search paths.

Proposals:
a) files which are presented to READ/PRINT (i.e., in a context where an OPEN file is required) will ONLY scan against the set of files which are open with appropriate access.

    This is a change from Interlisp-10 semantics, where if you do an INFILE(FOO), and then create a NEW VERSION of FOO, and

then do READ(FOO), you will get a FILE NOT OPEN error.

b) we should implement the notion of a "search path" device, e.g. {LISPUSERS} and {SOURCES}. The system will support assigning search paths to a device (new function), so that one can say {SOURCES} = {PHYLUM}<LISPCORE>WIND> , {PHYLUM}<LISPCORE>SOURCES>

Doing an INFILEP on {SOURCES}xxxx will return a full filename of {PHYLUM}<LISPCORE>WIND>xxxx, i.e., the name returned will be fully qualified. OUTFILE on {SOURCES} will write on the FIRST directory on the search path.

The general idea here is to take what is currently done via the error mechanism and DIRECTORIES and FINDFILE and instead build it in in at a lower level. This will allow some more rational implementations of the facilities.

It will also make more logical the link between the "connected" directory and the search path; that is, one can either connect to {SOURCES} or to {WIND} or to {LISPUSERS}. It will remove the distinction between FINDFILE and INFILEP in the non-spelling-correction case.

Finally, all of this is relatively easily implemented in Interlisp-10! Interlisp-10 already supports a (undocumented) feature where if you PUTPROP(LISPUSERS DIRECTORIES (<LISPUSERS> <LISP>)) and attempt to FINDFILE(LISPUSERS:filename), it will in fact search those directories.

Comments?

Date: 24-Sep-82 8:49:48 PDT (Friday)
From: Masinter.PA
Subject: Re: Bitmap editor
In-reply-to: SHEIL's message of 21 SEP 1982 1603-PDT
To: SHEIL
cc: burton, lispsupport

I always wanted to do EDITBM on a "window".

One general way of handling this is to have a general "coersion" function which coerces to "clipped bitmap", with the relatively obvious coersions for windows/displaystreams/bitmaps/regions....

Date: 24 Sep 1982 1538-PDT
From: Friedland
Subject: interlisp d bug
To: cschmidt, rindfleisch

renamefile on {DSK} doesn't work. If you have a file A 100 pages long and a file B 200 pages long and (RENAMEFILE A B), you end up with a file B 200 pages long, its first 100 being the old A and the last 200 being garbage from the old B. You have to (DEFILE B) before the renamefile. This despite what the manual says.

PEter

Date: 29 SEP 1982 2012-PDT
From: JONL.PA
Subject: Undefined Function

Once in a while, I mistype a DEFINEQ, and wind up with an s-expression in the definition cell of some litatom which is *almost* what I wanted -- it

just lacks the word LAMBDA. The error message you get when you try to run such a function is "Undefined Function" -- wouldn't it be better to reserve that msg for the case of a definition cell which is either NIL or NOBIND, and print something more informative for the case where it contains something like ((X) (LIST X (TIMES 2 X)), or (() (PRINT 5)).

Date: 29 SEP 1982 2016-PDT
From: JONL.PA
Subject: PUNTing a broken compilation
To: lispbug↑

1) There needs to be an advertised way to do a BCOMPL which ignores errors which occur during the compilation of a single funciton, so that a single call to BCOMPL will proceed thru the while file, finding perhaps other errors before ending.
2) I tried the unadvertised function (PUNT) after one of my macros caused a BREAK during compilation, and it appeared as though the then-current break window became the normal TTY display stream.

Date: 30 SEP 1982 1119-PDT
From: SHEIL.PA
Subject: two comments on the RESETFORM macro

Currently, the RESETFORM macro evaluates the resetform (a) outside of errorset protection and (b) some time (during which an interrupt can occur) before the resetlst entry for undoing it is made. This could be disasterous if one is resetting, for example, a display stream clipping region and the user bombed you out. [(a) may or may not be a bug or non-feature, depending on how one reads the manual; (b) is nasty to fix and probably requires interrupt protection]

Also, the RESETFORM macro doesnt do a very good job if one passes it a LAMBDA expression as the reset function (two copies wind up in the compiled code). [The motivation for this is for two arg fns like DSPCLIPPINREGION, as

```
      (RESETFORM ((LAMBDA (X) (DSPCLIPPINGREGION X window)) NEWREG)
            forms)
```

is much more elegant than

```
      (RESETLST (RESETSAVE NIL (LIST 'DSPCLIPPINGREGION
                                (DSCLIPPINGREGION NEWREG window)
                                window))
            forms)
```

In fact, since I just realized that, it might be worth noting this trick in the
manual for other slow thinkers.]

*** I really want to shift to this notation in DEDIT, so this patch would be greatly appreciated ***

Beau

PS: From a slightly broader point of view, it would be nice to have a wizard scrutinize these macros as (a) this is not the first non-feature report for them (b) they dont look to be as good code as they could be. Last time I raised this, the discussion quickly expanded to include respecifying the whole error handling machinery (and thus nothing happened). Perhaps a useful intermediate step would be to define a few more useful abstractions, such as CATCH and THROW, which we could start using in our code to replace the convoluted RESETLST constructions that tend to generate these discussions in the first place.

****

Date: 1 OCT 1982 1430-PDT
From: SHEIL.PA
Subject: Glitch in RENAME

To: LISPSUPPORT

If one has a variable FOO which is used in some file BAR and you wish to rename
it to FUM, (RENAME 'FOO 'FUM 'VARS 'BAR) will bomb with complaint "no VARS defn
for FOO" unless FOO has a top level binding.

Beau
****
00705 00024 UU
Date: 1 OCT 1982 1611-PDT
From: SHEIL.PA
Subject: PP and PRETTYPRINT glitches
To: LISPSUPPORT

Some time ago, PP (and PP* and PPT) had LOCALVARS declarations added so that their variables would not interfere with EVALVs from PRETTYPRINT. Unfortunately, the ERRORSET implementation causes all these to be SPECIAL in Interlisp-D anyway. Pending a more general resolution of this problem, it would be nice if these fns were patched to avoid the fact that (PP X) for example, does absolutely nothing. [Major motivation: This is irritating if you know what is going on but absolutely inexplicable if you dont].

Manual note: ARGLIST of PRETTYPRINT does not match manual spec.
****
Date: 3-Oct-82 21:05:18 PDT (Sunday)
From: Masinter.PA
Subject: Re: PP and PRETTYPRINT glitches
In-reply-to: SHEIL's message of 1 OCT 1982 1611-PDT
To: SHEIL
cc: LISPSUPPORT

An additional (more general) fix is for the compiler to rename the variables which are auto-SPECVARed because of the ERRORSET hack.
****
Date: 3 OCT 1982 2327-PDT
From: KAPLAN.PA
Subject: Clispify/dwimify bug

If (don't ask why) the atoms < and > have top-level values, then CLISPIFY((LIST (FOO))) is (< (FOO) >) which then doesn't dwimify back.

As a minimum, this (and all) clisp transformations should not be performed in environments where they won't dwimify properly.

****
Date: 4 OCT 1982 0514-PDT
From: JONL.PA
Subject: MASTERSCOPE Message

If you edit a macro, MasterScope will correctly tell you that certain functions depend upon it; but when calling UNSAVEFNS, it always prints the message "Loading FooFunction", regardles wof whether it is really loading it, or merely UNSAVEDEFing it.
****
Date: 4 Oct. 1982 8:26 am PDT (Monday)
From: Stefik.PA
Subject: Re: Interlisp-D planning

Larry and Bill, -

      The Lisp features {desired by Mark et al.) were

(1) Ability to have functions without naming them.
(2) Ability to hash on strings (or unintered atoms?) for our LOOPS uids.
(3) Access to an efficient (say B-tree) database (perhaps cedar?).

Of more immediate importance will be some participation by yous guys in design reviews of LOOPS, and consulting on performance tuning. Mark

****
Subject: lisp.tasks
* UNBREAK work on internal block functions just like TRACE
* DUMMYFRAMEP definition correction
* compiler optimization NEWOPTFLG=T
* free code deleterefs pointers therefrom

Interlisp-10 problems
* DELFILE BUG
* GETSTREAM
* CALLSCCODE returns duplicate values
* add ALLOCSTRING
* make NCREATE allocate system datatypes too

Date: 6 Oct. 1982 9:44 am PDT (Wednesday)
From: Bobrow.PA
Subject: Re: Interlisp-D planning
In-reply-to: Masinter's message of 5-Oct-82 19:45:57 PDT (Tuesday)
To: Masinter
cc: Stefik, Bobrow, vanMelle

1) Must function call always go through an atom? Perhaps the name slot on the stack could be made in the "naked" case to point to the fn data object. Inspect
macros might allow at least finding out about arguments expected.
One might even have a string in such an object to name it (this would work for our methods).

2) The current atom hash table code made available as string hashing would do quite nicely at this stage for us, I think. More than 2↑16 atoms would be nice,
but is not right, since we want separate name spaces with overlapping names, and I don't think we need anything but string hashing.

3) Eventually we might want to use file based indexes for knowledge bases (e.g.
BTrees) when they get too large. The current Alpine project is NOT planning to
provide a BTree index interface at the moment (I checked with Mark Brown) Most of the indexing stuff is done now on the clients side of the Cedar database.

Stats runs show that our GetValue and PutValue are remarkably slow (about 250 microseconds on a Dorado) and take most of the time, as we expected. Help on redesign on that would be most welcome.

danny

Date: 4 OCT 1982 0514-PDT
From: JONL.PA
Subject: MASTERSCOPE Message
To: LISPBUG↑

If you edit a macro, MasterScope will correctly tell you that certain functions depend upon it; but when calling UNSAVEFNS, it always prints the message "Loading FooFunction", regardles of whether it is really loading it, or merely UNSAVEDEFing it.

Date: 4-Oct-82 12:53:41 PDT (Monday)
From: Masinter.PA
Subject: lisp.tasks
To: masinter
cc: , Masinter.PA

* Compiler: [14 NOV 1981 1815-PST] (FUNCTION (LAMBDA --)) expression used as a value to be stored into a record slot. The compiler did produce a suitable subfunction, but then compiled as the value of the (FUNCTION --) expression the free varaible NEWVALUE.
* Name conflicts with system fns
* EVAL edit macro in editor needs ERSETQ instead of NLSETQ.
* correct handling of meta, blank keys
* periodicallyreclaim be sensitive to mouse events, process suspension.
* Breakcheck problem associated with heavy swapping on first uba or udf.
* improve interface to stats for other things than fn call
* make STACK FULL non-fatal error
* (APPEND circular) bug
* Can create ARRAYs > 2↑16.
* STORAGE) function prints garbage negative numbers in the 3rd column.
* UNBREAK work on internal block functions just like TRACE
* BRKDWNRESULTS print out
* interaction of code which rewrites filemaps and RADIX
* rename low level functions
* memory map diagnostics
* DUMMYFRAMEP definition correction
* Interlisp-10 problems:
   * DELFILE BUG
   * samedir has problems on tops20: directoryname neq filenamefield
   * CALLSCCODE returns duplicate values
   * add ALLOCSTRING
   * make NCREATE allocate system datatypes too
* Code for generating Interpress from Tops-20 and Vax (Troff, Scribe, ...)
* small demo
* LISPXSTATS
* compiler optimization NEWOPTFLG=T
* free code deleterefs pointers therefrom

Misc bugs
      Masterscope recursion with long names
BQUOTE

Date: 6-Oct-82 13:21:14 PDT (Wednesday)
From: Masinter.PA
Subject: questions & comments from Schoen
To: LispSupport
cc: Masinter.PA

document VRAID package; I think as a LispUsers package.

 we should provide FLOUT and READ/WRITEBINARYBITMAP for fast i/o of floatps and bitmaps.

Schoen has a VAXPRINT package (possibly similar to VanBuers) which causes the vax to hardcopy screen bitmaps. He doesn't know what the Versatec they use is exactly, but it has 2112 dots across, he says.

Date: 8 OCT 1982 1620-PDT
From: SHEIL.PA
Subject: Lisp task list
To: MASINTER
cc: lispcore↑

I just spent 15 mins reading it. Very comprehensive; fine job; don't envy

you the task of prioritizing it! A couple of additonal points:

ERRORSET compilation.
Possibility of marking the stack rather than introducing new function call. If not, the making compiler suppress or rename forced new specvars.
Error handling
Proposal to make ↑D work by ↑E (incompatible but worth it?)
Some improvement over current mess.
Failing that, debugging optimization of RESETLST/SAVE/FORM
Global vars
If RESETVARSLST is known not to be declared global, lets fix it rather than documenting it! Mike Sannella needs some help to get new manual to indicate which system params are global - maybe he could propose a list derived from the manual and we could dispose of this one.
Garbage collection
Compiled code blocks (pointers therefrom)
Hash arrays [urgent; current incompatibility + perf problem]
File package
Hasdef problems
Fixing EDIT interfaces to use same.

Beau

Date: 11-Oct-82 16:41:58 PDT (Monday)
From: vanMelle.PA
Subject: CASEARRAY for READ
To: LispSupport

Yet another (perhaps the same) request for (RAISE T) for files...

Mail-from: Arpanet host SU-SCORE rcvd at 11-OCT-82 1414-PDT
Date: 11 Oct 1982 1402-PDT
From: David E. Smith <CSD.SMITH at SU-SCORE>
Subject: lower case
To: vanmelle at PARC-MAXC
Stanford Phone: (415)497-1809

I need a way of forcing interlisp to be case independent for file input as well as terminal input. Read macros won't do it because I don't want the lower case letters to be break characters and "ALWAYS" forces this. Advising or rewriting READC presumably wouldn't work either because strings and characters prefaced by "%" would then get upcased.

How do I do this? Am I forced to rewrite LOAD and all of its accomplices? Crufty and/or release dependent solutions will not be sneezed at. Help!

-- de2

Date: 12-Oct-82 13:11:06 PDT (Tuesday)
From: vanMelle.PA
Subject: Font assumptions
To: Burton
cc: LispSupport

If you do (FONTSET 'STANDARD) to turn off fonts (e.g., to make a fontfree file), subsequent calls to the inspector die in a \DTEST of FONTDESCRIPTOR because DEFAULTFONT is NIL.

I wonder how many other places make such assumptions.

Incidentally, herewith a reminder that \DTESTFAIL desperately needs to produce a better error message, at least incorporating its second arg

(the intended type).

Bill

Date: 14-Oct-82 16:36:57 PDT (Thursday)
Subject: Re: Schlumberger URGENT
In-reply-to: Raim.EOS's message of 14 Oct. 1982 10:24 am PDT (Thursday)

Eric (and users in general) should avoid doing (APPLY 'IMAX LST) and instead write (for X in LST maximum X).

The limit of number of arguments to a function is indeed 80; it is possible that we could bump it, but there still would be a fixed limit.

Larry

Date: 15-Oct-82 15:05:36 PDT (Friday)
From: vanMelle.PA
Subject: LARGEST/SMALLEST
To: LispCore↑

It has been pointed out that these names are confusing, due to the ambiguity of what you might want the iterative to return. I propose that LARGEST be called MAXIMIZING, SMALLEST be MINIMIZING, and that there also me oprs MAXIMUM and MINIMUM. Since FIND is a synonym of FOR, we could thus have:

```
(find X in L maximizing (FOO X))
          returns the X for which FOO is largest, and
(for X in L maximum (FOO X))
          returns the largest value of FOO over L.
```

Bill