```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 49
IMPSUB.PLR;1    PAGE 1  IMP Subroutines

                                .stitl IMP Subroutines

.comnt |
RELOAD
Routine to force a reload by smashing all checksums.
and then jumping to the restart code.
Call with JSB R4,RELOAD. never returns.
|

                        Page LCode              ;called from all pages
                    .
                    reload:
1308 7078 00B4      setmap M0,mappkg            ;jump to code on the package page
130C 3078 FC00
1310 4008 4162      jmp relpkg

FC00 0400               Page PkgCode

                    relpkg:
4162 7078 02FA      lda r7,hotlim               ;force reload; smash all cksums.
4166 1207           addm r0,(-r7)               ;smash local checksum
4168 7078 00AE      lda r7,mapcom
416C 3078 FC04      sta r7,%map2

4170 4898           lda r1,=nmseg               ;how much memory * 16
4172 4838 7E00      lda r3,=nmseg* D8* H200- H200   ;last map
                    repeat
4176 4828 0020        lda r2,= H20              ;16 per word
417A 5059 809E        lda r5,m2#memseg(-r1)     ;existing memories.
                      repeat                    ;for each page
417E 575A 0EB6          if r5 .bit. bittab(-r2) ;page exists.
4182 9A0D
4184 3038 FC02            sta r3,%map1
4188 7078 60B8            lda r7,m1#tlimit      ;last cksummed word.
418C 8008                 ifnot quit            ;page truly there.
418E 49F2                   sub r7,=words       ;addr of last word in cksum.
4190 9B06                   if nminus & r7 > =m1#tlimit     ;;m2
4192 4E78 60B8
4196 8C03
4198 3207                     addm r0,(r7)      ;smash if within cksum region.
419A 8001                     ifnot quit
                              endif
                            endif
                          endif
419C 4938 0200          sub r3,= H200           ;next page.
41A0 88EF             until loop
                    endrepeat
41A2 4E90           until r1 = =0               ;done all mem segments.
41A4 81E9
                    endrepeat

41A6 4008 0A1C      tr wst                      ;now restart system.
```

```
.comnt |
FLUSH and FLUSHB
Routines to add buffers to the system free list.
FLUSHB first restores map 2 to MAPV2.
R2 enters with address of buffer CHAIN word.
Inline argument is which ownership bit to clear; if other bits are still
on the buffer is still owned elsewhere, so nothing more happens.
Otherwise, the appropriate buffer accounting counter is incremented and
NF adjusted as needed.
|

                        Page LCode

1314 4868 02F8  oflush: lda sp,=lstack
1318 900D               br flush

131A 4868 02F8  oflshb: lda sp,=lstack

131E 1076       routine flushb, local r1, inline r1, arg r2
1320 1016
1322 6017
1324 307E 0002

1328 7078 00D2  lda r7,mapv2    ;this entry restores V2 page
132C 3078 FC04  sta r7,%map2

1330 9006       entry flush, local r1, inline r1, arg r2
1332 1076
1334 1016
1336 6017
1338 307E 0002

133C 7078 00D2  lda r7,mapv2                    ;* check maps
1340 7678 808C  if r7 <> m2#slfptr              ;*
1344 9103
                 ; Trap 1143,<:map error in flush - page 49>
1346 3078 FC04    sta r7,%map2
                endif
134A 4E28 92C0  if r2 < =chain } r2 > junk
134E 9204
1350 7628 63CA
1354 8C03
1356 E2E3         Trap 1343,<;tried to flush non-buffer - page 49>
1358 9032       else
135A 771A 06A0    ifnot r1 .bit. where-chain(r2)        ;bad owner
135E 8A05
1360 707A 06A0      lda r7,where-chain(r2)
1364 E2E5           Trap 1345,<:tried to flush non-owned buffer - page 49>
1366 902B         else                          ;good owner:
1368 7078 A3C8      lock r7,nf                  ;adjust counters first
136C 9AFE
136E 751A 06A0      eor r1,where-chain(r2)      ;remove owner bit
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 51
IMPSUB.PLR;1    PAGE 3  IMP Subroutines

1372 4F18 FFC1      if r1 .bit. #-1?whctr                ;more owners
1376 9A06
1378 301A 06A0        sta r1,where-chain(r2)     ;just restore owners
137C 3078 A3C8        unlock r7,nf
1380 901E           else
1382 4A18 63D8        add r1,=cntrs
1386 1016             push r1             ;points to proper counter
1388 4891             lda r1,=1
138A 2196             subm r1,@(sp)+      ;decrement source count
138C 9B02             ifnot minus         ;returned one to pool
138E 4AF1               add r7,=1         ;so bump NF
                      endif
1390 3012             sta r1,(r2)         ;becomes queue end
1392 4890             lda r1,=0
1394 301A 06A0        sta r1,where-chain(r2)     ;no owner or count
1398 3078 A3C8        unlock r7,nf        ;done with counters
139C 707E 0002        lda r7,2(sp)        ;* debugging aid: caller
                ;     sta r7,2(sp)
13A0 307A 09F0        sta r7,flushd-chain(r2)    ;buffer is active
13A4 7078 A3C6        lock r7,freend      ;free list tail
13A8 9AFE
13AA 8906             if odd              ;list was broken
13AC 7078 A3C4          lock r7,free      ;or empty
13B0 9AFE
13B2 4878 A3C4          lda r7,=free      ;now free will be unlocked
                      endif
13B6 3027             sta r2,(r7)         ;;free  ;enqueue buffer
13B8 3028 A3C6        unlock r2,freend
                    endif
                  endif
                endif

13BC 6016       endroutine flushb
13BE 6006
```

```
.comnt | FREGET, FGET
Routine that obtains a buffer from the free list.
Inline argument is what buffer counter to charge.
Returns with unpacked (XIM) address in r1, CHAIN
word address in R2 Map 2 is set up to point to the buffer.
FAIL return if free list empty or no count available.
  Entry FGET with WHERE counter in r3 already
|

13C0 1076       routine fget, arg r3, result r1-r2/m2, local r3
13C2 1036
13C4 9006       entry freget, inline r3, result r1-r2, local r3, result m2
13C6 1076
13C8 1036
13CA 6037
13CC 307E 0002
13D0 7078 00D2    lda r7,mapv2          ;always set map 2
13D4 3078 FC04    sta r7,%map2
                  begin ckcount
13D8 4813           lda r1,r3
13DA 4B18 003E      and r1,=whctr               ;just counter field
13DE 7028 A3C8      lock r2,nf            ;lock counters
13E2 9AFE
13E4 7079 63D8      lda r7,cntrs(r1)      ;get source count
13E8 9B06           ifnot minus           ;must use slush pool
13EA 7628 63D0        if r2 <= minf       ;no more in pool
13EE 9C02
13F0 9036               leave ckcount     ;fix lock and fail return
                      endif
13F2 49A1             sub r2,=1           ;remove one from pool
                    endif
13F4 4AF1           add r7,=1
13F6 3079 63D8      sta r7,cntrs(r1)      ;modify counter
13FA 3028 A3C8      unlock r2,nf          ;done NF. get buffer from FREE

                  ;(OVER)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 53
IMPSUB.PLR;1    PAGE 5  IMP Subroutines

                ;FREGET continued

13FE 7028 A3C4      lock r2,free                ;now get a buffer
1402 9AFE
1404 991B           ifnot odd           ;non-empty free list
1406 707A 06A0        lda r7,where-chain(r2)
140A 8A17             if zero           ;no stray owners
140C 303A 06A0          sta r3,where-chain(r2)  ;new count, owner
1410 7032               lda r3,(r2)             ;next free buffer
1412 8909               if odd          ;just emptied free list
1414 7078 A3C6            lock r7,freend        ;must modify endpointer
1418 9AFE
141A 7032                 lda r3,(r2)   ;in case new FLUSH happened.
141C 8902                 if odd                ;still empty
141E 48F1                   lda r7,=1   ;so mark end too
                          endif
1420 3078 A3C6            unlock r7,freend
                        endif
1424 3038 A3C4          unlock r3,free
1428 707E 0002          lda r7,2(sp)    ;get caller
142C 307A 09F0          sta r7,flushd-chain(r2) ;save caller in flushd
1430 4078 153E          call qunpack    ;set up r1, m2
1434 6036               return
1436 6006
                      endif
1438 E2E1             Trap 1341,<;free list error, non-zero where - page 52>
                    endif
143A 48F1           lda r7,=1
143C 3078 A3C4      unlock r7,free      ;;odd   ;bye bye free list
1440 7028 A3C6      lock r2,freend              ;now lock free list's tail
1444 9AFE
1446 9902             if even                   ;there is no freelist tail)
1448 E20A               Trap 1012,<;threw away free list tail - page 52>
                      endif
144A 3078 A3C6      unlock r7,freend    ;;odd   ;tail is now indeed a tail
144E 7028 A3C8      lock r2,nf                  ;undo counts
1452 9AFE
1454 3179 63D8      subm r7,cntrs(r1)
1458 9B02    .      ifnot minus                 ;using slush
145A 4AA1             add r2,=1         ;adjust shared pool
                    endif
                  end ckcount
145C 3028 A3C8    unlock r2,nf          ;done with counts
1460 6036         fail return
1462 6076
1464 4FF0
1466 4007
                endroutine freget       ;;fget
```

```
.comnt |
WHEORB and WHEORM
Routines to diddle (via EOR) where ownership or counter bits
WHEORB first fixes map 2
R2 enters with CHAIN word address. Bits to change are inline.
|

1468 4868 02F8  owheob: lda sp,=lstack

146C 1076       routine wheorb, local r1, inline r1, arg r2
146E 1016
1470 6017
1472 307E 0002

1476 7078 00D2  lda r7,mapv2
147A 3078 FC04  sta r7,%map2            ;fix V2 page map first

147E 9006       entry wheorm, local r1, inline r1, arg r2
1480 1076
1482 1016
1484 6017
1486 307E 0002

148A 7078 00D2  lda r7,mapv2            ;* map check
148E 7678 808C  if r7 <> m2#slfptr      ;*
1492 9103
                 ; Trap 1144,<;map error in nwheom - page 53>
1494 3078 FC04    sta r7,%map2          ;*
                endif                   ;*
1498 7078 A3C8  lock r7,nf              ;lock counters
149C 9AFE
149E 351A 06A0    eorm r1,where-chain(r2)
14A2 3078 A3C8  unlock r7,nf            ;done counters

14A6 6016       endroutine wheorb
14A8 6006
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 55
IMPSUB.PLR;1    PAGE 7  IMP Subroutines

.comnt |
TTSKPUT, TSKPUT
Queues a packet for TASK and pokes the TASK PID
TTSKPUT - timestamp the buffer first in IT word
TSKPUT  - just queue it and poke TASK
R2 has packet pointer (CHAIN word).
|

14AA 1076       routine ttskput, arg r2

14AC 1016         save r1                       ;need this for UNPACK
14AE 4078 1512    call unpack                   ;point to buffer for timestamp
14B2 70F8 62E8    set it(r1) = @clock           ;timestamp buffer
14B6 3079 0096
14BA 7078 00D2    setmap m2,mapv2               ;fix map2
14BE 3078 FC04
14C2 6016         restore r1

14C4 9002       entry tskput, arg r2
14C6 1076

14C8 48F1         set (r2) = #1                 ;last buffer on queue
14CA 3072
14CC 7078 A3B6    lock ltq                      ;lock task queue.
14D0 9AFE
14D2 30A8 63BA      sta r2,@etq                 ;enqueue buffer
14D6 3028 63BA      sta r2,etq
14DA 3008 A3B6    unlock ltq
14DE 4878 0040    set @pid = =task              ;poke task.
14E2 30F8 00AC

14E6 6006       endroutine ttskput/tskput

                ;POKEM - set rare flag and poke
                ;TRYMOD - poke modem output
                ;r4 enters with modem id block

14E8 1076       routine pokem, arg r4
14EA 307C 002A    set i2mpok(r4) = r7   ;$debug ;set rare events flag
14EE 9002        entry trymod
14F0 1076
14F2 787C 000E    ldab r7,motpid(r4)    ;output pid.
14F6 30F8 00AC    sta r7,@pid
14FA 6006       endroutine trymod
```

```
.comnt |
UNPCKC and UNPACK
Unpack a packed buffer address into R1 (XIM) and M2.
R2 enters with packet CHAIN word address.
UNPCKC has inline ownership bit(s) to check; FAIL return if not owned.
|

14FC 4868 02F8  ounpck: lda sp,=lstack

1500 6017       routine unpckc, inline r1, arg r2, result r1, result m2
1502 1076

1504 771A 06A0  tst r1,where-chain(r2)          ;this owner okay?
1508 8A04       if zero                         ;no}
150A 6076         fail return                   ;do no more
150C 4FF0
150E 4007
                endif

1510 9002       entry unpack, arg r2, result r1, result m2
1512 1076

1514 7018 00D2  lda r1,mapv2                    ;* check maps first
1518 7618 808C  if r1 <> m2#slfptr              ;*
151C 9103
                 ; Trap 1147,<;* map error in unpack - page 55>
151E 3018 FC04   sta r1,%map2                   ;*
                endif                           ;*
1522 701A FCB0  lda r1,point-chain(r2)          ;packed pointer
1526 3018 FC04  sta r1,%map2                    ;set map2
152A A294       sll r1,4                        ;to get page offset
152C 4B18 1FFE  and r1,=packm
1530 4C18 8000  ior r1,=m2                      ;address returned XIM

1534 6006       endroutine unpckc

.comnt |
CUNPACK
like UNPACK, but first sets CHAIN word odd, and loads CHAN into R3.
|

1536 1076       routine cunpack, arg r2, result r1-r3, result m2

1538 703A 0350    lda r3,chan-chain(r2)
153C 9002       entry qunpack, arg r2, result r1-r2, result m2
153E 1076
1540 48F1         lda r7,=1
1542 3072         sta r7,(r2)
1544 4078 1512    call unpack

1548 6006       endroutine cunpack
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 57
IMPSUB.PLR;1    PAGE 9  IMP Subroutines

.comnt |
DEQUE
routine to remove a buffer from a queue.
inline argument is ownership bit to check, R1 enters with queue address.
Returns after an UNPACK, with R1 unpacked buffer pointer (XIM),
R2 buffer CHAIN word address, and R3 CHAN for this buffer.
M2 is set up to access the buffer.
Fail return if buffer error or queue empty
|

154A 6037       routine deque, inline r3, arg r1, result r1-r3, result m2
154C 1076

154E 7028 00D2  lda r2,mapv2                    ;* check map setting
1552 7628 808C  if r2 <> m2#slfptr              ;*
1556 9103
                 ; Trap 1146,<;* map error in deque - page 56>
1558 3028 FC04    sta r2,%map2                  ;*
                endif                           ;*
155C 7021       lda r2,(r1)                     ;get queue head
155E 990F       ifnot odd                       ;there's a buffer
1560 773A 06A0    tst r3,where-chain(r2)        ;owned?
1564 9A09         ifnot zero                    ;yes
1566 7072           lda r7,(r2)                 ;next buffer
1568 8903           if odd                      ;no more
156A 3019 0002        sta r1,words(r1)          ;point endpointer at start
                    endif
156E 3071           sta r7,(r1)                 ;new queue head
1570 4078 1536      call cunpack                ;and set up R1, R3, M2
1574 6006           return
                  endif
1576 E2A2         Trap 1242,<;buffer ownership error - page 56>
1578 48F1         lda r7,=1
157A 3071         sta r7,(r1)                   ;empty the queue
                endif
157C 7619 0002  if r1 <> 2(r1)                  ;queue empty check
1580 9104
1582 E2F0         Trap 1360,<;fixed half-empty queue - page 56>
1584 3019 0002    sta r1,2(r1)                  ;and fix it
                endif
1588 6076       fail return                     ;got no buffer
158A 4FF0
158C 4007

                endroutine deque
```

```
.comnt  |
sioin - Start I/O input to buffer whose chain pointer is in r2.

        This routine starts input while checking for a subset of
possible interface failures, in an attempt to prevent a buffer from
being read into the wrong place.  Will not detect failures
in the high-order 7 bits of a 20-bit address, nor in the low
4, but will check the middle 9.

        call sioin with:
        lda r2,<chain word>
        lda r5,<io blk>
        call sioin,buflen       ;assumes read into first word of buffer.

|

158E 1076       routine sioin,local r1,arg r2,local r3,inline r3,arg r5
1590 1016
1592 1036
1594 6037
1596 307E 0004
159A 49B2         sub r3,=hrdoff        ;compensate for hardware offset.
159C 701A FCB0    lda r1,point-chain(r2) ;buffer address.
15A0 4078 0ED6    call xsioin           ;start transfer.
15A4 6036       endroutine sioin
15A6 6016
15A8 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 59
IMPSUB.PLR;1    PAGE 11          IMP Subroutines

.comnt |
hsioin  - Start host input, to buffer whose chain is in r2.
fhsioin - Start host input for fake host

HSIOIN starts host input to specified place. Special handlers
are called via HITABLE for actual hardware (SIOIN) or for
fake hosts of all kinds (i.e., VDH, fakes, TIP); the latter
routine is FHSIOIN. See HITABLE in HACCON.PLR.
|

.comnt |
        inline R1 = start of data
        inline R3 = length of data
        R2 = chain pointer
        R4 = host parameter block
        R5 = device address
|

15AA 1076       routine hsioin,local r1,inline r1,local r3,inline r3,arg r2
15AC 1016
15AE 1036
15B0 6017
15B2 6037
15B4 307E 0004
                ;arg r4-r5
15B8 302C 0032    sta r2,hibf(r4)               ;save buffer being filled
15BC 4939 0002    sub r3,=hrdoff(r1)            ;length of data
15C0 A694         srl r1,4
15C2 721A FCB0    add r1,point-chain(r2)        ;start of data in block
15C6 707C 0002    lda r7,hostyp(r4)             ;for this host type
15CA 40FF 12BE    call @hitable(r7)             ;do host input
15CE 6036       endroutine
15D0 6016
15D2 6006

.comnt |
        R1 = packed start address
        R3 = length of input
        R5 = device
|
15D4 1076       routine fhsioin, arg r1, arg r3, arg r5

15D6 301D 0002    sta r1,starti(r5)             ;start input here
15DA A294         sll r1,4                      ;unpack and form end addr
15DC 4B18 1FFE    and r1,=packm
15E0 4A31         add r3,r1
15E2 303D 0004    sta r3,endi(r5)               ;end of input
15E6 4C18 8000    ior r1,=m2
15EA 301D 0010    sta r1,fakesi(r5)             ;JAM buffer pointer
15EE 4878 2000    lda r7,=hbusy
15F2 347D 0006    iorm r7,statih(r5)            ;make interface busy
15F6 787D 000F    ldab r7,bhpid(r5)
15FA 30F8 00AC    sta r7,@pid                   ;and wake up FAKE

15FE 6006       endroutine fhsioin
```

```
.comnt |
division routine
r1 contains divisor r2 contains dividend
returns with quotient in r1, remainder in r2
|
1600 1076       routine divide,arg r1-r2,result r1-r2,uses r3
1602 48F1          lda r7,=1     ;initialize counter-when 0, stop
1604 48B0          lda r3,=0     ;initialize remainder accumulaaor
                   repeat
1606 A191             rla r1,1   ;do next binary place
1608 A1B1             rla r3,1
160A 4E32             if r3 >= r2
160C 9202
160E 4932                sub r3,r2
                      endif
1610 A2F1             sll r7,1   ;advance "counter"
1612 8AFA           until zero
                   endrepeat
1614 A191          rla r1,1
1616 4823          lda r2,r3
1618 6006       endroutine divide
```

```
.INSERT "IMPDDT"
    .INSRT IMPDDT
```

```
                        .stitle IMP's DDT code


     0010       .radix  H10             ;this file in hex

                        page Vars
7280            ccled:  .blkw 6                 ;tty's leader (default to DDT here)
728C            clled:  .blkw 6                 ;leader for ":" messages
7298            ddtled: .blkw 6                 ;where ddt sends its responses
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 63
IMPDDT.PLR;1    PAGE 2  IMP's DDT code

     0000                .parity 0               ;no parity bits please

                    ;IMP DDT wakeup

                            Page LCode
                    jjddt:
161A 4078 0FBC         call finddt
161E 4008 4C46         tr ddtwak
FC00 0200                   Page DDTCode
                    ddtwak:
4C46 4078 42B8         call ddtpol
4C4A 7018 A272         lda r1,ddpoke+foo           ;PIDs to poke?
4C4E 9A08              ifnot zero                  ;yup
4C50 48F4                 poke bltpid
4C52 30F8 00AC
4C56 4818 0012            lda r1,=ddtpid           ;and restart DDT
4C5A 4078 1750            call rstart
                       endif
4C5E 4008 1072      tr loopmv


                    ;imp TTY wakeup

                            Page LCode
                    jjtty:
1622 4078 0FBC         call finddt
1626 4008 4C62         tr ttywak
FC00 0200              Page DDTCode
                       ttywak:
4C62 4078 4134         call ttypol
4C66 7078 00A0         lda r7,stime                ;update time of IMP's polling
4C6A 4A78 0078         add r7,=bltrat
4C6E 3078 6266         sta r7,poltim
4C72 7078 A268         if m3#dxtflag               ;pass chars to DDT?
4C76 9A05
4C78 4878 0012            set @pid = =ddtpid       ;if so, poke DDT
4C7C 30F8 00AC
                       endif
4C80 7018 A270         lda r1,ttpoke+foo
4C84 9A05              ifnot zero                  ;process to restart
4C86 4818 0014            lda r1,=ttypid           ;TTY PID level
4C8A 4078 1750            call rstart
                       endif
4C8E 90E8           tr loopmv
```

```
                ;crosspatch to IMP. dispatched from  C with
                ;  host in r4 (prefix)  imp in r3 (ddaccum+sum)
     EF00               implnk= 0357_ H8          ;nominal link to use
4C90 1076          routine cntlc, uses r1, arg r3-r5
4C92 4078 479C        call iddtout,'C              ;echo a "C" for crosspatching
4C96 0043
4C98 4875             ifnot r5     ::pfx           ;two args
4C9A 8A0E
4C9C 1056                 SAVE R5
4C9E 7058 62E2            LDA R5,MINE
4CA2 4ED1               CMP R5,= D1
4CA4 9106                 BE ISTR
4CA6 4078 44CA                    CALL RUBOUT
4CAA E10E                 Trap 416,<;Invld attmpt to use Xpatch - page 63>
4CAC 6056                         RESTORE R5
4CAE 6006                         RETURN
                ISTR:
4CB0 6056                 RESTORE R5
4CB2 4848 00FD           Lda r4,# HFD             ;default ddt fake
                      endif
4CB6 4C48 8000        ior r4,#getpri               ;always priority
4CBA 3048 7284        sta r4,ccled+hstl            ;set up dest host
4CBE 3038 7286        sta r3,ccled+dstl            ;put into leader
4CC2 4078 44D4        call argreset                ;reset arguments
4CC6 90F4          endroutine cntlc

FC00 0200             $dopatch ddtdisp+< 03*words>, DDTCode
43A6 4C90                cntlc
FC00 0200             page DDTCode
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 65
IMPDDT.PLR;1    PAGE 4  IMP's DDT code

                ;IMP's Control-O: turn override on/off
4CC8 1076       routine cntlo, uses r1
4CCA 7018 62E2     LDA R1,MINE
4CCE 4E91            CMP R1,= D1
4CD0 9107               BE MSTR
4CD2 7018 4C44     LDA R1,OVRTAB+2
4CD6 4E91            CMP R1,= D1
4CD8 9103               BE MSTR
                NOTME:
4CDA E10F               Trap 417,<;Invld attmpt to enable ovride - page 64>
4CDC 6006               RETURN
                MSTR:
4CDE 7078 00B6     setmap m2,mapfak             ;need fake page
4CE2 3078 FC04
4CE6 7018 729E     lda r1,ddtled+dstl
4CEA 7878 729D     ldab r7,ddtled+hstl+1        ;check for valid NCC site
4CEE 4E78 00FC     ifnot r7 >= #0FC & r1 = mine } byte r1 = m2#nccimp
4CF2 9207
4CF4 7618 62E2
4CF8 910C
4CFA 7E18 80C6
4CFE 9109
4D00 A2F8             sll r7,8                  ;combine host,imp
4D02 4C17             ior r1,r7
4D04 48F4             lda r7,=lovrtab
                      repeat
4D06 8802                if loop
4D08 90EA                   return
                         endif
4D0A 561F 4C42         until r1 = ovrtab(-r7)   ;got a okay source
4D0E 81FC
                      endrepeat
                   endif
4D10 4078 479C     call iddtout, '0      ;type the char
4D14 004F
4D16 4878 0010     lda r7,#ovrbit        ;now flip switch
4D1A 4818 0046     lda r1,='F            ; turn off?
4D1E 3578 62AE     eorm r7,anom
4D22 7778 62AE     tst r7,anom
4D26 9A02          ifnot zero            ;nope, now is on
4D28 4A98             add r1,='N-'F      ;change what we type
                   endif
4D2A 4078 4798     call ddtout
4D2E 4078 4820     call itextout, %3space
4D32 465E
4D34 7078 00D2     setmap m2,mapv2              ;don't need fakes no mo'
4D38 3078 FC04
4D3C 90D0             RETURN
                endroutine cntlo

FC00 0200          $dopatch ddtdisp+< 017*words>,DDTCode
43BE 4CC8             cntlo
FC00 0200          Page DDTCode
```

```
                .stitle OPHELP: IMP Operator commands

                ;some routines to facilitate ophelps

.comnt |
NEWARG to put something into the DDT input buffer and
 prefix with whatever already in buffer
 PUTARG to just put something into buffer.  |
4D3E 1076       routine newarg, arg r1
4D40 1016          save r1                      ;save this
4D42 4078 479C     call iddtout, ascicomma      ;type a comma
4D46 002C
4D48 7078 A2B4     lda r7,ddaccum+foo           ;get accumulated argument
4D4C 7278 A2B6     add r7,sum+foo               ;add whatever
4D50 3078 62BA     sta r7,prefix                ;to make prefix
4D54 3008 62B0     set pfx
4D58 6016          pop r1                       ;get number back
4D5A 9002       entry putarg
4D5C 1076
4D5E 3018 62B4     sta r1,ddaccum               ;save new argument
4D62 3008 62B2     set something                ;and say it exists
4D66 4078 4830     call numout
4D6A 7078 A2BC     lda r7,locopen+foo           ;stop unexpected stores
4D6E 6006       endroutine putarg       ;;newarg

.comnt |
TLOOKUP to get char from tty and check if match char
 in table @inline 1, then return corresponding entry
 from inline 2  |
4D70 6047       routine tlookup, inline r4, inline r5, result r3, uses r1/r3-r5
4D72 6057
4D74 1076
4D76 4078 4D98     call opget                   ;get char to match
4D7A 4834          lda r3,r4                    ;save start
                   repeat
4D7C 6874             ldab r7,(r4)+             ;get char from table
4D7E 8A06             if zero                   ;no chars left
4D80 7078 A2BC           lda r7,locopen+foo     ;close any open location
4D84 6076                fail return            ;lose
4D86 4FF0
4D88 4007
                      endif
4D8A 4E71             until r7 = r1             ;got right one
4D8C 81F8
                   endrepeat
4D8E 4943          sub r4,r3                    ;how far we've gone
4D90 A2C1          sll r4,1                     ;get dispatch/value/data
4D92 4A45          add r4,r5                    ; out of table in r5
4D94 5034          lda r3,(-r4)                 ;get proper argument
4D96 6006       endroutine tlookup
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 67
IMPDDT.PLR;1    PAGE 6  OPHELP: IMP Operator commands

.comnt |A character get routine that ignores semis from
TTY/DDT fakes. returns r1    |

4D98 1076       routine opget, result r1
                   repeat
4D9A 4078 47D0        call ddtget
4D9E 4E18 003B      while r1 = #semicolon
4DA2 8104
4DA4 4078 4798        call ddtout               ;send back
4DA8 90F9          endrepeat
4DAA 6006       endroutine opget
```

```
.comnt |        don't want this this time
;get message generator parameters
routine opmessgen
   call opget
   if r1 = #'M                  ;right confirmation
      lda r1,#faktyp            ;page setting
      call putarg               ;put into ddt's stuff
      lda r1,#mgnl              ;where it is
      call newarg
   endif
endroutine opmessgen

   $dopatch ddtdisp+<'G*words>,ddtcode
      opmessgen+ovrrid
   page ddtcode
|

.comnt |  don't want this this time
;set up lights
routine oplites
   call opget
   lda r7,#watch1               ;where to put
   and r3,#packm                ;just offset
   ior r3,#m2                   ;through m2
   lda r2,mapvar                ;default map if none given
   if r5        ;;pfx           ;map given
      lda r2,lmap(r4)  ;;prefix ;use logical only
   endif
   ifnot r1 = #'U               ;not upper light
      add r7,#watch2-watch1     ;maybe lower
      ifnot r1 = #'L            ;not lower
         ifnot r1 = #'R         ;not reset- no good
            return
         endif
         lda r1,#watch          ;it's a reset,
         sta r1,watch1          ;so fix upper
         lda r3,#watchh         ;set  up for lower
      endif
   endif
   sta r3,(r7)+ ;;watch1 watch2 ;set address
   sta r2,(r7)  ;;watm1 watm2   ;and map
endroutine oplites

   $dopatch  ddtdisp+<'L*words>, ddtcode
      oplites+ovrrid
   page DDTCode

|
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 69
IMPDDT.PLR;1    PAGE 8  OPHELP: IMP Operator commands

.comnt |
do modem and host ophelps
"M" gets modem, "H" gets host
"B" returns block, "L" loops at interface, "M" loops at Bell modem
"U" unloops |

4DAC 1076       routine opmodem
4DAE 4A38 0018     add r3,#<<m2pblk-h2pblk>/words>-nfh  ::ddaccum sum
4DB2 9002       entry ophost
4DB4 1076
4DB6 4078 44D4     call argreset                ;get rid of current input
4DBA A2B1          sll r3,1     ::ddaccum sum   ;byte index
4DBC 4AB8          add r3,#nfh2
4DBE 4B38 00FF     and r3,# HFF                 ;remove byte overflo if any
4DC2 4E38 0048     if r3 <= #m2pend-h2pblk      ;make sure reasonable
4DC6 9C0E
4DC8 702B 6326        lda r2,h2pblk(r3)         ;get block (lge modem # get host})
4DCC 990B             ifnot odd                 ;block exists
4DCE 4B28 7FFE           and r2,#7FFE           ;neg (when device goes) OK
4DD2 4078 4D70           call tlookup, blist, ctable    ;decide what to do
4DD6 4DE8
4DD8 4DEE
4DDA 9A04                ifnot fail             ;okay to do
4DDC 4812                   lda r1,r2           ;move block around
4DDE 4073                   call (r3)           ;do it
4DE0 6006                   return
                         endif
                      endif
                   endif
4DE2 4078 44CA     call rubout
4DE6 90FD       endroutine ophost

FC00 0200          $dopatch ddtdisp+<'M*words>,ddtcode
443A 4DAC             opmodem
FC00 0200          $dopatch ddtdisp+<'H*words>,ddtcode
4430 4DB4             ophost
FC00 0200          Page DDTCode
```

```
4DE8   42       blist:   .asciz /BLMU/
4DE9   4C
4DEA   4D
4DEB   55
4DEC   00
                .even

4DEE            table ctable
4DEE 4D5C          putarg                    ;B- return block address
4DF0 4DF6          intloop                   ;L- loop at interface
4DF2 4DFC          modloop                   ;M- loop externally at modem
4DF4 4E02          unloop                    ;U- unloop modem/host
                endtable ctable

                ;loop interface internally
                intloop:
4DF6 4818 4000     lda r1,#hloopi       ;;mloopi
4DFA 9005         br blkinsrt

                ;loop interface at modem (modems and VDHs)
                modloop:
4DFC 4818 8000     lda r1,#mloope
4E00 9002         br blkinsrt

                ;clear all loops
                unloop:
4E02 4890          lda r1,#0

4E04 1076       routine blkinsrt

4E06 707A 0020     lda r7,iobloc(r2)    ;distinguish between host and get device addr
4E0A 7077          lda r7,(r7)          ;get devtyp,,devnum
4E0C A6F8          srl r7,8             ;just devtp
4E0E 4EF2          if r7 = #hostid_-8   ;a host
4E10 8104
4E12 301A 0024        sta r1,hiloop(r2) ;set the bit
4E16 9003          else
4E18 301A 001E        sta r1,mloop(r2)
                   endif
4E1C 4078 4820     call itextout, %3space
4E20 465E

4E22 6006       endroutine blkinsrt
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 71
IMPDDT.PLR;1    PAGE 10         OPHELP: IMP Operator commands

.comnt | IMP panic and nice starts, loads, halts here|

.comnt |NICE stops: dispatched from "N"|
                opnice:
4E24 1076        save r7
                   call tlookup, stops, nstpflags       ;get confimation char, then flag
4E26 4078 4D70
4E2A 4E58
4E2C 4E62
4E2E 9006          br pstops

.comnt |PANIC stops: dispatched from "P"|
                oppanic:
4E30 1076       save r7
                   call tlookup, stops, pstpflags       ;get confirmation char and then4E32 4078 4D70   save r7
4E36 4E58
4E38 4E5C

.comnt |common routine to put flag into nice stop|
                pstops:
4E3A 9A0C          ifnot fail
4E3C 7078 62B4        if ddaccum = #dempas      ;good password
4E40 4E78 CAFE
4E44 8107
4E46 3038 651C           sta r3,nsrtf
4E4A 4078 4820           call itextout, %3space
4E4E 465E
4E50 6006                pop r0          ;;return
                      endif
                   endif
4E52 4078 44CA     call rubout           ;no good
4E56 6006       pop r0

4E58   48       stops:  .asciz /HSR/             ;halt, start, reload
4E59   53
4E5A   52
4E5B   00
                .even

4E5C            table pstpflags                  ;panic stop flags
4E5C 0003          p.halt
4E5E 0001          p.restart
4E60 0002          p.reload
                endtable pstpflags

4E62            table nstpflags                  ;flags for nice stop ,
4E62 FFFF          n.halt
4E64 FFFD          n.restart
4E66 FFFE          n.reload
                endtable nstpflags

FC00 0200          $dopatch ddtdisp+<'N*words>,ddtcode
443C 4E24             opnice
FC00 0200          $dopatch ddtdisp+<'P*words>,ddtcode
4440 4E30             oppanic
FC00 0200          page DDTCode
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 72
IMPDDT.PLR;1    PAGE 11         OPHELP: IMP Operator commands

.comnt |Return iokill address: dispatch on "K"|
4E68 1076       routine opkillio
4E6A 4078 4D98     call opget
4E6E 4E18 0049     if r1 = #'I
4E72 8108
4E74 4890             lda r1,#0                ;page to find iokill
4E76 4078 4D5C        call putarg
4E7A 4818 411E        lda r1,#iokill
4E7E 4078 4D3E        call newarg
                   endif
4E82 6006       endroutine opkillio

FC00 0200          $dopatch ddtdisp+<'K*words>,ddtcode
4436 4E69             opkillio+ovrrid
FC00 0200          page DDTCode

.comnt |get help for a neighbor asking for reload
dispatch from "R"|
4E84 1076       routine opreload
4E86 4078 4D98     call opget
4E8A 4E18 004E     if r1 = #'N                  ;good confirm
4E8E 8107
4E90 3038 6566        sta r3,pchelp     ;;ddaccum sum   ;where to send setups
4E94 4078 4820        call itextout, %3space
4E98 465E
4E9A 9003           else
4E9C 4078 44CA        call rubout
                   endif
4EA0 6006       endroutine opreload

FC00 0200          $dopatch ddtdisp+<'R*words>,ddtcode
4444 4E85             opreload+ovrrid
FC00 0200          page DDTCode

.comnt |Clear traps: <esc>|
4EA2 1076       routine opclearill
4EA4 4078 479C     call iddtout, '$
4EA8 0024
4EAA 4078 4D98     call opget
4EAE 4E18 0043     if r1 = #'C                  ;confirmed
4EB2 810F
4EB4 4838 00A2        lda r3,=cilend-cilovf
                      repeat
4EB8 574B A0C0           tst r4,cilovf+foo(-r3)
4EBC 88FE             until loop
                      endrepeat
4EBE 49B1             sub r3,=1 ;;minus        ;what a bum: set r3 minus
4EC0 3038 6206        sta r3,dinit             ;set flag to refresh display
     0001             .lif nz PSE
4EC4 7038 A162        lda r3,pilovp+foo
4EC8 4078 4820        call itextout, %3space   ;confirm
4ECC 465E
4ECE 9003           else
4ED0 4078 44CA        call rubout
                   endif
4ED4 6006       endroutine opclearill
```

```
$dopatch ddtdisp+< H1B*words>,ddtcode
```

```
FC00 0200
43D6 4EA2            opclearill
```

```
                ;DDT timeout dispatch

FC00 0200          page DDTCode

     4ED6       ddttab=.
4ED6 0000          .word 0
                vdhp4:
4ED8 1CBC          tosst
     4EDA          pddttab=.              ;points to free entry in ddttab
4EDA 0000          .word 0
4EDC 0000          .word 0
4EDE 0000          .word 0
4EE0 0000          .word 0
4EE2 0000          .word 0
4EE4 0000          .word 0
4EE6 0000          .word 0
4EE8 0000          .word 0
4EEA 0000          .word 0
4EEC 0000          .word 0
                eddttab:
4EEE 0000          .word 0      ;end of list

                ;define macro for inserting into ddt timeout list
                .macro dstolist subrt
                  .if nz pddttab-<eddttab-words>        ;if any room left
                    $dopatch pddttab, ddtcode   ;add the entry
                      subrt
                    pddttab=.              ;get set for next entry
                    tosst
                    .word 0
                  .iff
                    .error DDT timeout list overflowed
                  .endc
                .endm

                ;get table into page pointer
FC00 0200          $dopatch m0#topntr, ddtcode
40BC 4ED6             ddttab

     0008       .radix  010             ;octal for everybody else
```

```
.INSERT "DISPLY"
    .INSRT DISPLY
```

```
                .stitle IMP video display handler

     0010       .Radix  H10             ;this source in hex
     0000       .parity 0               ;7 bit ascii here
                page Vars
                  locdef dsplok,<;display variables lock - page 75>

                page DDTVars
5B3E            dspsp:  .blkw 1
5B40            dssyssp:        .blkw 1
5B42              stack ds,< D12>
5B5A
5B5A

                ;some useful display characters:
     0008         home= 010                      ;home char on infotons
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 77
DISPLY.PLR;1    PAGE 2  IMP video display handler

                        page LCode

162A 1076       routine jdsply, uses r1-r5
162C 4078 0FBC    call finddt                   ;get the DDT page
1630 4078 1650    call dsppol                   ;and poll DISPLAY
1634 6006       endroutine jdsply

                page LCode


1636 1076       routine dspsleep, local r1-r5
1638 1016
163A 1026
163C 1036
163E 1046
1640 1056
1642 3068 5B3E    sta sp,dspsp                  ;save our stack
1646 7068 5B40    lda sp,dssyssp                ; and get system's stack
164A 3008 72A4    unlock m1#dsplok              ;release both stacks
164E 6006         pop r0                        ;return to system
                dsppol:                         ;entry from system
1650 1076         save r7                       ;save return
1652 7078 B2A4    lock r7,dsplok                ;get control of stacks
1656 9AFE
1658 3068 5B40    sta sp,dssyssp                ;save system stack
165C 4868 5B5A    lda sp,#dsstack               ;default stack
1660 4877         lda r7,r7     ;;dsplok
1662 9921           bo dsplyloop                ;broken lock- reset
1664 7068 5B3E    lda sp,dspsp                  ;get display's stack
1668 6056       endroutine dspsleep
166A 6046
166C 6036
166E 6026
1670 6016
1672 6006
```

```
                ;table of addresses to display.

     0001       .if z PSE                       ;ARPANET case:

                .comnt |
                table dsploc
                        mapcom
                        maprel
                        mapddt
                        mapddt+2        ;;mapcod ;if imp is there
                        mapddt+4        ;;mapfak ;if imp is there
                        mapddt+6        ;;maptip ;if tip is there
                        mapvar
                        mapvar+2        ;;mapv2 ;if imp
                        mapvar+4        ;;maptvr
                        lmap+%nvarsp    ;;mapb1 ;if imp
                        lmap+%nvarsp+2  ;;mapb2
                        maprel+%ncodep
                        mapddt+%ncodep
                        mapddt+%ncodep+2        ;;mapcod+%ncodep
                        mapddt+%ncodep+4        ;;mapfak+%ncodep
                        mapddt+%ncodep+6        ;;maptip+%ncodep
                endtable dsploc
                |
                .iff    ;z PSE                  ;PSE case

1674            table dsploc
     0010         .rept  D16
                    hdisp+<.rpcnt*words>        ;hosts
1674 6300        .endr
1676 6302
1678 6304
167A 6306
167C 6308
167E 630A
1680 630C
1682 630E
1684 6310
1686 6312
1688 6314
168A 6316
168C 6318
168E 631A
1690 631C
1692 631E
     0008       .rept  D8
                  mdisp+<.rpcnt*words>          ;modems
1694 62F0        .endr
1696 62F2
1698 62F4
169A 62F6
169C 62F8
169E 62FA
16A0 62FC
16A2 62FE
                endtable dsploc
```

```
.endc    ;z PSE
```

```
                dsplyloop:
                repeat                          ;while the seas are wet
                   repeat
16A4 4078 1636        call dspsleep
16A8 4AC1             add r4,#1
16AA 4B48 00FF        and r4,#0FF
16AE 8AFB            until zero                 ;only every 6 seconds
                     endrepeat
16B0 7078 626A       ifnot dspflag                      ;okay to display
16B4 8A2A
16B6 7078 A206          if dinit+foo            ;need to display traps
16BA 9A27
16BC 4078 171E            call idspout, asciff  ;move to home on some terminals
16C0 000C
     0001                 .if nz PSE            ;Platform: display modem/host status
16C2 48A0                   lda r2,#0           ;index into table of maps to display
                            repeat              ;write maps
16C4 601A 1674                lda r1,dsploc(r2)+  ;a variable to write
16C8 7011                     lda r1,(r1)         ;its contents
16CA 4078 103C                call hexout, dspout ;type this map
16CE 171A
16D0 4078 171E                call idspout, ascispace           ;space in between
16D4 0020
16D6 4E28 0030                until r2 >= #ldsploc      ;end of table
16DA 92F5
                            endrepeat
16DC 4078 170A              call dspeol
                          .endc             ;;nz PSE
16E0 48A0                 lda r2,#0               ;index to traps
                          repeat
16E2 48BA                   lda r3,#1cilbuf/words       ;counts up to next trap
                            repeat              ;print trap line
16E4 601A 60C2                lda r1,cilops(r2)+
16E8 4078 103C                call hexout, dspout       ;display number
16EC 171A
16EE 4078 171E                call idspout, ascispace
16F2 0020
16F4 49B1                     sub r3,#1         ;count one word
16F6 8AF7                     until zero        ;until next trap
                            endrepeat
16F8 4078 170A              call dspeol         ;set up next line
16FC 4E28 00A0              until r2 >= #cilend-cilops  ;until past end
1700 8204
1702 707A 60C2              while cilops(r2)    ;stop after first zero
1706 8AEE
                          endrepeat
                        endif
                     endif
1708 90CE         endrepeat                      ;forever
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 80
DISPLY.PLR;1    PAGE 5  IMP video display handler


170A 1076       routine dspeol, uses r1
170C 4078 171E     call idspout, ascicr
1710 000D
1712 4078 171E     call idspout, ascilf
1716 000A
1718 6006       endroutine dspeol

171A 1076       routine dspout, arg r1
171C 9003        entry idspout, inline r1, uses r1
171E 6017
1720 1076
                   repeat
1722 4078 0FD2        call rbfput, ttyobf       ;try to put char on tty
1726 6296
1728 810B              while equal              ;buffer full- wait
172A 48F6                set dsppok = #dsppid
172C 3078 6274
1730 4878 0014           set @pid = #ttypid              ;start the terminal running
1734 30F8 00AC
1738 4078 1636           call dspsleep
173C 90F3          endrepeat
173E 6006       endroutine dspout
```

```
                ;two routines to control display (called via ddt)
1740 1076       routine cntll, arg r1
1742 4078 4798     call ddtout                  ;clear screen
1746 4878 8000     lda r7,#sign
174A 3478 6206     iorm r7,dinit                ;mark screen as needing display
174E 6006       endroutine cntll

FC00 0200          $dopatch ddtdispatch+< 014*words>, DDTCode
43B8 1740            cntll
FC00 0200          page DDTCode


                ;turn display on and off. Dispatched from  D
4EF0 1076       routine dspswitch, arg r1
4EF2 4078 4820     call itextout, dsptxt
4EF6 4F16
4EF8 4818 004E     lda r1,#'N                   ;default to typing "n"
4EFC 7078 A26A     lda r7,dspflag+foo           ;check flag
4F00 8A05          if zero                      ;was off, turn on now
4F02 3008 626A        sta r0,dspflag
4F06 4818 0046        lda r1,#'F                ;to type "off"
                   endif
4F0A 4078 4798     call ddtout                  ;type last char
4F0E 4078 4820     call itextout, crlftxt
4F12 458E
4F14 6006       endroutine dspswitch

FC00 0200          $dopatch ddtdisp+<4*words>,ddtcode
43A8 4EF0             dspswitch
FC00 0200          page ddtcode

4F16   44       dsptxt: .asciz /DO/
4F17   4F
4F18   00
                .even


     0008       .radix  010             ;default octal for the rest
```

```
.INSERT "IMP.LOCAL",LOCAL
    .INSRT IMP.LOCAL
```

```
                .stitle IMP local memory code

                        Page LCode

        .comnt  |

    Call rstart via r7 with a pid in r1.  The pid will be saved
  in a circular buffer (ring), if that pid is not already stored in
  the ring.  r2 and r3 are destroyed; if the ring is full rstart will
  trap and ignore the call.
    The specified pid is poked by rstgo, which itself has a very
  small pid assigned.  Each time rstgo is run, it pokes
  one and only one pid from ring, and removes that pid from ring.
|

1750 7038 A4B2  Rstart: lda r3,ringlk   ;lock the ring structure.
1754 9AFE                bz .-4         ;wait until we own the ring.
1756 7038 64B6           lda r3,ringf   ;pointer to first entry in ring.
175A 7028 64B4           lda r2,ringc   ;number of entries in ring.
175E 9A11                bz rst02       ;empty, don't search ring.
1760 6E13       rst00:   cmpb r1,(r3)+  ;Is this pid value already in ring?
1762 9113                be rst03       ;found, don't enter same pid twice.
1764 4E38 64D0           cmp r3,=ringe  ;Last byte in the ring?
1768 9203                bg rst01       ;no, continue search.
176A 4838 64B8           lda r3,=ring   ;yes, re-initialize pointer.
176E 49A1       rst01:   sub r2,=1      ;look at next entry.
1770 8AF8                bnz rst00

1772 7028 64B4           lda r2,ringc   ;enter another pid into buffer.
1776 4E28 0018           cmp r2,=ringln ;is the buffer full?
177A 9203                bg rst02       ;no, all is well.
177C E2C8                Trap 1310,<;ringc overflow in rstart - page 82>
177E 9005                br rst03       ;ring is full, take error exit.

1780 4AA1       rst02:   add r2,=1      ;increase number of entries.
1782 3028 64B4           sta r2,ringc
1786 3813                stab r1,(r3)   ;store the pid in the buffer.

1788 3008 A4B2  rst03:   unlock ringlk  ;unlock ring.
178C 48B2                lda r3,=stkpid ;Poke the restarter pid so
178E 30B8 00AC           sta r3,@pid    ;remover (Rstgo) will be run.
1792 4007                jmp (r7)       ;return to caller.
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 84
IMP.LOCAL;1     PAGE 2  IMP local memory code

                ;routine to call block transfer routine BLT
                ;runs at very low priority (4)
                ;calls rstart if still more to do upon exit

1794 7078 00B0  bltcal: setmap m0,mapre1
1798 3078 FC00
179C 7018 5DBC          lda r1,bltst    ;;bltact
17A0 7418 5EB0          ior r1,pkcst    ;;pkcact
17A4 8910               if odd                  ;if either BLT or PKC active
17A6 4078 0990            call polblt           ;poll BLT process
17AA 7078 00D0            setmap <m1,m3>,mapvar ;all maps were smashed
17AE 3078 FC02
17B2 3078 FC06
17B6 7078 00D2            setmap m2,mapv2
17BA 3078 FC04
17BE 4894                 lda r1,=bltpid
17C0 4078 1750            call rstart
                        endif
17C4 4008 1086          jmp loopm
```

```
                ;shared reliability timeout routines

                ; test r6 to see if it is a subroutine call
                ; call return with r2 = target of caller
                ; retx doesn't check r2

.comnt |
retchk: cmp r2,-2(r6)    ;does call target match?
         qutpat retuq    ;tsk
         bne retu3       ;no: return not-equal condition
retx:    lda r5,-4(r6)   ;was it a jsb r7, ?
         qutpat retuq    ;tsk
        cmp r5,=.first jsb r7,0
retu3:  jmp (r7)         ;fail returns not-equal condition

retuq:  rst %e           ;clear equal flag.
        jmp (r7)         ;return not-equal
|
FC00 0400                Page PkgCode

                ;routine to put a broken packet on diagnostic queue.
41AA 1076       routine ckqput,arg r2
41AC 4078 146C    call wheorb,whm2i?whf2h       ;move to fake queue.
41B0 C000
41B2 7078 AFCA    lock staiob+lockfd    ;enqueue bad guy
41B6 9AFE
41B8 30A8 63C2      sta r2,@eckq
41BC 3028 63C2      sta r2,eckq
41C0 3008 AFCA    unlock staiob+lockfd

41C4 7078 00B6    lda r7,mapfak
41C8 3078 FC04    sta r7,%map2
41CC 3008 9EC6    sta r0,m2#<dgsb+snon>
41D0 7078 00D2    lda r7,mapv2
41D4 3078 FC04    sta r7,%map2
41D8 6006       endroutine ckqput
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 86
IMP.LOCAL;1     PAGE 4  IMP local memory code

                        Page Lcode
.comnt  |
lpcksubr - store packet length from r3, then cksum packet.
pcksubr - cksum packet to length in bufe.

        These routines are the system's packet checksummer.  Stores
(lpcksubr only) packet length, then calculates
correct offset in subchn to do checksum calculation.  r3 returns
with the checksum.  returns zero if checksum ok, non-zero if
checksum bad.
|

17C8 1076       routine lpcksubr,arg r1/r3,local r5,result r3
17CA 1056
17CC 49B2         sub r3,=hrdoff        ;compensate for hardware offset.
17CE 3039 0090    sta r3,bufe(r1)       ;save buffer end pointer.

17D2 9003       entry pcksubr,arg r1,local r5,result r3
17D4 1076
17D6 1056
17D8 7039 0090    lda r3,bufe(r1)       ;buffer end pointer.
17DC 4AB2         add r3,=hrdoff        ;compensate for hardware length offset.
17DE 48F0         lda r7,=0             ;figure out where to start subchn
17E0 4973         sub r7,r3             ;negative length.
17E2 A6B1         srl r3,1              ;initialize checksum to include length.
17E4 4851         lda r5,r1             ;address of buffer to checksum.
17E6 407F 1888    call subchn(r7)       ;checksum packet.
17EA 8A05         if zero               ;checksum good.
17EC 6056           return zero
17EE 6076
17F0 4FF0
17F2 4007
                  endif
17F4 6056       endroutine lpcksubr     ;,pcksubr ;normally returns non-zero.
17F6 6006

                ; subtract chain
                ;r3: accumulates checksum
                ;r5: pointer to what to checksum

     0048               .rept  D72
                .xlist
                        sub r3,(r5)+
                .list
                        .endr
1888 4007       subchn: jmp (r7)
```

```
                ;**********hot hi**********

                ; put leaders onto the control queues
                ;   also see rfledp/ledpc
                ; called from hi,forus,dedtrn, and impdwn
                ; r1 is destroyed
                ; r2 is destroyed
                ; r3= transaction block (preserved)
                ; r4= host parameter block
                ; r6 is destroyed
                ; r7= entry
                ;  uses temp2
                ;enter at ledp0 with leader already set-up
                ;enter at ledp2 with new trhstl in r1 (forus)
                ;enter at ledpf to fix up leader queue (see dedtrn)

                page LCode

188A 701B 0004  ledp0:  lda r1,trhstl(r3)       ;will copy back
188E 706C 405C  ledp2:  lda r6,lockih(r4)       ;lock the host
1892 9AFE                bz .-4
1894 304B 000E          sta r4,trhost(r3)       ;new owner
1898 4828 4008          lda r2,= H4000+ttledr
189C 302B 000C          sta r2,trntim+<0*trstat>(r3)
18A0 301B 0004          sta r1,trhstl(r3)       ;maybe new if enter at ledp2
18A4 706C 005E          lda r6,nxtled(r4)       ;any leaders pending here?
18A8 8A04                bnz ledp1      ;yes
18AA 486B 76F0  ledpf:  lda r6,=-trnblk(r3)     ;no, get our trnblk offset
18AE A6E4               srl r6,4
18B0 4A68 0100  ledp1:  add r6,= H100   ;and count this new one
18B4 306C 005E          sta r6,nxtled(r4)       ;into host's pointer
18B8 300C 405C          unlock lockih(r4)
18BC 786C 000E          ldab r6,hotpid(r4)
18C0 30E8 00AC          sta r6,@pid
18C4 4007               jmp (r7)


                ;call here via r7 to flush trpack if it exists
                ;r3 has trnblk address

18C6 1076       routine trnfls, uses r2, arg r3
18C8 702B 000E    lda r2,trpack(r3)
18CC 9A06         ifnot zero
18CE 352B 000E      eorm r2,trpack(r3)
18D2 4078 1332      call flush,whhi
18D6 2000
                  endif
18D8 6006       endroutine trnfls
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 88
IMP.LOCAL;1     PAGE 6  IMP local memory code

                ;********** imp to host *************

                        page HLCode

                ; pid entry ih
                ; start of main loop ihnext

18DA 000A       nonops:  H8+2   ::hosfake hostip  ;bit table of host types
                                                  ;fakes and TIPs get no nops

18DC 7049 81C0  ih:      lda r4,mblks(r1)
18E0 703C 404E           lda r3,ihloc(r4)
18E4 9AFE                 bz .-4
18E6 705C 0020           lda r5,iobloc(r4)
18EA 701C 0054           lda r1,ihgoin(r4)       ;also not under lockih
18EE 9A24                 bz ihr0         ;host timed out - reset the world
18F0 703D 000C           lda r3,statoh(r5)
18F4 4F38 2000           tst r3,=hbusy
18F8 8A14                 bnz ihdb2       ;hardware not yet done
18FA 4F38 0100           tst r3,=hquit
18FE 9A06                 bz ihq05        ;no quit
1900 702C 0034           lda r2,hisp(r4)
1904 E604                Trap 3004,<;Host output err - page 87>
1906 48B4                lda r3,=hstqut
1908 901E                br ihr5

190A 701C 405C  ihq05:   lda r1,lockih(r4)
190E 9AFE                 bz .-4
1910 703C 0056           lda r3,ihwq(r4)
1914 408C 0050           jmp @ihlo(r4)

1918 300C 405C  ihdba:   unlock lockih(r4)
191C 307C 0050  ihdb:    sta r7,ihlo(r4)
1920 300C 404E  ihdb2:   unlock ihloc(r4)
1924 4008 108E           tr loop
```

```
                ;wait software reset completion

1928 4078 1918  ihde0:  jsb r7,ihdba     ;sleep a bit
192C 783C 001F  ihdead: ldab r3,hihd(r4)        ;hi signals via hihd
1930 4EB4               cmp r3,=hninit  ;4 means resetting
1932 91FB                be ihde0       ;resetting still
1934 9030               br ihr7         ;for a reset sequence

1936 48B2       ihr0:   lda r3,=htardy
1938 707D 0006          if statih(r5) .nbit. #hready
193C 4F78 1000
1940 8A02
1942 48B1                 lda r3,=hrdown
                        endif

1944 707C 405C  ihr5:   lock lockih(r4)         ;now lock host
1948 9AFE
194A 707C 0002          ifnot hostyp(r4)        ;;hosreal ;can't reset fakes
194E 8A05
1950 4878 0100            lda r7,=hreset+<<data& HF>/words>
1954 307D 000C            sta r7,statoh(r5)     ;;ihledr ;reset output
                        endif
1958 300C 0054          set ihgoin(r4)
195C 4E48 7210          if r4 <> #fakeh3        ;dont try to discard discard
1960 911A
1962 7E3C 001F            if byte r3 <> hihd(r4)        ;no change
1966 9105
1968 383C 001F              stab r3,hihd(r4)
196C 3008 6512              set ophgo
                          endif
1970 4078 2F40  ihr1:     call ledget
1974 90FE                  br ihr1              ;more on leader queue
1976 4868 02F8            lda sp,=lstack
197A 702C 4068            set r2 = ihbuff+foo(r4)       ;was a buffer being sent?
197E 9A03                 ifnot zero           ;yes
1980 4078 5C1A              call ihdmpb        ;mark buffer inc., dump it
                          endif
1984 485C 0060            lda r5,=shq(r4)
1988 4078 5C06            call ihdump
198C 485C 0064            lda r5,=shpq(r4)
1990 4078 5C06            call ihdump
                        endif

1994 4895       ihr7:   lda r1,=5
1996 301C 005A          sta r1,specal(r4)
199A 707C 0002          lda r7,hostyp(r4)       ;get the host type
199E 707F 0EB6          lda r7,bittab(r7)       ;get NONOPS bit, this type
19A2 7778 18DA          if r7 .nbit. nonops     ;if this host is normal
19A6 8A03
19A8 787C 401C            ldab r7,homode+foo(r4)        ;then start old format
                        endif
19AC 4078 1918          call ihdba              ;sleep here
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 90
IMP.LOCAL;1     PAGE 8  IMP local memory code

                ;r3= queue to use
                ;r7=time to use

19B0 4868 04B0  ihnext: lda r6,=sec30
19B4 306C 0052          sta r6,ihtt(r4)
19B8 703C 005A          lda r3,specal(r4)
19BC 8A10                bnz ih1        ;special control message
19BE 4078 2F40          call ledget
19C2 9042                br ihctl       ;non empty control queue
19C4 483C 0064          lda r3,=shpq(r4)
19C8 7063               lda r6,(r3)
19CA 8947                bno ih5        ;non empty priority queue
19CC 49B4               sub r3,=shpq-shq
19CE 7063               lda r6,(r3)
19D0 894D                bno ih6        ;non empty regular queue
19D2 300C 405C          unlock lockih(r4)
19D6 701C 4052  ih0:    lda r1,ihtt+foo(r4)
19DA 903C               br ih4

19DC 4B38 001F  ih1:    and r3,= H1F
19E0 49B1               sub r3,=1
19E2 303C 005A          sta r3,specal(r4)       ;number of nops to send
19E6 9A22                bz ihn2        ;now send a reset
19E8 781C 001C          ldab r1,homode(r4)
19EC 8A0B                bnz ihn1       ;if already in new format
19EE 300C 405C          unlock lockih(r4)
19F2 4818 0400          lda r1,=cnop* H100      ;send an old nop
19F6 486C 8012          lda r6,=ihledr+hendf+2(r4)
19FA 4078 5BCC          jsb r7,ihls
19FE 4078 191C          jsb r7,ihdb

1A02 300C 405C  ihn1:   unlock lockih(r4)
1A06 781C 0029          ldab r1,hihost+1(r4)    ;send a new nop
1A0A 301C 0014          sta r1,ihledr+hstl(r4)
1A0E 7018 62E2          lda r1,mine
1A12 301C 0016          sta r1,ihledr+dstl(r4)
1A16 701C 4018          lda r1,ihledr+midl+foo(r4)      ;;subtyp
1A1A 4818 0F00          lda r1,=hicode
1A1E 48B4               lda r3,=cnop
1A20 486C 8018          lda r6,=ihledr+hendf+midl(r4)
1A24 4078 5BCC          jsb r7,ihls
1A28 9015               br ih4

1A2A 4818 0F00  ihn2:   lda r1,=hicode  ;send a new reset
1A2E 48BA               lda r3,=creset
1A30 301C 0010          sta r1,ihledr+netl(r4)
1A34 303C 0012          sta r3,ihledr+typl(r4)
1A38 783C 401F          ldab r3,hihd+foo(r4)    ;;hostup
1A3C 9A05                bz ihctl
1A3E 703C 402C          lda r3,deadsc+foo(r4)
1A42 3008 6512          sta r0,ophgo
1A46 300C 405C  ihctl:  unlock lockih(r4)
1A4A 486C 8018          lda r6,=ihledr+hendf+midl(r4)
1A4E 4078 5B68          jsb r7,ihlsn    ;leader words in place
1A52 4078 191C  ih4:    jsb r7,ihdb     ;wait for control or nothing to do
1A56 90AD       ihidle: br ihnext
```

```
1A58 4873        ih5:    lda r7,r3
1A5A 702C 0060           lda r2,shq(r4)
1A5E 9908                 bo ih7 ;empty reg queue but full pri queue
1A60 702A 0350           lda r2,chan-chain(r2)
1A64 712E 0350           sub r2,chan-chain(r6)
1A68 8B03                 bnm ih7        ;pri queue has older message
1A6A 487C 0060   ih6:    lda r7,=shq(r4)
1A6E 303C 0056   ih7:    sta r3,ihwq(r4) ;assume higher-priority is good
1A72 7027                lda r2,(r7)     ;;shq,shpq
1A74 706A 0350           lda r6,chan-chain(r2)
1A78 7168 62DA           sub r6,time     ;calculate time left for this message
1A7C 4EE2                if r6 < =2      ;packet is old
1A7E 8202
1A80 48E2                  set r6 = =2   ;so give it at least 2 left
                         endif
1A82 306C 0052           sta r6,ihtt(r4)
1A86 4E68 0258           if r6 < #sec15  ;old message, put on q
1A8A 8203
1A8C 307C 0056             sta r7,ihwq(r4)       ;oops, there's an old message
                         endif
1A90 703C 0056           lda r3,ihwq(r4) ;now reget queue we picked
1A94 7023                lda r2,(r3)     ;its head
1A96 4078 14FC           call ounpckc,whih
1A9A 0040
1A9C 8A03                if fail
1A9E 4008 5D92             tr iherr
                         endif
1AA2 4861                lda r6,r1
1AA4 300C 405C   ihgtgo: unlock lockih(r4)  ;patched to JMP IHGTWY by PTIP
1AA8 781E 0009           ldab r1,seqh+1(r6)      ;source host
1AAC 703E 009C           lda r3,bufb(r6) ;pointer to receive block
1AB0 9908                 bo ihraw       ;no receive block: raw pkt
1AB2 701B 65B6           lda r1,rmctl(r3)        ;handling type
1AB6 A494                sra r1,4
1AB8 4B18 8700           and r1,=priled}maxled   ;;gethan
1ABC 7C1B 65B4           iorb r1,rmhost(r3)      ;remote host is source
1AC0 483C 0010   ihraw:  lda r3,=ihledr(r4)
1AC4 4878 0F00           lda r7,=hicode
1AC8 2073                sta r7,(r3)+            ;;netl
1ACA 707E 0002           lda r7,typh(r6)
1ACE 4B78 0F00           and r7,=trcbit+pflags
1AD2 2073                sta r7,(r3)+            ;;typl
1AD4 2013                sta r1,(r3)+            ;;hstl
1AD6 707E 0006           lda r7,srch(r6)
1ADA 2073                sta r7,(r3)+            ;;srcl
1ADC 707E 000E           lda r7,midh(r6)
1AE0 2073                sta r7,(r3)+            ;;midl
1AE2 7076                lda r7,(r6)             ;;neth
1AE4 3073                sta r7,(r3)             ;;lenl
1AE6 7078 00D2           lda r7,mapv2
1AEA 3078 FC04           sta r7,%map2
1AEE 786C 001C           ldab r6,homode(r4)
1AF2 4B68 001F           and r6,= H1F            ;padding
1AF6 4A6C 001A           add r6,=ihledr+lenl(r4)
1AFA 4078 5B68           jsb r7,ihlsn
1AFE 4078 191C           jsb r7,ihdb     ;wait for packet leader to go
1B02 400[illegible]5D08  jmp ihpack      ;on warm page
```

```
.INSERT "CONFIG"
    .INSRT CONFIG
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 93
CONFIG.PLR;1    PAGE 2  IMP Configuration Code

                        .stitle IMP Configuration Code

     0001       .if nz MISw             ;M/I bus special code

                        Page LTVars     ;space here for now

0040            locflg: .blkw 1         ;local copy of last CKILL

                .endc ;nz MISw

                        Page LCode

.comnt |
CONFIG
Configuration code central dispatch.
Calls routines which configure the system.
Each code page has address of routine to call as the first
entry in its timeout table ---TAB, which TOPNTR points to.
Poked every slow tick by TIMEOUT (TO).
|

                ;process con,pid=conpid

                conloop:                        ;start here first time around
                repeat                          ;forever
1B06 4890         lda r1,=0
                  repeat                        ;for all code pages
1B08 6079 00B0      lda r7,lmap(r1)+
1B0C 9911           ifnot odd                   ;page exists
1B0E 3078 FC00        sta r7,%map0
1B12 7078 40BC        lda r7,topntr
1B16 9A0C             ifnot zero                ;there is a table
1B18 7027               lda r2,(r7)             ;first table entry
1B1A 9A0A               ifnot zero
1B1C 1016                 save r1               ;remember LMAP index
1B1E 7078 00D2            setmap m2,mapv2       ;default map 2
1B22 3078 FC04
1B26 4072                 call (r2)             ;call indicated routine
1B28 6016                 restore r1            ;get back LMAP index
1B2A 4078 1B42            call csleep           ;rest a bit too
                        endif
                      endif
1B2E 4E18 0010      until r1 = =%ncodep         ;done all code pages
1B32 81EB
                  endrepeat
1B34 4878 FFFF    lda r7,=-1
1B38 3078 620A    sta r7,sidflg                 ;let TIMEOUT run
1B3C 4078 1B62    call cslee2                   ;sleep without poke
1B40 90E3       endrepeat
```

```
                ;CSLEEP
                ;Put the CONFIG process to sleep
                ;assumes M1 set to MAPVAR

1B42 1076       routine csleep, local m0, arg m1, result m3
1B44 7078 40BE
1B48 1076

                  ;poke conpid          ;reawaken us too
1B4A 4878 002E    lda r7,=conpid
1B4E 30F8 00AC    sta r7,@pid           ;don't yet trust the poke macro.
1B52 4078 1B62    call cslee2           ;and resume here

1B56 6076       endroutine csleep
1B58 707F 00B0
1B5C 3078 FC00
1B60 6006

1B62 1076       routine cslee2,local r1-r5,local m2,arg m1,result m3
1B64 1016
1B66 1026
1B68 1036
1B6A 1046
1B6C 1056
1B6E 7078 80BE
1B72 1076

1B74 3068 654E    sta sp,confsp         ;save the stack pointer too .
1B78 3008 6550    unlock m1#conlok
1B7C 4008 1072    jmp loopmv

                con:                    ;enter here from LOOP
1B80 7078 A550    lock conlok
1B84 9AFE
1B86 9907         if nodd & sidflg              ;stack still good
1B88 7078 620A
1B8C 9A04
1B8E 7068 654E      lda sp,confsp               ;restore stack
1B92 9009         else  ;;odd } not sidflg      ;use initial stack
1B94 4868 6540      lda sp,=constack-<7*words>  ;proper stack pointer
1B98 4878 1B06      lda r7,=conloop     ;and process start
1B9C 3078 654C      sta r7,constack-words
1BA0 3008 620A      sta r0,sidflg       ;let CONFIG run now
                  endif

1BA4 6076       endroutine cslee2
1BA6 707F 00B0
1BAA 3078 FC04
1BAE 6056
1BB0 6046
1BB2 6036
1BB4 6026
1BB6 6016
1BB8 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 95
CONFIG.PLR;1    PAGE 4  IMP Configuration Code

FC00 0000               page Relcode

                ;reliability page timeout dispatch

567C 5680       reltim: relcon                  ;only has CONFIG code
567E 0000               0

                ;RELCON
                ;maintain IMP configuration stuff
                ; (1) check blocks in M2PBLK,V2PBLK,H2PBLK
                ; (2) check PIDs for RTC,modems,hosts,TIPs,VDH's
                ; (3) scan I/O devices for new ones to configure

5680 1076       routine relcon, uses r1-r5

     0001       .iif nz MISw            ;M/I bus code running
5682 48F0               clear ckill
5684 3078 6552

                ;(1) test each block in M2PBLK,V2PBLK,H2PBLK
                ; neg means not yet up

                ;loop for M2PBLK

5688 4828 8000    lda r2,=rup4us                ;initialize flooding mask
568C 4818 0010    lda r1,=nmd*words
                  repeat
5690 5049 635E      lda r4,m2pblk(-r1)
5694 9B12           ifnot minus                 ;modem exists.
5696 787C 0002        if byte phflag(r4) .nbit. #inhrst ;live line here
569A 4FF2
569C 8A03
569E 7429 0EB6          ior r2,bittab(r1)       ;so add its bit to mask
                      endif
56A2 4078 5A72        call mtest
56A6 8A07             if fail                   ;modem has troubles
56A8 4878 8000          lda r7,=sign
56AC 3479 635E          iorm r7,m2pblk(r1)      ;deallocate modem
56B0 4078 1BBA          call grrpkts            ;reroute its traffic
                      endif
56B4 4078 1B42        call csleep
                    endif
56B8 4E90           until r1 = =0               ;test each modem.
56BA 81EB
                  endrepeat
56BC 3028 72AA    sta r2,rupmsk                 ;store completed mask
```

```
                ;loop for V2PBLK
                ;calls null routine unless TIP and/or VDH package loaded

56C0 4818 0030    lda r1,=nrh*words             ;only for non-fakes
                  repeat
56C4 5049 6376      lda r4,v2pblk+<nfh*words>(-r1)      ;its param blk
56C8 9B15           ifnot minus                 ;some host here
56CA 7079 632E        lda r7,h2pblk+<nfh*words>(r1)     ;HOSTYP in host block
56CE 707F 0002        lda r7,hostyp(r7)         ;what host variety
56D2 40FF 12F0        call @chostest(r7)        ;call a host checker
56D6 8A0C             if fail                   ;something's busted
56D8 4878 8000          lda r7,=sign
56DC 3479 6376          iorm r7,v2pblk+<nfh*words>(r1)  ;mark host gone
56E0 4854               lda r5,r4
56E2 7079 632E          lda r7,h2pblk+<nfh*words>(r1)   ;HOSTYP in host block
56E6 707F 0002          lda r7,hostyp(r7)               ;what host variety
56EA 40FF 12FC          call @chosinit(r7)      ;and reinit it
                      endif
56EE 4078 1B42        call csleep
                    endif
56F2 4E90           until r1 = =0
56F4 81E8
                  endrepeat

                ;loop for H2PBLK

56F6 4818 0038    lda r1,=th*words
                  repeat
56FA 5049 6326      lda r4,h2pblk(-r1)
56FE 9B0A           ifnot minus                 ;host exists, is being used.
5700 4078 5AC4        call hotest
5704 8A05             if fail                   ;host has troubles
5706 4838 8000          lda r3,=sign
570A 3439 6326          iorm r3,h2pblk(r1)      ;forget this host
                      endif
570E 4078 1B42        call csleep
                    endif
5712 4E90           until r1 = =0
5714 81F3
                  endrepeat
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 97
CONFIG.PLR;1    PAGE 6  IMP Configuration Code

                ;(2) test dispatch structures

5716 4078 1B42    call csleep
571A 4818 0100    lda r1,=lbase
                  repeat
571E 5029 80C0      lda r2,base(-r1)
5722 7049 81C0      lda r4,mblks(r1)    ;;minus
5726 4878 0016      lda r7,#lccbase             ;base entries to look for
                    repeat
572A 562F 5650        if r2 = ccbase(-r7)       ;this entry matches
572E 810B
5730 40FF 5666          call @ccheck(r7)        ;call the checker
5734 8A07               if fail                 ;some trouble with it
5736 7079 80C0            set r7 = base(r1)     ;old dispatch for diagnosis
                          ;  Trap 1004,<;Clrd bad BASE dispatch tbl - page 96>
573A 4878 1064            set base(r1) = #bad
573E 3079 80C0
                        endif
5742 9002               break                   ;look no further
                      endif
5744 88F3             until loop                ;no more in CCBASE
                    endrepeat
5746 4F9E           if r1 .nbit. = HE           ;sleep after each group of 8.
5748 8A03
574A 4078 1B42        call csleep
                    endif
574E 4E90           until r1 = =0
5750 81E7
                  endrepeat
```

```
                ;(3) test io space

5752 48A6         lda r2,=useiol-words          ;do all of useio
                  repeat
5754 4858 001E      lda r5,= D15*words          ;start on next segment
                    repeat
5758 707A 5E98        if useio(r2) .bit. bittab(r5)     ;device exists.
575C 777D 0EB6
5760 9A0A
5762 1056               save r5
5764 A2D3               sll r5,3                ;each bit becomes 10} bytes
5766 725A 4116          add r5,iobase(r2)       ;to get full address
576A 4078 5864          call consub
576E 6056               restore r5              ;restore index
5770 4078 1B42          call csleep
                      endif
5774 49D2             sub r5,=2                 ;we will prefer f-bus devices
5776 8BF1             until minus
                    endrepeat
5778 49A2           sub r2,=2
577A 8BED           until minus
                  endrepeat

577C 48A8         lda r2,=nfh*words             ;for all fakes
                  repeat
577E 505A 636E      lda r5,v2pblk(-r2)          ;io bloc?
5782 9903           if even
5784 4078 5864        call consub
                    endif
5788 4EA0           until r2 = =0
578A 81FA
                  endrepeat
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 99
CONFIG.PLR;1    PAGE 8  IMP Configuration Code


                ;now count how many we're configured for

578C 4818 0010    lda r1,=nmd*words             ;max modems
                  repeat
5790 5079 635E      lda r7,m2pblk(-r1)
5794 8B02           until nminus                ;got last modem
5796 88FD           until loop                  ;no modems
                  endrepeat
5798 4A92         add r1,=words                 ;min of 1 modem
579A 3018 6324    sta r1,modems                 ;save this information
579E 4818 0030    lda r1,=nrh*words             ;max real hosts
                  repeat
57A2 5079 632E      lda r7,h2pblk+<nfh*words>(-r1)
57A6 8B02           until nminus                ;where have I seen
57A8 88FD           until loop                  ; this code before?
                  endrepeat
57AA 4A9A         add r1,=<nfh+1>*words         ;min of 1 real host
57AC 3018 6322    sta r1,hosts                  ;let others use this


     0001         .if nz MISw                   ;M/I code now to check CKILL
57B0 7018 6552    set bufflg,r1 = ckill         ;remember CKILL, tell Stage MM
57B4 3018 5E3E
57B8 4078 1B42    call csleep                   ;force wakeup here}
57BC 7618 0040    if r1 <> locflg               ;my last memory of CKILL
57C0 910C
57C2 3018 0040      sta r1,locflg               ;update it
57C6 3008 A550      unlock conlok
57CA 4878 002E      set @pid = =conpid          ;get everyone here}
57CE 30F8 00AC
57D2 E102           Trap 402,<;Changing buffer page allocation - page 98>
57D4 4048 0A1C      jsb r4,wst                  ;and back into Stage
                  endif
                .endc ;nz MISw

57D8 6006       endroutine relcon
```

```
                ;RRPKTS
                ;Reroute packets waiting for this modem
                ;packets will be held by TASK, since INCH word is 0
                ;modem param block pointer in R4
FC00 0400               page RutCode
41DA 1076       routine rrpkts, arg r4, local r1-r3, uses r5
41DC 1016
41DE 1026
41E0 1036
41E2 483C 0034     lda r3,=ssentq(r4)           ;start w/unacked but sent packets
                   repeat                       ;empty this q
41E6 7013             lda r1,(r3) ;;spriq sregq ;get head of q
41E8 48A1             lda r2,=1                 ;mark q empty
41EA 3023             sta r2,(r3)               ;  mark head
41EC 303B 0002        sta r3,epriq-spriq(r3)    ;point tail at head..
                      repeat                    ;give contents of q to tsk
41F0 4821                lda r2,r1              ;buffer to re-route
41F2 990A             until odd                 ;q empty, done
41F4 4078 1480           call wheorm,whi2m?whtsk        ;change ownership
41F8 0101
41FA 300A 09F0           sta r0,flushd-chain(r2)        ;hold off buffer timeout
41FE 7012                lda r1,(r2)           ;next buffer
4200 4078 14AA           call ttskput
4204 90F6             endrepeat
4206 4AB4             add r3,=4                 ;get to next q
4208 4E3C 003C     until r3 > =sregq(r4)
420C 8CED
                   endrepeat

420E 702C 4026     lda r2,snding+foo(r4)
4212 9A08          ifnot zero
4214 4078 1480           CALL WHEORM, WHI2M?WHTSK
4218 0101
421A 300A 09F0           SET FLUSHD-CHAIN(R2)
421E 4078 14AA           CALL TTSKPUT
                ENDIF
4222 702C 4028      LDA R2,later+foo(r4)              ;not to be flushed
4226 9A0E        IFNOT ZERO
4228 890A             if odd                    ;routing
422A 4B28 FFFE           and r2,=-2             ;clear odd bit
422E 783C 001A           ldab r3,modem(r4)
4232 703B 0EB6           lda r3,bittab(r3)      ;bit for this modem
4236 4078 55D2           call rupfls     ;;uses r5      ;flush our use on RUPQ
423A 9004             else
423C 4078 131E           CALL FLUSHB, WHI2M
4240 0100
                      endif
                    endif

                ;(OVER)
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 101
CONFIG.PLR;1    PAGE 10          IMP Configuration Code

4242 701C 001C     lda r1,maxchn(r4)
4246 A691          srl r1,1
4248 4991          sub r1,=1
424A 301C 0050     sta r1,slots(r4)
424E 6036        endroutine rrpkts
4250 6026
4252 6016
4254 6006

1BBA 1076        global grrpkts=rrpkts
1BBC 7078 40BE
1BC0 1076
1BC2 7078 00B4
1BC6 3078 FC00
1BCA 4078 41DA
1BCE 8A0A
1BD0 6076
1BD2 707F 00B0
1BD6 3078 FC00
1BDA 6076
1BDC 4FF0
1BDE 4007
1BE0 9007
1BE2 6076
1BE4 707F 00B0
1BE8 3078 FC00
1BEC 6006
FC00 0400
```

```
FC00 0000               page RelCode
                ;MHPCHK
                ;check that a given PID matches given param block
                ;R1 is PID, R4 param block
                ;look at MINPID(HINPID,BHPID), MOTPID(HOTPID,HBPID)

57DA 1076       routine mhpchk,arg r1,arg r4

57DC 7E1C 000F    if byte r1 <> minpid(r4) & byte r1 <> motpid(r4)
57E0 9107
57E2 7E1C 000E
57E6 9104
57E8 6076           fail return ;;hinpid hotpid ;PID mismatch
57EA 4FF0
57EC 4007
                  endif

57EE 6006       endroutine mhpchk


57F0 1076       routine hoschk,arg r4,arg r1

57F2 4844         lda r4,r4
57F4 9B0F         ifnot minus
57F6 4828 0038      lda r2,=th*words            ;for all host blocks
                    repeat
57FA 564A 6326        if r4 = h2pblk(-r2)       ;found this one
57FE 8108
5800 4078 57DA          call mhpchk             ;now check its PIDs
5804 8A04               if fail
5806 6076                 fail return           ;oops
5808 4FF0
580A 4007
                        endif
580C 6006               return                  ;found block and PIDs okay
                      endif
580E 4EA0             until r2 = =0             ;for all hosts blocks
5810 81F5
                    endrepeat
                  endif
5812 90FA         fail return                   ;missing host block

                endroutine hoschk
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 103
CONFIG.PLR;1    PAGE 12          IMP Configuration Code

5814 1076       routine rtcchk, arg r4, arg r1

5816 48A4         lda r2,=bsadil
5818 7038 5D5A    lda r3,usebus                 ;buses and rtc's that exist
                  repeat
581C 573A 0EC6      if r3 .bit. bittab+ H10(-r2)        ;RTC this bus?
5820 9A09
5822 707A 0320        lda r7,bsadrs(r2)         ;bus start
5826 707F 0008        lda r7,rtcpds(r7)         ;first switch register is pid num
582A A6F8             srl r7, D8                ;put it low order
582C 4E71             if r7 = r1                ;match base?
582E 8102
5830 6006               return                  ;all we need know
                      endif
                    endif
5832 88F5           until loop
                  endrepeat
                  Trap 414,<;RTC lost--assoc w/traps 10A and 108 (H) - page 102>
5834 E10C
02
5836 6076         fail return
5838 4FF0
583A 4007

                endroutine rtcchk

583C 1076       routine modchk, arg r1, arg r4

583E 4844         lda r4,r4                     ;status not carried over subroutine entry
5840 8B04         if minus                      ;dead parameter block
5842 6076           fail return                 ;so don't do anything
5844 4FF0
5846 4007
                  endif
5848 9B0D         ifnot minus
584A 4828 0010      lda r2,=nmd*words           ;find us in M2PBLK
                    repeat
584E 564A 635E        if r4 = m2pblk(-r2)
5852 8106
5854 4078 57DA          call mhpchk             ;check PIDs
5858 8A02               if fail
585A 90F4                 fail return           ;not one of ours
                        endif
585C 6006               return                  ;okay PID dispatch
                      endif
585E 4EA0             until r2 = =0
5860 81F7
                    endrepeat
                  endif
5862 90F0         fail return                   ;no such modem

                endroutine modchk
```

```
                ;subroutine that configures any device
                ;calls device configuration routine with device number in r2,
                ;device address in r5

5864 1076       routine consub,arg r5,local r2,uses r1,uses r3-r4
5866 1026

5868 7035         lda r3,(r5)
586A 4823         lda r2,r3
586C A6B7         srl r3,7
586E 4B38 007E    and r3,= H7E          ;chuck magic modem bit
5872 4B28 00FF    and r2,= HFF
5876 4E38 0020    if r3 < =ldevtab      ;see if it's good.
587A 8203
587C 40FB 5884      call @devtab(r3)    ;call device configuration routine
                  endif

5880 6026       endroutine consub
5882 6006

5884            table devtab
5884 07D6         rsucceed      ;0 - no such device type
5886 58A4         cmodem        ;1 - Low Speed Modem (LSM)
5888 5936         chost         ;2 - Host Local Controller or HoST (HLC, HST)
588A 07D6         rsucceed      ;3 - Checksum/Block Transfer (CBT)
588C 07D6         rsucceed      ;4 - Synchronous Line Interface (SLI)
588E 58A4         cmodem        ;5 - High Speed Modem (HSM)
5890 07D6         rsucceed      ;6
5892 07D6         rsucceed      ;7
5894 07D6         rsucceed      ;8
5896 07D6         rsucceed      ;9
5898 07D6         rsucceed      ;A
589A 07D6         rsucceed      ;B
589C 07D6         rsucceed      ;C - Adapter, MultiLine controller (AML)
589E 07D6         rsucceed      ;D
58A0 07D6         rsucceed      ;E
58A2 07D6         rsucceed      ;F
                endtable devtab
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 105
CONFIG.PLR;1    PAGE 14          IMP Configuration Code

                ;CMODEM
                ;Configure a modem parameter block; build it if needed.
                ; enter with r2 device number from interface
                ; r5-interface address

58A4 1076       routine cmodem, arg r2, arg r5
58A6 7E28 12EC    if byte r2 >= vd.clp          ;this is a VDH
58AA 9204
58AC 40F8 12EE      call @icvdh                 ;configure vdh here.
58B0 9042         else
58B2 4EA8           if r2 < =nmd                ;number too large?
58B4 8240
58B6 A2A1             sll r2,1
58B8 704A 635E        lda r4,m2pblk(r2)         ;current param blk
58BC 8B3A             if minus                  ;needs initializing
58BE 4078 5A3C          call bldblk,modlen
58C2 00F0
58C4 9A35               if success
58C6 304A 635E            sta r4,m2pblk(r2)
58CA 382C 001A            stab r2,modem(r4)
58CE 4078 5BC6            call tst2pid,m2i,i2m
58D2 205A
58D4 1D3A
58D6 9A2B               if success
     0001       .lif nz MISw
58D8 4078 5B98            call cmichk
58DC 305C 0020            sta r5,iobloc(r4)
58E0 48B1                 lda r3,=1
58E2 487C 004A            lda r7,=smiq(r4)
58E6 307C 004C            sta r7,emiq(r4)
58EA 3037                 sta r3,(r7)
58EC 487C 0034            lda r7,=ssentq(r4)
58F0 307C 0036            sta r7,esentq(r4)
58F4 3037                 sta r3,(r7)
58F6 303C 0038            sta r3,spriq(r4)
58FA 303C 003C            sta r3,sregq(r4)
58FE 7078 63CA            lda r7,junk
5902 307C 0044            sta r7,filling(r4)
5906 307C 0046            sta r7,nxtbf(r4)
590A 4078 1CDC            call linei2           ;finish the initializing
590E 300C 4048            unlock lmiq(r4)       ;finally, unlock locks
5912 300C 4000            unlock lockm(r4)
5916 300C 4042            unlock m2iloc(r4)
591A 300C 4024            unlock i2mloc(r4)
591E 300C 40B8            unlock rtimrl(r4)
5922 300C 4092            unlock dlock(r4)
5926 4078 1CF4            call killin           ;lines start dead
592A 9002               else
                          Trap 1006,<;BASE/MBLKS wrong for M2I/I2M - page 104>
592C E206
                        endif
                      endif
592E 9003           else
5930 4078 59BC        call devinuse
                    endif
                  endif
```

5934 6006 endroutine cmodem

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 106
CONFIG.PLR;1    PAGE 15          IMP Configuration Code

                ;CHOST
                ;Configure a host parameter block; build new one if needed.
                ; r2 contains host parameter block pointer index
                ; r5 has device address

5936 1076       routine chost, arg r2, arg r5
5938 4AA4         add r2,=nfh                   ;put fakes at front
593A 4B28 00FF    and r2,= HFF
593E 4E28 001C    if r2 < =th                   ;legal host?
5942 820A
5944 A2A1           sll r2,1
5946 704A 6326      lda r4,h2pblk(r2)           ;param block this host
594A 8B04           if minus                    ;in need of configuring
594C 4078 5958        call bldhst               ;build host block (shared with VDH).
5950 9003           else
5952 4078 59BC        call devinuse
                    endif
                  endif
5956 6006       endroutine chost
```

```
                ;BLDHST
                ;Build a host parameter block, initialize it if we get one.

5958 1076       routine bldhst, arg r2, arg r4-r5
595A 4078 5A3C    call bldblk,holen             ;assign a parameter block
595E 0070
5960 9A2D         if success                    ;got one
5962 304A 6326      sta r4,h2pblk(r2)           ;set H2PBLK
5966 382C 001D      stab r2,holhn(r4)           ;local host index
596A 4078 5BC6      call tst2pid,hi,ih          ;set up PID dispatches
596E 2B46
5970 18DC
5972 8A03           if fail                     ;PID conflicts
5974 E61A             Trap 3032.<;BASE/MBLKS wrong for HI/IH - page 106>
5976 9022           else
5978 4838 26E8        lda r3,=higo
597C 303C 0004        sta r3,hilo(r4)           ;initial HI dispatch
5980 48A4             lda r2,=hosvdh            ;assume a VDH
5982 305C 0020        sta r5,iobloc(r4)         ;real i/o hardware?
5986 8B08             if minus
     0001           .if nz MISw
5988 4078 5B98           call cmichk
598C 4838 0100           lda r3,=hreset         ;;data ihledr trledr
5990 303D 0006           sta r3,statih(r5)      ;reset input side
5994 48A0                lda r2,=hosreal        ;real host
                      endif
5996 781C 001D        ldab r1,holhn(r4)         ;host number (offset)
599A A691             srl r1,1                  ;"hardware" host number to r1
599C 4994             sub r1,=nfh               ;fakes must be big
599E 8B06             if minus
59A0 48A2                lda r2,=hosfake        ;this is software host
59A2 4878 0080           lda r7,= H80           ;new format flag
59A6 387C 001C           stab r7,homode(r4)     ;start new
                      endif
59AA 381C 0029        stab r1,hihost+1(r4)      ;save our host address
59AE 302C 0002        sta r2,hostyp(r4)         ;and REAL/FAKE/VDH flag
59B2 300C 4000        unlock lockhi(r4)         ;unlock HI
59B6 4078 59FC        call ihsini               ;init IH side
                    endif
                  endif
59BA 6006       endroutine bldhst
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 108
CONFIG.PLR;1    PAGE 17          IMP Configuration Code

59BC 1076       routine devinuse, arg r4-r5

59BE 765C 0020    if r5 <> iobloc(r4)           ;this isn't main device
59C2 911C
59C4 703C 0022      lda r3,altio(r4)            ;any spare device?
59C8 9A04           if zero } r3 = iobloc(r4)   ;no
59CA 763C 0020
59CE 8113
59D0 4838 00FE        lda r3,= HFE
59D4 4873             lda r7,r3
59D6 733D 0006        and r3,statim(r5) ;;statih
59DA 737D 000C        and r7,statom(r5) ;;statoh ;check PIDs
59DE 7E3C 000F        if byte r3 = minpid(r4) & byte r7 = motpid(r4)
59E2 8107
59E4 7E7C 000E
59E8 8104
59EA 305C 0022          sta r5,altio(r4)        ;become the spare interface
59EE 9002              else
59F0 E205               Trap 1005,<;Dbld PID intfc differs CALL MAINT (H) - page 107>
7
                       endif
59F2 9004             else
59F4 4E53              if r5 <> r3              ;we aren't spare either?
59F6 9102
59F8 E203               Trap 1003,<;2 interfaces, one device (CALL MAINT) - page 107>
7
                       endif
                     endif
                   endif

59FA 6006       endroutine devinuse
```

```
                ;IHSINI
                ;initialize IH software and hardware status
                ;uses r1,3, r4 host, r5 iobloc

59FC 1076       routine ihsini,arg r4-r5,uses r1,uses r3

59FE 48B1         lda r3,=1
5A00 487C 0060    lda r7,=shq(r4)
5A04 307C 0056    sta r7,ihwq(r4)               ;make IHWQ valid
5A08 307C 0062    sta r7,ehq(r4)                ;empty SHQ/EHQ
5A0C 3037         sta r3,(r7)
5A0E 487C 0064    lda r7,=shpq(r4)              ;empty SHPQ/EHPQ
5A12 307C 0066    sta r7,ehpq(r4)
5A16 3037         sta r3,(r7)
5A18 707C 0002    ifnot hostyp(r4)      ::hosreal ;got real hardware
5A1C 8A05
5A1E 4838 1100      lda r3,=hready+hreset       ;;data ihledr trledr
5A22 303D 000C      sta r3,statoh(r5)           ;reset it
                  endif
5A26 703C 4054    lda r3,ihgoin+foo(r4)         ;clear ihgoin for reset
5A2A 4838 1A56    lda r3,=ihidle
5A2E 303C 0050    sta r3,ihlo(r4)               ;init IHLO dispatch
5A32 300C 405C    unlock lockih(r4)
5A36 300C 404E    unlock ihloc(r4)

5A3A 6006       endroutine ihsini
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 110
CONFIG.PLR;1    PAGE 19          IMP Configuration Code

                ;BLDBLK
                ;(allocate) and clear a parameter block
                ; call bldblk,<block length> ;(multiple of  H10)
                ;call with old block address (or -1) in r4
                ;returns new block address in r4 if success
                ;clobbers r1

5A3C 1076       routine bldblk, arg r4, inline r1, local r1
5A3E 1016
5A40 6017
5A42 307E 0002

5A46 4B48 7FFF    and r4,= H7FFF                ;clear inactive bit
5A4A 890D         if odd                        ;not allocated yet
5A4C 7048 6524      lda r4,dynxt                ;next available block address
5A50 4874           lda r7,r4
5A52 4A71           add r7,r1                   ;new end of DYBLKS
5A54 8B06           if minus    ;;dyblke        ;overflow
5A56 E207             Trap 1007,<;BLDBLK: dynamic blocks area full - page 109>
5A58 6016             fail return
5A5A 6076
5A5C 4FF0
5A5E 4007
                    endif
5A60 3078 6524      sta r7,dynxt
                  endif
5A64 4A14         add r1,r4                     ;now clear the block
5A66 48F0         lda r7,=0
                  repeat
5A68 1071           sta r7,(-r1)
5A6A 4E14           until r1 = r4               ;do all the block
5A6C 81FE
                  endrepeat

5A6E 6016       endroutine bldblk
5A70 6006
```

```
                ;MTEST
                ;Test a running modem
                ; enter with r4=par block, r1=modem number
                ; check the block for internal consistancy

5A72 1076       routine mtest, arg r1, arg r4, local r2
5A74 1026

5A76 7E1C 001A    if byte r1 = modem(r4)
5A7A 8123
5A7C 4078 5B48      call tst2dev                ;check our interface(s)
5A80 8A0B           if fail
5A82 4855             lda r5,r5                 ;got a spare?
5A84 8A06             if zero                   ;no
5A86 E110               Trap 420,<;modem lost on Prim bus CALL MAINT (H) - page 110>

5A88 6026               fail return
5A8A 6076
5A8C 4FF0
5A8E 4007
                    endif
                    Trap 424,<;swapping modem interfaces CALL MAINT (H) - page 110>
5A90 E114
110
5A92 305C 0020      sta r5,iobloc(r4)
                  endif
5A96 7075         lda r7,(r5)                   ;get device type
5A98 4D78 0100    eor r7,=modid                 ;clear device type
5A9C 4F78 7B00    if r7 .nbit. = HFF00?modid?hmodid?magmod    ;some modem
5AA0 8A10
5AA2 4B78 00FF      and r7,= HFF                ;just modem number
5AA6 7E78 12EC      if byte r7 < vd.clp         ;not a VDH modem
5AAA 820B
5AAC A2F1             sll r7,1
5AAE 4E71             if r7 = r1                ;the right device
5AB0 8108
5AB2 4078 5BC6          call tst2pid,m2i,i2m    ;see about its PIDs
5AB6 205A
5AB8 1D3A
5ABA 9A03               if success              ;all checks out
5ABC 6026                 return
5ABE 6006
                        endif
                      endif
                    endif
                  endif
                endif
5AC0 E409         Trap 2011,<;scrambled modem parameter block - page 110>
5AC2 90E3         fail return

                endroutine mtest
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 112
CONFIG.PLR;1    PAGE 21         IMP Configuration Code

                ;HOTEST
                ;Test running host
                ; r1=index into h2pblk - r4=par block - r7=return

5AC4 1076       routine hotest,arg r1,arg r4,uses r1-r3,uses r5

5AC6 4078 5B48    call tst2dev                  ;check our interface(s)
5ACA 8A2E         if fail
5ACC 4855           lda r5,r5                   ;got a spare?
5ACE 8A05           if zero                     ;no
                      Trap 421,<;Host lost on Prim buss CALL MAINT (H) - page 111>
5AD0 E111

5AD2 6076             fail return
5AD4 4FF0
5AD6 4007
                    endif
                    Trap 425,<;swapping host interfaces CALL MAINT (H) - page 111>
5AD8 E115
11
5ADA 4828 0078      lda r2,=sec3                ;allow lots of time for reset
5ADE 4838 26D2      lda r3,=hiidle              ;will become hilo
5AE2 707C 4000      lock lockhi(r4)             ;lock hi
5AE6 9AFE
5AE8 302C 000C      sta r2,hitt(r4)             ;timer to allow ready-line flap
5AEC 48F4           lda r7,=hninit              ;for hihd
5AEE 387C 001F      stab r7,hihd(r4)            ;new state
5AF2 303C 0004      sta r3,hilo(r4)             ;and dispatch
5AF6 305C 0020      sta r5,iobloc(r4)           ;and replace interface
5AFA 300C 4000      unlock lockhi(r4)           ;unlock hi
5AFE 4828 0100      lda r2,=hreset              ;bit to reset hardware
5B02 4838 192C      lda r3,=ihdead              ;new dispatch
5B06 707C 404E      lock ihloc(r4)              ;lock hardware
5B0A 9AFE
5B0C 707C 405C      lock lockih(r4)             ;lock ih side too
5B10 9AFE
5B12 707C 4052      lda r7,ihtt+foo(r4)         ;no timing
5B16 303C 0050      sta r3,ihlo(r4)             ;new dispatch
5B1A 302D 000C      sta r2,statoh(r5)           ;reset it
5B1E 300C 405C      unlock lockih(r4)           ;done host stuff
5B22 300C 404E      unlock ihloc(r4)
                  endif
5B26 4078 5BC6    call tst2pid,hi,ih            ;check our dispatches
5B2A 2B46
5B2C 18DC
5B2E 9A0B         if success
5B30 4879 FFF8      lda r7,=-nfh*words(r1)      ;convert H2PBLK index
5B34 A6F1           srl r7,1                    ;get word offset
5B36 4B78 00FF      and r7,= HFF                ;just right byte
5B3A 7575           eor r7,(r5) ;;statd         ;match hardware ID
5B3C 4D78 0200      eor r7,=hostid
5B40 8A02           if zero                     ;it's a match
5B42 6006             return
                    endif
                  endif
5B44 E628         Trap 3050,<;scrambled host parameter block - page 111>
```

```
endroutine hotest
```

```
Pluribus IMP J301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 113
CONFIG.PLR;1    PAGE 22          IMP Configuration Code

                ;TST2DEV
                ;Test a doubled device (IOBLOC and ALTIO); call with
                ; param blk in R4, return with io address in R5
                ;Fail return -> main device is gone; return alternate in R5
                ;Succeed if main interface (and spare if any) is in USEIO

5B48 1076       routine tst2dev,arg r4,arg m1,arg m3,result r5,local r1
5B4A 1016

     0001       .if nz MISw                     ;special testing in MI machines
                  repeat
5B4C 705C 0020      lda r5,iobloc(r4)           ;main device
5B50 701C 0022      lda r1,altio(r4)            ;and spare
5B54 4F58 1000      until r5 .bit. = H1000 } r1 .nbit. = H1000
5B58 8A0A
5B5A 4F18 1000
5B5E 9A07
5B60 301C 0020      sta r1,iobloc(r4)           ;swap to F device
5B64 305C 0022      sta r5,altio(r4)            ;if that helps
5B68 E103           Trap 403,<;Swapping to F device - page 112>
5B6A 90F1         endrepeat
5B6C 4811         lda r1,r1                     ;test ALTIO
                .iff ;nz MIsw
                  lda r5,iobloc(r4)             ;get main device
                  lda r1,altio(r4)              ;test alternate first
                .iftf ;nz MISw          ;must get here with ZERO test on ALTIO
5B6E 9A07         ifnot zero
5B70 4078 5BA8      call vfindev                ;does it exist?
5B74 8A04           if fail                     ;nope)
5B76 351C 0022        eorm r1,altio(r4)
                      Trap 422,<;spare interface disappeared CALL MAINT (H) - page 112>
5B7A E112
e 112
                    endif
                  endif
5B7C 4815         lda r1,r5                     ;now test main device
5B7E 4078 5BA8    call vfindev                  ;it has to exist
5B82 8A07         if fail                       ;it's gone
5B84 705C 4022      lda r5,altio+foo(r4)        ;try spare
5B88 6016           fail return                 ;return it to caller
5B8A 6076
5B8C 4FF0
5B8E 4007
                  endif
                .ift ;nz MISw
5B90 4078 5B98    call cmichk
                .endc ;nz MISw

5B94 6016       endroutine tst2dev
5B96 6006
```

```
     0001        .if nz MISw

                 routine cmichk, arg r5, nosave
5B98 4855          lda r5,r5
5B9A 8B06          if minus                        ;if it's really hardware
5B9C 4F58 1000       if r5 .nbit. = H1000          ;main device not on F-bus
5BA0 8A03
5BA2 3008 6552         set ckill                   ;must not use 0 memory buffers
                     endif
                   endif
5BA6 4807          endroutine cmichk

                 .endc   ;nz MISw
                 ;VFINDEV
                 ;Find a hardware device or VDH "device"
                 ;Calls FINDEV, then checks V2PBLK if fail

5BA8 1076        routine vfindev,arg r1

5BAA 4078 11C4     call findev
5BAE 8A0B          if fail                         ;real device exists
5BB0 4878 0038       lda r7,=th*words              ;total hosts, length of v2pblk
                     repeat
5BB4 561F 636E         if r1 = v2pblk(-r7)         ;found VDH device
5BB8 8102
5BBA 6006                return
                       endif
5BBC 88FC              until loop
                     endrepeat
5BBE 6076            fail return
5BC0 4FF0
5BC2 4007
                   endif

5BC4 90FB        endroutine vfindev
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 115
CONFIG.PLR;1    PAGE 24          IMP Configuration Code

                ;TST2PID
                ;test both PIDs of given device
                ; R5 enters with device address, R4 is param blk
                ; inline are dispatches for input, output PIDs
                ; sets up MINPID(HINPID) and MOTPID(HOTPID)

5BC6 1076       routine tst2pid,arg r4-r5,inline r3,inline r2,local r1
5BC8 1016
5BCA 6037
5BCC 6027
5BCE 307E 0002

5BD2 1026         save r2                       ;need this later
5BD4 4824         lda r2,r4                     ;param blk for MBLKS
5BD6 701D 0006    lda r1,statim(r5)     ;;statih ;input PID
5BDA 4B18 00FE    and r1,= HFE
5BDE 4078 1174    call inbas2                   ;check this dispatch
5BE2 9A10         if success                    ;so far, so good
5BE4 381C 000F      stab r1,minpid(r4)  ;;hinpid ;remember in param blk
5BE8 6036           pop r3                      ;get output dispatch
5BEA 701D 000C      lda r1,statom(r5)   ;;statoh ;now output PID
5BEE 4B18 00FE      and r1,= HFE
5BF2 4078 1174      call inbas2                 ;and check it
5BF6 9A05           if success                  ;all's well
5BF8 381C 000E        stab r1,motpid(r4)        ;;hotpid ;remember it too
5BFC 6016             return
5BFE 6006
                    endif
5C00 9002           else
5C02 6076             pop r7                    ;fix up stack
                    endif
5C04 6016         fail return                   ;signal caller
5C06 6076
5C08 4FF0
5C0A 4007

                endroutine tst2pid
```

```
.INSERT "FASTTO"
    .INSRT FASTTO
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 117
FASTTO.PLR;1    PAGE 1  Fast Timeout (25 ms)

                        .stitle Fast Timeout (25 ms)

FC00 0400               page PkgCode            ;following code on package page.

4256            table faspok            ;table of pids to poke each 25 ms.
4256 0006         dsppid
4258 0014         ttypid
425A 0010         bktpid
425C 0016         ihtpid
425E 0022         ttopid
4260 0044         rutpi                 ;poke these each 25 ms
                endtable faspok

                ;CTOHOT
                ;Handles 25 millisecons RTC PID
                ;Performs various periodic checks, some at 125 ms intervals
                ;Pokes STO (slow timeout) every 625 ms

4262 1076       routine ctohot, arg r4

4264 7078 00B0    setmap m2,maprel              ;check out rely page.
4268 3078 FC04
426C 701C 0172    lda r1,pidget-2(r4)           ;get addr of this clock
4270 4A94         add r1,=rtcadd-pidrcl
4272 4078 4426    call cknice           ;check nice stop stuff
4276 7031         lda r3,(r1)                   ;poke whoever pid'ed us
4278 48D1         lda r5,=1                     ;args for CKCLOCK and us
427A 4078 437A    call ckclock                  ;check main/spare clock
427E 9A79         if success                    ;we're main clock
4280 3258 62DA      addm r5,time
4284 3258 62DE      addm r5,sync
4288 3258 8094      addm r5,m2#sytime
428C 8403           if carry                    ;carry out.
428E 3258 9EAA        addm r5,m2#sytim2         ;on rely)
                    endif
4292 7078 62D2      setmap m2,watm1             ;maintain core copy of lights
4296 3078 FC04
429A 70A8 62D0      lda r2,@watch1
429E 8001           ifnot quit
                    endif
42A0 3028 6000      sta r2,watch0
42A4 7078 62D6      setmap m2,watm2
42A8 3078 FC04
42AC 70A8 62D4      lda r2,@watch2
42B0 8001           ifnot quit
                    endif
42B2 3028 6002      sta r2,watch0+2
42B6 7078 00D2      setmap m2,mapv2
42BA 3078 FC04
```

```
42BE 7078 620A      lda r7,sidflg               ;rerun con now?
42C2 9B06           ifnot minus
42C4 4838 002E        lda r3,=conpid
42C8 30B8 00AC        sta r3,@pid
42CC 9050           else
42CE 4078 43E0        call rxmchek              ;check modems for retransmissions
42D2 7028 A4A6        lock r2,cycle
42D6 9AFE
42D8 4928 2000        sub r2,= H2000
42DC 8B1C             if minus                  ;medium timeout every 128 ms
42DE 4A28 9C00          add r2,= H2000- H400+sign
42E2 9B07               ifnot minus             ;time for slow timeout.
42E4 4828 9001            lda r2,= H9001
42E8 4878 0042            lda r7,=stopid
42EC 30F8 00AC            sta r7,@pid
                        endif

              ;do various line state stuff every medium timeout (max.)

42F0 7018 6324          lda r1,modems
                        repeat
42F4 5049 635E            lda r4,m2pblk(-r1)
42F8 9B07                 ifnot minus
42FA 1016                   save r1
42FC 4078 44D8              call phdedl         ;DEDL at medium to
4300 4078 44B2              call mpoke          ;MPOKE at medium to
4304 6016                   restore r1
                          endif
4306 4E90                 until r1 = =0
4308 81F6
                        endrepeat

430A 48F1               lda r7,=1
430C 3278 6520          addm r7,tikcnt          ;bump line clock
4310 4078 447E          call hpoke              ;keep hosts running
                      endif
4314 3028 A4A6        unlock r2,cycle

4318 48A8             lda r2,=nfh*words
                      repeat
431A 505A 636E          lda r5,v2pblk(-r2)
431E 9909               ifnot odd
4320 787D 000F            ldab r7,bhpid(r5)
4324 30F8 00AC            sta r7,@pid
4328 787D 000E            ldab r7,hbpid(r5)
432C 30F8 00AC            sta r7,@pid
                        endif
4330 88F5               until loop
                      endrepeat
4332 489C             lda r1,=lfaspok           ;how many pids to poke
                      repeat
4334 5079 4256          lda r7,faspok(-r1)      ;next pid
4338 30F8 00AC          sta r7,@pid             ;poke it
433C 88FC               until loop
                      endrepeat
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 119
FASTTO.PLR;1    PAGE 3  Fast Timeout (25 ms)

433E 4858 63BC          lda r5,=srq               ;append srq to stq
4342 7015               lda r1,(r5)                       ;unlocked is okay
4344 9914               ifnot odd                 ;srq has stuff.
4346 7078 A3B6            lock ltq
434A 9AFE
434C 3098 63BA            sta r1,@etq
4350 7028 63BE            lda r2,erq
4354 48F1                 lda r7,=1
4356 3075                 sta r7,(r5)
4358 3058 63BE            sta r5,erq
435C 3028 63BA            sta r2,etq              ;put secondary queue on end of task q
4360 3008 A3B6            unlock ltq
4364 4878 0040            lda r7,=task
4368 30F8 00AC            sta r7,@pid             ;poke TASK.
                        endif
                      endif
436C 4078 127E        call do25.6                 ;do OPSYS 25.6 ms timeout
                    endif
4370 7078 00D2      setmap m2,mapv2
4374 3078 FC04

4378 6006         endroutine ctohot

                  ;Create global entry to CTOHOT if needed

     0001         .if z LBig                     ;different Pkg page
                    global tohot=ctohot
                  .iff
     4262           tohot=ctohot
                  .endc
```

```
                ;CKCLOCK
                ;checks two real-time clocks against each other
                ;called with R1/address of clock whose PID we got,
                ;R5/ 1

437A 1076       routine ckclock, arg r1, arg r5, uses r4

437C 48C3         lda r4,=3             ;constant for timers
437E 7618 62E8    if r1 <> clock
4382 911C
4384 7078 62E6      lda r7,clk2up       ;system know we're here?
4388 4F78 FFFC      if r7 .bit. =-1?3
438C 9A01
                  ;    Trap 411,<;backup clock working again (H) - page 119>
                    endif
438E 7078 A2EA      lock clklok
4392 9AFE
4394 4E18 F006      if r1 <> =besclk
4398 910E
439A 3158 62E4        subm r5,clk1up    ;main clock still happy?
439E 9B0A             ifnot minus
43A0 3048 62E6          sta r4,clk2up
43A4 3008 A2EA          unlock clklok
43A8 3258 62DC          addm r5,timea   ;bump alternate timer
43AC 6076               fail return
43AE 4FF0
43B0 4007
                      endif
                      Trap 410,<;Main clock stppd swtchd to bkup RTC (H) - page 119>
43B2 E108
19
                    endif
43B4 3018 62E8      sta r1,clock        ;make us be main clock
43B8 900F          else
43BA 7078 9D5A      lda r7,m2#usebus
43BE 4BF3           and r7,=<1_<bsadil_-1>>-1
43C0 4B7F FFFF      and r7,=-1(r7)
43C4 8A02           if zero             ;only one clock in system
43C6 6006             return
                    endif
43C8 7078 A2EA      lock clklok
43CC 9AFE
43CE 3158 62E6      subm r5,clk2up
43D2 8A02           if zero
43D4 E10A             Trap 412,<;no working backup RTC (H) - page 119>
                    endif
                  endif
43D6 3048 62E4    sta r4,clk1up
43DA 3008 A2EA    unlock clklok

43DE 90F4       endroutine ckclock
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 121
FASTTO.PLR;1    PAGE 5  Fast Timeout (25 ms)

                ;RXMCHEK: check modems for retransmissions to go
                ;loops through all modems and compares sent time
                ;of first buffer on SENTQ with current time to
                ;determine if we need to set I2MPOK to flag retransmission

43E0 1076       routine rxmcheck, uses r1/r2/r4/r5
43E2 48D0          set r5 = #0
                   repeat
43E4 604D 635E        lda r4,m2pblk(r5)+        ;get modem block
43E8 9B1B             ifnot minus               ;there is a modem here
43EA 707C 0034           set r7 = ssentq(r4)    ;check to see if anything
43EE 9918                ifnot odd              ;something there to look at
43F0 707C 4000              lock lockm(r4)      ;now lock and look
43F4 9AFE
43F6 702C 0034                 lda r2,ssentq(r4)
43FA 9911                      ifnot odd         ;still something there
43FC 4078 1512                    call unpack   ;get to buffer
4400 9A0E                         ifnot fail    ;succesfully
4402 7079 009A                       lda r7,st(r1)      ;sent time
4406 727C 0032                       add r7,mrtime(r4)  ;retrans interval
440A 71F8 62E8                       sub r7,@clock      ;minus time now
440E 8B03                            if minus           ;retransmit, then
4410 4078 14E8                          call pokem      ;set flag and poke
                                     endif
4414 7078 00D2                       setmap m2,mapv2    ;restore map
4418 3078 FC04
                                  endif
                               endif
441C 3004                   unlock (r4) ;;lockm
                         endif
                      endif
441E 7658 6324       until r5 = modems          ;until we've checked all
4422 81E1
                   endrepeat
4424 6006       endroutine rxmcheck
```

```
                ;check and do nice/panic stop as necessary
4426 1076       routine cknice
4428 7078 6516    if nsftof = # HOBAD           ;should we stop machine?
442C 4E78 0BAD
4430 8126
4432 7079 0002      lda r7,2(r1)
4436 A6F8           srl r7, H8              ;get 25.6ms clock PID
4438 30F8 00AC      sta r7,@PID             ;poke it to get everyone here
443C 7018 651C      lda r1,nsrtf                    ;get nice-stop flag.
4440 9B02           ifnot minus
4442 4994             sub r1,=4             ;make panic look like nice stop
                    endif
4444 4A92           add r1,=2   ;;nsrtf             ;check flag for dispatch.
4446 9A0B           ifnot zero
4448 9B06             ifnot minus                   ;halt us.
444A 4818 001E          lda r1,=%lvl4+%servc      ;shut off level 4
444E 3011               sta r1,(r1)               ;by pointing it at itself
4450 0889               inh .L4+.L1
4452 0000               hlt
                      endif
4454 7048 6518        lda r4,nspc                   ;who started this
4458 4008 0A08        jmp ws                        ;restart
                    endif                           ;its a reload request
445C 7028 651E      lda r2,nstlin
4460 9A0A           ifnot zero              ;line number set, so hold on one line
4462 49A1             sub r2,=1
4464 A2A1             sll r2,1              ;make line numbe into offset
4466 704A 635E        lda r4,m2pblk(r2) ;this parameter block
446A 9B05             ifnot minus           ;if we have such modem
446C 707C 0020          set m2#rlddev = iobloc(r4)      ;reload from it
4470 3078 9EAC
                      endif
                    endif
4474 7048 6518      lda r4,nspc             ;remember who started all this
4478 4008 1308      jmp reload              ;now force a reload
                  endif
447C 6006       endroutine cknice
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 123
FASTTO.PLR;1    PAGE 7  Fast Timeout (25 ms)

                ;****** keep host running

447E 1076       routine hpoke, uses r1, uses r3-r5
4480 7018 6322    lda r1,hosts          ;all hosts
4484 4998         sub r1,=nfh2          ;just real hosts
                  repeat
4486 5049 632E      lda r4,h2pblk+nfh2(-r1)
448A 9B12           ifnot minus
448C 707C 0002        ifnot hostyp(r4)  ;;hosreal ;if a real host
4490 8A0F
4492 783C 001F          ldab r3,hihd(r4)        ;reset in progress?
4496 4EB4               if r3 <> =hninit
4498 910B
449A 4838 1000            lda r3,=hready        ;;data trledr ihledr
449E 705C 0020            lda r5,iobloc(r4)
44A2 303D 000C            sta r3,statoh(r5)
44A6 743C 0024            ior r3,hiloop(r4)
44AA 303D 0006            sta r3,statih(r5)
                        endif
                      endif
                    endif
44AE 88EC           until loop
                  endrepeat
44B0 6006       endroutine
```

```
                ;MPOKE
                ;poke modem interfaces to hold off watchdog
                ;also maintains low-order address bits (both directions)
                ;and interface loop/crosspatch state (input side)
                ;R4 enters with parameter block address

44B2 1076       routine mpoke, arg r4,uses r5

44B4 705C 0020    lda r5,iobloc(r4)             ;modem address
44B8 707C 001E    lda r7,mloop(r4)
44BC 4B78 C000    and r7,=mloopi+mloope
44C0 307C 001E    sta r7,mloop(r4)              ;loop state
44C4 307D 0006    sta r7,statim(r5)     ;;neth  ;hold loop state, low address
44C8 48F0         lda r7,=0
44CA 307D 000C    sta r7,statom(r5)     ;;neth  ;hold hardware wdt, low address
44CE 787C 000F    ldab r7,minpid(r4)
44D2 30F8 00AC    sta r7,@pid

44D6 6006       endroutine mpoke
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 125
FASTTO.PLR;1    PAGE 9  Fast Timeout (25 ms)

                ;Package page timeout table if needed

                getpage Warm.....,$mapcd          ;get page maps for tests
                getpage PkgCode.....,$mappk
                getpage RutCode.....,$maprt

     0000       .if nz $mappk-$mapcd

                pkgtab: 0               ;Config code entry-
                .if z $mappk-$maprt     ;pick up routing timeouts too?
                        rupqck
                .endc ;;z $mappk-$maprt
                        0               ;spare
                        0               ;end of timeout table

                .endc ;;nz $mappk-$mapcd
```

```
.INSERT "STO"
    .INSRT STO
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 127
STO.PLR;1       PAGE 1  Slow Timeout Dispatch

                .stitle Slow Timeout Dispatch

.comnt |
STO
Slow Timeout code central dispatch. (640 ms.)
Calls all system slow timeout routines.
Each code page has a list of routines pointed at by TOPNTR. The
first routine is a configuration routine and is skipped by STO.
When STO finds a list entry zero, it changes pages and proceeds on the
next logical page.
Poked every slow tick by Fast Timeout.
|
                        Page LCode


                ;compatibility crock}
1BEE 4868 6500  tossx:  lda sp,=stostack-<2*words>
1BF2 9014               br stnxt

                ;process sto,pid=stopid

                stoloop:                        ;start here first time around
                repeat                          ;forever
1BF4 4890         lda r1,=0
                  repeat                        ;for all code pages
1BF6 6079 00B0      lda r7,lmap(r1)+
1BFA 9915           ifnot odd                   ;page exists
1BFC 3078 FC00        sta r7,%map0
1C00 7038 40BC        lda r3,topntr
1C04 9A10             ifnot zero                ;there is a table
1C06 4AB2                 add r3,=2
                          repeat
1C08 6023                   lda r2,(r3)+
1C0A 9A0D                   until zero
1C0C 1016                   save r1,r3
1C0E 1036
1C10 7078 00D2              setmap m2,mapv2
1C14 3078 FC04
1C18 4072                   call (r2)
                            stnxt:
1C1A 6036                   restore r1,r3
1C1C 6016
1C1E 4078 1C34              call tsleep
1C22 90F3                 endrepeat
                        endif
                      endif
1C24 4E18 0010      until r1 = =%ncodep         ;done all code pages
1C28 81E7
                  endrepeat
1C2A 4078 1C54    call tslee2                   ;sleep without poke
1C2E 90E3       endrepeat
```

```
                ;TSLEEP
                ;Put the STO process to sleep
                ;assumes M1,M3 set to MAPVAR

1C30 4868 6500  otsleep:        lda sp,=stostack-<2*words>      ;compatibility crock.

1C34 1076       routine tsleep, local m0, arg m1, arg m3
1C36 7078 40BE
1C3A 1076

                  ;;poke stopid          ;reawaken us too
1C3C 4878 0042    lda r7,=stopid
1C40 30F8 00AC    sta r7,@pid           ;don't yet trust the poke macro.
1C44 4078 1C54    call tslee2           ;and resume here

1C48 6076       endroutine tsleep
1C4A 707F 00B0
1C4E 3078 FC00
1C52 6006

1C54 1076       routine tslee2,local r1-r5,local m2,arg m1,arg m3
1C56 1016
1C58 1026
1C5A 1036
1C5C 1046
1C5E 1056
1C60 7078 80BE
1C64 1076

1C66 3068 6504    sta sp,stosp          ;save the stack pointer too
1C6A 3008 A506    unlock stolok
1C6E 4008 1072    jmp loopmv

                ;(OVER)
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 129
STO.PLR;1       PAGE 3  Slow Timeout Dispatch

                sto:                    ;enter here from LOOP
1C72 7078 A506    lock stolok
1C76 9AFE
1C78 8904         if odd                ;have we been timed out
1C7A 7078 A508      lda r7,stoinit+foo  ;yes, show we need to reinit
1C7E E104           Trap 404,<;lock timed out - page 128>
                  endif
1C80 7078 620A    lda r7,sidflg                 ;may we run yet?
1C84 9B07         ifnot minus                   ;no, we need to init.
1C86 7078 A508      lda r7,stoinit+foo          ;mark as needing init.
1C8A 3008 A506      unlock stolok
1C8E 4008 108E      tr loop
                  endif
1C92 7078 6508    ifnot stoinit
1C96 8A06
1C98 3008 6508      set stoinit                 ;we're initialized now
1C9C 4868 6504      lda sp,=stostack            ;proper stack pointer
1CA0 90AA           tr stoloop                  ;and start process
                  endif
1CA2 7068 6504    lda sp,stosp                  ;restore stack ptr

1CA6 6076       endroutine tslee2
1CA8 707F 00B0
1CAC 3078 FC04
1CB0 6056
1CB2 6046
1CB4 6036
1CB6 6026
1CB8 6016
1CBA 6006

                ;STO subroutines to help manage pages

                ;routine to set init timer on each page.

1CBC 1076       routine tosst

1CBE 48F3         lda r7,=3
1CC0 3078 40B4    sta r7,intime

1CC4 6006       endroutine tosst

                ;routine to make sure timeout runs on every page
                ;trigger restart if not

1CC6 1076       routine tickin

                  repeat
1CC8 7078 40B4      lda r7,intime
1CCC 8A04           while zero
1CCE 4078 1C34      call tsleep
1CD2 90FB         endrepeat
1CD4 49F1         sub r7,=1
1CD6 3078 40B4    sta r7,intime

1CDA 6006       endroutine tickin
```

```
.INSERT "UPDWN"
    .INSRT UPDWN
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 131
UPDWN.PLR;1     PAGE 1  Line Up/Down Code

                        .stitle Line Up/Down Code

     0010               .radix  H10          ;hex for this file

FC00 0400               Page PkgCode

.comnt |
 PHDEDL
 line state logic. runs off medium timeout
 entered with R4/ modem parameter block
  |
     0020       hdrst  =  D32   ;time to reset hardware (protocol ticks)

44D8 1076       routine phdedl, arg r4, local r2, uses r1, uses r3, uses r5
44DA 1026

44DC 707C 00B2    lda r7,tiktim(r4)
44E0 7178 6520    sub r7,tikcnt                 ;modulo arithmetic
44E4 4F78 FF00    if r7 .bit. = HFF00           ;time to do this line
44E8 9A2A
44EA 797C 00B1      subb r7,tikrat(r4)
44EE 317C 00B2      subm r7,tiktim(r4)          ;save next tick's time
44F2 48A1           lda r2,=1                   ;constant for everybody
44F4 707C 4000      lock lockm(r4)
44F8 9AFE
44FA 322C 00B4      addm r2,ihytik(r4)          ;count tick for err reports
44FE 322C 00AA      addm r2,lmiss(r4)           ;count ticks between misses
4502 785C 0002      ldab r5,phflag(r4)          ;line state now
4506 783C 0003      ldab r3,auxcnt(r4)          ;and counter
450A 4078 4540      call dodedl                 ;do it for this line
450E 4B58 FFF7      and r5,=-1?rcdhel           ;clear hello received
4512 385C 0002      stab r5,phflag(r4)          ;update line state
4516 383C 0003      stab r3,auxcnt(r4)          ;new counter value
451A 702C 4006      set r2 = ckerrs+foo(r4)     ;how many bedd inputs?
451E 4EA5           if r2 >= #5                 ;too many
4520 9202
4522 E415             Trap 2025,<;Too many modem h/w cksum errors - page 130>
                    endif
4524 300C 4000      unlock lockm(r4)
4528 702C 0044      lda r2,filling(r4)          ;hold timers for our buffers
452C 8B03           if minus                    ;there's one here
452E 300A 09F0        set flushd-chain(r2)      ;mark it
                    endif
4532 702C 0046      lda r2,nxtbf(r4)            ;same service for NXTBF
4536 8B03           if minus                    ;there's one here
4538 300A 09F0        set flushd-chain(r2)      ;mark it
                    endif
                  endif

453C 6026         endroutine phdedl
453E 6006
```

```
.comnt |
line transition code:
R5 contains the lines state word
R4 is the modem parameter block
R3 has counter (use depends on state)
R2 always has a 1
|
4540 1076       routine dodedl, arg r2-r5, uses r1, result r3, result r5

4542 7078 72B8    ifnot rutinit                 ;routing 6-second delay
4546 8A03
4548 48D3           set r5 = #inhrst}master     ;hold us in RESET state
454A 48B0           clear r3
                  endif

                  select r5 .bit. =             ;various cases

454C 4FD2          case inhrst                  ;resetting
454E 9A2C
4550 4C58 0081      ior r5,#shihy}master        ;always send Hello/IHY and be master
4554 4078 14E8      call pokem
4558 4EB0           if r3 = =0                  ;entering reset
455A 8103
455C 4078 470C        call linei1               ;reset hardware
                    endif
4560 1056           save r5                     ;save over calls
4562 4078 46AE      call clinei2                ;reset modem state stuff,
4566 4078 1BBA      call grrpkts                ;and dump its traffic
456A 6056           restore r5
456C 4AB1           add r3,=1                   ;ticks in reset
456E 7E3C 00AE      if byte r3 > k(r4) & r5 .bit. =rcdhel
4572 8C15
4574 4FD8
4576 9A13
4578 4DD2             eor r5,=inhrst            ;out of reset
457A 4878 01D4        lda r7,=medmin            ;set up 1 min of med ticks
457E 7278 6520        add r7,tikcnt             ;set alarm clock
4582 307C 009C        sta r7,rutwait(r4)        ;when we can come up
4586 48B0             lda r3,=0                 ;clear hit/miss cnt
4588 383C 00A2        stab r3,odelt(r4)         ;master miss count
458C 4078 4648        call rtrclr               ;reset retrans timers for SPF
4590 7078 62E2        lda r7,mine
4594 7E7C 0005        if byte r7 < neigh(r4)
4598 8202
459A 4DD1               eor r5,=master          ;I'm slave
                      endif
                    endif
459C 4E38 0020      if r3 >= =hdrst             ;time to reset again
45A0 9202
45A2 48B0             lda r3,=0
                    endif

                  ;(OVER)
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 133
UPDWN.PLR;1     PAGE 3  Line Up/Down Code

45A4 9051          case master          ;master, not resetting
45A6 4FD1
45A8 9A3F
45AA 4C58 0080      ior r5,=shihy               ;send hello
45AE 4078 14E8      call pokem                  ;set i2mpok and poke modem
45B2 4FD4           if r5 .bit. =phup           ;line alive
45B4 9A04
45B6 4078 4662        call knmiss               ;see if line still good
45BA 9035            else                       ;master, coming up
45BC 4AB1             add r3,=1          ;default getting a hihy
45BE 793C 00B0        subb r3,NUP(r4)           ;how close to comming up are we
45C2 8B14             if minus                  ;still coming up, use consec cntr
45C4 4FD8               if r5 .bit. =rcdhel     ;got a hello
45C6 9A06
45C8 7F7C 40A2            tstb r7,odelt+foo(r4) ;clear misses
45CC 7A3C 00B0            addb r3,NUP(r4)       ;fix count again
45D0 900C                else                   ;got no hello
45D2 4078 4648            call rtrclr            ;start over again
45D6 48B0                 lda r3,=0              ;zero hit count
45D8 3A2C 00A2            addbm r2,odelt(r4)     ;bump misses
45DC 787C 00A2            if byte odelt(r4) >= k(r4)
45E0 7E7C 00AE
45E4 9202
45E6 4CD2                   ior r5,=inhrst       ;gotta go down
                          endif
                        endif
45E8 901E              else                       ;r3 plus=> into wait state, use KOON
45EA 8A0F               if zero                   ;just come into wait, set up KOON
45EC 487C 0008            lda r7,=ldelt(r4)
                          repeat
45F0 5F7F 40A2              tstb r7,odelt+foo(-r7)      ;clear deltas
45F4 4E74                   until r7 = r4
45F6 81FD
                          endrepeat
45F8 787C 00AF            ldab r7,N(r4)          ;init k-out-of-N stuff
45FC 307C 00AC            sta r7,tdelt(r4)
4600 387C 00A2            stab r7,odelt(r4)
4604 7F7C 40A1            tstb r7,kpoint+foo(r4)        ;clear pointer to odelt
                        endif
4608 7A3C 00B0          addb r3,NUP(r4) ;fix count of ticks
460C 7078 6520          lda r7,tikcnt   ;current time
4610 717C 009C          sub r7,rutwait(r4)      ;how much time left in wait
4614 9B06               if nminus               ;time to be up
4616 4078 4724            call lineup
461A 7C5C 0002            iorb r5,phflag(r4)
461E 48B0                 clear r3              ;reset AUXCNT always
                        endif
4620 4078 4662          call knmiss             ;check line
                      endif
                    endif
                  ;(OVER)
```

```
4624 9011           case rcdhel                    ;slave, got hello
4626 4FD8
4628 9A03
462A 48B0            lda r3,=0                     ;clear miss count

462C 900D           default                        ;slave, no hello
462E 4AB1            add r3,=1                     ;incr miss count
4630 7E3C 00AE       if byte r3 > k(r4)            ;too many consec
4634 8C09
4636 4FD4              if r5 .bit. =phup           ;was line up?
4638 9A05
                    ;      Trap 2016,<;Slave missed 4 in a row - page 133>
463A 4078 1CF4         call killin                 ;yes, kill it
463E 785C 0002         ldab r5,phflag(r4)
                     endif
4642 48B0              lda r3,=0                   ;always reset auxcnt
4644 4CD2              ior r5,=inhrst              ;back to reset
                    endif
                  endselect

4646 6006         endroutine dodedl


                  ;RTCLR- clear SPF retransmission timers

4648 1076         Routine RTRCLR, arg r4

464A 707C 40B8    lock rtimrl(r4)                 ;lock timers
464E 9AFE
4650 487C 0020    lda r7,=<<<nimp-1>_-3>+1>*words(r4)     ;loop control
                  repeat
4654 577F 40BC      tst r7,rtimrs+foo(-r7)        ;clear a word of timers
4658 4E74           until r7 = r4
465A 81FD
                  endrepeat
465C 300C 40B8    unlock rtimrl(r4)

4660 6006         endroutine rtrclr
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 135
UPDWN.PLR;1     PAGE 5  Line Up/Down Code


.comnt | KNMISS
  Called when in master mode to do the K out of N calculation.
  Expects the line state in r5, returns new value in r5
  Returns new TDELT in R1, AUXCNT in R3
|
4662 1076          routine knmiss, arg r3-r5, result r1/r3/r5

4664 4FD8            if r5 .nbit. =rcdhel          ;we missed one
4666 8A23
4668 701C 40AA         lda r1,lmiss+foo(r4)        ;how long since last miss
466C 4F18 FF00         if r1 .bit. = HFF00         ;max diff?
4670 9A03
4672 4818 00FF           lda r1,= HFF              ;eliminate modulo err
                       endif
4676 787C 00A1         ldab r7,kpoint(r4)          ;where is oldest miss
467A 49F1              sub r7,=1                   ;get to oldest slot
467C 8B04              if minus                    ;last slot we're supposed to use
467E 787C 00AE           ldab r7,k(r4)             ;how many slots we have
4682 49F2                sub r7,=2                 ;save k-1 misses (from 0 to k-2)
                       endif
4684 387C 00A1         stab r7,kpoint(r4)          ;save for next miss
4688 4A74              add r7,r4                   ;get to right modem block
468A 791F 00A2         subb r1,odelt(r7)           ;time between misses for oldest
468E 3A1F 00A2         addbm r1,odelt(r7)          ;save newest, and remove old
4692 721C 00AC         add r1,tdelt(r4)            ;add sum of last k-1 misses
                       if byte r1 < n(r4)          ;does the new one break the camel's back?
4696 7E1C 00AF
469A 8207
469C E40C                Trap 2014,<;Master line died - page 134>
469E 4078 1CF4           call killin               ;must kill it for k-out-of-n
46A2 48B0                clear r3                  ;reset AUXCNT
46A4 785C 0002           ldab r5,phflag(r4)
                       endif
46A8 301C 00AC         sta r1,tdelt(r4)            ;and save sum
                     endif

46AC 6006          endroutine knmiss
```

```
                ;LINEI2
                ;Reset modem state, including various line state control
                ;parameters and logical channel variables and tables.

46AE 1076       routine clinei2, arg r4, uses r1, local r2
46B0 1026

1CDC 1076       global linei2 = clinei2
1CDE 7078 40BE
1CE2 1076    .
1CE4 7078 00B4
1CE8 3078 FC00
1CEC 4078 46AE
1CF0 4008 1BCE
FC00 0400

                ;first, set line protocol stuff

46B2 782C 001A    ldab r2,modem(r4)

46B6 701A 4102    lda r1,channel(r2)            ;channels for this line
46BA 707A 4112    lda r7,irrtime(r2)            ;retransmit time for this line
46BE 301C 001C    sta r1,maxchn(r4)             ;num of chans this line.
46C2 307C 0032    sta r7,mrtime(r4)             ;and retransmit rate
46C6 702A 40F2    lda r2,lineprot(r2)           ;protocol version,,route rate
46CA 382C 00A0    stab r2,rutspd(r4)            ;set routing interval
46CE A6A8         srl r2, H8                    ;get protocol version
46D0 382C 001B    stab r2,clockm(r4)            ;this is "line speed"
46D4 707A 4122    lda r7,protcnt(r2)
46D8 307C 00B0    sta r7,NUP(r4)        ;ticks to come up,,med ticks per line tick
46DC 707A 4132    lda r7,knvalu(r2)
46E0 307C 00AE    sta r7,k(r4)          ;;N     ;set up k,n for this line
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 137
UPDWN.PLR;1     PAGE 7  Line Up/Down Code

                ;Now initialize the RSEX,TSEX,INFREE,CHNBSY bits

46E4 4828 FFFF    lda r2,=-1
46E8 4E18 0010    if r1 = = H8*2                ;exactly 8 channels?
46EC 8102
46EE A2A8           sll r2, H8
                  endif

46F0 4814         lda r1,r4
46F2 48F8         lda r7,=Nmdlim/ D16
                  repeat
46F4 3029 0062      sta r2,rsex(r1)
46F8 3029 0072      sta r2,infree(r1)
46FC 3029 0052      sta r2,tsex(r1)
4700 6779 4082      tst r7,chnbsy+foo(r1)+
4704 49F1           sub r7,=1
4706 8AF7           until zero
                  endrepeat

4708 6026       endroutine clinei2
470A 6006

                ;LINEI1
                ;reset hardware for modem parameter block in R4

470C 1076       routine linei1, arg r4, uses r1

470E 701C 0020    lda r1,iobloc(r4)     ;device address
4712 4878 0100    lda r7,=mreset
4716 3079 000C    sta r7,statom(r1)
471A 747C 001E    ior r7,mloop(r4)
471E 3079 0006    sta r7,statim(r1)

4722 6006       endroutine linei1
```

```
.comnt |
LINEUP - bring a line up instantly
Assumes that lockm for the line is already locked.
Resets delay and retrans clock. Changes line state.
Checks if time to come out of init state.
Flags for routing update and trouble report to go.
|

4724 1076       Routine LINEUP, arg r4, local r5
4726 1056

4728 785C 001A  ldab r5,modem(r4)              ;doubled modem number
472C 705D 12D6  lda r5,propdl(r5)              ;propagation delay for this line
4730 A6D3       srl r5,smoshf                  ;shift by averaging shift
4732 8A02       if zero
4734 48D1         lda r5,=1                    ;never report zero delay
                endif
4736 4E58 00FF  if r5 >= =dlinf
473A 9203
473C 4858 00FE    lda r5,=dlinf-1              ;or infinite
                endif
4740 707C 4092  lock dlock(r4)                 ;lock delay vars
4744 9AFE
4746 385C 009A  stab r5,delave(r4)             ;initial delay is equal to
474A 385C 009B  stab r5,delbas(r4)             ;propagation delay
474E 707C 4098  lda r7,dellow+foo(r4)          ;clear averaging counters
4752 707C 4096  lda r7,delhi+foo(r4)
4756 707C 4094  lda r7,dpcnt+foo(r4)
475A 300C 4092  unlock dlock(r4)
475E 707C 40B8  lock rtimrl(r4)
4762 9AFE
4764 A6D2        srl r5,srtshf          ;convert to 25.6 ms ticks
4766 305C 00BA  sta r5,rtrclk(r4)       ;;rtrtic :set clock rate, clear count
476A 300C 40B8  unlock rtimrl(r4)
476E 48F4       lda r7,=phup
4770 3C7C 0002  iorbm r7,phflag(r4)            ;mark line up
4774 787C 4003  ldab r7,auxcnt+foo(r4)         ;clear protocol counter
4778 787C 0005  if byte neigh(r4) <> mine      ;looped line?
477C 7E78 62E2
4780 9103
4782 3008 72A6    set iniflg                   ;no, bring us out of init state
                endif
                 ;  Trap 2720,<;Line up, r7=neighbor - page 137>
4786 3008 6512  set ophgo                      ;send trouble report
478A 3008 72B2  set rupsnd                     ;and routing update
478E 4878 0044  set @pid = #rutpi              ;poke routing
4792 30F8 00AC

4796 6056       endroutine lineup
4798 6006
```

```
                        page LCode

.comnt |
KILLIN - kill a line instantly
Assumes that lockm for the line is already locked.
Resets line state, delay values, and retrans clock.
Flags a SPF update to be sent and a trouble report.
|

1CF4 1076       Routine KILLIN, arg r4

1CF6 48F3       setb phflag(r4) = =inhrst}master        ;go into reset state
1CF8 387C 0002
1CFC 787C 4003  ldab r7,auxcnt+foo(r4)          ;zero reset ticks
1D00 787C 4005  ldab r7,neigh+foo(r4)           ;clear neighbor number
1D04 707C 4092  lock dlock(r4)                  ;lock this lines delay vars
1D08 9AFE
1D0A 4878 FFFF  set delave(r4) = =dlinf* H101   ;;delbas ;set infinite delay
1D0E 307C 009A
1D12 300C 4092  unlock dlock(r4)
1D16 3008 6512  set ophgo                       ;send trouble report
1D1A 3008 72B2  set rupsnd                      ;and routing update
1D1E 4878 0044  set @pid = #rutpi                       ;poke routing
1D22 30F8 00AC
1D26 707C 40B8  lock rtimrl(r4)
1D2A 9AFE
1D2C 4878 0019  set rtrclk(r4) = =rticd ;;rtrtic ;reset SPF retrans clock
1D30 307C 00BA
1D34 300C 40B8  unlock rtimrl(r4)

1D38 6006       endroutine killin

     0008               .radix  010     ;return to octal
```

```
.INSERT "MODEM"
    .INSRT MODEM
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 141
MODEM.PLR;2     PAGE 1  IMP Modem IO Code

                        .stitle IMP Modem IO Code

                        Page Lcode

1D3A 1076       routine i2m, arg r1
1D3C 7049 81C0      lda r4,mblks(r1)
1D40 705C 0020      lda r5,iobloc(r4)
1D44 707C 4024      lock i2mloc(r4)             ;for hardware, LATER
1D48 9AFE
1D4A 707D 000C          lda r7,statom(r5)
1D4E 4F78 2000          if r7 .nbit. =mbusy     ;modem not busy.
1D52 8A60
1D54 4B78 0100              and r7,=mquit       ;;statom
1D58 9A06                   ifnot zero
1D5A 702C 0026                  lda r2,snding(r4)   ;R2: reports buffer CHAIN word
1D5E E404                       Trap 2004,<;modem output got quit - page 140>
1D60 307D 000C                  sta r7,statom(r5)   ;;mreset
                            endif
     0001       .lif z LBig
                            setmap m0,mappkg
                                        ;handle output just completed
1D64 707C 4000              lock lockm(r4)              ;for SNDING, ESENTQ
1D68 9AFE
1D6A 702C 4026              lda r2,snding+foo(r4)       ;first look for data packet
1D6E 9A10                   ifnot zero                  ;there is one
1D70 707A 06A0                  if where-chain(r2) .nbit. #whi2m    ;lost ownership
1D74 4F78 0100
1D78 8A03
1D7A E40B                           Trap 2013,<;lost SNDING buffer - page 140>
1D7C 9007                       else
1D7E 48F1                           set (r2) = #1       ;this buffer will end queue
1D80 3072
1D82 30AC 0036                      set @esentq(r4) = r2    ;enqueue it
1D86 302C 0036                      set esentq(r4) = r2
                                endif
1D8A 3004                       unlock (r4) ;;lockm
1D8C 9012                   else
1D8E 3004                       unlock (r4) ;;lockm
1D90 702C 4028                  lda r2,later+foo(r4)    ;else try packet to flush
1D94 9A0E                       ifnot zero
1D96 890A                           if odd              ;it's a flooded packet
1D98 4B28 FFFE                          and r2,#-2      ;remove odd bit
1D9C 783C 001A                          ldab r3,modem(r4)   ;get our modem bit
1DA0 703B 0EB6                          lda r3,bittab(r3)
1DA4 4078 55D2                          call rupfls     ;and flush our use
1DA8 9004                           else
1DAA 4078 1332                          call flush,whi2m    ;just free the buffer
1DAE 0100
                                    endif
                                endif
                            endif

                ;more I2M over
```

```
                ;I2M continued
                ;Look for high-priority, low-probability things
                ;for I2M to do:
                ;  (1) send a hello/i heard you msg
                ;  (2) flood an SPF update
                ;  (3) send a demand reload
                ;  (4) send a packet core packet
                ;  (5) send a retransmission if necessary

1DB0 707C 402A            if i2mpok+foo(r4)              ;some important stuff
1DB4 9A11
1DB6 48A0                    clear r2                    ;use an indexed dispatch
                             repeat
1DB8 607A 479A                  lda r7,i2mlst(r2)+       ;next to call
1DBC 1026                       save r2
1DBE 4077                       call (r7)                ;do a check
1DC0 9A07                       if success               ;it sent something
1DC2 300C 4024                     unlock i2mloc(r4)     ;done I2M
1DC6 6026                          restore r2
1DC8 300C 002A                     set i2mpok(r4)        ;schedule another check
1DCC 6006                          return
                                endif
1DCE 6026                       restore r2
1DD0 4E28 000A                 while r2 < #li2mlst       ;more to do
1DD4 92F2
                             endrepeat
                          endif


FC00 0400                  page PkgCode                 ;lovely little hack to keep table out of way

479A                    table i2mlst
479A 47A4                 i2mhihy
479C 4808                 i2mfld
479E 4834                 i2mdemrel
47A0 4870                 i2mpkc
47A2 1E18                 i2mrxm
                        endtable i2mlst

                           page LCode
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 143
MODEM.PLR;2     PAGE 3  IMP Modem IO Code

                                    ;check for regular stuff
1DD6 787C 0002            if byte phflag(r4) .bit. #phup :if up, maybe send data
1DDA 4FF4
1DDC 9A1B
1DDE 707C 4000               lock lockm(r4)              ;need this for the next checks
1DE2 9AFE

                                    ;look for data to send now
1DE4 481C 0038               set r1 = #spriq(r4)         ;send priority first
1DE8 7021                    set r2 = (r1)               ;packet here?
1DEA 8909                    if odd                      ;nope
1DEC 4A94                       add r1,#sregq-spriq      ;try regular Q
1DEE 7021                       set r2 = (r1)            ;packet here?
1DF0 8906                       if odd                   ;nope
1DF2 4078 1F16                     call i2mnul ;;unlock lockm    ;maybe send a null
1DF6 300C 4024                     unlock i2mloc(r4)     ;done modem out
1DFA 90E9                          return
                                endif
                             endif
1DFC 4078 1EC6               call i2msfc                 ;assign us a channel
1E00 9A08                    if success                  ;can send one
                ;               lda r7,#20               set retransmission count
1E02 4878 0020                  SET STIMER(R1)=# D32
1E06 3079 0094
1E0A 4078 1F56                  call i2msnd              ; and send packet
1E0E 9002                    else                        ;some impossible case
1E10 3004                       unlock (r4)      ;;lockm :give up
                             endif
                          endif

                       endif     ;;mbusy
1E12 300C 4024      unlock i2mloc(r4)
1E16 90DB         endroutine i2m
```

```
.comnt |
Build and send a Hello/IHY
|
FC00 0400              Page PkgCode

47A4 1076       routine I2MHIHY, arg r4, uses r1-r3/r5

47A6 707C 4000    lock lockm(r4)                ;must control phflag
47AA 9AFE
47AC 783C 0002      ldab r3,phflag(r4)
47B0 4F38 0080      if r3 .nbit. =shihy         ;there's no hello to send
47B4 8A05
47B6 3004             unlock (r4)       ;;lockm
47B8 6076             fail return               ;done here
47BA 4FF0
47BC 4007
                    endif
47BE 4D38 0080      eor r3,#shihy               ;clear this bit
47C2 383C 0002      stab r3,phflag(r4)          ;and change flag
47C6 3004         unlock (r4)   ;;lockm
47C8 4B38 FF05    and r3,=helst                 ;clear internal info
47CC 4C38 A202    ior r3,=ruttyp+combit+iherdu+spfon    ;nominal HIHY
                ;  set r1 = =nullhd+seqh(r4)    ;last word of msg
47D0 48A5         set r2 = =<seqh/words+1>      ;build checksum
47D2 7078 62DE    lda r7,sync
47D6 307C 0018    sta r7,nullhd+seqh(r4)              ;;seqh  ;build HIHY backwards
47DA 4927         sub r2,r7
47DC 7078 62E2    lda r7,mine
47E0 307C 0016    sta r7,nullhd+srch(r4)              ;;srch
47E4 4927         sub r2,r7
47E6 48F0       lda r7,#0
47E8 307C 0010  sta r7,nullhd+neth(r4)
47EC 787C 001A   ldab r7,modem(r4)
47F0 A2F7        sll r7,7
47F2 307C 0010  sta r7,nullhd+neth(r4)
47F6 4927        sub r2,r7
47F8 4923        sub r2,r3
47FA 302C 0014    sta r2,nullhd+chkh(r4)             ;;chkh
47FE 303C 0012    sta r3,nullhd+typh(r4)             ;;typh
                ihycum:
4802 4078 1FDC    call i2mxmit                ;send it}

4806 6006       endroutine i2mhihy
```

```
.comnt |
I2MFLD- send out routing update

If buffer on rupq for this modem, send it out unless line is
in reset or neighbor not running SPF.
Turn on retransmission timer for this IMP.
|

4808 1076       Routine I2MFLD, arg r4, uses r2-r3, uses r5

480A 783C 001A    ldab r3,modem(r4)             ;our modem number
480E 703B 0EB6    lda r3,bittab(r3)             ;our mask bit
                  repeat
4812 4078 5584      call rupdeq                 ;attempt to get an update
4816 8A04           if fail
4818 6076             fail return               ;nothing for us
481A 4FF0
481C 4007
                    endif
481E 4078 17D4      call pcksubr                ;check its checksum first
4822 9A05           until zero                  ;okay
4824 E406           Trap 2006,<;Bad cksum in routing update - page 144>
4826 4078 55D2      call rupfls                 ;ignore this one, try again
482A 90F4         endrepeat
482C 4CA1         ior r2,=1                     ;flag so I2MRLA will get it
                imcfld:                         ;hook to count updates sent
482E 4078 1FD4    call i2mqxmit                 ;send the update

4832 6006       endroutine i2mfld
```

```
                ;send a demand reload if asked

FC00 0400               page PkgCode

4834 1076       routine i2mdemrel
4836 705C 402C    lda r5,demand+foo(r4)         ;get demand flag
483A 9A18         ifnot zero                    ;its set}
483C 305C 0016      sta r5,nullhd+srch(r4)      ;set demand flag
4840 4878 CAFE      lda r7,=dempas              ;password
4844 307C 0018      sta r7,nullhd+seqh(r4)
4848 4878 C100      lda r7,=rldtyp+demrel
484C 307C 0012      sta r7,nullhd+typh(r4)
4850 4878 7407      lda r7,#-<dempas+rldtyp+demrel-<pkth/words>>
4854 4975           sub r7,r5                   ;make checksum
4856 307C 0014      sta r7,nullhd+chkh(r4)      ; and make it look good
485A 48D0           lda r5,#0                   ;;odeven endbit dscpkt
485C 48B0           lda r3,#0                   ;;chanum
485E 481C 0010      lda r1,#nullhd(r4)          ;where to send from
4862 3031           sta r3,(r1)                 ;;neth
4864 4078 1FDC      call i2mxmit                ;send it
4868 6006           return
                  endif
486A 6076         fail return
486C 4FF0
486E 4007
                endroutine i2mdemrel
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 147
MODEM.PLR;2     PAGE 7  IMP Modem IO Code

                ;send packet core or routing, if needed.

FC00 0400               page PkgCode

4870 1076       routine i2mpkc, arg r4, uses r1-r3, uses r5

4872 702C 002E    lda r2,sblk(r4)               ;check for packet core.
4876 8A04         if zero                       ;no packet core block to send
4878 6076           fail return                 ;not done yet
487A 4FF0
487C 4007
                  endif
487E 4078 1480    call wheorm,whi2m             ;add i2m ownership
4882 0100
4884 48F1         lda r7,=1                     ;pkc retransmit count.
4886 317A 0350    subm r7,chan-chain(r2)
488A 8B06         if minus                      ;transmitted enough times
488C 707C 402E      lda r7,sblk+foo(r4)         ;free sblk for next
4890 4078 131E      call flushb,whf2h           ;flush that buffer.
4894 4000
                  endif
4896 4078 48A4    call pkchold                  ;hold off resets
489A 4078 1512    call unpack                   ;get buffer addresses.
489E 4078 1FD4    call i2mqxmit                 ;transmit routing buffer.

48A2 6006       endroutine i2mpkc

                ;PKCHOLD
                ;hold off modem resets if packet core is active

48A4 1076       routine pkchold,arg r4
48A6 787C 0002    if byte phflag(r4) .bit. #inhrst      ;liable to reset?
48AA 4FF2
48AC 9A04
48AE 48F1           setb auxcnt(r4) = =1
48B0 387C 0003
                  endif
48B4 6006       endroutine pkchold
```

```
                ;retransmit first on sent queue, if needed.

                        page LCode

1E18 1076       routine i2mrxm, arg r2-r5, uses r1-r3/r5
1E1A 707C 4000     lock lockm(r4)
1E1E 9AFE
1E20 787C 0002     if byte phflag(r4) .nbit. #phup      ;send only if line up
1E24 4FF4
1E26 8A03
1E28 3004             unlock (r4)       ;;lockm
1E2A 6006             return                        ;succeed to do nothing else
                   endif
1E2C 481C 0034     lda r1,=ssentq(r4)
1E30 4078 154A     call deque, whi2m
1E34 0100
1E36 8A05          if fail
1E38 3004             unlock (r4)       ;;lockm
1E3A 6076             fail return
1E3C 4FF0
1E3E 4007
                   endif
1E40 7079 009A     set r7 = st(r1)                  ;it's last send time
1E44 727C 0032     add r7,mrtime(r4)                ;plus retransmit time
1E48 71F8 62E8     sub r7,@clock                    ;less time now
1E4C 9B0F          ifnot minus                      ;don't retransmit yet.
1E4E 7078 00D2        setmap m2,mapv2               ;requeue at head of sent q
1E52 3078 FC04
1E56 707C 0034        lda r7,ssentq(r4)
1E5A 3072             sta r7,(r2)                   ;put q after this packet.
1E5C 8903             if odd
1E5E 302C 0036           sta r2,esentq(r4)
                      endif
1E62 302C 0034        sta r2,ssentq(r4)             ;packet onto q.
1E66 3004             unlock (r4)       ;;lockm
1E68 90E9             fail return
                   endif
1E6A 4853          lda r5,r3                        ;save chan over call
1E6C 4078 17D4     call pcksubr                     ;checksum packet.
1E70 9A12          ifnot zero                       ;bad checksum}
1E72 E411             Trap 2021,<;broken cksum on retransmission - page 147>
1E74 4815             lda r1,r5                     ;get CHAN to free channel
1E76 A694             srl r1,4                      ;ack quadrant
1E78 4A1C 0082        add r1,#chnbsy(r4)            ;where bits are in block
1E7C 4875             lda r7,r5                     ;low bits of channel #
1E7E 4BFE             and r7,= HE                   ;just those
1E80 707F 0EB6        lda r7,bittab(r7)             ;channel bit
1E84 3D71             eorbm r7,(r1)  ;;chnbsy       ;free for use
1E86 48F1             lda r7,#1
1E88 327C 0050        addm r7,slots(r4)             ;count this freed
1E8C 4078 41AA        call ckqput                   ;send it for analysis
1E90 3004             unlock (r4)       ;;lockm
1E92 90D4             fail return
                   endif
1E94 4835          lda r3,r5                        ;recover chan
1E96 7051          lda r5,(r1)  ;;neth odeven
1E98 48            lda r7,=1
```

```
1E9A 3179 0094      subm r7,stimer(r1)          ;decrement retransmit cntr.
1E9E 8A11           if zero                     ;0 mod 32 rexmits.
1EA0 4F58 2000         if r5 .bit. =dscpkt      ;packet marked for discard?
1EA4 9A05
```

```
                        Trap 2022,<;64 retransmissions: killed line - page 148>
1EA6 E412
1EA8 4078 1CF4            call killin
1EAC 900A               else
                          Trap 2023,<;32 retransmissions: discard packet - page 148>
1EAE E413
1EB0 4C58 2000            ior r5,=dscpkt          ;set dscbit for i2msnd
1EB4 4078 2382            call cdelay             ;compute delay.
1EB8 4878 0020            lda r7,= D32            ;try another 32 times.
1EBC 3079 0094            sta r7,stimer(r1)
                        endif
                      endif
1EC0 4078 1F56        call i2msnd                 ;send packet in r2.
1EC4 90B3           endroutine i2mrxm
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 150
MODEM.PLR;2     PAGE 9  IMP Modem IO Code

                ;select a free channel for data packet.
1EC6 1076       routine i2msfc, arg r1-r2, arg r4, result r1-r3, result r5
1EC8 48B0         lda r3,=0             ;select a free channel.
1ECA 487C 0082    lda r7,=chnbsy(r4)
                  repeat
1ECE 6857           ldab r5,(r7)+
1ED0 4E58 00FF      until r5 <> = HFF
1ED4 810A
1ED6 4A38 0010      add r3,= H8*2
1EDA 763C 001C      if r3 = maxchn(r4)
1EDE 8104
1EE0 6076             fail return
1EE2 4FF0
1EE4 4007
                    endif
1EE6 90F4         endrepeat

                  ;r3 is free channel group.
1EE8 1016         save r1
1EEA 4891         lda r1,=1             ;channel mark bit.
                  repeat
1EEC 4F51           while r5 .bit. r1
1EEE 9A04
1EF0 A291           sll r1,1
1EF2 4AB2           add r3,=2
1EF4 90FC         endrepeat

                  ;r3 is channel, r1 is bit
1EF6 1C17         iorbm r1,(-r7)        ;mark channel used
1EF8 48D0         lda r5,=0
1EFA 7F1F FFD0    if byte r1 .nbit. tsex-chnbsy(r7)     ;get the odeven bit
1EFE 8A03
1F00 4858 8000      lda r5,=odeven      ;set the bit
                  endif
1F04 303A 0350    sta r3,chan-chain(r2) ;clobbers chan if bad buffer
1F08 6016         restore r1            ;restore queue head pointer.
1F0A 4078 154A    call deque,whi2m
1F0E 0100
1F10 8A02         if fail
1F12 90E7           fail return
                  endif
1F14 6006       endroutine i2msfc
```

```
                        page LCode

                ;send nulls to acknowledge received packets. if needed.
1F16 1076       routine i2mnul, arg r4
1F18 707C 0030    if snull(r4)          ;we need to send any nulls.
1F1C 9A04
1F1E 4078 1F28      call i2msnl         ;send null
1F22 9002          else
1F24 3004           unlock (r4)  ;;lockm        ;free modem vars lock.
                  endif
1F26 6006       endroutine i2mnul       ;always succeeds to terminate loop.

                ;send nulls
1F28 1076       routine i2msnl, arg r4
1F2A 4858 8100    lda r5,=rutnul+ruttyp ;setup the null
1F2E 48A5         lda r2,=<seqh+2>/words
1F30 4925         sub r2,r5             ;;typh
1F32 481C 0018    lda r1,=nullhd+seqh(r4)
1F36 7038 62DE    lda r3,sync
1F3A 3031         sta r3,(r1)           ;;seqh
1F3C 4923         sub r2,r3
1F3E 7038 62E2    lda r3,mine
1F42 1031         sta r3,(-r1)          ;;srch
1F44 4923         sub r2,r3
1F46 1021         sta r2,(-r1)          ;;chkh
1F48 1051         sta r5,(-r1)          ;;typh
1F4A 48B0         lda r3,=0             ;;chanum
1F4C 48D0         lda r5,=0             ;;odeven endbit dscpkt
1F4E 1031         sta r3,(-r1)          ;;neth
                imcnull:                ;hook to count nulls
1F50 4078 1F74    call i2mdun           ;now, actually send null.
1F54 6006       endroutine i2msnl
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 152
MODEM.PLR;2     PAGE 11          IMP Modem IO Code

                ;modem output utility routines

                ;r1:buffer address,r2:chain,r3:chn*2,r4:i2mblk,
                ;r5:odeven and dscpkt(needs masking)
1F56 1076       routine i2msnd, arg r1-r5
                  imcumpkt:     ;cumstats im hook
1F58 302C 0026    sta r2,snding(r4)     ;buffer being sent
1F5C 70F8 62E8    lda r7,@clock
1F60 3079 009A    sta r7,st(r1)         ;note time sent
1F64 4B58 A000    and r5,=odeven}dscpkt
1F68 787C 0002    ldab r7,phflag(r4)
1F6C 8903         if odd                ;;master
1F6E 4C58 4000      ior r5,=endbit              ;mark us as high imp
                  endif

                ;r1:buffer address, r3:chn*2,r4:i2mblk,
                ;r5:odeven/endbit/dscpkt
1F72 9002       entry i2mdun, arg r1, arg r3-r5
1F74 1076
1F76 702C 0030    lda r2,snull(r4)      ;which ACK group to send?
1F7A 4B2A FFFF    and r2,=-1(r2)        ;do lowest first
1F7E 752C 0030    eor r2,snull(r4)
1F82 352C 0030    eorm r2,snull(r4)
1F86 3004         unlock (r4)  ;;lockm
1F88 707C 001C    lda r7,maxchn(r4)     ;8 or 16 special case
1F8C 4E78 0010    cmp r7,= D8*2
1F90 707C 001C    lda r7,maxchn(r4)     ;8 or 16 special case
1F94 4E78 0010    cmp r7,= D8*2
1F98 487C 0062    lda r7,=rsex(r4)
1F9C 8C08         if l  ;;if r7 > # D16*2       ;many channels- general case
                    repeat
1F9E A6A1             srl r2,1
1FA0 9A05             until zero
1FA2 4A58 0100        add r5,= H100
1FA6 4AF1             add r7,=1
1FA8 90FB           endrepeat
1FAA 9006           else                  ;special case- 8/16 channels
1FAC 702C 4030        lda r2,snull+foo(r4)        ;will ACK all channels
1FB0 A2B7             sll r3,7                    ;move chanum over
1FB2 7C5F 0001        iorb r5,1(r7)               ;insert high ACKs
                    endif
1FB6 4C53           ior r5,r3                     ;setup packet chn-id (neth)
1FB8 7151           sub r5,(r1)                   ;;neth
1FBA 3251           addm r5,(r1)                  ;;neth ;fix neth
1FBC 3159 0004      subm r5,chkh(r1)              ;correct checksum
1FC0 7837           ldab r3,(r7)                  ;;rsex
1FC2 7939 0003      subb r3,typh+1(r1)            ;update main ACK field
1FC6 3239 0002      addm r3,typh(r1)              ;fixed acks
1FCA 3139 0004      subm r3,chkh(r1)              ;and checksum
1FCE 4078 1FDC      call i2mxmit                  ;send buffer in snding
1FD2 6006       endroutine i2msnd
```

```
                ;r4:i2mblk, r1:packet, r2:chain
                ;if r1 neg then snding=chain else send null.
1FD4 1076       routine i2mqxmit, arg r1/r2/r4, uses r1/r2/r5
1FD6 302C 0028    sta r2,later(r4)              ;remember output bufer

                ;just xmit buffer here
1FDA 9002       entry i2mxmit, arg r1, arg r4, uses r1-r2, uses r5
1FDC 1076
1FDE 4811         lda r1,r1                     ;null if positive
1FE0 8B0E         if minus                      ;;m2 mendf ;if real buffer.
1FE2 4821           lda r2,r1
1FE4 7219 0090      add r1,bufe(r1)
1FE8 4B28 1FFE      and r2,=packm
1FEC A6A4           srl r2,4
1FEE 7428 808C      ior r2,m2#slfptr
1FF2 7078 00D2      setmap m2,mapv2
1FF6 3078 FC04
1FFA 9008         else
1FFC 481C 2018      lda r1,=nullhd+seqh+mendf-m1(r4)
2000 482C A010      lda r2,=nullhd-m1(r4)
2004 A6A4           srl r2,4
2006 7428 00D0      ior r2,mapvar             ;start hardware here
                  endif
200A 705C 0020    lda r5,iobloc(r4)
200E 302D 0008    sta r2,starto(r5)
2012 707D 000A    lda r7,endo(r5)
2016 A6F4         srl r7,4
2018 4D72         eor r7,r2
201A 4B78 01FF    and r7,=packm_-4
201E 9A04         ifnot zero                    ;error in xmit
2020 E405           Trap 2005,<;Start pointer write failed - page 152>
2022 302D 0008      sta r2,starto(r5)
                  endif
2026 301D 000A    sta r1,endo(r5)
202A A2A4         sll r2,4                      ;now unpack the  starting address
202C 4878 1FFF    lda r7,= H1FFF
2030 4B17         and r1,r7
2032 4B27         and r2,r7
2034 4912         sub r1,r2                     ;R1 = # of bytes transmitted
2036 A691         srl r1,1                      ;R1 = # of words transmitted
2038 4A94         add r1,=4                     ;Include 8 of 9 framing characters
203A 707C 400C    lda r7,flipper+foo(r4)        ;hack to add 1/2 word. (X16 style})
203E 8A04         if zero
2040 300C 000C      set flipper(r4)
2044 4A91           add r1,=1
                  endif
2046 321C 0008    addm r1,loword(r4)            ;increment low-order
204A 8B07         if minus                      ;increment high-order if "carry"
204C 48F1           lda r7,=1
204E 327C 000A      addm r7,hiword(r4)
2052 A7F1           rrl r7,1    ;;sign          ;quick get sign bit
2054 357C 0008      eorm r7,loword(r4)          ;and clear it in low order
                  endif
2058 6006       endroutine i2mxmit
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 154
MODEM.PLR;2     PAGE 13          IMP Modem IO Code

                ; m2iloc locks the hardware busy bit, filling and nxtbf

205A 1076       routine m2i, arg r1             ;r1 is PID level.
205C 7049 81C0    lda r4,mblks(r1)              ;modem block pointer.
                ;;m2ihw:
2060 705C 0020    lda r5,iobloc(r4)             ;modem device address.

2064 70B8 62E8    lda r3,@clock                 ;timestamp for arriving packet.

2068 707C 4042    lock m2iloc(r4)               ;lock hardware level
206C 9AFE
206E 702D 0006    lda r2,statim(r5)             ;device input status.
2072 4F28 2000    if r2 .bit. =mbusy            ;modem still busy
2076 9A04
2078 300C 4042      unlock m2iloc(r4)           ;release modem.
207C 9054          else                         ;transfer done, look at new packet.
207E 701D 0004      lda r1,endi(r5)             ;buffer end pointer.
2082 9B24           ifnot minus                 ;;mendf ;no end flag.
2084 4F28 0100        if r2 .bit. =mquit        ;;statim
2088 9A03
208A E402               Trap 2002,<:modem input quit (INPUT LOST) - page 153>
208C 9004              else
208E 48F1               lda r7,=1               ;just count hardware errors.
2090 327C 0006          addm r7,ckerrs(r4)      ;in param block.
                      endif
2094 4878 0100        lda r7,=mreset            ;reset modem.
2098 747C 001E        ior r7,mloop(r4)
209C 307D 0006        sta r7,statim(r5)
20A0 703C 0044        lda r3,filling(r4)        ;start input to next buffer.
20A4 702C 0046        lda r2,nxtbf(r4)
20A8 302C 0044        sta r2,filling(r4)
20AC 4078 158E        call sioin,bufend         ;start modem input.
20B0 008E
20B2 4078 21B0        call m2inxt               ;get new buffer
20B6 300C 4042        unlock m2iloc(r4)         ;release hardware.
20BA 4823             lda r2,r3          ;;filling
20BC 7628 63CA        if r2 <> junk
20C0 9104
20C2 4078 131E          call flushb,whm2i       ;return buffer to system.
20C6 8000
                      endif
```

```
                ;end bit set, so no error.  Handle input normally.
20C8 902E            else                        ;packet has end flag.
20CA 1016             save r1,r3                 ;save end pointer.
20CC 1036
20CE 702C 0046        lda r2,nxtbf(r4)           ;next buffer to use
20D2 4078 158E        call sioin,bufend          ;start modem up on this buffer.
20D6 008E

20D8 705C 0044        lda r5,filling(r4)         ;previous input buffer
20DC 302C 0044        sta r2,filling(r4)         ;current (new) input buffer
20E0 4078 21B0        call m2inxt                ;get new buffer
20E4 707C 4048        lock lmiq(r4)              ;now queue the packet
20E8 9AFE
                      unlock m2iloc(r4)          ;Bill says to watch out for this lock order.
20EA 300C 4042
20EE 6036             restore r1,r3              ;end pointer.
20F0 6016
20F2 7658 63CA        if r5 <> junk     ;;filling ;real data in buffer.
20F6 9115
20F8 4825               lda r2,r5                ;gain access to buffer
20FA 4851               lda r5,r1                ;save end pointer.
20FC 4078 1500          call unpckc,whm2i        ;unpack and check.
2100 8000
2102 8A03               if fail                  ;not our buffer
2104 E4C1                 Trap 2301,<;filling buffer error - page 154>
2106 900D                 else
2108 3039 0096              sta r3,it(r1)        ;timestamp buffer.
210C 3059 0090              sta r5,bufe(r1)      ;save end pointer
2110 7078 00D2              setmap m2,mapv2
2114 3078 FC04
2118 30AC 004C              sta r2,@emiq(r4)     ;queue onto software queue.
211C 302C 004C              sta r2,emiq(r4)
                          endif
                        endif
2120 300C 4048          unlock lmiq(r4)          ;release software queue.
                      endif
                    endif
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 156
MODEM.PLR;2     PAGE 15          IMP Modem IO Code

                ;process software queue until empty or some work done.

     0001       .if z LBig
                  setmap m0,mappkg              ;set map 0 to pkg page if not code map.
                .endc

                  repeat
2124 481C 004A      lda r1,=smiq(r4)            ;come here if hardware is busy
2128 7071           lda r7,(r1)
212A 993A           until odd
212C 707C 4000      lock lockm(r4)
2130 9AFE
                ;       someday, 'bz m2ihw'
2132 707C 4048      lock lmiq(r4)
2136 9AFE
2138 4078 154A      call deque,whm2i
213C 8000
213E 8A05           if fail
2140 300C 4048        unlock lmiq(r4)
2144 3004             unlock (r4)  ;;lockm
2146 902C             break
                    endif
2148 300C 4048      unlock lmiq(r4)
214C 7079 0090      lda r7,bufe(r1)             ;end pointer from endi
2150 4971           sub r7,r1                   ;;m2 ;bumming sign bit from hardware
2152 9B06           if minus } r7 > =bufend-hrdoff } r7 < =srch
2154 4E78 008C
2158 9C03
215A 4EF6
215C 8207
215E 4EF6             if r7 < =srch
2160 8203
                        Trap 2003,<;Modem input short (BAD DATA INPUT LOST) - page 155>
2162 E403
155
2164 9002               else
                        Trap 2001,<;Modem bad end ptr *** CALL MAINT *** (H) - page 155>
2166 E401
 155
                      endif
```

```
2168 9016            else                          ;end ptr ok.
216A 3079 0090        sta r7,bufe(r1)              ;save buffer length.
                    micumi:                        ;Cum Stats Modem Inputs Hook
216E 3049 0092        sta r4,inch(r1)              ;save input channel.
2172 4078 17D4        call pcksubr                 ;checksum packet.
2176 9A07             ifnot zero
2178 E410               Trap 2020,<;Modem s/w failure - page 156>
217A 4078 41AA          call ckqput
217E 3004               unlock (r4)  ;;lockm
2180 90D2               next
2182 9009              else
2184 7031               lda r3,(r1)                 ;;neth
2186 7059 0002          lda r5,typh(r1)
218A A3D4               rll r5,4
218C 4BDE               and r5,=<paktyp+combit>_- D12   ;ready to dispatch
218E 40FD 21A0          call @m2idsp(r5)            ;on packet type
2192 8A06               break if succeed
                       endif
                     endif
2194 3004            unlock (r4)  ;;lockm          ;free modem lock before flush.
2196 4078 131E       call flushb,whm2i
219A 8000
219C 90C4          endrepeat
219E 6006        endroutine m2i


21A0             table m2idsp                      ;dispatch on packet type.
21A0 21D2          m2ireg                          ;Regular Data packet.
21A2 07D8          rfail                           ;Incompatible Regular packet.
21A4 21D2          m2ireg                          ;Control packet.
21A6 07D8          rfail                           ;Incompatible Control packet.
21A8 490E          m2irut                          ;Routing and Null packets.
21AA 490E          m2irut                          ;same with combit set
21AC 48B6          m2irld                          ;Reload packets.
21AE 07D8          rfail                           ;Incompatible Reload packet.
                 endtable m2idsp
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 158
MODEM.PLR;2     PAGE 17          IMP Modem IO Code


                ;routine to get a new input buffer and set it up
                ;as the next buffer

21B0 1076       routine m2inxt, arg r4, uses r1-r2, result r2
21B2 4078 13C6    call freget,whm2i+nmi          ;get next free buffer.
21B6 8000
21B8 8A06         if fail                        ;no free buffer
21BA 48F1           lda r7,=1                    ;count times this happens
21BC 3278 64AE      addm r7,mnobuf
                micume:                          ;cumstats modem errors hook.
21C0 7028 63CA      lda r2,junk                  ;read next into junk.
                  endif
21C4 302C 0046    sta r2,nxtbf(r4)               ;next input buffer
21C8 7078 00D2    setmap m2,mapv2                ;fix map 2
21CC 3078 FC04
21D0 6006       endroutine m2inxt

                ;r1 = packet buf pointer, r2 = chain, r3 = neth, r4 = modem block.

21D2 1076       routine m2ireg, arg r1-r4
21D4 787C 0002    ldab r7,phflag(r4)
21D8 9903         ifnot odd              ;;master
21DA 4D38 4000      eor r3,=endbit               ;he's higher imp
                  endif
21DE 4FF2         if r7 .bit. =inhrst            ;in reset
21E0 9A04
21E2 6076           fail return
21E4 4FF0
21E6 4007
                  endif
21E8 4F38 4000    if r3 .bit. =endbit            ;line suddenly looped.
21EC 9A05
21EE E5C2           Trap 2702,<;Suddenly looped line - page 157>
21F0 4078 1CF4      call killin
21F4 90F7           fail return
                  endif
21F6 4FF4         if r7 .nbit. =phup
21F8 8A01
                ;   Trap 2710,<;Accepting pkt on dead line - page 157>
                  endif
21FA 4078 2200    call doak
21FE 6006       endroutine m2ireg
```

```
FC00 0400                Page PkgCode

48B6 1076        routine m2irld, arg r2, arg r4

48B8 7079 0002     if typh(r1) .bit. =demrel     ;demand reload packet?
48BC 4F78 0100
48C0 9A13
48C2 7079 0008       if seqh(r1) = =dempas       ;yes, password correct?
48C6 4E78 CAFE
48CA 810B
48CC 4838 FFFE         lda r3,=n.reload          ;yes, assume nice reload
48D0 7059 0006         lda r5,srch(r1)
48D4 9B04              ifnot minus               ;panic demand?
48D6 3058 651E           sta r5,nstlin           ;yes, start with this line
48DA 48B2                lda r3,=p.reload        ;panic reload flag
                       endif
48DC 3038 651C         sta r3,nsrtf
                     endif
48E0 6076            fail return               ;always flush this packet
48E2 4FF0
48E4 4007
                   endif
48E6 3004          unlock (r4)  ;;lockm
48E8 4078 48A4     call pkchold                ;no modem resets please
48EC 4078 146C     call wheorb,whm2i?whf2h
48F0 C000
48F2 7078 AF48     lock lockfd+coriob
48F6 9AFE
48F8 30A8 656A     sta r2,@efhcq
48FC 3028 656A     sta r2,efhcq
4900 3008 AF48     unlock lockfd+coriob
4904 7878 6F43     ldab r7,bhpid+coriob        ;give fake jam 2 a wakeup
4908 30F8 00AC     sta r7,@pid
490C 6006        endroutine m2irld
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 160
MODEM.PLR;2     PAGE 19          IMP Modem IO Code


                ;M2IRUT - all type 2 (routing) messages come here
                ;They are further subdivided by subtype
                ;into nulls, old routing, new line up/down
                ;and spf updates.

490E 1076       routine m2irut, arg r1, arg r4, uses r5
4910 7859 0002    ldab r5,typh(r1)              ;pkt type word
4914 4BD7         and r5,=pflags_- H8
4916 A2D1         sll r5,1                      ;now a good offset
4918 40FD 4926    call @rutdsp(r5)
491C 8A04         if fail
491E 6076           fail return
4920 4FF0
4922 4007
                  endif
4924 6006       endroutine m2irut

4926            table rutdsp                    ;dispatches for type 2 messages
4926 07D8               rfail                   ;old routing (ignored)
4928 4936               m2inul                  ;null
492A 4940               m2ihihy                 ;new line up/down
492C 4A10               m2irup                  ;spf update
492E 07D8               rfail
4930 07D8               rfail
4932 07D8               rfail
4934 07D8               rfail
                endtable rutdsp

                ;routines to process null and routing cases of routing packets.

4936 1076       routine m2inul                  ;null with acks
4938 7031         lda r3,(r1)           ;;neth
493A 4078 2200    call doak
493E 6006       endroutine m2inul
```

```
  .comnt |
  modem to imp routine to decipher hello (ihy) pkt and set flags in
  the modem block for DEDL. Called from M2I with:
  r1 - buffer pointer
  r4 - modem block
  m2 set up for the pkt
  |

4940 1076       Routine M2IHIHY, arg r1/r4/m2/m3, uses r3/r5, local r2
4942 1026

4944 782C 0002  setb r2 = phflag(r4)            ;get line state
4948 7839 0000  ldab r3,neth(r1)
494C 383C 009F  stab r3,rmodn(r4)
4950 7039 0006  lda r3,srch(r1)                 ;neighbor's IMP number
4954 787C 001A  ldab r7,modem(r4)               ;doubled modem number
4958 303F 63A6  sta r3,m2nghb(r7)               ;update one of the neighbor tables
495C 7E78 72BD  if byte r7 = thd                ;line to accept sync on?
4960 8105
4962 7079 0008    lda r7,seqh(r1)               ;yes, copy network sync time
4966 3078 62DE    sta r7,sync
                endif
496A 7E3C 0005  if byte r3 <> neigh(r4)         ;neighbor change?
496E 910D
4970 787C 0005    ldab r7,neigh(r4)             ;yes, get old neighbor
4974 9A08         ifnot zero                    ;was there a neighbor?
4976 E5C6           Trap 2706,<;Neighbor IMP number changed - page 160>
4978 4078 1CF4      call killin                 ;yes, kill line (zeroes neigh)
497C 6026           fail return
497E 6076
4980 4FF0
4982 4007
                  endif
4984 383C 0005    stab r3,neigh(r4)             ;no, record new neighbor
                endif


                ;more m2ihihy over leaf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 162
MODEM.PLR;2     PAGE 21          IMP Modem IO Code


                ;M2IHIHY continued

4988 782C 0002  ldab r2,phflag(r4)              ;get line state
498C 4FA2       if r2 .nbit. =inhrst            ;don't change lnei during reset
498E 8A1B
4990 7638 62E2    if r3 <> mine                 ;multiple looped lines okay
4994 9116
4996 1026           save r2                     ;for a temp
4998 4828 0010      lda r2,=nmd*words           ;search for duplicates in lnei
                    repeat
499C 505A 635E        lda r5,m2pblk(-r2)                ;modem block address
49A0 9B0E             ifnot minus } r5 = r4     ;don't check this line
49A2 4E54
49A4 910C
49A6 7E3D 0004          if byte r3 = lnei(r5)   ;match?
49AA 8109
49AC 787D 0002            ldab r7,phflag(r5)    ;yes, so get line state
49B0 4FF4                 if r7 .bit. =phup     ;is line up?
49B2 9A03
49B4 6026                   restore r2          ;yes. have double line to
49B6 90E3                   fail return         ;neighbor, leave this one down
                          endif
49B8 787D 4004            ldab r7,lnei+foo(r5)  ;line dead, clear lnei
                        endif
                      endif
49BC 88F0             until loop
                    endrepeat
49BE 6026           restore r2  ;;phflag        ;restore current line state
                  endif
49C0 383C 0004    stab r3,lnei(r4)              ;update this line's lnei
                endif
49C4 48F1       lda r7,=1
49C6 327C 00B6  addm r7,gotihy(r4)              ;count a received Hello/IHY

                ;(OVER)
```

```
                ;M2IHIHY continued

49CA 7059 0002  lda r5,typh(r1)                 ;get protocol bits
49CE 4BD5       and r5,=phup}master
49D0 7638 62E2  if r3 = mine                    ;is line looped?
49D4 8102
49D6 4DD1         eor r5,=master                ;reverse sense
                endif
49D8 4D25       eor r2,r5
49DA 4FA3       if r2 .nbit. =master+inhrst
49DC 8A05
49DE E5C5         Trap 2705,<;Master/slave mismatch - page 162>
49E0 4078 1CF4    call killin                   ;kill line
49E4 90CC         fail return
                endif
49E6 48B8       lda r3,=rcdhel                  ;mark we received hello/ihy
49E8 7C3C 0002  iorb r3,phflag(r4)              ;for timeout processing
49EC 990F       ifnot odd       ;;master
49EE 4C38 0080    ior r3,=shihy                 ;slave must send immediately
49F2 4078 14E8    call pokem                    ;set i2mpok and poke modem
49F6 4D53         eor r5,r3                     ;check that states match
49F8 4FD4         if r5 .bit. =phup             ;line states different in master
49FA 9A08
49FC 4FB4           if r3 .bit. =phup           ;was slave up?
49FE 9A04
                    ;    Trap 2015,<;Slave obeys master down - page 162>
4A00 4078 1CF4        call killin               ;yes, must go down now
4A04 90BC             fail return
                    endif
4A06 4078 4724      call lineup                 ;no, then time to bring it up
                  endif
                endif
4A0A 3C3C 0002  iorbm r3,phflag(r4)             ;update line  flags
4A0E 90B7       fail return                     ;always flush packet

                endroutine m2ihihy
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 164
MODEM.PLR;2     PAGE 23          IMP Modem IO Code


.comnt | M2I SPF Routines

M2IRUP - Process input routine update

Determine which lines update should be sent out.
Determine whether to accept update for processing.
Acks retransmissions, floods new updates.
|

4A10 1076       Routine m2irup, arg r1-r2, arg r4, uses r3, uses r5

4A12 1026       push r2                         ;save r2
4A14 48B0       lda r3,=0                       ;assume packet is not retrans
4A16 7051       lda r5,(r1)             ;;neth
4A18 8B0A       if minus                ;;rupret
4A1A 4838 8000    lda r3,=rupret                ;packet is a retransmission
4A1E 3131         subm r3,(r1)          ;;neth  ;turn retry bit off
4A20 3239 0004    addm r3,chkh(r1)              ;fix checksum
4A24 783C 001A    ldab r3,modem(r4)             ;modem number
4A28 703B 0EB6    lda r3,bittab(r3)             ;bit for our line, send ack back
                endif
4A2C 7829 0007  ldab r2,srch+1(r1)      ;;rupsrc ;source IMP number
4A30 7628 62E2  if r2 = mine
4A34 8110
4A36 7078 72A6    ifnot iniflg                  ;somebody else has our number}
4A3A 8A0D
4A3C 3004           unlock (r4) ;;lockm         ;nothing we should do
4A3E 3008 6518      set nspc                    ;remeber us causing this
4A42 48F2           set nsrtf = =p.reload       ;panic into loader
4A44 3078 651C
4A48 4878 0BAD      set nsftof = # HOBAD        ;a defunct boston newspaper
4A4C 3078 6516
4A50 4008 1072      jmp loopmv                  ;let timeout kill it for real
                  endif
                endif
4A54 A2A1       sll r2,1                        ;double IMP number
4A56 1026       save r2                         ;preserve IMP number
4A58 4078 4A90  call m2irck                     ;validate update
4A5C 9A05       if succeed                      ;succeed on ACK or new update
4A5E 3008 B2C6    unlock spfrtl                 ;done with spfrut
4A62 4078 593C    call rtoff                    ;ack or new: stop timer
                endif
4A66 6016       pop r1                          ;IMP*2 to R1
4A68 6026       pop r2                          ;get back original r2
4A6A 4EB0       if r3 <> =0                     ;if anyone must process
4A6C 910F
4A6E 3004         unlock (r4)  ;;lockm          ;unlock m2i blk, no more fails
4A70 4078 25E8    call ncmove,cntrs+nrut        ;move into routing count
4A74 63E0
4A76 8A04         if fail                       ;no room in routing
4A78 48F1           lda r7,=1                   ;take it anyway, but
4A7A 3278 5EAC      addm r7,rupnbf              ;count occurances
                  endif
4A7E 4078 146C    call wheorb,whm2i}whrut       ;change ownership
4A82 8080
```

```
4A88 6006         return
                  endif
4A8A 6076         fail return                    ;no one should process, flush
```

```
4A8C 4FF0
4A8E 4007

                endroutine m2irup
```

```
.comnt |
M2IRCK - validate SPF update using age and serial number.

Locks SPFRTL and returns with it still locked if succeed.
Fail return on old update.
Succeed with new update (serial > current)
or ack or rebound (serial = current).
Updates spfrut with new serial number and age if new.

r1 - buffer pointer
r2 - source IMP number (doubled)
r3 - mask bits for who should process update
r4 - modem block
r5 - neth
|

4A90 1076       routine M2IRCK, arg r1-r5, result r3, uses r5

4A92 4F58 7000  if r5 .nbit. =rupage
4A96 8A04
4A98 6076         fail return                   ;age of update zero, bug?
4A9A 4FF0
4A9C 4007
                endif
4A9E 7059 0006  lda r5,srch(r1)                 ;get update's serial number
4AA2 4B58 3F00  and r5,=rupsno                  ;keep aligned in left byte
4AA6 7078 B2C6  lock spfrtl                     ;don't accept on two modems
4AAA 9AFE
4AAC 707A 72FC  lda r7,spfrut(r2)               ;get current age and serial
4AB0 4F78 0070  if r7 .bit. =spfage             ;if our age zero, accept update
4AB4 9A12
4AB6 4B78 3F00    and r7,=spfsno                ;not zero, check serial number
4ABA 4975         sub r7,r5                     ;current - input
4ABC 8A05         if zero                       ;input = current?
4ABE 48F1           lda r7,=1                   ;yes
4AC0 3278 5E8C      addm r7,rupdp1              ;add to duplicate count
4AC4 6006           return                      ;ack or rebound, just return
                  endif
4AC6 4F78 2000    if r7 .nbit. =rupsho          ;within half the range?
4ACA 8A07
4ACC 48F1           lda r7,=1                   ;no
4ACE 3278 5E8A      addm r7,rupobs              ;count an obsolete update
4AD2 3008 B2C6      unlock spfrtl
4AD6 90E1           fail return                 ;input must be old
                  endif
                endif                           ;update is acceptable
4AD8 7C51       iorb r5,(r1)            ;;neth  ;combine new age and serial
4ADA 755A 72FC  eor r5,spfrut(r2)               ;insert into spfrut
4ADE 4B58 3F70  and r5,=spfage+spfsno
4AE2 755A 72FC  eor r5,spfrut(r2)
4AE6 715A 72FC  sub r5,spfrut(r2)               ;delta sum
4AEA 325A 72FC  addm r5,spfrut(r2)
4AEE 3158 72C4  subm r5,spfsum                  ;adjust checksum
4AF2 7038 72AA  set r3 = rupmsk                 ;flood new update out all lines

4AF6 90F        endroutine m2ircк
```

```
                    Page LCode

                ;process acknowledgements. r1= buffer pointer, r2 = point
                ;word, r3 = neth, r4= modem block.
2200 1076       routine doak, arg r1-r4
2202 707C 001C    lda r7,maxchn(r4)
2206 4E78 0010    if r7 <= = D8*2               ;8 and 16 channel special case.
220A 9C0D
220C 4853           lda r5,r3           ;;neth
220E 7D5C 0053      eorb r5,tsex+1(r4)          ;get chan 8-15 ACKs
2212 3D5C 0053      eorbm r5,tsex+1(r4)         ;and mark TSEX
2216 A2D8           sll r5, H8                  ;to high byte
2218 A4B7           sra r3, H7                  ;CHAN * 2 now
221A 4B38 801E      and r3,=odeven+<chanum_-7>  ;remove extraneous bits
221E 48F0           lda r7,=0                   ;always ACK group 0
2220 4008 224A   jmp newloop
                endif
2224 707C 001C    lda r7,maxchn(r4)
2228 4E78 0010    if r7 <= = D8*2               ;8 and 16 channel special case.
222C 9C0C
222E 4853           lda r5,r3           ;;neth
2230 7D5C 0053      eorb r5,tsex+1(r4)          ;get chan 8-15 ACKs
2234 3D5C 0053      eorbm r5,tsex+1(r4)         ;and mark TSEX
2238 A2D8           sll r5, H8                  ;to high byte
223A A4B7           sra r3, H7                  ;CHAN * 2 now
223C 4B38 801E      and r3,=odeven+<chanum_-7>  ;remove extraneous bits
2240 48F0           lda r7,=0                   ;always ACK group 0
2242 9004       else
2244 48D0           lda r5,=0                   ;no high ACKs
2246 7871           ldab r7,(r1)        ;;neth
2248 4BFF           and r7,= HF                 ;ack group
                endif

                ;r1:buffer, r3:chan, r5:high acks, r7:Ack Group.
                newloop:
224A 1076         push r7                       ;save ack group
224C 4A7C 0052    add r7,=tsex(r4)              ;point to right TSEX
2250 7C59 0003    iorb r5,typh+1(r1)            ;pick up low ACKs
2254 7D57         eorb r5,(r7)  ;;tsex
2256 3D57         eorbm r5,(r7) ;;tsex          ;and mark TSEX changes
2258 1056         push r5                       ;save acks for us to do
225A 7079 0002    lda r7,typh(r1)               ;this a null?
225E 8B06         if minus      ;;ruttyp
2260 3004           unlock (r4) ;;lockm
2262 4078 131E      call flushb,whm2i
2266 8000
2268 902F         else
226A 4853           lda r5,r3                   ;now check for duplicates
226C A6D4           srl r5,4
226E 4BDF           and r5,= HF                 ;128 chans
2270 4A5C 0072      add r5,=infree(r4)
2274 787D FFF0      ldab r7,rsex-infree(r5)
2278 4833           lda r3,r3
227A 8B03           if minus    ;;odeven
227C 4D78 00FF        eor r7,# HFF
                    endif
```

```
2280 4BBE            and r3,= HE
2282 737B OEB6       and r7,bittab(r3)
2286 7F75            if byte r7 .bit. (r5)        ;;infree
2288 9A0F
228A 3D75              eorbm r7,(r5)              ;;infree
228C 3004              unlock (r4)  ;;lockm
228E 495C 0072         sub r5,=infree(r4)         ;just ack group
2292 A2D8              sll r5, H8                 ;to LH
2294 4C57              ior r5,r7                  ;include bits
2296 4078 146C         call wheorb,whm2i?whtsk
229A 8001
229C 305A 0350         sta r5,chan-chain(r2)      ;signal to task
22A0 4078 14C6         call tskput
22A4 9011             else
22A6 4EF0              if r7 = =0                         ;true duplicate?
22A8 810B
22AA 495C 0072           sub r5,=infree(r4)               ;yes, get ack group
22AE 48F1                lda r7,=1
22B0 A275                sll r7,r5
22B2 347C 0030           iorm r7,snull(r4)                ;ack on duplicate
22B6 787C 000E           ldab r7,motpid(r4)
22BA 30F8 00AC           sta r7,@pid                      ;wake up i2m to send them
                       endif
22BE 3004              unlock (r4)  ;;lockm
22C0 4078 131E         call flushb,whm2i
22C4 8000
                     endif
                   endif
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 169
MODEM.PLR;2     PAGE 27          IMP Modem IO Code

               ;done with r2 - not using pointer to the packet either.


22C6 6016          pop r1                        ;acks to do
22C8 6036          pop r3                        ;restore channel number.
22CA A0B4          sla r3,4       ;:carry        ;base of ACK group *2 also clears carry
22CC 4E90          if r1 <> =0
22CE 913D
                     repeat
                       repeat
22D0 A591                rra r1,1                ;odd bit goes to carry
22D2 9403                until carry
22D4 4AB2                add r3,=2
22D6 90FD              endrepeat
22D8 1016              push r1
22DA 4818 0100         lda r1,=whi2m
22DE 482C 0034         lda r2,=ssentq(r4)
22E2 707C 4000         lock lockm(r4)
22E6 9AFE
                       begin findackedpacket
                         repeat
22E8 4852                  lda r5,r2             ;locate the packet
22EA 7025                  lda r2,(r5)
22EC 9914                  until odd             ;exhausted SENTQ
22EE 771A 06A0             if r1 .nbit. where-chain(r2)
22F2 8A05
22F4 48A1                    lda r2,=1
22F6 3025                    sta r2,(r5) ;:odd   ;we're flushing no packet
22F8 E40A                    Trap 2012,<;bad sentq - page 168>
22FA 900D                    break
                           endif
22FC 763A 0350             if r3 = chan-chain(r2)
2300 8109
2302 7072                    lda r7,(r2)
2304 3075                    sta r7,(r5)
2306 8903                    if odd
2308 305C 0036                 sta r5,esentq(r4)
                             endif
230C 4078 234A               call clrchn
2310 9013                    leave findackedpacket
                           endif
2312 90EB                endrepeat

2314 702C 4026           lda r2,snding+foo(r4)   ;check buffer being sent
2318 9A09                if NZero & r3 = chan-chain(r2)  ;got buffer
231A 763A 0350
231E 8106
2320 302C 0028             sta r2,later(r4)      ;flush when done
2324 4078 234A             call clrchn           ;fix ack bits
2328 9006                 else
232A 302C 0026             sta r2,snding(r4)     ;save this for future use
232E E414                  Trap 2024,<;Unexpected ACK - page 168>
2330 4078 1CF4             call killin           ;kill line now
                         endif
2334 48A1                lda r2,=1
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 170
MODEM.PLR;2     PAGE 28          IMP Modem IO Code

                ;here after finding (or not) acked packet

2336 3004             unlock (r4)  ;;lockm
2338 4822             lda r2,r2
233A 9904             ifnot odd                 ;got a buffer to flush
233C 4078 1332          call flush,whi2m        ;off modem retransmit queue
2340 0100
                      endif
2342 4AB2             add r3,=2 ;;carry         ;clears carry
2344 6016             pop r1
2346 8AC5             until zero
                    endrepeat
                  endif

2348 6006       endroutine doak
```

```
                ;clear channel busy bit
234A 1076       routine clrchn, arg r2-r4, uses r1
234C 4813         lda r1,r3                     ;clear chnbsy bit
234E A694         srl r1,4
2350 4A1C 0082    add r1,=chnbsy(r4)
2354 4873         lda r7,r3
2356 4BFE         and r7,= HE
2358 707F 0EB6    lda r7,bittab(r7)
235C 3D71         eorbm r7,(r1)                 ;;chnbsy
235E 48F1         lda r7,=1
2360 327C 0050    addm r7,slots(r4)
2364 327C 0040    addm r7,thrupt(r4)
2368 4078 1512    call unpack
236C 7071         if (r1) .nbit. =dscpkt ;;neth ;don't do delay for discards
236E 4F78 2000
2372 8A03
2374 4078 2382      call cdelay
                  endif
                trcm2i:                         ;hook for trace package
2378 7078 00D2    setmap m2,mapv2
237C 3078 FC04
2380 6006       endroutine clrchn
```

```
.INSERT "DEL"
    .INSRT DEL
```

```
                        .stitl Packet Delay Computation

                        page Dummy
0000                .=0
0000                kb32:   .blkw 1
0002                kb48:   .blkw 1
0004                kb64:   .blkw 1
0006                kb128:  .blkw 1
0008                kb256:  .blkw 1
000A                kb390:  .blkw 1
FC00 0400                   Page RutCode

4AF8                  table tdtab           ;pointers to tdel tables for each speed.
4AF8 4B04               tb32kb              ;32 Kbps
4AFA 4B14               tb48kb              ;48 Kbps
4AFC 4B24               tb64kb              ;64 Kbps
4AFE 4B34               tb128kb             ;128 Kbps
4B00 4B44               tb256kb             ;256 Kbps
4B02 4B54               tb390kb             ;390 Kbps
                      endtable tdtab

4B04                  table tb32Kb          ;32 kb tdel table.
4B04 004E               .word  D78
4B06 0077               .word  D119
4B08 009F               .word  D159
4B0A 00CB               .word  D203
4B0C 00F4               .word  D244
4B0E 011C               .word  D284
4B10 0148               .word  D328
4B12 0171               .word  D369
                      endtable tb32Kb

4B14                  table tb48Kb          ;48 Kb tdel table.
4B14 0034               .word  D52
4B16 004F               .word  D79
4B18 006A               .word  D106
4B1A 0087               .word  D135
4B1C 00A2               .word  D162
4B1E 00BD               .word  D189
4B20 00DA               .word  D218
4B22 00F5               .word  D245
                      endtable tb48Kb

4B24                  table tb64Kb          ;64 Kb tdel table.
4B24 0027               .word  D39
4B26 003B               .word  D59
4B28 004F               .word  D79
4B2A 0065               .word  D101
4B2C 0079               .word  D121
4B2E 008E               .word  D142
4B30 00A4               .word  D164
4B32 00B8               .word  D184
                      endtable tb64Kb

4B34                  table tb128Kb         ;128 Kb tdel table.
4B34 0014               .word  D20
4B36 00                 .word  D30
```

```
4B38 0028          .word  D40
4B3A 0032          .word  D50
4B3C 003C          .word  D60
4B3E 0046          .word  D70
4B40 0050          .word  D80
4B42 005A          .word  D90
                 endtable tb128Kb

4B44               table tb256Kb          ;256 Kb tdel table.
4B44 000A          .word  D10
4B46 000F          .word  D15
4B48 0014          .word  D20
4B4A 0019          .word  D25
4B4C 001E          .word  D30
4B4E 0023          .word  D35
4B50 0028          .word  D40
4B52 002D          .word  D45
                 endtable tb256Kb

                 table tb390Kb          ;390 Kb tdel table.
```

```
4B54
4B54 0007          .word  D7
4B56 000A          .word  D10
4B58 000D          .word  D13
4B5A 0010          .word  D16
4B5C 0014          .word  D20
4B5E 0017          .word  D23
4B60 001A          .word  D26
4B62 001D          .word  D29
                 endtable tb390Kb


                 ;smoother parameters.

     0008        thrini= D8       ;thresh init value
     0002        thrdcy= D2       ;thresh decay value
     0186        dsfreq= D10*< D10000/ D256>     ;smoother calling frequency(10 seconds)
     0003        delshf=3         ;places to right shift packet delay
     0003        smoshf=3         ;places to right shift average delay
     0002        srtshf=2         ;place to shift from packet delay (6.4 ms)
                                  ;to routing retransmit time (25.6 ms units)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 175
DEL.PLR;1       PAGE 3  Packet Delay Computation


.comnt |
CDELAY - calculate delay for a packet through the IMP.
Delay = sent time - arrival time + transmission delay
      + propagation delay
Delay is computed in 800 microsecond units and then right
shifted by delshf and added into the double precision sum.
The packet count is also increased by one.
|

                page LCode
2382 1076       Routine CDELAY, arg r1, local r1-r2, arg r4
2384 1016
2386 1026

2388 7029 009A  lda r2,st(r1)                   ;sent time
238C 7129 0096  sub r2,it(r1)                   ;arrival time
2390 787C 001B  ldab r7,clockm(r4)              ;which modem clock to use.
2394 7019 0090  lda r1,bufe(r1)                 ;buffer length in bytes
2398 499E       sub r1,=data-hrdoff             ;don't count header
239A A693       srl r1,3                        ;<data word length/8>*words
239C 721F 4AF8  add r1,tdtab(r7)                ;transmission delay entry
23A0 7221       add r2,(r1)                     ;add in transmission delay
23A2 A6A3       srl r2,delshf                   ;divide by power of 2
23A4 4891       lda r1,=1                       ;add in delay for packet
23A6 707C 4092  lock dlock(r4)                  ;lock delay struct.
23AA 9AFE
23AC 322C 0098  addm r2,dellow(r4)              ;add packet delay to cum delay.
23B0 8403       if carry
23B2 321C 0096    addm r1,delhi(r4)             ;if needed, increase high delay.
                endif
23B6 321C 0094  addm r1,dpcnt(r4)               ;increment packet count
23BA 300C 4092  unlock dlock(r4)

23BE 6026       endroutine CDELAY
23C0 6016
23C2 6006
```

```
FC00 0400               page RutCode

.Comnt |
TOSMOOTH - controls calling of delay modules.
Times averaging interval and at the end of the interval calls
SMOOTH for each line and then calls COMPAR.
|

4B64 1076       Routine TOSMOOTH, uses r1-r5

4B66 7058 72B6  lda r5,smoclk
4B6A 49D2       sub r5,=words                   ;tick another fast tick
4B6C 9B07       ifnot minus
4B6E 4E58 0010    if r5 < =nmd*words            ;end of interval?
4B72 8203
4B74 4078 4B8C      call smooth                 ;yes, smooth for line in r5
                  endif
4B78 9007         else
4B7A 4078 4C14    call compar                   ;all lines have been done,
4B7E 4858 030C    lda r5,=dsfreq*words          ;compare, then reset clock
4B82 4078 23C4    call rutsleep                 ;sleep before going on
                endif
4B86 3058 72B6  sta r5,smoclk

4B8A 6006       endroutine TOSMOOTH
```

```
.comnt |
SMOOTH - compute average packet delay for a line.
R5 contains modem number (doubled) for line to average.
Sum of individual packet delays and the count of packets
on the line have been recorded by CDELAY.
Delay total has to be scaled to 16 bits from 32.
Then divide is called to compute the average.
The result is then shifted by smoshf compensated by
the number of shifts done prior to dividing to scale to
16 bits.
|

4B8C 1076       Routine SMOOTH, arg r5, uses r1-r4

4B8E 704D 635E  lda r4,m2pblk(r5)               ;get modem blk address
4B92 9B22       ifnot minus
4B94 787C 0002    ldab r7,phflag(r4)            ;line state bits
4B98 4818 00FF    lda r1,=dlinf         ;delay on dead line
4B9C 4FF4         if r7 .bit. =phup             ;is it UP?
4B9E 9A1A
4BA0 707C 4092      lock dlock(r4)              ;yes, get delay sum and count
4BA4 9AFE
4BA6 701C 4098      lda r1,dellow+foo(r4)       ;clear and get low 16 bits
4BAA 702C 4094      lda r2,dpcnt+foo(r4)        ;count of packets
4BAE 703C 4096      lda r3,delhi+foo(r4)        ;high 16 bits of delay sum
4BB2 300C 4092      unlock dlock(r4)
4BB6 4078 4BD8      call davrge
4BBA 761D 4142      if r1 < bias (r5)           ;Don't report too small a delay
4BBE 8203
4BC0 701D 4142        lda r1,bias (r5)
                    endif
4BC4 761D 4152      if r1 > delmax (r5) ;nor too large a delay either
4BC8 8C03
4BCA 701D 4152        lda r1,delmax (r5)
                    endif
4BCE 4078 23C4      call rutsleep               ;time for a rest
                  endif
4BD2 381C 009A    stab r1,delave(r4)            ;current average delay
                endif

4BD6 6006       endroutine smooth
```

```
.comnt |
DAVRGE - calculate average packet delay from total delay
and total packet count.
First scale total delay to 16 bits and remember number of
places shifted for post divide shift.
r1 is low 16 bits of delay
r3 is high 16 bits of delay
r2 is packet count
|

4BD8 1076       Routine DAVRGE, arg r1-r3, result r1, local r5
4BDA 1056
4BDC 48C0       clear r4
4BDE 4EB0       repeat until r3 = =0            ;shift out all bits in r3
4BE0 9106
4BE2 0208         clear carry                   ;carry used in linked shift
4BE4 A5B1         rra r3,1                      ;next bit into carry
4BE6 A591         rra r1,1                      ;pick it up in r1
4BE8 4AC1         add r4, =1
4BEA 90FA       endrepeat
4BEC 4EA0       if r2 ==0                       ;no samples at all
4BEE 810B
4BF0 704D 635E    lda r4, m2pblk(r5)            ;get parameter block
4BF4 781C 001B    ldab r1, clockm(r4)           ;line speed
4BF8 7019 4AF8    lda r1, tdtab(r1)             ;its transmission delay table
4BFC 7019 000A    lda r1, 5*words(r1)           ;6th entry for typical packet
4C00 48C0         lda r4, =0                    ;no shifting please
4C02 9003       else
4C04 4078 1600    call divide                   ;compute average
                endif
4C08 A214       sll r1, r4                      ;correct for pre-shift
4C0A 721D 12D6  add r1,propdl(r5)               ;now add propagation delay
4C0E A693       srl r1,smoshf
4C10 6056       endroutine davrge
4C12 6006
```

```
.comnt |
COMPAR - determine if update should be generated.
Called when all lines have been smoothed to see if any
of the current average delays have exceeded the threshhold
for reporting.  If so, set RUPSND so RUPGEN will generate
an update.  If not, decrement  thresh by the decay value.
|

4C14 1076       Routine COMPAR, uses r4-r5

4C16 4858 0010  lda r5,=nmd*words               ;look at each line
                repeat
4C1A 504D 635E    lda r4,m2pblk(-r5)            ;modem block address
4C1E 9B1E         ifnot minus
4C20 787C 009A      ldab r7,delave(r4)          ;current average
4C24 797C 009B      subb r7,delbas(r4)          ;last reported average
4C28 8B04           if minus                    ;we want absolute value
4C2A 4D78 FFFF        eor r7,=-1                ;do two's complement
4C2E 4AF1             add r7,=1
                    endif
4C30 7E78 72A9      if byte r7 >= thresh        ;exceeded thresh?
4C34 9213
4C36 4858 0010        lda r5,=nmd*words         ;yes, report new average on all lines
                      repeat
4C3A 504D 635E          lda r4,m2pblk(-r5)
4C3E 9B0A               ifnot minus
4C40 707C 4092            lock dlock(r4)        ;lock out line up/down
4C44 9AFE
4C46 787C 009A            ldab r7,delave(r4)    ;copy current to base
4C4A 387C 009B            stab r7,delbas(r4)
4C4E 300C 4092            unlock dlock(r4)
                        endif
4C52 88F4               until loop
                      endrepeat
4C54 3008 72B2        set rupsnd                ;generate an update
4C58 6006             return
                    endif
                  endif
4C5A 88E0         until loop
                endrepeat
4C5C 7878 72A9  ldab r7,thresh                  ;no line exceeded thresh
4C60 49F2       sub r7,=thrdcy                  ;so just decay it
4C62 8B02       if minus                        ;but not below zero
4C64 48F0         lda r7,=0
                endif
4C66 3878 72A9  stab r7,thresh

4C6A 90F7       endroutine compar
```

```
.INSERT "ROUTE"
    .INSRT ROUTE
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 181
ROUTE.PLR;1     PAGE 1  Routing Process

                .stitle Routing Process

.comnt |
SPF Routing Algorithm Routines
WARNING: only sufficient for 7-bit IMP numbers}
|
     0010               .radix  H10     ;let's use hex

                ;SPF data definitions

                        page RutVars

                ;Some SPF variables

5E80            rupmap: .blkw 1                 ;saved buffer map setting
5E82            rupadd: .blkw 1                 ;saved buffer address
5E84            rupend: .blkw 1                 ;end of line entries in packet

5E86            head:   .blkb 1         ;head of main process list
5E87            scn:    .blkb 1         ;scan list for subtree
5E88            subtre: .blkb 1         ;subtree list
5E89                    .blkb 1         ;unused

                        page Vars       ;collect these where visible

72A6            iniflg: .blkw 1 ;0=>no non-looped line has come up yet
72A8            ticimp: .blkb 1 ;IMP*2 last ticked by TICAGE
72A9            thresh: .blkb 1 ;current delay threshhold for sending update
72AA            rupmsk: .blkw 1 ;mask of all configured lines+rup4us
72AC            fldior: .blkw 1 ;ior of all masks on rupq
72AE            srupq:  .blkw 1 ;start of routing update queue
72B0            erupq:  .blkw 1 ;tail of routing update queue
72B2            rupsnd: .blkw 1 ;non-zero=> send an update
72B4            rretry: .blkw 1 ;non-zero -> set rupsnd next 25 ms tick
72B6            smoclk: .blkw 1 ;delay smoother clock

                ;Testing and Measurement counters

                        page RutVars

5E8A            rupobs: .blkw 1 ;number of updates judged too old by m2i
5E8C            rupdp1: .blkw 1 ;number of updates current but already seen
5E8E            rpkrec: .blkw 1 ;number of updates processed by us
5E90            rlnrec: .blkw 1 ;number of line entries in packets we processed
5E92            rqlsum: .blkw 1 ;sum of queue lengths encountered
5E94            rqlmax: .blkw 1 ;maximum queue length encountered
5E96            rupcnt: .blkw 1 ;number of updates generated
5E98            rtrcnt: .blkw nmd       ;number of retransmissions generated per line
5EA8            rtical: .blkw 1 ;number of calls to rtinc
5EAA            retcal: .blkw 1 ;number of calls to retree
5EAC            rupnbf: .blkw 1 ;number of times modem couldn't get buffer
5EAE            rpgnbf: .blkw 1 ;number of times rupgen couldn't get buffer
5EB0            rtrnbf: .blkw 1 ;number of times rtrgen couldn't get buffer

5EB2            rupqct: .blkw 1 ;current length of the routing queue
```

```
                        page Vars               ;things system sees

72B8            rutini: .blkw 1 ;0 -> routing process reinitializing

72BA            ppflag: .blkw 1         ;flag: post pass (SPFPOST) should run
72BC            dtflag: .blkb 1         ;flag: SPF tree may have changed
72BD            thd:    .blkb 1         ;line on which to accept sync
72BE            rupt25: .blkw 1         ;timer for 25 ms routing ticks
72C0            ruptck: .blkw 1         ;timer for 8 sec checksum ticks

72C2            ltbsum: .blkw 1         ;LTB checksum
72C4            spfsum: .blkw 1         ;SPFRUT checksum
                locdef spfrtl,<;Lock on common SPF tables - page 180>
                locdef rutlok,<;Lock on routing processing - page 180>
72CA            rutsp:  .blkw 1         ;saved SP for routing process
72CC            stack rut,< D25>        ;stack for routing process
72FE
72FE


                ;tables by IMP, shared with other processes:

72FE            table spfrut,words              ;indexed by IMP*2
72FE              .blkw nimp
     8000           spfded =  H8000     ;;sign  ;this IMP dead or unreachable
                    ;unused =  H4000
     3F00           spfsno =  H3F00             ;current serial number
     0100           spfsn1 =  H100              ;unit for above
                    ;unused =  H80
     0070           spfage =  H70               ;age of current update
     0010           spfag1 =  H10               ;one-bit of age field
     000F           routef =  H0F               ;index of best route line
                endtable spfrut

                ;routing update message fields
                ;;neth
     8000       rupret= H8000                   ;retry packet, must be echoed
     7000       rupage= H7000                   ;age of data when sent
     0F00       rupnn= H0F00                    ;line entries in packet
     0100         rupnn1 = rupnn-1&rupnn?rupnn  ;low bit of RUPNN

                ;;srch
     3F00       rupsno= H3F00                   ;serial number for this update
     2000       rupsho= H2000                   ;high order bit of serial number
     00FF       rupsrc= HFF                     ;source of the update

                ;line entry - first one at seqh
                ;;seqh
     FF00       rupdel= HFF00                   ;delay(distance) along this line
     00FF       rupnei= HFF                     ;neighbor at other end of line

                ;mask field - in CHAN
                ;;chan
     8000       rup4us= H8000                   ;process update ourselves
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 183
ROUTE.PLR;1     PAGE 3  Routing Process

FC00 0400               page RutCode

4C6C 0050       spfcrash:       D80             ;IMP who should crash on SPFERR

.comnt |
Top level of Routing Code
Checks for work to do:
  1. 25 ms timeout of various routing data bases:
    a. Routing retransmission timeout
    b. Delay smoother calculation
  2. Every 8 seconds - verify checksum of SPFRUT and LTB
  3. Flag (RUPSND) to generate a new routing update
  4. Routing updates to process on RUPQ
All routines are interlocked by process lock RUTLOK.
|

     00EA       ri6sec = 6*< D10000/ D256>      ;6 second delay restarting
     0138       rc8sec = 8*< D10000/ D256>      ;8 second delay checksumming

                rtloop:

4C6E 4868 72FE    lda sp,=rutstack              ;initial stack setting
4C72 48F0         clear rutini                  ;force all our lines down
4C74 3078 72B8
4C78 7058 62E2    lda r5,mine                   ;who I am
4C7C A2D1         sll r5,1                      ;doubled
4C7E 4078 53FC    call rtinit                   ;reset routing tables
4C82 7018 62DA    lda r1,time                   ;25.6 ms clock
4C86 4A18 00EA    add r1,=ri6sec
                  repeat
4C8A 4078 23D0      call rutoff                 ;dismiss us
4C8E 7618 62DA      while r1 >= time            ;await 6-second lapse
4C92 82FC
                  endrepeat
4C94 70F8 62E8    set rupt25 = @clock           ;reset our 25 ms, cksum timers
4C98 3078 72BE
4C9C 7078 62DA    set ruptck = time
4CA0 3078 72C0

                ;(main loop over)
```

```
                ;main Routing loop

                  repeat                          ;forever
4CA4 4878 0100      lda r7,# D256                 ;100 us. ticks on RTC
4CA8 7098 62E8      lda r1,@clock
4CAC 7118 72BE      sub r1,rupt25                 ;when we last ran
4CB0 9B03           if minus } r1 >= r7           ;enough time gone by
4CB2 4E17
4CB4 920C
4CB6 3278 72BE        addm r7,rupt25              ;now we ran
4CBA 4078 5748        call rtrto                  ;do the ticks
4CBE 4078 4B64        call tosmooth               ;do delay smoothing
4CC2 7078 B2B4        lda r7,m3#rretry            ;retry RUPGEN maybe
4CC6 3478 72B2        iorm r7,rupsnd
4CCA 90ED             next
                    endif
4CCC 4878 0138      lda r7,#rc8sec                ;chksum time interval
4CD0 7018 62DA      lda r1,time                   ;time now
4CD4 7118 72C0      sub r1,ruptck                 ;when to check checksums
4CD8 9B03           if minus } r1 >= r7           ;now}
4CDA 4E17
4CDC 9206
4CDE 3278 72C0        addm r7,ruptck              ;when to check again
4CE2 4078 5436        call rupcksum
4CE6 90DF             next
                    endif
4CE8 7078 B2B2      if m3#rupsnd                  ;time to generate an update
4CEC 9A04
4CEE 4078 564A        call rupgen                 ;get one
4CF2 90D9             next
                    endif
4CF4 7078 72AC      lda r7,fldior                 ;bits for RUPQ packets
4CF8 8B04           if minus                      ;got a new routing update in
4CFA 4078 4D0A        call rutspf
4CFE 90D3             next
                    endif
4D00 3008 72B8      set rutinit                   ;initialization done
4D04 4078 23D0      call rutoff                   ;dismiss us
4D08 90CE         endrepeat
```

```
.comnt |
RUTSLEEP/RUTOFF
Put the Routing process to sleep, and return to LOOP.
RUTSLEEP pokes us to resume later, RUTOFF doesn't.
Saves all registers, and map 2.
|
                        page LCode

23C4 1076       routine rutsleep,arg m1,result m0-m1/m3

23C6 4878 0044    set @pid = =rutpi
23CA 30F8 00AC

23CE 9002       entry rutoff,arg m1,result m0-m1/m3
23D0 1076

23D2 1016         save r1-r5,m2                 ;save useful stuff
23D4 1026
23D6 1036
23D8 1046
23DA 1056
23DC 7078 80BE
23E0 1076
23E2 3068 72CA    sta sp,rutsp                  ;save our stackpointer
23E6 3008 72C8    unlock m1#rutlok              ;done the process
23EA 4008 1072    jmp loopmv

                rutwak:
     0000         .if nz $maprt-$mapcd
                    setmap m0,maprut            ;get to routing code
                  .endc
23EE 7078 B2C8    lock rutlok                   ;resume the process
23F2 9AFE
23F4 8903         if odd        ;;rutlok        ;reinit or timeout
23F6 4008 4C6E      tr rtloop                   ;re-enter routing loop
                  endif
23FA 7068 72CA    lda sp,rutsp                  ;use saved SP
23FE 6076         restore r1-r5,m2              ;get back regs
2400 707F 00B0
2404 3078 FC04
2408 6056
240A 6046
240C 6036
240E 6026
2410 6016

2412 6006       endroutine rutsleep/rutoff
```

```
.comnt |
SPF - The Shortest-Path First Routing Algorithm
Builds up proper routing tables, using topology information and
line "distance" (i.e., delay or bandwidth) estimates, and then
maintains the routing as updates are processed.
|
FC00 0400              page RutCode            ;this is in common

.comnt |
SPF Database Management Routines

RUTSPF
Searches routing queue for unprocessed packets.  Looks for entries
in the database which are missing from the packet being processed
and calls SPF to declare the missing line dead.  It then calls
SPF for each line in the packet.  When finished, it disposes of
the packet and looks for another one.
|

4D0A 1076       routine rutspf, uses r1-r5

                  repeat
4D0C 4838 8000      lda r3,=rup4us              ;select packets queued for us
4D10 4078 5584      call rupdeq                 ;attempt to get a packet
4D14 9A0D           until fail                  ;no more to process, exit
4D16 4078 17D4      call pcksubr                ;checksum the packet'
4D1A 9A03           ifnot zero                  ;bad checksum}}
4D1C E407             Trap 2007,<;bad update checksum - page 184>
4D1E 9003            else
4D20 4078 4D34        call dospf
                    endif
4D24 4838 8000      lda r3,=rup4us              ;finished with packet
4D28 4078 55D2      call rupfls                 ;flush if last user
4D2C 90F0         endrepeat
4D2E 4078 23C4    call rutsleep                 ;wait here

4D32 6006       endroutine rutspf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 187
ROUTE.PLR;1     PAGE 7  Routing Process

                :DOSPF
                ;Process the updates in this packet; called with packet in
                ;R1/M2. Do pre-pass and post-pass too. Call SPF as needed

4D34 1076       routine dospf,arg r1/m2,local r2,uses r1/r3-r5/m2
4D36 1026

4D38 7078 808C    set rupmap = m2#slfptr        ;remember buffer's page
4D3C 3078 5E80
4D40 3018 5E82    sta r1,rupadd                 ;save buffer address
4D44 7871         ldab r7,(r1)  ::neth
4D46 4BFF         and r7,=rupnn_- H8            ;line entries in packet
4D48 A2F1         sll r7,1                      ;words to bytes
4D4A 4A71         add r7,r1                     ;buffer offset after updates
4D4C 3078 5E84    sta r7,rupend
4D50 7819 0007    ldab r1,srch+1(r1)            ;source IMP number
4D54 A291         sll r1,1                      ;double it
4D56 7078 00D2    setmap m2,mapv2               ;get to routing vars
4D5A 3078 FC04
4D5E 4858 03FE    lda r5,#ntbidx                ;for this source's SPF lines
4D62 7049 85DE    lda r4,ntb(r1)
4D66 4B45         and r4,r5
4D68 7359 85E0    and r5,ntb+words(r1)
4D6C 4E45         repeat until r4 = r5  ;find entries not in packet
4D6E 910E
4D70 787C 82C0      if byte ltb+ltbdst(r4) <> #dlinf    ;only do live lines
4D74 4E78 00FF
4D78 9107
4D7A 4078 4DB0        call rupsearch            ;find it in packet
4D7E 7078 00D2        setmap m2,mapv2           ;get back to routing vars
4D82 3078 FC04
                    endif
4D86 4AC2           add r4,=words               ;next line this IMP
4D88 90F2         endrepeat
4D8A 7048 5E82    lda r4,rupadd         ;now call SPF for each entry
4D8E 7648 5E84    repeat until r4 = rupend
4D92 910D
4D94 7078 5E80      setmap m2,rupmap            ;to our buffer again
4D98 3078 FC04
4D9C 683C 0008      ldab r3,seqh(r4)+           ;new distance over line
4DA0 682C 0008      ldab r2,seqh(r4)+           ;neighbor at other end
4DA4 A2A1           sll r2,1                    ;double it
4DA6 4078 4DE2      call spf                    ;process line entry
4DAA 90F2         endrepeat

4DAC 6026       endroutine dospf
4DAE 6006
```

```
                ;RUPSEARCH
                ;Make sure this update has an entry for each live
                ;line in database from its source. Emit dummy
                ;"line dead" messages to SPF for each line that has
                ;gone away. R1/source IMP*2, R4/line index

4DB0 1076       routine rupsearch,arg r1/r4/m2,uses r2/r3/m2

4DB2 782C 82C1    ldab r2,ltb+ltbnay(r4)        ;neighbor this line
4DB6 A6A1         srl r2,1                      ;un-doubled
4DB8 7078 5E80    setmap m2,rupmap              ;to update buffer again
4DBC 3078 FC04
4DC0 7038 5E82    lda r3,rupadd                 ;for all lines in update
4DC4 7638 5E84    repeat until r3 = rupend
4DC8 9107
4DCA 7E2B 0009      if byte r2 = seqh+1(r3)     ;find an update
4DCE 8102
4DD0 6006             return                    ;all's well this neighbor
                    endif
4DD2 4AB2           add r3,=words               ;to next entry
4DD4 90F8         endrepeat
4DD6 A2A1         sll r2,1                      ;re-double neighbor
4DD8 4838 00FF    lda r3,#dlinf                 ;dummy dead line
4DDC 4078 4DE2    call spf

4DE0 90F8       endroutine rupsearch
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 189
ROUTE.PLR;1     PAGE 9  Routing Process


.comnt |
SPF
Traps and ignores updates for out-of-range IMP numbers.
Ignores looped line entry, calls FNDENT to find or create entry for
both directions of the line, ignores the entry if no change over
existing entry, computes change in distance and calls RTINC if node
is reachable.  In the post-pass to RTINC, it marks nodes with
finite PDIST reachable and those with infinite, unreachable.
Unreachable nodes are severed from the tree.
|

4DE2 1076       routine spf,arg r1-r3,local r1,local r4-r5,uses r2-r3/m2
4DE4 1016
4DE6 1046
4DE8 1056

4DEA 4E90         if r1 <= =0 ! r1 > =nimp*words ! r2 <= =0 ! r2 > =nimp*words
4DEC 8C09
4DEE 4E18 00FE
4DF2 9C06
4DF4 4EA0
4DF6 8C04
4DF8 4E28 00FE
4DFC 8C03
                ;   Trap 2401,<;SPF: Ignoring illegal IMP in update - page 187>
4DFE 4078 53D0      call spferr         ;kill system for now
                ;   fail return
                ;   return                      ;don't kill SPF though
                  endif
4E02 4E12         if r1 = r2                    ;don't process looped lines
4E04 8105
4E06 6056           return
4E08 6046
4E0A 6016
4E0C 6006
                  endif
4E0E 7078 00D2    setmap m2,mapv2               ;get to routing vars
4E12 3078 FC04
4E16 4852         lda r5,r2
4E18 4821         lda r2,r1
4E1A 4815         lda r1,r5                     ;swap ends first
4E1C 4078 4EF6    call fndent                   ;check for entry for d->s
                ; if success                    ;no database errors
4E20 4812           lda r1,r2
4E22 4825           lda r2,r5                   ;swap ends back
4E24 4078 4EF6      call fndent                 ;find one for s->d
                ;   if success                  ;database okay
4E28 4853             lda r5,r3                 ;remember new line distance
4E2A 793C 82C0        subb r3,ltb+ltbdst(r4)    ;compute delta dist
4E2E 9A17             ifnot zero                ;some change
4E30 3A3C 82C0          addbm r3,ltb+ltbdst(r4) ;update this line dist
4E34 3938 72C2          subbm r3,ltbsum         ;and LTB checksum

                ;more SPF over leaf
```

```
                ;SPF continued

4E38 4E58 00FF          if r5 = =dlinf          ;it's dead}
4E3C 8103
4E3E 4838 3FFF            lda r3,=pdinf         ;use worst case path distance
                        endif
                ;;      call typupd             ;print the update
4E42 7079 87DC          lda r7,pdist(r1)        ;is source reachable?
4E46 A2F2               sll r7,2        ;;pdst
4E48 4E78 FFFC          if r7 <> =pdinf_2       ;yes, so run SPF
4E4C 9108
4E4E 4078 501C            call rtinc            ;process incremental change
                ;         if success
4E52 7078 B2BA              if m3#ppflag        ;post pass necessary?
4E56 9A03
4E58 4078 4E5E                call spfpost      ;do post pass
                ;             if success        ;done everything now
                ;               return
                ;             endif
                            endif
                ;         endif
                ;         fail return
                        endif
                      endif
                ;     return
                ;   endif
                ; endif
                ; fail return

4E5C 90D5       endroutine spf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 191
ROUTE.PLR;1     PAGE 11         Routing Process

                ;SPFPOST
                ;do the SPF post-pass
                ;look for nodes that have just become unreachable,
                ;and sever them from the routing tree.

4E5E 1076       routine spfpost,uses r1-r5

4E60 4828 00FE    lda r2,=nimp*words            ;look at each IMPs entry
4E64 4818 3FFF    lda r1,#-1?1flg               ;pre-load constants
4E68 4848 8000    lda r4,#spfded

                ;most of SPFPOST over leaf
```

```
                ;SPFPOST continued

                  repeat
4E6C 4078 23C4      call rutsleep               ;rest a bit now
4E70 7078 B2C6      lock spfrtl                 ;lock on shared data
4E74 9AFE
                    repeat
4E76 707A 87DC        lda r7,pdist(r2)          ;tree distance this node
4E7A A2F2             sll r7,2  ;;pdst          ;quick mask
4E7C 4E78 FFFC        if r7 <> #pdinf_2         ;reachable
4E80 910A
4E82 4852               lda r5,r2               ;maybe THD
4E84 507A 72FE          lda r7,spfrut+noimp0(-r2) ;;loop  ;node unreachable??
4E88 8B05               if minus                ;yes, currently
4E8A 314A 72FE            subm r4,spfrut+noimp0(r2)     ;let net acces it
4E8E 3248 72C4            addm r4,spfsum        ;and fix cksum
                        endif
4E92 9024             else                      ;not reachable now
4E94 507A 72FE          lda r7,spfrut+noimp0(-r2) ;;loop        ;marked unreachable?
4E98 9B21               ifnot minus     ;;spfded ;no, must sever from tree
4E9A 1056                 save r5               ;need another temp
4E9C 4858 03FE            lda r5,#ntbidx
4EA0 603A 85E0            lda r3,ntb+noimp0(r2)+        ;re-increment R2}
4EA4 4B35                 and r3,r5
4EA6 735A 85E0            and r5,ntb+words(r2)  ;for all line entries, this node
4EAA 4E35                 repeat until r3 = r5
4EAC 9112
4EAE 707B 85E0              lda r7,lflag(r3)    ;this line's use
4EB2 A2F1                   sll r7,1            ;how?
4EB4 8B0B                   if minus    ;;lflgb ;back link
4EB6 781B 82C1                ldab r1,ltb+ltbnay(r3)
4EBA 4078 5330                call findnay       ;find its forward line
                ;               if fail           ;some error
                ;                 unlock spfrtl
                ;                 restore r5      ;fix the stack
                ;                 fail return
                ;               endif
4EBE 4818 3FFF                lda r1,#-1?lflg   ;;lflgu
4EC2 331C 85E0                andm r1,lflag(r4) ;remove forward line
4EC6 4848 8000                lda r4,#spfded    ;restore the constant
                            endif
4ECA 231B 85E0              andm r1,lflag(r3)+  ;;lflgu ;no use for this line
4ECE 90EE                 endrepeat
4ED0 6056                 restore r5            ;get back THD candidate
4ED2 124A 72FE            addm r4,spfrut+noimp0(-r2)    ;;loop
4ED6 3148 72C4            subm r4,spfsum        ;this node unreachable
                        endif
                      endif
4EDA 4F28 001E        while r2 .bit. #1E        ;16 IMPs per pass
4EDE 8ACC
                    endrepeat
4EE0 3008 B2C6      unlock spfrtl               ;release data base
4EE4 88C4           until loop                  ;condition set in 3} places
                  endrepeat

                ;SPFPOST concludes over
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 193
ROUTE.PLR;1     PAGE 13         Routing Process

                ;SPFPOST concluded

4EE6 701D 72FC    lda r1,spfrut(r5)             ;get route for SYNC
4EEA 4B9F         and r1,=routef                ;modem to lowest IMP,.
4EEC 4991         sub r1,=1                     ;numbered from 0,
4EEE A291         sll r1,1                      ;and doubled (FE) if we're it)
4EF0 3818 72BD    stab r1,thd                   ;is THD (SYNC time source)

4EF4 6006       endroutine spfpost
```

```
.comnt |
FNDENT Call with source for line in R1 and destination for line in R2.
Returns index for line entry in r4.  Searches the line entries of
source for one with the proper destination.  If none is found, a
search is made for a usurpable dead entry, ie a line entry that has no
NAY field; OR a line entry that is dead whose corresponding reverse
direction is also dead.  In the case of a dead line whose reverse
direction is also dead, the NAY field of both directions is cleared to
indicate that they are free.  The IMP number associated with the
usurpable line entry is found by searching the NTB index table,
starting at the source line, for an index just greater than that of
the usurpable entry.
    The index of the usurpable entry and the IMP number associated with it
are used to expand the tables as necessary, ie an impnumber the same as the
source does not need to be expanded.
|

4EF6 1076       routine fndent,arg r1-r2,local r3,local r5,result r4
4EF8 1036
4EFA 1056

4EFC 4848 03FE      lda r4,#ntbidx              ;we'll look for match first
4F00 7059 85DE      lda r5,ntb(r1)              ;in source's line entries
4F04 4B54           and r5,r4
4F06 7349 85E0      and r4,ntb+words(r1)
4F0A 4E54           if r5 > r4                  ;some database error}
4F0C 8C03
4F0E 4078 53D0         call spferr              ;stop SPF then
                ;      fail return
                    endif
                    repeat                      ;search entries
4F12 4E45            until r4 = r5              ;no more entries
4F14 9109
4F16 49C2              sub r4,#words
4F18 7E2C 82C1         if byte r2 = ltb+ltbnay(r4)
4F1C 8104
4F1E 6056                 return                ;entry exists this pair
4F20 6036
4F22 6006
                       endif
4F24 90F7           endrepeat
4F26 1026           save r2                     ;save dest of new line

                ;more FNDENT over leaf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 195
ROUTE.PLR;1     PAGE 15         Routing Process

                ;FNDENT continued: can't find pair; look now for usurpable entry

4F28 49C2           sub r4,#words               ;let us increment early
4F2A 1016           push r1                     ;set this up for later
                    repeat                      ;find usurpable entry
4F2C 4AC2              add r4,=words            ;next line slot
4F2E 4E48 0320         if r4 >= #nline*words    ;oops}}
4F32 9203
4F34 4078 53D0            call spferr           ;can't find free slot
                       endif
4F38 7078 86DE         lda r7,ntbendi           ;get pointer to spare entries
4F3C 4B78 03FE         and r7,#ntbidx           ;just index
4F40 4E47              if r4 >= r7              ;spare entry, can always use
4F42 9204
4F44 4828 0100            set r2 = #ntbendi-ntb ;equiv imp number
4F48 9037              else                     ;not spare, must check
4F4A 787C 82C0            ldab r7,ltb+ltbdst(r4) ;distance across this line
4F4E 4E78 00FF            NEXT if r7 <> #dlinf  ;can't use, not dead
4F52 81ED
4F54 6026                 pop r2                ;begin search for IMP# here
                          repeat
4F56 705A 85E0               lda r5,ntb+words(r2)       ;end of lines
4F5A 4B58 03FE               and r5,#ntbidx       ;just index
4F5E 4E54                    until r5 > r4        ;line belongs to this one
4F60 9C03
4F62 4AA2                    add r2,#words        ;try next IMP
4F64 90F9                 endrepeat
4F66 1026                 save r2                ;continue from here next time
4F68 787C 82C1            ldab r7,ltb+ltbnay(r4) ;neighbor on this line
4F6C 9A21                 ifnot zero             ;not free, must check more
4F6E 4E71                    NEXT if r7 = r1 & r2 = 2(sp)       ;other end of new line
4F70 8104
4F72 762E 0002
4F76 91DB
4F78 4858 03FE               lda r5,#ntbidx       ;get to neighs lines
4F7C 703F 85DE               lda r3,ntb(r7)       ;where his lines are
4F80 4B35                    and r3,r5
4F82 735F 85E0               and r5,ntb+words(r7)       ;where his lines end
4F86 4078 23C4               call rutsleep        ;whew}, need a rest
                             repeat               ;look for other end of our line
4F8A 4E35                       if r3 >= r5       ;didn't find line}}}
4F8C 9203
4F8E 4078 53D0                     call spferr    ;bad news
                                endif
4F92 7E2B 82C1                until byte r2 = ltb+ltbnay(r3)     ;found the line
4F96 9103
4F98 4AB2                     add r3,#words
4F9A 90F8                  endrepeat
4F9C 787B 82C0             ldab r7,ltb+ltbdst(r3)
4FA0 4E78 00FF             NEXT if r7 <> #dlinf       ;reverse not dead, can't use
4FA4 81C4
4FA6 3228 72C2             addm r2,ltbsum             ;clear nay in reverse
4FAA 392B 82C1             subbm r2,ltb+ltbnay(r3)    ;entry usurpable now
                        endif
4FAE 707C 82C0          lda r7,ltb(r4)          ;get contents of usurpable entry
4FB2 3278 72C2          addm r7,ltbsum          ; remove its contribution to cksum
```

```
4FB6 9002           break                       ;have usurpable entry in r4
4FB8 90BA          endrepeat
4FBA 6076          pop r7                       ;fix stack

                ;r1= line source, r2= IMP# of usurpable slot, r4= usurpable slot
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 196
ROUTE.PLR;1     PAGE 15.1       Routing Process

                :   (sp)= Dest for new line
```

```
                ;FNDENT: have useable entry, now expand as req'd

                ;now we have:
                ;   r1 = source for new line
                ;   r2= IMP number of usurpable slot
                ;   r4= usurpable slot in LTB
                ;   (sp)= Dest for new line

4FBC 4E21           if r2 > r1                  ;expand, no free in source lines
4FBE 8C23
4FC0 4078 23C4        call rutsleep             ; but take a rest first
4FC4 4831             lda r3,r1                 ;where to begin changeing indexes
4FC6 48F2             lda r7,#words             ;use this to change NTB indexes
4FC8 7059 85E0        lda r5,ntb+words(r1)      ;free slot will be here- save
4FCC 4B58 03FE        and r5,#ntbidx            ;just index
                      repeat                    ;move pointers to ltb first
4FD0 227B 85E0           addm r7,ntb+words(r3)+ ;move this index
4FD4 4E32             until r3 = r2             ; including last one
4FD6 81FD
                      endrepeat
4FD8 4E45             repeat until r4 = r5      ;when to stop moving
4FDA 9115
                         repeat                 ;to limit strip times
4FDC 507C 82C0              lda r7,ltb(-r4)     ;line data
4FE0 307C 82C2              sta r7,ltb+words(r4)        ;move it
4FE4 707C 85E0              lda r7,lflag(r4)    ;flags
4FE8 757C 85E2              eor r7,lflag+words(r4)
4FEC 4B78 C000              and r7,#lflg
4FF0 357C 85E2              eorm r7,lflag+words(r4)
4FF4 4874                   lda r7,r4           ;check for rest time
4FF6 4975                   sub r7,r5
4FF8 4F78 003E             until r7 .nbit. #3E  ;done 32
4FFC 8AF0
                         endrepeat
4FFE 4078 23C4           call rutsleep          ;time for a break
5002 90EB             endrepeat
                   endif

5004 6026          pop r2                       ;neighbor on line we're entering
5006 487A FF00     lda r7,#dlinf_ H8(r2)        ;initial value for LTB
500A 307C 82C0     sta r7,ltb(r4)               ;put it into it's place
500E 3178 72C2     subm r7,ltbsum               ;and fix cksum as well
5012 4878 3FFF     lda r7,=-1?lflg              ;no line usage yet
5016 337C 85E0     andm r7,lflag(r4)    ;;lflgu

501A 9082       endroutine fndent

                ;now :
                ;   r4 = LTB index for free slot
                ;   r1 = source
                ;   r2 = dest
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 198
ROUTE.PLR;1     PAGE 17         Routing Process

.comnt |
RTINC
Update the routing tree incrementally. Update is passed with
source IMP on line in R1, dest IMP on line in R2, the
change in the line distance in R3, new distance in R5
|

501C 1076       routine rtinc, arg r1-r3/r5, uses r1-r5

501E 48F1       lda r7,=1                       ;count a call to RTINC
5020 3278 5EA8  addm r7,rtical
5024 4078 5352  call findbak                    ;get father of dest
5028 9A04       if fail } byte r1 <> ltb+ltbnay(r4)     ;unreachable or
502A 7E1C 82C1
502E 9137                                       ;this line not in tree
5030 4878 3FFF    lda r7,=pdst                  ;mask for distance
5034 7039 87DC    lda r3,pdist(r1)              ;get dist to dest
5038 4B37         and r3,r7     ;:pdst lflg     ;chuck flags
503A 4A35         add r3,r5                     ;dist via this line
503C 737A 87DC    and r7,pdist(r2)      ;:pdst lflg
5040 4937         sub r3,r7                     ;compare to current dist
5042 9B02         ifnot minus                   ;no better, so done
5044 6006           return                      ;(or if dead line)
                  endif
5046 4E78 3FFF    if r7 = =pdinf                ;wasn't reachable before
504A 8106
504C 3008 72BA      set ppflag                  ;may need post-pass
5050 3008 72BC      set dtflag
5054 900F          else
5056 4878 3FFF      lda r7,=-1?lflg
505A 337C 85E0      andm r7,lflag(r4)   ;:lflgu ;this line out of tree
505E 4851           lda r5,r1                   ;save R1
5060 781C 82C1      ldab r1,ltb+ltbnay(r4)      ;old father
5064 4078 5330      call findnay                ;find the old fwd line
                ;   if fail                     ;topology error
                ;     fail return
                ;   endif
5068 4815           lda r1,r5                   ;restore R1
506A 4878 3FFF      lda r7,=-1?lflg
506E 337C 85E0      andm r7,lflag(r4)   ;:lflgu ;this line now free
                  endif

                ;more RTINC over leaf
```

```
                ;RTINC continued

5072 4078 5330    call findnay                  ;now find new fwd line
                ; if fail                      ;some failure
                ;   fail return
                ; endif
5076 4878 8000    lda r7,=lflgf
507A 347C 85E0    iorm r7,lflag(r4)             ;and mark it
507E 4852         lda r5,r2                     ;save R2
5080 4821         lda r2,r1                     ;and swap with R1
5082 4815         lda r1,r5
5084 4078 5330    call findnay                  ;now for back line
                ; if fail                      ;not there???
                ;   fail return
                ; endif
5088 4812         lda r1,r2                     ;restore R1. R2
508A 4825         lda r2,r5
508C 4878 C000    lda r7,=lflgb
5090 347C 85E0    iorm r7,lflag(r4)             ;mark back line
5094 4078 537E    call copyrut                  ;and propagate route info
                ; if fail
                ;   fail return
                ; endif
5098 4078 23C4    call rutsleep                 ;rest time
                endif
509C 4078 50A6  call retree                     ;compute subtree to check
                ;if success
50A0 4078 51E2    call search                   ;process to-do list
                ; if success
                ;   return                      ;all complete
                ; endif
                ;endif
                ;fail return                   ;some topology failure

50A4 90D0       endroutine rtinc
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 200
ROUTE.PLR;1     PAGE 19          Routing Process

.comnt|
RETREE
Routine to scan the subtree of a node, update distances to all nodes
in the subtree, and put neighbors not in the subtree on the scan list.
R2 is the subtree root node, R3 is the change in distance.
|

50A6 1076       routine retree, arg r2-r3, uses r1-r5

50A8 4812       lda r1,r2                       ;R1 is more convenient
50AA 48F1       lda r7,=1                       ;count a call to us
50AC 3278 5EAA  addm r7,retcal
50B0 4878 FEFF  lda r7,=-1?<lstost_ H8>         ;(do this at top??)
50B4 4848 00FE  lda r4,=nimp*words
                repeat
50B8 137C 86E0    andm r7,lst+noimp0(-r4)       ;clear on-subtree flags
50BC 88FE         until loop
                endrepeat
50BE 4E38 3FFF  if r3 = =pdinf                  ;not reachable this path
50C2 8106
50C4 3008 72BA    set ppflag                    ;flag to run post-pass
50C8 3439 87DC    iorm r3,pdist(r1)
50CC 900D        else
50CE 7079 87DC    lda r7,pdist(r1)              ;new path distance
50D2 4B78 3FFF    and r7,#pdst
50D6 4A73         add r7,r3
50D8 4E78 3FFF    if r7 >= #pdinf               ;too big ???
50DC 9203
50DE 4078 53D0      call spferr         ;SPF must be messed up)}
                ;   fail return
                  endif
50E2 3239 87DC    addm r3,pdist(r1)             ;just update it
                endif
50E6 48F0       lda r7,=0
50E8 3878 5E86  stab r7,head                    ;empty the list
50EC 3878 5E87  stab r7,scn                     ;empty the scan list

                ;more RETREE over leaf
```

```
                ;more RETREE

                repeat                              ;put root onto subtree first
50F0 3879 86DF    stab r7,lst+lstlst(r1)
50F4 3818 5E88    stab r1,subtre
50F8 48F1         lda r7,=lstost
50FA 3C79 86DE    iorbm r7,lst(r1)                  ;put this node on subtree
50FE 4858 03FE    lda r5,=ntbidx                    ;mask for index
5102 7049 85E0    lda r4,ntb+words(r1)              ;end of this node's lines
5106 4B45         and r4,r5                         ;just index
5108 7359 85DE    and r5,ntb(r1)                    ;beg. of this node's lines
510C 4E45         repeat until r4 = r5
510E 9127
5110 507C 85E0      lda r7,lflag(-r4)
5114 8BFC           next ifnot minus     ;;lflg     ;this line free
5116 A2F1           sll r7,1
5118 9BFA           next if minus        ;;lflgf    ;use is forward
511A 782C 82C1      ldab r2,ltb+ltbnay(r4)         ;scan neighbor
511E 7878 5E87      ldab r7,scn
5122 387A 86DF      stab r7,lst+lstlst(r2)
5126 3828 5E87      stab r2,scn
512A 4078 537E      call copyrut                   ;correct its route (?)
                ;   if fail
                ;     fail return                  ;topology bad
                ;   endif
512E 4E38 3FFF      if r3 = =pdinf                 ;already unreachable?
5132 8103
5134 4873             lda r7,r3                    ;then so is dest
5136 900C            else
5138 7079 87DC         lda r7,pdist(r1)            ;correct its path dist
513C 4B78 3FFF         and r7,=pdst
5140 7A7C 82C0         addb r7,ltb+ltbdst(r4)
5144 4E78 3FFF         if r7 >= #pdinf             ;overflow field?
5148 9203
514A 4078 53D0           call spferr               ;must be a bug
                ;        fail return
                       endif
                     endif
514E 757A 87DC       eor r7,pdist(r2)
5152 4B78 3FFF       and r7,=pdst
5156 357A 87DC       eorm r7,pdist(r2)
515A 90D9          endrepeat
515C 7818 5E87     ldab r1,scn                     ;any to scan?
5160 9A0A          until zero
5162 4078 23C4     call rutsleep                   ;take a break
5166 7879 86DF     ldab r7,lst+lstlst(r1)          ;take scan top
516A 3878 5E87     stab r7,scn
516E 7878 5E88     ldab r7,subtre                  ;and put on subtree
5172 90BF        endrepeat

                ;more RETREE over leaf
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 202
ROUTE.PLR;1     PAGE 21         Routing Process

                ;RETREE continued

                repeat
5174 7828 5E88    ldab r2,subtre                ;now look through subtree
5178 9A1D         until zero                    ;it's empty
517A 787A 86DF    ldab r7,lst+lstlst(r2)        ;pop next
517E 3878 5E88    stab r7,subtre
5182 4858 03FE    lda r5,=ntbidx
5186 704A 85E0    lda r4,ntb+words(r2)          ;for all neighbors
518A 4B45         and r4,r5
518C 735A 85DE    and r5,ntb(r2)
                  repeat
5190 4E45           until r4 = r5
5192 910D
5194 507C 85E0      lda r7,lflag(-r4)           ;status of this line
5198 9BFC           next if minus       ;;lflgu ;fwd or back
519A 781C 82C1      ldab r1,ltb+ltbnay(r4)      ;neighbor here
519E 9AF9           next if zero                ;ignore usurpables
51A0 7879 86DE      ldab r7,lst(r1)             ;already done?
51A4 99F6           next if odd ;;lstost        ;yes
51A6 4078 51B4      call dorouter
                ;   if fail
                ;     fail return
                ;   endif
51AA 90F3         endrepeat
51AC 4078 23C4    call rutsleep                 ;rest now
51B0 90E2       endrepeat

51B2 6006       endroutine retree
```

```
.comnt |
DOROUTER
Subroutine to call ROUTER on behalf of RETREE,
for a line that is not in the routing tree already.
Called with R1/line dest, R2/line source, R3/change in delay
R4/line pointing out of subtree (R2 -> R1)
Computes proper line (forward or back) to use,
and calls ROUTER on it.
|

51B4 1076       routine dorouter,arg r1-r3,local r4,uses r1
51B6 1046

51B8 4EB0         if r3 > =0                    ;line got worse
51BA 8C04
51BC 4078 5330      call findnay                ;get line into subtree
                  ;   if fail                   ;topology bad
                  ;     fail return
                  ;   endif
51C0 900E           else                        ;line got better
51C2 7079 87DC      lda r7,pdist(r1)
51C6 701A 87DC      lda r1,pdist(r2)            ;current dist this node
51CA 4B18 3FFF      and r1,=pdst
51CE 4B78 3FFF      and r7,=pdst                ;and dist via subtree
51D2 4E71           if r7 <= r1                 ;got no better route
51D4 9C03
51D6 6046             return                    ;so we're done
51D8 6006
                    endif
51DA 4812           lda r1,r2                   ;line source
                  endif
51DC 4078 524A    call router                   ;R4 has line already
                  ; if fail                     ;topology bad
                  ;   fail return
                  ; endif

51E0 90FB       endroutine dorouter
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 204
ROUTE.PLR;1     PAGE 23         Routing Process

.comnt |
SEARCH
Routine to process the to-do list. List is already set up in HEAD.
Mungs R1-4. Calls ROUTER as needed.
|

51E2 1076       routine search, uses r1-r4

                repeat                          ;for all list items
51E4 7828 5E86    ldab r2,head
51E8 9A30         until zero                    ;empty list
51EA 3828 72BC    stab r2,dtflag                ;* tree may change now *
51EE 787A 86DF    ldab r7,lst+lstlst(r2)
51F2 3878 5E86    stab r7,head                  ;dequeue one
51F6 4878 FFFD    lda r7,=-1?lstol
51FA 3B7A 86DE    andbm r7,lst(r2)
51FE 4078 5352    call findbak                  ;get father node, if any
5202 9A0F         if success                    ;there is one
5204 781C 82C1      ldab r1,ltb+ltbnay(r4)      ;get its dest IMP
5208 4078 5330      call findnay                ;now get forward line
                ;   if fail                     ;bad topology
                ;     fail return
                ;   endif
520C 707C 85E0      lda r7,lflag(r4)
5210 4B78 3FFF      and r7,#-1?lflg
5214 4C78 8000      ior r7,#lflgf               ;add this line to tree
5218 307C 85E0      sta r7,lflag(r4)            ;as forward line
521C 4078 537E      call copyrut                ;dst gets src's route
                ;   if fail                     ;some failure
                ;     fail return
                ;   endif
                  endif
5220 4812           lda r1,r2                   ;looking at son now
5222 4828 03FE    lda r2,=ntbidx
5226 7049 85E0    lda r4,ntb+words(r1)          ;now do son's neighbors
522A 4B42         and r4,r2
522C 7329 85DE    and r2,ntb(r1)
5230 4E42         repeat until r4 = r2
5232 9108
5234 507C 85E0      lda r7,lflag(-r4)           ;this line's use?
5238 A2F1           sll r7,1
523A 9BFB           next if minus       ;;lflgb ;it's a back-link
523C 4078 524A      call router
                ;   next ifnot fail             ;no failure
                ;   fail return
5240 90F8         endrepeat
5242 4078 23C4    call rutsleep                 ;whew} let's rest
5246 90CF       endrepeat

5248 6006       endroutine search
```

```
.comnt |
ROUTER
Subroutine to (re)process a particular node. If things got better for
that node, remove it from the tree and add it to the process list.
Remove duplicates on that list, if any. Expects R4 to have the path
and R1 the node in question.
|

524A 1076       routine router, arg r1/r4, local r1-r2, local r4-r5
524C 1016
524E 1026
5250 1046
5252 1056

5254 4821       lda r2,r1                       ;we want different regs
5256 787C 82C0  ldab r7,ltb+ltbdst(r4)          ;delay this line
525A 4E78 00FF  if r7 <> =dlinf                 ;it's alive
525E 9164
5260 701A 87DC    lda r1,pdist(r2)              ;path dist to src
5264 4B18 3FFF    and r1,=pdst
5268 4E18 3FFF    if r1 <> =pdinf               ;we have a path here
526C 915D
526E 4A71           add r7,r1                   ;dist via this line
5270 4E78 3FFF      if r7 > =pdinf              ;more than infinite .
5274 8C03
5276 4078 53D0        call spferr               ;how can this happen?
                ;     fail return
                    endif
527A 781C 82C1      ldab r1,ltb+ltbnay(r4)      ;dst node
527E 7059 87DC      lda r5,pdist(r1)            ;its present distance
5282 4B58 3FFF      and r5,=pdst
5286 4E75           if r7 < r5                  ;new path is better
5288 824D
528A 7579 87DC        eor r7,pdist(r1)          ;store new distance
528E 4B78 3FFF        and r7,=pdst
5292 3579 87DC        eorm r7,pdist(r1)
5296 4078 5330        call findnay              ;get back line
                ;     if fail
                ;       fail return
                ;     endif

                ;more ROUTER over leaf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 206
ROUTE.PLR;1     PAGE 25         Routing Process

                ;ROUTER continued

529A 4854               lda r5,r4               ;and remember for later
529C 4821               lda r2,r1               ;get present back-link
529E 4078 5352          call findbak
52A2 9A0E               if success
52A4 4878 3FFF            lda r7,=-1?lflg       ;take this line out of tree
52A8 337C 85E0            andm r7,lflag(r4)     ;;lflgu
52AC 781C 82C1            ldab r1,ltb+ltbnay(r4) ;and find forward line
52B0 4078 5330            call findnay
                ;         if fail               ;bad topology
                ;           fail return
                ;         endif
52B4 4812                 lda r1,r2             ;(restore R1 for later)
52B6 4878 3FFF            lda r7,=-1?lflg       ;and mark it unused
52BA 337C 85E0            andm r7,lflag(r4)     ;;lflgu
                        endif
52BE 4828 D7A7          lda r2,=head-<lst+lstlst> ;offset if empty list
52C2 1056               save r5                 ;remember back link
52C4 7059 87DC          lda r5,pdist(r1)
52C8 4B58 3FFF          and r5,=pdst            ;dist to this one
52CC 7878 5E86          ldab r7,head
52D0 9A0B               ifnot zero              ;any in list
                          repeat
52D2 704F 87DC              lda r4,pdist(r7)    ;check down list
52D6 4B48 3FFF              and r4,=pdst
52DA 4E54                   until r5 < r4       ;insert in order
52DC 9205
52DE 4827                   lda r2,r7           ;chain down one
52E0 787F 86DF              ldab r7,lst+lstlst(r7)
52E4 8AF7                   until zero
                          endrepeat
                        endif

                ;more ROUTER over leaf
```

```
                ;ROUTER continued

52E6 7859 86DF          ldab r5,lst+lstlst(r1)    ;remember link for later
52EA 3879 86DF          stab r7,lst+lstlst(r1)
52EE 381A 86DF          stab r1,lst+lstlst(r2)    ;;head  ;insert this node in list
52F2 7829 86DE          ldab r2,lst(r1)           ;its flags
52F6 4FA2               if r2 .bit. =lstol        ;this node on list already
52F8 9A0D
52FA 4871                 lda r7,r1
52FC 7E1F 86DF            repeat until byte r1 = lst+lstlst(r7)   ;find other
5300 9107
5302 787F 86DF              ldab r7,lst+lstlst(r7)        ;occurence of it
5306 8AFB                   next ifnot zero       ;chaining to a good one
5308 4078 53D0              call spferr   ;some SPF trouble
                ;           restore r4            ;fix up stack
                ;           fail return
530C 90F8                 endrepeat
530E 385F 86DF            stab r5,lst+lstlst(r7)  ;restore old list ptr
                        endif
5312 4CA2               ior r2,=lstol
5314 3829 86DE          stab r2,lst(r1)           ;this node on list now
5318 6046               restore r4                ;fix up its back link
531A 4878 C000          lda r7,=lflgb
531E 347C 85E0          iorm r7,lflag(r4) ;;lflg  ;type of use
                      endif
5322 4078 23C4        call rutsleep               ;time for a rest
                    endif
                  endif

5326 6056         endroutine router
5328 6046
532A 6026
532C 6016
532E 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 208
ROUTE.PLR;1     PAGE 27         Routing Process

                ;FINDNAY
                ;Find line from node in R1 to node in R2
                ;Returns index for LTB, LFLAG in R4
                ;Call SPFERR if can't find the line

5330 1076       routine findnay, arg r1-r2, result r4

5332 4878 03FE    lda r7,=ntbidx
5336 7049 85E0    lda r4,ntb+words(r1)          ;for all node 1's lines
533A 4B47         and r4,r7
533C 7379 85DE    and r7,ntb(r1)
                  repeat
5340 4E47           if r4 <= r7                 ;node 2 not there}
5342 9C03
5344 4078 53D0        call spferr               ;stop SPF}}
                ;     fail return               ;and tell our caller
                    endif
5348 49C2           sub r4,=words               ;next neighbor
534A 7E2C 82C1      until byte r2 = ltb+ltbnay(r4)      ;got node 2
534E 81F9
                  endrepeat

5350 6006       endroutine findnay

                ;FINDBAK
                ;Find the back link from this node, if any
                ;Returns line table index in R4, arg node is in R2
                ;Fail if no back link

5352 1076       routine findbak, arg r2, result r4, local r1
5354 1016

5356 4878 03FE    lda r7,=ntbidx
535A 704A 85E0    lda r4,ntb+words(r2)          ;for all lines
535E 4B47         and r4,r7
5360 737A 85DE    and r7,ntb(r2)
                  repeat
5364 4E47           if r4 <= r7                 ;no back link
5366 9C05
5368 6016             fail return
536A 6076
536C 4FF0
536E 4007
                    endif
5370 49C2           sub r4,=words               ;next link
5372 701C 85E0      lda r1,lflag(r4)            ;its use
5376 A291           sll r1,1
5378 8BF6           until minus          ;;lflgb ;back link we want
                  endrepeat

537A 6016       endroutine findbak
537C 6006
```

```
                ;COPYRUT
                ;Copy route of node in R1 to node in R2
                ;Handles special case that R1 is us.
                ;Call SPFERR if can't find our neighbor

537E 1076       routine copyrut,arg r1/r2,local r3
5380 1036

5382 7039 72FC    lda r3,spfrut(r1)             ;copy father's route
5386 4BBF         and r3,=routef
5388 8A11         if zero                       ;I'm the source
538A 48B0           lda r3,=0                   ;really wants modem number
538C A6A1           srl r2,1                    ;un-double IMP
                    repeat
538E 607B 635E        lda r7,m2pblk(r3)+
5392 9B04             until nminus & byte r2 = lnei(r7) ;line to this node
5394 7E2F 0004
5398 9107
539A 4E38 0010        next if r3 < =nmd*words   ;check more of table
539E 92F8
53A0 4078 53D0        call spferr               ;trouble: stop system
                ;     fail return
53A4 90F5           endrepeat
53A6 A6B1           srl r3,1                    ;get actual modem
53A8 A2A1           sll r2,1
                  endif
53AA 753A 72FC    eor r3,spfrut(r2)
53AE 4BBF         and r3,=routef
53B0 9A0E         ifnot zero                    ;route changing
53B2 7078 B2C6      lock spfrtl
53B6 9AFE
53B8 753A 72FC      eor r3,spfrut(r2)           ;copy src route to dst
53BC 713A 72FC      sub r3,spfrut(r2)
53C0 323A 72FC      addm r3,spfrut(r2)
53C4 3138 72C4      subm r3,spfsum              ;and adjust checksum
53C8 3008 B2C6      unlock spfrtl
                  endif

53CC 6036       endroutine copyrut
53CE 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 210
ROUTE.PLR;1     PAGE 29          Routing Process

                ;SPFERR
                ;An error in the SPF database was encountered.
                ;Never returns (goes to WS or RELOAD)

                ;routine spferr
                spferr:

53D0 1076         push r7                       ;save our caller
53D2 E500         Trap 2400,<;SPF error forced restart - page 208>
53D4 4078 23C4    call rutsleep                 ;get everyone here
53D8 6046         pop r4                        ;get back our caller
53DA 3008 B2C8    unlock rutlok
53DE 4878 0044    set @pid = #rutpi             ;get everybody to do this
53E2 30F8 00AC
53E6 3048 6518    set nspc = r4                 ;remember who told us to crash
53EA 7078 62E2    if mine = spfcrash            ;should I dump?
53EE 7678 4C6C
53F2 8103
53F4 4008 0A08      jmp ws
                  endif
53F8 4008 0A08    jmp ws                        ;just restart

                ;endroutine spferr

.comnt |
RTINIT
Initialize the shortest-path tree. Sets all path distances to
infinite, and empties the tree, clears all topology info.
R5 enters with my IMP number * 2.
|

53FC 1076       routine rtinit, arg r5, uses r1-r3

53FE 4828 00FE  lda r2,=nimp*words            ;how many IMPs
5402 48F0       lda r7,=0                     ;for initialization
5404 4818 3FFF  lda r1,=pdinf                 ;to make distance infinite
5408 4838 8000  lda r3,=spfded                ;and nodes unreachable
                repeat
540C 101A 87DE    sta r1,pdist+noimp0(-r2)    ;;lflg  ;everyone's unreachable
5410 303A 72FE    sta r3,spfrut+noimp0(r2)    ;no route
5414 307A 86E0    sta r7,lst+noimp0(r2) ;;lflg ;no lists
5418 307A 85E0    sta r7,ntb+noimp0(r2) ;;lflg ;no lines at first
541C 88F8         until loop                  ;for all IMPs
                endrepeat
541E 3078 86DE  sta r7,ntbendi  ;;lflg        ;clear extra NTB entry
5422 307D 87DC  sta r7,pdist(r5)        ;;lflg ;set root distance to 0
5426 307D 72FC  sta r7,spfrut(r5)             ;and mark us reachable
542A 3078 72C2  sta r7,ltbsum                 ;empty table sum is 0
542E 48F0       set spfsum = #-<nimp-1>*spfded ;initial SPFRUT checksum
5430 3078 72C4

5434 6006       endroutine rtinit
```

```
                ;RUPCKSUM
                ;Checksum routing data
                ;Validates checksums on important routing databases:
                ;  LTB, since this data is what others send us
                ;  SPFRUT, since routing decisions are based upon it

5436 1076       routine rupcksum,uses r3-r5

5438 4858 72FE    set r5 = #spfrut+noimp0       ;where to start out
543C 48B0         clear r3                      ;initial sum
543E 7078 B2C6    lock spfrtl                   ;shut off modifications
5442 9AFE
     0001         .rept nimp/ D72
                    call subchn-< D72*words>    ;do a 72-IMP chunk
5444 4078 17F8    .endr
5448 4078 181A    call subchn-<<nimp-< D72*<nimp/ D72>>>*words>
544C 7078 72C4    set r7 = spfsum               ;sum to compare
5450 3008 B2C6    unlock spfrtl                 ;done locked stuff
5454 4E37         if r3 <> r7                   ;checksum broken??
5456 9103
5458 4078 53D0      call spferr         ;oops - restart
                  endif
545C 4078 23C4    call rutsleep                 ;whew}
5460 48C0         clear r3,r4                   ;init for another chksum
5462 4834
5464 7058 86DE    lda r5,ntbendi                      ;how long ntb is
5468 4B58 03FE    and r5,=ntbidx                ;don't want all these bits
546C 4945         sub r4,r5                     ;negatize it
546E 4858 82C0    set r5 = #ltb                 ;checksumming LTB
5472 4E48 FF70    repeat until r4 >= #- D72*words
5476 8208
5478 4078 17F8      call subchn-< D72*words>    ;do another 72
547C 4A48 0090      add r4,= D72*words          ;how much table we did this time
5480 4078 23C4      call rutsleep               ;then rest
5484 90F7         endrepeat
5486 407C 1888    call subchn(r4)               ;do the last bit
548A 7078 72C2    set r7 = ltbsum               ;this should be sum
548E 4E37         if r3 <> r7                   ;oops}
5490 9103
5492 4078 53D0      call spferr
                  endif

5496 6006       endroutine rupcksum
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 212
ROUTE.PLR;1     PAGE 31         Routing Process

     0000       .if nz $maprt-$mappk     ;need following from both pages
                  page LCode
                .endc

.comnt |
RUPENQ - queue routing update
Assumes m2 set for chain pointers.
Queue on rupq, poke routing if rup4us set.
Set fldior so timeout knows who to poke.
Set retransmit timers on all modems for which we are enqueuing

r1 - (doubled) IMP number of source
r2 - chain pointer
r3 - mask bits to be put in chan
r4 - source modem (if any)
|

5498 1076       routine RUPENQ, arg r1-r4/m2, local r5
549A 1056

549C 303A 0350    sta r3,chan-chain(r2)         ;set mask bits
54A0 8A02         if zero
54A2 E50A           Trap 2412,<:Queuing packet for no one - page 210>
                  endif
54A4 48F1         lda r7,=1
54A6 3072         sta r7,(r2)                   ;set end of queue indicator
54A8 7078 A4A4    lock lockro                   ;rupq protected by lockro
54AC 9AFE
54AE 30A8 72B0    sta r2,@erupq                 ;queue on rupq
54B2 3028 72B0    sta r2,erupq
54B6 4F38 8000    if r3 .bit. =rup4us           ;are we to process it?
54BA 9A3C
54BC 7078 5EB2      lda r7,rupqct               ;number of packets 4us on rupq
54C0 7678 5E94      if r7 > rqlmax              ;is this the max so far?
54C4 8C1C
54C6 3078 5E94        sta r7,rqlmax             ;yes, record it
54CA 1016               SAVE R1-R7
54CC 1026
54CE 1036
54D0 1046
54D2 1056
54D4 1066
54D6 1076
54D8 7018 5EB2          LDA R1,RUPQCT           ;CURRENT QUEUE
54DC 7028 5E98          LDA R2,RTRCNT           ;RETRANS PER LINE
54E0 7038 5E96          LDA R3,RUPCNT           ;# UPDS GENERATED
54E4 7048 5E94          LDA R4,RQLMAX           ;MAX QUEUE REACHED
54E8 7058 5E8E          LDA R5,RPKREC           ;UPDS PROCESSED
54EC E557             Trap 2527,<:Queue count check - page 210>
54EE 6076               RESTORE R1-R7
54F0 6066
54F2 6056
54F4 6046
54F6 6036
54F8 6026
54FA 6016
```

```
54FC 3278 5E92      addm r7,rqlsum              ;sum queue lengths encountered
5500 48D1           lda r5,=1                   ;use r5
5502 3258 5E8E      addm r5,rpkrec              ;count update rec'd
5506 7258 5EB2      add r5,rupqct               ;increase queue count
550A 3058 5EB2      sta r5,rupqct               ;crash into loader if RUPQ >= 20
550E 4E58 0016      if r5 >= # D22              ;too large ))
5512 9204
5514 E555              Trap 2525,<;Queue count too high - page 210>
5516 4048 0A1C         jsb r4,wst               ;kill it all, quickly
                    endif
551A 1016           push r1                     ;need this again later
551C 4078 1512      call unpack                 ;need to look inside
5520 7871           ldab r7,(r1)                ;;neth
5522 4BFF           and r7,=rupnn_- H8          ;number of lines in update
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 213
ROUTE.PLR;1     PAGE 31.1       Routing Process

5524 3278 5E90      addm r7,rlnrec              ;count number of entries received
5528 6016           pop r1                      ;restore doubled IMP
552A 7078 00D2      setmap m2,mapv2             ;switch back to v2 page
552E 3078 FC04
                  endif
5532 4821           lda r2,r1                   ;free r1 for use here
5534 7018 72AC      lda r1,fldior               ;old ones doing already
5538 3438 72AC      iorm r3,fldior              ;add new ones yet to do
553C 3008 A4A4      unlock lockro
                  ;(OVER)
```

```
                ;Now poke all who matter

5540 4833         lda r3,r3                     ;bits we just set
5542 8B07         if minus      ;;rup4us
5544 4878 0044      set @pid = =rutpi           ;poke routing
5548 30F8 00AC
554C 4D38 8000      eor r3,=rup4us              ;and clear its bit
                  endif
5550 9A18         ifnot zero
5552 1046           push r4                     ;source modem at stack top
5554 7058 6324      lda r5,modems               ;some modems to poke, too
                    repeat
5558 573D 0EB6        if r3 .bit. bittab(-r5)
555C 9A0F
555E 704D 635E          lda r4,m2pblk(r5)       ;a modem to get
5562 9B0C               ifnot minus             ;modem still good?
5564 7646                 if r4 <> (sp)         ;not source modem either
5566 9105
5568 1026                   save r2
556A 4078 5958              call rton           ;turn on this imp's timer
556E 6026                   restore r2
                          endif
5570 771D 0EB6            if r1 .nbit. bittab(r5)
5574 8A03
5576 4078 14E8              call pokem          ;set i2mpok and poke modem
                          endif
                        endif
                      endif
557A 4ED0             until r5 = =0
557C 81EE
                    endrepeat
557E 6046           pop r4                      ;restore source modem
                  endif

5580 6056       endroutine rupenq
5582 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 215
ROUTE.PLR;1     PAGE 33         Routing Process


.comnt |
RUPDEQ - get a buffer from the queue of routing updates.
Since these buffers are used by more than one process,
a buffer is not actually dequeued, instead
the address of its chain word is returned.

r3 - bit for calling process.  This bit must be on in chan
word of buffer for it to be selected.

Returns chain address in r2, buffer address in r1. (does unpack)
|

5584 1076       Routine RUPDEQ, arg r3, result r1-r2, result m2, uses r5

5586 7738 72AC  if r3 .nbit. fldior             ;this caller on queue?
558A 8A04
558C 6076         fail return                   ;nope
558E 4FF0
5590 4007       endif
5592 7078 00D2  setmap m2,mapv2                 ;get buffer vars
5596 3078 FC04
559A 4858 72AE  lda r5,=srupq                   ;scan the routing update queue
559E 7078 A4A4  lock lockro                     ;must search queue locked
55A2 9AFE
                repeat
55A4 7025         lda r2,(r5)                   ;chain to next buffer
55A6 8908         if odd
55A8 4D38 FFFF      eor r3,=-1
55AC 3338 72AC      andm r3,fldior              ;clear caller's bit
55B0 3008 A4A4      unlock lockro
55B4 90EC           fail return                 ;nothing on queue for caller
                  endif
55B6 4078 2414    call rupwhc                   ;check for queue broken
55BA 8A02         if fail
55BC 90E8           fail return                 ;queue broken, return nothing
                  endif
55BE 773A 0350    until r3 .bit. chan-chain(r2)
55C2 8A03
55C4 4852         lda r5,r2
55C6 90EF       endrepeat
55C8 3008 A4A4  unlock lockro
55CC 4078 1512  call unpack                     ;get buffer address

55D0 6006       endroutine rupdeq
```

```
                        page LCode

.comnt |
RUPWHC - check ownership bits on rup queue
fix queue if discovered to be broken

r2 - current chain pointer
r5 - previous chain pointer
|

2414 1076       Routine RUPWHC, arg r2, arg r5

2416 707A 06A0  if where-chain(r2) .nbit. =whrut
241A 4F78 0080
241E 8A0B
2420 E503         Trap 2403,<;routing broken queue - page 214>
2422 48F1         lda r7,=1                     ;terminate queue at last good one
2424 3075         sta r7,(r5)
2426 3058 72B0    sta r5,erupq
242A 3008 A4A4    unlock lockro                 ;release routing out lock
242E 6076         fail return
2430 4FF0
2432 4007
                endif

2434 6006       endroutine rupwhc
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 217
ROUTE.PLR;1     PAGE 35         Routing Process

     0000       .if nz $maprt-$mappk     ;need following from both pages
                  page LCode
                 .iff ;;nz $maprt-$mappk
FC00 0400         page RutCode
                .endc

.comnt |
RUPFLS - routing update flush routine

Turn off this process's bit and if no others on, do
a real flush of the buffer.
Switches to mapv2.

r3 - bit for calling process. (not preserved)
r2 - address of this buffer's chain word
|

55D2 1076       Routine RUPFLS, arg r2-r3, uses r5, uses r3, result m2

55D4 7078 00D2  setmap m2,mapv2                 ;switch to chain words
55D8 3078 FC04
55DC 7078 A4A4  lock lockro                     ;lock the update queue
55E0 9AFE
55E2 707A 06A0  lda r7,where-chain(r2)          ;check if owned by routing
55E6 4F78 0080  if r7 .nbit. =whrut
55EA 8A05
55EC E504         Trap 2404,<:Buffer no longer owned by routing - page 215>
55EE 3008 A4A4    unlock lockro
55F2 6006         return                        ;it was probably timed out
                endif
55F4 707A 0350  lda r7,chan-chain(r2)           ;get the process mask
55F8 4D73       eor r7,r3                       ;turn off caller's bit
55FA 4F73       if r7 .bit. r3                  ;was it off?
55FC 9A03
55FE E505         Trap 2405,<:Caller's bit not on - page 215>
5600 4D73         eor r7,r3                     ;leave it off
                endif
5602 4833       lda r3,r3                       ;buffer for SPF itself?
5604 8B04       if minus        ::rup4us
5606 48B1         lda r3,=1                     ;decrement queue length count
5608 3138 5EB2    subm r3,rupqct
                endif

                ;OVER
```

```
                ;more RUPFLS

560C 307A 0350  sta r7,chan-chain(r2)           ;re-write mask word
5610 8A1A       if zero
5612 4858 72AE    lda r5,=srupq                 ;this was last user
5616 4832         lda r3,r2                     ;save buffer to match on
                  repeat                        ;find it on update queue
5618 7025           lda r2,(r5)                 ;chain to next buffer
561A 8905           if odd
561C E506             Trap 2406,<;Rupq buffer missing - page 216>
561E 3008 A4A4        unlock lockro
5622 90E8             return                    ;????
                    endif
5624 4078 2414      call rupwhc                 ;check if still on rupq
5628 8A02           if fail
562A 90E4             return                    ;queue has been broken
                    endif
562C 4E23           until r2 = r3
562E 9103
5630 4852           lda r5,r2
5632 90F3         endrepeat
5634 7072         lda r7,(r2)                   ;remove it from the chain
5636 3075         sta r7,(r5)
5638 8903         if odd                        ;last one on the queue?
563A 3058 72B0      sta r5,erupq                ;yes, update the end pointer
                  endif
563E 4078 1332    call flush,whrut              ;now do a real flush
5642 0080
                endif
5644 3008 A4A4  unlock lockro

5648 90D5       endroutine rupfls
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 219
ROUTE.PLR;1     PAGE 37         Routing Process


FC00 0400               page RutCode

.comnt |
RUPGEN - build a routing update
Called by routing process if RUPSND set.
Uses average line delays calculated by delay package.
|

564A 1076       Routine RUPGEN, uses r1-r5

564C 48F8       setb thresh = =thrini           ;reset thresh when sending update
564E 3878 72A9
5652 7078 72A6  if iniflg                       ;don't send during init
5656 9A58
5658 4078 13C6    call freget,nrut+whrut        ;try to get a buffer for update
565C 0088
565E 8A07         if fail
5660 48F1           lda r7,=1
5662 3278 5EAE      addm r7,rpgnbf              ;count times we couldn't get one
5666 3008 72B4      set rretry                  ;none, turn flag back on later
566A 904E          else
566C 1026           save r2                     ;save the chain pointer
566E 48D0           lda r5,=0                   ;init the checksum register
5670 4821           lda r2,r1                   ;copy the buffer pointer
5672 1016           save r1                     ;save the buffer pointer
5674 7018 651A      lda r1,nsrutd               ;check nice-stop flag
5678 9A03           ifnot zero
567A 4818 00FF        lda r1,=dlinf             ;report all lines dead
                    endif
567E 7038 6324      lda r3,modems               ;loop through all modem blocks
                    repeat
5682 504B 635E        lda r4,m2pblk(-r3)        ;address of a modem block
5686 9B0F             ifnot minus
5688 787C 0004          ldab r7,lnei(r4)        ;don't report unused
568C 9A0C               if nzero & r7 <> mine   ;or looped lines
568E 7678 62E2
5692 9109
5694 A2F8                 sll r7, H8            ;;rupnei
5696 7C7C 009B            iorb r7,delbas(r4)    ;;rupdel
569A 4C71                 ior r7,r1     ;;dlinf ;set dead if nice-stopping
569C A7F8                 rrl r7, H8            ;switch bytes
569E 207A 0008            sta r7,seqh(r2)+      ;put in line entry
56A2 4957                 sub r5,r7             ;include in checksum
                        endif
                      endif
56A4 4EB0             until r3 = =0
56A6 81EE
                    endrepeat


                  ;(OVER)
```

```
                ;RUPGEN continued

56A8 6016           restore r1
56AA 4921           sub r2,r1                   ;number of neighbor entries
56AC A6A1           srl r2,1                    ;undouble
56AE 7038 62E2      lda r3,mine
56B2 A2B1           sll r3,1                    ;doubled imp number
56B4 7078 B2C6      lock spfrtl
56B8 9AFE
56BA 707B 72FC      lda r7,spfrut(r3)
56BE 4A78 0100      add r7,=spfsn1              ;increment the serial number
56C2 757B 72FC      eor r7,spfrut(r3)
56C6 4B78 3F00      and r7,=spfsno              ;new serial number
56CA 757B 72FC      eor r7,spfrut(r3)           ;insert back into spfrut
56CE 4C78 0070      ior r7,=spfage              ;set max age
56D2 4847           lda r4,r7                   ;need this later
56D4 717B 72FC      sub r7,spfrut(r3)
56D8 327B 72FC      addm r7,spfrut(r3)          ;update table now
56DC 3178 72C4      subm r7,spfsum              ;and its checksum
56E0 3008 B2C6      unlock spfrtl
56E4 4B48 3F00      and r4,=spfsno
56E8 A6B1           srl r3,1                    ;undouble the imp number
56EA 4C34           ior r3,r4                   ;combine with the serial number
56EC 3039 0006      sta r3,srch(r1)             ;;rupsrc, rupsno
56F0 4953           sub r5,r3                   ;include in checksum
56F2 484A 0070      lda r4,#spfage(r2)          ;combine age and number of entries
56F6 7038 72AA      lda r3,rupmsk               ;this should be processed by all
56FA 6026           restore r2
56FC 48F1           lda r7,=1
56FE 3278 5E96      addm r7,rupcnt              ;count an update generated
5702 4078 5708      call rupbld                 ;share code with rtrgen
                  endif
                endif

5706 6006       endroutine rupgen
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 221
ROUTE.PLR;1     PAGE 39         Routing Process


.comnt |
RUPBLD - finish building an update packet and queue it

r1 - buffer address
r2 - chain address
r3 - mask for which processes should look at this update
r4 - left byte of neth (in right byte of r4) age+number of neighbors
r5 - checksum on update built so far
|

5708 1076       Routine RUPBLD, arg r1-r5, uses r1-r5

570A 4874       lda r7,r4                       ;save neth word
570C A2C8       sll r4, H8                      ;move into left byte
570E 3041       sta r4,(r1)             ;;neth
5710 4954       sub r5,r4                       ;include in checksum
5712 4848 A300  lda r4,=ruttyp+combit+ruptyp
5716 3049 0002  sta r4,typh(r1)
571A 4954       sub r5,r4
571C 4BFF       and r7,=rupnn_- H8              ;get number of line entries
571E 4AF4       add r7,=seqh_-1                 ;add in words in header
5720 4A57       add r5,r7                       ;include length in checksum
5722 3059 0004  sta r5,chkh(r1)                 ;packet now complete .
5726 A2F1       sll r7,1                        ;number of bytes in packet
5728 49F2       sub r7,=hrdoff
572A 3079 0090  sta r7,bufe(r1)
572E 48C0       clear r4                        ;have RUPENQ set all timers
5730 7819 0007  ldab r1,srch+1(r1)              ;get source IMP
5734 A291       sll r1,1                        ;doubled
5736 7078 00D2  setmap m2,mapv2
573A 3078 FC04
573E 4078 5498  call rupenq                     ;share code with m2i
5742 4078 23C4  call rutsleep                   ;need a rest now

5746 6006       endroutine rupbld
```

```
.comnt |
RTRTO - tick routing update retransmission timers
If a timer expires, call rtrgen to build a retransmission.
Called every 25ms by routing process.
|

5748 1076       Routine RTRTO, uses r1-r5

574A 7058 6324  lda r5,modems                   ;loop through all modems
                repeat
574E 504D 635E    lda r4,m2pblk(-r5)            ;modem block address
5752 9B44         ifnot minus
5754 787C 0002      if byte phflag(r4) .nbit. =inhrst
5758 4FF2
575A 8A40
                                                ;don't tick line in reset
575C 48A1           lda r2,=1
575E 707C 40B8      lock rtimrl(r4)             ;lock timers and clock
5762 9AFE
5764 392C 00BA      submb r2,rtrclk(r4)         ;decrement clock
5768 8A35           if zero
576A 787C 00BB        ldab r7,rtrtic(r4)        ;clock has fired
576E 387C 00BA        stab r7,rtrclk(r4)        ;reset it

                ;(OVER)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 223
ROUTE.PLR;1     PAGE 41         Routing Process


                ; RTRTO continued

5772 483C 0020          lda r3,=<<<nimp-1>_-3>+1>*words(r4)     ;address+1 of last timer
                        repeat
5776 501B 00BC            lda r1,rtimrs(-r3)    ;get a word of timers
577A 9A2A                 ifnot zero            ;skip if all zero
577C 4821                   lda r2,r1           ;copy timer word
577E A6A1                   srl r2,1            ;two bits per timer
5780 4C21                   ior r2,r1           ;or them together
5782 4B28 5555              and r2,# H5555      ;now ones for active timers
5786 4912                   sub r1,r2           ;decrement active timers
5788 301B 00BC              sta r1,rtimrs(r3)   ;update timers
578C A691                   srl r1,1            ;now find expired timers
578E 741B 00BC              ior r1,rtimrs(r3)   ;ones for still active timers
5792 4D18 FFFF              eor r1,=-1          ;ones for all off timers
5796 4B12                   and r1,r2           ;remove ones that were off before
5798 9A1B                   ifnot zero          ;any new ones go off?
579A 48A0                     lda r2,=0         ;yes, generate retransmission(s)
579C 4811                     lda r1,r1 ;:odd
                              repeat
579E 8915                       if odd          ;loop for timer(s) that expired
57A0 300C 40B8                    unlock rtimrl(r4)     ;release lock for now
57A4 4078 23C4                    call rutsleep ;sleep before retransmitting
57A8 4078 57E0                    call rtrgen   ;generate a retransmission
57AC 9A05                         if success    ;all's well
57AE 707C 40B8                      lock rtimrl(r4)     ;get lock again
57B2 9AFE
57B4 900A                         else
57B6 707C 40B8                      lock rtimrl(r4)     ;get lock again
57BA 9AFE
57BC A2A2                           sll r2,2    ;double timer index
57BE 707A 0EB6                      lda r7,bittab(r2)   ;get one bit for this timer
57C2 347B 00BC                      iorm r7,rtimrs(r3)  ;turn timer back on
57C6 A6A2                           srl r2,2    ;and hope for next tick
                                  endif
                                endif
57C8 4AA1                       add r2,=1       ;count which timer in this word
57CA A692                       srl r1,2 ;:odd  ;shift next timer to right
57CC 8AE9                       until zero      ;stop when all timers done
                              endrepeat
                            endif
                          endif
57CE 4E34                 until r3 = r4
57D0 81D3
                        endrepeat
                      endif
57D2 300C 40B8        unlock rtimrl(r4)         ;unlock the timers
57D6 4078 23C4        call rutsleep             ;sleep between modems
                    endif
                  endif
57DA 4ED0         until r5 = =0
57DC 81B9
                endrepeat

57DE 6006       endroutine rtrto
```

```
.comnt |
RTRGEN - generate a retransmission packet
r2 - index into this rtimers word (group of 8 IMPs)
r3 - index into rtimers
r4 - modem block address
m2 - mapv2

Fail returns if unprocessed update on rupq for this imp
or unable to get a buffer.
Succeed returns if update queued or if it should be ignored.

N.B. Depends on r5 being saved last
|

57E0 1076       Routine RTRGEN, arg r2-r4/m2, local r1-r5
57E2 1016
57E4 1026
57E6 1036
57E8 1046
57EA 1056

57EC 4934       sub r3,r4                       ;(IMP number-1)/8 (doubled)
57EE A2B2       sll r3,2                        ;((IMP number-1)/8)*8
57F0 4A3A 0001  add r3,=<noimp0/words>(r2)      ;source IMP number (undoubled)
57F4 781C 001A  ldab r1,modem(r4)               ;which modem we are
57F8 7019 0EB6  lda r1,bittab(r1)               ;its bit
57FC 4C18 8000  ior r1,#rup4us                  ;and the for us bit
5800 4858 72AE  lda r5,=srupq                   ;check for unprocessed updates
5804 7078 B2C6  lock spfrtl                     ;lock out m2i
5808 9AFE
580A 7078 A4A4  lock lockro                     ;and routing output
580E 9AFE

                ;(OVER)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 225
ROUTE.PLR;1     PAGE 43          Routing Process


                ; RTRGEN continued

                repeat
5810 7025       ● lda r2,(r5)
5812 8904         if odd
5814 3008 A4A4      unlock lockro               ;nothing on queue for this IMP
5818 9026           break                       ;proceed to generate retrans
                  endif
581A 4078 2414    call rupwhc                   ;check for queue broken
581E 9A23         until fail                    ;can't contain another now
5820 707A 0350    lda r7,chan-chain(r2)
5824 4F17         if r1 .bit. r7                ;if for us to take care of
5826 9A1D
5828 1016           save r1                     ;save for later
582A 4078 1512      call unpack                 ;must look at this one
582E 7E39 0007      if byte r3 = srch+1(r1)     ;from same imp?
5832 8112
5834 3008 A4A4        unlock lockro             ;yes, wait until it's processed
5838 3008 B2C6        unlock spfrtl
583C 7078 00D2        setmap m2,mapv2
5840 3078 FC04
5844 6016             restore r1                ;fix the stack
5846 6056             fail return
5848 6046
584A 6036
584C 6026
584E 6016
5850 6076
5852 4FF0
5854 4007
                    endif
5856 7078 00D2      setmap m2,mapv2
585A 3078 FC04
585E 6016           restore r1                  ;restore for next iteration
                  endif
5860 4852         lda r5,r2
5862 90D7       endrepeat
5864 A2B1       sll r3,1                        ;double the imp number
5866 702B 72FC  lda r2,spfrut(r3)               ;get age and serial number
586A 3008 B2C6  unlock spfrtl                   ;done with spfrut
586E 704B 85E0  lda r4,ntb+words(r3)            ;compute number of line entries
5872 714B 85DE  sub r4,ntb(r3)
5876 4B48 03FE  and r4,=ntbidx
587A A6C1       srl r4,1
587C 9A08       if zero } r4 > =rupnn_- H8 } r3 = =0 } r3 > =nimp*words
587E 4ECF
5880 9C06
5882 4EB0
5884 9104
5886 4E38 00FE
588A 8C08
588C E507         Trap 2407,<:Retrans w/bad length or IMP - page 223>
588E 6056         return                        ;leave timer off
5890 6046
5892 6036
```

```
5896 6016
5898 6006
                endif
                ;(OVER)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 226
ROUTE.PLR;1     PAGE 44         Routing Process


                ; RTRGEN continued

589A 4852       lda r5,r2                       ;save spfrut entry
589C 4B28 0070  and r2,=spfage                  ;check age
58A0 9A41       if nzero                        ;don't retransmit if age zero
58A2 4C4A 0080    ior r4,=rupret_- H8(r2)       ;combine retry,age,number of lines
58A6 4078 13C6    call freget,whrut+nrut        ;all ok, try to get buffer
58AA 0088
58AC 8A05         if fail
58AE 48F1           lda r7,=1
58B0 3278 5EB0      addm r7,rtrnbf              ;count times couldn't get a buffer
58B4 90C9           fail return                 ;leave timer on, try again in 25ms
                  endif
58B6 4875         lda r7,r5                     ;copy spfrut entry
58B8 48D0         lda r5,=0                     ;clear checksum register
58BA 4B78 3F00    and r7,=spfsno                ;serial number
58BE A6B1         srl r3,1                      ;undoubled imp number
58C0 4C73         ior r7,r3                     ;combine
58C2 3079 0006    sta r7,srch(r1)
58C6 4957         sub r5,r7                     ;include in checksum
58C8 A2B1         sll r3,1                      ;double imp number again
58CA 1016         save r1-r2
58CC 1026
58CE 7078 00D2    setmap m1,mapv2               ;get to routing tables
58D2 3078 FC02
58D6 702B 65DE    lda r2,m1#ntb(r3)
58DA 4878 03FE    lda r7,=ntbidx
58DE 4B27         and r2,r7                     ;starting index for node's lines
58E0 737B 65E0    and r7,m1#ntb+words(r3)       ;first index for next node's lines
                  repeat                        ;loop through line entries
58E4 683A 62C0      ldab r3,m1#ltb+ltbdst(r2)+  ;get distance this line
58E8 4E38 00FF      if r3 = =dlinf              ;dead or usurped line
58EC 8104
58EE 4AA1             add r2,=1 ;;ltbnay        ;skip neighbor
58F0 49C1             sub r4,=rupnn1_-8         ;reduce line count this message
58F2 9008            else
58F4 A7B7             rrl r3,7                  ;make room for neighbor
58F6 6C3A 62C0         iorb r3,m1#ltb(r2)+ ;;ltbnay ;complete entry
58FA A7B1              rrl r3,1                 ;position data correctly
58FC 2039 0008         sta r3,seqh(r1)+         ;and store in buffer
5900 4953              sub r5,r3                ;adjusting checksun
                     endif
5902 4E27            while r2 < r7
5904 92F0
                  endrepeat
5906 7078 00D0    setmap m1,mapvar              ;fix map 1 now
590A 3078 FC02
590E 6026         restore r1-r2
5910 6016
5912 7036         lda r3,(sp)                   ;get original r5(modem number)
5914 48F1         lda r7,=1
5916 327B 5E98    addm r7,rtrcnt(r3)            ;count retrans on this line
591A 703B 0EB6    lda r3,bittab(r3)             ;mask bit for this line
591E 4078 5708    call rupbld                   ;share code with rupgen
                endif
```

5922 90B6 endroutine rtrgen

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 227
ROUTE.PLR;1     PAGE 45          Routing Process


FC00 0400               page PkgCode    ;called from modem code

.comnt |
Routines for turning retransmission timers on and off
|

5924 1076       Routine RTSET, arg r2/r4, result r1/r3, uses r2

5926 49A2       sub r2,=noimp0                  ;timers begin with IMP 1
5928 4812       lda r1,r2                       ;copy imp number (doubled)
592A A693       srl r1,3                        ;<impnumber/8>*words
592C 4A14       add r1,r4                       ;add into modem block index
592E 4BAE       and r2,= HE                     ;impnumber modulo 8(doubled)
5930 48B3       lda r3,=3                       ;compute mask for this IMP
5932 A232       sll r3,r2                       ;into position
5934 707C 40B8  lock rtimrl(r4)
5938 9AFE

593A 6006       endroutine rtset


593C 1076       Routine RTOFF, arg r2/r4,local r1,local r3,uses r2
593E 1016
5940 1036

5942 4078 5924  call rtset                      ;get index and mask for timer
5946 4D38 FFFF  eor r3,=-1                      ;complement mask
594A 3339 00BC  andm r3,rtimrs(r1)              ;turn off timer
594E 300C 40B8  unlock rtimrl(r4)

5952 6036       endroutine rtoff
5954 6016
5956 6006


5958 1076       Routine RTON, arg r2/r4,local r1,local r3,uses r2
595A 1016
595C 1036

595E 4078 5924  call rtset                      ;get index and mask for timer
5962 3439 00BC  iorm r3,rtimrs(r1)              ;turn on two bit timer
5966 300C 40B8  unlock rtimrl(r4)

596A 6036       endroutine rton
596C 6016
596E 6006
```

```
.comnt |
TICAGE - tick ages in spf database.
Called every slow tick.
Does a fraction of the IMPs each slow tick so that all IMPs are
ticked once during the tick interval.
|
FC00 0200               page DDTCode
     000C       spftic= D8000/ D640             ;8 seconds in slow ticks

4F1A 1076       Routine TICAGE, uses r1-r4

4F1C 7818 72A8  ldab r1,ticimp                  ;get last IMP ticked.
4F20 8A03       if zero                         ;time to reset?
4F22 4818 00FE    lda r1,=nimp*words            ;yes, start at highest IMP
                endif
4F26 48AB       lda r2,=<nimp-1/spftic>+1       ;number of IMPs to do per slow tick
4F28 4838 0070  lda r3,=spfage                  ;mask for age field
4F2C 4848 0010  lda r4,=spfag1                  ;one in age field
4F30 7078 B2C6  lock spfrtl
4F34 9AFE
                repeat
4F36 5739 72FE    if r3 .bit. spfrut+noimp0(-r1);only tick non-zero ages
4F3A 9A05
4F3C 3149 72FE      subm r4,spfrut+noimp0(r1)   ;tick age
4F40 3248 72C4      addm r4,spfsum              ;and checksum
                  endif
4F44 9803         until loop                    ;can exit last fraction early
4F46 49A1         sub r2,=1                     ;count IMPs done this loop
4F48 8AF7         until zero                    ;exit when fraction done
                endrepeat
4F4A 3008 B2C6  unlock spfrtl
4F4E 3818 72A8  stab r1,ticimp                  ;save for next time

4F52 6006       endroutine ticage

FC00 0200             dstolist ticage           ;put this on ddt timeout list
4EDA 4F1A
4EDC 1CBC
4EDE 0000

FC00 0400               page RutCode
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 229
ROUTE.PLR;1     PAGE 47         Routing Process


.comnt |
RUPQCK - verify rupq count of packets with tsk4us bit on
Called every slow timeout
|

5970 1076       Routine RUPQCK, arg m2, uses r1-r5

                repeat
5972 4890         lda r1,=0                     ;init queue counter
5974 7048 72AA    lda r4,rupmsk                 ;mask of live lines
5978 4858 72AE    lda r5,=srupq
597C 7078 A4A4    lock lockro                   ;lock the routing queue
5980 9AFE
                  repeat
5982 7025           lda r2,(r5)                 ;next entry on queue
5984 890B           if odd
5986 7038 5EB2        lda r3,rupqct             ;end of queue, get old count
598A 4E13             if r1 <> r3               ;do they differ?
598C 9104
598E E508               Trap 2410,<;Rupqct wrong - page 227>
5990 3018 5EB2          sta r1,rupqct           ;correct it
                      endif
5994 3008 A4A4        unlock lockro
5998 6006             return                    ;done
                    endif
599A 4078 2414      call rupwhc                 ;check for queue broken
599E 9A12           until fail                  ;start all over again if so
59A0 774A 0350      if r4 .nbit. chan-chain(r2) ;no more to do?
59A4 8A0B
59A6 7072             lda r7,(r2)               ;dequeue this one
59A8 3075             sta r7,(r5)
59AA 8903             if odd                    ;new end of queue
59AC 3058 72B0          sta r5,erupq
                      endif
59B0 E509             Trap 2411,<;Recovered unused buffer - page 227>
59B2 4078 1332        call flush,whrut          ;free it
59B6 0080
59B8 90E5             next
                    endif
59BA 8B02           if minus            ;;rup4us
59BC 4A91             add r1,=1                 ;count one for our SPF
                    endif
59BE 4852           lda r5,r2                   ;chain to next entry
59C0 90E1         endrepeat
59C2 90D8       endrepeat

59C4 90EA       endroutine RUPQCK

     0008               .radix  010     ;back to octal
```

```
.INSERT "TASK"
    .INSRT TASK
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 231
TASK.PLR;1      PAGE 1  TASK- Store and Forward

                .stitle TASK- Store and Forward
                ;TASK local variables

                        PAGE Lvars

027E            tskbuf: .blkw 1 ;buffer we are working on
                repbit:         ;reply's tstate bit (overlays ralshf)
0280            ralshf: .blkw 1 ;amount to shift rstate,rtype
0282            tskhst: .blkw 1 ;which host (hostno)
0284            tskbts: .blkw 1 ;saved copy of task state bits
```

```
                ;************task**********

                        Page LCode

                ;r1=buffer pointer
                ;r2=packet pointer (CHAIN word)
                ;r3=CHAN word
                ;r4=input modem block
                ;r5=output modem block
                ;r6=sp
                ;r7=subroutines,temp

2436 1076       routine tsk,uses r1-r5

2438 7078 A3B6    lock ltq                      ;access TASK queue
243C 9AFE
243E 4818 63B8      lda r1,=stq
2442 4078 154A      call deque,whtsk
2446 0001
2448 8A04           if fail                     ;none to process
244A 3008 A3B6        unlock ltq
244E 6006             return                    ;back to loop
                    endif
2450 3008 A3B6    unlock ltq                    ;done with queue
2454 4878 0040    lda r7,=task
2458 30F8 00AC    sta r7,@pid                   ;run TASK on next packet
245C 7049 0092    lda r4,inch(r1)               ;source of packet
2460 7071         if (r1) .bit. =dscpkt ;;neth
2462 4F78 2000
2466 9A03
2468 E421           Trap 2041,<;Pkt w/discard bit discarded - page 230>
246A 9029          else
246C 7079 000C      lda r7,dsth(r1)
2470 9A25           ifnot zero                  ;zero IMP illegal

                ;more of TASK over leaf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 233
TASK.PLR;1      PAGE 3  TASK- Store and Forward


               ;TASK continued

2472 A2F1              sll r7,1
2474 705F 72FC         lda r5,spfrut(r7)          ;get proper S/F route
2478 9B21              ifnot minus                ;IMP isn't dead
247A 4BDF                and r5,=routef           ;which route?
247C 8A06                if zero                  ;for us)
247E 4078 24C8             call tskack            ;ACK its source then
2482 4861                  lda r6,r1              ;old buffer pointer reg.
2484 4078 2FAC             call forus             ;and process the packet
                ;          return                 ;FORUS doesn't return
                         endif
2488 A2D1                sll r5,1                 ;look up modem block
248A 705D 635C           lda r5,m2pblk-words(r5)
248E 8B03                if minus                 ;this modem doesn't exist?
2490 E420                  Trap 2040,<;No rte for task pkt - page 231>
2492 9011                 else
2494 4078 25E8             call ncmove,cntrs+nsf ;move count into S/F
2498 63DE
249A 9A0D                  if success
249C 787D 0002               if byte phflag(r5) .bit. =phup      ;line's up
24A0 4FF4
24A2 9A09
24A4 707D 0050                 lda r7,slots(r5)  ;any free slots?
24A8 9B06                      ifnot minus       ;yes, we can forward it
24AA 4078 24C8                   call tskack     ;ack source
24AE 4078 2554                   call tsksf      ;queue it on output modem
24B2 90CE                        return          ;all done
                               endif
                             endif
                ;            if r3 <> =0         ;from a modem:
                ;              call tskreq       ;requeue it for task later
                ;              return
                ;            endif
                           endif
                         endif
24B4 4078 2506           call tsknak             ;refuse and flush a packet
24B8 90CB                return
                       endif
                     endif
24BA E5C3            Trap 2703,<;Flushing pkt for dead IMP - page 231>
                   endif
24BC 4078 24C8     call tskack                   ;ACK its source
24C0 4078 131E     call flushb,whtsk             ;get rid of packet
24C4 0001

24C6 90C4        endroutine tsk
```

```
                ;TSKACK
                ;TASK routine to ACK the source of this packet
                ;R1/M2 are packet pointer, R3 CHAN word this packet,
                ;R4 = INCH (packet source)
                ;take appropriate action based on R4 and R3

24C8 1076       routine tskack,arg r3/r4/m2

24CA 4EC0         if r4 > =0                    ;not a reroute or requeued pkt
24CC 8C1C
24CE 4EB0           if r3 = =0                  ;from a host
24D0 8105
24D2 4078 2598        call tskakh,tskfok        ;ACK the host
24D6 0001
24D8 9016           else                        ;from a modem
24DA 707C 4000        lock lockm(r4)
24DE 9AFE
24E0 4873             lda r7,r3                 ;which ACK group
24E2 A6B8             srl r3, H8                ;from CHAN left half
24E4 4A43             add r4,r3
24E6 3C7C 0072        iorbm r7,infree(r4)       ;channel is free
24EA 3D7C 0062        eorbm r7,rsex(r4)         ;complement odeven sex
24EE 4943             sub r4,r3
24F0 48F1             lda r7,=1                 ;count a packet ACK'ed
24F2 327C 004E        addm r7,mithru(r4)
24F6 A273             sll r7,r3
24F8 347C 0030        iorm r7,snull(r4)         ;set bit of group to ACK
24FC 300C 4000        unlock lockm(r4)
2500 4078 14F0        call trymod
                    endif
                  endif

2504 6006       endroutine tskack
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 235
TASK.PLR;1      PAGE 5  TASK- Store and Forward

                ;TSKNAK
                ;NACK (negative acknowledge) the source of this packet
                ;R1/M2 are packet pointer, R2 packet CHAIN word,
                ;R3 CHAN word this packet, R4 = INCH (packet source)
                ;Returns M2 = MAPV2
                ;take appropriate action based on R4, R3

2506 1076       routine tsknak,arg r1-r4/m2,result m2

2508 4EC0         if r4 > =0                    ;not from reroute or TASK
250A 8C15
250C 4078 131E      call flushb,whtsk           ;get rid of buffer
2510 0001
2512 4EB0           if r3 = =0                  ;from a host
2514 8105
2516 4078 2598        call tskakh,tskfrf        ;refuse it
251A 0002
251C 900B            else                       ;from a modem
251E 707C 4000        lock lockm(r4)            ;get modem
2522 9AFE
2524 4873             lda r7,r3
2526 A6F8             srl r7, H8                ;ACK group
2528 4A74             add r7,r4
252A 3C3F 0072        iorbm r3,infree(r7)       ;free this input channel
252E 300C 4000        unlock lockm(r4)          ;done modem
                    endif
2532 6006           return
                  endif

2534 9002       entry tskreq                    ;enter here to requeue for TASK
2536 1076

2538 7078 00D2    setmap m2,mapv2               ;restore map 2
253C 3078 FC04
2540 7078 A3B6    lock ltq                      ;access task's aux. queue
2544 9AFE
2546 30A8 63BE    sta r2,@erq
254A 3028 63BE    sta r2,erq
254E 3008 A3B6    unlock ltq                    ;done task's queues

2552 90F0       endroutine tsknak/tskreq
```

```
                ;TSKSF
                ;Process a TASK packet for store/forward
                ;R1/M2 packet pointer, R2 packet CHAIN word
                ;R3 CHAN word for packet, R4 INCH for packet
                ;R5 is destination modem

2554 1076       routine tsksf, arg r1/r2/r4/r5/m2

2556 48F0         clear inch(r1)                ;note this packet is S/F'd
2558 3079 0092
255C 70F8 62E8    set qt(r1) = @clock   ;;TRACE ;note queue time for trace
2560 3079 0098
2564 484D 003A    lda r4,=epriq(r5)             ;assume it's priority
2568 7079 0002    if typh(r1) .nbit. =pribit    ;it's regular
256C 4F78 1000
2570 8A02
2572 4AC4           add r4,=eregq-epriq
                  endif
2574 4078 146C    call wheorb,whtsk?whi2m       ;change ownership to M2I
2578 0101
257A 48B1         lda r3,=1
257C 707D 4000    lock lockm(r5)                ;get the modem block
2580 9AFE
2582 313D 0050    subm r3,slots(r5)             ;reduce available channels
2586 30A4         sta r2,@(r4)  ;;sregq sprig   ;and enqueue the packet
2588 3024         sta r2,(r4)   ;;eregq epriq
258A 300D 4000    unlock lockm(r5)              ;done with modem
258E 787D 000E    ldab r7,motpid(r5)            ;start it
2592 30F8 00AC    sta r7,@pid

2596 6006       endroutine tsksf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 237
TASK.PLR;1      PAGE 7  TASK- Store and Forward

                ;TSKAKH
                ;ACK or NACK a host from TASK.
                ;Inline arg: TSKFOK or TSKFRF bit to set in HIBITS
                ;R4 has host address. Also optionally free the trn blk
                ;associated with this message for host (HITRAN, TSKFRE)

2598 1076       routine tskakh,arg r4,inline r1,local r1
259A 1016
259C 6017
259E 307E 0002

25A2 707C 4000    lock lockhi(r4)               ;get access to host
25A6 9AFE
25A8 7C1C 001E    iorb r1,hibits(r4)
25AC 8914         if odd & r1 .bit. =tskfre     ;;tskfok
25AE 4F94
25B0 9A12
25B2 4D94           eor r1,=tskfre
25B4 1036           save r3                     ;need another temp now
25B6 703C 402E      lda r3,hitran+foo(r4)       ;host's trn blk
25BA 7078 00D2      setmap m1,mapv2             ;to get to tran block
25BE 3078 FC02
25C2 4878 FF7F      lda r7,=-1?ttresv
25C6 3B7B E00D      andbm r7,trstat+<m1-m2>(r3) ;release trn blk to task
25CA 7078 00D0      setmap m1,mapvar            ;restore map for everything else
25CE 3078 FC02
25D2 6036           restore r3
                  endif
25D4 381C 001E    stab r1,hibits(r4)            ;host's flags
25D8 300C 4000    unlock lockhi(r4)
25DC 787C 000F    ldab r7,hinpid(r4)            ;get the host running now
25E0 30F8 00AC    sta r7,@pid

25E4 6016       endroutine tskakh
25E6 6006
```

```
                ;CMOVE
                ;Move the count for this buffer into a new pool
                ;Called with R2/ buffer CHAIN address, inline/ address
                ;of new counter to use. Fail return if no count
                ;available. Return with map 2 set for buffer.

25E8 1076       routine ncmove,inline r1,arg r2,local r1,local r3-r4,result m2
25EA 1016
25EC 1036
25EE 1046
25F0 6017
25F2 307E 0006

25F6 9005       entry xcmove,arg r1-r2,local r1,local r3-r4,result m2
25F8 1076
25FA 1016
25FC 1036
25FE 1046

2600 7078 00D2    setmap m2,mapv2
2604 3078 FC04
2608 7038 A3C8    lock r3,nf                    ;access buffer variables
260C 9AFE
260E 704A 06A0    lda r4,where-chain(r2)        ;old count of this buffer
2612 4B48 003E    and r4,=whctr
2616 4A48 63D8    add r4,=cntrs
261A 4E41         if r4 <> r1                   ;need to move it at all
261C 911E
261E 7071           lda r7,(r1)                 ;value of new count
2620 9B14           ifnot minus                 ;it needs some of pool
2622 7638 63D0        if r3 <= minf             ;but pool is exhausted
2626 9C10
2628 7074               if (r4) <= =0           ;and old count isn't pool
262A 4EF0
262C 9C0D
262E 3038 A3C8            unlock r3,nf          ;can't use new count
2632 707A FCB0            setmap m2,point-chain(r2)     ;fix map2
2636 3078 FC04
263A 6046                 fail return
263C 6036
263E 6016
2640 6076
2642 4FF0
2644 4007
                        endif
                      endif
2646 49B1             sub r3,=1                 ;borrow from pool
                    endif
                ;OVER
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 239
TASK.PLR;1      PAGE 9  TASK- Store and Forward

                ;CMOVE continued - finish up counters

2648 48F1           lda r7,=1
264A 3174           subm r7,(r4)                ;reduce old count
264C 9B02           ifnot minus                 ;it was from pool
264E 4AB1             add r3,=1                 ;return pool count
                    endif
2650 3271           addm r7,(r1)                ;increase new count
2652 4914           sub r1,r4
2654 321A 06A0      addm r1,where-chain(r2)     ;and fix WHERE for new cnt
                  endif
2658 3038 A3C8    unlock r3,nf                  ;done counters
265C 707A FCB0    setmap m2,point-chain(r2)     ;set map2
2660 3078 FC04

2664 6046       endroutine ncmove/xcmove
2666 6036
2668 6016
266A 6006

                ;Old CMOVE
                ;Inline is new counter, R2 is buffer, return uses
                ;M2/R6 for buffer by old convention

266C 6017       routine ocmove,nosave,arg r2/r6,inline r1,uses r1/r3

266E 4836         lda r3,r6                     ;preserve old R6
2670 4868 02F8    lda sp,=lstack                ;and get a stack
2674 1076         push r7                       ;save our caller
2676 4078 25F8    call xcmove                   ;our private entry
267A 9A05         if success                    ;we'll skip on success
267C 6076           pop r7                      ;fix up regs
267E 4863           lda r6,r3
2680 400F 0002      skip return                 ;signal success
                  endif
2684 6076         pop r7
2686 4863         lda r6,r3                     ;fix regs

2688 4807       endroutine ocmove               ;non-skip on fail
```

```
.INSERT "IMP.WARM",WARM
    .INSRT IMP.WARM
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 241
IMP.WARM;1      PAGE 1  TASK- Store and Forward


                .stitle  *** warm page ***

                        page HLCode
                ; *** warm hi ***

                ; start at hi with hilo initialized to higo

                ;receive a host going down

268A 701B 0008  hidown: lda r1,trmidl(r3)       ;get dead status
268E 301C 002C          sta r1,deadsc(r4)

2692 4868 02F8  hidisc: lda sp,=lstack
                        repeat
2696 705C 0020            lda r5,iobloc(r4)
269A 702C 4034            lda r2,hisp+foo(r4)
269E 9A04                 ifnot zero    ;buffer in progress.
26A0 4078 1332              call flush,whhi
26A4 2000
                          endif
26A6 702C 0032            lda r2,hibf(r4)
26AA 9A04                 until zero
26AC 4078 2C9E            jsb r7,hinbwt ;in case i/o is active
26B0 90F3               endrepeat
26B2 701D 0004  hidis3: lda r1,endi(r5) ;;hendf ;eom?
26B6 9B20                bm hiwait      ;found the end of the message being flushed
26B8 7028 63CA          lda r2,junk     ;input into junk
26BC 4868 02F8          lda sp,=lstack
26C0 4078 15AA          call hsioin,0,bufend
26C4 0000
26C6 008E
26C8 4078 2B70  hidis4: jsb r7,hiwfe    ;waits here during nice stop
26CC 707C 4032          lda r7,hibf+foo(r4)     ;;junk  ;get rid of input
26D0 90F1               br hidis3
```

```
                ;imp has reset host interface
                ;wait 2/3 second before proceeding

26D2 8A04       hiidle: repeat while zero
26D4 4078 2B3A            call hiwm
26D8 90FD               endrepeat
26DA 4891               lda r1,=hrdown
26DC 381C 001F          stab r1,hihd(r4) ;start host as down
26E0 3008 6512          sta r0,ophgo     ;tell the ncc
26E4 4078 2B3A          jsb r7,hiwm      ;higo is a legit dispatch (from bldf/rh)
26E8 701C 0002  higo:   lda r1,hostyp(r4)        ;TIP, VDH or real?
26EC 7079 0EB6          lda r7,bittab(r1)        ;get bit for this host type
26F0 7778 18DA          tst r7,nonops
26F4 9ADF               bz hidis3        ;normal host: discard first msg

                ; hihd    0=> up    1=>down ready line    2=>down tardy
                ;   4=>nice-stop or soft reset in progress

26F6 7018 62E0  hiwait: lda r1,myimp
26FA 9BE7                bm hidis4       ;going down nice stop -- wait
26FC 4878 FFF3          lda r7,=-1?tskfre?hirset
2700 3B7C 001E          andbm r7,hibits(r4)      ;some error paths need this
2704 703C 002E          lda r3,hitran(r4)
2708 8A07               if zero          ;need new trnblk
270A 4078 2CCC            jsb r7,hittgo ;give us some time
270E 4078 5AE2            jsb r7,trnput ;get a new trnblk
2712 303C 002E            sta r3,hitran(r4)
                        endif
                hicuml: ;cumstats hook.
2716 787C 001C          ldab r7,homode(r4)
271A 9A37                bz hiled2       ;if expecting old format
271C 482B 800A          lda r2,=trnetl+<words*5>-m2(r3)
2720 4078 2C1C          jsb r7,hiledi    ;read 6 words
2724 812D                bne hils1       ;if old format
2726 9A03                tnz hildi2      ;if too short
2728 4008 2C12
272C 701B 0002          lda r1,trtypl(r3)
2730 381C 0027          stab r1,hipkth+1(r4)
2734 4B18 0F00          and r1,=trcbit+pflags
2738 301B 0002          sta r1,trtypl(r3)
273C 701B 0004          lda r1,trhstl(r3)
2740 4B18 F8FF          and r1,=-1?maxled        ;;getmax ;temp default
2744 301B 0004          sta r1,trhstl(r3)
2748 4B18 F000          and r1,=getpri}getmax    ;bits for mes blk
274C 8B03               bnm .+6
274E 4C18 0800           ior r1,#getpri_-4
2752 A294               sll r1,4
2754 301C 002A          sta r1,hihand(r4)
2758 9055               br hiled4
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 243
IMP.WARM;1      PAGE 3  *** warm page ***


                ;switch to new format if nop

275A 4B28 00FF  hiled5: and r2,=mestyp
275E 4EA4               cmp r2,=cnop
2760 8111                bne hils2              ;bad leader format
2762 707C 0002          lda r7,hostyp(r4)       ;check for any non-real
2766 9A03                tnz hinop      ;:VDH hosreal :already read 6 words
2768 4008 2B10
276C 482B 8006          lda r2,=trledr+<words*3>-m2(r3)
2770 4078 2C16          jsb r7,hiledm   ;read 4 more words
2774 8ADA                tnz hildi2     ;too short
2776 701B 0004          lda r1,trmidl-<2*words>(r3)
277A 4008 2B14          tr hinop3

                ;switch to old format if nop

277E 4078 2B28  hils1:  jsb r7,hinopt   ;maybe its a NOP
2782 48A3       hils2:  lda r2,=cwrgft  ;wrong format leader
2784 4008 2AFC          tr hierr
```

```
                ;here expecting old format leader

2788 482B 8002  hiled2: lda r2,=trledr+words-m2(r3)     ;;trtypl
278C 4078 2C1C          jsb r7,hiledi    ;input two words
2790 8ACC                tnz hildi2     ;if too short
2792 702B 0002            lda r2,trledr+words(r3)       ;;trtypl
2796 91E2                be hiled5      ;if new format
2798 4078 2B28          jsb r7,hinopt    ;a NOP?
279C 302B 0008          sta r2,trmidl(r3)
27A0 7013               lda r1,(r3)     ;;trledr
27A2 A792               rrl r1,2        ;place bits for:
27A4 4B18 0C00          and r1,=trcbit+octbit   ;trace and octal flags
27A8 301B 0002          sta r1,trtypl(r3)       ;to type header
27AC 7013               lda r1,(r3)     ;;trledr ;once again:
27AE 4B18 003F          and r1,=desti   ;just IMP number
27B2 301B 0006          sta r1,trdstl(r3)       ;that's dest leader
27B6 7013               lda r1,(r3)     ;;trledr ;once again:
27B8 A696               srl r1,6        ;host to bottom
27BA 4B18 0103          and r1,=<forimp+desth>_-6       ;host bits
27BE 4F18 0100          if r1 .bit. =forimp_-6  ;for fake host?
27C2 9A02
27C4 4994               sub r1,=nfh     ;fakes are 374-377 (octal)
                        endif
27C6 301B 0004          sta r1,trhstl(r3)       ;dest host leader
27CA 7013               lda r1,(r3)     ;;trledr ;(yawn) once again:
27CC 4B18 8000          and r1,=priled  ;for priority check
                ;;      ior r1,=getmax  ;until further notice
27D0 301C 002A          sta r1,hihand(r4)       ;remember for later
                ;;      and r1,=getpri}priled   ;just priority again
                ;;      ior r1,=maxled  ;until further notice
27D4 341B 0004          iorm r1,trhstl(r3)      ;into leader
27D8 7813               ldab r1,(r3)    ;;trledr
27DA 4878 0F00          lda r7,=hicode  ;make leader pretty
27DE 3073               sta r7,(r3)     ;;trledr trnetl
27E0 4B9F               and r1,=hicode_- H8     ;low 4 bit are msg type
27E2 9A05                bz hiled8      ;reg mess
27E4 4E93               cmp r1,=rawpkt
27E6 9103                be hiled8
27E8 4E94               cmp r1,=cnop    ;NOP
27EA 810A                bne hiled9
27EC 4828 FFF0  hiled8: lda r2,=-1?subtyp       ;clear subtyp
27F0 332B 0008          andm r2,trmidl(r3)
27F4 4E93               cmp r1,=rawpkt
27F6 8104                bne hiled9
27F8 341B 0008          iorm r1,trmidl(r3)      ;set subtyp to 3
27FC 4890               lda r1,=0               ;now a regular message}
27FE 381C 0027  hiled9: stab r1,hipkth+1(r4)    ;new format mess type
                ;continues over leaf
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 245
IMP.WARM;1      PAGE 5  *** warm page ***

2802 781C 401F  hiled4: ldab r1,hihd+foo(r4)    ;;hostup
2806 9A05               ifnot zero              ;host was down.
2808 701C 402C            lda r1,deadsc+foo(r4)
280C 3008 6512            sta r0,ophgo
                        endif

                ;now dispatch on message type

2810 781C 0027          ldab r1,hipkth+1(r4)    ;message type
2814 A291               sll r1,1                ;set up for dispatch
2816 4E18 0011          cmp r1,=hide-hi2-1      ;out of range?
281A 8C03                tl hibadc      ;yes
281C 4008 2AFA
2820 4089 2824          jmp @hi2(r1)    ;dispatch on msg type

                ;dispatch table
2824 2836       hi2:    hireg   ;0 - reg  (patched to HIGTWY by PTIP)
2826 2692               hidisc  ;1 - imp format error
2828 268A               hidown  ;2 - host going down
282A 2AFA               hibadc  ;3
282C 2B10               hinop   ;4 - nop
282E 2692               hidisc  ;5 - mess for discard
2830 2AFA               hibadc  ;6
2832 2AFA               hibadc  ;7 - used in code for hibadc reference
2834 2B0C               hiifei  ;8 - imp format error with id
                hide:
```

```
                ;got a regular message

2836 4878 4085  hireg:  lda r7,= H4000+ttreq+ttrfnm+ttresv ;1-pkt request
283A 307B 000C          sta r7,trntim+<0*trstat>(r3)    ;state for errors
283E 782C 001C          ldab r2,homode(r4)      ;padding mode
2842 4928 8082          sub r2,= H80+2+m2
2846 8B06                bnm hireg4     ;don't need to read any padding
2848 4838 8F50          lda r3,=hipad   ;padding goes here
284C 4A23               add r2,r3
284E 4078 2C16          jsb r7,hiledm   ;read the padding
2852 701D 0004  hireg4: lda r1,endi(r5) ;;hendp
2856 8B03                tm hildi2      ;if leader is too short
2858 4008 2C12
     0001       .IF NZ VHA
285C 4868 02F8          lda sp,=lstack          ;get us some temps
2860 4078 3D40          call hivha              ;modify leader if need be
2864 8A04          .    if fail                 ;troubles doing it
2866 48A2                 lda r2,=cldrp         ;dest dead subcode
2868 4008 2ADE            tr hidead
                        endif
                .ENDC
                higtgo:                 ;patched by PTIP
286C 781B 0005          ldab r1,trhstl+1(r3)    ;dest host
2870 381C 0028          stab r1,hihost(r4)      ;for mess num search
2874 701B 0006  hireg5: lda r1,trdstl(r3)       ;dest IMP

2878 9B12         ifnot minus } zero } r1 > =nimp
287A 9A11
287C 4E18 007F
2880 9C0E
2882 7028 62E0      lda r2,myimp                ;check if we are up
2886 9A07           if nzero & r1 <> mine
2888 7618 62E2
288C 9104
288E 4078 2B3A        call hiwm                 ;must wait if not, and not for us
2892 90F1             br hireg5                 ;try again
                    endif
2894 A291           sll r1,1                    ;double IMP number
2896 7019 72FC      lda r1,spfrut(r1)           ;get route to this IMP
289A 8B03           bnm hireg3  ;;spfded        ;branch if not dead
                  endif
289C 4008 2ADC    jmp hiimpd                    ;destination dead
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 247
IMP.WARM;1      PAGE 7  *** warm page ***


                ;Fast Local Hosts here:
                ;  (use trnblk as reassem blk)
                ; 2. Call hinbuf
                ; 3. Get nre, return nhi space
                ; 4. Bufb gets Tran blk addr
                ;    (plus sign to tell IH to queue RFNM)
                ; 5. Queue buff in tran blk
                ; 6. If not last pkt go to 2.
                ; 7. Link packets and call t2h0

28A0 4078 2CCC  hireg3: jsb r7,hittgo   ;give us 15 seconds
28A4 4078 2C60          jsb r7,hinbuf   ;start reading a pkt
28A8 8A03                tz hiblk       ;couldn't get a buffer
28AA 4008 2B06
28AE 707B 0008          lda r7,trmidl(r3)       ;raw packet (type 3)?
28B2 4BFF               and r7,=subtyp
28B4 4EF3               cmp r7,=rawpkt
28B6 812D                bne hireg6     ;if not a raw packet
28B8 4078 2C9E          jsb r7,hinbwt   ;wait for read to complete
28BC 8A28                bnz hirds      ;timed out
28BE 4868 02F8          lda sp,=lstack
28C2 4078 2CD6          call hiset      ;use any old rmblk (or 0)
28C6 9A23                bfail hirds    ;hisp failure.
28C8 4865               lda r6,r5
28CA 48F3               lda r7,=rawpkt
28CC 3471               iorm r7,(r1)    ;;midh
28CE 4078 5A96          jsb r7,fndhac   ;check hac words
28D2 301E 000A          sta r1,pkth(r6) ;;hacmem
28D6 8917                bno hirmds     ;raw packets prohibited
28D8 707D 40D0          lda r7,haccom(r5)
28DC 701C 0036          lda r1,hiendi(r4)
28E0 9912                bo hirmds      ;hardware error
28E2 8B11                bnm hirmds     ;multi-packet message
28E4 2079 0002          sta r7,words(r1)+       ;;m2
28E8 4E1E 008C          cmp r1,=bufend-words(r6)
28EC 9C0C                bl hirmds      ;too long by one word
28EE 4078 2D5A          jsb r7,hipktr   ;checksum packet
28F2 4078 2B96          call hi2tsv     ;give packet to task
28F6 9A0B                bz hirds       ;blocked by task
28F8 707C 4026          lda r7,hipkth+foo(r4)   ;for thruput
28FC 702C 4034          lda r2,hisp+foo(r4)
2900 4008 2A06          tr hirfsh       ;flush and count raw packet

2904 7078 00D2  hirmds: lda r7,mapv2
2908 3078 FC04          sta r7,%map2
290C 4008 2692  hirds:  tr hidisc
```

```
2910 4078 2DD4  hireg6: jsb r7,mesget   ;get a mess #
2914 307B 000A          sta r7,trluse(r3)       ;note local use number
2918 301B 0004          sta r1,trhstl(r3)       ;has messno, tm blk num
291C 302C 0026          sta r2,hipkth(r4)       ; and foreign use number

                ;trace goodies go here

2920 4078 2C9E          jsb r7,hinbwt   ;wait for buffer to fill up.
                ;waiting for first pkt
2924 9A03                tnz hipslo     ;if timer went off
2926 4008 2A7A
292A 4877               lda r7,r7       ;;hendf
                higtx:                  ;PTIP return point
292C 9B5A                bm hipk1a      ;yes, single pkt
292E 8903                to hiperr      ;;herrf ;if ready line flapped
2930 4008 2A82
2934 4078 2C60          jsb r7,hinbuf   ;start next pkt read
2938 9A33                bz hipblj      ;no buffer
293A 4896               lda r1,=ttmult?ttreq    ;mark trnblk as a multi
293C 3D1B 000D          eorbm r1,trstat(r3)     ;  non-request
2940 4078 2EFA          jsb r7,tallyg   ;any alloc from our dest?
2944 9072                br hiplt0      ;yes
                ;get an allocate for this guy
2946 4818 8001          lda r1,=mltpkt}reqtyp   ;mark us a multi
294A 341C 0026          iorm r1,hipkth(r4)      ;in our pkth
294E 4868 02F8          lda sp,=lstack
2952 4078 2CD6          call hiset      ;copy header
2956 9A38                bfail hiplc7   ;hisp failure.
2958 4865               lda r6,r5
295A 4078 2D54          jsb r7,hipkte   ;set up checksum
295E 4078 5AE2          jsb r7,trnput   ;get a new trnblk
2962 303C 000A          sta r3,hisav7(r4)       ;remember new trnblk
2966 705C 002E          lda r5,hitran(r4)
296A 4898               lda r1,=trnl/2
296C 6075       hicpt1: lda r7,(r5)+    ;copy in old trnblk contents
296E 2073               sta r7,(r3)+
2970 4991               sub r1,=1
2972 8AFD                bnz hicpt1
2974 4818 4086          lda r1,= H4000+ttreq+ttmult+ttresv ;remember the request
2978 301D FFFC          sta r1,trntim+<0*trstat>-trnl(r5)  ;set up old trnblk
297C 702C 0034          lda r2,hisp(r4)
2980 4078 2B9E          jsb r7,hi2tsk
2984 4853               lda r5,r3       ;start using new trnblk
2986 703C 000A          lda r3,hisav7(r4)
298A 303C 002E          sta r3,hitran(r4)
298E 4877               lda r7,r7       ;did hi2tsk block?
2990 8A08                bnz himg8m
2992 387D 000D          stab r7,trstat(r5)      ;free trn blk
2996 4858 7FFF          lda r5,=-1?mltpkt
299A 335C 0026          andm r5,hipkth(r4)
299E 907F       hipblj: br hipblk       ;and maybe this might help?
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 249
IMP.WARM;1      PAGE 9  *** warm page ***

29A0 4078 2DD4  himg8m: jsb r7,mesget   ;get new mess no for mess8
                ;   trluse should be unchanged
29A4 302C 0026          sta r2,hipkth(r4)
29A8 301B 0004          sta r1,trhstl(r3)       ;messno, tm blk to trnblk
29AC 702C 0034  hiwall: lda r2,hisp(r4) ;wait for req to go out
29B0 701A 06A0          lda r1,where-chain(r2)
29B4 4F18 DFC1          tst r1,=-1?whctr?whhi
29B8 9A0A                bz hireq3      ;if it's gone
29BA 4078 2B3A          jsb r7,hiwm
29BE 8AF7                bnz hiwall
29C0 E68F               Trap 3217,<;Host blkd requesting alloc - page 247>
29C2 4008 2AC8          tr hiures       ;can't reuse this buffer

29C6 E6C6       hiplc7: Trap 3306,<;Clbbrd hisp requesting alloc - page 247>
29C8 4008 2AC8          jmp hiures      ;release trnblk for inc timeout

                ;now ensure we have an allocate

29CC 4078 2EFA  hireq3: jsb r7,tallyg   ;any alloc from our dest
29D0 902C                br hiplt0      ;yes
29D2 4078 2B3A          jsb r7,hiwm
29D6 8AFB                bnz hireq3
29D8 E68D               Trap 3215,<;Host blkd awaiting alloc - page 247>
29DA 9061               br hipblk

29DC E6C7       hipk1c: Trap 3307,<;Hisp clbbrd in pkt - page 247>
29DE 9075               br hiures       ;let the mess time out

29E0 4818 4001  hipk1a: lda r1,=reqtyp+lstpkt   ;mark as req for 1
                hicumt:         ;hook for cum stats.
29E4 341C 0026          iorm r1,hipkth(r4)      ;mark up hi pkth
29E8 302B 000E          sta r2,trpack(r3)
29EC 4078 2D60  h2tx:   jsb r7,hipkt
29F0 9AF6                bz hipk1c      ;hisp was bad
                hicump:         ;hook for cum stats.
29F2 4078 2B96          jsb r7,hi2tsv
29F6 9A53                bz hipblk
29F8 702C 4034          lda r2,hisp+foo(r4)
29FC 701C 0026          lda r1,hipkth(r4)
2A00 4F18 00F0          if r1 .bit. =pktnum     ;single pkt message.
2A04 9A04
                          hirfsh:
2A06 4078 1314            call oflush,whhi      ;last pkt of multi (or raw pkt).
2A0A 2000
                        endif
2A0C 705C 000A          lda r5,hisav7(r4)       ;dest of this message
2A10 781C 0027          ldab r1,hipkth+1(r4)
2A14 48B0               lda r3,=0
2A16 4868 02F8          lda sp,=lstack
2A1A 4078 5B24          call hothru,htpmtn
2A1E 003A
2A20 705C 0020          lda r5,iobloc(r4)
2A24 4008 26F6          jmp hiwait
```

```
                ;top of packet loop

2A28 4818 0040  hiplt0: lda r1,=ttgvba  ;now we have an alloc
2A2C 3C1B 000D          iorbm r1,trstat(r3)
2A30 4818 8000          lda r1,=mltpkt
                hicumm:         ;hook for cum stats.
2A34 341C 0026          iorm r1,hipkth(r4)
2A38 4078 2D60  himess: jsb r7,hipkt    ;set up header and bufr control
2A3C 9AD0                bz hipk1c      ;hisp was bad
2A3E 4078 2B96          jsb r7,hi2tsv   ;will flush trnblk if last pkt
2A42 9A2D                bz hipblk      ;hi2tsk tests hitt
2A44 702C 4034          lda r2,hisp+foo(r4)
2A48 4078 1314          call oflush,whhi
2A4C 2000
2A4E 4818 0010          lda r1,=-pktnum&pktnum  ;incremental packet no.
2A52 321C 0026          addm r1,hipkth(r4)
2A56 4078 2C9E          jsb r7,hinbwt   ;wait for read to complete
2A5A 8A10                bnz hipslo     ;if timer went off
2A5C 4877               lda r7,r7       ;;hendf ;eom?
2A5E 9BC7                bm h2tx        ;end of message
2A60 9911                bo hiperr      ;;herrf ;ready line flap
2A62 707C 0026          lda r7,hipkth(r4)
2A66 4B78 00F0          and r7,=pktnum
2A6A 4E78 0070          cmp r7,=<-pktnum&pktnum>*7
2A6E 9115                be hiplng       ;branch if pkt no=7
2A70 4078 2C60          jsb r7,hinbuf   ;get a new buffer
2A74 8AE2                bnz himess     ;hitt not run out
2A76 E691               Trap 3221,<;Host blkd middle of 8-pkt - page 248>
2A78 9012               br hipblk
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 251
IMP.WARM;1      PAGE 11          *** warm page ***

                ;host error exits
                ;first group after message begun

2A7A 48A2       hipslo: lda r2,=cslows  ;cslows=2
2A7C 9014               br hisubc       ;mark mess as too slow

2A7E 48A5       hipbad: lda r2,=cimper  ;imp-detected i/o error
2A80 9012               br hisubc

2A82 701D 0006  hiperr: lda r1,statih(r5)       ;quit err?
2A86 4F18 0100          tst r1,=hquit
2A8A 9A03               bz .+6
                         Trap 3002,<;Host input err (Numerous traps call maint) - page 249>
2A8C E602
ge 249
2A8E 9002                br .+4
2A90 E6A0                 Trap 3240,<;Error during host input data - page 249>
2A92 4898               lda r1,=cerror  ;error during data
2A94 48A0               lda r2,=0
2A96 9008               br hisub2       ;fix type too

2A98 48A1       hiplng: lda r2,=clong   ;clong=1
2A9A 9005               br hisubc       ;mark mess as too long

2A9C 48F0       hipblk: set trpack(r3) = #0
2A9E 307B 000E
2AA2 48A4               lda r2,=cblock  ;cblock=4
2AA4 4899       hisubc: lda r1,=cinctr  ;incomplete transmission
2AA6 4078 5B0C  hisub2: jsb r7,hierc    ;error code to trnblk
2AAA 48F3               lda r7,=inmtyp  ;incomplete message
2AAC 387C 0027          stab r7,hipkth+1(r4)    ;to tell dest
2AB0 4868 02F8          lda sp,=lstack
2AB4 4078 2CD6          call hiset
2AB8 9A0F                bfail hisub3   ;bad hisp too}
2ABA 4865               lda r6,r5
2ABC 4078 2D54          jsb r7,hipkte   ;send just the header part (use r1)
2AC0 4078 2B9E          jsb r7,hi2tsk
2AC4 8A19                bnz hidisj     ;let discard flush hisp
2AC6 E692               Trap 3222,<;Task blkd inc msg - page 249>

                ;Enter here after error that lost a buffer from hisp,
                ; or fall in if Task refused an incomplete message
                ;Just releases the trnblk, so the message number timeout
                ; will clean up the message. Not much else we could do
                ; here except try to get a new buffer from the free list,
                ; but why bother?

2AC8 703C 402E  hiures: lda r3,hitran+foo(r4)   ;release the trnblk
2ACC 4878 FF7F          lda r7,=-1?ttresv       ;other bits are all set up
2AD0 3B7B 000D          andbm r7,trstat(r3)
2AD4 9011               br hidisj       ;forget rest of message
                ;error message is actually sent to host by inc reply code

2AD6 E6C8       hisub3: Trap 3310,<;Bad hisp for bad message - page 249>
2AD8 90F8               br hiures
```

```
                ;host error exits, before message started, or if no recovery

2ADA E7C4       hidie2: Trap 3704,<;dest died in hi - page 250>
2ADC 48A0       hiimpd: lda r2,=cimpd   ;dest IMP dead
2ADE 4EA1       hidead: cmp r2,=chstd   ;is it a dest host dead?
2AE0 9102                be .+4 ;yup, use status
2AE2 48F0                lda r7,=0      ;nope, give no status
2AE4 4897               lda r1,=cdestd
2AE6 307B 000A          sta r7,trdeds(r3)       ;save reason for host dead
2AEA 4078 5B0C  hi2h:   jsb r7,hierc
2AEE 703C 402E          lda r3,hitran+foo(r4)
2AF2 4078 188A          jsb r7,ledp0    ;put this trn blk on host
2AF6 4008 2692  hidisj: jmp hidisc

                ;Got a bad Host-IMP code

2AFA 48A2       hibadc: lda r2,=cillgl  ;cillgl=2
2AFC 4891       hierr:  lda r1,=cerrld
2AFE 4878 0F00          lda r7,=hicode
2B02 3073               sta r7,(r3)     ;;trnetl ;new format
2B04 90F3               br hi2h

2B06 48A4       hiblk:  lda r2,=cblock  ;host blocked
2B08 4899               lda r1,=cinctr  ;incomplete transmission
2B0A 90F0               br hi2h

                ;Got an error message from a host

2B0C EFC8       hiifei: Trap 7710,<;Host sent err w/id - page 250>
2B0E 90F4               br hidisj

                ;got a NOP from a host

2B10 701B 0008  hinop:  lda r1,trmidl(r3)       ;get desired padding
2B14 4B9F       hinop3: and r1,=subtyp          ;entry from old format code
2B16 4E99               cmp r1,=himaxp
2B18 8C02               bnl hinop2
2B1A 4899                lda r1,=himaxp ;truncate padding request to maximum
2B1C A291       hinop2: sll r1,1        ;;words
2B1E 4C18 0080          ior r1,= H80    ;indicate new format
2B22 381C 001C  hinop1: stab r1,homode(r4)
2B26 90E8               br hidisj

                ;check for old format NOP

2B28 4E18 0400  hinopt: cmp r1,=cnop_ H8        ;got a NOP?
2B2C 91FB                be hinop1      ;yes, take it
2B2E 4007               jmp (r7)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 253
IMP.WARM;1      PAGE 13          *** warm page ***

.comnt  |
hipok - sleep with immediate wakeup
hiwm - sleep, wait for event or clock tick.

        These routines are the HI sleep routines; hipok
sleeps after poking its own PID; while hiwm sleeps, and waits
either for a tick (hitt positive), or for packet arrival (hitt
negative).  Always returns with hitt in r7, and the cc set
to match r7.  Eventually, convert to stack routines using
per-host stacks.

Takes r4 set to host block pointer.
Returns r3: tran blk, r4: host blk, r5: io blk, r6:lstack
|
                routine hipok, nosave, arg r4, result r3-r6
2B30 781C 000F    ldab r1,hinpid(r4)    ;poke host input PID to shorten nap.
2B34 3098 00AC    sta r1,@pid

2B38 9001       entry hiwm, nosave, arg r4, result r3-r6
2B3A 307C 0004    sta r7,hilo(r4)       ;save return address.
                  repeat                ;dismiss until ready
2B3E 300C 4000      unlock lockhi(r4)   ;release process state
2B42 4008 1086      tr loopm            ;sleep via PID dispatch

                    hi:                 ;come here from loop.
2B46 7049 81C0      lda r4,mblks(r1)    ;host block pointer.
2B4A 705C 0020      lda r5,iobloc(r4)   ;io device pointer.
2B4E 707C 4000      lock lockhi(r4)     ;lock HI state
2B52 9AFE
2B54 702C 000C      lda r2,hitt(r4)     ;Host timer: minus => wait for hardware
2B58 8B06           while minus & statih(r5) .bit. =hbusy
2B5A 707D 0006
2B5E 4F78 2000
2B62 8AEE
                  endrepeat
2B64 703C 002E    lda r3,hitran(r4)     ;Transaction block address.
2B68 4872         lda r7,r2             ;set cc on hitt.
2B6A 408C 0004    jmp @hilo(r4)         ;return to caller.
2B6E 4807       endroutine      ;;hipok,hiwm
```

```
.comnt  |
hiwfe - wait forever for hardware to become idle.

        This routine sleeps, and returns only when
the host hardware busy bit, hbusy, becomes clear.  Uses
hiwm to sleep.

Takes r4 = host block pointer.
Returns r3: tran blk, r4: host blk, r5: io blk, r6:lstack
|

                routine hiwfe, nosave, arg r4, result r3-r6
2B70 307C 000A    sta r7,hisav7(r4)
                  repeat
2B74 4078 2B3A      call hiwm
2B78 707D 0006      while statih(r5) .bit. =hbusy
2B7C 4F78 2000
2B80 8AFA
                  endrepeat
2B82 707C 000A    lda r7,hisav7(r4)
2B86 4807       endroutine hiwfe
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 255
IMP.WARM;1      PAGE 15          *** warm page ***

                ;Host Input Subroutines

                ; give a packet to task and wait until task takes it
                ; call on r7 - pkt in r2 - clobbers all but r2,3,4
                ; returns r7 and its cc: 0-blocked by task, not zero-ok
                ; flushes buffer if blocked and called from a back host
                ;*** called from backs (gvtskc,d) and hi (gvtskm,hi2tsk)
                ;*** entry point hi2tsv (hi) saves trdstl via r3 in hisav7

2B88 4838 0078  gvtskc: lda r3,=sec3    ;backs wait little
2B8C 303C 000C          sta r3,hitt(r4) ;since no recovery
2B90 304E 0092  gvtskm: sta r4,inch(r6)
2B94 9005               br hi2tsk

2B96 703B 0006  hi2tsv: lda r3,trdstl(r3)       ;save dest for caller
2B9A 303C 000A          sta r3,hisav7(r4)
2B9E 307C 0008  hi2tsk: sta r7,htemp7(r4)
2BA2 302C 0006          sta r2,htemp(r4)
                  repeat
2BA6 4878 FFFC      set r7 = =-1?tskfok?tskfrf
2BAA 3B7C 001E      andbm r7,hibits(r4)
2BAE 4868 02F8      set sp = =lstack
2BB2 4078 146C      call wheorb,whtsk
2BB6 0001
2BB8 48F0           set chan-chain(r2) = =0     ;make source be host
2BBA 307A 0350
2BBE 4078 14AA      call ttskput
                    repeat                      ;wait for task to do its stuff
2BC2 4078 2B3A        call hiwm                 ;sleep for a while
2BC6 702C 0006        set r2 = htemp(r4)        ;get buffer back
2BCA 707A 06A0        set r7 = where-chain(r2)  ;check ownership
2BCE 4F78 2000        if r7 .nbit. =whhi        ;not ours any more}
2BD2 8A03
2BD4 E699               Trap 3231,<;Lost buffer in hi2tsk - page 253>
2BD6 9012               br fret ;;fail return
                      endif
2BD8 787C 001E        setb r7 = hibits(r4)      ;what did task do?
2BDC 9910             bodd ret  ;;tskfok        ;all ok, return success
2BDE 4FF2            until r7 .bit. =tskfrf     ;refused, try again
2BE0 9AF1
                    endrepeat
2BE2 707C 000C      set r7 = hitt(r4)           ;check timeout
2BE6 8AE0           until zero                  ;timed out, refuse host
                  endrepeat
2BE8 707C 0002    set r7 = hostyp(r4)   ;tell host what happened
2BEC 8B06         if minus              ; but if back, lose buffer
                    ;r6 left by hiwm
2BEE 4078 1332      call flush, whhi
2BF2 2000
2BF4 E69A           Trap 3232,<;Bk blkd awaiting task - page 253>
2BF6 9002          else                 ;reals and fakes just find out
2BF8 E68E           Trap 3216,<;Host blkd awaiting task - page 253>
                  endif
                  fret:
2BFA 48F0         lda r7,=0             ;actually return 0 in r7 (fail ret)
                  ret:
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 256
IMP.WARM;1      PAGE 16          *** warm page ***

2C00 702D 0006  hilede: lda r2,statih(r5)
2C04 4F28 0100          if r2 .bit. =hquit
2C08 9A02
2C0A E603                 Trap 3003,<;Host input err in leader - page 254>
                        endif
2C0C 48A0               lda r2,=cerr32  ;error in leader or padding
2C0E 4008 2AFC          tr hierr        ;report the error

2C12 48A1       hildi2: lda r2,=cshort  ;leader and padding is too short
2C14 90FD               tr hierr

                ;set up input parameters to point to leader storage area
                ;call with r3 pointing to first desired data word,
                ;  r2 pointing to last word, omit map
                ;return eom indicator as sign in r7
                ;  cc nz if too short, e if new format (usually).
                ;entry points: hiledi for initial read, hiledm to read
                ;  more leader or padding

2C16 701D 0004  hiledm: lda r1,endi(r5) ;;hendf
2C1A 9BFC                bm hildi2      ;if too short

2C1C 307C 0008  hiledi: sta r7,htemp7(r4)
2C20 302C 0006          sta r2,htemp(r4)
2C24 4813               lda r1,r3       ;;trnblk ;start leder input here
2C26 4B18 1FFE          and r1,=packm           ;now pack this place
2C2A 4921               sub r2,r1               ;length for input
2C2C A694               srl r1,4
2C2E 7418 00D2          ior r1,mapv2
2C32 4832               lda r3,r2               ;move arg to the right place
2C34 4868 02F8          lda sp,=lstack
2C38 707C 0002          lda r7,hostyp(r4)       ;input by host type
2C3C 40FF 12BE          call @hitable(r7)
2C40 4078 2B70          jsb r7,hiwfe
2C44 707D 0004          lda r7,endi(r5) ;;herrf
2C48 99DC                bo hilede      ;error during leader
2C4A 757C 0006          eor r7,htemp(r4)        ;check length
2C4E 7013               lda r1,(r3)     ;;trnetl
2C50 4B18 0F00          and r1,=hicode
2C54 4E18 0F00          cmp r1,=hicode  ;return cc
2C58 4F78 1FFE          tst r7,=packm   ;cc nz if too short, e if new format
2C5C 408C 0008          jmp @htemp7(r4)
```

```
.comnt  |
hinbuf - set up a new host input buffer.

        This routine gets a new buffer for host input, then
sets up the host interface to read into that buffer.  If it
can't get a buffer, it sleeps a tick, then tries again.
If that fails, it will trap, and return a zero condition code.

Takes r4: host blk.
Returns r3: tran blk, r4: host blk, r5: io blk
|

                routine hinbuf, arg r4, nosave
2C60 307C 0008    sta r7,htemp7(r4)     ;save return address
2C64 4868 02F8    lda sp,=lstack        ;stack needed by freget and hsioin
                  repeat                ;try to get host input buffer.
2C68 4078 13C6      call freget,whhi+nhi ;get buffer.
2C6C 200E
2C6E 8A04           until succeed       ;we got it}
2C70 4078 2B3A      call hiwm           ;sleep a tick
2C74 8AFA           until fail          ;zero => timed out.
                  endrepeat

2C76 8A03         if fail               ;fail => timed out
2C78 E68A           Trap 3212,<;Host blkd awaiting free buffer - page 255>
2C7A 900F         else                  ;got buffer.
2C7C 302C 0032      sta r2,hibf(r4)     ;save buffer chain word.
2C80 7078 00D2      setmap m2,mapv2     ;needed by hsioin
2C84 3078 FC04
2C88 705C 0020      lda r5,iobloc(r4)   ;device address for start input
2C8C 4078 15AA      call hsioin,data,bufend ;host start input
2C90 0010
2C92 008E
2C94 703C 002E      lda r3,hitran(r4)   ;set cc non-zero
                  endif
2C98 408C 0008    jmp @htemp7(r4)       ;return to caller.
2C9C 4807       endroutine hinbuf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 258
IMP.WARM;1      PAGE 18          *** warm page ***

                ;wait for input to finish
                ;return r2 hisp,r7 hiendi, cc nz means reset

2C9E 307C 0008  hinbwt: sta r7,htemp7(r4)
2CA2 4878 FDA8          lda r7,=-sec15  ;init hitt for hardware
2CA6 307C 000C          sta r7,hitt(r4)
2CAA 4078 2B30          jsb r7,hipok    ;wait for host
2CAE 4078 2CCC          jsb r7,hittgo   ;set software timer
2CB2 702C 4032          lda r2,hibf+foo(r4)
2CB6 302C 0034          sta r2,hisp(r4) ;returns hisp in r2
2CBA 707D 0004          lda r7,endi(r5) ;save hardware length in hstblk
2CBE 307C 0036          sta r7,hiendi(r4) ;returns endi in r7
2CC2 4898               lda r1,=hirset  ;set cc nz if we were reset
2CC4 7F1C 001E          tstb r1,hibits(r4)
2CC8 408C 0008          jmp @htemp7(r4)


2CCC 4818 0258  hittgo: lda r1,=sec15   ;normal time for a host
2CD0 301C 000C          sta r1,hitt(r4) ;now waiting for imp
2CD4 4007               jmp (r7)
```

```
                ;Set up the header of a packet
                ;Call: r7/return, r4/host, r3/trnblk
                ;Return: r1/=midh(r5), r2/hisp(r4), r5/buffer unpacked
                ;  r1 is arg to hipkte below for error case.
                ;  Bad buffer in hisp returns fail condition.

2CD6 1076       routine hiset, arg r3-r4, result r1-r2, result r5
2CD8 702C 4034    lda r2,hisp+foo(r4)
2CDC 4078 1500    call unpckc,whhi
2CE0 2000
2CE2 8A04         if fail
2CE4 6076           fail return
2CE6 4FF0
2CE8 4007         endif
2CEA 4851         lda r5,r1                     ;r1 will move through packet.
2CEC 7078 808C    set %map3 = m2#slfptr ;copy map2 to map3
2CF0 3078 FC06
2CF4 1076         save r7               ;we'll need this again
2CF6 7078 00D2    setmap m2,mapv2       ;so we can get to trnblk
2CFA 3078 FC04
2CFE 4A18 2000    add r1,#m3-m2         ;reference buffer through map3
2D02 48F0         lda r7,=0
2D04 2071         sta r7,(r1)+                  ;;neth = 0
2D06 707B 0002    lda r7,trtypl(r3)             ;flags
2D0A 2071         sta r7,(r1)+                  ;;typh
2D0C 48F0         lda r7,=0
2D0E 2071         sta r7,(r1)+                  ;;chkh=0
2D10 7078 62E2    lda r7,mine                   ;my imp number
2D14 2071         sta r7,(r1)+                  ;;srch
2D16 787B 0004    ldab r7,trhstl(r3)            ;mesnum
2D1A 2871         stab r7,(r1)+                 ;;seqh
2D1C 707C 0030    lda r7,hioldb(r4)             ;message block this message
2D20 787F 6937    ldab r7,tmctl+1(r7)           ; rmblk num
2D24 2871         stab r7,(r1)+                 ;;seqh+1
2D26 707C 0026    lda r7,hipkth(r4)             ;pkth set up
2D2A 2071         sta r7,(r1)+                  ;;pkth
2D2C 707B 0006    lda r7,trdstl(r3)             ;dest IMP
2D30 2071         sta r7,(r1)+                  ;;dsth
2D32 707B 0008    lda r7,trmidl(r3)             ;messid, subtyp
2D36 4B78 FFF0    and r7,=messid                ;subtyp must be 0
2D3A 3071         sta r7,(r1)                   ;;midh
2D3C 7078 00D0    setmap m3,mapvar      ;fix map3
2D40 3078 FC06
2D44 6076         restore r7            ;buffer map
2D46 3078 FC04    sta r7,%map2  ;restore it
2D4A 4918 2000    sub r1,#m3-m2         ;fix buffer pointer
2D4E 302C 0034    sta r2,hisp(r4)               ;restore hisp.
2D52 6006       endroutine hiset
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 260
IMP.WARM;1      PAGE 20          *** warm page ***

                ;Set up header and checksum packet
                ;Call Hipkt with args as for hiset (previous page)
                ; it calls hiset first thing
                ;Call hipkte with preferred endpointer in r1 to handle
                ; error cases to send inc msg to destination.
                ;Return:
                ; r2/ hisp(r4), r3/hitran(r4), r5/iobloc(r4), r4/host
                ;Bad hisp in hiset returns zero condition code.
                ;uses htemp7(r4) and temp2
                ;return with mapv2 in %map2 in all paths

                ;Special entry with r1/ =midh(r6) for error case
                ;r2 is not used by this entry point

2D54 307C 0008  hipkte: sta r7,htemp7(r4)
2D58 900F               br hipk1

                ;Special entry with r1/ haccom word for raw packets
2D5A 307C 0008  hipktr: sta r7,htemp7(r4)
2D5E 9013               br hipk2

2D60 307C 0008  hipkt:  sta r7,htemp7(r4)
2D64 4868 02F8          lda sp,=lstack
2D68 4078 2CD6          call hiset      ;set up header first
2D6C 9A2B                bfail hipk3    ;hisp failure
2D6E 4865               lda r6,r5
2D70 701C 0036          lda r1,hiendi(r4)       ;and its end pointer.
2D74 8B08                bnm hipk2      ;;hendf
2D76 4878 4000  hipk1:  lda r7,=lstpkt  ;At end: put in last packet bit
2D7A 347E 000A          iorm r7,pkth(r6)
2D7E 48F4               lda r7,=tskfre  ;tell task to release trnblk
2D80 3C7C 001E          iorbm r7,hibits(r4)
2D84 4916       hipk2:  sub r1,r6
2D86 4B18 1FFE          and r1,=packm
2D8A 301E 0090          sta r1,bufe(r6)         ;proper hardware end
2D8E 4878 008C          lda r7,=bufend-hrdoff
2D92 4971               sub r7,r1
2D94 9B19               bm hipker
2D96 4816               lda r1,r6
2D98 4868 02F8          lda sp,=lstack
2D9C 4078 17D4          call pcksubr            ;length in BUFE already
2DA0 3039 0004          sta r3,chkh(r1)
2DA4 703C 002E          lda r3,hitran(r4)
2DA8 70F8 62E8          lda r7,@clock
2DAC 3079 0096          sta r7,it(r1)
2DB0 3049 0092          sta r4,inch(r1)
2DB4 7078 00D2          setmap m2,mapv2
2DB8 3078 FC04
2DBC 4861               lda r6,r1
2DBE 705C 0020          lda r5,iobloc(r4)       ;set condition code non-zero
2DC2 408C 0008  hipk3:  jmp @htemp7(r4)

2DC6 E6F0       hipker: Trap 3360,<:HI bad packet length - page 258>
2DC8 7018 00D2          lda r1,mapv2
2DCC 3018 FC04          sta r1,%map2
2DD0 4008 2A7E          jmp hipbad      ;go inform our HOST
```

```
                ;Get a message number for an input message, a request for 8,
                ;or an incomplete message.
                ;return r1: mess num,local tm block (trhstl),
                ;       r2: foreign use (pkth), r7: local use (trluse).

2DD4 307C 000A  mesget: sta r7,hisav7(r4)
2DD8 705C 0030  mg00:   lda r5,hioldb(r4)
2DDC 4078 2E42          jsb r7,mgtest   ;check last used tm blk
2DE0 4858 0380          lda r5,=tmnum*tmlen
2DE4 9006               br mg03

2DE6 767D 6934  mg02:   cmp r7,tmhost(r5)
2DEA 8105               bne mg04
2DEC 4078 2E42          jsb r7,mgtest   ;check all current tm blks
2DF0 707C 0028  mg03:   lda r7,hihost(r4)
2DF4 4958 0010  mg04:   sub r5,=tmlen
2DF8 8BF7                bnm mg02
2DFA 4858 0370          lda r5,=<tmnum-1>*tmlen
2DFE 707D 6932  mg05:   lda r7,tmimp(r5)
2E02 8B0F                bnm mg06
2E04 706D A930          lda r6,tmlock(r5)
2E08 9AFE                 bz .-4
2E0A 707D 6932          lda r7,tmimp(r5)
2E0E 8B07                bnm mg08
2E10 707D 6938          lda r7,tmmess(r5)
2E14 4FFC               tst r7,=age0* HC
     0001       .if nz LBig                     ;must jump to common
2E16 9A03                tnz mg07       ;found a free tm blk
2E18 4008 59E6
                .iff
                         bnz mg07       ;found a free tm blk
                .endc
2E1C 300D A930  mg08:   sta r0,tmlock(r5)
2E20 4958 0010  mg06:   sub r5,=tmlen
2E24 8BED                bnm mg05
2E26 702B 0006  mg09:   lda r2,trdstl(r3)       ;check dest imp state
2E2A A2A1               sll r2,1
2E2C 706A 72FC          lda r6,spfrut(r2)       ;reachable?
2E30 8B03               if minus        ;;spfded
2E32 4008 2ADA            jmp hidie2            ;destination dead
                        endif
2E36 4078 2B3A          call hiwm
2E3A 8ACF                bnz mg00       ;no timeout yet
2E3C E68C               Trap 3214,<;Host blkd awaiting mes num or blk - page 259>
2E3E 4008 2B06  mg01:   tr hiblk
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 262
IMP.WARM;1      PAGE 22          *** warm page ***


FC00 0400                page Warm                ;rest is low-duty path

59C6 4078 1314  mg15:    call oflush,whhi         ;delete this when it's a back host
59CA 2000
59CC 705C 0030           lda r5,hioldb(r4)        ;release this tm blk
59D0 706D A930           lda r6,tmlock(r5)
59D4 9AFE                 bz .-4
59D6 4878 8000           lda r7,=sign
59DA 347D 6932           iorm r7,tmimp(r5)
59DE 300D A930           sta r0,tmlock(r5)
59E2 4008 2B06           jmp hiblk
```

```
                ;found a free tm blk

59E6 305C 0030  mg07:   sta r5,hioldb(r4)
59EA 706B 0006          lda r6,trdstl(r3)
59EE 306D 6932          sta r6,tmimp(r5)
59F2 706C 0028          lda r6,hihost(r4)
59F6 306D 6934          sta r6,tmhost(r5)
59FA 706C 002A          lda r6,hihand(r4)
59FE 306D 6936          sta r6,tmctl(r5)
5A02 706D 6938          lda r6,tmmess(r5)
5A06 4A68 0010          add r6,=luse0   ;new use of this blk
5A0A 4B68 00F0  mg21:   and r6,=luse+<0*age>+<0*messno>
5A0E 306D 6938          sta r6,tmmess(r5)
5A12 4868 40FF          lda r6,=tminit+ HFF
5A16 306D 693A          sta r6,tstate(r5)
5A1A 786C 001D          ldab r6,holhn(r4)       ;which local host I am
5A1E 306D 693E          sta r6,tmstp(r5)        ;;tmstop.tmalt,.tmlhn
5A22 300D A930          sta r0,tmlock(r5)
5A26 4078 39B0          jsb r7,getfre   ;watch htemp7
5A2A 4878 5000          lda r7,=qertyp+pribit   ;for typh
5A2E 307E 0002          sta r7,typh(r6)
5A32 707C 0030          lda r7,hioldb(r4)
5A36 703F 6932          lda r3,tmimp(r7)        ;destination IMP number
5A3A 9B29                bm mg23                ;dest IMP died
5A3C 303E 000C          sta r3,dsth(r6)         ;where to reply to
5A40 7038 62E2          lda r3,mine             ;my IMP number
5A44 303E 0006          sta r3,srch(r6)         ;into buffer
5A48 783F 6939          ldab r3,tmmess+1(r7)    ;;luse, age
5A4C A6B4               srl r3,4        ;position luse to low-order
5A4E 7C3F 6936          iorb r3,tmctl(r7)       ;handling type
5A52 383E 000E          stab r3,midh(r6)        ;midh left half
5A56 A6F4               srl r7,4        ;block num to right
5A58 387E 000F          stab r7,midh+1(r6)      ;midh right half
5A5C 48B9               lda r3,=gabtyp  ;get-a-block
5A5E 303E 000A          sta r3,pkth(r6) ;pkt cod in pkth
5A62 4078 5A96          jsb r7,fndhac   ;find the hac words
5A66 301E 0010          sta r1,data(r6)
5A6A 701D 40D0          lda r1,haccom(r5)
5A6E 301E 0012          sta r1,data+2(r6)
5A72 4816               lda r1,r6
5A74 4838 0014          lda r3,=<data+4>
5A78 4868 02F8          lda sp,=lstack
5A7C 4078 17C8          call lpcksubr   ;calculate checksum
5A80 3039 0004          sta r3,chkh(r1) ;and stick it in packet
5A84 4861               lda r6,r1
5A86 4078 2B90          jsb r7,gvtskm   ;now put it into store-and-forward
5A8A 9A9E                bz mg15        ;timed out, give up tm blk
5A8C 4078 131A  mg23:   call oflshb.whhi        ;give it up, tsk has it
5A90 2000
5A92 4008 2DD8          tr mg00
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 264
IMP.WARM;1      PAGE 24          *** warm page ***

                        page HLCode

                ;subroutine to test transmit msg block
                ;return if fail, go to main line via hisav7

2E42 4F58 FC0F  mgtest: tst r5,= HFC0F  ;;rmnum tmnum rmlen tmlen ;*$
2E46 9A03               bz .+6          ;*$
2E48 E611                Trap 3021,<;illegal message blk in hi - page 262>
2E4A 901C                br mtest2      ;*$
2E4C 706D A930          lda r6,tmlock(r5)
2E50 9AFE                bz .-4
2E52 706B 0006          lda r6,trdst1(r3)       ;dest IMP #
2E56 766D 6932          cmp r6,tmimp(r5)
2E5A 8112                bne mtest1
2E5C 706C 0028          lda r6,hihost(r4)       ;dest,,srce host #s
2E60 766D 6934          cmp r6,tmhost(r5)
2E64 810D                bne mtest1
2E66 786C 001D          ldab r6,holhn(r4)       ;my source host #
2E6A 7E6D 693F          cmpb r6,tmlhn(r5)
2E6E 8108                bne mtest1     ;different IMP in me
2E70 706C 002A          lda r6,hihand(r4)       ;handling type
2E74 756D 6936          eor r6,tmctl(r5)
2E78 4B68 F000          and r6,=gethan
2E7C 9A04                bz mg10        ;this is the right block
2E7E 300D A930  mtest1: sta r0,tmlock(r5)
2E82 4007       mtest2: jmp (r7)


2E84 305C 0030  mg10:   sta r5,hioldb(r4)
2E88 707D 693A          lda r7,tstate(r5)
2E8C 890A                bno mg11       ;still busy
2E8E A3F1               rll r7,1        ;interesting bits to odd, sign
2E90 992C                bo mg22        ;;tmrset ;block was reset
2E92 8B0A                bnm mg12       ;;tminit ;block not being init
2E94 706D 6938          lda r6,tmmess(r5)
2E98 4FEC               tst r6,=age0* HC
2E9A 9A03                tnz mg21       ;resend get-a-block
2E9C 4008 5A0A
2EA0 300D A930  mg11:   sta r0,tmlock(r5)        ;wait, then try it again
2EA4 90C1               tr mg09
```

```
                ;more of mgtest subr

2EA6 707D 693E  mg12:   lda r7,tmstp(r5)        ;;tmstop
2EAA 9BFB               bm mg11         ;if stop bit is on wait
2EAC 781D 693B          ldab r1,tstate+1(r5)
2EB0 A691               srl r1,1
2EB2 381D 693B          stab r1,tstate+1(r5)
2EB6 4893               lda r1,=msto_- H8       ;set inc q timer
2EB8 3C1D 693A          iorbm r1,tstate(r5)
2EBC 701D 6938          lda r1,tmmess(r5)
2EC0 4879 0100          lda r7,=mess0(r1)       ;incr mess num
2EC4 4FFC               tst r7,=age0* HC        ;reduce age to 4 (or nop)
2EC6 9A03               bz mg13         ;still young
2EC8 4CFF                ior r7,=age
2ECA 49FB                sub r7,=age-<age0*4>
2ECC 307D 6938  mg13:   sta r7,tmmess(r5)
2ED0 702D 6936          lda r2,tmctl(r5)
2ED4 300D A930          sta r0,tmlock(r5)
2ED8 4B28 0F00          and r2,=fuse    ;foreign use for pkth
2EDC A6D4               srl r5,4
2EDE 4B18 FF00          and r1,=messno
2EE2 4C15               ior r1,r5       ;local tm block #
2EE4 408C 000A          jmp @hisav7(r4)

2EE8 8BDC       mg22:    bnm mg11       ;block was being reset: wait
2EEA 707D 693C          lda r7,tmdeds(r5)       ;dead status
2EEE 782D 6938          ldab r2,tmmess(r5)      ;and subtype
2EF2 300D A930          sta r0,tmlock(r5)
2EF6 4008 2ADE          jmp hidead      ;send reply(ies)

FC00 0400       page Warm

                ;set up hac stuff for get-a-block and raw packets
                ;returns r1:hacmem, r5:offset to hacs

5A96 701C 0028  fndhac: lda r1,hihost(r4)       ;both callers need this
5A9A 301E 0008          sta r1,seqh(r6)
5A9E 781C 001D          ldab r1,holhn(r4)
5AA2 4998               sub r1,=nfh*words       ;use real number, minus if fake
5AA4 A398               rll r1, H8      ;swap bytes
5AA6 7418 62E2          ior r1,mine     ;low bits FF if Fake
5AAA 4858 0010          lda r5,=lhaccs  ;how many specials
5AAE 561D 40E0          cmp r1,hacspc(-r5)
5AB2 9102                be .+4         ;got a match
5AB4 88FD                bnlp .-6       ;look more (first is default)
5AB6 701D 40C0          lda r1,hacmem(r5)
5ABA 4007               jmp (r7)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 266
IMP.WARM;1      PAGE 26          *** warm page ***

               ;allocate a free transaction block
               ;in: r4   out:r4 host,r3 pntr to block,r5 unchanged,others smash

5ABC 4A38 0010 trnpt2: add r3,=trnl    ;next slot
5AC0 4E38 8F50         cmp r3,=trnblk+trntot
5AC4 811A              bne trnpt1      ;back for more
5AC6 3008 A4A8         sta r0,trnlok
5ACA 307C 0008         sta r7,htemp7(r4)
5ACE 4078 2B3A         jsb r7,hiwm
5AD2 8A06               bnz trnpt4     ;if hitt=0 we are blocked
5AD4 E690              Trap 3220,<;Host blkd awaiting trnblk - page 264>
5AD6 4878 0258         lda r7,=sec15   ;take another shot at it
5ADA 307C 000C         sta r7,hitt(r4)
5ADE 707C 0008 trnpt4: lda r7,htemp7(r4)

5AE2 4838 8910 trnput: lda r3,=trnblk
5AE6 701C 0002         lda r1,hostyp(r4)       ;;VDH ;fake?
5AEA 4E92              if r1 <> #hosfake       ;no, use first two slots
5AEC 9103
5AEE 4A38 0020           add r3,=2*trnl        ;else reserve 2 slots for DDT
                       endif
5AF2 7018 A4A8         lda r1,trnlok
5AF6 9AFE               bz .-4

5AF8 781B 000D trnpt1: ldab r1,trstat(r3)
5AFC 8AE0               bnz trnpt2     ;this trn blk busy
5AFE 4818 4080         lda r1,= H4000+ttresv   ;block now belongs to HI
5B02 301B 000D         sta r1,trstat+<0*trntim>(r3)
5B06 3008 A4A8         sta r0,trnlok
5B0A 4007              jmp (r7)
```

```
                ;Give our Host a 2-wd message

                ;Enter at hierc with r1/reply type, r2/dead host status

5B0C 752B 0008  hierc:  eor r2,trmidl(r3)       ;save subtype in midl leader
5B10 4BAF               and r2,=subtyp
5B12 352B 0008          eorm r2,trmidl(r3)
5B16 381B 0003          stab r1,trtypl+1(r3)    ;save reply type in leader
5B1A 4818 0078          lda r1,=sec3
5B1E 301C 000C          sta r1,hitt(r4) ;give us a little time
5B22 4007               jmp (r7)

                ;Get our HOST an allocate
                ;skip return on failure., tstate in r1

                page HLCode

2EFA 705C 0030  tallyg: lda r5,hioldb(r4)
2EFE 701D A930          lda r1,tmlock(r5)
2F02 9AFE               bz .-4
2F04 701D 693A          lda r1,tstate(r5)
2F08 4F18 3C00          tst r1,=tmall
2F0C 9A06               ifnot zero      ;found an allocate
2F0E 4918 0400           sub r1,=tmall0
2F12 301D 693A           sta r1,tstate(r5)
2F16 49F2                sub r7,=2
                        endif
2F18 7018 12EA          lda r1,lngimp
2F1C 9B04               if minus } r1 = mine    ;this IMP get extra allocate?
2F1E 7618 62E2
2F22 8104
2F24 7018 12E8           lda r1,lngall  ;yes, get its value
2F28 9002               else            ;no, use default
2F2A 489A                lda r1,=iniall
                        endif
2F2C 7D1D 693E          eorb r1,tmstp(r5)       ;set timer field
2F30 4B18 003F          and r1,=tmalt
2F34 3D1D 693E          eorbm r1,tmstp(r5)
2F38 300D A930          sta r0,tmlock(r5)
2F3C 400F 0002          jmp 2(r7)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 268
IMP.WARM;1      PAGE 28          *** warm page ***

.comnt |
hothru - accumulate host throughput.

        This routine is called both by IH and HI, to add in
the throughput contribution of a given message to the throughput
totals for a given host.

        Call with:
        ldab r1,pkth+1  ;number of packets in this message.
        lda r3,<number of words>
        lda r4,<host blk>
        lda r5,<source/destination IMP word>
        call hothru
         htpmfn/htpmtn  ;for IH and HI respectively.

|

FC00 0400               Page Warm

                routine hothru, local r1-r2, inline r2, local r4-r5, arg r1, arg r3-r5
5B24 1076
5B26 1016
5B28 1026
5B2A 1046
5B2C 1056
5B2E 6027
5B30 307E 0008
5B34 4A42         add r4,r2                     ;pointer to throughput stats
5B36 A694         srl r1,4                      ;;pktnum
5B38 4A91         add r1,=1                     ;0-7 goes to 1-8
5B3A 323C 0010    addm r3,htpwti-htpmtn(r4)     ;;htpwfi
5B3E A2D1         sll r5,1
5B40 705D 72FC    lda r5,spfrut(r5)             ;for us test
5B44 4B58 800F    and r5,=routef+spfded
5B48 8A02         if zero                       ;is this from/to us?
5B4A 4AC8           add r4,=htppfl-htppfn       ;select "to/from local"
                  endif
5B4C 321C 0004    addm r1,htpptn-htpmtn(r4)     ;count packets.
5B50 8403         if carry                      ;overflow.
5B52 311C 0004      subm r1,htpptn-htpmtn(r4)
                  endif
5B56 4891         lda r1,=1                     ;count messages.
5B58 3214         addm r1,(r4)
5B5A 8402         if carry                      ;overflow.
5B5C 3114           subm r1,(r4)
                  endif
5B5E 6056       endroutine hothru
5B60 6046
5B62 6026
5B64 6016
5B66 6006
```

```
                ;*** warm ih ***

                ;leader send

                routine ihlsn, nosave
    0001        .IF NZ VHA
5B68 3068 02F6    sta r6,lstack-words           ;"push" old R6
5B6C 4868 02F6    lda sp,=lstack-words
5B70 1076         save r7
5B72 4078 3E26    call ihvha                    ;change leader in place
5B76 6076         restore r7
5B78 6066         pop r6
                .ENDC
                ihcuml: ;cum stats hook.
5B7A 781C 001C          ldab r1,homode(r4)      ;decide on n/o format
5B7E 8A2B               if zero
                          ;convert to old format
5B80 703C 0018            lda r3,ihledr+midl(r4)
5B84 781C 0013            ldab r1,ihledr+typl+1(r4)     ;message type
5B88 8A07                 if zero
5B8A 4823                   lda r2,r3
5B8C 4BAF                   and r2,=subtyp
5B8E 4EA3                   if r2 = =rawpkt
5B90 8103
5B92 4DB3                     eor r3,=rawpkt    ;clear subtyp field
5B94 4893                     lda r1,=rawpkt
                            endif
                          endif
5B96 A298                 sll r1, D8    ;;ihcode
5B98 702C 0012            lda r2,ihledr+typl(r4)
5B9C 4B28 0C00            and r2,=trcbit}octbit
5BA0 A3A2                 rll r2,2
5BA2 4C12                 ior r1,r2
5BA4 702C 0014            lda r2,ihledr+hstl(r4)
5BA8 4B28 8000            and r2,=priled
5BAC 4C12                 ior r1,r2     ;;getpri
5BAE 782C 0015            ldab r2,ihledr+hstl+1(r4)
5BB2 4E28 00FC            if r2 >= = H100-nfh
5BB6 9202
5BB8 4AA4                   add r2,=nfh ;;frmimp
                          endif
5BBA A2A6                 sll r2,6
5BBC 4C12                 ior r1,r2     ;;srceh
5BBE 741C 0016            ior r1,ihledr+dstl(r4)        ;;srcei
5BC2 4B68 8000            and r6,=hendf         ;preserve eom bit
5BC6 4C6C 0012            ior r6,=ihledr+2(r4)
                ;(OVER)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 270
IMP.WARM;1      PAGE 30          *** warm page ***

5BCA 9001       entry ihls, nosave
5BCC 301C 0010            sta r1,ihledr(r4)
5BD0 303C 0012            sta r3,ihledr+2(r4)
                        endif
5BD4 705C 0020          lda r5,iobloc(r4)
5BD8 483C 0010          lda r3,=ihledr(r4)
5BDC 4B38 1FFE          and r3,=packm
5BE0 A6B4               srl r3,4
5BE2 7438 00D0          ior r3,mapvar
5BE6 303D 0008          sta r3,starto(r5)
5BEA 4816               lda r1,r6               ;end pointer
5BEC 4868 02F8          lda sp,#lstack          ;get a stack
5BF0 1076               push r7                 ;save our caller
5BF2 482C 0010          lda r2,=ihledr(r4) ;;VDH ;watch H2V hack
5BF6 707C 0002          lda r7,hostyp(r4)       ;now, what type host?
5BFA 40FF 12CA          call @hotable(r7)       ;call the right handler
5BFE 301D 000A          sta r1,endo(r5)         ;finally, start output
5C02 6076               pop r7                  ;restore caller

5C04 4807       endroutine ihlsn/ihls



FC00 0400               Page Warm

5C06 1076       routine ihdump, arg r5, uses r1-r3
                  repeat
5C08 4815           lda r1,r5           ;head of queue.
5C0A 4078 154A      call deque,whih     ;get next thing from queue
5C0E 0040
5C10 9A04           until fail          ;nothing left.

5C12 4078 5C1A      call ihdmpb ;dump this buffer
5C16 90F9         endrepeat
5C18 6006       endroutine ihdump

5C1A 1076       routine ihdmpb,arg r1-r2/m2
5C1C 48F3         lda r7,=inmtyp        ;incomplete message.
5C1E 3479 000A    iorm r7,pkth(r1)
5C22 7078 00D2    lda r7,mapv2
5C26 3078 FC04    sta r7,%map2
5C2A 7078 B26C    lock lockih+fakeh3
5C2E 9AFE
5C30 30A8 7272      sta r2,@ehq+fakeh3  ;queue packet on discard.
5C34 3028 7272      sta r2,ehq+fakeh3
5C38 3008 B26C    unlock lockih+fakeh3
5C3C 6006       endroutine ihdmpb
```

```
                        page HLCode
                    ;get a leader for a host.
                    ;copy leader words into the host
                    ;skip 2 if there are none.

2F40 782C 005E  ledget: ldab r2,nxtled(r4)      ;get next leader, if any
2F44 9A11                bz ledg6       ;none to be found
2F46 49A1               sub r2,=1
2F48 786C 405F          ldab r6,nxtled+1+foo(r4)        ;get just trnblk offset
2F4C A2E4               sll r6,4
2F4E 4836               lda r3,r6       ;and init limit for search
2F50 764E 891E  ledg3:  cmp r4,trhost+trnblk(r6)        ;right for this leader?
2F54 910B                be ledg1       ;yes, a winner
2F56 4968 0010          sub r6,=trnl    ;to next trn blk
2F5A 8B03                bnm ledg4      ;not too small
2F5C 4868 0630          lda r6,=trntot-trnl     ;last in table
2F60 4E36       ledg4:  cmp r3,r6       ;done search?
2F62 81F7                bne ledg3      ;no, look more
2F64 E6D8               Trap 3330,<;ih lost a trnblk - page 269>
2F66 4AF2       ledg6:  add r7,=2       ;and fail return
2F68 901F               br ledg8

                lgcum:  ;cumstats hook.
2F6A 481C 0010  ledg1:  lda r1,=ihledr(r4)      ;copy leader to host blk
2F6E 603E 8910  ledg0:  lda r3,trnetl+trnblk(r6)+
2F72 2031               sta r3,(r1)+
2F74 4E1C 001A          cmp r1,=ihledr+midl+2(r4)
2F78 81FB                bne ledg0
2F7A 49EA               sub r6,=midl+2
2F7C 781E 8913          ldab r1,trtypl+trnblk+1(r6)     ;what is reply
2F80 4E97               cmp r1,=cdestd  ;dest dead?
2F82 810D                bne ledg5      ;nope, no bother
2F84 701E 891A          lda r1,trnblk+trdeds(r6)        ;any reason?
2F88 9A0A                bz ledg5       ;no, proceed normally
2F8A 301E 8918          sta r1,trmidl+trnblk(r6)        ;host status in messid/subtyp
2F8E 4896               lda r1,=chstat  ;host dead status
2F90 381E 8913          stab r1,trtypl+trnblk+1(r6)     ;new state
2F94 4890               lda r1,=0
2F96 381E 8914          stab r1,trhstl+trnblk(r6)       ;clear these bits for host
2F9A 9008               br ledg7        ;no change count

2F9C 4890       ledg5:  lda r1,=0
2F9E 301E 891E          sta r1,trhost+trnblk(r6)
2FA2 301E 891C          sta r1,trntim+<0*trstat>+trnblk(r6)
2FA6 382C 005E  ledg8:  stab r2,nxtled(r4)
2FAA 4007       ledg7:  jmp (r7)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 272
IMP.WARM;1      PAGE 32          *** warm page ***

.comnt |
********** forus **********

process packets for us
enter from task at forus.

registers: r2/ pkth to begin with, then gets bits:
     mesnxt - this mes next to go
     mesout - mes out of range
     meseen - this mes in last 8
     hostng - host is running(0) or not(1)
   r4/host param block
   r5/message block
   r6/packet buffer pointer
   r1,3,7/various

enter with r2/chain pointer,r6/buffer pointer
task variables in local so not locked
|

2FAC 3028 027E  forus:  sta r2,tskbuf   ;won't be needing it
2FB0 701E 0002          lda r1,typh(r6) ;check for raw packets
2FB4 4B18 C000          and r1,=paktyp
2FB8 8A08                bnz notraw
2FBA 701E 000E          lda r1,midh(r6)
2FBE 4B9F               and r1,=subtyp
2FC0 4E93               cmp r1,=rawpkt
2FC2 8103                te gudraw
2FC4 4008 33A6
2FC8 702E 000A  notraw: lda r2,pkth(r6)

                ;check transmit or receive message block

2FCC 48D0               lda r5,=0
2FCE 4FA4               tst r2,=trnrep
2FD0 9A03               ifnot zero              ;a reply, use tmblks
2FD2 4858 0380            lda r5,=tmblks-rmblks
                        endif
2FD6 787E 0009          ldab r7,fblock(r6)      ;my block
2FDA 4E78 0038          cmp r7,=rmnum   ;;tmnum
2FDE 9203                tng blker2     ;block number too high
2FE0 4008 30EC
2FE4 A2F4               sll r7,4        ;;rmlen tmlen ;* times 16=blk length
2FE6 4A57               add r5,r7
2FE8 707D A5B0          lock rmlock(r5) ;;tmlock ;lock the block
2FEC 9AFE
2FEE 707D 65B2          lda r7,rmimp(r5)        ;;tmimp
2FF2 767E 0006          cmp r7,srch(r6)
2FF6 8179                bne blkerr     ;wrong sender
2FF8 707D 65B8          lda r7,rmmess(r5)       ;;tmmess ;get my use number
2FFC A2F4               sll r7,4        ;position in foreign use position
2FFE 4B78 0F00          and r7,=fuse
3002 4D27               eor r2,r7       ;incoming use number
3004 4F28 0F00          tst r2,=fuse    ;match?
3008 8A70                bnz blkerr     ;wrong use
```

```
                ;got message block. set up host status

300A 784D 65B5          ldab r4,rmhost+1(r5)    ;local host
300E 4818 FFFF          lda r1,=-1      ;assume hac ok. so r3 is irrelevant
3012 4078 36EE          jsb r7,hostnm

                ;Check message number
                ;range (want - got):
                ; < -7    out of range high
                ; -1 - -7 (no bit) in next 7 (legal trans)
                ; 0        next to go (legal trans or inc trans)
                ; 1 - 8    seen in last 8 (legal reply or inc trans)
                ; 9+       out of range low

3016 4C28 0800          ior r2,=mesnxt  ;assume have next to go
301A 703D 65B8          lda r3,rmmess(r5)
301E 4C38 00FF          ior r3,= HFF
3022 713E 0008          sub r3,seqh(r6) ;one we got
3026 A4B8               sra r3, D8
3028 9A0B                bz forus4      ;we got next to go
302A 4D28 0C00          eor r2,=mesnxt+mesout   ;assume out
302E 487B 0007          lda r7,=7(r3)
3032 9B06                bm forus4      ;out of range high
3034 4EFF               cmp r7,= HF
3036 9C04                bl forus4      ;out of range low
3038 A6F2               srl r7,2
303A 752F 36A6          eor r2,mestst(r7)       ;fix for seen, out

                ;Set up for trn dispatch
                ;r5/my block, r3/messno offset

303E 3028 0284  forus4: sta r2,tskbts   ;remember state bits
3042 701E 000E          lda r1,midh(r6) ;his block
3046 4872               lda r7,r2       ;get copy of pkth
3048 4BFF               and r7,=pktcod  ;just its type
304A A2F1               sll r7,1        ;words-bytes
304C 408F 3050          jmp @fordis(r7) ;dispatch on message type

3050 3454       fordis: fmess,freq,fgvb,fincm   ;regular, request, giveback, incomplete
3052 33CE
3054 3368
3056 337E
3058 31AE               frfnm, frfal, fdead, fincr      ;RFNM, RFNM/alloc, dead RFNM,
305A 3108
305C 3192
305E 31D4
                         ;inc RFNM
3060 3070               fincq, fgetb, freset, fbad      ;inc query, get blk, reset blk,
3062 329C
3064 5C3E
3066 3298
                         ; (nothing)
3068 3182               foutof, fgetr, fresrq, fresrp   ;out of range, get blk reply,
306A 5C74
306C 3192
306E 5C60
                         ;reset request, reset reply
```

```
                ;Got an incomplete query.
                ;Try to recover as well as possible
                ;a5/my block


3070 4078 379A  fincq:  jsb r7,regch2    ;some reg mess checks
3074 4F28 0400          tst r2,=mesout   ;totally out?
3078 8A35               bnz sndout       ;yes, send out-of-range
307A 4F28 0200          tst r2,=meseen   ;in last 8?
307E 8A12               bnz chkdup       ;yes, see if we can send dupl reply
3080 4F28 0800          tst r2,=mesnxt   ;incq on next to give host?
3084 9A2F               bz sndout        ;no, too far in future.
3086 E7C8               Trap 3710,<;Sending incomplete reply - page 272>
3088 4F92               tst r1,=2
308A 9A03               tnz inctr0       ;in msg state
308C 4008 3390
3090 707E 0002          lda r7,typh(r6)
3094 4F78 0400          tst r7,=alloc    ;for multi packet case?
3098 8A03               tz incm1         ;no
309A 4008 33A0
309E 4008 3386          tr inctr1

30A2 702D 65BC  chkdup:           lda r2,rmtype(r5)
30A6 A623               srl r2,r3
30A8 4BA3               and r2,=3                ;nominal message type
30AA 4E90               if r1 = =0
30AC 8113
30AE E7C5                 Trap 3705,<;Sending duplicate reply - page 272>
30B0 4AA4                 add r2,=trnrep         ;fix for how reptab works
30B2 48B3                 lda r3,=3              ;assume hihd=3
30B4 4EC0                 ifnot r4 < =0 } deadsc(r4)
30B6 9208
30B8 707C 002C
30BC 8A05
30BE 783C 001F              ldab r3,hihd(r4)     ;get hihd now
30C2 4C38 FFE0              ior r3,= HFFE0       ;time=unknown
                          endif
30C6 303E 0010            sta r3,data(r6)        ;put in dead status
30CA 48B0                 lda r3,=0              ;0 subtyp so not raw
30CC 303E 0002            sta r3,typh(r6)        ;qer type -> reg type
30D0 900A               else
30D2 4E93                 if r1 = =3 } r1 = =2 & r2 = =0
30D4 9105
30D6 4E92
30D8 8105
30DA 4EA0
30DC 8103
30DE 4008 3270              tr fqok              ;things will progress, ignore.
                                        ;msg is in reply state or is current msg.
                          endif
                          sndout:
30E2 E7C1                 Trap 3701,<;Sending out-of-range - page 272>
                        endif
30E4 4008 31F2          tr replyx
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 275
IMP.WARM;1      PAGE 35          *** warm page ***

                ;here if no message block
                ;getblks are legal sometimes
                ;a5/my block

30E8 300D A5B0  blkerr: sta r0,rmlock(r5)
30EC 4BAF       blker2: and r2,=pktcod  ;what type packet?
30EE 4EA9               cmp r2,=gabtyp  ;a get-a-block?
30F0 8103               bne blker5      ;not a getblk
30F2 4008 32A0          jmp blkget      ;it's a get-a-block

30F6 4878 4500  blker5: lda r7,= H4500  ;bits 8 (incq),10 (reset b),14 (reset r)
30FA A672               srl r7,r2       ;get one of these bits?
30FC 9903                bo blker6      ;yes, try to recover
                ;       Trap 3100,<;Block error, no recovery - page 273>
30FE 4008 326E  fqokju: jmp fqoku

3102 701E 000E  blker6: lda r1,midh(r6) ;foreign IMP block #
                          ; Trap 3101,<;Block error, trying recovery - page 273>
3106 907A               br replyu       ;send out-of-range, reset reply, reset

                ;Got a allocate (+ implicit RFNM if multi-packet)
                ;here with r3/no. we want-no. we got, r2/flags,
                ;r4/host, r5/tmblk, r6/buffer


3108 4078 37EE  frfal:  jsb r7,repchk   ;common reply/rfnm checks
310C 4078 3920          jsb r7,ftrngt   ;arg: r6/buffer,r5,returns r6
3110 E642                Trap 3102,<;No trnblk for alloc - page 273>
3112 905E                br rfdone      ;ah well}
3114 4F92               tst r1,=ttmult  ;multi-packet sent?
3116 9A16                bz rfal1       ;was 1-pkt req
3118 4F94               tst r1,=ttreq
311A 9A04                bz rfal8d      ;was a message
311C 3D1B 000D          eorbm r1,trstat(r3)     ;release trnblk
3120 9005               br rfal8c

3122 4078 3806  rfal8d: jsb r7,rfledp   ;send our host the rfnm
3126 4078 3824          jsb r7,ledpc
312A 48F4       rfal8c: lda r7,=tmall0/ H100
312C 3A7D 65BA          addbm r7,rstate+<0*tstate>(r5)
3130 4078 385A          jsb r7,repfix   ;mark rfnm in trn mes bits
3134 4844               lda r4,r4
3136 9BE4                bm fqokju
3138 783C 000F          ldab r3,hinpid(r4)
313C 30B8 00AC          sta r3,@pid
3140 90DF               br fqokju
```

```
                ;got a one-packet allocate
                ;put stop bit in tran mes blk
                ;mark tran blk to retransmit message

3142 4B94       rfal1:  and r1,=ttreq   ;request outstanding?
3144 9A1D                bz rfal1e      ;no, must be duplicate
3146 3D1B 000D          eorbm r1,trstat(r3)     ;turn off ttreq.
314A 702B 000E          lda r2,trpack(r3)       ;packet to really send
314E 352B 000E          eorm r2,trpack(r3)      ;clear it out
3152 300D A5B0          sta r0,rmlock(r5)       ;;tmlock
3156 4078 14FC          call ounpck,whhi
315A 2000
315C 9A0F                bfail rfal1f
315E 7079 000A          lda r7,pkth(r1)
3162 4BFF               and r7,=pktcod  ;present type of packet.
     0000       .lif nz regtyp
                        sub r7,=regtyp
3164 3279 0004          addm r7,chkh(r1)
3168 3179 000A          subm r7,pkth(r1)
316C 48F0               lda r7,=0
316E 3079 0092          sta r7,inch(r1)
3172 4078 146C          call wheorb,whhi+whtsk
3176 2001
3178 9077               br rfal1x

317A E6CA       rfal1f: Trap 3312,<;Bad trnblk buffer - page 274>
317C 9079               br fqoku

317E E650       rfal1e: Trap 3120,<;Got a duplicate Allocate 1 - page 274>
3180 9078               br fqok
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 277
IMP.WARM;1      PAGE 37          *** warm page ***

                ;Got an out-of-range reply
                ;reset if its in range


3182 4078 37EE  foutof: jsb r7,repchk   ;common reply checks
3186 E7C0               Trap 3700,<;Got an Out-of-range - page 275>.
3188 4878 8000          lda r7,=tmrset
318C 307D 65BA          sta r7,rstate+<0*tstate>(r5)
3190 902C               br replyb       ;send reset right away

                fdead:          ;got Dead RFNM
                fresrq:         ;got a reset request
3192 701D 65B8          lda r1,rmmess(r5)       ;;tmmess
3196 4F9C               tst r1,=age0* HC
3198 9A04                bz frrq1       ;too young
319A 4C9F               ior r1,=age
319C 301D 65B8          sta r1,rmmess(r5)       ;;tmmess
31A0 4E78 001C  frrq1:  cmp r7,=rrqtyp*2
31A4 9166                be fqok        ;back will reset
31A6 4878 0080          lda r7,=tmstop  ;really fdead, set stop bit
31AA 3C7D 65BE          iorbm r7,tmstp-tmblks+rmblks(r5)
                frfnm:          ;got Regular RFNM
31AE 4078 37EE          jsb r7,repchk   ;common checks
31B2 4078 3920          jsb r7,ftrngt   ;get our tran block
31B6 E643                Trap 3103,<;No trnblk for RFNM or dead RFNM - page 275>
31B8 900B                br rfdone
31BA 306B 000A          sta r6,trdeds(r3)       ;note correct dead status
31BE 4078 3806  rfnm0:  jsb r7,rfledp
31C2 4868 02F8  rfnm1:  lda sp,=lstack  ;sp needed by trnfls
31C6 4078 18C6          call trnfls     ;don't need the packet any more
31CA 4078 3824          jsb r7,ledpc
31CE 4078 385A  rfdone: jsb r7,repfix   ;mark mess done
31D2 9096               br fqokju

                ;Got an Inc RFNM
                ;If we originated, do ledp0, else like other RFNM's.

31D4 4078 37EE  fincr:  jsb r7,repchk   ;common checks
31D8 4078 3920          jsb r7,ftrngt   ;find tran block
31DC E646                Trap 3106,<;No trnblk for inc RFNM - page 275>
31DE 90F8                br rfdone      ;sigh
31E0 781B 0003          ldab r1,trtyp1+1(r3)    ;any reply yet?
31E4 9AED                bz rfnm0       ;no, use common exit
31E6 90EE               br rfnm1        ;now join main path
```

```
                ;make a reply from a message
                ;r1/for blk,use,r2/pkth,a3/new midh,a5/mes blk,a6/buffer


31E8 783E 000A  replyb: ldab r3,pkth(r6)        ;use number
31EC A2B8               sll r3, D8
31EE 7C3E 0009          iorb r3,seqh+1(r6)      ;and block of source
31F2 701D 65B6  replyx: lda r1,rmctl(r5)
31F6 300D A5B0          sta r0,rmlock(r5)       ;unlocks
31FA 381E 0009  replyu: stab r1,seqh+1(r6)      ;his block
31FE 4B18 0F00          and r1,=fuse    ;his use number
3202 4BAF               and r2,=pktcod  ;want just incoming code
3204 7C1A 3696          iorb r1,reptab(r2)      ;new packet code
3208 301E 000A          sta r1,pkth(r6) ;to pkth
320C 303E 000E          sta r3,midh(r6)
3210 7028 027E          lda r2,tskbuf
3214 4078 266C          jsb r7,ocmove   ;need space from back pool
3218 63E4                cntrs+nbak
321A 903D                br replos
321C 4836               lda r3,r6       ;indexes through buffer
321E 4828 4300          lda r2,=hac+noblk+qertyp
3222 732E 0002          and r2,typh(r6) ;get pkttyp, hac, noblk bits
3226 4C28 1000          ior r2,=pribit  ;all replies have priority
322A 4890               lda r1,=0
322C 2013               sta r1,(r3)+    ;neth
322E 2023               sta r2,(r3)+    ;typh
3230 2013               sta r1,(r3)+    ;chkh
3232 7073               lda r7,(r3)     ;foreign IMP
3234 702E 000C          lda r2,dsth(r6) ;what was destination? (me)
3238 307E 000C          sta r7,dsth(r6) ;dsth
323C 2023               sta r2,(r3)+    ;srch
323E 4816               lda r1,r6
3240 4838 0014          lda r3,=data+4
3244 4868 02F8          lda sp,=lstack
3248 4078 17C8          call lpcksubr
324C 3039 0004          sta r3,chkh(r1) ;stuff it in
3250 7028 027E          lda r2,tskbuf
3254 3528 027E          eorm r2,tskbuf
3258 48F0               lda r7,=0
325A 3079 0092          sta r7,inch(r1) ;clear inch
325E 7078 00D2          lda r7,mapv2
3262 3078 FC04          sta r7,%map2
3266 300A 0350  rfail1x: sta r0,chan-chain(r2)   ;set flag-can't ack these
326A 4078 14AA          call ttskput
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 279
IMP.WARM;1      PAGE 39          *** warm page ***

                ; Exit routines
326E 48D1       fqoku:  lda r5,=1               ;no mes blk any more.
3270 7078 00D2  fqok:   lda r7,mapv2
3274 3078 FC04          sta r7,%map2
3278 7028 027E          lda r2,tskbuf
327C 9A06               ifnot zero
327E 4868 02F8            lda sp,=lstack
3282 4078 131E            call flushb,whtsk
3286 0001
                        endif
3288 4855               lda r5,r5
328A 9903               ifnot odd
328C 300D A5B0            unlock rmlock(r5)     ;;tmlock
                        endif
3290 4008 108E          tr loop

                ;reply dropped for no space

3294 E60A       replos: Trap 3012,<;Reply lost-no space - page 277>
3296 90EC               br fqoku

                ;got a bad packet code

3298 E645       fbad:   Trap 3105,<;Got msg w/illegal pkt code 13 - page 277>
329A 90EB               br fqok ;chuck it

                ;got a get-a-block message

329C 300D A5B0  fgetb:  sta r0,rmlock(r5)
32A0 48D1       blkget: lda r5,=1       ;no block yet
32A2 701E 0010          lda r1,data(r6)         ;source hacmem
32A6 703E 0012          lda r3,data+2(r6)       ; and haccom
32AA 4B28 F0FF          and r2,=-1?fuse         ;clear fuse field
32AE 4078 36EA          jsb r7,hostno
32B2 9A0B                bz blkg0       ;okay to get a block
32B4 303E 0010          sta r3,data(r6) ;save dead status
32B8 701E 000E          lda r1,midh(r6) ;recover source mes blk
32BC 48B0               lda r3,=0       ;no source block
32BE 307E 0002  replu2: sta r7,typh(r6) ;mark for got or got-no block
32C2 909C               tr replyu       ;reply and ack
```

```
                ;now find him a receive block
                ;if already have, return dupl got-a-block
                ;if no free, ignore

32C4 300D A5B0  blkg3:  sta r0,rmlock(r5)
32C8 701E 000E  blkg0:  lda r1,midh(r6) ;refetch messid,subtype
32CC 48D0               lda r5,=0*rmblks
32CE 703E 0006          lda r3,srch(r6)
32D2 48F1               lda r7,=1
32D4 704D 65B2  blkg4:  lda r4,rmimp(r5)
32D8 9B0F               bm blkg5        ;a free one

                ;probably should lock and recheck here

32DA 4E43               cmp r4,r3
32DC 8104               bne blkg7       ;not same imp
32DE 761D 65B6          cmp r1,rmctl(r5)
32E2 9112               be blkg6        ;same foreign block - dupl
32E4 4A58 0010  blkg7:  add r5,=rmlen
32E8 4E58 0380          cmp r5,=rmlen*rmnum
32EC 81F4               bne blkg4       ;try more
32EE 4857               lda r5,r7
32F0 8911               bno blkg1       ;got free one
32F2 E7C2               Trap 3702,<;No free rm blk - page 278>
32F4 90BD               tr fqoku

32F6 4877       blkg5:  lda r7,r7
32F8 89F6               bno blkg7       ;got spare already
32FA 704D 65B8          lda r4,rmmess(r5)
32FE 4FCC               tst r4,=age0* HC
3300 9AF2               bz blkg7        ;too young to use
3302 4875               lda r7,r5
3304 90F0               br blkg7        ;remember spare

3306 E7C6       blkg6:  Trap 3706,<;Rcvd dup Get-a-block - page 278>
3308 703D 65B8          lda r3,rmmess(r5)
330C 4B38 00F0          and r3,=luse
3310 9024               br blkg2        ;send duplicate got-a-block
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 281
IMP.WARM;1      PAGE 41          *** warm page ***

                ;initialize up a new receive block

3312 704D A5B0  blkg1:  lda r4,rmlock(r5)
3316 9AFE                bz .-4
3318 704D 65B2          lda r4,rmimp(r5)
331C 8BD4                bnm blkg3      ;someone else got it
331E 303D 65B2          sta r3,rmimp(r5)
3322 703E 0008          lda r3,seqh(r6)
3326 A7B8               rrl r3, H8      ;dest host from msg is our local
3328 303D 65B4          sta r3,rmhost(r5)
332C 301D 65B6          sta r1,rmctl(r5)
3330 7048 0282          lda r4,tskhst
3334 783C 001D          ldab r3,holhn(r4)
3338 383D 65BF          stab r3,rmlhn(r5)       ;into mes blk
333C 703D A5BA          lda r3,rstate+foo(r5)
3340 703D A5BC          lda r3,rmtype+foo(r5)
3344 703D 65B8          lda r3,rmmess(r5)
3348 4A38 0010          add r3,=luse0
334C 4B38 00F0          and r3,=luse
3350 303D 65B8          sta r3,rmmess(r5)
3354 300D A5B0          sta r0,rmlock(r5)       ;done setup
3358 303E 0008  blkg2:  sta r3,seqh(r6) ;clear lh of this word
335C A2B4               sll r3,4        ;to fuse position
335E A6D4               srl r5,4        ;my block number
3360 4C35               ior r3,r5       ;making up midh word
3362 4878 4000          lda r7,=qertyp  ;successful got-a-block
3366 90AC               br replu2       ;go for getblk reply
```

```
                ;got a giveback

3368 4078 3790  fgvb:   jsb r7,regchk   ;regular message checks
336C 8A71               bnz sterr       ;was in message state
336E 4078 38D8          jsb r7,reasf8   ;find an 8-pkt allocate
3372 E606               Trap 3006,<:No alloc to give back - page 280>
3374 9003               br gudgv2
3376 4078 3868          jsb r7,reasf    ;free the reas blk
337A 4891       gudgv2: lda r1,=1       ;rfnm
337C 9013               br gudral

                ;got inc on some msg
                ;clean up, return inc rep

337E 4078 3790  fincm:  jsb r7,regchk   ;regular message checks
3382 8A07               bnz inctr0      ;not idle
3384 820E               bng incm1       ;not multi
3386 4078 38D8  inctr1: jsb r7,reasf8
338A E607               Trap 3007,<:Incq/incm w/gvb, but no alloc to gb - page 280>
 338C 900A               br incm1
338E 9007               br inctr2       ;go free and unlock reas blk

3390 783E 0008  inctr0: ldab r3,seqh(r6)        ;get messnum, for blk
3394 4078 38A8          jsb r7,reasgt   ;any message fragment?
3398 E600               Trap 3000,<:No message for incm or incq - page 280>
339A 9003               br incm1        ;oh well
339C 4078 3868  inctr2: jsb r7,reasf    ;free reas blk, flush buffers
33A0 4893       incm1:  lda r1,=3       ;inc
33A2 48B2       gudral: lda r3,=2       ;msg state
33A4 9041               br ralput       ;fix states and search messtk

                ;got a raw packet

33A6 48D1       gudraw: lda r5,=1       ;no mes blk now
33A8 701E 000A          lda r1,unhacm(r6)       ;source hacmem
33AC 703E 0090          lda r3,bufe(r6) ;packet length
33B0 48F2               lda r7,=words   ;mark message one word shorter
33B2 317E 0090          subm r7,bufe(r6)
33B6 4A36               add r3,r6       ;get to end of packet
33B8 7033               lda r3,(r3)     ;source haccom
33BA 4078 36EA          jsb r7,hostno
33BE 8A49               bnz fqokgo      ;dest dead: ignore
33C0 7028 027E          lda r2,tskbuf
33C4 4078 266C          jsb r7,ocmove   ;get reassembly space
33C8 63DC               cntrs+nre       ;space source
33CA 9043               br fqokgo       ;no space: lose
33CC 901A               br gdraw3       ;okay, give host the packet
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 283
IMP.WARM;1      PAGE 43          *** warm page ***

                ;Got a request

33CE 4078 3790  freq:    jsb r7,regchk    ;reg mess checks
33D2 8A3E                 bnz sterr       ;not in idle state
33D4 923B                 bg gudrq8       ;:mltpkt ;multi-packet
33D6 4892                 lda r1,=2       ;dead rfnm to send
33D8 4F28 0100            tst r2,=hostng  ;dest host alive?
33DC 8AE3                 bnz gudral      ;died recently

                ;1-pkt req. take mess if there's room

33DE 7028 027E            lda r2,tskbuf
33E2 4078 266C            jsb r7,ocmove   ;find some space
33E6 63DC                  cntrs+nre      ; in reassembly
33E8 901D                  br gudrq2      ;none to be had
33EA 7028 0284            lda r2,tskbts   ;recover old state
33EE 4F28 0800            tst r2,=mesnxt
33F2 9A1E                  bz gudrq5      ;not next to go
33F4 4890                 lda r1,=0
33F6 48B2                 lda r3,=2
33F8 4078 36AA            jsb r7,rallyp
33FC 7028 027E            lda r2,tskbuf
3400 3528 027E  gdraw3:   eorm r2,tskbuf
3404 70B8 62E8            lda r3,@clock
3408 303E 0098            sta r3,qt(r6)
340C 305E 009C            sta r5,bufb(r6) ;prepare buffer for ih
3410 703E 0090            lda r3,bufe(r6) ;setup length in bytes
3414 49BE                 sub r3,=data-2
3416 4078 1468            jsb r7,owheob   ;give buffer to ih
341A 0041                  whtsk?whih
341C 4812                 lda r1,r2
341E 4008 3612            jmp t2h0        ;give it to host

3422 4892       gudrq2:   lda r1,=2       ;alloc 1 to send
3424 48B1       gudrq3:   lda r3,=1       ;request
3426 4078 36AA  ralput:   jsb r7,rallyp
342A 4008 3566            jmp nxmes0
```

```
                ;Got a 1-pkt req out of order

342E 4833       gudrq5: lda r3,r3       ;;cntrs+nre ;from cmove
3430 8BF9                bnm gudrq2     ;no take unless in guarantee
3432 4078 396C          call reasal     ;get a free reas blk.
3436 81F6                bne gudrq2     ;none to be had
3438 305A 0004          sta r5,rid(r2)  ;set up the reas blk
                        ;lda r3,=1      ;returned by reasal }}
343C 383A 0009          stab r3,ral(r2) ;how much allocated
3440 701D 65B8          lda r1,rmmess(r5)       ;want use number
3444 381A 0007          stab r1,ruse(r2)
3448 9034               br gudrq6       ;into midst of 8-pkt code

                ;Got a request for 8. Queue it

344A 4891       gudrq8: lda r1,=1       ;pending allocate 8
344C 90EC               br gudrq3       ;put it in rally

344E E608       sterr:  Trap 3010,<;Rstate violation - page 282>
3450 4008 3270  fqokgo: jmp fqok

                ;Got a message packet

3454 4078 3790  fmess:  jsb r7,regchk   ;normal checks
3458 9207                bg gudtr8      ;;mltpkt ;multi-packet fragment
345A 9AFA                bz sterr       ;must be in message state if 1-pkt
345C 4078 38DE          jsb r7,reasf1   ;find an all for this guy
3460 E7CA                Trap 3712,<;No alloc for 1-pkt msg - page 282>
3462 90F7                br fqokgo
3464 9026               br gudrq6       ;process the packet as in req 1
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 285
IMP.WARM;1      PAGE 45          *** warm page ***

                ;a piece of an 8-packet

3466 783E 0008  gudtr8: ldab r3,seqh(r6)        ;messno, mes blk
346A 4078 38A8          jsb r7,reasgt   ;return reas blk in a2
346E 901D                br newmes
3470 0000                0      ;no reas block. start new one
3472 305E 009C  oldmes: sta r5,bufb(r6) ;prepare buffer for ih
3476 70F8 62E8          lda r7,@clock
347A 307E 0098          sta r7,qt(r6)
347E 703E 000A          lda r3,pkth(r6) ;this was smashed
3482 4843               lda r4,r3       ;pkth
3484 A6C4               srl r4,4        ;pktnum to low-order
3486 4BCF               and r4,=pktnum/ D16     ;just the number
3488 707E 0090          lda r7,bufe(r6) ;calculate length of packet
348C 49FE               sub r7,=data-2
348E 7068 00D2          lda r6,mapv2
3492 3068 FC04          sta r6,%map2
3496 486A 000A          lda r6,=reasq(r2)
349A 7016       oldm3:  lda r1,(r6)
349C 991B                bo oldm4       ;add it to the end
349E 7649 0350          cmp r4,chan-chain(r1)
34A2 8C17                bnl oldm5      ;if it goes here
34A4 4861               lda r6,r1
34A6 90FA               br oldm3

                ;get a reas block for this multi

34A8 4078 38D8  newmes: jsb r7,reasf8   ;find a reas block
34AC E605                Trap 3005,<:No reas blk for alloc 8-pkt msg - page 283>
34AE 9057                br fqokj
34B0 4818 00FF  gudrq6: lda r1,= HFF    ;largest byte
34B4 381A 0008          stab r1,rmax(r2)
34B8 301A 000A          sta r1,reasq(r2)        ;odd is good enough
34BC 701A 400E          lda r1,rsfbt+foo(r2)    ;clear byte count
34C0 4892               lda r1,=2       ;have a mes num now
34C2 301A 0002          sta r1,rsf(r2)  ;;reasst ;clear rsf
34C6 781E 0008          ldab r1,seqh(r6)        ;message number
34CA 381A 0006          stab r1,remess(r2)
34CE 90D2               br oldmes       ;can now process packet
```

```
34D0 9144       oldm5:  be oldmsq       ;if duplicate packet
34D2 327A 000E  oldm4:  addm r7,rsfbt(r2)       ;sigma pkt length
34D6 7078 027E          lda r7,tskbuf
34DA 3578 027E          eorm r7,tskbuf
34DE 304F 0350          sta r4,chan-chain(r7)   ;note the packet number
34E2 3017               sta r1,(r7)     ;chain the pkt to the reas blk
34E4 8903               bno .+6
34E6 307A 000C           sta r7,reasqe(r2)      ;note last packet
34EA 3076               sta r7,(r6)
34EC 7068 A3C8          lda r6,nf       ;lock where word
34F0 9AFE                bz .-4
34F2 701F 06A0          lda r1,where-chain(r7)  ;move pkt count to nre
34F6 4B18 003E          and r1,=whctr
34FA 4918 0043          sub r1,=nre-whtsk+whih
34FE 311F 06A0          subm r1,where-chain(r7)
3502 3068 A3C8          sta r6,nf
3506 4A18 641B          add r1,=nre-whtsk+whih+cntrs    ;address of old counter
350A 3018 017E          sta r1,temp1
350E 4890               lda r1,=0       ;amount to adjust if not last
3510 4F38 4000          tst r3,=lstpkt  ;this the last packet?
3514 9A0C                bz notla       ;not last packet
3516 384A 0008          stab r4,rmax(r2)        ;this will be highest pkt num
351A 4833               lda r3,r3       ;doing single-packet?
351C 9915                bo oldm6       ;;reqtyp ;single packet out of order request
351E 481C 0001          lda r1,=1(r4)   ;0-7 => 1-8
3522 791A 0009          subb r1,ral(r2) ;how much allocated
3526 9A03               bz notla        ;don't waste needless time
3528 4078 37BA           jsb r7,fixnre  ;return surplus from reassembly
352C 4991       notla:  sub r1,=1       ;present pkt returns nal space
352E 48F1               lda r7,=1       ;adjustment for source
3530 7038 A3C8          lda r3,nf       ;lock counters
3534 9AFE                bz .-4
3536 3218 63D4          addm r1,nal     ;fix nal the right amount
353A 31F8 017E          subm r7,@temp1  ;return source count
353E 9B02               bm notlb        ;was within range
3540 4AB1                add r3,=1      ;return common space too
3542 3038 A3C8  notlb:  sta r3,nf       ;unlock again
3546 4891       oldm6:  lda r1,=1
3548 3A1A 0002          addbm r1,rsf(r2)
354C 300A 4000          sta r0,reaslk(r2)
3550 4890               lda r1,=0       ;;rmtype
3552 48B2               lda r3,=2       ;;rstate        ;message state
3554 4008 3426          tr ralput               ;fix states and search messtk

3558 300A 4000  oldmsq: sta r0,reaslk(r2)
355C 4008 3270  fqokj:  jmp fqok
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 287
IMP.WARM;1      PAGE 47          *** warm page ***

                ;process messages with getblk bit
                ;r1/his block

                ;got a reset block

FC00 0400       page Warm

5C3E 4878 8000  freset: lda r7,=sign    ;deaden this block
5C42 347D 65B2          iorm r7,rmimp(r5)
5C46 4008 31E8          jmp replyb      ;unlock,reply,and exit

                ;got a got-no-block (dead or refused)
                ;r7/dead reason

5C4A 387D 65B8  tmnot:  stab r7,rmmess(r5)      ;;tmmess ;save dead reason
5C4E 4838 8000          lda r3,=tmrset
5C52 343D 65BA          iorm r3,rstate(r5)      ;;tstate
5C56 701E 0010          lda r1,data(r6) ;dead subcode
5C5A 301D 65BC          sta r1,rmtype(r5)       ;;tmdeds ;save reason in tm blk
5C5E 9009               br fqokj2

                ;got a reset reply

5C60 4818 8000  fresrp: lda r1,=sign    ;deaden this block
5C64 341D 65B2          iorm r1,rmimp(r5)       ;;tmimp
5C68 701D A5BA          lda r1,rstate+foo(r5)   ;;tstate
5C6C 9B02               bm fqokj2
5C6E E644               Trap 3104,<:Res rep when not resetting - page 285>
5C70 4008 3270  fqokj2: jmp fqok        ;bye bye block

                ;received a got-a-block

5C74 4838 4000  fgetr:  lda r3,=tminit
5C78 733D 65BA          and r3,rstate(r5)       ;;tstate
5C7C 9AFA               bz fqokj2       ;already active
5C7E 787E 0002          ldab r7,typh(r6)        ;did we get a block?
5C82 4BF3               and r7,=<hac+noblk>_- H8        ;test errors
5C84 8AE3               bnz tmnot       ;;chacc cldrp chstd ;nope, no block
5C86 353D 65BA          eorm r3,rstate(r5)      ;;tstate
5C8A 4B18 0FFF          and r1,=gethan?<-1>     ;don't want these:
5C8E 341D 65B6          iorm r1,rmctl(r5)       ;;tmctl ;handling type alredy there
5C92 781C 000F          ldab r1,hinpid(r4)
5C96 3098 00AC          sta r1,@pid
5C9A 90EB               br fqokj2

                page HLCode
```

```
                ;message stack processing
                ; we got a message we could give host, or have completed
                ; a message, and must search the message stack for more
                ; complete ones to give him.
                ;Make this an independent strip someday }}}

3560 300A 4000  srcst4: sta r0,reaslk(r2)
3564 90FC               br fqokj

3566 7078 00D2  nxmes0: lda r7,mapv2
356A 3078 FC04          sta r7,%map2
356E 703D 65BC          lda r3,rmtype(r5)
3572 701D 65BA          lda r1,rstate(r5)
3576 9016               br nxmes3       ;entry for noo change to mess no

3578 703D 65BC  nxmes1: lda r3,rmtype(r5)
357C 701D 65BA          lda r1,rstate(r5)
3580 A792       nxmes4: rrl r1,2
3582 A7B2               rrl r3,2
3584 301D 65BA          sta r1,rstate(r5)
3588 303D 65BC          sta r3,rmtype(r5)
358C 48A1               lda r2,=mess0/ H100
358E 3A2D 65B8          addbm r2,rmmess(r5)
3592 702D 65B8          lda r2,rmmess(r5)
3596 4FAC               tst r2,=age0* HC
3598 9A05                bz nxmes3      ;young block
359A 4DA4               eor r2,=age0*4
359C 4BAF               and r2,=age
359E 352D 65B8          eorm r2,rmmess(r5)
35A2 4F93       nxmes3: tst r1,=3
35A4 9ADC                bz fqokj       ;idle
35A6 890B                bno nxmes6     ;mess state
35A8 4F92               tst r1,=2
35AA 8AD9                bnz fqokj      ;reply state
35AC 4FB2               tst r3,=2       ;already in alloc to send?
35AE 8AD7                bnz fqokj      ;yes, can't go farther
35B0 4CB2               ior r3,=2
35B2 99E7                bo nxmes4      ;all 8 to send - bump mess no
35B4 EFD8               Trap 7730,<;Illegal rstate/type - page 286>
35B6 303D 65BC          sta r3,rmtype(r5)
35BA 90D1               br fqokj

35BC 4FB3       nxmes6: tst r3,=3
35BE 9A06                bz smstk       ;see if mess is in
35C0 4C91       nxmes5: ior r1,=1       ;mes to reply
35C2 4FB2               tst r3,=2
35C4 8ADE                bnz nxmes4     ;inc or dead
35C6 4DB1               eor r3,=1
35C8 90DC               br nxmes4
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 289
IMP.WARM;1      PAGE 49          *** warm page ***

                ;the next guy to go may be in messtk

35CA 783D 65B8  smstk:  ldab r3,rmmess(r5)
35CE 4078 38A8          jsb r7,reasgt
35D2 90C5                br fqokj
35D4 0000                0      ;none there? okay
35D6 786A 0008          ldab r6,rmax(r2)
35DA 7E6A 0002          cmpb r6,rsf(r2)
35DE 82C1                bng srcst4     ;not complete
35E0 7018 0284          lda r1,tskbts   ;finally, checking dest dead
35E4 4F18 0100          tst r1,=hostng
35E8 9A09                bz smsok       ;host is okay
35EA 4078 3868          jsb r7,reasf    ;flush the message
35EE 701D 65BA          lda r1,rstate(r5)       ;refetch state bits
35F2 703D 65BC          lda r3,rmtype(r5)
35F6 4CB2               ior r3,=2       ;for dead reply
35F8 90E4               br nxmes5

35FA 701A 000A  smsok:  lda r1,reasq(r2)        ;first pkt
35FE 706A 000C          lda r6,reasqe(r2)       ;last pkt
3602 703A 000E          lda r3,rsfbt(r2)        ;total length in bytes
3606 48F1               lda r7,=1
3608 387A 0003          stab r7,reasst(r2)
360C 300A 4000          sta r0,reaslk(r2)
3610 4826               lda r2,r6
                ;fall into t2h0
```

```
                ;here with r1/first packet,r2/last,r3/bytes so far.
                ;  r5/rmblk

3612 4868 02F8  t2h0:   lda sp,=lstack
3616 1056               save r5          ;save messblock pointer.
3618 4851               lda r5,r1        ;save r1 over unpack
361A 4078 1512          call unpack      ;last pkt
361E A2B3               sll r3,3         ;bytes to bits.
3620 7219 0090          add r1,bufe(r1) ;last data word offset.
3624 7011               lda r1,(r1)      ;last data word.
3626 9A04               ifnot zero
                          repeat
3628 49B1                   sub r3,=1
362A A791                   rrl r1,1            ;find low-order 1-bit
362C 8BFE                   until minus
                          endrepeat
                        endif
362E 4875               lda r7,r5        ;exch r5,r2
3630 4852               lda r5,r2
3632 4827               lda r2,r7
3634 7078 00D2          lda r7,mapv2
3638 3078 FC04          sta r7,%map2
363C 4078 1500          call unpckc,whih        ;first pkt
3640 0040
3642 8A03               if fail
3644 E6D0                 Trap 3320,<;Bad buffer in t2h - page 288>
3646 9020               else
3648 3031                 sta r3,(r1)           ;;neth
364A 7048 0282            lda r4,tskhst
364E 483C 0062            lda r3,=ehq(r4)
3652 7079 0002            lda r7,typh(r1)       ;get typh for priority check.
3656 A2F3                 sll r7,3              ;priority bit to sign
3658 8B02                 if minus              ;;pribit
365A 4AB4                   add r3,=ehpq-ehq
                          endif
365C 7078 00D2            lda r7,mapv2
3660 3078 FC04            sta r7,%map2
3664 7078 62DA            lda r7,time
3668 4A78 04B0            add r7,=sec30         ;allow 30 seconds for sending.
366C 307A 0350            sta r7,chan-chain(r2)
3670 707C 405C            lock lockih(r4)
3674 9AFE
3676 30A3                   sta r2,@(r3)                ;;ehq,ehpq
3678 3053                   sta r5,(r3)           ;;ehq,ehpq
367A 300C 405C            unlock lockih(r4)
367E 787C 000E            ldab r7,hotpid(r4)    ;poke host.
3682 30F8 00AC            sta r7,@pid
                        endif
3686 6056               restore r5
3688 8903                 to fqok               ;taking raw pkt.
368A 4008 3270
368E 4008 3578          tr nxmes1               ;mes blk, maybe more messages done.
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 291
IMP.WARM;1      PAGE 51          *** warm page ***

                ;dispatch tables

                ;reply codes for duplicates, rfnms

3692    05      frpcod: .byte crfnm,crfnm,cdestd,cinctr
3693    05
3694    07
3695    09

                ;nominal reply types for replies from task

3696    06      reptab: .byte dedtyp, dedtyp, dedtyp, dedtyp    ;for dest dead
3697    06
3698    06
3699    06
369A    04              .byte rfntyp, rfatyp, dedtyp, inrtyp    ;for dupl replies
369B    05
369C    06
369D    07
369E    0C              .byte outtyp, gbrtyp, rrptyp, 0 ;for block stuff
369F    0D
36A0    0F
36A1    00
36A2    0A              .byte rsbtyp, rsbtyp, rsbtyp, 0 ;for block replies
36A3    0A
36A4    0A
36A5    00

                ;bits for fixing meseen,mesout

36A6 0400       meststst: mesout  ;-1 - -7: (no bit)
36A8 0600               meseen+mesout   ;1 - 7: seen
```

```
                ;subroutines

.comnt |
rally format:
rstate, rmtype each have 8 2-bit fields, with the following values
rstate:   0=idle    1=request  2=message   3=reply
rtype: 0 rfnm sent  illegal    mes going  send rfnm
       1 r/al sent  req8 recd  givb recd  send r/al
       2 dead sent  send all1  dead recd  send dead
       3 incr sent  send all8  incr recd  send incr

messno always has next to give host

legal messno ranges:
all idles: n-1 to n-8 for duplicate replies
req8/givb/dead/incr recd: n to n+7
send all1: n to n+7
send all8, all replies: n-1 to n-8
mes going: n to n+7 (8-pkt and in mes stk)
           n+1 to n+7 (1-pkt and in messtk)
           n-1 to n-8 (on host queue)
put an entry into rstate/rtype
new rstate in r3, rtype in r1, call on r7, uses r6, temp1
expects ralshf to be set up
|

36AA 3078 017E  rallyp: sta r7,temp1
36AE 7078 0280          lda r7,ralshf
36B2 A217               sll r1,r7
36B4 751D 65BC          eor r1,rmtype(r5)
36B8 731F 36DA          and r1,ralbts(r7)
36BC 351D 65BC          eorm r1,rmtype(r5)
36C0 A237               sll r3,r7
36C2 753D 65BA          eor r3,rstate(r5)
36C6 733F 36DA          and r3,ralbts(r7)
36CA 353D 65BA          eorm r3,rstate(r5)
36CE 4838 0026          lda r3,=bklpid
36D2 30B8 00AC          sta r3,@pid
36D6 4088 017E          jmp @temp1

                ;rally rstate and rtype masks

36DA 0003       ralbts: 03
36DC 000C                H0C
36DE 0030                H030
36E0 00C0                H0C0
36E2 0300                H0300
36E4 0C00                H0C00
36E6 3000                H03000
36E8 C000                H0C000
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 293
IMP.WARM;1      PAGE 53          *** warm page ***

        .comnt |
test host state and access: r1/hacmem,r2/forus flags,r3/haccom,
   r4/(in hostnm)our host,r5/rmblk(if any),r6/buffer,r7/return
 returns: bad args/fqok
          failure/set hostng=1,cc nz,r3 reason,r7 new typh
          success/r4 to hstblk
 uses temp1,temp2,r1,r3,r7; others as noted above
|
36EA 784E 0008  hostno: ldab r4,seqh(r6)        ;set up host number
36EE 3078 017E  hostnm: sta r7,temp1    ;entry point with r4 set up already
36F2 3038 0180          sta r3,temp2    ;save up haccom
36F6 7038 62E0          lda r3,myimp
36FA 9A0A                bz hostn4
36FC 48F4               lda r7,=hninit          ;IMP not initialized yet
36FE 703E 0006          lda r3,srch(r6)         ;source IMP this msg
3702 A2B1               sll r3,1                ;doubled
3704 703B 72FC          lda r3,spfrut(r3)       ;get route to this IMP
3708 4B38 800F          and r3,=routef+spfded
370C 8A3A                bnz hostnd             ;not from us: return dest dead
370E 4E48 0018  hostn4: cmp r4,=nrh             ;max hosts per IMP
3712 9207                bg hostn0              ;real host
3714 4948 00FC          sub r4,= H100-nfh       ;test for proper fake number
3718 8B14                bnm hostn1             ;legal fake
371A 4878 4300  hostn5: lda r7,=hac+noblk+qertyp        ;hac failure
371E 9035               br hostn6

                hostn0:
    0001        .IF NZ VHA                      ;do the inverse VHA check
3720 3068 02F6    sta r6,lstack-words           ;"push" old SP
3724 4868 02F6    lda sp,=lstack-words
3728 1016         save r1                       ;preserve r1 too
372A 701E 0002    lda r1,words(sp)              ;and load it with buff addr
372E 4078 3E7A    call tskvha                   ;check address
3732 8A04         if fail                       ;no good
3734 6016           restore r1
3736 6066           pop r6
3738 90F1           br hostn5                   ;send dest dead
                  endif
373A 6016         restore r1
373C 6066         pop r6
                .ENDC
373E 4AC4               add r4,=nfh     ;skip fakes
```

```
3740 A2C1       hostn1: sll r4,1
3742 487C FFF8          lda r7,=-nfh*words(r4)  ;minus if fake
3746 A3F8               rll r7, H8      ;swap bytes
3748 7478 62E2          ior r7,mine     ;and get my IMP number
374C 4838 0010          lda r3,=lhaccs          ;how many haccs special
3750 567B 40E0          cmp r7,hacspc(-r3)      ;match an entry?
3754 9102                be .+4         ;got a match
3756 88FD                bnlp .-6       ;more to search, last is default
3758 771B 40D0          tst r1,haccom(r3)       ;ours talk to his?
375C 8A06                bnz hostnb     ;yup
375E 7418 0180          ior r1,temp2    ;if same mem group(s)
3762 771B 40C0          tst r1,hacmem(r3)       ;his talk to ours?
3766 9ADA                bz hostn5      ;nope
3768 48F3       hostnb: lda r7,#hnexis  ;host doesn't exist
376A 704C 6326          lda r4,h2pblk(r4)
376E 3048 0282          sta r4,tskhst
3772 9B07                bm hostnd      ;no such host
3774 787C 001F          ldab r7,hihd(r4)        ;;hostup
    0001        .IF NZ VHA
                                        ;need not check leader mode if VHA
3778 9A0A                bz hostn3      ;go ahead with connection, cc z
                .IFF ;NZ VHA
                         bnz hostn7     ;host dead
                        ldab r7,homode(r4)      ;his address reportable?
                         bnz hostnc     ;ok if new format(now set cc z})
                        lda r3,srch(r6) ;otherwise check for low host/imp
                        tst r3,=-1?srcei
                         bnz hostn9     ;source imp too big
                        ldab r3,srchst(r6)
                        lda r5,r5       ;got a mes blk?
                        bo .+6  ;nope
                         ldab r3,rmhost(r5)     ;;tmhost ;remote host
                        cmp r3,= H100-nfh
                        bg hostna
                         add r3,=nfh
                hostna: tst r3,=-1?<<frmimp}srceh>/ H40>        ;;chstd}
                         bz hostn3      ;go ahead with connection, cc z
                hostn9: lda r7,=hac+qertyp      ;refuse, can't report source addr
                        br hostn6
                hostnc: tst r7,=0       ;set cc zero
                        br hostn3       ; and return
                .ENDC ;NZ VHA
377A 703C 002C  hostn7: lda r3,deadsc(r4)       ;use dead status
377E 8A03                bnz hostn2             ;(if any)
3780 483F FFE0  hostnd: lda r3,=chnoti(r7)      ; else report hihd
3784 4878 4100  hostn2: lda r7,=noblk+qertyp
3788 4C28 0100  hostn6: ior r2,=hostng  ;set cc nz
378C 4088 017E  hostn3: jmp @temp1
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 295
IMP.WARM;1      PAGE 55          *** warm page ***

.comnt |
common checks for regular messages
return cc: z idle state, nz msg state, g mltpkt, ng sing pkt
|

3790 4F28 0600  regchk: tst r2,=meseen+mesout
3794 9A03               tnz fqok        ;duplicate
3796 4008 3270
379A 4D38 FFFF  regch2: eor r3,=-1
379E 4AB1               add r3,=1
37A0 A2B1               sll r3,1
37A2 3038 0280          sta r3,ralshf   ;amount to shift for rstate, rtype
37A6 701D 65BA          lda r1,rstate(r5)
37AA A613               srl r1,r3
37AC 4B93               and r1,=3
37AE 8904               if odd          ;can't do a thing in request or reply state
37B0 4DA8                 eor r2,=inqtyp        ;unless it's an inc query.
37B2 4FAF                 tst r2,=pktcod
37B4 8AF1                 tnz fqok      ;it isn't, so it can be ignored.
                        endif
37B6 4EA0               cmp r2,=0       ;set g, e con codes
37B8 4007               jmp (r7)        ;return successful

                ;nre, nal subroutines
                ;adjustment (must be negative) in r1, uses r3,r6, return in r7

37BA 7038 A3C8  fixnre: lda r3,nf       ;lock the counters
37BE 9AFE                bz .-4
37C0 900B               br nal2 ;drop into the following

37C2 7038 A3C8  fixnae: lda r3,nf       ;lock counts
37C6 9AFE                bz .-4
37C8 3218 63D4          addm r1,nal     ;adjust nal
37CC 8B05               bnm nal2
37CE EFD0               Trap 7720,<:Nal gone neg - page 293>
37D0 4890               lda r1,=0
37D2 3018 63D4          sta r1,nal      ;reset nal
37D6 7068 63DC  nal2:   lda r6,cntrs+nre        ;now do nre
37DA 9B02                bm nre2        ;not using slush
37DC 4A36               add r3,r6       ;momentarily return surplus
37DE 4A61       nre2:   add r6,r1       ;adjust nre
37E0 9B02                bm nre3        ;now within min
37E2 4936               sub r3,r6       ;now take surplus again
37E4 3068 63DC  nre3:   sta r6,cntrs+nre        ;new nre
37E8 3038 A3C8          sta r3,nf       ;new nf, unlock
37EC 4007               jmp (r7)        ;done
```

```
                ;common reply checking subroutine

37EE 4F28 0200  repchk: tst r2,=meseen
37F2 9AD2                tz fqok        ;out of range, ignore
37F4 4818 0100          lda r1,= H100
37F8 A613               srl r1,r3
37FA 771D 65BA          tst r1,rstate+<0*tstate>(r5)
37FE 8ACC                tnz fqok       ;duplicate, discard
3800 3018 0280          sta r1,repbit   ;save for repfix
3804 4007               jmp (r7)        ;succeed

                ;set up IMP-Host reply type

3806 4BAF       rfledp: and r2,=pktcod  ;just packet code
3808 701B 0008          lda r1,trmidl(r3)               ;get messid
380C 4B18 FFF0          and r1,#-1?subtyp       ;;cimpd ;clear subtyp (default subtyp)
3810 4EA6               if r2 = #dedtyp                 ;dest host dead
3812 8102
3814 4C91                 ior r1,=chstd                 ;tell host
                        endif
3816 301B 0008          sta r1,trmidl(r3)               ;put into tran block
381A 781A 368E          ldab r1,frpcod-rfntyp(r2)       ;get appropriate IH code
381E 381B 0003          stab r1,trtypl+1(r3)            ;message type
3822 4007               jmp (r7)

                ;queue a reply on a Host

3824 781B 000D  ledpc:  ldab r1,trstat(r3)      ;reply required?
3828 8914                bno ledpc2     ;;ttrfnm ;no, just free trnblk
382A 7018 0284          lda r1,tskbts
382E 4F18 0100          tst r1,=hostng  ;if our host has died
3832 8A0F                bnz ledpc2
3834 701D 65B6          lda r1,rmctl(r5)        ;;tmctl ;now for handling type
3838 4B18 7000          and r1,=getmax  ;just max pkts
383C A694               srl r1,4        ;max pkts into place
383E 702D 65B6          lda r2,rmctl(r5)        ;;tmctl ;test pri bit here
3842 8B03               bnm ledpc1      ;;getpri ;not priority
3844 4C18 8000           ior r1,=priled ;make it priority
3848 7C1D 65B4  ledpc1: iorb r1,rmhost(r5)      ;;tmhost ;dest (remote) host
384C 4008 188E          tr ledp2        ;now lock and queue it

3850 701B 000C  ledpc2: lda r1,trntim(r3)       ;;trstat ;free trnblk
3854 351B 000C          eorm r1,trntim(r3)      ;and clear it
3858 4007               jmp (r7)

                ;common reply fixer subroutine
                ;put tmess bit in, plus timeouts

385A 7018 0280  repfix: lda r1,repbit   ;set up by repchk
385E 341D 65BA          iorm r1,rstate+<0*tstate>(r5)
3862 300D A5B0          sta r0,rmlock+<0*tmlock>(r5)    ;won't need this no more
3866 4007               jmp (r7)
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 297
IMP.WARM;1      PAGE 57          *** warm page ***

                ;reassembly subroutines

                ;flush the pkts in a reas blk and free it
                ;in: r2/reas blk, uses: r1,r3,r4,r6,temp1,temp2

3868 3078 0180  reasf:  sta r7,temp2
386C 7078 00D2          lda r7,mapv2
3870 3078 FC04          sta r7,%map2
3874 783A 0003          ldab r3,reasst(r2)      ;any of msg in?
3878 990B               bo reasf0       ;no
387A 4832               lda r3,r2       ;need r2 for flush
387C 702B 000A          lda r2,reasq(r3)
3880 7042       reasf1: lda r4,(r2)
3882 4078 1314          call oflush,whih
3886 0040
3888 4824               lda r2,r4
388A 89FB               bno reasf1      ;more to flush
388C 4823               lda r2,r3       ;restore reas blk ptr
388E 781A 0002  reasf0: ldab r1,rsf(r2) ;how many so far.
3892 791A 0009          subb r1,ral(r2) ;how many to return
3896 4078 37C2          jsb r7,fixnae   ;fix both nal and nre
389A 4891               lda r1,=1
389C 381A 0003          stab r1,reasst(r2)      ;frees block
38A0 300A 4000          sta r0,reaslk(r2)
38A4 4088 0180          jmp @temp2
```

```
                ;find a reassembly block
                ;a3/messno,a5/rmblk,a7/return,gets reas blk in a2/skip 4 if win

38A8 4828 6CB0  reasgt: lda r2,=messtk
38AC 701A 4000  relook: lda r1,reaslk(r2)
38B0 9AFE                bz .-4
38B2 781A 0003          ldab r1,reasst(r2)
38B6 9909                bo reasg1      ;no id for this one yet (free or only all)
38B8 7E3A 0006          cmpb r3,remess(r2)
38BC 8106                bne reasg1     ;wrong mess
38BE 765A 0004          cmp r5,rid(r2)
38C2 8103                bne reasg1     ;wrong rblk
38C4 400F 0004          jmp 4(r7)

38C8 300A 4000  reasg1: sta r0,reaslk(r2)
38CC 4A28 0010          add r2,=lrblk
38D0 4E28 6E30          cmp r2,=emsstk  ;past end yet?
38D4 92EC                bg relook      ;nope, keep trying
38D6 4007               jmp (r7)        ;fail = non-skip

                ;find an allocated reassembly block
                ; reasf8 finds an 8-pkt allocate, reasf1 finds a 1-pkt.
                ; a5=message block, returns reas block in a2, uses a1,a3
                ; returns with reas block locked

38D8 7038 12E6  reasf8: lda r3,pkts8
38DC 9002               br reasfz

38DE 48B1       reasf1: lda r3,=1
38E0 4828 6CB0  reasfz: lda r2,=messtk
38E4 701A 4000  reasfv: lda r1,reaslk(r2)
38E8 9AFE                bz .-4
38EA 781A 0003          ldab r1,reasst(r2)
38EE 4E93               cmp r1,=3       ;this an allocate?
38F0 9109                be reasfx      ;yup - so it's got possibilities
38F2 300A 4000  reasfw: sta r0,reaslk(r2)       ;unlock block
38F6 4A28 0010          add r2,=lrblk
38FA 4E28 6E30          cmp r2,=emsstk  ;past end of reas blks?
38FE 92F3                bg reasfv      ;nope
3900 4007               jmp (r7)        ;yup - fail return

3902 765A 0004  reasfx: cmp r5,rid(r2)
3906 81F6                bne reasfw
3908 7E3A 0009          cmpb r3,ral(r2) ;correct size?
390C 81F3                bne reasfw     ;nope, keep looking
390E 781A 0007          ldab r1,ruse(r2)        ;correct use?
3912 751D 65B8          eor r1,rmmess(r5)
3916 4B18 00F0          and r1,=luse
391A 8AEC                bnz reasfw     ;no, keep looking
391C 400F 0004          jmp 4(r7)       ;a winner: return leaving block locked (})
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 299
IMP.WARM;1      PAGE 59          *** warm page ***

                ;find a transaction block
                ;in r5/tmblk,r6/buffer,r7/return, uses temp1
                ;if win, skip 4 with r1/trstat, r3/trnblk, r6/dead status
                ;restores %map2 to mapv2
                ;uses temp1, also

3920 3078 017E  ftrngt: sta r7,temp1    ;save return
3924 707E 0008          lda r7,seqh(r6) ;message number
3928 4B78 FF00          and r7,=mesnum  ;just message number
392C 4835               lda r3,r5       ;mes blk index
392E A7B4               rrl r3,4        ;my block
3930 4C7B FFC8          ior r7,=-<<tmblks-rmblks>_-4>(r3) ;r7 gets messno, block
3934 706E 0010          lda r6,data(r6) ;maybe dead status
3938 7038 00D2          lda r3,mapv2
393C 3038 FC04          sta r3,%map2
3940 4838 8910          lda r3,=trnblk
3944 781B 000D  ftrg1:  ldab r1,trstat(r3)
3948 9A0B                bz ftrg2       ;not in use
394A 4F18 0088          tst r1,=ttledr+ttresv
394E 8A08                bnz ftrg2      ;being used for something else
3950 767B 0004          cmp r7,trhstl(r3)
3954 8105                bne ftrg2      ;no message num/blk match
3956 7078 017E          lda r7,temp1
395A 400F 0004          jmp 4(r7)      ;got it

395E 4A38 0010  ftrg2:  add r3,=trnl
3962 4E38 8F50          cmp r3,=trnblk+trntot
3966 81EF               bne ftrg1      ;more to search
3968 4088 017E          jmp @temp1     ;failed to find it


                ;allocate a reassembly block
                ;return cc e if found, block locked
                ;;gudrq5 ;wants this to return r3 = 1)

                routine reasal,nosave, uses r2-r3
396C 4828 6CB0    lda r2,=messtk
                  repeat
3970 703A 4000      lock r3,reaslk(r2)
3974 9AFE
3976 783A 0003      ldab r3,reasst(r2)
397A 4EB1           break if r3 = =1    ;found a free one.
397C 9108
397E 300A 4000      unlock reaslk(r2)
3982 4A28 0010      add r2,=lrblk
3986 4E28 6E2F      until r2 > =emsstk-1        ;thru reas blks. cc never equal.
398A 8CF3
                  endrepeat
398C 4807         endroutine reasal
```

```
                ;********** back hosts **********

                ;main entrance to back - identical to hi

                        Page LCode      ;MUST be local

398E 7049 81C0  back:   lda r4,mblks(r1)
3992 701C 4000          lda r1,lockhi(r4)
3996 8A03                tz loop        ;check-lock
3998 4008 108E
399C 705C 001C          lda r5,bmessb(r4)
     0001               .lif nz LBig
39A0 7078 00B2            setmap M0,mapddt      ;if code on DDT page, change M0
39A4 3078 FC00
39A8 707C 000C          lda r7,hitt(r4) ;tested at return
39AC 408C 0004          jmp @hilo(r4)   ;dispatch (must match hi wakeup)

                ;get a free buffer and set it up for back

39B0 307C 0008  getfre: sta r7,htemp7(r4)
                        repeat
39B4 4868 02F8            lda sp,=lstack
39B8 4078 13C6            call freget,whhi+nbak
39BC 200C
39BE 8A04                 until succeed
39C0 4078 2B3A            call hiwm
39C4 90F8               endrepeat
39C6 4861               lda r6,r1
39C8 4838 0012          lda r3,=data+2  ;last addr used
39CC 303E 0090          sta r3,bufe(r6)
39D0 4AE4               add r6,=chkh    ;move into packet
39D2 48B0               lda r3,=0       ;for packet beginning
39D4 3036               sta r3,(r6)     ;chkh
39D6 1036               sta r3,(-r6)    ;typh
39D8 1036               sta r3,(-r6)    ;neth
39DA 408C 0008          jmp @htemp7(r4)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 301
IMP.WARM;1      PAGE 61          *** warm page ***

.comnt |
 back5 - send replies

 rstate     rtype=00      rtype=01      rtype=10     rtype=11
idle=00       rfnm1 sent    all8 sent     dead sent    inc sent
request=01    illegal       req8 rcvd     all1 to send all8 to send
message=10    message       gvb rcvd      dead rcvd    inc rcvd
reply=11      rfnm1 to send rfnm8 to send dead to send inc to send
|

                          Page LCode    ;place in local memory for speed

39DE 4AB4        b510:    add r3,=2*words
39E0 4E38 0020            cmp r3,= H8*2*words
39E4 8118                  bne b53
39E6 300D A5B0   b59:     sta r0,rmlock(r5)
39EA 765C 001C   b54:     cmp r5,bmessb(r4)       ;nothing to do?
39EE 8107                 bne b54a        ;if there might be more
39F0 4078 2B3A            jsb r7,hiwm
39F4 705C 001C   back5:   lda r5,bmessb(r4)
39F8 4878 5555            lda r7,= H5555
39FC 4958 0010   b54a:    sub r5,=rmlen
3A00 8B03                 bnm b52
3A02 4858 0370             lda r5,=<rmnum-1>*rmlen
3A06 777D 65BA   b52:     tst r7,rstate(r5)       ;any request/reply?
3A0A 9AF0                  bz b54                 ;nope
3A0C 701D A5B0            lda r1,rmlock(r5)
3A10 9AFE                  bz .-4
3A12 48B0                 lda r3,=0
3A14 701D 65BA   b53:     lda r1,rstate(r5)
3A18 771B 0EB6            tst r1,bittab(r3)
3A1C 9AE1                  bz b510        ;idle or message state
3A1E 702D 65B2            lda r2,rmimp(r5)
3A22 8B04                  bnm b58
3A24 702D A5BA            lda r2,rstate+foo(r5)   ;clear it
3A28 90DF                 br b59
```

```
                ;we have a receive block with something to do

3A2A 4868 0014  b58:    lda r6,=sec3/6  ;1/2 sec max wait
3A2E 306C 000C          sta r6,hitt(r4)
3A32 706B 0EB8          lda r6,bittab+2(r3)
3A36 4F16               tst r1,r6
3A38 8A44                bnz b55        ;reply state
3A3A 701D 65BC          lda r1,rmtype(r5)
3A3E 4F16               tst r1,r6
3A40 9ACF                bz b510       ;not all1 or all8 to send
3A42 356D 65BC          eorm r6,rmtype(r5)
3A46 771B 0EB6          tst r1,bittab(r3)
3A4A 9A13                bz b516

                ; allocate 8 to send

3A4C 706B 0EB6          lda r6,bittab(r3)
3A50 4078 518E          jsb r7,bakset
3A54 1004                regtyp+pribit+rfntyp   ;set up rfnm
3A56 4868 8000          lda r6,=mltpkt  ;this is a multi reply
3A5A 346C 0014          iorm r6,bpkth(r4)
3A5E 7068 12E6  b51d:   lda r6,pkts8
3A62 4078 3B2A          jsb r7,bgeta
3A66 9A14                bz b51e        ;give up after 1/2 second
3A68 703D 65BA          lda r3,rstate(r5)       ;old mesblk's rstate
3A6C 8A49                bnz b514       ;some message past the one we have is in
3A6E 90F8               br b51d ;none is - we must wait for alloc

                ; allocate 1 to send

3A70 756B 0EB6  b516:   eor r6,bittab(r3)
3A74 4078 518E          jsb r7,bakset
3A78 1004                regtyp+pribit+rfntyp   ;set up rfnm
3A7A 48E8               lda r6,=< H800&mesnum>/ H100    ;off by 8 messages
3A7C 3A6C 0012          addbm r6,bseqh(r4)      ;adjust messno accordingly
3A80 48E1       b519:   lda r6,=1
3A82 4078 3B2A          jsb r7,bgeta
3A86 8AFD                bnz b519       ;try for 1/2 second
3A88 48B0               lda r3,=0       ;in this window of 8
3A8A 48E3               lda r6,=3
3A8C 9003               br b51t

3A8E 48B8       b51e:   lda r3,= D8*mesn0/ H100 ;adjust from last to this 8
3A90 48E1               lda r6,=1
3A92 701D A5B0  b51t:   lda r1,rmlock(r5)       ;lock the mes blk
3A96 9AFE                bz .-4
3A98 7A3C 0012          addb r3,bseqh(r4)       ;mes num we were sending
3A9C 793D 65B8          subb r3,rmmess(r5)      ;our offset from mes num
3AA0 4B38 00FF          and r3,= HFF    ;mod 256 arithmetic
3AA4 4EB7               cmp r3,=7
3AA6 8C02                bnl b51f       ;mes num still in range
                          ;  Trap 7741,<;Back can't back up - page 300>
3AA8 9009               br b51g
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 303
IMP.WARM;1      PAGE 63          *** warm page ***

                b51f:     ;  Trap 7740,<;Bk bypass alloc - page 301>
3AAA A2B1                 sll r3,1        ;index to rstate bits (2 per msg)
3AAC A263                 sll r6,r3
3AAE 356D 65BA            eorm r6,rstate(r5)      ;10 -> 01
3AB2 48E2                 lda r6,=2       ;0 -> 10 or 11
3AB4 A263                 sll r6,r3
3AB6 356D 65BC            eorm r6,rmtype(r5)      ;mes to req state
3ABA 300D A5B0  b51g:     sta r0,rmlock(r5)       ;done for now
3ABE 909B                 br back5        ;go to another block

3AC0 756B 0EB6  b55:      eor r6,bittab(r3)
3AC4 4078 518E            jsb r7,bakset
3AC8 1004                  regtyp+pribit+rfntyp   ;set up rfnm
3ACA 771B 0EB8            tst r1,bittab+2(r3)
3ACE 9A07                 bz b57  ;rfnm or rfnm/all
3AD0 48A2                 lda r2,=dedtyp-rfntyp   ;an adjustment for deads
3AD2 771B 0EB6            tst r1,bittab(r3)
3AD6 9A61                 bz b513 ;rfnm/dead
3AD8 48A3                 lda r2,=inrtyp-rfntyp   ;inc adjustment
3ADA 905F                 br b513

3ADC 771B 0EB6  b57:      tst r1,bittab(r3)
3AE0 9A5E                 bz b515 ;rfnm only
3AE2 4838 8000            lda r3,=mltpkt
3AE6 343C 0014            iorm r3,bpkth(r4)
3AEA 7068 12E6  b511:     lda r6,pkts8
3AEE 4078 3B2A            jsb r7,bgeta
3AF2 9A06                  bz b514                ;time's up
3AF4 703D 65BA            lda r3,rstate(r5)       ;looking for anyone in req or reply
3AF8 4F38 5555            tst r3,= H5555
3AFC 9AF7                 bz b511         ;no other reply to piggyback upon

3AFE 703D A5B0  b514:     lda r3,rmlock(r5)
3B02 9AFE                  bz .-4
3B04 703C 0012            lda r3,bseqh(r4)
3B08 4C38 00FF            ior r3,= HFF
3B0C 713D 65B8            sub r3,rmmess(r5)
3B10 A4B8                 sra r3, H8
3B12 4AB8                 add r3,= H8
3B14 9B08                 bm b517 ;window slid past???
3B16 A2B2                 sll r3,2
3B18 703B 0EB6            lda r3,bittab(r3)
3B1C 4D38 FFFF            eor r3,=-1
3B20 333D 65BC            andm r3,rmtype(r5)      ;rf/al sent -> rfnm sent
3B24 300D A5B0  b517:     sta r0,rmlock(r5)
3B28 903A                 br b515
```

```
3B2A 4856       bgeta:  lda r5,r6       ;remember adjustment
3B2C 7028 A3C8          lda r2,nf       ;lock counts
3B30 9AFE                bz .-4
3B32 7038 63DC          lda r3,cntrs+nre        ;how many in ih now?
3B36 8B04               bnm bgeta0      ;over min, must go to nf
3B38 4A36               add r3,r6       ;will we put it over?
3B3A 9B0B               bm bgeta2       ;no, just take nre
3B3C 4853               lda r5,r3       ;how much must come from nf
3B3E 4925       bgeta0: sub r2,r5       ;adjust nf
3B40 7628 63D0          cmp r2,minf     ;enough room?
3B44 8207                bng bgeta1     ;yup, we'll take it
3B46 4A25       b51b:   add r2,r5       ;no, give it back
3B48 3028 A3C8          sta r2,nf
3B4C 4008 2B3A          jmp hiwm

3B50 48D0       bgeta2: lda r5,=0       ;how much nf we took
3B52 3268 63DC  bgeta1: addm r6,cntrs+nre       ;taking nre space
3B56 3268 63D4          addm r6,nal     ;and nal
3B5A 3028 A3C8          sta r2,nf       ;unlock counts
3B5E 4817               lda r1,r7       ;save HIWM return
3B60 4078 396C          call reasal     ;find 1 free reas blk
3B64 910A                be b518        ;got a good one
3B66 4871               lda r7,r1       ;restore return
3B68 7028 A3C8          lda r2,nf       ;lock nf again
3B6C 9AFE                bz .-4
3B6E 3168 63DC          subm r6,cntrs+nre       ;we've remembered adjustments
3B72 3168 63D4          subm r6,nal
                ; we should count this case
3B76 90E8               br b51b ;go return excess and sleep

3B78 386A 0009  b518:   stab r6,ral(r2)
3B7C 705C 001C          lda r5,bmessb(r4)
3B80 305A 0004          sta r5,rid(r2)
3B84 701D 65B8          lda r1,rmmess(r5)
3B88 381A 0007          stab r1,ruse(r2)
3B8C 4893               lda r1,=3               ;set to 'no-name'
3B8E 301A 0002          sta r1,rsf(r2)          ;;reasst ;and clear rsf
3B92 300A 4000          sta r0,reaslk(r2)
3B96 48A1               lda r2,=rfatyp-rfntyp   ;adjust for rfnm.allocate
3B98 322C 0014  b513:   addm r2,bpkth(r4)
3B9C 4078 5022  b515:   jsb r7,bsend
3BA0 4008 39F4          jmp back5
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 305
IMP.WARM;1      PAGE 65          *** warm page ***


               ; back6 - incompletes

     0001              .if nz LBig     ;if assembling with big local
FC00 0200                page DDTCode  ;place in DDTCode
                         .iff
                          page Warm     ;else place in Warm
                         .endc

4F54 A491       b67:     sra r1,1
4F56 7A1D 6938           addb r1,tmmess(r5)
4F5A 4991                sub r1,=1
4F5C A298                sll r1, D8
4F5E 4821                lda r2,r1       ;r2 is argument to bfixt
4F60 A6D4                srl r5,4        ;mes blk offset to rh
4F62 4C15                ior r1,r5       ;so: mes num,tr mes blk
4F64 A2D4                sll r5,4
4F66 4878 0088           lda r7,=ttledr}ttresv   ;these must be 0
4F6A 4838 8910           lda r3,=trnblk
4F6E 7F7B 000D  b6a:     tstb r7,trstat(r3)      ;any bad bits?
4F72 8A04                 bnz b6d        ;yes, pass over this block
4F74 761B 0004           cmp r1,trhstl(r3)
4F78 910B                 be b6b ;found right trnblk
4F7A 4A38 0010  b6d:     add r3,=trnl
4F7E 4E38 8F50           cmp r3,=trnblk+trntot
4F82 81F6                 bne b6a        ;loses if long wait for 2nd trnblk
4F84 E647                Trap 3107,<;No trnblk for inc query - page 303>
4F86 4078 51D8           jsb r7,bfixt
4F8A 5408                 qertyp+pribit+alloc+inqtyp
4F8C 901C                br b68

4F8E 48F9       b6b:     setb trtypl+1(r3) = #cinctr     ;mark block as incomplete
4F90 387B 0003
4F94 701B 0008           lda r1,trmidl(r3)
4F98 4C9F                ior r1,=subtyp
4F9A 499C                sub r1,=subtyp-clost
4F9C 301B 0008           sta r1,trmidl(r3)
4FA0 787B 000D           ldab r7,trstat(r3)
4FA4 787B 000D           if byte trstat(r3) .bit. =ttgvba        ;used an alloc
4FA8 4F78 0040
4FAC 9A09
4FAE 4078 51D8             call bfixt, qertyp+pribit+alloc+inqtyp
4FB2 5408
4FB4 4838 8000             lda r3,=mltpkt
4FB8 343C 0014             iorm r3,bpkth(r4)            ;return alloc
4FBC 9004                 else
4FBE 4078 51D8             call bfixt,  qertyp+pribit+inqtyp
4FC2 5008
                         endif
4FC4 707D 65B2  b68:     lda r7,rmimp(r5)                ;remote imp#
4FC8 706D 65B4           lda r6,rmhost(r5)               ;remote,,local hosts
4FCC E7C7                Trap 3707,<;Sending inc query - page 303>
4FCE 4078 5022           jsb r7,bsend
```

```
4FD2 4858 0370  back6:  lda r5,=<tmnum-1>*tmlen
4FD6 4818 C300          lda r1,=tmrset+tminit+msto
4FDA 771D 693A  b62:    tst r1,tstate(r5)
4FDE 8A1C               bnz b64
4FE0 305C 001C          sta r5,bmessb(r4)
4FE4 4078 2B30          jsb r7,hipok
4FE8 4818 0100          lda r1,=msto0   ;about 13 seconds
4FEC 703D A930          lda r3,tmlock(r5)
4FF0 9AFE               bz .-4
4FF2 703D 693A          lda r3,tstate(r5)
4FF6 4F38 C300          tst r3,=tmrset+tminit+msto      ;active and timed out?
4FFA 8A0C               bnz b66
4FFC 341D 693A          iorm r1,tstate(r5)
5000 707D 6932          lda r7,tmimp(r5)
5004 9B07               bm b66
5006 4818 FFF0          lda r1,=- D8*2
500A 6739 0EC6  b65:    tst r3,bittab+< D8*2>(r1)+
500E 9AA3               bz b67
5010 88FD               bnlp b65
5012 300D A930  b66:    sta r0,tmlock(r5)
5016 4958 0010  b64:    sub r5,=tmlen
501A 8BE0               bnm b62
501C 4078 2B3A          jsb r7,hiwm
5020 90D9               br back6
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 307
IMP.WARM;1      PAGE 67          *** warm page ***


5022 307C 000A  bsend:  sta r7,hisav7(r4)
5026 4078 39B0          jsb r7,getfre
502A 4878 0010          lda r7,=data    ;packet length
502E 307E 0090          sta r7,bufe(r6) ;into packet
5032 48F9               lda r7,=<data+2>/words  ;total words in packet
5034 705C 0010          lda r5,btyph(r4)        ;new type header
5038 4975               sub r7,r5       ;in checksum
503A 305E 0002          sta r5,typh(r6) ;into type header
503E 7058 62E2          lda r5,mine     ;I'm source
5042 305E 0006          sta r5,srch(r6)
5046 4975               sub r7,r5       ;in checksum
5048 485C 0012          lda r5,=bseqh(r4)
504C 4AE8               add r6,=seqh    ;where to start copy
                        repeat
504E 6035                 lda r3,(r5)+
5050 2036                 sta r3,(r6)+
5052 4973                 sub r7,r3
5054 4E5C 001C            until r5 = =bdata+2(r4)
5058 81FB
                        endrepeat
505A 4968 0012          sub r6,=data+2  ;put pointer back to pkt beginning
505E 307E 0004          sta r7,chkh(r6) ;save new checksum
5062 4078 2B88          jsb r7,gvtskc
5066 9A04               ifnot zero      ;still have buffer.
5068 4078 1314            call oflush,whhi      ;release buffer
506C 2000
                        endif
506E 408C 000A          jmp @hisav7(r4)
```

```
                ; back7 - send resets
5072 4858 0370  back7:  lda r5,=<tmnum-1>*tmlen
5076 48E0               lda r6,=0
5078 703D 6932  b71:    lda r3,tmimp(r5)
507C 743D 693A          ior r3,tstate+<0&tmrset>(r5)
5080 8B02               bnm .+4
5082 4AE1                add r6,=1
5084 4958 0010          sub r5,=tmlen
5088 8BF8                bnm b71
508A 7828 64D3          ldab r2,tclip+1
508E 4078 5176          jsb r7,b7sub
5092 3028 64D2          sta r2,tclip
5096 4078 2B30          jsb r7,hipok

509A 4858 0370          lda r5,=<tmnum-1>*tmlen
509E 703D A930  b72:    lda r3,tmlock(r5)
50A2 9AFE                bz .-4
50A4 703D 6932          lda r3,tmimp(r5)
50A8 9B23                bm b73
50AA 702D 6938          lda r2,tmmess(r5)
50AE 4BAF               and r2,=age
50B0 703D 693A          lda r3,tstate(r5)
50B4 4F38 C000          tst r3,=tmrset+tminit
50B8 9A04                bz b702        ;block is open
50BA 4EAF               cmp r2,=age     ;reach max age yet?
50BC 910A                be b74         ;yes, reset
50BE 9018               br b73

50C0 7628 64D2  b702:   cmp r2,tclip
50C4 9215                bg b73 ;not old enuf to reset
50C6 4D38 00FF          eor r3,= HFF
50CA 4F38 3CFF          tst r3,=tmall+ HFF
50CE 8A10                bnz b73        ;wait for cleanup
50D0 4828 8000  b74:    lda r2,=tmrset
50D4 302D 693A          sta r2,tstate(r5)
50D8 702D 6938          lda r2,tmmess(r5)
50DC 4CAF               ior r2,=age     ;set age to 4
50DE 49AB               sub r2,=age-<age0*4>
50E0 302D 6938          sta r2,tmmess(r5)
50E4 48A0               lda r2,=0       ;no mes num for reset
50E6 4078 51D8          jsb r7,bfixt
50EA 500A                qertyp+pribit+rsbtyp   ;reset-the-block message
50EC 9035               br b79

50EE 300D A930  b73:    sta r0,tmlock(r5)
50F2 4958 0010          sub r5,=tmlen
50F6 8BD4                bnm b72

                ;more back7 over leaf
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 309
IMP.WARM;1      PAGE 69          *** warm page ***

                ; need to count outstanding reset requests

50F8 4078 2B30          jsb r7,hipok
50FC 4858 0370          lda r5,=<rmnum-1>*rmlen
5100 48E0               lda r6,=0
5102 703D 65B2  b75:    lda r3,rmimp(r5)
5106 8B02               bnm .+4
5108 4AE1                add r6,=1
510A 4958 0010          sub r5,=rmlen
510E 8BFA               bnm b75
5110 7828 64D1          ldab r2,rclip+1
5114 4078 5176          jsb r7,b7sub
5118 3028 64D0          sta r2,rclip
511C 4078 2B30          jsb r7,hipok
5120 4858 0370          lda r5,=<rmnum-1>*rmlen
5124 703D A5B0  b76:    lda r3,rmlock(r5)
5128 9AFE                bz .-4
512A 703D 65B2          lda r3,rmimp(r5)
512E 9B17               bm b77
5130 702D 65B8          lda r2,rmmess(r5)
5134 4BAF               and r2,=age
5136 7628 64D0          cmp r2,rclip
513A 9211               bg b77
513C 702D 65B8          lda r2,rmmess(r5)
5140 4B28 FFF0          and r2,=-1?age
5144 4CA4               ior r2,=4
5146 302D 65B8          sta r2,rmmess(r5)
514A 707C 4012          lda r7,bseqh+foo(r4)
514E 4825               lda r2,r5
5150 4078 51E2          jsb r7,bfixr
5154 500E                qertyp+pribit+rrqtyp   ;reset-request message
5156 4078 5022  b79:    jsb r7,bsend
515A 908C       back7j: br back7

515C 300D A5B0  b77:    sta r0,rmlock(r5)
5160 4958 0010          sub r5,=rmlen
5164 8BE0                bnm b76
5166 4818 0019          lda r1,= D25    ;wait awhile
516A 301C 000C          sta r1,hitt(r4)
516E 4078 2B3A          jsb r7,hiwm
5172 8AFE                bnz .-4
5174 90F3               br back7j
```

```
                ;Some BACK subroutines

                ;decide how old a block must be to be reset
                ;if r6<6, r2=max(r2-1,4), else r2=min(r2+1,age)

5176 4EE6       b7sub:  cmp r6,=6       ;want this many blocks
5178 9206                bg b7sub1
517A 4AA1               add r2,=1*age0
517C 4EAF               cmp r2,=age
517E 8C02               bnl .+4
5180 48AF                lda r2,=age
5182 4007               jmp (r7)

5184 49A1       b7sub1: sub r2,=1*age0
5186 4EA4               cmp r2,=4*age0
5188 8202               bng .+4
518A 48A4                lda r2,=4*age0
518C 4007               jmp (r7)
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 311
IMP.WARM;1      PAGE 71          *** warm page ***

                ;set up a reply (rfnm, may become rf/al, dead, or inc reply)
                ;r3/index to bittab entry for rstate/rmtype, r5/rec mes blk
                ;r6/bits to change rstate by (eor), r7/return
                ;set up dead status,mess num, go to bfixc. return r1/rmtype

518E 356D 65BA  bakset: eorm r6,rstate(r5)
5192 782D 65BF          ldab r2,rmlhn(r5)       ;local host
5196 48E3               lda r6,=3       ;default host status if none
5198 702A 6326          lda r2,h2pblk(r2)
519C 9B0A               ifnot minus
519E 706A 002C            lda r6,deadsc(r2)
51A2 8A07                 if zero
51A4 786A 001F              ldab r6,hihd(r2)
51A8 8A02                   if zero
51AA 48EF                     lda r6,=chcup     ;just now coming up
                            endif
51AC 4C68 FFE0              ior r6,=chnoti      ;unknown time
                          endif
                        endif
51B0 306C 001A          sta r6,bdata(r4)        ;dead host status in data
51B4 702D 65B8          lda r2,rmmess(r5)
51B8 4B28 FFF0          and r2,=-1?age
51BC 4CA4               ior r2,=4
51BE 302D 65B8          sta r2,rmmess(r5)
51C2 4863               lda r6,r3       ;r3: bittab index
51C4 A2E6               sll r6,6
51C6 4A2E F800          add r2,=- H800(r6)      ;adjust to previous window
51CA 302C 0012          sta r2,bseqh(r4)
51CE 702C 4018          lda r2,bmidh+foo(r4)    ;clear it
51D2 701D 65BC          lda r1,rmtype(r5)
51D6 900E               br bfixc
```

```
                ;set up a back message
                ;r2/message number (left half)
                ;in line arg: pkt type, flags for bfixt in lh, pkt code in rh

51D8 302C 0012  bfixt:  sta r2,bseqh(r4)        ;hit message number
51DC 4825               lda r2,r5
51DE 4A58 0380          add r5,=tmlock-rmlock
51E2 A6A4       bfixr:  srl r2,4                ;entry point
51E4 382C 0019          stab r2,bmidh+1(r4)     ;my block
51E8 782D 65B9          ldab r2,rmmess+1(r5)    ;;tmmess
51EC A6A4               srl r2,4        ;;luse  ;luse to low end
51EE 382C 0018          stab r2,bmidh(r4)
51F2 6027       bfixc:  lda r2,(r7)+    ;now get arg
51F4 382C 0015          stab r2,bpkth+1(r4)     ;just pkt code
51F8 4B28 FF00          and r2,=<-1>?akbits     ;clear low half
51FC 302C 0010          sta r2,btyph(r4)        ;for type header
5200 702D 65B2          lda r2,rmimp(r5)        ;;tmimp
5204 302C 0016          sta r2,bdsth(r4)
5208 782D 65B7          ldab r2,rmctl+1(r5)     ;;tmctl :where for. blk # lives
520C 382C 0013          stab r2,bseqh+1(r4)     ;;fblock;into our seqh
5210 782D 65B6          ldab r2,rmctl(r5)       ;;tmctl ;now for foreign use
5214 4BAF               and r2,=fuse/ H100              ;just the use no.
5216 382C 0014          stab r2,bpkth(r4)               ;into our params
521A 300D A5B0          sta r0,rmlock(r5)       ;;tmlock
521E 305C 001C          sta r5,bmessb(r4)
5222 4007               jmp (r7)
```

```
                ; back9 - send give backs


5224 4F68 003F  b95:    tst r6,=tmalt   ;reduce alloc time if nz
5228 9A04               bz b96
522A 49E1                sub r6,=tmalt0
522C 386D 693E           stab r6,tmstp(r5)
5230 300D A930  b96:    sta r0,tmlock(r5)
5234 4958 0010  b97:    sub r5,=tmlen
5238 8B05                bnm b94
523A 4078 2B3A          jsb r7,hiwm
523E 4858 0370  back9:  lda r5,=<tmnum-1>*tmlen
5242 4878 3C00  b94:    lda r7,=tmall
5246 777D 693A          tst r7,tstate(r5)       ;any allocates?
524A 9AF5                bz b97                 ;no, try next block
524C 703D A930          lda r3,tmlock(r5)       ;yes, get into block
5250 9AFE                bz .-4
5252 703D 6932          lda r3,tmimp(r5)        ;any dest?
5256 8B03               if minus                ;nope
5258 703D A93A            lda r3,tstate+foo(r5) ;block is free
                        endif
525C 786D 693E          ldab r6,tmstp(r5)       ;;tmalt
5260 703D 693A          lda r3,tstate(r5)
5264 89E0                bno b95        ;can't get a messno
5266 4B37               and r3,r7       ;;=tmall
5268 9AE4                bz b96         ;alloc was just used
526A 702D 6938          lda r2,tmmess(r5)
526E 4BAF               and r2,=age
5270 7628 64D2          cmp r2,tclip    ;see back7
5274 8204                bng b98        ;block is old
5276 4F68 003F          tst r6,=tmalt
     0001       .if nz PSE
527A 8AD5                bnz b95
                .iff
                         bz b98         ;alloc is old
                        cmp r3,=2*tmall0        ;more than 2? (was 1)
                         bnl b95        ;nope, don't give back any
                .endc
```

```
527C 48B4       b98:    lda r3,=tmall0_- H8     ;reduce allocate count
527E 393D 693A          subbm r3,tstate(r5)
5282 783D 693B          ldab r3,tstate+1(r5)    ;change message bits
5286 A6B1               srl r3,1
5288 383D 693B          stab r3,tstate+1(r5)
528C 702D 6938          lda r2,tmmess(r5)
5290 483A 0100          lda r3,=mess0(r2)       ;increment messno
5294 303D 6938          sta r3,tmmess(r5)       ;restore
5298 4078 51D8          call bfixt, regtyp+pribit+gvbtyp        ;giveback message
529C 1002
529E 703C 4018          lda r3,bmidh+foo(r4)    ;no subtype for no.raw
52A2 4838 8000          lda r3,=mltpkt
52A6 343C 0014          iorm r3,bpkth(r4)       ;givebacks are multi-packet
52AA 4078 3BA4          call ltrnput    ;set up a trnblk for give backs
52AE 702D 65B8  ltrtn:  lda r2,rmmess+<0*tmmess>(r5)    ;for local use
52B2 302B 000A          sta r2,trluse(r3)       ;into trn blk
52B6 702C 0012          lda r2,bseqh(r4)        ;message #
52BA 4B28 FF00          and r2,=mesnum
52BE A6D4               srl r5,4        ;block #
52C0 4C2D FFC8          ior r2,=<<rmlock-tmlock>/ D16>(r5)      ;recover tm blk number
52C4 302B 0004          sta r2,trhstl(r3)
52C8 4828 4040          lda r2,= H4000+ttgvba   ;used an allocate
52CC 302B 000C          sta r2,trntim+<0*trstat>(r3)    ;release trnblk
52D0 4078 5022          jsb r7,bsend
52D4 4958 0380          sub r5,=tmlock-rmlock   ;undo bfixt adjustment
52D8 90AE               br b97

     0001       .if z LBig              ;if code is on Warm
                  ltrnput = trnput      ;then page swapping in not necessary
                .iff
                  page LCode
                  ltrnput:              ;"routine" ltrnput
3BA4 7078 00B4      setmap m0,mapcod    ;swap from DDT to Warm
3BA8 3078 FC00
3BAC 4078 5AE2      call trnput         ;trnput is on Warm page
3BB0 7078 00B2      setmap m0,mapddt    ;and swap back
3BB4 3078 FC00
3BB8 4008 52AE    jmp ltrtn             ;return to main code
                .endc
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 315
IMP.WARM;1      PAGE 75          *** warm page ***


FC00 0200                page DDTCode

                ;dedrea: scans the reassembly blocks and frees up any associated
                ;with a dead receive message block

52DA 4828 6CB0  dedrea: lda r2,=messtk  ;set to search reas blk table
52DE 706A 4000  ded46:  lda r6,reaslk(r2)
52E2 9AFE                bz .-4
52E4 781A 0003           ldab r1,reasst(r2)      ;legal states are 1,2,3
52E8 9A25                bz ded44
52EA 4F18 FFFC           tst r1,= HFFFC
52EE 8A22                bnz ded44
52F0 4F92                tst r1,=2       ;block in use?
52F2 9A17                bz ded41        ;no: ignore free blocks
52F4 703A 0004           lda r3,rid(r2)
52F8 706B 65B8           lda r6,rmmess(r3)
52FC 7D6A 0007           eorb r6,ruse(r2)
5300 4B68 00F0           and r6,=luse
5304 8A17                bnz ded44       ;old use- not ours anymore)
5306 705B 65B2           lda r5,rmimp(r3)
530A 9B14                bm ded44        ;reset block
530C 4E92                cmp r1,=2       ;got a message started?
530E 8109                bne ded41       ;no
5310 786A 0006           ldab r6,remess(r2)      ;reas blk mess num
5314 796B 65B8           subb r6,rmmess(r3)      ;less next to give host
5318 4B68 00FF           and r6,= HFF    ; ... mod 256
531C 4EE7                cmp r6,=7       ;should be 0-7
531E 9C0A                bl ded44        ;nope, lose
5320 300A 4000  ded41:  sta r0,reaslk(r2)
5324 4A28 0010  ded47:  add r2,=lrblk
5328 4E28 6E30           cmp r2,=emsstk
532C 92D9                bg ded46        ;more to do
532E 4008 1BEE           jmp tossx

                ded44:  ;  Trap 3202,<;Recovd an old reas block - page 313>
5332 4078 3868           call lreasf
5336 90F7                br ded47

FC00 0200          dstolist dedrea
4EDC 52DA
4EDE 1CBC
4EE0 0000


     0001       .if nz Lbig
     3868          lreasf = reasf
                .iff
                  page Lcode
                  lreasf:
                     setmap m0,mapcod
                     call reasf
                     setmap m0,mapddt
                  jmp ded47
                .endc
FC00 0400                page Warm
```

```
                ;warm page timeout dispatch table

5C9C 0000       codtab: 0       ;no config this page
     0000       .if z $mapcd-$maprt             ;routing is on this page
5C9E 5970               rupqck
                .endc ;;z $mapcd-$maprt
5CA0 5CAA               conchk
5CA2 3EBE               gvharel
5CA4 07D6               rsucceed        ;;jed
5CA6 07D6               rsucceed        ;;htest
5CA8 0000               0

FC00 0400               page Warm

                ;checksum configuration table

5CAA 1076       routine conchk, uses r3, uses r5
5CAC 48B0         lda r3,=0
5CAE 4858 40C0    lda r5,=hacmem        ;init for sub chain
5CB2 4078 1856    call subchn-conlen    ;checksum it
5CB6 3038 6554    sta r3,concer         ;remember error to ease patching
5CBA 9A06         ifnot zero            ;checksum bad.
5CBC 4838 1000      lda r3,=hdcbit      ;it's not okay
5CC0 3438 62AE      iorm r3,anom        ;remember in anom
5CC4 9005         else
5CC6 4838 EFFF      lda r3,=<-1>?hdcbit
5CCA 3338 62AE      andm r3,anom        ;clear bit in anom
                  endif
5CCE 6006       endroutine
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 317
IMP.WARM;1      PAGE 77          *** warm page ***

                ;give all hosts wakeups every 25 milliseconds
                ; ihtt counts down waiting for hardware - hitt counts up
                ;     on hardware and down on software


                page LCode
3BBC 1076       routine ihtc,uses r1-r5,arg r1
3BBE 7078 00B2    setmap m0,mapddt
3BC2 3078 FC00
3BC6 4008 5338    jmp jihtc
FC00 0200         page ddtcode
                  jihtc:
5338 7058 6322    lda r5,hosts                  ;limit of wakeup index
533C 4078 53D6    call chktc,m1#tcgo            ;get an index to do now
5340 64D4
5342 704A 6326    lda r4,h2pblk(r2)             ;access host block
5346 9B1C         ifnot minus                   ;not inactive
5348 705C 0020      lda r5,iobloc(r4)
534C 707C 404E      lock ihloc(r4)
5350 9AFE
5352 707C 0052      lda r7,ihtt(r4)             ;first, count IH timer
5356 4B78 07FF      and r7,= H7FF               ;keep it reasonable
535A 49F1           sub r7,=1
535C 307C 0052      sta r7,ihtt(r4)
5360 8A03           if zero                     ;time to reset ih
5362 307C 0054        sta r7,ihgoin(r4)
                    endif
5366 707C 0002      lda r7,hostyp(r4)           ;do fast host timeouts
536A 40FF 5382      call @hftable(r7)           ;special for fakes
536E 300C 404E      unlock ihloc(r4)
5372 787C 000E      ldab r7,hotpid(r4)          ;poke host output
5376 30F8 00AC      sta r7,@pid
537A 4078 538E      call hitc                   ;do common BACK/HI checks
                  endif
537E 4008 3BCA    jmp xihtc
                  page LCode
                  xihtc:
3BCA 7078 00B4    setmap m0,mapcod
3BCE 3078 FC00
3BD2 6006       endroutine ihtc

FC00 0200       Page DDTCode

                ;fast timeout host checking routines
5382            table hftable
5382 07D6         rsucceed                      ;reals need none
5384 5414         fasfak                        ;fakes get a little help
5386 07D6         rsucceed                      ;(VDH: VDHF)
5388 07D6         rsucceed                      ;TIPs: none
538A 07D6         rsucceed                      ;spare
538C 07D6         rsucceed                      ;spare
                endtable hftable
```

```
                ;HITC
                ;perform checks for HI/BACK at 25 ms timeout

538E 1076       routine hitc,arg r4-r5,uses r3
5390 48B1         lda r3,=1
5392 707C 4000    lock lockhi(r4)
5396 9AFE
5398 707C 000C    lda r7,hitt(r4)
539C 9A10         ifnot zero
539E 9B04           ifnot minus                 ;waiting for imp; don't reset.
53A0 313C 000C        subm r3,hitt(r4)
53A4 900C           else                        ;waiting for host
53A6 323C 000C        addm r3,hitt(r4)
53AA 8A09             if zero                   ;reset after 30 seconds.
53AC 48F8               lda r7,=hirset          ;note reset happened
53AE 3C7C 001E          iorbm r7,hibits(r4)
53B2 707C 0002          lda r7,hostyp(r4)       ;;VDH :what type host
53B6 9B03               ifnot minus ;;forbak ;no back hosts please
53B8 40FF 53CA            call @hrtable(r7)     ;do the right reset
                        endif
                      endif
                    endif
                  endif
53BC 300C 4000    unlock lockhi(r4)
53C0 787C 000F    ldab r7,hinpid(r4)
53C4 30F8 00AC    sta r7,@pid
53C8 6006       endroutine hitc

                ;routines to reset hosts
53CA            table hrtable
53CA 541A         resreal                       ;reset real host hardware
53CC 07D6         rsucceed                      ;fakes - do nothing
53CE 07D6         rsucceed                      ;(VDH: VDHITO)
53D0 07D6         rsucceed                      ;TIPs: none
53D2 07D6         rsucceed                      ;spare
53D4 07D6         rsucceed                      ;spare
                endtable hrtable
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 319
IMP.WARM;1      PAGE 79          *** warm page ***


                ;CHKTC
                ;Compute a new index for IHTC/BTC to use
                ;called with R1/ PID just fielded (IHTPID/BTCPID),
                ;inline R4/ pointer to TCGO/TBKGO, R5/ limit for R4

53D6 6047       routine chktc,arg r1/r5,inline r4,uses r2-r5,result r2,nosave
53D8 702C 0002    lda r2,words(r4)              ;first, validate limit word
53DC 9B03         if minus } r2 > r5            ;troubles
53DE 4E25
53E0 8C03
53E2 305C 0002      sta r5,words(r4)            ;force a good value
                  endif
53E6 702C 4000    lock r2,m3-m1(r4)             ;get proper index
53EA 9AFE
53EC 49A2         sub r2,=2                     ;next to check
53EE 4832         lda r3,r2
53F0 9B04         if minus } zero } r3 > r5     ;out of range?
53F2 9A03
53F4 4E35
53F6 8C02
53F8 4835           lda r3,r5                   ;set it high again
                  endif
53FA 763C 0002    if r3 = words(r4)             ;end of a cycle
53FE 8107
5400 49B2           sub r3,=words               ;update limit
5402 8A02           if zero                     ;reacjed bottom end
5404 4835             lda r3,r5                 ;use highest index
                    endif
5406 303C 0002      sta r3,words(r4)            ;new cycle limit
540A 9003         else
540C 3098 00AC      sta r1,@pid                 ;poke ourselves
                  endif
5410 3034         unlock r3,(r4)                ;and new current index
5412 4807       endroutine chktc

                ;routine to run at fast timeout for fakes

                routine fasfak,nosave,arg r4,uses r1
5414 781C 401F    ldab r1,hihd+foo(r4)          ;fakes are always up
5418 4807       endroutine fasfak

                ;routine to reset real hosts

                routine resreal,nosave,uses r1
541A 4818 0100    lda r1,=hreset+<<data& HF>/2> ;;trledr
541E 301D 0006    sta r1,statih(r5)             ;reset a real host
5422 4807       endroutine resreal
```

```
                ;BTC
                ;Check back hosts every 25 ms
                ;Calls HITC for most of its work

FC00 0200       page DDTCode

5424 7038       b2pblk: bbk0,bbk1,bbk2,bbk3             ;back hosts
5426 7058
5428 7078
542A 7098
542C            b2pend: .blkw 0

                page LCode
3BD4 1076       routine btc,arg r1,uses r1-r5
3BD6 7078 00B2    setmap m0,mapddt
3BDA 3078 FC00
3BDE 4008 542C    jmp jbtc
FC00 0200         page ddtcode
                  jbtc:
542C 48D8         lda r5,=b2pend-b2pblk          ;limit for our index
542E 4078 53D6    call chktc,m1#tbkgo            ;get next offset to check
5432 64D8
5434 704A 5424    lda r4,b2pblk(r2)              ;point to back host
5438 4078 538E    call hitc                      ;and do the checks
543C 4008 3BE2    jmp xbtc
                  page LCode
                  xbtc:
3BE2 7078 00B4    setmap m0,mapcode              ;set map for loop
3BE6 3078 FC00
3BEA 6006       endroutine btc
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 321
IMP.WARM;1      PAGE 81          *** warm page ***

.comnt |

    We get here from the pid dispatch when it discovers the
  stkpid value.  The pid value is in r1.  The next pid is removed
  from the ring buffer, and written to the pid.
|
FC00 0400               page Warm

5CD0 1076       routine rstgo, arg r1, uses r2
5CD2 7078 A4B2    lock ringlk           ;lock the ring buf.
5CD6 9AFE
5CD8 7078 64B4      lda r7,ringc        ;number of entries in the buffer.
5CDC 9A13           ifnot zero
5CDE 49F1             sub r7,=1         ;decrease number of entries.
5CE0 3078 64B4        sta r7,ringc
5CE4 9A03             ifnot zero        ;entries left in ring buffer.
5CE6 3098 00AC          sta r1,@pid     ;poke the stkpid.
                      endif
5CEA 7078 64B6        lda r7,ringf      ;address of next entry in ring.
5CEE 6827             ldab r2,(r7)+     ;get the next pid from ring...
5CF0 30A8 00AC        sta r2,@pid       ;...and poke the pid with it.
5CF4 4E78 64D0        if r7 >= =ringe   ;past last word of buffer.
5CF8 9203
5CFA 4878 64B8          lda r7,=ring    ;yes, re-initialize pointer.
                      endif
5CFE 3078 64B6        sta r7,ringf      ;update the next entry pointer.
                    endif
5D02 3008 A4B2    unlock ringlk         ;unlock the ring.
5D06 6006       endroutine rstgo
```

```
                ihpack:
                repeat
5D08 701C 0056    set r1 = ihwq(r4)             ;queue to use
5D0C 4078 154A    call deque,whih               ;SP okay from IHDB call after leader
5D10 0040
5D12 9A3E           bfail iher2                 ;end of queue before EOM?
5D14 300C 405C    unlock lockih(r4)             ;done queue lock
5D18 302C 0068    set ihbuff(r4) = r2           ;remember buffer we're sending
5D1C 4831         set r3 = r1
                ihcump:                 ;cum stats hook.
5D1E 7079 009C    set ihbufb(r4) = bufb(r1)     ;save stuff from buffer
5D22 307C 006A
5D26 7079 0008    set ihseqh(r4) = seqh(r1)     ;only important on last buffer
5D2A 307C 006E
5D2E 7079 000A    set ihpkth(r4),r7 = pkth(r1)  ;returning PKTH in R7
5D32 307C 006C
5D36 721B 0090    add r1,bufe(r3)               ;bufe has length in bytes
5D3A A2F1         sll r7,1      ;;lstpkt        ;put lstpkt bit in sign
5D3C 9B06         ifnot minus   ;;hendf         ;not last packet.
5D3E 707B 009C      set r7 = bufb(r3)           ;mes blk?
5D42 9903           ifnot odd                   ;not raw packet, thus no eom bit.
5D44 4B18 7FFF        and r1,=-1?hendf          ;clear eom bit
                    endif
                  endif
5D48 7078 00D2    setmap m2,mapv2               ;now get buff start
5D4C 3078 FC04
5D50 707A FCB0    set r7 = point-chain(r2)
5D54 4AF1         add r7,#data/ H10             ;data is 10} into buffer
5D56 307D 0008    set starto(r5) = r7           ;do reliability check in a little while
5D5A 4B38 1FFE    and r3,#packm                 ;prepare for FSIOUT
5D5E 482B 8010    set r2 = #m2+data(r3)         ;get to buffer start
5D62 4837         set r3 = r7                   ;proper STARTO value
5D64 707C 0002    set r7 = hostyp(r4)           ;now by host type
5D68 40FF 12CA    call @hotable(r7)             ;do output stuff
5D6C 301D 000A    set endo(r5) = r1             ;start output
5D70 301C 0058    set ihlstp(r4) = r1           ;remember end flag
5D74 4078 191C    call ihdb                     ;wait for output of packet
5D78 702C 4068    set r2 = ihbuff+foo(r4)       ;get packet just sent
5D7C 4078 1512    call unpack                   ;map it in for IHDONE, trace
                trcih:                  ;trace hook
5D80 707C 0058    set r7 = ihlstp(r4)
5D84 9B29           bminus ihdone               ;done last packet
5D86 4078 131E    call flushb,whih              ;flush pkts 1 - n-1
5D8A 0040
5D8C 90BE       endrepeat
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 323
IMP.WARM;1      PAGE 83          *** warm page ***


                ;various IH error handlers

5D8E E6E8       iher2:  Trap 3350,<:Bad ih queue struct - page 321>
5D90 9002               br iherrc
5D92 E6C2       iherr:  Trap 3302,<:Bad buffer on host queue - page 321>
5D94 703C 0056  iherrc: set r3 = ihwq(r4)       ;recover queue
5D98 48F1               set (r3) = =1
5D9A 3073
5D9C 303B 0002          sta r3,ehq-shq(r3)      ;;ehpq
5DA0 300C 405C  iher3:  unlock lockih(r4)
5DA4 905C               br ihex2

                ;routine to check real host output start pointer write

5DA6 1076       routine hsiout,arg r3/r5

5DA8 707D 000A    lda r7,endo(r5)
5DAC A6F4         srl r7,4
5DAE 4D73         eor r7,r3                     ;compare old STARTO
5DB0 4B78 01FF    and r7,=packm_-4              ;bits to be checked
5DB4 9A04         ifnot zero
5DB6 E60B           Trap 3013,<:Start ptr write failed - page 321>
5DB8 303D 0008      sta r3,starto(r5)           ;retry just once
                  endif

5DBC 6006       endroutine hsiout

                ;and one for fakes

5DBE 1076       routine fsiout,arg r2/r5

5DC0 302D 0012    sta r2,fakeso(r5)
5DC4 4878 2000    lda r7,=hbusy
5DC8 347D 000C    iorm r7,statoh(r5)
5DCC 787D 000E    ldab r7,hbpid(r5)
5DD0 30F8 00AC    sta r7,@pid

5DD4 6006       endroutine fsiout
```

```
5DD6 300C 405C  ihdone: sta r0,lockih(r4)
5DDA 705C 0016          set r5 = ihledr+dstl(r4)        ;used saved leader
5DDE 3018 017E          sta r1,temp1
5DE2 781C 006D          setb r1 = ihpkth+1(r4)  ;used copy of PKTH
5DE6 703C 001A          lda r3,ihledr+len1(r4)
5DEA A6B4               srl r3,4
5DEC 4868 02F8          lda sp,=lstack
5DF0 4078 5B24          call hothru,htpmfn      ;count host throughput
5DF4 003C
5DF6 7068 017E          lda r6,temp1
5DFA 705C 006A          set r5 = ihbufb(r4)     ;use saved BUFB
5DFE 992C                bo ihex        ;raw pkt: no rfnm
5E00 701C 006C          set r1 = ihpkth(r4)     ;use saved PKTH
5E04 A694               srl r1,4        ;position our use number
5E06 707D A5B0          lda r7,rmlock(r5)
5E0A 9AFE                bz .-4
5E0C 751D 65B8          eor r1,rmmess(r5)
5E10 4B18 00F0          and r1,=luse
5E14 8A2A               bnz ihr2        ;wrong luse, but 'normal'
5E16 787C 006E          setb r7 = ihseqh(r4)    ;use saved SEQH
5E1A 4AF8               add r7,= D8     ;from last 8
5E1C 797D 65B8          subb r7,rmmess(r5)
5E20 4B78 00FF          and r7,= HFF    ; mod 256
5E24 4EF7               cmp r7,=7
5E26 9C21                bl ihr2        ;mes num out of range
5E28 A2F2               sll r7,2
5E2A 703F 0EB6          lda r3,bittab(r7)
5E2E 343D 65BA          iorm r3,rstate(r5)      ;reply now
5E32 701C 006C          set r1 = ihpkth(r4)     ;use saved PKTH
5E36 8B03               if minus
5E38 343D 65BC            iorm r3,rmtype(r5)    ;mark rf+al
                        endif
5E3C 4B9F               and r1,=pktcod          ;type of this buffer
5E3E 4E93               if r1 = #inmtyp         ;is it an incomplete?
5E40 8105
5E42 743F 0EB8            ior r3,bittab+2(r7)   ;get type/state = 3
5E46 343D 65BC            iorm r3,rmtype(r5)    ;in rally table.
                        endif
5E4A 300D A5B0          unlock rmlock(r5)
5E4E 4878 002C          lda r7,=bk0pid
5E52 30F8 00AC          sta r7,@pid
5E56 4078 131A  ihex:   call oflshb,whih        ;flush buffer.
5E5A 0040
5E5C 781C 000E  ihex2:  ldab r1,hotpid(r4)
5E60 3098 00AC          sta r1,@pid
5E64 4008 19D6          jmp ih0

5E68 300D A5B0  ihr2:   sta r0,rmlock(r5)
5E6C E698               Trap 3230,<;Bad rm blk for mes on host q - page 322>
5E6E 90F4               br ihex
```

```
.INSERT "FAKSUB"
    .INSRT FAKSUB
```

```
                .stitle Fake Host Subrouines
                ;Fake Host Common Routines

                        page LVars

     0008       nstrip =  D8            ;nominal strip = 800 microseconds
0286            ptime:  .blkw 1         ;time we should quit this strip

FC00 0600               page FakCode

                ;DOZE
                ;Put the Fake Host "JAM" (Host-to-IMP) side to sleep.
                ;Saves everything, expects map 1, R5 set up

40E0 1076       routine fdoze,arg r5/m1,local r1-r4,local m2
40E2 1016
40E4 1026
40E6 1036
40E8 1046
40EA 7078 80BE
40EE 1076

3BEC 1076       global doze = fdoze
3BEE 7078 40BE
3BF2 1076
3BF4 7078 00B6
3BF8 3078 FC00
3BFC 4078 40E0
3C00 4008 1BCE
FC00 0600

40F0 306D 0018    sta sp,dozesp(r5)                 ;put away stack pointer
                  repeat                            ;while fake HST busy
40F4 300D 0014      unlock 0#lockfd(r5)
40F8 4008 1072      jmp loopmv                      ;back to LOOP

                        page LCode
                dozew:          ;PID entry here: arg r1/m1/m3
3C04 7078 00B6      lda r7,mapfak
3C08 3078 FC00      sta r7,%map0
3C0C 4008 40FC      jmp fdozew
FC00 0600               page FakCode
                fdozew:
40FC 70F8 01A4      lda r7,@lclock              ;use STAGE's clock
4100 4AF8           add r7,=nstrip
4102 3078 0286      sta r7,ptime                ;when we should quit
4106 7059 81C0      lda r5,mblks(r1)            ;restore fake host block
410A 707D 4014      lock lockfd(r5)             ;get fake lock
410E 9AFE
4110 990A           ifnot odd                   ;not new or broken lock
4112 706D 0018        lda sp,dozesp(r5)         ;get back stack pointer
4116 9A07             ifnot zero                ;not newly initialized
4118 707D 0006          lda r7,statih(r5)       ;fake HST output
411C 4F78 2000          break if r7 .bit. =hbusy        ;return if host can run
4120 8A0B
4122 90E9               next                    ;sleep if nothing to do
                      endif
```

```
                    endif
4124 4078 418C      call pfbnum                 ;get h2pblk offset
4128 4EF8           if r7 < #nfh2               ;only for software fakes
412A 8205
                  ;   Trap 1760,<:Init Jam Fake Host - page 324>
412C 486D 004E      lda sp,#dzstack(r5)         ;empty stack
4130 408F 4E2A      jmp @dozepc(r7)             ;initial dispatch
                    endif
4134 90E0         endrepeat

4136 6076       endroutine fdoze
4138 707F 00B0
413C 3078 FC04
```

```
4140 6046
4142 6036
4144 6026
4146 6016
4148 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 328
FAKSUB.PLR;1    PAGE 2  Fake Host Subrouines

                ;JAM
                ;Send one word of data through the fake Host interface.
                ;Call with R5/fake Host block, M1/ IMP Vars page.
                ;Data word in R1

414A 1076       routine fjam, arg r1, arg r5/m1, local m2, local r1,uses m3
414C 7078 80BE
4150 1076
4152 1016

3C10 1076       global jam = fjam
3C12 7078 40BE
3C16 1076
3C18 7078 00B6
3C1C 3078 FC00
3C20 4078 414A
3C24 90EE
FC00 0600

4154 707D 0002    lda r7,starti(r5)             ;high-order memory address
4158 3078 FC04    sta r7,%map2
415C 707D 0010    lda r7,fakesi(r5)             ;input buffer pointer
4160 2017         sta r1,(r7)+                  ;stash the word
4162 481F FFFE    lda r1,=-hrdoff(r7)           ;where it went
4166 711D 0004    sub r1,endi(r5)               ;last word to fill
416A 4B18 1FFF    and r1,=0#-1
416E 8A04         if zero                       ;end of buffer
4170 4078 4224      call jampkt
4174 9005         else
4176 307D 0010      sta r7,fakesi(r5)           ;save buffer pointer
417A 4078 424E      call jamchk                 ;check if rest time
                  endif

417E 6016       endroutine fjam
4180 6076
4182 707F 00B0
4186 3078 FC04
418A 6006


                ;routine to calculate H2PBLK index into R7

418C 1076       routine pfbnum,result r7,arg r5
418E 7075         set r7 = (r5) ::statd
4190 4AF4         add r7,=nfh                   ;fakes are low in H2PBLK
4192 4B78 00FF    and r7,# HOFF                 ;kill overflow, interface type
4196 A2F1         sll r7,1                      ;and double for byte offset
4198 6006       endroutine pfbnum
```

```
                ;JAMLEADER
                ;JAM all 6 words of a new format leader.
                ;Call with pointer to the leader in R2, R5/fake Host block,
                ;M1/ IMP Vars page.

419A 1076       routine fjamleader,arg r2/m1/m2,arg r5/m1,local r1-r3,result m3
419C 1016
419E 1026
41A0 1036

3C26 1076       global jamleader = fjamleader
3C28 7078 40BE
3C2C 1076
3C2E 7078 00B6
3C32 3078 FC00
3C36 4078 419A
3C3A 90E3
FC00 0600

41A2 4078 418C    call pfbnum                   ;get H2PBLK index
41A6 707F 6326    set r7 = h2pblk(r7)           ;and host block
41AA 8B03         if minus                      ;no host?
41AC E3F9           Trap 1771,<;No host block - page 327>
41AE 901D         else
41B0 707F 0032      lda r7,hibf(r7)             ;host's input buffer
41B4 9A03           ifnot zero                  ;host expecting a message
41B6 E3F8             Trap 1770,<;Host wanted a buffer - page 327>
41B8 9018           else
41BA 701D 0002        lda r1,starti(r5)         ;packed start address
41BE 3018 FC06        sta r1,%map3              ;use map 3
41C2 A294             sll r1,4                  ;unpack
41C4 4B18 1FFF        and r1,#0#-1
41C8 48BC             lda r3,=len1+words        ;length of leader
                      repeat
41CA 6072               lda r7,(r2)+            ;next word from caller
41CC 2079 A000          sta r7,m3(r1)+          ;into input area
41D0 49B2               sub r3,=words           ;count a word
41D2 8AFC               until zero
                      endrepeat
41D4 7078 00D0        setmap m3,mapvar          ;restore map 3
41D8 3078 FC06
41DC 4078 4236        call jamhol               ;mark transfer complete, DOZE
41E0 6036             return
41E2 6026
41E4 6016
41E6 6006

                    endif
                  endif
41E8 4078 4224    call jampkt                   ;do a 1-word transfer, DOZE
41EC 6036         fail return                   ;signal failure to caller
41EE 6026
41F0 6016
41F2 6076
41F4 4FF0
41F6 4007
```

```
endroutine fjamleader
```

```
                ;JAMEND
                ;Send last data word (via JAM) and then one padding word
                ;Call with R5/fake Host block, M1 /IMP Vars page
                ;R1 has word to send

                routine fjamend, arg r1, arg r5/m1, local m2, uses r1, result m3
41F8 1076
41FA 7078 80BE
41FE 1076

3C3C 1076       global jamend = fjamend
3C3E 7078 40BE
3C42 1076
3C44 7078 00B6
3C48 3078 FC00
3C4C 4078 41F8
3C50 90D8
FC00 0600

4200 4078 414A    call fjam             ;first send the data word
4204 4818 8000    lda r1,=sign          ;and then padding
4208 707D 0002    lda r7,starti(r5)     ;high-order buffer address
420C 3078 FC04    sta r7,%map2
4210 309D 0010    sta r1,@fakesi(r5)    ;put it into buffer
4214 4078 4224    call jampkt           ;mark packet done

4218 6076       endroutine fjamend
421A 707F 00B0
421E 3078 FC04
4222 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 331
FAKSUB.PLR;1    PAGE 5  Fake Host Subrouines

                ;Auxiliary fake host routines

                ;JAMPKT
                ;Mark this input complete and poke the proper HI process
                ;Expects R5/fake Host block, M1/IMP Vars page.
                ;R1 has proper state for end bit (0 or 8000 if EOM).
                ;Enter at JAMHOL with literal word for ENDI (for fast
                ;message generator).

4224 1076       routine jampkt, arg r1, arg r5, arg m1, result m3

4226 707D 0010    lda r7,fakesi(r5)     ;current buffer pointer
422A 4B78 1FFF    and r7,=0#-1          ;only low-order address bits
422E 4C71         ior r7,r1             ;end-of-message bit
4230 307D 0004    sta r7,endi(r5)       ;set up "hardware"

4234 9002       entry jamhol, arg r5, arg m1, result m3
4236 1076

4238 707D 0006    lda r7,statih(r5)     ;host input status
423C 4B78 DFFF    and r7,=-1?hbusy      ;clear busy bit
4240 307D 0006    sta r7,statih(r5)
4244 30F8 00AC    sta r7,@pid           ;and poke the PID
4248 4078 40E0    call fdoze            ;await Host's poke

424C 6006       endroutine jampkt

                ;JAMCHK, JAMPOK
                ;JAMCHK checks if time for strip break, and does it.
                ;JAMPOK always breaks. DOZE after poking ourselves.
                ;Expects R5/fake Host block, M1/IMP Vars page

424E 1076       routine jamchk,arg r5/m1,result m3

4250 7078 0286    lda r7,ptime                  ;time for a rest?
4254 71F8 01A4    sub r7,@lclock
4258 8B09         if minus                      ;rest, but come right back

425A 9002       entry jampok, arg r5, arg m1, result m3
425C 1076

425E 787D 000F      ldab r7,bhpid(r5)           ;JAM side pid
4262 30F8 00AC      sta r7,@pid
4266 4078 40E0      call fdoze
                  endif

426A 6006       endroutine jampok
```

```
                ;FJAM1B
                ;Routine to send one buffer's worth of data to a host
                ;Traps if host code was expecting a leader

426C 1076       routine fjam1b,arg r1-r2/m2/r5,local r1,local r3-r4,result r2
426E 1016
4270 1036
4272 1046

4274 4078 418C    call pfbnum
4278 704F 6326    set r4 = h2pblk(r7)           ;this guy needs R4
427C 8B03         if minus                      ;no host?
427E E3FD           Trap 1775,<;No host block? - page 330>
4280 9022          else                         ;host exists
4282 703C 0032       set r3 = hibf(r4)
4286 8A03            if zero                    ;host wants leader
4288 E3FC              Trap 1774,<;Host wanted a leader - page 330>
428A 901D            else
428C 7019 0090         set r1 = bufe(r1)        ;remember endpointer
4290 7638 63CA         if r3 <> junk            ;don't flush the junk buffer
4294 9110
4296 7078 00D2           setmap m2,mapv2        ;get to buffer variables
429A 3078 FC04
429E 302C 0032           set hibf(r4) = r2              ;host's new buffer
42A2 704B 06A0           set r4 = where-chain(r3)       ;trade buffer usage
42A6 707A 06A0           set where-chain(r3) = where-chain(r2)
42AA 307B 06A0
42AE 304A 06A0           set where-chain(r2) = r4
42B2 4823                set r2 = r3            ;return host's input buffer
                       endif
42B4 4B18 9FFF         and r1,#sign#-1          ;make a proper XOM offset and
                                                ;retain sign (EOM bit)
42B8 301D 0004         set endi(r5) = r1        ;set endpointer for host
42BC 4078 4236         call jamhol              ;mark transfer complete
42C0 4078 4224         call jampkt              ;set endptr, poke host
                     endif
                   endif

42C4 6046       endroutine fjam1b
42C6 6036
42C8 6016
42CA 6006
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 333
FAKSUB.PLR;1    PAGE 7  Fake Host Subrouines

                ;JAMBUF
                ;Send a buffer to HI
                ;Calls FJAM1B to send it, FLUSHB to free buffer returned
                ;from HI.

42CC 1076       routine fjambuf,arg r1-r2/m2/r5

42CE 4078 426C    call fjam1b                   ;send a buffer
42D2 4078 131E    call flushb,whv2h             ;get rid of buffer from HI
42D6 0800

42D8 6006       endroutine fjambuf

3C52 1076       global jambuf = fjambuf
3C54 7078 40BE
3C58 1076
3C5A 7078 00B6
3C5E 3078 FC00
3C62 4078 42CC
3C66 90CD
FC00 0600
```

```
                ;SUCK side Routines

                ;WAIT
                ;Put the fake Host SUCK side to sleep
                ;Call with R5/fake Host block, M1/IMP Vars page.

42DA 1076       routine fwait, arg r5/m1, local m2, local r1-r4,result m3
42DC 7078 80BE
42E0 1076
42E2 1016
42E4 1026
42E6 1036
42E8 1046

3C68 1076       global wait = fwait
3C6A 7078 40BE
3C6E 1076
3C70 7078 00B6
3C74 3078 FC00
3C78 4078 42DA
3C7C 90C2
FC00 0600

42EA 306D 001A    sta sp,waitsp(r5)     ;stash away stackpointer
42EE 300D 0016    unlock 0#lockfw(r5)   ;and free the lock
42F2 4008 1072    jmp loopmv

                        page LCode
                waitw:          ;PID entry here: argr1/m1/m3
3C7E 7078 00B6    lda r7,mapfak
3C82 3078 FC00    sta r7,%map0          ;get to FAKE page
3C86 4008 42F6    jmp fwaitw
FC00 0600               page FakCode
                fwaitw:
42F6 70F8 01A4    lda r7,@lclock                ;use STAGE's clock
42FA 4AF8         add r7,=nstrip
42FC 3078 0286    sta r7,ptime                  ;when we should quit
4300 7059 81C0    lda r5,mblks(r1)              ;get proper fake Host block
4304 707D 4016    lock lockfw(r5)               ;lock it
4308 9AFE
430A 990E         ifnot odd                     ;Not new or broken lock
430C 706D 001A      lda sp,waitsp(r5)           ;restore common stack
4310 9A0B           ifnot zero                  ;Not a new block
4312 6046             return                   ;continue the fake
4314 6036
4316 6026
4318 6016
431A 6076
431C 707F 00B0
4320 3078 FC04
4324 6006
                    endif
                  endif
4326 4078 418C    call pfbnum                   ;Get host index
432A 4EF8         if r7 < #nfh2                 ;A standard fake host
432C 8205
                ;   Trap 1761,<;Init IMP Fake Host - page 332>
```

```
432E 486D 0080       set sp = #wtstack(r5)        ;Get an empty stack
4332 408F 4E22       jmp @waitpc(r7)              ;And start fake at top
                   endif
                                          ;non-IMP fakes may die here...

4336 90EE        endroutine fwait
```

```
                ;SUCK
                ;Get one word of data from the Host IH side.
                ;R1 returns the data. Call with R5/fake Host block,
                ;M1/IMP Vars page. Fail return with last data word
                ;of a message.

4338 1076       routine fsuck, arg r5/m1, uses m3, result r1, local m2
433A 7078 80BE
433E 1076

3C8A 1076       global suck = fsuck
3C8C 7078 40BE
3C90 1076
3C92 7078 00B6
3C96 3078 FC00
3C9A 4078 4338
3C9E 90B1
FC00 0600


                  repeat
4340 707D 000C      lda r7,statoh(r5)   ;Host output state
4344 4F78 2000      until r7 .bit. =hbusy       ;until sending data
4348 8A04
434A 4078 42DA      call fwait
434E 90F9         endrepeat
4350 707D 0008    lda r7,starto(r5)     ;high-order buffer address
4354 3078 FC04    sta r7,%map2
4358 707D 0012    lda r7,fakeso(r5)     ;buffer pointer
435C 7017         lda r1,(r7)           ;next data word
435E 757D 000A    eor r7,endo(r5)       ;match buffer limit?
4362 4F78 1FFF    if r7 .bit. =0#-1     ;more bits this packet?
4366 9A09
4368 707D 0012      lda r7,fakeso(r5)   ;restore fakeso
436C 4AF2           add r7,=words       ;on to next word
436E 307D 0012      sta r7,fakeso(r5)   ;save pointer
4372 4078 4420      call sukchk         ;strip break if time
4376 900C         else
4378 4078 43F8      call sukpkt         ;complete the buffer
437C 8A09           if fail             ;was EOM
437E 6076             fail return       ;signal caller
4380 707F 00B0
4384 3078 FC04
4388 6076
438A 4FF0
438C 4007
                    endif
                  endif

438E 6076       endroutine fsuck
4390 707F 00B0
4394 3078 FC04
4398 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 336
FAKSUB.PLR;1    PAGE 10         Fake Host Subrouines

                ;SUCKLEADER
                ;SUCK in a complete leader
                ;Call with R2/leader buffer to fill, R5/fake Host block
                ;M1/IMP Vars page. Calls SUCK to do its work.
                ;Fail return if EOM in leader.

                routine fsuckleader, arg r2/m1/m2, arg r5/m1, uses m3, local r1-r3
439A 1076
439C 1016
439E 1026
43A0 1036

3CA0 1076       global suckleader = fsuckleader
3CA2 7078 40BE
3CA6 1076
3CA8 7078 00B6
3CAC 3078 FC00
3CB0 4078 439A
3CB4 90A6
FC00 0600

                  repeat                        ;until host posts output
43A2 707D 000C      lda r7,statoh(r5)           ;output status
43A6 4F78 2000      until r7 .bit. #hbusy       ;until "active"
43AA 8A04
43AC 4078 42DA      call fwait                  ;get control later
43B0 90F9         endrepeat
43B2 701D 0008    lda r1,starto(r5)             ;get map setting
43B6 4B18 FE00    and r1,#mapmsk
43BA 7618 00D0    if r1 <> mapvar               ;not sending a leader
43BE 9107
43C0 E3FA           Trap 1772,<;Host sending a buffer - page 334>
43C2 4878 FFFF      lda r7,#-1                  ;make an illegal msg type
43C6 307A 0002      sta r7,typ1(r2)
43CA 900A          else
43CC 751D 0008      eor r1,starto(r5)           ;just page offset
43D0 A294           sll r1,4                    ;make XOM pointer
43D2 48BC           lda r3,=len1+words          ;count words
                    repeat
43D4 6079 6000        lda r7,m1(r1)+    ;;mapvar ;next leader word
43D8 2072             sta r7,(r2)+              ;into caller's area
43DA 49B2             sub r3,=words
43DC 8AFC             until zero                ;done the 6 words
                    endrepeat
                  endif
43DE 4078 43F8    call sukpkt                   ;mark interface idle
43E2 8A07         if fail                       ;was just a leader
43E4 6036           fail return                 ;let caller know
43E6 6026
43E8 6016
43EA 6076
43EC 4FF0
43EE 4007

                  endif

43F0 6036       endroutine fsuckleader
```

43F4 6016
43F6 6006

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 337
FAKSUB.PLR;1    PAGE 11         Fake Host Subroutines

                ;Auxilliary SUCK side fake Host routines

                ;SUKPKT
                ;Marks the current buffer complete and pokes the IH side
                ;Enter with R5/fake Host block, M1/IMP Vars page.
                ;Fail return if EOM.

43F8 1076       routine sukpkt, arg r5, arg m1

43FA 707D 000A    lda r7,endo(r5)       ;remember for EOM bit
43FE 1076         push r7
4400 707D 000C    lda r7,statoh(r5)     ;output status
4404 4B78 DFFF    and r7,=-1?hbusy      ;mark it done
4408 307D 000C    sta r7,statoh(r5)
440C 30F8 00AC    sta r7,@pid           ;and poke IH code
4410 4078 4420    call sukchk           ;break strip maybe
4414 6076         pop r7                ;get back ENDO
4416 8B04         if minus              ;was EOM
4418 6076           fail return
441A 4FF0
441C 4007
                  endif

441E 6006       endroutine sukpkt

                ;SUKCHK, SUKPOK
                ;SUKCHK - Conditionally do SUKPOK if strip time
                ;SUKPOK - Always poke ourselves and call WAIT.
                :Expect R5/fake Host block, M1/IMP Vars page

4420 1076       routine sukchk,arg r5/m1,uses m3

4422 7078 0286    lda r7,ptime                  ;time for a rest?
4426 71F8 01A4    sub r7,@lclock
442A 8B09         if minus

442C 9002       entry sukpok, arg r5, arg m1, result m3
442E 1076

4430 787D 000E      ldab r7,hbpid(r5)
4434 30F8 00AC      sta r7,@pid
4438 4078 42DA      call fwait
                  endif

443C 6006       endroutine sukpok

                ;SUCMSG
                ;This routine throws away the whole message

443E 1076       routine sucmsg,arg r5,uses r1,result m3
                  repeat
4440 4078 43F8      call sukpkt
4444 8AFE           until fail
                  endrepeat
4446 6006       endroutine sucmsg
```

```
                ;SUCBUF, FSUCBUF
                ;Get next IH buffer
                ;Waits for IH to post next output, returns M2 set to MAPV2,
                ;buffer CHAIN in R2, unpacked in R1, BUFE set up in buffer.
                ;Fail return when messed up with IH.

4448 1076       routine fsucbuf,local r4,arg r1/m2/r5,result r1-r2
444A 1046

444C 707D 000C    repeat while statoh(r5) .nbit. #hbusy
4450 4F78 2000
4454 8A04
4456 4078 42DA      call fwait                  ;wait for next I/O
445A 90F9         endrepeat
445C 704D 0008    set r4 = starto(r5)           ;find what IH is sending
4460 4B48 FE00    and r4,#mapmsk
4464 7648 00D0    if r4 = mapvar                ;sending from Vars page
4468 8103
                    ;this means the host code was sending a leader
                    ;we'll throw it away, unfortunately
446A E3FB           Trap 1773,<:Host sending leader - page 336>
446C 9024          else
446E 4078 418C      call pfbnum
4472 704F 6326      set r4 = h2pblk(r7)         ;host's block
4476 8B03           if minus                    ;there is no host?
4478 E3FE             Trap 1776,<:No host block? - page 336>
447A 901D           else
447C 702C 0068        set r2 = ihbuff(r4)       ;get current buffer
4480 7018 00D2        set m2,r1 = mapv2
4484 3018 8000
4488 4078 1500        call unpckc,whih          ;check IH ownership
448C 0040
448E 8A03             if fail                   ;hmmm
4490 E3FF               Trap 1777,<:Bad buffer from IH - page 336>
4492 9011             else
4494 707D 000A          set bufe(r1) = endo(r5) ;copy endpointer to BUFE
4498 3079 0090
449C 7078 00D2          set m2 = mapv2          ;now access buffer tables
44A0 3078 8000
;               ;       setbit where-chain(r2),#whh2v   ;add fake's bit
44A4 4878 0400          set r7 = #whh2v         ;add our bit
44A8 347A 06A0          iorm r7,where-chain(r2) ;to buffef
44AC 4078 43F8  call sukpkt             ;get IH going again
44B0 6046               return                  ;and all's well
44B2 6006

                      endif
                    endif
                  endif
44B4 4078 43F8    call sukpkt                   ;let IH run anyway
44B8 6046         fail return
44BA 6076
44BC 4FF0
44BE 4007


                endroutine fsucbuf

3CB6 10         global sucbuf = fsucbuf
```

```
3CB8 7078 40BE
3CBC 1076
3CBE 7078 00B6
3CC2 3078 FC00
3CC6 4078 4448
3CCA 909B
FC00 0600
```

```
.INSERT "FAKES"
    .INSRT FAKES
```

```
                .stitle Fake Hosts

                ;*******fake 0 to process chrs from host to tty****

                        Page FakVars

5E00            fhthld: .blkw 6
5E0C            fhtwrd: .blkw 1 ;count of words per line in octal
5E0E            flagop: .blkw 1
5E10            temwrd: .blkw 1

FC00 0600               Page FakCode

                fht:
                repeat                          ;forever
                  repeat
44C0 4828 5E00      lda r2,=fhthld
44C4 4078 439A      call fsuckleader
44C8 8A0E           until succeed
44CA 7078 5E02      lda r7,fhthld+typl
44CE 4F78 0400      if r7 .bit. =octbit } flagop ; octal print leader.
44D2 8A04
44D4 7078 5E0E
44D8 9A05
44DA 4078 4588        call fhtpld
44DE 4078 45C2        call printc
                    endif
44E2 90EF         endrepeat
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 342
FAKES.PLR;1     PAGE 2  Fake Hosts

44E4 7078 5E02    lda r7,fhthld+typl
44E8 4F78 0400    if r7 .bit. =octbit } flagop  ;octal print.
44EC 8A04
44EE 7078 5E0E
44F2 9A0D
44F4 4078 4588      call fhtpld
                    repeat
44F8 4078 45A0        call wordp
44FC 4078 4338        call fsuck
4500 8AFC             until fail
                    endrepeat
4502 4078 45A0      call wordp                  ;print last word
4506 4078 45C2      call printc                 ;and <CR><LF>
450A 9014          else
                     repeat
450C 4078 4338         call fsuck
4510 8A06              if fail
4512 A798                rrl r1, D8
4514 8B03                if minus
4516 4078 45D8             call f2tput
                         endif
451A 900C                break
                       endif
451C A798              rrl r1, D8
451E 8A04              if zero
4520 4078 443E           call sucmsg
4524 9007                break
                       endif
4526 4078 45D8         call f2tput
452A A798              rrl r1, D8
452C 4078 45D8         call f2tput
4530 90EE            endrepeat
                   endif
4532 90C7        endrepeat
```

```
                ;******** fake 0 to process chrs from tty to host

                fth:
                repeat
4534 4078 4608    call t2fget
4538 4E18 00BB    if r1 = =psemicolon
453C 810E
453E 4828 728C      lda r2,=clled
4542 4078 419A      call fjamleader
                    repeat
4546 4078 4608        call t2fget
454A 4E18 00BB        until r1 = =psemicolon
454E 9104
4550 4078 414A        call fjam
4554 90F9           endrepeat
4556 9015          else
4558 4828 7280      lda r2,=ccled
455C 4078 419A      call fjamleader
4560 4E18 0080      if r1 = =SCntlP } r1 = = H90        ;shift-control P
4564 9104
4566 4E18 0090
456A 8109
456C 4878 80FD        lda r7,=getpri+ HFD       ;route to local DDT
4570 3078 7284        sta r7,ccled+hstl
4574 7078 62E2        lda r7,mine
4578 3078 7286        sta r7,ccled+dstl
                    endif
457C 4078 414A      call fjam
                  endif
4580 4890         lda r1,=0                     ;send final word of zeros
4582 4078 41F8    call fjamend
4586 90D7       endrepeat
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 344
FAKES.PLR;1     PAGE 4  Fake Hosts

4588 1076       routine fhtpld, result r1
458A 4078 45C2    call printc
458E 48A0         lda r2,=0
                  repeat
4590 601A 5E00      lda r1,fhthld(r2)+
4594 4EAC           until r2 = =6*words
4596 9104
4598 4078 45A0      call wordp
459C 90FA         endrepeat
459E 6006       endroutine fhtpld

45A0 1076       routine wordp, arg r1, uses r4
45A2 4078 103C    call hexout, f2tput
45A6 45D8
45A8 48C1         lda r4,=1
45AA 3148 5E0C    subm r4,fhtwrd        ;count words
45AE 8B05         if minus              ;end of line
45B0 48CF           lda r4,= D15        ;set up for next line
45B2 3048 5E0C      sta r4,fhtwrd       ; (let tty do nec cr)
45B6 9005         else
45B8 4818 0020      lda r1,='
45BC 4078 45D8      call f2tput
                  endif
45C0 6006       endroutine wordp
```

```
45C2 1076       routine printc
45C4 489D         lda r1,=ascicr                    ;send <CR>, <LF>
45C6 4078 45D8    call f2tput
45CA 489A         lda r1,=ascilf
45CC 4078 45D8    call f2tput
45D0 48FF         lda r7,= D15                    ;reset words per line cntr.
45D2 3078 5E0C    sta r7,fhtwrd
45D6 6006       endroutine printc


45D8 1076       routine f2tput, arg r1/r5, local r1-r2
45DA 1016
45DC 1026
45DE 4B18 007F     and r1,= H7F
45E2 9A10          ifnot zero
                      repeat
45E4 4078 0FD2           call rbfput, ttyobf     ;put char into tty
45E8 6296
45EA 810C                while equal             ;tty full, wait
45EC 787D 000E             ldab r7,hbpid(r5)     ;have tty poke us
45F0 3078 627A             sta r7,f2tpok         ; when ready
45F4 4878 0014             set @pid = #ttypid    ;get tty to run
45F8 30F8 00AC
45FC 4078 42DA             call fwait            ;sleep on it
4600 90F2               endrepeat
                      endif
4602 6026       endroutine
4604 6016
4606 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 346
FAKES.PLR;1     PAGE 6  Fake Hosts

                ;tty fake continued

4608 1076       routine t2fget, result r1, arg r5
                   repeat
460A 7078 030C        lda r7,debugm
460E 9B09             ifnot minus
4610 4078 1008           call rbfget, ttyibf
4614 627E
4616 8108             while equal
4618 787D 000F              ldab r7,bhpid(r5)      ;poke this when ready
461C 3078 627C              sta r7,t2fpok
                          endif
4620 4078 40E0        call fdoze
4624 90F3          endrepeat
4626 4C18 0080     ior r1,= H80                    ;always parity
462A 6006       endroutine t2fget
```

```
FAKES.PLR;1     PAGE 7  Fake Hosts

                ;***** fake host 1 to ddt *****

     003B       semicolon= ';                    ;define semicolon character.
     00BB       psemicolon= semicolon} H80       ;semicolon with parity bit.
     0080       SCntlP= 0} H80           ;shift control pee

                        Page FakVars

5E12            ddtmpl: .blkw 6 ;put ddt IH leaders here first.

FC00 0600               Page FakCode

                fhd:
                repeat
                  repeat
462C 4828 5E12      lda r2,=ddtmpl
4630 4078 439A      call fsuckleader
4634 9AFC           while fail
                  endrepeat

4636 489C         lda r1,=len1+words            ;leader length
                  repeat
4638 5079 5E12      lda r7,ddtmpl(-r1)          ;copy leader for jam
463C 3079 7298      sta r7,ddtled(r1)
4640 88FC           until loop
                  endrepeat

                  repeat
4642 4078 4338      call fsuck
4646 9A0C           until fail
4648 A798           rrl r1, D8
464A 8A04           if zero
464C 4078 443E        call sucmsg
4650 9007             break
                    endif
4652 4078 4690      call f2dput
4656 A798           rrl r1, D8
4658 4078 4690      call f2dput
465C 90F3         endrepeat

465E 4818 003B    lda r1,=semicolon
4662 4078 4690    call f2dput
4666 90E3       endrepeat

                ;end of fhd
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 348
FAKES.PLR;1     PAGE 8  Fake Hosts

                ;******* fake host 1 from ddt to host *****
                fdh:
                repeat
                  repeat
4668 4078 46C8      call d2fget
466C 4E18 003B      until r1 <> =semicolon
4670 91FC
                  endrepeat

4672 4828 7298    lda r2,=ddtled
4676 4078 419A    call fjamleader
                  repeat
467A 4078 414A      call fjam
467E 4078 46C8      call d2fget
4682 4E18 003B      until r1 = =semicolon
4686 81FA
                  endrepeat
4688 4890         lda r1,=0
468A 4078 41F8    call fjamend

468E 90ED       endrepeat

                ;end fdh
```

```
                ; Fake Host 1 (DDT) utility routines

                ; send char in r1 to ddt.

4690 1076       routine f2dput, arg r1, arg r5
                  repeat
4692 7078 030C      lda r7,debugm
4696 9B0D           ifnot minus                 ;not S/A TTY/DDT
4698 7078 A260        lock f2dl
469C 9AFE
469E 7878 6276        ldab r7,f2db
46A2 9A0A             until zero
46A4 787D 000E        ldab r7,hbpid(r5)
46A8 3078 6278        sta r7,f2dpok
46AC 3008 A260        unlock f2dl
                    endif
46B0 4078 42DA      call fwait
46B4 90EF         endrepeat
46B6 3818 6276    stab r1,f2db
46BA 4878 0012    lda r7,=ddtpid
46BE 30F8 00AC    sta r7,@pid
46C2 3008 A260    unlock f2dl
46C6 6006       endroutine f2dput


                ; get next char from ddt in r1.

46C8 1076       routine d2fget, arg r5, result r1
                  repeat
46CA 7078 030C      lda r7,debugm
46CE 9B0D           ifnot minus                 ;not S/A TTY/DDT
46D0 7078 A25E        lock d2fl
46D4 9AFE
46D6 7818 A26C        ldab r1,d2fb+foo
46DA 8A0A             while zero
46DC 787D 000F        ldab r7,bhpid(r5)
46E0 3078 626E        sta r7,d2fpok
46E4 3008 A25E        unlock d2fl
                    endif
46E8 4078 40E0      call fdoze
46EC 90EF         endrepeat

46EE 4878 0012    lda r7,=ddtpid
46F2 30F8 00AC    sta r7,@pid
46F6 3008 A25E    unlock d2fl
46FA 6006       endroutine d2fget
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 350
FAKES.PLR;1     PAGE 10         Fake Hosts

                ;Fake Host 2 Suck (IMP to Host) Side
                ;Read in message leader, then dispatch on first data word.
                ;If it's positive, we got a param change message.
                ;If negative, it must be packet core

                        Page FakVars

5E1E            fh2led: .blkw 6 ;storage for leader

FC00 0600               Page FakCode

                fh2suc:
                repeat                  ;forever
46FC 4828 5E1E    lda r2,=fh2led
                  repeat                ;until non-leader
4700 4078 439A      call fsuckleader
4704 9AFE           until success       ;ignore control messages
                  endrepeat
4706 4078 4338    call fsuck            ;get a data word
470A 9A08         if succeed            ;long enough message
470C 4811           lda r1,r1           ;pkt core or param?
470E 8B04           if minus            ;pkt core
4710 4078 471C        call fhcsuc               ;process it
4714 9003           else
                pccall:         ;hook onto this call for param change
4716 4078 443E        call sucmsg               ;null routine
                    endif
                  endif
471A 90F1       endrepeat
```

```
                ;FHCSUC
                ;Process incoming packet core messages
                ;1) get a buffer, and copy all of the message into it.
                ;2) if for us (and not magic modem), pass it to PKCIC (packet core).
                ;3) else if magic modem, copy down the data within the buffer
                ;4)      send to the appropriate modem.
                ;Enter with R5/ fake param block, R1/ first data word, leader
                ;is in FH2LED already.

     0001       .iif nz PSE             ;Suffer fast lines and a slow NOC in Platform
     03C0               crpcnt =  D32* D30      ;30 seconds (?)

471C 1076       routine fhcsuc,arg r1,arg r5,uses r1-r4,uses m2

471E 1016       save r1                         ;look at data later
                repeat
4720 4078 13C6    call freget,whf2h+nsf ;use store/forward pool
4724 4006
4726 8A04         until success                 ;no buffers
4728 4078 42DA    call fwait
472C 90FA       endrepeat
472E 4841       lda r4,r1                       ;put away buff ptr
4730 48F0       lda r7,=0
4732 2071       sta r7,(r1)+    ;;neth          ;block length later
4734 4838 C000  lda r3,=rldtyp
4738 2031       sta r3,(r1)+    ;;typh          ;pkt core type
473A 2071       sta r7,(r1)+    ;;chkh          ;cksum set later
473C 7038 62E2  lda r3,mine                     ;my IMP number
4740 2031       sta r3,(r1)+    ;;dsth
4742 2071       sta r7,(r1)+    ;;cpsatd        ;no dest yet
4744 7038 5E22  lda r3,fh2led+hstl              ;source host
4748 2031       sta r3,(r1)+    ;;cpdh
474A 7038 5E24  lda r3,fh2led+dstl              ;and IMP
474E 2031       sta r3,(r1)+    ;;cpdi
4750 7038 5E26  lda r3,fh2led+midl              ;and mess-id
4754 2031       sta r3,(r1)+    ;;cpdl
4756 4831       lda r3,r1                       ;buffer pointer
4758 6016       restore r1                      ;get back data
                repeat                          ;through message
475A 2013         sta r1,(r3)+                  ;store data
475C 4078 4338    call fsuck                    ;get next word
4760 8AFD         until fail                    ;just got last
                endrepeat
4762 2013       sta r1,(r3)+                    ;store last word
4764 4814       lda r1,r4                       ;restore buffer start address
4766 4934       SUB R3,R4
4768 4B38 FFFC  and r3,=-4                      ;force even number of words
476C A6B1       srl r3,1                        ;bytes-words
476E 3031       sta r3,(r1)                     ;set length
4770 A2B1       sll r3,1                        ;bytes again
4772 4078 17C8  call lpcksubr           ;store length and compute checksum.
4776 3239 0004  addm r3,chkh(r1)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE. 352
FAKES.PLR;1     PAGE 12          Fake Hosts

477A 7039 0010  lda r3,cmark(r1)              ;line at dest
477E 4873       lda r7,r3
4780 4BB7       and r3,=cpline                ;just line num
4782 A2B1       sll r3,1
4784 8A07       if zero                       ;it's for us
4786 4F78 0100    ifnot r7 .bit. =cmmbit      ;ignore if magic modem
478A 8A03
478C 4078 3CCC      call fhcsl                ;but process regular ones
                  endif
4790 902E       else
4792 701C 0010    lda r1,cmark(r4)            ;get bit flags
4796 1016         save r1                     ; and remember for later
4798 4F78 0100    if r7 .bit. =cmmbit         ;this is a magic modem pkt
479C 9A10
479E 1036           save r3
47A0 4818 0012      lda r1,=cmmbeg            ;where MM data starts
47A4 707C 0090      lda r7,bufe(r4)           ;current eom
47A8 4A74           add r7,r4                 ;last data word
47AA 311C 0090      subm r1,bufe(r4)          ;new length
47AE 4A14           add r1,r4                 ;beginning of MM data
                    repeat                    ;for all data
47B0 6031             lda r3,(r1)+            ;copy it down
47B2 2034             sta r3,(r4)+
47B4 4E17             until r1 > r7
47B6 8CFD
                    endrepeat
47B8 6036           restore r3
47BA 9007         else
47BC 707B 63A4      lda r7,m2nghb-words(r3)   ;neighbor this line
47C0 307C 0008      sta r7,cpsatd(r4)         ;into packet
47C4 317C 0004      subm r7,chkh(r4)          ;fix chksum too
                  endif
47C8 6016         restore r1                  ;get back flags word
47CA 704B 635C    lda r4,m2pblk-words(r3)     ;no modem 0
47CE 7078 00D2    setmap m2,mapv2
47D2 3078 FC04
47D6 4B18 0080    and r1,=crpbit              ;repeating this pkt?
47DA 9A03         ifnot zero                  ;yes
47DC 4818 03C0      lda r1,=crpcnt            ;how many times
                  endif
47E0 301A 0350    sta r1,chan-chain(r2)
47E4 4078 47F4    call fhcsblk                ;set SBLK this modem
47E8 9A02         if success                  ;gave away buffer
47EA 6006           return
                  endif
                endif
47EC 4078 131E  call flushb,whf2h             ;done with buffer
47F0 4000

47F2 90FC       endroutine fhcsuc
```

```
                ;FHCSL
                ;Call PKCIC from FHCSUC with a buffer for our PKT CORE
                ;R1 is unpacked buffer pointer, M2 set up

FC00 0000               page RelCode

5C0C 1076       routine fhcsr,arg r1,arg m2,local r2,uses r1,uses r3-r4
5C0E 1026

3CCC 1076       global fhcsl = fhcsr
3CCE 7078 40BE
3CD2 1076
3CD4 7078 00B0
3CD8 3078 FC00
3CDC 4078 5C0C
3CE0 9090
FC00 0000

                repeat
5C10 7078 00B0    setmap m3,maprel
5C14 3078 FC06
5C18 7078 BEAE    lock pkclok                   ;lock pkt core params
5C1C 9AFE
5C1E 4078 4BEA    call pkcic,bltfak             ;do this packet
5C22 4000
5C24 8A0E         until success                 ;done
5C26 787D 000E    ldab r7,hbpid(r5)             ;poke us when PKT CORE done
5C2A 3078 5DD0    sta r7,bltpok
5C2E 3008 BEAE    unlock pkclok                 ;unlock pkt core params.
5C32 7078 00D0    setmap m3,mapvar
5C36 3078 FC06
5C3A 4078 3C68    call wait                     ;global entry
5C3E 90E9       endrepeat
5C40 3008 BEAE  unlock pkclok                   ;done PKT CORE
5C44 7078 00D0  setmap m3,mapvar
5C48 3078 FC06

5C4C 6026       endroutine fhcsl
5C4E 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 354
FAKES.PLR;1     PAGE 14          Fake Hosts

                ;FHCSBLK
                ;Set up send-the-block pointer (SBLK) with this buffer
                ;Call with R5/fake param block, R4/target modem param block,
                ;R2/buffer CHAIN pointer word.
                ;Fail return if we waited too long to set it, else buffer
                ;is gone

FC00 0600               page FakCode

47F4 1076       routine fhcsblk,arg r2/m2,arg r4-r5,uses r3

47F6 703C 002E  lda r3,sblk(r4)                 ;got a buffer already?
47FA 9A17       ifnot zero                      ;yes, must wait
47FC 4818 03C0    lda r1,=crpcnt                ;try to hurry it out
4800 707C 4000    lock lockm(r4)                ;Keep modem code away
4804 9AFE
4806 311B 0350    subm r1,chan-chain(r3)        ;by clearing repeat count
480A 3004         unlock (r4)   ;;lockm         ;done locked stuff
480C 4838 0078    lda r3,= D3* D40              ;give up in 3 seconds
                  repeat
4810 4078 42DA      call fwait                  ;nothing to do now
4814 707C 002E      lda r7,sblk(r4)             ;gone yet?
4818 9A08           until zero                  ;yes
481A 49B1           sub r3,=1                   ;else count a wait
481C 8AFA           next ifnot zero             ;all's okay
481E E3C2           Trap 1702,<;Flushing Reload Packet (NO BUF SP) - page 352>
4820 6076           fail return
4822 4FF0
4824 4007
4826 90F5         endrepeat
                 endif
4828 302C 002E  sta r2,sblk(r4)                 ;now send this buffer
482C 4078 14E8  call pokem                      ;and tell modem so

4830 6006       endroutine fhcsblk
```

```
                ;Fake Host 2 Routines

                ;JAM side
                ;Trace sending process is currently missing
                ;Check for packet core messages to send:
                ; 1) call PKCOC to see if any packet core from us
                ; 2) check FHCQ for packets from our neighbors

                        Page FakVars

5E2A            fhcqld: .blkw 6 ;leader send area

FC00 0600               Page FakCode


                fh2jam:
                repeat                          ;forever
4832 4078 40E0    call fdoze                    ;rest here always
4836 4078 3CE2    call fhcjl                    ;check our PKT CORE
483A 8A2C         if fail                       ;nothing doing
483C 4818 6568      lda r1,=sfhcq               ;see if any from M2I
4840 4078 154A      call deque,whf2h
4844 4000
4846 9AF6           next if fail                ;none there either
4848 7049 0092      lda r4,inch(r1)             ;which modem
484C 783C 001A      ldab r3,modem(r4)           ;M2PBLK offset
4850 7079 0006      lda r7,cpsats(r1)           ;source id
4854 307B 63A6      sta r7,m2nghb(r3)           ;into neighbor table
4858 A6B1           srl r3,1                    ;line number for later
485A 4AB1           add r3,=1
485C 7049 0008      lda r4,cpsatd(r1)           ;destination for setup
4860 8A16           if zero                     ;no, we might help
4862 7049 0002        lda r4,typh(r1)           ;report neighbor's status
4866 E3C1             Trap 1701,<;Neighbor IMP wants a reload - page 353>
4868 3839 0019        stab r3,cforl(r1)         ;remember line num
486C 7035             lda r3,(r5)               ;FH2 host number
486E 7078 A566        lda r7,m3#pchelp          ;redirect it?
4872 8A05             if zero                   ;no
4874 7878 40C5          ldab r7,ssimp           ;default dest then
4878 7838 40C4          ldab r3,sshost
                      endif
487C 3079 000C        sta r7,dsth(r1)           ;set dest IMP #
4880 3839 000B        stab r3,cpdh(r1)          ;and Host
4884 4878 B400        lda r7,=csetup            ;help out 316s
4888 3079 0010        sta r7,cmark(r1)
                    endif
488C 7039 0090      lda r3,bufe(r1)             ;get back endpointer
4890 4A31           add r3,r1
                  endif
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 356
FAKES.PLR;1     PAGE 16          Fake Hosts

                ;Get here with buffer ready to JAM
                ;leader is in the buffer

4892 4878 0F00    lda r7,=hicode
4896 3078 5E2A    sta r7,fhcqld+netl            ;set up our leader area
489A 7879 000B    ldab r7,cpdh(r1)              ;dest host
489E 3078 5E2E    sta r7,fhcqld+hstl
48A2 7079 000C    lda r7,dsth(r1)               ;dest IMP
48A6 3078 5E30    sta r7,fhcqld+dstl
48AA 7079 000E    lda r7,cpdl(r1)               ;and mess id
48AE 3078 5E32    sta r7,fhcqld+midl
48B2 4842         lda r4,r2                     ;save buffer pointer
48B4 4828 5E2A    lda r2,=fhcqld                ;where leader starts
48B8 4078 419A    call fjamleader
48BC 4824         lda r2,r4                     ;restore buffer pointer
48BE 4849 0010    lda r4,=cmark(r1)             ;beginning of data
                  repeat
48C2 6014           lda r1,(r4)+
48C4 4E43           until r4 >= r3              ;to end of data
48C6 8204
48C8 4078 414A      call fjam                   ;send the word
48CC 90FB         endrepeat
48CE 4078 41F8    call fjamend                  ;send last word
48D2 4078 131E    call flushb,whf2h             ;done with buffer
48D6 4000
48D8 90AD       endrepeat

                ;end of FH2JAM
```

```
                ;FHCJL
                ;Routine which checks our packet core process for any to send
                ;Expects R5/fake Host block, M1,3/IMP Vars
                ;Returns fail if none to send. If succeed, R2 has buffer
                ;pointer and R1 unpacked (M2) endpointer, M2 set up.

FC00 0000               page RelCode

5C50 1076       routine fhcjr, uses r1-r4, arg r5, result r1-r3, result m2

3CE2 1076       global fhcjl = fhcjr
3CE4 7078 40BE
3CE8 1076
3CEA 7078 00B0
3CEE 3078 FC00
3CF2 4078 5C50
3CF6 9085
FC00 0000

5C52 7078 5EB0    lda r7,pkcst                  ;see if PKT CORE active
5C56 892F         if odd        ;;pkcact       ;'tis
5C58 4078 13C6      call freget,whf2h+nhi       ;allocate a buffer
5C5C 400E
5C5E 9A2B           if success                  ;got one '
                      repeat                    ;until PKT CORE gives us bfr
5C60 7078 5EB0          lda r7,pkcst            ;still active?
5C64 8925               while odd       ;;pkcact
5C66 7078 00B0          setmap m3,maprel        ;set map for pkclok
5C6A 3078 FC06
5C6E 1016               save r1
5C70 7078 BEAE          lock pkclok
5C74 9AFE
5C76 4078 4D58          call pkcoc,bltfak       ;get buffer filled
5C7A 4000
5C7C 9A0B               if success              ;PKCOC done
5C7E 3008 BEAE            unlock pkclok
5C82 7078 00D0            setmap m3,mapvar
5C86 3078 FC06
5C8A 4831                 lda r3,r1             ;address of end of data.
5C8C 6016                 restore r1
5C8E 8110                 break ifnot equal     ;nothing to do, really.

5C90 6006                 return
                        endif
5C92 6016               restore r1
5C94 3008 BEAE          unlock pkclok           ;done PKT CORE
5C98 787D 000F          ldab r7,bhpid(r5)       ;our PID
5C9C 3078 5DD0          sta r7,bltpok           ;for BLT to poke
5CA0 7078 00D0          setmap m3,mapvar
5CA4 3078 FC06
5CA8 4078 3BEC          call doze               ;wait some now
5CAC 90DA             endrepeat
5CAE 4078 131E        call flushb,whf2h         ;free our buffer
5CB2 4000
                    endif
                  endif
5CB4 60[illegible]  fail return
```

```
5CB6 4FF0
5CB8 4007
```

```
endroutine fhcjl
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE.359
FAKES.PLR;1     PAGE 18          Fake Hosts


                ;*******discard********

                        Page FakVars

     30D4       sowdtm =  D12500        ;5 minutes, 20 sec (stime units)
5E36            disled: .blkw 6         ;discard leader buffer
5E42            countd: .blkw 1         ;pkts and leaders discarded

FC00 0600               Page FakCode

                discard:
                repeat                  ;forever
48DA 48F1         lda r7,=1
48DC 3278 5E42    addm r7,countd
48E0 4828 5E36    lda r2,=disled
48E4 4078 439A    call fsuckleader
48E8 8A12         if fail
48EA 7878 5E39      ldab r7,disled+typl+1
48EE 4EF5           if r7 = =crfnm              ;a RFNM- hold off WDT
48F0 810D
48F2 7028 00A2        lda r2,stim2              ;get hi order time
48F6 7078 00A0        lda r7,stime              ;get low order current time
48FA 4A78 30D4        add r7,#sowdtm            ;how long until dead
48FE 8402             if carry
4900 4AA1               add r2,#1               ;correct hi order
                      endif
4902 3028 625C        sta r2,wdgtim+words       ;save hi time
4906 3078 625A        sta r7,wdgtim             ;and lo time
                    endif
490A 9006         else
490C 4078 4338      call fsuck
4910 9A03           if succeed  ;got a packet.
4912 4078 443E        call sucmsg               ;dump all pkts
                    endif
                  endif
4916 90E2       endrepeat
```

```
                ;********** statistics **********

                        PAGE Fakvars

                ; *** statistics variables

5E44            tortm:  .blkw 1 ;lengths, cycles 0-3*2
5E46            tortd:  .blkw 1 ;dests, cycles 0-7*4
5E48            statdt: .blkw  D8*2     ;8 torture test destinations
5E68            mgnl:   .blkw 1 ;init to 8, negative for torture test
                        ;length in words: 0 to  H1F7 legal
5E6A            statb:  .blkw 0
0000            .=0     ;layout of stat blocks
0000                    .blkw 6         ;six words of leader (must be first)
000C            snon:   .blkw 1         ;0 => off
000E            statf:  .blkw 1         ;frequency
0010            olds:   .blkw 1         ;old value of sync
0012            impoff: .blkw 1         ;offset sync for my imp #
     0014       statl=.-0
5E6A            .=statb
                ;current stat routines:
5E6A            mgsb:   .blkb statl     ;message generator, MUST be first}
5E7E            stsb:   .blkb statl     ;srep trouble report
5E92            thrusb: .blkb statl     ;throughput statistics
5EA6            trpsb:  .blkb statl     ;traps report
5EBA            dgsb:   .blkb statl     ;buffer checksum error dump (diagrp)
5ECE            hwsb:   .blkb statl     ;software watchdog leader (to us)
5EE2            tlsb:   .blkb statl     ;TENEX Logger leader
5EF6            tlttcp: .blkw 6         ;TCP leader for TENEX logger
5F02            tttcpe: .blkw 0         ;end of TCP header

5F02            snapbf: .blkw  H40      ;TENEX logger buffer

5F82            throtc: .blkw 1
5F84            banom:  .blkw 1
5F86            trblck: .blkw 1
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 361
FAKES.PLR;1     PAGE 20          Fake Hosts

FC00 0600                Page FakCode

4918 5E6A       stbp:    mgsb
491A 5EE2                tlsb
491C 5E7E                stsb     ;status statistic block
491E 5E92                thrusb   ;throughput stats block
4920 5EA6                trpsb    ;trap stats block
4922 5EBA                dgsb
4924 5ECE                hwsb
4926 FFFF       stbpsp:  -1
4928 FFFF                -1

492A 49C2       cawl:    genm
492C 4A48                tlog
492E 4A7E                status
4930 4B3A                thrurpt
4932 4C18                traprp
4934 4D4E                diagrp
4936 4C6C                holdwd
4938 0000       cawlsp:  0
493A 0000                0
     0012       statot=.-cawl

493C 0400       canom:   msgbit
493E 0000                0
4940 0000                0
4942 0000                0
4944 0000                0
4946 0000                0
4948 0000                0
494A 0000       anomsp:  0
494C 0000                0
```

```
                ;jam side stats to send assorted msgs.
                ;top level infinite repeat to do dispatches

                stats:                  ;statistics fake host multiplexer.
                repeat
494E 4818 0010    lda r1,=statot-words  ;start at end of list.
                  repeat
4952 7079 493C      lda r7,canom(r1)    ;stats bit for NCC
4956 7029 4918      lda r2,stbp(r1)     ;this one's block
495A 9B2B           ifnot minus         ;there is a stat here.
495C 703A 000C        lda r3,snon(r2)   ;on/off switch
4960 8A06             if zero           ;Stat off
4962 4D78 FFFF          eor r7,=-1      ;change bit,
4966 3378 62AE          andm r7,anom    ;and note turning off
496A 9023             else              ;Stat is on.
496C 3478 62AE          iorm r7,anom    ;record turning on this guy
4970 48F0               lda r7,=0       ;clear checksum for the stat
4972 3078 5F86          sta r7,trblck
4976 1016               save r1
4978 707A 000E          lda r7,statf(r2) ;freq to send this at
497C 8A04               if zero
497E 40F9 492A            call @cawl(r1) ;call this stat package.
4982 9016               else
4984 7038 62DE            lda r3,sync   ;check if time to send this
4988 713A 0010            sub r3,olds(r2)       ;truncated time of last run
498C 8B04                 if minus
498E 4D38 FFFF              eor r3,=-1  ;make time dif plus
4992 4AB1                   add r3,=1
                          endif
4994 713A 0012            sub r3,impoff(r2)     ;account for impno
4998 4937                 sub r3,r7     ;if one period passed
499A 9B0A                 ifnot minus
499C 49F1                   sub r7,=1   ;remember this time, by truncating
499E 4D78 FFFF              eor r7,=-1
49A2 7378 62DE              and r7,sync ;lower bits off sync
49A6 307A 0010              sta r7,olds(r2)     ;and saving result
49AA 40F9 492A              call @cawl(r1)      ;call this stats guy, r2= ldr.
                          endif
                        endif
49AE 6016               restore r1
                      endif
                    endif
49B0 4992           sub r1,=words       ;next one to do
49B2 8BD0           until minus
                  endrepeat
49B4 4078 40E0    call fdoze            ;sleep a while
49B8 90CB       endrepeat
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 363
FAKES.PLR;1     PAGE 22          Fake Hosts


                ;message generator, sends messages of garbage until enough
                ;words sent.
                ;r2 points at 6 wd leader
                ;keep proper msg generator statistics

                        Page FakVars

                ;message generator statistics

5F88            genmcnt: .blkw 1        ;how many messages ever sent
5F8A            genmtim: .blkw 1        ;when last message finished

FC00 0600               Page FakCode

                ;torture test message lengths

49BA 0008       mlngth:  H8
49BC 0048                H48
49BE 01F0                H1F0
49C0 01F8                H1F8

49C2 1076       routine genm, arg r2, uses r2-r3
49C4 4078 419A    call fjamleader
49C8 7038 5E68    lda r3,mgnl           ;length of msg in wds
49CC 8B17         if minus
49CE 7028 5E44      lda r2,tortm        ;do torture test
49D2 603A 49BA      lda r3,mlngth(r2)+  ;length for this msg
49D6 4BA6           and r2,=6           ;limit range
49D8 3028 5E44      sta r2,tortm
49DC 8A0F           if zero
49DE 7028 5E46        lda r2,tortd      ;pntr to old dest
49E2 7328 001E        and r2, H1E       ;limit these
49E6 607A 5E48        lda r7,statdt(r2)+        ;new host to send to
49EA 3078 5E6E        sta r7,mgsb+hstl
49EE 607A 5E48        lda r7,statdt(r2)+        ;new imp
49F2 3078 5E70        sta r7,mgsb+dstl
49F6 3028 5E46        sta r2,tortd      ;save for later
                    endif
                  endif
49FA 4B38 01FF    and r3,= H1FF         ;max mess len
49FE A2B1         sll r3,1              ;in bytes
4A00 49B2         sub r3,=2             ;0 sends no wds
4A02 702D 0004    lda r2,endi(r5)       ;end of our buf
4A06 712D 0010    sub r2,fakesi(r5)     ;how much data fits
4A0A 4B28 1FFE    and r2,=packm         ;disregard map
4A0E 4E32         repeat while r3 > r2  ;send until enough pkts to fit wds
4A10 8C06
4A12 493A 0002      sub r3,=2(r2)       ;'send' this many
4A16 4078 4236      call jamhol         ;no frills, just jam hole buf
4A1A 90FA         endrepeat
4A1C 323D 0010    addm r3,fakesi(r5)    ;fake length of last
4A20 4078 41F8    call fjamend          ;end msg
4A24 7028 62DA    lda r2,time           ;get the time
4A28 3028 5F8A    sta r2,genmtim        ;save it in msg generator statistics
4A2C 48A1         lda r2,=1
```

```
4A32 782D 000F     ldab r2,bhpid(r5)      ;poke myself
4A36 30A8 00AC     sta r2,@pid
4A3A 6006        endroutine genm
```

```
.comnt |
TENEX Logger routines

Messages sent to TENEX using TCP-Internet header. Listening process
on TENEX prints changes in status, plus all snapshots.
|

     4000       tlogfreq= H4000          ;send config every 8 mins

                ;Dispatch table

4A3C            table tltdsp
4A3C 4A50         tlt1                          ;Common Buses
4A3E 4A54         tlt2                          ;Common Memories
4A40 4A5C         tlt3                          ;Processor Names
4A42 4A60         tlt4                          ;I/O Devices
4A44 4A64         tlt5                          ;RTC's
4A46 4A6C         tlt6                          ;Processor State.
                endtable tltdsp

                ;Main dispatch control.  Called from stats

4A48 1076       routine tlog, arg r2, arg r5
4A4A 6006       endroutine tlog
```

```
                ;subroutine to send a table.
                ;enter with r2 = length in bytes.

4A4C 1076       routine tltsub, arg r2, arg r5
4A4E 6006       endroutine tltsub



                ;Table 1 - Common Buses

4A50 1076       routine tlt1, arg r2, result r2, uses r1, uses r3-r4
4A52 6006       endroutine tlt1
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 366
FAKES.PLR;1     PAGE 25          Fake Hosts

                ;table 2 - Common memories

4A54 1076       routine tlt2, arg r2, result r2, local r5, uses r1, uses r3-r4
4A56 1056
4A58 6056       endroutine tlt2
4A5A 6006


                ;table 3 - Processor names

4A5C 1076       routine tlt3, arg r2, result r2, uses r1
4A5E 6006       endroutine tlt3
```

```
                ;table 4 - I/O devices

4A60 1076       routine tlt4, arg r5, uses r1-r4
4A62 6006       endroutine tlt4
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 368
FAKES.PLR;1     PAGE 27         Fake Hosts

                ;table 5 - RTC's

4A64 1076       routine tlt5, arg r2, result r2, local r5, uses r1, uses r3-r4
4A66 1056
4A68 6056       endroutine tlt5
4A6A 6006


                ;table 6 - processor state

4A6C 1076       routine tlt6, arg r2, uses r1
4A6E 6006       endroutine tlt6
```

```
                ;imp status fake, send status info to NCC
4A70            table tsts                      ;table of things to send
     0001       .lif z VHA                      ;ARPANET things
                  0,0,-1,tipver,sysver,rstrld+foo,cpflgs
     0001       .lif nz VHA                     ;PLATFORM
4A70 0000         0,0,-1,m2#vhaser,sysver,rstrld+foo,cpflgs
4A72 0000
4A74 FFFF
4A76 883E
4A78 40C2
4A7A A320
4A7C 40CA
                endtable tsts

4A7E 1076       routine status, arg r2, uses r1-r4
4A80 4078 419A    call fjamleader       ;arg r2
4A84 4818 80C6    lda r1,=sign+ 0306              ;report code
4A88 4078 4D42    call cksjam                     ;send
4A8C 4078 4C78    call stathd                     ;standard header
     0001       .if nz VHA                      ;VHA serial number check
4A90 7078 00B2    setmap m2,mapvha                ;for VHA serial number
4A94 3078 FC04
                .endc
4A98 48AE         lda r2,=ltsts
                  repeat        ;send things out of table
4A9A 501A 4A70      lda r1,tsts(-r2)
4A9E 9A03           ifnot zero } odd    ;a pointer if possible addr
4AA0 9902
4AA2 7011             lda r1,(r1)
                    endif
4AA4 4078 4D42      call cksjam
4AA8 4EA0           while r2 > =0
4AAA 9CF8
                  endrepeat
4AAC 7078 00B0    setmap m2,maprel
4AB0 3078 FC04
4AB4 7018 8096    lda r1,m2#segcon                ;dead procs
4AB8 7518 9D64    eor r1,m2#procex                ;existing procs
4ABC 7078 00D2    setmap m2,mapv2
4AC0 3078 FC04
4AC4 4078 4D42    call cksjam
4AC8 4890         lda r1,=0                       ;two more blanks
4ACA 4078 4D42    call cksjam
4ACE 4078 4D42    call cksjam
4AD2 4078 4CAC    call trbsb2                     ;send anomalies and buf reports
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 370
FAKES.PLR;1     PAGE 29          Fake Hosts

4AD6 48B0          lda r3,=0
                   repeat
4AD8 604B 635E       lda r4,m2pblk(r3)+
4ADC 9B1C            ifnot minus
4ADE 782C 001B         ldab r2,clockm(r4)        ;line speeds
4AE2 A2A5        sll r2,5
4AE4 781C 009F   ldab r1,rmodn(r4)
4AE8 4A21        add r2,r1
4AEA A7A8        rrl r2, D8
                 ;       rrl r2,3
4AEC 701C 40B4         lda r1,ihytik+foo(r4)     ;routing sent,keep between 0,ff
4AF0 4871              lda r7,r1
4AF2 717C 40B6         sub r7,gotihy+foo(r4)     ;number missed
4AF6 1076              push r7                   ;save this
4AF8 4078 4D2C         call tstjam
4AFC 6016              pop r1                    ;routing misses
4AFE 782C 0004         ldab r2,lnei(r4)
4B02 A7A7              rrl r2,7
4B04 787C 0002         ldab r7,phflag(r4)        ;get line state
4B08 4DF4              eor r7,=phup              ;reverse sense of "up" bit
4B0A A5F3              rra r7,3  ;;phup carry    ;get up bit to carry
4B0C A5A1              rra r2,1  ;;carry sign    ;thence to sign
4B0E 4078 4D2C         call tstjam
4B12 9008            else
4B14 4890              lda r1,=0                 ;no routing on non-line
4B16 4078 4D42         call cksjam
4B1A 4818 8000         lda r1,=sign              ;default is dead line
4B1E 4078 4D42         call cksjam
                     endif
4B22 7638 6324       until r3 >= modems
4B26 92D9
                   endrepeat
4B28 4818 0800     lda r1,= H800
4B2C 3018 5E8C     sta r1,stsb+statf
4B30 7018 5F86     lda r1,trblck
4B34 4078 41F8     call fjamend
4B38 6006        endroutine status
```

```
                ;modem and host traffic statistics
4B3A 1076       routine thrurpt, arg r2, uses r2-r4
4B3C 4078 419A    call fjamleader                       ;arg r2
4B40 4818 80CD    lda r1,=sign+ 0315            ;report code for NU
4B44 4078 4D42    call cksjam
4B48 7018 62DE    lda r1,sync                   ;custom-made statistics header
4B4C 4078 4D42    call cksjam
4B50 7018 6322    lda r1,hosts
4B54 4998         sub r1,=nfh2
4B56 7618 6324    if r1 < modems                ;compute host (modem) count right
4B5A 8203
4B5C 7018 6324      lda r1,modems
                  endif
4B60 4078 4C8A    call stath2
4B64 7018 6522    lda r1,idles                  ;idle count
4B68 4078 4D42    call cksjam
4B6C 4890         lda r1,=0
4B6E 4078 4D42    call cksjam                   ;blank: #tip users
4B72 48B0         lda r3,=0
                  repeat
4B74 4890           lda r1,=0
4B76 604B 635E      lda r4,m2pblk(r3)+
4B7A 9B03           ifnot minus
4B7C 701C 4040        lda r1,thrupt+foo(r4)     ;modem throughput
                    endif
4B80 4078 4D42      call cksjam
4B84 701C 400A      set r1 = hiword+foo(r4)     ;high-order thruput
4B88 702C 4008      set r2 = loword+foo(r4)     ;low-order (get both at once)
4B8C 1026           save r2                     ;save it
4B8E 4078 4D42      call cksjam                 ;send high
4B92 6016           restore r1                  ;get back low
4B94 4078 4D42      call cksjam                 ;and send it
4B98 7638 6324      until r3 >= modems
4B9C 92EC
                  endrepeat
4B9E 48B8         lda r3,=nfh2          ;throughput for reals only
                  repeat
4BA0 704B 6326      lda r4,h2pblk(r3)   ;do host throughput
4BA4 9B04           if minus } r3 >= hosts      ;if no host here
4BA6 7638 6322
4BAA 9208
4BAC 48C8             lda r4,=<hctrl/words>-2
                      repeat              ;zero the host throughput files
4BAE 4890               lda r1,=0
4BB0 4078 4D42          call cksjam
4BB4 49C1               sub r4,=1
4BB6 8AFC             until zero
                      endrepeat
4BB8 9009           else
4BBA 482C 0010        lda r2,=hctrl-4(r4)       ;A genuine host
                      repeat                    ;send out host thruput stats
4BBE 601C 403A          lda r1,htpmtn+foo(r4)+
4BC2 4078 4D42          call cksjam
4BC6 4E42             while r4 < r2
4BC8 92FB
                      endrepeat
                    endif
```

```
4BCA 4873           lda r7,r3                   ;a modem with this number?
4BCC 49F8           sub r7,=nfh2
4BCE 604B 6356      lda r4,m2pblk-nfh2(r3)+
                    if minus } r7 >=  modems
```

```
4BD2 9B04
4BD4 7678 6324
4BD8 9207
4BDA 4890              lda r1,=0                ;no -- just send zeros
4BDC 4078 4D42         call cksjam
4BE0 4078 4D42         call cksjam
4BE4 900D            else
4BE6 701C 400A         lda r1,hiword+foo(r4)    ;zero the counts
4BEA 707C 4008         lda r7,loword+foo(r4)    ;X16 compatibility crock
4BEE 8B02              if minus                 ;this is the WEIRD X16 double precision
4BF0 4A91                add r1,=1              ;you figure this out}
                       endif
4BF2 1076              push r7                  ;send high order word first
4BF4 4078 4D42         call cksjam
4BF8 6016              pop r1                   ;now the low order word
4BFA 4078 4D42         call cksjam
                     endif
4BFE 7078 6324       lda r7,modems              ;decide if we're done
4C02 4AF8            add r7,=nfh2
4C04 7638 6322     until r3 >= hosts & r3 >= r7
4C08 92CC
4C0A 4E37
4C0C 92CA
                   endrepeat
4C0E 7018 5F86     lda r1,trblck
4C12 4078 41F8     call fjamend
4C16 6006        endroutine thrurpt
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 373
FAKES.PLR;1     PAGE 31         Fake Hosts

                ;trap report: finds traps and phantom traps if any and reports
                ;them to NCC. Turns itself off if no more traps, otherwise freq
                ;is left zero to empty tables quickly.

     0001       .if z PSE                       ;ARPANET case
                  nilops=cilops
                  nilend=cilend
                .iff ;z PSE                     ;PSE case
     6166         nilops=pilops
     6206         nilend=pilend
                .endc ;z PSE

4C18 1076       routine traprp, arg r2, uses r1-r4

                   ;inserted all statements above call to fjamleader
                   ; to prevent sending zero traps to NCC - 10/26/86 BKM

4C1A 7018 6166     lda r1,nilops(r0)            ;check first word of traps
4C1E 9A26          ifnot zero                   ;if trap then send it else don't
4C20 4078 419A       call fjamleader            ;arg r2
4C24 4818 80C8       lda r1,=sign+ 0310         ;pluribus "310" report
4C28 4078 4D42       call cksjam                        ; containing traps
4C2C 4078 4C78       call stathd                        ;header stuff
4C30 4078 4CAC       call trbsb2                        ;anomaly word and buf cnts
4C34 4838 008C       lda r3,=<nilend-nilops-lcilbuf>    ;last trap block
                     repeat
4C38 707B 6166         until nilops(r3)         ;trap here
4C3C 8A04
4C3E 4938 0014         sub r3,=lcilbuf          ;to next latest buffer
4C42 8AFB              until zero                       ;got to first
                     endrepeat
     0001          .lif nz PSE                 ;PSE special trap case
4C44 3038 6164       sta r3,pilovp+words              ;update RR pointer
4C48 48CA            lda r4,=lcilbuf/words            ;how much to send
                     repeat
4C4A 601B 6166         lda r1,nilops(r3)+
4C4E 4078 4D42         call cksjam
4C52 49C1              sub r4,=1                        ;another sent
4C54 8AFB              until zero
                     endrepeat

4C56 304B 6152       sta r4,nilops-lcilbuf(r3) ;;r4 = 0 ;clear and free ILLOP block
4C5A 4838 0800       lda r3,= H800                    ;frequency for this stat
4C5E 3038 5EB4       sta r3,trpsb+statf         ;reset frequency
4C62 7018 5F86       lda r1,trblck
4C66 4078 41F8       call fjamend
                   endif
4C6A 6006       endroutine traprp


4C6C 1076       routine holdwd, arg r2
4C6E 4078 419A     call fjamleader
4C72 4078 41F8     call fjamend
4C76 6006       endroutine
```

```
                ;auxilliary stats routines

                ;write out headers of standard info
4C78 1076       routine stathd, uses r1
4C7A 7018 62DE    lda r1,sync                   ;time
4C7E 4078 4D42    call  cksjam
4C82 7018 6322    lda r1,hosts
4C86 4998         sub r1,=nfh2
4C88 9002       entry stath2, uses r1
4C8A 1076
4C8C A691         srl r1,1
4C8E 4078 4D42    call  cksjam
4C92 4894         lda r1,=nfh                   ;# fake hosts
4C94 4078 4D42    call  cksjam
4C98 7018 6324    lda r1,modems
4C9C A691         srl r1,1
4C9E 4078 4D42    call  cksjam
4CA2 4818 007F    lda r1,=nimp                  ;# imps
4CA6 4078 4D42    call cksjam
4CAA 6006       endroutine stathd
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 375
FAKES.PLR;1     PAGE 33         Fake Hosts

                ;buffer allocation stats, anomalies, host and modem
                ;counts
4CAC 1076       routine trbsb2, uses r1, uses r3-r4
4CAE 7018 62AE    lda r1,anom                   ;anomalies word
4CB2 3018 5F84    sta r1,banom                  ;copy for comparison
4CB6 4078 4D42    call  cksjam
4CBA 7018 63CE    lda r1,maxnf                  ;# free bufs.
4CBE 4078 4D42    call  cksjam
4CC2 7018 63C8    lock r1,nf-foo                ;read w/o locking
4CC6 9AFE
4CC8 7618 63CE    if r1 <= maxnf
4CCC 9C0A
4CCE 7118 63D0      sub r1,minf                 ;to get remaining slush pool
4CD2 4E18 0028      if r1 >= = D40
4CD6 9203
4CD8 4818 0028        lda r1,= D40              ;make it good
                    endif
4CDC 8B02           if minus
4CDE 4890             lda r1,=0                 ;should never get here
                    endif
                  endif
4CE0 4078 4D42    call  cksjam
4CE4 7018 63DE    lda r1,cntrs+nsf              ;get excess s/f
4CE8 8B02         if minus
4CEA 4890           lda r1,=0
                  endif
4CEC 4078 4D42    call  cksjam
4CF0 7018 63DC    lda r1,cntrs+nre
4CF4 7038 63D4    lda r3,nal
4CF8 4913         sub r1,r3                     ;we include nal in nre
4CFA 8B02         if minus
4CFC 4890           lda r1,=0                   ;else zero if in min
                  endif
4CFE 4078 4D42    call  cksjam
4D02 4813         lda r1,r3
4D04 4078 4D42    call  cksjam
4D08 48B8         lda r3,=nfh2
                  repeat
4D0A 48F3           lda r7,=3           ;default status
4D0C 6048 6326      lda r4,h2pblk(r3)+
4D10 9B04           ifnot minus
4D12 787C 001F        ldab r7,hihd(r4)
4D16 4BFF             and r7,= HF
                    endif
4D18 A294           sll r1,4
4D1A 4C17           ior r1,r7
4D1C 4FB7           next if r3 .bit. =7
4D1E 8AF6
4D20 4078 4D42      call cksjam
4D24 7638 6322      until r3 >= hosts
4D28 92F1
                  endrepeat
4D2A 6006       endroutine trbsb2
```

```
                ;TSTJAM - do a little for line stuff, then:
                ;CKSJAM include arg in checksum and then jam arg

4D2C 1076       routine tstjam,arg r1-r2

4D2E 4F18 FF00    if r1 .bit. = HFF00
4D32 9A06
4D34 8B03           if minus
4D36 4890             clear r1
4D38 9003            else
4D3A 4818 00FF         lda r1,= HFF
                     endif
                   endif
4D3E 4C12          ior r1,r2

4D40 9002       entry cksjam, arg r1
4D42 1076
4D44 3118 5F86     subm r1,trblck               ;do checksum
4D48 4078 414A     call  fjam                   ;put into a packet.
4D4C 6006       endroutine cksjam

                ;diag report: gets buf that failed soft cksm and send
                ;to NCC for analysis.
4D4E 1076       routine diagrp,uses r1-r3
                  repeat
4D50 4818 63C0      lda r1,=sckq               ;cksum error diagnostic dump
4D54 4078 154A      call deque,whf2h
4D58 4000
4D5A 9A22           until fail
4D5C 1026           push r2                    ;save chainword
4D5E 1016           push r1                    ;save buf pointer
4D60 4828 5EBA      lda r2,=dgsb               ;for sending leader
4D64 4078 419A      call fjamleader            ;arg r2
4D68 4818 8003      lda r1,=sign+3             ;report code
4D6C 4078 4D42      call cksjam
4D70 6026           pop r2                     ;buf pntr
4D72 4812           lda r1,r2
4D74 4078 4D42      call cksjam                ;send buffer as well
4D78 483A 00A0      lda r3,=buflen(r2)
                    repeat
4D7C 6012             lda r1,(r2)+             ;get word from buf.
4D7E 4078 4D42        call cksjam
4D82 4E23             until r2 >= r3
4D84 92FC
                    endrepeat
4D86 7018 5F86      lda r1,trblck
4D8A 4078 41F8      call fjamend
4D8E 6026           pop r2                     ;get chainword
4D90 4078 131E      call flushb,whf2h
4D94 4000
4D96 48F0           lda r7,=0
4D98 3078 5F86      sta r7,trblck              ;restart checksum
4D9C 90DA         endrepeat
4D9E 48F0         lda r7,=0
4DA0 3078 5EC6    sta r7,dgsb+snon             ;turn us off
4DA4 6006       endroutine diagrp
```

```
                ;*** fake config ***

4DA6 1076       routine fakcon
4DA8 4896         lda r1,=nfh2-2              ;2 * how many fakes.
                  repeat
4DAA 7059 636E      lda r5,v2pblk(r1)         ;fake ioblock pointer.
4DAE 8905           if odd                    ;fake needs initialization.
4DB0 4078 4E00        call gofh               ;initialize fake.
4DB4 4078 1B42        call csleep             ;nap a while.
                    endif
4DB8 4992           sub r1,=2                 ;next fake.
4DBA 8BF8           until minus
                  endrepeat
4DBC 4078 54DE    call conclk                 ;configure RTC's
4DC0 6006       endroutine fakcon


4DC2 1076       routine bldfh, arg r1, arg r5
4DC4 4871         lda r7,r1                   ;Host number calculation.
4DC6 A6F1         srl r7,1
4DC8 4A78 02FC    add r7,=hostid+400-nfh      ;Fakes are OFC - OFF
4DCC 3075         sta r7,(r5)                 ;Device Type and number.

4DCE 4A18 0030    add r1,=fakip0              ;PID levels.
4DD2 381D 0007    stab r1,statih+1(r5)
4DD6 4A98         add r1,=fakop0-fakip0
4DD8 381D 000D    stab r1,statoh+1(r5)
4DDC 4A18 FFE0    add r1,=hbpid0-fakop0
4DE0 381D 000E    stab r1,hbpid(r5)
4DE4 4825         lda r2,r5
4DE6 4078 1160    call inbase,waitw
4DEA 3C7E
4DEC 4A18 FFF0    add r1,=bhpid0-hbpid0
4DF0 381D 000F    stab r1,bhpid(r5)
4DF4 4078 1160    call inbase,dozew
4DF8 3C04
4DFA 4A18 FFF8    add r1,=0-bhpid0            ;Recover Host index.
4DFE 6006       endroutine bldfh
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 378
FAKES.PLR;1     PAGE 36         Fake Hosts

4E00 1076       routine gofh, arg r1            ;r1 is fake index.
4E02 7049 4E3A    lda r4,ifakeh(r1)             ;host block address.
4E06 7059 4E32    lda r5,iiobl3(r1)             ;fake io block
4E0A 3049 6326    sta r4,h2pblk(r1)             ;host parameter block address.
4E0E 3059 636E    sta r5,v2pblk(r1)             ;fake host io block address.

4E12 4078 4DC2    call bldfh                    ;set up constants in io block.

4E16 48F1         set r7 = #1                   ;So an init happens
4E18 307D 4014    sta r7,lockfd(r5)             ;release fake for use by all.
4E1C 307D 4016    sta r7,lockfw(r5)
4E20 6006       endroutine gofh

4E22            table waitpc
4E22 44C0         fht                           ;Fake to TTY.
4E24 462C         fhd                           ;Fake to DDT.
4E26 46FC         fh2suc                        ;Fake to Packet Core.
4E28 48DA         discard                       ;Fake to Discard.
                endtable waitpc

4E2A            table dozepc                    ;Initial Jam side PC.
4E2A 4534         fth                           ;TTY to Fake.
4E2C 4668         fdh                           ;DDT to Fake.
4E2E 4832         fh2jam                        ;Packet Core to Fake.
4E30 494E         stats                         ;Statistics to Fake.
                endtable dozepc

4E32            table iiobl3                    ;Fake IO block.
4E32 6E30         iobl3                         ;TTY Fake.
4E34 6EB2         iobl4                         ;DDT Fake.
4E36 6F34         iobl5                         ;Packet Core Fake.
4E38 6FB6         iobl6                         ;Discard/Stats Fake.
                endtable iiobl3

4E3A            table ifakeh                    ;Fake Host Parameter blocks.
4E3A F0C0         sign+bhi0
4E3C F130         sign+bhi1
4E3E F1A0         sign+bhi2
4E40 F210         sign+bhi3
                endtable ifakeh
```

```
.INSERT "FAKREL"
    .INSRT FAKREL
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 380
FAKREL.PLR;1    PAGE 1  IMP Fakes Reliability

                .stitle IMP Fakes Reliability
                ;******* timeout on fake page ********

                ;timeout dispatch table
4E42 4DA6       faktab: fakcon  ;config routine
4E44 1CC6               tickin
4E46 4E68               rnal
4E48 5A0A               basetp
4E4A 51B2               dedh
4E4C 5440               messto
4E4E 4EB2               rcntrs
4E50 50AA               buft
4E52 5356               bt0
4E54 5140               impdwn
4E56 5928               gostat
4E58 548E               rihs
4E5A 54C6               rfake
4E5C 59F2               bktst
4E5E 52BE               trbtim
4E60 5206               lites
4E62 5ADE               nschk
4E64 1CBC               tosst
4E66 0000               0
```

```
                ;maintain the allocate count nal
FC00 0600               Page FakCode

4E68 1076       Routine rnal,uses r1-r3
4E6A 48B0         lda r3,=0                     ; accumulates nal
4E6C 48A1         lda r2,=1                     ; idle reasst value
4E6E 4818 0170    lda r1,=emsstk-lrblk-messtk   ; Last reasblk index.
                  repeat                        ; for each reassembly block
4E72 7E29 6CB3      cmpb r2,messtk+reasst(r1)   ; idle block?
4E76 910D           ifnot equal                 ; ifnot, do count allocates here.
4E78 7079 ACB0        lock messtk+reaslk(r1)    ; lock block to count.
4E7C 9AFE
4E7E 7E29 6CB3          cmpb r2,messtk+reasst(r1)       ; just freed?
4E82 9105               ifnot equal             ; No, count nal.
4E84 7A39 6CB9            addb r3,messtk+ral(r1)
4E88 7939 6CB2            subb r3,messtk+rsf(r1)
                        endif
4E8C 3009 ACB0        unlock messtk+reaslk(r1)
                    endif
4E90 4918 0010      sub r1,=lrblk               ; next reas blk.
4E94 8BEF          until minus
                  endrepeat

4E96 4818 63D6    lda r1,=trenda                ; which trend
4E9A 7138 63D4    sub r3,nal                    ; how far off?
4E9E 4078 4FFC    call fixer                    ; smooth fixing for fuzzy count.
4EA2 7018 A3C8    lock r1,nf                    ; lock the counts.
4EA6 9AFE
4EA8 3238 63D4      addm r3,nal                 ; now adjust
4EAC 3018 A3C8    unlock r1,nf                  ; unlock
4EB0 6006       endroutine rnal
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 382
FAKREL.PLR;1    PAGE 3  IMP Fakes Reliability

                ;count modem queues to verify slots count
                ;can be made a routine by replacing uses of r6 with r3

FC00 0200               Page DDTCode
5440 7018 6324  mqcnt:  lda r1,modems
5444 5049 635E  mqcnt1: lda r4,m2pblk(-r1)
5448 9B35                bm mqcnt8
544A 787C 0002          ldab r7,phflag(r4)      ;is line down?
544E 4FF4               tst r7,=phup
5450 9A31                bz mqcnt8              ;yes, so chnbsy may be wrong
5452 487C 0082          lda r7,=chnbsy(r4)
5456 706C 001C          lda r6,maxchn(r4)       ;calc slots in use
545A A6E1               srl r6,1
545C 49E1               sub r6,=1
545E 705C 4000          lda r5,lockm(r4)
5462 9AFE                bz .-4
5464 716C 0050          sub r6,slots(r4)
5468 4826               lda r2,r6       ;account for busy channels
546A 6057       mqcnt2: lda r5,(r7)+    ;;chnbsy
546C 9A05                bz mqcnt4
546E 49A1       mqcnt3: sub r2,=1
5470 4B5D FFFF          and r5,=-1(r5)
5474 8AFD                bnz mqcnt3
5476 4E7C 0092  mqcnt4: cmp r7,=chnbsy+<nmdlim*words/ D16>(r4)
547A 81F8                bne mqcnt2
547C 705C 0026          lda r5,snding(r4)       ;count buffer, if any
5480 9A02                bz mqcnt5
5482 49E1                sub r6,=1      ;else count it
5484 485C 0034  mqcnt5: lda r5,=ssentq(r4)      ;these are busy channels
5488 4078 54BA          jsb r7,countq
548C 4E26               cmp r2,r6       ;this checks chnbsy
548E 810B                bne mqcnt6     ;loses
5490 485C 003C          lda r5,=sregq(r4)       ;now check slots
5494 4078 54BA          jsb r7,countq
5498 485C 0038          lda r5,=spriq(r4)
549C 4078 54BA          jsb r7,countq
54A0 4866               lda r6,r6       ;checks slots count
54A2 9A06                bz mqcnt7      ;matches
54A4 E4C8       mqcnt6: Trap 2310,<;modem state mismatch - page 380>
54A6 4868 02F8          lda sp,=lstack  ;need a stack
54AA 4078 1CF4          call killin
54AE 300C 4000  mqcnt7: sta r0,lockm(r4)
54B2 4811       mqcnt8: lda r1,r1
54B4 8AC8                bnz mqcnt1
54B6 4008 1BEE          tr tossx

                ;count a queue into r6

54BA 7055       countq: lda r5,(r5)     ;get next buffer
54BC 9903                bo counq3      ;empty queue
54BE 49E1               sub r6,=1
54C0 8BFD                bnm countq     ;falls though on error
54C2 4007       counq3: jmp (r7)

FC00 0200          dstolist mqcnt
4EDE 5440
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 383
FAKREL.PLR;1    PAGE 4  IMP Fakes Reliability

.comnt |
RCNTRS
maintain the buffer counters
try for minmi=m+2lm+1,minsf=3lm,minhi=lh,minbak=3,minre=1+8lh,minrut=4
 where m=total modems, lm=live lines, lh=live hosts(m+5lm+9lh+9)
if too few buffers use minmi=m+1,minsf=m,minhi=2,minbak=1,minre=9,
 minrut=4; then put sum of these in minf(2m+17)
|

                ;routine rcntrs

4EB2 1076       routine rcntrs, uses r1-r5
4EB4 4858 FFFD    lda r5,=-3
4EB8 3058 646C    sta r5,newcts+nbak    ;default
4EBC 4818 FFFF    lda r1,=-1            ;minmi
4EC0 48B0         lda r3,=0             ;minsf
4EC2 7058 6324    lda r5,modems         ;to count modems
                  repeat
4EC6 504D 635E      lda r4,m2pblk(-r5)  ;any param blk?
4ECA 9B08           ifnot minus
4ECC 4991             sub r1,=1         ;1 for minmi
4ECE 787C 0002        if byte phflag(r4) .bit. =phup    ;if line is up
4ED2 4FF4
4ED4 9A03
4ED6 4992               sub r1,=2       ;bump minmi
4ED8 49B3               sub r3,=3       ; and minsf
                      endif
                    endif
4EDA 88F6           until loop
                  endrepeat
4EDC 1036         push r3               ;remember minsf
4EDE 48B0         lda r3,=0             ;next nvdhi
4EE0 48A0         lda r2,=0             ;next nhi
4EE2 7058 6322    lda r5,hosts          ;count hosts
4EE6 49D8         sub r5,=nfh2          ;reals only
                  repeat
4EE8 504D 632E      lda r4,h2pblk+nfh2(-r5)     ;next param blk
4EEC 9B0A           ifnot minus
4EEE 787C 001F        ifnot byte hihd(r4)       ;ready line?
4EF2 8A02
4EF4 49A1               sub r2,=1       ;;hostup ;1 for minhi
                      endif
4EF6 707C 0002        if hostyp(r4) = #hosvdh   ;VDH gets 3 each
4EFA 4EF4
4EFC 8102
4EFE 49B3               sub r3,=3
                      endif
                    endif
4F00 88F4           until loop
                  endrepeat
```

```
4F02 6046          pop r4                  ;get back minsf
4F04 4852          lda r5,r2               ;for minre:
4F06 A2D3          sll r5,3                ;8 lh
4F08 49D1          sub r5,=1               ;+ 1
4F0A 48F7          lda r7,=7               ;minf sum, start w/nbak+nrut
4F0C 4971          sub r7,r1               ;add up all the counts
4F0E 4972          sub r7,r2
4F10 4973          sub r7,r3
4F12 4974          sub r7,r4
4F14 4975          sub r7,r5
4F16 7178 40D0     sub r7,intpti           ;add ptip counter
4F1A 7178 40D2     sub r7,intpit           ;add other ptip counter
4F1E 7678 63CE     if r7 >= maxnf          ;more than total buffs
4F22 9216
4F24 A491            sra r1,1              ;cut r1 in half
4F26 4841            lda r4,r1             ;minsf = <m+2lm+1>/2
4F28 4991            sub r1,=1             ;minmi = <m+2lm+1>/2+1
4F2A 4828 FFFE       lda r2,=-2            ;minhi = 2
4F2E A4B1            sra r3,1              ;minvdh = <3v>/2
4F30 4858 FFFF       lda r5,=-1            ;minbak = 1
4F34 3058 646C       sta r5,newcts+nbak
4F38 4858 FFF7       lda r5,=- D9          ;minre = 9
4F3C 4878 0010       lda r7,= D16          ;minre+minhi+minbak+minrut
4F40 4971            sub r7,r1             ;+minmi
4F42 4973            sub r7,r3             ;+minvdh
4F44 4974            sub r7,r4             ;+minsf
4F46 7178 40D2       sub r7,intpit         ;and TIP counters
4F4A 7178 40D0       sub r7,intpti
                   endif
4F4E 3078 63D0     sta r7,minf             ;save new minf
```

```
4F52 3028 646E    sta r2,newcts+nhi
4F56 4878 6460    lda r7,=newcts        ;speed things a bit
4F5A 2017         sta r1,(r7)+  ;;nmi
4F5C 2037         sta r3,(r7)+  ;;nvdhi
4F5E 2057         sta r5,(r7)+  ;;nre
4F60 2047         sta r4,(r7)+  ;;nsf
4F62 4848 FFFC    lda r4,=-4            ;nrut always
4F66 2047         sta r4,(r7)+  ;;nrut
4F68 7058 40D2    lda r5,intpit         ;for ntipit
4F6C 2057         sta r5,(r7)+  ;;ntipit
4F6E 7078 40D0    set newcts+ntipti = intpti
4F72 3078 648E
4F76 4078 1C34    call tsleep           ;take a break
4F7A 7018 63CC    lda r1,maxbuf         ;index thru where table
                  repeat
4F7E 48F1           lda r7,=1           ;for addm
                    repeat
4F80 5039 9960        lda r3,where(-r1) ;next buff free?
4F84 9A05             ifnot zero
4F86 4B38 003E          and r3,=whctr
4F8A 327B 6460          addm r7,newcts(r3)
                      endif
4F8E 4F18 003E        while r1 .bit. = H3E      ;until sleep time
4F92 8AF7
                    endrepeat
4F94 3018 6514      sta r1,totmp1       ;remember r1
4F98 4078 1C34      call tsleep         ;and sleep again
4F9C 7018 6514      lda r1,totmp1       ;more to count?
4FA0 8AEF           until zero
                  endrepeat
4FA2 7038 63D4    lda r3,nal            ;remember nal is in nre
4FA6 3238 6464    addm r3,newcts+nre
                ;more rcntrs over leaf
```

```
                ;more rcntrs
4FAA 4858 0044    lda r5,=numcts*words  ;index for adjusting
                  repeat                ;adjust counters
4FAE 703D 645E      lda r3,newcts-words(r5)     ;next one
4FB2 713D 63D6      sub r3,cntrs-words(r5)      ;error in it
4FB6 481D 641A      lda r1,=trends-words(r5)    ;ptr to trend
4FBA 4078 4FFC      call fixer            ;fuzzy adjustment
4FBE 7078 A3C8      lock r7,nf            ;lock counts
4FC2 9AFE
4FC4 123D 63D8      addm r3,cntrs(-r5)    ;adjust it
4FC8 3078 A3C8      unlock r7,nf          ;and unlock again
4FCC 88F1           until loop            ;table done.
                  endrepeat
4FCE 4078 1C34    call tsleep             ;wait a little more
4FD2 4818 0044    lda r1,=numcts*words    ;will loop thru again
4FD6 7038 63CE    lda r3,maxnf            ;total buffers in system
4FDA 7048 A3C8    lock r4,nf              ;lock counts
4FDE 9AFE
                  repeat                  ;include count in nf
4FE0 5029 63D8      lda r2,cntrs(-r1)     ;next count
4FE4 9B02           ifnot minus           ;counter in shared pool.
4FE6 4932             sub r3,r2           ;adjust new nf
                    endif
4FE8 88FC           until loop
                  endrepeat
4FEA 4934         sub r3,r4               ;error in nf
4FEC 4818 63D2    lda r1,=trendf          ;ptr to trend
4FF0 4078 4FFC    call fixer              ;fuzzy adjust
4FF4 4A43         add r4,r3               ;change nf
4FF6 3048 A3C8    unlock r4,nf            ;unlock with fixed nf
4FFA 6006       endroutine rcntrs
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 387
FAKREL.PLR;1    PAGE 8  IMP Fakes Reliability

.comnt  |
FIXER
Maintain fuzzy counters.  enter with a error in
r3, ptr to trend in r1.  return amount to fix in r3.  fix if
trend has accumulated more than  fclip in some direction.
maintain trend.  |

4FFC 1076       routine fixer,arg r1,arg r3,local r2
4FFE 1026
     000B       fclip= D11      ; must be less than 14

5000 7021         lda r2,(r1)   ; fetch old trend
5002 A0A1         sla r2,1
5004 7221         add r2,(r1)   ; times 3
5006 A0B4         sla r3,4      ; 4 bits fraction
5008 4A23         add r2,r3     ; new trend
500A A4A2         sra r2,2      ; is exponentially weighted average
500C 4832         lda r3,r2
500E 8B02         if minus
5010 4AB5           add r3,=-<2*< H10-fclip>- H10+1>     ; ha}
                  endif
5012 4AB5         add r3,= H10-fclip
5014 4B38 FFF0    and r3,=- H10
5018 4923         sub r2,r3     ; fixed trend
501A 3021         sta r2,(r1)   ; restore trend
501C A4B4         sra r3,4      ; amount to fix cntr

501E 6026       endroutine
5020 6006
```

```
                ;check a point word
                ; r2/chain word
5022 1076       routine chkpnt,arg r2,local r3
5024 1036
5026 0210         rst %f1                       ; default is success.
5028 703A FCB0    lda r3,point-chain(r2)        ; get the point word.
502C 4B38 FE00    and r3,=mapmsk
5030 7638 00D2    if r3 = mapv2                 ; Buffer on V2 page.
5034 811C
5036 707A FCB0      lda r7,point-chain(r2)      ; get point again.
503A 4D37           eor r3,r7                   ; remember offset for later.
503C 4AFA           add r7,=buflen/ H10
503E 717A FCA6      sub r7,<point-chain>-<5*words>(r2)
5042 9A14           ifnot zero
5044 A2B4             sll r3,4                  ; Adjust to ease checking.
5046 4938 02B0        sub r3,=0#<mnbuf2-buflen>
504A 4E38 0F70        if r3 <> =0#point & r3 <> =0#chain & r3 <> =0#chan
504E 910E
5050 4E38 12C0
5054 910B
5056 4E38 1610
505A 9108
505C 4E38 1960          if r3 <> =0#where & r3 <> =0#flushd
5060 9105
5062 4E38 1CB0
5066 9102
5068 0290                 sst %f1               ; Point word is bad.
                        endif
                      endif
                    endif
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 389
FAKREL.PLR;1    PAGE 10          IMP Fakes Reliability

506A 9018          else                              ; Buffer on other page.
                     ;;rst %e
506C 4818 000C       lda r1,=bfmpnd-bufmap            ; On any buffer page?
                     repeat
5070 5079 563A         lda r7,bufmap(-r1)             ; Get pointer to page.
5074 9903              until even & r3 = (r7)         ; buffer on buffer page.
5076 7637
5078 9102
507A 88FB              until loop                          ; Buffer not on buffer page.
                     endrepeat

507C 9103            ifnot equal                      ; Buffer not on buffer page.
507E 0290              sst %f1                        ; Broken buffer pointer.
5080 900D            else
5082 753A FCB0         eor r3,point-chain(r2)         ; Offset on page.
5086 4E38 01F6         if r3 <> =< H2000-buflen>/ H10 ; Not first on page
508A 9108
508C 703A FCAE           lda r3,point-chain-2(r2)           ; Point of prev. buffer.
5090 713A FCB0           sub r3,point-chain(r2)             ; less this point.
5094 4EBA                if r3 <> =buflen/ H10              ; Difference wrong
5096 9102
5098 0290                  sst %f1                          ; Whoops, flag as error.
                         endif
                       endif
                     endif

509A 8506          if f1                                    ; pointword bad.
509C E101            Trap 401,<;smashed buf ptr--may cause retrans - page 387>
509E 4838 FFFF       lda r3,=-1
50A2 3038 80BE       sta r3,m2#type4k                       ; smash v2 type word.
                   endif
50A6 6036        endroutine
50A8 6006
```

```
.comnt |
Test the buffer system.  First check each buffer for timeout and mark
it.  Then check the consistency of the free list, salvaging any valid
prefix if possible.  This whole process should take BUFCYCLE / 16
ticks.
|

     07FF       bufcycle = packm/4
                ;# ticks per buffer timeout cycle * 16.
                ;must be one less than a power of 2 and > maxbuf.

50AA 1076       routine buft
                begin buftick
50AC 7048 64AC    lda r4, buftim                ;local copy
                  repeat                        ;until end of buffers
50B0 4B48 07FF      and r4,=bufcycle            ;keep buffer timeout cycle reasonable
50B4 482C 92C0      lda r2,=chain(r4)           ;r2 is chain ptr corresp. to offset in r4
50B8 7648 63CC      until r4 >= maxbuf          ;end of buffers
50BC 8213
50BE 4078 5022      call chkpnt                 ;check this buffer's point word
50C2 703A 09F0      lda r3,flushd-chain(r2)
50C6 8906           if odd                      ;still marked
50C8 707A 06A0        lda r7,where-chain(r2)    ;fyi
50CC E281             Trap 1201,<;Recovd a timed-out buf - page 388>
50CE 4078 5106        call flpool               ;free this buffer
                    endif
50D2 48F1           lda r7,=1                   ;now mark buffer by making it odd
50D4 347A 09F0      iorm r7,flushd-chain(r2)
50D8 4AC2           add r4,=2                   ;next iteration, look at next buffer
50DA 4FCE           ifnot r4 .bit. = HE         ;each tick do only 8 buffers
50DC 8A02
50DE 9011             leave buftick
                    endif
50E0 90E8         endrepeat

                  ;if just finished marking buffers, check freelist
50E2 810D         if equal                      ;(see previous "until")
50E4 4078 3CF8      call chkfree
50E8 8A06           if fail                     ;salvage what we can from free list
50EA 3058 A3C6        sta r5,freend             ;last good free buffer (set by chkfree)
50EE 9903             if even                   ;nonempty list
50F0 48A1               lda r2,=1               ;terminate list here
50F2 3025               sta r2,(r5)
                      endif ;even
                    endif ;fail
50F4 3018 A3C4      unlock r1,free
50F8 4B48 FFF0      and r4,= HFFF0              ;must cycle around to 0
                  endif ;equal
50FC 4A48 0010    add r4,= H10                  ;keep counting to remain on schedule
                end buftick
5100 3048 64AC  sta r4,buftim                   ;remember for next tick
5104 6006       endroutine buft
```

```
.comnt |
Check consistency of free list.  Returns last good free buffer in r5 or
1 if there are none.  Locks FREE.  Returns failure if there is a
problem with the free list or if the free list is empty and freend was
unlocked.  This is a bum; effectively a fail return is taken iff r5
should be stored in FREEND.  In the case where r5 is even, since it
points to the last good buffer the list, the list should be terminated
there.
|

                    page LCode

3CF8 1076       routine chkfree, result r5, uses r1-r3
3CFA 4078 5022    call chkpnt                   ;check point wd for junk
3CFE 48B0         lda r3,=0                     ;counts free list. 1 more than # found
3D00 48D1         lda r5,=1                     ;r5 is last good free buf. So far, none.
3D02 7018 A3C4    lock r1,free
3D06 9AFE
3D08 9915         if even                       ;nonempty free list
3D0A 4821           lda r2,r1
                    repeat                      ;until end of free list
3D0C 707A 06A0        if where-chain(r2)
3D10 9A05
                        Trap 1302,<;free list buf err--WHERE nonzero - page 389>
3D12 E2C2
3D14 6076               fail return
3D16 4FF0
3D18 4007
                      endif
3D1A 300A 09F0        sta r0,flushd-chain(r2)   ;"unmark" this buffer so won't time out
3D1E 4AB2             add r3,=2                 ;count this buffer
3D20 4852             lda r5,r2                 ;last good buf so far
3D22 7022             lda r2,(r2)               ;get next buffer
                      if r3 > maxbuf            ;surely in a loop (maxbuf = 2*no. of bufs)
3D24 7638 63CC
3D28 8C03
3D2A E208               Trap 1010,<;free list in loop - page 389>
3D2C 90F4               fail return
                      endif
3D2E 89EF             until odd                 ;(from lda r2,(r2)).
                    endrepeat
3D30 9007         else                          ;empty free list
3D32 7078 A3C6      if freend                   ;if nobody else is, check freend
3D36 9A04
3D38 9902             if even
3D3A E209               Trap 1011,<;lost the free list - page 389>
                      endif
3D3C 90EC             fail return
                    endif ;freend
                  endif ;even
3D3E 6006       endroutine chkfree
```

```
                ;routine used by fakes only, to free a buffer to shared pool


FC00 0600               page FakCode

5106 1076       routine flpool, local r1, arg r2
5108 1016
510A 7018 A3C8    lock r1,nf                    ;lock shared pool
510E 9AFE
5110 4A91         add r1,=1                     ;add buffer to shared pool.
5112 48F0         lda r7,=0
5114 307A 06A0    sta r7,where-chain(r2)
5118 3018 A3C8    unlock r1,nf                  ;release shared pool.
511C 48F1         lda r7,=1
511E 3072         sta r7,(r2)                   ;last buffer on list.

5120 300A 09F0    sta r0,flushd-chain(r2)       ;mark buffer as recently flushed.

5124 7018 A3C6    lock r1,freend
5128 9AFE
512A 8906         if odd
512C 7078 A3C4      lock free
5130 9AFE
5132 4818 A3C4      lda r1,=free
                  endif
5136 3021         sta r2,(r1)
5138 3028 A3C6    unlock r2,freend
513C 6016       endroutine flpool
513E 6006
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 393
FAKREL.PLR;1    PAGE 14          IMP Fakes Reliability


                ;*******dedtrn********

                ; a routine to periodically search transaction blocks
                ; for messages, message fragments, etc. from a dead stream.
                ; fixed to clean up TRNBLKs of dead message streams properly.
FC00 0200               page DDTCode
54C4 4838 8910  dedtrn: lda r3,=trnblk
54C8 781B 000D  deds0:  ldab r1,trstat(r3)
54CC 9A78                bz deds2       ;empty transaction block
54CE 4E98               cmp r1,=ttledr  ;on leader queue?
54D0 9157                be deds1       ;sure nuff
54D2 4F18 0080          tst r1,=ttresv  ;host reserve it?
54D6 8A7D                bnz deds12     ;yes, make sure it's ok
54D8 785B 0005          ldab r5,trhstl+1(r3)    ;mes blk
54DC 4E58 0038          cmp r5,=tmnum
54E0 8257                bng deds9      ;number too big
54E2 A2D4               sll r5,4        ;its index
54E4 707D A930          lda r7,tmlock(r5)
54E8 9AFE                bz .-4
54EA 7E1B 000D          cmpb r1,trstat(r3)
54EE 8145                bne deds16             ;not really ours anymore
54F0 787B 0005          ldab r7,trhstl+1(r3)
54F4 A2F4               sll r7,4
54F6 4E75               cmp r7,r5
54F8 8140                bne deds16
54FA 707D 6938          set r7 = tmmess(r5)     ;put impnumber check at end
54FE 757B 000A          eor r7,trluse(r3)
5502 4F78 00F0          tst r7,=luse
5506 8A6F                bnz deds7
5508 707B 0004          set r7 = trhstl(r3)
550C 4C78 00FF          ior r7,= HFF
5510 717D 6938          sub r7,tmmess(r5)
5514 8B68                bnm deds7
5516 4A78 0800          add r7,= H800
551A 9B65                bm deds7
```

```
551C 707D 6932          set r7 = tmimp(r5)      ;check dest imp
5520 8B2C               if minus                ;block idle, maybe need to clean up
5522 4868 02F8            set sp = #lstack      ;trnfls needs this
5526 4078 18C6            call trnfls           ;get rid of buffer
552A 4F91                 if r1 .nbit. =ttrfnm  ;;trstat ;this doesn't need reply
552C 8A06
552E 707B 000C              lda r7,trntim(r3)   ;;trstat ;free block w/o sending reply
5532 357B 000C              eorm r7,trntim(r3)  ;clear it out
5536 9021                 else                  ;make and send reply
5538 707B 000A              lda r7,trdeds(r3)   ;clear deadstatus
553C 357B 000A              eorm r7,trdeds(r3)
5540 48F7                   setb trtypl+1(r3) = =cdestd ;set up dest dead reply
5542 387B 0003
5546 48FF                   set r7 = =subtyp    ;put in "dest imp dead" subtyp
5548 347B 0008              iorm r7,trmidl(r3)  ;turn on all bits first
554C 48FF                   set r7 = =subtyp-cimpd      ;then change .
554E 317B 0008              subm r7,trmidl(r3)  ;..just subtyp
5552 701D 6936              set r1 = tmctl(r5)  ;fix mess handling stuff
5556 4B18 7000              and r1,#getmax
555A A694                   srl r1,4
555C 707D 6936              lda r7,tmctl(r5)    ;check priority bit
5560 8B03                   if minus    ;;priled
5562 4C18 8000                ior r1,#priled
                            endif
5566 7C1D 6934              iorb r1,tmhost(r5)  ;include foreign host
556A 784D 693F              setb r4 = tmlhn(r5) ;get local host
556E 704C 6326              set r4 = h2pblk(r4)
5572 9B03                   ifnot minus         ;good hosts only
5574 4078 188E                call ledp2        ;KILLS STACK
                            endif
                          endif
                        endif
5578 300D A930  deds16:  sta r0,tmlock(r5)
557C 9020               br deds2
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 395
FAKREL.PLR;1    PAGE 16          IMP Fakes Reliability


557E 704B 000E  deds1:  lda r4,trhost(r3)        ;host it's queued on
5582 7068 6322          lda r6,hosts             ;set up for search
5586 564E 6326  deds5:  cmp r4,h2pblk(-r6)       ;he there?
558A 910D                be deds6        ;sure, no worries
558C 88FD               bnlp deds5       ;check the whole table
558E 48C1       deds9:  lda r4,=1
5590 394B 000C          subbm r4,trntim(r3)
5594 8A14                bnz deds2       ;still young
5596 E681               Trap 3201,<;Flushing an old trnblk - page 393>
5598 48F0               lda r7,=0        ;get rid of trnblk
559A 307B 000E          sta r7,trhost(r3)
559E 307B 000C          sta r7,trntim(r3)       ;;trstat        ;free the block
55A2 900D               br deds2         ;back into the loop
55A4 707C 405C  deds6:  lda r7,lockih(r4)
55A8 9AFE                bz .-4
55AA 764B 000E          cmp r4,trhost(r3)
55AE 810E                bne deds10
55B0 701C 005E          lda r1,nxtled(r4)       ;any leaders for host?
55B4 8A0B                bnz deds10      ;yes, it's okay
55B6 E683               Trap 3203,<;requeueing trnblk for IH - page 393>
55B8 4078 18AA          jsb r7,ledpf     ;poke host for this leader
55BC 4A38 0010  deds2:  add r3,=trnl
55C0 4E38 8F50          cmp r3,=trnblk+trntot
55C4 8182                bne deds0       ;do all transaction blocks
55C6 4008 1BEE          jmp tossx

55CA 300C 405C  deds10: sta r0,lockih(r4)
55CE 90F7               br deds2

55D0 7068 6322  deds12: lda r6,hosts             ;find owner
55D4 504E 6326  deds13: lda r4,h2pblk(-r6)
55D8 9B04                bm deds15
55DA 763C 002E          cmp r3,hitran(r4)
55DE 91EF                be deds2        ;found him
55E0 88FA       deds15: bnlp deds13
55E2 90D6               br deds9         ;let him time out

55E4 E684       deds7:  Trap 3204,<;trnblk/tmblk mismatch - page 393>
55E6 4868 02F8          lda sp,=lstack  ;needed by trnfls
55EA 4078 18C6          call trnfls
55EE 48F0               set trntim(r3) = #0      ;;trstat
55F0 307B 000C
55F4 90C2               br deds16

FC00 0200          dstolist dedtrn
4EE0 54C4
4EE2 1CBC
4EE4 0000
FC00 0600               page FakCode
```

```
                ;send all hosts an imp-going-down if igdown set

5140 7018 5F8C  impdwn: lda r1,igdown   ;send mess?
5144 7218 5F8E          add r1,igdflag
5148 9A33               bz impdx        ;nope
514A 7058 6322          lda r5,hosts
514E 49D2               sub r5,=words   ;last possible host
5150 704D 6326  impdlp: lda r4,h2pblk(r5)
5154 9B26                bm impdl4      ;skip if no host
5156 783C 001F          ldab r3,hihd(r4)        ;host alive?
515A 8A23                bnz impdl4     ;;hostup ;nope, skip him
515C 7078 A4A8          lock trnlok
5160 9AFE
5162 4838 8F40          lda r3,=trnblk+trntot-trnl      ;last trn blk
5166 787B 000D  impdl2: ldab r7,trstat(r3)      ;tran blk state
516A 9A0B                bz impdl3      ;got a free one
516C 4938 0010          sub r3,=trnl    ;try another
5170 4E38 8910          cmp r3,=trnblk
5174 82F9                bng impdl2     ;look more
5176 3008 A4A8          sta r0,trnlok   ;unlock trn blks
517A 4078 1C30          jsb r7,otsleep  ;try again later
517E 90E9               br impdlp

5180 4818 0F00  impdl3: lda r1,=ihcode  ;for new-format
5184 2013               sta r1,(r3)+    ;into tran blk: trnetl
5186 4892               lda r1,=cimpdn  ;leader type
5188 2013               sta r1,(r3)+    ;trtypl
518A 4890               lda r1,=0
518C 2013               sta r1,(r3)+    ;trhstl
518E 2013               sta r1,(r3)+    ;trdstl
5190 7018 5F8C          lda r1,igdown   ;imp midl
5194 3013               sta r1,(r3)     ;trmidl
5196 49B8               sub r3,=trmidl  ;back to head of trn blk
5198 4078 188A          jsb r7,ledp0    ;put it in host
519C 3008 A4A8          sta r0,trnlok
51A0 49D2       impdl4: sub r5,=words   ;more hosts?
51A2 8BD7                bnm impdlp     ;yup, keep on
51A4 48F0               lda r7,=0       ;done, turn us off
51A6 3078 5F8C          sta r7,igdown   ;by zeroing igdown
51AA 3078 5F8E          sta r7,igdflag
51AE 4008 1BEE  impdx:  jmp tossx       ;done this routine
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 397
FAKREL.PLR;1    PAGE 18          IMP Fakes Reliability

                ;       timeout verification that ring is still good.
                ;       traps with r1 = old ringf, r2 = old ringc
FC00 0200               page ddtcode
55F6 1076       routine rngchk, uses r1-r2
55F8 7018 64B6    lda r1,ringf          ;ring pointer.
55FC 7028 64B4    lda r2,ringc          ;ring counter.
5600 9B0A         if minus } r2 > =ringln } r1 < =ring } r1 >= = ringe
5602 4E28 0018
5606 9C07
5608 4E18 64B8
560C 9204
560E 4E18 64D0
5612 9209
5614 4878 64B8      lda r7,=ring        ;bad, so re-initialize the ring structure.
5618 3078 64B6      sta r7,ringf        ;pointer.
561C 48F0           lda r7,=0           ;counter initial value.
561E 3078 64B4      sta r7,ringc
5622 E2C9           Trap 1311,<;ring structure broken in timeout - page 395>
                  endif
5624 6006       endroutine

FC00 0200          dstolist rngchk
4EE2 55F6
4EE4 1CBC
4EE6 0000
FC00 0600          page fakcode
```

```
                ;for all real hosts - set hihd=1 if ready line down, hihd=0
                ;if ready line up and hihd is  D1
                ;also count myimp down

51B2 1076       routine dedh
51B4 7078 62E0    lda r7,myimp
51B8 9A04         ifnot zero                    ;we're not up to net.
51BA 49F1           sub r7,=1
51BC 3078 62E0      sta r7,myimp
                  endif

51C0 7078 6322    lda r7,hosts          ;all hosts
51C4 49F8         sub r7,=nfh2          ;exclude fakes
                  repeat
51C6 504F 632E      lda r4,h2pblk+nfh2(-r7)
51CA 9B1C           ifnot minus                 ;no host here.
51CC 48A1             lda r2,=hrdown
51CE 781C 001F        ldab r1,hihd(r4)          ;current state
51D2 4E94             if r1 <> =hninit          ;not initializing
51D4 9117
51D6 705C 0020          lda r5,iobloc(r4)
51DA 703D 0006          lda r3,statih(r5)
51DE 4F38 1000          if r3 .bit. =hready     ;ready line up.
51E2 9A07
51E4 4D21                 eor r2,r1
51E6 9A03                 ifnot zero            ;host already up(hihd was 0)
51E8 88EF                   next ifnot loop     ;do next host
51EA 6006                   return              ;no more, quit
                          endif
51EC 302C 002C            sta r2,deadsc(r4)     ;clear host down status
                        endif
51F0 4E21               if r2 <> r1             ;changed
51F2 9108
51F4 3008 6512            sta r0,ophgo
51F8 382C 001F            stab r2,hihd(r4)      ;;hostup
51FC 9A03                 ifnot zero            ;host down.
51FE 302C 0052              sta r2,ihtt(r4)     ;ready flap > reset him
                          endif
                        endif
                      endif
                    endif
5202 88E2           until loop
                  endrepeat
5204 90F3       endroutine dedh
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 399
FAKREL.PLR;1    PAGE 20          IMP Fakes Reliability

                ;routine to make up lights word like 316's have

5206 1076       routine lites
5208 4818 0020    lda r1,= D16*words            ;for 16 hosts only
520C 4838 FFFF    lda r3,=-1
                  repeat
     0001       .if nz PSE                     ;PSE feature: fancy DISPLAY
5210 4858 B00F       lda r5,= HB00F             ;non-existant Host
                .iftf ;nz PSE
5214 5049 632E       lda r4,h2pblk+nfh2(-r1)    ;host blk?
5218 9B08            ifnot minus          ;there is one
521A 787C 001F         ldab r7,hihd(r4)
521E 8A02              if zero            ;up
5220 4DB1                eor r3,=1        ;turn this host on
                       endif
                .ift ;nz PSE
5222 735C 0020         and r5,iobloc(r4) ;get A000 or B000
5226 4C57              ior r5,r7         ;and host state
                .iftf ;nz PSE
                    endif
                .ift ;nz PSE
5228 4871           lda r7,r1
522A A2F7           sll r7,7             ;Host number in left half
522C 4C57           ior r5,r7
522E 3059 6300      sta r5,hdisp(r1)     ;put away for Display
5232 A7B1           rrl r3,1             ;PSE lights left-to-right
                .iftf ;nz PSE
5234 88EE           until loop
                .iff    ;nz PSE
                    rll r3,1             ;ARPA host lights right-to-left
                .iftf   ;nf PSE
                  endrepeat
5236 3038 62EE    sta r3,watchh ;save up host lites
```

```
                .iff    ;nz PSE
                  lda r3,=0             ;initial mask for modems
                  lda r1,modems         ;do modems we have
                .ift    ;nz PSE
523A 4838 FFFF    lda r3,=-1            ;initial mask for modems
523E 4818 0010    lda r1,#nmd*words     ;do all m2pblk
                .iftf   ;nz PSE
                  repeat
                .ift ;nz PSE
5242 4858 D00F      lda r5,= HD00F      ;non-existant modem
                .iftf ;nz PSE
5246 5049 635E      lda r4,m2pblk(-r1)  ;now for a line
524A 9B0C           ifnot minus                 ;none here
                .ift ;nz PSE
524C 735C 0020        and r5,iobloc(r4)         ;get C000/D000
5250 7C5C 001E        iorb r5,mloop(r4)         ;loop bits (40 or 80)
                .iftf ;nz PSE
5254 787C 0002        if byte phflag(r4) .bit. =phup
5258 4FF4
525A 9A03
525C 4DB1               eor r3,=1             ;line up
                .ift ;nz PSE
525E 9002              else                   ;down
5260 4CD1               ior r5,=1
                .iftf ;nz PSE
                     endif
                   endif
5262 A7B1          rrl r3,1                   ;left-to-right lites
                .ift ;nz PSE
5264 4871          lda r7,r1                  ;modems 0-7
5266 A2F7          sll r7,7                   ;number in left
5268 4C57          ior r5,r7
526A 3059 62F0     sta r5,mdisp(r1)           ;store for Display
                .iftf ;nz PSE
526E 88EA          until loop
                  endrepeat
                .iff    ;nz PSE
                  eor r3,watchh         ;add in the host lights
                  sta r3,watch          ;save the word now
                .ift    ;nz PSE
5270 A6B8         srl r3, D8            ;get just 8 modems
5272 3838 62D8    stab r3,watchs        ; into upper lites
                .endc   ;nz PSE
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 401
FAKREL.PLR;1    PAGE 22          IMP Fakes Reliability

                ;check lite pointers

5276 7038 62D2    lda r3,watm1
527A 3038 FC04    sta r3,%map2
527E 70B8 62D0    lda r3,@watch1        ;see this location?
5282 8011          qutpat badlit
5284 7038 62D6    lda r3,watm2
5288 3038 FC04    sta r3,%map2
528C 70B8 62D4    lda r3,@watch2
5290 800A          qutpat badlit        ;no see => fix pointers
                idledo:
5292 7038 00D2    lda r3,mapv2
5296 3038 FC04    sta r3,%map2

                ;******* count the number of times through the idle loop
                ;         every half second

529A 7028 A2CE    lda r2,idlec+foo
529E 3028 6522    sta r2,idles

52A2 6006       endroutine lites

52A4 4818 62D8  badlit: lda r1,=watchs  ;reset light defaults
52A8 3018 62D0          sta r1,watch1
52AC 4818 62EC          lda r1,=watch
52B0 3018 62D4          sta r1,watch2
52B4 90EF               br idledo
```

```
                ;now check to send trouble report immediately

52B6 5EBA       trbtab: dgsb    ;diagnostic queue leader ;see bum below
52B8 5EA6               trpsb   ;traps
52BA 5E7E               stsb    ;status
52BC 5E92               thrusb  ;throughput
     0008       trbtl=.-trbtab

52BE 1076       routine trbtim
52C0 7078 5E76     lda r7,mgsb+snon             ;get mess gen switch
52C4 8B04          if minus                     ;must count up
52C6 48F1             lda r7,#1
52C8 3278 5E76        addm r7,mgsb+snon         ;so count it up
                   endif
52CC 7858 40C6     ldab r5,nccimp       ;where's ncc
52D0 A2D1          sll r5,1
52D2 4890          lda r1,=0
52D4 702D 72FC     lda r2,spfrut(r5)    ;route to NCC?
52D8 9B02          ifnot minus  ;;spfded
52DA 4891             lda r1,=1
                   endif
52DC A6D1          srl r5,1     ;;nccimp ;restore nccimp
52DE 7848 40C7     ldab r4,ncchst
52E2 7028 40C8     lda r2,ncclnk
52E6 48F8          lda r7,=trbtl
                   repeat
52E8 503F 52B6        lda r3,trbtab(-r7)        ;get leader buffer area
52EC 304B 0004        sta r4,hstl(r3)   ;destination host
52F0 305B 0006        sta r5,dstl(r3)   ;destination imp
52F4 302B 0008        sta r2,midl(r3)   ;mess id (link)
52F8 9804              until loop
52FA 301B 000C        sta r1,snon(r3)   ;bums by not executiing on last of loop
52FE 90F5          endrepeat
5300 4B28 FFF0     and r2,=messid       ;regular pkt
5304 3028 5E9A     sta r2,thrusb+midl   ;thrupt may be multi pkt
5308 3028 5EC2     sta r2,dgsb+midl     ;diag always multi
530C 3058 5EE8     sta r5,tlsb+dstl     ;Set Plulog destination to NCC
5310 3048 5EE6     sta r4,tlsb+hstl     ; (other inits in fakini)
5314 4811          lda r1,r1            ;did we turn things on?
5316 9A1F          ifnot zero
5318 7018 5F82        lda r1,throtc
531C 4E18 0032        cmp r1,= D50
5320 9104             ifnot equal
5322 4A91                add r1,=1
5324 3018 5F82           sta r1,throtc
                      endif
5328 9B16             ifnot minus       ;;throtc r1     ;ok to send
532A 48A5                lda r2,=5      ;throttle constant
532C 7018 A512           lda r1,ophgo+foo
5330 7518 62AE           eor r1,anom
5334 7518 5F84           eor r1,banom
5338 9A06                ifnot zero             ;something to send, trigger status
533A 3128 5F82              subm r2,throtc
533E 4890                   lda r1,=0
5340 3018 5E8C              sta r1,stsb+statf   ;temporary set of freq to 0
                         endif
```

```
5344 7078 6166            if nilops+cilloc       ;any reg traps?
5348 9A06
534A 3128 5F82              subm r2,throtc
534E 4890                   lda r1,=0
5350 3018 5EB4              sta r1,trpsb+statf
                          endif
                        endif
                      endif
5354 6006           endroutine trbtim
```

```
                ; time the message block stuff

5356 4894       bt0:    lda r1,=4       ;fast rate = 640 ms
5358 7878 650B          ldab r7,mintim+1
535C 49F1               sub r7,=1
535E 9902               bo .+4
5360 4C92                ior r1,=2      ;middle rate - 1.28 sec
5362 8B04               bnm bt1
5364 4C91                ior r1,=1      ;slow rate - 58.6 sec
5366 4878 005A           lda r7,= D90
536A 3078 650A  bt1:    sta r7,mintim
536E 4858 0370          lda r5,=<tmnum-1>*tmlen
5372 482D A930  bt2:    lda r2,=tmlock(r5)
5376 4078 5396          jsb r7,agetm
537A 4958 0010          sub r5,=tmlen
537E 8BFA               bnm bt2
5380 4858 0370          lda r5,=<rmnum-1>*rmlen
5384 482D A5B0  bt3:    lda r2,=rmlock(r5)
5388 4078 53C0          jsb r7,ageing
538C 4958 0010          sub r5,=rmlen
5390 8BFA               bnm bt3
5392 4008 1BEE          jmp tossx
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 404
FAKREL.PLR;1    PAGE 25          IMP Fakes Reliability

                ;subroutines to age message blocks

5396 703A 0000  agetm:  lock r3,0(r2)   ;;tmlock
539A 9AFE
539C 703A C00A          lda r3,tstate-tmlock(r2)        ;control bits
53A0 4D38 C000          eor r3,=tminit+tmrset   ;check if these on
53A4 4B38 C000          and r3,=tminit+tmrset
53A8 8A0F                bnz agetx      ;nope, not both
53AA 703A C008          lda r3,tmmess-tmlock(r2)        ;now check age
53AE 4FBC               tst r3,= HC*age0        ;over 3?
53B0 9A0B                bz agetx       ;nope, no touch
53B2 4838 8000  bt04:   lda r3,=sign    ;bit to say idle
53B6 343A C002          iorm r3,tmimp-tmlock(r2)        ;into imp number
53BA 703A 000A          lda r3,tstate-tmlock+foo(r2)    ;smash tstate too
53BE 9027               br bt6  ;done this block

53C0 703A 0000  ageing: lock r3,0(r2)   ;;rmlock
53C4 9AFE
53C6 704A C002  agetx:  lda r4,tmimp-tmlock(r2) ;;rmimp
53CA 9B21                bm bt6
53CC 4E48 007F          cmp r4,=nimp
53D0 9CF1                bl bt04        ;invalid IMP #
53D2 A2C1               sll r4,1
53D4 704C 72FC          lda r4,spfrut(r4)       ;route to this IMP
53D8 8B06               if minus        ;;spfded ;IMP died - reset block
53DA 4838 FFF0            set r3 = =-1?age      ;allow time for cleanup
53DE 333A C008            andm r3,tmmess-tmlock(r2)
53E2 90E8                 tr bt04
                        endif
53E4 784A C00F          ldab r4,tmlhn-tmlock(r2)        ;;rmlhn ;local host
53E8 7648 6322          cmp r4,hosts
53EC 9203               bg bt01
53EE E658                Trap 3130,<;Bad local Host in message block - page 402>
53F0 90E1                br bt04
53F2 704C 6326  bt01:   lda r4,h2pblk(r4)       ;get its param blk
53F6 9B04                bm bt02        ;hardware gone
53F8 784C 001F          ldab r4,hihd(r4)        ;host state?
53FC 9A08                bz bt6         ;;hostup ;he's okay
53FE 703A C008  bt02:   lda r3,tmmess-tmlock(r2)        ;;rmmess ;set age = max
5402 4FBC               tst r3,=age0* HC        ;if not too young
5404 9A04                bz bt6
5406 4CBF               ior r3,=age
5408 303A C008          sta r3,tmmess-tmlock(r2)        ;;rmmess
```

```
540C 703A C008  bt6:    lda r3,tmmess-tmlock(r2)
540O 4BBF               and r3,=age
5412 7F1B 5430          tstb r1,tictim(r3)
5416 9A04               bz bt4
5418 48B1                lda r3,=1
541A 323A C008           addm r3,tmmess-tmlock(r2)
541E 3002       bt4:    unlock (r2)
5420 4F58 007E          tst r5,= H8*tmlen-words
5424 8A05               if zero
5426 1076                 save r7
5428 4078 1C34            call tsleep
542C 6076                 restore r7
                        endif
542E 4007               jmp (r7)

5430   04       tictim: .byte 4,4,4,4,2,2,2,2
5431   04
5432   04
5433   04
5434   02
5435   02
5436   02
5437   02
5438   02               .byte 2,2,2,2,1,1,1,0
5439   02
543A   02
543B   02
543C   01
543D   01
543E   01
543F   00
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 406
FAKREL.PLR;1    PAGE 27          IMP Fakes Reliability

                ;time out tmess for incompletes
                ;3 tm mes blks every slow tick

5440 1076       routine messto
5442 7058 A4AA    lda r5,messt  ;clears in case of trouble}
5446 4078 5458    call messts
544A 4078 5458    call messts
544E 4078 5458    call messts
5452 3058 A4AA    sta r5,messt
5456 6006       endroutine

5458 1076       routine messts
545A 707D 6932    lda r7,tmimp(r5)              ;message block.
545E 9B12         ifnot minus                   ;non-idle block.
5460 707D A930      lock tmlock(r5)             ;lock for duration of test.
5464 9AFE
5466 787D 693B      ldab r7,tstate+1(r5)
546A 4D78 00FF      eor r7,= HFF
546E 9A08           ifnot zero                  ;outstanding messages.
5470 787D 693A        ldab r7,tstate(r5)
5474 4FF3             tst r7,=msto_- H8
5476 9A04             ifnot zero                ;mess blk not yet timed out.
5478 49F1               sub r7,=msto0_- H8
547A 387D 693A          stab r7,tstate(r5)
                      endif
                    endif
547E 300D A930      unlock tmlock(r5)
                  endif
5482 4958 0010    sub r5,=tmlen
5486 8B03         if minus
5488 4858 0370      lda r5,=<tmnum-1>*tmlen
                  endif
548C 6006       endroutine
```

```
                ;***  test the ih software structures ***

                ;table of valid ih dispatches

548E 1076       routine rihs
5490 4818 0038          lda r1,=th*words        ;how many hosts
                        repeat
5494 5049 6326            lda r4,h2pblk(-r1)
5498 9B14                 ifnot minus           ;some host here.
549A 707C 404E              lock ihloc(r4)
549E 9AFE
.comnt | This must go in warm or ih must go in local
        lda r6,ihlo(r4) ;ih dispatch
        lda r2,=ihdb    ;probable called routine
        jsb r7,return
         be rihs10      ;sure 'nuff
        lda r2,=ihdba   ;possible alternate
        jsb r7,return
         be rihs10      ;ah, yes
        Trap 3030,<;ihlo is a mess - page 405>
        br rihs11       ;reinit his ih
rihs10:
|
54A0 702C 0056              lda r2,ihwq(r4)
54A4 4E2C 0060              if r2 <> =shq(r4) & r2 <> =shpq(r4)
54A8 910A
54AA 4E2C 0064
54AE 9107
54B0 E619                     Trap 3031,<;ihwq is a mess - page 405>
                              ;;rihs11:
54B2 705C 0020                lda r5,iobloc(r4)
54B6 4078 59FC                call ihsini       ;reinitialize IH side.
54BA 9003                   else
54BC 300C 404E                unlock ihloc(r4)
                            endif
.comnt |                ;must be on warm page, or HI in local.
        lda r7,lockhi(r4)
         bz .-4
        lda r6,htemp7(r4)
        jsb r7,retx
        lda r6,hilo(r4)
        jsb r7,retx     ;fix this next time
        sta r0,lockhi(r4)
|
                          endif
54C0 4E90                 until r1 = =0
54C2 81E9
                        endrepeat
54C4 6006       endroutine rihs
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 408
FAKREL.PLR;1    PAGE 29          IMP Fakes Reliability

                ;***** check the fake host stuff *****

FC00 0600                Page FakCode

54C6 1076       routine rfake
54C8 4898         lda r1,=nfh2
                  repeat
54CA 5059 636E      lda r5,v2pblk(-r1)
54CE 9903           ifnot odd
54D0 4078 4DC2        call bldfh
                    endif
54D4 4078 1C34      call tsleep
54D8 4E90           until r1 = =0
54DA 81F8
                  endrepeat
54DC 6006       endroutine rfake
.comnt |        lda r3,=nfh2
rfake1: lda r1,v2pblk(-r3)      ;this fake okay
         bo rfake7      ;not configured yet
        jsb r7,bldfh
        lda r2,lockfd(r1)       ;lock just in case
         bz .-4
        lda r6,dozet(r1)
        cmp r6,=fthgo
        be rfake3       ;just initialized
        lda r2,=doze
        jsb r7,retchk
        be rfake3       ;okay dispatch
        lda r2,=jam     ;maybe jam was called
        jsb r7,retchk
         be rfake3
        lda r2,=jamend
        jsb r7,retchk
         be rfake3
        lda r2,=jamhol  ;another call
        jsb r7,retchk
         bne rfake5     ;bad dispatch
rfake3: unlock lockfd(r1)
        lock r6,lockfw(r1)
        lda r6,waitt(r1)
        lda r2,=wait
        jsb r7,retchk
        be rfake4       ;okay dispatch
        lda r2,=suck    ;maybe this caller
        jsb r7,retchk
         be rfake4      ;good caller
        sub r6,=2       ;suck sometimes bumps return)
        jsb r7,retchk
         bne rfake5     ;bad dispatch
```

```
rfake4: cmp r6,=sucwt+4 ;wait from a suck?
        bne rfake6      ;no, so ignore waitt3
        lda r6,waitt3(r1)
        lda r2,=suck    ;called routine
        jsb r7,retchk
        be rfake6       ;okay dispatch
rfake5: Trap 3052,<;scrambled fake io block - page 407>
        lda r7,=-1
        sta r7,v2pblk(r3)       ;remove from v2pblk
rfake6: sta r0,lockfw(r1)       ;unlock now
rfake7: sta r3,rfakt3   ;save to sleep
        jsb r7,tsleep
        lda r3,rfakt3   ;now, where were we
        bnz rfake1      ;do all the fake hosts
|;continues from above

.comnt | *** also allow rest here ***
        lda r6,fhtr1+14
        lda r2,=fhtp
        jsb r7,retchk
        lda r6,ftha7
        lda r2,=t2fg
        jsb r7,retchk
        lda r6,fhda7
        lda r2,=f2dp
        jsb r7,retchk
|
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 410
FAKREL.PLR;1    PAGE 31          IMP Fakes Reliability

                ;*** rtc pid dispatches

54DE 1076       routine conclk
54E0 7078 00B0    lda r7,maprel
54E4 3078 FC04    sta r7,%map2                  ;get to stage vars
54E8 7048 9D5A    lda r4,m2#usebus              ;what buses and RTC's
54EC 7078 00D2    lda r7,mapv2
54F0 3078 FC04    sta r7,%map2
54F4 48D0         lda r5,=0                     ;no PTIP so far
54F6 3058 5E10    STA R5,temwrd
54FA 4894         lda r1,=bsadil
                  repeat
54FC 5749 0EC6      if r4 .bit. bittab+ H10(-r1)
5500 9A25
5502 7029 0320        lda r2,bsadrs(r1)         ;what bus then
5506 703A 0008        lda r3,rtcpds(r2)         ;get pid this clock is hung on
550A 4873             lda r7,r3
550C 4B78 00FF        and r7,= HFF              ;first the fast clock
5510 9A0D             ifnot zero
5512 4ED0               if r5 = =0              ;no Quick timeout guy yet.
5514 8104
5516 7058 40CC            lda r5,qtoadr         ;Quick timeout routine
551A 9006               else
551C 765F 80C0            if r5 <> base(r7)     ;no clock there yet.
5520 9103
                                                ;problems if > 2 clocks
5522 4858 113C                lda r5,=loop3             ;null dispatch
                            endif
                          endif
5526 305F 80C0            sta r5,base(r7)       ;fix up base dispatch
                        endif
552A A6B8               srl r3, D8
552C 4878 4262          lda r7,=tohot
5530 307B 80C0          sta r7,base(r3)
5534 707A 000A          lda r7,rtcsws(r2)
5538 3078 62E2          sta r7,mine             ;now, got an IMP number?
553C 9A04               if zero ) r7 > =nimp    ;IMP number invalid
553E 4E78 007F
5542 8C04
                          Trap 413,<:IMP num invld (CK RTC SWTCHS) (H) - page 408>
5544 E10B
5546 4048 1308            jsb r4,reload   ;a drastic solution to the problem
                        endif
                      endif

554A 1046               SAVE R4-R5              ;CREATE COME TEMPS
554C 1056
554E 483A 000F          LDA R3,#RTCTEM+3(R2)    ;TEMP BYTE ADDRESS
5552 4851               LDA R5,R1               ;TEMP REF TABLE INNDEX
5554 A2D1               SLL R5,1                ;SHIFTLEFT 1 BIT
5556 4AD3               ADD R5,=3               ;TEMP REF TABLE INNDEX
                        REPEAT                  ;FOR ALL SENSORS, THIS RTC
5558 787D 40D8            LDAB R7,TEMREF(R5)    ;GET REFERENCE VALUE
555C 7043                 LDA R4,(R3)           ;READ 2 TEMPERATURES
555E 4833                 LDA R3,R3
5560 8803                 IF EVEN       ;IF CHECKING LEFT HALF
```

```
                        ENDIF
5564 4B48 00FF          AND R4,# HFF            ;JUST 8 BITS
5568 4E28 E000          IF R2 = = HE000 } R2 = = HF000
556C 9104
556E 4E28 F000
5572 8109
5574 4E47               IF R4 > R7              ;TEMPERATURE TOO HIGH
5576 8C07
5578 E10D                 Trap 415,<;Node ovrhting--R2=BUS  R4=TEMP (H) - page 408>
557A 1076                 SAVE R7
557C 48F1                 LDA R7,=1
557E 3278 5E10            ADDM R7,temwrd
5582 6076                 RESTORE R7
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 411
FAKREL.PLR;1    PAGE 31.1       IMP Fakes Reliability

                          ENDIF
                          ENDIF
5584 49D1                 SUB R5,=1             ;NEXT REFERENCE VALUE
5586 49B1                 SUB R3,=1             ;NEXT TEMP READING
5588 4E3A 000C            UNTIL R3 < #RTCTEM(R2)  ;DONE THIS RTC
558C 82E6
                        ENDREPEAT
558E 6056               RESTORE R4-R5           ;GET BACK SAVED REGS.
5590 6046
5592 4E90             until r1 = =0
5594 81B4
                    endrepeat

5596 48B0                 LDA R3,=0
5598 7638 5E10            CMP R3,temwrd
559C 8206                 BLE CONC1
559E 4838 0020               LDA R3,=HTSBIT
55A2 3438 62AE               IORM R3,ANOM
55A6 9005                    BR CONC2                          ;THE IMP IS JUST FINE
55A8 4838 FFDF  CONC1:       LDA R3,=<-1>?HTSBIT
55AC 3338 62AE               ANDM R3,ANOM
55B0 8000       CONC2:       NOP

55B2 6006       endroutine conclk
```

```
                ;buffer variables initialization
                ;on mapv2 page are point, chain, chan, where, and flushd tables
                ; fill the remaining space with buffers,
                ; starting from the top of each "hole"


55B4 1076       routine bufini, uses r1-r4
55B6 4890         lda r1,=0                     ;how many buffers so far

55B8 4848 FFF4    lda r4,=bufmap-bfmpnd         ;for all extant buf pages.
                  repeat                        ;setup page.
55BC 602C 5646      lda r2,bfmpnd(r4)+  ;get ptr to full buf page
55C0 9905           ifnot odd           ;page enabled
55C2 7022             lda r2,(r2)       ;get buf page pointer.
55C4 9903             ifnot odd         ;page exists
55C6 4078 5622          call dobufs     ;setup bufs
                      endif
                    endif
55CA 4844           lda r4,r4
55CC 8AF8           until zero          ;no more pages in table.
                  endrepeat

55CE 4838 12C0    lda r3,=0#<point+mnbuf2>      ;last word in point
                  repeat                ;put buffers midst tables.
55D2 4938 00A0      sub r3,=buflen      ;room for a buffer?
55D6 4E39 0F7A      until r3 <= =0#<point+<5*words>>(r1)        ;no room left.
55DA 8C18
55DC 4823           lda r2,r3           ;yes - build its address in r2
55DE A6A4           srl r2,4            ;packed address.
55E0 7428 00D2      ior r2,mapv2
55E4 2029 8F70      sta r2,point(r1)+
55E8 4A28 0035      add r2,=mnbuf2/ H10 ;offset to next table
55EC 2029 8F70      sta r2,point(r1)+
55F0 4A28 0035      add r2,=mnbuf2/ H10 ;..and to next
55F4 2029 8F70      sta r2,point(r1)+
55F8 4A28 0035      add r2,=mnbuf2/ H10 ;...
55FC 2029 8F70      sta r2,point(r1)+
5600 4A28 0035      add r2,=mnbuf2/ H10 ;.. and to the last
5604 2029 8F70      sta r2,point(r1)+
5608 90E5         endrepeat

560A 4992         sub r1,=2             ;reserve last buffer for junk
560C 3018 63CC    sta r1,maxbuf         ;length of table
5610 4829 92C0    lda r2,=chain(r1)     ;address of last buffer.
5614 3028 63CA    sta r2,junk           ;used as junk buffer.
5618 A691         srl r1,1              ;and buffer count
561A 4A91         add r1,=1             ;..offset 'cause nf is a lock
561C 3018 63CE    sta r1,maxnf
5620 6006       endroutine bufini
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 413
FAKREL.PLR;1    PAGE 33          IMP Fakes Reliability

                ;set up an all-buffers page. start from very top.

5622 1076       routine dobufs, arg r2, uses r1, uses r3
5624 483A 01F6    lda r3,=< H2000-buflen>/ H10(r2)      ;make the point word
                  repeat                                ;put buffer in table.
5628 2039 8F70      sta r3,point(r1)+                   ;into the table
562C 49BA           sub r3,=buflen/ H10                 ;back up one
562E 4E3A 000C      until r3 < =<0#pagebc>/ H10(r2)     ;past beginning.
5632 82FB
                  endrepeat
5634 6006       endroutine dobufs
```

```
                ;***** initialization*****

                ;table of maps that cannot move

                ;mapv2 must be first for go1 loop to leave right map setup
                ;tables of maps to check and set up for buffers.
                ;set entry odd if not to be used (for bufmap)
5636 00D2       mapct:  mapv2.mapvar
5638 00D0
563A 00E0       bufmap: mapb1,mapb2,mapb3,mapb4,mapb5,mapb6
563C 00E2
563E 00E4
5640 00E6
5642 00E8
5644 00EA
     0010       lmapct=.-mapct
                bfmpnd:
     0000       .iif nz vmap1-lmapct, .error vmap/bufmap mismatch

                ;tables to drive IMP initialization

                ;first: initialize large contiguous blocks:
                ; format: <base,length,init>
                ;processed in order (unlike the rest)

                itabs:
5646 620A               sidflg,m2-sidflg,0      ;zero vars
5648 1DF6
564A 0000
564C 8910               trnblk,m3-trnblk,0      ;zero buf vars
564E 16F0
5650 0000
5652 80C0               base,lbase,bad          ;empty base table
5654 0100
5656 1064
5658 6326               h2pblk,husend-h2pblk,-1 ;;v2pblk m2pblk ;config tables
565A 0080
565C FFFF
565E 5E00               fakvst,m1-fakvst,0      ;finally, zero fake vars.
5660 0200
5662 0000
     001E       itabsl=.-itabs
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 415
FAKREL.PLR;1    PAGE 35          IMP Fakes Reliability

                ;now, individual words: <loc,value>

5664 62D0       itabc:  watch1,watchs   ;lites
5666 62D8
     0001       .lif z PSE
                        watch2,watch    ;combined display
     0001       .lif nz PSE
5668 62D4               watch2,watchh   ;16 host down below
566A 62EE
566C A4A6               cycle, H9001    ;timeout counters
566E 9001
5670 A506               stolok,2
5672 0002
5674 A4D4               tcgo,2          ;host timeouts
5676 0002
5678 A4D8               tbkgo,2
567A 0002
567C 64D6               tcgoa,2
567E 0002
5680 64DA               tbkgoa,2
5682 0002
5684 6322               hosts,nfh2
5686 0008
5688 62E0               myimp, D60      ;coming up counter
568A 003C
568C 5F92               nsstack-words,nicestop  ;nice-stop dispatch
568E 5A38
5690 5F94               nsspsv,nsstack-<2*words> ;nice-stop stack pointer.
5692 5F90
5694 7280               ccled+net1,hicode       ;fake hosts
5696 0F00
5698 7284               ccled+hst1,getpri+ HFD  ;tty dest = DDT fake
569A 80FD
569C 64B6               ringf,ring      ;restarter pointer
569E 64B8
56A0 5E92               thrusb+net1,hicode      ;more fakes
56A2 0F00
56A4 5E7E               stsb+net1,hicode
56A6 0F00
56A8 5EA6               trpsb+net1,hicode
56AA 0F00
56AC 5E6A               mgsb+net1,hicode
56AE 0F00
56B0 5EBA               dgsb+net1,hicode
56B2 0F00
56B4 5ECE               hwsb+net1,hicode
56B6 0F00
56B8 5EE2               tlsb+net1,hicode
56BA 0F00
```

```
56BC 5EA0               thrusb+statf, H800
56BE 0800
56C0 5E8C               stsb+statf, H800
56C2 0800
56C4 5EB4               trpsb+statf, H800
56C6 0800
56C8 5EC8               dgsb+statf, H1
56CA 0001
56CC 5EDC               hwsb+statf, H800
56CE 0800
56D0 5EF0               tlsb+statf,tlogfreq
56D2 4000
56D4 5E6E               mgsb+hstl, HFF  ;msg gen to discard
56D6 00FF
56D8 5ED2               hwsb+hstl, HFF
56DA 00FF
56DC 5EEA               tlsb+midl, 0233_ H8
56DE 9B00
56E0 5E78               mgsb+statf,1
56E2 0001
56E4 5E68               mgnl, H8
56E6 0008
56E8 5EF6               tlttcp
56EA   0A                .byte mynet,ncc_-6     ;dest net,,host
56EB   00
56EC 5EF8               tlttcp+2,ncc&desti     ;dest IMP
56EE 0001
56F0 5EFA               tlttcp+4,mynet_ H8+ HFF :source net and host
56F2 0AFF
56F4 5F00               tlttcp+ H0A, 015_4     ;dest format
56F6 00D0
56F8 0000               0,0             ;spares
56FA 0000
56FC 0000               0,0
56FE 0000
5700 0000               0,0
5702 0000
5704 0000               0,0
5706 0000
5708 0000               0,0
570A 0000
570C 0000               0,0
570E 0000
     00AC       itabcl=.-itabc
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 417
FAKREL.PLR;1    PAGE 37          IMP Fakes Reliability

                ;now table to copy some things  <loc,ptr(value)>

5710 7286       itabi:  ccled+dstl,mine ;fake hosts
5712 62E2
5714 5E70               mgsb+dstl,mine
5716 62E2
5718 5ED4               hwsb+dstl,mine  ;my own discard
571A 62E2
571C 5EFC               tlttcp+6,mine   ;TCP source
571E 62E2
5720 6524               dynxt,dyinit
5722 40D6
5724 0000               0,0             ;spare
5726 0000
5728 0000               0,0
572A 0000
572C 0000               0,0
572E 0000
     0020       itabil=.-itabi

                ;now table of locks to unlock (set to 1)

5730 B2A4       itabl:  dsplok  ;display
5732 A550               conlok  ;configuration lock
     0001       .if nz VHA
5734 B3FC               vhalok  ;vha parameters
                .endc
5736 A2EA               clklok  ;clock reliability lock
5738 A3C8               nf      ;free buffer count
573A A3C4               free    ;free list head
573C A3C6               freend  ; and tail
573E A3B6               ltq     ;task queue lock
5740 A4A4               lockro  ;routing out buffers lock
5742 B2C8               rutlok  ;routing process lock
5744 B2C6               spfrtl  ;lock on SPF common table
5746 A4A8               trnlok  ;free transaction block lock
5748 A4B2               ringlk  ;restarter lock
574A A25E               d2fl    ;DDT and TTY locks
574C A260               f2dl
574E A264               ddtlok
5750 A262               ttylok
5752 A2AC               ttyobf+rbflok
5754 A294               ttyibf+rbflok
5756 626A               dspflag
5758 5EDA               hwsb+snon
575A 5EEE               tlsb+snon
     002C       itabll=.-itabl

                ;now a table of queues to set up (word following points to
                ; head, head is set to 1)

575C 63B8       itabq:  stq     ;task queue
575E 63BC               srq     ;task retry queue
5760 63C0               sckq    ;checksum queue
5762 6568               sfhcq   ;core fake host queue
5764 72AE               srupq   ;SPF update queue
```

```
uribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 418
FAKREL.PLR;1    PAGE 38          IMP Fakes Reliability

                ;VMAPCH - check that none of the variables or buffer pages moved,
                ; or was re-created (INTIME 0).

5766 1076       routine vmapch, uses r1

5768 4818 0010    lda r1,=lmapct
                  repeat
576C 5079 5636      lda r7,mapct(-r1)   ;check to see if maps changed
5770 990F           ifnot odd           ;some entries not to be used
5772 7077             lda r7,(r7)       ;this is map setting
5774 7679 6556        if r7 <> vmap(r1)
5778 9104
577A 6076               fail return     ;page moved}
577C 4FF0
577E 4007
                      endif
5780 9907             ifnot odd         ;there is a page
5782 3078 FC04          sta r7,%map2    ;get to this page
5786 7078 80B4          lda r7,m2#intime        ;its intime
578A 8A02               if zero
578C 90F7                 fail return   ;init time run out or new page
                        endif
                      endif
                    endif
578E 88EF           until loop
                  endrepeat

5790 6006       endroutine vmapch
```

```
                ;       Stage calls here for init check

5792 1076       routine fakini, uses r1-r5, uses m1-m3

5794 7048 40B4    lda r4,intime                 ;timeout running?
5798 9A18         ifnot zero                    ;yes
579A 4078 5766      call vmapch                 ;see if pages new or moved
579E 9A15           if success                  ;all okay
57A0 7078 00B0        lda r7,maprel             ;get to rel page
57A4 3078 FC04        sta r7,%map2
57A8 7078 9EB0        lda r7,m2#pkcst
                      if odd & r7 .bit. =pkcext ;:pkcact ;still trying to reload}
57AC 890B
57AE 4F78 0100
57B2 9A08
57B4 3578 9EB0          eorm r7,m2#pkcst        ;clear bits to stop pkc
57B8 7018 9ED4          lda r1,m2#pkctyp
57BC 7028 9ED6          lda r2,m2#pkcadd
57C0 E102               Trap 402,<;Changing buf pg alloc - page 417>
                      endif
57C2 6076             return equal              ;all is well
57C4 0281
57C6 4007
                    endif                       ;vars page moved
                  endif                         ;timeout stopped
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 420
FAKREL.PLR;1    PAGE 40          IMP Fakes Reliability

                ;We want to reinitialize

57C8 8000         notrap                        ;F-illop for diagnosis
57CA 6076         fail coreturn                 ;need consensus
57CC 4FF0
57CE 4077
57D0 1076
57D2 7028 00D0    lda r2,mapvar
57D6 3028 FC06    sta r2,%map3                  ;stage doesn't do this
57DA 7028 00D2    lda r2,mapv2
57DE 3028 FC04    sta r2,%map2

                ;now start table-driven init

57E2 4818 FFE2    lda r1,=-itabsl               ;first table done in order
                  repeat
57E6 6029 5664      lda r2,itabs+itabsl(r1)+    ;loc
57EA 6039 5664      lda r3,itabs+itabsl(r1)+    ;length
57EE 6049 5664      lda r4,itabs+itabsl(r1)+    ;value
                    repeat
57F2 2042             sta r4,(r2)+              ;do a word
57F4 49B2             sub r3,=2                 ;count 2 bytes
57F6 8AFE             until zero
                    endrepeat
57F8 4E90           until r1 = =0
57FA 81F6
                  endrepeat

                ;now constants

57FC 4818 00AC    lda r1,=itabcl
                  repeat
5800 5029 5664      lda r2,itabc(-r1)   ;value
5804 10A9 5664      sta r2,@itabc(-r1)  ;into loc
5808 88FC           until loop
                  endrepeat

                ;now locks

580A 4818 002C    lda r1,=itabll
580E 48A1         lda r2,=1
                  repeat
5810 10A9 5730      sta r2,@itabl(-r1)  ;ptr(lock)
5814 88FE           until loop
                  endrepeat
```

```
                ;now queues

5816 489A         lda r1,=itabql        ;r2=1 already
                  repeat
5818 5039 575C      lda r3,itabq(-r1)   ;ptr(queue)
581C 303B 0002      sta r3,2(r3)        ;set endpointer
5820 3023           sta r2,(r3)         ;and begin
5822 88FB           until loop
                  endrepeat

                ;*** dispatch tables

5824 4078 5A1A    call baseto

                ;*** imp number

5828 4078 54DE    call conclk   ;get clock(incl imp #)

                ;*** reassembly blocks

582C 4838 6CB0    lda r3,=messtk
5830 48F1         lda r7,=1
                  repeat
5832 300B 4000      sta r0,reaslk(r3)
5836 387B 0003      stab r7,reasst(r3)
583A 4A38 0010      add r3,=lrblk
583E 4E38 6E30      until r3 >= =emsstk
5842 92F8
                  endrepeat

                ;*** message table and allocate stuff

5844 4818 0370    lda r1,=tmblks-rmblks-rmlen   ;;tmlen
5848 4838 FFFF    lda r3,=-1
                  repeat
584C 3039 65B2      sta r3,rmimp(r1)
5850 3039 6932      sta r3,tmimp(r1)
5854 3009 A5B0      sta r0,rmlock(r1)
5858 3009 A930      sta r0,tmlock(r1)
585C 4918 0010      sub r1,=rmlen       ;;tmlen
5860 8BF6           until minus
                  endrepeat
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 422
FAKREL.PLR;1 .  PAGE 42          IMP Fakes Reliability

                ;***   free list

5862 4078 55B4    call bufini   ;initialize the point words

5866 7048 63CC    lda r4,maxbuf ;how many buffers * 2
                  repeat
586A 49C2           sub r4,=2
586C 9B06           until minus
586E 482C 92C0      lda r2,=chain(r4)
5872 4078 5106      call flpool ;flush and count nf
5876 90FA         endrepeat

                ;*** back hosts ***

5878 48D8         lda r5,=b2pend-b2pblk
                  repeat
587A 504D 5984      lda r4,ibt3(-r5)    ;next back
587E 4078 598C      call breini ;init him
5882 4ED0           until r5 = =0
5884 81FB
                  endrepeat

5886 7028 00B0    lda r2,maprel ;get to rely
588A 3028 FC02    sta r2,%map1
588E 4818 FFFF    lda r1,=-1
5892 3018 7EB6    sta r1,m1#pkcrct      ;force pkt core params clear
5896 7028 00D0    lda r2,mapvar
589A 3028 FC02    sta r2,%map1

                ;initialize statistics fake host

589E 4838 0020    lda r3,=2*words* D8
                  repeat
58A2 501B 5954      lda r1,gostt(-r3)
58A6 9B03           ifnot minus
58A8 7518 62E2        eor r1,mine
                    endif
58AC 381B 5E49      stab r1,statdt+1(r3)
58B0 501B 5954      lda r1,gostt(-r3)
58B4 301B 5E48      sta r1,statdt(r3)
58B8 88F5           until loop
                  endrepeat

58BA 4078 5928    call gostat   ;statistics constants
```

```
                ;do the tabled copying

58BE 4818 0020    lda r1,=itabil
                  repeat
58C2 50B9 5710      lda r3,@itabi(-r1)  ;ptr(value)
58C6 10B9 5710      sta r3,@itabi(-r1)
58CA 88FC           until loop
                  endrepeat

                ;finally, remember what pages we initialized

58CC 4818 0010    lda r1,=lmapct
                  repeat
58D0 5029 5636      lda r2,mapct(-r1)   ;get pointer to local map entry
58D4 9909           ifnot odd           ;good entry in mapct
58D6 7022             lda r2,(r2)       ;get map from local
58D8 3029 6556        sta r2,vmap(r1)   ;into vars page
58DC 9905             ifnot odd         ;got a page here
58DE 3028 FC04          sta r2,%map2    ;to this page
58E2 3008 80B4          sta r0,m2#intime        ;mark it initialized
                      endif
                    endif
58E6 88F5           until loop
                  endrepeat

                ;chek for reload/restart

58E8 48A1         lda r2,=rstmrk        ;we've restarted for sure
58EA 7018 5926    lda r1,oldimp
58EE 7118 62E2    sub r1,mine           ;did our core come from us?
58F2 9A0F         ifnot zero            ;somebody else's bod}
58F4 3218 40B6      addm r1,cksum       ;fix our cksum
58F8 3118 5926      subm r1,oldimp      ;make this our core
58FC 48A2           lda r2,=rldmrk      ;tell ncc its a reload
58FE 7038 00C6      lda r3,mapfak+%ncodep       ;get spare fake
5902 9907           ifnot odd                   ;we've got a spare
5904 3038 FC04        sta r3,%map2              ;now fix cksum etc on spare
5908 3218 80B6        addm r1,m2#cksum
590C 3118 9926        subm r1,m2#oldimp ;mark spare core as ours too.
                    endif
                  endif
5910 3028 6320    sta r2,rstrld         ;leave info around for stats

                ;call OPSYS init routine

5914 4818 12B0    lda r1,=itable
                patinit:                ;so it can be patched (TIP)
5918 4078 11FE    call doinit

591C E100         Trap 400,<;IMP reinit (H) - page 421>

591E 4893         lda r1,=3     ;stop future inits
5920 3018 40B4    sta r1,intime

5924 6006       endroutine fakini       ;;returns loop  :done init .

5926            oldimp: .blkw 1         ; imp from which we got o  core
```

```
                ;GOSTAT
                ;calculate and store constants
                ;offset send time for STATS processes using IMP # mod 64

5928 1076       routine gostat, uses r1-r5

592A 4818 0012    lda r1,=statot
                  repeat
592E 5059 4918      lda r5,stbp(-r1)          ;master STATs list
5932 990F           ifnot odd                 ;got a STAT here
5934 48B0             lda r3,=0
5936 702D 000E        lda r2,statf(r5)        ;freq this STAT
593A A6A1             srl r2,1                ;less than flat out?
593C 9A08             ifnot zero
593E 7048 62E2          lda r4,mine           ;IMP number
5942 A7C6               rrl r4,6              ;IMP # mod 64 to high end
                        repeat
5944 A1C1                 rla r4,1            ;a bit of IMP number
5946 A1B1                 rla r3,1            ;multiply impno by interval
5948 A6A1                 srl r2,1
594A 8AFD                 until zero          ;done multiply
                        endrepeat
                      endif
594C 303D 0012        sta r3,impoff(r5)       ;store at IMPOFF
                    endif
5950 88EF           until loop
                  endrepeat

5952 6006       endroutine gostat
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 425
FAKREL.PLR;1    PAGE 45          IMP Fakes Reliability

5954 00FF       gostt:   HFF,0
5956 0000
5958 00FF                HFF,1
595A 0001
595C 80FF                getpri+ HFF,1
595E 0001
5960 80FF                getpri+ HFF,0
5962 0000
5964 00FF                HFF,sign+7
5966 8007
5968 0003                3,1
596A 0001
596C 0000                0,0
596E 0000
5970 00FF                HFF,1
5972 0001

5974 002C       ibt1:    bk0pid
5976 0026                bk1pid
5978 0028                bk2pid
597A 002A                bk3pid
597C 39F4       ibt2:    back5
597E 5072                back7
5980 523E                back9
5982 4FD2                back6
5984 7038       ibt3:    bbk0,bbk1,bbk2,bbk3     ;;b2pblk ;must match b2pblk
5986 7058
5988 7078
598A 7098

                ; (re) initialize a back host. also called from rely code

598C 1076       routine breini, uses r3, arg r4-r5

598E 703D 597C    lda r3,ibt2(r5)
5992 303C 0004    sta r3,hilo(r4)

5996 9002       entry bcomt, uses r3, arg r4-r5
5998 1076

599A 703D 5974    lda r3,ibt1(r5)
599E 383C 000F    stab r3,hinpid(r4)
59A2 304B 81C0    sta r4,mblks(r3)
59A6 4878 398E    lda r7,=back
59AA 307B 80C0    sta r7,base(r3)
59AE 4838 8000    lda r3,=forbak                ;fake and back flag
59B2 303C 0002    sta r3,hostyp(r4)             ;for special hi handling
59B6 300C 4000    unlock lockhi(r4)

59BA 6006       endroutine breini
```

```
59BC 0022       basepo: ttopid  ;22
59BE 0002       basep:  stkpid  ;2
59C0 0004               bltpid  ;4
59C2 0012               ddtpid  ;12
59C4 002E               conpid  ;2E
59C6 0040               task    ;40
59C8 0044               rutpi   ;44 - last pid poked by slow to
59CA 0006       basep2: dsppid  ;6
59CC 0010               bktpid  ;10
59CE 0014               ttypid  ;14
59D0 0016               ihtpid  ;16
59D2 0042               stopid  ;42
59D4 00FC               nopid   ;FC
59D6 0000               0       ;0
59D8 5CD0       basec:  rstgo
59DA 1794               bltcal
59DC 161A               jjddt
59DE 1B80               con
59E0 2436               tsk
59E2 23EE               rutwak
59E4 162A               jdsply
59E6 3BD4               btc
59E8 1622               jjtty
59EA 3BBC               ihtc
59EC 1C72               sto
59EE 1068               nopids
59F0 1142               emty
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 427
FAKREL.PLR;1    PAGE 47         IMP Fakes Reliability

                ;test trace

.comnt |
routine trach
  lda r7,trcptr
  if r7 < =trcbuf } r7 > =trcbuf+trclen
    Trap 3340,<;trace ptr screwed up - page 425>
    lock trclok
    call gotrac
  endif
endroutine trach

routine gotrac
  lda r7,=trcbuf
  sta r7,trcptr
  unlock trclok
endroutine gotrac
|

                ;check the back ground guys

59F2 1076       routine bktst, uses r1, uses r3-r5

59F4 48D8         lda r5,=b2pend-b2pblk ;r5 will point into b2pblk
                  repeat
59F6 504D 5984      lda r4,ibt3(-r5)
59FA 707C 4000      lock lockhi(r4)
59FE 9AFE
5A00 4078 5998      call bcomt          ;common fixes
5A04 4ED0           until r5 = =0
5A06 81F8
                  endrepeat

5A08 6006       endroutine bktst
```

```
                ;maintain fixed pids in base

5A0A 1076       routine basetp, uses r1, uses r3-r4

5A0C 489E         lda r1,=basep2-basepo ;poke some pids
                  repeat
5A0E 5039 59BC      lda r3,basepo(-r1)
5A12 30B8 00AC      sta r3,@pid
5A16 88FC           until loop
                  endrepeat

5A18 9002       entry baseto, uses r1, uses r3-r4
5A1A 1076

5A1C 4818 001A    lda r1,=basec-basep   ;pids to set up
                  repeat
5A20 5039 59D8      lda r3,basec(-r1)
5A24 7049 59BE      lda r4,basep(r1)
5A28 303C 80C0      sta r3,base(r4)
5A2C 88FA           until loop
                  endrepeat
5A2E 7038 40CE    lda r3,ttoadr ;copy from local param
5A32 3038 80E2    sta r3,base+ttopid

5A36 6006       endroutine basetp
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 429
FAKREL.PLR;1    PAGE 49          IMP Fakes Reliability

.comnt |
nice stop code |

     0008       to5sec= D8              ;8 slow ticks are about 5 seconds.
     0018       to15sec= D24            ;24 are thus 15 seconds

.comnt |
nice stop variables |

                    Page FakVars

5F8C            igdown: .blkw 1         ;if set, send imp going down
5F8E            igdflag:.blkw 1         ;if set, send imp going down with code=igdown
5F90            stack ns,<2>            ;nice-stop stack
5F94
5F94
5F94            nsspsv: .blkw 1         ;stack pointer.

FC00 0600               Page FakCode

                nicestop:
                repeat                  ;while IMPSYS running
5A38 7018 6516    lda r1,nsftof                 ;have we already done this
5A3C 4E18 0BAD    ifnot r1 = # HOBAD
5A40 9140
5A42 7018 651C      lda r1,nsrtf                ;check nice-stop flag.
5A46 9A3D           ifnot zero                  ;any stop in progress.
5A48 8B36             if minus                  ;nice halt
                        Trap 1700,<;Imp going down (OPRTOR INIT) - page 427>    ;tell NCC about it.
5A4A E3C0
5A4C 3008 5F8E          set igdflag             ;send an imp going down
5A50 48F0               lda r7,=0
5A52 3078 5F8C          sta r7,igdown           ;use down immediately code
5A56 4078 5ACC          call nsleep,to5sec      ;wait 5 seconds for that to happen
5A5A 0008

                  ;stop host traffic by marking them dead.

5A5C 48F4               lda r7,=hninit          ;host not init.
5A5E 7018 6322          lda r1,hosts            ;mark all hosts this way.
                        repeat
5A62 5029 6326            lda r2,h2pblk(-r1)    ;next host.
5A66 9B03                 ifnot minus           ;host exists.
5A68 387A 001F              stab r7,hihd(r2)    ;mark host
                          endif
5A6C 88FB                 until loop
                        endrepeat

5A6E 4878 FFFF          lda r7,=-1              ;stop hosts from coming alive
5A72 3078 62E0          sta r7,myimp
5A76 4078 5ACC          call nsleep,to5sec      ;wait for 5 seconds.
5A7A 0008
```

```
                ;stop s/f traffic by giving infinite routing

5A7C 3008 651A          set nsrutd              ;flag - declare lines dead in routing
5A80 3008 72B2          set rupsnd              ;SPF - send update
5A84 4878 0044          set @pid = =rutpi               ;poke routing process
5A88 30F8 00AC
5A8C 4078 5ACC          call nsleep,2           ;need only wait two slow ticks
5A90 0002

                ;kill off modems

5A92 7018 6324          lda r1,modems           ;all modems
                        repeat                  ;for each modem.
5A96 5049 635E            lda r4,m2pblk(-r1)    ;modem parameter block.
5A9A 9B08                 ifnot minus           ;modem there.
5A9C 707C 4000              lock lockm(r4)
5AA0 9AFE
5AA2 4078 1CF4              call killin
5AA6 300C 4000              unlock lockm(r4)
                          endif
5AAA 4E90                 until r1 = =0
5AAC 81F5
                        endrepeat
5AAE 4078 5ACC          call nsleep,to5sec      ;wait 5 seconds.
5AB2 0008
                      endif
5AB4 3008 6518        set nspc                  ;Calling address
5AB8 4878 0BAD        set nsftof = # HOBAD      ;tell fast timeout to stop all procs
5ABC 3078 6516
                    endif
                  endif
5AC0 3018 651A    sta r1,nsrutd ;;nsrtf ;keep out of routing's hair
5AC4 4078 5ACC    call nsleep,1                 ;sleep 1 slow tick always.
5AC8 0001
5ACA 90B7       endrepeat

                ;(OVER)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 431
FAKREL.PLR;1    PAGE 51          IMP Fakes Reliability

                ;sleep routine for nice stop.

5ACC 6017       routine nsleep, inline r1, uses r2-r5
5ACE 1076
                  repeat
5AD0 4991           sub r1,=1
5AD2 9B0A           until minus                 ;time runs out.
5AD4 1016           push r1
5AD6 3068 5F94      sta sp,nsspsv               ;save stack pointer.
5ADA 4008 1BEE      tr tossx
                    nschk:
5ADE 7068 5F94      lda sp,nsspsv               ;restore stack pointer.
5AE2 6016           pop r1
5AE4 90F6         endrepeat
5AE6 6006       endroutine
```

```
0000        .if nz VDHSw
              .INSERT "VDHCOD"
            .endc ;nz VDHSw

0001        .if nz VHA
              .INSERT "VHA"
                .INSRT VHA
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 433
VHA.PLR;1       PAGE 1  Virtual Host Addressing

                .stitle Virtual Host Addressing

                ;NSA Platform Network
                ;Virtual Host Addressing (VHA) Package

                ;Provides for readdressing of messages upon entry to the
                ;Platform Network from a Virtual Host Address Space into
                ;the Physical Address Space which the IMPs use, and the
                ;resolution of such addresses back into Virtual Host
                ;Addresses upon exit from the network.

                ;parameters

     007F       nVHAi = nimp                    ;lots of space for identity map
     0190       maxvha= D400
                        page Vars               ;locks on Vars page

                locdef vhalok,<;Lock on VHA inverse translation table>
73FE            vhinit: .blkw 1                 ;flag that zeroes when IMP reinits

                        page VHAVars

5B5A            table vhinvt                    ;IMP-to-VHALIS index
5B5A              .blkw nVHAi                   ; IMPs for now
                endtable vhinvt
5C58              .blkw 1                       ;end of VHALIS

5C5A            vharix: .blkw 1                 ;current entry to check
5C5C            vharxi: .blkw 1                 ;current inverse entry to check
5C5E            vhaoser: .blkw 1                ;serial number last check

5C60            table vhalis                    ;list of VHA address by IMP,host
5C60               .blkw maxvha
                endtable vhalis
```

```
                ;Host Input (HI) routine
                ;Called after leader has been transformed into "new"
                ;(i.e. internal) format in the transaction block,
                ;but before MESGET has been called. Transforms from a
                ;Virtual to a Physical address. If no mapping is defined
                ;defined for this Virtual Address, return a destination
                ;dead, subtype 2 message.

                        page LCode

                ;HIVHA
                ;Host Input routine for transforming virtual to
                ;physical addresses. Called with R3/transaction block,
                ;R4/host parameter block. Fail if no mapping this VHA.
                ;Adjust transaction block address if map defined.

3D40 1076       routine hivha,arg r3-r4,uses r1

3D42 705B 0006     lda r5,trdstl(r3)
3D46 781B 0005      ldab r1,trhstl+1(r3)
3D4A 4E18 00FC      if r1 >= = H100-nfh & r1 <= =  H100
3D4E 9205
3D50 4E18 0100
3D54 9C02
3D56 6006           return
                    endif
3D58 707C 001C    lda r7,homode(r4)
3D5C 8B07         if minus                      ;new leader
                ; if byte homode(r4) .bit. = H80       ;new leaders
3D5E 787B 0007     if byte trdstl+1(r3) ;physical address
3D62 9A02
3D64 90F9            return
                    endif
3D66 4D15          eorb r1,r5
3D68 9009          else
3D6A A296           sll r1,6                     ;recreate old leader
3D6C 741B 0006      ior r1,trdstl(r3)
3D70 381B 0005      stab r1,trhstl+1(r3)
3D74 48F0           clear trdstl(r3)             ;but mark trn blk as "virtual"
3D76 307B 0006
                  endif
3D7A 4078 3D98    call vhamap                    ;look up physical address
3D7E 8A04         if fail
3D80 6076           fail return                  ;no mapping this VHA
3D82 4FF0
3D84 4007
                  endif
3D86 48F0          clear trdstl(r3)
3D88 307B 0006
3D8C 381B 0007    stab r1,trdstl+1(r3)           ;store dest IMP (LH always =0)
3D90 A798         rrl r1, H8                     ;get host number
3D92 381B 0005    stab r1,trhstl+1(r3)           ;and store it

3D96 90E0       endroutine hivha
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 435
VHA.PLR;1       PAGE 3  Virtual Host Addressing

                ;VHAMAP
                ;Look up physical host from virtual one, putting result
                ;into R1. If no map defined or out of range, fail return.
                ;R1/virtual host number (0-255), returns physical host,,IMP

                        Page LCode

3D98 1076       routine vhamap,arg r1,result r1,local m0
3D9A 7078 40BE
3D9E 1076

3DA0 7078 00B2    setmap m0,mapvha              ;map in the VHA tables
3DA4 3078 FC00
3DA8 7618 4840    if r1 < vhalen                ;in range
3DAC 820B
3DAE A291           sll r1,1                    ;get word offset
3DB0 7019 4842      lda r1,vhatab(r1)           ;get physical address
3DB4 9A07           ifnot zero                  ;one is defined
3DB6 6076             return
3DB8 707F 00B0
3DBC 3078 FC00
3DC0 6006
                    endif
                  endif
3DC2 6076         fail return
3DC4 707F 00B0
3DC8 3078 FC00
3DCC 6076
3DCE 4FF0
3DD0 4007

                endroutine vhamap
```

```
.comnt |
;VHAINV
;Look up a virtual host address, given its physical
;address. Does a quick identity mapping check, then
;searched the table for a match. MAY RUN FOR UP TO
;3 MILLISECONDS WORST CASE.
;Fail return if no Virtual for this Physical

        page LCode

routine vhainv,arg r1,result r1,local m0

  setmap m0,mapvha              ;get to VHA tables
  lda r7,r1                     ;repack host,,imp to old leader
  srl r7,2
  eor r7,r1
  and r7,= HFF
  sll r7,1                      ;get word index
  ifnot r1 = vhatab(r7)         ;not identity map
    lda r7,=-lvhatab-words      ;search forward
    repeat
      until r1 = vhatab+lvhatab+words(r7)+
      until loop                ;not found
    endrepeat
    if loop                     ;fail case
      fail return               ;no virtual for this physical
    endif
    add r7,=lvhatab             ;get to positive index
  endif
  srl r7,1                      ;get a host number

endroutine vhainv
|
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 437
VHA.PLR;1       PAGE 5  Virtual Host Addressing

                ;INVMAP
                ;Look up a vitrual Host number, given its physical address.
                ;Uses calculated inverse tables (1) by IMP, giving index in
                ;table 2 for host 0 of this IMP, and (2) by IMP/Host, giving
                ;virtual Host number for each physical Host in the network.
                ;Arguments are R1/IMP number, R2/Host on that IMP.
                ;Succeed return with Virtual Host number in R1, index
                ;into VHALIS in R2. Fail if no physical host in list.

                        page LCode

3DD2 1076       routine invmap,arg r1-r2,result r1-r2,local m0
3DD4 7078 40BE
3DD8 1076

3DDA 7078 00B2    setmap m0,mapvha              ;get to VHA tables.
3DDE 3078 FC00
3DE2 A291         sll r1,1                      ;word index by IMP
3DE4 4E18 00FE    if r1 < =lvhinvt              ;IMP no. too big
3DE8 8217
3DEA A2A1           sll r2,1
3DEC 7078 B3FC      lock vhalok                 ;prevent changing
3DF0 9AFE
3DF2 7229 5B5A      add r2,vhinvt(r1)
3DF6 7629 5B5C      if r2 < vhinvt+words(r1)    ;make sure it's in range
3DFA 820C
3DFC 701A 5C60        lda r1,vhalis(r2) ;look up virtual name
3E00 9A09             ifnot zero                ;there is one
3E02 3008 B3FC          unlock vhalok
3E06 6076               return
3E08 707F 00B0
3E0C 3078 FC00
3E10 6006
                      endif
                    endif
3E12 3008 B3FC      unlock vhalok
                  endif
3E16 6076         fail return                   ;no VHA this Host/IMP
3E18 707F 00B0
3E1C 3078 FC00
3E20 6076
3E22 4FF0
3E24 4007

                endroutine invmap
```

```
                ;IHVHA
                ;Perform Physical to Virtual Host Address Mapping for
                ;IH side. Adjusts address in the IH leader area (IHLEDR)
                ;in the current parameter block (via R4).

                        Page LCode

3E26 1076       routine ihvha,arg r4,uses r1-r2

3E28 782C 0015    ldab r2,ihledr+hstl+1(r4)     ;pick up host number
3E2C 781C 0017    ldab r1,ihledr+dstl+1(r4)     ;get IMP who sent leader
3E30 8A07         if zero                       ;already virutal
3E32 707C 001C      lda r7,homode(r4)           ;which mode
3E36 9B03           ifnot minus                 ;old leaders
3E38 4812              lda r1,r2                ;get virtual packed
3E3A 9010              br ihvha1                ;and stuff it in
                    endif
3E3C 901E           else                        ;not already virtual
3E3E 4078 3DD2        call invmap               ;got a virtual address?
3E42 9A13             if success
                ;       if byte homode(r4) .bit. = H80    ;new format
3E44 707C 001C          lda r7,homode(r4) ;;sign
3E48 8B09               if minus                ;new format
3E4A 777C 4016            tst r7,ihledr+dstl+foo(r4)      ;set IMP <- 0
3E4E 381C 0015            stab r1,ihledr+hstl+1(r4)       ;and host <- VH number
3E52 A698               srl r1, H8
3E54 381C 0016           stab r1,ihledr+dstl(r4)
3E58 9007                else
                ihvha1:
3E5A A796                  rrl r1,6        ;;srceh ;just Host
3E5C 381C 0015             stab r1,ihledr+hstl+1(r4)
3E60 A69A                  srl r1, D10     ;;srcei ;just IMP
3E62 301C 0016             sta r1,ihledr+dstl(r4)
                         endif
3E66 9009                else                       ;inverse map failed
3E68 787C 0015            if byte ihledr+hstl+1(r4)< = H100-nfh     ;real host, tho
3E6C 4E78 00FC
3E70 8204
3E72 701C 0016              lda r1,ihledr+dstl(r4)  ;report IMP number
3E76 E302                   Trap 1402,<;IHVHA: No virtual address found>
                          endif
                        endif
                      endif

3E78 6006       endroutine ihvha
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 439
VHA.PLR;1       PAGE 7  Virtual Host Addressing

                ;TSKVHA
                ;Task For Us routine to check source host in an
                ;incoming packet. R1/M2/packet pointer,
                ;R5/ RM message block, if any

                        page LCode

3E7A 1076       routine tskvha,arg r1/m2,arg r5,local r2-r3,local r5,uses r1
3E7C 1026
3E7E 1036
3E80 1056

3E82 7829 0009    ldab r2,seqh+1(r1)            ;assume host in msg
3E86 4855         lda r5,r5                     ;any RM block?
3E88 9903         ifnot odd
3E8A 782D 65B4      ldab r2,rmhost(r5)  ;get RM host number
                  endif
3E8E 4E28 00FC    if r2 < = H100-nfh & r2 <= = H100             ;a real host
3E92 8212
3E94 4E28 0100
3E98 9C0F
3E9A 7019 0006      lda r1,srch(r1)             ;sending IMP
3E9E 4852           lda r5,r2                   ;remember host for Trap
3EA0 4831           lda r3,r1                   ; and IMP
3EA2 4078 3DD2      call invmap                 ;this port got a VHA?
3EA6 8A08           if fail                     ;nope
3EA8 E303             Trap 1403,<:TSKVHA: No virtual address this source>
3EAA 6056             fail return
3EAC 6036
3EAE 6026
3EB0 6076
3EB2 4FF0
3EB4 4007
                    endif
                  endif

3EB6 6056       endroutine tskvha
3EB8 6036
3EBA 6026
3EBC 6006
```

```
                ;VHA timeout routine VHAREL
                ;check serial number and one entry in VHALIS each wakeup
                ;if serial number has changed or the entry lookw wrong,
                ;recompute the inverse tables VHINVT/VHALIS.

FC00 0200               page VHACode

5626 1076       routine vharel,uses r1-r5

5628 7078 73FE    ifnot vhinit                  ;system reinit
562C 8A0C
562E 7078 B3FC      cklock vhalok               ;lock out others
5632 48F0           lda r7,=0
5634 4818 0100      lda r1,=lvhinvt+words       ;empty the reverse table
                    repeat
5638 1079 5B5A        sta r7,vhinvt(-r1)        ;clear all entries
563C 88FE             until loop
                    endrepeat
563E 3008 B3FC      unlock vhalok
5642 9022          else
5644 7078 5C5E      if vhaoser = vhaser         ;same serial #
5648 7678 483E
564C 811D
564E 7048 5C5A        lda r4,vharix             ;next to check
5652 4AC2             add r4,=words
5654 9B04             if minus } r4 >= vhinvt+lvhinvt   ;time to wrap
5656 7648 5C58
565A 9202
565C 48C0               lda r4,=0
                      endif
565E 3048 5C5A        sta r4,vharix
5662 4078 578E        call vhlchk               ;check VHALIS entry
5666 9A0F             if success                ;so far, so good
5668 7018 5C5C          lda r1,vharxi           ;r1 is arg to vhmchk
566C 4A91               add r1,=1
566E 9B04              if minus } r1 >= vhalen
5670 7618 4840
5674 9202
5676 4890               lda r1,=0
                       endif
5678 3018 5C5C        sta r1,vharxi
567C 4078 5768          call vhmchk             ;check forward map
5680 9A02               if success              ;all's in order
5682 6006                 return
                        endif
                      endif
5684 E301             Trap 1401,<;VHAREL: Detected VHA table error>
                    endif
                  endif
5686 3008 73FE    set vhinit                    ;note intialization
568A 7078 483E    set vhaoser = vhaser          ;note where we are now
568E 3078 5C5E
                ;(OVER)
```

```
Pluribus IMP 1301      PLURIBUS V2.9B  25-Jun-87 10:57:29      PAGE 441
VHA.PLR;1       PAGE 9  Virtual Host Addressing

                ;Get here when inverse table must be recomputed
                ;Put VHA addresses into reverse lookup table

5692 7018 4840    lda r1,vhalen                 ;length of this VHA table
5696 A291         sll r1,1                      ;in bytes
                  repeat
                    repeat
5698 5039 4842        lda r3,vhatab(-r1)        ;next V->P mapping
569C 9A4C             ifnot zero                ;a valid one
569E A6B8               srl r3, H8              ;Host number
56A0 7829 4843          ldab r2,vhatab+1(r1)
56A4 A2A1               sll r2,1                ;and IMP * 2
56A6 4E28 00FE          if r2 >= =1vhinvt       ;too big for inverse table
56AA 9203
56AC E305                  Trap 1405,<;VHAREL: VHA IMP number too big>
56AE 9043               else
56B0 A691                 srl r1,1              ;true VHA number
56B2 A2B1                 sll r3,1
56B4 723A 5B5A            add r3,vhinvt(r2)     ;slot this Host/IMP
56B8 4E38 0320            if r3 >= =1vhalis     ;out of inverse list space}
56BC 9207
56BE E304                   Trap 1404,<;VHAREL: Too many VHA numbers>
56C0 48F0                   clear r1,vhinit     ;try to reinit anyway
56C2 3078 73FE
56C6 4817
56C8 9039                   break               ;but stop checking
                          endif
                ;(OVER)
```

```
                ;Get here looking to insert a new VHA.  See whether its
                ; slot is available, and make roon for it if not.

56CA 763A 5B5C              if r3 >= vhinvt+words(r2)     ;must make room}
56CE 9230
56D0 484B 0002                lda r4,=words(r3)   ;make at least one slot
56D4 714A 5B5C                sub r4,vhinvt+words(r2)     ;total needed
56D8 1016                     save r1,r3          ;will need these later
56DA 1036
56DC 7018 5C58                lda r1,vhinvt+lvhinvt       ;end of VHALIS
56E0 4A14                     add r1,r4           ;new end
56E2 48F0                     lda r7,=0
                              repeat              ;move up end of VHALIS
56E4 1079 5C60                  sta r7,vhalis(-r1)        ;clearing as we go
56E8 7618 5C58                  until r1 = vhinvt+lvhinvt
56EC 81FC
                              endrepeat
56EE 3248 5C58                addm r4,vhinvt+lvhinvt      ;update end index
56F2 4818 00FE                lda r1,=lvhinvt     ;next IMP's indices to move
                              repeat              ;for all IMPs higher than ours
56F6 5059 5B5C                  lda r5,vhinvt+words(-r1)  ;where next IMP now is
56FA 4E12                       until r1 = r2
56FC 9117
56FE 4835                       lda r3,r5
5700 4934                       sub r3,r4         ;and where it was
5702 7078 B3FC                  lock vhalok       ;prevent access while changing
5706 9AFE
                                repeat
5708 7639 5B5A                    until r3 = vhinvt(r1)   ;until got our IMP
570C 9108
570E 507B 5C60                    set vhalis(-r5),r7 = vhalis(-r3)
5712 107D 5C60
5716 357B 5C60                    eorm r7,vhalis(r3)      ;and clear old one
571A 90F7                       endrepeat
571C 3059 5B5A                  sta r5,vhinvt(r1) ;update this IMP's base
5720 3008 B3FC                  unlock vhalok     ;table is consistent again
5724 4078 1C34                  call tsleep       ;rest often
5728 90E7                     endrepeat
572A 6036                     restore r1,r3       ;get back our regs
572C 6016
                            endif                 ;slot is now available
                ;(OVER)
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 443
VHA.PLR;1       PAGE 11         Virtual Host Addressing

                ;Here with VH address to insert in VHALIS slot

572E 301B 5C60            sta r1,vhalis(r3)     ;and set inverse VH address
5732 A291                 sll r1,1              ;and fix to word offset
                        endif
                      endif
5734 4F18 001E        while r1 .bit. = H1E      ;16 entries per sleep
5738 8AB0
                    endrepeat
573A 4078 1C34      call tsleep
573E 4E90           until r1 = =0
5740 81AC
                  endrepeat
5742 7048 5C58    lda r4,vhinvt+lvhinvt          ;now must check for old VHAs
5746 9A0F         ifnot zero                     ;if there are any at all...
                    repeat
                      repeat
5748 49C2               sub r4,=words
574A 4078 578E          call vhlchk             ;this entry okay
574E 8A04               if fail                 ;no, but maybe just old
                ;         clear vhalis(r4)      ;so get rid of it
5750 48F0               lda r7,=0
5752 307C 5C60          sta r7,vhalis(r4)
                        endif
5756 4F48 001E          while r4 .bit. = H1e    ;16 entries per sleep
575A 8AF7
                      endrepeat
575C 4078 1C34        call tsleep
5760 4EC0             until r4 = =0
5762 81F3
                    endrepeat
                  endif
5764 E300         Trap 1400,<;VHAREL: finished VHALIS recomputation>

5766 908E       endroutine vharel


                ;
                ;=====CALLIS set vharel to global (call it gvharel)
                ;
3EBE 1076               global gvharel=vharel
3EC0 7078 40BE
3EC4 1076
3EC6 7078 00B2
3ECA 3078 FC00
3ECE 4078 5626
3ED2 4008 1BCE
FC00 0200
```

```
                ;Routines to check VHA entries

                ;VHMCHK
                ;looks up physical address from virtual, than looks up
                ;virtual fromt he result. Fail if they don't match.
                ;R1/ VHA to look up, returned in R3. R1 returns reverse
                ;VHA, R2 index into VHALIS for inverse lookup but only
                ;in succeed case.

FC00 0200               page VHACode

5768 1076       routine vhmchk,arg r1,result r1-r3

576A 4831         lda r3,r1                     ;remember our argument
576C 9A10         ifnot zero                    ;no need to check empties
576E 4078 3D98      call vhamap                 ;get physical address
5772 9A0D           if success                  ;if there is one..
5774 4821             lda r2,r1                 ;unpack the result
5776 A6A8             srl r2, H8                ;just Host
5778 4B18 00FF        and r1,= HFF              ;just IMP
577C 4078 3DD2        call invmap               ;now inverse map it
5780 9A03             if fail } r1 <> r3        ;map problems
5782 4E13
5784 9104
5786 6076               fail return
5788 4FF0
578A 4007
                      endif
                    endif
                  endif

578C 6006       endroutine vhmchk

                ;VHLCHK
                ;Check indicated entry in VHALIS
                ;VHMCHK does all the work; R4 is index of VHALIS entry
                ;to check.

578E 1076       routine vhlchk,arg r4,result r1-r4

5790 701C 5C60    lda r1,vhalis(r4)             ;get the VHA to check
5794 9A09         ifnot zero                    ;if there's one here
5796 4078 5768      call vhmchk                 ;validate it
579A 9A03           if fail } r2 <> r4          ;problems with this entry
579C 4E24
579E 9104
57A0 6076             fail return
57A2 4FF0
57A4 4007
                    endif
                  endif

57A6 6006       endroutine vhlchk
```

```
Pluribus IMP 1301       PLURIBUS V2.9B  25-Jun-87 10:57:29       PAGE 445
IMP.MAIN;1      PAGE 7.19       Virtual Host Addressing

                .endc

FC00 0000       $finstage                       ;do finish stuff.
40DE 7D50
40E0 7B00
40E2 7E80
40E4 7E00
40E6 FFFF
40E8 FFFF
40EA FFFF
40EC FFFF
40FA 0001
40FC 0001
40FE 0000
4100 4000
559E 0002
4104 0010
4106 0020
4108 0030
410A 0050
410C 0006
410E 0016
4110 0022
4112 003A
4114 0004
4116 E100
4118 E200
411A F100
411C F200
5016 5020
5018 5056
501A 53EA
501C 5496
501E 1072
40BA 0000
40BC 567C
031A 04DC
0314 00C8
0316 0001
0318 0001
0320 E000
0322 F000
0324 8000
0326 8000
                ; $itable                       ;put in init tables, etc.
FC00 0600       $finimp                         ;fix IMP parameters
40D6 7420

                lstlim <LCode,RelCode,DDTCode,Warm,FakCode,PkgCode>
                lstlim <LTVars,LVars,Vars,RelVars,DDTVars,VHAVars,PkgVars,FakVars>
                lstlim <VarPat,LKPatch,RKPatch>
```

```
.iif p2, .outsym imp.sym
```