```
  File created:    23-Aug-90 17:10:14   {DSK}<LISPFILES>TCP>TCPDEBUG.;2

   changes to:    (VARS TCPDEBUGCOMS)

previous date:    15-Feb-89 13:41:39   {DSK}<LISPFILES>TCP>TCPDEBUG.;1

  Read Table:    INTERLISP

     Package:    INTERLISP

        Format:    XCCS
```

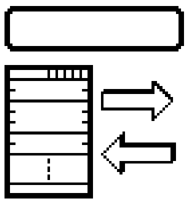
```
(RPAQQ TCPDEBUGCOMS
       ((COMS ;; standard TCP small servers

              (FNS TCP.SINK.SERVER TCP.TELNET.SERVER \TCP.SINK.PROCESS TCP.ECHO.SERVER \TCP.ECHO.PROCESS))
        (COMS ;; TCP tracing and debugging info

              (DECLARE%: EVAL@COMPILE DONTCOPY (FILES (LOADCOMP)
                                                      TCP)
                     (CONSTANTS LIGHTGRAYSHADE))
              (GLOBALVARS TCPTRACEFLG TCPTRACEFILE TCPTRACEMENU \TCP.ELAPSED.TIME \TCP.DEBUGGABLE)
              (INITVARS TCPTRACEFLG TCPTRACEFILE TCPTRACEMENU \TCP.ELAPSED.TIME NETTRACETITLEREG)
              (VARS (\TCP.DEBUGGABLE T))
              (BITMAPS NETTRACEICON NETTRACEMASK)
              (FILES (SYSLOAD)
                     TCP)
              (FNS TCP.PRINT.SEGMENT \TCP.PRINT.OPTIONS \TCP.PRINT.ELAPSED.TIME \TCP.PRINT.SEGMENT.QUEUE
                   TCPTRACE \TCPTRACEMENU.ITEMFN \TCPTRACEMENU.DISPLAYFN TCP.DRIBBLE))
        (COMS ;; miscellaneous TCP debugging

              (GLOBALVARS \TCP.LOSSAGE \TCP.LOOPBACK.QUEUE \TCP.LOOPBACK.EVENT \TCP.MASTER.SOCKET)
              (INITVARS \TCP.LOSSAGE \TCP.LOOPBACK.QUEUE \TCP.LOOPBACK.EVENT)
              (FNS TCP.DEBUG TCP.WATCHER DUMMY\IP\Transmit\Packet \TCP.CHECK.INPUT.QUEUE TCP.FAUCET TCP.ECHOTEST
                   TCP.QUIET.ECHOTEST TCP.SINKTEST GENERATE.RANDOM.CHARS COPYBYTESTREAM TCP.COPYTOWINDOW
                   TEST.CHECKSUM))))
```

;; standard TCP small servers

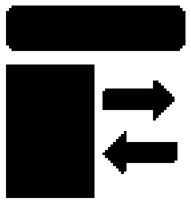
```
(DEFINEQ

(TCP.SINK.SERVER
  [LAMBDA (PORT)                                                  (* ecc "14-May-84 16:32")
    (bind STREAM do (if (SETQ STREAM (TCP.OPEN NIL NIL (OR PORT \TCP.SINK.PORT)
                                            'PASSIVE
                                            'INPUT T))
                        then (ADD.PROCESS `(\TCP.SINK.PROCESS %, STREAM)
                                    'NAME "TCP Sink"])

(TCP.TELNET.SERVER
  [LAMBDA NIL                                                     (* ejs%: "20-Jun-85 12:38")
    (LET ((INSTREAM (TCP.OPEN NIL NIL \TCP.TELNET.PORT 'PASSIVE 'INPUT))
          OUTSTREAM)
         (COND
            (INSTREAM (SETQ OUTSTREAM (TCP.OTHER.STREAM INSTREAM))
                   (ADD.PROCESS (LIST '\TCP.ECHO.PROCESS (KWOTE INSTREAM)
                                      (KWOTE OUTSTREAM))
                          'NAME "Telnet echo")
                   (ADD.PROCESS '(TCP.TELNET.SERVER))
                   (GENERATE.RANDOM.CHARS OUTSTREAM])

(\TCP.SINK.PROCESS
  [LAMBDA (STREAM)                                                (* ejs%: " 7-Jun-85 13:11")
    (RESETSAVE NIL (LIST (FUNCTION CLOSEF)
                         STREAM))
    (replace (STREAM ENDOFSTREAMOP) of STREAM with (FUNCTION NILL))
    (until (EOFP STREAM) do (BIN STREAM])

(TCP.ECHO.SERVER
  [LAMBDA (PORT)                                                  (* ecc "14-May-84 16:35")
    (bind STREAM do (if (SETQ STREAM (TCP.OPEN NIL NIL (OR PORT \TCP.ECHO.PORT)
                                            'PASSIVE
                                            'INPUT T))
                        then (ADD.PROCESS `(\TCP.ECHO.PROCESS %, STREAM %, (TCP.OTHER.STREAM STREAM))
                                    'NAME "TCP Echo"])

(\TCP.ECHO.PROCESS
  [LAMBDA (INSTR OUTSTR)                                          (* ejs%: "25-Mar-86 18:07")
    (RESETSAVE NIL (LIST (FUNCTION CLOSEF)
                         INSTR))
    (RESETSAVE NIL (LIST (FUNCTION CLOSEF)
                         OUTSTR))
    (bind C until (OR (NOT (OPENP INSTR 'INPUT))
                      (EOFP INSTR))
       do [COND
             [(CAR (NLSETQ (READP INSTR)))
              (SETQ C (CAR (NLSETQ (BIN INSTR]
             (T (FORCEOUTPUT OUTSTR)
                (SETQ C (CAR (NLSETQ (BIN INSTR]
          [COND
             (C (NLSETQ (BOUT OUTSTR C]
          (if (OR (NOT (NLSETQ (READP INSTR)))
                  (NOT (OPENP INSTR 'INPUT))
                  (EOFP INSTR))
              then (NLSETQ (FORCEOUTPUT OUTSTR])

)
```

;; TCP tracing and debugging info

```
(DECLARE%: EVAL@COMPILE DONTCOPY

(FILESLOAD (LOADCOMP)
       TCP)

(DECLARE%: EVAL@COMPILE

(RPAQQ LIGHTGRAYSHADE 1025)

(CONSTANTS LIGHTGRAYSHADE)
)
)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS TCPTRACEFLG TCPTRACEFILE TCPTRACEMENU \TCP.ELAPSED.TIME \TCP.DEBUGGABLE)
)

(RPAQ? TCPTRACEFLG NIL)

(RPAQ? TCPTRACEFILE NIL)

(RPAQ? TCPTRACEMENU NIL)

(RPAQ? \TCP.ELAPSED.TIME NIL)

(RPAQ? NETTRACETITLEREG NIL)

(RPAQQ \TCP.DEBUGGABLE T)

(RPAQQ NETTRACEICON
```



```
                    )

(RPAQQ NETTRACEMASK
```



```
                    )

(FILESLOAD (SYSLOAD)
       TCP)

(DEFINEQ

(TCP.PRINT.SEGMENT
  [LAMBDA (SEGMENT FILE NOFROMTOFLG DATAFLG)                    (* ejs%: "20-Jun-85 16:06")
    (PROG ((SEPR "")
           (COMMA ",")
           (SEQ (fetch TCP.SEQ of SEGMENT))
           (LEN (\TCP.DATA.LENGTH SEGMENT))
           (FLAGS (fetch TCP.CTRL of SEGMENT))
           TOP BASE)
          (if (NOT NOFROMTOFLG)
              then (printout FILE "from " %# (\IP.PRINT.ADDRESS (fetch TCP.SRC.ADDR of SEGMENT)
                                                            FILE)
                                "."
                                (fetch TCP.SRC.PORT of SEGMENT)
                                " to " %# (\IP.PRINT.ADDRESS (fetch TCP.DST.ADDR of SEGMENT)
                                                   FILE)
                                ":"
                                (fetch TCP.DST.PORT of SEGMENT)
                                T))
              (printout FILE SEQ)
              [SETQ TOP (SUB1 (IPLUS SEQ LEN (\TCP.SYN.OR.FIN FLAGS T]
              (if (\32BIT.LT SEQ TOP)
                  then (printout FILE ".." TOP))
              (printout FILE "/" (fetch TCP.ACK of SEGMENT)
                     " [")
              (if (BITTEST FLAGS \TCP.CTRL.URG)
                  then (printout FILE SEPR "URG")
                       (SETQ SEPR COMMA))
              (if (BITTEST FLAGS \TCP.CTRL.ACK)
                  then (printout FILE SEPR "ACK")
                       (SETQ SEPR COMMA))
              (if (BITTEST FLAGS \TCP.CTRL.PSH)
                  then (printout FILE SEPR "PSH")
                       (SETQ SEPR COMMA))
              (if (BITTEST FLAGS \TCP.CTRL.RST)
                  then (printout FILE SEPR "RST")
                       (SETQ SEPR COMMA))
              (if (BITTEST FLAGS \TCP.CTRL.SYN)
                  then (printout FILE SEPR "SYN")
                       (SETQ SEPR COMMA))
              (if (BITTEST FLAGS \TCP.CTRL.FIN)
                  then (printout FILE SEPR "FIN")
                       (SETQ SEPR COMMA))
              (printout FILE "] window = " (fetch TCP.WINDOW of SEGMENT)
                     " checksum = "
                     (fetch TCP.CHECKSUM of SEGMENT)
                     " length = " LEN T)
              (if (IGREATERP (fetch TCP.DATA.OFFSET of SEGMENT)
                        \TCP.MIN.DATA.OFFSET)
                  then (\TCP.PRINT.OPTIONS SEGMENT FILE))
              (if (AND DATAFLG (NOT (ZEROP LEN)))
                  then (printout FILE "Contents:")
                       (SETQ BASE (fetch TCP.CONTENTS of SEGMENT))
                       (for (I _ 0) to (SUB1 LEN) do (PRIN1 (CHARACTER (\GETBASEBYTE BASE I))
                                                             FILE))
                       (TERPRI FILE])
```

(\TCP.PRINT.OPTIONS

```
  [LAMBDA (SEGMENT FILE)                                            (* ejs%: "20-Jun-85 13:22")

              (* * Process the options in a TCP header)

    (printout FILE "Options: ")
    (bind (OPTIONBASE _ (fetch (TCPSEGMENT TCP.OPTIONS) of SEGMENT))
          (OPTIONOFFSET _ 0)
          OPTION eachtime (SETQ OPTION (\GETBASEBYTE OPTIONBASE OPTIONOFFSET)) until (EQ OPTION \TCPOPT.END)
       do (SELECTC OPTION
                 (\TCPOPT.END (printout FILE "end")
                              (add OPTIONOFFSET 1))
                 (\TCPOPT.NOP (printout FILE "nop")
                              (add OPTIONOFFSET 1))
                 (\TCPOPT.MAXSEG
                      [printout FILE "maxseg: " (LOGOR (LLSH (\GETBASEBYTE OPTIONBASE (IPLUS OPTIONOFFSET 2))
                                                               BITSPERBYTE)
                                                        (\GETBASEBYTE OPTIONBASE (IPLUS OPTIONOFFSET 3]
                      (add OPTIONOFFSET (\GETBASEBYTE OPTIONBASE (ADD1 OPTIONOFFSET))))
                 (RETURN))
              (printout FILE " "])
```

(\TCP.PRINT.ELAPSED.TIME

```
  [LAMBDA (FILE)                                                    (* ecc "23-Apr-84 12:32")
    (if (MEMB 'TIME TCPTRACEFLG)
        then (PROG ((NOW (SETUPTIMER 0 NIL 'MILLISECONDS))
                    INTERVAL)
                   (SETQ INTERVAL (IDIFFERENCE NOW (OR \TCP.ELAPSED.TIME NOW)))
                   (SETQ \TCP.ELAPSED.TIME NOW)
                   (printout FILE (IQUOTIENT INTERVAL 1000)
                          "." .I3..T (IMOD INTERVAL 1000)
                          " "])
```

(\TCP.PRINT.SEGMENT.QUEUE

```
  [LAMBDA (CALLER QUEUE FILE)                                       (* ecc "18-Apr-84 14:38")
    (PROG ((SEGMENT (fetch SYSQUEUEHEAD of QUEUE)))
          (printout FILE .TAB0 0 CALLER ":" T)
          (while SEGMENT do (TCP.PRINT.SEGMENT SEGMENT FILE T)
                              (SETQ SEGMENT (fetch QLINK of SEGMENT])
```

(TCPTRACE

```
  [LAMBDA NIL                                                          ; Edited 15-Apr-87 15:22 by jrb:
    (PROG (MW)
          (if (WINDOWP TCPTRACEFILE)
              then (TOTOPW TCPTRACEFILE)
                   (RETURN))
          (SETQ TCPTRACEFILE (CREATEW))
          [WINDOWADDPROP TCPTRACEFILE 'CLOSEFN (FUNCTION (LAMBDA (WINDOW)
                                                           (if (EQ WINDOW TCPTRACEFILE)
                                                               then (SETQ TCPTRACEFLG NIL)
                                                                    (SETQ TCPTRACEFILE T]
          (DSPFONT (FONTCREATE 'GACHA 8)
                 TCPTRACEFILE)
          (DSPSCROLL T TCPTRACEFILE)
          [if (NOT (type? MENU TCPTRACEMENU))
              then (SETQ TCPTRACEMENU (create MENU
                                              TITLE _ "TCP Trace Window"
                                              ITEMS _ '(("Incoming" RECV "Trace incoming segments")
                                                        ("Time" TIME "Print elapsed time between events")
                                                        ("Transitions" TRANSITION "Trace connection state
                                                               transitions")
                                                        ("Outgoing" SEND "Trace outgoing segments")
                                                        ("Contents" CONTENTS "Print contents of segments when
                                                               tracing")
                                                        ("Checksums" CHECKSUM "Trace segments with bad checksums")
                                                        )
                                              MENUROWS _ 2
                                              CENTERFLG _ T
                                              WHENSELECTEDFN _ (FUNCTION \TCPTRACEMENU.ITEMFN)))
            else (FOR ITEM IN (FETCH (MENU ITEMS) OF TCPTRACEMENU)
                     DO (IF (MEMB (CADR ITEM)
                                  TCPTRACEFLG)
                            THEN (SHADEITEM ITEM TCPTRACEMENU LIGHTGRAYSHADE)
                          ELSE (SHADEITEM ITEM TCPTRACEMENU WHITESHADE]
          (ATTACHMENU TCPTRACEMENU TCPTRACEFILE 'TOP)
          [SETQ MW (CAR (WINDOWPROP TCPTRACEFILE 'ATTACHEDWINDOWS]
          (WINDOWADDPROP MW 'REPAINTFN (FUNCTION \TCPTRACEMENU.DISPLAYFN))
          (WINDOWADDPROP MW 'RESHAPEFN (FUNCTION \TCPTRACEMENU.DISPLAYFN])
```

(\TCPTRACEMENU.ITEMFN

```
  [LAMBDA (ITEM MENU MOUSEKEY)                                         (* ecc "23-Apr-84 13:37")
    (PROG (FLG)
          (if (NULL ITEM)
              then (RETURN))
          (SETQ FLG (CADR ITEM))
          (if (MEMB FLG TCPTRACEFLG)
              then (SHADEITEM ITEM MENU WHITESHADE)
                   (SETQ TCPTRACEFLG (DREMOVE FLG TCPTRACEFLG))
            else (SHADEITEM ITEM MENU LIGHTGRAYSHADE)
                 (SETQ TCPTRACEFLG (CONS FLG TCPTRACEFLG])
```

(\TCPTRACEMENU.DISPLAYFN

```
  [LAMBDA (WINDOW)                                                     (* ecc "23-Apr-84 13:49")
    (PROG [(MENU (CAR (WINDOWPROP WINDOW 'MENU]
          (for ITEM in (fetch ITEMS of MENU) when (MEMB (CADR ITEM)
                                                        TCPTRACEFLG)
             do (SHADEITEM ITEM MENU LIGHTGRAYSHADE])
```

(TCP.DRIBBLE

```
  [LAMBDA (FORM FILE)                                                  (* ecc "18-Apr-84 14:39")
    (if (NULL FILE)
        then (SETQ FILE '{DSK}TCP.Transcript))
    (RESETLST
        (RESETSAVE TCPTRACEFILE (OPENFILE FILE 'OUTPUT))
        (RESETSAVE TCPTRACEFLG T)
        (RESETSAVE NIL (LIST (FUNCTION CLOSEF?)
                             TCPTRACEFILE))
        (PRINT FORM TCPTRACEFILE)
        (TERPRI TCPTRACEFILE)
        (EVAL FORM))])
)
```

;; miscellaneous TCP debugging

```
(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \TCP.LOSSAGE \TCP.LOOPBACK.QUEUE \TCP.LOOPBACK.EVENT \TCP.MASTER.SOCKET)
)

(RPAQ? \TCP.LOSSAGE NIL)
```

```
(RPAQ? \TCP.LOOPBACK.QUEUE NIL)

(RPAQ? \TCP.LOOPBACK.EVENT NIL)

(DEFINEQ

(TCP.DEBUG
  [LAMBDA (ON?)                                                   (* edited%: "21-May-84 13:56")
    (if ON?
        then (TCP.STOP)
             (if (NOT (DEFINEDP 'REAL\IP\Transmit\Packet))
                 then (MOVD 'IP\Transmit\Packet 'REAL\IP\Transmit\Packet))
             (MOVD 'DUMMY\IP\Transmit\Packet 'IP\Transmit\Packet)
             (if (NULL \TCP.LOOPBACK.EVENT)
                 then (SETQ \TCP.LOOPBACK.EVENT (CREATE.EVENT)))
             (if (NULL \TCP.LOOPBACK.QUEUE)
                 then (SETQ \TCP.LOOPBACK.QUEUE (create SYSQUEUE)))
             [if (NOT (FIND.PROCESS 'TCP.WATCHER))
                 then (ADD.PROCESS '(TCP.WATCHER]
      else (if (DEFINEDP 'REAL\IP\Transmit\Packet)
               then (MOVD 'REAL\IP\Transmit\Packet 'IP\Transmit\Packet))
           (DEL.PROCESS 'TCP.WATCHER)
           (\TCP.INIT])

(TCP.WATCHER
  [LAMBDA NIL                                                     (* ecc " 3-May-84 11:10")
                                                                  (* process to handle software loopback of segments)
    (RESETSAVE NIL (LIST (FUNCTION \FLUSH.PACKET.QUEUE)
                         \TCP.LOOPBACK.QUEUE))
    (bind SEGMENT do (SETQ SEGMENT (\DEQUEUE \TCP.LOOPBACK.QUEUE))
                     (if SEGMENT
                         then (\TCP.PACKET.FILTER SEGMENT \TCP.PROTOCOL)
                       else (AWAIT.EVENT \TCP.LOOPBACK.EVENT])

(DUMMY\IP\Transmit\Packet
  [LAMBDA (EPKT)                                                  (* ejs%: " 5-Jan-85 16:57")
                                                                  (* Software loopback.)
    (PROG ([OK (NOT (AND \TCP.LOSSAGE (EQ (RAND 1 \TCP.LOSSAGE)
                                          1]
           SEGMENT)
          (CHECK (OR (NULL (fetch QLINK of EPKT))
                     (SHOULDNT "transmitting queued segment")))
          (if OK
              then (SETQ SEGMENT (\ALLOCATE.ETHERPACKET))
                   (\BLT (\IPDATABASE SEGMENT)
                         (\IPDATABASE EPKT)
                         (FOLDHI (ADD1 (fetch (IP IPTOTALLENGTH) of EPKT))
                                BYTESPERWORD)))
          (if (EQ (fetch EPREQUEUE of EPKT)
                  'FREE)
              then (\RELEASE.ETHERPACKET EPKT)
            elseif (type? SYSQUEUE (fetch EPREQUEUE of EPKT))
              then (\ENQUEUE (fetch EPREQUEUE of EPKT)
                          EPKT))
          (if OK
              then (\ENQUEUE \TCP.LOOPBACK.QUEUE SEGMENT)
                   (NOTIFY.EVENT \TCP.LOOPBACK.EVENT])

(\TCP.CHECK.INPUT.QUEUE
  [LAMBDA (TCB)                                                   (* edited%: "22-May-84 15:32")
                                                                  (* perform consistency check on the input queue)
    (PROG ((QUEUE (fetch TCB.INPUT.QUEUE of TCB))
           CURSEG SEQ1 TOP1 NEXTSEG SEQ2 TOP2)
          (SETQ CURSEG (fetch SYSQUEUEHEAD of QUEUE))
      LOOP
          (if (NULL CURSEG)
              then (RETURN T))
          (SETQ SEQ1 (fetch TCP.SEQ of CURSEG))
          (SETQ TOP1 (IPLUS SEQ1 (fetch TCP.DATA.LENGTH of CURSEG)))
          (if (AND (\32BIT.LEQ SEQ1 (fetch TCB.RCV.NXT of TCB))
                   (\32BIT.GT TOP1 (fetch TCB.RCV.NXT of TCB)))
              then (SHOULDNT "incorrect RCV.NXT")
                   (RETURN NIL))
          (SETQ NEXTSEG (fetch QLINK of CURSEG))
          (if (NULL NEXTSEG)
              then (RETURN T))
          (SETQ SEQ2 (fetch TCP.SEQ of NEXTSEG))
          (SETQ TOP2 (IPLUS SEQ2 (fetch TCP.DATA.LENGTH of NEXTSEG)))
          (if (\32BIT.LT SEQ2 SEQ1)
              then (SHOULDNT "input queue out of order")
                   (RETURN NIL))
          (SETQ CURSEG NEXTSEG)
          (GO LOOP])
```

```
(TCP.FAUCET
  [LAMBDA (HOST PORT NLINES)                                     (* ejs%: "20-Jun-85 12:20")
    (PROG [(STREAM (if HOST
                       then (TCP.OPEN HOST (OR PORT \TCP.SINK.PORT)
                                      NIL
                                      'ACTIVE
                                      'OUTPUT)
                     else (TCP.OPEN NIL NIL (OR PORT \TCP.SINK.PORT)
                                    'PASSIVE
                                    'OUTPUT]
          (RESETLST
              (RESETSAVE NIL (LIST (FUNCTION CLOSEF)
                                   STREAM))
              (GENERATE.RANDOM.CHARS STREAM NLINES))])

(TCP.ECHOTEST
  [LAMBDA (HOST NLINES)                                          (* ecc "14-May-84 17:07")
    (PROG [(STREAM (TCP.OPEN HOST \TCP.ECHO.PORT NIL 'ACTIVE 'OUTPUT]
          (ADD.PROCESS (BQUOTE (TCP.COPYTOWINDOW %, (TCP.OTHER.STREAM STREAM))
                             'NAME "TCP Echo Tester"))
          (GENERATE.RANDOM.CHARS STREAM NLINES)
          (TCP.CLOSE.SENDER STREAM])

(TCP.QUIET.ECHOTEST
  [LAMBDA (HOST NLINES)                                          (* ecc "25-May-84 13:24")
    (PROG [(STREAM (TCP.OPEN HOST \TCP.ECHO.PORT NIL 'ACTIVE 'OUTPUT]
          (ADD.PROCESS (BQUOTE (\TCP.SINK.PROCESS %, (TCP.OTHER.STREAM STREAM))
                             'NAME "TCP Echo Tester"))
          (GENERATE.RANDOM.CHARS STREAM NLINES)
          (TCP.CLOSE.SENDER STREAM])

(TCP.SINKTEST
  [LAMBDA (PORT VISIBLEFLG)                                      (* ecc "14-May-84 17:28")
    (TCP.COPYTOWINDOW (TCP.OPEN NIL NIL (OR PORT \TCP.SINK.PORT)
                                'PASSIVE
                                'INPUT)
           VISIBLEFLG])

(GENERATE.RANDOM.CHARS
  [LAMBDA (STREAM NLINES)                                        (* ejs%: " 7-Jun-85 12:34")
    (bind (N _ 0) while (NEQ N NLINES) do (add N 1)
                                          (printout STREAM "This is byte number " (GETFILEPTR STREAM)
                                                 "." T)
                                          (BLOCK])

(COPYBYTESTREAM
  [LAMBDA (INSTR OUTSTR VISIBLEFLG)                              (* ejs%: " 7-Jun-85 13:44")
    (if VISIBLEFLG
        then (bind (N _ 1)
                   (C _ NIL) while (OPENP INSTR 'INPUT) do (SETQ C (BIN INSTR))
                                                           (printout OUTSTR N ": " C)
                                                           (if (AND (ILEQ C 127)
                                                                    (IGEQ C 32))
                                                               then (printout OUTSTR " (" %# (BOUT OUTSTR C)
                                                                           ")"))
                                                           (TERPRI OUTSTR)
                                                           (add N 1))
      else (bind C while (AND (OPENP INSTR 'INPUT)
                              (NOT (EOFP INSTR)))
              do (COND
                    ((SETQ C (BIN INSTR))
                     (BOUT OUTSTR C])

(TCP.COPYTOWINDOW
  [LAMBDA (STREAM VISIBLEFLG)                                    (* ejs%: "13-Apr-85 16:01")
    (RESETLST
        (RESETSAVE NIL (LIST (FUNCTION CLOSEF?)
                             STREAM))
        (PROG ((WIN (CREATEW NIL "Stream Output")))
              (DSPSCROLL T WIN)
              (COPYBYTESTREAM STREAM WIN VISIBLEFLG)
              (printout WIN .TAB0 0 "[End of stream]"))))])

(TEST.CHECKSUM
  [LAMBDA (STR STR2)                                             (* ecc "24-Apr-84 13:11")
    (if (NULL STR2)
        then (\COMPUTE.CHECKSUM (\ADDBASE (fetch (STRINGP BASE) of STR)
                                         (fetch (STRINGP OFFST) of STR))
```

```
                        (fetch (STRINGP LENGTH) of STR))
       else (\16BIT.COMPLEMENT (\16BIT.1C.PLUS (\COMPUTE.CHECKSUM (\ADDBASE (fetch (STRINGP BASE) of STR)
                                                                                       (fetch (STRINGP OFFST) of STR))
                                                               (fetch (STRINGP LENGTH) of STR)
                                                               T)
                                            (\COMPUTE.CHECKSUM (\ADDBASE (fetch (STRINGP BASE) of STR2)
                                                                          (fetch (STRINGP OFFST) of STR2))
                                                    (fetch (STRINGP LENGTH) of STR2)
                                                    T])

)

(PUTPROPS TCPDEBUG COPYRIGHT ("Xerox Corporation" 1985 1986 1987 1989 1990))
```

FUNCTION INDEX

VARIABLE INDEX

CONSTANT INDEX