```
024154 045667 5           lda dispn x          ;bufchk and rstsws enable
024155 010001 5           sta ireg
024156 102001 5           jmp ireg i

;t.o routines return here
024157 025711 5   to6:    irs todisp           ;any more to dispatch?
024160 003153 5            jmp to4             ;yes, go do it.
024161 003121 5           jmp todb             ;no, timeout is done
```

```
024162 000000 5   resett: 0                    ;reset all transmit tables
024163 011572 5          sta rstimp
024164 033573 5          stx rstblk
024165 005732 5          lda [-ntsb]
024166 011574 5          sta rstcnt
024167 005733 5          lda [tsbtab]          ;*reset transaction blocks
024170 011575 5          sta rstptr
024171 001001 5          .inh fre
                                               ;****move down later
024172 073575 5 0 rest1: ldx rstptr
024173 044044 5 0        lda lms x
024174 141050 5 0        cal
024175 101040 5 0        snz                   ;anything there?
024176 003250 5 0         jmp rest3            ;no
024177 015734 5 0        add [tmblkt]          ;yes,
024200 011604 5 0        sta comblk            ;save transmit block pointer

024201 044154 5 0        lda ldi x             ;get foreign imp
024202 021541 5 0        jst rstchk            ;to be reset?
024203 003250 5 0         jmp rest3            ;no
024204 044220 5 0        lda lid x             ;set subcode
024205 006032 5 0        ana [linkno]
024206 013577 5 0        era rstsbc
024207 050220 5 0        sta lid x
024210 140040 5 0        cra                   ;make tsb free
024211 050044 5 0        sta lms x
024212 050000 5 0        sta lch x             ;and clear chain ptr
024213 066264 5 0        ima lpk x             ;clear and get pkt ptr
024214 101040 5 0        snz                   ;a pkt there?
024215 003222 5 0         jmp rest2            ;no
024216 010000 5 0        sta 0                 ;yes, flush it
024217 120115 5 0        jst flushi i
024220 024224 5 0        irs nres              ;and give back to reassembly

024221 073575 5 0        ldx rstptr
024222 044110 5 0 rest2: lda lcd x             ;get tsb local host
024223 023735 5 0        cas [tsbgvb]          ;is it a giveback?
024224 003250 5 0         jmp rest3            ;yes, no mess to host
024225 003250 5 0         jmp rest3            ;yes, no mess to host
024226 141050 5 0        cal 0&desths          ;no, leave dest host
024227 050110 5 0        sta lcd x             ;and get local host
024230 073604 5 0        ldx comblk
024231 121736 5 0        jst [hostno] i        ;get local host
024232 011603 5 0        sta comhst            ;save it for q index
024233 044160 5 0        lda mbfor x           ;get handling type
024234 006075 5 0        ana [mbhand]
024235 073575 5 0        ldx rstptr
024236 052110 5 0        era lcd x
024237 050110 5 0        sta lcd x             ;into tsb
024240 005576 5 0        lda rstcde            ;set mess type
024241 050044 5 0        sta lms x
024242 004000 5 0        lda 0                 ;put tsb block
024243 073603 5 0        ldx comhst
024244 167737 5 0        ima [ehrq] ix         ;on host reply queue
024245 010001 5 0        sta ireg
024246 005575 5 0        lda rstptr
024247 110001 5 0        sta ireg i
```

```
024250 025575 5 0 rest3:  irs rstptr              ;loop over all tsb entries
024251 025574 5 0         irs rstcnt
024252 003172 5 0          jmp rest1
024253 000401 5 0         .enb t.o
024254 005740 5           lda [-nmb]
024255 011606 5           sta mbtcnt
024256 073734 5           ldx [tmblkt]            ;don't do block #0
024257 021623 5   rest5:  jst mbtget               ;*reset trn message blocks
024260 003337 5            jmp rest7
                  .lck h2i
;all done
024261 003306 5 4          jmp rest6              ;age 4-15--check it out
024262 101040 5 4 rest51: snz                     ;timer set?
024263 064250 5 4          irs mbtim x            ;yes,  age
024264 044340 5 4         lda mbmes x
024265 006037 5 4         ana [mbrst+mbinit]
024266 100040 5 4         sze
024267 012037 5 4          era [mbrst+mbinit]
024270 101040 5 4         snz                     ;is rst or init on?
024271 003257 5 4          jmp rest5              ;neither or both
024272 105741 5 4         lda [tim1to] i          ;yes, must inc in 1.8 seconds
024273 013572 5 4 rest52: era rstimp              ;-1 is active state, **if t.o
024274 100040 5 4         sze                     ;time ticked for inc?
024275 003257 5 4          jmp rest5              ;no
024276 044340 5 4         lda mbmes x             ;yes
024277 006034 5 4         ana [mbinct]
024300 101040 5 4         snz                     ;is inc timer already out?
024301 003257 5 4          jmp rest5              ;yes
024302 044340 5 4         lda mbmes x             ;no? decr inc
024303 016062 5 4         sub [mbinc1]
024304 050340 5 4         sta mbmes x
024305 003257 5 4         jmp rest5

024306 011574 5 4 rest6:  sta rstcnt              ;save timer
024307 044340 5 4         lda mbmes x
024310 007742 5 4         ana [mbrst+mbinit+mbstp+mesbts]
024311 012007 5 4 rest60: era [mesbts]
024312 100040 5 4         sze                     ;any msgs out or rst,init, or
024313 003316 5 4          jmp rest62             ;yes,maybe cannot age
024314 005574 5 4 rest61: lda rstcnt              ;no, try to age
024315 003262 5 4         jmp rest51

024316 006035 5 4 rest62: ana [mbrst+mbinit+mbstp]
024317 100040 5 4         sze                     ;rst, init or stp on?
024320 003323 5 4          jmp rest63             ;yes
024321 105743 5 4         lda [tim15s] i          ;no, msgs outstanding, so
024322 003273 5 4         jmp rest52              ;try to tick inc tr timer
```

```
024323 012035 5 4 rest63: era [mbrst+mbinit+mbstp]
024324 100040 5 4         sze                        ;all three on?
024325 003331 5 4          jmp rest64                ;no, go try to age
024326 050000 5 4         sta mbimp x                ;yes, so flush
024327 025605 5 4         irs mbtfre                 ; count
024330 003257 5 4         jmp rest5

024331 012071 5 4 rest64: era [mbrst]                ;are we stopping?
024332 101040 5 4         snz
024333 003314 5 4          jmp rest61                ;yes, age+tick inc t.o quickl
024334 044340 5 4         lda mbmes x
024335 006007 5 4         ana [mesbts]
024336 003311 5 4         jmp rest60

                    .lev t.o
024337 111744 5   rest7:  sta [b1fcnt] i             ;save free count
024340 073745 5           ldx [-th]
024341 001001 5   rest9:  .inh h2i
024342 145746 5 4         lda [hib1+th] ix
024343 013573 5 4         era rstblk
024344 101040 5 4         snz                        ;blk ptr match?
024345 165747 5 4          irs [hibg+th] ix          ;yes, tell H2I we stole it's
024346 000401 5 4         .enb t.o
024347 024000 5           irs 0                      ;loop over all hosts
024350 003341 5            jmp rest9
024351 103162 5           jmp resett i


024352 000000 5   resetr: 0                           ;reset all receive tables
024353 011572 5           sta rstimp
024354 033573 5           stx rstblk
024355 005750 5           lda [-nreab+1]             ;*reset reassembly
024356 011574 5           sta rstcnt
024357 073751 5           ldx [reastb+1]
                    resr1:
024360 001001 5           .inh all
024361 044040 5 0         lda rms x                  ;anything there?
024362 101040 5 0         snz
024363 003375 5 0          jmp resr2                 ;no
024364 006005 5 0         ana [77] 0&blknum&nmb      ;yes, get block number
024365 015752 5 0         add [rmblkt]
024366 011604 5 0         sta comblk                 ;its block pointer
024367 105604 5 0         lda comblk i               ;its imp #
024370 021541 5 0         jst rstchk
024371 003375 5 0          jmp resr2                 ;not reset
024372 033575 5 0         stx rstptr                 ;reset, save reas ptr
024373 121753 5 0         jst [reasf] i
024374 073575 5 0         ldx rstptr
                    resr2:
024375 000401 5 0         .enb t.o
024376 024000 5           irs 0
024377 025574 5           irs rstcnt
024400 003360 5            jmp resr1
```

```
024401 073754 5           ldx [reqstk]          ;*reset request stack
024402 001001 5           .inh all
024403 044000 5 0 resr3:  lda 0 x
024404 101040 5 0         snz                   ;anything there?
024405 003412 5 0          jmp resr5            ;no
024406 021512 5 0         jst rstpk1            ;yes, check it out
024407 100000 5 0          skp                  ;no reset
024410 021532 5 0           jst rstpk2          ;reset pkt
024411 003403 5 0         jmp resr3             ;and go on

024412 005745 5 0 resr5:  lda [-th]
024413 011574 5 0         sta rstcnt
024414 072006 5 0         ldx zero
024415 033603 5 0         stx comhst
024416 005755 5 0 resr6:  lda [shq]             ;* resett packets on host reg
024417 015603 5 0         add comhst            ;ptr to current host queue
024420 073603 5 0 resr6a: ldx comhst
024421 011602 5 0         sta rstihq            ;save
024422 153756 5 0         era [ihwq] ix         ;same as ih queue?
024423 100040 5 0         sze
024424 003446 5 0          jmp resr7            ;no
024425 144300 5 0         lda ihspi ix          ;yes, reset packets in host i
024426 006015 5 0         ana [ 177776 ]
024427 101040 5 0         snz                   ;a packet going out? (as oppo
024430 003446 5 0          jmp resr7            ;no
024431 105602 5 0         lda rstihq i          ;yes, get next pkt
024432 021512 5 0         jst rstpk1            ;or to be reset?
024433 003446 5 0          jmp resr7            ;no
024434 073603 5 0         ldx comhst            ;yes, force ignoring of rfnm

024435 164305 5 0         irs ihiri ix          ;by setting IHIR (ignore RFNM
024436 105602 5 0         lda rstihq i          ;no, get first packet on queu
024437 010000 5 0 resr6b: sta 0                 ;packet pointer
024440 044013 5 0         lda pkth x
024441 006070 5 0         ana [lstpkt]
024442 100040 5 0         sze                   ;last packet?
024443 003447 5 0          jmp resr7a           ;yes
024444 044000 5 0         lda ptrc x            ;no, get next packet
024445 003437 5 0         jmp resr6b            ;and loop
```

```
024446 073602 5 0 resr7:  ldx rstihq          ;packet pointer
024447 005602 5 0 resr7a: lda rstihq          ;queue pointer
024450 015757 5 0         add [ehq-shq]
024451 011601 5 0         sta rstpqt          ;ptr to queue end ptr
024452 044000 5 0 resr7b: lda ptrc x
024453 101040 5 0         snz                 ;anything there?
024454 003464 5 0          jmp resr7d         ;no, quit
024455 021512 5 0 resr7c: jst rstpk1          ;yes, check for reset
024456 003452 5 0          jmp resr7b         ;go back
024457 021532 5 0         jst rstpk2          ;reset
024460 044000 5 0         lda ptrc x
024461 100040 5 0         sze                 ;last one on queue?
024462 003455 5 0          jmp resr7c         ;no, go back
024463 133601 5 0         stx rstpqt i        ;yes, fix end pointer
024464 005602 5 0 resr7d: lda rstihq          ;get current queue pointer
024465 015760 5 0         add [shpq-shq]      ;*reset packets on host pri q
024466 023761 5 0         cas [shpq+th]       ;finished with pri queue?
024467 003472 5 0          jmp resr7e         ;yes
024470 003472 5 0          jmp resr7e         ;yes
024471 003420 5 0         jmp resr6a          ;no, loop
                   resr7e:
024472 000401 5 0         .enb t.o
024473 025603 5           irs comhst
024474 001001 5           .inh all
024475 025574 5 0         irs rstcnt          ;loop over all hosts
024476 003416 5 0          jmp resr6
024477 005740 5 0         lda [-nmb]
024500 011606 5 0         sta mbtcnt
024501 073752 5 0         ldx [rmblkt]        ;don't do block #0
024502 021623 5 0 resr8:  jst mbtget           ;*reset receive message blks

024503 003510 5 0          jmp resr9          ;all done
024504 101000 5 0          nop                ;ignore skip or double-skip
024505 101040 5 0         snz                 ;timer set?
024506 064250 5 0          irs mbtim x        ;yes,  age
024507 003502 5 0         jmp resr8

                   .lev t.o
024510 111762 5   resr9:  sta [b2fcnt] i      ;save free count
024511 103352 5           jmp resetr i
```

```
                        .lev t.o
                        .lck all
024512 000000 5 0 rstpk1: 0                     ;chk a pkt for reset
024513 033575 5 0         stx rstptr
024514 010000 5 0         sta 0
024515 044015 5 0         lda midh x            ;check for raw pkts
024516 006011 5 0         ana three 0&subunc
024517 012011 5 0         era three
024520 100040 5 0         sze                   ;fake receive block 0 if raw

024521 044012 5 0          lda seqh x           ;form receive block ptr
024522 141050 5 0         cal 0&blknum
024523 015752 5 0         add [rmblkt]
024524 011604 5 0         sta comblk
024525 044011 5 0         lda srch x
024526 021541 5 0         jst rstchk            ;to be reset?
024527 100000 5 0          skp                  ;no
024530 025512 5 0           irs rstpk1          ;yes
024531 103512 5 0         jmp rstpk1 i

024532 000000 5 0 rstpk2: 0                     ;do pkt reset
024533 044000 5 0         lda ptrc x
024534 111575 5 0         sta rstptr i          ;fix chain ptrs on queue
024535 120115 5 0         jst flushi i          ;flush packet
024536 024224 5 0         irs nres              ;fix reassembly count
024537 073575 5 0         ldx rstptr            ;restore x to orig
024540 103532 5 0         jmp rstpk2 i

                        .lev t.o
                        .lck all
;skip return if imp is dead or reset
024541 000000 5 0 rstchk: 0                     ;chk an imp no for reset (pre
024542 023572 5 0         cas rstimp            ;is this the one we should re
024543 100000 5 0          skp                  ;no
024544 003562 5 0           jmp rstch2          ;yes
024545 033600 5 0         stx rstcht            ;save x
024546 010000 5 0         sta 0
024547 144131 5 0         lda spfrti ix
024550 006043 5 0         ana [spfded+route]
024551 073600 5 0         ldx rstcht            ;restore x
024552 101400 5 0         smi 0&spfded          ;is this imp dead?
024553 103541 5 0          jmp rstchk i         ;no, just return
024554 004041 5 0         lda [cdestd]          ;yes, setup
024555 011576 5 0         sta rstcde            ;dead code
024556 140040 5 0         cra 0&cimpd           ;..and subcode
024557 011577 5 0 rstch1: sta rstsbc
024560 025541 5 0         irs rstchk            ;skip return= dead or reset
024561 103541 5 0         jmp rstchk i

024562 005604 5 0 rstch2: lda comblk
024563 013573 5 0         era rstblk
024564 100040 5 0         sze                   ;same blk ptr?
024565 103541 5 0          jmp rstchk i         ;no
024566 005763 5 0         lda [cinctr]          ;set up incomplete code
024567 011576 5 0         sta rstcde
024570 004011 5 0         lda three 0&clost     ;and subcode
024571 003557 5 0         jmp rstch1            ;for reset imp
```

```
                        .lev var
024572          V   rstimp: .block 1                ;dest imp to reset, -1= none

024573          V   rstblk: .block 1                ;blk ptr to notify
024574          V   rstcnt: .block 1                ;temp counter
024575          V   rstptr: .block 1                ;temp pointer
024576          V   rstcde: .block 1                ;ih code to send: dead or inc
024577          V   rstsbc: .block 1                ;imp-host subcode to send
024600          V   rstcht: .block 1                ;temp save x
024601          V   rstpqt: .block 1                ;ptr to queue end ptr
024602          V   rstihq: .block 1                ;temp save ihwq
024603          V   comhst: .block 1
024604          V   comblk: .block 1
024605          V   mbtfre: .block 1
024606          V   mbtcnt: .block 1

                        .lev t.o
024607 024000  5    mbtgt1: irs 0
024610 033604  5            stx comblk              ;set up blk ptr
024611 044000  5            lda mbimp x
024612 101040  5            snz
024613 003621  5             jmp mbtgt2             ;block not active
024614 021541  5            jst rstchk              ;active. to be reset?
024615 003632  5             jmp mbtgt4             ;no, check age
024616 140040  5            cra                     ;yes, so
024617 050000  5            sta mbimp x             ;  flush it
024620 050340  5            sta mbsta x             ;clear state word for recv bl
024621 025605  5    mbtgt2: irs mbtfre              ;count a free block
024622 100000  5            skp
024623 000000  5    mbtget:  0
                        .lck h2i
;** entrance
024624 000401  5 4          .enb t.o
024625 025606  5            irs mbtcnt
024626 003607  5             jmp mbtgt1             ;loop back
024627 140040  5            cra
024630 027605  5            ima mbtfre             ;get free blk cnt
024631 103623  5            jmp mbtget i

                        mbtgt4:
024632 001001  5            .inh h2i
024633 044250  5 4          lda mbtim x             ;get age
024634 006072  5 4          ana [mbage]
024635 025623  5 4          irs mbtget              ;skip for ages 0-15
024636 022072  5 4          cas [15.]
024637 101000  5 4           nop
024640 103623  5 4           jmp mbtget i           ;age=15 (also >[!!]), always
024641 023764  5 4          cas [11.]               ;get respective timer
024642 105765  5 4           lda [tim1mn] i         ;ages 12-14, time= 1 min
024643 101000  5 4           nop                    ;   (timers are minus, will f
024644 022011  5 4          cas three
024645 105766  5 4           lda [tim2to] i         ;ages 4-11, time= 2 ticks (1
024646 101000  5 4           nop
024647 101400  5 4          smi                     ;got a timer yet?
024650 025623  5 4           irs mbtget             ;no, age<4... double skip
024651 101400  5 4          smi
024652 105741  5 4           lda [tim1to] i         ;and time= 1 tick
024653 013572  5 4          era rstimp              ;-1 is active state, **if t.
024654 103623  5 4          jmp mbtget i            ;so ac=0 on return  ==>  acti
```

```
                .lev con
;slow t.o jobs dispatch, followed immediately by fast t.o dispatch

024655 031314 C   disps:  bufchk                  ;check buffer management
024656 036120 C           spfclk                  ;tick ages of spf data
024657 023000 C           jobs1                   ;slow t.o clean-up
02466C 024000 C   dispf:  rstout                  ;must precede imtc
024661 023102 C           imtc                    ;check for m2i tasks to do
024662 023070 C           hitc                    ;run down H2I allocate timers
024663 033256 C           smooth                  ;average delays, must precede
024664 034362 C           ruptic                  ;tick spf retrans timers, sen
024665 034105 C           rupgen                  ;send spf update if necessary
024666 026640 C           swchs                   ;calls swch; monitor things
       024667     dispn=.

024667 177771 C   dspcnt: dispf-dispn             ;fast loop counter init value
02467C 177766 C           disps-dispn             ;slow loop counter init value

024671 177771 C   tikcnt: clockf-clockn           ;fast loop clock counter init
024672 177762 C           clocks-clockn           ;slow loop clock counter init

024673 177777 C   rstcls: -1 0&clocks            ;reset values for slow clocks
024674 177776 C           -2
024675 177773 C           -5
024676 177770 C           -8.                     ;5 seconds in slow ticks
024677 177764 C           -12.                    ;8 seconds in slow ticks
02470C 177747 C           -25.
024701 177640 C           -96.
024702 177406 C   rstclf: -250.                   ;reset values for fast clocks
024703 177603 C   rst125: -125.                  ;note: wattic changes if thes
024704 177716 C   rst050: -50.
024705 177747 C   rstslo: -25.
024706 177766 C   rst2md: -10.
024707 177773 C   rstmed: -5
02471C 177777 C   rstfst: -1 0&nfclks
       024711     rstcln=.

                .lev var
024711         V  todisp: .block 1                ;dispatch loop cntr (set from
024712         V  toslow: .block 1                ;slow tick counter
```

```
                .stitl timeout: finish up slow jobs
                .lev t.o
                .section pg23
023000 105551 5 jobs1:  lda [mgon] i            ;tick msg gen dead man
023001 100400 5         spl                     ;...switch if mgon neg.
023002 125551 5          irs [mgon] i
023003 004160 5         lda tsklck              ;has TASK locked us out?
023004 100040 5         sze
023005 003014 5          jmp jobs1a             ;if so, skip RESETT/RESETR ca
023006 004014 5         lda minus1              ;dead imp clean up
023007 073552 5         ldx [tmblkt]
023010 121553 5         jst [resett] i
023011 004014 5         lda minus1
023012 073554 5         ldx [rmblkt]
023013 121555 5         jst [resetr] i
023014 121556 5 jobs1a: jst [juqc] i            ;adjust queue counters

;validate the add chain. Since the first "MAXPL" entries are ADD MAXPL+HEDR-1 X
;instructions (all the way to ADD 1+HEDR-1 X), we check that instruction
;sequence, then check the return jump.

023015 073557 5         ldx [-maxpl]
023016 005560 5         lda [add maxpl+hedr-1 x]
023017 153561 5 addsml: era [addtop+maxpl] ix   ;this word a match?
023020 100040 5         sze
023021 000000 5          %crash                 ;add chain broken
023022 153561 5         era [addtop+maxpl] ix   ;fix ac
023023 016052 5         sub [1]
023024 024000 5         irs 0
023025 003017 5          jmp addsml

023026 105561 5         lda [addbot] i          ;get return jump
023027 013562 5         era [ (jmp chksum i) ]  ;check it
023030 100040 5         sze
023031 000000 5          %crash                 ;add chain return jump broken


023032 024152 5         irs times               ;count time in slow ticks
023033 025047 5         irs wdtime              ;check software w.d.t.
023034 003036 5          jmp jobstr             ;now check t.r. bursts.
023035 000000 5         %crash                  ;software wdt fired

023036 005052 5 jobstr: lda trclok              ;time to count down tr burst?
023037 141206 5         aoa
023040 100040 5         sze
023041 102124 5          jmp toret i            ;return if no
023042 105563 5         lda [trcnts] i
023043 016052 5         sub [1]                 ;decrement t.r. count
023044 101400 5         smi                     ;past zero?
023045 111563 5          sta [trcnts] i         ;no
023046 102124 5         jmp toret i
```

```
                        .lev var
023047          V       wdtime: .block 1

;fast and slow clocks
        000007          nfclks=7                        ; # fast clocks
                        clocks:
023050          V       tim1to: .block 1                ;1 slow tick
023051          V       tim2to: .block 1                ;2 slow ticks
023052          V       trclok: .block 1                ;3 sec
023053          V       tim5s:  .block 1                ;5 sec
023054          V       tim8s:  .block 1                ;8 sec
023055          V       tim15s: .block 1                ;15 sec
023056          V       tim1mn: .block 1                ;1 minute
023057          V       clockf: .block nfclks
        023066          clockn=.

                        .lev t.o
023066 000000 5         jstt.o: 0                       ;dummy t.o subr
023067 103066 5                 jmp jstt.o i
```

```
;Run down the allocate counters for hosts

                    .lev t.o
023070 073564 5   hitc:   ldx [-th]
023071 001001 5   hitc1:  .inh h2i
023072 145565 5 4         lda [hial+th] ix        ;get allocate counter
023073 100400 5 4         spl                     ;it running?
023074 014062 5 4          add [400]              ;yes, tick it
023075 151565 5 4         sta [hial+th] ix
023076 000401 5 4         .enb t.o
023077 024000 5           irs 0                   ;step to next
023100 003071 5            jmp hitc1
023101 102124 5           jmp toret i             ;done

;Watch for modem retransmissions, etc.

023102 005566 5   imtc:   lda [-ch]               ;start with modem 0
023103 010177 5           sta tot
023104 072177 5   imtc1:  ldx tot                 ;set up loop index
023105 145567 5           lda [mblock+ch] ix      ;does this modem exist?
023106 101040 5           snz
023107 003177 5            jmp i2mtc4             ;if not, skip all this
023110 010000 5           sta 0
023111 011202 5           sta imtmbl              ;save modem block pointer
023112 044117 5           lda line x              ;line silent?
023113 100400 5           spl
023114 003145 5            jmp torup              ;yes, skip retrans. check
023115 044202 5           lda retflg x            ;has i2m serviced last retran
023116 100040 5           sze                     ;..request
023117 003145 5            jmp torup              ;no, check for rup's
023120 044125 5           lda retrqi x            ;pointer to retrans. queue he
023121 010000 5           sta 0
023122 044000 5           lda q.forw x            ;compare first buffer pointer
023123 010001 5           sta ireg                ;to queue header pointer
023124 012000 5           era 0
023125 073202 5           ldx imtmbl              ;restore modem block pointer
023126 101040 5           snz                     ;does forward ptr point to he
023127 003145 5            jmp torup              ;yes, queue empty
```

```
023130 044136 5   toret0: lda i2mrtt x          ;get retrans. time
023131 010002 5          sta jreg               ;save it
023132 072001 5          ldx ireg               ;buffr addr
023133 000010 5          %rdclok
023134 056121 5          sub itst x             ;time to retransmit ?
023135 100400 5          spl
023136 140407 5           tca
023137 016002 5          sub jreg
023140 073202 5          ldx imtmbl             ;restore x=modem block
023141 100400 5          spl
023142 003145 5           jmp torup             ;no, check for rup's
023143 004001 5          lda ireg               ;get buffer pointer
023144 050202 5          sta retflg x           ;set retransmit flg for i2m
023145 140040 5   torup:  cra
023146 066201 5          ima rup4to x           ;any rups to send
023147 100040 5          sze                    ;no ,check demand core
023150 050204 5           sta sndrup x          ;yes ,set send rup flg for i2
023151 054115 5          add slt x              ;any demand core packets
023152 054202 5          add retflg x           ;or retransmissions
023153 054141 5          add i2mrem x           ;or modems to reset
023154 054160 5          add zsend x            ;any line up/down's
023155 001001 5          .inh all
023156 066143 5 0        ima i2mack x
023157 100040 5 0        sze                    ;is buffer acked?
023160 140600 5 0         scb                   ;yes,
023161 066143 5 0        ima i2mack x
023162 141216 5 0        aca                    ;1=acked
023163 101040 5 0        snz                    ;anything for i2m?
023164 003176 5 0         jmp i2mtc3            ;no check next mod
023165 050140 5 0        sta i2mrar x           ;yes, set high pri. flg
023166 044137 5 0        lda i2mbsy x           ;is I2M willing to take a pok
023167 100040 5 0        sze
023170 003176 5 0         jmp i2mtc3            ;if not, skip poke, leave I2M
023171 044025 5 0        lda pcb.num x          ;get modem number
023172 010000 5 0        sta 0
023173 144142 5 0        lda i2mpci ix          ;get ptr to I2M's PCB
023174 010000 5 0        sta 0                  ;into X for GPR
023175 000043 5 0        gpr
023176 000401 5 0 i2mtc3: .enb t.o
023177 024177 5   i2mtc4: irs tot
023200 003104 5           jmp imtcl
023201 102124 5          jmp toret i

                         .lev var
023202         V  imtmbl: .block 1               ;holds modem block pointer
```

```
                .stitl FAKE HOSTS
                .INCLUDE fak.m4
;fakehosts

;Discard (DSUCK) is a mess.



                .section pg25
                .lev bck

;This is the DDT Fake Host (FH1).  Its "suck" side simply supplies characters
;through the I2DDT mailbox word.  Its "jam" side does output directly to the
;host code and also processes the DDT commands, etc.

025000 021172 9   ddtnew: jst ddtnl
                  ddjini:
025001 021074 9   ddtlop: jst ddtnin          ;get a number, set up DDTNFL
025002 013337 9           era [ 57 ]          ;a slash?
025003 101040 9           snz
025004 003025 9            jmp ddts
025005 013340 9           era [ 57 ? 10 ]     ;a backspace?
025006 101040 9           snz
025007 003043 9            jmp ddtu
025010 012012 9           era [ 10 ? 15 ]     ;a carriage return?
025011 101040 9           snz
025012 003057 9            jmp ddtcr
025013 012013 9           era [ 15 ? 12 ]     ;a line feed?
025014 101040 9           snz
025015 003050 9            jmp ddtlf

025016 004005 9   ddterr: lda [ 77 ]          ;here on all errors, type a ?
025017 121341 9           jst [ddtcho] i
025020 003000 9           jmp ddtnew          ;new line, etc.

025021        9   ddtval: .block 1
025022        9   ddtadr: .block 1
025023        9   ddtnum: .block 1
025024        9   ddtnfl: .block 1
```

```
;Here on slash.

025025 021067 9   ddts:   jst ddtarg          ;make a good arg
025026 011022 9           sta ddtadr
025027 003035 9           jmp ddtop2

025030 011022 9   ddtsta: sta ddtadr          ;set up open addr
025031 005022 9   ddtopn: lda ddtadr          ;get the address
025032 021143 9           jst ddtnou          ;output it
025033 005337 9           lda [ 57 ]
025034 121341 9           jst [ddtcho] i      ;a slash
025035 021200 9   ddtop2: jst ddtso           ;a space
025036 105022 9           lda ddtadr i        ;the contents
025037 011021 9           sta ddtval          ;remember it
025040 021143 9           jst ddtnou
025041 021200 9           jst ddtso           ;a space
025042 003001 9           jmp ddtlop

;Here on back-space, open previous

025043 021061 9   ddtu:   jst ddtclo          ;close this loc
025044 021172 9           jst ddtnl
025045 005022 9           lda ddtadr          ;less one loc
025046 016052 9           sub [1]
025047 003030 9           jmp ddtsta          ;open it

;Here on line-feed, open next

025050 021061 9   ddtlf:  jst ddtclo
025051 025022 9           irs ddtadr          ;next location
025052 004027 9           lda [ 15 ]          ;get a carriage return
025053 121341 9           jst [ddtcho] i
025054 140040 9           cra                 ;and a padding of nul
025055 121341 9           jst [ddtcho] i
025056 003031 9           jmp ddtopn

;Here on carriage return

025057 021061 9   ddtcr:  jst ddtclo
025060 003000 9           jmp ddtnew          ;new line

;Subroutine to close off a location

025061 000000 9   ddtclo: 0
025062 025024 9           irs ddtnfl          ;anything typed?
025063 103061 9            jmp ddtclo i       ;no, just return
025064 005023 9           lda ddtnum          ;yes, get number
025065 111022 9           sta ddtadr i        ;deposit it
025066 103061 9           jmp ddtclo i        ;and return

;Subroutine to find an arg if one exists

025067 000000 9   ddtarg: 0
025070 005023 9           lda ddtnum          ;assume the real arg
025071 025024 9           irs ddtnfl          ;anything given
025072 005021 9            lda ddtval         ;nothing there, use last type
025073 103067 9           jmp ddtarg i        ;return
```

```
;Subroutine to get a "number". Dot equals the current location. Can do
;Plus N, too.

025074 000000 9   ddtnin: 0
025075 140040 9           cra                   ;set flag and value to nil
025076 011023 9           sta ddtnum
025077 011024 9           sta ddtnfl

025100 021111 9   ddtnpl: jst ddtni0            ;here on plus, get it
025101 027023 9           ima ddtnum            ;save char, get accumulated
025102 015142 9           add ddtnm0
025103 027023 9           ima ddtnum            ;save accumulated num, read b

025104 013342 9   ddtniw: era [ 53 ]            ;another plus?
025105 101040 9           snz
025106 003100 9            jmp ddtnpl           ;yes
025107 013342 9           era [ 53 ]
025110 103074 9           jmp ddtnin i          ;return

;Here to get a single term. Sets DDTNFL if gets something. Returns with value
;in DDTNM0 and AC=terminating char.

025111 000000 9   ddtni0: 0
025112 140040 9           cra
025113 011142 9           sta ddtnm0

025114 021166 9   ddtni1: jst ddtcin            ;get a character, with echo
025115 011164 9           sta ddttmp            ;save it
025116 017343 9           sub [ 56 ]            ;a dot?
025117 101040 9           snz
025120 003133 9            jmp ddtdot           ;yes
025121 016053 9           sub [ 60-56 ]         ;comvert to a number
025122 100400 9           spl
025123 003140 9            jmp ddtnie           ;less than ASCII 0, punt
025124 022055 9           cas [ 70-60 ]
025125 003140 9            jmp ddtnie           ;gt 8, punt
025126 003140 9            jmp ddtnie           ;eq 8, punt

025127 027142 9           ima ddtnm0            ;all's well, get accumulated
025130 041475 9           lgl 3                 ;make room for 3 bits
025131 015142 9           add ddtnm0
025132 003134 9           jmp ddtni2

025133 005022 9   ddtdot: lda ddtadr            ;get current addre
025134 011142 9   ddtni2: sta ddtnm0            ;update number
025135 004014 9           lda [-1]              ;set number flag
025136 011024 9           sta ddtnfl
025137 003114 9           jmp ddtni1            ;and loop

025140 005164 9   ddtnie: lda ddttmp            ;get saved char
025141 103111 9           jmp ddtni0 i          ;and return

025142         9  ddtnm0: .block 1

;Subroutine to output a number from A.

025143 000000 9   ddtnou: 0
025144 041677 9           alr 1                 ;put sign in LSB, shyft left
025145 011164 9           sta ddttmp            ;save it
```

```
025146 006052 9            ana [ 1 ]              ;only LSB
025147 015344 9            add [ 60 ]             ;convert to ASCII
025150 121341 9            jst [ddtcho] i

025151 004020 9            lda [ -5 ]
025152 011165 9            sta ddttm2

025153 005164 9   ddtno1:  lda ddttmp             ;get current value
025154 041675 9            alr 3                  ;put MS digit into LS digit
025155 011164 9            sta ddttmp             ;save new value
025156 006013 9            ana [ 7 ]
025157 015344 9            add [ 60 ]             ;to ASCII
025160 121341 9            jst [ddtcho] i
025161 025165 9            irs ddttm2             ;more?
025162 003153 9             jmp ddtno1            ;yes

025163 103143 9            jmp ddtnou i           ;no, return

025164        9   ddttmp:  .block 1
025165        9   ddttm2:  .block 1
```

```
;Subroutine to filter incoming characters.

025166 000000 9   ddtcin: 0
025167 121345 9           jst [ddtchi] i          ;get it
02517C 007346 9           ana [ 177 ]             ;only low 7 bits
025171 103166 9           jmp ddtcin i            ;return to caller

;Subroutine to output a CRLF

025172 000000 9   ddtnl:  0
025173 004027 9           lda [ 15 ]
025174 121341 9           jst [ddtcho] i
025175 004025 9           lda [ 12 ]
025176 121341 9           jst [ddtcho] i
025177 103172 9           jmp ddtnl i

;Subroutine to output 2 spaces

025200 000000 9   ddtso:  0
025201 004057 9           lda [ 40 ]
025202 121341 9           jst [ddtcho] i
025203 004057 9           lda [ 40 ]
025204 121341 9           jst [ddtcho] i
025205 103200 9           jmp ddtso i
```

```
;Subroutine to get a character from the net (via memory window called I2DDT)

025206 000000 9   ddtchi: 0
025207 072052 9           ldx [fhpddt-nh]          ;get our fake host number
025210 100000 9           skp                      ;first time though, don't deb
025211 120111 9   ddtciw:  jst doze i              ;doze for a while waiting for
025212 140040 9           cra
025213 026162 9           ima i2ddt                ;get character from mailbox,
025214 101040 9           snz
025215 003211 9            jmp ddtciw              ;nothing there, try later
025216 103206 9           jmp ddtchi i             ;else return

;Subroutine to put a character into the net directly.

025217 000000 9   ddtcho: 0
025220 033250 9           stx ddtcox               ;save X
025221 011247 9           sta ddtcot               ;save the character in a temp
025222 072052 9           ldx [fhpddt-nh]          ;get our host number
025223 004074 9           lda [ldrnlf]             ;JAM in the leader
025224 120110 9           jst jam i
025225 005331 9           lda ddtsrc               ;get destinatio flags
025226 141044 9           car 0&ldrmty
025227 120110 9           jst jam i
025230 005332 9           lda ddtsrc+1             ;get handling type/host
025231 120110 9           jst jam i
025232 005333 9           lda ddtsrc+2             ;get dest imp
025233 120110 9           jst jam i
025234 005334 9           lda ddtsrc+3             ;get dest link
025235 120110 9           jst jam i
025236 140040 9           cra                      ;length
025237 120110 9           jst jam i
025240 005247 9           lda ddtcot               ;get character back
025241 120110 9           jst jam i                ;send it
025242 073347 9           ldx [100000+fhpddt-nh]   ;set EOM
025243 004071 9           lda [100000]             ;plus padding
025244 120110 9           jst jam i
025245 073250 9           ldx ddtcox               ;restore X
025246 103217 9           jmp ddtcho i

025247         9  ddtcot: .block 1                 ;temp, holds character
025250         9  ddtcox: .block 1                 ;temp, holds X register
```

```
;This is the complete SUCK side of FH1 (DDT). All we do is try to keep the
;window (I2DDT) full and the leader area (DDTSRC) up to date.

025251 001001 9   ddsini: .inh all                    ;start here initially
025252 145350 9 0         lda [i2hpct+nh] ix         ;get i2h pcb
025253 010000 9 0         sta 0
025254 044044 9 0         lda i2hpcb.swcsr x         ;declare host ready line up
025255 007351 9 0         ana [-1 ? %c18csr.down ? %c18csr.flapped ]
025256 050044 9 0         sta i2hpcb.swcsr x         ;clear hdown and flapped bits
025257 072052 9 0         ldx [fhpddt-nh]            ;restore host number
02526C 000401 9 0         .enb bck
025261 120112 9   ddstop: jst suck i                 ;get leader
025262 120112 9           jst suck i
025263 011331 9            sta ddtsrc
025264 120112 9           jst suck i
025265 011332 9            sta ddtsrc+1
025266 120112 9           jst suck i
025267 011333 9            sta ddtsrc+2
02527C 120112 9           jst suck i
025271 100000 9            skp
025272 003261 9             jmp ddstop               ;leader only, try for another
025273 011334 9           sta ddtsrc+3
025274 120112 9           jst suck i
025275 100000 9            skp
025276 003261 9             jmp ddstop               ;chew up length
025277 120112 9   dds1:   jst suck i
02530C 003310 9            jmp dds2
025301 011330 9           sta ddsw
025302 141050 9           cal
025303 101040 9           snz
025304 003261 9            jmp ddstop               ;nothing in this word
025305 005330 9           lda ddsw
025306 021316 9           jst ddsgiv                 ;give this to DDT
025307 003261 9           jmp ddstop                 ;and continue

02531C 011330 9   dds2:   sta ddsw
025311 021316 9           jst ddsgiv
025312 005330 9           lda ddsw
025313 141340 9           ica
025314 021316 9           jst ddsgiv
025315 003277 9     .     jmp dds1

025316 000000 9   ddsgiv: 0
025317 141140 9           icl
02532C 026162 9           ima i2ddt                  ;give it to DDT
025321 100040 9           sze
025322 000000 9            %crash                   ;DDSGIV: mailbox overrun
025323 120113 9   ddsgw:  jst wait i                  ;give DDT a chance
025324 004162 9           lda i2ddt                  ;get mailbox again
025325 100040 9           sze
025326 003323 9            jmp ddsgw                 ;if still has char, wait
025327 103316 9           jmp ddsgiv i               ;else return

02533C        9   ddsw:   .block 1
025331        9   ddtsrc: .block 6                   ;DDT's leader area
```

```
                        .stitl fake host 3
                        .section pg26
                        .lev bck
;statistics programs

                        stjini:                         ;for INI
026000 140040 9         bkst:   cra                     ;set up ncc host
026001 011715 9                 sta statl2+tpon-snon    ;for trouble report
026002 011714 9                 sta statl2+diagon-snon  ;and diag leaders
026003 105735 9                 lda [noc.nochst] i
026004 011707 9                 sta statl3+tpon-snon 0&ncch
026005 011706 9                 sta statl3+diagon-snon 0&ncch
026006 105736 9                 lda [noc.nocimp] i
026007 011701 9                 sta statl4+tpon-snon
026010 011700 9                 sta statl4+diagon-snon
026011 105737 9                 lda [noc.noclnk] i
026012 011672 9                 sta statl5+diagon-snon
026013 011673 9                 sta statl5+tpon-snon
026014 004200 9                 lda diagq               ;turn on diag according to wh
026015 011663 9                 sta diagon              ;  there is anything to go
026016 140040 9                 cra
026017 026154 9                 ima ludflg              ;is line up/down flg on?
026020 010155 9                 sta sp306               ;if so send special 306 msg
026021 101040 9                 snz                     ;..but not 307
026022 003030 9                  jmp nolud
026023 072014 9                 ldx minus1              ;send special 306 (hack for n
026024 021123 9                 jst givldr              ;send leader
026025 145740 9                 lda [cawl+5] ix
026026 010001 9                 sta ireg
026027 102001 9                 jmp ireg i              ;go send special 306

026030 072020 9         nolud:  ldx minus5
026031 033647 9                 stx sp8
026032 073647 9         sp3:    ldx sp8                 ;loop here for each stat proc
026033 045665 9                 lda snon+5 x
026034 101040 9                 snz                     ;is this stat program turned
026035 003147 9                  jmp sp91               ;no
026036 004153 9                 lda sync                ;pick up current time
026037 057647 9                 sub olds+5 x            ;compare with last time this
026040 100400 9                 spl                     ;correct for timer overflow
026041 140407 9                  tca                    ;every 30 minutes
026042 057724 9                 sub statf+5 x           ;look at interval between pro
026043 100400 9                 spl                     ;is it time to activate this
026044 003147 9                  jmp sp91               ;no
026045 140040 9                 cra                     ;yes, init a time-skewing tem
026046 043054 9                 jmp statx+5 x
                        .lev con
026047 003111 C         statx:  jmp stat6               ;sync - snapshot
026050 003074 C                 jmp stat3               ;skew - cumstat
026051 003111 C                 jmp stat6               ;sync - message gen
026052 003111 C                 jmp stat6               ;sync - diag
026053 003074 C                 jmp stat3               ;skew - trbl rept
```

```
                    .lev bck
;message generator
 026054 072011 9    genm:    ldx three 0&fhpsts      ;fake host 3 - statistics
 026055 005724 9             lda mgnl                ;pick up length of message
 026056 006033 9             ana [777]
 026057 140401 9             cma                     ;gives right number, excludin
 02606C 011073 9             sta mgcnt
 026061 004151 9             lda time
 026062 011734 9             sta mgtime              ;for measurements
 026063 005733 9             lda mgno                ;*send mess gen #
 026064 025733 9             irs mgno
 026065 003070 9              jmp genm1
 026066 003070 9             jmp genm1
 026067 120110 9    genm1:   jst jam i               ;*send mgnl words or pkts ??
 02607C 025073 9    genm1:   irs mgcnt
 026071 003067 9              jmp genml
 026072 003146 9             jmp givlst              ;*send padding and end messag

                    .lev var
 026073         V   mgcnt:   .block 1                ;temp counter for mess length
```

```
                        .lev bck
026074 045724 9   stat3:  lda statf+5 x          ;yes, skew reports by imp num
026075 040572 9           ars 6                  ;div by 100
026076 101040 9           snz
026077 003111 9            jmp stat6             ;too small an interval to ske
026100 011651 9           sta count1             ;build multiply loop
026101 004106 9           lda mine
026102 006005 9           ana [77]               ;skew on low bits only
026103 140407 9           tca
026104 011652 9           sta count2
026105 140040 9           cra
026106 015651 9   stat5:  add count1             ;count1 times count2
026107 025652 9           irs count2
026110 003106 9            jmp stat5
026111 011650 9   stat6:  sta skewt              ;is skewt = (mine/100)*statf

026112 045724 9           lda statf+5 x
026113 140407 9           tca
026114 006153 9           ana sync
026115 015650 9           add skewt
026116 051647 9           sta olds+5 x
026117 021123 9           jst givldr             ;*send stat leader
026120 145740 9           lda [cawl+5] ix        ;call stat program
026121 010001 9           sta ireg
026122 102001 9           jmp ireg i

026123 000000 9   givldr: 0
026124 004074 9           lda [ldrnlf]           ;new leader flag
026125 021237 9           jst give
026126 045716 9           lda stat12+5 x         ;flags/type
026127 021237 9           jst give
026130 045710 9           lda stat13+5 x         ;handling/host
026131 021237 9           jst give
026132 045702 9           lda stat14+5 x         ;imp
026133 021237 9           jst give
026134 045674 9           lda stat15+5 x         ;link/subtype
026135 021237 9           jst give
026136 140040 9           cra                    ;length
026137 021237 9           jst give
026140 103123 9           jmp givldr i
```

```
026141 000000 9   givpad: 0
026142 073741 9          ldx [100000+fhpsts-nh]
026143 004071 9          lda sign
026144 120110 9          jst jam i
026145 103141 9          jmp givpad i

026146 021141 9   givlst: jst givpad              ;*send padding and terminate
026147 025647 9   sp91:   irs sp8                 ;loop back over each stat pro
026150 003032 9            jmp sp3
026151 140040 9          cra                      ;test if anomaly occurrred
026152 027512 9          ima trbstf
026153 101040 9          snz
026154 003175 9           jmp sp93                ;if not, jump ahead
026155 140040 9          cra                      ;reinit t.r. leader for anom

026156 011715 9          sta statl2+tpon-snon
026157 105735 9          lda [noc.nochst] i
026160 011707 9          sta statl3+tpon-snon
026161 105736 9          lda [noc.nocimp] i
026162 011701 9          sta statl4+tpon-snon
026163 004011 9          lda [lcunct]             ; use raw for 310's
026164 113737 9          era [noc.noclnk] i
026165 011673 9          sta statl5+tpon-snon
026166 072014 9          ldx [-1]                 ;x=-1 so givldr will use t.r.
026167 021123 9          jst givldr
026170 103742 9          jmp [anom] i             ;ready to do anom report

                                                  ;(anom returns here)
026171 027517 9   sp92:   ima ntrcks               ;this replaces conclu,givlst
026172 140407 9          tca
026173 021237 9          jst give
026174 021141 9          jst givpad
026175 072011 9   sp93:   ldx three                ;for tpon nonzero; also as do
026176 033664 9          stx tpon                 ;guarantee trouble reports on
026177 120111 9          jst doze i
026200 003000 9          jmp bkst
```

```
                    .lev bck
026201 140040 9     diag:   cra                     ;send broken pkts to ncc for
026202 001001 9             .inh all
026203 126200 9 0           ima diagq i
026204 026200 9 0           ima diagq               ;get pkt off diag queue
026205 000401 9 0           .enb bck
026206 011653 9             sta diagp               ;save pntr
026207 004011 9             lda three               ;*send diag code
026210 021237 9             jst give
026211 004163 9             lda hltloc              ;*send hlt pc
026212 021237 9             jst give
026213 004164 9             lda hlta                ;*send hlt a reg
026214 021237 9             jst give
026215 004165 9             lda hltx                ;*send hlt x reg
026216 021237 9             jst give
026217 005653 9             lda diagp               ;*send pkt pntr
026220 010000 9             sta 0
026221 021237 9             jst give
026222 005743 9             lda [-bufl]
026223 011654 9             sta diagc               ;set up send counter
026224 044000 9     diag1:  lda 0 x                 ;*send  bad packet
026225 021237 9             jst give
026226 024000 9             irs 0
026227 025654 9             irs diagc
026230 003224 9              jmp diag1
026231 073653 9             ldx diagp
026232 001001 9             .inh all
026233 120115 9 0           jst flushi i
026234 024224 9 0           irs nres                ;give back to reassembly
026235 000401 9 0           .enb bck
026236 003146 9             jmp givlst


026237 000000 9     give:   0                       ;stat calls to jam
026240 033655 9             stx stt8                ;save x-reg
026241 072011 9             ldx [fhpsts-nh]
026242 120110 9             jst jam i
026243 073655 9             ldx stt8
026244 103237 9             jmp give i
```

```
                .stitl status and traffic reports
                .lev bck
;trouble reports to ncc at host 0 at bbn
026245 121744 9   trbl:   jst [octchk] i          ;check for octet mismatches
026246 005745 9           lda [306]               ;*send trbl rept code
026247 021474 9           jst stlead              ;* sends system parameters
026250 121746 9           jst [paklup] i          ;*send packages state
026251 021522 9           jst ntgive
026252 140040 9           cra                     ;*send restart & reload info

026253 021522 9           jst ntgive
026254 004105 9           lda vers                ;*send imp version no
026255 021522 9           jst ntgive
026256 140040 9           cra                     ;spare, was tipver
026257 021522 9           jst ntgive
026260 004014 9           lda [ -1 ]              ;spare, was host number under
026261 021522 9           jst ntgive
026262 140040 9           cra                     ;spare, was host test sent
026263 021522 9           jst ntgive
026264 140040 9           cra                     ;spare, was host test receive
026265 021522 9           jst ntgive
026266 105747 9           lda [crshp] i           ;crash PC
026267 021522 9           jst ntgive
026270 105750 9           lda [crsha] i           ;crash A
026271 021522 9           jst ntgive
026272 105751 9           lda [crshcd] i          ;crash code
026273 021522 9           jst ntgive
026274 121752 9           jst [bufs] i            ;*sends buffer and anomaly in
026275 073753 9           ldx [-ch]               ;take a snapshot of line erro
026276 033513 9   ntr1:   stx ntrtmp
026277 145754 9           lda [mblock+ch] ix      ;is this an imp modem?
026300 100040 9           sze
026301 003304 9            jmp ntr2               ;yes
026302 021470 9           jst givzro              ;no, send two zeros
026303 003345 9           jmp ntr3
026304 010000 9   ntr2:   sta 0
026305 140040 9           cra
026306 001001 9           .inh (t.o,tsk)
026307 066177 9 5         ima rtrcvd x
026310 056200 9 5         sub rtssnt x            ;subtract z-packets sent coun
026311 140407 9 5         tca                     ;compute no of z-packets miss
026312 100400 9 5         spl
026313 140040 9 5          cra                    ;must be a positive number
026314 022007 9 5         cas [377]
026315 004007 9 5          lda [377]
026316 101000 9 5          nop
026317 011520 9 5         sta e321                ;save as error count for this
026320 004014 9 5         lda minus1
026321 066200 9 5         ima rtssnt x            ;send no of routing mess sent
026322 022007 9 5         cas [377]
026323 004007 9 5          lda [377]
026324 101000 9 5          nop
026325 100400 9 5         spl                     ;was rtssnt still -1?
026326 140040 9 5          cra                    ;yes, make it zero
026327 011514 9 5         sta ntrtm1
026330 044026 9 5         lda m2ipcb.speed x      ;get line speed bits
026331 041462 9 5         lgl 14.                 ;put in 1&2 bit positions
026332 006037 9 5         ana [140000]            ;clear remainder of rmflag
026333 013514 9 5         era ntrtm1
```

```
                .ret bck
;*send # routing mess sent & line speed
026335 044117 9          lda line x          ;now error+thruput counts for
026336 100040 9          sze
026337 004061 9           lda [200]
026340 052120 9          era lnei x          ;pick up neighbor imp number

026341 141340 9          ica
026342 013520 9          era e321            ;*send no of routing mess mis
026343 021522 9          jst ntgive
026344 073513 9          ldx ntrtmp
026345 024000 9  ntr3:   irs 0
026346 003276 9           jmp ntr1
026347 027517 9          ima ntrcks          ;a=0 now
026350 140407 9          tca
026351 021237 9          jst give            ;*send checksum
026352 021141 9          jst givpad          ;*padding and end of message

026353 140040 9          cra
026354 026155 9          ima sp306           ;is this special line up/down
026355 100040 9          sze
026356 003030 9           jmp nolud          ;yes, don't send 307 msg
; next, send 307 message every 8th time
026357 025521 9          irs thrupc          ; is now the 8th?
026360 003457 9           jmp ntr13          ; nope
026361 004022 9          lda [-8.]           ; yes, reset counter
026362 011521 9          sta thrupc
026363 072014 9          ldx minus1
026364 021123 9          jst givldr          ;*send leader
026365 005755 9          lda [307]           ;*stat message code
026366 021474 9          jst stlead          ;*send system parameters
026367 105756 9          lda [backx] i       ;*send # of back loops
026370 021522 9          jst ntgive
026371 140040 9          cra                 ;spare, was tpopen
026372 021522 9          jst ntgive
026373 073753 9          ldx [-ch]           ;send line stats
026374 033513 9  ntr8:   stx ntrtmp
026375 145754 9          lda [mblock+ch] ix  ;is this an imp modem
026376 100040 9          sze
026377 003402 9           jmp ntr6           ;yes
026400 021470 9          jst givzro          ;no, send two zeros
026401 003411 9          jmp ntr12
026402 010000 9  ntr6:   sta 0
026403 140040 9          cra
026404 066122 9          ima thrupt x
026405 021522 9          jst ntgive          ;*packet thruput
026406 066121 9          ima thrupw x
026407 021522 9          jst ntgive          ;*word thruput
026410 073513 9          ldx ntrtmp
026411 024000 9  ntr12:  irs 0
026412 003374 9           jmp ntr8
026413 072032 9          ldx [-nh]           ;now host throughputs for ea
026414 004022 9  ntr5:   lda [-8.] 0&ch      ;and for each of 8 flavors
026415 011514 9          sta ntrtm1
026416 005757 9          lda [ntrtab]
026417 011516 9          sta ntrtm2
026420 105516 9  ntr4:   lda ntrtm2 i
026421 010001 9          sta ireg
026422 140040 9          cra
```

```
026423 166001 9          ima ireg ix              ;*send host throughput count
026424 021522 9          jst ntgive
```

```
026425 025516 9           irs ntrtm2
026426 025514 9           irs ntrtm1
026427 003420 9            jmp ntr4
026430 033513 9           stx ntrtmp
026431 121760 9           jst [nt0000] i          ;check if endex is also imp m
026432 003435 9            jmp ntr10              ;yes, returns with A=mblock a
026433 021470 9           jst givzro              ;no, send 0's instead
026434 003451 9           jmp ntr11

026435 010000 9   ntr10:  sta 0                   ;put mblock address into X
026436 001001 9           .inh all
026437 140040 9 0         cra
026440 066133 9 0         ima i2mwt2 x            ;2nd modem word
026441 011515 9 0         sta nttemp              ;save locally
026442 140040 9 0         cra
026443 066132 9 0         ima i2mwtn x            ;1st modem word
026444 000401 9 0         .enb bck
026445 021522 9           jst ntgive              ;send 1st
026446 005515 9           lda nttemp              ;
026447 021522 9           jst ntgive              ;send 2nd
026450 073513 9           ldx ntrtmp
026451 024000 9   ntr11:  irs 0
026452 003414 9            jmp ntr5
026453 027517 9   conclu: ima ntrcks              ;a=0 now
026454 140407 9           tca
026455 021237 9           jst give                ;*send cksum
026456 021141 9           jst givpad              ;*send padding
026457 004007 9   ntr13:  lda [fhldis]            ;send dummy message to discar
026460 011707 9           sta statl3+tpon-snon
026461 004106 9           lda mine
026462 011701 9           sta statl4+tpon-snon
026463 140040 9           cra                     ;(make sure it's not raw)
026464 011673 9           sta statl5+tpon-snon
026465 072014 9           ldx minus1
026466 021123 9           jst givldr              ;*send leader
026467 003146 9           jmp givlst              ;*send padding and end messag

026470 000000 9   givzro: 0
026471 021522 9           jst ntgive              ;enter with A=0
026472 021522 9           jst ntgive
026473 103470 9           jmp givzro i
```

```
026474 000000 9   stlead: 0                      ;sends system parameters
026475 011517 9           sta ntrcks
026476 021237 9           jst give
026477 004151 9           lda time               ;*send time stamp
026500 021522 9           jst ntgive
026501 004056 9           lda [nh]               ;*send # real hosts
026502 021522 9           jst ntgive
026503 004054 9           lda [fh]               ;* send fake hosts
026504 021522 9           jst ntgive
026505 005761 9           lda [ch]               ;* send # phone lines
026506 021522 9           jst ntgive
026507 105762 9           lda [mxnimp] i         ;* send # of imps
026510 021522 9           jst ntgive
026511 103474 9           jmp stlead i
```

```
                        .lev var
026512                  trbstf: .block 1
026513                  ntrtmp: .block 1
026514                  ntrtm1: .block 1
026515                  nttemp: .block 1                ;modem wrd storage (for ncc)

026516                  ntrtm2: .block 1
026517                  ntrcks: .block 1                ;checksum for trouble rept
026520                  e321:   .block 1                ;no of errors on each line
026521                  thrupc: .block 1                ;8th counter for 307 msgs

                        .lev bck
026522 000000 9         ntgive: 0                       ;build checksum
026523 000401 9                 .enb bck
026524 027517 9                 ima ntrcks
026525 015517 9                 add ntrcks
026526 027517 9                 ima ntrcks
026527 021237 9                 jst give                ;and give a word to imp via
026530 140040 9                 cra
026531 103522 9                 jmp ntgive i
```

```
;bits in sws (trouble report anomalies):
; 100000 -  * unused
;  40000 -  * unused
;  20000 -  * unused
;  10000 - host data checksum bad
;   4000 - packet trace is on
;   2000 - m.generator is on
;   1000 - statistics is on
;    400 -  * unused
;    200 - trace is on
;    100 -  * unused
;     40 -  * unused
;     20 -  * unused (used to be override)
                .lev con
026532 003417 C   swstab: hacsum                ;10000
026533 026665 C           pton
026534 026662 C           mgon
026535 026661 C           son
026536 026660 C           snon
026537 026657 C           tron
02654C 026543 C           mprotf
026541 000006 C           zero
026542 000006 C           zero                  ;20
       000011     swstn=.-swstab
                .lev var
026543         V  mprotf: .block 1
                .lev t.o
026544 000000 5   swch:   0                     ;set up switches for trouble
026545 105763 5           lda [trcnts] i        ;count burst of t.r. in 30 se
026546 016025 5           sub [10.]
026547 101400 5           smi                   ;too big?
02655C 103544 5            jmp swch i           ;yes.. quit for now.
026551 004065 5           lda [4000]            ;set up freq for ncc trbl rep
026552 011723 5           sta statf+tpon-snon   ;dest is set in back
026553 072014 5           ldx minus1
026554 004163 5           lda hltloc
026555 100040 5           sze                   ;if hltloc is non-zero
026556 133764 5            stx [sws] i          ;fire off a trbl rept
```

```
026557 072017 5           ldx [-hstatsize]        ;number of hstat words
026560 032001 5           stx ireg
026561 072006 5           ldx [0]                 ;swchtmp counts hosts
026562 033656 5           stx swchtmp
026563 072017 5   swch0:  ldx [-4]                ;four hosts per hstat word
026564 032002 5           stx jreg
026565 073656 5           ldx swchtmp             ;get next host number
026566 140040 5           cra
026567 041674 5   swch2:  alr 4
026570 010177 5           sta tot
026571 145765 5           lda [host] ix           ;is host up?
026572 012052 5           era [hstup]
026573 100040 5           sze
026574 145766 5            lda [hdown] ix         ;yes, report 0
026575 101040 5           snz                     ;no, have a reason?
026576 145765 5            lda [host] ix          ;no,use hihd
026577 006072 5           ana [hstwhy]            ;yes, use it
026600 012177 5           era tot
026601 024000 5           irs 0                   ;go to next host number
026602 024002 5           irs jreg
026603 003567 5            jmp swch2
026604 033656 5           stx swchtmp             ;save host number
026605 072001 5           ldx ireg                ;ireg tells which hstat word

026606 167767 5           ima [hstat+hstatsize] ix
026607 153767 5           era [hstat+hstatsize] ix
026610 100040 5           sze
026611 133764 5            stx [sws] i            ;any negative number is OK fo
026612 024001 5           irs ireg                ;move to next set of 4 hosts

026613 003563 5            jmp swch0
;now look through swstab
026614 073770 5           ldx [-swstn]
026615 140040 5           cra
026616 010177 5   swch4:  sta tot
026617 045543 5           lda swstab+swstn x
026620 010001 5           sta ireg
026621 104001 5           lda ireg i
026622 100040 5           sze
026623 004071 5            lda sign               ;set bit if flag is nonzero
026624 012177 5           era tot
026625 041677 5           alr 1
026626 024000 5           irs 0
026627 003616 5            jmp swch4
026630 040664 5           arr 12.                 ;put bits back in place
026631 127764 5           ima [sws] i
026632 113764 5           era [sws] i
026633 101040 5           snz
026634 103544 5            jmp swch i
026635 011512 5           sta trbstf              ;tell trbl rpts that sws fire
026636 125763 5           irs [trcnts] i
026637 103544 5           jmp swch i

026640 021544 5   swchs:  jst swch
026641 102124 5           jmp toret i
                  .lev var
026642 000000 V   olds:    0
026643 000000 V            0
026644 000000 V            0
```

```
026645 000000 V              0
026646 000000 V              0
```

```
026647          V   sp8:     .block 1                ;counter for which stat prog
026650          V   skewt:   .block 1                ;amount of time to skew stat
026651          V   count1: .block 1
026652          V   count2: .block 1
026653          V   diagp:   .block 1                ;pkt pntr
026654          V   diagc:   .block 1                ;loop counter
026655          V   stt8:    .block 1                ;give temp x
026656          V   swchtmp:          .block 1       ;holds host number
```

```
                        .lev var
                                ;defplc(/parameters table)
                        paramt:
026657          V       tron:   .block 1                ;0-trace on
026660          V       snon:   .block 1                ;1- *unused
026661          V       son:    .block 1                ;2-10-sec stat on
026662          V       mgon:   .block 1                ;3-mess gen on
026663          V       diagon: .block 1                ;4-diag on
026664          V       tpon:   .block 1                ;5-trbl rept on
026665          V       pton:   .block 1                ;6-pkt trace on

026666          V       tlink:  .block 1                ;7-trace link
026667          V       stat15: .block 1                ;10- *unused
026670          V               .block 1                ;11-10-sec link
026671          V               .block 1                ;12-mess gen link
026672          V               .block 1                ;13-diag link
026673          V               .block 1                ;14-trbl rept link

026674          V       tdsti:  .block 1                ;15-trace dest imp
026675          V       stat14: .block 1                ;16- *unused
026676          V               .block 1                ;17-10-sec dest imp
026677          V               .block 1                ;20-mess gen dest imp
026700          V               .block 1                ;21-diag dest imp
026701          V               .block 1                ;22-trbl rept dest imp

026702          V       tdsth:  .block 1                ;23-trace dest hand/host
026703          V       stat13: .block 1                ;24- *unused
026704          V               .block 1                ;25-10-sec dest hand/host
026705          V               .block 1                ;26-mess gen dest hand/host
026706          V               .block 1                ;27-diag dest hand/host
026707          V               .block 1                ;30-trbl rept dest hand/host


026710          V       tdstf:  .block 1                ;31-trace dest flags/type
026711          V       stat12: .block 1                ;32- *unused
026712          V               .block 1                ;33-10-sec dest flags/type
026713          V               .block 1                ;34-mess gen dest flags/type

026714          V               .block 1                ;35-diag dest flags/type
026715          V               .block 1                ;36-trbl rept dest flags/type

026716          V       tf:     .block 1                ;37-auto trace freq
026717          V       statf:  .block 1                ;40- *unused
026720          V               .block 1                ;41-10-sec freq
026721          V               .block 1                ;42-mess gen freq
026722          V               .block 1                ;43-diag freq
026723          V               .block 1                ;44-trbl rept freq
```

```
026724          V   mgnl:   .block 1              ;45-mess gen length

026725          V   ptf:    .block 1              ;46-pkt trace freq

026726          V   rttunt: .block 1              ;47-round trip time units

026727          V   atdsti: .block 1              ;50-auto trace dest imp
026730          V   atdsth: .block 1              ;51-auto trace dest host
026731          V   atsrci: .block 1              ;52-auto trace source imp
026732          V   atsrch: .block 1              ;53-auto trace source host

                            ;defplc(/message generator number)
026733          V   mgno:   .block 1

        000055      paraml=.-paramt
026734          V   mgtime: .block 1              ;for measurements--must follo
```

```
                .section pg27
                .lev bck

;octchk (called by trbl) checks each line to make sure
;that a line that is up is using the number of octets
;it is configured for.  if a line is using less octets
;than it is configured for, that is because its neighbor
;is configured for less (which we discovered via the NETH
;word of line up/down packets).  the line is allowed to
;be up in this case, but send a trap to make sure the
;ncc doesn't ignore this.
027000 000000 9   octchk: 0
027001 073630 9           ldx [-ch]
027002 032002 9   octch1: stx jreg                ;use ntrtmp to loop thru all
027003 145631 9           lda [mblock+ch] ix      ;does this modem exist?
027004 101040 9           snz
027005 003026 9            jmp octch2             ;no
027006 010000 9           sta 0                   ;yes
027007 044117 9           lda line x              ;is line up?
027010 100040 9           sze
027011 003025 9            jmp octch3             ;no, go on to next line
027012 044107 9           lda cfoctet x           ;our config value
027013 052106 9           era maxoctet x          ;value being used
027014 101040 9           snz                     ;are they equal?
027015 003025 9            jmp octch3             ;yes, go on to next line
027016 044025 9           lda pcb.num x           ;get line number for trap
027017 010001 9           sta ireg
027020 044027 9           lda m2ipcb.nchan x      ;get number of channels in us
027021 001001 9           .inh all
027022 072001 9 0         ldx ireg
                                ;defhlt(/80. line up with too few channels)
027023 120120 9 0         jst hltjst i
027024 000401 9 0         .enb bck
027025 072002 9   octch3: ldx jreg
027026 024000 9   octch2: irs 0                   ;next line
027027 003002 9            jmp octch1
027030 103000 9           jmp octchk i

027031 005632 9   anom:   lda [310]               ;*stat message code
027032 121633 9           jst [stlead] i          ;*send system parameters
027033 021043 9           jst bufs                ;*send buffer info
027034 026163 9           ima hltloc              ;*send halt pc reg
027035 121634 9           jst [ntgive] i
027036 004164 9           lda hlta                ;*send halt a reg
027037 121634 9           jst [ntgive] i
027040 004165 9           lda hltx                ;*send halt x reg
027041 121634 9           jst [ntgive] i
027042 103635 9           jmp [sp92] i            ;checksumming and return

                .lev bck
027043 000000 9   bufs:   0
027044 105636 9           lda [sws] i             ;* send anomaly word, hosts &
027045 121634 9           jst [ntgive] i
027046 121634 9           jst [ntgive] i          ;not used
027047 072017 9           ldx minus4 0&counta&counts ;*send counts for free, re
027050 001001 9   ntr7:   .inh all
027051 044222 9 0         lda nfa+4+counta-counta x ;the counta/s is for the co
027052 056226 9 0         sub nfs+4+counts-counts x ;I know it's weird but so i
027053 121634 9 0         jst [ntgive] i
```

```
                        .ret bck
027054 024000 9                 irs 0
027055 003050 9                  jmp ntr7
027056 072017 9                 ldx [-hstatsize]
027057 145637 9         bufs1:  lda [hstat+hstatsize] ix ;*send host status
027060 121634 9                 jst [ntgive] i
027061 024000 9                 irs 0
027062 003057 9                  jmp bufs1
027063 103043 9                 jmp bufs i

027064 000000 9         paklup: 0                       ;loop over package hooks and

027065 072032 9                 ldx [-npaks]            ;flag present/absent states
027066 140040 9                 cra
027067 010102 9         pak11:  sta paks
027070 145640 9                 lda [bck00.hook+npaks] ix ;check a hook
027071 013641 9                 era [jstbck]            ;look for unhook
027072 100040 9                 sze
027073 004052 9                  lda one                ;bit means package present
027074 012102 9                 era paks
027075 040677 9                 arr 1
027076 024000 9                 irs 0
027077 003067 9                  jmp pak11              ;loop back
027100 041660 9                 alr npaks               ;put bits back
027101 010102 9                 sta paks                ;save for ddt inspection
027102 103064 9                 jmp paklup i


; skips if index is NOT configured imp modem
027103 000000 9         nt0000: 0
027104 105642 9                 lda [ntrtmp] i          ;contains index (-NH to -1)
027105 022015 9                 cas [-nh+ch]            ;in range -nh to -nh+ch-1 ?
027106 003112 9                  jmp nt0001             ;no, in range -nh+ch to 0
027107 003112 9                  jmp nt0001             ;ditto
027110 145643 9                 lda [mblock+NH] ix      ;in range, but is it config'd
027111 101040 9                 snz                     ;note the NH usage, for -NH 1
027112 025103 9         nt0001:  irs nt0000             ;no, so skip
027113 103103 9                 jmp nt0000 i            ;yes, A=mblock address

                        .lev con
027114 046636 C         ntrtab: htpmtn+nh
027115 046656 C                 htpmfn+nh
027116 046676 C                 ntpptn+nh
027117 046716 C                 htppfn+nh
027120 046736 C         ntrt1:  htpmtl+nh
027121 046756 C                 htpmfl+nh
027122 046776 C                 ntpptl+nh
027123 047016 C                 htppfl+nh
```

```
                    .lev (m2i,i2h,t.o,tsk)
027124 000000 1   hltwrd: 0
027125 026163 1           ima hltloc
027126 100040 1           sze                     ;previous hlt?
027127 003142 1            jmp hltwr1             ;yes, do not disturb
027130 004163 1           lda hltloc
027131 010164 1           sta hlta
027132 032165 1           stx hltx
027133 004015 1           lda minus2
027134 015124 1           add hltwrd
027135 010163 1           sta hltloc
027136 104163 1           lda hltloc i
027137 010163 1           sta hltloc              ;save loc of hlt
027140 004164 1           lda hlta
027141 103124 1           jmp hltwrd i

027142 026163 1   hltwr1: ima hltloc
027143 103124 1           jmp hltwrd i

;fake imp-to-host 3 - discard
                    .lev bck
027144 001001 9   dssini: .inh all                ;start here initially
027145 145644 9 0         lda [i2hpct+nh] ix      ;get i2h pcb pointer
027146 010000 9 0         sta 0
027147 044044 9 0         lda i2hpcb.swcsr x      ;declare host ready line up
027150 007645 9 0         ana [-1 ? %c18csr.down ? %c18csr.flap]
027151 050044 9 0         sta i2hpcb.swcsr x      ;clear hdown and flapped bits
027152 000401 9 0         .enb bck
027153 072011 9           ldx [fhpdis-nh]         ;restore host number
027154 003160 9           jmp stxy                ;and begin at stxy

027155 000000 9   dsuck:  0
027156 120112 9           jst suck i              ;next word
027157 103155 9            jmp dsuck i            ;not eom
027160 021155 9   stxy:   jst dsuck               ;start of new message
027161 021155 9           jst dsuck
027162 012012 9           era [5] 0&crfnm         ;a rfnm?
027163 100040 9           sze
027164 003167 9            jmp strep1             ;no
027165 005646 9           lda [-500.]             ;yes, reset software wdt to 5
027166 111647 9           sta [wdtime] i
027167 120112 9   strep1: jst suck i              ;eat up the message
027170 003167 9            jmp .-1                ;until EOM
027171 003160 9           jmp stxy
```

```
                    .stitl parameter change and packet core
;fake imp-to-host 2  -  parameter change +packet core
                    .section pg30
                    .lev bck
030000 001001 9   prsini: .inh all                  ;start here initially
030001 145534 9 0         lda [i2hpct+nh] ix        ;get i2h pcb pointer
030002 010000 9 0         sta 0
030003 044044 9 0         lda i2hpcb.swcsr x        ;declare host ready line up
030004 007535 9 0         ana [ -1 ? %c18csr.down ? %c18csr.flap ]
030005 050044 9 0         sta i2hpcb.swcsr x        ;clear hdown and flapped bits
030006 000401 9 0         .enb bck
030007 072053 9           ldx [fhppkc-nh]           ;restore host number
030010 003012 9           jmp best                  ;and begin at best

030011 021041 9   bestf:  jst bestfl                ;discard rest of message
030012 120112 9   best:   jst suck i                ;begin to accept next message
030013 120112 9            jst suck i
030014 120112 9             jst suck i
030015 011077 9              sta pkcldr             ;save source host
030016 120112 9           jst suck i
030017 011100 9            sta pkcld1               ;save source imp
030020 021046 9           jst isuck                 ;ignore if rfnm
030021 021046 9           jst isuck                 ;or too short
030022 011105 9           sta pkccct
030023 021046 9           jst isuck                 ;or if first data word is las
030024 100400 9           spl                       ;packet core?
030025 003106 9            jmp pkcmsg               ;yes
030026 003031 9           jmp bestp                 ;no, parameter number

030027 021046 9   bestl:  jst isuck                 ;get parameter number
030030 101400 9           smi                       ;in range
030031 023536 9   bestp:   cas [paraml-1]
030032 003011 9             jmp bestf               ;no, bad.
030033 101000 9             nop
030034 015537 9           add [paramt]              ;ptr to param tab
030035 011041 9           sta bestfl
030036 021046 9           jst isuck                 ;get param value
030037 111041 9           sta bestfl i              ;set parameter to new value
030040 003027 9           jmp bestl

030041 000000 9   bestfl: 0                         ;(also used as temp)
030042 072053 9           ldx [fhpprm-nh]           ;restore fh index
030043 120112 9           jst suck i                ;gobble up message
030044 003043 9            jmp .-1
030045 103041 9           jmp bestfl i

030046 000000 9   isuck:  0                         ;suck with discard return on
030047 120112 9           jst suck i
030050 103046 9            jmp isuck i              ;ok
030051 003012 9           jmp best                  ;flush
```

```
                    .stitl packet core
;common variables
;setup message
                    .lev var

030052 007400 V   pkclea: ldrnlf                ;leader area
030053 000000 V           0
030054        V   pkcfrh: .block 1              ;foreign host
030055        V   pkcfri: .block 1              ;foreign imp
030056        V   pkcfrl: .block 1              ;foreign link
030057 000000 V           0                     ;last word of leader
030060        V   pkcfrm: .block 1              ;foreign line no.
030061 000376 V   pkclch: fhlpkc                ;local host
030062        V   pkclci: .block 1              ;local imp
030063 000000 V           0                     ;local link
030064 000000 V   pkclct: 0                     ;local line no.
030065        V   pkcadr: .block 1              ;start core address
030066        V   pkcsiz: .block 1              ;transfer size
030067        V   pkcst1: .block 1              ;send setup flag, 0 if receiv
030070        V   pkcsr1: .block 1              ;send/rcv flag, pkcsnd=rcving
      000017      pkcstl=.-pkclea

030071        V   pkcstf: .block 1              ;our send setup flag (>0 => s
030072        V   pkcsrf: .block 1              ;our send/receive flag (0=>do
                                                ;<0 => rcv, >0 => send at tha

;imp-to-host side variables
030073        V   pkcbfr: .block 1              ;modem out buffer pointer
030074        V   pkcbff: .block 1              ;modem out start ptr, flag
030075        V   pkcbf2: .block 1              ;modem out 1st/retrans flag,
030076        V   pkcmdn: .block 1              ;modem out modem no.
030077        V   pkcldr: .block 1              ;source, source host
030100        V   pkcld1: .block 1              ;source imp
030101        V   pkclin: .block 1              ;saved local line no
030102        V   pkcln2: .block 1              ;saved (pkclin&17) - 1
030103        V   pkctmp: .block 1              ;temp ptr
030104        V   pkctmc: .block 1              ;temp counter
030105        V   pkccct: .block 1              ;length of a core message
```

```
;used by abort code
        000001      pkcsb=1                 ;busy code
        000002      pkcsbc=2                ;bad coretype code
        000004      pkcshi=4                ;generic abort - reason in hi
        000001      ctexa=1                 ;exa imp coretype
        000002      ctnmfs=2                ;nmfs macrocode coretype
        000003      ctucode=3               ;mbb microcode coretype
        000004      ctreg=4                 ;mbb registers coretype
        000005      ctdisp=5                ;mbb dispatch memory coretype
        000006      cttape=6                ;cassette coretype

;parameters
        001000      pkc13s=1000             ;13 sec in 25.6 ms ticks
        005000      pkctmo=5*pkc13s         ;65 sec receive timeout
        000001      pkcsnd=1                ;25.6 ms nominal send frequen
        004704      pkcrtt=2500.            ;retransmit time (1/4 sec)
        000200      pkcrpt=200              ;repeat bit
        000320      pkcrl=208.              ;retr. limit (52-sec/pkcrtt)
```

```
;imp-to-host side
                .lev bck
030106 023540 9 pkcmsg: cas [134000]            ;abort?
030107 003341 9          jmp pkcmdo             ;modem out
030110 103541 9          jmp [pkcabt] i         ;yes
030111 023542 9         cas [133000]            ;core?
030112 003341 9          jmp pkcmdo             ;modem out
030113 003227 9          jmp pkccor             ;yes
030114 023543 9         cas [132000]            ;no, setup?
030115 003341 9          jmp pkcmdo             ;modem out
030116 100000 9          skp                    ;yes
030117 103544 9           jmp [bestf] i         ;no, not pkc msg
030120 021211 9         jst pkclok              ;can we take it?
030121 103545 9          jmp [pkcbsy] i         ;no,send abort
030122 021177 9         jst pkcsuc              ;yes, get foreign address
030123 011054 9         sta pkcfrh
030124 021177 9         jst pkcsuc
030125 011055 9         sta pkcfri
030126 021177 9         jst pkcsuc              ;and link
030127 006032 9         ana [ldrmid]
030130 011056 9         sta pkcfrl
030131 021177 9         jst pkcsuc              ;and line no.
030132 011060 9         sta pkcfrm              ;save coretype and line no.
030133 006007 9         ana [377]               ;get line number
030134 013543 9         era [132000]            ;form first word of setup msg
030135 027060 9         ima pkcfrm              ;save, get coretype
030136 007546 9         ana [177000]
030137 011064 9         sta pkclct              ;save as local coretype
030140 121547 9         jst [pkctct] i          ;validate coretype
030141 021177 9         jst pkcsuc              ;and starting address
030142 011065 9         sta pkcadr
030143 021177 9         jst pkcsuc              ;and transfer size
030144 011066 9         sta pkcsiz
030145 021177 9         jst pkcsuc              ;and send setup flag
030146 011071 9         sta pkcstf
030147 021177 9         jst pkcsuc              ;and send/receive flag
030150 011072 9         sta pkcsrf
030151 101400 9         smi                     ;receiving?
030152 003160 9          jmp pkcms1             ;no, sending
030153 140040 9         cra                     ;setup message send setup fla
030154 011067 9         sta pkcst1
030155 004052 9         lda one 0&pkcsnd        ;setup message send/receive f
030156 011070 9         sta pkcsr1
030157 103544 9         jmp [bestf] i
```

```
030160 004052 9   pkcms1: lda [1]                ;setup message send setup fl
030161 011067 9          sta pkcst1
030162 005550 9          lda [-pkctmo]           ;setup message send/receive f
030163 011070 9          sta pkcsr1
030164 004014 9          lda [-1]                ;init send counter
030165 111551 9          sta [pkcstc] i
030166 005055 9          lda pkcfri              ;is this a virgin setup send?
030167 100040 9          sze
030170 003011 9           jmp bestf              ;no
030171 005077 9          lda pkcldr              ;yes, fill in source
030172 011054 9          sta pkcfrh
030173 005100 9          lda pkcld1
030174 011055 9          sta pkcfri
030175 025071 9          irs pkcstf              ;send setup receive
030176 003011 9          jmp bestf

030177 000000 9   pkcsuc: 0                      ;get next word
030200 120112 9          jst suck i
030201 103177 9           jmp pkcsuc i
030202 021204 9   pkceom: jst pkcclc             ;if eom, clear connection
030203 003012 9          jmp best

030204 000000 9   pkcclc: 0                      ;clear connection
030205 140040 9          cra
030206 011072 9          sta pkcsrf
030207 011071 9          sta pkcstf
030210 103204 9          jmp pkcclc i

030211 000000 9   pkclok: 0                      ;lock check
030212 005077 9          lda pkcldr              ;is the host
030213 022056 9          cas [nh]                ;fake?
                         ;defplc(/nop to disable loading other imps)
030214 100000 9   lodnop:  skp
030215 103211 9            jmp pkclok i          ;don't talk to this host
030216 013054 9          era pkcfrh              ;is it the host we were talki
030217 015055 9          add pkcfri
030220 013100 9          era pkcld1
030221 101040 9          snz
030222 003225 9           jmp pkclks             ; yes, skip return
030223 005072 9          lda pkcsrf              ;no, were we locked?
030224 101040 9          snz
030225 025211 9   pkclks:  irs pkclok            ; no, skip return
030226 103211 9          jmp pkclok i
```

```
030227 005072 9   pkccor: lda pkcsrf          ;receiving?
030230 100400 9          spl                  ;if not, ignore all cores
030231 021211 9           jst pkclok          ;yes. ok to let in?
030232 103544 9            jmp [bestf] i      ;no, flush (not abort)
030233 005105 9          lda pkccct           ;get length of message from l
030234 040474 9          lgr 4                ;convert to words (div by 16)
030235 016052 9          sub [1]              ;don't count first word
030236 140407 9          tca                  ;use to discover end of messa
030237 011105 9          sta pkccct           ;might not work if no padding
030240 021327 9          jst pkccsu           ;get addr
030241 103544 9           jmp [bestf] i       ;empty, so flush
030242 011103 9          sta pkctmp           ;save addr
030243 021306 9          jst pkccts           ;check coretype, get size
030244 005065 9          lda pkcadr           ;now check address
030245 023103 9          cas pkctmp
030246 103544 9           jmp [bestf] i       ;<expected, quit
030247 003252 9           jmp pkclo1          ;=expected, proceed
030250 025071 9          irs pkcstf           ;>expected, kick send setup f
030251 103544 9          jmp [bestf] i

030252 072053 9   pkclo1: ldx [fhppkc-nh]
030253 025104 9          irs pkctmc           ;all done?
030254 100000 9           skp
030255 003264 9            jmp pkclo2         ;yes
030256 021327 9          jst pkccsu           ;get core word
030257 003202 9           jmp pkceom
030260 073103 9          ldx pkctmp           ;address into x
030261        9   pkcwrt: .block 1            ;set to a write instruction
030262 025103 9          irs pkctmp           ;increment address
030263 003252 9          jmp pkclo1

030264 021327 9   pkclo2: jst pkccsu           ;get next addr
030265 003273 9           jmp pkclo3          ;no more in message
030266 101040 9          snz                  ;****to be removed****
030267 003273 9           jmp pkclo3          ;****u. must be fixed****
030270 011103 9          sta pkctmp           ;subsequent pieces not checke
030271 021306 9          jst pkccts           ;but coretype must be checked
030272 003252 9          jmp pkclo1           ;continue
```

```
030273 005103 9   pkclo3: lda pkctmp          ;update addr
030274 027065 9          ima pkcadr
030275 017103 9          sub pkctmp
030276 015066 9          add pkcsiz
030277 011066 9          sta pkcsiz           ;and size
030300 101400 9          smi
030301 101040 9           snz                 ;are we finished with transfe
030302 103552 9            jmp [pkcclr] i    ;yes, clear connection
030303 005550 9          lda [-pkctmo]        ;no, renew timeout
030304 011072 9          sta pkcsrf
030305 103544 9          jmp [bestf] i

030306 000000 9   pkccts: 0                    ;get size and coretype
030307 021327 9          jst pkccsu
030310 003202 9           jmp pkceom
030311 011261 9          sta pkcwrt           ;save size
030312 007546 9          ana [177000]         ;get coretype
030313 023064 9          cas pkclct           ;same as in setup?
030314 103553 9           jmp [pkcbct] i      ;no, bad coretype
030315 100000 9           skp                 ;yes
030316 103553 9            jmp [pkcbct] i     ;no, bad coretype
030317 040467 9          lgr 9.               ;turn into index
030320 010000 9          sta 0
030321 045333 9          lda pkcwtb-1 x       ;get correct instruction
030322 027261 9          ima pkcwrt           ;set and get size
030323 006033 9          ana [777]            ;mask away core type
                         ;defplc(cassette package hook)
030324 140401 9   casjam: cma                  ; -size - 1
030325 011104 9          sta pkctmc           ;set counter
030326 103306 9          jmp pkccts i

030327 000000 9   pkccsu: 0                    ;suck core word
030330 021177 9          jst pkcsuc           ;get a word
030331 025105 9          irs pkccct           ;count down number of data wo
030332 025327 9           irs pkccsu          ;skip if not last word
030333 103327 9          jmp pkccsu i

         000001     rdumbb=1
         000001     rdrmbb=1
         000001     rddmbb=1
         000001     wrumbb=1
         000001     wrrmbb=1
         000001     wrdmbb=1

;instructions to write a word; x=address
030334 110000 9   pkcwtb: sta 0 i              ;(ctexa)
030335 110000 9          sta 0 i              ;(ctnmfs)
030336 000001 9          wrumbb
030337 000001 9          wrrmbb
030340 000001 9          wrdmbb

030341 011101 9   pkcmdo: sta pkclin           ;save line info
030342 023554 9          cas [131000]         ;legal mess type?
030343 023555 9           cas [135000]
030344 003011 9            jmp bestf          ;no
030345 003011 9            jmp bestf          ;no
030346 006072 9          ana [17]             ;get modem number
030347 010000 9          sta 0
030350 100040 9          sze
```

```
030351 016072 9            sub [ch+1]
030352 101400 9           smi                    ;is it legal?
030353 003011 9            jmp bestf             ;no, flush rest of message
030354 145556 9           lda [mblock-1] ix      ;yes, is this an imp modem?
030355 101040 9           snz
030356 003011 9            jmp bestf             ;no, flush rest of message
030357 005074 9   pkcmd0: lda pkcbff             ;modem flag on?
030360 101040 9           snz
030361 003377 9            jmp pkcmd2            ;no, so go ahead
030362 005075 9           lda pkcbf2
030363 101040 9           snz                    ;yes, buffer noticed by i2m?

030364 003373 9            jmp pkcmd1            ;not yet, wait
030365 073073 9           ldx pkcbfr
030366 004014 9           lda [-1]               ;stop retransmitting
030367 050005 9           sta inch x             ;by reducing count to -1
030370 044121 9           lda itst x             ;and setting sent time older

030371 017557 9           sub [pkcrtt]           ;by the retransmit interval
030372 050121 9           sta itst x             ;so the last retr. will occur
                  pkcmd1:
030373 000401 9           .enb bck
;wait
030374 072053 9           ldx [fhppkc-nh]
030375 120113 9           jst wait i
030376 003357 9           jmp pkcmd0
```

```
030377 140040 9   pkcmd2: cra                    ;get buf
030400 001001 9           .inh fre
030401 120116 9 0         jst getfri i
030402 003373 9 0          jmp pkcmd1           ;none available, wait
030403 000401 9 0         .enb bck
030404 033073 9           stx pkcbfr
030405 004000 9           lda 0
030406 014072 9           add [data+1]         ;first word of data
030407 011103 9           sta pkctmp
030410 005560 9           lda [-odata1+1]      ;max data allowed
030411 011104 9           sta pkctmc
030412 072053 9           ldx [fhppkc-nh]
030413 120112 9   pkcmd3: jst suck i           ;get data word
030414 100000 9            skp
030415 003435 9             jmp pkcmd5         ;finished
030416 111103 9           sta pkctmp i         ;save in buf
030417 025103 9           irs pkctmp
030420 025104 9           irs pkctmc
030421 003413 9            jmp pkcmd3
030422 021041 9           jst bestfl           ;too much data, flush rest of
030423 005101 9           lda pkclin
030424 073100 9           ldx pkcld1
030425 001001 9           .inh all
                          ;defhlt(/40. oversize packet core on line=a from imp=
030426 120120 9 0         jst hltjst i
030427 073073 9 0         ldx pkcbfr           ;return buf
030430 120115 9 0         jst flushi i
030431 024224 9 0         irs nres
                  pkcmdx:
030432 000401 9 0         .enb bck
030433 072053 9           ldx [fhppkc-nh]
030434 003012 9           jmp best
```

```
030435 073073 9   pkcmd5: ldx pkcbfr          ;set pkt size
030436 005561 9           lda [odata1+hdr1]
030437 015104 9           add pkctmc
030440 100100 9           slz                 ;odd?
030441 141206 9            aoa                ;yes, make it even
030442 141240 9           icr 0&pktsiz
030443 052004 9           era wrdc x
030444 141044 9           car 0&usecnt
030445 052004 9           era wrdc x
030446 050004 9           sta wrdc x
030447 005101 9           lda pkclin          ;get repeat bit word
030450 006061 9           ana [pkcrpt]        ;is repeat bit on?
030451 100040 9           sze
030452 005562 9            lda [-pkcrl+1]     ;yes, retrans till limit reac
030453 016052 9           sub [1]             ;else use -1 for one transmit
030454 050005 9           sta inch x          ;set retransmission limit
030455 005101 9           lda pkclin          ;compute modem #
030456 006072 9           ana [17]            ;extract modem number
030457 016052 9           sub [1]
030460 011102 9           sta pkcln2          ;save modem number
030461 010000 9           sta 0
030462 145563 9           lda [mblock] ix     ;get modem block pointer
030463 011076 9           sta pkcmdn
030464 005101 9           lda pkclin          ;magic modem?
030465 141140 9           icl
030466 101100 9           sln
030467 003506 9            jmp pkcspk         ;no, send packet core
030470 004072 9           lda [data+1]        ;yes, first word of mm data
030471 015073 9   pkcmd6: add pkcbfr
030472 011074 9           sta pkcbff          ;first word to send, set flag
030473 073076 9           ldx pkcmdn
030474 050140 9           sta i2mrar x        ;poke i2m
030475 001001 9           .inh all
030476 044137 9 0         lda i2mbsy x        ;is it willing to be poked?
030477 100040 9 0         sze
030500 003432 9 0          jmp pkcmdx         ;if I2MBSY<>0, it's busy -- s
030501 073102 9 0         ldx pkcln2          ;restore modem number
030502 144142 9 0         lda i2mpci ix       ;get ptr to I2M's PCB
030503 010000 9 0         sta 0
030504 000043 9 0         gpr
030505 003432 9 0         jmp pkcmdx
```

```
                        .lev bck
030506 073076 9   pkcspk: ldx pkcmdn              ;find neighbor imp #
030507 044203 9           lda pkcnbr x
030510 073073 9           ldx pkcbfr              ;fix up buf as packet core
030511 050012 9           sta seqh x              ;sat dest
030512 044004 9           lda wrdc x              ;get size
030513 141140 9           icl 0&pktsiz
030514 050006 9           sta neth x
030515 004037 9           lda [sptyp+sptcpt+sptpkc] ;packet core
030516 050007 9           sta typh x
030517 005100 9           lda pkcld1
030520 050014 9           sta dsth x              ;source imp
030521 005077 9           lda pkcldr
030522 050013 9           sta pkth x              ;source host
030523 005101 9           lda pkclin              ;dest line
030524 050016 9           sta data x
030525 020002 9           jst chksum              ;compute checksum
030526 000000 9            %crash                 ;packet core buffer smashed
030527 140407 9           tca
030530 054010 9           add chkh x
030531 050010 9           sta chkh x              ;save checksum
030532 004031 9           lda [hedr]              ;first word of packet core da
030533 003471 9           jmp pkcmd6
```

```
;host-to-imp side (+trace fake host)


;host-to-imp side variables
                .section pg27
                .lev var
027172          V   pkcqbf: .block 1              ;packet core queue buffer ptr
027173          V   pkcptr: .block 1              ;temp ptr
027174          V   pkccnt: .block 1              ;temp counter
027175          V   pkclk1: .block 1              ;previous value of time
027176          V   pkcstc: .block 1              ;25.6 ms counter for sending


                .lev bck
                trjini:                           ;needed for INI
027177 004106 9     btre:   lda mine              ;init setup message
027200 111650 9             sta [pkclci] i
027201 121651 9             jst [pkcclc] i        ;init packet core fh
027202 120111 9     btrlop: jst doze i            ;wait one background loop
027203 073652 9             ldx [spcq]            ;anything on packet core queu
027204 001001 9             .inh fre
027205 120121 9 0           jst getqi i
027206 003322 9 0            jmp pkcflg           ;no, go check flags
027207 033172 9 0           stx pkcqbf
027210 140040 9 0           cra                   ;clear for leader
027211 066012 9 0           ima seqh x            ;for anybody specific?
027212 100040 9 0           sze
027213 003273 9 0            jmp pkcpl1           ;yes
027214 044021 9 0           lda data+3 x          ;no, save pg 1 checksum
027215 011174 9 0           sta pkccnt
027216 044005 9 0           lda inch x            ;get reload code; input line

027217 141050 9 0           cal                   ;get input line only
027220 010000 9 0           sta 0
027221 145653 9 0           lda [mblock] ix       ;get and save modem block add
027222 010001 9 0           sta ireg
027223 073172 9 0           ldx pkcqbf            ;get packet address
027224 044005 9 0           lda inch x
027225 141140 9 0           icl
027226 010002 9 0           sta jreg              ;save reload code
027227 044011 9 0           lda srch x            ;save info
027230 072001 9 0           ldx ireg              ;get modem block
027231 050203 9 0           sta pkcnbr x
027232 004002 9 0           lda jreg              ;reload code 0 (swdt)?
027233 100040 9 0           sze
027234 003266 9 0            jmp pkcply           ;no, send 81. trap instead
027235 044025 9 0           lda pcb.num x         ;set x=trunk number
027236 010000 9 0           sta 0
027237 005174 9 0           lda pkccnt            ;get pg 1 checksum
                            ;defhlt(/41. got "setup send" with pg 1 checksum=a on
027240 120120 9 0           jst hltjst i
027241 000401 9 0 pkcplz:   .enb bck
027242 073172 9             ldx pkcqbf
027243 005654 9             lda [fhlpkc]          ;assume packet core fh
027244 050013 9             sta pkth x
027245 140040 9             cra
027246 026104 9             ima pkcrld            ;reset and check reload flag

027247 100040 9             sze                   ;set?
```

```
027250 003256 9          jmp pkcp10          ;yes
027251 044007 9          lda typh x          ;no, is this a reload request
027252 006072 9          ana [17]            ; check ??kdl
```

```
027253 101040 9             snz
027254 003316 9              jmp pkcplk             ;no
027255 004106 9             lda mine               ;yes, reload from me
027256 050014 9   pkcpl0: sta dsth x
027257 005655 9             lda [132000]           ;"setup"
027260 050016 9             sta data x
027261 044005 9             lda inch x             ;our line #
027262 141206 9             aoa
027263 052022 9             era data+4 x           ;preserving his core type
027264 050022 9             sta data+4 x
027265 003277 9             jmp pkcpl2

                    .lev bck
                    .lck fre
027266 044025 9 0 pkcply: lda pcb.num x            ;set x=trunk number
027267 010000 9 0           sta 0
027270 005174 9 0           lda pkccnt             ;get pg 1 checksum
                            ;defhlt(/81. got setup send: no dump required)
027271 120120 9 0           jst hltjst i
027272 003241 9 0           jmp pkcplz
```

```
                        .lev bck
                        .lck m2i
                        pkcpl1:
027273 000401 9 1               .enb bck
027274 012106 9                 era mine                ;for us?
027275 100040 9                 sze
027276 003316 9                  jmp pkcplk             ;no
027277 105656 9   pkcpl2:       lda [pkclea] i          ;set up leader in packet
027300 050011 9                 sta srch x
027301 004000 9                 lda 0
027302 014024 9                 add [srch]              ;start of leader
027303 072021 9                 ldx [-6]                ;length of leader
027304 021476 9                 jst pkcmov              ;*send leader
027305 073172 9                 ldx pkcqbf
027306 044004 9                 lda wrdc x
027307 141140 9                 icl 0&pktsiz
027310 016055 9                 sub [hdrl]
027311 140407 9                 tca                     ;-size (2 wrds leader + data)
027312 026000 9                 ima 0                   ;data length
027313 015657 9                 add [data]              ;start of data
027314 021476 9                 jst pkcmov              ;*send data
027315 021512 9                 jst pkcpad              ;*send padding and end messag
027316 073172 9   pkcplk:       ldx pkcqbf              ;free buf
027317 001001 9                 .inh all
027320 120115 9 0               jst flushi i
027321 024224 9 0               irs nres

027322 000401 9 0 pkcflg:       .enb bck
027323 005526 9                 lda pkcast              ;send abort flag set?
027324 101040 9                 snz
027325 003334 9                  jmp pkcfl0             ;no, check send setup flag
027326 005660 9                 lda [pkcale]            ;start of abort
027327 072022 9                 ldx [-pkcabl]           ;length of abort
027330 021476 9                 jst pkcmov
027331 021512 9                 jst pkcpad              ;send padding and end message
027332 140040 9                 cra
027333 011526 9                 sta pkcast
027334 127661 9   pkcfl0:       ima [pkcstf] i          ;and check send setup flag
027335 101040 9                 snz
027336 003343 9                  jmp pkcfl1             ;check to send
027337 005656 9                 lda [pkclea]            ;start of "setup" message
027340 073662 9                 ldx [-pkcstl]           ;length of message
027341 021476 9                 jst pkcmov              ;*send setup message leader a
027342 021512 9                 jst pkcpad              ;*send padding and end messag
027343 072053 9   pkcfl1:       ldx [fhppkc-nh]
```

```
027344 004151 9           lda time
027345 027175 9           ima pkclk1
027346 013175 9           era pkclk1
027347 101040 9           snz                     ;25.6 ms tick yet?
027350 003474 9            jmp pkcdun
027351 105663 9           lda [pkcsrf] i          ;yes, what are we doing?
027352 022006 9           cas [0]
027353 003365 9            jmp pkcf12             ;>0, sending
027354 003474 9            jmp pkcdun
027355 125663 9           irs [pkcsrf] i          ;<0, receiving, bump timeout

027356 100000 9            skp                    ;still going
027357 003474 9             jmp pkcdun
027360 007664 9           ana [-(3*pkc13s+1)]     ;time to send setup send?
027361 013664 9           era [-(3*pkc13s+1)]
027362 101040 9           snz
027363 125661 9            irs [pkcstf] i         ;yes, kick flag
027364 003474 9           jmp pkcdun

027365 025176 9   pkcf12: irs pkcstc              ;time to send core?
027366 003474 9            jmp pkcdun
027367 140407 9           tca                     ;yes, reset counter
027370 011176 9           sta pkcstc
027371 105665 9           lda [pkcsiz] i          ;check size first
027372 100040 9           sze
027373 003376 9            jmp pkcsd0
027374 111663 9           sta [pkcsrf] i          ;zero size -- close transacti
027375 003474 9           jmp pkcdun

027376 005656 9   pkcsd0: lda [pkclea]            ;start of leader
027377 072021 9           ldx [-6]                ;length of leader
027400 021476 9           jst pkcmov              ;*send leader
027401 105666 9           lda [pkcfrm] i          ;a 'core' message
027402 012063 9           era [133000?132000]
027403 120110 9           jst jam i
027404 105666 9           lda [pkcfrm] i
027405 141050 9           cal
027406 100040 9           sze                     ;going to live or dead imp?
027407 005667 9            lda [763?72]           ;dead, use single packet
027410 013670 9           era [763]               ;live, use multi-packet
027411 140407 9           tca
027412 011174 9           sta pkccnt              ;-size
027413 115665 9           add [pkcsiz] i
027414 127665 9           ima [pkcsiz] i          ;update transfer size
027415 100400 9           spl                     ;was old size >32k?
027416 003423 9            jmp pkcsd1             ;yes, then surely not done
027417 105665 9            lda [pkcsiz] i         ;else, has new size
027420 101400 9           smi                     ;gone from pos. to neg?
027421 101040 9            snz                    ;or to 0?
027422 003466 9             jmp pkcsdf            ;yes, then all done
027423 105671 9   pkcsd1: lda [pkcadr] i          ;save current address
027424 011173 9           sta pkcptr              ;in pointer
027425 017174 9           sub pkccnt              ;form new address
027426 111671 9           sta [pkcadr] i          ;and save
027427 005173 9           lda pkcptr              ;*send address
027430 120110 9           jst jam i
027431 005173 9           lda pkcptr
027432 101040 9           snz
027433 120110 9            jst jam i              ;if addr=0, double it
```

lda pkccnt

```
027435 140407 9           tca                     ;*send size
027436 113672 9           era [pkclct] i          ;set coretype too
027437 120110 9           jst jam i
027440 105672 9           lda [pkclct] i
027441 040467 9           lgr 9.
027442 010000 9           sta 0
027443 045460 9           lda pkcrtb-1 x          ;get proper read instr.
027444 011446 9           sta pkcrd               ;for this coretype
027445 073173 9   pkcsd2: ldx pkcptr              ;*send core words: set x to a
027446        9   pkcrd:  .block 1                ;do read i/o from right coret
027447 072053 9           ldx [fhppkc-nh]
027450 120110 9           jst jam i
027451 025173 9           irs pkcptr
027452 025174 9           irs pkccnt
027453 003445 9            jmp pkcsd2
027454 140040 9           cra
027455 120110 9           jst jam i               ;*send next addr=0
027456 021512 9           jst pkcpad              ;*send padding and end messag
027457 072053 9           ldx [fhppkc-nh]
027460 003474 9           jmp pkcdun

;instructions to read a word
027461 104000 9   pkcrtb: lda 0 i                 ;(ctexa)
027462 104000 9           lda 0 i                 ;(ctnmfs)
027463 000001 9           rdumbb                  ;x=addr (doubled) byte into a
027464 000001 9           rdrmbb                  ;x=addr (doubled) byte into a
027465 000001 9           rddmbb                  ;x=address word into a


027466 017174 9   pkcsdf: sub pkccnt              ;original size left
027467 140407 9           tca                     ;-size
027470 011174 9           sta pkccnt              ;use it instead of max
027471 005673 9           lda [-(pkc13s-1)]       ;set to receive
027472 111663 9           sta [pkcsrf] i          ;without setup sends for a wh
027473 003423 9           jmp pkcsd1

; fake host 2, H2I side, exits through here
027474 120147 9   pkcdun: jst trace.fh2 i         ;call trace, if loaded
027475 003202 9           jmp btrlop              ;then resume fh2 h2i loop
```

```
027476 000000 9   pkcmov: 0                    ;start in a, -size in x
027477 011510 9          sta pkcmv1
027500 033511 9          stx pkcmv2
027501 072053 9          ldx [fhppkc-nh]
027502 105510 9          lda pkcmv1 i
027503 120110 9          jst jam i
027504 025510 9          irs pkcmv1
027505 025511 9          irs pkcmv2
027506 003502 9           jmp .-4
027507 103476 9          jmp pkcmov i

027510         9   pkcmv1: .block 1             ;temps
027511         9   pkcmv2: .block 1
```

```
                        .lev bck
027512 000000 9         pkcpad: 0                       ;send padding and end message
027513 073674 9                 ldx [100002] 0&fhppkc
027514 004071 9                 lda [100000]
027515 120110 9                 jst jam i
027516 103512 9                 jmp pkcpad i

                        .lev var
027517 007400 V         pkcale: ldrnlf                  ;abort message built here
027520 000000 V                 0
027521        V         pkcafh: .block 1
027522        V         pkcafi: .block 1
027523        V         pkcafl: .block 1
027524 000000 V                 0
027525        V         pkcafm: .block 1                ;abort type + line no.
027526        V         pkcast: .block 1                ;abort reason (status bits)
       000010           pkcabl=.-pkcale

                        .lev bck
027527 121675 9         pkcabt: jst [pkclok] i          ;asked to abort our connectio
027530 103676 9                  jmp [bestf] i          ;no, just ignore
027531 001001 9                 .inh all
                                ;defhlt(/43. packet core abort recieved)
027532 120120 9 0               jst hltjst i
027533 000401 9 0               .enb bck
027534 121651 9         pkcclr: jst [pkcclc] i          ;clear connection
027535 103676 9                 jmp [bestf] i

027536 004052 9         pkcbsy: lda [pkcsb]             ;busy status bit
027537 021555 9                 jst pkcsab              ;send abort
027540 103676 9                 jmp [bestf] i           ;now flush packet


027541 000000 9         pkctct: 0                       ;test coretype
027542 040467 9                 lgr 9.                  ;isolate coretype
027543 022053 9                 cas [ctnmfs]            ;range check, ctnmfs->cttape
027544 101000 9                  nop
027545 022031 9                  cas [cttape]
027546 003551 9                   jmp pkcbct            ;<ctnmfs or >cttape, illegal

                                ;defplc(cassette package hook)
027547 003551 9         castyp:   jmp pkcbct            ;=cttape, only legal for pack
027550 103541 9                 jmp pkctct i            ;ok, exit
```

```
027551 004053 9   pkcbct: lda [pkcsbc]          ;bad coretype status bit
027552 021555 9           jst pkcsab            ;send abort
027553 121651 9           jst [pkcclc] i        ;clear connection
027554 103676 9           jmp [bestf] i         ;now flush packet

027555 000000 9   pkcsab: 0                     ;send abort
027556 027526 9           ima pkcast            ;one already in progress?
027557 100040 9           sze
027560 003577 9            jmp pkccab           ;yes
027561 005526 9           lda pkcast
027562 001001 9           .inh all
                          ;defhlt(/44. sending packet core abort)
027563 120120 9 0         jst hltjst i
027564 000401 9 0         .enb bck
027565 100100 9           slz
027566 003601 9            jmp pkcsa1           ;send abort for busy (code 1)
027567 072020 9           ldx [-5]              ;send abort for bad coretype
027570 145677 9   pkccpa: lda [pkcfrh+5] ix     ;copy fields from setup
027571 051526 9           sta pkcafh+5 x        ;to abort
027572 024000 9           irs 0
027573 003570 9            jmp pkccpa
027574 013700 9           era [132000?134000]   ;change to abort code
027575 011525 9           sta pkcafm
027576 103555 9           jmp pkcsab i

027577 011526 9   pkccab: sta pkcast           ;can't send abort, one in pro
027600 103555 9           jmp pkcsab i          ;just return

;send abort for busy
027601 021615 9   pkcsa1: jst pkcasu
027602 011521 9           sta pkcafh            ;build abort
027603 021615 9           jst pkcasu            ;from received setup
027604 011522 9           sta pkcafi
027605 021615 9           jst pkcasu            ;get link
027606 006032 9           ana [ldrmid]
027607 011523 9           sta pkcafl
027610 021615 9           jst pkcasu            ;ignore length
027611 006007 9           ana [377]
027612 013701 9           era [134000]          ;set abort type
027613 011525 9           sta pkcafm
027614 103555 9           jmp pkcsab i          ;exit

027615 000000 9   pkcasu: 0                     ;get next word of message
027616 120112 9           jst suck i
027617 103615 9            jmp pkcasu i
027620 140040 9           cra                   ;if eom, cancel abort
027621 011526 9           sta pkcast
027622 103702 9           jmp [best] i

;Keep in order for BKST

027623 026146 9   cawl:   givlst                ;unused
027624 026146 9           givlst                ;cumstats
027625 026054 9           genm                  ;message generator
027626 026201 9           diag                  ;diagnostic sender
027627 026245 9           trbl                  ;trouble reports
```

```
                .stitl UTILITIES
                .INCLUDE utl.m4
;utilities

;background checksummer no-op'ed
```

```
                    .stitl general interrupt, buffer routines
                    .section pg31
;doesn't change x
                    .lev fre
031000 000000 0     gfree:  0                       ;get a free buffer
031001 105671 0             lda [sfq] i
031002 101040 0             snz
031003 103000 0              jmp gfree i            ;no buffers, no skip
031004 011021 0             sta gfreet             ;leave a clue
031005 105021 0             lda gfreet i           ;remove from queue
031006 111671 0             sta [sfq] i
031007 100040 0             sze                    ;last one?
031010 003013 0              jmp gfree1            ;no
031011 005671 0             lda [sfq]              ;yes, update end ptr
031012 111672 0             sta [efq] i            ;to point to start
031013 140040 0     gfree1: cra                    ;clear chain ptr
031014 111021 0             sta gfreet i 0&ptrc
031015 005021 0             lda gfreet
031016 024222 0             irs nfs                ;keep count
031017 025000 0             irs gfree              ;skip=success
031020 103000 0             jmp gfree i

                    .lev var
031021        V     gfreet: .block 1               ;last buffer used

                    .lev (h2i,tsk)
;subr to rangecheck imp number
031022 000000 4     impchk: 0
031023 022006 4             cas zero
031024 023037 4              cas mxnimp            ;>0
031025 103022 4               jmp impchk i         ;=0 or >nimp
031026 101400 4               smi                  ;<0 or =nimp
031027 025022 4                irs impchk          ;>0 and <,=nimp
031030 103022 4             jmp impchk i

                    .lev all
                    .lck all
;Subr get change X from host number to host config block pointer
031031 000000 0     hcnf:   0                       ;enter with X=host number
031032 004000 0             lda 0                  ;get it
031033 041475 0             lgl 3. 0&cf.size       ;times size of each config bl
031034 015673 0             add [confighost]       ;plus where host #0 is yeilds
031035 010000 0             sta 0
031036 103031 0             jmp hcnf i             ;return

                    .lev con
031037 000200 C     mxnimp: nnodes
```

```
                    .lev all
031040 000000 0     .hltjs: 0                       ;general hltncc routine
031041 120126 0             jst hltncc i
031042 103040 0             jmp .hltjs i

                    .lev t.o
;Adjust queue counters for queue area

031043 000000 5     juqc:   0
031044 072017 5             ldx [counta-counts]     ;get negative size
031045 001001 5             .inh all
031046 145674 5 0   qc1:    lda [ccounta + (counts-counta)] ix
031047 157675 5 0           sub [ccounts + (counts-counta)] ix
031050 100400 5 0           spl
031051 000000 5 0            %crash                 ;JUQC: queue counter went nega
031052 151674 5 0   qc2:    sta [ccounta + (counts-counta)] ix
031053 140040 5 0           cra
031054 151675 5 0           sta [ccounts + (counts-counta)] ix
031055 024000 5 0           irs 0
031056 003046 5 0            jmp qc1
031057 000401 5 0           .enb t.o
031060 103043 5             jmp juqc i

;subroutine to get from a queue
                    .lev (i2m,i2h,t.o,tsk,bck)
                    .lck fre
031061 000000 2 0   getq:   0
031062 044000 2 0           lda 0 x
031063 101040 2 0           snz
031064 103061 2 0            jmp getq i             ;nothing on queue
031065 044000 2 0           lda ptrc x
031066 010001 2 0           sta ireg
031067 104001 2 0           lda ireg i
031070 100040 2 0           sze                    ;is this the only thing on q?
031071 003075 2 0            jmp getq1             ;no
031072 004000 2 0           lda 0                  ;yes, end points to start
031073 050102 2 0           sta queuee-queueb x
031074 140040 2 0           cra
031075 066000 2 0 getq1:    ima 0 x                ;remove from queue
031076 010000 2 0           sta 0
031077 140040 2 0           cra                    ;***important for back4
031100 050000 2 0           sta ptrc x 0&1ch       ;clear chain pointer
031101 025061 2 0           irs getq
031102 103061 2 0           jmp getq i

;subroutine to get from a modem queue
                    .lev (i2m,i2h,t.o,tsk,bck)
                    .lck fre
031103 000000 2 0 mgetq:    0
031104 044000 2 0           lda 0 x
031105 101040 2 0           snz
031106 103103 2 0            jmp mgetq i            ;nothing on queue
031107 044000 2 0           lda ptrc x
031110 010001 2 0           sta ireg
031111 104001 2 0           lda ireg i
031112 100040 2 0           sze                    ;is this the only thing on q?
031113 003117 2 0            jmp mgetq1            ;no
031114 004000 2 0           lda 0                  ;yes, end points to start
031115 050002 2 0           sta mqueuel x
```

```
031116 140040 2 0          cra
031117 066000 2 0 mgetq1:   ima 0 x              ;remove from queue
031120 010000 2 0          sta 0
031121 140040 2 0          cra                   ;***important for back4
031122 050000 2 0          sta ptrc x 0&1ch      ;clear chain pointer
031123 025103 2 0          irs mgetq
031124 103103 2 0          jmp mgetq i
```

```
                .lev fre
; *** flush routine ***
;call with interrupts locked
;doesn't change x
031125 000000 0   flush:  0
031126 044004 0           lda wrdc x            ;one less use
031127 016052 0           sub one
031130 066004 0           ima wrdc x
031131 006013 0           ana [usecnt]
031132 022052 0           cas one               ;is count now zero?
031133 103125 0            jmp flush i          ;>0, still in use
031134 100000 0            skp                  ;=0
031135 000000 0             %crash              ;buffer flushed too many tim
031136 004000 0           lda 0                 ;to be stolen?
031137 113676 0           era [bufflu] i
031140 101040 0           snz
031141 003152 0            jmp flush1           ;yes
031142 105672 0           lda [efq] i
031143 010001 0           sta ireg
031144 132001 0           stx ireg i            ;no, add to end of queue
031145 133672 0           stx [efq] i
031146 024216 0           irs nfa               ; free packet count
031147 005125 0           lda flush             ;write caller's address into
031150 050120 0           sta itst-1 x          ;right before ITST for debugg
031151 103125 0           jmp flush i

031152 011433 0   flush1: sta bufflu            ;clear flag
031153 004157 0           lda maxr              ;and decrement maxr
031154 016052 0           sub [1]
031155 010157 0           sta maxr
031156 103125 0           jmp flush i
```

```
                    .lev fre
;get free buffer for all but modem in, skip return on success
;expects minimum remaining bufs required in a
031157 000000 0   getfre: 0                         ;must work in exa mode for si
031160 021176 0           jst maxchk                ;enough buffers ?
031161 103157 0            jmp getfre i             ;no
031162 021000 0           jst gfree                 ;yes, try to get one
031163 103157 0            jmp getfre i             ;none available after all
031164 010000 0           sta 0                     ;got one
031165 024220 0           irs nrea                  ;count in reassembly
031166 005677 0           lda [(hdr1+2)<<8.+1] 0&pktsiz&usecnt&trcpkt&trcrrt
031167 050004 0           sta wrdc x                ;init size=header+2 data word
031170 140040 0           cra                       ;init neth
031171 050006 0           sta neth x
031172 004106 0           lda mine
031173 050011 0           sta srch x                ;source imp
031174 025157 0           irs getfre                ;success return
031175 103157 0           jmp getfre i

                    .lev fre
;check maxr, minf occupancy, expects minimum remaining
;bufs required in a, skips if there is enough room
031176 000000 0   maxchk: 0
031177 011220 0           sta maxcht
031200 014221 0           add nala
031201 016225 0           sub nals
031202 014220 0           add nrea
031203 016224 0           sub nres
031204 016157 0           sub maxr
031205 101400 0           smi                       ;a+r+rem+1>maxr?
031206 103176 0            jmp maxchk i             ;yes, don't take
031207 004216 0           lda nfa
031210 016222 0           sub nfs
031211 016221 0           sub nala
031212 014225 0           add nals
031213 016027 0           sub [minf]
031214 017220 0           sub maxcht
031215 101400 0           smi                       ;minf+rem+a>free?
031216 025176 0            irs maxchk               ;no, ok to take
031217 103176 0           jmp maxchk i              ;yes, don't take

                    .lev var
031220         V  maxcht: .block 1                  ;minimum remainder
```

```
;routine to do checksum of packet, skips if length ok
;the return address is saved at each interrupt level
                .lev bck
;call with x=buf ptr
031221 044004 9   chksm2: lda wrdc x
031222 141140 9           icl 0&pktsiz          ;get pkt size (header+data)
031223 140407 9           tca
031224 023700 9           cas [-maxpl]
031225 003230 9            jmp chksm1           ;< max length
031226 003230 9            jmp chksm1           ;= max length
031227 102002 9   chksmr: jmp chksum i          ;bad length, don't skip

031230 024002 9   chksm1: irs chksum            ;skip return
031231 010001 9           sta ireg              ;save -size
031232 015701 9           add [addbot]          ;start point in add chain
031233 026001 9           ima ireg
031234 102001 9           jmp ireg i            ;jump into add chan
```

```
                    .stitl host throughput counting
                    .lev h2i
;8-way breakdown of host throughput
031235 000000 4   htpmt:  0                        ;count messages to net
031236 001001 4            .inh all
031237 033312 4 0          stx htphst
03124C 072015 4 0          ldx minus2
031241 021267 4 0          jst htpsub
031242 000401 4 0          .enb h2i
031243 103235 4            jmp htpmt i

031244 000000 4   htppt:  0                        ;packets to net
031245 121254 4            jst htm.htmhok i        ;htm pkg called from here
031246 001001 4            .inh all
031247 033312 4 0          stx htphst
03125C 072006 4 0          ldx zero
031251 021267 4 0          jst htpsub
031252 000401 4 0          .enb h2i
031253 103244 4            jmp htppt i

                    htm.htmhok:
031254 010333 4            jsth2i

                    .lev i2h
031255 000000 3   htpmf:  0                        ;messages from net
031256 033312 3            stx htphst
031257 072014 3            ldx minus1
03126C 021267 3            jst htpsub
031261 103255 3            jmp htpmf i

031262 000000 3   htppf:  0                        ;packets from net
031263 033312 3            stx htphst
031264 072052 3            ldx one
031265 021267 3            jst htpsub
031266 103262 3            jmp htppf i
```

```
                        .lev (i2h,h2i)
031267 000000 3   htpsub: 0
031270 012106 3          era mine
031271 100040 3          sze                    ;inter- or intra-node?
031272 004017 3           lda minus4
031273 015702 3          add [ntrt1+2]
031274 011313 3          sta htpsp1
031275 145313 3          lda htpsp1 ix
031276 016056 3          sub [nh]
031277 011313 3          sta htpsp1
031300 005312 3          lda htphst
031301 016056 3          sub [nh]
031302 073312 3          ldx htphst
031303 101400 3          smi                    ;fake host?
031304 103267 3           jmp htpsub i          ;yes, don't count
031305 145313 3          lda htpsp1 ix
031306 101400 3          smi                    ;overflowed?
031307 141206 3           aoa                   ;no, count
031310 151313 3          sta htpsp1 ix
031311 103267 3          jmp htpsub i

                        .lev var
031312         V   htphst: .block 1
031313         V   htpsp1: .block 1
```

```
                        .stitl buffer stealing and returning
                        .lev t.o
                        bufchk:
031314 001001 5                 .inh all
031315 005433 5 0               lda bufflu              ;are we waiting for flush
031316 100040 5 0               sze                     ;  to steal something?
031317 003357 5 0                jmp bufchx             ;yes, just exit and wait
031320 005440 5 0               lda bufbsy              ;no, anything old to steal?
031321 100040 5 0               sze
031322 003346 5 0                jmp bufch1             ;yes, go do it
031323 005434 5 0               lda bufrt1              ;no, anything to return?
031324 100040 5 0               sze
031325 003372 5 0                jmp bufch2             ;yes, go do it
031326 005436 5 0               lda bufrq1              ;no, anything new to steal?
031327 101040 5 0               snz
031330 003357 5 0                jmp bufchx             ;no, just exit
031331 073703 5 0               ldx [-bufbcl]           ;find a free bufbch slot
031332 045551 5 0 bufch8:       lda bufbch+bufbcl x
031333 101040 5 0               snz
031334 003341 5 0                jmp bufch7             ;got one, x=index into bufbch
031335 024000 5 0               irs 0
031336 024000 5 0               irs 0                   ;not yet: all full?
031337 003332 5 0                jmp bufch8             ;no, keep looking
031340 003357 5 0               jmp bufchx              ;yes, just exit

031341 033442 5 0 bufch7:       stx bufidx              ;save index: use when stealin
031342 005436 5 0               lda bufrq1
031343 011440 5 0               sta bufbsy
031344 005437 5 0               lda bufrq2
031345 011441 5 0               sta bufbs2
031346 005440 5 0 bufch1:       lda bufbsy              ;check for end of range
031347 023441 5 0               cas bufbs2
031350 000000 5 0                %crash                 ;BUFCHK: past it: crash
031351 003361 5 0                jmp bufch4             ;yes, clear bufbsy & set buft
031352 021443 5 0               jst bchsch              ;check if buffer is stolen
031353 003357 5 0                jmp bufchx             ;stop! already stolen!
031354 011433 5 0               sta bufflu              ;ask flush to steal it
031355 015704 5 0               add [bufl]
031356 011440 5 0 bufch5:       sta bufbsy              ;bufbsy=buffer to steal next
                        bufchx:
031357 000401 5 0               .enb t.o
031360 102124 5                 jmp toret i

;
; stealing done, finish up bookkeeping
;       1) copy bufrq1, bufrq2 to bufbch
;       2) clear bufrq1, bufrq2 and bufbsy
;
031361 073442 5   bufch4:       ldx bufidx
031362 140040 5                 cra
031363 027437 5                 ima bufrq2
031364 051552 5                 sta bufbch+bufbcl+1 x
031365 140040 5                 cra
031366 027436 5                 ima bufrq1
031367 051551 5                 sta bufbch+bufbcl x
031370 140040 5                 cra
031371 003356 5                 jmp bufch5              ;clear bufbsy, and exit.
```