File created: 31-Mar-2024 09:38:10 **{DSK}<home>larry>il>medley>sources>AINTERRUPT.;7**

edit by: lmm

changes to: (VARS AINTERRUPTCOMS)

previous date: 31-Mar-2024 09:27:57 {DSK}<home>larry>il>medley>sources>AINTERRUPT.;5

Read Table: XCL

Package: INTERLISP

Format: XCCS

```
(RPAQQ AINTERRUPTCOMS
       ((COMS                                                    ; handling interrupts
              (FNS INTCHAR INTERRUPTCHAR INTERRUPTED LISPINTERRUPTS \\DOHELPINTERRUPT \\DOHELPINTERRUPT1
                   \\DOINTERRUPTHERE \\PROC.FINDREALFRAME \\SETPRINTLEVEL \\SETRECLAIMMIN GETINTERRUPT
                   CURRENTINTERRUPTS SETINTERRUPT RESET.INTERRUPTS INTERRUPTABLE))
        (INITVARS (LISPINTERRUPTS '((LISPINTERRUPTS (2 BREAK MOUSE)
                                                    (4 RESET MOUSE)
                                                    (5 ERROR MOUSE)
                                                    (7 HELP T)
                                                    (16 PRINTLEVEL)
                                                    (20 (CONTROL-T))
                                                    (127 RUBOUT T)))))
        (GLOBALVARS LISPINTERRUPTS)
        (COMS ;; ^T this is actually not very useful any more, and the percentages are wrong
              (FNS CONTROL-T \\CONTROL-T.PRINTRATIO)
              (INITVARS (\\CONTROL-T.DEPTH 3)
                     (\\CONTROL-T.BACKSLASH)
                     (LAST^TTIMEBOX (CLOCK 0))
                     (LAST^TSWAPTIME)
                     (LAST^TDISKIOTIME 0)
                     (LAST^TGCTIME 0)
                     (LAST^TNETIOTIME 0))
              (GLOBALVARS \\CONTROL-T.DEPTH \\CONTROL-T.BACKSLASH LAST^TTIMEBOX LAST^TSWAPTIME LAST^TDISKIOTIME
                     LAST^TNETIOTIME LAST^TGCTIME \\MISCSTATS)
              (ADDVARS (\\SYSTEMCACHEVARS LAST^TSWAPTIME)))
        (INITVARS (\\CURRENTINTERRUPTS)
               (\\INTERRUPTABLE)
               (INTERRUPTMENUFONT))
        (ADDVARS (FONTVARS (INTERUPTMENUFONT DEFAULTFONT T)))
        (VARS \\SYSTEMINTERRUPTS)
        (DECLARE\: EVAL@COMPILE DONTCOPY (ADDVARS (NOFIXFNSLST CONTROL-T))
               (LOCALVARS . T)
               (GLOBALVARS \\CURRENTINTERRUPTS \\SYSTEMINTERRUPTS INTERRUPTMENUFONT))
        (DECLARE\: EVAL@COMPILE (EXPORT (ADDVARS (SYSSPECVARS \\INTERRUPTABLE))
                                        (PROP INFO UNINTERRUPTABLY)
                                        (PROP DMACRO UNINTERRUPTABLY)
                                        (ALISTS (PRETTYPRINTMACROS UNINTERRUPTABLY)))
               DONTCOPY
               (EXPORT (RECORDS INTERRUPTSTATE)
                     (PROP DMACRO \\TAKEINTERRUPT))
               (MACROS \\SYSTEMINTERRUPTP))
        (DECLARE\: DONTEVAL@LOAD DOCOPY (P (INTCHAR T)))))

;; handling interrupts

(DEFINEQ

(INTCHAR
  (LAMBDA (CHAR TYP/FORM HARDFLG TABLE)                          ; Edited 31-Mar-2024 09:16 by lmm
                                                                 ; Edited 17-Sep-92 10:41 by jds
    ;; this function is the non-undoable version of INTERRUPTCHAR;  INTERRUPTCHAR calls it
    (PROG (VAL SYSDEF OLDINT)
          (SELECTQ CHAR
                (NIL                                             ; this is illegal, so don't do anything about it
                     (RETURN))
                (T ;; (INTCHAR T) means restore interrupts to the 'standard' setting
                   (|for| CHAR |in| (GETINTERRUPT NIL TABLE) |do| (SETQ VAL (NCONC (INTCHAR CHAR NIL NIL TABLE)
                                                                                    VAL)))
                                                                 ; turn off all user interrupts --- (GETINTERRUPT) returns list of
                                                                 ; user interrupts
                   (MAPC (LISPINTERRUPTS)
                         (FUNCTION (LAMBDA (LST)
                                     (SETQ VAL (NCONC (INTCHAR (CAR LST)
                                                               (CADR LST)
                                                               (CADDR LST)
                                                               TABLE)
                                                      VAL)))))
                   ;; and reset all SYSTEM interrupts to default --- (LISPINTERRUPTS) returns a list of argument lists for INTCHAR
                                                                 ; and VAL has been set to a valid arg list for INTCHAR
                   (RETURN VAL))
```

```
            NIL)
          (COND
             ((LISTP CHAR)                                          ; Call from undoing or resetform.  CHAR is a list of characters
                                                                    ; followed by typ/form arguments.
              (|while| CHAR |do| (SETQ VAL (NCONC (INTCHAR (|pop| CHAR)
                                                           (|pop| CHAR)
                                                           (|pop| CHAR)
                                                           TABLE)
                                                   VAL)))
              (RETURN VAL)))
          (COND
             ((NOT (FIXP CHAR))
              (COND
                 ((\\SYSTEMINTERRUPTP CHAR)
                  ;; CHAR can be an interrupt character class, meaning the character which is currently assigned to that interrupt --- this is most
                  ;; useful in, say, (INTCHAR (QUOTE HELP)) which says turn off the character whose class is HELP
                  (SETQ CHAR (OR (GETINTERRUPT CHAR TABLE)
                                 (ERRORX (LIST 27 CHAR)))))
                 (T                                                 ; turn single character into character code
                    (SETQ CHAR (APPLY* 'CHARCODE CHAR))))))
          (SETQ VAL (AND (SETQ OLDINT (GETINTERRUPT CHAR TABLE))
                         (LIST CHAR (CAR OLDINT)
                               (CADR OLDINT))))
          (COND
             ((EQ TYP/FORM T)                                       ; just return value indicating what it was.
              (RETURN VAL))
             ((AND TYP/FORM (LITATOM TYP/FORM)
                   (SETQ SYSDEF (ASSOC TYP/FORM \\SYSTEMINTERRUPTS)))
                                                                    ; System interrupt -- get its default HARDFLG
              (OR HARDFLG (SETQ HARDFLG (CADR SYSDEF)))))
          (COND
             ((AND (EQ (CAR OLDINT)
                       TYP/FORM)
                   (EQ (CADR OLDINT)
                       HARDFLG))                                    ; if the character is already set up, just return
              (RETURN)))
          (COND
             (OLDINT (SETINTERRUPT CHAR NIL TABLE)))
          (COND
             ((NULL TYP/FORM)                                       ; just leave character disabled

              )
             (T                                                     ; make a user interrupt
                (COND
                   ((AND SYSDEF (SETQ OLDINT (GETINTERRUPT TYP/FORM TABLE)))
                    ;; if a system interrupt and there is another character assigned to that channel, turn that character off
                    (SETINTERRUPT OLDINT NIL TABLE)
                    (|push| VAL OLDINT TYP/FORM NIL)))
                (SETINTERRUPT CHAR TYP/FORM TABLE HARDFLG)
                (|push| VAL CHAR NIL NIL)))
          (RETURN VAL))))
```

(INTERRUPTCHAR

```
(LAMBDA (CHAR TYP/FORM HARDFLG TABLE)                               (* |lmm| "14-May-85 16:56")
  (PROG ((VAL (INTCHAR CHAR TYP/FORM HARDFLG TABLE)))
        (AND LISPXHIST (UNDOSAVE (LIST 'INTERRUPTCHAR VAL NIL NIL TABLE)))
        (RETURN VAL))))
```

(INTERRUPTED

```
(LAMBDA NIL                                                         ; Edited 28-Jun-90 18:43 by jds
  ;; This function gets control whenever an "interrupt" of some sort is signalled to Lisp, apart from the timer and keyboard-I/O handling interrupts.  It
  ;; dispatches to the proper handler routine for the "hard-wired" interrupt types, and signals the appropriate soft interrupt for interrupt characters.
  (DECLARE (GLOBALVARS \\INTERRUPTSTATE)
           (USEDFREE \\MOUSEBUSY \\INTERRUPTABLE))
  (COND
     ((NULL \\INTERRUPTABLE)
      (SETQ \\PENDINGINTERRUPT T)
      (|replace| (INTERRUPTSTATE IN-PROGRESS) |of| \\INTERRUPTSTATE |with| 0))
     (T (COND
           ((|fetch| (INTERRUPTSTATE ETHERINTERRUPT) |of| \\INTERRUPTSTATE)
            (\\MAIKO.ETHER-INTERRUPT)
            (|replace| (INTERRUPTSTATE P-ETHERINTERRUPT) |of| \\INTERRUPTSTATE |with| NIL)))
           (COND
              ((|fetch| (INTERRUPTSTATE LOGMSGSPENDING) |of| \\INTERRUPTSTATE)
               (\\MAIKO.CONSOLE-LOG-PRINT)
               (|replace| (INTERRUPTSTATE P-LOGMSGSPENDING) |of| \\INTERRUPTSTATE |with| NIL)))
           (COND
              ((|fetch| (INTERRUPTSTATE IOINTERRUPT) |of| \\INTERRUPTSTATE)
               (\\MAIKO.IO-INTERRUPT)
               (|replace| (INTERRUPTSTATE P-IOINTERRUPT) |of| \\INTERRUPTSTATE |with| NIL)))
           (COND
              ((|fetch| (INTERRUPTSTATE STORAGEFULL) |of| \\INTERRUPTSTATE)
```

```
            (\\DOSTORAGEFULLINTERRUPT)
            (|replace| (INTERRUPTSTATE P-STORAGEFULL) |of| \\INTERRUPTSTATE |with| NIL))
           ((|fetch| (INTERRUPTSTATE STACKOVERFLOW) |of| \\INTERRUPTSTATE)
            (\\DOSTACKFULLINTERRUPT)
            (|replace| (INTERRUPTSTATE P-STACKOVERFLOW) |of| \\INTERRUPTSTATE |with| NIL))
           ((|fetch| (INTERRUPTSTATE VMEMFULL) |of| \\INTERRUPTSTATE)
            (\\DOVMEMFULLINTERRUPT)
            (|replace| (INTERRUPTSTATE P-VMEMFULL) |of| \\INTERRUPTSTATE |with| NIL))
           ((|fetch| (INTERRUPTSTATE GCDISABLED) |of| \\INTERRUPTSTATE)
            (\\DOGCDISABLEDINTERRUPT)
            (|replace| (INTERRUPTSTATE P-GCDISABLED) |of| \\INTERRUPTSTATE |with| NIL))
           ((|fetch| (INTERRUPTSTATE WAITINGINTERRUPT) |of| \\INTERRUPTSTATE)
            (LET* ((CH (|fetch| (INTERRUPTSTATE INTCHARCODE) |of| \\INTERRUPTSTATE))
                   (INTERRUPT (CDR (ASSOC CH (|fetch| (KEYACTION INTERRUPTLIST) |of| \\CURRENTKEYACTION)))))
                  (|replace| (INTERRUPTSTATE INTCHARCODE) |of| \\INTERRUPTSTATE |with| 0)
                  (COND
                     (INTERRUPT (LET* ((CLASS (CAR INTERRUPT))
                                       (HARDFLG (CADR INTERRUPT))
                                       (THISPROC (THIS.PROCESS))
                                       (INTERRUPTED.PROC (COND
                                                             ((OR (NULL THISPROC)
                                                                  (EQ HARDFLG T))
                                                              THISPROC)
                                                             ((EQ HARDFLG 'MOUSE)
                                                              (LET ((MP THISPROC))
                                                                ; Interrupt MOUSE proc if it's busy, else the tty process
                                                                   (COND
                                                                      ((COND
                                                                          ((EQ (PROCESSPROP MP 'NAME)
                                                                               'MOUSE)
                                                                           \\MOUSEBUSY)
                                                                          ((SETQ MP (FIND.PROCESS 'MOUSE))
                                                                           (PROCESS.EVALV MP '\\MOUSEBUSY)))
                                                                       MP)
                                                                      (T (TTY.PROCESS)))))
                                                             ((EQ HARDFLG 'WHICHW)
                                                                ; Interrupt the process that owns the window the mouse is in
                                                              (AND (GETD 'WHICHW)
                                                                   (LET ((W (WHICHW)))
                                                                      (AND W (WINDOWPROP W 'PROCESS)))))
                                                             (T (TTY.PROCESS))))))
                                  (COND
                                     ((EQ THISPROC INTERRUPTED.PROC)
                                      (|replace| (INTERRUPTSTATE WAITINGINTERRUPT) |of| \\INTERRUPTSTATE
                                         |with| NIL)
                                      (\\DOINTERRUPTHERE CLASS)
                                      (|replace| (INTERRUPTSTATE P-WAITINGINTERRUPT) |of| \\INTERRUPTSTATE
                                         |with| NIL))
                                     ((NULL INTERRUPTED.PROC)
                                                                ; Nobody qualified, so dismiss interrupt
                                      (|replace| (INTERRUPTSTATE WAITINGINTERRUPT) |of| \\INTERRUPTSTATE
                                         |with| NIL)
                                      (|replace| (INTERRUPTSTATE P-WAITINGINTERRUPT) |of| \\INTERRUPTSTATE
                                         |with| NIL)
                                      NIL)
                                     ((\\PROCESS.MAKEFRAME INTERRUPTED.PROC (FUNCTION \\DOINTERRUPTHERE)
                                             (LIST CLASS CH HARDFLG))
                                      (|replace| (INTERRUPTSTATE WAITINGINTERRUPT) |of| \\INTERRUPTSTATE
                                         |with| NIL)
                                      (|replace| (INTERRUPTSTATE P-WAITINGINTERRUPT) |of| \\INTERRUPTSTATE
                                         |with| NIL))
                                     (T                         ; Couldn't build frame, so leave interrupt pending
                                        (SETQ \\PENDINGINTERRUPT T)))))))))))
```

(**LISPINTERRUPTS**

```
(LAMBDA NIL                                                 ; Edited 31-Mar-2024 06:25 by lmm
                                                            (* |jds| "30-Sep-85 12:35")

        (* * |Returns| \a |list| |of| |the| "standard" |interrupt-character| |settings| |for| |Interlisp-D.|
        |These| |are| |used,| |e.g.,| |in| INTCHAR |to| |reset| |things| |to| |the| |default| |state.|)

   (FOR R IN LISPINTERRUPTS JOIN (APPEND (CDR R)))))
```

(**DOHELPINTERRUPT**

```
(LAMBDA NIL                                                 (* |bvm:| "27-JUL-83 18:37")
  (PROG (PROC)
        (COND
           ((NULL (THIS.PROCESS))
            (FLASHWINDOW)
            (\\DOHELPINTERRUPT1))
           ((NULL (SETQ PROC (PROGN (FLASHWINDOW)
                                    (\\SELECTPROCESS "Interrupt which process?"))))
                                                            (* |Interrupt| |declined|)
            NIL)
           ((EQ PROC (THIS.PROCESS))
```

```
            (\\DOHELPINTERRUPT1))
           ((\\PROCESS.MAKEFRAME PROC (FUNCTION \\DOHELPINTERRUPT1)))
           (T                                                    (* |Couldn't| |build| |frame,| |so| |leave| |interrupt| |pending|)
              (SETQ \\PENDINGINTERRUPT T))))))
```

(\\DOHELPINTERRUPT1

```
(LAMBDA NIL                                                      (* |bvm:| "11-AUG-83 11:56")

        (* |Does| HELP/BREAK |interrupt| |in| |the| |current| |process.| W\e |treat| ^B |same| |as| ^H\, |except| |that| |former| |always|
        |occurs| |in| |tty| |process.| BREAK |interrupt| |used| |to| |just| |do| \a
        (ERRORX (LIST 18 NIL)) |instead| |of| |calling| INTERRUPT)

   (COND
      ((NULL \\INTERRUPTABLE)                                    (* |Unlikely,| |but| |could| |occur| |if| |someone| |blocked| |while|
                                                                 |uninterruptable|)
       (FLASHWINDOW))
      (T (PROG (OLDTTY)
               (OR (TTY.PROCESSP)
                   (SETQ OLDTTY (TTY.PROCESS (THIS.PROCESS))))
               (COND
                  ((EQ (|fetch| PROCNAME |of| (THIS.PROCESS))
                       'MOUSE)
                   (SPAWN.MOUSE (THIS.PROCESS))))
               (CLEARBUF T T)                                    (* |Find| |name| |of| \a |real| |frame| |before| INTERRUPTED\,
                                                                 |so| |break| |message| |can| |be| |nice.|)
               (INTERRUPT (\\PROC.FINDREALFRAME)
                      NIL 2)
               (COND
                  (OLDTTY (TTY.PROCESS OLDTTY))))))))
```

(\\DOINTERRUPTHERE

```
(LAMBDA (CLASS)
   (DECLARE (USEDFREE \\INTERRUPTABLE))                          (* |bvm:| "18-Jul-85 12:37")

        (* * |Perform| |the| CLASS |interrupt| |in| |the| |currently| |running| |process|)

   (COND
      ((NOT \\INTERRUPTABLE)
       (SETQ \\PENDINGINTERRUPT T))
      (T (SELECTQ CLASS
              (RESET (\\CLEARSYSBUF T)
                     (RESET))
              (ERROR (\\CLEARSYSBUF T)
                     (SETERRORN 47)
                     (ERROR!))
              (HELP                                              (* |Does| \a ^B |in| |process| |selected| |by| |user|)
                    (\\DOHELPINTERRUPT))
              (BREAK (\\DOHELPINTERRUPT1))
              (CONTROL-T (CONTROL-T))
              (STORAGE (\\SETRECLAIMMIN))
              (PRINTLEVEL (\\SETPRINTLEVEL))
              (RUBOUT (FLASHWINDOW)
                      (\\CLEARSYSBUF T))
              (RAID (RAID))
              (COND
                 ((LITATOM CLASS)
                  (SET CLASS T))
                 (T (\\EVAL CLASS))))))))
```

(\\PROC.FINDREALFRAME

```
(LAMBDA (POS)                                                    (* |bvm:| "18-Jul-85 13:00")

        (* |Returns| |the| |name| |of| |the| |first| |interesting| |frame| |before| POS\, |or| |the| |caller| |if| POS = NIL)

   (|for| I |from| (COND
                     (POS 0)
                     (T -2))
      |by| -1 |do| (SELECTQ (SETQ $$VAL (STKNTHNAME I POS))
                       ((INTERRUPTED \\INTERRUPTFRAME \\INTERRUPTED \\DOHELPINTERRUPT \\DOHELPINTERRUPT1
                                \\DOBUFFEREDTRANSITIONS \\DOINTERRUPTHERE \\PROCESS.GO.TO.SLEEP BLOCK AWAIT.EVENT
                                MONITOR.AWAIT.EVENT GETMOUSESTATE)
                           NIL)
                       (RETURN $$VAL)))))
```

(\\SETPRINTLEVEL

```
(LAMBDA NIL                                                      (* |lmm| "30-Dec-85 17:08")
   (DECLARE (GLOBALVARS \\TCARPRINTLEVEL \\TCDRPRINTLEVEL))
   (PROG (BUF OLB OSB CARN)
         (\\BOUT \\TERM.OFD (CHARCODE BELL))
         (SETQ OLB (LINBUF T))
         (SETQ OSB (SYSBUF T))
         (CLEARBUF T T)
         (PRIN3 "set printlevel to: " T)
```

```
          (PROG ((N 0)
                 CH)
            LP   (SELCHARQ (SETQ CH (\\GETCHAR))
                           ((0 1 2 3 4 5 6 7 8 9)
                                (SETQ N (IPLUS (ITIMES N 10)
                                               (IDIFFERENCE CH (CHARCODE 0))))
                                (GO LP))
                           ((\. !)                                    (* CARN |is| |set| |if| |we've| |already| |seen| \a |comma|)
                                (COND
                                   (CARN (SETQ \\TCARPRINTLEVEL CARN)
                                         (SETQ \\TCDRPRINTLEVEL N))
                                   (T (SETQ \\TCARPRINTLEVEL N)))
                                (COND
                                   ((EQ CH (CHARCODE !))                (* |Make| |it| |permanent|)
                                    (PRINTLEVEL \\TCARPRINTLEVEL \\TCDRPRINTLEVEL))))
                           (\, (COND
                                  ((NOT CARN)
                                   (SETQ CARN N)                        (* |This| |is| |the| |first| |comma|)
                                   (SETQ N 0)
                                   (GO LP))))
                           NIL)                                         (* |Restore| |buffers| |cleared| |with| CLEARBUF)
            )
          (COND
             ((SETQ BUF (SYSBUF T))
              (BKSYSBUF BUF)))
          (SETQ \\SYSBUF OSB)
          (AND (SETQ BUF (LINBUF T))
               (LINBUF))
          (SETQ \\LINBUF OLB))))
```

(**SETRECLAIMMIN**

```
  (LAMBDA NIL                                                           (* |lmm| "30-Dec-85 17:08")
    (PROG (BUF OLB OSB CH)
          (\\BOUT \\TERM.OFD (CHARCODE BELL))
          (SETQ OLB (LINBUF T))
          (SETQ OSB (SYSBUF T))
          (CLEARBUF T T)
          (PRIN3 "set RECLAIMMIN to: " T)
          (PROG ((N 0))
            LP   (SELCHARQ (SETQ CH (\\GETCHAR))
                           ((0 1 2 3 4 5 6 7 8 9)
                                (SETQ N (IPLUS (ITIMES N 10)
                                               (IDIFFERENCE CH (CHARCODE 0))))
                                (GO LP))
                           (\. (RECLAIMMIN N))
                           NIL))
          (COND
             ((SETQ BUF (SYSBUF T))
              (BKSYSBUF BUF)))
          (SETQ \\SYSBUF OSB)
          (AND (SETQ BUF (LINBUF T))
               (LINBUF))
          (SETQ \\LINBUF OLB))))
```

(**GETINTERRUPT**

```
  (LAMBDA (CHAR TABLE)                                                  ; Edited 31-Mar-2024 09:20 by lmm
                                                                        ; Edited 17-Sep-92 10:41 by jds
```

;; Return the interrupt, if any, defined for CHAR in keyaction table TABLE.

;; NIL => all user interrupts

;; T => all system interrupts

```
    (OR TABLE (SETQ TABLE \\CURRENTKEYACTION))
    (SELECTQ CHAR
         (NIL                                                           ; Non-system interrupts
              (|for| X |in| (|fetch| (KEYACTION INTERRUPTLIST) OF TABLE) |unless| (\\SYSTEMINTERRUPTP (CADR X))
                 |collect| (CAR X)))
         (T                                                             ; All system interrupts
            (|for| X |in| (|fetch| (KEYACTION INTERRUPTLIST) OF TABLE) |collect| (CAR X)))
         (COND
            ((NUMBERP CHAR)
             (CDR (FASSOC CHAR (|fetch| (KEYACTION INTERRUPTLIST) OF TABLE))))
            (T (|for| X |in| (|fetch| (KEYACTION INTERRUPTLIST) OF TABLE) |when| (EQ CHAR (CADR X))
                  |do|                                                  ; Find CHAR in system class.
                       (RETURN (CAR X))))))))
```

(**CURRENTINTERRUPTS**

```
  (LAMBDA (TABLE)                                                       (* |bvm:| "18-Jul-85 12:37")
    (APPEND (|fetch| (KEYACTION INTERRUPTLIST) |of| (OR TABLE \\CURRENTKEYACTION)))))
```

(**SETINTERRUPT**

```
  (LAMBDA (CHAR CLASS TABLE HARDFLG)                                    ; Edited 20-Nov-87 11:00 by Snow
    (OR TABLE (SETQ TABLE \\CURRENTKEYACTION))
```

```
    (LET (TEM)
         ;; This code assumes that the variable (FETCH (KEYACTION INTERRUPTLIST) TABLE) is an alist of the form ((CHAR CLASS)(CHAR
         ;; CLASS) etc.)
         (COND
            ((NULL CHAR)                                          ; some mistake
             NIL)
            (((\\SYSTEMINTERRUPTP CHAR)                           ; If this is a system interrupt, then this is turning it off
             (SETINTERRUPT (GETINTERRUPT CHAR TABLE)
                    NIL TABLE))
            ((SETQ TEM (FASSOC CHAR (|fetch| (KEYACTION INTERRUPTLIST)
                                             TABLE)))             ; CHAR is currently an interrupt
             (COND
                ((AND (EQ (CADR TEM)
                          CLASS)
                      (EQ (CADDR TEM)
                          HARDFLG))                               ; No change
                 NIL)
                ((NULL CLASS)                                     ; REMOVE FROM INTERRUPT CHARACTER SET
                 (|change| (|fetch| (KEYACTION INTERRUPTLIST)
                                    TABLE)
                           (DREMOVE TEM DATUM)))
                (T                                                ; Assign new interrupt to CHAR
                   (|change| (CDR TEM)
                             (LIST CLASS HARDFLG)))))
            ((NULL CLASS))
            (T                                                    ; Brand new interrupt
               (|push| (|fetch| (KEYACTION INTERRUPTLIST)
                                TABLE)
                       (LIST CHAR CLASS HARDFLG)))))))
```

(RESET.INTERRUPTS

```
  (LAMBDA (|PermittedInterrupts| |SaveCurrent?|)
    (DECLARE (GLOBALVARS \\CURRENTKEYACTION))                     ; Edited 20-Nov-87 10:44 by Snow
    ;; Returns list of previous settings, for use by RESETFORM but only when 2nd arg is non-NIL.  --- PermittedInterrupts is a list of triples of the form
    ;; (charcode interrupt hardness)
    (COND
       (|PermittedInterrupts| (SETQ |PermittedInterrupts| (|for| TRIPLE |in| |PermittedInterrupts|
                                                             |collect| (COND
                                                                          ((OR (NLISTP TRIPLE)
                                                                               (NOT (CHARCODEP (CAR TRIPLE)))
                                                                               (NLISTP (CDR TRIPLE)))
                                                                           (\\ILLEGAL.ARG |PermittedInterrupts|))
                                                                          ((NLISTP (CDDR TRIPLE))
                                                                          ; Not a triple, so default the hardness to system hardness
                                                                           (LIST (CAR TRIPLE)
                                                                                 (CADR TRIPLE)
                                                                                 (CADR (ASSOC (CADR TRIPLE)
                                                                                              \\SYSTEMINTERRUPTS)))
                                                                          )
                                                                          (T TRIPLE))))))
    (UNINTERRUPTABLY
        (PROG1 (AND |SaveCurrent?| (|fetch| (KEYACTION INTERRUPTLIST) |of| \\CURRENTKEYACTION))
            (|replace| (KEYACTION INTERRUPTLIST) |of| \\CURRENTKEYACTION |with| |PermittedInterrupts|)))))
```

(INTERRUPTABLE

```
  (LAMBDA (FLAG)                                                  (* |lmm| "18-APR-82 13:52")
    (PROG1 \\INTERRUPTABLE (SETQ \\INTERRUPTABLE FLAG))))

)

(RPAQ? LISPINTERRUPTS
       '((LISPINTERRUPTS (2 BREAK MOUSE)
                 (4 RESET MOUSE)
                 (5 ERROR MOUSE)
                 (7 HELP T)
                 (16 PRINTLEVEL)
                 (20 (CONTROL-T))
                 (127 RUBOUT T))))

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS LISPINTERRUPTS)
)

;; ^T this is actually not very useful any more, and the percentages are wrong

(DEFINEQ
```

(CONTROL-T

```
  (LAMBDA (POS OUT)                                               ; Edited  6-Dec-86 04:57 by lmm
    (OR OUT (SETQ OUT (GETSTREAM PROMPTWINDOW 'OUTPUT)))
    (|if| (AND (HASTTYWINDOWP)
               (NEQ (TTYDISPLAYSTREAM)
```

```
                OUT)
            (WFROMDS (TTYDISPLAYSTREAM))
            (OPENWP (WFROMDS (TTYDISPLAYSTREAM))))
        |then| (FLASHWINDOW (TTYDISPLAYSTREAM)
                    1 10))
    (UNINTERRUPTABLY                                    ; UNINTERRUPTABLY only so you can't type ^T during ^T
        (PROG ((STKI (COND
                        ((STACKP POS)
                         0)
                        (T (SETQ POS 'CONTROL-T)
                           -3)))
               TEMP SWAPDELTA NETIODELTA DISKIODELTA GCDELTA KEYBOARDDELTA TOTALDELTA)
              (SETQ TEMP (STKNTHNAME STKI POS))
              (PRINTOUT OUT "Process: " (PROCESS.NAME (THIS.PROCESS))
                     ", ")
              (|printout| OUT (|do| (SELECTQ TEMP
                                        ((\\INTERRUPTFRAME \\INTERRUPTED INTERRUPTED \\DOINTERRUPTHERE)
                                                                    ; Skip over these
                                            (SETQ TEMP (STKNTHNAME (|add| STKI -1)
                                                                   POS)))
                                        ((\\GETCHAR \\GETKEY \\TTYBACKGROUND)
                                            (SETQ TEMP (STKNTHNAME (|add| STKI -1)
                                                                   POS))
                                            (SETQ $$VAL "wait in "))
                                        ((BLOCK \\BACKGROUND AWAIT.EVENT MONITOR.AWAIT.EVENT
                                                \\PROCESS.GO.TO.SLEEP)
                                                                    ; Forms of blocking
                                            (SETQ TEMP (STKNTHNAME (|add| STKI -1)
                                                                   POS))
                                            (SETQ $$VAL "waiting in "))
                                        (RETURN (OR $$VAL "in "))))))
              (|bind| (CNT _ 0) |do| (COND
                                        ((XCL::INTERESTING-FRAME-P TEMP)
                                         (PRIN2 TEMP OUT T)
                                         (COND
                                            ((EQ (|add| CNT 1)
                                                 \\CONTROL-T.DEPTH)
                                             (RETURN))
                                            (T (|printout| OUT " in "))))))
                                    (SETQ TEMP (STKNTHNAME (|add| STKI -1)
                                                           POS)))
              (COND
                 ((NULL LAST^TSWAPTIME)                     ; Just initialize the first time
                  (SETQ LAST^TTIMEBOX (CLOCK))
                  (SETQ LAST^TDISKIOTIME (|fetch| DISKIOTIME |of| \\MISCSTATS))
                  (SETQ LAST^TNETIOTIME (|fetch| NETIOTIME |of| \\MISCSTATS))
                  (SETQ LAST^TGCTIME (|fetch| GCTIME |of| \\MISCSTATS))
                  (SETQ LAST^TSWAPTIME (|fetch| SWAPWAITTIME |of| \\MISCSTATS)))
                 (T ;; calculates the amount of time spent not in disk wait since the last control-T.  Considers only time outside of key board wait.
                    (SETQ TOTALDELTA (IPLUS (IMINUS LAST^TTIMEBOX)
                                            (SETQ LAST^TTIMEBOX (\\CLOCK0 LAST^TTIMEBOX))))
                    (SETQ SWAPDELTA (IPLUS (IMINUS LAST^TSWAPTIME)
                                           (SETQ LAST^TSWAPTIME (|fetch| SWAPWAITTIME |of| \\MISCSTATS))))
                    (SETQ DISKIODELTA (IPLUS (IMINUS LAST^TDISKIOTIME)
                                             (SETQ LAST^TDISKIOTIME (|fetch| DISKIOTIME |of| \\MISCSTATS))))
                    (SETQ NETIODELTA (IPLUS (IMINUS LAST^TNETIOTIME)
                                            (SETQ LAST^TNETIOTIME (|fetch| NETIOTIME |of| \\MISCSTATS))))
                    (SETQ GCDELTA (IPLUS (IMINUS LAST^TGCTIME)
                                         (SETQ LAST^TGCTIME (|fetch| GCTIME |of| \\MISCSTATS))))
                    (\\CONTROL-T.PRINTRATIO SWAPDELTA TOTALDELTA "% Swap" NIL OUT)
                    (\\CONTROL-T.PRINTRATIO DISKIODELTA TOTALDELTA "% DskIO" NIL OUT)
                    (\\CONTROL-T.PRINTRATIO NETIODELTA TOTALDELTA "% Network" NIL OUT)
                    (\\CONTROL-T.PRINTRATIO GCDELTA TOTALDELTA "% GC" NIL OUT)))
              (TERPRI OUT)))))
```

(\\CONTROL-T.PRINTRATIO

```
  (LAMBDA (N TOTAL LABEL NEWLINE STREAM)                ; Edited  4-Dec-86 21:13 by lmm
    (COND
       ((NEQ N 0)
        (COND
           (NEWLINE (TERPRI STREAM))
           (T (|printout| STREAM ", ")))
        (COND
           ((OR (IGREATERP N TOTAL)
                (ILESSP N 0))
            (|printout| STREAM "??"))
           (T (|printout| STREAM .I2 (IQUOTIENT (ITIMES N 100)
                                                TOTAL))))
        (|printout| STREAM LABEL)))))

)

(RPAQ? \\CONTROL-T.DEPTH 3)

(RPAQ? \\CONTROL-T.BACKSLASH )
```

```
(RPAQ? LAST^TTIMEBOX (CLOCK 0))

(RPAQ? LAST^TSWAPTIME )

(RPAQ? LAST^TDISKIOTIME 0)

(RPAQ? LAST^TGCTIME 0)

(RPAQ? LAST^TNETIOTIME 0)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \\CONTROL-T.DEPTH \\CONTROL-T.BACKSLASH LAST^TTIMEBOX LAST^TSWAPTIME LAST^TDISKIOTIME
       LAST^TNETIOTIME LAST^TGCTIME \\MISCSTATS)
)

(ADDTOVAR \\SYSTEMCACHEVARS LAST^TSWAPTIME)

(RPAQ? \\CURRENTINTERRUPTS )

(RPAQ? \\INTERRUPTABLE )

(RPAQ? INTERRUPTMENUFONT )

(ADDTOVAR FONTVARS (INTERUPTMENUFONT DEFAULTFONT T))

(RPAQQ \\SYSTEMINTERRUPTS ((BREAK MOUSE)
                           (CONTROL-T)
                           (ERROR MOUSE)
                           (ERRORX)
                           (HELP T)
                           (OUTPUTBUFFER T)
                           (PRINTLEVEL)
                           (RAID T)
                           (RESET MOUSE)
                           (RUBOUT T)
                           (STORAGE)))

(DECLARE\: EVAL@COMPILE DONTCOPY

(ADDTOVAR NOFIXFNSLST CONTROL-T)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \\CURRENTINTERRUPTS \\SYSTEMINTERRUPTS INTERRUPTMENUFONT)
)
)

(DECLARE\: EVAL@COMPILE

;; FOLLOWING DEFINITIONS EXPORTED

(ADDTOVAR SYSSPECVARS \\INTERRUPTABLE)

(PUTPROPS UNINTERRUPTABLY INFO EVAL)

(PUTPROPS UNINTERRUPTABLY DMACRO ((X . Y)
                                  ((LAMBDA (\\INTERRUPTABLE)
                                     (PROGN X . Y))
                                   NIL)))

(ADDTOVAR PRETTYPRINTMACROS
          (UNINTERRUPTABLY
              LAMBDA
              (FORM)
              (PROG ((POS (IPLUS 4 (POSITION))))
                    (PRIN1 "(")
                    (PRIN2 (CAR FORM))
                    (OR (EQ COMMENTFLG (CAAR (SETQ FORM (CDR FORM))))
                        (TAB POS 0))
                    (PRINTDEF FORM POS T T FNSLST)
                    (PRIN1 ")"))))


;; END EXPORTED DEFINITIONS
;; FOLLOWING DEFINITIONS EXPORTED

(DECLARE\: EVAL@COMPILE

(BLOCKRECORD INTERRUPTSTATE (
```

```
                                ;; This is the structure used to communicate between the emulator and Lisp re interrupts.  There is a bit per
                                ;; interrupt type, plus space for the character code that caused a keyboard interrupt.

                                ;; This must match the INTSTAT definition in lispemul.h

                                ;; PENDING-INTERRUPT FLAGS:

                                (LOGMSGSPENDING FLAG)                ;  Log/Console msgs need printing.
                                (ETHERINTERRUPT FLAG)                ; Ether packet read finished.
                                (IOINTERRUPT FLAG)
                                (GCDISABLED FLAG)                    ; No mroe room in GC tables.
                                (VMEMFULL FLAG)                      ; VMEM is full!!
                                (STACKOVERFLOW FLAG)                 ; Stack overflowed.
                                (STORAGEFULL FLAG)                   ; Ran out of storage, atoms, etc.
                                (WAITINGINTERRUPT FLAG)

                                ;; INTERRUPTS-IN-PROCESS MASK:

                                (P-LOGMSGSPENDING FLAG)              ;  Log/Console msgs need printing.
                                (P-ETHERINTERRUPT FLAG)              ; Ether packet read finished.
                                (P-IOINTERRUPT FLAG)
                                (P-GCDISABLED FLAG)                  ; No mroe room in GC tables.
                                (P-VMEMFULL FLAG)                    ; VMEM is full!!
                                (P-STACKOVERFLOW FLAG)               ; Stack overflowed.
                                (P-STORAGEFULL FLAG)                 ; Ran out of storage, atoms, etc.
                                (P-WAITINGINTERRUPT FLAG)
                                (INTCHARCODE WORD))
        (BLOCKRECORD INTERRUPTSTATE (;; Alternative view of the structure:
                                     (PENDING BITS 8)                ; Pending-interrupt flags
                                     (IN-PROGRESS BITS 8)            ; Mask to prevent re-interrupt for an interrupt in progress
                                     (NIL WORD))))
)

(PUTPROPS \\TAKEINTERRUPT DMACRO ((PREFORM POSTFORM)
                                  (DECLARE (GLOBALVARS \\PENDINGINTERRUPT))
                                  (COND
                                     ((AND \\PENDINGINTERRUPT (INTERRUPTABLE~=NILUPTHESTACK))
                                      PREFORM
                                      ((LAMBDA (\\INTERRUPTABLE)
                                         (\\CALLINTERRUPTED))
                                       T)
                                      POSTFORM))))

;; END EXPORTED DEFINITIONS

(DECLARE\: EVAL@COMPILE

(PUTPROPS \\SYSTEMINTERRUPTP MACRO ((KEY)
                                    (ASSOC KEY \\SYSTEMINTERRUPTS)))
)
)

(DECLARE\: DONTEVAL@LOAD DOCOPY

(INTCHAR T)
)
```

FUNCTION INDEX

VARIABLE INDEX

MACRO INDEX

RECORD INDEX

PROPERTY INDEX