File created: 16-May-90 20:13:11 **{DSK}<usr>local>lde>lispcore>sources>LLTIMER.;2**

changes to: (VARS LLTIMERCOMS)

previous date: 13-May-88 15:29:44 {DSK}<usr>local>lde>lispcore>sources>LLTIMER.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

;;


```
(RPAQQ LLTIMERCOMS
       ([COMS
```

;;; Lowest level Clock stuff

```
              (FNS \CLOCK0 \DAYTIME0 \GETINTERNALCLOCK \SETDAYTIME0 CLOCKDIFFERENCE \SECONDSCLOCKGREATERP
                   \CLOCKGREATERP \RCLOCK0)
              (FNS CLOCK0)
              (OPTIMIZERS \RCLOCK0)
              (INITVARS (\RCLKMILLISECOND 1680))
              (GLOBALVARS \RCLKSECOND \RCLKMILLISECOND)
              [COMS ;; Maiko-specific elements
                    (FNS \MAIKO.DAYTIME \MAIKO.DAYTIME0 \MAIKO.CLOCK0 \MAIKO.CLOCK \MAIKO.COPY-TIME-STATS
                         \MAIKO.SETTIME)
                    ;; The elements of \MAIKO.MOVDS get movd by \MAIKO.FAULTINIT
                    (ADDVARS (\MAIKO.MOVDS (\MAIKO.DAYTIME DAYTIME)
                                           (\MAIKO.DAYTIME0 \DAYTIME0)
                                           (\MAIKO.CLOCK0 \CLOCK0)
                                           (\MAIKO.CLOCK0 CLOCK0)
                                           (\MAIKO.CLOCK CLOCK)
                                           (\MAIKO.SETTIME \NS.SETTIME)
                                           (\MAIKO.SETTIME \PUP.SETTIME)
                                           (\MAIKO.SETTIME SETTIME)
                                           (\MAIKO.COPY-TIME-STATS CL::%%COPY-TIME-STATS]
              (DECLARE%: DONTCOPY (EXPORT (MACROS \UPDATETIMERS)
                                                        ; Locations in alto emulator
                                          (CONSTANTS (\RTCSECONDS 378)
                                                     (\RTCMILLISECONDS 380)
                                                     (\RTCBASE 382)
                                                     (\OFFSET.SECONDS 0)
                                                     (\OFFSET.MILLISECONDS 2)
                                                     (\OFFSET.BASE 4)
                                                     (\ALTO.RCLKSECOND 1680000)
                                                     (\ALTO.RCLKMILLISECOND 1680)
                                                     (\DLION.RCLKMILLISECOND 35)
                                                     (\DLION.RCLKSECOND 34746)
                                                     (\DOVE.RCLKMILLISECOND 63)
                                                     (\DOVE.RCLKSECOND 62500)))
                    ;; Locked stuff.  Have to lock anything used by pagefault code, including the ufns that they use until all microcodes have
                    ;; them
                    (ADDVARS (INEWCOMS (ALLOCAL (ADDVARS (LOCKEDFNS \CLOCK0 \GETINTERNALCLOCK \BOXIDIFFERENCE
                                                            \BOXIPLUS \BLT \SLOWIQUOTIENT)
                                                         (LOCKEDVARS \RCLKSECOND \RCLKMILLISECOND \MISCSTATS]
       [COMS                                            ; basic date and time
              (FNS CLOCK DAYTIME ALTO.TO.LISP.DATE LISP.TO.ALTO.DATE)
              (DECLARE%: EVAL@COMPILE DONTCOPY (EXPORT (PROP MACRO ALTO.TO.LISP.DATE LISP.TO.ALTO.DATE]
       (COMS                                            ; DURATION and TIMER things
              (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS TIMER.MAKESAFETIMER TIMER.TIMEREXPIRED? EXPAND.SETUPTIMER
                                                       )
                                                        ; Following macro needn't be installed since the function call is
                                                        ; fairly slow anyway
                    (MACROS SETUPTIMER.DATE)
                    (FNS \SETUPTIMERmacrofn)))
       (COMS ;; macros for dealing with timers
              (MACROS SETUPTIMER)
              (MACROS \TIMER.TIMERP \TIMER.MAKETIMER \TIMER.PLUS \TIMER.DIFFERENCE \TIMER.IN.SECONDS
                      \TIMER.IN.MILLISECONDS \TIMER.IN.TICKS)
              (FNS \SETUPTIMERmacrofn \CanonicalizeTimerUnits)
              (FNS SETUPTIMER SETUPTIMER.DATE TIMEREXPIRED? TIME.UNTIL)
              (VARS (\TIMEREXPIRED.BOX (SETUPTIMER 0)))
              (GLOBALVARS \TIMEREXPIRED.BOX \RCLKMILLISECOND \RCLKSECOND))
       ;; Arrange for the proper compiler.
       (PROP FILETYPE LLTIMER)))
```

;;; Lowest level Clock stuff

```
(DEFINEQ

(\CLOCK0
  [LAMBDA (BOX)                                                    (* lmm "11-Sep-84 11:58")

          (* Stores millisecond clock in BOX. Do this by fetching the current millisecond clock and adding in the number of
          milliseconds since the clock was last updated)

    (SETQ BOX (\DTEST BOX 'FIXP))
    (UNINTERRUPTABLY
        (\GETINTERNALCLOCK \OFFSET.MILLISECONDS BOX)
        [bind (EXCESS _ (LOCF (fetch EXCESSTIMETMP of \MISCSTATS))) while (OR (IGREATERP EXCESS \RCLKSECOND)
                                                                                (ILESSP EXCESS 0))
          do

          (* Excess time. unsigned, is more than a second, so clock has not been updated in ages
          (perhaps someone sat in Raid for a while)%. We don't want IQUOTIENT here to do a CREATECELL, so do some of the
          division by subtraction. Instead of \RCLKSECOND, it would really be better to use
          \RCLKMILLISECOND*MAX.SMALL.INTEGER, but this is a rare case already, so be lazy)

            (\BOXIPLUS BOX 1000)
            (\BOXIDIFFERENCE EXCESS \RCLKSECOND)
          finally                                                  (* Now it is safe to use IQUOTIENT)
              (RETURN (\BOXIPLUS BOX (IQUOTIENT (COND
                                                  ((IGREATERP EXCESS MAX.SMALL.INTEGER)
                                                   EXCESS)
                                                  (T (fetch (FIXP LONUM) of EXCESS)))
                                          \RCLKMILLISECOND])])

(\DAYTIME0
  [LAMBDA (BOX)                                                    (* bvm%: "24-JUN-82 15:39")
    (UNINTERRUPTABLY
        (\GETINTERNALCLOCK \OFFSET.SECONDS (\DTEST BOX 'FIXP)))])

(\GETINTERNALCLOCK
  [LAMBDA (CLOCKOFFSET BOX)                                        (* bvm%: "24-JUN-82 15:39")

          (* Stores in BOX the contents of internal timer denoted by CLOCKOFFSET
          (0 = SECONDS, 2 = MILLISECONDS)%. Excess time is in EXCESSTIMETEMP.
          Must be called UNINTERRUPTABLY)

    (\BLT (LOCF (fetch SECONDSTMP of \MISCSTATS))
          (LOCF (fetch SECONDSCLOCK of \MISCSTATS))
          (UNFOLD 3 WORDSPERCELL))                                 (* Copy system clocks into scratch area, so there is no update
                                                                   conflict)
    (\BLT BOX (\ADDBASE (LOCF (fetch SECONDSTMP of \MISCSTATS))
                        CLOCKOFFSET)
          WORDSPERCELL)                                            (* Copy clock to caller)
    (\BOXIDIFFERENCE (\RCLK (LOCF (fetch EXCESSTIMETMP of \MISCSTATS)))
           (LOCF (fetch BASETMP of \MISCSTATS)))                   (* Compute processor time since clock was updated)
    BOX])

(\SETDAYTIME0
  [LAMBDA (BOX)                                                    (* bvm%: " 8-Jul-85 20:39")
                                                                   (* Sets the seconds calendar to contents of BOX)
    (SETQ BOX (\DTEST BOX 'FIXP))
    (UNINTERRUPTABLY
        (\RCLK (LOCF (fetch BASETMP of \MISCSTATS)))

          (* Reset the base; clocks will not be adjusted for at least a second after this)

        (\BLT (LOCF (fetch SECONDSTMP of \MISCSTATS))
              BOX WORDSPERCELL)
        [LET [(TMP (ITIMES 1000 (fetch (FIXP LONUM) of BOX]

          (* Need to set msecs clock to 1000 * secs clock, but try not to do too much bignum arithmetic...)

            (replace (FIXP LONUM) of (LOCF (fetch MILLISECONDSTMP of \MISCSTATS)) with (LOGAND TMP MAX.SMALLP))
            (replace (FIXP HINUM) of (LOCF (fetch MILLISECONDSTMP of \MISCSTATS))
               with (IPLUS16 (LRSH TMP BITSPERWORD)
                             (LOGAND (ITIMES (fetch (FIXP HINUM) of BOX)
                                             1000)
                                     MAX.SMALLP]
        (\BLT (LOCF (fetch SECONDSCLOCK of \MISCSTATS))
              (LOCF (fetch SECONDSTMP of \MISCSTATS))
              (UNFOLD 3 WORDSPERCELL))                             (* Finally store them all at once, uninterruptably)
        [COND
           ((EQ \MACHINETYPE \DANDELION)                           (* Tell the iop the new time, too)
            (repeatwhile (IGEQ (fetch DLPROCESSORCMD of \IOPAGE)
                               \DL.PROCESSORBUSY))
            (replace DLPROCESSOR2 of \IOPAGE with (\GETBASE BOX 1))
            (replace DLPROCESSOR1 of \IOPAGE with (\GETBASE BOX 0))
            (replace DLPROCESSORCMD of \IOPAGE with \DL.SETTOD)
```

```
            (replace DLTODVALID of \IOPAGE with 0)
            (repeatwhile (IGEQ (fetch DLPROCESSORCMD of \IOPAGE)
                               \DL.PROCESSORBUSY))
            (repeatwhile (EQ (fetch DLTODVALID of \IOPAGE)
                             0]
        (\PROCESS.RESET.TIMERS))
    BOX])
```

(**CLOCKDIFFERENCE**

```
  [LAMBDA (OLDCLOCK)                                          (* bvm%: "24-JUN-82 15:40")
    (UNINTERRUPTABLY
        (IPLUS (\BOXIDIFFERENCE (\CLOCK0 (LOCF (fetch CLOCKTEMP0 of \MISCSTATS)))
                       OLDCLOCK)))])
```

(**\SECONDSCLOCKGREATERP**

```
  [LAMBDA (OLDCLOCK SECONDS)                                  (* bvm%: " 7-Dec-83 15:27")
    (UNINTERRUPTABLY
        (\BLT (LOCF (fetch CLOCKTEMP0 of \MISCSTATS))
              (LOCF (fetch SECONDSCLOCK of \MISCSTATS))
              WORDSPERCELL)
        (IGREATERP (\BOXIDIFFERENCE (LOCF (fetch CLOCKTEMP0 of \MISCSTATS))
                           OLDCLOCK)
               SECONDS))])
```

(**\CLOCKGREATERP**

```
  [LAMBDA (OLDCLOCK MSECS)                                    (* bvm%: "17-Dec-83 16:38")
```

(* * True if more than MSECS milliseconds have elapsed since OLDCLOCK was set)

```
    (UNINTERRUPTABLY
        (IGREATERP (\BOXIDIFFERENCE (\CLOCK0 (LOCF (fetch CLOCKTEMP0 of \MISCSTATS)))
                           OLDCLOCK)
               MSECS))])
```

(**\RCLOCK0**

```
  [LAMBDA (BOX)                                               (* JonL "19-APR-83 01:47")
    (\RCLK (\DTEST BOX 'FIXP])

)

(DEFINEQ
```

(**CLOCK0**

```
  [LAMBDA (BOX)                                               (* bvm%: " 1-APR-83 15:26")
```

(* Store millisecond clock at BOX. Unfortunately, there are still a few folks that call this without a true box, so accomodate them for now)

```
    (COND
       ((EQ (NTYPX BOX)
            \FIXP)
        (\CLOCK0 BOX))
       (T (\MP.ERROR \MP.CLOCK0 "Call to CLOCK0 with arg not a number box.  ^N to continue." BOX)
          (UNINTERRUPTABLY
              (\BLT BOX (\CLOCK0 (LOCF (fetch CLOCKTEMP0 of \MISCSTATS)))
                    WORDSPERCELL))
          BOX])

)


(DEFOPTIMIZER \RCLOCK0 (BOX)
                       `(\RCLK (\DTEST ,BOX 'FIXP)))

(RPAQ? \RCLKMILLISECOND 1680)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \RCLKSECOND \RCLKMILLISECOND)
)
```

;; Maiko-specific elements

```
(DEFINEQ
```

(**\MAIKO.DAYTIME**

```
  [LAMBDA (BOX)                                               ; Edited  2-May-88 16:20 by MASINTER
    (SUBRCALL GETUNIXTIME 5 BOX])
```

(**\MAIKO.DAYTIME0**

```
  [LAMBDA (BOX)                                               ; Edited  2-May-88 16:20 by MASINTER
    (SUBRCALL GETUNIXTIME 4 BOX])
```

```
(\MAIKO.CLOCK0
  [LAMBDA (BOX)                                          ; Edited  2-May-88 16:19 by MASINTER
    (SUBRCALL GETUNIXTIME 0 BOX])

(\MAIKO.CLOCK
  [LAMBDA (N BOX)                                        ; Edited  2-May-88 16:11 by MASINTER
    (SUBRCALL GETUNIXTIME N BOX])

(\MAIKO.COPY-TIME-STATS
  [LAMBDA (REFERENCE-BLOCK DESTINIATION-BLOCK)           ; Edited  2-May-88 17:16 by MASINTER
    (SUBRCALL COPYTIMESTATS REFERENCE-BLOCK DESTINIATION-BLOCK])

(\MAIKO.SETTIME
  [LAMBDA (RETFLG)                                       ; Edited 13-May-88 15:22 by MASINTER
    (CL:UNLESS (AND RETFLG (NOT (STRINGP RETFLG)))
        (SETQ \TimeZoneComp (SUBRCALL GETUNIXTIME 8 NIL)))
    (\PROCESS.RESET.TIMERS)
    (DAYTIME])

)

;; The elements of \MAIKO.MOVDS get movd by \MAIKO.FAULTINIT

(ADDTOVAR \MAIKO.MOVDS (\MAIKO.DAYTIME DAYTIME)
                       (\MAIKO.DAYTIME0 \DAYTIME0)
                       (\MAIKO.CLOCK0 \CLOCK0)
                       (\MAIKO.CLOCK0 CLOCK0)
                       (\MAIKO.CLOCK CLOCK)
                       (\MAIKO.SETTIME \NS.SETTIME)
                       (\MAIKO.SETTIME \PUP.SETTIME)
                       (\MAIKO.SETTIME SETTIME)
                       (\MAIKO.COPY-TIME-STATS CL::%%COPY-TIME-STATS))

(DECLARE%: DONTCOPY

;; FOLLOWING DEFINITIONS EXPORTED

(DECLARE%: EVAL@COMPILE

(PUTPROPS \UPDATETIMERS MACRO [NIL

          (* * Moves excess time from the processor clock to our software clocks.
          Needs to be run often, uninterruptably, preferably from the vertical retrace interrupt)
                                                  (* Get processor clock)
                                    (PROG [(EXCESS (\BOXIDIFFERENCE (\RCLK (LOCF (fetch RCLKTEMP0 of \MISCSTATS))
                                                                           )
                                                                    (LOCF (fetch BASECLOCK of \MISCSTATS]
                                          (RETURN (COND
                                                     ((OR (IGEQ EXCESS \RCLKSECOND)
                                                          (ILESSP EXCESS 0))
                                                                  (* More than one second has elapsed since we updated clocks)
                                                      (\BOXIPLUS (LOCF (fetch BASECLOCK of \MISCSTATS))
                                                                 \RCLKSECOND)
                                                                  (* Increment base by one second)
                                                      (\BOXIPLUS (LOCF (fetch MILLISECONDSCLOCK of \MISCSTATS))
                                                                 1000)
                                                                  (* Increment clocks by 1 second)
                                                      (\BOXIPLUS (LOCF (fetch SECONDSCLOCK of \MISCSTATS))
                                                                 1)
                                                      T])
)

(DECLARE%: EVAL@COMPILE

(RPAQQ \RTCSECONDS 378)

(RPAQQ \RTCMILLISECONDS 380)

(RPAQQ \RTCBASE 382)

(RPAQQ \OFFSET.SECONDS 0)

(RPAQQ \OFFSET.MILLISECONDS 2)

(RPAQQ \OFFSET.BASE 4)

(RPAQQ \ALTO.RCLKSECOND 1680000)

(RPAQQ \ALTO.RCLKMILLISECOND 1680)

(RPAQQ \DLION.RCLKMILLISECOND 35)
```

```
(RPAQQ \DLION.RCLKSECOND 34746)

(RPAQQ \DOVE.RCLKMILLISECOND 63)

(RPAQQ \DOVE.RCLKSECOND 62500)

(CONSTANTS (\RTCSECONDS 378)
       (\RTCMILLISECONDS 380)
       (\RTCBASE 382)
       (\OFFSET.SECONDS 0)
       (\OFFSET.MILLISECONDS 2)
       (\OFFSET.BASE 4)
       (\ALTO.RCLKSECOND 1680000)
       (\ALTO.RCLKMILLISECOND 1680)
       (\DLION.RCLKMILLISECOND 35)
       (\DLION.RCLKSECOND 34746)
       (\DOVE.RCLKMILLISECOND 63)
       (\DOVE.RCLKSECOND 62500))
)

;; END EXPORTED DEFINITIONS

(ADDTOVAR INEWCOMS (ALLOCAL (ADDVARS (LOCKEDFNS \CLOCK0 \GETINTERNALCLOCK \BOXIDIFFERENCE \BOXIPLUS \BLT
                                             \SLOWIQUOTIENT)
                                     (LOCKEDVARS \RCLKSECOND \RCLKMILLISECOND \MISCSTATS))))
)

;; basic date and time

(DEFINEQ

(CLOCK
  [LAMBDA (N BOX)                                                    (* lmm "15-OCT-82 11:44")
    (SELECTQ (OR N 0)
         (0                                                          (* time of day in MS)
            [\CLOCK0 (COND
                        ((type? FIXP BOX)
                         BOX)
                        (T (CREATECELL \FIXP])
         (1                                                          (* time this VM was started)
            (fetch STARTTIME of \MISCSTATS))
         (2                                                          (* run time for this VM)
            (\BOXIDIFFERENCE (\BOXIDIFFERENCE (\BOXIDIFFERENCE (\BOXIDIFFERENCE [\CLOCK0 (COND
                                                                                          ((type? FIXP BOX)
                                                                                           BOX)
                                                                                          (T (CREATECELL \FIXP
                                                                                                         ]
                                                                                (LOCF (fetch SWAPWAITTIME of \MISCSTATS)))
                                                                     (LOCF (fetch KEYBOARDWAITTIME of \MISCSTATS)))
                                                          (LOCF (fetch STARTTIME of \MISCSTATS)))
                                               (LOCF (fetch GCTIME of \MISCSTATS))))
         (3                                                          (* GC TIME)
            (fetch GCTIME of \MISCSTATS))
         (\ILLEGAL.ARG N])

(DAYTIME
  [LAMBDA NIL                                                        (* bvm%: " 8-Jul-85 20:01")
    (ALTO.TO.LISP.DATE (\DAYTIME0 (CREATECELL \FIXP])

(ALTO.TO.LISP.DATE
  [LAMBDA (DATE)                                                     (* bvm%: "18-FEB-81 00:35")

          (* DATE is a 32-bit unsigned integer. To avoid signbit lossage, we subtract MIN.INTEGER from DATE, thereby making day
          0 in the middle of the range. Do this by toggling the high-order bit to avoid integer overflow.)

    (LOGXOR DATE -2147483648])

(LISP.TO.ALTO.DATE
  [LAMBDA (DATE)                                                     (* bvm%: "18-FEB-81 00:35")
    (LOGXOR DATE -2147483648])

)

(DECLARE%: EVAL@COMPILE DONTCOPY

;; FOLLOWING DEFINITIONS EXPORTED

(PUTPROPS ALTO.TO.LISP.DATE MACRO ((DATE)
                                   (LOGXOR DATE -2147483648)))

(PUTPROPS LISP.TO.ALTO.DATE MACRO ((DATE)
                                   (LOGXOR DATE -2147483648)))
```

```
)

;; END EXPORTED DEFINITIONS
;; DURATION and TIMER things

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS TIMER.MAKESAFETIMER DMACRO (OPENLAMBDA (TIMER BOX)
                                       (\PUTBASEFIXP BOX 0 TIMER)
                                       BOX))

(PUTPROPS TIMER.TIMEREXPIRED? DMACRO ((OLDTIMER INTERVAL)
                                      (UNINTERRUPTABLY
                                          (IGEQ (\BOXIDIFFERENCE OLDTIMER INTERVAL)
                                                0))))

(PUTPROPS EXPAND.SETUPTIMER MACRO (L (\SETUPTIMERmacrofn L T)))
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS SETUPTIMER.DATE MACRO ((DTS TIMER)
                                 (SETUPTIMER (IDIFFERENCE (IDATE DTS)
                                                          (IDATE))
                                        TIMER
                                        'SECONDS
                                        'SECONDS)))
)

(DEFINEQ
```

(\SETUPTIMERmacrofn

```
  [LAMBDA (X NOERRORCHKS)                                    (* lmm "12-Apr-85 13:46")
    (PROG ((INTERVALFORM (CAR X))
           (TIMERFORM (CADR X))
           (TimerUnits (CONSTANTEXPRESSIONP (CADDR X)))
           (IntervalUnits (CONSTANTEXPRESSIONP (CADDDR X)))
           (CLOCKFNNAME))
          (if (OR (NULL TimerUnits)
                  (NULL IntervalUnits))
              then
```

(* If either of the units are true computibles, then we can't select clock functions at macroexpansion time.)

```
                (RETURN 'IGNOREMACRO))
          (SETQ TimerUnits (CANONICAL.TIMERUNITS (CAR TimerUnits)))
          [SETQ IntervalUnits (if (NULL (CAR IntervalUnits))
                                  then TimerUnits
                                else (CANONICAL.TIMERUNITS (CAR IntervalUnits]
```

(* Notice how the following SELECTQ may also modify the code expression for the INTERVALFORM to do any necessary transformations between the specifiend timer units and the specified interval units.)

```
          (SETQ CLOCKFNNAME (SELECTQ TimerUnits
                                (TICKS (SELECTQ IntervalUnits
                                           ((MILLISECONDS)
                                               (SETQ INTERVALFORM `(ITIMES %, INTERVALFORM \RCLKMILLISECOND)))
                                           ((SECONDS)
                                               (SETQ INTERVALFORM `(ITIMES %, INTERVALFORM \RCLKSECOND)))
                                           NIL)
                                       '\TIMER.IN.TICKS)
                                (MILLISECONDS (SELECTQ IntervalUnits
                                                  (TICKS (SETQ INTERVALFORM `(IQUOTIENT %, INTERVALFORM
                                                                                \RCLKMILLISECOND)))
                                                  (SECONDS (SETQ INTERVALFORM `(ITIMES %, INTERVALFORM 1000)))
                                                  NIL)
                                              '\TIMER.IN.MILLISECONDS)
                                (SECONDS (SELECTQ IntervalUnits
                                             (MILLISECONDS (SETQ INTERVALFORM `(IQUOTIENT %, INTERVALFORM 1000)
                                                                 ))
                                             (TICKS (SETQ INTERVALFORM `(IQUOTIENT %, INTERVALFORM \RCLKSECOND)
                                                            ))
                                             NIL)
                                         '\TIMER.IN.SECONDS)
                                (SHOULDNT)))
          [if (NOT NOERRORCHKS)
              then (SETQ TIMERFORM (if (CONSTANTEXPRESSIONP TIMERFORM)
                                       then '(\TIMER.MAKETIMER)
                                     else (LET [(FORM '(COND
                                                          ((\TIMER.TIMERP Timer?)
                                                           Timer?)
                                                          (T (\TIMER.MAKETIMER]
                                                (if (NLISTP TIMERFORM)
                                                    then (SUBST TIMERFORM 'Timer? FORM)
```

```
                                          else `([LAMBDA (Timer?)
                                                   (DECLARE (LOCALVARS Timer?))
                                                   %, FORM]
                                                 %, TIMERFORM]
          (RETURN `(\TIMER.PLUS (%, CLOCKFNNAME %, TIMERFORM)
                        %, INTERVALFORM])

)
)

;; macros for dealing with timers

(DECLARE%: EVAL@COMPILE

(PUTPROPS SETUPTIMER MACRO (X (\SETUPTIMERmacrofn X)))
)

(DECLARE%: EVAL@COMPILE

[PROGN (PUTPROPS \TIMER.TIMERP MACRO ((X)
                                      (FIXP X)))
       (PUTPROPS \TIMER.TIMERP DMACRO ((X)
                                       (TYPENAMEP X 'FIXP)))]

(PUTPROPS \TIMER.MAKETIMER DMACRO (NIL (NCREATE 'FIXP)))

(PUTPROPS \TIMER.PLUS DMACRO ((OLDTIMER INTERVAL)
                              (\BOXIPLUS OLDTIMER INTERVAL)))

(PUTPROPS \TIMER.DIFFERENCE DMACRO ((TIMER2 TIMER1)
                                    (IDIFFERENCE TIMER2 TIMER1)))

(PUTPROPS \TIMER.IN.SECONDS DMACRO ((OLDTIMER)
                                    (\DAYTIME0 OLDTIMER)))

(PUTPROPS \TIMER.IN.MILLISECONDS DMACRO ((OLDTIMER)
                                         (\CLOCK0 OLDTIMER)))

(PUTPROPS \TIMER.IN.TICKS DMACRO ((OLDTIMER)
                                  (\RCLOCK0 OLDTIMER)))
)

(DEFINEQ
```

(\SETUPTIMERmacrofn

```
  [LAMBDA (X NOERRORCHKS)                                    (* lmm "12-Apr-85 13:46")
    (PROG ((INTERVALFORM (CAR X))
           (TIMERFORM (CADR X))
           (TimerUnits (CONSTANTEXPRESSIONP (CADDR X)))
           (IntervalUnits (CONSTANTEXPRESSIONP (CADDDR X)))
           (CLOCKFNNAME))
          (if (OR (NULL TimerUnits)
                  (NULL IntervalUnits))
              then
```

(* If either of the units are true computibles, then we can't select clock functions at macroexpansion time.)

```
                (RETURN 'IGNOREMACRO))
          (SETQ TimerUnits (CANONICAL.TIMERUNITS (CAR TimerUnits)))
          [SETQ IntervalUnits (if (NULL (CAR IntervalUnits))
                                  then TimerUnits
                                else (CANONICAL.TIMERUNITS (CAR IntervalUnits]
```

(* Notice how the following SELECTQ may also modify the code expression for the INTERVALFORM to do any necessary transformations between the specifiend timer units and the specified interval units.)

```
          (SETQ CLOCKFNNAME (SELECTQ TimerUnits
                                (TICKS (SELECTQ IntervalUnits
                                           ((MILLISECONDS)
                                               (SETQ INTERVALFORM `(ITIMES %, INTERVALFORM \RCLKMILLISECOND)))
                                           ((SECONDS)
                                               (SETQ INTERVALFORM `(ITIMES %, INTERVALFORM \RCLKSECOND)))
                                           NIL)
                                       '\TIMER.IN.TICKS)
                                (MILLISECONDS (SELECTQ IntervalUnits
                                                  (TICKS (SETQ INTERVALFORM `(IQUOTIENT %, INTERVALFORM
                                                                                \RCLKMILLISECOND)))
                                                  (SECONDS (SETQ INTERVALFORM `(ITIMES %, INTERVALFORM 1000)))
                                                  NIL)
                                              '\TIMER.IN.MILLISECONDS)
                                (SECONDS (SELECTQ IntervalUnits
                                             (MILLISECONDS (SETQ INTERVALFORM `(IQUOTIENT %, INTERVALFORM 1000)
                                                           ))
                                             (TICKS (SETQ INTERVALFORM `(IQUOTIENT %, INTERVALFORM \RCLKSECOND)
                                                    ))
                                             NIL)
                                         '\TIMER.IN.SECONDS)
```

```
                                (SHOULDNT)))
          [if (NOT NOERRORCHKS)
              then (SETQ TIMERFORM (if (CONSTANTEXPRESSIONP TIMERFORM)
                                       then '(\TIMER.MAKETIMER)
                                     else (LET [(FORM '(COND
                                                        ((\TIMER.TIMERP Timer?)
                                                         Timer?)
                                                        (T (\TIMER.MAKETIMER]
                                                (if (NLISTP TIMERFORM)
                                                    then (SUBST TIMERFORM 'Timer? FORM)
                                                  else `([LAMBDA (Timer?)
                                                           (DECLARE (LOCALVARS Timer?))
                                                           %, FORM]
                                                         %, TIMERFORM]
          (RETURN `(\TIMER.PLUS (%, CLOCKFNNAME %, TIMERFORM)
                                %, INTERVALFORM])
```

(\CanonicalizeTimerUnits

```
  [LAMBDA (X)                                                    (* lmm "12-Apr-85 13:09")
```

(* Generally, the U-CASE versions have been "beat out" by the CANONICAL.TIMERUNITS.FOR.MISC macro; but there are ocasional calls to this function directly such, as in \DURATIONTRAN and the TIMEREXPIRED? macro.)

```
    (PROG ((Y X)
           CONVERTEDP)
      A    (RETURN (SELECTQ Y
                            (TICKS 'TICKS)
                            ((NIL MILLISECONDS MS)
                                'MILLISECONDS)
                            (SECONDS 'SECONDS)
                            (if (NOT CONVERTEDP)
                                then (SETQ Y (U-CASE Y))
                                     (SETQ CONVERTEDP T)
                                     (GO A)
                              else (ERROR '|Invalid arg for timer units| X])

)

(DEFINEQ
```

(SETUPTIMER

```
  [LAMBDA (INTERVAL OldTimer? timerUnits intervalUnits)          (* lmm "12-Apr-85 13:19")
    (SETQ INTERVAL (IPLUS INTERVAL 0))
```

(* If an error or coercion is to occur on this one, do it before the call to the clock-funciton)

```
    (if (NOT (\TIMER.TIMERP OldTimer?))
        then (SETQ OldTimer? (\TIMER.MAKETIMER)))
    (SETQ timerUnits (CANONICAL.TIMERUNITS timerUnits))
    (SETQ intervalUnits (if (NULL intervalUnits)
                            then timerUnits
                          else (CANONICAL.TIMERUNITS intervalUnits)))
```

(* Notice that in each wing of the SELECTQ below, the modification to INTERVAL is done before the clock-function call implicit in SETUPTIMER)

```
    (SELECTQ timerUnits
        ((TICKS)
            (SELECTQ intervalUnits
                ((MILLISECONDS)
                    (SETQ INTERVAL (ITIMES \RCLKMILLISECOND INTERVAL)))
                ((SECONDS)
                    (SETQ INTERVAL (ITIMES \RCLKSECOND INTERVAL)))
                NIL)
            (EXPAND.SETUPTIMER INTERVAL OldTimer? 'TICKS))
        ((MILLISECONDS)
            (SELECTQ intervalUnits
                ((TICKS)
                    (SETQ INTERVAL (IQUOTIENT INTERVAL \RCLKMILLISECOND)))
                ((SECONDS)
                    (SETQ INTERVAL (ITIMES 1000 INTERVAL)))
                NIL)
            (EXPAND.SETUPTIMER INTERVAL OldTimer? 'MILLISECONDS))
        ((SECONDS)
            (SELECTQ intervalUnits
                ((MILLISECONDS)
                    (SETQ INTERVAL (IQUOTIENT INTERVAL 1000)))
                ((TICKS)
                    (SETQ INTERVAL (IQUOTIENT INTERVAL \RCLKSECOND)))
                NIL)
            (EXPAND.SETUPTIMER INTERVAL OldTimer? 'SECONDS))
        (SHOULDNT])
```

(SETUPTIMER.DATE

```
  [LAMBDA (DTS OldTimer?)                                        (* Pavel " 6-Oct-86 21:46")
```

```
    (SETUPTIMER (IDIFFERENCE (IDATE DTS)
                             (IDATE))
           OldTimer?
           'SECONDS
           'SECONDS])
```

(TIMEREXPIRED?

```
  [LAMBDA (TIMER ClockValue.or.timerUnits)                    (* lmm "12-Apr-85 13:19")
    (COND
       ((NOT (\TIMER.TIMERP TIMER))                           (* Do the check out here so that an error won't happen
                                                                 underneath the UNINTERRUPTABLY)
        (LISPERROR "ILLEGAL ARG" TIMER))
       ((\TIMER.TIMERP ClockValue.or.timerUnits)              (* Note that in Interlisp-D the TIMER.TIMEREXPIRED? macro
                                                                 will clobber its first arg.)
        (TIMER.TIMEREXPIRED? (TIMER.MAKESAFETIMER ClockValue.or.timerUnits \TIMEREXPIRED.BOX)
               TIMER))
       (T
          (* Distribute thru the SELECTQ this way so that Interlisp-10 compiler can optimize out the boxing.
          Leave the UNINTERRUPTABLY so that Interlisp-D won't interrupt between putting the value in \TIMEREXPIRED.BOX and
          the IGEQ test.)

          (SELECTQ (CANONICAL.TIMERUNITS ClockValue.or.timerUnits)
               ((TICKS)
                    (TIMER.TIMEREXPIRED? (\TIMER.IN.TICKS \TIMEREXPIRED.BOX)
                           TIMER))
               ((MILLISECONDS)
                    (TIMER.TIMEREXPIRED? (\TIMER.IN.MILLISECONDS \TIMEREXPIRED.BOX)
                           TIMER))
               ((SECONDS)
                    (TIMER.TIMEREXPIRED? (\TIMER.IN.SECONDS \TIMEREXPIRED.BOX)
                           TIMER))
              NIL])
```

(TIME.UNTIL

```
  [LAMBDA (TIMER UNITS)                                       (* lmm "12-Apr-85 13:47")
    (COND
       ((NOT (\TIMER.TIMERP TIMER))                           (* Do the check out here so that an error won't happen
                                                                 underneath the UNINTERRUPTABLY)
        (LISPERROR "ILLEGAL ARG" TIMER))
       (T (SELECTQ (CANONICAL.TIMERUNITS UNITS)
              (TICKS (\TIMER.DIFFERENCE TIMER (\TIMER.IN.TICKS \TIMEREXPIRED.BOX)))
              (MILLISECONDS (\TIMER.DIFFERENCE TIMER (\TIMER.IN.MILLISECONDS \TIMEREXPIRED.BOX)))
              (SECONDS (\TIMER.DIFFERENCE TIMER (\TIMER.IN.SECONDS \TIMEREXPIRED.BOX)))
              (SHOULDNT])

)

(RPAQ \TIMEREXPIRED.BOX (SETUPTIMER 0))

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \TIMEREXPIRED.BOX \RCLKMILLISECOND \RCLKSECOND)
)
```

;; Arrange for the proper compiler.

```
(PUTPROPS LLTIMER FILETYPE :FAKE-COMPILE-FILE)
```

FUNCTION INDEX

MACRO INDEX

CONSTANT INDEX

VARIABLE INDEX

PROPERTY INDEX

OPTIMIZER INDEX