*File created:* 12-Nov-85 19:11:43 **{ERIS}<IRIS>KOTO>IRISIO.;2**

*changes to:* (VARS IRISIOCOMS)
(FNS IRIS.SENDFS)

*previous date:* 9-Sep-85 13:47:28 {ERIS}<IRIS>KOTO>IRISIO.;1

*Read Table:* OLD-INTERLISP-FILE

*Package:* INTERLISP

*Format:* XCCS



```
(RPAQQ IRISIOCOMS
       [(COMS (* User level primitives)
              (FNS IRIS.GEXIT IRIS.GFLUSH IRIS.GINIT IRIS.GRESET)
              (MACROS IRIS.GFLUSH))
        (COMS (* Lower level primitives)
              (FNS IRIS.RECBS IRIS.RECFS IRIS.RECLS IRIS.RECSS IRIS.SENDBS IRIS.SENDFS IRIS.SENDLS IRIS.SENDQS
                   IRIS.SENDSS IRIS.SETFASTCOM)
              (MACROS IRIS.DOSYNC IRIS.ECHOFF IRIS.ECHOON IRIS.FLUSHG IRIS.GCMD IRIS.GETGCHAR IRIS.GEXIT
                      IRIS.GFINISH IRIS.PUTGCHAR IRIS.REC32 IRIS.REC6 IRIS.RECB IRIS.RECCR IRIS.RECF IRIS.RECL
                      IRIS.RECO IRIS.RECOS IRIS.RECS IRIS.SEND6 IRIS.SEND8 IRIS.SENDB IRIS.SENDC IRIS.SENDF
                      IRIS.SENDL IRIS.SENDO IRIS.SENDS SPPINPUTSTREAM SPPSTREAM?)
              (CONSTANTS (STDERR T)
                     (IRIS\AESC 46)
                     (IRIS\RESC 126)
                     (IRIS\TESC 16)))
        [DECLARE: EVAL@LOAD DONTCOPY (P (LOADDEF (QUOTE FLOATP)
                                                 (QUOTE RECORD)
                                                 (QUOTE LLARITH]
        (INITVARS (IRISCONN)
                  (IRISSPPON T))
        (DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                              (NLAML)
                                                                              (LAMA])
```

(* * User level primitives)

```
(DEFINEQ

(IRIS.GEXIT
  [LAMBDA (stream)
    (if stream
        then (IRIS.FLUSHG stream)
      else (IRIS.FLUSHG IRISCONN])

(IRIS.GFLUSH
  [LAMBDA (stream)
    (if stream
        then (IRIS.FLUSHG stream)
      else (IRIS.FLUSHG IRISCONN])

(IRIS.GINIT
  [LAMBDA (STREAM)                                    (* LeL, " 3-Sep-85 17:18")
    (if (NOT STREAM)
        then (SETQ STREAM IRISCONN))
    (IRIS.SETFASTCOM STREAM)                          (* Assumes that we communicate on the net)
    (IRIS.XGINIT STREAM)
    (IRIS.FLUSHG STREAM])

(IRIS.GRESET
  [LAMBDA (STREAM)                                    (* LeL, " 3-Sep-85 17:18")
    (IRIS.XGRESET STREAM)
    (IRIS.FLUSHG STREAM])

)

(DECLARE: EVAL@COMPILE

(PUTPROPS IRIS.GFLUSH MACRO [arg?                     (* Just for speed...)
                             (if arg?
                                 then (CONS (QUOTE IRIS.FLUSHG)
                                            arg?)
                               else (QUOTE (IRIS.FLUSHG IRISCONN])
)
```

(* * Lower level primitives)

```
(DEFINEQ
```

## (IRIS.RECBS

```
[LAMBDA (values stream)                                    (* LeL, " 6-Sep-85 14:15")
                                                           (* Receive an array of bytes and fill VALUES)
  (PROG (nLongs nBytes)
        (SETQ nLongs (LRSH (IPLUS (SETQ nBytes (IRIS.RECL stream))
                                  3)
                           2))                             (* Number of longs -
                                                           FIXP -
                                                           to recieve)
        (if (NEQ IRIS\RESC (IRIS.GETGCHAR stream))
            then (PRINT "IRIS.RECBS: error in array transport" STDERR)
                 (while (SPP.READP stream) do (BIN stream))    (* Flush input)
                 (RETURN))
        [for i from 0 to (SUB1 nLongs) as ptr from (ARRAYORIG values) by 3
           bind aLong (arrayMax _ (IPLUS (ARRAYORIG values)
                                         nVals))
           do (SETQ aLong (IRIS.REC32 stream))             (* Recieve 6 six-bits words to make a long)
              (if (IRIS.DOSYNC i)
                  then (IRIS.GETGCHAR stream)
                       (IRIS.PUTGCHAR IRIS\AESC stream)
                       (IRIS.FLUSHG stream))
              (for j from 0 to 2 when (LEQ (IPLUS ptr j)
                                           arrayMax)
                 do (SETA values (IPLUS ptr j)
                           (LOGAND 255 (LRSH aLong (LLSH j 3]
        (IRIS.GETGCHAR stream])
```

## (IRIS.RECFS

```
[LAMBDA (values stream)                                    (* LeL, " 6-Sep-85 12:50")
                                                           (* Common subroutine to IRIS.RECFS and IRIS.RECLS)
  (PROG (nLongs)
        (SETQ nLongs (IRIS.RECL stream))
        (if (NEQ IRIS\RESC (IRIS.GETGCHAR stream))
            then (printout STDERR "IRIS.RECFLS: error in array transport" T)
                 (while (SPP.READP stream) do (BIN stream))    (* Empty the stream buffer)
                 (RETURN))
        (for i from 0 to (SUB1 nLongs) as ptr from (ARRAYORIG values) bind aLong (aFloat _ (NCREATE 'FLOATP))
           do (SETQ aLong (IRIS.REC32 stream))
              (if (IRIS.DOSYNC i)
                  then (IRIS.GETGCHAR stream)
                       (IRIS.PUTGCHAR IRIS\AESC stream)
                       (IRIS.FLUSHG stream))
              (replace (FLOATP HIWORD) of aFloat with (LRSH aLong 16))
              (replace (FLOATP LOWORD) of aFloat with (LOGAND aLong 65535))
              (SETA values ptr aFloat))
        (IRIS.GETGCHAR stream])
```

## (IRIS.RECLS

```
[LAMBDA (values STREAM)                                    (* LeL, " 6-Sep-85 10:22")
                                                           (* Recieve an array of longs)
  (PROG (nLongs)
        (SETQ nLongs (IRIS.RECL STREAM))
        (if (NEQ IRIS\RESC (IRIS.GETGCHAR STREAM))
            then (PRINT "IRIS.RECLS: error in array transport" STDERR)
                 (while (SPP.READP stream) do (BIN stream))
                 (RETURN))
        (for i from 0 to (SUB1 nLongs) as ptr from (ARRAYORIG values) bind aLong
           do (SETQ aLong (IRIS.REC32 STREAM))
              (if (IRIS.DOSYNC i)
                  then (IRIS.GETGCHAR STREAM)
                       (IRIS.PUTGCHAR IRIS\AESC STREAM)
                       (IRIS.FLUSHG STREAM))
              (SETA values ptr aLong))
        (IRIS.GETGCHAR STREAM])
```

## (IRIS.RECSS

```
[LAMBDA (values stream)                                    (* LeL, " 6-Sep-85 14:17")
                                                           (* Recieve an array of SMALL INTEGERS)
  (PROG (nLongs nShorts)
        (SETQ nLongs (LRSH (ADD1 (SETQ nShorts (IRIS.RECL stream)))
                           1))
        (if (NEQ IRIS\RESC (IRIS.GETGCHAR stream))
            then (PRINT "IRIS.RECSS: error in array transport" STDERR)
                 (while (SPP.READP stream) do (BIN stream))
                 (RETURN))
        [for i from 0 to (SUB1 nLongs) as ptr from (ARRAYORIG values) by 2 bind aLong
           do (SETQ aLong (IRIS.REC32 stream))
              (if (IRIS.DOSYNC i)
                  then (IRIS.GETGCHAR stream)
                       (IRIS.PUTGCHAR IRIS\AESC stream)
                       (IRIS.FLUSHG stream))
              (SETA values ptr (LRSH aLong 16))
              (if (OR (LESSP i (SUB1 nLongs))
```

```
                    (EVENP nShorts))
                then (SETA values (ADD1 ptr)
                          (LOGAND 65535 aLong]
        (IRIS.GETGCHAR stream])
```

## (IRIS.SENDBS

```
[LAMBDA (values nVals stream)                                   (* LeL, " 9-Sep-85 05:29")
                                                                (* Send an array of bytes)
  (PROG (nLongs)
        (SETQ nLongs (LRSH (IPLUS nVals 3)
                           2))
        (COND
           ((ARRAYP values)
            (IRIS.SENDL nVals stream)                           (* Fill a 32 bits word starting from highest byte :)
            (for i from 0 to (SUB1 nLongs) as ptr from (ARRAYORIG values) by 4
               bind aLong (arrayMax _ (IPLUS (ARRAYORIG values)
                                             nVals))
               do [SETQ aLong (for j from 0 to 4 when (LEQ (IPLUS ptr j)
                                                           arrayMax)
                                 sum (LLSH (ELT values (IPLUS ptr j))
                                           (LLSH j 3]
                  (if (IRIS.DOSYNC i)
                      then (IRIS.PUTGCHAR IRIS\AESC stream))
                  (IRIS.SENDL aLong stream)))
           ((LISTP values)
            (IRIS.SENDL nVals stream)
            (for i from 0 to (SUB1 nLongs) bind (ptr _ values)
               do (SETQ aLong (for j from 24 to 0 by -8 when ptr sum (LLSH (pop ptr)
                                                                           j)))
                  (if (IRIS.DOSYNC i)
                      then (IRIS.PUTGCHAR IRIS\AESC stream))
                  (IRIS.SENDL aLong stream])
```

## (IRIS.SENDFS

```
[LAMBDA (values nVals stream)                                   (* gbn "11-Nov-85 19:48")

        (* * Sends an array or (possibly two-layered) list of numbers)

  (COND
     ([AND (ARRAYP values)
           (NUMBERP (ELT values (ARRAYORIG values]               (* An array of numbers)
      (IRIS.SENDL (LLSH nVals 2)
             stream)
      (for i from 0 to (SUB1 nVals) as ptr from (ARRAYORIG values) do (if (IRIS.DOSYNC i)
                                                                         then (IRIS.PUTGCHAR IRIS\AESC stream))
                                                                     (IRIS.SENDF (ELT values ptr)
                                                                            stream)))
     ((AND (LISTP values)
           (NUMBERP (CAR values)))                              (* A list of numbers)
      (IRIS.SENDL (LLSH nVals 2)
             stream)
      (for i in values as counter from 0 do (if (IRIS.DOSYNC counter)
                                                then (IRIS.PUTGCHAR IRIS\AESC stream))
                                            (IRIS.SENDF i stream)))
     ((AND (LISTP values)
           (POSITIONP (CAR values))
           (NUMBERP (CAAR values)))                             (* A list of positions)
      (IRIS.SENDL (LLSH nVals 2)
             stream)
      (for i in values bind (counter _ -1) do (if (IRIS.DOSYNC (add counter 1))
                                                  then (IRIS.PUTGCHAR IRIS\AESC stream))
                                              (IRIS.SENDF (CAR i)
                                                     stream)
                                              (if (IRIS.DOSYNC (add counter 1))
                                                  then (IRIS.PUTGCHAR IRIS\AESC stream))
                                              (IRIS.SENDF (CDR i)
                                                     stream)))
     [(AND (LISTP values)
           (LISTP (CAR values))
           (NUMBERP (CAAR values)))                             (* A list of list of numbers)
      (IRIS.SENDL (LLSH nVals 2)
             stream)
      (for i in values bind (counter _ -1) do (for j in i eachtime (add counter 1)
                                                 do (if (IRIS.DOSYNC counter)
                                                        then (IRIS.PUTGCHAR IRIS\AESC stream))
                                                    (IRIS.SENDF j stream]
     (T (ERROR values "-- is not an list [of list]/array of numbers"])
```

## (IRIS.SENDLS

```
[LAMBDA (values nVals stream)                                   (* LeL, " 9-Sep-85 02:14")

        (* * Sends an array or (possibly two-layered) list of numbers)

  (COND
```

```
       ([AND (ARRAYP values)
             (NUMBERP (ELT values (ARRAYORIG values]            (* An array of numbers)
        (IRIS.SENDL (LLSH nVals 2)
              stream)
        (for i from 0 to (SUB1 nVals) as ptr from (ARRAYORIG values) do (if (IRIS.DOSYNC i)
                                                                           then (IRIS.PUTGCHAR IRIS\AESC stream))
                                                                         (IRIS.SENDL (ELT values ptr)
                                                                                stream)))
       ((AND (LISTP values)
             (NUMBERP (CAR values)))                            (* A list of numbers)
        (IRIS.SENDL (LLSH nVals 2)
              stream)
        (for i in values as counter from 0 do (if (IRIS.DOSYNC counter)
                                                  then (IRIS.PUTGCHAR IRIS\AESC stream))
                                              (IRIS.SENDL i stream)))
       [(AND (LISTP values)
             (LISTP (CAR values))
             (NUMBERP (CAAR values)))                           (* A list of list of numbers)
        (IRIS.SENDL (LLSH nVals 2)
              stream)
        (for i in values bind (counter _ -1) do (for j in i eachtime (add counter 1)
                                                   do (if (IRIS.DOSYNC counter)
                                                          then (IRIS.PUTGCHAR IRIS\AESC stream))
                                                      (IRIS.SENDL j stream]
       (T (ERROR values "-- is not an list [of list]/array of numbers"])
```

## (IRIS.SENDQS

```
  [LAMBDA (values nVals stream)                                 (* LeL, " 2-Sep-85 12:47")
    (IRIS.SENDL (LLSH nVals 3))
    (COND
       ((ARRAYP values)
        (for i from 0 to (LLSH nVals 1) by 2 as ptr from 0 by 8
           do (if (IRIS.DOSYNC i)
                  then (IRIS.PUTGCHAR IRIS\AESC stream))
              (IRIS.SENDL (LOGOR (LLSH (ELT values ptr)
                                       16)
                                 (LLSH (ELT values (IPLUS ptr 1))
                                       24)
                                 (LLSH (ELT values (IPLUS ptr 2))
                                       8)
                                 (ELT values (IPLUS ptr 3)))
                          stream)
              (if (IRIS.DOSYNC (IPLUS i 1))
                  then (IRIS.PUTGCHAR IRIS\AESC stream))
              (IRIS.SENDL (LOGOR (LLSH (ELT values (IPLUS ptr 4))
                                       24)
                                 (LLSH (ELT values (IPLUS ptr 5))
                                       16)
                                 (ELT values (IPLUS ptr 6))
                                 (LLSH (ELT values (IPLUS ptr 7))
                                       8))
                          stream)))
       [(LISTP values)
        (for i from 0 to (LLSH nVals 1) by 2 as ptr from values by 8
           do (if (IRIS.DOSYNC i)
                  then (IRIS.PUTGCHAR IRIS\AESC stream))
              (IRIS.SENDL (LOGOR (LLSH (CAR values)
                                       16)
                                 (LLSH (CADR values)
                                       24)
                                 (LLSH (CADDR values)
                                       8)
                                 (CADDDR values))
                          stream)
              (SETQ values (NTH values 5))
              (if (IRIS.DOSYNC (IPLUS i 1))
                  then (IRIS.PUTGCHAR IRIS\AESC stream))
              (IRIS.SENDL (LOGOR (LLSH (ELT values (CAR values))
                                       24)
                                 (LLSH (ELT values (CADR values))
                                       16)
                                 (CADDR values)
                                 (LLSH (CADDDR values)
                                       8))
                          stream)
              (SETQ values (NTH values 5]
       (T (ERROR values "-- neither an array nor a list"])
```

## (IRIS.SENDSS

```
  [LAMBDA (values nVals stream)                                 (* LeL, " 6-Sep-85 14:20")

          (* * Sends an array or list of numbers shorts (SMALLPs))

    (LET ((nLongs (LRSH nVals 1))
          (nBytes (LLSH nVals 1)))
```

```
         (COND
            ([AND (ARRAYP values)
                  (NUMBERP (ELT values (ARRAYORIG values]        (* An array of numbers)
             (IRIS.SENDL nBytes stream)
             (for i from 0 to (SUB1 nLongs) as ptr from (ARRAYORIG values) by 2 bind aLong
                do (SETQ aLong (ELT values ptr))
                   (if (OR (LESSP i nLongs)
                           (EVENP nVals))
                       then (add aLong (LLSH (ELT values (ADD1 ptr))
                                             16)))
                   (if (IRIS.DOSYNC i)
                       then (IRIS.PUTGCHAR IRIS\AESC stream))
                   (IRIS.SENDL aLong stream)))
            ((AND (LISTP values)
                  (NUMBERP (CAR values)))                        (* A list of numbers)
             (IRIS.SENDL nBytes stream)
             (for i from 0 to (SUB1 nLongs) bind aLong (pnt _ values)
                do (SETQ aLong (pop pnt))
                   (if pnt
                       then (add aLong (LLSH (pop pnt)
                                             16)))
                   (if (IRIS.DOSYNC i)
                       then (IRIS.PUTGCHAR IRIS\AESC stream))
                   (IRIS.SENDL i stream)))
            (T (ERROR values "-- is not an list [of list]/array of numbers"])
```

## (IRIS.SETFASTCOM

```
  [LAMBDA (STREAM)                                               (* gbn "19-Mar-85 21:02")
    (IRIS.GCMD 1 STREAM])

)

(DECLARE: EVAL@COMPILE

(PUTPROPS IRIS.DOSYNC MACRO ((i)
                             (COND
                                ((EQ 0 (LOGAND i 7)))
                                (T NIL))))

(PUTPROPS IRIS.ECHOFF MACRO ((STREAM)
                             (STREAMPROP STREAM (QUOTE IRIS\ECHOFLAG)
                                    NIL)))

(PUTPROPS IRIS.ECHOON MACRO ((STREAM)
                             (STREAMPROP STREAM (QUOTE IRIS\ECHOFLAG)
                                    T)))

(PUTPROPS IRIS.FLUSHG MACRO (= . SPP.FORCEOUTPUT))

(PUTPROPS IRIS.GCMD MACRO ((CMD STREAM)                          (* Sends a command)
                           (BOUT STREAM IRIS\TESC)               (* Escape character)
                           (IRIS.SEND6 CMD STREAM)               (* ...followed by the number in two six bits transmission)
                           (IRIS.SEND6 (LRSH CMD 6)
                                  STREAM)))

(PUTPROPS IRIS.GETGCHAR MACRO ((STREAM)
                               (BIN (SPPINPUTSTREAM STREAM))))

(PUTPROPS IRIS.GEXIT MACRO ((stream)
                            (if stream
                                then (IRIS.FLUSHG stream)
                              else (IRIS.FLUSHG IRISCONN))))

(PUTPROPS IRIS.GFINISH MACRO ((stream)                           (* null defn)
                              (IRIS.FLUSHG stream)))

(PUTPROPS IRIS.PUTGCHAR MACRO ((onechar SPPSTREAM)
                               (BOUT SPPSTREAM onechar)))

(PUTPROPS IRIS.REC32 MACRO ((stream)
                            (for j from 0 to 30 by 6 sum (LLSH (IRIS.REC6 stream)
                                                               j))))

(PUTPROPS IRIS.REC6 MACRO ((STREAM)

          (* Recieve a 6 bit word; we substract 32 because the other end add3s 32 to avoid sending control characters)
                                                                 (* NO LONGER ANDS 63)
                          (IDIFFERENCE (IRIS.GETGCHAR STREAM)
                                 32)))

(PUTPROPS IRIS.RECB MACRO [LAMBDA (STREAM)                       (* Receive a byte)
                                                                 (* is passed the spp outputstream, so must grab the input
                                                                 stream from it)
                           (SETQ STREAM (SPPINPUTSTREAM STREAM))
                           (while (NEQ IRIS\RESC (BIN STREAM)))
                           (LOGOR (IRIS.REC6 STREAM)
```

```
                                (LLSH (IRIS.REC6 STREAM)
                                      6])

(PUTPROPS IRIS.RECCR MACRO ((STREAM)                          (* recieve a CarriageReturn)
                            (IRIS.GETGCHAR STREAM)            (* OR (EQ (IRIS.GETGCHAR STREAM)
                                                              (IPLUS 32 (CHARCODE CR))) (ERROR "IRIS.RECCR received
                                                              a non-carriage return from the IRIS"))
                            ))

(PUTPROPS IRIS.RECF MACRO [LAMBDA (SPPSTREAM)                 (* gbn "17-Jun-85 17:31")

          (* receive a float. uses IRIS.RECL to receive a 32 bit word and convert it to float)

                           (PROG (AFLOAT ALONG)
                                 (SETQ ALONG (IRIS.RECL SPPSTREAM))
                                 (SETQ AFLOAT (NCREATE (QUOTE FLOATP)))
                                 (replace (FLOATP HIWORD) of AFLOAT with (LRSH ALONG 16))
                                 (replace (FLOATP LOWORD) of AFLOAT with (LOGAND ALONG 65535))
                                 (RETURN AFLOAT])

(PUTPROPS IRIS.RECL MACRO ((stream)
                           (while (NEQ IRIS\RESC (IRIS.GETGCHAR stream)) do NIL)
                           (IRIS.REC32 stream)))

(PUTPROPS IRIS.RECO MACRO ((STREAM)                           (* Recieve a boolean)
                           (IRIS.RECB STREAM)))

(PUTPROPS IRIS.RECOS MACRO ((values STREAM)                   (* Recieve an array of boolean)
                            (IRIS.RECBS values STREAM)))

(PUTPROPS IRIS.RECS MACRO [(stream)                           (* Recieve a SMALL INTEGER)
                           (while (NEQ (IRIS.GETGCHAR stream)
                                       IRIS\RESC)
                              do NIL)
                           (LET* ((1stbyte (IRIS.REC6 stream))
                                  (2ndbyte (IRIS.REC6 stream)))
                                 (LOGOR 1stbyte (LLSH 2ndbyte 6)
                                        (LLSH (IRIS.REC6 stream)
                                              12])

(PUTPROPS IRIS.SEND6 MACRO [(n STREAM)                        (* Add 32 to avoid sending control characters)
                            (BOUT STREAM (IPLUS 32 (LOGAND 63 n])

(PUTPROPS IRIS.SEND8 MACRO ((n STREAM)
                            (BOUT STREAM n)))

(PUTPROPS IRIS.SENDB MACRO ((VALUE STREAM)                    (* Send a byte)
                            (IRIS.SEND8 VALUE STREAM)))

(PUTPROPS IRIS.SENDC MACRO ((string stream)                   (* Send a string of characters)
                                                              (* should probably allocate a global resource)
                            (IRIS.SENDBS (NCONC1 (CHCON string)
                                                 0)
                                   (ADD1 (NCHARS string))
                                   stream)))

(PUTPROPS IRIS.SENDF MACRO ((value stream)                    (* Send a float)
                            (LET ((float (FLOAT value)))
                                 (IRIS.SEND8 (\GETBASEBYTE float 0)
                                        stream)
                                 (IRIS.SEND8 (\GETBASEBYTE float 1)
                                        stream)
                                 (IRIS.SEND8 (\GETBASEBYTE float 2)
                                        stream)
                                 (IRIS.SEND8 (\GETBASEBYTE float 3)
                                        stream))))

(PUTPROPS IRIS.SENDL MACRO [LAMBDA (VALUE STREAM)             (* Sends a 32 bit integer)
                             (SELECTQ (TYPENAME VALUE)
                                  (SMALLP (if (ILESSP VALUE 0)
                                              then (IRIS.SEND8 255 STREAM)
                                                   (IRIS.SEND8 255 STREAM)
                                            else (IRIS.SEND8 0 STREAM)
                                                 (IRIS.SEND8 0 STREAM))
                                          (IRIS.SEND8 (LOGAND (LRSH VALUE 8)
                                                              255)
                                                 STREAM)
                                          (IRIS.SEND8 (LOGAND VALUE 255)
                                                 STREAM))
                                  (FIXP (IRIS.SEND8 (\GETBASEBYTE VALUE 0)
                                                 STREAM)
                                        (IRIS.SEND8 (\GETBASEBYTE VALUE 1)
                                                 STREAM)
                                        (IRIS.SEND8 (\GETBASEBYTE VALUE 2)
                                                 STREAM)
                                        (IRIS.SEND8 (\GETBASEBYTE VALUE 3)
                                                 STREAM))
```

```
                                (ERROR VALUE "can't be sent thru IRIS.SENDL (neither an FIXP nor a SMALLP)"])

(PUTPROPS IRIS.SENDO MACRO ((value STREAM)                    (* send a boolean)
                            (IRIS.SENDB value STREAM)))

(PUTPROPS IRIS.SENDS MACRO ((value STREAM)                    (* Send a SMALL INTEGER (16 bits))
                            (IRIS.SEND8 (LOGAND 255 (LRSH value 8))
                                  STREAM)
                            (IRIS.SEND8 (LOGAND 255 value)
                                  STREAM)))

(PUTPROPS SPPINPUTSTREAM MACRO ((OUTPUTSTREAM)                (* gbn "17-Jun-85 17:40")
                                (fetch (SPPCON SPPINPUTSTREAM) of (fetch (STREAM F1) of OUTPUTSTREAM))))

(PUTPROPS SPPSTREAM? MACRO [LAMBDA (STREAM)
                             (AND (TYPENAME STREAM (QUOTE STREAM))
                                  (TYPENAMEP (fetch F1 of STREAM)
                                         (QUOTE SPPCON])
)

(DECLARE: EVAL@COMPILE

(RPAQQ STDERR T)

(RPAQQ IRIS\AESC 46)

(RPAQQ IRIS\RESC 126)

(RPAQQ IRIS\TESC 16)

(CONSTANTS (STDERR T)
       (IRIS\AESC 46)
       (IRIS\RESC 126)
       (IRIS\TESC 16))
)

(DECLARE: EVAL@LOAD DONTCOPY

(LOADDEF (QUOTE FLOATP)
       (QUOTE RECORD)
       (QUOTE LLARITH))
)

(RPAQ? IRISCONN )

(RPAQ? IRISSPPON T)

(DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA )
)

(PUTPROPS IRISIO COPYRIGHT ("Xerox Corporation" 1985))
```

## FUNCTION INDEX



## MACRO INDEX



## CONSTANT INDEX



## VARIABLE INDEX