File created: 21-Oct-88 12:30:07 **{QV}<BRIGGS>LISP>RS232CNETWORK.;49**

changes to: (FNS \\RS232C.EVENTFN)
(VARS RS232CNETWORKCOMS)

previous date: 13-Oct-88 18:05:26 {QV}<BRIGGS>LISP>RS232CNETWORK.;48

Read Table: XCL

Package: INTERLISP

Format: XCCS


```
(RPAQQ RS232CNETWORKCOMS
       ((COMS (DECLARE\: FIRST DONTEVAL@LOAD DOCOPY (FILES (SYSLOAD)
                                                           DLRS232C))
              (DECLARE\: FIRST EVAL@LOAD DONTCOPY (FILES (LOADCOMP)
                                                         LLETHER 10MBDRIVER LLNS PUP (SOURCE)
                                                         DLRS232C)))
        (COMS ;; Addition to DOVERS232C Opie definitions
              (DECLARE\: DONTCOPY (EXPORT (RECORDS |Dove.i8274.WR7|)
                                          (CONSTANTS * |Dove.i8274.WR7.Constants|))))
        ;; This should have been in the DLRS232C file
        (GLOBALVARS \\DLRS232C.OUTPUT.TIMEOUT)
        ;; a hint that the code should install itself as the network handler...
        (INITVARS (*RS232C-NETWORK* T)
               (*RS232C-NETWORK-DIALING-TIMEOUT* 30)
               (*RS232C-NETWORK-AUTODIAL*))
        (GLOBALVARS *RS232C-NETWORK* *RS232C-NETWORK-DIALING-TIMEOUT* *RS232C-NETWORK-AUTODIAL*)
        (FNS \\DVRS232C.SET.PARAMETERS \\DVRS232C.INIT \\DLRS232C.INIT RS232C.INIT \\RS232C.HANDLE.PACKET
             \\RS232CNETWORKINIT \\DLRS232C.CREATE.NDB)
        ;; because it needed to know to reinit the rs232c if that was the network interface...
        (FNS \\ETHEREVENTFN)
        ;; because it was closing too many sockets when it shouldn't have...
        (FNS TURN.OFF.ETHER)
        ;; because it didn't check to see if it had an NDBIQ or NDBTQ before dequeuing...
        (FNS \\DVRS232C.SHUTDOWN)
        ;; because etherpackets are actually 2 pages, and we want to use it all if necessary (bytesperpage - etherheader + 8bytes from encapsulation
        ;; which we put data into)
        (VARS (\\DLRS232C.DEFAULT.PACKET.LENGTH 968)
              (\\RS232C.OUTPUT.PACKET.LENGTH 968))
        ;; because translate.3to10 isn't the right thing on a phone line... in actual fact, the things that call translate.3to10 should know better than to do
        ;; so on a phone line, but since we're masquerading as a 10Mb/s net due to some other bogosity this is the easiest place to fix it.
        (FNS \\TRANSLATE.3TO10)
        (VARS (\\RS232CNETWORK.NSHOSTNUMBER \\MY.NSHOSTNUMBER))
        (GLOBALVARS \\RS232CNETWORK.NSHOSTNUMBER \\DLRS232C.LOCAL.NDB)
        ;; because this one has some bugs fixed, and it must not reinitialize a running RS232 driver.
        (FNS \\RS232C.EVENTFN)
        ;; because it has to use the \LOCALNDBS if you route packets to net# 0, not the "known" NDBs.
        (FNS \\ROUTE.XIP)
        ;; the IOCB status dataLost (5) was missing
        (CONSTANTS * |Dove.RS232MiscConstants|)
        (FNS \\DVRS232C.PARSE.STATUS)
        ;;
        (RECORDS RS232C.INIT)
        (VARS (RS232C.DEFAULT.INIT.INFO (|create| RS232C.INIT |using| RS232C.DEFAULT.INIT.INFO)))
        ;;
        (RECORDS RS232CNETWORK.ENCAPSULATION)
        (CONSTANTS \\RS232CNETWORKENCAPSULATION.WORDS \\RS232CNETWORKTYPE.PUP \\RS232CNETWORKTYPE.XIP)
        (FNS \\RS232CNETWORKENCAPSULATE)
        ;; these are because they used RS232C.ENCAPSULATION, which changed
        (RECORDS RS232C.ENCAPSULATION)
        (FNS \\DVRS232C.INPUT.INTERRUPT \\DLRS232C.INPUT.INTERRUPT \\DLRS232C.START.DRIVER
             \\DLRS232C.SEND.PACKET \\RS232C.FORCEOUTPUT \\RS232C.GETNEXTBUFFER \\RS232C.TRACE.PACKET)))

(DECLARE\: FIRST DONTEVAL@LOAD DOCOPY

(FILESLOAD (SYSLOAD)
       DLRS232C)
)

(DECLARE\: FIRST EVAL@LOAD DONTCOPY
```

```
(FILESLOAD (LOADCOMP)
       LLETHER 10MBDRIVER LLNS PUP (SOURCE)
       DLRS232C)
)
```

;; Addition to DOVERS232C Opie definitions

```
(DECLARE\: DONTCOPY
```

;; FOLLOWING DEFINITIONS EXPORTED

```
(DECLARE\: EVAL@COMPILE

(BLOCKRECORD |Dove.i8274.WR7| ((|mustBe7EinSDLC| BYTE)))
)

(RPAQQ |Dove.i8274.WR7.Constants| ((|sdlcFlag| 126)))

(DECLARE\: EVAL@COMPILE

(RPAQQ |sdlcFlag| 126)

(CONSTANTS (|sdlcFlag| 126))
)
)
```

;; END EXPORTED DEFINITIONS

;; This should have been in the DLRS232C file

```
(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \\DLRS232C.OUTPUT.TIMEOUT)
)
```

;; a hint that the code should install itself as the network handler...

```
(RPAQ? *RS232C-NETWORK* T)

(RPAQ? *RS232C-NETWORK-DIALING-TIMEOUT* 30)

(RPAQ? *RS232C-NETWORK-AUTODIAL* )

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS *RS232C-NETWORK* *RS232C-NETWORK-DIALING-TIMEOUT* *RS232C-NETWORK-AUTODIAL*)
)

(DEFINEQ
```

(\\DVRS232C.SET.PARAMETERS

```
  (LAMBDA (PARAMETERLIST)                                   ; Edited 24-Aug-88 16:46 by Briggs
```

;; Set the RS232 line parameters for the 1186.

;;; PARAMETERLIST is in association list format. This function sets the parameters of the IOP accordingly

```
    (DECLARE (GLOBALVARS \\DLRS232C.OUTPUT.TIMEOUT))
    (COND
       (PARAMETERLIST
        (|bind| NOTFOUND (|rsWorkListImage| _ (|\\DoveIO.ByteSwap| (|fetch| (|Dove.RS232DCB| |rsWorkList|)
                                                                    |of| |\\DoveRS232C.DCBPointer|)))
              (|rsCommandWorkListImage| _ (|\\DoveIO.ByteSwap| (|fetch| (|Dove.RS232DCB| |rsCommandWorkList|)
                                                                 |of| |\\DoveRS232C.DCBPointer|)))
              MAJORFLG COMMANDWORK PROP VAL BAUDRATE |for| PROP.VAL |in| PARAMETERLIST
          |do| ((SETQ PROP (CAR PROP.VAL))
                (SETQ VAL (CDR PROP.VAL))
                (SELECTQ PROP
                     (FRAME.TIMEOUT
                          (COND
                             ((<= 0 VAL 255)                    ; Make sure we got a legit value.

                              )
                             (T (\\ILLEGAL.ARG VAL)))
                          (COND
                             ((NEQ VAL (|\\DoveIO.ByteSwap| (|fetch| (|Dove.RS232DCB| |rsFrameTimeoutValue|)
                                                                |of| |\\DoveRS232C.DCBPointer|)))
                              (|replace| (|Dove.RS232DCB| |rsFrameTimeoutValue|) |of| |\\DoveRS232C.DCBPointer|
                                 |with| (|\\DoveIO.ByteSwap| (FIX (TIMES 10 VAL)))))))
                     (CORRESPONDENT
                          (|replace| (|Dove.RS232DCB| |rsTTYHost|) |of| |\\DoveRS232C.DCBPointer|
                             |with| (COND
                                       ((EQ VAL RS232C.CP.TTYHOST)
                                        |\\DoveIO.ByteTRUE|)
                                       (T |\\DoveIO.ByteFALSE|))))
```

```
                    (SYNCH.CHAR                                    ; Not supported on Dove
                            NIL)
                    ((STOP.BITS |NoOfStopBits|)
                         (COND
                            ((<= 0 VAL 2)                          ; Make sure we got a legit value.

                             )
                            (T (\\ILLEGAL.ARG VAL)))
                         (|replace| (RS232C.INIT |NoOfStopBits|) |of| RS232C.DEFAULT.INIT.INFO |with| VAL)
                         (COND
                            ((NEQ (|fetch| (|Dove.i8274.WR4| |stopBits|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                              |rsWR4ofi8274|)
                                                                                    |of| |\\DoveRS232C.DCBPointer|
                                                                                    )))
                                  (SELECTC VAL
                                       (1 |oneStopBit|)
                                       (1.5 |oneAndHalfStopBit|)
                                       (2 |twoStopBits|)
                                       (COND
                                          ((FEQP VAL 1.5)
                                           |oneAndHalfStopBit|)
                                          (T (\\ILLEGAL.ARG VAL)))))
                             (SETQ MAJORFLG T)
                             (SETQ |rsWorkListImage| (BITSET |rsWorkListImage| |rsWorkWR4|))
                             (|replace| (|Dove.i8274.WR4| |stopBits|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                           |rsWR4ofi8274|)
                                                                                 |of| |\\DoveRS232C.DCBPointer|))
                                |with| (SELECTC VAL
                                           (1 |oneStopBit|)
                                           (1.5 |oneAndHalfStopBit|)
                                           (2 |twoStopBits|)
                                           (COND
                                              ((FEQP VAL 1.5)
                                               |oneAndHalfStopBit|)
                                              (T (\\ILLEGAL.ARG VAL))))))))
                    ((PARITY |Parity|)
                         (|replace| (RS232C.INIT |Parity|) |of| RS232C.DEFAULT.INIT.INFO |with| VAL)
                         (COND
                            ((NEQ VAL (COND
                                         ((NOT (|fetch| (|Dove.i8274.WR4| |enableParity|)
                                                   |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR4ofi8274|) |of|
                                                                                              |\\DoveRS232C.DCBPointer|
                                                                                              ))))
                                          'NONE)
                                         ((EQ (|fetch| (|Dove.i8274.WR4| |parityOddOrEven|)
                                                 |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR4ofi8274|) |of|
                                                                                              |\\DoveRS232C.DCBPointer|
                                                                                              )))
                                              |parityOdd|)
                                          'ODD)
                                         ((EQ (|fetch| (|Dove.i8274.WR4| |parityOddOrEven|)
                                                 |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR4ofi8274|) |of|
                                                                                              |\\DoveRS232C.DCBPointer|
                                                                                              )))
                                              |parityEven|)
                                          'EVEN)))
                             (SETQ MAJORFLG T)
                             (SETQ |rsWorkListImage| (BITSET |rsWorkListImage| |rsWorkWR4|))
                             (COND
                                ((EQ VAL 'NONE)
                                 (|replace| (|Dove.i8274.WR4| |enableParity|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                                   |rsWR4ofi8274|)
                                                                                         |of|
                                                                                         |\\DoveRS232C.DCBPointer|
                                                                                         ))
                                    |with| NIL))
                                (T (|replace| (|Dove.i8274.WR4| |enableParity|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                                   |rsWR4ofi8274|)
                                                                                         |of|
                                                                                         |\\DoveRS232C.DCBPointer|
                                                                                         ))
                                    |with| T)
                                   (|replace| (|Dove.i8274.WR4| |parityOddOrEven|)
                                      |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR4ofi8274|) |of|
                                                                                         |\\DoveRS232C.DCBPointer|
                                                                                         ))
                                      |with| (SELECTQ VAL
                                                 (EVEN |parityEven|)
                                                 (ODD |parityOdd|)
                                                 (\\ILLEGAL.ARG VAL)))))))
                    ((CHAR.LENGTH |BitsPerSerialChar|)
                         (COND
                            ((<= 5 VAL 8)                          ; Make sure we got a legit value.

                             )
                            (T (\\ILLEGAL.ARG VAL)))
                         (|replace| (RS232C.INIT |BitsPerSerialChar|) |of| RS232C.DEFAULT.INIT.INFO |with| VAL)
```

```
                    (COND
                       ((NEQ VAL (|fetch| (|Dove.i8274.WR5| |txCharLength|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                              |rsWR5ofi8274|)
                                                                                    |of|
                                                                                 |\\DoveRS232C.DCBPointer|
                                                                                       ))))
                        (SETQ MAJORFLG T)
                        (SETQ |rsWorkListImage| (BITSET |rsWorkListImage| (LOGOR |rsWorkWR3| |rsWorkWR5|)))
                        ;; Set the bits in the UART register:
                        ;;  8-bit chars  1 1
                        ;;  7-bit chars  0 1
                        ;;  6-bit chars  1 0
                        ;;  5-bit chars  0 0
                        (|replace| (|Dove.i8274.WR5| |txCharLength|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                              |rsWR5ofi8274|)
                                                                                    |of|
                                                                                 |\\DoveRS232C.DCBPointer|
                                                                                 ))
                           |with| (SELECTQ VAL
                                     (8 3)
                                     (7 1)
                                     (6 2)
                                     (5 0)
                                     (\\ILLEGAL.ARG VAL)))
                        (|replace| (|Dove.i8274.WR3| |rxCharLength|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                              |rsWR3ofi8274|)
                                                                                    |of|
                                                                                 |\\DoveRS232C.DCBPointer|
                                                                                 ))
                           |with| (SELECTQ VAL
                                     (8 3)
                                     (7 1)
                                     (6 2)
                                     (5 0)
                                     (\\ILLEGAL.ARG VAL))))))
                 ((LINE.SPEED |BaudRate|)
                    (LET ((NV (CDR (SASSOC VAL \\DVRS232C.BAUD.RATES))))
                         (COND
                            (NV (|replace| (**RS232C.INIT** |BaudRate|) |of| RS232C.DEFAULT.INIT.INFO |with| VAL)
                                (SETQ \\DLRS232C.OUTPUT.TIMEOUT (\\RS232C.PACKET.TIMEOUT VAL))
                                (COND
                                   ((AND (SETQ VAL NV)
                                         (NEQ VAL (|\\DoveIO.ByteSwap| (|fetch| (|Dove.RS232DCB|
                                                                                   |rsBaudRateChA|)
                                                                          |of| |\\DoveRS232C.DCBPointer|))))
                                    (SETQ MAJORFLG T)
                                    (SETQ |rsWorkListImage| (BITSET |rsWorkListImage| |rsNewBaudRate|))
                                    (|replace| (|Dove.RS232DCB| |rsBaudRateChA|) |of| |\\DoveRS232C.DCBPointer|
                                       |with| (|\\DoveIO.ByteSwap| VAL))))))))
                 ((FLOW.CONTROL |FlowControl|)
                    (SETQ MAJORFLG T)
                    (|replace| (**RS232C.INIT** |FlowControl|) |of| RS232C.DEFAULT.INIT.INFO |with| VAL)
                    (COND
                       ((OR (LISTP VAL)
                            (AND (OR (STRING.EQUAL VAL "xonxoff")
                                     (STRING.EQUAL VAL "xon-xoff")
                                     (STRING.EQUAL VAL "xon/xoff"))
                                 (SETQ VAL (CONSTANT (|create| RS232C.XONXOFF
                                                               FLAG _ 1
                                                               XON.CHAR _ (CHARCODE ^Q)
                                                               XOFF.CHAR _ (CHARCODE ^S))))))
                        (|replace| (|Dove.RS232FlowControl| |type|) |of| (|fetch| (|Dove.RS232DCB|
                                                                                   |rs232FlowControl|)
                                                                          |of| |\\DoveRS232C.DCBPointer|)
                           |with| (COND
                                     ((ZEROP (|fetch| (RS232C.XONXOFF FLAG) |of| VAL))
                                      |noFlowControl|)
                                     (T |XOnXOffFlowControl|)))
                        (|replace| (|Dove.RS232FlowControl| XO\n) |of| (|fetch| (|Dove.RS232DCB|
                                                                                   |rs232FlowControl|)
                                                                          |of| |\\DoveRS232C.DCBPointer|)
                           |with| (|\\DoveIO.ByteSwap| (OR (|fetch| (RS232C.XONXOFF XON.CHAR) |of| VAL)
                                                          0)))
                        (|replace| (|Dove.RS232FlowControl| |XOff|) |of| (|fetch| (|Dove.RS232DCB|
                                                                                   |rs232FlowControl|)
                                                                          |of| |\\DoveRS232C.DCBPointer|)
                           |with| (|\\DoveIO.ByteSwap| (OR (|fetch| (RS232C.XONXOFF XOFF.CHAR) |of| VAL)
                                                          0))))
                       (T                                  ; No flow control.
                          (|replace| (|Dove.RS232FlowControl| |type|) |of| (|fetch| (|Dove.RS232DCB|
                                                                                   |rs232FlowControl|)
                                                                          |of| |\\DoveRS232C.DCBPointer|)
                             |with| |noFlowControl|))))
                 (LINE.TYPE (LET ((|WR1Base| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR1ofi8274|) |of|
```

```
                                                                       |\\DoveRS232C.DCBPointer|
                                                  )))
                                (|WR3Base| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR3ofi8274|) |of|
                                                                       |\\DoveRS232C.DCBPointer|
                                                  )))
                                (|WR4Base| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR4ofi8274|) |of|
                                                                       |\\DoveRS232C.DCBPointer|
                                                  )))
                                (|WR5Base| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR5ofi8274|) |of|
                                                                       |\\DoveRS232C.DCBPointer|
                                                  )))
                                (|WR7Base| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR7ofi8274|) |of|
                                                                       |\\DoveRS232C.DCBPointer|
                                                  ))))
                               (SELECTC VAL
                                   (RS232C.LT.ASYNCH
                                       (|replace| (RS232C.INIT |LineType|) |of| RS232C.DEFAULT.INIT.INFO
                                          |with| 'ASYNCH)
                                       (|replace| (|Dove.RS232DCB| |rs232Mode|) |of|
                                                                       |\\DoveRS232C.DCBPointer|
                                          |with| |asynchMode|)
                                       (|replace| (|Dove.i8274.WR1| |extInterruptEnable|) |of| |WR1Base|
                                          |with| NIL)
                                       (|replace| (|Dove.i8274.WR3| |enterHuntMode|) |of| |WR3Base|
                                          |with| NIL)
                                       (|replace| (|Dove.i8274.WR3| |rxCRCenable|) |of| |WR3Base|
                                          |with| NIL)
                                       (|replace| (|Dove.i8274.WR3| |addrSearchMode|) |of| |WR3Base|
                                          |with| NIL)
                                       (|replace| (|Dove.i8274.WR3| |syncCharLoadInhibit|) |of| |WR3Base|
                                          |with| T)
                                       (|replace| (|Dove.i8274.WR4| |clockRate|) |of| |WR4Base| |with| |x16clk|
                                          )
                                       (|replace| (|Dove.i8274.WR5| |txCRCenable|) |of| |WR5Base|
                                          |with| NIL)
                                       (SETQ |rsWorkListImage| (BITSET |rsWorkListImage|
                                                                      (LOGOR |rsWorkWR1| |rsWorkWR3|
                                                                             |rsWorkWR4| |rsWorkWR5|
                                                                             |rsChangeMode|)))
                                       (SETQ MAJORFLG T))
                                   (RS232C.LT.BIT.SYNCH
                                       (|replace| (RS232C.INIT |LineType|) |of| RS232C.DEFAULT.INIT.INFO
                                          |with| 'SYNCH)
                                       (|replace| (|Dove.RS232DCB| |rs232Mode|) |of|
                                                                       |\\DoveRS232C.DCBPointer|
                                          |with| |synchMode|)
                                       (|replace| (|Dove.i8274.WR1| |extInterruptEnable|) |of| |WR1Base|
                                          |with| T)
                                       (|replace| (|Dove.i8274.WR3| |enterHuntMode|) |of| |WR3Base|
                                          |with| T)
                                       (|replace| (|Dove.i8274.WR3| |rxCRCenable|) |of| |WR3Base|
                                          |with| T)
                                       (|replace| (|Dove.i8274.WR3| |addrSearchMode|) |of| |WR3Base|
                                          |with| NIL)
                                       (|replace| (|Dove.i8274.WR3| |syncCharLoadInhibit|) |of| |WR3Base|
                                          |with| NIL)
                                       (|replace| (|Dove.i8274.WR4| |clockRate|) |of| |WR4Base| |with| |x1clk|)
                                       (|replace| (|Dove.i8274.WR4| |synchCharControl|) |of| |WR4Base|
                                          |with| |SdlcHdlc|)
                                       (|replace| (|Dove.i8274.WR4| |stopBits|) |of| |WR4Base| |with|
                                                                                  |enableSyncModes|
                                          )
                                       (|replace| (|Dove.i8274.WR5| |txCRCenable|) |of| |WR5Base|
                                          |with| T)
                                       (|replace| (|Dove.i8274.WR5| |modeSDLCOrCRC16|) |of| |WR5Base|
                                          |with| SDLC)
                                       (|replace| (|Dove.i8274.WR7| |mustBe7EinSDLC|) |of| |WR7Base|
                                          |with| |sdlcFlag|)
                                       (SETQ |rsWorkListImage| (BITSET |rsWorkListImage|
                                                                      (LOGOR |rsWorkWR1| |rsWorkWR3|
                                                                             |rsWorkWR4| |rsWorkWR5|
                                                                             |rsWorkWR7| |rsChangeMode|)))
                                       (SETQ MAJORFLG T))
                                   (ERROR "Illegal line type" VAL))))
                 (RESET.RING.HEARD
                      (|replace| (|Dove.RSLatchedStatus| |ringHeard|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                  |rsLatchedStatus|)
                                                                       |of| |\\DoveRS232C.DCBPointer|
                                                                            ))
                         |with| NIL))
                 (RESET.BREAK.DETECTED
                      (|replace| (|Dove.RSLatchedStatus| |breakDetected|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                  |rsLatchedStatus|)
                                                                            |of|
                                                                       |\\DoveRS232C.DCBPointer|
                                                                            ))
                         |with| NIL))
```

```
                    (RESET.DATA.LOST
                         (|replace| (|Dove.RSLatchedStatus| |dataLost|) |of| (LOCF (|fetch| (|Dove.RS232DCB|
                                                                                             |rsLatchedStatus|)
                                                                                        |of| |\\DoveRS232C.DCBPointer|)
                                                                                   )
                                 |with| NIL))
                    ((REQUEST.TO.SEND RTS)
                         (SETQ COMMANDWORK T)
                         (SETQ |rsWorkListImage| (BITSET |rsWorkListImage| |rsWorkWR5|))
                         (COND
                            ((|replace| (|Dove.i8274.WR5| |rts|) |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR5ofi8274|)
                                                                                |of| |\\DoveRS232C.DCBPointer|))
                                 |with| VAL)
                             (SETQ |rsCommandWorkListImage| (BITSET |rsCommandWorkListImage| |rtsCommand|)))
                            (T (SETQ |rsCommandWorkListImage| (BITCLEAR |rsCommandWorkListImage| |rtsCommand|)))
                            ))
                    ((DATA.TERMINAL.READY DTR)
                         (SETQ COMMANDWORK T)
                         (SETQ |rsWorkListImage| (BITSET |rsWorkListImage| |rsWorkWR5|))
                         (COND
                            ((|replace| (|Dove.i8274.WR5| |dtr|) |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR5ofi8274|)
                                                                                |of| |\\DoveRS232C.DCBPointer|))
                                 |with| VAL)
                             (SETQ |rsCommandWorkListImage| (BITSET |rsCommandWorkListImage| |dtrCommand|)))
                            (T (SETQ |rsCommandWorkListImage| (BITCLEAR |rsCommandWorkListImage| |dtrCommand|)))
                            ))
                    (|ModemControl|
                         (|for| SIGNAL |in| VAL |do| (SELECTQ SIGNAL
                                                            (RTS (SETQ COMMANDWORK T)
                                                                 (SETQ |rsWorkListImage| (BITSET |rsWorkListImage|
                                                                                                 |rsWorkWR5|))
                                                                 (COND
                                                                    ((|replace| (|Dove.i8274.WR5| |rts|)
                                                                        |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR5ofi8274|
                                                                                           )
                                                                                     |of| |\\DoveRS232C.DCBPointer|))
                                                                        |with| VAL)
                                                                     (SETQ |rsCommandWorkListImage| (BITSET
                                                                                                    |rsCommandWorkListImage|
                                                                                                          |rtsCommand|
                                                                                                    )))
                                                                    (T (SETQ |rsCommandWorkListImage| (BITCLEAR
                                                                                                    |rsCommandWorkListImage|

                                                                                                          |rtsCommand|
                                                                                                     )))))
                                                            (DTR (SETQ COMMANDWORK T)
                                                                 (SETQ |rsWorkListImage| (BITSET |rsWorkListImage|
                                                                                                 |rsWorkWR5|))
                                                                 (COND
                                                                    ((|replace| (|Dove.i8274.WR5| |dtr|)
                                                                        |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsWR5ofi8274|
                                                                                           )
                                                                                     |of| |\\DoveRS232C.DCBPointer|))
                                                                        |with| VAL)
                                                                     (SETQ |rsCommandWorkListImage| (BITSET
                                                                                                    |rsCommandWorkListImage|
                                                                                                          |dtrCommand|
                                                                                                    )))
                                                                    (T (SETQ |rsCommandWorkListImage| (BITCLEAR
                                                                                                    |rsCommandWorkListImage|

                                                                                                          |dtrCommand|
                                                                                                     )))))
                                                            (SETQ NOTFOUND T))))
                         (SETQ NOTFOUND T)))
               |finally| (COND
                         (COMMANDWORK (|replace| (|Dove.RS232DCB| |rsCommandWorkList|) |of| |\\DoveRS232C.DCBPointer|
                                              |with| (|\\DoveIO.ByteSwap| |rsCommandWorkListImage|))
                                 (SETQ MAJORFLG T)))
                    (COND
                       ((NOT MAJORFLG)
                        (RETURN (NOT NOTFOUND)))
                       (T (SETQ |rsWorkListImage| (BITSET |rsWorkListImage| |workFori8274|))
                          (|replace| (|Dove.RS232DCB| |rsWorkList|) |of| |\\DoveRS232C.DCBPointer| |with| (
                                                                                                  |\\DoveIO.ByteSwap|

                                                                                                    |rsWorkListImage|
                                                                                                       ))
                          (|\\DoveIO.NotifyIOP| (|fetch| (|Dove.RS232FCB| |rs232WorkMask|) |of|
                                                                                             |\\DoveRS232C.FCBPointer|
                                                     ))
                          (|repeatwhile| (BITTEST (|\\DoveIO.ByteSwap| (|fetch| (|Dove.RS232DCB| |rsWorkList|)
                                                                                |of| |\\DoveRS232C.DCBPointer|))
                                                  |workFori8274|)
                             |do| (BLOCK))
                          (RETURN (NOT NOTFOUND)))))))))
```

(\\DVRS232C.INIT

```
(LAMBDA (|BaudRate| |BitsPerSerialChar| |Parity| |NoOfStopBits| |FlowControl| |LineType|)
                                                                    ; Edited 28-Sep-88 23:32 by briggs

;;; Initialize the IOP

    (SETQ |\\DoveRS232C.FCBPointer| (|\\DoveIO.GetHandlerIORegionPtr| |DoveIO.rs232Handler|))
    (SETQ |\\DoveRS232C.DCBPointer| (\\ADDBASE |\\DoveRS232C.FCBPointer| (CONSTANT (MESASIZE |Dove.RS232FCB|))))
    (\\DVRS232C.SHUTDOWN)
    ;; Changes 20-Jan-87 by JDS:
    ;;   FRAME.TIMEOUT from 5 to 32Q, to match Mesa
    ;;  DATA.TERMINAL.READY to NIL to match Mesa
    (\\DLRS232C.CREATE.NDB)
    (SELECTQ |LineType|
        ((ASYNCH ASYNC NIL)
            (\\DVRS232C.SET.PARAMETERS `((FRAME.TIMEOUT . 5)
                                         (CORRESPONDENT \,@ RS232C.CP.TTYHOST)
                                         (RESET.RING.HEARD . T)
                                         (RESET.BREAK.DETECTED . T)
                                         (RESET.DATA.LOST . T)
                                         (REQUEST.TO.SEND . T)
                                         (DATA.TERMINAL.READY)
                                         (LINE.TYPE \,@ RS232C.LT.ASYNCH)
                                         (|NoOfStopBits| \,@ |NoOfStopBits|)
                                         (|Parity| \,@ |Parity|)
                                         (|BitsPerSerialChar| \,@ |BitsPerSerialChar|)
                                         (|BaudRate| \,@ |BaudRate|)
                                         (|FlowControl| \,@ |FlowControl|)))
            (\\DVRS232C.ISSUE.SHORT.COMMAND ON)
            (SETQ \\DLRS232C.OUTPUT.TIMEOUT (\\RS232C.PACKET.TIMEOUT |BaudRate|)))
        ((SYNC SYNCH)
            (\\DVRS232C.SET.PARAMETERS `((FRAME.TIMEOUT . 0)
                                         (CORRESPONDENT \,@ RS232C.CP.NS.ELEMENT)
                                         (RESET.RING.HEARD . T)
                                         (RESET.BREAK.DETECTED . T)
                                         (RESET.DATA.LOST . T)
                                         (REQUEST.TO.SEND . T)
                                         (DATA.TERMINAL.READY)
                                         (LINE.TYPE \,@ RS232C.LT.BIT.SYNCH)
                                         (|Parity| \,@ |Parity|)
                                         (|BitsPerSerialChar| \,@ |BitsPerSerialChar|)
                                         (|BaudRate| \,@ |BaudRate|)))
            (\\DVRS232C.ISSUE.SHORT.COMMAND ON)
            (SETQ \\DLRS232C.OUTPUT.TIMEOUT 0))
        (\\ILLEGAL.ARG |LineType|))
    ;; default init info has been updated by \DVRS232C.SET.PARAMETERS;  the FDEV create fn will create the FDEV if it does not exist, and insert
    ;; this into the FDEV regardless .
    (\\RS232C.CREATE.FDEV RS232C.DEFAULT.INIT.INFO)
    (SETQ \\RS232C.READY T)
    (SETQ \\RS232FLG T)))
```

(\\DLRS232C.INIT

```
(LAMBDA (|BaudRate| |BitsPerSerialChar| |Parity| |NoOfStopBits| |FlowControl|)
                                                                    ; Edited 13-Jul-88 19:20 by Briggs

;;; Initialize the IOP

    ;; let's catch the case when the user said some odd combination of XOnXoff capitalization/hyphenation.
    (|if| (OR (STRING.EQUAL |FlowControl| "xonxoff")
              (STRING.EQUAL |FlowControl| "xon-xoff")
              (STRING.EQUAL |FlowControl| "xon/xoff"))
        |then| (SETQ |FlowControl| '(1 17 19)))
    (COND
       ((NOT (|fetch| (DLRS232C.HDW.CONF RS232C.ABSENT) |of| \\IOPAGE))
        (\\DLRS232C.SHUTDOWN)
        (COND
           ((\\RS232C.ISSUE.SHORT.COMMAND ON)
            (SETQ \\DLRS232C.PARAMETER.CSB (LOCF (|fetch| (IOPAGE DLRS232CPARAMETERCSBLO.11) |of| \\IOPAGE)))
            (|replace| (DLRS232C.PARAMETER.CSB FRAME.TIMEOUT) |of| \\DLRS232C.PARAMETER.CSB |with| 5)
            (|replace| (DLRS232C.PARAMETER.CSB CORRESPONDENT) |of| \\DLRS232C.PARAMETER.CSB |with| RS232C.CP.TTYHOST)
            (|replace| (DLRS232C.PARAMETER.CSB SYNCH.CHAR) |of| \\DLRS232C.PARAMETER.CSB |with| 0)
            (|replace| (DLRS232C.PARAMETER.CSB RESET.RING.HEARD) |of| \\DLRS232C.PARAMETER.CSB |with| T)
            (|replace| (DLRS232C.PARAMETER.CSB RESET.BREAK.DETECTED) |of| \\DLRS232C.PARAMETER.CSB |with| T)
            (|replace| (DLRS232C.PARAMETER.CSB RESET.DATA.LOST) |of| \\DLRS232C.PARAMETER.CSB |with| T)
            (|replace| (DLRS232C.PARAMETER.CSB REQUEST.TO.SEND) |of| \\DLRS232C.PARAMETER.CSB |with| T)
            (|replace| (DLRS232C.PARAMETER.CSB DATA.TERMINAL.READY) |of| \\DLRS232C.PARAMETER.CSB |with| T)
            (|replace| (DLRS232C.PARAMETER.CSB STOP.BITS) |of| \\DLRS232C.PARAMETER.CSB
               |with| (SELECTC |NoOfStopBits|
                           (1 0)
                           (2 1)
                           (ERROR "ILLEGAL NUMBER OF STOP BITS (MUST BE 1 OR 2)" |NoOfStopBits|)))
```

```
            (|replace| (DLRS232C.PARAMETER.CSB LINE.TYPE) |of| \\DLRS232C.PARAMETER.CSB |with| RS232C.LT.ASYNCH)
            (|replace| (DLRS232C.PARAMETER.CSB PARITY) |of| \\DLRS232C.PARAMETER.CSB |with| (SELECTQ |Parity|
                                                                                                (ODD 1)
                                                                                                (EVEN 2)
                                                                                                0))
            (|replace| (DLRS232C.PARAMETER.CSB CHAR.LENGTH) |of| \\DLRS232C.PARAMETER.CSB |with| (IDIFFERENCE
                                                                                                  |BitsPerSerialChar|
                                                                                                  5))
            (|replace| (DLRS232C.PARAMETER.CSB SYNCH.COUNT) |of| \\DLRS232C.PARAMETER.CSB |with| 0)
            (|replace| (DLRS232C.PARAMETER.CSB LINE.SPEED) |of| \\DLRS232C.PARAMETER.CSB
               |with| (OR (CDR (SASSOC |BaudRate| \\DLRS232C.BAUD.RATES))
                          (ERROR "ILLEGAL BAUD RATE" |BaudRate|)))
            (SETQ \\DLRS232C.OUTPUT.TIMEOUT (\\RS232C.PACKET.TIMEOUT |BaudRate|))
            (|replace| (DLRS232C.PARAMETER.CSB INTERRUPT.MASK) |of| \\DLRS232C.PARAMETER.CSB |with| 0)
            (COND
               ((LISTP |FlowControl|)
                (|replace| (DLRS232C.PARAMETER.CSB FLOWCONTROL.ON) |of| \\DLRS232C.PARAMETER.CSB |with| (CAR
                                                                                                        |FlowControl|
                                                                                                        ))
                (|replace| (DLRS232C.PARAMETER.CSB FLOWCONTROL.XON.CHAR) |of| \\DLRS232C.PARAMETER.CSB
                   |with| (OR (CADR |FlowControl|)
                               0))
                (|replace| (DLRS232C.PARAMETER.CSB FLOWCONTROL.XOFF.CHAR) |of| \\DLRS232C.PARAMETER.CSB
                   |with| (OR (CADDR |FlowControl|)
                               0)))
               (T (|replace| (DLRS232C.PARAMETER.CSB FLOWCONTROL.ON) |of| \\DLRS232C.PARAMETER.CSB |with| 0)))
            (\\DLRS232C.ISSUE.SHORT.COMMAND MAJOR.SET.PARAMETERS)
            (COND
               ((|fetch| (DLRS232C.PARAMETER.OUTCOME SUCCESS) |of| \\IOPAGE)
                (\\DLRS232C.CREATE.NDB)
                (\\RS232C.CREATE.FDEV (SETQ RS232C.DEFAULT.INIT.INFO
                                            (|create| RS232C.INIT
                                                   |BaudRate| _ |BaudRate|
                                                   |BitsPerSerialChar| _ |BitsPerSerialChar|
                                                   |Parity| _ |Parity|
                                                   |NoOfStopBits| _ |NoOfStopBits|
                                                   |FlowControl| _ |FlowControl|
                                                   |LineType| _ 'ASYNCH)))
                (SETQ \\RS232C.READY T)
                (SETQ \\RS232FLG T))
               (T (HELP "Error setting parameters for RS232C"))))
         (T (HELP "Unable to activate RS232C interface"))))
      (T (HELP "There is no RS232C hardware in your machine!")))))
```

(**RS232C.INIT**

```
(LAMBDA (BAUDRATE BITSPERCHAR PARITY STOPBITS FLOWCONTROL LINETYPE)
                                                                    ; Edited  8-Jul-88 16:37 by Briggs
                                                                    ; User interface to low level initialization
   (SELECTC \\MACHINETYPE
        (\\DANDELION (COND
                        ((NULL BAUDRATE)
                         (APPLY (FUNCTION \\DLRS232C.INIT)
                                RS232C.DEFAULT.INIT.INFO))
                        ((ZEROP BAUDRATE)
                         (ERROR "Invalid baudrate"))
                        ((LISTP BAUDRATE)
                         (APPLY (FUNCTION \\DLRS232C.INIT)
                                BAUDRATE))
                        (T (\\DLRS232C.INIT BAUDRATE BITSPERCHAR PARITY STOPBITS FLOWCONTROL))))
        (\\DAYBREAK (COND
                       ((NULL BAUDRATE)
                        (APPLY (FUNCTION \\DVRS232C.INIT)
                               RS232C.DEFAULT.INIT.INFO))
                       ((ZEROP BAUDRATE)
                        (ERROR "Invalid baudrate"))
                       ((LISTP BAUDRATE)
                        (APPLY (FUNCTION \\DVRS232C.INIT)
                               BAUDRATE))
                       (T (\\DVRS232C.INIT BAUDRATE BITSPERCHAR PARITY STOPBITS FLOWCONTROL LINETYPE))))
        (ERROR "RS232 is currently not supported on " (MACHINETYPE)))))
```

(**RS232C.HANDLE.PACKET**

```
(LAMBDA (PACKET)                                                    ; Edited 28-Sep-88 00:23 by briggs

        (* * |Handle| \a |received| |packet| |from| |the| RS232 |device|)

   (COND
      ((|type?| FDEV \\RS232C.FDEV)
       (LET* ((INSTREAM (|fetch| (RS232C.DEVICEINFO INSTREAM) |of| (|fetch| (FDEV DEVICEINFO) |of| \\RS232C.FDEV)))
              MAX.BUFFERS PACKET.QUEUE NDB)
             (COND
                ((AND (|type?| STREAM INSTREAM)
                      (|type?| SYSQUEUE (SETQ PACKET.QUEUE (|fetch| (RS232C.STREAM PACKET.QUEUE) |of| INSTREAM)))
                      (EQ (|fetch| (STREAM ACCESS) |of| INSTREAM)
                          'INPUT)
```

```
                    (NEQ 0 (|fetch| (RS232C.ENCAPSULATION RS232C.LENGTH) |of| PACKET)))
                (\\ENQUEUE PACKET.QUEUE PACKET)
                (|add| (|fetch| (RS232C.STREAM QUEUE.LENGTH) |of| INSTREAM)
                       1)
                (NOTIFY.EVENT (|fetch| (RS232C.STREAM EVENT) |of| INSTREAM)))
               ((AND (EQ \\MACHINETYPE \\DAYBREAK)
                     *RS232C-NETWORK*
                     (EQ (|fetch| (|Dove.RS232DCB| |rs232Mode|) |of| |\\DoveRS232C.DCBPointer|)
                         |synchMode|)
                     (EQ (|fetch| RS232CNETWORK.STATUS |of| PACKET)
                         T))
                ;;
                (SELECTC (|ffetch| RS232CNETWORK.TYPE |of| PACKET)
                     (\\RS232CNETWORKTYPE.XIP
                         (|freplace| EPTYPE |of| PACKET |with| \\10MBTYPE.XIP)
                         (\\HANDLE.RAW.PACKET PACKET))
                     (\\RS232CNETWORKTYPE.PUP
                         (SETQ NDB (|ffetch| EPNETWORK |of| PACKET))
                         (COND
                            ((NULL (|ffetch| NDBTRANSLATIONS |of| NDB))
                             (|freplace| NDBTRANSLATIONS |of| NDB |with| (LIST (CONS (|ffetch| (PUP PUPSOURCENET)
                                                                                        |of| PACKET)
                                                                                   (|ffetch| (
                                                                                       RS232CNETWORK.ENCAPSULATION

                                                                                          RS232CNETWORK.SOURCEHOST
                                                                                            )
                                                                                     |of| PACKET))))))
                         (|freplace| EPTYPE |of| PACKET |with| \\10MBTYPE.PUP)
                         (\\HANDLE.RAW.PACKET PACKET))
                     (\\RELEASE.ETHERPACKET PACKET)))
               (T (\\RELEASE.ETHERPACKET PACKET)))))
    (T (\\RELEASE.ETHERPACKET PACKET)))
(|freplace| EPUSERFIELD |of| PACKET |with| NIL)))
```

(**RS232CNETWORKINIT**

```
(LAMBDA (EVENT)                                                    ; Edited 25-Aug-88 10:59 by Briggs
```

;; ensure that RS232 is ready to run (bleah!) -- this code had better not care about being run twice! (it doesn't at the moment)

;; The reason we do this is that we are running *before* the rs232 event fn normally gets run, and we require the initialization of the IOCB pages.
;; At this point \RS232FLG will most likely be NIL (at least on startup it is) so the only thing that will happen is the IOCB page allocation

(**RS232C.EVENTFN** \\RS232C.FDEV EVENT)

;; RS232 may be shutdown at this point, start it with the current parameters

(**RS232C.INIT**)

;; the Codex 2260 modems take a few seconds to actually hang up after DTR drops, we must wait for them to lower DSR, and then a couple more
;; seconds before they are ready to dial again.

```
(COND
   ((RS232MODEMSTATUSP 'DSR)
    (|while| (RS232MODEMSTATUSP 'DSR) |do| (BLOCK) |finally| (DISMISS 2000))))
(COND
   ((OR (FMEMB EVENT '(RESTART NIL))
        *RS232C-NETWORK-AUTODIAL*)
```

;; DTR raised should cause a properly configured modem to dial

```
    (|printout| PROMPTWINDOW T "[Raising DTR]")
    (RS232C.SET.PARAMETERS '((DATA.TERMINAL.READY . T)))
    (|until| (RS232MODEMSTATUSP 'DSR) |forDuration| *RS232C-NETWORK-DIALING-TIMEOUT* |timerUnits| 'SECONDS
       |do| (BLOCK) |finally| (|if| (NOT (RS232MODEMSTATUSP 'DSR))
                                  |then| (|printout| PROMPTWINDOW T "[Data set not ready after "
                                                     *RS232C-NETWORK-DIALING-TIMEOUT* " seconds]")
                                  |else| (|printout| PROMPTWINDOW T "[Data set ready]")))))))
```

(**DLRS232C.CREATE.NDB**

```
(LAMBDA NIL                                                        ; Edited 24-Aug-88 18:21 by Briggs
```

(* * DLRS232C |face| |entry| |for| |driver| |initialization.| |Note| |that| |the| |driver| |resembles| |closely| |the| 10MB |Ethernet|
|driver.| |This| |will| |hopefully| |simplify| |our| |lives| |when| |we| |try| |to| |support| |Clusternet| |communications|)

```
   (COND
      (*RS232C-NETWORK* (SETQ \\DLRS232C.LOCAL.NDB
                               (\\DLRS232C.START.DRIVER (|create| NDB
                                                                  NETTYPE _ 10
                                                                  NDBPUPNET# _ 0
                                                                  NDBNSNET# _ 0
                                                                  NDBTASK# _ 0
                                                                  NDBBROADCASTP _ (FUNCTION NILL)
                                                                  NDBPUPHOST# _ 0
                                                                  NDBTRANSMITTER _ (FUNCTION \\DLRS232C.SEND.PACKET)
                                                                  NDBENCAPSULATOR _ (FUNCTION \\RS232CNETWORKENCAPSULATE)
                                                                  NDBCSB _ NIL
                                                                  NDBETHERFLUSHER _ (FUNCTION RS232C.SHUTDOWN)
                                                                  NDBCANHEARSELF _ NIL
```

```
                                                        NDBIPNET# _ 0
                                                        NDBIPHOST# _ 0
                                                        NDBPUPTYPE _ \\EPT.PUP)))
```

;; if there is a local NDB already link the rs232 one to it.

```
            (COND
               (\\LOCALNDBS (|replace| NDBNEXT |of| \\DLRS232C.LOCAL.NDB |with| \\LOCALNDBS)))
```

;; Set rs232 as the primary network connection

```
            (SETQ \\LOCALNDBS \\DLRS232C.LOCAL.NDB))
      (T (SETQ \\DLRS232C.LOCAL.NDB (\\DLRS232C.START.DRIVER (|create| NDB
                                                              NDBTRANSMITTER _ (FUNCTION
                                                                                \\DLRS232C.SEND.PACKET)
                                                              NDBENCAPSULATOR _ (FUNCTION NILL)
                                                              NDBBROADCASTP _ (FUNCTION NILL)
                                                              NDBETHERFLUSHER _ (FUNCTION RS232C.SHUTDOWN)
                                                              NDBCANHEARSELF _ NIL)))))))
)
```

;; because it needed to know to reinit the rs232c if that was the network interface...

```
(DEFINEQ
```

(\\ETHEREVENTFN

```
  (LAMBDA (DEV EVENT)                                       ; Edited 24-Aug-88 18:21 by Briggs
    (SELECTQ EVENT
         ((NIL AFTERLOGOUT AFTERSYSOUT AFTERMAKESYS AFTERSAVEVM RESTART)
              (PROG (NDB TURNOFFNS TIMESET)
                    (SETQ \\PUP.READY (SETQ \\NS.READY (SETQ \\IP.READY)))
                    (\\SETETHERFLAGS)
                    (\\SETLOCALNSNUMBERS)
                    (\\FLUSHNDBS EVENT)
                    (SETQ \\3MBLOCALNDB (COND
                                           (\\3MBFLG (SETQ \\LOCALNDBS (\\3MB.CREATENDB \\3MBFLG)))))
                    (SETQ \\10MBLOCALNDB (COND
                                            (\\10MBFLG (SETQ NDB (\\10MB.CREATENDB \\10MBFLG))
                                                   (COND
                                                      (\\LOCALNDBS (|replace| NDBNEXT |of| \\LOCALNDBS |with| NDB))
                                                      (T (SETQ \\LOCALNDBS NDB)))
                                                   NDB)))
                    (|for| (DB _ \\LOCALNDBS) |by| (|fetch| NDBNEXT |of| DB) |while| DB
                       |do| (\\LOCKWORDS DB (|fetch| DTDSIZE |of| (\\GETDTD (NTYPX DB)))))
```

;; if the rs232 device is acting as the network connection it should be reinitialized before we attempt to start NS

```
                    (COND
                       (*RS232C-NETWORK* (\\RS232CNETWORKINIT EVENT)))
                    (COND
                       ((OR \\NSFLG (SETQ TURNOFFNS \\10MBFLG))
```

(* |Start| NS |before| |Pup| |so| |that| |when| |on| 10 |we| |can| |find| |out| |our| |pup| |number,| |which| |is| |done| |via| NS |protocol|)

```
                           (\\NSINIT EVENT)
                           (SETQ TIMESET (\\NS.SETTIME))))
                    (\\STARTPUP EVENT)
                    (OR TIMESET (AND (EQ \\PUP.READY T)
                                     (\\PUP.SETTIME))
                           (SELECTC \\MACHINETYPE
                                (\\DANDELION (NEQ 0 (|fetch| DLTODVALID |of| \\IOPAGE)))
                                (\\DAYBREAK (|\\DoveMisc.TODValid|))
                                (IGREATERP (IDATE)
                                       (CONSTANT (IDATE " 1-JAN-84 12:00"))))
                           (|printout| PROMPTWINDOW T "[Time not set]"))
                    (COND
                       (TURNOFFNS (STOPNS)))
                    (COND
                       (\\GATEWAYFLG (\\INIT.GATEWAY)))
              T))
         ((BEFOREMAKESYS BEFORELOGOUT BEFORESYSOUT BEFORESAVEVM)
              (COND
                 ((EQ EVENT 'BEFORESAVEVM)                  (* |Save| |passwords| |in| |place| |outside| |vmem| |to| |avoid|
                                                            |having| |to| |reenter| |them| |later|)
                  (\\STASH.PASSWORDS))
                 (T                                         (* N\o |need| |to| |flush| |this| |before| SAVEVM)
                    (CLRHASH \\ETHERPORTS)))
              (CLRHASH LOGINPASSWORDS))
         NIL)))
)
```

;; because it was closing too many sockets when it shouldn't have...

```
(DEFINEQ
```

(TURN.OFF.ETHER

```
  (LAMBDA NIL                                                    ; Edited 17-Oct-88 12:14 by Briggs
    (BREAKCONNECTION T)
    (DEL.PROCESS '\\PUPGATELISTENER)
    (CLOSEPUPSOCKET (OPENPUPSOCKET \\PUPSOCKET.ROUTING T)
           T)
    (DEL.PROCESS '\\NSGATELISTENER)
    (CLOSENSOCKET (OPENNSOCKET |\\NS.WKS.RoutingInformation| T)
          T)
    (DEL.PROCESS '\\IPLISTENER)
    (DEL.PROCESS '\\IPGATELISTENER)
    (\\FLUSHNDBS 'RESTART)))

)

;; because it didn't check to see if it had an NDBIQ or NDBTQ before dequeuing...

(DEFINEQ

(\\DVRS232C.SHUTDOWN
  (LAMBDA NIL                                                    ; Edited 12-Jul-88 00:49 by Briggs

          (* * |Disables| RS232C |if| |currently| |running|)

    (LET (PACKET)
         (COND
            (\\DLRS232C.LOCAL.NDB (SETQ \\RS232C.READY (SETQ \\RS232FLG NIL))
                   (DEL.PROCESS (|fetch| NDBWATCHER |of| \\DLRS232C.LOCAL.NDB))
                   (BLOCK)
                   (\\DVRS232C.ABORT.QUEUE (|fetch| (|Dove.RS232FCB| |rsQueueRxChA|) |of| |\\DoveRS232C.FCBPointer|)
                          )
                   (\\RS232C.ISSUE.SHORT.COMMAND ABORT.INPUT)
                   (\\DVRS232C.ABORT.QUEUE (|fetch| (|Dove.RS232FCB| |rsQueueTxChA|) |of| |\\DoveRS232C.FCBPointer|)
                         )
                   (\\RS232C.ISSUE.SHORT.COMMAND ABORT.OUTPUT)
                   (\\RS232C.ISSUE.SHORT.COMMAND OFF)
                   (|\\Dove.ClearQueueBlock| (|fetch| (|Dove.RS232FCB| |rsQueueTxChA|) |of|
                                                                                |\\DoveRS232C.FCBPointer|
                                                 ))
                   (|\\Dove.ClearQueueBlock| (|fetch| (|Dove.RS232FCB| |rsQueueRxChA|) |of|
                                                                                |\\DoveRS232C.FCBPointer|
                                                 ))
                   (AND (|fetch| NDBIQ |of| \\DLRS232C.LOCAL.NDB)
                        (|while| (SETQ PACKET (\\DEQUEUE (|fetch| NDBIQ |of| \\DLRS232C.LOCAL.NDB)))
                           |do| (\\TEMPUNLOCKPAGES PACKET (FOLDHI \\DLRS232C.DEFAULT.PACKET.LENGTH BYTESPERPAGE))
                                (\\RELEASE.ETHERPACKET PACKET)))
                   (AND (|fetch| NDBTQ |of| \\DLRS232C.LOCAL.NDB)
                        (|while| (SETQ PACKET (\\DEQUEUE (|fetch| NDBTQ |of| \\DLRS232C.LOCAL.NDB)))
                           |do| (\\TEMPUNLOCKPAGES PACKET (FOLDHI \\DLRS232C.DEFAULT.PACKET.LENGTH BYTESPERPAGE))
                                (\\RELEASE.ETHERPACKET PACKET)))))))

)

;; because etherpackets are actually 2 pages, and we want to use it all if necessary (bytesperpage - etherheader + 8bytes from encapsulation which we
;; put data into)

(RPAQQ \\DLRS232C.DEFAULT.PACKET.LENGTH 968)

(RPAQQ \\RS232C.OUTPUT.PACKET.LENGTH 968)

;; because translate.3to10 isn't the right thing on a phone line... in actual fact, the things that call translate.3to10 should know better than to do so on a
;; phone line, but since we're masquerading as a 10Mb/s net due to some other bogosity this is the easiest place to fix it.

(DEFINEQ

(\\TRANSLATE.3TO10
  (LAMBDA (PUPHOSTNUMBER NDB)                                    ; Edited 11-Jul-88 17:12 by Briggs
    ;; Translate from an PUPHOSTNUMBER to a NSHOSTNUMBER for the indicated network.  If we don't have the translation, we initiate a probe for
    ;; it and return NIL

    ;; Bletch: if the NDB we got was the \DLRS232C.LOCAL.NDB then we'll just return a bogus nshostnumber -- no one cares in this case.

    (COND
       ((EQ NDB \\DLRS232C.LOCAL.NDB)
        \\RS232CNETWORK.NSHOSTNUMBER)
       ((CADR (ASSOC PUPHOSTNUMBER (|ffetch| NDBTRANSLATIONS |of| (\\DTEST NDB 'NDB)))))
       ((PROG ((MYPUPHOSTNUMBER (|ffetch| NDBPUPHOST# |of| NDB))
               PACKET)
              (COND
                 ((EQ MYPUPHOSTNUMBER 0)                         ; We don't know who we are yet
                  (RETURN)))
              (SETQ PACKET (\\ALLOCATE.ETHERPACKET))
              (|replace| EPTYPE |of| PACKET |with| \\EPT.3TO10)
              (|freplace| TRANSOPERATION |of| PACKET |with| \\TRANS.OP.REQUEST)
              (|freplace| TRANSPUPHOST |of| PACKET |with| PUPHOSTNUMBER)
              (|freplace| TRANSSENDERNSHOST |of| PACKET |with| (\\LOCALNSHOSTNUMBER))
              (|freplace| TRANSSENDERPUPHOST |of| PACKET |with| MYPUPHOSTNUMBER)
```

```
              (ENCAPSULATE.ETHERPACKET NDB PACKET BROADCASTNSHOSTNUMBER \\TRANS.DATALENGTH \\10MBTYPE.3TO10)
              (AND XIPTRACEFLG (\\MAYBEPRINTPACKET PACKET 'PUT))
              (|freplace| EPREQUEUE |of| PACKET |with| 'FREE)
              (TRANSMIT.ETHERPACKET NDB PACKET)                    ; We didn't find out this time, but we will later on
              (RETURN))))))

)

(RPAQ \\RS232CNETWORK.NSHOSTNUMBER \\MY.NSHOSTNUMBER)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \\RS232CNETWORK.NSHOSTNUMBER \\DLRS232C.LOCAL.NDB)
)
```

;; because this one has some bugs fixed, and it must not reinitialize a running RS232 driver.

```
(DEFINEQ
```

(\\RS232C.EVENTFN

```
  (LAMBDA (DEVICE EVENT)                                           ; Edited 21-Oct-88 12:26 by Briggs
    (SELECTQ EVENT
         ((AFTERLOGOUT AFTERMAKESYS AFTERSYSOUT AFTERSAVEVM)
              (COND
                 ((AND \\DLRS232C.IOCB.PAGE \\DLRS232C.IOCB.ENDPAGE)
                  (|bind| (BASE _ \\DLRS232C.IOCB.PAGE)
                         DONE |until| DONE |do| (\\DONEWEPHEMERALPAGE BASE T)
                                                (COND
                                                   ((NEQ BASE \\DLRS232C.IOCB.ENDPAGE)
                                                    (SETQ BASE (\\ADDBASE BASE WORDSPERPAGE)))
                                                   (T (SETQ DONE T))))))
              ;; don't try to initialize rs232 if it is already running (from the \ETHEREVENTFN when using the rs232 network interface.
              (COND
                 ((AND \\RS232FLG (NOT \\RS232C.READY)
                       (SELECTC \\MACHINETYPE
                            (\\DANDELION (NOT (|fetch| (DLRS232C.HDW.CONF RS232C.ABSENT) |of| \\IOPAGE)))
                            (\\DAYBREAK T)
                            NIL))
                  (RS232C.INIT (OR (AND \\RS232C.FDEV (|fetch| (RS232C.DEVICEINFO INIT) |of| (|fetch| (FDEV DEVICEINFO)
                                                                                                |of| \\RS232C.FDEV)))
                                   RS232C.DEFAULT.INIT.INFO)))))
        NIL)))

)
```

;; because it has to use the \LOCALNDBS if you route packets to net# 0, not the "known" NDBs.

```
(DEFINEQ
```

(\\ROUTE.XIP

```
  (LAMBDA (XIP READONLY)                                           ; Edited 13-Oct-88 18:02 by Briggs
```

;; Encapsulates XIP, choosing the right network and immediate destination host. Returns an NDB for the transmission. Unless READONLY is ;; true, defaults source and destination nets if needed

;;; modified to use \LOCALNDBS instead of (OR known ndbs...)

```
    (GLOBALRESOURCE \\ROUTEBOX.HOST (PROG ((NET (|fetch| XIPDESTNET |of| XIP))
                                           PDH ROUTE NDB)
                                          (COND
                                             ((EQ 0 NET)
                                              (OR (SETQ NDB \\LOCALNDBS)
                                                  (RETURN)))
                                             ((SETQ ROUTE (\\LOCATE.NSNET NET))
                                              (SETQ NDB (|fetch| RTNDB |of| ROUTE)))
                                             (T (RETURN)))
                                          (SETQ PDH (COND
                                                       ((AND ROUTE (NEQ (|fetch| RTHOPCOUNT |of| ROUTE)
                                                                        0))
                                                        (|fetch| RTGATEWAY# |of| ROUTE))
                                                       ((EQ (|fetch| NETTYPE |of| NDB)
                                                            10)
                                                        (LOADNSHOSTNUMBER (LOCF (|fetch| XIPDESTWORD1 |of| XIP))
                                                                          \\ROUTEBOX.HOST)
                                                        \\ROUTEBOX.HOST)
                                                       ((EQNSHOSTNUMBER (|fetch| XIPDESTHOST |of| XIP)
                                                                        BROADCASTNSHOSTNUMBER)
                                                                                  ; On 3, broadcast goes to zero
                                                        0)
                                                       ((PROGN (LOADNSHOSTNUMBER (LOCF (|fetch| XIPDESTWORD1
                                                                                                |of| XIP))
                                                                                 \\ROUTEBOX.HOST)
                                                               (\\TRANSLATE.10TO3 \\ROUTEBOX.HOST NDB)))
                                                       (T (RETURN))))
                                          (|replace| EPNETWORK |of| XIP |with| NDB)
                                          (ENCAPSULATE.ETHERPACKET NDB XIP PDH (|fetch| XIPLENGTH |of| XIP)
```

```
                                                \\EPT.XIP)
                                        (COND
                                           ((NOT READONLY)
                                            (COND
                                               ((EQ 0 NET)
                                                (|replace| XIPDESTNET |of| XIP |with| (|fetch| NDBNSNET# |of| NDB))))
                                            (|replace| XIPSOURCENET |of| XIP |with| (|fetch| NDBNSNET# |of| NDB))))
                                        (RETURN NDB)))))

)

;; the IOCB status dataLost (5) was missing

(RPAQQ |Dove.RS232MiscConstants|
       ((|noFlowControl| 0)
        (|XOnXOffFlowControl| 256)
        (|asynchMode| 0)
        (|synchMode| 1)
        (|IOCBpollRxOrTx| 0)
        (|IOCBcomplete| 1)
        (|IOCBaborted| 2)
        (|IOCBframeTimeout| 3)
        (|IOCBdisaster| 4)
        (|rsNoClient| 0)
        (|rsNormal| 1)
        (|rsDebugger| 2)
        (|latchRingHeard| 32)
        (|latchDataLost| 64)
        (|latchBreakDet| 128)))

(DECLARE\: EVAL@COMPILE

(RPAQQ |noFlowControl| 0)

(RPAQQ |XOnXOffFlowControl| 256)

(RPAQQ |asynchMode| 0)

(RPAQQ |synchMode| 1)

(RPAQQ |IOCBpollRxOrTx| 0)

(RPAQQ |IOCBcomplete| 1)

(RPAQQ |IOCBaborted| 2)

(RPAQQ |IOCBframeTimeout| 3)

(RPAQQ |IOCBdisaster| 4)

(RPAQQ |rsNoClient| 0)

(RPAQQ |rsNormal| 1)

(RPAQQ |rsDebugger| 2)

(RPAQQ |latchRingHeard| 32)

(RPAQQ |latchDataLost| 64)

(RPAQQ |latchBreakDet| 128)

(CONSTANTS (|noFlowControl| 0)
       (|XOnXOffFlowControl| 256)
       (|asynchMode| 0)
       (|synchMode| 1)
       (|IOCBpollRxOrTx| 0)
       (|IOCBcomplete| 1)
       (|IOCBaborted| 2)
       (|IOCBframeTimeout| 3)
       (|IOCBdisaster| 4)
       (|rsNoClient| 0)
       (|rsNormal| 1)
       (|rsDebugger| 2)
       (|latchRingHeard| 32)
       (|latchDataLost| 64)
       (|latchBreakDet| 128))
)

(DEFINEQ

(\\**DVRS232C.PARSE.STATUS**
  (LAMBDA (IOCB)                                                ; Edited 11-Jul-88 14:14 by Briggs
    (LET ((|rsIOCBType| (|fetch| (|Dove.RS232IOCB| |rsIOCBType|) |of| IOCB)))
         (LET ((STATUS (SELECTC (|fetch| (|Dove.RS232IOCB| |currentOpStatus|) |of| IOCB)
                                (|IOCBpollRxOrTx|
                                    '|PollRxOrTx|)
```

```
                            (|IOCBaborted|
                               '|Aborted|)
                            (|IOCBdisaster|
                               '|Disaster|)
                            (|IOCBframeTimeout|
                                 (COND
                                    ((EQ |rsIOCBType| |rsIOCBTypeRx|)
                                     T)
                                    (T '|FrameTimeout|)))
                            (|IOCBdataLost|
                               '|DataLost|)
                            (|IOCBcomplete|
                                 (COND
                                    ((EQ |rsIOCBType| |rsIOCBTypeTx|)
                                     T)
                                    (T (LET ((|rsIocbSB1Base| (LOCF (|fetch| (|Dove.RS232IOCB| |rsIocbStatusByte1|)
                                                                        |of| IOCB))))
                                            (COND
                                               ((|fetch| (|Dove.RSLatchedStatus| |dataLost|)
                                                   |of| (LOCF (|fetch| (|Dove.RS232DCB| |rsLatchedStatus|)
                                                                 |of| |\\DoveRS232C.DCBPointer|)))
                                                '|DataLost|)
                                               ((|fetch| (|Dove.i8274.RR1| |rxOverrunError|) |of| |rsIocbSB1Base|)
                                                '|DataLost|)
                                               ((|fetch| (|Dove.i8274.RR1| |parityError|) |of| |rsIocbSB1Base|)
                                                '|ParityError|)
                                               ((|fetch| (|Dove.i8274.RR1| |crcFramingError|) |of| |rsIocbSB1Base|)
                                                (COND
                                                   ((EQ (|fetch| (|Dove.RS232DCB| |rs232Mode|) |of|
                                                                                        |\\DoveRS232C.DCBPointer|
                                                                )
                                                        |asynchMode|)
                                                    '|asynchFramingError|)
                                                   ((|fetch| (|Dove.i8274.RR1| |endOfFrameSDLCMode|) |of|
                                                                                                |rsIocbSB1Base|
                                                             )
                                                    '|checksumError|)
                                                   (T T)))
                                               (T T))))))
                                  '|Disaster|)))
                (COND
                   ((AND (NEQ STATUS T)
                         (NEQ STATUS '|Aborted|)
                         STATUS)
                    (COND
                       ((OR (EQ \\RS232C.REPORT.STATUS T)
                            (AND (EQ \\RS232C.REPORT.STATUS 'OUTPUT)
                                 (EQ |rsIOCBType| |rsIOCBTypeTx|))
                            (AND (EQ \\RS232C.REPORT.STATUS 'INPUT)
                                 (EQ |rsIOCBType| |rsIOCBTypeRx|)))
                        (|printout| RS232C.ERROR.STREAM T "RS232 error: " (SELECTQ STATUS
                                                                                (|Aborted| "Operation aborted")
                                                                                (|Disaster| "Error during
                                                                                            transmission, data
                                                                                            lost")
                                                                                (|FrameTimeout|
                                                                                    "transmission timeout")
                                                                                (|DataLost| "data lost")
                                                                                (|ParityError|
                                                                                    "parity error")
                                                                                (|asynchFramingError|
                                                                                    "transmission frame out of
                                                                                    sync")
                                                                                (|checksumError|
                                                                                    "checksum error")
                                                                                STATUS)
                         T)))))
                STATUS))))

)

;;

(DECLARE\: EVAL@COMPILE

(RECORD RS232C.INIT (|BaudRate| |BitsPerSerialChar| |Parity| |NoOfStopBits| |FlowControl| |LineType|))
)

(RPAQ **RS232C.DEFAULT.INIT.INFO** (|create| RS232C.INIT |using| RS232C.DEFAULT.INIT.INFO))

;;

(DECLARE\: EVAL@COMPILE

(ACCESSFNS RS232CNETWORK.ENCAPSULATION ((RS232ETHERBASE (LOCF (|fetch| (ETHERPACKET EPENCAPSULATION) |of| DATUM)))
                                        (RS232CNETWORK.STATUS (|fetch| (ETHERPACKET EPUSERFIELD) |of| DATUM)
```

```
                                                    (|replace| (ETHERPACKET EPUSERFIELD) |of| DATUM |with| NEWVALUE)))
       (BLOCKRECORD RS232ETHERBASE ((RS232CNETWORK.LENGTH WORD)
                                    ;; Length of data in words
                                    (NIL 3 WORD)
                                    ;; Padding to align sync packet data with EPBODY
                                    (RS232CNETWORK.TYPE WORD)
                                    ;; phone encapsulation type
                                    (RS232CNETWORK.SOURCEWORD1 3 WORD)
                                    ;; 48 bit source host number
                                    (RS232CNETWORK.DATA WORD))
              (ACCESSFNS RS232CNETWORK.SOURCEWORD1 (RS232CNETWORK.SOURCEHOST (\\LOADNSHOSTNUMBER (LOCF DATUM))
                                                                             (\\STORENSHOSTNUMBER (LOCF DATUM)
                                                                                    NEWVALUE)))))
)

(DECLARE\: EVAL@COMPILE

(RPAQQ \\RS232CNETWORKENCAPSULATION.WORDS 4)

(RPAQQ \\RS232CNETWORKTYPE.PUP 64)

(RPAQQ \\RS232CNETWORKTYPE.XIP 192)

(CONSTANTS \\RS232CNETWORKENCAPSULATION.WORDS \\RS232CNETWORKTYPE.PUP \\RS232CNETWORKTYPE.XIP)
)

(DEFINEQ

(\\RS232CNETWORKENCAPSULATE
  (LAMBDA (NDB PACKET PDH LENGTH TYPE)                    ; Edited 24-Aug-88 12:13 by briggs
    ;; encapsulates packets for transmission over the rs232 synchronous link
    (SELECTC TYPE
         (\\EPT.XIP ;; XIPs have a 1 word checksum which is not included in the length, so we must account for it here in the transmission length.
                    (|replace| RS232CNETWORK.LENGTH |of| PACKET |with| (IPLUS LENGTH (UNFOLD
                                                                                    \\RS232CNETWORKENCAPSULATION.WORDS
                                                                                           BYTESPERWORD)
                                                                                    2)))
         (\\EPT.PUP ;; PUPs have a 1 word checksum which is not included in the length, so we must account for it here in the transmission length.
                    (|replace| RS232CNETWORK.LENGTH |of| PACKET |with| (IPLUS LENGTH (UNFOLD
                                                                                    \\RS232CNETWORKENCAPSULATION.WORDS
                                                                                           BYTESPERWORD)
                                                                                    2)))
         (SHOULDNT "Bad type for encapsulation"))
    (|replace| RS232CNETWORK.TYPE |of| PACKET |with| (SELECTC TYPE
                                                          (\\EPT.XIP \\RS232CNETWORKTYPE.XIP)
                                                          (\\EPT.PUP \\RS232CNETWORKTYPE.PUP)
                                                          (SHOULDNT "Bad type for encapsulation")))
    (|replace| RS232CNETWORK.SOURCEHOST |of| PACKET |with| \\MY.NSHOSTNUMBER)
    PACKET))

)

;; these are because they used RS232C.ENCAPSULATION, which changed

(DECLARE\: EVAL@COMPILE

(ACCESSFNS RS232C.ENCAPSULATION ((RS232CBASE (LOCF (|fetch| (ETHERPACKET EPENCAPSULATION) |of| DATUM)))
                                 (RS232C.STATUS (|fetch| (ETHERPACKET EPUSERFIELD) |of| DATUM)
                                                (|replace| (ETHERPACKET EPUSERFIELD) |of| DATUM |with| NEWVALUE)))
       (BLOCKRECORD RS232CBASE ((RS232C.LENGTH WORD)            (* |Length| |of| |packet| |in| |words|)
                                (NIL 3 WORD)                    (* |padding| |to| |align| |sync| |data| "body" |with| EPBODY)
                                (RS232C.DATA WORD)              (* |Data| |starts| |here|)
                                )
              (ACCESSFNS RS232C.DATA ((RS232C.PACKET.BASE (LOCF DATUM)))))
       (TYPE? (|type?| ETHERPACKET DATUM)))
)

(DEFINEQ

(\\DVRS232C.INPUT.INTERRUPT
  (LAMBDA (NDB)                                           ; Edited  9-Jul-88 17:51 by Briggs
    ;; Poll the IOP to see if there are any input requests completed
    (LET ((PACKET (|fetch| SYSQUEUEHEAD |of| (|fetch| NDBIQ |of| NDB)))
          IOCB ACCEPTSTATUS)
         (COND
            ((AND PACKET (SETQ IOCB (|fetch| EPNETWORK |of| PACKET))
                  (NEQ (|fetch| (|Dove.RS232IOCB| |currentOpStatus|) |of| IOCB)
                       |IOCBpollRxOrTx|))
             (\\DEQUEUE (|fetch| NDBIQ |of| NDB))
```

```
            (|replace| RS232C.STATUS |of| PACKET |with| (SETQ ACCEPTSTATUS (\\DVRS232C.PARSE.STATUS IOCB)))
            (\\DVRS232C.DEQUEUE.IOCB IOCB (|fetch| (|Dove.RS232FCB| |rsQueueRxChA|) |of| |\\DoveRS232C.FCBPointer|)
                  )
            (PROG ((LENGTH (|\\DoveIO.ByteSwap| (|fetch| (|Dove.RS232IOCB| |rsTransferCountChA|) |of| IOCB))))
                  (|replace| (RS232C.ENCAPSULATION RS232C.LENGTH) |of| PACKET |with| LENGTH)
                  (|replace| EPNETWORK |of| PACKET |with| NDB)
                  (COND
                     ((IGREATERP LENGTH (CONSTANT (UNFOLD \\MIN2PAGEBUFLENGTH BYTESPERWORD)))
                      ;; The DLion ether code doesn't dirty the pages of an etherpacket.  There are hints in the Mesa RS232C face that the IOP
                      ;; doesn't dirty the pages of an RS232C packet either.  Hence, we dirty the second page of the packet if it's long enough
                      ;; to warrent it
                      (\\PUTBASE PACKET (SUB1 (ITIMES WORDSPERPAGE 2))
                             0)))
                  (COND
                     (\\RS232FLG (\\ENQUEUE \\DLRS232C.RAW.PACKET.QUEUE PACKET)))
                  (\\TEMPUNLOCKPAGES PACKET (FOLDHI \\DLRS232C.DEFAULT.PACKET.LENGTH BYTESPERPAGE)))

         (* * I\f RS232 |is| |still| |alive,| |queue| |up| |another| |packet| |for| |the| |receiver|)

            (COND
               (\\RS232FLG (SETQ PACKET (\\DLRS232C.ALLOCATE.PACKET))
                      (\\TEMPLOCKPAGES PACKET (FOLDHI \\DLRS232C.DEFAULT.PACKET.LENGTH BYTESPERPAGE))
                      (|replace| EPNETWORK |of| PACKET |with| IOCB)
                      (\\DLRS232C.QUEUE.INPUT.IOCB IOCB (|fetch| RS232C.PACKET.BASE |of| PACKET)
                             \\DLRS232C.DEFAULT.PACKET.LENGTH)
                      (\\ENQUEUE (|fetch| NDBIQ |of| NDB)
                             PACKET)))))
      ACCEPTSTATUS)))
```

(**DLRS232C.INPUT.INTERRUPT**

```
(LAMBDA (NDB)                                                    (* |ejs:| " 7-Sep-85 22:01")

         (* * |Poll| |the| IOP |to| |see| |if| |there| |are| |any| |input| |requests| |completed|)

   (LET ((PACKET (|fetch| SYSQUEUEHEAD |of| (|fetch| NDBIQ |of| NDB)))
         IOCB NEXTIOCB ACCEPTSTATUS)
        (|if| (AND PACKET \\DLRS232C.ACTIVE.GET (NOT (|fetch| (DLRS232C.IOP.GET.FLAG BUSY) |of| \\IOPAGE))
                   (SETQ IOCB (|fetch| EPNETWORK |of| PACKET))
                   (EQ \\DLRS232C.ACTIVE.GET IOCB))
            |then| (\\DEQUEUE (|fetch| NDBIQ |of| NDB))
                   (|if| (NULL (SETQ \\DLRS232C.GET.QUEUE.START (SETQ NEXTIOCB (|fetch| (DLRS232C.IOCB NEXT)
                                                                                  |of| IOCB))))
                       |then| (SETQ \\DLRS232C.GET.QUEUE.END NIL))
                   (SETQ ACCEPTSTATUS (OR (|fetch| (DLRS232C.IOCB SUCCESS) |of| IOCB)
                                          (|fetch| (DLRS232C.IOCB TRANSFER.STATUS) |of| IOCB)))
                   (PROG ((LENGTH (|fetch| (DLRS232C.IOCB RETURNED.BYTE.COUNT) |of| IOCB)))
                         (|replace| (RS232C.ENCAPSULATION RS232C.LENGTH) |of| PACKET |with| LENGTH)
                         (|replace| EPNETWORK |of| PACKET |with| NDB)
                         (COND
                            ((AND (EQ \\MACHINETYPE \\DANDELION)
                                  (IGREATERP LENGTH (CONSTANT (UNFOLD \\MIN2PAGEBUFLENGTH BYTESPERWORD))))

         (* * |The| |DLion| |ether| |code| |doesn't| |dirty| |the| |pages| |of| |an| |etherpacket.|
         |There| |are| |hints| |in| |the| |Mesa| RS232C |face| |that| |the| IOP |doesn't| |dirty| |the| |pages| |of| |an| RS232C |packet|
         |either.| |Hence,| |we| |dirty| |the| |second| |page| |of| |the| |packet| |if| |it's| |long| |enough| |to| |warrent| |it|)

                               (\\PUTBASE PACKET (SUB1 (ITIMES WORDSPERPAGE 2))
                                      0)))
                            (\\ENQUEUE \\DLRS232C.RAW.PACKET.QUEUE PACKET)
                            (\\DLRS232C.FINISH.GET.AND.PUT IOCB)
                            (|if| NEXTIOCB
                                |then| (\\DLRS232C.START.INPUT NEXTIOCB))
                            (\\TEMPUNLOCKPAGES PACKET (FOLDHI \\DLRS232C.DEFAULT.PACKET.LENGTH BYTESPERPAGE)))
                         (PROGN (SETQ PACKET (\\DLRS232C.ALLOCATE.PACKET))
                                (\\TEMPLOCKPAGES PACKET (FOLDHI \\DLRS232C.DEFAULT.PACKET.LENGTH BYTESPERPAGE))
                                (|replace| EPNETWORK |of| PACKET |with| IOCB)
                                (\\DLRS232C.QUEUE.INPUT.IOCB IOCB (|fetch| RS232C.PACKET.BASE |of| PACKET)
                                       \\DLRS232C.DEFAULT.PACKET.LENGTH)
                                (\\ENQUEUE (|fetch| NDBIQ |of| NDB)
                                       PACKET)))
      (COND
         ((AND ACCEPTSTATUS (NEQ ACCEPTSTATUS T)
               (OR (EQ \\RS232C.REPORT.STATUS T)
                   (EQ \\RS232C.REPORT.STATUS 'INPUT)))
          (\\DLRS232C.PARSE.STATUS ACCEPTSTATUS 'IN)))
      ACCEPTSTATUS)))
```

(**DLRS232C.START.DRIVER**

```
(LAMBDA (NDB RESTARTFLG)                                         (* |ejs:| "19-Jun-85 17:52")

         (* * |Device-specific| RS232C |startup|)

         (* * |Get| |some| IOCB |space|)
```

```
    (OR (\\DLRS232C.ALLOCATE.IOCBS)
        (ERROR "Unable to create IOCB pool"))
    (|replace| NDBTQ |of| NDB |with| (|create| SYSQUEUE))

          (* * |Initialize| |the| |device| |at| |the| IOP |level|)

    (\\DLRS232C.STARTUP NDB)

          (* * |Load| |the| |initial| RS232C |input| |queue|)

    (LET ((LEN 0)
          (IQ (|fetch| NDBIQ |of| NDB)))
         (COND
            (IQ (SETQ LEN (\\DLRS232C.LOADINPUTQ NDB (|fetch| SYSQUEUEHEAD |of| IQ))))
            (T (|replace| NDBIQ |of| NDB |with| (SETQ IQ (|create| SYSQUEUE)))))
         (|bind| IOCB PACKET |to| (IDIFFERENCE \\DLRS232C.IDEAL.INPUT.LENGTH LEN)
            |while| (SETQ IOCB (\\DLRS232C.GET.IOCB 'INPUT)) |do| (SETQ PACKET (\\DLRS232C.ALLOCATE.PACKET))
                                                                 (\\TEMPLOCKPAGES PACKET (FOLDHI
                                                                                          \\DLRS232C.DEFAULT.PACKET.LENGTH
                                                                                             BYTESPERPAGE))
                                                                 (|replace| EPNETWORK |of| PACKET |with| IOCB)
                                                                 (\\DLRS232C.QUEUE.INPUT.IOCB IOCB
                                                                        (|fetch| (RS232C.ENCAPSULATION
                                                                                     RS232C.PACKET.BASE)
                                                                           |of| PACKET)
                                                                        \\DLRS232C.DEFAULT.PACKET.LENGTH)
                                                                 (\\ENQUEUE IQ PACKET)
                                                                 (|add| LEN 1))
         (|replace| NDBIQLENGTH |of| NDB |with| LEN)

          (* * |This| |process| |will| |eventually| |be| |replaced| |by| |interrupts|)

         (|replace| NDBWATCHER |of| NDB |with| (ADD.PROCESS (LIST (FUNCTION \\DLRS232C.WATCHER)
                                                               (KWOTE NDB))
                                                 'RESTARTABLE
                                                 'SYSTEM
                                                 'AFTEREXIT
                                                 'DELETE))
        NDB)))
```

(**DLRS232C.SEND.PACKET**

```
  (LAMBDA (NDB PACKET EVENT)                                    ; Edited 22-Dec-86 14:00 by lmm
    (PROG ((DROPIT (AND \\RS232C.LIGHTNING (EQ 0 (RAND 0 \\RS232C.LIGHTNING))))
           IOCB BUFLENGTH)
          (UNINTERRUPTABLY
              (|replace| EPTRANSMITTING |of| PACKET |with| T)
              (COND
                 (DROPIT                                        ; Fake transmission
                        (\\ENQUEUE (|fetch| NDBTQ |of| NDB)
                               PACKET)
                        (|replace| EPNETWORK |of| PACKET |with| NIL))
                 (T (SETQ IOCB (\\DLRS232C.GET.IOCB 'OUTPUT))
                    (CL:ASSERT (NOT (NULL IOCB)))
                    (|replace| EPNETWORK |of| PACKET |with| IOCB)
                    (SETQ BUFLENGTH (|fetch| (RS232C.ENCAPSULATION RS232C.LENGTH) |of| PACKET))
                    (\\TEMPLOCKPAGES PACKET (COND
                                               ((IGEQ BUFLENGTH (CONSTANT (UNFOLD \\MIN2PAGEBUFLENGTH
                                                                                  BYTESPERWORD)))
                                                2)
                                               (T 1)))
                    (LET ((CLOCK (CREATECELL \\FIXP)))
                         (\\CLOCK0 CLOCK)
                         (|replace| EPTIMESTAMP |of| PACKET |with| CLOCK))
                                                                ; Put on microcode queue
                    (\\ENQUEUE (|fetch| NDBTQ |of| NDB)
                           PACKET)
                    (SELECTC \\MACHINETYPE
                         (\\DANDELION (|replace| (DLRS232C.IOCB SYNCH.EVENT) |of| IOCB |with| EVENT)
                                      (\\DLRS232C.QUEUE.OUTPUT.IOCB IOCB (|fetch| (RS232C.ENCAPSULATION
                                                                                      RS232C.PACKET.BASE)
                                                                                 |of| PACKET)
                                                                         BUFLENGTH))
                         (\\DAYBREAK (|replace| (|Dove.RS232IOCB| |rsLispSynchEvent|) |of| IOCB |with| EVENT)
                                     (\\DLRS232C.QUEUE.OUTPUT.IOCB IOCB (|fetch| (RS232C.ENCAPSULATION
                                                                                     RS232C.PACKET.BASE)
                                                                                |of| PACKET)
                                                                        BUFLENGTH))
                         (\\NOMACHINETYPE))
                    T))                                         ; Put on driver's queue to pick up after microcode finishes with it

              )
          (RETURN (AND IOCB T)))))
```

(**RS232C.FORCEOUTPUT**

```
  (LAMBDA (STREAM WAITFORFINISH)                                ; Edited 29-May-87 15:27 by Snow
```

```
    (COND
       ((OPENP STREAM 'OUTPUT)
        (LET ((PACKET (|fetch| (STREAM CBUFPTR) |of| STREAM))
              (EVENT (|fetch| (RS232C.STREAM EVENT) |of| STREAM)))
             (COND
                ((|type?| ETHERPACKET PACKET)
                 (|replace| (RS232C.ENCAPSULATION RS232C.LENGTH) |of| PACKET
                    |with| (IDIFFERENCE (|fetch| COFFSET |of| STREAM)
                                (CONSTANT (UNFOLD (IPLUS (INDEXF (FETCH (RS232C.ENCAPSULATION RS232C.DATA)
                                                                  OF T))
                                                         (INDEXF (FETCH EPENCAPSULATION OF T)))
                                          BYTESPERWORD))))
                 (\\RS232C.TRACE.PACKET PACKET 'OUTPUT)
                 (|replace| COFFSET |of| STREAM |with| (|replace| CBUFSIZE |of| STREAM
                                                           |with| (|replace| CBUFMAXSIZE |of| STREAM |with| 0)))
                 (|replace| CBUFPTR |of| STREAM |with| NIL)
                 (\\DLRS232C.SEND.PACKET \\DLRS232C.LOCAL.NDB PACKET (AND WAITFORFINISH EVENT))
                 (COND
                    (WAITFORFINISH (|while| (|fetch| EPTRANSMITTING |of| PACKET) |do| (AWAIT.EVENT EVENT)))
                    (T (BLOCK)))))))))
```

(\\RS232C.GETNEXTBUFFER

```
(LAMBDA (STREAM WHATFOR NOERRORFLG)                                    (* |ejs:| "24-Dec-85 14:05")
   (LET
    ((QUEUE (|fetch| (RS232C.STREAM PACKET.QUEUE) |of| STREAM))
     (EVENT (|ffetch| (RS232C.STREAM EVENT) |of| STREAM))
     (OLDPACKET (|ffetch| (STREAM CBUFPTR) |of| STREAM))
     (LASTBUFFER (|ffetch| (RS232C.STREAM LASTBUFFER) |of| STREAM))
     NEXTPACKET)
    (SELECTQ WHATFOR
        (READ (COND
                 ((|ffetch| (RS232C.STREAM DID.BACKFILEPTR) |of| STREAM)
                  (UNINTERRUPTABLY
                      (|freplace| (RS232C.STREAM DID.BACKFILEPTR) |of| STREAM |with| NIL)
                      (|swap| (|ffetch| CBUFPTR |of| STREAM)
                              (|ffetch| (RS232C.STREAM LASTBUFFER) |of| STREAM))
                      (|swap| (|ffetch| CBUFSIZE |of| STREAM)
                              (|ffetch| (RS232C.STREAM LASTBUFFER.CBUFSIZE) |of| STREAM))
                      (|freplace| COFFSET |of| STREAM |with| (UNFOLD (CONSTANT (IPLUS (INDEXF (|fetch| (
                                                                                                RS232C.ENCAPSULATION
                                                                                                   RS232C.DATA)
                                                                                                |of| T))
                                                                                       (INDEXF (|fetch| EPENCAPSULATION
                                                                                                |of| T))))
                                                                                BYTESPERWORD))
                      T))
                 (T (COND
                       (OLDPACKET (COND
                                     (LASTBUFFER (\\RELEASE.ETHERPACKET LASTBUFFER)))
                                  (|freplace| (RS232C.STREAM LASTBUFFER) |of| STREAM |with| OLDPACKET)
                                  (|freplace| (RS232C.STREAM LASTBUFFER.CBUFSIZE) |of| STREAM
                                     |with| (|ffetch| CBUFSIZE |of| STREAM))
                                  (|freplace| CBUFPTR |of| STREAM |with| NIL)
                                  (|freplace| COFFSET |of| STREAM |with| (|freplace| CBUFSIZE |of| STREAM |with| 0))))
                    (|until| (SETQ NEXTPACKET (\\DEQUEUE QUEUE)) |do| (AWAIT.EVENT EVENT)
                       |finally| (|add| (|fetch| (RS232C.STREAM QUEUE.LENGTH) |of| STREAM)
                                        -1)
                                 (\\RS232C.TRACE.PACKET NEXTPACKET 'INPUT)
                                 (|freplace| CBUFSIZE |of| STREAM
                                    |with| (IPLUS (|fetch| (RS232C.ENCAPSULATION RS232C.LENGTH) |of| NEXTPACKET)
                                                  (|freplace| COFFSET |of| STREAM
                                                     |with| (UNFOLD (CONSTANT (IPLUS (INDEXF (|fetch| (
                                                                                                RS232C.ENCAPSULATION
                                                                                                   RS232C.DATA)
                                                                                                |of| T))
                                                                                       (INDEXF (|fetch| EPENCAPSULATION
                                                                                                |of| T))))
                                                                                BYTESPERWORD))))
                                 (|freplace| CBUFPTR |of| STREAM |with| NEXTPACKET))))
              T)
        (WRITE (COND
                  ((NEQ (|fetch| COFFSET |of| STREAM)
                        (CONSTANT (UNFOLD (IPLUS (INDEXF (|fetch| (RS232C.ENCAPSULATION RS232C.DATA) |of| T))
                                                 (INDEXF (|fetch| EPENCAPSULATION |of| T)))
                                          BYTESPERWORD)))
                   (\\RS232C.FORCEOUTPUT STREAM))
               (|freplace| CBUFSIZE |of| STREAM |with| (|freplace| CBUFMAXSIZE |of| STREAM |with|
                                                                                  \\RS232C.OUTPUT.PACKET.LENGTH
                                                                                  ))
               (|freplace| COFFSET |of| STREAM |with| (CONSTANT (UNFOLD (IPLUS (INDEXF (|fetch| (RS232C.ENCAPSULATION
                                                                                                   RS232C.DATA)
                                                                                                |of| T))
                                                                                       (INDEXF (|fetch| EPENCAPSULATION
                                                                                                |of| T)))
                                                                                BYTESPERWORD)))
               (|freplace| CBUFPTR |of| STREAM |with| (SETQ NEXTPACKET (\\ALLOCATE.ETHERPACKET)))
```

```
                (|freplace| EPREQUEUE |of| NEXTPACKET |with| 'FREE)
                T)
          (ERROR "Illegal stream operation " WHATFOR)))))
```

(\\RS232C.TRACE.PACKET

```
(LAMBDA (PACKET FORWHAT)                                    ; Edited 5-Nov-87 11:54 by FS
  (COND
     ((TYPENAMEP PACKET 'ETHERPACKET)
      (SELECTQ RS232C.TRACEFLG
           (T (|printout| RS232C.TRACEFILE T FORWHAT ": ")
              (|bind| CH |for| CHINDEX |from| (CONSTANT (TIMES BYTESPERWORD (IPLUS (INDEXF (|fetch| (
                                                                                    RS232C.ENCAPSULATION
                                                                                       RS232C.DATA)
                                                                                 |of| T))
                                                                           (INDEXF (|fetch| EPENCAPSULATION
                                                                                 |of| T)))))
                 |to| (SUB1 (IPLUS (|fetch| (RS232C.ENCAPSULATION RS232C.LENGTH) |of| PACKET)
                                   (CONSTANT (TIMES BYTESPERWORD (IPLUS (INDEXF (|fetch| (RS232C.ENCAPSULATION
                                                                                    RS232C.DATA)
                                                                              |of| T))
                                                                        (INDEXF (|fetch| EPENCAPSULATION
                                                                              |of| T)))))))
                 |do| (SETQ CH (\\GETBASEBYTE PACKET CHINDEX))
                      (COND
                         ((< (LOGAND CH 127)
                             (CHARCODE SPACE))
                          (CL:FORMAT RS232C.TRACEFILE "[~o]" CH))
                         (T (CL:WRITE-CHAR (CL:INT-CHAR CH)
                                   RS232C.TRACEFILE)))))
           (PEEK (PRIN1 (SELECTQ FORWHAT
                            (INPUT "+")
                            "!")
                        RS232C.TRACEFILE))
           NIL)))))

)

(PUTPROPS RS232CNETWORK COPYRIGHT ("Xerox Corporation" 1988))
```

FUNCTION INDEX

CONSTANT INDEX

VARIABLE INDEX

RECORD INDEX